ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV044

LESSONS LEARNED MOVING FROM PHARLAP TO LINUX RT

Cédric Charrondiere, Odd Oyvind Andreassen, Diego Sierra Maillo Martinez, Joseph Tagg, Thomas Zilliox, CERN, Geneva, Switzerland

Abstract

The start of the Advanced Proton Driven Plasma Wakefield Acceleration Experiment (AWAKE) [1] facility at CERN in 2016 came with the need for a continuous image acquisition system. The international scientific collaboration responsible for this project requested low and high resolution acquisition at a capture rate of 10Hz and 1 Hz respectively. To match these requirements, GigE digital cameras were connected to a PXI system running PharLap, a real-time operating system, using dual port 1Gbps network cards. With new requirements for a faster acquisition with higher resolution, it was decided to add 10Gbps network cards and a Network Attached Storage (NAS) directly connected to the PXI system to avoid saturating the network. There was also a request to acquire high-resolution images on several cameras during a limited duration, typically 30 seconds, in a burst acquisition mode. To comply with these new requirements PharLap had to be abandoned and replaced with NI Linux RT.

This paper describes the limitation of the PharLap system, and the lessons learned during the transition to NI Linux RT. We will show the improvement of CPU stability and data throughput reached.

INTRODUCTION

A plasma wakefield is a type of wave generated by particles travelling through a plasma. By harnessing these wakefields, accelerating gradients hundreds of times higher than those produced in current radiofrequency cavities can be achieved [2], allowing for more compact accelerators. AWAKE is a proof of principle experiment that aims to demonstrate this in a scalable way, sending proton beams through plasma cells to generate these fields, which subsequently accelerate electrons to high energy over a short distance.

One important observable is the shape and position of the proton beam halo along the beamline, that must be acquired in real-time. To handle the 10Hz image acquisition on 10 cameras simultaneously it was decided to use a PXI running PharLap.

Due to the lack of supported drivers, issues with timing and performance, the system was later upgraded to NI Linux RT to benefit from its flexibility.

Motivation

The cameras are positioned along the beam line for several purposes (Fig. 1). The cameras that are on the virtual laser diagnostic line are used to measure the characteristics of the laser used to initiate the plasma. Other cameras are used to image the path of the laser and proton beams to align them in the plasma cell. Finally, some cameras are positioned to visualize the low energy electron beam at different points of its path before it is accelerated by the proton generated wakefield in the plasma cell.

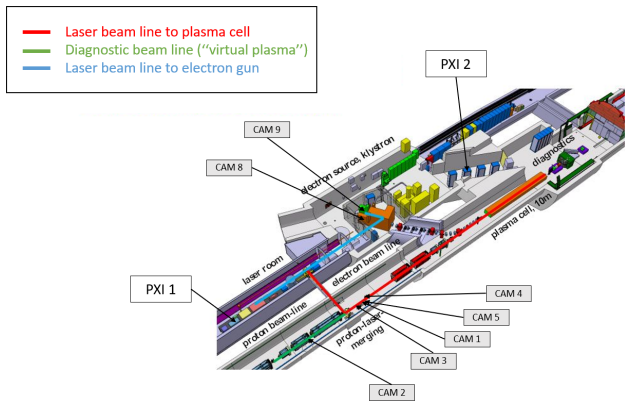

Figure 1: Camera locations in the Awake Experimental Area.

HARDWARE TOPOLOGY

The AWAKE camera acquisition system acquires images from ethernet based, digital GigE cameras. The system publishes resampled images (resampling factor 5x5) at 10 Hz and publishes the full-size images on an SPS extraction event (once per AWAKE cycle). The basic system topology is shown below (Fig. 2)

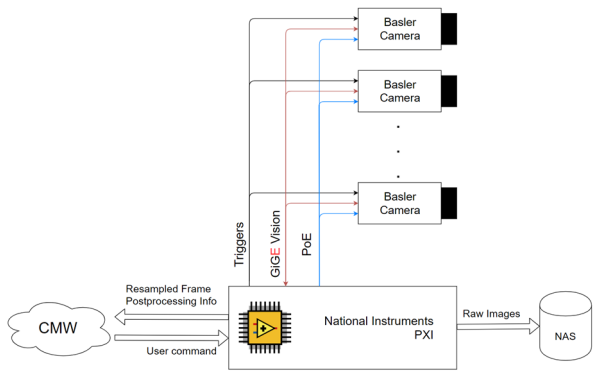

Figure 2: System topology.

