Aliaga Varea, RJ.; Monzó Ferrer, JM.; Spaggiari, M.; Ferrando Jódar, N.; Gadea Gironés, R.; Colom Palero, RJ. (2011). PET System Synchronization and Timing Resolution Using High-Speed Data Links. IEEE Transactions on Nuclear Science. 58(4):1596-1605. doi:10.1109/TNS.2011.2140130.

PET System Synchronization and Timing Resolution using High-Speed Data Links

Ramón J. Aliaga, José M. Monzó, Michele Spaggiari, Néstor Ferrando, Rafael Gadea and Ricardo J. Colom

***Abstract*—Current PET systems with fully digital trigger rely on early digitization of detector signals and the use of digital processors, usually FPGAs, for recognition of valid gamma events on single detectors. Timestamps are assigned and later used for coincidence analysis. In order to maintain a decent timing resolution for events detected on different acquisition boards, it is necessary that local timestamps on different FPGAs be synchronized. Sub-nanosecond accuracy is mandatory if we want this effect to be negligible on overall timing resolution. This is usually achieved by connecting all boards to a common backplane with a precise clock delivery network; however, this approach forces a rigid structure on the whole PET system and may pose scalability problems.**

As an alternative, we propose a backplane-less PET system architecture in which DAQ boards are connected by single full-duplex high-speed data links. Data encoding with embedded clock is used to correct frequency differences between local oscillators. Timestamp synchronization between FPGAs with clock period resolution is maintained by means of data transfers in a way similar to the IEEE 1588 standard. Finer resolution is achieved by reflection of received clocks and phase difference measurement on the transmitter. It is crucial that data transceivers have very low latency uncertainty in order to achieve the desired timestamp accuracy; we discuss the availability of off-the-shelf hardware for these implementations.

***Index Terms*—Data acquisition (DAQ), positron emission tomography (PET), serial links, synchronization, timing resolution.**

I. Introduction

THERE is a trend among PET systems to move the digitization step closer to the photodetector outputs, implementing an increasing part of the required signal processing in digital devices. These changes are arguably motivated by advances in FPGA technology and availability of faster ADCs, whereby analog signals may be transferred to the digital domain with a smaller information loss. Research of systems with fully digital trigger, where trigger signals such as the last dynode output from PMTs undergo a minimal shaping stage before analog-to-digital conversion, is increasingly common [1]–[3]. Additionally, many PET systems are shifting from discrete implementation of the analog processing stages to integrated analog front-end architectures with higher performance [4]–[6], in order to increase spatial and time resolution.

Manuscript received Jun 14, 2010; revised Feb 21, 2011; accepted Mar 31, 2011.

This work was supported in part by the Spanish Ministry of Science and Innovation under FPU Grant AP2006-04275 and CICYT Grant FIS2010-21216-C02-02.

The authors are with the Instituto de Instrumentación para Imagen Molecular (I3M), Universidad Politécnica de Valencia, 46022 Valencia, Spain (e-mail: raalva@upvnet.upv.es).

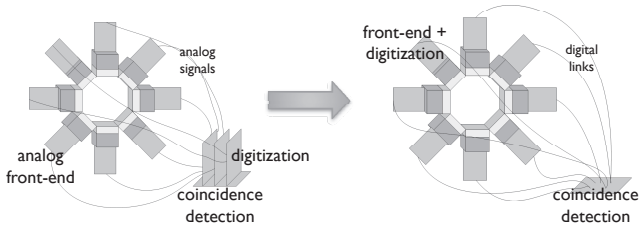

Fig. 1. Left: PET system architecture with a concentrated digital section. Right: Proposed PET system architecture with digitization on front-end cards and a separate central digital unit.

This has the added benefit of reducing board size requirements for front-end electronics.

In a usual PET system electronics setup, as shown in Fig. 1 (left), DAQ boards and higher-level digital processors are integrated with a common backplane, and thus physically separated from the detectors and front-end electronics [6]–[8]. However, the described trends naturally carry a tendency to integrate the increasingly simplified analog front-end, ADCs and first stages of digital processing together on the same circuit board. The resulting boards may be then mounted directly on the photodetector rings in order to reduce cabling and simplify system assembly. This electronic assembly, as shown in Fig. 1 (right), strongly favors the physical separation of the detector-and-DAQ board set from the central processing unit in the system, which would be responsible for coincidence detection, data collection and communication with the external device that performs image reconstruction.

The challenge for such a system setup lies in the synchronization of all DAQ components. This is easier with a single DAQ board or when all DAQ boards are within the same crate; in this case, synchronization is usually achieved with a system-synchronous clocking scheme, where a simple clock tree is implemented using clock buffers and then distributed to all DAQ boards through controlled backplane connections. Clock skew and jitter figures are rarely documented [9], although they can occasionally be inferred [10]. Sometimes, a combined systemwide skew and jitter figure is given [11], [12]. Usual values range from 150 to 500 ps, being higher for larger or physically distributed systems, such as Fig. 1 (right), due to accumulated time uncertainty and jitter from a larger number of clock distribution components, including cables. Constant skews may be automatically compensated by periodic calibration, along with trigger signal delay mismatches [13], [14], although calibration procedures are usually lengthy and costly [15], [16]. However, actual skew values are not entirely

deterministic and may show small variations between power cycles, as well as fluctuate with variations in operating conditions.

On the other hand, coincidence timing resolution in digital PET is continuously improving. Advances in timing algorithms and especially increases in ADC sampling rates allow tighter shapings to be applied to trigger signals so that pile-up and event time uncertainty are reduced. With FWHM coincidence resolutions below 2 ns [17] and expected to improve significantly [18], the effect of mis-synchronization between DAQ boards becomes increasingly important, since any discrepancy between reference times on acquisition FPGAs is directly reflected on the measured difference between timestamps from potentially coincident events. Hence, synchronization errors on the order of 500 ps already have a non-trivial effect on the measured timing resolution figures.

In this paper, we describe an alternative clocking and synchronization scheme that allows time references in different DAQ boards, connected only by single full-duplex high-speed data links, to be matched with a resolution in the 100 to 200 ps range, with synchronization operations taking up a small part of the whole data link bandwidth. We also propose a PET system architecture that takes advantage of this clocking method and allows the DAQ boards to be merged with the analog front-end directly on the detector location and physically separated from the rest of the digital processors with reduced cabling requirements while maintaining a timestamp skew on the order of 100 ps. While other similar synchronization methods such as [19] have been proposed, they mainly focus on the compensation of propagation delay mismatches and variations over very long links. In our case, we focus on maintaining a very low reference time uncertainty between boards and removing the need for external timing calibration.

II. Proposed System Architecture

Our proposed DAQ system architecture follows the hierarchic division of circuit boards outlined in Fig. 2, resulting in a tree topology. Boards on the lowest level are used to digitize detector signals, detect single gamma events and transmit them to their parent boards, connected to them only by means of a full-duplex data link. They may also contain the analog front-end and be directly coupled to the associated photodetectors; the number of detectors supported by each one is arbitrary. Boards on the higher levels are responsible for coincidence detection and may be connected with additional data links. The highest level contains a single board that concentrates all coincidence data and transmits them to an external processor that executes the image reconstruction algorithm.

