Abstract

This document presents information and the radiation results of the Single Event Upset (SEU) testing performed on an 80S32 microcontroller device at ESA/ESTEC Cf-252 radiation facilities.
# Table of Contents

1 Revision History
2 General Information
   2.1 Scope
   2.2 Applicable documents – References
   2.3 Definition of terms
   2.4 List of acronyms
   2.5 People Involved
3 Test component details
4 Test setup
   4.1 Radiation test facility
   4.2 Test conditions
      4.2.1 Temperature and Humidity
      4.2.2 Lighting
      4.2.3 Vacuum
      4.2.4 DUT values
   4.3 Component preparation
   4.4 Test setup
      4.4.1 Test board
      4.4.2 Remote keypad
      4.4.3 Test harness
   4.5 Data logging, programming and monitoring equipment
   4.6 FPGA H/W
5 Test Software Description
   5.1 General Information
   5.2 Memory Test
      5.2.1 Brief Description
      5.2.2 Detailed Description
   5.3 Memory & Arithmetic Test
      5.3.1 Brief Description
      5.3.2 Detailed Description
6 Analysis Software Description
   6.1 memonly
   6.2 mem_math
7 Radiation procedure with SEU monitoring
   7.1 1st run
   7.2 2nd run
   7.3 ISD S.A. Responsibilities
   7.4 ESTEC CASE responsibilities
8 Test Results
   8.1 Summary of test results
   8.2 Upset rate
List of Figures

Figure 4.1: Cf-252 Energy distribution................................................................. 9
Figure 4.2: Cf-252 Energy distribution................................................................. 9
Figure 4.3: Cf-252 LET distribution of fission products...................................... 9
Figure 4.4: CASE radiation facility................................................................. 10
Figure 4.5: The baseplate used for the tests. External diameter is shown......... 10
Figure 4.6: Test setup block diagram............................................................ 12
Figure 8.1: Cross-section per bit vs LET (Atmel)........................................ 24
Figure 8.2 - Errors Running Total for Memory-Test........................................ 26
Figure 8.3 - Errors per time interval for Memory-Test (time interval = 1hour)... 26
Figure 8.4 - Detected errors running total for the Memory & Arithmetic test... 29
Figure 8.5: - Detected errors per time interval for the Memory & Arithmetic test (time interval = 1hour)... 29
Figure 10.1: 80S32 DUT lidded and top marking......................................... 31
Figure 10.2: 80S32 DUT de-lidded showing the exposed die inside the cavity... 31
Figure 10.3: 80S32 DUT inside the socket of the test board......................... 32
Figure 11.1: Test board inside the chamber. Belljar and implosion guard in place 33
Figure 11.2: Cable details showing the vacuum release valve (yellow knob)... 33
Figure 11.3: Test board on the baseplate. Source holder not in place............ 34
Figure 11.4: Details of the test board showing the rubber feet and the IDC connectors.. 34
Figure 11.5: General arrangement of the test setup..................................... 35
List of Tables

Table 3.1: Details of the component used for testing.........................................................................................................................8
Table 4.1: key parameter values for the DUT................................................................................................................................................11
Table 4.2: Summary of signals passing through the test harness...........................................................................................................14
Table 6.1: Log files and associated parser application..........................................................................................................................18
Table 6.2: Example of output files.........................................................................................................................................................19
Table 7.1: Durations and test/analysis software for each run...................................................................................................................20
Table 8.1: Summary of test results.........................................................................................................................................................22
## Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Author</th>
<th>Modification description</th>
</tr>
</thead>
<tbody>
<tr>
<td>05/02/2009</td>
<td>1.0</td>
<td>K. Makris, E. Politis, T. Lampaounas</td>
<td>Creation</td>
</tr>
<tr>
<td>03/03/2009</td>
<td>1.1</td>
<td>K. Makris, E. Politis, T. Lampaounas</td>
<td>Updates: §2.1, §2.2, §4.2.1, §4.2.3, §4.4.1, §5.2.2, §5.3.2, Chapter 6, Table 7.1, §7.1, Table 8.1, §8.2, §8.3, §8.4  &lt;br&gt; Additions: Table 6.1, Table 6.2, Chapter 9, APPENDIX D, APPENDIX E.</td>
</tr>
<tr>
<td>10/03/2009</td>
<td>1.2</td>
<td>K. Makris, E. Politis, T. Lampaounas</td>
<td>Updates: sections: 5.2.1, 5.2.2, 5.3.1, 8.3.2, equation (5) on pg.24.</td>
</tr>
<tr>
<td>29/04/2009</td>
<td>1.3</td>
<td>K. Makris, E. Politis, T. Lampaounas</td>
<td>Additions: section 8.3.3  &lt;br&gt; Updates: chapter 9</td>
</tr>
</tbody>
</table>
2  General Information

2.1  Scope

This document presents the results of Single Event Upset (SEU) testing performed on a single 80S32 component using the ESA Californium-252 Assessment for Single event Effects (CASE) facility at ESTEC in Noordwijk, The Netherlands. Information regarding the test setup and its installation in CASE facility are also reported. The results include the number of detected errors induced by radiation effects on the correct execution of special test programs, the exact time of occurrence during the runs and the number of successfully corrected (recovered) errors.

The main purpose of the testing was to perform a preliminary characterization (pre-testing) of the device in terms of sensitivity to radiation effects under specific operating conditions. This preliminary characterization was focused on evaluating the radiation performance of various structural blocks of the DUT, particularly the internal memory blocks, the logic elements and the EDAC algorithm. The characterization strategy was based on counting all the accumulated errors detected by the internal EDAC mechanism during the execution of special routines which were able to distinguish between corrected (recovered) and uncorrected (non-recovered) errors. Any abnormal behavior or anomaly was monitored and logged. The goal of testing was the acquisition of sufficient amount of data during the runs which is presented in this document. The characterization of the DUT in terms of SEU sensitivity will be completed after post-processing and interpreting the statistical sample collected during the tests.

The report begins by providing information about the device selected to perform the tests. Then, details are provided both for the test setup and for the software used for testing and analysis. The results of all test runs are reported along with their corresponding graphs. Calculations of characteristic parameters such as the error rate, test cross-section and cross-section per bit of the test runs are also reported. Representative photographs of both the component and the test setup along with the details of all radiation runs are included on separate Appendices of this report.

2.2  Applicable documents – References

2.3 Definition of terms

The following definitions are stated on [1] and presented here as a reference.

**Single Event Upset (SEU)**

SEU is also known as a soft error. The change of state of a latched logic state from one to zero or vice-versa. A single event upset is non-destructive and the logic element can be rewritten or reset.

**Linear Energy Transfer (LET) or Stopping power**

The amount of energy deposited per unit length along the path of the incident ion. It is expressed in units of MeV/mg/cm$^2$, which is the energy per unit length divided by the density of the irradiated medium.

The total duration of the radiation campaign was three days. It started on 12\textsuperscript{th} of January and ended on 14\textsuperscript{th} of January 2009. For convenience, each date of the test will be mentioned as follows:

- Day-1: Monday, 12\textsuperscript{th} of January 2009
- Day-2: Tuesday, 13\textsuperscript{th} of January 2009
- Day-3: Wednesday 14\textsuperscript{th} of January 2009

2.4 List of acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUT</td>
<td>Device Under Test</td>
</tr>
<tr>
<td>EDAC</td>
<td>Error Detection And Correction</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field Programmable Gate Array</td>
</tr>
<tr>
<td>JTAG</td>
<td>Joint Test Action Group</td>
</tr>
<tr>
<td>LET</td>
<td>Linear Energy Transfer</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
</tr>
<tr>
<td>SEE</td>
<td>Single Event Effect</td>
</tr>
<tr>
<td>SEU</td>
<td>Single Event Upset</td>
</tr>
<tr>
<td>UART</td>
<td>Universal Asynchronous Receiver/Transmitter</td>
</tr>
<tr>
<td>USART</td>
<td>Universal Synchronous - Asynchronous Receiver/Transmitter</td>
</tr>
<tr>
<td>ZIF</td>
<td>Zero Insertion Force</td>
</tr>
</tbody>
</table>

2.5 People Involved

<table>
<thead>
<tr>
<th>Name</th>
<th>Location</th>
<th>E-mail address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tasos Lampaounas</td>
<td>ISD, Athens</td>
<td><a href="mailto:lampaoun@isd.gr">lampaoun@isd.gr</a></td>
</tr>
<tr>
<td>Efstratios Politis</td>
<td>ISD, Crete</td>
<td><a href="mailto:spolitis@isd.gr">spolitis@isd.gr</a></td>
</tr>
<tr>
<td>Kostas Makris</td>
<td>ISD, Athens</td>
<td><a href="mailto:kmakris@isd.gr">kmakris@isd.gr</a></td>
</tr>
</tbody>
</table>
3 Test component details

The SEU testing was performed using only one device from the inventory of available samples. The total number of available samples was eight (8) and they were all supplied by ESA-ESTEC. From the total number of available samples, three (3) of them were delivered to ISD S.A at a previous date. Those three components were serialized by ESTEC prior to delivery with serial numbers (S/N): S/N#3, S/N#4 and S/N#5 respectively. The rest of the available samples (5) remained at ESA-ESTEC. In the scope of testing, ISD S.A. has equipped the test setup with the devices S/N#3 and S/N#5 but neither of these two components were eventually used to perform the test. The test was carried out using a component supplied by the stock of ESA-ESTEC. It should be noted that there was no particular restriction on selecting the component to execute the test.

Table 3.1 shows the details of the selected component. Photographs of the actual part depicting the complete top marking and the die orientation inside the cavity are included in Appendix A. The internal architecture, functionality and pin list of the component are covered in detail in [4].

