Reconfigurable Timing Unit for NSTX-U*

S. DeLuca, P. Sichta, G. Tchilinguirian
Princeton Plasma Physics Laboratory, Princeton, NJ USA
sdeluca@pppl.gov

Abstract - Due to the longer pulse length of the NSTX-U device, a significant portion of the CAMAC equipment used for data acquisition and control will be supplanted with instrumentation and control (I&C) technologies such as CompactPCI and PXI Express. The Timing and Synchronization System (T&S) is used to synchronize data acquisition and control throughout the experimental complex. A non–CAMAC solution is needed to support the new and upgraded I&C systems for NSTX–U. An instrumentation-bus-agnostic, FPGA–based T&S system is under development. The primary component is the Reconfigurable Timing Unit (RTU). The RTU has been designed to be fully compatible with NTSX's original T&S. The RTU is the 3rd NSTX-designed FPGA-based timing device. Unlike its predecessors, the RTU will rely on commercial products: LabVIEW software and sbRIO hardware, both from National Instruments. Our experience in product development, integration with the NSTX-U computing and control environment, and timing performance will be presented.

Keywords-data acquisition; control; timing

I. INTRODUCTION

NSTX-U is the largest experimental project at the Princeton Plasma Physics Laboratory (PPPL). An essential facility for NSTX-U research is the central timing and synchronization system (T&S) [1] which is used to synchronize control and data acquisition systems throughout the experimental complex. The present timing system has been constructed from CAMAC [2] modules, many of which have been in service for 25 years. A CAMAC-only solution necessitated that a CAMAC system be installed nearby newer technologies such as PC-based instruments, compactPCI or PXI equipment, or Programmable Logic Controllers.

An instrumentation-bus-agnostic T&S system is being developed for NSTX-U; the primary component of this is the Reconfigurable Timing Unit (RTU). It is called reconfigurable because through software, the device is able to perform timing functions previously fulfilled by using several CAMAC modules. A core requirement is that the RTU be compatible with the existing system. It has been designed to provide timing functions, electrical characteristics, field wiring interface, and a user software perspective that is very similar to the original CAMAC-based T&S. The form, fit, and function of the RTU's predecessor, the Universal Networked Timer (UNT) [3], will be preserved. The RTU product development experience, to-date results, and future plans will be illustrated.

II. NSTX TIMING & SYNCHRONIZATION SYSTEM

The present NSTX T&S is described in [1]. The core system has been in use since 1981, supporting both TFTR [4] and NSTX [5]. The NSTX T&S architecture is shown in fig. 1.

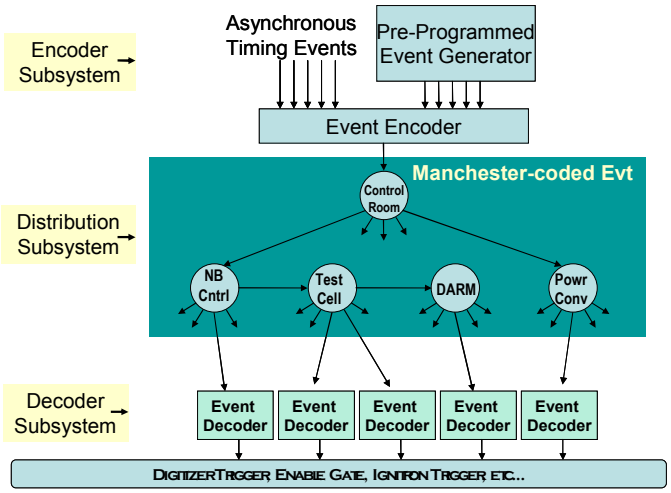

Figure 1. General Architecture of the NSTX Timing and Synchronization System. The RTU will first be used as an "Event Decoder".

The system provides synchronization of NSTX's many subsystems and diagnostics with the plasma 'shot cycle', using sixteen unique timing events. Most of these are pre-programmed but the system does support asynchronous events, as well. The events are encoded (1 MHz differential Manchester code), distributed, and decoded throughout the experimental complex. Once decoded, a number of CAMAC module types are used to provide specific timing functions, such a post-event delay, timed gates, or pulse trains. Even newer I&C technologies, such as a PXI system, would need this CAMAC infrastructure nearby in order to synchronize with the shot cycle.

