2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION CHINA

accellera SYSTEMS INITIATIVE

Catapult HLS & HLV Platform – Enabling Agile & High Quality IC Design

Shanghai | September 20, 2023

Agenda

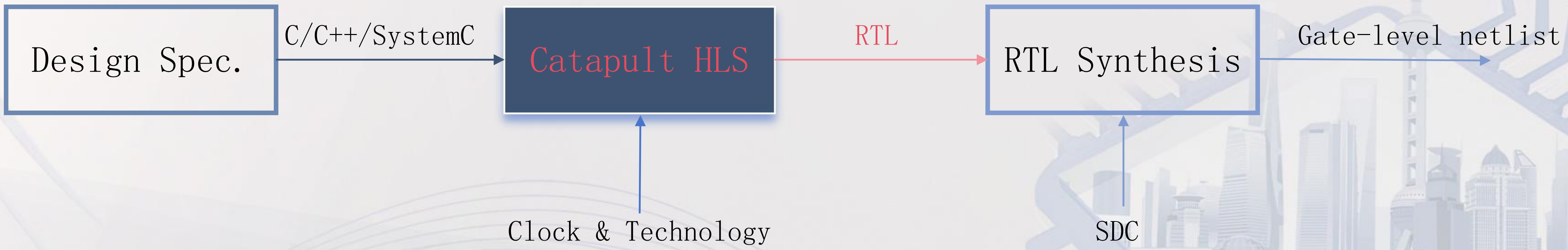
- HLS Introduction
- HLS Design & Verification Flow
- Customer Success Stories

What is HLS ?

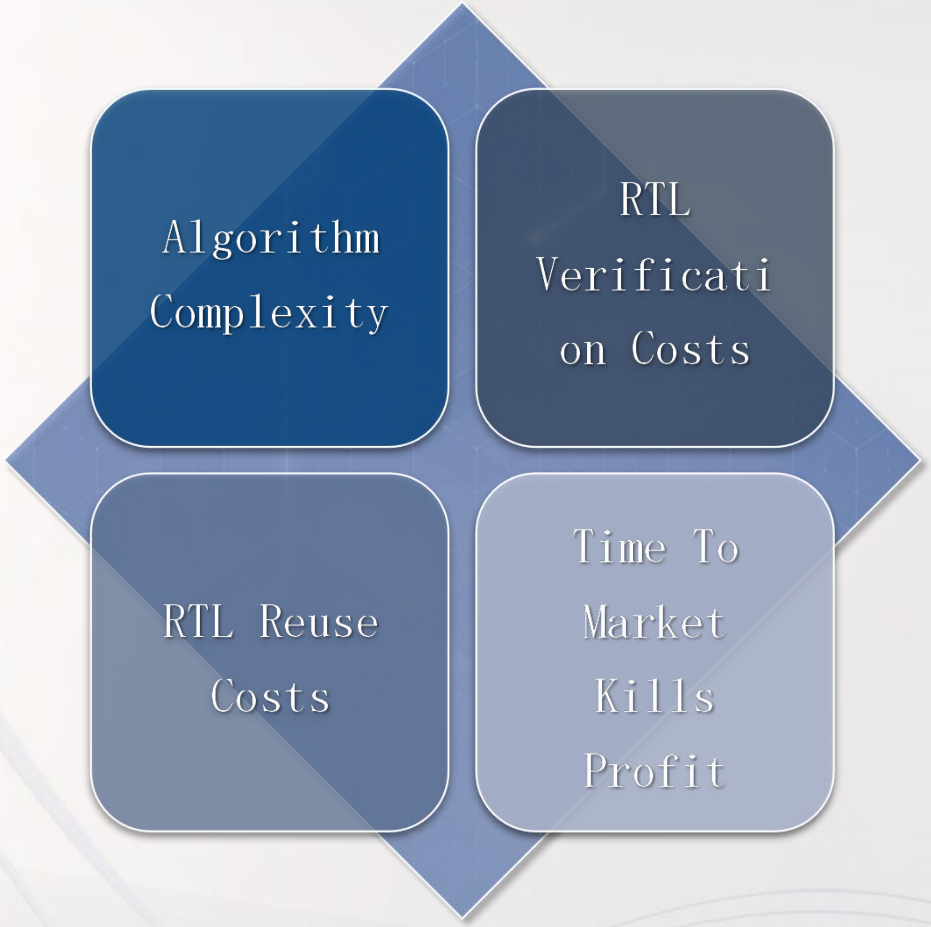
High-level synthesis (HLS) creates RTL implementations from abstract specifications described with C, C++, SystemC, etc.

