19th IMEKO TC 4 Symposium and 17th IWADC Workshop
Advances in Instrumentation and Sensors Interoperability
July 18-19, 2013, Barcelona, Spain

IEEE1588 V2 Clock Distribution in FlexRIO Devices: Clock Drift Measurement

D.Sanz[1], M.Ruiz[1], J.M.Lopez[1],R.Castro[2],J.Vega[2], E.Barrera[1]

[1]*Grupo de Investigación en Instrumentación y Acústica Aplicada, Universidad Politécnica de Madrid, Madrid, Spain*
dsanz@i2a2.upm.es, mariano.ruiz@upm.es,juanmanuel.lopez@upm.es, eduardo.barrera@upm.es
[2]*Asociacion EURATOM/CIEMAT, Madrid, Spain*
rodirgo.castro@ciemat.es,jesus.vega@ciemat.es

Keywords: IEEE1588, RIO/FlexRIO, FPGA, Virtual instrumentation

***Abstract*-**Instrumentation in scientific experiments requires highest accurate to correlate acquired data with a time reference. In order to implement data acquisition channels synchronized to a reference clock, it is necessary to deploy a distributed clock based on computer devices with specific hardware for data acquisition, and to use a technology capable of synchronizing every computer in the system. Although there are several technologies to synchronize devices as the Precision Time Protocol (PTP) IEEE1588 V2 [1], not always is trivial to use it to get a good performance. This work studies how can be used the drift measures between clocks to get them synchronized. The solution has been implemented using a PTP-clock (implemented in a IEEE1588 PXI device), to every PXI/PXIe [2] and the FPGA based FlexRIO devices [3], able to reconfigure its hardware. The PXI/PXIe bus will be the key element to propagate the clock from the timing board to every RIO/FlexRIO card, achieving to phase their internal clocks with the PTP.

I. IEEE 1588 Overview

The PTP is a protocol defined to synchronize distributed devices (nodes) in a network. The protocol defines a PTP-Clock as every element in the system with the ability of synchronizing with a Master PTP-Clock (in the hierarchy there is a master clock implemented with an atomic clock, a GPS, etc). This means that every PTP-clock can behave as a Slave Clock (SC), or Master Clock (MC), depending on its accuracy attributes, e.g., the jitter of the internal oscillator. The objective of the PTP protocol is to get phase synchronization between every hardware real-time clock present in every PTP-Clock. At the moment, with the PTP V2 it is possible to synchronize nodes below one hundred nanoseconds, although there is an enhanced version, called White Rabbit PTP, [4] that reduces the drift up to 10 ps among nodes separated by 5 Km. This implementation requires a very specific hardware design, which is not yet standardized. To explain briefly the complex mechanism of the protocol, the basic rules and procedures are going to be exposed.

All the PTP-Clocks have to be interconnected in the same network, where there are defined different domains and subdomains. The network can be implemented with the IEEE 802.3 [5] standard, the most used, or other ones like: ControlNet, DeviceNet or PROFINET. Although IEEE1588 define these network interfaces for its implementation, other ones have been used to implement the PTP proving new alternatives like wireless networks, [6]. The Ethernet interface is the one chosen for presenting this solution.

Once nodes are connected, and its PTP-engine is started, every PTP-clock interchanges with other Clocks its attributes information, using specific UDP [1] PTP-messages. In this way, every clock can compare itself with the other ones, using the best master clock algorithm (BMC) [1], to establish which one is going to be the MC and which ones are the slaves.

From this point, every clock tries to put its internal clock in phase with respect to the master. The mechanism used by each PTP-Clock uses a software calculations and a specific hardware able to syntonize its clock, and register *TimeStamps*. *TimeStamp* registration is required for the PTP, to get precise time moments at the forwarding and reception of PTP-messages.

There are five types of PTP-Clocks [1] that permit several network configurations maintaining the accuracy. To simplify the synchronization explanation, only two ordinary PTP-Clock, one as master and the other one as slave, will be used in the following paragraphs.