The RTU is being designed as a 'drop-in' to replace the final two functions: event decoding and specific timing functions. This one device, through user configuration, system support software, and its FPGA, will interface any control system technology with the T&S. Unlike commercial-off-the-shelf (COTS) timing systems, the RTU is not bound to any specific instrumentation standard.

* Work performed under the auspices of the US Department of Energy by Princeton Plasma Physics Laboratory under Contract No. DE-AC02-09CH11466.

A. Review of Prior Work

Since 2003, NSTX has developed two FPGA–based T&S systems [3,6], and these have performed well in limited deployment. These projects only supported a subset of the NSTX timing functions. Progress towards a complete device was hindered because of the time and expense of software development, including: 1) FPGA code development, 2) FPGA code maintenance, and 3) interfacing to NSTX's software infrastructure: EPICS [7], MDSplus [8], and LabVIEW [9]. An excerpt from the UNT [3] mentions the product development challenge, *"A wide range of expertise has been required for this project. This included learning integrated development environments for the FPGA code and the MCU code; programming in C, and LabVIEW; EPICS and MDSplus device-level (network) programming; network and cyber-security administration; understanding Modbus protocols, interface circuit design; printed circuit board layout, fabrication, test, and technical writing."*

Unlike the earlier FPGA-based timing devices at NSTX, the RTU is built upon proprietary, commercial off–the–shelf (COTS) products from National Instruments (NI) such as Single–Board RIO hardware and LabVIEW software. The decision to favor a proprietary solution over the previously-desired open and multi–sourced standards approach will be discussed.

III. NEW TIMING UNIT DESIGN

With NSTX operations on hold during NSTX-U's construction period, this project need was able to be addressed again, with fresh insight from years of stalled-project hindsight and a decade of evolved-technology.

A. Greater Reliance on COTS

A 100% COTS solution would not have met the need to maintain compatibility with the existing T&S while also being able to not be tied to any one instrumentation form factor. New COTS and open-source software components were available to simplify the design, development, and full integration of a new timing unit. Two candidates were considered, 1) PC-104 (an open standard) and 2) LabVIEW-FPGA (proprietary). LabVIEW-FPGA was selected.

The primary factor in this decision was the availability of a recent FPGA hardware product from National Instruments, the Single Board Reconfigurable Input Output (sbRIO) model 9606. This small board and its associated software addressed all of the (former) UNT project's problems. All configuration, programming, compiling, downloading, and communication is done using LabVIEW

The sbRIO-9606, as its name implies, is a single board. It contains the Spartan-6 LX45 FPGA, on-board RAM, 96 inputs/outputs, and a real-time LabVIEW (LabVIEW-RT) running on an on-board CPU. The device has a network interface and single power supply connector. To interface with external equipment, a single high-density connector is provided. Mechanically interfacing this connector is well documented in the NI documentation. The sbRIO is able to automatically load its FPGA bitfile, LabVIEW-RT program, and begin running automatically on power-up.

B. Maintainability Risks Using COTS

Relying on a single vendor presents risk, but considering the difficulties on RTU predecessors, it was acceptable. National Instruments (NI) is an industry leader and has made a substantial investment in developing its FPGA and RIO products. ITER had decided to accept LabVIEW-developed FPGA RIO products [10], and NI had been considering long-term product support for ITER products. Other projects at PPPL are using LabVIEW-FPGA. The growing popularity of the FPGA in scientific research implies that the LabVIEW-FPGA and RIO products will be an enduring product line.

While there is confidence that the general product line and software tool chain will continue, the model 9606 is likely to be eclipsed with better FPGA technology within a few years. To protect the engineering investment with the RTU chassis, and support a different (future) sbRIO board, we have designed a very simple printed circuit board to interface the 9606 with the other components in the RTU chassis. Thus only this board will require a redesign to accommodate a new sbRIO.