<table>
<thead>
<tr>
<th>Part type</th>
<th>ADV80S32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part description</td>
<td>8-bit microcontroller</td>
</tr>
<tr>
<td>Package type</td>
<td>100 LEAD MQFP ceramic with gold-plated leads.</td>
</tr>
<tr>
<td>Available samples</td>
<td>8</td>
</tr>
<tr>
<td>Top marking of the DUT</td>
<td>5962-00B0202V8C FMAADV80S32 0148-FR37586L</td>
</tr>
</tbody>
</table>

Table 3.1: Details of the component used for testing.

ESA-ESTEC recommended that the component used for testing should be reused as a DUT component, if any further radiation characterization tests are to be scheduled in the future. For this reason and to avoid any logistics-inventory problems, the component with S/N#5 was exchanged with the component of Table 3.1 by the end of this radiation campaign.
4 Test setup

This paragraph contains information about the radiation test facility and the actual setup used to perform the radiation tests.

4.1 Radiation test facility

The test facility used for the campaign was the ESA Californium-252 Assessment of Single-event Effects (CASE) facility at ESTEC, The Netherlands. The CASE system uses a 1 to 3 microcurie Cf-252 source which is safe to handle in the laboratory, provided that the normal precautions for the handling of radioactive sources are observed. The spontaneous fission of Californium-252, which has an effective half-life of 2.65 years, produces a wide range of high-energy particles having an average LET of 43 MeV/(mg/cm²). Cf-252 produces an average of 3.7 neutrons per fission with an average energy of 2.3 MeV. [6]

The Cf-252 source used in the test had an activity of 1.23uCi with a standard uncertainty of ±30%.

The following diagrams are from [6] and presented here as a reference.

The DUT chamber consists of a dome-shaped bell-jar made of glass, lying on top of a 400mm X 400mm metal base plate. The bell-jar has an internal diameter of 270mm, height of 250mm plus a 100mm radius dome (Figure 4.4). The baseplate includes a variety of feedthrough connectors for electrical connections. The Cf-252 source is placed on metal holder, which can move freely up and down with a help of rod for height adjustments above the DUT. The DUT test board is placed inside the chamber and stands freely on the baseplate, while the source holder is aligned directly above the DUT. The air inside the chamber is pumped out using a vacuum pump via a special port on the baseplate. The value of the vacuum is monitored in real time via a meter located near the chamber. The CASE system may use a variety of baseplates with the differentiating factor to be the number and type of the feedthrough connectors. The baseplate used in the test is illustrated in Figure 4.5. During all tests, an aluminium screened implosion guard was placed around the belljar to protect the DUT against sensitivity to light and electromagnetic interference.
Figure 4.4: CASE radiation facility

Figure 4.5: The baseplate used for the tests. External diameter is shown.
4.2 Test conditions

This paragraph provides information about key parameters of the test.

4.2.1 Temperature and Humidity

Air-conditioning in the laboratory which hosts the CASE facility maintains a temperature of 22º C ± 2º C with a humidity level of 40% ±10% [8]. Since the DUT operated under vacuum conditions, there was no possibility for cooling the chip using natural convection. Due to the lack of any alternative cooling mechanism, the DUT temperature is expected to be higher than the ambient temperature. However, there was no provision for measuring the temperature of the chip separately during the runs, so a specific value cannot be reported.

4.2.2 Lighting

As mentioned in section 4.1 an aluminum screened implosion guard was used around the belljar, to protect the DUT from ambient lighting. Thus, during all tests the vacuum chamber area remained darkened.

4.2.3 Vacuum

The target operating vacuum value inside the chamber was $10^{-2}$ mbar, although this condition was never reached during the tests. The actual vacuum reading on all tests was stabilized on $2 \times 10^{-2}$ mbar and this was eventually achieved after approximately 5 hours of pumping. It should be noted that the vacuum pump was operating continuously during the tests and was switched on at the beginning of each test. A vacuum of $6 \times 10^{-2}$ mbar was established after approximately 2 hours.

4.2.4 DUT values

Table 4.1 specifies the values of key parameters for the DUT. The baud rate is referred to the value selected for the UART port of the test setup.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core voltage-source</td>
<td>+5.15VDC regulated supply using the on board regulator LMS1585 [3].</td>
</tr>
<tr>
<td>I/O voltage-source</td>
<td>+5.15VDC regulated supply using the on board regulator LMS1585 (the same source as the core supply) [3].</td>
</tr>
<tr>
<td>Clock frequency</td>
<td>22 MHz</td>
</tr>
<tr>
<td>Clock source</td>
<td>FPGA device of the test-board [3].</td>
</tr>
<tr>
<td>UART Baud rate</td>
<td>9600 bps, no parity</td>
</tr>
</tbody>
</table>

Table 4.1: key parameter values for the DUT.

1 mbar x 100 = Pascal
4.3 Component preparation

Since the penetration depth of the fission fragments of the Cf-252 source is only a few tens of microns and in order to increase the fission product influence at the DUT, the device itself needs to be de-lidded (decapsulated) to expose the die surface. From the total number of available samples three devices were de-lidded. The first device was decapsulated 1 month before carrying out the test in ESA-ESTEC laboratories, in order to determine the difficulty of the de-lidding process.

Two more devices were successfully de-lidded in ESA-ESTEC laboratories during day-1 of the radiation campaign in order to have spare devices available for the tests (backup). From the new devices that were de-lidded, the first was the sample S/N#5 while the second was the component which eventually used to perform the tests (Table 3.1).

Before de-lidding, the selected DUT component (Table 3.1) was subject to some basic functional tests by executing the “Instruction Set” & “Performance” tests under normal ambient conditions (atmospheric pressure, no radiation exposure) to check the correct operation. After the de-lidding operation, the particular component was subject again to the same functional tests to determine if any damage occurred during the de-lidding procedure. These tests did not only verified the correct operation of the DUT component, but also the functionality of the test board and the overall setup. Both the DUT component and the test board were exhibited the expected behavior.

Photographs of both lidded and de-lidded DUT component are shown in Appendix A.

4.4 Test setup

The test setup of the radiation campaign is graphically represented in Figure 4.6.
The hardware setup provided by ISD S.A. consisted of the following major parts:

- test board with external power supply
- remote keypad
- test harness (cabling)
- host PC with ALTERA® ByteBlaster™
- laptop PC with external flash memory programmer.

### 4.4.1 Test board

The test board (validation board) consists of a motherboard and a daughterboard PCB (DUT board). The daughterboard contains the socket, which hosts the DUT and is placed on top of the motherboard in a 'stacked' arrangement. The exposed die of the DUT is directly accessible from the radioactive source via an opening on the top side of the socket. The motherboard includes an FPGA device, memory devices, the power supply and various I/O interfaces. The technical specifications of the test board (also referred to as validation board) are presented with details in [3]. The user manual of the test board is given on [5], including a detailed description of the connectivity between the different elements of the test setup as well.

Since the CASE baseplate does not provide a specific mounting option for the test board, the latter was placed on the baseplate using a set of M3 brass spacers, mounted on the 4 corners of the motherboard. In order to prevent the motherboard from skewing from its original position during positioning of the belljar, the brass spacers were terminated on rubber feet for increased friction. The clearance between the baseplate and the bottom side of motherboard PCB was set to 58mm in order
to allow the cables to run and lie comfortably underside. The final position of the the test board was chosen properly so as the central point of the source holder to lie directly above the center of the exposed die. This optical alignment was performed manually. The height of the source holder was adjusted on the vertical rod in order the distance between the source holder and the top side of the DUT socket to be approximately 1mm. The overall distance between the source holder and the die surface was 10 mm.

The complete system (motherboard plus the DUT board) was able to fit properly inside the belljar and the cabling was found to be compatible. In general, no hardware problem was encountered.

The DUT was powered exclusively from the motherboard via the appropriate on-board regulators (see [3] for more details). The board was powered by an external +8VDC/1A power supply mounted on a wall power outlet of the lab. Via a switch it was possible to turn on and off the power supply from outside the belljar.

Photographs of the installed test board inside the CASE system are shown in Appendix B.

4.4.2 Remote keypad

A remote keypad located outside the vacuum chamber permitted remote communication with the test board during the runs. The remote keypad is shown in the test setup photographs in Appendix B and is described in more detail in [5]. Via the remote keypad it was possible to perform the following operations:

- selection between the RESET and RUN mode of the DUT by toggling the key #1 on the keypad.
- optical indication of the above current status on the LCD.
- Variation of UART baud rate by selecting among others from a list (Table 4.1).
- Via a H/W that was developed during day-2 of the radiation campaign, display of the following values in Hex format:
  - number of corrected errors
  - number of uncorrected errors
  - the memory address on which the last error has occurred
  - number of arithmetic errors

4.4.3 Test harness

The term test harness is used to describe all the feedthrough cables that were utilized to interface the test board with the rest of the equipment located outside the vacuum chamber. The test harness consists of 2 sets of ribbon cables (4 ribbon cables in total), indicated with the letters A-D and shown with thick lines on Figure 4.6. The thick dotted line depict the ribbon cables connecting the connectors inside and outside the chamber on the baseplate. The first set (A,B) is used to connect the equipment (host PC, power supply, etc.) with the connectors outside the chamber, while the second set (C,D) to connect the test board with the connectors lying inside the chamber. The ribbon cables at one end were terminated in 40 pin female IDC connectors in order to mate properly with the IDC sockets available on the baseplate. On the other end they were terminated in connectors suitable to mate with the headers of the test board and the equipment outside the chamber.

