

CASEE 2017

2017 Eight Argentine Symposium and Conference on
Embedded Systems (CASE)

■ Regular papers.



www.sase.com.ar

University of Buenos Aires, Argentina

IEEE Catalog Number CFP1746V-PRT | ISBN 978-987-46297-1-5

2017 Eight Argentine Symposium and Conference on Embedded Systems (CASE)

Regular Papers

August 9th-11th, 2017
School of Engineering - University of Buenos Aires
Buenos Aires, Argentina



2017 Eight Argentine Symposium and Conference on Embedded Systems CASE : regular papers / José Lipovetzky, Maximiliano Antonelli, Diego Brengi, Luciana De Micco, Mariano García Inza, Ariel Lutenberg ... [et al.]. - 1a ed ilustrada. - Ciudad Autónoma de Buenos Aires : ACSE - Asociación Civil para la investigación, Promoción y Desarrollo de Sistemas Eléctricos Embebidos, 2017.
100 p. ; 29 x 20 cm.

ISBN 978-987-46297-1-5

1. Ingeniería. 2. Actas de Congresos. 3. Ingeniería Electrónica. I. Lipovetzky, José
CDD 621.39

ISBN Date: 9/07/2017

2017 Eight Argentine Symposium and Conference on Embedded Systems (CASE)

Regular Papers

Printed Book ISBN 978-987-46297-1-5
E-Book ISBN 978-987-46297-2-2

Printed Book IEEE CATALOG CFP1746V-PRT
IEEE XPLORE COMPLIANT CFP1746V-ART

Editors:

Antonelli, Maximiliano	UNMDP/ICyTE/
Brengi, Diego	INTI/UNLaM/FIUBA
De Micco, Luciana	UNMDP/ICyTE/CONICET
García Inza, Mariano	FIUBA
Lipovetzky, José	IB/CNEA/CONICET
Lutenberg, Ariel	FIUBA/CONICET/ACSE

Reprint Permission is permitted with credit to the source.
All rights reserved.
Copyright 2017 by IEEE and ACSE.

Preface

The development of Embedded Systems is crucial for the development of industry, technology and science; and it is an area which has grown in the past years in Argentina and South America.

The SASE and CASE are intended to promote the development of embedded systems in the country and the region with the following activities:

- Presentation of scientific and technological works at the conference.
- Hands-on Workshops.
- Technical Lectures (Tutorials).
- Plenary sessions.
- Contest for students projects
- A program to assign electronic equipment to Universities.
- Travel and lodging grants for graduate and postgraduate students, professors and researchers of Argentina.

The objectives of the SASE are:

- To allow a more fluid exchange between academia and industry.
- To promote the exchange between researchers and students from different universities from Argentina and other countries.
- To allow the diffusion of scientific and technological development done in the region and worldwide.
- To encourage students from the country to get interested in the development of Embedded Systems.
- To coordinate and promote the actualization of curriculum contents related to Embedded Systems in undergraduate and graduate programs at the universities of Argentina.

This year, topic areas included at CASE are: Architecture of Microprocessors, ASICs, DSPs, FPGA and HDLs, Implementation of Embedded Systems, Embedded Systems Linux, Communications and Protocols, Embedded Software, Robotics, RTOS and Bioengineering. Scientific works were accepted in three categories, Regular Papers, Technical Forum, and Poster. This book contains the works accepted as Regular Papers. After an exhaustive peer review process only fourteen papers were accepted in this distinguished category and published here. Some of the other ones were moved to the categories of technological forum or poster.

We hope you enjoy the following papers, and the conference.

CASE 2017 Organizing Committee

Sponsors

Diamond Sponsors

- Cika Electrónica S.R.L.
- Electrocomponentes S.A.
- Synopsys

Platinum Sponsors

- Semak

Gold Sponsors

- CADIPEL
- Ernesto Mayer S.A.
- Probattery
- Emtech
- Asembli S.A.
- Dai Ichi Circuitos
- ClariPhy Argentina
- Telit
- Vicda Argentina

Silver Sponsors

- Digi
- Ubidots
- L&R Ingeniería

Institutions which support SASE

Organizing Institution

- ACSE (Asociación Civil para la Investigación, Promoción y Desarrollo de los Sistemas Electrónicos Embebidos)

Co-organizing Institutions

- RUSE (Red Universitaria de Sistemas Embebidos)

Sponsoring

- ANPCyT (Agencia Nacional de Promoción Científica y Tecnológica)
- CADIEEL (Cámara Argentina de Industrias Electrónicas, Electromecánicas y Luminotécnicas)
- CAPER (Cámara Argentina de Proveedores y Fabricantes de Equipos de Radiodifusión)
- CONICET (Consejo Nacional de Investigaciones Científicas y Técnicas)
- Fundación Sadosky (Investigación y Desarrollo en TIC)
- IEEE Argentina (Institute of Electrical and Electronics Engineers)
- IEEE CASS (Circuits and Systems Society)
- ISOC (Internet Society)

Supporting

- AADECA (Asociación Argentina de Control Automático)
- ADIMRA (Asociación de Industriales Metalúrgicos de la República Argentina)
- CAI (Centro Argentino de Ingenieros)
- CAME (Confederación Argentina de la Mediana Empresa)
- CAMOCA (Cámara Argentina de Máquinas de Oficina, Comerciales y Afines)
- CASEL (Cámara Argentina de Seguridad Electrónica)
- CEIL (Cámara de Empresas Informáticas del Litoral)
- CESSI (Cámara de Empresas de Software y Servicios Informáticos)
- CIIECA (Cámara de Industrias Informáticas, Electrónicas y de Comunicaciones del Centro de Argentina)
- CNEA (Comisión Nacional de Energía Atómica)
- CONFEDI (Consejo Federal de Decanos de Ingeniería)
- INTI (Instituto Nacional de Tecnología Industrial)
- MinCyT (Ministerio de Ciencia, Tecnología e Innovación Productiva)
- Red UIE (Red Universitaria de Ingeniería en Electrónica)
- RUNIC (Red Universitaria de Ingeniería en Computación)

Supporting Universities

- UBA: Universidad de Buenos Aires
- UNAJ: Universidad Nacional Arturo Jauretche
- UNC: Universidad Nacional de Córdoba
- UNCA: Universidad Nacional de Catamarca
- UNCOMA: Universidad Nacional del Comahue
- UNCUYO: Universidad Nacional de Cuyo
- UNER: Universidad Nacional de Entre Ríos
- UNICEN: Universidad Nacional del Centro de la Provincia de Buenos Aires
- UNLAM: Universidad Nacional de La Matanza
- UNLP: Universidad Nacional de La Plata
- UNLu: Universidad Nacional de Luján
- UNNE: Universidad Nacional del Nordeste
- UNNOBA: Universidad Nacional del Noroeste de la Provincia de Buenos Aires
- UNM: Universidad Nacional de Misiones
- UNMDP: Universidad Nacional de Mar del Plata
- UNPA: Universidad Nacional de la Patagonia Austral
- UNPSJB: Universidad Nacional de la Patagonia San Juan Bosco
- UNQ: Universidad Nacional de Quilmes
- UNR: Universidad Nacional de Rosario
- UNRC: Universidad Nacional de Rio Cuarto
- UNS: Universidad Nacional del Sur
- UNSA: Universidad Nacional de Salta
- UNSJ: Universidad Nacional de San Juan
- UNSL: Universidad Nacional de San Luis
- UNSM: Universidad Nacional de San Martín
- UNT: Universidad Nacional de Tucumán
- UNTF: Universidad Nacional de Tres de Febrero
- UTN-FRA: Universidad Tecnológica Nacional - Facultad Regional Avellaneda
- UTN-FRBA: Universidad Tecnológica Nacional - Facultad Regional Buenos Aires
- UTN-FRBB: Universidad Tecnológica Nacional - Facultad Regional Bahía Blanca
- UTN-FRD: Universidad Tecnológica Nacional - Facultad Regional Delta
- UTN-FRH: Universidad Tecnológica Nacional - Facultad Regional Haedo
- UTN-FRLR: Universidad Tecnológica Nacional - Facultad Regional La Rioja
- UTN-FRM: Universidad Tecnológica Nacional - Facultad Regional Mendoza
- UTN-FRN: Universidad Tecnológica Nacional - Facultad Regional Neuquén
- UTN-FRP: Universidad Tecnológica Nacional - Facultad Regional Paraná
- UTN-FRRE: Universidad Tecnológica Nacional - Facultad Regional Resistencia

- UTN-FRRG: Universidad Tecnológica Nacional – Facultad Regional Rio Grande
- UTN-FRSF: Universidad Tecnológica Nacional – Facultad Regional San Francisco
- UTN-FRSN: Universidad Tecnológica Nacional – Facultad Regional San Nicolás
- UTN-FRVT: Universidad Tecnológica Nacional – Facultad Regional Venado Tuerto
- UTN-FRVM: Universidad Tecnológica Nacional – Facultad Regional Villa Mercedes
- UTN-FRT: Universidad Tecnológica Nacional – Facultad Regional Tucumán
- UADE: Universidad Argentina de la Empresa
- CAECE: Universidad CAECE
- FASTA: Fraternidad de Agrupaciones Santo Tomás de Aquino
- ITBA: Instituto Tecnológico de Buenos Aires
- IUA: Instituto Universitario Aeronáutico
- UBP: Universidad Blas Pascal
- UCC: Universidad Católica de Córdoba
- UCU: Universidad Católica de Uruguay
- UCSE: Universidad Católica de Santiago del Estero
- UM: Universidad de Mendoza
- UREP: Universidad de la República del Uruguay

Technical Program Committee 2017

SASE General Chair

- Dr. Ariel Lutenberg (FIUBA/CONICET/ACSE)

CASE Technical Program Chair:

- Msc. Diego Brengi (INTI/UNLaM/FIUBA, IEEE-CASS Member)

CASE Publication Chairs:

- Dr. Luciana De Micco (UNMDP/ICyTE/CONICET, IEEE Senior Member)
- Dr. José Lipovetzky (IB/CNEA/CONICET, IEEE-CASS Member)

CASE Evaluation Committee:

- Eng. Maximiliano Antonelli (UNMDP/IcyTE)
- Dr. Mariano García Inza (FIUBA)

CASE Topic Chairs:

- Bioengineering: Eng. Juan Manuel Reta (UNER)
- DSPs: Dra. María Liz Crespo (ICTP)
- Embedded Linux: Ing. Alejandro Furfaro (UTN-FRBA)
- Embedded Software: Dr. Ricardo Medel (Ascentio Technologies)
- FPGAs, HDLs and ASIC: Eng. Salvador Tropea (INTI/UTN-FRBA)
- Implementation of Embedded Systems: Dr. Gustavo Sutter (UAM) and Msc. Cristian Sisterna (UNSJ)
- Processor's architecture: Eng. Alejandro Furfaro (UTN-FRBA)
- Protocols and communications: Eng. Gustavo Mercado (UTN-FRM)
- Robotics: Claudio Verrastro (CNEA)
- RTOS: Dr. Ricardo Cayssials (UNS)
- Wireless communications: Dr. Leonardo Rey Vega (FIUBA)

Reviewers

Aguirre, Fernando Leonel
Alessandrini, Gustavo
Alvarez, Nicolás
Antonelli, Maximiliano
Arias, Ricardo
Bos, Patricio
Brenzi, Diego
Bulacio, Matías Fernando
Burgos, Enrique Sergio
Calarco, Nicolás
Carbonetto, Sebastián
Carrá, Martín
Cayssials, Ricardo
Comas, Edgardo
Crespo, Maria Liz
De Jesús, Sergio
De Micco, Luciana
Escudero, Gustavo
Ferraó, Hilda Noemí
Ferreira, Pablo Alejandro
Filomena, Eduardo
Fraire, Juan Andrés
Furfaro, Alejandro
Gabian, Gabriel
Gak, Joel
Galleguillo, Juan
Gavinowich, Gabriel
Golmar, Federico
Gómez, Pablo
Grimblatt, Victor
Hernandez Tabares, Lorenzo
Hidalgo, Roberto
Larosa, Facundo Santiago

Leiva, Lucas
Lipovetzky, José
Lutenberg, Ariel
Martos, Pedro
Medel, Ricardo
Melo, Rodrigo Alejandro
Mercado, Gustavo
Miguez, Matías
Monte, Gustavo
Montilla Cabrera, Juan Vicente
Ovilla, Brisbane
Pantelides, Carlos
Pazos, Sebastián Matías
Perez Paina, Gonzalo
Pérez, Martín
Pérez, Santiago
Permingeat, Alejandro
Petrashin, Pablo Antonio
Pucheta, Julián
Puga, Gerardo Ludovico
Reta, Juan Manuel
Rey Vega, Leonardo
Sambuco Salomone, Lucas
Sanca, Gabriel Andrés
Sistema, Cristian
Sutter, Gustavo
Taffernaberry, Carlos
Tropea, Salvador
Verrastro, Claudio
Zabaleta, Omar Gustavo
Zacchigna, Federico G.
Zaradnik, Ignacio
Zecchin, Danilo

Table of Contents

Preface.	iii
Sponsors.	v
Technical Program Committee.	xi
Design and verification of search and tracking modules for a FPGA-based GPS receiver	1
<i>Facundo Santiago Larosa, Martín Mignone and Nicolás Álvarez</i>	
sAPI (simpleAPI), a hardware-independent C library for Embedded Systems programming	7
<i>Eric Nicolas Pernia and Félix Safar</i>	
A New RM/DM Low Cost Schedulability Test	13
<i>Jose M. Urriza, Francisco E. Paez, Mariano Ferrari, Ricardo Cayssials and Javier D. Orozco</i>	
FPGA Quantum Computing Emulator Using High Level Design Tools	19
<i>Agustín Silva and Omar Gustavo Zabaleta</i>	
Offline Domotic System using voice comands	25
<i>Javier Omar Errobidart, Alejandro José Uriz, Esteban Lucio González, Ivan Exequiel Gelosi and Juan Alberto Etcheverry</i>	
WSN Clock Synchronization by Network-coded Messages	31
<i>Pablo Briff, Ariel Lutenberg, Leonardo Rey Vega, Fabián Vargas, Mohammad Patwary and Rolando Carrasco</i>	
Open-source embedded framework for Unmanned Ground Vehicle control using CIAA	35
<i>Facundo Pessacg, Matías Alejandro Nitsche, Adrián Teijeiro, Diego Martín and Pablo De Cristóforis</i>	
Software Patterns for Asymmetric Multiprocessing Devices on Embedded Systems: a performance assessment	41
<i>Pedro Martos and Alejandra Garrido</i>	
Implementation of an AXI-compliant lock-in amplifier on the RedPitaya open source instrument	47
<i>Luis Horacio Arnaldi</i>	

Real-Time Gaze-Tracking Embedded-System	53
<i>Cristian Ordoñez, Eduardo Blotta and Juan Pastore</i>	
A virtualized version of MIL-STD-1553	59
<i>Pablo Nicolás Solivellas, Hernán Ponso, Andrés Grop, Manuel Amor and Diego Salvador Fusari</i>	
Highly configurable Ethernet controller for HW/SW co-debugging	65
<i>Ricardo Cayssials, Damián Banfi, Lorenzo De Pasquale, Diego Martínez and Edgardo Ferro</i>	
Design of a Smart Lock on the Galileo Board	71
<i>Matías Presso, Diego Scafati, José Marone and Elías Todorovich</i>	
Characterization of Sensors and Design of an Embedded Photodetectors Array for Beam Profile Measurements in Radiotherapy	77
<i>Horacio Mateos, José Lipovetzky, Fabricio Alcalde, Martín Pérez, Pablo Cappagli and Mariano Gomez Berisso</i>	
Author index	83

Design and verification of search and tracking modules for a FPGA-based GPS receiver

Larosa, F. S. , Mignone, M. N.
Departamento de Ingeniería Electrónica
Universidad Tecnológica Nacional
Facultad Regional Haedo
Email: facundolarosa@gmail.com

Álvarez, N.
Escuela de Ciencia y Tecnología
Universidad de San Martín
Facultad de Ingeniería
Universidad de Buenos Aires
Email: nalvare2001@yahoo.com.ar

Abstract—GPS receivers constitute a topic of great importance since they have application in many fields of science and industry as geodesy, aviation, security and defense to name a few. The understanding of their internals, including the nature of the signals and the algorithms involved in their processing are crucial in the development of software defined receivers, which are the most common GPS receivers nowadays for custom applications. In this work, development and testing of search and tracking modules on a field programmable gate array (FPGA) is presented. Additionally, the need for such kind of receiver is presented along with a general architecture of the proposed system.

Keywords- GPS, Navigation, FPGA

I. INTRODUCTION

The use of GPS receivers is of great importance in navigation for a wide range of applications: ground vehicles, unmanned aerial vehicles (UAV), launchers, satellites, etc. Nonetheless, commercial GPS receivers often have performance limitations for certain applications where, for example, refresh rate of the receiver's solution is not fast enough for navigation purposes or additional data is desirable as observables like pseudoranges, carrier wavelength shift or carrier Doppler shift in order to use advance navigation techniques like tightly-coupled navigation algorithms [1].

The design of a GPS has the advantage that it can be customized for a given navigation platform with minimal complications as most of the receiver structure can be changed either because it is built on programmable logic (using a hardware description language) or in a microcontroller firmware.

Even more, a GPS can be upgraded incrementally to include other satellite constellations like GLONASS, GALILEO or BeiDou or to integrate its output with inertial sensors (accelerometers, gyroscopes, etc.), for example.

The GPS constellation is nominally composed of 24 satellites which orbit the Earth in approximately 55° inclination elliptical orbits in orbital planes separated 60° in azimuth. Orbits have a medium radius of 26,000 km (MEO, Medium Earth Orbits) having a period of approximately 12 hours (half a sidereal day) [2].

Satellites propagate different signals of military and civilian use. The most common civilian signal is the L1 signal which

is used to broadcast a navigation message composed of the satellite's health state, its orbital parameters (ephemerides) and correction parameters [3]. It is through these parameters that a satellite position with respect to the Earth can be calculated. Along with an indirect measurement of the distance of the receiver in reference to the satellite (called pseudorange), the receiver position can also be calculated using a variety of methods, for example by a least-square optimization [4]. The remaining section of this work is organized as follows: in section II a short overview of GPS signal generation is presented; section III presents general and design considerations to be used as references in the implementation stage; section IV describes the implementation of the modules and finally section V presents the conclusions and the future work to be done.

II. GPS SIGNAL GENERATION OVERVIEW

In order to understand the need and structure of the designed modules a short introduction of GPS signals is presented. The most relevant signal the L1 civilian signal which is transmitted by every satellite using the same carrier frequency of 1575.42 MHz. Each satellite multiplies its own navigation message with a pseudo-random sequence (often called Course Acquisition Code or CA Code) which is unique to each of them. This CA Code has a period of 1023 chips (bit designation is reserved for data bits while chips is used for CA Code bits) and a bit rate of 1.023 Mbps so the sequence is repeated every 1 ms. Finally, the composed digital signal is used to modulate the phase of the carrier as in shown in Fig. 1.

CA codes belong to the family of Gold pseudo-random codes. These codes are useful because of their particular correlation characteristics [4]. Autocorrelation of Gold codes has a maximum when the codes are in phase and is negligible when the autocorrelated code phase differs in more than a chip. Also, cross-correlation of two different Gold codes (i.e., codes belonging to different satellites) is negligible for any phase value. In this way, satellites navigation messages can be obtained correlating incoming GPS signal with a local signal replica of a given GPS satellite signal.

However, as satellites move along its orbits, the apparent carrier frequency is shifted due to Doppler effect. This shift changes in function of the satellite velocity and their relative position [6].

In order to recover any of the GPS satellite signals the phase of CA code and its central carrier frequency must be found as to extract the navigation message from the carrier and CA code. The action of finding the CA code phase and carrier frequency (or its deviation from the nominal central frequency) is performed by the search module. Once a satellite CA code phase and central frequency is found, this result is forwarded to a tracking module which performs continuous tracking of a given signal in CA code phase and carrier frequency.

III. CONSIDERATIONS

1) *General considerations:* The architecture of the proposed GPS receiver is shown in Fig. 2.

The first stage of the receiver is the front end module which process GPS signal so it can be subsequently processed in real time by the field programmable gate array (FPGA) modules. In Fig. 3 a block diagram of the front end is shown.

The RF signal is received by the antenna (an active antenna is usually used in order to improve overall noise figure [7]), preamplified and then filtered to reject image frequency. Subsequently it is mixed with a local oscillator signal in order to downconvert it to an intermediate frequency (IF). Then, it is filtered again to reject undesired mix products (high order terms) and digitized. It is common practice in these cases, to digitize using few resolution bits (up to three). In our case, one bit digitizing was used as it allows simpler search and

tracking modules with a loss of about 2.42 dB [8], which is allowable for the design.

The digitized IF signal is then processed in the FPGA modules which perform real time operations as search and tracking. The navigation message obtained by the FPGA modules is then forwarded to the microcontroller unit which calculates the navigation solution (position and velocity of the vehicle) and perform control operations over the FPGA modules. In our work, parameters shown in Table I where consistently used.

TABLE I
FRONT END PARAMETERS.

Parameter	Value
IF	4.092MHz
Sampling frequency	16.368MHz
Signal resolution	1 bit

These parameters are related with monolithic front end integrated circuits like Maxim 2769 or Skyworks SE4150L which are possible commercial front ends which could be used in conjunction with the designed modules.

2) *Design considerations:* For module design a Xilinx Spartan 3E 500 FPGA was selected as platform, which was available for this purpose. However, it was sought to generate a design as portable as possible avoiding instantiation of device vendor specific modules (IP cores) with the exception of internal RAM blocks and digital clock managers (DCM). In this manner, the designed modules could be ported with least effort to a different platform if necessary.

High level modules (those with a greater grade of complexity) where designed using a finite state machine with datapath (FSMD) implementation [9].

A FSMD is composed of a finite state machine which performs control operations (control path), examines external commands (through its control inputs) and depending of its present state generates control signals which operate the sequential circuits that perform data processing (data path). A general block diagram is shown in Fig. 4.

This implementation favors a maximum degree of control over the design, portability and reduces to a minimum resource utilization on the FPGA. On the other hand, it demands more development time as different circuits need to be developed in the lowest level: combinational circuits, registers, etc.

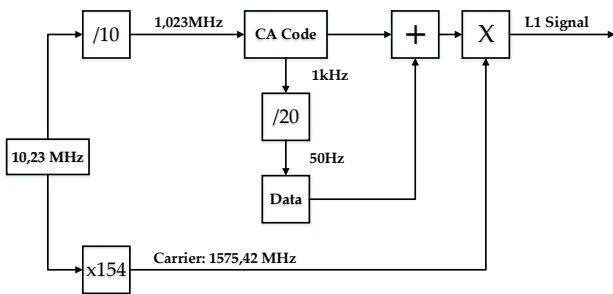


Fig. 1. General block diagram of L1 signal generation.

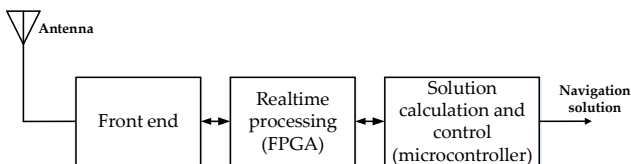


Fig. 2. Block diagram of GPS receiver.

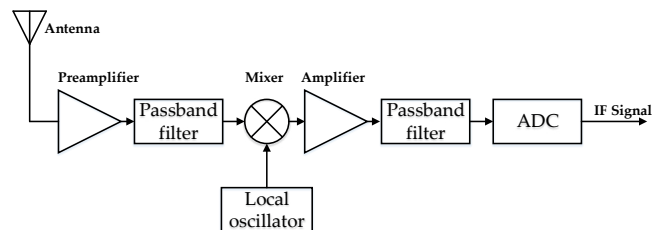


Fig. 3. Block diagram of GPS receiver front end.

IV. IMPLEMENTATION

1) *Input module*: The input module adapts signals incoming from the front end (IF data and clock) so it can be used by the designed modules. This module convert signals to the FPGA clock domain, avoids metastable states and number the incoming samples. This last requisite is important since signal processing is done synchronously. External interface of input module is shown in Fig. 5. *clkInput* and *dataInput* are clock and data inputs to where front end signals are connected. *clkSystem* is the system clock and it has been set to 81.84 MHz for this design. This frequency is an integer multiple of front end clock frequency ($81.84\text{MHz} = 5 * 16.368 \text{ MHz}$). This clock selection was made in order to use a single clock reference for both front end and FPGA modules in a future implementation. *dataOutput* and *dataReady* are the output signals which are used by the search and tracking modules along with their sample number *phaseOut*.

2) *Search module*: The search module correlates data input signal with local generated replicas for different:

- Satellite number
- IF frequency
- CA code phase

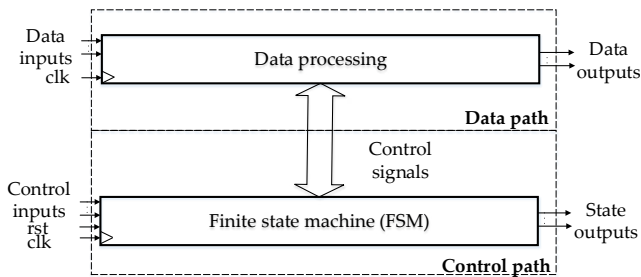


Fig. 4. General block diagram of a FSM.

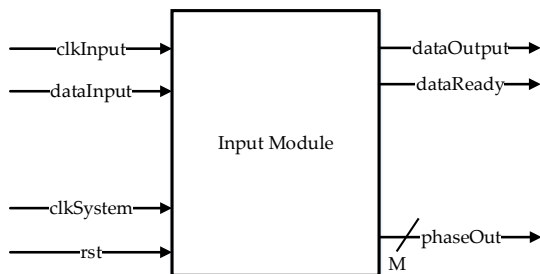


Fig. 5. Input module interface.

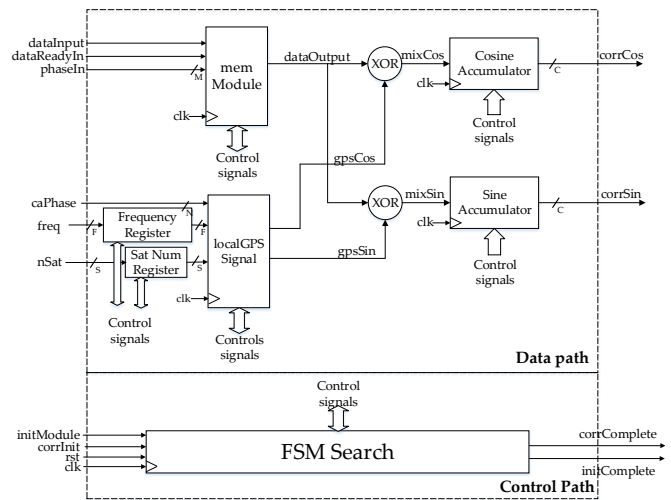


Fig. 6. Block diagram of search module.

The result of this correlation is used to determine if a given satellite is present on the IF signal and if so, its present IF frequency (which may be shifted from the nominal IF frequency due to Doppler shift) and CA code shift. These two results (IF frequency and CA code shift) are used as inputs by a tracking module for continuous tracking of that satellite signal in both frequency and CA code phase. In Fig. 6 a block diagram of the search module is presented.

In order to understand how search module works certain blocks inside the structure must be clarified. The block named *memModule* is composed of two first in first out (FIFO) buffers. At any moment, one of the buffers is being written with incoming data and the other stores fresh data samples which can be read and processed by the module. The length of the buffer is enough to accommodate an entire period of CA code (i.e. 1 ms of samples). When the buffer being written is completed, their roles are exchanged. In this manner, several advantages are achieved. In the first place, data samples can be processed at system clock frequency (instead of data clock) thus improving processing speed, also there is no need of waiting the beginning of a new period of the signal.

The block named *localGPSSignal* locally generates a GPS signal replica (i.e. a sinewave carrier mixed with CA code) of a given satellite and IF frequency and writes it in an internal memory. Once the signal is recorded it can be reproduced at the module output without further penalty. The *localGPSSignal* module along with the *memModule* module allow fast processing of data signals as they can be updated at system clock frequency. Finally, the two signals (the front end data signal and the local replica) are correlated by the exclusive or (xor) operation and accumulation. As to achieve phase invariance respect to input carrier phase two branches are calculated in quadrature, one with a cosine replica and the other with a sine replica.

3) *Search module verification*: In order to verify that the local GPS signal replicas were generated correctly, a MATLAB function named *GenGPSSignal* was developed that generates a synthetic GPS signal at the same rate that the front end. The function allowed to set a given IF frequency, Doppler shift, initial carrier and CA code phase, satellite number and quantization bits. Also, a *SearchFFT* function was written that generated a correlation matrix which is composed of the correlation values of a given signal and GPS signal replicas for different central frequencies and CA code shifts. The FFT search method [10] was used to speed up simulations. The signals (*LocalGPSSignal*) generated inside search module were exported to a text file using the standard VHDL *textio* package. Then, the correlation results of these signals were compared with those generated by the *GenGPSSignal*, named *TestGPSSignal*. Verification experience is summarized in Fig. 7. Correlation properties of both signals proved to be similar.

In Fig. 8 and 9 both *LocalGPSSignal* and *TestGPSSignal* correlation properties are shown for a particular case where CA code phase was set at 1021.5 chips and frequency displacement from central frequency was 0 Hz. As two quantities (carrier frequency and CA code phase) can be varied, in Fig. 8 CA code phase is variable while carrier frequency is held fixed. Conversely, in Fig. 9 carrier frequency is variable while CA code phase is held fixed.

In order to verify the search module a test scheme was developed which is summarized in Fig. 10.

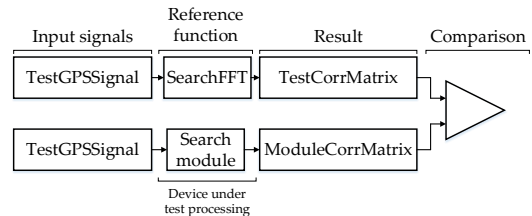


Fig. 10. Verification scheme of search module.

In the first place, a test signal was generated using the scripts previously described in this section (*TestGPSSignal*). Secondly, this signal was processed by two different ways. In one hand, it was processed by the *SearchFFT* function to produce as result a correlation matrix (*TestCorrMatrix*) which is used as reference. On the other hand, it was injected in the search module testbench through a text file which was then reproduced as the input of the module. Then, this input signal was correlated for different carrier frequencies and CA code phases. The result of this operations was recorded in a correlation matrix (*ModuleCorrMatrix*). Finally, both matrices were compared. In Fig. 11 and 12 an example is shown where the correlation peak was located at 5.5 chips of CA code phase and -1000Hz displacement from central frequency.

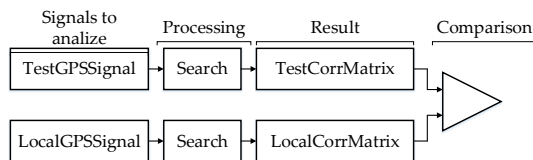


Fig. 7. Verification scheme of locally generated GPS signals.

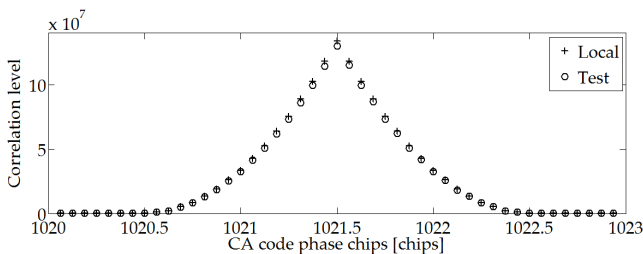


Fig. 8. Comparison between correlation properties of test and generated GPS signals. CA code phase is variable while carrier frequency is held fixed at correlation peak.

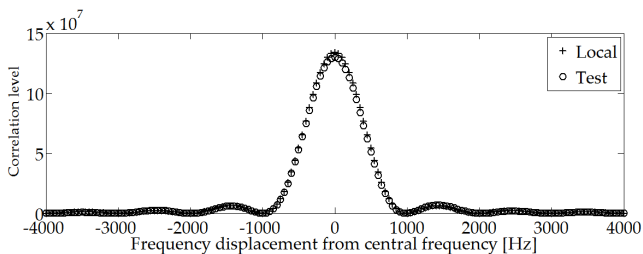


Fig. 9. Comparison between correlation properties of test and generated GPS signals. Carrier frequency is variable while CA code phase is held fixed at correlation peak.

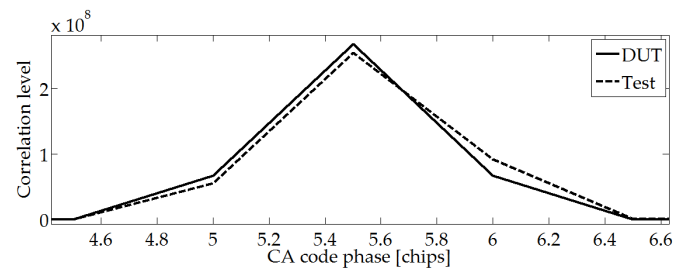


Fig. 11. Comparison between correlation processes accomplished by script (continuous line) and FPGA module (dashed line). CA code phase is variable while carrier frequency is held fixed at correlation peak.

4) *Tracking module*: The purpose of tracking module is to generate data to allow the tracking of a given GPS satellite signal.

In Fig. 13 a block diagram of the module is shown. This module was implemented as a FSM as described in a previous section. The control path is composed of a finite state machine which generates control signals for the numerically controlled oscillator (NCO), CA generator, accumulators and output registers blocks in the data path. The data path comprises:

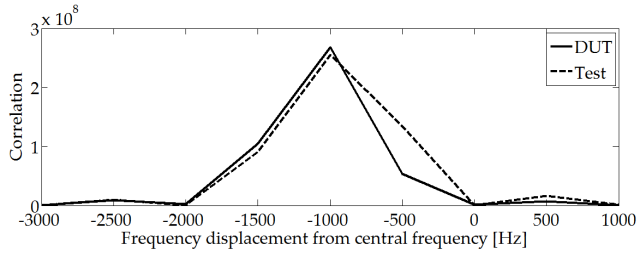


Fig. 12. Comparison between correlation processes accomplished by script (continuous line) and FPGA module (dashed line). Carrier frequency is variable while CA code phase is held fixed at correlation peak.

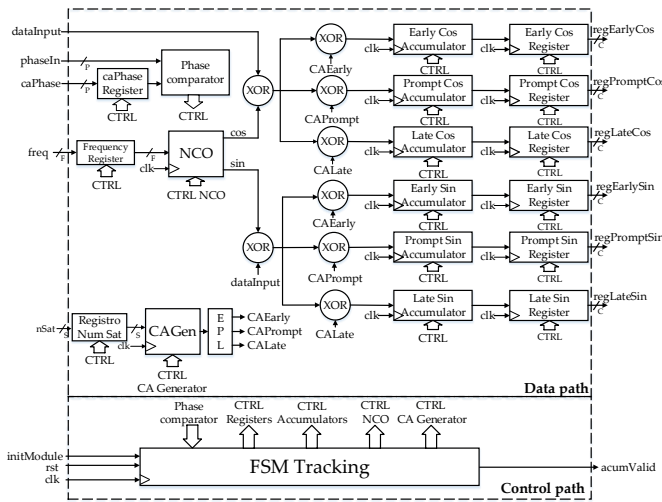


Fig. 13. Tracking module block diagram.

- **NCO:** This block generates local IF carrier signal which frequency is determined by input *frequency*. This input is a signed integer which allow generation of signal in the range of $[f_I - 8kHz, f_I + 8kHz]$, where f_I is the IF of 4.092 MHz. In this case, the internal register of the NCO was defined as a 32 bit register allowing a frequency precision of 0.02Hz [11].
- **CA generator:** This block generates a CA code sequence for a given satellite as indicated by *nSat* input. The block incorporates a shifter that allows generation of three CA code replicas (*CAEarly*, *CAPrompt* and *CALate*) which are shifted half a chip between them (*CALate* is delayed half a chip and *CAEarly* is advanced half a chip respect to *CAPrompt*).
- **Accumulators:** This register increments (or decrements) its value in case input signal coincides (or does not coincide) with the local replica.
- **Registers:** Input registers store signal generation parameters like IF frequency, satellite number and CA code phase. Output registers accumulate the last valid generated value of the accumulators.

In order to understand how the module allows tracking of

the incoming GPS signal, the analysis is divided in two cases. In the first case, it is assumed that input signal carrier coincides with IF frequency, while CA code phase could be different. In the case that incoming signal and local *CAPrompt* are in phase, correlation is maximum for the *CAPrompt* branch, while *CAEarly* and *CALate* branches present half the maximum value [10]. Fig. 14 illustrate this case.

For example, if incoming signals is delayed in reference to local signal, correlation of the *CALate* branch will increase while *CAPrompt* and *CAEarly* values will diminish. In this case, CA phase of the local replica must be compensated in order not to lose track of the incoming signal. Fig. 15 illustrate this case.

Following the same line of thought, in the second case it was assumed that the incoming signal and the local replica present the same CA code phase while their carrier frequency could vary. It is important to note that the tracking module structure (setting aside the *CAEarly* and *CALate* branches) resembles a Costas loop architecture (without the feedback loop which will be provided externally). This kind of architecture is of common use in tracking of GPS signals [12] where accumulators replace the classical low pass filter used in these configurations. When CA phase is locked, variation of phase and frequency of the carrier can be followed by means of the *CAPrompt* branches.

5) *Tracking module verification:* Tracking module produces at its output correlation values between the incoming signal and six local signal replicas which are generated by mixing the sine and cosine IF NCO signals with three versions of the CA code signal (early, prompt and late). These values are used to determine if incoming signal is locked in phase with the local replica (in both, carrier frequency and CA phase).

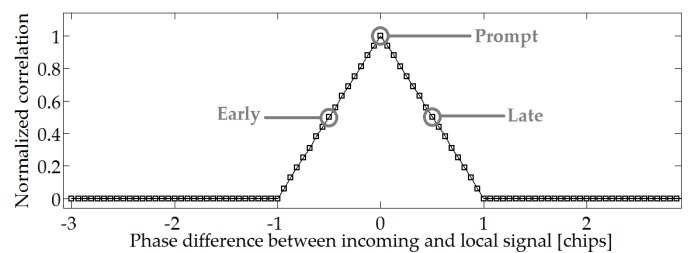


Fig. 14. Normalized correlation of incoming and replica signals (CA code in phase).

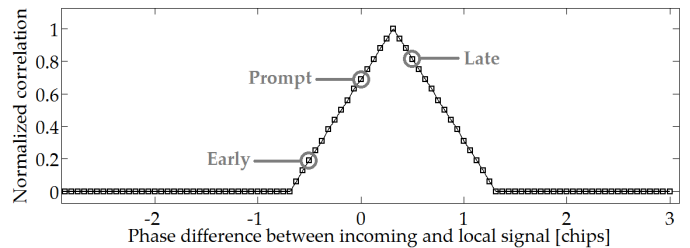


Fig. 15. Normalized correlation of incoming and replica signals (input signal is delayed in reference to local replica).

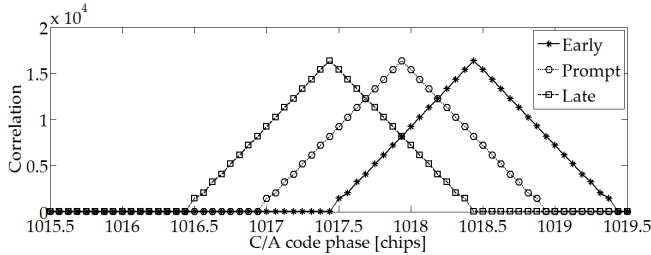


Fig. 16. Correlation values for early, prompt and late branches. The CA code phase is varied in the neighbourhood of the correlation peak.

The module was tested through functional testbenches which included testing of CA code generators, locking of input signal and correct correlation of the input signal and local replicas. For a detailed list of the tests and a complete description of inputs and outputs of the modules please see [13]. In order to illustrate module operation, it was fed at its input using a synthetic signal with displacement from central frequency $\Delta f_c = 0\text{Hz}$ and CA code shift $n_0 = 1018$ chips. In Fig. 16 correlation values for each branch (Early, Prompt and Late) are shown. Correlation is calculated as the root square of sine and cosine squared accumulators.

6) *Use of FPGA resources:* In Tables II and III utilization of FPGA resources is shown for our design reference (Xilinx Spartan 3E 500) for a both search and tracking modules. The amount of resources used is shown for a single module.

TABLE II
RESOURCES ALLOCATED FOR A SINGLE SEARCH MODULE.

Resource	Used	Available	Percentage of use
Slice Flip Flops	214	9.312	2%
4 Input LUTs	326	9.312	3%
Occupied slices	278	4.656	5%
RAMB16s	4	20	20%

TABLE III
RESOURCES ALLOCATED FOR A SINGLE TRACKING MODULE.

Resource	Used	Available	Percentage of use
Slice Flip Flops	342	9.312	3%
4 Input LUTs	261	9.312	2%
Occupied slices	282	4.656	6%
RAMB16s	0	20	0%

Both modules were synthesized with the clock timing constraint of 81.84 MHz and in both cases this constraint was met.

V. CONCLUSION

GPS L1 signals could be studied and analyzed through several scripts. These scripts were used to verify search and tracking modules and could be used to test the front end to be built.

Also, search and tracking modules could be described, synthesized and verified using the proposed implementation

techniques. The use of FPGA resources was reduced as to implement the modules in a low or middle range FPGA. Additionally, the operation frequency was reached without using any sort of optimization tool.

Next steps will be to build a front end circuit to test designed modules, which have been presently tested using synthetically generated signals.

ACKNOWLEDGMENT

The authors would like to thank Universidad Tecnológica Nacional which financially supported this work through UTN PID 3636 and UTN PID 4525.

Also, as this work was developed as a final thesis project for Carrera de Especialización de Sistemas Embebidos (CESE) of Universidad de Buenos Aires, we would like to thank to all its teaching staff and its director, Ariel Lutenberg who continuously strive for excellence.

REFERENCES

- [1] A. Hasan et al, *A Review of Navigation Systems (Integration and Algorithms)*, in *Australian Journal of Basic and Applied Sciences*, vol 3, 2009, pp.943-959.
- [2] J.B. Tsui, "Satellite constellation" in *Fundamentals of Global Positioning Systems Receivers: A software approach*, Hoboken, New Jersey, John Wiley and Sons, 2005, ch. 3, pp.31-33
- [3] J.B. Tsui, "Basic GPS Concepts" in *Fundamentals of Global Positioning Systems Receivers: A software approach*, Hoboken, New Jersey, John Wiley and Sons, 2005, ch. 2, pp.13-15
- [4] J.B. Tsui, "GPS C/A Code Signal Structure" in *Fundamentals of Global Positioning Systems Receivers: A software approach*, Hoboken, New Jersey, John Wiley and Sons, 2005, ch. 5, pp.79-100
- [5] R. Gold, "Optimal binary sequences for spread spectrum multiplexing" in *IEEE Trans. on Information Theory* vol. 13, pp.619-621, 1967
- [6] J. Zhang et al, "On the relativistic Doppler effect for precise velocity determination using GPS", *Journal Of Geodesy*, vol. 80, pp.104-110, 2006
- [7] L. Couch, "Wire and Wireless Communications Applications" in *Digital and Analog Communication Systems*, New Jersey, Pearson Education, 2013, 8th Edition, ch. 8, pp.603-609
- [8] C. Hegarty, *Analytical Model for GNSS Receiver Implementation Losses*, The MITRE Corporation.
- [9] P. Chu, "FSMD" in *FPGA Prototyping by VHDL Examples*, Hoboken, New Jersey, John Wiley and Sons, 2008, ch. 6, pp.127-157
- [10] F. Johansson, "GPS Satellite Signal Acquisition and Tracking" Available: <http://www.sm.luth.se/csee/courses/sms/019/1998/navstar/navstar.pdf>.
- [11] Xilinx Inc., "LogiCORE IP DDS Compiler v4.0." Available: go.gl/QVmjnw
- [12] E. Kaplan, "Satellite Signal Acquisition, Tracking, and Data Demodulation" in *Understanding GPS: principles and applications*, Norwood, Massachusetts, Artech House, 2006, ch. 5, pp.163-170
- [13] F. Larosa, "Diseno e implementacion" in *Modulo de busqueda, seguimiento y decorrelacion para un sistema GPS sobre FPGA*, Buenos Aires, Argentina, 2016, pp 43-35. Available: <http://laboratorios.fi.uba.ar/lse/tesis/LSE-FIUBA-Trabajo-Final-CESE-Facundo-Larosa-2016.pdf>

sAPI (*simpleAPI*), a hardware-independent C library for Embedded Systems programming

Ing. Eric Nicolás Pernia

Departamento de Ciencia y Tecnología
Universidad Nacional de Quilmes (UNQ)
Quilmes, Buenos Aires, Argentina

eric.pernia@unq.edu.ar / ericpernia@gmail.com

Mg. Ing. Félix Safar

Departamento de Ciencia y Tecnología
Universidad Nacional de Quilmes (UNQ)
Quilmes, Buenos Aires, Argentina

felix.safar@unq.edu.ar

Abstract— A library design is presented aimed to standardize C language programming on microcontroller-based platforms. This library defines a simplified Application Programming Interface (API) that abstracts the most common modes of use of typical peripherals found in current microcontrollers in the marketplace. In this way, it becomes possible to program them with no need of knowing details on the underlying architecture. This design promotes and enables hardware-independent programming, reducing overall complexity of embedded systems development. Reference implementation on the EDU-CIAA-NXP platform is detailed.

Keywords—*sAPI; API; HAL; Embedded Systems; Microcontroller; C microcontroller library;*

I. INTRODUCTION AND BACKGROUND

Currently the majority of embedded systems programming based on microcontrollers is written in C language, using libraries for handling the processing core (or cores) and peripherals. Consequently, a C library is an integral part of any embedded microcontroller-based system design, even in cases when programming is done in another high-level language.

In this paper we present a library design to program microcontroller-based Embedded Systems platforms in a simplified way. It allows the most common modes of typical peripherals of a microcontroller. It also outlines main features in a reference implementation on the EDU-CIAA-NXP platform [1].

This work is part of a group of initiatives promoted by the *Universidad Nacional de Quilmes* [2] to collaborate within the framework of the CIAA Project [3], to facilitate the development and teaching of Embedded Systems in Argentina. Other initiatives include: Ladder PLC programming environment (IDE4PLC [4]), Java programming for CIAA [5], and Firmata4CIAA [6] to facilitate graphical blocks programming in BYOB Snap! [7] language for use in secondary schools.

II. MOTIVATION AND CONTEXT

There are a great variety of microcontroller-based platforms in the market, and while they all expose microcontrollers with similar characteristics and compatible peripherals, it is observed that their libraries are very different. This is because each manufacturer and/or associated companies offer their own libraries written in C language, which are designed with strong dependence upon the underlying microcontroller architecture that these platforms contain.

There are also many libraries that achieve good hardware abstraction across platforms, but none of them have been adopted as a general standard. Among several, it can be cited:

- In the automotive industry there is a standard called AUTOSAR [8] aimed for the standardization of automotive electronic systems. This architecture proposes very extensive and complex to implement library definitions, making difficult to learn and use. Also, it is not yet been extended to other industries.
- Driver libraries based on POSIX [9] (can be found on systems with embedded Linux such as Raspberry Pi [10]). Its abstraction has been very useful for the standardization of drivers for PC's with Unix-compatible operating systems. However, it is so far from the physical hardware that in practice is not very used to program embedded systems.
- Embedded systems hobbyists have pushed standardization of programming through a library known as Wiring [11] which is available for multiple platforms (such as the popular Arduino [12]). Although this library achieves great ease of use and fast learning, it contains some technical inaccuracies that provoke unwanted vices in the learning of programming microcontrollers. It also lacks an API definition for the use of timers, peripherals widely used in microcontroller applications.
- Currently there are many manufacturers of microcontrollers that have acquired licenses for the

manufacturing of microcontrollers with processing cores of Cortex architectures [13] designed by the company ARM [14]. For them there is a standard library called CMSIS [15] (Cortex™ Microcontroller Software Interface Standard) to program of the processing cores and interrupt controllers but does not extend to peripherals where each manufacturer seeks to differentiate itself from its competitors.

- Other well-known companies in the field of embedded systems programming such as Microchip [16] provide hardware-dependent libraries and automated code generators. These tools, aimed to accelerate the development timeframes, do not perform very well when an application requires more advanced modes, making programming difficult since some configurations steps over the others cause coding problems.

From previous examples it is observed that for the realization of a standard library satisfying different users a balance must be achieved among following requirements:

- Size of API definition.
- Hardware dependency.
- Level of abstraction.
- Complexity of learning and use.
- Peripherals and supported modes.
- Scalability.

Due to these reasons, in order to overcome above exposed difficulties it was decided to elaborate an API definition of a standard library for microcontrollers in C language.

III. DESCRIPTION OF SAPI LIBRARY

A. Methodology of work

In order to develop the sAPI library the following practices were chosen:

- Use of Git repository [17], within github site [18].
 - Development workflow through Fork and Pull Requests on github site.
 - Issues are opened within github for each feature to design and/or improve.
- Code production and documentation:
 - Writing files with .h extension for each module, documented using Doxygen [19] tool, and later writing the .c extension files with their implementation.
 - In addition, a document with source code styles is defined.
- General documentation (outside source code) written in Markdown [20] language, which allows exporting contents to html, latex, and pdf formats, among others.

B. Design principles

Based upon aforementioned items and other good practices, it was decided to use the following premises to guide the design:

- Support programming of core processing and main peripherals along with most common modes of operation.
- Possess a level of abstraction sufficient to become independent of microcontroller hardware while maintaining the identity and key concepts of each module that contains the microcontroller.
- Keep moderate the extension of API definition extension.
- Use of simple names to facilitate learning and utilization.
- Explicitly specify the dependency between library modules and the use of physical resources in each particular implementation for each platform.

C. Library's architecture

An application based upon sAPI library contains at least the software layers described in Figure 1.

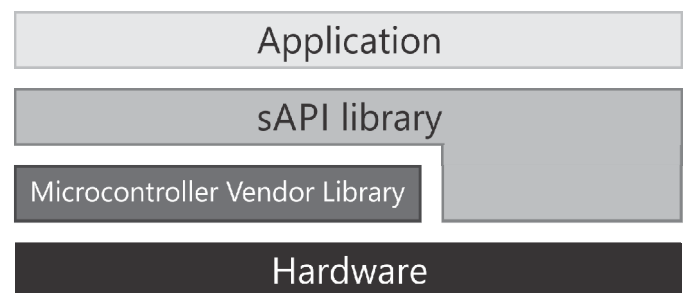


Figure 1, software layers for an application based upon sAPI library.

The sAPI library must isolate the user application from the underlying hardware by forming an abstraction layer of the hardware. Depending on the particular implementation, existing libraries provided by the microcontroller manufacturer of a given platform may be used.

The sAPI library model in turn consists of software modules, grouped into the software layers that are summarized in Figure 2.

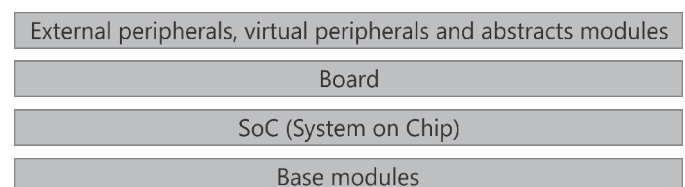


Figure 2, sAPI library software layers.

The characteristics of each layer are shown below. Then the composition of a library module (generic) is exposed.

1) External peripherals, virtual peripherals and abstract modules

The main feature of this type of modules is that specific structures must be assigned because they do not exist until they are "created" and "initialized". Among these, the following are included:

- External peripherals connected to the platform to be used, such as an I2C or SPI connected chips and/or modules.
- Virtual peripherals, for example, a software-implemented I2C that uses GPIOs hardware to operate.
- Abstract modules: they represent modules that incorporate a higher level of abstraction which allow performing advanced tasks such as management of timing of an operative system and management of buffers. These modules are highly portable as they are designed without hardware dependency.

2) Board

It contains definitions at full platform level, by defining, among other things, the names of platform physical connectors (pins, terminals, and other connectors) that are associated with lower level modules.

3) SoC (System on Chip)

This layer groups modules that model the processing core (or cores, for the case of multicore systems) and the internal physical peripherals of the platform. These modules having physical existence in the microcontroller of the platform, and are not to be "created", rather they are only "initialized". In other words, they always exist within the program; they have defined fixed names and cannot be destroyed during the program. These are:

- CORE: models a processing core.
- GPIO: models a single general purpose I/O pin (pin) as well as a set of pins (port). The written or read value of a pin is of the Boolean type.
- ADC: models an Analog-to-Digital converter peripheral. The read value is of unsigned integer type.
- DAC: models a Digital-to-Analog converter peripheral. The written value is of unsigned integer type.
- TIMER: models a Timer/Counter peripheral.
- RCT: models a Real Time Clock peripheral.
- UART: models a serial communication Universal Asynchronous Receiver Transmitter peripheral.
- SPI: models a Serial Peripheral Interface peripheral.

- I2C: models an Inter-Integrated-Circuit peripheral.

4) Base modules

These model common parts associated with the entire library and correspond to:

- Data types.
- General board initialization.

5) Composition of a library module

Each module contains a defined set of data types, properties, and methods. Each of the modules contains a set of properties that are grouped into a structure associated with the module. This is defined as a data type with the following structure:

```
typedef struct{
    <type> property1;
    <type> property2;
    ...
}<module>_t;
```

This structure is the area where the module is mapped and is accessible from the public API only as a named index. An enumerated type associated with the name of the module is then defined to access it by name (on physical peripherals, e.g. I2C0, SPI1, UART4, etc.). This type definition is below:

```
typedef enum{
    <MODULE0>,
    <MODULE1>,
    ...
}<module>_t;
```

Typical properties of a physical peripheral are:

- configParamName: peripheral configuration value.
- power: turning it on or off.
- value: value to read or write.
- eventName: some event associated to the peripheral (either interrupt or other).
- eventNameCallback: a structure with a pointer to function and pointer to parameter that the user can pass to that function (either interrupt or other).

Finally, a set of access methods for each module, including:

- Module initialization. This turn on the module if necessary and initialize it with the most typical configuration used:

```
<module>Init (
    <MODULEi>,
    <mostCommonPropertyValue> |
```

```
<MODIFY_FLAG1> | ... |
<MODIFY_FLAGn>
);
```

- Property values assignment:

```
<module><PropertyName>Set (
  <MODULEi>,
  <propertyValue> | MODIFY_FLAG1 |
    MODIFY_FLAG2(value) | ... |
    MODIFY_FLAGn
);
```

- Property values getter:

```
<propertyValue> = <module><PropertyName>Get(
  <MODULEi> );
```

- In order to simplify coding, most modules have an alternative API with read, write and configure methods:

```
<propertyValue> = <module>Read<PropertyName>(
  <MODULEi> );

<module>Write<PropertyName>( <MODULEi>, value );

<module>Config(<MODULEi>, param_1, ..., param_n);
```

D. Reference implementation

A reference implementation of sAPI library has been carried out on the EDU-CIAA-NXP platform of CIAA project (based on the NXP LPC4337 microcontroller [21]). This implementation currently (as of April 2017) has the following modules:

1) Modules contained in reference implementation

External peripherals:

- External peripherals map.
- Seven segment Displays.
- Matrix Keypad.
- Angular Servo SG90 (0 to 180°).
- Magnetometer sensor (compass) HMC5883L.

Abstract modules:

- Delays:
 - Inexact Delay (for academic use).
 - Blocking Delay timer based.
 - Non Blocking Delay timer based.
- Buffers:
 - FIFO/LIFO buffer.
 - Circular buffer.

- Serial outputs (Print):
 - Console serial output.
 - Debugging serial output.

Virtual peripherals:

- Software I2C (using GPIO's).

Board:

- Board peripheral map.
- Platform general initialization (Board).

SoC:

- Internal peripheral map.
- Processing core (CORE).
- General purpose Input/Output (GPIO).
- Universal Asynchronous Tx/Rx (UART).
- Analog-to-Digital converter (ADC).
- Digital-to-Analog converter (DAC).
- Inter-IC serial bus (I2C).
- Real Time Clock (RTC).
- Low Power Modes (Sleep).
- Timer:
 - Periodic Interrupt (Ticker).
 - Count to overflow (Overflow).
 - Count to match (Match).
 - Input capture (InputCapture).
 - Pulse Width Modulation (PWM).

Base Modules:

- Data Types.
- Timer Periodic Interrupt to use as time base in time-triggered Operating Systems (Tick).

Each module includes one or more sample programs to demonstrate library utilization.

2) Application examples

In this section we expose two examples to show the sAPI library.

Blinky basic example:

```
// BlinkyExample.c
#include "sapi.h"
```



```
int main( void )
(
    boardInit();
    while(TRUE)
    {
        gpioToggle( LED1 );
        delay( 500 );
    }
)
```

Blinky tick event-based example:

```
// BlinkyTickExample.c
#include "sapi.h"

// Function that execute at every tick event
bool_t tickCallback( void* )
(
    if( tickRead() >= 500 ){
        gpioToggle( LED1 );
        tickWrite(0);
    }
)

int main( void )
(
    boardInit();
    // Configure tick event callback function
    tickEventCallbackSet( tickCallback );
    // Configure tick event at 1 ms rate
    tickInit(1);
    while(TRUE)
    {
        // Do other stuff
    }
)
```

E. Partial implementation in other platforms

There are also partial implementations of the same library for the following platforms:

- Industrial Automation and control board CIAA-NXP [22] (an industrial computer board with NXP LPC4337 microcontroller).
- LPCXpresso 1769 (NXP LPC1769 microcontroller) development board [23].
- Atmel ATmega32A [24] custom development Board.
- Intel Quark D-2000 Development Board [25].
- Microchip PIC18LF46K22 [26] custom development Board.
- Silicon Labs Happy Gecko Development Kit [27].
- Silicon Labs Pearl Gecko Development Kit [28].

IV. USE CASES

The first version of the sAPI library (made in 2015 within the context of the Java [5] project) has been used by the first author as an example of hardware abstraction layer in university courses. These courses are: the postgraduate course "Microprocessor Programming" within the "Carrera de Especialización en Sistemas Embebidos" (CESE) of FI-UBA

[29] where the first author has been Professor in charge for three editions, and the course "Digital Systems" within the Engineering Degree in "Ingeniería en Automatización y Control Industrial" (IACI) of the "Universidad Nacional de Quilmes" (UNQ) [2] where the author currently serves as Instructor.

Starting in 2016, it was decided to use the library as part of the "Cursos Abiertos de Programación de Sistemas Embebidos" (CAPSE [30]) organized by ACSE [31]. In this way the library has been extended considerably to explain the use of all typical microcontrollers' peripherals with impressive learning results for students both at advanced levels and those who take their first steps in learning microcontroller programming.

In addition, sAPI library was made available to anyone since it is freely available published online under a modified BSD [32] license on the author's github site [33] and has been adopted by a significant number of users.

Finally, in December 2016 it was decided to use the library as a standard library for the CIAA Project platforms. This led to a thorough review and improvement of it currently being worked on.

V. CONCLUSIONS AND FUTURE WORK

Based upon the results obtained in using the library for teaching different courses at different levels and the acceptance of multiple programmers for their individual projects, it is believed that it has been possible to design a library that allows programming in C language in a simplified form in microcontroller-based embedded platforms without the need to know in detail their architecture. In particular, for experienced programmers, the sAPI library is able to streamline the development of applications through its direct utilization, extension or reconfiguration according to their needs; or even taking it as a basis to understand the programming of a particular platform architecture by reviewing its source code.

Within the field of teaching embedded systems programming, it allows concentrating the efforts of novice students in understanding programming applications on embedded systems as a whole, rather than learning the programming of a single particular platform, making it possible to understand the important concepts regardless of the underlying hardware. Then, once training progress a student can understand how the library is made for a particular platform and take it as an example of a hardware abstraction layer. In this way, it is emphasized at all times the fundamental fact of designing applications independent of hardware, a real benefit given the speed of change of these hardware devices in current marketplace.

The library is still being ported to other microcontrollers and improving its definition. It is expected to complete by the end of this year 2017 the whole library implementation for the following platforms:

- CIAA-NXP (full-blown industrial level board).

- Pico-CIAA [34] (NXP microcontroller).
- CIAA-PIC (Microchip microcontroller).

It is also intended to work within near future in the development of a simulator, capable of running a program written in C using sAPI library, on a virtual board created within a PC environment, to facilitate teaching of programming without having the need of an available microcontroller's physical platform.

ACKNOWLEDGMENT

First To Martín Ribelotta whose experience and insight has been and continues to be much valuable support for the revision process of the library.

To Dr. Ing. Ariel Lutemberg and Dr. Ing. Pablo Gómez who relied on using sAPI for the courses at CAPSE.

To the coordinators of Proyecto CIAA who relied and decided to adopt sAPI as a standard library, in particular to general coordinator Esp. Ing. Pablo Ridolfi.

To the students of CESE (FI-UBA), IACI (UNQ), and CAPSE, who used and received it very enthusiastically boosting its development.

To the Departamento de Ciencia y Tecnología (UNQ), led by Dra. Alejandra Zinni for supporting Embedded Systems engineering projects.

Last, to Ing. Leonardo Gassman who proposed the sAPI name for the library during the Java Project at UNQ.

REFERENCES

- [1] EDU-CIAA-NXP. Educational board version of "Computadora Industrial Abierta Argentina", of "Proyecto CIAA". Available: 2017-05-01. [Online]: <http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:edu-ciaa:edu-ciaa-nxp>
- [2] Universidad Nacional de Quilmes. Institutional web available: 2017-05-01. [Online]: <http://www.unq.edu.ar>
- [3] "Proyecto CIAA", "Computadora Industrial Abierta Argentina". Available: 2017-05-01. [Online]: <http://www.proyecto-ciaa.com.ar/>
- [4] IDE4PLC: Ladder PLC programming environment. Available: 2017-05-01. [Online]: <http://ide4plc.org/>
- [5] Pernia, E. N., Gomez, P. M., Martos, E. I. P. I., & Sager, G. (2015). "Desarrollo de Firmware y Software para programar la CIAA en lenguaje JAVA con aplicación en entornos Industriales". Available: 2017-05-01. [Online]: <http://laboratorios.fi.uba.ar/lse/tesis/LSE-FIUBA-Trabajo-Final-CESE-Eric-Pernia-2015.pdf>
- [6] Firmata4CIAA. A program to run a Firmata Client on EDU-CIAA-NXP board. Available: 2017-05-01. [Online]: <https://github.com/ciaa/Firmata4CIAA>
- [7] BYOB Snap! Programming language. Available: 2017-05-01. [Online]: <http://snap4arduino.org>
- [8] AUTOSAR (AUTomotive Open System ARchitecture). Available: 2017-05-01. [Online]: <https://www.autosar.org/>
- [9] Why POSIX for embedded systems? Article available: 2017-05-01. [Online]: http://www.qnx.com/developers/docs/660/index.jsp?topic=%2Fcom.qnx.doc.neutrino.sys_arch%2Ftopic%2Fintro_Why_POSIX.html
- [10] Raspberry Pi platform. Available: 2017-05-01. [Online]: <https://www.raspberrypi.org/>
- [11] Wiring: open-source programming framework for microcontrollers. Available: 2017-05-01. [Online]: <http://wiring.org.co/>
- [12] Arduino platform. Available: 2017-05-01. [Online]: <https://www.arduino.cc/>
- [13] ARM Architectures. Available: 2017-05-01. [Online]: <https://www.arm.com/products/processors/instruction-set-architectures/index.php>
- [14] ARM. Available: 2017-05-01. [Online]: <https://www.arm.com/>
- [15] CMSIS - Cortex Microcontroller Software Interface Standard. Available: 2017-05-01. [Online]: <https://www.arm.com/products/processors/cortex-m/cortex-microcontroller-software-interface-standard.php>
- [16] Microchip. Available: 2017-05-01. [Online]: <http://www.microchip.com/>
- [17] Git repository Reference. Available: 2017-05-01. [Online]: <https://git-scm.com/docs>
- [18] Github site. Available: 2017-05-01. [Online]: <https://github.com/>
- [19] Doxygen. Generate documentation from source code. Available: 2017-05-01. [Online]: <http://www.stack.nl/~dimitri/doxygen/>
- [20] Markdown language. Available: 2017-05-01. [Online]: <https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet>
- [21] NXP LPC4337 microcontroller Available: 2017-05-01. [Online]: <http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/lpc-cortex-m-mcus/lpc4300-cortex-m4-m0/32-bit-arm-cortex-m4-m0-mcu-up-to-1-mb-flash-and-136-kb-sram-ethernet-two-high-speed-usb-lcd-emc:LPC4337FET256>
- [22] CIAA-NXP platform. Available: 2017-05-01. [Online]: http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:hardware:ciaa_nxp:ciaa_nxp_inicio
- [23] LPCXpresso 1769 development kit (NXP LPC1769 microcontroller) Available: 2017-05-01. [Online]: https://www.embeddedartists.com/products/lpcxpresso/lpc1769_xpr.php
- [24] Atmel ATmega32A microcontroller. Available: 2017-05-01. [Online]: <http://www.microchip.com/wwwproducts/en/ATmega32A>
- [25] Intel Quark D-2000 Development Board Available: 2017-05-01. [Online]: <http://www.intel.la/content/www/xl/es/embedded/products/quark/mcu/d2000/overview.html>
- [26] Microchip PIC18LF46K22 microcontroller. Available: 2017-05-01. [Online]: <http://www.microchip.com/wwwproducts/en/PIC18F46K22>
- [27] Silicon Labs Happy Gecko Development Kit. Available: 2017-05-01. [Online]: <http://www.silabs.com/products/mcu/32-bit/efm32-happy-gecko>
- [28] Silicon Labs Pearl Gecko Development Kit. Available: 2017-05-01. [Online]: <http://www.silabs.com/products/mcu/32-bit/efm32-pearl-gecko>
- [29] Postgraduate course "Microprocessor Programming" within the "Carrera de Especialización en Sistemas Embebidos" (CESE) of FI-UBA. Institutional web available: 2017-05-01. [Online]: <http://laboratorios.fi.uba.ar/lse/cursos.html>
- [30] "Cursos Abiertos de programación de Sistemas Embebidos" (CAPSE). Institutional web available 2017-05-01. [Online]: http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=educacion:cursos:cursos_programacion_ciaa
- [31] ACSE. Institutional web. Available 2017-05-01. [Online]: <http://www.sase.com.ar/asociacion-civil-sistemas-embebidos/>
- [32] Modified BSD 3 clause license. Available: 2017-05-01. [Online]: <https://opensource.org/licenses/BSD-3-Clause>
- [33] First author's github site. Available: 2017-05-01. [Online]: <https://github.com/epernia/sapi>

A New *RM/DM* Low Cost Schedulability Test

José M. Urriza¹, Francisco E. Páez^{1,2}, Mariano Ferrari^{2,3,4}

¹Depto. de Informática / ³Matemática – Fac. Ingeniería –
Universidad Nacional de la Patagonia San Juan Bosco
⁴CESIMAR – ²CONICET
Puerto Madryn, Argentina
josemurriza@unp.edu.ar, fpaez@unpata.edu.ar, mferrari7@gmail.com

Ricardo Cayssials⁵, Javier D. Orozco^{2,5}

⁵Depto. de Ingeniería Eléctrica y Computadoras
Universidad Nacional del Sur
Bahía Blanca, Argentina
jorozco@uns.edu.ar, ricardo.cayssials@uns.edu.ar

Abstract— This paper focuses on reducing the computational cost of iterative algorithms used to evaluate the schedulability of a Real-Time System scheduled by the Rate Monotonic (*RM*) or Deadline Monotonic (*DM*) policies. These algorithms calculate the worst case response time of each task of a real-time system. Several runtime strategies for fault tolerance mechanisms, power consumption methods and execution of non-real-time tasks, among others, require low computational cost schedulability algorithms to be efficiently implemented in real applications.

The algorithm proposed is based in mathematical properties of the real-time schedulability test, that reduce noticeably the number of iterations needed to converge to the final solution. Through simulations, it is found that the proposed algorithm produces a significant reduction in the average temporal cost reaching in some cases, a complexity reduction from $O(n^2)$ to $O(n \cdot \log(n))$, with respect to classical schedulability algorithms.

Keywords—Real-Time System; Scheduling; Mixed Critical Systems; Application-Defined Scheduling

I. INTRODUCTION

Real-time systems (*RTS*) are those systems in which results should be produced before a certain *deadline* [1]. Several design-time schedulability methods have been proposed to guarantee that a set of real-time tasks will meet their deadlines during runtime when they are executed by a processor. Most of these methods have a Computational Cost (*CC*) that turns them unfeasible to be executed at runtime in a real application. A schedulability method with a high *CC* could take so much processor time that produces missed deadlines on the real-time tasks analyzed. Consequently, a bounded *CC* is mandatory for a feasible runtime schedulability method. Several schedulability methods have been proposed to be implemented during runtime.

An exact schedulability method for an arbitrary number of real-time tasks with a low *CC* may be useful for an efficient utilization of the system resources. Consequently, efficient quality of service mechanisms, additional system features, and power saving strategies may be proposed with low *CC* scheduling strategies.

Real-Time Operating Systems (*RTOS*) are usually utilized to manage the system resources. One of the main functions of a *RTOS* is the schedulability of the real-time tasks. This function is implemented by the *scheduler*. Generally,

additional *RTOS* functions such as power saving strategies and non-real-time task execution, among others, are implemented as scheduler extensions. A low *CC* scheduling test method may improve the performance of these extensions.

In this paper, a low *CC* scheduling method is proposed that introduces improvements to previous iterative methods.

In the following a brief introduction to *RTS* scheduling is presented, and previous and related works are reviewed. Next, in Section II, the worst case response time analysis and its previous implementations as an iterative method are studied. In Section III the new method and its implementation as an iterative algorithm is presented. In Section IV, the performance of the algorithm is compared against previous methods, through several tests implemented on a development board. In Section V the results are discussed, and an analysis of the temporal *CC* is performed. Finally, the conclusions and future works are addressed in Section VI.

A. Real-Time System scheduling

Multi-tasks mono-processor *RTS* are modeled as a set S of n periodic tasks. Each task i (τ_i) is characterized by its *period*, T_i , its *deadline*, D_i , and its *worst case execution time*, C_i :

$$S(n) = \{(C_1, T_1, D_1), (C_2, T_2, D_2), \dots, (C_n, T_n, D_n)\}$$

Each time that a task requires to be executed, it is said that it is *invoked*. Because tasks are considered periodic, they are invoked periodically. Each time that τ_i is invoked, it requires to be executed at most C_i units of time by the processor to be completed. A task τ_i should be completed before its D_i to be considered *schedulable*. If all the tasks are schedulable, then the system is said to be *schedulable*, otherwise is *non-schedulable*.

When more than one task is requiring to be executed, then the highest priority one is chosen for execution. We assume the *Rate Monotonic* (*Deadline Monotonic*) fixed priority assignment in which tasks with smaller period (*deadline*) are assigned a higher priority. The utilization factor (U) of a *RTS* is:

$$U = \sum_{i=1}^n \frac{C_i}{T_i} \text{ with } i=1,2,\dots,n$$

In [2] is proposed the first schedulability analysis for a real-time multi-task mono-processor system. More advanced techniques compute the *Worst Case Response Time (WCRT)* of the tasks to analyze the system schedulability. The *WCRT* of τ_i , denoted R_i , is the maximum interval that the processor takes to complete a task invocation.

B. Related Work

Equation (1) is an exact condition to determine the schedulability of a *RTS*. In 1986, Joseph and Pandya ([3]) proved that (1) hasn't got an analytical solution and they introduced, for the first time, the iterative scheduling method to determine the R_i of a real-time task. Further works proposed different alternatives to this method [4, 5, 6, 7, 8] and diverse mechanisms to resolve the method iteration [6, 7, 8, 9]. All of these alternatives are based on the search of a solution for (1).

$$t^{q+1} = C_i + \sum_{j=1}^{i-1} \left\lceil \frac{t^q}{T_j} \right\rceil C_j \quad \text{with } t^q \leq D, \quad (1)$$

In (1) $\lceil \cdot \rceil$ is the ceiling operator, and t^q is the instant t at iteration q .

In [6] is proposed the *Response Time Analysis (RTA)* mechanism that starts the iteration of τ_i with the initial seed value $t^0 = R_{i-1} + C_i$. The *Hyperplanes Exact Test (HET)* is proposed in [7] as a recursive mechanism to solve (1) and is improved in [10]. In [8, 9], the *RTA* mechanism is improved as alternatives *RTA2* and *RTA3*, respectively. The algorithm proposed in this paper is based in these methods to produce a new lower *CC* scheduling analysis method.

II. WORST CASE RESPONSE TIME CALCULUS.

The calculus of R_i takes into account the execution of C_i required by τ_i in addition to the execution of all the invocations of the higher priority tasks. The minimum t that solves (1) determines R_i .

Mathematically, a *Fixed Point (FP)* of a function f is defined as a value t such that $t = f(t)$. When f is a function of time, then the expressions point t and instant t are equivalent.

Equation (1) is solved by finding the first *Fixed Point (FP)* for each real-time task of the system. From [11], it is proved that a *FP* of (1) is an attractor and consequently (1) converges to it. Low *CC* real-time scheduling methods try to reduce the number of iterations and calculations required to converge to the first *FP* of (1).

A. Iterative Algorithm

In the following, the algorithms introduced in [3, 6] along the improvements proposed in [8] and [9] is presented.

The iterative algorithm proposed in [3] begins the iteration with the seed $t^0 = 0$. In [6], the *CC* of the algorithm is reduced by starting for τ_i , with seed $t^0 = R_{i-1} + C_i$. When the iterative

algorithm is applied to a schedulable τ_i , it will take $m+1$ iterations until the first *FP* is found (Kleene theorem [12]).

Next, it is shown the iterations required by the algorithm proposed in [3] to find the *FP* of (1) for an arbitrary task τ_i :

$$\text{Iteration } 0 \quad \left\lceil \frac{t^0}{T_1} \right\rceil C_1 + \left\lceil \frac{t^0}{T_2} \right\rceil C_2 + \dots + \left\lceil \frac{t^0}{T_{i-1}} \right\rceil C_{i-1} + C_i = t^1$$

$$\dots\dots\dots$$

$$\text{Iteration } m-1 \quad \left\lceil \frac{t^{m-1}}{T_1} \right\rceil C_1 + \left\lceil \frac{t^{m-1}}{T_2} \right\rceil C_2 + \dots + \left\lceil \frac{t^{m-1}}{T_{i-1}} \right\rceil C_{i-1} + C_i = t^m$$

$$\text{Iteration } m \quad \left\lceil \frac{t^m}{T_1} \right\rceil C_1 + \left\lceil \frac{t^m}{T_2} \right\rceil C_2 + \dots + \left\lceil \frac{t^m}{T_{i-1}} \right\rceil C_{i-1} + C_i = t^{m+1}$$

The first *FP* is reached in iteration $m-1$ with t^m , but is verified in iteration m when $t^m = t^{m+1}$.

The expression $\lceil t/T_j \rceil C_j$ calculates the workload that τ_j produce at interval $[0, l_j + D_j]$, where l_j is the last invocation instant of τ_j .

We define A_j^q as the workload of τ_j until the instant t^q at iteration q :

$$A_j^q = \left\lceil \frac{t^q}{T_j} \right\rceil C_j \quad \text{with } 0 \leq q \leq m \quad \text{and } 1 \leq j \leq i-1 \quad (2)$$

With the A_j^q definition, the previous iterations may be expressed as:

$$\text{Iteration } 0 \quad A_1^0 + A_2^0 + \dots + A_{i-1}^0 + C_i = t^1 \quad (3)$$

$$\dots\dots\dots$$

$$\text{Iteration } m-1 \quad A_1^{m-1} + A_2^{m-1} + \dots + A_{i-1}^{m-1} + C_i = t^m$$

$$\text{Iteration } m \quad A_1^m + A_2^m + \dots + A_{i-1}^m + C_i = t^{m+1}$$

Most of the iteration algorithms utilize the result produced in previous iterations as the seed t^m for the next one. Since the iteration stops when $t^m = t^{m+1}$, this happens when:

$$t^m = A_1^{m-1} + A_2^{m-1} + \dots + A_{i-1}^{m-1} + C_i = A_1^m + A_2^m + \dots + A_{i-1}^m + C_i = t^{m+1}$$

and consequently, this happens if and only if:

$$A_1^{m-1} = A_1^m, A_2^{m-1} = A_2^m, \dots, A_{i-1}^{m-1} = A_{i-1}^m$$

Using successive iterations q and $q+1$, (1) can be rewritten as:

$$t^{q+1} - C_i - \sum_{j=1}^{i-1} A_j^q = 0, \quad t^q - C_i - \sum_{j=1}^{i-1} A_j^{q-1} = 0; \Rightarrow \quad t^{q+1} = t^q + \sum_{j=1}^{i-1} A_j^q - A_j^{q-1} \quad (4)$$

In [8], it is proved that for any j , if $A_j^q \neq A_j^{q-1}$, it is possible to calculate A_{j+1}^q with $t^{q+1} = t^q + A_j^q - A_j^{q-1}$, where t^{q+1} is the new value of t^q for the same iteration. Consequently, it is possible to get a new seed t^{q+1} when $A_j^q \neq A_j^{q-1}$, before completing the current iteration. This is used in [8] to reduced the number of iterations and therefore reduce the *CC* of the algorithm. However, as the value of A_j^q should be stored, the memory required is increased with a linear order complexity ($O(n)$) which is acceptable for most real applications.

It is also proved in [8] that A_j^q does not change in its validity interval $(l_j, l_j + D_j] = (l_j, I_j^q]$ with:

$$I_j^q = \lceil t^q/T_j \rceil T_j = l_j + D_j$$

Then the value of A_j^q , and also the value of I_j^q , should only be updated when t^q is outside of its validity interval, $t^q > I_j^q$.

The following Theorem presented in [9], allows to finalize the R_i search for τ_i , if in iteration q , all I_j^q with $1 \leq j \leq i$ are greater than or equal to the final t^q .

Theorem 1:

For a given set of n tasks, if at the end of any iteration all I_j^q with $1 \leq j \leq i$, are greater than or equal to t^q ($I_j^q \geq t^q$) then the algorithm already found a FP at t^q .

III. RTA4 SCHEDULING ALGORITHM

In this section, the RTA4 scheduling algorithm is described. It is based on the scheduling algorithms RTA2 [8] and RTA3 [9]. In the following, two theorems and a lemma are presented, which the RTA4 algorithm uses.

The Theorem 2 allows to determine R_i of τ_i using the I_j^q of the last iteration where τ_{i-1} found its FP and with $I_i^0 = D_i$.

Theorem 2:

The R_i for any τ_i is $R_i = t^0 = R_{i-1} + C_i$ if all I_j^q are greater or equal than the seed t^0 of τ_i ($I_j^q \geq t^0 = R_{i-1} + C_i$ with $1 \leq j \leq i$) and it is a fixed point.

Proof:

If $I_j^q \geq t^0 = R_{i-1} + C_i$, for all I_j^q in the FP of τ_{i-1} , with $1 \leq j \leq i-1$, and also $I_i^0 \geq R_{i-1} + C_i$, then no A_j^0 will change for equation (3) and consequently $R_i = t^0 = R_{i-1} + C_i$ is a FP for τ_i . \square

According to Theorem 2, the schedulability of τ_i may be determined without any iterations and consequently the CC of the scheduling analysis is reduced.

The remaining lemma and theorem allow A_j^q to be calculated in a new way, which reduces the CC under certain conditions, as explained below.

If for any iteration q it is found that $A_j^q \neq A_j^{q-1}$, then the workload of τ_i changes at t^q and consequently (1) does not stop at the iteration $q+1$. Note nevertheless that by Theorem 1 in RTA3 this may not be the case.

In addition, the inspection point is increased for the following task as $t^{q+1} = t^q + A_j^q - A_j^{q-1}$. It is possible then that τ_j generates new workload in the interval (t^q, t^{q+1}) , and then the workload A_j^q calculated at t^{q+1} is greater than A_j^q calculated at t^q . If this is the case, it is known at iteration q , that in the iteration $q+1$ there is an increase in the workload of τ_j (A_j^{q+1}) and consequently there is no FP in it. Therefore, it is convenient to iterate over τ_j , until the workload of τ_j is the same for t^q and t^{q+1} . Let p be the index of the iteration over τ_j , in the general iteration q , such that $A_j^{q,p}$ is the workload at $t^{q,p}$. The iteration to find a new local FP for τ_j is:

$$t^{q,p+1} = t^{q,p} + A_j^{q,p} - A_j^{q,p-1} \quad (5)$$

where $t^{q,0} = t^q$, $A_j^{q,p} = \left\lceil \frac{t^{q,p}}{T_j} \right\rceil C_j$ for $p > 0$ and for $p = 0$, $A_j^{q,-1} = A_j^{q-1}$

$$t^{q,1} = t^{q,0} + A_j^{q,0} - A_j^{q,-1} = t^q + A_j^q - A_j^{q-1} \quad (6)$$

The following Lemma proves that the local FP can be calculated without the need to iterate over p .

Lemma 1:

The recurrence $t^{q,p+1} = t^{q,p} + A_j^{q,p} - A_j^{q,p-1}$ converges to a FP $t^q *$ that maximizes $A_j^{q,p}$ with:

$$t^q * = \left\lceil \frac{t^q - A_j^{q-1}}{T_j - C_j} \right\rceil C_j + t^q - A_j^{q-1}$$

where $\left\lceil \frac{t^q - A_j^{q-1}}{T_j - C_j} \right\rceil C_j = \left\lceil \frac{t^q *}{T_j} \right\rceil C_j$ is the workload of τ_j at $t^q *$.

Proof:

Expression (5) is a second degree recurrence equation since the value of $t^{q,p+1}$ depends on two previous values of p . Let first apply the induction principle to show that (5) is equivalent to

$$t^{q,p+1} = t^q + A_j^{q,p} - A_j^{q-1} \quad (7)$$

This is a first-degree recurrence equation since t^q and A_j^{q-1} are constants that do not depend on p . In fact, we have already observed in (6) that (7) is valid for $p = 0$. Assuming that (7) is valid for $p = k$, we would have $t^{q,k+1} = t^q + A_j^{q,k} - A_j^{q-1}$. If we now calculate $t^{q,k+2}$ according to (5) we have:

$$t^{q,k+2} = t^{q,k+1} + A_j^{q,k+1} - A_j^{q,k} = t^q + A_j^{q,k} - A_j^{q-1} + A_j^{q,k+1} - A_j^{q,k} = t^q + A_j^{q,k+1} - A_j^{q-1}$$

then (7) is valid also for $p = k+1$ and it follows from induction that (7) is valid for all $p \geq 0$. In order to find a FP of (7) observe that the values of $t^{q,p}$ are increasing and converge to the point $t^q *$ where the step function $\left\lceil t/T_j \right\rceil C_j + t^q - A_j^{q-1}$ cuts off the identity line t . To calculate this point considers the intersection of the line $(C_j/T_j)t + t^q - A_j^{q-1}$ with the identity, which corresponds to (Figure 1):

$$t^1 = T_j \left(\frac{t^q - A_j^{q-1}}{T_j - C_j} \right)$$

Note then that:

$$\left\lceil \frac{t^1}{T_j} \right\rceil C_j + t^q - A_j^{q-1} = \left\lceil \frac{t^q *}{T_j} \right\rceil C_j + t^q - A_j^{q-1} = t^q *$$

And the lemma follows from these equalities. \square

The next theorem shows how to calculate the inspection point t^{q+} maximizing the increase in the workload of τ_j

Theorem 3:

In the iteration q , the value t^{q+} due to the maximum increase in the workload τ_j at t^q can be calculated as:

$$t^{q+} = t^q + A_j^{q *} - A_j^{q-1} \quad (8)$$

where $A_j^{q *} = \left\lceil \frac{t^q - A_j^{q-1} *}{T_j - C_j} \right\rceil C_j$

Proof:

The value of t^{q+} in (8) corresponds to $t^q *$ in Lemma

1, when the workload of τ_j in iteration $q-1$ was calculated as A_j^{q-1} instead of A_j^q . From Lemma 1 it also follows that:

$$A_j^{q*} = \left\lceil \frac{t^{q*}}{T_j} \right\rceil C_j = \left\lceil \frac{t^{q*}}{T_j} \right\rceil C_j$$

This is the workload of τ_j until the instant t^{q*} on iteration q . \square

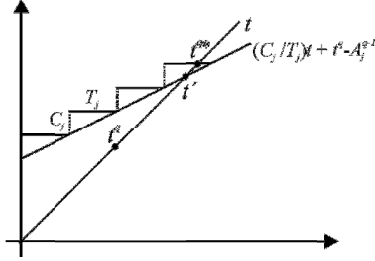


Fig. 1. Lemma 1 / Theorem 3 (workload vs. time)

A. RTA4 Algorithm

At the beginning, A_j and I_j values for all the tasks are initialized as $A_j = C_j$ and $I_j = T_j$. Then, the algorithm continues iterating for all tasks according to the mechanism showed in the following pseudo-code:

```

function RTA4( rts )
    t+ = R1 = C1
    min =  $\begin{cases} \text{if } RM \Rightarrow I_1 \\ \text{if } DM \Rightarrow \min_{i=1}^n I_i \end{cases}$ 
    for i=2 to n
        t+ = t+ + Ci
        while t+ > min
            min = Ii
            for j=i-1 to 1
                if t+ > Ij then
                    adiff = t+ - Aj
                    tmp =  $\lceil a_{diff} / (T_j - C_j) \rceil$ 
                    Aj = tmp * Cj
                    Ij = tmp * Tj
                    t+ = Aj + adiff
                if t+ > Di then return False
            end if
            if min > Ij then min = Ij
        end for
        end while
        Ri = t+
    end for
    return True
end function
    
```

B. Example

In this section the scheduling analysis of the RTS $S(5) = \{(5,27,27), (2,52,52), (2,64,64), (60,248,248), (191,976,976)\}$ using the RTA4 algorithm is presented.

If $I_j^q \neq I_j^{q-1}$ the notation *a.* is used, else the notation *b.* is employed. In addition, when one of the two theorems is applied, the notation *c.* will be used.

a. If $I_j^q \neq I_j^{q-1} \Rightarrow \left\lceil \frac{t^q - A_j^{q-1}}{T_j - C_j} \right\rceil C_j$, b. If $I_j^q = I_j^{q-1} \Rightarrow (A_j^q; I_j^q) = (A_j^{q-1}; I_j^{q-1})$, c. \Rightarrow

RTA4

```

τ1  i = 1; if C1 < D1 ⇒ R1 = C1
    q = 0; t0 = R1-1 + C1; A10 = C1; I10 = D1
    For i=2
        Until Ijq > tq+1
            For j=i-1
                tq+ = tq +  $\left\lceil \frac{t^q - A_j^{q-1}}{T_j - C_j} \right\rceil C_j - A_j^{q-1}$ 
            end For
            q = q + 1
        end Until
    end For
τ1  C1 = 5 < D1 = 27 ⇒ R1 = 5
τ2  q = 0, t0 = 5 + 2 = 7, θ(Ajq; Ijq) = {(2;52), (5;27)}, min Ijq = 27 ⇒ R2 = 7
τ3  q = 0, t0 = 7 + 2 = 9, θ(Ajq; Ijq) = {(2;64), (2;52), (5;27)},
    min Ijq = 27 ⇒ R3 = 9
τ4  q = 0, t0 = 9 + 60 = 69, θ(Ajq; Ijq) = {(60;248), (2;64), (2;52), (5;27)}
    q = 0
    t+ = 69 +  $\left\lceil \frac{4,128}{64-2} \right\rceil 2 - 2 = 71$ ; t+ = 71 +  $\left\lceil \frac{4,104}{52-2} \right\rceil 2 - 2 = 73$ ;
    t+ = 73 +  $\left\lceil \frac{20,108}{27-5} \right\rceil 5 - 5 = 88$ ; min Ijq = 104 ⇒ R4 = 88
τ5  q = 0, t0 = 88 + 191 = 279,
    θ(Ajq; Ijq) = {(191;976), (60;248), (4;128), (104;52), (20;108)};
    q = 0
    t+ = 279 +  $\left\lceil \frac{120,496}{248-60} \right\rceil 60 - 60 = 339$ ; t+ = 339 +  $\left\lceil \frac{12,384}{64-2} \right\rceil 2 - 4 = 347$ 
    t+ = 347 +  $\left\lceil \frac{14,364}{52-2} \right\rceil 2 - 4 = 357$ ; t+ = 357 +  $\left\lceil \frac{80,432}{27-5} \right\rceil 5 - 20 = 417$ 
    q = 1
    t+ = (120;496) = 417; t+ = 417 +  $\left\lceil \frac{14,448}{64-2} \right\rceil 2 - 12 = 419$ 
    t+ = 419 +  $\left\lceil \frac{18,468}{52-2} \right\rceil 2 - 14 = 423$ ; t+ = (80;432) = 423
    min Ijq = 432 ⇒ R5 = 423
    
```

The RTA4 algorithm works in a similar way as RTA3, except for the calculation of A_j^q . However, it performs fewer calculations of ceil/floor operations and loops. The results are $RTA = 32$, $RTA2 = 28$, $RTA3 = 12$ and $RTA4 = 9$ ceils/floors, which can be found in the annex.

IV. EVALUATION

In this section, the performance of the RTA4 algorithm is compared against RTA, RTA2, RTA3 and HET2 (HET with the modification introduced in [10]). Two comparison criteria were utilized. The first is the time elapsed in microseconds that the algorithm takes to analyze the schedulability. The second is the number of ceil/floor operations performed which is the invariant of the algorithms and is used as a CC metric.

The algorithms were implemented using the C programming language, and compiled with GCC ARM 5.3.1

with *-Os* optimization level. Then, they were executed in a *mbed* NXP LPC1768 (ARM Cortex-M3 at 96 Mhz) board.

As an integer data type was used for the tasks parameters, the macro $(x/y)+(x \bmod y \neq 0)$ was used to calculate ceilings and floors, rather than the C library functions. This resulted in a significant reduction of the temporal *CC* by three orders of magnitude.

A set of *RTS* of 10, 20, 50 and 100 tasks were randomly generated. The *U* was categorized in 70%, 75%, and 80% to 100% with 2% steps. The periods of the tasks were uniformly distributed in 4 groups of 25-1000, 25-10000, 25-100000 and 25-1000000. For each group, 10000 *RTS* were generated per *U*, with deadlines set equal to periods ($T_i = D_i$).

Due to space constrains, only the μs measurement results for *RTS* of the 25-10000 group and the number of ceil/floor operations for *RTS* with 100 tasks are presented. All the results can be found in the annex, at <http://www.rtsg.unp.edu.ar>.

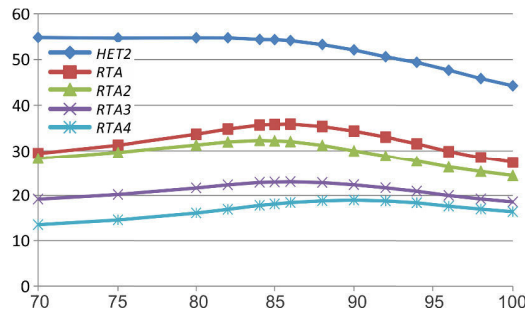


Fig. 2. 10 Tasks, *DU* 25-10000 (μs vs. *U*)

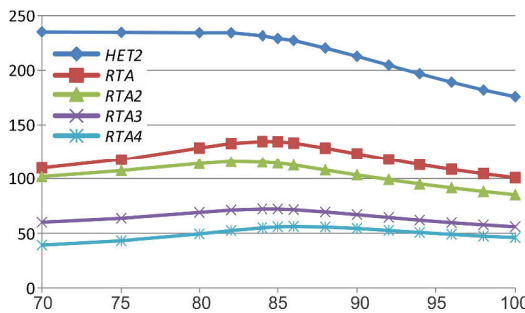


Fig. 3. 20 Tasks, *DU* 25-10000 (μs vs. *U*)

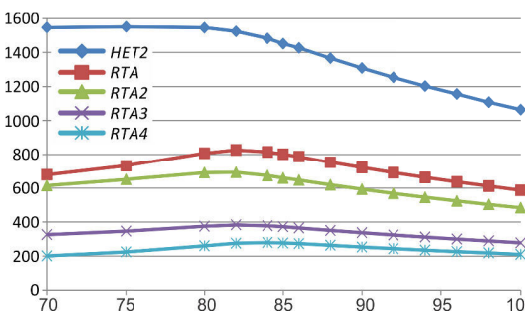


Fig. 4. 50 Tasks, *DU* 25-10000 (μs vs. *U*)

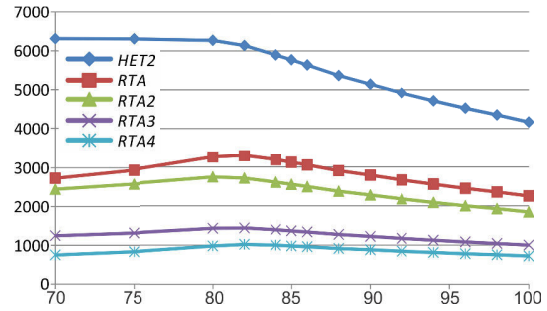


Fig. 5. 100 Tasks, *DU* 25-10000 (μs vs. *U*)

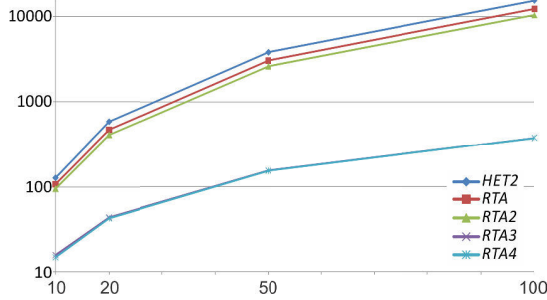


Fig. 6. 100 Tasks, *DU* 25-10000 (invariants vs. number of tasks) $U=82\%$

V. RESULTS AND ANALYSIS

Data distributions of the results were similar for the different groups of *RTS* evaluated.

Figures 2-5 shows the average measured time in μs required for the scheduling algorithms to determine the schedulability of a *RTS* through different total utilization factors.

Figure 6 shows the average *invariants* number required for each algorithm to determine the schedulability for $U=82\%$. It can be noted that the *CC* measured for most of the methods increased two orders of magnitude (from 100 to 10000) when the number of tasks increases from 10 to 100. On the other hand, *RTA4* produces the lowest *CC* among the algorithms evaluated.

Next, an asymptotic analysis of the temporal *CC* is presented, in order to find an average computational order for the algorithms. Based on the sets of 10 tasks, it is determined whether the average growth of the temporal *CC* of the evaluated algorithms follows a pattern of known computational order.

The analysis is performed for a $U = 82\%$. Then, let P_n be the temporal *CC* of the set of n tasks (with $n = 20, 50, 100$) and P_{10} the temporal *CC* of the set of 10 tasks.

If the algorithms follow a growth of order $O(n \cdot \log(n))$ or $O(n^2)$, the result can be expressed as a function of n , where k is a constant:

$$P_n/P_{10} = k \cdot n \cdot \log(n) \quad \text{or} \quad P_n/P_{10} = k \cdot n^2$$

TABLE 1 - ASYMPTOTIC ANALYSIS ($U=82\%$) - TEMPORAL CC

Methods Sets	RTA		RTA2		RTA3		RTA4		HET2	
	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}
DU 25-10000 20	0,14713	0,00957	0,14041	0,00913	0,12309	0,00801	0,12043	0,00783	0,16491	0,01073
DU 25-10000 50	0,28013	0,00952	0,25816	0,00877	0,20340	0,00691	0,19406	0,00659	0,32872	0,01117
DU 25-10000 100	0,47426	0,00949	0,42931	0,00859	0,32416	0,00648	0,30402	0,00608	0,56221	0,01124

TABLE 2 - ASYMPTOTIC ANALYSIS ($U=82\%$) - CC IN INVARIANTS

Methods Sets	RTA		RTA2		RTA3		RTA4		HET2	
	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}	k_{\log}	k_{n^2}
DU 25-10000 20	0,16495	0,01073	0,16103	0,01048	0,10327	0,00672	0,10587	0,00689	0,17121	0,01114
DU 25-10000 50	0,32979	0,01121	0,31391	0,01067	0,10208	0,00347	0,10549	0,00358	0,34255	0,01164
DU 25-10000 100	0,57152	0,01143	0,53345	0,01067	0,08979	0,00180	0,09294	0,00186	0,57737	0,01155

If the value of k is cleared, it must be kept constant for sets of 20, 50 and 100 tasks to be able to assert that the average computational order is bounded.

$$k_{\log} = P_n / (P_{10} \cdot n \cdot \log(n)) \quad \text{or} \quad k_{n^2} = P_n / (P_{10} \cdot n^2).$$

Table 1 presents the results of the asymptotic analysis for the temporal CC. It can be observed that RTA and HET2 methods follow very closely an $O(n^2)$. The algorithms RTA2, RTA3 and RTA4, do not fit either function accurately. However, it is clear that it is greater than $O(n \cdot \log(n))$ and less than $O(n^2)$, especially for RTA4.

Table 2 presents the results of the asymptotic analysis for the CC in invariants. It can be observed that RTA, RTA2 and HET2 follows an $O(n^2)$ with very good precision. The RTA3 and RTA4 algorithms, conform to an $O(n \cdot \log(n))$.

It is observed that the CC in invariants between RTA3 and RTA4 is practically the same. However, the temporal CC is smaller, because RTA4 needs to perform a smaller amount of *for* and *while loops* than RTA3.

VI. CONCLUSIONS AND FUTURE WORKS

The RTA4 scheduling algorithm is proposed as a low CC algorithm for schedulability analysis. The mechanism was implemented on an embedded system and evaluated through several sets of randomly generated RTS. The performance was compared against the most referenced algorithms in the real-time literature. Results showed that RTA4 outperforms them for different number of tasks and different total utilization factors. Moreover, the complexity of RTA4 increases at a lower rate than the others algorithms as the number of tasks increases. Future works will extend the proposed algorithm for accounting *Jitter*, *Offset* and *Blocking* times, use RTA4 as a service on several RTOS and apply the improvements in the iterative process into others algorithms that use similar iterative methods.

REFERENCES

- [1] J. A. Stankovic, "Misconceptions About Real-Time Computing: A Serious Problem for Next-Generations Systems," *IEEE Computer*, vol. Octubre, pp. 10-19, 1988.
- [2] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment," *Journal of the ACM*, vol. 20, pp. 46-61, 1973.
- [3] M. Joseph and P. Pandya, "Finding Response Times in Real-Time System," *The Computer Journal (British Computer Society)*, vol. 29, pp. 390-395, 1986.
- [4] J. P. Lehoczky, L. Sha, and J. K. Strosnider, "Enhanced Aperiodic Responsiveness in Hard Real-Time Environments," in *IEEE Real-Time Systems Symposium*, 1987, pp. 261-270.
- [5] N. C. Audsley, A. Burns, M. F. Richardson, K. Tindell, and A. J. Wellings, "Applying New Scheduling Theory to Static Priority Preemptive Scheduling," *Software Engineering Journal*, vol. 8, pp. 284-292, 1993.
- [6] M. Sjödin and H. Hansson, "Improved Response-Time Analysis Calculations," in *IEEE 19th Real-Time Systems Symp.*, 1998, pp. 399-409.
- [7] E. Bini and C. B. Giorgio, "Schedulability Analysis of Periodic Fixed Priority Systems," *IEEE Trans. on Computers*, vol. 53, pp. 1462-1473, November 2004.
- [8] J. M. Urriza, J. D. Orozco, R. Cayssials, and L. Schorb, "Reduced Computational Cost in the Calculation of Worst Case Response Time for Real Time Systems," *Journal of Computer Science & Technology*, vol. 9, pp. 72-81, 2009.
- [9] J. M. Urriza, F. E. Paez, J. D. Orozco, and R. Casysials, "Computational Cost Reduction for Real-Time Schedulability Tests Algorithms," *IEEE Latin America Transactions*, vol. 13, pp. 3714-3723, 2015.
- [10] R. I. Davis, A. Zabus, and A. Burns, "Efficient Exact Schedulability Tests for Fixed Priority Real-Time Systems," *IEEE Transactions on Computers*, vol. 57, pp. 1261-1276, 2008.
- [11] J. M. Urriza, R. Cayssials, and J. D. Orozco, "Modelado de Sistemas de Tiempo Real Planificados por RM o DM: Caracterización y Análisis," in *XXXIV Conferencia Latinoamericana de Informática, CLEI 2008*, Santa Fe, Argentina, 2008, pp. 1435-1444.
- [12] Z. Manna, S. Ness, and J. Vuillemin, "Inductive Methods for Proving Properties of Programs," *Communications of the ACM*, vol. 16, pp. 491-502, August 1973.
- [13] L. Sha, R. Rajkumar, and J. P. Lehoczky, "Priority Inheritance Protocols: An Approach to Real-Time Synchronization," *IEEE Transactions on Computers*, vol. 39, pp. 1175-1185, 1990.

FPGA Quantum Computing Emulator Using High Level Design Tools

Agustin Silva, Omar Gustavo Zabaleta

Instituto de Investigaciones Científicas y Tecnológicas en Electrónica (ICYTE),

Facultad de Ingeniería Electrónica

Universidad Nacional de Mar del Plata

Buenos Aires, Mar del Plata (+54 223 4816600)

Email: agustinsilva447@gmail.com

Abstract—Although, there have been considerable advances in quantum technology, a great scale quantum computer will not be available until about a few years. In the meantime, it requires quantum emulators which allow to study the existing quantum algorithms and the design of new ones. We propose a flexible method to emulate quantum circuits which reduces design and processing time. In this context, we propose a design framework that takes advantage of field-programmable gate array (FPGA) parallel processing technology in combination with high-level programming tools provided by Vivado[®]. In order to show the design method working Quantum Fourier Transform (QFT) is implemented. Every step of the design is described and circuit performance is studied for different cases of input vector data sizes.

1. Introduction

Quantum computers promise to solve problems that would otherwise be intractable with conventional computers [1]. Their practical importance arise from the exponential speedup in computation of certain algorithmic tasks over classical computation [2]. Therefore, they stopped being interesting for physicists and mathematicians only but also for biologists [3], economists [4], engineers and many other disciplines in the research world today [5], [6], [7], [8].

Quantum superposition and quantum entanglement are the main features that quantum computing takes advantage overcoming the classic performance. Such properties of quantum computers are clearly a great advantage, but a significant challenge to put into practice at large scale. The main quantum computing drawback is quantum decoherence. The coherence state, fundamental to a quantum computers operation, is destroyed when it is affected by environment. As a consequence, the physical requirements of manipulating a system at quantum scale are considerable, touching on the realms of superconductor, nanotechnology, and quantum electronics, as well as others. Although many technology leaders, see for example [9], are putting a lot of energy and quantum computers have come closer to reality, it is necessary to wait a little longer to get a physical quantum computer able to replace a classic computer in performing important calculations.

The construction of efficient algorithms for solving classically unmanageable problems is a sensible and difficult task in the field of quantum computing. Moreover, in the absence of a physical quantum computer, an alternative technology is necessary to test performance and error sensitivity of quantum circuits, and where to implement quantum heuristic algorithms designed. Many approaches have been done in that direction, each of them with advantages and disadvantages, [10], [11], [12]. The principal issue to deal with is quantum parallelism, superposition an entanglement, which is not possible via software simulation because of sequential nature of software-based simulations. On the other hand, hardware emulators can mimic quantum parallelism more closely but at expense of exponential growth resources, so, it is essential in this case the right choice of the hardware architecture.

The best way to describe quantum algorithms is by means of quantum circuits. In the same way that conventional circuits, quantum circuits consist of gates connected by channels that carry information from one place to another [13]. Moreover, quantum gates operations are represented as unitary matrices which act on the quantum states. Because quantum states grow exponentially as the number of quantum bits (qubits) increase, it is a great challenge to yield an efficient technique to mimic quantum operators on classic hardware.

In this paper we propose a quantum circuit emulator that focuses on quantum algorithms designers that can harness FPGAs parallel computation capacities with minimal or no knowledges of FPGA architecture. In this framework, high-level design tools, which can design the application at system-level with high-level programming languages, are promoted by FPGA vendors, such as Vivado High Level Synthesis (HLS), provided by Xilinx, Altera SDK for OpenCL, Stratus HLS by Cadence, Symphony C Compiler by Synopsys, etc., see [14] and references therein. In this case we have chosen Xilinx Vivado[®] design suite [15], which allow to synthesise C/C+ code to Xilinx FPGAs, to design a quantum circuits emulator. Thus, because of its role in some of the more important algorithms of quantum computation such as Shor factorization, phase finding and discrete logarithm, the QFT circuit is chosen to test the

quantum emulator we are presenting.

All measures are made working with the ZedBoard Zynq-7000 ARM/FPGA SoC Development Board from Xilinx which works with a System of chip model XC7Z020-CLG484-1 and a microprocessor Dual-core ARM CortexTM-A9.

The remainder of the paper is organized as follows: Section 2 gives a brief overview of quantum mechanics and computation. In Section 3 the QFT model and its implementation on FPGA. In Section 4 the design performance is analysed and results are compared with the same QFT algorithm implemented on the board embedded microprocessor. Finally, Section 5 concludes the paper.

2. Theoretical Background

In this section we present some fundamentals of quantum theory and quantum computation.

2.1. Quantum Computation

As the *bit* is the minimum information in classical computation, the *quantum bit* or *qubit* is the minimum information for quantum computation and quantum information. The qubit is typically a microscopic system, such as an atom or polarized photon. The Boolean states 0 and 1 are represented by a fixed pair of distinguishable states of the qubit (for example, electron states: spin up $|0\rangle = \uparrow$ and spin down $|1\rangle = \downarrow$) where “ $| \ \rangle$ ” is Dirac notation [16]. Qubits can also exist in a continuum of intermediate states, or “superpositions”, represented mathematically as unit vectors in a two-dimensional complex vector space (the “Hilbert space”) spanned by the basis vectors,

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Then the qubit state is mathematically represented by a linear superposition:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad (1)$$

where $|\alpha|^2$ and $|\beta|^2$ are the probabilities that state ψ be $|0\rangle$ or $|1\rangle$ respectively after measurement and thus the normalization condition $|\alpha|^2 + |\beta|^2 = 1$ must be satisfied.

A n qubits system may be in 2^n basis states, as well as all possible superpositions of them. Thus, states of a pair of qubits, for example, lie in a four-dimensional Hilbert space $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. When state $|\psi^{\otimes n}\rangle$ is separable, it can be written as

$$|\psi^{\otimes n}\rangle = |\psi\rangle_0 \otimes |\psi\rangle_1 \otimes \dots \otimes |\psi\rangle_{n-1}. \quad (2)$$

where \otimes is the Kronecker product.

Quantum states evolution is determined by

$$|\psi_2\rangle = U|\psi_1\rangle, \quad (3)$$

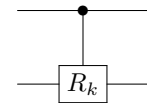
where U are unit operators (a necessary condition to satisfy quantum computation reversibility) in a Hilbert space. See

[16], [17] for a complete introduction to quantum computation and quantum information.

2.1.1. Gates. As in classical computation, quantum computation represents evolution operators by quantum gates. Then quantum gates connected by wires or free space conform quantum circuits. Some relevant quantum gates to the QFT circuit are described below. An important transformation used in quantum information processing is performed by the Hadamard gate (H). It is a single-qubit gate that performs the following transformations

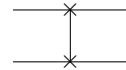
$$|0\rangle \xrightarrow{H} \frac{|0\rangle + |1\rangle}{\sqrt{2}} \quad \text{and} \quad |1\rangle \xrightarrow{H} \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Phase-Shift Control: A 2-qubits gate where one qubit is the control and the other is the target. The target phase shifts only if the control state is 1. Then the followings are the matrix and the circuital symbol of the Phase-Shift Control gate.



$$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Swap: The swap gate is a 2-input/2-output gate that simply exchanges the bit values it is handed. The matrix and the circuital symbol of the Swap gate are:



$$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.2. Quantum Fourier Transform (QFT)

The Fourier Transform is a transformation used to convert a signal from time domain to frequency domain being very important for a great variety of engineering processes from spectroscopy to telecommunications. Its discrete version, the so-called discrete Fourier transform (DFT), has a computationally very efficient implementation in the form of the fast Fourier transform (FFT).

The Quantum Fourier Transform is analogous to the classical FFT and by exploiting the advantages of quantum parallelism can be executed in considerable less time when it is run on a quantum computer.

Let's define the input state vector $|\Psi\rangle$ as:

$$|\Psi\rangle = \frac{1}{\sqrt{2^n}} * \sum_{j=0}^{2^n-1} f(j\Delta t) * |j\rangle \quad (4)$$

where $f(j\Delta t)$, are the states probability amplitudes. Then, the QFT on $|\psi\rangle$ is defined by the following expression:

$$QFT \rightarrow \frac{1}{\sqrt{2^n}} * \sum_{k=0}^{2^n-1} \sum_{j=0}^{2^n-1} f(j\Delta t) * e^{2\pi i(\frac{jk}{2^n})} * |j\rangle \quad (5)$$

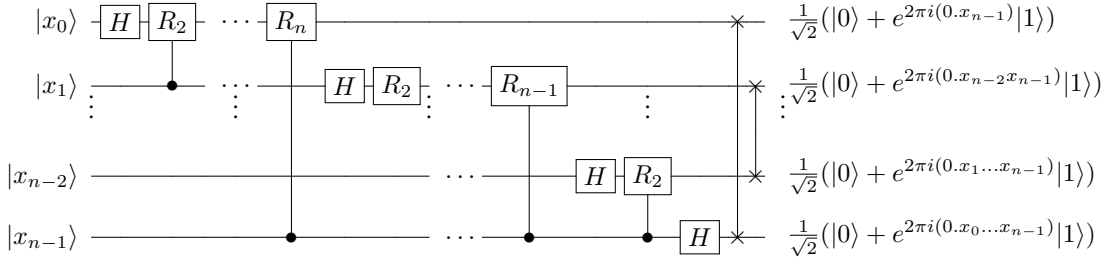


Figure 1. N-qubits QFT circuit

By rearranging the equation appropriately [18], it is possible to build the QFT circuit, see figure 1, as a function of the quantum gates introduced in section 2:

$$QFT_{2^n} |j\rangle = \frac{1}{\sqrt{2^n}} (|0\rangle + e^{2\pi i 0 \cdot j_n} |1\rangle) (|0\rangle + e^{2\pi i 0 \cdot j_{n-1} j_n} |1\rangle) \dots (|0\rangle + e^{2\pi i 0 \cdot j_1 j_2 \dots j_n} |1\rangle), \quad (6)$$

$$O = (000 \dots 0)^T \begin{matrix} I_N \\ \downarrow \\ QFT_N \\ \downarrow \\ O = I_N \cdot M^{QFT} \end{matrix}$$

$$M^{QFT} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & w & w^2 & \dots & w^{N-1} \\ 1 & w^2 & w^4 & \dots & w^{2(N-1)} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & w^{N-1} & w^{2(N-1)} & \dots & w^{(N-1)(N-1)} \end{bmatrix}$$

$$w = e^{\frac{2\pi i}{N}}$$

Figure 2. Block representation of QFT IPCore

3. FPGA implementation

Despite many quantum algorithms run exponentially faster than their classic counterpart, this advantage cannot be exploited in any classic computer. However, it is possible to mimic quantum behaviour in order to infer and also detect possible system malfunctions errors. Eventually, quantum emulation opens up the possibility of designing circuits before quantum computers be possible.

Implementation of circuits on pure FPGA-based platform are tasks usually performed by engineers because low-level languages are used and hardware details that have to be handled. On the other hand, reconfigurability and parallel processing have become FPGA a frequently used tool in quantum computing research. However, most quantum computing research is done with High Level language software simulations instead of low level languages such as VHDL or Verilog.

In that context we study the possibility to implement a Quantum Emulator on FPGA by designing at high level

using Xilinx Vivado IDE and present the complete analysis, advantages and disadvantages, of the QFT circuit design as an example. Furthermore, in what follows each programming stage is explained.

Vivado IDE includes three programs:

- Vivado HLS
- Vivado 2016.4
- Vivado SDK

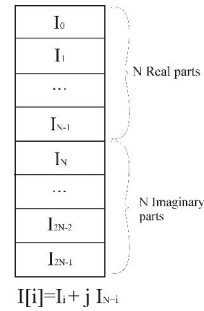


Figure 3. Real and Imaginary parts of elements from the complex input vector.

3.1. Vivado HLS

Figure 2 shows a block representation of IPCore (Intellectual Property Core) designed and in Fig. 4 the source code implemented in Vivado HLS.

At this stage the IPCore is generated. The complete circuit described by quantum gates presented in Fig. 1 can be satisfactorily represented by a matrix. The whole point of the IPCore is multiplied the input vector with this matrix. This code is written in C++ language and directives that define how to convert it into RTL code. Vivado HLS also provide a test bench platform for C++ program that allows to setting input parameters for example.

The implemented code contemplates the QFT as a single block instead of a set of gates. The main reason is to use a smaller amount of space in the FPGA and emulate transformations of greater input vectors. With that in mind, the Quantum Fourier Transform is taken as the matrix product between the input state vector and the matrix which is the equivalent of the set of all quantum gates of the circuit. It should be noted that the fact of programming at high level

```

#include <hls_math.h>

#define PI 3.1416
#define M 16 // Number of states

void VQFTAXIBUS(float E[2*M], float S[2*M])
{
#pragma HLS INTERFACE s_axilite port=return bundle=CTRL_BUS
// Control directives
#pragma HLS RESOURCE variable=E core=RAM_1P_BRAM
// Storage directive
#pragma HLS INTERFACE bram port=E
// Storage directive
#pragma HLS RESOURCE variable=S core=RAM_1P_BRAM
// Storage directive
#pragma HLS INTERFACE bram port=S
// Storage directive
    int j,k;
    float m,n;
    for (j=0;j<M;j++)
    {
#pragma HLS UNROLL
        // Looping directive
        S[j]=0;
        S[j+M]=0;
        for (k=0;k<M;k++)
        {
#pragma HLS UNROLL
            // Looping directive
            m = hls::cos(2*PI*j*k/M)/hls::sqrt(M);
            n = hls::sin(2*PI*j*k/M)/hls::sqrt(M);
            S[j] =S[j]+(m)*E[k]-(n)*E[k+M];
            S[j+M]=S[j+M]+(m)*E[k+M]+(n)*E[k];
        }
    }
}

```

Figure 4. Source code implemented in Vivado HLS showing the directives applied.

allows a flexible code. Therefore, different QFT gate sizes (1QFT, 2QFT, 3QFT, etc) can be analysed by setting only a parameter corresponding to the number of input qubits.

Input vectors contain N complex elements, which are represented in a single vector in order to optimize RAM resources. Furthermore it is not optimal to create two vectors one for real parts and another for imaginary parts. Data disposition scheme is displayed in fig. 3.

Then, a single vector is enough to represent both real and imaginary parts.

3.1.1. Directives. Perhaps directives are the most important concept in Vivado, because the objective oriented design differs with different directives chosen. For example, a less FPGA resources or a high processing speed design. The last one is our design goal.

In this work three types of directives were used: communication protocols, storage mode and loops implementations. For the communication protocol, AXILITE was opted, the reasons of this choice will be developed in a forthcoming section.

In order to decrease significantly the processing time a RAM block inside the FPGA was used regarding the input and output vectors storage instead of using the external RAM in the development board.

Directives have also great relevance on loops implementations. After studying the different kinds of directives the 'unroll' method was chosen because it allows to perform

several 'for' cycles in parallel, so the algorithm is performed in the smallest number of clock cycles.

3.2. Vivado 2016.4

Once the creation of the desired IPCore is completed, it is necessary to make the connections to communicate the microprocessor (ZYNQ7000, in our case) and the IPCore.

3.2.1. AxiLite. The protocol chosen for the communication between ZYNQ7000 and the IPCore is AxiLite. The reason is mainly because the control bus that communicates the IP Core with the ZYNQ7000 works with a type of communication that handles small amount of data and it is not constantly used. The bus handles actions such as: notify that communication will start, finish, data will be sent, the program will wait until a task is complete, among others. Another reason why it was decided to use this protocol is that Vivado gives the option to auto-route an important part of connections while working with AxiLite avoiding a great amount of design time. As explained later, though some of the connections must be modified, the main work is usually made automatically by the system.

3.2.2. Connections. The most important part of the whole design is probably to make the connections to establish communication between the chip (ZYNQ7000) and the IPCore. Then, some considerations must be taken into account.

First, Vivado aforementioned auto-connections have some problems to be fixed manually. Automatic connections are made so that information from ZYNQ7000 is sent to the IPCore but not in the other direction. Therefore, in order to access to the processing results it is necessary to connect manually two blocks called AXI-BRAM-Controller, see Fig. 5.

Another consideration is that a block called AxiTimer will be added, which allows, with a high level of precision, to measure how long each of the different processes performed by both the ZYNQ7000 and the IPCore will take.

3.3. Vivado SDK

Vivado SDK is the last stage where the algorithms of the QFT are executed in both microprocessor and FPGA. With this program and using C++ the memory addresses of the vectors and the input values will be assigned.

To establish the input values in the FPGA, pointers must be created and assigned to the corresponding addresses in the RAMs located in the block diagram. After doing this, these RAMs can be written and read using those pointers.

3.3.1. Drivers. The IPCore drivers are automatically generated, however they must be initialized using the available function prototypes, which are not only useful to initialize the processor but also to control it.

3.3.2. Axi Timer. Using prototypes of functions generated in the driver of the AxiTimer block, a library was created to facilitate repeated measurements.

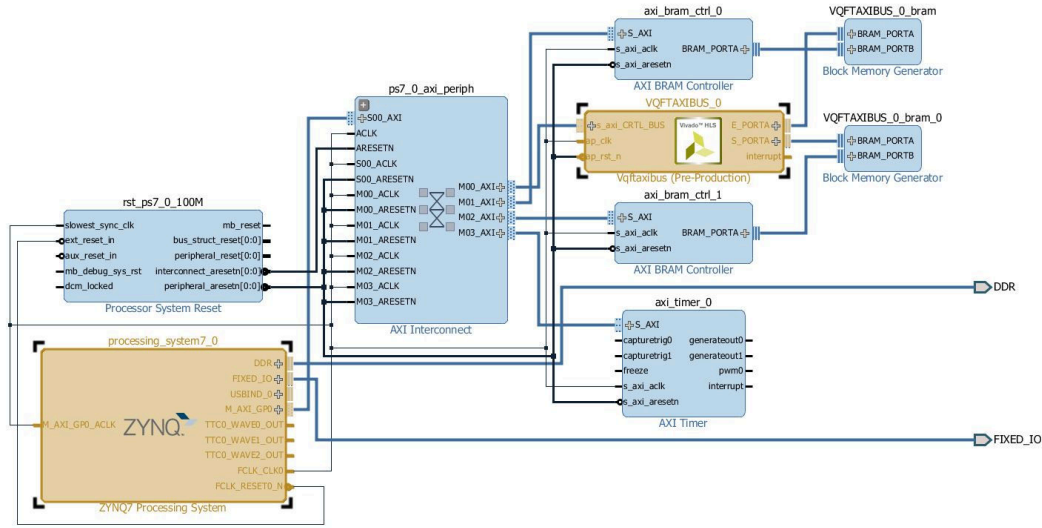


Figure 5. Vivado design block diagram

```

void XVqftaxibus_Start(XVqftaxibus *InstancePtr);
u32 XVqftaxibus_IsDone(XVqftaxibus *InstancePtr);
u32 XVqftaxibus_IsIdle(XVqftaxibus *InstancePtr);
u32 XVqftaxibus_IsReady(XVqftaxibus *InstancePtr);
void XVqftaxibus_EnableAutoRestart(XVqftaxibus *InstancePtr);
void XVqftaxibus_DisableAutoRestart(XVqftaxibus *InstancePtr);

void XVqftaxibus_InterruptGlobalEnable(XVqftaxibus *InstancePtr);
void XVqftaxibus_InterruptGlobalDisable(XVqftaxibus *InstancePtr);
void XVqftaxibus_InterruptEnable(XVqftaxibus *InstancePtr, u32 Mask);
void XVqftaxibus_InterruptDisable(XVqftaxibus *InstancePtr, u32 Mask);
void XVqftaxibus_InterruptClear(XVqftaxibus *InstancePtr, u32 Mask);
u32 XVqftaxibus_InterruptGetEnabled(XVqftaxibus *InstancePtr);
u32 XVqftaxibus_InterruptGetStatus(XVqftaxibus *InstancePtr);
    
```

Figure 6. Function Prototypes

TABLE 1. TIMES OBTAINED FOR THE DIFFERENT NQFT

	FPGA (μseg)	ZYNQ 7000 (μseg)
1QFT	1.17	6.18
2QFT	1.56	21.67
3QFT	2.35	101.53
4QFT	3.91	436.06
5QFT	8.78	1836.41
6QFT	115.47	7609.74

TABLE 2. LUT USAGE AS A FUNCTION OF QFT SIZE

	1QFT	2QFT	3QFT	4QFT	5QFT	6QFT
LUT(%)	9	18	39	55	175	399

4. Experimental Results

Table 1 shows simulations times of the NQFT (being N the number of input qubits) in the microprocessor (ZYNQ7000) and in the design in the FPGA.

The improvements in processing time are remarkable, this is due to the main advantage that FPGAs have over microcontrollers: its capacity to perform different tasks in parallel which is impossible in all devices that work sequentially, such as microcontrollers, always performing instruction by instruction. The processing times for the NQFT simulation show an exponential increase with N on the microprocessor while the corresponding increase is linear on the FPGA.

The improvements are obvious, the relationship between the processing time and the number of qubits tends to be linear rather than exponential, however this phenomenon is compensated by an exponential increase in the resources required in the FPGA. The number of qubits we can work with is limited because of the resources we have. To clarify, in table 2 is shown the percentage of LookUp-Tables (LUTs)

required according to the size of NQFT implemented.

Because it is not possible to implement 5QFT and 6QFT in the FPGA we work with, see table 2, their times were calculated from the parameters (latency, in clock cycles) given by simulation.

Figure 7 shows another consequence of this limitation. There it is possible to see a linear growing time as QFT size goes from 1 to 4 qubits, but such a linearity is broken when simulating 5 and 6 cases. Since there is not enough LookUp-Tables, the FPGA makes use of DSPs slowing down the processing times. Despite this drawback the processing time is still significantly lower than software simulations.

5. Conclusions

Parallel processing makes quantum emulation on FPGA more efficient than software simulation. A hardware system of that kind and easy to handle can be very useful for those quantum computing researchers that have minimal or no knowledges of HDL, but are familiar with a more common programming language such as C/C++ instead.

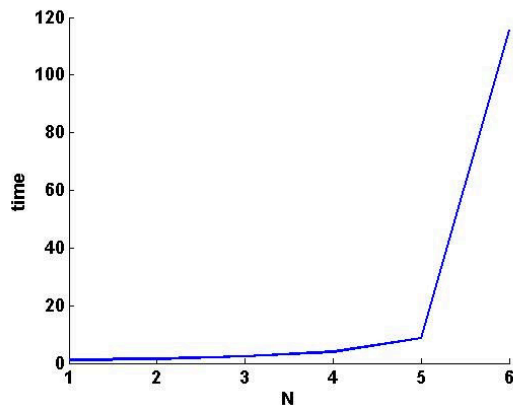


Figure 7. FPGA processing time (μseg) vs NQubits.

In this framework we addressed a quantum circuits emulation system that takes advantage of HLS tools provided by Vivado[®], that let the user to program the FPGA by starting from the desired algorithm written in C++ plus some specific directives. Furthermore, with the proposed methodology the designer can study from the more basic quantum IP core, a simple quantum gate, to a more complex algorithm performance, which is the QFT.

The emulator performance was analysed implementing one of the most important algorithms of quantum computing which is the n -qubits QFT for $n = 1$ to 6. The processing times obtained from the circuit running the algorithm on the FPGA were compared with the same algorithm running on the microprocessor, and it concludes that the FPGA parallelism properties were tapped efficiently. The maximum data size we can manage is clearly restricted by the FPGA capacity. In this case, the FPGA optimally places the 5-QFT circuit. However, though the system succeed in placing 6-QFT, the algorithm execution times are not as expected. This problem can be resolved easily working in other FPGA with more resource.

Finally, although emulators can improve performance in a remarkable way compared with software simulations, the main drawback of quantum emulation is the exponential increase in the amount of resources required.

Acknowledgement

This material is based upon work supported by the Joint ICTP-IAEA School on Hybrid Reconfigurable Devices for Scientific Instrumentation (smr2737) and the International Atomic Energy Agency (IAEA) Department of Nuclear Sciences and Applications under the Nuclear Instrumentation Programme (1.4.3.3).

References

[1] D. C. Marinescu and G. M. Marinescu, *Approaching Quantum Computing*. Upper Saddle River, NJ: Pearson Prentice Hall, 2005.

- [2] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, "Quantum algorithms revisited," *Proc. R. Soc. Lond.*, vol. 454, pp. 339–354, 1998.
- [3] S. Hagan, S. R. Hameroff, and J. A. Tuszyński, "Quantum computation in brain microtubules: Decoherence and biological feasibility," *Physical Review E*, vol. 65, no. 6, p. 061901, 2002.
- [4] E. W. Piotrowski and J. Stadkowski, "Quantum game theoretical frameworks in economics," in *The Palgrave Handbook of Quantum Models in Social Science*. Springer, 2017, pp. 39–57.
- [5] K. Kumar, N. A. Sharma, R. Prasad, A. Deo, M. T. Khorshed, M. Prasad, A. Dutt, and A. S. Ali, "A survey on quantum computing with main focus on the methods of implementation and commercialization gaps," in *Computer Science and Engineering (APWC on CSE), 2015 2nd Asia-Pacific World Congress on*. IEEE, 2015, pp. 1–7.
- [6] C. M. Arizmendi and O. G. Zabaleta, "Stability of couples in a quantum dating market," *special IJAMAS issue "Statistical Chaos and Complexity*, vol. 26, pp. 143–149, 2012.
- [7] O. G. Zabaleta and C. M. Arizmendi, "Quantum game techniques applied to wireless networks communications," *Journal of Advances in Applied and Computational Mathematics*, vol. 1, pp. 3–7, 2014.
- [8] O. Zabaleta, J. Barrangú, and C. Arizmendi, "Quantum game application to spectrum scarcity problems," *Physica A: Statistical Mechanics and its Applications*, vol. 466, pp. 455 – 461, 2017. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0378437116306793>
- [9] Paving the path to universal quantum computing. [Online]. Available: <https://www.ibm.com/blogs/think/2017/03/ibm-quantum/>
- [10] A. U. Khalid, Z. Zilic, and K. Radecka, "Fpga emulation of quantum circuits," in *Computer Design: VLSI in Computers and Processors, 2004. ICCD 2004. Proceedings. IEEE International Conference on*. IEEE, 2004, pp. 310–315.
- [11] M. Khalil-Hani, Y. Lee, and M. Marsono, "An accurate fpga-based hardware emulation on quantum fourier transform," *Quantum*, vol. 1, p. a1b3, 2015.
- [12] H. J. Garcia and I. L. Markov, "Simulation of quantum circuits via stabilizer frames," *IEEE Transactions on Computers*, vol. 64, no. 8, pp. 2323–2336, 2015.
- [13] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *Phys. Rev. A*, vol. 52, pp. 3457–3467, November 1995.
- [14] S. Qin and M. Berekovic, "A comparison of high-level design tools for soc-fpga on disparity map calculation example," *arXiv preprint arXiv:1509.00036*, 2015.
- [15] (2016) Vivado design suite user guide. [Online]. Available: www.xilinx.com/support/documentation/sw_manuals/xilinx2016_4/ug893-vivado-ide.pdf
- [16] P. R. Kaye, R. Laflamme, and M. Mosca, *An Introduction to Quantum Computing*. Oxford University Press, 2007.
- [17] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
- [18] S. Imre and F. Balazs, *Quantum Computing and Communications: an engineering approach*. John Wiley & Sons, 2013.

Offline Domotic System using voice comands

Javier Errobidart, Alejandro José Uriz, Esteban González, Iván Exequiel Gelosi, Juan Alberto Etcheverry
Communications Laboratory
ICyTE (CONICET - National University of Mar del Plata)
Mar del Plata, Argentina
Email: ajuriz@fi.mdp.edu.ar, elgonzal@fi.mdp.edu.ar

Abstract—In this paper, an user interface and a communication platform for a modular home automation system is presented. Its main objective is to contribute to the comfort and autonomy of users who suffer from some type of disability. At the same time, it proposes to give a simple and economic solution to the problematic of the accessibility, since although there are many options of automation in the market, are few the options to centralize the control of the different home equipment. Also, the system does not require an Internet connection to operate, an advantage in countries where the mobile 4G Internet is not a standard.

The user interface has been developed using a speech recognition module which allows the identification of commands from different users. Modules communication is established via WiFi and was implemented using an ESP8266 commercial chip. This integrated hardware complains the 802.11 standard, contains the complete TCP stack and has a programable microcontroller.

Two protocols were exclusively developed, one for the association of the modules to the WiFi network and another for the communication of the different nodes involved in the home automation system. Both are based on the IoT (Internet of Things) approach and give the system decentralized communication. Although it does not require internet connection, optionally the system can be upgraded to establish a remote access.

Index Terms—Domotic enviroment, voice commands, IEEE 802.11, WiFi Network.

I. INTRODUCTION

Domotics [1] is an area of technological development that aims to contribute to energy management, comfort, security, communication and accessibility different equipment and variables in a home. This work is focused on the benefits of domotics for handicap people. Additionally, it will be taken into account others advantages of domotics.

The main objective of this development is to build an user interface that allows control through a hands-free system. Consequently, this could increase the life quality of people with motor disabilities by contributing to their comfort and autonomy. There are several options to carry out a domotics hands-free system. The best choice for each individual will depend on his/her capabilities. Last years works presented Bbrain-Computer-Interface (BCI) systems based on evoked potentials [2], and the use of cameras for tracking eyes movements [3]. A robust option is to use a voice control interface to operate the domotics system. This is an economic and non-invasive choice, which can be used by people with strong motor impairments. There are several works that use voice control systems, but the performance of this type of

systems depend on the training data [4]. Thus, an interesting option to face out this problem is the EasyVR [8] interface. This embedded system allow to recognize voice commands from up to 32 speakers (with a previous training). But, one of the best features of the EasyVR is a second operation mode, which allows to recognize a set of established basic sentences pronounced by any speaker (without a previous training). The sentences recognized in this mode are numbers and directions (up, down, right, left, back, next). Although, the number of sentences is limited, this operation mode can be used by several individuals without the training requirements. Another advantage of the EasyVR module is that there is plenty of libraries to be used with several families of microcontrollers, and it can understand commands in several languages. Finally, one of the best features of the adopted module is that it does not require an internet connection to process the audio, which is particularly useful in providing the user with reliable performance during the use of the system.

Once the order is recognized, the second stage of the domotics system must carry out the correspondent task and communicate it to the nodes installed in the environment. In this case, due to the fact that the central node of the system -controlled by the user- must be mobile, and then the remaining nodes can easily be reconfigured, it was choose wireless communication between nodes. Also, due to the distance requirements and the feasibility of using the wireless networks facilities commonly installed in home environments, the WiFi standard was adopted.

In order to establish a versatile operation, it was considered a modular home automation scheme and it was decided to use the Internet of Things (IoT) approach [5] for controlling different devices in a unified way. This means that each module has to be addressable and must have its variables available on the network for any user.

The ESP8266 [6] embedded module was adopted as the wireless interface. This has the necessary hardware features to accomplish the 802.11 specification [7], [9], [10] and it implements the complete TCP stack. The system needs the support of a WiFi network host to establish the connection. However there may be communication between the modules without having access to the internet. This capability is used to improve reliability during use. Consequently, the operation does not depend on services of third parties like ISP, DNS, servers, etc.

In order to install the modules and make the communication between them versatile and dynamic, it is necessary a protocol that supports these functions. With this aim, two protocols were developed. The first for the nodes association to the WiFi network, which is responsible for connecting a large number of modules to the network. The second controls the communication and the state of the nodes once connected.

This work is organized as follows: in Section II the first protocol is described. While, in Section III the protocol that allows the device interaction is presented. Section IV the hardware implementation, which is installed in rehabilitation center, is described. Finally, in Section V, the conclusions of the work are summarized and the future research lines are presented.

II. NETWORK ACCESS PROTOCOL

The first step for using the domotics system is to link the existing nodes to the WiFi network. To accomplish the specifications of this project (the device was developed for handicapped people), the association of each device should be simple. And, it is desired that the user uses the largest number of terminals (nodes). Thus, a simple mechanism must be provided to connect them. In Figure 1 an scheme of the proposed architecture is presented.

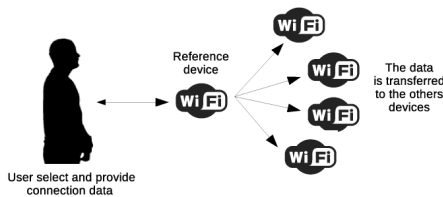


Figure 1. Scheme of the proposed architecture.

Each WiFi station (network node) needs to know the SSID (Service Set Identifier) and PSK (Pre Shared key) of the AP (Access Point) or reference device to establish the connection. This process requires that an administrator configures each node with the necessary data. If the number of nodes is large (which is common in home automation environments), this initiation procedure becomes unpractical.

The WiFi Alliance created the WPS (WiFi Protected Setup) [7], [9] protocol to simplify the association of the stations to a WiFi network which has three operation methods. The PBC (Push Button Configuration method) is the most desirable for this project. In this case, by pushing a button on the AP, it executes the "discovery" function. In this mode, the AP can detect stations that want to connect to him. During this time interval, the WPS mode must also be started at the stations to be connected, by pressing the corresponding button. Then, the node will connect to the Access Point through negotiation, and the "discovery" mode will turn off in each device. Although this process is the simplest to implement, WPS is not available on all APs and, depending on the manufacturer, it may fail.

Given the previously mentioned compatibility problems of the WPS mode, it was decided to create a new protocol to achieve a massive station association to the WiFi network. It is based on the features that provides the module ESP8266, which is used as wireless interface in each device (AP and stations).

The ESP8266EX is a low cost WiFi shield that integrates the TCP stack and the necessary hardware for the medium access specified by the 802.11 standard. It also has ADC, PWM outputs, SPI interfaces, serial communication, among others of the most common peripherals that can be found in actual microcontrollers. It is manufactured in different modules which allow the addition of flash memory, antenna (internal or external), etc. The manufacturer offer two SDKs that allow scheduling routines: RTOS (Real Time Operating System) and NRTOS (Not real Time Operating System). By mean of these, it is possible optimize the processing power of the 32-bit Xtensa LX106 CPU integrated in the ESP8266EX.

The microcontroller embedded into the ESP8266 allows to use the peripherals ports to obtain a simple association protocol. The main idea of the protocol is that the WiFi connectivity data would be transmitted only to one of the modules and then it share the data with the others that are wishing to associate with the reference device (or home AP). This is done using the ability of the ESP8266 to function as **softAP** (Software Access Point). Once the user are connected to the softAP of the ESP8266 module, a data channel is established through a TCP port.

Consequently, in the development of this novel protocol it was established that each module of the system can operate in two modes during the WiFi association: as a **reference device** or as a **query device**. In the first mode, the module operates in softAP mode waiting for the user to connect and send the data to the corresponding TCP port (through an application). It is also the mode that will then share the data with the requesting devices. In the second mode, modules, using a special routine, will connect to the softAP of the reference device to request the data and obtaining them in an unattended way.

This association procedure works while exists only one reference device, and the others modules are requesting devices. Also, it is required that the application of the user and the requesting devices know the SSID and PSK of the softAP. It is a configuration that is established by an administrator prior to the intervention of the user, and is stored in each module. The reference device will also play a fundamental role in the second protocol developed, which administrates the domotics functions, and it is presented in the Section III.

Data transfer is done over TCP/IP. Each time that a device requires to send data, each field of the same will be identified by a lexeme that the processor can analyze syntactically and, separate the information unequivocally. The following is exemplified in the case of sending SSID and PSK.

The string of characters sent would be:

```
{|NID}NetName{|PSK>Password{||}
```


The microprocessor extracts the SSID `NetName` and the PSK `Password`. The lexemes `{|NID }` and `{| PSK}` respectively are used to extract them. These precede the characters that indicate the data. The characters `{|` indicate the end of a data field, therefore, the beginning of any lexeme indicates the end of the data field that precedes it. The lexeme `{| }` marks the end of the character string.

This method allows to send several types of data in a same TCP/IP message in a very simple way and with low resource consumption in the processor. However, to ensure the operation, it is required that the combination of characters used to form the lexemes be reserved only for data signaling. Otherwise, a misinterpretation of content may be generated. This method of signaling data on the TCP protocol is used not only in the WiFi association protocol, but also in the second implemented protocol.

The process of association to the WiFi network begins when the reference device does not detect a network to connect with the pre-recorded data. Then, it activates the softAP mode by making available an SSID and PSK known by the user application. This allows the establishment of a TCP connection once the user connects to the WiFi infrastructure and to the IP protocol provided by the ESP8266 softAP. Then, the application will list the available networks and allow the selection or entry of an SSID and then request the PSK.

Once the user has entered the necessary information for the WiFi association, the reference device will connect to the WiFi infrastructure of the home AP. When the connection is successful, the reference device uses the home AP to perform the communication between the nodes that compose the domotic environment. It also keeps the softAP mode active so that the SSID and PSK of the AP can also be transferred to the consulting devices that need it. In this instance, the initiative is taken by the requesting devices through its own unattended routine, while the reference device responds only when scans that a new equipment is connected to its softAP.

This scheme allows the propagation of the SSID and PSK to all the devices that integrate the system, regardless of the quantity. Thus, in this configuration the administrator must enter the SSID and PSK information through the application only once. Figure 2 shows the flow diagram of the proposed protocol.

III. DOMOTIC PROTOCOL

As was previously described, it was necessary to develop a second protocol that allows communication between the modules of the system in order to let them to carry out the correspondent functions. This functions include control and data collection of peripherals. The main tasks of the developed protocol are described in the next Subsections.

A. Discovering the peripherals connected to the system

It is necessary that each module is presented to its peers in the system. This is important to allow the modules to correctly interact with the active devices, depending on the

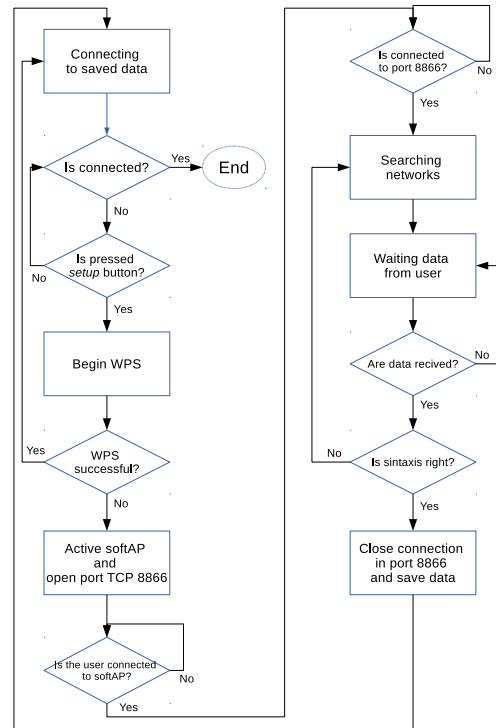


Figure 2. Flow Diagram WiFi Association developed protocol.

characteristics of each module associated to the system. To establish communication, they must know the IP addresses of each module. As in the first instance this data is lacking, the domotic protocol that was developed establishes that each module must be announced to the reference device. This implies that each node sends important information, such as the IP address, so that it is stored by the reference device and accessed by other devices that need them.

In order to the query modules to find the reference device, its IP address will always be the same. This is accomplished by changing the last field of the IP address assigned previously by the home AP to a known fixed value. The change is done by means of a routine when the reference device is connected to the AP. Therefore, each module knows that the address where it should be announced matches its own IP except for the last octet which has a default value. For this method to be valid, this address should not be used in any other device at the time the reference device is connected to the host AP. In the implementation, it was used the last field value=250. It can be chosen because normally is outside the default DHCP allocation pool that most of the home APs bring.

The reference device stores the announcements messages of all the peripherals that are entering the home AP. It also keeps the registration updated as the query devices will repeat the announcement message periodically for this purpose. This stored information can be retransmitted to any module that requests it.

The announcement message will contain the lexeme {—

IPA} to mark the content of the IP address of the device that is sending it.

B. Device Identification

In order to manage the communication properly each device must have an unambiguous identification that confers an identity on the system. Thus, the MAC address of each module was adopted. Among the data sent in the announcement message is this MAC address. From this data the devices are indexed in the registers of the reference device. For doing this, the lexeme {| MAC} is used in the announcement message which informs the MAC address of the device that sent that message.

C. Device properties identification

The modules can be actuators or sensors whit two states (on/off) or multiple states. Also, may be controller modules which controls the actuators and acquire data from the sensors. Each module of the system will interact in a particular way with each type of device starting from a routine programmed for each case. These routines can only be triggered from the information of the type of device with which the interaction is to be performed. This information is sent in the "type" field in the announcement message, which is a number from 0 to 255. This is indicated by the lexeme {| TYP}.

D. Establish a syntax for communication

The syntax allows the interpretation of data sent via the TCP protocol. It will be used the framework of lexemes that was introduced in the association protocol to WiFi with the addition of new ones.

Based on the tasks that the protocol establishes, the advertisement message was generated. In addition to {| IPA}, {| MAC} and {| TYP}, this message also includes {| NAM} for the display name.

The announcement message is identified with the lexeme {| ANU}, consequently when the device receives this message, it applies a subroutine that allows for data acquisition and registers updating. Other types of messages can be implemented which will have to have a particular lexeme to be identified so that the processor can attend to it with the appropriate subroutine. This method offers the possibility of using new lexemes to create new types of messages according the specific needs of the designer.

E. Take control of connected devices

During operation the modules could be turned off or their IP address changed. For this reason, it is important to keep the registers updated with this information. The quering devices must execute a routine that set up the sending of messages to the reference device with the value of its internal registers at periodic intervals. In this way the reference device updates the data of its own registers. If the update message does not arrive in the established time interval, it is assumed that the corresponding device was disconnected.

To control the events, the reference device has an internal clock and stores the time value of each received message in a

Table I
 LEXEMES USED IN THE PROJECT.

Lexemes	Function
{ NID}	Network WiFi name
{ NSN}	Index of scanned WiFi Network
{ PSK}	WiFi access key
{ MAC}	MAC address device
{ IPA}	Assigned IP address
{ NAM}	Visualize name
{ TYP}	Device type
{ ANU}	Announcement message
{ AKN}	Acknowledgment of receipt
{ SOH}	Turn ON
{ SOL}	Turn OFF
{ }	End of message

special register. In each program cycle it compares the current time with the saved in the register and in this way obtains a value that is equivalent to the time elapsed since the last announcement message.

$$\text{Current time} - \text{Saved time} = \text{Time elapsed}$$

In order to be able to carry an independent control for each identified module, the reference device creates a register for the announcement messages for each one. Figure 3 presents a flow diagram of the proposed verification system.

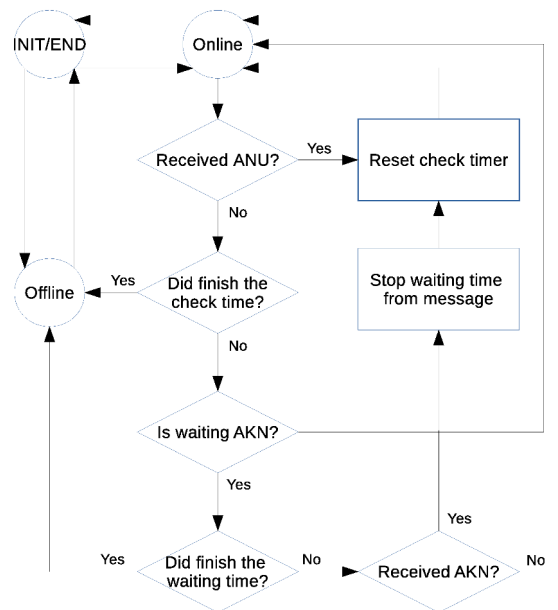


Figure 3. Flow diagram of the proposed verification system.

The number of devices of the network in principle is limited to the maximum number of connections that the host AP can establish. However, the memory capacity of the reference device is also another limiting, since it establishes the capacity of the register and, consequently, the total number of devices that can be identified.

Figure 4 presents the flow diagram of the domotic proposed protocol.

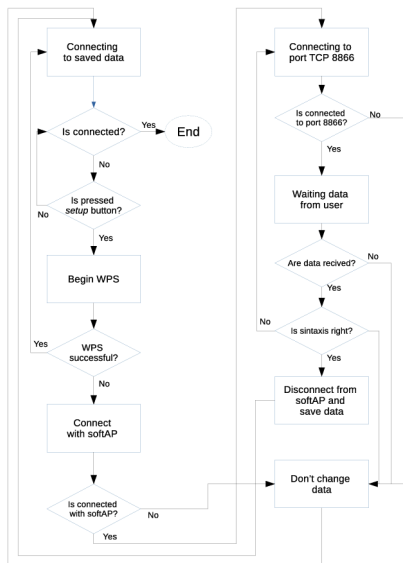


Figure 4. Flow Diagram domotic protocol.

IV. SYSTEM IMPLEMENTATION

A. Registers of Properties and Availability

The reference device must be able to establish two registers, one for the properties and other for the availability. The first register is the one that saves the data of MAC, IP, type of device and name, among others. The second register is in charge of saving the values of the clock to be able to take control of the availability of each device in the host network.

In addition to these values, the property register has a flag indicating whether the register has been used, it is done to avoid overwrite the data device unless it was strictly necessary. Also, the availability register has a flag that indicates if the device is currently connected to the network, facilitating the verification of this state.

By limiting the display name field to 8 characters, you get a total size for the record per device that is 19 bytes (6 for MAC + 4 for IP + 8 for display name + 1 for type) in addition to one bit of the Registration flag. While for the availability register, the internal clock length is 4 bytes, and to represent the availability status of each device is reserved 1 flag byte.

In the implementation of the reference device, it was reserved memory for registers according to the number of query modules that are planned to be used. For the case of the prototype developed in this project, it was used the Arduino Mega development board. This platform was chosen because it is possible simultaneously implement the user interface, and the reference device. In the implementation, was reserved memory for 8 devices, not for a hardware memory limitation, but for a design criteria, since for the purpose of the prototype that final number of modules was sufficient (the final user of

the prototype is a rehabilitation center whit only one domotic room). However, this can be easily modified from the source code. Figure 5 presents the implemented register assignation.

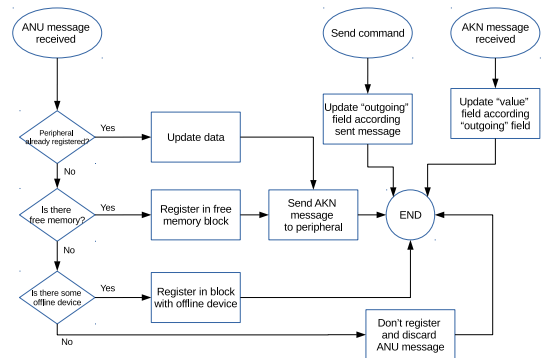


Figure 5. Flow diagram of the implemented register assignation.

