A compact plug-in module for LHC-like trigger emulation

P. Kokkas [1], N. Manthos [1], G. Sidiropoulos [1], P. Vichoudis [2]

[1] University of Ioannina, Ioannina, Greece
[2] CERN, 1211 Geneva 23, Switzerland

Georgios.Sidiropoulos@cern.ch

Abstract

A compact trigger emulation module for evaluating electronic systems for LHC applications has been built using off-the-shelf components. The emulator, which is based on an FPGA, generates both programmable and true-random trigger patterns in compliance with the LHC triggering rules. For the true-random trigger part, the source of randomness is the avalanche effect on a transistor emitter-base diode. The system can be used either as a plug-in module for VME systems or as a standalone device controlled via a standard USB link by a PC running LabVIEW.

I. Motivation

The need of a flexible device emulating realistically the random arrival of particles on a detector is apparent, especially during the construction/testing phases of the corresponding on and off-detector electronic systems. Having the ability to define certain parameters such as the mean rate and the distribution of the randomly generated trigger signals would make such a device extremely useful. Additionally, if the device could be portable and work in stand-alone mode, it would make it even more attractive. All these led to the development of a compact plug-in trigger emulator module based on a source of true randomness.

Such a device, apart from being useful for High Energy Physics experiments, is also suitable for nuclear physics and radiology experiments where the emission of a radioactive source needs to be emulated.

II. Hardware Implementation

The trigger emulator plug-in module is based on an FPGA and a random noise generation circuitry. The circuitry that forms the random generator produces pulses with random intervals. The source of randomness is the avalanche effect on a transistor emitter-base diode, biased by a digitally controlled voltage source.

Figure 1 illustrates the schematic of the random number generator. The transistor on the left side of the schematic is the avalanche noise generator. The output is amplified by a two-stage transistor amplifier. The amplified noise signal is superimposed on a configurable DC level (created by a digital potentiometer) and it is fed to an FPGA internal gate which acts as a comparator and produces random transitions. The programmable voltage source on the left side of the schematic controls the biasing of the transistor emitter-base diode, changing in this way the mean value of the distribution. Figure 2 displays the distribution of 1024 random triggers generated by the emulator, where the solid curve is an exponential fit to the histogram.

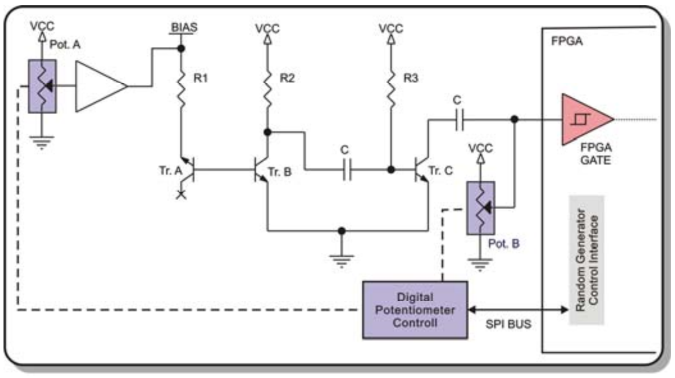

Figure 1: Schematic of the true random interval generator.

The output of the random generator is treated by a digital processing circuitry implemented in an FPGA. Its principal function is to measure the time intervals between the transitions generated at the output of the logic gate extracting in this way random numbers. Additionally it applies specific rules and functionality depending on the application (details of the functionality can be found in section III). Finally the FPGA provides the necessary communication interface for setting the desired working parameters. The module includes also an optical transceiver for inter-connection with the various off-detector electronic systems.

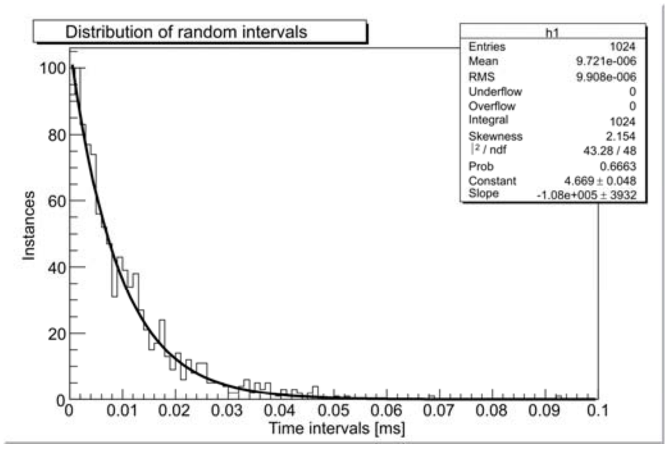

Figure 2: The exponential distribution of 1024 generated true random trigger intervals with mean rate ~100 KHz.

