



Radiation Effects on ARM Devices

Steven M. Guertin

Jet Propulsion Laboratory / California Institute of Technology
Pasadena, CA

This work was performed at the Jet Propulsion Laboratory, California Institute of Technology,
Under contract with the National Aeronautics and Space Administration (NASA)
This work was funded by the NASA Electronic Parts and Packaging Program (NEPP)

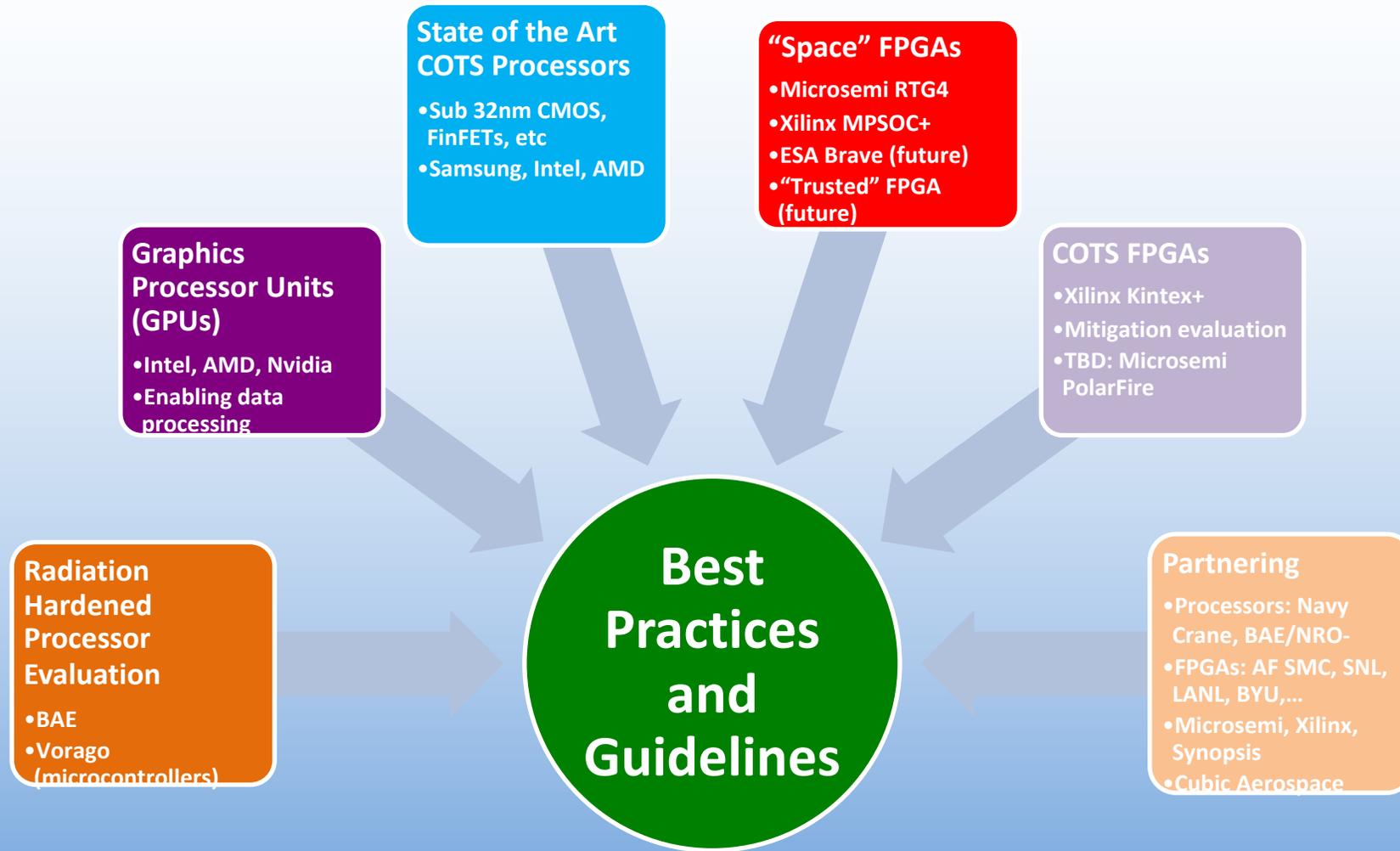
Copyright 2019 California Institute of Technology. Government Sponsorship is acknowledged.



Outline

- SOCs/Program Perspective
- Goals
- Background on ARM and Target Devices
- Radiation Effects Evaluation Methods
- Results – Snapdragon 835
- Looking to the Future?
- Conclusion

NEPP – Processors, Systems on a Chip (SOC), and Field Programmable Gate Arrays (FPGAs)



Potential future task areas:

artificial intelligence (AI) hardware, Intel Stratix 10

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, Andrew Daniel, 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...