B. Hardware Implementation

The prototype consists of an user interface and two actuator modules of two states (on-off). The user interface has an LCD display to show the menu and information of the devices. It also has the EasyVR voice recognition module, which allows you to operate the menu by means of voice commands (without Internet). As was previously described, for the WiFi interface, an ESP8266 module was used. The core of the reference device is an Arduino Mega development board. It was decided that the user interface was also the reference device. This was done because it is more efficient way of storing the information of the registers directly in the device. Therefore, the device and availability registers are managed by the Arduino board [12]. While, the actuator modules (whose function is to control two LEDs according to user commands), were implemented in an ESP8266. The model chosen was the ESP-01, which has two general purpose pins that allow to accomplish the required tasks. A flow diagram of the user interface and actuator modules is presented in the Figure 6.

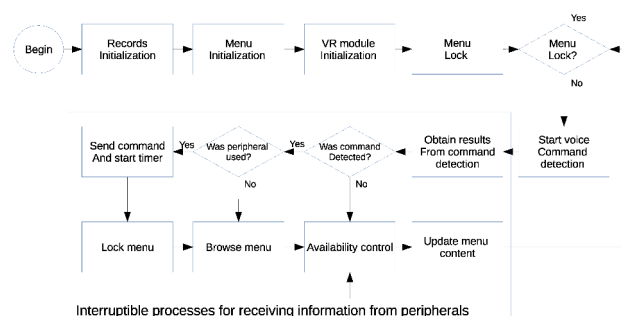


Figure 6. Flow diagram the implemented system.

The tasks listed below are executed in the Arduino Mega board there the user interface and reference module are implemented:

- Control of the user graphic interface (LCD Display).
- Voice Interface Control (EasyVR).
- WiFi Interface Control (ESP8266).
- Manage the user menu.

Actuator devices implemented in the ESP-01 module, carry-out the functions listed below:

- Reception and interpretation of commands.
- Send acknowledgment of receipt in response.
- Send announcement messages to the reference module periodically.
- Control of digital outputs/inputs.

The user interface program update in real-time the information shown on the display, which depends on the status of the user menu and the memory registers. Figure 7 shows a screenshot of the implemented navigation menu. It was divided in two levels: the first is used to select the device to control. Then, the second level shows the state of the correspondent device, and it is possible to change its state.

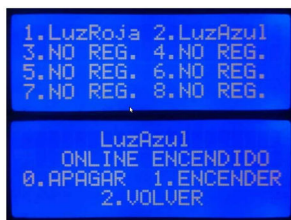


Figure 7. Screenshot of the implemented navigation menu. In the upper image, the initial menu is presented. Then, the second image shows the available options for the device corresponding to the item 1.

The EasyVR module detects voice commands. In the case of a correctly identification, the information is saved until it is consulted by serial port. Therefore, it is necessary to check in each loop the identification status of the EasyVR module. As the voice commands are used to navigate the user menu, each detected command triggers a modification in the status of the command, which is registered in the memory of the Arduino board.

The system was tested in different environments. In the tests was found a failure in the speech recognition module when it is used near to a WiFi station. In this case identification is blocked, resulting in erroneous detection that difficulties the navigation of the menu. Experimental measurements demonstrated that the prototype must be at least 50cm from another WiFi station to operates correctly. It was verified that this failure due to interference does not produce false positives in the terminals.

V. CONCLUSIONS

The prototype worked according to the design, more important yet, it was installed in a Public Rehabilitation Center.

The protocols developed allow the user to operate the system in a simple way, independently the model of WiFi nodes

used in the implementation. The low cost of the technologies used allows the system to be accessible to more people. In addition, the protocols establish a basis for future extensions and could adapt to particular needs of different users.

The prototype was designed to work without access to internet, which is an advantage. However, if the home AP has internet access, it is possible to install a server which allows remote monitoring and control.

The system has security vulnerabilities, since any user or device that knows the protocol and the IP addresses could spoof. To avoid this, it is necessary to implement an additional protocol of authentication that allows to establish groups of devices.

The user interface was developed in order that could be used in different environments (nodes have generic names). Also, additional circuitry can be added to each node to solve especial requirements, and to interact with previously home-installed equipment. New peripherals could also be developed to solve the particular needs of the user, which is vital in the case of handicapped people.

In addition to technical features, a low cost system was achieved. This is important in order to generate a technological solution which can improve the quality of life of people with strong motor disabilities (more than 50000 in Argentina). It was done, choosing integrated solutions, which have a high trade of performance-cost.

In the future the application of energy saving methods, increase in the number of peripherals, multiple user interfaces and personalized voice commands will be developed. Most of these improvements will be possible thanks to the integrated features of the ESP8266 and EasyVR.

REFERENCES

- [1] Colegio de Ingenieros Especialistas de Córdoba, 2012, *Domótica Registrada. Guía de contenidos mínimos para la elaboración de un proyecto de domótica* <http://www.ciec.com.ar/images/archivos/Domotica-CIEC.pdf>
- [2] D. Piccinini, N. Lopez, E. Perez, S. Ponce and M.E. Valentinuzzi, *Assistive home care system for people with severe disabilities*, 4th IEEE Biosignals and Birobotics Conference (ISSNIP), Rio de Janeiro, Brazil, 2013.
- [3] N.M. Dalgaard, E.L. González, A.J. Uriz, J.C. Tulli, F. Denk and P.D. Agüero. *Control mediante señales electro-oculográficas*. Proceedings of VI Jornadas Argentinas de Robótica (JAR). Buenos Aires, Argentina, 2010.
- [4] P.D. Agüero, A.J. Uriz, J.C. Tulli and E.L. González, *Improved algorithms for speaker verification embedded in a low cost dsPIC*, Proceedings of RPIC 2013, pp. 415-420, 2013.
- [5] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, *Internet of Things (IoT): A vision, architectural elements, and future directions*, Future Generation Computer Systems, vol. 29, no. 7, pp. 1645-1660, 2013.
- [6] Espressif Website, *ESP8266 Datasheet*. <https://espressif.com/en/support/download/documents>. 2016.
- [7] Rabbit, *An Introduction to Wi-Fi* http://ftp1.digi.com/support/documentation/0190170_b.pdf
- [8] Veeear, 2016, *EasyVR 3 User Manual Release 1.0.14* <http://www.veear.eu/files/EasyVR%203%20User%20Manual%201.0.14.pdf>
- [9] Forouzan B.A. *TCP/IP Protocol Suite (4)* New York: Mc Graw Hill
- [10] M.C. Liberatori. *Redes de datos y sus aplicaciones*. Ed. EUDEM. Argentina, 2016
- [11] E. Serna-Prez, 2010, *Análisis Léxico* <http://www.paginasprodigy.com/edserna/cursos/compilador/notas/Notas2.pdf>
- [12] Arduino Website, 2016, *Arduino Mega Board* <https://www.arduino.cc/en/Main/arduinoBoardMega>

WSN Clock Synchronization by Network-coded Messages

Pablo Briff^{1**}, Ariel Lutenberg^{1*}, Leonardo Rey Vega^{2*}, Fabian Vargas^{3**},
Mohammad Patwary^{4**}, and Rolando Carrasco⁵

¹ LSE, Facultad de Ingeniería, Universidad de Buenos Aires, Buenos Aires, Argentina

² LPSC, Facultad de Ingeniería, Universidad de Buenos Aires, Buenos Aires, Argentina

³ SiSC Group, Faculty of Engineering, Catholic University, PUCRS, Porto Alegre, Brazil

⁴ School of Computing and Digital Technology, Birmingham City University, Birmingham, UK

⁵ School of Electrical and Electronic Engineering, Newcastle University, Newcastle upon Tyne, UK

* Member, IEEE

** Senior Member, IEEE

Abstract—In this paper a novel method to improve the Wireless Sensor Network synchronization process by introducing network-coded messages is proposed. The presented technique outweighs the effects of the channel's outage probability by employing the available data to refine the estimates of the pairwise clock offsets. By virtue of the proposed technique, the mean square error of the estimator is significantly reduced for a given energy budget. The claimed features are supported by extensive simulation results for sensors communicating over Rayleigh fading channels.

Index Terms—clock synchronization, network coding, outage probability, wireless sensor networks.

I. INTRODUCTION

Wireless Sensor Networks (WSN's) nodes are intrinsically designed to operate at low power and with limited processing capabilities. This allows the construction of inexpensive sensor nodes which can be deployed in large numbers. Energy management is paramount in the field of WSN. A key aspect of a WSN that enables it to achieve its endurance target is time synchronization, which allows the implementation of data fusion, transmission scheduling and energy optimization schemes through power scheduling profiles. Typically, WSN synchronization involves some form of clock offset estimation for each WSN node. Still, time synchronization leads to an inevitable energy expenditure which is exacerbated by the message loss caused by the wireless channel's outage probability [1]. It is at this stage that using network-coded (NC) timing messages to compensate for the effect of the outage probability constitutes an appealing option for enhancing the clock estimation process. A number of recent works (e.g. [2], [3]) have explored the potential of network coding theory applied to WSN in relation to energy efficiency and performance enhancement, although these have not investigated applicability of such a tool to WSN clock synchronization.

In this work we propose a novel WSN synchronization strategy that exploits network coding theory to reduce the nodes' energy expenditure, which to the best of the authors' knowledge has not been disclosed before.

P. Briff is the corresponding author (pbriff@fi.uba.ar).

II. MODEL STATEMENT

Consider a WSN consisting of N nodes denoted u_1, u_2, \dots, u_N exchanging data via wireless messages. It is customary to assume that all nodes are constructively identical, as well as the received messages are affected by equal physical-layer Additive White Gaussian Noise (AWGN) power levels, denoted as σ^2 [4, p. 37]. The wireless channel studied in this paper is considered to experience flat and fast fading, and therefore the probability of success of each message is considered statistically independent [5]. WSN nodes synchronization is attained through message exchange. Message loss occurs mainly due to the channel's outage probability, defined as the probability that the received signal power falls under a minimum acceptable threshold [1]. Let $P_{out_{ij}}$ denote the outage probability of the channel when node u_i sends message x_{ij} to node u_j . Then, node u_j will correctly receive x_{ij} with a probability equal to $1 - P_{out_{ij}}$. Traditional approaches to deal with message loss include a combination of message retransmission and an increase in the transmit power. It is well-known that energy efficiency can be attained by wisely regulating the transmit power of the sender nodes [1]. These methods, however, do not exploit the available data to devise a mechanism that outweighs the effects of the outage probability while preserving the energy and synchronization quality. The remainder of this paper focuses on providing a working solution to the aforementioned challenge.

III. CLOCK OFFSET ESTIMATION USING NETWORK CODING

The existing trade-off between number of transmitted messages and transmit power has been extensively studied in the literature [1]. Increasing the number of transmissions while reducing the transmit power is an appealing strategy since the synchronization quality increases with increasing number of received messages. However, operating at low transmit power poses the challenge that the outage probability increases the message error rate. In order to outweigh the effects of the outage probability when operating at low transmit power levels, we propose an innovative clock offset estimation algorithm that introduces the concept of *network coding* [6, p. 411] in

the synchronization process. A particular case of a network coding strategy is the *linear network code*, in which data redundancy is added by a linear combination of the available data [7]. The advantage of linear network codes relies on their implementation simplicity. The application of NC WSN synchronization is two-fold: firstly, providing information redundancy to compensate for the inevitable message loss in the clock offset estimation process; and secondly, equalising the estimation error among the nodes in the network, as explained below.

A. Network-coded Clock Offset Estimation

Let node u_1 broadcast its clock offset measurement x_1 to its neighbours at time instant t_k , denoted as $x_1(k)$, which corresponds to synchronization round k . In general, $x_i(k) = x_i + n(k)$, where x_i is the real clock offset parameter of node u_i and $n(k)$ is the application-level additive noise with variance σ_n^2 and typically with Gaussian or exponential distribution. For example, assume that u_4 fails to receive message $x_1(k)$ whereas the remaining nodes store their reading of $x_1(k)$ in their internal memory, as shown in Fig. 1.

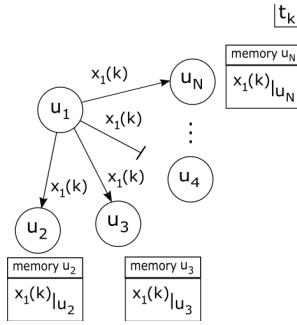


Fig. 1. Node u_1 broadcasts its clock offset measurement x_1 at time t_k .

At a following instant of time t_{k+1} , node u_2 broadcasts its own internal clock measurement $x_2(k+1)$ as well as the reading of u_1 's clock offset as depicted in Fig. 2, denoted as $x_{12}(k+1)$, thus providing information redundancy with respect to $x_1(k)$. Due to the fact that this approach doubles the number of transmitted messages m per node, the transmit power S shall be halved to attain the same energy expenditure $E \propto mS$ [1].

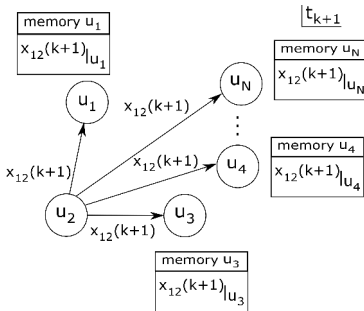


Fig. 2. Node u_2 broadcasts its measurement of node u_1 's clock offset.

Noting that $x_{12}(k+1) \triangleq x_1(k+1) - x_2(k+1)$, u_4 is able to reconstruct u_1 's clock offset from u_2 's retransmission, denoted as $\tilde{x}_1^{(2)}(k+1)|_{u_4}$, as follows:

$$\tilde{x}_1^{(2)}(k+1)|_{u_4} = x_{12}(k+1)|_{u_4} + x_2(k+1)|_{u_4} \quad (1)$$

where $x_{12}(k+1)|_{u_4}$ is the relative clock offset between u_1 and u_2 observed from u_4 , and $x_2(k+1)|_{u_4}$ is u_2 's clock measurement observed from u_4 at time t_{k+1} . The superscript in $\tilde{x}_1^{(2)}(k+1)|_{u_4}$ indicates that the estimate has been produced from a retransmission of u_2 . Eq. (1) allows u_4 to reconstruct the information contained in the lost message $x_1(k)$ by combining other information available or coded in the network. In general, the reconstructed message $\tilde{x}_i^{(l)}(k)|_{u_j}$ is given by

$$\tilde{x}_i^{(l)}(k)|_{u_j} = x_{il}(k)|_{u_j} + x_i(k)|_{u_j}. \quad (2)$$

Thus, a NC estimate of u_i 's clock offset produced at node u_j , denoted as $\hat{x}_i(k)|_{u_j}$, can be obtained as

$$\hat{x}_i(k)|_{u_j} = \hat{x}_i(k-1)|_{u_j} + a_i (\hat{x}_i(k-1)|_{u_j} - x_i(k)|_{u_j}) + \sum_{\substack{l=1 \\ l \neq i, j}}^N b_l (\hat{x}_i(k-1)|_{u_j} - \tilde{x}_i^{(l)}(k)|_{u_j}) \quad (3)$$

where the weighting coefficients a_i, b_l can be modified to prioritize the direct measurements $x_i(k)|_{u_j}$ over the NC reconstructions $\tilde{x}_i^{(l)}(k)|_{u_j}$, or vice versa. Both $x_i(k)|_{u_j}$ and $\tilde{x}_i^{(l)}(k)|_{u_j}$ are subject to inevitable message loss. Moreover, the computation of $\tilde{x}_i^{(l)}(k)|_{u_j}$ relies on two sets of measurements, i.e. $x_{il}(k)|_{u_j}$ and $x_l(k)|_{u_j}$, being correctly received by u_j . Therefore, let us define the error factors $\mu_i(k)|_{u_j}, \mu_l(k)|_{u_j}, \mu_{il}(k)|_{u_j}$ as follows: $\mu_i(k)|_{u_j} = 1$ if u_j receives x_i from u_i at t_k , or 0 otherwise; $\mu_l(k)|_{u_j} = 1$ if u_j receives x_l from u_l at t_k , or 0 otherwise; and, $\mu_{il}(k)|_{u_j} = 1$ if u_j receives x_i from u_l at t_k , or 0 otherwise. Then (3) can be rewritten as

$$\hat{x}_i(k)|_{u_j} = \hat{x}_i(k-1)|_{u_j} + \alpha_i(k)|_{u_j} (\hat{x}_i(k-1)|_{u_j} - x_i(k)|_{u_j}) + \sum_{\substack{l=1 \\ l \neq i, j}}^N \beta_{il}(k)|_{u_j} (\hat{x}_i(k-1)|_{u_j} - \tilde{x}_i^{(l)}(k)|_{u_j}) \quad (4)$$

with

$$\alpha_i(k)|_{u_j} \triangleq a_i \mu_i(k)|_{u_j} \\ \beta_{il}(k)|_{u_j} \triangleq b_l \mu_{il}(k)|_{u_j} \mu_l(k)|_{u_j}. \quad (5)$$

Eq. (4) outlines the generalized NC estimate of u_i 's clock offset produced by u_j at time t_k , considering the probability of error due to the channel's outage probability. By taking the expectation on both sides of (4), it is straightforward to note that $E[\hat{x}_i(k)|_{u_j}] = x_i|_{u_j}$, i.e. (4) is an unbiased estimator of x_i . Moreover, the optimal set of weighting coefficients a_i, b_l that minimizes the estimator's mean square error (MSE) can be obtained by using convex optimization methods, which will be investigated in future work.

B. Estimation Error Equalization

An extended application of NC messages in WSN synchronization is the equalization of the estimated clock offsets. For instance, consider the pairwise clock offset estimate $\hat{x}_{i,n}(k)|_{u_j} \triangleq \hat{x}_i(k)|_{u_j} - \hat{x}_i(k)|_{u_n}$. Naturally, $\hat{x}_i(k)|_{u_j} \equiv \hat{x}_{i,i}(k)|_{u_j}$. The closed-loop sum of consecutive pairwise clock offset estimates is given by

$$\sum_{i=1}^N \hat{x}_{i,n}(k)|_{u_j} = \xi(k)|_{u_j} \quad (6)$$

with $n = (i + 1) \bmod N$. In a noise-free communication channel, $\xi(k)|_{u_j} = 0$. However, in real-world applications the measurement noise, disturbances and inaccuracies in the construction of the nodes lead to $\xi(k)|_{u_j} \neq 0$. Hence such error can be shared, or equalized, among all nodes in the network by updating the clock offset estimates as

$$\hat{x}_i^*(k)|_{u_j} = \hat{x}_i(k)|_{u_j} + \frac{1}{N} \xi(k)|_{u_j} \quad (7)$$

where $\hat{x}_i^*(k)|_{u_j}$ denotes the equalized estimate of node u_i 's clock offset produced by node u_j at time instant t_k . Note that since the estimator described in (7) is also an unbiased estimator of x_i , MSE coincides with the estimator variance.

IV. HARDWARE CONSIDERATIONS IN NETWORK-CODED SYNCHRONIZATION

A typical WSN node hardware (HW) architecture consists of a low-power microcontroller unit (μ C), a power unit, a transceiver, a memory unit, an analog-to-digital (ADC) converter, and a sensor device [8, p. 10], as depicted in Fig. 3.

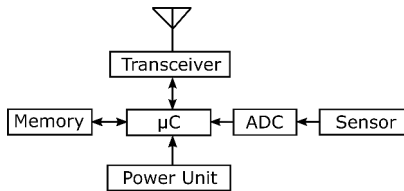


Fig. 3. WSN node architecture.

While the network-coded synchronization herein proposed does not introduce additional HW requirements to the sensor network, the following key considerations shall be made at design time:

- *Power consumption*: typically being the most important parameter in WSNs due to the limited energy resources in the nodes, the number of wireless transmissions must be carefully chosen in order to minimize the nodes' energy consumption, as detailed in [1].
- *Network size*: the algorithmic complexity of the NC synchronization is a function of the network size N . For N large, network clustering may be required in order to reduce the number of computations and transmissions per node.
- *Memory size*: careful RAM and flash memory allocation must be made due to the limited memory size in WSN

nodes. The proposed algorithm has a complexity of $\mathcal{O}(N)$ on the real-valued (float data-type) clock offset estimates $\hat{x}_i(k)|_{u_j}$ and its equalized variant $\hat{x}_i^*(k)|_{u_j}$, as given by (4) and (7) respectively. This constitutes a low complexity which only grows linearly with the network size N .

- *Processing power and sampling time*: the node's microcontroller shall execute total number of code instructions within a sampling time, typically in the range of hundreds of microseconds to hundreds of milliseconds, depending on the application.
- *Data transmission rate*: the rate at which wireless messages are sent via the transceiver shall be selected based on the power consumption requirements and the desired total synchronization time. For the NC synchronization, the data transmission rates may not be necessarily in line with the microcontroller's sampling time if a message queuing mechanism is put in place in order to store the received messages.

Therefore, based on the aforementioned considerations, it can be concluded that the synchronization technique proposed in this paper is well-suited for resource-constrained WSN nodes since it does not significantly increase the processing power and memory requirements of such devices.

V. SIMULATION RESULTS

Consider a WSN containing $N = 5$ nodes labeled u_1, u_2, \dots, u_5 deployed at random spatial positions, with a wireless channel experiencing Rayleigh fading effects. For example, let the real clock offsets be: $x_1 = 1.1s$, $x_2 = 1.2s$, $x_3 = 1.3s$, $x_4 = 1.4s$, and $x_5 = 1.5s$. The performance of the proposed approach is bench-marked against the well-known moving average (MA) estimator used by the Reference Broadcast Synchronization (RBS) [4, p. 20], given by

$$\hat{x}_i(k)|_{u_j} = \frac{1}{L} \sum_{r=k-L+1}^k x_i(r)|_{u_j} \quad (8)$$

where L is the number of samples in the MA estimation window. The MA estimation is performed at a transmit power $S_0 = 100mW$, whereas the NC estimator transmit messages are transmitted at $S_0/2 = 50mW$. The received signals are contaminated by physical-layer noise with power $\sigma^2 = 0.1W$ and application-level noise with variance $\sigma_n^2 = 10^{-3}$.

Fig. 4 shows the MSE of $\hat{x}_2^*(k)|_{u_1}$ for the MA estimator and several variants of the NC estimator, as a function of the number of synchronization rounds. The latter consistently outperforms the MA estimator despite operating at half-power, by virtue of the NC information that outweighs the effects of the outage probability. Although the number of synchronization rounds dominates the MSE, varying the weighting coefficients a_i, b_l also helps reduce the MSE of the estimator, as illustrated in Fig. 5 for the Equalized NC (ENC) estimator variant.

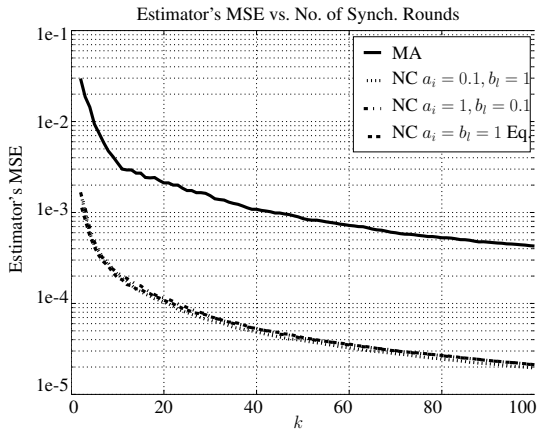


Fig. 4. MSE of the MA estimator (solid line) and MSE NC estimator with $a_i = 0.1, b_l = 1$ (dotted-line), $a_i = 1, b_l = 0.1$ (dash-dot line), $a_i = b_l = 1$ with error equalization (dash line) vs. No. of Synchron. Rounds k .

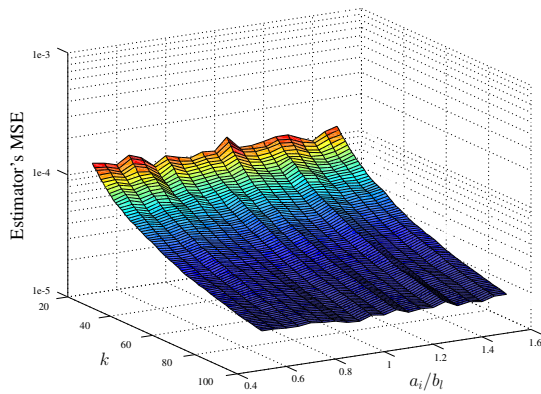


Fig. 5. MSE of the ENC estimator vs. No. of Synchron. Rounds k and weighting coefficients a_i, b_l .

VI. CONCLUSIONS

The presented technique exploits network coding theory in order to outweigh the effects of the channel's outage probability effects. It has been shown that the network-coded unbiased estimator of the clock offset outperforms the well-known moving average estimator for a given energy consumption budget. Furthermore, the proposed technique is well-suited for resource-constrained WSN nodes due to its low algorithmic complexity. This constitutes a powerful tool for energy-constrained WSN's that aim to attain the expected time synchronization performance at a low energy expenditure.

REFERENCES

- [1] P. Briff, A. Lutenberg, L. R. Vega, F. Vargas, and M. Patwary, "Generalised trade-off model for energy-efficient wsn synchronisation," *Electronics Letters*, vol. 51, no. 3, pp. 291–292, 2015.
- [2] B. Khodabakhshi and M. Khalily, "An energy efficient network coding model for wireless sensor networks," *Procedia Computer Science*, vol. 98, pp. 157–162, 2016.
- [3] Y. Yan, "Wireless sensor networks-cooperation and network coding for performance enhancement: Theories and experiments," Uppsala University, 2012.
- [4] E. Serpedin and Q. Chaudhari, *Synchronization in Wireless Sensor Networks: Parameter Estimation, Performance Benchmarks, and Protocols*. Cambridge University Press, 2009.
- [5] B. Sklar, "Rayleigh fading channels in mobile digital communication systems - part 1: Characterization," *IEEE Communications Magazine*, pp. 136–146, 1997.
- [6] R. W. Yeung, *Information theory and network coding*. Springer Science & Business Media, 2008.
- [7] I. Lopetegui, R. A. Carrasco, and S. Boussakta, "Voip design and implementation with network coding schemes for wireless networks," in *Communication Systems Networks and Digital Signal Processing (CSNDSP), 2010 7th International Symposium on*. IEEE, 2010, pp. 857–861.
- [8] M. Matin and M. Islam, *Overview of wireless sensor network*. INTECH Open Access Publisher, 2012.

Open-source embedded framework for Unmanned Ground Vehicle control using CIAA

Facundo Pessacg*, Matías Nitsche*, Adrián Teijeiro[†], Diego Martín[‡] and Pablo De Cristóforis[†]

*CONICET-Universidad de Buenos Aires, Instituto de Investigación en Ciencias de la Computación (ICC)

[†]Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales, Departamento de Computación

[‡]Facultad de Arquitectura, Diseño y Urbanismo, Universidad de Buenos Aires

Abstract—In this paper a new open-source and open-hardware framework for Unmanned Ground Vehicle (UGV) control is described. The hardware is based on CIAA (*Computadora Industrial Abierta Argentina*) and Arduino modules while the software is based on NuttX real-time operating system (RTOS). For communication, the MAVlink protocol is used, also enabling interfacing with ROS (Robot Operating System). The framework can be employed for different style UGVs (differential drive, omnidirectional, skid steer) with different number of motorized wheels. It is designed for low-cost and ease of integration and adoption. The proposed framework was employed as the control system of a four-wheel skid-steer rover prototype developed by the authors, which is also described as a part of this work.

I. INTRODUCTION

Mobile robotics has become one of the most interesting areas in the field of robotics. The development of new ground robot prototypes has made significant progress in recent decades, which has allowed their use in different environments, for a variety of purposes. For example, in factories [1], [2], [3], precision agriculture [4], [5] or space exploration [6].

In order to develop all these applications it is first necessary to have robotic research platforms where new algorithms and methods can be tested. Nowadays, there are many commercially available ground mobile robots designed to meet research needs in robotics labs. Research-oriented robot platforms are generally designed for ease of integration to existing systems and mounting different kinds of sensors and processing units.

Most commercial mobile robots have their own control system that are designed exclusively for the application which they are marketed for and its locomotion type (differential, skid steer, omnidirectional) [7] and number of actuators. These proprietary systems are closed technology and therefore make modifications or maintenance of the system to a large extent difficult. Besides these disadvantages, the other main drawback of commercial systems is their cost. For instance, the well-known research-oriented mobile robotic base called Husky [8] from ClearPath, costs around 30,000 u\$s in the basic A200 model. It is very difficult for universities or research labs from developing countries to afford these costs and hence this severely limits the possibilities of buying or upgrading these platforms to carry out field robotics research.

On the other hand, low-cost open hardware platforms can be found, such as TurtleBot [9]. However, these robots are

designed mainly for educational purposes and do not satisfy the research needs for field robotics.

When considering new low-cost and commercial off the shelf (COTS) based hardware prototypes, the Arduino [10] ecosystem is undeniably a widely used platform. Arduino is an open-source and open-hardware initiative started in 2003. Most Arduino board designs are based on an Atmel 8-bit AVR micro-controllers and, more recently, on 32-bit ARM micro-controllers. Moreover, due to its open nature, third-party new micro-controller development boards can be easily integrated to the Arduino framework.

One particularly popular robotic control framework corresponds to the ArduPilot project [11]. This framework targets ground, aerial, terrestrial and even underwater robots. While initially based on the Arduino framework and on AVR microcontrollers (using the ArduPilotMega board), the project now targets modern ARM-based micro-controller boards (also known as *autopilot* boards).

Another widely used open robotics framework is the Pixhawk [12] project. With similar goals to that of ArduPilot project, this project is more research-oriented, being started in 2009 by the Computer Vision and Geometry Lab at ETH, Zurich. Nowadays, the Pixhawk project supports many different autopilot boards.

While ease of use by non-experts has allowed Arduino to be used as the basis of robotic platforms, for efficient real-time operation of larger complex systems other embedded software frameworks can be considered. In fact, the ArduPilot project is now based on the same base software as Pixhawk uses, and thus can run on the same series of autopilot boards. Instead of Arduino, the Pixhawk project is based on the NuttX [13] RTOS (Real-Time Operating System), which offers a rich hardware abstraction layer and real-time programming resources such as threads and tasks.

In terms of control of ground mobile-robots, these frameworks present limitations. First, while an ArduRover variant exists, it mainly targets remote-control style cars with Ackerman steering. Moreover, there is no support for tight closed-loop control of each actuator (based on incremental encoders, for example). Second, these projects are large and development cycles are short enough to difficult developing stable versions of the framework upon which to base any new extensions.

One particularly interesting open-source and open-hardware system is the CIAA (*Computadora Industrial Abierta Ar-*

gentina) [14]. The CIAA is presented as the world's first and only open industrial computer. On one hand, it is designed for the exigencies of industrial applications. On the other hand it is open, since all information about its hardware design, firmware, software, etc. is freely available under the permissive BSD License. The CIAA computer presents itself as an interesting platform to implement the proposed framework and is thus used in this work for this purpose.

In this paper a new open-source control system for wheeled robots is described. The chosen hardware for the system are CIAA (Computadora Industrial Abierta Argentina) and Arduino modules while the software runs on NuttX real-time operating system (RTOS). For external communication the MAVLink protocol was used which allows the system to be controlled by ROS (Robot Operating System). To tackle the diversity of possible robot locomotion types, the presented framework is scalable to different motor and wheel configurations, both in terms of the software and the hardware.

II. SYSTEM OVERVIEW

The proposed system called OpenRover is comprised of a series of interconnected modules (see Figure 1). The Robot Controller Module (RCM) is the main component of the system and is in charge of controlling all of the robot subsystems, either directly or indirectly via the Motor Controllers Module (MCM). Each MCM is in charge of controlling an individual motor and is connected to the Robot Controller (via I²C bus). The system is thus designed to support an arbitrary number of motors and locomotion types. The Robot Controller can also optionally employ low-level sensors such as sonar range-finders for collision avoidance or an Inertial Measurement Unit (IMU).

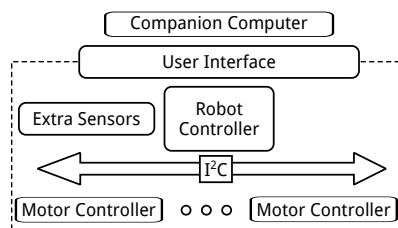


Fig. 1. Module description of the OpenRover software-hardware framework.

For the purpose of control of the robotic platform via an external companion computer (such as an embedded SBC or laptop computer), the Robot Controller offers a communication interface using the MAVlink protocol. Given that MAVlink-ROS communication bridges already exist, this also enables ROS-based control of the robot. Communication can be established using different types of serial protocol adapters, via a wired interface or wirelessly using Bluetooth or WiFi.

Finally, as part of the framework's User Interface, the proposed system includes a control panel where the robot is to be turned on/off and where auxiliary information can be displayed (e.g. battery charge level).

III. HARDWARE DESCRIPTION

While not really tied to a specific hardware platform, the aim is to employ readily available COTS hardware. In this sense, we propose to use CIAA computer, an open-hardware initiative intended to be used in industrial environments. The CIAA in this case is used as the Robot Controller module.

On the other hand, since one of the design considerations for the proposed framework is to support robots with varying number of motors, the MCM are to be implemented via independent micro-controllers. In order to reduce costs, small Arduino compatible boards were used. Specifically, Arduino Pro Mini boards were used for the current version of the system. The CIAA and the Arduino modules intercommunicate using the simple I²C bus, which is scalable to a high number of devices and only requires two wires.

In order to integrate the CIAA and Arduino boards in a single hardware system, three circuit boards were designed. First, a daughterboard (also known as *shield* in the Arduino terminology and as *poncho* for the case of the CIAA) for the CIAA was designed which includes a series of connectors to the other modules in the system. Second, a circuit board was also designed for the Arduino modules to be connected together via the I²C bus to the CIAA. A third circuit board was designed which consists of a control panel for the robot. These hardware units are described below.

A. Motor Controllers

The main goal of these modules is to perform closed-loop control of motor angular velocity. For this purpose, each Motor Controller is comprised of: an Arduino Pro Mini module, an incremental quadrature encoder, the motor driver and the motor itself (see Figure 2).

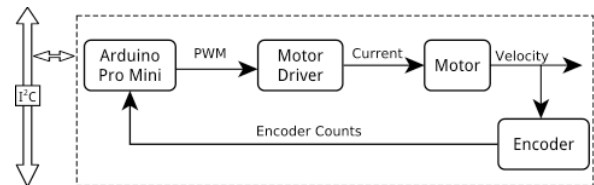


Fig. 2. Motor Controller module description

Using the incremental quadrature encoder attached to the motor axis, the Arduino module senses the angular velocity of the motor. Since the Arduino module does not feature a specific quadrature decoder logic, a simple interrupt-based routine is used.

Based on a given velocity setpoint (sent by the Motor Controller via the I²C bus) and using a PID (Proportional Integrative Derivative) controller, the Arduino module computes a given PWM duty to be sent to the motor driver. The driver then translates this PWM pulse to a higher voltage and high current signal in order to move the motor at a given speed.

The MTM is in charge of also maintaining the angular position of the wheel (knowing the gear reduction ratio and wheel diameter) for the purpose of robot odometry estimation,

typically used as input to more complex robot localization methods.

Finally, the MCM also senses the current applied by the motor driver in order to estimate battery consumption by this motor and also to detect stall conditions, which should trigger an emergency halt of the remaining motors.

A top-view of the designed MCM daughter-board (for the case of four-motors) is presented in Figure 3.

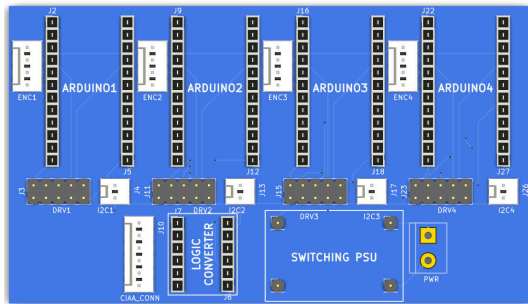


Fig. 3. **Motor Controller Module circuit-board**: in this case four-motors are supported, each with a connector leading to the corresponding driver (DRV_n), quadrature encoder (ENC_n) and I²C bus ($I2C_n$). It also regulates input power to the rest of the system and includes a logic-level converter to translate 5V (used by Arduino modules) to 3.3V levels (used by CIAA)

B. CIAA

The CIAA board is currently available mainly in two forms: an industrial-oriented version and an educational version. The main difference between these is that the educational version does not feature many of the external components geared toward handling high current or high voltage external devices. For this reason, the industrial version does not have as many available I/O pins as the educational version. Thus, for the proposed framework we chose to use the latter.

The CIAA includes an ARM Cortex-M4 LPC4337 micro-controller with ARM Cortex-M0 co-processor. The micro-controller has 1 MB of flash memory, 136 kB SRAM and 16 kB EEPROM. It operates at a clock frequency of more than 204 MHz.

C. CIAA shield

A new *shield* or *poncho* for the CIAA was developed, which connects directly to the male I/O pins of the CIAA and exposes a series of connectors needed to communicate with the different hardware modules that comprise the system (see Figure 4 for designed circuit-board).

In the first place, this shield has one connector for I²C communication with the MCM, sending desired velocity commands and receiving the rotational displacement information of each motor for robot odometry computation. This connector also carries the motor enable signal, which turns on or off all motor drivers, which is directly controlled by the CIAA. This feature is necessary in case the CIAA needs to stop the motors immediately, for example if internal faults are detected in some of the hardware modules. This same enable signal

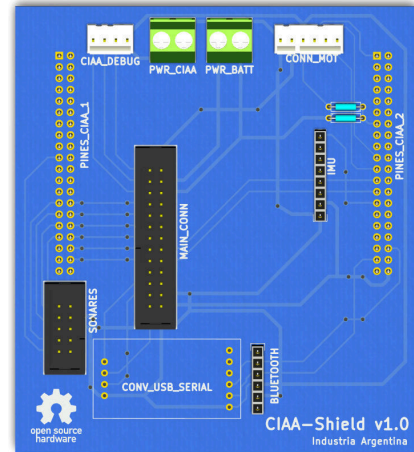


Fig. 4. **CIAA shield**: top-view of the board which connects to the CIAA male header pins on the underneath. On the top part, there are connectors to the MCM board (CONN_MOT), for power input from batteries (PWR_BATT), for communication (via USB-RS232 converter or Bluetooth adapter), for the optional sonar board and for the control panel (MAIN_CONN).

can be interrupted by the user by pressing the emergency stop button of the control panel (see Section III-D).

Another feature of the shield is the possibility of connecting an Inertial Measurement Unit (IMU) to the same I²C bus. This sensor gives a better robot motion estimation and could be integrated by means of sensor fusion or simply by reporting raw values to the external computer for further processing. The current version of the shield is designed based on the pinout of the MPU6050 IMU.

For the purpose of communication to an external computer (either embedded in the robot, externally mounted Laptop, an Android device, etc.) this shield exposes a serial connection which can be accessed in different ways. First, it is possible to connect via wire by means of an USB-RS232 adapter. Another possibility is to communicate wirelessly via Bluetooth using a HC-06 or similar adapter. Another option not included in current version is to also support WiFi communication using the low-cost ESP8266 module.

In addition to the aforementioned communication port, the CIAA debug connection is also exported for flashing and debugging the CIAA board and to provide access to the interactive NuttX serial console.

Another connector present in the shield is used monitoring the battery voltage by means of a resistor divider connecting to an ADC capable pin of the CIAA.

For the purpose of controlling and monitoring the robot via an external panel, a connector is also included which exposes: motor enable signal, input power line, an I²C connection for a graphic display and several connections for push buttons (for interaction via display).

Finally, a connector which allows for the CIAA to control a sonar range-finder array is also included. This feature is conceived for obstacle avoidance or basic mapping of vehicle surroundings. The connector is designed to support up to

sixteen sonars by means of a an external demultiplexer which should be present in the sonar-array control board.

D. Control Panel

The final hardware component of the proposed framework is an external panel which permits control of basic aspects of the robot (Figure 5). First, this panel exposes a power on-off switch and a power indication LED. Second, it features an emergency-stop push-button which interrupts the motor enable line and also signals the CIAA about the emergency condition.

The panel also features two USB connectors: one for wired communication with an external computer (when not using the Bluetooth wireless module on the CIAA shield) and another one connected to the CIAA debug port.

A piezo-buzzer is included for encoding certain events (boot OK, boot failure, etc.) via specific sequence of tones. A graphical display is also included to inform of the robot state.

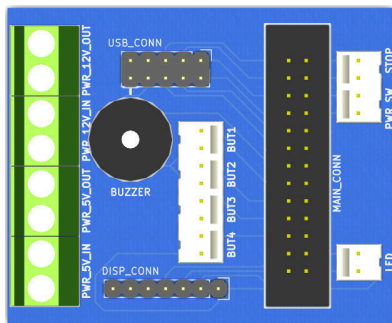


Fig. 5. **Control panel circuit-board:** the main connector goes to the CIAA “poncho”, the USB connector goes to external panel-mounted USB ports. Other connectors expose the logic-power line to turn on/off robot control, the motor-enable line for emergency stop button and the graphical display.

IV. SOFTWARE DESCRIPTION

In addition to the development of the hardware components of the framework, the embedded software (or *firmware*) of the CIAA and Arduino modules is here described.

In general terms, the driving goals for the embedded software development are to be: a) easily maintainable, b) based on common standards and c) not directly tied to a specific hardware platform.

For the case of the Arduino modules, the Arduino libraries and IDE were used for rapid prototyping and simple software development of the corresponding firmware. While not as efficient as ad-hoc code, they were chosen for simple understanding and maintenance of the system. Moreover, Arduino nowadays is a standard among hobbyist and educational environments.

For the case of the CIAA, since this board will be in control of many different tasks with real-time constraints, the use of an embedded Real-Time Operating System was decided. For this reason, the NuttX RTOS was chosen. While there are many freely available RTOS, NuttX has a series of features which stand-out among similar projects. Also, while the CIAA project includes a proposed firmware and RTOS, at present

date this support is not yet complete. Moreover, it involves CIAA-specific interfaces which were undesirable given the previously stated driving goals for this framework.

A. Arduino Firmware

The firmware of the Arduino modules is in charge of controlling each motor of the robot. For this purpose, the Arduino modules are connected via two interrupt-capable I/O pins to the incremental quadrature encoder attached to the motor’s axis and to the motor driver via two PWM capable pins.

For decoding the quadrature encoder signals, the Arduino `Encoder` library was used, which computes the incremental rotation of the encoder after each edge of either channel A or B signals using 16-bit resolution.

For controlling motor velocity, two PWM signals are used. The Atmega328p microcontroller has one 16-bit resolution timer for finer control. Moreover, default PWM frequency is around 1 kHz which is low enough for motor commutation to be audible. For these reasons, the `TimerOne` library is used which exposes the 16-bit timer changing the PWM frequency to 20 kHz.

Using the sensed incremental rotation of the motor shaft and given a certain angular velocity setpoint, closed-loop control of the motor is achieved via a simple PID controller. For this end, the `PID` Arduino library is used.

In order to sense the correct operation of motors, the current-sense outputs of the motor driver are connected to ADC capable pins of the Arduino module. The motor drivers used in current version (IBT-2, featuring a BTS7960 IC) also inform of any overcurrent or overtemperature condition via a fixed current on this pin.

Finally, for communication to the CIAA, an I²C protocol was defined. This protocol is used for setting the velocity setpoint, reading the angular position of the wheel, setting PID constants, reading instantaneous and cumulative motor current and enabling or disabling the PID.

The standard I²C protocol lets the Main Controller (i.e. the CIAA) of the system to control each motor independently of the specific hardware. Thus, this design considers eventual upgrading of the MCM hardware.

B. CIAA Firmware

The firmware for the CIAA is based on the NuttX RTOS. It has been widely used in other projects of development of robots control systems, both terrestrial and aerial. This has a strong impact in considering future CIAA project hardware developments and would even allow the use of other existing hardware platforms. NuttX mostly follows the architecture of the GNU/Linux OS, with a similar configuration utility and including the concept of architecture and device drivers, thus offering a standardized interface.

In order to integrate the CIAA to NuttX, support for the on-board LPC4337 micro-controller was first added. A second support layer was defined to integrate the particular CIAA board. This requires assigning specific functions for each pin

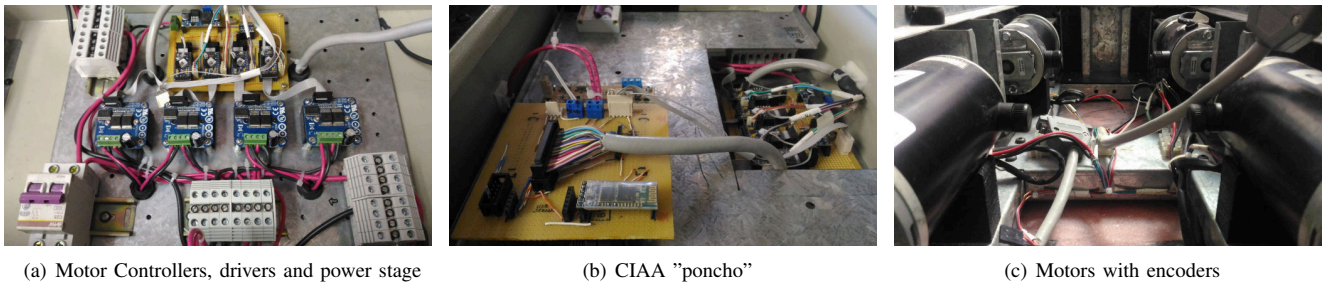


Fig. 6. Proposed system hardware prototypes in ElementAll robot

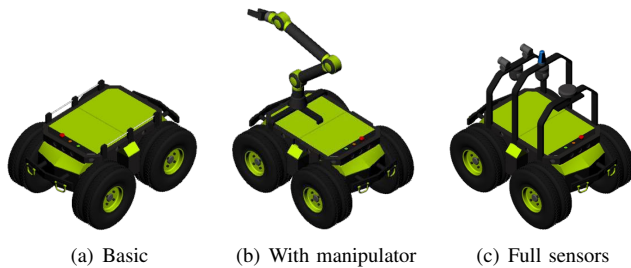


Fig. 7. Different configurations for ElementAll rover: (a) basic configuration with only on-board sonar array, (b) with 6DoF manipulator, (c) with stereo-cameras, LIDAR and RTK-GPS. In all cases with two set of wheels in parallel for greater traction.

according to its use and writing specific drivers for custom components, such as the I²C based MCM or external sonar control board. Finally, in the application layer of the RTOS, a user-level `rover_control` process is defined which holds most of the project-specific logic as a series of different tasks:

1) *Communication*: The first task is to monitor the serial communication line in order to receive external commands (e.g. desired robot linear and angular velocities) and inform about the robot state (e.g. odometry estimation). For communication, the MAVlink protocol is used, which is supported by a large number of GCS (Ground Control Software) used to monitor and control unmanned vehicles (typically aerial). Moreover, there is a `mavros` MAVlink to ROS bridge which translates messages from one system to the other. In this way, by supporting the MAVlink protocol, it is possible to interact with the proposed framework using standard ROS interfaces.

2) *Battery Monitoring*: A battery monitoring thread is started by the main task and measures the voltage via one of the on-board ADC ports. Two thresholds are defined which trigger different tones of the buzzer for early alert of low-battery and critical charge-level to the user.

3) *Motor Control Module*: One of the main tasks performed by the CIAA is to control the vehicle's motors via the MCMs. When the CIAA boots, communication via I²C is attempted in order to identify all MCMs in the bus. Only after this identification step is successful, the motor enable line is activated in order to enable the motor drivers. Since the user can halt the motors by pressing the emergency stop button (which opens the enable line circuit), the CIAA also monitors

the state of the stop button and resets the velocity setpoint of all motors to zero in order to avoid a jump when the emergency button is depressed. Finally, there is a watchdog which also disables the motors in case of communication timeout.

Another task is to compute the forward kinematic model of the robot (differential, omnidirectional, etc.) in order to translate desired linear and angular robot velocity to angular wheel velocity. Similarly, the inverse kinematic model is also computed in order to estimate the robot motion from wheel rotation obtained from incremental encoders. The equations to compute the forward and inverse kinematic models for different locomotion configurations can be found in [15].

V. SYSTEM IMPLEMENTATION FOR THE ELEMENTALL UGV PROTOTYPE

The framework presented in this work was implemented as part of the hardware/software system of a UGV prototype, the *ElementAll* robot (see Figures 7 and 6). This integration experience served the purpose of attesting the system design in terms of adaptation to a robotic research platform. In the following sections the ElementAll rover is first described and then integration considerations are discussed.

A. Rover Description

The ElementAll prototype is a skid-steer rover with four independently driven wheels. It was designed to operate in challenging terrain and weather conditions, such as high temperatures or under the rain or snow. It is conceived to be a research incremental platform, therefore the chassis is designed to allow different configurations and many sensors to be mounted on it. The basic model is only equipped with a sonar ring, however cameras, lasers range finders, IMUs and GPS are considered to may be added for specific perception purposes enhancing its autonomous navigation capabilities (see Figure 7).

The ElementAll is driven by four 100W 12V brushed DC motors from REMSSI, with a maximum speed of 2000 rpm, and 0.45 Nm torque. Motors have corresponding STM gearbox with a 1:50 reduction ratio. This gives an approximate global torque of 50 Nm (considering gearbox efficiency). The tires are 4 ply rated, 4.80/4.0-8 size with aluminum rims. Wheels have an outer diameter of 0.39 m and are 0.09 m wide.

The on-board energy source consists of two to eight 12V 9Ah lead-acid batteries connected in parallel. Motor drivers

are IBT-02, which are based on two BTS7960 half H-Bridge ICs. The maximum continuous current for this driver is 43A and features integrated over-current and over-temperature protection. As added protection, a breaker switch is included in the main power circuit.

B. System Integration

During the construction of the ElementAll prototype, the proposed framework was integrated as part of its control and communication system. Particular aspects of this integration stage are described here.

1) *Motor Controllers integration:* Since the ElementAll robot features a standard set of motor drivers featuring PWM control, current-sense and enable lines, the proposed Motor Controllers required no adaptation for interfacing. On the other hand, integration with motor incremental encoders required determining the appropriate encoder resolution. Since Arduino Pro Mini boards were used and since quadrature decoder logic is based on interrupt processing, encoder resolution was limited to 512 ticks per revolution. Using 4x counting, effective angular resolution is around 0.17 degrees/tick. For the ElementAll prototype the capacitive encoders AMT10X from CUI Inc. were chosen. A switching regulator with a wide range of input voltage from 3V to 40V is used for the power-supply (present in the Motor Controllers board), which allows for using different battery technologies seamlessly.

In Fig 6(a) a prototype for the Motor Controllers board installed inside the lower part of the ElementAll cabinet is presented. In Fig 6(c), the internal disposition of motors inside the metal frame can be seen, with encoders attach on the back of the motor axes.

2) *Robot Controller integration:* For the Robot Controller component, the CIAA “poncho” presented in previous sections was prototyped (Fig 6(b)) and integrated to the ElementAll cabinet and connected to the rest of the electronic components.

After installation of the corresponding hardware, ElementAll specific parameters were defined for use in the Robot Controller: motor number, disposition and orientation, wheel diameter, encoder ticks per revolution and gearbox reduction ratio. Moreover, PID constants also needed to be determined. In order to tune these control loops, individual motor velocities were monitored via MAVlink protocol for different values of P, I and D constants. After observing correct response, these were permanently set.

Another required step was to tune ADC conversion parameters for precise battery voltage level monitoring. After setting the low and critical thresholds for battery monitoring, correct operation of the alarms was checked.

After completing required integration and configuration steps, rover communication with external computers was tested. For this purpose, the serial-Bluetooth adapter was used and both a Desktop computer with ROS installed and an Android Smartphone with the MAVlink compatible QGround-Control application were tested. Via this communication interface it was possible to monitor internal robot parameters as well as control the robot via high-level velocity commands.

VI. CONCLUSION

In this work, a new open-source framework for Unmanned Ground Vehicles is presented, supporting different types of robot locomotion and different motor number and configurations. The system hardware is based on CIAA (*Computadora Industrial Abierta Argentina*) and Arduino microcontrollers while the software is based on NuttX real-time operating system (RTOS). For external communication the MAVLink protocol was used which allows the system to be controlled by ROS (Robot Operating System). The proposed system presented in this work was integrated into the ElementAll prototype UGV. This integration experience allowed to verify the correctness and suitability of the chosen system design.

VII. FUTURE WORK

As a future work we propose to finish the design, construction and integration of the sonar control-board with the proposed system. Moreover, we intend to produce final printed-circuit boards for all four hardware components. Also, we plan to release all relevant source-code, circuit schematics and circuit-board files in public repositories for adoption of the system in other platforms. Finally, we would also like to integrate the proposed framework in other similar robots to evaluate the scalability of the system.

ACKNOWLEDGMENT

The research was supported by PICT 2015-3167 of the Argentinian Ministry of Science and Technology and UBACyT 20020130300035BA of the University of Buenos Aires.

REFERENCES

- [1] S. Berman, E. Schechtman, and Y. Edan, “Evaluation of automatic guided vehicle systems,” *Robotics and Computer-Integrated Manufacturing*, vol. 25, no. 3, pp. 522–528, 2009.
- [2] H. Martínez-Barbera and D. Herrero-Perez, “Development of a flexible agv for flexible manufacturing systems,” *Industrial Robot: An International Journal*, vol. 37, no. 5, pp. 459–468, 2010.
- [3] H. Martínez-Barberá and D. Herrero-Pérez, “Autonomous navigation of an automated guided vehicle in industrial environments,” *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 4, pp. 296–311, 2010.
- [4] D. Bochtis, S. Vougioukas, H. Griepentrog *et al.*, “A mission planner for an autonomous tractor,” *Transactions of the ASABE*, vol. 52, no. 5, pp. 1429–1440, 2009.
- [5] D. Johnson, D. Naffin, J. Puhalla, J. Sanchez, and C. Wellington, “Development and implementation of a team of robotic tractors for autonomous peat moss harvesting,” *Journal of Field Robotics*, vol. 26, no. 6-7, pp. 549–571, 2009.
- [6] R. Kerr, “Hang on! curiosity is plunging onto mars,” *Science*, vol. 336, no. 6088, pp. 1498–1499, 2012.
- [7] J. Borenstein, H. Everett, L. Feng *et al.*, *Where am I? Sensors and methods for mobile robot positioning*, 1996, vol. 119, no. 120.
- [8] Husky. [Online]. Available: <https://www.clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/>
- [9] (2017) Turtlebot. [Online]. Available: <http://www.turtlebot.com/>
- [10] Arduino. [Online]. Available: <https://www.arduino.cc/>
- [11] APM. (2014) Ardupilot. [Online]. Available: <http://www.ardupilot.org>
- [12] L. Meier, D. Honegger, and M. Pollefeys, “PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms,” *2015 IEEE ICRA*, pp. 6235–6240, 2015.
- [13] G. Nutt. (2014) Nuttx. website. [Online]. Available: <http://www.nuttx.org>
- [14] Computadora Industrial Abierta Argentina (CIAA). [Online]. Available: <http://www.proyecto-ciaa.com.ar>
- [15] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. MIT press, 2011.

Software Patterns for Asymmetric Multiprocessing Devices on Embedded Systems: a performance assessment

Pedro Ignacio Martos

GPSIC & LSE – Facultad de Ingeniería
Universidad de Buenos Aires
Ciudad Autónoma de Buenos Aires, Argentina
pmartos@fi.uba.ar / pimartos@gmail.com

Alejandra Garrido

LIFIA – Facultad de Informática
Universidad Nacional de La Plata & CONICET
La Plata, Prov.de Buenos Aires, Argentina
garrido@lifa.info.unlp.edu.ar

Abstract—In embedded systems there is a variant of Multicore System on Chip devices (MSoC devices) where not all the computing elements (processor cores) are equal. The differences in the cores of these devices range from different hardware architectures using the same instruction set to completely different processors working together inside the same device. These SoCs are called “Asymmetric Multi Processing Devices” (AMP Devices). In order to help developers to take advantage of the possibilities that these devices may offer in the context of embedded systems, software design patterns have been defined, describing software architectural solutions with known uses. However, there are still no experimental results showing the benefits of these solutions. In this work we measure the performance of a design pattern called Mini Me, applied on an AMP device configuration, and compare it against two Symmetric Multiprocessing Device (SMP Device) configurations. The evaluations show a better than expected computing performance of the AMP Configuration using the design pattern Mini Me.

Keywords— *Embedded Systems Patterns; Asymmetric Multiprocessing Patterns.*

I. INTRODUCTION

Embedded Systems (E.S), as opposed to general purpose systems, are systems developed for a very specific purpose. In some cases the final user of the system can configure them (even program them), but the system is not intended to change its purpose. These systems are called “embedded” because they are part of a bigger system in a device with a specific functionality [1].

An Asymmetric Multicore Processor (AMP), is a processor where the computing elements (“cores”) have different characteristics. They can vary from the same processor running with different clock speeds, up to completely different processor architectures (i.e., 64 bit cores with 32 bit cores) in the same processor.

These processors are studied because they have been shown to provide improved computing performance [2], and also in the (performance)/(power) and (performance)/(silicon area) ratios [3], so they are attractive for E.S. implementations.

However, designing an E.S. that can take advantage of AMPs’ improved performance is hard. For example, performance asymmetry may adversely affect behavior of many workloads on commercial servers and make them less scalable [2].

With the goal of helping developers identify good architectural decisions when implementing software on AMPs and take advantage of their very specific characteristics, we have been working on the specification of a pattern language for AMP Embedded Systems [4]. A pattern language is a collection of interconnected design patterns or good practices in a specific domain [5]. Each design pattern in the language identifies a recurrent solution to a problem in a specific context [6]. Thus, a pattern conveys a small nugget of design and architectural knowledge to solve a problem within a certain context and after resolution it leaves the system in a new context, where there are new problems to be resolved by the other patterns in the language [7].

In this work, we apply one of that software patterns, called “Mini-Me”, on an AMP configuration, to evaluate its performance against a traditional Symmetric Multiprocessing (SMP) configuration. In a few words, the pattern Mini-Me proposes to address the low power vs. high performance requirements using an AMP with high performance cores and low power cores with the same ISA. Thus, the low power cores become a “mini” version of the high performance cores.

In the work by Balakrishnan et al., authors demonstrate that using an AMP configuration causes a performance gain with generic workloads of multithreaded commercial applications [2]. In their experiment, they approximate performance asymmetry by varying the individual processor frequencies in a multicore system, i.e., varying the speed of clocks in each processor. Similarly, the purpose of our study is to evaluate if an AMP still gives higher performance when using a “Mini-Me” configuration running an E.S. Thus, we intend to answer these 2 questions:

Q1. Is performance still “better than expected” in an AMP configuration where asymmetry comes from processors with different hardware? (as opposed to processors with the same hardware but different clock speeds)

Q2. Is improved performance sustained when the workload is specific to an E.S.? (as opposed to being a generic workload)

To answer these questions we use a computing platform for E.S. containing 4 complex cores and 4 simple cores, which have the advantage that processors can be turned on/off dynamically, thus facilitating the configuration of a symmetric or asymmetric platforms. Moreover, we use a well-known E.S. multiprocessor benchmark suite for our study: ParMiBench [8].

The rest of this paper is structured as follows: Section 2 presents related work. Section 3 describes the benchmark and details of the performance assessment. Section 4 shows the experimental results and Section 5 presents the conclusions and future work. Finally, we include an Appendix with the complete description of the pattern Mini-Me and its template for the sake of self-containment.

II. RELATED WORK

The AMP processors are studied as general computing elements because of the advantages that they offer, in particular, their computing performance is over the expected average. Studies show that the AMP configurations with two complex processors and two simple processors have better performance than the average of four complex processors and four simple processors over a wide variety of general computing loads (application server, database server, web server, scientific computing, video compression, massive software compilation) [2, 3]. In Fig.1 (extracted from [2]) we can see the performance comparison between SMP and AMP architectures for the different general computing loads. In that work, the AMP architecture was made by reduction of the system clock of some processors, so nf/ms scale means n fast cores and m slow cores running at $1/scale$ the speed of fast cores (all the cores are equal except for their clock speed). The total computing power of a system of these characteristics is $(n+m/scale)$. Symmetric configurations were 4f-0s; 0f-4s/4 and 0f-4s/8 and asymmetric configurations were 3f-1s/4; 3f-1s/8; 2f-2s/4; 2f-2s/8; 1f-3s/4 and 1f-3s/8

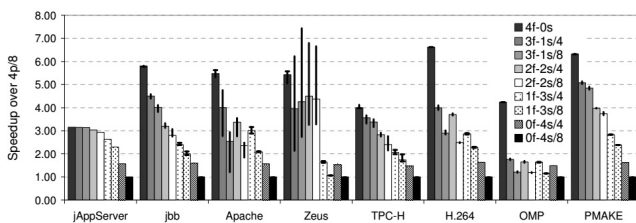


Fig. 1. Performance scalability for different SMP and AMP configurations.

Those results are valid for a general purpose computing device using standard workloads. However, in order to get results representative of the E.S. domain, we need to use a benchmark with E.S. oriented tasks. One such benchmark is ParMiBench [8], which is a multicore evolution of the MiBench [9] benchmark, a well-known suite used to evaluate uniprocessor E.S. performance.

As we explained in the introduction, a shortcoming of the work of Balakrishnan et al. for our purpose is that they approximate performance asymmetry by varying clock speed

in each individual processor of a multicore system. While that can give a good approximation, we aim at evaluating performance when the hardware of the processors is different.

Furthermore, we have been working on the specification of a pattern language for AMP Embedded Systems [4]. This pattern language contains architectural software patterns, i.e., patterns at the level of architectural design, which are intended to help developers take advantage of the specific characteristics of asymmetric platforms in the design of E.S. Other works on pattern languages related to this domain are Hammer's patterns for fault tolerant software [7] and White's patterns for embedded systems [10]. One of the patterns in the language of architectural patterns for AMP E.S. is "Mini-Me" [4], which is used in the context of a battery powered E.S., with long periods of low activity interrupted by short periods of very intensive activity. This pattern addresses the problem of preserving battery power when there are tasks that must be executed in both periods. Our intention in this work is to evaluate the performance of the architecture proposed by Mini-Me in the context of E.S.

III. PERFORMANCE ASSESSMENT

The hardware platform used in our experiment was an Odroid XU4 Single Board Computer (SBC) [11], using a Samsung Exynos 5422 octacore Processor, which has four complex cores (ARM Cortex A15@2GHz) and four simple cores (ARM Cortex A7@1.4GHz). Both type of cores share the same Instruction Set Architecture (ISA), so the software is not aware about the kind of core where it is running, thus making the platform well suited to test the Mini-Me Software Pattern implementation. The Operating System was a vanilla Ubuntu Mate 16.04 tailored for that specific hardware platform obtained from the hardware platform web site. No special configuration / optimizations were made during the tests except to turn on/off the processors to implement the SMP and AMP configurations [12].

The ParMiBench [8] [13] [14] test suite has four E.S. subdomains: Automotive, Networks, Office and Security:

The **Automotive** tests are "BasicMath", which performs simple mathematical calculations, e.g., cubic function solving, angle conversions from degrees to radians, and integer square root. The input data set used for benchmarking is a fixed set of constants; and "Susan", which is an image recognition application for recognizing corners and edges. The input data is a complex picture.

The **Network** category represents embedded processors in network devices like switches and routers. The work done by these embedded processors involves shortest path calculations, tree and table lookups, and data input/output. The algorithms used to demonstrate the networking category are finding a shortest path in a graph and creating and searching a "Patricia tree" data structure. The "Dijkstra" benchmark constructs a large graph in an adjacency matrix representation and then calculates the shortest path between every pair of nodes using repeated applications of Dijkstra's algorithm. Dijkstra's algorithm is a well known solution to the shortest path problem and completes in $O(n^2)$ time. The "Patricia" test implements a Patricia tree: a data structure used in place of full trees with

very sparse leaf nodes. Branches with only a single leaf are collapsed upwards in the tree to reduce traversal time at the expense of code complexity. Often, Patricia trees are used to represent routing tables in network applications. The input data for this benchmark is a list of IP traffic from a highly active web server for a 2 hour period. The IP numbers are disguised.

The **Office** category includes text manipulation algorithms to represent office machinery like printers and scanners with OCR recognition. The “StringSearch” benchmark finds a specific word in a number of given phrases by employing case sensitive or insensitive comparison algorithms.

The **Security** test is a SHA hash algorithm that produces a 160-bit message digest for a given input. It is often used in the secure exchange of cryptographic keys and for generating digital signatures. It is also used in the well-known MD4 and MD5 hashing functions. The input data set is a large ASCII text file of an article found online.

We executed the ParMiBench suite in three different configurations (two SMP and one AMP). The SMP configurations were 4c0s and 0c4s; and the AMP configuration was 2c2s. A XcYs configuration means X complex processors (ARM Cortex A15) and Y simple processors (ARM Cortex A7). It must be noted that in this particular SoC (Exynos 5422) the hardware interrupt controller is hardwired to CPU0 (a simple Cortex A7 processor). Thus, in practice it is impossible to disable CPU0 at all (because this also would disable all system hardware interrupts); so the configuration 4c0s is in fact a 4c1s with the benchmark running in the four complex processors. To assess the influence of that simple core, we also run the benchmark in the 4c4s full SMP configuration, and we verified that changing from 4c1s to 4c4s has only a very modest 5% in performance increase. Thus, we can conclude that the influence of 3 simple cores when the benchmark runs in the 4 complex cores has very little impact in the benchmark results. Therefore, the influence of 1 simple core will be less than that and it is valid to consider the configuration 4c1s equal to 4c0s for the benchmark results.

For each system configuration (SMP and AMP) we obtained the total execution time, and to take into account execution time variances from the operating system, each test was executed 10 times. For each test in each configuration we obtained the shortest, the largest and the average execution times; and also the total benchmark execution time (10 runs of each test).

IV. EXPERIMENTAL RESULTS

Tables 1 to 6 show the minimum, maximum, average and total execution time for each test of the benchmark over the three configurations. The tables have two extra columns: one with the expected performance and the final with the improvement over the expected performance of the AMP configuration. The expected performance function $Exp.Perf(2c2s)$ is the average between the SMP complex configuration (4c0s) performance, $Perf(4c0s)$ and the SMP simple configuration (0c4s) performance $Perf(0c4s)$, as defined in Equation 1:

$$Exp.Perf(2c2s) = [Perf(4c0s) + Perf(0c4s)] / 2 \quad (1)$$

Where $Perf$ is the performance (execution time) of the particular configuration. Meanwhile, the improvement over the expected performance $AMP_Impr()$ function is the ratio between the expected performance of the 2c2s AMP configuration and the real performance of that configuration, as defined in Equation 2:

$$AMP_Impr() = Exp.Perf(2c2s) / Perf(2c2s) \quad (2)$$

TABLE 1: BASICMATH

	4c0s	2c2s	0c4s	Exp.Perf	AMP Imp.
Min.	140.45	180.13	227.37	183.91	1.02
Max.	141.46	182.72	230.74	186.10	1.02
Avg.	140.79	181.25	229.66	185.23	1.02
Total	1407.90	1812.54	2296.64	1852.27	1.02

TABLE 2: SUSAN

	4c0s	2c2s	0c4s	Exp.Perf	AMP Imp.
Min.	0.0060	0.0060	0.0097	0.0079	1.31
Max.	0.0071	0.0090	0.0115	0.0093	1.04
Avg.	0.0062	0.0066	0.0104	0.0083	1.25
Total	0.0620	0.0664	0.1038	0.0829	1.25

TABLE 3: DIJKSTRA

	4c0s	2c2s	0c4s	Exp.Perf	AMP Imp.
Min.	0.0181	0.0179	0.0293	0.0237	1.33
Max.	0.0210	0.0219	0.0335	0.0273	1.25
Avg.	0.0193	0.0190	0.0310	0.0252	1.33
Total	0.1928	0.1928	0.3104	0.2516	1.30

TABLE 4: PATRICIA

	4c0s	2c2s	0c4s	Exp.Perf	AMP Imp.
Min.	0.0060	0.0059	0.0096	0.0078	1.33
Max.	0.0071	0.0080	0.0129	0.0100	1.25
Avg.	0.0063	0.0064	0.0101	0.0082	1.29
Total	0.0628	0.0635	0.1013	0.0821	1.29

TABLE 5: STRINGSEARCH

	4c0s	2c2s	0c4s	Exp.Perf	AMP Imp.
Min.	213.77	255.52	599.49	406.63	1.59
Max.	216.51	259.57	608.80	412.65	1.59
Avg.	215.34	257.42	602.88	409.11	1.59
Total	2153.40	2574.25	6028.76	4091.08	1.59

TABLE 6: SHA

	4c0s	2c2s	0c4s	Exp.Perf	AMP Imp.
Min.	0.0623	0.0721	0.1354	0.0989	1.37
Max.	0.0765	0.0818	0.1650	0.1207	1.48
Avg.	0.0669	0.0759	0.1482	0.1075	1.42
Total	0.6026	0.7592	1.4819	1.0422	1.37

In Table 7 we show the total execution time of the ParMiBench benchmark suite for the three configurations, the expected performance and the AMP configuration improvement over that expected value.

TABLE 7: PARMIBENCH TOTAL EXECUTION TIME

	4c0s	2c2s	0c4s	Exp.Perf	AMP Imp.
Tot.Ex.Time	3562.22	4387.87	8327.40	5944.81	1.35

In Figure 2 we show the average AMP configuration improvement for each type of test.

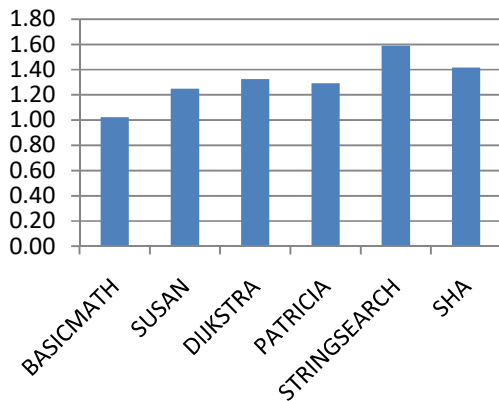


Fig. 2. AMP Configuration Improvement for each type of test

Analyzing the results in Tables 1 to 7 and in Figure 2 we can see that the BASICMATH test shows the expected performance improvement (the average between both SMP configurations), so for E.S. applications that make intense use of math, the use of the Mini-Me architectural pattern doesn't affect adversely the performance of the system. For the other tests, we found a performance improvement from 1.25 to 1.6 times over the expected performance; with an average of 1.4 times.

The reason for these improvements can be explained by the hardware differences between both processors [15]: Cortex A15 is a high end, triple-issue, out-of-order processor core which also implements virtualization instructions, hardware-accelerated integer division, and 40-bit virtual memory addressing extensions, so it has a very good single-threaded peak performance [16]. Figure 3 from [15] shows the Cortex A15 instruction pipeline.

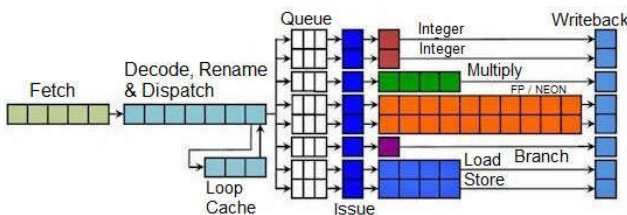


Fig. 3. ARM Cortex A15 Instruction Pipeline

On the other side, Cortex A7 processor is an in order, partial dual issue machine. The dual integer pipelines are eight stages long; the Cortex A7 combines full ALU (labeled "integer" in Figure 4 below) and partial ALU (labeled "dual-issue") structures, thereby enabling dual issue instruction execution for some integer operations. However, both conventional multiplication and NEON SIMD operations are single issue only. These architectural differences between both processors, and taking into account power and die area, generate a roughly equivalence of one Cortex A15 to four Cortex A7 [16]. Figure 4 from [15] shows the Cortex A7 instruction pipeline.

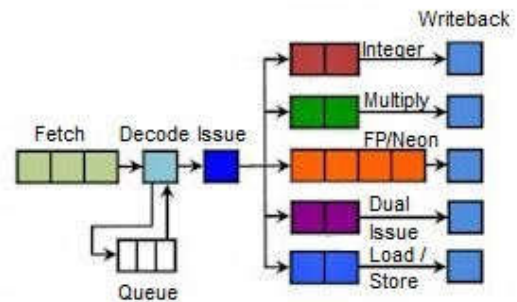


Fig. 4. ARM Cortex A7 Instruction Pipeline

Therefore, a SoC that implements two Cortex A15 and two Cortex A7 processors will have a better than expected performance because the Cortex A15 processors will have a high single thread peak performance and the Cortex A7 will produce a high multithreading peak performance.

V. CONCLUSIONS AND FUTURE WORK

We advocate for the definition and use of design patterns, which carry the knowledge of experts in a specific domain, thus helping developers to understand the problems that they may encounter in the domain, and the possible solutions with their weighted consequences. Moreover, in the particular context of E.S., it is very important to have real performance measurements of the solutions proposed by each pattern, and this is, to our knowledge, the first work on that direction.

In this work we measured the performance of the design pattern called Mini Me, using a benchmark for E.S. oriented tasks. Our evaluation was guided by two research questions. The first question: "Is performance still "better than expected" in an AMP configuration where asymmetry comes from processors with different hardware?" has been proved to be true by our experiment, as we still have a "better than expected" performance in a AMP system. The second question: "Is improved performance sustained when the workload is specific to an E.S.?" has been also found to be true, as results show that we can expect a performance improvement of about 1.4 times on average over the expected performance of the AMP system.

Future work includes running experiments on other design patterns in the pattern language for AMP-based embedded systems and extending, as well as discovering other patterns in the language.

REFERENCES

- [1] Steve Heath, *Embedded Systems Design*, 2nd Ed. Elsevier, (2002)
- [2] Balakrishnan, Rajwar, Upton, Lai, The impact of performance asymmetry in emerging multicore architectures, *Proceedings of the 32nd International Symposium on Computer Architecture (ISCA '05)*. IEEE. (2005)
- [3] Fedorova, Saez,Shelepov, Prieto, Maximizing power efficiency with asymmetric multicore systems, *Communications of the ACM*, Vol 52 Issue 12 (2009)
- [4] P. Martos, *Architectural Patterns for Asymmetric Multiprocessing Devices on Embedded Systems*, *Proceedings of the 11th Latin American Conference on Pattern Languages of Programs (SugarLoaf PLoP '16)*. Hillside Group (2016)
- [5] R. Hanmer, *Pattern-Oriented Software Architecture For Dummies*, John Wiley & Sons, 2013
- [6] Gamma, Help, Johnson, Vlissides. *Design Patterns*. Addison-Wesley, 1995.
- [7] Robert S. Hanmer, *Patterns for fault tolerant software*, Wiley Series in Software Design Patterns. John Wiley & Sons (2007)
- [8] S.M.Z.Iqbal, Y.Liang, H.Grahn., "ParMiBench – An Open-source Benchmark for Embedded Multiprocessor Systems", *IEEE Computer Architecture Letters* Vol 9 Issue 2. IEEE (2010)
- [9] M.R.Guthaus, J.S.Ringenberg, T.Austin, T.Mudge, R.B.Brown, *MiBench: a Free, commercially representative embedded benchmark suite*, Proc. of the IEEE International Workshop on Workload Characterization (WWC-4), IEEE (2001)
- [10] E. White, "Making Embedded Systems: Design Patterns for Great Software", O'Reilly Media (2011)
- [11] "ODROID-XU4", http://www.hardkernel.com/main/products/prdt_info.php, (retrieved May,2017)
- [12] <https://forum.odroid.com/viewtopic.php?f=93&t=16525> (retrieved May,2017)
- [13] <https://sites.google.com/site/parmibench/> (retrieved May,2017)
- [14] <https://github.com/cota/parmibench> (retrieved May,2017)
- [15] <https://www.bdti.com/InsideDSP/2011/11/17/ARM> (retrieved June,2017)
- [16] http://www.eetimes.com/author.asp?section_id=36&doc_id=1318968 (retrieved June, 2017)

APPENDIX A. PATTERN "MINI-ME"

This section presents the description of the pattern evaluated in this work, to make the paper self-contained.

First, it is important to note that the patterns in the language for AMP-based E.S. are described with a specific template, as follows:

- **Context:** summarizes situations in which you may find the pattern useful
- **Problem:** provides a brief summary of the problem which is addressed by the pattern
- **Forces:** describe the conflicts of interest occurring in the problem
- **Solution:** describes how the pattern offers a solution that balances the forces in the problem
- **Consequences:** this section describes how the use of the pattern affects the system.

- **Hardware implications:** The application of the pattern has some hardware implications that should be taken into account.
- **Portability:** In E.S, it is common to have to port the software to other hardware platforms; this section describes how the pattern affects the portability.
- **Overall strengths and weaknesses:** summarizes pros and cons of the pattern.
- **Related patterns and alternative solutions:** discusses alternative solutions and/or related patterns that could be of interest.

PATTERN "MINI-ME"

Context: In a battery power embedded system (i.e. the cell phone), there are long periods of low activity interrupted by short periods of very intensive activity that can't be predicted in advance, Some tasks need to run continuously (for example the ones related to the cell network management). These tasks must be executed in both periods: when the cell phone is idle and when the user is using the cell phone with a specific purpose (phone call, gameplay, document reading, etc).

Problem: When running on a battery powered embedded system, it's necessary to preserve the battery power of the system and there are tasks that must be executed in periods of very intensive activities as well as low activities.

Forces:

- Low power cores are better for power saving in low activity periods.
- High performance cores are better for application performance in high activity periods.
- There are tasks that must run in both kind of periods.
- When different cores have the same ISA, the software is easier to develop and maintain (software upgrades).

Solution: Implement the system using an AMP processor with high performance cores and low power cores with the same ISA, because the way to preserve battery power is using a processor with less power requirements. Also, because both kind of cores have the same ISA, for the point of view of the software that must be executed in both periods (high activity and power saving), it is indifferent in which core is running at any time. When both type of cores share the same ISA, the low power cores became a smaller version of the high performance cores, so the low power cores are "mini" high performance cores.

Consequences: Because both type of cores share the same ISA, the operating system has no restriction about the core where a process/task can run. But when the process/task run in the low power core, there is a performance penalty, so the OS scheduler should take this into account for the process/task execution priority and processor time allocation.-

Hardware implications: Some SMP multicores have a configuration register for the system clock of each core. By configuring different values for that register, we can convert a

SMP system in a AMP system where all cores have the same ISA.

Portability: As both type of cores share the same ISA, the software can run on any core without modification.

Overall Strengths and Weaknesses:

+ The AMP with cores having the same ISA has a minimum impact on the software that runs on it and the system has less power consumption.

- There is a performance penalty when the process/task runs on the low power cores and the operating system could not be aware of that.

Related patterns and alternate solutions: This pattern is well suited when the processes/tasks don't have real time requirements, because they run sometimes in a high performance core and sometimes in a low power core, so the execution predictability and performance could be very

difficult to establish. For this type of processes/task, the "Dedicated Processor" or "Optimized Execution" patterns would be better. This pattern is a specialization of the "Asymmetric Multiprocessing" pattern.

Known Uses: A commercial AMP processor that applies this pattern is the Samsung Exynos 5 Octa, which has 4 high performance cores (ARM Cortex A15) and 4 low power cores (ARM Cortex A7) inside the AMP. Both kind of cores have the same processor technology and ISA (ARM V7-A). Other commercial processor that applies this pattern is the AllWinner A80. AllWinner processors are very popular in tablets and set-topboxes that run Android OS. ARM Company calls this pattern "The big.LITTLE Technology". The pattern's name comes from "Dr. Evil" character from "Austin Powers" movie; who has a clone of himself, but with 1/8 of his height, he calls the clone "Mini Me". The low power cores are the "Mini Me" of the high performance cores.

Implementation of an AXI-compliant lock-in amplifier on the RedPitaya open source instrument

L. H. Arnaldi

Laboratorio Detección de Partículas y Radiación
Centro Atómico Bariloche - Instituto Balseiro
Bariloche, Rio Negro (8400)
Email: arnaldi@cab.cnea.gov.ar

Abstract—This paper describe the implementation of a lock-in amplifier (LIA) which conforms to the AXI standard for on-chip communication. The design and implementation is based on the Zynq[®] field programmable gate array (FPGA) present in the open-source instrument RedPitaya. The designed architecture is a mixed solution between VHDL hardware modules and software modules, running within an ARM CPU. General design criteria are summarized for the all-digital implementation. Simulation results for the performance of some of the designed intellectual property (IP) blocks are also presented to verify its operation.

Index Terms—Phase-sensitive detection, FPGA, Zynq, RedPitaya, all-digital phase locked loop.

I. INTRODUCTION

Lock-in amplifiers are measurement instruments widely used in science and engineering. They are powerful instruments to detect and measure periodic signals in presence of a large quantity of noise. The LIA use a technique known as phase-sensitive detection to single out a component of interest from a noisy signal, at a specific reference frequency and phase, attenuating the rest of the spectrum. The output is a DC component proportional to the signal amplitude at the frequency of interest. The functional description of this technique is well known and is widely addressed in literature [1]–[3].

For a long time lock-in amplifiers were based on analog electronics. However, digital LIAs have become increasingly popular in many experimental setups due to the combination of flexibility, cost and performance. If sufficient precision is used in all calculations, the linear response range of each digitized component can be arbitrarily large, making the design process easier. Also, using digital multipliers, almost ideal digital mixers and filters can be created.

Recently, several FPGA-based lock-in amplifiers for different applications (including one with sub-ppm resolution) have been reported [4]–[7]. Advantages of FPGA-based LIA systems were investigated in [5] and [6], where emphasis was given to show the FPGA's flexibility in fulfilling more than a single digital task at the same time and aiming to obtain great flexibility on key design parameters, such as sampling and lock-in frequency. In [5] there is an implementation using a Zynq[®] FPGA, tailored to a low-frequency application. None of these publications, however, presents a procedure for designing an all-digital LIA from scratch, tailored to a RedPitaya board, using the AXI standard and showing the details of the calculations, as it is intended in this paper.

The implementation of the lock-in amplifier used is the well-known dual-phase LIA [6]. Details are given for the design and construction of the FPGA-based LIA along with some testing of the final product.

The analog signals are acquired using a dual-channel, 14-bit A/D converter and mixed with a pair of quadrature reference signals. The reference signal can be derived from an external source or generated internally by a programmable direct digital synthesis module. The external reference signal is sampled and then converted to a pulse train by a zero crossing detector module. This pulse train feeds the reference input of an all-digital phase locked loop (ADPLL). The ADPLL generates the required quadrature sine waves. Once mixed down, the signals are fed through programmable digital filters to a dual-core ARM CPU for the determination of the amplitude and phase of the input signal. The output signals can be directly routed to the analog output ports, from the ARM CPU. The ARM processor acts as a high level control system allowing to control the filters operation for example, giving the advantage of maximal flexibility, as well as a useful tool in debugging.

Constructing these modules in an FPGA allows all operations after A/D conversion to be done digitally at high speed. The target platform for this system is a Zynq[®] (xc7z010c1g400-1) FPGA, present in the RedPitaya open-source board, but the design concepts can be easily adapted to another platforms. As a reasonable expectation, input signal sine waves operating at frequencies of up to some MHz should be easily analyzed at the ADC's clock rate of 125 MHz.

The following sections describe the design of the digital electronics of the realized instrument. Section II gives an overview of the AXI standard. In Section III the main characteristics of the RedPitaya open-source instrument are presented. In Section IV, a design flow for the LIA is described. Section V presents some simulations of the designed blocks. Finally, conclusions are provided in Section VI.

II. THE AXI STANDARD

AXI stands for Advanced eXtensible Interface, and the current version is AXI4, which is part of the ARM AMBA[®] 3.0 open standard. Many devices and IP blocks produced by third party manufacturers and developers are based on this standard.

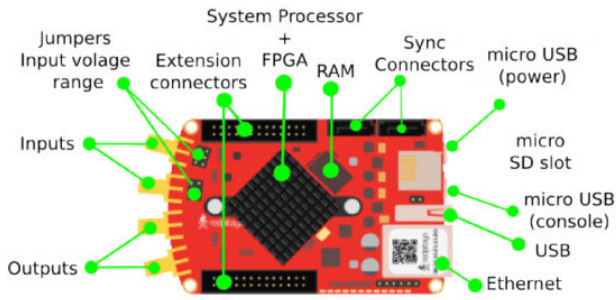


Fig. 1. RedPitaya board hardware overview. (Source <https://wiki.redpitaya.com/>)

The AMBA standard was originally developed by ARM for use in microcontrollers, with the first version being released in 1996. Since then, the standard has been revised and extended, and it is now described by ARM as “the *de facto* standard for on-chip communication” [8]. The focus is now on System-on-Chip (SoC), including SoCs based on FPGAs or, in the case of Zynq[®], a device which includes FPGA fabric [9].

Nowadays there are three flavours of AXI4, each of which represents a different bus protocol, as summarised below. The choice of AXI bus protocol for a particular connection depends on the desired properties of that connection.

- AXI4 [10] : For memory-mapped links, and providing the highest performance: an address is supplied followed by a data burst transfer of up to 256 data words (or “data beats”).
- AXI4-Lite [10] : A simplified link supporting only one data transfer per connection (no bursts). AXI4-Lite is also memory-mapped: in this case an address and single data word are transferred.
- AXI4-Stream [11] : For high-speed streaming data, supporting burst transfers of unrestricted size. There is no address mechanism; this bus type is best suited to direct data flow between source and destination (non memory mapped).

For this case then, the best choice is to use the AXI4-Stream protocol in the LIA implementation, due to its inherent capacity for high-speed data transmission.

III. THE REDPITAYA BOARD

Fig. 1 shows an overview of the RedPitaya board. RedPitaya is a low-cost (~ 400 USD) analog signal generation/measurement electronics [12]. Its specifications are shown in Table I. The RedPitaya is equipped with dual fast ADC and dual fast DAC, i.e., it is easy to acquire and generate two sine waves of arbitrary phase difference simultaneously. These are the important specifications for its usage as LIA. It runs a Linux on it and can be controlled/accessed via several ways: browsers via PC or tablet (it can run a web server), usb-serial console, and SSH protocol. The user can easily write/modify source code to control the FPGA and CPU by using open-source software. For example, it is possible to implement own

TABLE I
SPECIFICATIONS OF REDPITAYA

Processor	Xilinx Zynq-7010 SoC ARM dual core CPU and Artix 7 FPGA
System disk	MicroSD
RAM	512MB
Access	USB console, Ethernet, WiFi dongle
Power	5V x 2A max., 0.9A typ.
Fast DAC	Dual channel, 14-bit, 125MSPS
Fast ADC	Dual channel, 14-bit, 125MSPS
Other I/Os	Slow DAC x 4, slow ADC x 4, digital I/O x 16, daisy chain connector

modules on FPGA by using VHDL or Verilog, and run C programs on the ARM CPU. Various open-source applications are also available.

IV. SYSTEM ARCHITECTURE

The implemented system follows the Xilinx paradigm in constructing the new designs like interconnected intellectual properties blocks, so each module in this design was programmed and constructed like an intellectual property block. The modules were programmed using the Xilinx Vivado[®] EDA software [13]. There is a top level module, called *system_wrapper.vhd*, which define all other supporting VHDL modules. These modules are analysed in detail in this section. All the IP blocks conforms to the AXI4-Stream standard, which is best suited for this application. The top level implementation of the LIA can be seen in Fig. 2 as a block diagram.

A. ADC and DAC interfaces

These modules act as interfaces between the FPGA and the external world. They read and write data to the external integrated circuits through a 14-bit bus. The design of the IP blocks is targeted to the DAC1401D125 IC for the D/A conversion and to the LTC2145-14 IC for the A/D conversion.

B. Zero crossing detector (ZCD)

This module takes the digitized reference input and waits for zero crossing. In order to prevent noise in the triggering signal, a hysteresis was added for detecting both positive and negative going zero crossing. To prevent any phase shifts, the output is always triggered at the zero crossing. The external reference is then converted to a pulse train to feed the reference input of the all-digital phase locked loop block.

C. All-digital phase locked loop

A phase locked loop (PLL) is needed in order to lock to an externally generated reference signal. The implementation of this module follows an all-digital phase locked loop architecture. The input signal is the digitized data stream of the reference signal and the output is a pair of locked quadrature sine waves. It can also use an internally generated reference signal, provided by a programmable direct digital synthesis

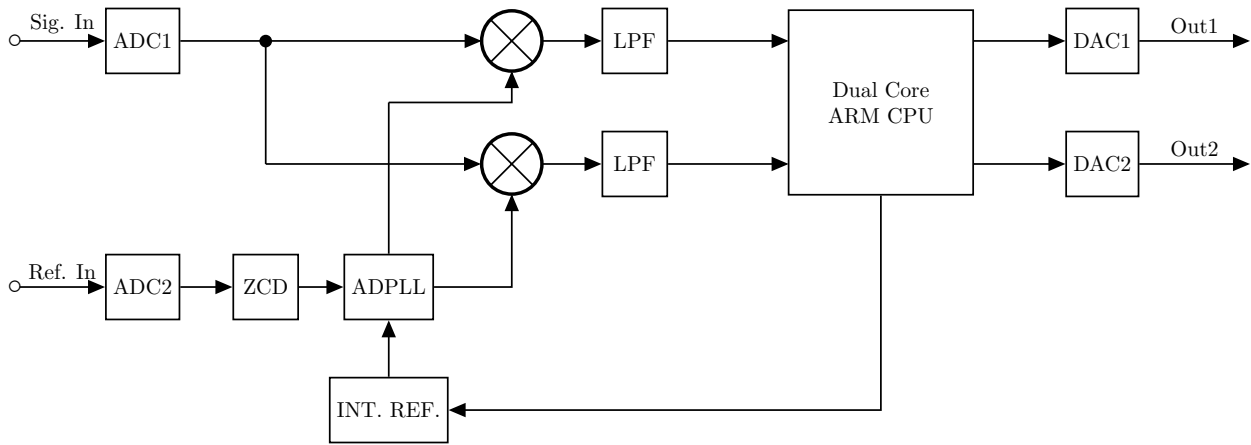


Fig. 2. Block diagram of the implemented LIA. It is composed of two A/D interfaces, a zero crossing detector (ZCD), an all-digital phase locked loop (ADPLL), a programmable direct digital synthesis unit (INT. REF.), two signed multipliers, two recursive low-pass filters (LPF), the dual core ARM CPU and two D/A interfaces.

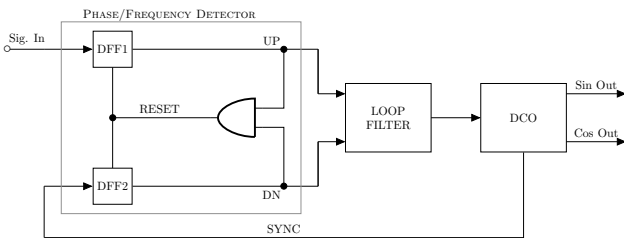


Fig. 3. ADPLL. As normal PLL the building blocks of the ADPLL are phase detector, loop filter, digital-controlled oscillator (DCO), and optional divide by N element.

block (INT. REF. in Fig. 2), which can be activated/deactivated by the processor.

The presented PLL can be approximated with a model of a continuous PLL. This simplifies the calculation of some generic PLL parameters. The chosen topology for the ADPLL can be seen in Fig. 3. The building blocks include a phase detector, a loop filter, a digital-controlled oscillator (DCO), and optionally a divide by N element (not shown in Fig. 3).

For simplicity, both design and conceptual, this document is focused on a design approach for type-II second-order all-digital phase locked loop [14], [15]. Details of the necessary calculations are given in [16] and [17]. In general, the approach showed here can be extended for the design of all-digital PLLs with different types and orders.

1) *Phase/Frequency sensitive detector*: A phase detector is a circuit capable of delivering an output signal that is proportional to the phase difference between its two input signals. In this design, a digital phase/frequency detector (PFD) is implemented. The labeled box in Fig. 3 shows the schematic of the PFD. Because the output signal of the PFD depends on phase error in the locked state of the PLL and on the frequency error in the unlocked state, a PLL that uses the PFD will lock under any condition, irrespective of the type

of loop filter used. For this reason the PFD is the preferred phase detector in PLLs [18]. The PFD is built from two D-type flip flops, whose outputs are denoted UP and DN (down), respectively. The PFD can be in one of three states:

- UP = 0, DN = 1 → State = -1
- UP = 0, DN = 0 → State = 0
- UP = 1, DN = 0 → State = 1

The actual state of the PFD is determined by the positive transition of the signals SYNC and Sig. In as shown in Fig. 3.

2) *Loop filter*: The loop filter acts upon the output of the phase detector to remove unwanted high frequency terms, and produce the signal that drives the DCO. The design of the loop filter is vital in defining the overall characteristics and behaviour of the PLL.

A digital equivalent of an analog second-order loop filter consists of a proportional path with a gain K_P and an integral path with a gain K_I . It is called a *Proportional + Integral* (PI) or *Lead-Lag* filter. Fig. 4 shows a schematic representation of the loop filter. The parameters of a digital loop filter K_P and K_I can be obtained from the parameters of an analog loop filter by using the bilinear transform (1)

$$s = \frac{2}{T_s} \frac{1 - z^{-1}}{1 + z^{-1}} \quad (1)$$

where T_s is the sampling time of the discrete system.

Eq. (2) gives the s-domain transfer function of the PI loop filter and eq. (3) is the transfer function after the application of the transformation (1):

$$\frac{u(s)}{e(s)} = K_P + \frac{K_I}{s} \quad (2)$$

$$\frac{u(z)}{e(z)} = K_P + K_I \frac{1}{1 - z^{-1}} \quad (3)$$

where K_P and K_I are the only unknown parameters that need to be determined for the ADPLL design at this stage.

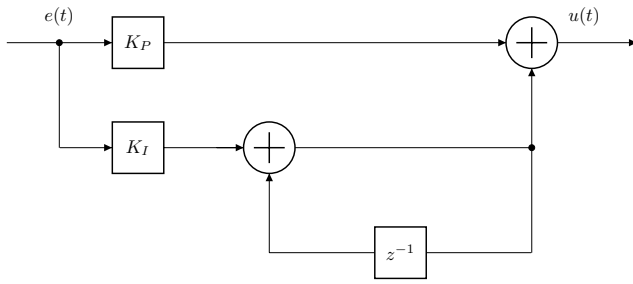


Fig. 4. Building blocks of the loop filter, it is a proportional + integral (PI) filter. The proportional output is $+K_P$ or $-K_P$ and the integral output is the accumulation of $+K_I$ or $-K_I$ over time.

Since $e(t)$ takes a value of either $+1$ or -1 , the proportional output will be $+K_P$ or $-K_P$. On the other hand, the integral output is the accumulation of $+K_I$ or $-K_I$ over time. This filter has one pole on the unit circle.

3) *Digital-controlled oscillator*: The programmable oscillator is implemented as an high accuracy digital-controlled oscillator (DCO). The loop filter provides a frequency tuning word (FTW) at the input of the DCO. The FTW is a fixed point 32-bit digital word, in 24.8 format, setting the working frequency. The DCO generates a pair of quadrature sine waves (14-bit resolution) to feed the mixers and a digital square wave of the same frequency feeding the phase comparator (*SYNC* signal in Fig. 3).

The output frequency of the DCO can be calculated as:

$$f_{out} = \frac{A f_{clk} FTW}{2^{N-P}} \quad (4)$$

where f_{clk} is the system clock, FTW is the frequency tuning word output from the loop filter, N is the word length of the phase accumulator ($N = 50$ in this case), P is the number of bits after the decimal point and A is a proportional factor obtained from:

$$\frac{A f_{clk}}{2^{N-P}} = 1 \quad (5)$$

The frequency resolution of this module, for 8-bit after the decimal point is:

$$\Delta f = \frac{1}{2^8} = 3.9 \times 10^{-3} Hz \quad (6)$$

D. Low-pass filters

The low-pass filters were designed to emulate a simple analog RC circuit with adjustable time constant. The modules were implemented as single pole recursive filters, allowing easy on-the-fly calculation of filter parameters when changing settings.

The single pole filter output is given by:

$$y[n] = a_0 x[n] + b_1 y[n-1] \quad (7)$$

The coefficients are found from these simple equations:

$$a_0 = 1 - x, \quad (8a)$$

$$b_1 = x \quad (8b)$$

The characteristics of the filter are controlled by the parameter, x , a value between zero and one. Physically, x is the amount of decay between adjacent samples. The higher the value of x , the slower the decay.

The value for x can be directly specified, or found from the desired time constant of the filter. Just as $R \times C$ is the number of seconds it takes an RC circuit to decay to 36.8% of its final value, d is the number of samples it takes for a recursive filter to decay to this same level:

$$x = e^{-1/d} \quad (9)$$

For instance, a sample-to-sample decay of $x = 0.86$ corresponds to a time constant of $d = 6.63$ samples. There is also a fixed relationship between x and the $-3dB$ cutoff frequency, f_c , of the digital filter:

$$x = e^{-2\pi f_c} \quad (10)$$

This provides three ways to find the “a” and “b” coefficients, starting with the time constant, the cutoff frequency, or just directly picking x [19]. The parameters can be generated by the ARM CPU from a user selected time constant.

E. Lock indicator

Although not shown in the schematics, a lock indicator was also implemented. The PFD lends itself to an effective and simple scheme, as described below. A two-input OR gate takes as its inputs the UP and DN outputs of the PFD. When the PLL is locked with small phase error, neither UP nor DN is true for any but very short intervals during each comparison cycle. When the PLL is out of lock, either UP or DN will be true, on average over many cycles, for 50% or more of the time. The basis of lock detection is to pass the output of the OR gate through a smoothing filter to extract its average dwell time in the true state and to compare that average against a suitable threshold (say, 25% average true dwell time). The PLL is deemed to be locked if the average true time is below the threshold and unlocked if the average true time is above the threshold [14]. The lock detector also include a timer that requires the lock indication to persist for a specified time interval before phase lock is declared. The timer is started when the average dwell time falls below threshold and reset to zero whenever the threshold is exceeded before the timer reaches its specified interval.

V. SIMULATIONS

Simulations were carried in order to test the behaviour of some of the blocks of the system. Fig. 5 shows the output of the DCO when it is applied a $FTW = (2560000000)_{10}$, corresponding to a generated output frequency of $f = 10$ MHz. In the same figure can be seen the two outputs I and Q (labeled

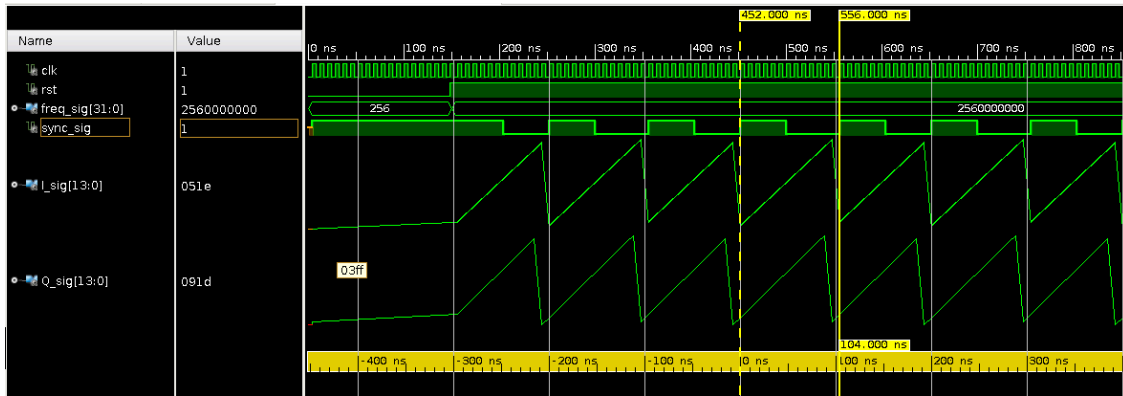


Fig. 5. Simulation of the output of the DCO. The FTW input is $(256000000)_{10}$, giving a frequency of 10 MHz at the output.

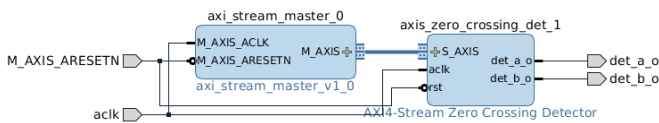


Fig. 6. Block design to test the ZCD module as seen in Vivado Design Suite. The module was designed taking into account the two ADC's of the RedPitaya board, hence the two outputs det_a_o and det_b_o .

I_sig and Q_sig respectively) and the $sync_sig$ signal ($SYNC$ signal in Fig. 3).

An arrangement was programmed to test the compliance of the modules with the AXI4-Stream standard. An AXI master module was coded to manage the interface and test the different blocks. As can be seen from Fig. 6, the ZCD module has the necessary interface signals to conform to the AXI4-Stream standard. The same test were carried for all the modules in this design. As an example, Fig. 7 shows the simulation of the AXI interface for the ZCD module.

Fig. 8 shows the behaviour of the phase detector output signals (UP and DN) when the input signal is in phase with the reference ($SYNC$) from the DCO and Fig. 9 shows the same output signals when they are not in a lock condition.

VI. CONCLUSIONS

The design of an highly configurable, low-cost, FPGA-based lock-in amplifier has been presented in this work. It was designed from scratch aiming to provide a flexible platform to use in different experiments.

The designed IP blocks conforms to the AXI standard, the *de facto* standard for on-chip communication. The behaviour of the modules were tested by simulations, showing good agreement with the standard.

The implementation of the system showed to be highly configurable and flexible, thanks to the use of the Zynq[®] system on chip (SoC) from Xilinx.

There are still future work in the characterization of the system performance through the use in-place with some experiments.

The proposed LIA architecture is simple, can be easily replicated following the design guidelines showed here, so resulting particularly suitable for its use as an educational tool in courses of FPGA-based systems. With respect to conventional LIAs typically working at low operating frequencies, the developed circuit is capable to manage high frequency signals, of the order of MHz.

REFERENCES

- [1] S. R. Systems, "About Lock-In Amplifiers," dec 1992.
- [2] P. Horowitz and W. Hill, *The Art of Electronics*. Cambridge University Press, mar 2006, vol. 1.
- [3] R. E. Simpson, *Introductory Electronics for Scientists and Engineers*. Addison-Wesley, abr 1987.
- [4] G. Gervasoni, M. Carminati, and G. Ferrari, "FPGA-based lock-in amplifier with sub-ppm resolution working up to 6 MHz," *2016 IEEE Int. Conf. Electron. Circuits Syst. ICECS 2016*, pp. 117–120, 2017.
- [5] G. Giaconia, G. Greco, L. Mistretta, and R. Rizzo, "Exploring FPGA-Based Lock-In Techniques for Brain Monitoring Applications," *Electronics*, vol. 6, no. 1, p. 18, 2017. [Online]. Available: <http://www.mdpi.com/2079-9292/6/1/18>
- [6] G. Macias-Bobadilla, J. Rodríguez-Reséndiz, G. Mota-Valtierra, G. Soto-Zarazúa, M. Méndez-Loyola, and M. Garduño Aparicio, "Dual-phase lock-in amplifier based on FPGA for low-frequencies experiments," *Sensors (Switzerland)*, vol. 16, no. 3, 2016.
- [7] A. Restelli, R. Abbiati, and A. Geraci, "Digital field programmable gate array-based lock-in amplifier for high-performance photon counting applications," *Rev. Sci. Instrum.*, vol. 76, no. 9, 2005.
- [8] ARM, "AMBA Open Specification." [Online]. Available: <http://www.arm.com/products/system-ip/amba/amba-open-specifications.php>
- [9] L. H. Crockett, R. A. Elliot, and M. A. Enderwitz, *The Zynq Book Tutorials for Zybo and Zedboard*. Strathclyde Academic Media, ago 2015.
- [10] ARM, "AMBA AXI and ACE Protocol Specification: AXI3, AXI4, and AXI-Lite, ACE and ACE-Lite." [Online]. Available: <http://www.arm.com/products/system-ip/amba/>
- [11] —, "AMBA 4 AXI4-Stream Protocol Specification." [Online]. Available: <http://www.arm.com/products/system-ip/amba/>
- [12] RedPitaya, "RedPitaya Open Source Instrument." [Online]. Available: <http://www.redpitaya.com/>
- [13] Xilinx Inc., "Vivado Design Suite." [Online]. Available: <https://www.xilinx.com/products/design-tools/vivado.html>
- [14] F. M. Gardner, *Phaselock Techniques*. Wiley, jul 2005, vol. 1.
- [15] M. Rice, *Digital Communications: A Discrete-Time Approach*. Prentice Hall, jul 2009.
- [16] V. Kratyuk, P. K. Hanumolu, U.-K. Moon, and K. Mayaram, "A Design Procedure for All-Digital Phase-Locked Loops Based on a Charge-Pump Phase-Locked-Loop Analogy," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 54, no. 3, pp. 247–251, mar 2007. [Online]. Available: <http://ieeexplore.ieee.org/document/4132962/>

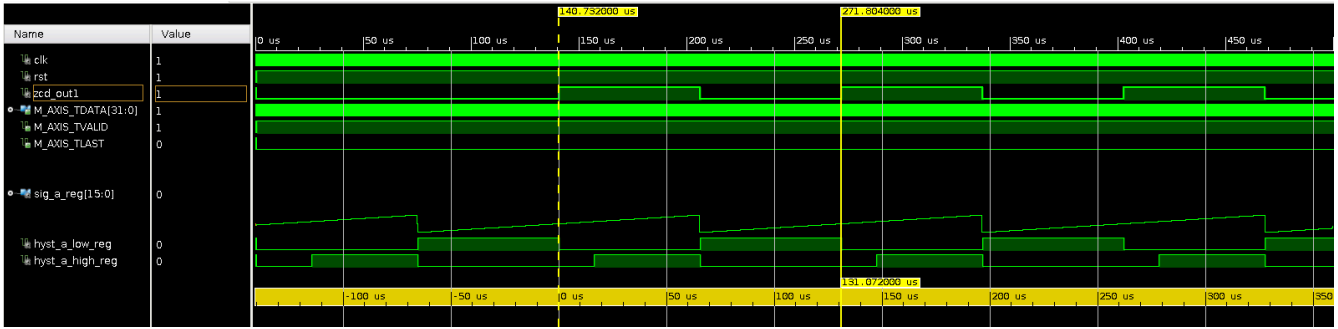


Fig. 7. Simulation of the output of the ZCD module and its AXI-Stream interface. The output is a signal of $f = 7629.3$ Hz (labeled *zcd_out1*). Can be seen the two registers holding the high and low hysteresis values (*hyst_a_high_reg* and *hyst_a_low_reg*, respectively). The block complies with the AXI standard.

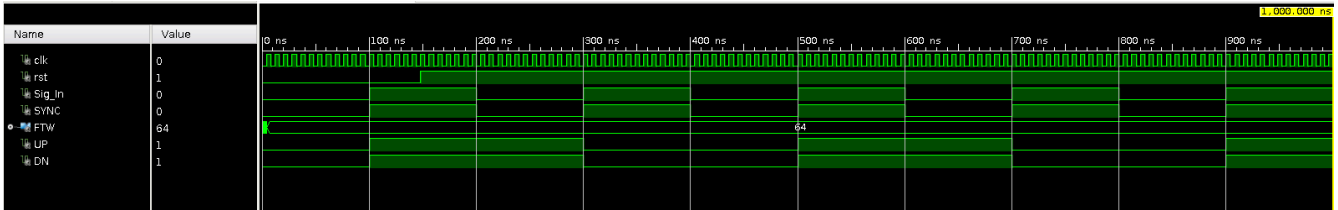


Fig. 8. Output of the phase detector, UP and DN signals. The input signal, *Sig_in* is locked to the reference, *SYNC*.

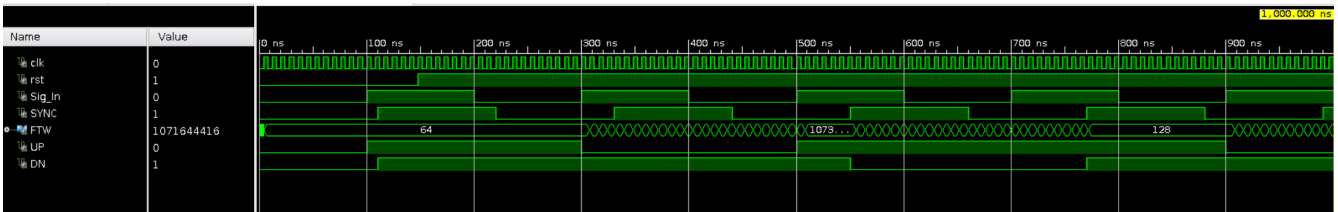


Fig. 9. Output of the phase detector, UP and DN signals. The input signal is not locked to the reference.

- [17] M. Kumm, H. Klingbeil, and P. Zipf, "An FPGA-Based Linear All-Digital Phase-Locked Loop," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 57, no. 9, pp. 2487–2497, sep 2010. [Online]. Available: <http://ieeexplore.ieee.org/document/5456240/>
- [18] R. E. Best, *Phase-Locked Loops*. McGraw Hill Professional, jun 2003, vol. 1.
- [19] S. W. Smith, *Digital Signal Processing: A Practical Guide for Engineers and Scientists*. Newnes, feb 2003, vol. 1.

Real-Time Gaze-Tracking Embedded-System

Cristian E. Ordoñez*[†], Eduardo L. Blotta* Juan I. Pastore*

*Laboratorio de Procesamiento Digital de Imágenes, Facultad de Ingeniería,
ICYTE-CONICET, U.N.M.D.P., Mar del Plata.

[†] Departamento de Informática, Facultad de Ingeniería, U.N.M.D.P.
ordonezc@fi.mdp.edu.ar

Abstract—This paper proposes a video-based Real-Time Gaze-Tracking system which allows to identify gaze commands to drive an electric powered wheelchair. The main aspect of the method is its ability to be implemented in a portable computing system, reduced both in processing capacity and in RAM memory. The Gaze-Tracking system is based on the Hough Transform, the use of shape features and the dark pupil effect. The method performance was validated using an own database, built under different ambient lighting conditions.

Index Terms—Real-Time-Processing, Embedded System, Hough Transform, Gaze-Tracking, Shape features.

I. INTRODUCTION

Severe injuries in the upper portion of the spinal cord, age, or diseases that affect motor neurons (MNDs), may prevent a person to perform tasks such as driving a common electric-powered wheelchair (EPW), usually controlled by a hand joystick.

Nowadays, there is a wide field of human-computer interfaces (HMIs) that allow to control an EPW without the need of using hands. Within this type of technology there are systems based on Electromyography (EMG) signals, Electroencephalogram (EEG) signals, sip and puff (SNP) sensors, tongue movement sensors, chin movement joysticks, and eye tracking methods.

EMG signal-based systems employ a sensor that converts muscle activity, usually from some muscle of the user's face, into small electrical impulses that are used to control an EPW [1] [2].

Brian-Computer interfaces (BCI) based on EEG signals have as disadvantage the high cost that involves linking the human brain to the device.

In SNP systems the user gives commands to the wheelchair by “sipping” and by “puffing” on a pneumatic tube like the one used in [4]. This method responds to the amount of pressure applied to the pneumatic tube and whether the sign of the pressure is negative or positive. These systems require a certain precision in each sip and puff and are not suitable for people with reduced respiratory capacity.

Interfaces that allow to control an EPW through the tongue are usually invasive, since they use an intra-oral device that registers the variations of inductance due to the movements of a ferromagnetic piercing that is placed in the user tongue [6] [7].

Chin-based devices use a force-sensing joystick usually controlled by neck flexion, extension and rotation [8].

Finally, eye tracking-based systems include a wide variety of methods. Within this type of systems are the devices based on sensors, which employ electrodes placed around the eye (Electro-oculogram, EOG), and video-based systems, where a video camera captures eye movements and a computer saves and analyzes the information sent. An advantage of EOG is its low influence on lighting conditions, however, it is an invasive technique that may be considered impractical for everyday use [9].

Many researchers present video-based eye tracking systems, where they implement techniques based on pattern recognition, corneal reflection points, dark-bright pupil effect through infrared illumination, eye models and hybrid techniques [10]. The implementation of these techniques, when computing power resources are available, does not represent challenges. However, when the design is oriented to a portable battery-powered computing system, the challenge is greater as it seeks to maximize computing efficiency in order to minimize consumption.

This work is an extension of [11]. An hybrid method based on the Hough Transform (HT), the use of shape features and the “dark pupil” effect is proposed to implement a wheelchair control embedded system using a portable low-cost, low-power computing system, based on a 32-bit microcontroller. In addition to the advances made in the pupil detection method, an infrared light source is added to the system in order to improve the pupil-iris contrast and a geometric criterion is established to identify movement commands for the EPW.

The method proposes adjustments that improve the HT performance in the pupil detection and also reduce the number of operations to be executed.

This paper is organized as follows: section II presents the theoretical bases, eye safety issues using infrared light and hardware description, section III details the method and section IV shows the results.

II. MATERIALS AND METHODS

A. Hough Transform

When extracting shape information of an object in an image, the most relevant information is often obtained around the boundaries or edges of the object [14], therefore the edges are used as a discriminative property. The Hough Transform (HT) is a global processing technique that allows to identify and locate objects whose shape is parameterized by the equation $g(v, c) = 0$, where v is a vector of coordinates and c is a vector

of coefficients. This technique is robust to noise presence and also to incomplete edges. HT turns a segmentation problem in the image space, in a peak detection problem in the parameters space. Paul Hough developed the HT in 1962 to detect straight lines [15]. Subsequently, Duda and Hart improved the Hough technique extending it to detect other types of parametrical curves [16]. Merlin and Faber [17] probed that the HT can be generalized to detect arbitrary shapes with a certain scale and orientation. Ballard eventually extended the scope of the work, formulating the Generalized Hough Transform (GHT) to efficiently detect arbitrary shapes with any orientation and scale [14] [18].

Starting from the premise that pupil has a circular shape, in this paper we propose to apply the Circle Hough Transform (CHT) in order to detect the pupil position. Although the premise of pupil circularity is not always satisfied, the CHT algorithm is good enough for our purpose. CHT is similar to the standard or linear HT, only the space parameters are different, corresponding in this case to the analytical model of the circle [19]:

$$(x - c_1)^2 + (y - c_2)^2 = c_3^2 \quad (1)$$

where (c_1, c_2) are the coordinates of the center of a circle of radius c_3 passing through (x, y) , on a cartesian coordinate system.

It is usual to use the parametric representation of the circle to implement the CHT [19]:

$$\begin{aligned} x &= c_1 + c_3 * \cos(\theta) \\ y &= c_2 + c_3 * \sin(\theta) \end{aligned} \quad (2)$$

where $\Theta \in (0, 2\Pi]$.

B. Shape features

The shape descriptor of a region is a measure of the degree of similarity with a given shape. The proposed method uses this measure as a criterion of discrimination between Hough candidates that produce local maxima in the accumulator space.

Most traditional shape descriptors are mere combinations of size parameters (area, perimeter and diameter) arranged in such a way that dimensions get canceled. Table I lists the most used shape descriptors. Each of these parameters captures an aspect of shape [12].

Shape features	Reference
$F = \frac{4\pi A}{P^2}$	Shape factor
$R = \frac{4A}{\pi D_{max}^2}$	Roundness
$R_a = \frac{D_{max}}{D_{min}}$	Aspect relationship
$E = \frac{l_f}{a_f}$	Elongation
$C = \frac{P_c}{P}$	Convexity

TABLE I: Shape features

Where:

A = area enclosed by the contour;

D_{min}/D_{max} = minimum/maximum diagonal of the area enclosed by the contour;

l_f = length: defined as the length along the axis of the curvilinear shape;

a_f = maximum width of the curvilinear shape;

P = perimeter;

P_c = convex perimeter: defined as the length of a convex line circumscribing the object to be measured.

C. Hardware Description

The proposed system runs on a *Discovery* kit, which is based on a *STM32F407VGT6* Micro Controller Unit (MCU), delivering 180 DMIPS @ 144 MHz, see Fig. 1. A base board, connected to the STM32F4 Discovery provides a micro SD Card slot and extension connectors for the LCD and camera boards. A digital camera board, featuring a 1.3 *Megapixel* CMOS OV9655 sensor and a 3"5 LCD board, with touch screen capability compose the whole system. More details on [24].



Fig. 1: Discovery Kit: main board, base board, camera and display



Fig. 2: Test bench

In order to evaluate the system performance a test bench was developed, as shown in Fig. 2. It allows to place the Discovery board and the LCD display to visualize the detection performed. In addition, it is possible to rotate and adjust the height at which the screen is located to improve the user comfort.

Regarding the Gaze Tracking glasses, an infrared light source was positioned next to the camera to illuminate the user's retina, as shown in Fig. 4.

Every camera sensor has the ability to capture infrared light but an infrared blocking inner filter protects the CCD sensor from high energy infrared light such as sunlight. This blocking filter was removed to allow the camera to capture that specific range of light spectrum. In addition, a filter to block visible light was added, to avoid undesired reflections on the cornea. To ensure the user's eye health it is important to take into account certain considerations. Any source of light radiation in large concentrations can cause damage to the human eye. The International Electrotechnical Commission (IEC) has established standards for human eye safety with various sources of light radiation. Document IEC-60825 deals with laser sources, and IEC-62471 deals with lamp sources and LEDs, under normal continuous or pulsed operation. This last document defines the maximum infrared radiation exposure (for wavelengths around 770 to 3,000 nm) allowed on the cornea for day-long continuous exposure. For this reason, the IR light source was set to deliver an irradiance of 3 mW/cm^2 , which is well below the IEC recommended maximum of 10 mW/cm^2 .



Fig. 4: Eye tracking glasses

III. IMPLEMENTATION

Figure 3 shows the block diagram of the proposed system. The method is composed of four stages, which are described below:

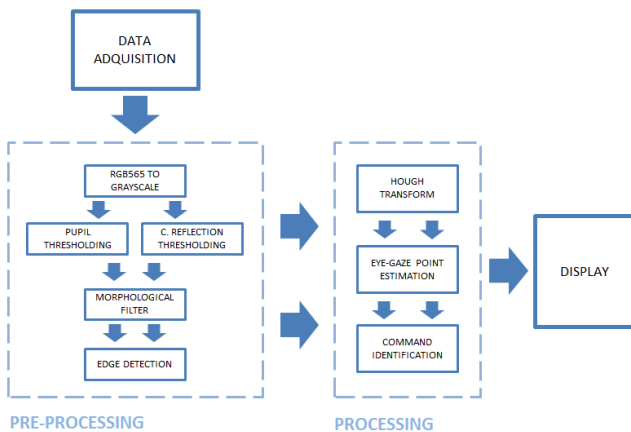


Fig. 3: Proposed method

1) Data acquisition:

Data acquisition is done through the video camera which is mounted on a support attached to the user's glasses, as shown in 4. This camera captures images from the user's eye and sends them to the processing unit. When the eye is illuminated by an infrared light positioned outside the optical axis, much of this light is absorbed by the pupil causing it to look darker. This effect is known as the "dark pupil effect", which improves the pupil-iris contrast, making easier the pupil segmentation.

2) Pre-processing:

The pre-processing stage receives the data sent by the camera and conditions it for the next instance.

Although the acquired images are in RGB-565 format, the processing is performed only on the blue channel since it has the highest entropy. This data set is used as input of a bunch of algorithms that allow to identify the center of the pupil and the center of cornea reflection. The blue channel information is applied to two thresholding algorithms. Based on the dark pupil effect, the pupil thresholding seeks to obtain the darker pixels. On the other hand, the corneal reflection produces a region of pixels of high-intensity values caused by the reflection of the infrared light on the user's eye, as shown in Fig. 6, so the thresholding logic seeks to obtain the brighter pixels.

Then, an opening morphological filter is applied using a linear structuring element of dimension 1×3 [19], allowing to eliminate noise and isolated pixels due to previous thresholding.

Finally, edge detection is performed employing a 3×3 kernel Laplacian operator [13].

3) Pupil center and corneal reflections points:

The circular nature of the pupil and the reflection of the cornea cause the problem to be reduced to the search for centers of circumference through the HT. One of the great limitations of this transformation is that there is none feature analysis of the candidates to be centers of circumferences. Usually these are obtained only by voting in the parameter space, selecting the global maxima in the accumulation matrix [18]. The proposed method suggests some modifications that reduce the number of computation operations and improve the criterion of selection of circumference candidates.

Fig. 5 illustrates how the method works, where p_1 and p_2 are two arbitrary points belonging to the edge of the searched circumference. Assuming that such circumference has a radius R_o between R_{min} and R_{max} , the angular space is discretized in n points, and for each border pixel, instead of computing all the circumferences crossing over that pixel, we compute only those that have that pixel as center. The point p_3 , where the proposed circumferences intersect, is the point where the number of votes in the accumulation space is maximum, and matches with the center of the searched circumference.

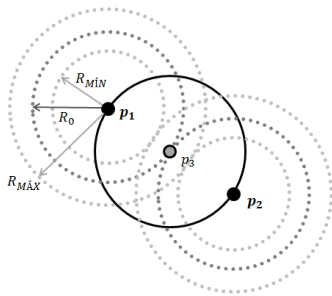


Fig. 5: Hough Transform implementation

The parametric space is obtained by Eq. 2.

Some imperfections in the image, in the edge detector, or simply the shape of the border under analysis, often produce regions that cause local or global maxima in the accumulator, despite not resemble at all to a circumference. To eliminate all these regions we propose to use the roundness shape features as a criterion of prior discrimination between regions that can produce local maxima, reducing in this way the number of candidates to be pupil centers. The candidate regions will be those whose roundness descriptor is approximately equal to 1.

4) Eye-gaze point estimation and command identification:

In eye tracking systems composed of a single camera and a single light source, if the user's head position remains fixed, relative to the camera position, it is possible to approximate the gaze direction, relative to the optical axis of the camera, computing the vector distance between the pupil centers and cornea reflection [22]. Thus, the gaze direction c_x, c_y is obtained through the distance vector between the pupil center c_{xpupil}, c_{ypupil} and the center of the cornea reflection c_{xIR}, c_{yIR} , according to the Eq. 3.

$$\begin{aligned} c_x &= c_{xpupil} - c_{xIR} \\ c_y &= c_{ypupil} - c_{yIR} \end{aligned} \quad (3)$$

It is known that the right human eye can rotate to the left up to 60° , to the right up to 90° , and upwards to 46° [23]. This means that the amplitude of eyeball movement is limited to the gaze direction. We use these known relationships as criterion to define the distance relationships between the center of the pupil and the center of the cornea reflexion to identify each movement command of the wheelchair: left, right and forward.

IV. RESULTS

To evaluate the performance of the method we used our own database, composed by two hundred pupil images acquired through the Discovery kit. These images belong to different people of different gender, age and eye color which were obtained under low, medium and high ambient lighting conditions. The performed tests include a total of 90 movement commands: 30 left, 30 right and 30 forward.

Environment Illumination	CHT Accuracy [%]	Proposed method Accuracy[%]
High	80	93.33
Average	58.62	86.2
Low	69.56	86.95

TABLE II: CHT and proposed method accuracy, tested on 200 images.

Fig. 6 (a) shows three captures taken under different ambient lighting conditions, (b) the performed edge detection and Hough candidates, (c) the center of circumferences obtained by the original CHT and (d) the center of circumferences obtained by the proposed method. In the very first two cases, the original CHT, performing the search by global maxima in the space accumulator, yields several points and is not able to discern between each of these centers. On the other hand, the proposed method, based on the roundness factor, selects the point whose neighborhood is more like a circumference and at the same time have the highest accumulator value.

Table II shows the accuracy of the original CHT versus the proposed method under different ambient lighting conditions. The threshold values were adjusted empirically in order to maximize the original CHT accuracy. Targets were manually created, and it was considered as right detection, distances less than three pixels. Although this value is arbitrary, it takes into account the error of manually creating each target. Based on these results, the proposed method improves the original CHT response between 16% to 46% depending on lighting conditions.

Table III and Fig. 7 show the performance of the algorithm for each of the three possible movement commands of the wheelchair: left, right and forward.

The processing time of each video frame during the tests was about of 100 ms.

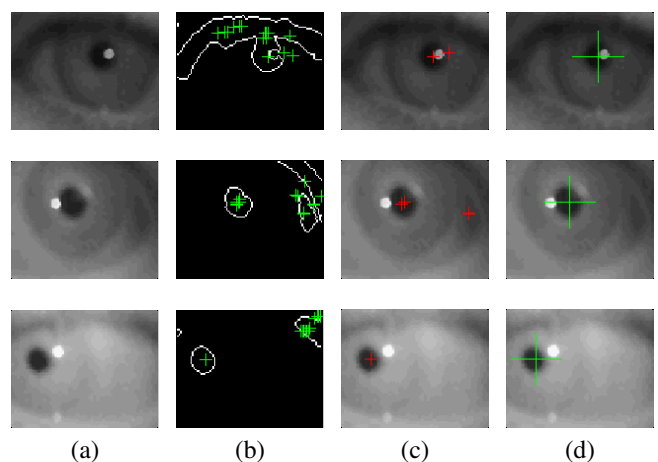


Fig. 6: Original CHT and proposed method performance under different ambient lighting conditions: (a) Original frame (b) Edge detection and CHT candidates (c) Pupil center by global maximum (d) Pupil center by proposed method

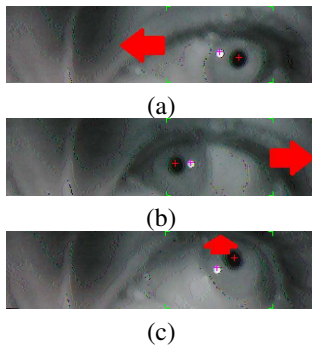


Fig. 7: (a) Subject eye pointing left (b) Subject eye pointing right (c) Subject eye pointing up

	COMMAND			
	LEFT[%]	RIGHT[%]	UP[%]	MEAN[%]
SUBJECT 1	92	89	80	87
SUBJECT 2	85	81	78	81
SUBJECT 3	90	89	83	87

TABLE III: Command identification accuracy, performed on 90 commands.

V. CONCLUSIONS

The proposed method allows real time gaze tracking using a very low cost computing platform, with satisfactory results under different ambient lighting conditions. The processing cost of each video frame allows to operate at a refresh rate of ten frames per second. Currently we are working on the implementation of the system on a commercial electric chair. As future work, in order to improve the system accuracy, we will research alternatives ways of selecting candidates for centers of circumferences of the CHT and the threshold value will be automated according to ambient illumination conditions. In addition, an algorithm to identify blinkers will be developed to reduce false detections and we will expand our eye database.

REFERENCES

[1] J. D. Simeral, S. P. Kim, M.J. Black, J.P. Donohue and L.R. Hochberg, "Neural control of cursor trajectory and click by a human with tetraplegia 1000 days after implant of an intracortical microelectrode array", *J. Neural Eng.*, vol. 8, no. 2, April 2011.

[2] Tsui, C.S.L., Pei Jia; Gan, J.Q., Huosheng Hu; Kui Yuan, "EMG-based hands-free wheelchair control with EOG attention shift detection, Robotics and Biomimetics", 2007. ROBIO 2007. IEEE International Conference on , vol., no., pp.1266,1271, 15-18 Dec. 2007. doi: 10.1109/ROBIO.2007.4522346

[3] D.A. Craig; H.T. Nguyen, "Adaptive EEG Thought Pattern Classifier for Advanced Wheelchair Control", Proceedings of the 29th Annual International, Conference of the IEEE EMBS, Cité Internationale, Lyon, France, August 2007.

[4] R. Berjon, M. Mateos, A. L. Barriuso, I. Muriel and G. Villarrubia, "Alternative human-machine interface system for powered wheelchairs", IEEE 1st International Conference on Digital Object Identifier, Serious Games and Applications for Health (SeGAH), 2011.

[5] Umeshkumar Jaiswar; Sanjna S. Repal, "Real Time Breath Processing Based Embedded Vehicle Control Using WSN", International Journal of Science and Research (IJSR), 2013, ISSN (Online): 2319-7064

[6] Gautham Krishnamurthy ; Maysam Ghovanloo, "Tongue Drive: A Tongue Operated Magnetic Sensor Based Wireless Assistive Technology for People with Severe Disabilities". Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium

[7] Lund, M.E.; Christensen, H.V.; Caltenco, H.A.; Lontis, E.R.; Bentsen, B.; Struijk, J.J., "Inductive tongue control of powered wheelchairs", Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE , vol., no., pp.3361,3364, Aug. 31 2010-Sept. 4 2010. doi: 10.1109/IEMBS.2010.5627923

[8] R. A. Cooper, M. L. Boninger, A. Kwarciak, and B. Ammer, "Development of power wheelchair chin-operated force-sensing joystick", Engineering in Medicine and Biology, 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society EMBS/BMES Conference, 2002. Proceedings of the Second Joint (Volume: 3)/23-26 Oct. 2002.

[9] Päivi Majaranta , Andreas Bulling, "Advances in Physiological Computing", Eye Tracking and Eye-Based Human-Computer Interaction, 2014, ISBN 978-1-4471-6392-3, pp 39-65

[10] Amer Al-Rahayfeh; Miad Faezipour, "Eye Tracking and Head Movement Detection: A State-of-Art Survey" IEEE Journal of Translational Engineering in Health and Medicine (Volume: 1), 2013, ISSN: 2168-2372

[11] Cristian E. Ordoñez; Juan I. Pastore; Virginia Ballarin; Eduardo L. Blotta, "Real-Time Iris-Tracking Embedded System" SASE 2015, FI-UBA, Bs. As., Argentina, ISBN 978-987-45523-3-4

[12] Pastore J.; Moler E., "Identificación de Senos Frontales mediante Factores de Forma y descriptores de Fourier", IEEE LATIN AMERICA TRANSACTIONS, VOL. 2, NO. 3, SEPTEMBER 2004 ISSN: 1548-0992, pp 157 - 161

[13] P.M. Merlin, D.J. Farber, "A parallel mechanism for detecting curves in pictures", IEEE Trans. Comput., C-24, pp. 96-98, 1975.

[14] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes", Pattern Recognition, vol 13(2), pp. 111-122, 1981.

[15] P.V.C. Hough, "Method and means for recognizing complex patterns", U. S. Patent 3069654, 1962.

[16] R.O. Duda, P.E. Hart, "Use of the Hough transform to detect lines and curves in pictures", Comm. Assoc. Computing, vol. 15, pp. 1-15, 1972.

[17] P.M. Merlin, D.J. Farber, "A parallel mechanism for detecting curves in pictures", IEEE Trans. Comput., C-24, pp. 96-98, 1975.

[18] J.S. Bruno, "Reconocimiento de burbujas en formularios inteligentes aplicando la Transformada de Hough", FRBA-UTN, Buenos Aires, Argentina, 2005.

[19] Rafael C. Gonzalez y Richard E. Woods. "Digital Image Processing". Prentice Hall 2nd Edition 2002.

[20] Daugman, John, "How iris recognition works", IEEE 2004.

[21] Mulvey, F., Villanueva, A., Sliney, D., Lange, R., Cotmore, S., Donegan, "Exploration of safety issues in Eyetracking", Communication by Gaze Interaction (COGAIN) IST-2003-511598: Deliverable 5.4. Available at <http://www.cogain.org/results/reports/COGAIN-D5.4.pdf>

[22] Elias Daniel Guestrin; Moshe Eizenman; "General Theory of Remote Gaze Estimation Using the Pupil Center and Corneal Reflections", IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, VOL. 53, NO. 6, JUNE 2006

[23] Spector, R.H.: "Visual fields" in Walker, H.K., Hall, W.D., and Hurst, J.W. (Eds): "Clinical methods: the history, physical, and laboratory examinations", Butterworths, Boston, 1990, 3rd edn.

[24] STM32F4DISCOVERY Expansion Boards Datasheets, www.element14.com/stm32f4-expansion.

[25] Miad Faezipour y Amer Al-Rahayfeh, "Eye Tracking and Head Movement Detection: A State-of-Art Survey". University of Bridgeport, Bridgeport, CT 06604, USA, 6 November 2013.

[26] UBIRIS database, <http://iris.di.ubi.pt/>

A virtualized version of MIL-STD-1553

Pablo Solivellas
 Avionics Engineer
 CITEA
 Argentine Air Force
 Argentina

Hernan Ponso
 Avionics Engineer
 CITEA
 Argentine Air Force
 Argentina

Andres Grop
 Avionics Engineer
 ESTEC
 European Space Agency
 The Netherlands

Manuel Amor
 Avionics Engineer
 Real-Time Systems Group
 UNRC
 Argentina

Diego Fusari
 Hardware Engineer
 CITEA
 Argentine Air Force
 Argentina

Abstract—The MIL-STD-1553B standard specifies a bidirectional data bus that has been used in defense and aerospace industry in a vast number of systems. However, its use for simulation and development environments might lead to unnecessary restriction on the design and usability due to the high cost incurred for hardware and harness procurement.

Since Ethernet networks are nowadays extensible present in simulation and development environments, integrating these existing communication architectures as part of the test-bench should reduce complexity while allowing flexibility and mitigating procurement cost.

In this article, a virtualized 1553 bus interface is presented with the aim to demonstrate the integration of a 1553 network architecture which operation and functional capabilities are independent from the physical layer. This is achieved by replacing specific hardware functionalities into software simulated components.

Index Terms—MIL-STD-1553, V1553, Virtualization, Simulation

I. MIL-STD-1553

In 1973, the United States Department of Defense first issued MIL-STD-1553 specifications as an Air Force standard to define mechanical, electrical, and functional characteristics of a time division multiplexed command/response data bus. Even if designed for military aviation its has been extensively used by civil aviation and other applications such as railway systems. This section provides a summary of the functional operation of a 1553 data bus. For a detailed description, see [1].

The bus works asynchronously, using a command/response protocol. Transmission of information is controlled by a bus controller (BC), who is in charge of initiating all transmissions. In addition to the bus controller, the standard allows to connect up to thirty one remote terminals (RT) to the bus, which are receivers of the commands coming from the bus controller.

The RTs process these commands and then send responses back to the BC. Figure 1 shows a generic 1553 bus architecture, in which the subsystem is simply a device using the bus for data transfer. Bus monitors are terminals that receive all the traffic but do not respond to any messages.

Information is transmitted on the bus in the form of messages, which can be formed by three types of words: command, data and status. Each word consists of sixteen bits, wherein the most significant bit is transmitted first, plus a three bit synchronization word and one parity bit. Bit transmission

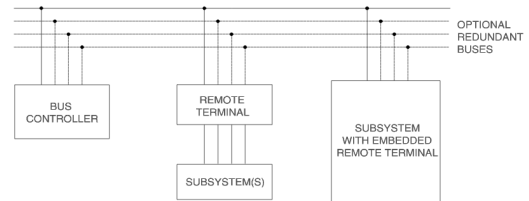


Fig. 1. Generic architecture of a 1553 bus.

time is one microsecond, achieving a overall data rate of one megabit per second. Figure 2 shows the three words format.

The standard defines several message formats. However, all follows the same general pattern. The BC start all messages by sending a command word to a RT. If the RT respond, it always responds first with a status word. Different message type are used in different data flow direction (i.e. BC to RT or RT to BC). The maximum number of data words that can be transmitted per message is thirty two [2]. In Figure 3 it is possible to observe some of the types of 1553 messages.

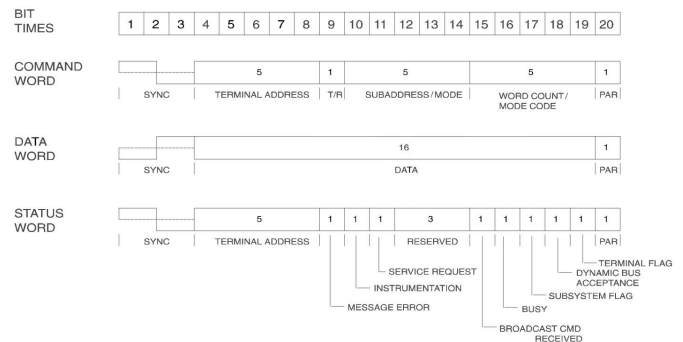


Fig. 2. MIL-STD-1553 word format [2]

The standard specifies clear time requirements for data transmission. The gap between messages shall be a minimum of four microseconds. A remote terminal must respond to the bus controller within four to twelve microseconds, and a bus controller must wait at least fourteen microseconds before considering that a response has not occurred [1].

The 1553 data bus is very reliable. Much of the standard is responsible for specifying requirements to achieve this reliability. The command/response protocol ensures that the

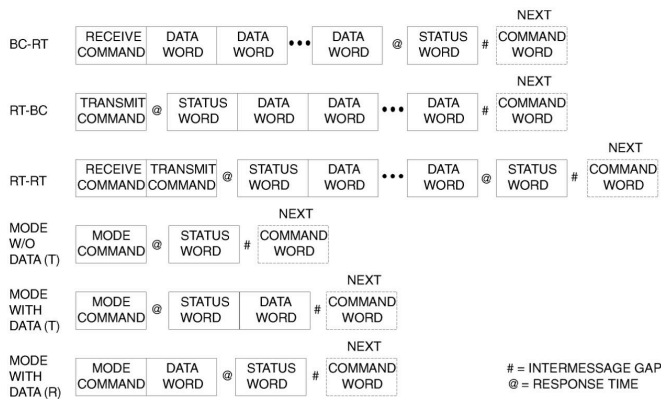


Fig. 3. MIL-STD-1553 messages [2]

bus controller knows whether a command was successfully processed or not. In addition, in a message, each word has a synchronization pattern and a parity bit to ensure a reliable data transmission.

To further increase the reliability of the data bus, the standard allows the use of redundant buses. These redundant buses provide more than one route for data transmission between subsystems and are widely used in practice.

II. MIL-STD-1553 APPLICATION WITHIN DEVELOPMENT AND SIMULATION ENVIRONMENTS

Software virtualization is the creation of a virtual version of some technological resource, such as a hardware platform, operating system, storage device or another network resources. Applied to hardware, virtualization is a technique that has been used since the 1960s [3], but in recent years there has been a growing trend towards the adoption of virtualized environments applied to test benches, achieving a significant cost reduction and an equivalent operation compared to the actual device. Particularly within the aerospace application field, an avionics system test bench is composed by a large number of processing units that share information using a MIL-STD 1553B data bus.

Nowadays, MIL-STD-1553B data bus is the most used bus within its type. It is used in all systems where data and system integrity are critical. It is widely used in avionics for aircraft, ships, submarines and ground vehicles such as tanks. It is also widely used in aerospace applied to numerous satellites and the international space station, although lately it is increasingly common to find it in commercial applications such as reactors, subway cars and oil stations.

The success of the standard is unquestionable. However, its use outside an operating environment, such as a research and development environment, may be unnecessarily restrictive [4]. Basically, the application of the standard involves having a network of 1553 cables and the necessary hardware to connect the different systems to the network.

A 1553 cable network consists of three fundamental components: cable stubs, stub couplers and bus terminals. In general,

1553 buses are located inside a vehicle. According to the standard, the maximum length of a cable stub is 6 meters. Using couplers it is possible to extend the network length to approximately 100 meters. However at this point, a lot of transmission lines experience is required to guarantee a working bus.

The other component of a 1553 data bus are cards that allow to connect a system or device to the cable network. The cards implement the functional aspects of the standard, for example, allow a system to send or receive data using 1553 protocol over the network cable, either as a BC or RT. MIL-STD-1553 cards are very expensive. Their prices vary between \$1000 and \$15000 dollars [5] [6] [7], depending upon the manufacturer, system interfaces and the capabilities of the card.

The development of a critical system requires taking into account rigorous testing phases. Once the system has been implemented and correspondingly tested, periodic maintenance is also required to check the performance and eliminate any operational failures.

In the system development stage the simulation makes it possible to accelerate software development and implicitly reduce project costs. When it comes to embedded systems development, tests can be carried out on both software and hardware.

In these environments, it would be interesting to use an existing communications infrastructure, such as an Ethernet network, rather than building a new network infrastructure. In this way, not only the existing infrastructure could be used, but also commercial hardware (i.e. Ethernet boards) could be used to interconnect the subsystems or devices to the network.

III. VIRTUAL -1553

Virtual-1553 emerges as a solution to some of the limitations associated with the use of a MIL-STD-1553 bus in a research and development environment. In essence, the goal of Virtual-1553, or V1553, is to replace the expensive hardware required to deploy a 1553 communications network with commercial hardware and existing communications networks. In this way, it is possible to achieve hardware independence, flexibility and a considerable cost reduction obtaining equivalent functional behavior.

Currently, Ethernet networks are widely extended and an Ethernet network interface is a standard component in any personal computer. V1553 creates a virtual MIL-STD-1553 data bus such that the 1553 cable network can be replaced by a commercial Ethernet network.

The key to MIL-STD-1553 protocol virtualization was to emulate hardware functionality in a software implementation. Basically, a "software-based 1553 board" was created that mimics the behavior of a real 1553 board. Most of the 1553 PC boards come with libraries that allow them to be controlled. In essence, V1553 turns that library into a "virtual 1553 board".

From a systems validation perspective, what matters most is what messages get transmitted over the data bus and when those messages are transmitted. It is possible to achieve a good

development, testing and simulation as long as the data reaches the destination correctly.

In order to make V1553 independent of the way in which messages are transmitted over the data bus, it is necessary to study in detail the types of words that constitute a MIL-STD-1553 message. As described in section I, the standard specifies three types of words: CW (Command Word), SW (Status Word) and DW (Data Word). In Figure 2 it can be seen that each word is 20 bits length. However, 4 of these 20 bits contain information used for transmission on the physical medium; 3 bits are used for synchronization and 1 bit for parity. These fields are added by the hardware before the transmission and then removed at reception. In V1553, these 4 bits are not included since they do not provide necessary information for the communication protocol, therefore, each word is 16 bits length instead of 20.

The big difference between MIL-STD-1553 and V1553 is that it does not adhere to the timing requirements of the data bus. As mentioned in section I, the standard is very strict with the timing for transmissions on the bus, for example, the minimum time between messages, the response time of a remote terminal, etc. V1553 does not guarantee compliance with any of these.

Basically there are two reasons why V1553 does not meet the time requirements required by the standard. First of all, V1553 is an implementation entirely by software. This means that it will run under the control of some operating system. It is a software product that is thought to be portable and does not restrict its use to real-time operating systems. It is for this reason that it is impossible to guarantee when and how often the software using V1553 will be allowed to run. Second, one of the main reasons for the creation of V1553 was the possibility of reusing existing communications networks, exclusively Ethernet networks without affecting functional validation issues. Transmission times depend on the exact nature of the communications network we are using. Due to the non-deterministic characteristics of Ethernet, it is not possible to guarantee or meet the time requirements specified by MIL-STD-1553.

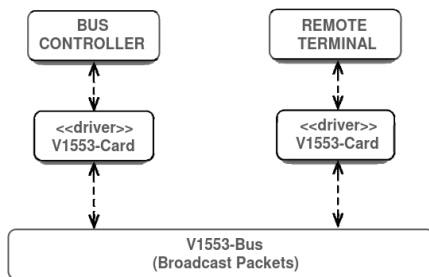


Fig. 4. Generic architecture of a V1553 bus.

Because V1553 must emulate hardware functionality using an existing network infrastructure, the design was divided into two main components; A library (or driver), called V1553-

Card, and the virtual data bus called V1553-Bus. A generic architecture of a V1553 bus can be seen in Figure 4.

A. V1553-Bus

The virtual data bus or V1553-Bus is responsible for porting the bus topology used in real communications to the software. The prototype was created using UDP/IP sockets as the communication mechanism and the broadcast transmission methodology available in Ethernet for the simulation of the bus topology. In Ethernet, broadcast communications are limited to a broadcast domain, in other words, to a local area network.

In this way, all transmissions will be received by all applications connected to the V1553-Bus. Figure 5 shows a V1553 architecture on an Ethernet network using UDP/IP.

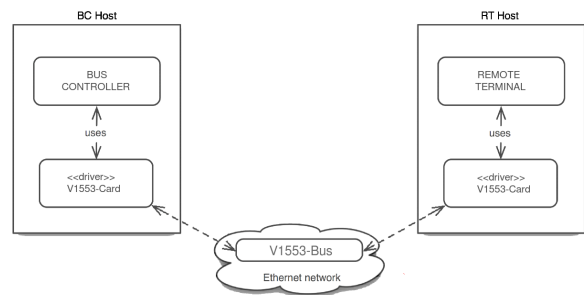


Fig. 5. V1553 architecture over an Ethernet network.

B. V1553-Card

The V1553-Card is a software library that implements the functionality of a real 1553 board together with the functional aspects of a MIL-STD-1553 data bus. The library (hereinafter the driver) is capable of acting either as a Bus Controller, a Remote Terminal or a Bus Monitor. From an application development perspective, using the V1553 driver or a real MIL-STD-1553 board must be completely transparent.

A design requirement is that, from the developer's perspective, the operation of the V1553 driver must be as similar as possible to operating with real MIL-STD-1553 hardware. UDP/IP sockets were used for the communications mechanism for their generality and simplicity.

IV. VIRTUAL-1553 BENEFITS

V1553 was designed with the goal to replace the real hardware interface needs. The decision on whether to use V1553 rely on whether the system is able to support at least 2 interfaces: a real hardware-based interface to be used during the full functional, qualification and acceptance tests, and operational phase of the project, and a virtual software-based interface to be used during the development phase. By using a virtualized interface the project can fundamentally benefit from hardware independence, design flexibility and overall cost reduction.

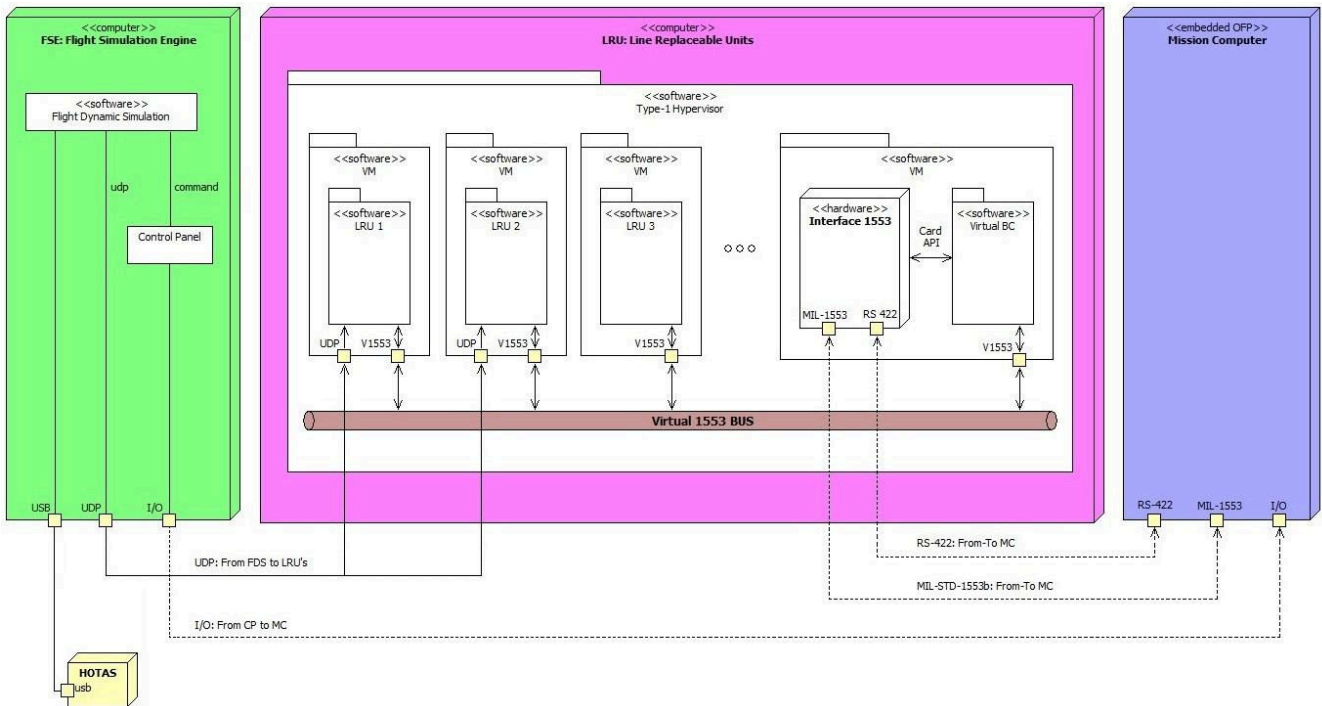


Fig. 6. SIL Communication Interfaces

A. Hardware Independence

The use of a virtualized interface requires the system software to have the flexibility to switch between real and virtual interfaces, therefore it must be designed to be independent of the hardware platform, specifically the communication devices. In addition, this allows the system to integrate hardware from multiple vendors without any major impact on the system architecture nor software implementation. Such design is shown in Figure 6.

