SpaceCubeX: A Path Towards Hybrid On-Board Processing Architectures

Matthew French and Andrew Schmidt – USC/ ISI

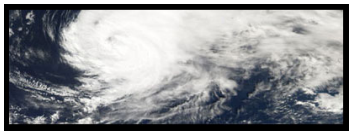
Thomas Flatley – GSFC

Carlos Villalpando – JPL

June 23, 2015

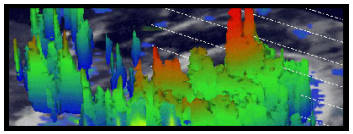
Motivation: NASA Earth Science Missions

- **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 → to global, continuous observations
- **Meeting inter-mission reusability goals**



ACE

GEO-CAPE



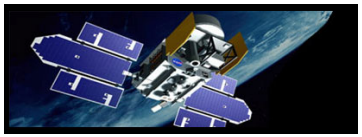
3D-Winds

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

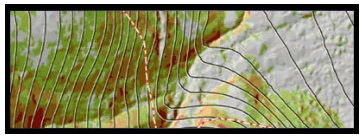
ASCENDS

HyspIRI

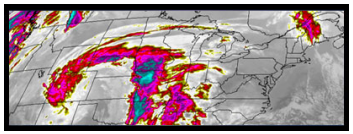
PACE



ICESat II



LIST



PATH

ESTO Earth Science Technology Office

SpaceCubeX Project

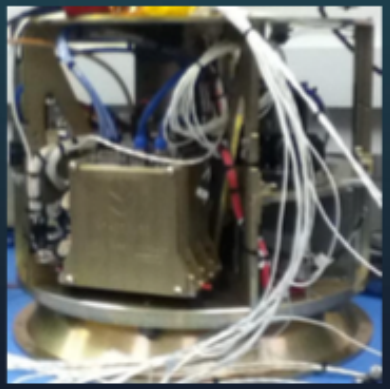
- **Develop high performance on-board computing capability for flight missions:**
  - Leverage heterogeneous processor types (COTS → Rad-Hard)
  - Incorporate industry experience with commercial device specialization
  - Improve computation/energy efficiency
  - Support persistent, long-term mission 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

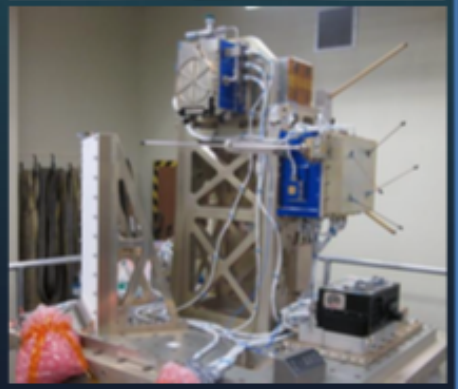
2009 STS-125
2009 MISSE-7
2013 STP-H4
2016 STP-H5

v1.5



2012 SMART

v2.0-EM



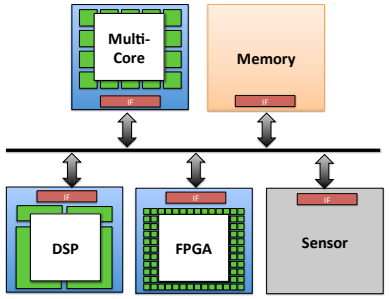
2013 STP-H4
2016 STP-H5

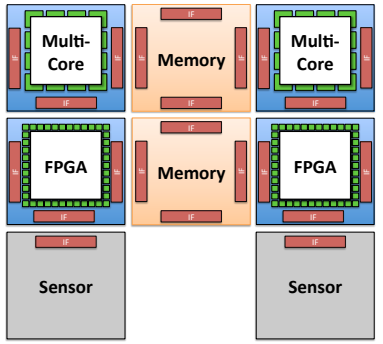
v2.0-FLT

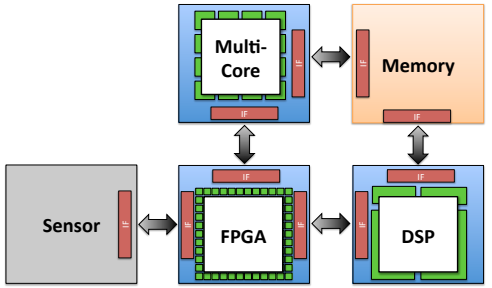
2015 GPS Demo
- Robotic Servicing
- Numerous proposals for Earth/Space/Helio

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…)

