Development of a Coincidence Sorter for Dedicated PET

Eleftheria Kostara[1], Giancarlo Sportelli[1,2], Maria-Giuseppina Bisogni[1,2], Fabrizio Palla[1]
FIIG – Functional Imaging and Instrumentation Group
[1]INFN, Sezione di Pisa, Largo B. Pontecorvo, 3, I-56127 Pisa, Italy
[2]Department of Physics, University of Pisa, Largo B. Pontecorvo 3, I-56127 Pisa, Italy
Email: kostara;giuseppina.bisogni@pi.infn.it

***Abstract*—Positron emission tomography is an imaging technique. PET is used in nuclear medicine and is aiming to improve monitoring of in-vivo treatment. A sorter architecture is developed and is going to be implemented in a Cyclone V FPGA in the data acquisition board of an in-beam PET system. The sorter is going to provide real time coincidence detection, aiming to discard non-accurate events. This is performed by merging and sorting incoming data packets by timestamp. A testbench is developed to verify the functionality and the performance of the event sorter. The sorter functionality is validated by comparing the results of the testbench and the sorted data that was produced by a python script. The requirement of 20MHz minimum throughput rate is confirmed.**

Keywords—Positron Emission Tomography (PET); coincidence sorter;

I. INTRODUCTION

Positron emission tomography (PET) is a functional imaging technique in nuclear medicine, that produces a three dimensional image of functional processes in the body. In PET, a positron emitting radioactive tracer is injected into the patient and spreads within the body, generating a number of positrons through the β^+ decays. The emitted positrons annihilate with electrons of the surrounding tissues producing two back-to-back 511keV photons. The emitted radiation from the body is registered in external detectors. The tomographic images of the tracer distribution in the body are reconstructed using mathematical algorithms [1][2].

Each coincidence event, as described in [3], represents a line in space connecting the two detectors, the line of response (LOR), along which the positron emission occurred. The coincidence events in PET fall into four categories. First, true coincidences occur when both photons do not have any interaction prior to their detection and no other event is detected within a predefined time interval, referred as coincidence window (CW). Second, in a scattered coincidence at least one of the detected photons has undergone at least one Compton scattering prior to its detection. Third, random coincidences occur when two photons, which are not arising from the same annihilation event, are localized by the detectors within the same CW. Finally, multiple coincidences occur when more than two photons are detected in different detectors within the CW. All of the detected coincidence events are recorded by the PET computer system as a raw data set. For the efficient image reconstruction, it is useful to discard the scattered and random coincidences and keep only the true.

The INSIDE (INnovative Solutions for In-beam DosimEtry in hadron therapy) project aims at the development of instruments and techniques for the monitoring of the dose delivered to a patient during hadron therapy treatment. The hadron therapy is based on the irradiation using protons and atomic nuclei (ions), called hadrons that are subjected to a strong nuclear force. The main goal of this project is to build a dedicated PET scanner able to give in-beam quality control of the proton and Carbon-12 therapy treatment, by using state-of-the-art detectors and digital electronics technology [4][5]. The sorter that was developed is going to be implemented in the motherboard, which is part of the data acquisition (DAQ) system. To keep only the accurate events and discard the others, it is necessary to merge and sort the acquired data by timestamp. This is the main function of the sorter in order to keep only the pairs of consecutive events that lie within the appropriate CW.

The INSIDE project follows the works started with the dedicated in-beam DoPET system (Dosimetry with a Positron Emission Tomograph) [6][7]. Its purpose was to demonstrate the feasibility of a technique for the proton therapy monitoring, immediately after the patient irradiation. With respect to DoPET, INSIDE aims to widen the detectors area and further reduce the dead time, so as to enable beam-on PET imaging [8][9][10] and construct a version for clinical use.

II. PET SET UP

The PET prototype designed by the INSIDE project, is made of two planar panels 10cm x 20cm wide. Each panel consists of 2 x 4 pixelated LFS (Lutetium Fine Silicate) scintillator matrices measure 5cm x 5cm (16 x 16 pixels, 3mm x 3mm x 20mm crystals). Each scintillator matrix is coupled to four matrices of 8 x 8 MPPC (Multi Pixel Photon Counter) from Hamamatsu with one-to-one crystal-detector coupling to form a detection module. Each module uses four 64-channel front-end (FE) ASICs, with on-chip multiplexing and time-to-digital (TDC) conversion resources based on the same 0.18um CMOS technology.

