Virtual Platform Based Development Environments for Low Power, Mixed Level Safety Critical System

D. Graham and L. Lapides, Imperas Software Ltd.

S. Schreiner and K. Grüttner, OFFIS

Acknowledgement:
This work was partially funded by the SAFEPOWER project under the EU Horizon2020 programme

http://safepower-project.eu

The authors give their thanks to all the organizations that participated in SAFEPOWER, both universities and commercial companies

Agenda

- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

Agenda

- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

Safety and Security in Embedded Systems

- More processors per SoC is being driven by demand for bigger, faster, more efficient systems
- Initial architectures just consolidated smaller SoCs into a larger SoC with minimal sharing of resources
  - This satisfies performance and domain isolation (for safety and security critical uses) requirements
  - Power consumption becomes a real issue, in terms of reliability and cooling costs

Mixed Criticality Systems

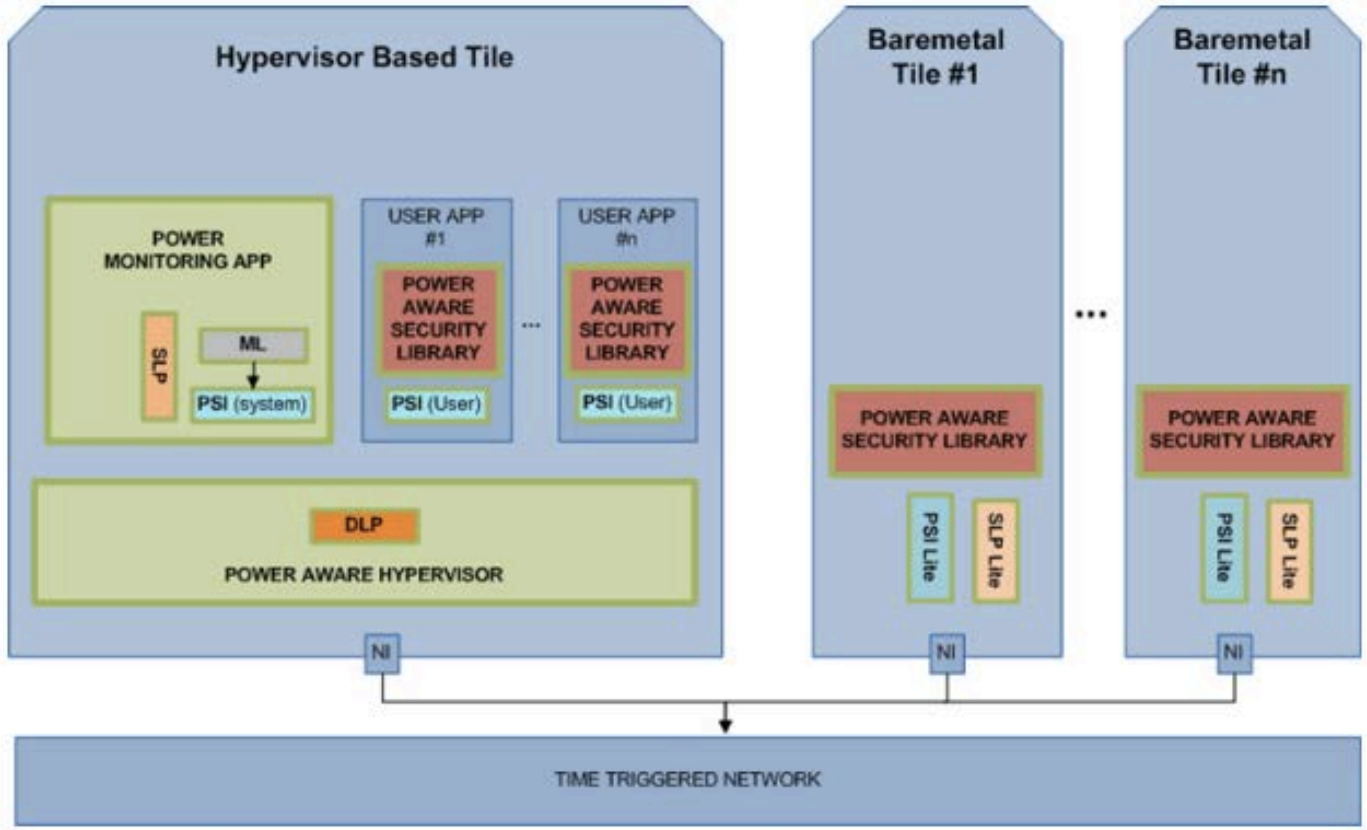
- Definition: an embedded system comprised of hardware, operating system (OS), middleware services and application software with multiple levels of criticality
- Mixed criticality systems are seen in multiple market segments including automotive, avionics, industrial controls, medical electronics
- The architecture of many of these systems uses the OS and/or a hypervisor to realize partitioning between the domains with differing levels of criticality
  - In this situation, the OS or hypervisor implementing the partitioning needs to be certified at the same criticality level as the most critical application

Agenda

- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

SAFEPOWER Reference Architecture

- SLP: Static Low-Power block
- PSI: Power Services Interface
- Hypervisor partitions for Power Monitoring App, User App #1, User App #2, …
- Tile “partitions” for Bare-metal Tile #1, Bare-metal Tile #2, …



