Krzysztof Marek Sielewicz
krzysztof.sielewicz@cern.ch

Report on radiation testing of the FPGA

Contents

Chapter 1

Measurement of CRAM cross-section of Kintex-7 325 FPGA

1.1. Radiation facility

The Prototype Readout Unit [1], hosting the Xilinx Kintex-7 325T FPGA, was used as a platform for radiation susceptibility testing. The irradiation experiments were conducted at the isochronous cyclotron at the Nuclear Physics Institute of the Academy of Sciences of the Czech Republic in Rez, near Prague. The machine provides a proton beam with an energy range from 6 to 37 MeV. The available proton flux ranges from 10^4 to 10^{14} Hz cm^{-2}, over a uniform area of about 2.5×2.5 cm^2. A dedicated dosimetry system presented in [2] was used to scan the beam profile and monitor its intensity during the irradiation.
Beam parameters:
— Particles: protons (particles are released in bursts $T_{BURST} = \frac{1}{150\,\text{Hz}} = 6.67$ ms, inside the burst particles come at 25 MHz),
— Flux: 10^6 to $1.1 \cdot 10^8$ Hz cm^{-2},
— Beam size: $25mm \times 25mm$,
— Energy: 30 MeV (measured on the surface of the FPGA).

1.2. Experimental measurements

The FPGA was irradiated in a set of tests, where the number of errors in the configuration memory was counted. At the beginning of each test, the FPGA was programmed and then exposed to radiation for a given time. After that, the configuration memory was read back (*.RBD file) and compared against a golden readback bitfile (*.RBD file)[1] and a mask file (*.MSD file)[2]. The complete test procedure is shown in Fig. 1.1 and the results are presented in Table 1.1.

[1] Golden readback bitfile - a bitfile used for initial programming of the FPGA.
[2] Mask file - it is a file that containts the information about bits which must be excluded from the comparison, because these can correspond to user memory, such as block or distributed RAM, SRLs, or DRP memories, or null memory locations [3].

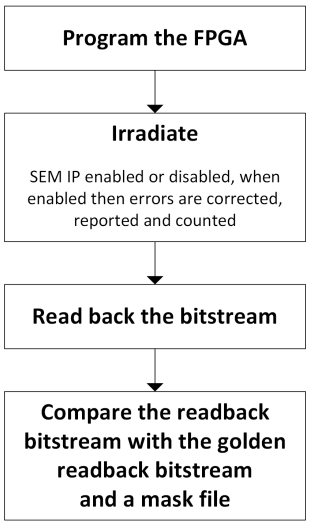


Figure 1.1: Procedure for measuring the number of errors in the configuration memory of the FPGA.

No.	Flux $\left(\mathrm{Hz\,cm^{-2}}\right)$	Time (s)	Errors in *.rbd [a]	Repaired errors [b]	Total
1	$9 \cdot 10^6$	600	1642	85	1727
2	$8.8 \cdot 10^6$	600	1301	0	1301
3	$8.8 \cdot 10^6$	120	229	15	244
4	$8.8 \cdot 10^6$	60	176	49	225
5	$8.8 \cdot 10^6$	60	235	10	245
6	$9 \cdot 10^7$	600	11965	10	11975
7	$1.05 \cdot 10^8$	600	13902	14	13916
8	$8.8 \cdot 10^6$	120	490	7	497
9	$8.65 \cdot 10^6$	600	1215	67	1282

[a] Errors found in the readback file.
[b] Errors repaired by the Xilinx Soft Error IP.

Table 1.1: Number of SEUs induced to the configuration memory of FPGA by the proton beam.

1.3. Calculation of the cross-section

In Table 1.2 the number of BRAM and CRAM bits in Kintex-7 325T is presented. The Kintex-7 325T cross-section can be calculated using formula 1.1.

$$\sigma_{CRAM} = \frac{\sum_{i=1}^{N} E_i}{N_{CB} \sum_{i=1}^{N} \Phi_i} = \frac{\sum_{i=1}^{N} E_i}{N_{CB} \sum_{i=1}^{N} F_i T_i} = \underline{3.07 \cdot 10^{-15}} \frac{cm^2}{bit} \tag{1.1}$$

Where:
E - total number of errors in each test
F - flux
T - time
$\Phi = FT$ - fluence

	Bits
BRAM	16,404,480
CRAM	75,144,416
Total	91,548,896

Table 1.2: Number of BRAM and CRAM bits in Kintex-7 325T FPGA [3,4].

The measured cross-section (using 30 MeV beam of protons) of the Kintex-7 325T equals $3.07 \cdot 10^{-15}\,\mathrm{cm}^{-2}\,\mathrm{bit}^{-1}$. This value is consistent with other tests carried out in [5–8].

Chapter 2

Testing firmware

2.1. Architecture of the testing firmware

A custom firmware has been designed for testing the operation of FPGA fabric elements and built-in block memory in a radiation environment. Its architecture is presented in Figure 2.1. The FPGA is filled up with identical modules called lanes. Each lane consists of a pattern generator, a test structure, a pattern checker, and an error counter. Tests structures are units that employ different radiation mitigation methods. Test vectors generated by the pattern checker are continuously verified at the end of the lane by the pattern checker. Error counters count the errors reported by the pattern checkers and are periodically read out over USB interface. Testing can be performed either in the beam test or by fault injection. A PC receiving the data is placed in a radiation-free area.

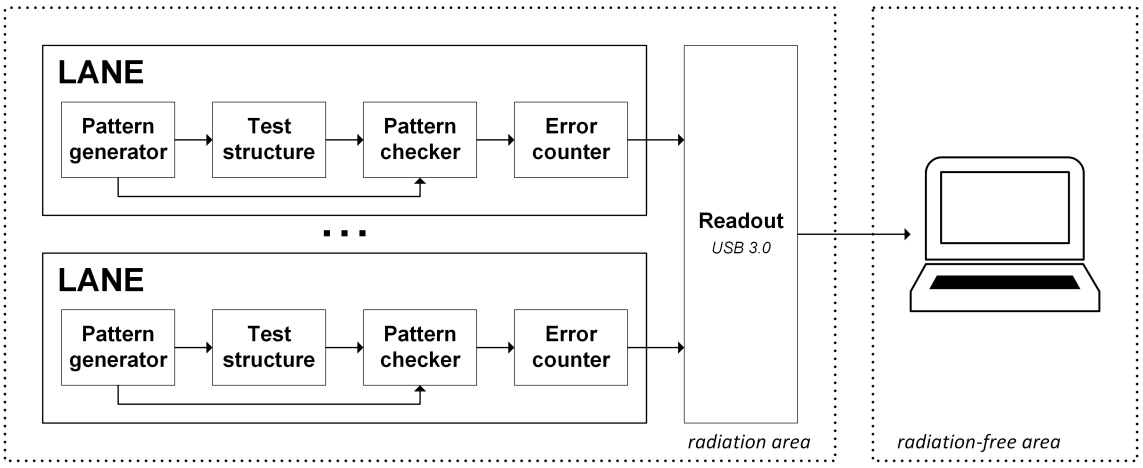


Figure 2.1: Architecture of the testing firmware.

2.2. Testing firmware of FPGA fabric elements

2.2.1. Architecture of the firmware and concept of operation

A variant of the testing firmware has been implemented to test the susceptibility to radiation of the FPGA fabric elements (combinational and sequential). It allows for studying the effectiveness of different schemes of triple modular redundancy and refreshing the configuration memory.

Its architecture is presented in Figure 2.2. It consists of many identical modules called lanes and an error readout module. Each lane consists of a triplicated pattern

generator, an array of logic test structures (called STEPs), a triplicated pattern checker and a discriminator. The logic test structure is replicated 64 times, forming an array which shifts the test vectors. The STEP consists of a hard-coded LUT transfer function (combinational logic) and an output register. Combinational logic increments the input data by 1. The pattern generator generates 6 bits[1] test vectors (from 0 to 63). After 64 clock cycles after reset, the data at the output of the array of STEPs will be a copy of the data generated by the pattern generator. The pattern checker compares the output from the array of STEPs to the output from the pattern generator. If a discrepancy is found, an error pulse is generated and an error counter is incremented. Also, when a discrepancy is found, in voting in the hardened pattern generator, pattern checker, or discriminator, a warning signal is asserted. During the test, the values stored in the error counters are periodically read out via the USB interface and saved for analysis.

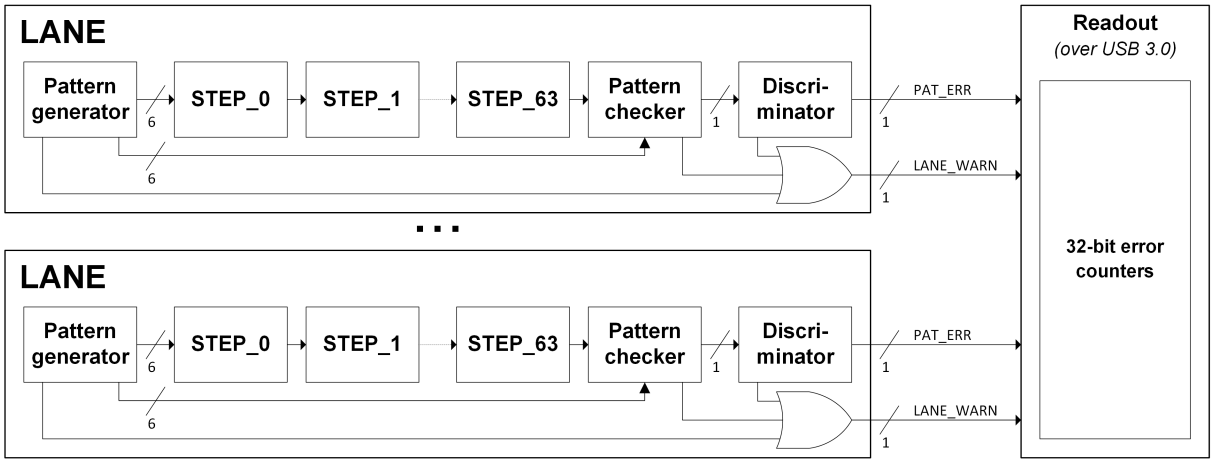


Figure 2.2: Architecture of the testing firmware with single lanes.

To test different redundancy schemes of the combinational logic and output register in the STEP, it is possible to replicate either the combinational logic, the output register, or both (Figure 2.3, 2.4, 2.5, 2.6). Also, the full array of STEPs (Figure 2.7) can be triplicated. In the STEP, it takes approximately 126 CRAM bits to configure the combinational logic, 8 to configure the output register, and 48 to configure the 6-bit voter.