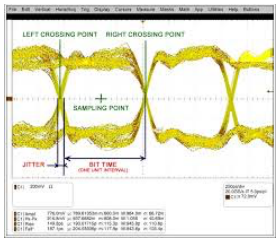


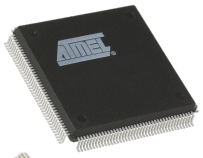


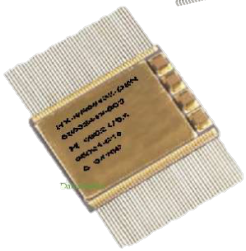


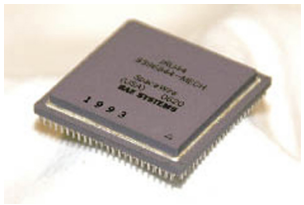
Hybrid Architecture Device Candidates

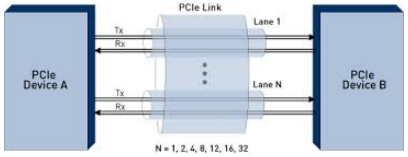
Category	Candidates
Multi-core	Boeing Maestro, Tilera Tile64, Aeroflex LEON4, ...
FPGA	Xilinx Virtex7, Altera StratixV, Microsemi, Achronix S22i
DSP	BAE RADSPEED, ClearSpeed CSX700, TI 320C6701 ...
Memory	Atmel M65609E, Honeywell HXSR06432, ...
Interconnects	PCIe (Gen2/3), XAUI, RapidIO, SpaceWire ...
Topologies	Bus, Mesh, Ring, Crossbar, Butterfly ...
Sensors I/O	RS-422, Ethernet, SpaceWire Direct I/O, I^2C ...











Microsemi

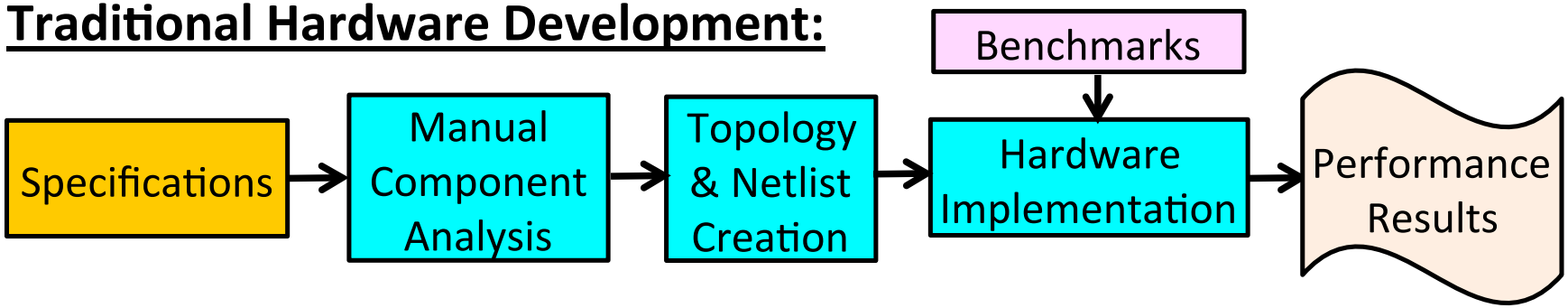
ALTERA

freescale

ESTO
Earth Science Technology Office

Traditional Development Process

Traditional Hardware Development:



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.

