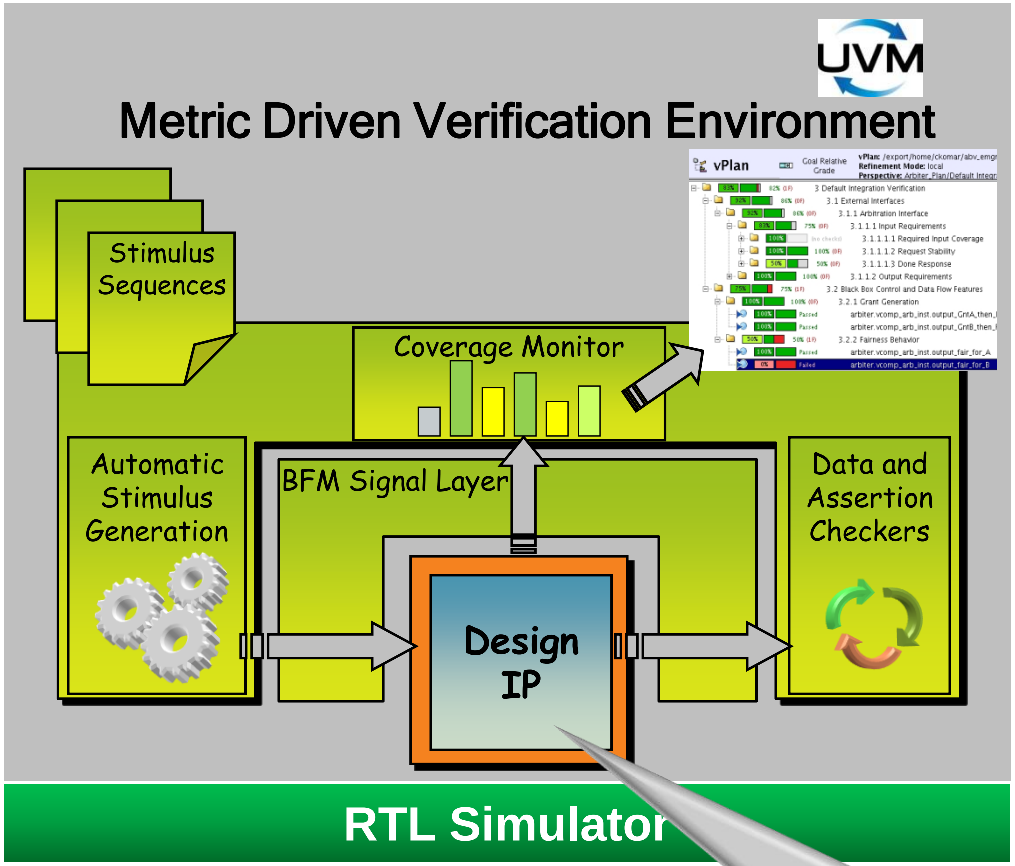
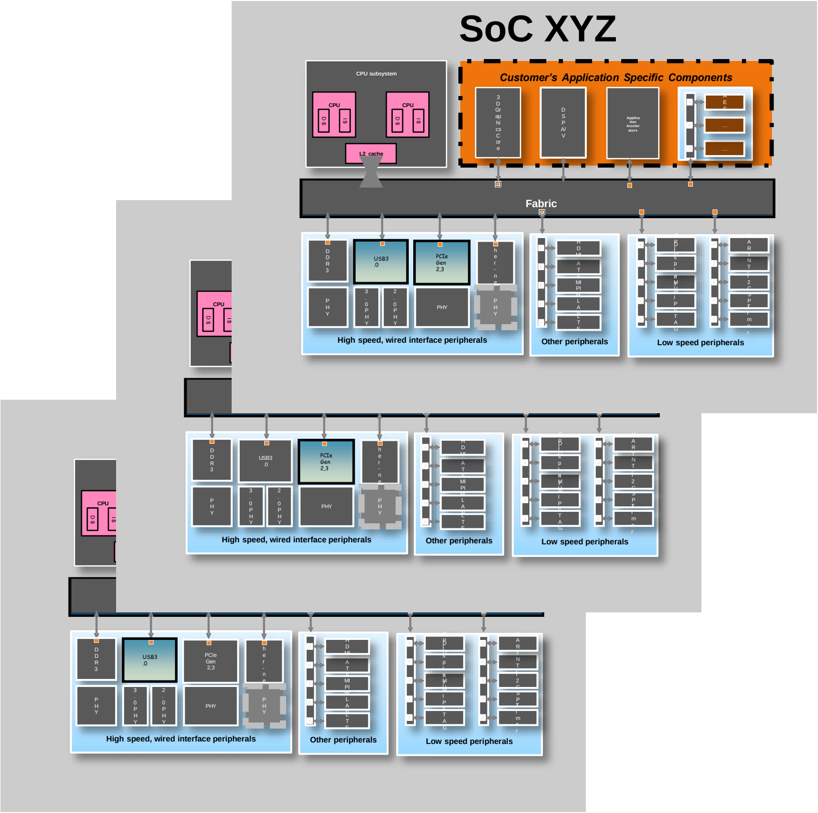
HW/SW Co-Verification Challenges and Solutions

Sandeep Gor – Staff Application Engineer
April 9, 2015

Past 10 years: IP bottom-up verification approach


UVM
Metric Driven Verification Environment
Stimulus Sequences
Coverage Monitor
Automatic Stimulus Generation
BFM Signal Layer
Data and Assertion Checkers
Design IP
RTL Simulator

```
vPlan   Goal Relative Grade   vPlan: /export/home/ckomar/abv_emgr
                              Refinement Mode: local
                              Perspective: Arbiter_Plan/Default Integr
83%   82% (1F)   3 Default Integration Verification
92%   86% (0F)   3.1 External Interfaces
92%   86% (0F)   3.1.1 Arbitration Interface
83%   75% (0F)   3.1.1.1 Input Requirements
100%  (no checks)  3.1.1.1.1 Required Input Coverage
100%  100% (0F)  3.1.1.1.2 Request Stability
50%   50% (0F)   3.1.1.1.3 Done Response
100%  100% (0F)  3.1.1.2 Output Requirements
75%   75% (1F)   3.2 Black Box Control and Data Flow Features
100%  100% (0F)  3.2.1 Grant Generation
100%  Passed     arbiter.vcomp_arb_inst.output_GntA_then_
100%  Passed     arbiter.vcomp_arb_inst.output_GntB_then_
50%   50% (1F)   3.2.2 Fairness Behavior
100%  Passed     arbiter.vcomp_arb_inst.output_fair_for_A
0%    Failed     arbiter.vcomp_arb_inst.output_fair_for_B
```



Comprehensive IP and Sub-System verification: should work in **ANY** SoC context

Paradigm Shift In the Market

- Ubiquity of software
- Move to Standards-based protocols and IPs
- Increasing complexity and # of IP’s to be integrated
- Mixed Signal and Low Power Verification Complexity
- Debug consuming over 50% of verification task
- Application / end-user driven requirements
- Shrinking time-to-money

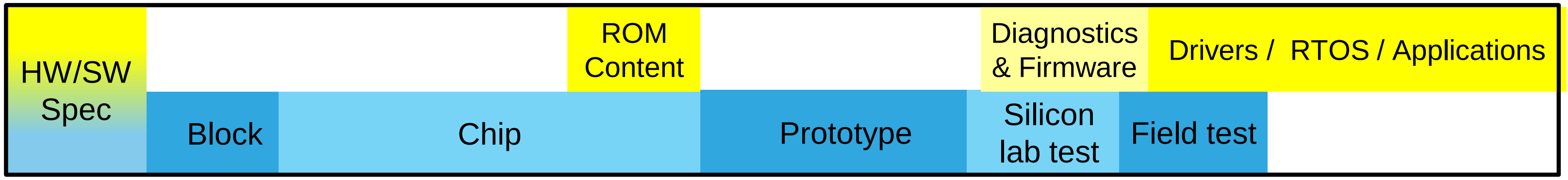
Design Trends

Shift from:
Serial HW/SW Development
To:
Parallel HW/SW Development

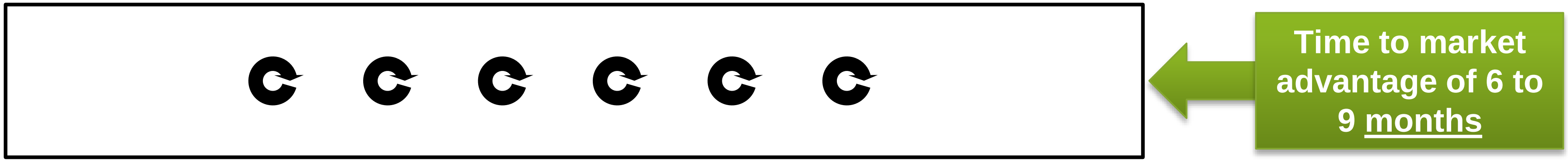
Shift from:
IP Creation
To:
SoC Integration

From serial to parallel HW/SW development

Serial HW->SW Development

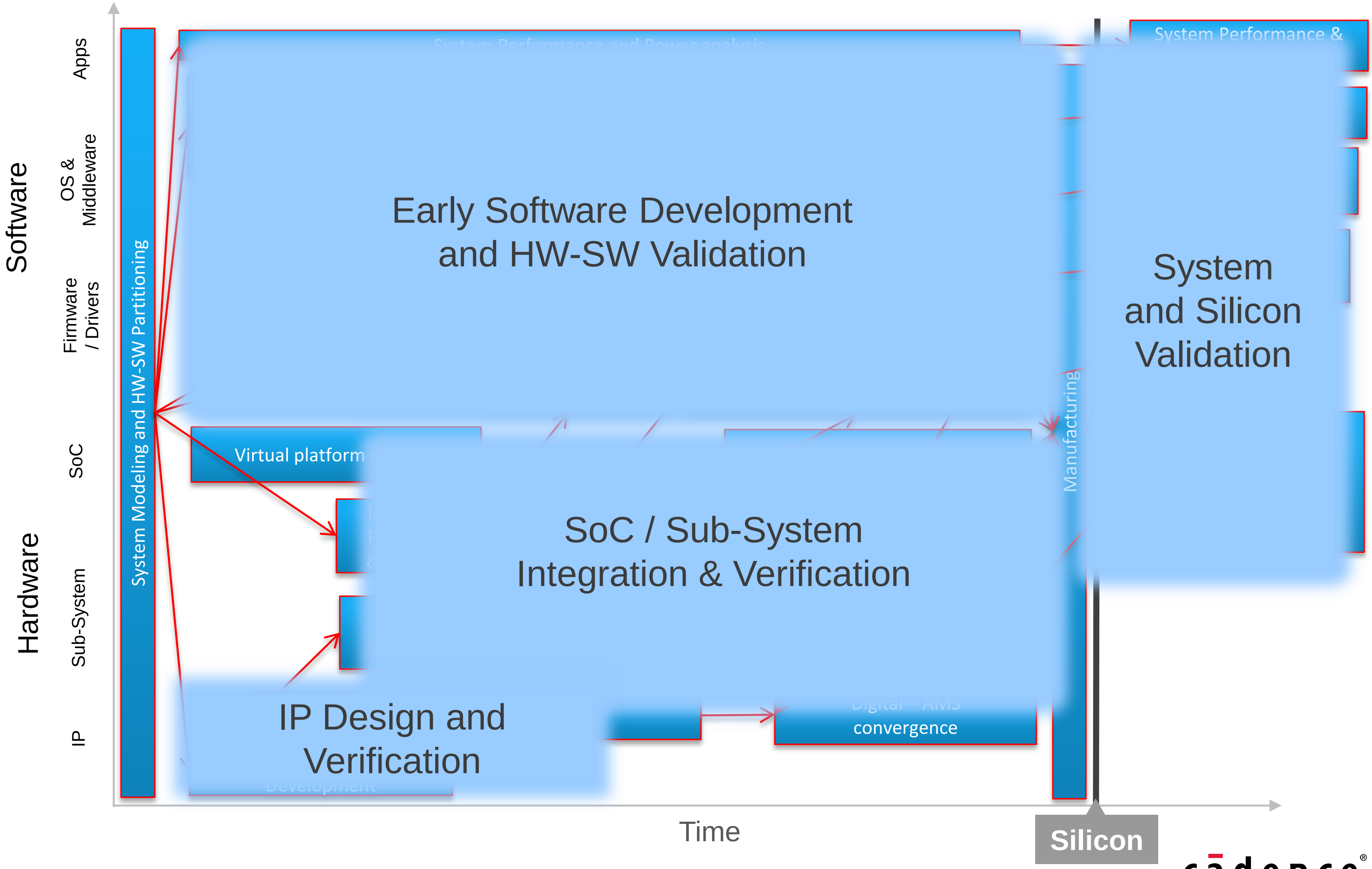


Parallel HW->SW Development



Integrate HW/SW Early and Often
HW designed in SW context
Software exposed to Spec changes

Ever-growing System Development Complexity


Software
Hardware
Apps
OS & Middleware
Firmware / Drivers
SoC
Sub-System
IP
System Modeling and HW-SW Partitioning
System Performance and Power analysis
System Performance &
Virtual platform
Manufacturing
Digital – AMS convergence
Development
Time
Silicon
Early Software Development and HW-SW Validation
System and Silicon Validation
SoC / Sub-System Integration & Verification
IP Design and Verification

Challenges in HW/SW Co-Verification

- HW/SW requires High Performance Platforms
- Multiple disconnected SoC simulation environments
  - Virtual Platform, RTL Sim, HW Acceleration/Emulation, FPGA Prototype, Post Silicon
- Reducing SoC integration time and effort
  - Integrating many design IPs and SW components
  - Requires significant time and effort to verify integration
  - High cost to re-integrate & re-verify changes
  - Debug is a major challenge to isolate the root cause
  - Verification effort for SoC derivatives is too high
- Verifying that SoC can support required SW applications
  - Increased software content to develop, integrate, & verify
  - SoC must be architected up front to support SW Use Cases
  - Must verify against functional, power & performance requirements

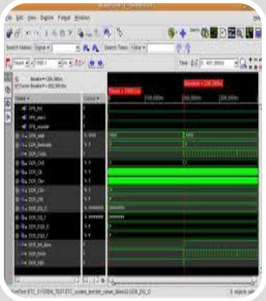
HW/SW Requires High Performance Platforms

SDK

- Highest speed
- Ignore hardware
- Earliest in the flow

Virtual Platform

- Almost at speed
- Less accurate (or slower)
- Before RTL
- Great to debug (but less detail)
- Easy replication



RTL Simulation

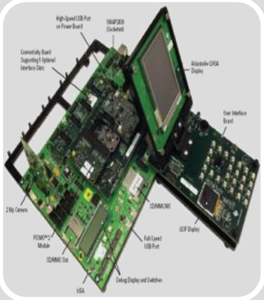
- KHz range
- Accurate
- Excellent HW debug
- Little SW execution

Acceleration Emulation

- MHz Range
- RTL accurate
- After RTL is available
- Good to debug with full detail
- Expensive to replicate

FPGA Prototype

- 10’s of MHz
- RTL accurate
- After stable RTL is available
- OK to debug
- More expensive than software to replicate