PTP protocol can synchronize devices, using Ethernet bus, with an accuracy of tens of ns. To achieve this accuracy, there are several constraints to take into account. The PTP devices (PTP-Clocks) need: a) a software core that manages the evolution of the PTP during the synchronization, b) a hardware element that is able to register *Time-Stamps* every time, an event PTP-message is sent or received through the Ethernet physical layer.
The way to get synchronization consists on a message exchange between the Master and the Slaves. The Master sends periodically two event messages: Sync and Follow up, (if two-step is configured), in which the outbound *TimeStamp* (*t1*) is included in the Follow up message. The outbound is considered just when the message passes to the physical layer of the Ethernet Stack protocol. In the other side, every slave PTP-Clock, *TimeStamps* the received event message, in (*t2*). In this way, every Slave PTP-clock is able to calculate the offset time respect to the Master PTP-Clock, using equation (1). But the result obtained doesn't permit to know the path delay introduced by the physical layer. Therefore every Slave PTP-clock, sends to the Master Clock a *Delay_Req* message, generating time (*t3*) and (*t4*) in the inbound packet reception. When Slave PTP-clocks receive the *Delay_Res*" message, they can apply the equation (2), being able to solve the unknown *MeanPathDelay*. Additional to this process, the Slave PTP-Clock, needs additional hardware mechanism, to adjust its internal clock, to phase respect the Master PTP-Clock. This is due to the inherent jitter of every oscillator used for the internal clock. Then, with the continuous offset measurement, and phase adjustment, it is possible for every Slave PTP-Clock to obtain an approximate drift, respect to the actual master Clock, by mean a control system procedure, that each manufacturer or developer decide how to integrate in the PTP-clock.

$$Offset = t2 - t1 - MeanPathDelay \qquad (1)$$

$$MeanPathDelay = [(t2 - t1) + (t4 - t3)]/2 \qquad (2)$$

II. Typical implementationof the IEEE1588 in a data acquisition (DAQ) system

Although the easiest way to implement the PTP is using software and hardware, in the last days, only hardware implementations are emerging [7]. Some of the most useful examples that use PTP, consist on a distributed DAQ system [8] able to *TimeStamps*: the samples acquired, and the hardware events detected. Commercial solutions, that integrate PTP protocol with high throughput DAQ devices, are not available and only customized COTS products or academic solutions, could achieve the objectives presented in this paper. The Figure 1 presents a basic schema of a possible solution to implement a synchronized DAQ system, composed by: a CPU with an operating system to manage all the devices, a timing board compliant with the IEEE 1588 V2 protocol, and *N* DAQ devices.

To implement *TimeStamping* for both acquisition and register events, the solution would require a set of hardware interruptions to indicate to the timing board all the acquired samples and possible events to *TimeStamping*. For this proposal, the communication between the timing board and the DAQ devices will require a customized design. If every DAQ device has several analog input channels, and the sampling rates required are more than 1MS/s, the timing board will require a set of buffers with high capacity to storage: *TimeStamps* for every channel from each DAQ device, and *TimeStamps* for every possible trigger event. To achieve high throughput bandwidth to pass all data acquired from DAQ devices, and all the *TimeStamps to* the CPU, it will need a high performance bus, e.g., PCIe. But the complexity of this implementation still endures, because of the DAQ system process has to dequeue all the data acquired and assign correctly their *TimeStamps*, overloading the CPU process. It seems obvious to think that the best solution for this proposal will be to integrate the PTP into every DAQ device. This will require additional costs, and if the number of DAQ cards in the distributed system, reach hundreds of thousands, PTP can lose some accuracy [9], due to every DAQ device will behave as a PTP node with its Ethernet port interface. Propagate the PTP clock to every DAQ devices [10], can be a solution, because of it doesn't needs new hardware costs designs, and for the PTP network, the number of PTP clocks continuous being the same.

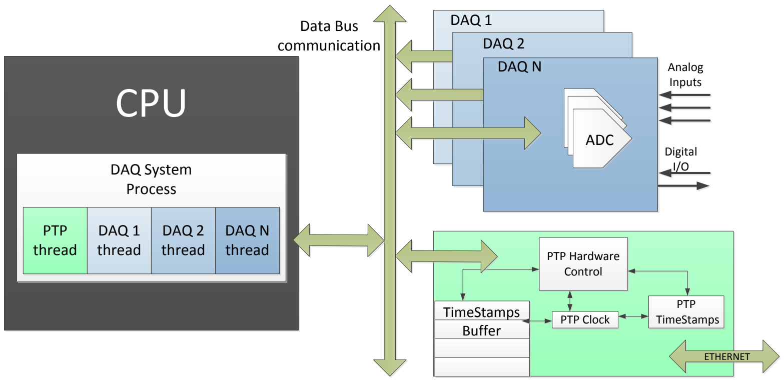


Figure 1. Schema of an example of synchronized DAQ system using IEEE 1588 PTP protocol

