SpaceCubeX: Hybrid Multi-core CPU / FPGA / DSP Flight Architecture

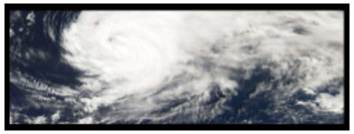
Matthew French, Andrew Schmidt, Gabe Weisz – USC / ISI
Tom Flatley, Gary Crum, Jonathan Bobblit – NASA GSFC
Carlos Villalpando, Robert Bocchino – NASA JPL

June 14th, 2016

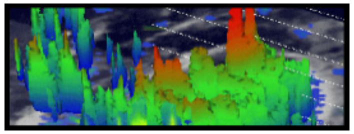
Earth Science Technology Office

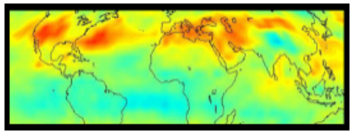
Motivation: Next Gen NASA Earth Science Missions

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

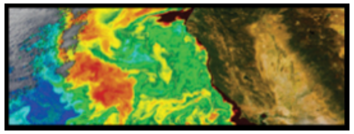

ACE

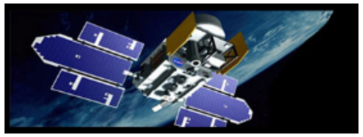
GEO-CAPE

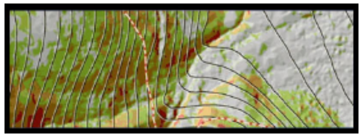

3D-Winds

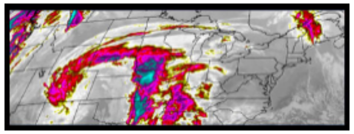

ASCENDS

HyspIRI


PACE


TSIS-1


LIST


PATH

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

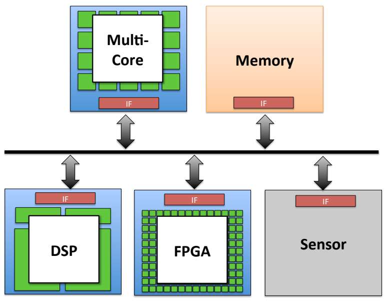
SpaceCubeX Project

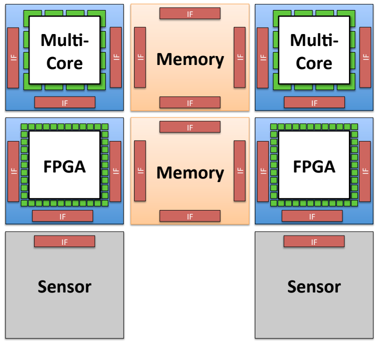
- Recent publications show hybrid computing architectures can be highly efficient in highly constrained SWAP scenarios
  - Also tracks trends in HPC, mobile computing
- Can we identify a hybrid computing architecture of high utility across a wide suite of Earth Science applications?

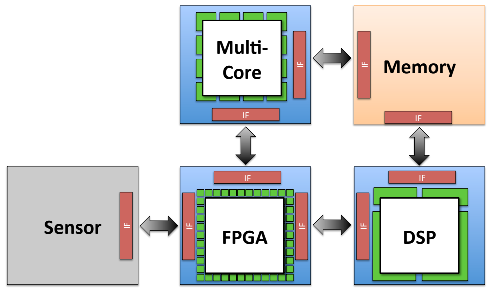
- SpaceCubeX Architecture Analysis Framework:
  - Enables selection of the most SWAP efficient processing architecture, with direct impact on mission capabilities, cost, and risk.
  - Minimal extensions to integrate new processors, such as the anticipated NASA High Performance Spaceflight Computer (HPSC), reducing time to initiate benchmarking by months.
  - Reduces risk due to supply chain disruptions by allowing a user to rapidly compare alternative component selections, quantify the impact, and update the processing architecture.
  - Leverages a wide suite of high performance embedded computing benchmarks and Earth science scenarios to ensure robust architecture characterization.
  - Utilizes a proven inter-task programming model to facilitate benchmark compilation and experimentation, while being fully interoperable with commercial compilers.

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…)

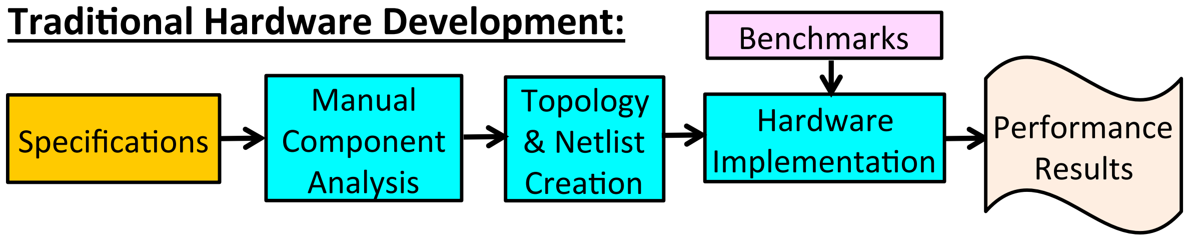


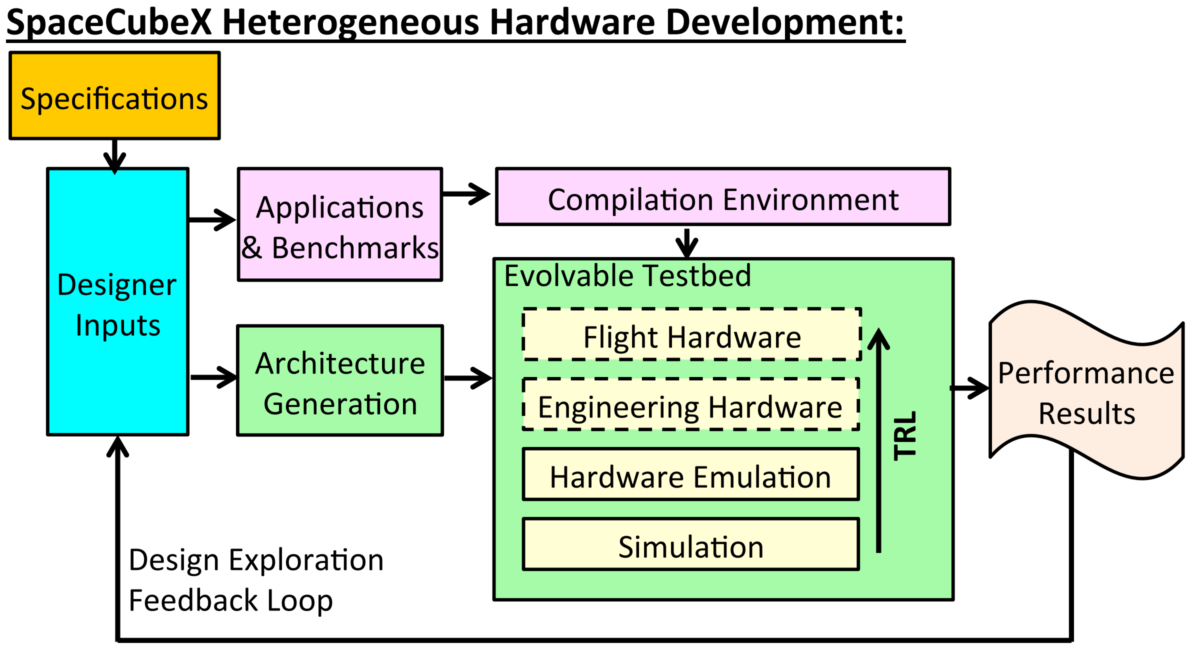




Reinvent Hardware Development Process

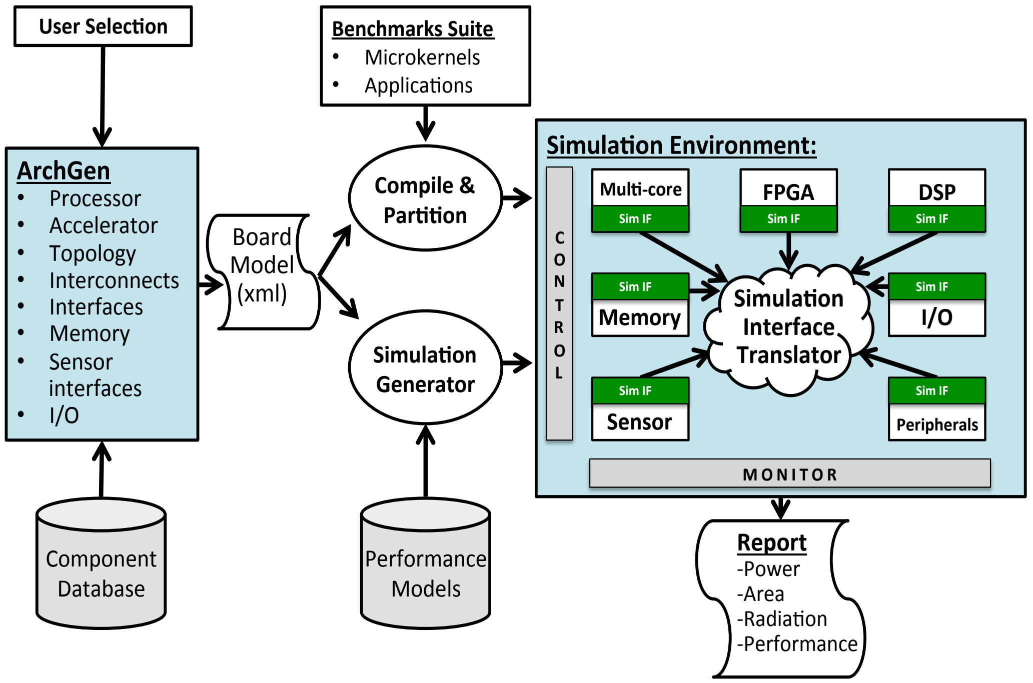
- Search space of hybrid combinations too complex for existing methods
- Create a development environment which enables design space exploration of architectures across a range of applications
- Components pulled from a database and architecture generated
- Same environment can be used as architecture increases TRL
- Performance can be tracked across architectures and across TRL development


