National Aeronautics and
Space Administration

High-Performance Spaceflight Computing (HPSC) Middleware Overview

Alan Cudmore

NASA Goddard Space Flight Center
Flight Software Systems Branch (Code 582)

Alan.P.Cudmore@nasa.gov

Agenda

- High Performance Spacecraft Computing (HPSC) Overview
- HPSC Overview
- HPSC Contract
- HPSC Chiplet Architecture
- HPSC System Software
- HPSC Middleware
- NASA HPSC Use Cases
- HPSC Ecosystem
- Summary/Status

High Performance Spaceflight Computing (HPSC) Overview

- The goal of the HPSC program is to dramatically advance the state of the art for spaceflight computing
- HPSC will provide a nearly two orders-of-magnitude improvement above the current state of the art for spaceflight processors, while also providing an unprecedented flexibility to tailor performance, power consumption, and fault tolerance to meet widely varying mission needs
- These advancements will provide game changing improvements in computing performance, power efficiency, and flexibility, which will significantly improve the onboard processing capabilities of future NASA and Air Force space missions
- HPSC is funded by NASA's Space Technology Mission Directorate (STMD), Science Mission Directorate (SMD), and the United States Air Force
- The HPSC project is managed by Jet Propulsion Laboratory (JPL), and the HPSC contract is managed by NASA Goddard Space Flight Center (GSFC)

HPSC Contract

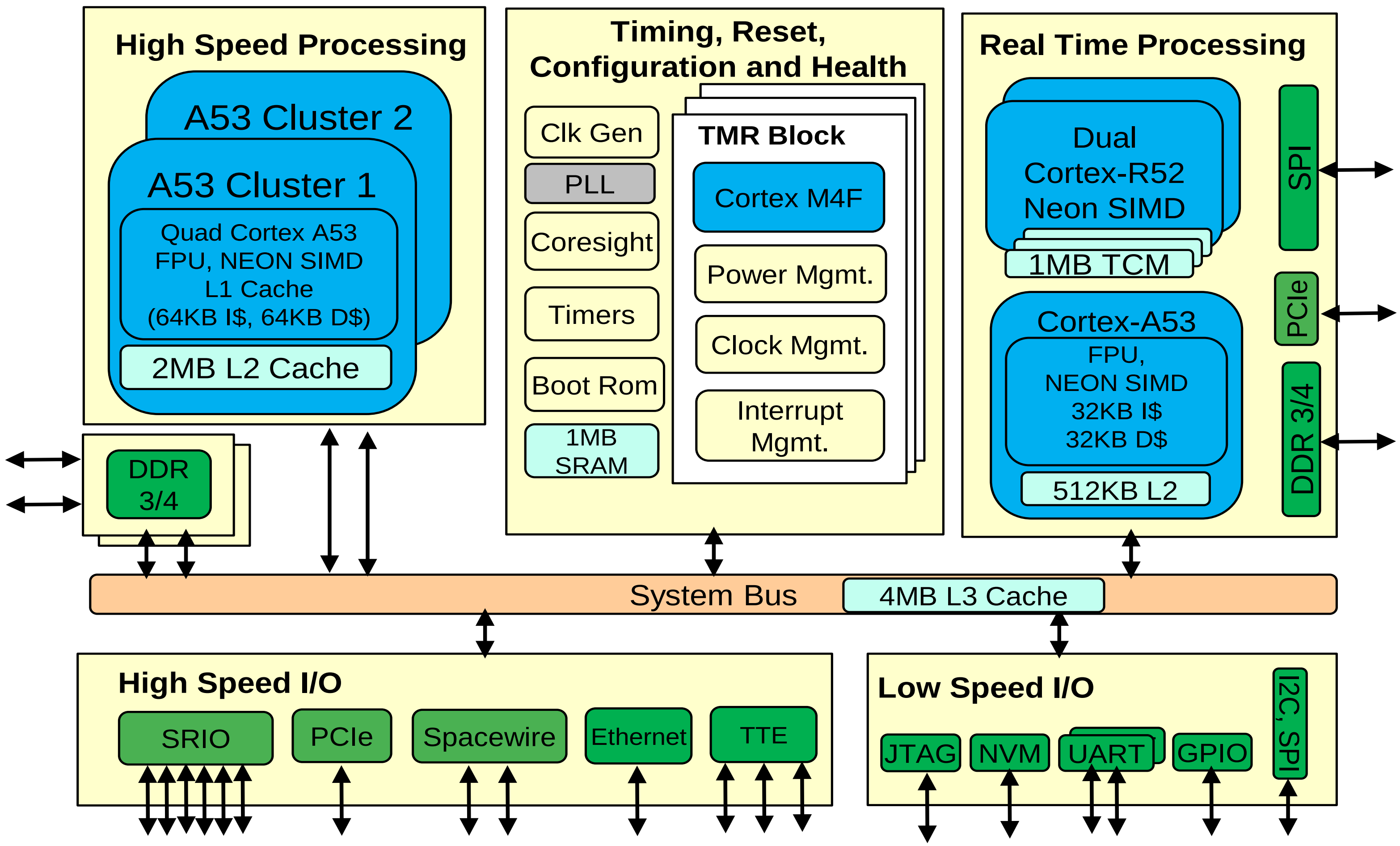
- Following a competitive procurement, the HPSC cost-plus fixed-fee contract was awarded to Boeing

- Under the base contract, Boeing will provide:
  - Prototype radiation hardened multi-core computing processors (Chiplets), both as bare die and as packaged parts
  - Prototype system software which will operate on the Chiplets
  - Evaluation boards to allow Chiplet test and characterization
  - Chiplet emulators to enable early software development

- Five contract options have been executed to enhance the capability of the Chiplet
  - On-chip Level 3 cache memory
  - Added a real-time processing subsystem containing two lockstepable ARM R-class processors and a single ARM A-class processor
  - Triple Time Triggered Ethernet (TTE) interfaces
  - Dual SpaceWire interfaces
  - Package amenable to spaceflight qualification