In summary, the signal types passing through each cable are shown in Table 4.2 below. The technical details of the test harness along with the relevant pin lists are covered in [5].
### Test setup

<table>
<thead>
<tr>
<th>Cable reference designator</th>
<th>Signals passing through</th>
</tr>
</thead>
<tbody>
<tr>
<td>A,C</td>
<td>Remote keypad</td>
</tr>
<tr>
<td>B,D</td>
<td>JTAG for the FPGA, 2 X UARTS (RS-232), 5 X USARTS, +8VDC, access to current sense resistors for power measurements on VCCA, VCCB power planes of the DUT.</td>
</tr>
</tbody>
</table>

*Table 4.2: Summary of signals passing through the test harness.*

Although the frequency of the signals passing through the cables is not too high, the length of each cable was carefully selected and kept to a minimum in order to maintain the signal integrity to maximum extend.

### 4.5 Data logging, programming and monitoring equipment

The test progress was monitored and logged through an external application (HyperTerminal) executing on a Windows™ host PC, connected to the validation board over an RS-232 link, as shown in Figure 4.6. A `.txt.` log file was generated for each test run and stored in the PC. Via the same PC it was possible to download a different configuration file to the FPGA without removing the belljar. FPGA configuration was performed using a JTAG interface, via an ALTERA® ByteBlaster™ USB programmer and the relevant application (Quartus™).

The programs targeting the DUT were written in the Flash-memory (AMIC2904) located on motherboard [3], using a universal external programmer from DATAMAN®, which was connected via a USB port with an auxiliary laptop PC. When the program of the DUT needed to be changed (e.g. between the tests, or during other functional tests), the belljar needed to be opened, the flash memory was removed from its ZIF socket on the motherboard and placed on the external programmer. After programming, the memory device was replaced in its socket on the motherboard.
4.6 FPGA H/W

The main purpose of the core hardware implemented inside the FPGA was the proper routing of the signals between the DUT and the peripheral devices (RS-232 transceivers, memories, remote keypad, etc.).

In order to improve the monitoring and logging features of the DUT during the 2nd test run and as a backup to the existing setup, it was decided to replicate into the FPGA the counters maintained in memory, and to be triggered by the test whenever the corresponding internal counter was incremented. Three counters along with the necessary logic were implemented inside the FPGA. The first two have been used for counting the memory corrected and uncorrected errors, whereas the third one has been used to count the number of arithmetic errors. Additionally, it was decided to store the memory address where the last event occurred in a register and display it on the LCD along with the counters, as mentioned in section 4.4.2.

A watchdog timer in the FPGA that upon expiration would reset the uC was also under consideration for implementation. The watchdog would be reset at regular time intervals by the test software. Given the time restrictions this feature was not implemented. All the additional HW was developed during day-2.
5 Test Software Description

5.1 General Information

Two test applications were developed using KEIL μVision3 v3.60 IDE. Both applications were implemented in assembly language due to the erroneous behavior when compiled directly from C code. Specifically, when the EDAC feature on internal memory is enabled, some variables -counters keeping track of detected errors- are affected and have their value inadvertently changed. This is obviously a case of memory access violation, most likely attributed to the pre-compiled libraries that are automatically linked into the executable.

5.2 Memory Test

5.2.1 Brief Description

The memory test initializes an array of 50 bytes in internal memory and another one of 50 bytes in XRAM. The value of each element in either array is its index itself. An interrupt routine for interrupt INMEMER is installed to count and correct detected errors. Both arrays are accessed sequentially in order to trigger the EDAC. Once all elements of both arrays have been accessed, the number of detected errors is sent through the UART and the test is repeated infinitely.

5.2.2 Detailed Description

The test sets-up parallel ports P1, P2 and P3 which are used for output with the exception of pins P3.0 and P3.1 that are reserved for the UART interface which is clocked at 9600 baud by Timer-2. Both EDAC and XRAM are enabled and then an array of 50 bytes in internal memory and another one of 50 bytes in XRAM are initialized, where the value of each element in either array is its index.

Four bytes are reserved in the internal memory to implement the two error detection counters (2 bytes each). These counters are incremented by an interrupt routine triggered by the interrupt INMEMER. The handler will increment one of the two counters depending on whether the error is flagged as recoverable or not and then it will reset register MPSTAT. If the error is recoverable, a pulse is created on P3.2 while P1 contains the address at which the error was detected. If the error is unrecoverable a pulse is created on P3.3. If the interrupt handler is invoked and MPSTAT indicates no error, the handler exits without affecting the counters or creating a pulse on the pins of P3.

The main loop of the application consists of sequentially accessing of all elements in both arrays followed by sending through the UART the values of the counters in hexadecimal and then incrementing P2 to indicate that the test is functioning as expected.

The total amount of exercised memory is 50 bytes of XRAM and 63 bytes of internal memory of which 4 bytes are used by the general purpose registers, 5 by the stack, 2 bytes each of the two counters and 50 bytes for the array. The size of the array in the internal memory could be increased, provided that sufficient space is reserved for the stack so that it doesn't grow into the EDAC. For the initial test-run it was decided to be set at a conservative 50 bytes.

For the source code of the test please see Appendix D. The complete project may be extracted from file seu_mem_only.zip.

5.3 Memory & Arithmetic Test

5.3.1 Brief Description

This test extends the memory test with the inclusion of 8-bit arithmetic operations such as addition, multiplication and increment. 128 bytes, instead of 50. For the arithmetic operations, the calculated
values are compared against the expected values, which are hard-coded into the test program. Detected errors are accumulated in a counter whose value is sent through the UART along with the values of the memory related counters.

5.3.2 Detailed Description

The arithmetic test is focused on 8-bit integer arithmetic because it is directly supported by the hardware, whereas 16-bit, 32-bit or floating number arithmetic would require a software implementation of the arithmetic operators.

First we count from 0 up to 99 by incrementing a counter and comparing its value against the value of the accumulator, which has been loaded from code memory. The loop is controlled by another counter decremented to 0.

Then the addition procedure is checked by adding the values of two registers and comparing against a fixed value. The first register is initialized to 0x00 and the other to 0xFF. At each iteration the former is incremented whereas the later is decremented.

Finally, the multiplication is checked by calculating the squares of the first 20 numbers. At each iteration the same value is loaded in both A and B registers and the result of the multiplication is checked against hard-coded values stored in two arrays one for each register. In all cases, if the obtained value is not equal to the expected one, the error counter is incremented by one.

The total amount of exercised memory is 128 bytes of XRAM and 62 bytes of Internal memory of which 4 bytes are used by the general purpose registers, 5 by the stack, 1 byte each of the three counters and 50 bytes for the array.

For the source code of the test please see Appendix E. The complete project may be extracted from file seu_mem+math.zip.
6 Analysis Software Description

Due to the simplicity of the logging strategy, large log files are produced even for relatively small runs, thus requiring a parser to process the data. Since each test produces a unique output and in order to keep the application simple, two command line applications were built each one specific to a single test. The two applications have been compiled with gcc v4.1.2 on CentOS 5.

The source code (in C) of the two parsers and the logs are supplied in separate files under the following names (Table 6.1):

<table>
<thead>
<tr>
<th>Parser</th>
<th>Log File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>memonly.c</td>
<td>monday_12_01_2009.txt</td>
<td>First run</td>
</tr>
<tr>
<td>mem_math.c</td>
<td>tuesday_13_01_2009.txt</td>
<td>Second run</td>
</tr>
</tbody>
</table>

Table 6.1: Log files and associated parser application.

6.1 memonly

This application is for parsing the log files produced by the 'Memory' test. It accepts as input the name of the log file and outputs a file named after the log-file and extended with the string "-results.txt".

The application processes the log-file a line at a time. From each line we extract the value of the recoverable and unrecoverable error counters. If the counter changes value, the line number is used to estimate the time that the event took place. Both line number and timestamp along with the counters are recorded. If a counter is decremented, it is considered as an upset. Lines that do not match the expected format are ignored and are reported as anomalies into the output file.

6.2 mem_math

This application is for parsing the log files produced by the 'Memory & Arithmetic Test' and is similar in operation to the memonly application. It accepts as input the name of the log file and outputs a file named after the log-file and extended with the string "-results.txt". It will also produce a file with the extension "-anomalies.txt" containing only the timestamps of occurred anomalies and a file with the extension "-seu.txt" containing only timestamps and the value of counters in a simplified form. These two files are meant to be used for chart plotting thus they are devoid of any formatting unlike the results file.

The application processes the log-file a line at a time. From each line we extract the value of the recoverable, unrecoverable and arithmetic error counters. If a counter changes value the line number is used to estimated the time the event took place. Both line number and timestamp along with the counters are recorded. If a counter is decremented, it is considered as an upset. Lines that do not match the expected format are ignored and are reported as anomalies into the output file.

In order to guard against false upsets, the application also accepts an optional parameter indicating the bit-flip threshold below which a negative value change is ignored, provided that the low value appears right after the high value. If the parameter is not supplied the threshold is assumed to be 0, i.e. every change is recorded.
The value of the threshold is used in the naming of the output files by prefixing the file-type identification strings. For a log file named “tuesday_13_01_2009.txt” the following output is produced (Table 6.2):

<table>
<thead>
<tr>
<th>File Type</th>
<th>Threshold = 0 (Default)</th>
<th>Threshold = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>\texttt{tuesday\textunderscore 13\textunderscore 01\textunderscore 2009\textunderscore 0\textunderscore results.txt}</td>
<td>\texttt{tuesday\textunderscore 13\textunderscore 01\textunderscore 2009\textunderscore 1\textunderscore results.txt}</td>
</tr>
<tr>
<td>anomalies</td>
<td>\texttt{tuesday\textunderscore 13\textunderscore 01\textunderscore 2009\textunderscore 0\textunderscore anomalies.txt}</td>
<td>\texttt{tuesday\textunderscore 13\textunderscore 01\textunderscore 2009\textunderscore 1\textunderscore anomalies.txt}</td>
</tr>
<tr>
<td>seu</td>
<td>\texttt{tuesday\textunderscore 13\textunderscore 01\textunderscore 2009\textunderscore 0\textunderscore seu.txt}</td>
<td>\texttt{tuesday\textunderscore 13\textunderscore 01\textunderscore 2009\textunderscore 1\textunderscore seu.txt}</td>
</tr>
</tbody>
</table>

\textit{Table 6.2: Example of output files.}
7 Radiation procedure with SEU monitoring

The total duration of the Cf-252 radiation campaign was three days:

- Day-1: Monday, 12th of January 2009
- Day-2: Tuesday, 13th of January 2009
- Day-3: Wednesday, 14th of January 2009

During that period two overnight SEU radiation runs (test runs) were performed and the results of each test were logged onto separate files on the host PC via the HyperTerminal application. The following table summarizes the information regarding each test run.