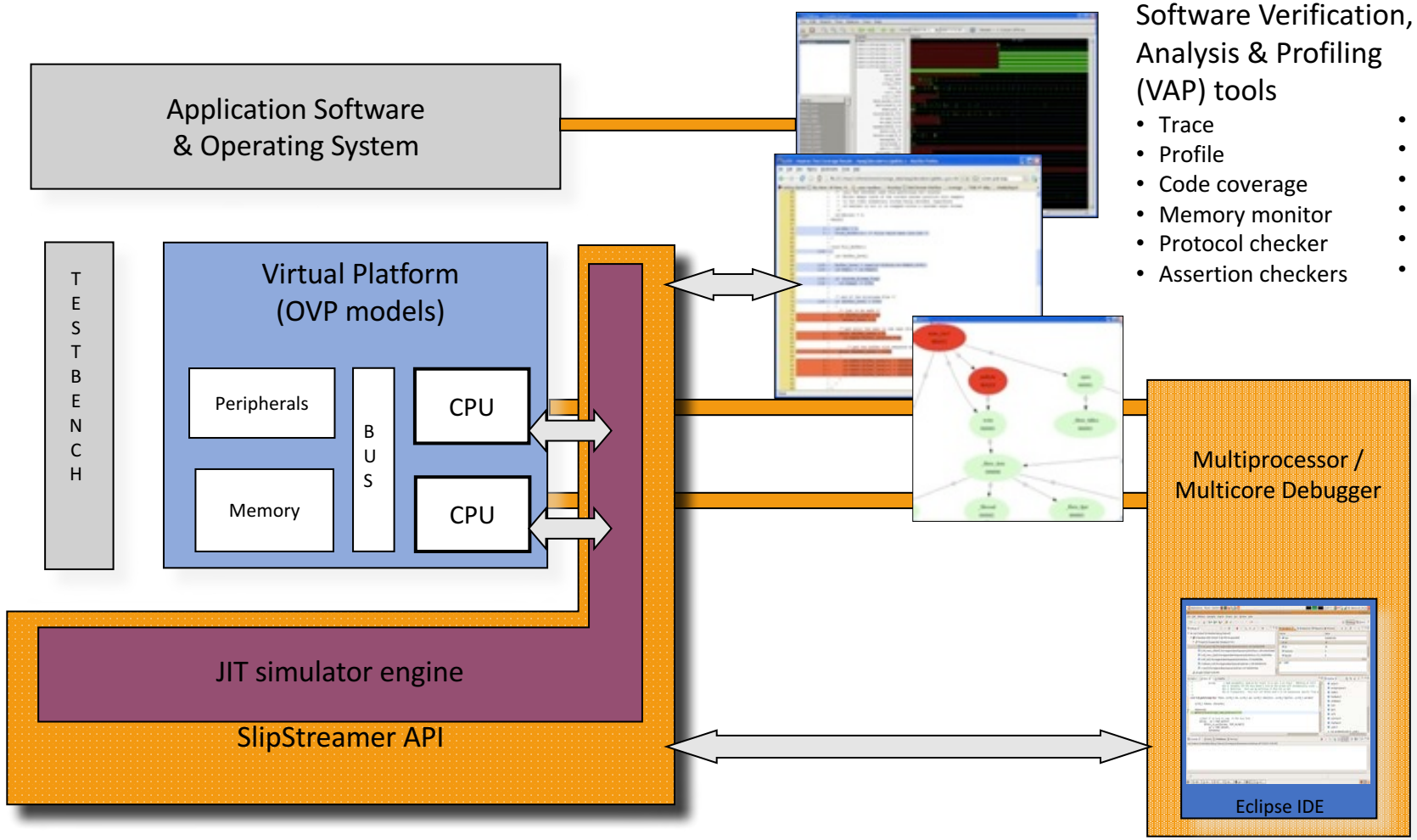
SAFEPOWER Reference Architecture

- Tile-based architecture
- Connected via time-triggered Network on Chip (NoC)
- Tiles are managed by a Type-1 hypervisor
  - Allows for an arbitrary amount of user partitions
  - Low-Power Monitoring Partition: Special partition on the hypervisor for monitoring and controlling power management services
  - Includes time-triggered task management
- Bare-metal tiles are not directly managed by the hypervisor
  - Connected to the time-triggered NoC
  - Implements a light version of the hypervisor application interface
  - Time-triggered behavior is executed based on a pre-computed communication schedule that triggers the message injection times
- Achieves both the spatial isolation and temporal independence required in safety standards such as IEC-61508
- Time triggered architecture provides deterministic scheduling of software tasks, with Worst Case Execution Time (WCET) analysis supporting the achievement of timing requirements

Agenda

- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

Imperas Environment for Embedded Software Development, Debug & Test



Software Verification, Analysis & Profiling (VAP) tools

- Trace
- Profile
- Code coverage
- Memory monitor
- Protocol checker
- Assertion checkers
- OS task tracing
- OS scheduler analysis
- Fault injection
- Function tracing
- Variable tracing
- …

Virtual Platform

- Models
  - Existing models from the Open Virtual Platforms (OVP) Library used, if available
    - Models of Arm, MicroBlaze processors
    - Various non-processor models
  - New models built using OVP APIs as required
    - Network Interface (NI)
    - Various peripheral components
    - Hierarchical, parameterized platform
  - Models are open source (http://www.ovpworld.org)
- Simulator product is Imperas M*SDK
  - Instruction accurate simulator engine (~250 million instructions per second performance)
  - MultiProcessor Debugger (MPD) for platform-centric debug
  - Verification, Analysis and Profiling (VAP) tools