B. Flexibility

In simulation environments, the use of real hardware-based components may be restrictive. Nonetheless, when using virtualized components (such as V1553 interfaces) the simulation system can be designed and implemented without complex harnessing and interfaces. The environment can be set-up anywhere where an Ethernet network is available.

This flexibility arises from the fact that most of the 1553 interface has been completely simulated in a software component. Therefore, all constraints due to physical availability of the different components are easily overcome. All interfaces (PCI, VME, etc) no longer limit the numbers of 1553 buses that a system can be connected to, since a new virtual interface can easily be added on.

When asked "Which is the real need to create connection to many virtual buses?", the answer is better understood when combining the virtual 1553 technology with Virtual Machines. Each VM can be assigned its own virtual interface, therefore, it is possible to completely replace complex real systems

and hardware-based interface, by using Virtual Machines and virtual buses (i.e. V1553). Combined, these two technologies have a lot to offer.

C. Cost reduction

The most important benefits of a virtualized interface is the significant reduction in implementation costs. The use of V1553 allows to reduce the cost associated to the procurement and deployment of a hardware based simulation environment. Moreover, since V1553 makes use of existing Ethernet networks, the needs for complex harnessing and interfaces is reduced. This way, the development of a system requiring a MIL-STD-1553B bus can be started in an early stage long before hardware procurement.

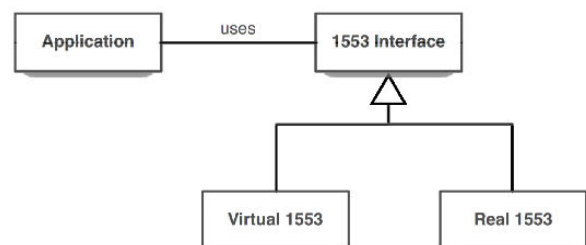


Fig. 7. Modular Software Design

V. VIRTUAL-1553 APPLICATION

Generally, developers and system integrators choose real-time operating systems (RTOS) such as Wind River VxWorks [9] and Green Hills Integrity [10] for critical flight software. However, in research and development environments, it is common to use operating systems such as Linux, Unix or Windows.

Taking into account that one of the main objectives of V1553 is cost reduction, both development and testing were carried out on a Linux operating system. In addition, the coding was performed using the ANSI-C standard allowing the driver to be portable to any operating system that supports the standard. On the other hand, it is very important to keep in mind that the use of a real-time operating system is a necessary but not sufficient condition for critical flight systems. As the development of V1553 is not certified under DO-178C (for any DAL¹ level) neither certifiable, that is, it was not developed under rules that serve as guide to certify, it is assumed that the software product will not be used in any critical flight system.

V1553 was required, developed and implemented within the Defense Research and Development Program (PIDDEF), funded by the Argentine Ministry of Defense, related to an avionics system. This project involves the development of a mission-critical operational flight software (OFP). It requires the development of a test bench that simulate the OFP to validate its operational behavior. It was subdivided into two systems: The SIL System (System Integration Laboratory) and the OFP-MC System. The SIL System is related to the Test Bench required for test and validate the OFP-MC System, while the latter involves the operational flight software running on the Mission Computer.

It is important to note that V1553 implementation was carried out only within SIL System, since OFP-MC System is not necessary for validation and testing purposes.

The SIL is composed of three major components [8], as illustrated in Figure 6. The first component, called *FSE Computer* (Flight Simulator Engine) is a computer that hold the software which solves the aircraft dynamic equation and works together with FlightGear². The second component, called *LRU Computer* is responsible for simulating each of the remote terminals that will constitute the avionics system. Finally, the third component is the new Mission Computer, which must be tested on the Test Bench.

Each simulated LRU within the SIL must be able to receive data from the V1553 bus, operate this data, convert the result to the correct format according to 1553 protocol and finally send this data on the V1553 bus. Because BC tasks are carried out by the OFP-MC System, a software component that fulfills a role of BC was implemented in order to replace the absence of the OFP-MC. This particular component use buffers where all command, data and status words are stored and then used

¹Development Assurance Level is determined from the safety assessment process by examining the effects of a failure condition in the system.

²FlightGear Flight Simulator, often shortened to FlightGear or FGFS, is a free open source multi-platform flight simulator developed by the FlightGear project since 1997.

when the MC requires it. In Figure 8 it is possible to observe a block diagram of the LRUs that make up the navigation system of the System Integration Laboratory.

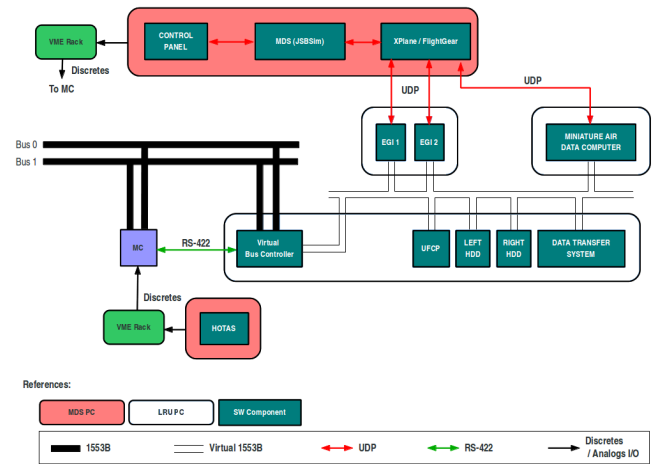


Fig. 8. Navigation system block diagram

In order to provide a greater functional security, a hypervisor was used in the LRU computer for the virtualization of each of the remote terminals. Unlike a conventional virtual machine, the hypervisor runs directly over the hardware instead of running on another operating system. This allows both the hypervisor and the virtual machines to run very efficiently. The hypervisor offers many benefits. As each remote terminal runs on a dedicated virtual machine, this increases the stability of the whole system because if a failure occurs in an RT, this failure will not propagate to the rest of the virtual machines. Figure 6 shows the communication interfaces between the different software and hardware components that make up the SIL.

VI. RESULTS

A test plan was proposed for the functional testing of V1553. The test plan consisted of monitoring the time and number of messages lost in communications between an RT and the BC with the virtual bus without any additional traffic load, with 50% and 100% traffic load respectively.

For each case, the analysis was carried out over about 1500 messages. On average, the round-trip time from the moment the RT receives a command from the BC, and subsequently this one get the RT response back, is around 120 microseconds. While it is expected that the time requirements specified by the standard can not be met, it is interesting to note that these times remain relatively constant within acceptable ranges. In addition, the percentage of lost messages recorded in the tests yielded values that can be considered negligible.

VII. CONCLUSIONS

To date, the role of simulation to support the design of complex avionic system is increasingly important. Computers are used in all stages of the project, from the design to

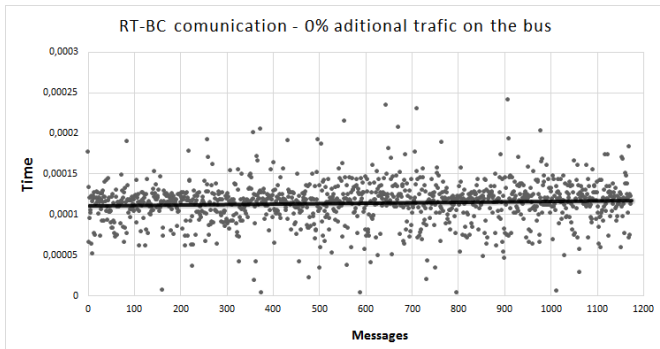


Fig. 9. End-to-end transmissions without additional traffic.

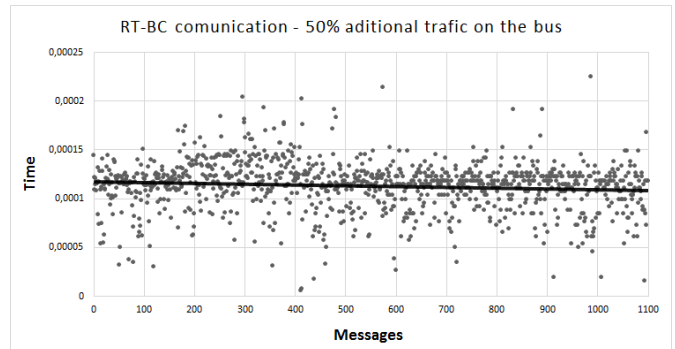


Fig. 11. End-to-end transmissions with 100% traffic load.

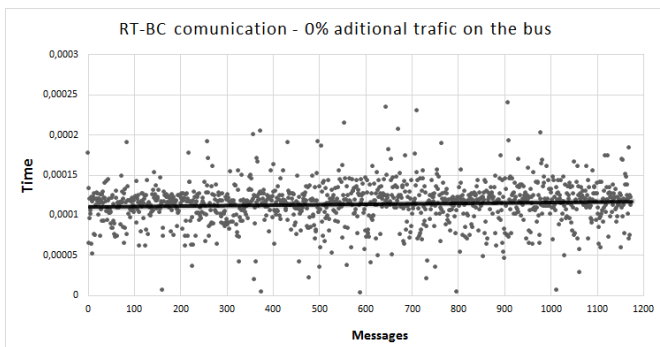


Fig. 10. End-to-end transmissions with 50% traffic load.

validation, testing and evaluation of system prototypes, providing knowledge of functional and operational performances of a system from early stages of development. By having a system integration laboratory, where the simulated subsystems of an aircraft resemble the actual ones, a system designer can integrate and evaluate complex systems such as an aircraft's critical mission computer.

V1553 is a technology that has a lot to offer in these simulated environments. Hardware independence, flexibility and cost reduction are great advantages considering that it offers a functional behavior equivalent to the real hardware. Although the response times are higher than those stipulated by the standard, these are kept constant within an acceptable range without significant messages loss. It is therefore possible to conclude that data delivery can be guaranteed.

It is also important to note that in the application field of this technology within the Argentine Air Force, it has made a strong contribution to data bus engineering because system engineers were able to carry out their work without relying on long times associated with the tedious processes of purchasing equipment.

VIII. FUTURE WORK

V1553 has proved to be a very useful technology in its application field. Working on their portability to deterministic Ethernet networks such as TTEthernet from TTTech [12] or ADFX from ARINC [11] would solve the temporal issues

analyzed and packet losses providing a high added value. In addition, the determinism offered by the above mentioned technologies will favor the results obtained by integrating the OFP-MC system to the tests.

ACKNOWLEDGMENT

The authors thank CITEA^s (Research & Development of Aeronautical Technologies Center - Argentine Air Force) members and GSTR^s (Real-Time System Group - National University of Río Cuarto) members for their feedback on earlier version of this work.

This work was partially funded by the PIDDEF 31/11 Project, dependent on the Research & Development Program for the Defense of the Ministry of Defense and by the PPI 18/B215 Project dependent on Science & Technical Secretary of the National University of Río Cuarto.

REFERENCES

- [1] United States Department of Defense, *MIL-STD-1553B - Aircraft Internal Time-Division Multiplexing Data Bus*, 1978, Washington, D.C.
- [2] Condor Engineering, *MIL-STD-1553B Tutorial*, June 5, 2000.
- [3] Everything VM, *History of Virtualization*, August 2011. [Online] Available: <http://www.everythingvm.com/content/history-virtualization>
- [4] N. Downing, *Virtual MIL-STD-1553*. 25TH Digital Avionics Systems Conference, October 2006.
- [5] Ballard Technologies, *MIL-STD-1553 Online Catalog*, April 2017. [Online] Available: <http://www.ballardtech.com/products.aspx/dir/protocol/MIL-STD-1553/>
- [6] Data Device Corporation, *MIL-STD-1553 Online Catalog*, April 2017. [Online] Available: <http://www.ddc-web.com/Products/MIL-STD-1553/Cards.aspx>
- [7] Excalibur Systems, *MIL-STD-1553 Online Catalog*, April 2017. [Online] Available: <http://www.mil-1553.com/mil-std-1553>
- [8] D. Primo and P. Solivellas and H. Ponso, and M. Amor, and G. Rodriguez, and A. Cararo, *Proposal for the design of a System Integration Laboratory for the new generation of avionics systems*, Congreso Argentino de Sistemas Embebidos, August 2013.
- [9] Wind River, *VxWorks*, Last visited april 2017. [Online] Available: <https://www.windriver.com/products/vxworks/>
- [10] Green Hills Software, *INTEGRITY-178*, Last visited april 2017. [Online] Available: http://www.ghs.com/products/safety_critical/integrity-do-178b.html
- [11] Condor Engineering, *AFDX Protocol Tutorial*, May 2005. [Online] Available: http://www.cems.uwe.ac.uk/a2-lenz/n-gunton/afdx_detailed.pdf
- [12] TTTech, *Time-Triggered Ethernet*, May 2017. [Online] Available: <https://www.ttttech.com/technologies/deterministic-ethernet/time-triggered-ethernet/>

Highly configurable Ethernet controller for HW/SW co-debugging

Ricardo Cayssials^{1,2}, Damián Banfi², Lorenzo De Pasquale¹, Diego Martinez³, Edgardo Ferro²

¹National University of Technology – Bahia Blanca Faculty

Department of Electronics Engineering

²Universidad Nacional del Sur

³Argentine Army – Department of Defense

Bahía Blanca - Argentina

{rcayssials}@frbb.utn.edu.ar

Abstract—FPGA devices allows designer to implement complex digital architectures that involve hardware and software components. Because of the different features of hardware and software design, diverse mechanisms and tools have been proposed for debugging and verification of architectures implemented on FPGA devices. Bus level transactions and data processing algorithms are usually difficult to manage together because differences between the inspection times involved. While hardware transactions are related to system clock periods, data processing algorithms are related to software execution times. In this paper, an Ethernet controller is proposed as a low invasive component to extract runtime hardware and software information for system debugging and verification. The information produced may be analyzed to locate hardware and software failures as well as to improve the system performance tuning its configuration parameters. A case study is proposed as a real application.

Keywords - FPGA, co-debug, Ethernet protocol, MII interface

I. INTRODUCTION

Modern FPGA devices allow engineers to implement very complex digital applications, challenging the classical design and debug techniques. Locating a hardware failure or a software bug in complex designs requires efficient techniques and methodologies able to deal with this complexity.

Even though many debug, simulating, emulation and verification tools have been proposed, neither of them can be proclaimed as the best one because all of them offers features more convenient for some circumstances than others. Consequently, the engineer has to manage some of these tools and chose the most adequate one according to either the stage of the project or the characteristics of the failure.

Altera offers several mechanisms and methodologies to debug digital designs implemented within its FPGA devices. TimeQuest timing analyzer, PowerPlay power analyzer, System Console, Transceiver Toolkit, SignalTap II Logic Analyzer, External Memory Interface Toolkit, Bus Analyzer Toolkit, and Simultaneous Switching Noise (SSN) analyzer are

This work was supported by the Program *Convenios Internacionales* of the Universidad Autónoma de Madrid, the PID Research Project UTI4461TC of the UTN-FRBB, the PGI Research Project 24/ZK25 of the UNS and the PIDDEF 07/12.

some of these tools. Most of these tools allow runtime debugging using the JTAG communication port.

Advanced digital design involves digital logic, hardware processing architectures, transceivers, DSP blocks, memory controllers and software components, among others. Debugging and verification tools intended for some developments may be inadequate for others. For instance, a System-on Programmable-Chip (SoPC) contains hardware as well as software components that have to work together in correct synchronization. However, while software is debugged with GDB (generally in an Eclipse environment) and External Memory Interface Toolkit, hardware is verified through simulation, system console and Bus Analyzer Toolkit. Find the adequate tool to track a bug may be complicated, even more when it is produced by a non-repetitive failure.

FPGA manufacturers offer development kits with support for designing and debugging. Most of these kits include Ethernet interfaces for network communications. In this paper, we propose an Ethernet controller aimed at hardware and software debugging for SoPC. The Ethernet controller is easily included in the FPGA design and allows to communicate runtime information to an external debugging and verification application running on a PC. The information that can be extracted from the FPGA during runtime covers data values, code execution, state of hardware signals and monitoring Avalon bus transactions. The Ethernet controller is designed based on the MII interface in order to be utilized in most of the FPGA development kits.

The paper is organized as follows: Section 2 describes the main hardware and software techniques to debug SoPC applications. Section 3 outlines the Ethernet interface and the communication protocol implemented for the controller proposed. The Avalon Bus used for SoPC for Altera devices is presented in Section 4. Section 5 details the Ethernet controller implemented for HW/SW co-debugging. The commands and responses defined for the Ethernet controller are explained in Section 6. Section 7 enumerates the main HW/SW co-debugging features to manage the system clock of the SoPC under debugging. A case study is explained in Section 8. The debugging performance of the Ethernet controller is evaluated in Section 9. Conclusions are drawn in Section 10.

II. HW/SW CO-DEBUG

SoPC architectures involve hardware as well as software components that has to work together. While software

components deals with data values storage in memory and code execution, hardware components handle signals status, state machines and time synchronization.

Several methodologies and tools have been proposed for hardware-software concurrent design (HW/SW co-design). Altera offers designers a suite of tools for designing SoPC. QSys is an Altera application tool for designing hardware architectures able to be synthesized into Altera FPGAs. Natively, Qsys allows engineers to design mono and multi processors architectures based on the Nios II soft processor ([1]). The Qsys integrates seamlessly with the SBT (Software Build Tools), based on Eclipse, to program and debug software applications to be executed by the hardware architecture designed in Qsys.

A SoPC implemented in a FPGA device contains several components, some of them may be *custom components*. SoPC components interconnect among them through the system bus. Altera defines for its SoPCs the Avalon Bus which contains different types of interfaces. The Component Editor ([2]) helps to create a Qsys component compliant with the Avalon Bus from a hardware description language (HDL) specification.

Hardware fault localization may be performed using the Bus Functional Models (BFMs) provided by Altera ([3]). The Verification IP Suite includes BFMs for different Avalon Bus interfaces to simulate the behavior and facilitate the verification of components.

Another alternative for hardware fault localization, when Avalon Bus is involved, is Altera System Console. Altera System Console is a system-level debugging and monitoring tool that helps designers to debug their design at full speed. Altera System Console allows designers to read and write Avalon Bus transactions to analyze and locate faults. When no Altera Nios II soft processor is involved, Altera recommends its SignalTap II embedded logic analyzer (available in Quartus II design software) useful for looking at a predetermined set of signals and examining their behavior in time ([4]).

On the other hand, Altera proposes GDB as its predefined tool for software fault localization tool. GDB is a software debugging tool protected by the GNU General Public License and it is included in the SBT suite ([5]). The software debugging tools communicate through the JTAG Debug Module included into the Altera Nios II soft-processors. The JTAG Debug Module provides software debugging facilities such as:

- Downloading programs to memory,
- Starting and stopping program execution,
- Setting breakpoints and watch points,
- Analyzing processor registers and memory content,
- Collecting real-time execution trace data.

When an engineer designs a SoPC, the design flow allows focusing on software functionality as well as the hardware architecture with tools efficiently integrated. However, when a fault needs to be located, then several tools are available and engineer has to take into account the details of both hardware and software components of the particular application to chose the adequate tool and mechanism to utilize.

While software debugging tools works at functions and code section execution level, hardware fault localization tools work at system level monitoring clock periods and bus transactions intervals. Consequently, it could be orders of

magnitude between the intervals considered in a software debugging strategy and a hardware one, making extremely difficult to locate a failure.

The Ethernet controller proposed facilitates the monitoring hardware signals during runtime. The module is able to handle the clock of the system to adequate the speed of the hardware to the bandwidth of the Ethernet communication link. The controller also provides the monitoring of Avalon Bus interfaces in order to analyze the behavior of the software application during runtime and getting a code profile.

III. ETHERNET COMMUNICATION PROTOCOL

Ethernet is a communication protocol for computer networks very successful since its beginning in 1970's and it was used as a basis for the IEEE 802.3 specification ([6]). In this paper, as it is common usage nowadays, Ethernet is used to refer to the IEEE 802.3 specification.

A reliable and efficient communications network may involve different aspects and protocols. The International Organization for Standardization (ISO), proposed the OSI (Open Systems Interconnection) reference model. The OSI reference model determines different layers in which communications can take place and the involved protocols. The IEEE 802.3 specification may be referenced to the layer 1 (Physical) and layer 2 (Data Link) of OSI reference model ([7]).

Several manufacturers offer integrated circuits that meet different physical layer versions of the IEEE 802.3 standard. These integrated circuits synchronize with the physical medium and receive and transmit frames that satisfy the Ethernet protocol. Even though physical medium may differ (coaxial, twister pair or fiber), interfaces with the upper layer have been proposed to reuse components among these different media.

MII (Medium Independent Interface) ([7]) is a standard interface to connect a layer 2 Ethernet 10/100 MBs/s MAC (Media Access Control) to a layer 1 PHY interface. Most of FPGA development kits with Ethernet interface support MII specification. Higher speeds Ethernet integrated circuits, for instance Gigabit Ethernet ones, proposed GMII or RGMII interfaces but they may also support MII specification for 10/100 Mb/s communications.

A. Ethernet Frame

MII interface allows to access physical medium from the MAC layer. Ethernet standard defines the information that a frame has to meet to be considered a valid frame. The frame contains the data to be communicated and it should have the following format:

TABLE 1: BASIC ETHERNET FRAME (FROM [8])

Length	Description
	Inter-frame
7 octets	Preamble
1 octet	Start Frame Delimiter (SFD)
6 octets	Destination Address (DA)
6 octets	Source Address (SA)
2 octet	Length(≤ 1500)/ Type (≥ 1536)
46 to 1500 octets	Data
	Pad (if data < 46 octets)
4 octets	Frame check Sequence (FCS)

The Length/Type field is a double purpose field: (1) the number of octet when lower than 1500, or (2) the type of payload when greater than 1536 (EtherType). For instance IPv4 = 0800H, ARP = 0806H.

The Ethernet controller proposed utilizes the EtherType field. It reduces the overhead of the protocol and consequently it improves the efficiency of the data communication and simplifies the hardware architecture to process the Ethernet frames. The default EtherType is 0801H but it can be modified when the Ethernet controller is added to the Qsys SoPC.

The Ethernet controller utilizes the data fields to transmit both the commands and the information between the Ethernet controller and the debug computer.

IV. AVALON BUS

Qsys is provided by Altera to design SoPC. Qsys allows designer to connect components in order to design the architecture of the SoPC. The Avalon Bus specification ([9]) is the standard interface proposed by Altera to connect components in an Altera SoPC. The Avalon Bus interface family defines appropriate interfaces for streaming high-speed data, reading and writing registers and memory, and controlling off-chip devices. These standard interfaces are designed into the components available in Qsys.

The most commonly used Avalon Bus interface in a SoPC is the Memory Mapped interface (Avalon-MM), natively implemented in Nios II soft-processor. The Avalon-MM is an address-based read/write interface with a master-slave connection. The Ethernet controller includes an Avalon-MM interface to monitor the transactions performed by the system components. System components that typically include Avalon-MM interfaces are microprocessors, memories controllers, UARTs, DMAs and timers.

Table 2 shows the Avalon-MM signals considered in by the Ethernet controller in a transaction handshaking.

TABLE 2: AVALON-MM SIGNALS (FROM [9])

Signal	Direction	Width	Description
address	Master-Slave	1-64	The byte address of the memory involved in the transaction
read	Master-Slave	1	Asserted to indicate a read transfer. If present, readdata is required
readdata	Slave-Master	8, 16, 32, 64, 128, 256, 512, 1014	The data driven from the slave to the master in response to a read transfer
write	Master-Slave	1	Asserted to indicate a write transfer. If present, writedata is required.
writedata	Master-Slave	8, 16, 32, 64, 128, 256, 512, 1024	Data for write transfers. The width must be the same as the width of readdata.
waitrequest	Slave-Master	1	Asserted by the slave when it is unable to respond to a read or write request. Forces the master to wait until the interconnect is ready to proceed with the transfer.

The Avalon-MM interface implemented in the Ethernet controller is similar to the one implemented by the Avalon-MM Pipeline Bridge in the Qsys component library in order to preserve compatibility among previous and future IP cores.

Changes were made to monitor the bridge signal and to hold the system transaction when Ethernet controller transmission FIFO is almost full.

V. ETHERNET CONTROLLER

Several Ethernet controllers have been proposed for FPGA devices ([10, 11, 12, 13]). All of these controllers are designed as peripheral to be utilized in a certain embedded system and consequently compatible with a system bus such as Avalon Bus, Wishbone, AHB, etc. Controllers are proposed as a configurable peripheral that can be adequately programmed using software functions executed by the system processor. Different functionalities can be achieved by modifying the values of the controller registers accessed through the memory map of the system. This kind of Ethernet controllers requires software routines to be executed by the system processor.

In [14], the Ethernet Debug Communication Link (EDCL) is proposed as an optional to the GRETH module for read and write access to an AHB bus (a reduced version is proposed in [15]). The EDCL does not offers monitoring nor clock and reset management of the system architecture.

The Ethernet controller proposed does not require any software routine executed by the system processor. The controller registers are configured through the Ethernet link from the debugging computer. In this way, the intrusion of the Ethernet controller on the SoPC is reduced because neither software routine nor system processor are required. Fig. 1 shows a layout of the Ethernet controller proposed.

The clock management unit is designed to manage the reset and clock of the Design Under Test hardware (DUT) according to the bandwidth available for data communication. When transmission FIFO (FIFO TX) is full, then the clock of the system may be configured to be hold until new space became available.

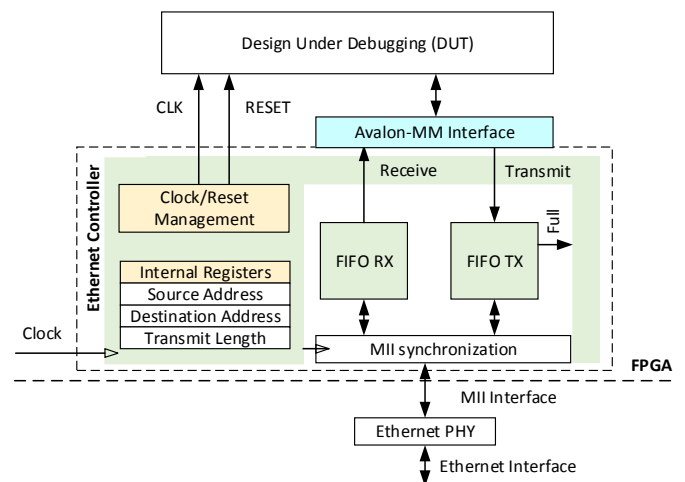


Figure 1. Ethernet controller architecture.

All the internal registers of the Ethernet controller are modified from the debugging computer through the Ethernet link. The communication link is established automatically taking advantage of modern Ethernet PHY interfaces and then all the Ethernet controller internal register are available for reading and writing from the debug computer. The

functionality of the Ethernet controller is configured through the values of these internal registers.

The architecture proposed changes the paradigm of previous Ethernet controllers. This unit was defined for co-debugging purposes, with easy to use, flexibility and efficiency features. The controller has one or more Avalon Bus interfaces for runtime debug and monitoring of the DUT. The Avalon Bus interfaces are integrated together with the Ethernet controller for a higher performance.

VI. DEBUG PROTOCOL PROPOSED

The communication between the debug computer and the Ethernet controller is designed to reduce the required FPGA resources and to improve the communication performance. A main feature of the Ethernet controller is the availability to be full configured from the debug computer. This feature requires full access to the internal control registers of the Ethernet controller from the debug computer and consequently the communication protocol has to be designed taking this feature into account. The Ethernet controller protocol has to support:

- read from and write to system memory: protocol should support memory ranges and data values to be read from and write to system memory, respectively. The Avalon Bus interface is required for easy adaptability when DUT is included in a SoPC.
- read from and write to internal registers of the Ethernet controller: the behavior of the Ethernet controller should be configured from the debug computer by modifying the internal registers and its status should be able to be read.
- continuous monitoring of system activity: the Ethernet controller could be configured to send all the Avalon Bus activity to the debug computer. Control registers of the Ethernet controller should be configured to enable and disable the monitoring activity.

The communication protocol proposed implements different packets (embedded into the data field of the Ethernet frame) to perform efficiently each one of the cases it has to support.

A. Debug computer to Ethernet controller frame

Table 3 shows the fields that the debug computer sends to the Ethernet controller building a *computer packet*. These fields are embedded into the data field of the Ethernet protocol of Table 1.

TABLE 3: DEBUG COMPUTER TO ETHERNET CONTROLLER PACKET FIELDS

Length	Description
3 octets	Logic Layer Control
1 octet	Sequence Number
2 octets	Payload Length
1 octet	Command
4 octets	Address / Register
2 octets	Data Length
Data Length(for write commands) or 0(for read/configure commands)	Data

Logic Layer Control: this field is included for Ethernet frame compatibility. It is reserved for future uses in the Ethernet controller definition.

Sequence Number: each computer frame is assigned with a consecutive sequence number. This number is used when retransmission are required by the Ethernet controller or an acknowledge is responded.

Payload Length: specifies the number of octets used by the computer frame.

Command: determines the command required by the debug computer to the Ethernet controller. Seven commands were specified (future commands could be added):

- Read register.
- Write register.
- Read system memory.
- Write system memory.
- Monitor Avalon bus interface
- Enable continuous mode monitoring
- Disable continuous mode monitoring

Address / Register: determines either the starting memory addressed or the first register involved in the debug computer command.

Data length: specifies the number of values to be read from or write to in a read and write command, respectively. When a register transfer is considered, the data length field includes the number of values to be transferred.

B. Response Ethernet controller to debug computer frame

The Ethernet controller always responds to a computer packet with a *controller packet*. The controller frame contains the same fields shown on Table 3. The Ethernet controller response preserves all the values of the computer packet but changes the data field according to the response.

VII. CO-DEBUG CONFIGURATION

The Ethernet controller offers system designers a highly efficient data communication link. This communication link is proposed to be used for transferring the information produced during a debugging/testing procedure of a HW/SW digital design. The nature of the information depends on the debug and test stages performed by the engineer during the fault localization procedure or performance measurement.

The Ethernet controller is configurable to adapt the data interface to the number of signals required to monitor. An Avalon Bridge component has been described to intercept the Avalon transactions. It interfaces with the Ethernet controller to monitor the changes of the address, data and control signals of every read and write transfer in a MM-interface of the Avalon Bus interface. A time stamp record is associated to every signal change. The Avalon Bridge component can be associated to each Avalon master or slave interface of the SoPC and then transmit the transaction information through the Ethernet controller.

Most of the cases, SoPC are designed to execute with high clock frequencies to get a better runtime performance. However, a higher clock frequency requires a higher

was aligned properly. Consequently, the data rate of the Radar video IP-core was doubled and the temporal constraints were met successfully.

The Cartesian-to-polar conversion is performed writing iteratively the polar memory in angle and radius sorted addresses with the corresponding data read from the Cartesian memory. The data conversion could be easily verified analyzing the data transferred from the read addresses to the write address. While the writing process is performed in consecutive memory addresses, the read transactions is performed in a random memory access mode. Burst and arbitration parameters could be adjusted to avoid unnecessary bus contentions. Moreover, the conversion data rate could be speeded down since it exceeded its temporal specification. By doing this, the Avalon Bus utilization was optimized for future system IP-core additions. All these analyzes could be more time consuming if there were accomplished with gdb for data consistency and SignalTap II or simulation for transaction performance.

IX. RESULTS

The debugging methodology proposed tries to be as less invasive during runtime as possible. However, changing the clock of the system may modify the behavior of the application with reference to asynchronous events. Even though these asynchronous events may be handled accordingly to reduce the side effects in the debug procedure, the behavior will be always affected. The bottleneck of this methodology is the limited bandwidth of the Ethernet link for the many signals that a digital design may contain. We measure the efficiency of the Ethernet link for different lengths of the frame. The Ethernet protocol defines fields, required to handle the communication (destination and source addresses, type, preamble, etc.). These fields demand bandwidth which is not available for data communication. Because these fields length are constants, the overhead that they produced is reduced when the frame length increases. The maximum bandwidth reached for different frame lengths and the overhead at 100 Mbps are shown on table 4.

TABLE 4: BANDWIDTH REACHED FOR DIFFERENT FRAME LENGTHS

Frame length (Bytes)	Overhead (%)	Bandwidth (Mb/s)
64	32.81	67.2
128	16.41	71.0
256	8.20	82.6
512	4.10	91.8
1024	2.05	91.9

It can be noted that data bandwidth improves when the frame length increases, but it could be not worth to define a greater length than 512 bytes in order to save FPGA memory resources.

TABLE 5: RESOURCE UTILIZATION

Family	Logic	Registers	Memory Bits
Cyclone IV	3,634 LEs	2,120	86,978
Arria II	2,092 ALUTs	2,122	86,978
Stratix III	2,100 ALUTs	2,168	86,978

On the other hand, the FPGA resource utilization is another requirement of the Ethernet controller. The resource utilization is shown on table 5 for Ethernet controller plus a Avalon Bus FIFO for a 32-bit data bus width and 26-bit address bus width.

The resource utilization becomes more noticeable and invasive when the Ethernet Controller includes more than one Avalon Bus interface to monitor. Each Avalon Bus interface includes a FIFO component for storage and multiplexing logic.

X. CONCLUSIONS

HW/SW co-design techniques allow engineers to implement complex systems on a single FPGA chip. The complexity of these systems requires diverse debugging and testing methods to localize failures and perform a functional verification and evaluation.

In this paper, we propose a low intrusive debug and verification mechanism that can be included as an alternative tool for co-debugging of hardware and software designs implemented on Altera FPGA devices. Altera, as well as others companies, have proposed different tools for hardware and software debugging. However, the highly exigent industry of embedded systems and SoPC demands the most diverse tools that could help to make the co-debug process efficiently.

A real SoPC application has been utilized as case study of the Ethernet controller proposed. The information obtained was firstly used to configure the diverse parameters of the Avalon Bus and the IP-cores to meet the design specifications and verify their functional behavior. After that, the debugging information was also useful to tune the access to the Avalon Bus of each IP-core of the SoPC in order to, meeting the design specification, allow adding future functionalities and IP-cores to the system.

REFERENCES

- [1] Altera Corporation, "Nios II Classic Processor Reference Guide", NII5V1, 2015.04.02, Altera, San Jose, April 2015.
- [2] Altera Corporation, "Creating Qsys Components", QN51022, 2013.11.4, Altera, San Jose, 2013.
- [3] Altera Corporation "Avalon Verification IP Suite – User Guide", UG-01073, 2015.06.04, Altera, San Jose, 2015.
- [4] Altera Corporation, "System-Level Debugging and Monitoring of FPGA Designs", WP-01170-1.0, November 2011, Altera, San Jose, 2015.
- [5] Altera Corporation, "Nios II Software Build Tools", NII52015-11.0.0, Altera, San Jose, May 2011.
- [6] IEEE, "802.3-2012 - IEEE Standard for Ethernet - Section 1", E-ISBN: 978-0-7381-7312-2, Dec. 28 2012.
- [7] IEEE, "802.12-1995 - 15. Medium Independent Interface (MII) Specifications", E-ISBN : 0-7381-0445-0, 1995.
- [8] IEEE, "IEEE Std 802.3-2008", 2008.
- [9] Altera Corporation, "Avalon Interface Specification", version 13.0, Altera, San Jose, 2014.
- [10] Altera Corporation, "Tripple-Speed Ethernet Megacore Function", UG01008, 2015.06.15, Altera, San Jose, 2015.
- [11] Michael Spanier, Alex Gorenstein, "Ethernet Communication Interface for FPGA", Final Project from Cornell ECE5760, Cornell University, New York, 2011.
- [12] Igor Mohor, "Ethernet IP Core", Open Cores, Open Project, Slovenia, October 2002.
- [13] Collin Durocher, Jeffrey Spiers, "Ethernet MII Based Transeiver for FPGA", Student Application Note, University of Alberta, Edmonton, Mach 2001.
- [14] www.cobham.com/gaisler, "GRLIB IP Core User's Manual", version 2017.2, May 2017.
- [15] Rodrigo A. Melo, Salvador Tropea, "IP core MAC Ethernet", Congreso de Microelectrónica Aplicada 2010, July 2010.
- [16] Robert Rodrigues, "RGMII Interface Timing Budgets", Texas Instruments, SNLA243, October 2015.

Design of a Smart Lock on the Galileo Board

Matias Presso, Diego Scafati, José Marone
Fac. de Ciencias Exactas, Univ. Nacional del Centro de la
Provincia de Buenos Aires, Argentina
marone@exa.unicen.edu.ar

Elías Todorovich
Univ. Nacional del Centro de la Pcia. de Bs. As. and
Universidad FASTA, Argentina
etodorov@exa.unicen.edu.ar

Abstract—In this work, a WiFi enabled door lock allows users to lock or unlock doors by codes that are managed from a computer or phone where new codes are issued or deleted including temporary codes to guests or office personnel. This work demonstrates an efficient smart lock system that can be configured remotely using Intel Galileo boards. The on-board server is connected through a mini-PCIe Wi-Fi card supported by Galileo. The Web server provides real-time information to remote administration regarding interesting events on the smart lock in the form of a responsive webpage. In addition, all the source code of the project is open and available for developers.

Keywords—smart lock; open source; Intel Galileo; IoT

I. INTRODUCTION

Internet has evolved from connecting computers to connecting IP-enabled things that can be employed for real time remote monitoring in the Internet of Things (IoT).

Between 2013 and 2014, Intel donated 50,000 Galileo boards to about 1,000 universities worldwide. This development board is compatible with Arduino to enable university students to innovate in the IoT. One of the impacts of this large-scale donation is a number of papers published with designs and applications related to IoT. For example, [1] demonstrates a method of real time remote monitoring of environmental parameters using Intel Galileo. In this method, an on-board server is connected to the Wi-Fi router that acts as a gateway for this network. The Web server provides the real time sensor information with dynamic updates in the form of a webpage to the remote client. Another application is a fall detection system for the elderly [2].

Connected devices and the related IoT technologies generate various smart embedded products that can be used in diverse applications, like home automation systems. A smart lock is one of the most important parts of a smart home. Authors in [3] present a smart lock system focusing on security issues. In a relatively older work [4], a ZigBee module is embedded in the digital door lock and the door lock acts as a central main controller of the overall home automation system.

The goal in this open source project is to develop an efficient smart-lock system using essentially a Galileo board.

A. Intel Galileo

Intel Galileo is the first in a line of Arduino-certified development boards based on Intel x86 architecture and is designed for the maker and education communities [5]. This project uses the first of the two Galileos, referred to as Gen 1, which was released in October 2013. Galileo runs a Linux operating system with the Arduino software libraries, so that existing "sketches" run on Galileo's processor. The board is also designed to be hardware and software compatible with the Arduino shield ecosystem (See Fig. 1). It can also be programmed by the Arduino software development environment (IDE) and libraries.

The 32-bit Quark SoC X1000 chip in Galileo is an attempt by Intel to compete within the IoT market. It is a single core, single thread processor based on an x86 Pentium design, and it runs at a clock speed of 400MHz. It has 512KB of embedded RAM and a 16 KBytes on-die L1 cache.

Galileo storage includes a programmable 8MB NOR flash, whose main purpose is to store the firmware (or bootloader) and the latest sketch, and a 256MB DRAM. Other key features include a micro-SD connector slot for cards of up to 32GByte of storage, a 10/100Mbps (bits per second) Ethernet port, a mini PCI-Express slot, a USB 2.0 host connector, and an RS-232 serial port connected via a 3.5mm jack. The USB Device ports allow for serial (CDC) communications over USB. This provides a serial connection to the Serial Monitor in the Arduino IDE.

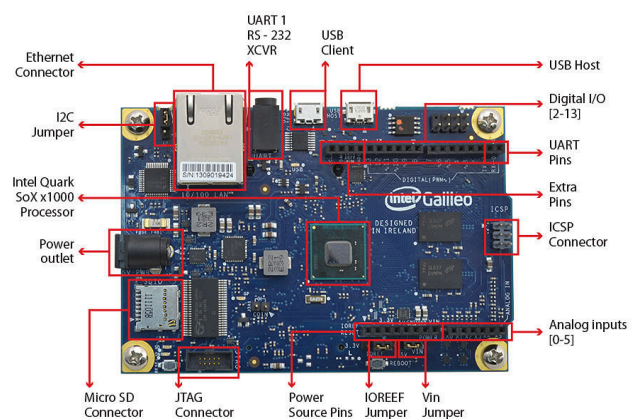


Fig. 1. Intel Galileo board front

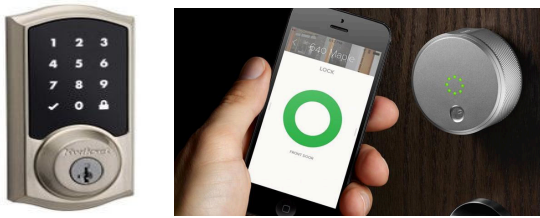


Fig. 2. Smart locks.

II. SMART LOCK

A smart or internet-controlled lock is useful for rental properties, home, apartment complexes, fraternities, or office use. This is a WiFi enabled door lock that allows users to lock or unlock doors remotely by a code, know when people unlock the door, and receive text alerts when codes are used. Door lock is managed from a computer or phone where new codes are issued or deleted including temporary codes to guests or office personnel. Some models include a keypad to enter a code and others simply let you use your phone to open and close doors (See Fig. 2).

One of the problems related to remotely managing property access is key exchange. Although it has associated copying and personnel costs, it can cause the disruption of early check-ins and overstays. In addition, there are security risks of unauthorized access. To solve this problem some internet-controlled lock companies even offer online integration with apps from hospitality service providers such as Airbnb.

Small business smart-locks using WiFi allow owners to remotely monitor and control access, to view history logs of employee access, and even to make sure doors are locked.

These locks are useful for office suites, health facilities, space sharing offices, interior offices, and utility rooms.

Many smart locks offer a mobile app that allows users to lock and unlock doors with a simple icon tap. Some offer a Web app that lets you control things from your desktop or laptop PC. These apps let you add permanent and temporary users and set access schedules for specific days and times.

The latest smart locks offer features like voice activation, auto-locking features, keyless touchpads for those times when you don't have your phone or your keys, and tamper and forced entry alarms that warn you of a possible break-in, and push, text, and email notifications that let you know who is coming and going in real-time. Some locks also integrate with other connected home devices. For example, you can have your doors unlocked 'when a smoke or CO alarm is triggered, or have certain smart lights turned on when a door is unlocked. You can even pair your lock with a connected doorbell cam, so you can see who is at the door before you unlock it.

Although there are several smart locks in the market, there are still two obvious reasons to develop an open source solution. One is the price, and the other is that those products have proprietary software and hardware.

III. DEVELOPMENT

Readers can download and study the source code and a step-by-step guide (in Spanish) [6] to implement this system in a Galileo board. This section explains the architecture and design of the system, to help readers not only to understand how the project works but also to be able to extend or modify this open source system.

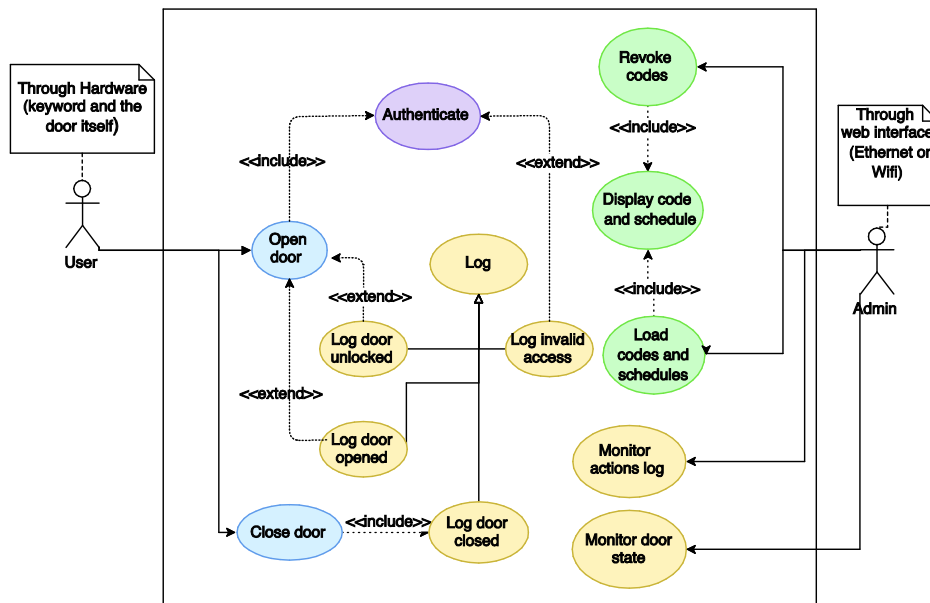


Fig. 3. UML use case diagram with the specification of the smart lock functionality.

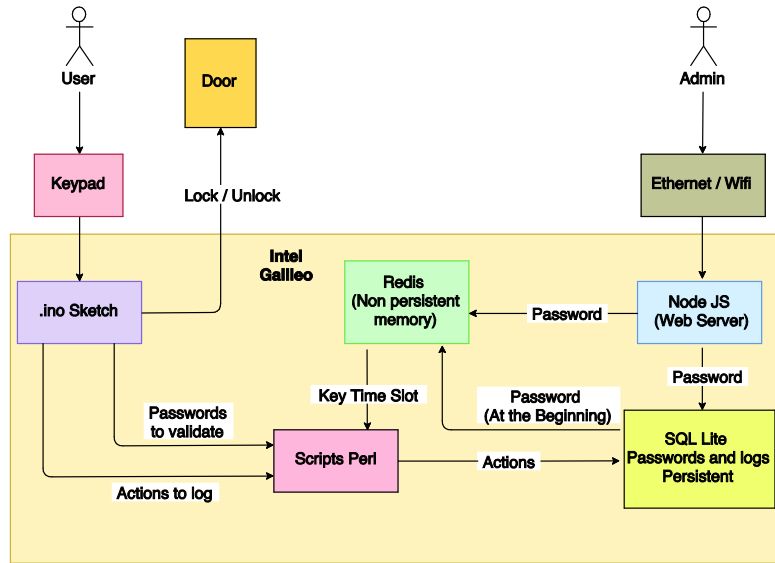


Fig. 4. Smart lock architecture.

A. Specification

For the smart lock defined for this project, there are two types of actors: on the one hand, the "users" who will use the lock to open the door and access the restricted site; on the other hand, the "administrators" who configure virtual keys and access date and times, and monitor the operation (see when and who open or close the door, etc.).

Users receive a code or virtual key by some text messaging service. Then, they can enter this code to unlock the door. Administrators can add codes and associate them to users and a time period, delete or disable codes, and see the logs. Logs show a list of users and timestamps when the door was unlocked, when it was opened, closed or locked, and the time

when an invalid code was attempted. Fig. 3 shows the smart lock functionality in an UML use case diagram.

Two additional functionalities were added, a display of the the current door status, and the possibility of synchronizing the internal clock in the smart lock by means of the web interface.

B. Architecture and Design

In this project, the Linux operating system (OS) running on the Galileo boards is used several times. This is one of the most important features of these cards. The OS can be accessed from Arduino sketches through system calls, i.e., the system() method, for example, to set up the WiFi card:

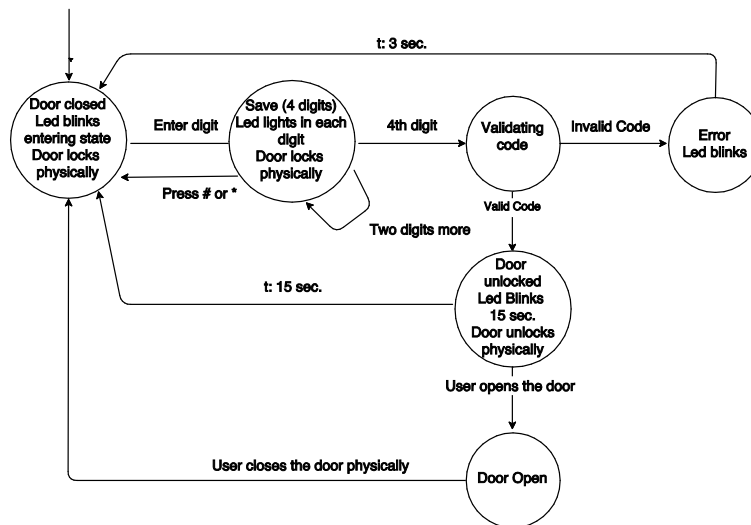


Fig. 5. Smart lock FSM.

```
system("connmanctl enable wifi");
```

Although security is one of the biggest IoT challenges, it is not a specific design issue considered in this article. However, as the system is running on a Linux OS, we encourage designers to apply at least the basic security measures such as changing root password, regularly updating packages and firmware, adding a domain name and SSL certificate to their board, among others.

Two forms of storage are observed in Fig. 4. SQLite is used for storing the keys and schedules in a persistent way. SQLite databases are stored as files in the filesystem, making it available after restarting the hardware (persistence). The problem with this kind of storage is the low performance retrieving results, in terms of response time. To improve the overall performance and speed up the response time to user operations, we decided to use Redis, an in-memory storage system. Redis is used to keep the keys in memory so that accesses can be validated quickly, delegating SQLite to the task of keeping the persistence of the data and working in an asynchronous way (without delaying the response time when possible). Other advantages of using Redis are its simple usage (no SQL queries are required) and its integration within all the technologies involved in the project, i.e., a variable stored in Redis can be read from a Perl script, a JS script, or the main .ino script. In other words, Redis is also used as a centralized repository for storing and sharing variables between programs in the OS. The SQLite database contains only two tables, one to store the codes and another to keep a record of the performed actions. Fig. 4 also illustrates the

Arduino code in execution as well as a web server in NodeJs (javascript). Intermediate scripts in Perl are used to access persistence options from Arduino (Redis / SQLite).

Fig. 5 shows the finite state machine (FSM) implemented as a sketch in Arduino and running on the Galileo board. When a user enters a digit using the keypad, the FSM goes from *locked* to *writing-key* state. After the user enters a 4-digit key, the FSM goes to the *validating* state, and then to *error* or *unlocked* state if the code is valid compared with the codes in the database. Finally, from the *unlocked* state, the FSM goes to *open* state when the user opens the door. Main state transitions are registered in a log.

C. Implementation

This system requires OS services that are not provided by the Linux embedded in the Galileo Board. Fortunately, Intel made available and maintains different Board Images based on the Yocto project (Linux) for extending the capabilities of the board. These custom distributions include drivers and an environment which can contain Redis, NodeJS, Perl, and/or other pre-installed tools and code interpreters, making them an excellent option for projects like this. For our particular project, we used the "Intel® IoT Developer Kit v1.5 Intel® Galileo Board Image".

A mini-PCIe WiFi card Atheros ar5b225 is used. This card is reused; it comes from a laptop that is out of service. However, the Intel® Centrino® Wireless-N 135 and Intel® Centrino® Advanced-N 6205 would be better options as they work with any Linux image provided by Intel.

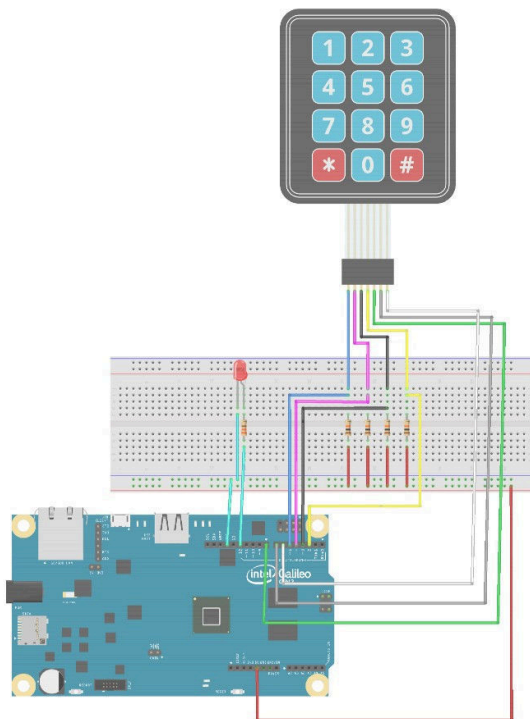


Fig. 6. Keypad connection

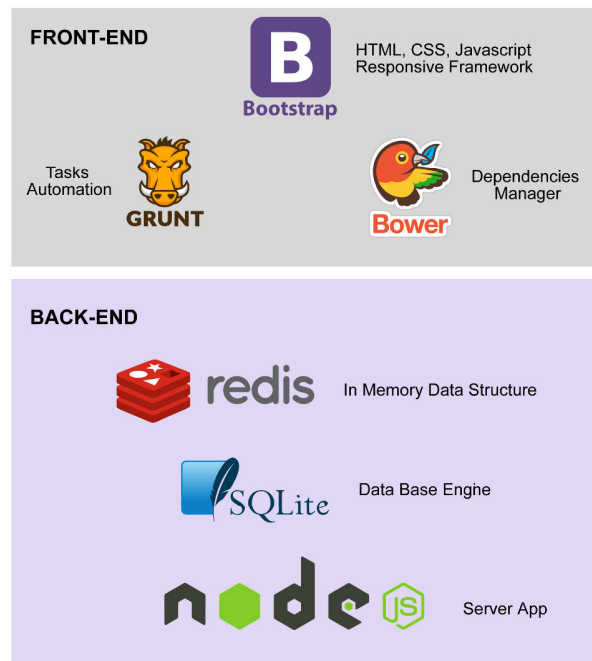


Fig. 7. Front-End / Back-End Tools and Technologies Ecosystem

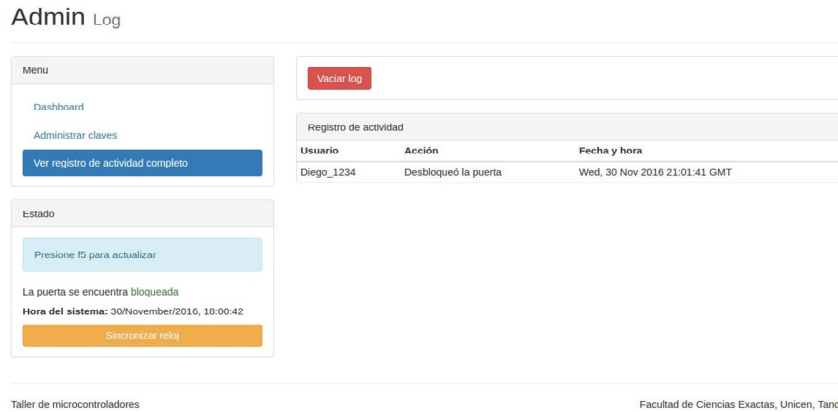


Fig. 8. Admin Log Screenshot

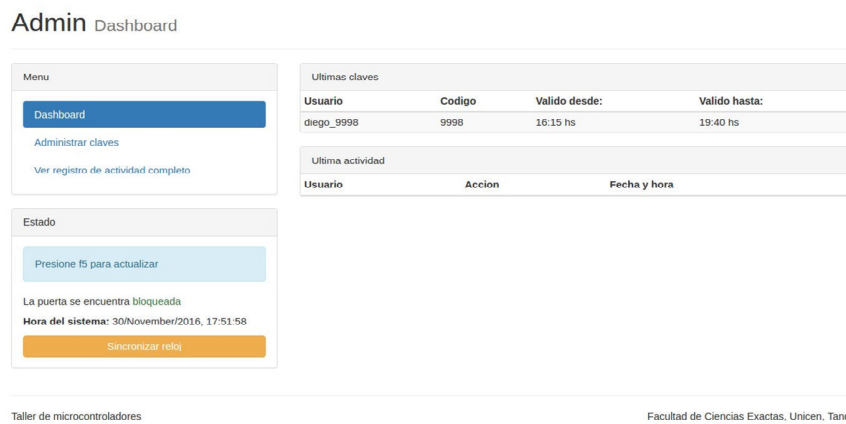


Fig. 9. Admin Log Dashboard

On Arduino cards, the connection from the keypad outputs to the card is direct because the Arduino pins provide internal pull-up resistors, but in this project pull-up resistors were added to avoid glitches, as shown in Fig. 6.

Regarding Arduino programming, the sketch mainly consists of the implementation of the FSM shown in Fig. 5. The FSM library for Arduino is used to implement the FSM states and transitions between states in an organized way. All these states expect an *enter*, an *update*, and an *exit* callback function.

Fig. 7 summarizes front-end and back-end tools and technologies used for the project development. NodeJs [8] is used and configured as a lightweight server application. Due to its single-threaded architecture, it minimizes memory usage and avoids the cost of context-switching between threads. These reasons make it an ideal tool for real-time tasks and

GPIO (general purpose input and output) interaction. Bower [9] and Grunt [10] are used as front-end development tools. The former for frameworks, libraries, assets, utilities, and dependencies management, and the latter for tasks automation. Both tools allow a modular development, and ease updating, maintenance and application debug. Bootstrap [11] was included as responsive front-end framework and SQLite [12] as database engine.

Redis [13] is an efficient in-memory key-value database open-source software. Many languages have bindings to Redis such as the ones used in this project JavaScript (Node.js) and Perl.

Figs. 8, 9 and 10 show the web application and database queries for Admin Log, Admin Dashboard, and Password Administration, respectively.

Admin Claves

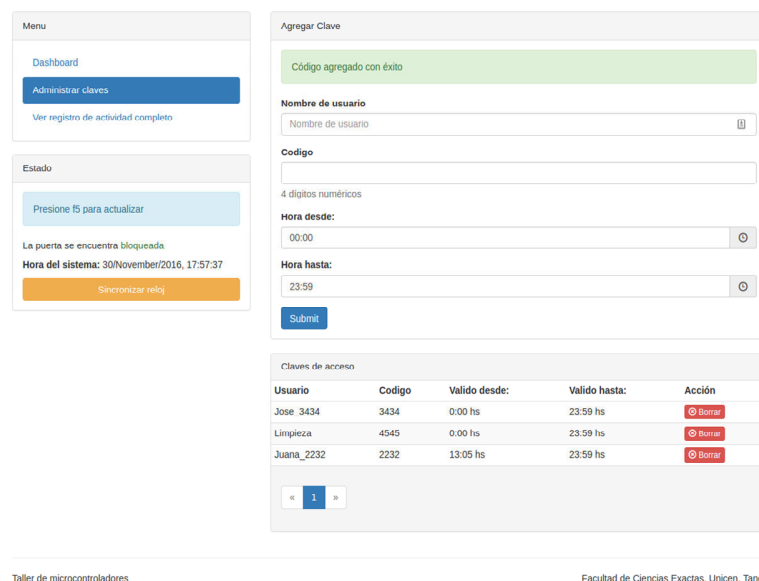


Fig. 10. Pass Admin Screenshot

IV. CONCLUSION

A complete smart lock open-source system is implemented on a Galileo gen 1 development board. This means that system administrators can access the web server running on the board itself, and the users can enter codes that are validated by on-board software too.

Authors provide a step-by-step guide to install and test the system. Besides, developers around the world can extend and modify the available source code with minimum complexity. Although this is the first available version of the proposed system, its architecture enables modular development, management, automation, and eases updating and maintenance by means of stable, professional, and widely accepted software tools.

ACKNOWLEDGMENT

Authors would like to thank: Intel Corporation for their support by providing Intel Galileo boards under Intel – University donation program as part of Intel higher education to the UNICEN, Argentina, and FASTA University, Argentina, and CIC PBA where M.P. belong to the Professional Staff.

REFERENCES

[1] J.J. Livingston, and A. Umamakeswari, "Internet of Things Application using IP-enabled Sensor Node and Web Server," Indian Journal of Science and Technology [Online], 8.S9 (2015), pp. 207-212.

[2] G. E. De Luca, E. A. Carnuccio, G. G. Garcia and S. Barillaro, "IoT fall detection system for the elderly using Intel Galileo development boards generation I," 2016 IEEE Congreso Argentino de Ciencias de la Informática y Desarrollos de Investigación (CACIDI), Buenos Aires, 2016, pp. 1-6.

[3] A. Kassem, S. E. Murr, G. Jamous, E. Saad and M. Geagea, "A smart lock system using Wi-Fi security," 2016 3rd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA), Beirut, 2016, pp. 222-225.

[4] Y. T. Park, P. Sthapit and J. Y. Pyun, "Smart digital door lock for the home automation," TENCON 2009 - 2009 IEEE Region 10 Conference, Singapore, 2009, pp. 1-6.

[5] Miguel de Sousa, Internet of Things with Intel Galileo, Packt Publishing, 2015.

[6] Diego Scafati, "Guía paso a paso: Cerradura digital con código en Galileo", available in: <https://github.com/dscafati/arduino-electronic-door-lock>

[7] Scott Rifenbark, "Yocto project mega-manual", available in: <http://www.yoctoproject.org/docs/2.2.1/mega-manual/mega-manual.html>

[8] Galileo Tutorial Networking and node.js Sensations 2014, Jason Wright, available in: <http://sensations.net/wp-content/uploads/2014/66/Sensations14-Networking-nodejs.pdf>

[9] Bower Package for the Web, <https://bower.io/docs/config/>

[10] Grunt Javascript Task Runner, <https://gruntjs.com/configuring-tasks>

[11] Bootstrap Responsive Framework, <http://getbootstrap.com/getting-started/>

[12] SQLite Documentation, <https://www.sqlite.org/docs.html>

[13] Redis Documentation, <https://redis.io/documentation>

Characterization of Sensors and Design of an Embedded Photodetectors Array for Beam Profile Measurements in Radiotherapy

H. Mateos, J. Lipovetzky, F. Alcalde Bessia, M. Pérez, P. Cappagli, and M. Gomez Berisso
Low Temperatures Laboratory and Medical Physics Department, Bariloche Atomic Center, CNEA.
CONICET, Balseiro Institute.

San Carlos de Bariloche 8400, Argentina. Telephone: (+54) 294 44451000 ext 5349
emails: horacio.mateos@ib.edu.ar, lipo@ib.edu.ar

Abstract—In this work, the dosimetric performance of several Commercial Of The Shelf photodiodes was studied. They were tested against gamma rays from a teletherapy source and were characterized with the aim of building a one-dimensional array of dosimeters able to measure the the spatial distribution of dose in a radiotherapy irradiation field. An embedded system comprising an array of 40 photodiodes and its acquisition electronics was built and tested. The dose resolution of each detector was 0.06 cGy and the spatial resolution of the system 0.5 cm.

Index Terms—Radiotherapy, Ionizing Radiation, Photodiodes, Quality Assurance.

I. INTRODUCTION

Radiotherapy is the medical discipline that takes advantage of the biological effects of ionizing radiation for the treatment of tumor diseases. The application of ionizing radiation on clinical environments involves multiple stages, from prescription to treatment execution. A relevant stage for treatment success is the quality assurance of the linear accelerator dose delivery. For example, on IMRT -Intensity-Modulated Radiation Therapy- it is fundamental to control the spatial dose distribution delivered by the system [1].

To perform the quality assurance of the irradiation equipment, the spatial distribution of dose delivered can be measured using radiocromic films or a planar array of electronic detectors. The later presents the advantage of immediate reading and evaluation, and also it can be embedded on the radiotherapy equipment or be portable. It is important that the detectors has a linear response to dose, because the variations could be lead to a wrong dosimetry, and consequently, an error in the dose delivered to the patient. Usually, an uncertainty of 5% is acceptable [1], [2].

These planar detectors can be built using PIN photodiodes which have appealing characteristics for dosimetry, specially for the measurement on small fields. This is due to its smaller active volumes and higher radiation sensitivities compared to the ionization chambers [3]. However, portable arrays of detectors used for the characterization of irradiation fields are expensive, ranging ten thousand dollar, and usually have ad-hoc designed detectors. One commonly used is MAPCHECK,

manufactured by *SUN NUCLEAR*. This system consists of a two-dimensional array of photodiodes, arranged symmetrically [4].

With the aim of developing low cost portable planar detector, we characterized and compared the performance of different low cost Commercial Of The Shelf -COTS- PIN photodiodes for their use in an array of sensors in radiotherapy. To reduce the cost of the final system, we used low COTS PIN photodiodes [6]. These devices are designed for optimal response to infrared and ultraviolet photons, but have been used in a large number of experiments with radiation with good results [2], [5]. Photodiodes are usually used as radiation detectors in a large number of applications: monitoring of X-ray equipment in industrial and medical applications, X-ray spectroscopy for identification of radioactive materials and spectroscopy of charged particles between others. Thanks to the versatility of these components, they can also be used in external beam radiotherapy [1].

After the selection of a photodiode with a high responsivity and low dark current, we built an array of aligned photodiodes, and the embedded electronics required to characterize the dose delivered during an irradiation. This first prototype is implemented as a proof of concept for future development of a two-dimensional array of detectors. However, a one-dimensional array is good enough for beam dose profile verification in contrast with treatment planification. Agreement between the delivered and planned dose is a quality assurance test for radioterapy treatments. [1].

The next Section, presents the characterization of the photodiodes selected in order to choose the best for this application. Also shows the embedded system and the final circuit boards designed and manufactured. Section III shows the response of the system designed to an X-Ray beam from a copper source. Finally Section IV presents the conclusions of this work.

II. DESCRIPTION OF THE SYSTEM

On the next section, will be shown the characterization, of the selected PIN photodiodes, against radiation, in order to

select the best suitable for the purpose of this work. Later, it will be show the design of the array of photodiodes and its associated electronics.

A. Photodiode Characterization

Since the photodiodes used in this work are manufactured to detect visible or near to visible light, the information published by the manufacturers has not data about the response against ionizing radiation. A first selection of four COTS photodiodes was done taking into account the parameters published by the vendors only: sensitive area, to obtain a high sensitivity; low reverse dark current to reduce systematic errors in dose measurement; low junction capacitance which is an indicator of thick depletion region; and cost. The four devices selected were VEMD5060X01, TEMD5110X01, BPW34FAS, and BPW34S. All of them had an active area of 7.5 mm^2 .

It is necessary to experimentally study their response to ionizing radiation had to be characterized measuring their photocurrent, their reverse voltage sensitivity dependence, angular response, dose rate dependency of the responsivity and dark current. The photodiodes were irradiated using a ^{60}Co source of a Telecobalt Therapy equipment -TERADI 800-. The dependence of sensitivity with bias voltage, incident angle, and dose rate was studied. In all cases, the dose rate and the dose delivered was a parameter from the Telecobalt Therapy equipment.

1) *Sensitivity and bias dependence:* To use the PIN photodiodes as radiation sensors, it is necessary to apply a reverse voltage to bias. We studied the dependence of the current generated by the photon beam as a function of the inverse voltage applied to the photodiode, which modulates the depth of the depletion region in the detector. Using a Keithley 617 electrometer, the radiation induced current in the devices was integrated during the time required to deliver 8 cGy . The bias voltage was varied between 1.0 V and 7.5 V and the irradiation was performed at a constant dose rate of $71.95 \frac{\text{cGy}}{\text{min}}$. In Fig. 1 we present the current measured in the four different devices as a function of reverse voltage. In Table I we show the current slope for each photodiode. The method of least squares was used to fit the data.

TABLE I: Current variation with applied bias for each photodiode.

Model	Bias Sensitivity Variation (BSV) [$\frac{\text{pA}}{\text{V}}$]
VEMD5060X01	-25.3
TEMD5110X01	-83.1
BPW34FAS	-153.6
BPW34S	-107.3

2) *Angular response:* We measured how much the response of the photodiodes varies with the angle of incidence of the irradiation beam. In order to homogenize the scatter radiation that the photodiodes may receive, the detectors were inserted into an acrylic cylinder of 1.5 cm of thickness. The beam adjustment was done in order to deliver 9 cGy of total dose in each angle -varying from 0° to 180° - with a bias voltage of 2.5 V . In Fig. 2 the results of the measurements are shown. An

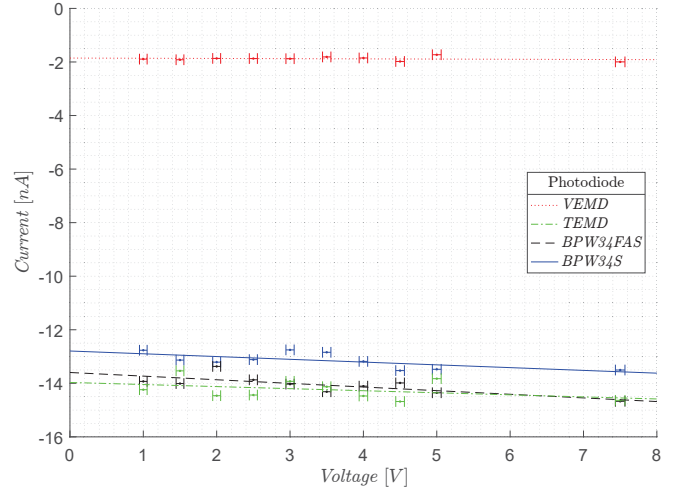


Fig. 1: Photocurrent as function of reverse voltage bias on the four devices.

angular dependence of the photodiode response is observed, the measured current reduces up to a 20% when the devices are irradiated from the side with respect to normal incidence i.e. 90° . Table II shows the maximum percentage variation of response respect of normal incidence measured in each photodiode.

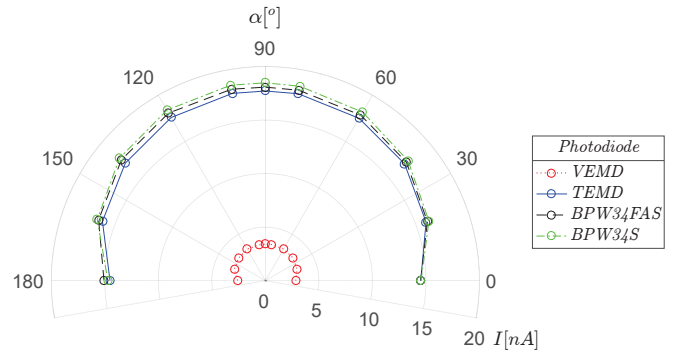


Fig. 2: Current as function of angle.

TABLE II: Maximum percentage sensitivity variation of response.

Model	Maximum percentage variation of response (MPVR)
VEMD5060X01	20%
TEMD5110X01	18%
BPW34FAS	18%
BPW34S	20%

3) *Response due to different dose rate:* The devices were irradiated at different dose rates, placing them at different distances from the ^{60}Co source, with a bias voltage of 2.5 V . Fig. 3 shows the current as a function of the dose rate for each photodiode, and Table III presents the sensitivity as function of dose rate.

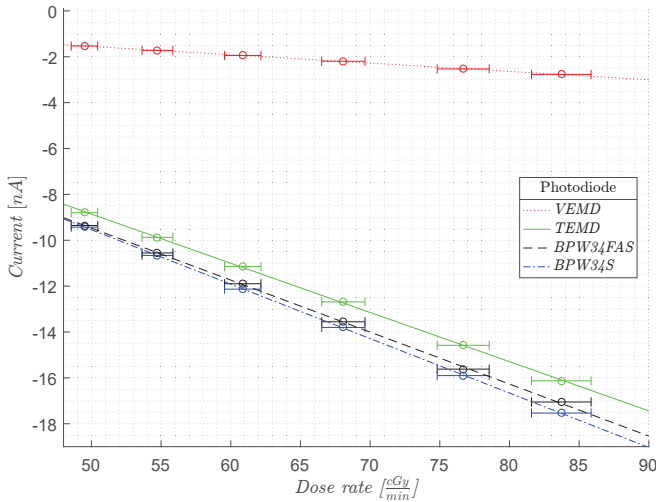


Fig. 3: Current as function of Dose Rate.

TABLE III: Sensitivity as function of dose rate.

Model	Sensitivity [$\frac{pAmin}{cGy}$]
VEMD5060X01	-36.4
TEMD5110X01	-214.5
BPW34FAS	-226.7
BPW34S	-237.3

No significative increase in the dark current of the devices was observed after a total dose of 100 Gy.

Among the four devices, we selected the BPW34FAS photodiode because it was the one with sufficient sensitivity, no aging on the dose range measured, and it comes factory obscured for visible light.

4) *Photodiode Selection*: In order to construct the circuit board, it is necessary select one of the four PIN photodiodes tested. In Table IV it is possible compare the tested characteristics and specific parameters.

TABLE IV: Comparison of the results of tests and characteristics on each photodiode.

Model	(BSV) [$\frac{pA}{V}$]	(MPVR) %	Sensitivity [$\frac{pA}{cGy}$]	Pre obscuration
VEMD5060X01	-25.3	20	-36.4	NO
TEMD5110X01	-83.1	18	-214.5	NO
BPW34FAS	-153.6	18	-226.7	YES
BPW34S	-107.3	20	-237.3	NO

The BPW34FAS was the photodiode which has the best combination of characteristics, furthermore it comes pre obscured from visible light, which helps the later obscuration.

III. ACQUISITION SYSTEM

The acquisition system consist of a current integrator circuit made with an operational amplifier with a capacitor in the feedback loop, as shown in -Fig. 4-. In order to reset the measurement, a reset transistor was added in parallel with the capacitor.

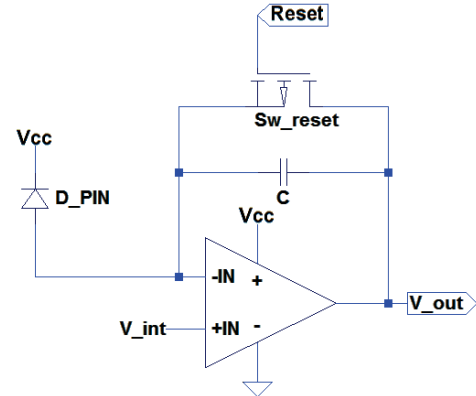


Fig. 4: Integrator schematic.

The output of the operational amplifier is proportional to the integrated photodiode current during the measurement. Initially, the capacitor C is at 0 V, as it was previously reset by switching on Sw_{reset} . This transistor has a very low leakage current, in the order of tens of picoamperes. Then, the transistor turned off, and the charge generated in the photodiode D_PIN is integrated, generating a voltage ramp on V_out . Measuring the initial and final voltage, the absorbed dose by the photodiode can be calculated. Due to the ultra low current generated by the PIN photodiode we have selected an operational amplifier with 1 pA of input bias current.

In order to acquire the voltage of each integrator of the array -40 photodiodes in a row-, it is necessary to have a system that can select each integrator, convert the signal from analog to digital and then send the value via USB to a PC, which shows the measured dose in real time. An Arduino UNO was used to design a system that carries out the required steps. Its software flow chart is shown in Fig. 5.

The Arduino select which input of the multiplexer it is necessary read, put the address, and acquire with the analog input the voltage value of each 5 integrator selected and the 6th input sense the voltage supply. This step it was repeat 50 times, then, the Arduino, average the samples to reduce the noise and send the values via USB. If the Arduino receive the order to reset the system, automatically put on his 5 outputs to the reset transistor a logic 1, which pull up the voltage of each capacitor to the 2.5 V, the start value. Finally, when the Arduino finish reading all 40 integrator, it will reset the counter and start the cycle again.

A. Implementation of the photodiode linear array

A prototype of the system was implemented using a linear array of 40 photodiodes, with their respective current integration circuit. The prototype was designed to achieve maximum linear density of photodiodes. Due to the size of the packaging of the diodes, the maximum density achieved was 2 detectors per cm, which gives 40 detectors in the length of 20 cm, sufficiently large to characterize most irradiation beams. In Fig. 6 we show a 3D model of the PCB with photodiodes.

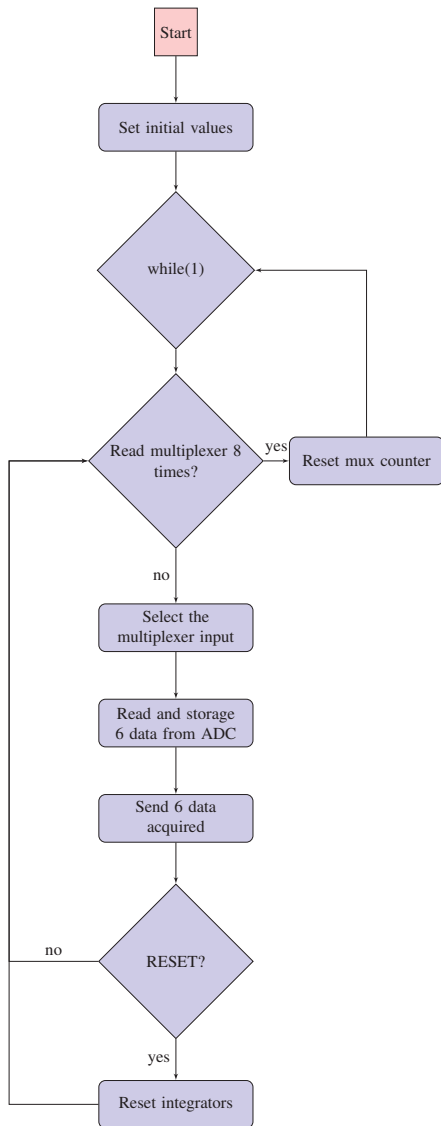


Fig. 5: Flow chart of program.

It is necessary count with a system capable of acquire analog signals, convert them into digital and send them via USB. Also it is necessary that the system could mange a series of digital outputs in order to control the address of multiplexers and the gates of the reset transistors. For all of that, it was chosen an Arduino UNO board. This system can manage all the requirements and also fits with the idea of the portable, cheap, and easy to use dosimetric system. Because the Arduino UNO only has 6 analog inputs it is necessary use multiplexer to addressed the voltage of each integrator on the respectively inputs.

In order to shield the photodiodes from environmental light, they were painted with black synthetic enamel, which, after 3 layers, creates a good filter for the spectrum of light between infrared and ultraviolet. In addition, the enamel does not conduct electric current, avoiding electrical leakage.

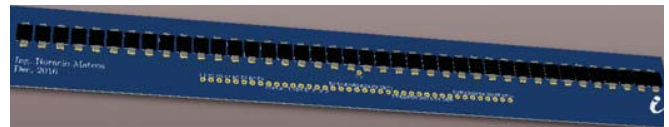


Fig. 6: Array of photodiodes circuit board.

The 40 acquisition circuits were placed in another board, which includes the multiplexers needed due to the fact that there are only 5 analog inputs available in the Arduino UNO. The board with the integrators and multiplexers is shown in Fig. 7. This board should be placed outside the irradiation field, since the devices showed a fast degradation with total ionizing dose [7] [8].

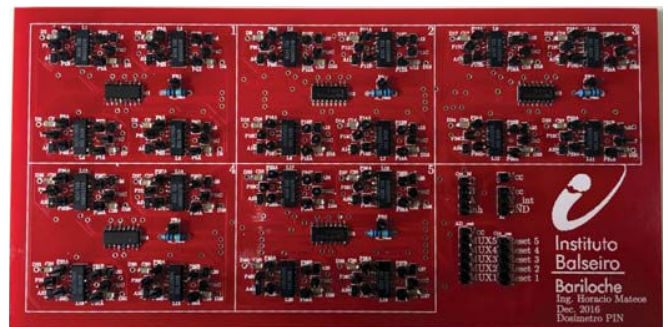


Fig. 7: PCB with integrator circuits and multiplexers.

The supply voltage was sensed with the sixth analogue input of the Arduino board, in order to perform a correction of the error on the voltage reference of the Digital Analog Converter.

IV. RESPONSE OF THE SYSTEM IN AN IRRADIATION FIELD

The ability of the system's prototype to characterize the spatial distribution of dose in an irradiation field was verified using a X-ray source from a SIEMENS KFL Cu-2K diffractometer at the Centro Atómico Bariloche. This experiment allowed to demonstrate the correct functioning of the prototype, in the measurement of dose profiles. In all cases, a bias voltage of 2.5 V was used. The dose rate it is not possible to obtain with certainty, because is not a parameter of the diffractometer.

A. Response of a single detector

The equipment was adjusted to deliver the highest possible X-ray intensity, and generated a beam with a field size of approximately 1 cm × 0.5 cm.

Fig. 8 shows the temporal response of the voltage in one of the photodiodes exposed in the center of the beam during an irradiation. It is observed that the voltage at which the integrators were before being irradiated is close to 2.5 V. During irradiation, this voltage drops due to the integration of the charge collected by the photodiode.

In order to reduce the voltage measurement error below 1.55 mV, 50 samples of the output voltage of each photodiode were averaged. Using a conversion factor obtained with the

measures on Fig. 3 and the value of the capacitor C , it is possible to reach a dose resolution of 0.06 cGy in each photodiode.

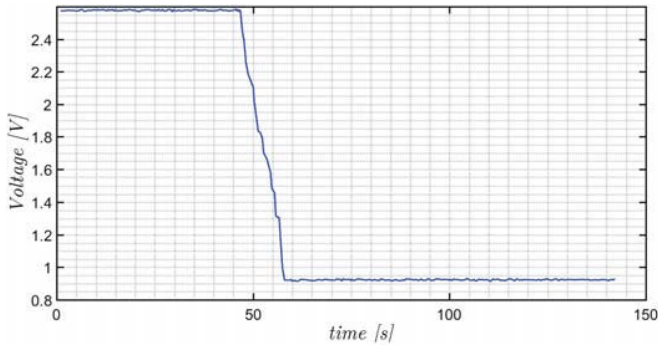


Fig. 8: Temporal response of a single photodiode faced to an irradiation beam.

B. Repeatability

The previous experiment was repeated ten times in order to study the repeatability of the response. The sensitivity values for each irradiation are shown in Table V.

TABLE V: Output voltage slope of the sensor in each irradiation.

Irradiation	Slope [$\frac{mV}{s}$]
1 st	-145.3
2 nd	-150.5
3 th	-152.6
4 th	-150.0
5 th	-147.5
6 th	-152.0
7 th	-151.9
8 th	-151.5
9 th	-150.4
10 th	-152.1

The mean value of the slopes and their standard deviation was $(-150 \pm 2) \frac{mV}{s}$, showing that the response is repeated along the irradiations within a variation of 1.33%.

C. Response to different dose rates

The system was tested using beams of different intensities to evaluate the response of the photodiodes. The intensity of the X-rays was varied by changing the current of the X-ray tube. Starting at 40 mA , the current was halved in each step until reaching 5 mA . In Fig. 9 different slopes can be seen along the time, which are due to the different intensities. The drop in voltage -which it is correlated to the total dose- of each irradiation are plotted as function of the intensity of the beam in Fig. 10. It can be observed an independence between dose measurement with the dose rates within the measurement error in the range of dose rates used.

The line shown in Fig. 10 verifies the linearity of the response in the energy range tested.

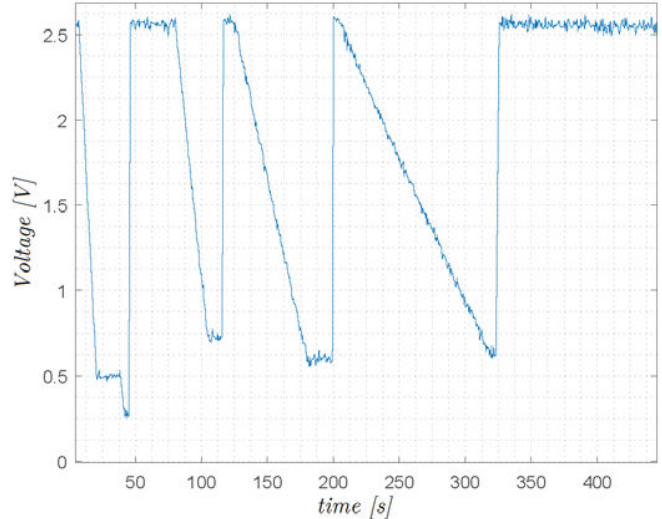


Fig. 9: Slopes of irradiation with 40, 20, 10 and 5 mA.

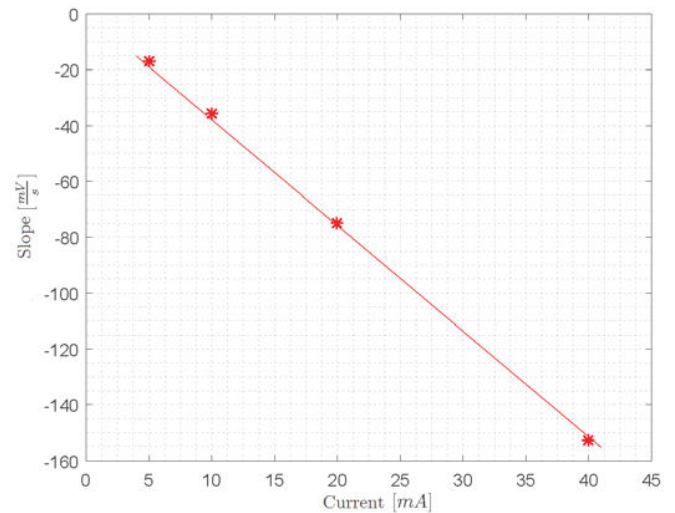


Fig. 10: Slopes of Fig. 9 vs. tube current.

D. Characterization of a beam dose profile

The system was tested against the X-ray beam of the diffractometer mentioned before with the aim of measuring the profile of doses delivered by the beam, which had a width of 1.0 cm . In order to show the measures on real time, an interface was designed, Fig. 11 shows a screen shot of a part of the interface that shows the measurements, plotting in a color code the spatial dose distribution in a simplified way. The system indicates that only three photodiodes were exposed to the beam, which is consistent with the spatial resolution of 0.5 cm and beam width of 1 cm . The other detectors receive negligible dose.

The interface also presents the plot shown in Fig. 12. The voltage of the diodes that are located outside of the beam are not modified from the initial value close to 2.5 V , while the photodiodes arranged around $(-2.75 \pm 0.50) \text{ cm}$, showed a

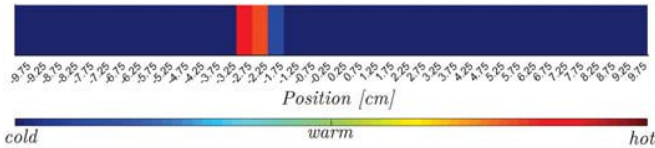


Fig. 11: Part of the interface which shows, simplicity, the absorbed dose of each photodiode.

shift in the output voltage due to the exposure to the irradiation beam, measured a higher dose manifested in a shift of the voltage of the integrators with respect to the initial (2.5 V).

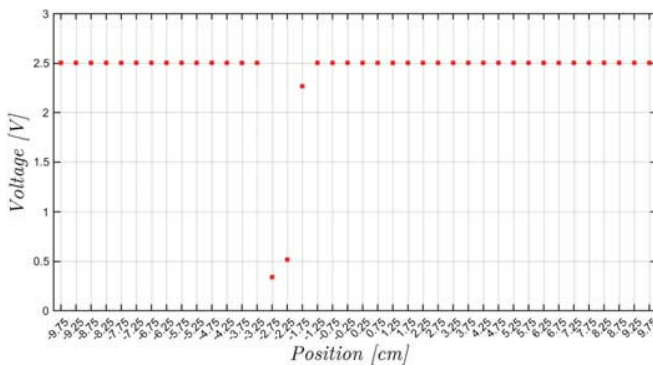


Fig. 12: Part of the interface which shows the value of each integrator.

V. CONCLUSION

A first prototype of a dosimetry system for quality assurance of teletherapy treatments was designed, manufactured and tested, consisting of a set of 40 detectors arranged 5 mm each other, capable of characterizing the profile of an irradiation beam. The prototype consists of three boards, one with the array of detectors that is located within the beam scope, the other with the charge integration electronics that is located outside the irradiation beam, and a third board, the Arduino, which is an embedded system for sampling, acquisition and communication via USB with a PC.

The response to gamma radiation of four different COTS PIN photodiodes was studied using a ^{60}Co source. From the four devices, the BPW34FAS was selected because it was the photodiode that shows more sensitivity, less variation of response due to angular irradiation and came from the factory with a visible light shield, which helps the subsequent obscuration.

To characterize the behavior of the final prototype - circuits boards with photodiode arrangement and associated electronics- it was irradiated in the facilities of the Centro Atómico Bariloche under an X-ray beam. It was verified that the final prototype circuit works as it was designed. Using the prototype, the beam profile of the irradiation was measured correctly. The repeatability of the measurements and the correct functioning at different dose rates were verified. The prototype allows the measurement of irradiation dose profiles

in real time and with a resolution of 0.06 cGy.

ACKNOWLEDGMENT

The authors would like to thank to the team at FUESMEN for their help with the TERADI irradiation equipment, to Federico Pomiro for his help with the use of the X-ray system. This work was done under grants PICT 2014-1996, PIP Res. 5205/15.

REFERENCES

- [1] Sanz D., Cappagli P., Alvarez G., Lopera R., "Significancia estadística de los controles de calidad dosimétricos realizados con matrices de detectores portátiles", Bariloche - Mendoza, Argentina (2016)
- [2] Rosenfeld A. "Electronic dosimetry in radiation therapy" *Radiation measurements S134-S153* (2006).
- [3] Low D., Moran J., "Dosimetry tools and techniques for IMRT", *Med. Phys.* **38** 1313 (2011)
- [4] Oh M., Malhotra H., Podgorsak M. "Dosimetric Comparison of a Semi-Conductor Array (MapCheck), EDR2 Film and Ion-Chamber in the Commissioning of Enhanced Dynamic Wedges On a Varian Linear Accelerator (21 EX)" *Med. Phys.* (2006)
- [5] Mondragn Contreras L., Ramirez Jimenez F. J. "PIN Photodiodes as Radiation Detectors in Accelerator Applications".
- [6] Andjelkovic M. S., Ristic G. S., "Feasibility study of a current mode gamma radiation dosimeter based on a commercial PIN photodiode and a custom made auto-ranging electrometer", *Nuclear Technology and Radiation Protection* **28** (2013).
- [7] Holmes-Siedle, A., & Adams, L. (1993). Handbook of radiation effects.
- [8] Mesenger G. C., Ash M. S. "The Effects of Radiation on Electronic Systems" *Van Nostrand Reinhold* (1992).
- [9] Whitaker J. C. (2005). The Electronics Handbook.

Author Index

Author	Page
Alcalde, Fabricio	77
Álvarez, Nicolás	1
Amor, Manuel	59
Arnaldi, Luis Horacio	47
Banfi, Damián	65
Blotta, Eduardo	0
Briff, Pablo	31
Cappagli, Pablo	77
Carrasco, Rolando	31
Cayssials, Ricardo	13,65
De Cristóforis, Pablo	35
De Pasquale, Lorenzo	65
Errobidart, Javier Omar	25
Etchevery, Juan Alberto	25
Ferrari, Mariano	13
Ferro, Edgardo	65
Fusari, Diego Salvador	59
Garrido, Alejandra	41
Gelosi , Ivan Exequiel	25
Gomez Berisso, Mariano	77
González, Esteban Lucio	25
Grop, Andrés	59
Larosa, Facundo Santiago	1
Lipovetzky, José	77
Lutenberg, Ariel	31
Marone, José	71
Martín, Diego	35
Martinez, Diego	65
Martos, Pedro	41
Mateos, Horacio	77
Mignone, Martín	1
Nitsche, Matías Alejandro	35
Ordoñez, Cristian	53
Orozco, Javier D.	13
Paez, Francisco E.	13
Pastore, Juan	53
Patway, Mohammad	31
Pérez, Martín	77
Penia, Eric Nicolas	7
Pessacg, Facundo	35
Ponso, Hernán	59
Presso, Matías	71
Rey Vega, Leonardo	31
Safar, Félix	7
Scafati, Diego	71
Silva, Agustín	19
Solivellas, Pablo Nicolás	59
Teijeiro, Adrián	35
Todorovich, Elías	71
Uriz, Alejandro José	25
Urriza, Jose M.	13
Vargas, Fabián	31
Zabaleta, Omar Gustavo	19



www.sase.com.ar

August 9th - 11th, 2017
University of Buenos Aires, Argentina



CASE 2017

CONGRESO ARGENTINO DE SISTEMAS EMBEBIDOS
CASE 2017

Libro de trabajos en modalidad
Foro Tecnológico y Resumen

Publicación realizada con el apoyo de:
**Agencia Nacional de Promoción
Científica y Tecnológica**



www.sase.com.ar

Facultad de Ingeniería | UBA
Buenos Aires, Argentina

ISBN: 978-987-46297-3-9

CASE 2017

Libro de Trabajos

Modalidades Foro Tecnológico y Resumen

Congreso Argentino
de
Sistemas Embebidos

9 al 11 de Agosto de 2017
Facultad de Ingeniería, Universidad de Buenos Aires,
Buenos Aires, Argentina

CONICET



AGENCIA
NACIONAL DE PROMOCION
CIENTIFICA Y TECNOLOGICA



Congreso Argentino de Sistemas Embebidos CASE 2017 : libro de trabajos en modalidad foro tecnológico y resumen / Maximiliano Antonelli, Mariano García Inza, José Lipovetzky , Luciana De Micco, Ariel Lutenberg ... [et al.] ; compilado por Diego Javier Brengi. - 1a ed ilustrada. - Ciudad Autónoma de Buenos Aires : ACSE - Asociación Civil para la investigación, Promoción y Desarrollo de Sistemas Eléctricos Embebidos, 2017.

190 p. ; 29 x 20 cm.

ISBN 978-987-46297-3-9

1. Ingeniería. 2. Ingeniería Electrónica. 3. Circuitos Electrónicos. I. Antonelli, Maximiliano II. Brengi, Diego Javier, comp.

CDD 621.39

Fecha de catalogación: 2017

Libro de Trabajos
Modalidades Foro Tecnológico y Póster
Congreso Argentino de Sistemas Embebidos - CASE 2017

Editores:

Antonelli, Maximiliano.	UNMDP/ICyTE
García Inza, Mariano.	FIUBA
Lipovetzky, José.	IB/CNEA/CONICET
De Micco, Luciana.	UNMDP/ICyTE/CONICET
Lutenberg, Ariel.	FIUBA/CONICET/ACSE
Brengi, Diego.	INTI/UNLaM/FIUBA

Copyright © 2017

Asociación civil para la investigación, promoción y desarrollo de los sistemas electrónicos embebidos.



Se otorga permiso para copiar y redistribuir este libro de trabajos, siempre que se mantengan los mensajes de copyright y autoría de la obra y sus partes.

Prefacio

El diseño de sistemas embebidos es un motor clave de la industria y del desarrollo científico y tecnológico, y es un campo que en los últimos años ha crecido notablemente en la Argentina, tanto en la academia como en la industria.

El SASE busca fomentar esta temática realizando las siguientes actividades:

- CASE: Congreso Argentino de Sistemas Embebidos, presentación trabajos científicos.
- Workshops: Talleres prácticos en la modalidad hands-on.
- Tutoriales: Charlas de capacitación.
- Plenarias: Conferencias y debates abiertos.
- Concurso de proyectos tecnológicos: sobre trabajos finales y materias de grado.
- Concurso de emprendimientos tecnológicos: destinado a promover emprendimientos electrónicos con viabilidad económica.
- Programa de equipamiento para universidades: para transferir a las universidades las donaciones de los auspiciantes.
- Becas de viaje y alojamiento: Ayudas económicas para viaje y estadía a estudiantes de grado, estudiantes de doctorado, docentes e investigadores, de Argentina y Latinoamérica.

Los objetivos que persigue el CASE son:

- Ofrecer un lugar de encuentro para investigadores y becarios de todo el país, fomentando la colaboración.
- Difundir en el medio académico los adelantos científicos y tecnológicos producidos a nivel mundial.
- Propiciar la presentación y discusión de trabajos de investigación desarrollados en Argentina.
- Estimular en los estudiantes universitarios avanzados el interés por la investigación en el área de los S.E.
- Difundir los proyectos de investigación mediante el desarrollo de un sitio web.
- Coordinar y actualizar los contenidos de S.E. de los programas de grado y posgrado de las universidades argentinas.

Las áreas temáticas del CASE se organizan de la siguiente manera: Arquitecturas de Procesadores, Bioingeniería, DSPs, FPGAs, HDLs y ASICs, Implementación de Sistemas Embebidos, Protocolos y Comunicaciones, Robótica, RTOS, Software Embebido, Linux embebido y comunicaciones inalámbricas. Dentro de cada una de

estas áreas se permiten las modalidades Artículo, Foro Tecnológico y Resumen, según el tipo de trabajo.

Los trabajos presentados al CASE fueron sometidos a un proceso de revisión por pares y posterior corrección. De este modo fueron seleccionados 23 trabajos en la modalidad foro tecnológico y 15 en la modalidad resumen.

Esta publicación se encuentra también disponible en forma online en la página www.sase.com.ar/case.

Esperamos que los trabajos recopilados en esta memoria sean de su interés y contamos con su participación en futuras ediciones del evento.

Atentamente,

Comité Organizador CASE

Auspiciantes Diamond

- Cika Electrónica S.R.L.
- Electrocomponentes S.A.
- Synopsys

Auspiciantes Platinum

- Semak

Auspiciantes Gold

- CADIPEL
- Ernesto Mayer S.A.
- Probattery
- Emtech
- Asembli S.A.
- Dai Ichi Circuitos
- ClariPhy Argentina
- Telit
- Vicda Argentina

Auspiciantes Silver

- Digi
- Ubidots
- L&R Ingeniería

Instituciones Auspiciantes

Institución Organizadora

- ACSE (Asociación Civil para la Investigación, Promoción y Desarrollo de los Sistemas Electrónicos Embebidos)

Instituciones Co-organizadoras

- RUSE (Red Universitaria de Sistemas Embebidos)

Instituciones Auspiciantes

- ANPCyT (Agencia Nacional de Promoción Científica y Tecnológica)
- CADIEEL (Cámara Argentina de Industrias Electrónicas, Electromecánicas y Luminotécnicas)
- CAPER (Cámara Argentina de Proveedores y Fabricantes de Equipos de Radiodifusión)
- CONICET (Consejo Nacional de Investigaciones Científicas y Técnicas)
- Fundación Sadosky (Investigación y Desarrollo en TIC)
- IEEE Argentina (Institute of Electrical and Electronics Engineers)
- IEEE CASS (Circuits and Systems Society)
- ISOC (Internet Society)

Instituciones Acompañantes

- AADECA (Asociación Argentina de Control Automático)
- ADIMRA (Asociación de Industriales Metalúrgicos de la República Argentina)
- CAI (Centro Argentino de Ingenieros)
- CAME (Confederación Argentina de la Mediana Empresa)
- CAMOCA (Cámara Argentina de Máquinas de Oficina, Comerciales y Afines)
- CASEL (Cámara Argentina de Seguridad Electrónica)
- CEIL (Cámara de Empresas Informáticas del Litoral)
- CESSI (Cámara de Empresas de Software y Servicios Informáticos)
- CIIECCA (Cámara de Industrias Informáticas, Electrónicas y de Comunicaciones del Centro de Argentina)
- CNEA (Comisión Nacional de Energía Atómica)
- CONFEDI (Consejo Federal de Decanos de Ingeniería)
- INTI (Instituto Nacional de Tecnología Industrial)
- MinCyT (Ministerio de Ciencia, Tecnología e Innovación Productiva)
- Red UIE (Red Universitaria de Ingeniería en Electrónica)
- RUNIC (Red Universitaria de Ingeniería en Computación)

Universidades Auspiciantes

- UBA: Universidad de Buenos Aires
- UNAJ: Universidad Nacional Arturo Jauretche
- UNC: Universidad Nacional de Córdoba
- UNCA: Universidad Nacional de Catamarca
- UNCOMA: Universidad Nacional del Comahue
- UNCUYO: Universidad Nacional de Cuyo
- UNER: Universidad Nacional de Entre Ríos
- UNICEN: Universidad Nacional del Centro de la Provincia de Buenos Aires
- UNLAM: Universidad Nacional de La Matanza
- UNLP: Universidad Nacional de La Plata
- UNLu: Universidad Nacional de Luján
- UNNE: Universidad Nacional del Nordeste
- UNNOBA: Universidad Nacional del Noroeste de la Provincia de Buenos Aires
- UNM: Universidad Nacional de Misiones
- UNMDP: Universidad Nacional de Mar del Plata
- UNPA: Universidad Nacional de la Patagonia Austral
- UNPSJB: Universidad Nacional de la Patagonia San Juan Bosco
- UNQ: Universidad Nacional de Quilmes
- UNR: Universidad Nacional de Rosario
- UNRC: Universidad Nacional de Río Cuarto
- UNS: Universidad Nacional del Sur
- UNSA: Universidad Nacional de Salta
- UNSJ: Universidad Nacional de San Juan
- UNSL: Universidad Nacional de San Luis
- UNSM: Universidad Nacional de San Martín
- UNT: Universidad Nacional de Tucumán
- UNTF: Universidad Nacional de Tres de Febrero
- UTN-FRA: Universidad Tecnológica Nacional - Facultad Regional Avellaneda
- UTN-FRBA: Universidad Tecnológica Nacional - Facultad Regional Buenos Aires
- UTN-FRBB: Universidad Tecnológica Nacional - Facultad Regional Bahía Blanca
- UTN-FRD: Universidad Tecnológica Nacional - Facultad Regional Delta
- UTN-FRH: Universidad Tecnológica Nacional - Facultad Regional Haedo
- UTN-FRLR: Universidad Tecnológica Nacional - Facultad Regional La Rioja
- UTN-FRM: Universidad Tecnológica Nacional - Facultad Regional Mendoza
- UTN-FRN: Universidad Tecnológica Nacional - Facultad Regional Neuquén
- UTN-FRP: Universidad Tecnológica Nacional - Facultad Regional Paraná
- UTN-FRRE: Universidad Tecnológica Nacional - Facultad Regional Resistencia

- UTN-FRRG: Universidad Tecnológica Nacional – Facultad Regional Rio Grande
- UTN-FRSF: Universidad Tecnológica Nacional – Facultad Regional San Francisco
- UTN-FRSN: Universidad Tecnológica Nacional – Facultad Regional San Nicolás
- UTN-FRVT: Universidad Tecnológica Nacional – Facultad Regional Venado Tuerto
- UTN-FRVM: Universidad Tecnológica Nacional – Facultad Regional Villa Mercedes
- UTN-FRT: Universidad Tecnológica Nacional – Facultad Regional Tucumán
- UADE: Universidad Argentina de la Empresa
- CAECE: Universidad CAECE
- FASTA: Fraternidad de Agrupaciones Santo Tomás de Aquino
- ITBA: Instituto Tecnológico de Buenos Aires
- IUA: Instituto Universitario Aeronáutico
- UBP: Universidad Blas Pascal
- UCC: Universidad Católica de Córdoba
- UCU: Universidad Católica de Uruguay
- UCSE: Universidad Católica de Santiago del Estero
- UM: Universidad de Mendoza
- UREP: Universidad de la República del Uruguay

Coordinación General SASE

- Dr. Ariel Lutenberg (FIUBA/CONICET/ACSE)

Coordinación CASE

- Dr. José Lipovetzky (IB/CNEA/CONICET)
- Dra. Luciana De Micco (UNMDP/ICyTE/CONICET)
- Mg. Diego Brengi (INTI/UNLaM/FIUBA)
- Ing. Maximiliano Antonelli (UNMDP/ICyTE)
- Dr. Mariano García Inza (FIUBA)

Chairs

- Bioingeniería: Ing. Juan Manuel Reta (UNER)
- DSPs: Dra. María Liz Crespo (ICTP)
- Linux Embebido: Ing. Alejandro Furfaro (UTN-FRBA)
- Software Embebido: Dr. Ricardo Medel (Ascentio Technologies)
- FPGAs, HDLs y ASIC: Ing. Salvador Tropea (INTI/UTN-FRBA)
- Implementación de Sistemas Embebidos: Dr. Gustavo Sutter (UAM) y
Msc. Cristian Sisterna (UNSJ)
- Arquitectura de procesadores: Ing. Alejandro Furfaro (UTN-FRBA)
- Comunicaciones y protocolos: Ing. Gustavo Mercado (UTN-FRM)
- Robótica: Claudio Verrastro (CNEA)
- RTOS: Dr. Ricardo Cayssials (UNS)
- Comunicaciones inalámbricas: Dr. Leonardo Rey Vega (FIUBA)

Revisores

Aguirre, Fernando Leonel
Alessandrini, Gustavo
Alvarez, Nicolás
Antonelli, Maximiliano
Arias, Ricardo
Bos, Patricio
Brenji, Diego
Bulacio, Matías Fernando
Burgos, Enrique Sergio
Calarco, Nicolás
Carbonetto, Sebastián
Carrá, Martín
Cayssials, Ricardo
Comas, Edgardo
Crespo, Maria Liz
De Jesús, Sergio
De Micco, Luciana
Escudero, Gustavo
Ferrao, Hilda Noemí
Ferreyra, Pablo Alejandro
Filomena, Eduardo
Fraire, Juan Andrés
Furfaro, Alejandro
Gabian, Gabriel
Gak, Joel
Galleguillo, Juan
Gavinowich , Gabriel
Golmar, Federico
Gómez, Pablo
Grimblatt, Victor
Hernandez Tabares, Lorenzo
Hidalgo, Roberto
Larosa, Facundo Santiago
Leiva, Lucas
Lipovetzky, José
Lutenberg, Ariel
Martos, Pedro
Medel, Ricardo
Melo, Rodrigo Alejandro
Mercado, Gustavo
Miguez, Matías
Monte, Gustavo
Montilla Cabrera, Juan Vicente
Ovilla, Brisbane
Pantelides, Carlos
Pazos, Sebastián Matías
Perez Paina, Gonzalo
Pérez, Martín
Pérez, Santiago
Permingeat, Alejandro
Petrashin, Pablo Antonio
Pucheta, Julián
Puga, Gerardo Ludovico
Reta, Juan Manuel
Rey Vega, Leonardo
Sambuco Salomone, Lucas
Sanca, Gabriel Andrés
Sisterna, Cristian
Sutter, Gustavo
Taffernaberry, Carlos
Tropea, Salvador
Verrastro, Claudio
Zabaleta, Omar Gustavo
Zacchigna, Federico G.
Zaradnik, Ignacio
Zecchin, Danilo

ÍNDICE

ARQUITECTURA DE PROCESADORES Y DSP	PAG
Análisis del estado del arte en la depuración de sistemas embebidos	3
An approach to a condition monitoring system for electric motors maintenance	8
Sistema de Adquisición y Reconstrucción de Señales con Xampling y Sensado Compresivo Caótico	14
Implementación Multi-Core en la Computadora Industrial Abierta Argentina – NXP®	20
BIOINGENIERIA Y ROBÓTICA	23
Design and development of a steady-state auditory evoked potential meter (SSEP)	25
Diseño e implementación de Sistema Embebido para Reconocimiento de Palabras en Español	31
Control Adaptivo de un Sistema de Comunicación Inalámbrico	37
Desarrollo de software de comunicación comandado por gestos utilizando una interfaz cerebro-computadora	44
Un Nuevo Enfoque para Experimentos de Lazo Cerrado en Neurociencias Basado en Hardware Abierto y Software Libre	46
Funciones Potenciales para Evasión de Obstáculos	48
Migración a la CIAA del sistema de control de espejos para un concentrador solar tipo Fresnel	50
IMPLEMENTACIÓN DE SISTEMAS EMBEBIDOS	53
Estimador de F0 en tiempo real basado en el algoritmo de YIN	55
Unidad de adquisición de sensores con aprovechamiento de recursos de hardware	61
Antenna Motion Control System with Ethernet Connectivity	67
Sensor de Intensidad y Orientación de Viento Basado en Acelerómetro	72
Nuevo sistema embebido inalámbrico para medición de efectos de radiaciones ionizantes Aplicación a un caso de estudio y perspectivas futuras	78
Osciloscopio y Analizador Lógico para Dispositivos Móviles	84
Avances en migración a 32 bits de una placa controladora orientada a mediciones industriales - Dimensionamiento, alternativas y compatibilidad requerida	90
Sistema de visualización de precios para supermercados	96
FPGAs, HDLs y ASICs	99
Unidad de control de actuadores y comunicaciones basada en FPGA	101
Repositorios abiertos para desarrollo con FPGAs	107
Desarrollo e Implementación de Generador y Codificador NRZ-L de tramas PCM utilizando System Generator de Xilinx	112
Bit-Synchronizer Implementation in a Low-Cost FPGA for UAV	114
Bloque generador de señales PWM basado en FPGA con utilización de Wishbone	116
RTOS y SOFTWARE EMBEBIDO	119
Port del Firmware CIAA para plataformas basadas en FPGA con softcore LEON 3	121
Aplicaciones didácticas de la EDU CIAA, Octave y GUI Editor	127
Sistema de Adquisición y Procesamiento de Imágenes Térmicas de Bajo Costo	131
Determinación de la actitud angular de una sonda suborbital mediante cámara digital y sistemas embebidos	137
Sintetizador Digital de Audio con Filtro, Efectos y Comunicación por USB bajo LPCXpresso y FreeRTOS	142
Reconocimiento Embebido de Habla Aislada Independiente del Hablante en Tiempo Real con HMMs y SVMs	144
Implementación de síntesis de voz en la plataforma Intel Galileo	146
Medidor Bidireccional Inteligente Para Redes Electricas Interconectadas	148
Lazo de Enganche de Fase para Convertidores Fotovoltaicos Integrados a Redes Eléctricas	150
PROTOCOLOS, COMUNICACIONES Y REDES INALÁMBRICAS	153
FPGA implementation of a low complexity decoder for LDPC codes over impulsive noise channels	155
Sistema de Registro Automático de Información para el Análisis y Gestión del Tránsito Vial	160
Development of an experimental telemetry system	165
Aplicaciones de Software para Visualización y Monitoreo aplicables a Vehículos No Tripulados y Cohetes Sonda	171
Ataque a implementación embebida de AES-128 aplicando Análisis de Correlación de Potencia	176

Foro Tecnológico

Resumen

**Arquitecturas
de
Procesadores
y
DSPs**

Análisis del estado del arte en la depuración de sistemas embebidos

M. Giura, M. Prieto, N. González, L. Sugezky, M. Trujillo, J. Cruz
Departamento de Ingeniería Electrónica – Facultad Regional Buenos Aires
Universidad Tecnológica Nacional
Argentina
mgiura@frba.utn.edu.ar

Abstract— El proceso de depuración en sistemas embebidos permite detectar errores en la lógica de la programación. En los últimos años han surgido técnicas para sistematizar este proceso. En particular, el uso de modelos en sistemas embebidos se ha vuelto cada vez más frecuente y necesario para describir su comportamiento, por lo que contar con herramientas que permitan realizar la depuración del sistema resulta imprescindible. En el marco del proyecto de investigación “Desarrollo de software de simulación para la integración con uModelFactory” (EIUTNBA0002436) se lleva adelante el presente trabajo orientado a relevar y caracterizar las herramientas y técnicas existentes para la depuración de sistemas embebidos con el objetivo de implementarlas en el software uModel Factory.

Keywords—UML; sistemas embebidos; modelo ; depuración

I. INTRODUCCIÓN

El uso de modelos se ha vuelto cada más frecuente ya que permiten describir el software de los sistemas embebidos, ayudan a comprender el sistema y a diseñar con un nivel de abstracción superior al de los lenguajes de programación.

Un modelo es una representación simplificada de un sistema que contempla las propiedades importantes del mismo desde un determinado punto de vista. Además de servir para lograr un conocimiento más profundo del problema, favorecen el intercambio de ideas entre las personas involucradas en el diseño.

La mayoría de los enfoques actuales en el desarrollo de software basado en modelos coinciden en [1,2]:

- Utilizar una representación gráfica del sistema a desarrollar
- Describir el sistema con un cierto grado de abstracción.
- Generar código ejecutable para el sistema embebido partiendo del propio modelo

Como representaciones de dichos modelos se destacan las máquinas de estados finitos (FSM por Finite State Machine) las cuales constituyen una herramienta gráfica que ha sido

utilizada tradicionalmente para modelar el comportamiento de sistemas electrónicos e informáticos. Como una amplia extensión al formalismo convencional de estas máquinas surgieron los diagramas de estado (Statecharts) los cuales se convirtieron en parte del estándar UML (Unified Modeling Language) para describir el comportamiento de sistemas o de modelos abstractos.

Los diagramas de estado muestran el conjunto de estados por los cuales pasa un objeto durante su vida en una aplicación [3]. Para el pasaje de este objeto por los estados del modelo se analiza su respuesta a eventos y se vincula con sus respuestas y acciones. Estos diagramas normalmente contienen estados, transiciones, eventos, acciones y actividades. Un evento es una ocurrencia que puede causar la transición de un estado a otro del sistema. Esta ocurrencia se puede deber a distintas condiciones como puede ser: una condición que toma el valor de verdadero o falso (expresión booleana); la recepción de una señal o mensaje externo; o el paso de cierto período de tiempo. Una acción es una operación atómica, que no se puede interrumpir por un evento y que se ejecuta hasta su finalización.

En la actualidad existen diferentes enfoques orientados a la depuración de sistemas embebidos, principalmente pueden clasificarse en intrusivos o no intrusivos. A su vez, podemos evaluar los mismos como de tiempo real o de tiempo diferido (Figura 1). En el presente trabajo se analizan diferentes propuestas y herramientas orientadas al proceso de depuración.

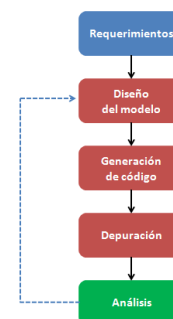


Figura 1. Enfoques utilizados en la actualidad

II. TÉCNICAS ACTUALES

Evaluación y análisis de los datos provenientes del proceso de depuración.

Bhagyalakshmi et al [4] centran su trabajo en el rastreo y la gestión de rastreo (tracing and trace management), donde abordan los problemas actuales del proceso de depuración mediante la recopilación de información sobre la ejecución del programa embebido.

En particular, focalizan su trabajo en cuatro cuestiones que deben ser gestionadas después de realizado proceso de rastreo, a saber:

- La heterogeneidad de los formatos de rastreo.
- El almacenamiento de grandes cantidades de datos de rastreo.
- La gestión del análisis del rastreo.
- La visualización de la información.

Partiendo de las ventajas y desventajas de las técnicas actuales de rastreo (datos “crudos” almacenados en archivos y archivos formateados con convenientes estructuras de datos), realizan una propuesta con un nuevo enfoque: utilizar una infraestructura de gestión de trazas para los sistemas embebidos que aborde los problemas de las cuatro cuestiones ya señaladas.

Para la heterogeneidad de formatos se propone un modelo de datos genérico que represente no solo los datos crudos, sino también la meta información relacionada con el rastreo y el análisis de resultados. Con este modelo propuesto, se le da soporte a trazas “multicore” en ventanas separadas y además, se soportan accesos random o procesamientos básicos como un filtrado, se facilita el registro personalizado de mensajes de rastreo para la GUI y se provee una separación entre la ventana de comandos de entrada y la ventana de salida, lo cual facilita el trabajo del usuario con la herramienta.

El sistema propuesto está basado en una arquitectura cliente-servidor y módulos asociados para la realización de la tarea que le dan soporte a las cuestiones señaladas anteriormente.

El proceso de depuración en tiempo real. Técnicas intrusivas y no intrusivas.

Dixon et al [5] relatan los contrastes entre la técnica tradicional de depuración denominada Run-control Debug y la más actual Real-time trace, recalando las claras ventajas de ésta última.

Para ello, señalan los problemas conocidos de la técnica tradicional intrusiva:

- No detecta problemas a máxima velocidad de clock
- No se posee la capacidad de detectar problemas relacionados con eventos externos en tiempo real

En contraste, la técnica de trazado en tiempo real no intrusiva, cada vez con más soporte en los nuevos procesadores, cuenta con las siguientes ventajas:

- No es necesario detener la aplicación a depurar.
- Se cuenta con la facilidad de capturar y almacenar detalles de la temporización (timing) en tiempo real de ejecución.
- Son capaces de capturar y almacenar eventos específicos de interés, descartando el resto, pudiendo observar hacia adelante o hacia atrás, en tiempo real, desde ese punto de interés (trigger condition), qué es lo que está pasando.

La traza captura el juego de instrucciones ejecutadas por el procesador en tiempo real y su temporización (alrededor de la condición de disparo o de todo el programa si no se selecciona una) y lo guarda en un buffer para ser analizadas con posterioridad. También puede capturar los datos utilizados por aquellas instrucciones. La información de temporización permite investigar “timing bugs” que de otro modo serían imposibles.

O’Keefe et al [6] retoman la técnica conocida como “Real-Time Trace” y la contrastan con técnicas tradicionales.

En primer lugar los autores se refieren a la utilización de la función “printf” como método básico de depuración, remarcando que no es aplicable en casos donde el código deba ser ejecutado en tiempo real, debido a que la impresión o transmisión de mensajes altera los tiempos de ejecución.

En segundo lugar, describen el método de depuración a través de una interfaz JTAG, también llamado “Run-Time Debug”, que permite ejecutar el programa paso a paso, insertar breakpoints, visualizar y modificar variables y registros, etc. En este caso, destacan como su principal inconveniente que para poder analizar el estado del programa es necesario pausar la ejecución del mismo, y esto cambia por completo el comportamiento en tiempo real del sistema, incluso pudiendo enmascarar por completo las fallas.

Estas desventajas de las técnicas tradicionales dan lugar a éste método alternativo, donde el sistema embebido cuenta con un módulo de “Real-Time Trace” (RTT) dentro del chip, recopilando información sobre la actividad del procesador. Se pueden registrar secuencias de valores del contador de programa (PC), lecturas o escrituras de memoria con sus valores asociados, lectura o escritura de registros, etc.

Todos esos datos combinados, brindan un panorama completo de la ejecución, para conocer exactamente qué fue lo que sucedió en el sistema embebido hasta que se produjo la falla. Importando esta información en una herramienta de análisis, se puede descubrir rápidamente el origen del problema.

Dentro de las ventajas claves, mencionan que este método permite ver cómo y por qué se arriba a cierto punto a través del historial de instrucciones, capturando los datos de manera no intrusiva, es decir, sin afectar el comportamiento en tiempo real de la aplicación. Además destacan que el “trace” de

instrucciones permite el “replay” de ejecución en modo offline y también determinar que porciones de código se ejecutan la mayor cantidad de tiempo.

Como consideraciones a tener en cuenta, remarcan que el módulo de RTT ocupa espacio dentro del chip, consume energía y en general necesita de un módulo de comunicación externo para poder volcar la información registrada. Dado que los casos de uso pueden ser muy diversos, los módulos de RTT son configurables dependiendo de la diversidad de datos que se deban registrar. La enorme cantidad de información generada, debe ser acotarla utilizando filtros que permiten definir condiciones puntuales y regiones de interés dentro del código.

Los autores concluyen que se trata de una herramienta muy útil que permite facilitar la localización de los “bugs” más complejos y así reducir los tiempos de desarrollo, en particular en etapas críticas de un proyecto.

Gracioli et al [7] analizan diferentes técnicas para el registro de datos provenientes del proceso de depuración, haciendo hincapié en la optimización de la cantidad de datos a almacenar. En particular el análisis hace foco en el registro de las interrupciones. También se señala la importancia de poder realizar en tiempo diferido el proceso de depuración.

El proceso de depuración bajo un sistema operativo embebido

J. Kraft et al [8] presentan el análisis de técnicas de depuración para sistemas embebidos basados en un sistema operativo. SO. Sin embargo, el análisis es trasladable a un sistema embebido que no cuente con un sistema operativo dado que se basa “que” hay que registrar, “cuando” y “por qué” a pesar de no poseer un administrador de tareas (scheduler) que cambie de tareas, se puede evaluar tanto como para el caso de máquinas de estado corriendo paralelo como en el caso de cambio de estados.

Su primera propuesta consiste en registrar una marca de tiempo (time-stamp) de los eventos elegidos a almacenar y que en principio estos eventos solo sean el cambio de tareas, pudiendo también tomar como posibles eventos a almacenar las IPCs y System Calls llamadas por las tareas.

En el resto del trabajo se analiza el “que” registrar, “cuando” hacerlo y “por qué”:

- Identificador de la tarea (el “que”).
Dado que el análisis está hecho para sistemas operativos proponen registrar el PID de la tarea y específicamente el STID (short task ID) para minimizar memoria de almacenamiento.
- Time-stamping (el “cuando”).

Como se dijo anteriormente se propone en el cambio de tarea y/o en el evento elegido registrar cuando ocurrió (time-stamp) y con los fines de ahorrar memoria de almacenamiento se propone (ya sea por medio del RTC si se dispone o por medio del clock en caso que sea constante) así registrar la diferencia entre un evento y el otro. Eligiendo la cantidad de bits que almacenen las temporizaciones más frecuentes y utilizar bits de extensión para el caso que el tiempo del evento sobrepase los bits elegidos.

- Task-switch (el “por qué”).
Almacenando la información del cambio de tareas se puede identificar cuando una tarea se bloquea o la misma se suspende o se termina. También se puede ver que tareas son las que tienen mayor prioridad.

Finalmente proponen una interfaz donde se pueden gráficamente los datos almacenados e identificar posibles errores en la ejecución de las tareas.

M. Desnoyers et al [9] destacan el uso del framework para rastreo implementado para Linux. En particular, presentan el uso de dicha herramienta, haciendo uso de la misma en la depuración de sistemas embebidos. Como parte del registro, señalan las modificaciones necesarias que implementaron para hacer uso del mismo en función de la detección de interrupciones.

La depuración en embebidos utilizando herramientas de bajo nivel

R. Mijat [10] presenta las limitaciones del proceso de depuración por software del sistema en general y presenta una herramienta de ARM y sus mejoras para dicho proceso en la depuración de las unidades del SoC completo.

Si entendemos que las aplicaciones de software para seguimiento y depuración son de corto alcance y que solo pueden dar una visión global de lo que está pasando en el sistema en lugar de una visión detallada del sistema ya que los sistemas modernos están constituidos por múltiples unidades de procesamiento, se necesita contar con una visión más detallada de cada una de ellas. Es por ello, que se presenta el CoreSight el cual es una tecnología propuesta por ARM la cual proporciona funcionalidades de seguimiento y depuración con el objetivo de tener información del SoC entero. CoreSight es un conjunto de componentes de hardware que pueden ser elegidos e implementados apropiadamente por el diseñador del sistema para extender la depuración. Para el uso de esta herramienta a cada unidad de procesamiento (DSP, CPU, acelerador gráfico, etc) se le asocia un bloque IP con una instrucción (ETM) o programa (PTM), de esta manera se le da seguimiento a la información de las instrucciones ejecutadas o al flujo de un programa.

En orden de mejorar las capacidades de depuración de la arquitectura CoreSight se agregaron dos nuevos componentes: System Trace Macrocell (STM) y el Trace Memory Controller (TMC). El STM está diseñado para el seguimiento de las actividades del sistema como ser la velocidad del bus APB, identificar cuando la FIFO está llena, entre otros eventos. El TMC proporciona información sobre los eventos de memoria producidos.

III. SOLUCIONES ACTUALES

Embedded UML Target Debugger [11]

La propuesta de esta solución consiste en una técnica compuesta por un módulo para depuración del lado de la interfaz más un módulo de monitoreo del lado del microprocesador. El módulo de monitoreo está incluido en el sistema operativo de tiempo real (RTOS) que se ejecuta en el microprocesador (Figura 2). El módulo de depuración que se encuentra del lado de la interfaz gráfica recibe los datos de seguimiento recopilados por el módulo de monitoreo en tiempo real y reconstruye e interpreta dicha información por medio del uso de diagramas UML como son diagramas de tiempo o secuencia.

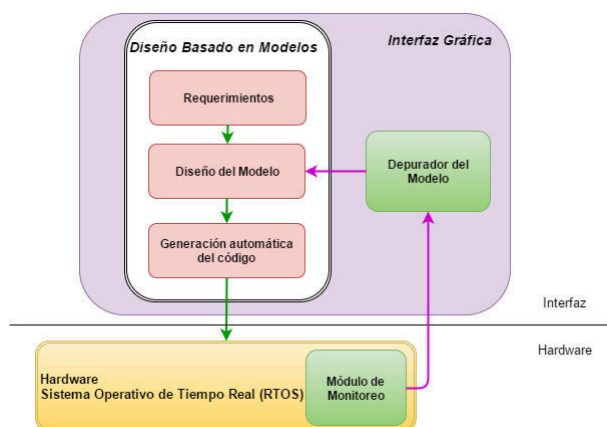


Figura 2. Diagrama de UML Target Debugger

Plantea la ventaja del desarrollo impulsado por modelos ya que el modelo puede ser probado y depurado en las primeras instancias del desarrollo utilizando la simulación. En base al trabajo sobre estos modelos se sabe que este método ayuda a identificar numerosos errores en las primeras fases del proceso. Por ello es necesario poseer en el momento de ejecución en el microprocesador una herramienta de depuración que permita la corrección de esos errores.

Otra desventaja de la simulación es la necesidad de simular todas las interfaces externas que el sistema necesita, por lo cual se puede volver una tarea muy compleja. Con el UML Target Debugger el modelo se puede ejecutar en el hardware real y poseer conexión con las interfaces externas reales. A su vez, dado que toda la información que el sistema de monitoreo obtiene del sistema embebido es proporcionado en tiempo real

es posible observar en la interfaz gráfica una animación en paralelo que el modelo es ejecutado en el hardware real. Finalmente la información obtenida en la depuración y evaluación hecha en la interfaz puede ser automatizada en el microprocesador por ejemplo con TestConductor [12] el cual permite automatizar pruebas a partir de información especificada por la solución descripta.

LieberLieber Embedded Engineer for Enterprise Architect [13]

Esta solución propone dentro de un mismo software tanto la generación del código en ANSI-C o C++ a partir de las estructuras UML, las máquinas de estados y las actividades del modelo como un módulo de depuración del modelo (UML).

LieberLieber Embedded Engineer por Enterprise Architect se diseñó como parte de un conjunto de herramientas creadas para el desarrollo de sistemas embebidos. Este desarrollo por medio del módulo de depuración UML ha resuelto un problema que venía influyendo a las herramientas de generación de código ya que el desarrollo y la depuración se daban en lugares diferentes, mientras el desarrollo se realizaba con el modelo UML la depuración se llevaba a cabo en el microprocesador en C o C++ sin conexión. Con la herramienta de depuración que contiene este software se permite la conexión entre la depuración del modelo desarrollado y el código generado que se ejecuta en el hardware.

El módulo de depuración UML permite inmediatamente después de transferir el código generado al hardware depurar el código directamente en el modelo. El depurador gestiona automáticamente la sincronización del modelo con el código fuente y permite al desarrollador ejecutar y seguir el progreso en dos niveles simultáneamente.

La ventaja de esta herramienta se encuentra en la posibilidad de contar en un mismo software tanto con la herramienta de diseño como la herramienta de depuración. A su vez, que permite realizar esta depuración a nivel del modelo sincronizado con el hardware.

IV. CONCLUSIONES

En el presente trabajo se evaluaron las propuestas realizadas por diferentes autores como así también herramientas comerciales y bajo licencia open-source, tanto para sistemas embebidos que utilizan un sistema operativo como en el caso "bare metal".

Dicha información será utilizada la implementación de herramientas de depuración para el software uModel Factory, orientado al diseño y simulación de sistemas embebidos utilizando diagramas de estado.

V. REFERENCIAS

- [1] G. Booch, J. Rumbaugh, I. Jacobson. "El Lenguaje Unificado de Modelado". Addison-Wesley 2nd Edition, 2006.
- [2] G. Booch, J. Rumbaugh, I. Jacobson. "El Proceso Unificado de Desarrollo de Software". Addison-Wesley 1st Edition, 2000.
- [3] C. Larman. "UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado". Prentice-Hall, 2003.
- [4] C. Bhagyalakshmi, P. Tembad. "Trace Management, Debug, Analysis and Testing Tool for Embedded Systems". Research and Reviews. International Journal of Innovative Research in Computer and Communication Engineering. ISSN ONLINE(2320-9801). 2015.
- [5] B. Dixon, O'Keeffe. "The Advantages of Real Time Tracedebug in Complex Embedded Systems". Ashling Microsystems. 2013.
- [6] J. Campbell, V. Kazantsev, H. O'Keeffe. "Real-time Trace: A Better Way to Debug Embedded Applications". White paper. Synopsys Inc. 2014.
- [7] G. Gracioli, S. Fischmeister. "Tracing Interrupts in Embedded Software". Journal of Systems Architecture. Volume 58, Issue 9, October 2012.
- [8] Kraft J., Wall A., Kienle H. "Trace Recording for Embedded Systems: Lessons Learned from Five Industrial Projects". Lecture Notes in Computer Science, vol 6418. Springer, Berlin, Heidelberg. 2010.
- [9] M. Desnoyers, M. Dagenais. "Low Disturbance Embedded System Tracing with Linux Trace Toolkit Next Generation". Embedded Linux Conference. 2006
- [10] R. Mijat. Better Trace for Better Software. ARM White Paper. 2010. Disponible en https://www.arm.com/files/pdf/Better_Trace_for_Better_Software_-_CoreSight_STM_with_LTTng_-_19th_October_2010.pdf
- [11] Embedded UML Target Debugger
Disponible en: <http://www.willert.de/assets/Datenblaetter/DatS-Embedded-UML-Target-Debugger-V9.0-EN-2016.pdf>
- [12] IBM Rational Rhapsody TestConductor Add On
Disponible en: https://www.ibm.com/support/knowledgecenter/SSB2MU_8.1.4/com.btc.tcatg.user.doc/pdf/RTC_User_Guide.pdf
- [13] LieberLieber Embedded Engineer for Enterprise Architect
Disponible en: <http://www.lieberlieber.com/en/embedded-engineer-for-enterprise-architect/>

An approach to a condition monitoring system for electric motors maintenance

Fernando Gabriel Orge

Comisión Nacional de Energía Atómica - Centro Atómico Constituyentes
Universidad Tecnológica Nacional - Facultad Regional Buenos Aires
fernandoorge@cnea.gov.ar

Abstract—This work proposes a general methodology for the vibration analysis of electrical motors by measurements made on non-rotating parts. A data acquisition system along with a set of digital signal processing techniques are used not only to assess the vibration severity of the observed motor but to diagnose incipient faults within it. The time-domain techniques are used to establish the vibration severity according to widely accepted industry standards, while the frequency-domain techniques are used to obtain the vibration signature of the motor. An experimental setup has been used and all these developments have been successfully tested.

Keywords—Condition monitoring, Vibration analysis, Digital signal processing

I. INTRODUCTION

The benefits of applying a condition monitoring program in a nuclear facility, including the reduction of maintenance time and maintenance costs have been proved [1]. In addition to this, a condition monitoring program allows getting a better understanding of the degradation mechanisms of the monitored component [2] [3]. Thus, the use of adequate non-destructive and non-invasive tests along with an accurate measurement analysis of the observed component will serve as an effective way to mitigate its ageing effects.

The focus of this work is put on the electrical motors that could be present in the primary and/or secondary coolant system of an experimental reactor, because they are responsible for the water circulation through the reactor piping, hence for its correct refrigeration.

According to [4] and [5], vibration measurements on non-rotating parts of a motor is an effective way to guarantee its safe and long-term operation. It also allows knowing how the vibration affects other elements in contact with it, such as pipes, pumps and other systems. Therefore, the vibrations of the motors are adopted as the basis for the diagnosis and prognosis of the reactor coolant systems. Consequently, the periodical observation of the vibration state of these electrical motors serves as a valuable assistance in any decision-making process where a preventive task should be performed in case an abnormal condition is observed.

II. MECHANICAL VIBRATIONS

A. Vibration severity

The vibration severity is the result of the vibration measurement process and is defined as the maximum magnitude

of vibration measured under steady-state operating conditions [4]. It is a common practice to consider the root-mean-square value of the vibration velocity since it can be related to the vibration energy.¹

Once the vibration severity is obtained, it must be assessed against four evaluation zones established on [4]. These zones may be summed up in:

1) *Zone A*: Within this zone, the motor is considered to be in optimal conditions.

2) *Zone B*: Within this zone, the motor is considered acceptable for long-term operation.

3) *Zone C*: Within this zone, the motor is considered unsatisfactory for long-term operation.

4) *Zone D*: Within this zone, the motor is considered the operating conditions are considered to be of sufficient severity to cause damage to the motor.

The values assigned to the zone boundaries depend on the motor's rated power, its velocity and its support assembly flexibility.

B. Support assembly flexibility

The support conditions are determined by the relationship between the motor and the foundation flexibilities. According to [5], the combined motor and support system should be classified as rigid or flexible.

The support assembly flexibility is related to the lowest natural frequency of the motor-support system. In case that the motor-support system cannot be classified by calculation, it may be classified by test (e.g. in a power trip test).

C. Vibration signature and vibration spectrum

The use of the vibration severity as a sole feature may not be suitable for root-cause analysis. For that reason, when the vibration severity departs from the expected value a frequency-domain analysis is mandatory.

The vibration signature is an intrinsic feature of the motor under test and serves as a measure of its health. The vibration spectrum is derived from the vibration signal acquired during the tests and should be compared to the vibration signature to assess the most likely causes of failure.

¹For a zero-mean stochastic process its variance is proportional to its energy, and since its standard deviation is equal to its RMS value, it follows that the RMS value is related to the energy of the process.

III. THE DATA ACQUISITION SYSTEM

A low-cost data acquisition system was designed and developed at the Argentinian Life Management Department of the Atomic Energy National Commission². The system is capable of acquiring the vibration signal and communicating with any personal computer (PC) for its analysis in a scientific programming environment.

Figure 1 shows a block diagram of the developed system and its main components. A piezoelectric accelerometer calibrated with an electrodynamic exciter is used as the mechanical transducer. The acceleration signal is then filtered by a sixth-order low-pass filter for its later digitalization. The analog-to-digital conversion is done by means of a 12-bit embedded SAR ADC with 11.3 ENOB. The digital signal controller (DSC) contains extensive digital signal processing functionality with a high-performance 16-bit microcontroller architecture. The DSC is responsible for the time domain analysis and the management of the communication interfaces, while the PC performs the frequency analysis.

A. Time domain analysis

Since velocity needs to be considered to determine the vibration severity of a motor, the acceleration signal integration is mandatory. Figure 2 shows how the acceleration signal is processed to obtain the root-mean-square (RMS) value of the velocity signal. Each step is described below.

1) *Scaling*: This stage is necessary for units adjustments.

2) *DC-Block Filtering*: A single order filter (high-pass filter) is used to decouple the acceleration signal.

3) *Numerical Integration*: The trapezoidal method is used to obtain the velocity signal.

4) *Band-Pass Filtering*: As stated in [5] and [6], the velocity signal should be analysed in a fixed bandwidth (i.e. 10 Hz to 1000 Hz). Thus, a fourteenth order IIR filter, with 0.1 dB ripple in the band pass, is used.

5) *RMS Value Calculation*: The computation of the RMS value of the velocity signal leads to the assessment of the vibration severity.

B. Frequency domain techniques: Welch's Method

Considering that both industrial and nuclear facilities are noisy environments, Welch's method [7] is a suitable tool to estimate the power spectrum of a motor and to mitigate the additive noise corruption.

²Departamento de Gestión de Vida de Instalaciones Nucleares y Convencionales de la Comisión Nacional de Energía Atómica de la República Argentina.

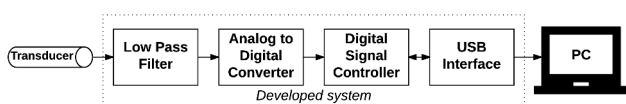


Figure 1. Main components of the designed and developed data acquisition system.

C. Feature extraction

After obtaining the power spectrum of the motor, a feature extraction for root-cause analysis is helpful. For this purpose two criteria are used.

1) *Q-Constant criterion*: The power spectrum is split into five q-constant regions to cover the measurement bandwidth. The q factor is the relationship between the centre frequency of a region and its bandwidth.

Each region defines an interval of integration over the power spectral density. Then, by calculating the square root of the obtained energy, an RMS velocity value can be assigned to each region.

2) *Fixed-bandwidth criterion*: Each q-constant region is divided into smaller bands with constant bandwidth. This bandwidth should be small enough to discriminate two near frequency components (e.g. In an induction motor, the rotational frequency is slightly less than the line frequency due to the motor's slip).

In the same way as before, an RMS velocity value can be assigned to each of these regions.

D. Time-Frequency domain techniques: Gabor's Transform

Most of the Discrete Fourier Transform Analysis have an inability when trying to localise temporal events. In this work, Gabor's Transform [8] is used to solve that limitation. Particularly is used in power trip tests to determine whether the motor-support system is rigid or not.

E. Normality test

As stated in [9], the normal distribution is the most common probability distribution encountered when studying noise and vibration signals. For that reason, when Gaussian noise is presumed in a test, data should be tested for normality.

As far as this work is concerned, a visual test is good enough to ensure that data has a normal distribution.

F. Stationarity test

Conforming to [4] and [5], vibration measurements should be made after achieving agreed operating conditions. In this work, the agreed conditions are the rated operating conditions, which can be related to the stationarity of a stochastic process.

This test consists of evaluating the statistical moments of the vibrational signal over the test duration. As far as none of them exhibits any tendency, it may be concluded that the process is stationary. Therefore the rated operating conditions are guaranteed.

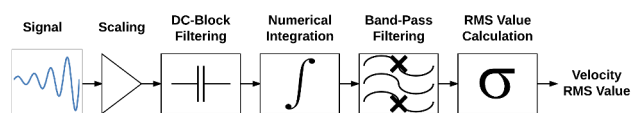


Figure 2. Time domain processing scheme to obtain the root mean square value of the velocity signal.

IV. A CASE STUDY

A. Experimental setup

In order to test the data acquisition system along with the analysis techniques, a three-phase squirrel-cage motor with rated power equal to 7.5 kW and rated speed of 2930 RPM was selected as the motor under test.

Figure 3 shows the arrangement of the elements during one of the tests. The piezoelectric accelerometer was mounted on the motor housing with the aid of a permanent magnet and a coaxial cable was used to connect the transducer to the data acquisition system.

B. Test procedure

Based on the guidelines of the previously mentioned standards, the test procedure presented in figure 4 was proposed to test the motor.

1) *Environmental evaluation*: The goal of this step was to evaluate the environmental contribution to the vibration measurement. A normality test was performed on the recorded signal while the motor was shut down.

2) *Motor-Support system classification*: A power trip test was used to characterise the system. This test consists of studying the dynamic behaviour of the motor during its stop by using the Gabor's Transform.

3) *Operating conditions evaluation*: The stationarity test was used to guarantee the rated operating conditions. The statistical moments, up to the fourth order, were evaluated during the test.

4) *Vibration severity evaluation*: Non-simultaneous measurements were made on the vertical and horizontal axes of the rotor's radial plane.

5) *Acceptance limits*: According to its rated power and speed and its type of support, the expected values for the motor under test were the ones shown in Table I. If the measured value departs from the expected, further analysis must be done to study the drift cause. Otherwise, the procedure finishes.

6) *Vibration spectrum estimation*: The vibration spectrum was obtained by applying Welch's Method to a 120-second-duration signal. With an overlap factor of 50% and a frequency resolution of 0.1 Hz, 23 power spectrums were averaged to obtain the estimated spectrum.

7) *Root cause analysis*: To further analyse the vibration spectrum both q-constant and fixed-bandwidth criteria were used.

Table I. VIBRATION SEVERITY EXPECTED VALUES FOR THE MOTOR UNDER TEST.

RMS Vel. [mm/s]	Evaluation Zone
Less than 1.12	Zone A
1.12 to 2.80	Zone B
2.80 to 7.10	Zone C
More than 7.10	Zone D

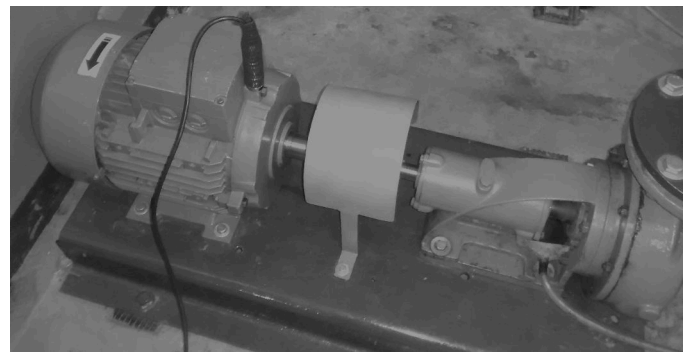


Figure 3. Measurement on the vertical axis of the radial plane of the motor under test.

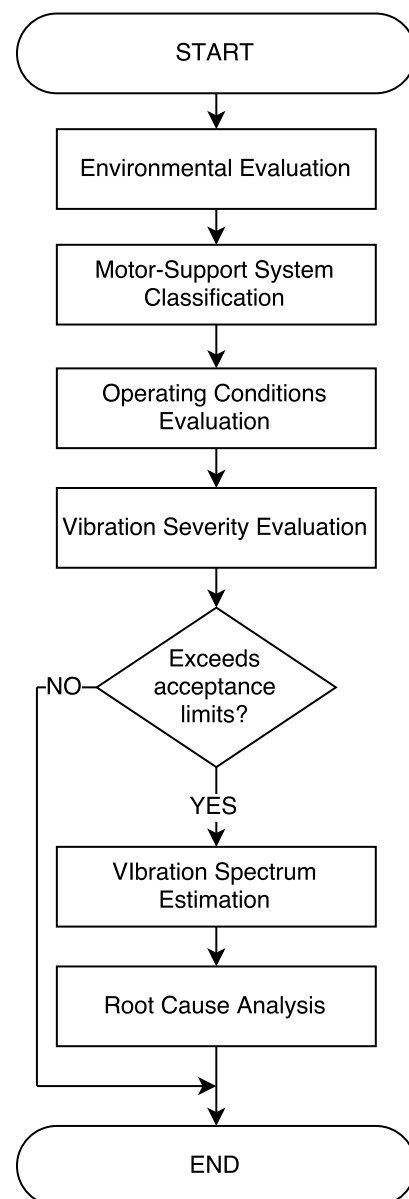


Figure 4. Step sequence of the test procedure.

C. Experimental results

The results of each procedure step are described below.

1) *Environmental evaluation*: Figure 5 shows the result of the normality test. Through this test, the random nature of the environmental noise was verified. A standard deviation of 0.61mm/s was observed.

2) *Motor-Support system classification*: Figure 6 shows the result of the Gabor's Transform used to study the stop transient during the power trip test. Both time and frequency axis are shown, as well as the magnitude of the frequency components in the colour scale.

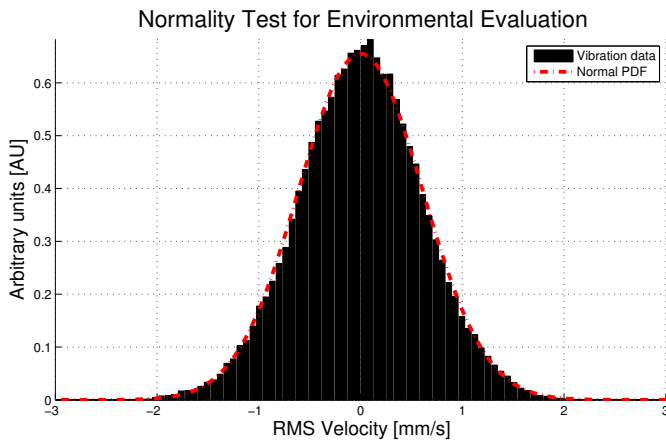


Figure 5. Normality Test for environmental evaluation.

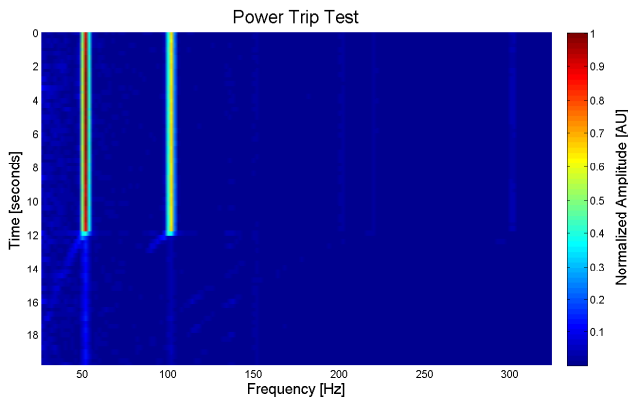


Figure 6. Gabor's Transform used to perform the power trip test.

Table II. DETAILS OF THE MEASUREMENTS.

Measurement Point	Radial Plane - Vertical Axis
RMS Velocity [mm/s]	(4.32 ± 0.63)
Level of Confidence	95.45%
Measurement Bandwidth [Hz]	10 to 1000
Vibration Severity	C

At the beginning of this test, the motor was turning at rated speed. At some point between 12 and 14 seconds, the power was turned off. Since then, it can be seen on figure 6 how the frequency content around 50 and 100 Hz decreases both in amplitude and frequency.

Considering that there is no evidence of resonance phenomena during the test, it may be concluded that the motor-support system is rigid.

The remaining content observed around 50 and 100 Hz could be attributed to a poor isolation of the motor housing and the permanent magnet used in the accelerometer mounting or to the extension of the coaxial cable.

3) *Operating conditions evaluation*: Figure 7 shows the result of the stationarity test for a 120-seconds-duration signal, where no tendencies were observed for any statistical moment.

4) *Vibration severity evaluation*: Table II and Table III show the measurements results. When comparing the maximum obtained value (4.32mm/s on the vertical axis) against the expected values shown in Table I, it concludes that the motor is within the evaluation zone C.

The uncertainty of the measurements was calculated in agreement with the *Guide to the Expression of Uncertainty in Measurement* [10].

5) *Acceptance limits*: The vibration severity exceeded the acceptance limits and therefore the motor was considered inadequate for long-term operation.