TABLE I. DATA PACKET

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W0	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W1	Detector ID						Crystal ID									
W2	E3	E2	E1	E0	Energy											
W3	Timestamp MSB															
W4	Timestamp LSB															
W5	Energy ID								Timestamp Fraction							

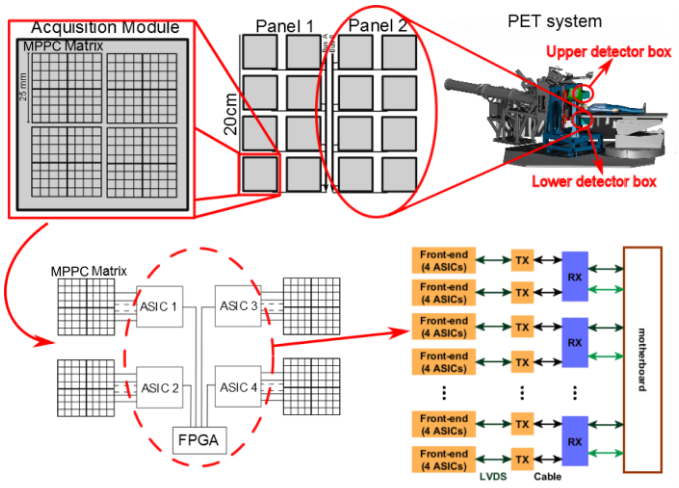

Fig. 1. Acquisition system on PET.

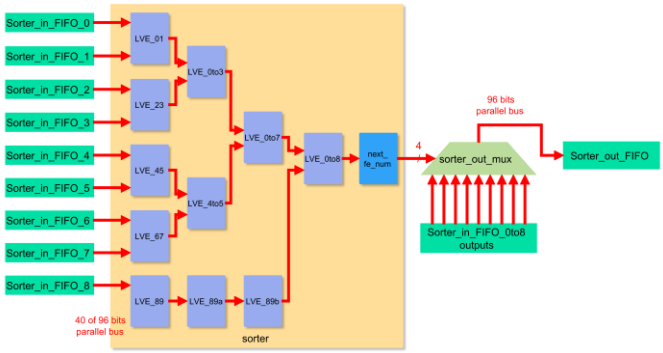

Fig. 2. Architecture of the event sorter.

TABLE II. PROPOSED FORM OF THE SORTER'S INPUT BITS

1 bit	4 bits	40 bits
Valid Data	FE number	Timestamp

A small low-cost FPGA board, the TX-board (transmitter), is mounted along with each FE-board in a separate enclosure to the PET block enclosure. The DAQ motherboard has nine slots. In each slot, the RX-board (receiver) is connected and acts as a data concentrator for the fiber-optic transceivers of the TX-boards and for communication between the DAQ system and the FE-boards. In each RX-board, two TX-boards are connected, while each TX-board is connected to the FE-boards (4 ASICs). The motherboard can manage up to nine RX-boards (Figure 1). The main data stream coming from the FE-boards contains encoded information about the detected photons, such as: the detector impact position, the deposited energy and the detection timestamp. The above data is stored in a data packet of 6 words of 16 bits each, as it is shown in the Table 1.

III. COINCIDENCE SORTER ARCHITECTURE

For the coincidence detection, the timestamp of the events must differ less than a CW of the order of 1ns. In order to detect coincidences within the data stream, it is necessary to merge and sort them by timestamp. Then, it is possible to find sets of events that fall within a CW by comparing each event with the following ones [11].

The proposed solution for the sorter is a pipelined architecture. The sorter compares the timestamps of the events at the front positions of each FIFO (First-In First-Out) and outputs the earliest event, which is stored to an output FIFO. (Figure 2).

The sorter is composed of pipelined LVEs. Each LVE (Lowest Value Extractor) compares the timestamps of two events coming from two FIFOs or two previous LVEs and outputs the earliest one. If the LVE has only one input or one of the FIFOs is empty, then it outputs the only available event. If all the FIFOs are empty, then the process never starts. Four stages are needed for nine FEs.

It is important to note that the data that is streamed along the sorter does not include all the event acquired info. This is done to save memory. For the sorting process only the timestamp, along with five additional bits, is used (Table 2). In the latest stage a multiplexer is used to recover all the data from the input FIFOs. The first bit shows if the corresponding input FIFO has valid data or not, checking the empty flag of the FIFO. The following four bits contain the number of the FE from which the event is coming and the last 40 bits are the timestamp.