Traditional Hardware Development:
Specifications
Manual Component Analysis
Topology & Netlist Creation
Benchmarks
Hardware Implementation
Performance Results



Simulation Testbed

- ArchGen:
  - Creates board models for evaluation in simulation/emulation environments
- Component Database:
  - Collection of components of interest for evaluation
- Performance Models:
  - Individual component parameters used by simulation environment and report generation
- Simulation Generator:
  - Builds simulation environment based on board model
- Compiler:
  - Sets up compilation environment based on board model
- Simulation Environment:
  - Run-time framework to evaluate benchmark applications on hybrid architecture model
- Report:
  - Details of system's metrics based on benchmark applications on candidate architecture model

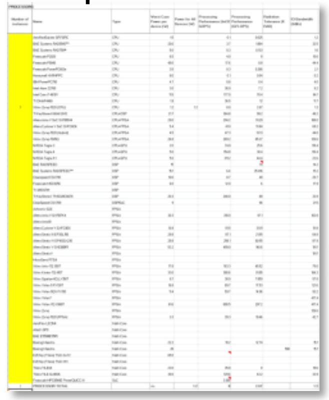


Year 1: Development and evaluation of simulation testbed across hybrid architectures

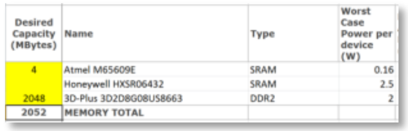
Architecture Generator (ArchGen)

- Python based script for rapid architecture prototyping
- Parse Excel spreadsheets for component database and user selections
- Modular capabilities for future and integration with Emulation Environment
- Added hooks for GUI front-end for User Selection via TkInter
- Produces XML-based Board Model file for:
  - Use by Simulation Generator/Compilation & Emulation Framework

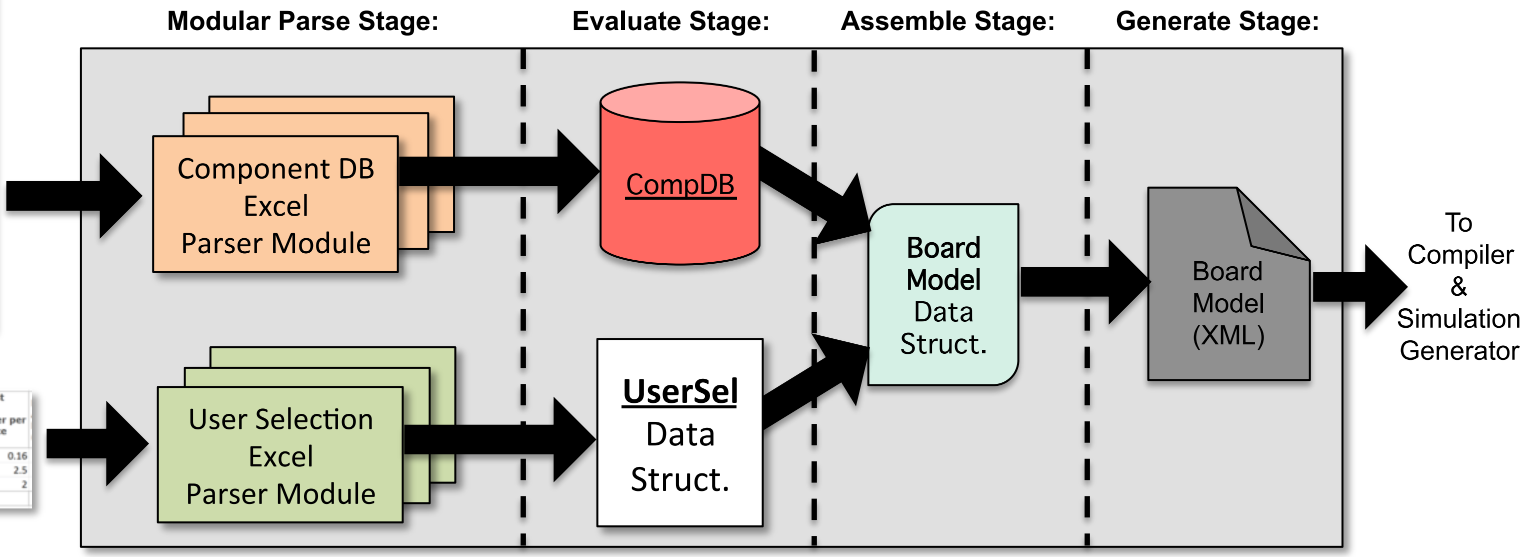
Component Database



User Selection



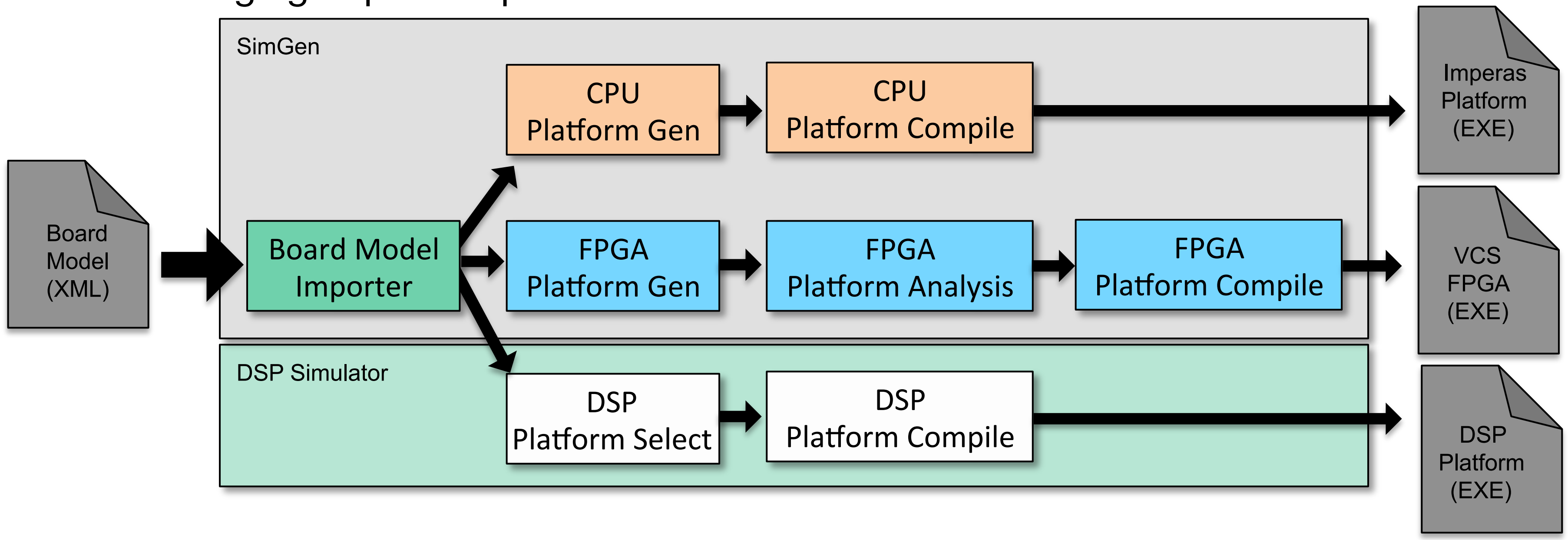
Desired Capacity (MBytes)	Name	Type	Worst Case Power per device (W)
4	Atmel M65609E	SRAM	0.16
	Honeywell HXSR06432	SRAM	2.5
2048	3D-Plus 3D2D8G08US8663	DDR2	2
2052	MEMORY TOTAL		



```
$ ArchGen CompDB.xlsx UserSel.xls HybrdFPGA HybridFPGA_1.xml
```

Simulation Generator (SimGen)

