**NuPRISM August 2020 Xilinx Firmware Status Report**

**Author(s):** R. Payne

**Release:** 1.0

**Release Date:** 21/08/2020

**Document Type:** Report

**Keywords:** IWCD, NuPRISM, Firmware, Xilinx, Altera, UltraScale+, Zynq, MAX10, Main Board, Electronics, mPMT, Co-op

**History of Changes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Release No.** | **Date** | **Description** | **Authour(s)** |
| 1.0 | 21/08/2020 | Initial Release | R. Payne |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

This document is intended to act as an onboarding device for the September 2020 firmware developer working on the NuPRISM mPMT mainboard. This document will introduce you to the experiment and provide an overview of the mainboard, which is the hardware platform you will be working on. Next, it will describe the project status from the point of view of the TRIUMF firmware developer and try to recommend a path moving forwards. Additionally, there will be a section referencing and describing resources that may help in the future.

If you have any further questions, the edev team is an excellent resource. You can also contact me directly by email at [rypayne@hotmail.ca](mailto:rypayne@hotmail.ca).

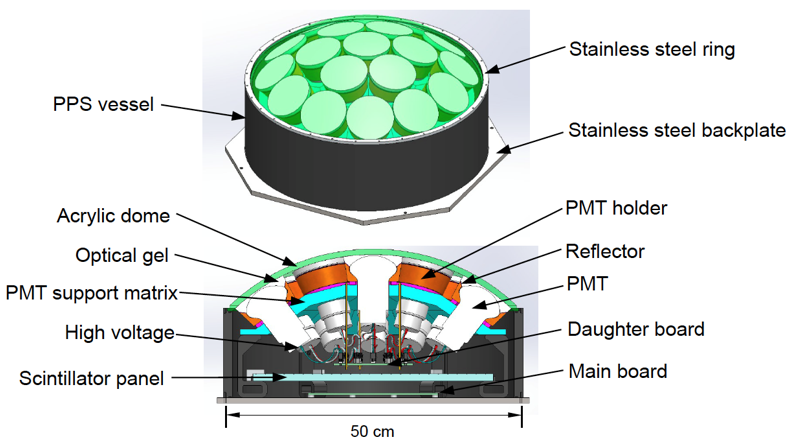
# Experiment

The IWCD experiment will place a water Cherenkov detector in a deep vertical pit in Japan. Approximately 500 multi-photomultiplier tubes (mPMT) make up the Cherenkov detectors photo-sensors. The experiment setup is shown in figure 1-1. Each mPMT consists of 20 photomultiplier tubes connected to the main board electronics. The mPMT is shown in figure 1-2. A mPMT concentrator card is responsible for providing higher level control and monitoring of the mPMTs. The experiment will be observing controlled events generated by a neutrino beam and a LED-based calibration source. It will also observe more random events from cosmic-ray muons.

A picture containing table

Description automatically generated

**Figure 1-1**: IWCD experiment setup



**Figure 1-2**: Diagram of the mPMT

# Mainboard Electronics

A high-level hardware block diagram is shown in figure 1-3. The SoC deserializes and processes data from 5 quad-channel ADCs. The ADC clock is generated by a clock-cleaner which can derive its clock from a crystal oscillator, SMA input, RJ45 input, or POE input. For connectivity, the SoC provides connection to RJ45, USB, Ethernet, and SFP. For system monitoring, the SoC is connected by I2C to several temperature, humidity, pressure, and voltage sensors. The MAX10 provides the SoC with UART connection to each PMT base, and has additional connection to the I2C bus, clock cleaner, and system clocks.

A screenshot of a cell phone

Description automatically generated

**Figure 1-3**: High-level hardware block diagram for mPMT main board

The clock cleaner takes its inputs from the SMA clock, RJ45 clock, POE clock, and a reference oscillator. The SMA clock and RJ45 clock are muxed and selected by the MAX10. On reset/startup the clock cleaner will use the oscillator as a reference. If either of the other two inputs are detected as valid, it will switch its reference to the valid clock. If both inputs are valid, a user-settable priority bit is referenced. While the input and output clocks are locked, device GPIO is activated and can be monitored. The generated clock is distributed to the 5 ADCs and used as a sample clock. The clock is also distributed to the MAX10 where it can be monitored or used for other functionality.