In Figure 3 the finite state machine of the LVE is shown. Its states are presented below:

- INIT: performs initialization right after the FPGA is reset
- RESET: resets the signals
- READY: checks if there is valid data in one of the two inputs of the LVE
- READ: checks which input is the lowest and outputs the corresponding data

- WAIT: checks if there is new data to the input in order to start the process again

The operation of the LVE, and in consequence the sorter, is described in detail below. Each LVE checks the first bit. If there is valid data in the FIFOs, then the first bit is asserted and each FIFO outputs the first timestamp, along with the additional bits, to the LVEs. Thereafter, the process starts and each LVE compares the timestamps and outputs the lowest timestamp along with the five additional bits into its output. If there is no valid data in one of the FIFOs, meaning that this FIFO is empty, then its first bit is not-asserted and the LVE outputs the available timestamp. In this way the case when there is only one input with data in the LVE is also covered. The first bit of the input without data is initialized to be not-asserted. Finally, if there is no valid data in any FIFOs, their first bit is not-asserted and the process never starts.

In the final stage, the "next_fe_num" component checks the four bits (FE number), that indicate the corresponding input FIFO which presents the earliest timestamp. Then, it outputs the next timestamp from that FIFO and the process starts again. Also, these four bits are used for reading the corresponding data packet, which is stored to the output FIFO through the multiplexer.

IV. Testbench Implementation

To verify the functionality and the efficiency of the sorter a testbench is required. For this purpose, the method illustrated in Figure 4 was developed.

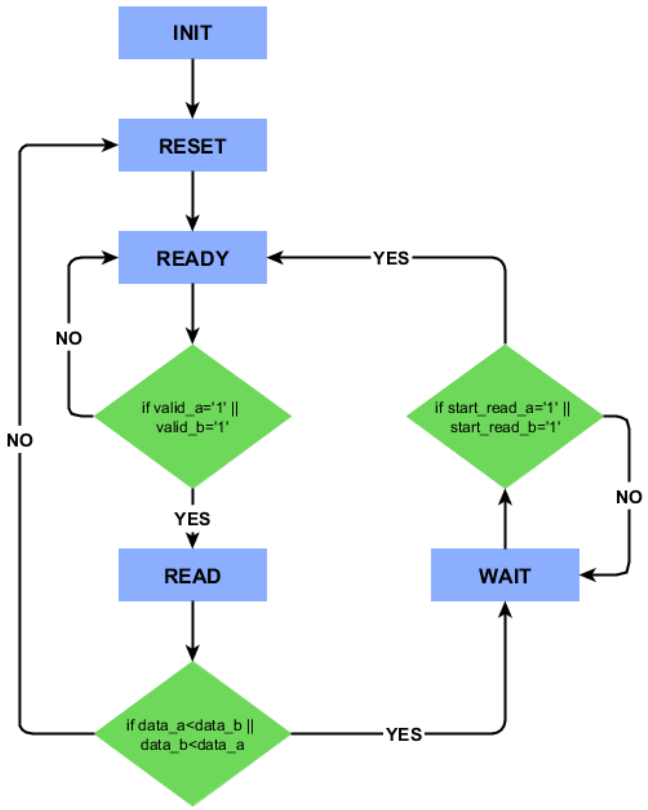

Fig. 3. FSM of the LVE.

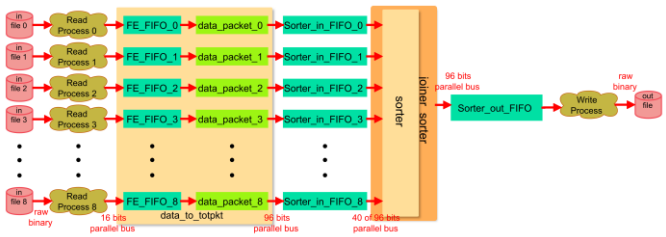

Fig. 4. Block diagram of the testbench.

A python script is used to create nine data files with a hundred random events each, as well as a file with the total amount of this data, but sorted. The data is converted, through a process, from binary form to a VHDL (Hardware Description Language) vector. Then, the converted data is stored in FIFOs (6 x 100 words of 16 bits). Following, the 6 words of a data packet are merged to one word and these new words (100 words of 96 bits) are stored to additional FIFOs. Thus, the part of the words, which contains the timestamp, is going through the sorter. In the end, the words, sorted by timestamp, are stored to an output FIFO and then, are converted and written to an output file on binary format.

Note that each input FIFO ("FE_FIFO" in the diagram) outputs 16 bits at a time corresponding to one word. For the timestamp, two and a half words (40 bits) are necessary, according to the Table 1. Thus, an additional FIFO ("Sorter_in_FIFO" in the diagram) is used to store each data packet to one word of 96 bits. The sorter uses only the 40 bits of the timestamp for the process, but in the output FIFO the whole data packet is stored.

