SpaceCubeX: A Path Towards Hybrid On-Board Processing Architectures

Matthew French and Andrew Schmidt – USC/ ISI
Thomas Flatley – GSFC
Carlos Villalpando – JPL
June 23, 2015
Motivation: NASA Earth Science Missions

- Instruments produce essential data to help scientists answer critical 21st century questions
  - Global climate change, air quality, ocean health, ecosystem dynamics, etc...
- On-board processing ~100-1,000x than previous missions (compression, storage, downlink)
  - Current/near-term data at rates \( >10^8 \) to \( 10^{11} \) bits/second
- Adding new capabilities such as low-latency data products for extreme event warnings
- Missions specifying instruments with significantly increased:
  - Temporal, spatial, and frequency resolutions \( \rightarrow \) to global, continuous observations
- Meeting inter-mission reusability goals

Hybrid architecture is a key cross-cutting technology directly applicable to missions recommended in the Decadal Survey
SpaceCubeX Project

• Develop high performance on-board computing capability for flight missions:
  – Leverage heterogeneous processor types (COTS → Rad-Hard)
  – Incorporate industry experience with commercial device specialization
  – Improve computation/energy efficiency
  – Support persistent, long-term mission operations
  – Provide potential infusion into (P)ACE, HyspIRI, GEO-CAPE, ICESat-2...

• Address key characteristics to support mission design:
  – Size, weight, area, power
  – Mission capabilities, cost, risk
  – Risks of supply chain disruption
  – Benchmark compilation for robust characterization
  – Commercial tools/compilers interoperability
SpaceCube Family

**v1.0**
- 2009: STS-125
- 2009: MISSE-7
- 2013: STP-H4
- 2016: STP-H5

**v1.5**
- 2012: SMART

**v2.0-EM**
- 2013: STP-H4
- 2016: STP-H5

**v2.0-FLT**
- 2015: GPS Demo
  - Robotic Servicing
  - Numerous proposals for Earth/Space/Helio
SpaceCube Next?

- **Multi-core processors more easily provide:**
  - General OS support
  - High-level functions
  - Coarse grained application parallelism

- **Co-processors then provide:**
  - Customized acceleration
  - High throughput
  - Fine-grained data parallelism

- **Sparked questions:**
  - What devices to use?
  - How to connect the devices?
  - How to program the system?
  - (and many more...)
## Hybrid Architecture Device Candidates

<table>
<thead>
<tr>
<th>Category</th>
<th>Candidates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-core</td>
<td>Boeing Maestro, Tilera Tile64, Aeroflex LEON4, ...</td>
</tr>
<tr>
<td>FPGA</td>
<td>Xilinx Virtex7, Altera StratixV, Microsemi, Achronix S22i</td>
</tr>
<tr>
<td>DSP</td>
<td>BAE RADSPEED, ClearSpeed CSX700, TI 320C6701 ...</td>
</tr>
<tr>
<td>Memory</td>
<td>Atmel M65609E, Honeywell HXSR06432, ...</td>
</tr>
<tr>
<td>Interconnects</td>
<td>PCIe (Gen2/3), XAUI, RapidIO, SpaceWire ...</td>
</tr>
<tr>
<td>Topologies</td>
<td>Bus, Mesh, Ring, Crossbar, Butterfly ...</td>
</tr>
<tr>
<td>Sensors I/O</td>
<td>RS-422, Ethernet, SpaceWire Direct I/O, I²C ...</td>
</tr>
</tbody>
</table>
Traditional Hardware Development: Specifications → Manual Component Analysis → Topology & Netlist Creation → Hardware Implementation → Performance Results

While number of processors available to aerospace industry is tractable, number of combinations of these elements and number of interconnect topology permutations makes this process difficult and error prone for a human to do manually.
SpaceCubeX Approach

- Develop avionics architecture and generator suite:
  - Candidate device analysis
  - Evaluate with benchmark applications
- Profile architectures in simulation environment and select leading candidates for emulation testbed
- Implement architecture with selected hardware in emulation testbed
- Evaluate implementation
- Assess results and lessons learned
SpaceCubeX Development Process

