RELIABILITY TESTS OF THE LHC BEAM LOSS MONITORING FPGA FIRMWARE

C. F. Hajdu, B. Dehning, S. Jackson, C. Zamantzas, CERN, Geneva, Switzerland

Abstract

The LHC Beam Loss Monitoring (BLM) system is one of the most complex instrumentation systems deployed in the LHC. In addition to protecting the collider, the system also needs to provide a means of diagnosing machine faults and deliver a feedback of losses to the control room as well as to several systems for their setup and analysis. It has to transmit and process signals from almost 4’000 monitors, and has nearly 3 million configurable parameters.

In a system of such complexity, firmware reliability is a critical issue. The integrity of the signal chain of the LHC BLM system and its ability to correctly detect unwanted scenarios and thus provide the required protection level must be ensured. In order to analyze the reliability and functionality, an advanced verification environment has been developed to evaluate the performance and response of the FPGA-based data analysis firmware. This paper will report on the numerous tests that have been performed and on how the results are used to quantify the reliability of the system.

INTRODUCTION

The Beam Loss Monitoring system [1] is one of the most critical among the numerous systems installed for the protection of the LHC. It has to prevent quenches in the superconducting magnets and protect the machine components against damage. The system comprises nearly 4’000 detectors, ionisation chambers and secondary emission-based monitors, mounted onto the elements under supervision. The analogue output signal of the sensors is digitised by data acquisition cards [2], generally referred to as Current to Frequency Converter (CFC), installed in the tunnel. The data is then transmitted to the Threshold Comparators (TC) [3] via redundant broadband optical links. The TCs, installed in VME crates distributed in surface buildings around the LHC, collect and analyse the data. Their FPGA-based processing algorithm calculates integrals of the signals over different time windows, compares them to their respective abort thresholds and can trigger a beam abort as appropriate through the Combiner and Survey (CS) card installed in the same VME crate.

The integrity of the whole signal chain needs to be verified in order to ensure that the system provides the required level of protection. Firmware reliability in particular is a key issue. Due to the great complexity and sequential nature of the design, exhaustive testing – that is, verification by applying every possible sequence of input combinations to the design and checking its outputs – of the TC firmware is impractical [4]. Therefore, an effort has been made to lay the foundations of a comprehensive verification environment instead, implementing different approaches of verification, each of them focusing on different aspects of the system under test (see Fig. 1).

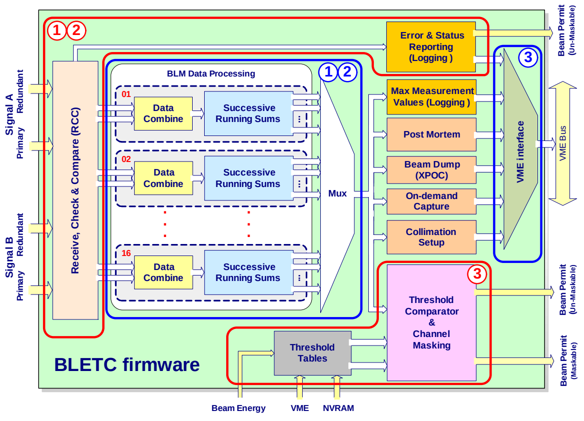
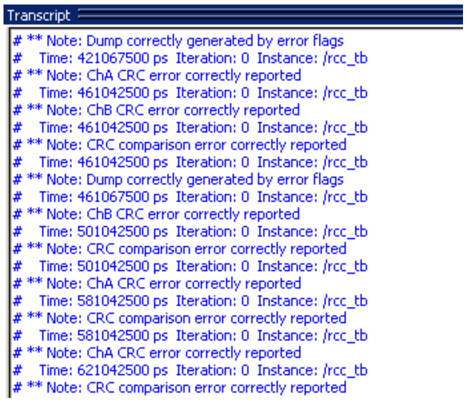


Figure 1: Block diagram showing the scope of the verification. Legend of the types of verification used: (1) Black box simulation, (2) Hardware-based approach, (3) Software-based approach.

Consequently, simulation testbenches, custom hardware and software suites have been designed and developed for the verification of critical blocks as well as to verify certain aspects of the behaviour of the installed system. The aim of this paper is to detail the implementation of these methods and the results obtained therewith.

