NEPP Processor Efforts 2017

Steven M. Guertin
steven.m.guertin@jpl.nasa.gov
818-321-5337
Jet Propulsion Laboratory / California Institute of Technology

Acknowledgment:
This work was sponsored by:
The NASA Electronic Parts and Packaging Program (NEPP)
Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

## Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRL</td>
<td>Air Force Research Laboratory</td>
</tr>
<tr>
<td>AMD</td>
<td>Advanced Micro Devices</td>
</tr>
<tr>
<td>ASU</td>
<td>Arizona State University</td>
</tr>
<tr>
<td>CMOS</td>
<td>Complimentary Metal Oxide Semiconductor</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>DDR</td>
<td>Dual Data Rate</td>
</tr>
<tr>
<td>DIP</td>
<td>Dual Inline Package</td>
</tr>
<tr>
<td>DUT</td>
<td>Device Under Test</td>
</tr>
<tr>
<td>FET</td>
<td>Field Effect Transistor</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field Programmable Gate Array</td>
</tr>
<tr>
<td>HPSC</td>
<td>High Performance Space Computer</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics Processing Unit</td>
</tr>
<tr>
<td>GSFC</td>
<td>Goddard Space Flight Center</td>
</tr>
<tr>
<td>ILP</td>
<td>Instruction-Level Parallelism</td>
</tr>
<tr>
<td>JPL</td>
<td>Jet Propulsion Laboratory</td>
</tr>
<tr>
<td>LANL</td>
<td>Los Alamos National Laboratory</td>
</tr>
<tr>
<td>LPP</td>
<td>Low Power Plus</td>
</tr>
<tr>
<td>MPSOC</td>
<td>Multiprocessor System on Chip</td>
</tr>
<tr>
<td>NASA</td>
<td>National Aeronautics and Space Adminstration</td>
</tr>
<tr>
<td>NEPP</td>
<td>NASA Electronic Parts and Packaging Program</td>
</tr>
<tr>
<td>NSWC</td>
<td>Naval Surface Warfare Center</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>POP</td>
<td>Package on Package</td>
</tr>
<tr>
<td>SBU</td>
<td>Single Bit Upset</td>
</tr>
<tr>
<td>SEE</td>
<td>Single Event Effects</td>
</tr>
<tr>
<td>SEL</td>
<td>Single Event Latchup</td>
</tr>
<tr>
<td>SOC</td>
<td>System on a Chip</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>TBD</td>
<td>To Be Determined</td>
</tr>
<tr>
<td>TID</td>
<td>Total Ionizing Dose</td>
</tr>
</tbody>
</table>

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Outline

• Intro/Processor Overview
• Processor & Microcontroller Tasks Review
• Partnering & Opportunities
• Trends and Test Methods
• Testing & Results – Snapdragon
• Testing & Results – P2020
• Results - Intel
• Future Directions…
• Summary
NEPP – Processors, Systems on a Chip (SOC), and Field Programmable Gate Arrays (FPGAs)

Best Practices and Guidelines

State of the Art COTS Processors
- Sub 32nm CMOS, FinFETs, etc
- Samsung, Intel, AMD

“Space” FPGAs
- Microsemi RTG4
- Xilinx MPSOC+
- ESA Brave (future)
- “Trusted” FPGA (future)

COTS FPGAs
- Xilinx Kintex+
- Mitigation evaluation
- TBD: Microsemi PolarFire

Graphics Processor Units (GPUs)
- Intel, AMD, Nvidia
- Enabling data processing

Radiation Hardened Processor Evaluation
- BAE
- Vorago (microcontrollers)

Partnersing
- Processors: Navy Crane, BAE/NRO-
- FPGAs: AF SMC, SNL, LANL, BYU,…
- Microsemi, Xilinx, Synopsis
- Cubic Aerospace

Potential future task areas:
artificial intelligence (AI) hardware, Intel Stratix 10

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
What are we trying to do?

- **Primary Purpose**
  - Utilize processors as “bleeding edge” CMOS evaluations with goals of determining failure sensitivities and modes as well as to provide guidance for future flight project testing
  - Evaluate emerging architectures for radiation tolerance such as multi-core, etc…
  - Partner with NASA/Mil-Aero developments of processors to enhance qualification processes and provide independent assessments
  - Provide selective radiation evaluation of small mission (aka CubeSat) electronics
What are we trying to do?