III. Hardware configuration for synchronize DAQ boards with a PTP device, measuring drift between clocks

Below is presented the hardware configuration to measure the drift and the offset to use it to get synchronization between them: a) A National Instruments PXI controller with an Intel Pentium 4 CPU, at 2.0 GHz, with 768 MB RAM memory, with LabVIEW real tim. b) A NI-PXI 6682 (PTP timing board) compliant with the IEEE1588 V2 protocol, c) two PXI-7851R RIO devices as DAQ cards, d) a NI PXI-1042Q chassis capable of housing all the devices exposed above. Although only RIO devices have been used, FlexRIO devices could be used in the same way, and with the same hardware configuration. Despite of asseverate that non-commercial solutions could solve the PTP limitations problem, the reason because this configuration is capable of implements the solution is due to the RIO/FlexRIO are reconfigurable devices, and a new communication protocol for synchronization has been defined. The RIO/FlexRIO devices are based on FPGA, this reason makes possible to implement into them not only the hardware for the acquisition but also a hardware Clock able to synchronize with the PXI 6682 timing board. Next sections explain the NI PXI 6682 timing board capabilities, and the RIO devices to have all them synchronized.

A. NI PXI 6682 timing board

The NI PXI 6682, is totally compliant with the IEEE 1588 V2 protocol. This device can synchronize its PTP clock with the rest of the nodes, through an Ethernet port. Then the clock reference for the entire chassis (controller and RIO devices) will be supplied by this timing board. As every PXI device, the timing board has a PXI bus, through by, It can generates digital signals to be read by the rest of the devices connected to the backplane of the chassis. Two PXI trigger lines are used to generate: 1) programmed future time trigger events, to indicate to the RIO devices the moments when they have to start the synchronization, see Figure 2. 2) A periodic pulse of 1 second to be used by the RIO devices, to measure the drift respect to the timing board. When the timing board will synchronize with other PTP nodes in the PTP network, this one could be MC or SC, as described in section 1. This will affect to the precision and accuracy synchronization of the RIO boards. This is due to the states of the timing board: a) When the timing board is MC, its hardware clock doesn’t get adjustment. The accuracy of the clock is characterized by the jitter of its internal oscillator. b) When the timing board is SC, its clock is continuously getting adjustment respect to the MC. Then every RIO board will be adjusting its clocks respect to a clock that is adjusting too. This situation will increase the drift oscillation between the timing board and the RIO devices.

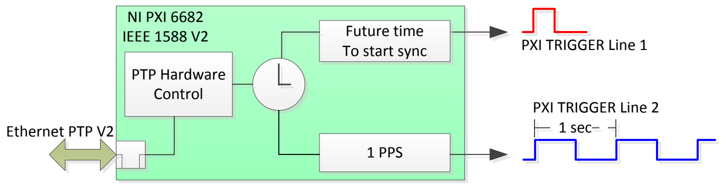


Figure 2. PXI 6682 timing board configuration for PTP-Clock distribution through PXI trigger line 1 and 2

B. RIO devices

RIO devices are based on FPGA and PXI technology.These technologies permit the implementation of intelligent DAQ systems, and all additional hardware functionalities required. With these flexible devices, it is possible to customize how the data acquired is packaged and passed to the CPU memory, through a PCI bus, and interconnect with all the devices inserted in the backplane of the chassis through the PXI trigger lines, as shows Figure 3. In this way all RIO devices will be interconnected with the timing board through the backplane of the chassis. To equip all the RIO devices with a clock able to synchronize with the timing board, the following items will be implemented:

- A hardware clock with PTP format [1]: a 64 bit register for seconds, and 32 bit register for nanoseconds. The sequential logic for increment this clock is configured with a 100MHz clock, in this way the resolution of the internal clock of every RIO device is of 10 ns. This clock will be updated taking into account the measures obtained.
- A specific counter that counts the clock cycles elapsed between rising edges detected in the PXI trigger line 2, to obtain the drift per second with respect to the timing board. See equation 3.

$$Drift = \left(10^8 - Clock_{cycles}\right) \times 10ns \qquad (3)$$

- An algorithm that uses the drift measured to adjust the hardware clock . This algorithm consist in add or substract a nanosecond unit every At_{drift}ns. See equation (4).

$$At_{drift} = \frac{10^9}{Drift} ns \qquad (4)$$

- An algorithm to measure the offset with respect to the timing board every second. This offset measurement will be used to adjust the clock, because of only the drift adjustment is not enough to obtain synchronization. Once the offset is obtained, a nanosecond unit is added or subtracted every At_{offset}ns, see equation (5). This offset values are used to obtain the results presented in the section IV.

$$At_{offset} = \frac{10^9}{offset\_From\_Timing\_Card} ns \qquad (5)$$

Once every RIO device has their own hardware clock, they can use it to *TimeStamp* every acquired sample, and every hardware event detected through digital inputs.

