International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 9, No. 1, March 2020, pp. 1~11
ISSN: 2089-4864, DOI: 10.11591/ijres.v9.i1.pp1-11

Filtering and acquisition of serial data frames using xilinx system generator

Adrián Stacul
Facultad de Posgrado, Universidad Tecn´ologica Nacional (UTN), Argentina
Instituto de Investigaciones Cient´ıficas y T´ecnicas para la Defensa (CITEDEF), Argentina
Email: astacul@citedef.gob.ar

Article Info

Article history:

Received Feb 27, 2019
Revised Oct 6, 2019
Accepted Nov 25, 2019

Keywords:

Bit-synchronizer F
Ground station
PCM frame
PGA
System generator

ABSTRACT

The main purpose of this paper is the design, development and implementation of a PCM bit-synchronizer based on a System Generator and Simulink model. 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.

Corresponding Author:

Adrián Stacul,
Laboratorio de T´ecnicas Digitales,
Instituto de Investigaciones Científicas y Técnicas para la Defensa,
San Juan Bautista de La Salle 4397 - Villa Martelli, Buenos Aires Argentina.
Email: astacul@citedef.gob.ar

1. INTRODUCTION

Over the past decade, R&D in sounding-type vectors and UAVs (Unmanned Aerial Vehicles) has enjoyed exponential growth in several disciplines: aeronautical systems, applied mechanics, on-board electronics, ground stations, real-time signal processing, etc. In this work, we will focus on ground-based signal processing methods to acquire PCM signals and distribute telemetric information to multiple monitoring clients [1, 2]. Both unmanned aerial systems and sounding rockets require a ground station for the acquisition of telemetric signals and real time data processing whether for the control and monitoring of the mission or for the evaluation of the different scientific experiments installed on the platform [3]. 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 [4]. 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 [5]. 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).

Some of the first bit-synchronizers systems was based on the generation of spectral line components by nonlinear filtering of the received bit stream, and extracting the line by a digital phase-locked loop (PLL) and data detection was realized by a digital matched filter (DMF) [6]. The implementation of these modules evolved over time with the need of speed from the utilization of TTL gates, then using microcontrollers, CPLDs and currently using a combination of Analog-Digital converters (ADC) [7] and FPGA devices to perform signal processing and data analysis applied for i.e. to complex navigation algorithms [8]. The motivation of this work was to exclude the ADCs to be able to use only low-cost FPGAs and to perform an analysis of the procurement behavior and associates by building arrays of data to images. The PCM signal input is still analog and enter to a digital-input of FPGA. This model converts directly the noisy analog signal to a Digital PCM (DPCM), perform a bit-synchronization and send the data to a PC through an USB connection [9, 10]. On the other hand, the cost factor of trademark products was analyzed. All these are licencedsoftware, and are very expensive, and very difficult to adapt to scientific designs and developments where changes are permanent. This module developed is very low cost.

2. DESIGN

In order to achieve PCM signal regeneration, it is necessary to filter and observe the permanent regime state of the signal. The level (zero or one) at the midpoint of each bit is maintained despite the noise. Figure 1 shows the difference between the original transmitted signal and the received signal (note that the latter is band-limited, so the abrupt flanks no longer exist). The complete model (Figure 2) shows two instances of the acquisition system: first, the filtering model (where you enter an ideal PCM frame with noise added) and second, the acquisition model (in charge of the synchronization).

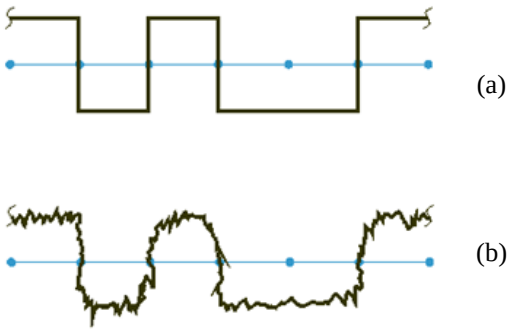

Figure 1. (a) Original signal transmitted and (b) Received signal reduced in bandwidth w/additional noise

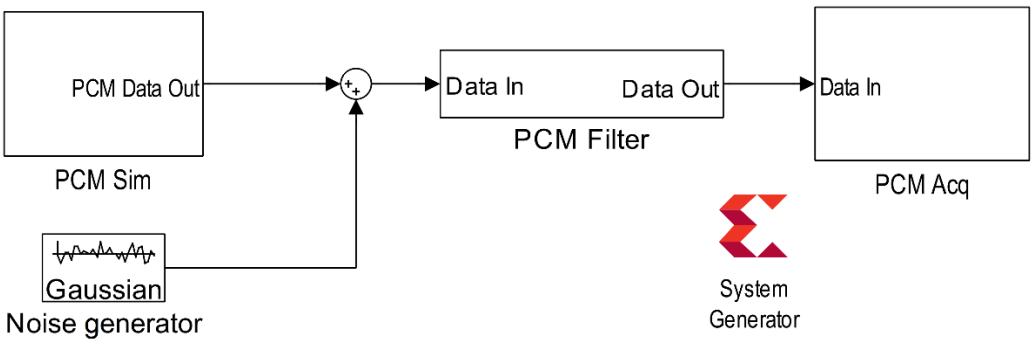

Figure 2. Complete PCM model with added noise, filtering and PCM acquisition

3. DEVELOPMENT

The model with System Generator [11] provides a high level tool for the development of high performance systems using Xilinx devices with FPGA technology, and thus defines and characterizes logic circuits to fulfill a specific function. The main difference between any HDL and the rest of the programming languages is that the description languages are synthesized, not compiled or executed like any other program.

