CSP: A Multifaceted Hybrid System for Space Computing

CHREC
NSF Center for High-Performance Reconfigurable Computing

MAPLD 2014

UF UNIVERSITY of FLORIDA
Virginia Tech VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY
BYU BRIGHAM YOUNG UNIVERSITY
THE GEORGE WASHINGTON UNIVERSITY WASHINGTON DC

Dr. Alan George
Professor of ECE
University of Florida

Gary Crum
Aerospace Technologist
Goddard Space Flight Center

Dr. Michael Wirthlin
Professor of ECE
Brigham Young University

Dr. Herman Lam
Associate Professor of ECE
University of Florida

Patrick Gauvin
Jonathan Urriste
Dylan Rudolph
Jacob Stewart
Chris Wilson
Chris Morales
Aaron Stoddard
Alex Wilson
Research Students

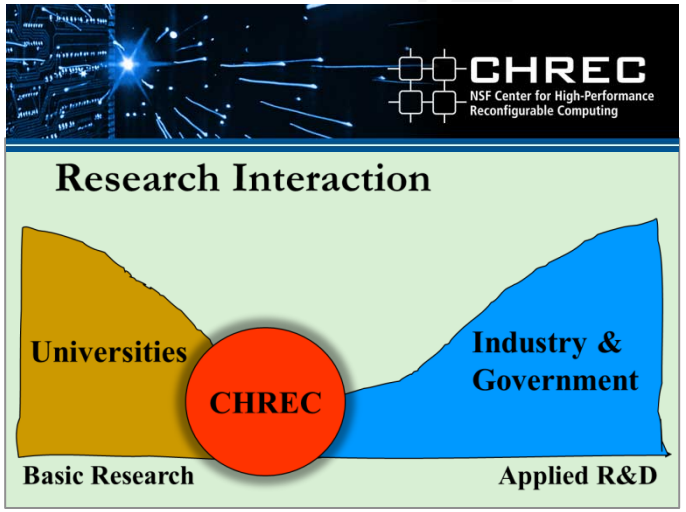
Outline: CHREC Space Processor (CSP)

- CSP Acknowledgements
- Advanced Space Computing
- CSP Hardware Architecture
- CSP Software Architecture
- Configuration Scrubbing and Processor Logging
- Conclusions

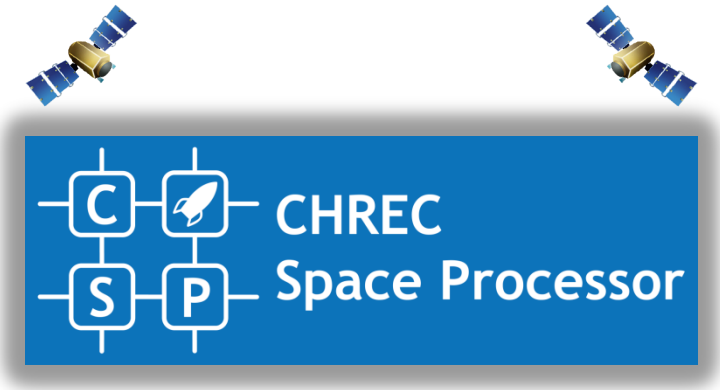
CSP Acknowledgements

National Science Foundation
WHERE DISCOVERIES BEGIN

- CSP is a research project at CHREC
  - NSF Center for High-Performance Reconfigurable Computing (CHREC)
    - Founded in 2007
    - Comprised of 4 university sites and 30 industry and government partners
    - Ranked by NSF as one of top national centers

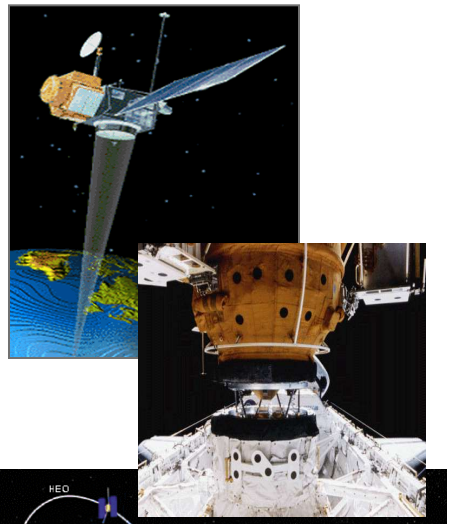


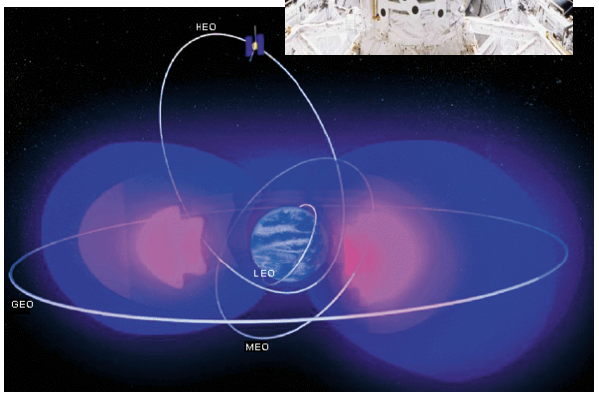
- CSP is a collaborative CHREC effort
  - Original partners:
    - University of Florida, NASA Goddard, and Brigham Young University
  - Additional partners:
    - NASA Kennedy, Honeywell, Space Micro, L-3 CE, NASA Johnson, NASA Ames, Xilinx, and growing!



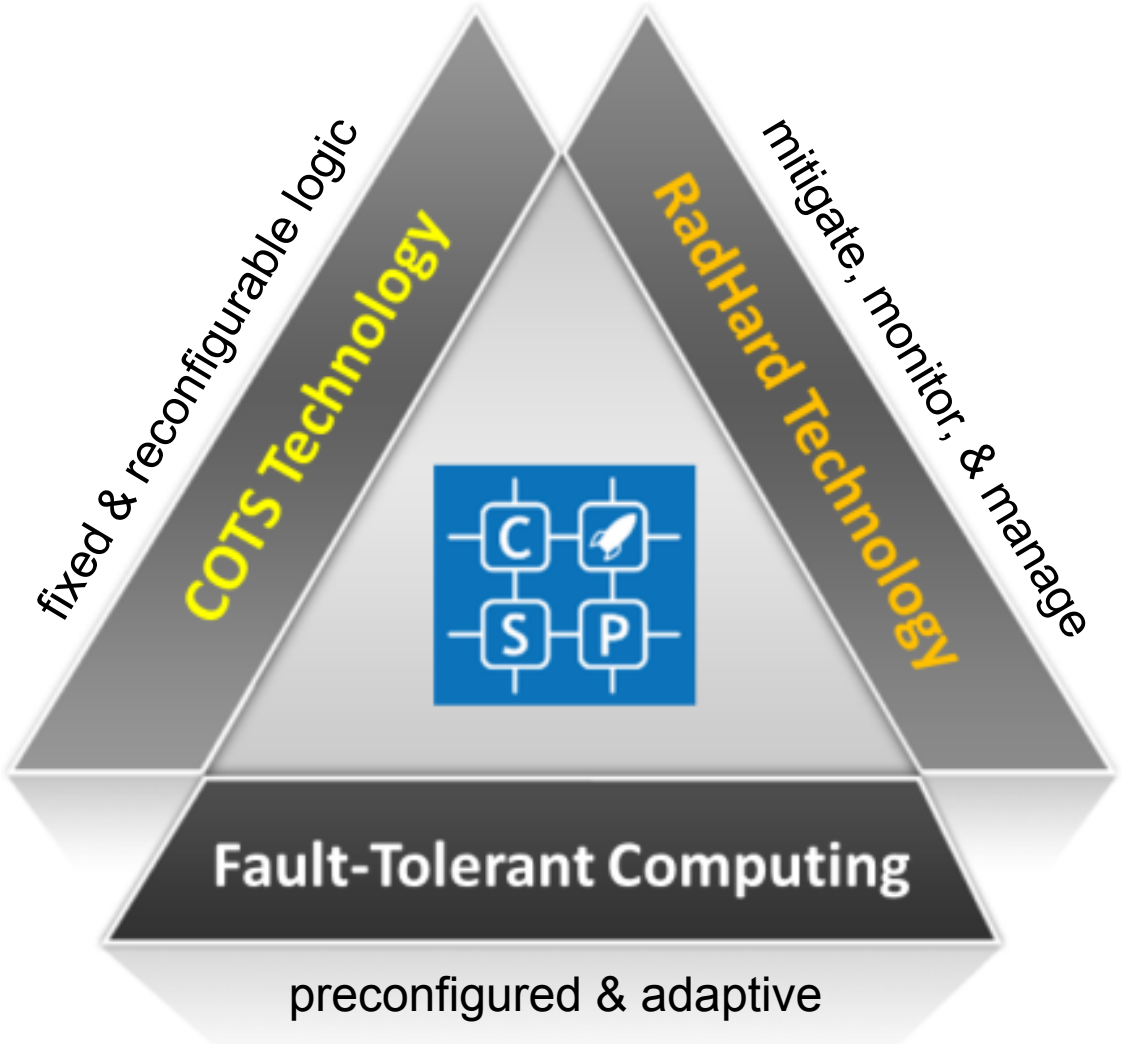
Advanced Space Computing

- What is it? (a.k.a. advanced spaceborne, space-based, or on-board processing)
  - New concepts, methods, and technologies to enable and deploy **high-performance computing in space** – for an increasing variety of missions and applications
- Why is it increasingly necessary and important?
  - **Escalating demands for sensor-data processing**
    - Downlink bandwidth to Earth is extremely limited
    - Sensor data rates, resolutions, and modes are dramatically increasing
    - Remote data processing from Earth is no longer viable
    - Must (pre)process sensor data in-situ, where it is captured
  - **Escalating demands for autonomous processing & control**
    - Remote control from Earth is often impractical
    - Severe propagation delays and bandwidth limits
    - Space and space-delivered vehicles requiring autonomy
    - Autonomy requires high-speed computing for decision-making
- Why is it difficult to achieve?
  - **Cannot simply strap a rocket to a Cray** ☺
    - Hazardous radiation environment in space
    - Platforms with limited power, weight, size, cooling, etc.





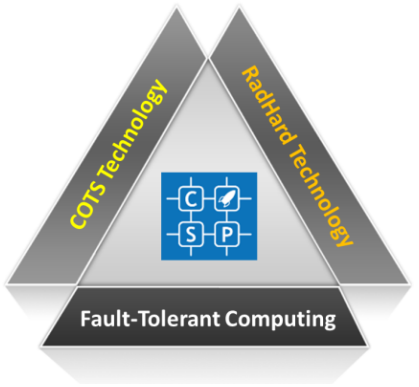
CSP Performability Concept



COTS = commercial off-the-shelf
RadHard= radiation-hardened

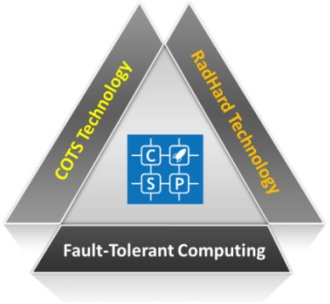
CSP Theme

- **CSP as new approach for space computing**
  - Multifaceted hybrid processing
    - Hybrid system architecture (COTS + RadHard)
    - Hybrid device architecture (Multicore + FPGA)
    - Novel mix of COTS, RadHard, & FTC
  - Unprecedented computing agility in space
    - Static & dynamic optimization of performance, power, reliability
    - COTS technology featured, augmented by RH & FTC
    - Scalable blocks for small, large, & clustered spacecraft
- **Potential uses and mission support**
  - Command & data handling, experiment & instrument control, data compression, sensor processing, attitude control, et al.