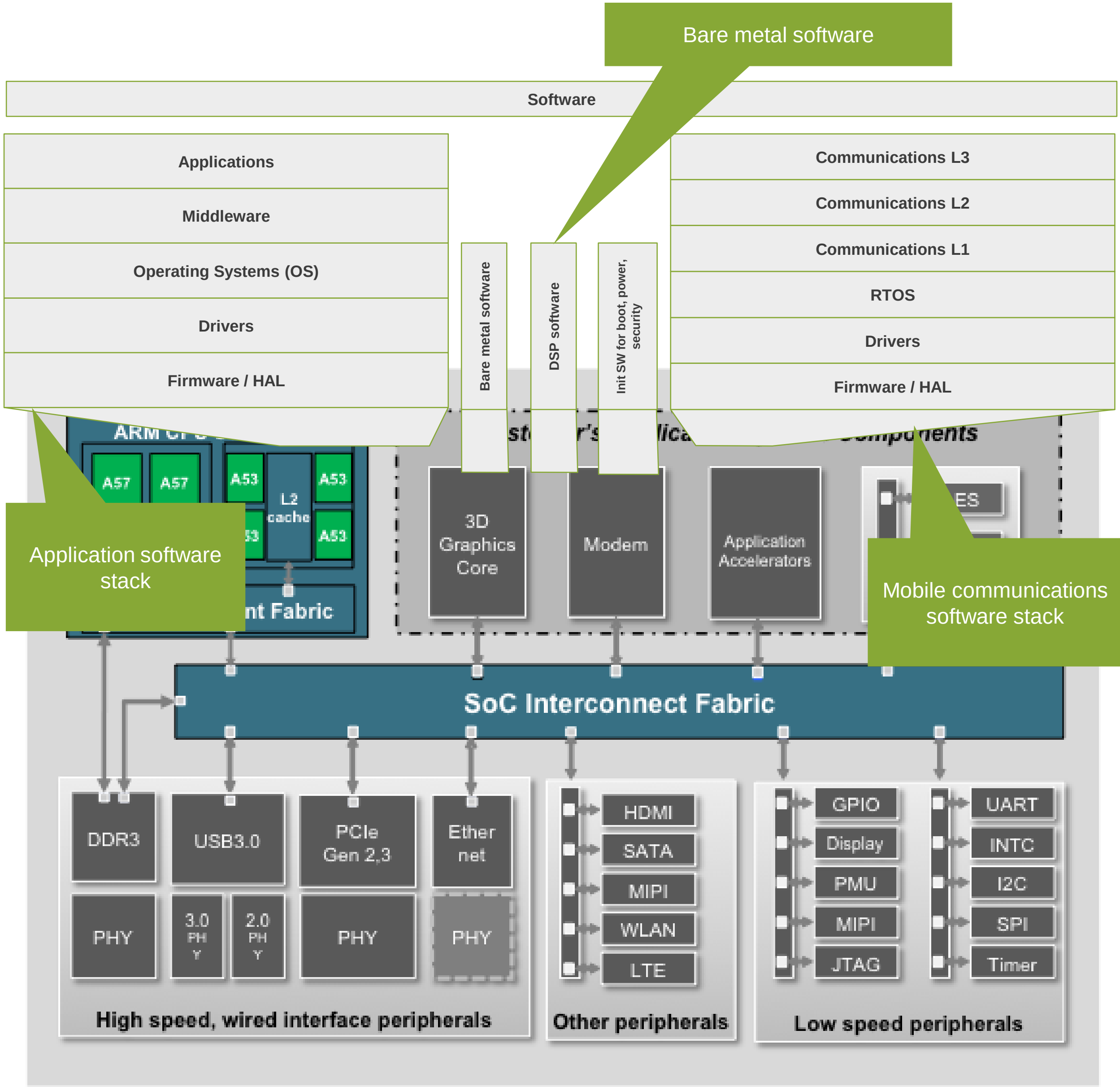


Prototyping Board

- Real time speed
- Fully accurate
- Post Silicon
- Difficult to debug
- Sometimes hard to replicate

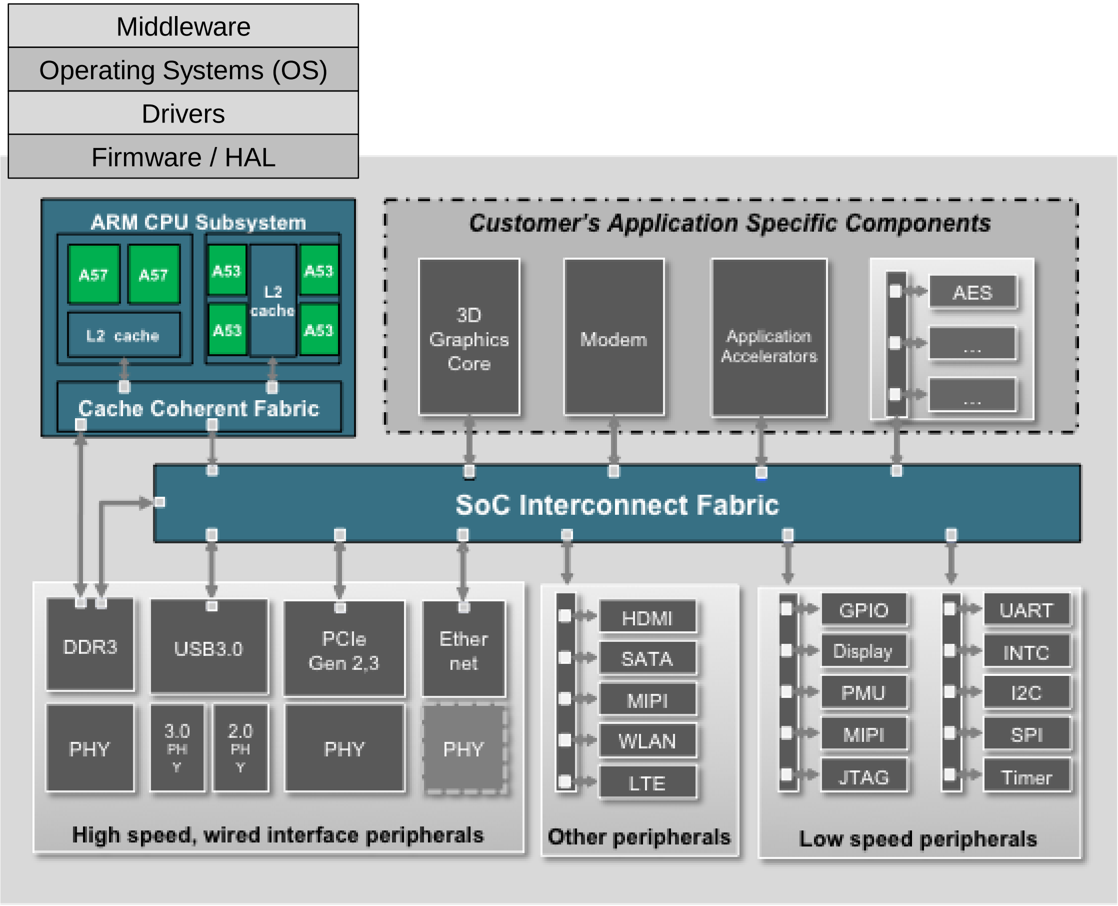
A System-centric Look at a Modern SoC

- Many IPs
  - Standard IO
    - Wifi, USB, PCIe, etc.
  - Standard internal interfaces
    - AXI, ACE etc
  - System infrastructure
    - Interconnect, interrupt control, power mangement, timers…
  - Differentiators
    - custom accelerators, modem…
- Many cores
  - Homogeneous
  - Heterogeneous
- Lots of software
  - Part of core functionality
    - communication stack, DSP software, GPU microcode…
  - User application sw infrastructure
    - Android, Linux…



SoC SW & Verification Challenges and Solutions

Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol

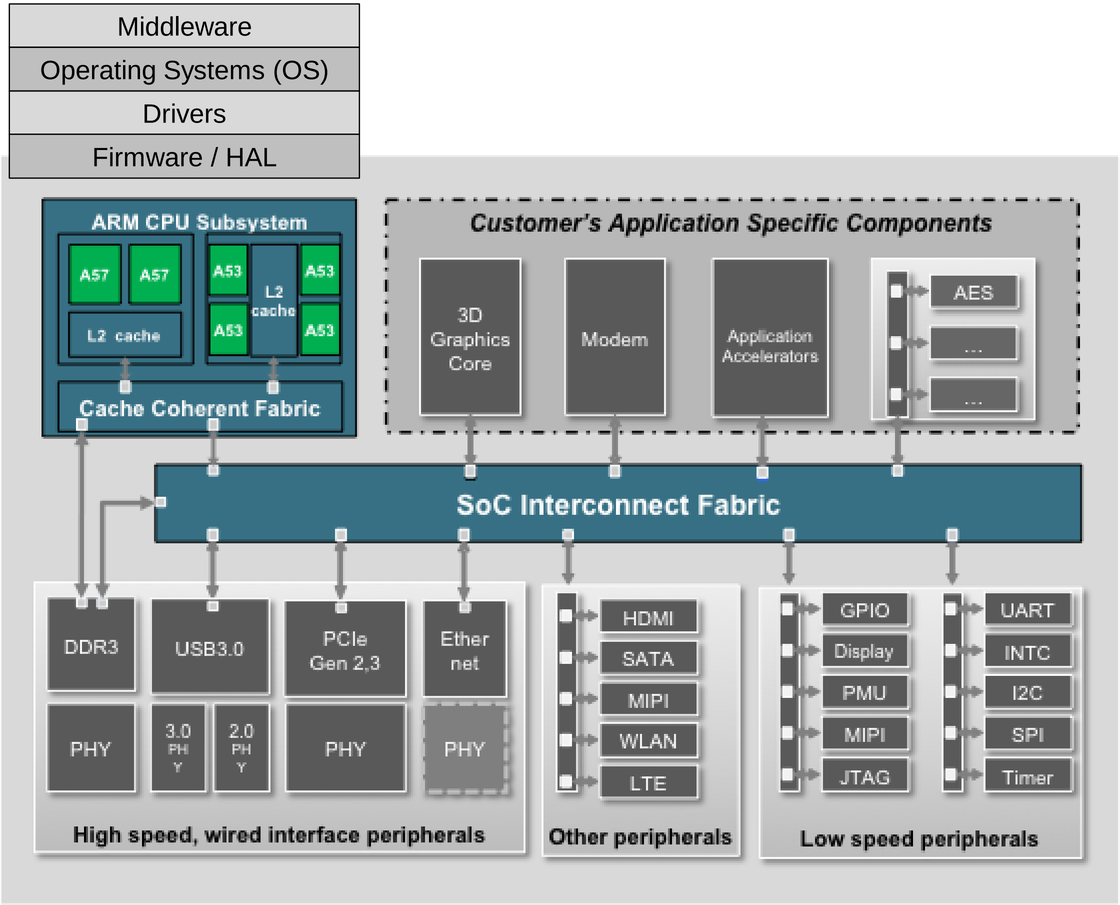


Development flow needs to support HW/SW integration & verification environments on multiple platforms

Planning & Management	Virtual Platform	Formal Tools	Simulators	Emulators	Prototyping Platform	Debug Tools

SoC SW & Verification Challenges and Solutions

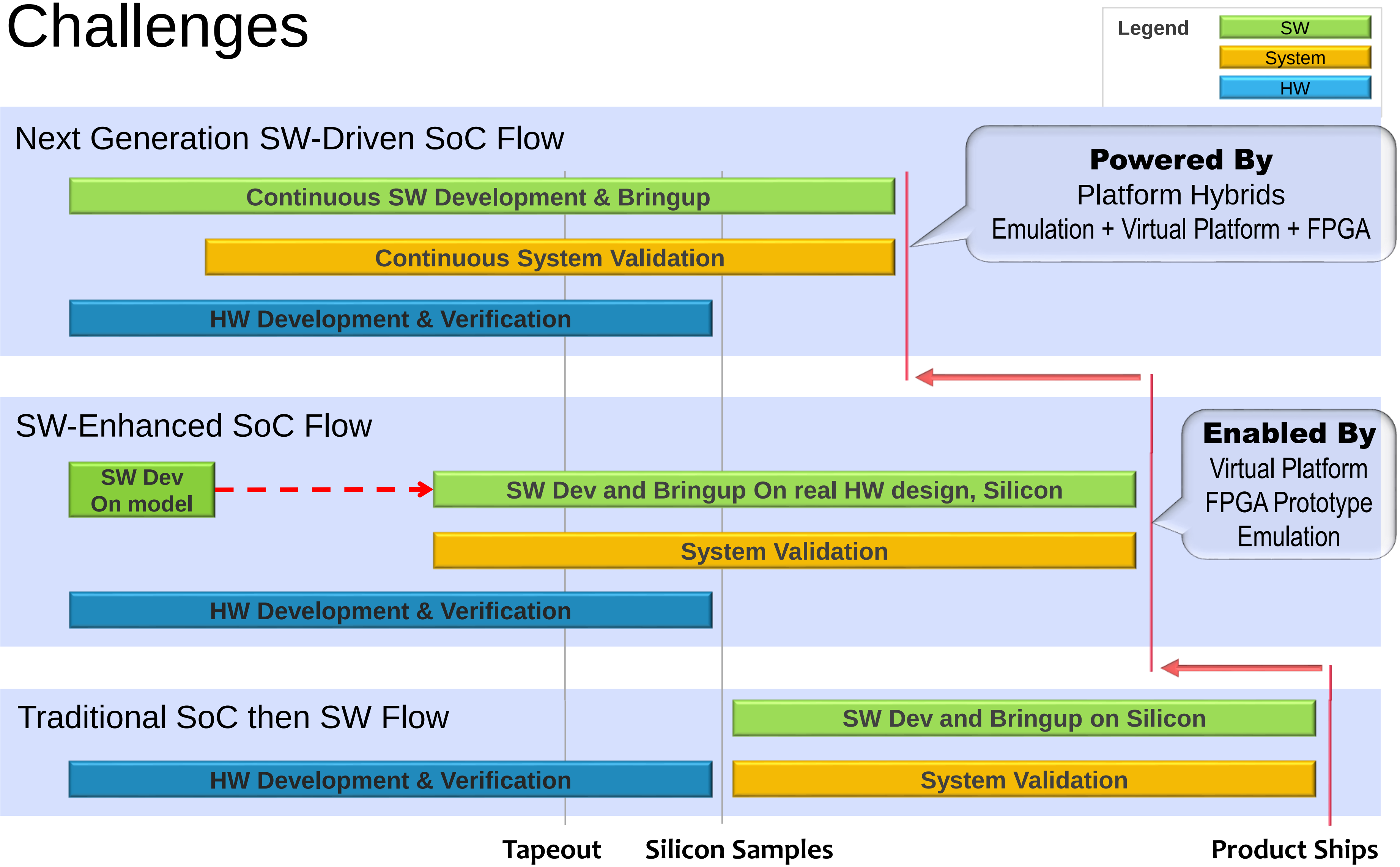
Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol



Development flow needs to support HW/SW integration & verification environments on multiple platforms

- Planning & Management
- Virtual Platform
- Formal Tools
- Simulators
- Emulators
- Prototyping Platform
- Debug Tools

Full OS level and above SW Validation Challenges



Legend: 🟩 SW, 🟨 System, 🟦 HW

Next Generation SW-Driven SoC Flow

Continuous SW Development & Bringup
Continuous System Validation
HW Development & Verification

Powered By
Platform Hybrids
Emulation + Virtual Platform + FPGA

SW-Enhanced SoC Flow

SW Dev On model
SW Dev and Bringup On real HW design, Silicon
System Validation
HW Development & Verification