SIMULATION

Functional simulation involves simulating the design description – in this case, in VHDL – to verify that the system meets the functional requirements stated in its


Transcript

```
# ** Note: Dump correctly generated by error flags
#    Time: 421067500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: ChA CRC error correctly reported
#    Time: 461042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: ChB CRC error correctly reported
#    Time: 461042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: CRC comparison error correctly reported
#    Time: 461042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: Dump correctly generated by error flags
#    Time: 461067500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: ChB CRC error correctly reported
#    Time: 501042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: CRC comparison error correctly reported
#    Time: 501042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: ChA CRC error correctly reported
#    Time: 581042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: CRC comparison error correctly reported
#    Time: 581042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: ChA CRC error correctly reported
#    Time: 621042500 ps  Iteration: 0  Instance: /rcc_tb
# ** Note: CRC comparison error correctly reported
```

Figure 2: Excerpt of the output of the automatic checker for a test performed on the receiver part using ModelSim.

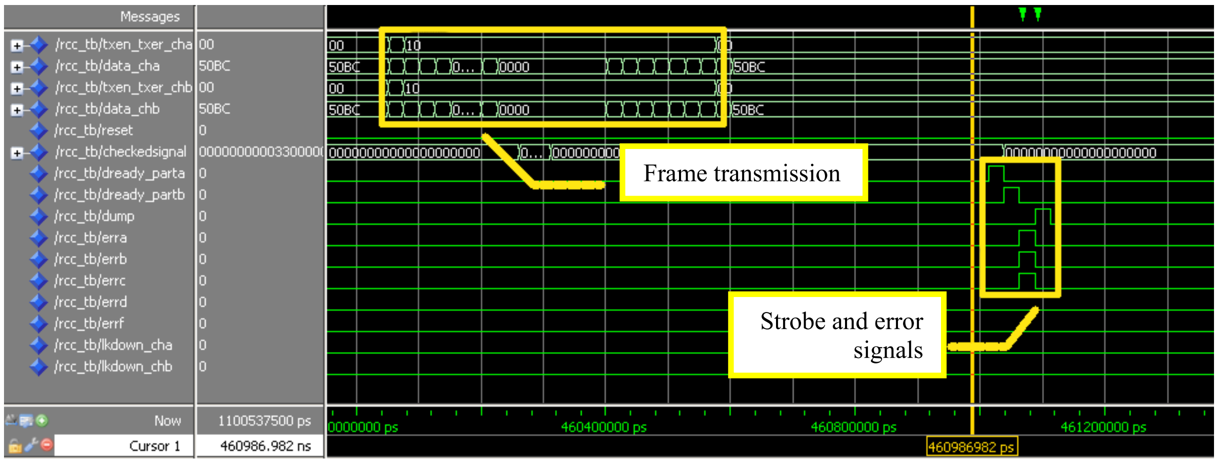


Figure 3: Example of the output waveform of the functional simulation performed on the receiver part using ModelSim.

specification. In order to carry out the functional simulation of the critical blocks of the Threshold Comparator firmware, several testbenches have been developed.

The development has been done following the rules of the "black box" methodology, that is, entirely based on the specification without any knowledge about the internal structure of the block. The testbench reads the stimulus to be passed to the Unit Under Test (UUT) from a text file and checks the outputs of the UUT versus the stimulus automatically. This facilitates regression testing: the testcases used for finding an error can very easily be re-executed for newer versions of the design, thus allowing to find out if it has regressed, that is, a previously fixed bug has reappeared. It also allows the comparison of the behaviour of different versions of the code by applying the same stimulus and comparing the outputs, thereby making it easier to detect new bugs.

The execution of the testbench was done in ModelSim. Sample outputs of the automatic checker and an output waveform can be seen in Fig. 2 and 3, respectively.

HARDWARE-BASED CHECK

In order to verify the quality of the deployed firmware and the behaviour of the Threshold Comparator card under real conditions, installed in the VME crate, a way to emulate the optical output signal of the CFC cards has been developed.

The hardware implementation of the Threshold Comparator consists of a standard VME-compatible FPGA carrier board [5], used throughout the Beam Instrumentation group at CERN, fitted with a mezzanine card [6] for the reception of the output signals of the CFC cards. Since every TC card receives signal from two CFC cards and the optical links are redundant, every TC mezzanine hosts four optical receivers.

For the purposes of this type of verification, a new mezzanine card and the corresponding custom FPGA firmware have been developed for the same FPGA carrier board (see Fig. 4). This mezzanine card hosts two Gigabit Optical Hybrid (GOH) transmitters [7], thus the setup allows the direct emulation of one CFC card. With the use of optical signal splitters, it is possible to increase the number of connections and drive several TC modules.

