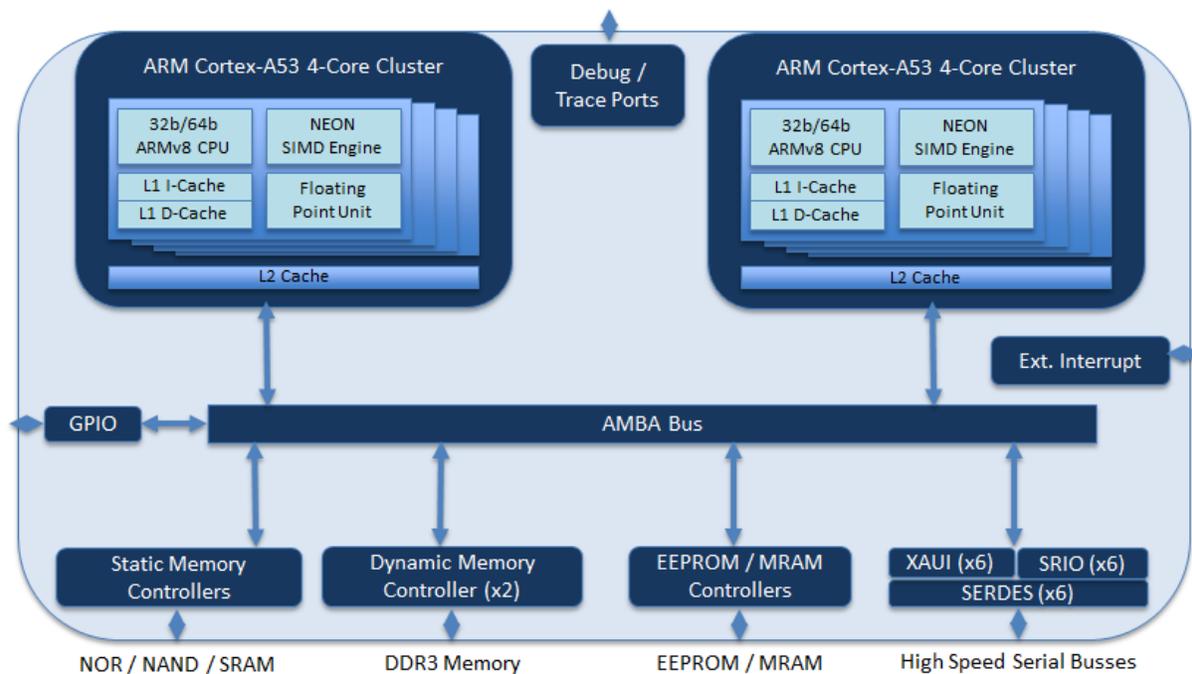


HPSC Requirements 10/08/2015

The government team has devised the HPSC “chiptlet” concept that (a) meets NASA’s future onboard computing needs, and (b) can be developed within the available funding profile. As illustrated below, the chiptlet concept leverages the COTS ARM A53 IP along with COTS peripheral IP, and can meet NASA’s performance, power, and radiation tolerance needs when implemented via existing RHBD technology. The chiptlet includes multiple Serial RapidIO (SRIO) for high bandwidth communication, and multiple interfaces to high speed off-chip memory. The SRIO interfaces can also be used as AMBA-bus bridges allowing multiple chiptlets to be tiled or cascaded to increase bandwidth or for improved fault tolerance.



Offerors recommending solutions that do not utilize the processor IP shown in the above diagram must provide in their future proposal an analysis indicating that their proposed design meets or exceeds the all requirements listed below and rationale as to why the proposed approach is superior to that shown above.