HLS is still hardware design

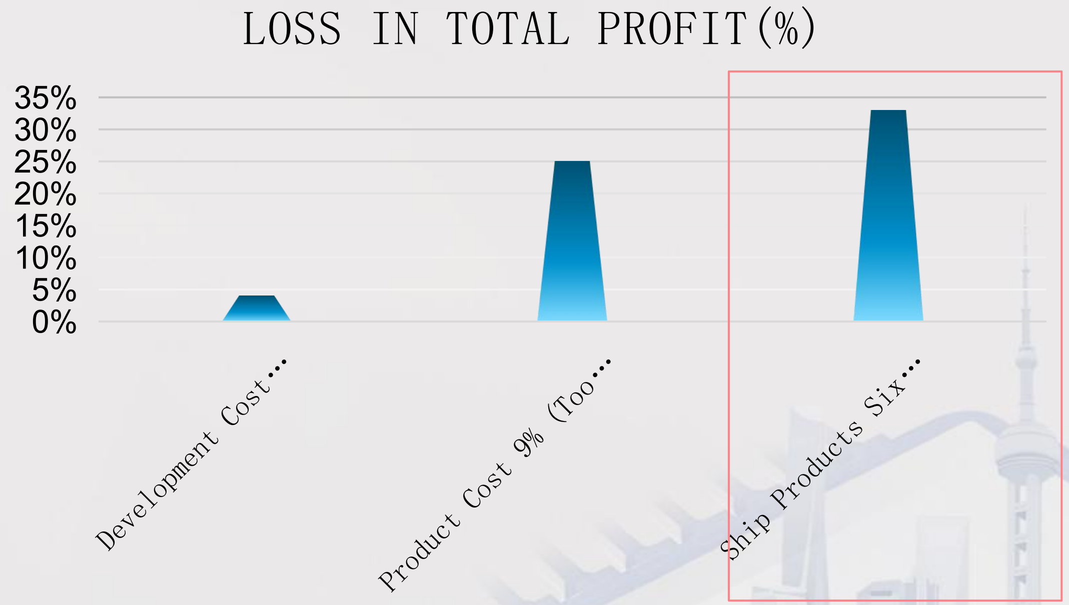
- HLS does NOT “translate” any working C++ code into a good HW
- HLS does NOT turn a SW engineer into a HW designer
- HLS does NOT replace RTL designers, it empowers them



Why HLS ? Time is Money



Being six months late can lose 33% of lifetime profit



* In a 20% growth rate market, with 12% annual price erosion and a five-year total product life.
Source: McKinsey & Co.

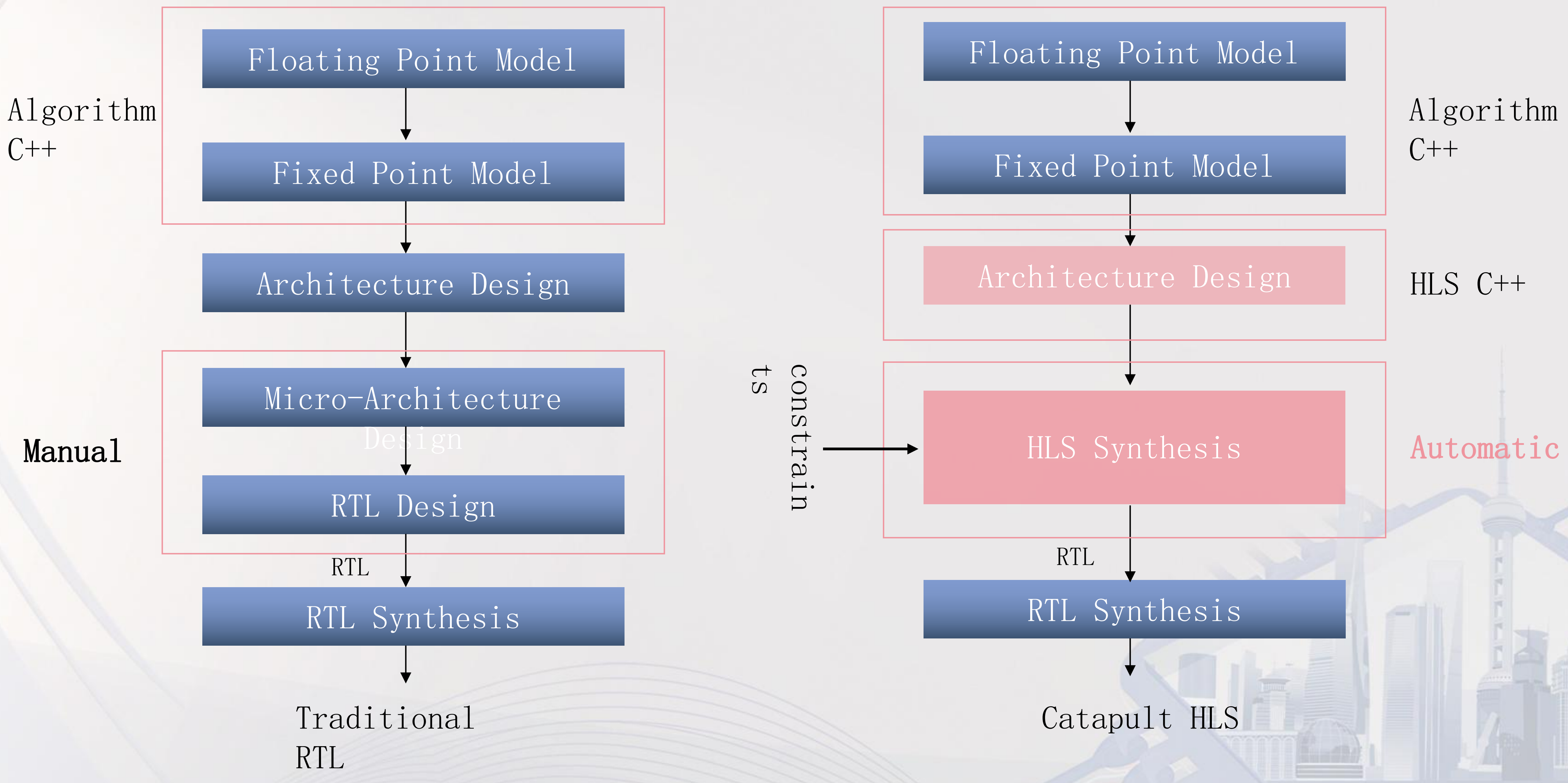
Market Drivers for High-Level Synthesis

Key Markets have Significant Pressures for New Designs

- Time To Market with competitive QoR
- Retarget I.P. from FPGA to SoC geometries
- Handle late changing specifications
- Verification and debug cost