Enabled By
Virtual Platform
FPGA Prototype
Emulation

Traditional SoC then SW Flow

SW Dev and Bringup on Silicon
HW Development & Verification
System Validation
Tapeout
Silicon Samples
Product Ships

cadence®

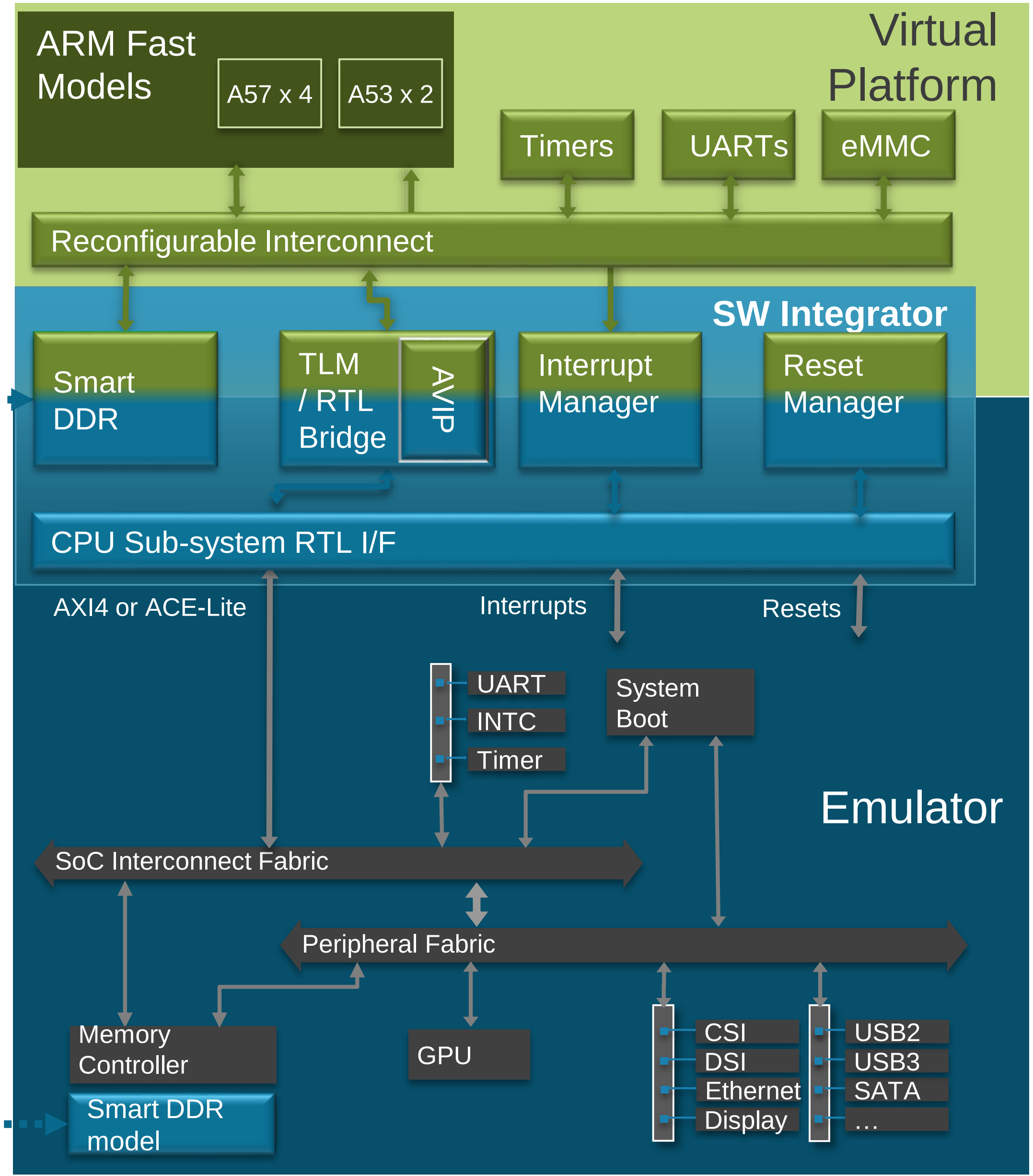
ARMv8/v7 Early OS & Software Bring Up

Execute SW at 100MHz
With standard or custom processor models

Plug and Play Integration with RTL
SoC-specific transactors and RTL I/F

Validate SoC + OS at 5-10 MHz
High-performance memory coherency

Reduce SoC Debug Effort
System Messages
HW / SW Debuggers

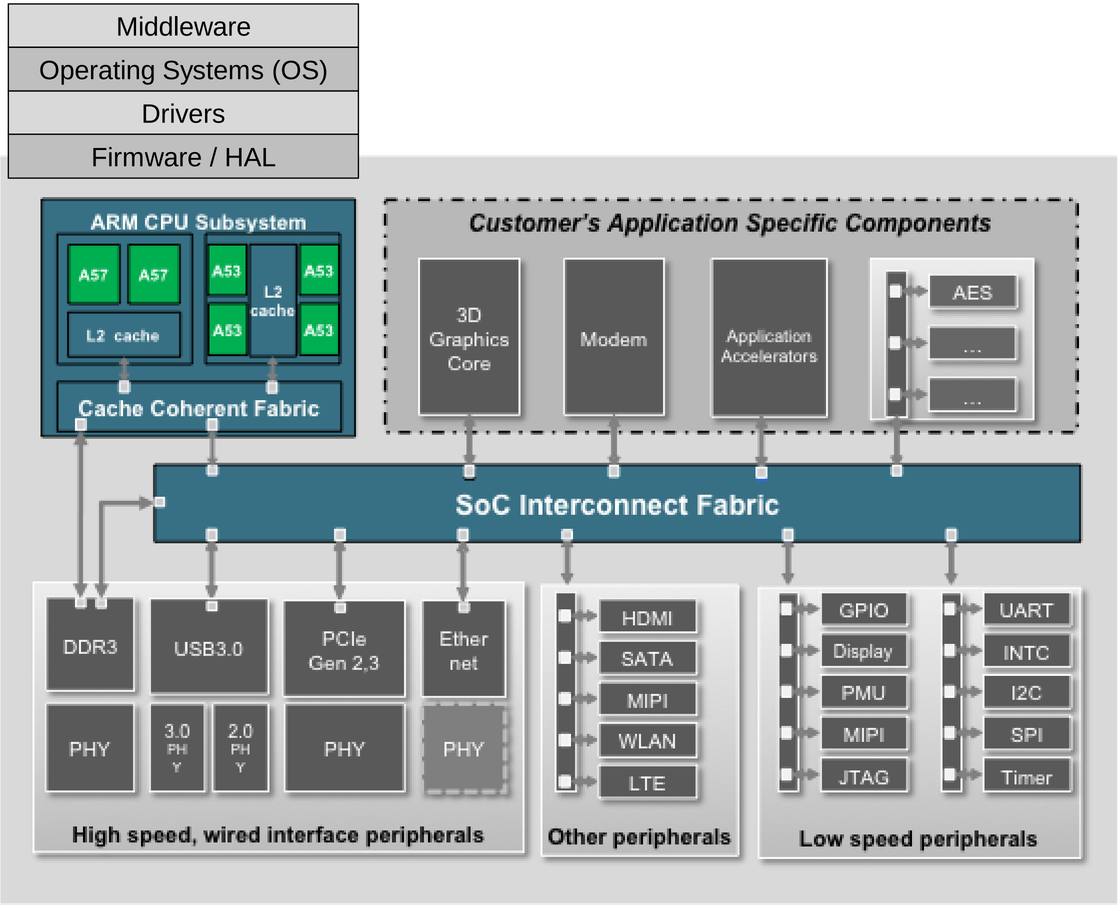


Component Color Key

Customer	RTL	⬛
	RTL	🟦
	TLM	🟩
	Mem I/F	◀ ··· ▶

SoC SW & Verification Challenges and Solutions

Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol



Development flow needs to support HW/SW integration & verification environments on multiple platforms

Planning & Management

Virtual Platform

Formal Tools

Simulators

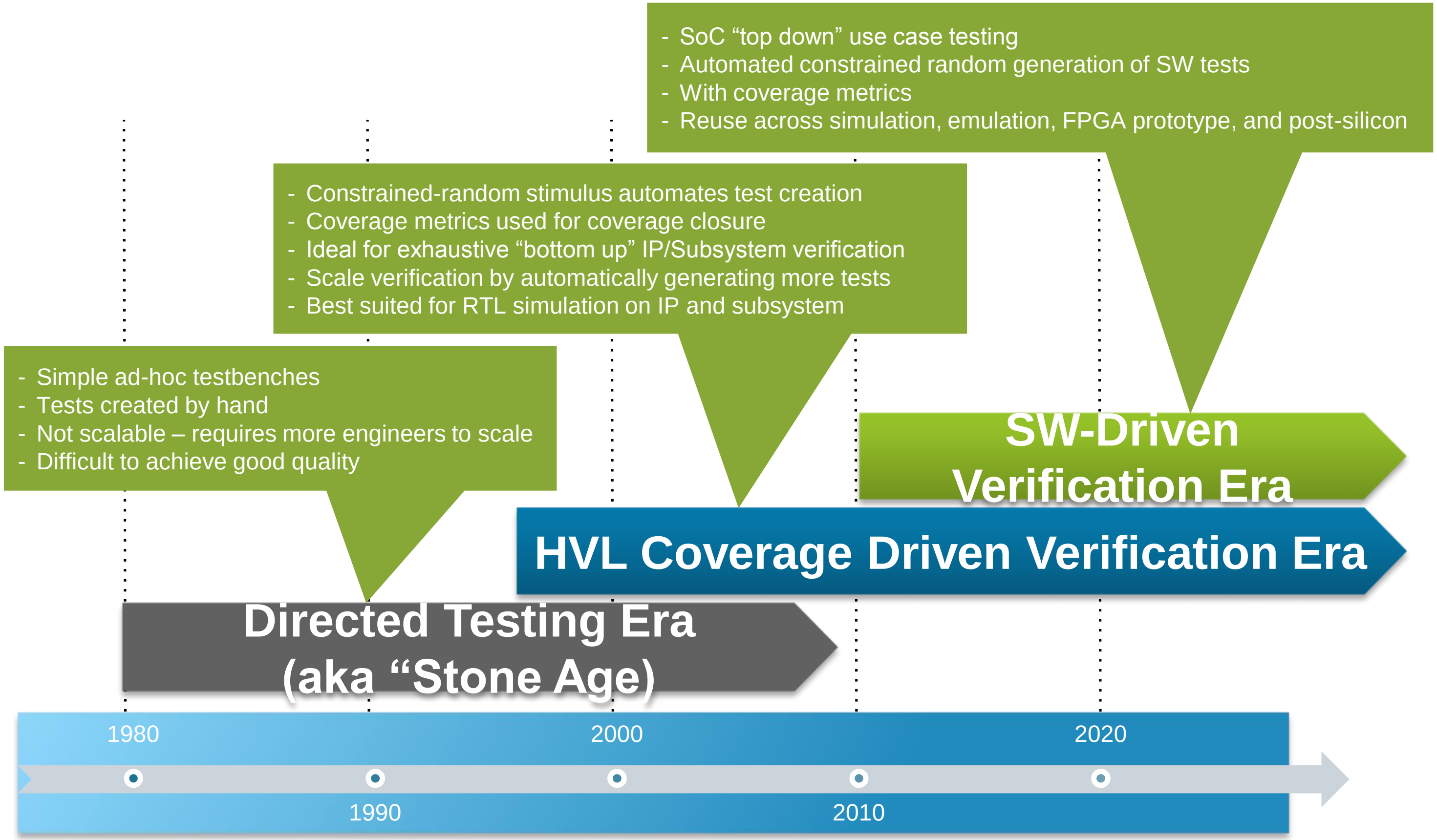
Emulators

Prototyping Platform

Debug Tools

The Eras of Verification

Looking at the past and into the future



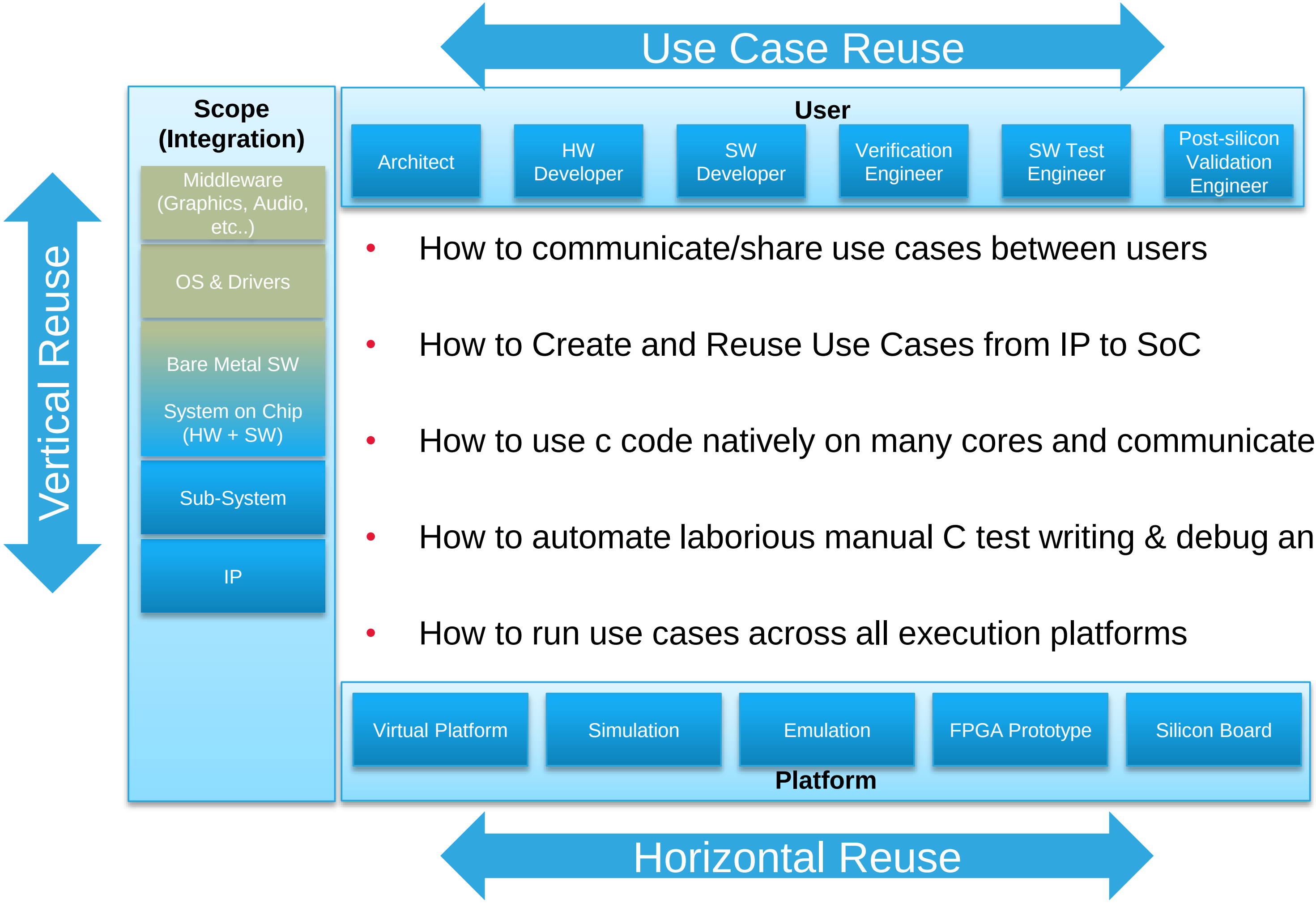
- SoC “top down” use case testing
- Automated constrained random generation of SW tests
- With coverage metrics
- Reuse across simulation, emulation, FPGA prototype, and post-silicon

- Constrained-random stimulus automates test creation
- Coverage metrics used for coverage closure
- Ideal for exhaustive “bottom up” IP/Subsystem verification
- Scale verification by automatically generating more tests
- Best suited for RTL simulation on IP and subsystem

- Simple ad-hoc testbenches
- Tests created by hand
- Not scalable – requires more engineers to scale
- Difficult to achieve good quality

