Exploring Next Generation SoC Architectures with Virtual Platforms and RISC-V

S. Davidmann, L. Moore, L. Lapides

Imperas Software Ltd.

Exploring Next Generation SoC Architectures with Virtual Platforms and RISC-V

S. Davidmann, L. Moore, L. Lapides

Imperas Software Ltd.

Imperas Focus

- "nobody designs a chip without simulation",
  at Imperas we believe that:

  "nobody should develop embedded software without simulation"

- Imperas develops simulators, tools and debuggers, and models (especially processor models) to help embedded systems developers get their software running…
  - and hardware developers get their designs correct

- 10+ years, self funded, profitable, UK based, team with much EDA (simulators, verification), processors, and embedded experience
- www.imperas.com
- www.OVPworld.org

RISC-V Impact on SoC Designs

- Architecture analysis, including (especially) custom instructions
- Software development, debug and test
- Processor and SoC verification

RISC-V Impact on SoC Designs

- **Architecture analysis, including (especially) custom instructions**
- Software development, debug and test
- Processor and SoC verification

Amdahl's Law

- A guideline for multi-core efficiency
  - IBM computer architect & entrepreneur
    - Left IBM when his ideas were rejected
    - Founded Amdahl computers:
      - Cheaper, faster, more reliable
      - IBM plug-compatible…
  - Amdahl's law (1967) is used in parallel computing to predict the theoretical speedup when using multiple processors

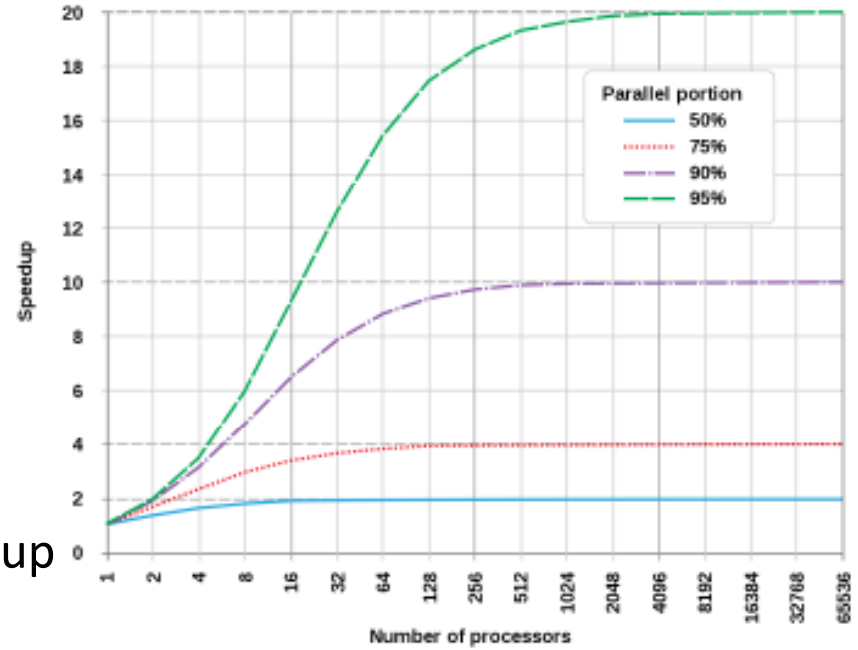
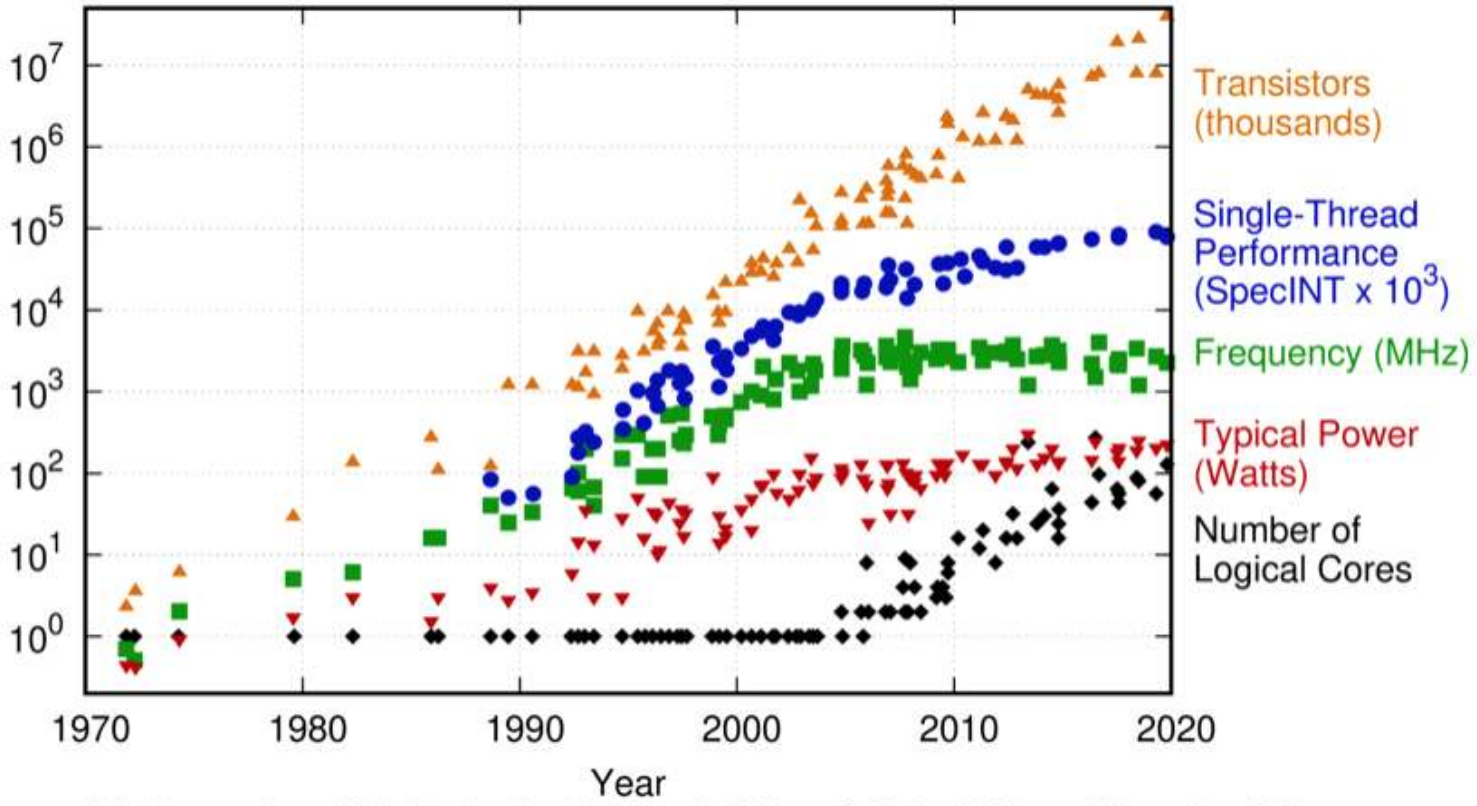


Image Source: Daniels220 at English Wikipedia

$$S_{latency}(s) = \frac{1}{(1 - p) + \frac{p}{s}}$$

- $S_{latency}$ is the theoretical speedup of the execution of the whole task;
- s is the speedup of the part of the task that benefits from improved system resources;
- p is the portion of execution time that the part benefiting from improved resources originally occupied.

48 Years of Microprocessor Trend data



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2019 by K. Rupp

Computation needed for ML/AI

- e.g. 1 Billion MACs for AlexNet – image recognition… training
- x86 is not getting faster
- So we go have to have special processing and run in parallel
- And that's where Amdahl's law comes in…
  - Performance is hindered by the bottleneck(s) – the serial pinch points…

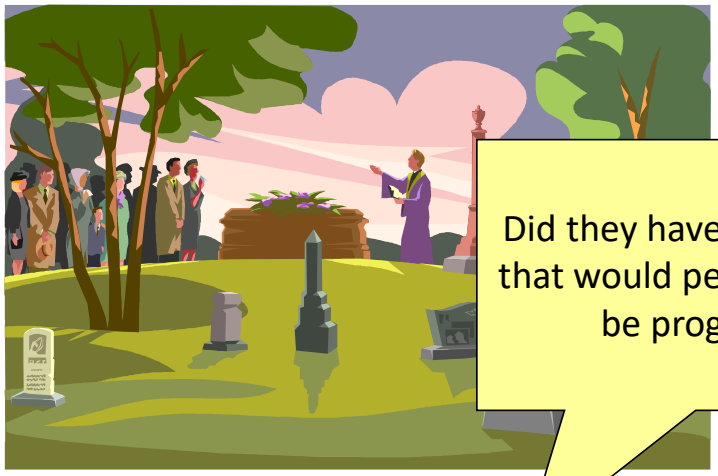
=>

- So it needs to be the correct parallel…
- Designers need to know that their algorithms run "well" on the configuration of hardware they select

Many different approaches to parallelism

Many failed
(a 2007 Imperas slide…)

imperas



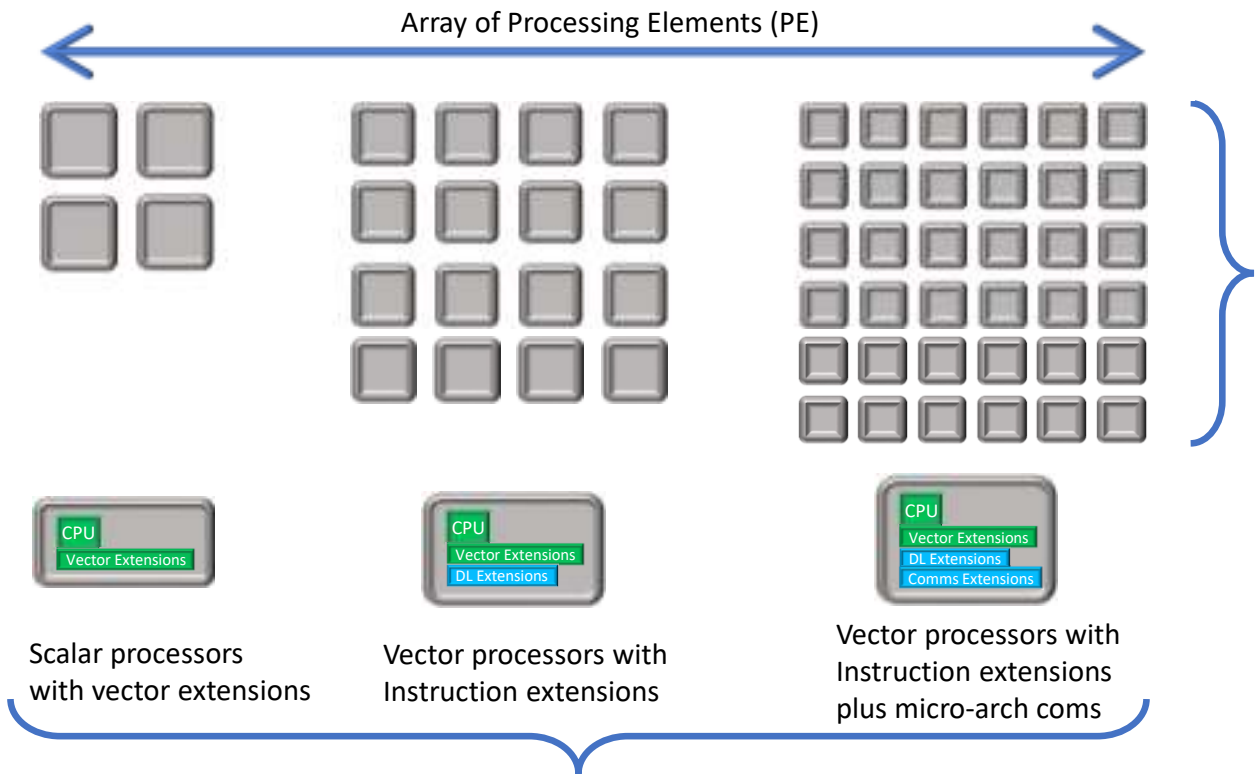
Did they have an architecture that would perform and could be programmed?

