FTF | FREESCALE TECHNOLOGY FORUM 2015

Automotive Performance Analysis for Virtual Prototypes

FTF-ACC-F1260

Manfred Thanner | Senior Member of Technical Staff

J U N E . 2 0 1 5

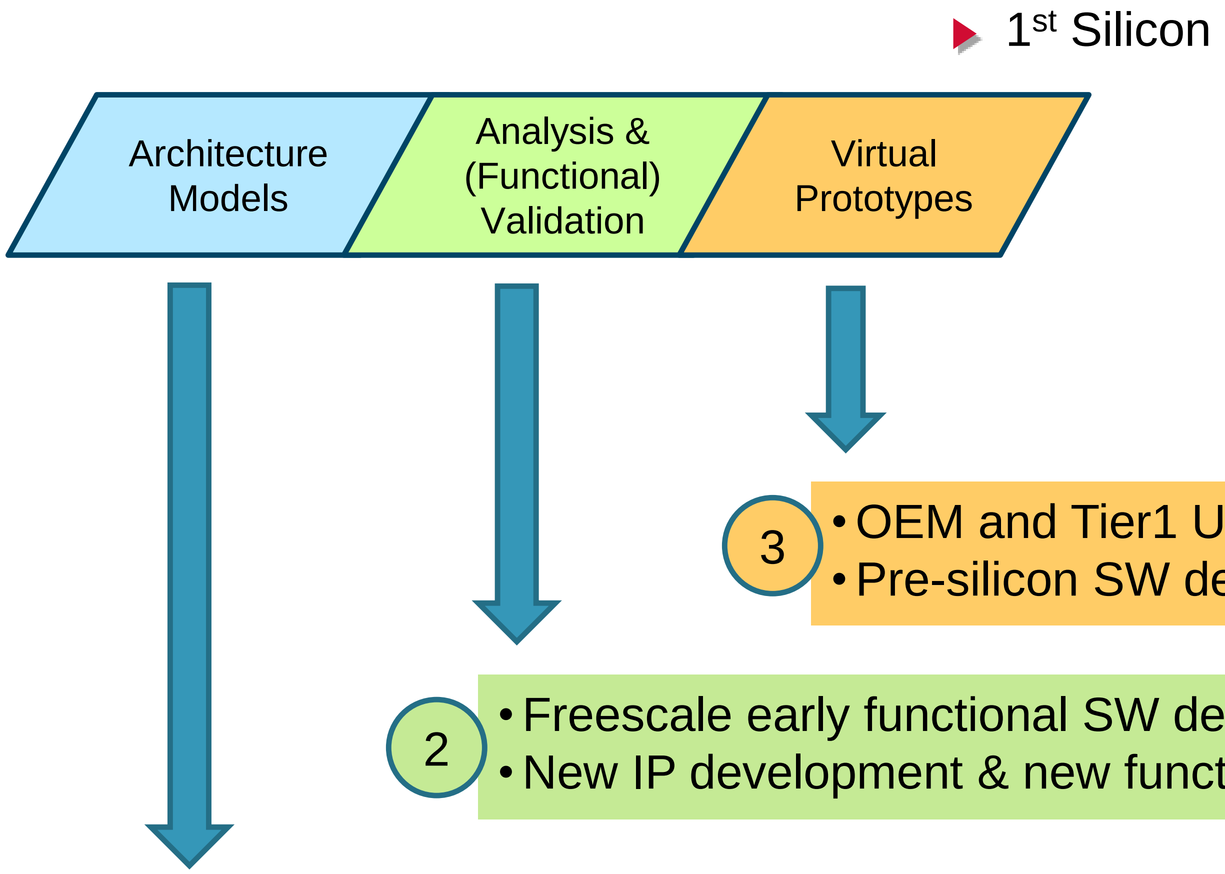
freescale™

External Use

Agenda

- Session Overview
- Use cases for Virtual Prototype
- Automotive Needs
- Model Build & Exchange
- Benchmark Categories
  - Loosely Timed
  - Approximate Timed
  - Error Injection Example
- Virtual Platform Partners
- Summary

Where are the Virtual Platforms being used



3
- OEM and Tier1 Users
- Pre-silicon SW development & Error Injection

2
- Freescale early functional SW development on accelerator IP
- New IP development & new functions (SPT, ISP, 2D-ACE)

1
- Freescale key „own interest“ to ensure correct performance targets are fullfilled, Performance based verification
- Primarily Systems and product definitions

Automotive Needs

Driving Factors for Automotive System Level Simulation

Increasing application and architecture complexity drive in Automotive the need for Model Based Design

Key Driving Factors:

- Power & Performance Envelope
  - More compute performance at the same clock speed within power budget
- Functional Safety
  - Error injection
- Pre-Silicon SW development

Model Supply Chain

Inter company model exchange

- Semi → Tier → OEM
- Model Extensions / add ons at each stage

Automotive Heterogeneous Simulation Requirements

Highly diverse modeling requirements by wide spread application scope and use cases

Heterogeneous system to be covered by automotive system simulation

- **ADAS** – Camera & Data fusion algorithms → **high speed** simulation
- **Powertrain** – High dynamic control loops →co-simulation with **plant models**
- **Body** – Network and I/O driven → **connectivity** to networking tools
- **Chassis & Safety** – Control loops, Networks → **plant models & connectivity**
- **Driver Information System** – Graphics, Video, Cameras → **virtual displays, speed**

Virtual Prototype Merits and Accuracy Requirements

- Early System Trade-Off Analysis
  - Manage the complexity
  - Analysis beyond CPU MIPs and memory footprint
- System evaluation of new IP ahead of first Silicon.
- Prepare system/user oriented test cases
- Early software development start
- MCU / system operation insight (perhaps not available in hardware)
  - Potential for fault injection scenario analysis
  - Potential for FMEA case scenario analysis

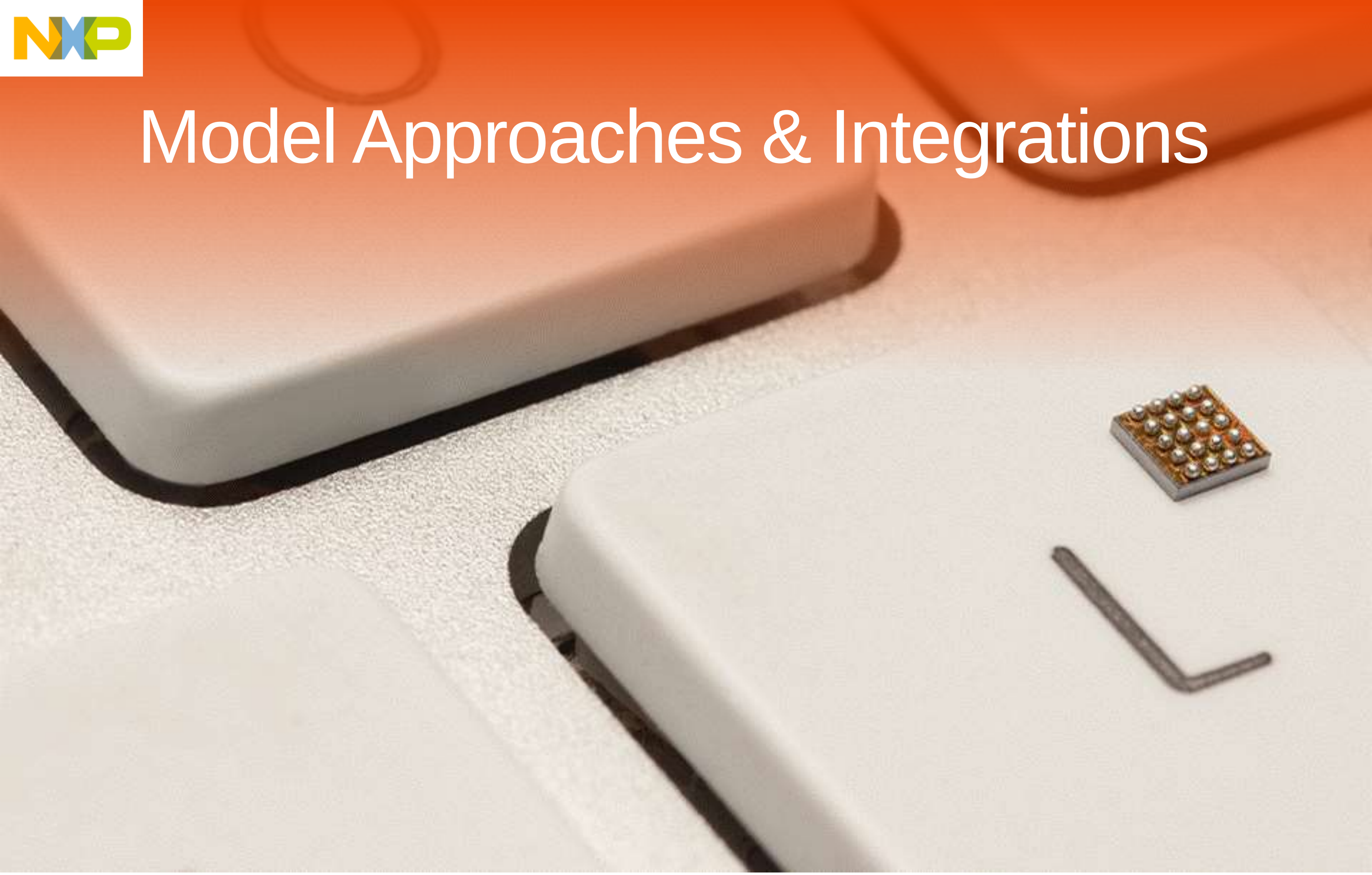
AT

AT/LT

LT

LT

→ Multiple use cases to be resolved by one model
→ Target to avoid multiple model investments
→ Ideal Case „one size fits all“



Model Approaches & Integrations

Standards Required for Affordable Eco Systems

- Virtualization has been used by many years in the industries
  - Initial company proprietary solutions
  - Future is fully standardized, inter operable models in an ecosystem
    - TLM, model to tool APIs, model to model standard definitions
- Automotive modeling standards and best practices are evolving
  - Standards basis for productivity gain & Eco System Build up

More Standards being needed going forward

1980s
- Company internal solutions
- Semi's, Tier1, OEMs
- Proprietary by nature
- Targeted to few use cases

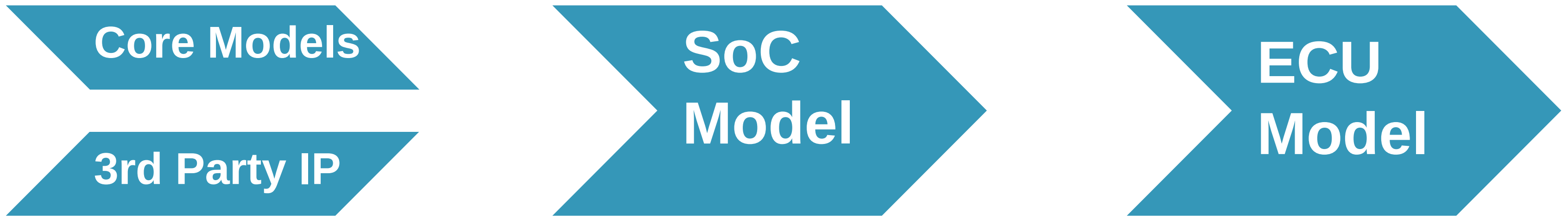
1990s
- Commercial solutions became available
- Often based on out placed company solutions
- First standardization efforts visible

2000s
- Standards evolve
- Model reuse
- Wider number of use cases addressed
- Commercial ecosystems became available
- Wider use

2010s
- Standardization on :
- TLM
- Model to tool
- Model exchange
- Model guide lines
- Standards & automations

Standards

System Level Model Focuses



IP Models

- Accurate IP development models
- Limited SW Error Injection capabilities

Accelerator / Core models

- Accurate vs. Speed option
  - Kcycles vs. xxx MIPS
- Debugger integration of accelerators
- Limited error inject capabilities for SW use case

SoC

- Key new features vs. model completeness
  - Functional vs. Performance analysis
- Reuse of IP model eco-system (core, GPU, interconnect, accelerators)
- SW Tool integrations

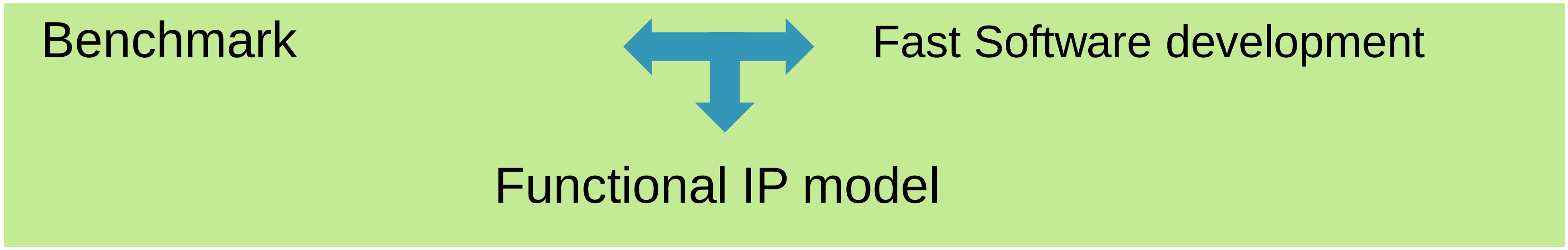
ECU

- Interconnect model to plant
  - Connect known comms IP
- High speed simulation to support multi-core use case
- Multi staged Tool & IP support chain

Risk of Requirements (Over) load

System Level models at creation time

- „Which model do we want to use“ !



