ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA035

FIRMWARE LAYER IMPLEMENTATION OF THE nBLM AND icBLM SYSTEMS FOR ESS PROJECT

W. Jalmuzna, G. Jabłoński, R. Kiełbik, W. Cichalewski *, TUL, DMCS, Łódź, Poland
I. Dolenc Kittelmann, F. S. Alves, H. Carling, S. Farina, K. Rosengren, T. J. Shea
ESS ERIC, Lund, Sweden

Abstract

Both ionization chamber Beam Loss Monitor (icBLM) and neutron Beam Loss Monitor (nBLM) systems are fundamental components of European Spallation Source (ESS) accelerator safety systems. Main responsibility of this system is instantaneous and reliable detection of accelerated proton beam loss that exceeds predefined safety threshold. Nowadays DMCS (as an in-kind partner to ESS) is responsible for beam loss detection algorithm implementation, evaluation and deployment in firmware. As a hardware platform for mentioned systems MTCA.4 based form factor electronic components have been chosen (delivered by IOXOS). This contribution focuses on both cases (nBLM and icBLM) firmware realisation presentation. Proposed and developed firmware structure and functional blocks that fulfils specified by ESS requirements are described. Additionally, some aspects of the system FPGA circuit resource usage and achieved performance is being discussed.

INTRODUCTION

The ESS is a material science facility, which is currently being built in Lund, Sweden and will provide neutron beams for neutron-based research [1]. The neutron production will be based on bombardment of a tungsten target with a proton beam of 5 MW average power. A linear accelerator (linac) [2] will accelerate protons up to 2 GeV and transport them towards the target through a sequence of a normal conducting (NC) and superconducting (SC) accelerating structures (Fig. 1).

As in case of all future high-power accelerators, ESS linac operation will be limited by beam losses if machine activation is to be kept low enough for hands-on maintenance. Moreover, loss of even a small fraction of intense ESS beam can result in a significant increase of irradiation levels, ultimately leading to damage to the linac equipment. BLM systems are designed to detect showers of secondary particles produced by lost beam particles interacting with the accelerator equipment. By providing information about beam loss levels, BLM systems play an important role in machine fine tuning as well as machine protection from beam-induced damage by detecting unacceptably high beam losses and promptly inhibiting beam production.

The ESS BLM consist of two types of systems, differing in detector technology [3]. The icBLM system is based on 266 ionisation chambers [4–6] as detectors, located almost exclusively throughout the SC parts of the linac. These detectors are simple and well established for beam loss monitoring purposes. Conversely, the nBLM system consists of 82 neutron detectors, specially designed to primarily cover the lower energy part of the ESS linac and thus complement the icBLM system.

Both systems are equipped with the read-out hardware supporting the real-time FPGA-based data processing.

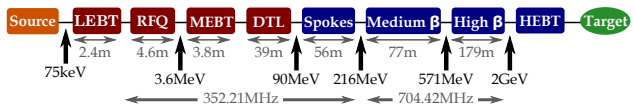

Figure 1: The ESS linac layout. Red colour represents the NC and blue the SC parts of the linac. [2]

HARDWARE AND FIRMWARE STRUCTURE

The platform standard selected for BLM systems is uTCA. This is quite complex standard with rich supporting infrastructure, which can accommodate wide range of extension modules (called AMCs) to implement all needed function. The main platform for BLM implementation is IFC1410 AMC module, which is 2 HPC FMC carrier. It provides ultimate computation power in terms of FPGA resources and is equipped by additional PPC processor to implement control system integration.

The application specific (icBLM and nBLM) data acquisition is performed by additional FMC modules (250 MHz AD3111 for nBLM and FMC-Pico-1M4 1 MHz for icBLM).

The general structure of the FPGA implementation is enforced by TOSCA framework, which provides access channels to PCIe and DDR3 resources. It also defines interfaces between FMC modules and user logic. Its simplified diagram is presented in Fig. 2. The parts implementing BLM algorithms are marked with green.