Data connections between different hierarchy levels are set up as master-slave connections, the master being the higher level board. The slave node is able to replicate the local clock frequency in the master. This way, the local oscillator frequency in the top level board is distributed following the system tree structure and all boards run with exactly the same reference clock frequency. It is not necessary for slave clocks to be phase locked with the top level oscillator, just frequency locked.

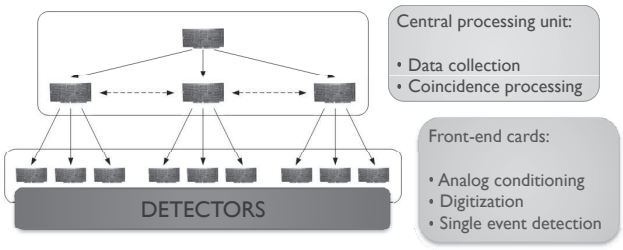

Fig. 2. Proposed circuit board hierarchy for a PET DAQ system. Solid lines represent data link connections. Arrows show the direction of clock frequency replication. Dashed lines are optional connections.

Each data link is able to synchronize master and slave with a given resolution τ_r, so local timestamps on different boards at the n-th level will be synchronized with a resolution bounded by $n \cdot \tau_r$. It is important to keep the total number of tree levels as small as possible in order to minimize the accumulation of time uncertainty. A maximum of three hierarchy levels seems appropriate, reducible to two for small systems such as single detector rings (i.e. a single parent board performing coincidence detection and a number of acquisition boards that are independently synchronized with the former). In order to achieve this, higher level boards should support as many downlink connections as possible.

III. Synchronization through Data Links

In this section, we explain how clock frequency replication and reference time synchronization with a resolution below the clock period can be achieved between FPGAs on physically separated circuit boards without a dedicated clock connection, using a single full-duplex data link. Since the link shall also be used for transmission of data related to the detected gamma events, the bandwidth reserved for synchronization operations should ideally be as small as possible.

A. Frequency Replication

Most modern FPGA families offer some sort of embedded high-speed transceiver hard IPs, such as the GXB on Altera devices [20] and the GTP family on Xilinx devices [21]. These hardware blocks, consisting of matched transmitter-receiver pairs, can be used to implement one-way serial links with maximum data rates being family-dependent but never lower than 3 Gbps. Such data rates can only be accomplished using self-synchronous signaling, where the transmitter data clock is embedded in the data signal. Fig. 3 shows the block diagram for a generic high-speed receiver. A PLL-based Clock Recovery Unit (CRU), seeded by an external reference clock, recovers the embedded serial data clock from signal transitions, whose frequency is equal to the link's bit rate, and then generates the parallel clock, whose frequency is equal to the word rate. The receiver then performs data sampling, deserialization, word alignment and physical layer decoding before handing the received word to the user through a clock-domain-crossing interface. It may be possible to bypass some of these blocks depending on implementation.

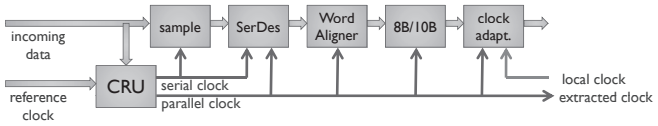

Fig. 3. Generic embedded high-speed receiver on FPGA.

Note that serial clock recovery is only possible if a minimum rate of signal transitions is guaranteed on the data link, in order to keep the CRU PLL locked. Hence, special physical-level coding must be used in order to force transitions even in the case of long strings of equal symbols. A typical approach supported by embedded FPGA transceivers is 8B/10B line coding [22]. This method encodes data bytes as sequences of ten physical bits, guaranteeing signal transitions in a bounded time interval (5 bit periods) under all circumstances, at the cost of 20% of the link bandwidth.

Although a different local oscillator may be used to clock the rest of the FPGA, the recovered parallel clock is available and can be used to drive all other logic blocks, so that transmitter and receiver nodes are clocked with exactly the same frequency. We can expand this method to achieve system-wide frequency syntonization following the hierarchy shown in Fig. 2, where each slave uses the recovered clock frequency to drive its synchronization logic, including the transmitters for its data links to lower nodes. In this way, every node in the system runs at the same clock frequency, derived from the local oscillator in the top node. The use of 8B/10B or a similar DC-balanced line code is mandatory in order to ensure that receiver PLLs stay locked at all times, as the loss of a received clock in a node carries malfunction of timing logic in said node and all other nodes dependant of it. Also, the transmitters must remain active at all times.

Two remarks must be made at this point. First, the previous discussion only contemplates the replication of clock frequency, disregarding the phase relationship between remote syntonized clocks. Second, using the recovered clock frequency to drive slave transmitters presents some practical problems due to jitter and internal transceiver structure. Both issues will be addressed in the following sections.

B. Coarse Synchronization

Each acquisition node runs a local timestamp counter. Even if all of them are clocked by exactly the same frequency, they may be started at different times and/or initial values and so they need to be synchronized. A standard method of timestamp synchronization across a data link is the Precision Time Protocol (PTP) [23]. The crucial step of said method is the transmission of the two synchronization frames shown on Fig. 4. A first frame is transmitted from master to slave, and local timestamp values at transmission and reception times, T_{M1} and T_{S1}, are recorded. A response frame from the slave is transmitted some time later, again recording transmission and reception timestamp values T_{S2} and T_{M2}. These four values can be used to compute local timestamp offset between master and slave as follows. Call t_{MS} and t_{SM} the time required for a frame to be transferred from master to slave and vice versa;

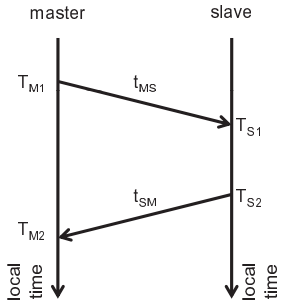

Fig. 4. Standard PTP two-frame synchronization procedure. Local arrival and departure timestamps are recorded and later used to compute one-way travel time in order to correct the slave timestamp offset.

these are initially unknown values. The aggregate flight time for both frames can be computed as

$$t_{MS} + t_{SM} = (T_{M2} - T_{M1}) - (T_{S2} - T_{S1})\,. \qquad (1)$$

PTP assumes that the link is symmetrical, i.e. $t_{MS} = t_{SM}$, so their value is obtained by halving the total flight time (1). The method can be extended to include the situation where both values are different but their difference $\Delta t = t_{SM} - t_{MS}$ is known, measured with a separate method. In this case, we can compute the values of t_{MS} and t_{SM} to know what the synchronized slave timestamp values should be. The slave timestamp counter is adjusted by adding

$$\Delta T_S = \frac{1}{2}\left[(T_{M1} + T_{M2}) - (T_{S1} + T_{S2}) - \Delta t\right]. \qquad (2)$$

After this operation, local timestamps on both ends are synchronized and remain so in the short term because their respective timestamp counters are locked to exactly the same clock frequency. Long term synchronization is maintained by periodical repetition of the above adjustment method.