<table>
<thead>
<tr>
<th>Test Run</th>
<th>Date/Time Started</th>
<th>Date/Time Terminated</th>
<th>Total Duration</th>
<th>Test Program</th>
<th>Analysis Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st RUN</td>
<td>Day-1/14:00(1)</td>
<td>Day-2/09:15(1)</td>
<td>19h 15m(2)</td>
<td>Memory Test</td>
<td>memonly</td>
</tr>
<tr>
<td>2nd RUN</td>
<td>Day-2/19:25</td>
<td>Day-3/14:56</td>
<td>19h 31m</td>
<td>Memory &amp; arithmetic test</td>
<td>mem_math</td>
</tr>
</tbody>
</table>

Table 7.1: Durations and test/analysis software for each run.

More details regarding each test run are presented in the following sections.

Notes:
1. The indicated time figures correspond to the start-stop times of the uC.
2. The time duration is referring to the total duration of the particular test run. It should be noted that the time duration of the uC producing legible data was approximately 12h (see §8.4.1).

7.1 1st run

The first test initiated on day-1 at 14:00 (power-up) and terminated on day-2 at 09:15 (power-down). The first test involved testing the EDAC capability with respect to the internal memory (imem) and XRAM using the ‘Memory Test’ software (§5.2). Using the ‘memonly’ application (§6.1), the analyzed data were sent over the UART interface to the external application (HyperTerminal) for logging into a file.

On day-2, 08:50 it was discovered that the HyperTerminal had stopped receiving data, although the uC was still transmitting. Transmission was confirmed by monitoring the RS-232 traffic through another application. This application showed that the uC was producing a repetitive pattern of characters, though carrying no resemblance to the output format coded in the test software.

Based on the size of the log-file it was estimated that HyperTerminal had ceased logging early in the morning of day-2. This estimate was later confirmed by the log-file analysis (see §8.4.1).

Given time restrictions, it was decided the next test run to be focused on running a single test for a long enough period to collect sufficient statistical data, rather than attempt to exercise many features by running multiple tests. In order to improve the logging capability of the test setup, it was decided to
7.2 2nd run

The belljar was opened, the onboard flash memory was removed, reprogrammed and replaced inside the socket on the motherboard.

The second test initiated on day-2 at 19:25 and terminated on day-3 at 14:56, about 19.5 hours. The second test involved testing the EDAC capability with respect to the internal memory (imem) and XRAM plus arithmetic operations in 8bit arithmetic using the 'Memory & Arithmetic Test' software (§5.3). Using the 'mem_math' application (§6.2), the results of program execution were delivered into a PC via the RS-232 port and logged in a new file.

At 09:50 local time it was observed that the internal counter was reset to 0 whereas the counter on the FPGA indicated 36 errors.

Since the test carried-on, it was assumed that the instruction sequence had been altered by jumping back at some point before the reset of the counters, either because the PC, or the instruction fetch or instruction decode unit had been affected. It is unlikely for the uC to have been reset because the start message sent at the beginning of the test was not found in the log-file around the time the event took place.

7.3 ISD S.A. Responsibilities

ISD S.A. was responsible for supplying the test board, compatible cables to interface with the CASE system (test harness), power supplies and the monitoring, programming and logging equipment for the experiments. ISD S.A was also responsible for assembling and mounting up the equipment and for performing the actual testing and log the results.

7.4 ESTEC CASE responsibilities

A qualified operator from ESTEC handled the radioactive material, sealed the vacuum chamber by placing both the belljar and the implosion guard above the baseplate and started the vacuum pump. After each radiation run had been terminated, he restored the atmospheric pressure inside the chamber by releasing the valve, removed the belljar-implosion guard and placed the radioactive source in a safe location.
8 Test Results

Because of the various uncertainties in the basic activity, thickness of the gold layer over the radiation source, the distance and penetration depth to the DUT, ESA-ESTEC recommends that a flat uncertainty of ±42% to be applied to the SEU rate results obtained from the CASE facility [8]. This means that we cannot expect to know the source activity to better than 42%.

In every case, it should be noted that the CASE facility can be considered as a pre-test for a Heavy Ion radiation campaign, as the results obtained here exhibit a significant uncertainty.

8.1 Summary of test results

In both test runs only corrected errors have been detected (i.e. less than three bit-flips) and no other errors have been indicated by either the FPGA or internal counters. Given the test conditions (vacuum, radiation source) and DUT technology (0.5um), the number and type of detected errors can be considered as both logical and expected.

The two interrupts on software’s normal execution (crashes on both test runs) may be assumed to be SEU events in flip-flops due to irradiation, as only a subset of the flip-flops is SEU hardened.

The test results for each radiation run are summarized in Table 8.1:

<table>
<thead>
<tr>
<th></th>
<th>1st RUN</th>
<th>2nd RUN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total duration (t)</td>
<td>12h</td>
<td>19h 31m</td>
</tr>
<tr>
<td>Corrected RAM errors (e)</td>
<td>7</td>
<td>42</td>
</tr>
<tr>
<td>Uncorrected RAM errors</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Possible flip-flop SEU events</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Upset rate $e_r$ (upsets/sec)</td>
<td>$1.62 \times 10^{-4}$</td>
<td>$5.98 \times 10^{-4}$</td>
</tr>
<tr>
<td>Test cross-section $X$ (cm$^2$)</td>
<td>$3.90 \times 10^{-6}$</td>
<td>$1.43 \times 10^{-5}$</td>
</tr>
<tr>
<td>Cross-section per bit $X_b$ (cm$^2$/bit)</td>
<td>$4.31 \times 10^{-9}$</td>
<td>$9.40 \times 10^{-9}$</td>
</tr>
</tbody>
</table>

Table 8.1: Summary of test results.

8.2 Upset rate

This section presents the upset rate (error rate) calculations in upsets/sec for both test runs.

8.2.1 1st Run

The duration of the run (as described in chapter 7) is about 12h.

The test duration ($t_1$) expressed in seconds is: $t_1 = 12 \times 3600$ sec = 43,200 sec

Since there were no uncorrected RAM errors detected, the calculation takes into account the number of corrected errors. The total number of corrected RAM errors (upsets) $e$ during the run was seven ($e_1 = 7$) (Table 8.1).

The upset rate ($e_r$) is calculated by dividing the total number of upsets with the test duration, as shown in (1)
2nd Run

The duration of the run (as described in chapter 7) is about 19.5h.
The test duration ($t_2$) in seconds is: $t_2 = 19.5 \times 3600$ sec = 70,200 sec
The total number of corrected RAM errors during the run was forty two ($e_2 = 42$) (Table 8.1).
The upset rate ($e_2$) is calculated by dividing the total number of upsets with the test duration, as shown in (2)

$$ e_2 = \frac{e_2}{t_2} = \frac{42 \text{upsets}}{70,200 \text{sec}} = 5.98 \times 10^{-4} \text{upsets/sec} \quad (2) $$

8.3 Cross-section

This section presents the cross-section calculations (in cm$^2$) for both test runs.

8.3.1 Test cross-section

The cross-section (X-section) is defined as the upset rate ($e_i$) divided by the flux ($\Phi$) of the radiation source.

The flux, among others, depends on the age of the radiation source [8]. As reported by ESA-ESTEC, the Cf-252 source used in both tests had a flux $\Phi$ of about 2500 particles/(cm$^2$ * min) - which is equivalent to 41.6 particles/(cm$^2$ * sec) - at a distance $d$ of about 10mm between the source and the DUT die surface. Because $d = 10$mm during the tests, we can use the above flux figure in the calculations.

The X-section $X_1$ for the 1st run is calculated below (3):

$$ X_1 = \frac{e_1}{\Phi} = \frac{1.62 \times 10^{-4} \text{upsets/sec}}{41.6 \text{particles/(cm}^2\cdot\text{sec)}} = 3.90 \times 10^{-6} \text{cm}^2 \quad (3) $$

$X_1$ is the X-section of the DUT, exercised with the memory test program (see §5.2).

The X-section $X_2$ for the 2nd run is calculated below (4):

$$ X_2 = \frac{e_2}{\Phi} = \frac{5.98 \times 10^{-4} \text{upsets/sec}}{41.6 \text{particles/(cm}^2\cdot\text{sec)}} = 1.43 \times 10^{-5} \text{cm}^2 \quad (4) $$

$X_2$ is the X-section of the DUT, exercised with the memory and arithmetic test program (see §5.3).

8.3.2 Cross-section per bit

The X-section per bit $X_{b1}$ for a particular test is defined as the test X-section ($X$) divided by the total number of bits $n$ covered during that test.

The 1st run was performed using the memory test software which covers 63 bytes in internal memory plus 50 bytes in the XRAM. Thus the total number of bytes is 63+50=113bytes = 904bits = $n_1$.