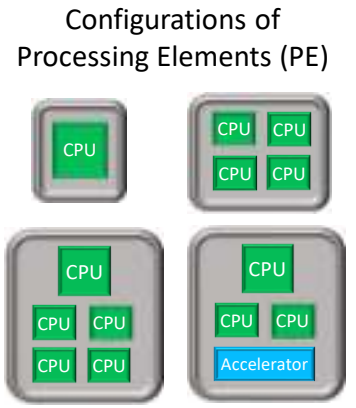
Quicksilver ? Equator ?
Chameleon ? PACT ?
Morphics ? Systolix ?
Chromatic Research ? Intrinsity ?
Triscend ? Adelante ?
BOPS ? ….?

But now there is established SW

- That has many requirements on hardware platforms to run efficiently

AI SoC Architecture Exploration





AI & Machine Learning Accelerators
- Datacenter: training & inference
- Edge: inference (mostly)
- Compute arrays with processor elements (PE) configured for
  - Scalar
  - Vector
  - Spatial
  - Communications
    - PE <–> PE & PE <-> NoC

How we help

Imperas

- We model the processors (250+ in library)
  - You can create your own, with own ISA, or you can add your own extensions / smarts to ours
- We have a library of the behavioural components (300+ in library)
  - You can add your own, or configure ours
- Our technology allows you to build/model the hierarchical platform (50+ in library)
  - Your configuration of processors, behavioral components and hierarchies
- Our simulators can simulate from core to system
  - We have industry leading simulation performance, 300MIPS -2 Billion instructions sec
  - Single core, multi-core, AMP, SMP
  - From bare metal to SMP Linux boot in under 10 secs

Imperas

- We model the processors (250+ in library)
  - You can create your own, with own ISA, or you can add your own extensions / smarts to ours
- We have a library of the behavioural components (300+ in library)
  - You can add your own, or configure ours
- Our technology allows you to build/model the hierarchical platform (50+ in library)
  - Your configuration of processors, behavioral components and hierarchies
- Our simulators can simulate from core to system
  - We have industry leading simulation performance, 300MIPS -2 Billion instructions sec
  - Single core, multi-core, AMP, SMP
  - From bare metal to SMP Linux boot in under 10 secs
- Full holistic multi-core platform debug (Eclipse based)
- Advanced tools for analysis and profiling
- Full capability for hardware verification
  - Works with SystemVerilog UVM simulators from Cadence, Mentor, Synopsys, Metrics

Open Virtual Platforms (OVP) Library of High-Performance Processor Models

- Over 250+ Fast Processor Models in OVP Library

- ARM®: Models for ARMv4™, v5™, v6™, v7™ and v8™ architectures
  - Including MMU, MPU, TCM, Thumb™, Thumb-2™, Jazelle™, SIMD, VFPv3, NEON™, TrustZone®, SVE, hardware virtualization instructions, …

- MIPS®: Models for microMIPS, nanoMIPS, MIPS32 and MIPS64 architectures
  - Verification, licensing, and distribution relationship, and free MIPSOpenOVPsim available from MIPS Open
  - Including MMU, MPU, DSP, FPU, MT, MSA, VZ architecture subsets

- Synopsys (ARC): ARC6xx, ARC7xx, EM families

- RISC-V: Andes, SiFive, Microsemi, OpenHW, lowRISC (pulp) + all 26 standard 32/64bit variants + vectors/bitmanip/crypto/dsp
  - Free riscvOVPsim/Plus ISS model used by RISC-V International Technical Committee task group for architectural tests (compliance)

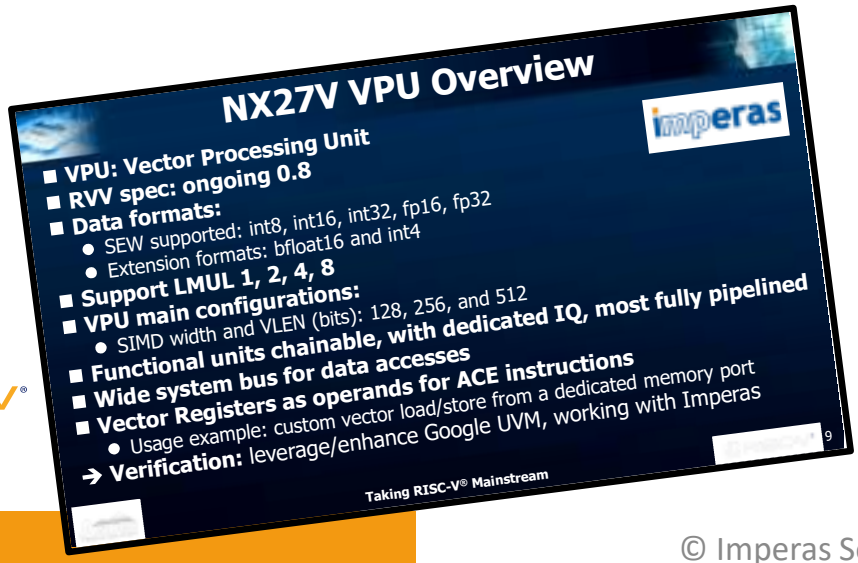
- Renesas: Models for RH850, V850 architectures; 16 bit microcontroller cores
  - RH850G3, V850 ES, E1, E1F, E2; RL78, M16C cores

- PowerPC
- Altera Nios II
- Xilinx Microblaze

"OVP is addressing key issues in software development for embedded systems. By supporting the creation of virtual platforms, OVP is enabling early software development and helping expand the ARM user community."
Noel Hurley, VP Business Development, ARM

Imperas works with the leaders for RISC-V Vector Extensions

- Andes certifies Imperas models and simulator as reference for new Andes RISC-V Vectors Core with lead customers and partners
  - Imperas code morphing simulation technology, virtual platforms and tools used by lead customers for early software development and high-level architectural exploration



"Andes has announced the new RISC-V family 27-series cores, which in addition to new and advanced features, include the new Vector extensions that are an ideal solution for our customers working on leading edge design for AI and ML. Andes is pleased to certify the Imperas model and simulator as a reference for the new Vector processor NX27V, and is already actively used by our mutual customers."

Charlie Hong-Men Su, CTO and Executive Vice President at Andes Technology Corp

RISC-V Summit 2019

Example: RISC-V Vector engine

- Imperas OVP model of RISC-V
  - Full specification, all user, privilege, vector instructions and functionality
  - Single core, single thread performance 300-2,000MIPS speed
    - 24 cores, single thread performance dhrystones 1,000 MIPS overall (40MIPS per core)
  - Parallel simulation: host core utilization ~2x overall for 4 cores (your mileage may differ)
- Uses very little RAM for simulation and uses sparse memory - so scales well
  - Seen simulations of platforms with up to 1024 cores
- Vector engine configurable for HW options: VLEN, SLEN, ELEN

Related experience

- Customer project
  - Full ML/AI engine
  - 150+cores
    - Many with RISC-V Vector engine
  - Runs simulation in 2hrs @ 500MIPS
    - Cross compiled software running on simulated CPUs
  - Allows software stack development
  - Allows hardware platform config, re-config, architectural changes
    - Explore performance options
    - Runs real software (production binaries) – can see how it interacts with HW config
  - Running in Imperas over a year before RTL commit
    - Customer has SW and is looking to design HW to make it work the way they want…
  - Also a by-product: kick-start SoC process by feeding models into HW DV at start

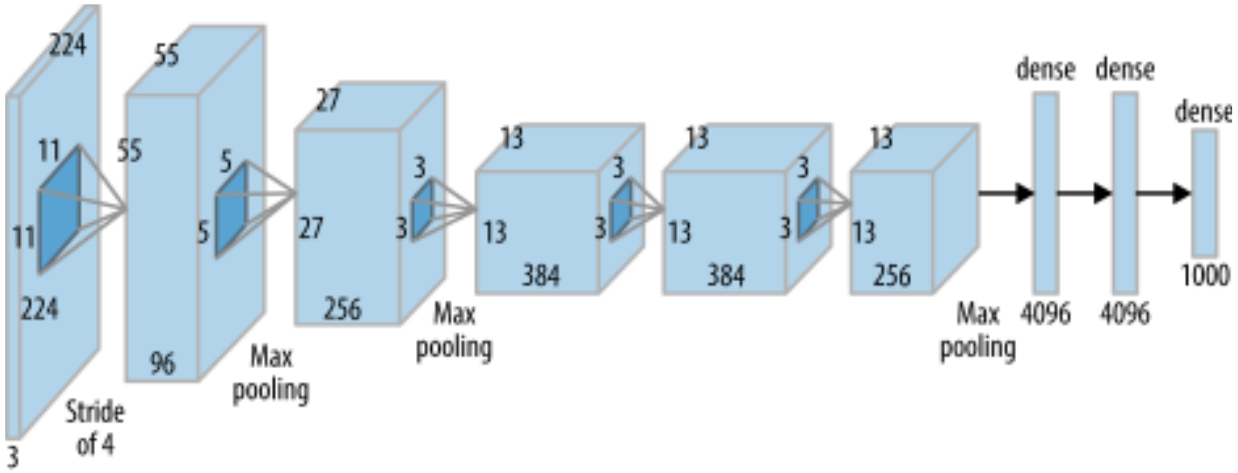
Another Example with Japanese partner

imperas

- Summary
  - Platform : ARM Cortex-A57 x 1 + RISC-V RV64GCV x 17
  - Application1 : AlexNet image recognition deep neural network
- Keypoints
  - “Imperas simulator can simulate heterogeneous virtual platform”
  - “Imperas also provides dedicated debugger which can debug hetero-system (ex. ARM and RISC-V) using one debugger at same time”
  - “Very fast. This example runs (at most) 10 times slower than native x64 execution on host PC”