- **Secondary Purposes**
  - Cross section vs. linear energy transfer (LET) information on device structures & Architectures
    - Test and qualification methods for processors
    - Build knowledge base of processor architectures
  - Provide total ionizing dose (TID) test data and parts program information
  - Gather information on various fabrication facilities
    - CMOS Nodes
    - On-shore vs. off-shore fabrication
  - Resilience of commercial processors
    - Keep abreast of developing technology trends and how to perform appropriate radiation testing
  - Device structure sensitivity to global device sensitivity

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Processors – Traditional and SOC

• Microprocessors
  – Traditional central processing units – CPUs
  – Modern desktop processors
  – Phone/Mobile processors
  – Kinda hard to find plain microprocessors these days

• System on a Chip (SOC)
  – Almost all modern processors incorporate few to many heterogenous functions
  – Not traditional SOC, but heading that way, and the definition of SOC is a disaster
    • “Smartphones and tablet don’t just use “processors”, they use what’s called a System-on-a-Chip (or SoC).” - http://www.ubergizmo.com/what-is/system-on-a-chip/
    • The multi-function chip in your phone is hijacking “SOC”

• Hybrid Stuff…
  – FPGAs (field programmable gate arrays) with built-in processor systems
Processors – GPUs and Microcontrollers

- **Graphics Processing Units (GPUs)** are high performance parallel processing machines
  - Some GPUs are available as CPUs...

- **Microcontrollers**
  - We will cover CubeSat and 32-bit microcontrollers here

- Where appropriate we are collaborating
  - Target devices
  - Architectures
  - Technology goals
  - Crossover items

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Task Partnering

• Engaging in collaborative efforts:
  – Adam Duncan & NSWC Crane folks
  – Carl Szabo, Ed Wyrwas, Ted Wilcox, and Ken LaBel, GSFC
  – Jeff George, Aerospace Corporation
  – Larry Clark, ASU
  – Heather Quinn, LANL, and other members of the Microprocessor and FPGA Mitigation Working Group
  – Sergeh Vartanian and Greg Allen, JPL
  – Vorago Technologies – collaborating on hardware/plans
  – Paolo Rech – GPU/Applications, UFRGS
  – Intel – informally
  – BAE Systems – team forming
  – Qualcomm Cybersecurity Solutions – team forming

• Looking for additional collaborators
  – Tester side – are you testing processors?
  – Manufacturer side – knowledge or hardware support
  – Application side – specific applications…

9To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Advanced Processors
- collaborative with NSWC Crane, others

High Performance Space Processor (HPSC)
- Joint NASA-AFRL Program for RH multi-core processor

14nm CMOS Processors (w/Navy Crane)
- Intel 14nm FinFET commercial
  - 5th and 6th generation
- Samsung 14nm LPP Snapdragon 820
- AMD Ryzen 14nm Global Foundries

10nm CMOS Processors
- Samsung 10nm Snapdragon 835
- Intel 10nm

Freescale Processors
- P2020 Communication Processor (w/Air Force)
- P5040 Network Processor

RH Processor
- BAE Systems RAD5510/5545
  - Leverages P5040 architecture

FY15 FY16 FY17 FY18

TBD – (track status)

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Microcontrollers
- collaborative with Vorago, others

CubeSat Microcontrollers
• MSP 430 w/Flash (1- and 5-)
• PIC 24 & 33
• Atmel AT91SAM9G20
• MSP 430 w/FRAM

32 – Bit Microcontrollers

Automotive-Grade Microcontrollers
• NXP MPC5606B Power Architecture MCU

Radiation-Hardened Microcontrollers
• Vorago VA10820 ARM Cortex-M0 MCU
• Vorago M4

Radiation Testing

FY15 FY16 FY17 FY18

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Deliverables

- SOC Test Guideline – in final review at JPL (JPL handling release editing)
  - Gathering materials for updated test guideline
- Radiation test data/reports on:
  - P2020 – SEE (single event effects) – Heavy Ion & Proton
  - Intel 14nm – including power device SEE failure related to firmware
  - AMD Ryzen 16nm (details TBD)
  - Samsung 14nm LPP/Snapdragon 820 SEE – Heavy Ion & Proton
  - RAD55xx radiation data (details TBD)
  - Samsung 10nm/Snapdragon 835 SEE (details TBD)
  - Vorago VA10820
  - Processor trends document
Justifications - Processors