1 General Processing Capabilities

Identifier	Requirement	Rationale
2.1.1.a	Shall provide general purpose processing cores that are at least 64 bits wide with full IEEE 754 Floating Point capability (including double precision), a 128 bits wide SIMD capability, a desired power performance is 5W for 15 GOPS with 50% utilization across all I/O and memory interfaces, and scalable to <1W for 1 GOP for 10% utilization on a single I/O or memory interface.	Provide multiple processing cores in order to support highly parallel applications, accelerate high-throughput kernels and to provide a high degree of granularity for power management, fault tolerance and program unit distribution.
2.1.1.b	Shall provide cache coherency across processor cores within the chiplet while also providing the capability for the coherency to be dynamically and selectively disabled.	Some applications need cache coherency so that they are operating on the same data. However, cache coherency in other application is problematic, for fault tolerance reasons. Cache coherency is not required across multiple chiplets.
2.1.1.c	Shall be based on commercially available hardware and software IP (processing cores, external I/O and memory interfaces, software stack and development environment).	Leverage available technologies from an active development ecosystem.
2.1.1.d	Shall be capable of executing multiple concurrent applications and support parallel processing across the set of processing cores.	The chiplet should readily support both multi-core and parallel (i.e. symmetric and asymmetric) applications.
2.1.1.e	Shall provide capability for tiling or cascading multiple chiplets in order to implement a multi-chiplet system for redundancy and/or increasing processing capability.	Readily scale MIPS or fault tolerance in the computing system by adding or removing chiplets.
2.1.1.f	Shall provide an elapsed timer for maintaining time knowledge across all processor cores in the system	A monotonically increasing timer that can be used as mission time (e.g. MET, SCT, etc.) and can also be used to time tag events.
2.1.1.f.1	The elapsed timer shall have a minimum resolution of [32] bits for seconds and [32] bits for subseconds that are read-accessible from all processor cores.	

2.1.1.f.2	The elapsed timer shall increment on the chiplet system clock.	
2.1.1.f.3	The elapsed timer shall synchronize on the external time pulse with a maximum latency of 10 nsec.	
2.1.1.f.4	The elapsed timer shall free run in the absence of the external time pulse.	
2.1.1.f.5	The elapsed timer shall be settable from software, in a fault tolerant manner.	Rogue software cannot simply make a single write operation and change the elapsed timer.
2.1.1.g	Shall provide a software programmable external pulse that is based on the elapse timer and is accessible from any processor core, in a fault tolerant manner.	A chiplet can generate the system sync pulse for the other chiplets.
2.1.1.h	Shall provide a processor core interrupt timer with a minimum resolution of [32] bits for seconds and [32] bits for subseconds that are read-accessible from the local processor core.	Defines a base rate group for a real-time control system.
2.1.1.h.1	The processor core interrupt timer shall increment on the chiplet system clock and maintain phase with the elapsed timer.	
2.1.1.h.2	The processor core interrupt timer shall generate an interrupt and reload the programmed value at each expiration of the local timer.	
2.1.1.h.3	The processor core interrupt timer shall be unmaskable.	
2.1.1.i	Shall have a dedicated watchdog timer per processor core that resets the processor core upon expiration.	Mechanism to verify critical software health on processor core.
2.1.1.i.1	The watchdog timer shall have a maximum expiration of at least 1024 seconds, assuming the maximum clock rate.	
2.1.1.i.2	The watchdog timer shall reload the maximum expiration value upon reset.	
2.1.1.i.3	The watchdog timer shall have a programmable expiration value.	