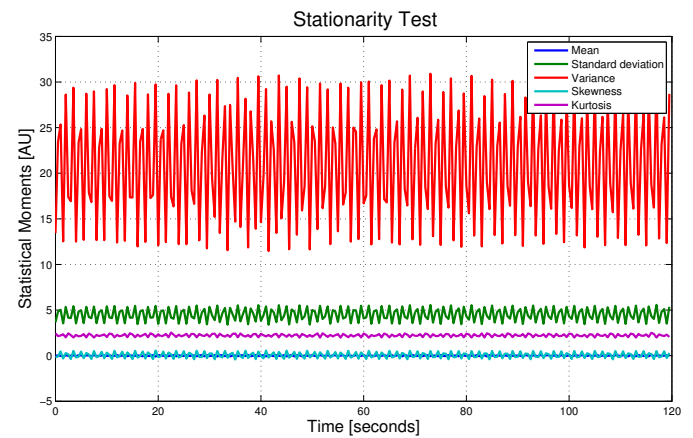
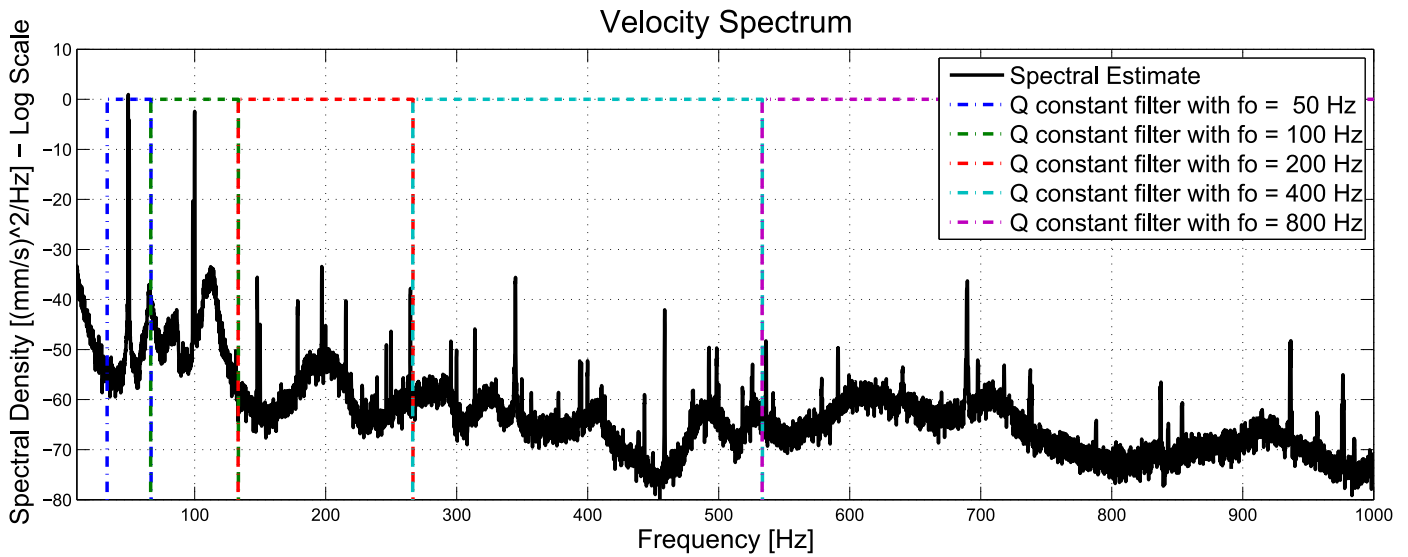


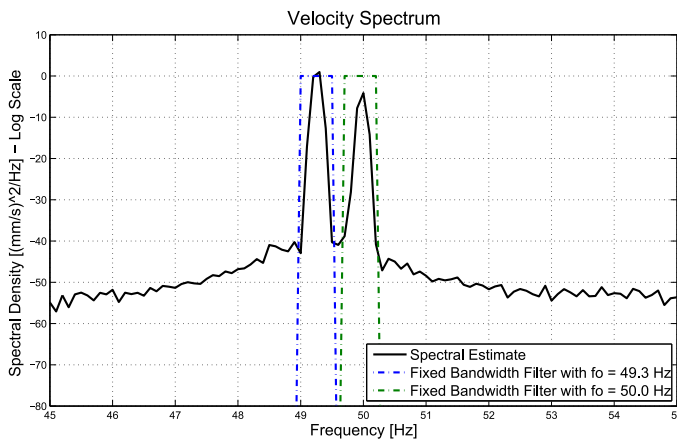
Figure 7. Stationarity test for operating conditions evaluation.

Table III. DETAILS OF THE MEASUREMENTS.

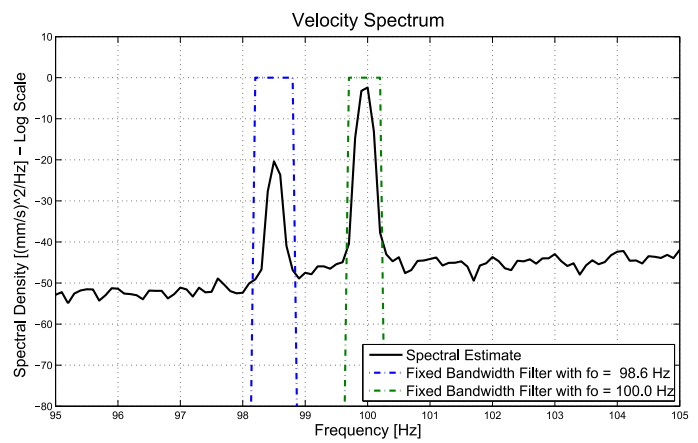
Measurement Point	Radial Plane - Horizontal Axis
RMS Velocity [mm/s]	(3.03 ± 0.57)
Level of Confidence	95.45%
Measurement Bandwidth [Hz]	10 to 1000
Vibration Severity	C



(a) Velocity Spectrum of the motor under test measured at the vertical axis.



(b) Spectrum around motor's rated speed.



(c) Spectrum around the first harmonic of the motor's rated speed.

Figure 8. (a) shows the motor spectrum over the measurement bandwidth, while (b) and (c) shows the same spectrum focused on 50 and 100 Hz respectively.

Table IV. Q CONSTANT ENERGY ANALYSIS WITH Q FACTOR EQUALS TO 1.5.

fo [Hz]	50	100	200	400	800	Total
RMS Vel. [mm/s]	3.61	2.32	0.16	0.11	0.12	4.32
Percentage [%]	69.83	28.84	0.13	0.06	0.07	100.00

Table V. FIXED BANDWIDTH ENERGY ANALYSIS WITH BANDWIDTH EQUALS TO 0.5 HZ.

fo [Hz]	$1 \times RPM$	$1 \times f_L$	$2 \times RPM$	$2 \times f_L$	Total
RMS Vel. [mm/s]	3.21	1.63	0.26	2.27	4.32
Percentage [%]	55.21	14.23	0.36	27.61	100.00

6) *Vibration spectrum estimation*: Figure 8.a shows the spectral estimate obtained from the measurement on the vertical axis. It can also be seen how with the use of 5 constant Q filters, it is possible to cover the entire measurement bandwidth.

Figures 8.b and 8.c exhibit the same spectrum as Figure 8.a but focused on 50 and 100 Hz respectively. In (b), the

frequency component observed at 49.3 Hz is related to the rotating speed ($1 \times RPM$) while the frequency component at 50 Hz is related to the line frequency ($1 \times f_L$). (c) shows the harmonics of each component ($2 \times RPM$ and $2 \times f_L$).

7) *Root cause analysis*: Table IV shows the RMS velocity value obtained for each q-constant region. The *Percentage %* row indicates the contribution of each region to the total value.

From this first analysis, all the defects related to a frequency component in the upper regions were dismissed (e.g. pump and bearing faults). Then, the fixed bandwidth criterion was applied to the 50-Hz and 100-Hz regions.

Table V shows the RMS velocity value obtained for each fixed-bandwidth region. The Percentage row indicates the relative contribution of each region to the total value.

With respect to the 50-Hz band, Table V shows that the mechanical component ($1 \times RPM$) is 4 times greater (approximately) than the electrical one ($1 \times f_L$). Considering that these two components are of the same order of magnitude, the most probable cause of failure, according to [11], is known as dynamic eccentricity and occurs as a consequence of possible bending of the rotor shaft or by local rotor heating.

In the 100-Hz band, the electrical component ($2 \times f_L$) is much larger than the mechanical one ($2 \times RPM$). Thus, the most probable cause of failure, according to [11], can be attributed to a weakness of the stator support, to an unbalanced phase resistance or to shorted stator laminations.

V. CONCLUSION

Throughout this work, the development of a general methodology to assess the vibration state of any electric motor, according to ISO standard 10.816, has been shown.

A data acquisition system, as well as all the necessary signal analysis techniques, have been developed.

These developments were successfully tested on a three-phase squirrel cage motor with rated power equal to 7.5 kW and rated speed of 2.930 RPM. The motor under test was proved to be unsatisfactory for long-term operation.

FURTHER WORK

The data acquisition system will be upgraded to achieve greater robustness and multichannel signal acquisition.

The analysis techniques will be migrated to Python programming language for the development of a vibration analysis software.

Other analysis techniques, like Principal Component Analysis, will be considered in order to provide greater reliability in fault diagnosis. Artificial Neural Networks will also be considered for the automation of root-cause analysis.

REFERENCES

- [1] IAEA, "On-line monitoring for improving performance of nuclear power plants. Part 2, Process and component condition monitoring and diagnostics," International Atomic Energy Agency, Austria, Technical Report No. NP-T-1.2, 2008.
- [2] —, "Ageing Management for Nuclear Power Plants," International Atomic Energy Agency, Austria, Safety Guide No. NS-G-2.12, 2009.
- [3] —, "Ageing Management for Research Reactors," International Atomic Energy Agency, Austria, Safety Guide No. SSG-10, 2010.
- [4] ISO, "Mechanical Vibration – Evaluation of machine vibration by measurements on non-rotating parts – Part 1: General guidelines," International Organization for Standardization, Switzerland, Standard ISO 10816-1:1995(E), 1995.
- [5] —, "Mechanical Vibration – Evaluation of machine vibration by measurements on non-rotating parts – Part 3: Industrial machines with nominal power above 15 kW and nominal speeds between 120 r/min and 15 000 r/min when measured *in situ*," International Organization for Standardization, Switzerland, Standard ISO 10816-3:1998(E), 1998.
- [6] API, "Machinery protection systems," American Petroleum Institute, USA, Standard API Standard 670 - 4th ed., 2000.
- [7] P. D. Welch, "The use of fast fourier transforms for the estimation of power spectra: A method based on time averaging over short modified periodograms," *IEEE Transactions on Audio and Electroacoustics*, 1967.
- [8] D. Gabor, "Theory of communication, part 1," *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering*, vol. 93, 1946.
- [9] A. Brandt, *Noise and Vibration Analysis: Signal Analysis and Experimental Procedures*, 1st ed. UK: John Wiley and Sons, 2011.
- [10] J. C. for Guides in Metrology, "Jcgm 100: Evaluation of measurement data - guide to the expression of uncertainty in measurement," JCGM, Tech. Rep., 2008.
- [11] G. H. Bate, "Vibration diagnostics for industrial electric motor drives," Brüel and Kjaer, Application Note BO 0269, n.d.

Sistema de Adquisición y Reconstrucción de Señales con Xampling y Sensado Compresivo Caótico

Mariano L. Acosta, M. Antonelli y Luciana De Micco
Instituto de Investigaciones Científicas y Tecnológicas en Electrónica (ICYTE)
Facultad de Ingeniería, Universidad Nacional de Mar del Plata (FIUNMdP)
Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)
Buenos Aires, Argentina.

Resumen—En este trabajo se presenta el diseño y simulación de un sistema de adquisición de señales empleando dos técnicas con gran auge en la actualidad: la compresión de datos en el dominio analógico (denominado Xampling) y sensado compresivo (CS) el cual permite la adquisición y reconstrucción eficiente de señales a partir de un número de muestras menor al requerido por el teorema de muestreo de Shannon-Nyquist. Un punto crítico en esta técnica es la elección de la implementación del muestreo, usualmente se utilizan secuencias aleatorias. Estas secuencias son computacionalmente costosas de generar, por lo que en este trabajo se estudia la utilización de secuencias caóticas. Se proponen cuantificadores de performance de los sistemas modelados y se reportan los resultados obtenidos de los cuales se desprende que los sistemas con caos presentan un comportamiento comparable al clásico.

I. INTRODUCTION

La técnica de sensado compresivo es un nuevo paradigma de adquisición de señales basado en el principio de que diversos tipos de señales poseen una representación dispersa en una base conocida [1]. En lugar de adquirir la señal a una frecuencia que cumpla con el teorema de Nyquist y luego aplicar un método de compresión para su posterior transmisión o almacenamiento, CS propone la compresión de la información y el muestreo en una sola etapa, permitiendo usar frecuencias de muestreo menores a la frecuencia de Nyquist. Esto último implica dos ventajas: (1) la reducción de la cantidad de memoria necesaria para el almacenamiento de las muestras y (2) la posibilidad de digitalizar señales de una frecuencia mucho mayor que las permitidas tradicionalmente. Ejemplos de aplicaciones con CS incluyen deconvolución de imágenes en radioastronomía [2], reconstrucción ciega de señales multibandas [3] y procesamiento de imágenes de resonancia magnética en tiempo real [4]. Un punto crítico de la técnica es la elección de la matriz que implementa el muestreo, ésta debe presentar un comportamiento aleatorio y tradicionalmente es generada a partir de secuencias estocásticas, las cuales son difíciles de implementar en hardware.

El caos determinista demostró que modelos deterministas muy simples originan señales de aspecto estocástico [5], [6]. La sensibilidad a las condiciones iniciales hace que en estos sistemas la predictibilidad sea a corto plazo, lo que ubica a estos sistemas en una posición intermedia entre determinista y estocástico. Como consecuencia se desarrollaron en los últimos años un número creciente de aplicaciones de los sistemas caóticos, empleándolos principalmente como generadores de ruido controlado, generadores de números pseudo aleatorios,

portadoras o señales llave en sistemas de encriptado, etc. El objetivo es el reemplazo de las secuencias aleatorias (Gaussiana, Bernoulli, etc) ya que estas usualmente son computacionalmente costosas de generar.

Existen trabajos que abordan la temática de combinar secuencias caóticas con sistemas de CS, en [7] los autores desarrollan un sistema para la adquisición y reconstrucción de señales dispersas en frecuencia utilizando como muestreo el mapa caótico de Hennon. Las mediciones se obtienen submuestreando la salida del sistema caótico excitado por la señal de sensado. Linh et al. [8] proponen un filtro de estructura caótica, exploran el uso de procesos deterministas caóticos en el diseño de las etapas de los filtros. Concluyen que es posible reconstruir exactamente una señal original rala en el tiempo a partir de sus muestras, superando la performance de los filtros aleatorios. En [9] se propone un método simple de construcción de la matriz de muestreo con secuencias caóticas. También realiza comparaciones experimentales usando matrices tradicionales para mostrar que los rendimientos entre estas matrices son casi iguales.

En este trabajo se pretende estudiar el reemplazo de las secuencias estocásticas empleadas en el proceso de CS por secuencias caóticas combinando esta técnica con la de Xampling. Para esto se desarrolló un sistema completo, se realizaron simulaciones y se evaluaron los resultados en términos de performance. Este trabajo es una primera etapa para el desarrollo del sistema en hardware, el cual se realizará para validar experimentalmente los resultados aquí obtenidos.

El artículo esta organizado de la siguiente manera. En la Sección II se realiza una revisión de las técnicas Xampling y Sensado Compresivo. En la Sección III se comenta las secuencias caóticas empleadas en este trabajo. La Sección 3 se describen los cuantificadores empleados para evaluar la performance de los sistemas. Finalmente los resultados y conclusiones estan en las Secciones V y VI respectivamente.

II. XAMPLING Y SENSADO COMPRESIVO

Sea la señal $x = \Psi\alpha$ un vector N dimensional discreto tal que $x \in \mathbb{R}^N$ que puede ser representado en una base ortonormal $\varphi \in \mathbb{C}^{N \times N}$ mediante una combinación lineal con el vector de coeficientes $\alpha \in \mathbb{R}^N$. Se dice que x es dispersa en la base Ψ si presenta K términos en α no nulos y $K \ll N$. En la práctica, se consideran los K elementos mayores y se desprecian los demás considerando un umbral. CS establece que si x cumple con el enunciado anterior,

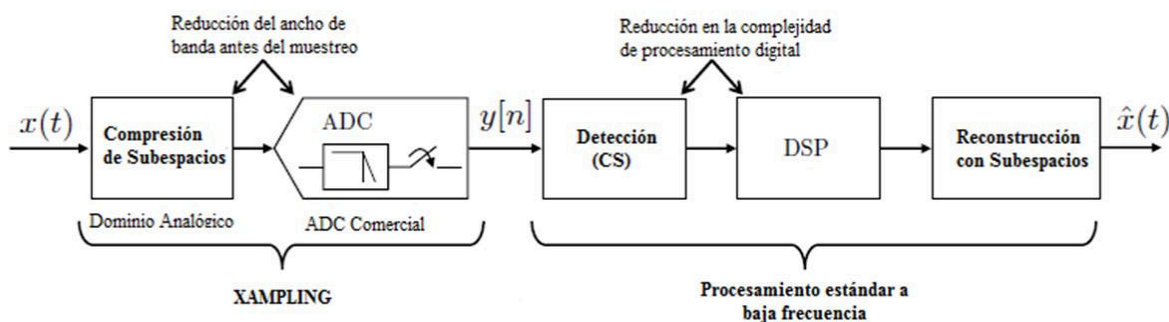


Figura 1. Cadena de procesamiento con Xampling y Sensado Compresivo.

entonces sólo es necesario recolectar $M = O(K \log(\frac{N}{K}))$ muestras de la señal proyectada en un espacio de sensado lineal obteniendo el vector $y = \varphi x$, $y \in \mathbb{R}^M$ y $\varphi \in \mathbb{R}^{M \times N}$ es el denominado Kernel de Compresión [10]. Reemplazando x por su representación en la base ortonormal se obtiene $y = \varphi \Psi \alpha = A \alpha$, definiendo la matriz $A \in \mathbb{C}^{M \times N}$. Entonces, una vez adquirida la señal comprimida y se debe recuperar x mediante la aproximación \hat{x} resolviendo el siguiente problema de optimización convexo:

$$\hat{\alpha} = \min \|\alpha\|_1 \quad (1)$$

con $\hat{\alpha} \in \mathbb{R}^N$ sujeto a:

$$\|A\alpha - y\|_2 \leq l$$

Siendo $\|\cdot\|_1$ y $\|\cdot\|_2$ las p-normas para $p = 1$ y $p = 2$ respectivamente, $\hat{\alpha}$ la aproximación vector de coeficientes α y l es el nivel de tolerancia [11]. Una vez hallado el vector de $\hat{\alpha}$ con 1 se procede a obtener la reconstrucción $\hat{x} = \varphi \hat{\alpha}$. Según [12], la reconstrucción está asegurada si A cumple con la propiedad Isométrica Restringida (RIP) y φ presenta un alto grado de incoherencia con Ψ .

Para poder realizar la operación de compresión lineal provista por la matriz A en el dominio analógico en vez del digital se recurre al paradigma de Xampling (presentado por primera vez en [13]) que modela a x como la unión de subespacios de Hilbert. Xampling consiste básicamente en la conversión analógica digital (X-ADC) mediante submuestreo a una tasa constante. La señal de entrada es comprimida en el dominio analógico previo a la conversión con dispositivos comerciales, realizando un efecto equivalente a la matriz Ψ . Luego, los subespacios que constituyen a la señal son detectados digitalmente en la etapa X-DSP con las muestras obtenidas. La parte X-DSP es realizada con CS, pero existen otras técnicas como los algoritmos ESPRIT y MUSIC [14].

En el caso particular de este trabajo, se optó como punto de partida la compresión por demodulación aleatoria [10]. El esquema del demodulador se muestra en la Fig. 2.

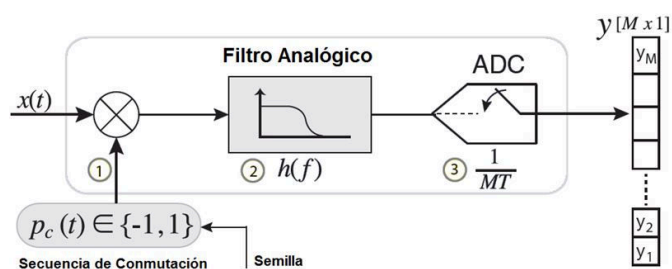


Figura 2. Demodulador aleatorio.

En este método de compresión de la información la señal de entrada $x(t)$ es multiplicada por una secuencia de pulsos pseudoaleatoria $P_c(t)$, cuya frecuencia debe ser por lo menos la frecuencia de Nyquist para reconstruir a $x(t)$. Luego de este proceso de dispersión espectral, se procede a filtrar el producto por un filtro pasabajos con respuesta al impulso $h(t)$ conocida. Finalmente, se muestrea el resultado a una frecuencia constante mucho menor que la requerida por Nyquist mediante un ADC comercial. Es importante que la frecuencia de corte del filtro sea menor a la mitad de la frecuencia de muestreo utilizada por el conversor.

Esta secuencia de pulsos pseudoaleatoria $P_c(t)$, que clásicamente se implementa con secuencias aleatorias, es la que se reemplazará por secuencias caóticas. Una representación simple de la señal digital $P_c(t)$ tiene la forma de un tren de pulsos modulados por la secuencia aleatoria c_i . Esta señal puede ser expresada como:

$$P_c(t) = \sum_i c_i p(t - iT_c) \quad (2)$$

donde T_c es el período del tren de pulsos y cumple con Nyquist.

III. MATRICES DE MUESTREO CAÓTICO

Como se dijo anteriormente en este trabajo se analizó el reemplazo de la matriz de muestreo Ψ que clásicamente se realiza mediante una secuencia aleatoria c_i por el de una secuencia caótica.

Cuadro I. ILLUSTRACIÓN DEL MÉTODO DE **Skipping**.

TWBM	TWBM2 (d=2)	TWBM3 (d=3)	TWBM4 (d=4)
0,010559404			
0,041791613	0,041791613		
0,160180296		0,160180296	
0,538090276	0,538090276		0,538090276
0,994196523			
0,023079185	0,023079185	0,023079185	
0,090186145			
0,328210418	0,328210418		0,328210418
0,881953358		0,881953358	
0,416446528	0,416446528		
0,972075269			
0,10857976	0,10857976	0,10857976	0,10857976
0,387160782			
0,949069244	0,949069244		
0,193347257		0,193347257	
0,62385638	0,62385638		0,62385638

Las secuencias caóticas estudiadas en este trabajo son las siguientes:

- LOG: Secuencia generada por el mapa caótico Logístico.
- TWBM: Secuencia generada por el mapa caótico Three-Way Bernoulli.
- FWTSM: Secuencia generada por el mapa caótico Four-Way Tailed Shift.
- TWBMd: Secuencia generada por la d-ésima iteración del mapa caótico TWBM.
- FWTSMd: Secuencia generada por la d-ésima iteración del mapa caótico FWTSM.

Las secuencias generadas por cada mapa se transforman en un código binario por medio de la regla usual de dinámica simbólica $([0, 0,5] \rightarrow -1, (0,5, 1] \rightarrow 1)$.

Como puede verse además de los mapas caóticos se consideraron versiones aleatorizadas de los mismos mediante un procedimiento de aleatorización conocido como skipping, [15]. Esto es debido a que las secuencias caóticas, a diferencia de las estocásticas, presentan estructuras internas que degradan su aleatoriedad. Este es el caso de las secuencias TWBMd y FWTSMd. El procedimiento de skipping consiste en saltar $(d - 1)$ valores de la secuencia caótica para obtener las secuencias TWBMd y FWTSMd. En otras palabras, se emplea, en lugar del mapa original M , su d -iteración esto es M^d . Esta técnica de aleatorización se utiliza rutinariamente (y con éxito) con mapas lineales por tramos en muchas aplicaciones [16]. En la Tabla I, se puede ver un ejemplo para distintos valores de d .

IV. CUANTIFICADORES DE PERFORMANCE

Para evaluar la performance que presentan las distintas configuraciones de los sistemas simulados se utilizaron tres cuantificadores. Dos que cuantifican la similitud de la señal recuperada con la original, en el tiempo y en frecuencia.

- Similitud temporal:

$$\langle S_t \rangle = \frac{\|x - \hat{x}\|_2}{\|x\|_2}$$

- Similitud espectral:

$$\langle S_e \rangle = \frac{\| |fft(x)| - |fft(\hat{x})| \|_2}{\| |fft(x)| \|_2} \quad (3)$$

También nos interesa la varianza $Var(S_e)$ y $Var(S_t)$ de estos cuantificadores dentro de las simulaciones realizadas para tener una buena estadística.

El tercer cuantificador es el valor medio de la cantidad de iteraciones $\langle I_m \rangle$ requeridas por el algoritmo de optimización empleado para resolver la ecuación (1). Cuanto menor sea el valor de estos cuantificadores indican una mejor performance del sistema. La señal recuperada tendrá menos error y aumentará la velocidad de convergencia del sistema.

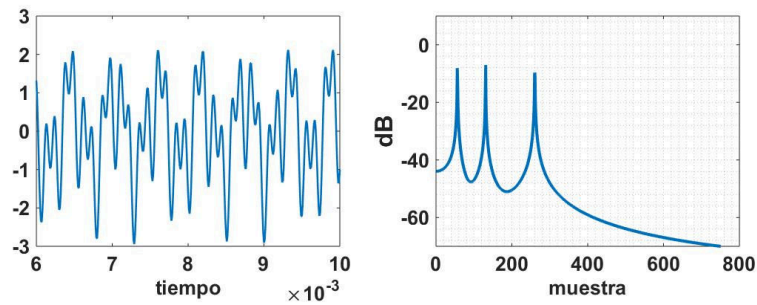
V. RESULTADOS

Una vez decidido el tipo de sistema a implementar, se realizó la simulación del mismo en el entorno *Matlab*[®] con una frecuencia de muestro general de $100kHz$, tres tonos de prueba de distintas frecuencias, ventana de procesamiento $N = 5000$ muestras, submuestreo a $10kHz$ con $M = 500$ muestras (90% menos), frecuencia de $Pc(t)$ en $20kHz$, base para la representación de las señales a la transformada discreta coseno (DCT) y un filtro FIR de fase lineal con frecuencia de corte en $4,5kHz$.

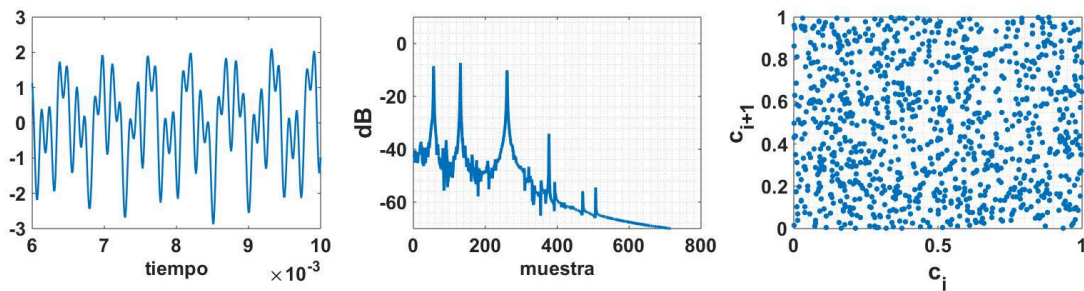
Para la generación de la matriz A utilizada en la ecuación (1) se requieren muestras de $h(t)$ y la secuencia $Pc(t)$ utilizada en la compresión. Cabe destacar que se programó un algoritmo novedoso de optimización por homotopía simple y eficiente presentado en 2015 por Lijun Zhang et al. en [17]. Los resultados de la simulación muestran un error cuadrático relativo a la secuencia $x[n]$ original igual a 0,0422.

Se realizaron pruebas con diferentes mapas caóticos y se reportan los resultados más interesantes en las que se puede ver que a medida que se utiliza la técnica de skipping mejora la reconstrucción de la señal. En la Tabla II puede verse los resultados obtenidos para los cuantificadores S_t y S_e , calculados según las ecuaciones (3). Para la el cálculo de los resultados obtenidos se promedió en 200 simulaciones para cada secuencia de muestreo. Se reportan en la Tabla los resultados más interesantes. Puede verse que los valores obtenidos para las secuencias caóticas son similares a los obtenidos mediante la secuencia clásica (rand).

En la Figura 3 puede verse la señal de prueba enviada (Fig. 3.a), y las señales recuperadas mediante un muestreo clásico (rand) (Fig. 3.b). En la Figura 4 se ven las señales recuperadas obtenidas utilizando una matriz de muestreo caótica. La primer y segunda columnas corresponden a la señal temporal y su espectro respectivamente de la señal recuperada, la tercer columna corresponde a la secuencia utilizada para el muestreo. Se eligió representar las secuencias en un plano c_{i+1} vs c_i ya que este plano pone en evidencia las estructuras internas de los mapas caóticos empleados. Puede verse que a medida que se utiliza una secuencia iterada mayor (skipping), las estructuras internas disminuyen, sin embargo, la mejora en la señal recuperada cuando se utiliza una secuencia con skipping más alto no es significativa. Por otro lado, en el caso de muestrear con secuencias generadas por el mapa Logístico la señal recuperada se asemeja más a la señal original, como puede verse en la Fig. 4. Esto se confirma con la Tabla II donde



(a) Original.



(b) Rand.

Figura 3. Señal original en tiempo, frecuencia y señal recuperada con muestreo clásico (secuencia Rand) para la generación de la matriz A .

Cuadro II. COMPARACIÓN DE PERFORMANCE CON LOS DISTINTOS TIPOS DE MUESTREOS.

c_i	$\langle S_t \rangle$	$Var(S_t)$	$\langle S_e \rangle$	$Var(S_e)$	$\langle I_m \rangle$
Rand	0,0635	0,0175	0,0480	0,0149	1054
LOG	0,0627	0,0177	0,0476	0,0150	1050
FWTSM	0,1083	0,0298	0,0828	0,0245	1029
TWTSM	0,0970	0,0272	0,0751	0,0226	1027
TWBM	0,0955	0,0245	0,0845	0,0246	1111
TWBM2	0,0647	0,0203	0,0516	0,0189	1171
TWBM4	0,0631	0,0187	0,0479	0,0165	1057

se ve que la señal recuperada en el caso del mapa Logístico presenta el menor error, esto es menor valor medio y menor varianza de los cuantificadores de similitud. En la Fig. 5 se muestran los histogramas de las similitudes temporales que se fueron obteniendo para las 200 muestras. Los resultados son consistentes con lo que muestran las Figuras 3 y 4, la similitud temporal media $\langle S_t \rangle$ lograda con el Logístico tiene menor valor que en el caso Rand. El resto de los caóticos consiguen resultados similares aunque no mejores que Rand.

En comparación con el paradigma clásico de muestreo, con el sistema diseñado hasta el momento se obtiene un aumento del 63,3% del ancho de banda, es decir, que se logra reconstruir la señal muestreando al 61,2% de la frecuencia de Nyquist.

VI. CONCLUSIONES

Se diseñó y simuló un sistema de adquisición de señales empleando Xampling y CS, a una frecuencia 61,2% menor

que la determinada por el teorema de muestreo de Shannon-Nyquist. Para la implementación del muestreo se estudió la utilización de secuencias caóticas. Se reportaron los resultados obtenidos de los cuales se desprende que los sistemas con caos presentan un comportamiento comparable al clásico. En particular las secuencias generadas con el mapa Logístico resultaron ser las mejores.

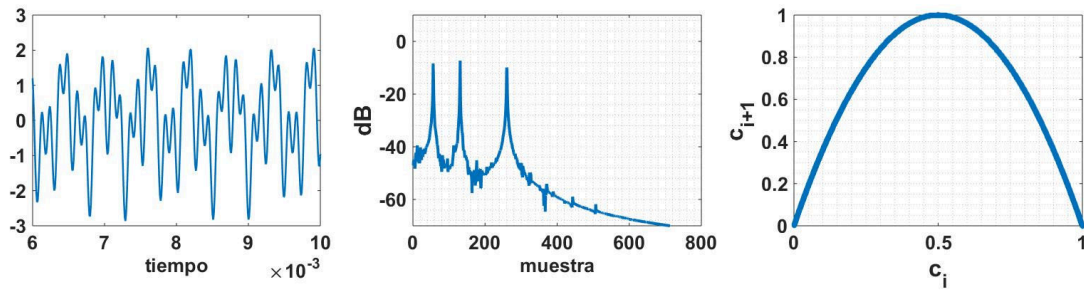
El trabajo futuro será implementar en hardware el sistema para analizar experimentalmente topologías alternativas del demodulador aleatorio con el fin de aumentar la incoherencia entre las matrices Ψ y φ . También, se desarrollarán algoritmos de corrección para la estimación a la respuesta al impulso ante variaciones de los componentes del filtro analógico con el fin de lograr la autocalibración del sistema.

AGRADECIMIENTOS

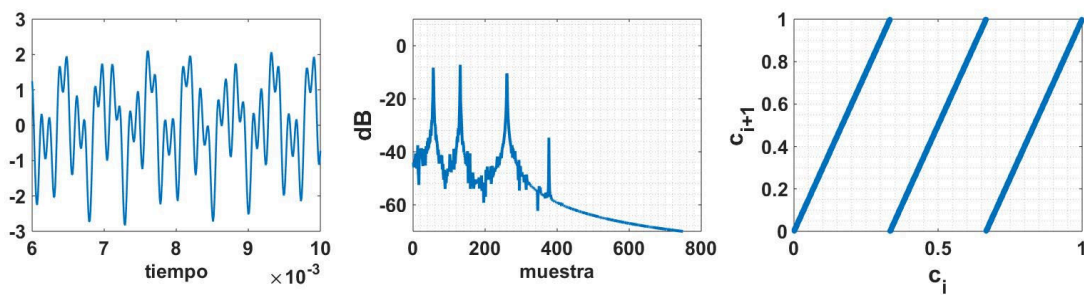
Este trabajo fue parcialmente financiado por el Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), (PIP 112-201101-00840), UNMDP Argentina e ICTP (International Centre for Theoretical Physics).

REFERENCIAS

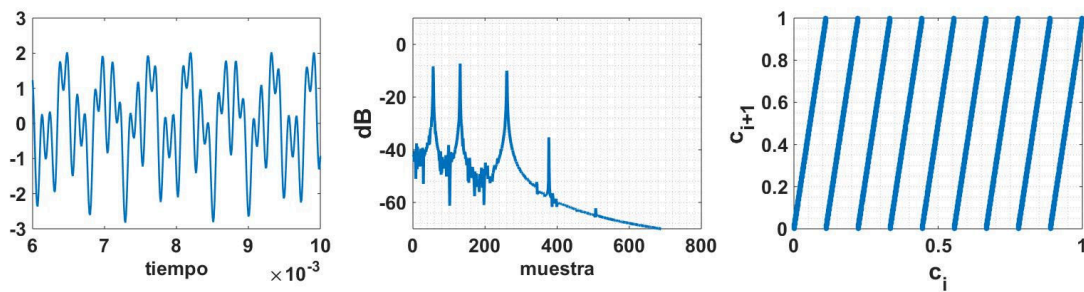
- [1] M. Davenport, "The fundamentals of compressive sensing," *IEEE Signal Processing Society Online Tutorial Library*, 2013.
- [2] F. Li, T. J. Cornwell, and F. de Hoog, "The application of compressive sampling to radio astronomy-i. deconvolution," *Astronomy & Astrophysics*, vol. 528, p. A31, 2011.



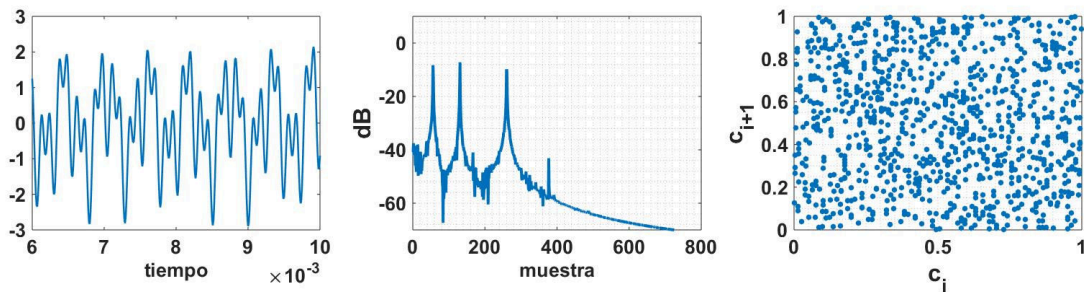
(a) Logístico.



(b) TWBM.



(c) TWBM2.



(d) TWBM4.

Figura 4. Señal recuperada en tiempo, frecuencia y secuencia caótica utilizada en el muestreo para la generación de la matriz A .

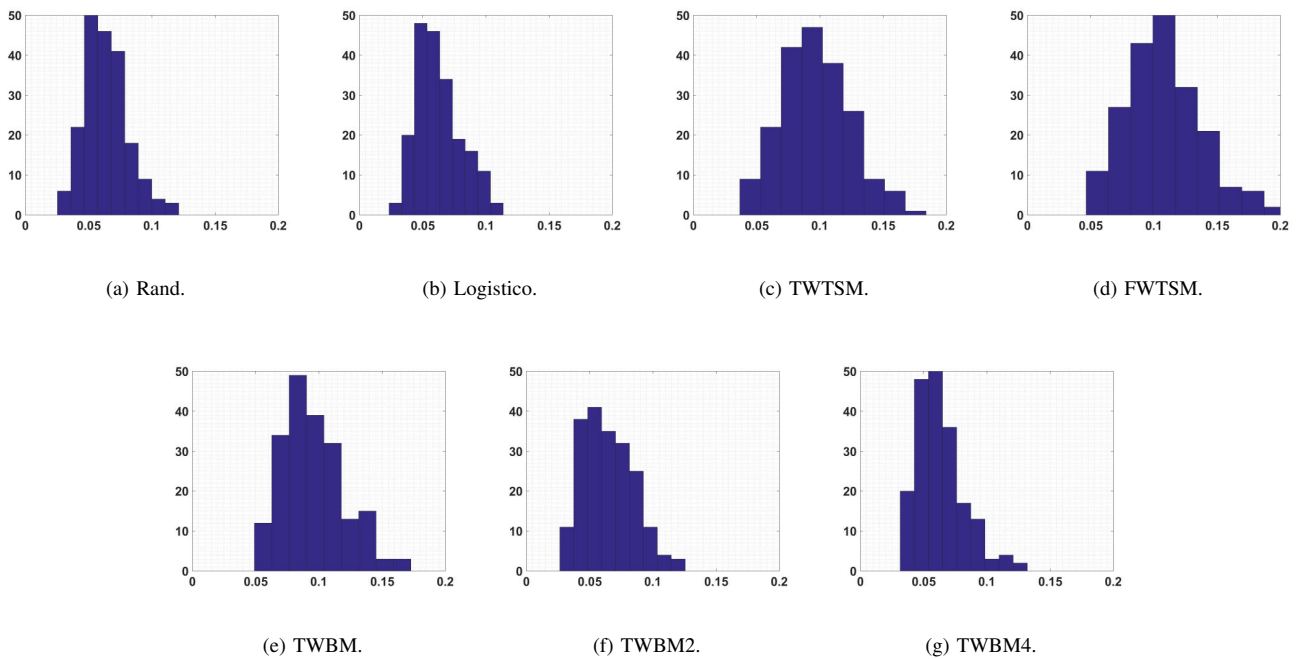


Figura 5. Histograma de la Similitud temporal S_t de las señales recuperadas con cada muestreo sobre las 200 simulaciones.

- [3] M. Mishali and Y. C. Eldar, "Blind multiband signal reconstruction: Compressed sensing for analog signals," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 993–1009, 2009.
- [4] D. S. Smith, J. C. Gore, T. E. Yankeelov, and E. B. Welch, "Real-time compressive sensing mri reconstruction using gpu computing and split bregman methods," *International journal of biomedical imaging*, vol. 2012, 2012.
- [5] L. M. Pecora, T. L. Carroll, G. A. Johnson, D. J. Mar, and J. F. Heagy, "Fundamentals of synchronization in chaotic systems, concepts, and applications," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 7, no. 4, pp. 520–543, 1997.
- [6] L. M. Pecora and T. L. Carroll, "Synchronization in chaotic systems," *Physical review letters*, vol. 64, no. 8, p. 821, 1990.
- [7] Z. Liu, S. Chen, and F. Xi, "A compressed sensing framework of frequency-sparse signals through chaotic system," *International Journal of Bifurcation and Chaos*, vol. 22, no. 06, p. 1250151, 2012.
- [8] N. Linh-Trung, D. Van Phong, Z. M. Hussain, H. T. Huynh, V. L. Morgan, and J. C. Gore, "Compressed sensing using chaos filters," in *Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian*. IEEE, 2008, pp. 219–223.
- [9] L. Yu, J. P. Barbot, G. Zheng, and H. Sun, "Compressive sensing with chaotic sequence," *IEEE Signal Processing Letters*, vol. 17, no. 8, pp. 731–734, 2010.
- [10] S. Kirolos, J. Laska, M. Wakin, M. Duarte, D. Baron, T. Ragheb, Y. Massoud, and R. Baraniuk, "Analog-to-information conversion via random demodulation," in *Design, Applications, Integration and Software, 2006 IEEE Dallas/CAS Workshop on*. IEEE, 2006, pp. 71–74.
- [11] S. Foucart and H. Rauhut, *A mathematical introduction to compressive sensing*. Birkhäuser Basel, 2013, vol. 1, no. 3.
- [12] Y. C. Eldar and G. Kutyniok, *Compressed sensing: theory and applications*. Cambridge University Press, 2012.
- [13] M. Mishali, Y. C. Eldar, and A. J. Elron, "Xampling: Signal acquisition and processing in union of subspaces," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4719–4734, 2011.
- [14] R. Roy and T. Kailath, "Esprit-estimation of signal parameters via rotational invariance techniques," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 37, no. 7, pp. 984–995, 1989.
- [15] L. De Micco, C. M. González, H. A. Larrondo, M. T. Martin, A. Platinio, and O. A. Rosso, "Randomizing nonlinear maps via symbolic dynamics," *Physica A*, vol. 387, pp. 3373–3383, 2008.
- [16] G. Setti, G. Mazzini, R. Rovatti, and S. Callegari, "Statistical modeling of discrete-time chaotic processes: Basic finite-dimensional tools and applications," *Proceedings of the IEEE*, vol. 90, no. 5, pp. 662–689, Mayo 2002.
- [17] L. Zhang, T. Yang, R. Jin, and Z.-H. Zhou, "A simple homotopy algorithm for compressive sensing," in *AISTATS*, 2015.

Implementación Multi-Core en la Computadora Industrial Abierta Argentina – NXP®

Marcos D. Aranda, Paola I. Beltramini, Sergio H. Gallina, Jesús Eduardo Cano, Gabriel A. Díaz, María Valeria Poliche and Marcelo D'amore

Laboratorio de Sistemas Embebidos - Facultad de Tecnología y Ciencias Aplicadas – UNCA
Catamarca – Argentina
lase@tecno.unca.edu.ar

A través del presente trabajo se expone un ejemplo de implementación de multiprogramación utilizando la Computadora Industrial Abierta Argentina (CIAA) en la educación universitaria, en el desarrollo y construcción de un *Laboratorio Portátil* para la enseñanza de electrónica básica en carreras de Ingeniería de la Facultad de Tecnología y Ciencias Aplicadas de la Universidad Nacional de Catamarca [1]. Específicamente se describe el diseño e implementación de dos etapas del Laboratorio que deben ejecutarse simultáneamente, un generador de señales y una interfaz gráfica de usuario, detallando la comunicación entre ambos.

La multiprogramación consiste en hacer residir en memoria central programas que son ejecutados de manera temporalmente imbricada por una misma unidad central, pasando de un programa a otros gracias al mecanismo de las interrupciones o de memoria compartida [2].

La primera versión de CIAA denominada CIAA-NXP®, está basada en un procesador LPC43xx de la empresa NXP Semiconductors y es la primera y única computadora que reúne dos cualidades:

1. Ser industrial, ya que su diseño está preparado para las exigencias de confiabilidad, temperatura, vibraciones, ruido electromagnético, tensiones, cortocircuitos, etc., que demandan los productos y procesos industriales.
2. Ser abierta, ya que toda la información sobre su diseño de hardware, firmware, software, etc. está libremente disponible en internet bajo la licencia Berkeley Software Distribution (BSP), para uso libre. [3]

El LPC43xx es un microcontrolador multicore que posee un núcleo ARM Cortex-M4 y un segundo núcleo ARM Cortex-M0. El Cortex-M4 se utiliza como procesador principal y luego de un reinicio, como controlador de sistema de nivel superior, y el Cortex-M0 puede ser utilizado como coprocesador y además puede realizar tareas de E/S en serie [4]. Entonces, el M4 puede comunicarse con uno o ambos núcleos del M0 a través del espacio de memoria compartida e interrupciones "Fig. 1". [5]

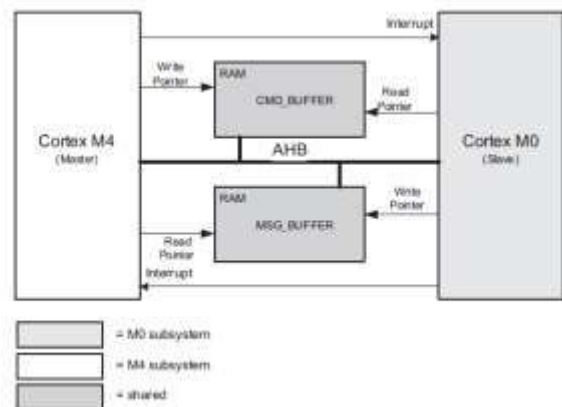


Fig. 1: Diagrama en bloques del core del LPC43xx

En el marco del proyecto, y con el propósito de utilizar la potencialidad multicore del LPC4337 de la CIAA, se desarrollan e implementan aplicaciones que deben correr en forma paralela:

- Sobre el núcleo M0, un generador de señales con las siguientes características técnicas: formas de onda triangular, cuadrada y senoide; de frecuencia variable entre 1 Hz a 5 KHz y amplitud variable de 2 Vpp (pico a pico) a 6 Vpp con un valor medio de 5V.
- Sobre el núcleo M4, aprovechando que éste cuenta con un set más amplio de instrucciones, se implementan las siguientes aplicaciones: un subsistema de comunicaciones para conectar el Laboratorio de escritorio del alumno con la red del aula (RS485); la interfaz gráfica del usuario y la aplicación propiamente dicha, consistente de una serie de premisas o pautas para que el alumno desarrolle con este laboratorio de escritorio.

La codificación del generador de señales se realiza de la siguiente manera:

1. Inicializar la placa, entradas y salidas digitales.
2. Inicializar el Timer (temporizador) de Interrupciones Repetitivas (RIT) cada 1 milisegundo.

3. Cuando el programa principal se ejecuta, cada 1 milisegundo se produce una interrupción del tipo RIT encargada de seleccionar que señal se desea visualizar con un OFFSET de 5V, capturando el código binario una llave selectora de cuatro posiciones. Esto se realiza comandando un circuito integrado (el CI 74148) que entrega las salidas digitales A0, A1.

- a. 00 “Ninguna señal seleccionada”
- b. 01 “Señal Cuadrada”
- c. 10 “Señal Triangular”
- d. 11 “Señal Sinusoide”

4. Para generar las señales triangular y sinusoide se implementa una interrupción utilizando el TIMER0, y la señal cuadrada utiliza el TIMER3, cuando se habilita una de ellas la otra se deshabilita y viceversa. Se cargan tres vectores con muestras suficientes con los valores de variación de amplitud de cada tipo de señal, y cada vector es invocado por diferentes períodos de tiempo de acuerdo a la selección del usuario. Estos períodos son ingresados como parámetro para las interrupciones de los TIMER. Un pulsador P1 permite variar la frecuencia entre 10 Hz y 5 KHz. Por su parte otro pulsador P2 se utiliza para cambiar entre dos valores la amplitud de las señales (6 Vpp y 2 Vpp).

5. El código compilado sin errores se descarga al segundo banco de la memoria flash del Cortex-M0.

Como interfaz gráfica se utiliza una pantalla táctil HMI (Human Machine Interface) Inteligente de 7" Nextion. Nextion es una solución que proporciona una interfaz de control y visualización entre el usuario, proceso, máquina, aplicación o dispositivo. Nextion se aplica principalmente a la IoT (Internet de las Cosas) o a la electrónica de consumo. Es la mejor solución para reemplazar el LCD tradicional. Este display incluye en la parte de hardware una serie de placas TFT y en la parte del software un editor de Nextion. La placa de Nextion TFT utiliza un único puerto serie para realizar la comunicación. [6]

El manejo de la pantalla se codifica de tal manera que se utilice el Cortex-M4 para la comunicación entre el microcontrolador y la interfaz gráfica de usuario.

Para ejecutar la aplicación del Generador de Señales que se encuentra alojado en el segundo banco de memoria del Cortex-M0, se escribe en el programa principal del Cortex-M4 el siguiente comando:

```
cr_start_m0(SLAVE_M0APP, (uint8_t *)0x1B000000)
```

Debido a que el procesador Cortex-M4 es el maestro, debe ejecutarse primeramente para que el coprocesador Cortex-M0 comience a trabajar.

Para lograr la comunicación entre ambos Core, se crea un espacio de memoria compartida de 8 Kb en la SRAM, utilizando un puntero a puntero con la siguiente instrucción:

```
#define SIGNAL_SQUARE_M0 (*(volatile U32 *) (0x10089000))
```

De esta manera se logra que los datos generados en el Cortex-M0 para las diferentes señales, puedan ser adquiridos por el Cortex-M4 por el uso de memoria compartida y graficar en el display las distintas señales.

A manera de ejemplo, para mostrar la implementación de la arquitectura multicore se selecciono una señal cuadrada de frecuencia 1 Hz y una amplitud de 2 Vpp que se muestra gráficamente en la pantalla, “Fig. 2”.

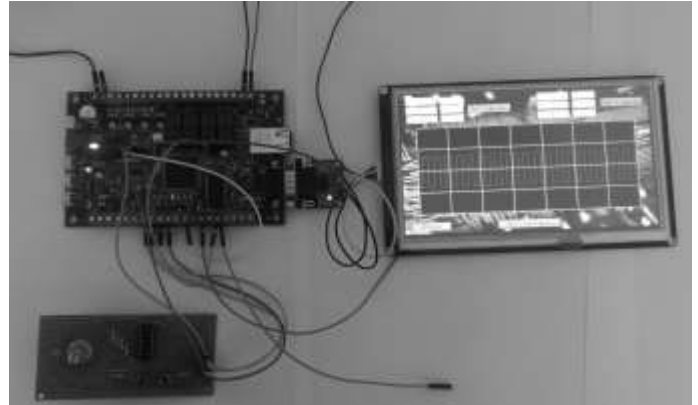


Fig.2: Ejemplo de implementación de multiprogramación: señal Cuadrada

De las pruebas realizadas se puede afirmar que usar la arquitectura multicore de la CIAA ha sido factible, sin mayor complejidad, y redundante en un uso eficiente de la unidad de procesamiento al lograr una mejor distribución de las tareas.

No obstante, se han encontrados limitaciones en la frecuencia máxima de la onda generada debido a la presencia de distorsiones, lo cual requiere seguir trabajando sobre los algoritmos para poder aumentar dicha frecuencia y ampliar las aplicaciones del Laboratorio.

Un resultado importante de destacar es el uso y funcionamiento de la CIAA, concebida para aplicaciones industriales, en un desarrollo de tecnología educativa.

REFERENCES

- [1] J. E. Cano, M. V. Poliche, P. I. Beltramini, S. H. Gallina, M. D' amore, L. Schneider, M. A. Aranda & F. S. Fama “Laboratorio portátil de escritorio para la enseñanza de la electrónica analógica y digital”, Producción Científica de la Facultad de Tecnología y Ciencias Aplicadas VI, ISBN: 978-987-661-233-3, 2016.
- [2] Multiprocesadores y Máquinas paralelas. <http://www.dte.eis.uva.es/Docencia/ETSII/SMP/BAK/tema4/ANEXO4.pdf>. Dpto. de Tecnología Electrónica. Escuela Técnica Superior en Ing. en Informática. Universidad de Sevilla. Última consulta 08/05/2017.
- [3] Proyecto Computadora Industrial Abierta Argentina (CIAA). <http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=start>. Última consulta 28/04/2017.
- [4] NXP B.V. Manual de Usuario UM10503 LPC43xx ARM Cortex-M4/M0 multi-core microcontroller. Rev. 1.9 2015.
- [5] M. D. Aranda, M. L. Ferraro, S. H. Gallina, N. S. Gallina “Sistema de Tiempo Real – Introducción al Diseño”. Ed. Asociación civil para la investigación, promoción y desarrollo de los sistemas electrónicos embebidos. 2015, ISBN 978-987-45523-7-2
- [6] Display Nextion Human Machine Interface (HMI) https://www.itead.cc/wiki/Nextion_HMI_Solution. Última consulta 28/04/2017.

Foro Tecnológico

Resumen

Bioingeniería y Robótica

Design and development of a steady-state auditory evoked potential meter (SSEP)

Elber Emanuel Sajama, Esteban Lucio Gonzalez, Juan C. Tulli Alejandro José Uriz, Iván Exequiel Gelosi

Communications Lab
National University of Mar del Plata
Mar del Plata, Argentina
Email: elgonzal@fi.mdp.edu.ar

CONICET - Communications Lab
National University of Mar del Plata
Mar del Plata, Argentina
Email: ajuriz@fi.mdp.edu.ar

Abstract—Conventional audiometry techniques explore the auditory apparatus through a conscious and voluntary response of the patient. But in some particular cases (for example in babies or disabled people) this response is not always reliable, or feasible. Due to this possibility, it was necessary to develop a new method that allows obtaining information from the auditory system and its pathway, without the collaboration of the patient. Thus, objective methods have been developed that explore the sensory system, but based on the study of some reflex or an unconscious response of the subject. So far, several electrophysiological techniques have been developed to perform objective audiometries detailed by frequencies. The most recent and promising technique in the audiometric field is the one that employs Steady-State Evoked Potential (SSEP) or also Auditory Steady-State Response (ASSR).

In this paper it is described a system that detect hearing loss at different frequencies by using a novel technique, which consists on the use of multiple frequencies and multiple amplitudes of stimulation, and the corresponding recording of Steady-State Evoked Potential SSEP

Index Terms—Audiometry, Hearing loss, Steady-State Evoked Potential (SSEP), Electromyography (EMG), Electrooculography (EOG), Electrocardiograph (ECG), Auditory Response.

I. INTRODUCTION

It has been shown that the early years of children's lives constitute a critical period in linguistic and intellectual development. Therefore, in the case of an auditory disorder, the fast detection and the subsequent treatment through the use of hearing aids and psychopedagogical intervention are fundamental to guarantee the development in children with auditory deficit [1]. This is why early detection of hearing impairment is a global health goal.

There are a number of methods that allow for a diagnosis of these cases, being the most commonly used, the record of Brainstem Auditory Evoked Potentials (BAEPs) or also called Brainstem Auditory Evoked Responses (BAERs), and Auditory Steady-State Response (ASSR) [2]. Its main advantages lie in their non-invasive nature. The ability to locate the lesion topographically, the immunity of the procedure's response to sleep or sedation of the patient and the ability to diagnose hearing impairment in cases where the patient can not communicate with the person responsible for the study,

and therefore it is impossible to carry out traditional studies such as a tonal audiometry.

The main difference between these two techniques, and which gives the ASSR a capability that BAEP lacks, is based on the form of the auditory stimulus. In the case of BAEP the stimulus is of the type called "click", which is a non-repetitive broadband signal. Thus, the measured response is in turn a transient signal in which the information is also in temporal form; While in the ASSR the stimulus is formed by a modulated and periodic narrowband signal, which carrier frequency ranging the order of human audible frequencies Modulated by low frequency pulses. So that the transient response produced by one of the pulses (low frequency signal) are superimposed with the following, forming a quasi-sine-like periodic response, which is the actual ASSR. Due to this periodicity characteristic of the ASSR, it is possible to perform a quantitative analysis in the frequency domain, resulting of interest only the amplitude of the response to a given frequency, coincident with the carrier frequency of stimulation. [3] [4] [5]

II. METHODS FOR HEARING EVALUATION

A. Proposed Electroaudiometric Techniques

Auditory Steady-State Response (ASSR) allows an audiometric evaluation to be performed objectively. This occurs when a repetitive stimulus is applied to the auditory apparatus, which, in turn, evokes an electric waveform whose frequency components are maintained constant in amplitude and phase over relatively long periods of time [6]. Steady-State responses occur when the stimuli have a sufficiently fast cadence to allow the response to one stimuli to overlap the response of the subsequent stimulus, that is, by stimulating the auditory system with a fast enough sequence of "clicks". The transient response caused by a stimulus overlaps with that of the preceding stimulus generating an approximately sinusoidal signal. Due to the periodicity presented by this type of potential, its response can be evaluated using quantitative techniques based on the frequency analysis. The scheme of the method can be seen in Figure 1.

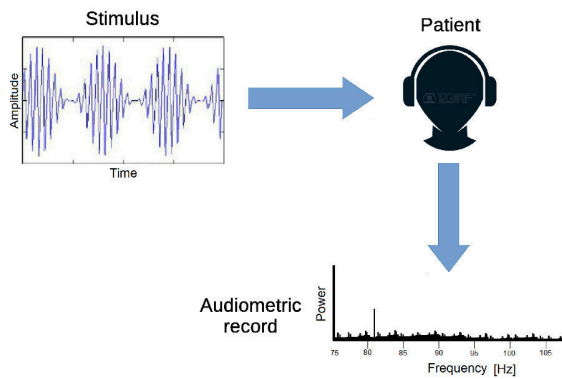


Fig. 1. AASR operating scheme.

III. PROPOSED SYSTEM.

This document shows the development of a system oriented to record electrophysiological signals, with the propose of obtaining a equipment capable of measuring mainly ASSR, at a lower cost than the equipment currently on the market. It also seeks to determine the patient's auditory thresholds depending on frequencies in an objective, automatic and easy way to interpret. As a secondary function, it is sought to record other electrophysiological signals, such as Electromyography (EMG), Electrooculography (EOG) and Electrocardiography (ECG) signals, in order to have a versatile device and not to be limited only to ASSR measurement.

This is carried out by a software running on a personal computer, which is responsible for data processing, data graphing, and the generation of auditory stimuli using its audio outputs. The computer is connected to a high-resolution acquisition board, which is responsible for converting different analog signals into their digital format and sending them to the computer. The general scheme of the system is presented in Figure 2.

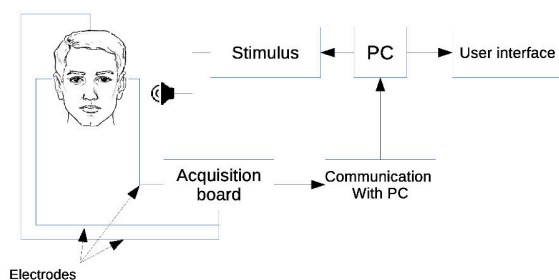


Fig. 2. General scheme of the proposed system.

A. Stimulus

Modulated amplitude (AM) signals, are used as auditory stimuli, with a modulation index of 100%. The carrier frequencies of these signals are located in the audible frequency range that is wanted to study in the patient, it is, the frequencies that can be listened by the human being. That ranges from

20Hz to 20KHz, but if necessary, it is possible to focus on a narrower band. The modulating signal or envelope, has a frequency equal to the inverse of latency. Latency is the time it takes to the stimulus to pass through the entire auditory canal, this latency is approximately 25msec, therefore the modulating frequency must be approximately 40Hz [7]. In turn this modulating signal is the one that must be observed in the spectral analysis if the patient hears the carrier frequency properly. The audio tones are generated using a mathematical software, and this are sent to the patient. This is done by the software designed in this work and reproduced by the computer audio card. An example of a modulated tone using Matlab is presented in Figure 3.

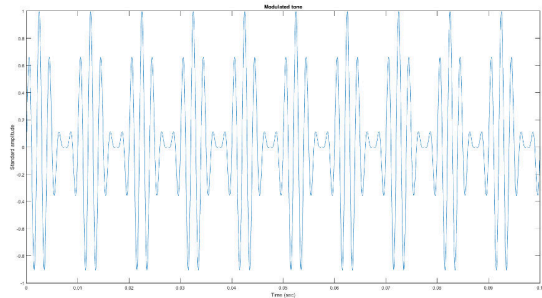


Fig. 3. Auditory stimulation signal. In Figure a 500Hz carrier frequency signal modulated by a signal of 100Hz can be seen.

B. Recording of the electrophysiological signals

Gold cup electrodes were used for the implementation of this module, these are fixed to the scalp with conductive paste, in order to keep the impedance as low as possible (less than 5 kΩ preferably) This allows to obtain a noise free recording [8].

For the acquisition block, it is necessary to emphasize that signals to be measured are of very low amplitude, therefore the acquisition block must have a delta-sigma analog to digital converter (ADC) [9] [10]. Basically, these converters consist of an oversampling modulator followed by a digital/decimation filter that together produce a high-resolution data-stream output. It is also necessary for the acquisition block to have good dynamic range, good signal-to-noise ratio (SNR), and a high effective bit number.

The effective number of bits is based on the equation for an ideal ADC, in which its ideal signal-to-noise ratio is shown in Equation 1.

$$SNR = 6.02 * N + 1,76dB \quad (1)$$

Where N is the number of bits. The latter being associated to the SNR by the Equation 2. In view of the foregoing, three AD converter were evaluated: AD7770 [11], ADS1299 [12] and MAX11410K [13]. If a 24-bit converter is chosen, its ideal SNR is 146.24dB, this would only be achieved if there were no noise, but in the real world the ideal SNR is never

achieved due to its own noise and system errors. Arranging the Equation 1 to obtain the effective N, resulting in

$$ENOB = (SNR_{real} - 1,76)/6,02 \quad (2)$$

Where SNR_{real} indicates the quality (SNR) of the acquired signal.

The circuit that best suits the needs of this project is the ADS1299. It is a 24-bit analog-digital (A/D) converter, with 8 channels (each of them having a programmable low-noise amplifier). Sampling rates can be chosen between $250SPS$ and $16KSPS$. It uses an SPI interface for chip configuration and for sending data. In addition it has incorporated specific functions for data acquisition of electrophysiological signals. The acquisition board also features a microcontroller PIC18F4550 [14], which is responsible for configuring the ADS1299, gathering the data and sending it to the computer. In turn also receives commands from the computer and sends them to the converter to control the start and end conversion of analog data to digital format. In this acquisition module was added an isolation between the ADS1299 and the microcontroller, since both of them have different power supplies. The microcontroller receives power from the computer, while the converter has its own isolated power supply. Therefore it is necessary to isolate both stages to prevent possible leaks from the computer reaching the patient. For doing this, an isolation buffer was used in the middle of the SPI communication and the control pins in both directions.

An image of the finished board is shown in Figure 4. It can be seen the integrated circuits used and the gold cup electrodes connected to the board.

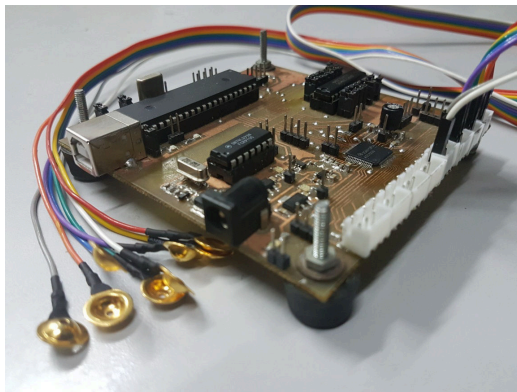


Fig. 4. Implemented board

C. Communication with the PC

Before starting with the description of the method used to communicate with the computer, it is necessary to highlight the available time to process the data in the microcontroller. This time is given by the time between samples of the ADS1299. The ADS1299 was configured with a speed of $2KSPS$. This sampling frequency was chosen to obtain at least 20 samples per each input signal cycle (evoked potential signals range the order of $100Hz$) and thus to achieve a better

graphic representation and more data for further processing. The available time between samples is obtained by inverting the sampling rate. This time is used to read the new data through the SPI port, also to format and transmit the data to the computer. The time interval, in this case, is $500\mu sec$.

The theoretical time required to read a single sample from the converter, is calculated by taking into account the total number of bits sent by it ($216bits$, which are composed of 24 status bits + 24 bits x 8 channels), divided by the speed of the module SPI ($12Mbps$). This gives a time of $18\mu sec$. Since we are using an 8-bit microcontroller, samples are read in 8-bit words. To measure the actual time used in reading a sample of the 8 channels, a logic analyzer was used connected to a pin of the PIC (channel 6: PIN_FLAG). This one is in high level while not receiving data, when a data is received, the PIN_FLAG goes low. In this case, the falling edge of this pin is used to indicate the start of data acquisition by SPI. An example of the time taken the SPI to read data from ADS1299 is shown in Figure 5. Time $T2$ marks the start of SPI communication, while time $T3$ indicates the end of the communication since the select chip is set to "1".

By subtracting $T3 - T2$, the time spent in the SPI communication is obtained. In this case the time was $62.72\mu sec$. If the time between $T4$ and $T2$ ($88.52\mu sec$) is taken into account, it provides information on the total time required for SPI communication and formatting acquired data. Therefore, by subtracting this time from the $500\mu sec$ (time between samples), only about $410\mu sec$ remain for communication with the computer. Due to these times, it was decided to choose the USB protocol to transfer the data to the computer.

Key factors that supported the choice of the USB protocol over RS232, were mainly its great advantages such as: device detection, data flow management, increased capacity to transfer data, higher transfer speed, error detection, *Plug and Play* support, power supply, among others.

One of the important subjects to be addressed is data transfer speed and packet latency. USB 2.0 supports 3 types of speeds: $1.5Mbps$ (Low Speed), $12Mbps$ (Full Speed) and $480Mbps$ (High Speed). The drawback is that the transmissions are half-duplex, and a host is needed for data flow control in both directions. For traffic control, time is divided into intervals called *frames* (in low and full speed) or *microframes* (for high speed). The host reserves a position of each frame or *microframe* for each transfer. For low and full speed, a frame has a period of $1msec$, and for high-speed traffic, the host divides the frame into 8 *microframes* of $125\mu sec$ [15].

There are four types of transfers, each handling different needs and devices can use the type that best suits your purposes. The types are: *control*, *bulk*, *interrupt* and *isochronous*.

The *bulk* transfer type was the best adapted to the requirements of this work. The advantages of this transfer type is that it has the ability to send a large amount of data (compared to other types) and at high speed. Its main disadvantage is that it does not have maximum medium priority, ie if the bus is busy, it must wait until it is released.

To make the connection between the computer and the

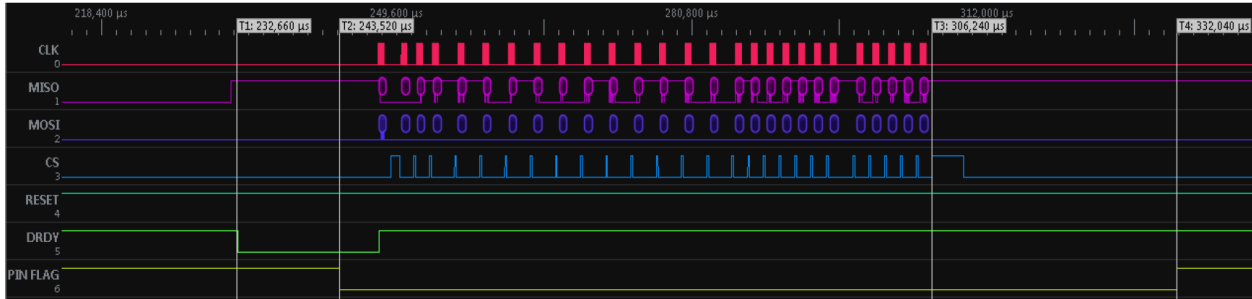


Fig. 5. The figure shows the time needed for processing both the SPI port and the data conditioning in the PIC.

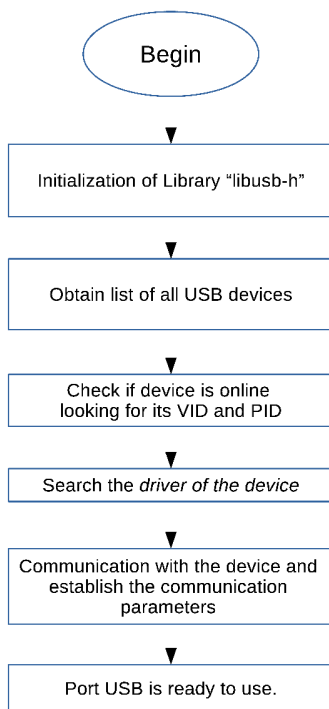


Fig. 6. Flow diagram of the USB configuration. If any of these steps are not met, the communication is aborted and must be started again.

acquisition board, it is necessary to perform a series of very specific steps of the handling of the hardware of the USB port, for which was used the library *libusb.h*. These steps can be seen in Figure 6

D. Software Analysis

It was designed a computer application to handle the complete system. It complies with the basic requirements proposed, such as reproducing the modulated tones, receiving the data sent by the acquisition board, graphing these data on the screen and performing the corresponding analysis. To these functions was added a preprocessing, which consists of digital low-pass filters and a $50Hz$ notch digital filter, in order to delete generic noise and line voltage noise, respectively. In the Figure 7, the block diagram of the application is presented.

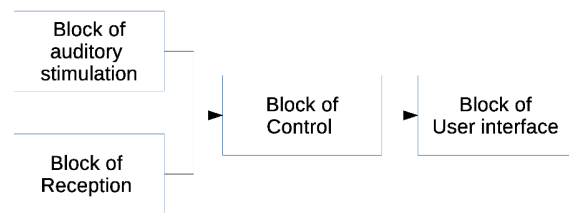


Fig. 7. Blocks that make up the application

This equipment allows to visualize the data in real time, this is due to the use of *threads* and the pipeline architecture, the *threads* allow to execute tasks in parallel, while the pipeline architecture allows to process data in a block, while the previous and later block processes previous and subsequent data respectively. In this way all blocks work in parallel, that is, each block is a *thread*.

A *thread* is the smallest processing unit that can be executed by the operating system. This section of code can be executed by the processor simultaneously with other *threads*. These *threads* can interact with each other, therefore it is necessary to implement mechanisms of communication and synchronization because the memory used by these *threads* is shared between all of them equally, and they all have access to the same segment of memory at any time, so, the integrity of the data can be compromised if no synchronization instructions are used.

In this system four *threads* will be used, one for each block that form the system. The *threads* are not all created at the same time, but there is a creation sequence, where one *threads* is created by another, this can be observed in Figure 8.

Regarding synchronization, each *thread* generates a synchronization call when needed. For example, when the Stimulation and Receive blocks respectively generate and receive a new data, a synchronization call is generated to store this new data in memory. In the case of the Control *threads*, it generates a synchronization call to see the amount of new data, when a certain amount of new data of both blocks was saved, it generates another synchronism to warn by means of a *threads* to the Block of User Interface which has data ready for graphing. The User Interface block generates synchronisms

to test the *threads* of the Control block, when it detects that it is positive, it generates a synchronism to read the memory and to graph the data in the screen. As the blocks are explained, more details of their operation will be given.

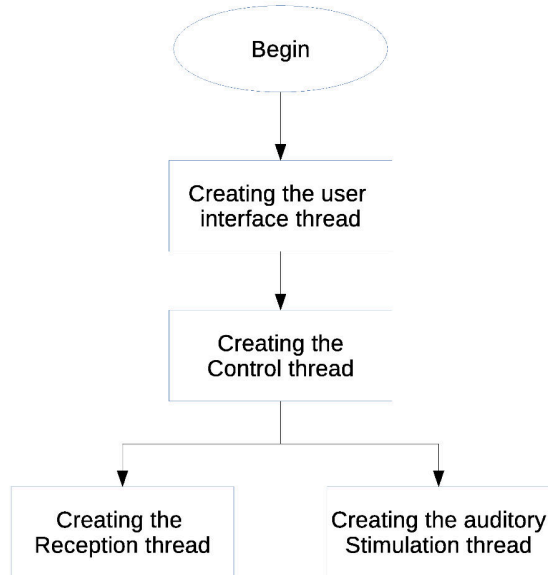


Fig. 8. Programming the *threads*.

Although it was said that the *threads* provide the possibility of executing tasks in parallel, in this case, the only *threads* that are executed in parallel are the Hearing Stimulation and Reception because it is necessary to obtain the response to the generated stimulus. The other *threads* do not need to be executed in parallel since these are dependent on the first two.

Auditory Stimulation Block, as mentioned before, this is responsible for reproducing the stimulation tone by handling the audio board of the computer. In order to do this, this block must previously be configured, that is, it must pass a series of parameters, where the most important are: the carrier frequency, modulating frequency, the amount of tones and the audio power. This block in turn sends the data it is reproducing to the user interface block, so that it can be graphed and the user can see them along with the evoked responses.

Receive block, just like the previous block, this needs to be configured before starting. The most important parameter is the number of pulses to be reproduced, with this information and an algorithm, this block determines the amount of sample that will be requested to the acquisition board. As the data enters the compute, this *threads* sends the data to the upper *thread* so that it can plot them and applies some processing if it indicates by the user interface.

Control Block, this block is vital for the operation of the system in real time, this is in charge of controlling the transfer of data from the Stimulation and Reception blocks to the user interface and to control the commands to Stimulation and Reception blocks. In addition it applies a pre-processing to

the data captured by the Reception block, this pre-processing basically is a filtering to adequate the signal before being processed in the strict sense.

User Interface Block, this block is the software behind the user interface itself, interface that is used to show graphs and to set parameters to both, the acquisition of evoked potentials and the pulses of the auditory stimulation blocks. In turn the interface presents a control layout similar to an oscilloscope to allow adjusting the acquired signals presentation. The processing for the detection of ASSR is also included in this block, this is done once the stimulation pulses have been transmitted and in turn, the acquisition has been accomplished.

The evoked response is analyzed in the frequency domain, however, it is first necessary to perform an average of the signals in order to improve the signal-to-noise ratio. This application proposed two ways of performing the average, the intra-channel average and the intercanal average. The intra-channel average is performed on the same channel. The buffer that holds the acquired data is first segmented into N segments, then the segments are averaged to obtain a single segment representing the response to the stimuli.

The intercanal average is performed by averaging the available M channels, in order to obtain a single channel that also represents the response to the stimuli. The average obtained in the time domain, by either method, is transformed into the frequency domain by means of the Fast Fourier Transform (FFT). If the patient listen to the carrier frequency, in the spectral representation a peak is detected at the modulation frequency.

Finally the interface presents two modes of operation as mentioned above. The primary mode is ASSR detection, while the second mode provides the function of recording different types of electrophysiological signals.

E. Analysis of other electrophysiological signals

This software allows for other secondary functions, such as tools for the analysis of acquired signals without the intervention of the auditory stimulator block. Thus becoming into a real time acquisition system for electrophysical signals. Different preprocesses can be applied to these signals, like filtering for example, and the algorithm of the FFT to determine some particular characteristic. It also includes a record stage, this last tool allows to save the signals acquired in raw format in a database, that is, it stores the signal as it was sent by the acquisition board. Saved signals can be reproduced later, and be processing without the necessity of having the acquisition board connected.

IV. RESULTS

The processing to analyze the response to the stimuli was tested using a sinusoidal signal provided by a signal generator, whose frequency is equal to the modulating frequency of the stimulus used. In this example, to form the stimulation signal a carrier frequency of $500Hz$ and a modulating frequency of $40Hz$ were used. In Figure 9 it can be seen that a $40Hz$ signal was acquired (this signal emulates the response to the

auditory stimuli). Then, the intra-channel averaging is applied, segmenting the data buffer in two. And as a last step an FFT is performed on the averaged data obtaining a peak at $40Hz$.

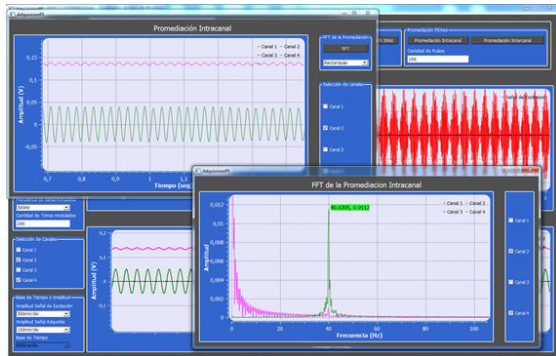


Fig. 9. Test of processing of a ASSR response using intra-channel averaging.

Tests were also performed by measuring electrooculography (EOG) signals, where they could be successfully recorded. When the patient looked up, a positive potential was recorded and when looking down, a negative potential was recorded, both with respect to the common mode voltage.

V. CONCLUSIONS

The processing for the analysis of steady-state evoked potentials worked correctly as well as software designed to display acquired signals. With the help of the *threads* and the pipeline architecture, a real-time data acquisition and sampling system was obtained.

The acquisition of electrophysiological signals with a higher signal level than brain signals could be observed on the screen with the help of digital filters. However it was found that recording of evoked responses to the auditory stimuli could not be properly done on some cases, because these signals are of the order of the microvolts, and they are very prone to be masked by noise. To deal with this problem a number of factor must be taken into account: The electrodes must be in very good contact with the scalp, which is not always possible due to hair or due to non-optimal skin preparation.

Even when there was done a big effort in the design of the board, it was noted that interference noise is present, because of that it was decided for a future improvement to redone the layout of the board. Moreover, other evoked potentials interfere with the response to auditory stimuli.

REFERENCES

- [1] R. L. Oscar Papazian, Israel Alfonso, "Actualizaciones en neurología infantil," *Medicina Buenos Aires*, vol. 67, 2007.
- [2] L. F. A. M. Asunción Martínez Fernández, Miguel Ángel Alanón Fernández, "Estudio comparativo entre potenciales evocados auditivos de estado estable, potenciales evocados auditivos de tronco cerebral y audiometría tonal liminar," *SCImago*, vol. 58, 2007.
- [3] D. A. M. Pedrón, "Potenciales evocados auditivos de estado estable a múltiples frecuencias: valoración de los estudios sobre localización de sus generadores cerebrales," *MEDISAN*, vol. 15, 2011.
- [4] M. P. R. P. Martínez-Beneito, A. Morant Ventura, "Potenciales evocados auditivos de estado estable a multifrecuencia como técnica de determinación de umbrales auditivos," *elsevier*, vol. 53, 2002.

- [5] y. o. Abalo, Maria Cecilia Pérez, "Los potenciales evocados auditivos de estado estable a múltiples frecuencias y su valor en la evaluación objetiva de la audición," *Auditio: Revista Electrónica de Audiología*, vol. 2, 2003.
- [6] O. G. Lins *et al.*, "Frequency-specific audiometry using steady-state responses," *Ear & Hearing*, vol. 17, 1996.
- [7] A. T. F. M. V. H. Eleina Mijares Nodarse, María Cecilia Pérez Abalo, "Caracterización electrofisiológica del potencial evocado auditivo de seguimiento a la modulación del estímulo acústico," *Acta Otorrinolaringológica Española*, vol. 62, 2011.
- [8] M. González, Juan Paul y AVECILLAS, "Diseño y construcción de un prototipo de electroencefalografía para adquisición de señales cerebrales," 2010.
- [9] T. Instruments., "How delta-sigma adcs work, part 1." 2016. [Online]. Available: <http://www.ti.com/lit/an/slyt423a/slyt423a.pdf>, Accessed: 16/10/2016
- [10] —, "How delta-sigma adcs work, part 2." 2016. [Online]. Available: <http://www.ti.com/lit/an/slyt438/slyt438.pdf>, Accessed: 16/10/2016
- [11] A. Devices, "Ad7770," 2016. [Online]. Available: <http://www.analog.com/media/en/technical-documentation/data-sheets/AD7770.pdf>, Accessed: 20/10/2016
- [12] T. Instruments., "Ads1299," 2016. [Online]. Available: <http://www.ti.com/lit/ds/symlink/ads1299.pdf>, Accessed: 20/10/2016
- [13] M. Integrated., "Max11410." 2016. [Online]. Available: <https://www.maximintegrated.com/en/products/analog/data-converters/analog-to-digitalconverters/ MAX11410.html>, Accessed: 20/10/2016
- [14] Microchip., "Ic18f2455/2550/4455/4550." 2016. [Online]. Available: <http://ww1.microchip.com/downloads/en/DeviceDoc/39632e.pdf>, Accessed: 21/10/2016
- [15] F. P. y. G. C. Aguirre, Andrés, "Interfaz usb genérica para comunicación con dispositivos electrónicos." 2007.

Diseño e implementación de Sistema Embebido para Reconocimiento de Palabras en Español

Gonzalo Ávila Alterach
Facultad de Ingeniería - Universidad de Buenos Aires
Ciudad Autónoma de Buenos Aires, Argentina
gonzaloavilaalterach@gmail.com

Resumen—Este trabajo consiste en el diseño e implementación de un sistema para reconocer palabras pronunciadas por el usuario, utilizando como plataforma a la EDU-CIAA, que realiza el procesamiento de los datos que son capturados por un micrófono. El proyecto es de gran relevancia, al no existir alternativas libres y gratuitas que sirvan para reconocer palabras, y que se presenten completas y listas para usar: por lo general es necesario tener datos de entrenamiento y realizar configuraciones complejas. El desarrollo está basado en Modelos Ocultos de *Markov* (*HMM*) e incluye los datos de fonemas en español y un módulo de pronunciación. Esto permite que un usuario pueda utilizar la biblioteca incluso sin tener conocimientos de cómo funciona internamente. En este artículo se describen consideraciones tenidas en cuenta a la hora de diseñar e implementar el sistema, tanto de *hardware* como de *software*.

I. INTRODUCCIÓN

I-A. Objetivo del proyecto

La finalidad del proyecto es desarrollar una biblioteca que permita reconocer palabras aisladas, en idioma español, esperando que el mismo sea de utilidad para toda persona que desee integrar un sistema de estas características en un diseño ya existente.

Se encuentran múltiples aplicaciones donde se podría integrar el proyecto: dentro de la interfaz de navegación de un sistema automotriz, para automatización del hogar, en aplicaciones industriales donde sea impráctico tener un teclado, en proyectos para personas con discapacidades motrices, entre otras.

La plataforma principal que se usó es la *EDU-CIAA-NXP*. La misma posee un microcontrolador *dual-core ARM Cortex M4/M0*, con 136 KB de memoria *RAM*, 1 MB de memoria *flash* y unidad de punto flotante. Las etapas correspondientes al reconocimiento propiamente dicho se realizan en la plataforma mencionada, por lo que no se requiere *software* adicional ejecutándose en una computadora.

Se encontraron diversos trabajos anteriores[1] que describen implementaciones de reconocedores de palabras en microcontroladores de la misma familia, utilizando en algunos casos otros algoritmos. También existen algunas bibliotecas abiertas[2], pero no poseen *ports* al microcontrolador utilizado.

Los modelos estadísticos necesarios para hacer funcionar correctamente al sistema, son generados por una computadora (proceso de entrenamiento), ya que requiere tener acceso a una cantidad considerable de datos y de tiempo de procesa-

miento, y no podría ser implementado eficientemente en un microcontrolador.

El sistema implementado sirve para vocabularios relativamente pequeños (menos de 30 palabras), es independiente del hablante (los modelos no están adaptados a una sola persona), y reconoce palabras aisladas, separadas con silencios.

I-B. Modelos Ocultos de *Markov*

Los modelos ocultos de *Markov* están basados en cadenas de *Markov*, en las que cada estado tiene posibilidad de emitir un símbolo estocástico, y pueden ser usados para modelar muchos sistemas físicos en los cuales solamente es observable una salida y no se conoce el estado interno del mismo. De ahí proviene el nombre de oculto: no es conocida la secuencia de estados pero sí son conocidos los valores emitidos por cada uno de ellos.

En particular, el problema que resuelve el reconocedor es extraer o estimar información sobre las secuencias de estados posibles a partir de dichas salidas conocidas.

I-C. Obtención de características

Las *features* o características son vectores que se extraen de los datos a analizar, con la intención de reducir las dimensiones del problema, manteniendo alguna relación que permita clasificar correctamente los datos. La elección de las *features* determina cuán complejo será calcularlos, y qué tan robusto será el reconocimiento ante ruido y la presencia de reverberaciones.

Los coeficientes utilizados para este trabajo fueron los denominados *Mel-Frequency Cepstral Coefficients (MFCC)*, que están inspirados en cómo responde el órgano de Corti del oído humano a los sonidos. Su comportamiento se puede modelar como diversos filtros adaptados a varias frecuencias, más espaciados en los agudos. También se tiene en cuenta la respuesta no lineal del oído ante la amplitud.

En general, el método de cálculo de dichos coeficientes *MFCC* es el siguiente:

1. Dividir la señal en bloques de 20 a 40 ms.
2. Estimar la densidad espectral de potencia de cada uno de los bloques, por ejemplo, haciendo una *Transformada de Fourier Discreta (DFT)* y hallar el módulo al cuadrado de cada uno de los valores.
3. Realizar una ponderación de los valores del espectro, usando filtros triangulares, separados en escala *Mel*,

como los de la figura 1. Este paso se denomina *binning* porque se acumulan los valores en diversos *bins* (del inglés, contenedores o tachos).

4. Tomar logaritmo al valor de cada *bin*, de forma similar a cómo responde el oído ante variaciones de amplitud.
5. Aplicar la *Transformada Coseno Discreta (DCT)* y usar los primeros coeficientes como características. Este paso intenta descorrelacionar los valores de cada *bin* y además comprime la información útil en menos coeficientes: una de las propiedades principales de esa transformada.

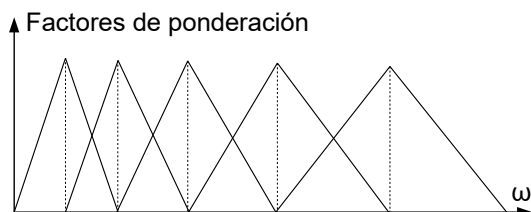


Figura 1. Ejemplo de 5 filtros triangulares en escala *Mel*.

Además, en la práctica, para realizar el primer paso se suelen tomar bloques superpuestos, logrando de esta forma minimizar los efectos de posibles transitorios, aplicando una ventana de *Hamming* a cada bloque. También se suelen agregar coeficientes *Delta* y *Aceleración* que codifican las estimaciones de la tasa de variación (velocidad) y aceleración de los coeficientes estáticos.

I-D. Modelos Utilizados

Para simplificar la cantidad de parámetros a estimar, se supone que los *features* se pueden modelar como variables aleatorias *Gaussianas* con matriz de covarianza diagonal. En particular, se hizo más complejo al modelo suponiendo que los estados emiten mezclas de *Gaussianas*.

Además, debido a la evolución temporal de las señales de voz, se acostumbra el uso de modelos con múltiples estados, correspondiendo cada uno a un segmento cuasiestacionario de la señal. Una topología de izquierda a derecha, como la de la figura 2, es la más utilizada en reconocedores de habla. En esta topología solo se permiten transiciones al mismo estado o al siguiente, si cambian las características del segmento de habla.

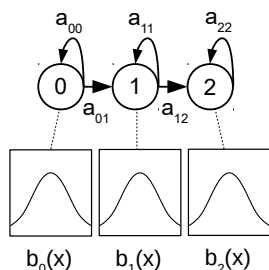


Figura 2. Modelo con topología de izquierda a derecha, con 3 estados, cada uno de los cuales tiene asociada una densidad de probabilidad $b(x)$.

En este tipo de topologías es importante elegir qué representa un modelo, y cuántos estados se le asignan a cada uno [3]. En el caso del trabajo implementado, se utilizan modelos para cada fonema, con 3 estados emisores para cada uno.

Se agrega también un modelo adicional para representar los posibles silencios presentes al comienzo y final de cada pronunciación, con la posibilidad de transicionar desde el tercer estado al primero y viceversa, como el de la figura 3.

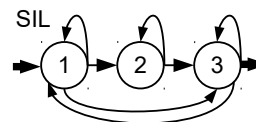


Figura 3. Modelo del silencio, agregado al principio y al final de las palabras.

El entrenamiento de los modelos se realizó usando la base de datos *latino40* [5], que fue provista por el *Grupo de Procesamiento del Habla* de la *FIUBA*. Dicha base de datos contiene 5000 grabaciones hechas por 20 hombres y 20 mujeres de diversos países de Latinoamérica, leyendo noticias de un diario. Se usó el kit de herramientas *HTK (Hidden Markov Model Toolkit)* para generar los modelos. La información de los procedimientos utilizados [4] está fuera del alcance de este artículo.

I-E. Aplicación del algoritmo de Viterbi

El algoritmo de Viterbi permite encontrar la probabilidad de la secuencia de estados (camino) más probable, para un modelo y observaciones Y dadas. El mismo utiliza programación dinámica para reducir la complejidad temporal, no requiriendo computar las probabilidades de todos los caminos, con la estrategia de guardar la secuencia óptima hasta cada estado, para todo instante.

Dada la pronunciación de una palabra, el algoritmo de reconocimiento usa el siguiente procedimiento para elegir la más probable:

- Por cada palabra posible se debe:
 - Generar un modelo expandido, formado por los fonemas que componen la palabra, incluyendo el modelo de silencio tanto al principio como al final. Este paso se observa en la figura 5.
 - Ejecutar el algoritmo de *Viterbi*, encontrando la probabilidad del camino más probable para ese modelo expandido. Este paso se observa en la figura 6.
- Finalmente, se debe elegir la palabra cuyo camino más probable tenga mayor probabilidad.

Este procedimiento también permite obtener la *enésima* palabra más probable, si se almacenan las probabilidades para cada palabra en un vector y luego se ordenan según su probabilidad, manteniendo los índices originales en otro vector.

II. DISEÑO E IMPLEMENTACIÓN

El diagrama de bloques del sistema completo puede ser encontrado en la figura 4.

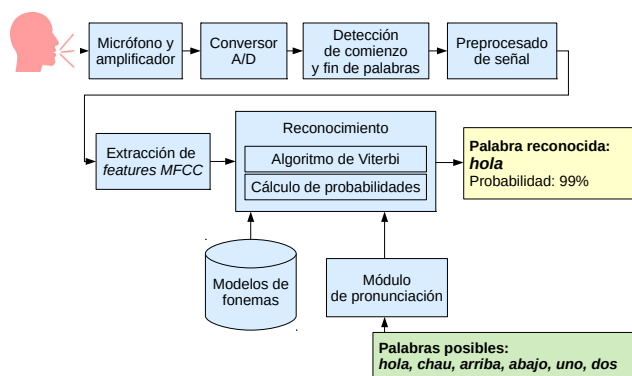


Figura 4. Diagrama de bloques del sistema implementado.

Para reducir el tiempo de procesamiento del microcontrolador, se diseñó la biblioteca precalculando la mayor cantidad de variables posibles, almacenándolas en tablas. Para ello, se desarrollaron otros programas, que a partir de los modelos (en el formato del *toolkit HTK*) las generan automáticamente y las guardan en archivos `.c` y `.h`.

El código completo, tanto de la biblioteca como de la aplicación de prueba y los tests de validación implementados puede ser descargado de forma gratuita [12]. Se dividió a la biblioteca en varios módulos, cuyo detalle de implementación se describe a continuación.

II-A. Módulo principal

El módulo principal se encarga de inicializar la biblioteca, de controlar la máquina de estados que detecta comienzos y fines de palabras, y entre otras cosas, de guardar las muestras capturadas en un *buffer* circular.

Cuando es necesario reconocer, también se encarga de dividir a las muestras en bloques, llamar a funciones del módulo que calcula los coeficientes *MFCC* de cada uno de ellos, para que luego el módulo de *Viterbi* ejecute ese algoritmo, y posteriormente el módulo principal ordene las palabras según su probabilidad.

La estrategia utilizada para detectar comienzos y fines está basada en la energía, porque es una de las variables más fáciles de calcular. Esta estrategia usa una máquina de tres estados. La misma considera el inicio de palabra cuando la energía de los últimos 100 ms supera un determinado umbral. El final de palabra se detecta cuando la energía es menor a otro umbral, siempre que durante un tiempo adicional después no vuelva a subir. Si esto sucediera, se considera que se trata de la misma palabra. De esta forma se consigue que palabras que tengan fonemas con silencios de longitud considerable sean detectadas correctamente.

II-B. Módulo MFCC

El módulo *MFCC* se encarga de calcular los coeficientes de un determinado bloque. Principalmente convierte las muestras a punto flotante, le resta su *offset* de continua y realiza la

secuencia descrita en I-C. Una explicación más detallada de cómo realiza cada paso el *toolkit HTK* se utilizó para implementarlo [10].

Este módulo fue realizado usando las funciones de *CMSIS-DSP* [9]: una biblioteca que permite el procesamiento de señales de forma eficiente en microcontroladores *Cortex-M*. La misma permitió realizar la *FFT* requerida y el cálculo del módulo al cuadrado de cada elemento de dicha transformada.

Los coeficientes de la ventana de *Hamming*, los de los filtros triangulares y los de la *DCT* se precalcularon, almacenándolos en tablas de *floats* de 32 bits en la memoria *flash* del microcontrolador.

II-C. Módulo con datos del Modelo

Este módulo contiene tablas con constantes, correspondientes al modelo de cada fonema y al silencio. Se genera automáticamente a través de un programa (`gen_modelo`) que convierte los modelos del formato usado por el *toolkit HTK* a un código C que posee los mismos valores en estructuras constantes.

Se realizaron algunas optimizaciones para reducir la memoria requerida:

- De las matrices de transición de los modelos sólo se almacenaron los valores distintos de cero, precalculando su logaritmo.
- Se invirtieron los valores de las varianzas, para que sólo sean requeridas multiplicaciones y no divisiones (en el microcontrolador tardan hasta 14 veces más).
- De cada mezcla de *Gaussianas* se precalculó la constante que no depende de las observaciones.

II-D. Módulo de probabilidades

El módulo calcula las probabilidades de observación de un vector de coeficientes, dado un determinado estado de un fonema. Cada estado tiene un mezcla de *Gaussianas* como función de densidad de probabilidad. Todas las *Gaussianas* tienen su matriz de covarianza diagonal, para de esta forma simplificar los cálculos a realizar. Además, todas las funciones trabajan calculando directamente el logaritmo de las probabilidades, para evitar calcular exponenciales.

Las expresiones usadas pueden ser obtenidas partiendo de la función de densidad de probabilidad de una *Gaussiana* multivariable de N dimensiones, tomando logaritmo y suponiendo a su matriz de covarianza $\Sigma \in R^{N \times N}$ como diagonal con elementos $\Sigma_{ii} = \sigma_i$:

$$\log(f(\vec{x})) = \log((2\pi)^{-N/2} |\Sigma|^{-1/2}) - \frac{1}{2} \sum_{i=1}^N (x_i - \mu_i)^2 \sigma_i^{-1} \quad (1)$$

El primer término no depende de las observaciones \vec{x} , por lo que puede ser precalculado y almacenado como una constante.

Posteriormente, se calcula la densidad de probabilidad de una mezcla de M *Gaussianas*, con pesos ϕ_j (que suman 1):

$$g(\vec{x}) = \sum_{j=1}^M \phi_j f_j(\vec{x}) \quad (2)$$

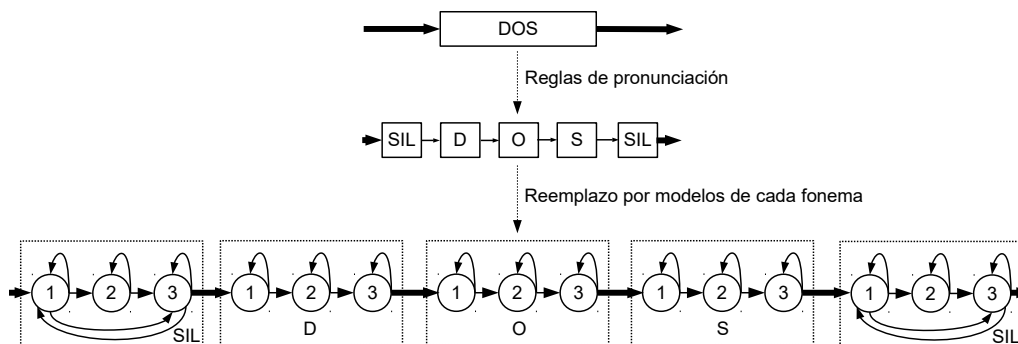


Figura 5. Ejemplo del paso de expansión: uso de reglas de pronunciación para convertir palabras en fonemas y el reemplazo por sus modelos respectivos.

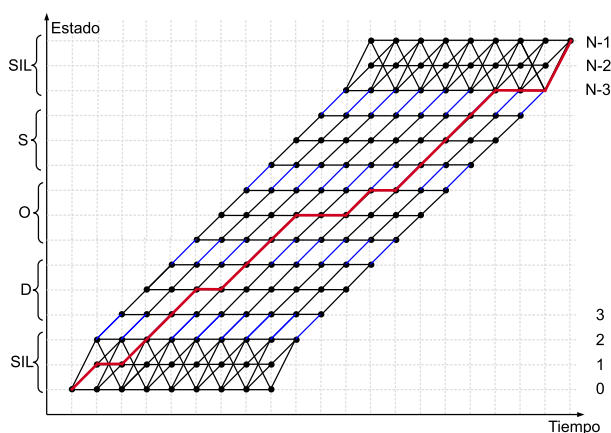


Figura 6. Ejemplo de resultado del algoritmo de Viterbi, para la palabra *dos*. Se marcaron con azules las transiciones entre un fonema y otro, y con rojo un posible camino.

Tomando logaritmo a (2) y operando con la sumatoria se obtiene:

$$\log g(\vec{x}) = \log \left(\sum_{j=1}^M \exp(\log(\phi_j) + \log(f_j(\vec{x}))) \right) \quad (3)$$

El término $\log(\phi_j)$ puede ser precalculado y $\log(f_j(\vec{x}))$ se puede obtener de (1). La expresión se puede reescribir como el logaritmo de una suma de exponentes:

$$\log g(\vec{x}) = \log(\exp(T_1) + \exp(T_2) + \exp(T_3) + \dots) \quad (4)$$

La misma se puede evaluar secuencialmente reescribiéndola como múltiples operaciones $\log\text{-sum-exp}(\log(e^a + e^b))$. Usando un truco matemático[6] se puede realizar esta operación y calcular las probabilidades de cada una de las *Gaussianas* que componen la mezcla e integrarlas sin necesidad de realizar el producto de las probabilidades.

II-E. Módulo de Viterbi

La implementación del algoritmo de *Viterbi* fue realizada con probabilidades en espacios logarítmicos, para evitar pro-

blemas numéricos [7]. A partir de la fórmula recursiva de *Viterbi*:

$$\Phi_t(j) = b_j(y_t) \max_{1 \leq i \leq N} a_{ij} \Phi_{t-1}(i) \quad (5)$$

Con $1 \leq j \leq N$ y $1 \leq t \leq T$, siendo N la cantidad de estados totales, T la cantidad de salidas observadas, b las funciones de densidades de probabilidad de cada estado y a_{ij} la probabilidad de transición de un estado a otro.

En lugar de almacenar los Φ , se almacenan los logaritmos de dichos elementos, llamándolos $\delta_t(j)$.

Las condiciones iniciales obligan a que valgan $\delta_1(i) = 0$ para $i = 1$ y $\delta_1(i) = -\infty$ para $i \neq 1$. Es decir, el primer estado tiene probabilidad máxima.

Tomando logaritmo a (5) y considerando la propiedad de que éste es monótono creciente, se puede intercambiar el orden del máximo y el logaritmo, obteniendo:

$$\delta_t(j) = \log \left(b_j(y_t) \max_{1 \leq i \leq N} a_{ij} \Phi_{t-1}(i) \right) \quad (6)$$

$$= \log(b_j(y_t)) + \max_{1 \leq i \leq N} (\log(a_{ij}) + \delta_{t-1}(i)) \quad (7)$$

Por lo tanto, es conveniente calcular directamente el logaritmo de las probabilidades de observación b_j , como se describió en la sección anterior. El logaritmo de las probabilidades de transición a_{ij} puede ser precalculado.

Como la recursión solamente depende del paso anterior no es necesario almacenar todos los δ en memoria. En la implementación realizada únicamente se almacenan los mismos para el instante de tiempo actual y el anterior. Luego de cada iteración se cambian los roles de los vectores, permitiendo reducir considerablemente el uso de memoria.

II-F. Módulo de pronunciación

Este módulo convierte una palabra en idioma español en su pronunciación, es decir, una secuencia de fonemas. Al utilizar reglas, y no un diccionario no hay restricción en las palabras posibles, y éstas pueden ser cambiadas incluso en tiempo de ejecución.

Las reglas de pronunciación fueron obtenidas basándose en prueba y error, y el documento con reglas de pronunciación

[11]. Se va tomando de a una las letras de la palabra que se desea pronunciar. Se emiten fonemas en función de la letra tomada, la anterior, la siguiente y la inmediata posterior.

II-G. Interfaz con aplicación de usuario

Con el objetivo de lograr que sea fácil integrar la biblioteca a cualquier tipo de aplicación, se definió una API simple con pocas funciones:

- Inicialización: se le pasa un vector de *strings* con las palabras que se desean reconocer.
- Almacenamiento de muestra: cada vez que se captura una nueva muestra es necesario llamar a esta función, y pasarle el valor de la misma.
- Detección periódica: cada 100 ms aproximadamente, la aplicación del usuario deberá llamar a esta función, que se encarga de verificar si comenzó o terminó una palabra.
- Reconocimiento: luego de detectar el fin de una pronunciación, la aplicación del usuario puede llamar a esta función para realizar el reconocimiento. La misma devuelve las N palabras más probables.

Asimismo, para lograr que la biblioteca sea más configurable, se definieron diversas *macros* modificables por el usuario.

II-H. Aplicación de prueba

La aplicación para probar la biblioteca desarrollada se diseñó a partir de los ejemplos *baremetal* de *LPCOpen* portados a la *EDU-CIAA-NXP* [8].

Se usó sobremuestreo para capturar la señal a 128000 muestras por segundo. Cada 8 datos capturados se hace un promedio de ellos y se los pasa a la biblioteca. De esta forma mejora la cantidad de *bits* efectivos y disminuye el *aliasing*, por actuar como un filtro pasabajos.

El código inicializa a la biblioteca (con las palabras que se desean reconocer), a la placa y al convertor analógico-digital. Este último es activado en modo ráfaga (*burst*), y se habilitan sus interrupciones.

Cuando suceden las interrupciones del convertor A/D, se acumulan los valores muestreados para armar el promedio. Además, periódicamente en el programa principal, se llama a la biblioteca para que revise si se detectó comienzo o fin de palabra. Si se detecta el fin, se comienza el reconocimiento, tras el cual se envían las palabras más probables a través de la UART.

II-I. Hardware de prueba

Para poder probar el sistema completo, se diseñó y construyó una placa amplificadora de micrófono, como se observa en la figura 7. La misma se realizó usando un amplificador operacional en configuración inversora, con ganancia de aproximadamente -100 (el valor no es crítico). La tensión de salida queda centrada respecto de la mitad de la tensión de alimentación. Se utilizó el amplificador operacional *Microchip MCP6002* que es *Rail-to-Rail*, para maximizar el rango dinámico de la señal a la entrada del convertor A/D. El amplificador operacional no utilizado fue conectado como está descripto por la hoja de datos, para minimizar el ruido

adicional inducido. El esquemático utilizado se observa en la figura 8. No se incluyó un filtro *antialias*, debido a que la combinación del micrófono utilizado (cuya curva de respuesta en frecuencia cae a partir de los 20 KHz), el amplificador operacional (que realimentado tiene un polo a aproximadamente 10 KHz) y el filtro promediador, hacen que el *aliasing* no sea tan significativo.

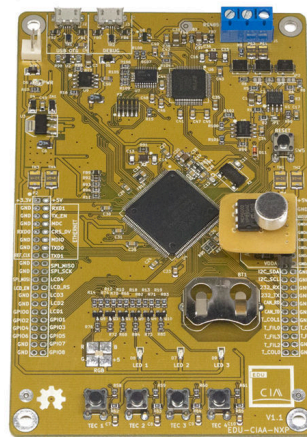


Figura 7. Sistema completo, *EDU-CIAA-NXP* con placa amplificadora de micrófono.

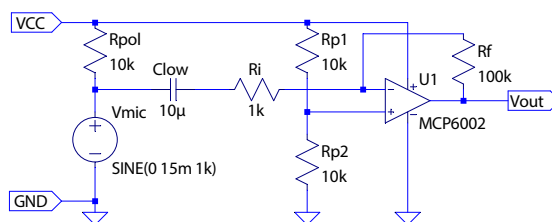


Figura 8. Circuito amplificador de micrófono electret implementado.

III. ENSAYOS Y RESULTADOS

III-A. Pruebas de validación implementadas

Se probó experimentalmente que el reconocimiento funciona correctamente. Usando datos de prueba compuestos por 150 pronunciaciones de dígitos, de un solo locutor, grabadas sin ruido y a una distancia correcta para no saturar el amplificador, se obtuvieron tasas de error cercanas al 2%. Como es de esperar, la tasa aumenta considerablemente con ruido de fondo, debido a que la estrategia usada para detectar comienzos y finales de palabras no es muy robusta.

Además de la biblioteca principal, a modo de validación de cada uno de los módulos de la misma, se desarrollaron diversos programas complementarios que permitieron:

- Comparar el cálculo de *MFCC* con los obtenidos mediante el *toolkit HTK*, resultando en diferencias menores al 0,01 en los coeficientes.

- Contrastar las tasas de error, con las del *toolkit HTK*, obteniendo tasas idénticas en ambos casos.
- Utilizar un diccionario de pronunciación en español para verificar que las reglas para convertir una palabra en sus fonemas funcionan correctamente.
- Realizar una aplicación de ejemplo para una computadora, que permita probar el reconocimiento sin necesidad de tener la plataforma *EDU-CIAA*.

III-B. Uso de recursos

Los valores de la tabla I fueron calculados utilizando 10 palabras y modelos para 28 fonemas, con 3 estados por fonema y mezclas de 16 *Gaussianas* por estado. Las *flags* de compilación se eligieron para utilizar de forma eficiente la unidad de punto flotante del microcontrolador.

Se calculó también el *Real Time Factor (RTF)*, que indica cuánto demora el reconocimiento en relación a la duración de la pronunciación, resultando en un valor de 2,02 ms por cada ms capturado. La función que más tiempo tarda es el algoritmo de *Viterbi*, que se ejecuta una vez por cada palabra posible. Si se agregasen más palabras sería conveniente optimizar esta función, para evitar que el sistema demorase mucho en reconocer. La captura de las muestras usa aproximadamente 15% del tiempo de *CPU*.

El *buffer* de audio es el que tiene asignada una mayor cantidad de memoria *RAM*, siguiéndole, el vector donde se almacenan los coeficientes *MFCC* del mismo. El primer banco de memoria *flash* contiene algunas constantes y las funciones, mientras que el segundo, es el que almacena el modelo.

Tabla I
RESUMEN DE USO DE MEMORIA

Banco de memoria	Bytes usados	Bytes totales	Porcentaje
Flash (A)	138 KB	512 KB	26,9 %
Flash (B)	422 KB	512 KB	82,4 %
RAM (1)	19 KB	32 KB	59,3 %
RAM (2)	32 KB	40 KB	80 %

IV. CONCLUSIÓN Y TRABAJO A FUTURO

Se logró implementar y evaluar el sistema propuesto, obteniendo valores dentro de lo esperado. Se validó experimentalmente el funcionamiento del mismo, detectando una clara influencia del ruido de fondo en las tasas de error, principalmente debido a la estrategia de detección de palabras utilizada. Además, se observó un tiempo de procesamiento aceptable y un consumo de memoria relativamente pequeño, que podría reducirse aún más, a cambio de un funcionamiento con mayores tasas de error.

Como trabajo a futuro, sería importante:

- Usar un conversor A/D externo de más *bits*, para reducir el ruido de cuantización, y agregar un filtro *antialias*.
- Implementar algún filtro adaptativo como el *LMS* y utilizar dos micrófonos para cancelar ruidos de fondo.

- Reemplazar el micrófono analógico por uno digital, que se comunique a través del protocolo *I²S*, reduciendo el tamaño y mejorando la calidad del audio.
- Optimizar más la implementación del algoritmo de *Viterbi* y el cálculo de las probabilidades.
- Usar *DMA* para reducir el tiempo de *CPU* usado para muestrear. Como alternativa, se pueden ejecutar las funciones de muestreo en el procesador *M0*, dejando el *M4* para el reconocimiento.
- Implementar un ejemplo para la *EDU-CIAA* usando el sistema operativo de tiempo real *FreeOSEK*. Para ello es necesario modificar el *driver* que maneja el *ADC* para soportar frecuencias de muestreo mayores a 1 kHz.
- Mejorar el algoritmo usado para detectar comienzos y fines de palabra, implementando uno basado en analizar la entropía de las muestras capturadas.
- Agregar soporte para realizar el procesamiento fuera del microcontrolador, enviando los coeficientes *MFCC* a través de *Internet*. De esta forma, se podrían utilizar técnicas de reconocimiento más actuales, que disminuyan las tasas de error y permitan reconocer oraciones.

AGRADECIMIENTOS

A la Ing. Patricia Pelle e Ing. Claudio Estienne, docentes de *Procesamiento del Habla*, por sus aportes en conocimiento valiosos para el desarrollo del proyecto.

Al Dr. Ing. Ariel Lutenberg, Ing. Juan Manuel Cruz e Ing. Jorge Graña, docentes de *Seminario de Sistemas Embebidos*, por el apoyo a este proyecto.

Al Ing. Pablo Ridolfi e al Ing. Martín Ribelotta, por su colaboración.

REFERENCIAS

- [1] Alvarez, Evin, Verrastro, *Implementation of a Speech Recognition System in a DSC*, 2016
- [2] Carnegie Mellon University, *PocketSphinx* [Online] Disponible: <https://github.com/cmusphinx/pocketsphinx>
- [3] Huang, Acero, Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*, 2001
- [4] Cambridge University Engineering Department, *The HTK Book*, 2006 [Online] Disponible: http://speech.ee.ntu.edu.tw/homework/DSP_HW2-1/htkbook.pdf
- [5] Entropic Research Laboratory, *LATINO-40 Spanish Read News*, 1995 [Online] Disponible: <https://catalog ldc.upenn.edu/LDC95S28>
- [6] Liangjie Hong, *Notes on Calculating Log Sum of Exponentials*, 2011 [Online] Disponible: <http://www.hongliangjie.com/2011/01/07/logsum/>
- [7] William H. Press, *Gaussian Mixture Models and EM Methods*, 2008 [Online] Disponible: <http://numerical.recipes/CS395T/lectures2008/11-GaussianMixtureModelsAndEM.pdf>
- [8] Martin Ribelotta, *Proyectos de ejemplo para la EDU-CIAA, usando LPCOpen*, 2015 [Online] Disponible: https://github.com/martinribeotta/ciaa_lpcopen_base
- [9] Keil, *CMSIS DSP Software Library*, 2015 [Online] Disponible: <http://www.keil.com/cmsis/dsp>
- [10] Danijel Koržinek, *PyHTK, HTK features in Python*, 2015 [Online] Disponible: <https://github.com/danijel3/PyHTK>
- [11] Wikcionario, *Reglas de pronunciación del español* [Online] Disponible: <https://es.wiktionary.org/wiki/Wikcionario:Referencia/ES/Pronunciacin>
- [12] Ávila Alterach, *Biblioteca liviana para reconocimiento de habla en español* [Online] Disponible: <https://github.com/gzalo/bla>

Control Adaptivo de un Sistema de Comunicación Inalámbrico

Narvaez Pablo

Facultad de Ingeniería, IESIING
Universidad Católica de Salta
Campo Castaños, Salta-capital. Argentina.
efeparo@yahoo.com.ar

Garcia Nicolas

Facultad de Ingeniería, IESIING
Universidad Católica de Salta
Campo Castaños, Salta-capital. Argentina.
hnge0594@gmail.com

Breslin Roberto

Facultad de Ingeniería, IESIING
Universidad Católica de Salta
Campo Castaños, Salta-capital. Argentina.
rbreslin@ucasal.net

Abstract - La demanda en el incremento de la capacidad de las redes inalámbricas ha motivado la reciente búsqueda hacia el desarrollo de sistemas que utilicen/reutilicen el espacio radioeléctrico de manera selectiva. Los sistemas de antenas inteligentes proporcionan oportunidades para incrementar la capacidad del mismo, tendientes a la obtención de calidad de servicio.

El presente proyecto consiste en la implementación de un sistema robótico cuyo efector final es una antena parabólica, el cual constará de un sensor de campo electromagnético, que indicará la intensidad de señal con el objetivo de que el brazo establezca la posición final de la antena según la mayor intensidad de señal del emisor y establecer el ángulo de arribo de la señal del emisor. El sistema podrá ser controlado de forma inalámbrica por una estación base, la cual recolectará los datos.

Keywords— Antena, Robótica, Intensidad de Campo electromagnético.

I. INTRODUCTION

Una antena inteligente es la combinación de un arreglo de antenas (arrays) con una unidad de Procesamiento Digital de Señales que optimiza los diagramas de transmisión y recepción dinámicamente en respuesta a una señal de interés en el entorno. Es aquella que, es capaz de generar haces directivos enfocados hacia una señal deseada, adaptándose a las condiciones radioeléctricas en cada momento.

Si bien en el mercado se encuentra con una gran variedad de antenas robóticas, sobre todo del tipo satelital, las mismas son costosas y con un software enlatado, no permitiendo modificar el mismo.

El propósito del proyecto consiste en la implementación del control de un brazo robótico de tres grados de libertad cuyo efector final es una antena parabólica. La misma constará de un sensor de campo magnético, el cual indicará la mayor intensidad de señal con el objetivo de que el brazo establezca la posición final de la antena según la mayor intensidad de señal

del emisor. El sistema podrá funcionar de manera autónoma o manual a través de una estación base, la cual coordinará y recolectará los datos de los distintos sistemas. El proyecto pretende establecer la vinculación entre la robótica y las telecomunicaciones con el objeto de dar el puntapié para el desarrollo de antenas inteligentes.

El sistema consta de los siguientes bloques constitutivos, como se puede observar en la figura 1.

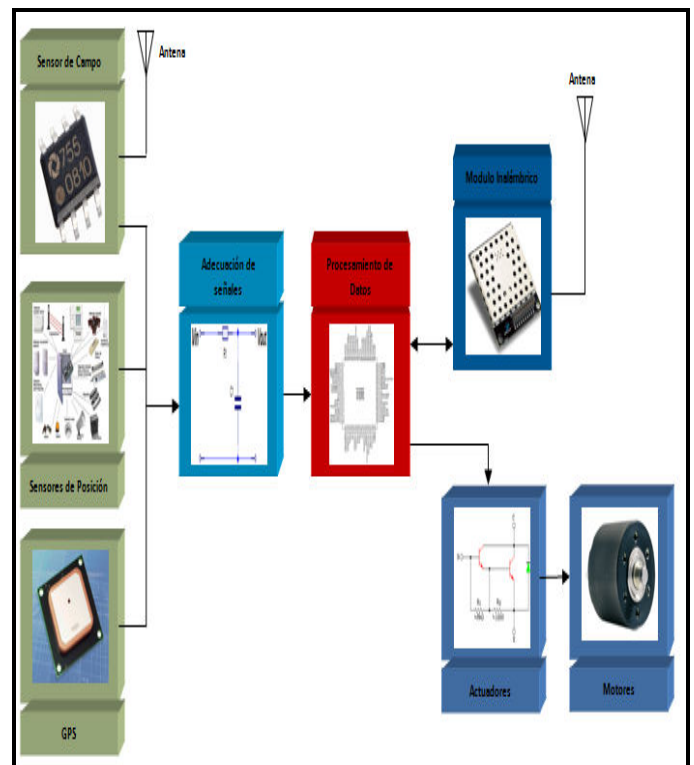


Figura 1. Sistema Completo.

II. SISTEMA ROBOTICO

La configuración Robótica seleccionada es del tipo brazo robótico de tres grados de libertad, ya que esta configuración permite mover la normal de la antena parabólica en todas direcciones. En la figura 2 se observa una foto del prototipo de brazo robótico.

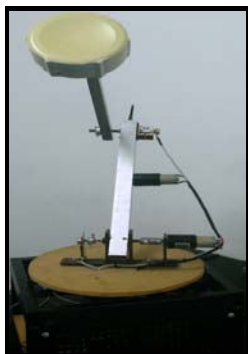


Figura 2. Brazo Robótico.

El cual consta de una base (Cintura) la cual tiene una rotación de 360° grados, la articulación del hombro que se desplaza en un ángulo de 180° grados y la articulación del codos que presenta un movimiento de aproximadamente 270°. En la figura 3 se puede observar las dimensiones y referencias antes mencionadas.

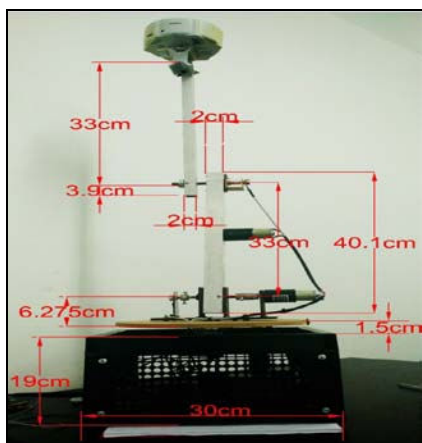


Figura 3. Dimensiones.

A. Espacio de trabajo

Para determinar el conjunto de puntos donde puede situarse el elemento terminal del robot se determinaron las ecuaciones del mismo en el plano y luego aplicando revolución se obtuvo el volumen de trabajo. A continuación se puede observar el espacio de trabajo figura 4.

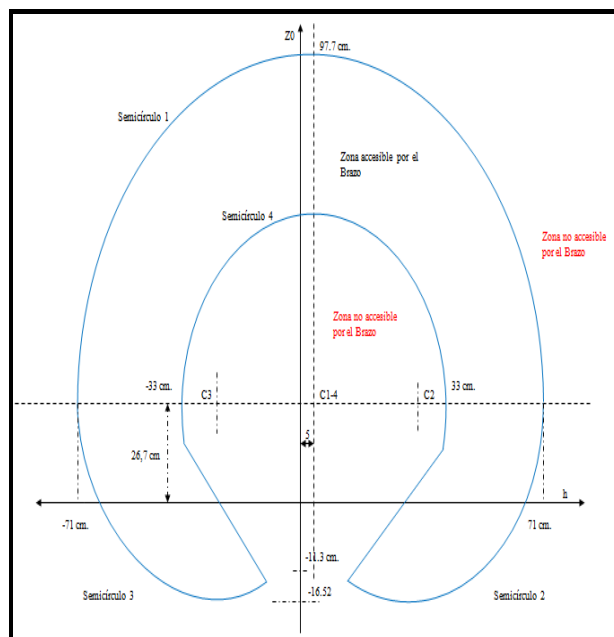


Figura 4. Espacio de Trabajo.

III. SENSORES

Los sensores empleados se pueden dividir según la siguiente categoría:

- De posición.
- De inclinación.
- De localización.
- De Intensidad de campo electromagnético.

A. Sensores de Posición

Los sensores empleados para medir el movimiento de cada articulación es un sensor del tipo potenciométrico multivuelta, el cual varía su resistividad según el movimiento de su eje, como se puede observar en la fig. 5.



Figura 5 Sensores potenciométricos.

B. Sensores de Inclinación

Además de los sensores de posición, se optó por agregar un sensor de inclinación el cual permite medir en los tres ejes (x,y,z) los grados de inclinación en base a la normal del chip.

El mismo se agrego como un método más de control de la posición para asegurar la posición final de la antena. El modulo empleado es MMA7660 el cual es un acelerómetro de tres ejes, con salidas de datos mediante comunicación I²C.

Los datos de inclinación por eje se pueden acceder para su lectura cada un determinado periodo de tiempo que se puede configurar en el modulo inclinometro, los mismos se acceden a través de registros internos del modulo.

C. Sensores de Localización

Para poder conocer la ubicación exacta del sistema, se emplea un modulo GPS para obtener su ubicación, así como también la hora y el día. Este mismo proporcionara el cálculo del ángulo de arribo de la señal RF en referencia a la ubicación de la antena como así también de la estación central. Esto permite referenciar cada sistema a la base y con ello referenciar el ángulo de arribo al sistema o la base. El modulo empleado es el smart GPS antenna A1035-d el cual posee comunicación serie y el formato de sus datos es según el estándar NMEA. La velocidad de transmisión empleada para la conexión del módulo con el microcontrolador es de 4800 baudios.

Los datos enviados al microcontrolador son la hora UTM, la latitud, la longitud, la fecha, la altitud, cantidad de satélites y el fix. Para lo cual se conforma un trama de 28 bytes de longitud.

D. Sensor de Energía RF

Para poder medir la intensidad de campo electromagnético se emplea el modulo LT5534 el cual es un detector de potencia de señales RF en el rango de 50mhz a 3ghz. El mismo con una antena dipolo en la frecuencia de 2.5ghz (wifii) nos permite medir la intensidad de campo electromagnético de las señales wifii.

El sensor proporciona una señal de salida en voltaje proporcional a la intensidad de campo electromagnético en dbm según la frecuencia como se puede observar en la figura 6[6].

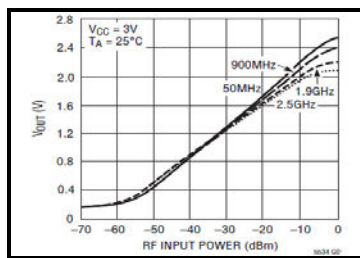


Figura 6 señal de salida.

IV. PROCESAMIENTO DE DATOS

El sistema de procesamiento de datos está conformado entorno a un microcontrolador de 8 bits de Microchip, de la gama media el PIC18F8722, el cual se selecciono ya que cumplía con los requerimientos del sistema y con la cantidad

necesaria de módulos internos para el manejo de los distintos sensores y módulos que debe controlar.

El sistema debe realizar múltiples tareas como se ilustran en la figura 7[1].

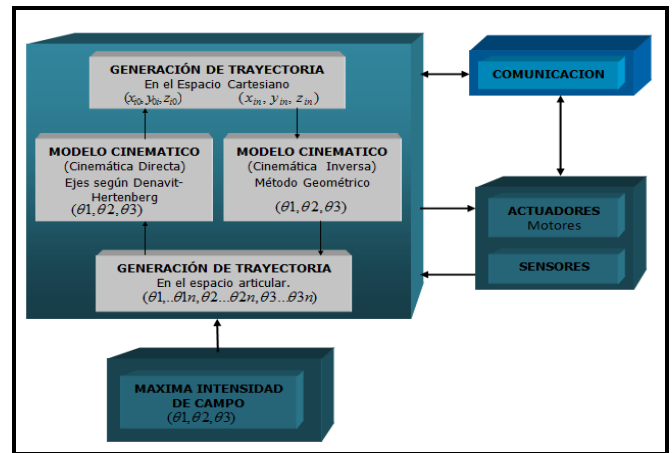


Figura 7. Tareas.

A. Lectura de Sensores

El microcontrolador realiza la lectura de las señales provenientes de los sensores de distintas formas, para los sensores de posición y de campo emplea el conversor AD interno para convertir los voltajes de entrada en señales digitales, en los sensores de inclinación el microcontrolador emplea el protocolo de comunicación I²C obteniendo una trama de datos (punto 3.2) la cual debe ser decodificada por el microcontrolador para obtener los datos del modulo y para el sensor GPS el microcontrolador emplea el protocolo de comunicación RS232 obteniendo una trama de datos (punto 3.3) del modulo la cual debe ser decodificada para obtener los datos del mismo.

Una vez obtenidos los datos digitales de cada sensor, el microcontrolador debe convertir dichos datos en las siguientes unidades estándares:

- Posición → Grados(θ_i)
- Campo → Intensidad(dbm)
- Inclinometro → Grados(θ_n)
- GPS → Hora, Grados

Las señales digitales adecuadas a la unidades estándares son empleadas en los siguientes bloques para realizar las tareas.

B. Máxima intensidad de campo Electromagnético

Para obtener las coordenadas de la máxima intensidad de campo electromagnético se implementara un método de optimización matemática perteneciente a las técnicas de búsqueda local, el algoritmo de búsqueda Tabú[2][3] adaptado a la capacidad de computo del sistema. El cual consiste en realizar un barrido de la superficie mediante cuadrantes, el cual comienza la búsqueda guardando en la memoria las posiciones de máxima intensidad de campo electromagnético, las cuales se

comparan luego y se realiza un nuevo barrido en un cuadrante que contiene los puntos de mayor intensidad de campo electromagnético, guardando los puntos de máxima intensidad, comparándolos nuevamente y repitiendo el proceso de barrido hasta alcanzar la condición de parada.

Debido a que la intensidad de la señal puede variar por las condiciones del entorno o simplemente por el tráfico de datos, el algoritmo tabú[2][3] se debió modificar censando la posición máxima alcanzada y realizando una nueva búsqueda en un entorno reducido sin pérdida de intensidad mínima del campo. Como se observa en la figura 8[2][3], los datos así obtenidos (grados) luego son empleados en la generación de trayectoria directa para calcular las coordenadas en el espacio cartesiano y con ellas obtener el ángulo de arribo de la señal de máxima intensidad de campo electromagnético.

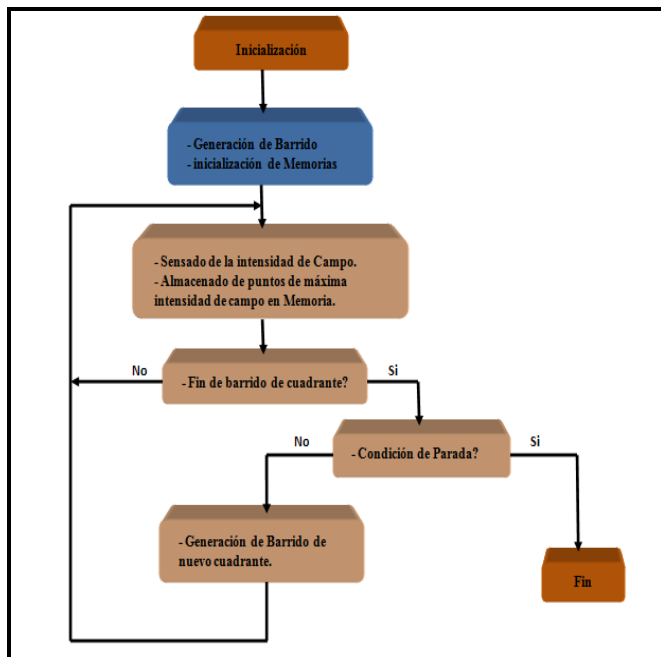


Figura 8. Algoritmo de detección de máxima intensidad de campo Electromagnético.

C. Generación de Trayectoria

La generación de trayectoria está compuesta por los siguientes bloques constitutivos como se pueden observar en la figura 9[1].

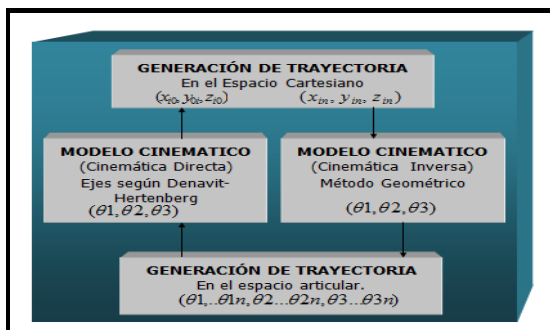


Figura 9. Generación de Trayectoria.

En la misma se muestran los dos caminos distintos que pueden seguir los datos.

En un caso los datos provenientes del sensor de intensidad de campo electromagnético, se ingresan al bloque de generación de trayectoria articular, continúan por el modelo cinemático directo y finalmente ingresan al bloque de generación de trayectoria cartesiana.

En el otro sentido los datos provienen del bloque de comunicación e ingresan al bloque de generación de trayectoria cartesiana, continúan por el modelo cinemático inverso y finalmente ingresan al bloque de generación de trayectoria articular.

A continuación se describe la función de cada bloque.

1) Generación de Trayectoria Cartesiana

La Generación de Trayectoria es el proceso mediante el cual se aproxima el camino deseado. A tal fin se utiliza una serie de puntos entre las coordenadas del punto inicial y final, que pueden ser calculados a partir de funciones polinomiales, lineales, etc.

Este bloque es el encargado de realizar una generación de trayectoria en el espacio cartesiano, el cual consiste en asignarle una función matemática a dos puntos del espacio cartesiano, el inicial y el final. Para aproximar la trayectoria. Además se generan una cantidad de puntos intermedios entre el punto inicial y el final con la función matemática seleccionada.

La función matemática seleccionada para este proyecto consistió en la función lineal y con una generación de puntos intermedios en función del tiempo variable según un espaciado mínimo entre puntos.

2) Modelo Cinemático (Directo)

La cinemática directa consiste en determinar cuál es la posición y orientación del extremo final del sistema robótico, con respecto a un sistema de coordenadas que se toma como referencia, conocidos los valores de las articulaciones y los parámetros geométricos de los elementos del robot.

En general, un sistema robótico está formado por eslabones unidos por articulaciones, de forma que cada par articulación-eslabón constituye un grado de libertad.

Debido a esto se puede establecer un sistema de referencia fijo situado en la base del robot y describir la localización de cada uno de los eslabones con respecto a dicho sistema de referencia.

El proceso consiste en fijar un sistema de coordenadas a cada eslabón, que se moverá con él de acuerdo a un conjunto de normas fijas.

Este bloque es el encargado de realizar dicha tarea empleando el algoritmo de Denavit-Hartenberg[1].

A partir de la disposición de ejes por articulación y empleando relaciones trigonométricas por eje y referenciando a la base se obtuvieron las siguientes ecuaciones espaciales:

$$x = \cos(\theta_b) * [\cos(\theta_h) * 33 + \cos(\theta_{cb}) * 38] \quad (1)$$

$$y = (\sin(\theta_b) * [\cos(\theta_h) * 33 + \cos(\theta_{cb}) * 38]) + 5 \quad (2)$$

$$z = 26,775 + (\sin(\theta_b) * 33) + (\sin(\theta_{cb}) * 38) \quad (3)$$

Donde

$$\theta_{cb} = \theta_c - \theta_h \quad (4)$$

3) Modelo Cinemático (Inverso)

El objetivo del problema cinemático inverso consiste en encontrar los valores que deben adoptar las coordenadas articulares del sistema robótico (θ_b , θ_h y θ_c) para que su extremo se posicione y oriente según una determinada localización espacial (x,y,z).

En el problema cinemático inverso el procedimiento de obtención de las ecuaciones es fuertemente dependiente de la configuración del sistema robótico. Si se consideran sólo los tres primeros grados de libertad, estos tienen una estructura planar, esto es, los tres primeros elementos quedan contenidos en un plano. Esta circunstancia facilita la resolución del problema.

Los métodos geométricos permiten obtener normalmente los valores de las primeras variables articulares, que son las que consiguen posicionar el sistema robótico. Para ello utilizan relaciones trigonométricas y geométricas sobre los elementos del sistema, recibiendo el nombre de método geométrico.

El cual en sí se basa en encontrar un suficiente número de relaciones geométricas en las que intervendrán las coordenadas del extremo del sistema robótico, sus coordenadas articulares y las dimensiones físicas de sus elementos.

Para el cálculo de las distintas coordenadas articulares y debido a las infinitas posibilidades de configuraciones del sistema robótico que pueden alcanzar el punto espacial se emplearon algoritmos recursivos con una condición inicial de arranque, basada en el espacio de trabajo y una final de parada correspondiente a una tolerancia de los datos.

Con motivo de que el eje auxiliar 1 presenta la misma orientación que el de la base con la única diferencia que se encuentra desplazado una cierta longitud, se puede considerar que el desplazamiento del ángulo θ_b se realiza en el eje de la base.

Debido a esto y como las coordenadas espaciales están referenciadas al eje de la base los valores de x_f e y_f permiten calcular dicho ángulo empleando la siguiente ecuación(5):

$$X_f * \sin(\theta_b) = Y_f * \cos(\theta_b) \quad (5)$$

Para facilitar el cálculo de la ecuación (5) se dividió en cuadrantes los posibles valores de ángulos de θ_b según el valor de x_f e y_f como se puede observar a continuación:

- Si y_f es positivo y x_f es positivo θ_b puede variar desde 0° a 90° .
- Si y_f es positivo y x_f es negativo θ_b puede variar desde 90° a 180° .

- Si y_f es negativo y x_f positivo θ_b puede variar desde 180° a 270° .
- Si y_f es negativo y x_f negativo θ_b puede variar desde 270° a 360° .

De esta manera se establece la condición de arranque del algoritmo para el cálculo de θ_b y la condición de parada es una tolerancia al error de 0,01. Como se puede observar en la siguiente figura 10.

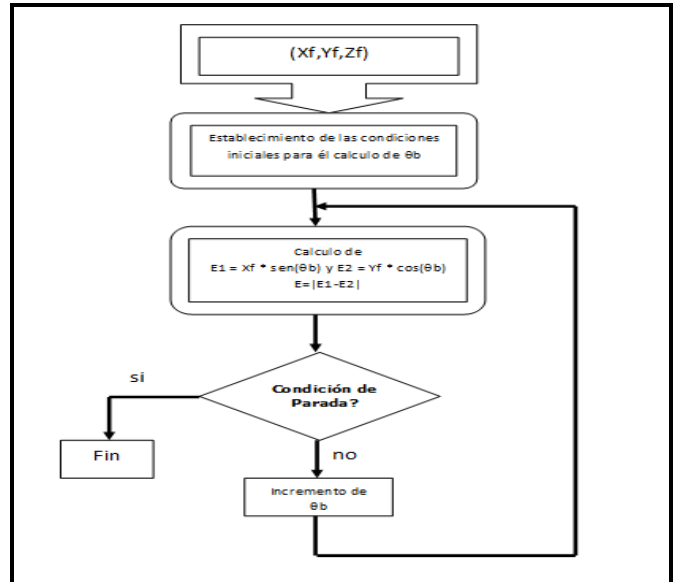


Figura 10. Algoritmo de cálculo de θ_b .

Una vez determinado el ángulo de base θ_b se procede a calcular los ángulos de hombro y codo para ello se emplean las ecuaciones (1), (2) y (3).

Como existen infinitos valores de ángulos que satisfacen dichas ecuaciones se estableció una condición de arranque según cuadrantes limitando el rango de búsqueda de los ángulos mediante los valores de X_f e Y_f como sigue:

- Si Y es positivo y el modulo de $|XY| < 38\text{cm}$ el rango de búsqueda es $\theta_h=90^\circ$ y θ_c puede variar desde 12° a 180° .
- Si Y es positivo y el modulo de $38\text{cm} < |XY| < 38 + 33\text{cm}$ el rango de búsqueda es $0^\circ < \theta_h < 90^\circ$ y θ_c puede variar desde 12° a 180° .
- Si Y es negativo y el modulo de $|XY| < 38\text{cm}$ el rango de búsqueda es $\theta_h=90^\circ$ y θ_c puede variar desde 180° a 270° .
- Si Y es negativo y el modulo de $38\text{cm} < |XY| < 38 + 33\text{cm}$ el rango de búsqueda es $0^\circ < \theta_h < 90^\circ$ y θ_c puede variar desde 180° a 270° .

Empleando esta condición de arranque se procedió a calcular los valores de las ecuaciones (1), (2) y (3) y calculando el error con los valores finales de X , Y y Z , si dicho valor es inferior al 0,01% se toma como válidos dichos valores de los ángulos de hombro y el codo. Como se puede observar en el siguiente gráfico 11 del algoritmo para el cálculo de θ_h y θ_c .

Figura 12. Trama de datos.

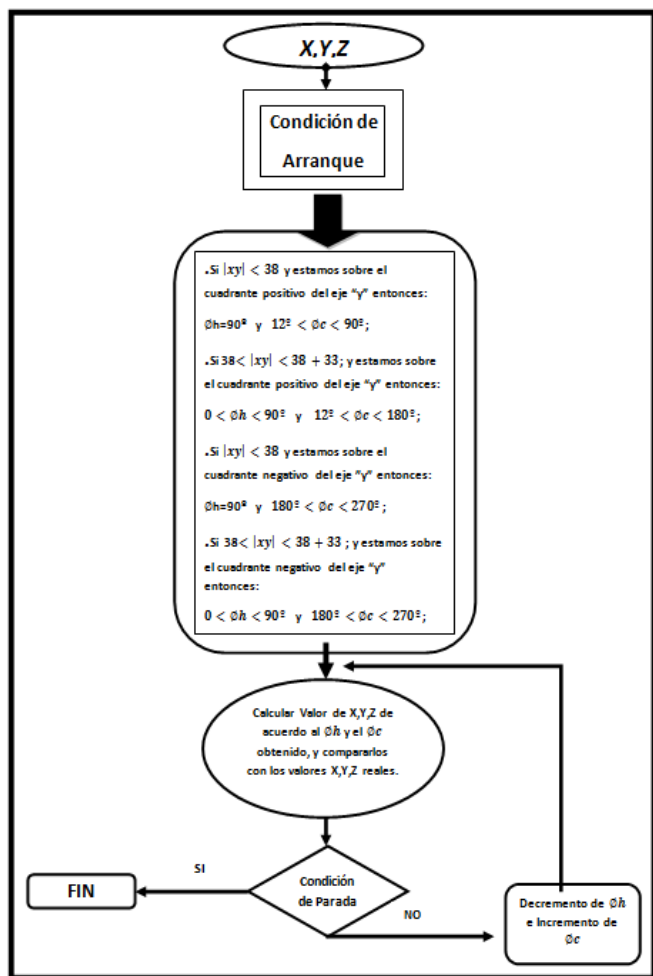


Figura 11. Algoritmo de cálculo de θ_h y θ_c .

4) *Generación de Trayectoria Articular*

Al igual que con la generación de trayectoria cartesiana este bloque es el encargado de realizar una generación pero en el espacio de cada articulación, empleando una función lineal para aproximar los puntos generados, a la su vez son generados en función del tiempo.

D. *Comunicación*

Este bloque es el encargado de realizar la comunicación de los datos de forma inalámbrica con la estación base o con otras sistemas para la transferencia de información como así también de datos de control.

Para poder realizar dicha transacción de datos se empleo una trama de comunicaciones como se observa en la siguiente figura 12.

Destino	Origen	Código	Datos	Datos	FCR	Fin
---------	--------	--------	-------	-------	-----	-----

La trama está compuesta por una cabecera destino y otra origen, las cuales me permiten direccionar los datos y que sean leídos solamente por el sistema direccionado.

El campo de código me permite controlar y/o solicitar información al sistema.

El campo de datos es variable según el requerimiento de la base.

El campo FCR es un campo para el control de errores no implementado aun.

E. *Control de motores*

Dicho bloque es el encargado de realizar el control de los motores los cuales son controlados con la tecnología de modulación PWM.

El microcontrolador una vez realizada la generación de trayectoria obtiene los ángulos de movimiento de cada motor y a partir de la posición inicial del brazo y la final establece una proporción del ancho del pulso por cada articulación según el camino a recorrer por articulación. De esta manera se lograra una que todas las articulaciones lleguen al punto final de manera simultánea y controlando el arranque y parada un movimiento suave.

V. *MODULO INALAMBRICO*

El módulo seleccionado es el AC4490 de Aerocom, ya presenta las siguientes características

- Posee modulación spread spectrum frequency hopping.
- Frecuencia de banda 902-928Mhz.
- Interface serial con baud rate de 1200 a 115200bps.
- 56 canales.
- Alcance vista libre 65km.
- Consumo de 1300mw.
- Tensión de alimentación 3,3V
- Buffer input output de 256 bytes.
- RF data rate 76,8 Kbps.
- Sensibilidad -100dbm.
- Interface conector de 20 pines.

Debido a la modulación spread spectrum y la banda de frecuencia lo hace invisible a la modulación de la antena, con lo cual se reducen las posibles interferencias.

Además dicho modulo puede funcionar en distintos modos como son:

- Punto a punto
- Punto a multipunto
- Cliente – Servidor.
- Peer to Peer.

El modulo presenta tres tipos de operación las cuales pueden ser:

- Transmisión.
- Recepción.
- Mediante comandos AT.

La transmisión como la recepción se la conoce generalmente como modo transparente.

La modalidad empleada en el presente trabajo es la modalidad transparente durante la transmisión de los datos.

El modo de comandos AT sirva para la configuración de parámetros y puede ser de forma inalámbrica; en caso de ser necesario.

VI. CONCLUSIONES

En el desarrollo del presente trabajo, la mayor complejidad hasta el momento se presentó en el diseño y construcción del sistema robótico. En cuanto a la electrónica de soporte se construyó de forma modular a fin de poder actualizar la electrónica en caso de ser necesario.

La trama conformada para la comunicación funciona correcta sin detectar cortes en la comunicación ni errores de recepción, ya que el modulo inalámbrico posee una trama rf con corrección y detección de errores; sin embargo a la trama conformada se le debe terminar la implementación de FCR, cuestión en la que se trabaja.

El control y movimiento de los motor reductores es suave y no presenta saltos o brincos durante el movimiento del sistema robótico, cuando es controlado o cuando realiza rutinas de reinicio, barrido, arranque suave y parada. Sin embargo las pequeñas vibraciones que presenta el sistema son

debidas a las partes mecánicas del mismo ya que es un prototipo.

El sistema responde bien al sensor de campo electromagnético y se encuentra en etapa de implementación de los algoritmos de generación de trayectoria y luego la obtención de datos experimentales.

La finalización del mismo permitirá continuar con otras líneas de investigación como la de transmisores inteligentes, el estudio de algoritmos para el control adaptativo del sistema robótico, el estudio de redes inalámbricas de antenas inteligentes y el procesamiento de datos distribuidos.

Si bien para el presente paper la elección de un brazo robótico de tres grados de libertad sea injustificada, se pretende medir con el mismo la radiación proveniente de pequeños equipos en distintos ángulos, e de ahí la necesidad de contar con tres grados de libertad y el agregado de la cinemática inversa.

REFERENCES

- [1] K.S. Fu, R.C. González, C.S.G. Lee. *ROBOTICA: Control, Detección, visión e inteligencia*. Editorial McGraw-Hill, 1990. 599 p.
- [2] Glover, F. and M. Laguna. (1997). *Tabú Search*. Kluwer, Norwell, MA.
- [3] Glover, F. "Tabú Search — Part I", *ORSA Journal on Computing* **1989** 1: 3, 190-206.
- [4] Hoja de datos del sensor de Inclinación MMA7060FC.
- [5] Hoja de datos del módulo Smart GPS antenna A1035-D
- [6] Hoja de datos del sensor de campo LT5534 de Linear Technology.
- [7] Hoja de datos del microcontrolador pic18f8722 de microchip.
- [8] Hoja de datos modulo inalámbrico AC4490 de Aerocom.
- [9] Hoja de datos del puente H L298N de STMicroelectronic.
- [10] Hoja de datos del motor reductor MR08D de Ignis.

Desarrollo de software de comunicación comandado por gestos utilizando una interfaz cerebro-computadora

B. Niro, J. Balbi, L. Ramos, N. González, F. Pose, M. Araujo
Departamento de Ingeniería Electrónica – Facultad Regional Buenos Aires
Universidad Tecnológica Nacional
Argentina
ngonzalez@frba.utn.edu.ar

Resumen— Los productos de apoyo permiten disminuir las barreras que enfrentan las personas con discapacidad y adultos mayores y así favorecer su accesibilidad, comunicación y autonomía. En la actualidad, el procesamiento de señales cerebrales puede ser utilizado como un medio para la creación de herramientas que brinden una real equiparación de oportunidades a aquellas personas que se ven limitadas diariamente. Este trabajo propone la creación de un software de comunicación a través de una interfaz cerebro-computadora mediante el uso de señales de EEG comandadas a través de gestos ya que es una técnica no invasiva, portátil y de bajo costo. Se propuso la realización de un módulo de entrenamiento y selección de gestos para luego poder articular el mismo con el software de comunicación.

Palabras claves— EEG ; comunicación, discapacidad ; procesamiento de señales

I. INTRODUCCIÓN

Los productos de apoyo, también conocidos como ayudas técnicas, son aquellos dispositivos que permiten disminuir las barreras a las que se enfrenta cualquier persona, sea esta del tipo social o de cualquier otro tipo.

Hoy en día, la aplicación de las tecnologías de la información y comunicación por las personas con discapacidad supone una condición necesaria para poder acceder a la educación, al trabajo, a la comunicación o al ocio[1].

Los productos de apoyo posibilitan una innegable mejora en la calidad de vida de las personas con “diversidad funcional”[2] al brindar una real equiparación de oportunidades a través de su uso, posibilitando la inserción social y laboral.

En los casos donde se trabaja con pacientes con parálisis cerebral (PC), esclerosis lateral amiotrófica (ELA) y traumatismo de cráneo (TEC), los productos de apoyo existentes brindan soluciones muy acotadas. Dentro de las limitaciones principales se encuentra la forma de acceso a la computadora como una herramienta la estimulación y rehabilitación, dentro de las dificultades en el acceso podemos distinguir movimientos involuntarios que hacen inviable el uso de mouse o teclados adaptados, el bajo tono muscular que no permite el traslado de miembros superiores o inferiores o la

fuerza suficiente para trabajar con un switch o pulsador adaptado.

II. MARCO TEÓRICO

En 1929, el científico Hans Berger desarrolló un sistema de exploración neurofisiológico basado en el registro de la actividad bioeléctrica cerebral (EEG).

En la actualidad existen diferentes métodos para registrar la actividad cerebral:

- Electroencefalograma (EEG).
- Magnetoencefalografía (MEG).
- Tomografía por emisión de positrones (PET).

A partir del registro del EEG es posible obtener diferentes tipos de señales para controlar un sistema basado en una interfaz cerebro-computadora.

Estas señales eléctricas pueden ser los llamados potenciales evocados (PE) que pueden ser medidos con electrodos en el cuero cabelludo como respuesta a un estímulo ya sea visual o auditiva. En particular los llamados PE P300 son usualmente utilizados en medición de la función cognitiva de los procesos de toma de decisiones y su obtención consta esencialmente en presentación de un conjunto de estímulos de los que solo unos pocos tienen relación a la intención del usuario. De este modo, los estímulos de interés, al ser infrecuentes y provocan la aparición del potencial en la actividad cerebral del usuario.

III. MATERIALES Y METODOS

Se utilizó como interfaz de adquisición un casco EMOTIV EPOC de 14 canales de adquisición, cuyas características técnicas pueden observarse en la Tabla 1.

Canales	14 canales de adquisición
Canales de referencia	2
Frecuencia de muestreo	128 Hz
Resolución	14 bits 1LSB = 0.51uV
Convertor	16 bit
Banda de paso	0.2-43 Hz
Filtros Notch	50 y 60 Hz
Rango dinámico	8400 uV(pp)

Tabla 1: Parámetros de adquisición EMOTIV

El diseño del software propuesto pretende facilitar al usuario la comunicación a través de la detección de expresiones faciales. En especial se concentró el trabajo sobre parpadeo, sonrisa, ceño fruncido (de ambas o de una ceja) y movimiento conjunto de las anteriores. Se utilizó para este fin, un conjunto de librerías provistas por la empresa EMOTIV [3], en particular, las propuestas dentro del reconocimiento facial.

IV. RESULTADOS

Se obtuvo una aplicación que se ejecuta en PC, basada en la utilización de una interfaz cerebro-computadora a través de la utilización del Casco Emotiv EPOC. Dicha aplicación posee un modo de entrenamiento para trabajar en función de las posibilidades de cada usuario en función de gestos repetitivos y voluntarios (figura 1).

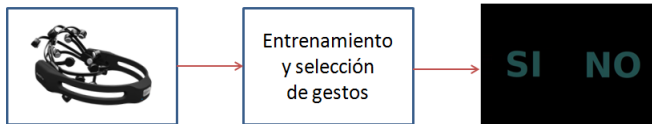


Figura 1: Diagrama en bloques de la aplicación

La aplicación cuenta con dos alternativas (Si / No). Esta aplicación fue desarrollada en C# utilizando el Framework 4.0 en trabajos anteriores [4] y fue modificada en este caso para ser utilizada a través de gestos realizados por el usuario.

