Mission-critical digital systems are often implemented on Field Programmable Gate Arrays (FPGAs), which allow for easy configuration of the hardware. FPGAs are especially desirable in these applications due to their cost-effectiveness for relatively small quantity productions and the ease with which they can be reused. The configurable nature of FPGAs presents one of its strongest advantages as well as one of its most significant limitations as compared to an Application Specific Integrated Circuit (ASIC), which is a fixed logic integrated circuit that is customized for a particular use. Non-permanent characteristic of FPGAs also causes problems with errors in the system, especially in environment subject to high radiation. Particles may cause portions of the configured circuitry to change states. FPGAs are therefore more prone to errors in their logic values and in its actual circuitry than a custom hardware solution such as an ASIC. Thus, here proof-of-concept for the system that is capable of detecting such errors and partially self-reconfiguring these corrupted areas is illustrated. This “self-healing (Self Correcting)” system is capable of gracefully recovering from soft errors while maintaining valid system outputs. The SEU controller macro and reference design can emulate an SEU by deliberately injecting an error into the FPGA configuration so that its subsequent detection and correction can be confirmed and SEU Injection of errors can also be used to assess SEU mitigation circuits implemented in a design. We describe the operation and architecture of the proposed logic design as well as its implementation in Xilinx virtex-5 FPGA.

**Keywords:** Gate array, logic design and digital system.

**INTRODUCTION**

Considering the remarkable success of configurable logic devices in areas such as telecommunications and defense applications, it is natural that an interest should arise in their use for Space based Electronics solutions as well. However, while such devices present numerous advantages in terms of design flexibility, they come with the draw back of being susceptible to bit upsets induced by radiation, more commonly known as “Single Events Upsets (SEUs)”. FPGAs have been very attractive for space applications over the past decade. Indeed, the main advantage provided by gate arrays is the elimination of the large overhead cost of developing custom ASICs. One of the major reasons being the prospect of performing post-launch design optimizations or changes in spacecraft objectives. Another advantage, their inherent reprogrammability feature has been fully exploited for prototyping purposes (Philippe and Greg, 2008), while antifuse technology has several inherent limitations that make SRAM-based FPGAs more attractive. First, once a device is programmed, it cannot be changed, additional devices have to be programmed and physically replace the installed devices. Second, available antifuse gate arrays are considerably smaller in gate count than SRAM configurable gate arrays. The problem with SRAM based FPGAs are known to be highly susceptible to SEUs. SEUs can result in deviations from expected component behavior. Single event upset (SEU) is defined by NASA as "Radiation-induced errors in microelectronic circuits caused when charged particles (usually from the radiation belts or from cosmic rays) lose energy by ionizing the medium through which they pass, leaving behind a wake of electron hole pairs” (Philippe and Greg, 2008). In the case of programmable logic devices such as FPGAs, SEU effects can be much more severe. Since FPGAs utilize a configuration memory array to define the logic function; an SEU occurring in a single bit in the array can lead to an unexpected alteration of the original design. An SEU may occur in analogue, digital, optical components, or may have effects in surrounding interface circuitry. FPGAs based on SRAM can be reprogrammed an unlimited number of times, even in the end-user system. Many FPGAs contain, in addition to memory elements such as flip-flops and random access memories (RAMs), a large static random access memory (SRAM) that establishes the overall application performed by the FPGA. An SEU induced bit-flip in the SRAM configuration memory, therefore, can alter the functionality of the FPGA. This makes SEUs of significantly more concern in FPGAs than in traditional application specific integrated circuits (ASICs) (Ohlsson et al., 1998).

The Single Event Upset occurs when radiation affects the transistors that are part of the look up table logic of the FPGAs RAM component. If the lookup table is affected by radiation, it can change the bit values associated with the hardware made up of the current FPGA design. SEU is a change of state caused by ions or electro-magnetic radiation striking a sensitive node (area) in a micro-electronic device (Bit-Flip) as shown in the figure 1. The state change is a result of the free charge created by ionization in or close to an important node of a logic element. In these FPGAs, a multitude of latches, also called memory cells or RAM bits, define all logic functions and on-chip interconnects.
Such latches are similar to the 6-transistor storage cells used in SRAMs, which has proved to be sensitive to single event upsets caused by high-energy neutrons (Lesea, 2009).