According to the above information, each "FE_FIFO", used in the testbench implementation, must have depth at least six times the number of the events and width 16 bits. As well, each "Sorter_in_FIFO" must have depth at least the number of the events and width 96 bits. Since the FPGA FIFOs are optimized in sizes equal to 2^k, the first FIFO has dimensions: D x W = 1024 x 16 and the second one: D x W = 128 x 96.

V. Results

The sorter must fit within a Cyclone V. The specific FPGA which is used, includes 149.5K logic elements (LEs), 6.86 Mb of embedded memory arranged as 10 Kb (M10K) blocks, 836 Kb memory logic array blocks (MLABs), 156 variable-precision digital signal processing (DSP) blocks, 7 fractional phase-locked loops (PLLs), 896 pins.

Table 3 shows the usage of the FPGA resources for the sorter. The sorter must have minimum throughput rate of 20MHz for a system clock of 200MHz. A maximum frequency of 252MHz for the system clock can be achieved by setting up properly the timing constraints using the multicycle path command for the system clock.

The testbench was implemented and tested using ModelSim. The generated data file from the simulation is equal to the one generated from the python script. Thus, the functionality of the sorter is verified. At least 10 clock cycles are necessary for the requirement of the minimum throughput rate of 20MHz for a clock of 200MHz. The proposed sorter

architecture needs seven clock cycles to output the earliest timestamp, which leads to a throughput rate of 28MHz.

TABLE III. FPGA USAGE

Resource	Used	Percentage
Logic utilization (in ALMs)	2,395	4%
Total registers	4,178	-
Total block memory bits	408,576	6%

VI. CONCLUSIONS

In this work a coincidence sorter is developed in order to provide real time coincidence detection. The coincidence events that are detected after the emitted radiation from the patient's body are recorded as a raw data set. This data is merged and sorted by timestamp in the event sorter, in order to be compared and the non-true coincidences to be rejected. The sorter is going to be implemented in the motherboard's FPGA, which belongs to the DAQ system of an in-beam PET. The functionality of the sorter is verified by developing a testbench and its simulation on ModelSim. The results coming from the simulation are equal with these from the python script and the requirement of 20MHz minimum throughput rate is confirmed.

The next step is to develop and implement a filter that will discard the non-true coincidences. The filter along with the sorter will be part of the coincidence network for the DAQ system. Also, since the sorter is tested in simulation, an important step is to test its functionality on a real FPGA device.

ACKNOWLEDGMENT

This work has received funding from the European Union's Seventh Framework Program for research, technological development and demonstration under grant agreement n^{o} 317446. The INSIDE project is funded by the MIUR (Ministero dell'Istuzione, dell'Università e della Richerca) of the Italian government under the program PRIN 2010-2011 project nr. 2010P98A75.

REFERENCES

[1] A. Del Guerra, "Ionizing radiation detectors for medical imaging", Singapore: World Scientific, 2004.

[2] P. Oehr, H-J. Biersack, and R.E. Coleman, "PET and PET-CT in oncology", Springer Science & Business Media, 2003.

[3] R. Badawi (1999), "Introduction to PET Physics", Retrieved from http://depts.washington.edu/nucmed/IRL/pet_intro/toc.html.

[4] M. Marafini, et al., "The INSIDE Project: Innovative Solution for In-beam DosimEtry in Handron therapy", Submitted to ACTA Physics Polonica Proceedings of II Symposium on Positron Emission tomography, Jagiellonian University of Krakov, 21-24 September 2014.

[5] F. Pennazio, et al., "A study of monitoring perfomances with the INSIDE project", Submitted to ACTA Physics Polonica Proceedings of II Symposium on Positron Emission tomography, Jagiellonian University of Krakov, 21-24 September 2014.

[6] S. Vecchio, et al. "A PET prototype for "in-beam" monitoring of proton therapy." Nuclear Science, IEEE Transactions on 56.1 (2009): 51-56.

[7] G. Sportelli, et al. "Reprogrammable acquisition architecture for dedicated positron emission tomography." Nuclear Science, IEEE Transactions on 58.3 (2011): 695-702.

[8] G. Sportelli, et al. "First full-beam PET acquisitions in proton therapy with a modular dual-head dedicated system." Physics in medicine and biology 59.1 (2014): 43.

[9] V. Rosso, et al. "A new PET prototype for proton therapy: comparison of data and Monte Carlo simulations." Journal of Instrumentation 8.03 (2013): C03021.

[10] A.C. Kraan, et al. "Proton range monitoring with in-beam PET: Monte Carlo activity predictions and comparison with cyclotron data." Physica Medica 30.5 (2014): 559-569.

[11] M-A. Tetrault, et al. "Real time coincidence detection engine for high count rate timestamp based PET." Nuclear Science, IEEE Transactions on 57.1 (2010): 117-124.