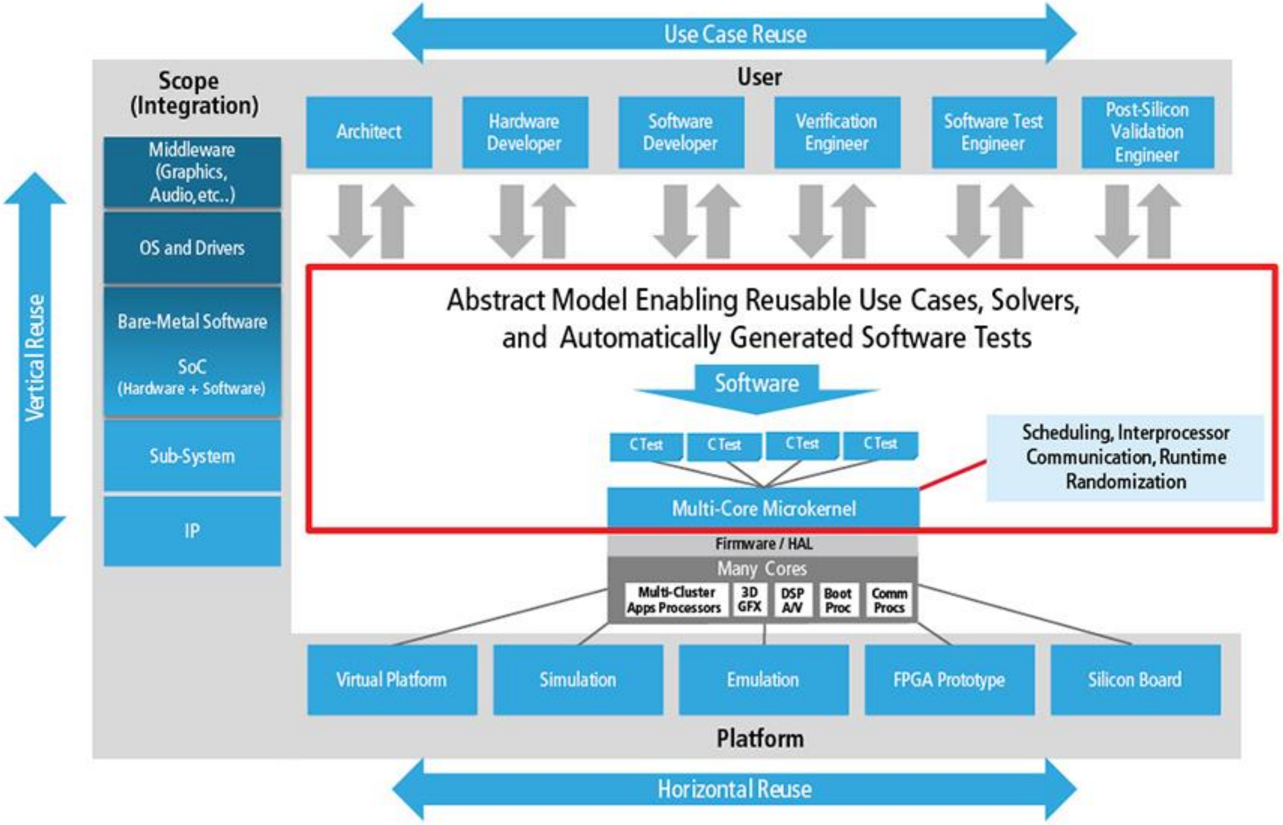
SW-Driven Verification Era
HVL Coverage Driven Verification Era
Directed Testing Era (aka “Stone Age)
1980
1990
2000
2010
2020

SoC level Verification & Validation Requirements



- How to communicate/share use cases between users
- How to Create and Reuse Use Cases from IP to SoC
- How to use c code natively on many cores and communicate between cores
- How to automate laborious manual C test writing & debug and add metrics
- How to run use cases across all execution platforms

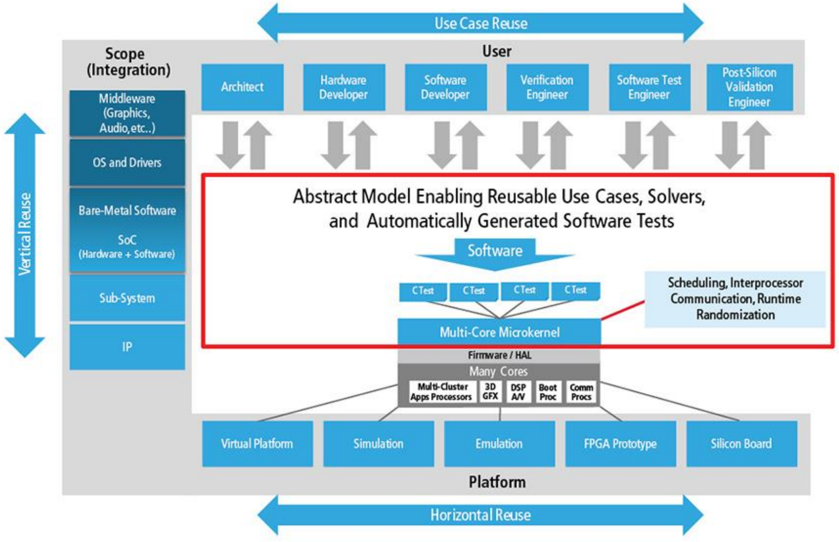
Solution : Test Creation at Abstract Level



Perspec System Verifier Technology

Solution

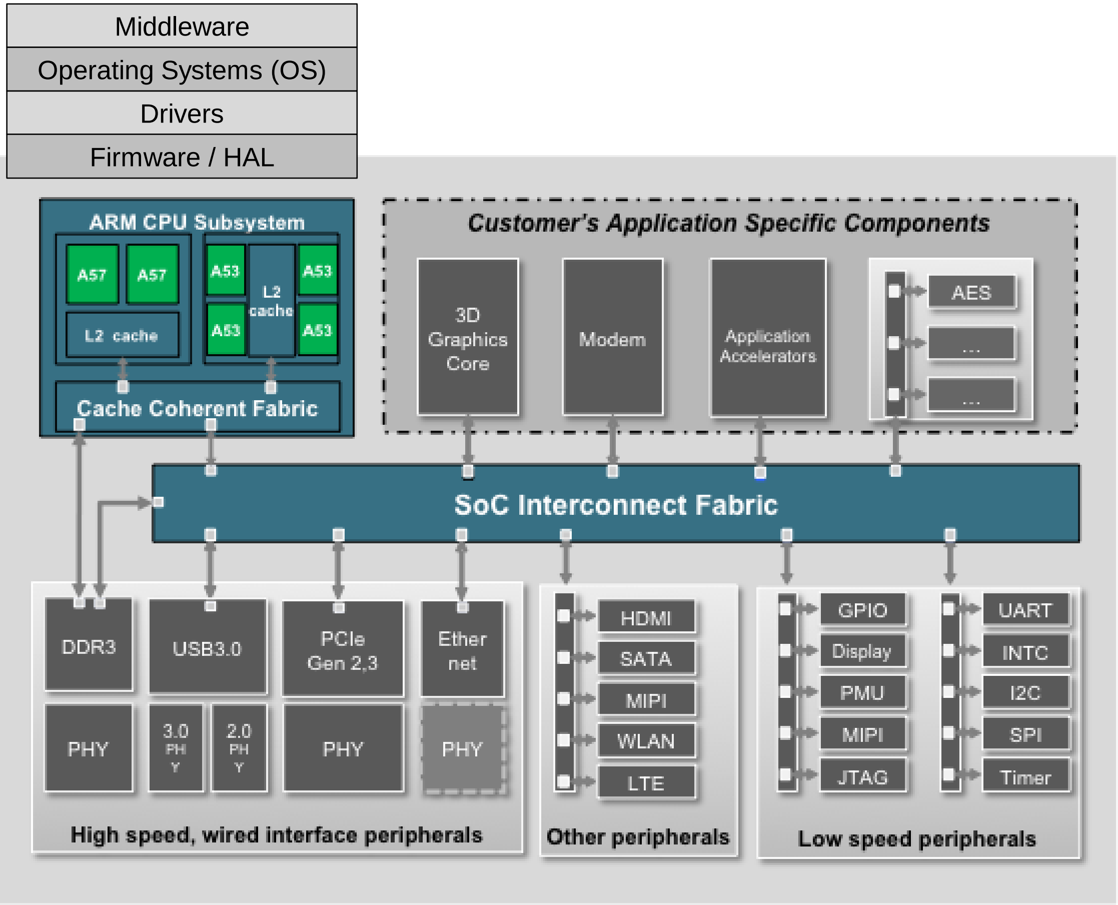
- Faster complex SoC test creation
- Abstraction: UML style use-case diagrams
- Automation: system use-case test generation
- Portability: reuse across all execution platforms
- Measurement: SoC-level HW/SW coverage metrics



Perspec System Verifier Technology

SoC SW & Verification Challenges and Solutions

Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol



Development flow needs to support HW/SW integration & verification environments on multiple platforms

Planning & Management

Virtual Platform

Formal Tools

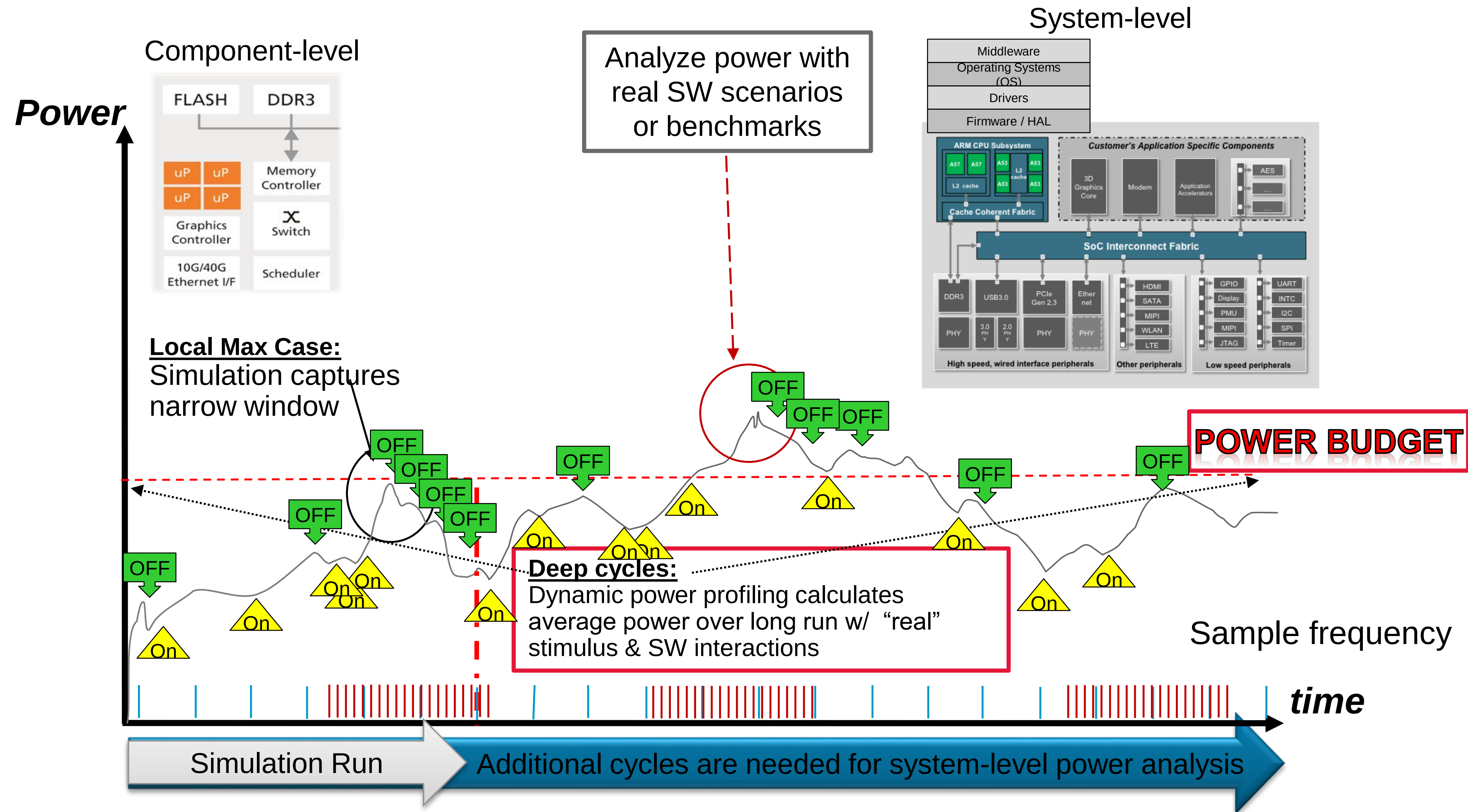
Simulators

Emulators

Prototyping Platform

Debug Tools

SoC Power Analysis Requires "*Deep*" Cycles
(@100MHz for 10 secs => 1 Billion cycles)



Analyze power with real SW scenarios or benchmarks

Local Max Case:
Simulation captures narrow window

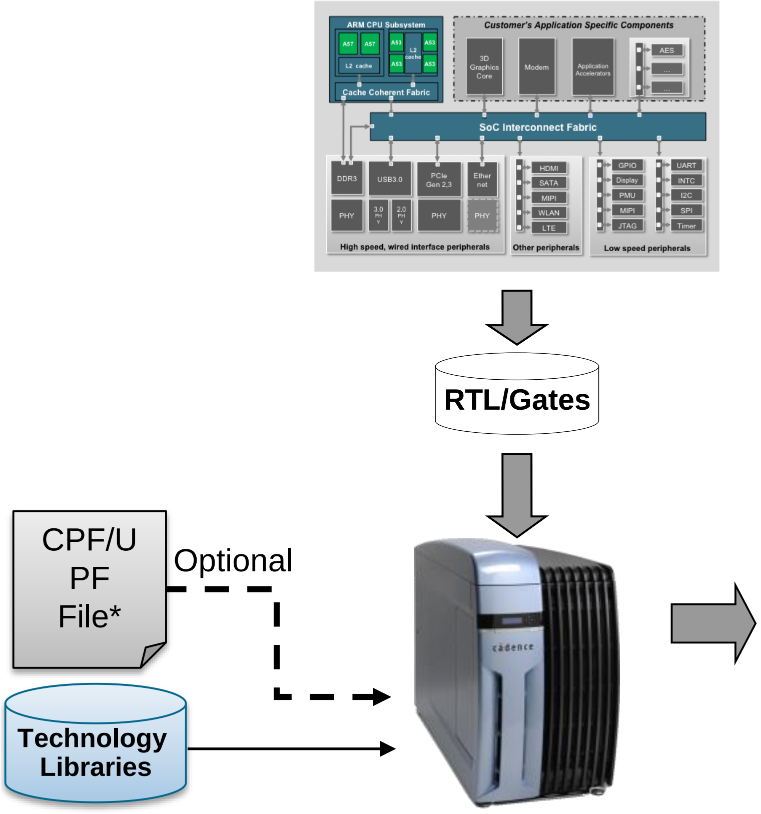
Deep cycles:
Dynamic power profiling calculates average power over long run w/ "real" stimulus & SW interactions

Simulation Run

Additional cycles are needed for system-level power analysis

Identify and analyze peak and average power at system level

Dynamic Power Analysis



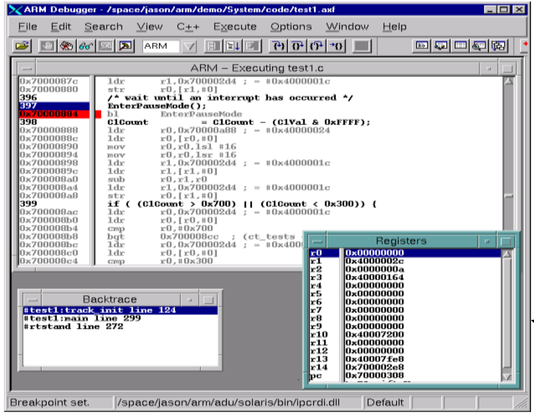
Emulator

Dynamic Power Analysis

- Peak and average power consumption over any time window
- Instance-based power navigation
- System-level power analysis for large SoC (with software)
- MHz Throughput
- No emulation capacity overhead
- Higher accuracy
- Calculate peak and average power