- Intel 14nm [broadwell & skylake] (10nm when available)
  - Board computers going into CubeSats (installed as assemblies), higher risk designs. Very low power (without screen)
  - Collaborative work identified TID and SEE anomalies → skylake
  - Group of people looking at proton facilities: compare and contrast.
  - Some use of higher power – but to get architecture straightened out. (board fail due to bios)

- AMD uP
  - Similar to Intel, comparison case – to skylake 6600; uncertain how low-power stuff goes.
  - Obtain data on GlobalFoundaries performance (16nm)

- Freescale
  - Architecture used in RAD750, Space Micro P400k-L, RAD55xx series

- Snapdragon
  - 14nm Samsung LPP data, and first look at 10nm Samsung
  - SOMs being used in board-level computers (installed as assemblies); and Smartphones in space
Justifications - Microcontrollers

- Microcontrollers
  - Earlier CubeSat devices – per devices used in CubeSat kits, and based on application suggestions
  - Advanced 32-bit microcontrollers are feature-packed:
    - 64kB (and up!) SRAM
    - 512kB (and up!) integrated Flash memory
    - 100 MHz+ operation
    - Large number of peripherals (interrupts, ADC/DAC, counters, clocks, CAN/SPI/I2C/Ethernet/USB controllers)
    - Multiple cores!
  - Targeted for specific niche markets
    - Easier OTS access to interesting test parts, like:
      - Automotive grade
        » Overlaps with mil/aero interest in temperature & reliability
      - Rad-hard designs available
# CubeSat Microcontroller Review

<table>
<thead>
<tr>
<th>Device</th>
<th>Manufacturer</th>
<th>CubeSat Kit</th>
<th>NASA Sats</th>
<th>Others</th>
<th>2015 Tests</th>
<th>TID conditions</th>
<th>2016 Tests</th>
<th>TID conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSP430F1611</td>
<td>TI</td>
<td>X</td>
<td></td>
<td></td>
<td>SEE/SEL/TID</td>
<td>Unbiased/biased, Dynamic, reprogramming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSP430F1612</td>
<td>TI</td>
<td>X</td>
<td></td>
<td></td>
<td>SEE/SEL/TID</td>
<td>Unbiased/biased, Dynamic, reprogramming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSP430F1618</td>
<td>TI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSP430F22619</td>
<td>TI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSP430FR5739</td>
<td>TI</td>
<td>X</td>
<td></td>
<td></td>
<td>SEE/SEL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSP430FR5739 non-EPI</td>
<td>TI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>Unbiased/biased, Dynamic, reprogramming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C8051F120</td>
<td>Silicon Labs</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIC24FJ256GA110</td>
<td>Microchip</td>
<td>X</td>
<td></td>
<td></td>
<td>SEE/SEL/TID</td>
<td>Unbiased/biased, Dynamic, reprogramming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dsPIC33FJ256GP710</td>
<td>Microchip</td>
<td>X</td>
<td></td>
<td></td>
<td>SEE/SEL/TID</td>
<td>Unbiased/biased, Dynamic, reprogramming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT91SAM9G20</td>
<td>Atmel</td>
<td>X</td>
<td>X</td>
<td></td>
<td>SEE/SEL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT91SAM7</td>
<td>Atmel</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATMEGA1281</td>
<td>Atmel</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATMEGA164P</td>
<td>Atmel</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATMEGA32U/8</td>
<td>Atmel</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATMEGA16U2</td>
<td>Atmel</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cortex-M3 MCU</td>
<td>ARM/General</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other ARM9</td>
<td>ARM/General</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PX32A</td>
<td>Parallax</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ColibriPXA270</td>
<td>Intel/Marvel</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sitara AM3505</td>
<td>TI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sitara AM3703</td>
<td>TI</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Commercial Trends

- Clarify what we’re talking about
  - Shrinking features
  - Increasing complexity

- Recently, microprocessors are getting more complex, not faster, not higher power