There are mainly five areas of Configurable Logic Blocks (CLB) that are affected by SEU as shown in Fig 2. Upsets in the logic (LUT), upsets in the customization routing bits inside the CLB, upsets in the routing connecting CLBs and pins, upsets in the CLB flip-flops (flip-flops) and upsets in Block RAM.

For the proof of concept of Self Healing System we have used modules of sampler as a user design, which is the critical part of the design, and then we have chosen up-sampler and down-sampler as a reconfigurable module. The emulation means detection and correction have been implemented with the help of SEU Monitor system which consists of SEU controller macro and monitor program. TMR has been done on the user design for first level of security against Single Event Effect (bit-flip). Finally the user design, SEU Monitor system, Soft core processor (MicroBlaze) (Ken, 2009) and Partial reconfiguration have been integrated to get the self healing system. In this paper, we present an efficient prototype of Self Correcting System (Self Healing), which works in conjunction with existing SEU detection mechanisms in Virtex-5 FPGA. This system can be synthesized and incorporated with user-defined digital applications in any Virtex-5 FPGA for detecting as well as self correction of SEUs during normal on-line system operation. We begin with an overview of existing SEU detection mechanisms (FPGA preambles and configuration memory) in Section 2. The design and the architecture of the proposed SEU detection and correction circuit are presented in Sections 3. Section 4 describes the implementation of Self correcting system. The paper concludes with results and a summary in Section 5 and 6 respectively (Stoica et al., 2006 and Heiner et al., 2008).

**FPGA preambles and configuration memory**: Like any other RAM, the configuration memory of an FPGA is partitioned into words, also called frames, which represent the smallest addressable unit of the memory for write and read operations. Virtex-5 frames consists of 1,312 bits (Kristan et al., 2007). Each frame includes a 12-bit field consisting of 11 Hamming bits and an overall parity bit (bits 640 to 651) for the frame data to provide the potential for single error correction (SEC) as well as double error detection (DED) in the frame data and 16 unused bits (Bits 656 to 671). The parity and Hamming bits are generated external to the FPGA by the configuration bit stream generation software and are subsequently downloaded with the application specific configuration data. However, system memory data subject to change during the operation of the FPGA, such as the contents of block RAMs and look-up tables (LUTs) used as distributed RAMs, are not covered by the parity and Hamming bits (Carmichael and Wei Tseng, 2008).

**FRAME ECC and ICAP Primitives**: Virtex-5 provide a specialized primitive, called FRAME ECC (error correcting code) as shown in figure 3, for detection and identification of single and double-bit errors in the frame data (Carmichael and Wei Tseng, 2008). For each frame read from the configuration memory, the Frame ECC module calculates the Hamming bits as well as the overall parity for the frame data, and compares these bits with the Hamming bits and parity for that frame stored in the configuration memory. Based on this comparison, the Frame ECC module produces indications for no error, single bit error, and double-bit error conditions in addition to a syndrome indicating the location of single bit errors, which has been summarized in Table I.

**Read back CRC is performed in this manner:**
1. Continuous Read back of configuration data in the background of a user design.
2. Dedicated logic Read back continuously in the background to check the CRC of the configuration memory content.
3. The first round of Read back CRC value is latched as the golden value for later comparison.
4. The subsequent rounds of Read back CRC value are compared against the golden value.
5. When a CRC mismatch is found, the CRCERROR pin of the FRAME_ECC_VIRTEX5 primitive is driven high.

The location of single-bit errors within the frame is indicated by the syndrome[10:0] outputs of the Frame ECC primitive, however some additional combinational logic is required to determine the exact bit-offset of the error within the configuration frame. An efficient algorithm for determining the bit-offset of the error in the range 0-1311 is shown in Equation 1, where S[10:0] are the Frame ECC syndrome outputs.


If the binary value of syndrome [10:0] is 0 or a power of 2, then the error is located in one of the parity bits, in which case the location of the bit error is determined as shown in Table 2. The output of the syndrome combinational logic is tied to the B port address inputs. In this manner, the erroneous bit, as indicated by syndrome [11:0], is inverted when the block RAM B port write enable is asserted. The repaired frame is then written back into the configuration memory via the A port 32-bit output to the Internal Configuration Access Port (ICAP) (Carl et al., 2000).