RH= radiation-hardened
FTC = fault-tolerant computing

COTS+RH+FTC on CSPv1



COTS

- Zynq-7020 hybrid SoC
  - Dual ARM A9/Neon cores
  - Artix-7 FPGA fabric + hard IP
- DDR3 memory

RadHard

- NAND flash
- Power circuit
- Reset circuit
- Watchdog unit

intersil

FTC = Fault-Tolerant Computing

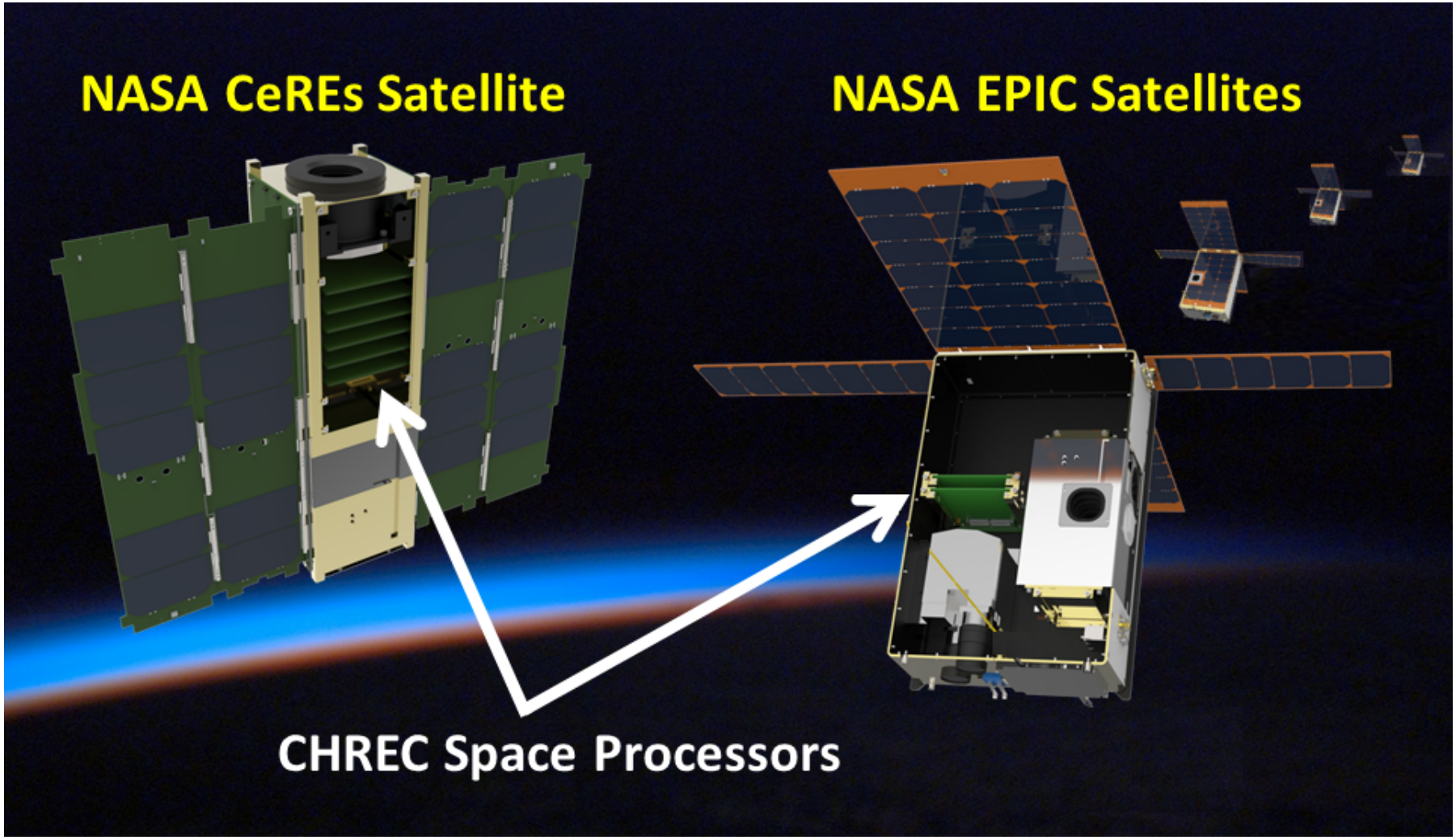
- Variety of mechanisms
  - External watchdog unit to monitor Zynq health and reset as needed
  - RSA-authenticated bootstrap (primary, secondary) on NAND flash
  - ECC memory controller for DDR3 within Zynq
  - ADDAM middleware with message, health, and job services
  - FPGA configuration scrubber with multiple modes
  - Internal watchdogs within Zynq to monitor behavior
  - Optional hardware, information, network, software, and time redundancy

NASA Missions for CSPv1

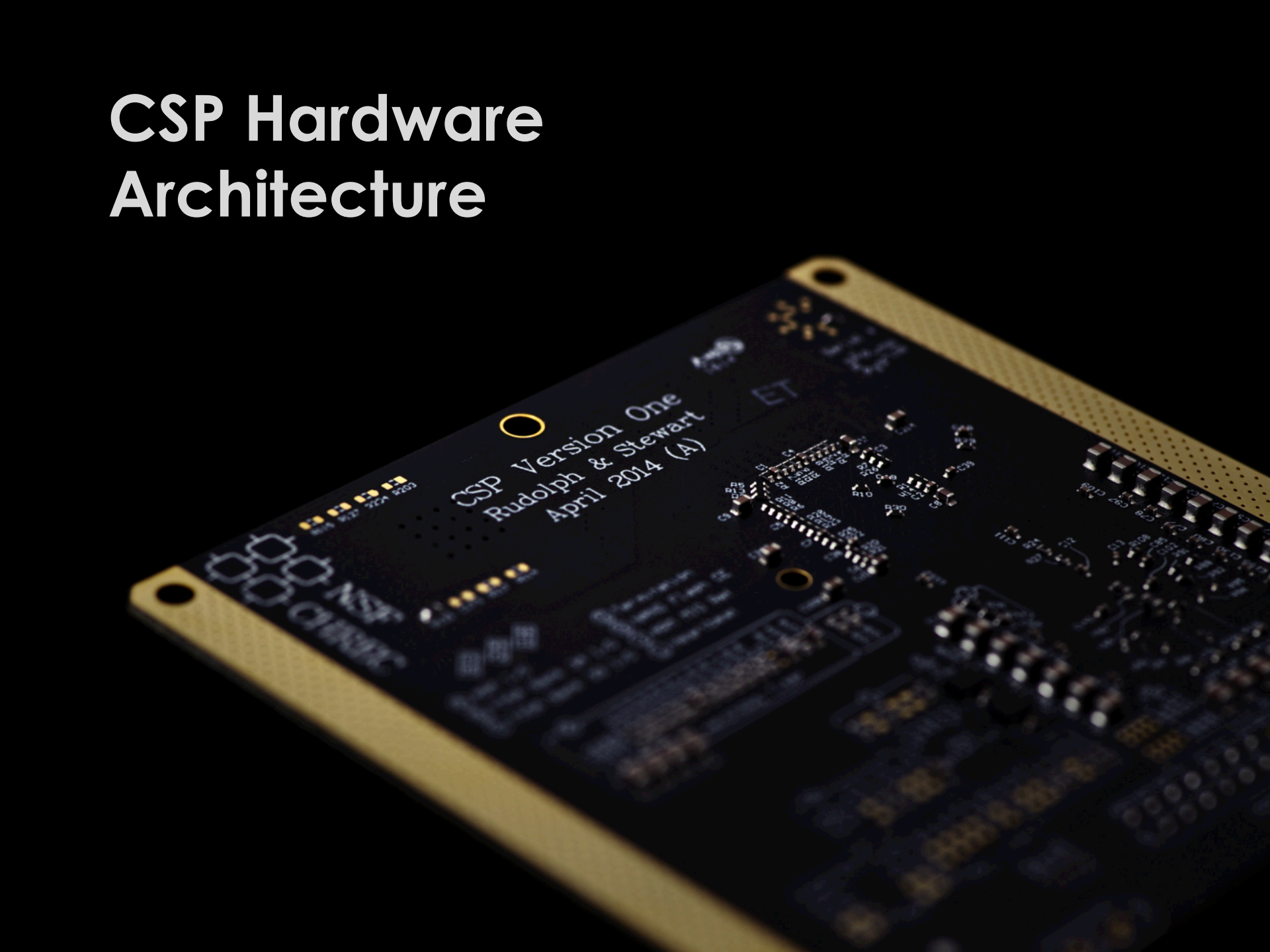
- **Two confirmed for NASA Goddard**
  - CSP featured as new technology for space computing
- **NASA technology mission**
  - STP-H5/ISEM on ISS – late 2014 delivery
  - Two CSPv1 computers working in tandem
  - SpaceWire, Camera Link, reconfiguration

- **NASA science mission**
  - CeREs Cubesat – early 2015 delivery to NASA
  - Heliophysics experiment in LEO
  - One CSPv1 computer for on-board processing
- **Additional missions in planning**
  - e.g., EPIC (Earth Photosynthesis Imaging Cluster) satellites

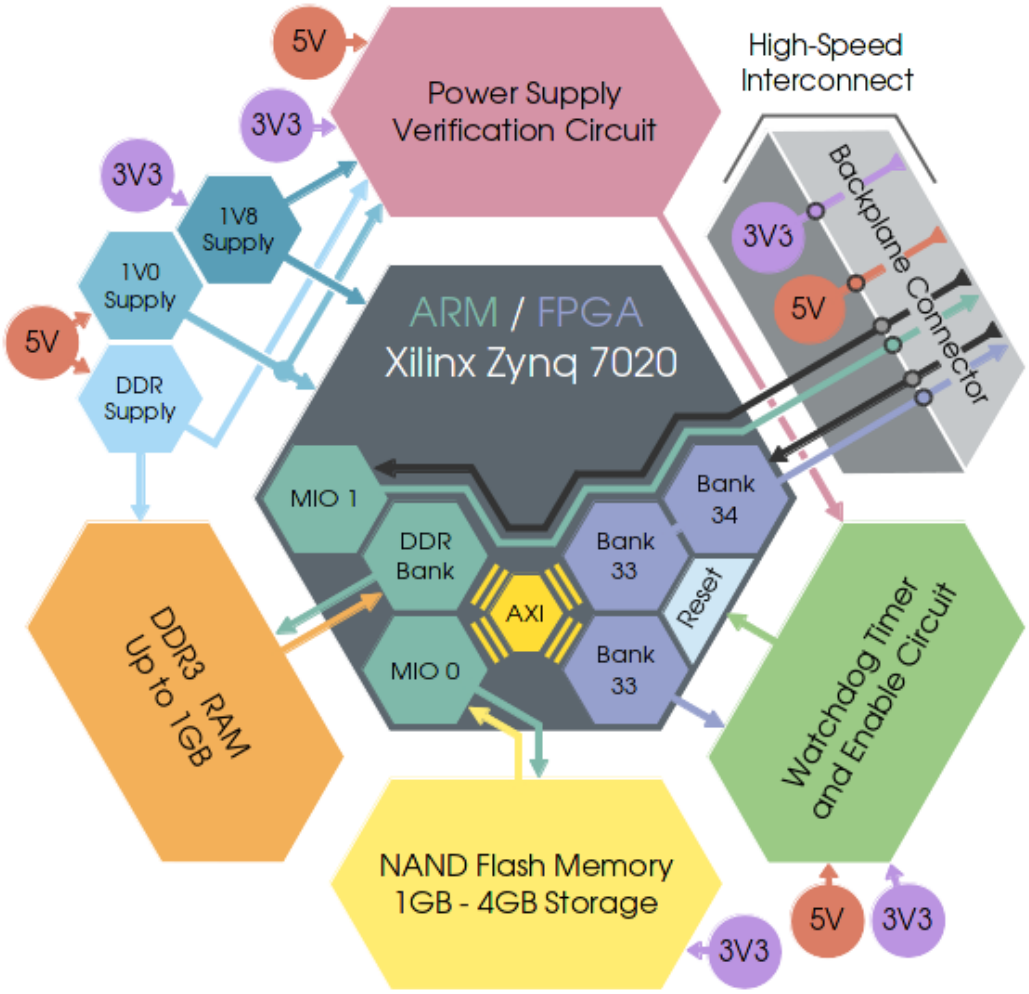
Artist Illustrations (c/o NASA)