2.1.1.i.4	The watchdog timer shall provide an indication that it expired and the indication is accessible to any processor core, for fault tolerance reasons.	A watchdog expiration would be monitored by a health and/or configuration monitor executing a core that is different from the processor on which the expiration occurred.
2.1.1.i.5	The watchdog timer shall have enable/disable control registers that are accessible in a fault tolerant manner.	Software can disable the watchdog, through some fault tolerant mechanism, so that lower criticality software to execute on a processor core without requiring timely resetting of the watchdog.
2.1.1.j	Shall provide a watchdog timer for the boot process.	Mechanism to verify a health boot on a processor core.
2.1.1.k	Shall provide each processor core with [4] general purpose software programmable timers that are accessible by its core.	Mechanism for time-critical, aperiodic functions.
2.1.1.k.1	The general purpose software programmable timer shall produce an interrupt upon expiration.	Mechanism for time-critical, aperiodic functions.
2.1.1.k.2	The general purpose software programmable timer shall have a programmable expiration value	Mechanism for time-critical, aperiodic functions.
2.1.1.l	Provide [8] general purpose interrupts for each processor core.	Limit for external interrupt sources per processor core.
2.1.1.m	Shall receive and selectively distribute a minimum of [8] external interrupts to the processor cores.	Limit for external interrupt sources per chiplet, which are routed to the processor cores.
2.1.1.n	Shall provide a Power on Reset (PoR) pin which has asynchronous assert and synchronous de-assert.	External reset for the chiplet will cause the entire chiplet to reset in a prescribed manner.
2.1.1.o	Shall provide the capability for software, in a fault tolerant manner, to run built-in self-tests and remove faulty cores or otherwise recover correct operations.	Detect and disable faulty IP cores during boot or application execution and allow software to reset/restart IP cores as a mechanism to recover the faulty IP core.
2.1.1.p	Shall support resetting individual processor cores through software control, in a fault tolerant manner.	Low level mechanism to reset user applications on a multi-core software system.
2.1.1.q	Shall support resetting individual 4-core cluster through software control, in a fault tolerant manner.	Mid level mechanism to reset user applications on a multi-core software system.

2 Power Management

Identifier	Requirement	Rationale
2.1.2.a	Shall provide the capability to dynamically power on/off IP cores via software control. Include the following, but not limited to: processors, memory, I/O and debug unit.	Scale capability and power consumption of the chiplet based on the IP cores required for the given user application.
2.1.2.b	Shall provide a capability to reduce the chiplet system clock rate to reduce the power consumption of the chiplet.	Scale power consumption of the chiplet based on the MIPS/MHz required for the given user application.
2.1.2.c	Shall provide a chiplet sleep/standby mode dissipating less than 100mW and performing no computational processing, while awaiting an external event in order to "wake up" in an operational state.	Allows the chiplet to be placed in a sleeping state after user application initialization so that it can quickly resume execution upon receipt of an external discrete event (e.g. external interrupt, external GPIO, etc.). The time to resume execution should be much less than the time for a cold boot and user application initialization.
2.1.2.c.1	Upon waking from the sleep/standby mode, the processor cores shall resume execution from the point at which they were put to sleep/standby or from a well-defined wake up state within [1] second.	
2.1.2.c.2	Shall maintain the elapsed timer while in sleep/standby mode.	
2.1.2.c.3	Shall maintain the external memories while in sleep/standby mode.	

3 Fault Tolerance

Identifier	Requirement	Rationale
2.1.3.a	Shall provide the capability to autonomously, in real time, detect errors, prevent propagation of these errors past well-defined error containment boundaries, resume proper execution, provide prompt notification to software, and log the errors.	IP core errors/faults (e.g. anomalous state, corrupted data, time out, etc.) are detected and corrected with tolerable variance in the execution time. Uncorrectable IP core errors/faults result in the IP being halted and the affected processor core(s) receiving an interrupt. Both corrected and uncorrected

		errors/faults are logged for later analysis by software.
2.1.3.b	Shall support N-Modular Redundancy (NMR) through hardware and/or software techniques.	The chiplet design should not preclude its use in triple or quintuple modular redundant system (i.e. voting logic is external to the chiplet). The voting logic is not necessarily done in hardware, but may be hardware assisted.
2.1.3.c	Shall support parallel checkpoint/rollback through hardware and/or software techniques.	Mechanism to capture the chiplet state for testing the chiplet hardware, debugging parallel applications and debugging multi-core applications.
2.1.3.d	Shall prevent applications or devices from reading and/or writing into address spaces reserved for other applications or devices, for purposes of security and fault tolerance.	Have memory space partitioning and memory access protection to safeguard against faulty software or faulty hardware accesses.
2.1.3.e	Shall prevent a single hardware error from causing a violation of address protection boundaries.	No SEE can cause the IP to violate the address protection boundaries.
2.1.3.g	Shall provide the capability to detect and disable communication from "babbling" IP cores.	Contain faulty IP cores so that they don't consume all the communication bandwidth and starve the other IP cores.
2.1.3.h	Shall provide fault tolerant mechanisms for management and configuration of the processor cores.	Rogue software cannot simply make a single write operation and change the management and configuration of the processor cores.
2.1.3.i	Shall provide error detection and protection for all internal memories.	Have local memories, caches, ALUs and GPRs be robust to radiation effects.
2.1.3.j	Uncorrectable internal memory errors shall cause a refresh operation or a reset, as appropriate.	
2.1.3.k	Shall store the addresses for the uncorrectable internal memory	The user can analyze the uncorrectable internal memory errors to detect failure trends and/or