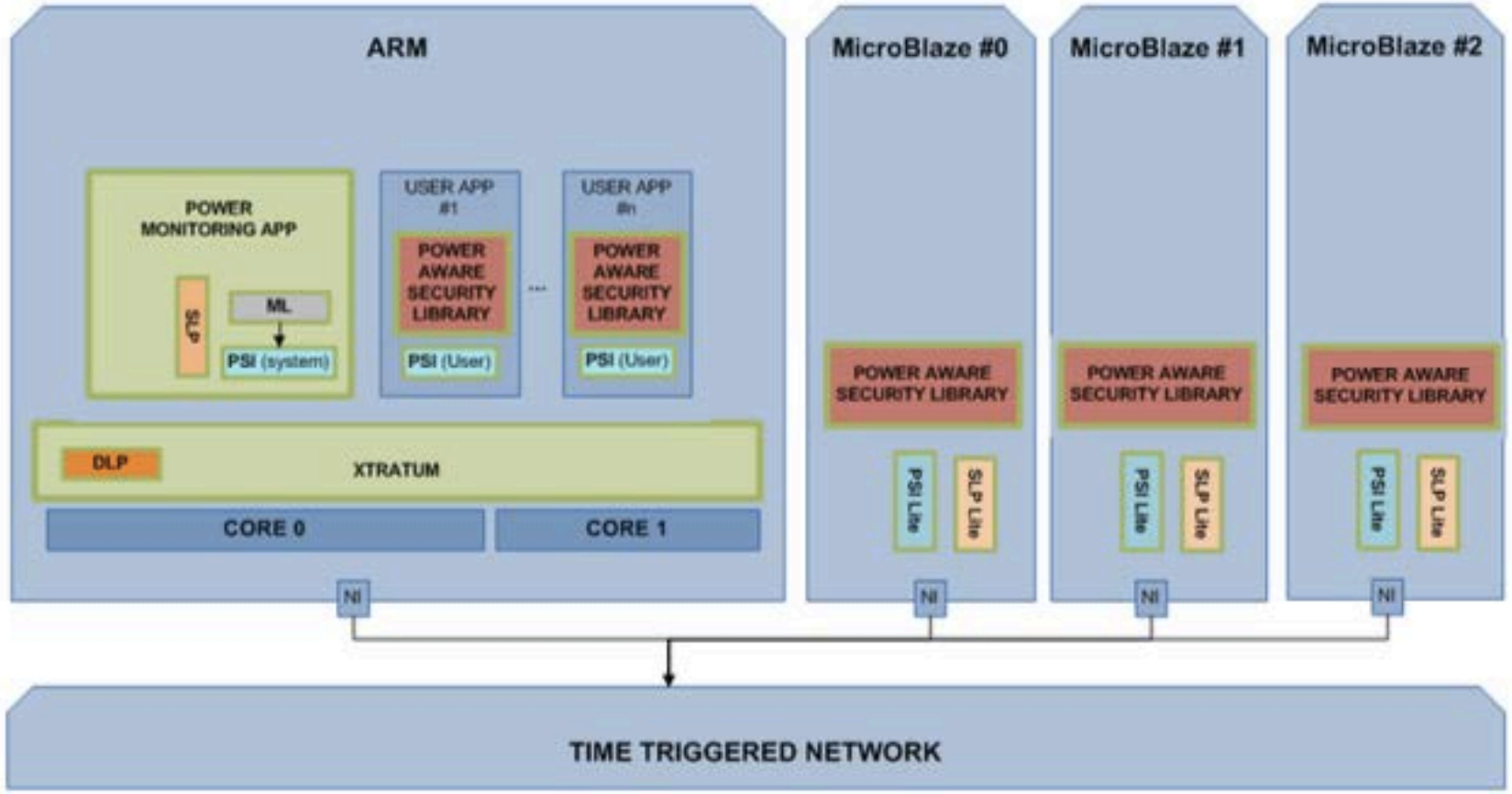
Agenda

- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

SafePower Platform

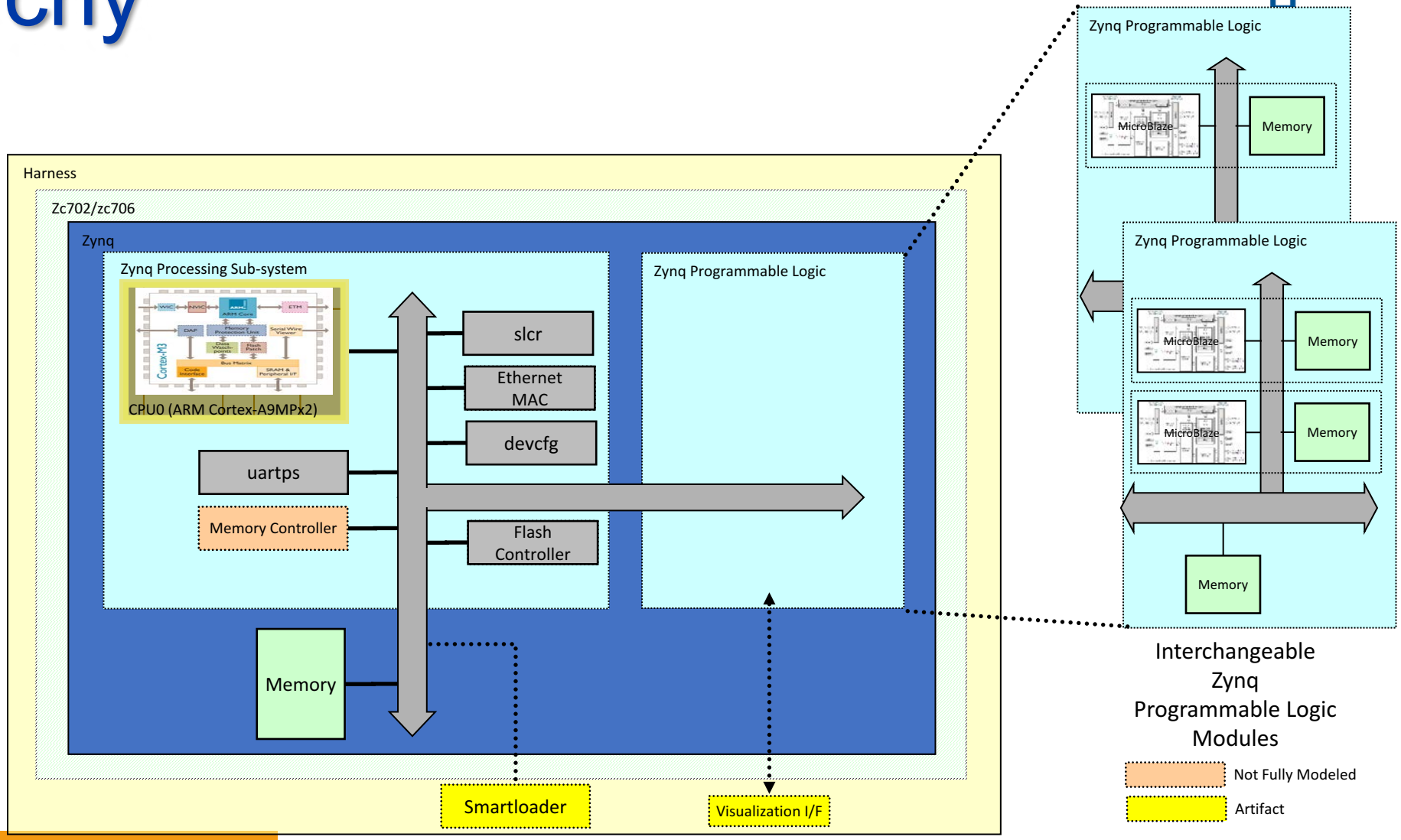
- Objective was to build the platform as both real hardware and a virtual platform
- Based upon the Xilinx Zynq 7000 board
- Processing System (PS)
  - Static hardware block including ARM Cortex-A9MPx2 processor
- Programmable Logic (PL)
  - Define any components and interconnects
    - MicroBlaze processors
    - Network-on-Chip nodes and interconnect
    - Memory
  - Can create and dynamically load any PL definition
- Interconnect
  - Fixed Connectivity between PS and PL
    - Address-mapped data and GPIO
- Power Control and Monitor
  - Power monitoring devices
    - Set voltages and obtain feedback of current values
  - Clock control
    - Dynamically change processor clock frequency – Dynamic Voltage and Frequency Scaling (DVFS)