The main processor currently in place is the Xilinx Zynq Ultrascale+ EG MPSoC. A block diagram depicting its internals is shown in figure 1-4. The MPSoC consists of a processing system (PS) and programmable logic (PL). The PL is the FPGA portion of the MPSoC and contains RAM blocks, DSP blocks, and high-speed connectivity blocks. The PS is a highly integrated system. Most notable it contains it a quad-core 64-bit application processing unit (APU), and dual-core 32-bit real-time processing unit (RPU). To manage the power consumption of this device, the platform management unit (PMU) can be used to suspend, wakeup, and power-down different components of the entire architecture. There are several ways to manage a wakeup of the APU. Some notable methods are by GPIO input or by request from the RPU.

A screenshot of a cell phone

Description automatically generated

**Figure 1-4**: Zynq Ultrascale+ EG MPSoC architecture diagram

For the incoming firmware developer, it is worth noting that the platform supports to hot-pluggable SoCs. The one you will be developing firmware on is the Enclustra XU1 which hosts Xilinx’s Zynq Ultrascale+ EG MPSoC. The other is the Enclustra SA2 which hosts Altera’s Cyclone V. Revision 0 of the mainboard was specifically designed for the SA2, but later versions will use the XU1.

# Project Status

The firmware portion of this project is undergoing 4 branches of development. The Altera branch, the Xilinx RTOS branch, the Xilinx Standalone branch, and the Xilinx Petalinux branch.

The Altera branch is being worked on by Dr. Yair Linn and is the most developed branch. Current testing of the integrated mPMT is being done at TRIUMF using this project.

The Xilinx RTOS branch is in early stages of development. This branch is being worked on by the Polish colleagues.

The Xilinx Standalone branch was developed by Ryan Payne. This branch was used to test board level function and run the deserializer HDL. While this branch could be used to quickly test low level function and run altered deserializer HDL, it should not undergo major development. This project and a guide to using it can be found here:

<https://edev-group.triumf.ca/fw/exp/nuprism/xu1/rev0>

The Xilinx Petalinux branch is the branch that you, the incoming firmware developer, will likely be working on. Lucky (or not so lucky) for you, this branch is in very early stages of development. The amount of code which you need to understand to begin working should not be too overwhelming. This project essentially just runs a demo of MIDAS, a data acquisition software developed at TRIUMF. This project and a guide to using it can be found here:

<https://edev-group.triumf.ca/fw/exp/nuprism/xu1/rev0-2020.1-petalinux>

A high-level task list to focus on includes the following:

* Add DMA to the HDL
* Integrate DMA drivers to Petalinux
* Add the deserializer to the project and connect to the DMA HDL
* Add SPI control for ADCs
* Add I2C control for the clock cleaner
* Add I2C control for sensors on the mainboard
* Add UART control for PMT bases
* Add MIDAS control of ADC, clock cleaner, sensor
* Add MIDAS readout of ADC data, sensor data

The route I suggest you take is the following:

1. Run through the tutorial in later section to setup your board, Petalinux environment, run the feature limited MIDAS demo. This will help you get familiarized with Vivado, Petalinux, and MIDAS
   1. It may be helpful to also run the standalone project and see the ADCs producing data. This will require you to download two versions of Vivado.
2. Run through the project guide foundhere:

<https://edev-group.triumf.ca/masoudm/darkside-data-processor-project>

This project was developed by Masoud Mokhtari, another co-op student, for an experiment called Darkside. I recommend you use this project to better understand DMA and MIDAS.