In our application of this method, arrival times T_{S1} and T_{M2} are the values of the local timestamp counters at the time when the first word of the respective synchronization frame is read from the receiver block and captured by the PTP logic at its active clock edge. Likewise, departure times T_{M1} and T_{S2} are the local timestamp values when the first words of the respective synchronization frames are written to the transmitter block, again at the same PTP logic's active clock edge. Hence, t_{MS} and t_{SM} cover not just physical propagation delay, but also added transceiver latency between user logic on both ends. We may assume that the timestamp counter and PTP logic in the same FPGA use the same clock, and that the clocks in both FPGAs have the same period T_{clk}, although their local phase may be different. In this setting, we can identify two steps in the synchronization method where rounding errors are introduced that limit overall time resolution to T_{clk}:

On one hand, the total flight time as computed in (1) is always an integer multiple of T_{clk}, because both terms in parentheses represent the time difference between specific active edges of the same clock. This is an obvious accuracy loss, since flight time depends on propagation time through external transmission lines and might take any intermediate value. The fundamental error is given by the measurement of reception times T_{S1} and T_{M2}. The receiver is able to

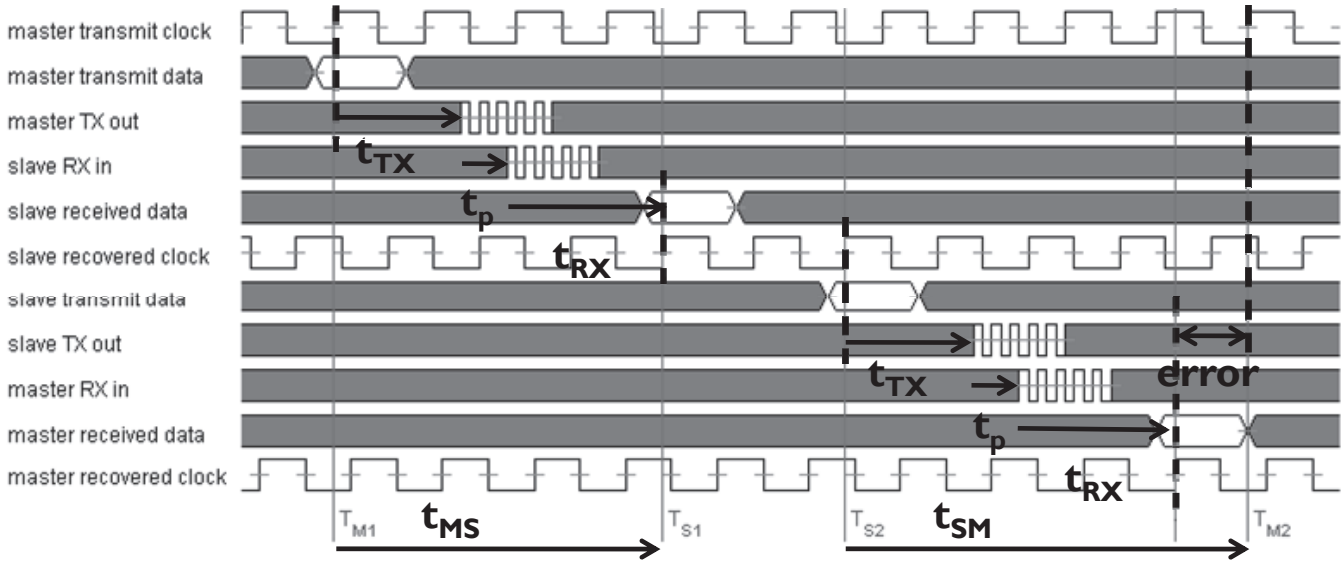

Fig. 5. Timing diagram for two-frame synchronization. Clock phase difference distorts the slave-to-master transfer time measure and needs to be compensated.

recover the parallel data clock from the received data stream so that the parallel clock's active edge is either perfectly aligned with the reception of a new data word, or misaligned with a readily available phase difference which can be inserted into the correction term Δt. However, there is a clock domain change between the receiver parallel clock and the timestamp clock. These clocks have the same frequency but there may be a phase difference that introduces an error of up to T_{clk} in the time measures. It is possible to avoid this error on one end by having the recovered parallel clock drive the local timestamp and PTP logic, but it is not possible to do this on both ends at the same time, because that would lead to an infinite loop in clock dependency. Fig. 5 shows a timing diagram for our selected implementation; here, the phase error is eliminated on the slave side by using the recovered clock directly (i.e. with no phase change) for the local synchronization logic and the other half-link's transmitter. However, the local and recovered clocks on the master side have different phases and an error is introduced, whose effect is to round T_{M2} up to the next active edge from the local clock, hence turning (1) into the nearest multiple of T_{clk}.

On the other hand, timestamp values are read from a simple counter with a period of T_{clk}. In particular, the correction factor given by (2) may only be applied with a resolution of T_{clk}, by changing the current value of the timestamp counter. Restricting timestamp values to integer multiples of T_{clk} neglects the fact that the phases of the local timestamp clocks on devices from different acquisition boards may be different. While this may have no effect on the synchronization procedure itself, it introduces timing errors when considering a third reference, such as the generation of gamma events whose arrival times to the acquisition boards have to be estimated with a finer resolution than T_{clk}.

The previous analysis suggests that several changes need to be applied to the usual synchronization scheme before we can guarantee resolutions well below T_{clk}. Local timestamp values need to have a fractional part that is able to be adjusted by the synchronization protocol, even if the local clock only updates its integer part. The correction term Δt needs to be measured with high accuracy; to achieve this, phase conditions between all featured clocks along the data path have to be controlled. Transmitter parallel clocks need to be aligned to local clocks, or have a constant and measurable phase difference. Likewise, the slave's local clock has to be aligned to the recovered parallel data clock or have a known phase difference. Finally, the phase difference between the recovered data clock and the local timestamp clock on the master side needs to be estimated accurately.

C. Transceiver Latency Requirements

In Fig. 5, partial flight time values t_{MS} and t_{SM} are decomposed into several terms: transmitter and receiver latencies (t_{TX} and t_{RX}), propagation time through physical media external to the FPGA (t_p), and an additional term to represent the phase error. Thus

$$\Delta t = (t_{TX,S} - t_{TX,M}) + (t_{RX,M} - t_{RX,S}) + (t_{p,MS} - t_{p,SM}) + \frac{\Delta\varphi}{2\pi} T_{clk} \quad (3)$$

where $\Delta\varphi$ is the phase difference, whose measurement method is explained in the next sections. We may assume that propagation times in both half-links are approximately equal if circuit board design and connection cable choice are adequate. Hence, in order to maximize the accuracy in the measurement of Δt, the first two terms in parentheses in (3) must have as little error as possible, i.e. transmitters and receivers need to have a very low latency uncertainty. We need to be able to know the exact value of $t_{TX,S} - t_{TX,M}$ and $t_{RX,S} - t_{RX,M}$ after every reset or power cycle.