Each camera is connected to the PXI system and powered by a PoE (Power over Ethernet) module. The triggering is handled through the FPGA as shown in the hardware architecture (Fig. 3)

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV044

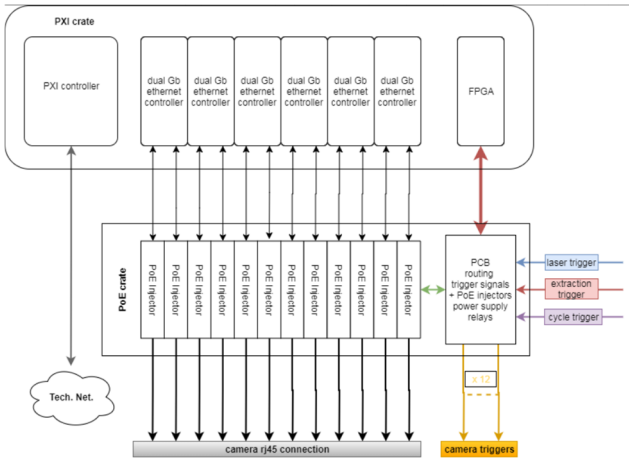

Figure 3: Hardware architecture.

Performance

Two OSes support the new 10Gbps card: NI Linux RT and Windows. We compared the CPU usage on both, using CPUs with relatively close performances, a NI PXIe-8840 for Windows and a NI PXIe-8861 for NI Linux RT. The CPU usage in Windows was around 50% at idle due to th Windows and CERN services running in the background. In production, with cameras acquiring, the PXI was oscillating between 97-99%, resulting in multiple frame losses. With NI Linux RT, the usage of the CPU is around 0% at idle and between 50 to 70%, depending on the number of cameras plugged into the PXI. This increase in available CPU power drastically reduced the frame losses and increased the stability of the system.

Network Attached Storage (NAS)

Publishing the raw images on the CERN network is not possible as its bandwidth is 1 Gbps and the current setup would require 4 Gbps. To allow the AWAKE team to acquire raw images at that rate in the specified amount of time (referred to as burst acquisition) a NAS is directly linked to the PXI's 10 Gbps card. The NAS used is a HP PROLIANT DL380 GEN9 running Linux.

OPERATING SYSTEM

PharLap

The main advantage of PharLap is that it is a lightweight RTOS (Real Time Operating System) with few software layers, which makes it robust and stable while maintaining determinism. This advantage is also a weak point in terms of debugging because when a RT application crashes, the whole OS stops responding, and a reboot of the device is required. In addition, because it does not implement memory virtualization, very little can be fixed at runtime without rebooting, requiring careful memory management. National Instruments will stop its support of PharLap in 2025 [3] with some new modules already not supported on this platform, forcing the community to stop using it for new projects or major upgrades.

In the case of AWAKE, a PXIe-8238 10Gbps ethernet card was required to log data to a NAS at high speed. This card is not supported by PharLap, which motivated us to change the OS.

Another issue is the timing synchronization daemon, SNTP 1.2, based on SNTP (Simple Network Time Protocol), that does not work properly on PharLap. When SNTP is synched, we measure a continuous oscillation of about +/- 0.02 seconds around the correct time. After a few hours (2 to 20+), we measure an abrupt offset of about 4 minutes, followed by oscillations of +/− 0.2 seconds.