- Heterogeneous with many structures

Fundamental Approaches

• Ideal:
  – Obtain SEE data on individual structures
    • By direct observation of N structures
    • In the same operating conditions as normal use
    • Utilizing debuggers or specialized test code
  – Divide out (normalize) any observations to the number of targets available
  – Maximize targets being tested

• Non-Ideal:
  – Run an operating system (OS) with a specified workload
    • Count events – beware normalization
    • Count crashes...
  – Run test software under an OS
    • Count events & crashes
  – Biggest issue is normalization

• Flight Like: (???)
  – This is something of a myth, because test conditions are not flight conditions... and you can’t get flight code
  – Accelerated tests are not inherently “flight-like” (e.g. latent errors)
Fundamental Approaches

• Ideal:
  – Obtain SEE data on individual structures
    • By direct observation of N structures
    • In the same operating conditions as normal use
    • Utilizing debuggers or specialized test code
  – Divide out (normalize) any observations to the number of targets available

• Non-Ideal:
  – Run an operating system (OS) with a specified workload
    • Count events – beware normalization
    • Count crashes…
  – Run test software under an OS
    • Count events & crashes
    • Biggest issue is normalization

• Flight Like:(???)
  – This is something of a myth, because test conditions are not flight conditions… and you can’t get flight code
  – Accelerated tests are not inherently “flight-like” (e.g. latent errors)

Also looking into ways to resolve test issues, lack of visibility, application of data, and limited documentation
- New approaches for low level data
- Hybrid methods to get “flight-like” information

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Test Challenges

- New hardware issues
  - Package on Package
  - Dedicated power chips – complex
  - Direct low-level hardware access may require custom test fixture hardware and software (SW) – simpler DUTs only
- TID coming back
  - Heterogenous structures – may include analog (thermal?)
- SEL (single event latchup) risks
  - Mixed IO voltages due to “other functions”
- SEE test problems due to
  - Lack of documentation
  - Interference from other device structures (i.e. the main processors may interfere with testing the memory controller)
  - Each SEE almost takes a root cause investigation
Eval Board Issues...

- Example is Snapdragon 800
- Package on package (POP) is a significant problem
- Semi-custom DDR4 device mounted to device under test (DUT)
- No datasheets
Snapdragon 820

- Key Features:
  - Quad-core Kyro CPU
    - Actually has ~9 distinct processors
    - big.LITTLE – 2 cores are faster, bigger, other two are smaller and slower
  - Low power DDR4
  - Universal Flash Storage
  - Hexagon 680 DSP with isolated sensor power
  - Camera controller
  - Hardware multimedia encode & decode
  - Adreno GPU

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Test Vehicle: Intrinsyc Open-Q 820

- Evaluation board for Snapdragon 820
- Hardware debug intentionally limited
- Uses system-on-module/carrier configuration
- 3GB DDR4 with POP setup
Snapdragon 820:
Tests Performed

- Heavy Ions @ TAMU
  - Ion selection range limited...
  - Android & custom code
- Protons @ MGH
  - ~1x10^{10} /cm² with 100, and 200 MeV, 5x10^{9}/cm² with 50 MeV
  - Android & custom code
- Neutrons at LANSCE
  - ~1x10^{11} /cm² with sea level neutron spectrum

<table>
<thead>
<tr>
<th>Beam</th>
<th>LET (MeV·cm²/mg)</th>
<th>Exposure (cm²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>1</td>
<td>1.2E+07</td>
</tr>
<tr>
<td>Ne</td>
<td>6</td>
<td>1.2E+04</td>
</tr>
<tr>
<td>Ar</td>
<td>15</td>
<td>4.1E+04</td>
</tr>
</tbody>
</table>
Stuck Bit
Annealing

Beam Exposure

Beam Testing

Board w/ DUT

Allow to “cool” (in test room)

Radiation Area

Heat Gun

Anneal in user area

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Results: Crashes

- Heavy Ions:

- Protons/Neutrons:
  - Proton curve →
  - Neutrons
    \[ \sigma \sim 1 \times 10^{-8}/\text{cm}^2 \]
Results: SBUs & Stuck Bits

- Limiting $\sigma$ for SBUs in Snapdragon:

- Stuck Bits during Boot & Anneal:

- Note also crunching data on bit errors in the 1GB memory region

<table>
<thead>
<tr>
<th>Board</th>
<th>Incr. Neutron Exposure (/cm²)</th>
<th>Annealing Duration (mins)</th>
<th>Annealing Temp (°C)</th>
<th>Stuck Bits - boot Before</th>
<th>Stuck Bits - boot After</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2.52E+10</td>
<td>30</td>
<td>175</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2.52E+10</td>
<td>15</td>
<td>175</td>
<td>14</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>0.00E+00</td>
<td>90</td>
<td>175</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3.12E+09</td>
<td>120</td>
<td>175</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2.81E+10</td>
<td>220</td>
<td>175</td>
<td>8</td>
<td>0</td>
</tr>
</tbody>
</table>

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
P2020 - Test Setup: Hardware

- P2020RDB-PCA unit used for testing
- Two serial connections used – 1 for each CPU core
- Utilized U-Boot software to start up the DUTs
- Used power system on board, with power supply from unit - Earlier testing showed no risk of SEL
- Also used BDI3000 debug cable plugged into debug port to allow direct communication
  - Supported on-board flash programming
  - Allowed direct readout of registers
Testing/Details

• Proton and Heavy Ion Testing
  – TRIUMF 11/2015
  – MGH 12/2015
  – LBL 12/2015 and 5/2016
  – TAMU 5/2016

• 5 boards/DUTs tested with protons

• 5 boards/DUTs tested with heavy ions

<table>
<thead>
<tr>
<th>Board</th>
<th>Energy (MeV)</th>
<th>Proton Exposure</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>100</td>
<td>2.00E+10</td>
</tr>
<tr>
<td>44</td>
<td>100</td>
<td>3.30E+10</td>
</tr>
<tr>
<td>28</td>
<td>100</td>
<td>1.00E+10</td>
</tr>
<tr>
<td>14</td>
<td>100</td>
<td>2.10E+10</td>
</tr>
<tr>
<td>32</td>
<td>100</td>
<td>1.80E+10</td>
</tr>
<tr>
<td>32</td>
<td>200</td>
<td>9.10E+09</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th># Boards</th>
<th>Ion</th>
<th>LET (MeV-cm²/mg)</th>
<th>Fluence (#/cm²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>N</td>
<td>1.2</td>
<td>3.84E+08</td>
</tr>
<tr>
<td>2</td>
<td>Ne</td>
<td>2.4</td>
<td>8.80E+07</td>
</tr>
<tr>
<td>3</td>
<td>Ne</td>
<td>2.8</td>
<td>4.50E+07</td>
</tr>
<tr>
<td>5</td>
<td>Ar</td>
<td>7.3</td>
<td>1.14E+08</td>
</tr>
<tr>
<td>1</td>
<td>Ar</td>
<td>8.6</td>
<td>5.89E+06</td>
</tr>
<tr>
<td>1</td>
<td>V</td>
<td>10.9</td>
<td>2.36E+07</td>
</tr>
<tr>
<td>3</td>
<td>Kr</td>
<td>25</td>
<td>6.43E+07</td>
</tr>
<tr>
<td>3</td>
<td>Kr</td>
<td>28.8</td>
<td>5.31E+05</td>
</tr>
<tr>
<td>1</td>
<td>Xe</td>
<td>49.3</td>
<td>9.98E+05</td>
</tr>
<tr>
<td>2</td>
<td>Xe</td>
<td>53.1</td>
<td>5.18E+05</td>
</tr>
</tbody>
</table>
Test Software

1) Register SBU – SBU in a processor register – also w/ external debugger
2) Register MBU – a register completely changes – also w/ debugger
3) L1 invalidates – an L1 cache line (with parity protection disabled) is lost
4) L1 SBU – this is a reported parity error when parity is enabled
5) L1 parity invalidations – parity-protected L1 cache loses valid line of data
6) L2 SBU – a SBU observed in L2 data (L2 tested w/ EDAC disabled)
7) External memory errors – not reported here
8) Watchdog – monitor the watchdog system for correct operation
9) Ethernet packet error – test for DUT packets received or transmitted
10) Flash Memory – errors reading or writing flash memory w/ external debug tools
Results: Heavy Ion Cache Errors

L1 Errors will cause app/OS crash unless in “write-through”

Bit errors are per-bit.