El casco utilizado envía una trama utilizando sockets como canal de comunicación al módulo de entrenamiento y selección de gestos. Este módulo, dependiendo el modo de trabajo (entrenamiento/selección) almacena la información o la envía al módulo que posee la interfaz de usuario, filtrando previamente los eventos producidos en función de lo realizado en la etapa de selección. El módulo envía una trama a la interfaz de usuario, la cual es decodificada y en consecuencia activa una opción o la otra.

El desarrollo del módulo de entrenamiento y selección (figura 2) permite reutilizar el mismo en diferentes aplicaciones. El programa pudo ser utilizado con éxito por tres personas jóvenes sin afecciones en su motricidad.

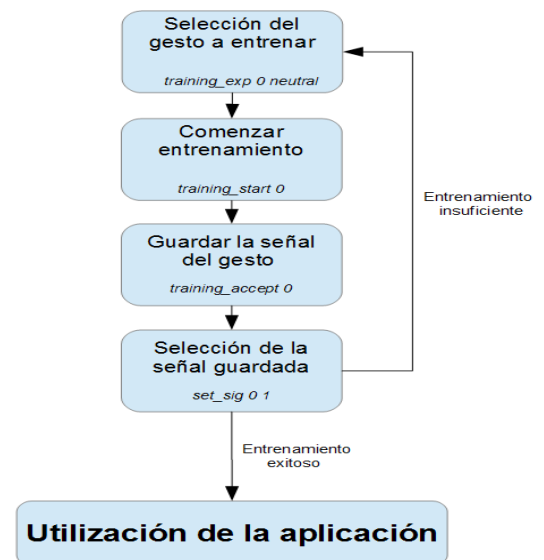


Figura 2: Diagrama en bloques del entrenamiento

V. CONCLUSIONES

La configuración inicial provista por las librerías del fabricante está diseñada para que se detecten las señales con cierto grado de éxito. Sin embargo, el entrenamiento posibilita un aumento en la precisión de detección de la aplicación para adaptar el procesamiento a las señales particulares de cada usuario. Entre más entrenamientos se realicen, la respuesta por parte del sistema será mejor frente a las señales del usuario, aumentando así la fidelidad del sistema.

Otro factor que influyó en el funcionamiento fue la concentración del usuario al momento de realizar el experimento.

VI. TRABAJO FUTURO

Como trabajos a futuro se buscará mejorar la etapa de entrenamiento, de forma de facilitar el trabajo a las familias o al equipo interdisciplinario que utilice esta aplicación.

A su vez se propondrá su uso en un comunicador de necesidades o comunicador avanzado. Como uso complementario se evaluará su aplicación como mouse adaptado.

VII. REFERENCIAS

- [1] R. Sanchez Montoya. Ordenador y Discapacidad. España, 2002.
- [2] J. Romañach, M. Lobato. Foro de Vida independiente. México, 2005. Disponible en www.asoc-ies.org/vidaindependen/docs/diversidad%20funcional_vf.pdf
- [3] Acceso a repositorio Emotiv. <https://github.com/Emotiv>
- [4] M. Araujo, N. González, F. Pose. Evaluación y detección de potenciales evocados sobre una interfaz cerebro computadora. Aranducon. Paraguay, 2016.

Un Nuevo Enfoque para Experimentos de Lazo Cerrado en Neurociencias Basado en Hardware Abierto y Software Libre

Demián García Violini

Instituto Tecnológico de Buenos Aires
(ITBA)
Universidad Nacional de Quilmes
(UNQ)
Comisión Nacional de Investigaciones
Científicas (CONICET)
Buenos Aires, Argentina.
ddgarcía@itba.edu.ar

Alejo Mosqueira

Universidad de Buenos Aires (UBA) –
Facultad de Ciencias Exactas (FCE)
Instituto de Fisiología y Biofísica
Bernardo Houssay (IFIBIO)
Buenos Aires, Argentina.
alejomosqueira@gmail.com

Félix Safar

Universidad Nacional de Quilmes
(UNQ)
Bernal, Buenos Aires, Argentina.
felix.safar@unq.edu.ar

Como es usual en neurociencias, las estimulaciones eléctricas son de uso corriente ya sea con fines científicos o incluso terapéuticos. Del mismo modo, en los últimos años se ha agregado la posibilidad de lograr intervenciones con estímulos lumínicos, permitiendo avances significativos en el entendimiento del código del cerebro [1]. Esto último, fue posible gracias a la técnica denominada optogenética, que permitió ampliar las posibilidades de estimulación en términos de precisión espacial, temporal o incluso por selección celular.

Por otro lado, el registro de la actividad cerebral, es una práctica que es usual desde hace ya varias décadas y utilizada tanto por científicos como médicos.

El objetivo principal de este trabajo es el desarrollo de un sistema de adquisición, análisis y procesamiento de datos en tiempo de ejecución que permita comandar el accionamiento de señales de estimulación, ya sean lumínicas, eléctricas, o incluso el disparo de eventos sonoros o visuales. Para esto, se toma la plataforma embebida de bajo costo desarrollada por la compañía Intan Technologies RHA2116-Eval [2] y se combina con el desarrollo de software en Labview de National Instruments. Cabe destacar, que todo el software y hardware desarrollado por Intan Technologies está realizado bajo el paradigma de software y hardware libre poniendo a disposición de quien requiera los esquemáticos electrónicos y códigos fuente del software. La combinación de un sistema embebido de adquisición con software de procesamiento en tiempo de ejecución, permite la automatización de experimentos electrofisiológicos *in-vivo*, por ejemplo, utilizando como criterio el análisis espectral de señales de LFP (por sus siglas en inglés **Local Field Potential**) [5].

La plataforma RHA2116-Eval, comprende dos sistemas diferenciables. El primero de ellos está constituido por un cabezal encargado de adquirir las señales provenientes de los electrodos (señales de entrada). Este cabezal está esencialmente constituido por un amplificador analógico

multiplexado de 16 canales (circuito integrado RHA2116), cuya ganancia fija en tensión es de 200 veces. Cada canal puede ser accedido de manera secuencial (por defecto) o aleatoria dependiendo de la configuración elegida mediante los pines de configuración.

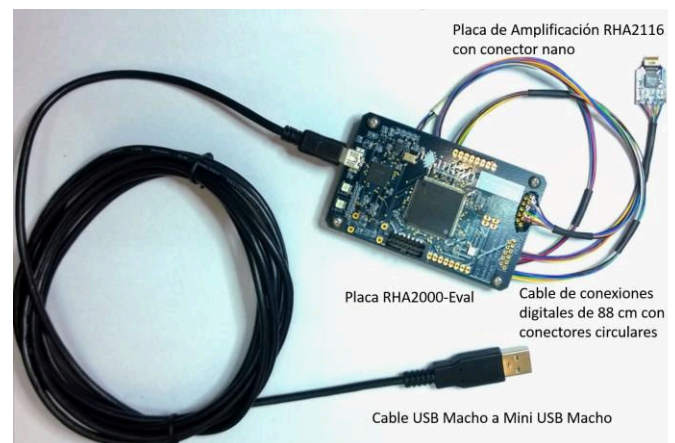


Figura 1: Plataforma Intan RHA2116-Eval. Imagen tomada de la hoja de datos del producto. Se muestran el cabezal y el módulo de retransmisión.

Estos valores luego son digitalizados (convertor AD7980 de 16 bits) y transmitidos por interfaz SPI. Estos valores transmitidos por el cabezal, son tomados por el sistema de captura y retransmisión estructurado en torno a una FPGA de la firma Xilinx (Spartan). Este módulo de captura y retransmisión, configura los datos de acuerdo a un protocolo definido por Intan. Además, da la posibilidad de incorporar 6 señales digitales externas auxiliares, por ejemplo, para incorporar marcas de sincronismo (por ejemplo, indicar el instante de acción de un estímulo visual cuando se estudian células nerviosas asociadas al sistema de visión). Todos estos

datos se configuran en paquetes de 3 bytes por cada electrodo en donde además se añade la información de las líneas auxiliares. Con esto, deberían considerarse 48 bytes para completar una transmisión de los 16 electrodos de entrada y 6 canales auxiliares. La Tabla 1 describe la organización de los bits de cada uno de los bytes que componen la trama. Allí, las referencias marcadas como Ch-n, con n= 0, 1, 2 y 3, codifican para cada canal, información de sincronismo y de las entradas auxiliares.

Byte 1	1	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	AD0
Byte 2	1	ADC13	ADC12	ADC11	ADC10	ADC9	ADC8	ADC7
Byte 3	0	0	Ch3	Ch2	Ch1	Ch0	ADC15	ADC14

Tabla 1: Definición de la trama de 3 bytes para cada uno de los canales.

Así, el módulo de retransmisión, envía los datos por USB (COM virtual) a una computadora personal para el monitoreo remoto *on-line*. Estos datos deberán ser recibidos y formateados para su visualización, análisis y procesamiento.

Este trabajo propone el desarrollo de un sistema de adquisición en Labview que permita, además de adquirir y procesar las señales que son enviadas por la interfaz USB, un análisis en tiempo de ejecución de los datos, mientras simultáneamente se comanda el disparo o modulación de alguna señal de estímulo. Como herramientas accesorias, se incorpora un filtro de notch para remoción de interferencias de línea y un filtrado ajustable por el usuario. También, la opción de remover medias que por características intrínsecas al amplificador, son añadidas al momento de la adquisición. De esta manera, se define un sistema de lazo cerrado como se muestra en la Figura 2.

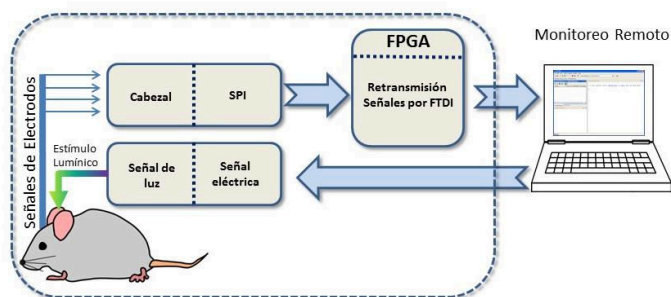


Figura 2: Lazo cerrado definido por el sistema desarrollado en Labview para la placa Intan RHA2116-Eval.

La Figura 3 muestra una captura de la interfaz con el usuario del entorno programado en Labview.

Ejemplo de Aplicación

Como ejemplo de las posibilidades de esta herramienta, se propone un experimento sobre una rata anestesiada afectada para sensibilizarla a estímulos ópticos (optogenética). Se desarrolló un sistema de detección de ritmo cerebral [3] que identifica el estado instantáneo diferenciando entre ritmos

delta y *theta*. Seleccionando el electrodo a estudiar, se comparan de manera instantánea la potencia normalizada en bandas *delta* (0.5 a 1.5 Hz) y *theta* (2.5 a 7 Hz). Esto último es utilizado como criterio de detección de estado, para comandar una señal de estímulo óptico que permita modular la transición *delta-theta* [4] para dos electrodos ubicados en el hipocampo y la habénula. Esta señal es comandada mediante una salida digital de una placa USB de National Instruments (NI USB-6212). La Figura 4, muestra los resultados en donde se evidencia una correcta detección del estado *delta* y una adecuada transición *delta-theta* transcurridos 270 segundos.



Figura 3: Interface con el usuario del entorno en Labview.

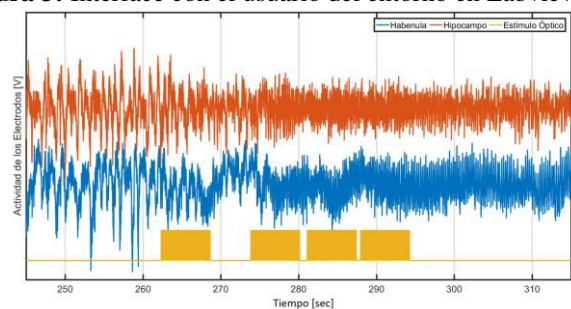


Figura 4: Ejemplo de aplicación sobre una rata anestesiada.

Conclusiones

Se presenta un sistema de adquisición, análisis basado en una plataforma de hardware de bajo costo, construida dentro del paradigma de software y hardware libre. Asimismo, propone una metodología de estímulo basada en el análisis en tiempo de ejecución de las señales que permite definir un lazo cerrado sobre sujeto (animal) de estudio.

References

- [1] K. Deisseroth, Optogenetics: Method of the year 2010, Nature Methods, vol. 8, no. 1, pp. 740-741, January 2011.
- [2] Intan Technologies, sitio web. <http://intantech.com/>. 2017, California, USA.
- [3] G. Buzsaki, Rhythms of the Brain, Oxford University Press 2017.
- [4] D. García Violini, Mosqueira A., P. Colmegna, J. Piriz, M. Belluscio, y R. Sánchez-Peña, "Characterization of Delta to Theta Transitions in the Hippocampus and Lateral Habenula Following Mechanic or Optogenetic Stimulation in Anesthetized Rats", Mar del Plata, Argentina, XXX Congreso Anual de la Sociedad Argentina de Investigación en Neurociencias.
- [5] Local Field Potential, online: https://en.wikipedia.org/wiki/Local_field_potential

Funciones Potenciales para Evasión de Obstáculos

Implementación en ROS con un robot terrestre

Julián Mauro Mateu Santos

Grupo de Procesamiento de Señales, Identificación y Control

Facultad de Ingeniería, Universidad de Buenos Aires,

Buenos Aires, Argentina

Email: julianmateu@gmail.com

Resumen—En el presente trabajo se implementa el algoritmo de funciones potenciales para evasión de obstáculos en el sistema ROS (*Robotics Operating System*), utilizando un robot terrestre (*Turtlebot*). Se simula utilizando el software *Gazebo* y se comparan los resultados de la simulación con pruebas realizadas con el robot físico. Luego se analizan las ventajas y desventajas del método y se proponen algunas mejoras para solucionar los problemas encontrados.

INTRODUCCIÓN

Existen varios métodos para evasión de obstáculos. Algunos de ellos buscan navegar desde la posición actual del robot hacia una meta prefijada, buscando un camino óptimo globalmente, minimizando el tiempo o la distancia recorrida. Sin embargo, dichos algoritmos son computacionalmente costosos para ser ejecutados en tiempo real, por lo que se requieren soluciones no óptimas, pero que puedan encontrar un camino de forma rápida y computacionalmente eficiente. Uno de estos métodos en tiempo real es el de funciones potenciales, que se implementa aquí.

Para la implementación de este algoritmo se utilizó el sistema ROS, y se simuló en el software *Gazebo*, utilizando un robot *Turtlebot*. Posteriormente se ejecutó el mismo programa simulado en el robot real.

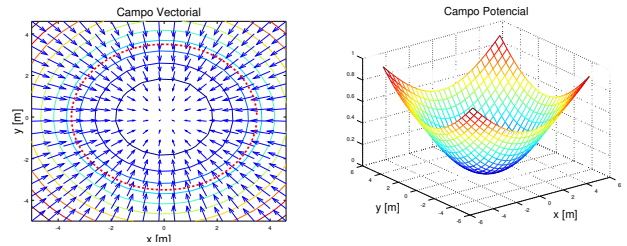
ALGORITMO DE FUNCIONES POTENCIALES

El algoritmo como se explica en [1] y en [2] se basa en definir un campo escalar, análogo al potencial utilizado en mecánica clásica y electromagnetismo, asociado a un campo vectorial opuesto a su gradiente. Si el potencial tiene máximos en los obstáculos y un mínimo en la meta, entonces el campo vectorial asociado representará la dirección de movimiento del robot que se aleja de los obstáculos y converge hacia la meta. Para obtener dicho campo, se superponen campos más sencillos y básicos: el **atractor** y el **repulsor**. El campo total se obtiene sumando un campo atractor centrado en la meta y varios campos repulsores, centrados en cada obstáculo.

Un campo atractor se representa de la siguiente forma:

$$\mathbf{F}_a(\rho, \theta) = \begin{cases} -\alpha\rho\hat{\rho} & \rho \leq s \\ -\alpha s\hat{\rho} & \rho > s \end{cases} \quad (1)$$

$$f_a(\rho, \theta) = \begin{cases} \alpha\frac{\rho^2}{2} & \rho \leq s \\ \alpha s(\rho - \frac{s}{2}) & \rho > s \end{cases} \quad (2)$$



(a) Gráfico del campo vectorial \mathbf{F}_a generado por un atractor. Los vectores se muestran con flechas azules, y en color las curvas de nivel del potencial. La línea roja punteada representa el radio de alcance s del campo. (b) Gráfico del campo escalar f_a (potencial) generado por un atractor.

Figura 1: Gráficos de los campos generados por un atractor.

donde la meta está en el origen, (ρ, θ) son las coordenadas polares del robot, α y s son constantes que controlan la amplitud y radio de alcance del campo respectivamente, y $\hat{\rho}$ es el versor radial. Se cumple que $\mathbf{F}_a(\rho, \theta) = -\nabla f_a(\rho, \theta)$. Puede verse el campo en los gráficos de la figura 1.

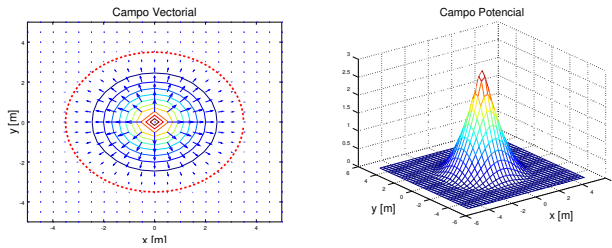
Análogamente, se define un campo repulsor, generado por un obstáculo puntual en el origen (figura 2):

$$\mathbf{F}_r(\rho, \theta) = \begin{cases} \beta(s - \rho)\hat{\rho} & \rho \leq s \\ \mathbf{0} & \rho > s \end{cases} \quad (3)$$

$$f_r(\rho, \theta) = \begin{cases} \beta(\frac{s^2}{2} - s\rho + \frac{\rho^2}{2}) & \rho \leq s \\ 0 & \rho > s \end{cases} \quad (4)$$

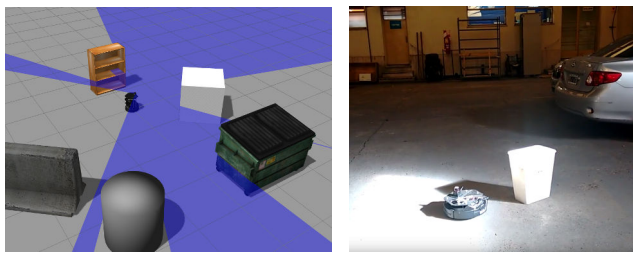
ROS Y GAZEBO

El sistema ROS permite codificar nodos que funcionan intercambiando mensajes entre sí. Por ejemplo, un nodo puede publicar la información de un sensor láser; otro nodo puede leerla y procesarla, y luego publicar un comando de velocidad vectorial; finalmente, otro nodo lee el comando y mueve el robot con la velocidad deseada. Esto permite que la simulación sea relativamente sencilla, utilizando *Gazebo*, un simulador que se puede integrar con ROS. *Gazebo* simula los nodos de sensores y actuadores, y la física del robot y el entorno, permitiendo utilizar los nodos de procesamiento sin modificaciones



(a) Gráfico del campo vectorial \mathbf{F}_r generado por un repulsor. (b) Gráfico del campo escalar f_a (potencial) generado por un repulsor.

Figura 2: Gráficos de los campos generados por un repulsor.



(a) Simulación en Gazebo. En el centro se ubica el robot Turtlebot del cual emanan los rayos del sensor láser radialmente. Se observa cómo éstos interactúan con diversos objetos. (b) Turtlebot real en un entorno con diversos obstáculos.

Figura 3: Simulación y ejecución del algoritmo con el robot real.

entre la simulación y la realidad. Es decir, el mismo programa que se ejecuta durante la simulación puede controlar el robot real.

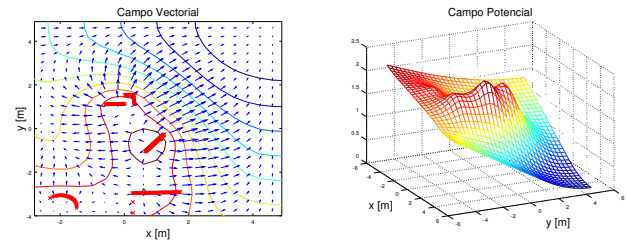
Existen varios robots y sensores integrados con ROS. En este trabajo se usó el Turtlebot, un robot cuyo modelo físico está detallado en Gazebo, y que posee nodos de ROS para obtener la información de sus sensores y comandar su movimiento. También se usó un sensor láser Hokuyo para detectar obstáculos. Una buena introducción a ROS y Gazebo puede encontrarse en [3].

RESULTADOS

Se simuló el algoritmo en Gazebo, utilizando varios entornos con obstáculos, como se ve en la figura 3a. Los gráficos de los campos correspondientes a dicho caso se muestran en la figura 4.

Si bien en casos como el que se muestra en las figuras 3a y 4 el algoritmo funciona satisfactoriamente, en otras configuraciones se observa un comportamiento propio del método de funciones potenciales: la presencia de mínimos locales en la función de potencial, hacia los que el robot se dirige y se detiene, pero que no corresponden a la posición de la meta deseada. Existen posibles soluciones a este problema, que se implementarán en el futuro.

Finalmente, se ejecutó el algoritmo simulado en el robot real, como se observa en la figura 3b, obteniendo los mismos



(a) Gráfico del campo vectorial correspondiente al potencial generado con la información del sensor láser. Las posiciones de los puntos medidos por el sensor se muestran con cruces rojas. (b) Gráfico del campo escalar co-generado con la información del sensor láser. La posición del robot corresponde al origen de coordenadas. La meta se encuentra en el punto $(x, y) = (5, 5)$.

Figura 4: Gráficos de los campos generados con la información del sensor láser. La posición del robot corresponde al origen de coordenadas. La meta se encuentra en el punto $(x, y) = (5, 5)$.

resultados que en la simulación: para algunas configuraciones el robot llega satisfactoriamente a la meta, esquivando los obstáculos, mientras que en otros casos llega a mínimos locales del potencial. El uso de la plataforma ROS permitió utilizar el mismo programa para la simulación y el robot real, siendo necesario únicamente un pequeño ajuste de las amplitudes relativas de los campos atractivo y repulsivo.

CONCLUSIONES

- Se logró implementar, simular y ejecutar en un Turtlebot de forma satisfactoria un algoritmo de evasión de obstáculos en tiempo real, con buena performance en recursos computacionales, y correcto en su solución en casos sin mínimos locales apreciables.
- Se comprobó a limitación del método de funciones potenciales, detectando la falla ante la presencia de mínimos locales.

PRÓXIMOS PASOS

El trabajo a futuro incluirá la implementación de mejoras al algoritmo para evitar caer en mínimos locales. Finalmente, se implementarán otros algoritmos de evasión de obstáculos en tiempo real, como por ejemplo los métodos desarrollados en [4] y [5], comparando performance, ventajas y desventajas de todos ellos.

REFERENCIAS

- Ronald C. Arkin. *Behavior-based Robotics*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.
- Maria Isabel Ribeiro. *Obstacle avoidance*. Instituto de Sistemas e Robótica, Instituto Superior Técnico, 2005.
- Aaron Martínez and Enrique Fernández. *Learning ROS for Robotics Programming*. Packt Publishing, 2013.
- J. Borenstein and Y. Koren. The vector field histogram – fast obstacle avoidance for mobile robots. *IEEE JOURNAL OF ROBOTICS AND AUTOMATION*, 7:278–288, 1991.
- C. Pozna, R. Precup, L. T. Koczy, and A. Ballagi. Potential field-based approach for obstacle avoidance trajectories. *TRANSACTIONS ON INTERNET RESEARCH - IPSI BGD*, 8(2):40–45, 2012.

Migración a la CIAA del sistema de control de espejos para un concentrador solar tipo Fresnel

Telmo Moya Grondona

Facultad de Ciencias Exactas
Sede Regional Metán – Rosario de la Frontera - UNSa
Salta, Argentina
telmomoya@gmail.com

Daniel Hoyos

Facultad de Ciencias Exactas
Universidad Nacional de Salta
Salta, Argentina
hoyosdani@gmail.com

Resumen— El trabajo presenta el uso de la Computadora Industrial Abierta Argentina (CIAA) para el sistema de control de espejos de un CLF. Este sistema de control funciona de manera distribuida sobre una red inalámbrica, implementando la lógica de control, etapas de potencia y los protocolos de comunicaciones necesarios.

Keywords— sistema de control; energía solar; concentrador Fresnel, CIAA.

I. INTRODUCCIÓN

En un concentrador lineal tipo Fresnel (CLF) un campo de espejos concentra la radiación solar directa en un absorbedor con el fin de convertir la energía solar en energía térmica. En el absorbedor se calienta agua hasta convertirse en vapor a alta temperatura y presión. Cada uno de los espejos debe orientarse según la trayectoria que sigue el movimiento relativo de la Tierra respecto del Sol.

La instalación estudiada se trata de un generador solar térmico de tipo Fresnel para la producción de vapor para usos industriales y generación eléctrica, siendo el primero en su tipo construido en Argentina. Está ubicado en la población de San Carlos, a unos kilómetros de Cafayate, en el sur de la provincia de Salta¹. Posee ocho espejos de 0,9 x 6,0 metros y un absorbedor de treinta metros de largo al que se conecta un motor recíprocante de vapor de 12 KWatts de potencia y un acumulador con 13 toneladas de hormigón. Cada espejo posee su propio sistema de control que lo mantiene orientado durante el día y lo resguarda durante la noche o en caso de granizo [1].

II. SISTEMA DE CONTROL

Los estrictos requerimientos mecánicos de un sistema CLF condicionan el resto del sistema de control. Para el sistema en estudio se diseñó, construyó, ensayó y optimizó el sistema motor-reductor con el objeto de simplificar su instalación y mantenimiento. Véase Figura 1.

En la primera implementación del sistema electrónico de control se utilizó un esquema centralizado para los ocho



Fig.1 Campo de espejos con sistemas motor-reducción

espejos que corría sobre Labview en una PC con conectividad cableada y Zigbee [2]. Posteriormente se optó por un modelo descentralizado, basado en una red de microcontroladores que interactúan entre sí para controlar el funcionamiento de la instalación [3]. Cada uno de estos subsistemas actúan en forma independiente y están compuestos por: motores paso a paso, interfaces de potencia y microcontroladores LPC1345 (ARM CortexM3) utilizando conectividad Bluetooth.

III. MIGRACIÓN A LA CIAA

Para una tercera versión del sistema de control se decidió utilizar la Computadora Industrial Abierta Argentina (CIAA) en su versión NXP. En su elección se tuvieron en cuenta los siguientes aspectos:

- **Confiabilidad:** Se trata de una plataforma sólida y probada, con múltiples casos de éxito.
- **Soporte:** El proyecto CIAA se encuentra activo, ofreciendo documentación, capacitación, soporte y actualizaciones permanentes.
- **Disponibilidad:** La CIAA NXP se encuentra actualmente disponible para la compra.
- **Software:** La versión anterior del software del sistema de control está escrito en C, lo que facilita la migración

utilizando las herramientas ofrecidas por el proyecto CIAA, como la IDE y el firmware, que dispone de las bibliotecas de drivers necesarias.

- **Hardware:** La transición al nuevo hardware no debe presentar mayores dificultades, por tratarse de un upgrade desde un microcontrolador ARM CortexM3 a un CortexM4. A nivel eléctrico tampoco existen dificultades ya que la CIAA es mucho más versátil.

El software desarrollado no utiliza sistema operativo, se trata de un proyecto “baremetal” en lenguaje C que corre sobre el Core M4 del microcontrolador LPC 4337 de la CIAA-NXP.

Las tareas llevadas a cabo para la migración del firmware a la CIAA básicamente implicaron adaptar el acceso a los periféricos del microcontrolador a través de los drivers del proyecto CIAA para UART, timers, conversor analógico-digital y puertos digitales de entrada/salida (GPIO). Mientras que para el manejo del RTC y la Eeprom se utilizaron bibliotecas del fabricante.

El sistema de control debe establecer la velocidad del motor paso a paso durante el día, para lo cual varía el tiempo entre pasos. Dada la complejidad de los cálculos necesarios se utilizan tablas precalculadas, existiendo 8 tablas diferentes, calculadas para sendos espejos. Cada una posee los valores de velocidad para períodos de tres minutos, sobre un total de 8 horas diarias que debe funcionar el sistema.

A fin de simplificar la instalación todos los sistemas poseen todas las tablas en memoria no volátil, pudiéndose seleccionar la tabla a utilizar. El requerimiento de dicha memoria puede calcularse mediante:

$$\text{ROM bytes} = \text{bytes por valor} * \text{cantidad de tablas} * \text{tiempo total} / \text{periodo}$$

Para almacenar los valores, que corresponden al tiempo entre pasos del motor expresados en milisegundos, se usan 16 bits = 2 bytes, la cantidad de tablas es 8, el tiempo total es 8*60 minutos y el período 3 minutos, lo que arroja un requerimiento total de memoria no volátil para las tablas de: 2560 bytes.

En la presente implementación se utilizan constantes en C para las tablas, lo que hace que se almacenen en la memoria Flash de programa, la cual por defecto es, de 512 KB para el core M4 de la CIAA NXP.

El manejo del motor paso a paso se hace utilizando cuatro pines de un puerto GPIO desde una ISR (rutina de servicio de interrupción) que se dispara con un timer (temporizador). Dicho timer se reprograma con los valores leídos de la tabla cada tres minutos, todos los días, desde las 10 hasta las 18 horas, para lo que se usa el RTC (reloj de tiempo real) de la CIAA. Llegadas las 18 horas el espejo se coloca en una posición de descanso y protección hasta el día siguiente. Existe también un botón, conectado a un pin GPIO, para utilizarse en caso de inclemencias climáticas (granizo

principalmente) el cual coloca el espejo en la posición de protección.

Por último el software incorpora un protocolo de monitoreo y control por puerto serie [4]. Esta funcionalidad se utiliza principalmente desde una aplicación Android vía Bluetooth, para lo que se conecta un módulo HC-06, en modo transparente, a la UART de la EDU-CIAA. Esto permite conocer el estado del sistema y eventualmente efectuar correcciones a la posición del espejo, que como puede verse funciona a lazo abierto.

El desarrollo se llevó a cabo con la EDU-CIAA (versión educativa de la CIAA) y se probó usando la etapa de potencia existente, que consiste en una placa independiente conteniendo un puente H basado en transistores MOSFET.

Dados los costos que implicarían la migración simultánea de los 8 sistemas a la nueva versión se previó la coexistencia de ambas versiones. Llegado el momento de reemplazar alguna placa se lo hará con la CIAA, en cuyo caso puede optarse por obviar la etapa de potencia existente y utilizar las salidas MOSFET de la CIAA, lo cual requiere cambiar los pines GPIO que se utilizan, pero que ya está previsto en el código del software.

IV. CONCLUSIONES

Los beneficios ofrecidos por la CIAA al sistema de control no se limitaron a superiores prestaciones de hardware, sino que alcanzaron también a aspectos relacionados a la calidad de la solución obtenida. La estructura de capas utilizada en el firmware la CIAA ayudó notablemente a organizar el software de control, con la consecuente mejora en términos de documentación, portabilidad y mantenimiento.

Está previsto el desarrollo de una nueva versión utilizando sistema operativo, también dentro de los lineamientos del Proyecto CIAA.

Cabe destacar la especial relevancia para esta implementación que tuvo la capacitación recibida por miembros del proyecto en las instancias de la Escuela Argentina para la Enseñanza de Sistemas Embebidos.

REFERENCIAS

- [1] L. Saravia, M. Gea, M. Hongn, D. Hoyos, H. Barcena, C. Placco, C. Cadena, S. Flores Larsen, P. Dellicompagni, M. Condorí, C. Martínez, C. Fernández, R. Caso, M. Altamirano, H. Suligoy, “Descripción de un generador solar térmico de tipo fresnel instalado en San Carlos, Salta”, Avances en Energías Renovables y Medio Ambiente, vol. 18, pp.03.17-03.26, 2014.
- [2] D. Hoyos T. Moya R. Echazú A. Hernández, “Sistema de control para el seguidor solar de un concentrador tipo Fresnel”, Avances en Energías Renovables y Medio Ambiente, vol. 13, pp. 161-167, 2009.
- [3] D. Hoyos, M. Hongn, V. H. Serrano, S. Esteban, T. Moya “Sistema de control para concentradores solares tipo Fresnel”, Avances en Energías Renovables y Medio Ambiente vol. 16, 2012.
- [4] D. Hoyos T. Moya M. Villena, V.H. Serrano, “Sistema de monitoreo y control remoto para un generador de electricidad con un concentrador Fresnel”, Avances en Energías Renovables y Medio Ambiente vol. 18, pp.08.69-08.78, 2014.

Foro Tecnológico

Resumen

Implementación de Sistemas Embebidos

Estimador de F_0 en tiempo real basado en el algoritmo de YIN

Franco S. Caspe; Adrián H. Laiuppa, Oscar A. Rodriguez, Miguel A. Banchieri, Christian L. Galasso
Departamento de Ing. Electrónica
Facultad Regional Bahía Blanca
Universidad Tecnológica Nacional.
francocaspe@hotmail.com, alaiuppa@gmail.com, arodrig@frbb.utn.edu.ar, mbanch@frbb.utn.edu.ar,
christian_galasso81@yahoo.com.ar

Resumen— Se describe la implementación de un estimador de F_0 , basado en el algoritmo de YIN[1], en un microcontrolador de la familia Cortex M4. Este sistema, orientado al análisis de voz humana, fue pensado para la operación en conjunto con dispositivos de generación y reproducción de audio frecuencias, por lo que se buscó un funcionamiento en tiempo real, sin pérdida de datos y con inmunidad al ruido, todo esto asegurando un consumo computacional acorde a la capacidad del microcontrolador.

Por último se detallan los ensayos realizados y las directivas tomadas para salvar algunas de las limitaciones del algoritmo y un error sistemático encontrado.

Palabras clave— Detección de F_0 ; Procesamiento de señales; Microcontrolador; Frecuencia fundamental; Sistema embebido.

I. INTRODUCCIÓN

Hoy en día existe una gran demanda de herramientas vinculadas al procesamiento de voz. Sus aplicaciones, que van desde la producción musical, hasta la fonoaudiología, pasando por la domótica, la clasificación de contenido, y los sistemas de seguridad, requieren la implementación de algoritmos que estimen la frecuencia fundamental y que sean cada vez más eficientes. Usualmente, se busca integrar estas funcionalidades en sistemas embebidos, campo que demostró contener un gran potencial de aprovechamiento por la variedad de usos finales que estos poseen.

Los microcontroladores utilizados en estos sistemas, generalmente cuentan con la capacidad de procesamiento necesaria para la implementación de un estimador. Es por ello que se decidió ensayar uno, sobre una arquitectura ARM Cortex M4 [2], extensamente difundida.

Por otro lado, las aplicaciones de sistemas embebidos centradas en procesamiento de audio, generalmente imponen la necesidad de una ejecución en tiempo real; es natural pensar que se preferirá operar con un sistema más potente y flexible, como una computadora personal, si los resultados son solicitados en forma diferida (sin restricciones de tiempo real).

Bajo este contexto se diseñó una plataforma de experimentación sobre la cual opera un estimador de frecuencia fundamental, que actúa en tiempo real, orientado a una aplicación musical, que se representa en el esquema mostrado

en la Figura 1.

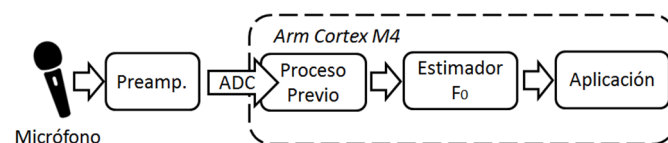


Figura 1. Esquema de la plataforma de experimentación. Se muestra el flujo de la información que será procesada por el bloque de aplicación.

Debido a la variedad de utilidades posibles, este trabajo está focalizado sobre los primeros bloques funcionales, explicando la captura y procesamiento de la señal, y finalmente como se detecta su frecuencia fundamental. Se buscó, así mismo, el menor tiempo de respuesta de forma de ponderar las prestaciones para su aplicación.

II. FUNDAMENTOS

Dentro de la gama de recursos musicales que son empleados comúnmente, y de los que interesa extraer la F_0 , la voz humana cantada es generalmente la más difícil de procesar correctamente, debido a que presenta una serie de características que complican la determinación de su frecuencia fundamental. Un análisis del espectro revela una gran cantidad de energía inarmónica, cuya magnitud puede lograr que un estimador sea desviado hacia un resultado erróneo, especialmente si éste espera una señal puramente armónica.

Es por eso que cualquier detector de F_0 , preparado para trabajar con voz debe incorporar en su modelo estrategias en las que se consideran los factores que apartan a la voz humana de una señal puramente armónica, a saber:

- Respuesta de la glotis a la vibración de las cuerdas vocales: Los inarmónicos que genera son los responsables del timbre de la voz [3].
- Consonantes: Estas fonías son naturalmente inarmónicas, su espectro frecuencial es denso dentro de un rango de frecuencias [4].
- Modulaciones: Llamados “vibratos” son variaciones naturales o intencionales sobre la amplitud y la frecuencia de la voz.

Otra característica, está vinculada a la percepción de tono, es la que genera en los algoritmos el denominado “error de octava”. Durante el canto, los resonadores acústicos del cuerpo (tórax, cabeza, nariz, otros) pueden modificar el nivel de energía de algún armónico, provocando que el algoritmo considere un pico frecuencial que no es la fundamental. Estos armónicos aumentados son conocidos como formantes. Sin embargo, auditivamente, el tono que se percibe es el correspondiente a la nota más grave y no a la de mayor energía [5].

En este campo, el algoritmo de YIN, descrito en 2002 por Alain de Cheveigne y Hideki Kawahara, supera la performance de otros métodos clásicos como Cepstrum o la interpretación de resultados obtenidos por FFT, disminuyendo los errores de octava, presentando mayor precisión y acelerando los tiempos de proceso [6].

Por otro lado, el microcontrolador seleccionado para el desarrollo del estimador, posee la simpleza suficiente para poder codificar la aplicación en bare machine, es decir sin ningún entorno de ejecución más que el propio chip, pudiendo de esta manera controlar finamente los tiempos de proceso, que serán detallados más adelante, para obtener la característica de tiempo real buscada.

Este microcontrolador, cuenta con una unidad de punto flotante de 32 bits, admite una frecuencia de reloj de hasta 168 MHz, aunque se lo operó a 96 MHz, y puede implementar puertos serie de velocidades no estándar, haciéndolo ideal para operar con dispositivos MIDI [7], de uso muy frecuente en producción musical. Por último, posee muy buenas características de gestión de energía, permitiendo administrar el consumo durante períodos de carga computacional y durante los tiempos de inactividad en donde no se detecta señal para procesar.

III. DESARROLLO

A. Muestreo de la señal

Considerando que se pretende trabajar con la información contenida en la voz humana y que la misma queda comprendida en la banda de bajas frecuencias, se definió como segmento útil del espectro al intervalo de frecuencias que comprende hasta los 4000 Hz [8]. En este segmento puede encontrarse tanto la frecuencia fundamental de la voz, que será procesada, como así también la inteligencia contenida, que es de interés para aplicaciones de transcripción a texto el cual es otro posible uso de los estimadores de F_0 .

Para concentrar la capacidad computacional del microcontrolador en la estimación, se evitó en principio el uso de un filtro digital, por lo que la frecuencia de corte superior fue entonces definida por un filtro activo de orden 2, integrado dentro del preamplificador de entrada, rechazándose así la banda audible que no contiene información, y oficiando también de filtro anti-alias.

La señal es digitalizada entonces a una velocidad de 8 kHz, con una resolución de 12 bits, que es la máxima resolución alcanzable por los ADC del microcontrolador, y establece un

piso de ruido que es despreciable frente al ruido de fondo capturado por el micrófono.

La información capturada y codificada, es volcada en un arreglo de 256 valores, ubicado en la memoria RAM, gestionado por el DMA del microcontrolador, automatizando el proceso y liberando al CPU de esta tarea. Este arreglo captura 32 ms de audio.

Teniendo en cuenta las limitaciones del algoritmo de YIN, postuladas en [1], empleando un buffer de 256 muestras, se podrá detectar períodos cuya longitud no supere las 128 muestras. Esto es debido al proceso de autocorrelación que el algoritmo ejecuta, en donde se necesitan capturar al menos dos ciclos completos de una señal para estimar correctamente su período; de manera que teóricamente, se podrá detectar una frecuencia de por lo menos 62,5 Hz (frecuencia correspondiente a un período de 128 valores, digitalizado a 8 kHz), ubicada al comienzo de la banda audible. El algoritmo no especifica una frecuencia de detección superior, por lo que, en principio, se espera una capacidad de reconocimiento algo superior a los 2000 Hz, frecuencia fundamental para la cual se detecta un período de 4 muestras.

B. Codificación del algoritmo de YIN

La implementación del estimador, se realizó mediante tres funciones separadas que integran la totalidad de los pasos a través de los cuales se logra obtener el período de la señal.

En primer lugar, se calcula, en formato de punto flotante lo que en el algoritmo se define como función diferencia, de longitud igual a un medio la longitud del buffer circular, como se muestra en la ecuación (1):

$$d_t(\tau) = \sum_{j=1}^W (x_j - x_{j+\tau})^2 \quad (1)$$

Donde x , es el arreglo de la señal de entrada, en formato entero, de longitud de 12 bits, acoplado a un valor constante de offset que idealmente equivale a la mitad del rango numérico que admite la resolución de muestreo.

Por otro lado, W es la longitud de la ventana sobre la cual se efectúa la diferencia y τ es la variable independiente que controla su deslizamiento.

El cálculo de $d_t(\tau)$, se implementó en una primera función, que admite el arreglo de valores muestreados, y devuelve los 128 valores de la función diferencia.

Posteriormente, debe normalizarse de forma acumulativa la función diferencia, obteniendo una secuencia de la misma longitud que la anterior, pero que admite valores entre 0 y 1. Este segundo paso, fue implementado en otra función separada. El proceso de normalización se muestra en la ecuación (2).

$$d'(\tau) = \begin{cases} 1, & \text{si } \tau=0, \\ d_t(\tau) / \left[(1/\tau) \sum_{j=1}^{\tau} d_t(j) \right] & \text{si } \tau \neq 0. \end{cases} \quad (2)$$

Como puede deducirse de las ecuaciones (1) y (2), la señal muestreada no necesita ser normalizada ni desacoplada digitalmente, puesto que las funciones calculadas dependen únicamente de las variaciones entre muestras, y este resultado se normaliza posteriormente. Evitando estos dos procesos, se logra incrementar la eficiencia del cómputo.

Posteriormente, siguiendo los pasos del algoritmo, se codificó una tercera función que incluye en primer lugar la búsqueda de la posición τ_{\min} dentro de la región de $d'(\tau)$ caracterizada por la presencia de valores menores a un determinado umbral denominado “umbral de YIN”. Es recomendable que este valor sea lo suficientemente estricto para evitar que el algoritmo genere errores de octava. Si no se encuentra ningún valor que cumpla con estas características, se buscará el mínimo global de toda la función.

Finalmente, se incluye en la misma función, el proceso de interpolación de una parábola positiva entre $(\tau_{\min} - 1)$ y $(\tau_{\min} + 1)$, que permite hallar, haciendo ciertas suposiciones sobre el contenido armónico de la señal, el valor final de período de x , denominado τ_{\min}' . Por último, la amplitud mínima de la parábola interpolada, es considerada por el autor del algoritmo como una medida de la energía aperiódica de la señal, teniendo en cuenta que, idealmente, este valor debería ser cero cuando la señal a comparar y la contenida en la ventana son perfectamente iguales. Este proceso se muestra en la Figura 2.

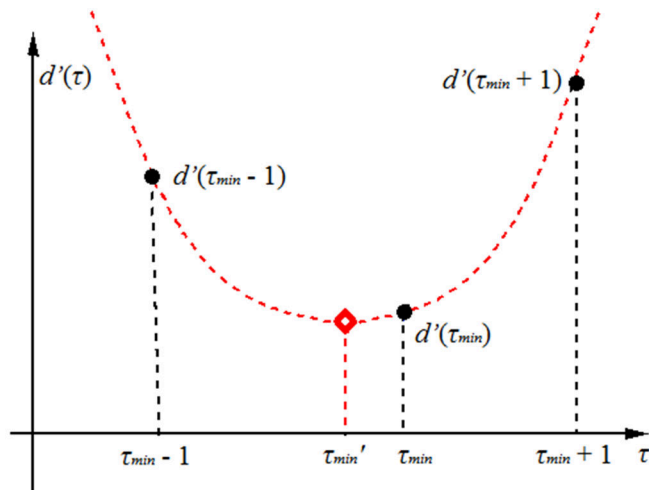


Figura 2. Búsqueda del mínimo e interpolación parabólica

C. Ajuste de las características de tiempo real

Considerando las tres funciones implementadas, y bajo las condiciones de operación anteriormente mencionadas, se midieron los tiempos de procesamiento de cada una de las tres operaciones que juntas permiten arribar al valor de período de la señal. Los tiempos se determinaron primero utilizando un

timer del microcontrolador, reseteado y disparado antes de comenzar cada operación, y luego se cotejaron utilizando una salida digital de señalización del microcontrolador, asignable en una operación unitaria, y un osciloscopio. Se creó una señal de prueba grabada como constante en memoria, que obligaba al algoritmo a ejecutar la búsqueda del mínimo más largo posible. Los resultados se muestran en la Tabla 1.

TABLA I. TIEMPO DE PROCESAMIENTO DE LAS OPERACIONES DE YIN

Operación	Tiempo [μs]
1. Cálculo de la función diferencia	7378
2. Normalización de la función diferencia	9494
3. Algoritmo de búsqueda e interpolación	9534
Tiempo total de procesamiento	26406

Con el objetivo de no interrumpir el muestreo para no perder información, y teniendo en cuenta que el acceso a memoria se realiza de forma progresiva en el tiempo durante el cálculo de la función diferencia (acorde la ventana se desliza hasta el final del arreglo), es posible realizar este cómputo en paralelo con el refresco del buffer, debido a que el análisis de la información se realiza más velozmente que la sobre escritura. Esto permite realizar una gestión fina del temporizado, evitando la necesidad de utilizar buffers dobles, por ejemplo, ampliando la gama de dispositivos que toleren la implementación. Un diagrama temporal se muestra en la Figura 3.

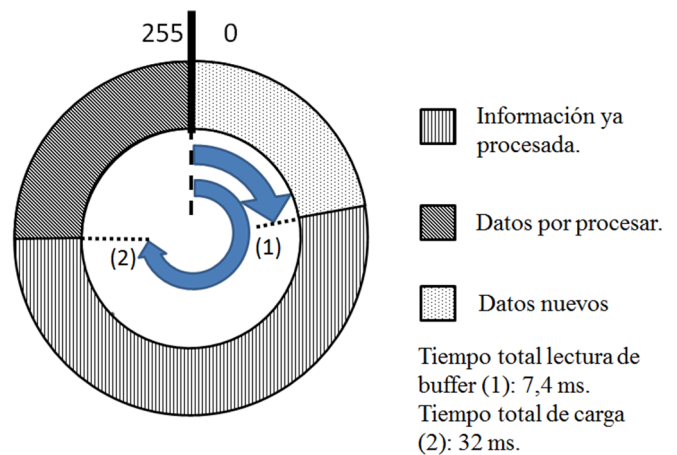


Figura 3. Diagrama de procesos en un instante temporal. La lectura progresiva de la información del buffer (2) se realiza más rápidamente que su sobreescritura (1).

Por otro lado, dado que para procesar 32 ms de audio, se tarda más de 26 ms, no fue posible aumentar de forma práctica la cadencia de detecciones, efectuándose entonces una estimación de frecuencia cada 32 ms. El microcontrolador tendrá un espacio de 5 ms para realizar las tareas de rutina como refresco del display, control de la interfaz de usuario, otros. El diagrama de la Figura 4 muestra la evolución temporal de los procesos.

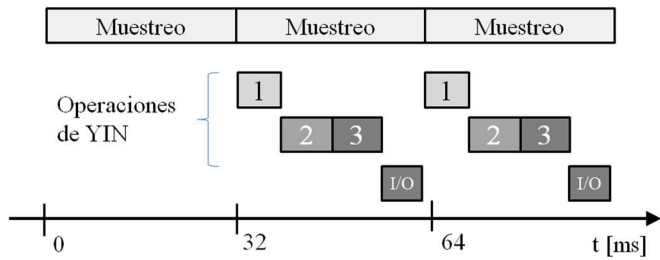


Figura 4. Diagrama temporal de respuesta. Téngase en cuenta que las operaciones anteriores y sucesivas de muestreo y cálculo no se muestran.

Habiendo determinado los tiempos de operación, y con la certeza de que no se perderá información, se puede afirmar que el sistema presenta la característica de tiempo real, por lo que interesa conocer el tiempo de respuesta.

Considérese un evento que se quiere detectar, como un cambio en la frecuencia fundamental de la señal de entrada. El máximo retardo en la detección estará dado por el tiempo de carga del buffer, sumado al tiempo de ejecución del estimador, dando un total de 58,4 ms. El esquema de proceso se muestra en la Figura 5.

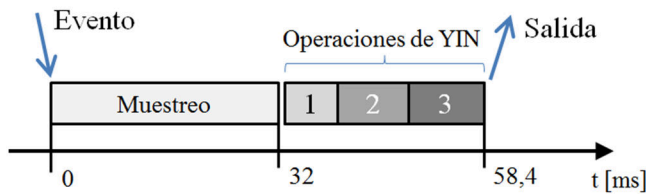


Figura 5. Diagrama temporal de respuesta. Téngase en cuenta que las operaciones anteriores y sucesivas de muestreo y cálculo no se muestran.

Para mejorar el tiempo de respuesta, se tuvieron en cuenta nuevamente los tiempos de acceso y de carga del buffer circular; si se dispara el cálculo de la función diferencia antes de que finalice el llenado, se puede lograr que ambas operaciones concluyan casi simultáneamente, reduciendo el tiempo de respuesta hasta 7 ms, como se muestra en la Figura 6.

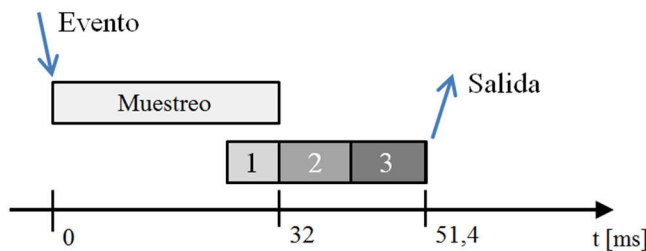


Figura 6. Reducción del retardo por paralelización del proceso de lectura/escritura.

D. Reducción de la sensibilidad al ruido

El algoritmo de YIN tiene la particularidad de asignar un período válido a un amplio rango de señales. El ruido eléctrico

o el ruido de fondo de un micrófono, por ejemplo, suelen presentar variaciones que, dentro del cuadro de análisis, se consideran de naturaleza armónica, produciéndose un valor de F_0 erróneo. La ocurrencia de este fenómeno es independiente de la amplitud de la señal de entrada (debido a que la función diferencia se normaliza).

Con el objetivo de evitar la implementación de un discriminador basado en el cálculo del valor RMS de la entrada, puede establecerse un segundo umbral que contempla la aperiodicidad mínima de la función diferencia normalizada.

Recordando que la aplicación del “umbral de YIN” define una región de búsqueda sobre $d'(\tau)$, el umbral de discriminación determinará si esa búsqueda es aplicable, analizando la presencia de al menos un valor menor a este límite, sobre la función mencionada.

Durante las pruebas previas a la implementación, se descubrió que un valor adecuado para el umbral de discriminación fue 0,2, mientras que el valor seleccionado para el umbral de YIN fue de 0,1. En las Figuras 7 y 8, se comparan los resultados logrados al aplicar o no la discriminación, utilizando como entrada una señal capturada por un micrófono electret desde una PC. En la misma, un vocalista ejecuta una escala ascendente nombrando cada una de las notas que canta. Los parámetros de operación se definieron iguales a los del microcontrolador.

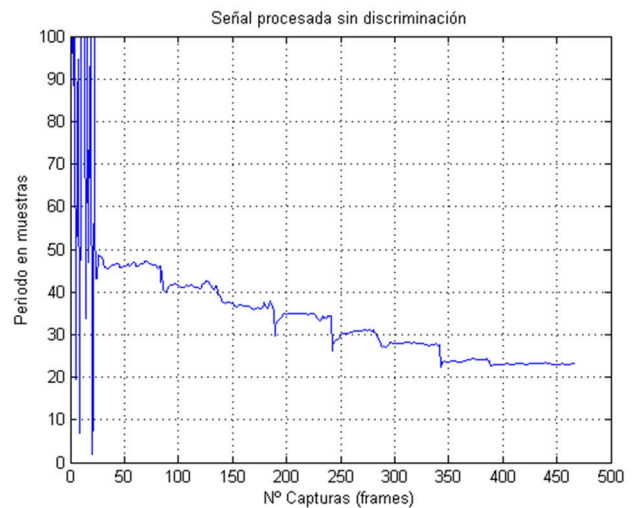


Figura 7. Evolución del período de una señal de voz cantada. El ruido de fondo al inicio de la grabación genera valores incorrectos.

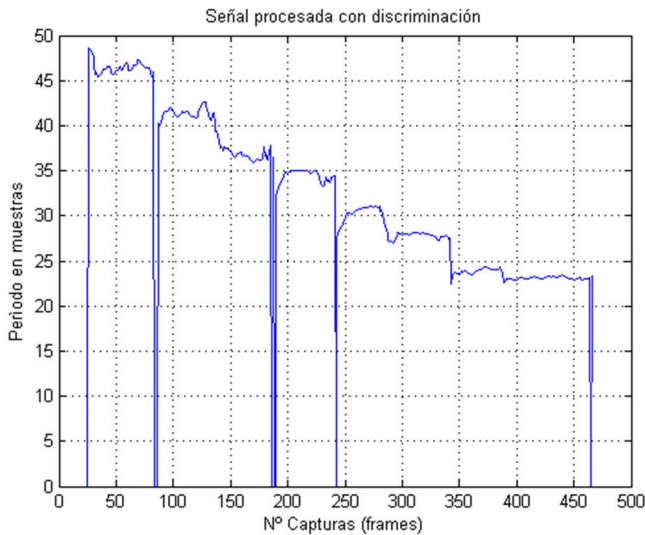


Figura 8. Procesamiento de la señal con discriminador de aperiodicidad. Se observa que no se detectan eventos hasta que la voz no se escucha en la grabación. Adicionalmente se detectan separadas varias de las notas ejecutadas (detección de consonantes).

IV. ENSAYOS REALIZADOS

Considerando el modelo establecido en la Figura 1, para el primer ensayo se diseñó un convertidor de frecuencia a MIDI, con el objetivo de poder observar en una escala musical la evolución de la frecuencia fundamental del audio captado por el micrófono.

El convertidor genera una nota musical cuando se detecta un nuevo evento, es decir, se obtiene un valor de período válido. Adicionalmente, se redondearán los valores de frecuencia fundamental hacia la nota más cercana de la escala.

El primer ensayo consistió en conectar un micrófono, y sin ejecutar nota alguna, elevar la ganancia del preamplificador de entrada, confirmando la ausencia de disparos por ruido.

Posteriormente se ensayó el dispositivo a ganancia reducida, observándose una detección correcta hasta una determinada nota ejecutada por el vocalista, donde se detecta en la salida, un desplazamiento de un semitono hacia los agudos. Se comprobó en esta instancia la correcta generación de los eventos de cambio y aparición de frecuencia a la entrada, aunque con un error en el valor detectado.

Para caracterizar el error encontrado en el ensayo anterior, se modificó el bloque de aplicación, con la función de mostrar el valor de frecuencia fundamental detectada, en un display instalado para tal fin. Se conectó un generador de funciones a la entrada, buscando realizar el ensayo con una señal armónica pura.

Se efectuaron dos barridos, partiendo desde el mínimo valor detectable (62,5 Hz), hasta la frecuencia considerada de detección factible, correspondiente a un período de 4 muestras.

Estos se efectuaron empleando dos valores distintos de umbral de YIN.

En las Figura 9, se muestran los valores de error relativo obtenido, en función de la fracción entre la frecuencia de entrada y la de muestreo. Se comienza a graficar desde $f/f_s = 0,01$, dado que para valores menores, el error observado no supera el 1 %.

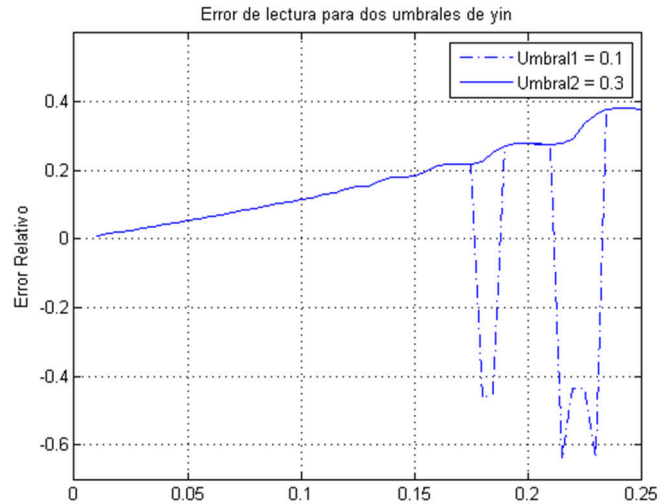


Figura 9. Gráfica del error relativo, para dos valores de umbral de operación, computado entre la frecuencia de entrada, y la lectura del equipo. Las gráficas generalmente coinciden, separándose en dos rangos distintos de frecuencias.

Observando la Figura 9, pueden extraerse dos conclusiones importantes: primero, se verifica la existencia de un error acumulativo con la frecuencia, independiente de las condiciones de operación del algoritmo. Por otro lado se observa que si se opera con un valor de umbral pequeño, aparecen dos regiones de detección en donde no se respeta la correspondencia uno a uno entre la frecuencia de entrada y la leída por el dispositivo. Se supone, que esto último se debe a que al trabajar con valores pequeños de umbral, y períodos de escasas muestras, la región de búsqueda que se abre en la función diferencia normalizada puede llegar a no contener el mínimo representante del período de la señal. Evidentemente, los valores de aperiodicidad aumentan cuando el período se achica. La falta de resolución en muestras puede llegar a ser el factor causante de este problema.

Posteriormente se descubrió que el error acumulativo se debía a una interpretación textual y errónea de la posición τ_{min} , que indica el valor del período de la señal. Debido a que el arreglo de la función diferencia comienza en la posición cero, una señal con un período de 64 muestras, tiene un mínimo en la posición 63, un período de 8 muestras, en la ubicación 7, y así sucesivamente. Si se toman como válidas las posiciones sobre la función diferencia, se generará un patrón de error relativo que crece con la frecuencia, acorde el período se achica. La solución a este problema fue justamente, sumar una unidad al valor calculado.

Respecto al quiebre de la correspondencia inequívoca entre la frecuencia de entrada y la lectura, se prefirió operar con un valor de umbral de YIN pequeño, debido a que la aplicación de

una restricción más laxa genera errores de octava en toda la banda útil de detección. El valor máximo de frecuencia para el cual no se presentan problemas, es del 17,5 % de la frecuencia de muestreo, lo que corresponde a un 35 % de la banda de Nyquist. Cabe aclarar que teóricamente, solo el 65 % inferior de esta banda presenta condiciones favorables de operación, debido a que la detección de períodos de 3 muestras o menos no es estable si no se sincroniza la frecuencia de muestreo con la señal de entrada.

De esta manera se definió el rango útil de operación del algoritmo, que está comprendido entre los 62,5 Hz y la frecuencia máxima de detección, que resulta ser de 1400 Hz, correspondiente a la nota F6, que escapa del rango de las tésituras vocales clásicas (la voz soprano tiene tabulada al DO6 como nota máxima).

V. CONCLUSIONES

El estimador presentado, cumplió con los requisitos postulados al principio del trabajo. Sin embargo, si se desea asegurar un comportamiento en tiempo real, sin pérdida de información, no será posible la implementación de alguna aplicación de relevancia debido a la falta de tiempo de operación disponible.

El cuello de botella presentado por el sistema quedó determinado por la capacidad y frecuencia de operación del procesador. Sin embargo, el Cortex M4 utilizado ofrece la posibilidad de incrementar en casi un 60% la velocidad de reloj utilizada para el presente desarrollo, pudiéndose así por ejemplo, disparar dos veces el estimador por cada llenado de un buffer más corto, o de ampliar las funcionalidades del sistema embebiendo una aplicación cuyo tiempo de operación dentro del bucle principal sea mayor a 5ms.

Por otro lado, es posible modificar la frecuencia de muestreo y la longitud de la ventana temporal para ajustar el rango de detección. Sin embargo, el incremento de la frecuencia de reloj y el aumento de la velocidad de muestreo dan lugar a nuevas condiciones de operación óptimas, de tiempo de respuesta mínimo. Si se cuenta con velocidades lo suficientemente altas, será la duración de la ventana temporal la que defina el retardo de la respuesta.

Siguiendo con esa idea, el retardo obtenido, de 52 ms, es generalmente demasiado grande para algunas aplicaciones en donde se requiere velocidad de análisis, como los sistemas de efectos digitales, utilizados en conciertos musicales y espectáculos. Deberán aplicarse las estrategias anteriormente mencionadas para lograr retardos adecuados a la aplicación, reservando también el espacio temporal adecuado para su operación.

Respecto al rango limitado de detección para valores pequeños de umbral de YIN, si se debe operar el estimador con otros instrumentos musicales de tésitura más amplia, puede implementarse un filtro IIR, de baja demanda computacional, delimitador de rango, para establecer una comparación entre la energía de la señal de entrada y salida del filtro. Este análisis

diferencial puede comandar el proceso de estimación de F_0 . Esta estrategia fue implementada posteriormente, incrementando la frecuencia de operación del microcontrolador, con resultados satisfactorios.

La implementación del algoritmo de YIN presentada resulta ideal para el desarrollo de sistemas embebidos con requerimientos de tiempo real, por la flexibilidad de su rango de operación, implementación y consumo computacional.

VI. TRABAJOS A FUTURO

Debido a que el tiempo de operación queda determinado en gran medida por la duración temporal del buffer circular, es interesante considerar un arreglo de longitud variable, a frecuencia de muestreo constante, con el que pueden reducirse drásticamente los tiempos de cálculo.

La expansión y compresión del buffer debería darse en torno a una pauta. Por ejemplo, si se están cantando notas agudas, la longitud puede reducirse, debido a que las mismas poseen un período reducido. La implementación de este mecanismo requiere el diseño de un mecanismo de toma de decisiones complejo.

REFERENCIAS

- [1] "YIN, a fundamental frequency estimator for speech and music" Disponible para acceso libre en: http://iwk.mdw.ac.at/lit_db_iwk/download.php?id=12777
- [2] Arm Developer. Detalle de la familia de microcontroladores Cortex M4. <https://developer.arm.com/products/processors/cortex-m/cortex-m4>
- [3] "Measures of the glottal source spectrum". Jody Kreiman, Bruce R. Gerratt, Norma Antónanzas-Barroso. <http://www.linguistics.ucla.edu/people/keating/Kreiman%20et%20al%202007.pdf>
- [4] "Speech spectra and Spectograms" de Robert Mannell. Pronunciación de consonantes y la evolución de su espectro en el tiempo. http://clas.mq.edu.au/speech/acoustics/speech_spectra/oral_stops.html
- [5] "The Singer's Formant" Detalle de la distribución frecuencial de la voz, bajo el fenómeno de resonancia. Disponible en: <http://www.ncvs.org/ncvs/tutorials/voiceprod/tutorial/singer.html>
- [6] "Pitch Extraction and Fundamental Frequency: History and Current Techniques" David Gerhard. Disponible para acceso público en: <http://www2.cs.uregina.ca/~gerhard/publications/TRdbg-Pitch.pdf>
- [7] Especificación del protocolo MIDI de interconexión de instrumentos musicales. Enlace a la página del estándar: <https://www.midi.org/>
- [8] "Defining Analog Voice" Nota técnica sobre, canales de comunicación de Cisco. <http://www.cisco.com/c/en/us/support/docs/voice/h323/8628-define-analog-voice.html>

Unidad de adquisición de sensores con aprovechamiento de recursos de hardware

Gastón Fabbietti¹ – Martín E. Morales¹ - Sergio G. Saluzzi¹ - Ariel Dalmas Di Giovanni¹ – Daniel A. Pastafiglia¹.

Laboratorio de Técnicas Digitales - Departamento de Electrónica Aplicada
Instituto de Investigaciones Científicas y Técnicas para la Defensa (CITEDEF)
Villa Martelli, Buenos Aires, Argentina

¹{gfabbietti,mmorales,ssaluzzi,adigiovanni,dpastafiglia}@citedef.gob.ar

Abstract—El presente trabajo describe el diseño, desarrollo e implementación de una electrónica de a bordo orientada a un vehículo aéreo no tripulado. La misma cuenta con características de adaptabilidad y escalabilidad para otros vehículos aéreos no tripulados similares, para vehículos con dinámicas más exigentes del tipo cohetes sonda y/o para nano plataformas satelitales. Dentro de la mencionada electrónica se destaca una unidad de sensado identificada como Unidad de Adquisición, que es la encargada de adquirir la información de distintos tipos de sensores abordados, agrupar dicha información en una trama de datos que será enviada a la unidad de procesamiento de a bordo, y almacenar en tiempo real estos datos obtenidos. Se destaca el desarrollo del firmware que está orientado a utilizar los recursos de hardware disponibles en el microcontrolador para disminuir la carga de cómputo de la Unidad de Procesamiento.

Keywords—VANT, Cortex-M4, sensores, arquitectura de firmware, DMA, SDIO, hardware dedicado.

I. INTRODUCCIÓN

En los últimos tiempos los drones o VANTs (vehículo aéreo no tripulado) pasaron de ser temas exclusivos de áreas de Defensa, o de investigación y desarrollo a ser accesibles por cualquier persona para un uso particular. Sin embargo, en el ámbito de la investigación y desarrollo continúa siendo una temática sobre la que se trabaja fuertemente ya que permite el desarrollo de capacidades específicas en múltiples disciplinas como la aeronáutica, la electrónica, sistemas embebidos, software, mecánica, entre otras. Dichas disciplinas están directamente relacionadas con las aplicaciones en coherencia y el ámbito espacial.

En los vehículos aéreos no tripulados es necesario contar con sistemas electrónicos a bordo, compuestos de unidades de procesamiento específicas y una unidad de instrumentación de un conjunto de sensores, que permita determinar los parámetros de navegación del mismo, es decir: posición, velocidad y actitud, como se menciona en [1].

En el marco del proyecto PIDDEF 01/ESP/15/BAA, se plantea como objetivo desarrollar una electrónica de a bordo orientada a un vehículo aéreo no tripulado. La misma se compone de dos unidades principales: una unidad de control de actuadores y comunicaciones, identificada como unidad de

procesamiento, y una unidad de adquisición de sensores de a bordo, identificada como unidad UAA, la cual es el objeto del presente trabajo.

La Unidad debe satisfacer los siguientes requerimientos funcionales: adquirir la información de distintos tipos de sensores abordados en un tiempo de muestreo determinado y periódico, disponer dicha información en una trama de datos para ser enviada a la unidad de procesamiento de a bordo, y almacenar la trama en un medio físico extraíble.

Los sensores necesarios son: una unidad inercial de mediciones, compuesta de magnetómetros, acelerómetros y giróscopos en tres ejes; un sensor de velocidad del aire basado en tubo de pitot; un sensor de altitud barométrico; un sensor de altura ultrasónico; un sensor de temperatura y un GPS (del inglés: *Global Positioning System*).

A su vez tanto la unidad UAA como las otras unidades del Proyecto deben cumplir con requerimientos no funcionales tales como: ser concebidas en forma escalar, reutilizable y con posibilidad de adaptarse sobre otras plataformas, distintas de un VANT específico, como por ejemplo un vector tipo sonda, un vehículo terrestre no tripulado, una plataforma nano satelital, entre otros. En forma práctica, este requerimiento implica entre otras cosas que la unidad sea configurable en términos de frecuencia de muestreo, o flexible a la incorporación o cambio del tipo de sensor utilizado sin que perjudique la performance temporal de la misma. Queda excluido de esta consideración el aspecto de robustez mecánica y de tolerancia a fallas que algunas aplicaciones específicas puedan exigir.

En función de todos los requerimientos se optó por desarrollar la unidad UAA en un dispositivo que integre los sensores, con un microcontrolador, tomando como referencias los trabajos [2], [3], y la experiencia previa en un desarrollo de similares características [4].

El microcontrolador seleccionado para tal fin es un Cortex-M4F de la empresa ST-Microelectronics [5], el cual posee una buena cantidad y variedad de interfaces de comunicaciones (I2C (del inglés: *Inter-Integrated Circuit*), SPI (*Serial Peripheral Interface*), UART (*Universal Asynchronous Receiver-Transmitter*), entre otros) que permiten vincular los sensores, un periférico SDIO (*Secure Digital Input Output*)

para interface de tarjetas tipo SD, y controladores DMA (*Direct access memory*). Uno de los aspectos importantes del trabajo es realizar el uso más eficiente de periféricos a fin de liberar la carga de la CPU (*Central Processing Unit*), de tareas asociadas directamente con periféricos.

El trabajo continúa en la sección II, donde se presenta la metodología de trabajo, en la sección III, el detalle del desarrollo, y en la sección IV se presentan los ensayos y resultados obtenidos.

II. METODOLOGÍA

La unidad UAA por lo antes expuesto involucra diseño y desarrollo de hardware, es decir la plataforma que integra el microcontrolador con los sensores, y de firmware, el programa que se ejecuta en el microcontrolador.

Dado que la mayor carga de trabajo estaría en el firmware, se consideró basar la plataforma microcontrolada de hardware en un kit comercial, con una placa ad-hoc para los sensores. De esta manera se dispondría de una plataforma para el desarrollo y evaluación de las rutinas de firmware que controlan y comandan cada sensor. Una vez verificado el funcionamiento se desarrollará un único circuito impreso que integra a los sensores y al microcontrolador, es decir prescindiendo del kit. Este último aspecto se encuentra en una etapa de diseño, y por tanto no forma parte del presente trabajo.

Para el desarrollo del firmware se fijaron pautas para garantizar el correcto funcionamiento del mismo, la rápida detección de errores, la modularidad, escalabilidad y reutilización.

La arquitectura del firmware se basó en la arquitectura *Foreground / Background* según se define en [6], en complemento con un modelo basado en capas de abstracción, tal como el que se muestra en Fig. 1. El objetivo del mismo es lograr la independencia de la plataforma de hardware utilizada. Las capas o niveles se comunican sólo con la capa inferior o superior, para mantener la portabilidad buscada.

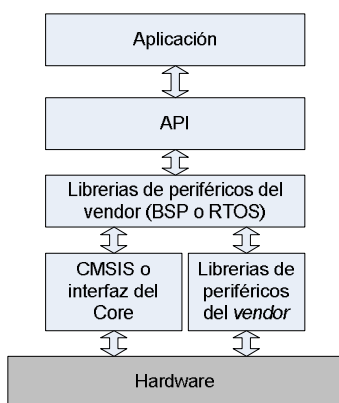


Fig. 1. Modelo de capas de abstracción.

La capa de aplicación es el programa de usuario que debe cumplir los requerimientos específicos, en este caso las funcionalidades de la unidad UAA. La capa API (*Application Programming Interface*) define las bibliotecas de código, que

también tienen como objetivo la independencia respecto del hardware utilizado. Estas bibliotecas contienen un conjunto de funciones que pueden utilizarse para diferentes usos. Este nivel es el que posee las rutinas reutilizables del desarrollo, en vistas de otras aplicaciones futuras. Las dos capas intermedias entre la API y el hardware son las que trabajan con los periféricos. Puede ser una capa tipo paquete de soporte (BSP: *Boards Support Package*), que interactúa con las librerías propias del fabricante (*vendor*) del microcontrolador, en este caso ST-Microelectronics y con el estándar dado por el diseñador del core, que en este caso por tratarse de una Cortex-M, el estándar es el CMSIS (*Cortex M Software Interface Standard*).

Finalmente, en complemento con la metodología antes detallada se utilizaron las pautas de proyecto para la codificación y documentación automática de forma tal que los códigos generados, que son parte del entregable del proyecto, sean consistentes independientemente de que sean generados por distintas personas. Para las reglas de codificación se adoptaron algunas de las reglas establecidas en [7], en conjunto con algunas reglas propias del laboratorio; y para la documentación automática se utilizó la herramienta Doxygen [8].

III. DESARROLLO

A. Desarrollo del hardware

Como se mencionase anteriormente la primer plataforma de hardware se basó en dos placas independientes, una basada en un kit comercial con el microcontrolador, y otra propietaria que integra los sensores, las cuales se vinculan una sobre otra mediante conectores estándar de 100 *mils* de paso, como se aprecia en la Fig. 2.

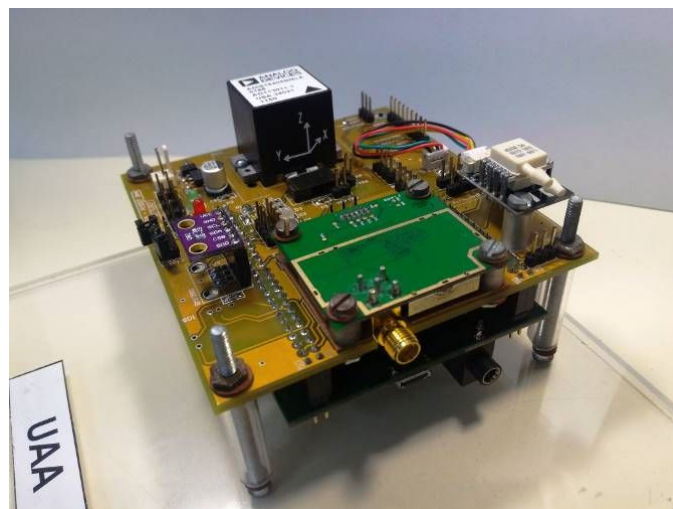


Fig. 2. Vista del primer prototipo.

El kit comercial utilizado es el STM32F4Discovery [9], que contiene un microcontrolador Cortex M4 STM32F407, una vista del kit se muestra en la Fig. 3.

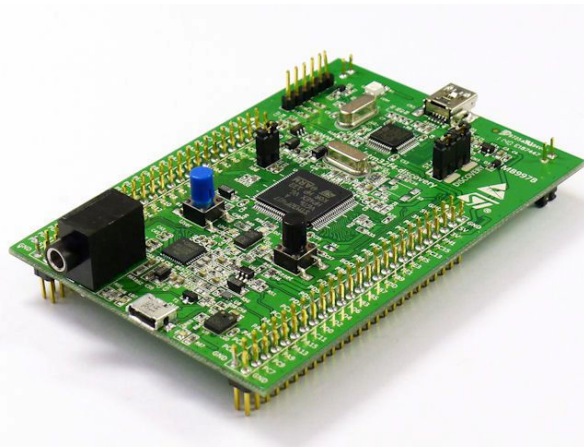


Fig. 3. Vista del Kit Discovery.

Por otro lado, la placa de sensores fue desarrollada en particular para el proyecto e integra los siguientes módulos: un receptor GPS MT-332, una unidad inercial de mediciones IMU (*Inertial Measurement Unit*) ADIS16405, un sensor de altitud barométrica y temperatura BMP280, un sensor de altura por ultrasonido y un sensor de velocidad del viento tipo tubo de pitot basado en un sensor de presión 4525DO. Además, incorpora los conectores para las entradas analógicas auxiliares vinculadas al convertor analógico digital del microcontrolador, las entradas digitales para estados generales de la electrónica de a bordo y el zócalo de la memoria SD. En la Fig. 4 se muestra una vista de la placa de sensores.

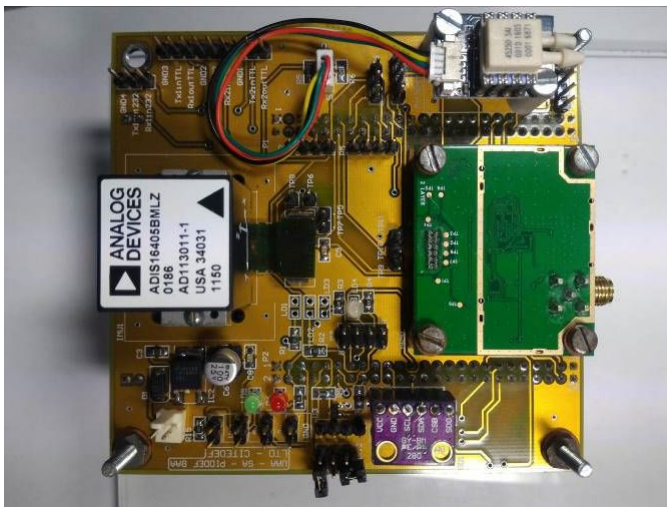


Fig. 4. Placa expansora de sensores.

B. Formato de trama

Como se comentó anteriormente, los datos obtenidos de los distintos sensores se agrupan en una trama, cuyo formato fue definido para poder corroborar pérdidas de tramas y validar la información, sobre todo luego de la transmisión inalámbrica de la misma. La trama de datos, como se aprecia en la Fig. 5, es de longitud fija: 64 bytes, posee un encabezado predefinido, un contador de secuencia de trama, ambos de 16 bits, un *checksum* de 8 bits, una carga útil de 59 bytes.

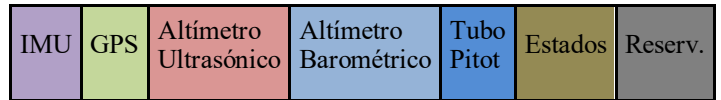
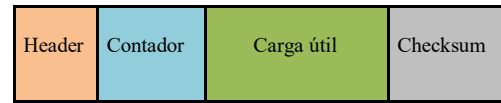


Fig. 5. Formato de trama de datos: Completa y descripción de la Carga útil.

C. Desarrollo del firmware

En función de lo expresado en la Sección Metodología, se utiliza la arquitectura de *firmware* para que cada tarea opere en forma cooperativa, sin utilizar lazos bloqueantes, conforme la experiencia obtenida en el trabajo [10].

Desde el enfoque del modelo de capas presentado en la Fig. 1, se utilizó para el nivel más bajo el estándar CMSIS y la librería de periféricos provista por el fabricante del microcontrolador [11]. Se desarrollaron librerías en las capas superiores: Aplicación, API y BSP. La capa de Aplicación se compone de las siguientes tareas:

- **Adquisición Sensores:** obtiene con una cadencia periódica controlada por el *Scheduler* del sistema los datos de los distintos sensores, a la vez que controla un *timeout* asociado a cada sensor para evitar demoras no deseadas en esta rutina.
- **Trama Armar:** una vez finalizada la adquisición, carga en la cola de transmisión los campos que conforman la trama, definiendo la palabra de sincronismo, actualizando el Contador y calculando el *Checksum* correspondiente según los datos obtenidos de los sensores en ese slot de tiempo.
- **Transmision Trama:** habilitando la interrupción de transmisión del periférico, y siempre que haya datos cargados en la cola de transmisión, envía la trama hacia la unidad de control y comunicación.
- **Recepcion Trama:** inicia la máquina de estados encargada de recibir tramas de comando y validar la misma verificando el *checksum*.
- **Decodificación:** recibida y validada la trama, obtiene el comando recibido y prepara el sistema para su ejecución.
- **Ejecución:** detiene el proceso de adquisición de sensores, genera la primera respuesta de *acknowledge*, ejecuta el comando recibido, envía la segunda respuesta de *acknowledge* con el resultado de la ejecución del comando y vuelve a iniciar el sistema al modo de funcionamiento correspondiente.

- **Supervisor:** controla los errores del sistema, ya sea si existiese en los distintos periféricos, en los sensores o en la SD, para notificarlos en la siguiente trama enviada.
- **Corrector:** rutina que corrige los errores encontrados sobre el sistema.
- **SD Proceso:** máquina de estados encargada de administrar el funcionamiento y uso de la memoria SD, permitiendo su escritura, lectura y borrado.

En la capa API el trabajo más intenso estuvo en desarrollar el manejador para cada sensor, que se comunica con una cola de datos con la capa de Aplicación, de esta manera si se cambia un sensor, dicha placa no se altera y a su vez si el sensor se usa en otra plataforma el código es portable.

D. Aprovechamiento de hardware

Desde el enfoque temporal, para cumplir los requerimientos, la unidad UAA debe transmitir una trama de información válida cada 5ms, cadencia muy superior a la necesaria para controlar la aeronave del proyecto, ya que según estudios de los especialistas aeronáuticos debe rondar los 100ms. A pesar de esta condición, se desarrolló la UAA con esta exigencia para poder utilizarla en vehículos que posean una dinámica superior.

Cada sensor posee un protocolo de comunicaciones e interfaz propia: algunos de ellos reportan por consulta, otros en forma autónoma, y en algunos casos aceptan ambas opciones. De los sensores enumerados en esta sección, podemos realizar la siguiente catalogación, según su operatoria:

- el GPS opera por reporte autónomo, sin necesidad de interrogarlo, entregando datos cada 1 segundo.
- la IMU debe interrogarse: se le envía un comando específico utilizando el protocolo SPI y responde con 24 bytes en un tiempo menor a 0,3 ms.
- el resto de los sensores responden contra un comando específico y la respuesta es de una cantidad de información reducida, con una cadencia mínima de 100 ms.

El único sensor que puede ser muestreado cada 5ms es el inercial, el resto posee tiempos de muestreo superiores. Por tanto, se decidió manejar todos los sensores, excepto el inercial, mediante un sistema de encuesta y respuesta, usando un sistema de interrupciones tradicional.

Para la adquisición del sensor inercial utilizando el periférico de comunicaciones SPI, se asoció un canal DMA que almacena los 24 bytes que reporta el sensor y luego genera una interrupción. De esta manera, la CPU no debe atender constantemente las interrupciones generadas por cada byte recibido vía SPI, lo cual generaría una gran demanda durante los 0,3 ms que demora la adquisición, sino que sólo atiende una interrupción por período de adquisición.

La otra tarea crítica, desde el enfoque de los recursos, fue el manejo de la memoria SD como se ha comprobado en [12]. Se corroboró que es más óptimo guardar la información en forma

de bloques que byte a byte, y que el uso de periféricos específicos como el SDIO aumenta la tasa de transferencia y baja el costo de cómputo de la CPU, sobre todo ante el problema de que las memorias SD presentan demoras aleatorias en la respuesta a determinados comandos. Para el manejo masivo de datos, es decir de los bloques de datos, se utilizó otro canal DMA asociado al periférico SDIO. El resultado obtenido fue que la tarea de almacenado en la SD se comporta como una tarea más del sistema, sin perjudicar la performance del mismo.

Este último concepto es vital para asegurar el determinismo temporal del sistema de adquisición y envío de datos con una cadencia específica.

IV. ENSAYOS Y RESULTADOS

A. Ensayos

Para verificar el funcionamiento de la unidad se evaluó la integridad de datos de la trama de salida, mediante el encabezado y el *checksum*, y la pérdida de tramas completas mediante el contador de secuencia.

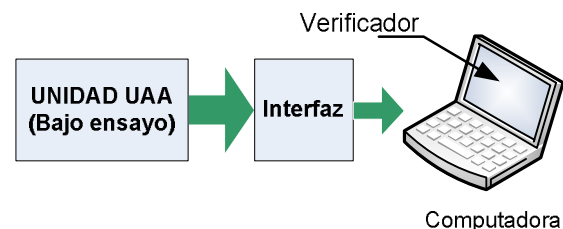


Fig. 6. Esquema de ensayos.

Para esto se dispuso del esquema de la Fig. 6, una computadora conectada al canal de salida de la unidad de interfaz EIA-232 compatible, con un software ad-hoc, que verifica los parámetros antes descritos en forma automática y almacena en disco los datos recibidos. En la Fig. 7 se observa el formulario visual de la aplicación de software, que contabiliza y presenta en pantalla la cantidad de tramas recibidas, la cantidad de datos erróneos por pérdida de secuencia y por error de *checksum*.

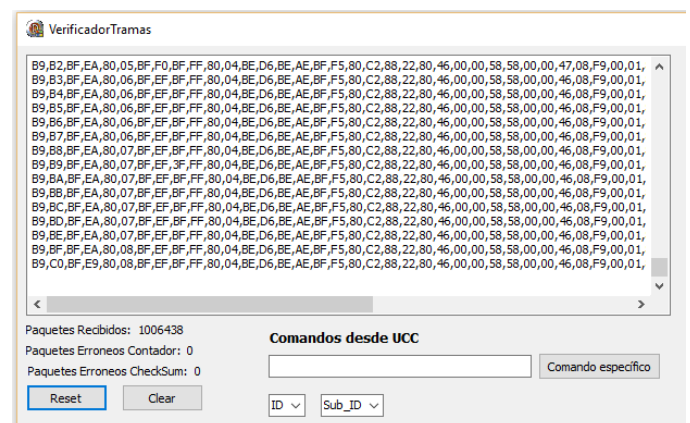


Fig. 7. Aplicación de verificación.

El otro paso de la verificación consistió en analizar la integridad y congruencia de los datos contenidos en la trama, es decir la información de datos de cada sensor.

Este modelo de ensayos tiene la ventaja de ser repetible, lo que permitió análisis parciales durante todo el desarrollo en búsqueda de la detección temprana de errores. Por otro lado, el hecho que la verificación sea automática elimina el error humano de observación como factor de error.

B. Resultados

En los ensayos realizados en el laboratorio durante más de dos horas de operación no se registraron pérdidas de paquetes, por lo tanto, la probabilidad de que ocurra esto en una aplicación real es muy baja.

En la Fig. 8, Fig. 9 y Fig. 10 se observan los resultados de una medición de aceleración, una de velocidad angular y una de altitud barométrica, procesados en una etapa posterior utilizando *scripts* de MatLab desarrollados para tal fin. El objetivo de graficar las mediciones de los sensores fue visualizar que midan de manera razonable, además de corroborar que no se produzcan datos erróneos durante el proceso de adquisición y armado de trama en la UAA.

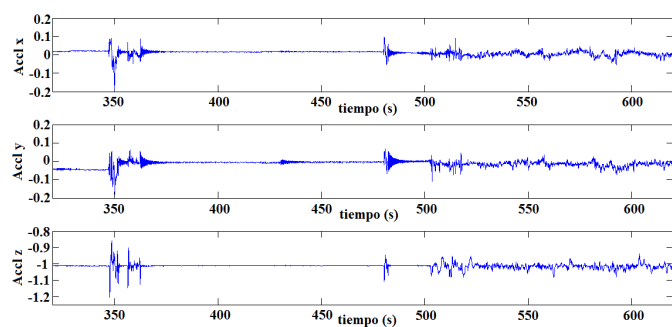


Fig. 8. Datos de aceleración

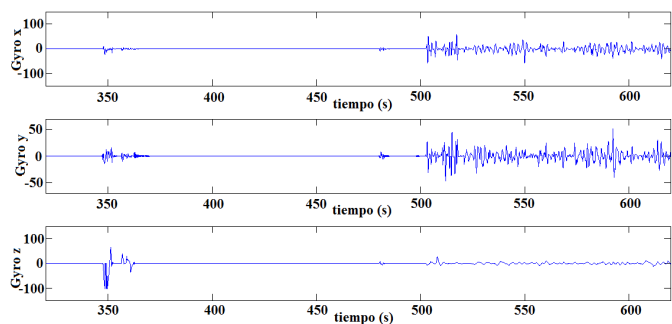


Fig. 9. Datos de velocidad angular.

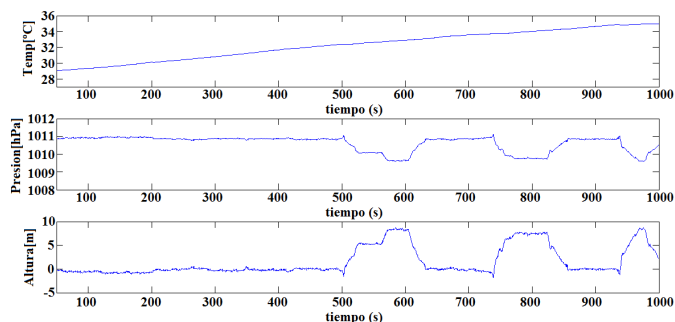


Fig. 10. Datos de altímetro barométrico.

A su vez, se verificaron las características temporales buscadas mediante un osciloscopio y unas banderas (*flags*) de código. Se midió la estabilidad del tiempo de adquisición y de envío de información. En la Fig. 11 se observan las mediciones de los tiempos de adquisición de los sensores, Trazo 3, y los tiempos de transmisión y el período de envío de trama, Trazo 4.

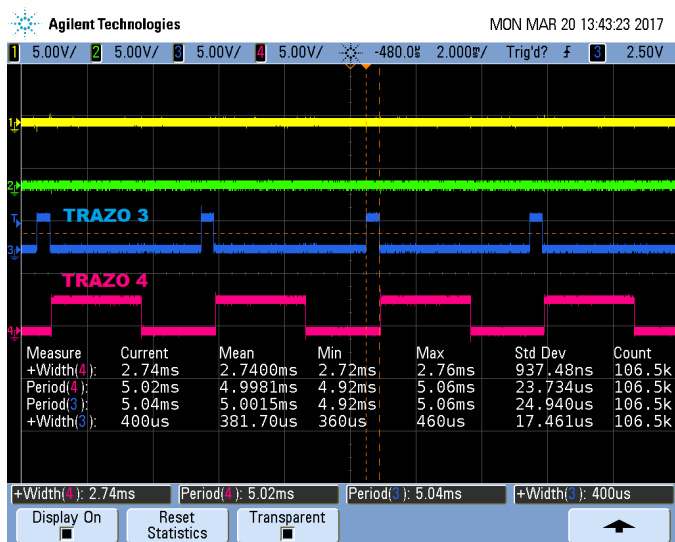


Fig. 11. Medición de tiempos.

Puede observarse la medición **+Width(4)**, la cual hace referencia al tiempo que demora la transmisión de los 64 bytes de la trama de datos. En promedio, unos 2,74 ms, transmitiendo a 230400 bps 8-N-1. La medición **Period(4)** especifica cada cuánto tiempo se comienza a transmitir una trama, que está dentro de los requerimientos establecidos, 5 ms, midiendo sobre la tarea Transmisión Trama descrita anteriormente. La medición **Period(3)** describe lo mismo, pero midiendo sobre la orden que inicia de adquisición de sensores. Por último, la medición **+Width(3)** demuestra que la tarea Adquisición Sensores demora como máximo 400 μ s, sin generar el acarreo de tiempos que puedan afectar a la cadencia de transmisión deseada.

Se observa en todos los casos, que el desvío estándar no tiene valores considerables que puedan alterar el correcto funcionamiento del sistema.

V. CONCLUSIONES

El diseño basado en la utilización de los recursos de hardware disponibles en un microcontrolador moderno posibilitó la liberación de la CPU de aquellas tareas en las cuales había una gran dependencia con periféricos, obteniendo como rédito que la CPU no deba atender sucesivas interrupciones de los periféricos, disminuyendo las latencias de las tareas de procesamiento. Esto permite dejar recursos de CPU libres para futuras expansiones de la unidad como por ejemplo: procesamiento de sensores.

En el caso puntual del sistema de almacenamiento, quedó demostrado que el uso del periférico SDIO evita que las variaciones temporales aleatorias en respuesta a determinados comandos de las memorias SD, no impacten en el sistema.

A su vez y en función de la metodología seguida, el firmware, las librerías y módulos desarrollados podrán utilizarse en otras aplicaciones donde sea necesario un sistema de instrumentación de a bordo, como por ejemplo nano plataformas satelitales, vectores tipo sonda, sistemas de adquisición de datos (*dataloggers*), por citar potenciales aplicaciones.

Como trabajos futuros se buscará bajar el tiempo de adquisición, de forma tal de obtener datos cada 2 ms pensando en la utilización de la UAA en sistemas con dinámicas más exigentes. Por otro lado, se buscará optimizar aún más el uso de periféricos en aquellas tareas en las cuales actualmente se conserva su uso mediante interrupciones simples, o por verificación (*pooling*) no bloqueante.

Finalmente, se pasará a producción y verificación el prototipo que integra al microcontrolador y los sensores en un único circuito impreso.

AGRADECIMIENTOS

Se agradece a todo el grupo de trabajo del PIDDEF 01/ESP/15/BAA que ha posibilitado que este desarrollo se concrete.

REFERENCIAS

- [1] Martín España, "Fundamentos de la navegación integrada". AADECA. 2010.
- [2] Claudio Pose, Federico Roasio, Juan Giribet, "Diseño de una computadora de navegación, guiado y control: Aplicación a un vehículo aéreo no tripulado". Actas de las VIII Jornadas Argentinas de robótica, JAR VIII 2014.
- [3] Alan Kharsansky, Ariel Lutenberg, "Diseño e implementación de un sistema embebido de control de actitud para aeronaves no tripuladas". Libro de memorias del Congreso Argentino de Sistemas Embebidos CASE 2013, 2013.
- [4] Edgardo A. Comas, Daniel A. Pastafiglia, Cristian R. Bruña, Ariel Dalmas Di Giovanni, Martín E. Morales, Agustín Dal Lago, Mauricio Burgos, "Sensado INS/GPS con monitoreo en tiempo real". Libro de memorias del Congreso Argentino de Sistemas Embebidos CASE 2013, 2013.
- [5] ST-Microelectronics, "RM0090 Reference Manual for STM32F405/415, STM32F407/417, STM32F427/437 and STM32F429/439 advanced ARM®-based 32-bit MCUs - DocID018909". Rev.13, 2016.
- [6] Gustavo Galeano, "Programación de Sistemas embebidos en C". Alfaomega, 2009.
- [7] National Aeronautics and Space Administration- NASA, "C Style Guide".1999.

- [8] Doxygen. [Online]
Disponble: www.doxygen.org.
- [9] ST-Microelectronics, "UM1472 User Manual - Discovery kit with STM32F407VG MCU - DocID022256". Rev.5, 2016.
- [10] Ariel Dalmas Di Giovanni, Daniel A. Pastafiglia, Raúl Cristian Bruña, Edgardo A. Comas, "Controlador para actuador de posición". Libro de memorias del Congreso Argentino de Sistemas Embebidos CASE 2015,2015.
- [11] ST-Microelectronics, "STSW-STM32065 STM32F4xx standard peripherals library - DocID028330". Rev.1, 2016.
- [12] García, Pablo Esteban. "Proyecto grabación de datos de alta velocidad sobre memoria SD". Congreso Argentino de tecnología espacial CATE2015, 2015.

Antenna Motion Control System with Ethernet Connectivity

D. Badino, J.P. Rumie Vittar
Centro de Investigación y Desarrollo de Tecnologías
Aeronáuticas (CITeA) Fuerza Aérea Argentina.
Las Higueras, Córdoba, Argentina.
jvittar@faa.mil.ar

D. Primo, M. Escobar, D. Diaz
Grupo de Sistemas de Tiempo Real (GSTR). Facultad
de Ingeniería. Universidad Nacional de Río Cuarto
Río Cuarto, Córdoba, Argentina
{dprimo, mescobar, ddiaz}@ing.unrc.edu.ar

Abstract—The present paper describes the design, development and implementation of the hardware and software of an embedded system for the control of movement of a pedestal for a communications antenna with the objective of improving range of communication between an earth station and an air mobile station. The system was designed with the intention of using COTS (Commercial Off-The-Shelf) components. Finally, the performance of the developed system was evaluated by performing measurements in the error of its azimuth and elevation positions to evaluate the accuracy of the encoder using statistical techniques.

Index Terms—ACU; Telemetry; Ethernet; Control; Protocol.

I. INTRODUCTION

The following describes the design of an embedded system for the control of movement of an antenna pedestal to track aircrafts. This development is implemented in the Aircraft Assessment and Training System (abbreviated to SEAA in Spanish) (Fig. 1), which is composed of two well-defined subsystems: the Capture and Transmission System (abbreviated to SCT in Spanish) and the Analysis and Evaluation System (abbreviated to SAE in Spanish) [1] [2]. The SCT is responsible for the acquisition and transmission of flight parameters, events, and other important parameters of the aircraft, to the SAE. The main function of the SAE is to display the evolution of the exercise in a visualization software, both in 2D and 3D.

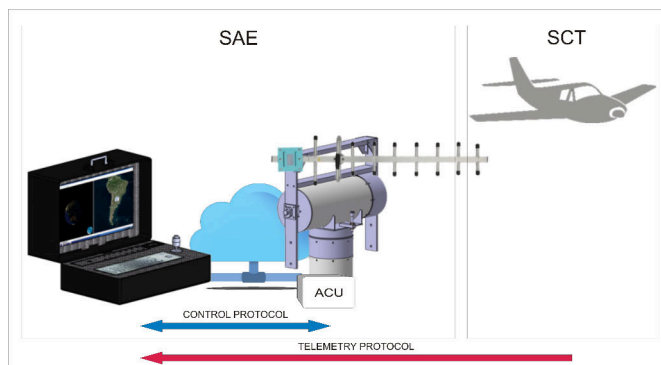


Fig. 1. Aircraft Assessment and Training System (SEAA)

With the objective of improving the radio coverage of the communication between the SCT and the SAE, a control

system called ACU (Antenna Control Unit) is implemented in the SAE for tracking mobile targets.

The SCT sends telemetry to the SAE through a proprietary protocol which contains, among other data, Latitude, Longitude and Altitude information of the aircraft.

In these systems, the type of antenna used affects the radio range and can increase the distance of communication. Due to particular issues, in the presented case, the SCT can only have an omnidirectional antenna which deploys energy equally in all directions, so it is not suitable for long range operation. Directional antennas emit the signal in one direction and therefore can reach greater distances with equal power in the transmitter. This can be analyzed by keeping in mind that the radiation lobe of a directional antenna compared to an isotropic antenna (Fig. 2) allows for a higher gain in a given direction.

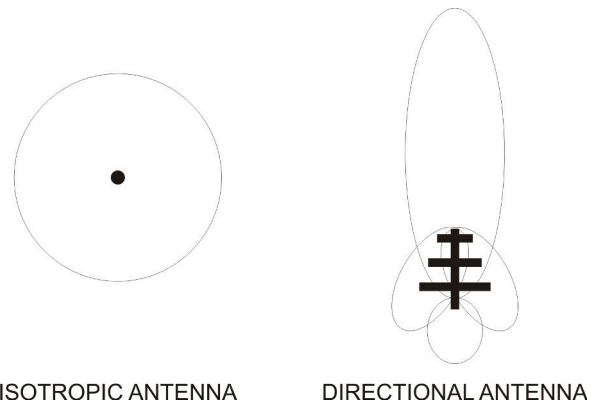


Fig. 2. Radiation diagram of an omnidirectional antenna and a directional antenna

II. BRIEF OVERVIEW OF THE ANTENNA

Since the SCT works at a frequency of 915MHz , a 15-element yagi antenna and a gain of 14.23dbi were selected for the pedestal, whose radiation lobe is shown in Figure 3. To calculate the radio link's distance the software Radiomobile [3] was used. The SCT was configured with a transmission power of 1W [2] and an omnidirectional antenna of 3dB ; with the directional antenna of 14.23 dB , a radioelectric distance of 97.11 km was calculated and in real tests the link reached

100 km. These values exceed by far, the simulated values for an omnidirectional antenna of 5 dB placed on land, which do not exceed 30 km.

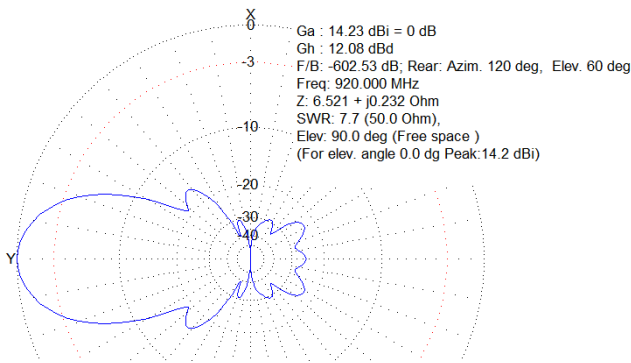


Fig. 3. Radiation diagram of the directional antenna used

III. RESOURCES

The tracking system in question basically consist of:

- Pedestal for antenna anchorage
- Electronics associated with control and monitoring

A. Pedestal for antenna anchorage

The pedestal for anchoring the antenna, is a mobile structure with two degrees of freedom (azimuth and elevation), which, by means of DC motors, encoders, and switches, is possible to control the pointing direction. It allows a rotation of 360 degrees for azimuth, and 152 degrees for elevation (from 14 to 166) as shown in Figure 4.

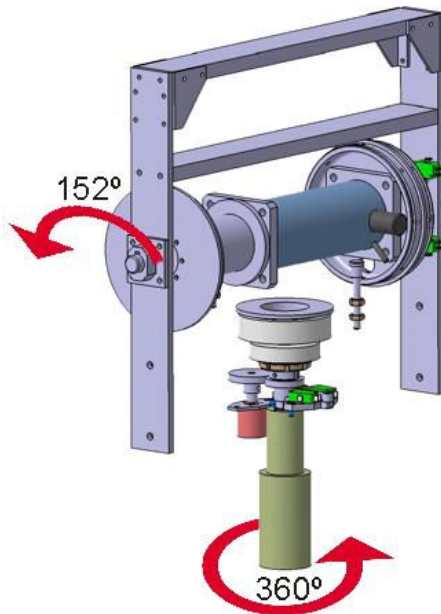


Fig. 4. Mechanical representation of the pedestal

Both axes of rotation are coupled through gears to their respective encoders. The encoders are of the incremental type and allow to infer the position by counting the pulses generated. According to the manufacturer's data sheet, a complete revolution around its axis generates 1000 pulses.

1) *Precision in Azimuth:* As mentioned, the encoder is coupled to the axis of rotation by a gear ratio such that each revolution of the motor, turns the encoder a $40/48^{th}$ of a revolution. The number of pulses that will be generated by the encoder when rotating 360 degrees of azimuth, taking into account that the encoder has a resolution of 1000 pulses per lap, is represented by the following formula:

$$ppl_{az} = 1000(pulses) * 40/48 = 833.33(pulses)$$

Where ppl_{az} is the total number of pulses for a complete lap of azimuth.

If for every 360 degrees of azimuth 833.33 pulses are generated, the number of degrees of azimuth per pulse can be calculated by the following:

$$Gpp_{az} = 360^\circ * 1(pulse)/ppl_{az} = 0.432^\circ$$

Where Gpp_{az} are the degrees of azimuth for each pulse delivered by the encoder.

Although it is possible electronically to achieve resolutions lower than one degree, due to the mechanical design, the pedestal has a free play of approximately 4 degrees for azimuth. For this reason the control firmware only performs movements that involve at least 4 degrees.

2) *Precision in Elevation:* Similarly for elevation, whose gear ratio is $81/48$ and considering the encoder's resolution:

$$ppl_{el} = 1000(pulses) * 81/48 = 1687.8(pulses)$$

Where ppl_{el} is the number of total pulses for a complete lap of elevation.

The number of degrees of elevation per pulse that the encoder generates is:

$$Gpp_{el} = 360^\circ * 1(pulse)/ppl_{el} = 0.213^\circ$$

Where Gpp_{el} are the degrees of elevation for each pulse delivered by the encoder.

As the elevation can only vary in 152 degrees, at most 712 pulses are counted. The elevation also has the mechanical flaw that gives, in this case, a free play of approximately 2 degrees. As in the case of the azimuth, the control firmware will only perform movements that involve at least 2 degrees.

B. Electronics associated to the control and monitoring

To select a board, it was taken into account that it needed to have an Ethernet connection in order to be able to monitor and control it remotely. Additionally it need to have the necessary PWM outputs to control the motors and the GPIO ports to read the limit switches. The board selected is one of the line "Launchpad Tiva C Connected" manufactured by Texas Instruments [4], based on an ARM Cortex-M4F microcontroller [5] and has the following characteristics:

- MCU High Performance TM4C1294NCPDT
- CPU of 120 MHz 32-bits ARM Cortex-M4
- 1MB Flash, 256 KB SRAM, 6KB EEPROM
- 10/100 Ethernet MAC+PHY integrated.
- Dual 12-bit 2MSPS ADCs, PWMs
- USB H/D/O and various serial interfaces more.

This board controls the motors through a high current H bridge based on the BTS 7960 integrated circuit, whose main characteristics are:

- Operating voltage from 5 to 27 Volts (B+)
- PWM motor speed control up to 25 kHz.
- Forward and backward motion control.
- Current limit of 30 A.

The basic scheme of operation consists of a "monitor and control" (M&C) scheme, where SAE software sends two types of frames to the implemented ACU: control frames or state frames.

When the system is powered on, the first thing that must be done is to initialize the pedestal. An initialization command is sent by the SAE software to the ACU that triggers a procedure by which the ACU, using the limit switches of the pedestal, identifies the 0 degree position for azimuth and the 14 degrees position for elevation.

Once the pedestal is initialized, the ACU is ready to receive movement commands (Fig. 5). Before sending any command, it's necessary to have the approximate location of the aircraft to track; the SAE performs the calculations and decides where to "aim" the antenna. It sends the appropriate commands to the ACU using UDP packets and upon reception the ACU controls the motors to make the movements, verifying the position through the encoders. The data received by the antenna, is sent to the SAE for aircraft monitoring, also using UDP messaging. The sequence of messages can be seen in Figure 6.

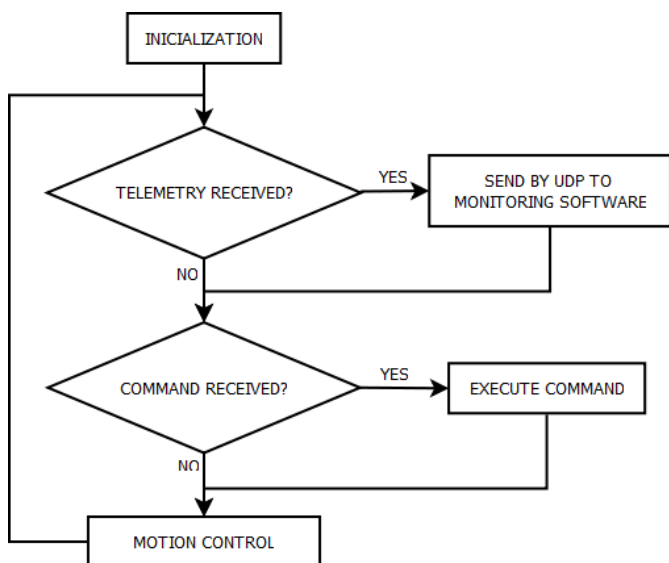


Fig. 5. Control Logic

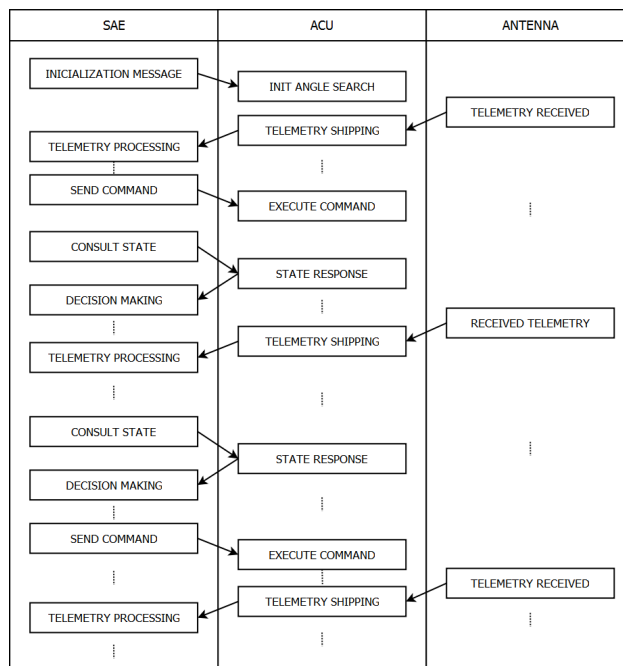


Fig. 6. Message Sequence

C. Control Protocol

The M&C scheme presented here corresponds to a master-slave structure, where communication is always initiated by the master, which in this case is the SAE control software. There are two types of frames: "command" frames, which carry movement instructions, and "state" frames, which allow the SAE to monitor the control that is being carried out. Control frames do not generate a response by the ACU, on the other hand state frames do. Thus, in order to confirm that a command was accepted and executed, a inquiry frame should be sent (Fig. 6).

The structure of both types of frames is similar, and basically consists of a header and subsequent data.

In Figure 7, the format of the frames can be seen, where the TYPE field indicates whether it's a command or a state frame, the field NUMBER CMD/STD indicates which command or status number it is, and the field DATA, carries the corresponding data depending on the type of frame.

1 BYTE	2 BYTES	VARIABLE LENGTH
TYPE	CMD/STD NUMBER	DATA
HEADER		PAYLOAD
UDP		
IP		
ETHERNET		

Fig. 7. Command frame format

There are initialization commands, configuration commands and movement commands (Table I). On the other hand, status messages allow to query the position status, the movement status, current configuration, and the overall status of the ACU. The response to these state frames, repeats the request header and include, in the data field, the parameters solicited by each request frame (Table II).

TABLE I
LIST OF COMMANDS

Command	Action
C00	Reset all registers
C01	Initialization
C02	Setting the desired azimuth and elevation angles, indicating their speed
C03	Moves the azimuth and the elevation indicating the speed without indicating a specific angle
C10	Set communication parameters with SAE software
C11	Reset parameters of communication with the SAE software to the default parameters
C90	Perform a Built in Test
C91	Erase error flags

TABLE II
LIST OF STATUS MESSAGES

Status	Action
S00	Request a general status
S01	Request current position of azimuth and elevation
S02	Request position and speed of azimuth and elevation
S03	Request azimuth and elevation motion status
S10	Set communication parameters with SAE software
S11	Request the communication parameters with SAE software
S90	Request the status of Built in Test
S91	Request the status of error flags

D. Telemetry Protocol

The frames received by the antenna are sent to the SAE software using UDP protocol. These types of frames do not respond to a M&C scheme like the control frames, but rather they are sent to the SAE software as soon as they arrive.

The format of these frames corresponds to the specified by the SEAA[1], and are not processed by the ACU as can be observed in Figure 5 and Figure 6.

IV. PERFORMANCE ANALYSIS

Tests were performed for azimuth and elevation and through statistical techniques parameters such as the mean value, the variance, the range and the interval of confidence [6] were obtained. In order to verify the performance of the ACU for azimuth, tests were performed with three azimuth angles that were used as reference. The ACU was issued a movement command for each of these angles and the resulting angle was measured with the encoder. The tests were repeated 10 times for each angle of reference and the measurements obtained are those shown in Table III.

TABLE III
SAMPLE OBTAINED FROM THE AZIMUTH ANGLE

Angle set by command: 45°	Angle set by command: 90°	Angle set by command: 180°
44	89	180
45	90	180
44	90	181
44	89	179
45	90	180
46	89	181
45	91	181
45	90	179
47	91	180
45	90	180

The differences between the angle selected by command and the angle measured by the encoder were calculated, and the following statistics summarizes the results:

Statistical Summary for Azimuth Error

- *Recount: 30*
- *Average: 0,0333333*
- *Standard Deviation: 0,808717*
- *Minimum: -2,0*
- *Maximum: 1,0*
- *Rank: 3,0*

Confidence Intervals for Azimuth Error: Intervals of Confidence of 99,0% for the mean $0,033333 \pm 0,406983[-0,37365; 0,440317]$

It can be seen that for the azimuth angle, the error value between the measured angle and the set value does not exceed 0.4 degrees 99% of the time. By performing an analysis similar to that done for the azimuth angle, but in this case for the elevation angle, the measurements obtained are those shown in Table IV.

TABLE IV
SAMPLE OBTAINED FROM THE ELEVATION ANGLE

Angle set by command: 45°	Angle set by command: 90°	Angle set by command: 135°
44	90	133
43	90	135
45	91	135
44	89	136
45	89	136
46	90	134
46	89	133
45	90	135
46	91	135
45	90	136

The differences between the angle selected by command and the angle measured by the encoder were calculated, and the following statistics summarizes the results.

Statistical Summary for Elevation Error

- *Recount: 30*
- *Average: 0,1333333*
- *Standard Deviation: 0,937102*
- *Minimum: -1,0*
- *Maximum: 2,0*
- *Rank: 3,0*

Confidence Intervals for Elevation Error: Intervals of Confidence of 99,0% for the mean $0,133333 \pm 0,47[-0,34; 0,60]$

It is noted that for the elevation angle, the error value between the measured angle and the set value does not exceed 0,47 degrees 99% of the time.

V. CONCLUSIONS

From the results it can be seen that the azimuth and elevation errors that are obtained between the angle set value and the angle to which the system is displaced is negligible compared to the error due to the free play of the gears. Additionally, the width of the radio beam of the selected antenna contributes to compensate for the mechanical error.

Thanks to the development of this ACU, it was possible to extend the range of coverage and the "training zone" of the aircrafts, by tripling the distance at which the system can capture data.

This design works for the so-called air-to-ground communications, where a line of sight is required between the transmitter and the receiver. This design uses $915MHz$ to send telemetry, thus avoiding interferences since it is a proprietary band.

Tests performed on the preciseness of the azimuth and elevation angle of the pedestal were done without the yagi antenna in order to evaluate the developed ACU. The inertia produced by the antenna due to its weight and shape is

negligible and the tests carried out can be considered as valid for whole intended system.

VI. FUTURE WORK

The next phase of the development of the system will focus on giving the system the ability to predict the trajectory of the mobile targets based on statistics calculated from recorded data. This ability is important to maintain the overall stability of the system when there is a momentary lost of communications between the SCT and the SAE.

REFERENCES

- [1] D. Daz; F. Escobar; J. Oviedo; J. P. Rumie Vittar; P. Solivellas. *Sistema para Evaluación de Adiestramiento de Combate (SEACO)*. pp 66. CASE 2014. Buenos Aires, Argentina. 2014. ISSN/ISBN: 978-987-45523-2-7.
- [2] J. P. Rumie Vittar, F. Escobar, D. Badino, G. Giugge. D. Primo, D. Diaz, P. Solivellas, N. Veglia, *Sistema de adquisición de datos de vuelo y transmisión de parámetros en tiempo real*. pp 85-86. CASE 2016. Buenos Aires, Argentina. 2016. ISSN/ISBN: 978-987-45523-8-9.
- [3] Radio Mobile. RF propagation simulation software. [Online] Avialable: <http://radiomobile.pelmew.nl/>
- [4] *Tiva C Series TM4C1294 Connected LaunchPad Evaluation KIT. EK-TM4C1294XL USER'S GUIDE*. March 2014 Revised October 2016 - Texas Instruments.
- [5] J. Yiu. *The Definitive Guide to ARM Cortex-M3 and Cortex-M4 processors. Third Edition.*, 18 Oct 2013. ISBN: 978-012-40808-2-9.
- [6] D. Montgomery; G. Runger; R. Pia Garca. *Probabilidad y estadística aplicadas a la Ingeniería*, 2004. 1 st. ed. Mexico: McGraw-Hill.

Sensor de Intensidad y Orientación de Viento Basado en Acelerómetro.

Lucrecia Inda, Carolina Hirschfeldt, Gustavo Monte

Universidad Tecnológica Nacional

Facultad Regional del Neuquén

Plaza Huincul, Argentina

Abstract – Este trabajo presenta un dispositivo diseñado para la medición simultánea de la intensidad y dirección del viento que utiliza un acelerómetro como forma alternativa y económica de sensar dicho fenómeno meteorológico. El sensor propuesto es muy robusto ante fuertes ráfagas de viento. Asimismo, el diseño del prototipo permite la versatilidad de ajuste para diferentes probabilidades de ocurrencia.

Keywords— sensor de viento; acelerómetro; sensor inteligente; microcontrolador; conectividad.

I. INTRODUCCIÓN

El presente trabajo describe el funcionamiento de un dispositivo diseñado con el fin de medir, en forma eficiente y económica, tanto la intensidad de viento como su dirección.

El principio de funcionamiento se basa en el movimiento pendular de un objeto que responde proporcionalmente ante los cambios de posición producidos por el viento. La adquisición de dicha posición permite obtener información sobre la velocidad y la orientación del viento. Para ello se utiliza una esfera que contiene en su interior un acelerómetro, posicionada en la parte inferior de una varilla. La misma se encuentra suspendida en un espacio abierto con una libertad de movimiento de 360°.

Los datos del acelerómetro se adquieren, procesan y envían a una página Web utilizando un microcontrolador, una PC funcionando como servidor y un módulo receptor/transmisor inalámbrico, Fig.1.

Se propone como una manera alternativa al anemómetro y debido al bajo costo de los componentes, el dispositivo final resulta muy económico. Además, variando el peso de la esfera contenedora, puede modificarse la sensibilidad del sensor para ser ajustado a diversos rangos de medición. El primer prototipo ensayado se observa en la Fig. 2. Consta de un tubo cilíndrico que termina en una esfera con la electrónica dentro, Fig. 3. La resistencia aerodinámica que presenta es independiente de la dirección del viento.

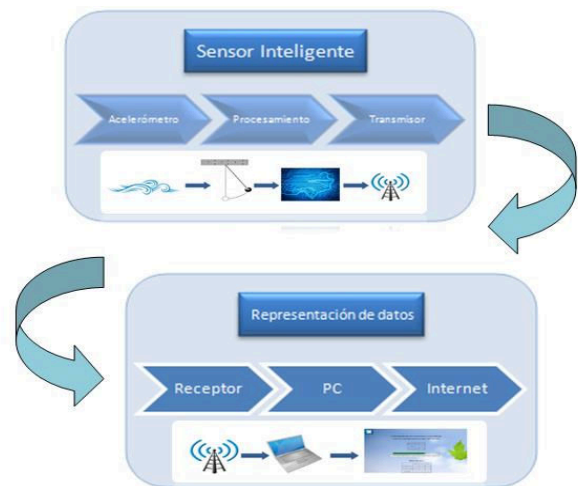


Fig.1. Diagrama de Bloques.



Fig.2. Diseño del prototipo.

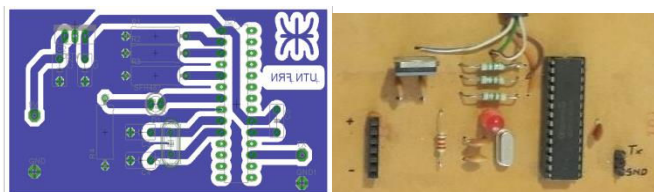


Fig. 3. Circuito del prototipo.

II. DESARROLLO

A. Principio de funcionamiento

Al estar posicionado en la parte inferior de la varilla, el acelerómetro refleja un movimiento pendular, por lo que se realiza una relación aproximada entre la intensidad de viento y la distancia presente respecto de la posición de reposo. Así, la intensidad de viento se obtiene midiendo la magnitud de separación entre la posición inicial del acelerómetro (reposo) y la posición final del mismo, una vez que es afectado por el viento, como se observa en la Fig. 4.

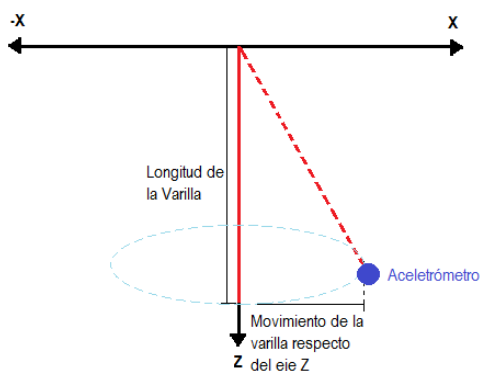


Fig.4. Movimiento de Varilla

Para ello se analizan los datos obtenidos al medir los ejes X e Y. Posteriormente, se obtiene dicha distancia calculando el módulo, la cual es proporcional a la velocidad.

$$|\text{Módulo}| = \sqrt{x^2 + y^2} \quad (1)$$

Esta fórmula permite obtener la velocidad independientemente de la dirección. Aplicando un factor de corrección λ (2) y teniendo en cuenta los valores iniciales del sensor (3), los cuales son distintos de cero, la fórmula resulta (4):

$$\text{Velocidad} = \lambda \cdot |\text{Módulo}| \quad (2)$$

$$|\text{Módulo}| = \sqrt{(x-x_0)^2 + (y-y_0)^2} \quad (3)$$

$$\text{Velocidad} = \lambda \cdot \sqrt{(x-x_0)^2 + (y-y_0)^2} \quad (4)$$

La dirección se obtiene analizando el ángulo entre el sensor y el plano horizontal X-Y que se encuentra en fase con los ejes cardinales: los ejes (-X, X) coinciden con el Este y

Oeste respectivamente, mientras que los ejes (-Y, Y) lo hacen con el Norte y Sur respectivamente como se observa en la Fig 5.

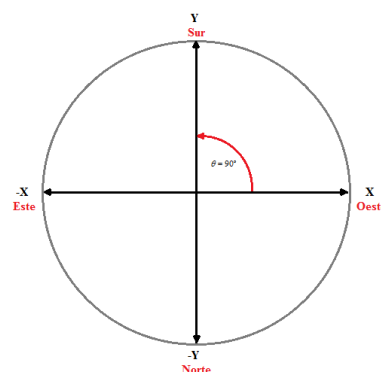


Fig. 5. Plano Horizontal

La fórmula utilizada para obtener el ángulo es la siguiente:

$$\theta = \arctan(y/x) \quad (5)$$

Teniendo en cuenta las condiciones iniciales del sensor la fórmula final resulta:

$$\theta = \arctan((y-y_0)/(x-x_0)) \quad (6)$$

B. Aspectos Constructivos del Sensor

Considerando la disposición de los ejes, se coloca el acelerómetro en el extremo inferior de una varilla con total libertad de movimiento. Se selecciona una varilla cilíndrica y una esfera para que el dispositivo presente la misma resistencia ante el flujo de aire en todas las direcciones. El acelerómetro elegido es el MMA7361L, el cual se encuentra disponible en un circuito impreso y tiene respuesta en CC, o sea funciona como inclinómetro, Fig. 6, [1].



Fig. 6. Acelerómetro MMA7361L.

El acelerómetro entrega la información en valores analógicos, aX, aY, aZ. La velocidad del viento se calcula de acuerdo al movimiento de la varilla: a medida que la varilla se separa de la posición de reposo, el acelerómetro refleja este cambio en la posición de acuerdo a las variaciones en los valores de X, Y, Z.

Si el ángulo calculado es 0° , entonces el viento sopla en dirección "Oeste". Se admite un determinado margen de error. En este caso si $\theta = 0^\circ \pm 10^\circ$ se considera en dirección "Oeste". Superando dicha tolerancia, la dirección cambia a

“Suroeste” o “Noroeste” según corresponda. En la Fig. 7 puede observarse los Puntos Cardinales y sus respectivos ángulos.

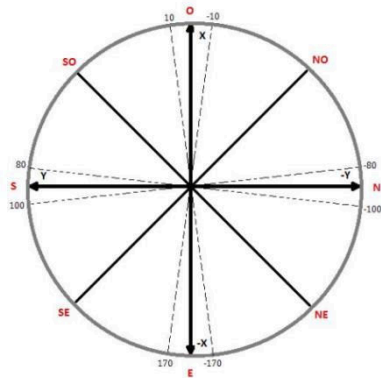


Fig. 7. Ángulos correspondientes a cada Punto Cardinal.

TABLA I. ÁNGULOS Y ORIENTACIÓN

Dirección del viento	Ángulo Mínimo (X-Y)	Ángulo Máximo (X-Y)
Oeste	-10°	10°
Sur - Oeste	10°	80°
Sur	80°	100°
Sur - Este	100°	170°
Este	170°	-170°
Noreste	-170°	-100°
Norte	-100°	-80°
Noroeste	-80°	-10°

El diseño del sensor se realiza analizando las velocidades promedio que ocurren en una zona determinada con mayor frecuencia, con el objeto de medir dichas velocidades con mayor precisión. Para ello se tienen en cuenta estudios relevados en la zona por la empresa Vortex e INTI [2][3].

En las Figs. 8 y 9 puede observarse tanto el gráfico de la rosa de vientos como el de probabilidades de ocurrencia de velocidades de viento en cercanías del establecimiento educativo UTN-FRN [4], [5], [6]. Las probabilidades de ocurrencia mayores se encuentran entre los 10.8 km/h y los 36 km/h.

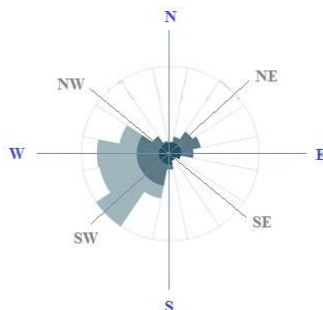


Fig. 8. Rosa de Vientos. Imagen extraída de Vortex.

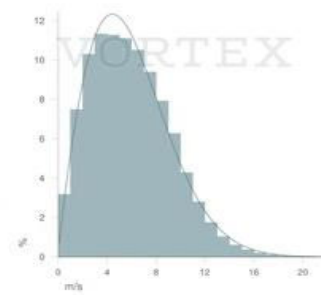


Fig. 9. Probabilidad de Ocurrencia. Imagen extraída de Vortex.

La imagen de la Rosa de Vientos muestra cuales son las direcciones de viento predominantes en una zona determinada, tomadas en un periodo de tiempo dado.

C. Contrastación y calibración

Los valores obtenidos desde el acelerómetro reflejan una de las propiedades físicas en las que se basa el principio de funcionamiento de este dispositivo: a medida que el péndulo se eleva, alejándose de la posición de reposo, la fuerza tangencial del peso de la esfera contenedora aumenta debido al aumento del ángulo, como se observa en la Fig. 10.

$$F_t = -mg \cdot \sin\theta \quad (7)$$

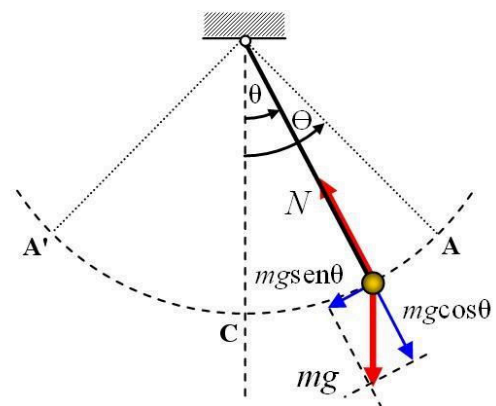


Fig. 10. Componentes del Peso. Imagen extraída de www.wikipedia.com

Se utilizó el anemómetro portátil LM-8000 como elemento patrón para contrastar el dispositivo [7]. En las tablas II y III, se observan las mediciones más representativas, tomadas en los ensayos experimentales junto con el cálculo de un factor de corrección.

TABLA II. CÁLCULO DE FACTOR DE CORRECCIÓN (VELOCIDADES > 20Km/h).

Sensor	Velocidad Patrón (Km/h)	Coef. de corrección
27,63	54,5	1,97
39,43	76,2	1,93
48,02	93,0	1,94
20,38	41,0	1,97
22,2	43,8	1,97

TABLA III. CÁLCULO DE FACTOR DE CORRECCIÓN

(VELOCIDADES < 20Km/h).

Sensor	Velocidad Patrón (km/h)	Coef. de corrección
6,93	20,0	2,88
6,05	17,4	2,88
4,64	13,6	2,93
5,79	18,0	3,1
6,36	19,0	2,98

Mediante el promedio de todas las muestras obtenidas, se obtiene el coeficiente de corrección final que varía según el rango de velocidad medida.

$\lambda = 2$, para superiores a 20 km/h.

$\lambda = 3$, para velocidades inferiores a 20 km/h.

Este coeficiente de corrección es utilizado en el microcontrolador para calcular la intensidad de viento, tanto instantánea como máxima, en km/h. El microcontrolador seleccionado es el PIC18F252 [8].

De las mediciones realizadas se puede obtener también el error producido. En el análisis se discrimina según el rango de velocidades y se calcula el error en cada caso. Esto se detalla en las Tablas IV y V.

TABLA IV. CÁLCULO DEL ERROR DE MEDICIÓN.
(VELOCIDADES > 20Km/h).

Veloc. Calibrada ($\lambda=2$)	Veloc. Patrón (km/h)	Error (%)
55	54,5	1,37
78,86	76,2	3,37
96,04	93	3,16
40,76	41	0,59
44,4	43,8	1,35

TABLA V. CÁLCULO DEL ERROR DE MEDICIÓN.
(VELOCIDADES < 20Km/h).

Veloc. Calibrada ($\lambda=3$)	Veloc. Patrón (km/h)	Error (%)
20,79	20,0	3,79
18,15	17,4	4,3
13,92	13,6	2,29
17,37	18,0	3,62
19,08	19,0	0,42

Con estos resultados, concluimos que la medición se ve afectada por la introducción de un factor de corrección fijo, el cual al ser resultado del promedio y redondeo, incrementa el error en la medición. Puede concluirse las especificaciones en la medición de intensidad de la siguiente manera:

$$Veloc_{>20km/h} = VelocCalibrada \pm 3,5\%$$

$$Veloc_{<20km/h} = VelocCalibrada \pm 4,5\%$$

D. Tratamiento de la Información

Los resultados obtenidos desde el microcontrolador se envían inalámbricamente a través un transmisor RF de 433MHz y son recibidos por una PC-servidor encargada de leerlos, respetando el siguiente orden [9]:

Orientación: Promedio: Máximo: Checksum

Se desarrolló un programa en lenguaje C, cuya función es adquirir los datos a través del puerto serie de la PC y almacenarlos en formato CSV [10],[11]. La información contenida en estos archivos es mostrada de forma adecuada en una página web, pero a su vez puede ser utilizada por otros programas para la creación de tablas, gráficos, rosas de vientos, etc.

Se crean dos archivos CSV principales:

- “Datosviento.csv”: guardado cada 10 minutos, contiene los datos de velocidades promedio, velocidad máxima medida, orientación instantánea, fecha y hora, de dicho lapso de tiempo.

- “Diaanterior.csv”: contiene los datos obtenidos el día anterior, almacenados cada 8 horas (a las 8hs, 16hs y 0hs), guardando la orientación, velocidades máximas y promedio que se produjeron.

E. Página Web

La programación de la página web se realiza en lenguaje PHP [12]. El objetivo de utilizar dicha página es que el usuario pueda tener acceso a los datos del sensor de viento a través de internet en todo momento. La información que se desea mostrar es la siguiente:

1) Intensidad promedio, orientación y velocidad máxima de los últimos 10 minutos, Fig. 11.

Intensidad Promedio	61.8 km/h
Orientacion	N-O
Velocidad Maxima	117.2 km/h

Fig. 11. Visualización de Datos Actuales en Página Web.

- Intensidad Promedio: se toman los datos y se realiza un promedio de todos los valores obtenidos durante ese período de tiempo.

- Orientación del viento: debido a la baja probabilidad de cambio de la dirección en lapsos de tiempo reducidos, se toma la última muestra.

- Velocidad Máxima: durante este lapso de tiempo se comparan los valores máximos y se almacena el máximo absoluto.

Se incluye un gráfico de barra dinámico, con el objetivo de ilustrar la intensidad de viento promedio actual. El rango de la barra es idéntico al rango de medición del sensor, es decir, desde 0 km/h hasta 90 km/h. En la misma se observan

diferentes colores según la intensidad:

- Verde: Indica que el viento es moderado, no hay alerta meteorológica. Corresponde a viento promedio entre 0 km/h y 60 km/h, Fig. 12.

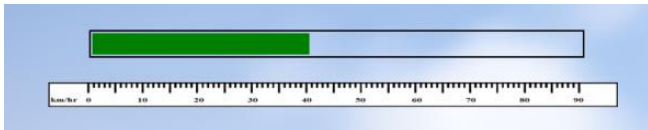


Fig. 12. Visualización de Intensidad de viento moderado.

- Amarillo: Corresponde a intensidades promedio superiores a 60 km/h e inferiores a 80 km/h, Fig. 13.

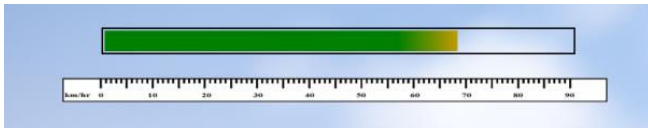


Fig. 13. Visualización de Intensidad de viento mayor.

- Rojo: Indica alerta meteorológica, se refiere a intensidades desde 80 km/h en adelante, se muestra un mensaje de alerta por los fuertes vientos, Fig. 14 [13].



Fig. 14. Visualización de Intensidad de viento fuerte.

2) Intensidad promedio, orientación y velocidad máxima ocurridos el día anterior en lapsos de 8 horas, como se observa en la Fig. 15.

	Manana	Tarde	Noche
Velocidad Maxima	57.3 km/h	96.0 km/h	117.2 km/h
Velocidad Promedio	36.83 km/h	85.71 km/h	66.51 km/h
Orientacion	S-E	S-O	S-E

Fig. 15. Visualización de datos del día anterior

F. Eficiencia Energética.

Con el fin de implementar un dispositivo que sea autónomo energéticamente, se incorpora una batería de gel de 6V y 4AH, en conjunto con un regulador Buck Step Down LM2596 ilustrado en la Fig. 16 [14].



Fig.16. Regulador Buck Step Down LM2596.

Se considera un consumo estimado del sistema de alrededor de 2mA y utilizando la ley de Peukert, se calcula la duración de la batería en estas condiciones [15]. Esto resulta:

$$t = H / [(I \cdot H) / C]^k \quad (8)$$

$$t = 20H / [(0,002A \cdot 20H) / 4AH]^{1.1} \quad (9)$$

$$t = 3169H \quad (10)$$

Donde:

t: es el tiempo en horas de autonomía.

C: capacidad de la batería en AH.

H: tiempo promedio de descarga en horas, especificado por el fabricante.

I: intensidad de corriente de consumo.

k: Constante de Peukert. (k=1.1 para baterías de gel).

Como puede observarse, los resultados obtenidos se encuentran alrededor de 132 días. Además, considerando que el dispositivo funciona a la intemperie, se coloca un panel solar capaz de cargar la batería y contribuir con la autonomía del dispositivo.

G. Análisis de Mercado.

Al analizar las opciones disponibles en el mercado, se calcula previamente el costo total estimado del prototipo propuesto, el cual se encuentra detallado en la Tabla VI:

TABLA VI. CÁLCULO DE COSTO DEL PROTOTIPO.

Cantidad	Producto	Precio por unidad [\$]	Precio total [\$]	Especificación
1	PIC18F252	\$184	\$184	DIP
1	Step Down	\$39	\$39	LM2596
1	Acelerómetro	\$94	\$94	MMA7361
1	Panel Solar	\$126	\$126	6V
1	Batería 6V	\$100	\$100	6V Recargable
1	Placa Pertinax	\$8	\$8	60mm x 70mm
1	Esfera plástica	\$15	\$15	φ100 mm
1	Varilla plástica	\$20	\$20	20mm x 800mm
1	Tx y Rx	\$40	\$40	433 MHz
Total			\$626	

Una vez obtenido el costo del prototipo propuesto, se procede a la comparación con instrumentos comerciales. Teniendo en cuenta que el dispositivo diseñado brinda información sobre la intensidad y la orientación del viento simultáneamente, demuestra tener ventaja respecto de la necesidad actual de dos dispositivos para la obtención de dichos parámetros, como lo son: un anemómetro y una veleta. Al consultar en el mercado, se encuentra que una veleta convencional cuesta alrededor de \$1000; mientras que pueden encontrarse anemómetros que van desde los \$2.500 a \$3000, para uso semiprofesional.

Cabe destacar que el costo de fabricación puede reducirse aún más utilizando componentes electrónicos más económicos. Los componentes utilizados en la realización de este prototipo pueden ser reemplazados por nuevas tecnologías con similares características, ya que no alterarían el principio de funcionamiento del mismo.

III. CONCLUSIÓN

La idea de este trabajo es brindar una alternativa novedosa y económica como instrumento de medición de un fenómeno meteorológico tan cotidiano como el viento y del que puede obtenerse un importante beneficio particularmente en el campo de la energía eólica. Es versátil debido a que brinda la posibilidad de modificar no sólo las características físicas constructivas, por ejemplo, reduciendo el tamaño de la esfera o el peso de la misma, sino también puede variarse el tiempo de actualización de los datos o la cantidad de muestras tomadas para el cálculo de la velocidad.

El dispositivo es muy robusto ante fuertes vientos, típicos en la zona de la Patagonia. En los ensayos experimentales, se sometió al dispositivo a intensidades de viento superiores a 90 Km/h. Además, presenta un error de medición comparable con anemómetros disponibles en el mercado. Dicho error que es menor al 5%, puede reducirse aún más mediante el ajuste del factor de corrección, ya que el utilizado fue redondeado a un número entero, sin el empleo de decimales.

Demuestra tener una gran ventaja frente a instrumentos de medición convencionales, como lo son los anemómetros ya que con un solo dispositivo se adquiere información de la intensidad y orientación del viento, lo que reduce significativamente los costos.

Este proyecto intenta representar la continua mejora y avance de tecnologías embebidas que permiten la obtención de resultados de manera más eficaz tendiendo a ser cada día más compactas y ofreciendo mayores herramientas para la toma de decisiones.

REFERENCIAS

- [1] Datasheet Acelerómetro MMA7361L. <http://www.nxp.com/assets/documents/data/en/datasheets/MMA7361L.pdf>
- [2] Vortex. Servicio online de modelado de viento. Consultado en diciembre del 2015. <http://www.vortexfdc.com>.
- [3] Instituto Nacional de Tecnología Industrial (INTI). <https://www.inti.gov.ar>.
- [4] Rosa de vientos. Brinda información gráfica de la orientación de viento que se da con mayor frecuencia en una zona determinada.

- https://es.wikipedia.org/wiki/Rosa_de_los_vientos
- [5] Probabilidad de ocurrencia. Gráfica en la cual se puede observar la frecuencia en la que ocurren diferentes intensidades de viento en una zona determinada.
- [6] Universidad Tecnológica Nacional – Facultad Regional del Neuquén (UTN - FRN). <http://www.frn.utn.edu.ar/>
- [7] http://www.lutron.com.tw/ugC_ShowroomItem.asp?hidKindID=2&hidTypeID=107
- [8] Datasheet Microcontrolador PIC18F252. <http://ww1.microchip.com/downloads/en/DeviceDoc/39564c.pdf>
- [9] Datasheet transmisor RF 433MHz. http://www.wenshing.com.tw/Data_Sheet/TRW-V8-433-P_433MHz_Wireless_RF_500mW_Transceiver_Module_Data_Sheet_E.pdf
- [10] Tutorial C++. Utilizado para leer datos recibidos a través del puerto serie y convertirlos en formato CSV. <http://www.cprogramming.com/tutorial/c++-tutorial.html>
- [11] Comma Separated Values (CSV). Formato utilizado para estandarizar los datos obtenidos y simplificar su lectura e interpretación. https://es.wikipedia.org/wiki/Valores_separados_por_comas.
- [12] Hypertext Preprocessor (PHP). Lenguaje de código abierto utilizado en entornos de desarrollo Web. Tutoriales consultados: www.php.net y <https://desarrolloweb.com/php/>.
- [13] Defensa Civil de la provincia emite alerta meteorológica y recomienda la suspensión de actividades cuando las intensidades de viento promedio alcanzan los 80 Km/h.
- [14] Datasheet Step Down LM2596. <http://www.ti.com/lit/ds/symlink/lm2596.pdf>
- [15] Ley de Peukert: presentada en 1897 por el científico alemán Wilhelm Peukert. Expresa el cambio en la capacidad de baterías recargables a diferentes velocidades de descarga.

Nuevo sistema embebido inalámbrico para medición de efectos de radiaciones ionizantes

Aplicación a un caso de estudio y perspectivas futuras

Pablo Alejandro Ferreyra y Daniel Sánchez
Laboratorio de Circuitos y Sistemas Robustos, (LCSR)
FCEfyN, UNC y Posgrado de Sistemas Embebidos, IUA.
Córdoba, Argentina

Mario Debray y Nahuel Vega
GAIANN Comisión Nacional de Energía Atómica
San Martín, Buenos Aires, Argentina

Felix Palumbo
Consejo Nacional de Investigaciones Científicas y Técnicas,
GAIANN Comisión Nacional de Energía Atómica y
Departamento de Ingeniería Electrónica, Facultad Regional
Buenos Aires, UTN

Alberto Fabián Gómez
Universidad Nacional de Chilecito
Chilecito,
La Rioja, Argentina

Resumen—El presente trabajo presenta un nuevo sistema que permite medir efectos de radiaciones ionizantes sobre diferentes dispositivos electrónicos. La seguridad, confiabilidad y facilidad de los procesos de medición se proveen por medio de una arquitectura basada en el monitoreo y control inalámbrico configurable por el usuario. Todos los dispositivos utilizados para el desarrollo del sistema son de bajo costo y fácil accesibilidad, tales como celulares o tabletas, microcontroladores y dispositivos COTS, (“Comercial-off-the-shelf”). Se muestra su aplicación al estudio del conteo de los eventos singulares en una memoria RAM estática. Se indican futuras aplicaciones y resultados esperados asociados a proyectos de relevancia actualmente en marcha.

Palabras claves— radiaciones; ionizantes; plataformas; test; mediciones; inalámbrico;

I. INTRODUCCION

La caracterización de los efectos de las radiaciones sobre componentes electrónicos, continúa dando origen a nuevas líneas de investigación y desarrollo, [1-4]. Una de estas líneas consiste en el desarrollo de plataformas de medición más flexibles, versátiles y accesibles. Hasta ahora las plataformas utilizadas se basan en conexiones alámbricas entre las distintas partes del sistema, [5-8]. El presente trabajo muestra la factibilidad de utilizar comunicaciones inalámbricas entre puntos estratégicos del sistema, lo cual trae aparejado numerosos beneficios. En particular, los requerimientos de seguridad de los procesos de medición se facilitan naturalmente, permitiendo al operador alejarse a una distancia prudencial, cómoda y conveniente. Además, el sistema ha sido concebido para permitir el control y monitoreo inalámbrico de la experiencia utilizando dispositivos de accesibilidad masiva tales como ce-

Trabajo realizado con el apoyo de la Secretaría de Ciencia y Tecnología de la Universidad Nacional de Córdoba.

lulares o tabletas, y tecnología de calidad COTS, (Commercial off-the-shelf), en todas sus partes y componentes.

En efecto, en la sección II, se da una descripción general del sistema donde puede observarse que sus tres partes principales han sido implementadas con tecnologías de bajo costo y fácil accesibilidad comercial, tanto en su hardware como en su software.

En la sección III, se describe el hardware del componente principal del sistema, denominado placa de interfaz. El hardware del dispositivo de control móvil o inalámbrico se describe en la sección IV, en tanto que el software de ambas partes y su interacción se describe en la V. La sección VI muestra el proceso general de ensayo aplicada a una SRAM (Static RAM) y finalmente en la sección VII se enumeran algunas conclusiones y trabajos futuros asociados al sistema presentado.

II. DESCRIPCIÓN GENERAL DEL SISTEMA

Para realizar los ensayos, se construyó una plataforma de test conformada por tres bloques principales:

Una placa de interfaz, una placa remota, y un dispositivo móvil, como se muestra en la "Fig. 1".

La placa de interfaz está basada en un microcontrolador de 8 bits. La placa remota tiene la finalidad de alojar al componente bajo prueba, dentro de la cámara de vacío por medio de conectores alámbricos apropiados. El dispositivo bajo prueba usado en este trabajo es una memoria SRAM (Static Random Access Memory), pero el sistema podría adaptarse fácilmente a otros tipos de memorias o dispositivos tales como FPGAs, microcontroladores, etc.

Para el dispositivo móvil, se desarrolló una aplicación que puede funcionar en cualquier tableta o celular. Dicha aplicación corre sobre sistema operativo Android, [9], y controla y monitorea a la placa de interfaz, por medio de un enlace inalámbrico. El hardware y el software desarrollado para la placa de interfaz, a su vez, posibilitan realizar la escritura, lectura y monitoreo del dispositivo bajo radiación. Además,

provee y mide la tensión de alimentación del mismo en forma continua. Es capaz de detectar y actuar sobre condiciones de cortocircuitos o corrientes excesivas. Durante las irradiaciones de la placa remota se encuentra dentro de la cámara de irradiación del micro haz de iones pesados del acelerador TANDAR en condiciones de alto vacío [10] y montada sobre un posicionador XYZ de 2 μm de paso. La placa interfaz se conecta con la placa remota por medio de un conector de alto vacío. Empleando un microscopio óptico digital se puede ubicar la zona de la muestra que se desea irradiar y alinearla con la posición del micro-haz de iones. El control de la placa de interfaz, la visualización de los resultados en tiempo real y el posterior almacenamiento de los datos se hace a través del dispositivo móvil. Las comunicaciones entre la placa de interfaz y el dispositivo móvil se realizan a través de una conexión inalámbrica según protocolo "Bluetooth", [11].

Como se muestra en la "Fig. 1", la Placa de Interfaz, está organizada en torno a un microcontrolador de 8 bits. Puede controlar el estado eléctrico (voltajes y consumos) y el lógico (intercambio de información), del dispositivo bajo prueba. El dispositivo bajo prueba puede ser cualquiera, (aunque el caso bajo estudio de esta presentación sea una SRAM). El microcontrolador opera con un reloj interno de 32 MHz. Como ya se mencionó, se comunica por medio del protocolo "bluetooth" hacia un dispositivo móvil (ej. teléfono celular o tableta). En el dispositivo móvil se ejecuta una aplicación de monitoreo y control desarrollada en App Inventor 2, [12]. Ésta aplicación comanda el tipo de test que se realiza y permite visualizar el avance del procedimiento de test. A su vez, permite cambiar fácilmente el software para realizar otro tipo de ensayo sobre otro tipo de dispositivos. Los resultados se almacenan en la memoria SD del dispositivo móvil, para luego, por medio de la tecnología de las "Google Fusion Tables", [13] se posibilite el compartir, consultar y visualizar las tablas de resultados del test vía web.

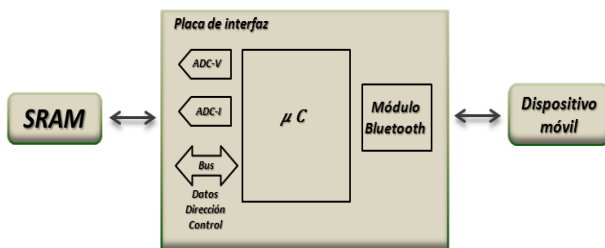


Fig. 1. Plataforma de Test

III. PLACA DE INTERFAZ

Está construida alrededor de un microcontrolador de bajo costo de 8 bits, (PIC16F1936). El PIC realiza las operaciones de escritura, lectura, control de alimentación y monitorización eléctrica (voltajes y consumos) de la SRAM, en este caso. La

placa de interfaz se comunica vía RS232 a un módulo bluetooth HC-06 [14], el cual transmite los datos del ensayo y recibe los comandos de control desde el dispositivo Android.

Además cuenta con un conector ICSP (In-Circuit Serial Programming) que permite programar el microcontrolador en el circuito, sin necesidad de desmontar el chip de la placa final, (ver "Fig. 2").

Como se muestra en la "Fig. 2", se utiliza el integrado MCP23S17 como expansión de puerto, de tal forma que ampliamos las salidas y entradas digitales en 16. Éste chip es controlado por el microcontrolador por medio de un bus SPI, (Serial Peripheral Interface), [15]. Con la expansión se obtiene el bus de direcciones para acceder a la memoria bajo prueba. El bus de datos es manejado por el puerto A del microcontrolador.