- Computer Vision and Neural Computing
- High Bandwidth and Cellular Communication
- Image Processing, Video and Compression

Traditional RTL Design Flow vs. Catapult HLS Design Flow



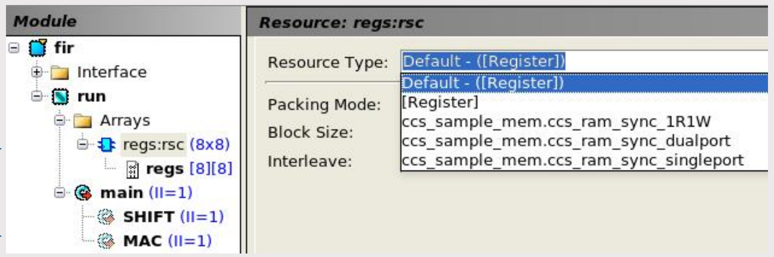
Catapult Synthesis - Micro Architecture Control

User control over the micro-architecture implementation

- Parallelism, Throughput, Area, Latency (loop unrolling & pipelini
- Memories (DPRAM/SPRAM/split/bank) vs Registers (Resource allocati

Explore μArchitectures with constraints(Not by changing the source coc

- Evaluate PPA alternatives for each design
- Memory access minimization
- Banking & interleaving

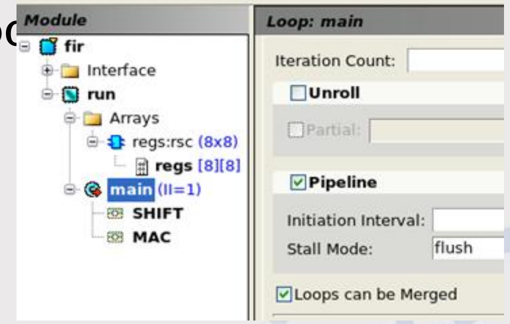


Module
- fir
  - Interface
  - run
    - Arrays
      - regs:rsc (8x8)
        - regs [8][8]
    - main (II=1)
      - SHIFT (II=1)
      - MAC (II=1)

Resource: regs:rsc

Resource Type: Default - ([Register])
Default - ([Register])
[Register]
ccs_sample_mem.ccs_ram_sync_1R1W
ccs_sample_mem.ccs_ram_sync_dualport
ccs_sample_mem.ccs_ram_sync_singleport

Packing Mode:
Block Size:
Interleave:

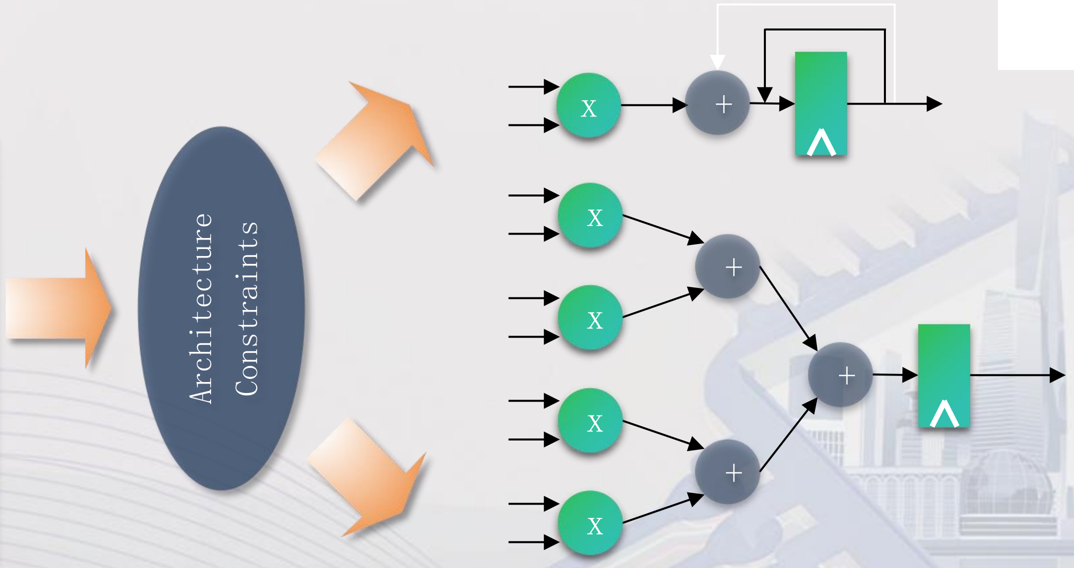


Module
- fir
  - Interface
  - run
    - Arrays
      - regs:rsc (8x8)
        - regs [8][8]
    - main (II=1)
      - SHIFT
      - MAC

Loop: main

Iteration Count:
☐ Unroll
☐ Partial:
☑ Pipeline
Initiation Interval:
Stall Mode: flush
☑ Loops can be Merged

```
int mac(
   char data[N],
   char coef[N]
) {
   int accum=0;
   for (int i=0; i<N; i++)
     accum += data[i] * coef[i];
   return accum;
}
```



Catapult Synthesis - Scheduling (Pipelining)