L1 bit errors are about 10x worse than block errors - $5 \times 10^5$ bits
- L2 is 100x worse

L2 block errors not tested but bit errors are EDAC-protected

- Register sensitivity (per bit) is similar to L1 & L2 cache bit sensitivity…
Results: Proton Cache Errors

Block errors also occurred with proton exposures.

Shows consistency across board-to-board results.

These errors would be silent even with parity protection.

Block errors with 100 MeV protons across 5 DUTs and two test facilities.
Results: Crashes & Strange Events

• Strange Events…
  – Bit error in test control register
  – Latent error caused readout problem after run was over
  – Bit error in test compare register caused runaway error reports
  – CPU showed delay and eventually recovered (though possibly slower than before)

-Proton crash sensitivity – many parts/conditions

– Consistent with older tests
– Highlights that when using the memory system, crash rate increases significantly
Results: Ethernet Testing

- No corrupt packets observed
  - 768-byte payload
  - 44 Mbps rate

- While testing for packet corruption, sensitivity limited by device crashes
  - unrelated to Enet

- Packet loss about the same in/out of beam ~ 0.01-0.1%

Ethernet Sensitivity - Crashes

Cannot test packets here due to crash interference
Results: Flash Memory

- Debugger was able to read and write during exposure
- Tested with the system suspended, just to check how the Flash interaction circuits responded
- Debugger connects through the processor flash memory interface (not directly to the Flash)
- Did not see evidence of any errors written or read from the Flash memory in any testing
- Similar results for watchdog

---

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Results & Data: Intel

- Test efforts reported at NSREC and other locations 2015 & 2016
  - NASA working with Navy Crane where appropriate
  - (Crane results on right)

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017

C. Szabo, 2015, IEEE
A. Duncan, 2016, IEEE
Future (/ Parallel) Efforts

• The future is now! (At least for commercial parts)
• We are looking at forward “SOC” trends
  – Integration of functions continues
  – Multiple, heterogenous processors
  – Dedicated power and other peripheral devices
  – Source/fabrication issues – fabless processors/SOCs

• Specify desired interfaces for processor data collection, to improve manufacturer interaction
• Clarify test goals in the environment of limited device information

• Other processors, such as those embedded in the Xilinx MPSOC (multiprocessor SOC), are handled under the NEPP FPGA efforts
Summary

- **Processor effort goal:**
  - Provide radiation performance information for relevant device families and technology nodes
  - Primary Goal: Utilize processors as “bleeding edge” CMOS evaluations with goals of determining failure sensitivities and modes as well as to provide guidance for future flight project testing

- **Looking at many devices:**
  - Intel 14nm, AMD 16nm, Qualcomm (Samsung) 14nm and 10 nm, BAE RAD55xx, Freescale P2020
  - Many recent tests, with more coming soon

- **Looking for additional collaborators**
  - Tester side – are you testing processors?
  - Manufacturer side – knowledge or hardware support
  - Application side – user input on where and how new can be used…
End
Microcontrollers and Microprocessors in Space

- Performance
  - Latest flattening due to focus on efficiency...

- Until about 2000, space processors were “close” to commercial devices.

- BAE’s RAD55XX series will bump up a bit.

Deployed devices in space missions

Information adapted from www.cpushack.com
Challenges - Microcontroller

- Many of the same capabilities (and thus issues) as CPU & GPU testing
  - Complex/opaque error signatures with integrated peripherals and integrated analog/power blocks
  - Non-ideal packaging for radiation test (high density ball grid array, flip chip, etc) – Not as much 48-DIP (dual inline package) anymore!
  - Potentially more direct low-level hardware access than CPU, but may require more custom test fixture hardware and software (SW) design work
- Need to correlate test results to real-world (flight) apps
  - High prevalence of “SEFI”-type (single event functional interrupt) events leads to a strong application-specific test result.
  - How will this perform with flight SW running inside OS?
Heavy Ion Setup
- Device Stack Estimate

- Using estimated thicknesses to get a range of estimated LETs

- Heavy ion testing (thus far) is general info, so it was most important to show chance of reaching sensitive region