The testing firmware was compiled in five different variants. In the first one, nothing was replicated. In the second variant, the output register of each STEP is triplicated and a voter selects the correct output data. In that scheme, the number of configuration bits required to configure the voter ($\approx$ 48 bits) is approximately 2 times higher than the number of bits necessary to configure the triplicated output register ($\approx$ 24 bits). In the third variant, the combinational logic is triplicated, and the correct data are selected by voting before storing in the output register. In this case, the number of configuration bits required to configure the triplicated combinational logic ($\approx$ 378 bits) is approximately 8 times higher than the number to configure the voter ($\approx$ 48 bits). In the forth variant, both the combinational logic and output register were triplicated and voted. In the last variant, each lane was triplicated and the correct data were selected by voting before comparing by the pattern checker. In this scenario, the ratio between the number of bits necessary to configure the triplicated array of STEPs ($\approx$ 24,192 bits) to the number of bits necessary to configure the voter ($\approx$ 48 bits) is the highest.

[1] A width of test pattern was set to 6 bits because the input of an LUT in Kintex-7 is 6 bits [9].

Depending on the selected redundancy scheme, the designs with 256, 160, 128 or 64 lanes have been implemented to utilize as many FPGA resources as possible, and to maximize the area susceptible to radiation. In Table 2.1, the utilization of resources by different variants of the testing firmware is presented. In the variant of the firmware where nothing was triplicated, utilization of the essential bits[2] per lane is the lowest. On the other hand, in the firmware where the full array of STEPs was triplicated, the utilization of essential bits per lane is the highest. In section 2.2.3, a cross-section of the lane in different variants is presented and a dependence between the amount of used essential bits and performance is discussed.

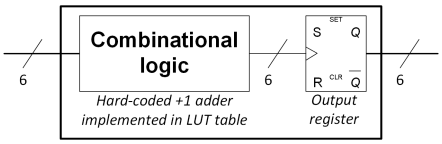


Figure 2.3: Basic STEP block design.

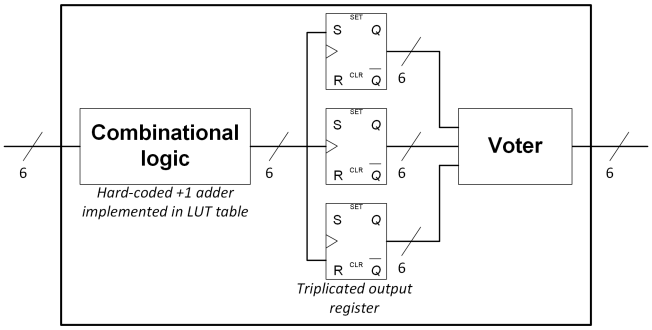


Figure 2.4: STEP block where output register is triplicated.

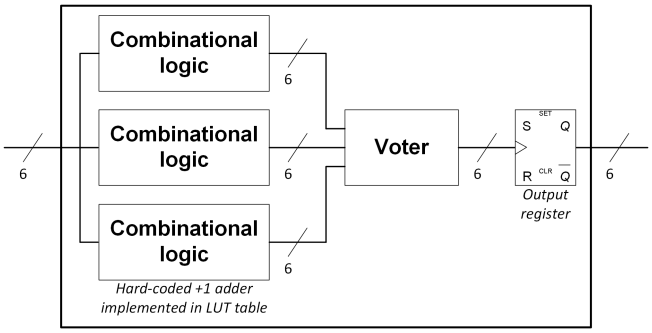


Figure 2.5: STEP block where combinational logic is triplicated.

[2] Essential bits are utilized CRAM bits that are responsible for the correct configuration of the circuit implemented in the FPGA device.

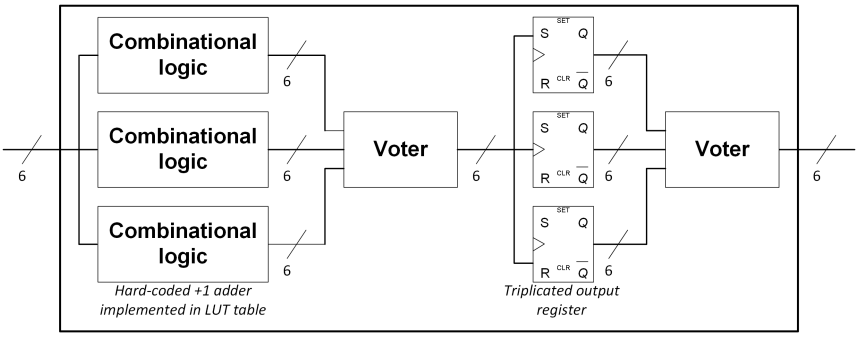


Figure 2.6: STEP block where both combinational logic and output register are triplicated.

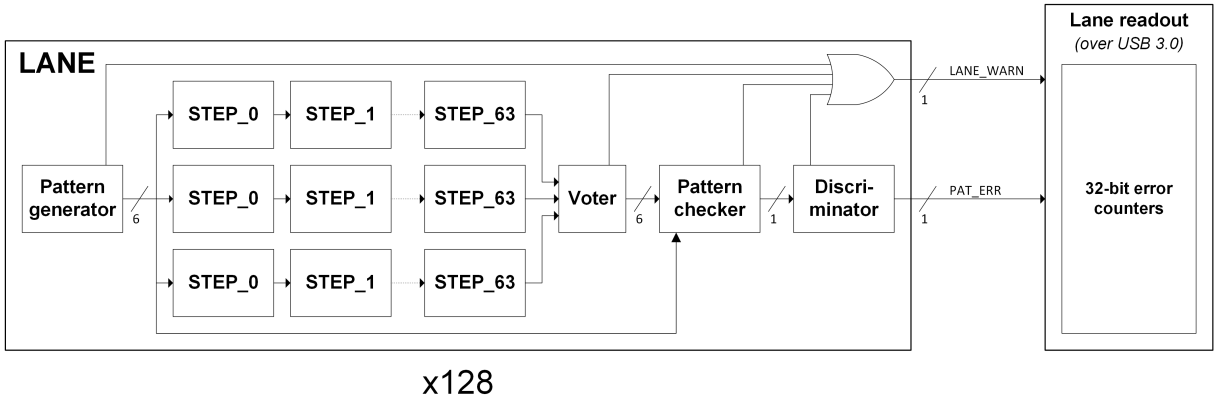


Figure 2.7: Architecture of the testing firmware with triplicated lanes.

Firmware	Lanes	Used resources per design						Essential bits	
		LUT	LUT per lane	LUT RAM	FF	FF per lane	BRAM	Essential bits per design	Essential bits per lane
Nothing triplicated	256	57%	0.22%	1%	35%	1%	25.9%	74%	0.28%
Registers triplicated	160	74%	0.46%	1%	52%	1%	37%	62.9%	0.39%
Logic triplicated	128	80%	0.63%	1%	18%	1%	32.3%	67.7%	0.53%
Logic and registers triplicated	64	80%	1.25%	1%	21%	1%	31.1%	68.8%	1.08%
Full lane triplication with voter	128	69%	0.54%	1%	42%	1%	28.7%	71.2%	0.56%

Table 2.1: Utilisation of the resources, in the Kintex-7 325T, by the testing firmware with different schemes of replication (values presented as percentages of the total amount of resources in the device).

2.2.2. Estimation of number of Single Event Upsets during irradiation and fault injection tests

Irradiation tests can be repeated in table-top tests by fault injection. Knowing the cross-section of the FPGA σ_{CRAM}, a beam intensity (flux F), an exposure time T and number of configuration bits of the CRAM memory N_{CB}, a number of SEUs induced during an irradiation test can be estimated (equation 2.1).

$$E = \sigma_{CRAM}\Phi N_{CB} = \sigma_{CRAM}FTN_{CB} \tag{2.1}$$

Then, an irradiation test can be repeated in a table-top test by injecting the same number of faults to the CRAM memory. The following is the calculation of the number of errors to inject for different fluencies used during conducted tests:

— $T = 120\,\mathrm{s},\ F = 7.4 \cdot 10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$
$\Phi = FT = 0.888 \cdot 10^9\,\mathrm{cm}^{-2}$
$E = \sigma_{CRAM}\Phi N_{CB} = 3.07 \cdot 10^{-15} \cdot 0.888 \cdot 10^9 \cdot 75144416 \approx \underline{205}$

— $T = 120\,\mathrm{s},\ F = 10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$
$\Phi = FT = 1.2 \cdot 10^9\,\mathrm{cm}^{-2}$
$E = \sigma_{CRAM}\Phi N_{CB} = 3.07 \cdot 10^{-15} \cdot 1.2 \cdot 10^9 \cdot 75144416 \approx \underline{277}$

— $T = 480\,\mathrm{s},\ F = 10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$
$\Phi = FT = 0.48 \cdot 10^9\,\mathrm{cm}^{-2}$
$E = \sigma_{CRAM}\Phi N_{CB} = 3.07 \cdot 10^{-15} \cdot 4.8 \cdot 10^8 \cdot 75144416 \approx \underline{111}$

2.2.3. Results

In this section results from beam and table-top fault injection tests are presented and discussed.

Tests without a configuration memory scrubber

Tests were carried out both in beam and table-top fault injection tests. Faults to the CRAM of the FPGA were injected using the Xilinx Soft Error Mitigation IP. The number of single event upsets induced to the CRAM memory of FPGA during irradiation was calculated and the corresponding number of faults were injected during a table-top fault injection test. Results obtained in the two tests are comparable and are presented in Table 2.2. A number of faulty lanes was measured and then the cross-section of the lane was calculated. The highest cross-section of the lane was measured for the firmware where only the STEP output register was triplicated. In this case, the ratio between the number of configuration bits necessary to configure the voter and triplicated register is the worst. The lowest cross-section of the lane was measured for the firmware where the entire array of STEPs was triplicated. There is a decrease in the cross-section by a factor of 20 between those firmwares. Figure 2.8 presents a visualisation of the operation of the logic testing firmware with fully triplicated lanes. The bottom plot presents warnings reported by the voter at the end of the lane. The upper plot shows the result of a comparison of voted data and the output from the pattern generator (no marks mean the operation was correct). Without scrubbing, errors in the CRAM are not corrected and the affected lanes stay malfunctioning until the end of the test.