Focus Categories

- Architecture – to support evaluation and use of processor architectures throughout NASA, including processor types and FPGA/Soft processors
- Implementations – to support evaluation and use of primary form-factors
- Fabrication Facilities/Technology – to obtain information on fabrication facilities and related technology (e.g. Samsung 7nm, 3D, etc.)
- Application/Use Case – to support ways of using devices for different NASA needs
- Develop data on specific devices/Methods for evaluation – support actual flight use, and understand that in many cases the project will have to evaluate their own part (but we can provide guidance)
- Manufacturers – to have an up-to-date tool set for understanding devices from each manufacturer.
- Collaborations – to engage manufacturers when they are available or can work with us, we want to harness this
- Test Method Development
- Guidelines and BOKs
- Recent work

Advanced Processors - Commercial



- collaborative with NSWC Crane, others

14nm CMOS Processors (w/Navy Crane)

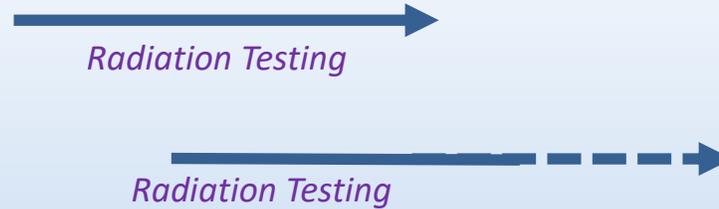
- Intel 14nm FinFET commercial
 - 5th and 6th generation
- Samsung 14nm LPP Snapdragon 820
- AMD Ryzen 14nm Global Foundries



Closing Out on processor side – see GPUs below.

10nm CMOS Processors

- Samsung 10nm Snapdragon 835
- Intel 10nm

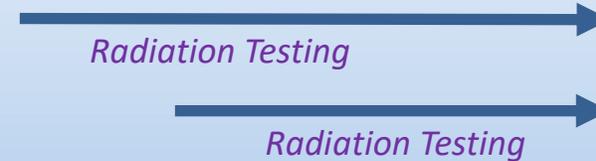


Radiation Testing

Radiation Testing

7nm CMOS Processes

- Samsung 7nm (Qualcomm)



Radiation Testing

Radiation Testing

GPUs

- nVidia 1050 (14 nm)
- Tegra TX1/TX2 (14 nm)
- Snapdragon 820 (14 nm)



Radiation Testing

Radiation Testing



FY17

FY18

FY19

FY20

Advanced Processors – Flight/RHBD

- collaborative with BAE Systems, HSPC, others



High Performance Space Processor (HPSC)

- Joint NASA-AFRL Program for RH multi-core processor



RH Processor

- BAE Systems RAD5510/5545
 - Leverages P5040 architecture

Radiation Testing (Collaborative with BAE Systems)



General

NASA-Specific

Freescale Processors

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



Radiation Testing



Radiation Testing



Closeout



FY17

FY18

FY19

FY20



CubeSat/SmallSat Microcontrollers

- No current plans

Automotive-Grade Microcontrollers

- No current plans

Radiation-Hardened Microcontrollers

- Vorago VA10820 ARM Cortex-M0 MCU
- Vorago M4

- Cobham UT32M0R500

Hard vs. Soft Core Processors

- Soft/Hard ARM in Xilinx FPGA



Test Review/Collaboration



Establishing Approach



Establishing Approach & Collaboration





Primary Goals

- We are working to get a handle on the architectural implications of SEE on the ARM platform
- ARM devices (even relatively newer high performance cores) are implemented in many different silicon environments
 - Cell phone processors at 7nm, microcontroller devices at 100s of nm,
 - Provides opportunities to directly compare process nodes, and to get data on the newest process nodes
- We would like to develop an approach that intelligently identifies what issues are inherent to the architecture, and what issues are primarily due to the fabrication
- Further, ARM cores (not just processors) have a significant number of configuration parameters that can affect SEE
 - Includes FT settings
 - In some cases devices are very well documented
 - Indication that ARM is interested in knowing how well their devices work in order to improve reliability options



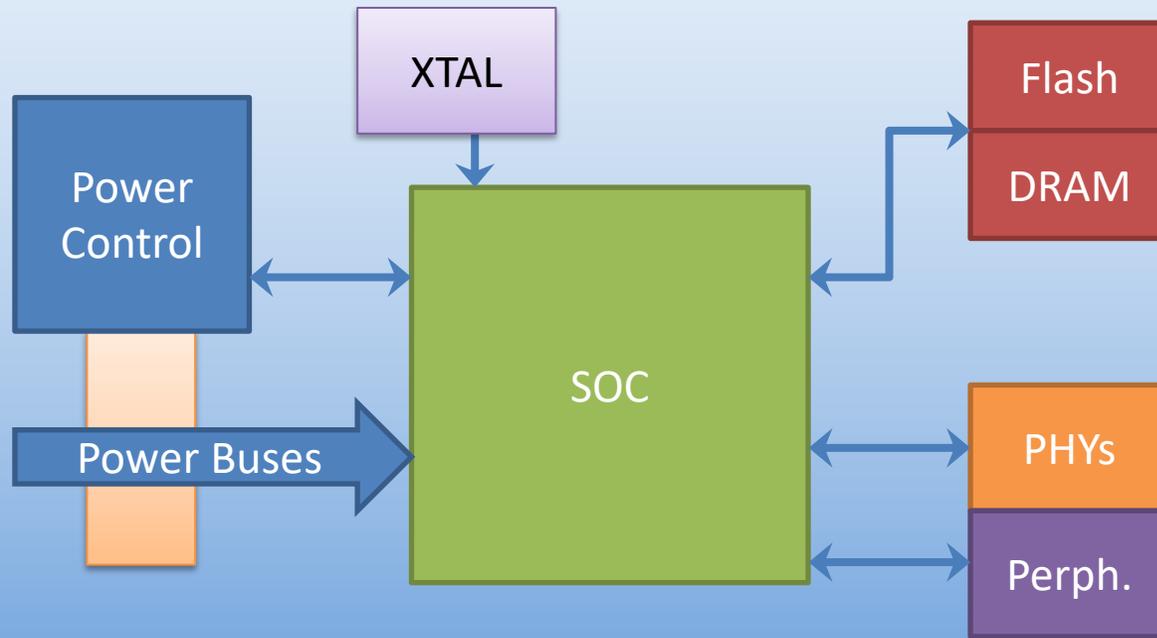
New Processors are Here - SOC's!