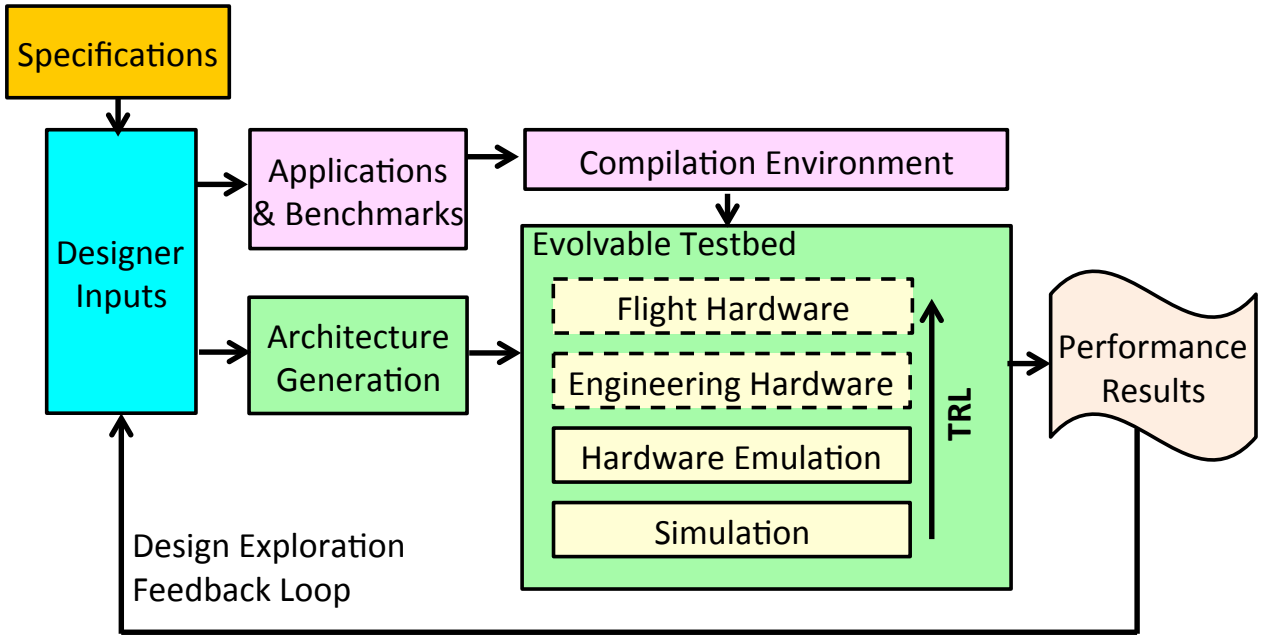
SpaceCubeX Approach

- **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**

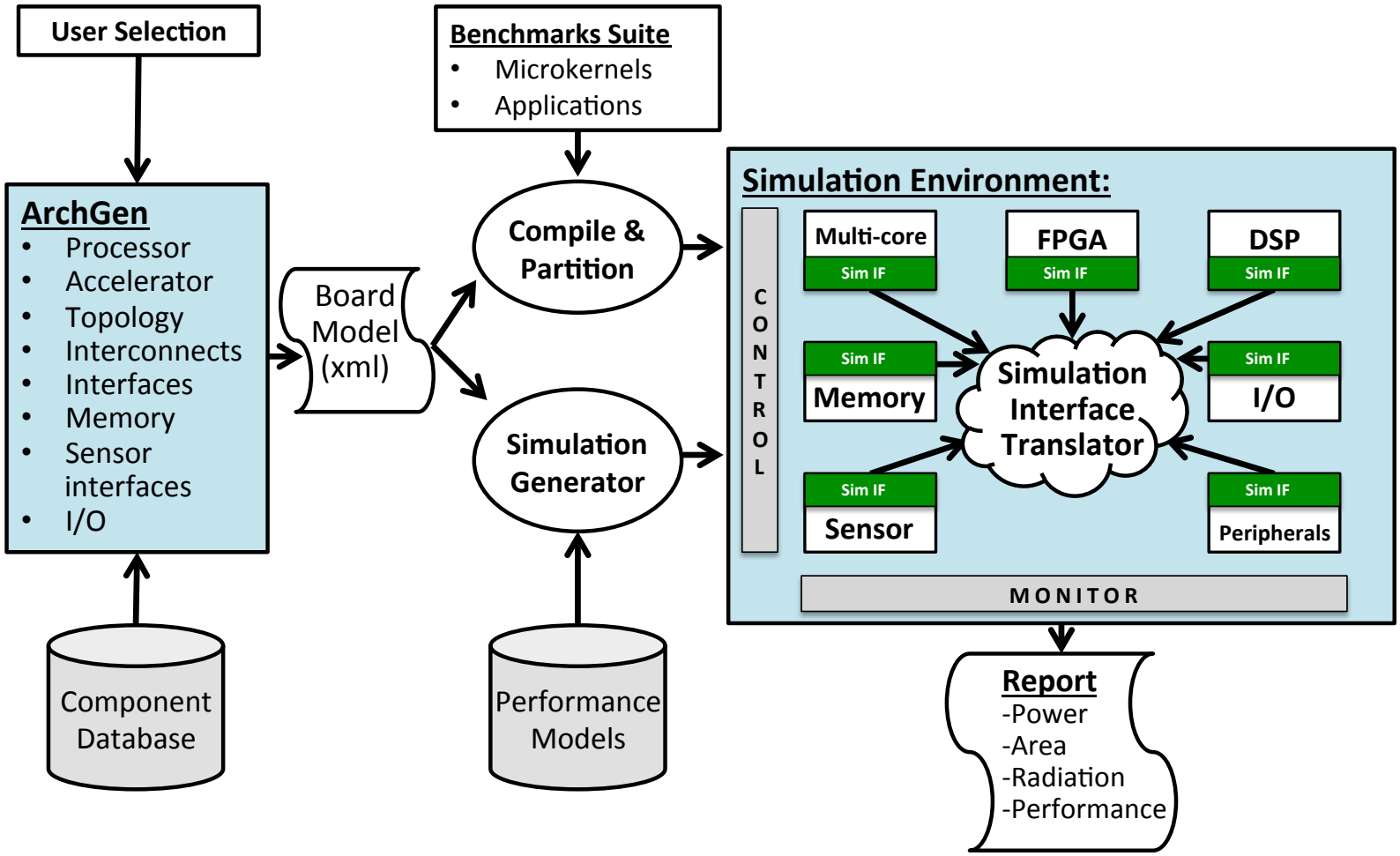
SpaceCubeX Development Process

- **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:

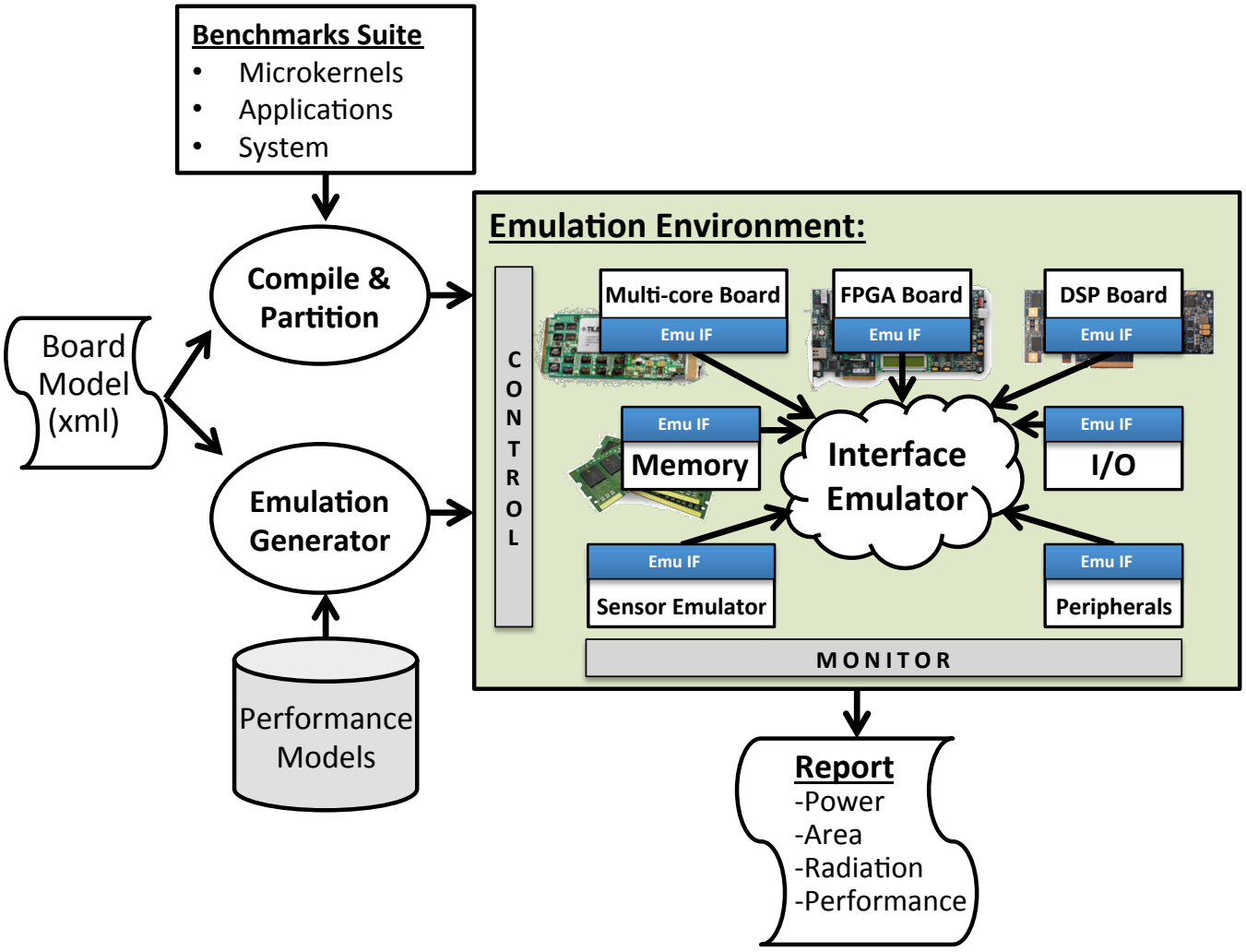


Simulation Testbed



SpaceCubeX extends traditional simulation to incorporate entire hybrid system

Emulation Testbed



SpaceCubeX emphasizes end-to-end board level architecture emulation

SpaceCubeX - Redsharc

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**

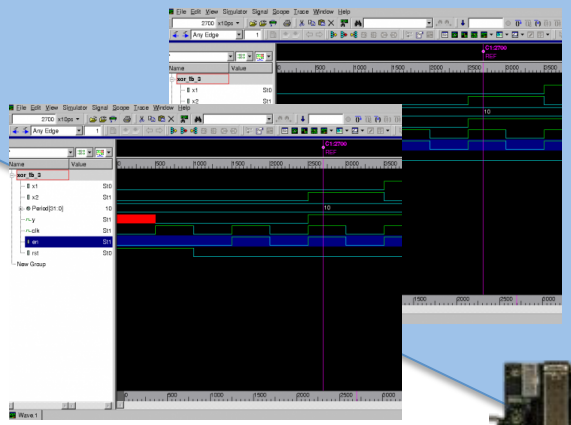
Redsharc Philosophy: Interoperate and leverage current and emerging devices, tools, and technology to make developers more productive