CSP Hardware Architecture



CSPv1 System Description

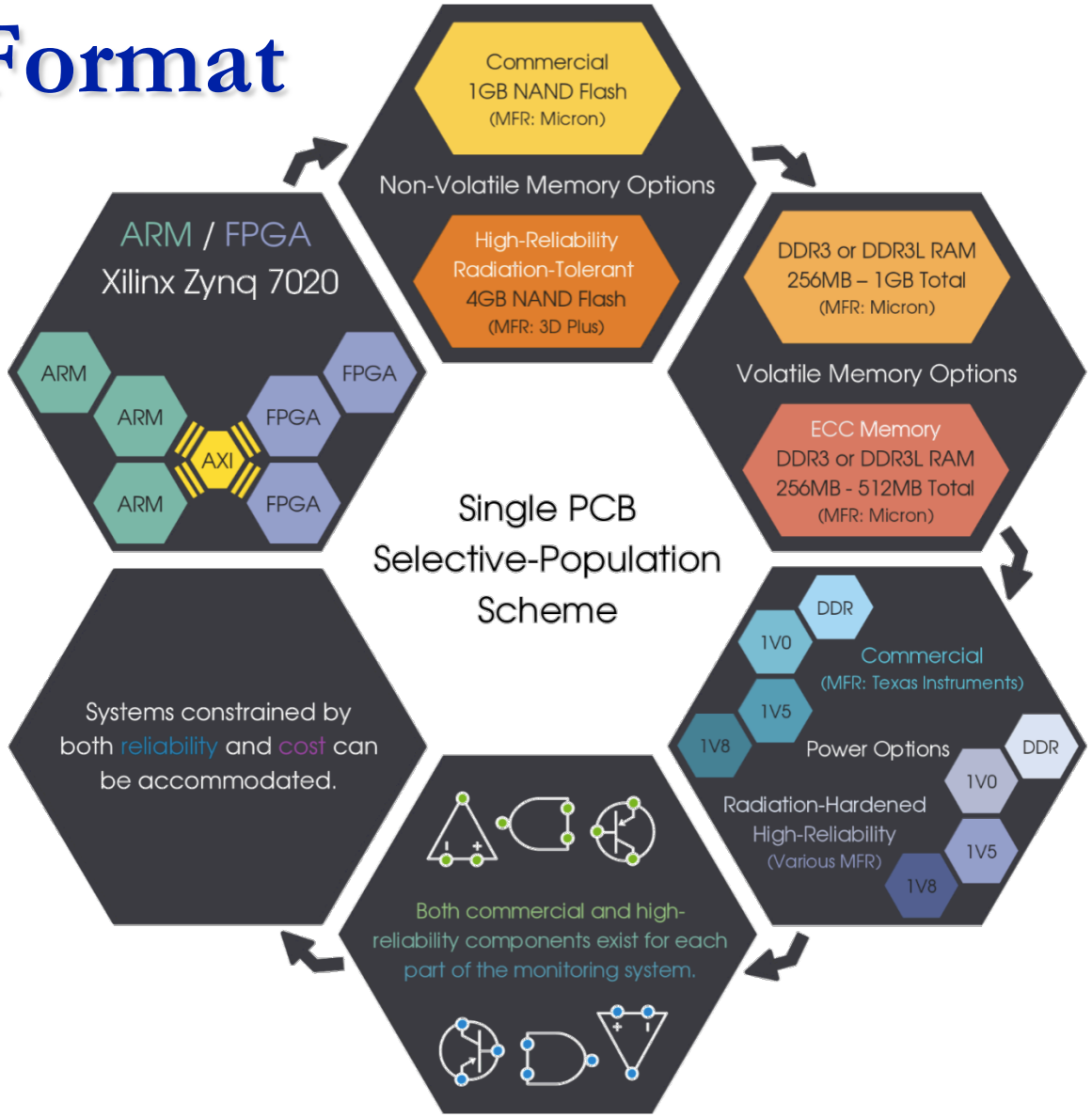


- Requires 3V3 and 5V0, other voltages generated on-board
  - 1.7 watts in minimum-power load state
- 160-pin Samtec Searray connector (no other I/O)
  - 60 High-Speed FPGA I/O pins
  - 26 High-Speed ARM I/O pins
- Watchdog circuit based around Intersil supervisor
- Other information:
  - 50-60 grams loaded
  - 1U form factor (10x10 cm)
  - 62 mil thickness, 12-layer PCB
  - CSPv1 configured for 2x256MB DDR3 RAM

CSPv1 Dual Format

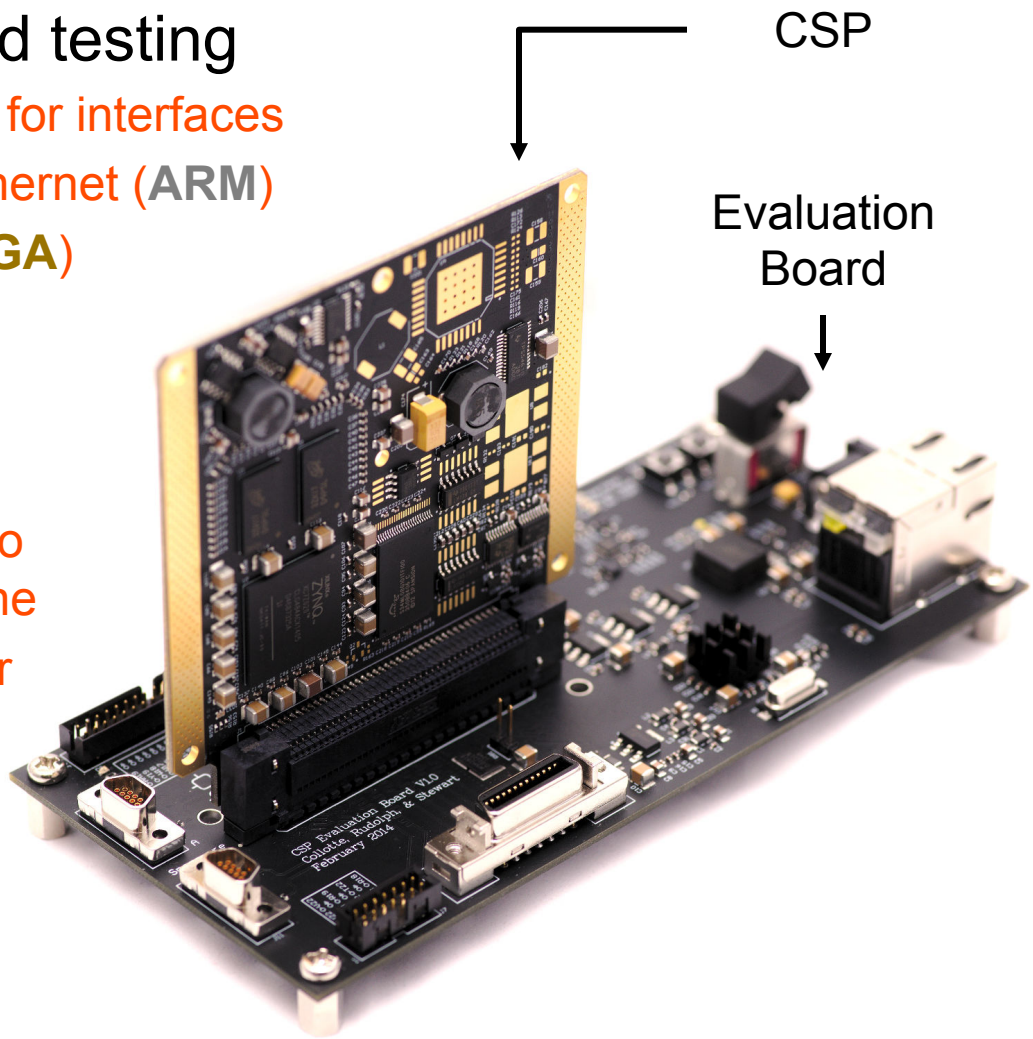
COTS Board
(COTS+FTC)

Hybrid Board
(COTS+RH+FTC)