- Lots of new and upcoming missions are using new devices (SOC architectures)
 - Cell phone processors/drone processors on Cubesats and Mars Helicopter
 - (see https://rotorcraft.arc.nasa.gov/Publications/files/Balaram_AIAA2018_0023.pdf)
 - RAD5545 – quad core 64-bit PowerPC being considered for several programs
 - Interest in GPUs (nVidia & AMD) for machine learning, Intel devices, Qualcomm cellphone processors, etc.
- SEE testing of these devices, especially commercial, is running into problems

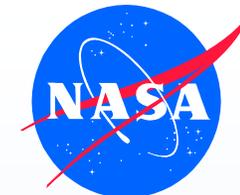


What is an SOC

- Anything that combines multiple functions to create a system
 - Does not need to be a single device that is a system by itself – SOC's may need support devices

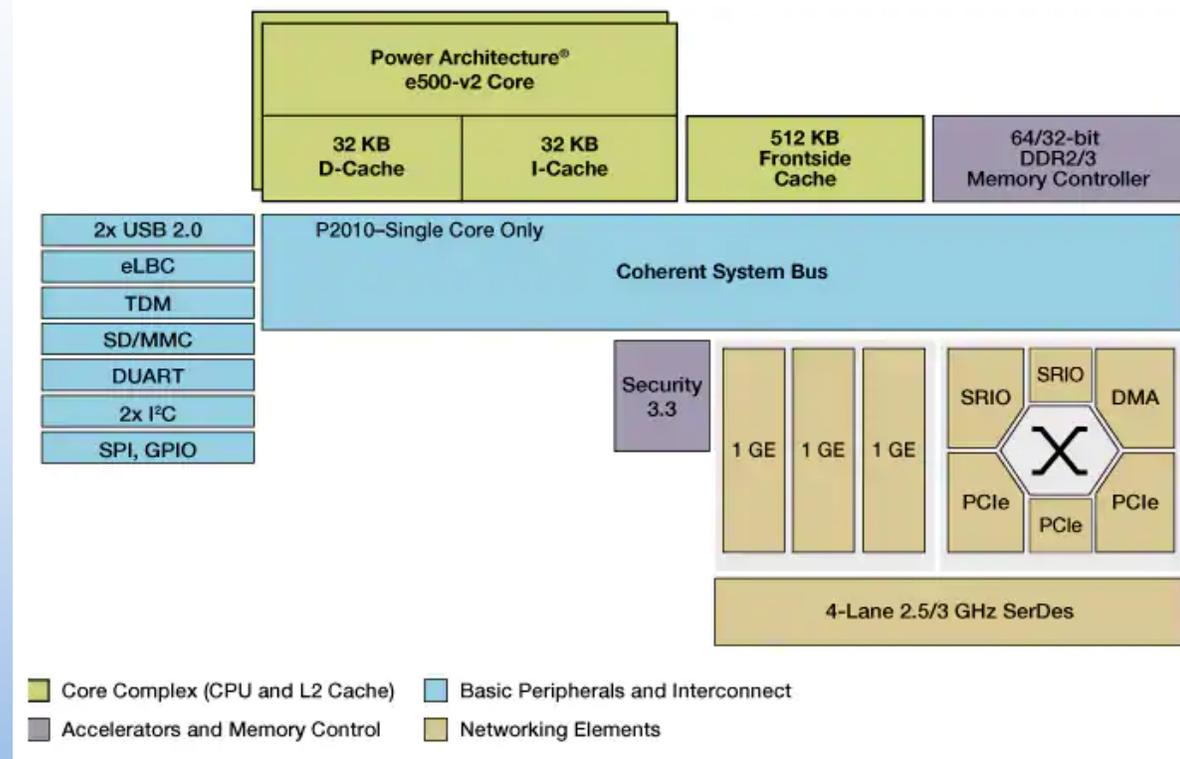


- most microcontrollers
- cell phone processors
- modern commercial processors
- BAE RAD5545
- GR712, UT699/700
- nVidia Tegra series
- drone processors
- FPGAs (different topic)



What do we know about our SOCs?

- 45 nm we had things like Freescale/NXP P2020



- We knew what peripherals we had on chip, capabilities, memory controllers, etc.