			Faulty lanes			
	Fluence $\left(\mathrm{cm}^{-2}\right)$	**Lanes**	**Mean**	**Std. dev.**	**Lane** σ $\left(\mathrm{cm}^2\,\mathrm{lane}^{-1}\right)$	**Error** $\delta\sigma$ $\left(\mathrm{cm}^2\,\mathrm{lane}^{-1}\right)$
Fault injection						
Nothing triplicated	$0.89 \cdot 10^9$	256	12.08	3.75	$5.31 \cdot 10^{-11}$	$1.72 \cdot 10^{-12}$
Registers triplicated	$0.89 \cdot 10^9$	160	10.60	2.93	$7.46 \cdot 10^{-11}$	$2.12 \cdot 10^{-12}$
Logic triplicated	$0.89 \cdot 10^9$	128	3.38	1.61	$2.97 \cdot 10^{-11}$	$1.47 \cdot 10^{-12}$
Logic and registers triplicated	$1.2 \cdot 10^9$	64	1.73	1.21	$2.26 \cdot 10^{-11}$	$1.33 \cdot 10^{-12}$
Full lane triplication with voter	$1.2 \cdot 10^9$	128	0.56	0.87	$0.37 \cdot 10^{-11}$	$0.19 \cdot 10^{-12}$
Beam test measurements						
Nothing triplicated	$0.89 \cdot 10^9$	256	12.71	3.76	$5.59 \cdot 10^{-11}$	$2.10 \cdot 10^{-12}$
Registers triplicated	$0.89 \cdot 10^9$	160	11.74	3.13	$8.26 \cdot 10^{-11}$	$3.03 \cdot 10^{-12}$
Logic triplicated	$0.89 \cdot 10^9$	128	2.65	1.84	$2.33 \cdot 10^{-11}$	$2.27 \cdot 10^{-12}$
Logic and registers triplicated	$1.2 \cdot 10^9$	64	2.32	1.04	$3.02 \cdot 10^{-11}$	$2.90 \cdot 10^{-12}$
Full lane triplication with voter	$1.2 \cdot 10^9$	128	1.00	0.97	$0.65 \cdot 10^{-11}$	$1.25 \cdot 10^{-12}$

Table 2.2: Comparison of the lane cross-sections obtained from the beam and fault injection tests (a CRAM scrubber was not employed).

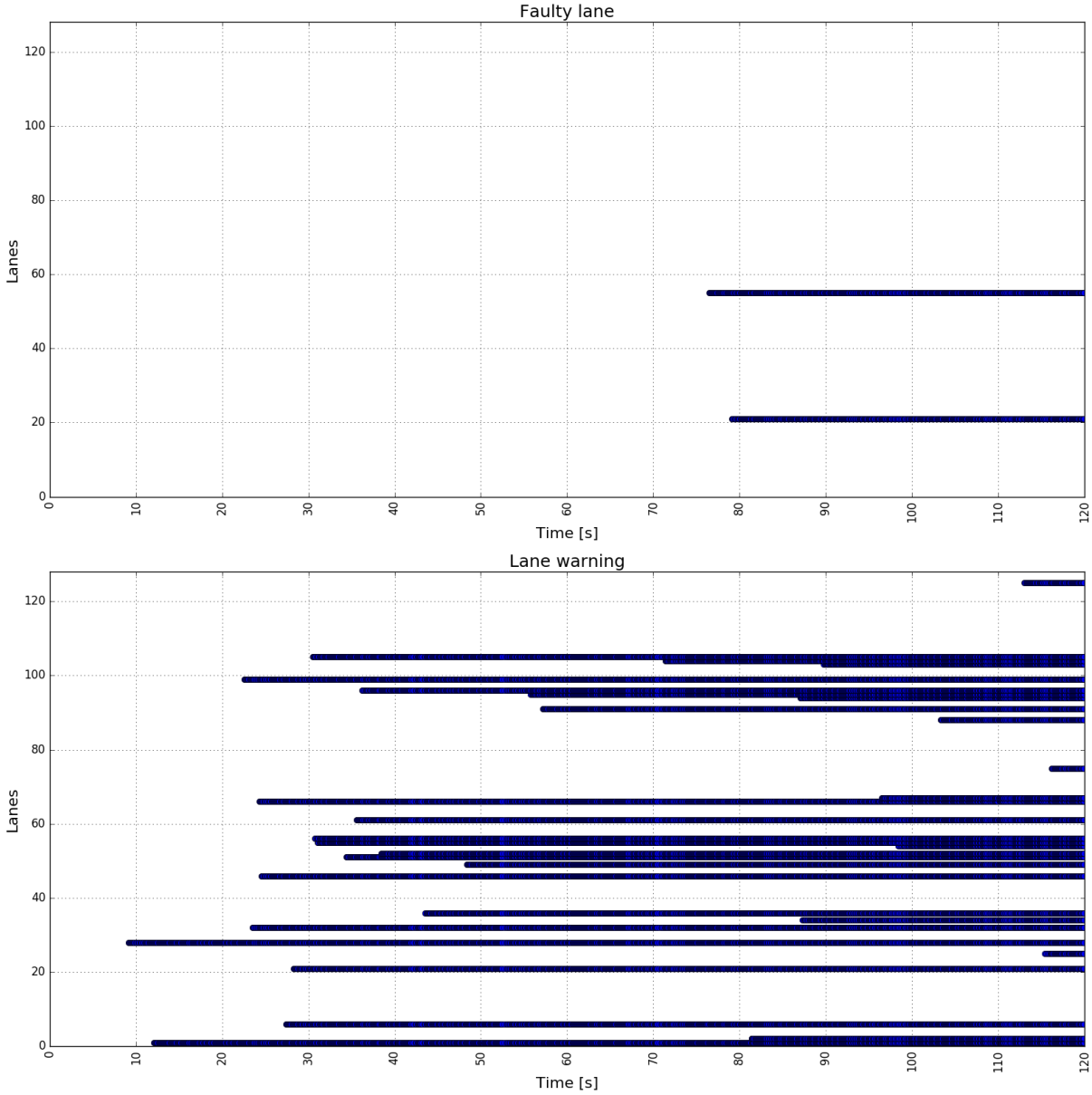


Figure 2.8: Visualisation of operation of the testing firmware with fully triplicated lanes ($T = 120\,\mathrm{s}$, $F = 10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, a CRAM scrubber was not employed).

Tests with configuration memory scrubbers

The logic testing firmware was also tested with configuration memory scrubbers. The custom active scrubber based on [10] and the JCM scubber [11] were utilized.

Tests with the custom active scrubber

During the custom active scrubber operation, all single event upsets are corrected by the Xilinx SEM IP, while multiple bit upsets are repaired by reconfiguring the entire faulty configuration frame via JTAG interface. Correction of the single-bit upset takes 610 µs [12] and repairing the multiple-bit upset (updating the entire configuration frame) takes 2 s. Thus, 2 s is the maximum time that the flipped CRAM bit remains uncorrected.

In Figure 2.9, operation of the testing firmware utilizing the basic STEP test structure (without using any radiation mitigation method) is presented. The custom active scrubber was employed. The upper plot shows how some of the lanes have got corrupted and then repaired. Also, different kinds of repaires are marked.

In Figure 2.10, another test of this same firmware is presented. An occurrence of the CRC error[3] is marked, which means that the SEM IP suspends its operation [12]. After that the CRAM bits are no longer corrected, and due to accumulation of SEUs in the configuration memory of the FPGA, more lanes start malfunctioning.

In Tables 2.3, 2.4 results of the tests with custom active scrubber are presented. In the case of the firmware with full lane triplication during all tests no errors were reported.

[3] CRC error - this is a global CRC error calculated from all configuration frames of the FPGA.

Firmware	Lanes	Flux ($\mathrm{Hz\,cm^{-2}}$)	Affected lanes		Active scrubber			
			Before[a]	After[b]	SBU[c]	MBU[d]	CRC[e]	Scrubber lifetime[f] (s)
Nothing triplicated	256							
TEST_0, run_0		10^6	5	4	80	2	1	218.97
TEST_0, run_1		$1 \cdot 10^6$	0	7	2	1	1	9.78
TEST_0, run_2		$1 \cdot 10^6$	0	9	10	0	1	34.67
TEST_0, run_3		$1 \cdot 10^6$	7	0	95	3	0	480.0
TEST_1, run_0		$8.7 \cdot 10^5$	12	0	133	5	0	480.0
TEST_1, run_1		$8.7 \cdot 10^5$	7	0	132	4	0	480.0
TEST_2, run_0		$8.7 \cdot 10^5$	3	0	116	5	1	473.74
TEST_3, run_0		$8.7 \cdot 10^5$	7	2	84	4	1	373.40
TEST_4, run_0		$8.7 \cdot 10^5$	11	0	147	2	0	480.0
Registers triplicated	160							
TEST_2, run_0		$1 \cdot 10^6$	3	10	51	5	1	145.28
TEST_3, run_0		$1 \cdot 10^6$	10	2	175	2	1	475.39
TEST_3, run_1		$1 \cdot 10^6$	14	0	176	3	0	480.0
TEST_4, run_0		$1 \cdot 10^6$	11	0	144	2	0	480.0
TEST_5, run_0		$8.7 \cdot 10^5$	6	0	114	6	0	480.0
TEST_6, run_0		$8.7 \cdot 10^5$	9	0	110	6	0	480.0
Logic triplicated	128							
TEST_1, run_0		$1 \cdot 10^6$	2	0	166	3	0	480.0
TEST_2, run_0		$1 \cdot 10^6$	2	0	159	3	0	480.0
TEST_3, run_0		$1 \cdot 10^6$	3	1	98	2	1	312.16
TEST_4, run_0		$1 \cdot 10^6$	2	1	168	7	0	480.0
TEST_6, run_0		$8.7 \cdot 10^5$	3	0	100	4	0	480.0
Logic and registers triplicated	64							
TEST_9, run_0		$9.7 \cdot 10^5$	2	0	143	1	0	480.0
TEST_10, run_0		$1 \cdot 10^6$	1	0	155	3	0	480.0
TEST_11, run_0		$8.7 \cdot 10^5$	0	0	115	2	0	480.0
TEST_12, run_0		$8.7 \cdot 10^5$	1	0	121	1	0	480.0

[a] Before SEM IP reported a CRC error.
[b] After SEM IP reported a CRC error.
[c] Single Event Upset (number of SBUs before SEM IP reported CRC error).
[d] Multiple Bit Upset (number of MBUs before SEM IP reported CRC error).
[e] Cyclic Redundancy Check error reported by SEM IP.
[f] Time after which SEM IP reported CRC error and suspended its operation.

Table 2.3: Data obtained during beam tests with active scrubber employed (run duration 480 s).