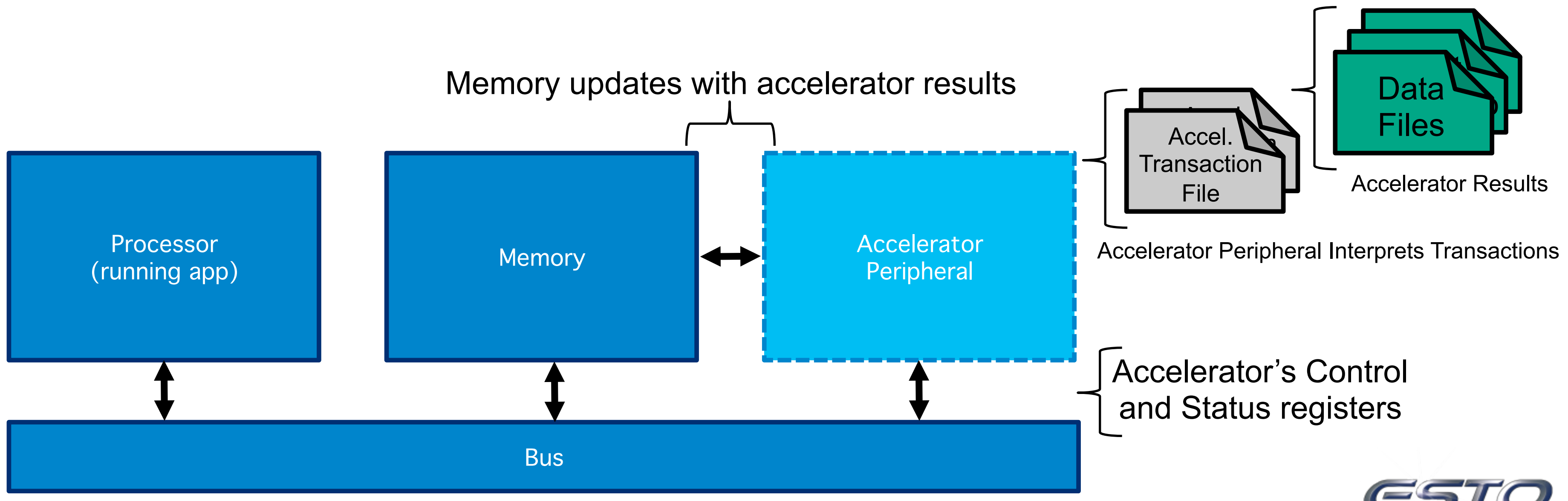
- Python based script to stitch together hybrid simulation
- Generate Simulation Environment from board model
  - Interoperability between multi-core processors, FPGAs, DSPs
  - Run-time platform to execute benchmark applications
  - Monitor and collect performance report for post-run analysis
- Added support for external co-simulator (e.g. DSP/FPGAs)
- Leveraging Imperas Open Virtual Platform Platform APIs and Libraries



Simulation Generation create platform executables for CPUs and FPGAs. DSP simulator already incorporates DSP simulator platform executable.

SpaceCubeX's Accelerator Peripheral Simulation Engine

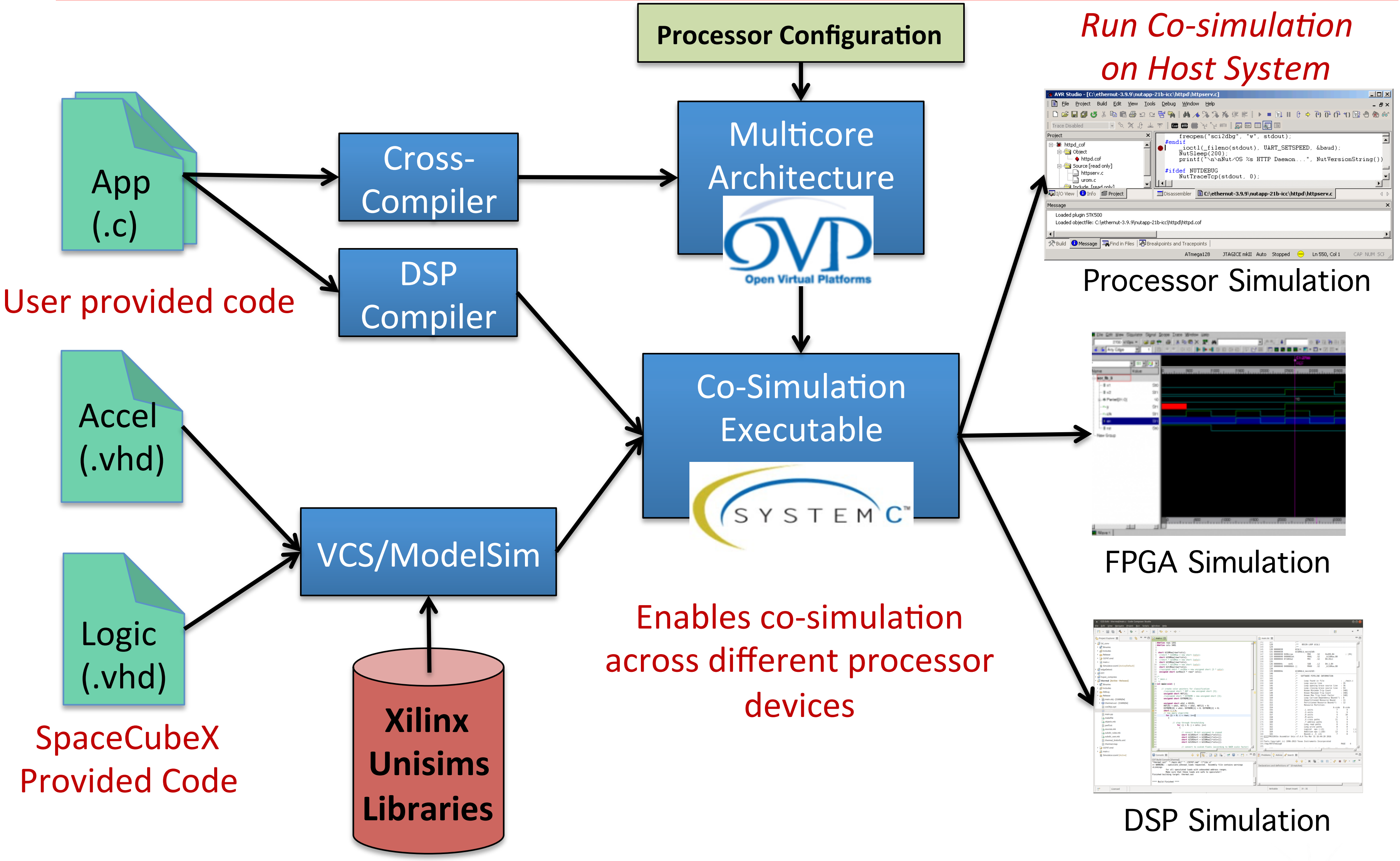
- Connects processor simulator with FPGAs and DSPs co-simulation environments
- Motivated by limited DSP simulation SystemC & TLM integration support
- Recognized need for similar functionality with future Emulation Environment
- Accelerator peripheral is transaction-based
  - Supports multiple transactions to processor's memory
  - Working on adding more fine-grain delays
    - Accelerator Delay – Time for accelerator to perform computation
    - Interface Delay – Time for accelerator to transfer data to memory/CPU
- Processor issues commands / checks status via Bus transactions
- Integrates with Redsharc FPGA simulations and DSP TI simulator



Memory updates with accelerator results
Data Files
Accel. Transaction File
Accelerator Results
Accelerator Peripheral Interprets Transactions
Processor (running app)
Memory
Accelerator Peripheral
Accelerator's Control and Status registers
Bus

9

Simulation Environment

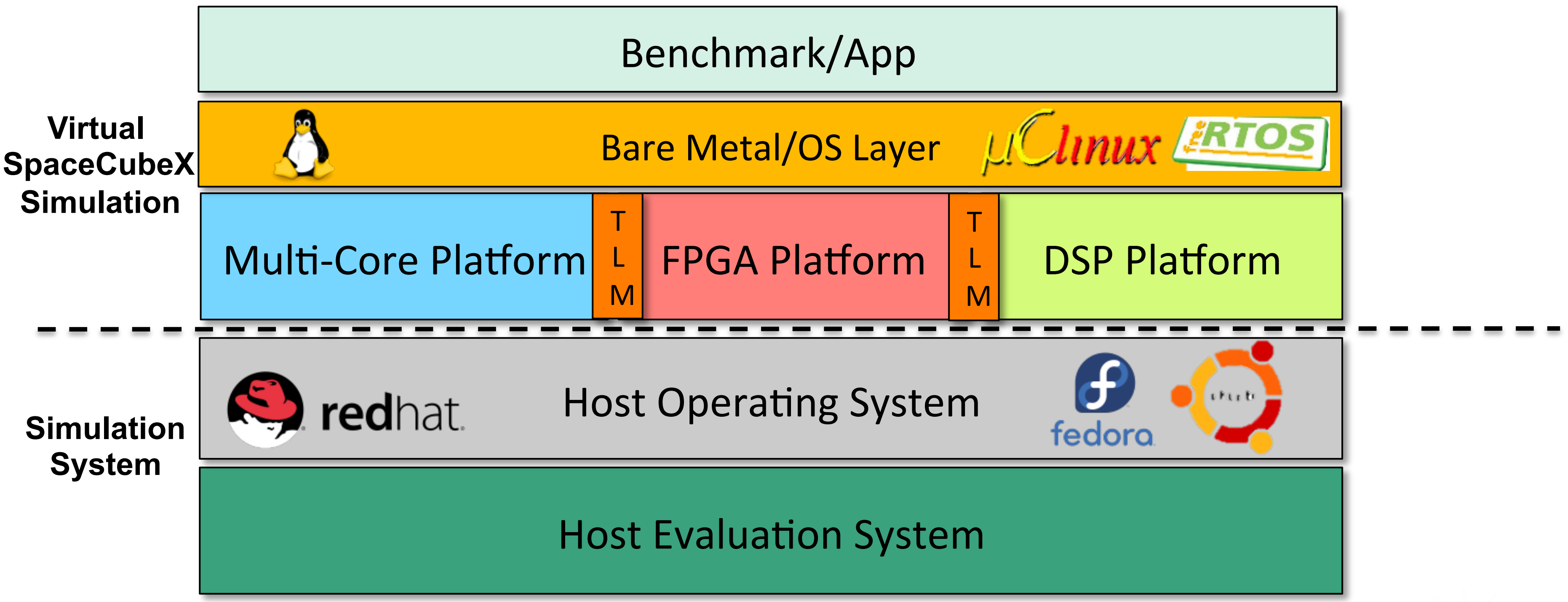


```
    freopen("sci2dbg", "w", stdout);
#endif
    _ioctl(_fileno(stdout), UART_SETSPEED, &baud);
    NutSleep(200);
    printf("\n\nNut/OS %s HTTP Daemon...", NutVersionString())

#ifdef NUTDEBUG
    NutTraceTcp(stdout, 0);
```

```
Loaded plugin STK500
Loaded objectfile: C:\ethernut-3.9.9\nutapp-21b-icc\httpd\httpd.cof
```