Devices of Interest

- Drone processors – like Snapdragon 801
- Cell phone processors –
like Snapdragon 820, 835, 845, 855 (7nm!)
- Microchip/Atmel SAM5D3
– (ARM A5 devices)
- Xilinx Ultrascale+ MPSoC
- TI TMS570 or similar – for fault tolerant ARM architecture



ARM-Based Microcontrollers w/Fault Tolerance



- Primarily focusing on similarities and differences from other ARM devices, and getting actual ARM-architecture fault-tolerance data
- Devices like: TI TMS570 “Hercules” , NXP S32S / MPC57, ST Micro SPC5, Cypress “Traveo” are being produced for automotive applications.
 - For Safety Applications
- Processing Cores (ARM-R series, PPC e200z0)
- Requirements for reliable operation -40 – +165 C
- Features included are delayed lockstep CPU, ECC on internal memories (end-end EDAC), advanced BIST (M-BIST / L-BIST / A-BIST), memory protection units, fault collection, & monitoring peripherals / supervisors (clocks, buses, voltages)
- Task seeks to evaluate protection features for use in natural space environment.



Other Architectures

- SOCs have their own ecosystems, though many buses are common
 - AXI, AMBA, etc.
 - Memory coherency, memory mapped peripherals common to all devices
 - Multiple microprocessors and support devices
- Freescale utilizes corenet/coherency bus, standard peripherals, etc.
- ARM has a set of buses, memory controllers, etc.
- Through other efforts under NEPP, there is data on Intel architecture
- Also of interest is RISC-V



Why ARM?

- Modern SOC's can't be considered stand-alone processors
 - Support devices must be considered, especially buses
- ARM or ARM support devices are used in many modern devices
 - Allows compare & contrast test approaches across similar-but-different devices
- In many cases, the architecture is well-documented
 - Especially in FPGAs how to properly use resources



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
 - Maximize targets being tested
- **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
- 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)

Snapdragon 835

- Samsung 10 nm
 - 8 Kryo 280 CPUs
 - Adreno 540 GPU
 - Hexagon DSP
- Using Intrinsic's 835 Mobile Hardware Development Kit – Android only, which is not desired...
- This board uses package-on-package (they essentially all do)
- No avenue to put Linux on the board.



What do we know about our SOCs?



Typical Datasheet

Snapdragon 835 processor

The diagram shows a central grid of components for the Snapdragon 835 processor. The components are arranged in a 4x2 grid. The top row contains the Snapdragon X16 LTE modem and the Adreno 540 Graphics Processing Unit (GPU). The second row contains Wi-Fi and the Qualcomm Spectra 180 Camera. The third row contains Qualcomm Aqstic Audio and the Kryo 280 CPU. The bottom row contains Qualcomm IZat Location and Qualcomm Haven Security. To the left of the grid are three callouts: Snapdragon X16 LTE, Qualcomm Hexagon DSP, and Qualcomm Kryo 280 CPU. To the right are two callouts: Qualcomm Adreno Visual Processing and Qualcomm Spectra Camera ISP. Each callout includes a yellow horizontal line and a brief description of the component's capabilities.

Snapdragon X16 LTE
World's first announced gigabit-class LTE modem

Qualcomm® Hexagon™ DSP
Tensorflow and Hallide Support

Qualcomm® Kryo™ 280 CPU
Our most power efficient architecture to date

Qualcomm® Adreno™ Visual Processing
25% Faster Graphics Rendering
60x More Display Colors*

Qualcomm Spectra™ Camera ISP
Smooth Zoom
Fast-Autofocus
True to Life Colors

Qualcomm Haven™ Security
First to support full biometric suite

- 10 nm we have things like Snapdragon 835
- Even the details of what chip capabilities are available on the eval boards require NDAs
 - This is not “by itself” a problem – but programs often have trouble with not sharing test data



Snapdragon 835 Results

- SEFI behavior with a cross section of about 1×10^{-4} cm² for LETs under 2 MeV-cm²/mg
- Independent of the test program used (video playback or graphics benchmark program)
 - All tests operated under the Android OS
- Up to an LET of about 2 MeV-cm²/mg the cross section for L2 bit errors was on the order of 1×10^{-11} cm²/bit.

Snapdragon 845 - TBD

- Samsung 10 nm
 - 8 Kyro 385 CPUs
 - Adreno 630 GPU
 - Hexagon DSP
- Using Intrinsyc's 845 Mobile Hardware Development Kit – Android only, which is not desired...
- No avenue to put Linux on the board.



