Run Control Software for the Upgrade of the ATLAS Muon to Central Trigger Processor Interface (MUCTPI)

Ralf Spiwoks[1,*], *Aaron* Armbruster[1], *German* Carrillo-Montoya[1], *Magda* Chelstowska[1], *Patrick* Czodrowski[1], *Pier-Olivier* Deviveiros[1], *Till* Eifert[1], *Nick* Ellis[1], *Philippe* Farthouat[1], *Gorm* Galster[1,2], *Stefan* Haas[1], *Louis* Helary[1], *Predrag* Kuzmanovic[1], *Orestis* Lagkas Nikolos[1], *Antoine* Marzin[1], *Thilo* Pauly[1], *Vladimir* Ryjov[1], *Kristof* Schmieden[1], *Marcos* Silva Oliveira[1], *Joerg* Stelzer[1], *Paschalis* Vichoudis[1], *Thorsten* Wengler[1]

[1]CERN, Switzerland
[2]NBI Copenhagen, Denmark

Abstract. The Muon to Central Trigger Processor Interface (MUCTPI) of the ATLAS experiment at the Large Hadron Collider (LHC) at CERN is being upgraded for the next run of the LHC in order to use optical inputs and to provide full-precision information for muon candidates to the topological trigger processor (L1TOPO) of the Level-1 trigger system. The new MUCTPI is implemented as a single ATCA blade with high-end processing FPGAs which eliminate double counting of muon candidates in overlapping regions, send muon candidates to L1TOPO, and muon multiplicities to the Central Trigger Processor (CTP), as well as readout data to the data acquisition system of the experiment. A Xilinx Zynq System-on-Chip (SoC) with a programmable logic part and a processor part is used for the communication to the processing FPGAs and the run control system. The processor part, based on ARM processor cores, is running embedded Linux prepared using the framework of the Linux Foundation's Yocto project. The ATLAS run control software was ported to the processor part and a run control application was developed which receives, at configuration, all data necessary for the overlap handling and candidate counting of the processing FPGAs. During running, the application provides ample monitoring of the physics data and of the operation of the hardware. [*]

1 The ATLAS Experiment at the LHC

The ATLAS experiment is a general-purpose experiment at the Large Hadron Collider (LHC) at CERN [1]. It observes proton-proton collisions at an energy of 13 TeV. With about 39 interactions in every bunch crossing (BC) every 25 ns, there are more than 10^9 interactions per second. The trigger system selects those events which are interesting to

[*] Corresponding author: Ralf.Spiwoks@cern.ch

[*] Copyright 2018 CERN for the benefit of the ATLAS Collaboration. CC-BY-4.0 license

physics and which can be recorded to permanent storage at a reasonable rate. The ATLAS trigger system consists of a Level-1 trigger based on custom electronics, which reduces the event rate to a maximum of 100 kHz, and a high-level trigger system based on commercial computers, which reduces the event rate to around 1 kHz on average.

2 The Level-1 Trigger System

The first-level trigger of the ATLAS experiment uses reduced-granularity information from the calorimeters and dedicated muon trigger detectors, see Figure 1. The trigger information is based on multiplicities and topologies of trigger candidate objects. The muon trigger is based on Resistive Plate Chambers (RPC) in the barrel region and Thin-Gap Chambers (TGC) in the end-cap region. The Muon to Central Trigger Processor Interface (MUCTPI) combines the muon candidate counts from the RPC and TGC, taking into account double counting of single muons that are detected by more than one chamber due to geometrical overlap of the muon chambers and the trajectory of the muon in the magnetic field. The MUCTPI sends the muon candidate information to the Topological Processor and the muon counts to the Central Trigger Processor (CTP), which combines the trigger information from the calorimeter trigger, the MUCTPI, and the Topological Processor in order to make the final Level-1 decision.