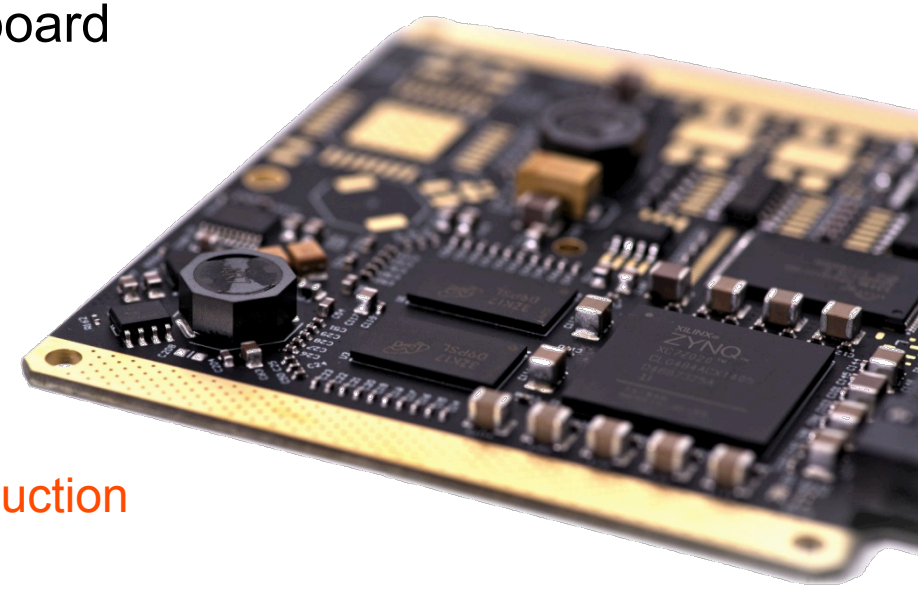
Integration Scheme

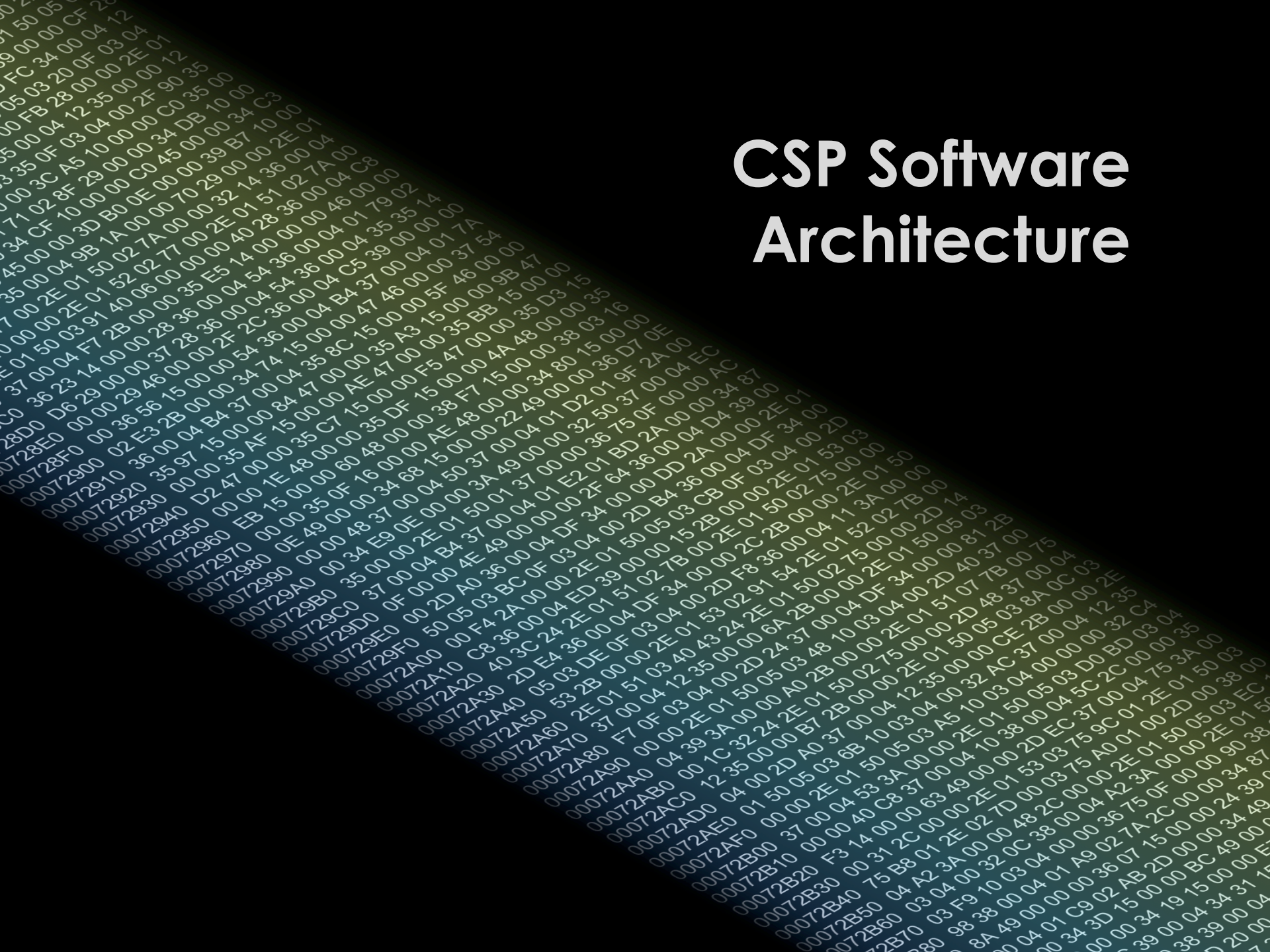
- Evaluation board for ground testing
  - Also serves as reference design for interfaces
  - USB Host and OTG / Gigabit Ethernet (**ARM**)
  - SpaceWire and Cameralink (**FPGA**)
  - 12 Unbound GPIO pins (**FPGA**)
- Flight integration options
  - **Backplane:** right-angle mating, multi-board stack all connected to single passive or active backplane
  - **Motherboard:** in-line mating pair of boards, only one CSP



CSPv1 Availability

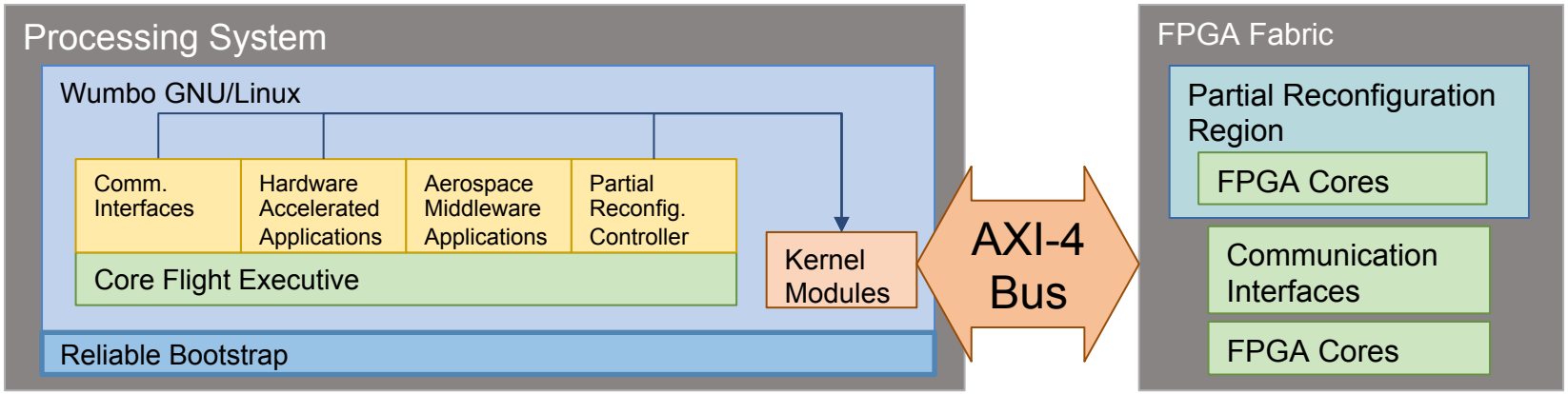
- One CSP board, two population schemes
  - Determine subsystems by mission specifications
    - Consumer off-the-shelf system: optimize cost
    - Hybrid system: maximize reliability
- Evaluation board facilitating system integration
  - Provides interfaces for communication links & debugging
- Integrate with backplane or motherboard
  - Determine by mission specifications
- Production status
  - In possession of production batch of all-COTS boards
  - Production batch of hybrid radiation-hardened / COTS boards en-route
  - In possession of two batches of production evaluation-boards



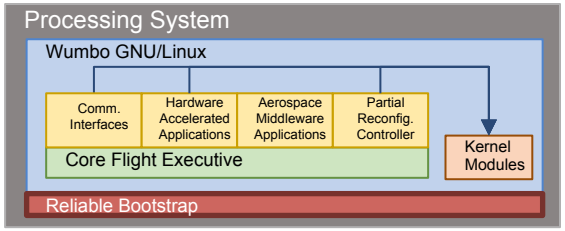

CSP Software
Architecture

System Description

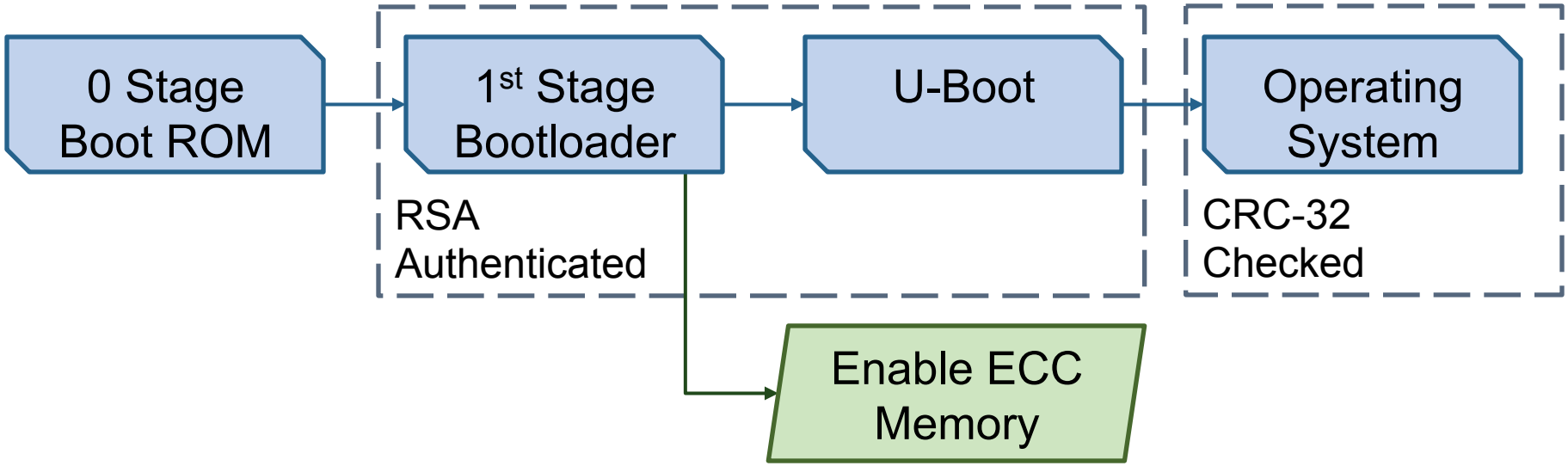
- Goals
  - Adapt to challenges of hazardous environments
  - Develop platform supporting hybrid computing
- Approach
  - Provide reliable bootstrap for run-time software protection with redundant boot images
  - Develop minimal operating system: Wumbo GNU/Linux with processor/FPGA comm.
  - Integrate system using framework for flight systems
  - Support communication interfaces desirable for aerospace applications
  - Facilitate application development on hybrid processing platforms
  - Provide methods for facilitating fault-tolerance in software development



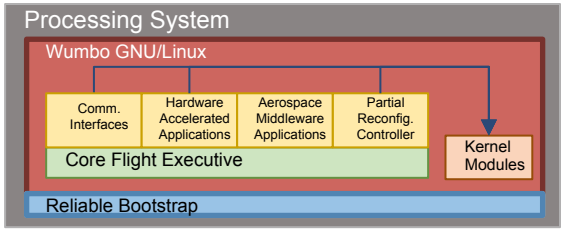
Reliable Bootstrap



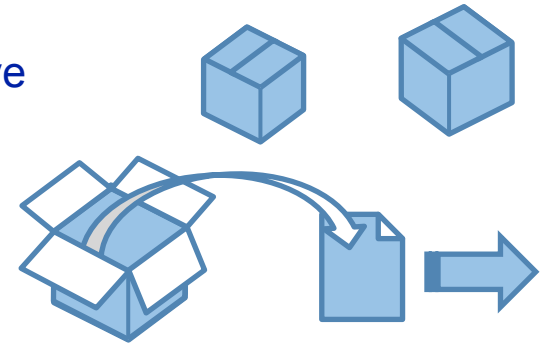
- Goal
  - Provide run-time software protection
- Approach
  - Employ multi-stage boot authentication with fallback options
    - Use RSA key-authentication engine in Zynq
  - Enable error-correcting-code memory during initialization