Firmware	Lanes	Flux ($\mathrm{Hz\,cm^{-2}}$)	Affected lanes: Before[a]	Affected lanes: After[b]	Active scrubber: SBU[c]	Active scrubber: MBU[d]	Active scrubber: CRC[e]	Active scrubber: Scrubber lifetime[f] (s)
Full lane triplication	128							
TEST_1, run_0		$9.7 \cdot 10^5$	0	0	135	6	0	480.0
TEST_2, run_0		$1 \cdot 10^6$	0	0	85	2	1	302.17
TEST_4, run_0		$1 \cdot 10^6$	0	0	60	0	1	246.43
TEST_5, run_0		$8.7 \cdot 10^5$	0	0	110	3	0	480.0
TEST_5, run_1		$8.7 \cdot 10^5$	0	0	125	2	0	480.0
TEST_6, run_0		$8.7 \cdot 10^5$	0	0	113	3	0	480.0
TEST_6, run_1		$8.7 \cdot 10^5$	0	0	134	2	1	380.13
TEST_1, run_0		$1 \cdot 10^6$	0	0	171	10	0	480
TEST_1, run_1		$1 \cdot 10^6$	0	0	178	6	0	480
TEST_1, run_2		$1 \cdot 10^6$	0	0	128	3	1	377.17
TEST_2, run_0		$1 \cdot 10^6$	0	0	171	6	0	480
TEST_2, run_1		$1 \cdot 10^6$	0	0	143	2	0	480
TEST_2, run_2		$1 \cdot 10^6$	0	0	130	2	1	385.60
TEST_3, run_0		$1 \cdot 10^6$	0	0	169	10	0	480
TEST_3, run_1		$1 \cdot 10^6$	0	0	180	4	0	480
TEST_3, run_2		$1 \cdot 10^6$	0	0	151	11	0	480
TEST_3, run_3		$1 \cdot 10^6$	0	0	30	0	1	78.38

[a] Before SEM IP reported a CRC error.
[b] After SEM IP reported a CRC error.
[c] Single Event Upset (number of SBUs before SEM IP reported CRC error).
[d] Multiple Bit Upset (number of MBUs before SEM IP reported CRC error).
[e] Cyclic Redundancy Check error reported by SEM IP.
[f] Time after which SEM IP reported CRC error and suspended its operation.

Table 2.4: Data obtained during beam tests with active scrubber employed (run duration 480 s) (cont.).

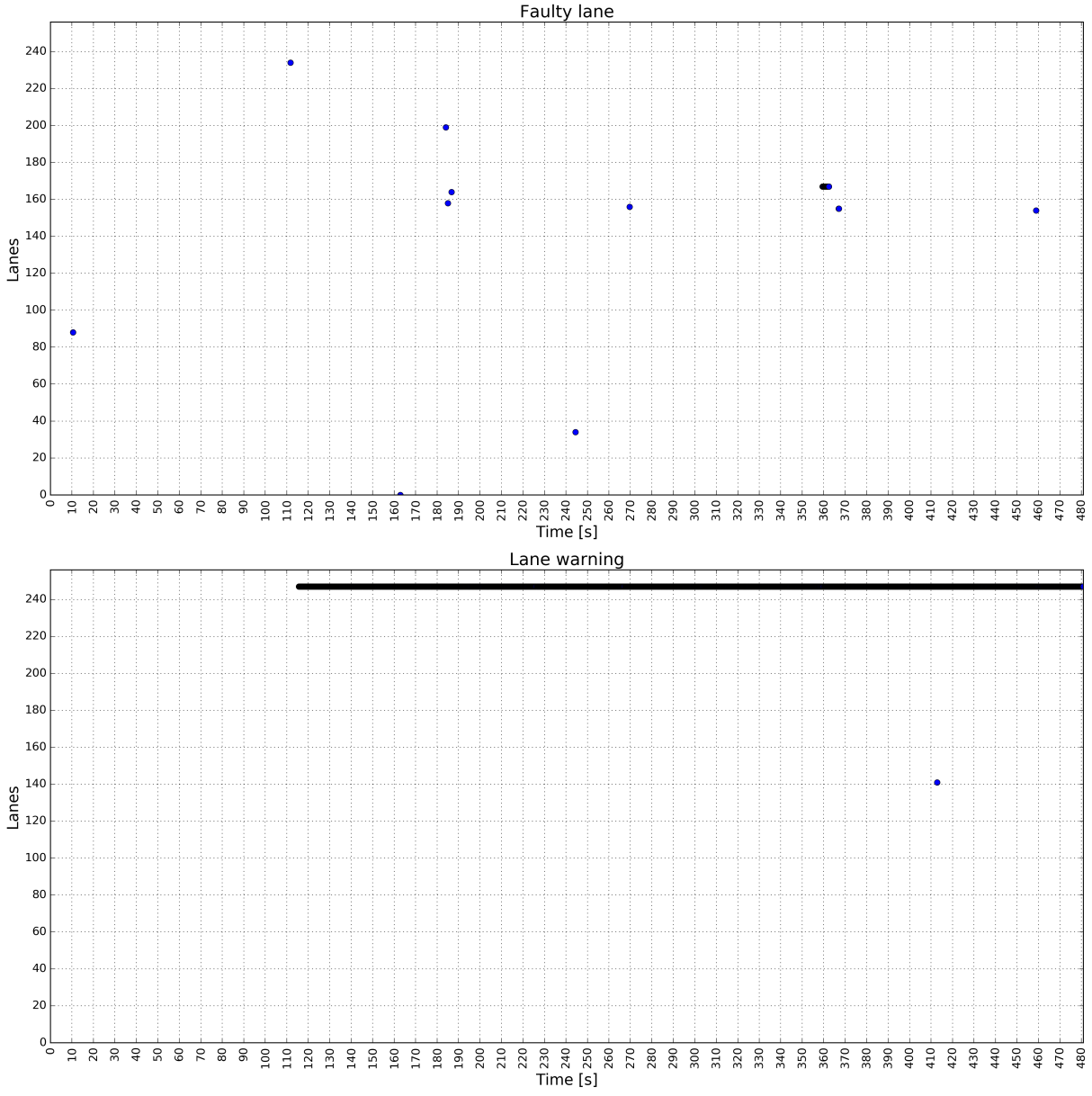


Figure 2.9: Visualisation of operation of the testing firmware utilizing basic STEP test structure with a custom active scrubber employed ($T = 480\,\mathrm{s}$, $F = 10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$).

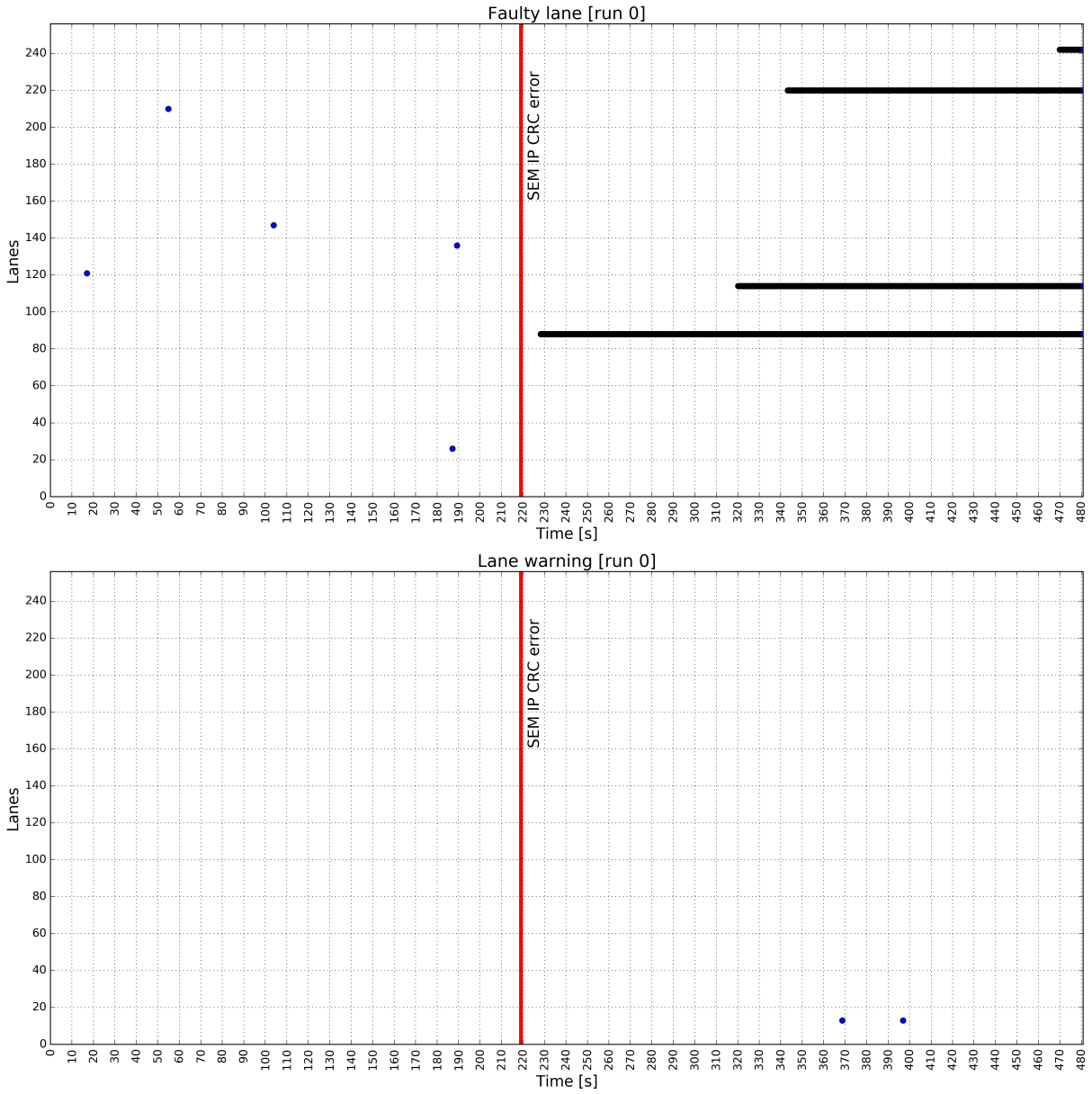


Figure 2.10: Visualisation of operation of the testing firmware utilizing basic STEP test structure with a custom active scrubber employed. Xilinx SEM IP reports a CRC error and suspends its operation ($T = 480\,\mathrm{s}$, $F = 10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$).

Tests with JCM scrubber

In Table 2.5, data obtained during irradiation tests are presented. The JCM scrubber was utilized to blindly scrub the CRAM of the FPGA. One JCM scrub cycle takes approximately 2 s. Thus, 2 s is the maximum time that the flipped CRAM bit remains uncorrected. During all the tests, no errors were reported for the firmware with full lane triplication. In Figure 2.11, a visualisation of operation of the testing firmware with a full lane triplication is presented. The JCM scrubber was employed. The test was 480 s long and the flux was set to $10^6\,\mathrm{p\,cm^{-2}\,s^{-1}}$. During this test, no errors were reported. The test was also repeated for a higher flux of $10^7\,\mathrm{p\,cm^{-2}\,s^{-1}}$. Again, no errors were reported (Figure 2.12).