DPA should be Flexible Solution for HW/SW Co-Verification

ARM RealView
Software Debugger



```
0x7000087c    ldr     r1,0x700002d4 ; = #0x4000001c
0x70000880    str     r0,[r1,#0]
396           /* wait until an interrupt has occurred */
397           EnterPauseMode();
0x70000884    bl      EnterPauseMode
398           ClCount         = ClCount - (ClVal & 0xFFFF);
0x70000888    ldr     r0,0x70000a88 ; = #0x40000024
0x7000088c    ldr     r0,[r0,#0]
0x70000890    mov     r0,r0,lsl #16
0x70000894    mov     r0,r0,lsr #16
0x70000898    ldr     r1,0x700002d4 ; = #0x4000001c
0x7000089c    ldr     r1,[r1,#0]
0x700008a0    sub     r0,r1,r0
0x700008a4    ldr     r1,0x700002d4 ; = #0x4000001c
0x700008a8    str     r0,[r1,#0]
399           if ( (ClCount > 0x700) || (ClCount < 0x300)) {
0x700008ac    ldr     r0,0x700002d4 ; = #0x4000001c
0x700008b0    ldr     r0,[r0,#0]
0x700008b4    cmp     r0,#0x700
0x700008b8    bgt     0x700008cc  ; (ct_tests
0x700008bc    ldr     r0,0x700002d4 ; = #0x400
0x700008c0    ldr     r0,[r0,#0]
0x700008c4    cmp     r0,#0x300
```

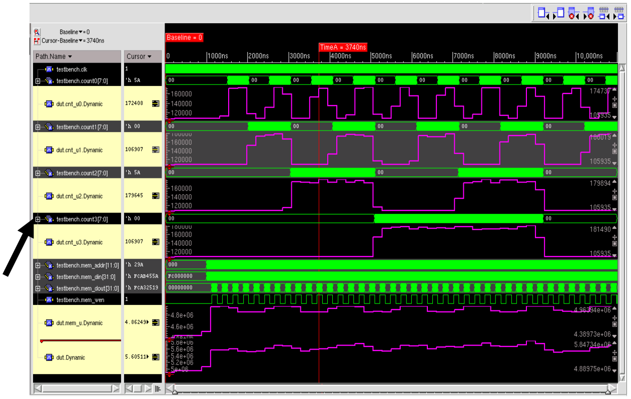
```
#test1:track_init line 124
#test1:main line 299
#rtstand line 272
```

```
r0   0x00000000
r1   0x4000002c
r2   0x0000000a
r3   0x40000164
r4   0x00000000
r5   0x00000000
r6   0x00000000
r7   0x00000000
r8   0x00000000
r9   0x00000000
r10  0x40007200
r11  0x00000000
r12  0x00000000
r13  0x40007fe8
r14  0x700002e8
pc   0x70000308
```

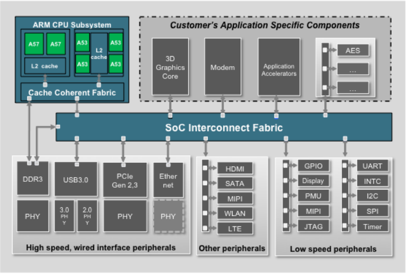
Breakpoint set. /space/jason/arm/adu/solaris/bin/ipcrdi.dll Default

Simulator
HDL Debugger

DPA option
Power Profiler



Path.Name	Cursor
testbench.clk	1
testbench.count0[7:0]	'h 5A
dut.cnt_u0.Dynamic	172408
testbench.count1[7:0]	'h 00
dut.cnt_u1.Dynamic	106907
testbench.count2[7:0]	'h 5A
dut.cnt_u2.Dynamic	179645
testbench.count3[7:0]	'h 00
dut.cnt_u3.Dynamic	106907
testbench.mem_addr[11:0]	'h 29A
testbench.mem_din[31:0]	'h FCAB455A
testbench.mem_dout[31:0]	'h FCA32519
testbench.mem_wen	1
dut.mem_u.Dynamic	4.86249
dut.Dynamic	5.60511



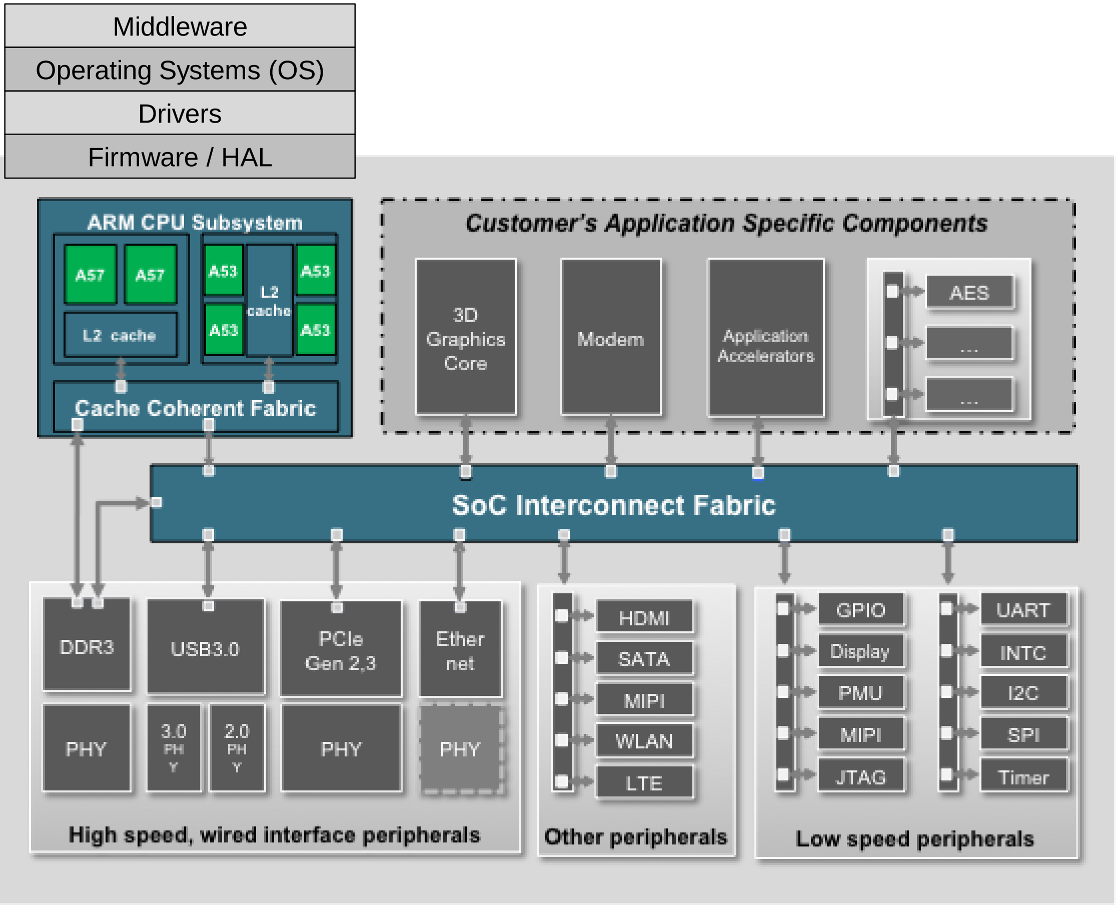
Emulator

High-performance verification platform from RTL acceleration to system emulation

A unique platform tuned for high-performance HW/SW co-verification and power profiling at the system-level perspective

SoC SW & Verification Challenges and Solutions

Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol



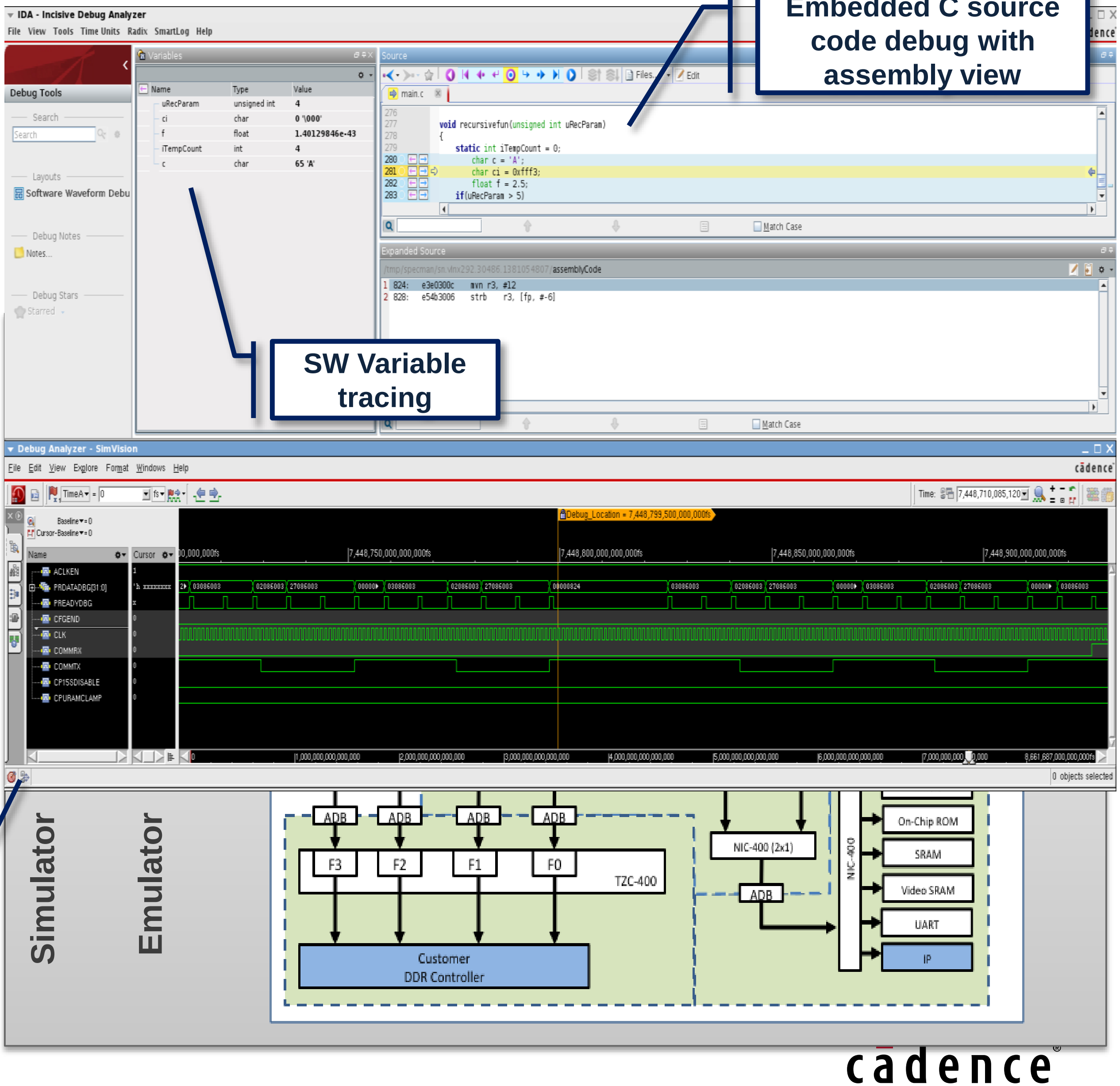
Development flow needs to support HW/SW integration & verification environments on multiple platforms

Planning & Management	Virtual Platform	Formal Tools	Simulators	Emulators	Prototyping Platform	Debug Tools

ARM v8 SoC HW/SW Debug

A53/A57 Post-process SoC Debug

- Integrated & synchronized HW/SW debug with testbench
- For verification & design teams
- Enables off-line debugging



Name	Type	Value
uRecParam	unsigned int	4
ci	char	0 '\000'
f	float	1.40129846e-43
iTempCount	int	4
c	char	65 'A'

```
void recursivefun(unsigned int uRecParam)
{
    static int iTempCount = 0;
        char c = 'A';
        char ci = 0xfff3;
        float f = 2.5;
    if(uRecParam > 5)
```

/tmp/specman/sn.vlnx292.30486.1381054807/assemblyCode

```
824:   e3e0300c    mvn r3, #12
828:   e54b3006    strb    r3, [fp, #-6]
```

Embedded C source code debug with assembly view

SW Variable tracing

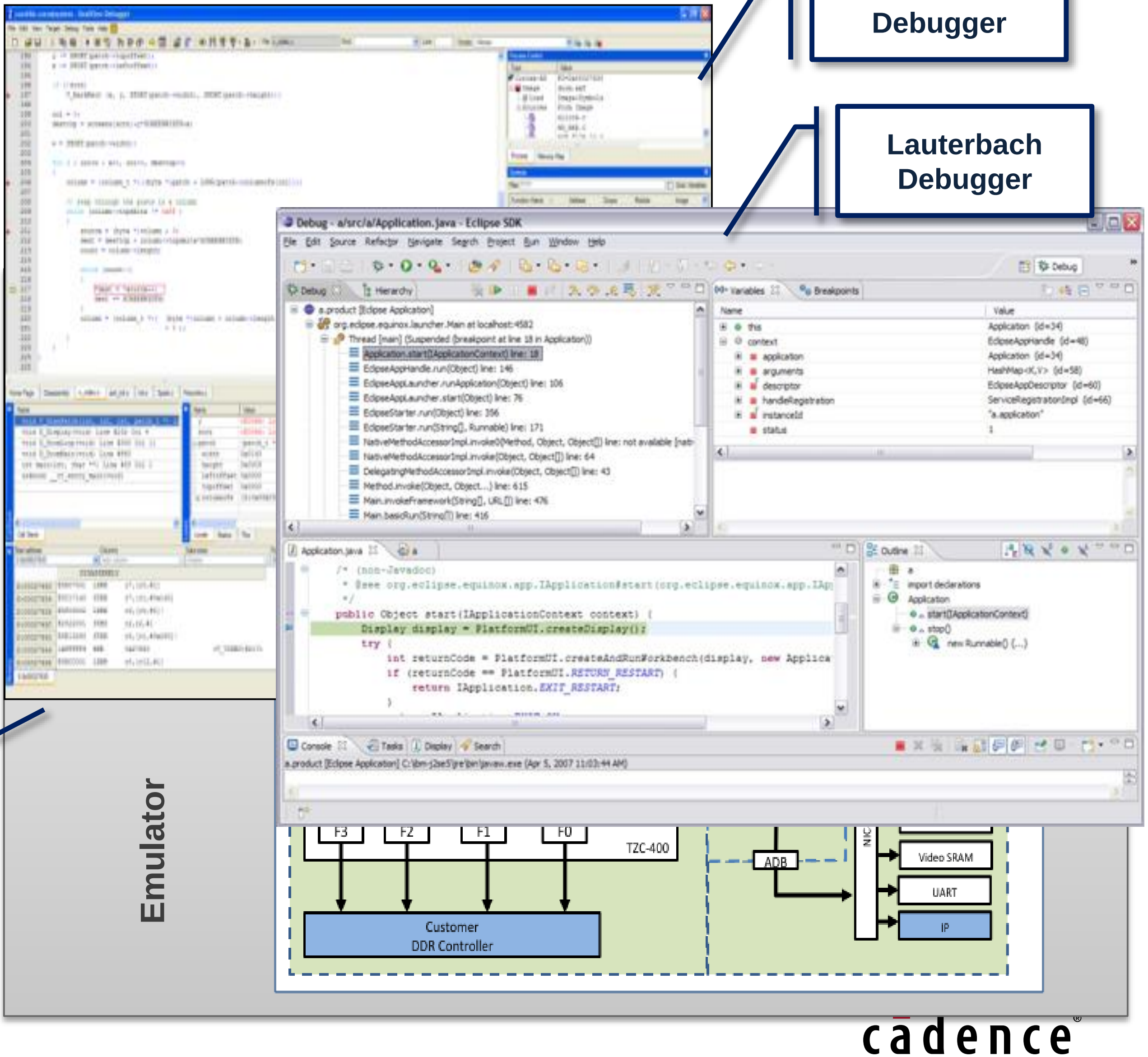
Synchronized with design & testbench debugger

cadence

ARM v8 SoC HW/SW Debug

A53/A57 JTAG SW Debugger

- Interactive SW debugging on Emulator
- Support for SW developers using RealView, Lauterbach, etc..



