COMPARISON OF SYNCHRONIZATION LAYERS FOR DESIGN OF TIMING SYSTEMS

A. Aulin Söderqvist, N. Claesson, J. Neves Rodrigues, Lund University, Lund, Sweden*
R. Tavčar, R. Štefanič, J. Dedič, Cosylab, Ljubljana, Slovenia†

Abstract

Two synchronization layers for timing systems in large experimental physics control systems are compared. White Rabbit (WR), which is an emerging standard, is compared against the well-established event based approach. Several typical timing system services have been implemented on an FPGA using WR to explore its concepts and architecture, which is fundamentally different from an event based. Both timing system synchronization layers were evaluated based on typical requirements of current accelerator projects and with regard to other parameters such as scalability. The proposed design methodology demonstrates how WR can be deployed in future accelerator projects.

INTRODUCTION

The timing system (TS) is an essential part of the control system (CS) in current accelerator projects. In general, the TS provides services to the CS according to the requirements of the accelerator, see Fig. 1. Commercial off the shelf (COTS) products with adequate firmware are available that fits certain accelerators. On the other hand, usually different machines have unique requirements that make customization inevitable.

A given COTS product together with part of its firmware may be called a synchronization layer (SL), as it provides synchronous action and a well defined interface, see Fig. 2. This article reviews two different SLs using this definition. Their interfaces are compared as they have direct implications on how the TS can be implemented.

A limited set of TS services were implemented to evaluate the interfaces of White Rabbit (WR) [1]. Other TSs were also studied in combination to this implementation, i.e., the Real-time Event Distribution Network (REDNET) [2], the SINAP timing system [3] and the General Machine Timing system (GMT) [4].

REDNET is a recently finished, Micro-Research Finland (MRF) based [5], TS, which is being deployed at MedAustron. MRF is a well-established event based SL.

The SINAP timing system is another event based TS, developed for SSRF (SINAP, Shanghai) and already successfully installed in other facilities.

GMT is being developed for GSI/FAIR and will be WR based. WR is an open source project and the reference implementations are manufactured by several suppliers.

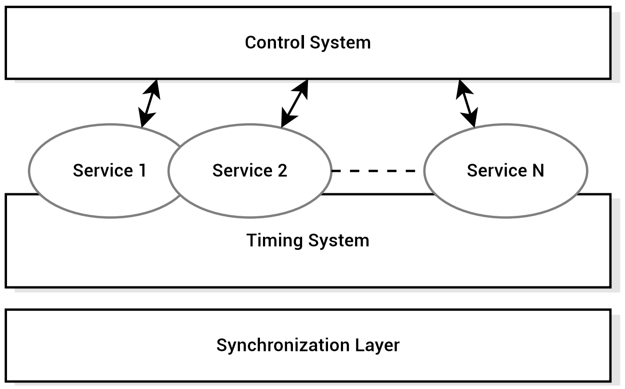

Figure 1: The control system needs several timing critical services, which are provided by the timing system. The timing system itself is constructed on a synchronization layer which provides the basic capabilites to enable implementation of such services.

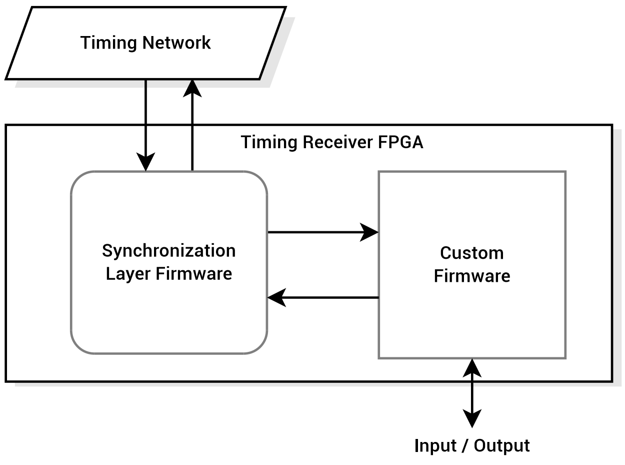

Figure 2: The timing receiver firmware can be seen as two separate modules. The synchronization layer, which is given by the vendor (MRF/WR), and the custom part, which is implemented as part of the timing system.

TIMING SYSTEM SERVICES

Given the task to design a TS it can be tempting to take a COTS product to see its capabilities and base the TS implementation on existing functionality. If a COTS is chosen before the real requirements are known it can lead to unnecessary work with adaptation of its features to match the required services. Instead requirements need to be defined

* et07aa0@student.lth.se, et07nc7@student.lth.se, joachim.rodrigues@eit.lth.se
† rok.tavcar@cosylab.com, rok.stefanic@cosylab.com, joze.dedic@cosylab.com

ISBN 978-3-95450-139-7

before choosing an existing COTS product to decide if it is sufficient or if a custom solution is needed.

There are typical requirements like accuracy and precision on a TS which are tightly coupled to the SL. Other than that there may be a multitude of other requirements that need special attention and more extensive investigation. Requirements need to define the services of the TS and decide the SL with lowest complexity fulfilling the needs.