Multi-objective scheduling

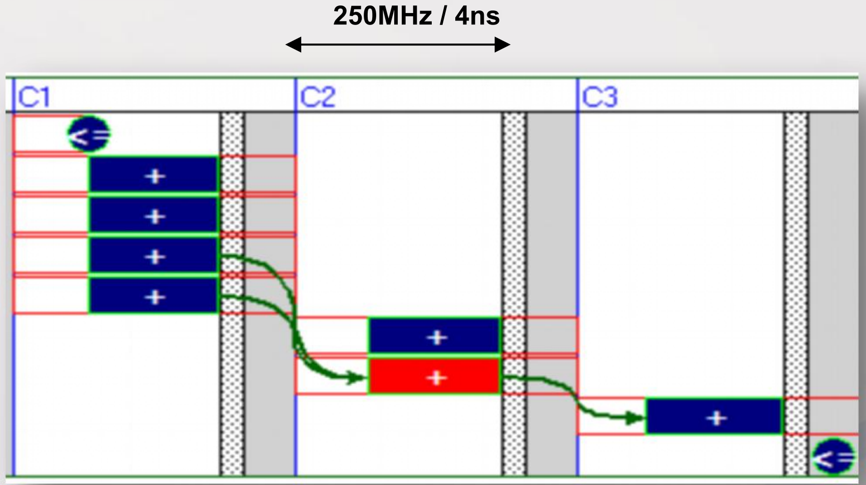
- Area/Latency driven data path scheduling

Arithmetic optimizations and bit-width trimming

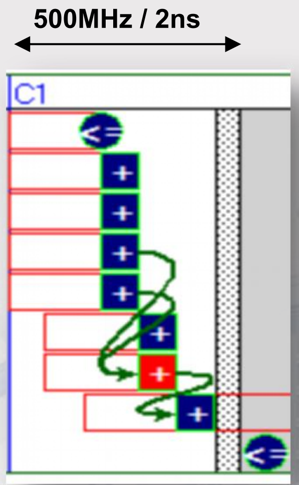
Eliminates RTL technology penalty of I.P. reuse

```
for (int i=0; i<8; i++){
  tmp+=a[i];
}
```

Technology Neutral
Description

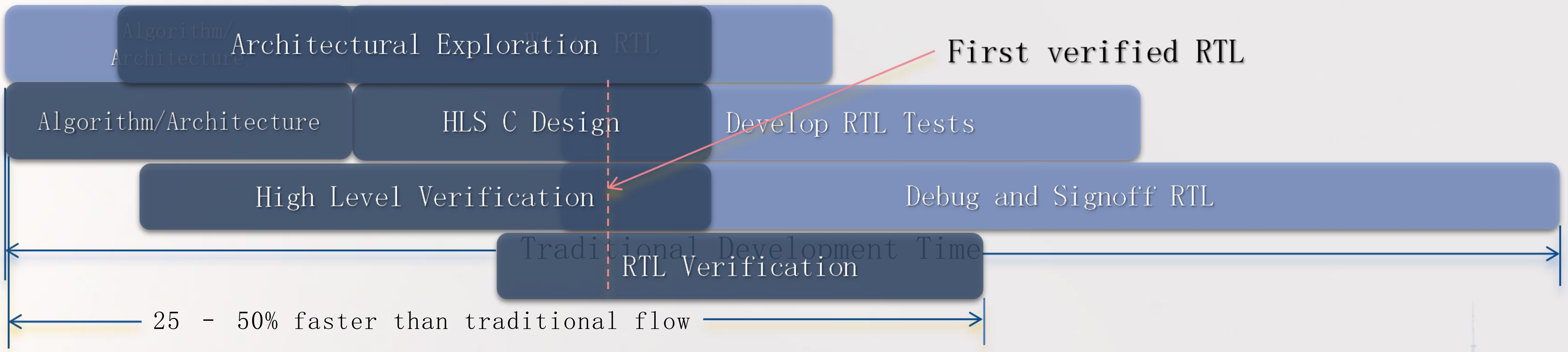


FPGA or SLOW ASIC
Delay of a 16bit add: 2.1 ns
Latency: 3 cycles



Faster Process
Delay of a 16bit add: 0.3 ns
Latency: 1 cycle

Cut Design/Verification Time and Cost



Shift Left for Design/Verification

- Start verification with Algorithm/Architecture teams – NEW Job description High-Level Verification Engineer
- Reduced verification costs by 80%, time by 50%
- Reduce from 12 weeks to 2 weeks and eliminated separate verification engineering all together