	errors in a memory that is accessible from any processor core.	determine if an IP core is becoming faulty.
--	--	---

4 Memory and I/O

Identifier	Requirement	Rationale
2.1.4.a	Shall provide ECC protected external memories.	Have external memories that are robust to radiation effects.
2.1.4.b	Shall provide a memory controller that concurrently detects and recovers from whole-chip memory failures (e.g. SEFI).	Have a robust memory controller that is able to operate through faults that are typical in a radiation environment and also typical of a long service life.
2.1.4.c	Shall provide memory controllers that support NOR Flash.	Application program storage
2.1.4.d	Shall provide memory controllers that support NAND Flash.	File system, engineering data, etc. storage
2.1.4.e	Shall provide memory controllers that support 2x DDR3 ports.	Computation
2.1.4.f	Shall provide memory controllers that support 2x SRAM ports.	Scratchpad storage
2.1.4.g	Shall provide memory controllers that support MRAM.	Boot software storage
2.1.4.h	Shall provide memory controllers that support EEPROM.	Boot software storage; can be multiplexed with MRAM pins
2.1.4.i	DDR3 memory controller shall complete and continue its refresh cycles even through a system reset.	So that DRAM is not corrupted
2.1.4.j	Shall allow user configurability to allow operation with only a single DDR3 or SRAM port.	Support system configurations where only a single port is populated.
2.1.4.k	Shall support DMA transfers between I/O and memory ports.	Support DMA transfers between IP cores on the chiplet (on-chip and off-chip).
2.1.4.l	Shall tolerate DDR3 memory faults from single event effects, including bit upsets, SEFIs, and whole-chip failures.	Have RAM that is able to operate through faults that are typical in a radiation environment and also typical of a long service life.
2.1.4.m	Shall tolerate and operate through a SEE on any single memory chip in the memory array.	Have reliable external memories in a radiation environment, for fault tolerance.
2.1.4.o	Shall provide a hardware implemented memory scrubbers for SRAM and DDR3 memories.	Have RAM that is able to operate through faults that are typical in a

		radiation environment and also typical of a long service life.
2.1.4.p	Memory scrubbers shall issue a maskable interrupt upon the occurrence of an uncorrectable bit error.	Have a memory scrubber that operates in the background and corrects memory errors. Uncorrectable memory errors are communicated to the affected processor core.
2.1.4.p.1	Memory scrubbers shall have dedicated register for correctable error counts.	The memory scrubber maintains statistics on errors for later analysis by software.
2.1.4.p.2	Memory scrubbers shall have dedicated register for uncorrectable error counts.	
2.1.4.p.3	Memory scrubbers shall have dedicated registers for first and last correctable error addresses.	
2.1.4.p.4	Memory scrubbers shall have dedicated registers for first and last uncorrectable error addresses.	The memory scrubber maintains pointers to the addresses range of the uncorrectable errors. Can be later used by software to map around the defective memory.
2.1.4.p.5	Memory scrubbers shall have dedicated registers for its control and status (e.g. enable/disable, scrub rate, etc.).	Separate registers for memory scrubber status and memory scrubber control.
2.1.4.q	Shall provide 6 serial I/O ports (can be implemented with multiple lanes) capable of being configured as XAUI or SRIO 3.1 for chiplet-to-chiplet and high-speed spacecraft data plane interconnect.	Chiplet will support high-speed serial communication.
2.1.4.r	Shall provide independent configuration of XAUI or SRIO serial port data rate and lane assignments.	
2.1.4.s	Shall provide a minimum of one Ethernet 10/100 Mbps interface.	Chiplet will support standard Ethernet communication.
2.1.4.t	Shall provide [32] GPIO that are accessible to any processor core.	General discrete signal mechanism that can be used for communication with digital logical that is external to the chiplet.
2.1.4.u	All I/O devices shall include an on-chip loop-back test mode for health verification.	Chiplet will support internal loopback on its I/O devices to interface health checking and also early software development.