Fundamental Services

Common to all TS is the need of coordinated actions in the distributed nodes. The CS issues a request for certain behaviour in the accelerator and it is up to the TS to carry out the request. This often leads to a sequencer which continuously generates actions to the Timing Receivers (TR), the actions which are needed to carry out the request from the CS. Depending on requirements there can be multiple sequencers.

TRs usually have trigger lines to front-end-devices to signal them at the right moment. Trigger lines are digital outputs generating timing critical pulses. These pulses must be executed at a specific point in time and have configurable length.

Other than triggering devices there is also a need of time stamping to provide accurate information on when any event, input or output, actually occurred. This is tightly coupled with tracking of time.

Machine Specific Services

There are also machine specific services that needs to be provided. An example is Virtual Accelerators (VAccs), which are implemented in REDNET [2]. Multiple VAccs run on the same accelerator. This allows multiple users to access and simultaneously reconfigure the TS via separate VAccs. It makes fast switching between configurations possible, which minimizes downtime. While an accelerator project is finalized it enables parallel commissioning of different parts.

Besides this there are often requirements for real-time data services to the TRs. For example front-end devices connected to TRs need configuration data.

SYNCHRONIZATION LAYER

The SL is an abstraction layer that conceals the complexity of synchronization from the TS, thus providing synchronous execution possibilities at distributed nodes. Knowing the difference between clock and time is an important prerequisite to understand SLs.

Clock, in the digital realm, is only pulses that are repeated at a given frequency. By letting two pulse trains have the same frequency and phase they are synchronized.

Time, on the other hand, can be defined in multiple ways. Usually it is the time past since a reference moment. In particle accelerators it is often a derivative of TAI (Temps atomique international, French name), which is the weighted average of over 200 atomic clocks, given in seconds. This can be obtained from GPS (Global Positioning System) signals.

Synchronization

MRF and WR synchronizes in two conceptual different ways. In MRF the most important synchronization is the clock. WR instead primarily focuses on synchronizing time.

Event-based such as MRF and SINAP is therefore considered real-time, with neglectable downlink latency, and the TRs act immediately. Whereas WR is less real-time due to the Ethernet standard. Both TSs have synchronized clock and time. But, since WR rely on absolute time for synchronized actions, it has to have a more precise time synchronization.

Interfaces

In event-based TSs the timing master (TM) sends an identifier over a synchronized network which all TRs simultaneously receive. This, in addition to a real-time distributed bus, is seen as its interface to the custom logic inside the FPGA (Field-programmable gate array) firmware.

On WR there are two interfaces, one for timing and one for communication. The timing interface gives you current TAI and a clock. The clock has the same frequency and phase in all TRs down to less than 1 ns. The communication interface is a MAC interface and can be used as a communication channel between all TMs and TRs.

Comparison

The performance measurements available cannot be compared in a fair manner. Instead characteristics which affect the TS implementation are given. Only WR and MRF performance figures are addressed, aiming to give a hint of the differences due to synchronization approach specifics.

The upstream performance is important because it is crucial for the services that require a loop-back from the TRs to the TM. MRF has a fast, but limited, concentrator and WR has a slower, but more generic, switch. The MRF concentrator delay is in the 100 – 200 ns range, according to [6]. The lowest delay achievable with a WR switch is in the 10 – 20 µs range, according to [7]. Since the WR switch is not finalized this may change in the future.

One study [8] has measured the offsets between the synchronized outputs of two WR nodes. This is called accuracy and it was measured to 517 ps at that set-up. Any similar published numbers for MRF was not found.

The same study also showed that the performance (jitter) between the outputs of two WR nodes was 119 ps. In MRF, the jitter given for one card is 15 ps RMS [5]. Assuming that there is no influence because of the network, this can give a hint about MRF performance.

Measuring accuracy and precision in a SL depends heavily on the set-up, number of fan-outs, cable length, environ-

ISBN 978-3-95450-139-7

mental differences, etc. In the future a dedicated test bed will be set-up to compare the SLs.

EXAMPLE IMPLEMENTATION OF A TIMING SYSTEM

The goal for the implemented TS is to explore the concept of WR. A fundamental functionality is time stamping of inputs and outputs as well as to trigger outputs. It is straight forward to implement time stamping in WR, which is why this section deals mostly with how to trigger outputs at certain moments, called trigger lines. These are realized by the block called GPIO Core in the architecture, Fig. 3.

The WR firmware [9] provides two interesting interfaces in TS point of view. There is the timing interface which contains all timing information needed. It provides a precise clock and the exact time with 8 nano seconds granularity. It also provides a MAC interface which enables network communication via Ethernet frames.

There is a fundamental difference between a TS implemented using WR and one using an event-based SL. WR has the exact time while the event-based system acts immidiatley on a received event. Because of this, designing the TR has to have a completely different approach.

In WR there is a high data overhead since it uses Ethernet frames for communication. If a TS based on WR is designed conceptually the same as a TS on an event based SL, TM pushing events downwards at the same rate as they are emitted, the rate of events received at the TR would be substantially lower. To alleviate this problem, local buffering of network messages in the TR is needed.