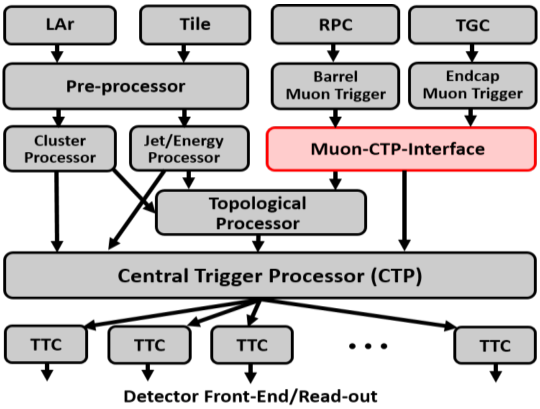


Figure 1. The Level-1 Trigger System showing the path of trigger data to and from the MUCTPI.

3 The Upgrade of the MUCTPI

The upgrade of the MUCTPI to be installed for Run 3 is part of the overall upgrade of ATLAS on the road to High-Luminosity LHC [2]. The new MUCTPI will use optical links to replace bulky electrical cables. Those links will allow the muon trigger detectors to send more muon candidates with more precise information. The new module will provide improved overlap handling and full-precision information to the Topological Processor. It will be built as a single ATCA blade, see Figure 2. It will receive 208 optical links and will use two FPGAs for the overlap handling, counting of muon candidates, and sending candidates to the Topological Processor. A third FPGA will provide the total count of muon candidates to the CTP and read out data to the data acquisition system. A System-on-Chip (SoC) will provide hardware control of the new MUCTPI and integrate it into the ATLAS run control system.

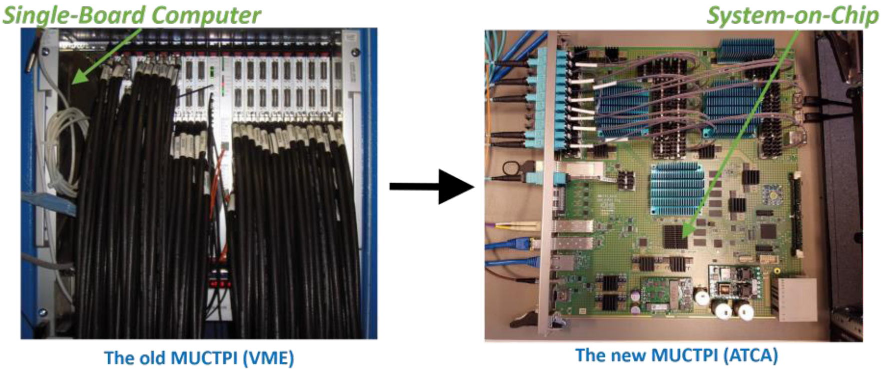


Figure 2. The Upgrade of the MUCTPI from a VME-based crate to a single ATCA blade.

4 The ATLAS Run Control on a System-on-Chip

The run control system [3] of the ATLAS experiment provides a framework for the control of the MUCTPI, e.g. start and stop commands, loading of configuration data, e.g. overlap lookup table (LUT) data, and collection of monitoring data, e.g. counter values. Due to the new technology (ATCA) new ways of communication between the MUCTPI and the run control system had to be investigated. With the old MUCTPI that communication was achieved using a single-board computer. An SoC, Xilinx Zynq 7Z030 [4], with a programmable logic part and a processor part will be used for the communication with the run control system and the processing FPGAs of the MUCTPI, see Figure 3. The programmable logic provides the communication with the processing FPGAs using Chip-2-Chip links, the configuration of the FPGAs using the Slave Serial protocol, and the configuration and monitoring of the MUCTPI hardware with its power, clock and optical modules using serial buses like I2C and SPI. The processor system runs embedded Linux and an ATLAS run control application, which uses low-level user application software developed for the access of all the hardware and processing FPGAs of the MCTPI.