The X-section per bit $X_{b1}$ for the 1st run is calculated below (5):
The 2nd run was performed using the memory plus arithmetic test software which covers 62 bytes in internal memory plus 128 bytes in the XRAM. Thus the total number of bytes is 62+128=190bytes = 1520bits = $n_2$.

The X-section per bit $X_{b2}$ for the 2nd run is calculated below (6):

$$X_{b2} = \frac{X_2}{n_2} = \frac{1.43 \times 10^{-5} \text{cm}^2}{1520 \text{bits}} = 9.40 \times 10^{-9} \text{cm}^2 / \text{bit} \quad (6)$$

### 8.3.3 Comparison with ASIC vendor-provided cross-sections

In [7], Atmel specifies the saturated cross-section of the MG2RT technology as $3 \times 10^{-7} \text{cm}^2$/bit. This means that our experimental cross-sections are about 30 – 70 times lower than values provided by Atmel. The exact reasons could not be determined, but the following issues possibly contribute to the discrepancy:

- There is an uncertainty (factor 2) on the true activity of the Californium source which depends on the age of the source.
- With an average LET of Cf-252 of 43 MeV $\text{cm}^2 / \text{mg}$, we do not fully reach the saturated cross-section (value of the horizontal asymptotic in the figure below). Experimental cross-sections observed during testing are lower than the saturated cross-sections obtained specified in the vendor documents. In addition, cross-sections obtained with Cf-252 are always a bit lower than cross-sections obtained in cyclotron heavy ion test. A possible reason for this is the insufficient penetration depth of the Cf-252 fission products. These effects can cumulate to a discrepancy of 30-40%.
- The distance between source and die could not be exactly measured. Increasing from 1 cm to 1.4 cm reduces the flux at the die surface by a factor of 2.
- Atmel figures are specified at Vcc = 4.5V. At 5V in our cases, the cross-section is reduced.
8.4 Detailed test results

8.4.1 1st Run – Memory Test

The following is an extract of the results file produced by application memonly when passed the log file produced by the memory test on Monday 12/01/2009 (Day-1). The effective duration of the test, within valid output data were collected, was about 12h (11h 48min).

================================================================================
Log filename: monday_12_01_2009
Number of Lines: 1800949
  Estimating: 23.6 sec / 1000 iter
Test Duration (hh:mm:ss): 11:48:22
================================================================================
Recoverable Errors: 7
  Un-Recoverable Errors: 0
  Number of Upsets: 0
  Number of anomalies: 236
================================================================================
  83823 (00:32:58), R=1 U=0
  181053 (01:11:12), R=2 U=0
  193245 (01:16:00), R=3 U=0
  505782 (03:18:56), R=4 U=0
  1166537 (07:38:50), R=5 U=0
  1174788 (07:42:04), R=6 U=0
  1622572 (10:38:12), R=7 U=0
================================================================================

Figure 8.1: Cross-section per bit vs LET (Atmel)
Test Results

Figure 8.2: Errors Running Total for Memory-Test

Figure 8.3: Errors per time interval for Memory-Test (time interval = 1 hour).
8.4.2 2nd Run – Memory & Arithmetic Test

The following is an extract of the results file produced by application `mem_math` when passed the log file produced by the memory & arithmetic test on Tuesday 13/01/2009 (Day-2). The arrows on the right-hand side indicate the negative value changes after the elimination of single-bit-flip changes in consecutive lines.

Note that the remaining upsets are all single-bit-flips furthermore all bit-flips occurred at the second bit of the first ASCII character representing the value of the recoverable counter. Thus it is safe to ignore them when plotting charts (Figure 8.4 & Figure 8.5).

The number of errors indicated in first part of the report, reflects the last legible counter values extracted from the log-file. These are not necessary the total number of errors (hence the second and more detailed part of the report) since that would require a more advanced parser.

The number of upsets represents the negative transitions that have been detected, however not all marked because some might have been ignored depending on the threshold value. This information is presented at the end of the report.

=================================================================
Log filename: tuesday_13_01_2009
Number of Lines: 2124652
Estimating: 33.1 sec / 1000 iter
Test Duration (hh:mm:ss): 19:32:05
=================================================================
Recoverable Errors: 10
Un-Recoverable Errors: 0
Arithmetic Errors: 0
Number of Upsets: 17
Number of anomalies: 41
=================================================================

<table>
<thead>
<tr>
<th>Time</th>
<th>Action</th>
<th>R</th>
<th>U</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00</td>
<td>START</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>00:00:07</td>
<td>START</td>
<td>R=1 U=0 M=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01:18:06</td>
<td>R=1 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00:44:47</td>
<td>R=2 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00:04:34</td>
<td>R=22 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01:10:21</td>
<td>R=2 U=0 M=0</td>
<td>&lt;&lt;&lt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00:34:27</td>
<td>R=3 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01:39:26</td>
<td>R=4 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01:59:42</td>
<td>R=24 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02:05:30</td>
<td>R=4 U=0 M=0</td>
<td>&lt;&lt;&lt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>02:26:35</td>
<td>R=5 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03:11:58</td>
<td>R=8 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03:42:15</td>
<td>R=28 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03:42:45</td>
<td>R=8 U=0 M=0</td>
<td>&lt;&lt;&lt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04:01:36</td>
<td>R=28 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04:21:36</td>
<td>R=8 U=0 M=0</td>
<td>&lt;&lt;&lt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04:43:41</td>
<td>R=9 U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>05:01:36</td>
<td>R=A U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>05:02:53</td>
<td>R=B U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>05:11:30</td>
<td>R=2B U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>05:13:21</td>
<td>R=B U=0 M=0</td>
<td>&lt;&lt;&lt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>05:47:08</td>
<td>R=C U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>06:17:03</td>
<td>R=D U=0 M=0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Test Results

728499 (06:41:53) ->  R=E  U=0  M=0
783915 (07:12:27) ->  R=F  U=0  M=0
784655 (07:12:52) ->  R=10 U=0  M=0
843583 (07:45:22) ->  R=30 U=0  M=0
873492 (08:01:52) ->  R=31 U=0  M=0
908260 (08:21:03) ->  R=32 U=0  M=0
915522 (08:25:03) ->  R=12 U=0  M=0 <<<
924409 (08:29:57) ->  R=13 U=0  M=0
937872 (08:37:23) ->  R=14 U=0  M=0
968757 (08:54:25) ->  R=15 U=0  M=0
1006154 (09:15:03) ->  R=16 U=0  M=0
1024008 (09:24:54) ->  R=17 U=0  M=0
1044899 (09:36:26) ->  R=18 U=0  M=0
1053067 (09:40:56) ->  R=38 U=0  M=0
1072101 (09:51:26) ->  R=39 U=0  M=0
1120464 (10:18:07) ->  R=3A U=0  M=0
1126973 (10:21:42) ->  R=3B U=0  M=0
1187655 (10:55:11) ->  R=3C U=0  M=0 <<<
1217234 (11:11:30) ->  R=1C U=0  M=0 <<<
1234175 (11:20:51) ->  R=1D U=0  M=0
1375923 (12:39:03) ->  R=1E U=0  M=0
1497509 (13:46:07) ->  R=1F U=0  M=0
1503643 (13:49:30) ->  R=20 U=0  M=0
1530344 (14:04:14) ->  R=0 U=0  M=0 <<<
1581565 (14:32:29) ->  R=1 U=0  M=0
1652205 (15:11:27) ->  R=21 U=0  M=0
1677698 (15:25:31) ->  R=1 U=0  M=0 <<<
1769761 (16:16:19) ->  R=2 U=0  M=0
1826936 (16:47:51) ->  R=3 U=0  M=0
1858032 (17:05:00) ->  R=4 U=0  M=0
1907980 (17:32:34) ->  R=5 U=0  M=0
1922822 (17:40:45) ->  R=6 U=0  M=0
1936408 (17:48:15) ->  R=7 U=0  M=0
1936884 (17:48:30) ->  R=8 U=0  M=0
1959346 (18:00:54) ->  R=28 U=0  M=0
1980433 (18:12:32) ->  R=8 U=0  M=0 <<<
2016911 (18:32:39) ->  R=9 U=0  M=0
2116312 (19:27:29) ->  R=A U=0  M=0

Max flipped bits per byte: 1
Ignored SEUs: 7

==================================================
Company Confidential
Page 31 of 59
Figure 8.4: Detected errors running total for the Memory & Arithmetic test.

Figure 8.5: Detected errors per time interval for the Memory & Arithmetic test (time interval = 1 hour).
9 Conclusions

The purpose of this experiment was to make a first assessment of the radiation endurance of the 80S32 device. Considering the limited duration of the experiments, the lack of adequate statistics and the uncertainty of the radiation source, we come to the conclusion that both trials give similar results in terms of cross section per bit. The obtained cross-section of correctable errors is one to two orders of magnitude below figures reported by Atmel [7]. The exact reasons for this discrepancy could not be fully explained. The rate of uncorrectable memory errors and of unexplained test program crashes which could possibly indicate SEU errors in flip-flops is even far below the correctable error rate. In summary, the results indicate a very good radiation robustness of the 80S32.

9.1 Suggestions for future developments

A future version of the test software should strive to maximize the SW cross-section by exercising as many SEU sensitive bits as possible. Currently we use 50% of the available memory. To reach 100% utilization would require careful study of the dynamic behavior of the S/W since the memory is shared with the stack and one should be careful not to let it grow into the EDAC region. The current test s/w could be modified to achieve a higher utilization by increasing the size of the array stored in the internal memory.

To improve the robustness of the test strategy, all error counting and logging should be performed outside the DUT. Otherwise, the counters themselves could be affected, thus presenting us with erroneous data. Furthermore, in the case of a latch-up or other severe event, all data would be lost.