This compact module (12.3cm x 4cm x 1.2cm) can be plugged as a mezzanine on various readout systems i.e. VME64 [1] but it can also work as a stand-alone device with the help of a support board that provides the necessary power and a USB interface in order to communicate with a standard PC. The support board can be viewed in figure 4.

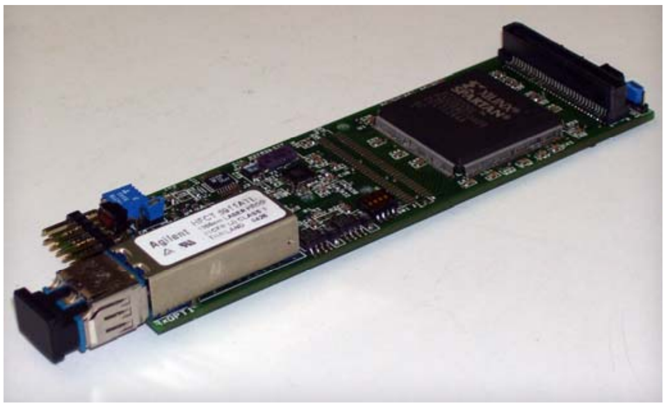

Figure 3: The trigger emulator plug-in module. On the left, the optical transceiver is shown while on the right, the FPGA and the mezzanine connector.

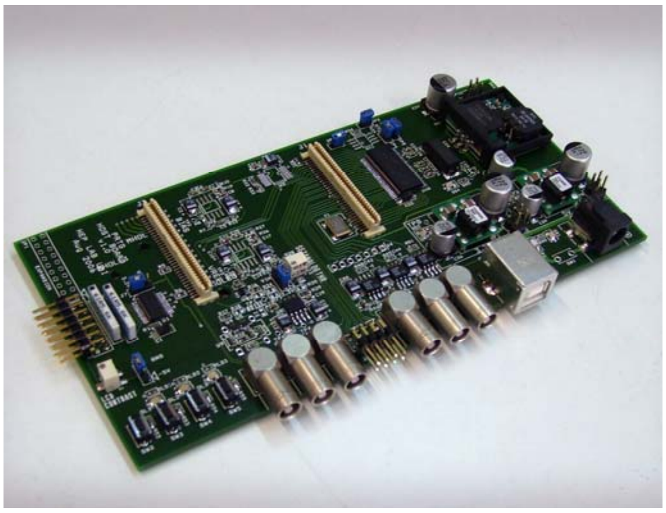

Figure 4: Photograph of the support board that provides power, additional clock, JTAG and USB interface to the trigger emulator mezzanine, used for evaluation purposes.

III. Functionality

The purpose of the trigger emulator is to generate sequences of trigger signals (trigger patterns) either based on the random number generator or on user defined parameters. In addition it can provide other timing and fast command information (clock signal, resynchronization signal etc) to the front-end (on-detector) system in a programmable way in order to facilitate various system tests. The 40.08MHz clock signal used in all LHC systems, is produced locally by a QPLL [2] ASIC configured as a local oscillator using a 160.32MHz quartz[1] [3] as a reference. Using the 40.08MHz clock, the triggering emulator generates four encoded fast commands as described in table 1.

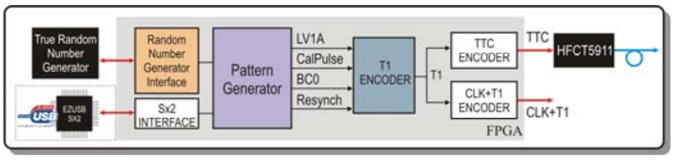

Figure 5: Block diagram of the functionality of the emulator.