C. New RTU features

The LabVIEW development tool-chain made it easy to introduce new capability. Two function-dependent front panel LED were added to aid the local operator about each timing channel's state. Additional electrically isolated digital inputs and outputs have been made available at the rear panel. The first order implementation for these will be simple, low-speed digital inputs and outputs, to be passed up the software stack to the high level applications. In addition, a number of clock/board/channel status bits were made available, since routing them up the software stack in the LabVIEW integrated environment is easy.

IV. NSTX SOFTWARE INTEGRATION

Since the RTU runs LabVIEW-RT, integration with NSTX's core software (EPICS and MDSplus) is routine; this has been done many times at NSTX. NI provides the EPICS interface software for LabVIEW-RT. MDSplus provides interfaces for LabVIEW, and these also work on LabVIEW-RT. Furthermore, recent advancements to fully integrate EPICS and MDSplus [11] will provide additional mechanisms to have the three main NSTX software applications communicate. Fig. 2 shows the RTU, and how it connects with the networked applications.

A. EPICS

EPICS is the software used to support integrated NSTX-U operations. Almost every subsystem at NSTX has been integrated with EPICS. Most engineering subsystem operators configure their (CAMAC) timing devices using an EPICS OPI (operator interface). This user interface need not change to support an RTU-based timing channel. The underlying EPICS records and database will need to be revised by an EPICS application programmer. The RTU's EPICS database-level

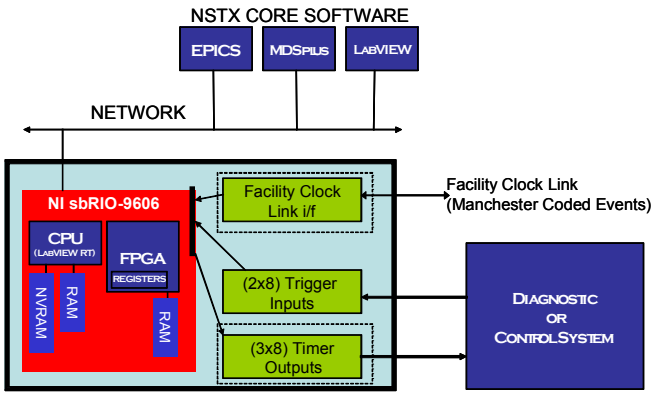

Figure 2. A block diagram of the RTU and integration with NSTX-U. This includes RTU inputs and outputs to the controlled equipment, the Facility Clock Link, and the network. The RTU will be capable of communication with NSTX's three major software applications, EPICS, MDSplus, and LabVIEW.

support has not yet been designed. It could use a series of standard record types as is done now with CAMAC, or, a new EPICS record type or device support module may be developed.

B. *MDSplus*

Most diagnostics use MDSplus for configuring their data acquisition and timing devices. As with EPICS, the user interface for configuring their RTU-based timing channel will remain very similar to the present GUI that is designed for CAMAC T&S modules. While the MDSplus GUI for the RTU has not yet been designed, the user interface will not change very much. Many of the default Motif based device configuration screens, accessed primarily via Traverser, will be reused. At PPPL these existing interfaces are either included with the MDSplus distribution or were developed in-house for internal use. These existing GUIs will be utilized as templates for configuring the RTU operational modes they will emulate. This will not only preserve the user experience but will necessitate only a back-end development effort to support this new flexible hardware. This should lead to seamless integration and a familiar user interface for experienced users.

V. FPGA Programming Experience

The main problem with previous incarnations of the RTU was the specialized skills needed to produce the FPGA HDL code that would emulate the CAMAC timing functions. The schematic capture feature of the Xilinx ISE webPACK [12] was used. This turned out to be very time consuming to use and maintain. The LabVIEW-FPGA programming environment made implementation of the timing functions relatively easy.