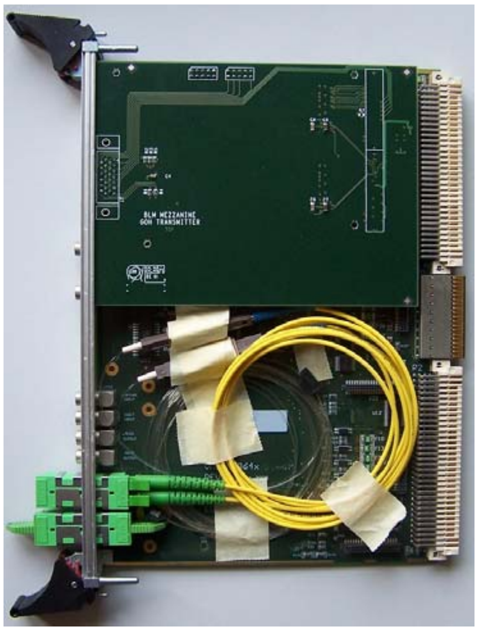


Figure 4: The VME64x carrier board with custom FPGA firmware and dual gigabit optical transmitter mezzanine developed to emulate the acquisition electronics.

This scheme makes it possible to emulate errors in the transmission or in the physical layer as well as to imitate wrong configurations. It allows the evaluation of the way the system handles the redundant data transmission and its response when any of the many self-checking mechanisms embedded in the transmission indicate a failure.

For example, single or multiple errors can be injected into the checksums, CFC card identity numbers, or frame sequence numbers being employed.

This setup also allows the transmission of arbitrary data to target the processing parts of the TC. The data can be either the direct output of the TC read back from a logging database, the contents of a TC internal circular buffer frozen after an event, or any imaginary loss scenario described in a text file and loaded into the VS internal memory. In that way, the verification environment can be used to compare the change of response between different versions of the firmware, or for quantifying the linearity of the internal data processing algorithms.

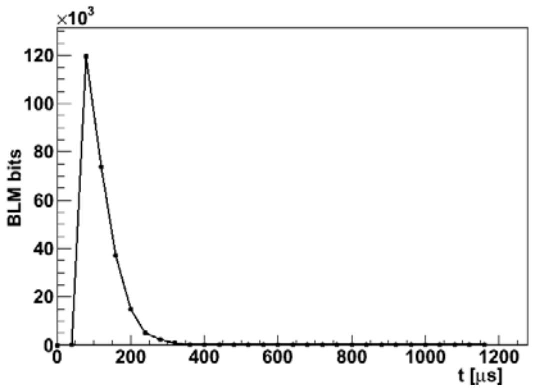


Figure 5: Recording of a single bunch loss in the LHC using the TC on-demand capture buffer. Each sample provides the integrated losses of the last 40 μs.

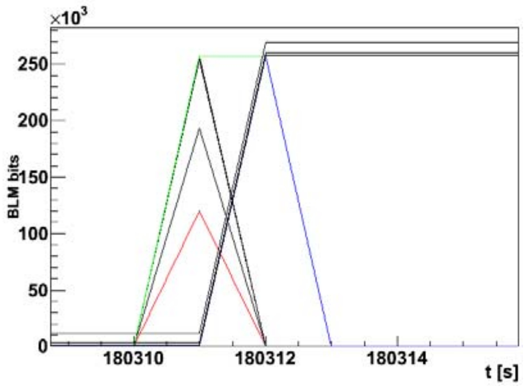


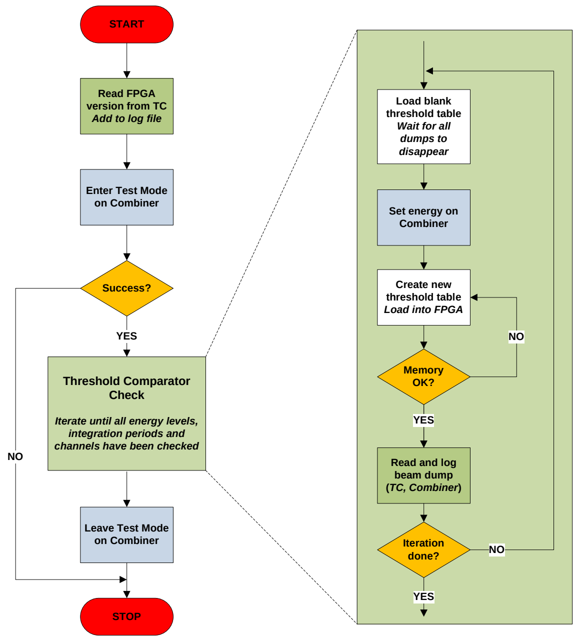
Figure 6: Time diagram of the response of the 12 integration histories as calculated by the TC following the VS transmission of the input stimulus shown in Fig. 5.

In addition, the playback of actual captured loss data from the LHC can provide valuable information to the physicists for the fine-tuning of the threshold values. For an example of data transmitted and the response of the system, see Fig. 5 and 6, respectively.