A potential problem can arise when an odd number of multiple bit errors occur in a single frame of configuration memory. These errors will cause both a syndrome mismatch and overall parity mismatch, which collectively alias as a single-bit error (Table 1). However, in this case, the syndrome outputs do
not necessarily indicate the location of any of the actual errors, and can erroneously point anywhere in the range 0 to $2^{11} - 1$ (2047). Since the actual frame data only exists in the range 0 to 1311, two scenarios are possible. In the first scenario, the odd-multiple bit error aliases as a single-bit error with the syndrome outputs pointing in the valid range of the frame data 0 to 1311. In response to the single-bit error indication, the SEU controller will invert the frame-bit pointed to by the syndrome, which will satisfy the Hamming code by creating a valid distance code word, and the modified frame will be written back into the configuration memory. The SEU controller will resume read back at the start of the configuration column containing the damaged frame. When the erroneous frame, now containing an even number of multiple errors, is read, the valid code word will cause a Hamming code match and an overall parity-bit match such that a “no bit error” indication is obtained. In the second scenario, when the frame containing an odd number of errors greater than one is read, the syndrome indicates an error bit location in the range from 1312 to 2047. The Frame ECC function is performed each time a frame is read via a configuration interface such as the external serial Boundary Scan interface or parallel SelectMAP interface. In addition to this, Xilinx Virtex-5 contains a 32-bit internal configuration access port (ICAP) primitives that provides access to the configuration memory from within the FPGA core (Jones, 2007). The Frame ECC function is also performed each time a frame is read via the ICAP. Since the Frame ECC does not provide error correction, circuitry must be added in the FPGA fabric that uses the ICAP and Frame ECC modules to cycle through all frames and to detect and correct SEUs in the configuration memory.

**SEU Detection and Correction Using SEU Monitor System:** In this method we have used the SEU controller macro provided by the Xilinx as shown in the figure 4 (Shadab, 2008). Inside the macro, the ICAP_VIRTEx5 and FRAME_ECC_VIRTEx5 primitives as described earlier are used to clock and observe the readback CRC circuit, which performs the SEU detection. The macro also includes a controller that connects to the other ports of these primitives to perform the operations necessary to locate and correct SEU errors using the built-in ECC facility. For test purposes, the connection to ICAP is used to facilitate the controlled injection of configuration errors.

The operation of the SEU controller is as follows:

1. A 1312-bit frame of configuration memory is read through the ICAP as forty-one 32-bit words. The frame data is stored in a block RAM.
2. If an error is indicated by the outputs of the Frame ECC primitive, the type of error is determined as shown in Table 1 (If bit $S[11] = 0$, then the whole frame must be restored. If bit $S[11] = 1$, then bits $S[10:0]$ are used to locate the error bit in the saved frame, and the bit is inverted) If the error indicates a double-bit error, the error output of the SEU controller is latched high and read back continues with the next frame of configuration memory. If a single-bit error is indicated, the location of the bit is determined from the syndrome and the erroneous bit is corrected (i.e. inverted) in the frame data stored in the block RAM.

- The repaired frame is written back into the configuration memory at the same frame address from which it was read.
- Read back resumes with the first frame of configuration memory in the configuration column containing the newly repaired frame.
- When a configuration column has been completely read and repaired, the SEU controller advances to the next configuration column in the array.

Along with this macro we require the SEU monitor system which we have created to handle the functionality of this macro as shown in figure 5. In this method we require the soft processor to monitor and to control the functionality of SEU controller macro and to provide the user interface. In our project we have used the MicroBlaze as a soft core processor and UART (RS232) as a standard IO of the entire system to provide the interface for the user. Figure 6 show the internal signal connection of the monitor and the controller macro.

The SEU controller macro and reference design can emulate an SEU by deliberately injecting an error into the FPGA configuration so that its subsequent detection and correction can be confirmed.

**Self correcting system:** Finally the all the modules have been intergraded into one system called Self correcting System. The SEU controller macro has been integrated in to the previously created Self reconfigurable system. The Block Diagram of the self correcting system is shown in figure 7. The concept of the final design is like this, the user design is triplicated by means of TMR method and that would be the Recongurable Module (RM), then using the SEU Monitor system we can emulate the SEU by injecting the error, detecting and then correcting the bit error by means of the facility of SEU monitor System and if this is not possible then the reconfiguration is carried out by two modes (Manual or Self). The System operates in different three modes:SEU Emulation mode, Manual Reconfiguration Mode and Automatic Reconfiguration mode. First mode (SEU Emulation mode), is the same as only for the emulating the SEUs by injecting, detecting and correcting the error into configuration memory, which is same as the only monitor system created earlier. The second mode (Manual Reconfiguration Mode) adds the feature of reconfiguration of the user logic into first
mode. In this reconfiguration is carried out if the error is detected and the monitor system cannot correct it or as and when user is need to do so. But the reconfiguration is carried out by user interface manually. Third mode (Automatic Reconfiguration mode) is automatic mode. If the system is kept into this mode, the system keep macro in mode 1 which is automatic detection and correction and if the correction is not possible by macro then it do the reconfiguration of the user logic.