Note that it is not necessary for t_{TX} and t_{RX} to remain constant across resets; they only need to be deterministic. For instance, it is common for receivers to have a latency characterization of the form

$$t_{RX} = C + n \cdot UI \quad (4)$$

where UI is the Unit Interval, i.e. the serial link's bit period, and n is an integer that can be different after each reset. Such an expression arises whenever the parallel clock generated by the CRU is obtained from the serial clock without regard to actual word alignment in the bit stream. In this case, the requirement that (4) be deterministic implies that C must be constant across resets and power cycles and the value of n must be available to the user logic. If these conditions are not met, external calibration would be required each time the system was turned on or reset. Ideally, C should also remain constant across instances of the same FPGA device, so a one-time calibration is not needed either, because constant terms in t_{RX} are canceled in (3).

D. Phase Measurement for Fine Synchronization

Measurement of the phase difference between the local timestamp clock and the received data clock can be implemented with the Digital Dual-Mixer Time Difference (DDMTD) method as described in [24]. This is an all-digital version of the well-known DMTD technique [25], where measured clocks are mixed with a tone with a similar frequency and filtered in order to obtain lower frequency signals with the same phase relationship. The main advantage of this method against other possibilities such as a Phase-Frequency Detector (PFD) is that it can be implemented entirely inside of the FPGA without external analog components.

Let $P_1(t)$ and $P_2(t)$ be the clock signals whose phase difference we want to obtain. These are assumed to be periodic signals with frequency $f_{clk} = 1/T_{clk}$. We generate a DMTD clock with frequency $f_D = 1/T_D$ and use it to sample both $P_i(t)$ signals; since we are dealing with digital signals, this can be implemented with simple flip-flops. The n-th sample is then

$$\begin{aligned} P_i(nT_D) &= P_i\left(n\frac{T_D}{T_{clk}}T_{clk}\right) \\ &= P_i\left(\left\{n\frac{T_D}{T_{clk}}\right\}T_{clk}\right) \end{aligned} \tag{5}$$

where $\{x\}$ denotes the fractional part of x; the second equality follows from the fact that P_i has period T_{clk}. This general formula can be simplified if we force a particular relationship between T_D and T_{clk}. The simplest case arises when

$$f_D = \frac{N}{N+1}f_{clk} \tag{6}$$

for an integer N. Then the n-th sample becomes

$$\begin{aligned} P_i(nT_D) &= P_i\left(\left\{\frac{n}{N}\right\}T_{clk}\right) \\ &= P_i\left(\frac{n \bmod N}{N}T_{clk}\right). \end{aligned} \tag{7}$$

Thus, we are effectively sampling one period of $P_i(t)$ that is stretched in time by a factor of N. In particular, the phase difference between the sampled signals is the same as between the original signals, but the time difference between rising edges of P_1 and P_2 is amplified by N and more easily measurable. Also, the set of samples has a period of N,

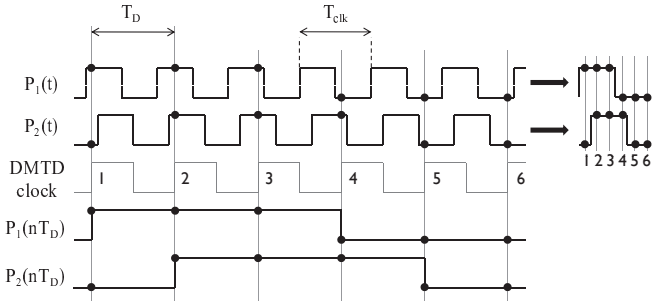

Fig. 6. Signal waveforms involved in DDMTD. On the right side, we show all input clock periods and sampling instants collapsed together, in order to illustrate the zooming effect of DDMTD sampling.

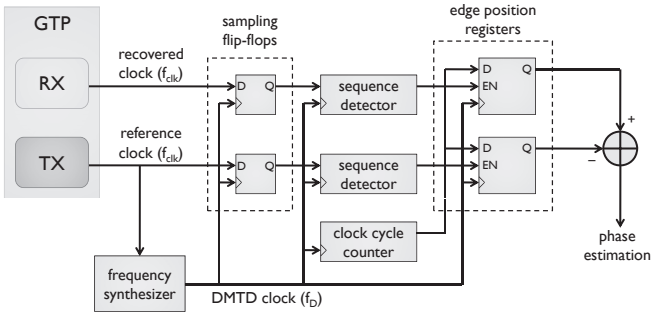

Fig. 7. FPGA implementation of all-digital DMTD.

assuming that the clock signals P_1 and P_2 remain stable. Fig. 6 illustrates the method with example waveforms for $N = 6$.

Fig. 7 shows an implementation of the phase measurement algorithm inside the FPGA. The DDMTD clock is synthesized directly from the most stable of both clock signals under measure by means of a PLL and/or DLL, so that it satisfies (6) exactly. Its phase relationship to the input clock signals is irrelevant. Both inputs are sampled and the logic looks for edges in the sampled bit strings; the sequence 0011 is interpreted as a positive edge. A synchronizer chain is used to limit metastability propagation. Whenever active edges in P_1 and P_2 are detected in sample numbers n_1 and n_2, respectively, we obtain a phase difference estimation given by

$$\Delta\hat{\varphi} = 2\pi\frac{(n_1 - n_2) \bmod N}{N} \tag{8}$$

Assuming all edges are detected, we obtain an estimate every $N+1$ input clock cycles. Phase difference is measured with a resolution of $2\pi/N$, so N represents a trade-off between time resolution and measurement time. Moreover, it is advantageous to make N a power of two, in order to simplify the implementation of the modulus operation in (8).

E. Phase Measure Refinement

Phase difference estimates are subject to errors derived from two primary sources. The first is metastability at the sampling flip-flops. The second is clock jitter; note that three different clocks are involved (both clocks under measure, and the generated sampling clock) and the jitter from each of them contributes to fluctuations in phase measurement. We remark that coincidence resolution is not affected by any of these jitter

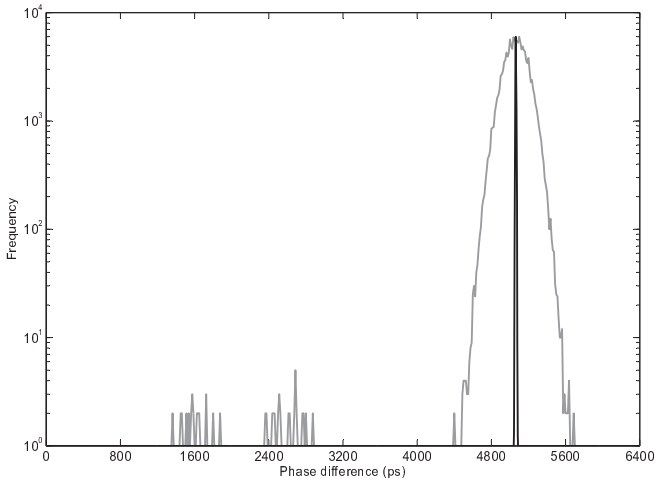

Fig. 8. Logarithmic histograms of DDMTD phase estimates for a link session, before (gray) and after (black) the clustering algorithm. The number of hits per bin has been increased by 1 because logarithm is not defined for 0. FWHM resolution is reduced from 280 ps to 10 ps, and absolute peak width is reduced from 1300 ps to 45 ps.

sources, but only by jitter on the ADC sampling clock: once the trigger signal has been digitized, it is effectively immune to clock cycle variations.