- Has the model being created for the initial use case with supply chain in mind → The models have to support the „next use case“
- Too high or too low abstraction level → someone might be dissappointed (e.g. cycle accurate cores vs. functional cores)

Contradicting Abstraction Levels Example

- Transactors make address model integrations on different abstraction levels
- Transactors don‘t solve abstraction mismatch for use case mismatch at creation time vs next step usage

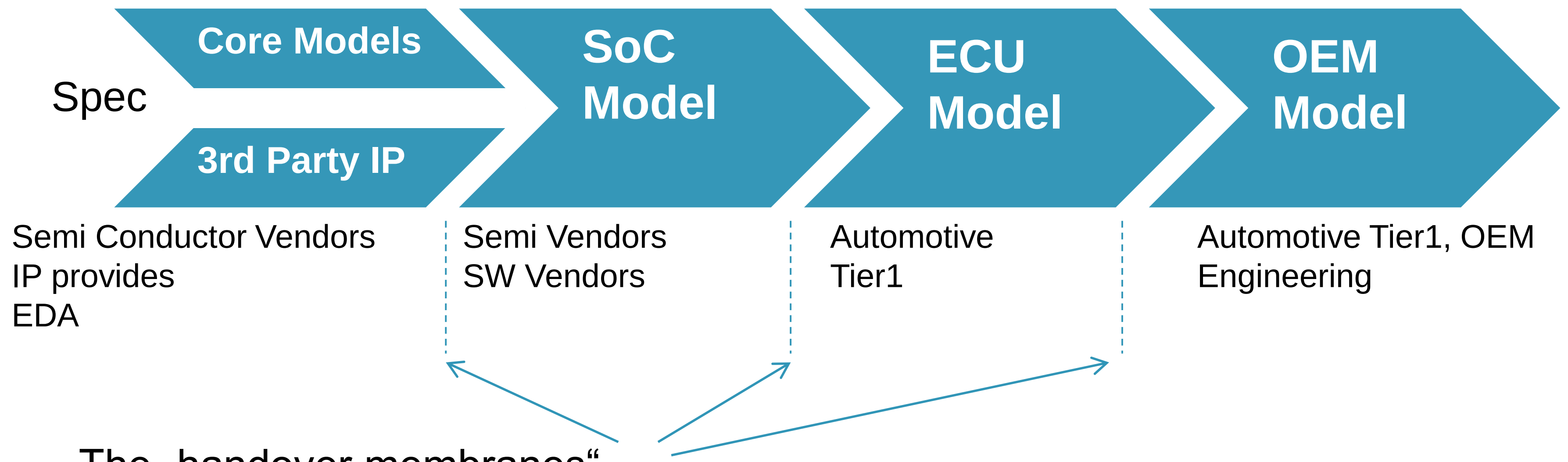
Model Completeness Overview

Model Type	Scope	Use Case
Core model ISS only	Core only, Instruction set covered (no exceptions), e.g., Lauterbach build in ISS	Running cross compiled code without interaction to peripherals
Core ISS With exceptions	Core model takes any exception into account; used often in high end speed models without timing	Code running with exceptions, e.g., software interrupts
Core model cycle count accurate	Used for profiling requires accurate memory models	Used for profiling, cycle count accuracy; periphery stub as memory
Platform model	Core, memory, IRQ, xbar Can be used for memory profiling	For OS porting/development, minimal peripheral features covered (max IRQ, timer)
Full chip model	Used all models	Covering all peripherals, EVB equivalent
Full chip model with plant model	Application development	Additional models are connected or co-simulated

Virtual Platforms

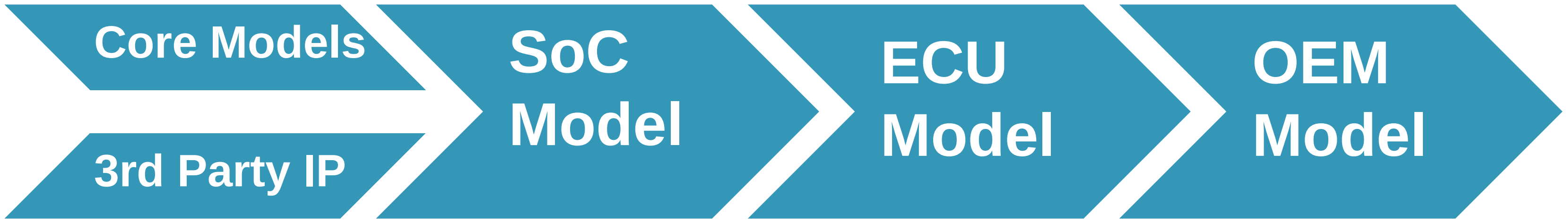
Build Process

Automotive Model Aggregation



- The „handover membranes“
- Dependency at each handover phase
  - Model abstraction (LT / AT) → increasing abstraction along the supply chain
  - License models to aligned→ Who invests? Who benefits ?
  - Tool vendor numbers increase (Software debuggers, Co-Simulations, etc.)
    - Increase eco-system complexity
  - First line support moves with the integration stages

Model Life Cycle



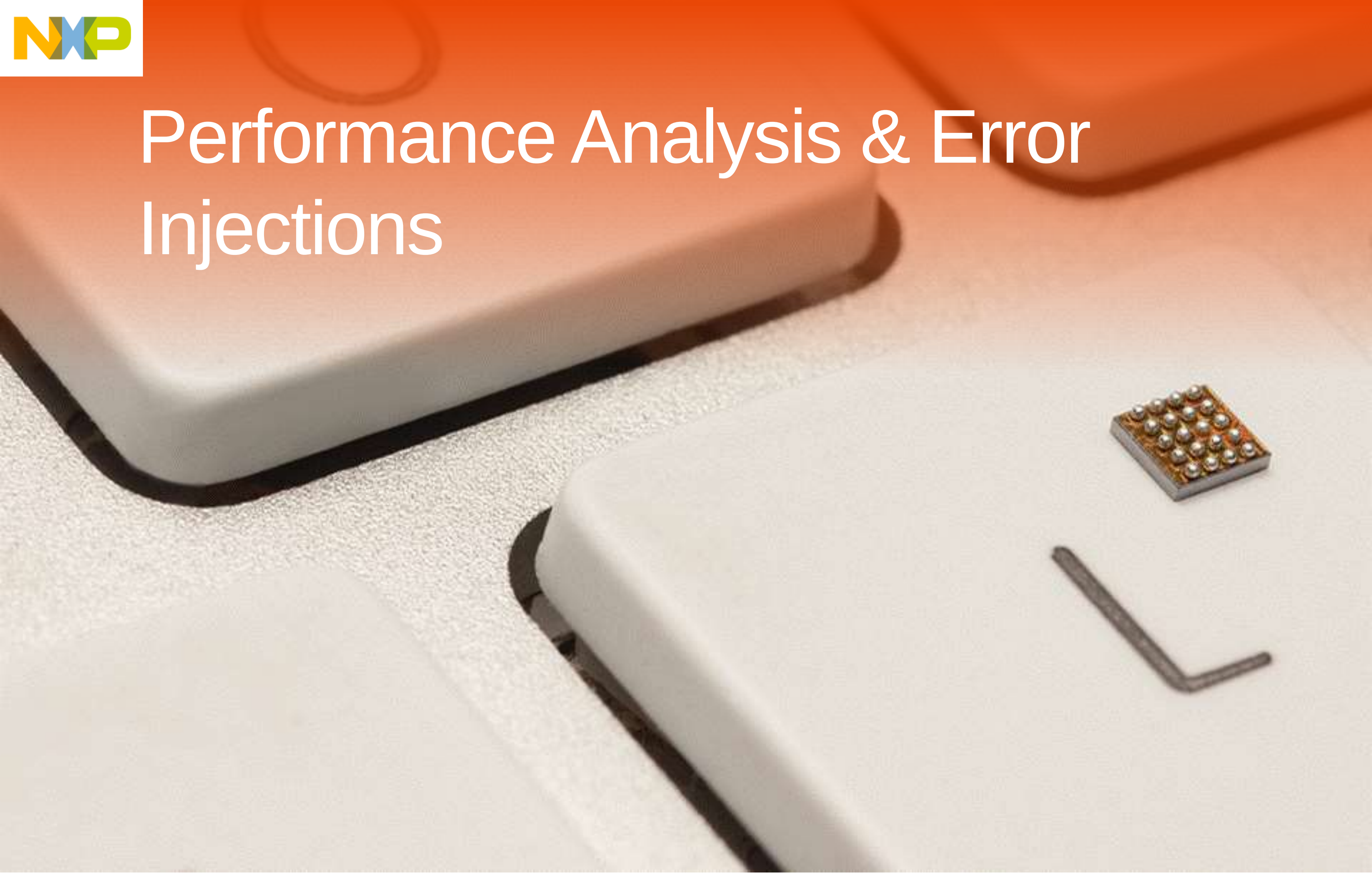
Model Integrations

- Virtual Prototype delivery through development Partners
  - Synopsys
  - ASTC

ECU Integration by Tool Partners

- ECU level integrations (Simulink, CANoe, etc.)
- 3rd party tool integration

→ ECU level Performance Analysis



Performance Analysis & Error Injections

Virtual Prototype Benchmarking

Benchmarking on Virtual prototypes are based on two main categories

Absolute benchmark based on cycle approximate model base

→ Approximated Timed Models

Relative benchmarks based on known baseline

→ Loosely Timed Models

Error Injection

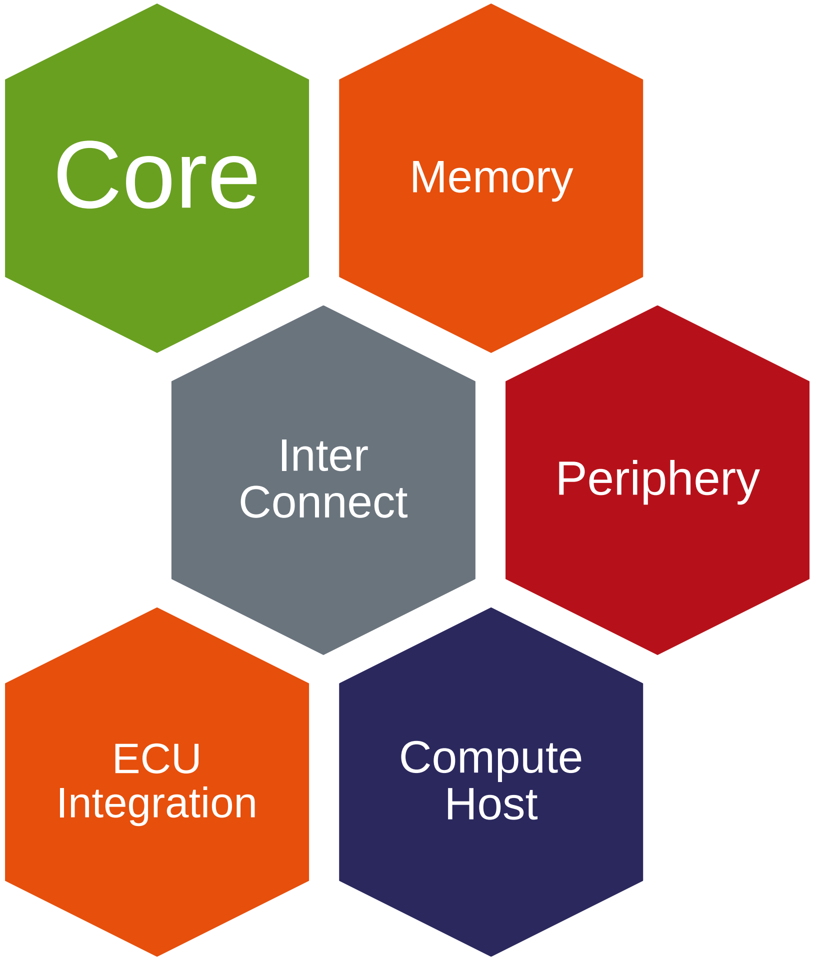
→ Functional Safety Software development

Cycle Approximate Performance Analysis

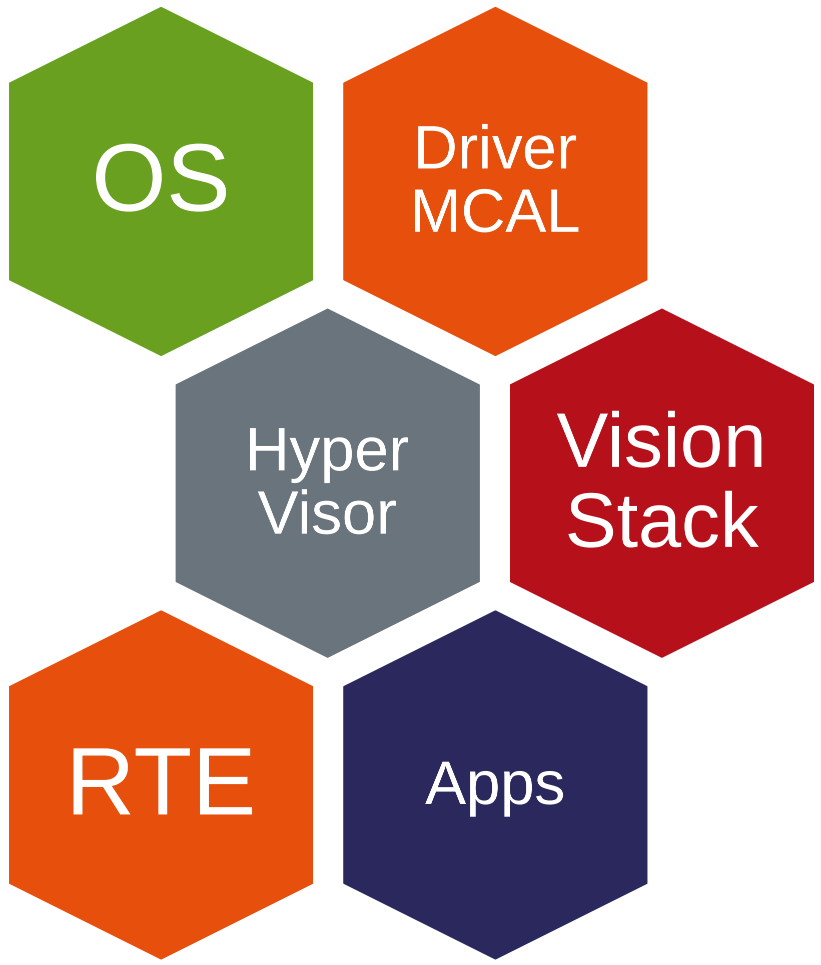
Cycle Approximate peformance models require full data path coverage

→Primary target are the core and memory platform

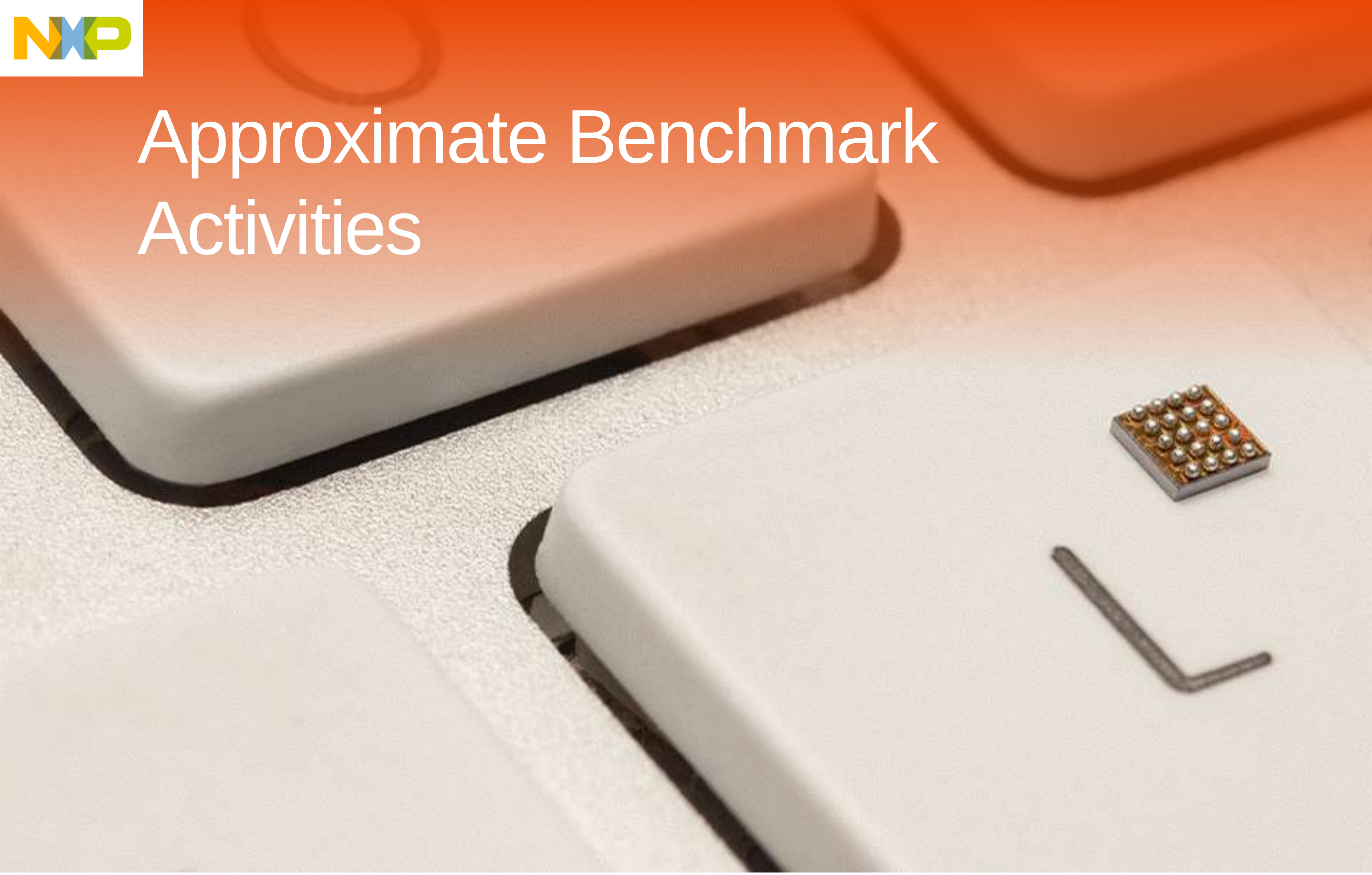
→Simulation speed is determined by accurate model setup