RESULTS AND DISCUSSION:
Figure 8 shows the RTL of Self Correcting System which includes the controller macro and soft core processor. Figure 9 and 10 shows the testing result on hyper terminal. Table 3 shows the resource utilization of the self correcting system.

CONCLUSION:
A Self Correcting System using the SEU monitor system and self reconfigurable system is presented in this paper, which is capable of injecting, detecting and correction the of single-bit errors manually and automatically when operated and in automatic mode for all Virtex-5 FPGAs. It is also capable of injecting and detection the double-bit errors in the FPGA configuration memory. The design is easily integrated in any existing user design with minimal resource overhead for detection and correction of single bit errors using the presented design which uses the soft core processor and controller macro. The SEU controller macro basically uses the ICAP and FRAME_ECC primitives of virtex-5 FPGA. The resource utilization for Virtex-5(xc5vlx110t-1ff1136) and Virtex-6(xc6vlx240t-1ff1156) has been summarized in table 3.

REFERENCE:
Note: * Available at www.xilinx.com.
Fig 7. Block Diagram of Self Correcting System

![Block Diagram of Self Correcting System](image)

Fig 8. RTL of Self Correcting System

![RTL of Self Correcting System](image)

Fig 9. Testing results of SEU monitor System on UART (Hyper Terminal) - 1

![Testing results of SEU monitor System on UART](image)

Fig 10 Testing results of Self Correcting System on UART (Hyper Terminal) – 2

![Testing results of Self Correcting System on UART](image)

Table 1. Frame ECC Error Codes.

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Condition (SYNDROMEVALID = 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No bit error</td>
<td>Hamming match, no parity error</td>
</tr>
<tr>
<td>1-bit correctable error (SEC)</td>
<td>Hamming mismatch, parity error</td>
</tr>
<tr>
<td>2-bit error detection (DED)</td>
<td>Hamming mismatch, no parity error</td>
</tr>
</tbody>
</table>

Table 2. Parity bit error diagnosis.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100000000001</td>
<td>640</td>
<td>100000000000</td>
<td>646</td>
</tr>
<tr>
<td>100000000010</td>
<td>641</td>
<td>100000000001</td>
<td>647</td>
</tr>
<tr>
<td>100000001000</td>
<td>642</td>
<td>100100000000</td>
<td>648</td>
</tr>
<tr>
<td>100000100000</td>
<td>644</td>
<td>101000000000</td>
<td>649</td>
</tr>
<tr>
<td>100000100000</td>
<td>645</td>
<td>100000000000</td>
<td>650</td>
</tr>
<tr>
<td>100000000000</td>
<td>646</td>
<td>100000000000</td>
<td>651</td>
</tr>
</tbody>
</table>

Table 3. Resource utilization

<table>
<thead>
<tr>
<th>Device</th>
<th>Virtex-5: xc5vlx110t-1ff1136</th>
<th>Virtex-6: xc6vlx240t-1ff1156</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available</td>
<td>Used</td>
<td>Utilization</td>
</tr>
<tr>
<td>Total Number Slice Registers</td>
<td>69,120</td>
<td>2,599</td>
</tr>
<tr>
<td>Number of LUTs</td>
<td>69,120</td>
<td>2,220</td>
</tr>
<tr>
<td>Number of occupied Slices</td>
<td>17,280</td>
<td>1,350</td>
</tr>
<tr>
<td>Number of bonded IOBs</td>
<td>640</td>
<td>4</td>
</tr>
<tr>
<td>Number of BlockRAM/FIFO</td>
<td>148</td>
<td>66</td>
</tr>
<tr>
<td>Number of DSP48</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td>Number of DCM, AdVs</td>
<td>12</td>
<td>1</td>
</tr>
<tr>
<td>Number of ICAP, VIRTEX</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Number of FRAME ECC, VIRTEX</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

***************