Both of these noise sources manifest themselves as a probability of error at the samples near the clock edges. Fig. 8 (gray) shows a typical histogram of raw DDMTD phase estimates for a single link session, where two distinct effects of sample uncertainty around the edges may be observed. On one hand, the large peak around the real phase difference has low resolution and its width depends on the combined jitter from all clocks. On the other hand, a small number of erroneous estimates appear near the opposite phase. These incorrect measures occur whenever a positive edge is detected as a negative edge or vice versa, e.g. after a string of erroneous consecutive samples near an edge. All of this implies that DDMTD outputs need to be filtered before an accurate phase estimate can be obtained. Because of the nature of these adverse effects, it is preferable to use a peak discrimination method for DDMTD post-processing instead of a simple moving average procedure.

We have chosen to implement a simple iterative method known as Kohonen's algorithm [26], that classifies phase measurements into one of two clusters, represented by values φ_0 and φ_1. For each phase estimation $\hat{\varphi}$, its distance to both representative values is compared and the smaller one determines the cluster to which it belongs. After that, the corresponding value φ_i is updated to move it closer to the phase estimation

$$\varphi_i[n+1] = \varphi_i[n] + \theta \cdot (\hat{\varphi} - \varphi_i[n]) \tag{9}$$

while the value from the other cluster maintains its current value. Parameter θ is chosen as a negative power of two in order to obtain an efficient FPGA implementation. This method eventually converges to a state where one of the values φ_i is confined to the center of the phase peak with a very small variation. The phase estimation output is simply the value of φ_i which gets updated more often. A saturated up/down counter is updated after each phase measurement depending on the selected cluster; the method is considered to have converged when the counter value remains within a certain distance of the saturation value. Fig. 8 (black) shows the histogram of filtered phase measurements after convergence for $\theta = 1/256$, consisting of a single, much narrower peak, with the same center value as the set of raw DDMTD measurements.

The performance of this clustering method is governed by the parameter θ. A small value will yield a narrower peak and hence better phase resolution, but convergence will be slower, as well as its response to changes in link conditions.

IV. Implementation Issues

A. FPGA and Transceiver Selection

The main constraint for the selection of a suitable platform for the implementation of these data links is the need for transceivers with a very low latency uncertainty. It is currently possible to accomplish this with off-the-shelf FPGAs from the two main vendors, Xilinx and Altera. The lowest-end solution with deterministic latency is the Xilinx Virtex-5 LXT family. Altera's roughly equivalent family, the Stratix II GX, features similar transceivers but it is not possible to configure them in a way that provides a deterministic latency mode, because several transceiver sub-blocks with unpredictable latency cannot be bypassed. One must resort to the newer, higher-end FPGA families from Altera to find deterministic latency modes. Such configurations are possible on Stratix IV GX devices according to available documentation [27], although we have not yet physically tested whether they actually support our data link specification. There is also the possibility to use an external transceiving chipset with a parallel data interface to the FPGA, however we have not found commercial transceivers with sufficiently small latency uncertainty. According to [28], the only off-the-shelf chipset with this feature was Agilent's G-Link [29], but its production has been discontinued.

Our tests have been implemented on the Virtex-5 LXT, so we focus on details for its transceivers, the GTPs [21]. In order to obtain deterministic latency, GTPs have to be configured in "RX phase alignment" and "TX phase alignment" modes. where GTP-generated parallel clocks, i.e. the recovered clock at the receiver and the internal transmit clock, are automatically phase-shifted so that the clock domain crossing FIFOs from Fig. 3 can be bypassed and no phase error is introduced. Under these conditions, t_{TX} is constant and t_{RX} is given by (4). The value of n depends on word alignment, changes after every reset and is not available to the user logic when using the automatic comma alignment in the GTP. This forces us to implement a custom word alignment state machine in user logic, that detects incorrect word boundaries and shifts the recovered clock using the GTP bit slip interface; n is then the total number of bits slipped until lock is achieved. Flight time skew (3) becomes then

$$\Delta t = (n_M - n_S) \cdot UI + \frac{\Delta\varphi}{2\pi} T_{clk} \tag{10}$$

where n_M and n_S are bit slip values in the master and slave, respectively, and $1\ UI = T_{clk}/10$ for 8B/10B single-width links.

B. Transceiver Clocking

The Virtex-5 GTPs are arranged into "tiles", i.e. groups of two GTPs (each including transmitter and receiver) sharing some clocking circuitry. In particular, they have a common reference clock input for their CRUs. The reference clock is used as a seed for the CRU PLL, but it is also used to directly generate the serial transmit clock for both transceivers. At a slave node, we want to transmit back to the master using exactly the same line rate as the master uses for transmission to the slave. To do so, it is necessary to have a GTP tile with a reference clock which is frequency locked with the oscillator from the master node, and locate the slave transmitter in it. There are three ways to accomplish this:

a) Transmit the master clock to the slave using additional cabling, and use it as the reference clock input for the GTP. This is possible but goes against our stated goals of minimizing link cabling.

b) Use the recovered clock as reference input for the transmitter GTP. Notice that we need to have a working reference clock for the receiver before we can establish the link from master to slave and then extract the recovered clock in the first place. This implies that we need to use two different reference inputs, and hence two different GTP tiles, for the receiver and the transmitter. Thus, this method probably forces us to waste GTP resources.

c) Modify scheme (b) so that the reference clock is generated by an oscillator that can be frequency-locked to the recovered clock after the reception link has been established. One possible circuit proposed by [30] is shown in Fig. 9. The reference clock input is driven by a voltage-controlled crystal oscillator (VCXO), which is part of a PLL using the recovered clock as input. The phase detector and loop filter are implemented inside of the FPGA and a DAC is used to feed the VCXO input. The VCXO is initially kept at its center frequency, and frequency locking starts when the recovered clock becomes available.

It must be noted that jitter requirements for CRU clock references are usually tight, but all of the proposed reference clock sources might feature a relatively high jitter level. For instance, total jitter in GTP reference clocks cannot exceed 40 ps [31], whereas recovered clocks have been shown to exhibit total jitter values higher by an order of magnitude [32]. For scheme (c), using a fully digital phase detector such as an Alexander detector [33] will also result in a VCXO output with relatively high jitter [34]. It is therefore advisable, and in most cases necessary, to add jitter cleaning circuitry to filter the reference clock before it enters the GTP.