Debug - a/src/a/Application.java - Eclipse SDK

```
a.product [Eclipse Application]
  org.eclipse.equinox.launcher.Main at localhost:4582
    Thread [main] (Suspended (breakpoint at line 18 in Application))
      Application.start(IApplicationContext) line: 18
      EclipseAppHandle.run(Object) line: 146
      EclipseAppLauncher.runApplication(Object) line: 106
      EclipseAppLauncher.start(Object) line: 76
      EclipseStarter.run(Object) line: 356
      EclipseStarter.run(String[], Runnable) line: 171
      NativeMethodAccessorImpl.invoke0(Method, Object, Object[]) line: not available [nat
      NativeMethodAccessorImpl.invoke(Object, Object[]) line: 64
      DelegatingMethodAccessorImpl.invoke(Object, Object[]) line: 43
      Method.invoke(Object, Object...) line: 615
      Main.invokeFramework(String[], URL[]) line: 476
      Main.basicRun(String[]) line: 416
```

Name	Value
this	Application (id=34)
context	EclipseAppHandle (id=48)
application	Application (id=34)
arguments	HashMap<K,V> (id=58)
descriptor	EclipseAppDescriptor (id=60)
handleRegistration	ServiceRegistrationImpl (id=66)
instanceId	"a.application"
status	1

```
    /* (non-Javadoc)
     * @see org.eclipse.equinox.app.IApplication#start(org.eclipse.equinox.app.IAp
     */
    public Object start(IApplicationContext context) {
        Display display = PlatformUI.createDisplay();
        try {
            int returnCode = PlatformUI.createAndRunWorkbench(display, new Applica
            if (returnCode == PlatformUI.RETURN_RESTART) {
                return IApplication.EXIT_RESTART;
            }
```

```
a.product [Eclipse Application] C:\ibm-j2se5\jre\bin\javaw.exe (Apr 5, 2007 11:03:44 AM)
```

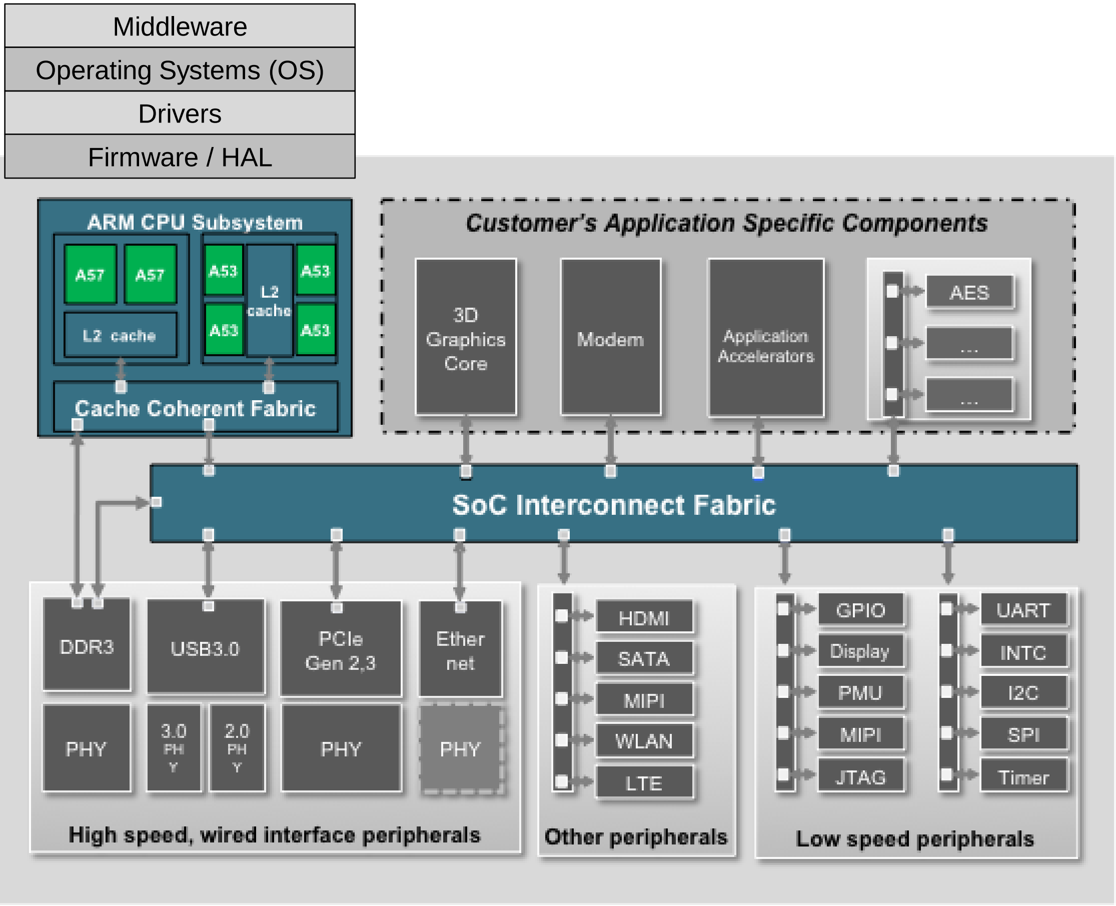
ARM RealView Debugger

Lauterbach Debugger

JTAG Debugger support for SW Developers on Emulator

SoC SW & Verification Challenges and Solutions

Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol



Development flow needs to support HW/SW integration & verification environments on multiple platforms

Planning & Management

Virtual Platform

Formal Tools

Simulators

Emulators

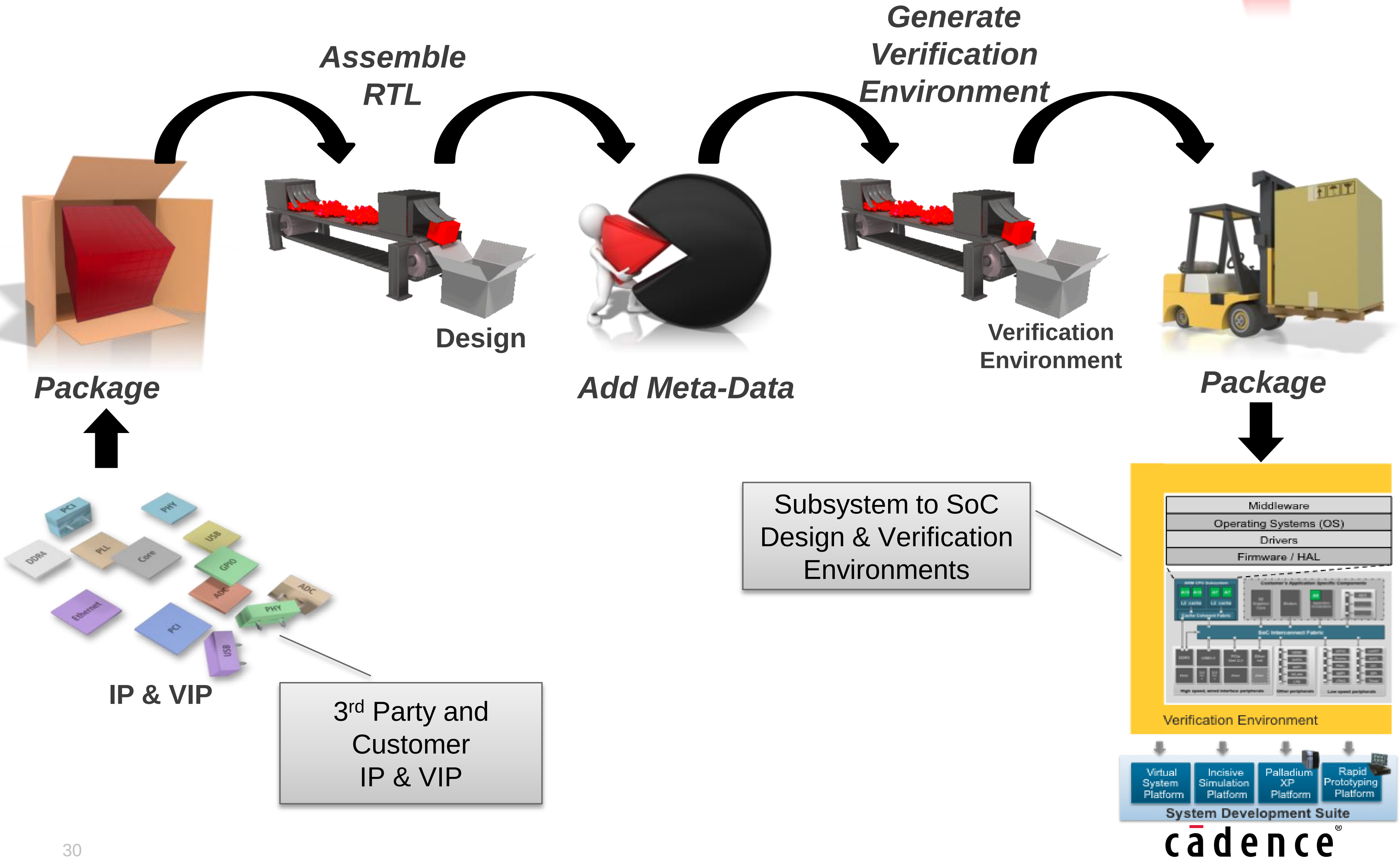
Prototyping Platform

Debug Tools

Challenges in ARM SoC Integration

- Significant engineering effort required to integrate & verify IPs (Customer IPs + 3rd Party IPs)
  - Each IP has many configuration options
  - IPs must be configured consistently together with SW Drivers & VIPs
  - Largely a manual, error-prone effort
- Need to develop multiple environments on different platforms for verification/validation & SW bring-up/integration
  - Virtual Platform, Simulation, Emulation, FPGA Prototype, Post-Silicon
  - Cumbersome integration work of RTL/SW/VIP repeated per platform
  - Some platforms have special requirements requiring model changes
- Late spec changes and/or derivative platforms are difficult to support within short market windows
  - Shift from ARM v7 to v8 was not planned by many mobile projects

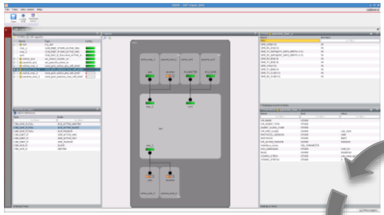
Flow Automation



Subsystem to SoC Design & Verification Environments

3rd Party and Customer IP & VIP

Factory Flow



ARM weaver socrates

IP Configuration & Integration

GUI for capturing SoC design & verification environment with additional meta-data

IPXACT

Rapidly generate SoC DVE configs

TB Integrator

- SW Assembly
- RegApp
- Embedded TB Assembly
- Interconnect WorkBench
- UVM e/SV TB Assembly
- Hybrid TLM/RTL Assembly
- Build & Run Generator

SoC Project Library

IP Packager

VIP Packager

Customer IP

ARM IP

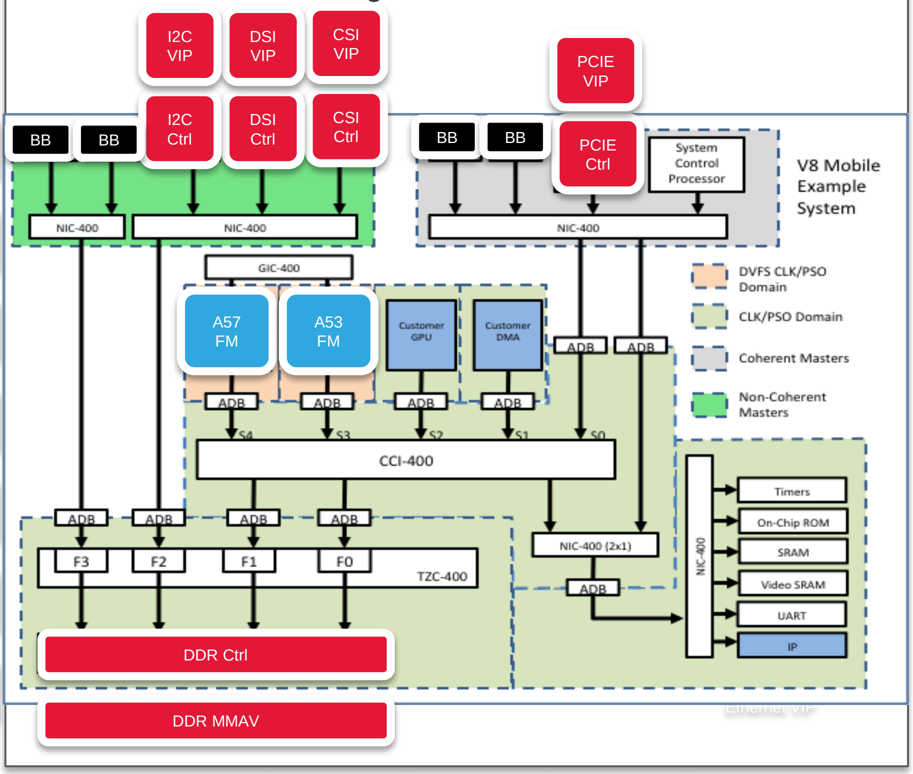
3rd Party IP

VIP

Package IP, VIP, eSW, Meta-data

Generate build & run scripts

ARM SoC Design & Verification Environment



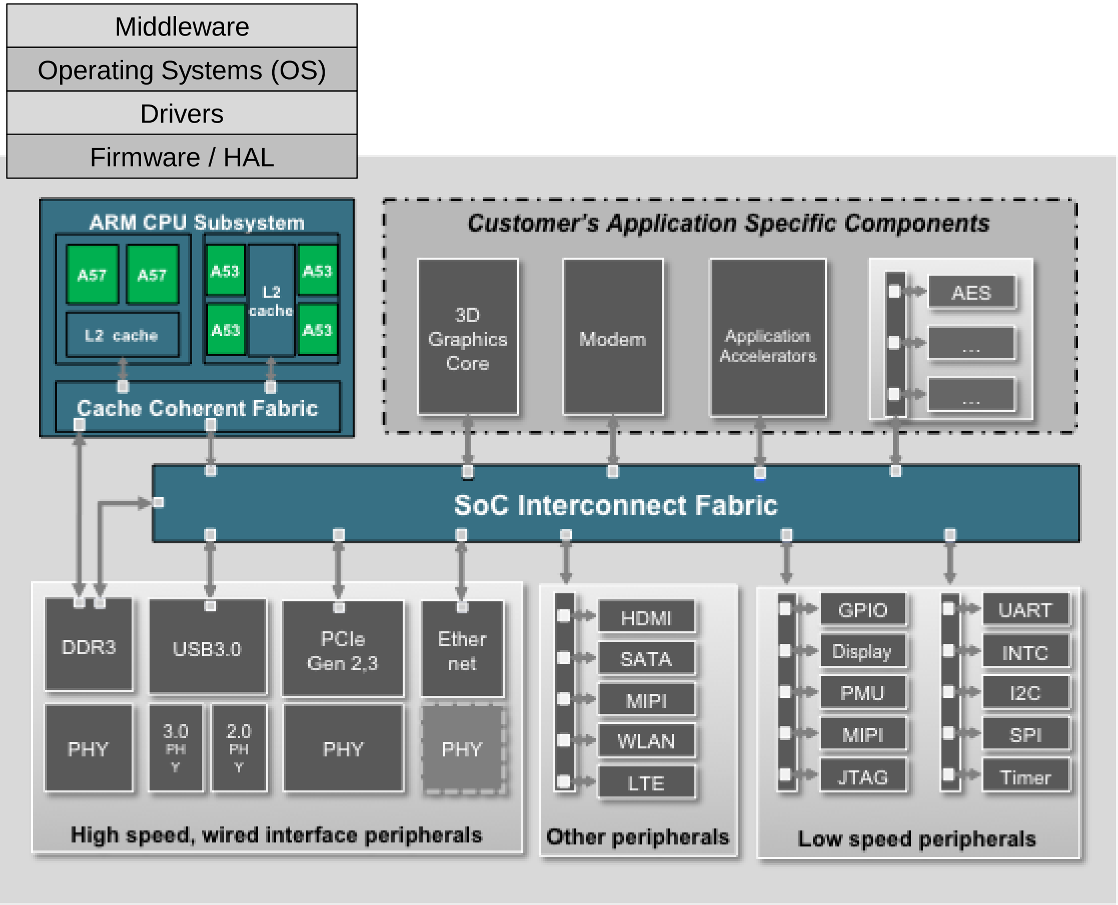
- Virtual Platform
- Simulation Platform
- Emulation Platform
- Prototyping Platform