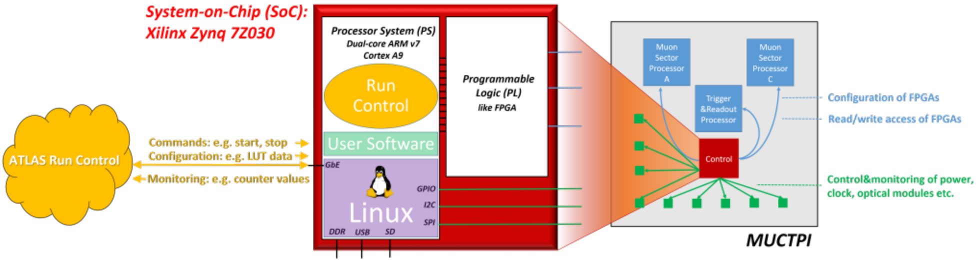


Figure 3. Run Control of the MUCTPI using a System-on-Chip, showing the place of the SoC as part of the hardware of the MUCTPI and as part of the the Run Control system of ATLAS.

4.1 Build Linux with Yocto

The software for the operating system, including a kernel module for accessing memory performing DMA, and all of the user software are built and maintained using the framework of the Yocto Project from the Linux Foundation [5], which allows one to build a complete embedded Linux system. Recipes for building the user software have been developed, in order to fetch, configure, and compile the software, and create all images necessary to boot and run the processor system.

4.2 Cross Compile Run Control

Yocto is used to provide a Software Development Kit (SDK), consisting of the cross compiler and all system libraries, see Figure 4. These are used together with a toolchain file for cross compilation with CMake [6] in order to build the ATLAS TDAQ software [7] including external software packages it depends on, e.g. the ROOT software [8]. The executables and libraries are installed in a directory which is mounted on the processor system using NFS.

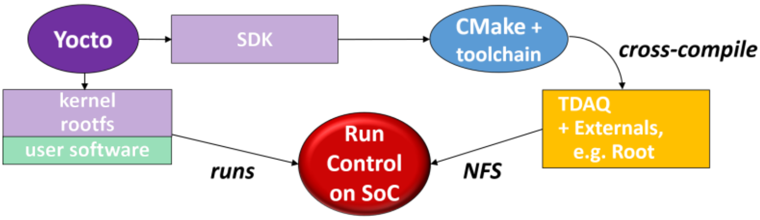
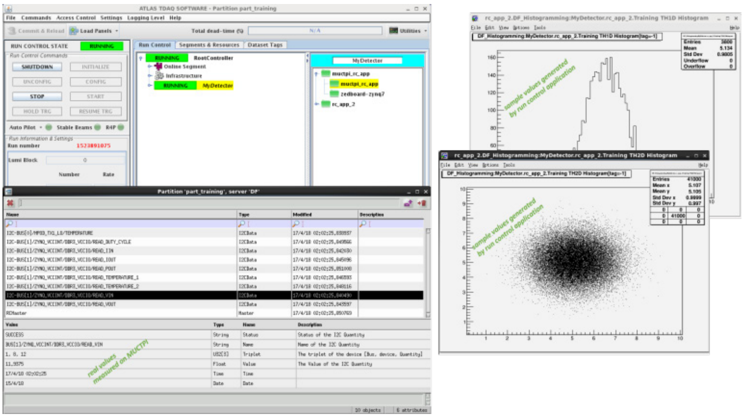


Figure 4. Run Control Software Build Flow, showing Yocto for the build of the kernel and a root file system, and the cross-compilation of the TDAQ software.

4.3 Operate Run Control on the SoC

An ATLAS run control application runs on the SoC. It responds to state transition commands, receives configuration data, accesses the hardware of the MUCTPI, and produces monitoring information. The GUI to interact with this application is shown in Figure 5, including examples of ROOT histograms produced for monitoring.