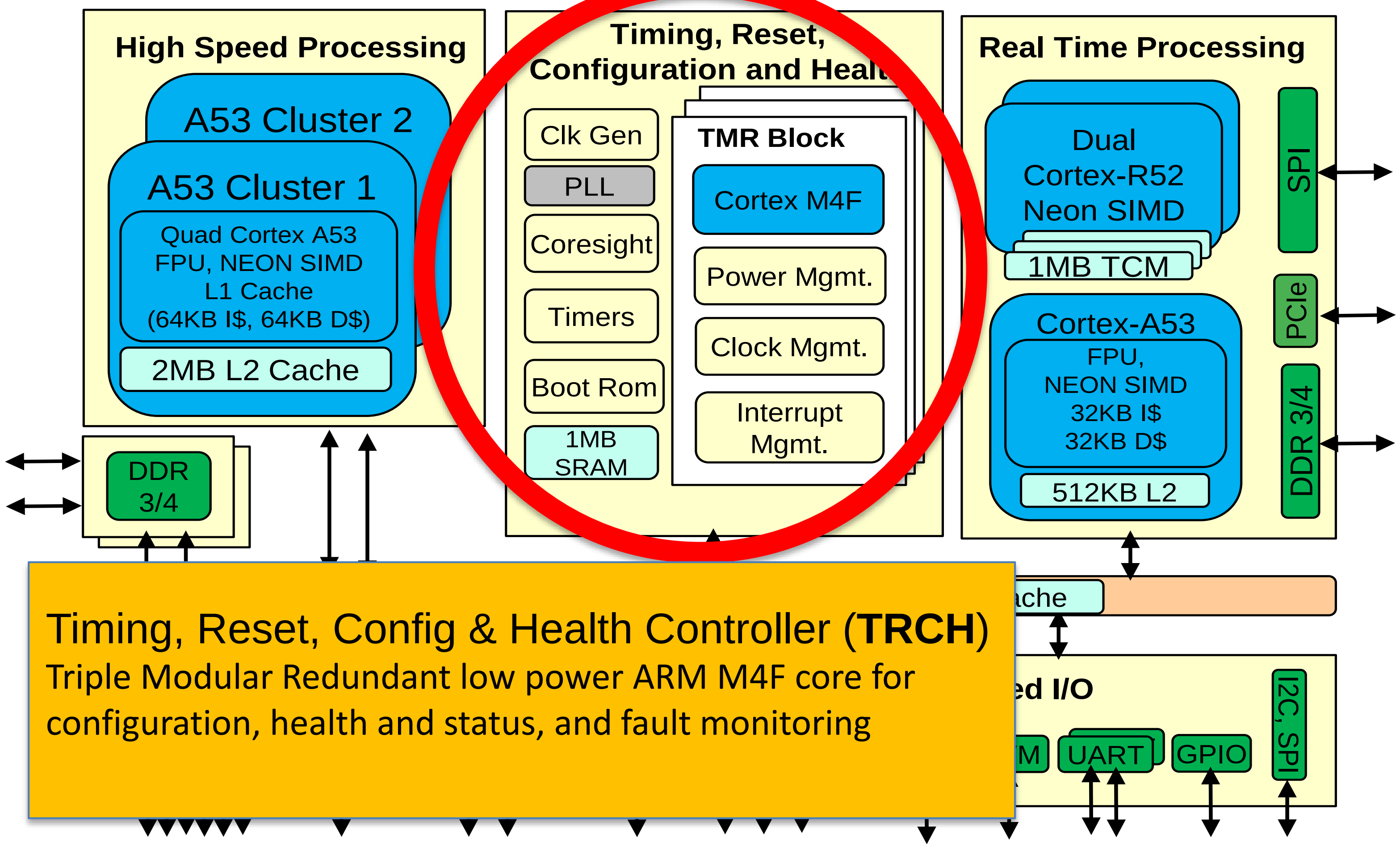
- Contract deliverables are due April 2021

HPSC Chiplet Architecture



This is a non-ITAR presentation, for public release and reproduction from FSW website.

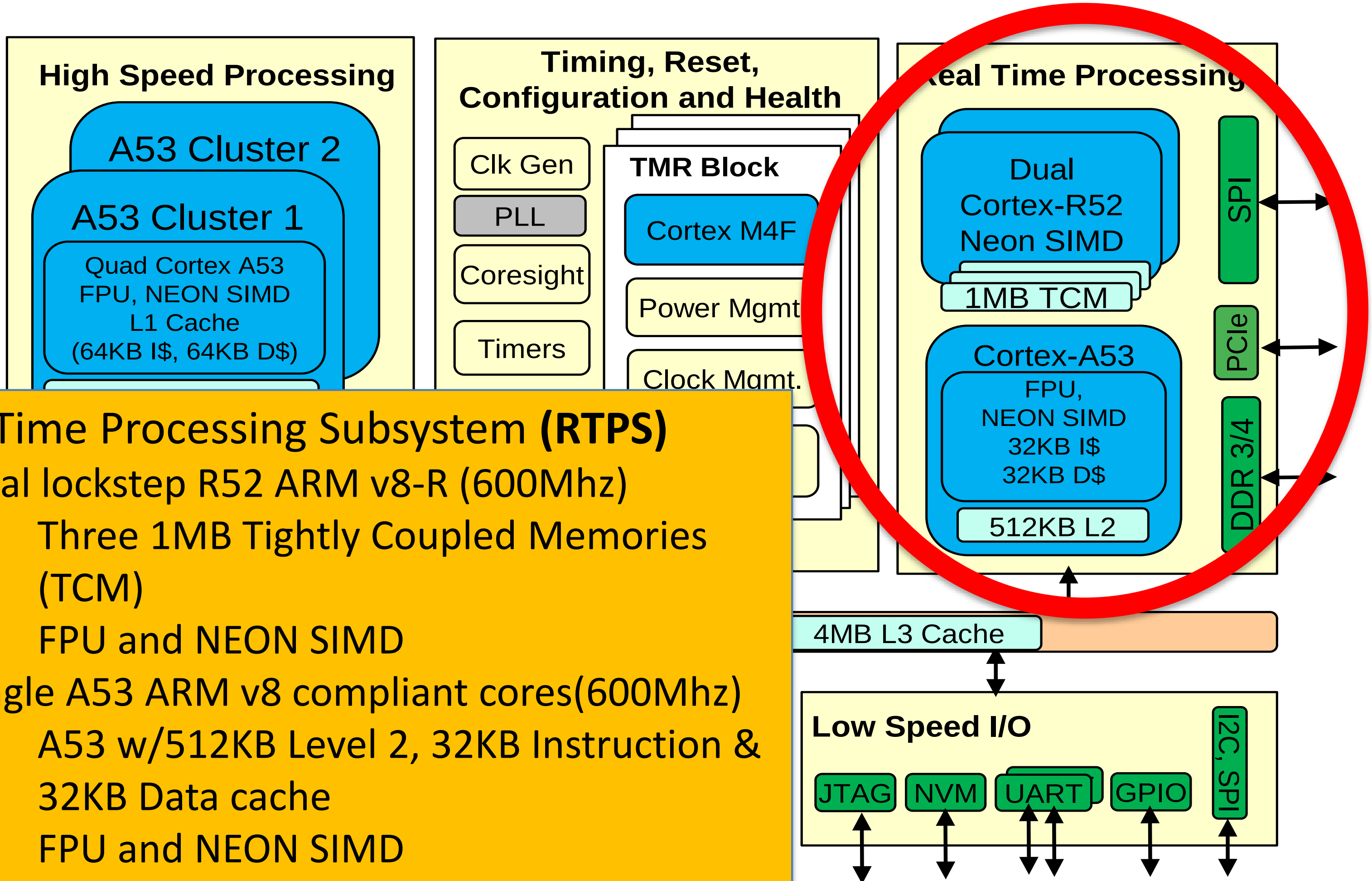
HPSC Chiplet Architecture



Timing, Reset, Config & Health Controller (TRCH)

Triple Modular Redundant low power ARM M4F core for configuration, health and status, and fault monitoring