2.1.4.v	Shall provide 32 bits of general purpose I/O, that can be configured to be single-ended or as 16 differential pairs.	Chiplet will support GPIO.
2.1.4.w	Shall provide serial console interface.	Chiplet will support a terminal interface to support embedded software development.

5 Packaging

Identifier	Requirement	Rationale
2.1.5.a	Shall employ a packaging approach that is amenable to space qualification.	Chiplet will have path-to-flight packaging.
2.1.5.b	Shall provide sufficient heat flow to dissipate worst-case thermal load at 125C ambient, assuming at 40C cold plate.	

6 Trace and Debug Support

Identifier	Requirement	Rationale
2.1.6.a	Shall provide a debug and trace capability in compliance with the ARM Debug Interface Architecture Specification v5.2 (or later) and/or JTAG to provide low-level hardware debugging capability.	Have a JTAG or equivalent mechanism for hardware and software testing.
2.1.6.c	Shall provide a debug and trace capability that enables a user to inject data patterns and test vectors into the chiplet memories and I/Os.	
2.1.6.d	Shall provide time correlated trace capability for software operating on individual processing cores, traffic on the on chip network, and traffic and state in memory and I/O controllers.	
2.1.6.e	Shall have a debug capability that provides independent control of each processing core that is placed in debug mode and, at a minimum, also provides access to the processor core registers, processor	

	core caches and processor core execution trace.	
2.1.6.f	Shall provide a time distribution capability for time-tagging information for peripheral I/O devices, debug unit, AMBA bus trace buffer, instruction buffers non-intrusively.	Have a time distribution and time tagging facility that allows the user to reconstruct the time ordered events in the chiplet.
2.1.6.h	Shall provide a user selection filter for the execution traces stored in the buffer that can be specified through the debug unit.	Can filter out superfluous execution trace information to focus on the execution thread of interest.
2.1.6.i	Shall provide a capability to store the time-tagged AMBA bus transactions.	Reconstruct the time ordered events of the AMBA bus and determine the bottlenecks, bus utilization statistics and other analyses on the interactions between the IP cores.
2.1.6.k	Shall provide a user selection filter for the AMBA bus transactions stored in the buffer that can be specified through the debug unit.	Can filter out superfluous bus transaction information to focus on the IP core of interest.
2.1.6.l	Shall have all stored data time-tagged to allow time correlation with an external clock or event.	Have a time distribution and time tagging facility that allows the user to reconstruct the time ordered events in the chiplet.
2.1.6.m	Shall provide a capability to store the instruction stream on any of the processing cores.	
2.1.6.n	The low-level hardware debugger shall at least have a breakpoint setting, halt on breakpoint, resume and single step capabilities.	Provide standard debugger capability.
2.1.6.o	Shall provide a debug/trace software tools that runs on a host Linux-based computer workstation.	Support a highly capable software development platform with open source software standards.
2.1.6.p	The debug software tools shall at least have a breakpoint setting, halt on breakpoint, resume and single step capabilities.	Provide standard debugger capability.