The first test (mem_only) relied exclusively on internal counters to record the errors, though it passed this information at regular intervals to an external application. The second test (mem_math) has been modified to support external logging along with the internal counters and rely on the FPGA for logging the number of errors along with their temporal and spatial data.

To further increase the robustness of the test strategy a watchdog could be implemented, allowing automatic reboot of the DUT. Such a feature would ensure optimal use of the available test-time without the presence of human operator. Had this feature being available during the first test-run we would have been able to recover from the SEU that caused the abrupt termination of logging.

Given the amount of time it is required for the vacuum in the radiation-chamber to reach the appropriate level, it would have been more practical if the test software of could be changed without opening the chamber. Although the uC features an In-System-Programming capability, this could not be used since it requires the use of the USART interface which has proven to be problematic. In a future setup, an external boot-loader stored in the FLASH could be used to download the application in the RAM and then set the uC in the “RAM Execution Mode”.
10 APPENDIX A – Test component photographs

Figure 10.1: 80S32 DUT lidded and top marking

Figure 10.2: 80S32 DUT de-lidded showing the exposed die inside the cavity.
Figure 10.3: 80S32 DUT inside the socket of the test board
11 APPENDIX B – Test setup photographs

Figure 11.1: Test board inside the chamber. Belljar and implosion guard in place.

Figure 11.2: Cable details showing the vacuum release valve (yellow knob)
Figure 11.3: Test board on the baseplate. Source holder not in place.

Figure 11.4: Details of the test board showing the rubber feet and the IDC connectors.
Figure 11.5: General arrangement of the test setup
12 APPENDIX C – Details of all radiation runs

12.1 1st Run – Memory Test

<table>
<thead>
<tr>
<th>Line</th>
<th>Time</th>
<th>Corrected errors</th>
<th>Uncorrected errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>83823</td>
<td>00:32:58</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>181053</td>
<td>01:11:12</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>193245</td>
<td>01:16:00</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>505782</td>
<td>03:18:56</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>1166537</td>
<td>07:38:50</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>1174788</td>
<td>07:42:04</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>1622572</td>
<td>10:38:12</td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

12.2 2nd Run – Memory & Arithmetic Test

Ignore single bit-flips in successive entries.

<table>
<thead>
<tr>
<th>Line</th>
<th>Time</th>
<th>Corrected errors (Hex)</th>
<th>Uncorrected errors</th>
<th>Arithmetic Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>32830</td>
<td>(00:18:06)</td>
<td>R=1</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>81195</td>
<td>(00:44:47)</td>
<td>R=2</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>117040</td>
<td>(01:04:34)</td>
<td>R=22</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>127538</td>
<td>(01:10:21)</td>
<td>R=2</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>171221</td>
<td>(01:34:27)</td>
<td>R=3</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>180269</td>
<td>(01:39:26)</td>
<td>R=4</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>217002</td>
<td>(01:59:42)</td>
<td>R=24</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>227495</td>
<td>(02:05:30)</td>
<td>R=4</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>265717</td>
<td>(02:26:35)</td>
<td>R=5</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>274632</td>
<td>(02:31:30)</td>
<td>R=6</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>312439</td>
<td>(02:52:21)</td>
<td>R=7</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>348005</td>
<td>(03:11:58)</td>
<td>R=8</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>402899</td>
<td>(03:42:15)</td>
<td>R=28</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>403793</td>
<td>(03:42:45)</td>
<td>R=8</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>437946</td>
<td>(04:01:36)</td>
<td>R=28</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>474216</td>
<td>(04:21:36)</td>
<td>R=8</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>514238</td>
<td>(04:43:41)</td>
<td>R=9</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>528607</td>
<td>(04:51:36)</td>
<td>R=A</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>549054</td>
<td>(05:02:53)</td>
<td>R=B</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>564656</td>
<td>(05:11:30)</td>
<td>R=2B</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>568020</td>
<td>(05:13:21)</td>
<td>R=B</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>629263</td>
<td>(05:47:08)</td>
<td>R=C</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>683495</td>
<td>(06:17:03)</td>
<td>R=D</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>728499</td>
<td>(06:41:53)</td>
<td>R=E</td>
<td>U=0</td>
<td>M=0</td>
</tr>
<tr>
<td>783915</td>
<td>(07:12:27)</td>
<td>R=F</td>
<td>U=0</td>
<td>M=0</td>
</tr>
</tbody>
</table>
Validation of 80S32 Microcontroller

APPENDIX C – Details of all radiation runs

<table>
<thead>
<tr>
<th>Time</th>
<th>Corrected errors (Hex)</th>
<th>Uncorrected errors</th>
<th>Arithmetic Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:18:06</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>00:44:47</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>01:34:27</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>01:39:26</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>02:26:35</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>02:31:30</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>02:52:21</td>
<td>7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>03:11:58</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Ignore & Correct single bit-flips in bit-5 of R counter
<table>
<thead>
<tr>
<th>Run</th>
<th>Start Time</th>
<th>Duration</th>
<th>Rad Level</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>514238</td>
<td>04:43:41</td>
<td>9</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>528607</td>
<td>04:51:36</td>
<td>A</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>549054</td>
<td>05:02:53</td>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>629263</td>
<td>05:47:08</td>
<td>C</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>683495</td>
<td>06:17:03</td>
<td>D</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>728499</td>
<td>06:41:53</td>
<td>E</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>783915</td>
<td>07:12:27</td>
<td>F</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>784655</td>
<td>07:12:52</td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>873492</td>
<td>08:01:52</td>
<td>11</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>908260</td>
<td>08:21:03</td>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>924409</td>
<td>08:29:57</td>
<td>13</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>937872</td>
<td>08:37:23</td>
<td>14</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>968757</td>
<td>08:54:25</td>
<td>15</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1006154</td>
<td>09:15:03</td>
<td>16</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1024008</td>
<td>09:24:54</td>
<td>17</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1044899</td>
<td>09:36:26</td>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1072101</td>
<td>09:51:26</td>
<td>19</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1120464</td>
<td>10:18:07</td>
<td>1A</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1126973</td>
<td>10:21:42</td>
<td>1B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1187655</td>
<td>10:55:11</td>
<td>1C</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1234175</td>
<td>11:20:51</td>
<td>1D</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1375923</td>
<td>12:39:03</td>
<td>1E</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1497509</td>
<td>13:46:07</td>
<td>1F</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1503643</td>
<td>13:49:30</td>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1530344</td>
<td>14:04:14</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1581565</td>
<td>14:32:29</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1769761</td>
<td>16:16:19</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1826936</td>
<td>16:47:51</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1858032</td>
<td>17:05:00</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1907980</td>
<td>17:32:34</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1922822</td>
<td>17:40:45</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1936408</td>
<td>17:48:15</td>
<td>7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1936884</td>
<td>17:48:30</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2016911</td>
<td>18:32:39</td>
<td>9</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2116312</td>
<td>19:27:29</td>
<td>A</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
This Appendix includes the source code of the memory test program used during the 1st radiation run on Day-1.

```
NAME MAIN

MPCON DATA 094H
SX1BUFH DATA 0D5H
SX2BUFH DATA 0DCH
SX3BUFH DATA 0E5H
T2 BIT 090H.0
SX4BUFH DATA 0ECH
FIFOCON DATA 0C7H
SX1BUFL DATA 0D4H
CSA0 BIT 080H.0
SX2BUFL DATA 0D8H
SX3BUFL DATA 0E4H
SX4BUFL DATA 0EBH
EXF2 BIT 0CAH.6
WAITMEM DATA 0C3H
TDIVCON DATA 0B1H
INT4 BIT 0B0H.7
CRCH DATA 0A8H
RCAP2H DATA 0CBH
T2EX BIT 090H.1
CRCL DATA 0AAH
RCAP2L DATA 0CAH
C_T2 BIT 0CAH.1
RCLK BIT 0CAH.5
TCLK BIT 0CAH.4
PDERAD DATA 0A1H
P20 BIT 0A0H.0
P12 BIT 090H.2
P21 BIT 0A0H.1
P22 BIT 0A0H.2
PGBKANK DATA 0B5H
P23 BIT 0A0H.3
RX BIT 0B0H.0
TX BIT 0B0H.1
P32 BIT 0B0H.2
DTBANK DATA 0B4H
P24 BIT 0A0H.4
P33 BIT 0B0H.3
P34 BIT 0B0H.4
P16 BIT 090H.6
TDIV DATA 0B2H
P17 BIT 090H.7
SX1CLK BIT 0A0H.7
SX2CLK BIT 090H.5
P36 BIT 0B0H.6
TXCON1 DATA 0B8H
IXERAD DATA 0BAH
TXCON2 DATA 0BBH
TXCON3 DATA 0BCH
```
Validation of 80S32 Microcontroller

SEU Radiation Test Report

APPENDIX D – Memory Test source code

SX1VAL    BIT  0A0H.6
SX2VAL    BIT  090H.4
CP_RL2    BIT  0C8H.0
ET2       BIT  0A8H.5
TF2       BIT  0C8H.7
TH2       DATA  0CDH
TL2       DATA  0CCH
SX1FREQH  DATA  0D7H
SX2FREQH  DATA  0DEH
PT2       BIT  0B8H.5
SX3FREQH  DATA  0E7H
SX4FREQH  DATA  0EEH
TR2       BIT  0C8H.2
SX1FREQL  DATA  0D6H
SX2FREQL  DATA  0DDH
EXTAD20   BIT  0B8H.3
SX3FREQL  DATA  0E6H
MPSTAT    DATA  095H
EXTAD21   BIT  0B8H.2
SX4FREQL  DATA  0EDH
EXTAD22   BIT  0B8H.1
EXTBUS    DATA  0B5H
EXTAD16   BIT  0B8H.7
P0DIR     DATA  0A4H
EXTAD17   BIT  0B8H.6
P1DIR     DATA  0A5H
SYSCON    DATA  093H
EXTAD18   BIT  0B8H.5
P1CON     DATA  0D8H
P2DIR     DATA  0A6H
EXTAD19   BIT  0B8H.4
P2CON     DATA  0E8H
P3DIR     DATA  0A7H
EXEN2     BIT  0C8H.3
P3CON     DATA  0F8H
T2CON     DATA  0C8H
IXEP1     DATA  0ACH
IXEP2     DATA  0ADH
IXEP3     DATA  0AEH
SX1CON1   DATA  0D2H
SX2CON1   DATA  0D9H
SX1CON2   DATA  0D3H
CRCIN     DATA  0A9H
SX3CON1   DATA  0E2H
SX2CON2   DATA  0DAH
SX4CON1   DATA  0E9H
SX3CON2   DATA  0E3H
SX4CON2   DATA  0EAH
EMCON     DATA  096H
SX1DT     BIT  0A0H.5
SX2DT     BIT  090H.3
CSCON     DATA  09EH
DPSEL     DATA  092H