ESTO Earth Science Technology Office

Redsharc Development Spectrum

Accelerating Development Maturity

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



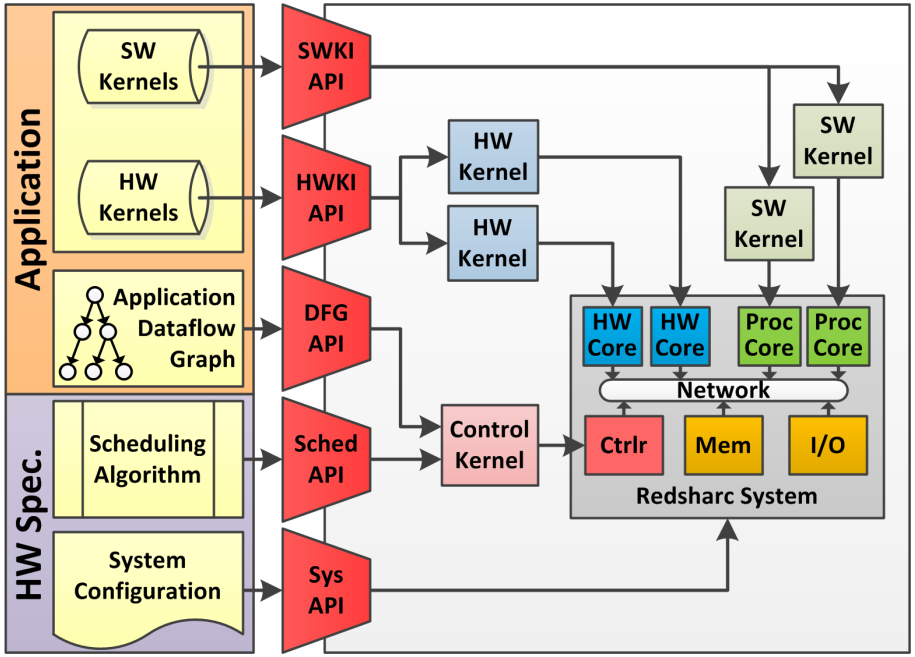
Towards Full-System Integration

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.