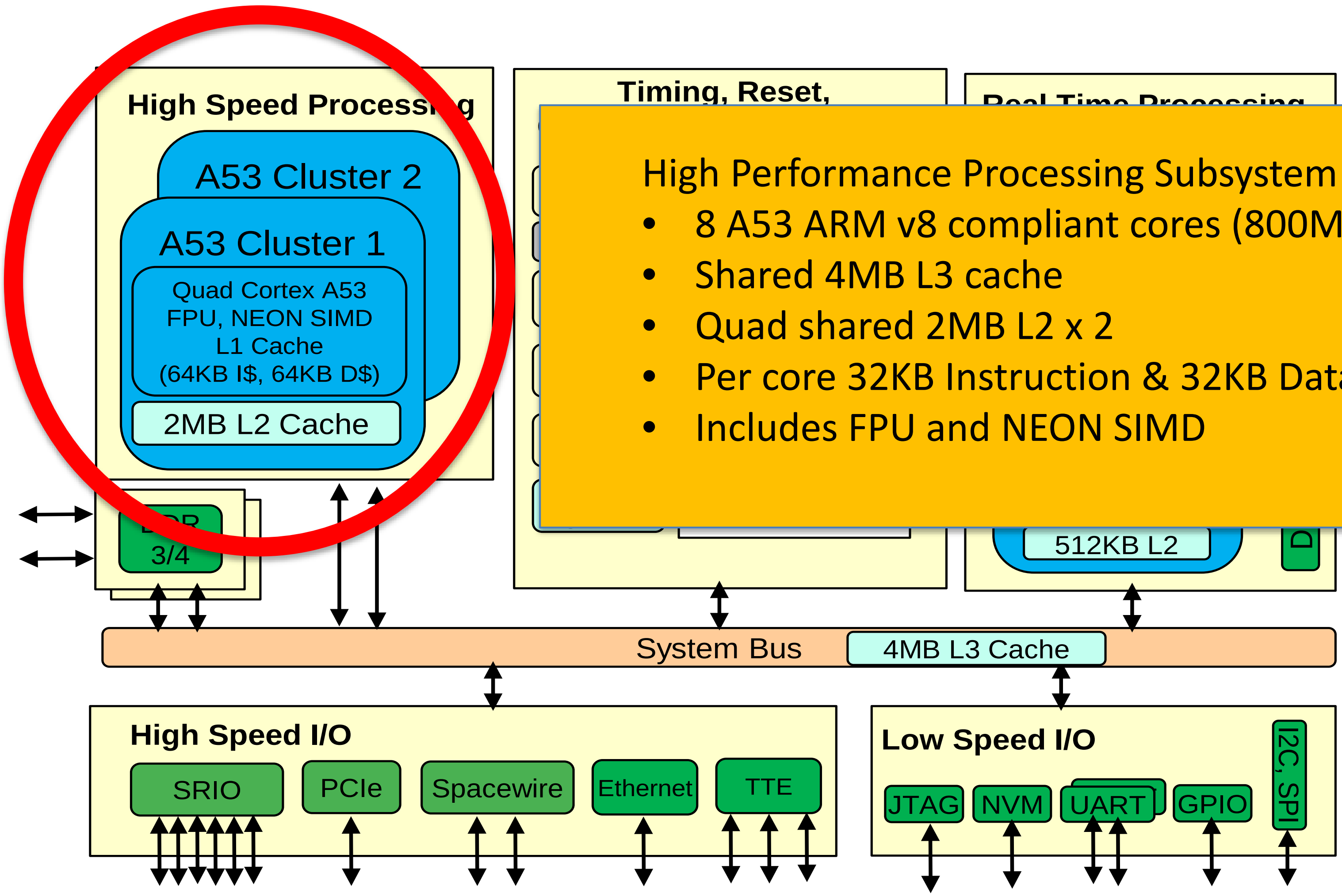
HPSC Chiplet Architecture



Real Time Processing Subsystem **(RTPS)**

- Dual lockstep R52 ARM v8-R (600Mhz)
  - Three 1MB Tightly Coupled Memories (TCM)
  - FPU and NEON SIMD
- Single A53 ARM v8 compliant cores(600Mhz)
  - A53 w/512KB Level 2, 32KB Instruction & 32KB Data cache
  - FPU and NEON SIMD
- Separate PCIe and DDR3/DDR4 interfaces

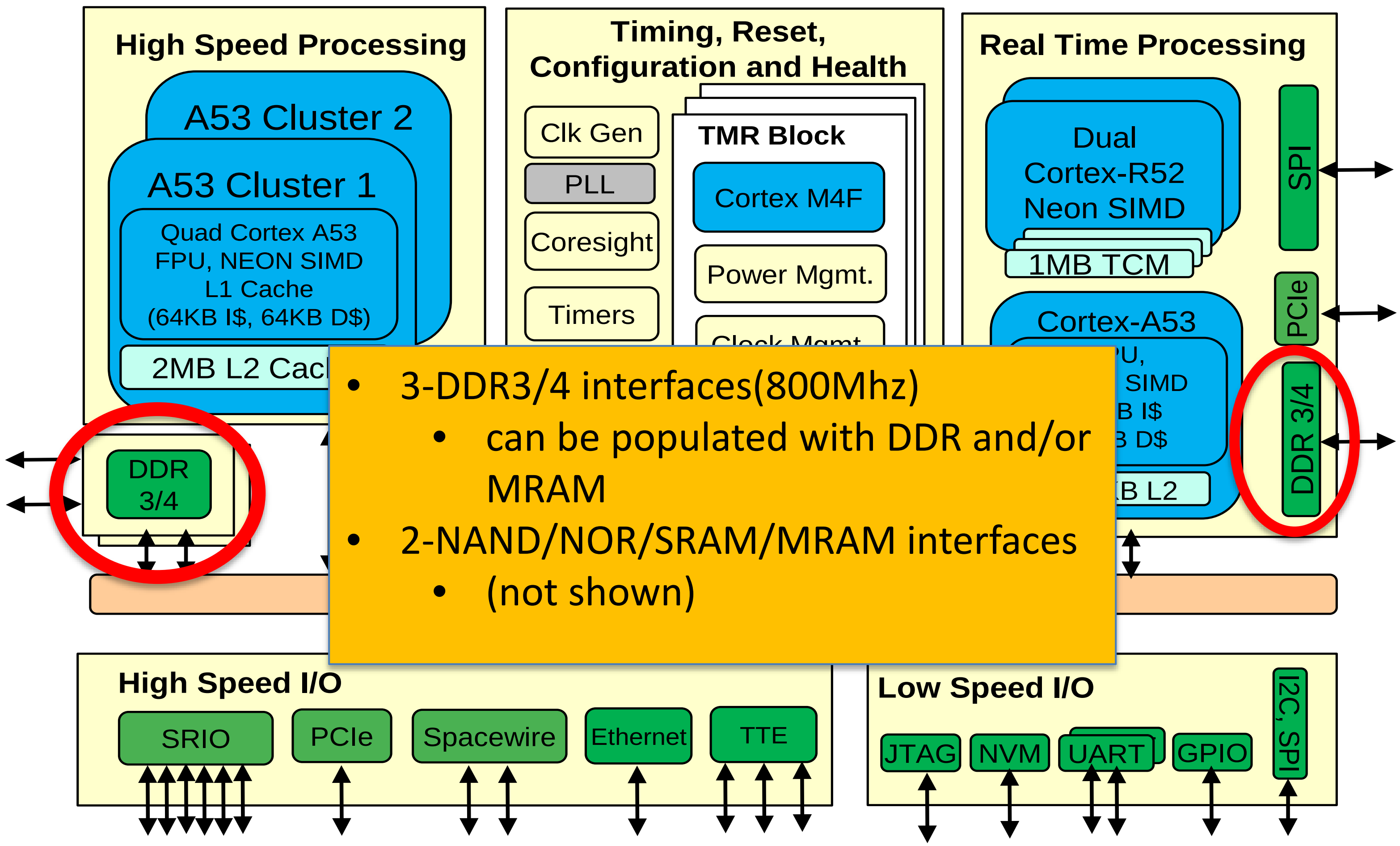
HPSC Chiplet Architecture



High Performance Processing Subsystem **(HPPS)**

- 8 A53 ARM v8 compliant cores (800Mhz)
- Shared 4MB L3 cache
- Quad shared 2MB L2 x 2
- Per core 32KB Instruction & 32KB Data cache
- Includes FPU and NEON SIMD