This is due to the fact that programming languages are defined as procedures. Instead, the hardware description is based on the definition of behaviors according to the inputs and the desired processing concurrently [12]. During the synthesis, the interconnection of the available resources in the FPGA is defined so they behave in the way described. It is part of the work of the development tools to carry out the necessary optimization to take less resources or for the block to operate at higher frequencies. The System Generator automatically translates the block development of a Simulink [13] model into HDL by optimizing FPGA timing and area requirements, and also generates the final binary file [14]. The implementation in hardware was performed on the 3PX1 development kit manufactured by Emtech [15], with a Spartan-6 FPGA (XC6SLX25 [16]). This board meets the basic needs to initiate the development and prototyping of specific system applications with FPGA technology. The board also includes a flash memory where to store the firmware, push-buttons to use as inputs and LEDs to use as status indicators. It was mounted on an open cabinet made of acrylic to give greater rigidity to the board, also to be able to unify it as a single module with BNC connectors for connection and disconnection without compromising the FPGA device. Three connectors were placed on the front panel, two with the PCM Transmitter outputs (data and clock) and a third connector with the input to the acquiring system, Figure 3. The hardware design for the FPGA was performed using MATLAB [17] in conjunction with the System Generator, a tool provided by Xilinx to work in that environment. The MATLAB-Code was used to implement the MATLAB language directly on an FPGA, eliminating the need to program under VHDL or Verilog.

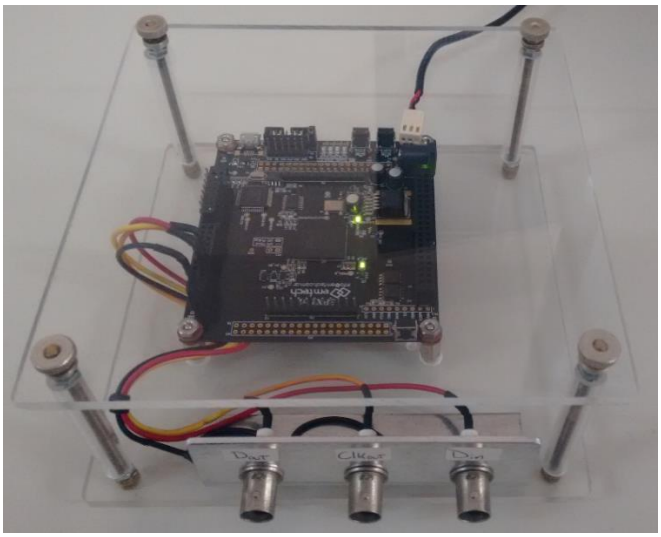

Figure 3. Implementation in the FPGA development kit

3.1. PCM filtering

To generate a realistic scenario, we add Additive White Gaussian noise (AWGN) to an ideal PCM signal. Thermal noise in an ideal resistor is approximately white, meaning that the power spectral density is nearly constant throughout the frequency spectrum (however see the section below on extremely high frequencies). When limited to a finite bandwidth, thermal noise has a nearly Gaussian amplitude distribution [18]. This design supports critical conditions where SNR could exceed -3dB (50% of the signal level). To achieve this, we perform a specific configuration of the "Noise generator" block and perform the noise measurement using the (1).

$$\text{SNR} = \frac{\text{Signal}}{\text{Noise}} = \left| \frac{\text{RMS Value of PCM data}}{\text{RMS Value of Noise generated}} \right| \quad (1)$$

The SNR measurement model can be seen in Figure 4 and the results give a value of approximately 0.6 (-2.2dB), ie the SNR in this case is set to be -2.2dB (60% of the signal level), whereby the condition is completely secured (Figure 5). This mentioned model to perform a SNR measurement has the particularity of doing it in real time, which is advantageous to test different simulation scenarios with different types of frames and times. This PCM filter sub-system implements a CIC filter (Cascaded Integrator-Comb). Implementations of CIC filters have structures that use only adders, subtracters, and delay elements. These structures make CIC filters appealing for their hardware-efficient implementations of multirate filtering [19].

In System Generator, the CIC filter block has a single input port and a single output port, xn and yn, M is the differential delay. In the decimator configuration, the sampling rate is reduced by a factor of R, sub-sampling the output of the last stage of the integrator [20]. As can be seen in Figure 6, the implemented model is a 10 step CIC filter that, despite consuming many resources and a large area of the FPGA, ensures a good filtering that meets the high level of requirement in the design. On the other hand, we must ensure that the width of each bit is kept equal to the output of the filter compared to its original ideal, so we adapt a filter 25:1, which generates 68-bit data at the end of the CIC filter [21]. These last data are compared to a constant to generate a digital pulse at the output of the subsystem. It was determined, by multiple tests and simulations, that the appropriate value of the constant is set to 5.1019 (gives a large value due to successive multiplications per input sample). Figure 7 shows Results of PCM data filtering. First channel: PCM Data w/noise, Second channel: CIC Output, Third channel: Filtering output. Figure 8 shows how the filtering is done correctly in a PCM data signal.