La medición de corriente se realiza como se muestra en la "Fig. 3". Es decir, se conecta la resistencia de medición entre la fuente de alimentación (5V IN) y la carga (5V OUT). La diferencia de voltaje en la resistencia de medición (RSH3//RSH4) es amplificada por la ganancia del circuito realimentado negativamente para obtener la tensión de medición de corriente (I_{SRAM} , en este caso).

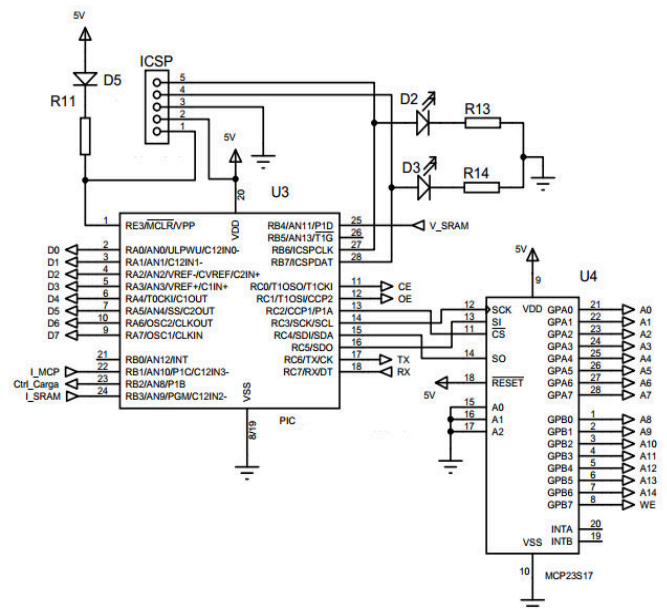


Fig. 2. Hardware de la Placa de interfaz

De ésta forma además de la medición de la corriente de alimentación de la memoria, es posible detectar las posibles elevadas corrientes debido a cortocircuitos en la placa bajo prueba.

Para ésta medición, se utiliza un amplificador diferencial y cuatro resistencias externas. Con este circuito se amplifica la pequeña caída de tensión a través de la resistencia de medición ($5\text{V IN} - 5\text{V OUT}$) y se multiplica éste valor por la ganancia

(R_9/R_7). Con este esquema se obtiene también un alto rechazo a la tensión de entrada en modo común.

$$V_o = (V_1 - V_2) \cdot \left(\frac{R_9}{R_7}\right) \quad R_6 = R_7 ; R_8 = R_9$$

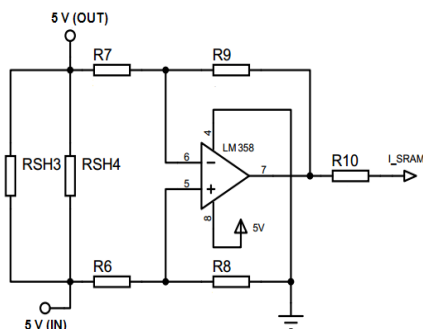


Fig. 3. Medición de corriente

El consumo de corriente típico de la memoria bajo prueba en operación es de 60 mA, y la placa permite configurar este valor como umbral. Cuando la medición de corriente sea mucho mayor a éste valor umbral programable, se deberá a una falla en el circuito, por lo que el microcontrolador corta la alimentación de la placa externa por medio del circuito de control de corriente, hasta que el cortocircuito desaparezca. Es importante resaltar las provisiones para hacer que la corriente de cortocircuito sea ajustable al dispositivo bajo prueba.

IV. DISPOSITIVO MÓVIL

Gracias a la conexión “bluetooth” entre el dispositivo móvil y la placa interfaz, es posible controlar inalámbricamente todo el desarrollo del ensayo radiaciones.

Para la implementación del control inalámbrico se utiliza un módulo bluetooth HC-06. El HC-06, se comunica con el microprocesador vía el protocolo RS232 y con la aplicación en el dispositivo Android por Bluetooth.

EL modulo Bluetooth HC-06 de la "Fig. 4", funciona sólo en modo esclavo. En fábrica es configurado de la siguiente manera:

- Nombre: “linvor”
- Contraseña: 1234
- Velocidad: 9600 baudios

Este dispositivo se inicia como desconectado, al ser alimentado ingresa a éste modo y se evidencia con el LED1 intermitente. Una vez que se estableció la conexión con otro dispositivo, pasa a modo conectado, con lo que el LED1 deja de parpadear y queda encendido.

La aplicación en “Android” (Control Test) se utiliza para seleccionar el tipo de test que se desea realizar. Se pueden cargar fácilmente distintos tipos de test en función del

dispositivo bajo estudio. Para el ejemplo de este trabajo, se pueden seleccionar cuatro diferentes tipos de test: escritura y lectura de ceros, de unos e incremental, además de “March test”. Estos ensayos se desarrollan escribiendo en las celdas de memoria un valor conocido y luego se realiza la lectura de la memoria para observar si alguna celda contiene un valor erróneo. Una vez conectado, los datos ingresados en RXD se transmiten vía bluetooth hacia el dispositivo conectado, y los datos recibidos se devuelven por el pin TXD.

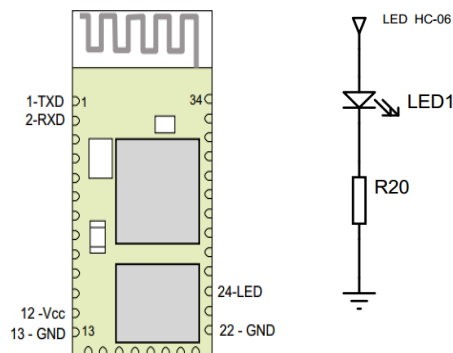


Fig. 4. Módulo bluetooth y led de estado de conexión.

El módulo “bluetooth” se alimenta con un regulador de tensión de 3.3 V, por lo que se debe realizar una adaptación de tensiones para poder conectarlo con el microcontrolador, "Fig. 5".

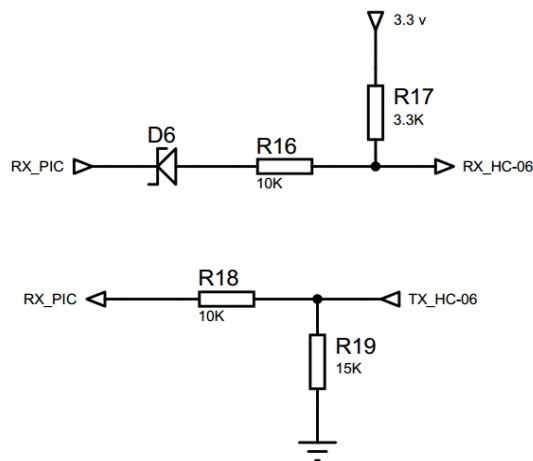


Fig. 5. Adaptación de tensiones.

V. SOFTWARE

La aplicación desarrollada en App Inventor 2, tiene cuatro pantallas (ver “Fig. 6”) para seleccionar la tarea a realizar, conexión bluetooth, Test, manejo de archivos y control de barrido en los ejes X e Y del micro-haz de irradiación (en desarrollo). Se está desarrollando una actualización para

permitir elegir diferentes dispositivos bajo prueba de forma totalmente automática y configurable por software.



00:12:10:16:04:56 linvor

Fig. 6. Pantalla principal de tareas.



Fig. 7. Pantalla de Test en Progreso.

Una vez establecida la conexión bluetooth con el dispositivo Android, se selecciona la pantalla de Test, para llevar a cabo el ensayo. Al ingresar, (ver “Fig. 7”), la aplicación muestra los valores de tensión y corriente medidos, de color verde en este caso. Si los valores estuvieran fuera de los máximos o mínimos permitidos, (60 miliamperes) o si se detectara algún error de otro tipo, (voltajes mayores a 5,5 Volts) cambiarían a rojo. Aquí se debe seleccionar el tipo de test a ensayar, por ejemplo, escritura de unos, de ceros, incremental o “March test”, para el caso de una SRAM. Luego se selecciona “iniciar test” y la aplicación envía el comando vía bluetooth a la placa de interfaz, para que finalmente el microcontrolador de inicio al test correspondiente.

Al finalizar, la aplicación muestra en color azul que el test ha finalizado, (ver “Fig. 8”). En ese momento, los resultados son enviados por la placa de interfaz hacia la aplicación en Android que los almacena en la memoria SD del dispositivo. Se permite indicar el nombre del archivo anexando la fecha y hora del instante en que se almacena.

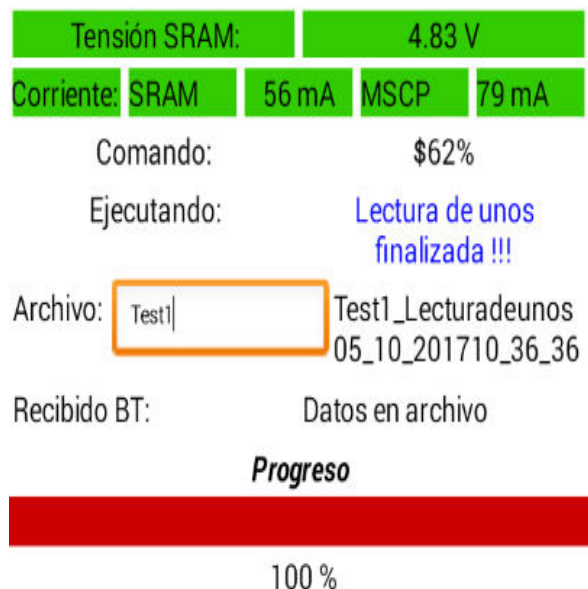


Fig. 8. Pantalla de Test Finalizado.

Ingresando en la pantalla de Archivos, por medio de un explorador de archivos, pueden verse los resultados, (Ver “Fig. 9”).

La Figura 9 muestra los resultados de un Test, donde las direcciones de memoria se dividen en 32 bloques, a la derecha de cada bloque se lee la cantidad de errores encontrados.

Al final de la columna, se ve la cantidad de errores totales.

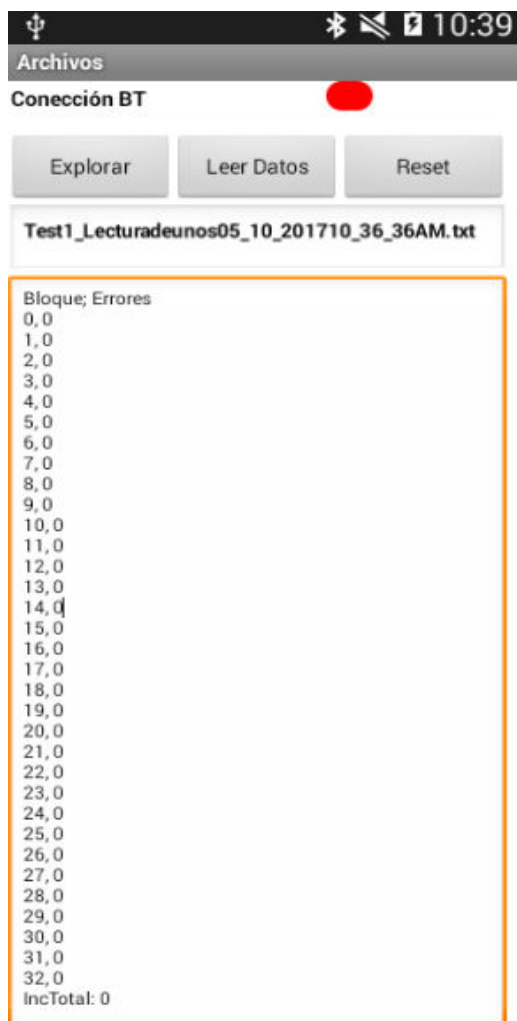


Fig. 9. Pantalla del archivo conteniendo los resultados.

VI. PROCESO DE MEDICIÓN

Para realizar el ensayo se diseñaron dos placas de circuitos impresos, (ver “Fig. 10”), una placa de interfaz y la placa de prueba, donde se inserta el dispositivo a ensayar. Al establecerse la conexión por bluetooth, el microprocesador envía los datos de tensión y corriente actuales.

Ésta última, es una tarea que se realiza en forma constante para verificar que se está dentro de los límites preestablecidos. Si se detecta un valor fuera de éstos límites, se deja de alimentar la placa de prueba y se envía los valores medidos al dispositivo de control remoto. Ésto se evidencia al ponerse en rojo las casillas con los valores leídos. Si los valores de tensión y corriente son aceptables, el microprocesador, al recibir el comando de inicio de test, comienza con el mismo. En el caso de tratarse de una memoria, comienzan con la

escritura de un patrón conocido en la misma. Al finalizar envía un comando de fin de proceso al dispositivo Android. En este punto se inicia el proceso de captura de resultados. En el ejemplo presentado, se inicia el proceso de lectura del patrón antes escrito en la SRAM.

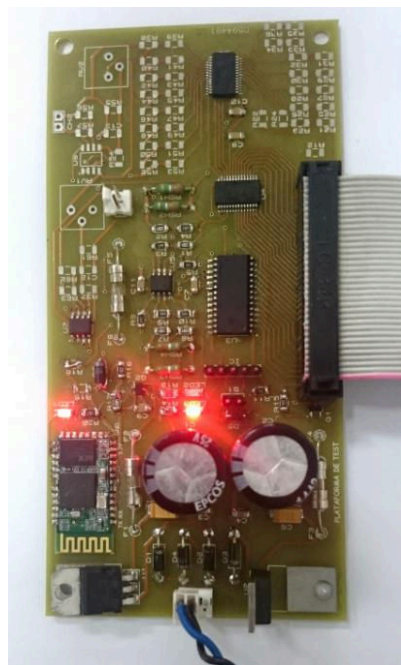


Fig. 10. Placa de interfaz y placa de prueba.

Finalmente se inicia el postprocesamiento y almacenamiento: En este caso se trata de leer toda la memoria y corroborar los datos por bloques, éstos son 32 bloques de 1024 bits cada uno. Los datos son enviados al dispositivo para ser almacenados y posteriormente almacenados.

VII. CONCLUSIONES Y TRABAJO FUTURO

El presente trabajo es una demostración tecnológica de la factibilidad de realizar mediciones de efectos de radiación ionizante con equipamiento liviano, muy versátil y de alta accesibilidad. Una variante del mismo equipamiento se puede utilizar para decapar los circuitos integrados como se describe en [16]. De esta manera todos los procesos y dispositivos tecnológicos para realizar mediciones de efectos de radiaciones ionizantes sobre componentes electrónicos en instalaciones tales como el TANDAR están cubiertos. Se han realizado gestiones para realizar mediciones bajo radiación utilizando el MiP del acelerador TANDAR. Se han planificado las mismas para el mes de agosto de 2017. Independientemente de estos resultados, se realizarán pruebas donde se usarán otras plataformas para emular fallas en los dispositivos bajo prueba, tales como la descrita en [17], a la cual se tiene acceso por haber participado en su desarrollo. En efecto, éste trabajo puede verse también como un complemento indispensable para verificar el funcionamiento de inyectores de fallas.

En relación a los trabajos futuros, se piensa utilizar esta plataforma para caracterizar los componentes de proyectos de relevancia nacional e internacional tal como el descrito en [18]. Además se piensa hacer evolucionar la plataforma para darle mayor velocidad y capacidad de procesamiento para permitir no sólo la medición experimental sino también la emulación de fallas usando una extensión de la misma. Con estos objetivos cumplidos, se dispondrá en un solo equipo todas las funcionalidades requeridas para calificación de sistemas críticos para uso en ambientes con radiación ionizante, tales como los descriptos en [19-25].

REFERENCES

- [1] Raoul Velazco, Pascal Fouillat, and Ricardo Reis. Radiation Effects on Embedded Systems. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
- [2] E. Ibe, H. Taniguchi, Y. Yahagi, K. i. Shimbo, and T. Toba. Impact of scaling on neutron-induced soft error in srams from a 250 nm to a 22 nm design rule. *IEEE Transactions on Electron Devices*, 57(7):1527–1538, July 2010.
- [3] A. Dixit and A. Wood. The impact of new technology on soft error rates. In *Reliability Physics Symposium (IRPS)*, 2011 IEEE International, pages 5B.4.1–5B.4.7, April 2011.
- [4] F. Serrano, J. A. Clemente, and H. Mecha. A methodology to emulate single event upsets in flip-flops using fpgas through partial reconfiguration and instrumentation. *IEEE Transactions on Nuclear Science*, 62(4):1617–1624, 2015.
- [5] M. Alderighi, F. Casini, S. D’Angelo, M. Mancini, D. M. Codinachs, S. Pastore, C. Poivey, G. R. Sechi, G. Sorrenti, and R. Weigand. Experimental validation of fault injection analyses by the flipper tool. *IEEE Transactions on Nuclear Science*, 57(4):2129–2134, Aug 2010.
- [6] F Faure, P Peronnard, R Velazco, and R Ecoffet. Thesic+: A flexible system for see testing. In *Proc. of RADECS*, 2002.
- [4] H. Quinn, K. Morgan, P. Graham, J. Krone, M. Caffrey, and K. Lundgreen. Domain crossing errors: Limitations on single device triplemodular redundancy circuits in xilinx fpgas. *IEEE Transactions on Nuclear Science*, 54(6):2037–2043, Dec 2007.
- [5] R. Velazco, G. Foucard, and P. Peronnard. Combining results of accelerated radiation tests and fault injections to predict the error rate of an application implemented in sram-based fpgas. *IEEE Transactions on Nuclear Science*, 57(6):3500–3505, Dec 2010.
- [6] R Velazco, S Rezgui, and E Reguer. Thesic: Una plataforma flexible para la validation funcional de circuitos integrados. In *VII Workshop IBERCHIP*, volume 21, 2001.
- [7] V. Pouget, A. Douin, G. Foucard, P. Peronnard, D. Lewis, P. Fouillat, and R. Velazco. Dynamic testing of an sram-based fpga by time-resolved laser fault injection. In *2008 14th IEEE International On-Line Testing Symposium*, pages 295–301, July 2008.
- [8] S. Rezgui, R. Velazco, R. Ecoffet, S. Rodriguez, and J. R. Mingo. Estimating error rates in processor-based architectures. *IEEE Transactions on Nuclear Science*, 48(5):1680–1687, Oct 2001.
- [9] Android. <https://www.android.com/>
- [10] S. Sondon et al., "Diagnose of radiation induced single event effects in a PLL using a heavy ion microbeam," 2013 14th Latin American Test Workshop LATW, Cordoba, 2013, pp. 1-5.
doi: 10.1109/LATW2013.6562682
- [11] Bluetootch. <http://ieeexplore.ieee.org/document/1016473/>
- [12] App Inventor. <http://appinventor.mit.edu/explore/>
- [13] Google Fusion tables. <https://sites.google.com/site/fusiontablestalks/home>
- [14] HC-06. <http://www.prometec.net/bt-hc06/>
- [15]SPI <http://ww1.microchip.com/downloads/en/DeviceDoc/spi.pdf>
- [16] Pablo A. Ferreyra, Roberto O. Lucci, José Arrieta and Daniel Sánchez, Desarrollo de un método de desencapsulado de circuitos integrados “COTS” para ensayos de radiación ionizante. In *16 SAM CONAMET Simposio Argentino de Materiales Congreso Internacional de Metalurgia y Materiales*, 22-25 November Cordoba, Argentina, 2016.
- [17] Miguel A. Solinas, Juan A. Fraire, Alexandre Cohelo, Pablo A. Ferreyra and Raoul Velazco. In *VII Congreso de Microelectrónica Aplicada uEA 2016*, 26-28 October San Luis Argentina 2016.
- [18] Carlos Barrientos, Anabela Ferral, Leandro Cara, Juan Fraire, Raoul Velazco, Pablo Madoery and Pablo A. Ferreyra, A segmented architecture approach to provide a continuous long-term adaptative and cost-effective glaciers monitoring system based on DTN communications and CubeSats platforms. En *primer Simposio Latinoamericano sobre pequeños satélites*, 1-10 March Buenos Aires Argentina 2017.
- [19] W. Mansour and R. Velazco. An automated seu fault-injection method and tool for hdl-based designs. *IEEE Transactions on Nuclear Science*, 60(4):2728–2733, Aug 2013.
- [20] Mojtaba Ebrahimi, Abbas Mohammadi, Alireza Ejlali, and Seyed Ghassem Miremadi. A fast, flexible, and easy-to-develop fpga-based fault injection technique. *Microelectronics Reliability*, 54(5):1000 – 1008, 2014.
- [21] W. Mansour, R. Velazco, R. Ayoubi, H. Ziade, and W. El Falou. A method and an automated tool to perform set fault-injection on hdl-based designs. In *2013 25th International Conference on Microelectronics (ICM)*, pages 1–4, Dec 2013.
- [22] Vaibhav Kale. Using the MicroBlaze Processor to Accelerate Cost Sensitive Embedded System Development. Xilinx, Inc., June 2016.
- [23] M. Shokrolah-Shirazi and S. G. Miremadi. Fpga-based fault injection into synthesizable verilog hdl models. In *Secure System Integration and Reliability Improvement, 2008. SSIRI '08. Second International Conference on*, pages 143–149, July 2008.
- [24] B. Rahbaran, A. Steininger, and T. Handl. Built-in fault injection in hardware - the fidyco example. In *Electronic Design, Test and Applications, Proceedings. DELTA 2004. Second IEEE International Workshop on*, pages 327–332, Jan 2004.
- [25] V. Alaminos, F. Serrano, J. Clemente, and H. Mecha. Nassy: An implementation of a low-cost fault-injection platform on a virtex-5 fpga. In *Conference on Radiation and its Effects on Components and Systems (RADECS)*, Sept 2012.

Osciloscopio - Analizador Lógico para Dispositivos Móviles

Ramiro Guerrero*, Rodrigo Abbas†, Esteban Volentini‡ y Daniel Cohen§

Facultad de Ciencias Exactas y Tecnología

Universidad Nacional de Tucumán

Tucumán, Argentina

*ramiro.g92@gmail.com, †rodrigoabbas@gmail.com, ‡evolentini@gmail.com, §dcohen@herrera.unt.edu.ar

Abstract—Los Dispositivos Móviles se han vuelto una parte de nuestra vida. El desarrollo de aplicaciones para móviles es algo común actualmente, pero sólo pocas intentan y logran efectivamente convertir al dispositivo móvil en un instrumento distinto, en una herramienta totalmente funcional y potencialmente capaz de reemplazar a sus equivalentes tradicionales, presentando incluso mayores ventajas.

En este artículo presentamos la implementación de un osciloscopio y analizador lógico, con un dispositivo móvil trabajando en conjunto con una Placa de Adquisición de Señales. Esto implicó el desarrollo de un firmware para sistemas embebidos y un software para dispositivos móviles que permiten adquirir y representar gráficamente señales eléctricas analógicas y digitales, variables en el tiempo.

El sistema planteado aprovecha el proceso de estandarización del Proyecto CIAA en instituciones académicas y la masificación del uso dispositivos móviles con sistema operativo Android.

I. INTRODUCCIÓN

Conforme avanza la tecnología se crean dispositivos móviles con mayor capacidad de almacenamiento y procesamiento que sus predecesores; con el fin de dar respuesta a los crecientes requerimientos de la comunicación y el entretenimiento. En este contexto surge la idea de aprovechar este recurso que casi toda persona lleva consigo y crear una herramienta que sea de utilidad para los estudiantes.

En la enseñanza de la Ingeniería Electrónica y disciplinas afines, queda rápidamente en evidencia la gran utilidad de los osciloscopios y analizadores lógicos para la implementación de laboratorios destinados a afianzar los conocimientos adquiridos. Lamentablemente es también evidente el alto costo de los mismos, y por lo tanto las fuertes limitaciones en la cantidad de equipos que pueden estar a disposición de los alumnos. En nuestra facultad, por ejemplo, los usuarios muchas veces deben compartir esta herramienta y depender de la disponibilidad del lugar donde se encuentra.

En este contexto, surge la idea de aprovechar este recurso de uso tan generalizado para trabajar en conjunto con la Plataforma CIAA (Computadora Industrial Abierta Argentina) que está penetrando progresivamente en los sectores educativo e industrial y con ellos crear una herramienta potente, accesible y que sea de gran utilidad para los usuarios. Como resultado utilizamos una CIAA como Placa de Adquisición y Transmisión de señales y un Dispositivo Móvil, para la recepción y representación gráfica de las mismas.

Dadas las limitaciones económicas en la adquisición y uso de herramientas como osciloscopios y analizadores lógicos para uso docente, existe en nuestro país un importante nicho para una herramienta portátil y de bajo costo, como la que se plantea en el presente artículo. Existen, incluso, otras necesidades que no se han podido subsanar debido a las mencionadas limitaciones, por ejemplo: uso extendido del osciloscopio en escuelas medias, mayor equipamiento en el monitoreo de señales biomédicas en hospitales públicos y a domicilio, primer diagnóstico “in situ” de fallas en equipos electrónicos de baja frecuencia, posibilidad de que los alumnos puedan usar el osciloscopio para trabajar en el lugar y el momento en que les quede más cómodo. Es claro, de todo lo anterior, que la herramienta a desarrollar no pretende competir con las prestaciones de un osciloscopio de alta gama, por el contrario, plantea ciertas limitaciones que no hacen a su objetivo de uso final.

A. Esquema General

El sistema propuesto está compuesto de dos partes: una Placa de Adquisición de Señales y un Dispositivo Móvil, que se comunican entre sí a través de sus puertos USB. Para la implementación del mismo se desarrolló una aplicación el sistema operativo Android y se utilizó una Placa EDU-CIAA-NXP, que posee un microcontrolador capaz de adquirir señales analógicas y digitales.

La placa EDU-CIAA-NXP permitirá la Adquisición de Señales mientras que el Dispositivo Móvil proveerá la interfaz por medio de la cual se podrá configurar la herramienta y/o visualizar los diferentes canales con sus respectivas mediciones.

II. PLACA DE ADQUISICIÓN

En la Fig.1 se muestra el Diseño Modular utilizado en el firmware de la Placa de Adquisición y a continuación se describirá cada módulo. Una vez establecida la conexión USB con el dispositivo móvil, el firmware de la placa está en condiciones de recibir dos clases de pedidos: los destinados a tomar las muestras y enviar los datos de las señales analógicas o digitales y los destinados a configurar los distintos módulos de manera de lograr la funcionalidad deseada.

A. Módulo de Comunicación

Esté módulo se implementó utilizando la interfaz USB Host 2.0 nativa que ofrece la placa de adquisición EDU-CIAA-NXP,

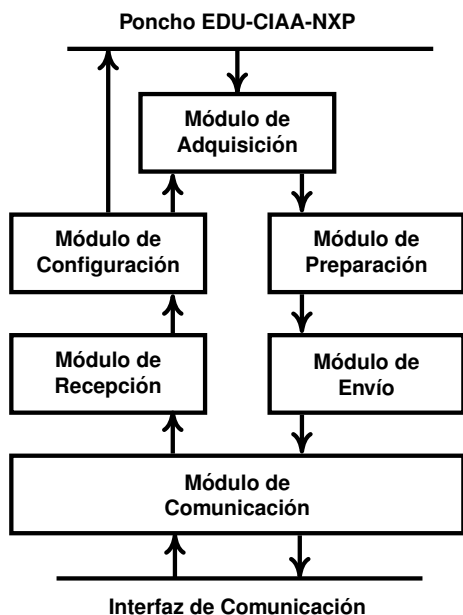


Fig. 1 Diseño modular - Firmware

aprovechando así la velocidad de transferencia provista (Full-Speed: 480Mbit/s teóricos), que permite aumentar la velocidad de adquisición y eliminar así un potencial "cuello de botella" en la transmisión de los datos.

El Módulo de Comunicación implementa el protocolo AOA (Android Open Accessory 2.0), que permitirá la conexión entre la placa y el dispositivo móvil con sistema operativo Android, y provee la funcionalidad de envío y recepción de datos.

Para lograr una conexión USB exitosa, se implementó una función encargada de abrir los canales de envío y recepción, para luego, siguiendo las directivas USB, establecer los canales denominados endpoints.

1) *AOA*: Una vez establecida la conexión USB, no es posible comunicarse directamente con un dispositivo Android, de manera que se implementa un protocolo especial, al que se denomina AOA que está diseñado para que Android acepte comunicarse con el hardware externo (en este caso, la Placa CIAA).

El protocolo está dividido en tres etapas principales:

- a- La Placa de Adquisición envía un pedido de control USB para verificar si el Dispositivo Móvil al que está conectada soporta el protocolo AOA.
- b- Se envía información acerca del dispositivo externo y para identificar la aplicación con la que debe interactuar.
- c- Por último se envía una solicitud para que el dispositivo Android ingrese al modo Accesorio.

Una vez finalizado este proceso, la Placa debe indicarle mediante un último pedido USB, cuáles son los canales de comunicación (envío y recepción).

B. Módulo de Recepción

Es el encargado de procesar los datos recibidos para poder configurar las diferentes funcionalidades.

Fue necesario crear un protocolo (RRCom) que permitiera diferenciar la información a fin de lograr la correcta comunicación entre la placa y el dispositivo móvil.

C. Módulo de Configuración

Configura la placa para la adquisición de la señal según los requisitos seleccionados por el usuario en la aplicación. Para lograrlo se establece comunicación con:

- a- El Módulo de Adquisición, para determinar las configuraciones necesarias para lograr una correcta adquisición de datos.
- b- El Módulo de Preparación, para preparar un paquete de datos que el dispositivo móvil pueda interpretar correctamente.
- c- El 'shield' para configurar la sensibilidad vertical y el acoplamiento, que son opciones para las cuales se necesita un hardware externo.

1) *Acceso Directo a Memoria - DMA*: El acceso directo a memoria (Direct Memory Access) es una característica encontrada en la mayoría de los microcontroladores. Este permite que se transfieran datos entre periféricos y memoria sin hacer uso del CPU permitiendo de esta manera realizar tareas en forma paralela.

Dada la necesidad de adquirir diferentes señales al mismo tiempo y atender las rutinas USB, se hizo uso de los canales (ocho) DMA presentes en el LPC 4337. De esta forma se logró el paralelismo necesario para la adquisición de dos o más señales manteniendo el procesador desocupado para atender pedidos USB o realizar otras tareas como el análisis del trigger de las señales muestreadas.

2) *Configurar Analógico*: Una vez recibida la orden del dispositivo móvil (previamente interpretada por el módulo de recepción) se ejecutan tres tareas principales:

- a- Se asignan los valores a las estructuras de datos de cada canal para poder realizar un tratamiento independiente y establecer la condición de trigger.
- b- Se habilitan los ADC según lo pedido y se asignan los canales necesarios para la transferencia de datos por DMA.
- c- Se habilita el timer SCT con la frecuencia de adquisición establecida por el usuario en la elección de la Sensibilidad Horizontal.

3) *SCT*: El State Configurable Timer es un temporizador especial de los microcontroladores de la familia LPC de la firma NXP. Permite generar interrupciones en otros periféricos para ejecutar ciertos eventos. Se lo usó para permitir configurar los valores de muestreo en los módulos ADC.

En la familia LPC43XX existen registros en los módulos ADCs que permiten configurar divisores para lograr menores tasas de muestreo. Se decidió utilizar SCT en vez de dichos divisores debido a que estos no permiten llegar a tasas menores a 100 KSPS. En cambio con esta tecnología de LPC se puede variar desde 400 KSPS a 1 SPS.

4) *Configuración Digital*: Una vez recibida la orden del dispositivo móvil, previamente interpretada por el módulo de recepción, se procede a la ejecución de estas tres tareas principales:

- a- Se configuran 8 pines de un puerto GPIO para la entrada de datos, y un pin GPIO para la entrada de clock externo.
- b- Se habilita el canal DMA para la transferencia y un timer para la adquisición de datos conectado a la entrada de clock externo.
- c- Se establece los valores necesario en la estructura de datos digital para la verificación de la condición de trigger.

D. Módulo de Adquisición

Este módulo adquiere datos analógicos utilizando los ADC0 y ADC1 cada vez que se realiza un pedido del timer SCT (según la sensibilidad horizontal establecida previamente). Además, adquiere 8 señales digitales utilizando el puerto GPIO conectado a un canal DMA para transferencia. Esto sólo se realiza cuando un evento de timer lo indica. Este evento es el que depende del clock externo, conectado a un pin de otro puerto GPIO.

E. Módulo de Preparación

Verifica las condiciones de Trigger y comienza a preparar el paquete de datos que se va a enviar cuando estas se cumplan.

Con el proceso de adquisición, es posible verificar la condición de Trigger (si se lo solicita). Para esto, el módulo accede a la misma dirección de memoria donde se están transfiriendo los datos e inicia la búsqueda del valor solicitado.

F. Módulo de Envío

Se encarga del envío de los datos solicitados. Según las directivas establecidas por el protocolo de comunicación propio (RRCom), es necesario indicar al inicio de cada paquete que se envía la procedencia de los datos, es decir, si el paquete en cuestión posee datos analógicos ó digitales y de qué canal provienen los datos cuando estos son digitales.

III. PROTOCOLO RRCom

A. RRCom

Luego de armada la conexión y establecida de comunicación entre el Dispositivo Móvil y la Placa de Adquisición, queda expuesta la necesidad de una comunicación a un nivel superior, entre las Aplicaciones en cada una, para lo cual se desarrolló el protocolo RRCom.

El protocolo se estructura en tramas diferentes, dependiendo el sentido de la comunicación.

B. Dispositivo Móvil a Placa de Adquisición

La comunicación desde el Dispositivo Móvil a la Placa consiste en el envío de una Trama RRCom de Pedidos. Cada trama es un paquete de bits que constituye la Unidad de Datos del Protocolo RRCom, y se representa en la Fig.2.

Cada Trama está compuesta por uno o más pedidos ya que el protocolo permite que los pedidos de distintos canales se

Cantidad de paquetes a continuación
Pedido 1
Pedido 2
...
Pedido N

Fig. 2 Trama RRCom de Pedidos

aniden en una misma trama, por lo que al inicio se agrega un byte que indica cuántos pedidos existen en ella.

Cada pedido es una estructura de 40 bits (5 bytes), como se puede observar en la Fig.3.

Pedir	Tipo	Tipo de clock	Acoplamiento	-	
Valor de trigger					
Valor de trigger		Tipo de trigger	Fuente trigger	Modo	-
Sensibilidad vertical			Canal pedido		
Sensibilidad horizontal			-		

Fig. 3 Pedido RRCom

- 1) El primer byte posee 4 elementos:
 - Pedir (1 bit): Indica a la Placa si se está realizando un pedido de señal.
 - Tipo (1 bit): Analógico o Digital.
 - Tipo de Clock (2 bits): En caso digital, indica con qué tipo de flanco se debe configurar la entrada de clock externo: Positivo, Negativo o Ambos
 - Tipo Acoplamiento (2 bits): Informa a la Placa qué tipo de acoplamiento se solicita, ya sea AC, DC o GND
- 2) El segundo byte contiene los primeros 8 bits de información sobre el valor del trigger para analógico.
- 3) El tercer byte posee 4 elementos:
 - Valor de Trigger (2 bits): Información que debe concatenarse con lo contenido en el segundo byte para tener el valor completo del nivel de trigger solicitado en caso Analógico.
 - Tipo Trigger (2 bits): Indica si la Placa debe configurar su trigger como flanco: Positivo o Negativo en caso Analógico y Positivo, Negativo o Ambos en caso Digital.
 - Fuente Trigger (2 bits): Comunica que fuente debe usarse en Analógico: Canal 1 (CH1), Canal 2 (CH2) o Alternado
 - Modo (2 bits): Es el modo de trigger que se debe configurar en caso analógico, ya sea como Automático (Auto) o Normal
- 4) El cuarto byte posee 2 elementos:
 - Sensibilidad Vertical (4 bits): índice del valor de sensibilidad vertical que debe usarse en caso analógico. Es denominado como índice dado que la Placa posee la misma tabla para seleccionar la sensibilidad, de

esta manera no enviamos un valor numérico con el consecuente ahorro de bits utilizados para tal fin.

- Canal Pedido (4 bits): Indica a qué canal tiene que responder la Placa el pedido.

5) El quinto byte posee 5 bits con información del índice de Sensibilidad Horizontal.

C. Placa de Adquisición a Dispositivo Móvil

La comunicación desde la Placa al Dispositivo Móvil consiste en el envío de una Trama RRCOM de Respuesta.

Cada trama es la respuesta de un pedido y tiene la estructura de la Fig.4

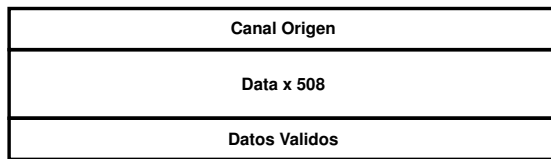


Fig. 4 Trama RRCOM de Respuesta

Donde:

- Canal Origen: Indica el canal al cual pertenece la información adquirida.
- Data: 508 Bytes con los datos adquiridos.
- Datos Válidos: Indica cuántos datos enviados son válidos. Este campo existe para ejecutar correctamente las funciones de lectura de buffer USB del lado del dispositivo móvil.

IV. APLICACIÓN MÓVIL

Se diseñó la aplicación de manera que, una vez establecida la conexión USB con la Placa de Adquisición, el usuario sea capaz de pedir el muestreo de las señales a la Placa de Adquisición y de visualizar gráficamente las señales obtenidas. En la Fig.5 se puede observar el diseño modular de la misma, el cual se describe en los párrafos siguientes.

A. Módulo de Comunicación

La conexión con la placa se realiza a través de la interfaz Micro USB que proveen los dispositivos móviles en los que funcionará la aplicación.

Para gestionar la comunicación, se debe implementar, por parte del dispositivo móvil, la Interfaz de Programación de Aplicaciones (API: Application Programming Interface) USB que Android provee para la conexión con dispositivos hardware externos.

El dispositivo móvil debe identificarse en la conexión, enviar sus datos a la Placa de Adquisición y ayudar a establecer el enlace.

B. Módulo de Envío

Para que la Placa de Adquisición envíe los datos que el dispositivo móvil solicita, y en la forma que los solicita, es necesario que éste realice un Pedido, estructurado según el protocolo local desarrollado (RRCOM). El mismo puede ser Simple o Compuesto, dependiendo si se requieren datos de uno

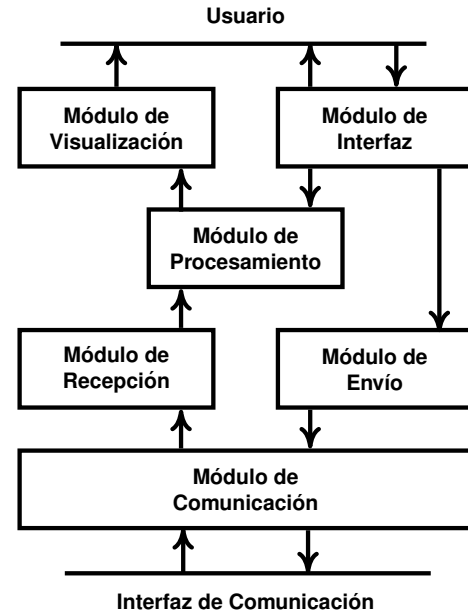


Fig. 5 Diseño modular

o de más canales simultáneamente. Este módulo se encarga de leer la lista que contiene los pedidos realizados por el usuario hasta el momento (e información adicional como la cantidad de pedidos, el tipo y número de canales) y arma la trama final que será enviada a la Placa, según la definición del Protocolo RRCOM. La trama resultante es la representada en la Fig.2.

Finalmente escribe la trama resultante en un buffer de salida, cuyo contenido está destinado a la Placa de Adquisición.

C. Módulo de Recepción

Inmediatamente, luego de haber realizado un Pedido a la Placa, se procede a esperar su respuesta. Cuando se detecta la cantidad de datos requeridos en un buffer de entrada, se los lee y se los almacena en un arreglo de bytes, al cual se le realiza una serie de verificaciones para detectar el tipo de señal proveniente, canal, datos válidos e inválidos. La trama recibida, según el Protocolo RRCOM, debe tener la estructura representada en la Fig.4. Si la trama recibida no cumple con alguna de las condiciones exigidas, entonces se la descarta y se realiza el pedido nuevamente.

D. Módulo de Procesamiento

Contiene la lógica de la aplicación, se encarga del tratamiento de los datos recibidos. También realiza acciones por pedidos del usuario y provee al Módulo de Visualización de datos procesados y listos para su disposición.

Algunas de las acciones que se realizan en este módulo, son intrínsecas de la señal recibida. Es decir, los datos de cada tipo de señal (Análogica o Digital), requieren un tratamiento diferenciado inmediatamente luego de ser recibidos y verificados. Como resultado de este proceso, se generan las estructuras adecuadas para que las señales puedan ser representadas gráficamente.

Otras acciones que se realizan en este módulo, son determinadas por el usuario, como por ejemplo:

- Operaciones entre Señales Analógicas: Suma o Resta.
- Modo “One Shot”: Representación instantánea y estática de la/s señal/es requerida/s.
- Personalizar Señales: Nombre y Color de cada canal requerido.

La razón por la que se decidió que todas las tareas mencionadas anteriormente se realizaran en el Dispositivo Móvil es para aprovechar las capacidades de procesamiento del mismo, además así se libera a la Placa de Adquisición de trabajos innecesarios.

E. Módulo de Interfaz

Se encarga de crear y disponer los elementos visuales e interactivos de la aplicación. El panel de control (con sus opciones de configuración), los botones, interruptores, perillas y panel de visualización fueron diseñados y representados de forma similar a los de un osciloscopio tradicional; a fin de aprovechar el conocimiento previo de la herramienta por parte del usuario y facilitar el uso intuitivo, reduciendo así los tiempos de capacitación en su operación.

Su principal responsabilidad es la generación y modificación de los pedidos, agregándolos a la lista sobre la cual trabaja el módulo de Envío (Pedidos Externos). También fija las condiciones y controles que los datos deberán verificar en el módulo de Procesamiento (Pedidos Internos). Estas operaciones son el resultado de la interacciones del usuario con los elementos de la interfaz.

F. Módulo de Visualización

Se encarga de la representación gráfica de las señales adquiridas por la Placa. El módulo debe recibir los datos del Módulo de Procesamiento, implementar las estructuras de datos correspondientes y mostrarlas en la pantalla del dispositivo, en la forma que el usuario elija.

Para este proceso, se investigó entre diferentes opciones y finalmente se optó por utilizar la librería abierta “MPAndroid-Chart”. Una ventaja adicional, pero no menor, es que permite el desarrollo para Dispositivos Móviles con iOS, ya que existe una versión de la misma librería para dicho Sistema Operativo.

V. RESULTADOS

A partir del desarrollo del Software para Dispositivos Móviles Android y el Firmware para la Placa de Adquisición EDU-CIAA-NXP, se logró obtener las herramientas planteadas como objetivo: un Osciloscopio - Analizador Lógico. Se probó el equipo con Generadores de diferentes señales y los resultados fueron los esperados.

En la Fig.6 se puede observar, en la vista del osciloscopio, la representación de dos señales senoidales. Se trabajó con señales con un rango de tensión de 0 [V] a 3.3 [V] y un rango de frecuencias de 0.1[Hz] a 40 [KHz].

En la Fig.7 se puede observar, en la vista del Analizador Lógico, la representación de ocho Señales Digitales. Se trabajó con señales con un rango de tensión de 0 [V] a 5 [V] y un rango de frecuencias de 0.1[Hz] a 3 [MHz].

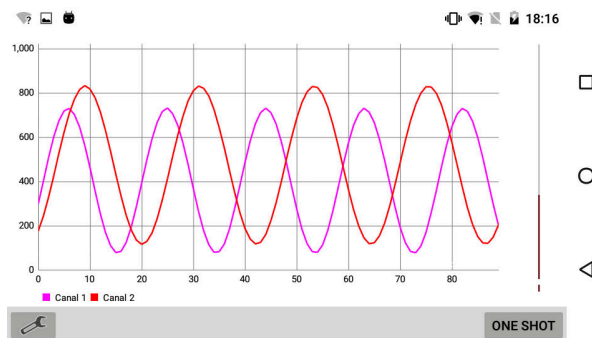


Fig. 6 Dos Señales Analógicas - Osciloscopio

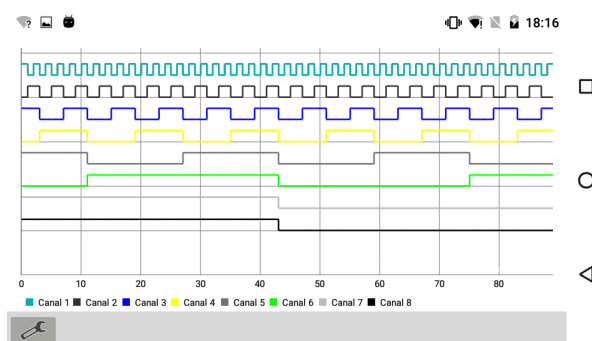


Fig. 7 Ocho Señales Digitales - Analizador Lógico

El panel de configuraciones se habilita pulsando el botón ubicado en la esquina inferior izquierda de la aplicación (Fig.8).

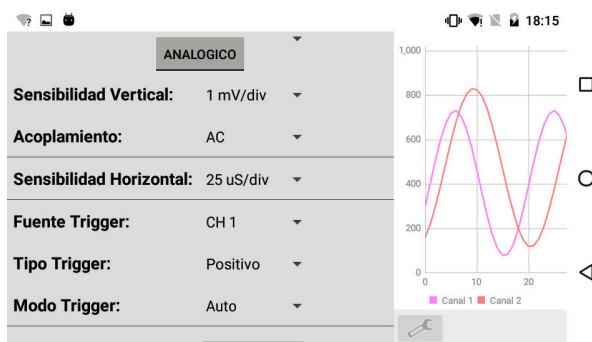


Fig. 8 Panel - Menú Osciloscopio

Dicho panel permite intercambiar entre el Menú del Osciloscopio y el del Analizador Lógico (Fig.9).

El usuario puede establecer configuraciones intuitivamente, con la misma facilidad que lo haría en las herramientas tradicionales. Por ejemplo en la Fig.10 se despliega las opciones para elegir la Sensibilidad Horizontal en el Menú del Osciloscopio.

VI. CONCLUSIONES

En este artículo, se diseñó e implementó un sistema para convertir un Dispositivo Móvil en un Osciloscopio y Analizador Lógico, usando una Placa de Adquisición la EDU-



Fig. 9 Panel - Menú Analizador Lógico

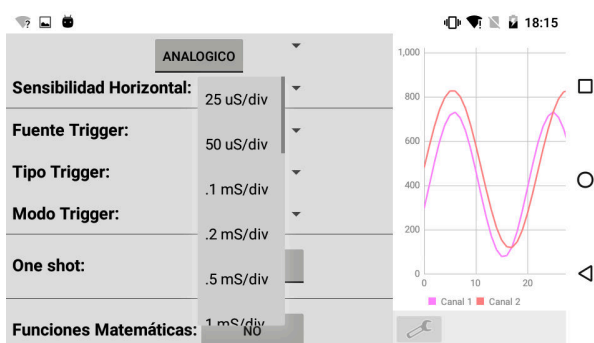


Fig. 10 Seleccionar Sensibilidad Horizontal - Menú Osciloscopio

CIAA-NXP. El mismo cumple con los objetivos planteados: tiene las mismas funcionalidades que los equipos tradicionales, y está abierto a mejoras significativas a costa de pequeños cambios gracias a su diseño modular.

A. Objetivos

Se cumplió con los objetivos propuestos, dado que el sistema resultante es:

- **Portable:** El usuario utiliza su propio Dispositivo Móvil y la Placa EDU-CIAA-NXP posee dimensiones reducidas en comparación con un Osciloscopio o un Analizador Lógico de Laboratorio. Esto permite a los usuarios independizarse del lugar de trabajo.
- **Económica:** Lo único que el usuario debe adquirir es la Placa EDU-CIAA-NXP, que tiene un costo de \$1000 que es muy bajo a comparación de un Osciloscopio y un Analizador Lógico, cuyos costos son alrededor de \$7000 y \$10000 respectivamente. Tanto el Dispositivo Móvil como los cables de transmisión de datos son provistos por el mismo.
- **Intuitiva:** La interfaz ha sido diseñada considerando tanto las costumbres del usuario utilizando aplicaciones móviles, como su conocimiento previo con Osciloscopios y Analizadores Lógicos.
- **Escalable:** Gracias a la modularidad de sus diseños, es posible cambiar/migrar el Firmware, el Software diseñados y las Plataformas utilizadas; a otras tecnologías sin grandes obstáculos para lograr nuevas y/o mejores funcionalidades.

Al trabajar con el microcontrolador LPC 4337 se logró obtener una herramienta con las siguientes características:

- **Osciloscopio:**
 - Canales Analógicos: 2
 - Rango de Tensión de Entrada: 0 [V] - 3.3 [V]
 - Frecuencia de Muestreo: 400 KSPS
 - Rango de Frecuencias: 0.1 [Hz] - 40 [KHz]
 - Sensibilidad Horizontal: 25 [uS/div] - 10 [S/div]
 - Resolución: 10 bits
- **Analizador Lógico:**
 - Canales Digitales: 2
 - Rango de Tensión de Entrada: 0 [V] - 5 [V]
 - Rango de Frecuencias: 0.1 [Hz] - 3 [MHz]

Al trabajar con la placa EDU-CIAA-NXP (una plataforma nacional que se está penetrando con rapidez en ámbitos industriales y académicos) este equipo podrá ser utilizada en colegios secundarios, universidades y centros de investigación a muy bajo costo.

Se podría mejorar el Rango de Tensión de Entrada agregando un Shield/Poncho que se encargue de acondicionar las señales. Con dicha modificación se puede competir con Osciloscopios Tradicionales en los ámbitos mencionados.

B. Futuras mejoras

Durante el desarrollo se identificaron algunas posibles mejoras para la herramienta. Algunas de ellas son:

- Utilizar otras tecnologías para la adquisición de datos, por ejemplo un Microcontrolador LPC4370 capaz de lograr 80MSPS, con lo cual podrían visualizar señales analógicas de frecuencias notablemente más altas.
- Desarrollar la aplicación para Dispositivos Móviles con iOS. Esta medida no fue implementada en el sistema original porque Apple requiere, entre otras cosas, una licencia especial (MFi) ya que se involucra Hardware externo y esto hubiese traído ciertas complicaciones y demoras innecesarias al proyecto.
- Funcionalidad de guardado de datos para futuros análisis.
- Adquisición de canales adicionales.

AGRADECIMIENTOS

Este trabajo esta financiado por el CIUNT, Consejo de Investigaciones de la UNT, bajo el proyecto E543/1.

REFERENCIAS

- [1] API Guide for Android Development
- [2] Android Open Accessory (AOA)
- [3] LPC435x/3x/2x/1x - Datasheet
- [4] UM10503 - LPC43xx/LPC43Sxx ARM Cortex-M4/M0 multi-core micro-controller
- [5] Library MPAndroidChart - Philipp Jahoda

Avances en migración a 32 bits de una placa controladora orientada a mediciones industriales

Dimensionamiento, alternativas y compatibilidad requerida

Leonardo Garberoglio (1) , Rafael B. Oliva (2)

(1) GADIB (Grupo de Análisis, Desarrollo e Investigaciones Biomédicas), UTN-FRSN, San Nicolás, Buenos Aires

(2) Universidad Nacional de la Patagonia Austral - Área Energías Alternativas y L&R Ingeniería

Río Gallegos, Argentina

roliva@lyr-ing.com

Resumen—Este trabajo presenta los avances en la migración a arquitectura ARM de una placa controladora orientada a aplicaciones industriales, basada en la arquitectura AVR de 8 bits y desarrollada a través de un subsidio ANR / FONTAR en 2009. Debido a las aplicaciones en que se utiliza (mayormente en instalaciones de energía renovable, monitoreo y registro de variables eléctricas) resulta importante en la migración la compatibilidad mecánica y de dimensiones de la placa, más la compatibilidad eléctrica y de programa con algunos de los periféricos existentes. Otros puntos relevantes son la reducción de consumo y sobre todo el aumento en las prestaciones y capacidad de cómputo de la nueva versión (*Abstract*)

Palabras clave — Placa controladora industrial, sistema embebido, arquitectura ARM

I. INTRODUCCION

Existe una importante oferta internacional de placas y subsistemas aplicables a control y medición, con grados muy diversos de capacidades, potencias de cómputo y costos. Sin embargo, algunas aplicaciones de tipo industrial pueden ameritar la producción local de equipos con capacidades específicas, que cubran la brecha entre los desarrollos para hobbyistas y las prestaciones de un equipo de alta gama. Se presenta en este caso los avances en el diseño de una placa que migra desde una arquitectura de controladores de 8 a 32 bits manteniendo características de formato físico buscando lograr una expansión de las capacidades de cómputo y rendimiento.

II. SISTEMA ACTUAL CL2BM1

A. Contexto del desarrollo

En el marco del proyecto ANR N° SC002/2003 (reformulado 2004), denominado “Sistema de Adquisición de Datos (Data Logger) de Bajo Costo y Arquitectura Abierta para Aplicaciones Ambientales y de Energía” [3] se desarrolló una placa basada en arquitectura AVR de 8 bits, con características de bajo costo, facilidad de uso con el fin específico de almacenar mediciones de temperatura, humedad, presión atmosférica, velocidad y dirección de viento, radiación y parámetros eléctricos de sistemas de energía renovable

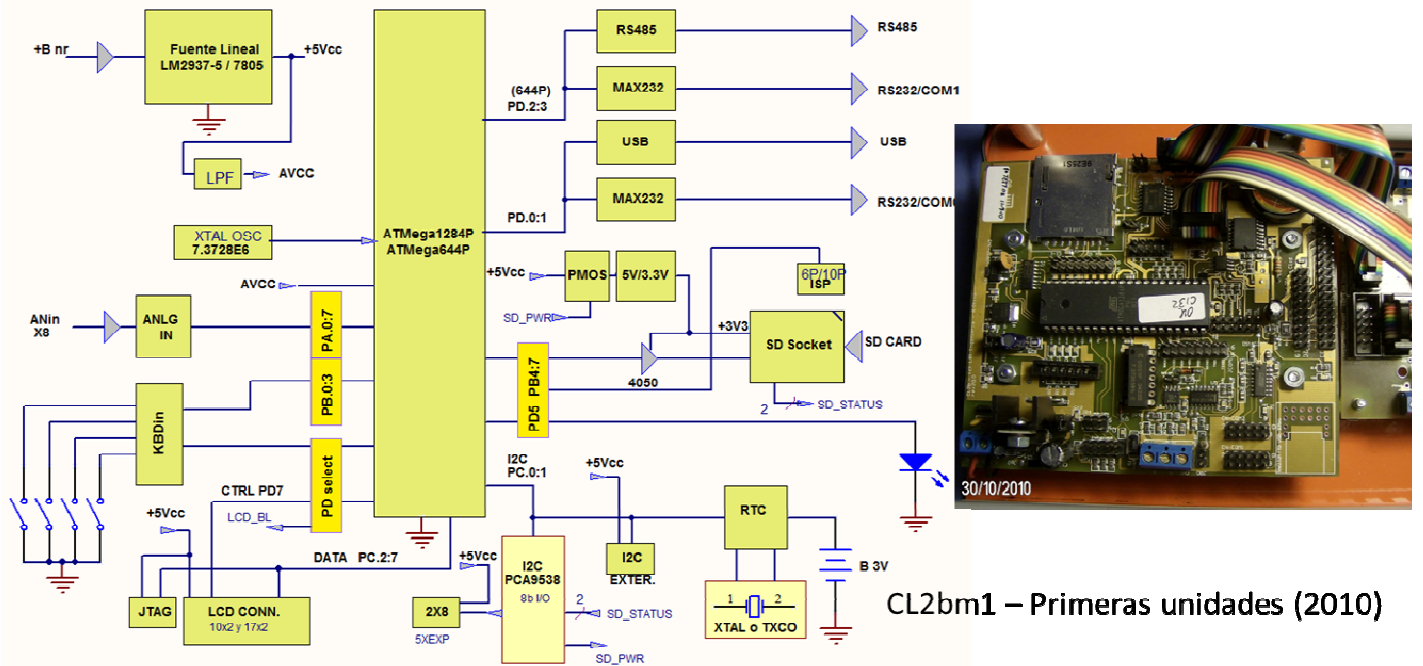
(tensión, corriente, potencia), con períodos de almacenamiento y funciones estadísticas configurables. La memoria de almacenamiento es de tipo no volátil. Se desarrolló un primer sistema que trabaja con almacenamiento EEPROM serial, y se culminó el desarrollo de la placa CL2bm1[7] con almacenamiento en tarjetas MMC/SD, basado en un procesador ATmega1284P [11]. En la Figura 1 se muestra el diagrama en bloques y las primeras placas producidas en 2010. El mercado principal que se buscó cubrir fue el académico-científico y el industrial. En [2] se presentaron aspectos de un sistema de medición PWRC2 basado en la placa CL2bm1 para obtener la curva característica de potencia $P(V)$ de un aerogenerador de baja potencia, donde V es la intensidad del viento, que es utilizado actualmente por el INTI [4], [5], [12] para la evaluación de aerogeneradores en consistencia con la norma IEC61400-12-1 [1] en su Anexo H.

En la Figura 2 se muestra la implementación mencionada para ensayo de pequeños aerogeneradores. De cada turbina se mide tensión y corriente producida, a efectos de determinar potencia eléctrica. Simultáneamente se mide intensidad y dirección de viento, que se envían por un bus RS485 a la unidad CPU. Allí el firmware realiza el proceso estadístico, almacenando los valores a intervalos requeridos por el ensayo en tarjetas tipo SD. En estos equipos el procesador ATmega1284P corre a 14.76 MHz y utiliza una unidad periférica M4/E con controlador Cypress PSoC 29466 [6], que implementa un convertor A/D triple TRIADC de 13 bits, para medición simultánea de tensión y corriente. Ambas unidades se conectan vía bus I2C. Externamente, al pie de las torres de medición se utiliza una unidad METEO para suministrar datos meteorológicos vía RS485.

B. Necesidad de evolución a mayores prestaciones

Tanto en las aplicaciones de ensayo [4] como en otras más recientes se registra la demanda de interfaces más amigables, mayor capacidad de registro de eventos y alarmas, y expansión de posibilidades de acciones de control y procesamiento de datos. Se optó por iniciar el desarrollo de una placa denominada CL3bm1 de dimensiones similares, pero con arquitectura ARM-Cortex M4F de 32 bits [8]

CL2_b Proto (AtMega644/1284P) Rev5F-2010



CL2bm1 – Primeras unidades (2010)

Fig. 1. Diagrama de la Placa CL2bm1 (izq) e implementación de las primeras unidades en 2010

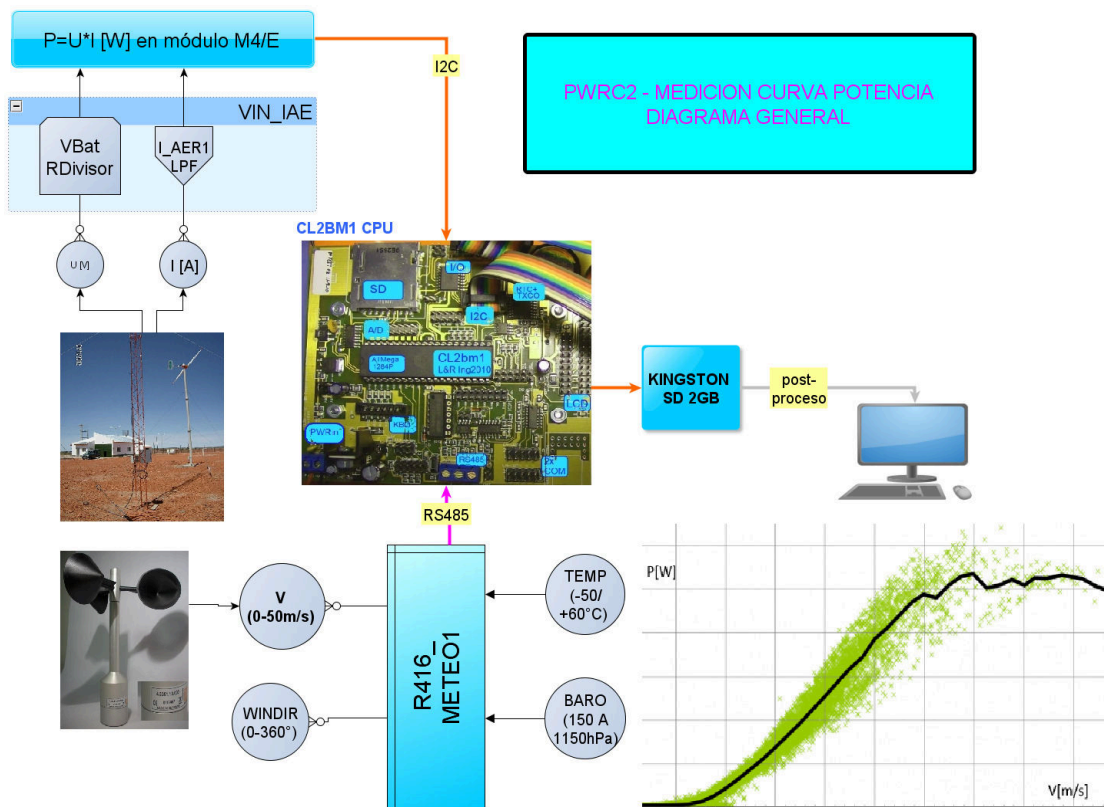


Fig. 2. Diagrama de la medición de curva de potencia basado en CL2bm1 - Campo de Pruebas INTI en Cutral C6 - Neuquén.

En la etapa de diseño, se fijaron algunas características deseables del microcontrolador a seleccionar para la placa CL3bm1:

- i. ARM Cortex M4F, debido a su amplia disponibilidad y adecuada relación precio / performance
- ii. 512K Flash mínimo, cuadruplicando la memoria disponible actual
- iii. 32K RAM mínimo
- iv. Un mínimo de 20 GPIO
- v. 2 Puertos Serial (1 de ellos con RS485)
- vi. 1 puerto USB
- vii. 2 puertos SPI, uno de ellos dedicado a display TFT
- viii. 2 puertos I2C, tolerantes a 5V con conector estándar.
- ix. Conversor A/D 12bits mínimo, 8 entradas
- x. Encapsulado LQFP 64 para facilitar armado.

En cuanto a los ítems a cumplir por la placa, se fijaron:

- 1. Fuente switching y management de periféricos por hardware (para reducción de consumo)
- 2. Incorporar interfase SD industrial, formato grande
- 3. Incorporar reloj tiempo real con TCXO, tipo DS32khz
- 4. Mantener el formato físico del predecesor CL2bm1.

En base a ello, después de un proceso de selección utilizando las herramientas provistas por ST se optó por un controlador STM32F411RE de arquitectura Cortex M4/F a 100 MHz con FPU y MPU, 512 KB de Flash y 128 KB de SRAM, ADC de 12 bits y USB (Figura 3) en encapsulado LQFP64 [9] .

El proveedor tiene además una herramienta IDE gratuita sin límite de código, librería HAL (con algunas limitaciones), y disponibilidad de una herramienta de inicialización de periféricos (CubeMX) que permite de forma gráfica obtener el código para configurar/inicializar los periféricos del uC, con un significativo ahorro de tiempo de desarrollo. No fue un detalle menor el hecho de que uno de los autores contara con experiencia en el uso de dichas herramientas, y varios diseños ya realizados. Por otro lado, esta línea de controladores tiene en cantidades como las demandadas por la aplicación (100x) un costo FOB del orden de 5.5usd, apenas superior al del controlador ATMEga1284P utilizado hasta el momento.

Asimismo, las placas de evaluación son de bajo costo, e incluyen la herramienta de debug estándar, tanto del procesador incluido como sobre placas de terceros. Existe además un caudal de información abundante en internet, foros y listas de correos, como así también notas de aplicación enfocadas a resolver diversos problemas típicos.

En la Figura 4 se observan detalles de un sistema similar en cuyo desarrollo participó el primer autor de este trabajo, basado en el STM32F411 de ST, utilizando un encapsulado similar. Se trata de un equipo de oteomisiones acústicas, desarrollado por GADIB-UTN [17] para la empresa Itam Care en el marco del Empretecno 6 /2013 [18].

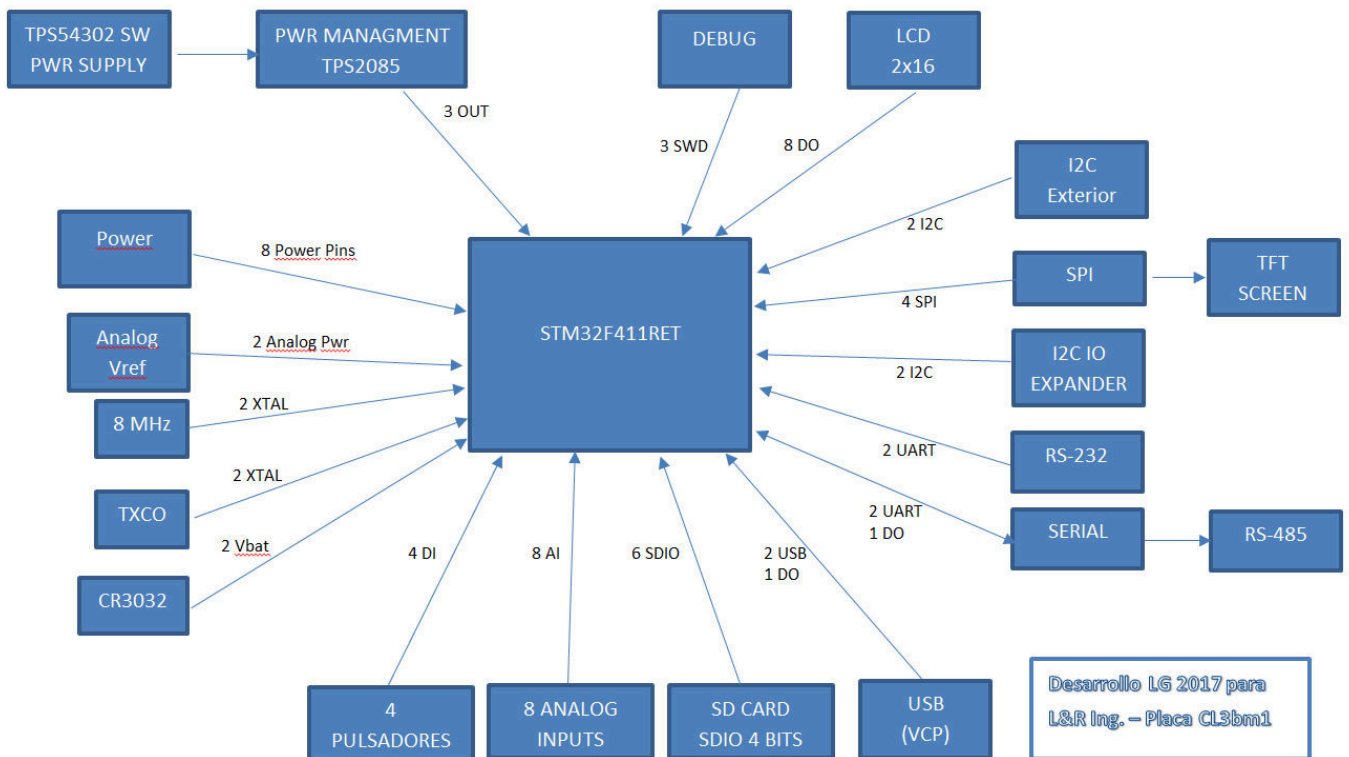
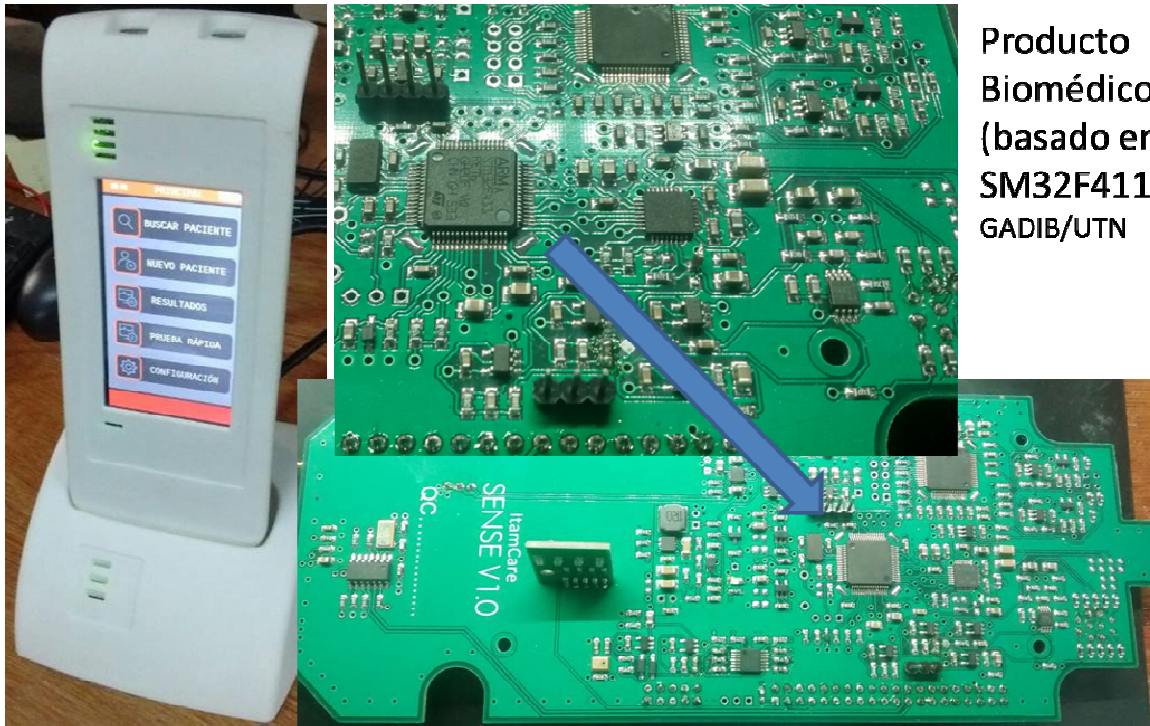


Fig. 3. Diagrama general del diseño en ejecución



Producto Biomédico (basado en SM32F411) GADIB/UTN

Fig. 4. Sistema similar en cuyo desarrollo participó el autor, basado en STM32F411.

En dicha aplicación el uC es el encargado de realizar las secuencias requeridas en los estudios para la detección de otoemisiones acústicas en recién nacidos, así como del procesamiento de las señales en tiempo real. Aunque no se trata de un controlador nuevo, tiene la ventaja de que sus librerías tienen un alto grado de depuración y hay un

importante caudal de aplicaciones resueltas de libre disponibilidad. Para los desarrollos preliminares se utilizará una placa de evaluación ST con dicho controlador, aunque en una versión LQFP100.

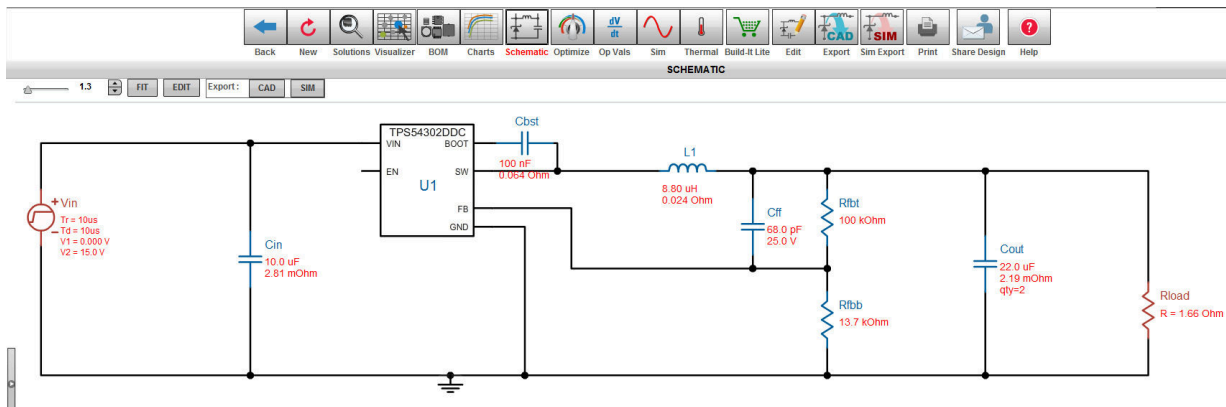


Fig. 5. Esquemático sugerido por Webench para la fuente de alimentación SMPS.

III. DISEÑO DE LA PLACA CL3BM1

A. Fuente de Alimentación

En el diseño de la nueva versión y por razones de eficiencia se ha reemplazado la fuente lineal basada en LDO LM2937-5 para obtener 5V y LM2937-3.3 para los 3.3 V por un circuito switching buck tradicional, para cuyo diseño se utilizó la

herramienta Webench de la firma Texas Instruments [10]. Los parámetros ingresados al aplicativo fueron tensión de entrada V_{in} entre 7 y 20 VCC y salida en V_o de 5 V a 3 A máximo. Asimismo, entre las múltiples opciones que entrega (160 en primera instancia) se agregaron como parámetros adicionales de diseño una eficiencia mayor o igual al 85% y con un costo no superior a los USD5.00 para todo el sistema. Debido a la importancia de mantener el formato original de la placa (el

tamaño de la fuente no puede ser superior al actual, muy simple) se agrega como filtro adicional los que poseen menos de 10 componentes, y se permite limitar también el área física que ocupa la fuente (utilizando componentes de mayor frecuencia). Limitamos el área a 400mm^2 lo cual restringió el diseño a solo 14 partes. De ellos, el más eficiente (90%), con menor área (232mm^2) y más económico (USD 2.61 para toda la fuente) resultó el TPS54302. Esta unidad tiene la ventaja adicional de reemplazar internamente el habitual diodo rápido de bloqueo por un MOSFET, y utilizar un encapsulado SOT23-6 Thin muy compacto. Se verificó la disponibilidad del mismo en los proveedores habituales, como así también su precio y la disponibilidad de componentes pasivos de valores próximos a los indicados por la herramienta de diseño (Figura 5).

Finalmente para la obtención de los 3.3V necesarios para el funcionamiento del uC y algunos periféricos se optó por un LDO (NCP1117ST33T3G) similar al utilizado en la CIAA-NXP [19], debido a que la diferencia de tensión ($5\text{V} - 3.3\text{V}$) es pequeña al igual que la corriente necesaria.

B. Distribución de potencia en la placa

Una característica deseable en para el sistema propuesto, a efectos de darle mayor flexibilidad es la capacidad de poder conmutar por software los periféricos que no se utilicen en cada aplicación, con el objetivo de maximizar la eficiencia energética del equipo. Para la placa CL3bm1 se eligió el TPS2085 de Texas Instruments, que es un circuito conmutador cuádruple (*Power Switch*) diseñado para la administración de potencia a baja tensión. Cada switch posee una entrada de habilitación independiente (Figura 6). En este diseño han elegido los siguientes periféricos conmutables:

- SD Card
- TFT Screen + LCD Display
- RS485 driver
- I2C Expander

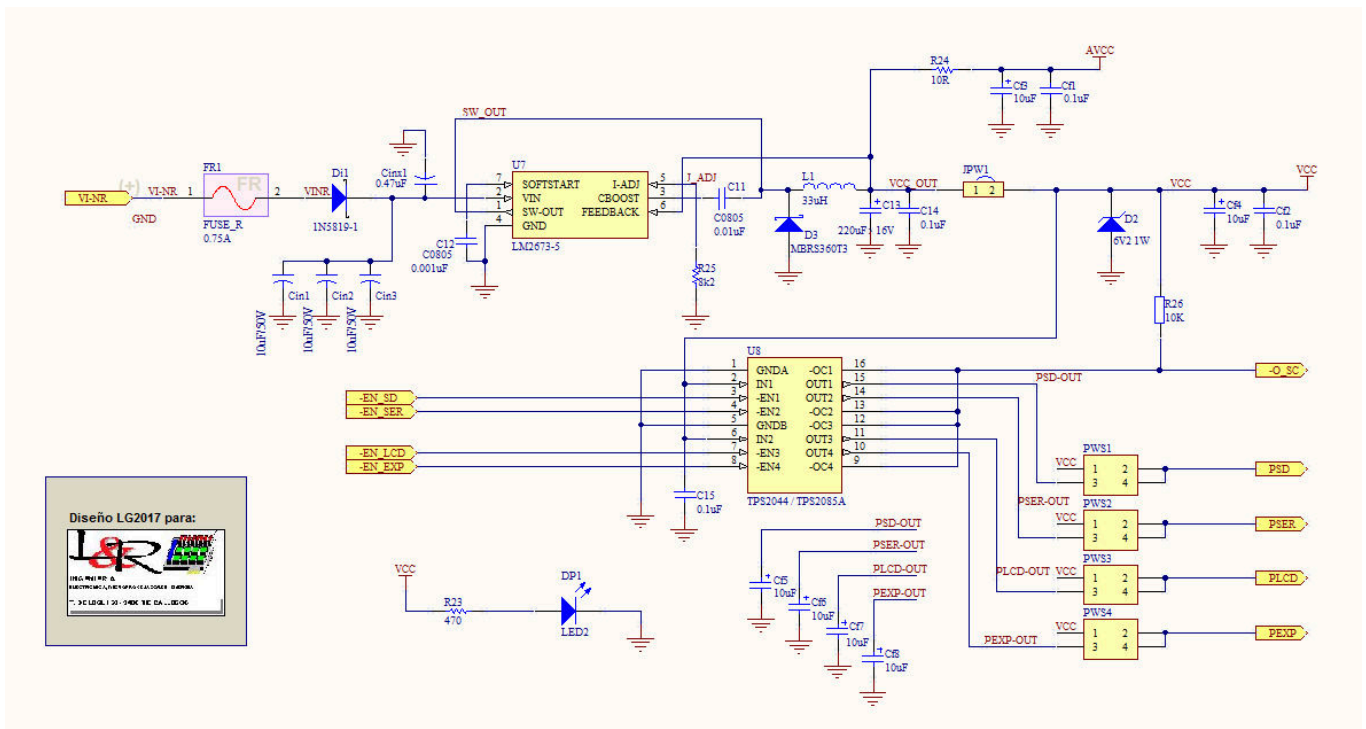


Fig. 6. Primera iteración (con fuente antigua) del sistema de PowerManagement en placa CL3bm1, utilizando TPS2085

C. SD Card

La placa CL2bm1 utiliza una interfaz SPI para manejar la escritura/lectura de una tarjeta SD tipo industrial. La librería utilizada es un FFS propietario (2006) de la extinta compañía PRIIO de Indianápolis, EEUU, que compila utilizando la herramienta Codevision AVR de HPInfoTech [10]. El sistema utiliza un FAT32 convencional pero con limitación a 2 GB. Debido al uso de dicha librería y a la reducida frecuencia de operación del ATmega1284 (en la CL2bm1 es 14.76 MHz) en algunas aplicaciones resulta una limitante el acceso a la tarjeta.

Para la nueva placa se utiliza la conocida Elm-Chan FatFS, de libre disponibilidad [16]. Esta librería está totalmente

programada en C99 estándar, pudiendo ser portada a casi cualquier sistema embebido. En particular las librerías HAL de ST y el generador de código CubeMX implementan dicha librería de forma nativa, pudiendo configurar la misma en forma gráfica.

La interfaz SPI del microcontrolador puede funcionar a 20MHz lo que otorga una tasa de transferencia significativamente superior a la de la versión CL2bm1. Debido a una cuestión de compatibilidad, se ha decidido mantener el conector SD Hirose de la línea DM1 [15]

D. Display TFT y LCD 2x16

Otro punto fundamental en la actualización del sistema es el agregado de una interfaz gráfica mucho más potente. El antiguo sistema contaba con una interfaz tradicional al controlador Hitachi HD44780 para LCD, limitado a 4 líneas de 20 caracteres. En muchas aplicaciones esto resulta suficiente, pero frecuentemente resulta inadecuada la limitación a formato texto y sistema de menús muy básicos.

Se ha decidido mantener la interfaz LCD alfanumérica pero se ha agregado un segundo puerto SPI para el manejo de un display tipo TFT-LCD (Thin Film Transistor / LCD), que se encuentran disponibles con diversos controladores. Algunos son muy conocidos como el PCD8544 (display del antiguo teléfono Nokia 5110) y otros más avanzados como el ILI9340 o 41. En cuanto a librerías gráficas estándar se cuenta con la uGFX [14], la cual permite de forma simple la programación de potentes interfaces de usuario, vistosos gráficos y hasta animaciones. La firma ST Microelectronics posee una librería propia para el controlador ILI9341, que además cuenta con soporte nativo en la uGFX, por lo que el uso de la misma no requiere un gran esfuerzo de desarrollo.

E. RTC con TCXO

Todo sistema registrador necesita una base de tiempo precisa y estable. Esto fue una premisa de diseño del CL2bm1, debido al rango de aplicaciones al que apuntaba, por lo cual al tradicional chip PCF8563 RTC se acopló un TCXO DS32kHz de Dallas, a pesar de su significativo costo inicial. El uso de este tipo de osciladores con compensación interna de temperatura se hace necesario cuando se desea mantener la fecha/hora de un equipo que necesita funcionar durante períodos largos sin acceso a GPS u otras correcciones remotas vía web, y ante la potencial variación significativa de la temperatura de operación de la placa. En esta nueva versión CL3bm1 sistema se mantiene esta base de tiempo de buena estabilidad, aunque se utilizará el RTC interno del uC, lo cual ahorra espacio de hardware y reduce el costo de la placa.

F. Entradas Analógicas y DMA

El STM32F411 posee un conversor A/D (Analogico a Digital ó ADC) de 12 bits y hasta 8 canales seleccionables en el encapsulado elegido. Para la placa CL3bm1 se prevé el uso de las 8 entradas para mediciones de señales analógicas, aunque su acondicionamiento se realizará en una placa externa. Para aplicaciones que requieran mayor velocidad de procesamiento de señales, se incorporó el manejo de dichos canales por DMA ó acceso directo a memoria, liberando al core uC de la tarea de la adquisición y almacenamiento en RAM y permitiendo que, por hardware, los canales del ADC se conecten directamente a la RAM.

IV. CONCLUSIONES

El presente trabajo muestra los avances en la migración de un diseño de placa con controlador de 8 bits de 2009 a una

arquitectura de procesador más nueva, a efectos de lograr una ampliación de capacidad utilizando una infraestructura de hardware existente.

AGRADECIMIENTOS

Los autores agradecen la cooperación del GADIB (Grupo de Análisis, Desarrollo e Investigaciones Biomédicas), UTN-FRSN, y el apoyo de la Universidad Nacional de la Patagonia Austral para la preparación del presente trabajo.

REFERENCIAS

- [1] IEC61400-12-1 (2005). "Wind Turbines –Part 12-1 Power Performance Measurements of electricity producing wind turbines" – International Standard 61400-12-1, IEC (International Electr.Com.) Geneva, Suiza. .
- [2] Oliva, R (2014) "Evaluación de incertidumbre en mediciones de potencia eléctrica en registradores automáticos - Caso de implementación para medición de curva de potencia de pequeños aerogeneradores", CASE / SASE 2014
- [3] http://www.lyr-ing.com/Assets/Images/Projects/ANR/ANR_LyR-TercerInformeTecnico08-2009a.pdf
- [4] A. Zappa, R. Oliva, J. Duzdevich, G. Martín, (2013) EVALUACIÓN DE CURVA DE POTENCIA EN PLATAFORMA DE ENSAYO PARA AEROGENERADORES DE BAJA POTENCIA, Acta de Asociación Argentina de Energías Renovables y Medio Ambiente - Vol. 1, pp. 06.89-06.98, 2013. ISBN 978-987-29873-0-5
- [5] [INTI-RUT N°FM-102-3090U, 2014] Certificado de Calibración / Medición RUT N° FM-102-3090 Unico, unidad PWRC2/12V-100A Estación #1 /L&R Ingeniería, solicitado por INTI-UT Energía-Neuquén, 5 y 6 de febrero de 2014. Emitido por INTI-Metrología, Av.G.Paz 5445 B1650WAB San Martín, Buenos Aires
- [6] <http://www.cypress.com/products/psoc-1>
- [7] http://www.lyr-ing.com/DocumentosLyR/CL2bm1/CL2bm1_caracteristicas_v3.pdf
- [8] <https://www.arm.com/products/processors/cortex-m/cortex-m4-processor.php>
- [9] http://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32f4-series/stm32f411/stm32f411re.html
- [10] <http://www.ti.com/lscs/ti/analog/webench/power.page>
- [11] <http://www.atmel.com/devices/ATMEGA1284P.aspx>
- [12] <http://www.inti.gov.ar/neuquen/index.php?seccion=aerogeneradores>
- [13] <http://www.hpinfotech.ro/>
- [14] <https://ugfx.io/>
- [15] <https://www.hirose.com/product/en/products/DM1/>
- [16] http://elm-chan.org/fsw/ff/00index_e.html
- [17] http://www.frsn.utn.edu.ar/frsn/selec_seccion.asp?IDSeccion=266&IDSub=368&IDSubSub=405
- [18] http://www.agencia.mincyt.gov.ar/upload/Res.409-13_EMPRETECNO_06.pdf
- [19] http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:hardware:ciaa_nxp:ciaa_nxp_inicio

Sistema de visualización de precios para supermercados

Lupi, Daniel^{1,2}; Zaradnik, Ignacio¹; Turconi, Diego¹; Dominguez, Facundo¹

¹Laboratorio de Inteligencia Ambiental Departamento de Ingeniería e Investigación Tecnológica, Universidad Nacional de la Matanza. Buenos Aires, Argentina; ²Fundación Argentina de Nanotecnología. Buenos Aires, Argentina
izaradnik@gmail.com

Se presenta un sistema de visualización de precios para supermercados basado en tecnología RFID/NFC (Identificación por Radio Frecuencia / Comunicación de Campo Cercano). Se detalla el hardware empleado, así como el firmware y el software desarrollado. El presente trabajo es parte de lo realizado en el marco del proyecto “Utilización de electrónica impresa para el desarrollo de sistemas de seguimiento, identificación y trazabilidad de productos manufacturados”.

El objetivo de dicho proyecto es la formación de recursos humanos en electrónica impresa, la cual brinda una solución a las necesidades de sensores y componentes electrónicos de bajo costo y alta calidad, entre las que podemos nombrar: pantallas flexibles, etiquetas y envases inteligentes, libros interactivos, sensores de presión y temperatura, etc. Además, permite reducir el tiempo de fabricación y los costos de producción, ya que no necesita una sala limpia como en el caso de la microelectrónica. A fin de estudiar los procesos, técnicas, materiales y otros se decidió tomar como aplicación testigo los sistemas inalámbricos de seguimiento y trazabilidad de productos, la cual dio nombre al proyecto, donde la electrónica impresa se puede emplear para la fabricación de las antenas de los mismos. Esta decisión representa una excelente área de estudio, a raíz del inagotable campo de aplicaciones que pudimos observar, como por ejemplo aplicaciones agropecuarias, médicas e industriales, que nos permite una gran variedad de ámbitos para el estudio de los procesos, las técnicas y los materiales asociados a la electrónica impresa.

Como parte del proyecto se planteó: un estudio del estado del arte de la tecnología asociada a los sistemas planteados [1]; un estudio de las distintas tintas, sustratos y materiales empleados en electrónica impresa; el desarrollo de una metodología para el diseño de las antenas; la elaboración de un banco de pruebas para las antenas diseñadas; y la implementación de una aplicación concreta de la tecnología. Entre las distintas aplicaciones se decidió por un sistema de visualización de precios para supermercados. La decisión se basó en la gran variedad de sustratos que ofrecen los distintos productos de un supermercado, lo que permite evaluar fácilmente el potencial de la Electrónica impresa.

El sistema está formado por: tags RFID/NFC, colocados en cada producto; un dispositivo lector-visualizador; un software de base de datos; y un software de gestión. Estos últimos dos

elementos forman parte de un modesto sistema EPR (enterprise resource planning - sistemas de planificación de recursos empresariales). Estos sistemas están pensados para proyectar y gestionar todos los recursos y sus usos en una compañía, y optimizar e incorporar todos los procesos de la organización. Sus aplicaciones van desde logística y transporte hasta el ámbito aeroespacial y defensa [2], y sus funcionalidades o módulos básicos incluyen: administración financiera; de recursos humanos; de producción; de ventas, distribución y logística; de la relación con los clientes y proveedores; de la cadena de suministros entre otros [3].

A raíz que la implementación del sistema se realizó en simultáneo con las otras tareas, es que para el mismo se emplearon tags comerciales y no los desarrollados. Los tags empleados están basados en el chip NTAG203 de NXP [4], compatible con los tag Tipo 2 de NFC Forum, figura N°1a. La figura N°1b muestra una imagen del lector empleado para la aplicación, el cual está basado en el chip PN532 de NXP [5]. El mismo soporta los estándares ISO/IEC 14443A/B.

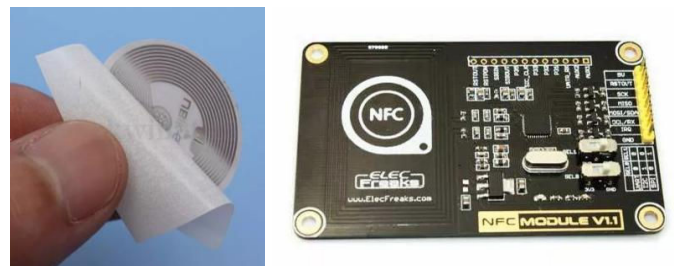


Figura N°1a. Tag NFC. N°1b Lector NFC

Se empleó la interfaz UART del dispositivo lector para conectarse con la placa de control y comunicación, la cual es el kit de desarrollo del módulo ATSAMW25[6]. El mismo está formado por un microcontrolador Cortex M0+ (ATSAMD21) y un transceptor de Wifi (ATWINC1500). A fin de visualizar la descripción y el precio del producto se conectó a la interfaz I2C un display de 16x2 caracteres de la firma Winstar, la figura N°2 presenta una imagen del sistema.

El sistema funciona de la siguiente manera: el producto, del cual se desea hacer la consulta, se acerca al lector; este realiza la lectura del número de identificación (ID) del tag y lo envía a través del puerto UART al módulo de control y comunicación; recibido el dato y verificada su integridad el mismo se reenvía a través de una conexión Wifi a un servidor

de base de datos; recepcionado el dato se busca en la base de datos la descripción y el costo, los cuales se envían al módulo de control y comunicación a través de la misma conexión Wifi y finalmente los datos se presentan en el display de caracteres asociado al módulo.

Para el desarrollo del firmware se tomó como base del proyecto el ejemplo WINC1500_WIFI_SERIAL_EXAMPLE, el cual es parte del Atmel Software Framework 3.29.0 (ASF) incluido en el IDE. El ejemplo transmite la información recibida por el puerto serie a una dirección IP. Como parte del proyecto se implementó la configuración de parámetros (dirección IP, el puerto, el nombre y la clave de la red Wifi) a través del puerto serie, la comunicación por I2C y el manejo del display.

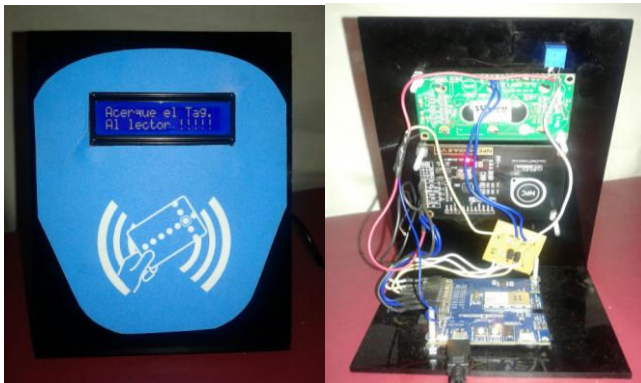


Figura N°2. Hardware del sistema.

En lo que respecta al software de base de datos y gestión se implementó un bloque que podría ser parte de muchos de los módulos previamente mencionados. Dicho bloque se encargará del Alta/Baja/Modificación de productos. La figura N°3 presenta la pantalla principal y la de nuestro módulo.



Figura N°3. Interfaz del Software.

El software desarrollado posee la misma arquitectura de los sistemas ERP, servidor / cliente con tres niveles, formada por tres capas lógicas: una capa de presentación, la cual consta de una interfaz gráfica de usuario; una capa de aplicación,

compuesta por programas que reciben y procesan las solicitudes generadas por los usuarios a través de la capa de presentación y una capa de base de datos, donde se gestionan los datos operativos y empresariales de toda la empresa [3].

Para el desarrollo de la capa de base de datos se empleó MySQL [7], el cual es un sistema de bases de datos relacional, multihilo y multiusuario con licencia GNU GPL y para las capas de presentación y aplicación se empleó la versión de prueba de LabWindows/CVI [8], el cual es un entorno de desarrollo integrado ANSI C de National Instruments que incluye herramientas de ingeniería con bibliotecas integradas para análisis y diseño de UI (interfaces de usuario).

El desarrollo realizado logra ejemplificar de modo sencillo el uso de la tecnología asociada a los sistemas de seguimiento, identificación y trazabilidad de productos manufacturados, y el uso de dispositivos RFID/NFC para este tipo de aplicaciones.

Si bien no fueron tenidas en cuenta cuestiones de seguridad en la comunicación inalámbrica, el módulo ATSAMW25 posee el circuito integrado de criptoautenticación ATECC508, el cual a futuro permitirá mejorar la seguridad.

Si bien, como se planteó previamente, el sistema se realizó en simultáneo con las otras tareas por lo que se usaron tags comerciales, y no los diseñados e impresos por nosotros, se evaluó el sistema con tags pertenecientes al kit de desarrollo de chip seleccionado, NT3H2111 [9]. Los resultados obtenidos fueron satisfactorios además de que permitió evaluar distintos factores de forma y sustratos.

REFERENCIAS

- [1] Canziani, Monica; Lupi, Daniel; Ortiz, Juan José; Slawski, Javier; Zaradnik, Ignacio, "Tecnologías inalámbricas para sistemas de seguimiento, identificación y trazabilidad de productos", Congreso de microelectrónica aplicada 2016, San Luis, Argentina.
- [2] Erik Fossler, Ole Henrik Leister, Carl Erik Moe, "ERP systems and competitive advantage: Some initial results", <http://www.diku.dk/~henglein/3gERP-workshop-2008/papers/fosser-leister-moe-newman.pdf>, última visita 30/04/2017.
- [3] Wei She, Bhavani Thuraisingham, "Security for Enterprise Resource Planning Systems", https://www.utdallas.edu/~bxt043000/Publications/Journal-Papers/DAS/J46_Security_for_Enterprise_Resource_Planning_Systems.pdf, última visita 30/04/2017.
- [4] NXP, "NTAG203 NFC Forum Type 2 Tag compliant IC with 144 bytes user", Rev. 3.0 — 17 October 2011.
- [5] NXP, "PN532/C1 Near Field Communication (NFC) controller", http://www.nxp.com/documents/short_data_sheet/PN532_C1_SDS.pdf, última visita 30/04/2017.
- [6] ATMEL, "ATSAMW25MR210PB", http://www.atmel.com/Images/Atmel-1-42618-SmartConnect-ATSAMW25-MR210PB_Datasheet.pdf, última visita 30/04/2017.
- [7] MySQL, <https://www.mysql.com/>, última visita 30/04/2017.
- [8] National Instruments, <http://www.ni.com/lwcv/whatis/esa/>, última visita 30/04/2017.
- [9] NXP, "NT3H2111/NT3H2211 - NTAG I2C plus, NFC Forum Type 2 Tag compliant IC with I2C interface", http://www.nxp.com/documents/data_sheet/NT3H2111_2211.pdf, última visita 20/12/2016.

Foro Tecnológico

Resumen

FPGAs, HDLs y ASICs

Unidad de control de actuadores y comunicaciones basada en FPGA.

Diseño, desarrollo e implementación

Emiliano H. Prato¹, Ariel Dalmas Di Giovanni¹, Daniel A. Pastafiglia¹
Laboratorio de Técnicas Digitales - Departamento de Electrónica Aplicada
Instituto de Investigaciones Científicas y Técnicas para la Defensa (CITEDEF)
Villa Martelli, Buenos Aires, Argentina.
¹{eprato, adigiovanni, dpastafiglia}@citedef.gob.ar

Abstract— El presente trabajo describe el diseño, desarrollo e implementación de una unidad para el control de actuadores de un vehículo aéreo no tripulado y la adaptación de las interfaces de comunicaciones con otras unidades de a bordo. La unidad se basa en una FPGA, es por tanto que el trabajo describe las justificaciones de selección de este dispositivo, la metodología de desarrollo y los recursos utilizados. Se pone un fuerte énfasis en la importancia y las ventajas obtenidas de utilizar un diseño modular, escalable y genérico, lo que da lugar a que la unidad encuentre aplicación más allá de los vehículos no tripulados, como: vectores del tipo sonda, nano plataformas satelitales, entre otros.

Keywords—FPGA, VHDL, PWM, procesamiento paralelo.

I. INTRODUCCIÓN

En los últimos tiempos los drones o VANTs (Vehículo Aéreo no Tripulado) pasaron de ser temas exclusivos de áreas de Defensa, de investigación y desarrollo a ser accesibles por cualquier persona para un uso particular. Sin embargo, en el ámbito de la investigación y desarrollo continúa siendo una temática sobre la que se trabaja fuertemente permitiendo el desarrollo de capacidades específicas en múltiples disciplinas como la aeronáutica, la electrónica, sistemas embebidos, software, mecánica entre otras. Las cuales son disciplinas directamente relacionadas con las aplicaciones en coherencia y el ámbito espacial.

En este contexto es que se desarrolla el Proyecto PIDDEF 01/ESP/15/BAA, el cual plantea como objetivo el desarrollo de una electrónica de a bordo (ECA) orientada a un vehículo aéreo no tripulado. Esta electrónica se compone de dos unidades principales: una unidad de adquisición de sensores de a bordo, y una unidad de control de actuadores y comunicaciones, identificada por las siglas UCC, la cual es el objeto del presente trabajo.

El sistema donde está inmersa la unidad UCC consiste en un enlace de comunicaciones para la bajada de los datos adquiridos y uno de subida de comandos, ambos a bordo de la aeronave, y una estación terrena la cual complementa el mando remoto del VANT. En la Fig. 1 se puede observar un esquema de bloques sintético del sistema.

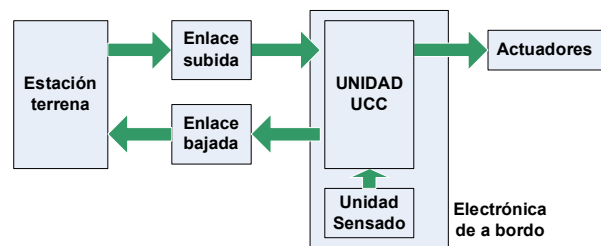


Fig. 1. Esquema general del sistema.

La unidad UCC debe satisfacer como requerimientos funcionales: interpretar los mensajes de comandos enviados desde la estación terrena para generar las acciones sobre los actuadores del VANT, resolver las interfaces de comunicaciones con el enlace de bajada y disponer de interfaces para evaluación de la electrónica en un banco de ensayos del tipo *hardware – in the loop* (HIL). Estos dos últimos aspectos corresponden a la segunda etapa del desarrollo de la unidad, por lo que no son parte de éste trabajo, el cual se enfoca en el desarrollo e implementación de lo que compete a la funcionalidad de recepción y procesamiento de los mensajes de comandos y generación de las señales sobre los actuadores.

Tanto la unidad UCC como las otras unidades del Proyecto deben cumplir con los requerimientos no funcionales tales como: ser concebidas en forma escalable y reutilizable, con la posibilidad de utilizarse sobre otras plataformas, como ser vectores tipo sonda, vehículos terrestres no tripulados, una plataforma nano satelital, entre otros. En términos prácticos este requerimiento implica entre otras cosas que la unidad sea adaptable a los cambios de interfaces físicas de comunicaciones y sus respectivas velocidades, a la cantidad y tipos de actuadores, y a su vez que estos cambios no perjudiquen la performance temporal de la misma. Queda excluido de esta consideración el aspecto de robustez mecánica y de tolerancia a fallas que algunas aplicaciones específicas puedan exigir.

En función de todos los requerimientos se optó por desarrollar a la unidad UCC en un dispositivo de lógica programable, en particular una FPGA (*Field Programmable Gate Array*), por las características de escalabilidad que posee en general, la gran cantidad de entradas-salidas, la

capacidad de cómputo paralelo lo que permite establecer en forma determinística los tiempos de cada proceso interno, y la generación de bloques de características no comercialmente disponibles, en concordancia a lo expresado en [1]. En contraposición, el costo a pagar es el elevado tiempo de desarrollo, el cual suele ser mayor que en el caso de utilizar otro dispositivo programable, como por ejemplo un microcontrolador.

El trabajo se lo estructura tal que en la Sección II se presentará la metodología de trabajo, en la Sección III se presenta el detalle del desarrollo expresado desde la concepción general de la unidad hasta el detalle de los distintos módulos que la componen. En la Sección IV se presenta un resumen de la implementación en hardware y resultados obtenidos.

II. METODOLOGÍA

La carga de trabajo se puede separar en dos grandes tareas la que compete al desarrollo del hardware (placas) y el de la descripción de hardware mediante lenguaje HDL (del inglés: *Hardware Description Languages*). Puesto que la mayor carga de trabajo estaba en esta última actividad se buscó utilizar plataformas de hardware comerciales adaptables a las necesidades para reducir la carga de trabajo en dicha tarea.

La metodología de trabajo para la descripción de hardware consistió en identificar los bloques principales de la unidad, cada bloque fue a su vez dividido en funcionalidades básicas a los que se denominaron módulos y éstos divididos en componentes, los cuales en muchos casos son compartidos por varios bloques.

Cada componente fue desarrollado en lenguaje VHDL, y cada uno posee un banco de prueba (*testbench*) para la evaluación de comportamiento del mismo. Superada esta instancia el componente era vinculado a otros a fin de reunir la funcionalidad de módulo o bloque según corresponda, el cual era evaluado siguiendo la misma metodología. Por otro lado, se maximizó el uso de definiciones genéricas (*generics*), para que cada módulo pueda ser configurable y reutilizable, como se establece en [2].

En otros casos, se realizaron evaluaciones de los módulos desarrollados con ayuda de otros bloques de hardware del proyecto, a fin de verificar funcionalidades parciales.

En particular, en aquellos módulos donde fuera necesario describir una máquina de estados finitos se han adoptado las reglas de diseño y la metodología de trabajo propuesta en [3].

Adicionalmente, se han establecido pautas de proyecto para la codificación y documentación automática de forma tal que los códigos generados, que son parte del entregable del proyecto, sean consistentes independientemente de que sean generados por distintas personas. Para las reglas de codificación en VHDL se adoptaron algunas de las reglas establecidas en [4], en conjunto con algunas reglas propias del laboratorio; y para la documentación automática se utilizó la herramienta Doxygen [5].

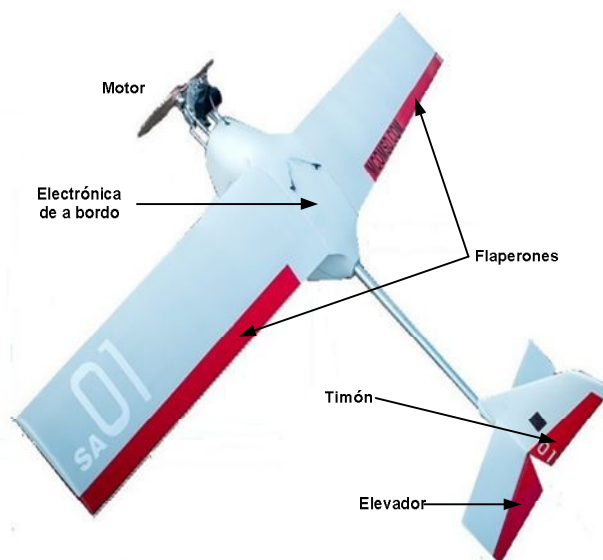


Fig. 2. Aeromodelo del Proyecto.

III. DESARROLLO

A. Introducción

Como plataforma de trabajo y ensayos se utiliza un aeromodelo construido especialmente para el proyecto, el cual albergará a la electrónica de a bordo y por tanto a la unidad UCC.

Como características generales el aeromodelo posee tres metros de envergadura, es propulsado por un motor naftero de dos tiempos y posee cuatro superficies de control: dos flaperones, un timón de profundidad, un elevador. En la Fig. 2 se observa una vista del aeromodelo.

Los actuadores para manejar las superficies de control y el acelerador son servomotores de corriente continua, controlados mediante una modulación de ancho de pulso.

Las referencias para estos actuadores están contenidas en una trama de comandos proveniente de una estación terrena. La trama es recibida a bordo por un receptor de comunicaciones que demodula la información y la presenta a la unidad UCC en un puerto serie asincrónico.

En la Fig. 3 se observa el formato de la trama que es de longitud fija: 20 bytes, posee un encabezado (2 bytes), un contador de secuencia de trama (2 bytes), un *checksum* (1 byte), un identificador de mensajes *ID* (1 byte), y una carga útil (14 bytes) que contiene las referencias para los actuadores. Es por tanto, que la unidad UCC deberá identificar la trama mediante el encabezado, extraer la carga útil, y verificar del *checksum*.

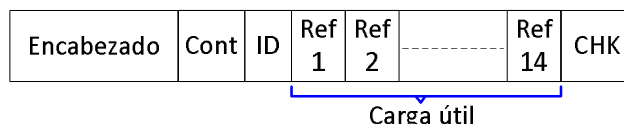


Fig. 3. Trama de información.

B. Placas y bloques principales

La unidad UCC se compone de dos placas. La primera alberga las interfaces de adaptación de niveles de tensión entre la FPGA y los módulos externos (transmisor, receptor y actuadores), e integra las entradas salidas en un conector único. La segunda es la que contiene la FPGA y la circuitería necesaria para su funcionamiento, la cual es una solución comercial. Los detalles del hardware se describirán en la Sección IV.- Implementación.

En la Fig. 4 se presentan los bloques internos a la FPGA. Se identifican los tres bloques principales: un decodificador de mensajes, un extractor con memoria y un controlador de actuación, cada uno de estos con sus módulos internos, los buses de interconexión *Wishbone* [6] son los que se detallan sombreados con un patrón de rayas.

En los siguientes apartados se presentan las características de cada bloque.

C. Decodificador de mensajes

Este bloque es el encargado de recibir los mensajes, con el formato de trama adecuado, proveniente del receptor de comunicaciones. Su función principal consiste en desempaquetar la carga útil contenida en el mensaje, es decir: validar la trama, discriminar el tipo de comando (según el campo de ID) y extraer la carga útil de la misma, la cual es almacenada en una memoria de doble puerto compatible con *Wishbone*, que se comparte con el bloque **Extractor con memoria**.

El módulo **Capa física** tiene como función convertir los datos en formato serie, salientes del receptor de comunicaciones, a un dato paralelo. Consiste en un componente de recepción serie asincrónica que convierte un dato serie asincrónico en un dato paralelo agrupando ocho bits por cada palabra paralela. La velocidad del canal no excede los 20kbp/s.

Una vez disponible el dato en forma paralela, los mismos ingresan al módulo **Decoder** que es el encargado de buscar el sincronismo de la trama a nivel de byte, asegurando que los datos a su salida corresponden a una trama, este bloque presenta una demora en el procesamiento de sólo dos ciclos de reloj.

Los datos salientes del **Decoder** se inyectan en forma secuencial al módulo Extractor de datos el cual valida el *checksum*, discrimina el tipo de comando mediante el campo de identificación del comando (ID) y guarda la carga útil extraída en la memoria compartida. La discriminación en si no es necesaria para el estado actual de la unidad, ya que solo se recibirán comandos de referencia, pero deja listo tanto el módulo en si, como al bloque principal en forma escalable para incorporar distintos tipos de comandos.

El tiempo de procesamiento que posee el **Decoder** y el **Extractor de datos** es tal que la frecuencia de datos de entrada al módulo debe ser ocho veces más lenta que el período del reloj. Para el caso de aplicación puntual con un reloj del sistema de 50MHz, la salida del módulo **capa física** debe ser como máximo de 6,25MHz.

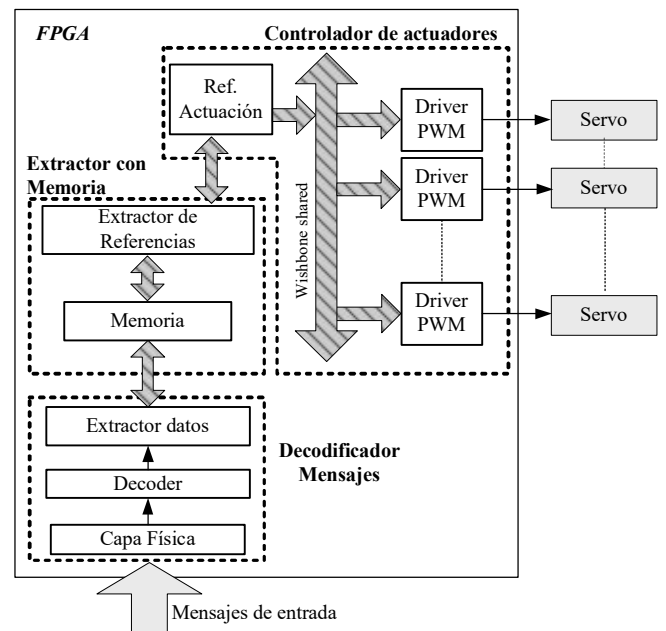


Fig. 4. Bloques principales.

En la Fig. 5 se presenta el proceso de desempaquetado antes descrito para un comando de referencia.

D. Extractor con memoria

Este bloque posee la memoria compartida que posee la carga útil del mensaje. El módulo **Extractor de referencias** que es en esencia un controlador o *master* entre dos periféricos *Wishbone*, es decir entre la memoria y el Controlador de actuadores. El extractor de referencias espera que haya una nueva información cargada en la memoria compartida y que el controlador no esté ocupado para comenzar a transferirle la nueva información. La gran importancia de este bloque es asegurar que la información se actualice adecuadamente, en tiempo, en el control de actuadores.

E. Controlador de actuadores

La función de este bloque es traducir las referencias contenidas en la carga útil del comando en las señales de control adecuadas a los actuadores, según lo establecido en las especificaciones de los mismos [7]. El tipo de señal necesaria es una señal de modulación por ancho de pulso, PWM (por las siglas en inglés). El módulo encargado de generar esta señal es el denominado **Driver PWM**. El cual genera una salida modulada en función de un valor de N bits de entrada. Se debe instanciar un módulo de este tipo para cada actuador a manejar.

El otro módulo que conforma este bloque es el de **Ref.Actuación**, siendo el encargado de transferir a cada driver el valor futuro obtenido del bloque **Extractor con Memoria**.

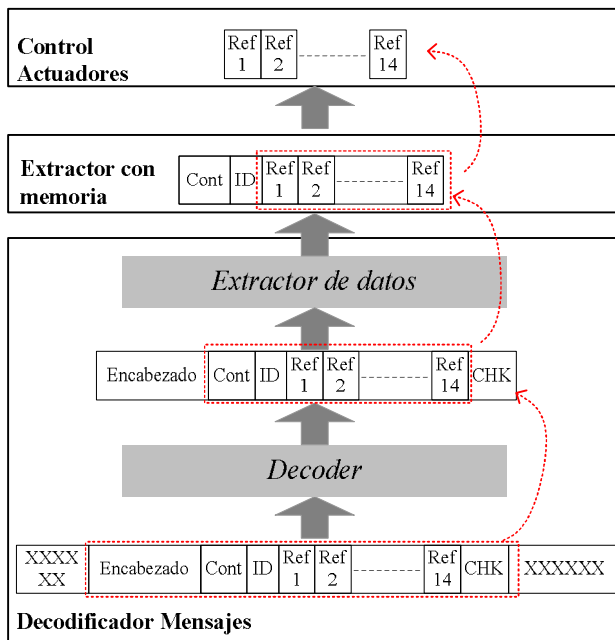


Fig. 5. Proceso de desempaqueo de trama.

La actualización de los datos se da en forma secuencial durante la ejecución de un período de PWM en un registro interno de cada driver, de esta manera el nuevo valor se actualiza al iniciar el próximo período. Este hecho es muy importante ya que garantiza que la orden de referencia de cada actuador se cumple en simultáneo tal lo comunicó la lógica de control (externa a la unidad), mediante el enlace de comunicaciones.

Con el propósito de que el bloque sea lo más genérico posible y simplificar las interconexiones se ha dispuesto un bus del tipo *Wishbone*, en topología *bus shared* para vincular a los bloques periféricos -Driver PWM-, y al maestro -Ref.Actuacion-. De esta manera el bloque es escalable mediante un *generic* en lo que respecta a la instanciación de periféricos del tipo **Driver PWM** y ampliable a otro tipo de drivers con modificaciones mínimas del módulo.

IV. IMPLEMENTACIÓN

A. Plataforma de trabajo

La implementación se realizó sobre una kit de desarrollo de la Empresa Emtech modelo 3PX1 [8], cuya imagen se presenta en la Fig. 6. El mismo posee una FPGA de Xilinx® modelo Spartan 6 6XC6SLX25-2FTG256C.

Se optó por utilizar este kit porque ofrece las características necesarias para el desarrollo y permite la integración con otro hardware por tener mapeados 148 pines de entrada-salida de la FPGA en conectores estándar de 0,1” de paso. También, en el mismo kit se incorpora toda la circuitería mínima necesaria para el funcionamiento de la FPGA. Además posee una dimensión acotada en 93 mm x 83 mm, reducida en comparación con otros kits de desarrollo disponibles en el mercado.

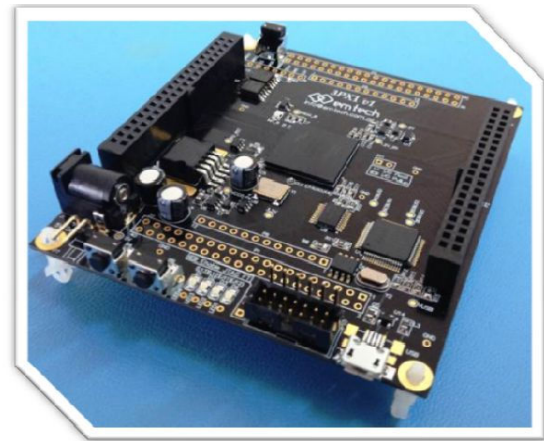


Fig. 6. Placa Emtech 3PX1.

Como se hiciese mención anteriormente para adaptar los niveles de tensión y corriente, y el agregado de conectores tipo DB para el vínculo con otras unidades de a bordo se agregó una placa de interfaces, que se conecta sobre los conectores del kit 3PX1. En la Fig. 7 se pueden observar ambas placas ensambladas, la de interfaces en la parte superior y la del kit FPGA en la parte inferior, las cuales conforman el hardware de la unidad UCC.

B. Recursos utilizados de la FPGA

El consumo de recursos en la FPGA de la solución completa para ocho actuadores es el que se especifica en la Tabla I. La información se presenta en tres columnas, destacando en la segunda columna la cantidad absoluta de recursos utilizadas y en la tercera el porcentaje que esa cantidad representa en función de los recursos disponibles en el dispositivo.

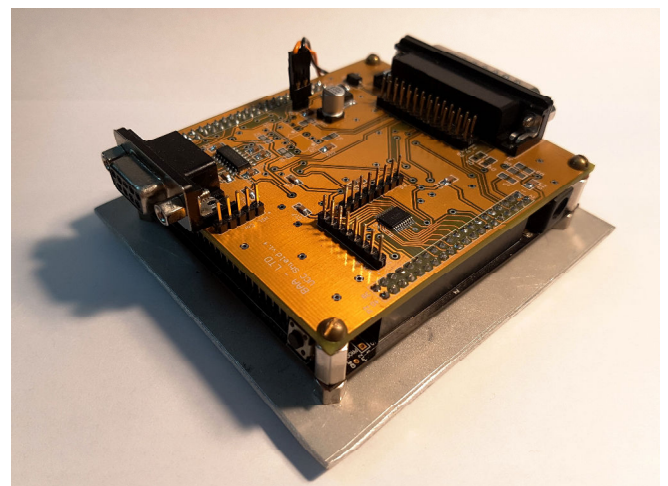


Fig. 7. Hardware de la unidad UCC.

TABLA I. RESULTADOS DE LA SÍNTESIS

Recurso	Cantidad utilizada	Porcentaje utilizado
Slice Register	324	1%
Slice LUT	332	2%
I/O	16	8%
Block RAMs	2	4%

A su vez el resultado de la síntesis informó que el diseño podría utilizarse hasta una frecuencia de reloj de 219MHz, lo cual está muy por encima del valor deseado para la aplicación que es de 50MHz, aprovechando el reloj disponible en el Kit 3PX1.

Se destaca que el agregado de un driver PWM consume aproximadamente 21 slice LUT y 9 slice Register.

C. Escenario de ensayo

Para verificar la unidad se realizaron dos tipos de ensayos.

El primer ensayo tuvo por objetivo verificar las salidas hacia los actuadores, mediante mediciones con un osciloscopio de tiempos de actividad, período de la señal y estabilidad. Para eso se aprovechó una herramienta estadística incorporada en dicho instrumento. Los comandos con las referencias para cada salida se enviaron desde una computadora vinculada a la unidad UCC, desde una aplicación específica. En la Fig. 8 se presenta el escenario de ensayo.

En la Fig. 9 se muestra el resultado de una de las mediciones realizada. Se observan cuatro salidas con distintos valores del ciclo de actividad. Se presentan en la misma imagen los datos estadísticos recolectados por el instrumento, donde se destaca como parámetro de importancia la alta estabilidad en frecuencia de las señales, lo que es un parámetro característico de un dispositivo FPGA.

Los otros ensayos que se realizaron consistieron en enviar comandos erróneos, sea en términos de encabezado, checksum, identificador (ID), entre otros, para analizar que el sistema de procesamiento responda de acuerdo a lo esperado.

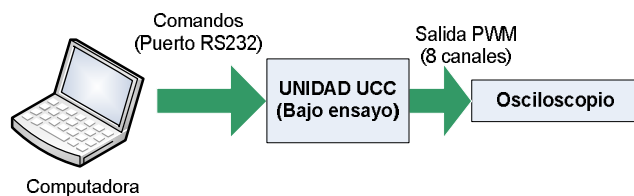


Fig. 8. Esquema de ensayo.

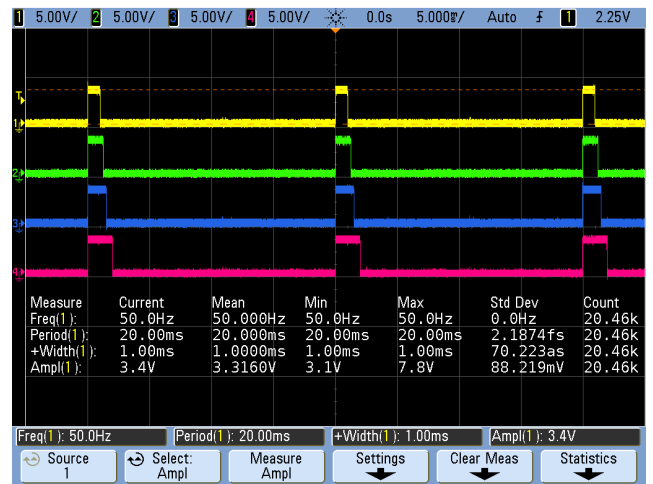


Fig. 9. Medición de señales de salida.

D. Ensayo sobre el VANT

Se realizaron ensayos de validación de toda la unidad conectándola a los actuadores, verificando que se controla adecuadamente en rango a los mismos, en la Fig. 10 se observa un ensayo de máxima excursión realizada sobre el elevador del aeromodelo.



Fig. 10. Ensayo en el elevador.

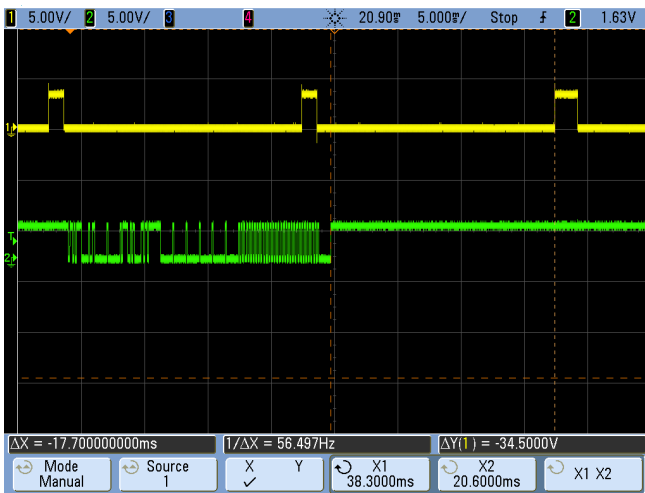


Fig. 11. Demora de actualización.

Otra verificación realizada consistió en evaluar la demora desde que se recibe el comando hasta que se actualiza la salida de los datos. Esto se verificó mediante la medición con osciloscopio, como los resultados presentados en la Fig. 11. En el canal 1 se mide la salida PWM y en el canal 2 se mide la señal serie de entrada de datos. Se envían dos comandos sucesivos de distinto tiempo, la demora se mide desde que se serializa el último comando (Canal 2) hasta que se actualiza la salida, se observa el cambio del ciclo de actividad (Canal 1), notar que la demora de 17,7 ms es inferior a los 20 ms de duración de un PWM.

V. CONCLUSIONES

Se ha logrado desarrollar, implementar y evaluar en laboratorio una unidad basada en FPGA capaz de procesar un mensaje de comandos y generar las señales de actuación del tipo PWM. La unidad es escalable en cantidad y tipo de actuadores a controlar, así como también es adaptable en términos de longitud de trama e interfaz física de comunicaciones.

Por el concepto modular de bloques IP reutilizables permite que cada uno de estos pueda usarse en otros proyectos o desarrollos futuros donde sea necesario satisfacer las funciones específicas que resuelve cada bloque.

De esta manera la unidad no sólo encuentra aplicación en sistemas aéreos no tripulados, por el contrario, la unidad, una parte de ella, o alguno de los módulos podrán utilizarse en aplicaciones como ser: unidades de procesamiento de vuelo en general, en vectores tipo sonda, nano plataformas satelitales, entre otros.

Desde el enfoque de hardware se logró un diseño que utiliza muy poco de los recursos disponibles en la FPGA, lo que deja un margen de ampliación muy grande para futuras expansiones de la unidad.

Por otro lado, el uso de una FPGA, por la característica de paralelismo, permitió que el tiempo de procesamiento pueda ser determinístico, que las señales hacia los actuadores no presenten desplazamientos de fase que afecten al actuador, y los cambios de velocidad de las interfaces de comunicaciones no impacten, en el resto de las bloques.

Finalmente, es importante notar que la aplicación se ejecuta con un reloj de sistema relativamente bajo, e incluso se podría disminuir a la mitad de su valor o un tercio del mismo, y el funcionamiento seguiría siendo apropiado, lo que permitiría que el consumo de la unidad disminuya.

Actualmente se continúa trabajando en la segunda etapa de esta unidad que consiste en el agregado de interfaces de comunicaciones para los vínculos con un banco de ensayos del tipo HIL, y con el enlace de comunicaciones de bajada de datos.

Como trabajo futuro se pretende embeber algoritmos asociados al control de actuadores más complejos que requieran de un controlador de a bordo.

AGRADECIMIENTOS

Se agradece a todo el grupo de trabajo del PIDDEF 01/ESP/15/BAA que ha posibilitado que este desarrollo se concrete.

REFERENCIAS

- [1] A. Anzueto, A. Avila y J. Quiroz, "Control de la marcha de un robot hexápodo autónomo implementado en una FPGA". Libro de memorias Congreso Argentino de Sistemas Embebidos (CASE 2013), 2013, pp. 167-172.
- [2] Pong P. Chu, RTL Hardware Design using VHDL. Coding for Efficiency, Portability, and Scalability. Wiley, 2006, Chapter 13, 14.
- [3] Volnei A. Pedroni, "Finite State Machines in Hardware: Theory and Design (with VHDL and SystemVerilog)". MIT press, primera edición, 2013.
- [4] European Space Agency, "VHDL Modelling Guidelines".2004.
- [5] Doxygen. [Online]
Disponible: www.doxygen.org.
- [6] Silicore, OpenCores, "Wishbone B4, Wishbone System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores", 2010.
- [7] Hi-Tec. Servomotor HS-5485HB.
- [8] Emtech. "Placa 3PX1 - Manual de usuario". 2015.

Repositorios abiertos para desarrollo con FPGAs

Rodrigo A. Melo, Bruno Valinoti
Instituto Nacional de Tecnología Industrial
Centro de Micro y Nanoelectrónica
Email: {rmelo, valinoti}@inti.gob.ar

Resumen—Los repositorios abiertos son una práctica generalizada en los proyectos tecnológicos, en búsqueda de compartir conocimiento y beneficiarse de los aportes de una comunidad. En este trabajo se presentan tres repositorios para el desarrollo con FPGAs, resultado de años en la temática, distribuidos bajo licencias libres: una biblioteca con código HDL, aplicaciones para usar las herramientas en forma independiente del fabricante y ejemplos de cómo utilizar características de distintas placas de desarrollo. Se describen sus características así como también el soporte brindado actualmente y cuáles son las mejoras planificadas para trabajo futuro. Se espera su expansión a través de las tareas cotidianas de los integrantes del grupo de desarrollo y a partir de la realización de Prácticas Profesionales Supervisadas, para las cuáles se consideran aptos, brindando las instalaciones y profesionales del centro.

Index Terms—FPGA, VHDL, Tcl, Python, Software Libre.

I. INTRODUCCIÓN

El uso de sistemas de control de versiones es una práctica ampliamente adoptada y difundida en el desarrollo de *software* y *hardware*. En la actualidad es común que empresas tecnológicas liberen y mantengan algunos de sus trabajos en plataformas con repositorios abiertos. La ventaja potencial es la creación de una comunidad alrededor, que los utilice activamente, reporte errores e incluso aporte mejoras al código. Git [1] es el control de versiones preferido para su implementación y GitHub [2] uno de los principales proveedores del servicio.

El grupo de desarrollo con FPGAs del Centro de Micro y Nanoelectrónica del Bicentenario de INTI, cuenta con años de trayectoria en la temática, durante los cuáles ha acumulado experiencia y creación de herramientas auxiliares.

En este trabajo se presentan los repositorios FPGA Lib [3], FPGA Helpers [4] y FPGA Examples [5]. Los mismos cuentan con recursos para el desarrollo con FPGAs y se distribuyen con licencias GPL 3 [6] o BSD de 3 cláusulas [7], están documentados en idioma Inglés y fueron desarrollados sobre sistemas Debian [8] GNU/Linux.

La estructura de este trabajo es la siguiente: La sección II introduce los tres repositorios. Las secciones III y IV contienen resultados y conclusiones. Finalmente, la sección V presenta el trabajo a futuro planificado.

II. REPOSITORIOS

II-A. FPGA Lib

Con el paso del tiempo y la participación en distintos desarrollos, es común la reutilización de código. A tal fin, es buena práctica la implementación de una biblioteca. FPGA Lib es la colección de *snippets* HDL y *scripts* que se utilizan

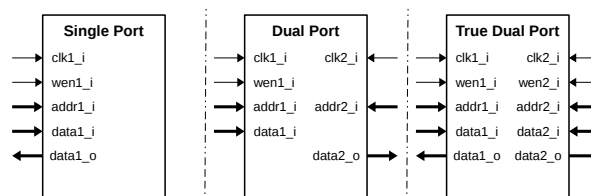


Figura 1. Tipos de memorias soportadas en paquete mems.

en los desarrollos del grupo de trabajo. Está organizada por directorios, a saber:

- **boards:** contiene archivos de *constraints* con asignación de pines para distintas placas de desarrollo.
- **makes:** *Makefiles* útiles.
- **scripts:** pequeñas herramientas escritas en Python.
- **vhdl:** código fuente descrito con este HDL.

La parte principal del repositorio, el código HDL, consta de pequeños bloques, procedimientos y funciones, organizados en paquetes:

- **mems:** descripción de memorias *Simple*, *Dual* y *True Dual Port* (Fig. 1), en forma tal que son inferidas por las herramientas de síntesis.
- **numeric:** compuesto principalmente por funciones de conversión entre enteros y *std_logic_vectors* no cubiertas por los paquetes *std_logic_1164* y *numeric_std* (ver Fig. 2(a) y Fig. 2(b)). Adicionalmente, cuenta con funciones para obtener logaritmo, mínimo y máximo de enteros, las cuáles resultan particularmente útiles para trabajar con valores de *generics*.
- **simul:** código útil para la realización de *testbenches*, como un generador de *clock* y *reset*, funciones de conversión entre *strings* y otros tipos (ver Fig. 2(c)), lectura y escritura de archivos con valores *std_logic* e impresión de mensajes en pantalla.
- **sync:** cadenas de *flip-flops*, divisores de frecuencia de trabajo para generación de habilitaciones, bloques que ayudan con la sincronización entre dominios de reloj y la utilización de *Giga Bit Transceivers*.
- **verif:** bloques útiles a la hora de verificar diseños en *hardware*, como un parpadeo para *leds*, o un generador y comparador de valores para probar datos transmitidos y recibidos.

Uno de los principales lineamientos es desarrollar código independiente de los fabricantes. Así, se prefiere la inferencia

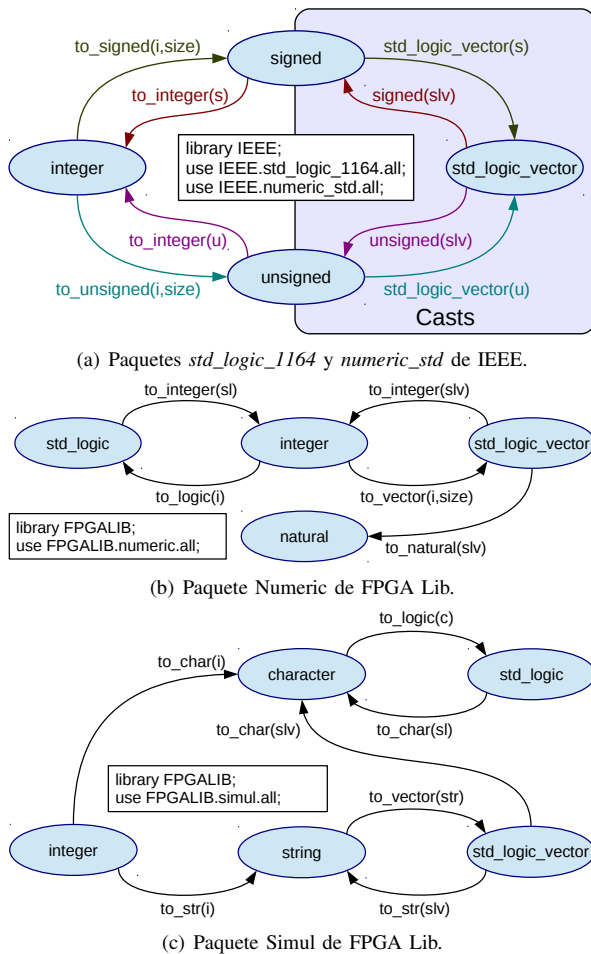


Figura 2. Conversiones entre tipos de datos.

de bloques frente a la instanciación de primitivas. Se trata también en lo posible, evitar la utilización de trucos propios de una herramienta en particular.

Finalmente, se cuenta con utilidades como un *Makefile* para GHDL [9] y un *script* Python para la obtención de números aleatorios, útil para estimular bancos de prueba.

FPGA Lib se distribuye bajo licencia BSD de 3 cláusulas.

II-B. FPGA Helpers

Este repositorio ofrece utilidades que ayudan a utilizar las herramientas de desarrollo para FPGAs en un modo independiente del fabricante. Está dividido en dos partes:

1. Archivos de *scripts Tool Command Language* (TCL) para realizar síntesis y programación, así como también un *Makefile* para facilitar la ejecución de los mismos.
2. Programas escritos en Python y Bash, que ayudan a generar archivos de opciones y ejecutar los *scripts* Tcl en forma efectiva para distintas tareas.

Los objetivos principales son:

- Facilitar la integración con sistemas de control de versiones. Los proyectos que generan las herramientas de los

fabricantes son sumamente complicados de llevar en los mismos.

- No depender de las particularidades de las herramientas. Se modifica un único archivo que presenta siempre los mismos comandos y variables a asignar.
- Obtener reproducibilidad y repetibilidad. No depende de una secuencia de pasos a realizar en una interfaz gráfica.
- Consumir menos recursos del sistema. Esto se obtiene evitando la interfaz gráfica, corriendo en una consola del sistema.

Existe una herramienta desarrollada por el CERN, escrita también en Python, denominada HDLMake [10], que cubre estos objetivos de forma diferente:

- Depende de un archivo de manifiesto en cada directorio involucrado (donde está el *Top Level* y donde haya archivos HDL).
- A partir de dichos archivos, genera un *Makefile* que ejecuta las herramientas desde línea de comandos en lugar de usar el interprete Tcl.
- Un cambio en un archivo de manifiesto, implica volver a ejecutar la herramienta para regenerar el *Makefile*, lo que genera dependencia de disponer de la misma.

Un desarrollo que utiliza FPGA Helpers precisa una única copia de los *scripts* Tcl y luego un archivo de opciones y su *Makefile* asociado por cada proyecto. Dichos archivos no se regeneran, sino que están preparados para ser editados fácilmente, y pasan a formar parte del proyecto, lo que evita dependencia de la herramienta. Preferimos Tcl a ejecución de múltiples comandos, dado que es un lenguaje soportado por las herramientas de los cuatro fabricantes principales de FPGAs, e implica aprender unos pocos comandos nuevos y poder compartir gran parte de código previamente desarrollado.

A continuación se encuentra la descripción de los archivos base compartidos por todo el desarrollo:

- **synthesis.tcl:** realiza la síntesis, implementación y generación de *bitstreams*. Permite seleccionar optimizaciones por área, velocidad y consumo de energía.
- **programming.tcl:** permite transferir *bitstreams* a dispositivos FPGAs. Adicionalmente soporta algunos tipos de memorias.
- **Makefile:** ejecuta los archivos Tcl con el interprete correspondiente según el fabricante seleccionado.

Luego, los archivos particulares para cada proyecto serán:

- **options.tcl:** archivo con opciones, tales como datos de dispositivos y archivos involucrados
- **Makefile:** incluye al *Makefile* base con unas pocas especificaciones de parámetros.

Además de la funcionalidad básica descrita, ofrecen algunas características extras tales como:

- Si el *script* de síntesis encuentra un archivo de proyecto para la herramienta seleccionada, lo utiliza (permitiendo así crearlo y configurarlo con la GUI del fabricante) y en caso contrario lo crea a partir de *options.tcl*.
- En *options.tcl* pueden utilizarse dos funciones para abstraer las tareas típicas de selección de dispositivo y ar-

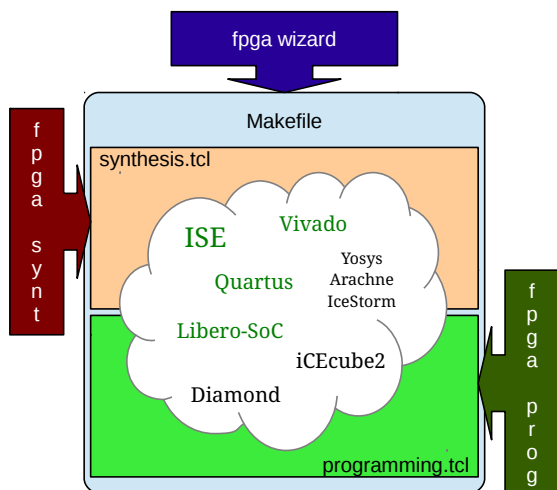


Figura 3. Esquema de relación entre scripts Tcl y Python

chivos del proyecto, evitando así la necesidad de conocer los comandos propios de cada herramienta para tal fin.

- La función para selección de dispositivo posee un parámetro para indicar en que herramienta es válido. Con esto se logra poder sintetizar el mismo proyecto para distintas herramientas sin encontrarnos con el problema de dispositivo no soportado.
- Si hacen falta directivas específicas de una herramienta, se tiene acceso en *options.tcl* a una variable que indica el nombre de la herramienta en ejecución. La misma sirve para evitar, haciendo uso de un condicional, errores de comando no soportado.
- El *Makefile* posee objetivos para abrir la GUI de la herramienta o ejecutar su consola Tcl.

Los programas Python facilitan el trabajo con los *scripts* Tcl:

- **fpga_setup**: el *Makefile* de los archivos Tcl, supone que la herramienta del fabricante se encuentra en el *PATH* del sistema y lista para ser ejecutada. Dicha preparación puede realizarse manualmente, estar automatizadas por ejemplo en el archivo *.bashrc* o puede llevarse a cabo con esta herramienta. La misma sirve para configurar *PATHs* y opciones de licencias, preparar la consola para ejecutar una única herramienta o todas las soportadas. Sólo válida para sistemas GNU/Linux.
- **fpga_wizard**: utilidad que a partir de preguntas interactivas, ayuda a generar el archivo de opciones y su *Makefile*.
- **fpga_synt**: permite sintetizar desde consola un proyecto creado con la herramienta del fabricante.
- **fpga_prog**: permite programar un bitstream del que dispongamos, sin necesidad de crearle un proyecto.

En la Fig. 3 se puede apreciar la relación entre archivos Tcl y que programa Python los utiliza.

La Fig. 4 muestra el flujo típico de trabajo para un proyecto con FPGA Helpers:

- Con *fpga_wizard* se crean los archivos *options.tcl* y *Makefile*.
- Si las herramientas no están agregadas al *PATH* del sistema, se puede utilizar *fpga_setup* para tal fin, necesaria una sola vez por sesión de consola (previamente tuvo que haber sido configurada al menos una vez).
- Mientras no haya cambios del tipo de dispositivo o agregado/eliminación de archivos, alcanza con ejecutar el comando *make*.
- Cuando haya una modificación, alcanzará con editar el archivo *options.tcl*.

FPGA Helpers hace uso de las Autotools [11] para la generación de un *tarball* distribuible. Además, está empaquetado para Debian GNU/Linux. Se distribuye bajo licencia GPL 3.

II-C. FPGA Examples

Cuando nuestro grupo de trabajo adquiere una nueva placa, o al investigar el uso de bloques en *hardware* para utilizar en un desarrollo, es práctica común realizar un ejemplo. Este repositorio los recolecta y ordena. Sirve entre otras cosas para:

- Tener ejemplos que funcionan, listos para probar.
- Aprender a utilizar características de las FPGA.
- Tener un punto de partida para diseños propios.
- Concentrar material de consulta y aprendizaje para nuevos integrantes del grupo.

Incluye a FPGA Helpers como submódulo para realizar la síntesis, implementación y generación de bitstream de forma automatizada. Alternativamente, puede utilizarse la GUI del fabricante para armar un proyecto que incluya los archivos HDL y de *constraints*. Incluye de igual modo a FPGA Lib por la utilización de algunos de sus bloques.

Por cada *kit* soportado, se cuenta con un directorio que incluye:

- Archivo *README* que detalla características de la placa (que contiene, cómo alimentar la placa y preparación del *hardware* para la programación).
- Dado que se trata de evitar la inclusión de código del fabricante, para evitar problemas de licencia, se cuenta con un directorio (opcional) denominado *resources*, que cuenta con lo necesario para su generación.
- Un directorio por cada ejemplo. Se utilizan nombres como : *gpios*, *clock*, *ddr*, etc. Si hay más de un ejemplo, se les agrega sufijos 2, 3, etc.

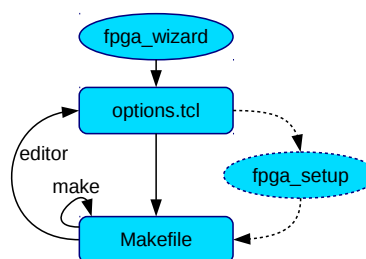


Figura 4. Flujo de desarrollo con FPGA Helpers

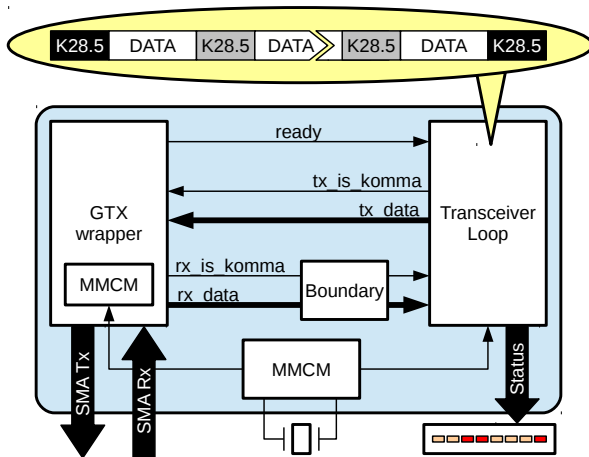


Figura 5. Transceiver

Cada directorio de ejemplo incluye mínimamente:

- Archivo *README* con instrucciones para reproducirlo.
- Archivo *top.vhdl* del proyecto y opcionalmente *wrapper.tcl*, por si hace falta simplificar el uso de un *core* del fabricante.
- Archivo de *constraints* con al menos asignación de pines.
- Archivos *options.tcl* y *Makefile* para utilizar con FPGA Helpers. Si hace falta generar un *core* del fabricante, se debe agregar el objetivo *prepare* en el archivo de *make*.
- Opcionalmente, para ejemplos complejos que lo ameriten, un directorio llamado *testbench* con *scripts* e instrucciones para realizar una simulación.

Hay ejemplos básicos, disponibles para todos los *kits* soportados, que muestran como trabajar con distintas fuentes de reloj y con las entradas/salidas disponibles, como *leds*, *switches* y *dip-switches*. Hay ejemplos avanzados para algunos *kits*, que hacen uso de la DDR, el puerto Ethernet y de algún *Giga Bit Transceiver* (como por ejemplo el de la Fig. 5).

FPGA Examples se distribuye bajo licencia BSD de 3 cláusulas.

III. RESULTADOS

El código VHDL de FPGA Lib está verificado y validado en *hardware*, dado que es utilizado activamente en los desarrollos del laboratorio.

Los *scripts* Tcl y programas de FPGA Helpers, soportan actualmente las herramientas ISE y Vivado de Xilinx, Quartus de Intel/Altera y Libero-SoC de Microsemi. En todos los casos se puede generar *bitstream* y programar la FPGA. En el caso particular de ISE, se puede además programar memorias SPI y BPI.

FPGA Examples provee actualmente 20 ejemplos distribuidos en 11 *kits* de los cuatro proveedores principales:

- **Xilinx:** Spartan-3 Development Kit, Avnet Spartan-6 FPGA LX9 MicroBoard, Spartan-6 FPGA SP601 Evaluation Kit, Virtex-6 FPGA ML605 Evaluation Kit, Digilent Zybo, Digilent Pynq y ZC706 Evaluation Board.

- **Intel/Altera:** DE0-Nano Development Board.
- **Microsemi:** M2S090TS-EVAL-KIT.
- **Lattice:** iCEstick Evaluation Kit y iCE40-HX8K Breakout Board.

IV. CONCLUSIONES

Se liberaron con licencias libres tres repositorios para el desarrollo con FPGAs. Consisten principalmente en código y *scripts* desarrollados y utilizados en los últimos años por nuestro laboratorio. El hecho de liberarlos contribuyó a ordenar los archivos en estructuras de directorios conocidas y a agregar documentación faltante.

Los principales lenguajes utilizados fueron VHDL, Tcl y Python. Se utilizaron *Makefiles* para la automatización de tareas. La principal plataforma de desarrollo fue Debian GNU/Linux.

Los *scripts* Tcl de FPGA Helpers deberían funcionar sobre cualquier SO, dado que dependen de las herramientas de los fabricantes y fueron desarrollados pensando en portabilidad entre las mismas. Así mismo, los *scripts* de Python deberían funcionar si existe el interprete. En el caso de FPGA Examples, no debería haber problemas según el SO, pero en caso de haberlos, alcanza con tomar y utilizar los archivos HDL y de *constraints* en la herramienta del fabricante.

V. TRABAJO FUTURO

Algunas de las características que consideramos implementar en FPGA Lib, son:

- Ampliar la oferta de bloques, procedimientos y funciones HDL.
- Agregar versiones en Verilog y soporte para iVerilog.

Para FPGA Helpers, se planificó:

- Agregar soporte en *scripts* Tcl a herramientas de Lattice Semiconductor.
- Agregar soporte a flujo de trabajo con herramientas Libres: Yosys [12], Arachne [13] y IceStorm [14].
- Desarrollar *script* que recolecte automáticamente los archivos del proyecto a partir de un *Top Level* indicado.
- Agregar interfaz gráfica simple y liviana, que facilite la creación interactiva del archivo de opciones y ejecute acciones asociadas.

En el futuro de FPGA Examples habrá:

- Ejemplos para más placas de desarrollo.
- Ejemplos para otros bloques o características disponibles.
- Ejemplos más avanzados dentro de los *kits* ya incluidos.

Cabe señalar que los tres repositorios son utilizados activamente por el grupo de desarrollo y se espera que evolucionen paulatinamente, de manera sostenida, durante varios años. Adicionalmente, están abiertos a contribuciones y son aptos para la realización de la Práctica Profesional Supervisada (PPS) del Sistema Educativo Universitario Argentino, lo cuál contribuiría a su crecimiento.

REFERENCIAS

- [1] L. Torvalds, J. Hamano *et al.* Git. [Online]. Available: <https://git-scm.com>
- [2] Github. [Online]. Available: <https://github.com>
- [3] INTI CMNB. FPGA Lib. [Online]. Available: https://github.com/INTI-CMNB-FPGA/fpga_lib
- [4] ——. FPGA Helpers. [Online]. Available: https://github.com/INTI-CMNB-FPGA/fpga_helpers
- [5] ——. FPGA Examples. [Online]. Available: https://github.com/INTI-CMNB-FPGA/fpga_examples
- [6] Free Software Foundation, Inc. GNU GENERAL PUBLIC LICENSE version 3. [Online]. Available: <https://www.gnu.org/licenses/gpl-3.0.en.html>
- [7] University of California. The 3-Clause BSD License. [Online]. Available: <https://opensource.org/licenses/BSD-3-Clause>
- [8] Debian Project. Debian: the universal operating system. [Online]. Available: <http://www.debian.org>
- [9] T. Gingold. Where VHDL meets gcc. [Online]. Available: <http://ghdl.free.fr/>
- [10] CERN. Hdlmake. [Online]. Available: <http://www.ohwr.org/projects/hdl-make>
- [11] GNU Project. The autotools. [Online]. Available: https://www.gnu.org/software/automake/manual/html_node/GNU-Build-System.html
- [12] C. Wolf. Yosys Open Synthesis suite. [Online]. Available: <http://www.clifford.at/yosys>
- [13] Arachne: Place and route tool for FPGAs. [Online]. Available: <https://github.com/cseed/arachne-pnr>
- [14] C. Wolf. Project IceStorm. [Online]. Available: <http://www.clifford.at/icestorm/>

Desarrollo e Implementación de Generador y Codificador NRZ-L de tramas PCM utilizando System Generator de Xilinx

Adrián Stacul
CITEDEF

Laboratorio de Técnicas Digitales
Buenos Aires, Argentina
Email: astacul@citedef.gob.ar

Cristian Bruña
CITEDEF

Departamento de Electrónica
Buenos Aires, Argentina
Email: cbruna@citedef.gob.ar

Resumen—El presente trabajo describe el diseño, desarrollo e implementación en FPGA (en inglés - *Field Programmable Gate Array*) de un Simulador de tramas PCM (en inglés - *Pulse Code Modulation*) utilizando la herramienta System Generator de Xilinx. Este módulo tiene como finalidad verificar sistemas de adquisición de datos telemétricos. El empleo de esta tecnología brinda flexibilidad para facilitar la adaptación del flujo PCM a diversos esquemas. Esta plataforma fue desarrollada para ser utilizada en ensayos de sistemas y subsistemas en el marco de programas de vectores sonda y vehículos aéreos no tripulados del Instituto de Investigaciones Científicas y Técnicas para la Defensa (CITEDEF).