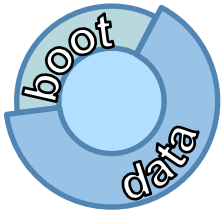


Wumbo GNU/Linux

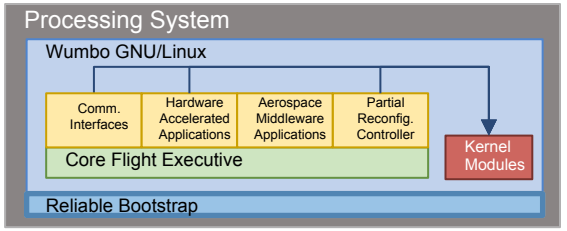


- Goal
  - Develop reliable and lightweight OS for flight deployment on Zynq
- Approach
  - Enable support for Zynq interface controllers
    - Use Xilinx patches for Ethernet, NAND, UART, static memory controller, etc.
  - Store redundant OS copies for fault tolerance
    - Contain OS root (initramfs) in small static boot archive
  - Read-only root filesystem for passive runtime protection
    - Contains only critical flight software
  - Data partition separate from boot images
    - Save non-critical applications, sensor, & experiment data here
  - Use BusyBox to reduce image size
    - BusyBox is a collection of many common UNIX tools in a single binary executable

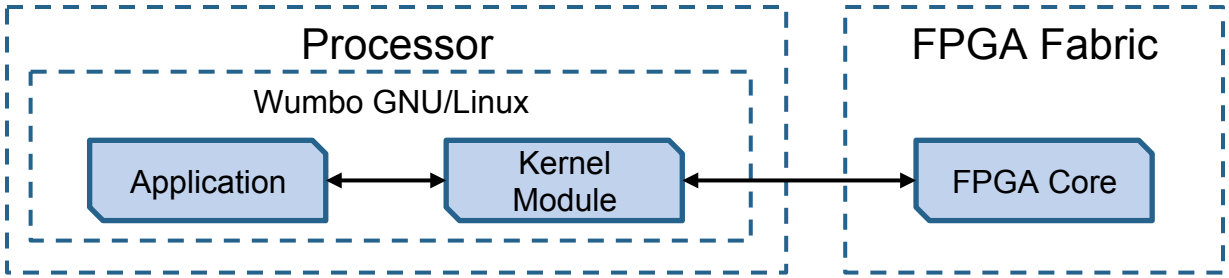




Wumbo on Hybrid Device

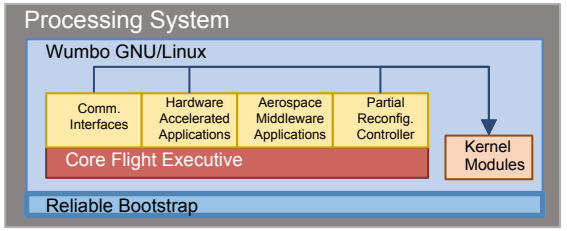


- Goal
  - Provide support for processor-FPGA communication
- Approach
  - FPGA cores accessed as peripherals via custom kernel modules
    - Kernel modules map device resources to virtual memory



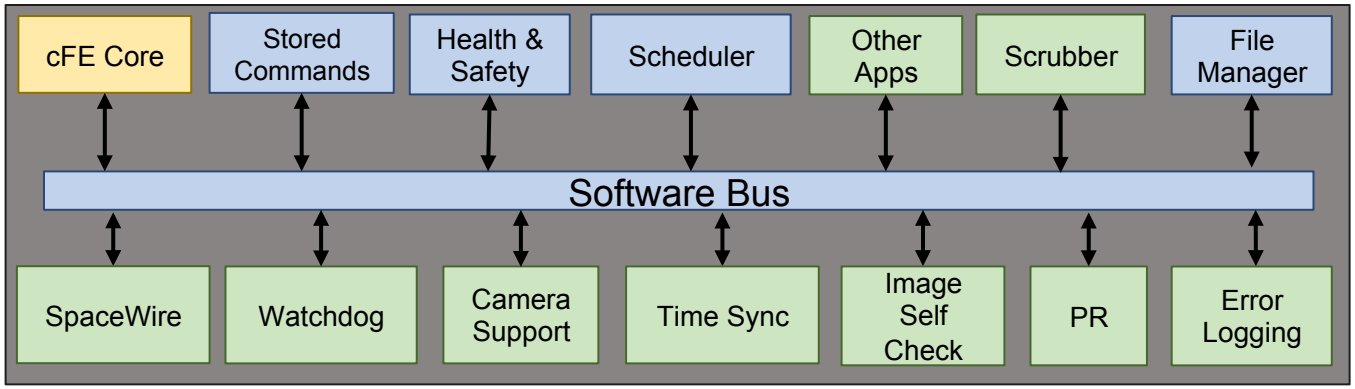
- Process
  - Develop application, kernel module for application, and FPGA core
  - Update device tree to reflect AXI address map
  - Compile code, generate bitstream, load binaries and run

Core Flight Executive

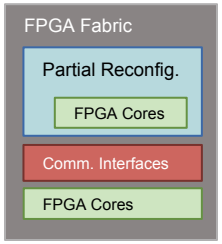


- Goal
  - Deploy system on a reusable framework for flight software
- Approach
  - Use NASA Goddard's command & data handling platform
    - Open source version available at SourceForge
  - Perform local device management, software messaging, & event generation
- Components
  - Core Flight Executive (cFE): Mission-independent software services
  - Core Flight System (CFS): Applications and libraries running on cFE

cFE Core
CFS Apps
Mission Apps

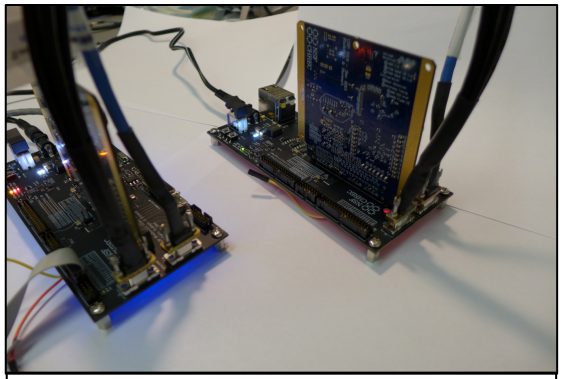


Communication Interfaces



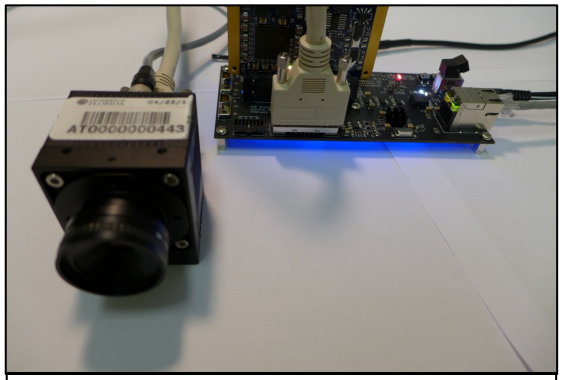
- Goal
  - Support interfaces desirable on space missions
- Approach for SpaceWire
  - Ported Virtex-5 core from NASA Goddard
  - Verified operation using link analyzer
  - Will serve as communication link with ISEM and other CSPs
- Approach for Camera Link
  - Developed in-house
  - Currently interfacing with visible-spectrum camera

SpaceWire FPGA Utilization*	
Resources	%Used
Register	4%
LUT	8%
Slice	15%
BRAM	3%
I/O	8%



SpaceWire across evaluation boards

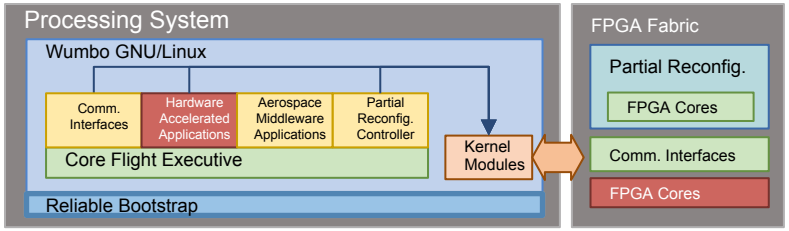
Camera Link FPGA Utilization	
Resources	%Used
Register	8%
LUT	11%
Slice	26%
BRAM	5%
I/O	7%



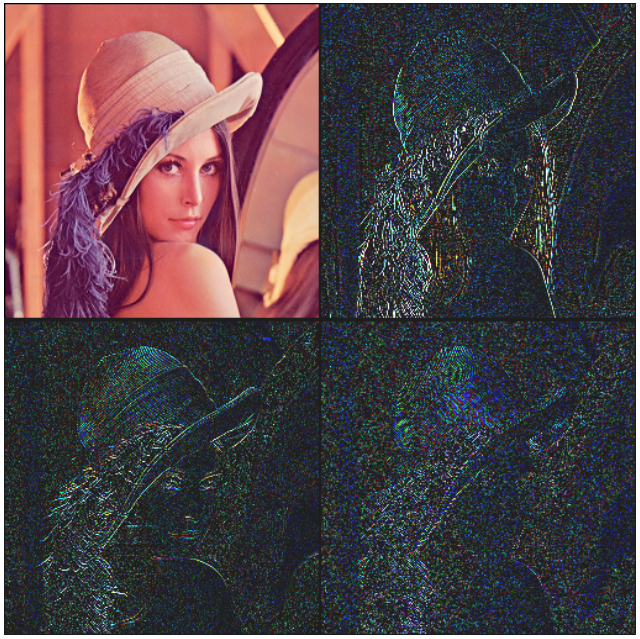
Camera Link cable connected to a CSP

*Utilization with two instances of SpaceWire core in fabric

Application Modules



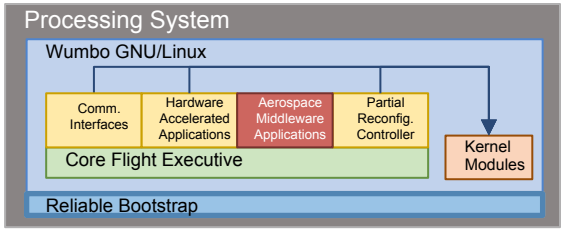
- Goal
  - Facilitate application development on hybrid systems
- Approach
  - Produce hardware-accelerated tools for integration with future applications
    - SIMD-accelerated programs using NEON engine on dual Cortex-A9 cores
    - FPGA cores
- Modules currently in development
  - JPEG2000
    - Lossless image compression
  - Singular Value Decomposition
    - Noise reduction and data analysis
  - 2D Convolution
    - Useful for various image processing and computer-vision algorithms