HPSC Chiplet Architecture

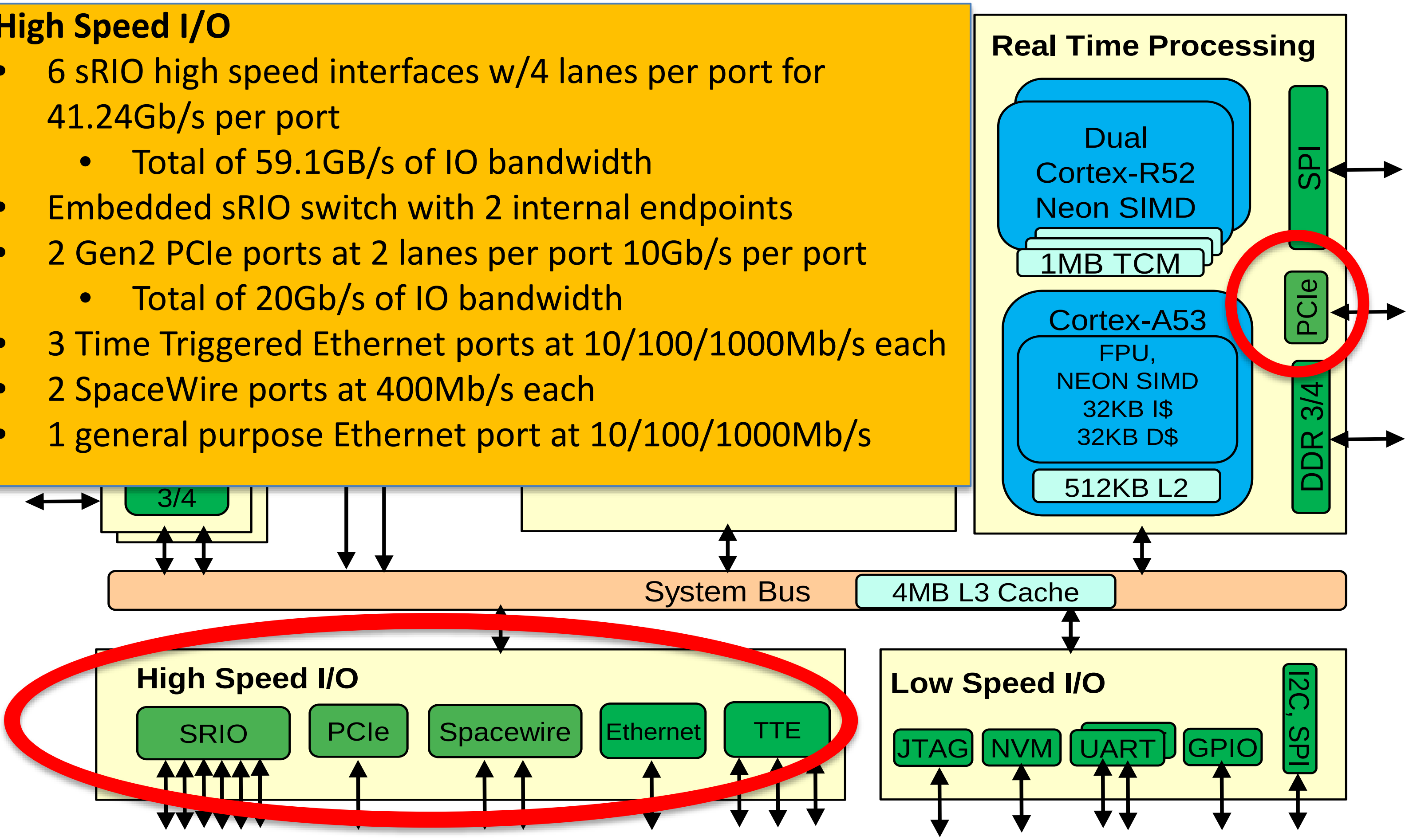


- 3-DDR3/4 interfaces(800Mhz)
  - can be populated with DDR and/or MRAM
- 2-NAND/NOR/SRAM/MRAM interfaces
  - (not shown)