Hypervisor is XtratuM from FentISS

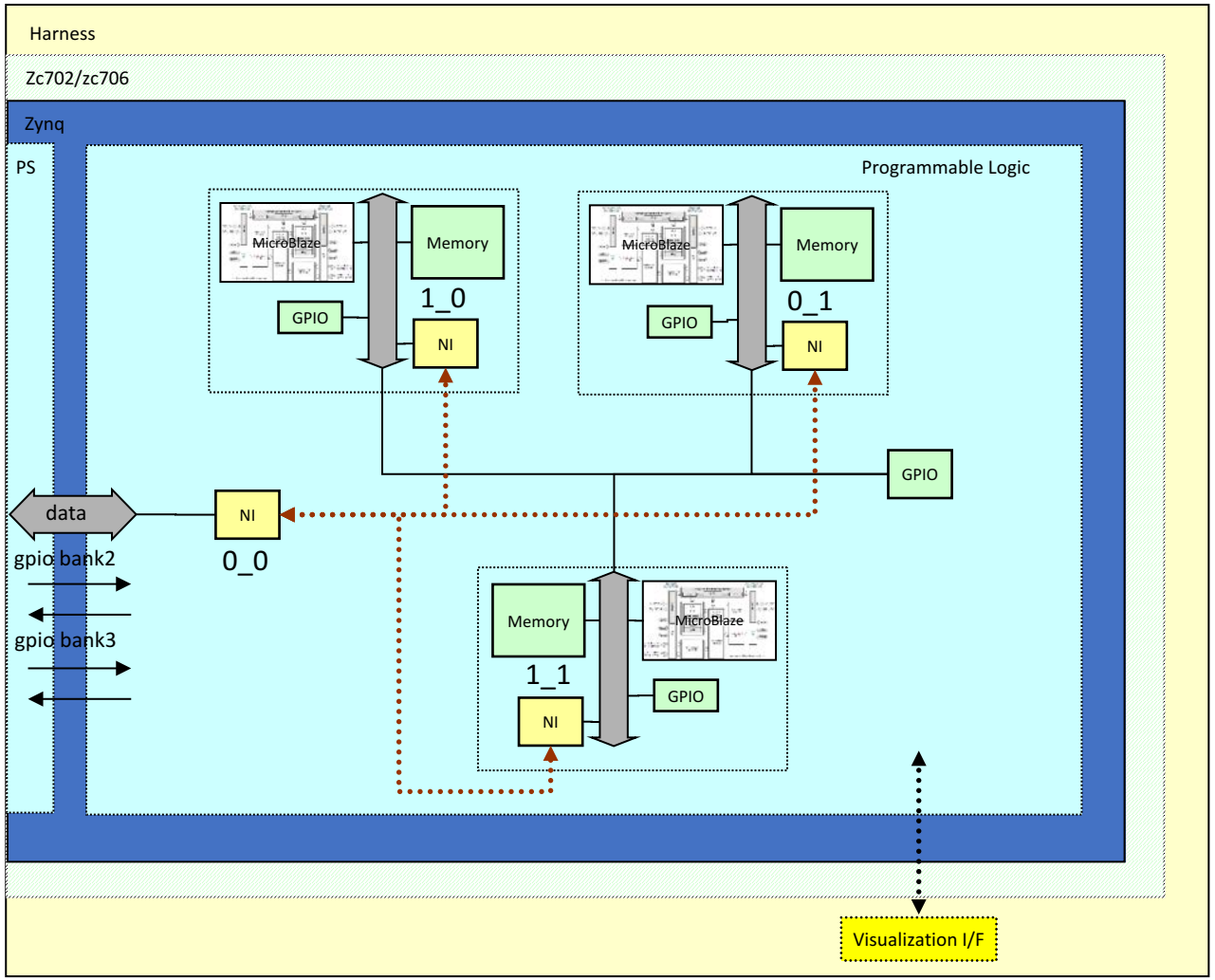


- Supports time-triggered task management
- Supports special partitions for system-level apps such as power monitoring and management