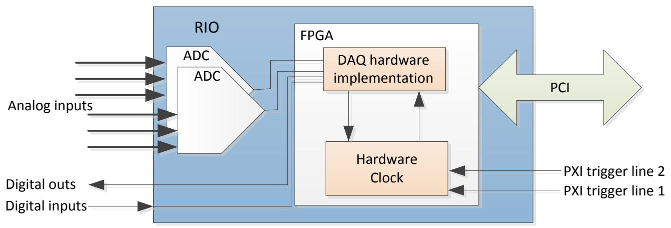


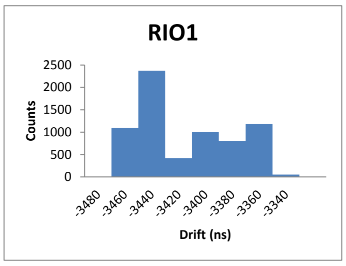
Figure 3. RIO device Configuration for implement a clock synchronized with a PTP-Clock

IV. Results: Drift and offset measures

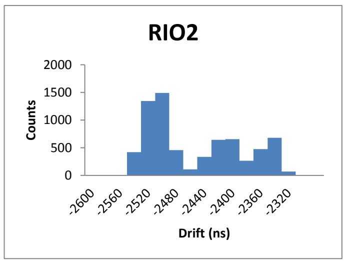
A software application, implemented in LabVIEW Real Time [11] running in the controller of the chassis, is required not only for the synchronization, but also to manage all the devices in the chassis. The application consist in let the user choose in which instant of time the synchronization process for the experiment will start. The experiment will study three cases. The first one, will get the drift and the offset measures from the FPGA calculations when the timing board is MC. The second one, will take the same measures from the FPGA, but with the timing board being SC in the PTP network. The third experiment will measure the offset between the timing board, being MC, and the RIO boards using and oscilloscope.

A. Results of the experiment 1: PXI 6682 board is master clock in the PTP network

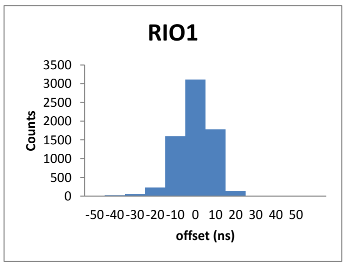
Each RIO board presents a different drift measurement with respect to the same timing board. The difference is about 1us. Being the timing board MC in the PTP network and the clock reference for both RIO devices, The offset obtained is similar with a synchronization precision of ±50ns, (see Figure 4).



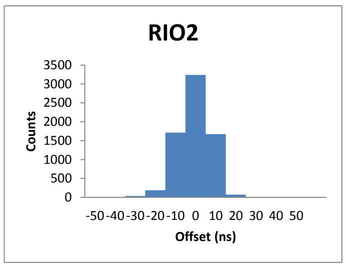
a)



c)



b)

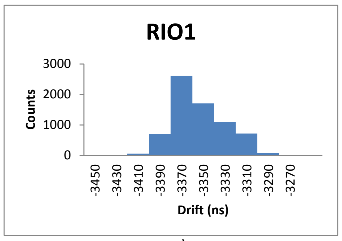


d)

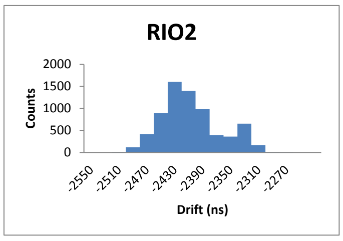
Figure 4. Drift and offset measures with respect to the timing board (being MC in the PTP network) of the RIO1(a, b), and RIO2 (c,d)

B. Results of the experiment 2: PXI 6682 board is slave clock in the PTP network

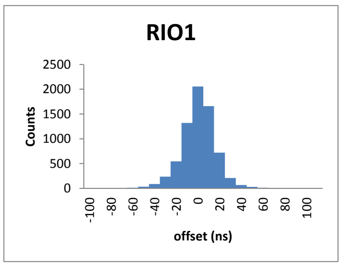
In this case, the timing board (being SC) is adjusting its drift with respect to other PTP MC clock continuously, producing that the maximum drift measured by one of the RIO devices (RIO1) increases in 40ns, (see Figure 5).