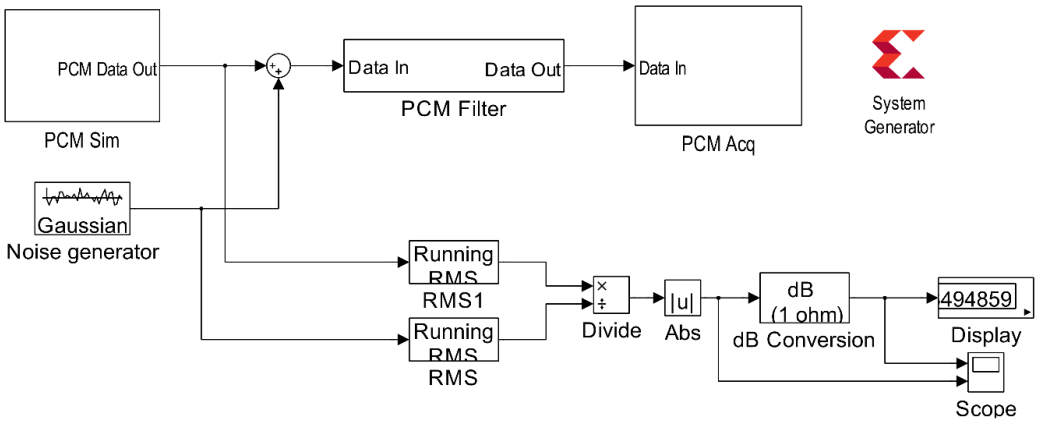

Figure 4. Model to measure SNR

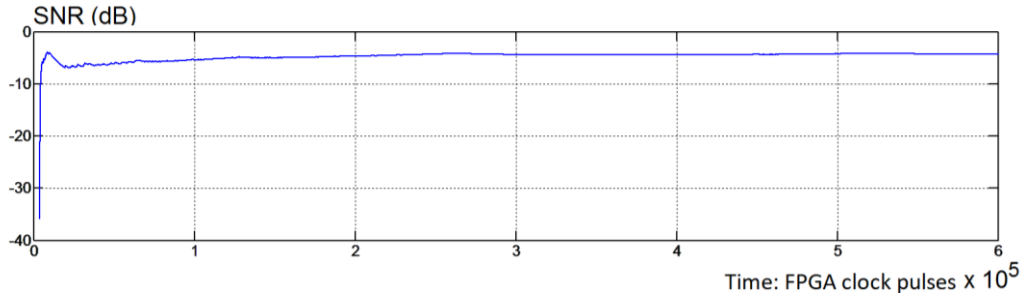

Figure 5. Results of SNR in PCM data with noise added

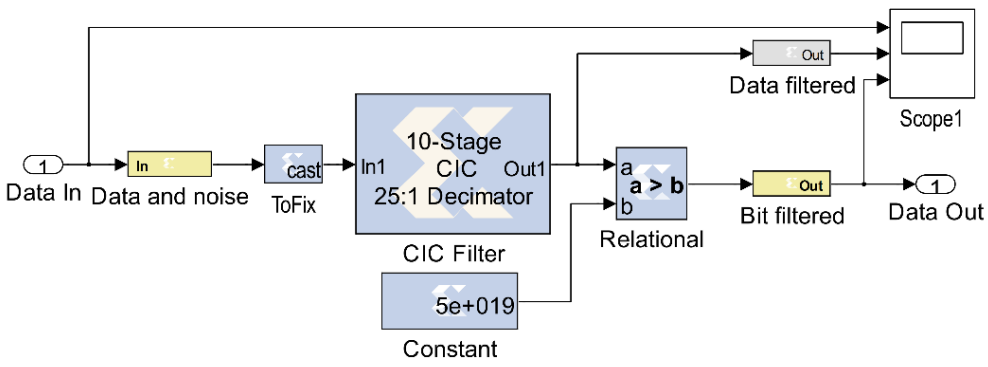

Figure 6. Implementation of PCM data filtering with CIC filter

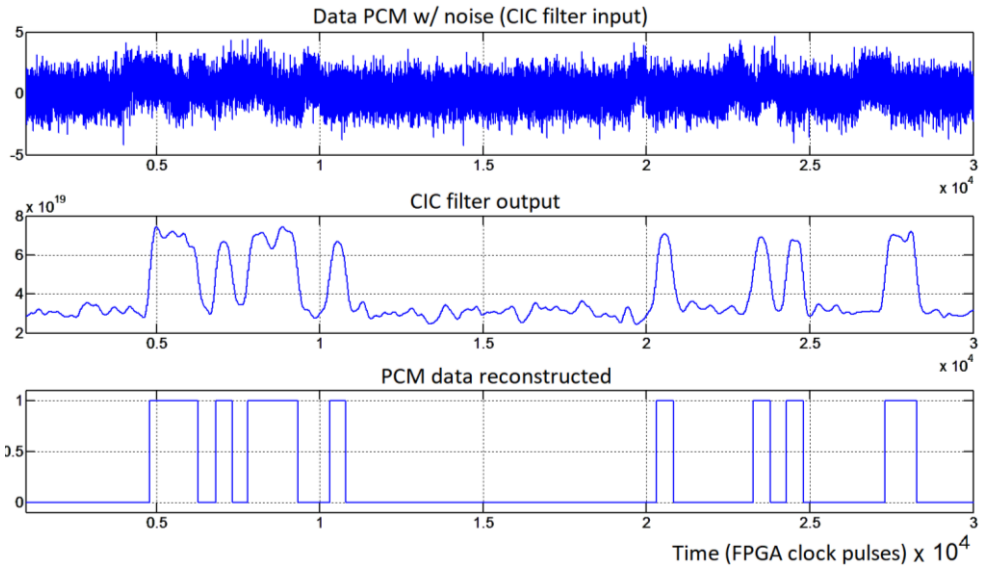

Figure 7. Results of PCM data filtering; first channel: PCM data w/noise, second channel: CIC output, third channel: filtering output