<table>
<thead>
<tr>
<th>Item</th>
<th>Material</th>
<th>Range of Thicknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Low</td>
</tr>
<tr>
<td>Beam Port</td>
<td>Aramica</td>
<td>25.4 μm</td>
</tr>
<tr>
<td>Air</td>
<td>Air</td>
<td>3 cm</td>
</tr>
<tr>
<td>DDR3 Top Plastic</td>
<td>Plastic</td>
<td>100 μm (1.18 g/cm³)</td>
</tr>
<tr>
<td>DDR3 Die</td>
<td>Silicon</td>
<td>250 μm</td>
</tr>
<tr>
<td>DDR3 Metalization</td>
<td>Aluminum</td>
<td>40 μm</td>
</tr>
<tr>
<td>Air Gap</td>
<td>Air</td>
<td>100 μm (1.18 g/cm³)</td>
</tr>
<tr>
<td>Snapdragon Top Plastic</td>
<td>Plastic</td>
<td>100 μm</td>
</tr>
<tr>
<td>Snapdragon Die</td>
<td>Silicon</td>
<td>500 μm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LET (MeV·cm²/mg)</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Range (μm Si)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-40 MeV</td>
<td>0.74</td>
<td>1.2</td>
<td>1380 Out of Range</td>
</tr>
<tr>
<td>Ne-40 MeV</td>
<td>1.8</td>
<td>9</td>
<td>701 Out of Range</td>
</tr>
<tr>
<td>Ar-40 MeV</td>
<td>10</td>
<td>20</td>
<td>123 Out of Range</td>
</tr>
</tbody>
</table>

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Observation: Crashes

• Every operating condition had crashes
  – Mostly these involved the test DUT no longer communicating
  – On later tests, we were able to use Android’s exception handlers to get some indication what was going on

• Required restarting the test after each crash
  – We developed an automated system to do this at LANSCE
  – System conditions/handling was complex for the 8 states and possible errors coming from each
  – At LANSCE this all had to be done while being irradiated
Observation: Stuck Bits

• We targeted the DDR4 device for data, because it had to be exposed regardless
  – This is frustrating because it really is not helpful to have two sources of errors
  – But the DDR4 device provided stuck bits as well as SBUs
• Proton and neutron testing \(\rightarrow\) about as many stuck bits as SBUs
  – Stuck bits caused the DUTs to have trouble booting
  – Usually a reasonable chance to get a handful of detectable stuck bits before a DUT was unable to boot
  – Android appears to have a retry option on some memory allocation to allow it to avoid bad memory regions
Future Work
Heavy Ion Testing

• Improved test system will enable testing with heavy ions
  – Registers
  – Crashes (with capture of exceptions)
  – DDR4 Errors

• Additional data on caches
  – We have modified cache test code, but so far the L1 caches have not shown bit errors
    • May have ECC or parity masking the errors

• Test code on all cores?

To be presented by Steven M. Guertin at NEPP Electronics Technology Workshop, June 26, 2017
Collaboration and Development

• We prefer to test low-level structures and develop system-level rates
  – Lack detail on applications to extrapolate to system-level (~application vulnerability factor [AVF])
  – Visibility on low-level structures is reducing
  – Newer devices are much more complex

• Challenges:
  – It is currently very difficult to get the right information for low-level tests (even with manufacturer support).
  – We cannot get flight-like software.

• Alternate approaches:
  – Estimated device usage for application – update/verify
  – White-paper indicating what information space users need to get the right data – without crossing NDA issues
    • I.e. need to know about hidden memories, but not why or how
  – Develop additional device-specific/architecture knowledge to enable improved low-level data
    • Use this along with device models to predict app sensitivity
Results: Register SBUs

- Three new test boards vs. old data
- Low counts – large errors
- 0033 data sometimes outlined for clarity
Results: Watchdog

• Monitored for correct change of states in the watchdog system
  – Has multiple states it can get into – with different types of exceptions that are called

• Tested for various LETs

• No indication, in all testing, of any error in watchdog system except:
  – Some indication of register errors changing timeframes for watchdog behavior
  – But the event rate was consistent with register upsets, not indicative of true watchdog sensitivity

• Highlights same problem as Ethernet – data limited by more common event types