- Updated to support hybrid architecture with focused design space exploration
- Utilize generation and analysis tools to eliminate manual fit analysis
  - Incorporate micro & complex benchmarks throughout testbed evaluation process
- Common infrastructure for code reuse (simulation → flight hardware)
  - Explore and compare several permutations in rapid succession

SpaceCubeX Heterogeneous Hardware Development:

Specifications

Designer Inputs

Applications & Benchmarks

Evolvable Testbed

Flight Hardware

Engineering Hardware

Hardware Emulation

Simulation

Compilation Environment

Evolvable Testbed

Performance Results

TRL

Design Exploration Feedback Loop

Specifications

Designer Inputs

Applications & Benchmarks

Evolvable Testbed

Flight Hardware

Engineering Hardware

Hardware Emulation

Simulation

Compilation Environment

Evolvable Testbed

Performance Results

TRL

Design Exploration Feedback Loop
SpaceCubeX extends traditional simulation to incorporate entire hybrid system
**Emulation Testbed**

**Benchmarks Suite**
- Microkernels
- Applications
- System

**Compile & Partition**
- Board Model (xml)

**Emulation Generator**

**Emulation Environment:**
- Multi-core Board
- FPGA Board
- DSP Board
- Memory
- Emulation Generator

**Performance Models**

**Emulator**
- Interface
- Hardware (FPGA, DSP, MPSoC)
- Software (Emulation, Modeling, Monitoring)

**SpaceCubeX** emphasizes end-to-end board level architecture emulation

**Report**
- Power
- Area
- Radiation
- Performance
REconfigurable Data-stream Hardware/Software ARChitecture:

- Provides rapid construction of Multi-System-on-Chip platforms (HW & SW)
- Enables design space exploration of heterogeneous resources (Dev. Spirals)
- System infrastructure supplied as IP, raising focus to algorithm development
- Combines situational awareness with high performance processing
  - On-chip run-time control
  - Dynamic resource management
  - System and kernel-level performance tuning
- Redsharc is not a high-level synthesis tool
  - Supports use of HLS tools use to create HW cores
  - Fills gaps in HLS tools: Integration, communication, and run-time management

Redsharc Philosophy: Interoperate and leverage current and emerging devices, tools, and technology to make developers more productive
Redsharc Development Spectrum

Accelerating Development Maturity

Host PC: SW  FPGA: SW  FPGA: HW  FPGA: HW  FPGA: HW
On-Chip     Kernel Simulation   System Simulation   System Run-time

Towards Full-System Integration

Redsharc infrastructure supports rapid testing across a variety of development environments to allow users to leverage preferred design and debugging tools such as Eclipse, GDB, ModelSim, and ChipScope.
Redsharc APIs

• Application Programming Interfaces:
  – Software Kernel Interface
  – Hardware Kernel Interface
  – Dataflow Graph (DFG)
  – System
  – Scheduling

• Run-time Execution
  – Control Kernel Operation

End-user leverages higher levels of abstraction to implement application on heterogeneous resources
Redsharc Generate flow constructs finished system from user provided design files
Software Compilation Flow

Applications & Benchmarks

Manual Task Partitioning

Multi-core Compiler

Accelerator Compiler

FPGA Synthesis

Hardware/Software Co-Design Library:
- Communications
- Interfaces
- Protocols
- Run-time controls