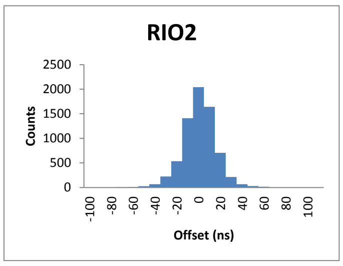
a)



c)



b)



d)

Figure 5. Drift and offset measures with respect to the timing board (being SC in the PTP network) of the RIO1(a, b), and RIO2 (c,d)

C. Results of the experiment 3: PXI 6682 board is master clock in the PTP network

The way used to obtain the offset between every RIO device with respect to the timing board with an oscilloscope (Agilent oscilloscope model: M5O7014A, with a sampling rate of 2GS/s, and a bandwidth of 100MHz) consists of programming every device to generate a digital trigger every second increment. After

17620 measures, one measure per second, the statistics results are exposed in the Table I.

TABLE I STATISTICS OF SYNCHRONIZATION BETWEEN TIMING BOARD AND RIO DEVICES MEASURING DIGITAL PPS TRIGGERS WITH AN OSCILLOSCOPE

Device Sync	Mean offset	Min offset	Max offset	Std Deviation
RIO1	-7.94ns	-61.5ns	13.5ns	7.42ns
RIO2	-1.75ns	-43.0ns	20.0ns	7.74ns

IV. Conclusions

The experiments executed to test the drift measurement proposal using a PXI timing board compliant with PTP V2, and RIO devices implementing a mechanism to synchronize its internal hardware clocks with the timing board, evidences the validation of the theoretical use of the drift measures between clocks to get synchronization.

In both cases, being the timing board MC and SC, it is demonstrated that independently of the drift variations among the time, the offset between clocks maintains the same precision. Obviously the precision is different in each case.

Comparing the results of the experiments 1 and 3, both cases in which the timing board is MC, is observed that the maximum offset are very similar. Two different ways of measure with similar results enforce the obtained results.

The regulator system implemented to adjust the internal clock of every RIO device, waits always 1s to obtain the correction factor, to be applied during the next second. One second is enough time to notice a great difference between the last correction factor applied and the ideal one, due to the jitter of the reference clock (timing board) and the internal oscillator of every RIO device. If drift and offset measures will be made with higher frequency, the correction would be made earlier and the precision will increase.

V. References

[1] IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)* , vol., no., pp.c1-269, July 24 2008.
[2] PXI Systems Alliance. [Online]. Available: http://www.pxisa.org/.
[3] RIO and FlexRIO Technology. [Online]. Available: http://www.ni.com/rseries/, http://www.ni.com/flexrio/.
[4] Lipinski, M.; Wlostowski, T.; Serrano, J.; Alvarez, P., "White rabbit: a PTP application for robust sub-nanosecond synchronization," *Precision Clock Synchronization for Measurement Control and Communication (ISPCS), 2011 International IEEE Symposium on* , vol., no., pp.25,30, 12-16 Sept. 2011.
[5] IEEE Standard for Ethernet - Sections 1-6," *IEEE Std 802.3-2012 (Revision to IEEE Std 802.3-2008)* , vol., no., pp.1,0, Dec. 28 2012.
[6] Hyuntae Cho; Jeonsu Jung; Bongrae Cho; Youngwoo Jin; Seung-Woo Lee; Yunju Baek, "Precision Time Synchronization Using IEEE 1588 for Wireless Sensor Networks," *Computational Science and Engineering, 2009. CSE '09. International Conference on* , vol.2, no., pp.579,586, 29-31 Aug. 2009.
[7] Correira, M., Sousa, J., Combo, A., Rodrigues, A.P., Carvalho, B.B., Batista, A.J.N., Gonçalves, B., Correia, C.M.B.A., Varandas, C.A.F., „Implementation of IEEE-1588 timing and synchronization for ATCA control and data acquisition systems," Fusion Engineering and Design, 87 (2012) 2178-2181.
[8] D.Sanz, M.Ruiz, R.Castro, J.Vega, J.M.Lopez. E.Barrera, N.Utzel, P.Makijarvi."Implementation of Intelligent Data Acquisition Systems for Fusion Experiments Using EPICS and FlexRIO Technology". *IEEE-NPSS Real Time Conference*, 2012.
[9] Dinh Thai Bui; Dupas, A.; Le Pallec, M., "Packet delay variation management for a better IEEE1588V2 performance," *Precision Clock Synchronization for Measurement, Control and Communication, 2009. ISPCS 2009. International Symposium on* , vol., no., pp.1,6, 12-16 Oct. 2009.
[10] D.Sanz, M.Ruiz, R.Castro, J.Vega, J.M.Lopez. E.Barrera. "IEEE1588 Clock Distribution for FlexRIO Devices in PXIe Platforms". IAEA 9th Technical Meeting on Control, Data Acquisition and Remote Participation for Fusion Research 2013.
[11] NI LabVIEW Real-Time. [Online]. Available: http://www.ni.com/labview/realtime/.