Simulation Environment Run-Time Architecture

- Simulation environment stack running on Host PC
- SpaceCubeX's Hybrid architectures are virtual system
- Able to provide support for bare metal and OS support in system
- Benchmarks using compilation flow can run in instruction accurate simulation and provide conventional debugging techniques



SpaceCubeX: Simulator Performance Evaluation

- Microbenchmarks used to evaluate run-time performance
  - Hardware platform performance based on real SpaceCube 2.0 system (Xilinx ML510 development board)
  - SpaceCubeX Simulator generated SpaceCube 2.0 simulation platform
- Performance differences within simulator tolerance

	Dhrystone Simulator Performance Evaluation Comparison				
	Runs	Clock Cycles/Instructions	Time (s)	DMIPS	Difference
Hardware Platform	20000	23651950	0.2365	48.12727607	-
SpaceCubeX Simulator	20000	7980020	0.2418	47.07260082	2.22%
Hardware Platform	200000	249200120	2.4920	45.67830574	-
SpaceCubeX Simulator	200000	80000020	2.4242	46.95502526	2.76%

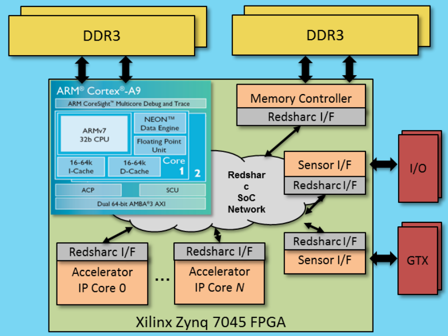
	Whetstone Simulator Performance Evaluation Comparison		
	Clock Cycles/Instructions	Time (s)	Difference
Hardware Platform	13334671	0.1333	
SpaceCubeX Simulator	4780862	0.1449	8.29%

ESTO
Earth Science Technology Office

Architecture Prototypes

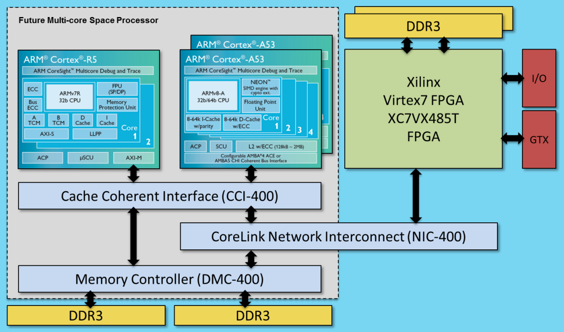
- Architectures selected for initial ‘seed’ to begin trade-space exploration
  - Further permutation later in effort
- Baseline: SpaceCube 2.0 Architecture:
  - Existing Flight system - One RadHard Aeroflex FPGA and two Virtex5

Zynq SoC Architecture



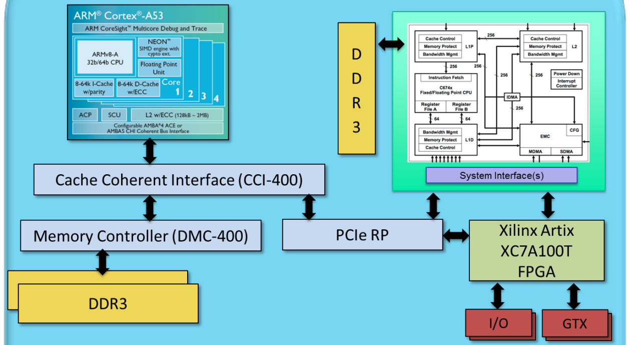
- ARM Cortex-A9 (embedded)
  - Dual ARMv7 800-1000 MHz
  - 64KB L1 Cache / 512 MB L2 Cache
  - Hard floating point unit and NEON data engine
- Xiliinx Zynq 7045 FPGA Fabric
  - Logic Cells: 350,000
  - Block RAM: 17,440 Kb
  - DSP slices: 900
  - GTX (12.5 Gb/s): 16

Hybrid FPGA Architecture


Future Multi-core Space Processor
ARM Cortex-R5
ARM Cortex-A53
DDR3
Xilinx Virtex7 FPGA XC7VX485T FPGA
I/O
GTX
Cache Coherent Interface (CCI-400)
CoreLink Network Interconnect (NIC-400)
Memory Controller (DMC-400)
DDR3
DDR3

- Processor
  - Dual ARM Cortex-A53 processors
    - Quad ARMv8-A 32b/64b Core 1.3 GHz
    - 32KB I-Cache / 32KB D-Cache
    - 1 MB L2 Cache
  - ARM Cortex-R5
    - ARMv7R 1.4 GHz
- Xilinx Virtex 7 FPGA (XC7VX485T)
  - Logic Cells: 485,760
  - Block RAM: 37,080 Kb
  - DSP slices: 2,800
  - GTX (12.5 Gb/s): 56

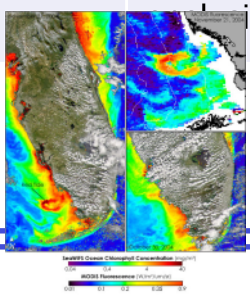
Hybrid DSP Architecture

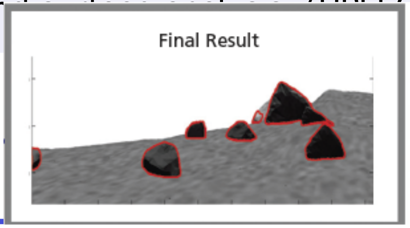

ARM Cortex-A53
DDR3
System Interface(s)
Cache Coherent Interface (CCI-400)
Memory Controller (DMC-400)
PCIe RP
Xilinx Artix XC7A100T FPGA
DDR3
I/O
GTX

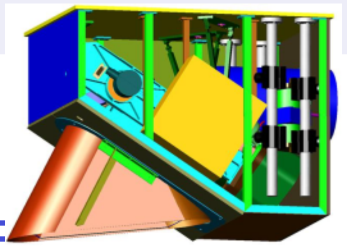
- Processor: ARM Cortex-A53
  - Quad ARMv8-A 32b/64b Core 1.3 GHz
  - 32KB I-Cache / 32KB D-Cache
  - 1 MB L2 Cache
- ClearSpeed CSX700 / BAE RADSPEED
  - TI C6747 DSP stand in
  - 250 MHz core clock frequency
  - 2 SIMD Cores
  - 96 processing elements / core
  - 8K L1 instruction cache / 4K L1 data cache

Application Benchmark Suite

Benchmark	Description
Micro Benchmarks	Kernels to benchmark architecture subcomponents and measure system viability.
NAS Parallel Benchmarks	NASA generated set of programs designed to help evaluate the performance of parallel supercomputers, derived from computational fluid dynamics (CFD) applications and consist of five kernels and three pseudo-applications
Packet Routing	2 kernels: Packet generation and transmission & Packet reception and verification
ATCORR	Atmospheric correction algorithm commonly used in Hyperspectral and other sensing applications
Hyperspectral Classifiers	Two classification kernels: Sulfur, Thermal
Hyperspectral Compression	Lossless Compression algorithm tuned for hyperspectral data
Image segmentation and segment analysis	Autonomous spacecraft tasking, geological feature identification, analysis, and data handling. (HPFEC-3)
Image Classification	Common image processing kernels including feature extraction, shape analysis, and surface analysis (HPFEC-4)



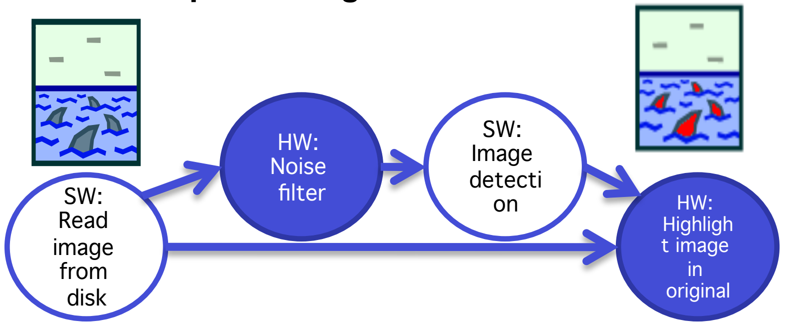




Application Mapping Process

- Existing applications
  - Port code, recompile
  - Existing FPGA extremely helpful
- New Applications (HPFEC-3, HPFEC-4)
  - Utilize REDSHARC to encapsulate kernels using common interface API to facilitate migrating kernels between heterogeneous elements (CPU, DSP, FPGA)
- REDSHARC: Reconfigurable Datastream Software / Hardware ARChitecture
  - REDSHARC infrastructure utilized to standardize simulation framework
  - Application developers target API and get infrastructure for ‘free’
- Optimization
  - Reasonable effort level approach taken
  - Goal to identify best board level architecture, not a mission level application optimization project