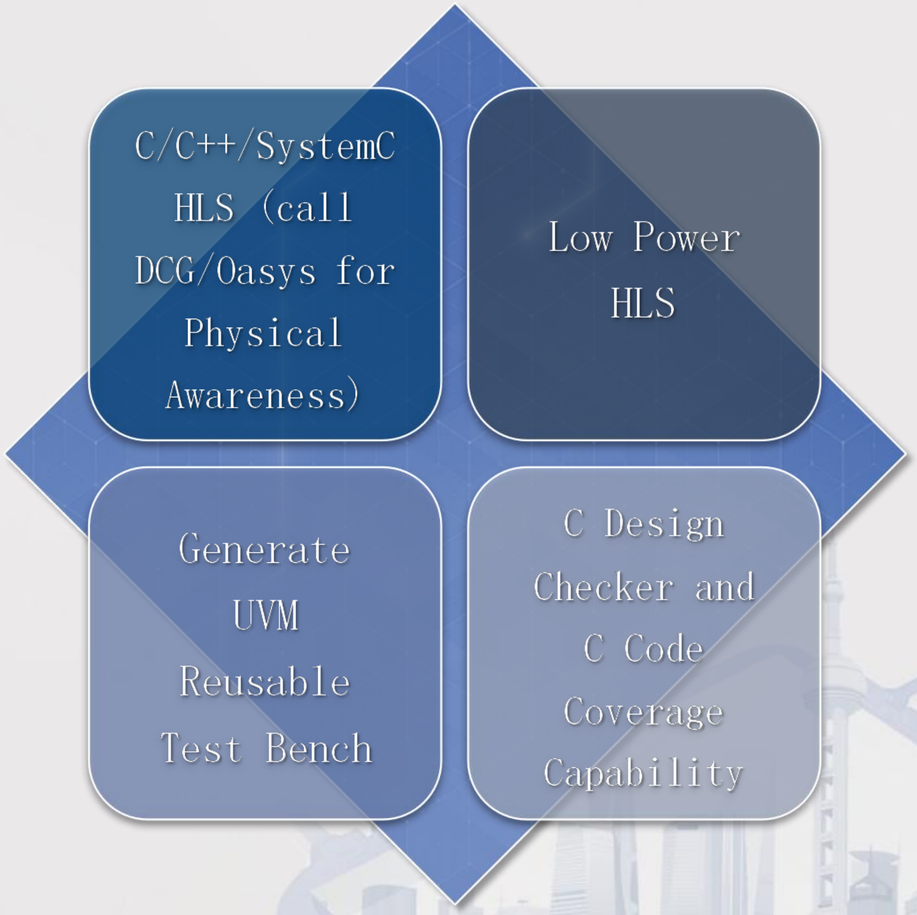
Introducing the Catapult HLS Platform

High Quality RTL Synthesized from C/C++/SystemC with Physical Awareness

Verification for C/C++/SystemC HLS Design and Production flow into RTL

Catapult Code Coverage

PowerPro under-the-hood for Best Power Optimized RTL


C/C++/SystemC HLS (call DCG/Oasys for Physical Awareness)
Low Power HLS
Generate UVM Reusable Test Bench
C Design Checker and C Code Coverage Capability

Language Freedom – C++ | System C

SystemC

Untimed, Loosely-timed,

Cycle-accurate

Exploration &

Implementation

Control Logic &

Algorithms

Flexibility to use the best language for a team, project or application

Teams may select one language, but company may use both

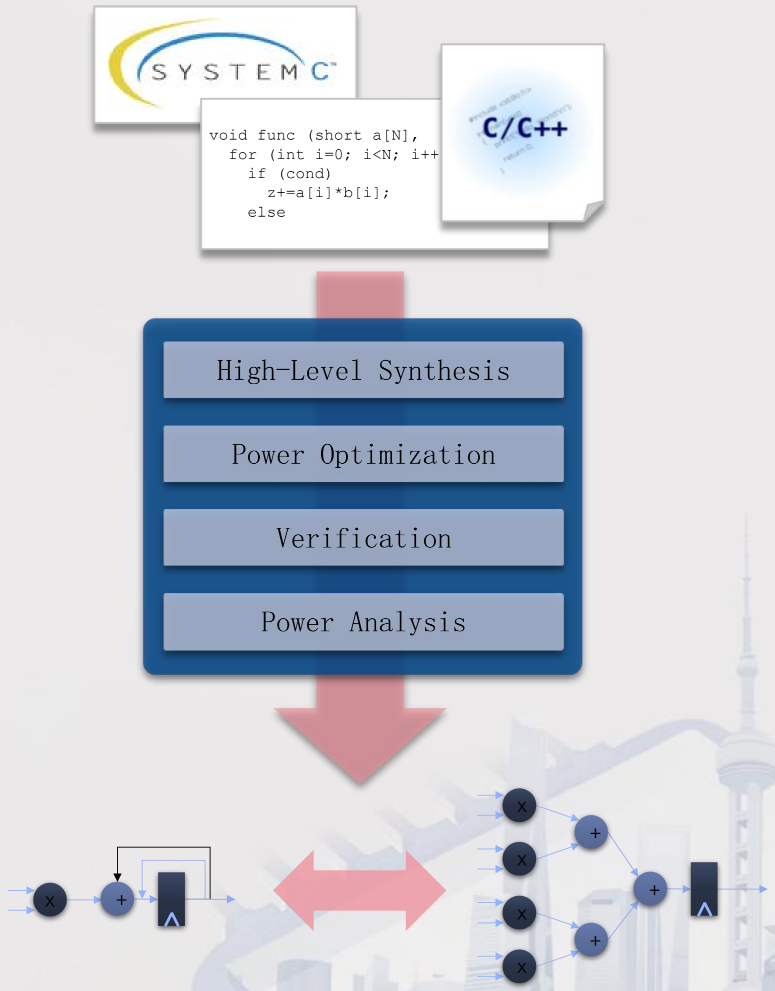
PowerPro Under-the-hood

Integrated Early RTL Power Estimation with PowerPro “under the hood”