C. Stability of Timestamp Synchronization

Our proposed implementation generates a measure of phase error $\Delta\varphi$ modulo T_{clk}. Since the phase measurement scheme has a finite resolution, measured phases will fluctuate around the center value. This creates a potential for error in the timestamp synchronization algorithm whenever $\Delta\varphi$ is close to the period crossing values. For example, if the phase difference is positive but very close to zero, it is possible for some samples of $\Delta\varphi$ to fluctuate below zero and be detected as $T_{clk} - \epsilon$ for small, positive ϵ. If one of those samples is used by the timestamp synchronization algorithm, Δt will have an error of T_{clk} and thus, by (2), timestamps will have a synchronization error of $T_{clk}/2$. Moreover, this error may appear intermittently as succesive iterations of the algorithm take values of $\Delta\varphi$ with alternating sign. The simplest solution to this issue is to establish guard intervals around the period crossing points, regard the link as unsuccessful whenever $\Delta\varphi$ falls within this range and force link resets until $\Delta\varphi$ has a safe value. Note that this increases average link establishment time, depending on the size of the guard intervals.

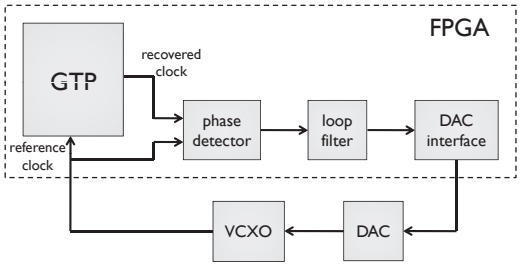

Fig. 9. Frequency replication circuit using only one GTP. Additional jitter cleaning circuitry may be necessary to filter the reference clock input.

D. ADCs and Delay Compensation

ADCs with serial outputs are very useful for trigger acquisition boards because the reduced number of digital traces simplifies board layout and reduces digital signal skew and integrity issues [35]. This requirement, coupled with a high sampling rate in order to improve timing resolution, results in ADCs with gigabit-speed transmitters and the need for GTPs to recover the sampled data in the FPGA. Latency in this data link, including ADC and FPGA receiver, is effectively added to the trigger signal latency as perceived by the FPGA. Delay mismatches between different boards should thus be corrected. Automatic correction allows us to use any (i.e. non-deterministic) transceiver to recover the samples, and avoids the need for manual calibration after each power cycle, as long as the photodetector-to-ADC delay remains constant.

We have chosen the AD9239 converter from Analog Devices [36], a quad-channel ADC that supports sampling frequencies up to 250 MHz with a high-speed serial output (up to 4 Gbps). We can take advantage of its multi-channel feature to implement the simple delay correction scheme outlined in Fig. 10. One of the ADC channels is used to sample an analog reference signal whose start can be triggered by FPGA logic. Receiver logic aligns all receiver channels so that the ADC-to-FPGA delay is identical for all of them by programming output test patterns in the ADC and selectively delaying input channels on the FPGA until the received patterns match exactly. The reference signal is then triggered and its start time is recovered from the read samples. This way, the time difference between the start of the signal and

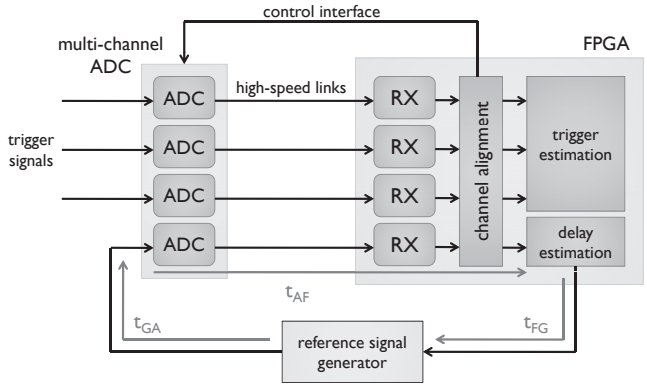

Fig. 10. Compensation circuit for trigger signal delay from ADC to FPGA.

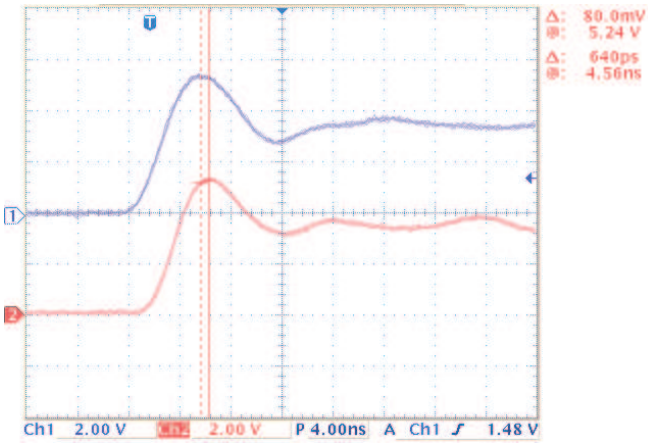

Fig. 11. Example oscilloscope screenshot showing the delay between the master (up) and slave (down) timestamp pulses. The proposed method yields a 620 ps delay, the oscilloscope measure is 640 ps.

its detection can be measured, which corresponds to the total delay $t_{FG} + t_{GA} + t_{AF}$; these constants stand for time delay from FPGA to reference signal generator, from generator to ADC input, and from ADC to FPGA, respectively. The measured value depends on electronic noise and jitter in ADC and FPGA clocks, so the procedure should be repeated several times in order to obtain an averaged, error-free value. The delay affecting the sampled trigger signals is only t_{AF}, however the other two components can be assumed to remain constant between identical circuit boards, so they do not affect coincidence resolution.

The reference signal needs to have an easily recognizable landmark indicating its digitally triggered start. We propose a simple analog linear ramp generator, because it allows simple online recovery of the start time with sub-clock cycle resolution by fitting a line to the recovered data. It is also insensitive to linear noise components, whereas high-frequency noise is handled by measure repetition and averaging. We must note, however, that this circuit's performance has not been tested experimentally.

V. Test Measurements

The proposed data link synchronization method was implemented using two Xilinx ML505 evaluation boards, each containing a XC5VLX50T device. Local oscillators were tuned to a 156.25 MHz frequency and data links were set up with 8-bit words and 8B/10B physical layer coding, resulting in a 1.5625 Gbps physical transmission speed and a 1.25 Gbps net data rate. These evaluation boards allow direct interface with the transmitter and receiver pins from a single GTP through SMA connectors. This means that clocking scheme (b) from section IV.B cannot be used, because we only have access to a single tile. Scheme (c) was implemented instead, using the circuit from Fig. 9 with a Connor-Winfield V702 156.25 MHz VCXO and a 16-bit, 180 kSPS DAC mounted on a small board connected to the ML505 expansion headers. The GTP reference clock input was filtered by one of the PLLs embedded in the FPGA. Master and slave FPGAs were connected only with four 1 m coaxial cables, implementing the two unidirectional differential data links. Both devices were configured with the same programming file; master or slave mode was selected with on-board switches, used to drive clock multiplexers and other logic inside the FPGA.