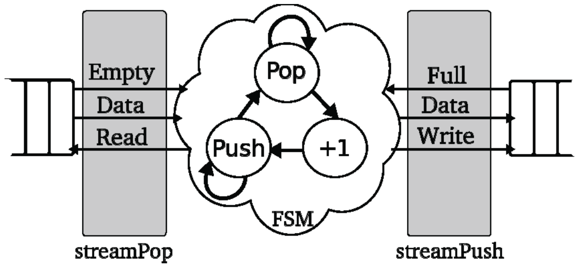
Task graph using heterogeneous processing elements



REDSHARC Uniform APIs

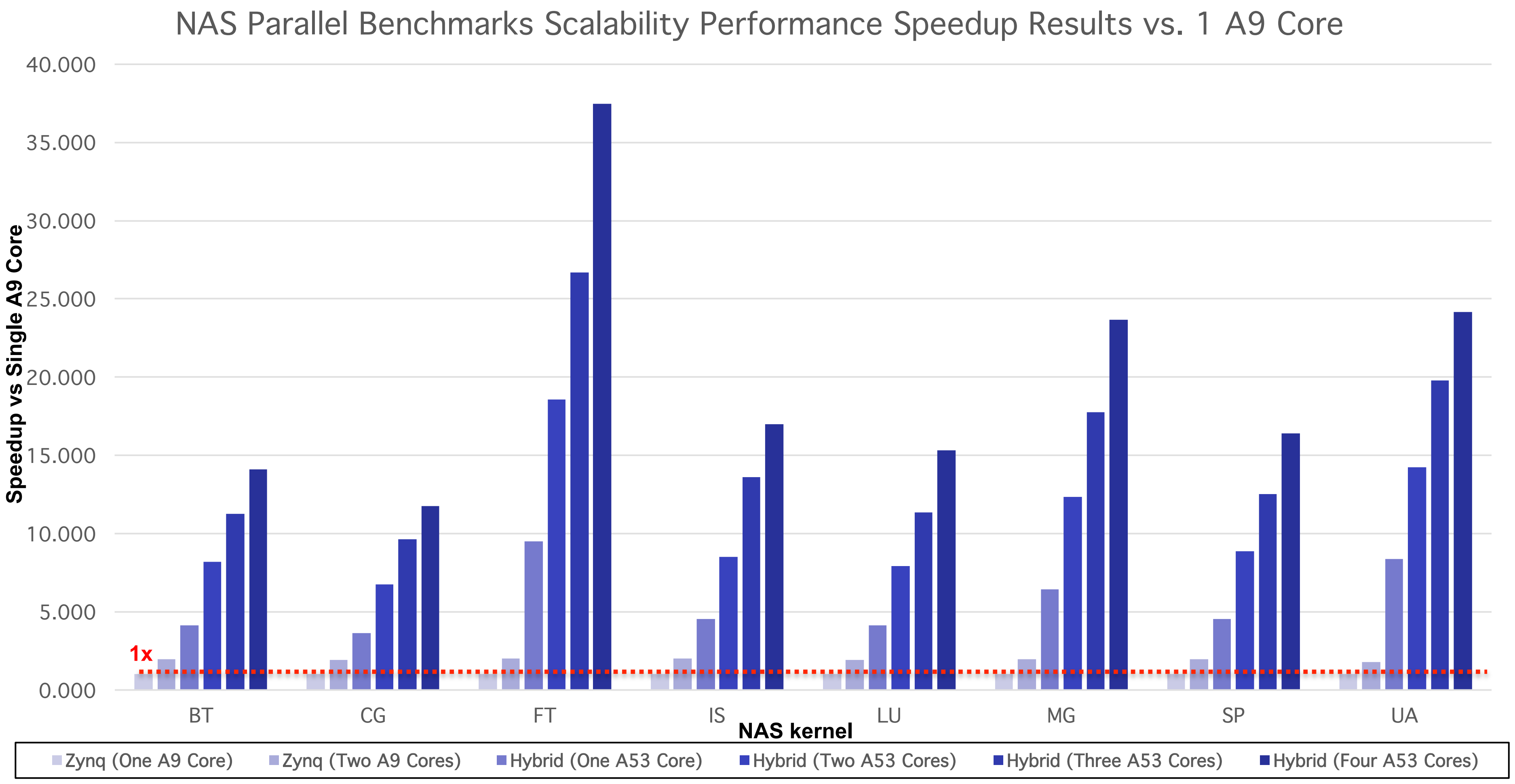
```
while(1) {
  // Pop element off of Stream
  temp = incoming.streamPop();
  //Increment by 1
  temp++;
  output.streamPush(temp);
}
```

Software API Instantiation



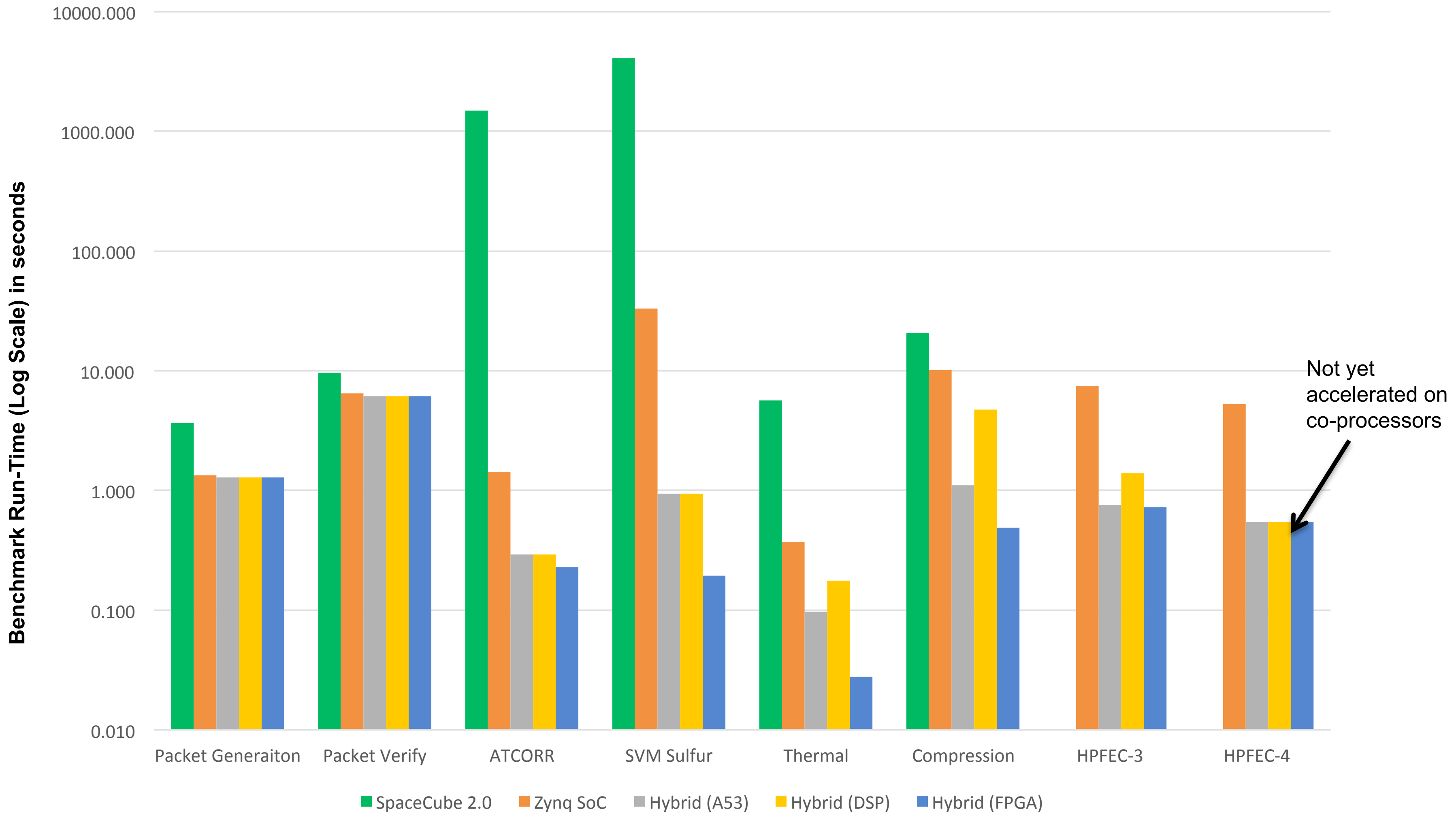
Hardware API Instantiation

NAS Parallel Benchmarks: Results



Multi-core performance scales with the number of cores.
Architectures enabling high performance (cluster in space) class computing capabilities.