Generate power optimized RTL with Catapult Ultra

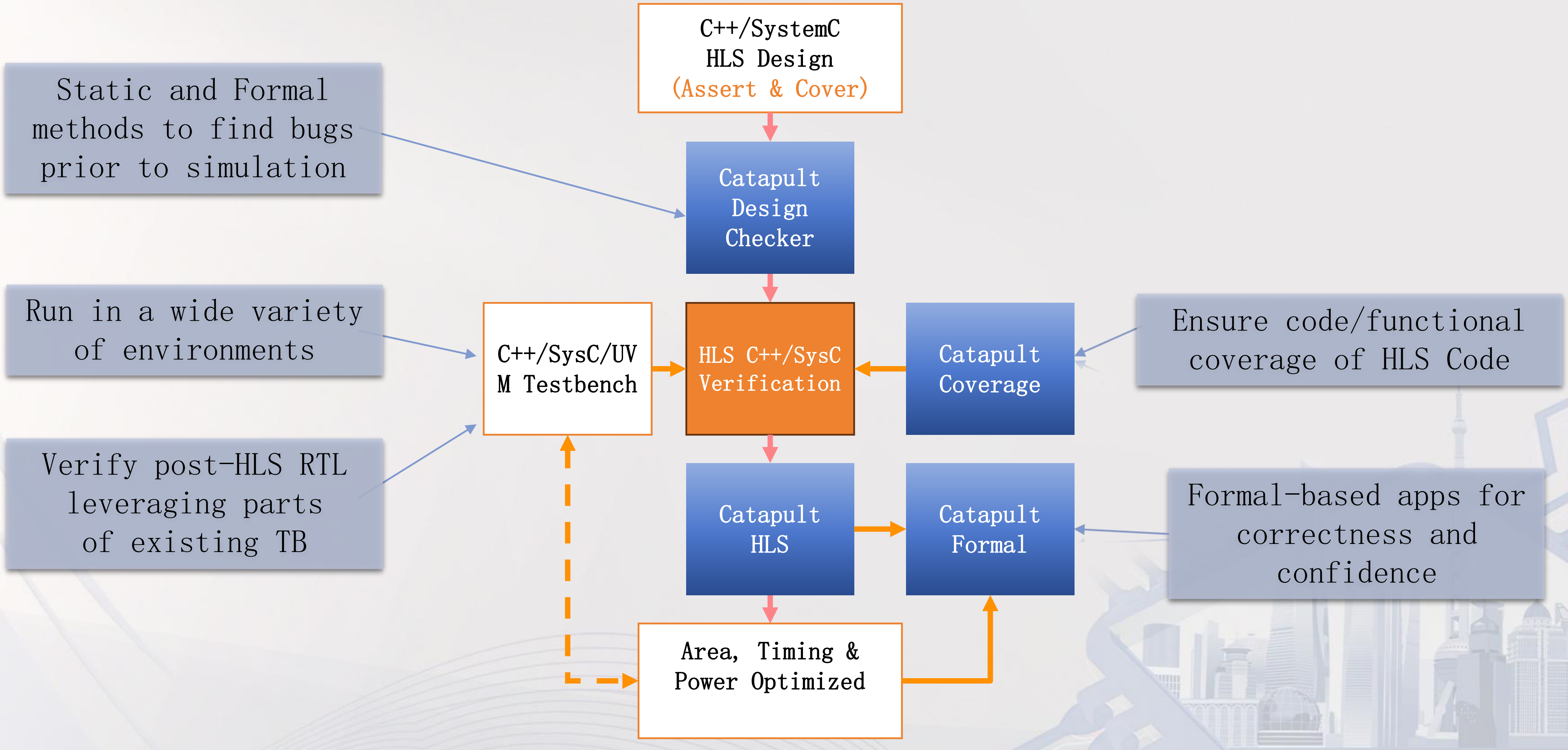
- Automatic Optimization with PowerPro Engine
- Converge rapidly on optimal solution

Only Catapult Ultra has this integrated technology



```
void func (short a[N],
  for (int i=0; i<N; i++
    if (cond)
      z+=a[i]*b[i];
    else
```

Catapult High-Level Verification Flow


Static and Formal methods to find bugs prior to simulation
Run in a wide variety of environments
Verify post-HLS RTL leveraging parts of existing TB
C++/SystemC HLS Design (Assert & Cover)
Catapult Design Checker
C++/SysC/UVM Testbench
HLS C++/SysC Verification
Catapult Coverage
Catapult HLS
Catapult Formal
Area, Timing & Power Optimized
Ensure code/functional coverage of HLS Code
Formal-based apps for correctness and confidence

Properties in HLS

Deploy properties to catch issues early

Catapult supports Immediate Assertions and Cover Properties in HLS C++ and SystemC

Catapult propagates assertions and cover properties from HLS source to RTL

Assertions in generated HDL
SVA, PSL or OVL

"Applying common RTL debug and verification techniques to HLS design source"

```
#include <ac_assert.h>

#pragma hls_design top
uint16 alu(uint8 a, uint8 b, opcode_t opcode)
{
   uint16 r = 0;
   switch(opcode) {
    case ADD:
      r = a+b;
      break;
    case SUB:
      assert(a>=b); // no negative results
      r = a-b;
      break;
    case DIV:
      assert(b!=0); // no divide-by-zero
      r = a/b;
      break;
  }

  // Cover all of the possible opcodes
  cover((opcode==ADD));
  cover((opcode==SUB));
  cover((opcode==DIV));

  return r;
}
```