Relative Performance Assessment



- Timing annotated models
- Large Software pacakges
- OS booting System
- Focus is on Functional Software stacks
- Interplay of SW deliverables



Approximate Benchmark Activities

Benchmark on Approximate Models

- Main Use cycle approximate execution
- Benchmark and taking time stamps
- Dedicated SW components visible

$\Downarrow$

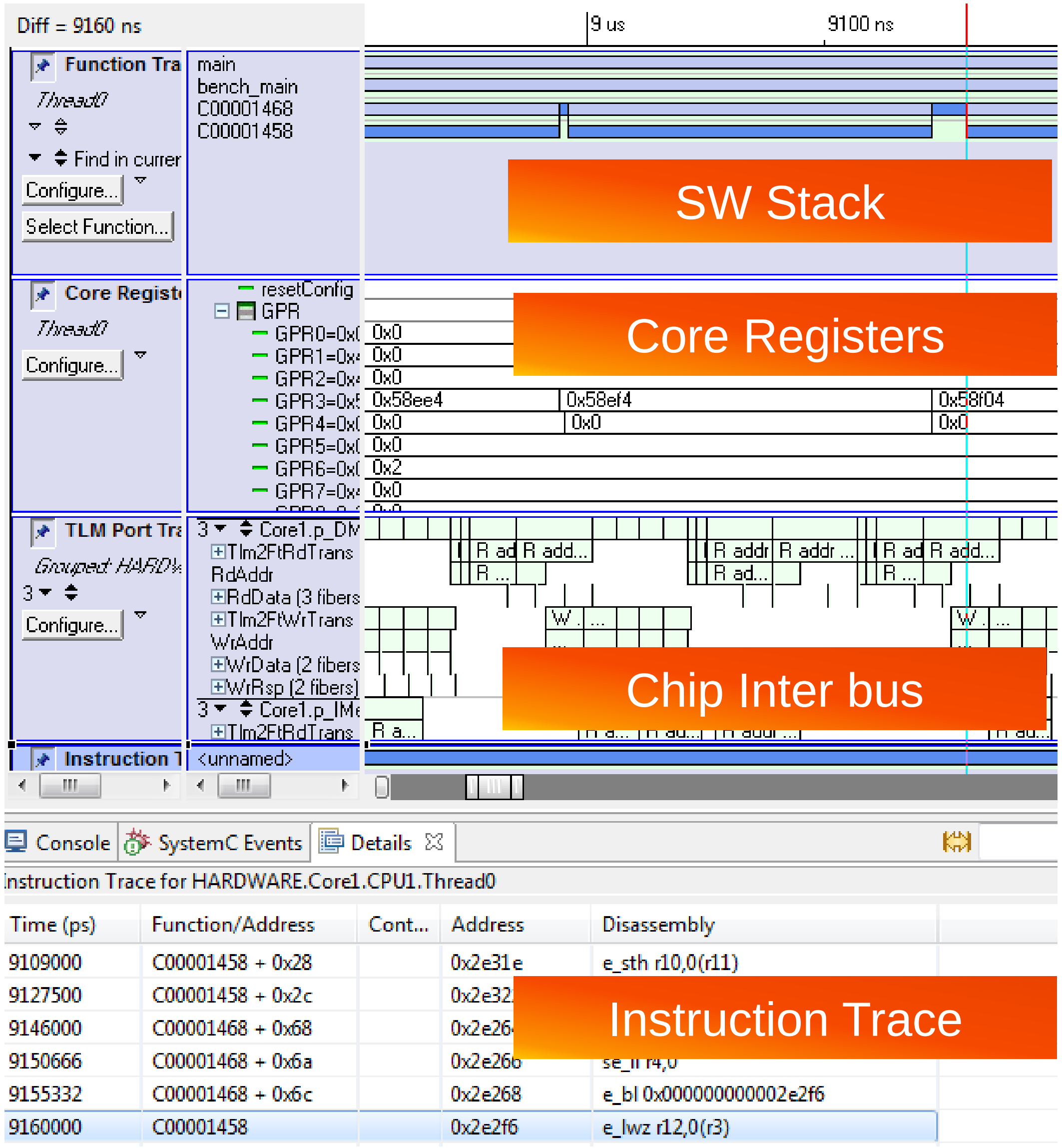
- But what is the root cause ?
- What is the sensitivity ?

$\Downarrow$

→Analysis capabilities allow to detect root causes
→Non-intrusive instrumention

Full Benchmark View

- SW Stack listed
  - Dedicated statistics available
- Core Register access
  - Shows compiler efficiency
- Individual Transactions view
  - Timing annotated
  - Internal bus traffice
- Instruction view
  - Cause of instruction to transactions being visible

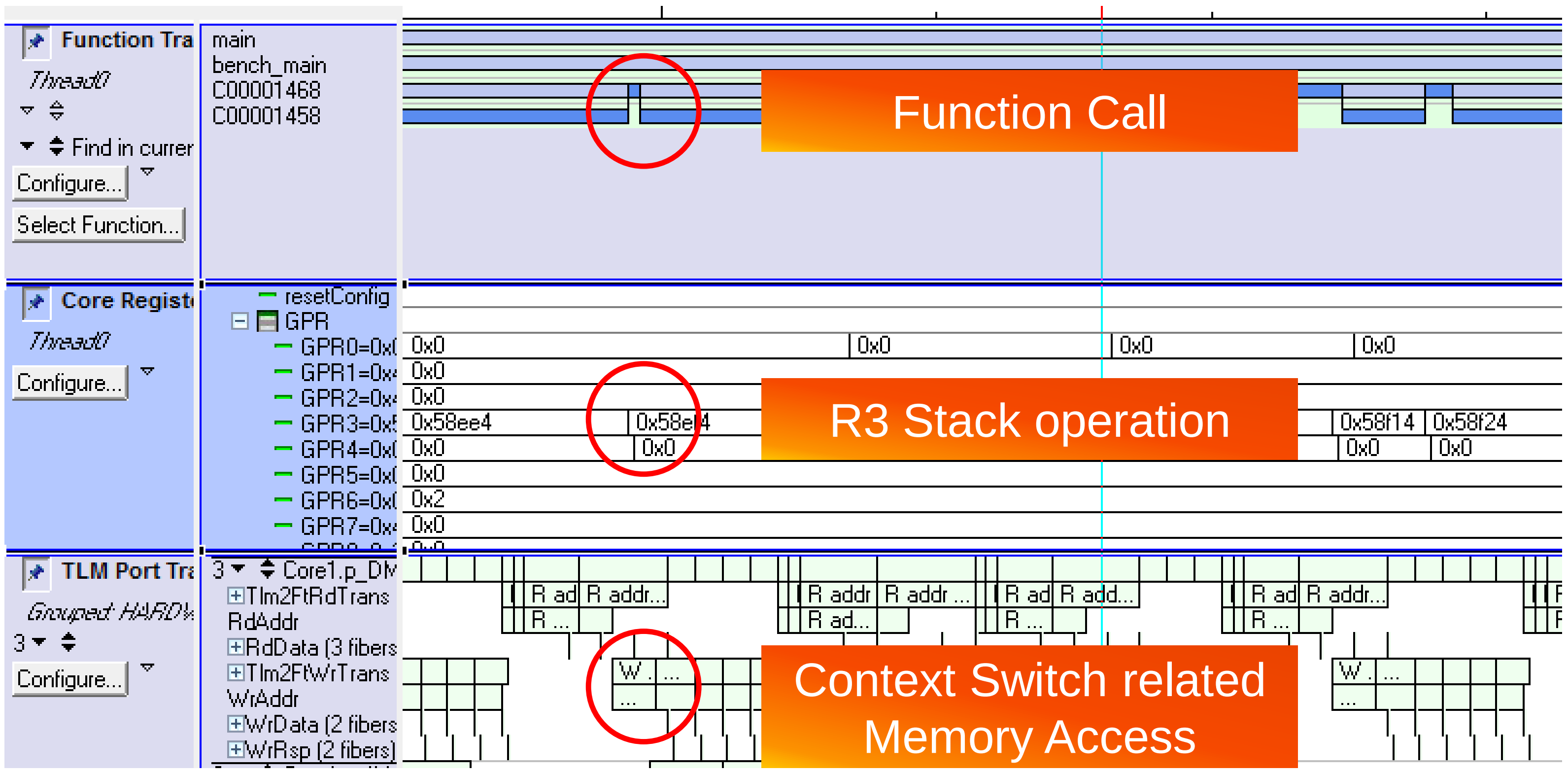


Instruction Trace for HARDWARE.Core1.CPU1.Thread0

Time (ps)	Function/Address	Cont...	Address	Disassembly
9109000	C00001458 + 0x28		0x2e31e	e_sth r10,0(r11)
9127500	C00001458 + 0x2c		0x2e32	
9146000	C00001468 + 0x68		0x2e26	
9150666	C00001468 + 0x6a		0x2e266	se_li r4,0
9155332	C00001468 + 0x6c		0x2e268	e_bl 0x000000000002e2f6
9160000	C00001458		0x2e2f6	e_lwz r12,0(r3)

Example Stack Operation

- SW to Transaction Root cause



PowerArchitecture Virtual Prototype

Loosely Timed Benchmark

What can be done with loosely Timed benchmarks

Loosely timed models can be used

→Large number of performance relevant data can be extracted

→Ease of control in Virtual Prototype

→Ease of HW and SW interaction through deeper logging capability

→Hardware event can be viewed concurrent to SW event

- Example: Core Register vs Function view

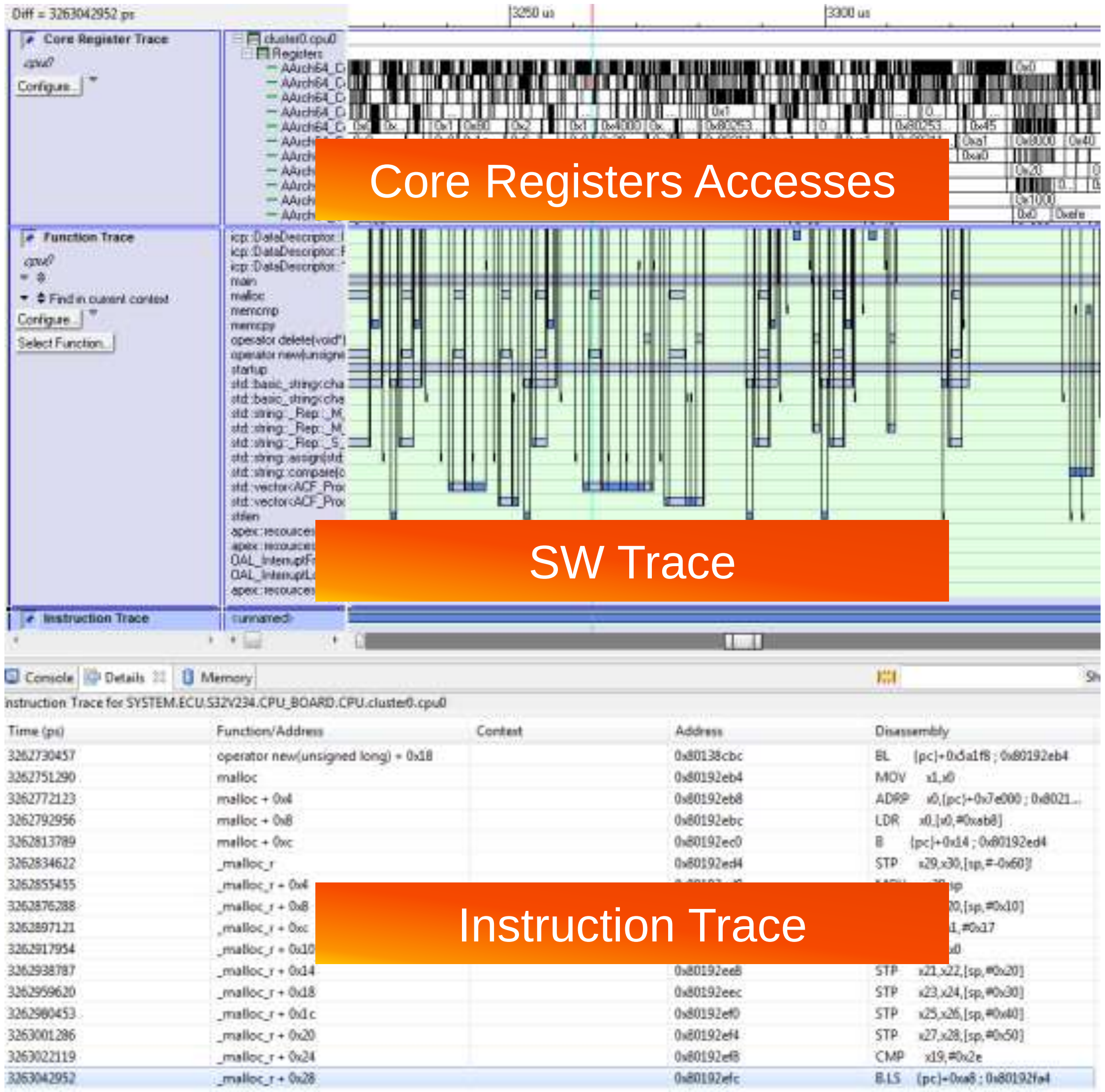
Code profiling

- Call stacks
- Instruction counts per function
- HW access rates
- Parametric change of W settings
- Cache Analysis

SW Stack Analysis

- Concurrent view of fully traceable SW
- Function View
- Core Register View
- Peripheral View
- Instruction View

→Virtual Prototype allows a „one stop view“ on in the SW hierarchy down to HW related events