FIRMWARE IMPLEMENTATION

The firmware implementations of nBLM and icBLM share common components, related mainly to mass data transfer, that are not provided by the TOSCA framework. Both systems need to provide large quantities of various types of data to the CPU. These data are buffered in two DDR3 memory blocks. The total bandwidth (read + write) of each of these memory blocks is 2 GB/s, when the memory controller is operating at 275 MHz. The data are logically

* wcichal@dmcs.pl

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA035

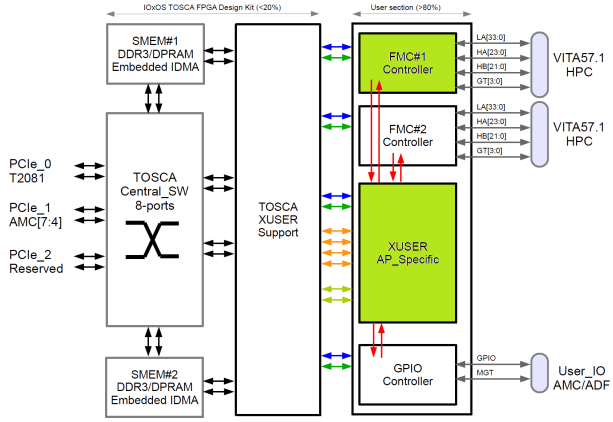

Figure 2: The TOSCA framework structure.

grouped into several (14 in case of nBLM, 9 for icBLM) independent data streams (called channels) and are available to the software via DMA through the PCI Express interface, both to the internal POWER CPU and the external CPU in MTCA chassis.

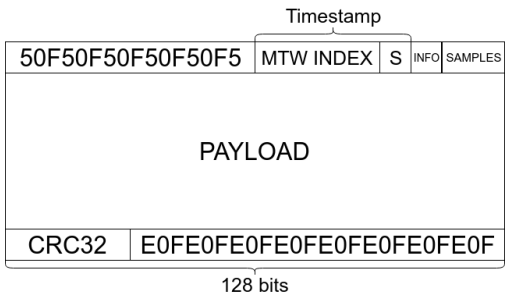

Figure 3: Layout of a data frame.

The data streams are divided into frames protected by 32-bit CRC (see Fig. 3), allowing time-stamping and integrity checking. The outputs from framers are sent to Data Channel Controllers, managing circular buffers in two DDR3 memory banks. The Data Channel Controllers are connected to arbiters, allowing to share the single memory access channel between different data streams. The arbiters use round-robin scheduling algorithm.

On the CPU side, two independent threads are reading data from each of the memory banks. To reduce the CPU load, interrupts are used to indicate that unread data are available in any of the buffers. The threshold for the amount of data generating an interrupt is configurable. In order to prevent keeping small amounts of data in buffers indefinitely, the latency timer is employed. It flushes the buffers periodically if the data rate is low.

nBLM Firmware

A single AMC module processes data from 6 ADC channels, the remaining 2 are treated as spares [7, 8]. The block diagram of a data processing path for a single channel is presented in Fig. 4. The final outcome of every path is the BEAM_PERMIT signal, indicating that the level of beam loss detected by a particular detector is below the threshold. The individual algorithm blocks are described in C++ and synthesised using Vivado High Level Synthesis. They are connected using the AXI Stream interface. The main data processing chain, that operates in lockstep at every 125-MHz clock cycle, consists of 7 blocks:

- Preprocessor: subtracts pedestal from data samples and performs comparison of samples with the threshold.
- Event detector: identifies "interesting events" and counts ADC saturations.
- Event aligner: delays one event, allowing simultaneous presentation of two subsequent events to the next block.
- Neutron counter: computes the number of neutrons causing "interesting events", taking into account two events at the MTW boundary if necessary.
- Neutron summarizer: produces the summary of neutrons counted within each MTW.
- Interlock logic: generates the BEAM_PERMIT signal.
- MPS interface: transmits BEAM_PERMIT signal to the Machine Protection System.

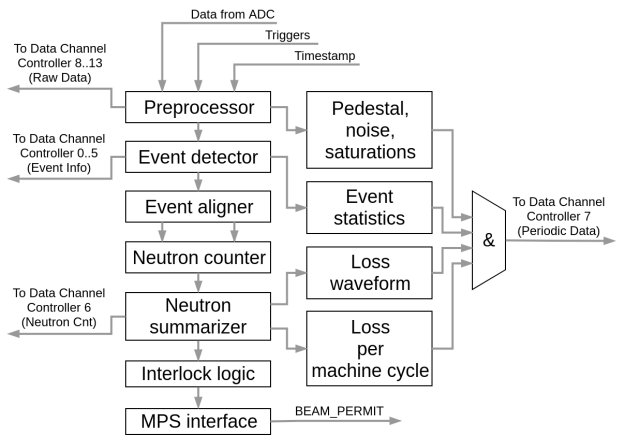

Figure 4: Data flow in a single channel.