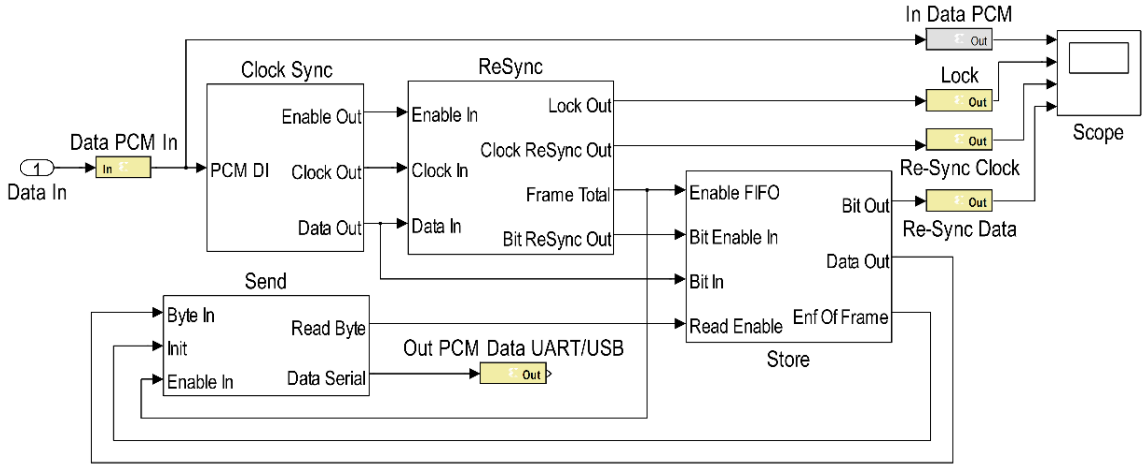

Figure 8. PCM acquisition model

3.2. PCM acquisition

In order to process the PCM incoming data, it is important to first determine the start and end of each packet. Since from the point of view of the entry there is no distinction, the data entry is continuous and asynchronous. The development challenge is to be able to detect the beginning of each package, a pattern recognition or sync word detection, to regenerate the data synchronously for further processing.

3.2.1. Clock-Sync

The algorithm developed for the re-synchronization block is based on the rising edge detection of PCM data input. This sub-system (Figure 9) performs a first PCM clock re-generation continuously and synchronizes automatically when a rising edge is detected in the incoming signal.

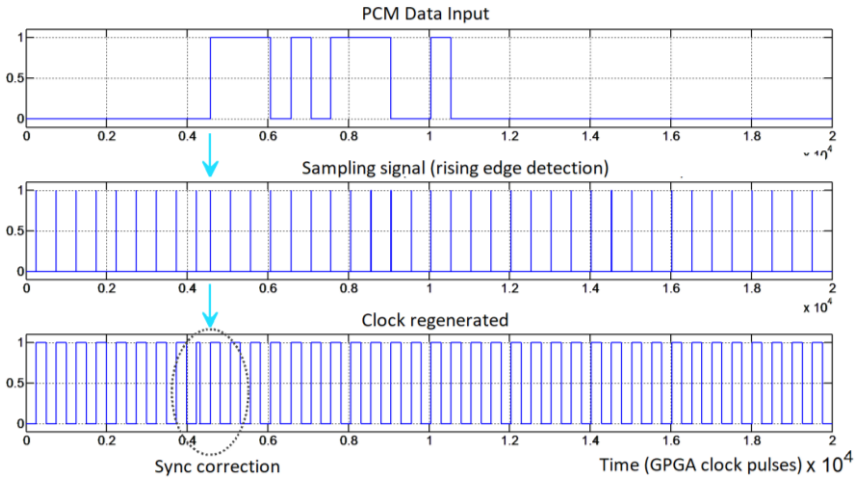

Figure 9. Results of PCM clock-sync; first channel: PCM data input, second channel: rising edge sync, third channel: clock synchronized

3.2.2. ReSync

The ReSync subsystem performs a pattern recognition of the sync-word to lock the incoming signal and re-synchronize it with the clock. To achieve this, two tandem blocks are used. The first one has a finite state machine and a shift register, to find the sync-word bit by bit (performs a bit-sync). The second generates additional signals to other blocks. The results of the simulation can be seen in Figure 10. In the last channel, we can see the incoming PCM data signal in the subsystem, and when the first rising edge is detected, the clock is re-synched, and a valid frame flag is activated, indicating a possible true PCM frame. From that moment, an algorithm starts to recognize the sync-word configurated (performed by a block which is called lock). Figure 10 shows how after 16 bits, the output is true indicating that the sub-system found a valid sync-word. In this case, the sync-word is 1110101110010000 (EB9016). Several frequency measurements were performed with an analyzer. The PCM data (output of the PCM simulator) have 99.9998 kHz with a standard deviation of 1.1248 MHz, with 125 samples per second in a total of 10 seconds of duration. A second frequency stability measurement was performed with the PCM filtering and acquisition in full operation, and the results of the PCM clock regenerated are 9.9998 kHz with a standard deviation of 1.5216 MHz (Figure 11). In conclusion, the error added by the model is practically null.

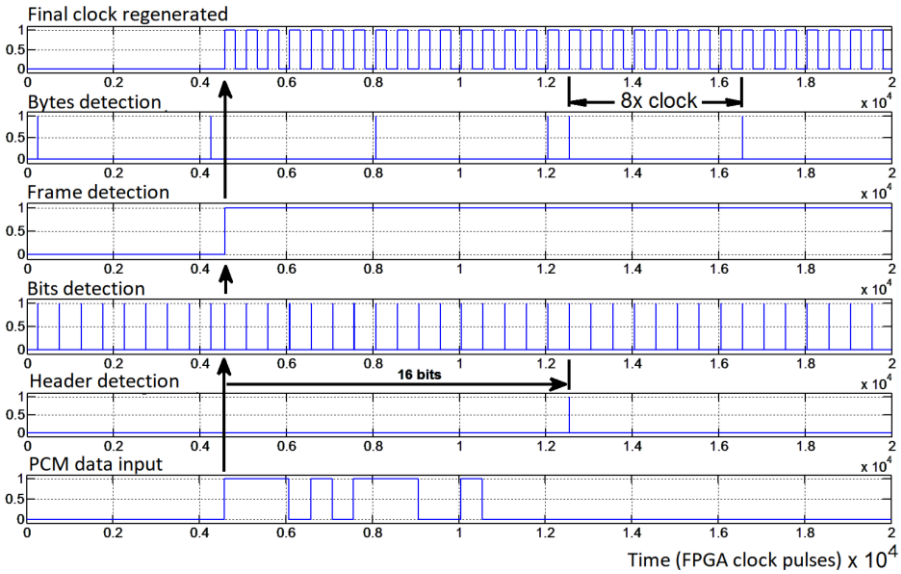

Figure 10. Results of PCM resync; first channel: PCM data clock regenerated, second channel: bytes in PCM frame, third channel: true valid PCM frame, fourth channel: bits in PCM frame, fifth channel: lock signal (sync-word detected), sixth channel: PCM data