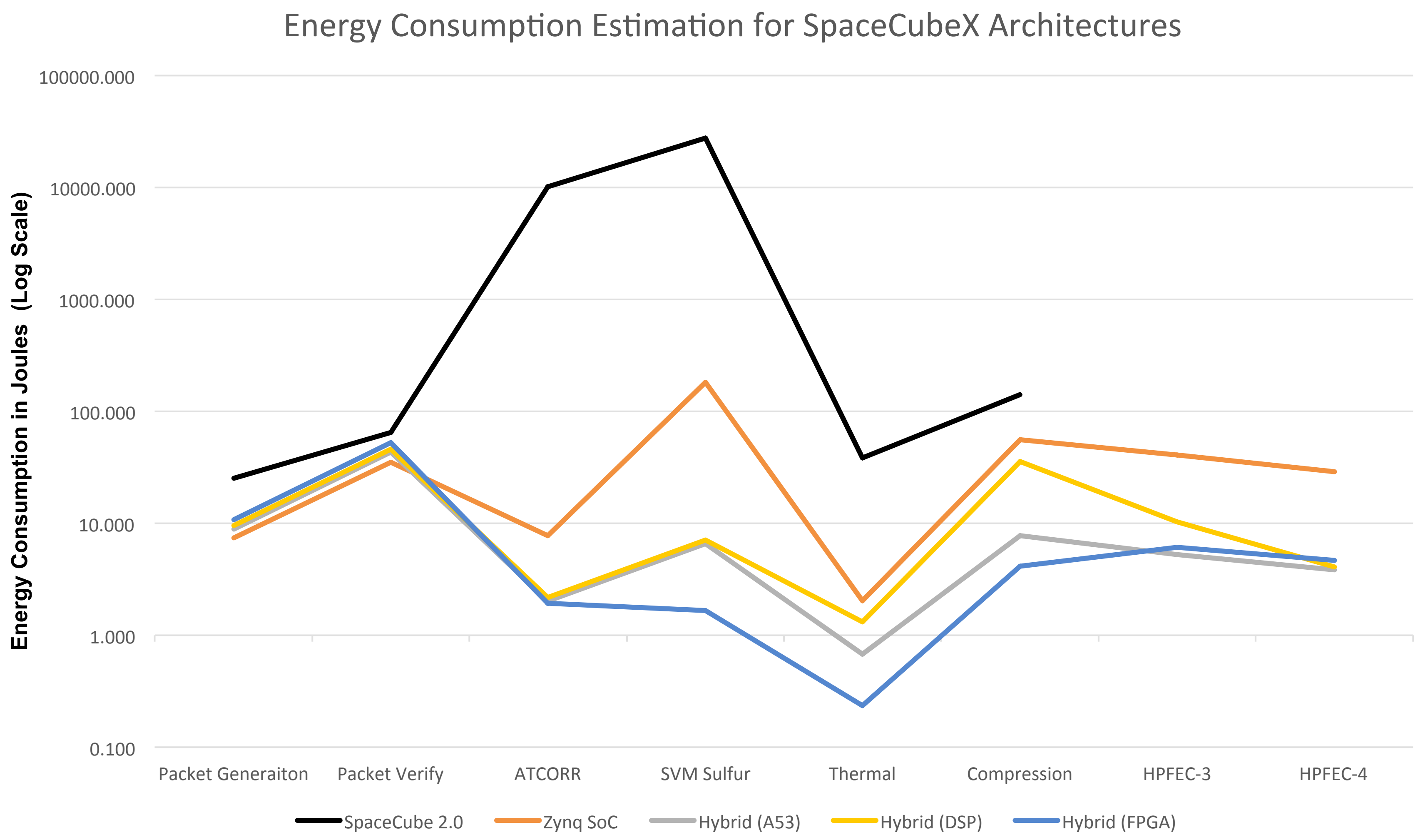
SpaceCubeX: Throughput Comparison



Hybrid DSP and FPGA Architectures provides orders of magnitude higher performance

Earth Science Technology Office

SpaceCubeX: Energy Consumption Comparison



Hybrid DSP and FPGA Architectures provide significant reduction in power usage

Earth Science Technology Office

Benchmarking Summary

- Ran over 200 benchmark application experiments on 8 different permuted architectures for processors, FPGAs, and DSPs
- General observations
  - ARM 53 significantly outperforms A9
  - Multi-core architectures provide fast, scalable approach
  - Hybrid architectures provide best performance / power
  - Hybrid DSP architecture lagging

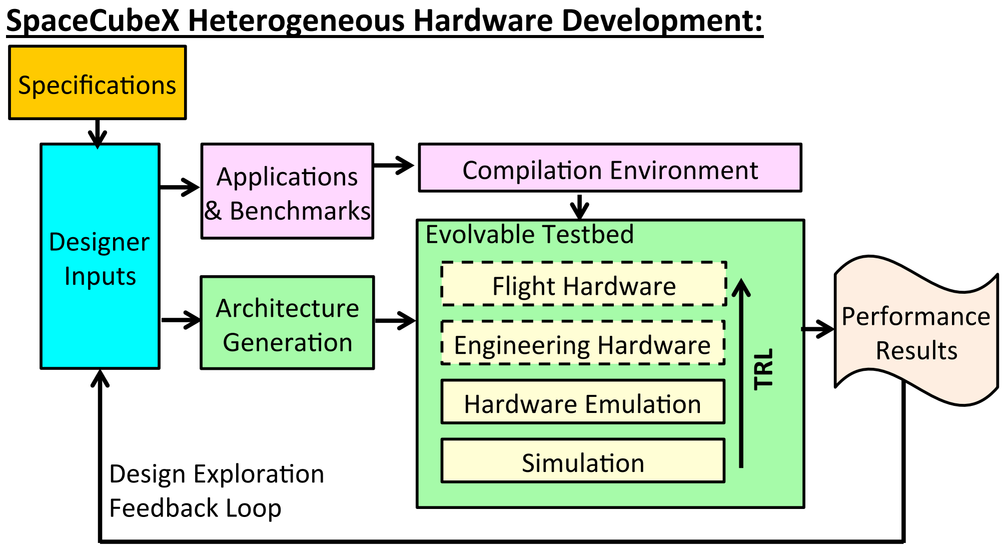
		Space Cube 2.0		Zynq		Hybrid			
Type	Name	1 PPC	2 PPCs	1 ARM A9	2 ARM A9s	1 ARM A53	4 ARM A53s	4 A53s + FPGA	4 A53s + DSP
Diagnostic	Memory Test	Y	Y	Y	Y	Y	Y	N/A	N/A
Diagnostic	Interfaces Test	Y	Y	Y	Y	Y	Y	N/A	N/A
Micro-Benchmarks	Dhrystone / Whetstone / Linpack	Y	N/A	Y	N/A	Y	N/A	N/A	N/A
Micro-Benchmark	NAS Parallel Benchmark s	Y	N/A	Y	Y	Y	Y	N/A	N/A
Application	GSFC Packet Generation	Y	Y	Y	Y	Y	Y	N/A	N/A
Application	GSFC Packet Validation	Y	Y	Y	Y	Y	Y	N/A	N/A
Application	SAR	N	N (Single Core)	Y	N (Single Core)	Y	N (Single Core)	N/A	N/A
Application	ATCORR	Y	N (Single Core)	Y	N (Single Core)	Y	N (Single Core)	Y	N
Application	SVM Sulfur	Y	N (Single Core)	Y	N (Single Core)	Y	N (Single Core)	Y	N
Application	Hyper. Thermal	Y	N (Single Core)	Y	N (Single Core)	Y	N (Single Core)	Y	Y
Application	Hyper. Compressio n	Y	Y	Y	Y	Y	Y	Y	Y
Benchmark	HPFEC-3	N	N	Y	Y	Y	Y	Y	Y

Summary

- Achieved first spiral of system capable of performing board level heterogeneous design space evaluation

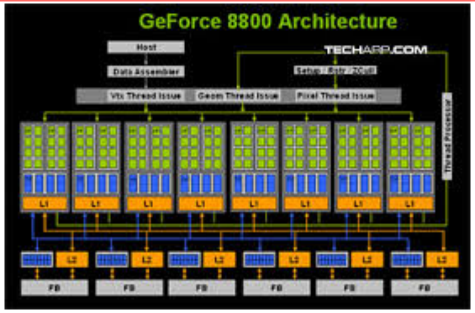
- System capable of supporting additional earth sensing applications

- Year 2
  - Additional architecture permutations
  - Migrating from Simulation to Emulation
  - Assessing additional application

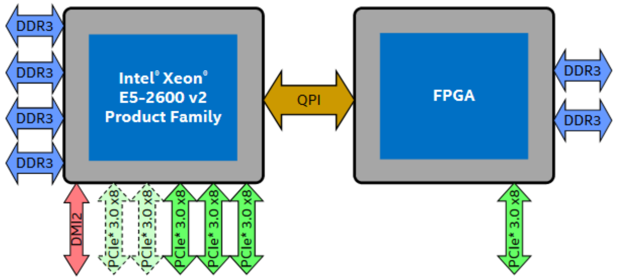


Potential Future Extensions: HPC

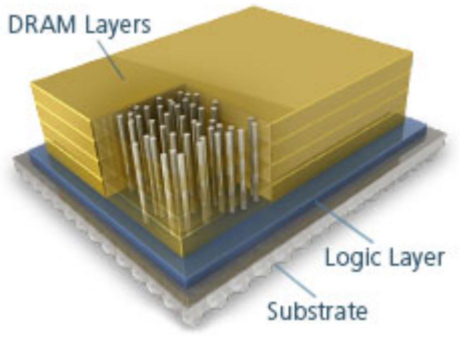
- Extensions to address High Performance Computation
  - GPU-based accelerators
  - Tightly coupled FPGAs
    - Intel HARP
  - More than Moore architectures
    - Micron Hybrid Memory cube
  - ISI has all three architectures in house
- Update framework to incorporate new architectures
- Application benchmarks representative of NASA HPC