Firmware	**Lanes**	**Flux** $(\mathrm{Hz\,cm^{-2}})$	**Time** (s)	**Fluence** $(\mathrm{cm^{-2}})$	**Affected lanes**
Nothing triplicated	256				
TEST_0, run_0		$1 \cdot 10^6$	480	$0.48 \cdot 10^9$	10
TEST_1, run_0		$1 \cdot 10^6$	480	$0.48 \cdot 10^9$	12
TEST_1, run_1		$1 \cdot 10^6$	480	$0.48 \cdot 10^9$	10
TEST_1, run_2		$1 \cdot 10^6$	480	$0.48 \cdot 10^9$	11
TEST_2, run_0		$1.02 \cdot 10^6$	480	$0.49 \cdot 10^9$	19
TEST_2, run_1		$1.02 \cdot 10^6$	480	$0.49 \cdot 10^9$	14
TEST_2, run_2		$1.02 \cdot 10^6$	480	$0.49 \cdot 10^9$	14
TEST_2, run_3		$1.02 \cdot 10^6$	480	$0.49 \cdot 10^9$	13
TEST_2, run_4		$1.02 \cdot 10^6$	480	$0.49 \cdot 10^9$	11
TEST_3, run_0		$1.05 \cdot 10^6$	480	$0.5 \cdot 10^9$	17
Full lane triplication	128				
TEST_0, run_0		$1.15 \cdot 10^6$	480	$0.55 \cdot 10^9$	0
TEST_0, run_1		$1.15 \cdot 10^6$	480	$0.55 \cdot 10^9$	0
TEST_0, run_2		$1.15 \cdot 10^6$	480	$0.55 \cdot 10^9$	0
TEST_0, run_3		$1.15 \cdot 10^6$	480	$0.55 \cdot 10^9$	0
TEST_1, run_0		$9.5 \cdot 10^5$	480	$0.46 \cdot 10^9$	0
TEST_1, run_1		$9.5 \cdot 10^5$	480	$0.46 \cdot 10^9$	0
TEST_1, run_2		$9.5 \cdot 10^5$	480	$0.46 \cdot 10^9$	0
TEST_1, run_3		$9.5 \cdot 10^5$	480	$0.46 \cdot 10^9$	0
TEST_1, run_4		$9.5 \cdot 10^5$	480	$0.46 \cdot 10^9$	0
TEST_2, run_0		$1 \cdot 10^7$	240	$2.4 \cdot 10^9$	0
TEST_2, run_1		$1 \cdot 10^7$	240	$2.4 \cdot 10^9$	0
TEST_2, run_2		$1 \cdot 10^7$	240	$2.4 \cdot 10^9$	0
TEST_3, run_0		$1 \cdot 10^7$	240	$2.4 \cdot 10^9$	0
TEST_3, run_1		$1 \cdot 10^7$	240	$2.4 \cdot 10^9$	0

Table 2.5: Data obtained during beam tests with JCM scrubber employed.

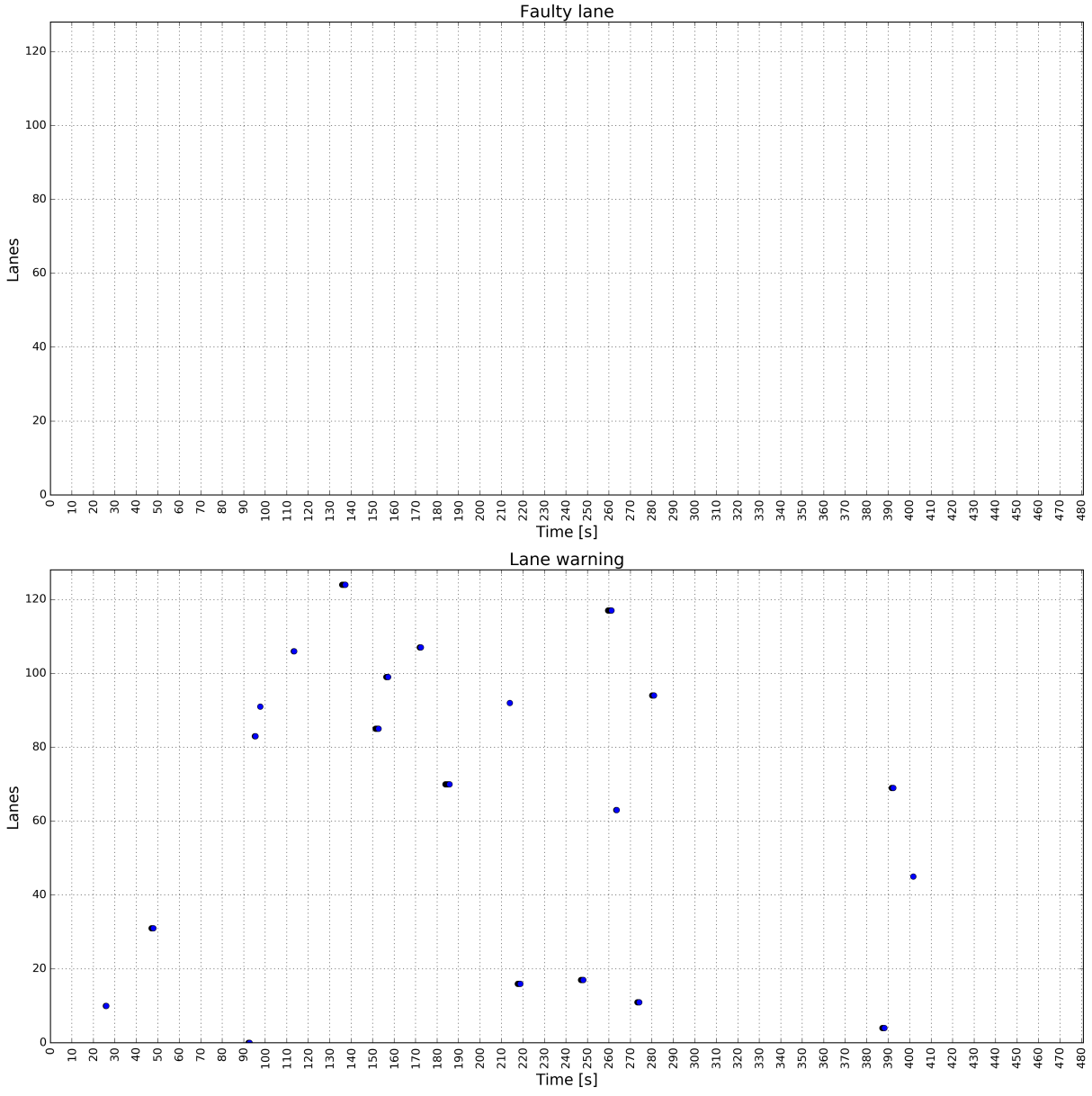


Figure 2.11: Visualisation of operation of the testing firmware with a full lane triplication with the JCM scrubber employed ($T = 480\,\mathrm{s}$, $F = 10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$).

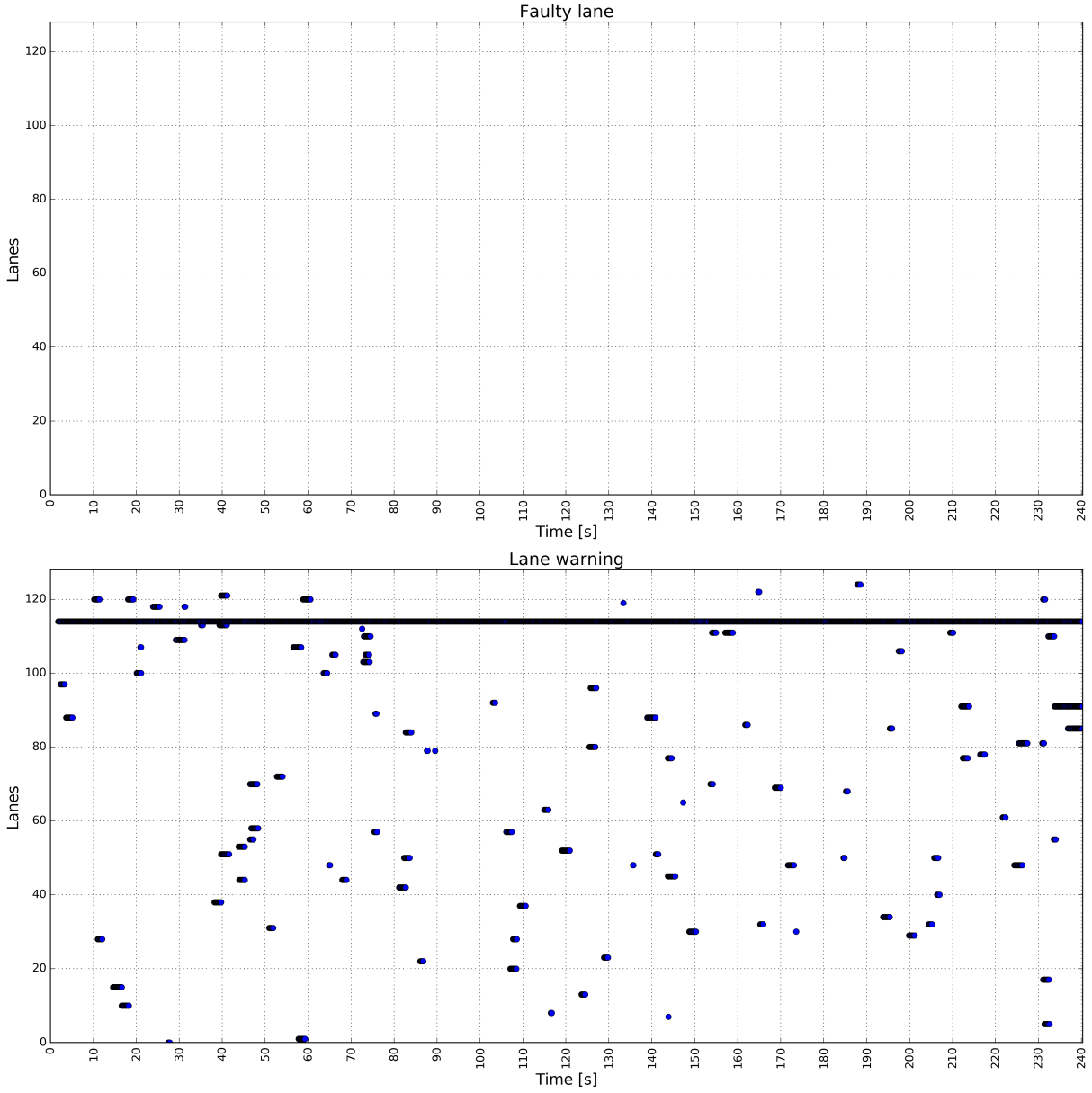


Figure 2.12: Visualisation of operation of the testing firmware with a full lane triplication with the JCM scrubber employed ($T = 480\,\mathrm{s}$, $F = 10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$).