Instruction Trace for SYSTEM.ECU.S32V234.CPU_BOARD.CPU.cluster0.cpu0

Time (ps)	Function/Address	Context	Address	Disassembly
3262730457	operator new(unsigned long) + 0x18		0x80138cbc	BL [pc]+0x5a1f8 ; 0x80192eb4
3262751290	malloc		0x80192eb4	MOV x1,x0
3262772123	malloc + 0x4		0x80192eb8	ADRP x0,[pc]+0x7e000 ; 0x8021...
3262792956	malloc + 0x8		0x80192ebc	LDR x0,[x0,#0xab8]
3262813789	malloc + 0xc		0x80192ec0	B [pc]+0x14 ; 0x80192ed4
3262834622	_malloc_r		0x80192ed4	STP x29,x30,[sp,#-0x60]!
3262855455	_malloc_r + 0x4		[illegible]	[illegible]
3262876288	_malloc_r + 0x8		[illegible]	[illegible]
3262897121	_malloc_r + 0xc		[illegible]	[illegible]
3262917954	_malloc_r + 0x10		[illegible]	[illegible]
3262938787	_malloc_r + 0x14		0x80192ee8	STP x21,x22,[sp,#0x20]
3262959620	_malloc_r + 0x18		0x80192eec	STP x23,x24,[sp,#0x30]
3262980453	_malloc_r + 0x1c		0x80192ef0	STP x25,x26,[sp,#0x40]
3263001286	_malloc_r + 0x20		0x80192ef4	STP x27,x28,[sp,#0x50]
3263022119	_malloc_r + 0x24		0x80192ef8	CMP x19,#0x2e
3263042952	_malloc_r + 0x28		0x80192efc	B.LS [pc]+0xa8 ; 0x80192fa4

S32V234 Virtual Prototype → Linux Boot

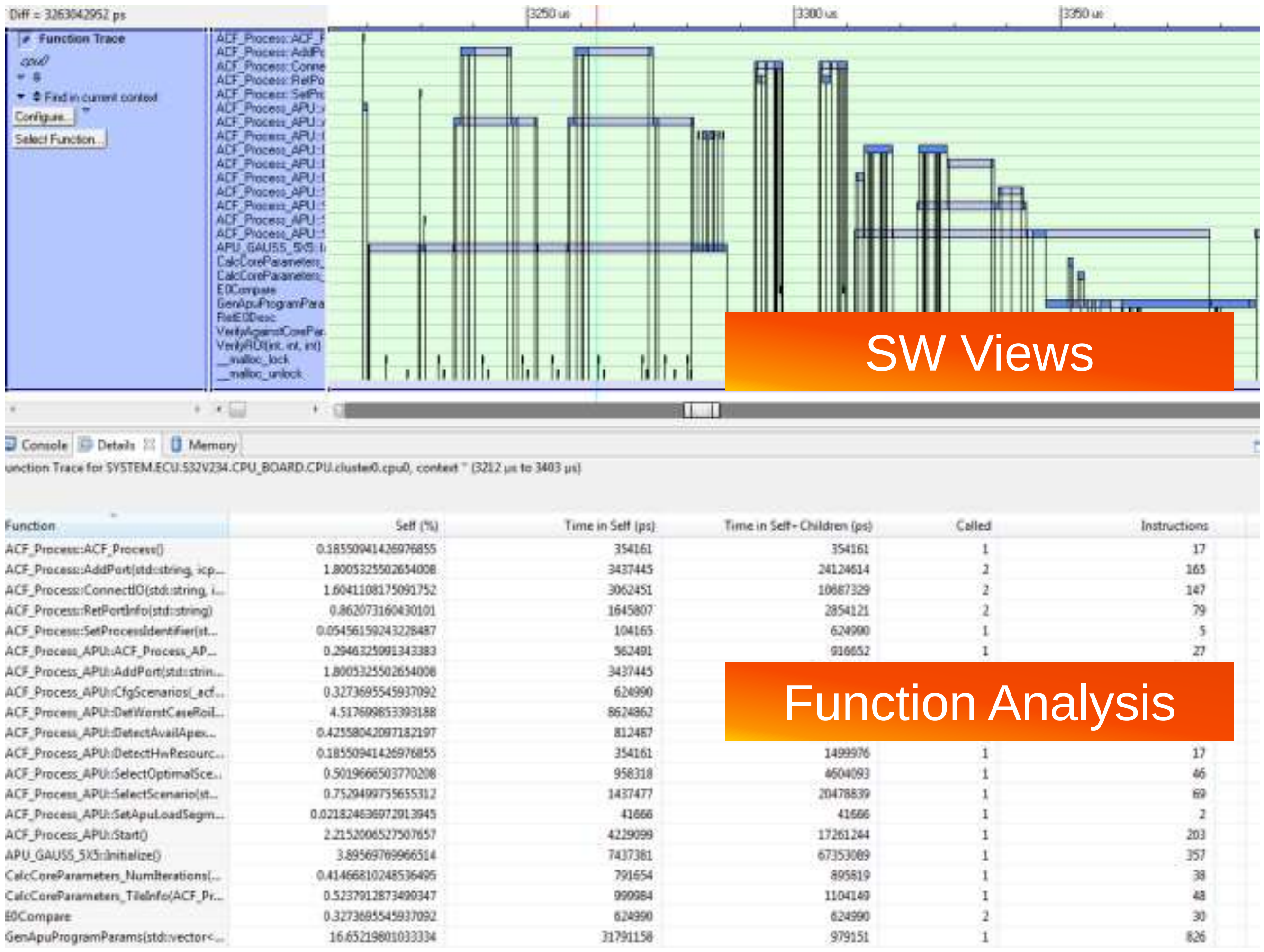
APEX Vision Example -S32V234 Virtual Prototype

SW Traces

Vision Accelerator softwre

Stats on Trace

- Host Code on ARM A53 core complex
- Approximate Timed view
- Function Statistic



SW Views

unction Trace for SYSTEM.ECU.S32V234.CPU_BOARD.CPU.cluster0.cpu0, context '' (3212 µs to 3403 µs)

Function	Self (%)	Time in Self (ps)	Time in Self+Children (ps)	Called	Instructions
ACF_Process::ACF_Process()	0.18550941426976855	354161	354161	1	17
ACF_Process::AddPort(std::string, icp...	1.8005325502654008	3437445	24124614	2	165
ACF_Process::ConnectIO(std::string, i...	1.6041108175091752	3062451	10887329	2	147
ACF_Process::RetPortInfo(std::string)	0.862073160430101	1645807	2854121	2	79
ACF_Process::SetProcessIdentifier(st...	0.05456159243228487	104165	624990	1	5
ACF_Process_APU::ACF_Process_AP...	0.2946325091343383	562491	916652	1	27
ACF_Process_APU::AddPort(std::strin...	1.8005325502654008	3437445	[illegible]	[illegible]	[illegible]
ACF_Process_APU::CfgScenarios(_acf...	0.3273695545937092	624990	[illegible]	[illegible]	[illegible]
ACF_Process_APU::DetWorstCaseRoiI...	4.517699853393188	8624862	[illegible]	[illegible]	[illegible]
ACF_Process_APU::DetectAvailApex...	0.4255804209718219	812487	[illegible]	[illegible]	[illegible]
ACF_Process_APU::DetectHwResourc...	0.18550941426976855	354161	1499976	1	17
ACF_Process_APU::SelectOptimalSce...	0.5019666503770208	958318	4604093	1	46
ACF_Process_APU::SelectScenario(st...	0.7529499755655312	1437477	20478839	1	69
ACF_Process_APU::SetApuLoadSegm...	0.021824636972913945	41666	41666	1	2
ACF_Process_APU::Start()	2.2152006527507657	4229099	17261244	1	203
APU_GAUSS_5X5::Initialize()	3.895697699665514	7437381	67353089	1	357
CalcCoreParameters_NumIterations(...	0.41466810248536495	791654	895819	1	38
CalcCoreParameters_TileInfo(ACF_Pr...	0.5237912873490347	999984	1104149	1	48
E0Compare	0.3273695545937092	624990	624990	2	30
GenApuProgramParams(std::vector<...	16.65219801033334	31791158	979151	1	826

Function Analysis

APEX ACF Software Profiling

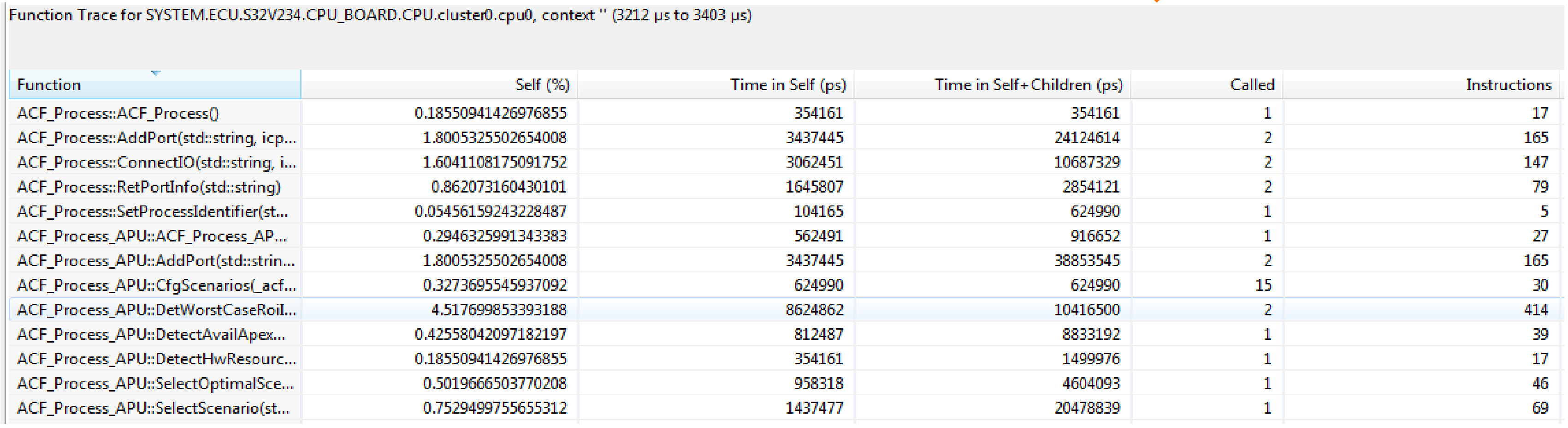
- Capturing on functional level the Code Execution On ARM A53 core
- Key performance metrics can be accessed

Function Names executed in scope

Relative Time being spend in functions

# of Calls

# of instructions per function

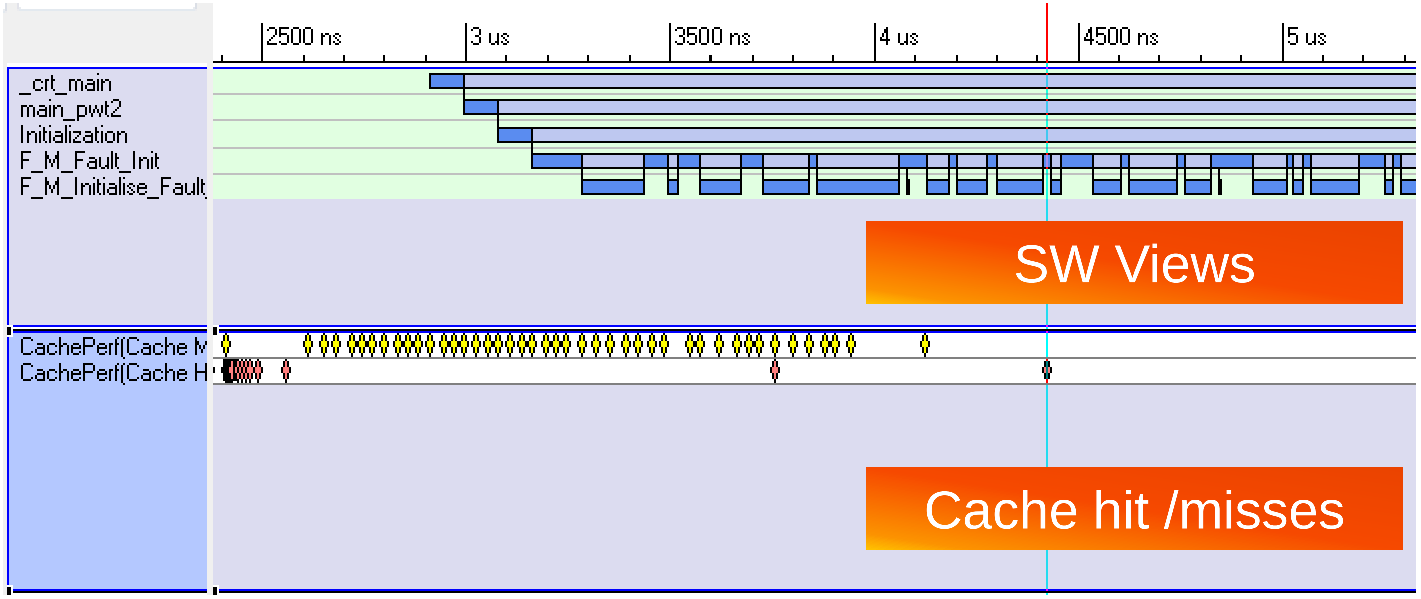


Function Trace for SYSTEM.ECU.S32V234.CPU_BOARD.CPU.cluster0.cpu0, context '' (3212 µs to 3403 µs)

Function	Self (%)	Time in Self (ps)	Time in Self+Children (ps)	Called	Instructions
ACF_Process::ACF_Process()	0.18550941426976855	354161	354161	1	17
ACF_Process::AddPort(std::string, icp...	1.8005325502654008	3437445	24124614	2	165
ACF_Process::ConnectIO(std::string, i...	1.6041108175091752	3062451	10687329	2	147
ACF_Process::RetPortInfo(std::string)	0.862073160430101	1645807	2854121	2	79
ACF_Process::SetProcessIdentifier(st...	0.05456159243228487	104165	624990	1	5
ACF_Process_APU::ACF_Process_AP...	0.2946325991343383	562491	916652	1	27
ACF_Process_APU::AddPort(std::strin...	1.8005325502654008	3437445	38853545	2	165
ACF_Process_APU::CfgScenarios(_acf...	0.3273695545937092	624990	624990	15	30
ACF_Process_APU::DetWorstCaseRoiI...	4.517699853393188	8624862	10416500	2	414
ACF_Process_APU::DetectAvailApex...	0.42558042097182197	812487	8833192	1	39
ACF_Process_APU::DetectHwResourc...	0.18550941426976855	354161	1499976	1	17
ACF_Process_APU::SelectOptimalSce...	0.5019666503770208	958318	4604093	1	46
ACF_Process_APU::SelectScenario(st...	0.7529499755655312	1437477	20478839	1	69