Nvidia Graphic Processor Architecture



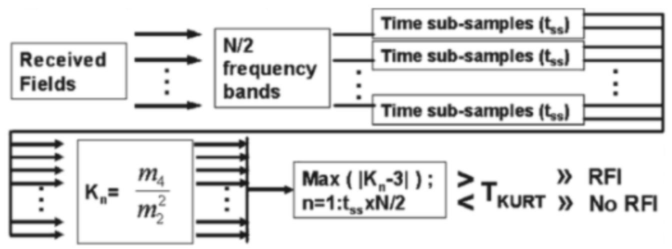
Intel's HARP Data center Architecture



Micron Hybrid Memory Cube

Increase NASA Community Engagement

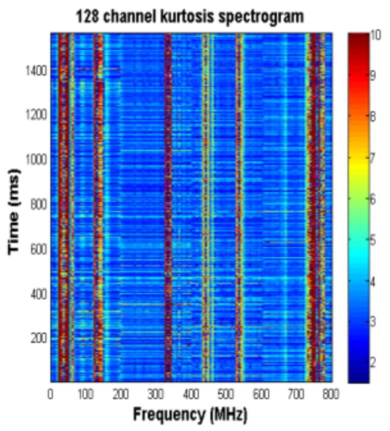
- SpaceCube-X Simulation Infrastructure mature enough to support broader community

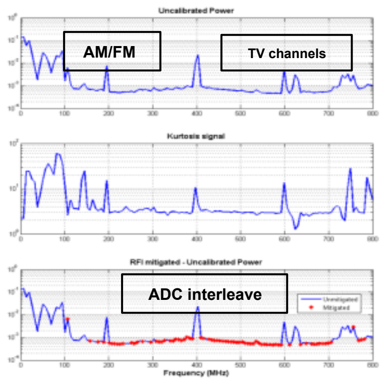


Received Fields
N/2 frequency bands
Time sub-samples (t_{ss})
$K_n = \frac{m_4}{m_2^2}$
Max (|K_n-3|) ; n=1:t_{ss}xN/2
> T_{KURT} » RFI
< T_{KURT} » No RFI

Fig. 3. Schematic of the kurtosis-detection model.

- Example: Radio Frequency Interference Detection and Mitigation Applications
  - JPL Group (Livesey, Kocz, Misra) approached SpaceCubeX team after ESTF2015
  - New approach discriminates between RFI and natural thermal emissions signals in L and C bands





Interested in how your application performs? Come talk to us!
Matthew French mfrench@isi.edu

QUESTIONS?

FPGA Simulation

REconfigurable Data-stream Hardware/Software ARChitecture:

- Provides rapid construction of System-on-Chip platforms
- Enables design space exploration of heterogeneous resources
  - Seamless HW/SW task migration supports spiral development
- Combines situational awareness with high performance processing
  - Dynamic resource management and performance tuning
- System infrastructure (network/interfaces) pre-designed and verified
  - Focus shifts to design and verification of hardware accelerator cores
- SpaceCubeX leverages Redsharc for rapid IP integration of accelerator peripherals into Hybrid FPGA Simulation Platform
  - JPL: Edge Detection & Image Convolution Kernels
  - ISI: Hyperspectral Image Compression, Thermal & Sulfur Classification Kernels

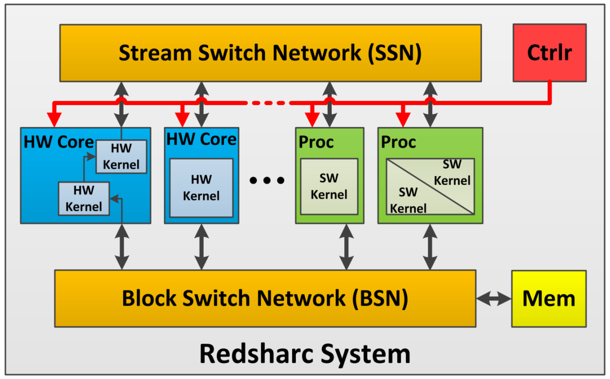
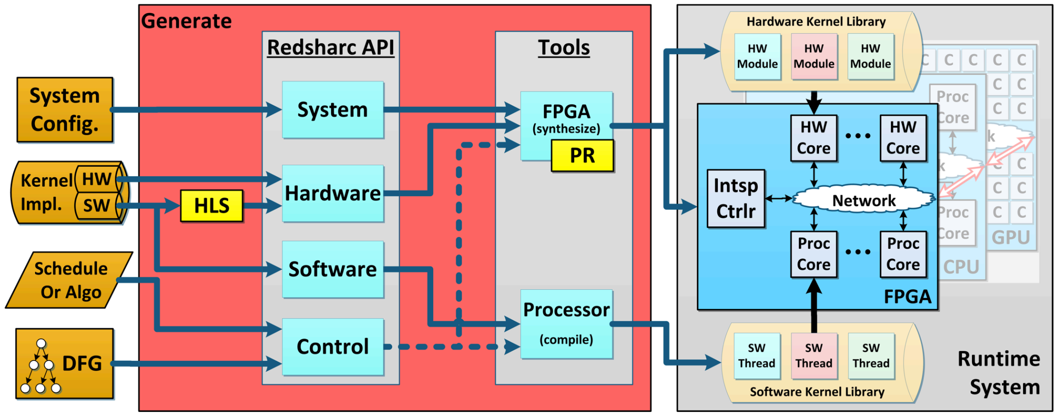
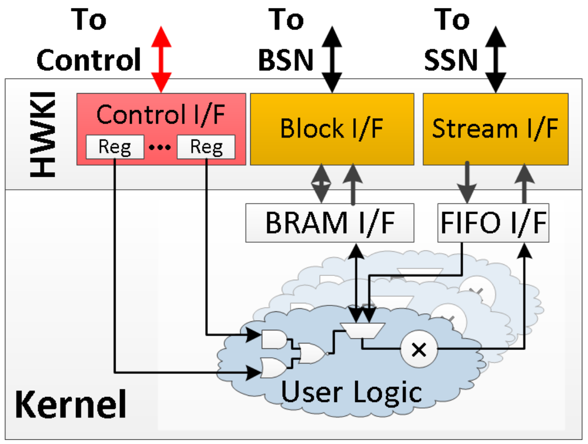


Diagram of Redsharc System Components on an FPGA



Redsharc's Build Infrastructure

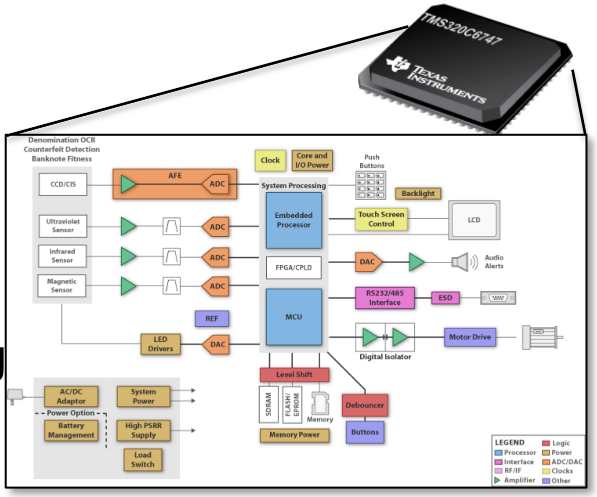


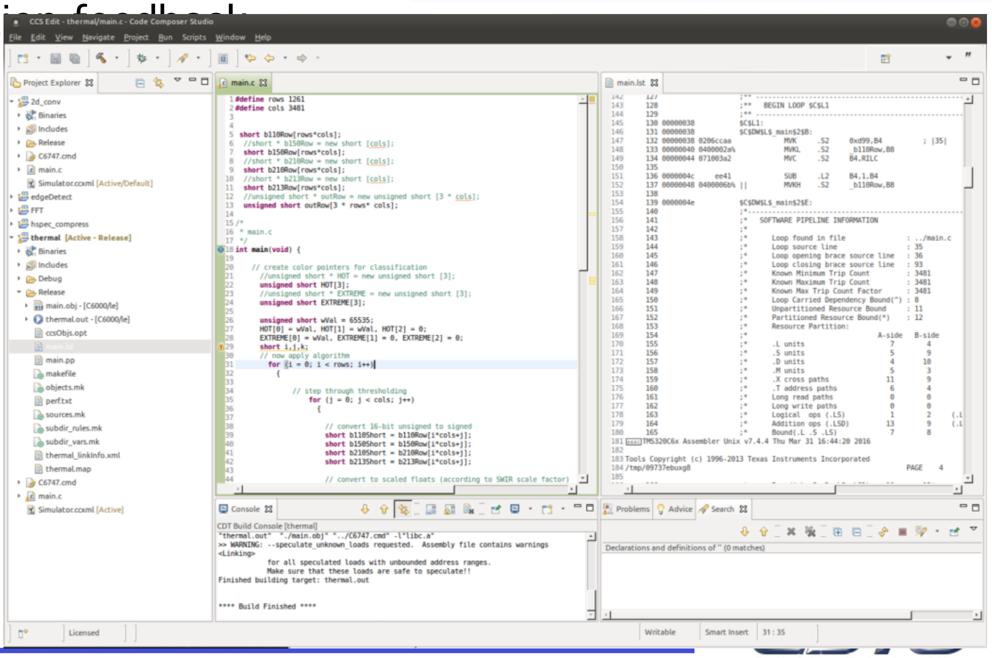
Redsharc's Hardware Kernel Interface (HWKI)