2.3. FIFO testing firmware

2.3.1. Architecture of the firmware

A variant of the testing firmware has been implemented to test the susceptibility to radiation of the FPGA's built-in block memory and the Hamming Error Injection and Correction Checking mechanism [13]. The firmware consists of replicated lanes. In each lane there are the triplicated data generator, the FIFO test structure, the data checker, and the error counter. The data checker generates 32-bit vectors which are written to the FIFO test structure. Then, the data checker reads data from it, and subtracts two consecutive vectors from each other. If the result of the subtraction is different than 1, then an error pulse is generated, and the corresponding error counter is incremented. Also, outputs from ECC mechanism informing about single bit upset or double bit upset are monitored. Error counters are periodically read out over USB interface.

FIFO_192 firmware

In Figure 2.13 achitecture of the testing firmware, with 192 test lanes, is presented. Each lane utilises a single FIFO consisting of two 36 kbit BRAM blocks. Usage of resources is shown in Table 2.6. *SBITERR* and *DBITERR* signals are connected to a single OR gate which output is read out by a single error counter. During beam or fault injection tests the following errors are expected:

— the single-bit BRAM error repaired by the ECC mechanism (I) - only while testing in radiation environment,
— the multi-bit BRAM error not repaired by the ECC mechanism (II) - only while testing in radiation environment,
— the error in the readout counter of the ECC mechanism (III),
— the error in the output FIFO routing (IV),
— the error in the input FIFO routing (V),
— the error in the comparator counter (VI).

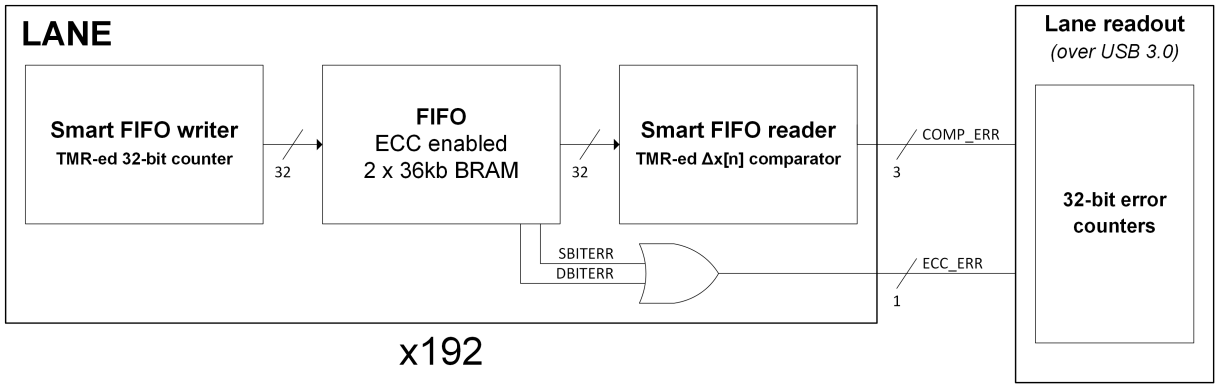


Figure 2.13: Architecture of the FIFO_192 testing firmware.

FIFO_4 firmware

In Figure 2.14 architecture of the testing firmware, with 4 lanes, is presented. Each lane utilises a chain of 48 FIFOs, each consisting of two 36 kbit BRAM blocks. Usage of resources is shown in Table 2.6. *SBITERR* and *DBITERR* signals of each FIFO are

connected to a error counter. Also, the warning signal signalising the error while voting in the data generator (Smart FIFO writer) is monitored.

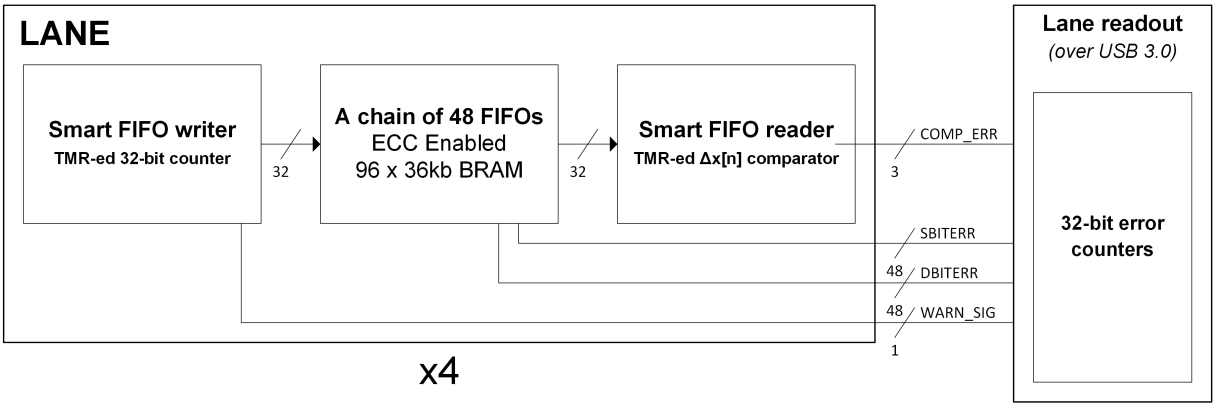


Figure 2.14: Architecture of the FIFO_4 testing firmware.

Firmware	Lanes	Used resources per design						Essential bits	
		LUT	LUT per lane	LUT RAM	FF	FF per lane	BRAM	Essential bits per design	Essential bits per lane
FIFO_576	192	68%	0.36%	1%	42%	0.22%	88%	37%	0.19%
FIFO chain	4	18%	4.5%	1%	17%	4.25%	88%	15.1%	3.75%

Table 2.6: Utilisation of the resources, in the Kintex-7 325T, by the FIFO testing firmware (values presented as percentages of the total amount of resources in the device).

2.3.2. Results

Built-in block memory and the Hamming Error Injection and Correction Checking mechanism have been tested without a CRAM scrubber, with the custom active scrubber, and the JCM scrubber. Tests were carried out with 30 MeV proton beam at the isochronous cyclotron located at the Nuclear Physics Institute of the Academy of Sciences of the Czech Republic in Řež near Prague.

The Kintex-7 BRAM cross-section was measured using formula 2.2.

$$\sigma_{BRAM} = \frac{\sum_{i=1}^{N} E_i}{N_{BB} \sum_{i=1}^{N} \Phi_i} = \frac{\sum_{i=1}^{N} E_i}{N_{BB} \sum_{i=1}^{N} F_i T_i} = \underline{5.07 \cdot 10^{-15}}\ \frac{cm^2}{bit} \tag{2.2}$$

Where:
E - total number of errors in each test
F - flux
T - time

$\Phi = FT$ - fluence
$N_{BB} = 2 \cdot 36 \cdot 1024 \cdot 48 \cdot 4 = 14155776$ - number of used BRAM bits

FIFO_192 firmware

In Table 2.7 data obtained during beam test for the FIFO_192 testing firmware are presented. A CRAM scrubber was not employed. Each run was 120 s and the flux was set to 10^7 Hz cm^{-2}.

In Table 2.8 data obtained during beam test of the FIFO_192 testing firmware with active scrubber employed are presented. Each run was 480 s and the flux was set to 10^6 Hz cm^{-2}.

In Table 2.9 data obtained during beam test of the FIFO_192 testing firmware with JCM scrubber employed are presented. Tests were carried out with the following parameters 480 s and 10^6 Hz cm^{-2} or 240 s and 10^7 Hz cm^{-2}.

FIFO_4 firmware

In Table 2.10 data obtained during beam test for the FIFO_4 testing firmware are presented. A CRAM scrubber was not employed. Each run was 120 s and the flux was set to 10^7 Hz cm^{-2}.

In Table 2.11 data obtained during beam test of the FIFO_4 testing firmware with active scrubber employed are presented. Each run was 480 s and the flux was set to 10^6 Hz cm^{-2}.

In Table 2.12 data obtained during beam test of the FIFO_4 testing firmware with JCM scrubber employed are presented. Tests were carried out with the following parameters 480 s and 10^6 Hz cm^{-2} or 240 s and 10^7 Hz cm^{-2}.

Test	Flux ($\mathrm{Hz\,cm^{-2}}$)	Time (s)	Errors				
			I[a]	II[b]	III[c]	IV[d]	V[e]
TEST_0, run_0	$1 \cdot 10^7$	120	90	0	0	5	6
TEST_3, run_2	$1 \cdot 10^7$	120	85	0	1	12	7
TEST_5, run_0	$1 \cdot 10^7$	120	94	0	0	6	14
TEST_7, run_0	$1 \cdot 10^7$	120	87	0	0	9	7
TEST_7, run_1	$1 \cdot 10^7$	120	82	0	0	7	10
TEST_8, run_0	$1 \cdot 10^7$	120	85	0	1	1	10
TEST_8, run_1	$1 \cdot 10^7$	120	80	0	0	6	15
TEST_9, run_1	$1 \cdot 10^7$	120	96	0	0	9	1
TEST_10, run_1	$1 \cdot 10^7$	120	78	0	1	6	10
TEST_11, run_0	$1 \cdot 10^7$	120	89	0	0	5	9
TEST_13, run_2	$1 \cdot 10^7$	120	75	0	1	4	8
TEST_13, run_4	$1 \cdot 10^7$	120	77	0	0	5	6
TEST_14, run_4	$1 \cdot 10^7$	120	87	0	0	9	7

[a] The single-bit BRAM error repaired by the ECC mechanism
[b] The multi-bit BRAM error not repaired by the ECC mechanism
[c] The error in the readout counter of the ECC mechanism
[d] The error in the FIFO routing
[e] The error in the comparator counter

Table 2.7: Data obtained during beam test for the FIFO_192 testing firmware (a CRAM scrubber was not employed).