ESTO
Earth Science Technology Office

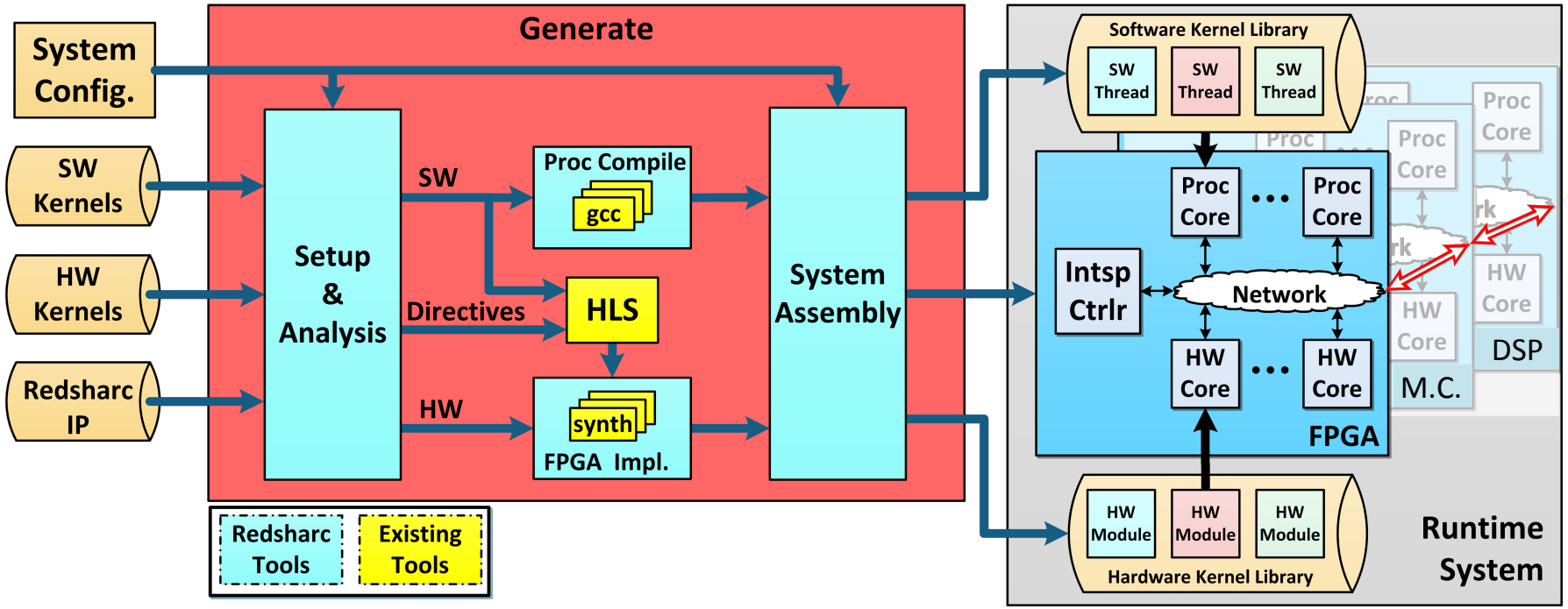
Redsharc APIs

- **Application Programming Interfaces:**
  - Software Kernel Interface
  - Hardware Kernel Interface
  - Dataflow Graph (DFG)
  - System
  - Scheduling
- **Run-time Execution**
  - Control Kernel Operation



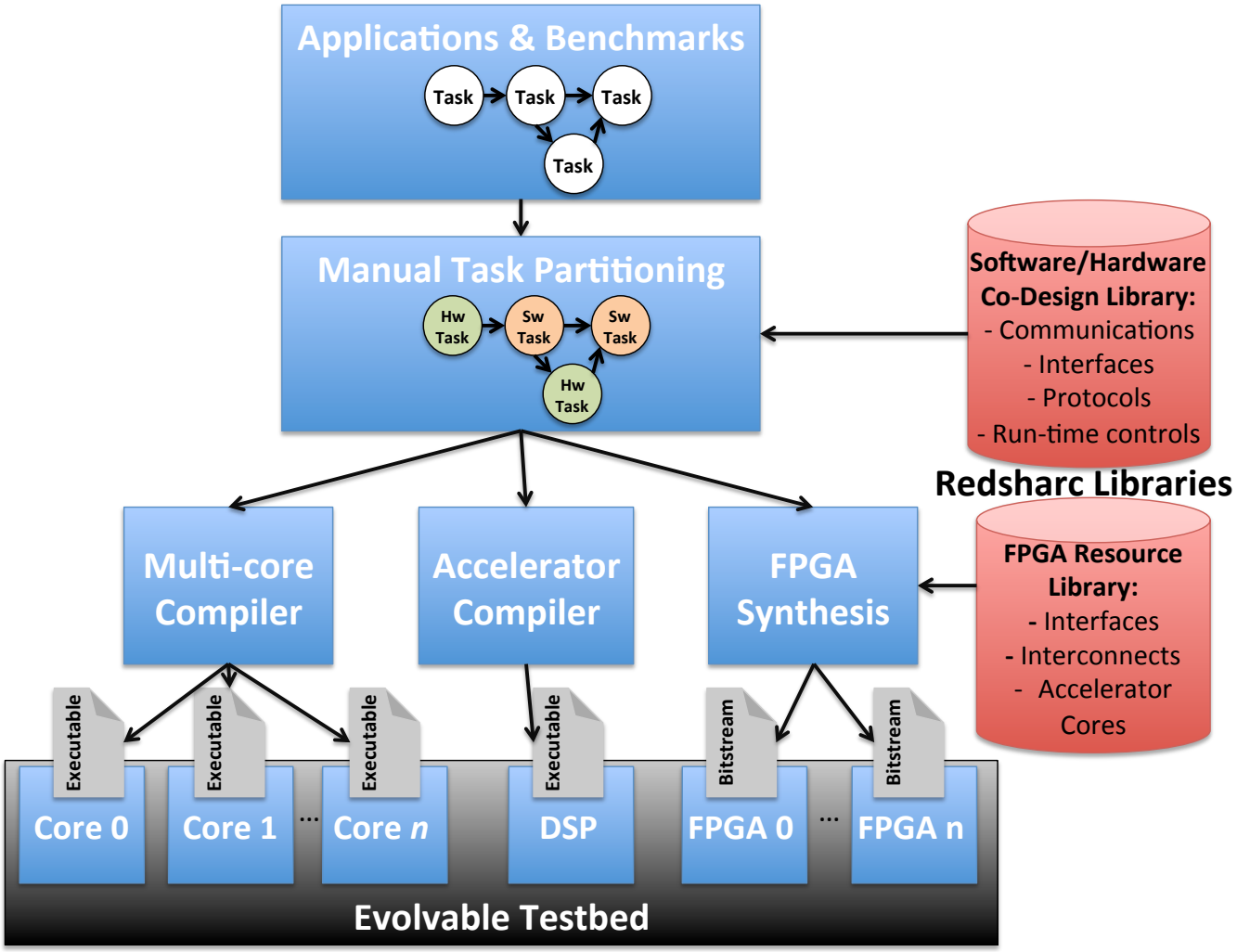
End-user leverages higher levels of abstraction to implement application on heterogeneous resources