The encoding scheme used (called "T1" [4]) is a 3-bit pattern, thus it requires 3 clock cycles to be transmitted serially. The first bit of the pattern is always '1' denoting the existence of a fast command and the remainder denotes which of the four fast commands is being transmitted ("00"=LV1A, "01"=BC0, "10"=Resynch, "11"= CalPulse).

The trigger emulator is developed in such way that it can generate bursts of triggers by specifying the time between consecutive triggers expressed as a number of clock cycles. Figure 6 shows how a burst of (one clock cycle width) pulses can be defined by a sequence of numbers (t1, t2, t3 … tN).

The implementation of the burst generator is based on a RAM, where the values t1, t2, t3 etc are stored. Firstly, the address pointer is reset and therefore the first value stored in the RAM (t1) appears at its output. This value is then loaded to a decrementing counter. When the counter reaches zero, the terminal count output is asserted. At the same time the address pointer of the RAM is incremented providing the next value (t2) to the decrementing counter and so on. Figure 7 shows a simplified block diagram of a burst generator based on a RAM.

The trigger emulator comprises four identical stages, one per fast command. From the four identical stages, only the one associated with the trigger generation can either be loaded from the true random number generator or defined by the user. The user has the option to mask any of the four generated fast commands.

Table 1: The encoding of the four fast commands.

Name	Description	Priority	Encoding
Resynch	Soft reset of the front-end chips	1 (high)	110
BC0	Bunch crossing zero	2	101
CalPulse	generate internal injection pulses	3	111
LV1A	Level 1 Trigger Accept	4 (low)	100

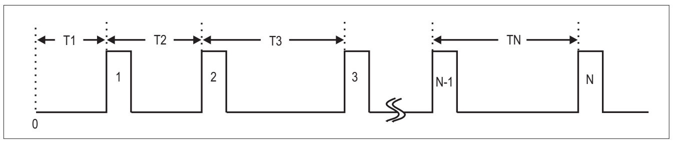

Figure 6: Programmable or random burst.

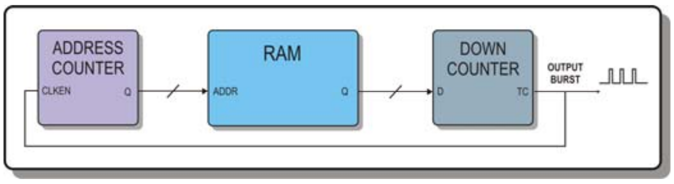

Figure 7: Programmable burst generation.

---

[1] This quartz has been specially produced for LHC applications.

The trigger emulator is able to generate bursts of up to 1024 triggers. The time between two consecutive pulses can be between 3 and 2^{32} clock cycles[2]. If the user wishes to generate bursts with N<1024 triggers, he can set the end of the burst by writing a zero (or an illegal) value to address N of the RAM.

It is worth mentioning that the emulator includes a machine that emulates the Proton and Ion bunch disposition at LHC [5]. Where there is supposed to be a gap in the disposition, the machine suppresses the (random or programmable) output of the trigger pattern generator.

The fast commands generated need to be encoded properly in order to be transmitted to the front-end electronics. Prior to the fast command encoder, a state machine is introduced to ensure the respect of the fast command priority. The encoding and priority of the signals can also be found in table 1. After the selection of the command with the higher priority, its encoding follows, as described in table 1.

In order to minimize the necessary connections for the transmission of the clock and the T1 information, an additional encoding combines the two signals into a single one. The trigger emulator implements two different methods of combining the clock encoding and the T1 information. The first method ("CLK+T1" [4]) combines the two signals by removing a pulse from the clock signal whenever the T1 signal is '1'. This is the encoding scheme used by the CMS Tracker front-end control chipset [6] and adopted by other sub-detectors. The second encoding method is based on a 160.32 MBaud bi-phase mark encoder with time-division, which multiplexes 2 channels using a balanced DC-free code. This encoding scheme is fully compatible with the scheme used for the CMS TTC system [7], although channel B which normally transmits broadcast and individually-addressed commands is currently not used. Figure 8 illustrates the TTC-like encoder implemented in the emulator.