There were two FPGA developers on the RTU project. To prepare for the RTU project, one engineer completed NI training for both LabVIEW-FPGA and LabVIEW-RT. This enabled a quick setup the sbRIO-9606 hardware, and preparation of a simple FPGA program and a LabVIEW program for the real-time portion of the sbRIO. The LabVIEW-RT program included LabVIEW variables that could easily be linked with EPICS, MDSplus, or another LabVIEW system. The second developer was an experienced LabVIEW programmer who also had deep knowledge of the existing CAMAC-based timing functions. This engineer was able to easily learn from the first how to develop FPGA programs, compile and download to the target sbRIO-9606. The development team's combination of general LabVIEW programming experience, knowledge of the timing functions, and specialized LabVIEW (RT and FPGA) training was successful in rapidly developing the RTU software.

A. *Novice Surprises*

The prospect of being able to implement the desired logic and timing functions into the FPGA with the ease of LabVIEW programming was exciting, especially since developing FPGA programs in the past had been highly labor intensive and it was hard to develop a reliable project schedule. While getting a basic program up and running with LabVIEW-FPGA was indeed quick and easy, getting our specific RTU application implemented proved to be more difficult than the initial successes predicted. A few examples of the unexpected are presented below:

- Integrating FPGA with networked applications: The "Main" program running on the LabVIEW Real Time portion of the sbRIO board could not directly access the FPGA memory or FPGA programs' global variables. This makes FPGA subVI to main communication a multi-step, complicated programming endeavor.
- FPGA memory: The use of FPGA memory has restrictions that are seemingly overly-restrictive, in regards to arbitration. This forced us to devise and evaluate numerous schemes to overcome this limitation. NI's "Help" information about the topic was not very useful.
- FPGA space usage: The UNT, the RTU's predecessor, used a circa-2005 FPGA, which was much smaller than the one on the sbRIO-9606. Yet, with the Xilinx ISE webPACK, we were able to put more timing emulations onto the UNT's FPGA than we could using the much larger FPGA using LabVIEW-FPGA. NI offered a number of suggestions for making more effective use of FPGA space. Some of these have been tried and produced little change.
- Compile time: We found that using NI's free remote compiler software, (we used the RHEL5 Linux version) reduced compile time by about 50%. Even with the Linux remote compile worker, our 60% space-consuming FPGA program took over 20 minutes to compile.

While these problems contributed towards the RTU FPGA software development taking longer than anticipated, the overall experience of implementing and integrating an FPGA-based timing product with NI products has been overwhelmingly positive.

VI. HARDWARE DESIGN

The RTU hardware design is also being done at PPPL, and is using electronic design automation design (EDA) packages. The printed circuit board design tool is by Mentor Graphics, a product called PADS [13]. This package contains schematic drawing, parts libraries, and automated printed circuit board layout facilities. It produces the 'gerber' files and parts lists for manufacturing and assembling the printed boards. Two PPPL-designed printed boards are being developed.

The 1U rack mountable RTU chassis design (depicted in fig. 3) is closely modeled after the prior timing device, the UNT. It supports 8 timing channels using field-removable header connectors. Both twisted pair copper and fiber optic facility clock link inputs are available. To design and fabricate the RTU chassis, under strong considerations is a free EDA tool [14]. This package empowers the designer to specify the dimensions of the enclosure, the front and rear openings, materials, finish, labeling, and color.

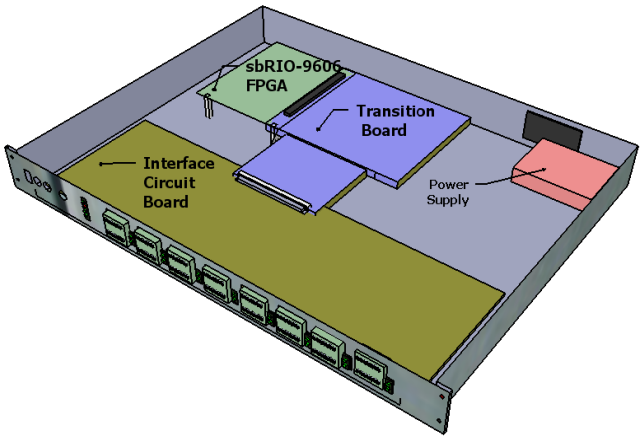

Figure 3. A representation of the 1U RTU chassis and its internal components.

