Low-resource synchronous coincidence processor for positron emission tomography

Giancarlo Sportelli [a,b,*], Nicola Belcari [c], Pedro Guerra [b], Andrés Santos [a,b]

[a] Biomedical Image Technologies, E.T.S.I.T., Universidad Politécnica de Madrid, 28040 Madrid, Spain
[b] Research Center in Bioengineering, Biomaterials and Nanomedicine, 50018 Zaragoza, Spain
[c] Department of Physics "E. Fermi", Università di Pisa, 56127 Pisa, Italy

ARTICLE INFO

ABSTRACT

We developed a new FPGA-based method for coincidence detection in positron emission tomography. The method requires low device resources and no specific peripherals in order to resolve coincident digital pulses within a time window of a few nanoseconds. This method has been validated with a low-end Xilinx Spartan-3E and provided coincidence resolutions lower than 6 ns. This resolution depends directly on the signal propagation properties of the target device and the maximum available clock frequency, therefore it is expected to improve considerably on higher-end FPGAs.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

Coincidence detection is likely to be the most sophisticated stage of a PET acquisition system and the one to which is dedicated the most expensive and cutting edge hardware. This is particularly true when the number of detectors increases, being the tight timing constraints harder to met.

Current coincidence processors are mainly based on two different approaches: AND-gating and Time-to-Digital conversion (TDC). In the former case, for each photon, a digital pulse of width W, typically of the order of a few nanoseconds, is generated and combined with the pulses coming from other detectors. The combinatorial circuit generates a coincidence trigger whenever two pulses overlap, resulting in a coincidence resolution of $2W$. AND-gating is a simple and cheap technique, as long as the number of detectors is low and the combinatorial network is small.

The latter approach consists in labelling each photon with a finely calculated timestamp, synchronous with a common distributed clock. Coincidences are thus resolved by computing differences in real time or off line. TDC approaches are more scalable but expensive [1–4]. Although TDC processing is by far superior to AND-gating counterpart, in terms of processing flexibility and timing resolution, it requires high-speed front-end electronics, able to acquire and transmit every single photon, including odd ones [5,6]. Bandwidth and memory requirements are thus dramatically increased as well as the system cost.

In the last decade, FPGAs have been successfully adopted in both AND-gating and TDC processing approaches [7–9]. This provided higher levels of flexibility, and the possibility of implementing fully functional systems on a single chip. Such an approach simplifies sensibly the design and reduces costs. However, the finite resources and pin-out of the device may limit the maximum achievable number of controlled detectors.

We propose a new AND-gating, FPGA-based method, able to achieve state of the art coincidence resolutions, with improved scalability and reduced resources usage. The new method consists in clocking a small region of the FPGA at the maximum speed available, and synchronizing incoming pulses within such an accelerated synchronous region. Coincidence detection can then be achieved by gating the narrow synchronised pulses, provided that special measures are taken in order to prevent synchronization failures and logic hazards. The synchronous nature of the processor greatly simplify the network gating implementation, thus allowing for complex and wide combinatorial possibilities.

2. Overall architecture

2.1. Operation concept

The coincidence processor operates at the maximum clock frequency available in the target FPGA, which ranges from 300 MHz in low-end devices to about 800 MHz in high-end ones. Achieving the maximum frequency requires proper resources usage and pipelining, but it has been shown to be possible if the clock domain is confined and separated from the acquisition custom logic (Fig. 1).

Multiple clock domain are possible with the use of embedded digital clock managers, and cross-domain synchronization chains, that prevent the propagation of flip-flop metastable states. It must be noted that the two synchronization batteries, at the coincidence processor input and at the system input, are the main responsible for coincidence processing latency. The latency introduced by the synchronization stage is $k\tau$, where k is the number of chained flip-flops and τ the clock period. The time $k\tau$ must be generally higher than the metastability *resolving time R*, which depends on the electrical characteristics of the target device [10].
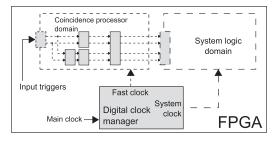
Within the synchronous domain, a coincidence is detected when two signals rise during the same clock period. This is accomplished by shaping the incoming signals to one-cycle pulses and feed both to a synchronous AND-gating network, as illustrated in Fig. 2. In this way all events separated by a delay bigger than the clock period τ are discarded. However, of all the events closer than τ, those that fall at the two sides of a clock rising edge will be also lost. We refer to this undesirable condition as *hazard*. One way to recover hazards could be to use two-cycles pulses. In this way, all events closer than 3τ are guaranteed to be resolved as coincidences.