Test	Flux	Time	Errors					Active scrubber			
			I[a]	II[b]	III[c]	IV[d]	V[e]	SBU[f]	MBU[g]	CRC[h]	Scrubber lifetime[i]
	$(\mathrm{Hz\,cm^{-2}})$	(s)									(s)
TEST_0, run_0	$1 \cdot 10^6$	480	44	0	0	3	3	155	5	0	480
TEST_0, run_1	$1 \cdot 10^6$	480	34	0	2	7	3	157	1	0	480
TEST_0, run_2	$1 \cdot 10^6$	480	38	0	0	5	1	179	2	0	480
TEST_0, run_3	$1 \cdot 10^6$	480	32	0	0	3	1	128	0	0	480
TEST_0, run_4	$1 \cdot 10^6$	480	44	0	0	5	3	132	11	0	480
TEST_0, run_5	$1 \cdot 10^6$	480	34	0	0	5	6	88	1	0	480
TEST_0, run_6	$1 \cdot 10^6$	480	39	0	0	4	3	155	4	0	480
TEST_0, run_7	$1 \cdot 10^6$	480	36	0	0	4	3	170	1	0	480
TEST_0, run_8	$1 \cdot 10^6$	480	34	0	0	3	0	157	2	0	480
TEST_1, run_1	$1 \cdot 10^6$	480	53	0	0	4	5	157	3	0	480
TEST_1, run_2	$1 \cdot 10^6$	480	31	0	0	3	1	151	1	0	480
TEST_1, run_3	$1 \cdot 10^6$	480	46	0	0	1	4	74	1	1	259.39
TEST_1, run_4	$1 \cdot 10^6$	480	44	0	0	1	5	132	5	0	480
TEST_2, run_0	$1 \cdot 10^6$	480	32	0	0	4	2	158	3	0	480
TEST_2, run_1	$1 \cdot 10^6$	480	56	0	0	5	2	80	3	1	232.74
TEST_2, run_2	$1 \cdot 10^6$	480	41	0	0	1	3	145	0	0	480
TEST_2, run_3	$1 \cdot 10^6$	480	31	0	0	0	3	57	3	1	163.42

[a] The single-bit BRAM error repaired by the ECC mechanism
[b] The multi-bit BRAM error not repaired by the ECC mechanism
[c] The error in the readout counter of the ECC mechanism
[d] The error in the FIFO routing
[e] The error in the comparator counter
[f] Single Event Upset (number of SBUs before SEM IP reported CRC error).
[g] Multiple Bit Upset (number of MBUs before SEM IP reported CRC error).
[h] Cyclic Redundancy Check error reported by SEM IP.
[i] Time after which SEM IP reported CRC error and suspended its operation.

Table 2.8: Data obtained during beam test of the FIFO_192 testing firmware with active scrubber employed.

Test	Flux ($\mathrm{Hz\,cm^{-2}}$)	Time (s)	Errors				
			I[a]	II[b]	III[c]	IV[d]	V[e]
TEST_3, run_0	$1 \cdot 10^6$	480	41	0	0	6	5
TEST_3, run_1	$1 \cdot 10^6$	480	40	0	1	4	5
TEST_3, run_2	$1 \cdot 10^6$	480	45	0	0	3	5
TEST_3, run_3	$1 \cdot 10^6$	480	33	0	0	2	4
TEST_3, run_4	$1 \cdot 10^6$	480	40	0	0	4	4
TEST_3, run_5	$1 \cdot 10^6$	480	37	0	0	1	2
TEST_3, run_6	$1 \cdot 10^6$	480	38	0	0	2	1
TEST_3, run_7	$1 \cdot 10^6$	480	36	0	0	3	5
TEST_3, run_8	$1 \cdot 10^6$	480	37	0	0	1	4
TEST_3, run_9	$1 \cdot 10^6$	480	35	0	0	3	1
TEST_4, run_1	$1 \cdot 10^7$	240	169	0	2	8	9
TEST_4, run_2	$1 \cdot 10^7$	240	161	0	1	9	10

[a] The single-bit BRAM error repaired by the ECC mechanism
[b] The multi-bit BRAM error not repaired by the ECC mechanism
[c] The error in the readout counter of the ECC mechanism
[d] The error in the FIFO routing
[e] The error in the comparator counter

Table 2.9: Data obtained during beam test of the FIFO_192 testing firmware with JCM scrubber employed.

Test	Flux $(\mathrm{Hz\,cm^{-2}})$	Time (s)	Errors I[a]	Errors II[b]
TEST_2, run_0	$1 \cdot 10^7$	120	91	0
TEST_2, run_1	$1 \cdot 10^7$	120	72	0
TEST_2, run_2	$1 \cdot 10^7$	120	77	0
TEST_2, run_3	$1 \cdot 10^7$	120	80	0
TEST_2, run_4	$1 \cdot 10^7$	120	82	0
TEST_2, run_8	$1 \cdot 10^7$	120	71	0
TEST_2, run_9	$1 \cdot 10^7$	120	68	0
TEST_2, run_10	$1 \cdot 10^7$	120	83	0
TEST_2, run_13	$1 \cdot 10^7$	120	65	0
TEST_2, run_14	$1 \cdot 10^7$	120	43	0
TEST_2, run_16	$1 \cdot 10^7$	120	79	0
TEST_2, run_17	$1 \cdot 10^7$	120	100	0
TEST_2, run_19	$1 \cdot 10^7$	120	96	0
TEST_2, run_20	$1 \cdot 10^7$	120	96	0
TEST_2, run_21	$1 \cdot 10^7$	120	88	0
TEST_2, run_22	$1 \cdot 10^7$	120	85	0
TEST_2, run_23	$1 \cdot 10^7$	120	74	0
TEST_2, run_24	$1 \cdot 10^7$	120	68	0

[a] The single-bit BRAM error repaired by the ECC mechanism
[b] The multi-bit BRAM error not repaired by the ECC mechanism

Table 2.10: Data obtained during beam test for the FIFO chain testing firmware without CRAM scrubber employed.

Test	Flux	Time	Errors		Active scrubber			
			I[a]	II[b]	SBU[c]	MBU[d]	CRC[e]	Scrubber lifetime[f]
	$(\mathrm{Hz\,cm^{-2}})$	(s)						(s)
TEST_0, run_0	$1 \cdot 10^6$	480	39	0	154	7	0	480
TEST_0, run_1	$1 \cdot 10^6$	480	40	0	179	10	0	480
TEST_0, run_2	$1 \cdot 10^6$	480	40	0	111	6	1	337.54
TEST_0, run_3	$1 \cdot 10^6$	480	34	0	83	2	1	227.69
TEST_0, run_5	$1 \cdot 10^6$	480	37	0	21	3	1	41.87
TEST_0, run_6	$1 \cdot 10^6$	480	28	0	164	8	0	480
TEST_0, run_7	$1 \cdot 10^6$	480	45	0	141	7	0	480
TEST_0, run_8	$1 \cdot 10^6$	480	30	0	165	12	0	480
TEST_0, run_9	$1 \cdot 10^6$	480	38	0	156	4	0	480
TEST_0, run_10	$1 \cdot 10^6$	480	29	0	174	6	0	480
TEST_0, run_11	$1 \cdot 10^6$	480	34	0	156	0	0	480
TEST_0, run_12	$1 \cdot 10^6$	480	42	0	115	7	1	298.97
TEST_0, run_13	$1 \cdot 10^6$	480	41	0	40	0	0	480
TEST_0, run_14	$1 \cdot 10^6$	480	33	0	148	4	0	480

[a] The single-bit BRAM error repaired by the ECC mechanism
[b] The multi-bit BRAM error not repaired by the ECC mechanism
[c] Single Event Upset (number of SBUs before SEM IP reported CRC error).
[d] Multiple Bit Upset (number of MBUs before SEM IP reported CRC error).
[e] Cyclic Redundancy Check error reported by SEM IP.
[f] Time after which SEM IP reported CRC error and suspended its operation.

Table 2.11: Data obtained during beam test of the FIFO chain testing firmware with active scrubber employed.

Test	Flux $(\mathrm{Hz\,cm^{-2}})$	Time (s)	Errors I[a]	Errors II[b]
TEST_1, run_0	$1 \cdot 10^6$	480	32	0
TEST_1, run_1	$1 \cdot 10^6$	480	45	0
TEST_1, run_2	$1 \cdot 10^6$	480	27	0
TEST_1, run_3	$1 \cdot 10^6$	480	35	0
TEST_1, run_4	$1 \cdot 10^6$	480	29	0
TEST_1, run_5	$1 \cdot 10^6$	480	33	0
TEST_1, run_6	$1 \cdot 10^6$	480	35	0
TEST_1, run_7	$1 \cdot 10^6$	480	27	0
TEST_1, run_8	$1 \cdot 10^6$	480	31	0
TEST_1, run_9	$1 \cdot 10^6$	480	42	0
TEST_2, run_0	$1 \cdot 10^7$	120	85	0
TEST_2, run_0	$1 \cdot 10^7$	120	75	0
TEST_2, run_0	$1 \cdot 10^7$	120	88	0
TEST_3, run_1	$1 \cdot 10^7$	120	116	0

[a] The single-bit BRAM error repaired by the ECC mechanism
[b] The multi-bit BRAM error not repaired by the ECC mechanism

Table 2.12: Data obtained during beam test for the FIFO chain testing firmware with JCM scubber employed.

Chapter 3

CRAM scrubbers

3.1. Custom active scrubber

The custom active scrubber (Fig. 3.1), based on [10], consists of the Xilinx Soft Error IP [12] and a dedicated Python-based software running on the PC. The SEM IP operates in the repair mode which means that it corrects configuration memory frames with single-bit errors. It also covers correction of multi-bit upset events when errors are distributed one per frame as a result of configuration memory interleaving.

The software continuously processes the output data sent by the SEM IP over UART interface. All the single bit errors that are reported by the SEM IP (Listing 3.1), are logged. Before operating the custom active scrubber, partial bitfiles must be generated. These are bitfiles (Fig. 3.2 (b)) which contain only configuration data for one configuration frame. When a multiple-bit bit error is detected, then based on the logical address in the SEM IP output message (Listing 3.2), a correct partial bitfile is fetched. Next, the physical address (PA) corresponding to the logical address (LA) is written to the frame address register (FAR) in the partial bitfile. Finally, the edited file is written to the FPGA over JTAG interface using Xilinx Impact configuration tool and the programming script presented on Listing 3.3.

Correction of the single-bit upset takes 610 µs [12] and repairing the multiple-bit upset (updating the entire configuration frame) takes 2 s. Thus, 2 s is the maximum time that the flipped CRAM bit remains uncorrected.