Catapult Design Checker
Static and Formal analysis to find issues early

Quickly and easily find coding bugs and errors before synthesis or simulation

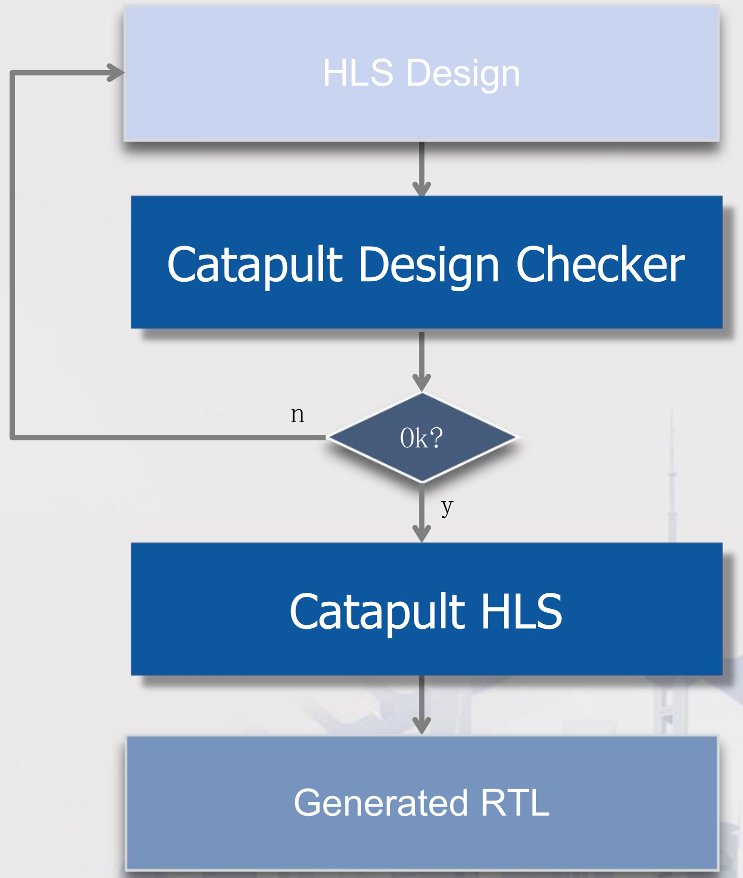
Some C++ language behavior not well defined or too ambiguous for hardware

- Leads to mismatches between C++ and RTL
- Difficult to debug in dynamic simulation

Combination of static ‘lint’ checks, Quality of Results (QofR) checks, and formal properties checking

- e.g. Out of bounds array read and writes (ABR, ABW) and Uninitialized memory reads (UMR)

“Clean HLS design source results in less debugging of posy-HLS RTL”



Catapult Code Coverage

Achieve Coverage Closure on HLS design source

Bring RTL coverage into HLS world

Match coverage concepts from RTL

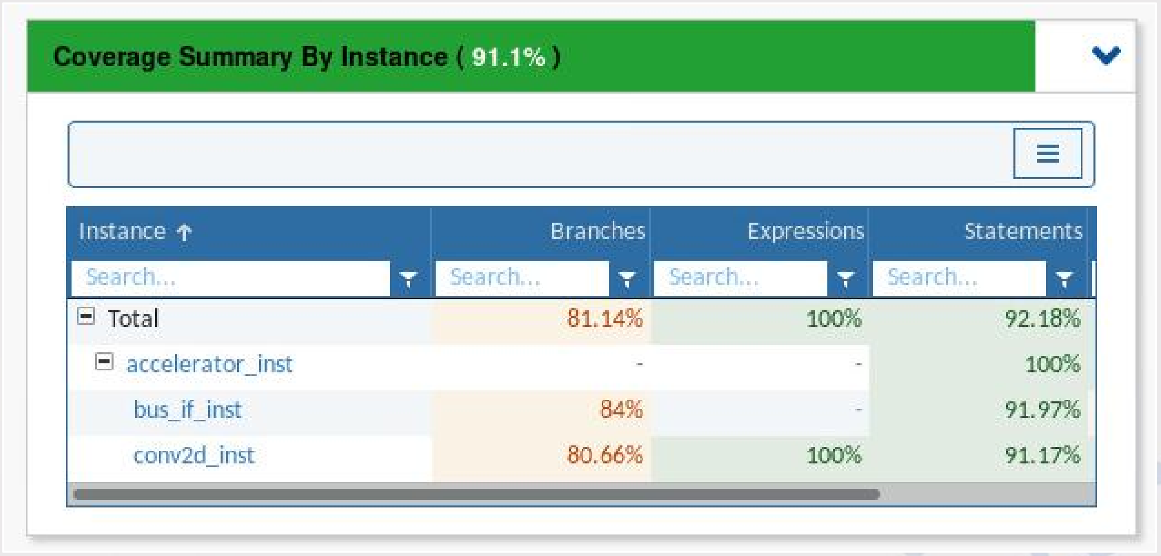
- Code coverage – statement, branch, focused expression coverage (FEC)
- Functional coverage – covergroups, coverpoints, bins & crosses

HLS-aware code coverage

Coverage data written into UCDB

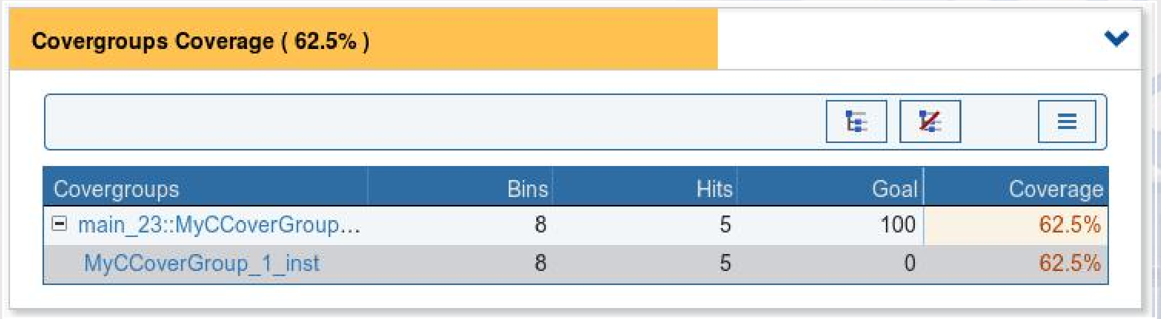
- Enables use of all Questa verification analysis & management utilities