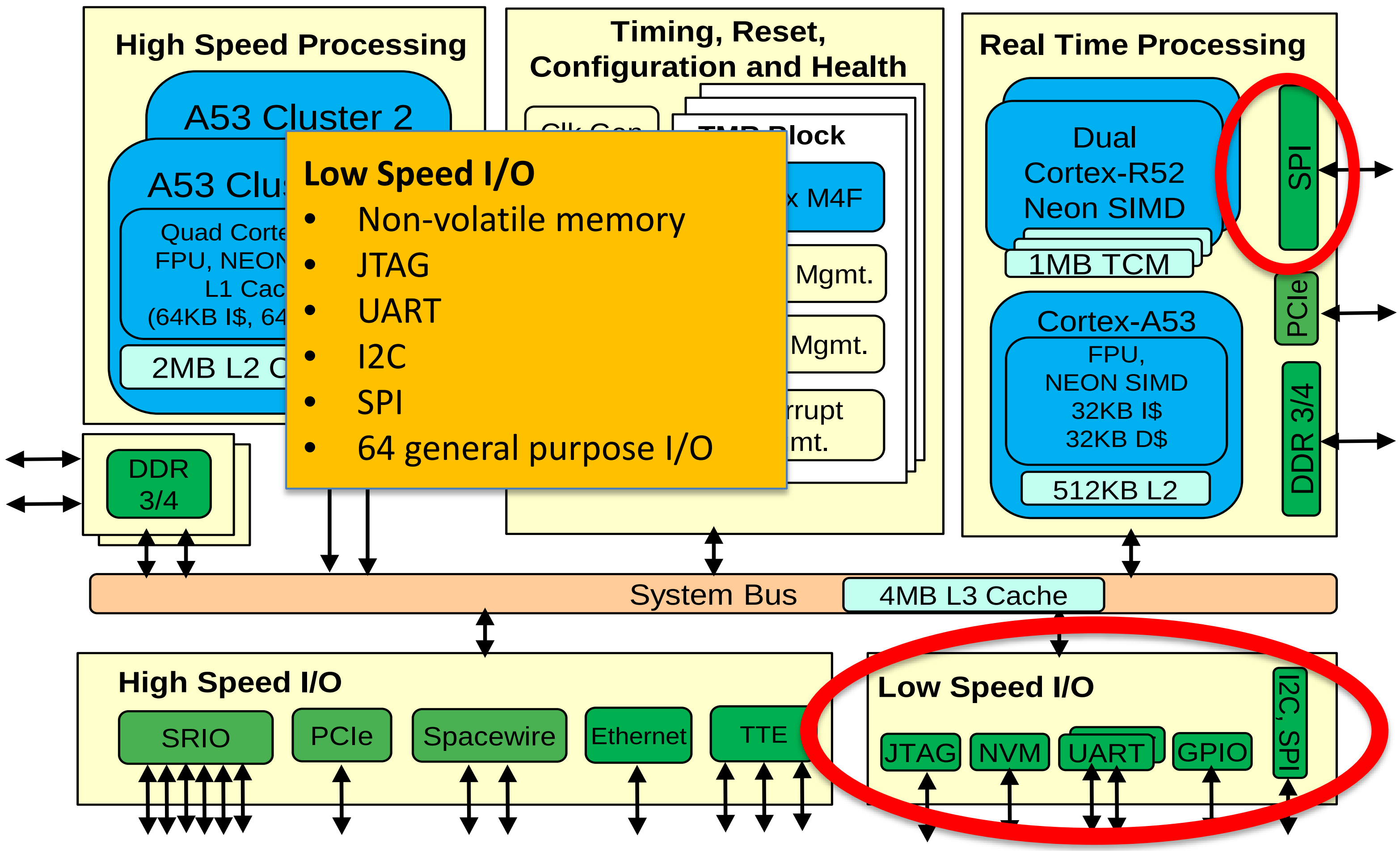
HPSC Chiplet Architecture

High Speed I/O

- 6 sRIO high speed interfaces w/4 lanes per port for 41.24Gb/s per port
  - Total of 59.1GB/s of IO bandwidth
- Embedded sRIO switch with 2 internal endpoints
- 2 Gen2 PCIe ports at 2 lanes per port 10Gb/s per port
  - Total of 20Gb/s of IO bandwidth
- 3 Time Triggered Ethernet ports at 10/100/1000Mb/s each
- 2 SpaceWire ports at 400Mb/s each
- 1 general purpose Ethernet port at 10/100/1000Mb/s



HPSC Chiplet Architecture



HPSC Chiplet Architecture
Fault Tolerance and Debug Features

HW based fault tolerance

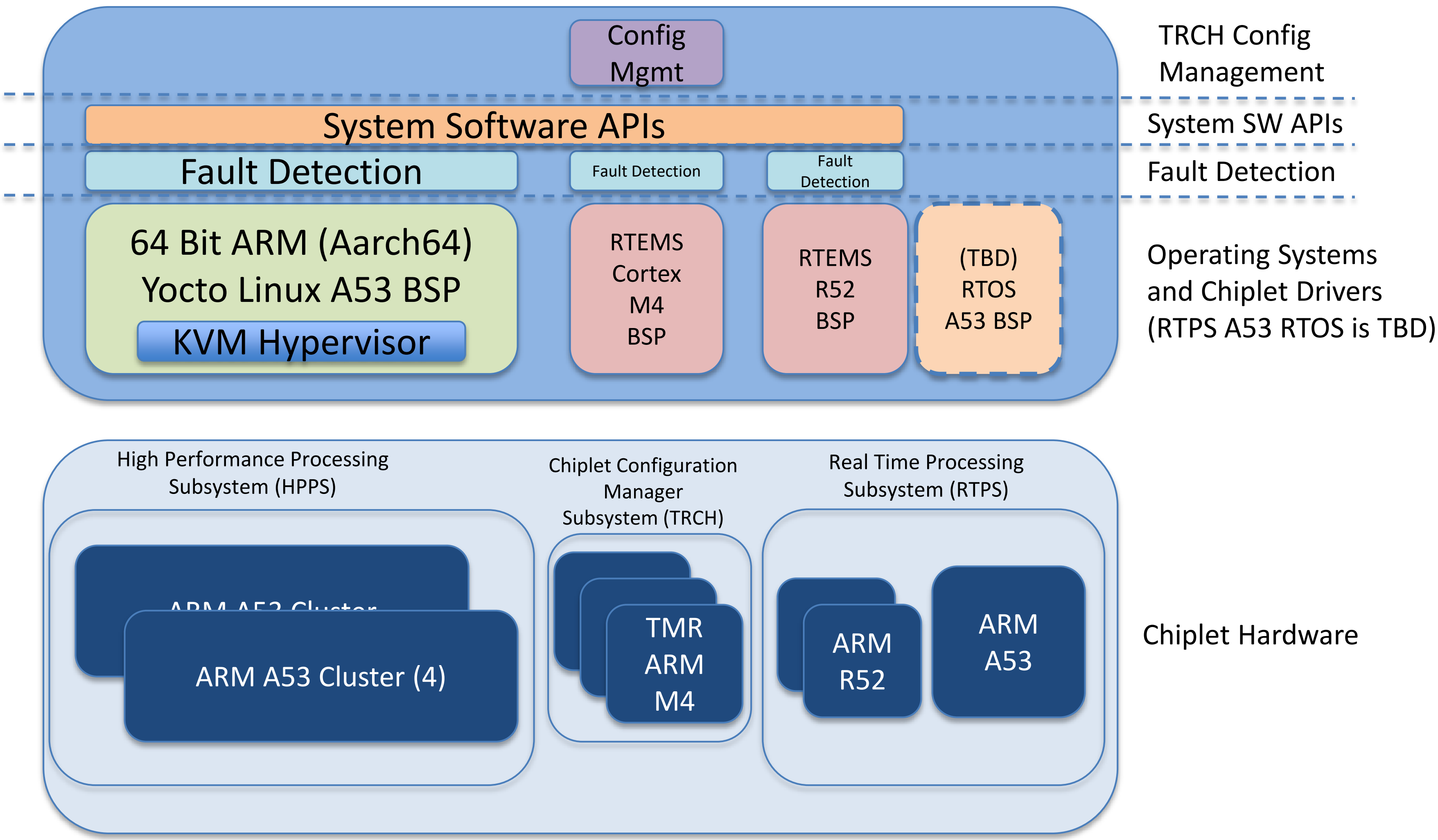
- **EDAC, scrubbing**
- **TMR of critical logic**
- **ARM TrustZone**
- **Access isolation regions**

ARM Coresight debug and trace subsystem

HPSC System Software

- **The HPSC Chiplet will be delivered with a complement of prototype system software developed by Boeing subcontractor University of Southern California/Information Sciences Institute (USC/ISI)**
- **The System Software leverages a large complement of existing open source software including:**
  - Libraries, operating systems, compilers, debuggers, and simulators.
  - Much of the of this software will be unmodified.
- **The System Software consists of:**
  1. Board support packages for Linux and RTEMS
  2. Development tools (e.g., compilers, debuggers, IDEs)
  3. Chiplet Configuration APIs
  4. Mailbox API
  5. Software-based fault tolerance libraries
  6. Chiplet emulators
- **The goal is to build a sustainable software ecosystem to enable full lifecycle software development.**