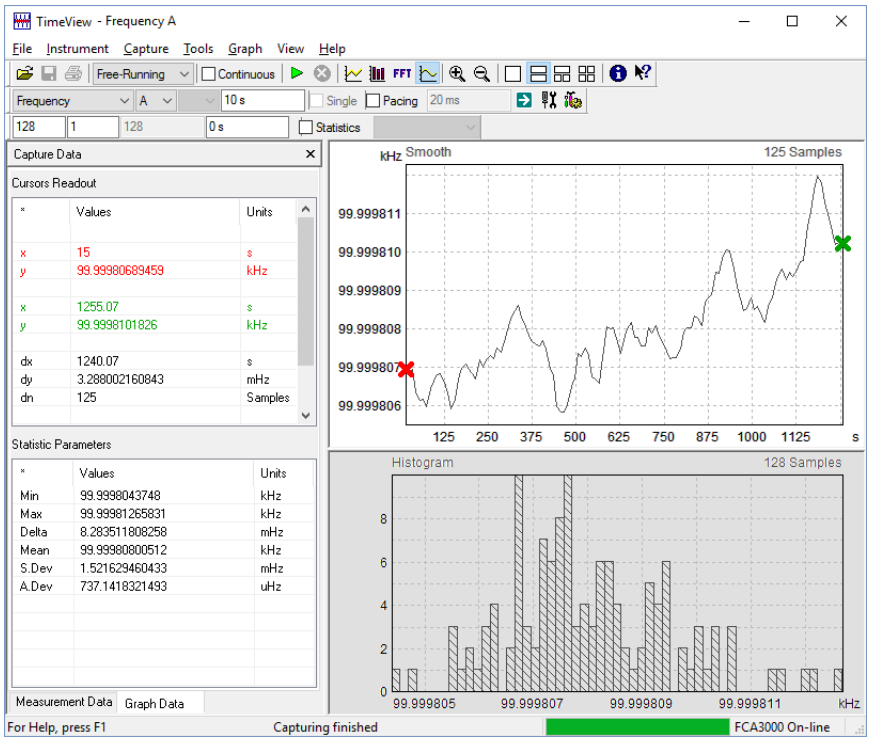

Figure 11. Results of regenerated PCM clock stability

3.2.3. Store

Frame storage in the system is essential for a protocol conversion or pre-processing algorithms. In this case, the acquisition system converts the incoming PCM data and sends it synchronized using the UART protocol, through to USB physical connection [22]. This sub-system combines M-Code block (MATLAB script code) taking the incoming flow of bits and the flag indicating a valid frame, and generates one byte for each channel of the frame. These bytes are stored in a distributed memory FIFO to decrease the resources and area of the FPGA, especially for low-cost devices where hardware resources are more limited without compromising the operation frequency. The operation of the Store sub-system is shown in Figure 12 and it can be observed how the sub-system takes the valid byte at the beginning of the frame, taking into account that the sync word is EB90 in hexadecimals.

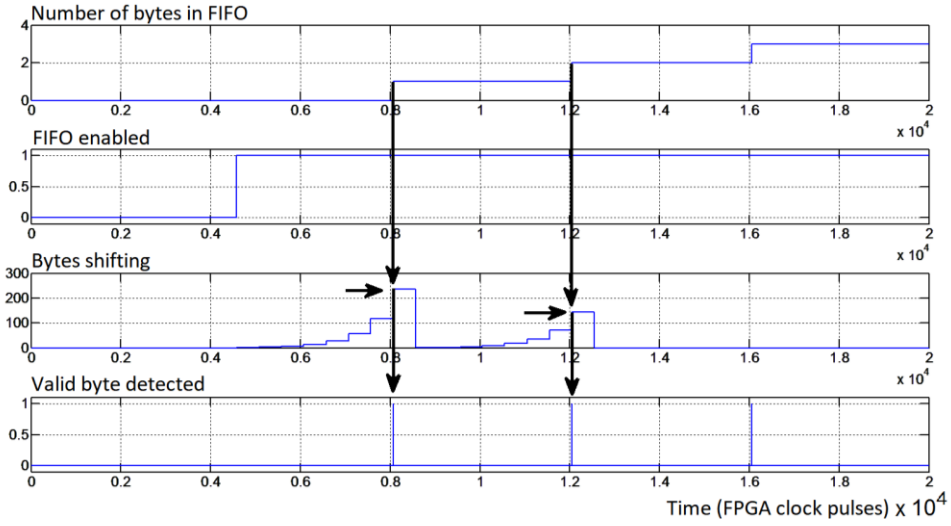

Figure 12. Operation of the store sub-system; first channel: number of bytes in FIFO, second channel: FIFO enable, third channel: shift register, fourth channel: valid byte

3.2.4. Send

The Send subsystem collects the data stored in the Store block and serializes them with an 8N1 UART protocol (i.e. 1 word start bit, 8 data bits and one stop bit). Figure 13 shows a simulation result of the converted PCM frame.