2.2. The gating network

Once all the inputs have become synchronous, coincidence gating is a relatively simple task. The combinatorial function is specific for a given PET geometry. We implemented a modularized dual planar PET geometry, in which each detector is made of n modules. With this geometry each module A_i of one side can receive a photon in coincidence with a module B_j of the other side. The gating function can then be expressed with the boolean expression:

$$C_{A,i} = A_i \cdot \sum_{j}^{n} B_j, \quad C_{B,j} = B_j \cdot \sum_{i}^{n} A_i$$

where $C_{A,i}$ and $C_{B,j}$ are the coincidence outputs. At high clock frequencies and with high detector numbers it might be required to divide the function in pipelined stages, in order not to incur in timing violations. Pipelining has the negative effect of increasing resources usage and detection latency by τ per stage.

Random events counting has been achieved with the delayed window technique. Delays have been realized using a series of shift registers. The gating function implements a new variant of the standard delayed window technique, in which only prompt events are triggered for acquisition [11]. The adopted technique has the advantage of eliminating the detection latency due to the window delay.
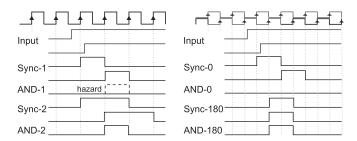
2.3. Dual phase hazard recovery

In order to reduce the achievable timing resolution for a given clock frequency, we used an alternative dual phase approach. The idea is to use one-cycle pulses within two identical instances of the circuitry in Fig. 2, each clocked at the same frequency but opposite phases. The outputs are then re-synchronized separately with the system clock and eventually OR-gated. Doing so, all coincident pairs that fall across the clocking rising edge on one instance, must fall completely within the clock period of the other instance (Fig. 3). This allows to recover the above mentioned hazards within a time window of $3\tau/2$, i.e. to halve the minimum coincidence resolution within a given target device. This approach could in principle be extended to more clock phases.
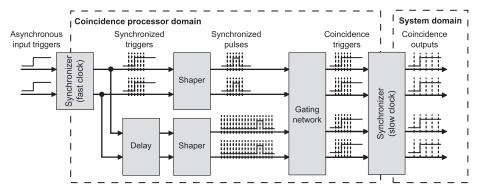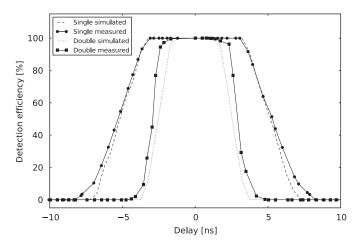
3. Results

3.1. Coincidence resolution

The proposed technique has been implemented and simulated for a low-end Spartan3E-1200 (Xilinx, San Jose, USA) target FPGA. We have also performed test syntheses runs for the newer, still

Fig. 1. Overall architecture of a FPGA acquisition system with an embedded synchronous coincidence processor. The digital clock manager is a standard FPGA component.

Fig. 3. Timing diagram of a hazard occurrence. Sync-1 and Sync-2 represent the signals for one-cycle and two-cycle shaping cases, respectively, for the single clocking version. Sync-0 and Sync-180 represent the signals in the two processor replicas, fed with opposite clocks.

Fig. 2. Detailed block diagram of the synchronous coincidence processor. Input pulses are synchronized with the boosted clock and combined according to a configurable AND-gating network. Coincidence outputs are then re-synchronized with the slower system clock.

Fig. 4. Simulated and measured coincidence windows for the two implemented processors. The single clock version results in a coincidence resolution of 10.4 ns FWHM, while the dual phase version results in 5.3 ns.

Fig. 5. FPGA resources utilization against the total number of detector channels for a dual planar detector assembly, with the single clock processing configuration.

Table 1
Pipeline stages required to perform synchronous AND-gating of two detector subsets on the Xilinx Spartan-3E (S3E) and Spartan-6 (S6) devices at a working clock frequency of 288 MHz.