Advantage for Automation in SoC Integration

- Rapid generation of ARM SoC design & verification environments
  - Pre-packaged ARM FM/RTL IP + CDNS IP/VIP
  - For performance modeling and analysis
  - For SW bring-up and development
  - For HW/SW integration and verification
  - Targeting multiple execution platforms, including hybrid platforms
- Reduce errors in manual integration based on paper specs
  - Single source, machine-readable specification data
- Improve turn-around time for late spec changes
  - By automating integration tasks
- Improve time to market for SoC derivatives
  - Quickly generate and verify incremental changes to SoC platform

SoC SW & Verification Challenges and Solutions

Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol



Development flow needs to support HW/SW integration & verification environments on multiple platforms

Planning & Management

Virtual Platform

Formal Tools

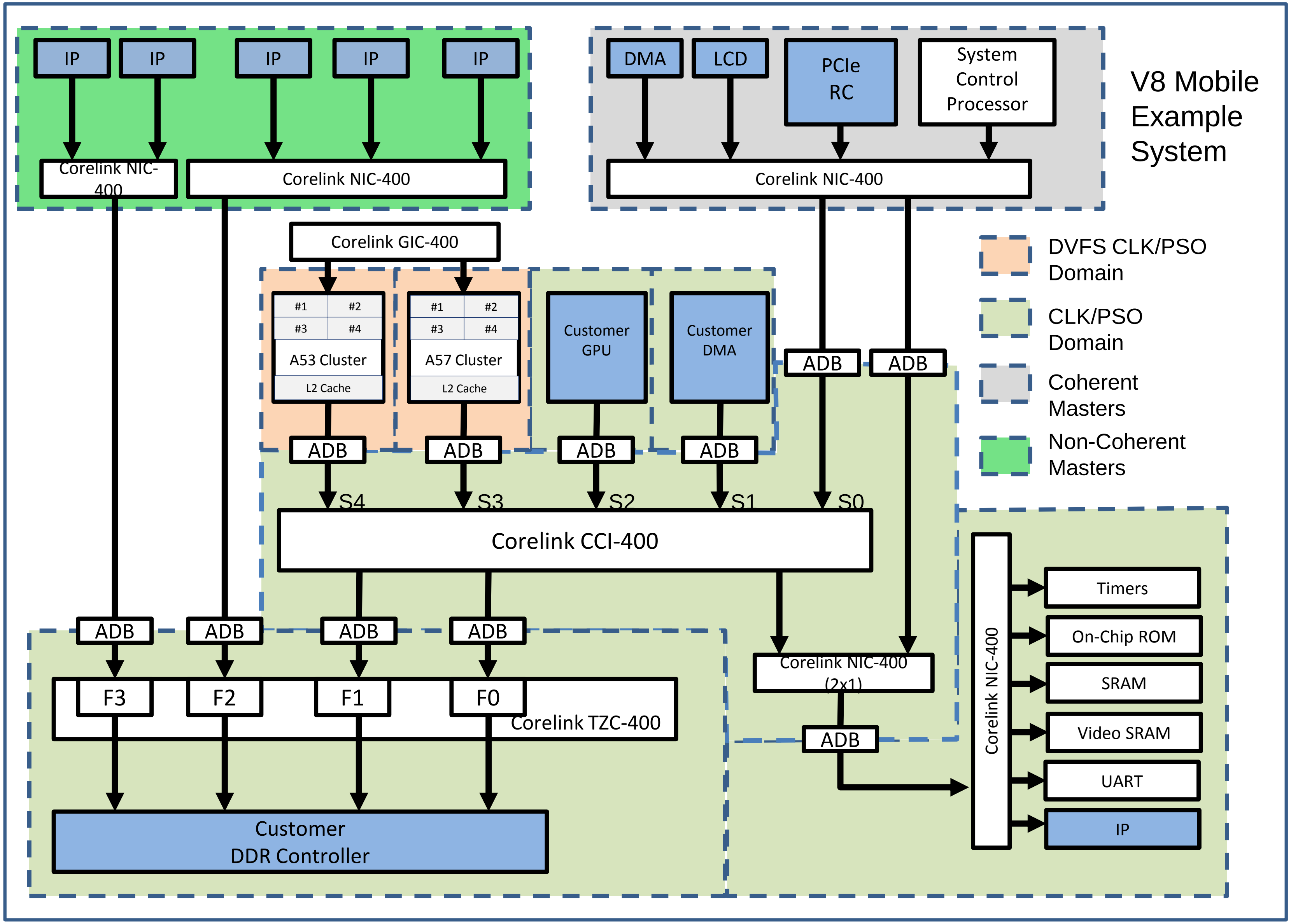
Simulators

Emulators

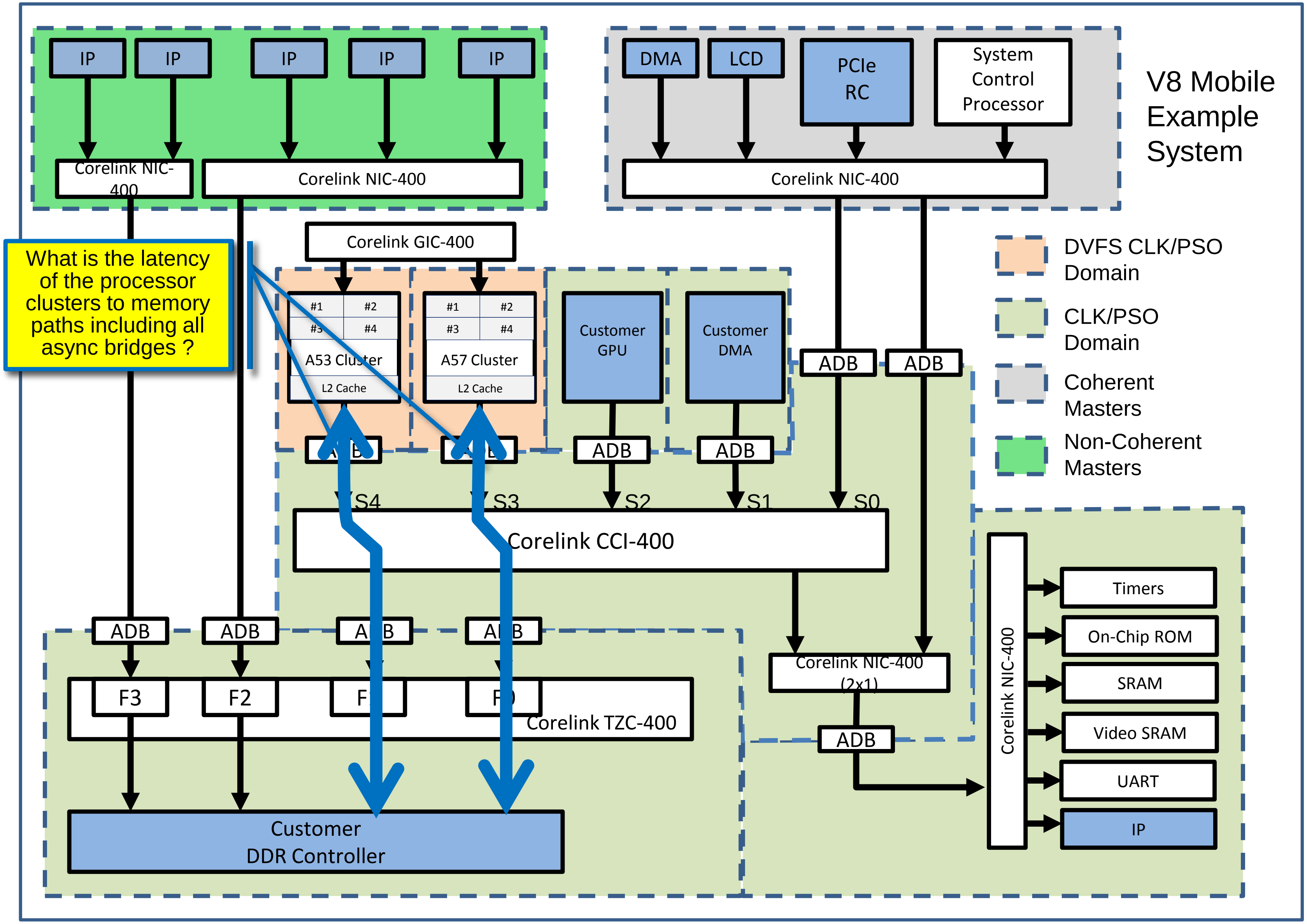
Prototyping Platform

Debug Tools

Representative ARM SoC



Performance Challenges



V8 Mobile Example System

What is the latency of the processor clusters to memory paths including all async bridges ?

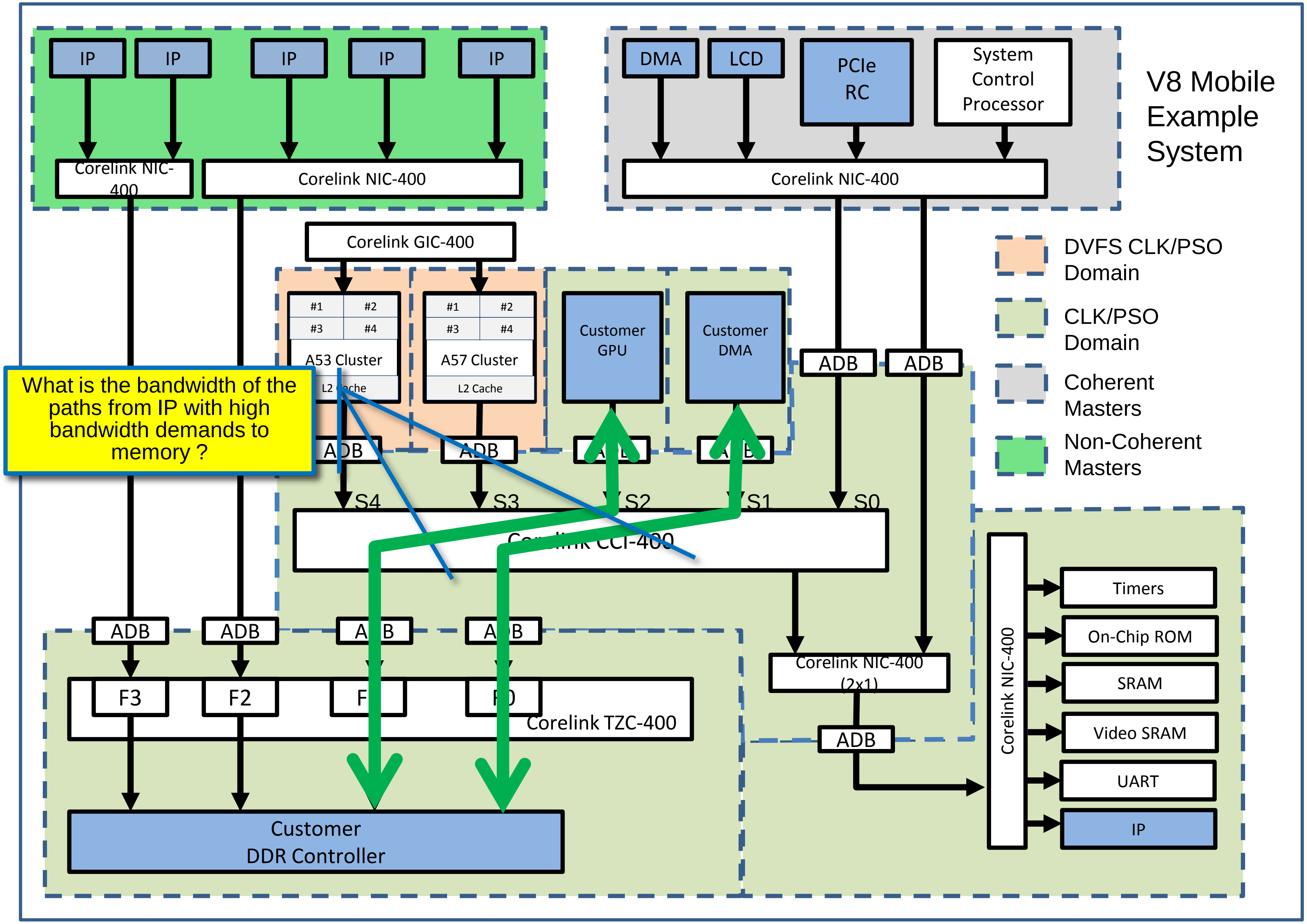
🟧 DVFS CLK/PSO Domain

🟩 CLK/PSO Domain

⬜ Coherent Masters

🟩 Non-Coherent Masters

Performance Challenges

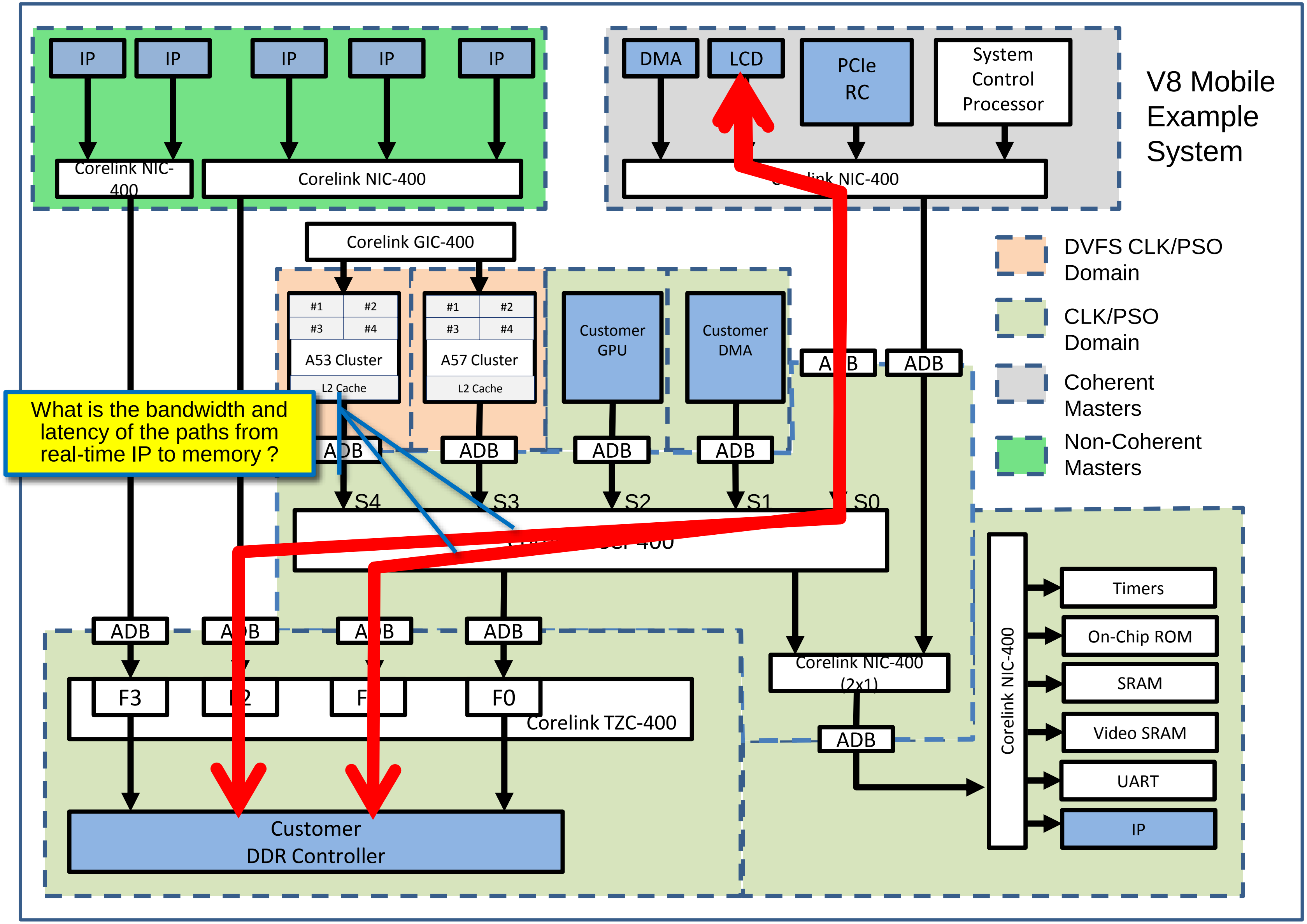


What is the bandwidth of the paths from IP with high bandwidth demands to memory ?