Redsharc Build Environment



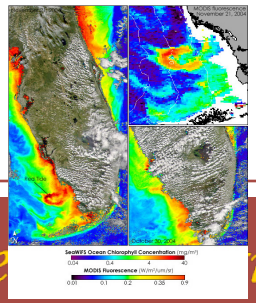
Redsharc Generate flow constructs finished system from user provided design files

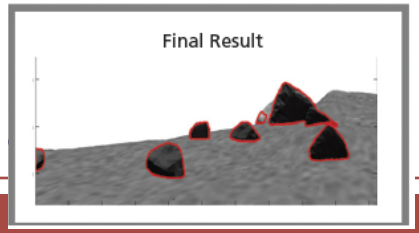
Software Compilation Flow

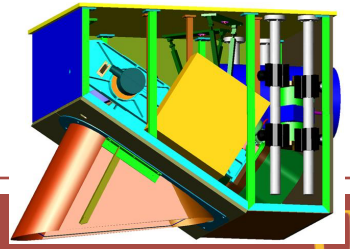


List of Benchmarks

Benchmark	Description
Micro Benchmarks	Kernels to benchmark architecture subcomponents and measure system viability.
ALHAT Autonomous Landing and Hazard Avoidance and Detection System	Analyze landing sites; Image processing (interpolation, correlation, kalman filter), 4 sensors, task/data parallelism.
ASPEN Automated Scheduling and Planning Environment	Artificial intelligence, multi-threaded, low level scheduler to satisfy constrained mission goals.
ROCKSTER Rock Segmentation Through Edge Regrouping	Autonomous spacecraft tasking, geological feature identification, analysis, and data handling.
ACE Mission Scenario Aerosol-Cloud-Ecosystem	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
Radar/Lidar Mission Scenario	Synthetic aperture radar and multi-beam lidar instruments produce voluminous raw data (future data rate concepts >10^{12} bits/second); benchmarks perform terrain, vegetation and/or glacial ice mapping; high data rate processing, data parallelism.
HysplRI Mission Scenario	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.