I. INTRODUCCIÓN

En sistemas aéreos no tripulados y en vectores tipo sonda se hace indispensable disponer de una estación terrena para la recepción y procesamiento de datos en tiempo real, tanto sea para el control y monitoreo de la misión como para la evaluación de los diferentes experimentos científicos instalados a bordo del vehículo. Un simulador PCM puede ser diseñado e implementado utilizando microprocesadores, micro-controladores o dispositivos reconfigurables como las FPGAs [1]. En general, los simuladores comerciales poseen licencia (llave en mano) y los códigos no son accesibles por el usuario, sino que los fabricantes limitan la licencia dependiendo su uso. Este es el motivo por el cual fue desarrollado un generador/simulador que tuviera la capacidad de reconfiguración para cada campo de trama [2].

En la búsqueda de la flexibilidad enunciada para la generación de datos PCM simulados, el entorno MATLAB/Simulink [3] permite crear y simular el diseño mediante bloques funcionales. En este contexto se puede hacer uso de una librería desarrollada por Xilinx que se integra en Simulink denominada System Generator [4]. Se busca de esta forma obtener un simulador de bajo costo que permita generar tramas con información de alta velocidad de diferentes magnitudes físicas, empleando diferentes métodos para la sincronización de encabezados de tramas y la deconmutación de datos en el sistema de adquisición en tierra responsable del procesamiento de la información [5].

Se parte de trama PCM de longitud y tasa de muestreo fijas, datos conocidos y codificación digital NRZ-L basado en el estándar IRIG106 – Capítulo 4 - Clase I [6], con la particularidad que el usuario puede adaptar la trama a los requerimientos de su proyecto modificando las constantes.

II. DISEÑO DEL MODELO

Con el fin de producir un modelo general válido para diferentes configuraciones (tanto en velocidades de transmisión como cadencia de datos) el diseño desarrollado posee 6 módulos principales que utilizan únicamente bloques de Xilinx (ver Fig. 1).

II-A. Sample Time Generator

Es un generador de pulsos que asegura una cadencia de datos definida, es decir, produce un pulso que define el tiempo que existe entre una trama y la siguiente. Este tiempo es constante, ya que el objetivo de este trabajo es reproducir una cadencia de transmisión fija. Se implementó utilizando un contador de 16 bits y un comparador.

II-B. Frame Generator

El generador de tramas que realiza una multiplexación de constantes que representan los 32 bytes de la trama y se almacenan en una memoria FIFO del tipo *Shift-Register*.

II-C. White Noise Adder

Uno de los aspectos de la confiabilidad de un adquisidor de datos telemétricos PCM está dada por su capacidad de reconstruir digitalmente la trama y realizar su sincronización a nivel de bit. Por ello se añadió al simulador/generador la capacidad de incorporar errores a la señal generada, habiendo contemplado un bloque que añade opcionalmente ruido blanco a la señal digital basado en una combinación del algoritmo de Box-Muller y el teorema central del límite [7]).

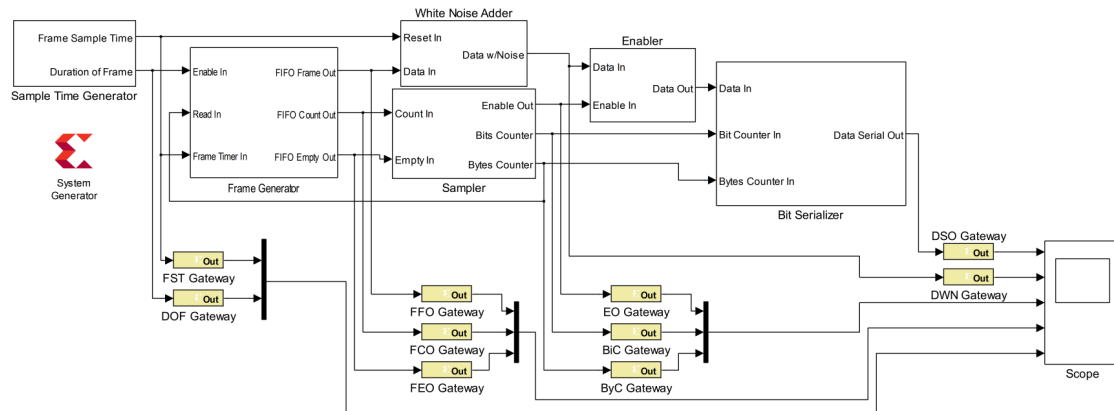


Figura 1. Modelo Completo del Sistema Generador y Codificador NRZ-L de tramas PCM

II-D. Sampler

Genera una señal mediante lógica combinacional y un contador, cuyo flanco de subida corresponde a una muestra de cada bit. Un segundo contador y comparador brindan otra señal de salida con flanco ascendente correspondiente a cada byte almacenado en memoria FIFO.

II-E. Enabler y Bit Serializer

El subsistema Enabler habilita la salida de datos por medio de un contador y un comparador, la cual está en estado bajo o cero durante la no transmisión por ser codificación NRZ-L. El Bit Serializer extrae cada uno de los 8 bits que componen el byte a transmitir utilizando máscaras. Luego por medio de una multiplexación en tiempo, se logra la serialización de la trama PCM. Dicha línea digital es la salida final de datos del modelo completo con codificación NRZ-L.

El modelo completo fue grabado en un kit de desarrollo modelo Atlys [8] de la firma Digilent que posee una FPGA Spartan-6 de Xilinx y pudo comprobarse a través de mediciones con osciloscopio que las salidas fueron correctas.

III. CONCLUSIONES

El modelo completo del sistema Generador/Simulador de tramas PCM con codificación NRZ obtuvo como resultado una cadencia de tramas de $100\mu s$ y anchos de bit de $100ns$, es decir, tramas de $100Khz @ 10Mbps$, en este caso se utilizó a modo de ejemplo el número 0503 en hexadecimal como palabra de sincronismo. Si bien este sistema es reconfigurable y el usuario puede optar por elegir su palabra de sincronismo que desee, el estándar de telemetría recomienda utilizar EB90 en hexadecimal para este tipo de aplicaciones (código de 16 bits) ya que minimiza la probabilidad de un “enganche” erróneo en el circuito de detección de sincronismo de un adquisidor de datos [9]. Actualmente el programa de Vectores Sonda de CITEDEF emplean una trama telemétrica de $1Mbps$, por lo cual esta plataforma generadora cumple con los requisitos del proyecto apropiadamente.

Como trabajo a futuro, el incremento de velocidades de procesamiento para tramas telemétricas ayudaría a obtener

resultados en tierra más confiables, mejor aprovechamiento de ancho de banda, mayor capacidad de ejercer acciones de control en tiempo real, que luego se traducirán en la mejora sustancial de los mecanismos de procesamiento de sistemas de tierra de alta performance. El esquema propuesto puede integrarse con aplicaciones que suplanten la información constante contenida en cada canal telemétrico, de forma tal que puedan emplearse simuladores de sensores abordables en vehículos aéreos o espaciales para ensayos aún más realistas que permitan evaluar sistemas y subsistemas en forma previa a la realización de la misión.

AGRADECIMIENTOS

Los autores agradecen especialmente a los Ing. Edgardo Comas y Daniel Pastafiglia del Departamento de Electrónica Aplicada por haber prestado su apoyo y colaboración institucional. También a los equipos de trabajo de los Laboratorios de Técnicas Digitales y de Navegación y Control de CITEDEF.

REFERENCIAS

- [1] R. Dorf. (2000) Telemetry. the engineering handbook.
- [2] E. Comas, A. Stacul, C. Bruña *et al.* (2015) Diseño, desarrollo e implementación de una estación terrena para cohetes sonda. ISBN 978-987-3806-24-7.
- [3] (2016) Simulink. Mathworks Inc. [Online]. Available: <http://www.mathworks.com/products/simulink/>
- [4] *System Generator for DSP - User Guide*, Xilinx, 2009. [Online]. Available: http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/sysgen_user.pdf
- [5] (2000) Radio frequency and modulation systems—part 1: Earth stations and spacecraft. Consultative Committee for Space Data Systems. National Aeronautics and Space Administration Washington, DC 20546, USA: CCSDS Secretariat Program Integration Division. [Online]. Available: <https://public.ccsds.org/Pubs/401x0b25.pdf>
- [6] R. C. C. Telemetry Group, “Pulse code modulation standards. telemetry standards, irig standard 106-13 (part 1)(chapter 4),” Secretariat, Range Commanders Council, White Sands Missile Range, New Mexico, 88002, Tech. Rep., 2013.
- [7] A. Ghazel, E. Boutillon, J.-L. Danger, G. Gulak, and H. Laamari, “Design and performances analysis of high speed awgn communication channel emulator,” 2001, iEEE PACRIM Conference.
- [8] (2016) Kit de desarrollo atlys. Digilent Inc. [Online]. Available: https://reference.digilentinc.com/_media/atlys:atlys:atlys_rm.pdf
- [9] S. Horan, Ed., *Telemetry. The Engineering Handbook*. Richard C. Dorf, 2000.

Bit-Synchronizer Implementation in a Low-Cost FPGA for UAV

Adrian Stacul

Institute of Scientific and Technical Research for Defense (CITEDEF)
Laboratory of Digital Techniques
Buenos Aires, Argentina
Email: astacul@citedef.gob.ar

Felipe Diniello

National Technological University (UTN)
Electronic Department
Buenos Aires, Argentina
Email: fdiniello@frba.utn.edu.ar

Abstract—The main purpose of this paper is the development of PCM bit-synchronizer in HDL. The entire system will be applied to a ground station with an ad-hoc telemetric data acquisition system to be applied in UAV monitoring and sounding rockets. Based on this information, the ground station will compute the platform trajectories, velocities and attitudes. In particular, this PCM module was built to be used in atmospheric sounding vector evaluations by the Institute of Scientific and Technical Research for Defense of Argentina (CITEDEF).

Index Terms—PCM Frame, Bit-synchronizer, FPGA, System Generator, Ground Station.

I. INTRODUCTION

Both unmanned aerial systems and sounding rockets require a ground station for the acquisition of telemetric signals and real time data processing[1] whether for the control and monitoring of the mission or for the evaluation of the different scientific experiments installed on the platform. The design of an acquisition system in a ground station is a complex task since it involves receiving data and sending it to the processing systems so that everything operates in real time. At the same time, on-board electronic systems are increasingly faster and easily adaptable to the requirements of the experiment. As a consequence, the data acquisition system changes constantly with every redesign of the platform. The aim of this work is to obtain a low-cost data acquisition system that allows the reception of high-speed PCM frames to decommute all of the channels with the physical magnitudes within the PCM frames. The module developed draws on the progress of different methods for the synchronization of frame headers and data decommuting in the ground acquisition system, which will perform the information processing task in real time.

II. DESIGN

The reconstruction of the telemetric data is divided into several stages. In the real system, the data received are limited in bandwidth and have additional noise. This poses a challenge since, in order to be able to discern between a logical '0' or '1', a proper criterion must be taken to avoid misinterpreting these values and altering the processing chain. In order to achieve PCM signal regeneration, it is necessary to observe the permanent regime state of the signal. The level (zero or one) at the midpoint of each bit is maintained despite the noise.

The re-synchronization principle of operation is based on a shift register of a larger size than that of the word to be found, and with a locally generated clock at a higher frequency than that of the signal to be detected.

The register has an interlaced parallel output, in which each bit is separated from the next by N registers. From this output, the first occurrence of the sync-word is expected to be found to activate the serial data output and a synchronized and regenerated clock.

Once this first event occurs, N/2 clock pulses must be expected to ensure that each oversampled bit is positioned in the middle of the current bit. Then, the serial output is already synchronized, and the output clock must be regenerated with an N times division of the internal clock. The advantage of working with an increasing multiplier is that it allows to find the center point of each bit more accurately, minimizing the probability of error.

There is no synchronicity between the input signal and the sampling frequency (the clock frequency multiplied by N) so it is important to take the medium sample as the most representative of the whole bit. This ensures that the transient periods of the signal have already ended. Oversampling reduces the error of finding such a mid-point, but excessive oversampling increases the resources of the FPGA unnecessarily.

III. DEVELOPMENT

Working in Simulink[2] with System Generator[3] is different from working with VHDL or Verilog directly[4], since Simulinks development methodology consists in interconnecting blocks graphically. When working with the Xilinx System Generator suite, all logical blocks of simulation and synthesis are available in the Simulink environment[5].

The development of the acquisition system was separated into three fundamental blocks (Fig. 1), each with a specific function within the processing chain. In this way, each module could be evaluated separately while adding the possibility of updating some of its parts in future developments.

A. Bit-Sync

This block has the telemetry signal as input, which consists of only serialized PCM data flow (note that in the acquiring system the PCM clock does not enter). The block is in charge

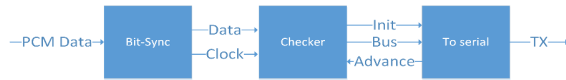


Fig. 1. Diagram in blocks of the acquisition system

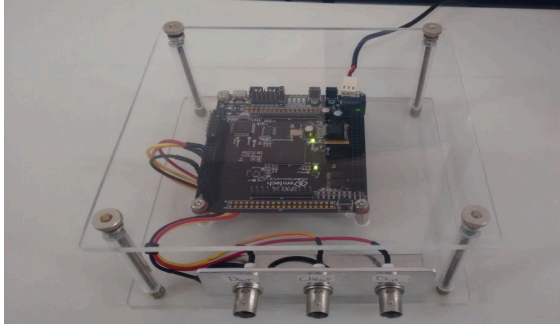


Fig. 2. Implementation in 3PX1 FPGA kit

of regenerating the data in such a way that its output is composed of the same PCM data and a synchronized clock signal. To this purpose, first, the sync word must be detected.

B. Checker

The checker block is an aggregate that is not included in the IRIG 106-13[6] standard specifications, but its function is necessary to detect errors that may occur in the transmission of the PCM frame. In addition, it is the block in charge of rearming the serial data into bytes. It has the synchronous signals as inputs (data and clock) and 8-bit buses as outputs.

Validation of the received data must be performed by corroborating the error detection code, which was calculated as the XOR of all data in the frame, excluding the sync word.

C. To serial

The function of this block is to transmit the data received to the ground-station computers through a USB serial interface. In order to do this, it has an 8 bit parallel data entry and two control signals, one that initiates the transmission and another to request the next data to be sent.

IV. IMPLEMENTATION

The implementation was carried out on a 3PX1[7] development kit manufactured by Emtech S.A.. It has a XC6SLX25 FPGA of the Xilinx Spartan 6 family [8]. One of the advantages of choosing this module over others available in the market, is that it is a low-cost segment and it has a serial/USB interface already connected directly to the FPGA. The card was mounted on an open cabinet made of acrylic to give it greater rigidity, which also had BNC connectors for connection and disconnection, without putting the FPGA kit at risk as shown in Fig.2.

V. CONCLUSION

The final result of our system can vary depending on the frame format as we use. For this example we used a 64-byte

frame that includes a 2-byte sync word. We have made use of a design methodology based on a high level environment, the System Generator with Simulink, for fast prototyping in FPGA. The methodology based on Simulink makes the design process more intuitive than those based on HDL, since it allows us to abstract from the hardware peculiarities. On the other hand, it also reduces the time of the design process, obtaining quickly a first model to be implemented in the FPGA.

Currently the CITEDEF sounding rockets program uses a 1Mbps telemetry frame, which is why this platform meets the project requirements properly. In the last decade, research on the subject of sounding vectors and UAV has enjoyed exponential growth in several disciplines: aeronautical systems, applied mechanics, on-board electronics, ground stations, real-time signal processing, data links, etc.

In this case, the introduction of signal processing methods on ground stations to study navigation algorithms has required increasingly complex and dynamic acquisition systems. As a future work, increasing telemetry frame rates would help to obtain more reliable results, which will then translate into substantial improvement in decommutation and processing mechanisms for the development of highly specific ground systems.

ACKNOWLEDGMENTS

The present R&D work was carried out under the supervision of my PhD thesis director, PhD Mario Lavorato, and the Head of the Applied Electronics Department, Eng Edgardo Comas, to whom I would like to express my deepest appreciation for making this study possible. In addition, I would also like to thank all the personnel working at the Laboratory of Digital Techniques in CITEDEF, who permanently collaborate in the development of software and hardware for this type of applications, especially to trainee engineer Felipe Diniello who worked on the implementation of the system. Finally, I'm very grateful to the CITEDEF authorities for the logistical support and to MINDEF (Department of Defense), which provides financial support to this type of programs and projects.

REFERENCES

- [1] Consultative Committee for Space Data Systems. (2000) Radio frequency and modulation systems - part 1: Earth stations and spacecraft. [Online]. Available: <https://public.ccsds.org/Pubs/401x0b25.pdf>
- [2] MathWorks Inc. (2016) Simulink. [Online]. Available: <http://www.mathworks.com/products/simulink/>
- [3] Xilinx. (2016) System generator for dsp user guide. [Online]. Available: http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/sysgen_user.pdf
- [4] P. Chu, *FPGA prototyping by Verilog examples*. Wiley, ISBN: 978-0-470-18532-2.
- [5] M. R. Valido *et al.*, "Metodologia de diseo en fpga usando xilinx system generator," 2012.
- [6] R.C.C. Telemetry Group, *Pulse code modulation standards. Telemetry Standards - IRIG standard 106-13 - part 1 - chapter 4*. Telemetry Group, 2013.
- [7] Emtech. Development kit 3px1. [Online]. Available: <http://www.emtech.com.ar/producto/placa-3px1>
- [8] Xilinx. Spartan-6 family overview. [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds160.pdf

Bloque generador de señales PWM basado en FPGA con utilización de Wishbone

Diseño, desarrollo e implementación

Ariel Dalmas Di Giovanni¹, Emiliano H. Prato¹, Daniel A. Pastafiglia¹
Laboratorio de Técnicas Digitales - Departamento de Electrónica Aplicada
Instituto de Investigaciones Científicas y Técnicas para la Defensa (CITEDEF)
Villa Martelli, Buenos Aires, Argentina.
¹{adigiovanni, eprato, dpastafiglia}@citedef.gob.ar

I. INTRODUCCIÓN

La modulación por ancho de pulso (PWM) es una técnica de modulación digital típica de los sistemas de control de servomotores [1]. Actualmente en el Laboratorio de Técnicas Digitales de CITEDEF se está trabajando en el desarrollo de electrónica de a bordo para vehículos aéreos no tripulados basada en FPGA. En particular, el vehículo del proyecto es un aeromodelo de ala alta cuyos actuadores se basan fundamentalmente en el tipo de servomotores descrito en [2]. En este contexto surge la necesidad genuina de desarrollar un bloque capaz de manejar múltiples módulos controladores de actuadores del tipo PWM, garantizando seguridad y robustez en ejecución.

El presente trabajo presenta el diseño, desarrollo e implementación de una solución adaptable en lógica programable.

II. ARQUITECTURA

El bloque posee una cantidad definida de salidas cada una asociada a un actuador del tipo PWM. Para mantener compatibilidad de interconexión con otros bloques de la electrónica de a bordo, se dispone de un puerto *Wishbone* [3] que permite acceder a una memoria de doble puerto interna y dos líneas de control de flujo para validar los datos contenidos en la memoria.

En la Fig. 1 se presenta un esquema de bloques de la arquitectura, se destacan los puertos antes mencionados, y un bus *Wishbone* del tipo *shared*. En éste se conectan como periféricos (*slaves*) los módulos PWM y la memoria interna que se desempeñan como *slaves*, el rol de *master* lo desempeña el módulo **wb_ref_actuacion**.

La señal de modulación requerida debe poseer un período de 20 ms, con un ciclo de actividad definido desde 1ms a 2 ms. En términos de resolución esta situación garantiza un paso mínimo de 1° por cada 10µs. Se puede deducir mediante la ecuación (1) que para cuantificar con $M = 8$ bits el ciclo de actividad no podrá exceder los 2560 µs, lo que está dentro de las especificaciones del actuador.

$$\delta = Res \cdot 2^M = 10 \mu s \cdot 2^8 = 2560 \mu s \quad (1)$$

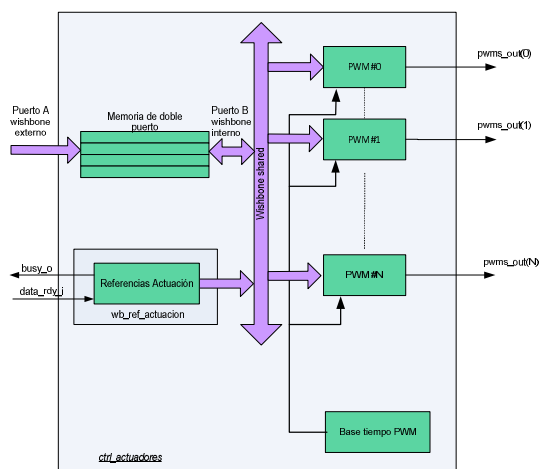


Fig. 1. Detalle de la arquitectura

El bloque que se ocupa de resolver la generación de tiempo es el denominado **base tiempo_pwm**, que dispone de tres salidas principales que se conectan a los distintos módulos PWM, como se muestra en la Fig. 2. La salida **ventana_disparo_o** habilita una ventana de tiempo durante el ciclo de actividad máximo, que se reinicia cada 20 ms. La salida **tc_pulsos_o** genera un pulso de período coincidente con el tiempo máximo de ciclo de actividad, y la salida **salida_pulsos_o** es el valor instantáneo del contador del tiempo de actividad, que cuenta de 0 a 255 pulsos.

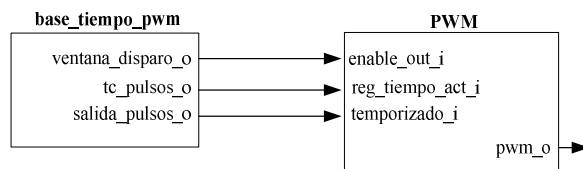


Fig. 2. Conexión base de tiempo a módulo PWM.

El módulo PWM posee una estructura basada en un comparador que define en estado activo la salida si la señal de un contador externo, provisto por **base_tiempo_pwm**, no alcanzó el valor de referencia, y en estado inactivo cuando la señal supera dicho valor. Este módulo además garantiza que se cumpla el periodo completo (20ms) de la señal, a pesar que se haya actualizado la referencia en el medio de un periodo. En la Fig. 3 se muestra un esquema resumido de la arquitectura del módulo PWM antes descrito.

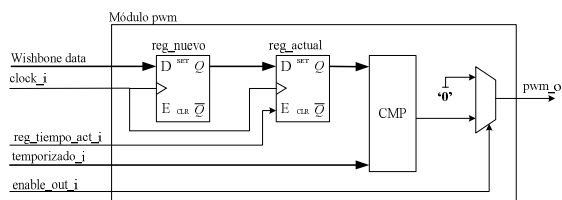


Fig. 3. Arquitectura interna del módulo PWM.

El dato proveniente del contador de 0 a 255 se ingresa al comparador mediante el puerto **temporizado_i**, que en si define el intervalo de ciclo activo máximo, superado el máximo valor la señal **tc_pulsos_o** pulsa y genera que se actualice el valor **reg_actual** con el valor de **reg_nuevo**.

III. IMPLEMENTACIÓN, ENSAYO Y VALIDACIÓN

El bloque fue completamente desarrollado en lenguaje de descripción de hardware, específicamente VHDL. Si bien se trabajó con una FPGA Spartan 6 de Xilinx®, y el entorno de desarrollo integrado ISE, se buscó que la descripción no quede sesgada a un dispositivo o fabricante particular, es decir, que pueda implementarse en dispositivos de otros fabricantes. Además, se maximizó el uso de *generics* [4] para hacer escalable los distintos bloques, y se siguió el flujo de diseño de máquinas de estado detallado en [5].

Como plataforma de trabajo se utilizó una placa 3PX1 de la Empresa Emtech modelo 3PX1 [6]. La misma posee una FPGA de Xilinx® modelo Spartan 6 6XC6SLX25-2FTG256C [7].

En la Fig. 4 se observa un esquema del banco de ensayos. En módulo **wb_tester** es un bloque adicional que actúa como fuente de datos para ensayar el **ctrl_actuadores**. Desde una PC se generan las referencias para cada canal y se mide con osciloscopio en las salidas **pwmN_o** para verificar que el valor de referencia enviado se impactó en la salida correspondiente.

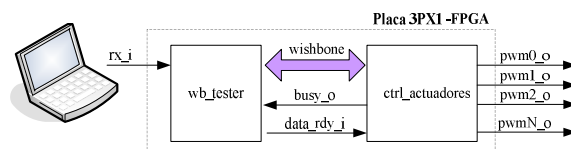


Fig. 4. Banco de ensayos del módulo.

IV. RESULTADOS

La implementación del bloque se realizó para 8 (ocho) PWMs de salida, periodo de PWM de 20 ms, ciclo de actividad máximo 2.56 ms, 8 (ocho) bits de ancho de bus de datos y frecuencia de reloj principal 50MHz. Los resultados de la síntesis son los expuestos en la Tabla I, y el diseño es apto hasta los 250 MHz de reloj principal.

TABLA I. RESULTADOS DE LA SÍNTESIS

Recurso	Cantidad utilizada	Porcentaje utilizado
<i>Slice Register</i>	180	< 1%
<i>Slice LUT</i>	245	< 2%
<i>I/O</i>	37	19%
<i>Block RAMs</i>	1	< 1%

V. CONCLUSIONES Y COMENTARIOS FINALES

Los resultados de las simulaciones y mediciones sobre la plataforma de *hardware* han sido satisfactorios, observando que las señales respetan los requerimientos iniciales de diseño. El diseño es adaptable con mínimas modificaciones para usar otros tipos de controladores siempre y cuando sean periféricos *Wishbone* compatibles.

El uso de *Wishbone* como bus preferido de interconexión fue sumamente importante para disminuir los tiempos de desarrollo, tanto de diseño, como de ensayos de comportamiento.

Entre las tareas detectadas como trabajos futuros se destacan la incorporación de otros controladores como por ejemplo del tipo PPM.

REFERENCIAS

- [1] Dora María Ballesteros, "Modulación PWM en FPGA basado en maquinas de estado finitos", Scientia et Technica Año XIV, No 38, Junio de 2008.
- [2] Hi-Tec. Servomotor HS-5485HB.
- [3] Silicore, OpenCores, "Wishbone B4, Wishbone System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores", 2010.
- [4] Pong P. Chu. "RTL Hardware Design using VHDL. Coding for Efficiency, Portability, and Scalability". Wiley, 2006, Chapter 13, 14.
- [5] Volnei A. Pedroni, "Circuit Design and Simulation with VHDL". MIT press, segunda edición, 2010, pp. 241-271.
- [6] Emtech. "Placa 3PX1- Manual de usuario". 2015.
- [7] Xilinx. "Spartan 6 - documentation", 2010. [Online].

Foro Tecnológico

Resumen

RTOS y Software Embebido

Port del Firmware CIAA para plataformas basadas en FPGA con softcore LEON 3

Gerardo L. Puga
Facultad de Ingeniería
Universidad Nacional de La Plata
La Plata, Buenos Aires, Argentina
Email: gerardo.puga@ing.unlp.edu.ar

Resumen—El presente trabajo aborda el desarrollo de una capa de portabilidad para el Firmware CIAA que permite ejecutar aplicaciones basadas en este último sobre sistemas de lógica programable FPGA que incluyan el softcore de código abierto LEON.

Este desarrollo amplía la base de plataformas de hardware del proyecto CIAA, permitiendo ejecutar aplicaciones basadas en la API del firmware oficial del proyecto sobre cualquier kit de desarrollo en FPGA que cuente con recursos suficientes para sintetizar un sistema mínimo basado en el procesador LEON 3.

I. INTRODUCCIÓN

El Proyecto de la Computadora Industrial Abierta Argentina (Proyecto CIAA) [1] es un esfuerzo conjunto llevado adelante por un numeroso conjunto de entidades pertenecientes a las esferas tanto públicas como privadas de la República Argentina, el cual tiene por objeto impulsar la modernización de los procesos productivos de la industria nacional al crear una serie de recursos tecnológicos de amplia disponibilidad, alta calidad y bajo costo que puedan ser utilizados como base para el desarrollo de productos competitivos con las ofertas externas.

Uno de los resultados más importantes de este esfuerzo es la Computadora Industrial Abierta Argentina (CIAA), la cual es una plataforma electrónica programable con calidad de grado industrial y diseño abierto. Esta plataforma se encuentra formada por un sistema de cómputo basado en un microprocesador de alto rendimiento y por una serie de interfaces de entrada/salida estándar que pueden interactuar de forma directa con todo tipo de actuadores y sensores industriales, como relés, interruptores, luces, fines de carrera, etc.

Por su finalidad industrial la CIAA fue diseñada utilizando técnicas y componentes de alta calidad que incrementan su confiabilidad cuando se la utiliza en ambientes agresivos para la electrónica, como los que se encuentran frecuentemente en las plantas de producción: ruido eléctrico, polvo, vibraciones, humedad, etc. Esta característica inmediatamente separa la CIAA de otras plataformas de cómputo como las Raspberry Pi, Arduino, etc., que no se encuentran preparadas para funcionar en este tipo de ambientes.

Otro hito del Proyecto CIAA fue la creación de la plataforma llamada Edu-CIAA, o CIAA Educativa. Ésta es una plataforma funcionalmente muy semejante a la CIAA, pero cuyo diseño se encuentra orientado hacia la utilización como

herramienta educativa en secundarios técnicos e instituciones terciarias, y como plataforma de entrenamiento de bajo costo en los cursos de programación de la CIAA industrial.

Muy temprano en el proyecto CIAA se tomó la decisión de crear múltiples variantes de cada computadora, basadas cada una de ellas en los microcontroladores de diversos fabricantes (NXP, Freescale, Microchip, entre otros) para eliminar de esta forma el riesgo de generar dependencia dentro del proyecto de los productos de un fabricante particular. Así es que existe una CIAA-NXP, una CIAA-Freescale, y varios tipos de Edu-CIAA.

Para evitar que esta multiplicidad de diseños de hardware fragmentara el conjunto de usuarios y simplificar la portabilidad de las aplicaciones entre las variantes de las computadoras CIAA es que se proyectó la creación de una interfaz de programación común a todas ellas. Esta interfaz se denominó Firmware CIAA [2], y es uno de los pilares del Proyecto CIAA. Este software unifica todas las versiones de hardware bajo una misma interfaz de programación que incluye un sistema operativo de tiempo real (FreeOSEK) y una serie de controladores para dispositivos entrada/salida con interfaces tipo POSIX, entre otros elementos.

El presente trabajo amplía el abanico de plataformas de hardware que es posible programar a través del Firmware CIAA al habilitar su utilización sobre plataformas basadas en sistemas de lógica programable de tipo FPGA. Esta variante presenta una flexibilidad que no comparten otras plataformas CIAA, las cuales tienen rígidamente definidas desde el momento de su diseño todas sus características finales: una plataforma de cómputo basada en lógica programable puede ser reconfigurada para cada proyecto individual, adecuando las características del sistema de procesamiento y en cierta medida de los dispositivos de entrada/salida disponibles a aquello que demanden las circunstancias.

Esta flexibilidad inherente hace que este tipo de sistemas tenga excelentes condiciones como herramienta para el prototipado rápido de sistemas, el desarrollo de sistemas con interfaces especializadas, y la creación de bancos de prueba para otros dispositivos, entre otras aplicaciones.

La amplia disponibilidad de kits de desarrollo para FPGA comerciales que pueden ser utilizados como plataforma de hardware para esta variante del Firmware CIAA le otorga un enorme campo de acción sin ni siquiera requerir la exis-

tencia de una versión de referencia de hardware provista desde el mismo Proyecto CIAA. La riqueza de interfaces de entrada/salida incluidas en los kits FPGA (interruptores, displays, video, audio, conversores A/D y D/A, PS/2, USB, puertos Ethernet, por dar una lista de los más frecuentes), ya probadas y funcionales, permite que en muchos casos todas las necesidades de entrada/salida de un diseño puedan ser satisfechas simplemente seleccionando un kit adecuado.

En este trabajo el procesador del sistema que ejecuta el Firmware CIAA es un System-on-Chip (SoC) sintetizado dentro de la FPGA y basado en el procesador LEON 3. Este último es un procesador RISC de 32 bits basado en la arquitectura SPARC V8 y cuyo código fuente se encuentra disponible en lenguaje de descripción de hardware VHDL bajo una licencia GPL. Es un procesador desarrollado por la firma Cobham Gaisler AB que es principalmente utilizado en aplicaciones de tipo aeroespacial asociado a FPGAs tolerantes a radiación, pero cuya flexibilidad, disponibilidad de herramientas y licencia GPL lo hacen atractivo para cualquier tipo de sistemas de procesamiento en FPGA.

El presente trabajo consiste entonces en la implementación de una capa de compatibilidad que permita la ejecución de aplicaciones basadas en el Firmware CIAA sobre sistemas de procesamiento LEON 3 en circuitos de lógica programable FPGA. El alcance del trabajo fue definido con el criterio de enfocar el desarrollo en aquellos elementos críticos necesarios para poner el sistema en marcha y validarlo: desarrollo de rutinas de bajo nivel del sistema operativo (inicialización, gestión de interrupciones, gestión de contextos de tarea), integración de la arquitectura al sistema de compilación y generación del sistema operativo, adaptación del banco de ensayos funcionales del sistema operativo, y desarrollo de los controladores del temporizador y de la consola serial.

La organización de lo que resta del trabajo es la siguiente. En la Sección II se describen las características de la plataforma, tanto desde la perspectiva del hardware subyacente, como de las interfaces de software del Firmware a las que prestará servicio el código desarrollado. La Sección III describe de forma resumida las características más sobresalientes del diseño realizado. La validación de port del Firmware CIAA se discute en la Sección IV, describiendo primeramente las plataformas de prueba utilizadas y luego la metodología y resultados obtenidos. Por último la Sección V cierra el presente trabajo con un resumen de los logros.

Este artículo se encuentra basado en un trabajo de fin de carrera de la Carrera de Especialización en Sistemas Embebidos de la Universidad de Buenos Aires, el cual puede ser consultado en la referencias bibliográfica [3].

II. PLATAFORMA

II-A. Firmware CIAA

El Firmware CIAA es una pieza de software no monolítica, dividida en una serie de módulos que agrupan sub-interfaces del sistema. Además de código propiamente dicho, el firmware está formado también por un sistema de compilación, carga y depuración de las aplicaciones, un banco de ensayos

funcionales para validación del sistema operativo y un sistema de generación estática del sistema operativo.

El firmware incluye un sistema operativo de tiempo real (FreeOSEK) que implementa el estándar de la industria automotriz OSEK [4]. Este es un sistema operativo estático, de muy bajo consumo de recursos, y que provee un conjunto de funcionalidades mínimas necesarias para la implementación de un sistema multitarea [5]. Este sistema operativo requiere de una etapa de generación previa a la compilación, durante la cual se lee la configuración de la aplicación y se la utiliza para generar de forma automática una serie de archivos de código de sistema operativo donde se encuentran declarados de forma estática todos los recursos materiales requeridos (espacios de memoria, bloques de control de dispositivos, bloques de control de tareas, etc.).

Para simplificar el acceso a los principales dispositivos de entrada/salida el firmware cuenta con una interfaz de acceso a los mismos con estilo POSIX que permite utilizar las llamadas *open*, *read*, *write*, *ioctl* y *close* sobre una serie de dispositivos virtuales que representan a las interfaces de hardware (puertos digitales, conversores, puertos seriales, etc.). Esto provee una interfaz portable a través de los diferentes modelos de hardware de la CIAA y libera al usuario de la programación de las interfaces más comunes.

El sistema de compilación permite la inclusión o exclusión de módulos en función de las necesidades de cada proyecto individual, pudiendo por lo tanto desarrollar aplicaciones tanto de tipo *bare-metal* como con sistema operativo OSEK con/sin interfaz POSIX.

A los fines del desarrollo del presente trabajo son particularmente relevantes los módulos del firmware denominados *rtos* (que contiene el sistema operativo FreeOSEK), *drivers* (controladores de bajo nivel de acceso a dispositivos), *posix* (interfaz tipo POSIX para acceder a los dispositivos a través de archivos de sistema) y *base* (rutinas de soporte de la arquitectura). También fue necesario realizar extensiones a los mecanismos de configuración de proyectos, y de generación y validación del sistema operativo FreeOSEK.

II-B. Sistemas basados en LEON 3

El procesador LEON 3 es un softcore que implementa el estándar IEEE-1754 de definición de la arquitectura SPARC versión 8 (SPARC V8) [6]. Es un procesador de tipo RISC de 32 bits, con unidad de punto flotante opcional, instrucciones de división y multiplicación por hardware opcionales, pipeline de 7 etapas y una eficiencia de aproximadamente 1,4 DMIPS/MHz. El procesador puede ser configurado en tiempo de diseño para modificar sus características (tamaño de caches, políticas de reemplazo, tipo de multiplicadores por hardware, presencia de la FPU, predictor de saltos, entre otros parámetros) dándole al diseñador la posibilidad de encontrar la relación de compromiso más adecuada entre la cantidad de recursos lógicos utilizados y potencia de cómputo del procesador.

El procesador LEON 3 forma parte de una biblioteca de núcleos IP más extensa denominada GRLIB [7], la cual

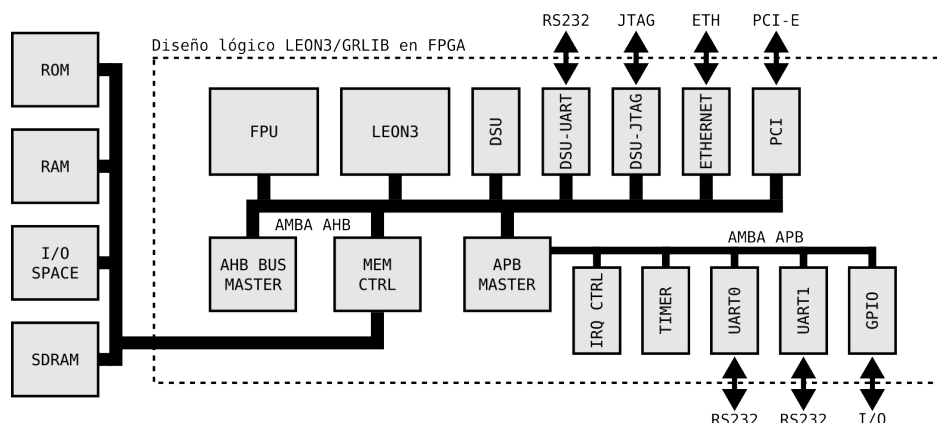


Figura 1. Sistema LEON 3 típico, compuesto por el procesador y por otros núcleos de la biblioteca GRLIB.

Cuadro I

LISTADO DE MÓDULOS ESENCIALES PARA EJECUTAR EL FIRMWARE CIAA SOBRE UN SISTEMA LEON 3.

Módulo	Función
LEON3	Procesador LEON3 que rige el sistema.
MCTRL	Controlador de memoria. Permite mapear una memoria Flash/ROM, una RAM y un rango de direcciones de IO en el mapa de memoria del procesador.
GPTIMER	Módulo temporizador de propósito general, utilizado para activar periódicamente el sistema operativo y permitirle medir el transcurso de intervalos de tiempo. El sistema debe contar con al menos un módulo GPTIMER, que contenga al menos un temporizador independiente.
IRQMP	Controlador de interrupciones del sistema. Utilizado para activar y desactivar interrupciones, así como también para simularlas durante el proceso de validación del sistema operativo OSEK.
DSU3	Unidad de soporte de depuración. Permite cargar y ejecutar programas en memoria, programar la memoria Flash/ROM, examinar el contenido de la RAM, depurar dispositivos, configurar breakpoints, etc. Debe estar acompañada de al menos una de las interfaces externas de la DSU: AHBUART, AHBJTAG, etc.

conforma una colección de núcleos IP compatibles entre sí que puede ser utilizada en el diseño de Systems-on-Chip alrededor de un bus central de tipo AMBA. Además del procesador LEON 3, la biblioteca también se encuentra formada por controladores de memoria, controladores de interrupciones, unidades de punto flotante, módulos de comunicación (UARTs, USB, SPI, I2C, Ethernet, etc.), gestores DMA, entre muchos otros módulos. Todos los módulos de la GRLIB se encuentran descritos en el lenguaje de descripción de hardware VHDL.

Tanto el procesador LEON 3 como la biblioteca GRLIB son sostenidos por la firma Cobham Gaisler AB. La mayor parte de los módulos se encuentran disponibles bajo una licencia de código abierto de tipo GPL v2 que permite la reutilización y modificación del código. Solamente los módulos más avanzados, como por ejemplo las versiones tolerantes a fallas del procesador o los controladores SpaceWire para aplicaciones

aeroespaciales, se encuentran cubiertos por licencias comerciales que deben ser adquiridas individualmente a través de Cobham Gaisler AB. La lista completa de los módulos que conforman la GRLIB así como de las licencias que aplican a cada uno puede ser consultada en [8].

En la Fig. 1 puede verse un ejemplo de sistema típico creado utilizando la GRLIB. En esta figura puede verse el bus central de tipo AMBA que coordina los demás integrantes del sistema. El bus tiene dos partes: un bus complejo de alta velocidad AHB donde se conectan los núcleos IP centrales que demandan grandes cantidades de ancho de banda y bajas latencias, y uno o más buses APB de baja velocidad para dispositivos periféricos (UARTs, I2C, etc.). Los elementos más importantes del sistema son el procesador LEON 3, el controlador de interrupciones, el controlador de memoria, el temporizador y uno o más canales de comunicación. El sistema cuenta también con una unidad de depuración (DSU, Debug Support Unit); esta última se complementa con una o más interfaces externas que permiten acceder al módulo de depuración a través de un canal de comunicación exterior (USB, UART, Ethernet, etc.).

El flujo de trabajo utilizado con la biblioteca GRLIB consiste en diseñar un sistema interconectando los núcleos IP necesarios, y configurar los parámetros de estos últimos de acuerdo a los requerimientos del sistema (por ejemplo la cantidad de UARTs, tamaño de los FIFOs, presencia o ausencia de unidad de punto flotante, etc.). Una vez definido el sistema, éste es sintetizado, sus puertos de entrada salida mapeados a pines, y finalmente la configuración resultante es programada en la memoria de la FPGA destino.

La DSU forma parte del sistema de depuración del procesador, el cual se completa externamente mediante un programa denominado GRMON. El GRMON es un software que corre en una computadora de escritorio y permite controlar las operaciones de carga y ejecución de programas en el procesador. El canal de comunicación entre la DSU y GRMON puede ser una conexión de tipo serial RS232, USB, JTAG, Ethernet u otra. GRMON puede utilizarse de forma independiente, siendo capaz de cargar programas en el sistema, darles inicio,

establecer *breakpoints*, programar las memorias Flash del sistema, etc., o también puede utilizarse para generar una pasarela entre el programa de depuración GNU GDB y el hardware. Esta última modalidad es la utilizada en el presente proyecto, por ser compatible con el esquema de carga de programas y depuración utilizado por el Firmware CIAA con otras arquitecturas (Cortex-M, por ejemplo).

Con el fin de satisfacer los objetivos del presente trabajo se determinó el conjunto mínimo de módulos de la biblioteca GRLIB que es necesario incorporar al diseño del SoC LEON 3 sintetizado en la FPGA es el que aparece en el Cuadro I, donde se discute la función de cada uno dentro del sistema.

III. DESARROLLO REALIZADO

El diseño de la capa de compatibilidad del firmware para el procesador LEON 3 consistió en el desarrollo de las rutinas de bajo nivel del sistema operativo (inicialización, gestión de interrupciones, gestión de contextos de tarea) y de los controladores del temporizador del sistema y de la consola serial, además de la integración de la nueva arquitectura al sistema de compilación del firmware.

En las sub-secciones siguientes se resumen las características más relevantes de los principales elementos desarrollados. Una cobertura más detallada de estos temas puede hallarse en la Memoria de Especialización en la que se basa este trabajo [3].

III-A. Inicialización

A diferencia de lo que sucede con los microcontroladores utilizados en el CIAA Industrial y la Edu-CIAA, los sistemas LEON 3 no tienen una estructura fija sino que sus características son dependientes de los parámetros de configuración seleccionados para cada proyecto particular. Esto significa que la cantidad y tipo de los dispositivos presentes en cada sistema debe considerarse variable, y que incluso la distribución de estos en mapa de memoria debe considerarse inicialmente desconocida durante la inicialización del sistema. Por esta razón para simplificar el desarrollo de software de sistema la GRLIB define un mecanismo denominado de *Plug-and-Play* (PnP) que permite detectar de forma automática la cantidad, tipo y posición en el mapa de memoria de todos los dispositivos que forman parte del sistema.

Utilizando el mecanismo de PnP el sistema operativo determina durante el arranque la posición y características del controlador de interrupciones y el temporizador del sistema, y los configura para permitir la administración de las interrupciones externas y la medición de intervalos de tiempo. La frecuencia del reloj de sistema es también inicialmente incierta, pero puede determinarse automáticamente sabiendo que el *bootloader* del sistema siempre deja configurado el *preescaler* del primer temporizador del sistema para que para que tenga una frecuencia de salida de 1 MHz

Al ser el LEON 3 un procesador de arquitectura SPARC V8 el sistema operativo debe asumir el control de la gestión de las ventanas de registros. Las ventanas de registros son un sofisticado mecanismo de pila utilizado por los procesadores

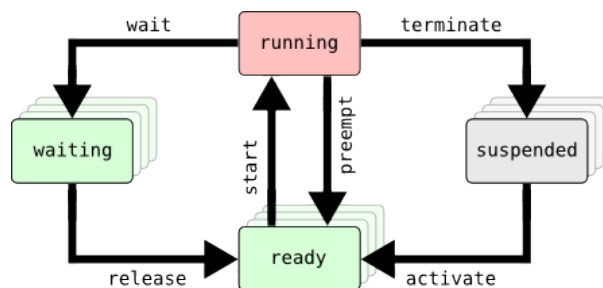


Figura 2. Estados de ejecución de las tareas en OSEK.

de tipo SPARC para reducir la cantidad de accesos a memoria durante los llamados a subrutinas y la atención a rutinas de servicio de interrupción.

Otros eventos relevantes llevados a cabo durante la inicialización son el reemplazo de la tabla de interrupciones del procesador por una que se encuentra administrada por la capa de compatibilidad del firmware, la inicialización de la caches del procesador y la inicialización de los contextos iniciales de todas las tareas de la aplicación de usuario.

III-B. Gestión de Interrupciones en FreeOSEK

El sistema operativo OSEK define dos tipos de interrupciones con habilitación independiente, denominadas categorías ISR1 e ISR2.

Los mecanismos de enmascaramiento provistos por el procesador LEON 3 solamente permiten bloquear interrupciones enmascarando todas las excepciones del sistema (internas y externas) o enmascarando las interrupciones externas según un criterio basado en los niveles de prioridad de ellas (fijos).

Estos mecanismos son suficientes para proteger segmentos de código cuya ejecución deba llevarse a cabo de forma atómica, pero no permiten gestionar las interrupciones de dos categorías ISR1 e ISR2 de forma separada.

Para lograr esta funcionalidad es que se utiliza un nivel de enmascaramiento adicional a través del controlador de interrupciones IRQMP de la GRLIB. Este último es un controlador de interrupciones que permite habilitar y deshabilitar cualquiera de las interrupciones de forma individual. Durante la etapa de generación del sistema operativo se determinan a partir de la configuración de la aplicación dos máscaras de bits que enumeren las interrupciones en cada categoría, y luego estas máscaras se utilizan para habilitar o deshabilitar grupos de interrupciones en el controlador IRQMP.

III-C. Cambios de Contexto en FreeOSEK

La arquitectura de procesadores SPARC V8 no define un mecanismo para la implementación de los cambios de contexto en sistemas multitarea, como si lo hacen por ejemplo los procesadores Cortex-M mediante la interrupción dedicada PendSV. Para implementar esta funcionalidad se debió recurrir a una solución basada exclusivamente en software y que se encuentra integrada con la gestión de interrupciones.

El sistema reconoce dos modos de funcionamiento, Contexto de Usuario y Contexto de Interrupción. El primero es el

modo definido por la ejecución de código de la aplicación de usuario de forma ordenada y secuencial, e incluyendo la ejecución de código de sistema operativo como consecuencia de llamadas a servicios. En este modo de operación la aplicación multitarea se encuentra ejecutando una o múltiples tareas de forma concurrente, pudiendo a lo sumo (pero no necesariamente) encontrarse una de ellas en posesión del recurso procesador (estado de ejecución *running* del sistema operativo OSEK, ver Fig. 2), mientras que las demás se encuentran repartidas entre los estados *ready*, *waiting* y *suspended*.

El Contexto de Interrupción es la ejecución continuada de código de rutinas de servicio de interrupción como consecuencia de una interrupción externa o interrupción por software. Esto puede deberse a la ejecución de una única rutina de servicio de interrupción, o a la ejecución de una colección de ellas de forma anidada. La relación entre contexto de usuario y de interrupción puede verse en la Fig. 3.

Debido al mecanismo de gestión de ventanas de registros de los procesadores SPARC el software de sistema debe gestionar los ingresos y egresos en Contexto de Interrupción (y entre niveles de interrupciones) envolviendo la rutina de interrupción de usuario en un marco que garantice que exista al menos una ventana libre al entrar y por lo menos una ventana ocupada al salir de la rutina de servicio. Para implementar los mecanismos de cambio de contexto en el port del Firmware CIAA se extendió la funcionalidad de este código de forma de tal que almacene el contexto de la tarea que se encuentra en modo *running* en el momento de ingreso en Contexto de Interrupción, y restaure el mismo durante el regreso a Contexto de Usuario. Debe señalarse que la identidad de la tarea en *running* al entrar y al salir puede o no ser la misma en ambos casos, dependiendo de que durante la ejecución del código de interrupción se produzca algún evento que provoque una modificación en este sentido (mediante el envío de eventos a tareas, el vencimiento de alarmas, etc.).

El mecanismo anterior cubre todos los casos de cambio de contexto que son consecuencia directa de la ejecución de una interrupción externa, pero existen otros casos en los que el cambio se debe a la ejecución de una llamada directa al sistema operativo por parte de la aplicación. Ejemplos de esto son la activación y la terminación de una tarea, el envío de eventos entre tareas, etc. Para implementar estos casos las interfaces correspondientes fueron implementadas usando interrupciones por software, quedando de esa forma automáticamente abarcadas por el mecanismo descripto previamente.

III-D. Controlador Serial

Para permitir un mecanismo de entrada/salida flexible durante el proceso de depuración y uso del port LEON 3 del Firmware CIAA se implementó un controlador de UART en el módulo *drivers*. Durante la inicialización del módulo este controlador autodetecta el tipo y la cantidad de UARTs presentes en el sistema usando el mecanismo de auto-descubrimiento de hardware *Plug-and-Play* de la GRLIB.

En función de la cantidad y tipo del hardware presente se registran con el sistema operativo una serie de dispositivos con

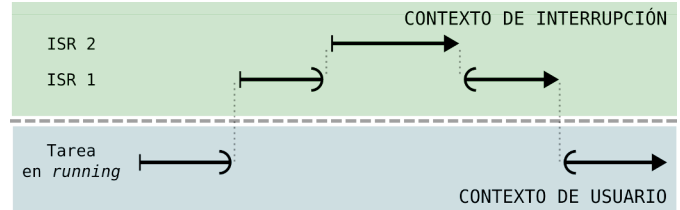


Figura 3. Relación de Contexto de Interrupción con el Contexto de Usuario, representando una transición de modo con interrupciones anidada y regreso al modo original.

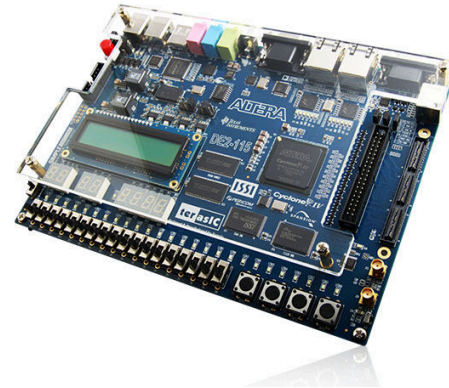


Figura 4. Kit de desarrollo DE2-115.

las correspondientes rutinas de callback necesarias para que la capa POSIX del módulo homónimo pueda acceder a ellos para realizar las operaciones *open*, *read*, *write*, *ioctl* y *close*.

IV. RESULTADOS

IV-A. Modelos de prueba

El port LEON 3 del Firmware CIAA fue ensayado utilizando dos modelos de hardware diferentes. El primero de ellos es un sistema LEON 3 sintetizado en la FPGA Cyclone IV de un kit de desarrollo Terasic DE2-115. Ver Fig. 4. El sistema mínimo implementado en esta FPGA consiste de un único procesador LEON 3, acompañado de los núcleos IP mínimos necesarios para ejecutar el port del Firmware (listados anteriormente en el Cuadro I), además de núcleos APBUART que implementan UARTs de comunicación externa. La frecuencia del sistema está provista por un reloj de 50 MHz.

El segundo modelo de ensayo se realizó utilizando el simulador TSIM comercializado por Cobham Gaisler AB. TSIM permite simular sistemas completos centrados alrededor de procesadores LEON 3, incluyendo no solamente el procesador sino también todos los módulos auxiliares (buses, caches, controladores de memoria, de interrupciones, UARTs, etc.). Al igual que ocurre con la configuración de un sistema LEON previo a la síntesis, todos los núcleos IP simulados se encuentran parametrizados para modificar sus características funcionales. La simulación es muy precisa, reflejando ciclo a ciclo la evolución del sistema en un hardware físico equivalente al simulado.

Al igual que GRMON, TSIM se presenta como un *target* remoto para GDB que permite la carga y ejecución de programas, lo que lo hace perfectamente intercambiable con un sistema físico accedido mediante el primero.

Utilizando este simulador se generó un sistema LEON 3 virtual en el cual se ejecutó el port del Firmware CIAA. Este sistema simulado tiene la ventaja de poder ser modificado con facilidad mediante parámetros que permiten introducir variantes como el tipo y la cantidad de las UARTs presentes y la frecuencia base del reloj del sistema. Esto fue extremadamente útil durante las fases de puesta en marcha y depuración del código.

IV-B. Validación

La parte más compleja de la comprobación del funcionamiento de la capa de compatibilidad es la validación del funcionamiento del sistema operativo sobre la nueva plataforma. Afortunadamente el Firmware CIAA provee una infraestructura que permite la ejecución semi-automática de una batería de más de doscientos ensayos individuales que evalúan el funcionamiento de todos los mecanismos internos del sistema operativo FreeOSEK (módulo *rtos*). Los casos de prueba incluyen la intervención de interrupciones y múltiples formas de cambio de contexto entre tareas, así como también una cantidad muy grande de variantes en la ejecución de decenas de programas de ensayo, lo que ejercita los mecanismos de gestión de ventanas de registros. Una ejecución exitosa de esta batería de ensayos completa significa por lo tanto un nivel de confianza muy elevado sobre el diseño de los algoritmos de bajo nivel del sistema operativo.

Una vez adaptada esta infraestructura para automatizar la ejecución de la batería de ensayos en la plataforma LEON 3 se ejecutó la misma sobre los dos modelos de prueba disponibles. La flexibilidad del sistema simulado en TSIM para definir el sistema simulado permitió ejecutar la batería múltiples veces haciendo variar en cada caso la frecuencia del sistema. La finalidad de esto último es la de modificar las relaciones temporales entre los eventos simulados y por lo tanto ampliar la cobertura de los ensayos realizados. Un resumen de los resultados de ejecutar la batería de ensayos en cada uno de los modelos de prueba puede encontrarse en el Cuadro II.

La verificación de la base de tiempo del temporizador del sistema operativo (parte del módulo *rtos*) y del controlador UART (módulo *drivers*) debió realizarse manualmente de forma separada. Para eso se diseñaron una serie de aplicaciones de prueba que ejercitaban diversos casos de uso del código a verificar, y se verificó en cada caso que su funcionamiento se encontrase en todo momento dentro de los parámetros de diseño del sistema.

Todos los ensayos realizados fueron satisfactorios comprobando de esta forma el correcto funcionamiento del código desarrollado para la capa de portabilidad así como también el de la estructuras auxiliares (generación de sistema operativo, compilación, carga de programas, ejecución de batería de ensayos, etc.).

Cuadro II
RESULTADOS DE LA VALIDACIÓN DEL MÓDULO *rtos*.

Plataforma	Frecuencia	Resultado
DE2-115	50 MHz	satisfactorio
TSIM	5 MHz	satisfactorio
TSIM	10 MHz	satisfactorio
TSIM	20 MHz	satisfactorio
TSIM	30 MHz	satisfactorio
TSIM	40 MHz	satisfactorio
TSIM	50 MHz	satisfactorio
TSIM	75 MHz	satisfactorio
TSIM	100 MHz	satisfactorio
TSIM	150 MHz	satisfactorio
TSIM	200 MHz	satisfactorio

V. CONCLUSIÓN

En este artículo se presentó el desarrollo de un port que permite ejecutar aplicaciones basadas en el Firmware CIAA en sistemas LEON 3 sintetizados dentro de circuitos de lógica programable de tipo FPGA. Este tipo de sistemas presenta una flexibilidad que la hace ideal como plataforma para prototipado y desarrollo de sistemas de procesamiento especializado.

La ejecución del trabajo consistió en el desarrollo de una capa de compatibilidad que contiene el código de bajo nivel necesario para ejecutar el sistema operativo FreeOSEK sobre el procesador LEON 3, y de un controlador de UART que permite utilizar dicho canal de comunicación a través de las interfaces tipo POSIX del firmware. Se implementaron también los mecanismos de generación del sistema operativo, compilación, depuración, carga de programa y validación automática para incorporar la nueva arquitectura.

El Firmware CIAA portado fue validado utilizando para ello una combinación de ensayos estandarizados y otros diseñados ad-hoc individualmente. Los primeros fueron utilizados para validar el funcionamiento del módulo *rtos*, mientras que los segundos se utilizaron para verificar el correcto desempeño del temporizador de sistema y del controlador de la UART en el módulo *drivers*. En todos los casos el comportamiento de port del Firmware CIAA para LEON 3 fue satisfactorio.

REFERENCIAS

- [1] Proyecto CIAA, "Computadora Industrial Abierta Argentina," 2014, disponible: 2016-06-25. [Online]. Available: <http://www.proyecto-ciaa.com.ar/>
- [2] Proyecto CIAA, "CIAA Firmware Project," 2014, disponible: 2016-06-25. [Online]. Available: <https://github.com/ciaa/Firmware>
- [3] G. L. Puga, *Desarrollo de capa de compatibilidad del Firmware CIAA para procesadores LEON3*, 2016.
- [4] *OSEK/VDX - Operating System*, OSEK/VDX Std., Rev. 2.2.3, 2 2005.
- [5] M. Cerdeiro, (2015, 11) Introducción a OSEK-OS - El Sistema Operativo del CIAA-Firmware. ACSE.
- [6] *IEEE Standard for a 32-Bit Microprocessor Architecture*, Std., 1995.
- [7] *GRLIB IP Library User's Manual*, Cobham Gaisler, January 2016. [Online]. Available: <http://www.gaisler.com/index.php/downloads/leongrplib>
- [8] *GRLIB IP Core User's Manual*, Cobham Gaisler, January 2016. [Online]. Available: <http://www.gaisler.com/index.php/downloads/leongrplib>

Aplicaciones didácticas de la EDU CIAA, Octave y GUI Editor

Enrique Sergio Burgos
Universidad Tecnológica Nacional
Facultad Regional Paraná
Paraná, Entre Ríos, Argentina
sergioburgos@frp.utn.edu.ar

Resumen—En este trabajo se presenta un método para vincular la Computadora Industrial Abierta Argentina en su versión Educativa (EDU CIAA) a Octave a través de una herramienta para el desarrollo de interfaces gráficas llamada GUI Editor.

Se analiza el mecanismo de comunicación y una aplicación didáctica, consistente en el diseño de filtros digitales pasa bajos cuya implementación puede ser realizada directamente sobre la EDU CIAA.

I. INTRODUCCIÓN

Octave [1] es una conocida herramienta de cálculo, de código abierto, que contando con numerosos paquetes, permite trabajar en diferentes áreas de conocimiento. A través de un lenguaje propio permite la creación de scripts para resolver problemas de cálculo.

GUI Editor es una herramienta creada para construir interfaces gráficas de modo visual, generando scripts que Octave puede interpretar. De este modo, constituye un entorno de desarrollo básico, que incorpora íconos para representar diferentes objetos que Octave trata como controles visuales. Trabajos anteriores han presentado las posibilidades del uso de GUI Editor y Octave para el desarrollo de simuladores con fines educativos [2], [3].

En las más recientes etapas de su desarrollo y a partir de las posibilidades que brinda el uso de herramientas matemáticas conjuntamente con sistemas embebidos (entre otros [4]), GUI Editor ha incorporado controles que permiten la conexión con hardware externo, posibilitando obtener mediciones y controlar actuadores. En este trabajo se presenta la EDU CIAA como una interfaz de hardware para esta herramienta con fines educativos. Analizándose la forma en la que establece el vínculo entre las diferentes partes del sistema y, a modo de ejemplo, un caso de aplicación en el diseño e implementación de filtros digitales.

II. OCTAVE, GUI EDITOR Y EDU CIAA

Octave ha evolucionado desde sus orígenes incorporando diferentes interfaces de usuario. En principio, utilizaba una interfaz de comandos que, a partir de la versión 4, se transformó en una interfaz visual desarrollada a partir de las librerías Qt.

Este cambio en su arquitectura hizo posible el uso de controles gráficos a través de la función *uicontrol*. Según sean los argumentos que se le proporcionen es posible crear diferentes controles (botones, cuadros de edición, etc.) e incorporarlos a

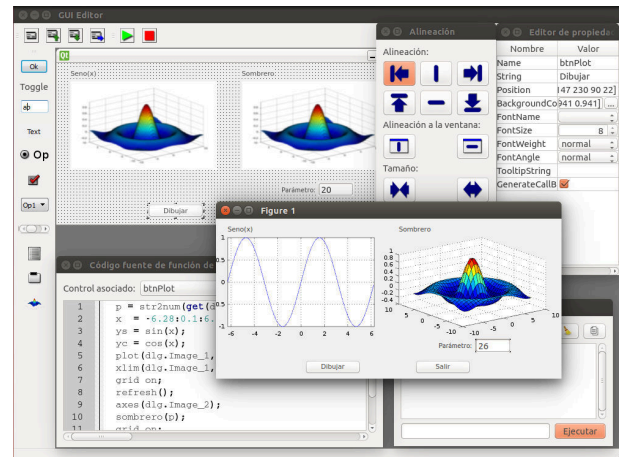


Figura 1: Aspecto de GUI Editor ejecutándose en Ubuntu Linux.

figuras, constituyendo ventanas que pueden cumplir el rol de interfaces gráficas para scripts.

Si bien en Octave existía la posibilidad de crear interfaces gráficas, no incluía ninguna aplicación que posibilitara la creación visual de ellas. En el marco de un proyecto de investigación y desarrollo, tendiente a la construcción de simuladores para la enseñanza de sistemas de control, se vio la necesidad de contar con una herramienta de este tipo, dando origen al desarrollo de GUI Editor.

GUI Editor (Fig. 1) ha sido desarrollado utilizando las librerías Qt y Qt Creator, por lo que mantiene la misma representación para los controles visuales que Octave. Es una aplicación que, en función del diseño creado por el desarrollador, genera un conjunto de scripts que permiten interactuar con el usuario.

Como muchos entornos de programación visual, asociados a cada control, se tienen un conjunto de propiedades. Éstas dependen del tipo de control del que se trate, y establecen su forma de representación y comportamiento. Como una propiedad más, aparece la posibilidad de asociar a cada control un callback. Esto es el equivalente a un evento, y cada control, tiene uno predeterminado según su tipo. Así, el callback asociado a un botón contendrá el conjunto de órdenes a ejecutar si se presiona en él, mientras que el callback asociado a un cuadro de edición contendrá el conjunto de órdenes a ejecutar

cuando se produzca alguna modificación sobre su contenido.

Además del editor visual de ventanas, con el objetivo de facilitar las tareas de edición asociadas a los callbacks, se incorpora un editor de código fuente basado en el control QScintilla [5]. Éste incorpora, entre otras características, resaltado de sintaxis automático y auto completado de expresiones.

Con el objetivo de agilizar el proceso de desarrollo, el entorno incorpora la posibilidad de probar (ejecutar) las interfaces invocando automáticamente a Octave. Por esto incorpora además una consola con la cuál es posible interactuar con el intérprete, pudiendo verificar el valor de variables y visualizar la salida de operaciones realizadas dentro de los callbacks.

Las interfaces son tratadas como documentos xml que contienen la definición de los controles utilizados así como el código fuente desarrollado. Esto posibilita la persistencia de las interfaces en archivos, pero para ser ejecutados en Octave requieren convertirse a scripts. Para esto, es necesario exportar la interfaz gráfica a un directorio particular, ya que cada callback es un archivo independiente invocado desde la aplicación principal. Algo similar ocurre cuando se prueba una interfaz gráfica dentro del editor, automáticamente se exporta a un directorio temporal y se invoca a Octave desde allí de modo transparente al usuario.

El uso de interfaces de hardware a través de GUI Editor implica un planteo diferente a lo presentado anteriormente. El primer factor que establece la diferencia es el hecho que Octave no cuenta con un control, objeto o función que, de modo nativo, represente el hardware. Sí se incorpora, a través del paquete instrument-control [6], la posibilidad de comunicación con dispositivos externos a través de interfaces de comunicación serie, tcp, vx11 entre otros. Así, la comunicación con hardware externo, y particularmente con la EDU CIAA, requirió la implementación de una clase que la represente en Octave llamada eduCIAA. A través de propiedades se representan las opciones de comunicación y con métodos, las acciones que se pueden realizar sobre la placa.

El vínculo entre la clase eduCIAA y la placa se implementó utilizando la interfaz de comunicación serie provista por el paquete instrument-control. Así, por cada mensaje que se le envía a un objeto de la clase, se transmiten órdenes a la placa para realizar tareas específicas. Para esto se implementó un protocolo de comunicación basado en paquetes. Donde cada uno está compuesto por entre 5 y 260 bytes, conteniendo una marca de inicio (el valor 0xB0), la cantidad de datos en el paquete, un valor que identifica la acción a ejecutar, 0 a 255 bytes que representan los argumentos de la acción (datos), una suma de comprobación y una marca de finalización (0x0B).

El firmware para la EDU CIAA se desarrolló como una aplicación en lenguaje C, baremetal, utilizando el IDE de la CIAA. El procesamiento del protocolo se realizó utilizando máquinas de estado (Fig. 2), donde el estado actual se representó como un campo de una estructura, mientras que las diferentes partes del protocolo (identificador de comando, argumentos y suma de comprobación) se almacenan en otros campos. Se utilizó un vector de punteros a funciones para realizar la transición de un estado al otro, recibiendo cada función la dirección de memoria de la estructura y el nuevo valor recibido (rxByte).

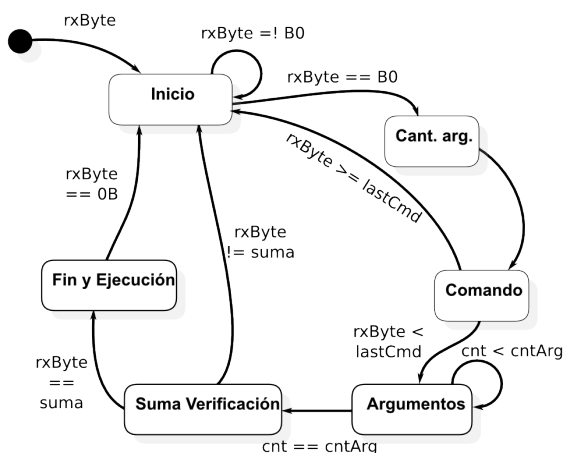


Figura 2: Diagrama de máquinas de estado utilizado para interpretar el protocolo de comunicaciones.

Una vez que se alcanza el último estado, al recibir el carácter indicador de fin de paquete, se invoca a una función contenida en otro vector de funciones para hacer efectiva la acción deseada. Cada una de estas funciones, recibe como argumento la dirección de memoria de la estructura que contiene la información de los datos que se han recibido.

La implementación utilizó la UART 2, conectada al adaptador USB de depuración y un mecanismo de interrupciones para procesar los valores recibidos.

Considerando el uso eminentemente didáctico que tiene la plataforma completa (Octave, GUI Editor y EDU CIAA), se definieron un conjunto de comandos básicos que tienen como objetivo servir de modelo para extender el protocolo implementado a casos particulares. Estos comandos se vincularon a métodos de la clase EDU CIAA, a saber:

- getADC: retorna el valor presente en una de las 3 entradas analógicas de la EDU CIAA.
- getTCK: retorna el estado de uno de los 4 pulsadores.
- setDAC: establece un valor de tensión analógico en la salida del DAC.
- setLed: enciende o apaga los uno de los 3 leds o uno de los 3 colores del led RGB.

Además se implementó un constructor que permite configurar el nombre del puerto de comunicación a utilizar y el valor de las salidas (leds y DAC).A modo de ejemplo se indica un fragmento de código script que crea un objeto de la clase eduCIAA e inicializa la salida del DAC y el estado del led 1 (Listado 1).

```

objCiaa = eduCIAA( ...
    "serialPort", "/dev/ttyUSB1", ...
    "ledNro_1", "On", ...
    "ledNro_2", "Off", ...
    "dacOutput", "512", ...
);
    
```

Listado 1: Inicialización de un objeto de la clase eduCIAA en Octave.

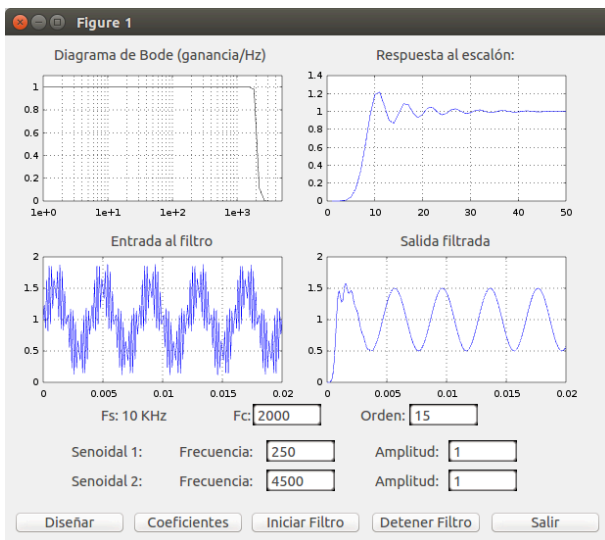


Figura 3: Interfaz del script desarrollado para el diseño de filtros digitales.

Una vez desarrollada la clase eduCIAA, la incorporación en GUI Editor se dio a través de un control de características no visuales. Éste, a diferencia de los demás controles disponibles, solo aparece como visible durante el ciclo de diseño de la aplicación. Permitiendo en esta etapa, y a través del editor de propiedades, configurar el puerto de comunicación y los pines de salida. Con el objetivo de simplificar la configuración se utilizó la interfaz de enumeración de puertos incluida en QExtSerialPort [7], de modo que el puerto de comunicaciones a utilizar debe seleccionarse de una lista de puertos disponibles en el sistema.

III. UN CASO DE APLICACIÓN

Con los métodos mencionados surgen aplicaciones directas para la EDU CIAA, tales como la adquisición de datos o control de actuadores, donde la computadora podría ser utilizada como interfaz sin realizar un procesamiento significativo. En estos casos, debido a la forma en la que se realiza la comunicación y la cantidad de capas de software utilizados, aparece como limitante la velocidad de respuesta. Siendo aplicable esta metodología a procesos de evolución lenta.

Sin embargo, la potencia del microcontrolador incorporado, permite la realización de procesos más complejos. Con el objetivo de explorar esta posibilidad con fines educativos, considerando antecedentes como [8] y [9], se desarrolló un script para el diseño de filtros digitales, pasa bajos, Butterworth.

El script permite establecer gráficamente la frecuencia de corte y el orden del filtro. Generando los coeficientes del mismo utilizando comandos del paquete signal [10] de Octave. Luego, para el filtro diseñado, representa la magnitud de la respuesta en frecuencia, la respuesta temporal al escalón y el resultado de aplicar al filtro la suma de dos señales senoidales de diferentes frecuencias. Finalmente, el filtro diseñado puede ser probado en la EDU CIAA, de modo automático, filtrando la señal captada por el canal 1 del ADC y actualizando la salida del DAC con el valor resultante del procesamiento.

En la implementación fue necesario realizar modificaciones a las interfaces presentadas anteriormente, tanto en el firmware de la EDU CIAA como en el protocolo de comunicaciones.

Este debió extenderse agregando comandos para transmitir los coeficientes del numerador y denominador del filtro e iniciar y detener el filtrado. Los coeficientes del filtro se transmitieron como valores de tipo double, realizando una conversión explícita antes de su transmisión desde Octave. Los valores resultantes utilizan el mismo tamaño y codificación que el tipo double en el entorno de desarrollo de la EDU CIAA utilizando el compilador arm-none-eabi-gcc versión 4.8.2.

Las características del protocolo utilizado establecen como limitación el uso de hasta 31 coeficientes para numerador y denominador. Por esto, y considerando el escaso consumo de recursos que tiene el firmware, los arreglos asociados a los coeficientes del denominador, numerador y auxiliares se definieron de modo estático tomando el máximo valor necesario de almacenamiento.

El firmware de la EDU CIAA se implementó de modo que, utilizando interrupciones, procesara los datos según arriban al puerto de comunicación serie. En el bucle principal se trabajó con dos banderas. Una indicaba cuándo debía realizarse el filtrado y la otra cuándo se tenía un nuevo dato proveniente del ADC.

En este esquema, al iniciarse el filtrado, se configura el ADC para trabajar por interrupciones de modo que tomara muestras 100.000 veces por segundo y, además, se actualice la bandera de filtrado. La función asociada a la interrupción del ADC, luego de 10 interrupciones, almacena el valor registrado, actualizándose la bandera vinculada a la disponibilidad de datos. El resultado efectivo, es una frecuencia de muestreo de 10 KHz. La implementación se hizo de este modo para mantener a la máxima frecuencia de trabajo el microcontrolador y bajar la frecuencia de muestreo.

Además, debido a que el sistema trabaja en tiempo real, se agregó una estructura de banderas que permitieran detectar cuando las operaciones matemáticas asociadas al proceso de filtrado toman más tiempo que el existente entre las muestras. Esta situación puede ocurrir según el orden del filtro que se utilice, ya que según aumenta el orden del filtro también aumenta la cantidad de operaciones matemáticas a realizar por cada nueva muestra.

Para la implementación del filtro se utilizó la segunda forma directa transpuesta, portando el código de [11] a lenguaje C. Para la generación de las formas de onda filtradas en Octave se utilizó el mismo algoritmo en su implementación original.

Al recibirse el comando de finalización de filtrado solo se actualiza la bandera que activa este proceso en el ciclo iterativo principal y desactiva las interrupciones del ADC.

La implementación realizada se probó en laboratorio haciendo uso de un osciloscopio digital, un generador de funciones, un sumador no inversor implementado con el operacional LF353, agregando a la entrada del ADC y a la salida del DAC un filtro RC. Se realizaron diferentes pruebas que consistieron en, para filtros de diferentes ordenes y frecuencias de corte, evaluar la atenuación realizada sobre una señal senoidal cuya frecuencia se variaba. También se ensayó el sistema de modo

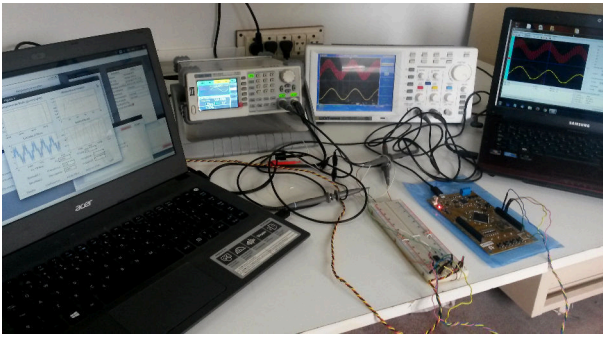


Figura 4: Fotografía de los equipos utilizados durante la realización de las pruebas del sistema.

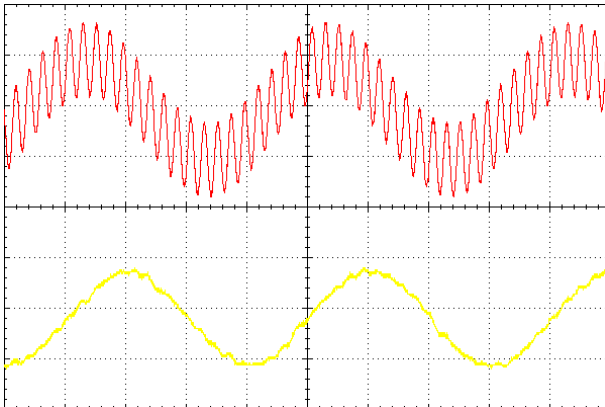


Figura 5: Captura de las señales observadas en el osciloscopio. En rojo se observa la entrada del filtro y en amarillo la salida filtrada.

de verificar la correspondencia entre la simulación mostrada por el script y los resultados físicos (Figs. 4 y 5).

Los resultados que se encontraron estaban dentro de lo esperado. En las pruebas donde se inyectó una única señal senoidal variando su frecuencia, se observó su atenuación conforme la frecuencia se acercaba y superaba la frecuencia de corte. En lo que respecta a la contrastación entre la simulación presentada y el resultado real, la simulación siempre tuvo resultados óptimos separando claramente las señales. Mientras que las mediciones realizadas mostraron un resultado similar al de la simulación cuando la frecuencia de las dos señales eran marcadamente diferentes (en el caso de la Fig. 5, 250 Hz y 4500 Hz con una frecuencia de corte de 2000 Hz). Según la frecuencia de las señales inyectadas al sumador se acercaban a la frecuencia de corte la señal resultante del proceso de filtrado se alejaba del resultado de la simulación, apareciendo restos de la señal atenuada.

IV. CONCLUSIONES

Los desarrollos aquí presentados han sido originados a partir de las tareas asociadas a la construcción de simuladores de sistemas de control, donde entradas y salidas forman parte de la simulación. En este trabajo se muestra una forma de utilizar mediciones reales en procesos de cálculo. Se ha mostrado una forma de interconexión entre software y hardware,

evaluándolas en una aplicación didáctica. El resultado no puede ser cuantificable, pero abre un sin fin de oportunidades para el desarrollo de actividades prácticas en la academia.

Intencionalmente no se profundizó en el algoritmo de implementación del filtro ni se intentó optimizar los cálculos que este realiza ya que en muchos casos, estas optimizaciones generan implementaciones complejas. Si se mostró que un caso práctico, tan simple como la implementación de un filtro digital pasa bajos, que puede diseñarse con herramientas de cálculo, haciendo uso de una interfaz de usuario simple, y posteriormente ser probado en un sistema embebido. Utilizando software y hardware abierto. Esto muestra las posibilidades de las herramientas presentadas para ser utilizadas en aplicaciones educativas que involucren los conocimientos asociados a más de una cátedra sin que sea necesario el uso de conocimientos avanzados de ningún área particular.

El trabajo presentado aún no ha sido utilizado en el ámbito educativo ya que ha sido de reciente realización. Se espera presentarlo en el futuro y analizar la respuesta de los alumnos.

Se prevé continuar en el desarrollo de interfaces como las presentadas de modo que sea posible incorporar núcleos de cálculo de diferentes tipos en sistemas embebidos como la EDU CIAA y permitan la aplicación de conceptos teóricos en casos prácticos analizando las diferentes aristas de su implementación. Un desafío por delante es desarrollar estructuras como las presentadas de modo que puedan utilizarse en ambientes industriales para realizar tareas de sintonía, identificación de sistemas o configuración de parámetros en sistemas complejos utilizando la CIAA.

REFERENCIAS

- [1] J. W. Eaton, "Gnu octave manual." <https://www.gnu.org/software/octave/>, Último acceso: 05-2017.
- [2] E. S. Burgos, O. E. Berardi, S. M. Vicario, F. A. Sala, H. Ramos, S. H. Pañoni, F. Sato, and E. Adam, "Simulaciones en la enseñanza de control de procesos mediante software libre," *IX Congreso Argentino de Enseñanza de Ingeniería (CAEDI) 2016*, pp. 1059–1068, 2016.
- [3] E. S. Burgos and E. Adam, "Desarrollo de interfaces gráficas para enseñanza de control de procesos," *XXV Congreso Argentino de Control Automático AADECA 2016*, pp. 1059–1068, 2016.
- [4] S. Silva, S. Soares, A. Valente, and S. T. Marcelino, "Digital sound processing using arduino and matlab," in *Science and Information Conference (SAI), 2015*. IEEE, 2015, pp. 1184–1191.
- [5] "Qscintilla," <https://www.riverbankcomputing.com/software/qscintilla/intro>, Último acceso: 05-2017.
- [6] A. Sutas and S. Mahr, "Instrument-control package, extra packages for gnu octave," <https://octave.sourceforge.io/instrument-control/>, Último acceso: 05-2017.
- [7] "Qextserialport - serial port library for qt2, qt3, qt4 and qt5," <https://qextserialport.github.io/>, Último acceso: 05/2017.
- [8] J. López, J. Fernández, M. Kuzman, W. Gemin, R. Rivera, M. Revuelta, and R. Hidalgo, "Desarrollo de filtros digitales embebidos en edu-ciaa," *Congreso Argentino de Sistemas Embebidos (CASE), 2016*, pp. 136–137, 2016.
- [9] M. E. Paz, O. A. Rodríguez, and C. L. Galasso, "Diseño de filtros digitales sobre un sistema embebido," *Congreso Argentino de Sistemas Embebidos (CASE), 2016*, pp. 31–36, 2016.
- [10] Varios, "The signal package," <https://octave.sourceforge.io/signal/>, mantenido por Mike Miller. Último acceso: 05-2017.
- [11] "2.161 signal processing: Continuous and discrete. fall 2008. massachusetts institute of technology: Mit opencourseware," <https://ocw.mit.edu>, Último acceso 05/2017, lecture 20, page 20-9.

Sistema de Adquisición y Procesamiento de Imágenes Térmicas de Bajo Costo

Gustavo Monte, Damián Marasco, Lucas Solorzano, Norberto Scarone, Ruben Bufanio

Universidad Tecnológica Nacional
Facultad Regional del Neuquén
Plaza Huinca, Argentina

Resumen—El diagnóstico mediante el análisis de imágenes térmicas es ampliamente empleado y es una de las herramientas predilectas del mantenimiento predictivo. Debido al alto costo de las cámaras infrarrojas, este análisis se efectúa mediante inspecciones manuales periódicas reutilizando el elemento sensor para diversas aplicaciones. La reciente aparición de cámaras infrarrojas de bajo costo ha posibilitado el desarrollo de aplicaciones fijas que observan y analizan el comportamiento térmico de sistemas críticos a lo largo del tiempo. Este trabajo presenta una propuesta de sensor térmico inteligente que adquiere y procesa la imagen de baja resolución en tiempo real. El procesamiento de la imagen térmica propuesta se basa en el análisis de máximos locales y los gradientes espaciales y temporales centrados en estos máximos. La adquisición y el procesamiento de la imagen se realiza en un microcontrolador de 16 bits resultando un sensor térmico compacto, de bajo costo y con la capacidad de entender y clasificar la evolución térmica del sistema observado.

Palabras clave—cámaras infrarrojas; microcontroladores; tiempo real; termografía; procesamiento imágenes térmicas; bajo costo.

I. INTRODUCCION

El término sensor inteligente fue acuñado en la década del 90 y hacía referencia a un transductor que tenía la capacidad de transmisión digital junto con la posibilidad de identificarse sobre una red de sensores. El desafío en esa época era disponer de dispositivos “*Plug and Play*”. Para lograr que transductores de distintas marcas pudieran identificarse se creó la familia de estándares IEEE 1451 [1], que introdujo la **TEDS** (*Transducer Electronic Data Sheet*), la hoja de datos electrónica del transductor.

En el tiempo actual el desafío es otro. Un sensor inteligente debería validar sus propias mediciones, extraer información y conocimiento directamente de la señal sensorial de manera tal de poder interactuar con otros sensores o transductores y el sistema de adquisición. Este “compartir” posibilita la ejecución de algoritmos de fusión sensorial y agiliza los procesos de minería de datos además de incrementar la confiabilidad del sistema involucrado.

Existe una relación inversa entre el procesamiento de la señal sensorial y el flujo de datos en bits/seg. A medida que se incrementa el procesamiento de la señal disminuye notablemente la cantidad de bits necesaria para reportar un evento. Un sensor basado en imágenes podría reportar la imagen completa para que sea procesada en forma remota (datos), una versión comprimida de la imagen, (información) o una extracción de alguna característica de relevancia para la aplicación (conocimiento). Estas tres clases de reporte generan miles de bits/seg, hasta algunos pocos bits/seg. en el caso de reportar solamente estados como por ejemplo, normal / anormal. La convergencia tecnológica es lo que posibilita este tipo de análisis y genera nuevos paradigmas de sensado, como es el caso particular propuesto de análisis de imágenes térmicas en posición fija y en forma continua.

En este trabajo se presenta el desarrollo de un sistema de adquisición y procesamiento de imágenes térmicas que emplea cámaras IR de bajo costo. Este sistema está orientado al monitoreo permanente de un proceso o dispositivo crítico que requiere monitoreo frecuente de su evolución térmica. La adquisición y procesamiento de un frame es llevada a cabo en un microcontrolador de 16 bits, generando una solución en un chip. En las secciones subsiguientes, se describe primero la naturaleza de una imagen térmica y la tecnología de las cámaras infrarrojas poniendo énfasis en sus virtudes y debilidades. Luego, se detalla la implementación de la comunicación microcontrolador - cámara IR para la configuración y adquisición de una captura. En la sección IV se describen los algoritmos empleados en el análisis de la imagen y como estos son implementados en el microcontrolador. Por último, se presentan resultados experimentales y las conclusiones.

II. TERMOGRAFIA INFRARROJA

A. Cámaras termográficas

Una cámara termográfica es un dispositivo que detecta el patrón de emisión de la escena de visión en el espectro de la longitud de onda infrarroja. En la Fig. 1 se observa el espectro electromagnético detallando las bandas correspondientes a la zona infrarroja. La cámara empleada en los ensayos para este trabajo [2] posee una sensibilidad menor a 50 mK (*miligrados Kelvin*), en la región LWIR (*Long Wave Infrared*) desde 8 a 14 μm .

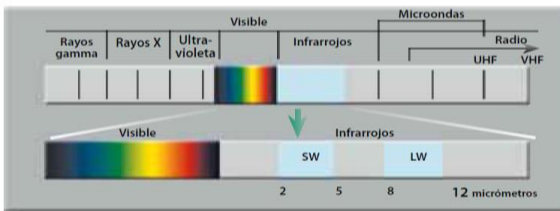


Fig. 1. Espectro electromagnético. La banda infrarroja se divide en dos subbandas principales: Short Wave IR correspondiente a una longitud de onda de 2 a 5 μm y Long Wave IR desde 8 a 12 μm .

Las cámaras termográficas están compuestas básicamente de una lente transparente a la energía infrarroja y de una matriz de $M \times N$ elementos sensibles que traducen esta energía a una señal eléctrica. Por lo tanto, las cámaras IR proporcionan la radiación de energía promedio en cada subdivisión de la escena según se proyecta a cada elemento de la matriz de sensores a través del sistema óptico. En el caso particular de la cámara empleada, la resolución es de 80×60 elementos sensibles. Cada elemento detector es un microbolómetro de óxido de vanadio, en el cual la temperatura varía según la radiación incidente. Los cambios de temperatura producen un cambio proporcional en la resistencia de cada detector. Mediante circuitos de sensado y conversores A/D la información matricial es enviada en forma secuencial mediante protocolos normalizados descriptos en la sección III.

B. Radiación y temperatura superficial

La potencia de radiación de una superficie real está dada por la ley de Stefan-Boltzmann:

$$\epsilon = \epsilon \sigma T^4 \quad (1)$$

Donde ϵ es la emisividad, $\sigma = 5.67 \cdot 10^{-8} \text{ W/m}^2\text{K}^4$, es la constante de Stefan-Boltzmann y T la temperatura de la superficie en grados Kelvin.

La emisividad es la relación de emisión referida a la emisión del cuerpo negro ideal, para cual toma el valor unitario. La emisividad es baja para metales pulidos. Como la emisividad es determinada por las características de la superficie del objeto, un metal pintado cambia su emisividad a la correspondiente a la pintura empleada.

Material	Emisividad (ϵ)
Cuerpo negro	1
Piel humana	0.98
Agua	0.98
Amianto	0.95
Cerámica	0.95
Barro	0.95
Cemento	0.95
Tejido	0.95
Grava	0.95
Papel	0.95
Plástico	0.95
Goma	0.95
Madera	0.95
Cobre (oxidado)	0.68
Acero inoxidable	0.1
Cobre (pulido)	0.02
Aluminio (pulido)	0.05

Fig. 2. Valores aproximados de emisividad para distintos materiales [3].

Si no se conoce la emisividad del objeto en la escena, (1) presenta dos incógnitas; ϵ y T , ya que las cámaras son sensibles a la radiación en el espectro infrarrojo.

Este análisis simplificado conduce a dos formas de emplear la termografía: análisis cualitativo y cuantitativo. La técnica de la termografía cualitativa consiste en detectar gradientes térmicos y patrones anormales en la escena de inspección. El defecto causante de este patrón térmico se localizará por comparación con otros cuerpos de esa misma condición. Una de las ventajas de esta técnica es que no necesita una medida exacta de la temperatura, sino que la clasificación normal/anormal se basa en análisis comparativo.

La termografía cuantitativa obtiene la temperatura en cada punto de la escena, lo que implica conocer la emisividad del objeto. La gran mayoría de las aplicaciones se basan en análisis cualitativo dado que la información del comportamiento térmico del objeto es función de la distribución espacial de temperaturas relativas. En aplicaciones fijas, como la propuesta en este trabajo, permite además obtener la distribución temporal al observar la misma escena a lo largo del tiempo. Esta dimensión adicional posibilita detectar condiciones anormales de operación o sobrecarga calculando los gradientes temporales de regiones de la escena.

Para ilustrar estos conceptos, en la Fig. 3 se observa la imagen térmica de un transformador de distribución de intermedia captura con nuestro sistema de adquisición. El transformador posee una distribución uniforme de temperaturas, pero se observa una zona rectangular más fría. Esta zona corresponde a la placa metálica de identificación de la máquina eléctrica que al tener un coeficiente de emisividad bajo se observa, erróneamente como más fría.

En una aplicación de adquisición fija, los elementos de baja emisividad son conocidos de antemano, por lo tanto, pueden ser descartados del análisis del comportamiento térmico.

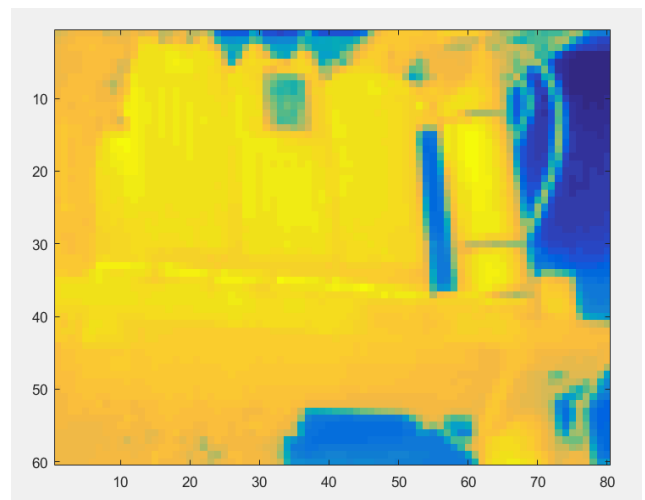


Fig. 3. Imagen térmica, obtenida con nuestro sistema de adquisición, de un transformador de distribución de intermedia. El rectángulo más frío observado corresponde a la placa identificatoria metálica de baja emisividad. Se observa una distribución homogénea de temperaturas.

III. SISTEMA DE ADQUISICIÓN

A. Cámara térmica

La cámara térmica empleada es marca FLIR de la serie LEPTON. Este dispositivo integra lentes de foco fijo, una matriz de 80x60 sensores infrarrojos, y electrónica para procesamiento digital. Puede operar tanto en su modo por default o ser configurado en otros múltiples modos, por medio de comandos y una interfaz de control integrada (CCI). En la Fig. 4 se detallan las principales especificaciones.

Características destacadas:

- Sensibilidad térmica < 50mK,
- Funciones para procesamiento de imágenes térmicas integradas, compensación automática de ambientes térmicos, filtros de ruido, corrección de no uniformidad y control de ganancia,
- Bajo consumo en operación (150 mW),
- Bajo consumo en modo Standby (4 mW).

Especificación	Descripción
Destacadas	
Tecnología del Sensor	Microbolómetros no enfriados de Oxido de Vanadio.
Rango espectral	Longitud de onda infrarroja, 8 a 14 uM
Formato de salida	Seleccionable 14 bit's u 8 bit's (AGC habilitado)
Eléctricas	
Reloj de entrada	25 MHz nominal
Interfaz de Video	VoSPI
Voltaje de alimentación	1.2 V, 2.5 V, 2.8 V
Ambiental	
Rango temperatura de operación óptimo	(-)10 °C a (+)65 °C

Fig. 4. Principales especificaciones de la cámara térmica utilizada.

La imagen, dentro de la cámara, es procesada por una serie de algoritmos para mejorarla, algunos por defecto y otros que pueden ser habilitados por medio de la interfase serial CCI. Dos algoritmos se destacan:

- NUC: es bloque de corrección de no-uniformidad,
- AGC: algoritmo de conversión de resolución completa a una imagen de formato más adaptable para mostrar.

Modos de Operación:

Modo Telemetría. Este modo está por defecto desactivado; si estuviera activado incluye en el frame de salida una serie de líneas de datos con información sobre algunos estados de la cámara y su configuración. La telemetría puede estar al inicio (Telemetry as Header) o al final (Telemetry as Footer) del frame.

Modos Radiométricos. Estos modos también están por defecto desactivados; estos modos afectan la función de transferencia entre la radiación incidente y el pixel de salida. La radiometría habilitada es preferida cuando la salida de la cámara se desea entender como una escena de temperatura proporcional. En este modo la conversión es constante sobre el rango de

operación completo. Se debe tener en cuenta que, para notar diferencias entre los modos de radiometría, debemos anteponer el modo AGC deshabilitado.

Radiometría deshabilitada: Con esta opción la temperatura es interpretada cercana a la mitad del rango de conversión, es decir que si la temperatura de un pixel de la escena es igual a la temperatura de la cámara el valor del conversor A/D estará en la mitad del rango. La respuesta a la radiación varía respecto a la temperatura propia de la propia cámara. En este modo se debe ajustar la temperatura de salida por pixel respecto a la temperatura ambiente en la cámara.

Radiometría habilitada: el dispositivo ajusta internamente la señal de cada pixel para tratar de compensar el efecto de la temperatura propia de la cámara. En este modo la temperatura de cada pixel, es independiente de la temperatura de la cámara.

Modo AGC: El control automático de ganancia es un método mediante el cual se optimiza el amplio rango de conversión, para dejarlo sobre un rango más apropiado para un sistema de visualización. La reducción en rango de valores tomados por la imagen se realiza por medio de una modificación propietaria del algoritmo de ecualización de histogramas acumulativos (Lepton's Variant of Histogram Equalization).

A1. Interfaz de Control:

El sistema de la cámara infrarroja Lepton, incluye una interfaz de control (CCI). La misma, es de dos conductores; similar al I2C. La diferencia sobre el I2C, es que la interfase de control es de 16 bits.

La dirección de la interfaz es 0x2A. En general, podemos utilizar esta interfaz para tres tipos de comandos:

- *Get:* lectura de datos,
- *Set:* escritura de datos,
- *Run:* ejecución de una función.

A2. Interfaz de Video:

El protocolo de transferencia es SPI, lo que permite una rápida y verificable transferencia de datos. El mismo está basado en paquetes sin temporización y sin requerimiento de control de flujo.

Toda transferencia de paquetes de datos (VoSPI packets), es controlada por el maestro y una vez recibidos los paquetes correspondientes a un frame, el maestro puede optar por des-seleccionar el dispositivo o continuar la transferencia de paquetes de datos. En este caso se generan paquetes de descarte, (VoSPI discard packets) hasta que se genere un nuevo frame.

A3. Formato del paquete VoSPI:

El paquete de video, consta de 4 bytes para encabezado y 160 bytes para datos de pixeles. En el encabezado se encuentran tanto el identificador de línea, 12 bits, como el código CRC de 16 bits empleando el siguiente polinomio generador:

$$x^{16} + x^{12} + x^5 + x^0 \quad (2)$$

ID	CRC	Pixeles
4 bytes		160 bytes
ID	CRC	Pixeles
xFxx	xxxx	Datos no utilizables

Fig. 5. Trama de pixeles enviados por la interfase SPI para datos validos y descarte. Inferior, trama de descarte codificada por el ID 0XxFxx.

Por cada pixel se envían dos bytes por lo tanto, una trama es una línea completa de 80 pixeles. La trama VoSPI de descarte se diferencia de las válidas por el ID. Todas las tramas con una 0xF en el tercer nibble son de descarte, los datos y el CRC no tienen validez. Las tramas de descarte tienen la función de mantener el flujo de datos por la SPI mientras un nuevo frame se encuentre disponible.

B. Microcontrolador

El microcontrolador elegido fue el DSPIC33EP256GP502 de *Microchip*, debido a principalmente a cuatro motivos: Velocidad de ejecución, tamaño de RAM, costo y arquitectura de 16 bits para procesar eficientemente los pixeles empaquetados también en 16 bits. Posee 32Kb de RAM, opera a 70 MIPS y posee interfaces SPI, I2C y USART, las tres empleadas en el adquisidor. Para enviar comandos y adquirir las imágenes se agregó un conversor USB-USART FT230.

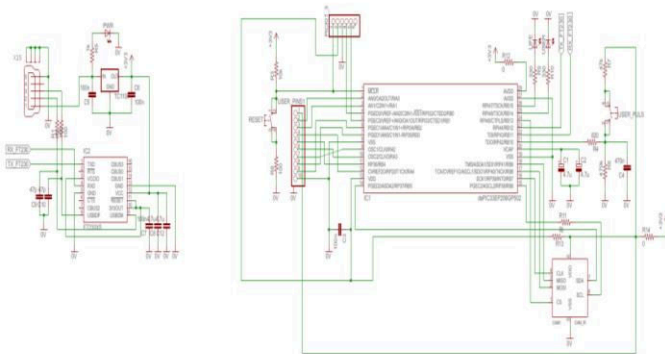


Fig. 6. Diagrama esquemático del adquisidor.

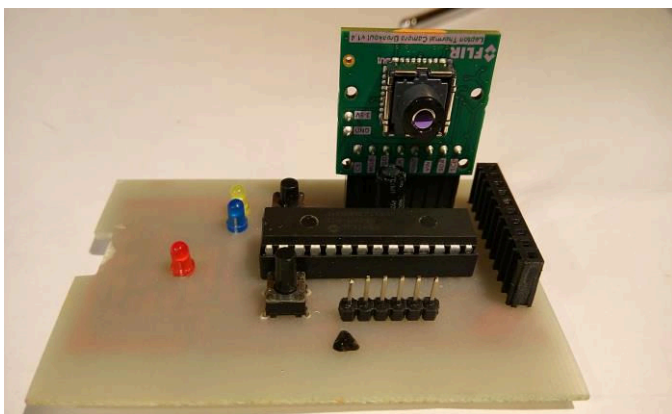


Fig. 7. Foto del prototipo de adquisidor.

En la Fig. 6 se observa el diagrama esquemático del adquisidor y en la Fig. 7 una foto del prototipo implementado.

C. Driver

El driver se ha desarrollado con el propósito de darle, a un sistema externo; una sencilla interfaz de configuración de la cámara y captura de imágenes mediante USB. Las funciones del driver son:

- Inicializar los periféricos de comunicación.
- Inicializar la cámara y verificar su normal funcionamiento.
- Esperar comandos de captura, configuración, envío de imagen, distribución térmica, validación de escena, procesamiento imagen térmica.

El software fue desarrollado según el paradigma de multitarea Estado-Cooperativo en lenguaje C empleando el compilador XC16 en el IDE MPLABX.

La función más importante del driver es establecer el sincronismo con la cámara en la interfase SPI. Conocido el formato de la trama de video, por medio de SPI iniciamos el proceso de transferencia de datos imponiendo la señal de CLK como maestro a una frecuencia de 20 MHz.

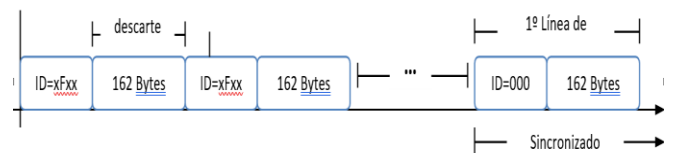


Fig. 8. Para el sincronismo se debe detectar la primera trama con ID=0000.

Para la sincronización, debemos descartar todas las tramas de video desechables hasta encontrar un identificador de línea correspondiente al inicio de un frame válido (ID = 0x0000), ver Fig. 8.

El sistema externo solo interactúa con el microcontrolador por medio del protocolo serial sobre USB para adquirir la imagen, especificar una nueva configuración del modo de captura, o requerir análisis térmico según los algoritmos descritos en la sección siguiente.

IV. PROCESAMIENTO DE LA IMAGEN TÉRMICA

A. Condiciones de borde

El procesamiento sobre la imagen térmica fue diseñado para ser realizado empleando los recursos del microcontrolador. Como no se conoce de antemano la escena para la aplicación final y para que sea genérico se implementaron algoritmos que extraen características comunes a numerosas aplicaciones. Además, existe la posibilidad de enviar por el puerto serial el frame completo para ser procesado en forma remota.

En general, en una imagen térmica, es útil determinar el valor absoluto de puntos calientes, el gradiente espacial de

temperatura centrado en estos puntos y la evolución temporal de ellos.

B. Maximos de la imagen

Para la detección de los máximos de la imagen, puntos calientes, se utilizó el estándar IEEE 21451-001-2017 [4], referida al tratamiento de señales aplicado a sensores inteligentes. En nuestro caso particular, se considera a la imagen como un arreglo de 60 segmentos de señales unidimensionales de 80 elementos. De acuerdo al estándar, la señal es codificada como una unión de segmentos conocidos, en [5][6]. hay una descripción completa de la codificación. Cada frame consta de 4800 pixeles. Como resultado de aplicar la técnica de muestreo sugerida en la IEEE 21451-001, se marcan pixeles solo en la unión de segmentos, quedando una cantidad considerablemente menor, que es función de la escena y de un error de interpolación prefijado. Estos pixeles se llaman esenciales y se considera a esta forma de describir la información como una compresión sin pérdida de la estructura de la información. En la unión de un segmento ascendente con uno descendente, se obtiene un máximo local de forma directa. Se procesan las filas de la imagen en forma secuencial para la obtención de los pixeles esenciales y de los máximos locales controlados por un error de interpolación.

C. Algoritmos de extracción de características

El frame adquirido es subdividido en 16 regiones de 20 x15 pixeles. Esta subdivisión es realizada por dos motivos. Primero, simplifica el análisis del comportamiento térmico y lo adapta a los recursos de microcontrolador y segundo, se obtiene una firma digital de la escena que es empleada para validarla, teniendo en cuenta que los procesos térmicos son lentos y el objetivo del adquisidor es observar procesos en forma continua.

Para cada región i,j con $i=1, 2, 3,4$ y $j=1, 2, 3,4$ se realizan los siguientes procesos.

- 1- Cantidad de pixeles esenciales. (Cpix)
- 2- Valor medio de los pixeles esenciales. (Mpix)
- 3- Cantidad de máximos. (CMax)
- 4- Máximo de mayor valor. (MMax)
- 5- Diferencia entre la media de los máximos y la media de los pixeles esenciales. (DifMpixMax)

Por lo tanto, se obtienen 80 indicadores que describen la escena, son sencillos de calcular y son empleados para validarla y analizarla.

D. Validación de la escena

Validar la escena en este contexto significa que se está adquiriendo la misma imagen o que hay energía térmica en juego en la escena. La diferencia entre dos frames seguidos obtenida en un tiempo mucho menor que las constantes de tiempo térmicas involucradas, debe ser pequeña. Dos motivos

principales generan grandes diferencias. Primero que no exista energía térmica significativa en la escena, por ejemplo, la maquina está apagada y segundo que la escena cambió en forma abrupta, por ejemplo, se interpuso algún objeto o persona. Para validar una escena se debe obtener una diferencia menor a un umbral, Thd , en una cantidad de frames consecutivos Kd . Para la validación se emplean los procesos 1 y 2 según las siguientes ecuaciones:

$$dif1 = \sum |Cpix_{i,j,n} - Cpix_{i,j,n-1}| \quad (3)$$

$$dif2 = \sum |Mpix_{i,j,n} - Mpix_{i,j,n-1}| \quad (4)$$

Donde $i=1, 2, 3,4$; $j=1, 2, 3,4$ y n es el frame actual y $n-1$ el anterior, con una diferencia temporal mucho menor que las constantes térmicas involucradas. Un nuevo frame será validado si se cumple la siguiente inecuación:

$$dif1 + dif2 < Thd \quad (5)$$

para Kd frames consecutivos.

E. Analisis térmico de frames validados

Se ejecutan dos procesos básicos: el crecimiento temporal de los máximos y la evolución espacial de la energía térmica en cada región. Se calculan los valores resultantes para cada uno de los 5 procesos en cada región. Se base a los resultados, se seleccionan las regiones que requieren atención para su seguimiento. Por ejemplo, en una región sin máximos no se realiza ningún calculo posterior. Se busca el máximo absoluto de la imagen y se registra la evolución temporal de las regiones vecinas en sucesivos frames, ver Fig. 9.

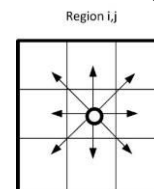


Fig. 9. Regiones vecinas centradas en el máximo absoluto de la imagen.

F. Resultados experimentales sobre imágenes reales

Se procesaron imágenes térmicas de un disipador de un transistor de potencia. Se adquirieron 30 frames a intervalos de 20 segundos. En la Fig.10 se observa la imagen térmica del disipador y en la Fig.11 la evolución temporal del máximo MMax de la imagen.

En la Fig.12 se presenta el estado térmico inicial y final del disipador luego de los 10 minutos de ensayo. Se calcularon los 5 procesos y es importante destacar la importancia del proceso 5, DifMpixMax, ya que un aumento de ese valor implica que el calor se está concentrando en pequeñas superficies o que la velocidad de generación de calor es mayor a la de evacuación. En la Fig. 13 se observa la captura de la imagen térmica de un motor, que a través de las rejillas de ventilación se puede inferir la temperatura del bobinado.

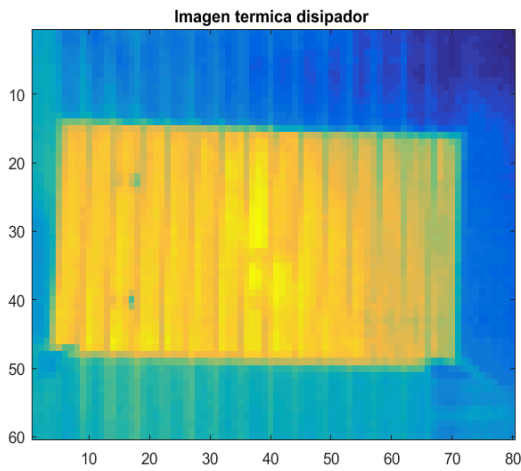


Fig. 10. Imagen termica del disipador bajo ensayo.

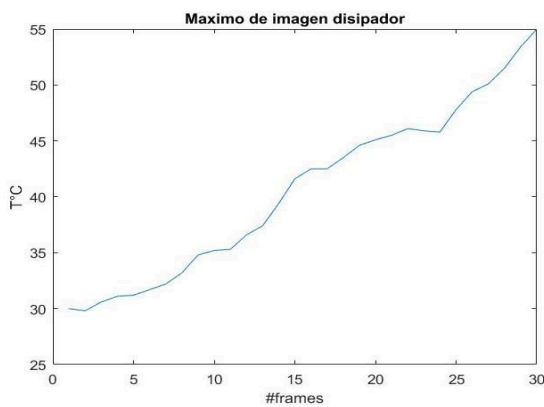


Fig. 11. Evolucion temporal del máximo MMax de la imagen disipador

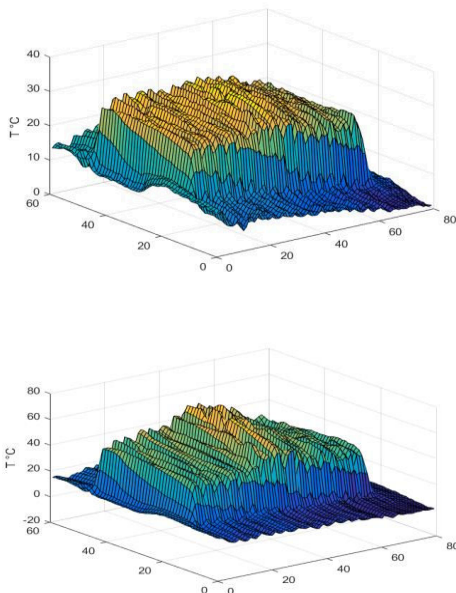


Fig. 12. Imagen termica del disipador al principio (superior) y al final del ensayo (inferior).

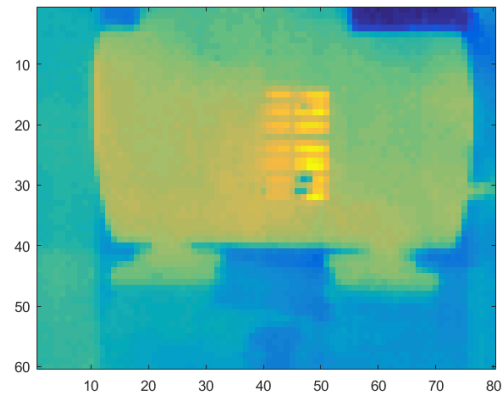


Fig. 13. Imagen térmica de un motor en el cual se puede medir la temperatura de los bobinados por las rejillas de ventilación. Capturada con el sistema de adquisicion.

V. CONCLUSIONES

La termografía presenta la gran ventaja de observar un proceso o dispositivo sin contacto. En el caso presentado en este trabajo la resolución es de 80 x 60. Aunque es baja, es como observar al proceso con 4800 termómetros. Al bajar los costos de las cámaras infrarrojas se disparan aplicaciones con capturas fijas que permiten evaluar el comportamiento térmico con la filosofía de un sensor inteligente. Las aplicaciones son innumerables, no solo para inferir patrones anormales, sino que además para validar el correcto desempeño de un sistema, por ejemplo, verificar que alcanza el régimen permanente con normalidad. Se proponen 5 indicadores de comportamiento térmico y el seguimiento de regiones que presentan características de evolución térmica.

En algunas aplicaciones también son importantes los valores mínimos de una imagen, como por ejemplo en estudios de eficiencia energética. Los máximos y los mínimos locales son resultados inmediatos del estándar IEEE 21451-001 y puede ser implementada en microcontroladores de mediana complejidad.

REFERENCIAS

- [1] E. Song and K. Lee, "Understanding IEEE 1451-networked smart transducer interface standard – what is a smart transducer?" IEEE Instr.Meas. Mag., vol. 11, no. 2, pp. 11–17, 2008.
- [2] <http://www.flir.com/cores/content/?id=66257>
- [3] <http://www.interempresas.net/Plastico/Articulos/31411-La-termografia-en-la-industria-del-plastico.html>.
- [4] <https://standards.ieee.org/findstds/standard/21451-001-2017.html>
- [5] G. Monte, "Sensor Signal Preprocessing Techniques for Analysis and Prediction" Industrial Electronics, 2008. IECON 2008. 34th Annual Conference of IEEE. pp 1788-1793. ISBN 978-114244-1766-7.
- [6] Monte ,G Abate, F; Huang, V;; Paciello V; Pietrosanto A; Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on Trabajo: "Real time transducer signal features extraction: A standard approach". Print ISSN: 1935-4576, Electronic ISSN: 2378-363X

Determinación de la actitud angular de una sonda suborbital mediante cámara digital y sistemas embebidos

Oviedo C. Carlos N.⁽¹⁾, Zimmermann Yamil E.⁽¹⁾, Turra Daniel N.⁽¹⁾,
González Gustavo J.⁽²⁾, Cova Walter J.D.⁽¹⁾⁽²⁾

⁽¹⁾GPS-Grupo Proyectos y Servicios

Universidad Tecnológica Nacional, Facultad Regional La Rioja,
San Nicolás de Bari (E) 1100, (5300) La Rioja, Argentina.

⁽²⁾Universidad Tecnológica Nacional, Facultad Regional Córdoba
Maestro M. López y Cruz Roja Argentina, (5000) Córdoba, Argentina.

Resumen— El presente trabajo detalla el desarrollo de un sistema embebido basado en hardware comercial de bajo costo para la captura y procesamiento de imágenes del horizonte de la tierra tomada a lo largo de la trayectoria apical de una carga útil suborbital con el objetivo de obtener información de la posición angular del cuerpo alrededor de su eje longitudinal. Ello requiere de un procesamiento digital para determinar la posición de la transición tierra-espacio (borde del horizonte) en el plano de la imagen, para luego utilizarla como una señal de actitud angular integrada en el sistema de estabilización de la carga útil. Para la implementación se empleó una single board computer Raspberry Pi con un sistema operativo basado en Linux, al que se integra una cámara RaspiCam, empleando Qt-5 como compilador. Se presenta entonces la utilización de una cámara digital, como sensor de horizonte de bajo costo, implementando un software embebido para el procesamiento de la imagen. Se evaluó el tiempo de procesamiento de imagen determinándose su compatibilidad con los requerimientos del sistema de control. Adicionalmente se analizó la eventualidad de un enfoque directo de la cámara hacia el sol, determinándose la necesidad de contar con un inhibidor para evitar que los efectos de persistencia de imagen afecten la performance del sistema de control.

Palabras clave— Sensor de horizonte; Procesador Raspberry Pi; Cámara digital; Tiempo de procesamiento; Persistencia de imagen.

I. INTRODUCCIÓN

La utilización de cámaras digitales para la determinación de la actitud de vuelo de micro satélites ha sido descrita en diversas publicaciones [1-3], las que asimismo presentan algoritmos de detección de horizonte basados en el contraste tierra-espacio tanto en el espectro visible como en el infrarrojo.

En las aplicaciones suborbitales (como la graficada en la Figura 1), debido a la corta duración del segmento de trayectoria útil, resulta de interés lograr una rápida estabilización de la actitud del vehículo (o carga útil), a fin de maximizar el tiempo disponible para la realización de las experiencias

El presente trabajo es parte del Proyecto de Investigación y Desarrollo Interfacultades financiado por la Universidad Tecnológica Nacional, código AMIFNLR0003995: *Desarrollo de Sensor de Horizonte Basado en Cámara Digital*, ejecutado en forma conjunta por las Facultades Regionales La Rioja y Córdoba, con el apoyo de los laboratorios del Centro de Investigaciones Aplicadas (CIA) perteneciente a la Fuerza Aérea Argentina.

científicas embarcadas. La brevedad de la misión, conduce a su vez a dar preferencia en la implementación del sistema de control de actitud, al empleo de sensores y actuadores robustos y de bajo costo, pero sin que los condicionamientos económicos afecten los requerimientos de confiabilidad operativa.

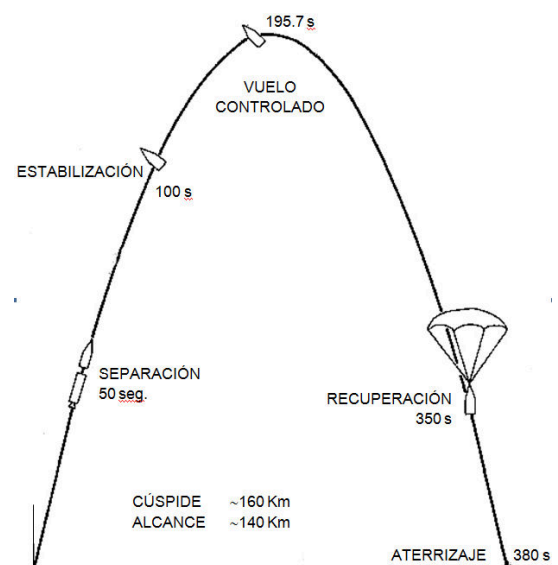


Fig. 1. Ejemplo de trayectoria suborbital y sus principales eventos.

El presente trabajo expone la implementación de un sistema de control de actitud basado en componentes electrónicos de uso civil y comercial y software libre, centrandose la presentación en los aspectos relacionados al procesamiento de imagen y su influencia sobre la performance global del sistema. La organización del trabajo es la siguiente: en la Sección II se sintetizan las condiciones generales de operación del sistema de control y los requerimientos de performance para el procesamiento de imágenes que de ellas se deducen. La selección de hardware y la plataforma operativa de software se exponen en la Sección III. En la

Sección IV se trata la evaluación del funcionamiento del sistema embebido implementado (en especial: tiempo de procesamiento y de persistencia de imágenes de alta luminosidad). Finalmente, las conclusiones y futuros desarrollos se exponen en la Sección V.

II. CONDICIONES DE OPERACIÓN

La Figura 2 muestra la posición nominal de la carga útil estabilizada a una altura (h) dada, para la cual el horizonte terrestre subtende el ángulo $\beta/2 \approx 90^\circ - [h \text{ (km)}]^{1/2}$. En esas condiciones hipotéticas, el eje longitudinal de la carga útil se encuentra orientado según alguna dirección unívocamente determinable con la instrumentación de abordaje (p.ej. con el campo magnético terrestre B , si se cuenta con magnetómetros), mientras que uno de los ejes transversales (asociado al eje óptico de la cámara) se encuentra “enganchado” con el horizonte visible. La existencia de perturbaciones aerodinámicas y gravitatorias –si bien son de baja intensidad a las alturas de vuelo consideradas– afectarán la orientación de la carga útil, por lo que en definitiva el vehículo se encontrará oscilando en un ciclo límite alrededor de su posición nominal.

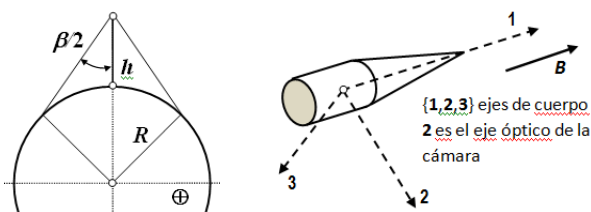


Fig.2. Ángulo del horizonte y esquema de la carga útil

Por otra parte, la limitación de los recursos energéticos disponibles durante el vuelo (p. ej. la cantidad de gas disponible, en el caso de aplicar estabilización por eyección de gas en operación bang-bang), obligan la existencia de un ciclo límite de movimiento, definido por una serie de factores tales como las tolerancias de posicionamiento (condicionadas a su vez por la naturaleza de la experiencia científica embarcada), la duración mínima de accionamiento, la disponibilidad temporal de información de los sensores angulares y la caracterización estadística de las perturbaciones actuantes sobre el cuerpo, tal como se discutiera en [4].

A fin de establecer la influencia que posee la disponibilidad de información de actitud angular respecto del horizonte sobre los requerimientos energéticos operativos, se realizaron una serie de simulaciones numéricas. Se plantearon experimentos ideales con el vehículo rotando alrededor de un único eje, considerándose nulas las condiciones iniciales para los otros ejes y asimismo nulas las perturbaciones. En las condiciones descriptas, la dinámica de la carga útil se corresponde con la de un cuerpo libre de rotar sin perturbaciones alrededor de su eje de simetría (doble integrador), cuya posición angular y velocidad es controlada mediante actuadores todo-nada (control bang-bang). En la situación considerada, las Figuras 3 y 4 presentan la evolución dinámica del cuerpo en el plano de fase, mostrando la velocidad angular (en grados/seg) versus posición (en grados) para dos casos extremos de las simulaciones numéricas.

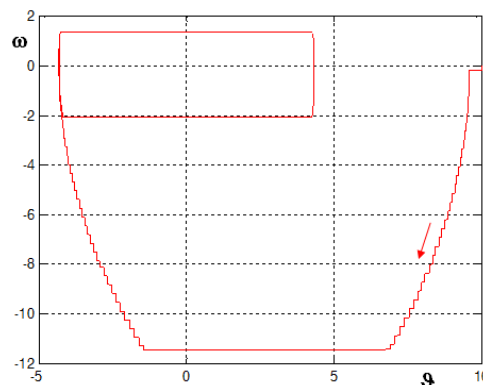


Fig. 3. Operación a 16 fotogramas por segundo.

El primer caso corresponde a una frecuencia de toma de imágenes de 16 fps (fotogramas por segundo o “frames per second”), mientras que el segundo corresponde a una frecuencia de 2 fps.

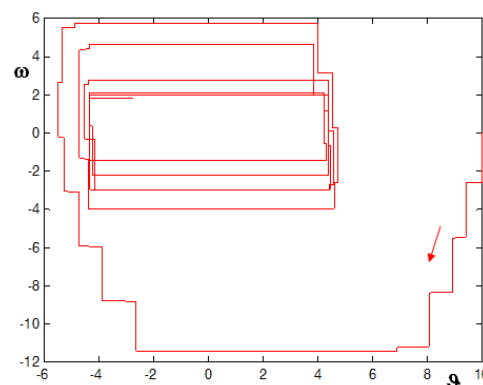


Fig. 4. Operación a 2 fps.

Las condiciones iniciales para las Figuras 3 y 4 son iguales, partiéndose de $\omega_0 = 0(^{\circ}/s)$ y $\theta_0 = 10(^{\circ})$. Los diagramas corresponden a una ley de control bang-bang que aplica aceleraciones de $\{-0.4, 0, +0.4\} \text{ rad/s}^2$ con una duración mínima de 0.150 s. El umbral de posicionamiento angular ha sido establecido en $\theta_R = 0.075 \text{ rad} = 4.3^{\circ}$.

Nótese que para los parámetros indicados, la amplitud teórica del ciclo límite en velocidad es $\Delta\omega = 0.15s \times 0.4 \text{ rad/s}^2 = 0.06 \text{ rad/s}$, es decir $\Delta\omega = 3.44^{\circ}/s$. Operando a 16 fotogramas por segundo el comportamiento dinámico de lazo cerrado se corresponde casi exactamente al que puede obtenerse para una disponibilidad continua del sensor angular, mientras que para 2 fotogramas por segundo el comportamiento se degrada muy sensiblemente, incrementándose el tiempo de adquisición (es decir el tiempo de entrada en ciclo límite) y consecuentemente se incrementa la energía que consume el sistema. En consecuencia resulta necesario determinar con precisión el tiempo requerido para obtener y procesar la información angular que proporciona la imagen captada por la cámara digital, a fin de validar su aplicabilidad como sensor.

III. MATERIALES Y MÉTODOS

Inicialmente se procedió a analizar cada uno de los elementos y procesos componentes del proyecto, tales como la

estabilización de un cuerpo en el espacio sometido a cuplas de reacción y los requerimientos de hardware. Luego se investigaron las opciones existentes en el mercado de placas de desarrollo y cámaras digitales. Entre las principales opciones se analizaron las siguientes con sus precios promedio (junio 2016):

- Arduino Due más cámara tipo OV7670 U\$D 100
- Beaglebone Black kit con cámara U\$D 125
- Raspberry Pi 3 modelo B con Raspicam U\$D 110

Cabe destacar que una cámara web convencional no es utilizable, debido a que generalmente brindan imágenes con compensación automática de brillo, color y balance de blancos, mientras que en nuestro caso deseamos trabajar con la imagen cruda (RAW) al objeto de tener una buena diferenciación entre claros y oscuros para determinar la transición tierra-espacio que define el horizonte.

Posteriormente se ensayaron distintos métodos y formatos para el procesamiento de imágenes, evaluando sus requerimientos temporales y de memoria. Pero en definitiva, la selección final estuvo condicionada por la disponibilidad inmediata de Raspberry y sus accesorios en el mercado local, ya que todas las placas analizadas cumplían con los requisitos mínimos y se encontraban dentro del presupuesto con el que se contaba. Los elementos seleccionados se describen en los apartados siguientes.

A. Raspberry Pi 3 modelo B

Entre los productos comerciales de hardware analizados se seleccionó Raspberry Pi 3 modelo B (Figura 5) debido a las facilidades de programación en distintos lenguajes, la posibilidad de contar con un extenso GPIO para el control a implementar y fundamentalmente por su puerto dedicado para cámara digital (RaspiCam), el cual como sabemos se puede expandir para agregar más cámaras como se desea hacer en un futuro.

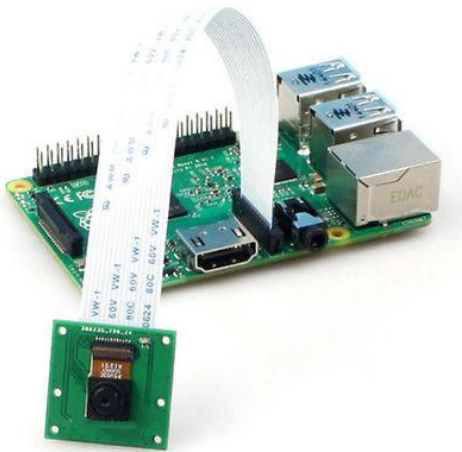


Fig. 5. Raspberry Pi 3 Modelo B junto con RaspiCam v1.3

B. RaspiCam V1.3

Esta cámara es de reducido tamaño $25 \times 20 \times 9$ mm, sensor OV5647 con resolución de hasta 5 megapíxeles

generando imágenes de 2592×1944 píxeles, foco fijo y grabación de vídeo a 1080p y 30 fps. Además, esta cámara permite la obtención de la imagen en distintas calidades y formatos, con compensación de brillo, color y balance de blancos. Pero su cualidad más importante es la capacidad de obtención de la imagen cruda (sin ninguna compensación ni procesamiento como los mencionados), sobre la cual es posible trabajar a nivel de pixel e individualizando los tres componentes primarios de color (RGB).

C. Entorno de Desarrollo de Software

Se optó por embeber el sistema operativo Raspbian y en el entorno de programación se empleó el compilador Qt versión 5. Raspbian es una distribución del sistema operativo GNU/Linux y por lo tanto libre basado en Debian Wheezy (Debian 7.0) para la placa de desarrollo (SBC) Raspberry Pi. Al ser una distribución de GNU/Linux cualquier software de código abierto puede ser utilizado en la propia placa de desarrollo.

Por lo que respecta al compilador, Qt es un framework multiplataforma orientado a objetos ampliamente usado para desarrollar programas que usen interfaz gráfica de usuario, así como también diferentes tipos de herramientas para la línea de comandos y consolas para servidores que no necesitan una interfaz gráfica de usuario. Qt utiliza el lenguaje de programación C++ de forma nativa, adicionalmente puede ser utilizado con varios otros lenguajes de programación.

IV. APLICACIÓN DESARROLLADA

A. Esquema Conceptual

En la Figura 6 se muestra el diagrama de bloques que representa conceptualmente el sistema de control de actitud para el eje de rolido de la carga útil.

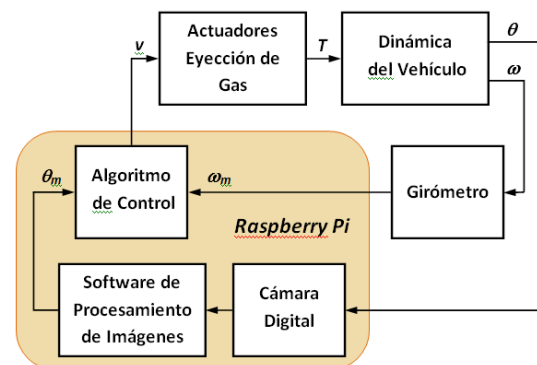


Fig 6. Diagrama de bloques del sistema de control de actitud.

En la placa de desarrollo se encuentran embebidos tanto el software de procesamiento de las imágenes obtenidas por la cámara, como el algoritmo de control que recibe la señal de velocidad angular ω_m (proporcionada por el girómetro) junto con el ángulo θ_m medido a través de la imagen. Su salida v activa a través de la GPIO de la placa Raspberry Pi la electrónica de comando de los eyectores de gas, generando una cupla T en el sentido necesario para reducir los errores en la actitud θ y en la velocidad angular ω del vehículo.

B. Procesamiento de Imagen

El programa embebido en Raspberry Pi 3 ordena la captura de una imagen mediante la cámara digital y posteriormente ejecuta el procesamiento pertinente para determinar el ángulo respecto del horizonte. Este procedimiento se repite periódicamente en intervalos que, estimativamente, no debieran superar los 50 ms.

La cámara digital envía a la CPU una imagen con formato RAW RGB sin compresión, y 8 bits de profundidad de color. La aplicación desarrollada en Raspberry Pi 3 analiza el color de cada pixel y efectúa su binarización por comparación contra un umbral preestablecido. Ya que se desea detectar el borde que delimita la tierra, se procesa solo el color azul, de manera tal de diferenciarla respecto del espacio. Luego del proceso, los pixeles más oscuros que el umbral quedarán en negro y los pixeles más claros se convertirán en blanco. Con ello, la imagen del horizonte queda binarizada, en base a la cual, mediante el procedimiento expuesto en [5] y que a continuación se sintetiza, se puede calcular el ángulo θ del eje de la cámara respecto del horizonte.

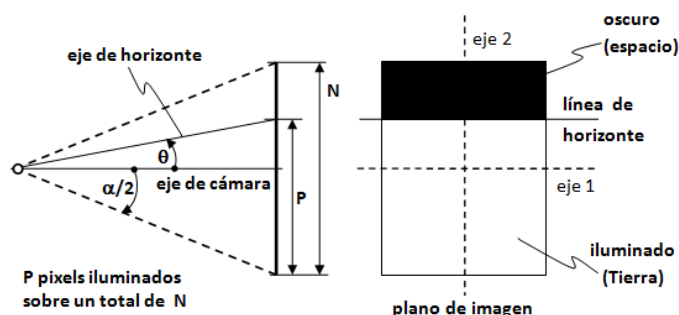


Fig. 7. Esquematación geométrica de la imagen del horizonte.

De la Figura 7 se deduce, en base a la semiapertura de cámara $\alpha/2$:

$$\frac{\tan(\theta)}{\tan(\alpha/2)} = \frac{(P - N/2)}{N/2} = \left(\frac{2P}{N} - 1\right)$$

$$\theta = \tan^{-1} \left[\left(\frac{2P}{N} - 1\right) \cdot \tan(\alpha/2) \right]$$

que para $\alpha \leq 30^\circ$ se puede aproximar con un error máximo menor que 0.15° mediante la expresión lineal:

$$\theta \approx \left(\frac{2P}{N} - 1\right) \cdot \tan(\alpha/2)$$

En [5] se discutió la influencia del desapuntado del eje longitudinal de la carga útil respecto de la dirección de referencia sobre el error en el ángulo θ sentido, determinándose que su influencia es mínima si el desapuntado está comprendido dentro de un cono de 15° semiapertura.

En definitiva, la componente azul de la imagen captada por la cámara es binarizada por comparación con un umbral, luego de lo cual se cuentan los puntos oscuros, cuya relación con la totalidad de puntos que integran la imagen es directamente proporcional al ángulo del horizonte.

Con el objeto de evaluar el procedimiento expuesto, se utilizó una imagen de la tierra obtenida por NASA [6] generándose la Figura 8, constatándose la correcta implementación del proceso de binarización.

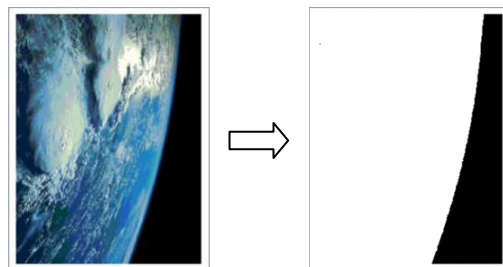


Fig. 8. Binarización de imagen.

Por lo que respecto al tiempo de procesamiento, obviamente será tanto mayor cuanto mayor sea la resolución (cantidad de píxeles de la imagen). Ahora bien, luego de la binarización, la relación entre la cantidad de píxeles blancos y negros de la imagen es independiente de la resolución hasta 480×640 píxeles, con lo que su influencia sobre la determinación del ángulo θ resulta despreciable. El tiempo que tarda el software implementado en procesar una imagen de 480×640 fue de 11.70ms, totalmente compatible con los requerimientos de la ley de control. Para evaluar el tiempo insumido por el algoritmo se utilizó la GPIO de la placa de desarrollo y un osciloscopio digital, programando cambios de estado de una salida GPIO entre el inicio y la finalización del procesamiento de imagen.

C. Imágenes de Alta Luminosidad

Un problema adicional que debió ser abordado es el relacionado con la persistencia de imágenes de alta luminosidad. Se plantea como interrogante si, al pasar el Sol por el campo visual de la cámara digital y ser fotografiado, puede producirse una persistencia de la imagen en la matriz de CCD debido a su elevada luminosidad. Aunque en el espacio cercano el Sol subtende un ángulo de 0.5° , si su imagen persistiera excesivamente en la matriz sensible de la cámara, podría engañar al sistema de control con una señal de falso horizonte.

Para responder al interrogante precedente, en una primera aproximación cualitativa, se diseñó un montaje experimental que permite simular el pasaje del Sol ante la cámara, aislando a ésta de toda otra fuente de iluminación, cuyo diagrama conceptual es el de la Figura 8.

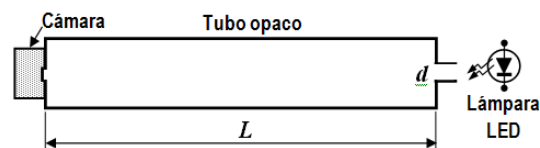


Fig. 8. Medición de persistencia: esquema.

Se trata de un tubo de PVC $\phi 110$ mm pintado en su interior de color negro mate para evitar reflexiones, cerrado en sus extremos, en uno de los cuales se ha montado la cámara digital y en el otro una lámpara LED que ilumina al interior del tubo

a través de un agujero de diámetro $d=12$ mm, con $L=1.153$ mm, cumpliéndose la relación $d/2L = \tan(0.25^\circ)$. La lámpara LED utilizada posee una temperatura de color de 6000 K muy cercana a la luz solar (5800 K). El nivel de iluminación que produce el sol a 100 Km de altura sobre la superficie terrestre es de unos 125.000 lux; habida cuenta que el rendimiento lumínico de una lámpara LED de alta calidad se encuentra en el orden de 100 a 130 lm/W, la potencia requerida para simular realísticamente la iluminancia solar en nuestro montaje resultaría prohibitiva. Es por ello que se ha optado por una simulación cualitativa con una iluminancia incidente en la cámara del orden de los 500 lux.

El encendido/apagado de la lámpara se controló con un dispositivo IGFET excitado desde una de las salidas GPIO de Raspberry, mientras que otra salida se empleó para señalar la toma de imagen. Mediante un osciloscopio se registraron ambas señales para determinar con precisión el tiempo entre las mismas. El programa de comando permitió la toma de fotografías sucesivas: el primero con LED encendida, y el siguiente con la lámpara apagada, estando ambas tomas separadas por un retardo de tiempo de duración variable. En las condiciones indicadas se llevaron a cabo una serie de mediciones, pudiéndose establecer que la persistencia máxima de la imagen es de 47 milisegundos.

De ninguna manera puede establecerse una relación de proporcionalidad entre la persistencia determinada a 500 lux y la que ocurriría para una iluminancia 250 veces mayor (sol directo); sin embargo el experimento cualitativo realizado ha permitido identificar una situación de riesgo que debe ser evitada. Ello conduce a recomendar fuertemente el empleo de algún inhibidor solar para evitar que la cámara tome imágenes directas del sol.

V. CONCLUSIONES

El presente trabajo ha contribuido a confirmar que el uso de single board computers junto con cámaras digitales comerciales de bajo costo pueden integrar ventajosamente (desde un punto de vista económico) un sistema de control de actitud angular de precisión moderada, utilizable en cargas útiles suborbitales.

Se pudo verificar que el software desarrollado para la binarización de la imagen, utilizando un sistema embebido, permite procesar una imagen de baja resolución (480x640) en un tiempo adecuado para su utilización en el sistema de control (11.70ms), validando de esta manera la posibilidad de

usar el hardware seleccionado para una aplicación control de actitud angular.

Se ha podido determinar la importancia práctica de la persistencia de imágenes de elevada luminosidad como las que se generan enfocando directamente al sol, por lo que en futuros desarrollos se deberá diseñar, construir y ensayar algún elemento o dispositivo inhibidor solar (basado –por ejemplo– en fotodiodos).

En la futura evolución del proyecto, se planea utilizar una maqueta de la carga útil, con un sistema de eyección de gas similar a los utilizados en vuelo, montada sobre una mesa con rodamiento de aire de un grado de libertad, en la que se instalará el hardware y la cámara discutidos en este trabajo para realizar simulaciones semi-reales a fin de verificar el funcionamiento del conjunto.

AGRADECIMIENTOS

Los autores desean agradecer a las autoridades de las Facultades Regionales La Rioja y Córdoba y al Centro de Investigaciones Aplicadas (CIA) de la Fuerza Aérea Argentina por el apoyo recibido, como asimismo reconocer el asesoramiento y la colaboración de los Ings. Juan P. Pedroni (director), Juan J. Gaspanello (becario graduado) y del Sr. Luis G. Hintz (becario alumno), también integrantes del equipo de este proyecto.

REFERENCIAS

- [1] Makovec K.L. (2001). A Nonlinear Magnetic Controller for Three-Axis Stability of Nanosatellites. Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, July 2001.
- [2] Makovec K.L., Turner A.J., Hall C.D., (2001). "Design and Implementation of a Nanosatellite Attitude Determination and Control System". Advances in the Astronautical Sciences, Astrodynamics 2001. Vol 109, Part I: 167-186.
- [3] Meller D., Sripruetkiat P., Makovec K.L., (2000). "Digital CMOS Cameras for Attitude Determination". 14th AIAA/USU Conference on Small Satellites. Enlace <http://digitalcommons.usu.edu/smallsat/2000/AII2000>
- [4] Cova W.J.D., González G.J., Pedroni J.P., Turra D.N., (2016). "Cámaras Digitales como Sensores de Actitud – Formulación de Requerimientos". X Jornadas de Ciencia, Tecnología y Arte Científico. Univ. Nac. de La Rioja – Ciudad de La Rioja, 31/10 al 3/11 2016.
- [5] Hintz L.G., Oviedo Codigoni C., Zimmermann Y.E., Cova W.J.D., Pedroni J.P., (2016). "Cámaras Digitales de bajo costo como Sensores de Actitud en Aplicaciones Suborbitales". AADECA 2016 – 25° Congreso Argentino de Control Automático. Buenos Aires, Argentina, 1 al 3/11/2016.
- [6] NASA National Aeronautics and Space Administration, imágenes públicas disponibles en www.nasa.gov/multimedia/imagegallery/

Sintetizador Digital de Audio con Filtro, Efectos y Comunicación por USB bajo LPCXpresso y FreeRTOS

Axel Lucas Gómez Caamaño (*Autor*)

Técnicas Digitales II, Departamento de Electrónica
UTN Facultad Regional Avellaneda
Buenos Aires, Argentina
gomezaxel.lucas@gmail.com

Hernán Matías Trinidad (*Autor*)

Técnicas Digitales II, Departamento de Electrónica
UTN Facultad Regional Avellaneda
Buenos Aires, Argentina
hernantri@gmail.com

RESUMEN

Utilizando el microcontrolador ARM de la familia Cortex M3 LPC1769 (NXP), o comúnmente denominado LPCXpresso, se desarrolló un sintetizador de audio digital bajo el sistema operativo embebido FreeRTOS. El sintetizador cuenta con un teclado de 2 octavas y $\frac{1}{2}$ de 32 teclas plásticas organizadas circuitualmente en una matriz de 8 filas x 4 columnas, 6 encoders, 7 botones y un display LCD de 128x64px.

Las funciones que posee el instrumento son:

- Generar una señal de acuerdo a la forma de onda seleccionada (cuadrada, senoidal, triangular o diente de sierra), a una frecuencia determinada por la tecla según la escala musical.
- Aplicarle el filtro de frecuencia seleccionada y con la posibilidad del agregado de diferentes efectos: overdrive, attack y delay.
- Replicar el comportamiento desde un aplicativo para PC (en Windows 7).

PALABRAS CLAVE

Sintetizador digital de audio, procesamiento digital de señales, Sistema embebido FreeRTOS, Microcontrolador Cortex M3, formas de onda, filtros pasa bajos, pasa altos, pasa banda, distorsión “overdrive”, “attack”, “delay”, Comunicación USB.

INTRODUCCIÓN

Un sintetizador de sonido es un instrumento musical electrónico capaz de generar, mezclar y filtrar distintas señales de sonido con formas de onda tales como cuadrada, senoidal, triangular y diente de sierra en diferentes frecuencias que pueden modificarse mediante el uso de filtros y otros efectos,

encontrando así distintos sonidos, resultando en nuevos timbres o imitando los timbres de otros instrumentos.

Este sintetizador se construye reutilizando la carcasa de un teclado electrónico comercial. Se utiliza el SO en tiempo real FreeRTOS corriendo en un microprocesador cortex M3, con capacidad para manejar diversas entradas (teclado, encoders, botones), controlar un display, generar ondas analógicas e implementar filtros digitales FIR.

SINTETIZADOR



Fig. 1 Imagen del Sintetizador digital de audio

Su diseño semi-compacto de 50 centímetros de largo y 7 centímetros de espesor. Puede ser alimentado con una fuente de tensión entre 7V y 14V de continua posee un consumo máximo de 1,8 Watt.

DIAGRAMA EN BLOQUES

Tal y como indica la Fig. 2, el microcontrolador del sintetizador tiene de entradas al teclado matricial, a los encoders de selección de parámetros, botones y selector de formas de onda, de selección de filtros y selección de frecuencias de corte de los filtros. Por otro lado, tiene de salidas a: parlante vinculado al convertor DAC, el display por comunicación paralela y comunicación USB bidireccional (Low Speed).

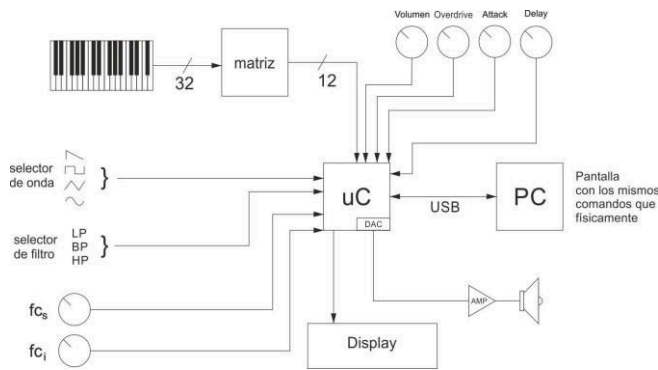
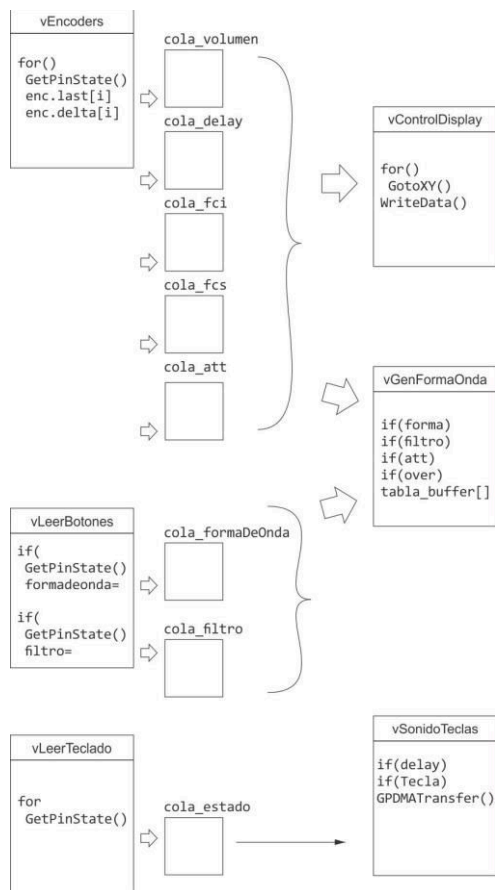


Fig. 2 Diagrama en bloques del Sintetizador digital de audio

Las herramientas utilizadas para vincular al sistema en su conjunto fueron las bibliotecas provistas por el fabricante (NXP) y el sistema operativo FreeRTOS como capa de abstracción superior.

FREERTOS

FreeRTOS es un sistema operativo embebido en tiempo real. Se caracteriza por su programación en tareas. Estas tareas son comandadas por un "Scheduler" el cual se encarga de decidir la tarea que debe ser ejecutada por el microcontrolador.



En la figura anterior podemos observar las tareas que producen datos, a la izquierda, y, a la derecha, las tareas que los utilizan y generan una respuesta. El medio de comunicación entre las tareas son las denominadas colas, donde se guardan y leen los datos. Las tareas se sincronizan mediante el uso de funciones provistas por el sistema operativo.

FUNCIONAMIENTO

Las entradas del teclado matricial, los encoders y los botones están conectadas a los pines del micro configurados como GPIO. Estas entradas se sensan cada un tiempo determinado suficiente para una buena detección (actualmente se registran cada un (1) milisegundo).

Las formas de onda y los coeficientes de los filtros (FIR) se encuentran en tablas constantes. Al iniciar se carga una configuración por defecto en una tabla que funciona como buffer de salida, cuando se cambia la configuración se modifican los valores de dicha tabla en base a lo modificado.

Al momento de tocar una tecla se ejecuta el DMA (acceso directo a memoria) que modifica los valores del DAC (convertor analógico digital) generando la señal que luego se amplifica con un integrado externo al microcontrolador.

Para filtrar la señal se utilizan funciones de DSP (Procesador Digital de Señales) de las bibliotecas de CMSIS que se incluye en la documentación del LPCXpresso. Los efectos son generados por medio de una tabla utilizada como buffer que se modifica según la configuración deseada. El volumen se controla mediante un encoder (0 a 1) y se lo multiplicará por el valor de la señal original, dando como resultado una señal regulada en amplitud.

A su vez en el display se pueden observar los valores de los parámetros que se están modificando, ya sea: la señal, el volumen, los filtros y/o los efectos.