V8 Mobile Example System

▨	DVFS CLK/PSO Domain
▨	CLK/PSO Domain
▨	Coherent Masters
🟩	Non-Coherent Masters

Performance Challenges



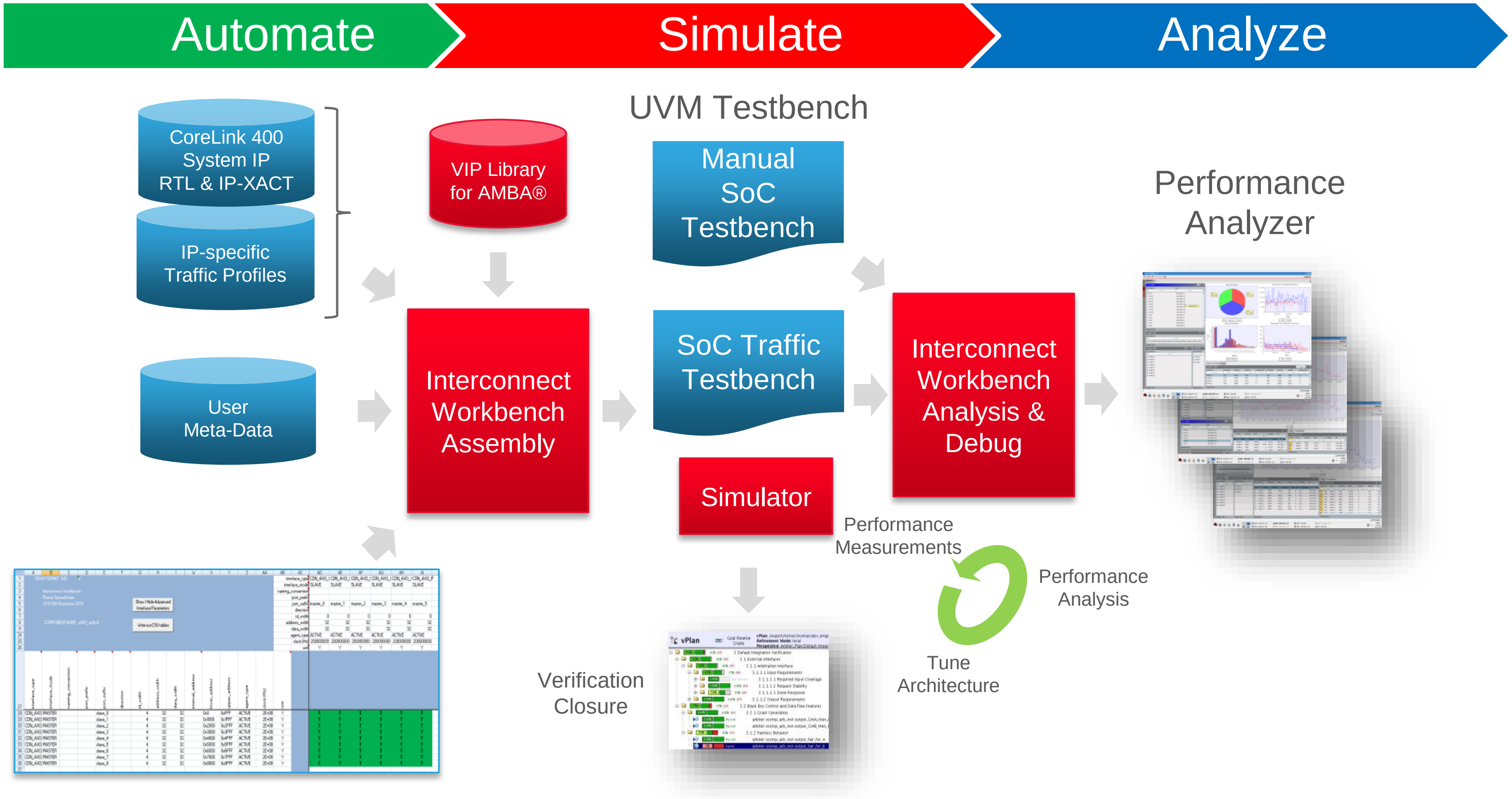
V8 Mobile Example System

What is the bandwidth and latency of the paths from real-time IP to memory ?

Performance Infrastructure Challenges

- Create and maintain Performance Testbench
- No standard Performance Monitors which are protocol agnostics
- Analysis of the Performance Data

Solution



Automate

Simulate

Analyze

CoreLink 400
System IP
RTL & IP-XACT

IP-specific
Traffic Profiles

User
Meta-Data

VIP Library
for AMBA®

Interconnect
Workbench
Assembly

UVM Testbench

Manual
SoC
Testbench

SoC Traffic
Testbench

Simulator

Verification
Closure

Interconnect
Workbench
Analysis &
Debug

Performance
Measurements

Performance
Analysis

Tune
Architecture

Performance
Analyzer

For Interconnect IP Integration

- Performance of use case traffic loads
- Verify configuration functionality

For SoC Integration

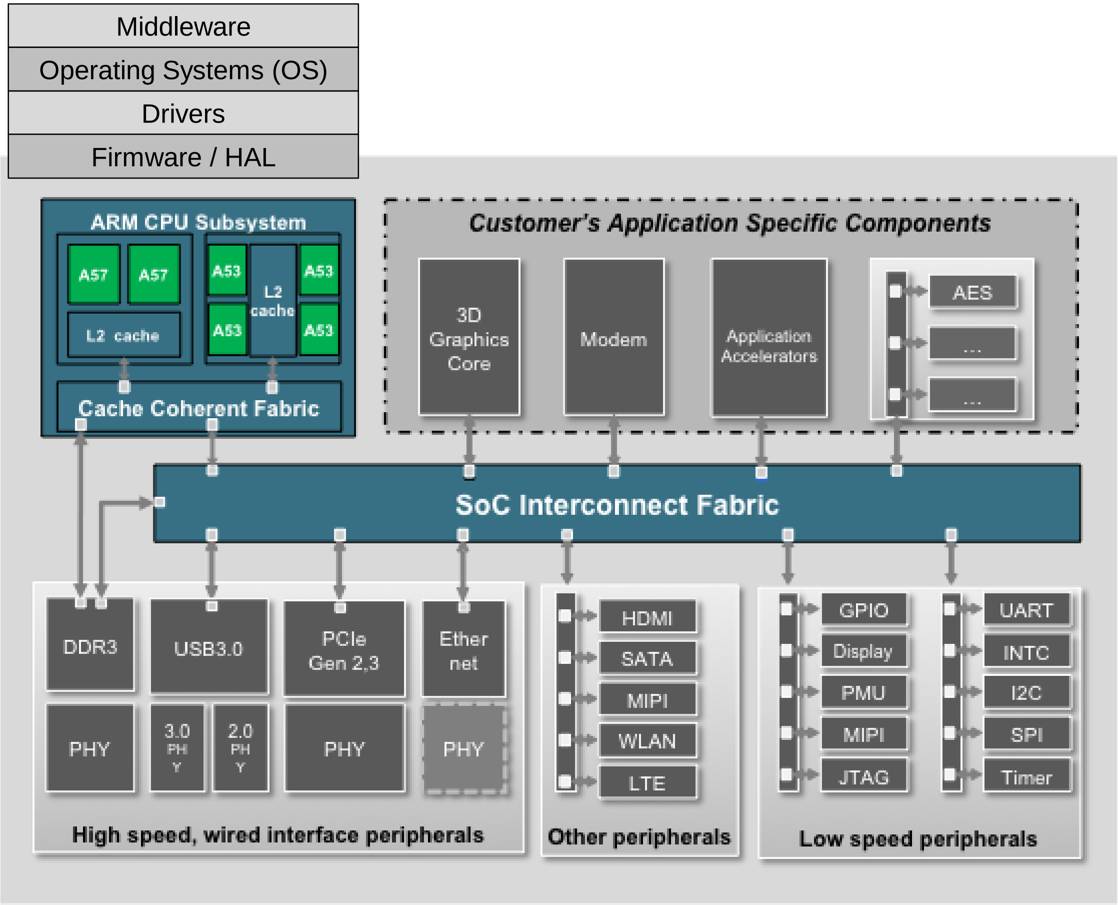
- Validate performance in context of IPs

Benefits

- Shorten performance tuning and analysis iteration loop from **days** to **hours**
- Reduce testbench development time from **weeks** to **hours**

SoC SW & Verification Challenges and Solutions

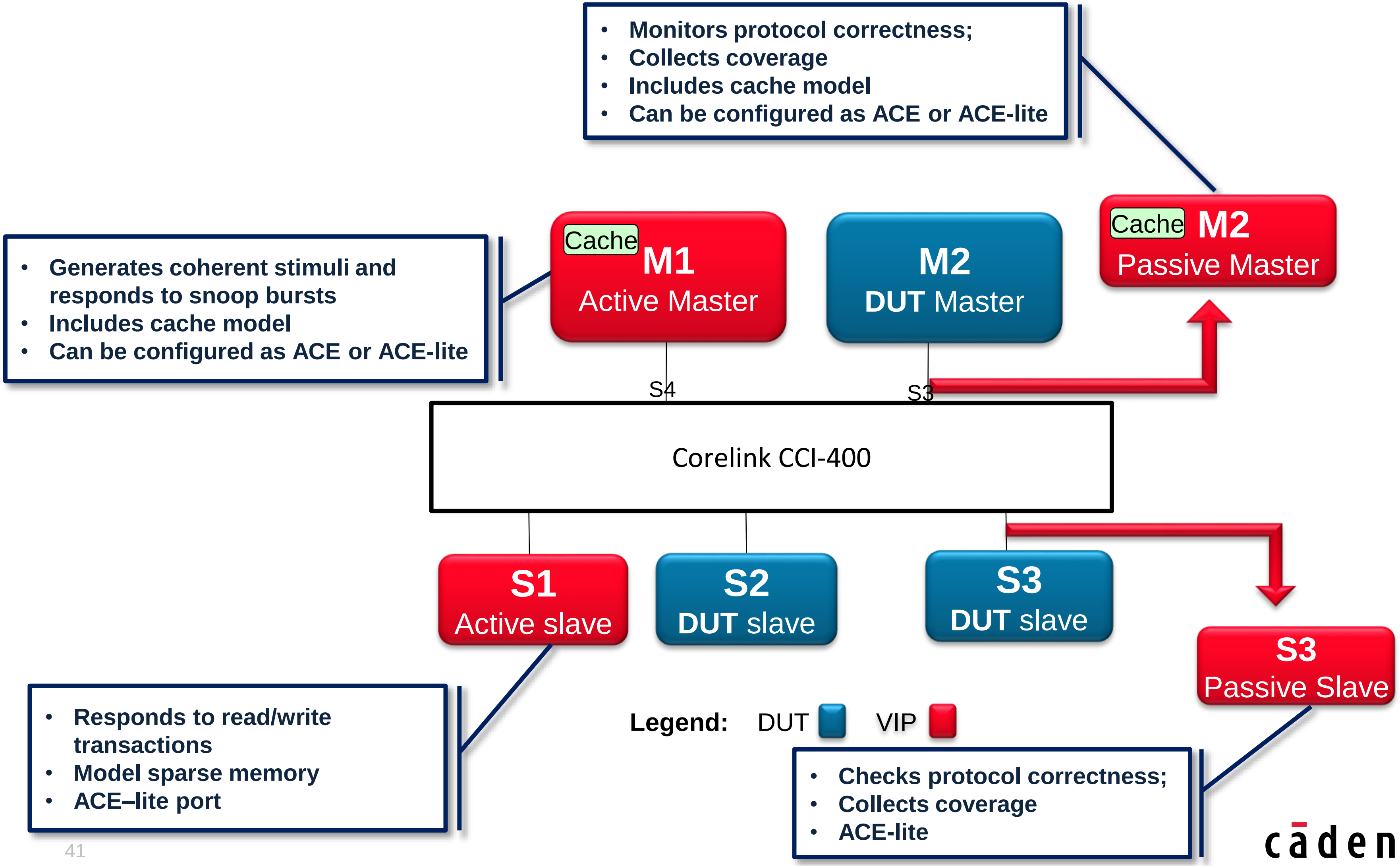
Hybrid for early OS & SW bring-up	Multi-core early SW bring-up and integration
SW-Driven use case verification	Processor based SW use case testing to verify multi-core cache & IO coherency, concurrency, PSO, etc…
Profiling real traffic	Power & thermal profiling and analysis of real world traffic
Synchronized embedded SW debug on ARM RTL CPU	Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms
Rapid SoC integration & verification	Integrating & verifying 100's of IPs & SW from ARM, internal teams, & 3rd parties
SoC verification and performance analysis	Characterizing & analyzing SoC performance & efficiently debugging issues
VIP	Verification of IPs on AMBA interconnect w/ adherence to cache coherent (ACE) protocol



Development flow needs to support HW/SW integration & verification environments on multiple platforms

- Planning & Management
- Virtual Platform
- Formal Tools
- Simulators
- Emulators
- Prototyping Platform
- Debug Tools

Cache Coherent ACE Verification IP



- Monitors protocol correctness;
- Collects coverage
- Includes cache model
- Can be configured as ACE or ACE-lite

- Generates coherent stimuli and responds to snoop bursts
- Includes cache model
- Can be configured as ACE or ACE-lite

Cache **M1** Active Master

M2 **DUT** Master

Cache **M2** Passive Master

S4

S3

Corelink CCI-400

S1 Active slave

S2 **DUT** slave

S3 **DUT** slave

S3 Passive Slave

- Responds to read/write transactions
- Model sparse memory
- ACE–lite port

Legend: DUT 🟦 VIP 🟥

- Checks protocol correctness;
- Collects coverage
- ACE-lite

Summary

- Major challenges and solutions

  - Early SW Bring-up & HW/SW Integration
    - Hybrid for early OS & SW bring-up
  - SW-driven use case verification
    - Abstract level test creations for horizontal and vertical reuse is the key
  - Power profiling and analysis of real world traffic
    - Dynamic power analysis by capturing hardware activities from use-cases/software runs on emulator and feed it to Synthesizer
  - HW/SW debug
    - Debgger with HW/SW views
    - JTAG SW Debugger support for Software
  - Rapid IP Integration
    - Automation is the key
  - SoC performance characterization, analysis, & verification
    - Automation in testbench creation
    - Analysis tools
  - Advance protocol checking
    - Advance VIPs for ACE/CHI are useful for protocol compliance

cadence®