As this chain controls the BEAM_PERMIT interlock signal, no stalls are allowed in the pipeline. Two ADC samples are processed in one clock cycle.

The remaining data processing blocks produce different statistics – "periodic data" – for operator panel visualisation purposes. In these data occasional intermittent stalls, leading to data loss (resulting e.g. from the lack of available DDR3 memory bandwidth), are acceptable.

The algorithm parameters can be set via PCI Express using the TOSCA TCSR interface. The periodic data and a selection of intermediate data – used for determining the state of BEAM_PERMIT on different stages of processing – are logically grouped into 14 independent data streams. The data flow diagram in the system is presented in Fig. 5.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA035

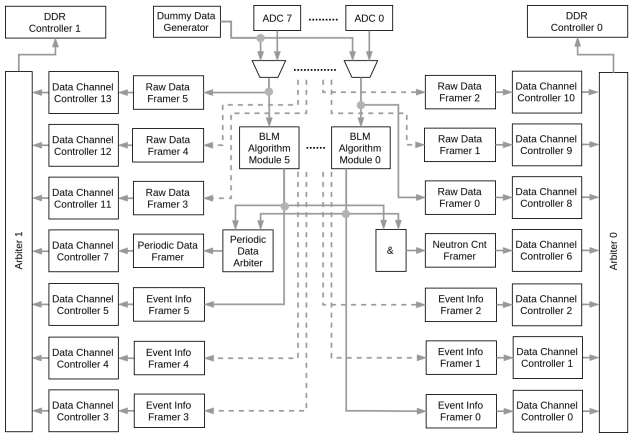

Figure 5: An overview of data flow in firmware.

The system is still in development phase, the interlock logic and MPS interface are not yet fully implemented.

Current FPGA resources usage have been summarised in Table 1.

Table 1: FPGA Resource Usage for nBLM Implementation

	Used	Total	Used [%]
LUT	136594	242400	56
LUTRAM	4378	112800	4
FF	146061	484800	30
BRAM	305	600	50
DSP	90	1920	5

icBLM Firmware

The single AMC module is equipped with 2 FMC-Pico-1M4 modules. The total number of processed channels is 8. The setup has much less strict requirements concerning sampling than nBLM - ADCs are sampling with 1 MSPS with 20 bit resolution. In this case all data streams are lower, so both processing and DAQ parts can be relaxed.

The icBLM firmware implements additional functional blocks used for diagnostic such as HV slow modulation detection and base-line detection. The main purpose of modulation detection block is to measure modulation frequency generated by detectors' power supply on all input channels (< 1Hz). To accomplish the task, special cascade of filters was implemented to act as an anti-aliasing filter. The result of filtering is fed to FFT block for exact frequency calculation. Base-line detection block measures background levels during not beam-qualified pulses. The result is used both for diagnostic and as a correction value for measured raw data.

FPGA resources usage for current icBLM implementation have been summarised in Table 2.

Table 2: FPGA Resource Usage for icBLM Implementation

	Used	Total	Used [%]
LUT	67152	242400	27
LUTRAM	6443	112800	5
FF	82889	484800	17
BRAM	274	600	45
DSP	26	1920	2

icBLM AND nBLM PROTECTION FUNCTIONS

The primary task of the blocks is to evaluate input signals against defined conditions and inhibit beam permission if necessary. The module can drop the BEAM_PERMIT signal on the line connected to the FBIS through the FBIS interface.

The data for each channel (arriving with 1 MHz frequency) is filtered by several filters (moving-average, relaxation filter, X/Y algorithm and simple average) and each filter output is compared with user defined thresholds. All results are used to calculate single BEAM_PERMIT signal for individual ADC channel using configurable logical function. The signals for all 8 channels are connected to additional block, which allows exclusion of selected channels from global BEAM_PERMIT signal. The final outcome of calculations is passed to FBIS interfaces and processed by external MPS systems.

HLS BASED ALGORITHMS IMPLEMENTATION

All algorithms described in previous section are implemented using Xilinx High Level Synthesis tool. The tools allows to describe desired functionality in behavioural way using C++ language. The main advantage of such approach is speed-up of development. Algorithms can be implemented without deep knowledge of internal FPGA structure and the tool automatically selects best possible functional block to perform defined operations. In addition, implementation on C++ level allows fast verification of the blocks by compiling the program in the development environment.