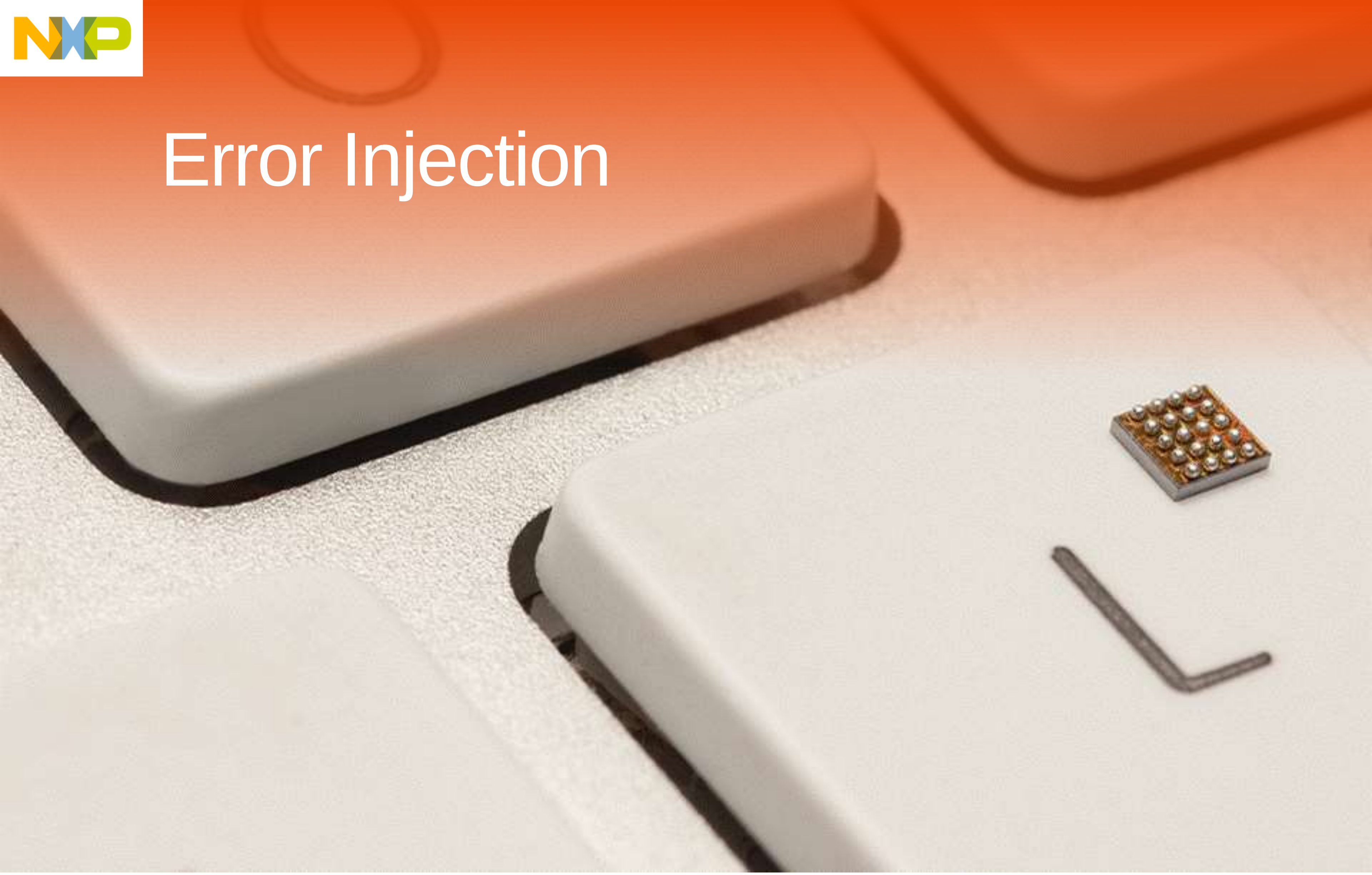
Cache Analysis

- Relative Performance analysis based on hit rates



- Cache Events can directly be viewed concurrent to the SW Stack

Dual Core Demo



Error Injection

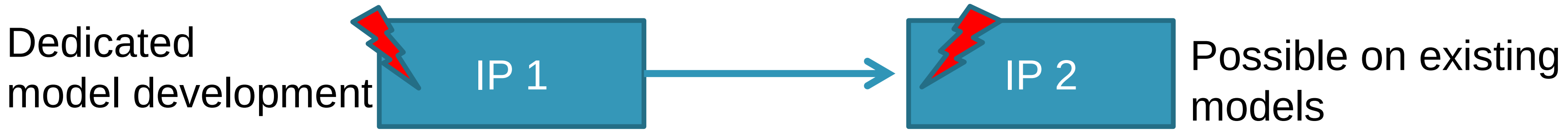
Error Injections for Functional Safety Software Test

Key Use Case:

- Functional safety software validation through error injection control
- Target users are software developers
- Models supporting the error inject mechanisms

Method:

- Point of Error Notification → easier model creation, only error management
- Point of Error Creation →higher model effort, error stage on each model to be supported, complex error management capability



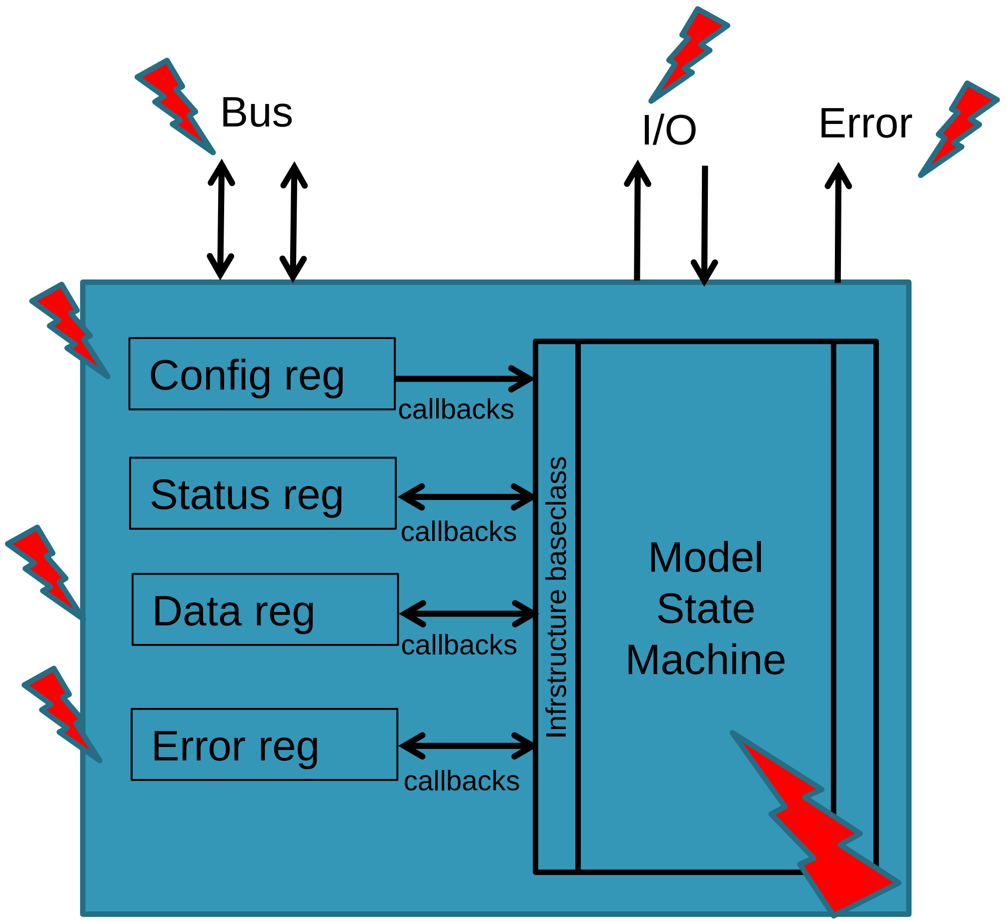
Examples

- Software Watchdog: Expiry, IRQ notification
- Cache Error: core model state machine vs. error management units

Example Error Inject types and model impact

Error inject at **point of notification** vs. **Error inject at point of error creation** to be decided at IP model level creation

Error Inject Type	Model Impact Error inject mechanisms
State machine Error	- Tool API needed - **Model modification**
Config Register Erro	- Tool API
Bus Error	- **Model modification**
Error notification	- Tool signal change - **Model modification**
I/O error	- Tool API - **output model modification**
Data reg	- Tool API - **Outdate modification**



Safety Elements – Module View

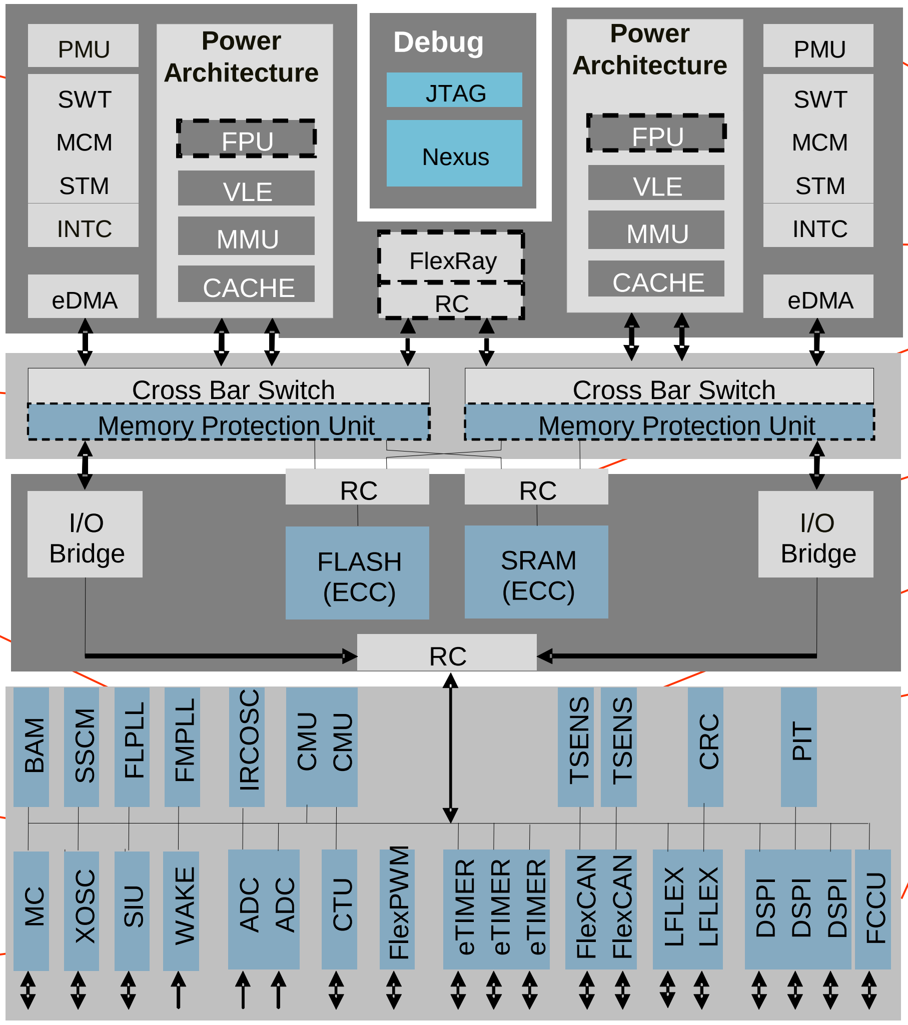
Sphere of Replication:
- *Replicated e200Core*
- *replicated eDMA*
- *redundant INTC, SWT, etc*
- *redundant MMU*
- *RC Units at Gates to non redundant sphere*

XBAR + MPU:
- *Redundant*
- *RC Units at Gates to non redundant sphere*

Clock Monitoring
- *Detects and mitigates clock disturbances*
- PLL

Timer
- *eTimer0 channels "isolated"*

ADC
- *On Line Assisted Hardware BIST*



PMU
- *internal Vreg*
- *redundant Vmonitor*

Flexray

Flash
- *ECC*

RAM
- *ECC*

Temp Sensor
- *redundant*

CRC Unit
- *Application Signature*

Fault Collection Unit
- *detects when errors have occurred*
- *indicates error to external*
- *independent of software operation*

Example - SWT recovers error on SW Task

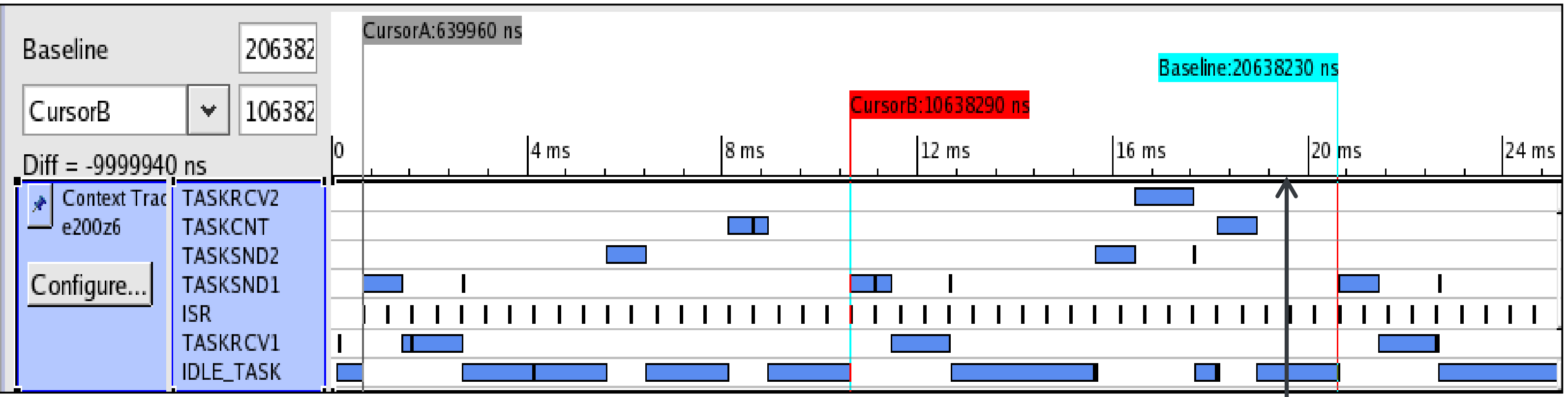
- System configuration:
  - SWT is configured to reset the HW after some time (11ms) without SW interaction
  - Periodic task (TASKCNT) that fires every 10ms resets the SWT count

Description

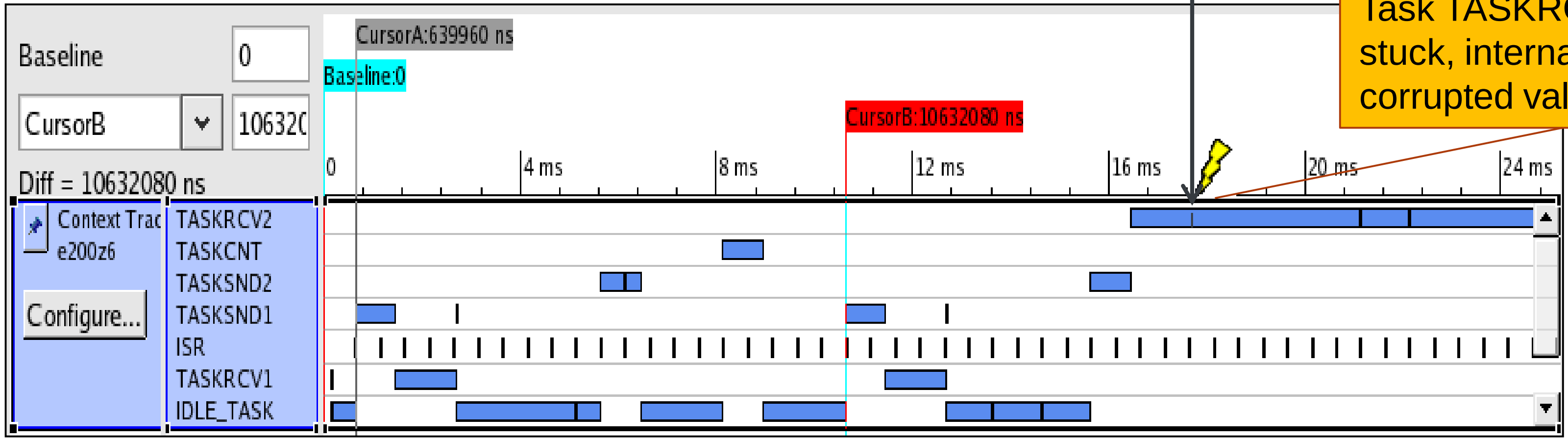
1. e200 core runs the SW application
2. Task TASKRCV2 gets a corrupted value in one of their computing loops
3. Task TASKRCV2 executes longer than expected blocking the processor(higher priority)
4. After 11ms, SWT resets the complete MCU
5. Execution starts from the beginning, SW in safe state again