As explained earlier, a *transition board* will be included to accommodate a future variation of the sbRIO-9606 board, while allowing the *interface circuit board* to remain unchanged.

VII. PRESENT STATUS & FUTURE WORK

The RTU FPGA program has been written for the initial four CAMAC module emulations and deployed onto the sbRIO hardware. Timing measurements have demonstrated that the RTU outperforms CAMAC. The largest improvement is that the RTU does not have the timing uncertainties that the CAMAC modules can exhibit.

The LabVIEW software which is running on the sbRIO-9606 has yet to be integrated with the NSTX-U networked applications. Using existing methods, this is now routine. However, the design may incorporate new MDSplus and EPICS integration mechanisms to provide a better solution. The final piece of software to be designed is for supporting access control. This will allow only authorized configuration of the RTU, which is necessary in a multi-user environment. EPICS, MDSplus, and LabVIEW all have features to support this.

One of two PPPL-designed printed circuit board designs is complete. The assembly details for the chassis are almost complete. The electrical isolation of the RTU's inputs and outputs, and the easy to use COTS software, will allow the RTU to be used for other interlock, timing, and data acquisition applications. For example, the RTU can be used to replace the CAMAC-based event encoder subsystem (fig. 1).

VIII. CONCLUSION

An event-system and electrically compatible, bus-agnostic timing and synchronization device will be completed shortly for NSTX-U. The RTU will provide the first of three elements to produce a non-CAMAC T&S for NSTX-U. The RTU will be fully integrated with NSTX-U core software: EPICS, MDSplus, and LabVIEW and the user interface will remain consistent with what users are accustomed to.

LabVIEW makes programming an FPGA a simple task. Using the two unified products, LabVIEW-FPGA software and sbRIO hardware, the RTU development and NSTX-U integration has been far more effective for PPPL than the traditional FPGA product development cycle that was used on prior FPGA-based timer projects.

REFERENCES

[1] Montague, J, and Sichta, P, "Synchronization of timing systems on TFTR." Fusion Engineering, 1991. 14th IEEE/NPSS Symposium on 1991: 810-813.

[2] Instrumentation, Modular. "Digital Interface System (CAMAC)." IEEE STD (1975): 583-1975.

[3] Sichta, P et al. "Development of a Universal Networked Timer at NSTX." Fusion Engineering 2005, Twenty-First IEEE/NPS Symposium on Sep. 2005: 1-4.

[4] Rauch, WA. "TFTR CAMAC data-acquisition system." Review of Scientific Instruments 57.8 (1986): 1898-1900.

[5] P. Sichta, J. Dong, G. Oliaro, P. Roney, "Overview of the NSTX Control System,", 8th Conference on Accelerator and Large Experimental Physics Control Systems, San Jose, USA (2001).

[6] Sichta, P et al. "Developments to supplant CAMAC with industry standard technology at NSTX." Fusion engineering and design 71.1 (2004): 129-133.

[7] "EPICS - Experimental Physics and Industrial Control System." 2012. 28 May. 2013 <http://www.aps.anl.gov/epics/>.

[8] "MDSplus." 2013. 28 May. 2013 <http://www.mdsplus.org/>

[9] "National Instruments: Test, Measurement, and Embedded Systems." 2013. 28 May. 2013 <http://www.ni.com/>

[10] Kim Changseung, NI FPGA EPCIS device support, ITER IDM document ITER_D_466LGB v1.0 , Feb 28, 2011.

[11] G. Manduchi, T.W. Fredian, J.A. Stillerman, MDSplus evolution continues, Fusion Engineering and Design, Volume 87, Issue 12, December 2012, Pages 2095-2099

[12] "ISE WebPACK Design Software - Xilinx." 2011. 8 Jun. 2013 <http://www.xilinx.com/products/design-tools/ise-design-suite/ise-webpack.htm>

[13] "PADS PCB Tools - Mentor Graphics." 2009. 8 Jun. 2013 <http://www.mentor.com/products/pcb-system-design/design-flows/pads/>

[14] "Front Panel Express: Front Panel Design Software and CAD ..." 2003. 8 Jun. 2013 <http://www.frontpanelexpress.com/>