HPSC System Software

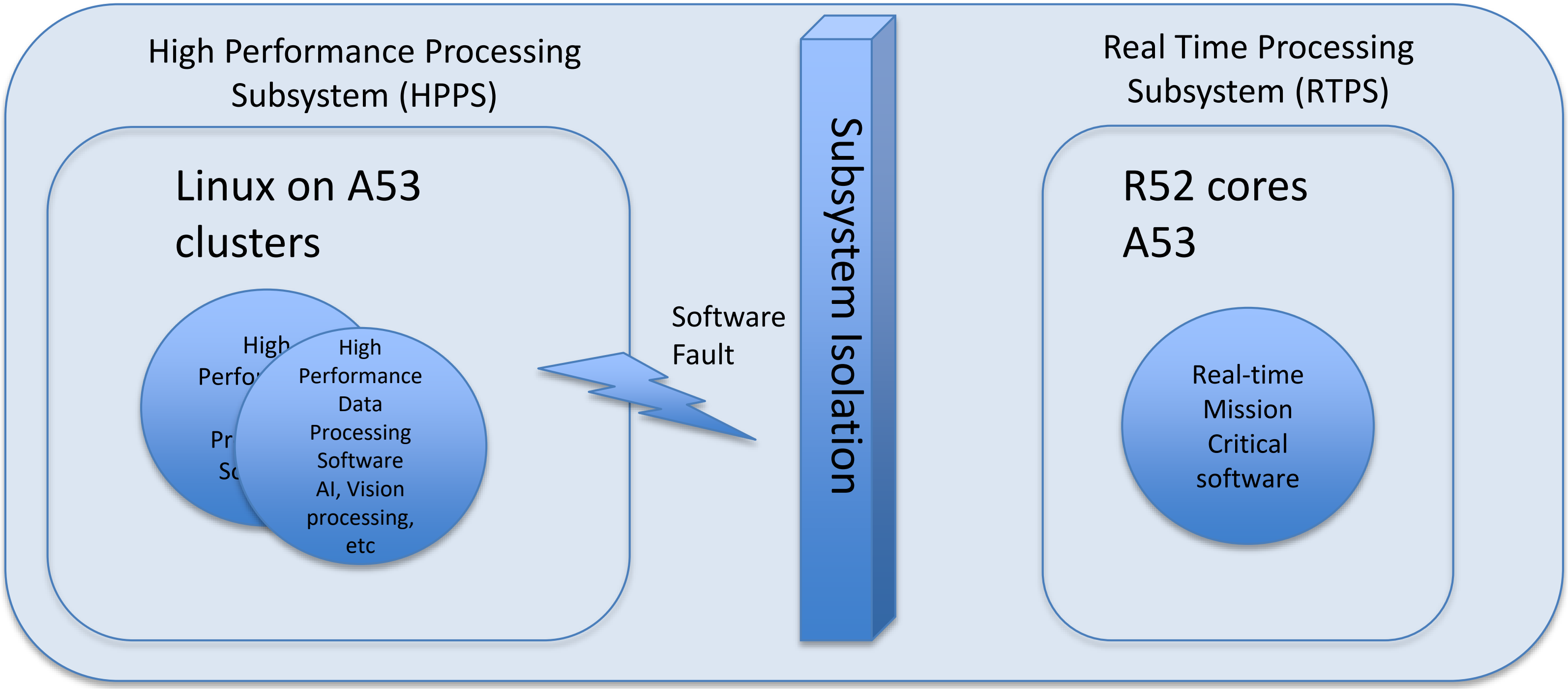


This is a non-ITAR presentation, for public release and reproduction from FSW website.

HPSC System Software

- **HPSC Reference Board Support Package (BSP) Features:**
  - TRCH Cortex-M4 will execute RTEMS with support for:
    - SRIO, low speed I/O, configuration registers, windowed-access into all memories.
  - RTPS Cortex-R52 will execute RTEMS with support for:
    - SRIO, PCIe, Spacewire, TTE, low speed I/O, RTPS DDR, windowed access into HPPS DDR, NVRAM.
  - HPPS Cortex-A53 clusters will execute Yocto Linux with support for:
    - SRIO, PCIe, Spacewire, Ethernet, TTE, HPPS DDR, NVRAM
- **HPSC Chiplet supports running Linux, which makes available a rich and familiar development environment. This improves efficiency of software development in particular for science applications**
  - Many science applications already start off development in Linux only to be ported to a different OS. With HPSC, no extra porting step is required, saving effort and reducing risk

HPSC Fault Isolation



- The HPSC Chiplet ensures that misbehavior in HPPS cannot interfere with critical functions in the RTPS
  - The HPSC Chiplet provides a Trusted Computing Base (TCB) that is small relative to the overall capability of the system.
  - For example, a minimal TCB would consist of flight control, communications, health monitor, and software update management running on the RTOS. These high criticality functions remain safely isolated from any misbehavior in Linux
- One of many potential examples of fault isolation in the HPSC
  - Other examples include the use of a hypervisor or partitioned OS on the HPPS

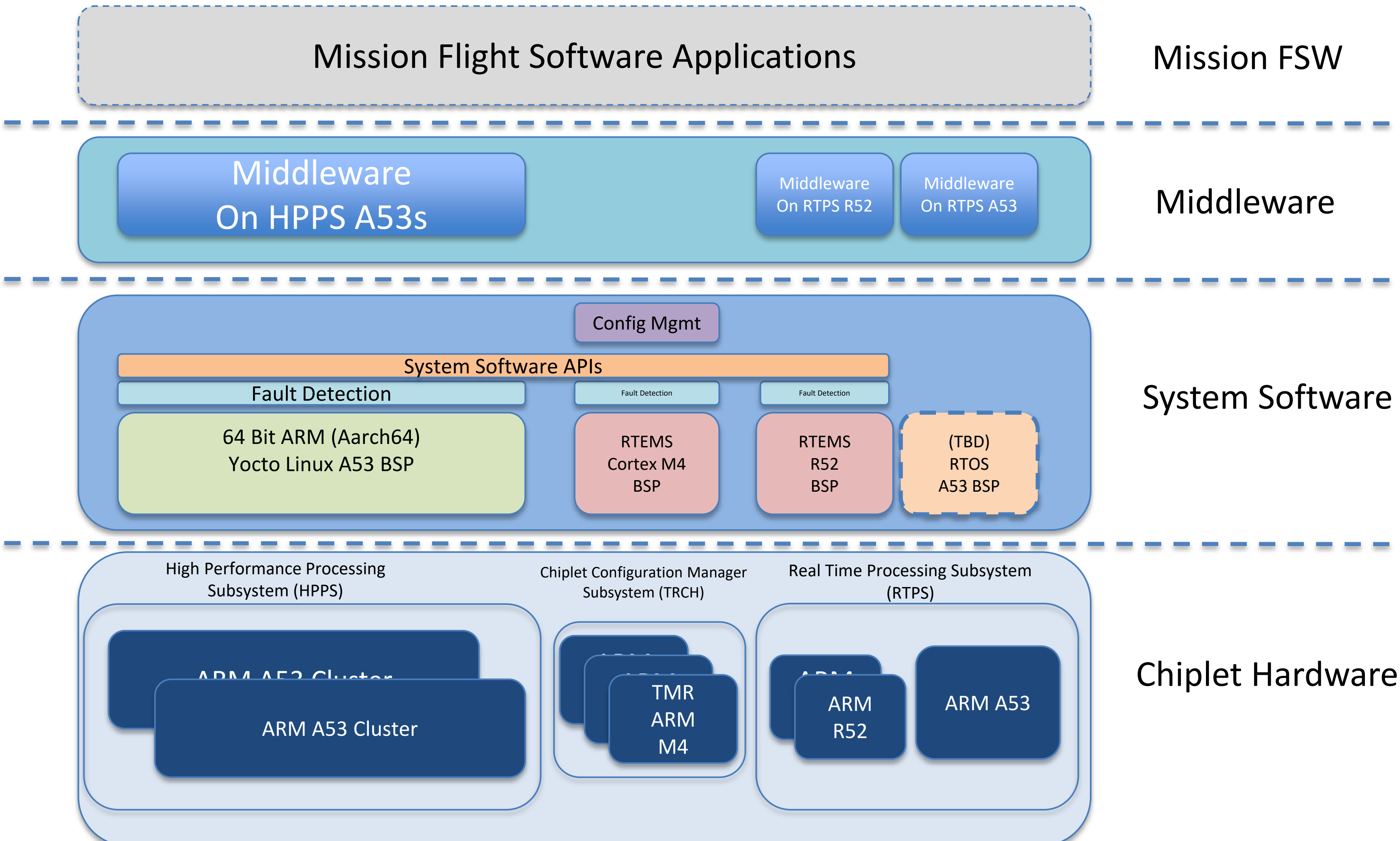
HPSC Middleware

- The Air Force Research Laboratory (AFRL) is funding NASA Jet Propulsion Laboratory (JPL) and NASA Goddard Space Flight Center (GSFC) to develop HPSC Middleware