Detectors	2	4	8	18	32	64	80	96
Stages								
S3E	0	0	0	0	1	2	3	4
S6	0	0	0	0	0	2	3	3

low-end, Xilinx Spartan6-16 target device, in order to compare resource usage and timing constraints.

Timing resolution has been studied with a reference clock frequency $\tau = 288$ MHz for both single and dual clock versions. Simulations have been carried out with Modelsim 6.5a (Mentor Graphics, San Jose, USA) by feeding the processor with two triggers separated by a variable delay, spanning 20 ns in steps of 20 ps with an additive Gaussian jitter ($\sigma = 0.5$ ns, $\mu = 0$). Simulations have been carried out on the post-synthesis simulation model generated by the Xilinx ISE 10.1i.

The measurement platform was based on a XEM3005 (Opal Kelly, Portland, USA) fast prototyping board. In order to generate two triggers with a variable delay, we used a non-compensated ring oscillator and a series of delay components within the FPGA fabric. In this way we were able to produce delays spanning 20 ns in steps of about 300 ps. Delays were measured externally with a TDS5054B (Tektronix, Beaverton, USA) digital oscilloscope and the signals fed back to the FPGA input pins.
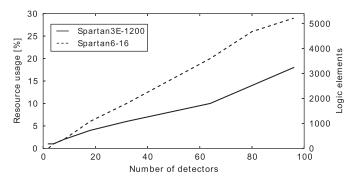
The measured coincidence resolutions are 10.4 ns FWHM for the single clock version, and 5.3 ns for the dual clock one (Fig. 4), as expected from our simulations. Preliminary trials make us expect that about 4.4 ns can be obtained with a four phases approach at 192 MHz.

3.2. Resources usage

The implemented synchronous gating network can resolve coincidences between two subsets, each made of a parameterized number of detectors. Coincidences can thus be detected between a detector from one subset and one from the other. However, this asset can be easily generalized to a greater number of subsets.

Fig. 5 shows the resources utilization for the single-clock processor version. For the dual phase processor the number of required cells is doubled. The resources dependency on the number of detectors is linear and in general very low even for tens of detectors. The I/O requirement is one input buffer per detector channel, i.e. one pin for single ended logic standards or two for double ended ones.

Table 1 shows the number of pipeline stages required for the gating network in order to satisfy the clocking constraints. Given the low number of required stages, higher-end FPGAs are expected not to require pipelining at all.

4. Conclusions

We have proposed a new FPGA-based, reconfigurable coincidence detection method, particularly useful for its simplicity and scalability. The method is implementable on any FPGA target device with standard HDL coding practices. The logic resources per detector pair are minimal even with lowest-cost devices. The technique also requires only one input buffer per channel, thus allowing to manage more than 100 channels within a single chip. The maximum achievable coincidence resolution is $3\tau/2$, which corresponds to 5.3 ns with the used prototype and is expected to be less than 2 ns in higher-end FPGAs.

References

[1] C. Damiani, et al., Nucl. Instr. and Meth. A 490 (2002) 356.
[2] B.K. Swann, et al., IEEE J. Solid-State Circuits 39 (2004) 1839.
[3] M. Conti, et al., Phys. Med. Biol. 50 (2005) 4507.
[4] D.P. McElroy, et al., Phys. Med. Biol. 50 (2005) 3323.
[5] P. Bento, et al., IEEE Nucl. Sci. Symp. Conf. Rec. (2004) 3796.
[6] S.J. Park, et al., IEEE Trans. Nucl. Sci. NS-55 (2008) 510.
[7] Y. Wang, et al., IEEE Trans. Nucl. Sci. NS-50 (2003) 1386.
[8] C. Wang, et al., IEEE Nucl. Sci. Symp. Conf. Rec. (2009) 3633.
[9] R. Fontaine, et al., IEEE Trans. Nucl. Sci. NS-56 (2009) 3.
[10] L.-S. Kim, R. Dutton, IEEE J. Solid-State Circuits 25 (1990) 942.
[11] N. Belcari, et al., IEEE Nucl. Sci. Symp. Conf. Rec. (2009) 3611.