Linux RT

As the CERN accelerator complex infrastructure is UNIX based, having an acquisition system based on a NI Linux RT distribution makes its integration into existing services and databases easier. Contrary to PharLap, NI Linux RT (based on the OpenEmbedded platform) ships with built in tools for compiling, debugging, administrating and monitoring our device by default. The behaviour of the built-in NTP timing service works, and debugging via ssh (secure shell) is more convenient since this can be done while the system is running.

Remote access and troubleshooting are relevant to most of our projects as they often are installed in inaccessible areas. As an added bonus, other languages such as Python are also available, opening new possibilities.

In the AWAKE application, we were wary of potential performance issues due to a saturation of the network bandwidth but were not able to measure it. Thanks to the default monitoring tools available in the NI Linux RT distribution, we were finally able to quantify it and fix one of the major issues of the system, which was caused by too many clients connected to the system simultaneously.

In addition to the different monitoring tools, we also benefit from all the different native logging systems. The ethernet logger allowed us to quantify the behaviour of each camera's communication, especially when they were exposed to too much radiation.

The main drawback of NI Linux RT is its backward compatibility; some drivers are not, and may never be, supported.

INTEGRATION

Timing

The different timestamps from the camera's driver, such as the frame arrival timestamp, are based directly on the PXI clock. Because PharLap does not have a trustworthy built-in NTP daemon, there is a permanent clock drift, which requires every timestamp to be corrected. NI Linux RT's NTP client on the other hand works well, which helps with camera timestamping.

With regards to the PXI time, Linux RT is more user-friendly than PharLap. While it is possible to correct in PharLap, Linux RT freed us from this constraint.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV044

Triggering

To sync the image acquisition with the AWAKE laser and other SPS events, a trigger is generated by the AWAKE timing system. It is wired to a CERN central timing receiver card (CTRp) [4] [5], an FPGA-based custom timing card manufactured at CERN. This card allows us to subscribe to every timing event of a specific accelerator, in our case the SPS, and to forward a specific event trigger to all the cameras.

On the early version of the NI Linux RT (up to version 20.6) some VISA functions were not working properly with the CTRp card, such as the subscribe to PXI event (Enable_PXI_VISA_event.vi). We should therefore not take for granted that custom PXI cards that work on PharLap will work out of the box on NI Linux RT.

Vendor Libraries

In general, when using hardware, software and firmware from NI, compatibility and maintenance is not an issue. However, as the years have passed, and PharLap has gone from being a mature to an obsolete platform, even vendor specific compatible drivers have become difficult to find. For the NI Linux RT based systems, the opposite has been the case. Here many of the libraries and drivers have either been ported to the platform in the last few years, or are in the process of being implemented. This poses a challenge for the end user when maintaining the system, leaving you in a situation where some hardware is supported in one environment and not the other. Many of the major hardware families have equivalent replacements available, but they are often not "plug-in" replacements, and might require different signal converters and cables, running at different voltages. This has led to some caution when upgrading systems procured in the 2015 to 2021 timeframe, carefully having to anticipate whether the hardware would have compatible libraries both on PharLap and Linux RT when designing new control system.

In the case of the AWAKE camera acquisition system, the choice was driven by the necessity of higher throughput to cover additional cameras, and a higher level of stability in terms of time drift. The "Lessons learned" moving to NI Linux RT, has been that not everything is a drop-in replacement from the vendor side, and sometimes small hardware changes or features introduced in newer versions, can impact the whole system and should be carefully studied.

3rd Party Libraries

Most libraries on NI based systems are installed using a specialized tool called MAX (Measurement and Automation Explorer). This is the case both for PharLap and NI Linux RT, however the underlying technology for the two platforms is significantly different.

For PharLap, the installation is done simply by formatting the target, priming it with a base operating system and installing drivers by copying them to the right location on disk. There are no convenient ways to keep track of what is installed on which target. To upgrade a driver you need to install the appropriate driver bundle on your development system, which is often versioned differently from what you see on the target. This often leads to confusion and maintenance overhead. To keep track of the libraries installed on the target, it is necessary to document every action taken during the course of development.