Redsharc Libraries

FPGA Resource Library:
- Interfaces
- Interconnects
- Accelerator Cores

Evolvable Testbed

Information Sciences Institute
# List of Benchmarks

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Micro Benchmarks</strong></td>
<td>Kernels to benchmark architecture subcomponents and measure system viability.</td>
</tr>
<tr>
<td><strong>ALHAT</strong></td>
<td>Analyze landing sites; Image processing (interpolation, correlation, kalman filter), 4 sensors, task/data parallelism.</td>
</tr>
<tr>
<td><strong>ASPEN</strong></td>
<td>Artificial intelligence, multi-threaded, low level scheduler to satisfy constrained mission goals.</td>
</tr>
<tr>
<td><strong>ROCKSTER</strong></td>
<td>Autonomous spacecraft tasking, geological feature identification, analysis, and data handling.</td>
</tr>
<tr>
<td><strong>ACE Mission Scenario</strong></td>
<td>Global, continuous real-time ocean radiometer for carbon assessment enabling research and coordinated activities based on live ocean color data; multi-spectral image, data parallel processing</td>
</tr>
<tr>
<td><strong>Radar/Lidar Mission Scenario</strong></td>
<td>Synthetic aperture radar and multi-beam lidar instruments produce voluminous raw data (future data rate concepts &gt;$10^{12}$ bits/second); benchmarks perform terrain, vegetation and/or glacial ice mapping; high data rate processing, data parallelism.</td>
</tr>
<tr>
<td><strong>HyspIRI Mission Scenario</strong></td>
<td>Hyper-spectral Earth science event detection algorithms (fire, flood, coastal spills); Autonomously reconfigures for Visible to Short Wavelength Infrared data for real-time mapping and transmission.</td>
</tr>
</tbody>
</table>
### Result Metrics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power</td>
<td>Watts</td>
</tr>
<tr>
<td>Estimated board size</td>
<td>cm²</td>
</tr>
<tr>
<td>Estimated board weight</td>
<td>kg</td>
</tr>
<tr>
<td>Processing performance</td>
<td>Instructions or Operations per second</td>
</tr>
<tr>
<td>Input Output Bandwidth</td>
<td>Bits per second</td>
</tr>
<tr>
<td>Radiation Tolerance</td>
<td>Total Ionizing Dose and Single Event Upset rate</td>
</tr>
<tr>
<td>Cost</td>
<td>$</td>
</tr>
</tbody>
</table>

*Provide designers and scientists more feedback earlier in development process*
Example Results

Performance unit (Area, Power, Processing...)

- Architecture 1
- Architecture 2
- Architecture 3

Sequential
Streaming
Server
Bit-level

Single CPU baseline
Getting “On-Board”

• We want to learn from the community
  – Understand applications and instruments
  – Advice on application/algorithm implementations
  – Suggestions on devices to consider

• How is the current technology performing?
• Hard restrictions on data precision?
• Application’s demand for radiation tolerance?
SpaceCubeX aims to:

- Alleviate computation limitations for on-board processing
- Enable selection of SWAP efficient processing architecture:
  - Impact on mission capabilities, cost, and risk
- Reduce risk due to supply chain disruption
- Require minimal extensions to integrate new devices
- Support persistent, long-term mission operations
- Provide potential infusion into many missions:
  - (P)ACE, HyspIRI, GEO-CAPE, ICESat-2...

The SpaceCubeX Project is a structured approach to assess and develop hybrid architectures, fundamentally changing avionics processing architecture development process.
QUESTIONS
BACKUP SLIDES
Project Milestones

- Project start 05/15
- Complete initial scenario benchmarks 11/15
- Initial architecture 12/15
- Final architecture 03/16
- Final benchmark package 04/16
- Initial emulation testbed 12/16
- Final emulation testbed 03/17
- Benchmark demonstration 04/17
Development Abstractions

Hardware Kernel Interface

Software Kernel Interface

Data accesses are abstracted from the kernel implementation

Function Name | Arguments | Description
--- | --- | ---
streamPush | element *e onto stream *s | Pushes element e onto stream s
streamPop | element *e | Pops the top element from stream s and stores the value in e
streamPeek | | Reads the top element from stream s and stores the value in e
blockWrite | element *e | Writes element e into block b at index i
blockRead | int index block *b | Reads and element from block b at index i and stores the value in e
## TRL Assessment

<table>
<thead>
<tr>
<th>Category</th>
<th>Component</th>
<th>Initial</th>
<th>6 Mos</th>
<th>12 Mos</th>
<th>18 Mos</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Architecture</td>
<td>Component Selection</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Interconnect Topology</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Control Software</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Hardware Architecture Subtotal</td>
<td></td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Programming Model</td>
<td></td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>3</strong></td>
<td><strong>3</strong></td>
<td><strong>4</strong></td>
<td><strong>4</strong></td>
<td><strong>5</strong></td>
</tr>
</tbody>
</table>