Problem overview (without SWT)

Normal operation



Task RCV2 stuck with SWT disable

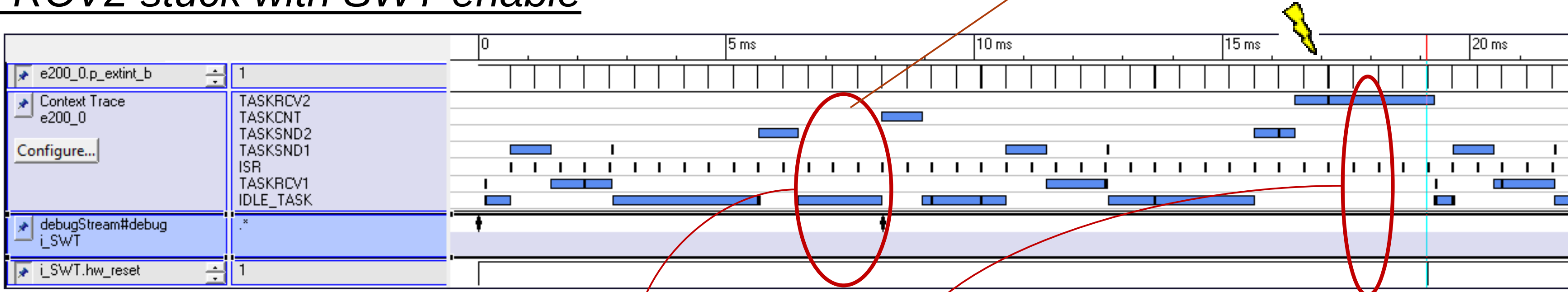


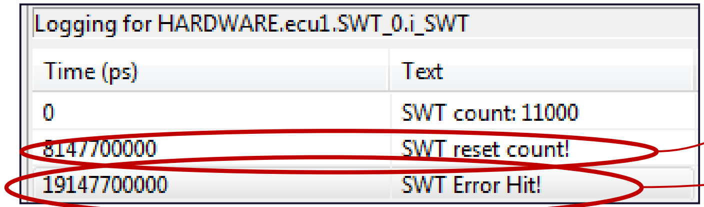
Task TASKRCV2 get stuck, internal loop gets corrupted value

SWT operation

SWT is configured to be triggered after 11ms without SW comm.
TaskCNT reset watchdog count during execution (task rate 10ms)

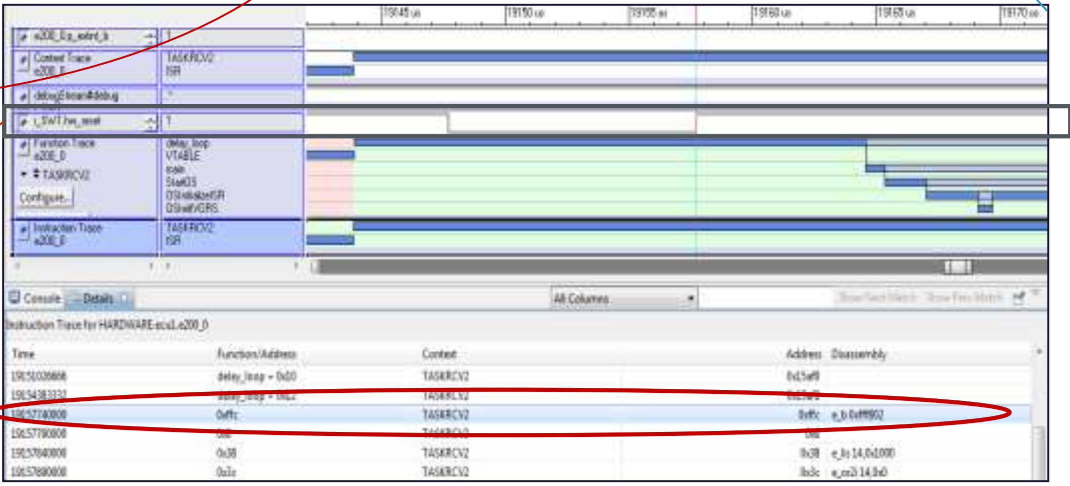
Task RCV2 stuck with SWT enable





Logging for HARDWARE.ecu1.SWT_0.i_SWT

Time (ps)	Text
0	SWT count: 11000
8147700000	SWT reset count!
19147700000	SWT Error Hit!



Instruction Trace for HARDWARE.ecu1.e200_0

Time	Function/Address	Context	Address	Disassembly
19151026868	delay_loop + 0x20	TASKRCV2	0xd15af8	
19154383332	delay_loop + 0x22	TASKRCV2	0xd15afa	
19157740000	0xffc	TASKRCV2	0xffc	e_b 0xfff802
19157790000	0x0	TASKRCV2	0x0	
19157840000	0x38	TASKRCV2	0x38	e_lis 14,0x1000
19157890000	0x3c	TASKRCV2	0x3c	e_or2i 14,0x0

SWT reset e200z6 core after 11ms inactivity

Erro Injection Summary

Easy control of Errors

„Revision control of errors“

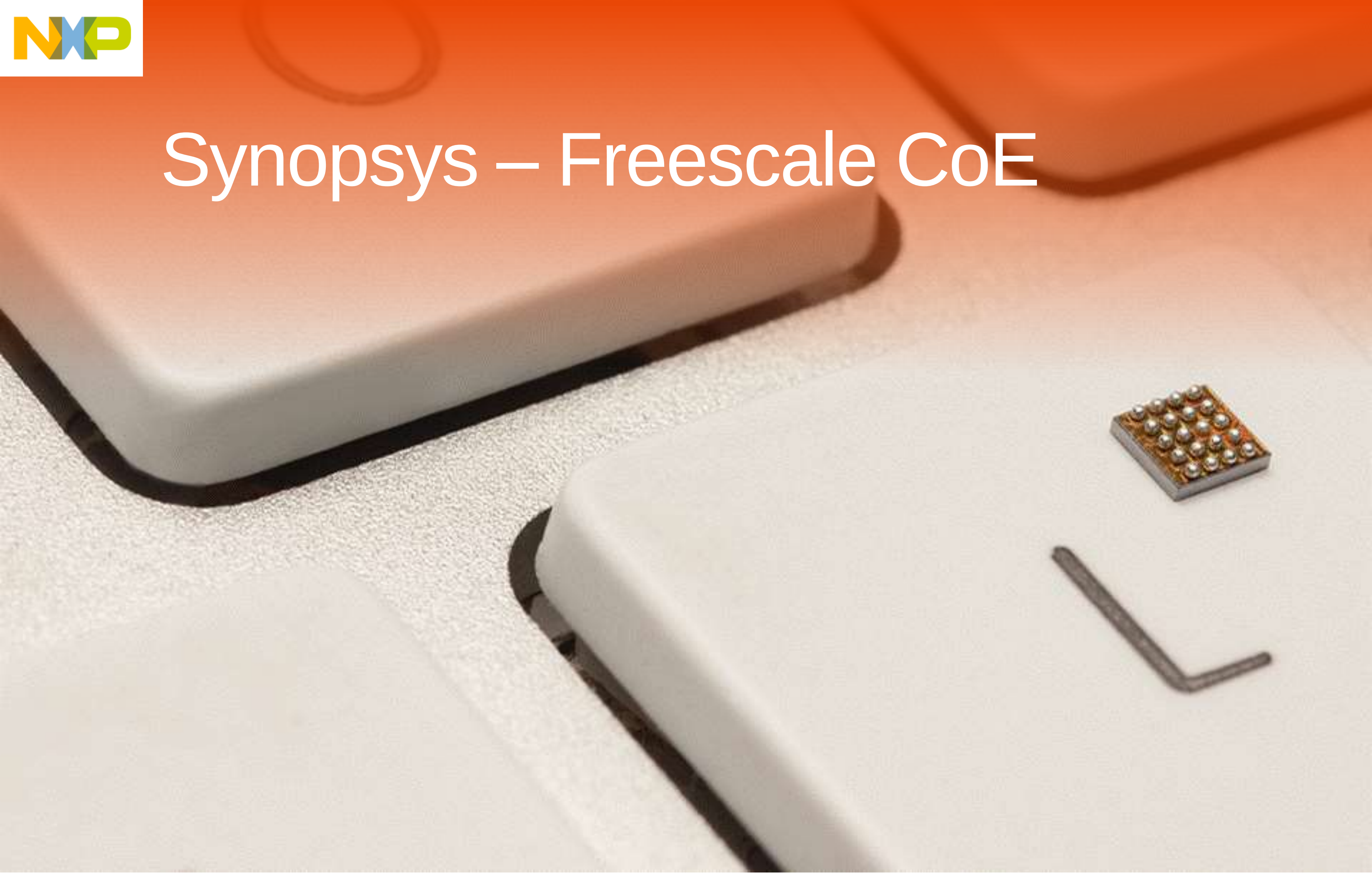
Extends to ECU level

Test suite development for errors

Allows soft and permanent error emulation

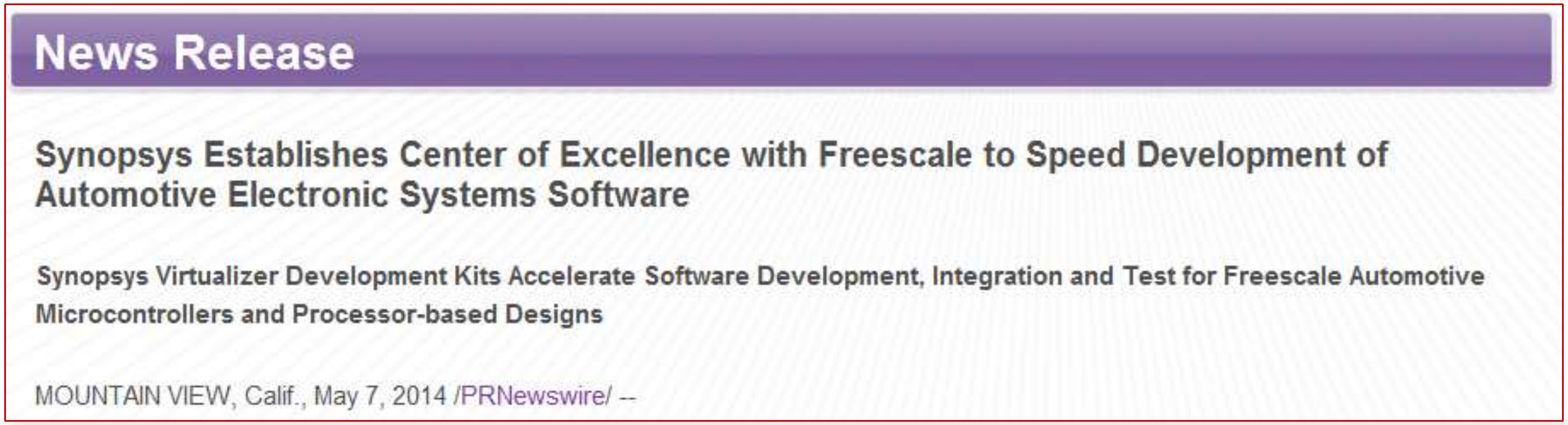
Regression on error software

Synopsys – Freescale CoE



Center of Excellence

- Synopsys Preferred Solution Provider for VDKs


News Release

Synopsys Establishes Center of Excellence with Freescale to Speed Development of Automotive Electronic Systems Software

Synopsys Virtualizer Development Kits Accelerate Software Development, Integration and Test for Freescale Automotive Microcontrollers and Processor-based Designs

MOUNTAIN VIEW, Calif., May 7, 2014 /PRNewswire/ --

CoE Activities (examples)

- IP model development
- Reference virtual prototypes
- Flow definition

Tier1/OEM

- Additional IP model development
- Customer platform assembly
- Support/training
- Use case deployment

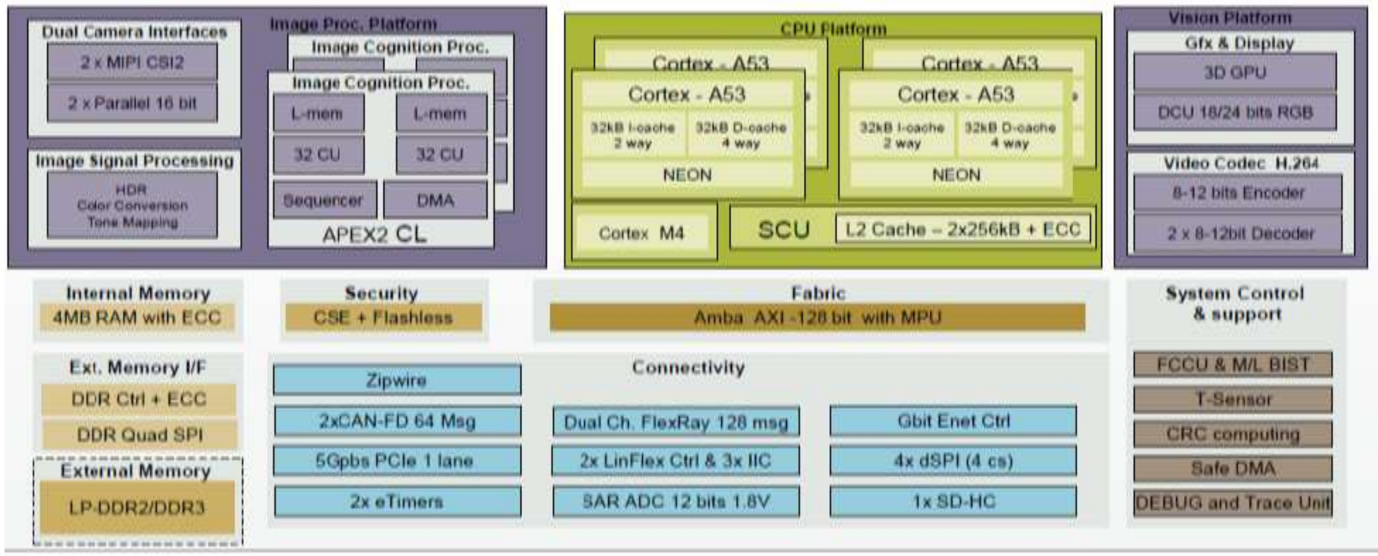
Long-term Alignment between Synopsys and Freescale

The Importance of a Center of Excellence

- Established for the long-term to support automotive development and product life cycles.
- Committed access to information and mutual support between engineering teams
  - Specifications, models, embedded software, software tools, etc.
- Simplified engagement
  - Commercial availability w/ roadmap
  - Optimized for Synopsys tools
  - Faster deployment

S32V234 Virtual Prototype

S32V234 ADAS Automotive Processor



Key Features	Benefits
Designed and manufactured to satisfy automotive reliability and ISO 26262 ASIL B functional safety requirements	Enables assessment to ISO 26262 ASIL B standard for automotive safety applications.
Quad 1 GHz ARM® Cortex®-A53 + ARM NEON™ core platform	9.2K DMIPS processing horsepower (without acceleration) for management of ADAS tasks.
ARM Cortex-M4 at 133 MHz for IO control and Autosar OS	Enables automotive OS such as Autosar to control interfaces to external devices without impacting ARM Cortex-A53 performance.
Embedded security	Security engine together with ARM TrustZone® technology provides protection against IP theft and malicious hacking.
Dual APEX-2 image processing engine	Allows high-performance, low-power processing of incoming image data.
Image signal processor	Performs image housekeeping tasks such as HDR and color conversion, plus some dedicated image processing tasks.
3D graphics processing unit (3D GPU)	For rendering 3D images. May also be used for additional image analysis tasks.
Video input: dual MIPI-SCI; dual video input unit (VIU)	Supports mono, stereo and surround view camera inputs. H.264 decode and encode also supported.
Memory interfaces	DRAM support for LPDDR2/DDR3L/DDR3 for high bandwidth data access, plus Dual QuadSPI for external flash.