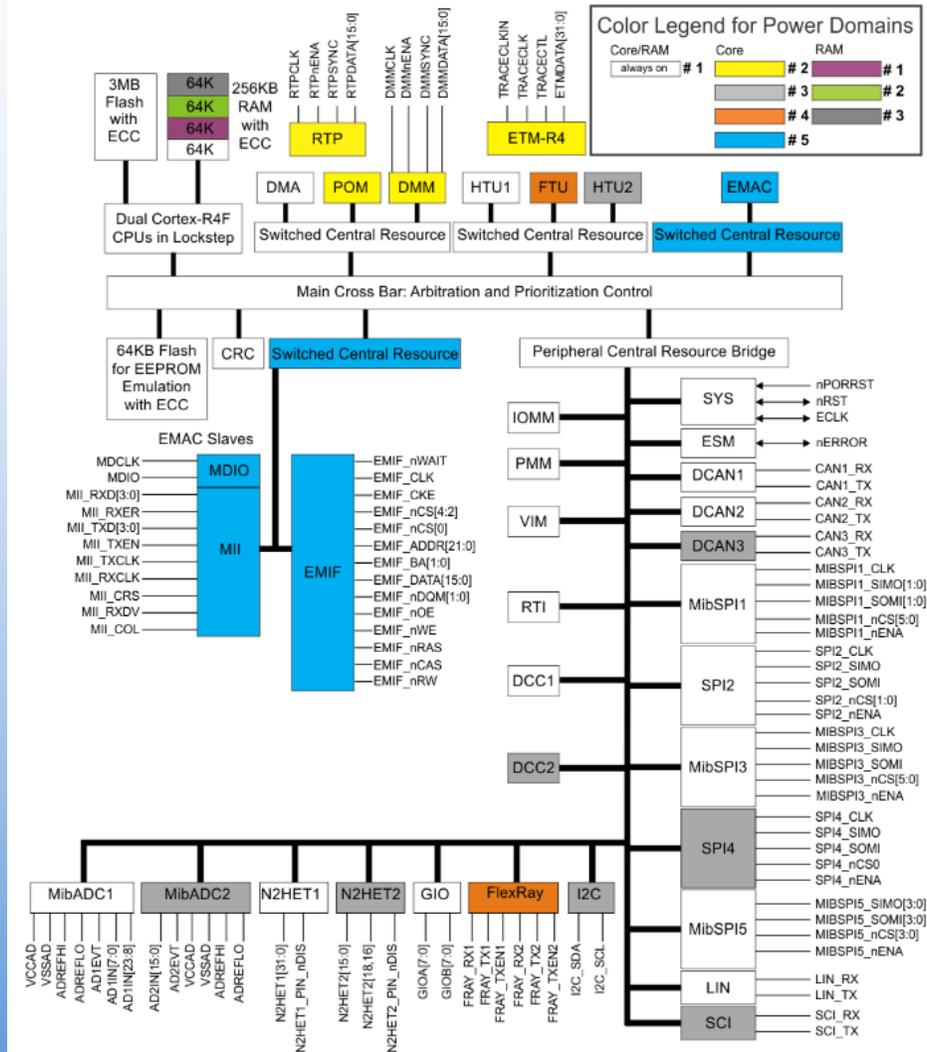


Snapdragon Test Development/Methods

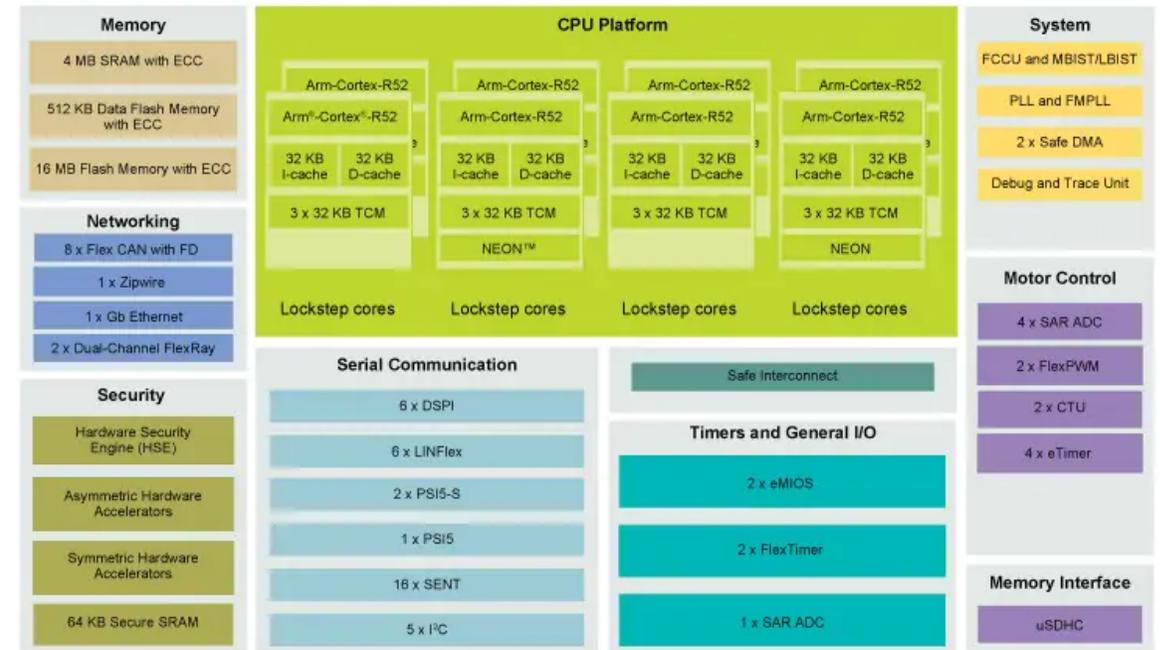
- Video playback test
- 3DMark benchmarking software
- Also porting some specific benchmark codes
 - Don't expect standard benchmarks to be of significant benefit
 - This is an SOC, not a processor – need SOC benchmarks...
 - SOC benchmarks are inherently not well defined...
 - But exploring altering the benchmarks for modern devices – e.g. Mean Work To Failure (MWTF) based on which resources are used – DSP, multiple cores, etc



Microcontroller Block Diagrams



NXP S32S Processor Block Diagram





Proposed Testing

- Crash cross section single core mode vs lockstep mode
- Bus fault protection during external memory SEFI
- Internal flash memory error modes
- Error / fault checking peripheral crash coverage
- Cortex R4 vs R5 vs R52 crash rate
- Internal Volatile memory error handling (SBU vs MBU)
- Parity protected peripheral memory improvement