To ensure a high emit rate it is not enough to buffer incoming messages, the overhead from Ethernet will still be a limiting factor. This is addressed by having sequences of output patterns stored in the TRs instead of the TM. The messages that are sent to the TRs are mapped with the sequences. Sequences stored locally enables low latency and high access rate.

Each instance in the output sequence must declare a moment in time when to emit. It is carried out when this time matches the exact time coming from the timing interface. In addition to this, there is configuration data such as pulse length which need be managed. The logic surrounding this adds latency. To be able to emit at a high rate, each output must have a buffer.

The architecture seen in Fig. 3 is a simplification of this implementation only showing how the WRPC interfaces are connected to the custom logic. Since the custom core uses a Wishbone interface, the Etherbone core was ideal to package the communication. The Etherbone core wraps wishbone operations in Ethernet frames.

RESULT

The implemented proof-of-concept TS provides a set of fundamental services, e.g., delivering synchronized pulses at multiple TRs.

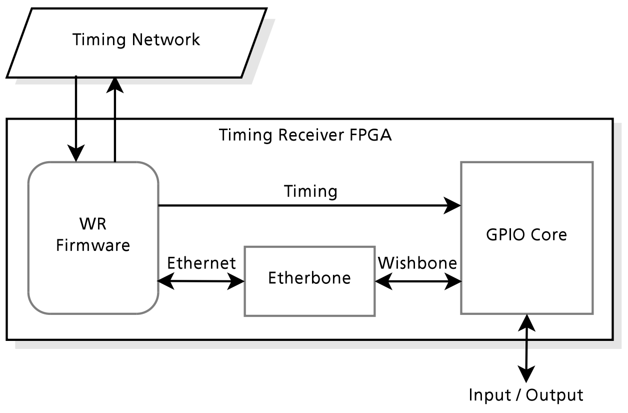

Figure 3: This is a simplification of the designed timing receiver architecture. It shows how the White Rabbit firmware interfaces are connected and what kind of data is being exchanged. Everything except the WR firmware is considered to be custom.

The result of the comparison is that, even though the interfaces are widely different, similar services can be implemented with both SLs. However, there will be conceptual architectural differences, such as, the impact from scheduling will require output queues in a WR based TS.

Strict requirements for low latency and deterministic loop-back latency can exclude WR as a viable option.

CONCLUSION

From the control system developer point of view there is no difference between the two synchronization layers. But, because of completely different approach to synchronization, there exists important implications to key characteristics which must be taken into account when choosing platform. Similar functionality is possible to implement with both.

To the timing system developer, on the other hand, the different interfaces will result in completely different timing system architectures. The White Rabbit architecture will revolve around absolute time scheduling. This is, nevertheless, already very important when designing a timing system for a particle accelerator. Event based timing system implementations will instead revolve around downstreaming precisely timed event triggers.

ACKNOWLEDGEMENT

This article was made possible thanks to collaboration with Cosylab d. d., Control Systems Laboratory, Ljubljana, Slovenia. Cosylab assisted with experience in timing systems and provided a White Rabbit platform as well as hardware from Micro-Research Finland and Shanghai Institute of Applied Physics.

REFERENCES

[1] P. Moreira, J. Serrano, T. Włostowski, P. Loschmidt and G. Gaderer, "White Rabbit: Sub-Nanosecond Timing Distribution over Ethernet", *Proceedings of ISPCS2009, Bresia, Italy*, 2009.

[2] R. Tavcar, R. Stefanic, Z. Kroflic, J. Dedic and J. Gutleber, "Timing system for Medaustron based on off-the-shelf MRF transport layer", *Proceedings of IPAC2011, San Sebastián, Spain*, 2011.

[3] M. Liu, C. X. Yin, L. Y. Zhao, D. K. Liu, "Development Status of Sinap Timing System", *Proceedings of IPAC2013, Shanghai, China*, 2013.

[4] D. Beck, R. Bar, M. Kreider, C. Prados, S. Rauch, W. Terpstra and M. Zweig, "The new White Rabbit based timing system for the FAIR facility", *Proceedings of PCaPAC2012, Kolkata, India*, 2012.

[5] J. Pietarinen, "MRF Timing System", *Timing Workshop CERN*, February 2008.

[6] J. Pietarinen, "Timing System with Two-Way Signalling", *EPICS Meeting Padova*, October 2008.

[7] M. Kreider, "The FAIR Timing master: A discussion of performance requirements and architectures for a high-precision timing system", *Proceedings of ICALEPCS2011, Grenoble, France*, 2011.

[8] M. Lipinski, T. Włostowski, J. Serrano, P. Alvarez, J. D. G. Cobas, A. Rubini and P. Moreira, "Performance results of the first White Rabbit installation for CNGS time transfer", *Proceedings of ISPCS2012, San Francisco, USA*, 2012.

[9] "Open Hardware Repository", 2009. [Online]. Available: http://www.ohwr.org.

ISBN 978-3-95450-139-7