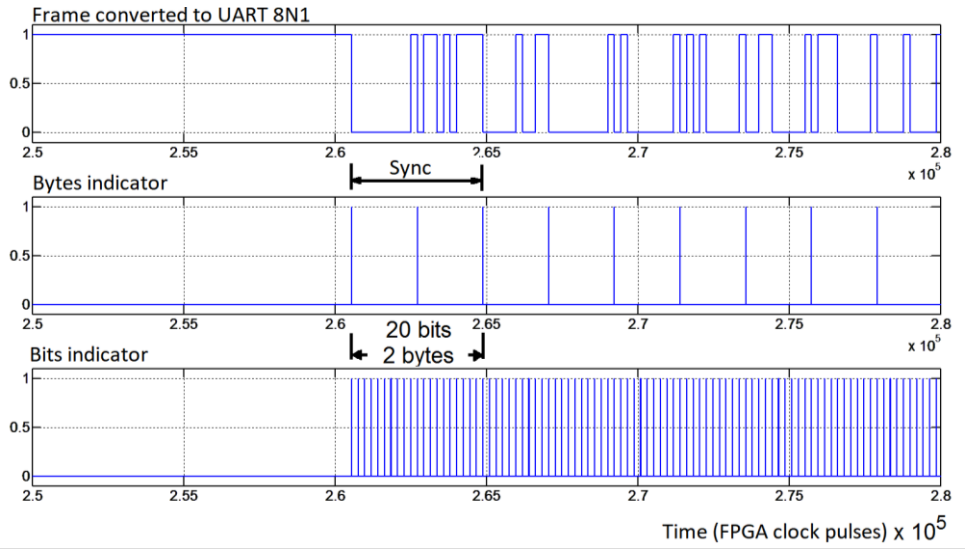

Figure 13. PCM frame converted to UART; first channel: transmit in UART protocol, second channel: flag indicating bytes in UART, third channel: flag indicating bits in UART

4. IMPLEMENTATION AND RESULTS

By connecting the PCM simulator (a known frame used as a pattern) to the acquiring system, we can receive the data captured in a computer through an usb-serial port. The data will be dumped into a vector called serialdata. To perform the proposed analysis, we will convert serialdata into a matrix, where each column represents a complete frame (from the sync word to the end of the frame) and the column number is the received frame number. A sector of the acquired matrix (serialdata) can be seen in Table 1, where it shows the first 28 bytes (from frame 412 to frame 521); in particular, in the byte 7 of the frame 416 an error occurs. Figure 14 shows the matrix in an HSV color chart by treating it like an image. The matrix will be cropped to keep only the payload by deleting the first 4 bytes and the last byte, called serialdatamatrix. A new matrix of constants is generated representing the ideal matrix (as if there were no errors) called datamatrix. The two matrices are subtracted to a new matrix called dif (2). The resulting image is shown in Figure 15.

$$\mathrm{dif}_{(m,n)} = \mathrm{serialdatamatrix}_{(m,n)} - \mathrm{datamatrix}_{(m,n)} \quad (2)$$

In total, 2102 errors were detected from a total of 1375232 bytes captured (21488 PCM frames at 5ms per frame with a length of 64 bytes each). Subtracting the first 4 bytes and the last (5 bytes in total), we have a total of 21488 frames of 59 bytes in payload, resulting in a total of 1267792 bytes. If we calculate the percentage of bytes with error vs. the total payload bytes, it gives an efficiency of 99.83%. The maximum error introduced by the acquiring system is 0.165%, assuming that the error occurs in the complete byte, i.e. the 8 bits are erroneous. This value is due to the fact that the acquiring system is re-engaged by each received frame. Other systems synchronize only once with the first valid sync-word. Therefore, if an error occurs in the medium of the acquisition of a frame, when it detects the sync-word of the next frame, it is re-accommodated. This shows that while the system introduces a small error in the telemetry chain, it is minimal and self-correcting, thus proving trustworthy for mission-critical applications. Figure 16 Shows the errors in PCM acquisition First: Number of errors accumulated in time, second: Residuals.

For the implementation of the PCM acquisition model, the base software and the FPGA device of Table 2 were used. The results of the final implementation, i.e. once the final hardware routing was generated after the synthesis, can be analyzed in Figure 17.

Table 1. A sector of serialdata

	Frame 413	Frame 414	Frame 415	Frame 416	Frame 417	Frame 418
Byte 1	235	235	235	235	235	235
Byte 2	144	144	144	144	144	144
Byte 3	217	217	217	217	9	9
Byte 4	94	95	96	97	215	216
Byte 5	5	5	5	5	5	5
Byte 6	6	6	6	6	6	6
Byte 7	7	7	7	30	7	7
Byte 8	8	8	8	31	8	8
Byte 9	9	9	9	32	9	9
Byte 10	10	10	10	33	10	10
Byte 11	11	11	11	34	11	11
Byte 12	12	12	12	35	12	12
Byte 13	13	13	13	36	13	13
Byte 14	14	14	14	37	14	14
Byte 15	15	15	15	38	15	15
Byte 16	16	16	16	39	16	16