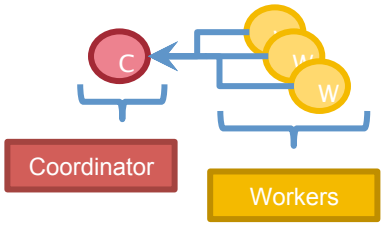


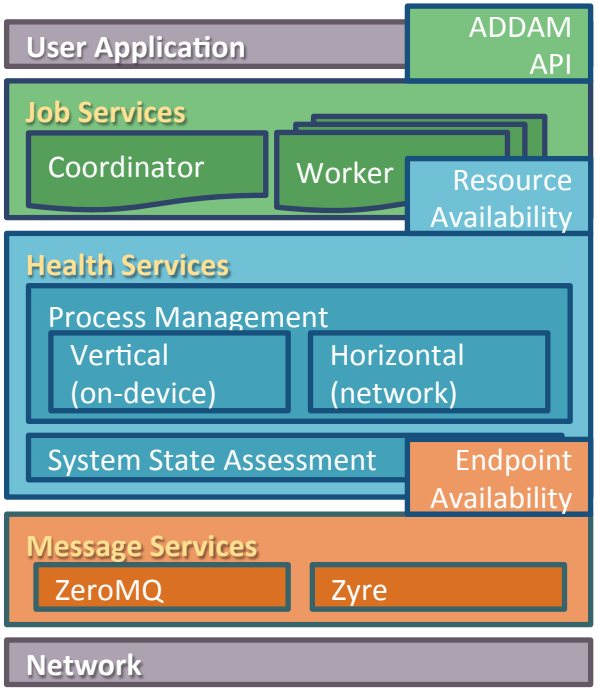
2D Cohen-Daubechies-Feauveau 5/3 wavelet transform applied to Lena, a component of JPEG2000
(brightness/contrast adjusted)

Aerospace Middleware

- Goal
  - Reliably run applications on parallel & distributed systems in hazardous environments
- Approach
  - Adaptive Distributed Dependable Aerospace Middleware: ADDAM
    - Spanning multiple cores & devices
    - Self-recovering system of agents adopt roles of coordinator or worker as needed
  - Job Services
    - Manage job replication, failover of user's job
    - Configurable fault-tolerance scheme
  - Health Services
    - Report resource availability
    - Maintain quorum of agents
  - Message Services
    - Perform endpoint discovery & availability broadcast

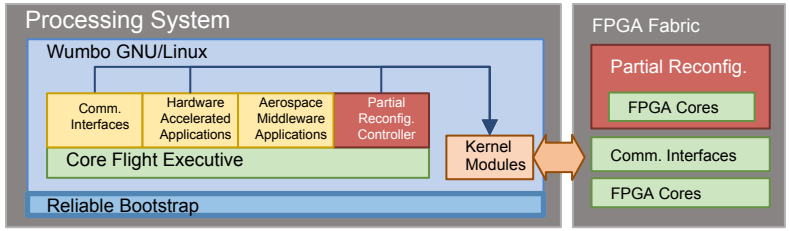




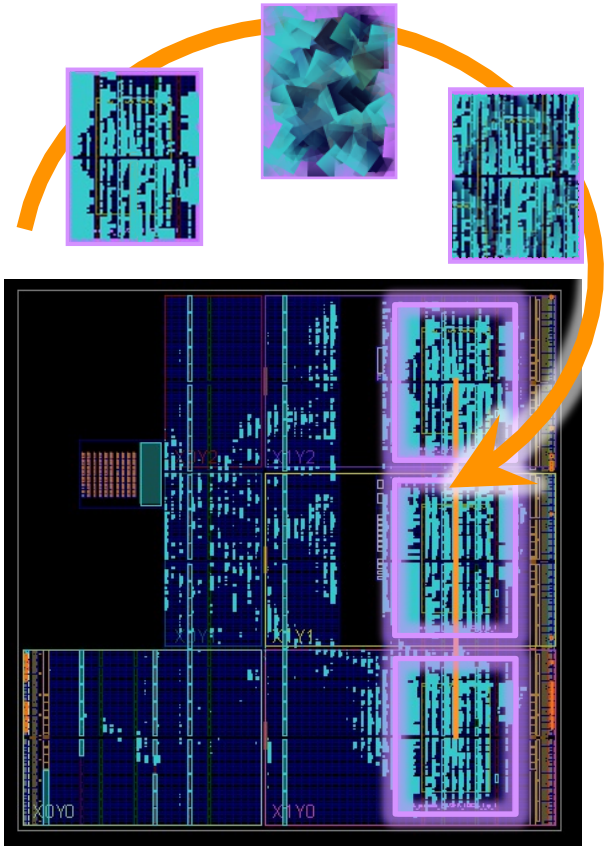


ADDAM Services Hierarchy

Partial Reconfig.



- Goals
  - Allow FPGA modification of specialized sections of reconfigurable logic at runtime
    - Load designs without reconfiguring entire FPGA
    - Selective application loading
  - Reduce vulnerable configuration area & power consumption
  - Extend post-mission operation and capabilities
- Approach
  - Incorporate reconfigurable fault tolerance
    - Replicate designs for increased tolerance
  - Dynamically combine and switch between multiple fault-tolerance techniques
    - Adjust based on environmental factors



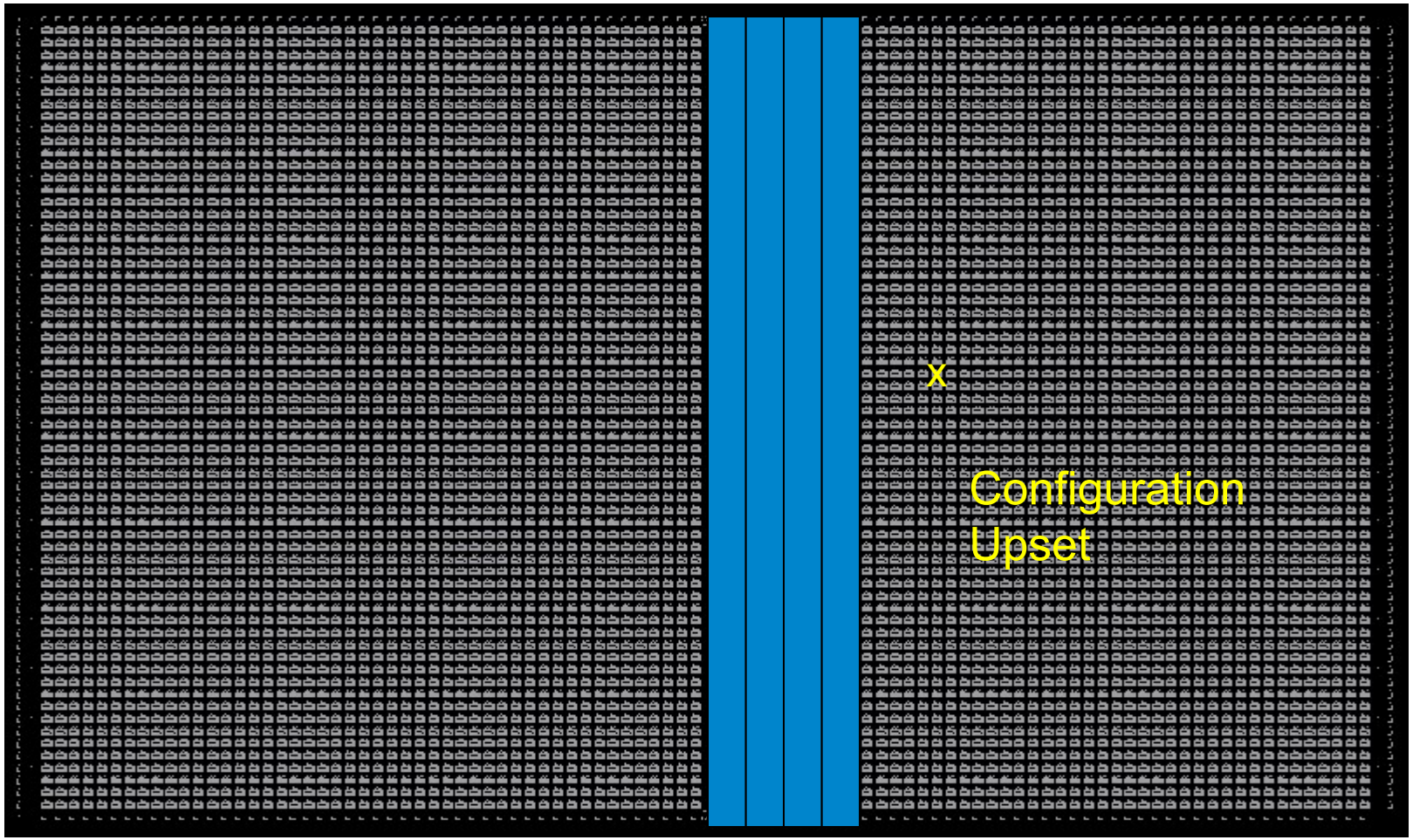
Zynq-7020 FPGA floor plan
w/ purple-outlined PR regions

Configuration Scrubbing and Processor Logging

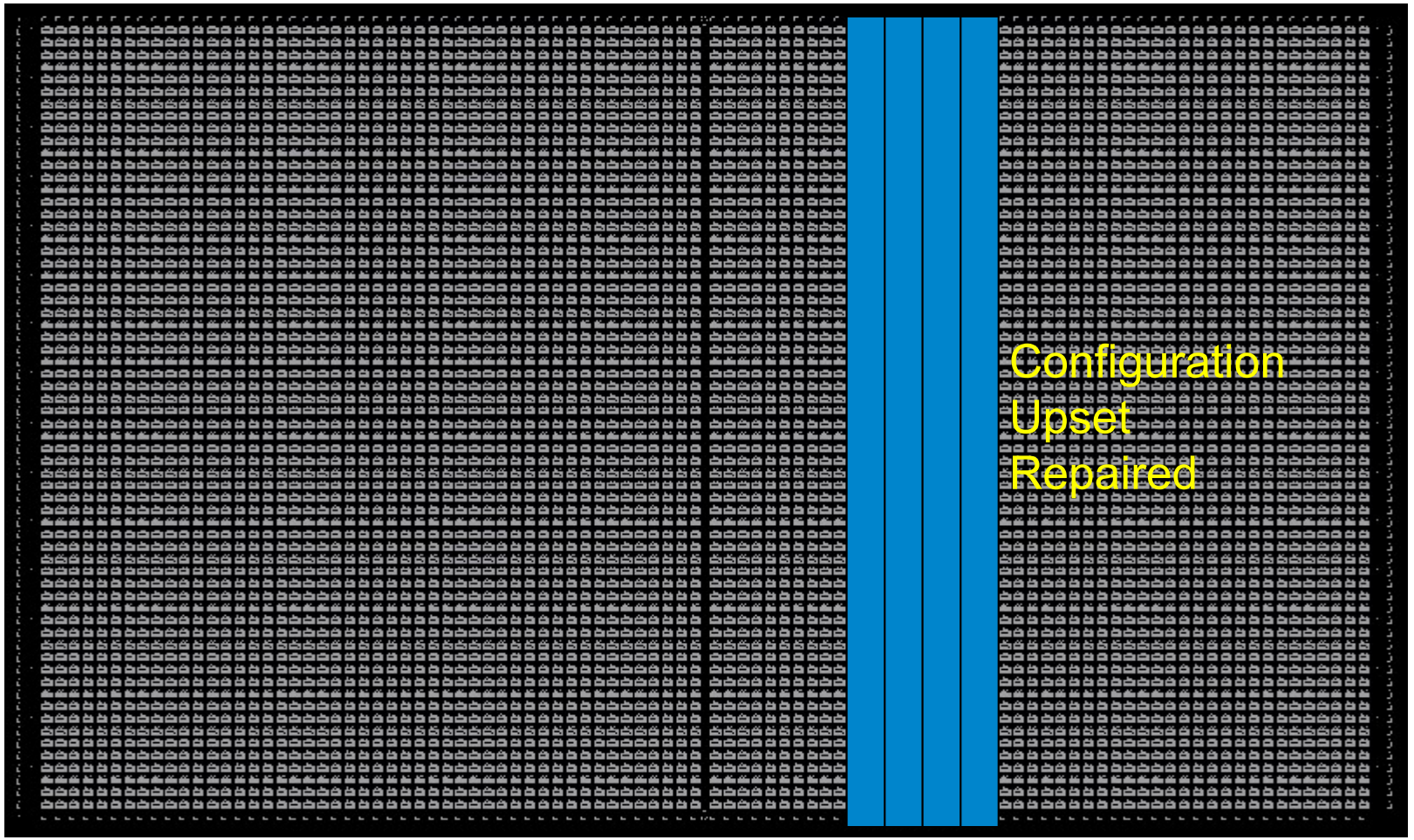
CSP PS Configuration Scrubbing

- Single Event Upsets can *modify* internal FPGA configuration memory
  - Modify Programmable Logic (PL) Behavior
  - Change internal PL State (Flip-Flops, BRAMs, etc.)
- Repair upsets within FPGA configuration memory
  - FPGA “Readback” used to determine memory state
  - Partial reconfiguration used to replace upset memory
- Scrubbing improves PL reliability
  - Prevent accumulation of PL configuration SEUs
  - Prevent more than one upset (preserve TMR behavior)