Intel CE3100

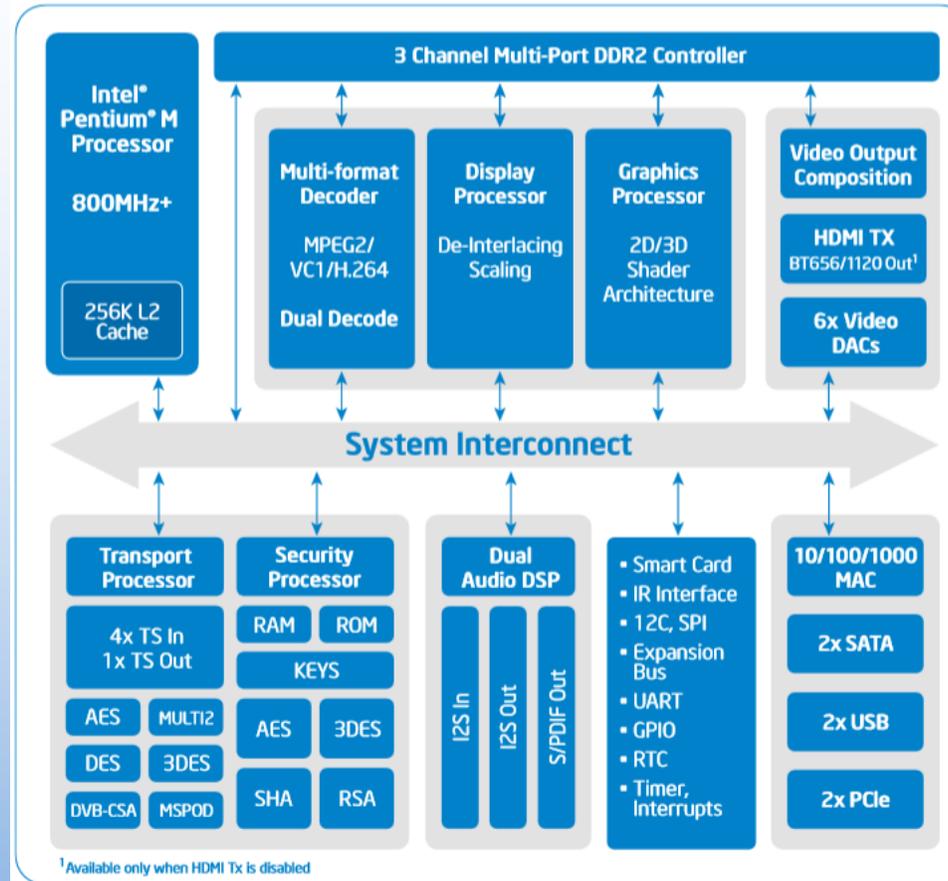


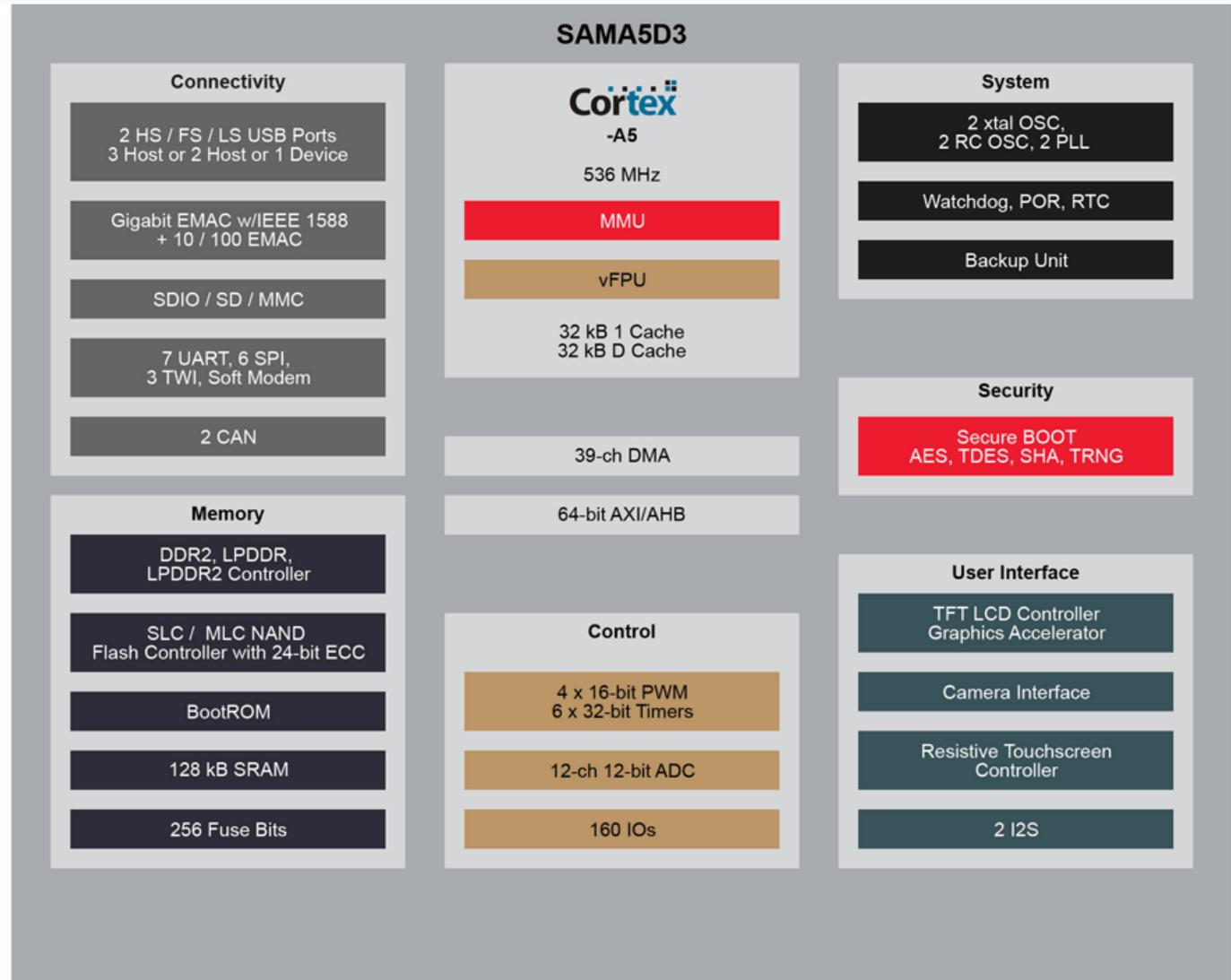
Figure 1 - Intel® Media Processor CE 3100 - Block Diagram