SOFTWARE-BASED CHECK

To make sure that the Threshold Comparator cards correctly decode the signals they receive and every one of them has the ability to request a beam abort whenever necessary, an exhaustive test of the block of the firmware comparing the measurements to predefined beam abort thresholds is required. This implies making all thresholds trigger a beam dump one by one. The 12 integrals of different lengths being calculated for each of the 16 detectors connected to one TC card at 32 beam energy levels correspond to 6’144 testcases per TC card, or 98’304 testcases for a VME crate of 16 TC cards.

In the VME crates hosting the TC cards and the CS card, a PowerPC-based CPU card, generally referred to as Front End Computer (FEC), is also part of the standard installation. Software processes running on its Linux-based LynxOS operating system have been developed to successively load purpose-made threshold maps into the memory of the TC card, thereby making one selected threshold trigger a beam abort, and check the results on both the TC and the CS cards. The flowchart of this process, called “Exhaustive Threshold Triggering”, can be seen in Fig. 7.


START
Read FPGA version from TC
Add to log file
Enter Test Mode on Combiner
Success?
YES
NO
Threshold Comparator Check
Iterate until all energy levels, integration periods and channels have been checked
Leave Test Mode on Combiner
STOP
Load blank threshold table
Wait for all dumps to disappear
Set energy on Combiner
Create new threshold table
Load into FPGA
Memory OK?
NO
YES
Read and log beam dump
(TC, Combiner)
Iteration done?
NO
YES
Figure 7: Simplified flowchart of the “Exhaustive Threshold Triggering” check executed on each TC module.

An iteration of the “Exhaustive Threshold Triggering” check on a single TC card requires about 12 hours to run and creates a log file with all executed testcases getting a “PASS” or “FAIL” grade.

During execution, the beam abort check also provides a robustness check of the highly critical non-volatile memory storing the threshold values and all other operational parameters by checking the contents of the memory against what was flashed. In case of mismatch, the flashing is repeated and the number of iterations required is saved into the log file.

During the development of the verification procedure, some doubts arose about the reliability of data transmissions over the VME interface. This led to the elaboration of a special test procedure which involved the execution of the same read-write operation 500’000 times in succession. The results obtained were consistent for all iterations, which led to a greater confidence in the robustness of the implemented VME interface.

In addition to these tests, a script has been developed to read out the unique card identifiers stored on a chip on each TC and CFC card. These identifiers can be checked very easily versus any previous known state, thus replaced cards or changes in connectivity can be detected. The whole process requires a few minutes to execute and provides a quick check after a technical stop or an intervention to the system before the more elaborate Management of Critical Settings (MCS) procedure [8] can be executed.

CONCLUSIONS

Testing by the black box methodology has revealed some violations of specification, which resulted in an in-depth review of the block under test. A considerable part of the block has been rewritten, either to follow the specifications more closely or to enhance testability.

The verification tools have been developed in a modular structure that allows the implementation and inclusion of additional checks if further doubts arise for any part of the system.

It has been shown that the versatility achieved by using different methods to test the system accelerates the development of such a verification environment and also allows covering more cases by selective targeting.

As an additional advantage, many of the tools implemented have been re-used, with only small modifications, in the tedious commissioning phase of the complete system, in this case, tracking down non-conformities in the construction and electronic parts.

Finally, the verification suite provides the required additional safeguards and lays the foundations of a release protocol to be followed for future modifications of the reprogrammable parts of the mission critical BLM system.

REFERENCES

[1] B. Dehning, et al. “*The Beam Loss Monitoring System*”, Chamonix 2004

[2] E. Effinger et al., “*The LHC Beam Loss Monitoring System's Data Acquisition Card*”, 12th Workshop on Electronics for LHC and future Experiments (LECC 06), Valencia, Spain.

[3] C. Zamantzas et al., “*The LHC Beam Loss Monitoring System’s Surface Building Installation*”, 12th Workshop on Electronics for LHC and future Experiments (LECC 06), Valencia, Spain.

[4] Kenneth L. Short, “*VHDL for Engineers*”, Prentice Hall, 2008, p. 254.

[5] Engineering Specification, “*VME64x Digital Acquisition Board for the LHC Trajectory and Closed Orbit System*”, LHC Project Document No. LHC-BP-ES-0002, Version 1.1, 09-03-2004.

[6] Schematic files of the BLM Mezzanine Card (Rev. 3), *https://edms.cern.ch/document/493588/3*

[7] GOL Reference Manual (Preliminary), http://ab-div-bdi-bl-blm.web.cern.ch/ab-div-bdi-bl-blm/Electronics/GOH/gol_manual_v1.9.pdf

[8] C. Zamantzas et al., *“Configuration and Validation of the LHC Beam Loss Monitoring System”*, 9th European Workshop on Beam Diagnostics and Instrumentation for Particle Accelerators (DIPAC09), Basel, Switzerland.