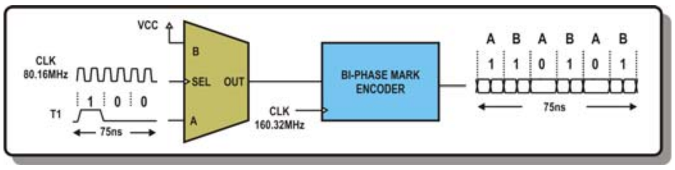

Figure 8: TTC encoding.

IV. User interface

The trigger emulator comes with a LabVIEW [8] user's interface when it works in stand-alone mode. This LabVIEW interface allows the user to define the working parameters (such as desired mean rate, number of triggers etc) of the functionality discussed in section III. The LabVIEW application includes among others, a graphical display of the fast timing signals generated by the emulator. It is worth mentioning that the application developed is compatible with both Windows and Linux operating systems.

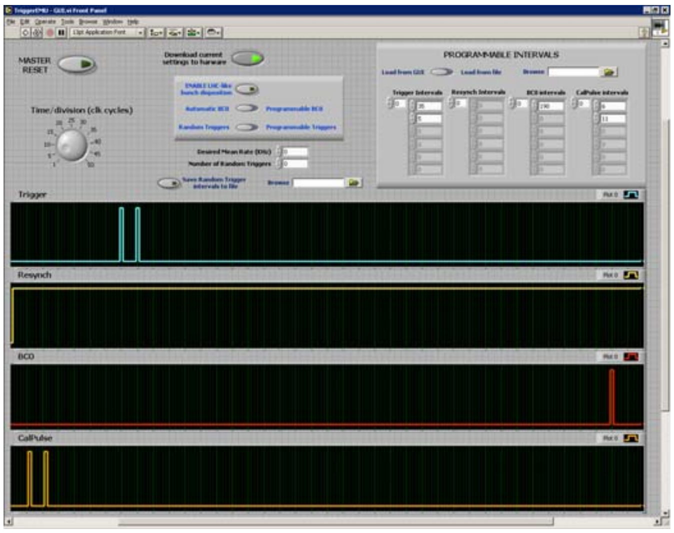

Figure 9: The front panel of the emulator's LabVIEW interface.

V. Summary and conclusion

A plug-in module able to emulate trigger and fast commands for LHC applications was built. The most important feature of this device is the ability to produce trigger patterns based on true random number generation. The source of randomness is the avalanche effect on a transistor emitter-base diode, biased by a digitally controlled voltage source.

The emulator produces randomly generated trigger pulses with time intervals following an exponential distribution with programmable mean between 1 KHz and 1 MHz (trigger rate). Therefore it can be used in emulating the arrival of particles on a detector enabling this way the testing of the corresponding on and off-detector electronic systems.

The device can either be used as a mezzanine card on various systems (VME based systems) or in stand alone mode controlled by a PC running LabVIEW.

Such a device, apart from being useful for High Energy Physics experiments, is also suitable for nuclear physics and radiology experiments where the emission of a radioactive source needs to be emulated.

VI. Acknowledgements

This work was partially supported by the Greek General Secretariat of Research and Technology and was co-funded by the European Union in the framework of the program "Heraklitos" of the "Operational Program for Education and Initial Vocational Training" of the 3rd Community Support Framework of the Hellenic Ministry of Education, funded by 25% from national sources and by 75% from the European Social Fund (ESF).

[2] In the case of CMS, values less than 3 are considered as illegal because the CMS trigger rules define that 3 LHC-clock cycles is the minimum time between two consecutive LV1A commands.

VII. REFERENCES

[1] IEEE standard 1101.1, 1998 edition.

[2] The QPLL project Web site: http://proj-qpll.web.cern.ch/proj-qpll/

[3] Micro Crystal “CC1F-T1A quartz crystal data sheet” http://www.microcrystal.com/productdocuments/variants/CC1F-T1A._4625.pdf

[4] B.G.Taylor, “Timing distribution at the LHC” presented at the 8th Workshop on electronics for LHC, 2002.

[5] P. Collier “Proton & Ion Bunch Disposition in the LHC, SPS & PS”, presented at CERN, 2003

[6] CMS Tracker Control Web site http://cmstrackercontrol.web.cern.ch /cmstrackercontrol/

[7] The TTC project Web site: http://ttc.web.cern.ch/TTC/intro.html

[8] National Instruments Corp. “LabVIEW 5.1 Function & VI Reference Manual”, 1998