Redsharc Philosophy: Interoperate and leverage current and emerging devices, tools, and technology

DSP Simulation

- Limited availability of DSP simulation environments
- Selected TI's Code Composer Studio (CCS)
  - Support for multiple DSPs
  - Instruction/cycle-accurate simulation
  - Scriptable access to running simulator and extracting results
  - Eclipse-based development environment
    - Optimizing C compiler with optimization feedback
- SpaceCubeX preliminary evaluation on TI's C6747 DSP:
  - Low power – ½W @ 300 MHz
  - Built-in floating point support - 4 SP adds and 2 SP multiplies/cycle
    - Also supports 4 16 bit x 16 bit integer multiply/adds per cycle
- Evaluation framework supports selection of different TI DSPs for hybrid platform analysis





```
#define rows 1261
#define cols 3481


 short b110Row[rows*cols];
  //short * b150Row = new short [cols];
  short b150Row[rows*cols];
  //short * b210Row = new short [cols];
  short b210Row[rows*cols];
  //short * b213Row = new short [cols];
  short b213Row[rows*cols];
  //unsigned short * outRow = new unsigned short [3 * cols];
  unsigned short outRow[3 * rows* cols];

/*
 * main.c
 */
int main(void) {

    // create color pointers for classification
      //unsigned short * HOT = new unsigned short [3];
      unsigned short HOT[3];
      //unsigned short * EXTREME = new unsigned short [3];
      unsigned short EXTREME[3];

      unsigned short wVal = 65535;
      HOT[0] = wVal, HOT[1] = wVal, HOT[2] = 0;
      EXTREME[0] = wVal, EXTREME[1] = 0, EXTREME[2] = 0;
      short i,j,k;
      // now apply algorithm
        for (i = 0; i < rows; i++)
          {

              // step through thresholding
                  for (j = 0; j < cols; j++)
                    {

                      // convert 16-bit unsigned to signed
                      short b110Short = b110Row[i*cols+j];
                      short b150Short = b150Row[i*cols+j];
                      short b210Short = b210Row[i*cols+j];
                      short b213Short = b213Row[i*cols+j];

                      // convert to scaled floats (according to SWIR scale factor)
```

```
   127                    ;**-------------------------------------------------
   128                    ;**   BEGIN LOOP $C$L1
   129                    ;**-------------------------------------------------
   130 00000038           $C$L1:
   131 00000038           $C$DW$L$_main$2$B:
   132 00000038 0206ccaa           MVK     .S2     0xd99,B4        ; |35|
   133 00000040 0400002a%          MVKL    .S2     _b110Row,B8
   134 00000044 071003a2           MVC     .S2     B4,RILC
   135
   136 0000004c     ee41           SUB     .L2     B4,1,B4
   137 00000048 0400006b% ||       MVKH    .S2     _b110Row,B8
   138
   139 0000004e           $C$DW$L$_main$2$E:
   140                    ;*-------------------------------------------------
   141                    ;*   SOFTWARE PIPELINE INFORMATION
   142                    ;*
   143                    ;*      Loop found in file               : ../main.c
   144                    ;*      Loop source line                 : 35
   145                    ;*      Loop opening brace source line   : 36
   146                    ;*      Loop closing brace source line   : 93
   147                    ;*      Known Minimum Trip Count         : 3481
   148                    ;*      Known Maximum Trip Count         : 3481
   149                    ;*      Known Max Trip Count Factor      : 3481
   150                    ;*      Loop Carried Dependency Bound(^) : 8
   151                    ;*      Unpartitioned Resource Bound     : 11
   152                    ;*      Partitioned Resource Bound(*)    : 12
   153                    ;*      Resource Partition:
   154                    ;*                                A-side   B-side
   155                    ;*      .L units                     7        4
   156                    ;*      .S units                     5        9
   157                    ;*      .D units                     4       10
   158                    ;*      .M units                     5        3
   159                    ;*      .X cross paths              11        9
   160                    ;*      .T address paths             6        4
   161                    ;*      Long read paths              0        0
   162                    ;*      Long write paths             0        0
   163                    ;*      Logical  ops (.LS)           1        2     (.L
   164                    ;*      Addition ops (.LSD)         13        9     (.L
   165                    ;*      Bound(.L .S .LS)             7        8
TMS320C6x Assembler Unix v7.4.4 Thu Mar 31 16:44:20 2016

Tools Copyright (c) 1996-2013 Texas Instruments Incorporated
/tmp/09737ebuxg8                                            PAGE    4
```

CDT Build Console [thermal]

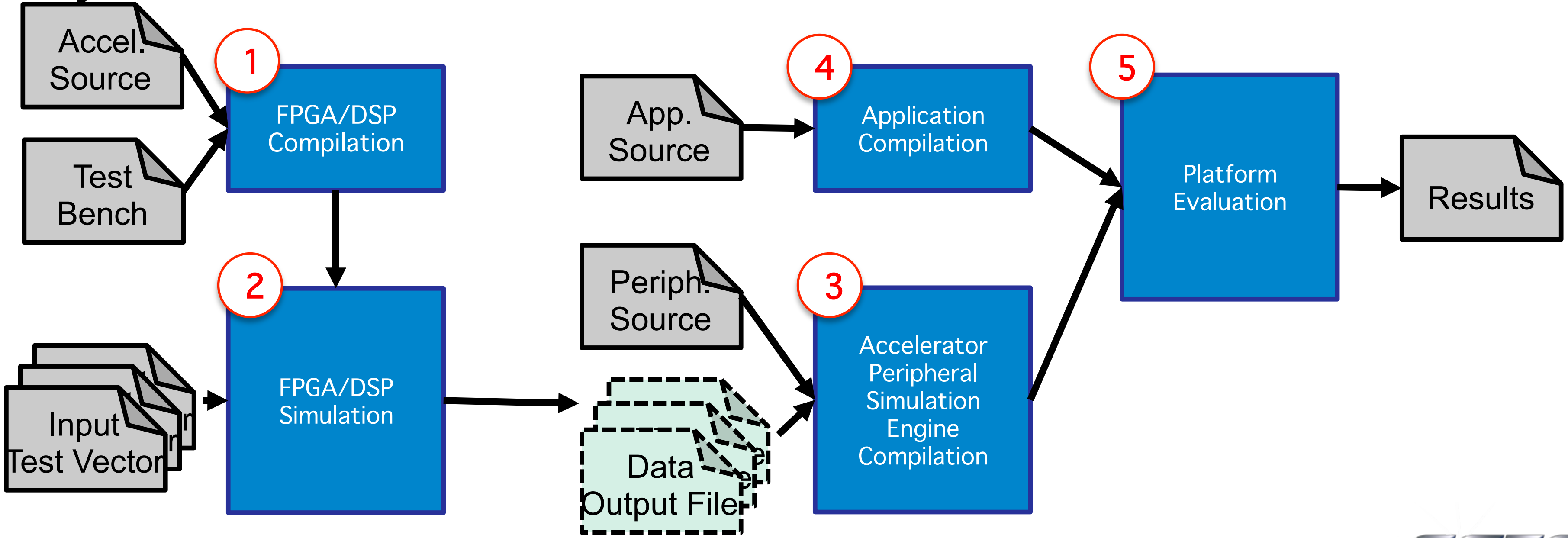
```
"thermal.out"  "./main.obj" "../C6747.cmd" -l"libc.a"
>> WARNING: --speculate_unknown_loads requested.  Assembly file contains warnings
<Linking>
           for all speculated loads with unbounded address ranges.
           Make sure that these loads are safe to speculate!!
Finished building target: thermal.out


**** Build Finished ****
```

Declarations and definitions of '' (0 matches)

Earth Science Technology Office

Accelerator Peripheral: Multi-Simulator Build Process

- Added support to compile, run, and collect co-simulator results
  1. Compile accelerator source code, test bench, and test vectors using standard tools (e.g. Synopsys VCS, TI's CCS)
  2. Run accelerator and collect output results (data and delays)
  3. Plug results into peripheral memory files
  4. Compile application (independent of co-simulator results/data)
  5. Run applications in simulator accessing memory files
- Peripheral can be integrated into different platform evaluation systems


Accel. Source
Test Bench
Input Test Vector
1
FPGA/DSP Compilation
2
FPGA/DSP Simulation
App. Source
Periph. Source
Data Output File
4
Application Compilation
3
Accelerator Peripheral Simulation Engine Compilation
5
Platform Evaluation
Results