Xilinx Zynq Virtual Platform Hierarchy



SafePower Virtual Platform NoC



NoC (Packetnet Network)

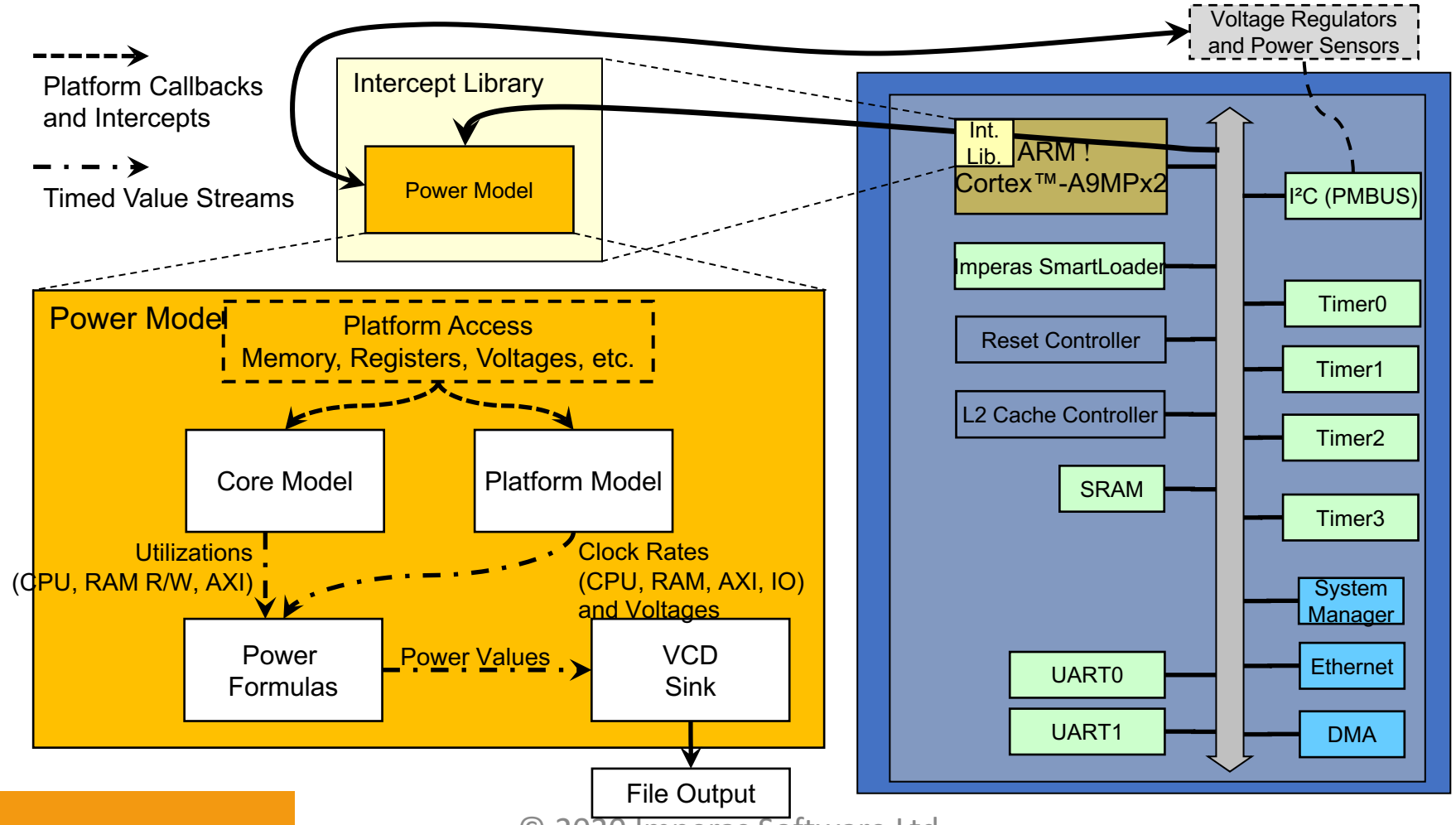
Agenda

- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

Virtual Platform Challenges

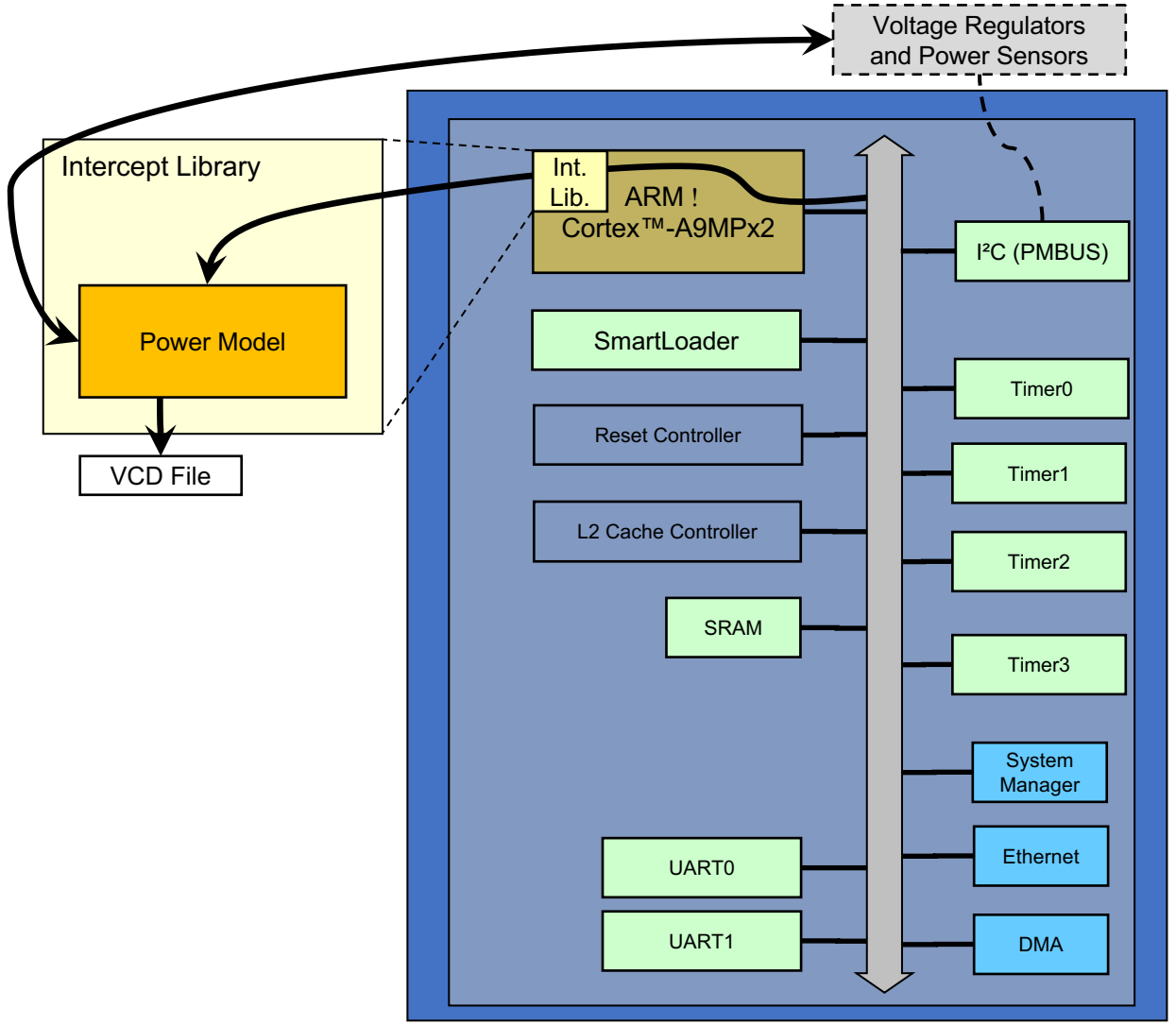
1) How to model DVFS
2) Fault injection testing environment
3) Support for time-triggered system

1) DVFS Implementation Uses Power Intercept Library



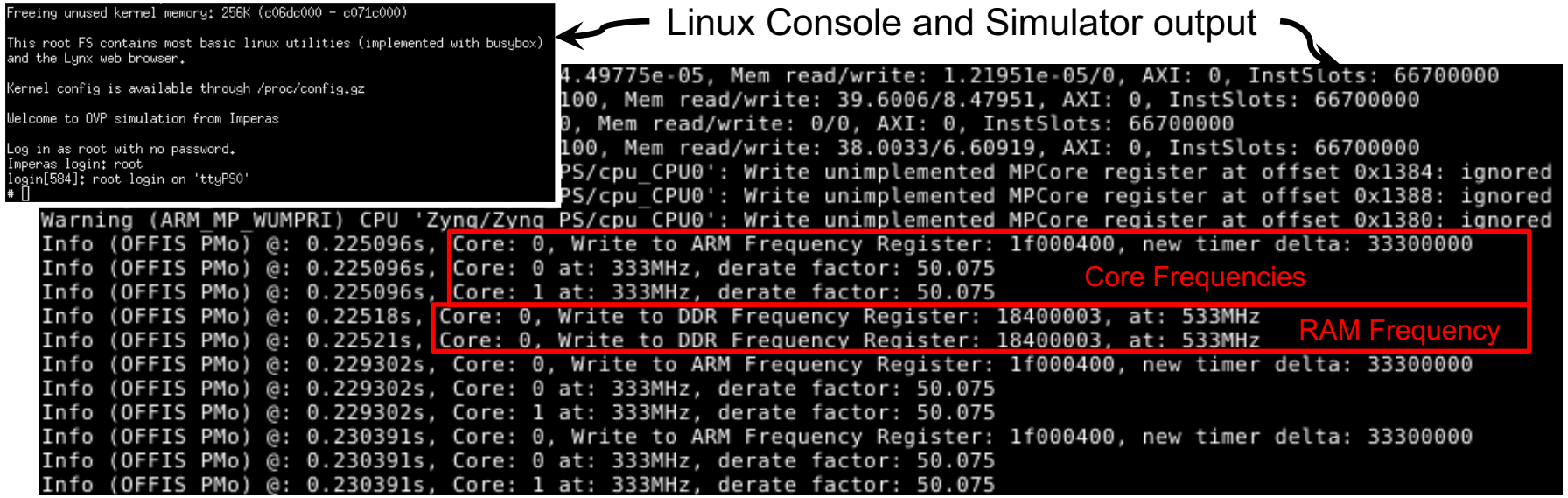
Virtual Power Sensor

- **Power Sensor** is represented by an intercepted I²C interface
- **Power Model** gets new voltage values and returns power values
- **That means:** Next to the frequency configuration the application is able to configure the voltages and to request the present power values



Executing Linux in VP with Attached Power Model

- Virtual Platform (VP) executes Linux and Power Model recognizes changes:
  - Core frequencies are reconfigured to 333MHz
  - RAM frequency is configured to 533MHz
- Power Model reconfigures MIPS rate of both cores