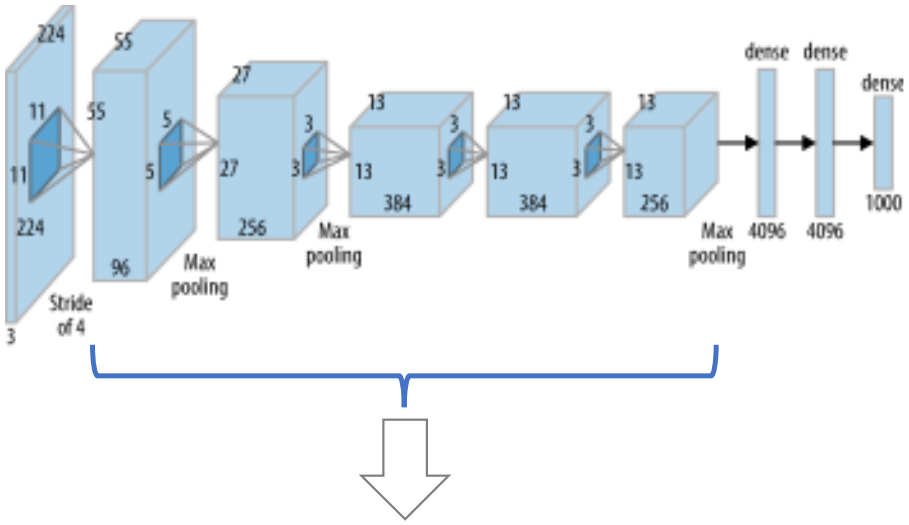
Imagenet with AlexNet deep neural network

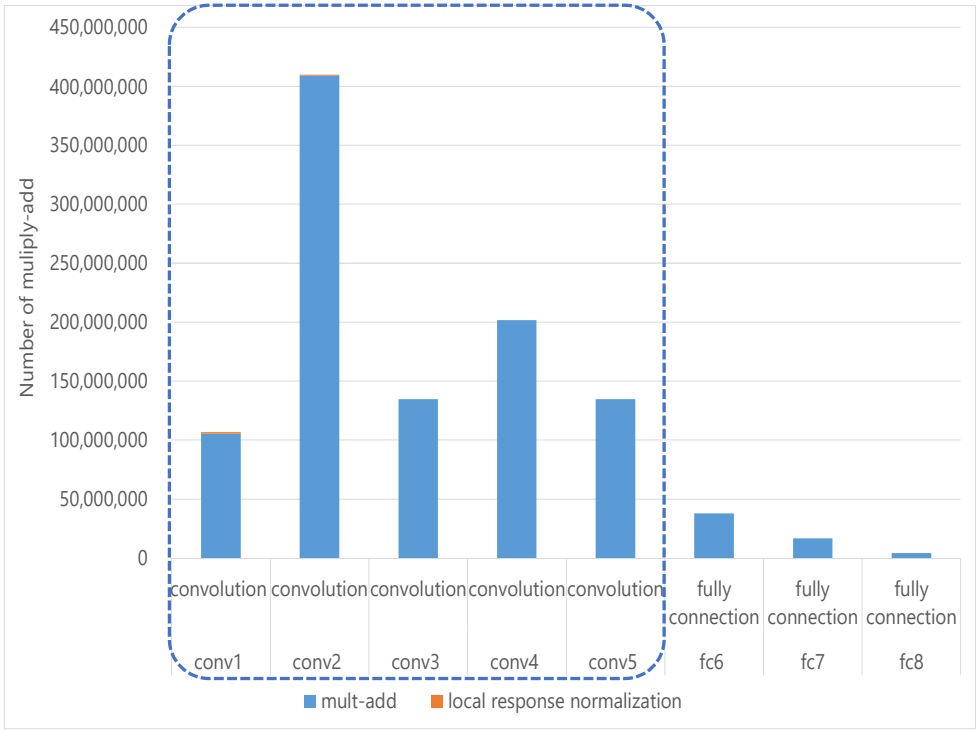
- AlexNet (University of Toronto, 2012)
  - https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96



- Hyper parameters
  - Number of Parameter : 58 M (float32)
  - Computation cost : 1,000 M (Number of multiply-add)

Parallelization for multiple core

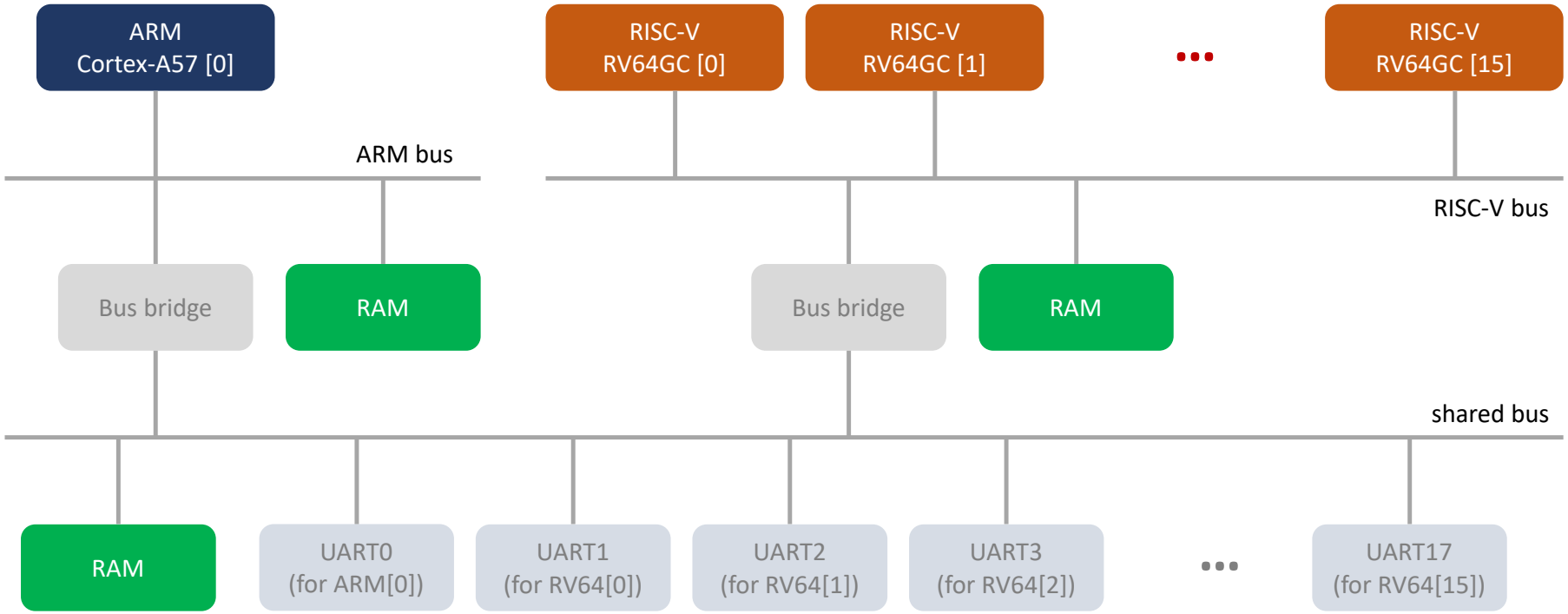




Convolution layers have a lot of calculation

Parallelized these layers to use 16 CPU cores

Platform

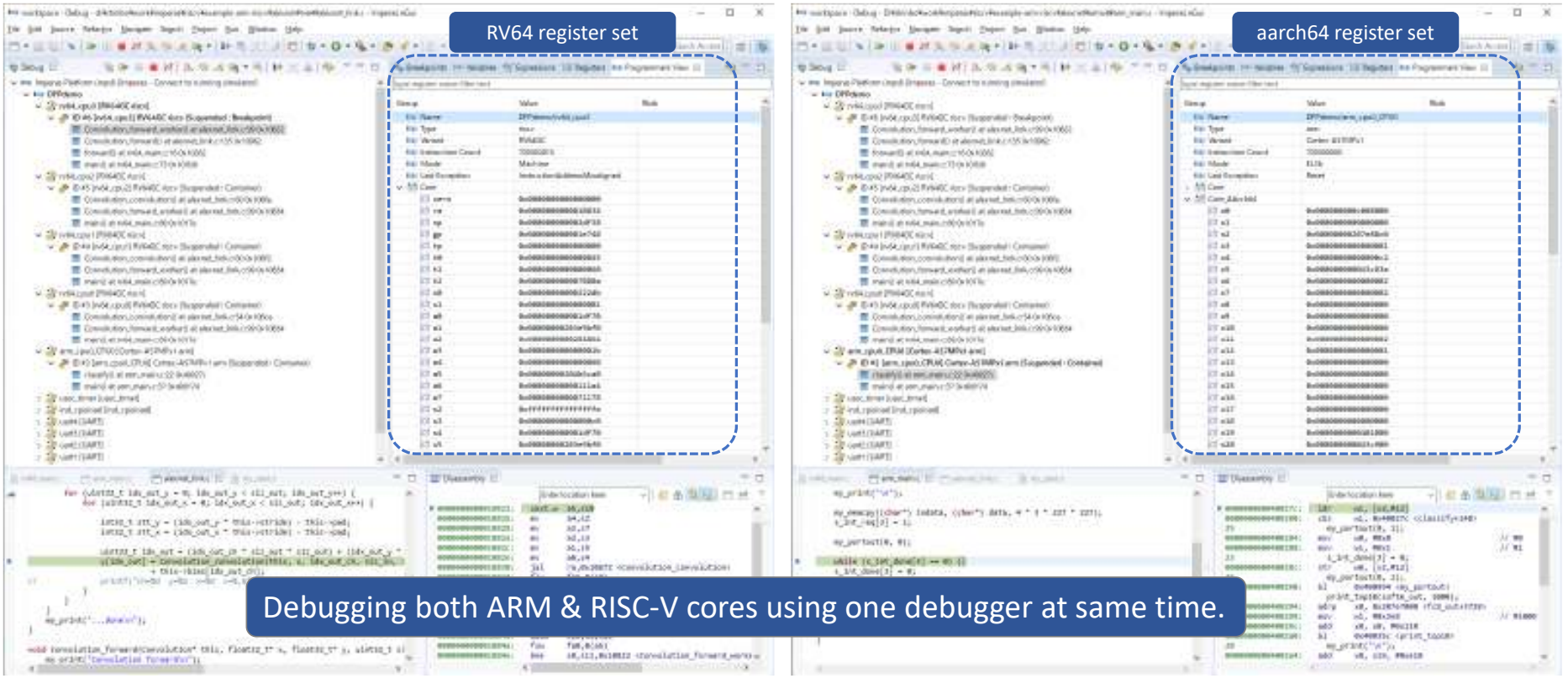


Demo

Demo

Executing simulation – different consoles

Single Multi-Processor Debug



Some classification results

tree_227x227.bin

0 : 937 : 0.024571 n07714990 broccoli
1 : 738 : 0.020480 n03991062 pot, flowerpot
2 : 990 : 0.014112 n12768682 buckeye, horse chestnut, conke
3 : 853 : 0.012724 n04417672 thatch, thatched roof
4 : 483 : 0.010819 n02980441 castle
5 : 595 : 0.010186 n03496892 harvester, reaper
6 : 703 : 0.009717 n03891251 park bench
7 : 984 : 0.009478 n11879895 rapeseed
8 : 410 : 0.008571 n02727426 apiary, bee house
9 : 958 : 0.008400 n07802026 hay