S32V234 Virtual Prototype – Synopsys VDK

Virtual Platform for pre-silicon software develpment (Fast Functional Model)

- A53 core clusters
  - Linux / Dhrystone initial setups available
- APEX2 IP accelerator
- Memories
- Selected peripherals
- Video probes

Debugger enabled

- Lauterbach T32

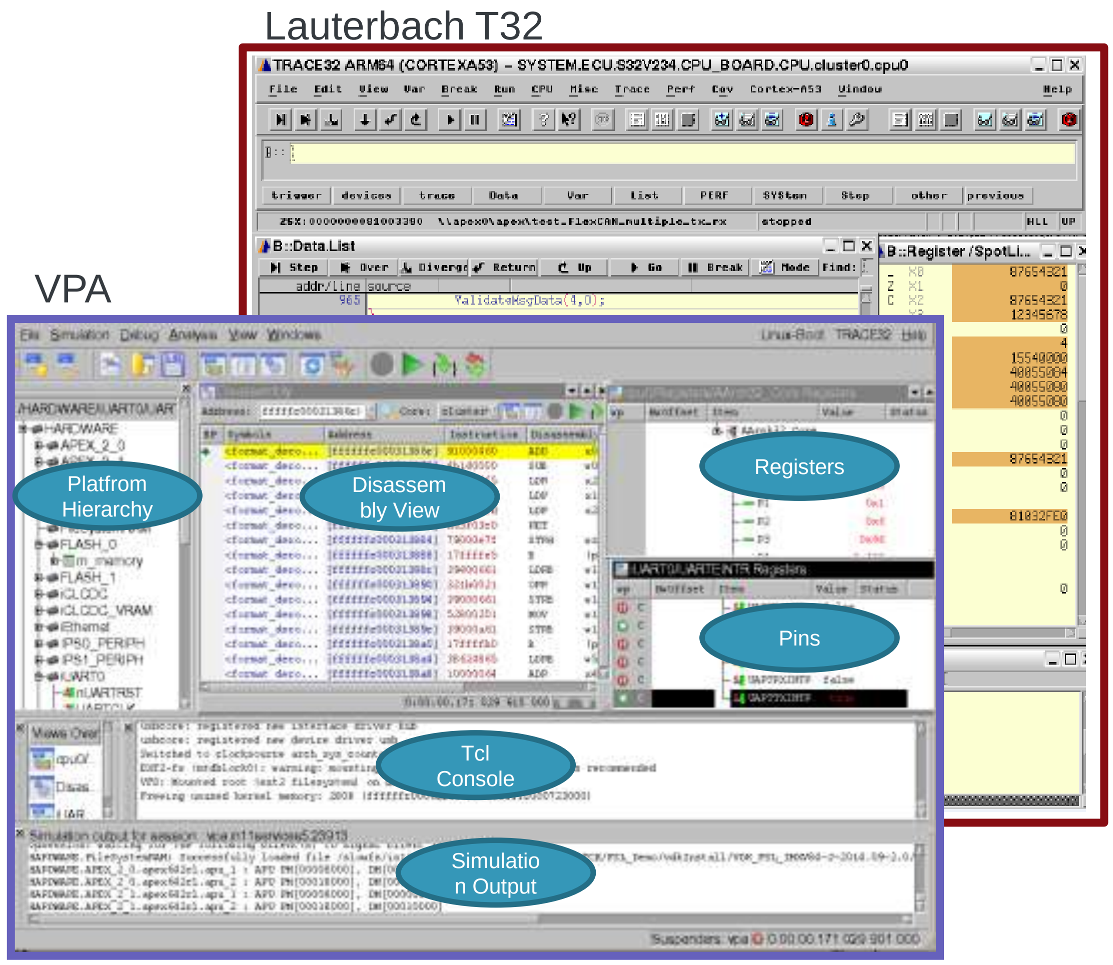
License and distribution through Synopsys

- Center of Excellence for joint model setup

Setup

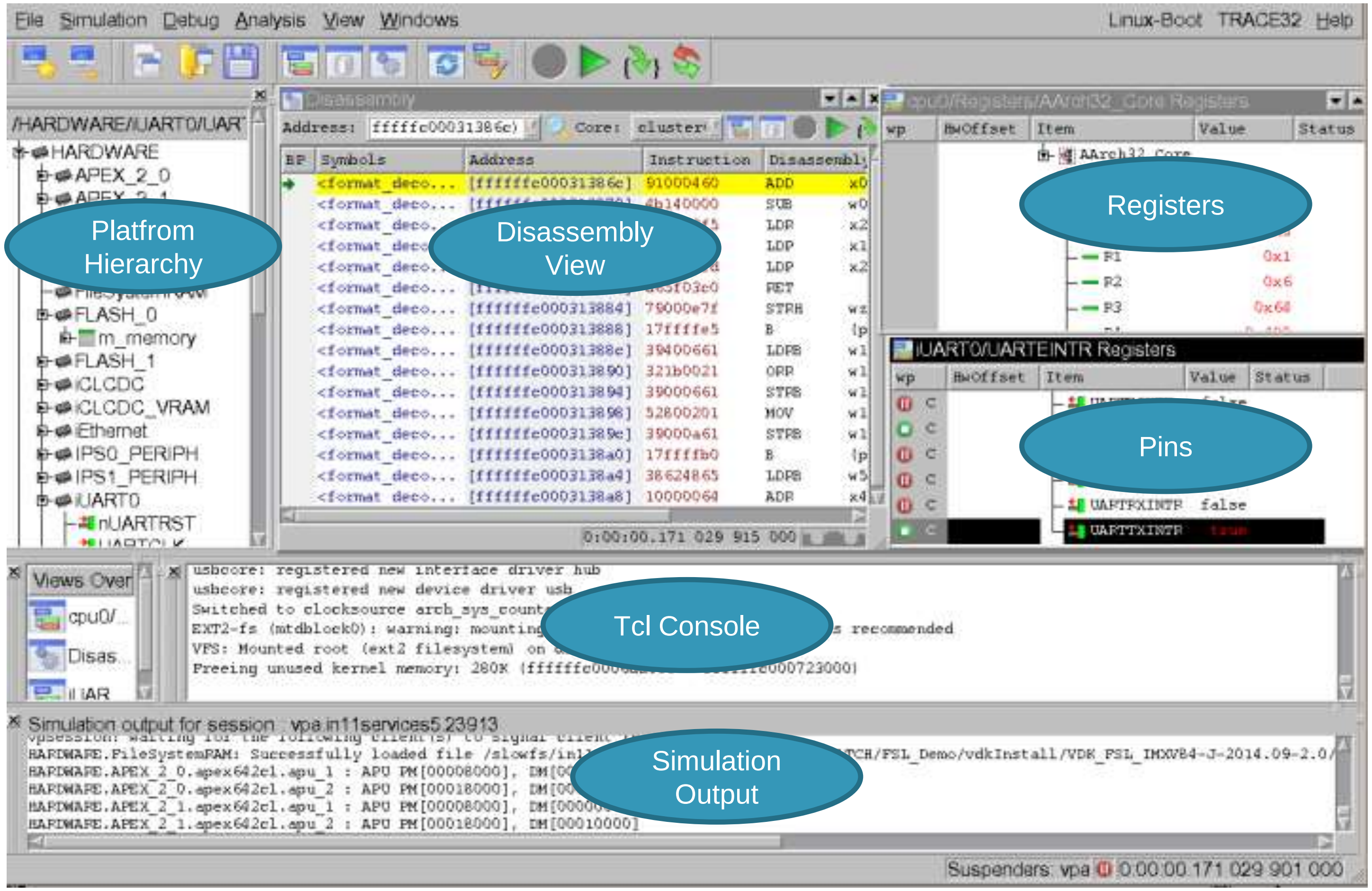
- Windows and Linux based solution

- Point of sales: Synopsys
- Contact: Marc Serugetthi



Virtual Platform Front End

„The GUI to the EVB“



```
Address: fffffc00031386c)   Core: cluster
```

BP	Symbols	Address	Instruction	Disassembly
➜	<format_deco...	[ffffffe00031386e]	91000460	ADD x0
	<format_deco...	[illegible]	4b140000	SUB w0
	<format_deco...	[illegible]	[illegible]	LDR x2
	<format_deco...	[illegible]	[illegible]	LDP x1
	<format_deco...	[illegible]	[illegible]	LDP x2
	<format_deco...	[illegible]	[illegible]03c0	RET
	<format_deco...	[ffffffe000313884]	79000e7f	STRH w
	<format_deco...	[ffffffe000313888]	17ffffe5	B (p
	<format_deco...	[ffffffe00031388e]	39400661	LDRB w1
	<format_deco...	[ffffffe000313890]	321b0021	ORR w1
	<format_deco...	[ffffffe000313894]	39000661	STRB w1
	<format_deco...	[ffffffe000313898]	52800201	MOV w1
	<format_deco...	[ffffffe00031389e]	39000a61	STRB w1
	<format_deco...	[ffffffe0003138a0]	17ffffb0	B (p
	<format_deco...	[ffffffe0003138a4]	38624865	LDRB w5
	<format_deco...	[ffffffe0003138a8]	10000064	ADR x4

```
0:00:00.171 029 915 000
```

wp	HwOffset	Item	Value	Status
		AArch32_Core		
		R1	0x1	
		R2	0x6	
		R3	0x68	

wp	HwOffset	Item	Value	Status
		UARTRXINTR	false	
		UARTTXINTR	true	

```
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
Switched to clocksource arch_sys_count
EXT2-fs (mtdblock0): warning: mounting                s recommended
VFS: Mounted root (ext2 filesystem) on
Freeing unused kernel memory: 280K (ffffffc000              ffc000723000)
```

Simulation output for session : vpa.in11services5.23913

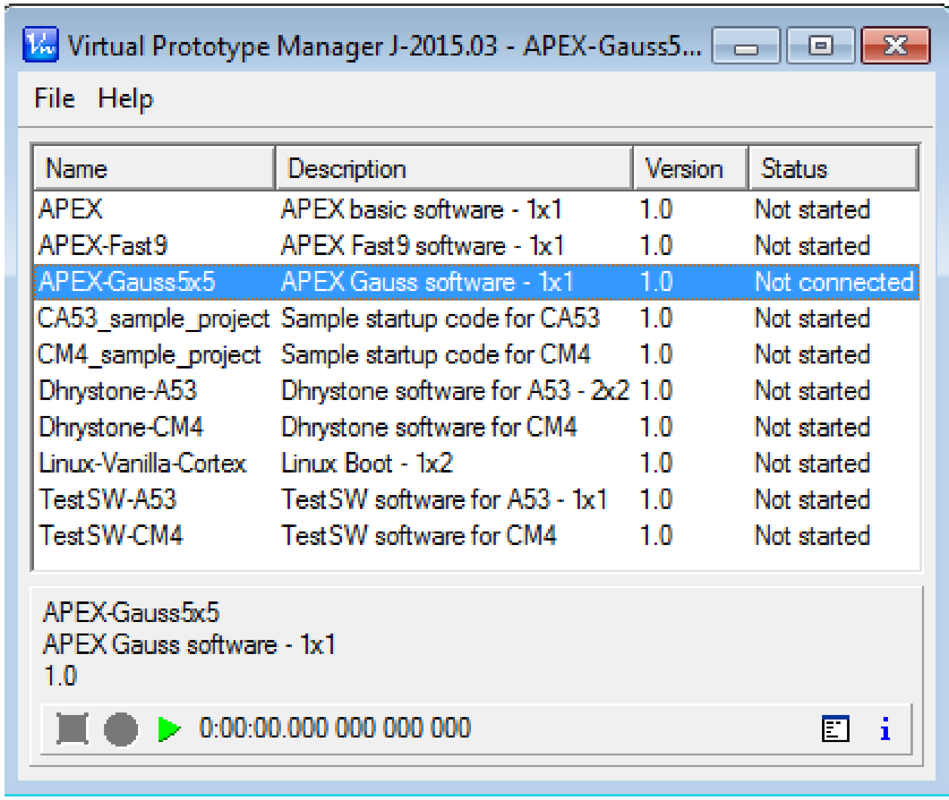
```
vpsession: waiting for the following client(s) to signal client
HARDWARE.FileSystemRAM: Successfully loaded file /slowfs/in1              TCH/FSL_Demo/vdkInstall/VDK_FSL_IMXV84-J-2014.09-2.0/
HARDWARE.APEX_2_0.apex642cl.apu_1 : APU PM[00008000], DM[00
HARDWARE.APEX_2_0.apex642cl.apu_2 : APU PM[00018000], DM[00
HARDWARE.APEX_2_1.apex642cl.apu_1 : APU PM[00008000], DM[0000
HARDWARE.APEX_2_1.apex642cl.apu_2 : APU PM[00018000], DM[00010000]
```

Suspenders: vpa 0:00:00.171 029 901 000

Release Content

- Virtual Prototype installed
- Set of Sample core for getting started → Ease of start
- Scriptable environment of own setups
- ECU level integration by Synopsys

→ See Synopsys Pedestal in tech lab



Name	Description	Version	Status
APEX	APEX basic software - 1x1	1.0	Not started
APEX-Fast9	APEX Fast9 software - 1x1	1.0	Not started
APEX-Gauss5x5	APEX Gauss software - 1x1	1.0	Not connected
CA53_sample_project	Sample startup code for CA53	1.0	Not started
CM4_sample_project	Sample startup code for CM4	1.0	Not started
Dhrystone-A53	Dhrystone software for A53 - 2x2	1.0	Not started
Dhrystone-CM4	Dhrystone software for CM4	1.0	Not started
Linux-Vanilla-Cortex	Linux Boot - 1x2	1.0	Not started
TestSW-A53	TestSW software for A53 - 1x1	1.0	Not started
TestSW-CM4	TestSW software for CM4	1.0	Not started