PL Configuration Scrubbing

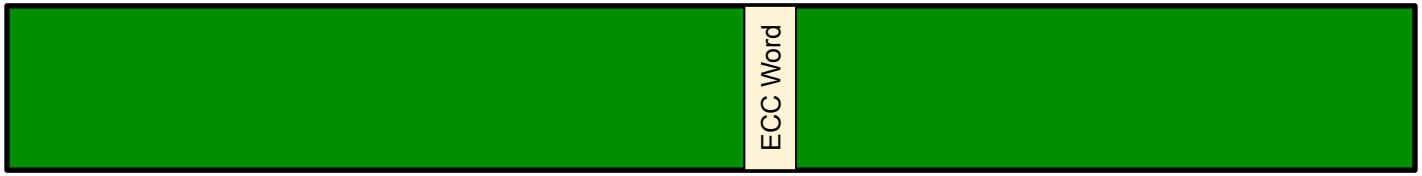


PL Configuration Scrubbing



Z-7020 Configuration Memory

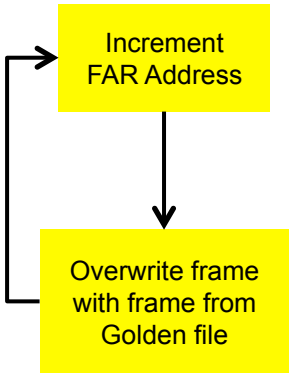
- 7 Series Configuration Data Format
  - 101 words per frame, 32-bits per word (1 ECC word)
  - 3,232 bits/frame



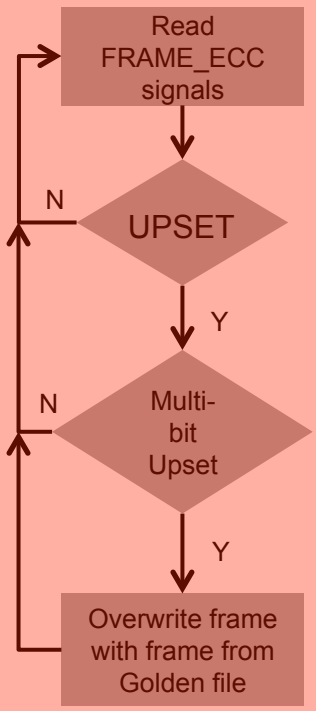
- 10,008 Total Configuration Frames (30.8 Mb)
  - 7,692 Type 0 Frames (logic/routing)
    - Only Type 0 Frames need to be stored for scrubbing
  - 2,316 Type 1 Frames (BRAM)
    - BRAM ECC and BRAM scrubbing used to protect BRAM data
- Two Files Needed: raw bitstream and mask file

FPGA Scrubbing Strategies

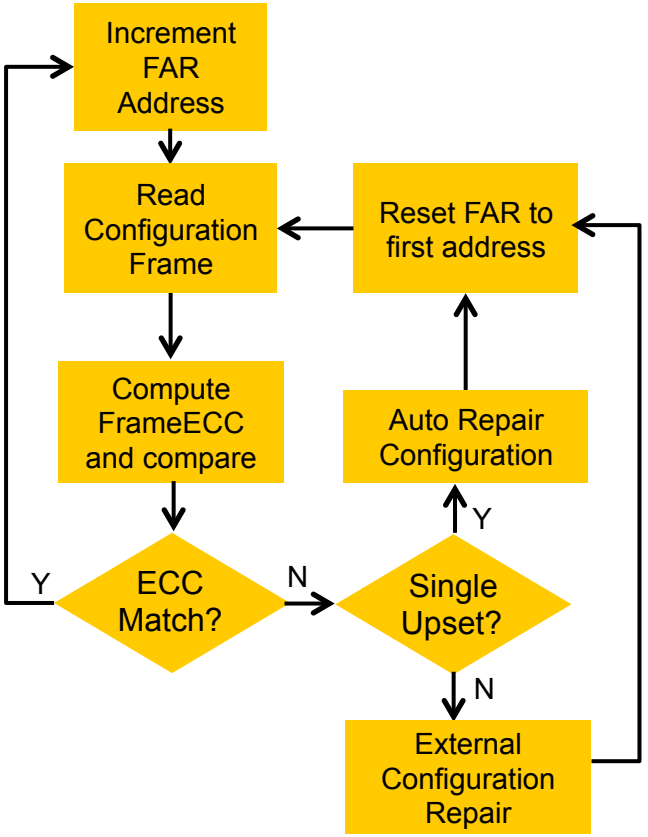
Blind Scrubbing



Readback Scrubbing

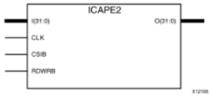


FrameECC Scrubbing



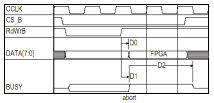
Series 7 Configuration Ports

ICAP



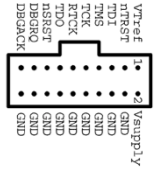
- Pros:
  - Limited external hardware required
  - IP available (SEM-IP)
  - Fast, programmable
- Cons:
  - Susceptible to single point failure
  - Requires internal mitigation (TMR?)

SelectMAP

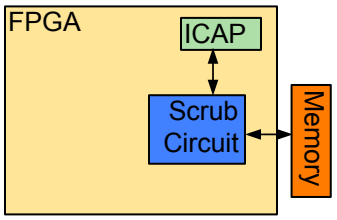


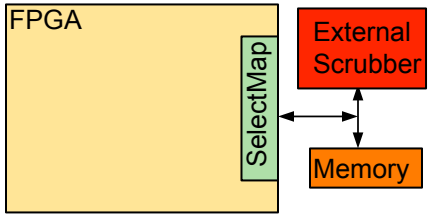
- Pros:
  - Well understood and tested
  - Fast, relatively simple interface
- Cons:
  - External protected support circuitry required

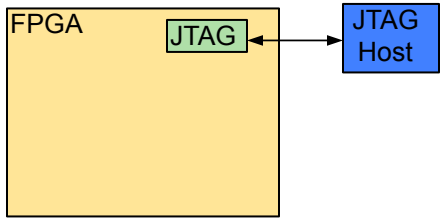
JTAG



- Pros:
  - JTAG widely supported
  - No additional hardware required
  - Supports user JTAG comm
- Cons:
  - Slow (bit serial, slow clock)
  - Difficult programmer interface







Processor Configuration Access Port (PCAP)

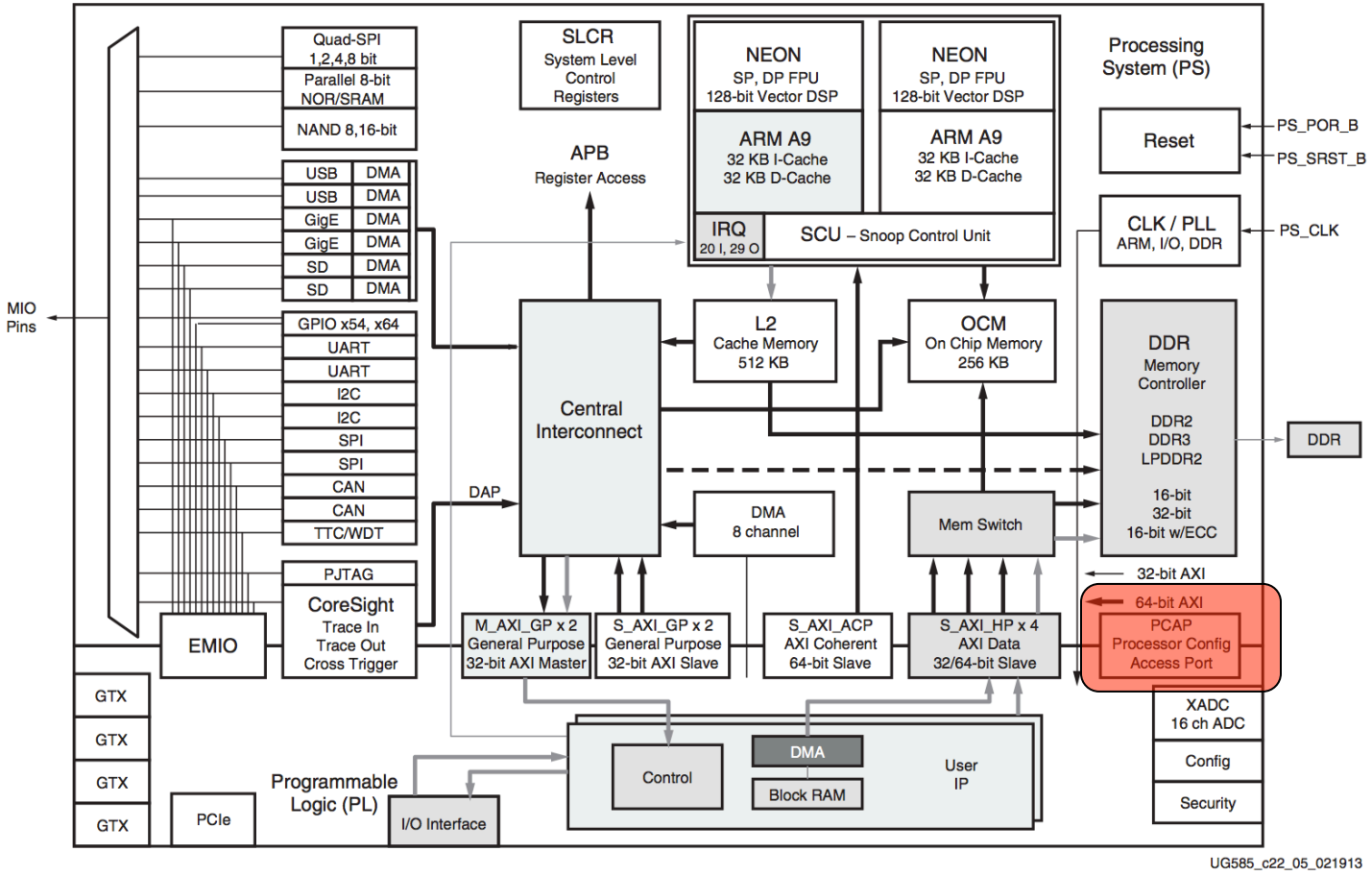
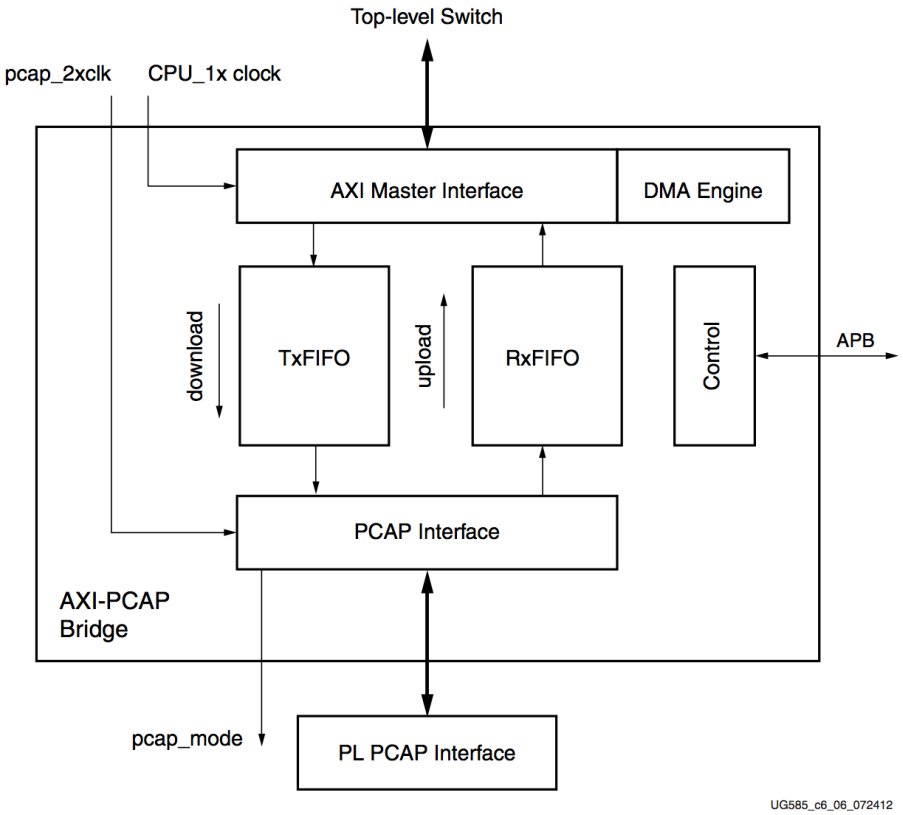