?STACK               SEGMENT DATA
segData             SEGMENT DATA
segXData            SEGMENT XDATA
segAppli            SEGMENT CODE
segInterrupt SEGMENT CODE
segStrRec SEGMENT CODE
segStrUnRec SEGMENT CODE
segStrNum SEGMENT CODE

; ========= CONSTANTS ===============
IMEMSIZE EQU 050D
XRAMSIZE EQU 0128D

; ========= ADDRESS 0x0000 ==========
CSEG AT 00H
JMP main

CSEG AT 00043H
LJMP imemInt

; ========= Populate Segments ========
RSEG STACK
DS 10

RSEG segStrRec
rec_string: DB 'r','e','c',';'

RSEG segStrUnRec
unrec_string: DB ' ',' ','u','n','r','e','c',';'

RSEG segStrNum
hex: DB '0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'

RSEG segXData
xcell: DS XRAMSIZE

RSEG segData
cell: DS IMEMSIZE
rec_cnt: DS 2
unrec_cnt: DS 2

RSEG segInterrupt
USING 0

imemInt:
PUSH ACC
PUSH PSW
PUSH AR1

; Check Internal Mem - Recoverable
MOV A,MPSTAT
ANL A,#03H
CJNE A,#02H,xram

SETB P32
MOV R1, IXERAD
MOV A, @R1
MOV @R1, A
CLR P32

; increment error counter
INC rec_cnt+01H
MOV A,rec_cnt+01H
JNZ rec_no_overflow
INC rec_cnt
rec_no_overflow:
    JMP intend

; Check XRAM - Recoverable
xram:
    MOV A,MPSTAT
    RR A
    RR A
    ANL A,#03H
    CJNE A,#02H,unrec
    SETB P32
    MOV R1, IXERAD
    MOVX A, @R1
    MOVX @R1, A
    CLR P32
    ; increment error counter
    INC rec_cnt+01H
    MOV A,rec_cnt+01H
    JNZ rec_no_overflow_b
    INC rec_cnt
rec_no_overflow_b:
    JMP intend

; Check Internal Mem - Unrecoverable
unrec:
    MOV A,MPSTAT
    CJNE A,#01H,intend
    SETB P33
    NOP
    CLR P33
    ; increment error counter
    INC unrec_cnt+01H
    MOV A,unrec_cnt+01H
    JNZ intend
    INC unrec_cnt
intend:
    MOV MPSTAT,#0FFH
    POP AR1
    POP PSW
    POP ACC
    RETI

RSEG segAppli
USING 0

main:
; Initialise SP
    MOV SP,#?STACK

; Configure P1, P2 & P3 for output
    MOV PICON,#00H
    MOV P1DIR,#0FFH
    MOV P2CON,#00H
    MOV P2DIR,#0FFH
    MOV P3CON,#03H
MOV P3DIR,#0FFH

; Init P2, it also helps to full the debugger due to P2 not being used
; for addressing in 80S32 as in Intel's 8052
MOV P1,#00D
MOV P2,#00D
CLR P32
CLR P33
CLR P34

; Configure UART to use Timer-2
MOV SCON,#050H  ; 0x52
MOV T2CON,#034H
MOV RCAP2L,#0B8H
MOV RCAP2H,#0FFH

; Enable Internal EDAC Interrupt (High Priority)
SETB EA
MOV IXEP1,#03H

; Enable XRAM
MOV SYSCON,#03H

; Enable EDAC for IntMem + XRAM
MOV MPCON,#0FH

; Initialize internal memory
MOV A,#00D
MOV R2,#IMEMSIZE
MOV R1,#cell
init_mem:
    MOV @R1, A
    INC A
    INC R1
    DJNZ R2, init_mem

; Initialize XRAM
MOV A,#00D
MOV R2,#XRAMSIZE
MOV R0,#xcell
init_xram:
    MOVX @R0, A
    INC A
    INC R0
    DJNZ R2, init_xram

; Initialize error counters
CLR A
MOV rec_cnt,A
MOV rec_cnt+01H,A

MOV unrec_cnt,A
MOV unrec_cnt+01H,A

; Check XRAM
test:
    MOV R2,#050D
MOV R3,#00D
MOV R0,#xcell

check_xram:
    MOVX A, @R0
    XRL A, R3
    INC R0
    INC R3
    DJNZ R2, check_xram

; Check Internal memory
MOV R2,#050D
MOV R3,#00D
MOV R0,#cell

check_imem:
    MOV A, @R0
    XRL A, R3
    INC R0
    INC R3
    DJNZ R2, check_imem

; Print results
CALL print_rec
CALL print_unrec

INC P2 ; we may monitor P2 through analyzer to confirm liveness manually

SETB P34 ; added after radiation test on Monday-12-01-2009
NOP       ; for the purposes of timing the main-program loop
CLR P34  ; with the aid of the FPGA

JMP test

; Function print_rec()

print_rec:
    MOV R3, #04D
    MOV DPTR,#rec_string

loop_rec:
    CLR A
    MOVC A,@A+DPTR
    CALL putchar
    INC DPTR
    DJNZ R3, loop_rec

; print high order nibble of MSB
MOV A, rec_cnt
SWAP A
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print low order nibble of MSB
MOV A, rec_cnt
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print high order nibble of LSB
MOV A, rec_cnt+01H
SWAP A
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print low order nibble of LSB
MOV A, rec_cnt+01H
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

RET

; Function print_unrec()
print_unrec:
MOV R3, #08D
MOV DPTR,#unrec_string
loop_unrec:
CLR A
MOVC A,@A+DPTR
CALL putchar
INC DPTR
DJNZ R3, loop_unrec

; print high order nibble of MSB
MOV A, unrec_cnt
SWAP A
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print low order nibble of MSB
MOV A, unrec_cnt
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print high order nibble of LSB
MOV A, unrec_cnt+01H
SWAP A
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print low order nibble of LSB
MOV A, unrec_cnt+01H
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

MOV A,#0AH
CALL putchar
MOV A,#0DH
CALL putchar
RET

; Function putchar()
putchar:
MOV SBUF,A
JNB TI, $
CLR TI
RET

END
This Appendix includes the source code of the memory plus arithmetic test program used during the 2nd radiation run on Day-2.

```
NAME MAIN
MPCON DATA 094H
SX1BUFH DATA 0D5H
SX2BUFH DATA 0DCH
SX3BUFH DATA 0E5H
T2 BIT 090H.0
SX4BUFH DATA 0ECH
FIFOCON DATA 0C7H
SX1BUFL DATA 0D4H
CSA0 BIT 080H.0
SX2BUFL DATA 0DBH
SX3BUFL DATA 0E4H
SX4BUFL DATA 0EBH
EXF2 BIT 0C8H.6
WAITMEM DATA 0C3H
TDIVCON DATA 0B1H
INT4 BIT 0B0H.7
CRCH DATA 0ABH
RCAP2H DATA 0CBH
T2EX BIT 090H.1
CRCL DATA 0AAH
RCAP2L DATA 0CAH
C_T2 BIT 0C8H.1
RCLK BIT 0C8H.5
TCLK BIT 0C8H.4
PDERAD DATA 0A1H
P20 BIT 0A0H.0
P12 BIT 090H.2
P21 BIT 0A0H.1
P22 BIT 0A0H.2
PGBANK DATA 085H
P23 BIT 0A0H.3
RX BIT 0B0H.0
TX BIT 0B0H.1
P32 BIT 0B0H.2
DTBANK DATA 084H
P24 BIT 0A0H.4
P33 BIT 0B0H.3
P34 BIT 0B0H.4
P35 BIT 0B0H.5
P37 BIT 0B0H.7
P16 BIT 090H.6
TDIV DATA 0B2H
P17 BIT 090H.7
SX1CLK BIT 0A0H.7
SX2CLK BIT 090H.5
P36 BIT 0B0H.6
TXCON1 DATA 0BAH
IXERAD DATA 09AH
TXCON2 DATA 0BBH
TXCON3 DATA 0BCH
```