- HPSC Middleware will provide a software layer that provides services to the higher-level application software to achieve:
  - Configuration management
  - Resource allocation
  - Power/performance management
  - Fault tolerance capabilities of the HPSC chiplet

- Serving as a bridge between the upper application layer and lower operating system or hypervisor, the middleware will significantly reduce the complexity of developing applications for the HPSC Chiplet

HPSC Middleware



Middleware Functions

- **The HPSC Middleware provides the following key functions:**
  - Boot and Load Services
  - Chiplet Configuration Management
  - Chiplet Resource Allocation and Management
  - Chiplet Power and Performance Management
  - Fault Tolerance and Redundancy Management
  - Messaging Services
- **These functions are implemented through 13 Middleware Services**

Middleware Services (1 of 2)

- **Middleware functions are provided by the following services**
  1. Deployment and Multicore Programming Services
     - Deployment of time critical vs computational intensive applications to RTPS & HPPS
     - Thread creation and assignment to specific processing cores for execution
     - Utilize multicore environment to establish parallel processing capability
       - Utilize Linux/MPI features
       - Utilize OpenMP library for multicore programming functions
  2. Machine Information Services
     - Get information about cores, clusters, chiplets, and system
  3. Resource Management/Arbitration Services
     - Manage configuration files and assign available resources to software clients
  4. Memory Management Services
     - MMU configuration, memory allocation to applications
  5. Interrupt Services
     - Configure Interrupt controller to route interrupts to specific cores at runtime
  6. Timing Services
     - Establish, synchronize, and distribute time

Middleware Services (2 of 2)

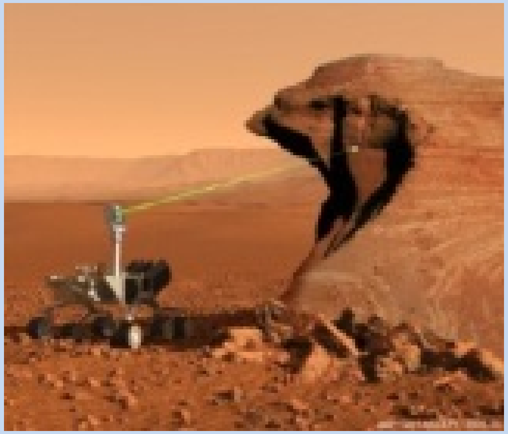
7. Messaging Services
   - Send/receive messages between tasks on the same or different cores and Chiplets
8. Power Management Services
   - Power on/off setting for cores/clusters/Chiplets, and set clock rates
9. Debug and Trace Services
   - Support the interface for software debugging during development
   - Support the capability for performance metrics collection during execution
10. Boot and Load Process and System Configuration Services
    - Provide an Initial Program Loader (IPL) and support various boot types/processes
11. Peripheral IO Services
    - Allocate and configure Chiplet Peripheral I/O to software clients
12. Multi-Chiplet Management Services
    - Configure and manage Inter-Chiplet interfaces (e.g., boot, messaging, resources)
13. Fault and Redundancy Management Services
    - Fault detection (HW/SW faults) and recovery

HPSC Use Cases – Rovers and Landers

Rover

Compute Needs
- Vision Processing
- Motion/Motor Control
- GNC/C&DH
- Planning
- Science Instruments
- Communication
- Power Management
- Thermal Management
- Fault detection/recovery

- **System Metrics**
- 2-4 GOPs for mobility(10x RAD750)
- >1Gb/s science instruments
- 5-10GOPs science data processing
- >10KHz control loops
- 5-10GOPS, 1GB/s memory BW for model based reasoning for planning



Lander

Compute Needs
- Hard Real time compute
- High rate sensors w/zero data loss
- High level of fault protection/ fail over

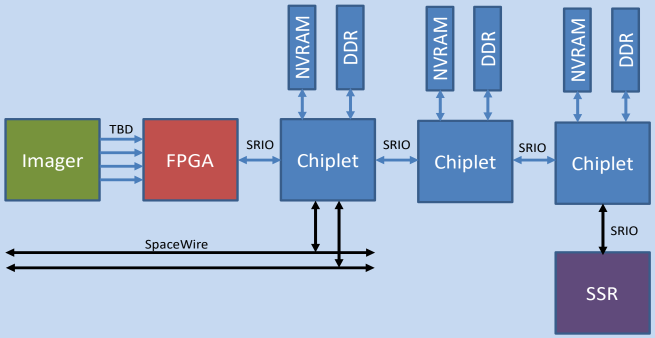
- **System Metrics**
- >10 GOPs compute
- 10Gb/s+ sensor rates
- Microsecond I/O latency
- Control packet rates >1Kpps
- Time tagging to microsecond accuracy

HPSC Use Cases - High Bandwidth Instruments and SmallSats/Constellations

High Bandwidth Instrument

Compute Needs
- Soft real time
- Non-mission critical
- High rate sensors
- Large calibration sets in NV memory

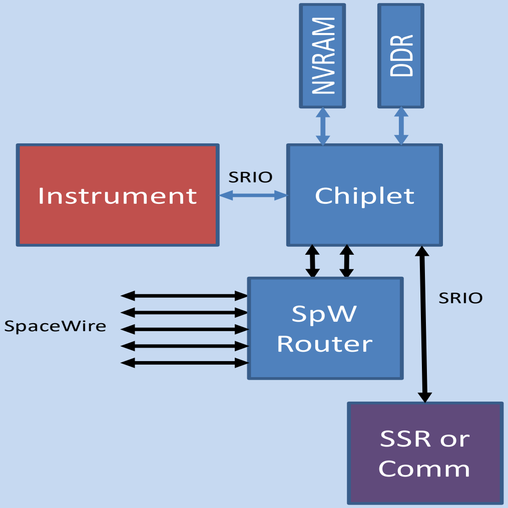
- **System Metrics**
- 10-20 GOPs compute
- >10GB/s memory bandwidth
- >20Gbps sensor IO data rates