DDMTD measurements were implemented using $N = 512$ by cascading two frequency synthesis elements: a DLL and a PLL with frequency multiplication factors of 16/19 and 32/27, respectively. Using a PLL as the last element resulted in reduced jitter on the sampling clock and improved performance. Kohonen's algorithm was implemented with $\theta = 1/256$. The timestamp synchronization algorithm was implemented, using 48-bit timestamps with 12 fractional bits and computing the slave correction offset as (2) and Δt according to (10). Guard intervals of ± 1 UI were established around $\Delta\varphi = 0$. Since only the slave timestamp gets corrected, the master timestamp's fractional part remains 0, and so the slave timestamp's fractional part should yield directly the phase difference between master and slave clocks.

In order to validate the method, the FPGAs were configured to output pulses whenever their local integer timestamps fell within a certain range. These pulses were captured and monitored using a Tektronix TDS3014C oscilloscope, and the time difference between said pulses was measured. Different tests were performed resetting and/or power cycling the evaluation boards. In all cases, the synchronization algorithm converged in a single iteration; further iterations yielded timestamp correction offsets within ± 20 ps, which can be attributed to the clustering algorithm's finite resolution, due to clock jitter. As expected, the measured time delay from the master pulse to the slave pulse remained in the range between 0 and $T_{clk} =$ 6.4 ns, and matched the slave timestamp's fractional part with an error below 200 ps. A sample oscilloscope output is shown in Fig. 11. Only ten different delay values were observed, a different one being selected at random after each reset. This is consistent with the fact that the slave timestamp clock is just the recovered receiver clock, which has a delay of t_{MS} with respect to the master clock, and this value varies in steps of $T_{clk}/10$ due to slave receiver latency.

Note that measurements were performed with a low-bandwidth, low-resolution oscilloscope (100 MHz and 1.25

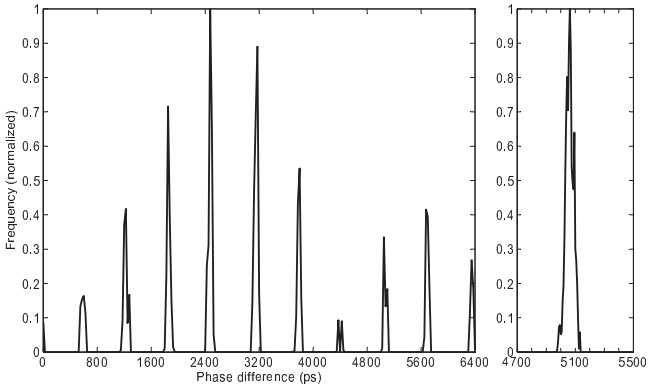

Fig. 12. Distribution of phase differences measured over 100 separate power cycles. Phase is displayed as the equivalent time delay for a 156.25 MHz clock. Left: raw phase measurements. Right: corrected for variation in receiver latency.

GSPS), hence these tests only provide validation of the synchronization method, but are not enough to adequately measure its resolution. An additional test was performed in order to obtain a more accurate resolution figure. In this case, a large number of phase estimations between transmitter and recovered clock were taken and recorded on the master FPGA without guard intervals. After that, the evaluation boards were turned off and on and new measurements were taken. The procedure was repeated 100 times and the combined histogram for all recorded phase measurements was obtained. The result is shown in Fig. 12 (left), where ten different, uniformly separated peaks can be observed. This is again consistent with the fact that all transmission latencies are constant except receiver latencies, which vary according to (4). Moreover, by computing

$$\left(\frac{\Delta\varphi}{2\pi} T_{clk} - (n_M + n_S) \frac{T_{clk}}{10} \right) \bmod T_{clk} \qquad (11)$$

the effect of varying latency can be eliminated and different phase peaks combined into a single one that yields an accurate estimation of overall phase resolution. Its histogram, represented in Fig. 12 (right), shows a single narrow peak with FWHM resolution of 60 ps and a peak-to-peak deviation of 200 ps.

VI. Conclusions

A method for the synchronization of sub-clock cycle resolution timestamps on two digital processors on remote boards has been presented. No dedicated hardware connections between acquisition boards are required, apart from a full-duplex data link supporting high data rates, which can be assumed to be present in the system anyway. However, the use of FPGAs with embedded high-speed transceivers that support specific low latency uncertainty configurations is mandatory. The clock distribution and synchronization scheme has been tested and validated for a single master-slave link. Unfortunately, an accurate value for achievable synchronization resolution could not be obtained due to low-resolution measurement methods. Instead, estimates on resolution bounds have been obtained by visual oscilloscope inspection and measure of phase difference resolution. These preliminary measurements suggest that individual links may be synchronized with uncertainty figures around 100 ps. This is similar to what can be achieved with a usual system-synchronous clocking scheme with a rigid structure, and should be good enough to avoid noticeable impact of board-to-board synchronization on coincidence timing resolution.

Additionally, a DAQ architecture for a PET system has been proposed that takes advantage of the above synchronization method, aiming to allow a flexible arrangement of digital acquisition boards without degradation of timing resolution for coincidence detection. A compensation scheme for trigger delay mismatches caused by the introduction of high-speed serial ADCs has also been proposed. However, none of this items have been experimentally demonstrated yet. Acquisition boards with analog input channels are currently under development. They will allow the testing of multi-level board hierarchies, as well as complete characterization of synchronization resolution by distributing common reference signals to different boards and comparing detected event timestamps.

References

[1] M. Streun, G. Brandenburg, H. Larue, C. Parl, and K. Ziemons, "The data acquisition system of ClearPET neuro - a small animal PET scanner," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 3 part 1, pp. 700–703, 2006.

[2] G. Hegyesi, J. Imrek, G. Kalinka *et al.*, "Ethernet based distributed data acquisition system for a small animal PET," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 4 part 2, pp. 2112–2117, 2006.

[3] R. Fontaine, F. Bélanger, N. Viscogliosi *et al.*, "The hardware and signal processing architecture of LabPET, a small animal APD-based digital PET scanner," *IEEE Trans. Nucl. Sci.*, vol. 56, no. 1, pp. 3–9, Feb. 2009.

[4] V. Herrero, R. J. Colom, R. Gadea *et al.*, "PESIC: An integrated front-end for PET applications," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 1 part 1, pp. 27–33, Feb. 2008.

[5] B. J. Pichler, W. Pimpl, W. Buttler *et al.*, "Integrated low-noise low-power fast charge-sensitive preamplifier for avalanche photodiodes in JFET-CMOS technology," *IEEE Trans. Nucl. Sci.*, vol. 48, no. 6, pp. 2370–2374, Dec. 2001.