APEX-Gauss5x5
APEX Gauss software - 1x1
1.0

0:00:00.000 000 000 000

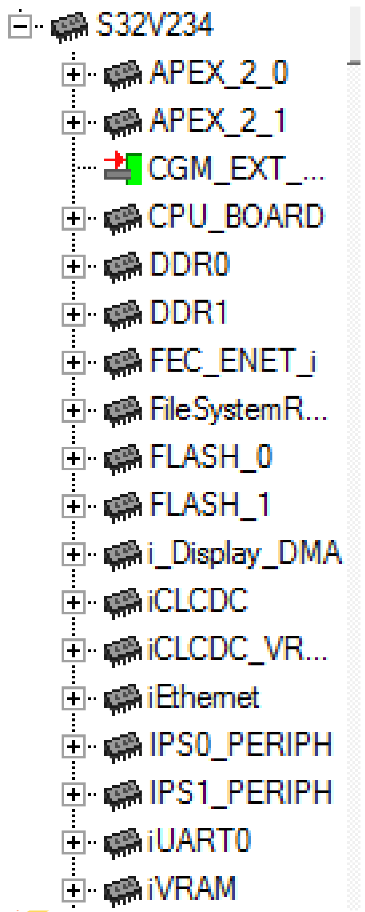
S32V234 Virtual Prototype Model Content R3.3.1 EA

- Functional SW Simulator Content
  - Cores
    - **4 x A53 FastISS cores → by ARM**
    - **Dual APEX2 → by Cognivue**
    - GIC-400
    - CCI-400
- Memories
  - Functional memories
- Peripherals
    - FEC_ENET AVB
    - MC_CGM, MC_ME, MC_RGM
    - PLLDID
    - DFS
    - SRC
    - FlexCAN-FD
    - SIUL-GPIO
    - STM
    - SWT
    - Stubs (QOS, QuadSPI, PDI, SDHC, MMDC0)

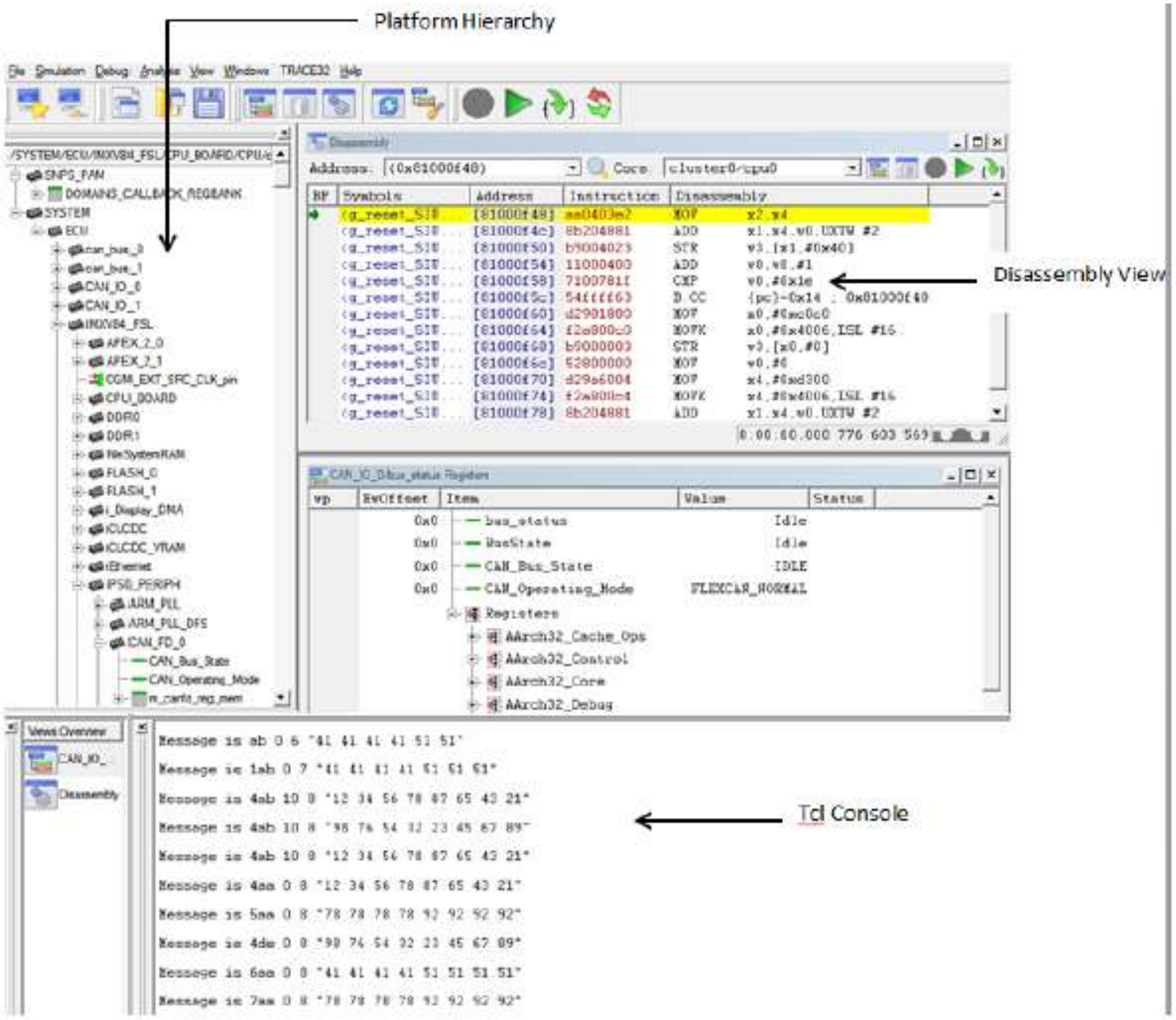
- Debugger
  - Lauterbach32
    - Full Multi-Core
- Connectivity
  - Framebuffer viewer
  - CAN I/O
  - Ethernet I/O
    - Host interface
  - Host Disk Mapping
  - UART console for Linux prompt
- VPA SW Analysis
  - VPA build in SW trace
  - HW breakpointing
- Memory load
  - Via TCL commands
- General scriptability

Tool Example

- VP Analyzer Front End



```
S32V234
  APEX_2_0
  APEX_2_1
  CGM_EXT_...
  CPU_BOARD
  DDR0
  DDR1
  FEC_ENET_i
  FileSystemR...
  FLASH_0
  FLASH_1
  i_Display_DMA
  iCLCDC
  iCLCDC_VR...
  iEthernet
  IPS0_PERIPH
  IPS1_PERIPH
  iUART0
  iVRAM
```



```
/SYSTEM/ECU/INXV84_FSL/CPU_BOARD/CPU/c
SNPS_PAM
  DOMAINS_CALLBACK_REGBANK
SYSTEM
  ECU
    can_bus_0
    can_bus_1
    CAN_IO_0
    CAN_IO_1
    INXV84_FSL
      APEX_2_0
      APEX_2_1
      CGM_EXT_SRC_CLK_pin
      CPU_BOARD
      DDR0
      DDR1
      FileSystemRAM
      FLASH_0
      FLASH_1
      i_Display_DMA
      iCLCDC
      iCLCDC_VRAM
      iEthernet
      IPS0_PERIPH
        ARM_PLL
        ARM_PLL_DFS
        CAN_FD_0
          CAN_Bus_State
          CAN_Operating_Mode
          m_canfd_reg_mem
```

```
Disassembly
Address: (0x81000f48)          Core: cluster0/cpu0

BP  Symbols           Address      Instruction  Disassembly
→   <g_reset_SIT ...  [81000f48]   aa0403e2     MOV   x2,x4
    <g_reset_SIT ...  [81000f4c]   8b204881     ADD   x1,x4,w0,UXTW #2
    <g_reset_SIT ...  [81000f50]   b9004023     STR   w3,[x1,#0x40]
    <g_reset_SIT ...  [81000f54]   11000400     ADD   w0,w0,#1
    <g_reset_SIT ...  [81000f58]   7100781f     CMP   w0,#0x1e
    <g_reset_SIT ...  [81000f5c]   54ffff63     B.CC  {pc}-0x14 ; 0x81000f48
    <g_reset_SIT ...  [81000f60]   d2901800     MOV   x0,#0xc0c0
    <g_reset_SIT ...  [81000f64]   f2a800c0     MOVK  x0,#0x4006,LSL #16
    <g_reset_SIT ...  [81000f68]   b9000003     STR   w3,[x0,#0]
    <g_reset_SIT ...  [81000f6c]   52800000     MOV   w0,#0
    <g_reset_SIT ...  [81000f70]   d29a6004     MOV   x4,#0xd300
    <g_reset_SIT ...  [81000f74]   f2a800c4     MOVK  x4,#0x4006,LSL #16
    <g_reset_SIT ...  [81000f78]   8b204881     ADD   x1,x4,w0,UXTW #2
                                          0:00:60.000 776 603 569
```

```
CAN_IO_0.bus_status Registers

vp  RvOffset  Item                     Value           Status
    0x0       bus_status               Idle
    0x0       BusState                 Idle
    0x0       CAN_Bus_State            IDLE
    0x0       CAN_Operating_Mode       FLEXCAN_NORMAL
              Registers
                AArch32_Cache_Ops
                AArch32_Control
                AArch32_Core
                AArch32_Debug
```

```
Views Overview
  CAN_IO_
  Disassembly
```

```
Message is ab 0 6 "41 41 41 41 51 51"
Message is 1ab 0 7 "41 41 41 41 51 51 51"
Message is 4ab 10 8 "12 34 56 78 87 65 43 21"
Message is 4ab 10 8 "98 76 54 32 23 45 67 89"
Message is 4ab 10 8 "12 34 56 78 87 65 43 21"
Message is 4aa 0 8 "12 34 56 78 87 65 43 21"
Message is 5aa 0 8 "78 78 78 78 92 92 92 92"
Message is 4da 0 8 "98 76 54 32 23 45 67 89"
Message is 6aa 0 8 "41 41 41 41 51 51 51 51"
Message is 7aa 0 8 "78 78 78 78 92 92 92 92"
```

#FTF2015

SW Team Quote on S32V234

- Hypervisor Development
  - Virtual Prototype enabled pre-silicon SW development of Hypervisor
  - Final porting to silicon required 3 days only

→Significant time to market advantage

→Dual OS + Hypervisor

→See Session FTF-ACC-F1354

Virtualization on ADAS Using XEN

Summary

Virtual Prototype Success Enables

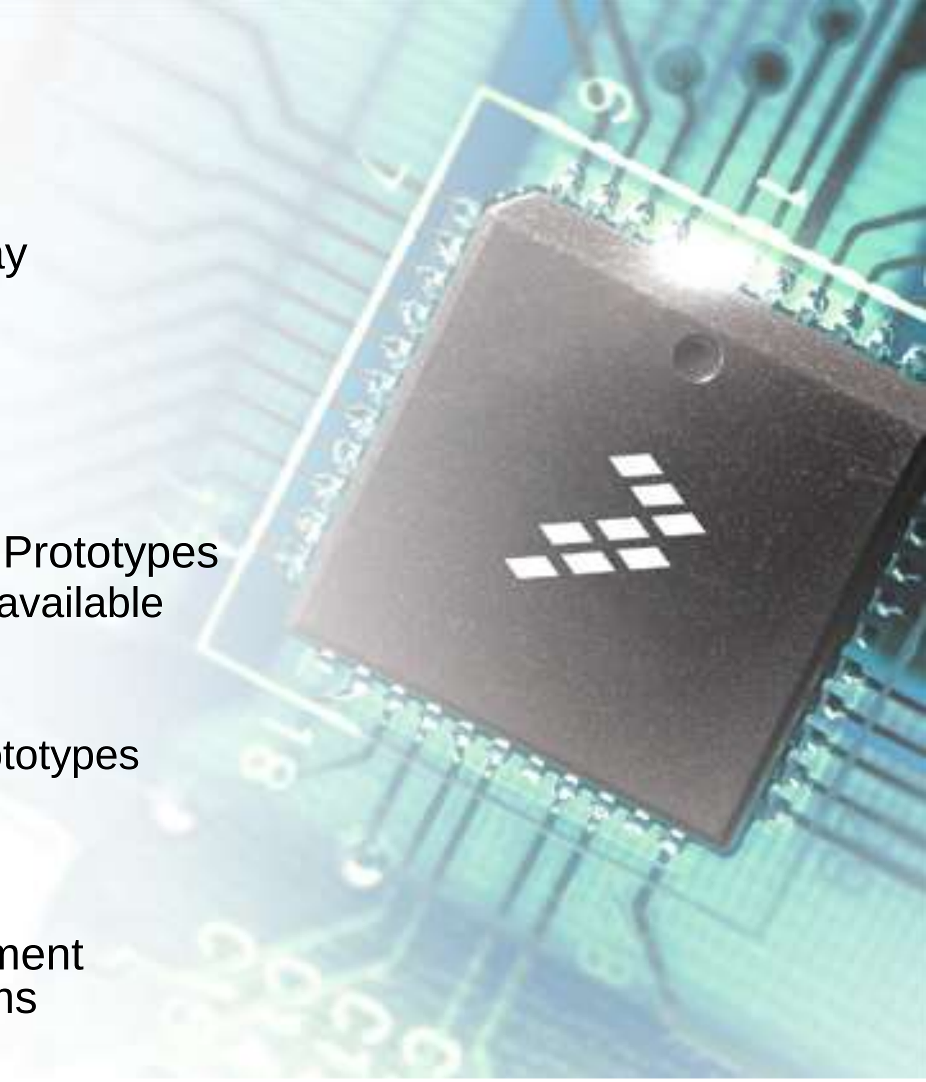
Collaboration enabled

- Eco system setup for
  - Inter company model exchange
- Changed SW development flow for effective virtual platform deployment
- High level model quality to achieve productivity based on Virtual platforms
- Full device models available 6-9 months ahead of silicon

Ahead of us

- Inter-company model exchange
  - Standards don’t support dynamic model plug- in
- Increasing user base for the upcoming SW development tasks, Support and maintenance infrastructure
  - Long model life cycle in automotive
- Change of EDA infrastructure and standards

Summary



- Pre-silicon SW development is possible today
  - Driver development
  - OS porting
  - Validation code development
  - Error Injection

- High Value Performance Analysis on Virtual Prototypes
  - Large instrumentations and HW & SW views available

- Virtual Prototype solution available
  - Tool Vendors and partners support virtual prototypes
  - Synopsys CoE
  - ASTC

- Virtual Prototypes address key SW development challenges Integration with Tool ECO systems

- Collaboration between tools developers, model developers and services providers plus our collective customers is essential to realize the full potential of virtualization from which all can benefit.

Session Summary

Virtual Prototypes of Freescale allow

- Early SW development
- Performance analysis with large HW/SW instrumentations
- Partner enablement through
  - Synopsys
  - ASTC Ltd
- Integration in ECU models

Thank you!

www.Freescale.com