pasta_227x227.bin

0 : 533 : 0.014579 n03207743 dishrag, dishcloth
1 : 770 : 0.013624 n04120489 running shoe
2 : 842 : 0.012774 n04371430 swimming trunks, bathing trunl
3 : 911 : 0.011252 n04599235 wool, woolen, woollen
4 : 415 : 0.011122 n02776631 bakery, bakeshop, bakehouse
5 : 658 : 0.010166 n03775071 mitten
6 : 748 : 0.010100 n04026417 purse
7 : 840 : 0.010091 n04367480 swab, swob, mop
8 : 883 : 0.009004 n04522168 vase
9 : 659 : 0.008804 n03775546 mixing bowl

cup_227x227.bin

0 : 504 : 0.157205 n03063599 coffee mug
1 : 968 : 0.105636 n07930864 cup
2 : 967 : 0.049826 n07920052 espresso
3 : 809 : 0.046456 n04263257 soup bowl
4 : 725 : 0.045708 n03950228 pitcher, ewer
5 : 441 : 0.016535 n02823750 beer glass
6 : 738 : 0.015744 n03991062 pot, flowerpot
7 : 901 : 0.015085 n04579145 whiskey jug
8 : 463 : 0.014723 n02909870 bucket, pail
9 : 969 : 0.013509 n07932039 eggnog

fujisan_227x227.bin

0 : 980 : 0.056212 n09472597 volcano
1 : 977 : 0.023318 n09421951 sandbar, sand bar
2 : 976 : 0.011876 n09399592 promontory, headland, head, foreland
3 : 978 : 0.011804 n09428293 seashore, coast, seacoast, sea-coast
4 : 975 : 0.011525 n09332890 lakeside, lakeshore
5 : 112 : 0.008982 n01943899 conch
6 : 979 : 0.008915 n09468604 valley, vale
7 : 972 : 0.008791 n09246464 cliff, drop, drop-off
8 : 974 : 0.008665 n09288635 geyser
9 : 930 : 0.008184 n07684084 French loaf

rabbit_227x227.bin

0 : 331 : 0.134504 n02326432 hare
1 : 333 : 0.114495 n02342885 hamster
2 : 332 : 0.112388 n02328150 Angora, Angora rabbit
3 : 330 : 0.074634 n02325366 wood rabbit, cottontail, cottc
4 : 279 : 0.030452 n02120079 Arctic fox, white fox, Alopex
5 : 356 : 0.017656 n02441942 weasel
6 : 338 : 0.015547 n02364673 guinea pig, Cavia cobaya
7 : 362 : 0.011832 n02447366 badger
8 : 361 : 0.010876 n02445715 skunk, polecat, wood pussy
9 : 357 : 0.010846 n02442845 mink

cat_227x227.bin

0 : 281 : 0.115859 n02123045 tabby, tabby cat
1 : 282 : 0.054218 n02123159 tiger cat
2 : 287 : 0.047298 n02127052 lynx, catamount
3 : 285 : 0.043925 n02124075 Egyptian cat
4 : 290 : 0.011876 n02128925 jaguar, panther, Panthera onca, F
5 : 292 : 0.011247 n02129604 tiger, Panthera tigris
6 : 286 : 0.011120 n02125311 cougar, puma, catamount, mount
7 : 289 : 0.009661 n02128757 snow leopard, ounce, Panthera ui
8 : 280 : 0.009410 n02120505 grey fox, gray fox, Urocyon cinere
9 : 288 : 0.008847 n02128385 leopard, Panthera pardus

lion_227x227.bin

0 : 291 : 0.103345 n02129165 lion, king of beasts, Panthera le
1 : 366 : 0.042367 n02480855 gorilla, Gorilla gorilla
2 : 367 : 0.031897 n02481823 chimpanzee, chimp, Pan troglo
3 : 261 : 0.030083 n02112350 keeshond
4 : 372 : 0.028271 n02486410 baboon
5 : 297 : 0.028121 n02134418 sloth bear, Melursus ursinus, U
6 : 369 : 0.027656 n02483708 siamang, Hylobates syndactylu
7 : 373 : 0.026115 n02487347 macaque
8 : 365 : 0.018029 n02480495 orangutan, orang, orangutang,
9 : 260 : 0.016685 n02112137 chow, chow chow

panda_227x227.bin

0 : 388 : 0.781814 n02510455 giant panda, panda, panda bear, coon bear, Ailuropoda melanoleuca
1 : 850 : 0.008743 n04399382 teddy, teddy bear
2 : 295 : 0.007253 n02133161 American black bear, black bear, Ursus americanus, Euarctos americanus
3 : 270 : 0.007223 n02114548 white wolf, Arctic wolf, Canis lupus tundrarum
4 : 232 : 0.006188 n02106166 Border collie
5 : 296 : 0.005935 n02134084 ice bear, polar bear, Ursus Maritimus, Thalarctos maritimus
6 : 222 : 0.005294 n02104029 kuvasz
7 : 279 : 0.005271 n02120079 Arctic fox, white fox, Alopex lagopus
8 : 805 : 0.005230 n04254680 soccer ball
9 : 294 : 0.004849 n02132136 brown bear, bruin, Ursus arctos

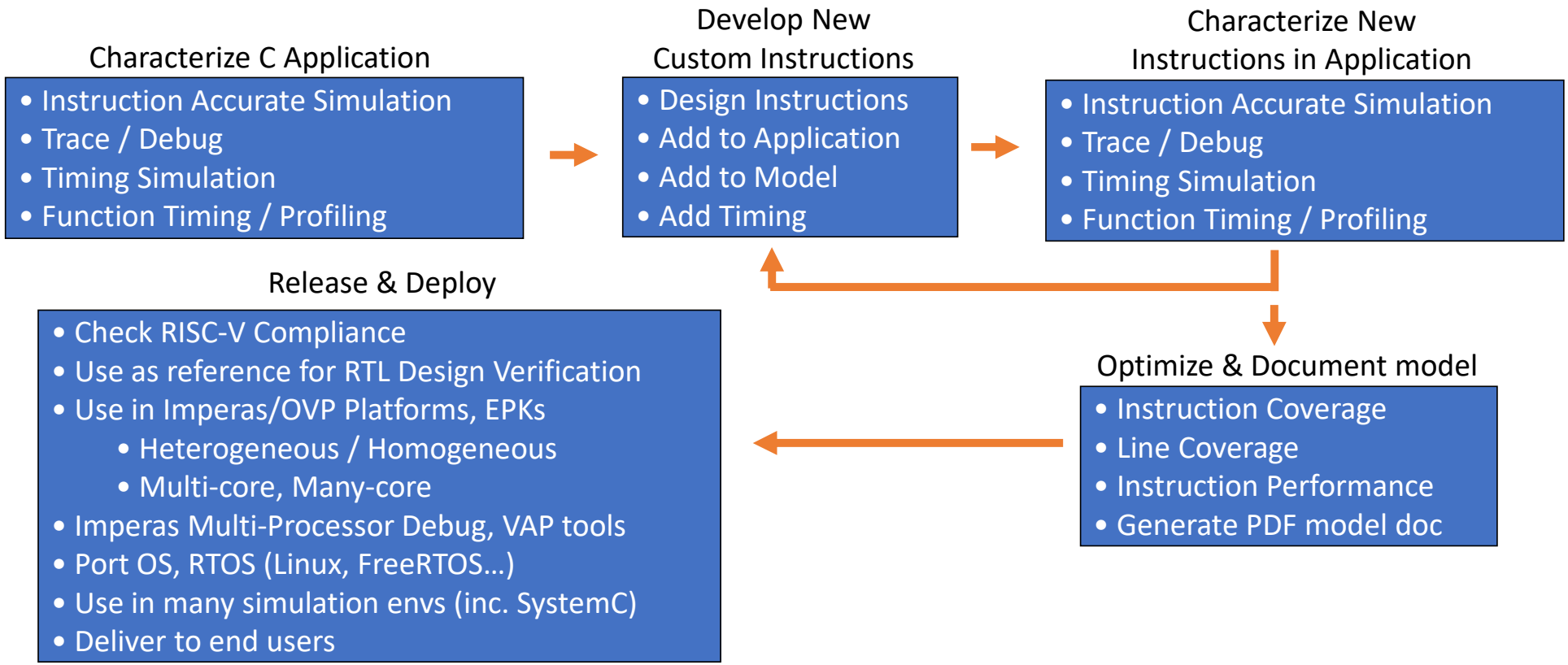
RISC-V Impact on SoC Designs

- **Architecture analysis, including (especially) custom instructions**
- Software development, debug and test
- Processor and SoC verification

How is Processor Performance Optimized?

- Move to multicore
- Optimize the pipeline
- Improve memory usage/latency
- Custom instructions for application/domain optimization

Flow to add new custom instructions



Flow to add new custom instructions

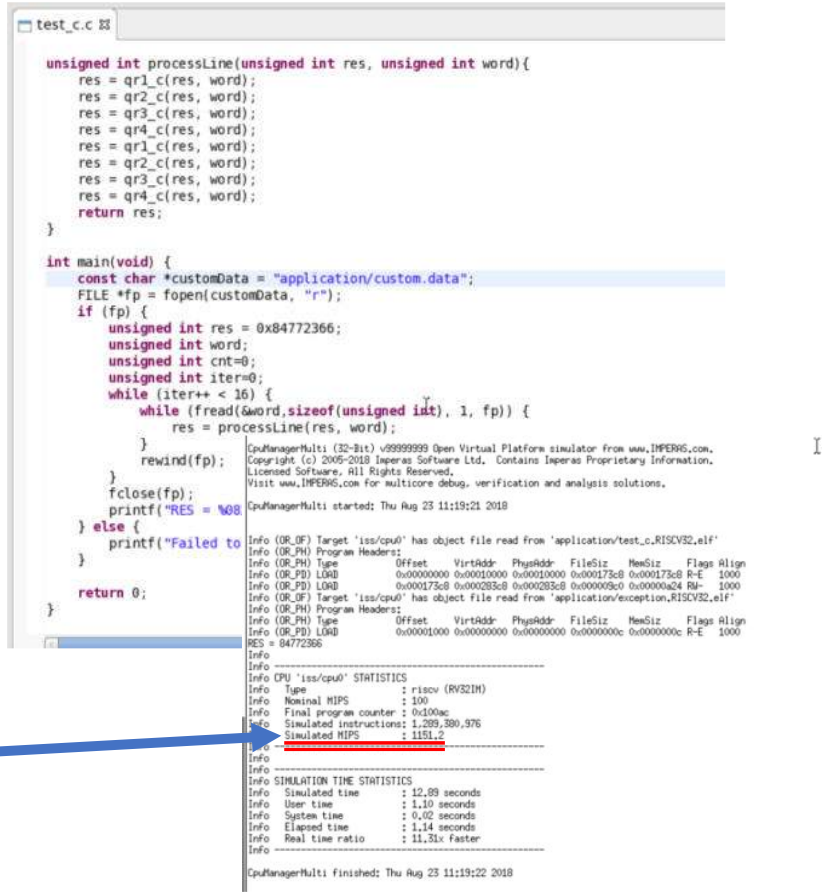
Characterize C Application

- Instruction Accurate Simulation
- Trace / Debug
- Timing Simulation
- Function Timing / Profiling