The Intel® Media Processor CE 3100 is a highly integrated system-on-a-chip that combines a high-performance Intel® architecture processor core with leading-edge video decoding and



SAMA5D3

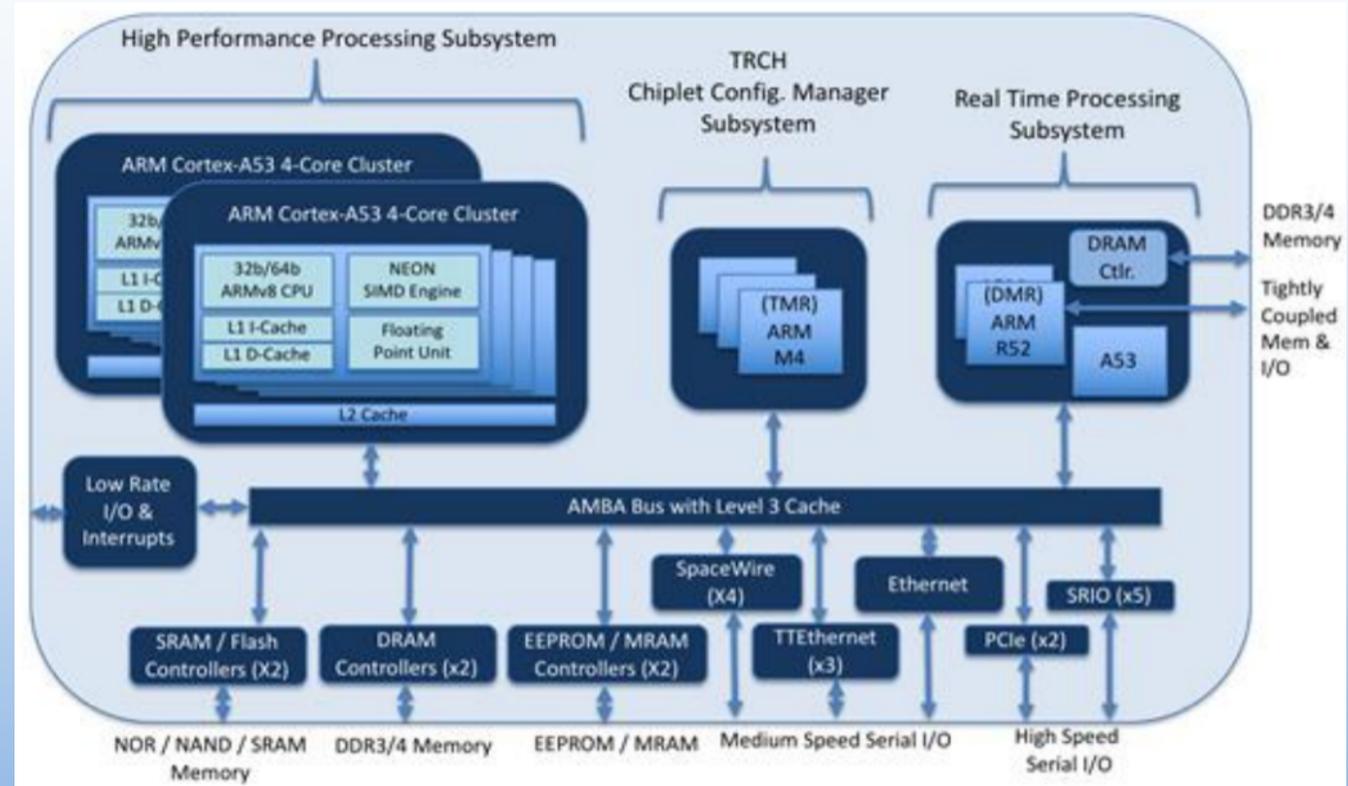
- A5-Based Microcontroller
- Xilinx UltraScale+ MPSoC has dual-core A5s for comparison
 - (And comparison to quad-core A53)
- Also working on getting A5-IP via collaboration