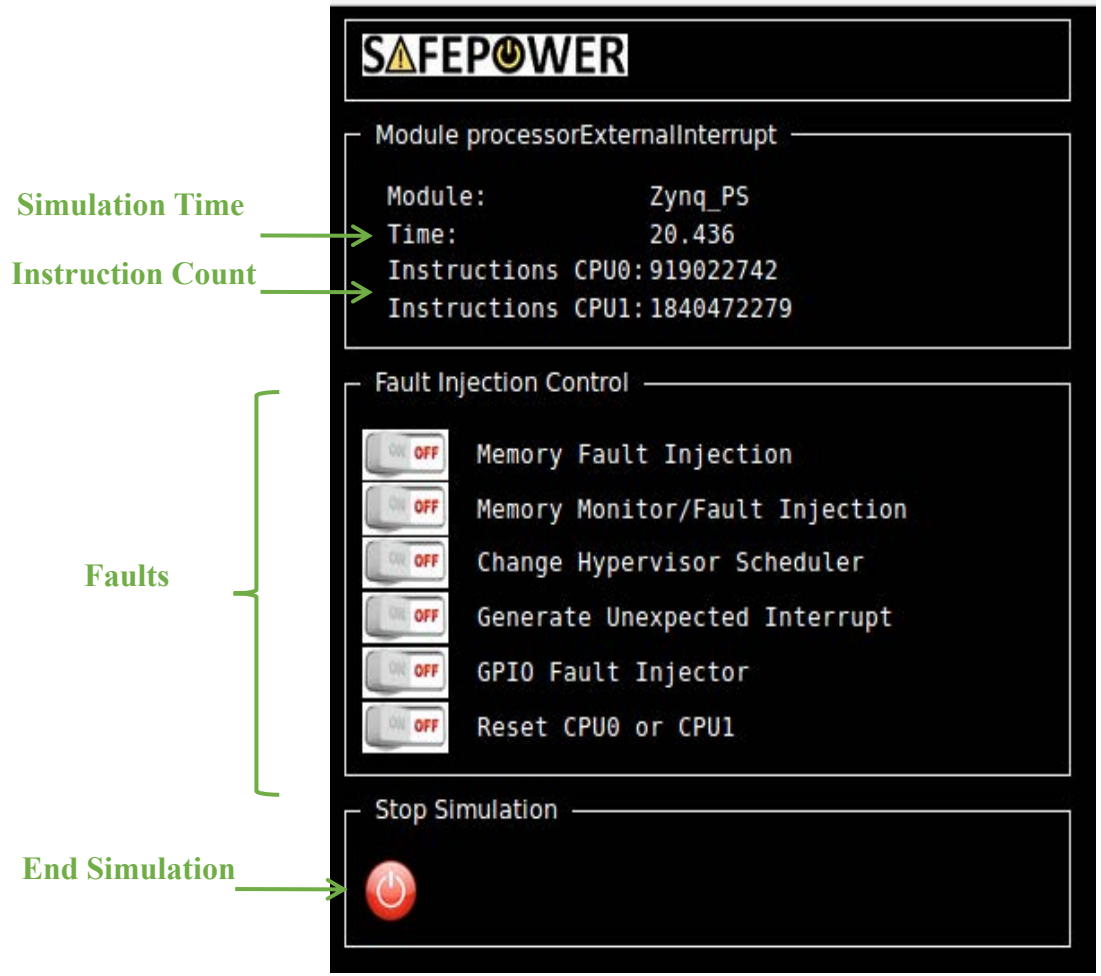
2) Fault Injection

- Fault injection framework should
  - Include different faults and fault campaigns
  - Apply simulation time to trigger injection of faults

- Faults implemented
  - Memory corruption
  - Memory monitoring and corruption
  - GPIO corruption
  - Switch partition scheduler
  - Reset CPUx
  - Uncontrolled interrupt

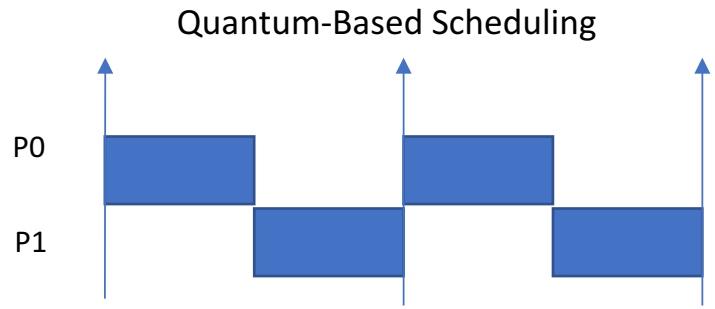
Fault Injection Execution Modes

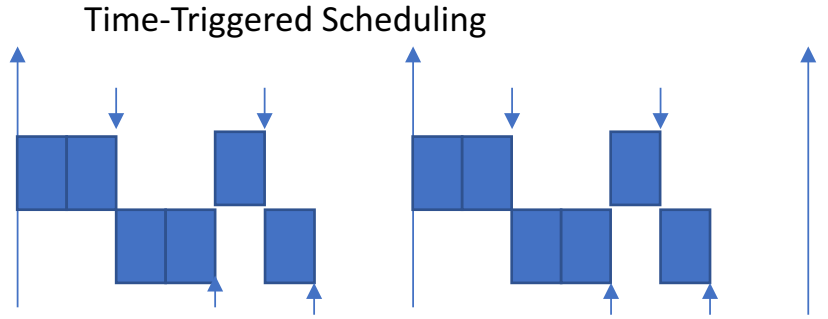
- Intercept library
- Intercept library + configuration file
- Sequential execution of faults
- HTML based user interface



3) Simulation: Time-Triggered Scheduling

- Default simulation scheduler
  - Round-robin scheduling of processor execution
  - Functionally correct but limited timing reference
- SAFEPOWER is a time-triggered system
  - Timing synchronisation defined statically
  - Time triggers do not align with simulation quanta
- Solution: Develop time-triggered scheduler
  - Applications are executed until reach next event
  - Execution scheduled to complete “work”
    - Detect points in application for synchronisation events