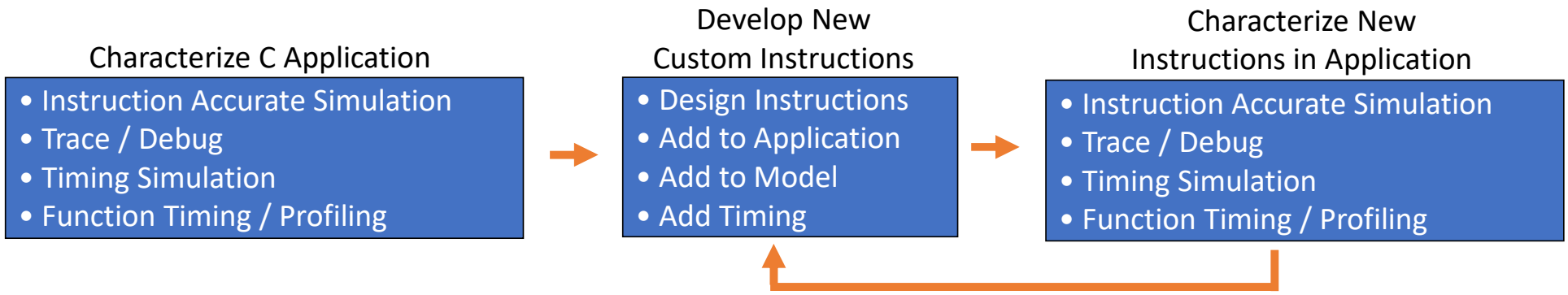
Instruction Accurate (IA) Simulation of C Application

- Cross compiled C application targeting RV32IM
  - Character stream encoder, with ChaCha20 encryption algorithm
- IA simulation
  - Imperas RISC-V ISS with configurable model of RISC-V specification selecting RV32IM
- Semihosting
  - Enables bare metal application to very simply access host I/O

➢runs fast
- Over 1 billion instructions a second (standard PC)
  - Linux and Windows supported host OS



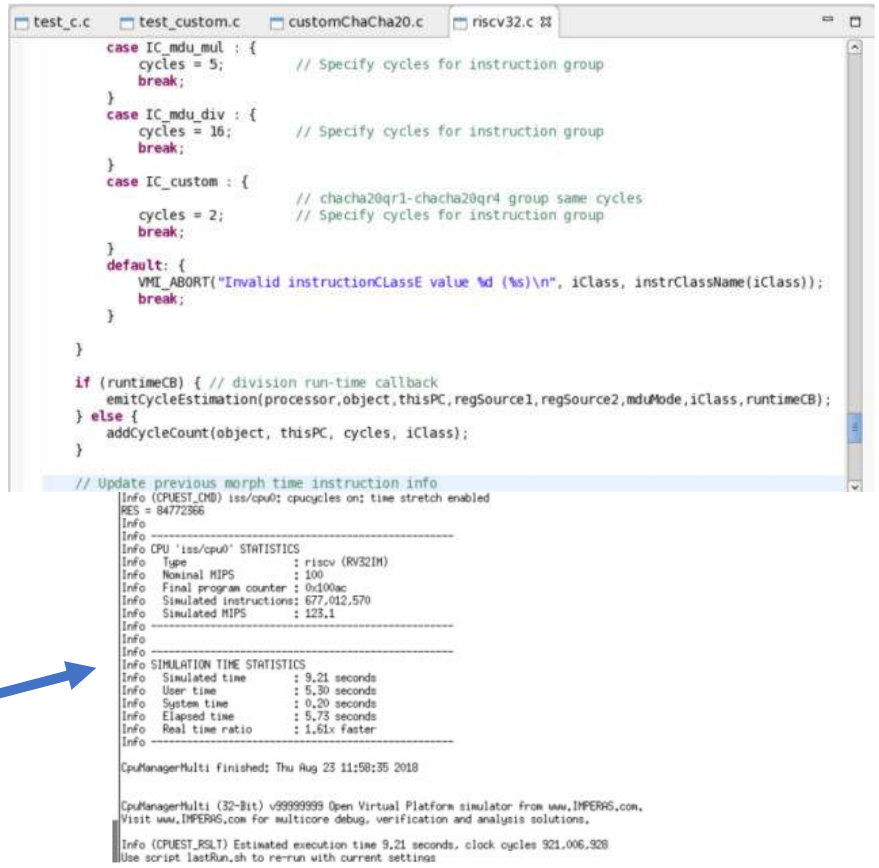
Flow to add new custom instructions



Cycle Approximate Simulation Including Custom Instructions

- IA simulation + timing annotation + custom instructions
  - Includes timing estimation for RV32IM processor
  - Need to add timing estimation for new custom instructions
- Simulate using C code application with inline assembler of custom extensions
- IA simulator + timing tool + custom extension instruction library

➢ See estimated improvement in throughput of application on new processor
  - ➢ Was 16.59 secs without custom instructions
  - ➢ ***Now 9.21 secs with custom instructions***

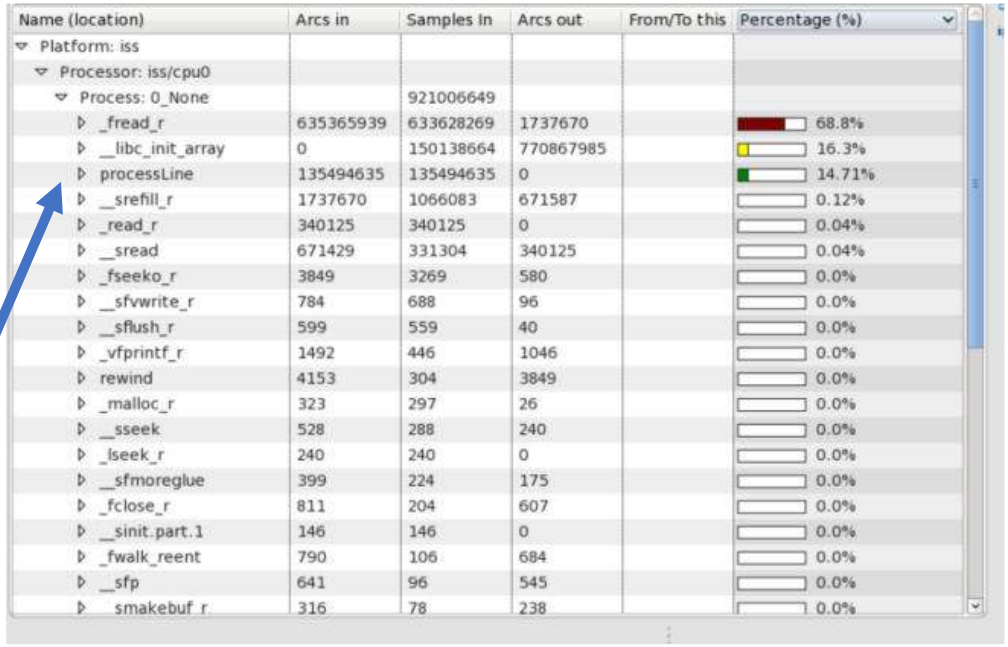


Function Profile of Application with Custom Instructions

- IA simulation + timing annotation + custom instructions with sampled profiling

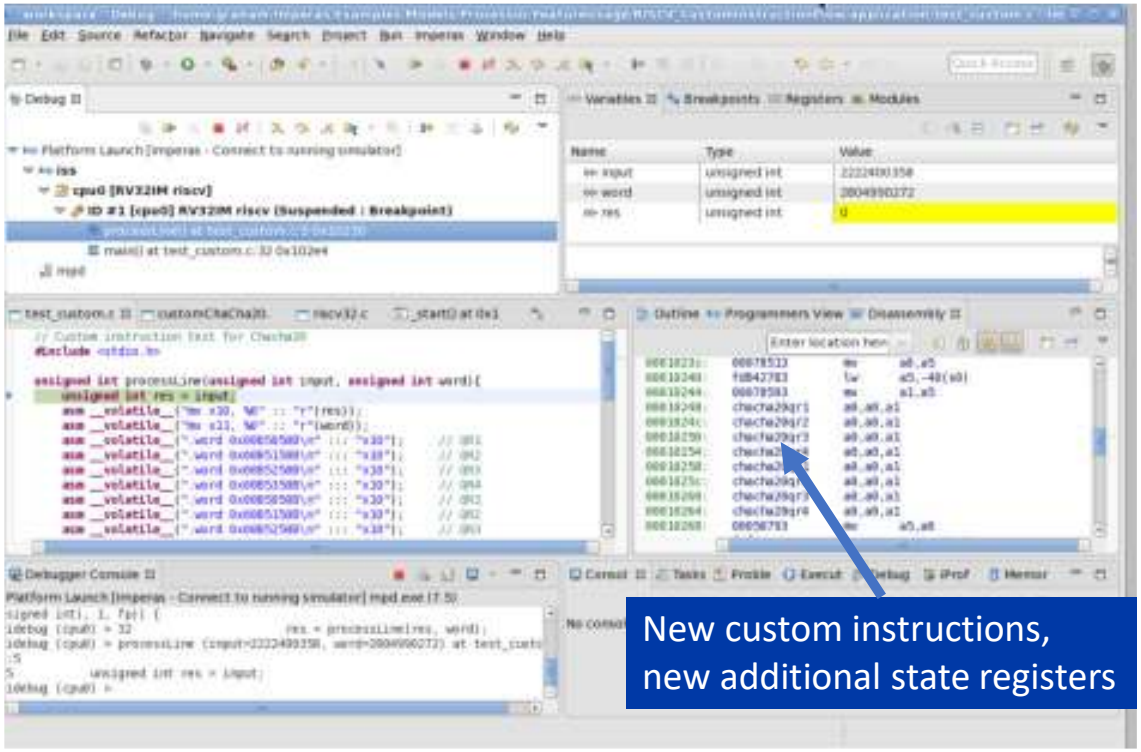
➢Shows where slowest function is
  - Now much faster…