1. Add DMA to your project and stream 12 bit “dummy” from the FPGA to the processing system, and finally over ethernet to your MIDAS host. I highly recommend that you integrate Masoud’s DMA drivers in order to do this.
2. Write code to talk to the ADCs and Clock Cleaner over SPI and I2C and control the current chip select system.
3. Once you have working DMA and can control the ADCs over SPI, connect the deserializer to the DMA block, and stream ADC readings over ethernet into your MIDAS host.
4. Consult with the team as to what the next steps are.

As a note:

The Gitlab repo has been where I have been pushing code that is not relevant to our Polish colleagues or the physicist. For the most part this code is developed as a learning exercise for me.

All code relevant that others may need to run should be placed on the NuPRISM GitHub located here:

<https://github.com/nuPRISM>

The XU1\_rev0 repo holds the 2018.3 standalone project.

The XU1\_petalinux\_rev0 repo holds the 2020.1 Petalinux project

The MAX10\_rev0 project holds the code which runs on the MAX10

For access to this code please contact Thomas Lidner.

# Resources

**Mainboard Electronics**

Given Gitlab access, the schematics for the mainboard can be found here:

<https://edev-group.triumf.ca/hw/exp/nuprism/nuprism-multi-channel-pmt-acquisition/rev00/-/tree/master/Altium/Project%20Outputs/2_Schematic%20Prints>

Additional schematics and documentation for the Enclustra module can be found here:

<https://download.enclustra.com/>

**Deserializer and Clocking Scheme**

The ADCs use DDR 2-wire serialization of 12 bit sampled at 125Msps. Special FPGA fabric is used to deserialize this data. The current deserializer HDL is based off another project done by the Stanford Linear Accelerator (SLAC). That project can be found here:

<https://github.com/slaclab/amc-carrier-core/blob/master/AppHardware/RtmRfInterlock/core/Ad9229Core.vhd>

The deserializer components inside of the FPGA logic take a fair bit of reading through documentation to understand. To summarize what I learnt in my research, I have written a report which can be found with this package under:

\Supporting\_Documents\XU1\_NuPRISM\_Derserializer\_and\_Clocking\_v1\_0

I highly recommend reading this document and before attempting to make any changes to the current deserializer.

**DMA**

The addition of DMA to this project may take up a decent chunk of your time. I highly recommend understanding and running Masoud’s project before making and attempt to integrate the project into you own.

The project can be found here:

<https://edev-group.triumf.ca/masoudm/darkside-data-processor-project/-/tree/master/>

A 2020.1 version of this project can be found here:

<https://edev-group.triumf.ca/fw/exp/nuprism/xu1/darkside-2020.1>

**MIDAS**

MIDAS is a front-end data acquisition software which will be used at TRIUMF and potentially in Poland to test the mPMT. The wiki for it can be found here:

<https://midas.triumf.ca/MidasWiki/index.php/Main_Page>

The wiki also contains a setup article, but I recommend you follow my guide to set it up as you do not need to access all the features present in the software. You can find that guide with the Petalinux project here:

<https://edev-group.triumf.ca/fw/exp/nuprism/xu1/rev0-2020.1-petalinux>

**Vivado and Petalinux**

As an intro to Vivado and Petalinux, Mohammad Sadri’s video series on ZYNQ and Petalinux are an excellent resource. I highly recommend watching them all to understand how to use to Xilinx tools. The videos can be found here:

<https://www.youtube.com/user/mamsadegh2/videos>

The Petalinux 2020.1 user guide will help you set up your Petalinux environment and set up a basic project:

<https://www.xilinx.com/support/documentation/sw_manuals/xilinx2020_1/ug1144-petalinux-tools-reference-guide.pdf>

To quickly set up an Ubuntu VM with Petalinux 2020.1, please follow the Petalinux quick guide found here:

<https://edev-group.triumf.ca/fw/exp/nuprism/xu1/rev0-2020.1-petalinux>

**SPI**

Here is a tutorial which gives you a nice intro to talking to SPI devices and setting up devices in the device tree

<https://www.beyond-circuits.com/wordpress/tutorial/tutorial26/>