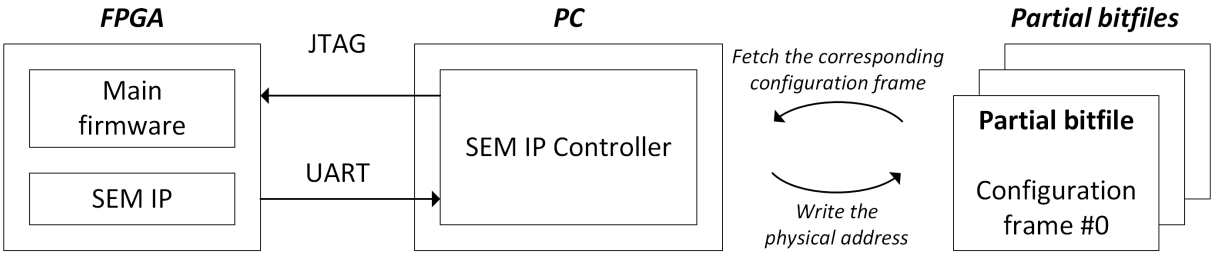
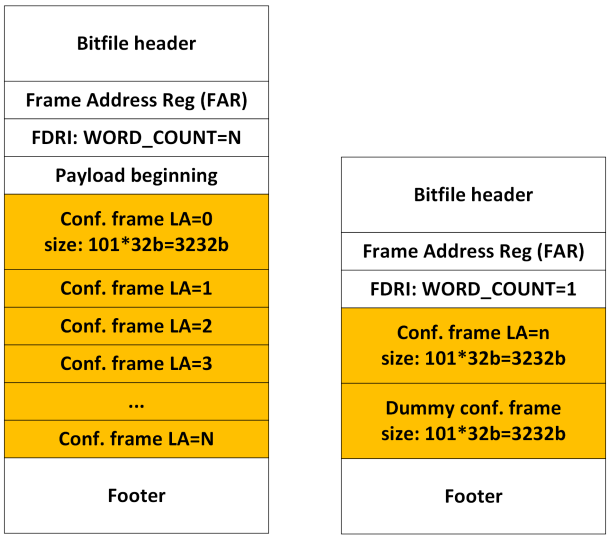


Figure 3.1: Architecture of the custom active scrubber.



(a) Main bitfile (b) Partial bitfile

Figure 3.2: Main bitfile and partial bitfile architectures

Listing 3.1: Report on correcting single bit error.

```
SC 04
SED OK
PA 01C00080
LA 00005AC9
WD 4B BT 07
COR
WD 4B BT 07
END
FC 00
SC 08
FC 40
SC 02
O>
```

Listing 3.2: Report on detecting 2-bit ECC error.

```
SC 04
DED
PA 00041923
LA 00001F63
COR
END
FC 60
SC 08
FC 60
SC 00
I>
```

Listing 3.3: Bash sript to load partial bitlife to an FPGA

```
#!/bin/bash
# Loads the bitfiles via Impact/JTAG cable.
# Usage: loadbit.sh <bitfile>

BATCHFILE=temp_load.impact

if [ ${#} != 1 ]
then
  echo 'Specify the bitfile!'
  exit 1
fi

BITFILE=${1}

rm -f ${BATCHFILE}

echo setmode -bscan >> ${BATCHFILE}
echo setcable -p auto >> ${BATCHFILE}

echo addDevice -p 1 -file ${BITFILE} >> ${BATCHFILE}

echo program -e -p 1 >> ${BATCHFILE}
echo quit >> ${BATCHFILE}

/opt/Xilinx/14.7/LabTools/LabTools/bin/lin64/impact -batch ${BATCHFILE}
wait
rm -f ${BATCHFILE}
```

References

[1] K. M. Sielewicz, G. Aglieri Rinella, M. Bonora, J. Ferencei, P. Giubilato, M. J. Rossewij, J. Schambach, and T. Vanat. Prototype readout electronics for the upgraded ALICE Inner Tracking System. *Journal of Instrumentation*, 12(01):C01008, 2017.

[2] T. Vanát, J. Pospíil, F. Kríek, J. Ferencei, and H. Kubátová. A System for Radiation Testing and Physical Fault Injection into the FPGAs and Other Electronics. In *2015 Euromicro Conference on Digital System Design*, pages 205–210, Aug 2015.

[3] Xilinx Inc. *7 Series FPGAs Configuration User Guide*, UG470 (v1.11) September 27, 2016.

[4] Xilinx Inc. *7 Series FPGAs Data Sheet: Overview*, DS180 (v2.2) December 15, 2016.

[5] Xilinx Inc. *Device Reliability Report*, UG116 (v10.5.1) December 19, 2016.

[6] M. J. Wirthlin, H. Takai, and A. Harding. Soft error rate estimations of the Kintex-7 FPGA within the ATLAS Liquid Argon (LAr) Calorimeter. *Journal of Instrumentation*, 9(01):C01025, 2014.

[7] M. Cannon, M. Wirthlin, A. Camplani, M. Citterio, and C. Meroni. Evaluating Xilinx 7 Series GTX Transceivers for Use in High Energy Physics Experiments Through Proton Irradiation. *IEEE Transactions on Nuclear Science*, 62(6):2695–2702, Dec 2015.

[8] D. S. Lee, M. Wirthlin, G. Swift, and A. C. Le. Single-Event Characterization of the 28 nm Xilinx Kintex-7 Field-Programmable Gate Array under Heavy Ion Irradiation. In *2014 IEEE Radiation Effects Data Workshop (REDW)*, pages 1–5, July 2014.

[9] Xilinx Inc. *7 Series FPGAs Configurable Logic Block*, UG474 (v1.8) September 27, 2016.

[10] A. D. Oancea, C. Stuellein, J. Gebelein, and U. Kebschull. A resilient, flash-free soft error mitigation concept for the CBM-ToF read-out chain via GBT-SCA. In *2015 25th International Conference on Field Programmable Logic and Applications (FPL)*, pages 1–4, Sept 2015.

[11] A. Gruwell, P. Zabriskie, and M. Wirthlin. High-speed FPGA configuration and testing through JTAG. In *2016 IEEE AUTOTESTCON*, pages 1–8, Sept 2016.

[12] Xilinx Inc. *Soft Error Mitigation Controller v4.1*, PG036 September 30, 2015.

[13] Xilinx Inc. *7 Series FPGAs Memory Resources*, UG473 (v1.12) September 27, 2016.

[14] Fernanda Kastensmidt and Paolo Rech. *FPGAs and Parallel Architectures for Aerospace Applications*. Springer, 2016.

[15] Niccolò Battezzati, Luca Sterpone, and Massimo Violante. *Reconfigurable field programmable gate arrays for mission-critical applications*. Springer Science & Business Media, 2010.

[16] Fernanda Lima Kastensmidt, Luigi Carro, and Ricardo Augusto da Luz Reis. *Fault-tolerance techniques for SRAM-based FPGAs*, volume 1. Springer, 2006.

[17] M. Berg. Field programmable gate array (FPGA) single effect (SEE) radiation testing.

[18] J. Schambach, M. J. Rossewij, K. M. Sielewicz, G. Aglieri Rinella, M. Bonora, J. Ferencei, P. Giubilato, and T. Vanat. ALICE inner tracking system readout electronics prototype testing with the CERN "Giga Bit Transceiver". *Journal of Instrumentation*, 11(12):C12074, 2016.

[19] A. Caratelli, S. Bonacini, K. Kloukinas, A. Marchioro, P. Moreira, R. De Oliveira, and C. Paillard. The GBT-SCA, a radiation tolerant ASIC for detector control and monitoring applications in HEP experiments. *Journal of Instrumentation*, 10(03):C03034, 2015.

[20] Jáno Gebelein. *FPGA fault tolerance in radiation environments*. PhD Thesis, 2016.

[21] ALICE Collaboration. Technical Design Report for the Upgrade of the ALICE Inner Tracking System. Technical Report CERN-LHCC-2013-024. ALICE-TDR-017, Nov 2013.

[22] G. Aglieri Rinella. The ALPIDE pixel sensor chip for the upgrade of the ALICE Inner Tracking System. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 2016.

[23] M. Mager. ALPIDE, the Monolithic Active Pixel Sensor for the ALICE ITS upgrade. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 824:434 – 438, 2016.

[24] S. Baron, J. P. Cachemiche, F. Marin, P. Moreira, and C. Soos. Implementing the GBT data transmission protocol in FPGAs. 2009.

[25] P. Moreira, A. Marchioro, and K. Kloukinas. The GBT, a proposed architecture for multi-Gb/s data transmission in high energy physics. *Proceedings of the Topical Workshop on Electronics for Particle Physics TWEPP-07, CERN-2007-07*, pages 332–336, 2007.

[26] *TowerJazz CMOS Image Sensor Technology*, 2016 (accessed February 1, 2016).

[27] P. Buncic, M. Krzewicki, and P. Vande Vyvre. Technical Design Report for the Upgrade of the Online-Offline Computing System. Technical Report CERN-LHCC-2015-006. ALICE-TDR-019, Apr 2015.

[28] A. Alici, A. Di Mauro, W. Riegler, and A.Tauro. Radiation Dose and Fluence in ALICE after LS2. Technical report, June 2015.

[29] A. Morsch and B. Pastircak. Radiation in ALICE Detectors and Electronic Racks. Technical report, April 2002.

[30] The ALICE Collaboration. The ALICE experiment at the CERN LHC. *Journal of Instrumentation*, 3(08):S08002, 2008.

[31] C. Färber, U. Uwer, D. Wiedner, B. Leverington, and R. Ekelhof. Radiation tolerance tests of SRAM-based FPGAs for the potential usage in the readout electronics for the LHCb experiment.

[32] LHCb Collaboration. LHCb PID Upgrade Technical Design Report. Technical Report CERN-LHCC-2013-022. LHCB-TDR-014, Nov 2013.

[33] K. Petridis J. Rademacker P. Garsed S.A. Wotton R. Cardinale A. Petrolini D. Clark U. Egede M. McCann M. Patel T. Savidge D. Websdale M. Brock N. Harnew M. Tacon M. Benettoni A. J. Brummitt S. Easo A. Papanestis S. Ricciardi F. F. Wilson C. D'Ambrosio C. Frei J. He D. Piedigrossi M. Adinolfi, J. Kariuki. LHCb Upgraded RICH 1 Engineering Design Review Report. Technical Report LHCb-PUB-2016-014. CERN-LHCb-PUB-2016-014, CERN, Geneva, Jun 2016.

[34] R. Cardinale A. Petrolini R. Cardinale A. Petrolini S. Easo C. D'Ambrosio C. Frei J. He D. Piedigrossi P. Garsed, S.A. Wotton. LHCb Upgraded RICH 2 Engineering Design Review Report. Technical Report LHCb-PUB-2016-015. CERN-LHCb-PUB-2016-015, CERN, Geneva, Jun 2016.

[35] C. Sturm V. Friese, editor. *CBM Progress Report 2011*. GSI, Darmstadt, 2012.

[36] L. Evans and P. Bryant. LHC Machine. *Journal of Instrumentation*, 3(08):S08001, 2008.

[37] J. Bartke. *Introduction to relativistic heavy ion physics*. World Scientific, Singapore, 2009.

[38] Jonathan M. Johnson and Michael J. Wirthlin. Voter insertion algorithms for fpga designs using triple modular redundancy. In *Proceedings of the 18th Annual ACM/SIGDA International*

Symposium on Field Programmable Gate Arrays, FPGA '10, pages 249–258, New York, NY, USA, 2010. ACM.