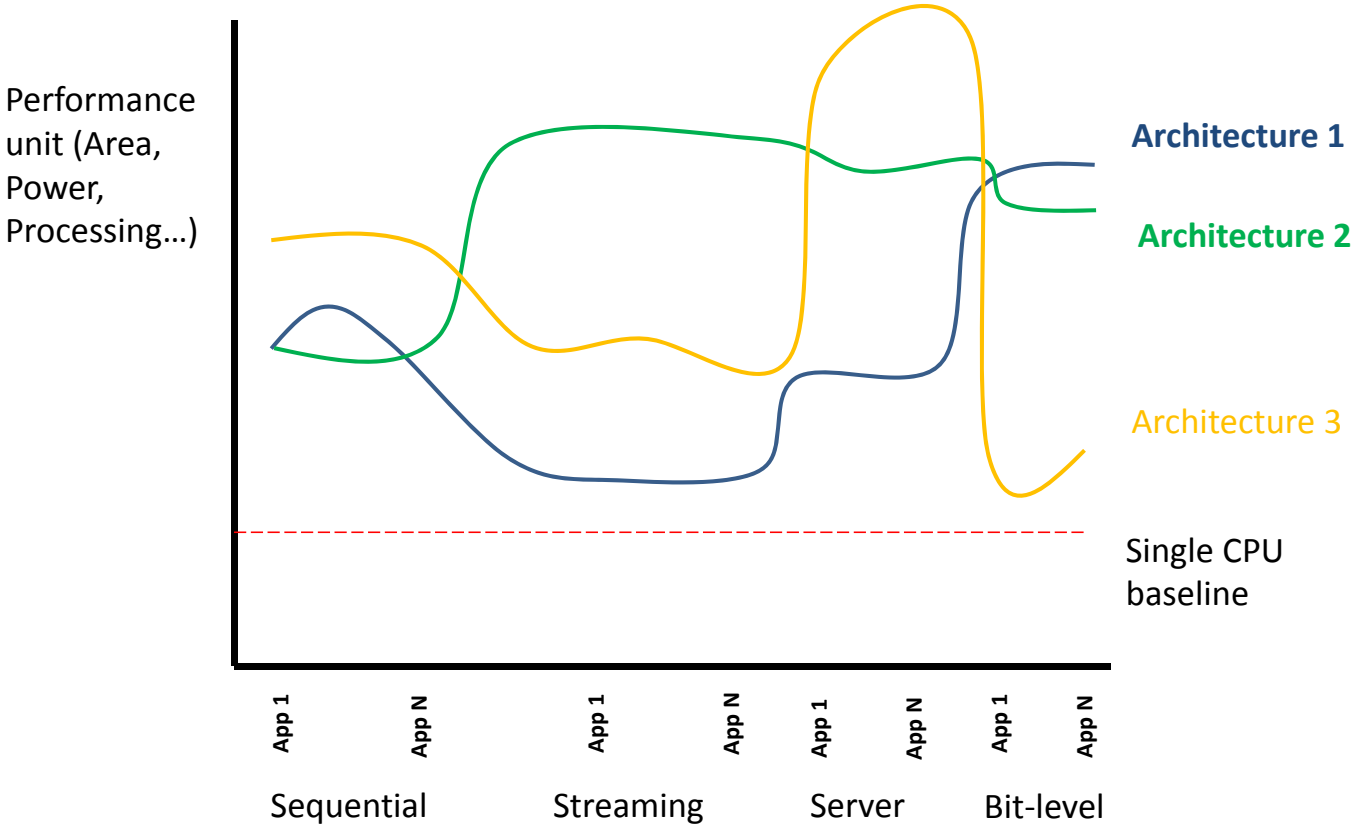
ESTO
Earth Science Technology Office

Result Metrics

Parameter	Units
Power	Watts
Estimated board size	cm^2
Estimated board weight	kg
Processing performance	Instructions or Operations per second
Input Output Bandwidth	Bits per second
Radiation Tolerance	Total Ionizing Dose and Single Event Upset rate
Cost	$

Provide designers and scientists more feedback earlier in development process

Example Results



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?**

Summary

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

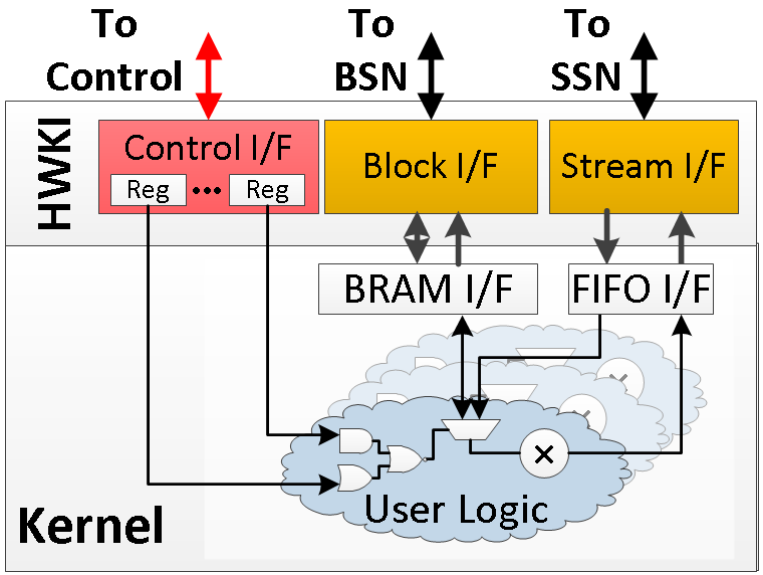
BACKUP SLIDES

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

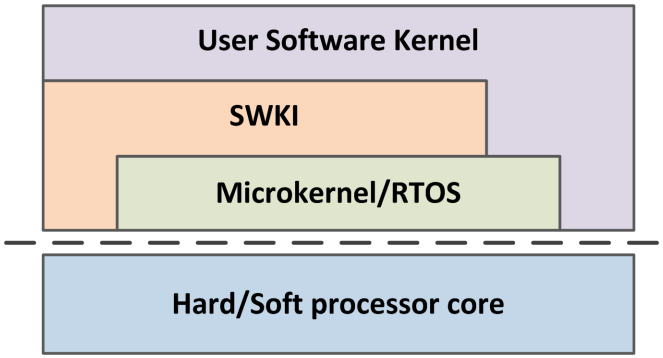
Development Abstractions

Hardware Kernel Interface



Software Kernel Interface

Function Name	Arguments	Description
streamPush	element *e stream *s	Pushes element e onto stream s
streamPop		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 int index block *b	Writes element e into block b at index i
blockRead		Reads and element from block b at index i and stores the value in e



Data accesses are abstracted from the kernel implementation

TRL Assessment

Category	Component	Initial	6 Mos	12 Mos	18 Mos	Final
Hardware Architecture	Component Selection	3	3	4	4	5
	Interconnect Topology	3	3	4	4	5
	Control Software	3	3	4	4	5
Hardware Architecture Subtotal		3	3	4	4	5
Programming Model		4	5	5	5	5
Total		**3**	**3**	**4**	**4**	**5**