➢Shows benefits of using custom instructions
  - ➢processLine was 21.35% now 16.3%



Name (location)	Arcs in	Samples In	Arcs out	From/To this	Percentage (%)
Platform: iss					
Processor: iss/cpu0					
Process: 0_None		921006649			
_fread_r	635365939	633628269	1737670		68.8%
__libc_init_array	0	150138664	770867985		16.3%
processLine	135494635	135494635	0		14.71%
__srefill_r	1737670	1066083	671587		0.12%
_read_r	340125	340125	0		0.04%
__sread	671429	331304	340125		0.04%
_fseeko_r	3849	3269	580		0.0%
__sfvwrite_r	784	688	96		0.0%
__sflush_r	599	559	40		0.0%
_vfprintf_r	1492	446	1046		0.0%
rewind	4153	304	3849		0.0%
_malloc_r	323	297	26		0.0%
__sseek	528	288	240		0.0%
_lseek_r	240	240	0		0.0%
__sfmoreglue	399	224	175		0.0%
_fclose_r	811	204	607		0.0%
__sinit.part.1	146	146	0		0.0%
_fwalk_reent	790	106	684		0.0%
__sfp	641	96	545		0.0%
__smakebuf_r	316	78	238		0.0%

Software Debug and Analysis Tools Automatically Work With the Custom Instructions




```
CpuManagerMulti started: Thu Aug 23 12:02:30 2018

Info (OR_OF) Target 'iss/cpu0' has object file read from 'application/test_custom,RISCV32,elf'
Info (OR_PH) Program Headers:
Info (OR_PH) Type           Offset     VirtAddr   PhysAddr   FileSiz    MemSiz     Flags Align
Info (OR_PB) LOAD           0x00000000 0x00010000 0x00010000 0x00017270 0x00017270 R-E   1000
Info (OR_PB) LOAD           0x00017270 0x00028270 0x00028270 0x000009c0 0x00000a24 RW-   1000
Info (OR_OF) Target 'iss/cpu0' has object file read from 'application/exception,RISCV32,elf'
Info (OR_PH) Program Headers:
Info (OR_PH) Type           Offset     VirtAddr   PhysAddr   FileSiz    MemSiz     Flags Align
Info (OR_PB) LOAD           0x00001000 0x00000000 0x00000000 0x0000000c 0x0000000c R-E   1000
Info 1330: 'iss/cpu0', 0x000000000010228(processLine+c): fca42e23 sw      a0,-36(s0)
Info 1331: 'iss/cpu0', 0x000000000001022c(processLine+10): fcb42c23 sw      a1,-40(s0)
Info 1332: 'iss/cpu0', 0x000000000010230(processLine+14): fdc42783 lw      a5,-36(s0)
Info   a5 a730c140 -> 84772366
Info 1333: 'iss/cpu0', 0x000000000010234(processLine+18): fef42623 sw      a5,-20(s0)
Info 1334: 'iss/cpu0', 0x000000000010238(processLine+1c): fec42783 lw      a5,-20(s0)
Info 1335: 'iss/cpu0', 0x00000000001023c(processLine+20): 00078513 mv      a0,a5
Info 1336: 'iss/cpu0', 0x000000000010240(processLine+24): fd842783 lw      a5,-40(s0)
Info   a5 84772366 -> a730c140
Info 1337: 'iss/cpu0', 0x000000000010244(processLine+28): 00078593 mv      a1,a5
Info 1338: 'iss/cpu0', 0x000000000010248(processLine+2c): chacha20qr1 a0,a0,a1
Info   a0 84772366 -> e2262347
Info 1339: 'iss/cpu0', 0x00000000001024c(processLine+30): chacha20qr2 a0,a0,a1
Info   a0 e2262347 -> 6e207451
Info 1340: 'iss/cpu0', 0x000000000010250(processLine+34): chacha20qr3 a0,a0,a1
Info   a0 6e207451 -> 10b511c9
Info 1341: 'iss/cpu0', 0x000000000010254(processLine+38): chacha20qr4 a0,a0,a1
Info   a0 10b511c9 -> c2e844db
Info 1342: 'iss/cpu0', 0x000000000010258(processLine+3c): chacha20qr1 a0,a0,a1
Info   a0 c2e844db -> 859b65d8
Info 1343: 'iss/cpu0', 0x00000000001025c(processLine+40): chacha20qr2 a0,a0,a1
Info   a0 859b65d8 -> ba49822a
Info 1344: 'iss/cpu0', 0x000000000010260(processLine+44): chacha20qr3 a0,a0,a1
Info   a0 ba49822a -> 79436a1d
Info 1345: 'iss/cpu0', 0x000000000010264(processLine+48): chacha20qr4 a0,a0,a1
Info   a0 79436a1d -> 35d5aeef
Info 1346: 'iss/cpu0', 0x000000000010268(processLine+4c): 00050793 mv      a5,a0
Info   a5 a730c140 -> 35d5aeef
Info 1347: 'iss/cpu0', 0x00000000001026c(processLine+50): fef42623 sw      a5,-20(s0)
Info 1348: 'iss/cpu0', 0x000000000010270(processLine+54): fec42783 lw      a5,-20(s0)
Info 1349: 'iss/cpu0', 0x000000000010274(processLine+58): 00078513 mv      a0,a5
RES = 84772366
Info
Info ------------------------------------------------------
Info CPU 'iss/cpu0' STATISTICS
Info   Type                : risc
Info   Nominal MIPS        : 100
Info   Final program counter : 0x10
Info   Simulated instructions: 677,
Info   Simulated MIPS      : 1209.
Info ------------------------------------------------------
Info
Info ------------------------------------------------------
```

New custom instructions
in trace disassembly

RISC-V Impact on SoC Designs

- Architecture analysis, including (especially) custom instructions
- **Software development, debug and test**
- Processor and SoC verification

Security is Critical

Nagravision

- Application
  - Devices that protect streaming video
    - Attach to smart tv or set top box
    - Build end user device, software and SoC
  - IoT devices

- Imperas use model
  - Peripheral models for the virtual platform are built by Nagravision (proprietary models) or modified from the OVP Library (standard I/O, e.g. USB)
  - Use Imperas debugger for software debug and for driver-peripheral model software-hardware co-debug
  - Use SW Verification, Analysis and Profiling (VAP) tools such as OS-aware tools, code coverage, memory analysis, …
  - High performance simulation is critical for Continuous Integration (CI) testing

"At NAGRA, we have adopted the Imperas virtual platform-based software development and test tools for our application and firmware teams. The simulation performance, and the tools for software analysis, have added significant value to our daily Agile Continuous Integration (CI) methodology. Our view is that software simulation is mandatory to reach metrics required for high quality secured products."

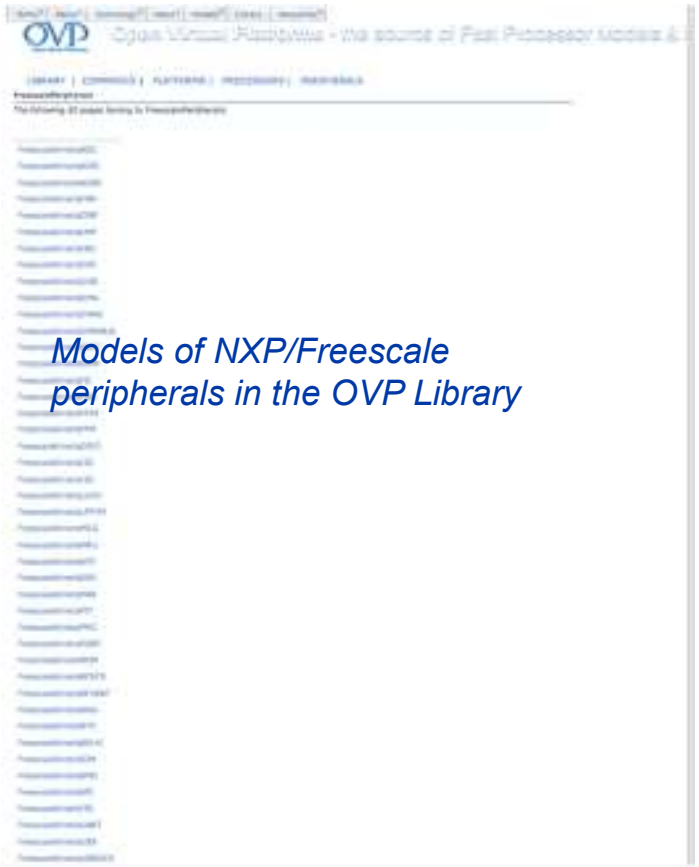
OVP Library of RISC-V Processor Models

- Existing Imperas Open Virtual Platforms (OVP) Fast Processor Models of …
  - Complete envelope models of RV32/64 IMAFDCEVB M/S/U privilege modes
    - **Vector, bit manipulation, crypto instructions were added as soon as specs stabilized**
  - Andes cores: A(X)25, N(X)25, N(X)25F, 27-series including NX27V, …
  - SiFive cores: SiFive Series 2, Series 3 (e.g. E31), Series 5 (e.g. E51, U54), Series 7
  - Open Source cores: OpenHW CORE-V (RI5CY), lowRISC ibex (zero-riscy), plus others
- Custom instructions easily added by user or by Imperas
  - New instructions are added in side file so as not to perturb the verified model
    - Imperas tools work with the complete processor model, including the custom instructions
  - Custom instructions can be analyzed for effectiveness using
    - Instruction coverage and profiling tools
  - Timing estimation tools can be extended to custom instructions
  - Video demo: http://www.imperas.com/risc-v-custom-instruction-design-and-verification-flow-0
- Models are open source, distributed under the Apache 2.0 open source license

"The Imperas virtual platform solutions for software development, debug and test, along with their open-source models, will help accelerate SoC and embedded software development for our customers."
Charlie Hong-Men Su, Ph.D., Andes Technology CTO

Open Virtual Platforms Peripheral Models

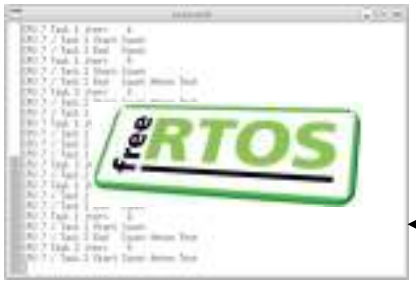
- 100s of peripheral models available in the OVP Library
- All models are open source
  - Distributed under the Apache 2.0 open source license
- All models have both C and SystemC interfaces
- iGen productivity tool enables easy building of peripheral models
- Imperas debugger supports peripheral introspection, such as break points on peripheral registers