ONDAS GENERADAS

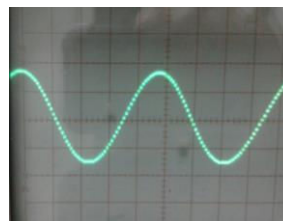


Fig. 4

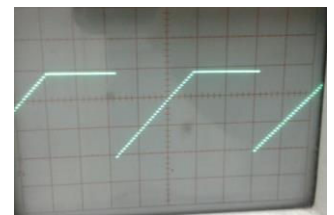


Fig. 5

El gráfico de la figura 4 es de una onda senoidal de 440Hz. La figura 5 muestra una onda diente de sierra de aumentada en amplitud pero con los picos recortados provocando así el efecto de distorsión, esto genera, consecuentemente, distorsión armónica.

REFERENCIAS

- [1] NXP Semiconductors - web: <http://www.nxp.com/>
- [2] FreeRTOS (Real Time Operating System) - web: <http://www.freertos.org/>

Reconocimiento Embebido de Habla Aislada Independiente del Hablante en Tiempo Real con HMMs y SVMs

Mariano Marufo da Silva
UTN-FRBA
mmarufodasilva@est.frba.utn.edu.ar

Diego A. Evin
CONICET-UBA
diegoevin@gmail.com

Abstract—Este trabajo describe el diseño e implementación embebida de algoritmos para el reconocimiento del habla basados en Modelos Ocultos de Markov (HMMs) y Máquinas de Vectores de Soporte (SVMs).

Se desarrolló un sistema para el reconocimiento automático de palabras aisladas independiente del hablante en tiempo real, basado en un microcontrolador de 32 bits ARM Cortex-M4F y se comparó su desempeño con los de un trabajo de investigación previo.

I. RECONOCIMIENTO DE HABLA AISLADA

Representando las características de las palabras pronunciadas a partir de la secuencia de vectores de observaciones O , el objetivo del reconocimiento automático de palabras aisladas es el de resolver:

$$\arg \max_{1 \leq i \leq V} (P(w_i|O)) \quad (1)$$

donde V es la cantidad de palabras en el diccionario del sistema y cada w_i es la palabra número i del mismo. Aplicando el teorema de Bayes obtenemos:

$$P(w_i|O) = \frac{P(O|w_i) \cdot P(w_i)}{P(O)} \quad (2)$$

Es decir que para un set de probabilidades $P(w_i)$ iguales, la palabra más probable de haber sido pronunciada depende sólo de $P(O|w_i)$.

II. MODELOS OCULTOS DE MARKOV

El habla puede representarse por un modelo estadístico compuesto por estados con transiciones probabilísticas entre ellos y funciones de densidad de probabilidad de emisión de distintos sonidos. En el reconocimiento de habla basado en HMMs cada modelo acústico (en este trabajo se consideró como modelos acústicos del sistema a las palabras a reconocer) se representa por un HMM.

Un modelo de Markov es una máquina de N estados finita que en cada instante t actualiza su estado, donde la transición del estado i al estado j se da a partir de la probabilidad discreta a_{ij} . Según el estado j en el que se encuentre en cada instante t genera un nuevo vector de observaciones o_t a partir de la densidad de probabilidad de salida $b_j(o_t)$ (figura 1).

El proceso de reconocimiento consiste en obtener la secuencia de vectores de observaciones O correspondiente a la

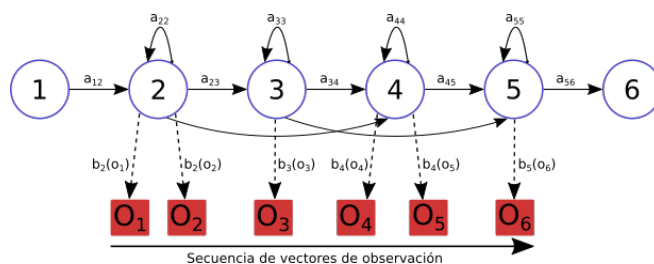


Fig. 1: Ejemplo de representación de un HMM.

palabra pronunciada, seguido por un cálculo de las verosimilitudes para cada uno de los modelos $P(O|w_i)$ para $1 \leq i \leq V$ y finalmente de la selección de la palabra cuyo modelo posea la máxima verosimilitud: $i^* = \arg \max_{1 \leq i \leq V} (P(O|w_i))$.

III. IMPLEMENTACIÓN EMBEBIDA

A. Scaling

El algoritmo de reconocimiento está compuesto por productorias de probabilidades, cuyos resultados tienden exponencialmente a 0. Este es un problema que genera *underflow* en muchos de los cálculos (especialmente teniendo en cuenta que el microcontrolador posee una FPU de precisión simple en lugar de doble), lo cual tiene como consecuencia una disminución en el grado de reconocimiento. Una mejora consiste en incorporar un procedimiento de escalamiento de las probabilidades a medida que se realizan los cálculos.

B. Probabilidades logarítmicas

Además del procedimiento de escalamiento, se decidió también implementar el algoritmo completo utilizando solamente probabilidades logarítmicas a lo largo del mismo. El principal inconveniente que se presentó al implementar esta solución, se dio al aplicarlo a una sumatoria de valores. En esos casos, se utilizó la siguiente aproximación:

$$\log \left(\sum f(x) \right) \approx \max(\log(f(x))) \quad (3)$$

IV. MÁQUINAS DE VECTORES DE SOPORTE

Las SVMs son modelos discriminativos que clasifican datos estimando hiperplanos de clasificación o separación de clases,

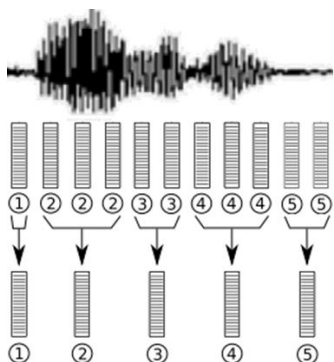


Fig. 2: Ejemplo de obtención del vector de largo fijo para SVM.

en lugar de modelar una distribución de probabilidad a partir de datos de entrenamiento, como lo hacen los HMMs.

El funcionamiento de las SVMs se basa en maximizar un margen: la distancia entre la frontera de clasificación y las muestras de entrenamiento. El margen indica cuánto ruido (uno de los principales problemas del reconocimiento de habla) agregado a muestras limpias es permitido en el sistema.

A. Sistema híbrido HMM/SVM

El poder de una representación HMM está en su habilidad de modelar la evolución temporal de la señal a través de un proceso Markov.

Por otro lado, las SVMs son clasificadores estáticos (los vectores con los que se trabaja deben tener todos la misma longitud), que tienen que ser adaptados para lidiar con la duración variable en las pronunciations del habla. Es por eso que se mantiene el framework HMM, trabajándose entonces con modelos híbridos HMM/SVM.

La forma de obtener una secuencia de vectores de observación de largo fijo a partir de la secuencia de habla de largo variable utilizando el framework HMM consiste en promediar aquellos vectores asociados (con mayor probabilidad) a los mismos estados (figura 2).

V. RESULTADOS

A. Resultados HMM

Los procesos de entrenamiento y testeo de los modelos HMM se realizaron utilizando una base de datos en español de 11 palabras (números 'cero' a 'diez') con 1089 audios para entrenamiento y 264 audios para testeo grabados en una cámara anecoica para trabajar con la mejor relación señal/ruido posible. Se testearon modelos con 8, 16 y 24 estados (tabla 1).

Si bien los mejores resultados en términos del Word Error Rate (WER) se obtuvieron utilizando modelos HMM con 16 estados y 4 mezclas cada uno, se eligió como tipo de modelo implementado en el prototipo final al de 16 estados y 2 mezclas, ya que la diferencia en el rendimiento es insignificante, mientras que el requerimiento de memoria es de la mitad.

Tabla I: Rendimiento de los modelos HMM [1].

Estados	Mezclas	Test [%]	WER [%]	Tamaño [KB]
8	2	88.64	11.36	57
	4	94.70	5.30	113
	8	93.56	6.44	226
	16	93.56	6.44	451
16	2	96.21	3.79	114
	4	96.59	3.41	228
	8	95.83	4.17	452
24	2	95.08	4.92	171
	4	95.83	4.17	340

Tabla II: Comparación de rendimientos: HMM vs. DTW [1].

Algoritmo	Test [%]	WER [%]	RTF ^a	Tamaño [KB]
HMM	96.21	3.79	0.37	114
DTW	65.22	34.78	0.54 ^b	111 ^c

^a El factor de tiempo real (RTF) se define como: $RTF = \frac{\text{Tiempo de reconocimiento}}{\text{Largo de la pronunciación}}$

^b Valor medio de varias repeticiones medidas.

^c Para templates de 0.69 segundos de largo.

B. Comparación HMM vs. DTW

La implementación propuesta fue comparada con un sistema que corre bajo la misma plataforma pero basado en Dynamic Time Warping (DTW) [2]. DTW es una técnica de *template matching* que realiza una alineación temporal entre dos pronunciations y calcula una medida de sus similitudes.

El rendimiento fue evaluado en términos de grado de reconocimiento (medido con el WER), velocidad (medida usando el RTF) y memoria requerida (tabla 2).

C. Resultados HMM/SVM

Los resultados para el modelo híbrido HMM/SVM fueron muy inferiores que para el caso HMM (tienen un WER más de cuatro veces mayor). Entonces, no se justifica su implementación embebida ya que su correspondiente algoritmo de reconocimiento consumiría mayores recursos del microcontrolador (en memoria de código y de datos), tendría un peor RTF y el WER resultante sería mayor.

VI. CONCLUSIONES

Una de las posibles razones de la reducción del rendimiento en el modelo híbrido HMM/SVM, respecto al caso HMM, puede haberse encontrado en el método de construcción de los vectores de largo fijo a partir de la secuencia de vectores de observación de largo variable. El hecho de promediar vectores de características elimina información potencialmente útil para el reconocimiento, que no termina siendo utilizada para discriminar entre una clase y otra por parte del clasificador SVM.

REFERENCES

- [1] Marufo da Silva, M., Evin, D. A., & Verrastro, S. (2016, November). *Speaker-independent embedded speech recognition using Hidden Markov Models*. In Ciencias de la Informática y Desarrollos de Investigación (CACIDI), IEEE Congreso Argentino de (pp. 1-6). IEEE.
- [2] Alvarez, A. G., Evin, D. A., & Verrastro, S. (2016). *Implementation of a Speech Recognition System in a DSC*. IEEE Latin America Transactions, 14(6), 2657-2662.

Implementación de síntesis de voz en la plataforma Intel Galileo

Andrés Tapari, Santiago José Martínez Larroza, Felix Matías Taborda, Guillermo Ariel Sanchez, Jesús Exequiel Benavídez, Juan Ignacio Moragues y Enrique Sergio Burgos
Universidad Tecnológica Nacional
Facultad Regional Paraná
Paraná, Entre Ríos
sergioburgos@frp.utn.edu.ar

I. INTRODUCCIÓN

La plataforma Intel Galileo [1] consta de diferentes placas de desarrollo (hasta el momento se han comercializado dos modelos distinguidos como generación 1 y generación 2) basadas en un *System on Chip (SoC)* denominado *Intel® Quark™ X1000*, con una frecuencia de reloj de 400 Mhz e incorpora un conjunto de instrucciones compatible con *Intel® Pentium®* de 32 bits [2].

En lo que a software respecta, existen diferentes alternativas y lenguajes de desarrollo para esta plataforma. En esta implementación se optó por el uso "*Intel® System Studio IoT Edition*" como entorno de desarrollo ya que este incorpora un IDE basado en Eclipse, todas las herramientas necesarias para el desarrollo de aplicaciones en lenguaje C y C++ así como un conjunto de librerías precompiladas. Entre las librerías que se cuentan disponibles cabe mencionar *mraa* [3], que permite realizar operaciones de entrada salida sobre la placa, y *openCV* para realizar diferentes tipos de procesamiento sobre imágenes.

El objetivo de este trabajo fue agregar capacidades de síntesis de voz a la plataforma Galileo utilizando la librería *eSpeak* [4], de modo que el sistema fuese capaz de informar al usuario el resultado de mediciones y/o procesamiento de modo audible.

II. LA LIBRERÍA ESPEAK

La librería *eSpeak* permite la síntesis de voz en diferentes idiomas (entre ellos el castellano) caracterizándola de diferentes maneras (por ejemplo según el sexo o la edad). Además es multiplataforma pudiéndose utilizar tanto, en diferentes distribuciones de sistemas Linux, como en Microsoft Windows, existiendo versiones compiladas para esta arquitectura. Este último detalle, su facilidad de uso y reducido tamaño hicieron que fuese la opción elegida para la implementación.

Debido a que esta librería no se encuentra incorporada en el paquete de software del sistema de desarrollo de Intel Galileo, su uso requiere la realización de tareas adicionales al desarrollo de las aplicaciones clásicas para esta plataforma. Por esto, a continuación abordaremos el proceso de construcción de aplicaciones, la forma de incorporación de esta librería al sistema y los resultados obtenidos.

III. PROCESO DE CONSTRUCCIÓN DE APLICACIONES

Cuando se utiliza "*Intel® System Studio IoT Edition*" como entorno de desarrollo, la placa sobre la que se ejecutará la aplicación deberá iniciar desde una imagen cargada en una tarjeta de memoria micro SD. Esta imagen contiene un conjunto de librerías que se corresponde con las librerías utilizadas en el proceso de compilación de la aplicación. De este modo, cuando se construye una aplicación desde un equipo PC, el binario generado es transmitido a la placa, se almacena en la tarjeta micro SD para luego ser ejecutado a través de una sesión de interfaz de comando seguro (*ssh*). El proceso de compilación utilizado se denomina compilación cruzada [5], ya que, desde un equipo (*PC host*) se construyen aplicaciones para ser ejecutadas en otro sistema (*target*).

Si analizamos detalladamente el proceso de desarrollo de la aplicación observaremos que, para construir una aplicación que utilice una librería particular, es necesario que los *headers* asociados a la librería se encuentren disponibles el equipo *host* y que la librería de carga dinámica esté disponible en el sistema destino (*target*).

Además, tanto el binario de la aplicación generado en el equipo PC como las librerías utilizadas en la placa Galileo, no deben poseer ninguna referencia a las aplicaciones y librerías disponibles en el equipo *host*. De tener alguna relación la aplicación no podrá ejecutarse. Ya sea por no poder satisfacer las dependencias o porque estas pueden no estar compiladas para la arquitectura donde se ejecutará la aplicación.

En este contexto, agregar una nueva librería al sistema de desarrollo en el *host* y al espacio de ejecución en el *target*, requiere la realización de un conjunto pasos no triviales.

eSpeak utiliza la librería *libportaudio* para acceder al sistema multimedia. Este acceso lo realiza a través de la interfaz provista por *ALSA (Advanced Linux Sound Architecture)*, la cual forma parte del sistema utilizado por la plataforma Galileo. Para construir ambas se utilizó una técnica conocida como *change root*. Ésta consiste en mover temporalmente el directorio raíz del PC *host* a un directorio donde se encuentran las herramientas de desarrollo. Al hacer esto, el contenido del sistema de archivos principal se oculta del entorno de compilación, pero es necesario crear una estructura de directorios con algunas herramientas básicas.

IV. ENTORNO DE DESARROLLO

Las herramientas de desarrollo para construir aplicaciones para el SoC X1000 pueden encontrarse en las carpetas `iss-iot-linux/devkit-x86/sysroots/quark-wrs-linux/` y `/iss-iot-linux/devkit-x86/sysroots/i586-poky-linux/` del directorio donde se halla instalado “Intel® System Studio IoT Edition”. Será necesario copiar el contenido de ambas al directorio donde se realizará la compilación de las librerías. Además de esto, deberemos crear algunas carpetas adicionales (*root*, *tmp* y *dev*) necesarias para que el entorno de desarrollo pueda funcionar.

Una vez creada la estructura de directorios anteriores, es necesario crear un dispositivo null dentro de la carpeta *dev* utilizando el comando `mknod (mknod dev/null c 1 3)`. Este dispositivo funciona como una fuente y un sumidero siendo utilizado por diferentes herramientas durante el proceso de compilación. Con esto, se completan las tareas previas al proceso de compilación.

El siguiente paso es construir *libportaudio* y *eSpeak*. Para esto se deben descomprimir los paquetes de código fuente dentro de la carpeta seleccionada, construir las aplicaciones y realizar el cambio temporal del directorio raíz ejecutando el comando `chroot <nombre_carpeta> /bin/bash`. Una vez hecho esto, el proceso de construcción de las librerías continúa de modo usual. Solo debe considerarse, por una cuestión de comodidad, establecer como directorio de destino para las librerías una ruta que se encuentre vacía. Así se podrá identificar claramente los archivos que resulten del proceso de compilación. En nuestro caso, utilizamos como directorio de destino `/opt/snd`. Nótese que esta ruta será relativa al directorio donde se encuentran las herramientas de compilación y no sobre el directorio raíz del sistema *host*.

Tras haber finalizado la construcción de las librerías es necesario realizar dos tareas. Por un lado la configuración del entorno de trabajo para que haga uso de las mismas y por otro copiar las librerías al sistema destino.

La primera tarea debe realizarse una vez creado un nuevo proyecto en Eclipse, desde las propiedades del mismo. Indicando la ruta donde se encuentran los *headers* asociados a las librerías, pero en este caso indicando la ruta completa respecto del sistema de archivos del equipo PC *host* (`<ruta_completa>/opt/snd/include`). Luego, de modo similar es necesario indicar la carpeta donde se encuentran las librerías resultante de la compilación también estableciendo la ruta absoluta dentro del sistema de archivos del PC *host* (`<ruta_completa>/opt/snd/lib`).

Por último, es necesario agregar los nombres de las librerías con las que debe enlazarse la aplicación (*portaudio* y *espeak*). Con esto finalizamos la configuración del proyecto para que haga uso de la nueva librería, nuestra aplicación podrá ahora hacer uso de las funciones exportadas por ellas.

Para que la aplicación que se desarrolle pueda ejecutarse en el equipo destino es necesario que se cuente con las librerías, para esto es necesario copiar el directorio donde se instalaron las librerías (en nuestro caso `/opt/snd`) al directorio raíz del sistema destino. Finalmente, es necesario asignar a la variable `LD_LIBRARY_PATH` este valor y exportarla en el espacio de

ejecución del sistema *target*. De este modo, la ruta indicada formará parte de la ruta de búsqueda de las bibliotecas para las aplicaciones que allí se ejecuten.

V. CONCLUSIONES

Con el objetivo de evaluar el rendimiento y consumo de recursos de la librería, se implementó una aplicación que repetía una frase predeterminada. Durante la ejecución de la misma se observó, utilizando el comando *top*, un consumo de recursos del procesador que osciló en las diferentes pruebas entre un 11 % y 13 %.

En lo que respecta al consumo de memoria RAM este fue evaluado utilizando *Valgrind* y la herramienta *massif* (Fig. 1). El consumo máximo de memoria alojada (*heap*) fue de 643,9 KB, estando entre 462 KB y 476 KB la memoria alojada por la librería *eSpeak*.

Por su parte, el conjunto de archivos requeridos para utilizar la librería *eSpeak* ocupó 3.1 MB y los archivos asociados a *libportaudio* 1.1 MB del espacio de almacenamiento disponible en la tarjeta de memoria del sistema.

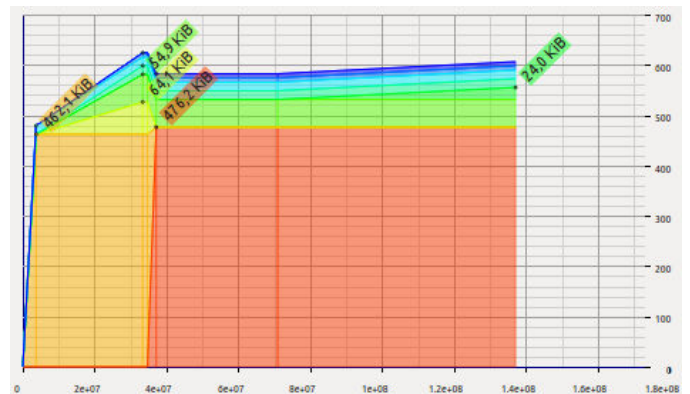


Figura 1: consumo de memoria RAM de los diferentes módulos que componen la aplicación.

En lo que respecta al proceso utilizado para incorporar librerías externas al entorno de desarrollo y de ejecución, constituye una ventaja frente a la alternativa de construir una nueva imagen completa. Además, abre la posibilidad a utilizar librerías no incluidas en el sistema *target* de modo simple y rápido.

REFERENCIAS

- [1] “Intel Galileo Board” Url: <https://software.intel.com/en-us/iot/hardware/galileo>. Último acceso 05/2017
- [2] “Datasheet Intel Galileo Gen 2 Development Board”, Url: <http://www.intel.com/content/dam/www/public/us/en/documents/datasheets/galileo-g2-datasheet.pdf>. Último acceso 05/2017
- [3] “Linux Library for low speed IO Communication in C”, Url: <http://mraa.io>. Último acceso 05/2017
- [4] “eSpeak text to speech”, Url: <http://espeak.sourceforge.net/>, Último acceso 05/2017
- [5] Hallinan, C. (2007). Embedded Linux primer: a practical, real-world approach, capítulo 12. Pearson Education India.

Medidor Bidireccional Inteligente Para Redes Electricas Interconectadas

Barrio Chaud, Nicolaz Suarez

Laboratorio de Electrónica
FMA. Universidad Católica de Santiago del Estero
Santiago del Estero, Republica Argentina
electrolab@ucse.edu.ar

Cesar Reynoso

Laboratorio de Electrónica
FMA. Universidad Católica de Santiago del Estero
Santiago del Estero, Republica Argentina
electrolab@ucse.edu.ar

Resumen- Se presenta el estado de avance en el diseño e implementación de un medidor de energía bidireccional con aplicaciones en redes de distribución eléctrica interconectadas. El medidor bidireccional realiza el registro de la energía generada por cada planta generadora y su diferencia neta, transfiriendo estos datos a un sistema centralizado de control y facturación utilizando enlaces GSM/SMS (Global System for Mobile Communications) y GPRS (General Packet Radio Service) [1]. Entre las prestaciones adicionales previstas se contemplan la medición de los parámetros eléctricos distorsión armónica y factor de potencia, así como el control remoto de corte y reconexión de la energía eléctrica por parte de la empresa prestaría del servicio de distribución. El diseño está basado en un entorno de desarrollo STM32F4 DISCOVERY, con unidad microcontroladora STM32F407VGT6 (ARM 32-bit, Cortex M4) [2], en el cual se implementó el software embebido con las funciones principales de cálculos de energías, parámetros de calidad eléctrica, control de corte y reconexión eléctrica, utilizando módulos periféricos para la comunicación GSM/GPR. El estado actual implementa la medición de la energía y el enlace remoto, estando en fase de desarrollo los algoritmos de cálculo de distorsión armónica, factor de potencia y otras prestaciones adicionales que se visualizaron durante la marcha del proyecto para dar mayor valor operativo al sistema, tal como un lector de tarjeta para sistemas prepagos de energía.

Índice de Términos— Energía, Medición, Sistemas Interconectados. Software Embebido.

I. INTRODUCCIÓN

La generación de energía alterna a partir de un sistema fotovoltaico integrado a la red de distribución eléctrica domiciliaria, requiere de un sistema de medición bidireccional que contemple el balance neto de la energía circulante con fines de facturación o de reconocimiento económico para el caso de aporte de energía por parte del cliente a la red de consumidores. Simultáneamente a este requisito básico, se le deben sumar funciones de monitoreo de variables relacionadas con la calidad del servicio eléctrico, principalmente por parte del cliente generador interconectado a la red, como el

contenido armónico y el factor de potencia, así como también la estabilidad en frecuencia y valor RMS de la onda generada, entre otros muchos parámetros a tener en cuenta. En el desarrollo aquí presentado, se utilizó a modo experimental un montaje basado en dos sistemas fotovoltaicos con convertidores DC-AC a 220 V y 50 Hz, interconectados entre sí entregando energía en forma conjunta a una misma carga. Este montaje experimental obedece a que se utilizan convertidores DC-AC de diseño propio (con enganche de fase PLL) [3], no homologados para su interconexión con la red eléctrica pública, entonces uno de ellos cumple el rol de dicha red a los fines de verificación y ensayos del prototipo del medidor bidireccional.

II. DESARROLLO

El medidor bidireccional fue diseñado con las siguientes especificaciones: conexión monofásica bifilar, 220 V, 50 Hz, 30 A máximos, CLASE 1, con funciones de medición de energía activa (kilovatios-hora), energía reactiva (kilovares-hora), y saldo neto de consumo energético por periodos específicos. A partir de estas prestaciones se implementó la comunicación inalámbrica utilizando tecnología GSM/GPRS por medio de un módulo SIM900 [4] en asociación con una SIM CARD de 6 pines bajo norma GSM 11.14, con la cual es posible disponer de información remota sobre datos del consumo y análisis de la red, como así también la posibilidad de realizar el corte y reconexión del servicio según corresponda. El medidor es descripto en la figura 1, en la cual el sistema de desarrollo STM32F4 DISCOVERY constituye el soporte de hardware para el software embebido, el cual realiza las funciones de cálculo de energías mediante el sensado de corrientes en la red e inversor de onda completa DC-AC por los respectivos sensores ACS712 de efecto Hall [5]. En este desarrollo los cálculos de valores RMS para las ondas adquiridas se realizan en forma algorítmica respetando la forma de onda no sinusoidal pura dado la presencia de contenido armónico, siendo una alternativa en estudio el uso de circuitos medidores específicos de valores eficaces verdaderos lo cual llevaría a una simplificación del software, pero con un aumento relativo del costo del producto final. El

control de conexión y reconexión de la energía de la empresa prestataria se implementa mediante el comando de un circuito simple a tiristor, según la orden recibida en forma remota. Al lado derecho de la figura 1, se indica el sistema de generación fotovoltaica con inversor DC-AC a 220 V, 50 Hz, el cual

utiliza el sensado de red para el control de fase mediante un PLL, como se explicó anteriormente este convertidor no posee homologación para su interconexión a la red pública por lo cual a nivel ensayos se procedió a utilizar un segundo convertidor en reemplazo de la red domiciliaria.

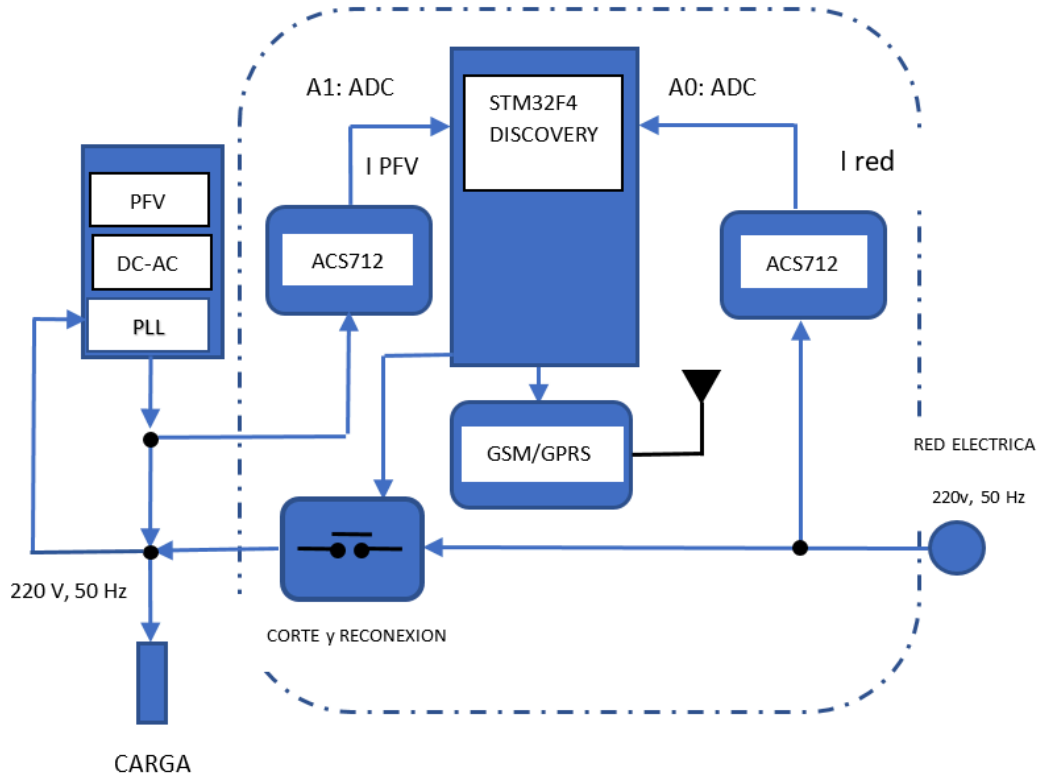


Fig. 1. Diagrama en Bloques del medidor bidireccional

III. ENSAYOS Y RESULTADOS

El medidor bidireccional aquí presentado está en proceso de desarrollo, habiéndose diseñado acorde a las especificaciones enunciadas, los ensayos preliminares cumplen con las mismas en cuanto al cálculo de energía activa y enlace remoto. En la continuidad del prototipo se trabaja a nivel algorítmico para la implementación de la transformada rápida de Fourier FFT (Fast Fourier Transform) lo que permitirá la determinación del contenido armónico, el cálculo del factor de potencia y otras funciones adicionales.

IV. CONCLUSIONES

Se presentó un avance sobre el desarrollo de un medidor bidireccional inteligente con conectividad remota. Se pretende lograr un prototipo experimental, con propósitos didácticos y evaluación sobre su potencial uso a nivel comercial acorde al costo y prestaciones comparativas a productos ya disponibles en el mercado. Si bien se desarrolló en torno al sistema

STM32F4 DISCOVERY, los algoritmos son extrapolables a otras plataformas como la EDU-CIAA u otras versiones de microcontroladores de 32 bits, con módulos DSP, dados los requerimientos de procesamiento. y seguimiento del punto máxima potencia para los paneles fotovoltaicos.

REFERENCIAS

- [1] Dogan Ibrahim, Ahmet Ibrahim. Microcontroller Based GSM/GPRS Projects. Editor VDM Verlag. 2010
- [2] http://www.st.com/content/ccc/resource/technical/document/user_manual/1/70/fe/4a/3f/e7/e1/4f/7d/DM00039084.pdf/files/DM00039084.pdf/jcr:content/translations/en.DM00039084.pdf. User manual Discovery kit with STM32F407VG MCU. 2016.
- [3] William F. Egan. Phase-Lock Basics Wiley-IEEE Press. 2007.
- [4] SIMCOM – SIM TECH ftp://imall.iteadstudio.com/IM120417009_IComSat/DOC_SIM900_Hardware%20Design_V2.00.pdf. 2016.
- [5] ALLEGRO. <https://www.sparkfun.com/datasheets/BreakoutBoards/0712.pdf>. Allegro Microsystems. 2007.

Lazo de Enganche de Fase para Convertidores Fotovoltaicos Integrados a Redes Electricas

Jorge Omar Perez , Hilda Noemi Ferrao

Laboratorio de Procesamiento Digital de Señales
DEEC. FACET. Universidad Nacional de Tucumán
República Argentina
jperez@herrera.unt.edu.ar, lpdi@herrera.unt.edu.ar

Gustavo Eduardo Juarez, Ruben del Valle Fadel
Laboratorios de Inteligencia Artificial y Control

DEEC. FACET. Universidad Nacional de Tucumán
República Argentina
gjuarez@herrera.unt.edu, rfadel@herrera.unt.edu.ar

Resumen- En este trabajo se presenta la implementación embebida de un lazo de enganche de fase (PLL), para aplicaciones de seguimiento de fase, amplitud y frecuencia en sistemas de energía fotovoltaica con convertidores DC/AC interconectados a redes de distribución eléctrica. Se plantea el desarrollo de una versión PLL digital totalmente embebida utilizando la plataforma de hardware EDU-CIAA-NXP con microcontrolador LPC4337, con realimentación opto acoplada de la señal de entrada, tratándose de un diseño basado principalmente en métodos de procesamiento y control digital de señales. Se debe considerar que el PLL forma parte de un sistema integral de conversión fotovoltaica en desarrollo en una única plataforma EDU-CIAA, por lo cual el PLL servirá como módulo de control del convertidor DC/AC que utiliza SPWM (Modulación de Ancho de Pulso sinusoidal) embebido para sintetizar la señal de salida en potencia mediante módulos MOSFET. El estado de avance al momento de esta presentación es a nivel de ensayos preliminares y programación parcial en el entorno IDE de la EDU-CIAA.

Índice de Términos— PLL, DC/AC, Filtros Digitales. Software Embebido.

I. INTRODUCCIÓN

Los generadores de tensión alterna acoplados a un nodo común están exigidos de cumplir con un estricto seguimiento de los parámetros de la red eléctrica principal, amplitud, frecuencia y sincronismo de fase a la cual aportan eventualmente energía. Es necesario un mecanismo automático de seguimiento y control en tiempo real que garanticen estas especificaciones. Entre los diferentes métodos empleados para este fin se pueden considerar los detectores de cruce por cero (detección para el sincronismo de fase) con reguladores por lazo realimentado de control, o en un modo más avanzado utilizar un mecanismo de control por PLL [1]. A partir de estas premisas se procedió a desarrollar en forma algorítmica un sistema PLL embebido en la plataforma EDU-CIAA-NXP [2] a modo de bloque de sincronismo para un convertidor DC/AC [3] monofásico de 220 V, 50 Hz acoplado a la red eléctrica. El diseño contempla el uso de un filtro pasa bajos digital FIR [4] de orden 10, y la utilización de un

algoritmo de Goertzel [5] para la detección de ausencia de energía en la frecuencia de red, para la activación de alarmas y salida de servicio del convertidor DC/AC.

II. MARCO TEORICO DE REFERENCIA

Un sistema PLL se compone de los bloques mostrados en la figura 1, en la cual se pueden ver tres bloques principales: a) un filtro pasa bajos destinado a recuperar la señal de error de fase, b) una etapa proporcional integral generadora de las señales de control y c) un oscilador controlado por tensión VCO [6] a modo de actuador. La totalidad del sistema se comporta en modo de regulador de lazo cerrado para enclavar la fase de salida en coincidencia con la fase de la señal de referencia (RED).

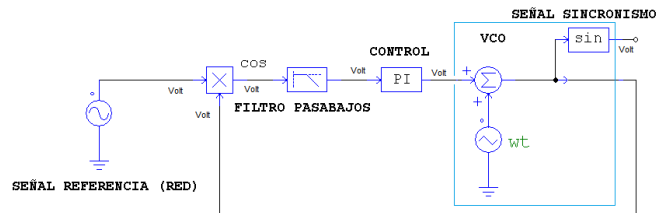


Fig. 1. PLL Diagrama Operativo

Considerando el multiplicador de entrada en el lazo de realimentación se puede encontrar algébricamente que la señal de entrada V_M al filtro pasa bajos es la dada por la ecuación 1, en la cual la señal de referencia es $V_i = A \sin(\omega_i t + \theta_i)$ y la de sincronismo o salida es $V_o = B \sin(\omega_o t + \theta_o)$.

$$V_M = A \sin(\omega_i t + \theta_i) \cdot B \sin(\omega_o t + \theta_o)$$

$$V_M = A \cdot B [\cos(\alpha) - \cos(\beta)] \quad (1)$$

$$\alpha = (\omega_o - \omega_i)t + (\theta_o - \theta_i)$$

$$\beta = (\omega_o + \omega_i)t + (\theta_o + \theta_i)$$

En la ecuación 1, el termino α contiene la señal de error de desfase, mientras que el termino β es información que debe ser eliminada a los fines del PLL, tal es la función del filtro pasa bajos el cual posee una frecuencia de corte superior por debajo de la frecuencia de red de 50 Hz. El bloque de controlador PI genera la señal de control para el oscilador VCO con lo cual se cierra el lazo de control y el efecto de seguimiento buscado. En la figura 2 se muestra una simulación del proceso descrito.

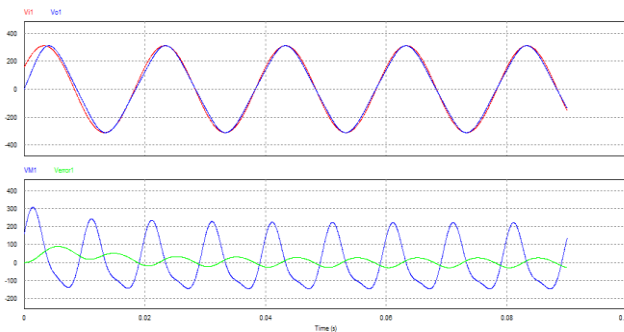


Fig. 2. Señales en el PLL, seguimiento y error de Fase

III. DISEÑO Y DESARROLLO DEL PLL EMBEBIDO

A los fines del presente trabajo el PLL implementado en forma aislada del resto del convertidor DC/AC, se corresponde a la configuración mostrada en la figura 3, utilizando como hardware la EDU-CIAA-NXP, un optoacoplador lineal IL300, y una red resistiva de realimentación para la salida de sincronismo V_o .

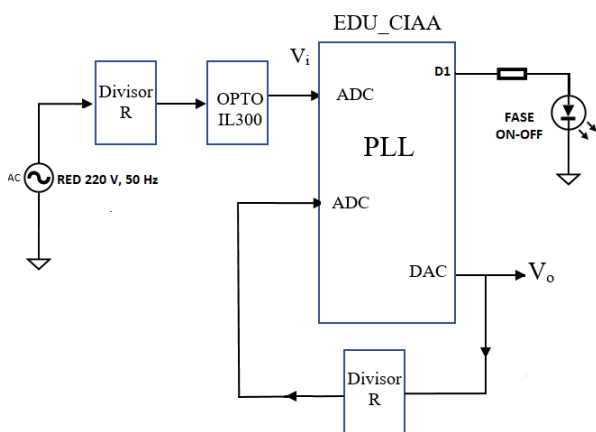


Fig. 3. Configuración para el ensayo del PLL

A nivel de software, el desarrollo comprende 3 funciones o módulos operativos dentro del proyecto principal de programación en lenguaje C: a) filtro pasa bajos, b) controlador PI, y c) el oscilador controlado por tensión VCO.

A- Filtro digital

La implementación del filtro digital se corresponde con el desarrollo algorítmico de una ecuación de diferencias de orden 10, correspondiente al diseño de un filtro FIR estructura directa pasa bajo con frecuencia de corte en 40 Hz.

B- Controlador PI

Se corresponde a un controlador digital Proporcional Integral, con valores de $k_p=50$ y constante $T_i=0.02$,

C- Oscilador VCO

El oscilador VCO se implementará en base a la síntesis digital de una onda triangular de 50 Hz, como portadora controlada por tensión según la configuración de un VCO que se muestra en la figura 1, para el control de frecuencia de la de la onda sinusoidal sincronizada.

IV. ENSAYOS Y RESULTADOS

Al momento de esta presentación el sistema está desarrollado a nivel del filtro y controlador digital, presentando las respuestas esperadas, en tanto se espera completar la programación del módulo VCO integrado a las etapas anteriores para la verificación del sistema completo según las especificaciones de diseño. Se debe implementar el filtro de Goertzel para determinar la presencia energía en 50 Hz, dando entrada en servicio al convertidor DC/AC en sincronismo con la red eléctrica (salida D1 de la EDU-CIAA).

V. CONCLUSIONES

La utilización de la EDU-CIAA en el desarrollo embebido del PLL, simplifica el hardware de soporte principalmente por las prestaciones y capacidades de su microcontrolador, más aún cuando el mismo se integre al convertidor DC/AC para el cual es diseñado. Se espera completar el desarrollo para la verificación completa del sistema.

REFERENCIAS

- [1] Encinas, J.. Phase Locked Loops. Springer. pp 1-45. 1993.
- [2] http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:edu-ciaa:edu-ciaa-nxp#bloques_funcionales. Proyecto CIAA.
- [3] Muhammad H. Rashid. Power electronics handbook. Third Edition., Ph.D., ELSEVIER . pp 357-367. 2011.
- [4] Rusty Allred. Digital Filters For Everyone Third Edition. Creative Arts & Sciences House. 2015. pp 20-76. 2016.
- [5] Rulph Chassaing and Donald Reay. Digital Signal Processing and Applications with the TMS320C6713 and TMS320C6416 DSK, Second Edition. John Wiley & Sons. pp 557-560. 2008.

Foro Tecnológico

Pósters

**Protocolos,
comunicaciones y
redes inalámbricas**

FPGA implementation of a low complexity decoder for LDPC codes over impulsive noise channels

Leonardo Arnone, Carlos Gayoso, Claudio González,
Miguel Rabini
Laboratorio de Componentes, UNMdP e ICYTE
leoarn@fi.mdp.edu.ar

Jorge Castiñeira Moreira
Laboratorio de Comunicaciones, UNMdP e ICYTE
CONICET, Argentina
casti@fi.mdp.edu.ar

Abstract—A simplified algorithm for decoding error-correcting Low Density Parity Check (LDPC) codes is implemented. The performance of the implemented decoder is studied under the effect of the Additive White Gaussian Noise (AWGN), the Rayleigh fading and the Middleton’s Class A noise channels. The problem of the distortion and Bit Error Rate (BER) performance degradation produced by impulsive noise leads to modifications in the initialization step of the iterative decoding algorithms for decoding LDPC codes, and this initialization results into increased complexity. This problem is sorted out by simply using an Euclidean distance metric decoder together with a signal amplitude limiter, which allows the use of a simple initialization step. The proposed decoder does not require knowledge of the signal-to-noise ratio of the received signal. It uses only additions and subtractions, avoiding the use of quotients and products. The algorithm can be easily implemented on programmable logic technology such as Field Programmable Gate Array (FPGA) devices with an excellent BER performance.

Keywords—Impulsive noise channels, Euclidean metric, LDPC codes, FPGA

I. INTRODUCTION

Among the different error-control techniques available nowadays, Low-Density Parity-Check (LDPC) codes stand out due to their excellent Bit Error Rate (BER) performance. However, these codes are related to high complexity aspects at the moment of a real implementation when such a performance is desired. On the other hand, the classic analysis done over error-control codes is usually performed over the well-known model of the Additive White Gaussian Noise (AWGN) channel, which is considered the standard environment over which a given code can operate. However, many real communications environments are characterized by non-Gaussian sources of noise. These communications channels are typically affected by what is known as impulsive noise. Power Line Communications (PLC) appears as one of the most interesting applications characterized by this sort of noise.

The Middleton’s Class-A [1] impulsive noise channel is usually considered as a good model for representing impulsive noise channels like the PLC channel. This channel model becomes a useful test for the operation of the proposed decoders in the design of LDPC codes for impulsive noise

channels. The intense peak values registered in the amplitude levels of this sort of noise, results into a strong degradation of the BER performance.

The Rayleigh fading channel, which is used for modeling communication scenarios where multipath effect takes place, is another channel of interest.

The aim of this paper is to do a Field Programmable Gate Arrays (FPGA) [2] implementation of an efficient LDPC decoder that uses squared Euclidean distance as its metric [3][4], suitable for operating over impulsive noise channels.

This implemented decoder utilizes only additions, subtractions and look up tables. This decoder can operate with parity check matrices of any size and type, and with any particular generating method, and its architecture can be easily configured for different code rates. On the other hand, this decoder does not require knowledge of the signal-to-noise ratio (SNR) of the channel.

II. LDPC DECODER

LDPC codes [5] are a powerful class of linear block codes characterized by a parity check matrix \mathbf{H} , which fits the condition $\mathbf{H} \circ \mathbf{c} = \mathbf{0}$ for any codeword \mathbf{c} . An LDPC decoder is essentially a decoding algorithm designed for finding a codeword $\hat{\mathbf{c}}$ (an estimate of the codeword \mathbf{c}), able to fit that condition.

The LDPC decoding algorithm is described over a bipartite graph depicted considering the relationship between the symbol nodes j , which represent the bits or symbols of the code vector \mathbf{c} , and the parity check nodes i , which represent the parity equations described in matrix \mathbf{H} . In this iterative process, each symbol node j sends to a parity check node i the estimation q_{ij}^x that this node generates with the information provided by all others parity check nodes connected to it.

Then, each parity check node i sends the estimation r_{ij}^x to each symbol node j generated with the information provided by the other symbol nodes connected to it, based on the fact that the parity check node i condition is satisfied, if the

symbol node j is in state x . This is an iterative process in which information is interchanged between these two types of nodes. This iterative process is stopped when the condition $\mathbf{H} \circ \mathbf{c} = \mathbf{0}$ is satisfied. In this case the corresponding decoded codeword is considered a valid codeword. Otherwise, the decoding algorithm stops after a given number of iterations are performed. In this case the decoded word may or may not be a codeword.

As it is well known the most commonly used iterative decoding algorithm for LDPC codes is the classic Sum-Product (SP) decoder, usually implemented in logarithmic form, which we denote as LogSP decoder.

In this LogSP decoder, samples from the channel are used to determine initialization estimations of the received signal that are then used in the iterative decoder for determining estimates of the message bits that were transmitted.

In the case of LDPC codes designed for operating over the AWGN channel, initialization of the LogSP decoder is done by using estimates calculated using the corresponding probability density function (pdf), which is a Gaussian pdf, evaluated for the two possible received values. In a transmission of normalized values of amplitude $+1$ or -1 , where message bits are in binary format, the binary value 0 is assigned a signal of amplitude -1 for instance, and the binary value 1 is assigned a signal of amplitude $+1$. Then, for this channel, estimates of the initialization values can be done by calculating functions $f(0)$ and $f(1)$, in a form we call statistical estimates initialization, as:

$$f(1) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(y_i-1)/(2\sigma^2)} \quad (1)$$

$$f(0) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(y_i+1)/(2\sigma^2)} \quad (2)$$

As seen in the above expressions, calculation of initialization estimates requires from the knowledge of the value of σ , that is, it requires to determining the noise power present at the channel. A simplified decoding algorithm that uses Soft Distance (SD) estimates instead of statistical estimates results into a simplified decoding algorithm, that leads to a BER performance without any noticeable degradation with respect to that of the classic LogSP algorithm, as shown in [5] [6].

In this simplified algorithm the estimates are square distances from the received sample to the two possible values:

$$d_0^2(j) = (y_j + 1)^2 \quad (3)$$

$$d_1^2(j) = (y_j - 1)^2 \quad (4)$$

III. SIMPLIFIED SOFT DISTANCE ALGORITHM (SSD)

In this section we describe the SD algorithm by applying it to the factor (Tanner) graph of a binary low-density parity-check (LDPC) error-correcting code. The graph has symbol

nodes and parity nodes, with edges, as defined by the parity-check matrix of the code, connecting the symbol and parity nodes. Soft distance metric information will be passed from symbol nodes to parity nodes (the horizontal step in the algorithm), and then from parity nodes to symbol nodes (the vertical step), in an iterative manner.

In this algorithm [3] [4] channel information is used as the input information to calculate the squared distance between each bit of the received vector and the two possible values of each bit at position j . Assuming the transmission of coded binary information $\mathbf{c} = (c_1 c_2 \dots c_j \dots c_n)$ in normalized polar format with signals of amplitudes ± 1 , and for a received vector $\mathbf{y} = (y_1 y_2 \dots y_j \dots y_n)$ we calculate (for $j = 1, 2, \dots, n$) the square distances (3) and (4).

As mentioned earlier, calculation of the channel estimates in this algorithm does not require the knowledge of the noise power, as happens in the LogSP algorithm [5]. This avoids the need of additional hardware in a given receiver to determine the noise power.

The SD antilog-sum algorithm described in [3][4] can be simplified to create the simplified soft distance (SSD) algorithm. Details of the derivation of the SSD decoding algorithm can be found in [6]. Table I summarizes the main steps of the proposed decoding algorithm for LDPC codes.

IV. MIDDLETON'S CLASS-A MODEL

Middleton's Class A noise model for non-Gaussian noise channels is frequently used for modeling impulsive noise channels. This impulsive noise is constituted of a background or Gaussian component with variance σ_g^2 , and of an impulsive component with variance σ_i^2 . The corresponding pdf is a Poisson weighted sum of Gaussian distributions given by:

$$P_n(n) = \sum_{m=0}^{\infty} \frac{A^m e^{-A}}{\sqrt{2\pi m!} \sigma_m} \exp\left(-n^2 / (2\sigma_m)^2\right) \quad (5)$$

With:

$$\sigma_m^2 = \sigma^2 \frac{m/A + \Gamma}{1 + \Gamma}, \quad \Gamma = \frac{\sigma_g^2}{\sigma_i^2} \quad \text{and} \quad \sigma^2 = \sigma_g^2 + \sigma_i^2 \quad (6)$$

In the above expressions, m is the number of impulses, A is the average number of impulses during interference time, and it is called the impulsive index, Γ is the ratio between the background noise power σ_g^2 and the impulsive noise power σ_i^2 , called the Gaussian-to-impulsive noise power ratio (GIR), σ_m^2 is the variance of the Poisson random variable m , and σ^2 is the total noise power.

V. INITIALIZATION STEP FOR DECODING ALGORITHMS FOR LDPC CODES

As explained in previous sections a suitable way of calculating the initialization values of a given iterative decoding algorithm for LDPC codes is by using the assumed pdf of the channel over which the code will operate, evaluated for the two possible transmitted symbols.

In the case of LDPC codes operating over the AWGN channel initialization is done by using Gaussian expressions (1) and (2). In the case of the SSD algorithm initialization values are given by more simple SD expressions (3) and (4). As mentioned earlier this procedure does not produce any degradation in BER performance with respect to the classic initialization step for the LogSP decoding algorithm.

If we consider non-Gaussian channels like the Middleton's Class A impulsive noise channel, initialization using Gaussian expressions can be also done, but with a degradation in BER performance. It is also possible to make use of the SSD algorithm, taking advantage of its added simplicity, and of avoiding to estimating the noise power present at the channel. In the case of non-Gaussian channels, the use of a signal amplitude limiter is a very simple and efficient technique for improving the BER performance. Results of the operation of Polar codes over this sort of channels using this initialization have been presented in [7].

In that paper the classic LogSP algorithm and the SSD algorithm are studied for Polar codes over the Middleton's Class A impulsive noise channel. The use of a signal amplitude limiter results into important BER performance improvements for both decoding algorithms in comparison to their performance when they are initialized using expressions of the Gaussian case, or expressions for the SSD algorithm.

Another alternative is to initialize the LogSP algorithm using estimates determined by the pdf corresponding to the Middleton's Class A channel, which is given by expression (5). It is clear that the use of this expression for the initialization step requires the knowledge of the noise power at the channel, and it is itself a rather complex expression to be implemented in FPGA.

We evaluate the initialization step of an iterative decoder for LDPC codes operating over the Middleton's Class A impulsive noise channel. A first alternative is to initialize the LogSP decoder Gaussian estimates, and assist the decoder with signal amplitude limiters. A second option is to initialize and use the SSD decoding algorithm, and also assist the decoding algorithm with a signal amplitude limiter. A third option is to initialize a LogSP decoding algorithm with initialization estimates that are calculated using the pdf for the Middleton's Class A impulsive noise (5).

Simulations were done to identify which of the possible options is the best alternative. Fig. 1 shows the BER performance of different decoding alternatives. Simulation results show the BER performance of the (1024, 512) LDPC code over a Middleton's Class-A impulsive noise channel, decoded using the LogSP decoder, the SSD decoder and the LogSP initialized with Class A pdf estimates decoder.

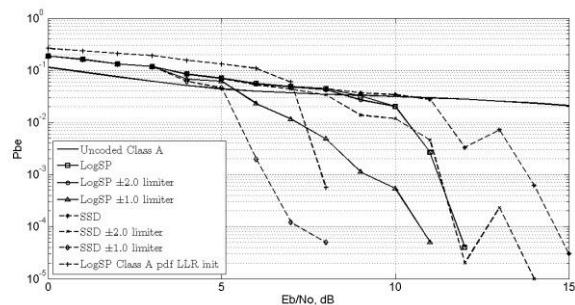


Fig. 1. BER performance of different alternative decoder for a (1024, 512) LDPC code over the Middleton's Class A impulsive noise channel.

In simulations, 10,000 messages of 512 bits each are transmitted. Simulations are done for different values of the limiting amplitude $Lim = \pm 2$ and $Lim = \pm 1$, set with respect to noise-free transmitted signals of amplitudes ± 1 .

Among the different alternatives, it is observed that the LogSP decoder initialized with estimates calculated using the pdf of the simulated channel outperforms the LogSP algorithm with Gaussian initialization, and also the SSD algorithm with SD initialization.

However, the use of signal amplitude limiters makes the SSD algorithm have the best BER performance. The initialization step using expression (5) can result into a rather complex FPGA implementation. As a lower complexity alternative, we propose the use of the SSD decoder assisted by a signal amplitude limiter. The second advantage in reduced complexity is that the estimation of the noise power present at the channel is not required. The proposed decoder for LDPC codes over non-Gaussian channels is presented in section VII.

VI. SIGNAL AMPLITUDE LIMITER

The signal amplitude limiter is simply a non-linear function in the time domain that limits the amplitude of the input signal to be in a determined amplitude range ± 1 . It is applied as the first block of the block diagram decoder to conform the sampled values obtained from the channel, which in our case is directly a scaled value of the channel sample that is input to the SSD decoder. The signal level limiter operates over the channel samples by applying the following rule:

$$\bar{y}_i = \begin{cases} Lim & \text{if } y_i > Lim \\ -Lim & \text{if } -y_i < -Lim \\ y_i & \text{otherwise} \end{cases} \quad (7)$$

VII. FPGA IMPLEMENTATION OF A SSD DECODER

An FPGA implementation of a LDPC decoder has been proposed in [9]. The proposed implementation in this paper is based on that implementation, but involves some slight simplifications in its structure, that can be seen in Table I.

TABLE I. SIMPLIFIED VERSION OF THE SOFT DISTANCE ALGORITHM

Initialization
$q_{ij}^0 = d_0^2(j), q_{ij}^1 = d_1^2(j)$
Horizontal Step
$t_{ij} = \sum_{k \in N(i)j} f_+(q_{ik}^0, q_{ik}^1) - \sum_{k \in N(i)j} f_-(q_{ik}^0, q_{ik}^1)$
If $q_{ij}^1 \geq q_{ij}^0$ $s_{ij} = 0$ else $s_{ij} = 1$
Vertical Step
if $\sum s_{ik}$ is even
$q_{ij}^0 = d_0^2(j) - \sum_{k \in M(j)i} f_+(t_{kj}, 0)$
$q_{ij}^1 = d_1^2(j) + \sum_{k \in M(j)i} f_-(t_{kj}, 0)$
if $\sum s_{ik}$ is odd
$q_{ij}^0 = d_0^2(j) + \sum_{k \in M(j)i} f_-(t_{kj}, 0)$
$q_{ij}^1 = d_1^2(j) - \sum_{k \in M(j)i} f_+(t_{kj}, 0)$
Decision
$\hat{d}_0^2(j) = r_{n_{ij}}^0 + q_{ij}^0$
$\hat{d}_1^2(j) = r_{n_{ij}}^1 + q_{ij}^1$
If $\hat{d}_1^2(j) < \hat{d}_0^2(j)$ then $\hat{c}(j) = 1$ else $\hat{c}(j) = 0$

The main modification on the FPGA topology for a LDPC decoder operating over a Middleton's class A impulsive noise channel, is the use of a signal amplitude limiter, which in spite of being a simple modification, results into a very significant improvement in BER performance.

It is noted that the signal amplitude limiter block allows us the use of SD metric, and of the SSD algorithm, which can be implemented with a very low complexity and leads to the implementation of an initialization step that is the least complex among the three cases studied. Fig. 2 shows the implementation of the decoder. The implementation reorganizes memory blocks. A direct calculation of parameters t_{ij} (Table I) avoids the use of intermediate parameters.

Matrix indexes of positions of ones of the matrix H are recorded in the memory "ROM H Matrix". Its size depends on the number of columns with ones in each row, and on the number of rows of the matrix. Memory "ROM Number Ones Rows" stores the number of columns with ones in each row.

This memory is used together with memory "ROM H Matrix", to get the position of each one of the matrix H .

Memories "ROM Look Up Tables" contain look-up tables $f_+(a, b)$ and $f_-(a, b)$.

Memories "RAMs Euclidean Distances" store squared distance values d_0^2 and d_1^2 each time a new received vector is input to the decoder. Its size depends on the length of the code.

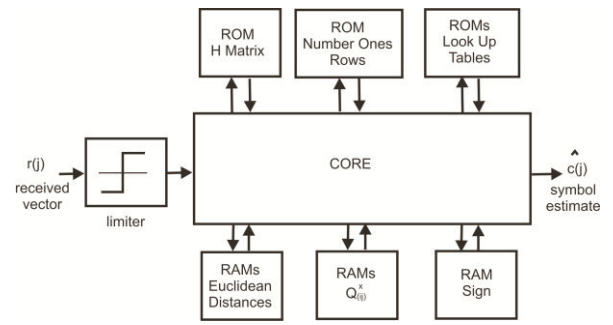


Fig. 2. FPGA implementation of a LDPC decoder for operating over Gaussian and non-Gaussian channels using ALTERA [2].

Memories "RAMs Q_{ij}^x " perform several functions: In the initialization step they store squared distance values d_0^2 and d_1^2 . In the horizontal step they store values $\sum f_+(q_{ij}^0, q_{ij}^1)$ and $\sum f_-(q_{ij}^0, q_{ij}^1)$.

Finally Memories "RAMs Q_{ij}^x " store again, in the vertical step, values q_{ij}^0 and q_{ij}^1 , so that they have a high degree of reuse.

Memory "RAM Sign" stores values s_{ij} and $\sum s_{ik}$ in the horizontal step.

Table II shows resources utilized in the FPGA implementation using a programmable logic device Cyclone II EP2C50F484C6 from Altera [2] for a (1024, 512) LDPC code.

Since size of most of the memories used in the implementation depends on the code size, resources requirements significantly reduce in the case of small size LDPC codes.

VIII. ADDITIONAL SIMULATION RESULTS FOR LDPC DECODERS OVER OTHER CLASSIC CHANNELS USING SIGNAL AMPLITUDE LIMITERS

All simulations compare the performance using both the classic LogSP algorithm [8] and the proposed SSD algorithm [6]. They are applied with and without the use of signal amplitude limiters, that in our case are set to values $Lim = \pm 2$ and $Lim = \pm 1$, for both decoders. This is done with respect to transmitted signal amplitude values of normalized form ± 1 .

TABLE II. RESOURCES IN FPGA IMPLEMENTATIONS OF A SSD DECODER, (1024, 512) LDPC CODE

Hardware used	EP2C50F484C6
Total logic elements	795
Total registers	431
Total memory bits	231424
Clock frequency	133.55MHz

The performance of the proposed scheme is equivalent to those of classic decoders, with additional advantages of reduced decoding complexity and the fact of avoiding the need of knowing the channel SNR for decoding calculations. From this point of view, the proposed algorithm is a better option than the classic decoding algorithms. In the case of the AWGN channel however, the use of amplitude limiters does not improve the BER performance and certainly produces degradation.

Fig. 3 shows the BER performance of LogSP algorithm and the SSD algorithm under the effect of AWGN channel.

Fig. 4 shows the BER performance of a (1024, 512) LDPC code operating over a Rayleigh fading channel, in a transmission of blocks of 10000 messages of 512 bits each.

It is seen that the performance of the SSD is degraded with respect to the LogSP when they are applied without the signal amplitude limiter. The best performance for the SSD decoder is verified for amplitude limiter $Lim = \pm 1$.

As seen in simulations the use of a simple signal amplitude limiter highly improves the BER performance of the LDPC decoder, and this improvement is higher in the case of the implemented SSD decoder. Thus, the proposed implementation can be successfully utilized for operating over different channels to provide a very good BER performance, including those channels where impulsive noise results into a severe BER performance degradation.

IX. CONCLUSIONS

Our results show that the BER performance of the SSD implemented decoder over non-Gaussian channels like the Rayleigh fading channel and the Middleton's Class A impulsive noise channel, are significantly improved with the use of a simple signal amplitude limiter. This performance is obtained without the need of estimating the channel SNR, and there is also a reduction in decoding complexity.

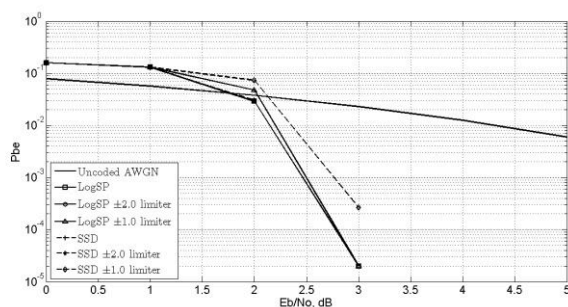


Fig. 3. BER performance of a LDPC code over an AWGN channel decoded using LogSP and SSD algorithms, for different values of the signal amplitude limiter.

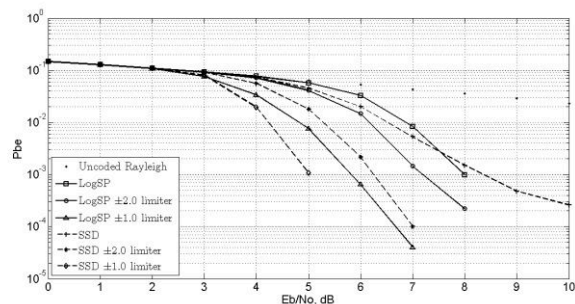


Fig. 4. BER performance of a LDPC code over a Rayleigh channel decoded using LogSP and SSD algorithms, for different values of the signal amplitude limiter.

REFERENCES

- [1] D. Middleton, "Statistical-physical models of electromagnetic interference," IEEE Trans. on Electromagnetic Compatibility, vol. 3, EMC-19, pp. 106-127, 1977.
- [2] www.altera.com, "Cyclone II FPGAs," On Line.
- [3] [4] P. G. Farrell, "Decoding Error-Control Codes with Soft Distance as the Metric," in Proc. Workshop on Mathematical Techniques in Coding Theory, Edinburgh, UK, 2008.
- [4] P. G. Farrell and J. Castiñeira Moreira, "Soft-Input Soft-Output Euclidean Distance Metric Iterative Decoder for LDPC Codes," in Proc. Argentine Symposium on Computing Technology (AST 2008), Santa Fe, Argentina, 2008.
- [5] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," Electronics Letters, vol. 33, pp. 457-458, 1997.
- [6] P. G. Farrell, L. J. Arnone, and J. Castiñeira Moreira, "Euclidean distance soft-input soft-output decoding algorithm for low-density parity-check codes," IET Communications, vol. 5, issue 16, pp. 2364-2370, 2011.
- [7] M. C. Liberatori, L. J. Arnone, J. Castiñeira Moreira, and P. G. Farrell, "Soft Distance Metric Decoding of Polar Codes," Springer International Publishing Switzerland 2015, J. Groth (Ed.): IMACC 2015, LNCS 9496, 2015, pp. 173-183. DOI: 10.1007/978-3-319-27239-9_10
- [8] L. Arnone, C. Gayoso, C. González and J. Castiñeira, "Sum-Subtract Fixed Point LDPC Decoder," Latin American Applied Research, vol. 37, pp. 17-20, 2007.
- [9] P. G. Farrell, L. J. Arnone and J. C. Moreira, "Euclidean distance soft-input soft-output decoding algorithm for low-density parity-check codes," in IET Communications, vol. 5, no. 16, pp. 2364-2370, 2011.

Sistema de Registro Automático de Información para el Análisis y Gestión del Tránsito Vial

Manuel Stillitano, Juan Carlos Bonadero, Alejandro Uriz, Juan Alberto Etcheverry, Magdalena Grau y Ramiro Ávalos

Laboratorio de Comunicaciones
ICyTE (CONICET – Universidad Nacional de Mar del Plata)
Mar del Plata, Argentina
ajuriz@fi.mdp.edu.ar

Abstract— En este trabajo se presenta un sistema automático de registro y transmisión de información para el análisis y planificación del tráfico urbano y mantenimiento de los caminos. Este sistema permite planificar de mejor manera el tránsito, realizar más eficientemente el mantenimiento de los canales por los cuales este circula y hacer estudios tanto sobre la ubicación geográfica de los problemas como su distribución temporal. El sistema se compone de dos tipos de dispositivos (M1 y M2) los cuales podrán vincularse de forma inalámbrica: los primeros con autonomía y movilidad, que se instalarán en vehículos; y los restantes con ubicación fija. El primer conjunto (los dispositivos M1) se encargará de registrar diversos parámetros del automóvil e identificar la presencia de una o varias situaciones predefinidas para luego registrar una serie de datos relevantes a la misma. El segundo conjunto (los dispositivos M2) se encargará de reconocer a cualquiera de los dispositivos M1 cuando se encuentren en su área de alcance y recolectará todos los datos almacenados en este. Luego los datos recolectados serán subidos a una base de datos on-line mediante conexión Ethernet para su almacenamiento y posterior análisis.

Keywords—registro de parámetros viales, sistema embebido, ordenamiento urbano.

I. INTRODUCCIÓN

La República Argentina posee una tasa de urbanización alta. En el año 2001 la población urbana representaba el 89,4% de la población total del país y para el 2010 ascendía al 91% [1-4]. Mientras la mitad de su población reside en las 6 grandes ciudades o aglomerados, un tercio (13 millones) lo hace en las 273 ciudades de tamaño medio, de entre 10 mil y 500 mil habitantes. Es para destacar el caso de la Región Metropolitana de Buenos Aires, donde habita el 37% de la población del país. Se puede sumar al análisis el aumento anual del parque automotor. La flota automotriz circulante a fines de 2014 fue de 919.123 vehículos más que en 2013 cuando se registraron 12.456.864 unidades. Esto confirmó a la Argentina como el país de la región con más vehículos por cantidad de habitantes, ya que tiene 3,2 habitantes por unidad, al cierre de 2014 [1-4]. Dichos crecimientos generan impactos notables en la naturaleza de las ciudades e implican inherentemente la intensificación de los niveles y tipos de desplazamiento de personas, bienes y productos. Debido a estas cambiantes

circunstancias nace la necesidad de actualizar y mejorar los sistemas y vías de transporte urbanos para así adaptarse a las mismas. Dicha necesidad interpela a los gobiernos locales hacia el desarrollo de sistemas de gerenciamiento más eficaces y la búsqueda de soluciones más eficientes para los continuos [5-7], crecientes y cambiantes problemas de movilidad y transporte interurbano. Es interesante agregar otro punto de vista al análisis de las problemáticas planteadas anteriormente, en este caso el del automovilista. Esto sería abordar el análisis desde la perspectiva de los protagonistas de los sucesos. En este trabajo se propone implementar un sistema distribuido de recolección de datos que permita disponer de información actualizada minuto a minuto en forma automática que permitan mejorar el análisis del tráfico urbano y la toma de decisiones.

II. DESCRIPCIÓN DEL SISTEMA

Un diagrama en bloques del sistema ideado completo puede verse en la Figura 1. El proyecto desarrollado abarcó los bloques incluidos por el recuadro punteado verde y está compuesto por los dispositivos M1 y M2, además del servidor y la base de datos donde se alojan los datos recolectados. El M1 es un dispositivo portátil que debe utilizarse en el vehículo, posee un microcontrolador, un GPS, un módulo lector de tarjeta SD y un módulo XBee [9]. El dispositivo M2 es un terminal fijo que posee un microcontrolador (el mismo del M1) un módulo Ethernet y un módulo XBee. A continuación se describe cada bloque.

A. Bloque de posicionamiento (GPS)

La tecnología de posicionamiento global está provista por el dispositivo EM-411 de la empresa GlobalSat [10,11]. Posee una interfaz de comunicación serie (USART) y entrega datos de ubicación, tiempo, velocidad, dirección y fecha a través de protocolo NMEA 0183 [12].

B. Almacenamiento de datos

El sistema de almacenamiento de datos está constituido por un módulo lector de tarjetas SD sobre la cual se guardan los datos en un archivo .csv (tipo MS Excel) sobre un sistema de archivos FAT32.

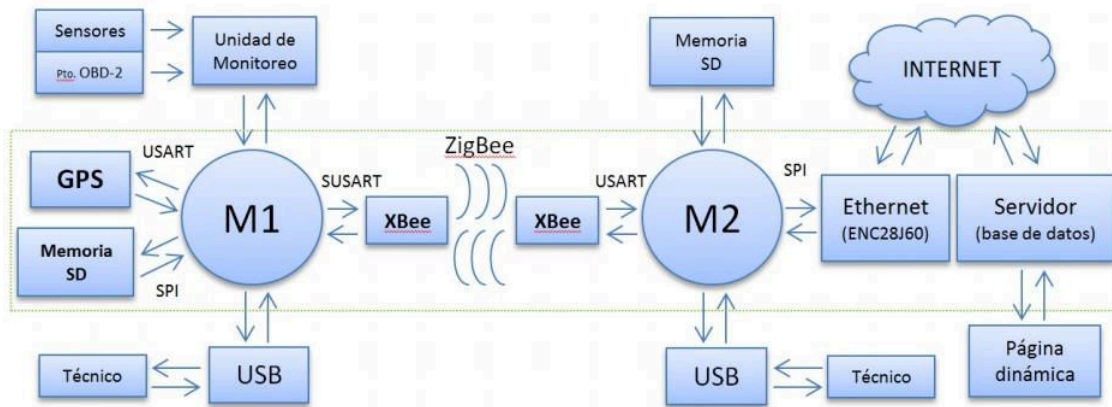


Fig. 1. Diagrama en bloques del sistema desarrollado.

C. Comunicación entre módulos

El dispositivo elegido fue el XBeePRO Series 2 de la empresa Digi International [8]. Provee un enlace inalámbrico entre el dispositivo M1 y M2. Este dispositivo tiene un interfaz serie y opera con el estándar de comunicaciones ZigBee [11,12]. Los XBee son utilizados por los dispositivos M1 y M2 como interfaz inalámbrica del sistema. Este dispositivo es de suma importancia para el proyecto ya que gracias a sus características y su funcionalidad permitió la idealización y realización de todo el sistema.

D. Comunicación Ethernet

El sistema de conexión para redes de área local está constituido por el módulo Ethernet ENC28J60 de la empresa Microchip [13]. Este módulo trabaja con una interface SPI y es utilizado para establecer una conexión TCP (socket) con un servidor y poder enviar la información que luego será subida a una base de datos por dicho servidor.

E. Servidor y base de datos implementados

El bloque encargado de responder a la solicitud del ENC28J60 para establecer una conexión TCP a través de Internet está formado por un sistema WAMP 2.2 [15]. El mismo incluye el servidor HTTP Apache 2.2.22 [16] de código abierto y el gestor de bases de datos MySQL 5.5.24 [17], entre otras herramientas. En el servidor se aloja una página con código PHP la cual se encarga de recibir los datos enviados por el M2, discriminarlos y luego enviarlos al gestor MySQL para que los almacene en una base de datos.

F. Servidor y base de datos implementados

El microcontrolador elegido para el proyecto fue el PIC18F27J53 de la empresa Microchip [14]. Es un microcontrolador de 8 bits, bajo consumo, alto rendimiento y con interface USB 2.0 fullspeed integrada. El mismo puede funcionar con dos modos distintos de oscilador externo y trabajar a una frecuencia de hasta 48MHz. Posee un encapsulado de 28-pines tipo SPDIP/SOIC /SSOP y la capacidad de remapear varios periféricos sobre 16 de sus 28 pines. Se encarga de coordinar módulos, administrar los datos entre los mismos y procesar la información

III. DESCRIPCIÓN DEL FIRMWARE DE LOS DISPOSITIVOS

El programa del dispositivo M1, el cual se presenta en la Figura 2, está basado en el uso de interrupciones para actuar cuando se recibe un comando u orden específica, adquirir los respectivos datos requeridos por las aplicaciones, analizarlos y procesarlos. El programa del dispositivo M2, el cual es mostrado en la Figura 3, está basado en la librería “TCP/IP Stack Software” de Microchip [13] con la cual realizan las funciones de conexión TCP. A su vez utiliza rutinas de interrupción con fines similares al dispositivo M1.

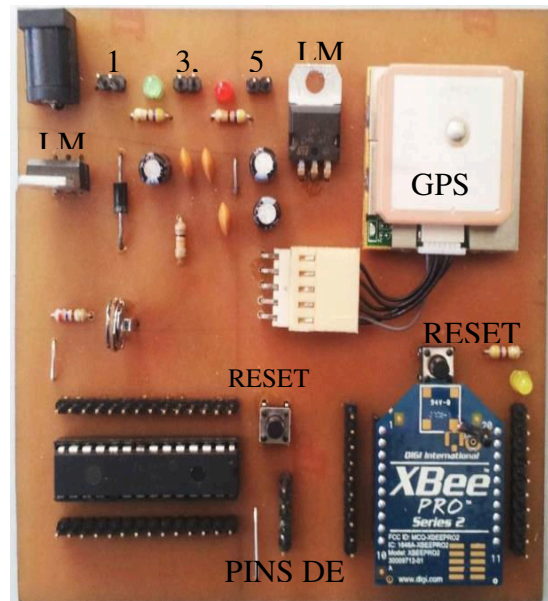


Fig. 2. Implementación del Dispositivo M1.

A. Firmware del dispositivo M1

Consiste de tres procesos principales y de varias funciones secundarias asociadas a dichos procesos. Primero se ejecuta un proceso de inicialización y configuración del hardware y los periféricos. Luego, se activan los procesos de atención a interrupciones en segundo plano y por último el programa entra en un bucle “while” infinito. El contenido del “while” consiste

en una rutina de encuestas para decidir si ejecutar diferentes subrutinas:

1) *Interrupción START-bit Rx-XBee*: es una rutina de alta prioridad, la única del M1, donde se monitorea la aparición de un flanco descendente proveniente del transmisor del XBee sobre el terminal designado del microcontrolador como receptor de la interfaz serie dedicada a dicho módulo. Dicha interfaz es una USART por software (SUSART). Al detectarse el flanco se activa la interrupción del Rx SUSART.

2) *Interrupción Rx SUSART*: mediante una estructura condicional se decide si atender la recepción del dato recibido o simplemente ignorarlo. Se espera la recepción de los caracteres “S”, “N” y “D”, en orden y uno después del otro. Al recibirse dicha secuencia de caracteres se setea una bandera para activar el envío de un paquete desde la memoria SD hacia el módulo XBee. Si el orden no es el correcto, se resetea el proceso.

3) *Interrupción Rx GPS*: se encarga de reconocer la recepción del carácter “\$”, que es el primer carácter de la cabecera que incluyen los paquetes enviados por el GPS. Si se verifica la recepción de dicho carácter, se activa una bandera para recibir y analizar la cabecera del paquete, a medida que se van recibiendo los sucesivos caracteres. Si la cabecera es la correcta, se activa una bandera para recibir el resto del paquete y guardarlo en un buffer para luego poder extraer los datos del mismo.

B. Firmware del Dispositivo M2

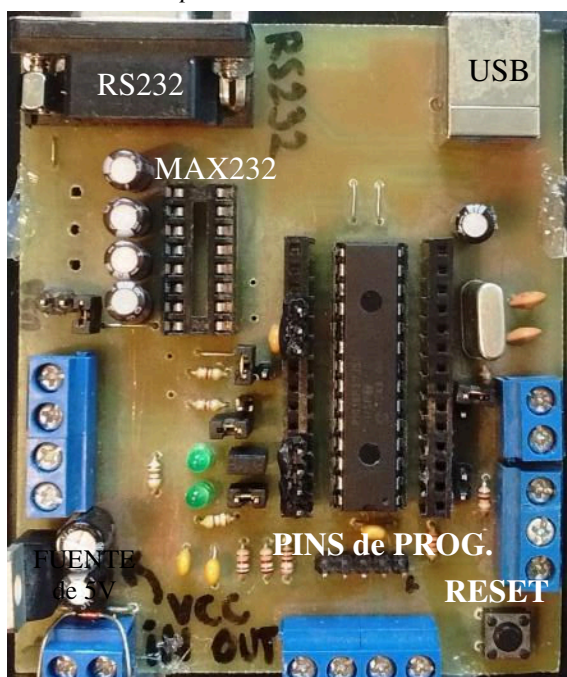


Fig. 3. Implementación del Dispositivo M2.

Está basado en la librería “TCP/IP Snack Software” de la cual se utilizó uno de los códigos de ejemplo que incluye la

librería. A partir del código original del ejemplo se desarrolló el código final, modificándolo para cumplir con las necesidades del hardware utilizado y agregándole:

1) *Interrupción Rx XBee*: se procesan los datos recibidos provenientes del módulo XBee. Dichos datos deberían ser los paquetes enviados por el dispositivo M1, los cuales poseen una cabecera “\$ZB”. Se analiza la correcta recepción de la cabecera mediante un proceso similar al usado en la rutina de atención a la interrupción de Rx SUSART. Al corroborarse que los datos entrantes corresponden a un paquete enviado por el M1, se procede a guardar en un buffer el resto de los datos entrantes llamando a una función encargada de realizar dicha tarea.

2) *Interrupción Tick (timer)*: es una rutina de baja prioridad donde se atiende una interrupción generada por un timer una vez por segundo. Dicha interrupción se utiliza para generar una señal intermitente con un LED y, en el caso de que no se esté procesando ningún paquete entrante, habilitar el envío de un nuevo comando ‘SND’. De esta forma, si algún M1 entra en el área de cobertura del M2, en un tiempo menor a un segundo recibirá un comando ‘SND’, el cual responderá con un paquete de datos.

IV. PRUEBAS DEL SISTEMA

A partir de un sencillo experimento se intentó obtener algunos parámetros de interés para el desarrollo del proyecto, tales como alcance del enlace RF, tiempo de establecimiento y volumen de datos de transmisión vs velocidad del vehículo.

El ensayo consistió en montar el dispositivo móvil dentro de un automóvil y realizar una serie de pasadas a distintas velocidades desde una distancia que superara al área de la cobertura del enlace inalámbrico hasta el punto donde estaba ubicado el módulo receptor y observar el desempeño del sistema. No se tuvo en cuenta el desempeño del sistema al alejarse el vehículo.

Se configuraron ambos módulos en modo AT para que establecieran un vínculo directo, a partir de sus direcciones de identidad. Desde una ventana del Laboratorio de Comunicaciones de la Facultad de Ingeniería de la UNMDP, que daba a la calle por donde pasaría el vehículo con el dispositivo móvil, se dispuso el módulo receptor conectado a la PC y al X-CTU para visualizar los paquetes recibidos. Luego, se realizaron tres pasadas a veinte, treinta y cuarenta kilómetros por hora, tomando las debidas precauciones de seguridad y siendo la última velocidad la máxima admisible para la experimentación (el límite de velocidad en calles es de 40 km/h). El módulo móvil se configuró para que entregara continuamente los datos que suministraba el GPS (una trama de datos por segundo). En la recepción, se volcaban los datos recolectados a un archivo de texto para luego analizarlos detenidamente. Del análisis hecho sobre los datos obtenidos se pudo extraer la posición geográfica del vehículo (en latitud y longitud a cada segundo), la hora UTC y la velocidad. Los resultados obtenidos fueron los siguientes:



Fig. 4. Gráfico de Google Earth con los puntos iniciales y finales de cada pasada de prueba.

- En la primera pasada, el primer punto registrado satisfactoriamente fue $38^{\circ} 00.6140' S, 057^{\circ} 35.0815' W$ a las 14:12:12.000 horas UTC, con una velocidad de 20,80 Km/h. Cuando se dice “satisfactoriamente” es debido a que en la recepción de los primeros datos se registran pérdidas de paquetes, hasta que se ingresa plenamente en lo que parece ser la zona de alcance. El tiempo de finalización, que corresponde a la ubicación $38^{\circ} 00.7289' S, 057^{\circ} 34.9296' W$, fue de 14:12:56.000 horas UTC.

- En la segunda pasada, el primer punto registrado satisfactoriamente fue $38^{\circ} 00.6249' S, 057^{\circ} 35.0638' W$ a las 14:17:15.000 horas UTC, con una velocidad de 29,85 Km/h. El tiempo de finalización fue de 14:17:45.000 horas UTC para el punto $38^{\circ} 00.7293' S, 057^{\circ} 34.9270' W$.

- En la tercera pasada, el primer punto registrado satisfactoriamente fue $38^{\circ} 00.6308' S, 057^{\circ} 35.0553' W$ a las 14:22:51.000 horas UTC, con una velocidad de 36,19 Km/h. El tiempo de finalización fue de 14:23:13.000 horas UTC para el punto $38^{\circ} 00.7266' S, 057^{\circ} 34.9291' W$.

A continuación se presenta un gráfico realizado con Google Earth indicando los tres puntos registrados en cada pasada, siendo en verde la primera pasada, en amarillo la segunda y en rojo la tercera.

3.4.2. Cálculos y estimaciones

Se puede ver claramente que la zona de alcance es mayor a 300 metros. Hay que tener en cuenta que el enlace producido en el ensayo tiene un espacio libre, por lo que un enlace con obstáculos debiera ser sustancialmente más reducido. Considerando que, según las especificaciones, la velocidad máxima de transferencia de datos del XBee es de 250Kbps y el tiempo de duración de cada pasada, se puede estimar un caudal de datos para cada velocidad:

- En el primer caso se registró un tiempo de enlace de 44 segundos y se recorrieron (según la “regla” que permite medir en el Google Earth) 300 metros, aproximadamente. Para una velocidad promedio de 24,5 km/h, se estima un caudal de datos de 11 Mb.

- En el segundo caso se registró un tiempo de enlace de 30 segundos y se recorrieron 276 metros, aproximadamente. Para una velocidad promedio de 33,12 km/h, se estima un caudal de datos de aproximadamente 7,5 Mb.

- En el tercer caso se registró un tiempo de enlace de 22 segundos y se recorrieron 250 metros, aproximadamente. Para una velocidad promedio de 40,91 km/h, se estima un caudal de datos de aproximadamente 5,5 Mb.

Para estimar un tiempo de establecimiento se necesita plantear algunas ecuaciones dado que solo se conoce los puntos donde se consolidan los enlaces y no el punto de inicio de asociación. Asignamos como “a” la distancia que hay entre el primer punto registrado para la primera pasada, y el punto de inicio de asociación. Análogamente se hace lo mismo con “b” y el primer punto de la segunda pasada, “c” y el primer punto de la tercera pasada y designamos a “ta” como el tiempo de asignación. Los datos son, las distancias entre cada punto y las velocidades en cada punto (en m/s). Se supone velocidad constante en cada intervalo y que todos los puntos tienen un mismo punto de inicio de asociación. Luego:

- $b-a = 32m$
- $c-a = 48m$
- $c-b = 16m$
- $va = 6,81 m/s$
- $vb = 9,2 m/s$
- $vc = 11,36 m/s$
- $va = a/ta$
- $vb = b/ta$
- $vc = c/ta$

Se puede plantear la siguiente ecuación:

$$a+(b-a) = b$$

$$ta \cdot va + (b-a) = ta \cdot vb$$

$$ta \cdot (vb - va) = (b-a)$$

$$ta = (b-a)/(vb - va)$$

Si se usa las combinaciones de los tres puntos se puede obtener tres valores de t_a , con los cuales calcularemos un promedio para obtener un valor final de t_a :

- $t_{a1} = 13,39$ segundos
- $t_{a2} = 10,55$ segundos
- $t_{a3} = 7,71$ segundos

$t_a \approx 10,45$ segundos

Si se considera el tiempo de establecimiento, se puede estimar la zona de alcance del enlace para las condiciones del ensayo. Sumando la distancia al primer paquete recibido y la distancia al punto estimado de inicialización de asociación se calcula una estimación del área de cobertura (“d”) para cada pasada y promediaremos los valores obtenidos, luego:

- $d_a = 332m + t_a.v_a = 403$ m
- $d_b = 300m + t_a.v_b = 396$ m
- $d_c = 283m + t_a.v_c = 401$ m

$d \approx 400$ m

Con los datos obtenidos se puede hacer una extrapolación para una situación con una velocidad de 60 km/h (velocidad máxima en avenidas), de la siguiente manera:

- Distancia efectiva de enlace = 400 metros – $t_a.V = 400m - 10,45s.16,67m/s = 225,8$ m.
- Volumen de datos a 60 km/h = $(225,8m/16,67s).250Kbps = 3,4$ Mb.

V. CONCLUSIONES

La información aportada por los dispositivos, actualizada minuto a minuto, permitirá hacer estudios estadísticos tanto sobre la ubicación geográfica de los problemas como su distribución temporal y grado de ocurrencia. El rango de uso potencial del proyecto es de hecho muy variado y altamente escalable, permitiendo el agregado en forma paulatina de nuevas funcionalidades con completa compatibilidad hacia atrás. Se puede considerar el sistema totalmente programable y expandible, siendo posible un desarrollo continuo en el código de operación de los dispositivos móviles y permitiendo la actualización de los mismos cuando sea necesario.

Para el trabajo a futuro se proponen los siguientes tópicos:

- Desarrollo y pruebas de los distintos dispositivos de sensado para el vehículo que ofrece el mercado local, como por ejemplo acelerómetros.
- Desarrollo de un subsistema y los respectivos algoritmos para la Unidad de Monitoreo.
- Desarrollo de una interfaz a base de protocolo USB que permita interconectar ambos dispositivos directamente a una PC, con el fin de facilitar el acceso al código de programación de los mismos.

• Desarrollo de una página dinámica que pueda acceder a la base de datos e interprete los datos recolectados. Dicho software debe mostrar, mediante una interfaz gráfica, un mapa de la ciudad y sus calles donde se puedan visualizar los puntos registrados por los dispositivos móviles. Este sitio también debe permitir filtrar los datos según el o los parámetros de interés.

VI. AGRADECIMIENTOS

Los autores desean agradecer a la Universidad Nacional de Mar del Plata, al FONCyT y al CONICET por los fondos recibidos para la ejecución de este proyecto.

VII. REFERENCIAS

- [1] Grupo del Banco Mundial, *Banco de datos del Banco Mundial*, 2016. <http://databank.bancomundial.org>.
- [2] Fundación Metropolitana, *Planificación y participación para la Gran Buenos Aires*. 2016. <http://metropolitana.org.ar>.
- [3] Observatorio Nacional de Datos de Transporte, *ONDat* 2015. <http://ondat.fra.utn.edu.ar>.
- [4] Ingeniería en Relevamientos Viales S.A., *IRV S.A* <http://www.irvsa.com.ar>.
- [5] Iteris, Inc., *Innovation for better mobility*. 2016. <http://www.iteris.com>.
- [6] Vaisala, *A global leader in environmental and industrial measurement*. 2016. <http://www.vaisala.com>.
- [7] Wavetronix LLC., *Making the world's traffic safer and more efficient*. 2016. <http://www.wavetronix.com>
- [8] Digi International Inc. www.digi.com.
- [9] Globalsat, WorldCom Corporation. www.globalsat.com.tw.
- [10] USGlobalSat Inc, 2014. http://usglobalsat.com/downloads/NMEA_commands.pdf.
- [11] SAE International, 2012. http://standards.sae.org/j2602/2_200509.
- [12] The ZigBee Alliance, *Control your World*, 2016. <http://www.zigbee.org>.
- [13] IEEE, *IEEE 802.15 WPAN™ Task Group 4 (TG4)*, 2016. <http://www.ieee802.org/15/pub/TG4.html>.
- [14] Microchip Technology Inc, 2016. <http://www.microchip.com>.
- [15] WAMPSEVER, 2016. <http://www.wampserver.es>.
- [16] The Apache Software Foundation, 2016. <https://www.apache.org>.
- [17] MySQL, 2016. <http://www.mysql.com>.

Development of an experimental telemetry system

Simón Lombardo, Santiago Rodríguez and Agustín Roncagliolo

Grupo SENyT, Facultad de Ingeniería, UNLP

La Plata, Argentina

slombardo@gmail.com, santiago.rodriguez@ing.unlp.edu.ar, agustinr@ing.unlp.edu.ar

Abstract—This article presents the design and implementation of a communications system to receive information about the status, height, speed, temperature or any other type of variable of interest from vehicles such as balloons or experimental rockets. In a first iteration, the system was designed to achieve transmission distances in the order of dozens kilometers but leaving some margin to extend the operating range to up to 100 km in line of sight. At the beginning of the project different factors such as the use of a frequency band suitable for this type of applications, the modulation scheme to use and a data protocol were analyzed. Subsequently, the most appropriate commercial devices to implement each of the system blocks were selected. With these components a first prototype of the telemetry board was built and the necessary tests to verify the correct performance of it were conducted. Finally, the design and implementation of the flight hardware was carried out and the final tests that fully validated the proposed design were performed.

I. INTRODUCTION

Within communications, telemetry is a widely used technique for the measurement of variables, either manually or automatically, in remote locations, difficult to access or having hostile environmental conditions. In general, telemetry systems are of the wireless type.

The main objective of telemetry systems is to measure physical and chemical variables, to know the state of processes and systems remotely. It is because of this, that we can find them in a wide variety of applications and processes from multiple areas like vehicular, space, biomedical and industrial.

A telemetry system usually consists of a transducer as an input device, a transmitter, a transmission medium (either wired or wireless), a receiver, signal processing devices, and finally some means for data storing or visualization. It should be noted that the modules that comprise it may vary depending on the application or the transmission medium. In addition, several current commercial devices integrate different modules of this scheme simplifying the design. A diagram of a typical telemetry system can be seen in Fig. 1.

The system proposed in this work allows reception of information about height, acceleration, angular velocity, temperature or any other type of variable of interest of vehicles such as balloons or experimental rockets. Specifically, the telemetry system is intended to be used in an experimental rocket project of the Center for Aerospace Technology (CTA) of the Faculty of Engineering of the National University of La Plata. The project consists of the development of a suborbital probe rocket as a platform for research and development of

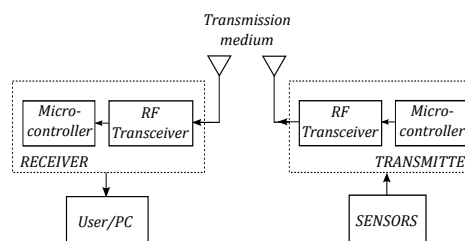


Fig. 1. Diagram of a typical telemetry system.

aerospace systems and studies of upper-atmosphere phenomena. The need for a communications module in each of the experimental rockets of the program drove the design and development of this telemetry link. The ultimate goal of the system is to receive and store data of the vehicle status and all the measurements that its instruments generate at a maximum height of 100 km above sea level and with a transmission rate that would not exceed 100 kbps, this last parameter depending on the quantity and nature of the variables to measure. It should be noted that for the first stages of the project the height reached by the rockets will not exceed 2 km. Therefore, the telemetry system was designed to achieve transmission distances of that order but leaving margin to extend it to up to 100 km (the distance required in the final stage of the project). The way to achieve this performance is to improve the reception chain with Low Noise Amplifiers (LNA) and High-Gain Antennas for example.

The following are some of the requirements for the first vehicles of the project:

- *Data to be transmitted during flight:* The variables of interest during flight can be many and consequently result in a complex telemetry system. The most important parameters to be measured are the acceleration and angular velocity of the rocket on the x , y and z axes.
- *Data to be transmitted prior to launch:* In the moments prior to the launch, data transmission is also very important. Either in descending or ascending direction, to receive measurements or to send some commands to the board.
- *Design of the flight board:* To accommodate the electronic boards and everything related to it, the available space is cylindrical, of 10 cm in diameter by 30 cm in height, located at the top of the rocket (nose cone). Because of

these space constraints and the large acceleration values at which the components will be subjected during launch, it was thought of a circular design mounted transversely to the longitudinal axis of the rocket.

- *Antenna Design:* The transmitting antenna must be as omnidirectional as possible since the rocket will rotate all the time on its own axis (in order to stabilize its trajectory) and that should not affect the transmission. The simplest alternative to carry out this function is monopole antenna [1]. For the frequency band commonly used in this type of applications (420-470 MHz) the monopole antenna should have a length of 17 cm. This allows it to be housed on the nose cone of the rocket reducing the signal attenuation produced by the rocket itself. In [13] we can find other designs for transmission antennas and the comparison between them.

It should be noted that the correct functioning of the RF link is of vital importance to obtain the measurements in real time. This is necessary since in many cases it will not be possible to recover the rocket and the extraction of the data stored by the board.

The rest of the paper is organized as follows. In Section 2 we analyze different factors related to the communication link. Section 3 describes the commercial devices selected to implement each of the system blocks. Section 4 shows the first prototype of the telemetry board and the tests performed to verify the correct performance of the selected chips. Section 5 presents the design and construction of the final flight board. Section 6 details the link tests and results obtained. Finally, Section 7 details the conclusions.

II. LINK ANALYSIS

To begin with the design of the system it was necessary to set some parameters as the transmission frequency and type of modulation. A complete analysis of the link was then carried out to evaluate feasibility and to know the theoretical limits of it.

A. Frequency band

The 433 MHz band was used for the design proposed in this work. This decision was made considering two fundamental criteria: the compatibility of a large number of commercial RF transceivers with this range of the spectrum and the possibility of emitting considerable power levels without the need of complex permits or legal authorizations. Within this band there are different types of services that overlap and cause interference, therefore some care must be taken when implementing the link.

B. Modulation scheme

In order to implement the system with a simple and robust receiver, to avoid synchronization problems and to lower costs, we opted for a non-coherent modulation, as is usual in this type of applications. Digital FSK (Frequency-shift keying) modulation is the most used modulation for this class of systems and it can be carried out with commercial components of low cost and so that was the chosen option.

C. Link budget

The link budget allows us to ensure that the receiver has at its input, the minimum signal power to achieve a certain reliability and estimated error rate. Note that in this section the link budget is performed for a link distance of 100 km. Remember that for the first stages of the project the height reached by the rockets is an order of magnitude lower and therefore in this stages there is an additional margin of 20 dB for the link budget.

1) *Free space propagation:* In a system such as the one presented in this work, where the link is made between a rocket several kilometers in height and an earth station with its antenna pointing practically towards the rocket, we can assume that there is no interference by reflected waves and also there will always be line of sight between the transmitter and the receiver. With these assumptions we can use a free space transmission model and apply the Friis equation

$$P_{tel} = 32.4 + 20 \log(f) + 20 \log(D) \quad (1)$$

In Equation 1 we can see that the free space transmission loss (P_{tel}) depends on the frequency of operation f (measured in megahertz) and the distance of the link D (measured in kilometers). Replacing in the previous equation with the estimated values for the present link, where the maximum height that would reach the rocket is around 100 km, we obtain:

$$P_{tel} = 32.4 + 20 \log(433) + 20 \log(100) = 125 \text{ dB} \quad (2)$$

2) *Availability and Fade Margin FM:* Assuming a model containing only transmission losses in the free space may prove too ideal for a practical link. Therefore we must add an additional margin to contemplate all possible additional losses that may appear. This margin is called the fade margin (FM). Considering an availability (percentage of time in which the link operates without interruptions) of 97% recommended by the ITU for this type of systems [2], an operating frequency in the 433 MHz band and a maximum transmission distance of 100 km, we obtain that the necessary FM is approximately 3.35 dB [3].

Considering that the maximum permissible Equivalent Isotropically Radiated Power (EIRP) in this band is 10.7 dBm and that the transmission losses will be the free space path plus the fading margin of 3.35 dB recommended, we can calculate the power that will receive our ground receiver (P_{rx}) with an estimated gain for receiver antenna (G_{rx}) of 6 dB as indicated by Equation 3.

$$P_{rx} = \text{EIRP} + G_{rx} - P_{tel} - \text{FM} \geq \text{sensibility} \quad (3)$$

Replacing the values for our link in the above equation, the estimated received power is -111 dBm. This power value is at the limit of the minimum that many of the commercial devices need to be able to operate correctly (approximately -110 dBm). Therefore, the link was theoretically feasible taking the necessary care that the commercial device chosen for reception has a sensitivity greater than the calculated value.

Also note that with the additional margin of 20 dB for the first stages of the project, we should reach the first objectives without problems.

III. COMMERCIAL SOLUTIONS

The work aims to develop a low cost and fast response telemetry system without losing the reliability of the link. To achieve this objective the system was implemented with commercial devices. Commercial solutions chosen to implement each of the modules of the telemetry system are discussed in detail below.

A. Transducers

In a telemetry system, the transducers involved are used to design the communication link since the number and nature of them fixes the amount and speed of data to transmit. In this work, the variables that are measured are: acceleration in x , y , z axes (principally in the z axis, if this is the longitudinal axis of the vehicle), angular velocity in the x , y , z axes, altitude of the ship and temperature of the printed circuit board.

1) *Accelerometer/Gyroscope*: The MPU-6050 of IN-SENSENSE [4],[5] is an Inertial Measurement Unit (IMU) that combines a three axes accelerometer and a three axes gyroscope giving a total of six degrees of freedom. Both the gyroscope and accelerometer have 16-bit digital outputs and can be configured to have different full-scale range for each magnitude.

2) *Barometer*: The MS5607-02BA03 is the latest generation of TE Connectivity barometric pressure sensor [6]. This chip was selected to measure the altitude of the vehicle with a resolution of 20 cm in a range of 1 kPa at 120 kPa.

3) *Thermistor*: To sense temperature on the telemetry board the MCP9700 of Microchip [7] was chosen. The SOT-23-3 device package allows accurate measurement of temperature on the circuit printed board.

B. RF Transceiver

The transceiver selected for the system was the Texas Instruments CC1200 [8]. It is capable of reaching a maximum of 1250 kbps transmission rate and the output power is configurable to up to 14 dBm. The user can choose different modulation schemes (FSK, MSK, OOK) as well as different frequency bands (164-190 MHz, 410-475 MHz and 820-950 MHz) with a great adjacent channel rejection.

C. Flash memory

In order to store the samples collected by the sensors to be backed up for any inconvenience of the RF module, an additional memory was included in the board. Thus the N25Q256A flash memory of MICRON TECHNOLOGY [9] was selected. This memory has an SPI bus and has a storage capacity of 256 Mbits.

D. Microcontroller

In a telemetry system the microcontroller must have the computing capacity and enough peripherals to process and transfer all the variables of interest. An ARM-Cortex-M3 microcontroller named LPC1769 from NXP Semiconductors [10] was chosen for this project, because it is recommended for applications requiring a high level of integration and low power dissipation. The device reaches CPU frequencies of up to 120 MHz, which makes it a high-speed version within the LPC1700 family. In addition, it has a wide variety of peripherals such as SPI, I2C, UARTs, required for this application.

IV. DEVELOPMENT MODEL

The Development Model board (DM) is the first prototype designed and built to validate the correct performance of the chips selected for the proposed telemetry system. This board was implemented with different commercial modules containing the same chips that would be used in the final flight design. This allowed shortening the development times.

The DM board was implemented in a single layer design, taking care of the dimensions of tracks and holes in order to be able to manufacture it without having to resort to a PCB manufacturing company, which would increase the cost.

The basic function of the DM is to take samples of acceleration and angular velocity using the accelerometer/gyroscope module, send them to the microcontroller for conditioning and finally transmit them through the RF module at the same time as they are stored in flash memory as a backup.

A. Main components

This section presents a brief description of the main modules of the board.

1) *Stick LPCXPRESSO 1769*: This module is the core of the board. It contains the microcontroller chosen for the project, the LPC1769, along with all the necessary components for its operation. It also contains a USB interface to easily debug programs developed through a PC.

2) *GY-521*: The GY-521 board contains the accelerometer/gyroscope (MPU-6050), a voltage regulator, a crystal and the all the passive components necessary for the correct operation of the chip .

3) *N25Q256A flash memory*: This surface mount flash memory was placed on the DM board.

4) *EM-CC1200*: This evaluation module of Texas Instruments integrates the RF transceiver necessary for data transmission and reception (the CC1200) together with the antenna suitable for the frequency band to be used, a crystal oscillator and all the passive components required for the correct operation of the chip.

B. Prototype construction

The PCB was made in a single layer design. This allowed the board to have a simple design, thus enabling it to be 'homemade' manufacture. In the bottom layer of the DM board the surface mount devices (SMD) were housed and

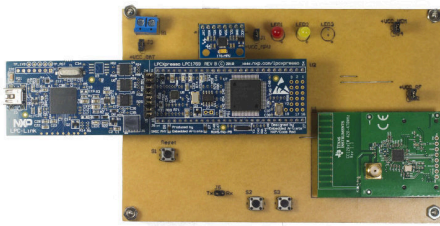


Fig. 2. Top view of the DM board with all its modules connected.

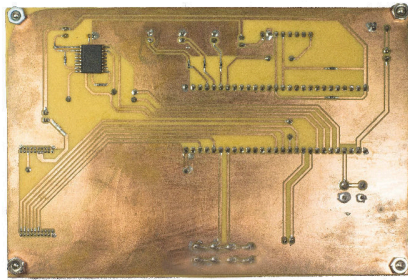


Fig. 3. DM board bottom view.

almost all of the routing was performed. On the top layer, the sockets for the modules and all the Through-Hole components (like terminal blocks, jumpers, LEDs and push-buttons) were mounted. Also, on the top layer, connections which were not possible to be made on the bottom layer were carried out. The manufactured and fully assembled prototype can be seen in the Fig. 2 and Fig. 3.

C. Software development and link tests

Once the DM board was fabricated, tests were performed in order to optimize the link performance. It began by transmitting data at short distances (a few meters) and with transmission rates below 50 kbps in order to improve the software of the board until finding the optimal configuration for each of its modules. With the passage of tests, the link range and the transmission speed were increased until reaching 1000 meters in line of sight with bit rates in the order of 100 kbps.

1) *Operating scheme:* To implement the DM board software, a free Integrated Development Environment (IDE) of NXP based on Eclipse was used, using C language and debugging the programs through the USB interface included in the LPCXpresso stick. A main program was implemented to acquire acceleration and angular velocity samples of the MPU-6050, to condition them through the microcontroller of the stick LPCXpresso and to transmit them in real time with the RF transceiver CC1200 at the same time that they were stored in the flash memory as backup.

2) *Lab tests:* Before carrying out the field tests, laboratory tests (at transmission distances of a few meters) were carried out between the DM board and one of the Texas Instruments

TABLE I
FINAL CONFIGURATION OF THE TRANSCIEVER

Parameter	Value
Modulation scheme	4-GFSK
Frequency deviation	50 kHz
Transmission rate	100 kbps
Packet length	60 bytes
Preamble	3 bytes
Sync word	32 bits
CRC	2 bytes
Error-correcting code	Enable
Output power	1 dBm



Fig. 4. 1 km link test developed in the hippodrome of La Plata city between the DM board and one of the TI evaluation boards (capture of Google Earth).

transceivers evaluation boards (TI) acquired. Using the software provided by that manufacturer [11], the CC1200 was configured with different values as it's shown in the following list:

- Modulation scheme: 2-FSK, 4-FSK, 2-GFSK, 4-GFSK and MSK.
- Frequency deviation: 10 to 100 kHz.
- Transmission rate: 25 kbps, 50 kbps and 100 kbps.
- Packet handling: packet length, headers, error-detecting and error-correcting codes.
- Output power: -40 to 10 dBm.

In [12] we can find similar tests performed to achieve the optimum configuration of other commercial transceivers. After the first stage of testing and analysis of the different configurations (comparing the number of errors, transmission rate and sensitivity in the receiver obtained with each one), the configuration shown in Table I was chosen to be loaded in the final software of the DM board.

3) *Outdoor tests:* Link tests were performed using the DM board as the transmitter and one of the Texas evaluation boards as the receptor. The tests were performed at one kilometer distance with line of sight between boards (see Fig. 4) and at 100 kbps data rate. Series of 5000 packets were transmitted in a lapse of time of 1 minute and with output powers of less than 1 mW. The results obtained were very satisfactory, achieving a packet error rate less than 1%.

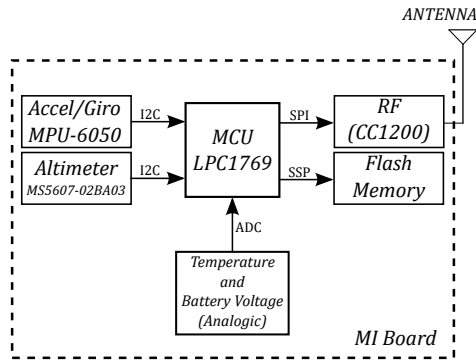


Fig. 5. EM board block diagram.

V. ENGINEERING MODEL

The Engineering Model board (EM) is the final flight board capable of performing data transmission from a rocket or experimental probe balloon. The EM board consists of different main blocks conformed of all chips detailed throughout the work. Fig. 5 shows a block diagram of the board where the interface used for communication with each module is indicated in each case.

A. Special design considerations

For the design of the EM board, a series of restrictions imposed by the system of which it will be a part of must be respected. The following list shows the main points that were considered prior to the actual design:

- Circular board shape to mount it directly on the body of the rocket from which it will transmit the telemetry.
- 3 mm diameter holes to fix the board to the rocket.
- Robust connectors to withstand significant mechanical stress.
- Connectors and holes near the edges of the board.
- Accelerometer/gyroscope mounted in the center of the board to simplify the processing of received data.
- Preferably surface mount (SMD) components to optimize available space and take advantage of the good RF qualities that they possess.
- Use of power and ground planes of each component of the board, dividing it into sectors that are fed independently.

B. Board construction

For the printed circuit of the EM, due to the quantity and complexity of the connections to be made, we opted for a four layers design. Two signal layers (enough to mount and connect all the components), a ground plane and a power plane. The dielectric material chosen to separate the signal layers from the power and ground planes was FR4 of dielectric constant $\epsilon_r \simeq 4$. To interconnect the different layers, vias were used with a suitable diameter for the type of components used and the tracks density of the board.

The width of the tracks in general is 8 mils with a separation between them of 16 mils in the optimum case. This size is due to the dimensions of the pins of main chips such as CC1200

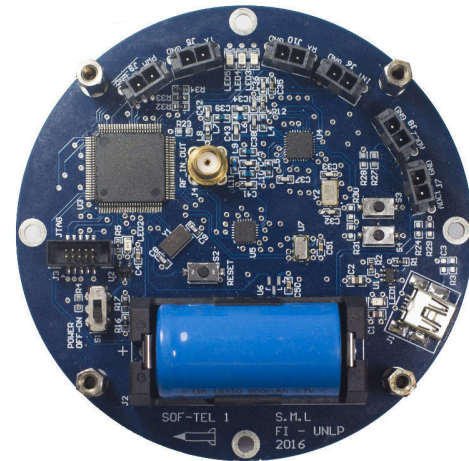


Fig. 6. Top view of finished EM board.

and LPC1769. For the connection of decoupling capacitors and other lines with greater power consumption, a track width of 15 mils was used, reducing the impedance of the same. On the other hand, lines such as those involved in the RF stage need to have a controlled impedance (50 Ω). To achieve this impedance, a width track of 19 mils was used. This was calculated with the LineCalc tool of the software Advanced Design System (ADS).

To power the board it was decided to include a Li-Ion 18350 rechargeable 900 mAh battery housed on the printed circuit, thus achieving independence from other systems present in the rocket or vehicle containing it. Before the selection of the battery, an analysis of the total consumption of the board was carried out. With this analysis it was estimated that the maximum current supplied by battery (with the transmitter at 10 dBm and 100 kbps) would not exceed 100 mA. So, with the Li-Ion 18350 battery, the board will have 9 hours of autonomy. This is more than enough to transmit data during the flight of the rocket and in pre-launch testing. The built and fully assembled board can be seen in Fig. 6.

VI. LINK TESTS AND RESULTS

Two link tests were performed to fully validate the design of the EM board. In both cases, one of the evaluation boards of Texas Instruments was used to receive the data transmitted by the EM board and send them to a PC for real-time visualization and subsequent processing. As explained throughout this paper, in the first stages of the project the height achieved by the rockets will not exceed 2 km. Considering this and assuming that we will always have line of sight between the rocket and the earth station, it was decided to perform the following link tests to validate the operation of the telemetry system. Note that for these validation tests a commercial receiver and a low-gain antenna (monopole antenna) were used.

A. 1 km test

A 1 km link test was carried out under the same test conditions detailed in Section IV-C3. Batches of 5000 packets

TABLE III
3 KM TEST RESULTS

Field	Value
Link range	3 km
Output power	4 dBm
Transmission rate	100 kbps
Average received power	-90 dBm
Packets received	4999
Packets lost	1
Packets OK	4992
Packets NOK	7
Packet error rate	0.16%

TABLE II
1 KM TEST RESULTS

Field	Value
Link range	1.02 km
Output power	1 dBm
Transmission rate	100 kbps
Average received power	-90 dBm
Packets received	5000
Packets lost	0
Packets OK	5000
Packets NOK	0
Packet error rate	0%



Fig. 7. 3 km link test developed in the city of Tandil (capture of Google Earth).

were transmitted with the EM board and were received with one of the evaluation boards of Texas Instruments. Fig. 4 illustrates the boards positions during test. The transmitted packets contained samples taken by different sensors of the board such as angular velocity, acceleration, altitude, battery voltage and board temperature. The results of the test are listed in Table II.

B. 3 km test

In this test, performed in the city of Tandil, the boards were located with a separation of 3 km as shown in Fig. 7. The packets were transmitted with the EM board from a hill in the outskirts of the city and were received with one of the Texas Instruments boards located on the terrace of a building in the town. As in the previous test, batches of 5000 packets were transmitted with the data taken by the sensors of the board.

The results were very satisfactory and are detailed in Table III.

This test confirmed the correct performance of the board and the possibility of further extending the link since the transceiver was not configured with the maximum allowed power (10 dBm approx.). Also the receiver and antennas used were not optimal. Future lines of work contemplate improvements to the receiver and in particular the receiving antenna. It will then be possible to perform link tests at a greater distance (approaching 100 km in line of sight) and in real conditions, either with a balloon or experimental rocket.

VII. CONCLUSION

A telemetry system capable of transmitting in real time different variables of interest from a rocket or experimental balloon was designed and developed. This system is able to operate at distances and transmission rates higher than other commercial devices and with a very low transmission power and battery consumption.

In principle, the most convenient frequency bands were investigated both from a legal and functional point of view. After this first phase, commercial solutions were chosen for the implementation of the system. Once the main components of the system were selected, a first prototype of telemetry board (DM board) was developed. With this prototype, different validation tests and software development tasks were carried out in parallel with the design and manufacture of the final flight board (EM board). After the construction of the latter, its operation was verified transmitting data with a rate of 100 kbps and over distances in the order of 3000 m.

The results obtained during the tests were very satisfactory and fully validated the proposed design.

REFERENCES

- [1] Constantine A. Balanis, *Antenna Theory: Analysis and Design*, 4th ed., January 2016.
- [2] ITU, *RS.1165-2 ITU-R Recommendation: Technical characteristics and quality criteria of radiosonde systems of the meteorological aids service in the 403 MHz and 1680 MHz frequency bands*, 2006.
- [3] Wayne Tomasi, *Sistemas de Comunicaciones Electricas*, 4th ed., DeVry Institute of Technology, Phoenix, Arizona, USA, 2003.
- [4] InvenSense, *MPU-6000/MPU-6050 Product Specification*, Rev. 3.4, 2013.
- [5] InvenSense, *MPU-6000/MPU-6050 Register Map and Descriptions*, Rev. 4.2, 2013.
- [6] TE Connectivity, *MS5607-02BA03 Barometric Pressure Sensor Datasheet*, 2015.
- [7] Microchip Technology Inc, *MCP9700/9700A Low-Power Linear Active Thermistor ICs Datasheet*, 2016.
- [8] Texas Instruments, *CC120X Low-Power High Performance Sub-1 GHz RF Transceivers User's Guide*, 2013.
- [9] Micron Technology, Inc, *Micron Serial NOR Flash Memory N25Q256A Datasheet*, Rev. 9/13, 2011.
- [10] NXP Semiconductors, *LPC176x/5x User manual*, Rev. 3.1, 2 April 2014.
- [11] Texas Instruments, *SmartRF Studio 7 Hands-on User Guide and Tutorial*, Rev. 1.2, August 2011.
- [12] G. Yang, K. Zhang, N. Meratnia and P. Havinga, *Finding Optimum Settings for a 433MHz Radio for Long Range Communication*, 2nd International Conference on Information Science and Control Engineering, pp. 239-244, 2015.
- [13] P. Sripho, S. Duangsi and M. Hongthong, *Comparison of antenna for DTI rocket telemetry system*, Second Asian Conference on Defence Technology (ACDT), pp. 105-110, 2016.

Aplicaciones de Software para Visualización y Monitoreo aplicables a Vehículos No Tripulados y Cohetes Sonda

Adrián Stacul
CITEDEF

Laboratorio de Técnicas Digitales
Buenos Aires, Argentina
Email: astacul@citedef.gob.ar

Sebastián Alvarez
CITEDEF

Departamento de Electrónica
Buenos Aires, Argentina
Email: salvarez@citedef.gob.ar

Resumen—Este trabajo describe el diseño, desarrollo e implementación de un conjunto de aplicaciones de software dedicadas a recibir datos externos y comunicarlos a múltiples clientes de monitoreo y visualización en tiempo real. El entorno tiene como finalidad realizar el registro, dentro de un puesto de tierra, de la información remota de una plataforma aeronáutica o aeroespacial. Este trabajo fue desarrollado para ser utilizado en ensayos de sistemas y sub-sistemas de vectores sonda y vehículos aéreos no tripulados del Instituto de Investigaciones Científicas y Tecnológicas para la Defensa (CITEDEF).

Palabras clave—Arquitectura de software, Tiempo real, Sistemas de puesto de tierra, UAV, Vectores Sonda.

I. INTRODUCCIÓN

Debido a la evolución constante de los sistemas electrónicos de a bordo, tanto en cohetes del tipo sonda como en vehículos aéreos no tripulados, se hace indispensable poseer un software robusto que monitoree de forma remota el estado de éstos. Consecuentemente el software relacionado aplicable a estas plataformas debe poseer una flexibilidad acorde a cambios constantes que puedan suceder en el ciclo de diseño de la plataforma electrónica. Este trabajo tiene como motivación implementar un conjunto de aplicaciones de software dentro de un puesto tierra, con la tarea de recibir y comunicar información telemétrica y realizar el procesamiento adecuado para el monitoreo remoto de una plataforma de vuelo. El desarrollo e implementación de un SDK (*Software Development Kit* – Kit de Desarrollo de Software) propietario le otorgó a la arquitectura servidor/cliente utilizada, la flexibilidad necesaria para que el tratamiento de datos pueda efectuarse de manera correcta, un claro ejemplo es la traducción de datos crudos a datos físicos reales. El diseño modular aplicado genera un eficiente uso de los recursos propios de los equipos utilizados para ejecutar el software [1].

II. DISEÑO

Como punto inicial en el diseño de este sistema, el servidor debe actuar como un enlace bidireccional entre el hardware remoto y el usuario del puesto en tierra, de tal manera que la latencia en la transmisión de la información sea mucho menor

que el tiempo de la respuesta natural de la planta. A su vez, este servidor debe comunicarse de forma paralela con los diferentes clientes conectados a éste (ver Fig. 1).

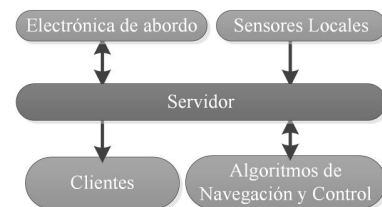


Figura 1. Diseño inicial del sistema

Manteniendo el concepto de modularidad aplicado en el diseño del sistema, se desarrollaron múltiples aplicaciones discriminando las tareas a cubrir. Cada proceso está asociado a una tarea específica y los mismos se intercomunican entre ellos. Esta característica aprovecha los recursos de un sistema al máximo, extrayendo el mayor beneficio de los actuales entornos compuestos de procesadores multitareas.

II-A. Servidor de Datos

El servidor de datos fue concebido para concentrar, relacionar y comunicar los diferentes módulos conectados al sistema. Cumple la función de intérprete entre diferentes protocolos de comunicación y manipulación de datos. Conociendo los formatos de los datos involucrados, recibe constantemente tramas provenientes de múltiples conexiones, decompila las mismas extrayendo los datos relevantes y adapta dicha información a nuevos grupos de datos, para finalmente entregarla de manera segura a múltiples clientes. Estos clientes pueden o no, coexistir dentro de este entorno. El servidor de datos (ver Fig. 2) está compuesto por un conjunto de módulos intercomunicados denominados ‘nodos’. El nodo principal encargado de la gestión de procesos monitorea al resto de los nodos secundarios, quienes tendrán acciones definidas de:

- Lectura de puertos
- Enrutamiento
- Conmutaciones y de-conmutaciones [2]

■ Interfaces

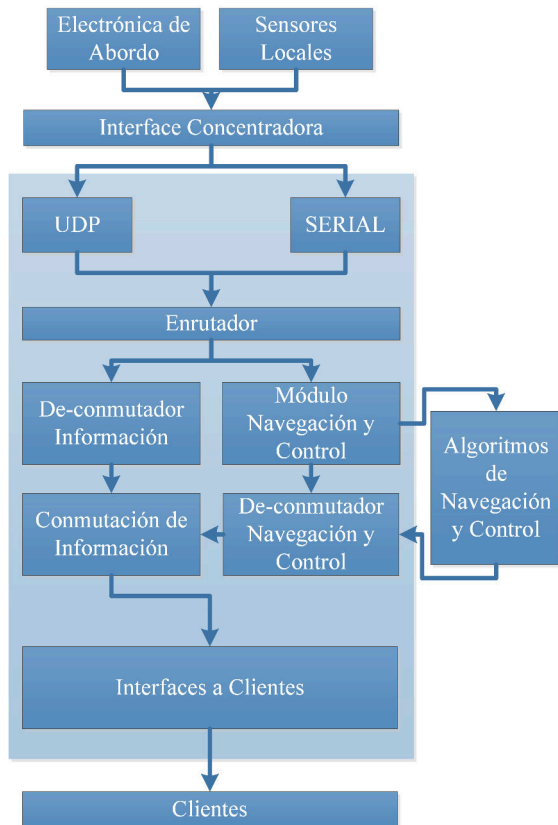


Figura 2. Estructura del servidor de datos

El sistema completo es capaz de escuchar la trama de datos telemétricos en tierra y decompilarla en diferentes canales de datos remotos, los cuales contienen la información física de los sensores a bordo y otros estados del vehículo. A su vez, también toma la información de los datos de sensores locales. Toda la información presente en el servidor, se compila en un nuevo formato y se entrega a los clientes que estén suscriptos al servidor bajo un protocolo de comunicación preestablecido. En la Fig. 3 se detallan todos los nodos que componen al servidor de datos y su relación de comunicación entre ellos.

II-A1. Nodo RAW: Este nodo es el encargado de escuchar la trama de datos cruda generada (denominada Trama RAW) y enviada por la electrónica de a bordo. Esta trama consiste de un paquete de datos de 64 bytes de longitud, una palabra de sincronismo y un control de errores.

El software fue diseñado para enviar dicho paquete de información al módulo de algoritmos de navegación y control sin generar retardos significativos que alteren la comunicación. En una rama de procesamiento paralela el módulo decompilará la trama basándose en los canales asignados y generará una estructura de datos interna que se llenará con la información de dichos canales. Este nodo mantiene una comunicación directa e inmediata con el Nodo OBJ, al cual le envía constantemente y en tiempo real la información decompilada.

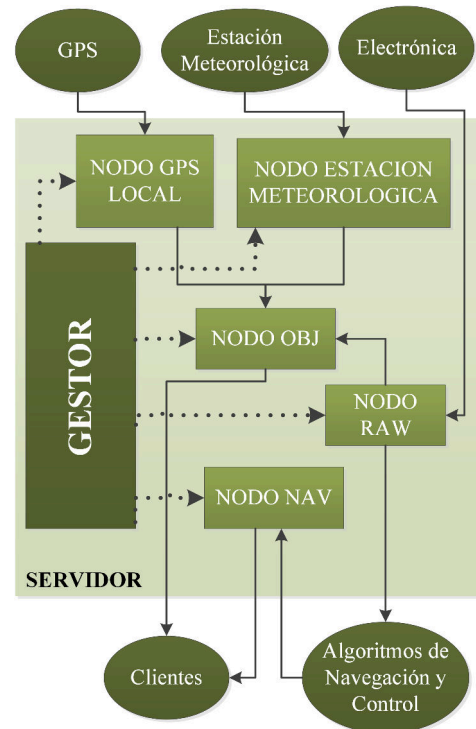


Figura 3. Relaciones internas en el servidor y sus nodos.

II-A2. Nodo GPS Local: Esta unidad de software tiene como función la lectura secuencial de tramas NMEA generadas por un sensor GPS conectado a la estación terrena. Discrimina las tramas configuradas, las reenvía a los clientes seleccionados y se comunica con el Nodo OBJ sirviéndole la información de manera inmediata.

II-A3. Nodo Estación Meteorológica: El Nodo Estación Meteorológica cumple la función de interpretar los datos recibidos desde una estación meteorológica local conectada al puesto a tierra. Decompila los canales de datos provenientes de la trama, y genera una sub trama de datos locales. Esta nueva sub-trama es enviada al Nodo OBJ. El empleo de este nodo y la relación del mismo con el resto de los nodos son de carácter optativos.

II-A4. Nodo OBJ: Este software es el encargado de concentrar la información recibida de los Nodos RAW, GPS Local y Estación Meteorológica, y compilarla en una nueva trama denominada RXOBJ. Paso siguiente a la generación de la trama, se dispone a enviar una copia a cada uno de los clientes suscriptos al Nodo. Dicha acción se lleva a cabo gracias a la lectura de la configuración existente en el archivo de configuración XML [3].

II-B. Clientes de Monitoreo y Registro

Presente la necesidad de representar gráficamente todos los datos que ingresan a el servidor, fue necesario desarrollar un conjunto de aplicativos clientes que permitan la visualización de los mismos. Se diferenciaron tres grupos de clientes basándose en el objetivo de cada uno (ver Fig. 4).

1. El grupo de Simuladores, que facilitan el uso de la plataforma sin la presencia de la electrónica, reproduciendo ejercicios anteriores o simplemente emulando tramas
2. El grupo denominado Persistencia, encargado de almacenar en disco toda la información presente en el servidor.
3. Los Clientes de Monitoreo encargados de representar en pantalla el histórico o inmediato de la información presente en el servidor.

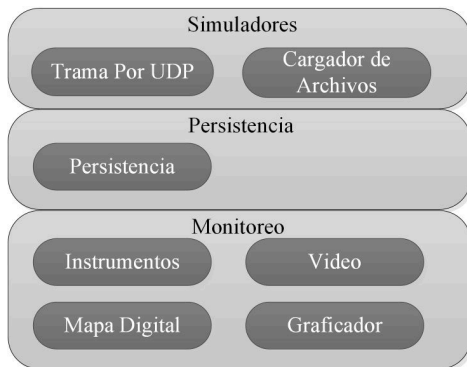


Figura 4. Nombre de clientes adoptados según su grupo

II-B1. Trama por UDP: Este software es capaz de enviar las múltiples tramas involucradas en el funcionar normal del sistema por Ethernet. El comportamiento del mismo es configurable a través de un archivo XML. Entre los parámetros que se pueden modificar se encuentran el destino de envío de la información generada y el tiempo de espera entre los paquetes a enviar, además se puede modificar el contenido de las tramas a enviar en tiempo real. Su diseño implementa técnicas de hilos múltiples (*multithreading*) para asegurar que el retardo en la generación y envío de tramas sea despreciable, y de esta forma asemejarse a una simulación real de los medios físicos emulados. El hecho de conocer los datos utilizados para estimular al servidor, permite evaluar el correcto funcionamiento de futuros clientes que se conecten al mismo.

II-B2. Cargador de Archivos: Este software permite reproducir ejercicios previamente registrados en el sistema. Su objetivo es brindar una herramienta para leer un archivo de tramas crudas, previamente capturado por el cliente de Persistencia y retransmitirlas al Nodo RAW del servidor emulando el funcionamiento de la electrónica a bordo. Este software cuenta con la capacidad de avanzar y retroceder sobre el ejercicio que está siendo reproducido para poder post-evaluar una misión en búsqueda de parámetros o sucesos particulares.

II-B3. Persistencia: El sistema de persistencia es el encargado de realizar el volcado a disco de todas las tramas presentes en el sistema de forma continua. Genera un archivo para cada grupo de información presente en el sistema. Todas las tramas decompiladas y guardadas en disco por este software están sincronizadas en la misma base de tiempo, permitiendo cruzar datos para post-evaluación de los mismos.

II-B4. Graficador: Este software es un monitor de variables físicas en tiempo real. Las magnitudes de las variables son

visualizadas en múltiples ejes cartesianos en forma periódica y continua. La aplicación tiene la capacidad de graficar 3 canales por cada eje graficado, generando por ejemplo, 18 representaciones, si el caso corresponde al muestreo de 6 ejes en una misma pantalla. Este accionar es configurable por medio de la modificación de un archivo XML donde también se podrá configurar otros parámetros como el máximo y mínimo de escala de cada eje cartesiano (ver Fig. 5).

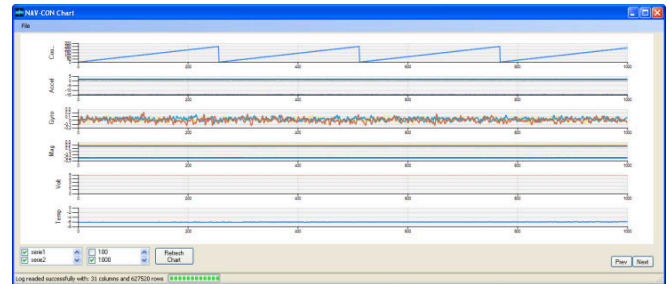


Figura 5. Vista del software "Graficador"

II-B5. Instrumentación: La interfaz gráfica de este cliente simula la aviónica de una aeronave estándar [4]. Al igual que el resto de los aplicativos, este software modifica y adapta su funcionamiento a las necesidades del usuario, por medio de la modificación de un archivo de configuración XML (ver Fig. 6).

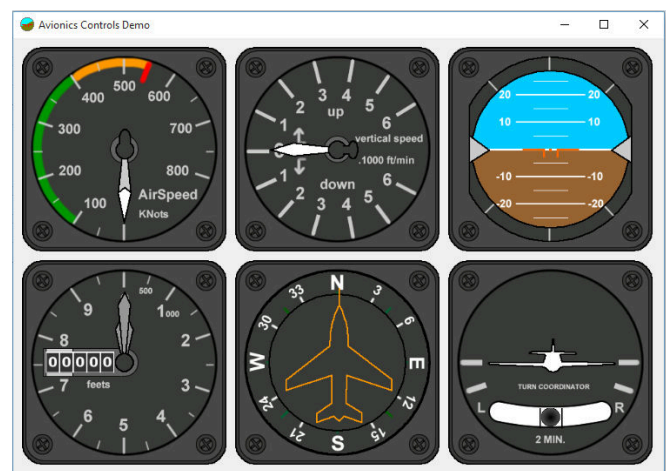


Figura 6. Vista del software "Instrumentación"

II-B6. Mapa Digital: Este software tiene la función y objetivo de graficar toda la información cartográfica del sistema. Agrupa los diferentes módulos GPS existentes así como también, muestra la información de posición del vehículo generada por los algoritmos de navegación. De esta forma, se puede utilizar este software para visualizar la posición del puesto tierra, la de la electrónica de abordo y la realizada por los algoritmos de navegación, en una misma pantalla. Toda la información mencionada anteriormente se impacta en tiempo real sobre una carta digital (ver Figura 7).

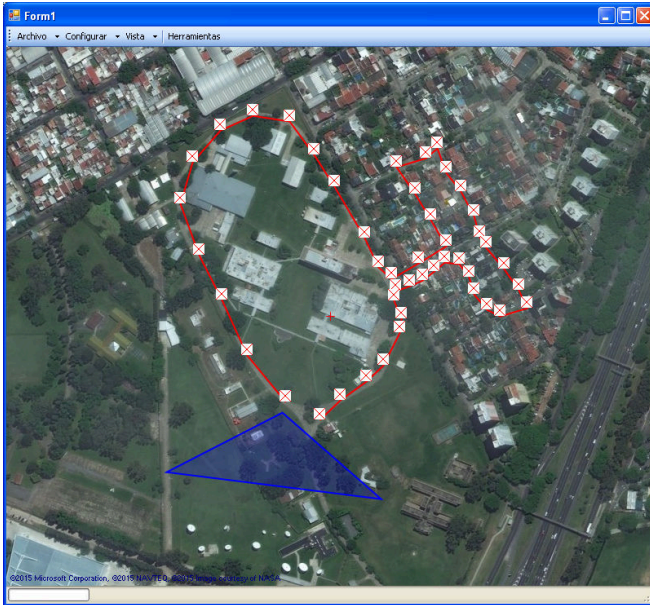


Figura 7. Vista del software “Mapa Digital”

II-B7. Video: Esta aplicación tiene como objetivo la captura y grabación de video en tiempo real, donde cada uno de los fotogramas proveniente de una capturadora externa de video está sincronizado con la información de la hora de un GPS local de alta precisión. Dicha sincronización se realiza en tiempo real y el software posee el dinamismo de adaptarse a diferentes resoluciones de video.

III. IMPLEMENTACIÓN

El entorno desarrollado es capaz de ser implementado en cualquier tipo de centro de cómputos a modo de estación terrena, que puede estar conformado por una única computadora, o en caso de ser necesario por un conjunto de estas, distribuyendo el procesamiento y las aplicaciones de monitoreo en diferentes puestos, o simplemente duplicando los aplicativos en diferentes máquinas para una redundancia en el control operativo. La etapa de desarrollo e implementación se realizó utilizando el entorno Visual Studio [5], para Windows 7 en adelante, generando de esta forma una compatibilidad con cualquier computadora moderna que cumpla con los requerimientos detallados en el Cuadro I. Debido a que se requiere una versatilidad de software amplia y se busca una migración ágil se desarrollaron todas las aplicaciones de formato portable y de escritorio, permitiendo de esta forma que al remplazar el hardware se pueda mantener una configuración de software ya establecida.

Se utilizaron C# y Delphi [6] como lenguajes principales para desarrollar la mayoría de las aplicaciones. La implementación de las librerías .NetFramework [7] facilitó la incorporación de las herramientas necesarias para desarrollar una plataforma ágil y de fácil mantenimiento, condiciones que resultaron favorables para un trabajo multidisciplinario

Tabla I
REQUERIMIENTOS DEL SOFTWARE

Mínimos		Recomendados	
CPU	Dual-Core	CPU	Quad-Core
RAM	4GB-DDR2	RAM	8GB-DDR3
Video	256DDR2 Mem Compartida	Video	1GB DDR5 Dedicada
Disco	>30GB en aquella máquina que corra el software de Mapa Digital		
Monitor	17"	Monitor	2x21"

donde se presentaron constantes cambios en los protocolos de comunicación y en la lógica interna de las aplicaciones.

Se aplicó una arquitectura de procesamiento distribuido y paralelo [8] en múltiples procesos, donde se instancian aplicaciones de escritorio en el mismo entorno de forma paralela. Esto facilita el trabajo, reduciendo el tiempo y el esfuerzo cuando debe actualizarse el funcionamiento de algún sector o módulo de software.

Se utilizó el formato de archivos XML como estándar para los archivos de configuración de todos los aplicativos desarrollados, donde cada uno mantiene una estructura interna común al resto. De esta forma, el sub-módulo de software encargado de procesar estos archivos de configuración XML [9] es común en los diferentes programas desarrollados. Otro de los aspectos fundamentales sobre los que se basó la arquitectura del sistema fue el modelo Cliente-Servidor aplicado en la distribución de tareas. La diferenciación en la designación de recursos permitió discriminar la etapa de tratamiento, adecuación y gestión de datos (Servidor), con la de visualización, actuación y utilización de los mismos (Clientes).

III-A. Arquitectura Servidor de Datos

El servidor se desarrolló sobre la base de un sistema de aplicaciones intercomunicadas, donde cada una consta de un grupo de algoritmos que resuelven una tarea específica basándose en el tipo de datos que procesa cada nodo. De esta forma, diferentes estímulos (tramas provenientes de diferentes módulos) son tratados de forma simultánea e individual, y confluyen en otro nodo, el cual concentra estos datos, los procesa y se encarga de comunicarlos en un formato nuevo a los clientes.

III-B. Metodología de Comunicación entre Procesos

Fue necesario resolver dos tipos de metodologías de comunicación entre procesos, la comunicación presente entre los nodos internos del servidor, y la comunicación entre los diferentes nodos y los módulos clientes suscriptos al servidor. La comunicación entre los nodos internos se resolvió con la tecnología Named Pipes [10] de IPC¹. Esta última permite comunicar un receptor asíncrono a modo de servidor con múltiples emisores funcionando como clientes. Esta técnica

¹La comunicación entre procesos (IPC – Inter-Process Communication) es un mecanismo que permite a los procesos comunicarse y sincronizarse entre sí, normalmente a través de un sistema de bajo nivel.

es considerada como una de las más veloces para comunicar procesos, siendo su principal limitante que los procesos deben residir en el mismo entorno. En cambio, la comunicación entre los módulos clientes y el servidor, se realizó con la implementación de sockets UDP.

III-C. Patrones de Diseño

En el desarrollo de software moderno resulta imprescindible la implementación sistemática de patrones de diseño para poder generar un proyecto de software responsable y mantenible en el tiempo. El software desarrollado para este proyecto cuenta con la implementación de varios patrones de diseño conocidos [11]. En las primeras etapas de desarrollo, se llevó a cabo un sub-sistema de *Logger* implementando el patrón *Singleton*. Resulta indispensable en sistemas amplios mantener una instancia de un único proceso de grabado de datos en disco, para hacer un debug confiable. Se implementó el patrón de diseño *Factory* conjuntamente con interfaces para realizar constructores abstractos, esto facilita el dinamismo a la hora de instanciar entradas al sistema. El mismo objeto puede instanciar un puerto de entrada físico o un socket UDP [12]. Por último, el uso de los patrones de diseños *Observer* y *Callback* fue necesario en la etapa de comunicación interna del software, para que una entrada se distribuya en múltiples objetos.

III-D. Conclusiones

En este trabajo, el diseño, desarrollo e implementación de un SDK propuesto a dado como resultado un conjunto de aplicaciones de software modulares, especializadas en el procesamiento y monitoreo en tiempo real de información telemétrica. Esta arquitectura planteada es responsable del procesamiento distribuido, de la intercomunicación de procesos y de la persistencia de datos. En la actualidad este sistema se utiliza para monitorear maniobras en vehículos no tripulados. Debido a que se consiguieron resultados exitosos en el monitoreo, se procedió a implementar todo el conjunto de software en un puesto de tierra de misión crítica para el monitoreo de futuros lanzamientos de cohetes sonda.

AGRADECIMIENTOS

El presente desarrollo fue realizado bajo un programa de proyectos con la temática de vehículos aéreos no tripulados bajo la dirección del Ing. Edgardo Comas y el Ing. Daniel Pastafiglia, a quienes nos gustaría expresar nuestro agradecimiento por hacer posible este trabajo. A su vez, a todo el personal del Laboratorio de Técnicas Digitales y el Laboratorio de Navegación de CITEDEF quienes ayudan continuamente en el proyecto con tareas de diseño de hardware y software.

REFERENCIAS

- [1] T. Kekceci, B. C. Ustundag, M. A. Guney, A. Yildirim, and M. Unel, "A modular software architecture for uavs," 2013.
- [2] L3-COM Telemetry-west. (2013) Telemetry tutorial. [Online]. Available: www.2.1-3com.com/tw/telemetry_tutorial/r_decommutation.html
- [3] H. M. Deitel and P. J. Deitel, *Como Programar En C#, 2007*, ch. Lenguaje De Marcado Extensible (XML).
- [4] ———, *Como Programar En C#, 2007*, ch. Concepto De Interfaz Grafica De Usuario.
- [5] Microsoft. (2015) Visual studio 2015. [Online]. Available: <https://www.visualstudio.com/>
- [6] Embarcadero. (2010) Rad-studio. [Online]. Available: <https://www.embarcadero.com/products/rad-studio>
- [7] Microsoft. (2016) .net documentation. [Online]. Available: <https://docs.microsoft.com/en-us/dotnet/>
- [8] F. G. Tinetti and A. D. Giusti, *Procesamiento Paralelo - Conceptos De Arquitecturas Y Algoritmos*. La Plata: Editorial Exacta, 1998.
- [9] W3C. (2013) Extensible markup language (xml). [Online]. Available: www.w3.org/XML
- [10] Microsoft. (2017) Named pipes. [Online]. Available: msdn.microsoft.com/en-us/library/aa365590.aspx
- [11] R. Helm, R. Johnson, and J. Vlissides., *Design Patterns: Elements of Reusable Object-Oriented Software*. Pearson Education, Addison Wesley, 2005.
- [12] J. Postel, *Internet Standard - RFC 768 - User Datagram Protocol*, 1980. [Online]. Available: www.rfc-es.org/rfc/rfc0768-es.txt

Ataque a implementación embebida de AES-128 aplicando Análisis de Correlación de Potencia

Gonzalo Ávila Alterach

Facultad de Ingeniería - Universidad de Buenos Aires
Ciudad Autónoma de Buenos Aires, Argentina
gonzaloavilaalterach@gmail.com

I. INTRODUCCIÓN

El análisis de potencia es un caso particular de los denominados ataques de canal lateral. En estos no se ataca a un algoritmo criptográfico en sí, sino a su implementación física. Se infieren datos secretos a partir de observaciones del consumo de potencia del dispositivo que implementa la función criptográfica. Este consumo varía en función de los datos procesados, filtrando información.

Estos ataques son importantes en la actualidad ya que existe una variedad de dispositivos embebidos cuya protección está basada en algoritmos criptográficos. A modo de ejemplo, existen múltiples microcontroladores que poseen *bootloaders* seguros, que solamente permiten cargar imágenes encriptadas con claves presentes dentro de los mismos. Aunque la función criptográfica sea computacionalmente segura, si una implementación física filtra parte de la clave, es posible encontrarla con algún ataque como el que se describe. Incluso en casos no exitosos se puede disminuir el espacio de claves.

El Análisis de Correlación de Potencia o *CPA* (del inglés, *Correlation Power Analysis*) es un tipo de análisis de potencia. Se basa en tener un modelo de consumo de potencia, que permita predecir el consumo que tiene el dispositivo, en función de los datos que está procesando. El modelo más sencillo es el basado en el peso de *Hamming*[1], que supone que la corriente medida en cada instante está correlacionada con la cantidad de *bits* encendidos de algún registro del dispositivo. Aplicando dicho modelo, se predice el consumo de potencia para cada hipótesis de clave y luego se calcula la correlación entre la predicción y las observaciones para cada instante. Las hipótesis con mayor correlación tienen alta probabilidad de ser verdaderas. Es necesario evitar que exista correlación para todas las hipótesis de clave. Por ejemplo, si consideramos el algoritmo *AES* (*Advanced Encryption Standard*), se pueden atacar a las salidas de las cajas de sustitución (*S-Boxes*) que, por definición, son alineales. De esta forma, sólo algunas hipótesis tendrán alta correlación.

II. DETALLES DE IMPLEMENTACIÓN

Se realizó una prueba de concepto del ataque a la implementación del algoritmo de cifrado *AES-128* en el microcontrolador *ATxmega16A4*. El microcontrolador se comunica con una computadora a través de la *UART* (mediante un conversor *USB-Serie TTL*), como se observa en la figura 1.

El microcontrolador recibe bloques de 16 *bytes*, los cifra con una clave secreta y finalmente, los envía a la computadora. Para simplificar el ataque, se agregó una señal de disparo, que se activa antes de comenzar con un cifrado. Se usó un cristal externo de 8 MHz en lugar del oscilador interno del microcontrolador, para minimizar las variaciones temporales entre capturas.

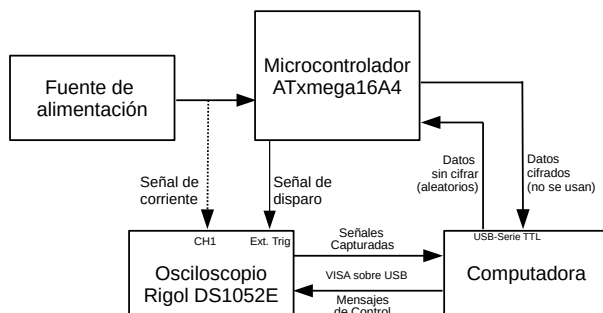


Figura 1. Banco de pruebas utilizado para el ataque.

El *firmware* se compiló usando *avr-gcc*. Se utilizó una implementación del algoritmo *AES-128* de dominio público[2].

Las capturas se realizaron mediante un osciloscopio digital *Rigol DS1052E*, con dos puntas pasivas de tensión *RP1100*. Para capturar la corriente que circula por el microcontrolador, se agregó una resistor en serie con la masa, como se observa en la figura 2. Se eligió un valor alto de resistencia (100 Ω) evitando, de esta forma, que la amplitud de la señal fuera muy pequeña. De haber elegido un valor mucho más alto, el dispositivo podría haber dejado de funcionar por disminución de la tensión de alimentación. Como alternativas para mejorar la relación señal-ruido, se pueden usar puntas de corriente, puntas diferenciales de tensión[3] o de campo eléctrico para realizar las capturas.

Considerando que los capacitores de desacople disminuyen la cantidad de información filtrada (suavizan la señal y eliminan los picos de corriente que se desean capturar) se posicionó a la resistencia de medición después de los mismos. La hipótesis del consumo de corriente se eligió de forma tal que un valor de tensión medido más bajo implica una mayor corriente, y se corresponde con mayor número de *bits* encendidos.

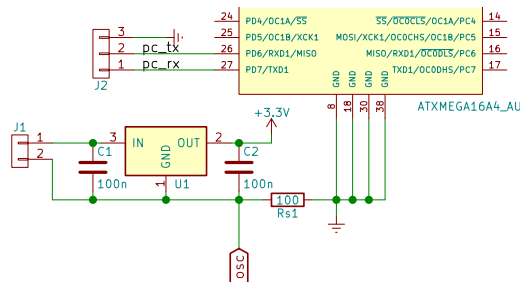


Figura 2. Medición de la corriente del microcontrolador usando resistencia *shunt* en masa.

Para realizar el ataque, existen diversos programas[4], pero no se encontró uno que soporte directamente al osciloscopio mencionado, por lo que se desarrolló uno propio[5]. El *software* configura automáticamente al osciloscopio para que capture la señal y envía bloques aleatorios de 16 bytes al microcontrolador para que sean cifrados con AES. La corriente usada es capturada, y se procesa para encontrar el valor más probable para cada *byte* de la clave, calculando los coeficientes de correlación correspondientes[6]. La comunicación con el osciloscopio se realiza por *USB* a través de la biblioteca *VISA* y los comandos propios del mismo[7].

Se configura al osciloscopio para que realice capturas de un millón de muestras (usando el modo *long memory*) y que muestree a 500 MSa/s. Esto equivale a 62,5 muestras por cada ciclo de *clock* del microcontrolador, y un tiempo máximo de captura de 2 ms. Para reducir la cantidad de datos a procesar, no es conveniente cambiar la escala de tiempo del osciloscopio (que agrega *aliasing* y perjudica al ataque), en cambio, es más adecuado realizar una decimación con un factor de 16 luego de la captura. Esto evita la aparición de *aliasing* debido a que esta operación implementa un filtro pasa bajos.

Además, a cada captura se le resta su media temporal para disminuir la influencia del ruido de 50 Hz, que se manifiesta como un *offset* (de valor similar en todos los puntos).

Se realiza también una alineación entre las señales, aplicando el desplazamiento que maximiza su correlación.

III. RESULTADOS

El ataque usando *CPA* funciona exitosamente: se logra extraer la mayoría de los *bytes* de la clave de forma correcta utilizando una baja cantidad de capturas: alrededor de 100, similar al número encontrado en otros trabajos[8].

Luego de preprocesar las muestras, por cada *byte* de la clave, el programa indica su valor más probable, el coeficiente de correlación máximo y en qué instante sucedió este último. En la figura 3, se observa cómo la tasa de éxito del ataque incrementa a medida que aumenta la cantidad de capturas, mostrando la posición de la clave correcta en la lista ordenada según el coeficiente de correlación. Esta posición también es llamada *PGE* (*Partial Guessing Entropy*). Una *PGE* menor implica que un atacante tiene que realizar menos pruebas, en promedio, para recuperar correctamente un cierto *byte* de la clave.

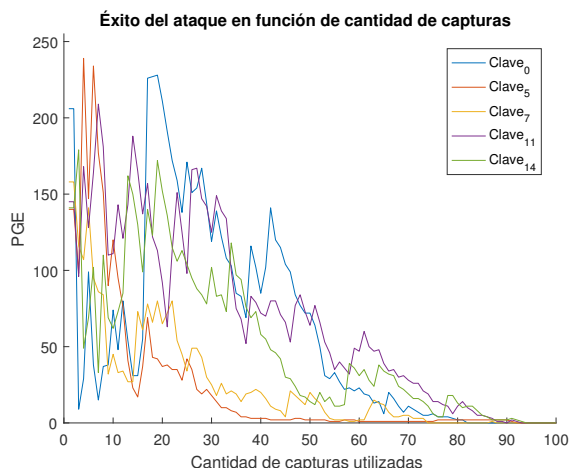


Figura 3. Gráfico de *PGE* (*Partial Guessing Entropy*) en función de la cantidad de capturas utilizadas, para algunos bytes de la clave. La *PGE* corresponde al número de hipótesis incorrectas que tienen correlaciones mayores que la correspondiente a la clave real.

El factor que limita la cantidad de capturas por segundo que se pueden realizar es el osciloscopio, ya que tiene tiempos de procesamiento considerables: aproximadamente 1 segundo desde que se activa el modo *run* hasta que puede capturar una señal y en promedio 3200 ms en transferir un millón de muestras de 8 bits hacia la computadora.

IV. CONCLUSIÓN Y TRABAJO A FUTURO

Se logró realizar eficazmente el ataque a la plataforma descrita, extrayendo la clave correcta mediante las capturas de la corriente usada por el microcontrolador. Se necesitaron agregar una señal de disparo para que el osciloscopio comience a muestrear siempre en el mismo instante y un cristal externo como fuente de *clock*, para disminuir las variaciones temporales entre distintas capturas.

Como trabajo a futuro, se podrían implementar contramedidas que permitan disminuir las probabilidades de éxito de los ataques, verificando su funcionamiento correcto. Además, mediante otras estrategias de captura se podrían atacar a implementaciones de algoritmos criptográficos en *hardware* o a microcontroladores más rápidos.

REFERENCIAS

- [1] Brier, Clavier, Olivier, *Correlation Power Analysis* [Online] Disponible: Disponible: <https://www.iacr.org/archive/ches2004/31560016/31560016.pdf>
- [2] Kokke, *Small portable AES128 in C* [Online] Disponible: <https://github.com/kokke/tiny-AES128-C>
- [3] O'Flynn, *Power Analysis for Cheapskates* [Online] Disponible: <https://goo.gl/Hc59vP>
- [4] NewAE, *ChipWhisperer* [Online] Disponible: <https://newae.com/tools/chipwhisperer>
- [5] Ávila Alterach, *análisis-potencia* [Online] Disponible: <https://github.com/gzalo/analisis-potencia>
- [6] NewAE, *Theory - Correlation Power Analysis* [Online] Disponible: <https://newae.com/sidechannel/cwdocs/theory.html>
- [7] Rigol, *DS1000DE Programming Guide* [Online] Disponible: <http://www.rigol.eu/products/digital-oscilloscopes/ds1000e/ds1052e/>
- [8] O'Flynn, Chen, *Side Channel Power Analysis of an AES-256 Bootloader* [Online] Disponible: <https://eprint.iacr.org/2014/899.pdf>

■
www.sase.com.ar

9 al 10 de Agosto de 2017

Facultad de Ingeniería | UBA
Buenos Aires, Argentina