7 Environmental Specifications

Identifier	Requirement	Rationale
2.1.7.a	Shall provide TID hardness to at least 1 Mrad (Si).	The chiplet is targeted for missions with a high radiation environment.
2.1.7.b	Shall provide prompt dose rate immunity to 1e10 rad(Si)/s.	
2.1.7.c	Shall provide dose rate survivability to 1e12 rad(Si)/s.	
2.1.7.d	Shall provide latch up immunity to an LET of at least 90 MeV-cm ² /mg.	
2.1.7.e	Shall have no more than 1e-10 uncorrected errors/bit-day, and no more than 1e-4 uncorrected errors/device-day, in Adam's 90% Worst Case GEO environment, behind 100 mils of Al.	Supersedes older 90% WC requirements.
2.1.7.f	Shall have no more than 1e-9 uncorrected errors/bit-minute, and no more than 1e-2 uncorrected errors/device-minute, for the worst 5-minute period of the October 1989 design case flare in CREME 96, behind 100 mils of Al.	
2.1.7.g	Shall operate within MIL-STD junction temperature range of -55 to +125 C.	The chiplet will be integrated in designs with other MIL-STD parts.

8 Reliability and Assured Integrity Specifications

Identifier	Requirement	Rationale
2.1.8.a	Chiplet shall provide a minimum of 100,000 hours of operation before a non-recoverable permanent fault with a 90% probability and 90% confidence level.	The chiplet is targeted for long duration missions with high reliability requirements.
2.1.8.b	A non-recoverable permanent fault in a core shall not affect the other cores on the chiplet.	The chiplet is targeted for long duration missions with high reliability requirements.
2.1.8.c	Chiplet shall provide Assured Integrity (absence of malicious functions/alterations) through either the design flow, the fabrication process, and/or the after fabrication verification process	The chiplet is targeted for defense missions and should be protected from malicious infiltration.

9 Evaluation Board Specifications

Identifier	Requirement	Rationale
2.1.9.a	Shall provide a minimum of two HPSC chiplets.	
2.1.9.b	Shall provide memory to populate all memory ports.	
2.1.9.c	Shall provide a boot ROM device pre-loaded with the boot code to initialize the Evaluation board to a point where OS and user applications can be loaded.	Evaluation board is ready to boot operating systems.
2.1.9.d	Shall provide the capability to fully exercise the power scaling options of the chiplet, and test access to measure current and voltage on each power supply of the chiplet and the board as a whole.	
2.1.9.f	Shall provide connectors or devices suitable for proper observation and usage of each I/O port on the chiplet.	
2.1.9.g	Shall provide host computer device drivers for the I/O interface that connects to the debug and trace port of the chip.	Provide drivers to the JTAG and ARM Debug Interfaces to support the debug software tools on the Linux host computer.
2.1.9.h	Shall provide a layout with sufficient to enable radiation testing of the chiplet.	
2.1.9.i	Shall provide a console interface to the evaluation board.	
2.1.9.j	Shall provide capability to use spare memory that can be substituted into use through software command, in a fault tolerant manner.	For fault tolerance, have a capability to use spare memory to handle the case when a nominal memory bank has degraded.

10 Booting

Identifier	Requirement	Rationale
2.1.10.a	Shall provide external pins to select between the minimum boot sequence and an additional boot sequence.	

2.1.10.b	Shall provide a minimum boot sequence that ensures booting up in a known good state.	A known good state is defined as a user application running on a single core. Needed for early test software development and user application development.
2.1.10.b.1	The minimum boot sequence shall allow user image selection at start up based on external pins.	Define a boot sequence that is the minimum set of IP cores needed to execute a user application on a single core. Needed for early test software development and user application development.
2.1.10.b.2	The minimum boot sequence shall verify the selected user image.	Verify that there is no data corruption in the user image.
2.1.10.b.3	The minimum boot sequence shall have a boot ROM that performs a variety of device health testing prior to application program loading and execution. The following functions should be included in a health check list of devices: DDR memory interface, Flash memory device interface/controller, Peripheral device I/O interfaces.	
2.1.10.c	Shall provide an additional boot sequence that provides complete test coverage of the chiplet.	Define a boot sequence that is the will utilize enables all IP cores needed to execute a different user application on each available core. Needed for parallel and/or multi-core user application development.
2.1.10.c.1	The additional boot sequence shall provide self-test support for each processor core.	
2.1.10.c.2	The additional boot sequence shall identify and disable faulty cores.	