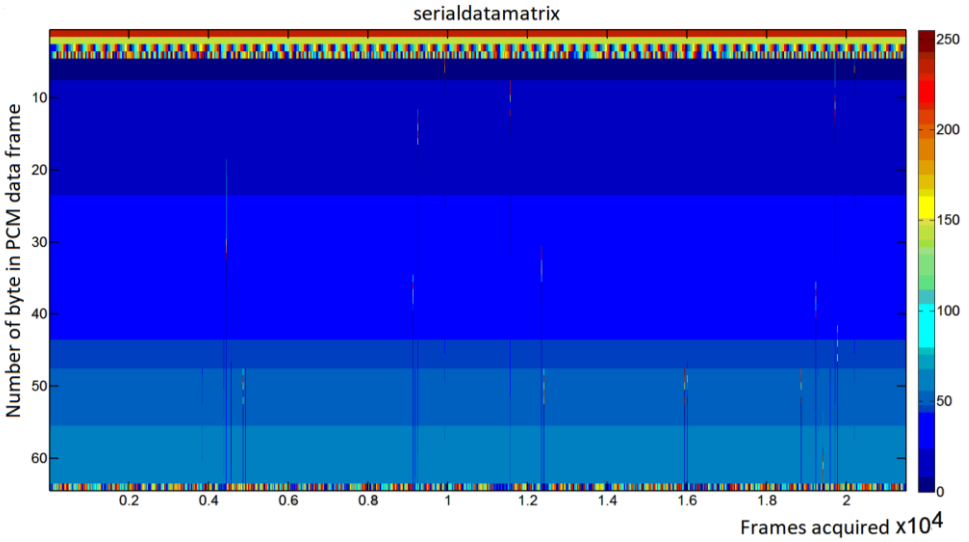

Figure 14. PCM frames in the matrix with HSV codification rows: channel of PCM data (bytes), columns: number of PCM frame received

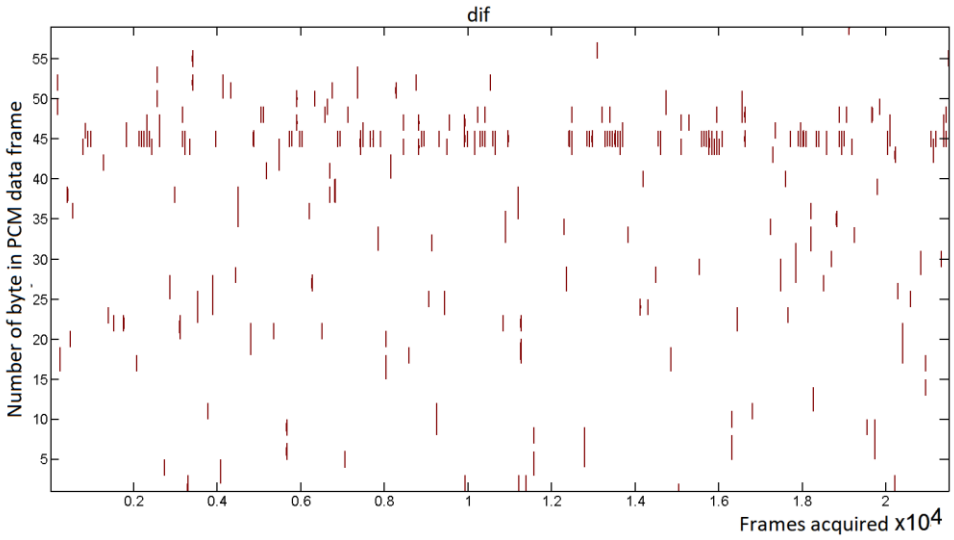

Figure 15. Errors in PCM acquisition rows: channel of PCM data (bytes), columns: number of PCM frame received

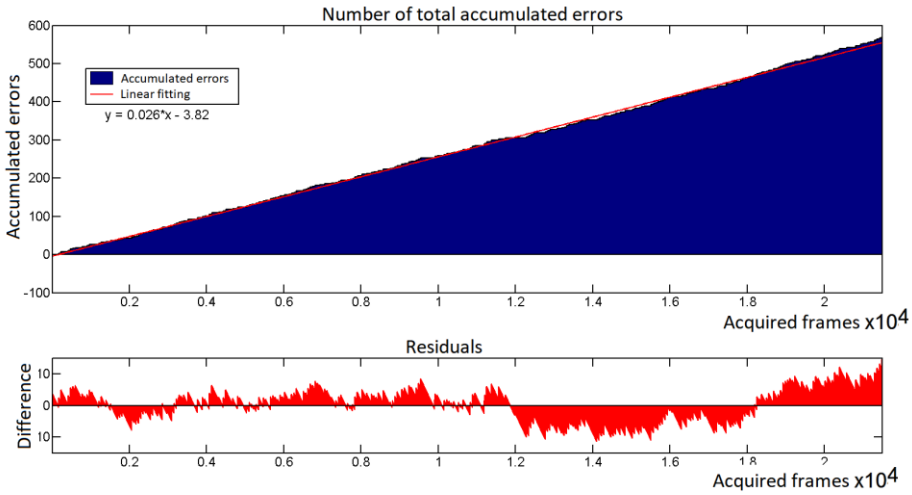

Figure 16. Errors in PCM acquisition first: number of errors accumulated in time, second: residuals

Table 2. Software version and target device for the system implementation

Product Version:	ISE:14.2 - P.28xd	Target Family:	Spartan6
OS Platform:	NT64	Target Device	Xc6slx25
Project ID		Target Package:	Ftg246
Registration ID	__0_0_0	Target Speed:	-3
Date Generated		Tool Flow	C.Line

```
Slice Logic Utilization:
  Number of Slice Registers:                147 out of  30,064    1%
    Number used as Flip Flops:              147
    Number used as Latches:                   0
    Number used as Latch-thrus:               0
    Number used as AND/OR logics:             0
  Number of Slice LUTs:                     183 out of  15,032    1%
    Number used as logic:                   175 out of  15,032    1%
      Number using O6 output only:           80
      Number using O5 output only:           82
      Number using O5 and O6:                13
      Number used as ROM:                     0
    Number used as Memory:                    0 out of   3,664    0%
    Number used exclusively as route-thrus:   8
      Number with same-slice register load:   0
      Number with same-slice carry load:      8
      Number with other load:                 0

Slice Logic Distribution:
  Number of occupied Slices:                 62 out of   3,758    1%
  Nummber of MUXCYs used:                   108 out of   7,516    1%
  Number of LUT Flip Flop pairs used:       204
    Number with an unused Flip Flop:         62 out of     204   30%
    Number with an unused LUT:               21 out of     204   10%
    Number of fully used LUT-FF pairs:      121 out of     204   59%
    Number of slice register sites lost
      to control set restrictions:            0 out of  30,064    0%
```

Figure 17. Place and route implementation report

5. CONCLUSION

In total, 230 frames were found with errors, and a maximum of 5 errors per frame. If an entire frame discard policy is applied in case we found only one error, in total it would be 1.07% of discarded frames. Another important analysis is to corroborate the amount of erroneous data as a function of time by performing a fitting of number of frames received vs. number of errors accumulated in time. The errors fit in a linear equation. In other words, the system is constant-time efficient.