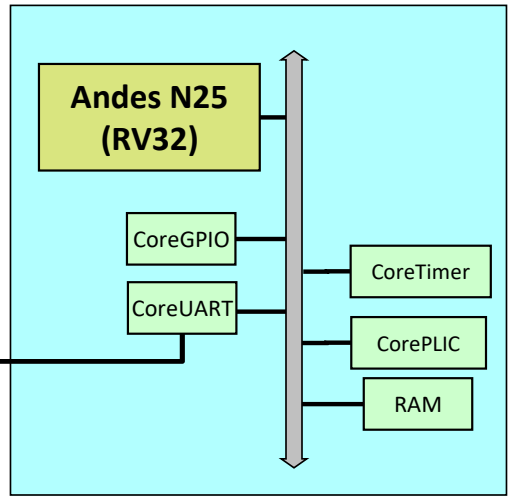


Models of NXP/Freescale peripherals in the OVP Library

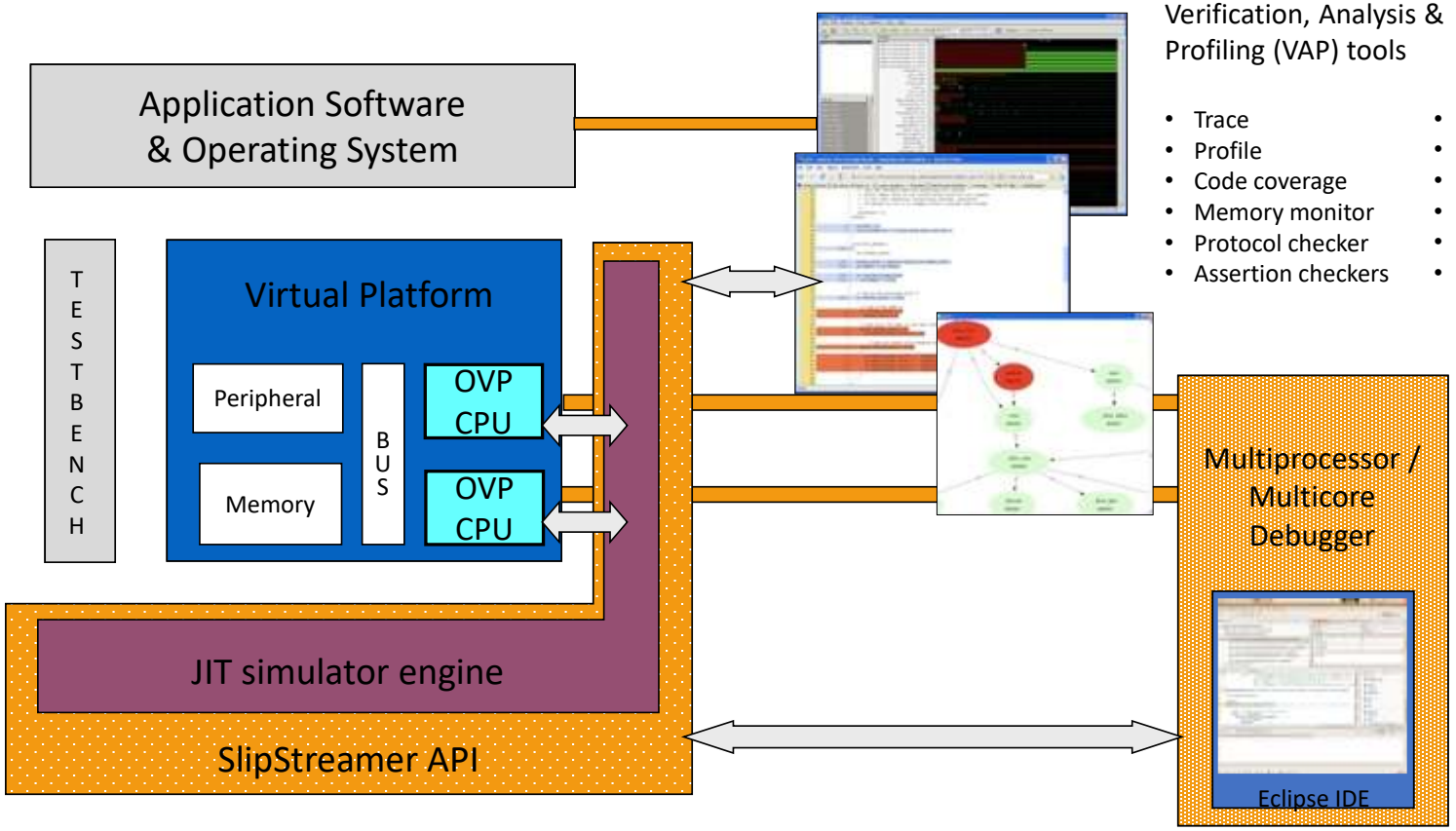
Extendable Platform Kits™ (EPKs™)

- EPKs are virtual platforms, with software running, to help users start quickly
- EPKs include
  - Individual models, binary and source
  - Platform model, binary and source
  - Software and/or OS running on platform
- Over 50 EPKs in library (Arm, MIPS, RISC-V, …)





Imperas Environment



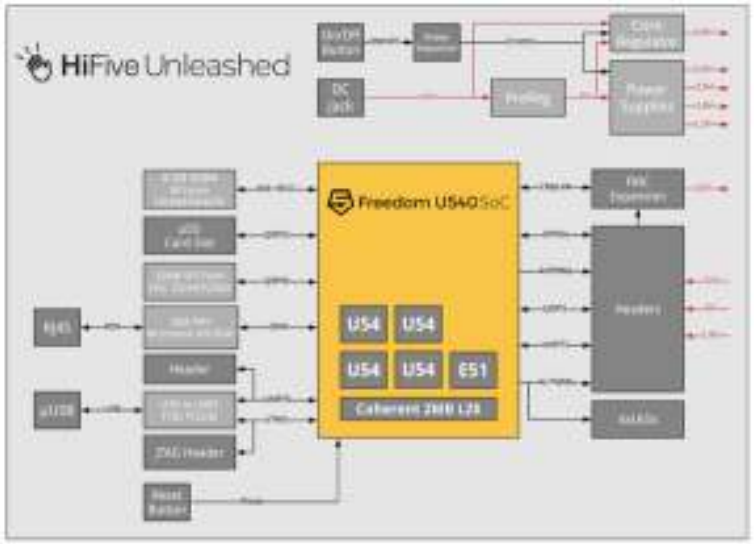
Verification, Analysis & Profiling (VAP) tools

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

Imperas Virtual Platform EPK based on SiFive U540 SoC

The Virtual Platform Provides a Simulation Environment Such That the Software Does Not Know That It Is Not Running On Physical Hardware

SiFive - Block Diagram

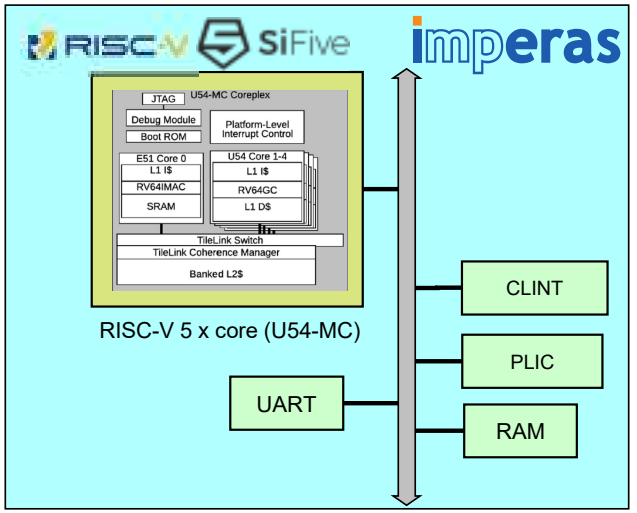


https://www.sifive.com

SiFive - Development Board

https://www.sifive.com

Imperas U540 Virtual Platform

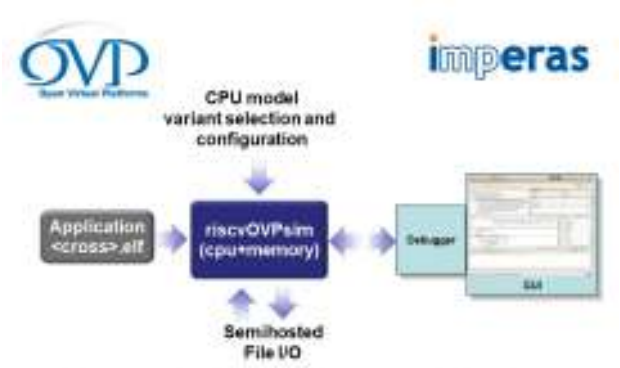


Under 10 seconds to boot up to Linux login prompt!

RISC-V Impact on SoC Designs

- Architecture analysis, including (especially) custom instructions
- Software development, debug and test
- **Processor and SoC verification**

riscvOVPsim, riscvOVPsimPlus Free, RISC-V ISS



Imperas riscvOVPsim Compliance Simulator

"The donation of a robust, commercial-quality simulator – riscvOVPsim – will enable our customers to adopt RISC-V even faster. This is the level of close industry collaboration that will drive the successful adoption of RISC-V."
Yunsup Lee, co-founder and CTO, SiFive

SiFive

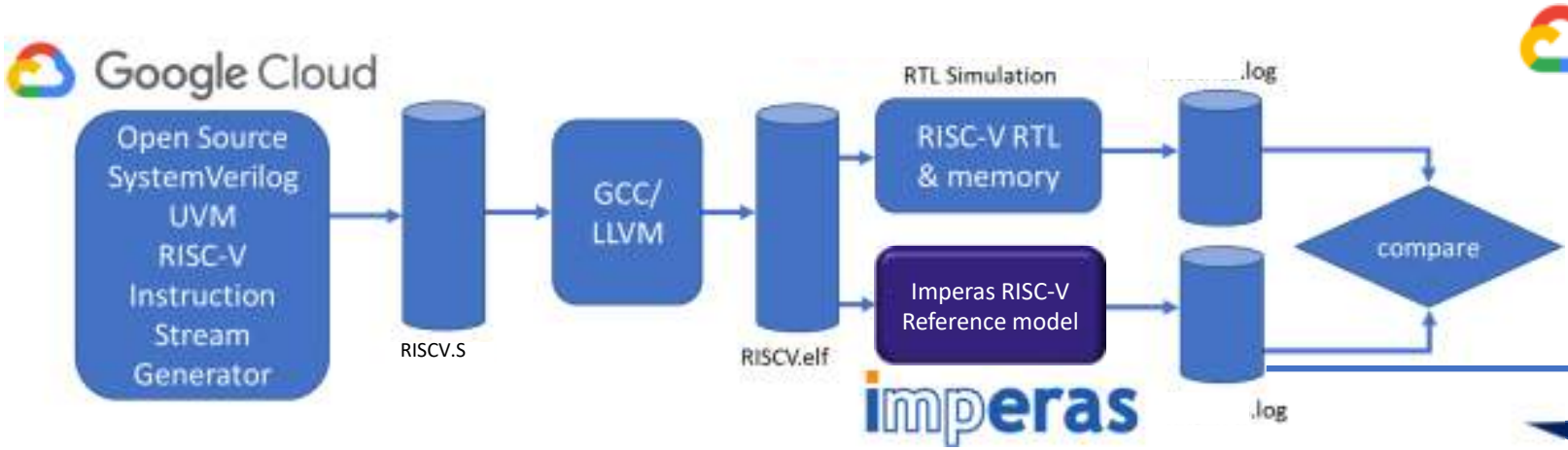
- Industrial quality, free ISS / reference models for compliance testing
  - https://github.com/riscv/riscv-arch-test
- Also used for test development, software development, design verification
- Implements the full RISC-V specification envelope
  - Configurable for all features and versions
- Includes full open source Apache 2.0 model
- Kept up to date for specification changes
  - Updated weekly for Vector, Bitmanip, Crypto spec changes
  - DSP ISA extension under development
- Works 'out of the box' with full tracing, debug, and many other options
- Video: http://www.imperas.com/riscvovpsim-a-complete-risc-v-iss-for-bare-metal-software-development-and-specification-compliance