2.1.10.c.3	The additional boot sequence shall store faulty core information in memory for later software access.	
2.1.10.c.4	The additional boot sequence shall verify the selected user image.	Verify that there is no data corruption in the user image.
2.1.10.c.5	The additional boot sequence shall have a boot ROM that performs a variety of device health testing prior to application program loading and execution. The following functions should be included in a health check list of devices: AMBA Bus interconnect, DDR memory interfaces, Flash memory device interfaces/controllers and all peripheral device I/O interfaces utilizing internal loopback.	
2.1.10.d	Support flash scratchpad memory to store data during the boot process.	Mechanism to debug a failed boot sequence.

11 Software Requirements

Identifier	Requirement	Rationale
2.2.a	Shall provide capability to execute different operating systems on different cores.	Does not preclude time and space partitioned operating systems.
2.2.b	Shall support both 32 bit and 64 bit operations.	Demonstrate compatibility with native 64-bit and legacy 32-bit applications.
2.2.c	Shall provide an operating system with the evaluation board that explicitly supports parallel processing, including a C/C++ development tool chain and a parallel debugger.	Enable test and characterization software development for parallel applications.
2.2.d	Shall provide a real-time operating system with the evaluation board that support dynamic resource allocation, including a C/C++ development tool chain and debugger.	Enable test and characterization software development for real-time applications.

2.2.e	For each operating system, shall provide a board support package enabling software access/control for at least the following: external/internal time source, support core-level clock and power control, support error logging with time tags, I/O assignment, interrupt assignment, processor assignment, memory allocation and protection to specific cores, and device drivers for all interfaces.	Supplied operating systems are ready to use and are able to exercise all the functions on the board.
2.2.f	Shall provide a parallel debugger/profiler/tracer with capability to accurately measure/trace each processing core's CPU and cache utilization, measure utilization of shared resources, and perform static dataflow analysis for parallel applications.	Supply a parallel debugger to support parallel application software development.
2.2.g	Shall provide a high fidelity behavior-level or a functional-level simulator (e.g. GEM5 derivative for the chiplet).	Provide a simulation of the hardware to support hardware testing and analysis.
2.2.h	Shall provide the API in the system software for supporting resource allocation, fault tolerance and power management, suitable for use by applications or future middleware.	Supplied operating systems are ready to use and are able to exercise all the functions on the board.

Acronym List

Term	Description
ALU	Arithmetic-Logic Unit
API	Application Program Interface
ASIC	Application-Specific Integrated Circuit
COTS	Commercial Off the Shelf
CPU	Central Processing Unit
DDR	Double Data Rate
EEPROM	Electrically Erasable Programmable Read-Only Memory
FLOPS	Floating-Point Operations per Second
FPGA	Field Programmable Gate Array
GOPS	Giga Operations per Second
GP	General Purpose
GPIO	General Purpose Input/Output
GPR	General Purpose Register
GPU	Graphic Processing Unit
HPSC	High Performance Space Computing
IEEE	Institute of Electrical and Electronics Engineers
IP	Intellectual Property
JTAG	Joint Test Action Group
LET	Linear Energy Transfer
MCM	Multi-Chip Module
MET	Mission Elapse Timer
MIPS	Million Instructions per Second
NASA	National Aeronautics and Space Administration
OS	Operating System
PoR	Power On Reset
RHBD	Radiation Hard by Design
RAM	Random Access Memory
ROM	Read-Only Memory
SCT	Spacecraft Timer
SEE	Single Event Effect
SEFI	Single Event Functional Interrupt
SOW	Statement of Work
SRIO	Serial RapidIO
TBD	To Be Determined
TID	Total Ionizing Dose