The resulting components can be easily integrated with other user logic implemented in VHDL/Verilog using standard AXI, AXIS or fifo interfaces.

During the implementation of BLM systems, we have concluded that the performance offered by the HLS tool is sufficient to select it for production environment.

SIMULATIONS

At each development stage the firmware modules can be verified using behavioural simulations. The simulation environment was prepared by the supplier of main BEE components, IOxOS. It encapsulates the behavioural model of the entire Tosca framework, DDR3 memory and sample user application module (XUSER AP_Specific block

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA035

in Fig. 2). Auxiliary functions allow to emulate an access of CPU (through the PCI Express interface) to the memory and the internal firmware registers (e.g. for simulation of configuring, resetting and enabling data channels, handling interrupts, etc.).

For the purpose of verification of nBLM or icBLM system, the model of sample user application module is replaced with the model of the system to be verified, which is automatically extracted from the Vivado project (by means of 'export_ip_user_files' and 'export_simulation' commands). Such an automation simplifies the creation of simulation model of the entire system regardless any changes introduced to Tosca framework or nBLM / icBLM modules.

The generated model allows to simulate around 1 microsecond of the system operation per second (using Intel(R) Core(TM) i7-6900K CPU @ 3.20GHz).

Apart from the global simulation model, its components have also individual testbenches. The entire nBLM main data processing chain has a separate testbench, allowing comparison of results from the original C++ code to its HLS hardware counterpart.

DEPLOYMENT AND HIGHER SOFTWARE LEVELS

The main software packages for BLM subsystems were implemented as EPICS modules developed in framework of E3. E3 is a framework developed by ESS team, which provides all EPICS infrastructure and flexible tools for application development and deployment.

In addition to EPICS layer, several low level testing tools have been implemented to allow fast and efficient debugging of firmware implementation. They can be run as Linux console programs and they allow to perform full configuration of algorithm blocks and data acquisition to dedicated files.

As a highest level of software, the dedicated graphical control panels have been created. The main tool for this implementation is Control System Studio software package. The control panels allow easy access to all functions of individual BLM modules and provide presentation layer for data acquisition subsystems.

CONCLUSIONS

The paper presents current status of development of nBLM and icBLM FPGA based systems. It also shows that modern tools such as Xilinx HLS are capable of creating most performance demanding modules, which can be used for final accelerator applications.

Some features of the systems are still not implemented, but overall functionality is sufficient to perform real evaluation in accelerator environment with beam.

ACKNOWLEDGEMENTS

This work has been partially supported by the Polish Ministry of Science and Higher Education, decision number DIR/WK/2018/02.

REFERENCES

[1] "ESS Technical Design Report," ESS, Tech. Rep. ESS-0016915, 2013.

[2] M. Eshraqi *et al.*, "The ESS linac," in *Proc. of 5th International Particle Accelerator Conference (IPAC 2014)*, MOXAP07, Dresden, Germany, 2014.

[3] I. Dolenc Kittelmann *et al.*, "Simulations and detector technologies for the Beam Loss Monitoring System at the ESS linac," in *Proc. of 57th ICFA Advanced Beam Dynamics Workshop on High-Intensity and High-Brightness Hadron Beams (HB2016)*, paper THAM6Y01, Malmö, Sweden, 2016.

[4] V. Grishin *et al.*, "Ionisation Chambers as Beam Loss Monitors for ESS linear accelerator," in *Proc. of 6th International Beam Instrumentation Conference (IBIC2017)*, paper WEPWC03, Grand Rapids, MI, USA, 2017.

[5] I. Dolenc Kittelmann, "Requirements and technical specifications - ESS ICBLM system," ESS, Tech. Rep. ESS-1158292, 2019.

[6] I. Dolenc Kittelmann *et al.*, *Ionisation Chamber Based Beam Loss Monitoring System for the ESS Linac*, Presented at 8th International Beam Instrumentation Conference (IBIC 2019), Malmö, Sweden.

[7] I. Dolenc Kittelmann *et al.*, Neutron sensitive Beam Loss Monitoring system for the ESS linac, presented at 8th International Beam Instrumentation Conference (IBIC 2019), Malmö, Sweden.

[8] G. Jabłoński *et al.*, "Fpga-based data processing in the neutron-sensitive beam loss monitoring system for the ess linac," in *2019 MIXDES - 26th International Conference "Mixed Design of Integrated Circuits and Systems"*, Jun. 2019, pp. 101–105. DOI: 10.23919/MIXDES.2019.8787166.