 SX1VAL   BIT  0A0H.6  
 SX2VAL   BIT  090H.4  
 CP_RL2   BIT  0C8H.0  
 ET2      BIT  0A8H.5  
 TF2      BIT  0C8H.7  
 TH2      DATA  0CDH  
 TL2      DATA  0CCH  
 SX1FREQH DATA  0D7H  
 SX2FREQH DATA  0DEH  
 PT2      BIT  088H.5  
 SX3FREQH DATA  0E7H  
 SX4FREQH DATA  0EEH  
 TR2      BIT  0C8H.2  
 SX1FREQL DATA  0D6H  
 SX2FREQL DATA  0DDH  
 EXTAD20  BIT  080H.3  
 SX3FREQL DATA  0E6H  
 MPSTAT   DATA  095H  
 EXTAD21  BIT  080H.2  
 SX4FREQL DATA  0EDH  
 EXTAD22  BIT  080H.1  
 EXTBUS   DATA  0B5H  
 EXTAD16  BIT  080H.7  
 P0DIR    DATA  0A4H  
 EXTAD17  BIT  080H.6  
 P1DIR    DATA  0A5H  
 SYSCON   DATA  093H  
 EXTAD18  BIT  080H.5  
 P1CON    DATA  0D8H  
 P2DIR    DATA  0A6H  
 EXTAD19  BIT  080H.4  
 P2CON    DATA  0E8H  
 P3DIR    DATA  0A7H  
 EXEN2    BIT  0C8H.3  
 P3CON    DATA  0F8H  
 T2CON    DATA  0C8H  
 IXEP1    DATA  0ACH  
 IXEP2    DATA  0ADH  
 IXEP3    DATA  0AEH  
 SX1CON1  DATA  0D2H  
 SX2CON1  DATA  0D9H  
 SX1CON2  DATA  0D3H  
 CRCIN    DATA  0A9H  
 SX3CON1  DATA  0E2H  
 SX2CON2  DATA  0DAH  
 SX4CON1  DATA  0E9H  
 SX3CON2  DATA  0E3H  
 SX4CON2  DATA  0EAH  
 EMCON    DATA  096H  
 SX1DT    BIT  0A0H.5  
 SX2DT    BIT  090H.3  
 CCON     DATA  09EH  
 DPSEL    DATA  092H  

?STACK   SEGMENT DATA  
segData   SEGMENT DATA  
segXData  SEGMENT XDATA  
segAppli  SEGMENT CODE
segMath SEGMENT CODE
segInterrupt SEGMENT CODE
segStrings SEGMENT CODE
segArrays SEGMENT CODE

; ========== CONSTANTS ===============
IMEMSIZE EQU 050D
XRAMSIZE EQU 0128D

; ========== ADDRESS 0x0000 =============
CSEG AT 00H
JMP main

CSEG AT 00043H
LJMP imemInt

; ====== Populate Segments ========
RSEG ?STACK
DS 10

RSEG segStrings
start_string: DB 'start',0Ah,0Dh
rec_string: DB 'rec:'
unrec_string: DB ' unrec:'
math_string: DB 'math:'
hex: DB '0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'

RSEG segArrays
array_a: DB 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31
,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,
54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,
74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89

a_comp: DB 0h,1h,4h,9h,10h,19h,24h,31h,40h,51h,64h,79h,90h,0A9h,0C4h,0E1h,0h,21h,44h,69h
b_comp: DB 0h,0h,0h,0h,0h,0h,0h,0h,0h,0h,0h,0h,0h,0h,1h,1h,1h

RSEG segXData
xcell: DS XRAMSIZE

RSEG segData
cell: DS IMEMSIZE
rec_cnt: DS 1
unrec_cnt: DS 1
math_cnt: DS 1

;=================================================================================
; INTERRUPT HANDLER
;=================================================================================
RSEG segInterrupt
USING 0
imemInt:
PUSH ACC
PUSH PSW
PUSH AR1

; Assign address to P1
MOV P1, IXERAD

; Check Internal Mem - Recoverable
MOV A, MPSTAT
ANL A, #03H
CJNE A, #02H, xram

SETB P32
MOV R1, IXERAD
MOV A, @R1
MOV @R1, A
CLR P32

; increment error counter
INC rec_cnt
JMP intend

; Check XRAM - Recoverable
xram:
MOV A, MPSTAT
RR A
RR A
ANL A, #03H
CJNE A, #02H, unrec

SETB P32
MOV R1, IXERAD
MOVX A, @R1
MOVX @R1, A
CLR P32

; increment error counter
INC rec_cnt
JMP intend

; Check Internal Mem - Unrecoverable
unrec:
MOV A, MPSTAT
CJNE A, #01H, intend

SETB P33
NOP
CLR P33

; increment error counter
INC unrec_cnt
intend:
MOV MPSTAT, #0FFH
POP AR1
POP PSW
POP ACC
RETI

;=====================================================================
; MAIN TEST
;
;===================================================================================================
RSEG segAppli
USING 0

main:
; Initialise SP
MOV SP, #0?STACK

; Configure P1, P2 & P3 for output
MOV P1CON, #00H
MOV P1DIR, #0FFH
MOV P2CON, #00H
MOV P2DIR, #0FFH
MOV P3CON, #03H
MOV P3DIR, #0FFH

; Init P2, it also helps to full the debugger due to P2 not being used
; for addressing in 80S32 as in Intel's 8052
MOV P1, #00D
MOV P2, #00D
CLR P32
CLR P33
CLR P34
CLR P35
CLR P36
CLR P37

; Configure UART to use Timer-2
MOV SCON, #050H
MOV T2CON, #034H
MOV RCAP2L, #0B8H
MOV RCAP2H, #0FFH

; Print start-up string
CALL print_start

; Enable Internal EDAC Interrupt (High Priority)
SETB EA
MOV IXEP1, #03H

; Enable XRAM
MOV SYSCON, #03H

; Enable EDAC for IntMem + XRAM
MOV MPCON, #0FH

; Initialize internal memory
MOV A, #00D
MOV R2, #IMEMSIZE
MOV R1, #cell
init_mem:
MOV @R1, A
INC A
INC R1
DJNZ R2, init_mem

; Initialize XRAM
MOV A, #00D
MOV R2, #XRAMSIZE
MOV R0, #xcell
init_xram:
MOVX @R0, A
INC A
INC R0
DJNZ R2, init_xram

; Initialize error counters
CLR A
MOV rec_cnt, A
MOV unrec_cnt, A
MOV math_cnt, A

; Check XRAM

test:
MOV R2, #XRAMSIZE
MOV R3, #00D
MOV R0, #xcell
check_xram:
MOVX A, @R0
XRL A, R3
INC R0
INC R3
DJNZ R2, check_xram

; Check Internal memory

check_imem:
MOV A, @R0
XRL A, R3
INC R0
INC R3
DJNZ R2, check_imem

; Indicate liveness
; SETB P34
; CLR P34

; Invoke math test
CALL math_test

; Print results
CALL print_rec
CALL print_unrec
CALL print_math

; Change line
MOV A, #0AH
CALL putchar
MOV A, #0DH
CALL putchar

; Indicate liveness
SETB P34
NOP
CLR P34
JMP test
```assembly
;===============================================================================
math_test:
RSEG segMath
USING 0

;test increment
CLR A
MOV DPTR,#array_a
MOV math_cnt,A
MOV R2,A
MOV R3,#090D

;test_inc:
CLR A
MOVC A,@A+DPTR
XRL A,R2
INC R2
JZ ok_inc
INC math_cnt

ok_inc:
INC DPTR
DJNZ R3,test_inc

;test 8bit addition
MOV R0,#00H
MOV R1,#0FFH
MOV R2,#0FFH

;test_add:
MOV A,R0
ADD A,R1
XRL A,#0FFH
INC R0
DEC R1
JZ ok_add
INC math_cnt

ok_add:
DJNZ R2,test_add

;test 8bit multiplication
CLR A
MOV B,A
MOV R0,A
MOV R3,#020D

;test_mul:
MUL AB
MOV R1,A
MOV R2,B
MOV A, R0
MOV DPTR,#a_comp
MOVC A,@A+DPTR
XRL A,R1
JZ ok_mul_A
INC math_cnt

ok_mul_A:
MOV A, R0
MOV DPTR,#b_comp
```
MOVC A,@A+DPTR
XRL A,R2
JZ ok_mul_B
INC math_cnt

ok_mul_B:
INC R0
MOV A,R0
MOV B,R0
DJNZ R3,test_mul

RET

;===============================================================
;
; print_rec()
;
;===============================================================
print_rec:
MOV R3, #04D
MOV DPTR,#rec_string

loop_rec:
CLR A
MOVC A,@A+DPTR
CALL putchar
INC DPTR
DJNZ R3, loop_rec

; print high order nibble
MOV A, rec_cnt
SWAP A
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print low order nibble
MOV A, rec_cnt
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

RET

;===============================================================
;
; print_unrec()
;
;===============================================================
print_unrec:
MOV R3, #08D
MOV DPTR,#unrec_string

loop_unrec:
CLR A
MOVC A,@A+DPTR
CALL putchar
INC DPTR
DJNZ R3, loop_unrec

; print high order nibble
MOV A, unrec_cnt
SWAP A
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar

; print low order nibble
MOV A, unrec_cnt
ANL A,#0FH
MOV DPTR,#hex
MOVC A,@A+DPTR
CALL putchar
RET

;=====================================================================
; print_math():
;=====================================================================
print_math:
    MOV R3, #07D
    MOV DPTR,#math_string
loop_math:
    CLR A
    MOVC A,@A+DPTR
    CALL putchar
    INC DPTR
    DJNZ R3, loop_math
    ; print high order nibble
    MOV A, math_cnt
    SWAP A
    ANL A,#0FH
    MOV DPTR,#hex
    MOVC A,@A+DPTR
    CALL putchar
    ; print low order nibble
    MOV A, math_cnt
    ANL A,#0FH
    MOV DPTR,#hex
    MOVC A,@A+DPTR
    CALL putchar
    RET

;=====================================================================
; print_start():
;=====================================================================
print_start:
    MOV R3, #07D
    MOV DPTR,#start_string
loop_start:
    CLR A
    MOVC A,@A+DPTR
    CALL putchar
    INC DPTR
    DJNZ R3, loop_start
    RET

;=====================================================================
; putchar():
;=====================================================================

putchar:
    MOV  SBUF, A
    JNB  TI, $  
    CLR  TI
    RET

END