For the implementation of the PCM acquisition model, the base software and the FPGA device were used. The results of the final implementation, i.e. once the final hardware routing was generated after the synthesis. Also, we can highlight that it occupies a very little area of the hardware. Therefore, this system is compatible with low-cost FPGA devices, where the area and resources are very limited.

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. Finally, I am very grateful to the CITEDEF authorities for the logistics support and to MINDEF (Argentine Department of Defense), which provides financial support to this type of programs and projects.

REFERENCES

[1] R.C.C. Telemetry Group, *Pulse code modulation standards*, Telemetry Standards - IRIG standard 106-13 - part 1 - chapter 4, Telemetry Group, 2013.
[2] M. Glass, "Irig 106 chapter 10 standardizes mil-std-1553b data recording," in *2007 IEEE/AIAA 26th Digital Avionics Systems Conference*, pp. 2.B.5–1–2.B.5–9, Oct. 2007.
[3] A. Stacul *et al*., "Diseo, desarrollo e implementacin de una estacin terrena para cohetes sonda," in *VI Congreso de Microelectrnica Aplicada, Buenos Aires*, Argentina, 2015.
[4] M. Frerking, *Digital Signal Processing in Communication Systems*, Prentice Hall, 1994.
[5] Consultative Committee for Space Data Systems, "Radio frequency and modulation systems part 1: Earth stations and spacecraft," 2000. [Online]. Available: https://public.ccsds.org/Pubs/401x0b25.pdf.
[6] A. E. Moghazy, G. Maral, and A. Blanchard, "Digital pcm bit synchronizer and detector," *IEEE Transactions on Communications*, vol. 28, no. 8, pp. 1197–1204, Aug. 1980.
[7] G. Sadhukhan, M. Sandhu, and R. P. Singh, "Vlsi based implementation of pcm mux encoder for range telemetry system," in *Proceedings of the IEEE INDICON 2004. First India Annual Conference 2004*, pp. 23–26, Dec. 2004.
[8] S. Sanmin and L. Wenyi, "A pulse code modulation decoding method by self-synchronizing with vhdl," in *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, vol. 13, pp. V13–412–V13–415, Oct. 2010.
[9] P. Zhang and R. Yang, "Pcm self-test module based on rs422 interface," in *Proceedings of 2011 International Conference on Electronics and Optoelectronics*, vol. 2, pp. V2–292–V2–294, Jul. 2011.
[10] M. A. Rafique, *et al.*, "A cost and resource efficient telemetry host station design using fpga," in *2018 15th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pp. 799–804, Jan. 2018.
[11] Xilinx Inc., "System generator for dsp user guide," 2016. [Online]. Available: http://www.xilinx.com/support/documentation/sw manuals/xilinx11/sysgen user.pdf .
[12] P. Chu, *FPGA prototyping by Verilog examples*, Wiley, 2008.
[13] MathWorks Inc., "Simulink," 2016. [Online]. Available: http://www.mathworks.com/products/simulink/.
[14] M.R. Valido *et al*., "Design methodology in FPGA using Xilinx System generator (in Spanish)," pp. 282-286, 2012.
[15] Emtech S.A., "Development kit 3px1," [Online]. Available: http://www.emtech.com.ar/producto/placa3px1.
[16] Xilinx, "Spartan-6 family overview," [Online]. Available: https://www.xilinx.com/support/documentation/data sheets/ds160.pdf.
[17] MathWorks Inc., "Matlab-the lenguaje of technical computing," 2016. [Online]. Available: https:// www.mathworks.com/products/matlab.html.
[18] J. Barry, E. Lee, and D. Messerschmitt, *Digital Communications*, Sprinter, pp. 69, 2004.
[19] R. Crochiere and L. Rabiner, *Multirate Digital Signal Processing*. Prentice Hall, 1983.
[20] E. Hogenauer, "An ecnomical class of digital filters for decimation and interpolation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 2, Apr. 1981.
[21] A. Oppenheim and R. Schafer, *Discrete-Time Signal Processing*, Prentice Hall, 1989.
[22] Electronic Industries Association, Engineering Department, "Interface between data terminal equipment and data communication equipment employing serial binary data interchange," Electronic Industries Association, Engineering Dept., 1969.
[23] N. Marda, G. Vaishnav, and U. S. N. Rao, "Bit error rate testing scheme for digital communication devices," in *Proceedings of The 2014 International Conference on Control, Instrumentation, Energy and Communication (CIEC)*, pp. 490–493, Jan. 2014.

BIOGRAPHIES OF AUTHORS

Adrián Stacul is a graduate in Electronics Engineering from the National Technological University (Universidad Tecnológica Nacional - UTN). His R&D main interest focuses on on-board hardware, flight centres applied to unmanned vehicles and ground stations for data processing. The Eng. Adrian Stacul is beginning in the scientific publications in these areas. He currently with the CITEDEF as a developer and researcher in topics of signal processing and projects co-director in CITEDEF with Defense Department (Ministerio de Defensa - MINDEF) of Argentina. He is Professor of Computers Architecture with de Systems Engineering Department at UTN and there he develops his Ph.D. thesis with mention to signal processing and images in the thematic of high-speed data acquisition.