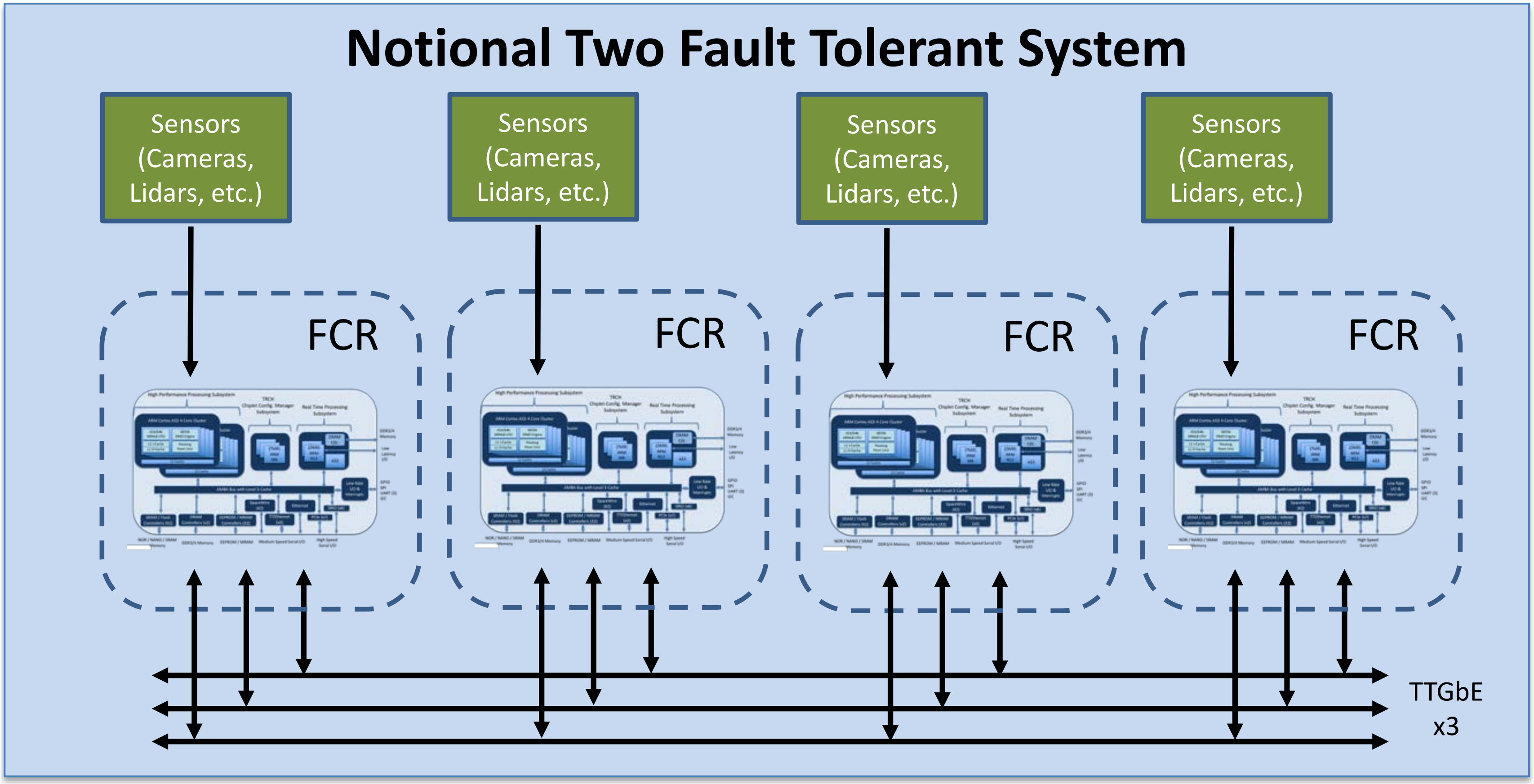
Smallsat

Compute Needs
- Hard and Soft real time
- GNC/C&DH
- Autonomy and constellation(cross link comm)
- Sensor data processing
- Autonomous science

- **System Metrics**
- 2-5Gbps sensor IO
- 1-10GOPs
- 1GB/s memory bandwidth
- 250Mbps cross link bandwidth



HPSC Use Cases – Human Exploration and Operations (HEO) Habitat/Gateway



Broader HPSC Ecosystem

- Beyond the HPSC Chiplet, System Software, and Middleware developments, further investments can implement a robust HPSC avionics ecosystem
  - Advanced Spaceflight Memory
  - Increased RTOS Support
  - Multi-Output Point-Of-Load Converters
  - Coprocessors (GPU, Neuromorphic, etc.)
  - Special Purpose Chiplets (Security Chiplet, etc.)
  - Advanced Packaging (Multiple Chiplets in a Package)
  - Single Board Computers

Conclusion/Status

- **Conclusion**
  - As illustrated by the NASA use cases, our future missions demand the capabilities of HPSC
  - Improved spaceflight computing means enhanced computational performance, energy efficiency, and fault tolerance
  - With the ongoing HPSC development, we are well underway to meeting future spaceflight computing needs
  - The NASA-developed Middleware will allow the efficient infusion of the HPSC Chiplet into those missions
  - Further investments can implement a full HPSC avionics ecosystem
- **Status**
  - USC/ISI delivered System Software Release 1.0 at the May 2018 HPSC Preliminary Design Review
    - Consists of a QEMU based software emulator and initial Yocto Linux based software development kit
  - The NASA JPL and GSFC HPSC Middleware team will complete Middleware release R1 this month (December 2018)
    - Middleware release R1 consists of a subset of the services, mostly targeting A53 Linux functionality while the hardware is being developed

Acknowledgements: Wes Powell (GSFC), Rich Doyle (JPL), Rafi Some (JPL), Jim Butler (JPL), Irene Bibyk (GSFC), Jonathan Wilmot (GSFC), John Lai (JPL), J.P. Walters (USC/ISI), and Jon Ballast (Boeing) for diagrams and use case definitions.

Acronyms (1)

Acronym	Meaning	Acronym	Meaning
AFRL	Air Force Research Laboratory	**FPGA**	Field programmable Gate Array
AMBA	Advanced Microcontroller Bus Architecture	**GNC**	Guidance Navigation and Control
API	Application Programmer Interface	**GOPS**	Giga Operations Per Second
ARM	Advanced RISC Machines	**GPIO**	General Purpose Input Output
BIST	Built In Self Test	**GPU**	Graphics Processing Unit
BSP	Board Support Package	**GSFC**	Goddard Space Flight Center
C&DH	Command and Data Handling	**HEO**	Human Exploration and Operations
CPU	Central Processing Unit	**HPPS**	High Performance Processing Subsystem
DDR	Double Data Rate	**HPSC**	High Performance Spacecraft Computing
DMIPS	Dhrystone Millions of Instructions per Second	**HW**	Hardware
DMR	Dual Modular Redundancy	**I/O**	Input / Output
DRAM	Dynamic Random Access Memory	**I2C**	Inter-Integrated Circuit
FCR	Fault Containment Region	**IDE**	Integrated Development Environment

Acronyms (2)

Acronym	Meaning	Acronym	Meaning
IPL	Initial Program Loader	**NVRAM**	Non Volatile Random Access Memory
ISA	Instruction Set Architecture	**PCIe**	Peripheral Component Interconnect express
ISI	Information Sciences Institute	**QEMU**	Quick Emulator
ITAR	International Traffic In Arms Regulations	**RTEMS**	Real Time Executive for Multiprocessor Systems
JPL	Jet Propulsion Laboratory	**RTOS**	Real Time Operating System
KVM	Kernel Based Virtual Machine	**RTPS**	Real Time Processing Subsystem
MIPS	Millions of Instructions per Second	**SCP**	Self Checking Pair
MMU	Memory Management Unit	**SIMD**	Single Instruction Multiple Data
MPI	Message Passing Interface	**SMD**	Science Mission Directorate
MRAM	Magnetoresistive Random Access Memory	**SPI**	Serial Peripheral Interface
NAND	NOT-AND logic	**SPW**	Spacewire
NASA	National Aeronautics and Space Administration	**SRAM**	Static Random Access Memory
NEON	Single Instruction Multiple Data architecture	**SRIO**	Serial Rapid Input Output

Acronyms (3)

Acronym	Meaning	Acronym	Meaning
SSR	Solid State Recorder		
STMD	Space Technology Mission Directorate		
SW	Software		
TBD	To Be Determined		
TMR	Triple Modular Redundancy		
TRCH	Timing, Reset, Configuration, and Health		
TTE	Time Triggered Ethernet		
UART	Universal Asynchronous Receiver Transmitter		
USC	University of Southern California		
VMC	Vehicle Management Computer		