[6] E. Albuquerque, P. Bento, C. Leong *et al.*, "The Clear-PEM electronics system," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 5, pp. 2704–2711, Oct. 2006.

[7] P. Bruyndonckx, Y. Wang, S. Tavernier, and P. Carnochan, "Design and performance of a data acquisition system for VUB-PET," *IEEE Trans. Nucl. Sci.*, vol. 48, no. 1, pp. 150–156, Feb. 2001.

[8] D. P. McElroy, M. Hoose, W. Pimpl, V. Spanoudaki, T. Schüler, and S. I. Ziegler, "A true singles list-mode data acquisition system for a small animal PET scanner with independent crystal readout," *Phys. Med. Biol.*, vol. 50, pp. 3323–3335, 2005.

[9] R. Fontaine, M.-A. Tétrault, F. Bélanger *et al.*, "Preliminary results of a data acquisition sub-system for distributed, digital, computational, APD-based, dual-modality PET/CT architecture for small animal imaging," in *Proc. NSS-MIC Conf.*, 2004, pp. 2296–2300.

[10] J. Imrek, D. Novák, G. Hegyesi *et al.*, "Development of an FPGA-based data acquisition module for small animal PET," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 5, pp. 2698–2703, Oct. 2006.

[11] C. J. Thompson and A. L. Goertzen, "A method for determination of the timing stability of PET scanners," *IEEE Trans. Med. Imag.*, vol. 24, no. 8, pp. 1053–1057, Aug. 2005.

[12] M. S. Musrock, J. W. Young, J. C. Moyers *et al.*, "Performance characteristics of a new generation of processing circuits for PET applications," *IEEE Trans. Nucl. Sci.*, vol. 50, no. 4, pp. 974–978, Aug. 2003.

[13] S. J. Park, S. Southekal, M. Purschke *et al.*, "Digital coincidence processing for the RatCAP conscious rat brain PET scanner," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 1, pp. 510–515, Feb. 2008.

[14] M.-A. Tétrault, N. Viscogliosi, J. Riendeau *et al.*, "System architecture of the LabPET small animal PET scanner," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 5, pp. 2546–2550, Oct. 2008.

[15] W. W. Moses and C. J. Thompson, "Timing calibration in PET using a time alignment probe," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 5, pp. 2660–2665, Oct. 2006.

[16] M. Bergeron, C. M. Pepin, L. Arpin *et al.*, "A handy time alignment probe for timing calibration of PET scanners," *Nucl. Instr. and Meth. A*, vol. 599, no. 1, pp. 113–117, Feb. 2009.

[17] J. M. Monzó, R. Esteve, C. W. Lerche *et al.*, "Digital signal processing techniques to improve time resolution in positron emission tomography," *IEEE Trans. Nucl. Sci.*, to be published.

[18] P. Guerra, J. E. Ortuño, G. Kontaxakis *et al.*, "Real-time digital timing in position emission tomography," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 5, pp. 2531–2540, Oct. 2008.

[19] P. Moreira, J. Serrano, T. Wlostowski, P. Loschmidt, and G. Gaderer, "White Rabbit: sub-nanosecond timing distribution over Ethernet," in *Proc. IEEE Int. Symp. on Precision Clock Synchronization (ISPCS)*, Oct. 2009, pp. 58–62.

[20] *Stratix II GX Device Handbook, Volume 2*, SIIGX5V2-4.3, Altera, 2007. [Online]. Available: http://www.altera.com/literature/hb/stx2gx/stxiigx_sii5v2.pdf

[21] *Virtex-5 FPGA RocketIO GTP Transceiver User Guide*, UG196, Xilinx, 2009. [Online]. Available: http://www.xilinx.com/support/documentation/user_guides/ug196.pdf

[22] A. X. Widmer and P. A. Franaszek, "A DC-balanced, partitioned-block, 8B/10B transmission code," *IBM J. Res. Develop.*, vol. 27, no. 5, pp. 440–451, Sep. 1983.

[23] *Standard for a precision clock synchronization protocol for networked measurement and control systems*, IEEE Std. 1588-2002, 2002.

[24] S. Liu, J. Tan, and B. Hou, "Multicycle synchronous digital phase measurement used to further improve phase-shift laser range finding," *Meas. Sci. and Tech.*, vol. 18, no. 6, pp. 1756–1762, 2007.

[25] D. W. Allan and H. Daams, "Picosecond time difference measurement system," in *Proc. 29th Ann. Symp. on Frequency Control*, Atlantic City, NJ, May 1975, pp. 404–411.

[26] T. Kohonen, *Self-Organization and Associative Memory*, 3rd ed. Berlin, Germany: Springer-Verlag, 1989.

[27] *Stratix IV GX Device Handbook, Volume 1*, SIV5V1-4.1, Altera, 2010. [Online]. Available: http://www.altera.com/literature/hb/stratix-iv/stratix4_handbook.pdf

[28] A. Aloisio, F. Cevenini, R. Giordano, and V. Izzo, "High-speed, fixed-latency serial links with FPGAs for synchronous transfers," *IEEE Trans. Nucl. Sci.*, vol. 56, no. 5, pp. 2864–2873, Oct. 2009.

[29] *Agilent HDMP 1032-1034 Transmitter-Receiver Chipset Datasheet*, Agilent, 2001.

[30] P. P. M. Jansweijer and H. Z. Peek, "Measuring propagation delay over a 1.25 Gbps bidirectional data link," National Institute for Subatomic Physics (NIKHEF), Tech. Rep. ETR 2010-01, May 2010. [Online]. Available: http://www.nikhef.nl/pub/services/biblio/technicalreports/ETR2010-01.pdf

[31] *Virtex-5 FPGA Data Sheet: DC and Switching Characteristics*, DS202, Xilinx, 2010. [Online]. Available: http://www.xilinx.com/support/documentation/data_sheets/ds202.pdf

[32] A. Aloisio, F. Cevenini, R. Giordano, and V. Izzo, "Characterizing jitter performance of multi gigabit FPGA-embedded serial transceivers," *IEEE Trans. Nucl. Sci.*, vol. 57, no. 2, pp. 451–455, Apr. 2010.

[33] J. D. H. Alexander, "Clock recovery from random binary signals," *Electronics Letters*, vol. 11, no. 22, pp. 541–542, Oct. 1975.

[34] J. Savoj and B. Razavi, *High-speed CMOS circuits for optical receivers*. Norwell, MA, USA: Kluwer Academic Publishers, 2001.

[35] D. Calvet, "A new interface technique for the acquisition of multiple multi-channel high speed ADCs," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 5, pp. 2592–2597, Oct. 2008.

[36] *Quad, 12-Bit, 170 MSPS/210MSPS/250 MSPS Serial Output 1.8V ADC*, AD9239, Analog Devices, 2010. [Online]. Available: http://www.analog.com/static/imported-files/data_sheets/AD9239.pdf

Available at http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5753966