Heterogenous Error Handling

- SEE performance of different subsystems does not need to be the same.
- SOCs already utilize this – think L2 cache error protection vs. L1 cache error trapping.
- But this is a hardware capability vs. a software implementation problem. The hope is that as the industry develops, it will become more common to have systems and software that can support subsystem isolation in the event of an error.
- This type of approach is taken in the HPSC program

<https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20180007636.pdf>





Future

- Tests planned:
 - Testing on Snapdragon 835, 845
 - TI TMS570 – specifically to evaluate with and without fault mitigation enabled
 - ARM Cortex A5 device, for comparison across multiple fabrication situations and soft/hard core IP
- Extract critical info on what fault tolerance works and what doesn't, in order to help specify fault tolerance settings in ARM devices.



end



Crashes

- Crash, Hang, Unhandled Exception, Error Mode, Trap in Trap, Inverted Beer Mug, Runaway Loop, Mouse in Computer (observed once on a test)
 - Often not possible to identify what actually causes the event
 - Always recovered by full power-cycle (turn the mug back over, remove the mouse), and sometimes recovered by just hitting reset
- Unless clearly defined in a test approach/report, you should assume the full set of synonyms (up to beverages and rodents).

Radiation Failures for Test



- In a perfect world, you test everything. In reality you have to pick things and build detection.
 - With limited info, you have to infer what you don't measure directly...
 - Try to understand sensitivity of enough basic structures and subsystems
- Testing basic structures and subsystems
 - SRAM-based targets – such as caches and some types of registers
 - Flip-Flop-based targets – generally the other registers and execution unit pipelines (if possible)
 - Exception/Trap/Interrupt systems (especially catching unexpected events)
 - Data flow through memory controllers, communications systems, etc.
- Testing complex devices with multiple subsystems and fault handling is not deterministic.
 - Cycle-by-Cycle comparison to expectation is problematic
 - Simple targets can be easily tested with simple hardware
 - Most everything else seems to require in-situ detection or using debugging hardware to repeat tests during exposure
- Performance benchmarks and manufacturer's hardware test codes must be modified for SEE testing (poor target control and duty cycle)



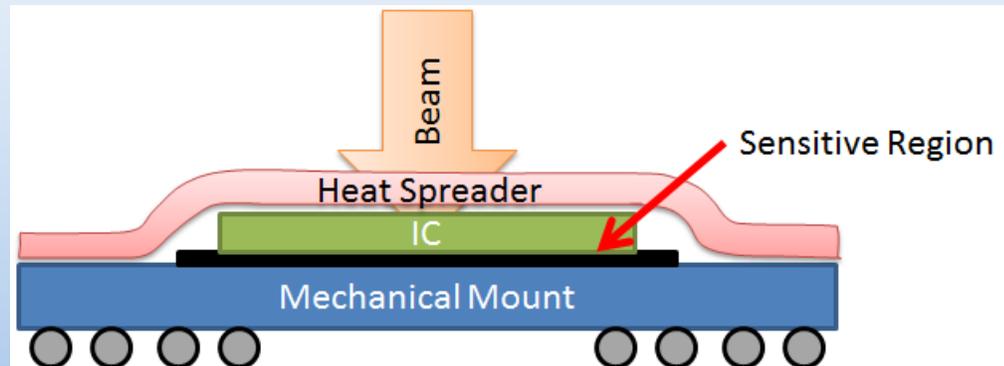


Testing Ideas Seen & Issues

- Test using code running on the DUT
 - E.g. Benchmarks, FFTs, etc...
- Use hardware verification tools
- Use a debugger to access registers and cache
- Use a debugger to run custom code
- Test registers – load values, check
- L1/L2 cache bits – load, check
- Test with operating system
 - Application/flight code?
- Results depend on code robustness
- Poor time-structure
- Quasi-static
- Very few bits...
- Error correction may obscure, code execution may depend on caches

Test Preparation

- Determination of appropriate test facility
 - Summarized: IUCF, UCD, Triumph, TAMU, BNL, UCB, UCL, RADEF, NSRL
- Establish package materials and determine depackaging...



- Selection of DUT board – custom vs. inexpensive manufacturer evaluation board?
 - 10's of k vs. 0.5-4k
 - Note that lower-end evaluation boards may be sufficient (~0.5k)
 - Might be able to rework mfr board with socket



General Trends

- Availability of documentation will get worse
- We are looking to engage commercial manufacturers to see if it is possible to get enough info to assess radiation sensitivity without compromising internal IP
- Multi-chip devices are already a problem, this will get worse. A lot of today's SOCs almost meet the definition of a hybrid or multi-chip module
- Test results will likely become more “high-level” requiring significant conservatism in how test results are applied to flight systems.



More Devices/Uses

- Efforts are underway to explore performance of SOCs with GPUs - cell phone processors, nVidia TX1
- As well as with dedicated GPU chips, can errors in the GPUs be contained? Can software & hardware handle the errors and continue running?
- Examples of GPU testing:
 - Memory transfer
 - Typical image generation/manipulation
 - AI/Machine learning algorithms