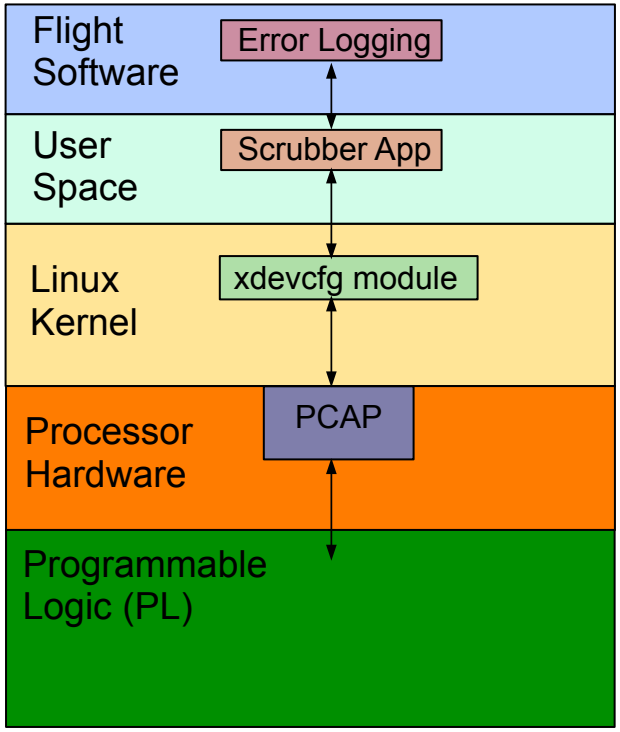
Figure 22-3: **Example High-Performance (HP) DMA Topology**

Processor Configuration Access Port (PCAP)

- PCAP port provides access to PL configuration from software
  - AXI Master Interface
  - DMA support, TX and RX Fifos
- Programmable Logic (PL) can be configured directly by Cortex A9 processors
  - Configuration done by a DMA transfer
  - Configuration bitfile in main memory
- Pros:
  - Easy to implement (software only)
  - No external hardware
- Cons:
  - Only as reliable as the Cortex A9 reliability



CSP Linux Scrubbing Architecture



- Operates within Wumbo Linux operating system
  - xdevcfg kernel module
    - "Owns" address space of PCAP
    - Transfers configuration/readback commands to PCAP via DMA
- Scrubber application
  - Stores bitfile/maskfile
  - Opens and interacts with "xdevcfg"
  - Implements "Readback" scrubbing through calls to xdevfg
  - Provides logging messages

Linux and Fault Handling/Logging

watchdog.c

```
while (true) {
    reset_watchdog();
    sleep();
}
```

If the kernel or a user process hangs, then watchdog will power cycle board

Errors on stdout
Kernel prints at high log level which appears on stdout over UART

user space

kernel modules

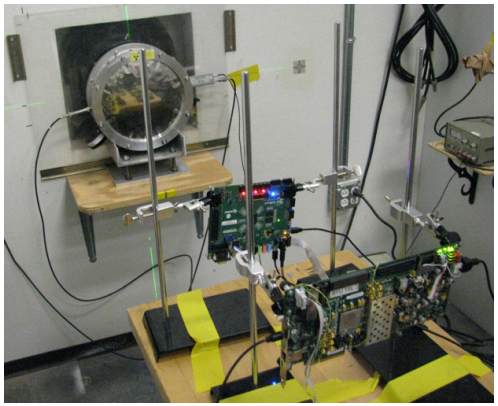
Watchdog Module
/dev/watchdog
Cannot be stopped or exited

Interrupt Handlers
L1, L2, OCM parity errors
Illegal instructions, unused interrupts, etc.

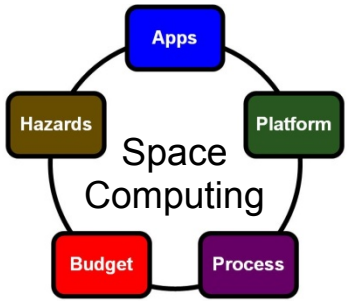
Error Detection And Correction (EDAC)
ECC on DDR Memory

Related CSP Scrubbing/Logging

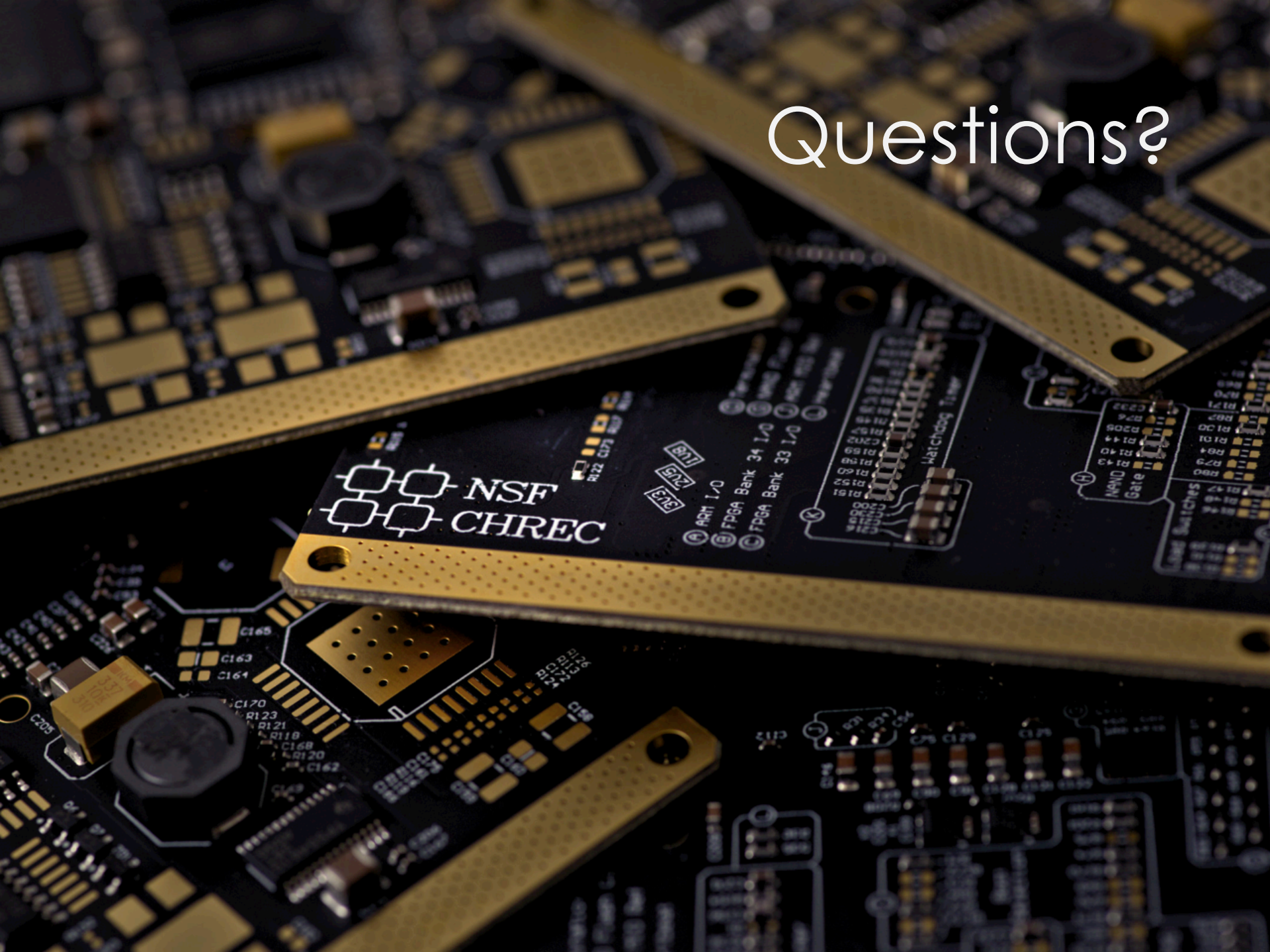
- Fault Injection
  - Scrubber easily supports fault injection
  - Random and targeted injection
- Client/Server Scrubbing Software Architecture
- Support for Partial Reconfiguration (PR)
- Frame ECC Scrubbing
- Upcoming Neutron Radiation Test
  - TRIUMF, Vancouver, BC (June 24-26)
  - Validate Linux-based PCAP Scrubber
  - Validate Linux Fault Tolerant Extensions



Conclusions



- **Major challenges lie ahead**
  - Escalating app demands in harsh environment
  - Tightening constraints of platform, budget, process
  - ***Necessitates adeptly doing more with less***
- **CHREC Space Processor**
  - Focus upon reconfigurable space computing
    - Adaptive (performance, reliability, power) for mission needs
  - Focus upon multifaceted hybrid computing
    - Agile mix of COTS + RadHard + FTC; fixed & reconfigurable
  - Focus upon scalable building blocks
    - Address needs of small, large, & clustered spacecraft


Questions?
NSF
CHREC

More information? www.chrec.org