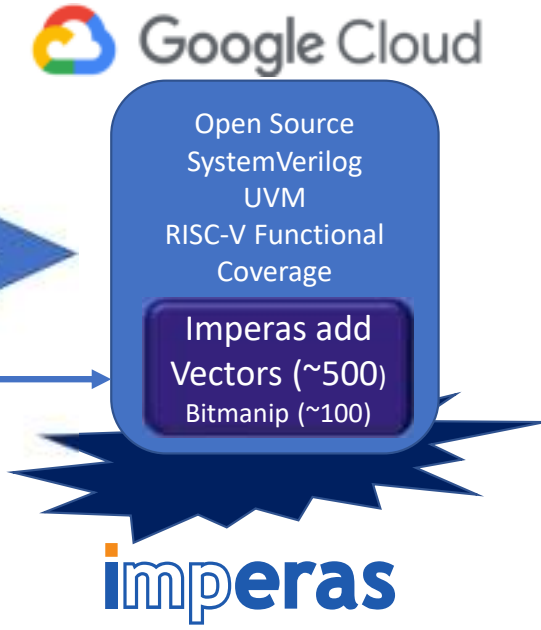
Compliance Is Not Verification

- Need to be clear what focus of testing is
  - Architecture
    - ISA Definition
  - Micro-Architecture
    - In-Order, Out-Of-Order, Simple-Scalar, Super-Scalar, Transactional Memory, Branch Predictors, …
- These are very different
  - One is about ISA specification
  - Other is about details of a specific implementation
  - This is the difference between “Compliance” and Design Verification (DV)

- In the RISC-V International working group, “Compliance” testing is checking the device works within the envelope of the agreed specification
  - i.e. “have you read and understood the specification”
  - Compliance testing is _*part*_ of but not a full hardware verification test plan …

Instruction Stream Generator Baseline Flow



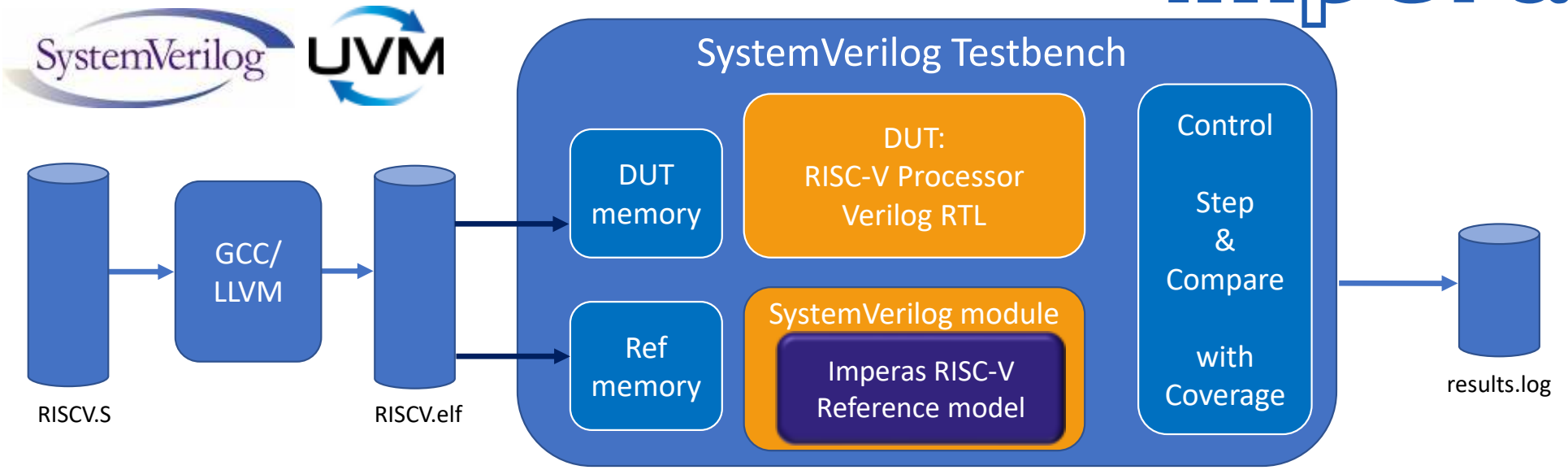


- Google: open source riscv-dv instruction stream generator
- EDA Partners : SystemVerilog design + UVM simulator for RTL
  - working with Cadence, Mentor, Synopsys, and Metrics RTL simulators
- Imperas: model and simulation golden reference of RISC-V CPU
- Imperas: extended this flow to support step-and-compare DV methodology

- Imperas added Vector and Bitmanip extension instructions to the Functional Coverage

(not publicly released)

Step and Compare Methodology



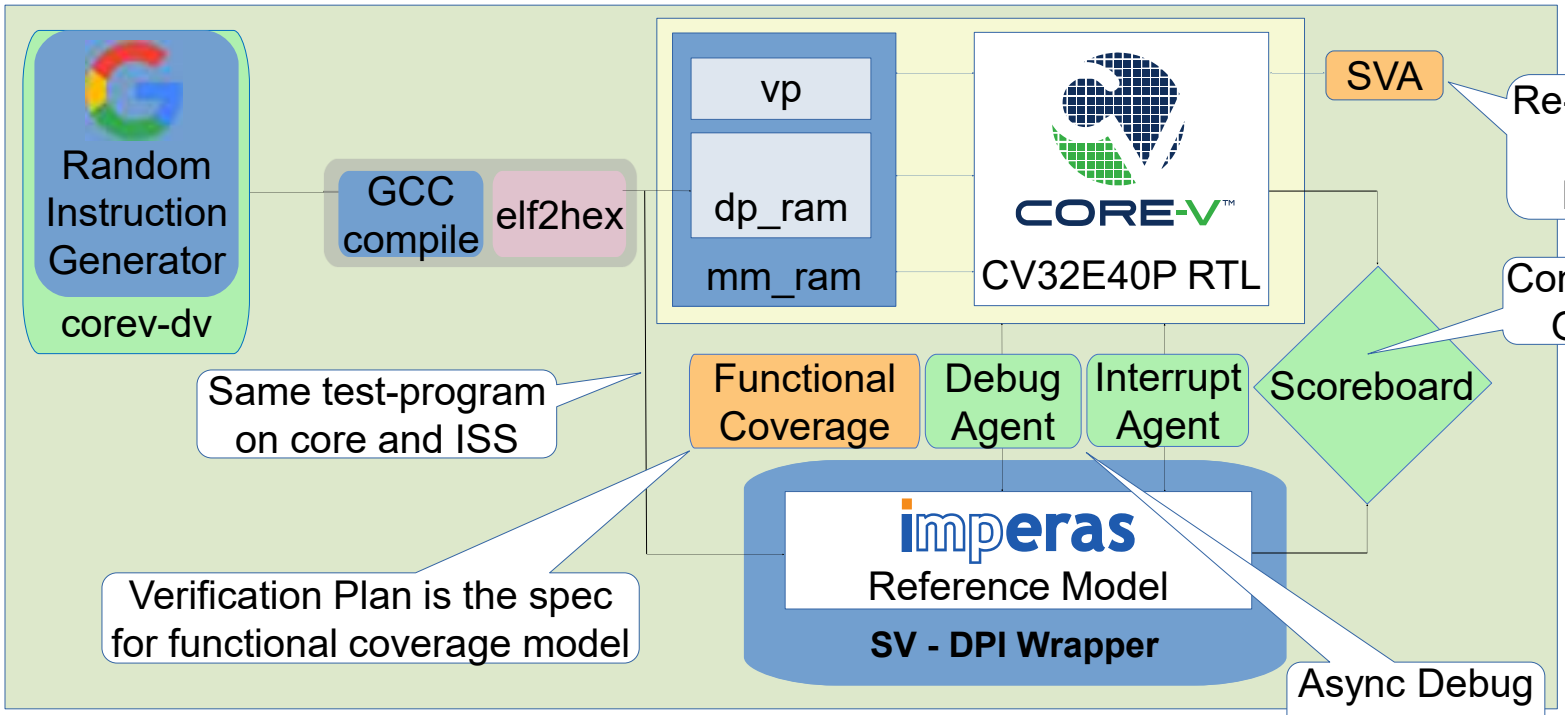
- Testbench loads .elf program into both memories, resets CPUs (RTL and OVP model)
- Steps CPUs (DUT and reference), extracting data, and comparing with coverage
- Advantages
  - Tests stop immediately upon failure – no wasted simulation cycles
  - There is no stored log file – test log data is dynamic
  - Supports indeterminate and asynchronous events (multi-hart processors and interrupts)

Imperas works with Mellanox/NVIDIA on RISC-V Processor Verification

- Imperas Leading RISC-V CPU Reference Model for Hardware Design Verification Selected by Mellanox
  - Verification tools and golden reference model provide support for RISC-V custom instruction extensions and full processor design verification

"We have selected Imperas simulation tools and RISC-V models for our design verification flow because of the quality of the models and the ease of use of the Imperas environment. Imperas reference model of the complete RISC-V specification, the ability to add our custom instructions to the model and their experience with processor RTL DV flows were also important to our decision."
Shlomit Weiss, Senior Vice President of Silicon Engineering at Mellanox Technologies (NVIDIA)

OpenHW Group use Imperas reference model in CORE-V-VERIF SystemVerilog UVM Testbench



Summary remarks

- Fast Imperas simulation allows software to run on virtual platforms many months before RTL commit: heterogeneous platforms with full OS support
- Allows analysis of performance on different hardware configuration choices
- With detailed SW analysis, profiling, performance and debug tooling
- All the current RISC-V specification features Plus custom instruction support
- Imperas RISC-V Reference Model for Processor DV and SystemVerilog UVM

Architectural Exploration, Software development and Processor DV

RISC-V Reference Models across all phases of development

Thank You

- Visit www.imperas.com/riscv and www.OVPworld.org/riscv for more information

riscvOVPsimPlus

free reference model and test suites available at

https://www.ovpworld.org/riscvOVPsimPlus/

Simon Davidmann
SimonD@imperas.com