For NI Linux RT, the environment is based on UNIX, and in addition to the standard toolchain distributed via a dedicated package manager (OPKG for NI Linux RT), NI have added a management tool called SaltStack, that takes care of monitoring what is installed and if there are upgrades available. In addition, multi-target support has been added, so one can now distribute upgrades to multiple targets in parallel.

This means that system replication is easier for smaller systems on PharLap, but large-scale system administration is by far simpler on NI Linux RT. For third party libraries, this is a huge advantage, allowing custom developed code to be managed side by side with the vendor's own drivers.

Network Booting

One of the main requirements for systems installed in and around the accelerators at CERN, is the ability to remotely recover and re-install the system in case of failure. This was typically done via PXE (Pre-eXecution Environment), however in the later years UEFI (Unified Extensible Firmware Interface) has become more common.

For PharLap, NI uses traditional PXE boot while NI Linux RT NI uses UEFI [6]. Both technologies can be deployed at CERN, but UEFI has more modern capabilities and integrates better into the current infrastructure.

Compilation Process, Debugging and Error Management

When developing 3rd party drivers and libraries such as the CERN middleware and timing libraries, the ability to test and validate the implementation is imperative.

In PharLap, because it is based on an older Windows NT kernel, all custom development requires the Visual Studio development environment, and a PharLap-compatible runtime.

To debug libraries, you first have to test the library under Windows and then rigorously test all functionality on PharLap. The problem is that PharLap does not have any proper debugging tools, and the error messages it produces are often obscure or too generic, forcing you to add extensive debug information as part of the implementation (such as printing information along the execution of the code).

On NI Linux RT, you have all the standard GCC and UNIX tool chains available, and debugging can easily be done in a comfortable way, saving both development and maintenance time.

Compilation Tools

As mentioned, PharLap relies on the Visual Studio toolchain to compile 3rd party software. Unfortunately, the development of a dedicated Visual Studio runtime for PharLap was stopped at version 2010, and Microsoft, the

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV044

owner of Visual Studio stopped its support for the runtime in its development environment in 2020 (official EOL was 2015 but extended EOL was pushed back top 2020). Furthermore, Visual Studio 2010 is not compatible with Windows 10 and beyond, making compilation of source code for PharLap based systems even more complicated [7].

For NI Linux RT, the standard gcc or g++ toolchain can be used, even on the target itself, and both development and validation is straight forward. If in addition you leverage automated builds through continuous integration and modern DevOps solutions (such as those provided by GitLab), any upstream change to a dependent source can be caught, compiled, and tested immediately as it is released. This has proven to be a huge time saver in terms of consolidation and maintenance.

CONCLUSION

Switching from PharLap to NI Linux RT brought performance improvements, reducing both the general resource and CPU usage. The UNIX environment simplified library and system administration. It also allowed features that were missing from PharLap such as the 10 Gbps network drivers and the NAS.

The next challenges will be to increase the image post-processing speed in order to increase the resolution of the cameras. As a result of the operating system change, we can now benefit from newer FPGA cards dedicated to vision analysis, allowing us to offload some calculations from the CPU.

REFERENCES

[1] https://home.cern/science/accelerators/awake

[2] https://home.cern/science/engineering/accelerating-radiofrequency-cavities

[3] https://www.ni.com/content/dam/web/pdfs/phar-lap-rt-eol-roadmap.pdf

[4] J. Serrano, P. Álvarez, D. Domínguez, and J. Lewis, "Nanosecond Level UTC Timing Generation and Stamping in CERN's LHC", in Proc. 9th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'03), Gyeongju, Korea, Oct. 2003, paper MP533, pp. 119-121.

[5] O.O.Andreassen *et al.*, "Evaluation of timing synchronization techniques on NI CompactRIO platforms", ICALEPCS 2019, New York, Oct. 2019, paper WEPHA27.

[6] O.O.Andreassen *et al.*, "Integrating COTS Equipment in the CERN Accelerator Domain", presented at the 17th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'19), New York, NY, USA, Oct. 2019, paper WEPHA026.

[7] https://docs.microsoft.com/en-us/lifecycle/products/visual-studio-2010