Agenda

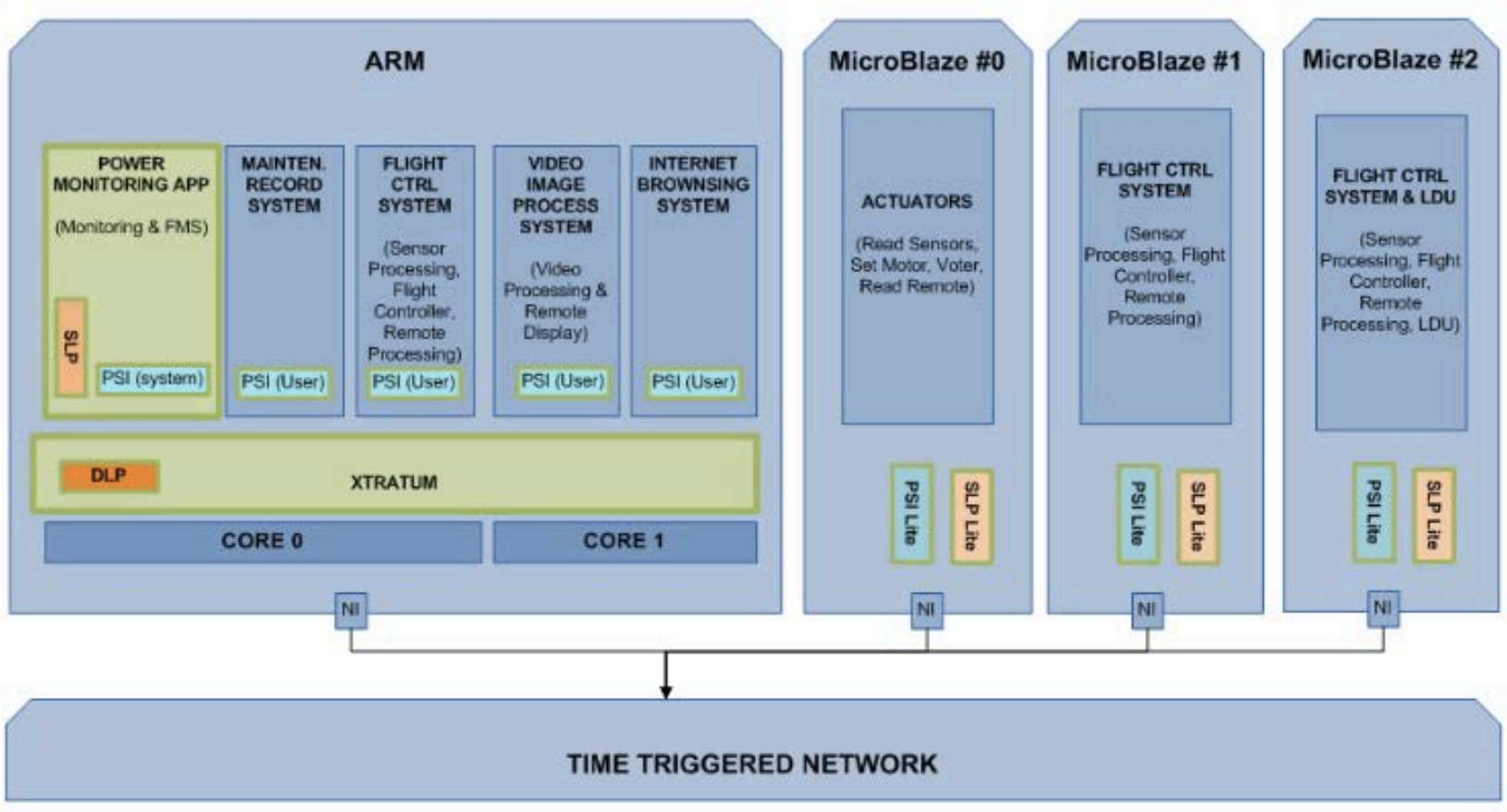
- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

Evaluation

- Railway and avionics use cases implemented
- Evaluate the functionality and timing with and without low power techniques on both the virtual platform and hardware platform
- Objectives
  - Demonstrate that the software components interact correctly with each other and with the hardware
  - Verify that the entire system complies with the requirements – functional and extra-functional, also when including low power techniques – of the use case specification
  - Show that power savings can be reached for safety critical applications in different domains

SAFEPOWER Avionics Use Case

- Hypervisor partitions for Power Monitoring App, Maintenance Record System, …
- Tile “partitions” for Actuators, Flight CTRL System, Flight CTRL System & LDU



Virtual Platform Results

- Virtual platform results were equivalent to hardware based results for the tests that could be run on both platforms
  - Virtual platform uses the same binaries, and so can be utilized in test and certification of safety critical applications

- Virtual platform provided benefits over the hardware platform for development, debug, analysis and verification of software applications

1) Execution control: Simulation is deterministic
2) Unified debug environment: Simultaneous debug of all application code executing on all processors in the platform, including access to peripherals
   - Analysis tools such as profiling, code coverage, dynamic assertions, etc. are implemented non-intrusively: no modification or instrumentation of source code required
   - Power Interface Library, implemented using Imperas SlipStreamer API (binary interception), enabled support for real time power management techniques such as DVFS within the virtual platform environment
3) Fault injection: The virtual platform provides visibility and observability, so faults can be injected anywhere in the platform, e.g. memory, processor registers, … Fault injection is implemented by an external library, so fault generation can be controlled

Agenda

- Safety and security in embedded systems
- SAFEPOWER architecture
- Virtual platform environment
- SAFEPOWER platform: Xilinx Zynq 7000
- Virtual platform challenges and solutions
- Results
- Summary

Summary

- Virtual platform was used for development of mixed-criticality system
- Several challenges were encountered and overcome:
  - Power modeling for power monitoring and management
  - Fault injection testing
  - Support of time-triggered system
- Virtual platform environment had equivalent results to hardware platform, with a number of advantages due to the controllability, observability, determinism and ease of automation of the software simulation

Thank you

Larry Lapides

LarryL@imperas.com