ATLAS TDAQ SOFTWARE - Partition part_training
File
Commands
Access Control
Settings
Logging Level
Help
Commit & Reload
Load Panels
Utilities
Run Control
Segments & Resources
Dataset Tags
Run Control Commands
SHUTDOWN
INITIALIZE
UNCONFIG
CONFIG
STOP
START
HOLD TRG
RESUME TRG
Auto Pilot
Stable Beams
R4P
Run Information & Settings
Number
Rate
RootController
Online Segment
Infrastructure
MyDetector
muctpi_rc_app
zedboard-zynq7
rc_app_2
Partition 'part_training', server 'DF'
real values measured on MUCTPI
rc_app_2.DF_Histogramming:MyDetector.rc_app_2.Training TH1D Histogram
DF_Histogramming:MyDetector.rc_app_2.Training TH1D Histogram[tag=-1]
160
140
120
100
80
60
sample values generated by run control application
rc_app_2.DF_Histogramming:MyDetector.rc_app_2.Training TH2D Histogram
DF_Histogramming:MyDetector.rc_app_2.Training TH2D Histogram[tag=-1]
1
2
3
4
5
6
7
8
9
10
sample values generated by run control application
File
Edit
View
Options
Tools
Help

RUN CONTROL STATE: RUNNING

Total dead-time (%): N/A

Run number: 1523891075

Lumi Block: 0

Name	Type	Modified	Description
I2C-BUS[0]/MP03_TX1_L0/TEMPERATURE	I2CData	17/4/18 02:02:25,838557	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_DUTY_CYCLE	I2CData	17/4/18 02:02:25,849566	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_IIN	I2CData	17/4/18 02:02:25,842680	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_IOUT	I2CData	17/4/18 02:02:25,845096	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_POUT	I2CData	17/4/18 02:02:25,851008	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_TEMPERATURE_1	I2CData	17/4/18 02:02:25,846593	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_TEMPERATURE_2	I2CData	17/4/18 02:02:25,848116	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_VIN	I2CData	17/4/18 02:02:25,840490	
I2C-BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_VOUT	I2CData	17/4/18 02:02:25,843597	
RCMaster	Master	17/4/18 02:02:25,850763	

Value	Type	Name	Description
SUCCESS	String	Status	Status of the I2C Quantity
BUS[1]/ZYNQ_VCCINT/DDR3_VCCIO/READ_VIN	String	Name	Name of the I2C Quantity
1, 8, 12	U32[3]	Triplet	The triplet of the device [Bus, device, Quantity]
11.9375	Float	Value	The Value of the I2C Quantity
17/4/18 02:02:25	Time	Time	
15/4/18	Date	Date	

10 objects | 6 attributes

Entries	3800
Mean	5.134
Std Dev	0.9805
Underflow	0
Overflow	0

Entries	41000
Mean x	5.107
Mean y	5.105
Std Dev x	0.9999
Std Dev y	0.997

0	0	0
0	41000	0
0	0	0

Figure 5. ATLAS Run Control GUI and Monitoring Data from a standalone test of the MUCTPI, showing some run control windows and histograms from a dummy run control application.

5 Summary

A System-on-Chip is successfully used to provide communication between the ATLAS run control system and the new MUCTPI. The Linux kernel and the root file system are built using the Yocto framework. That framework had to be extended in order to compile the user application software for access of the MUCTPI hardware. Because building the ATLAS run control software also in the same framework proved to be impractical, the software development kit provided by Yocto was used together with a toolchain file to cross compile the ATLAS run control software in its own build environment. Finally, a run control application was developed which runs on the SoC, controls and configures the MUCTPI, and reads monitoring information, including ROOT histograms, from it. In the future it will be investigated if it is possible to improve the building and maintenance of the software further by making use of a common Linux distribution like CERN CentOS. But it has already been shown that a SoC can replace the single-board computers of the previous VME-based system in order to run a run control application on the new ATCA-based systems.

References

1. ATLAS Collaboration, JINST 3 S08003 (2008).
2. M. Silva Oliveira et al., Proc. TWEPP 2018 (in preparation).
3. ATLAS Collaboration, CERN/LHCC/2003-22 (2003).
4. Xilinx Inc., https://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html.
5. The Yocto Project, https://www.yoctoproject.org.
6. The CMake Software, https://cmake.org.
7. ATLAS TDAQ Software, https://gitlab.cern.ch/atlas-tdaq-software.
8. CERN ROOT Software, https://root.cern.ch.