"Code coverage closure on HLS source translates into 80-85% out of box code coverage on RTL"



Coverage Summary By Instance (91.1%)

Instance	Branches	Expressions	Statements
Total	81.14%	100%	92.18%
accelerator_inst	-	-	100%
bus_if_inst	84%	-	91.97%
conv2d_inst	80.66%	100%	91.17%



Covergroups Coverage (62.5%)

Covergroups	Bins	Hits	Goal	Coverage
main_23::MyCCoverGroup...	8	5	100	62.5%
MyCCoverGroup_1_inst	8	5	0	62.5%

Automatic RTL Verification Environment

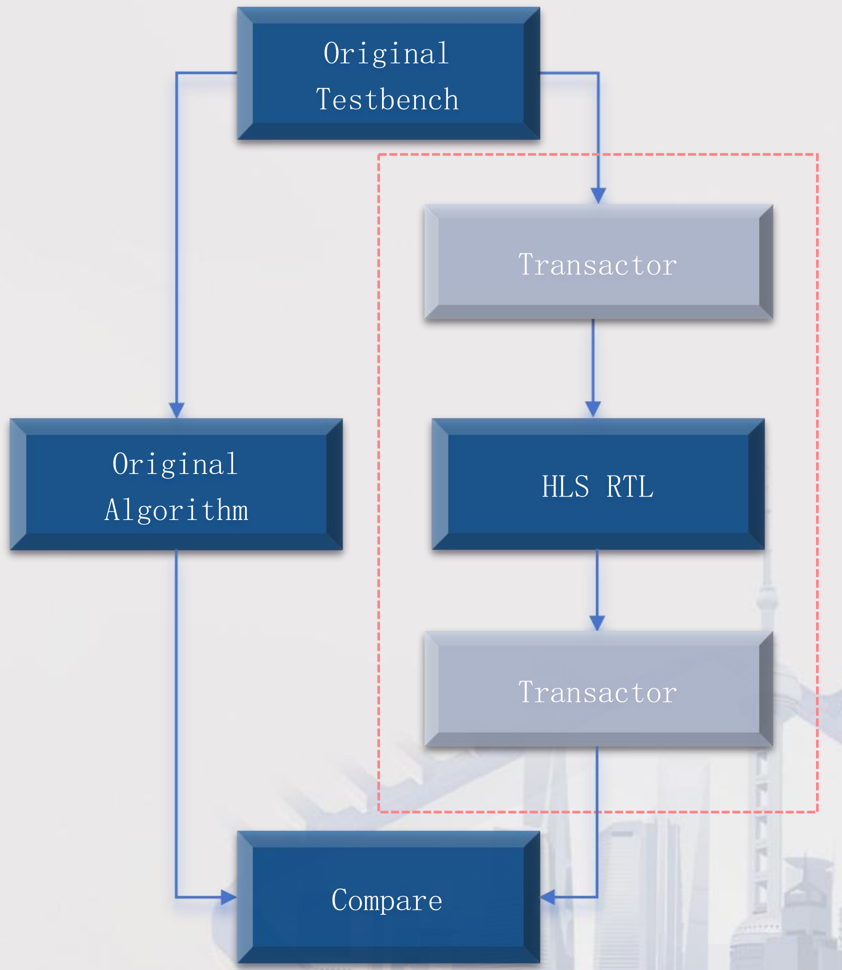
SCVerify/CoSim flow builds RTL functional test environment

- Questa/ModelSim
- NCSim | VCS-MX

Original C++/SystemC testbench is reused to simulate the RTL design

Transactors convert function calls to pin-level signal activity in C++

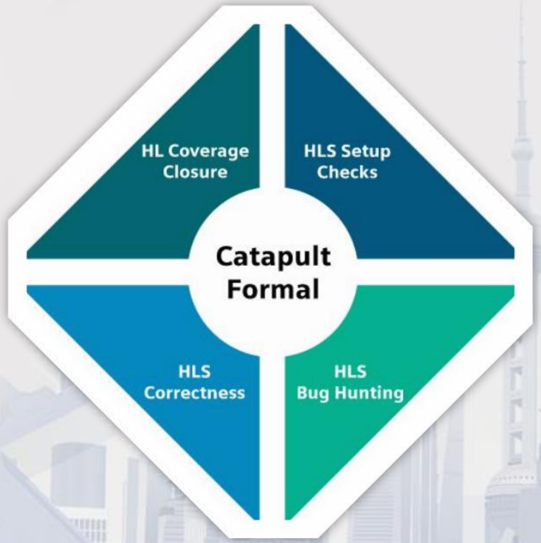
Not typically used by verification team



Catapult Formal

Catapult Formal is a suite of verification Apps controlled by the Catapult GUI or Catapult command TCL files.

- CFormal IMP : Synthesis constraint verification for memory access dependencies
- CFormal Idle : RTL verification of the synthetic Idle-detection logic
- CFormal Stall : RTL verification of Stall handshakes and wait controllers
- CFormal SLEC : C-to-RTL leaf module equivalence check
- +new Apps under construction (CCoverCheck and CFormalAssert)

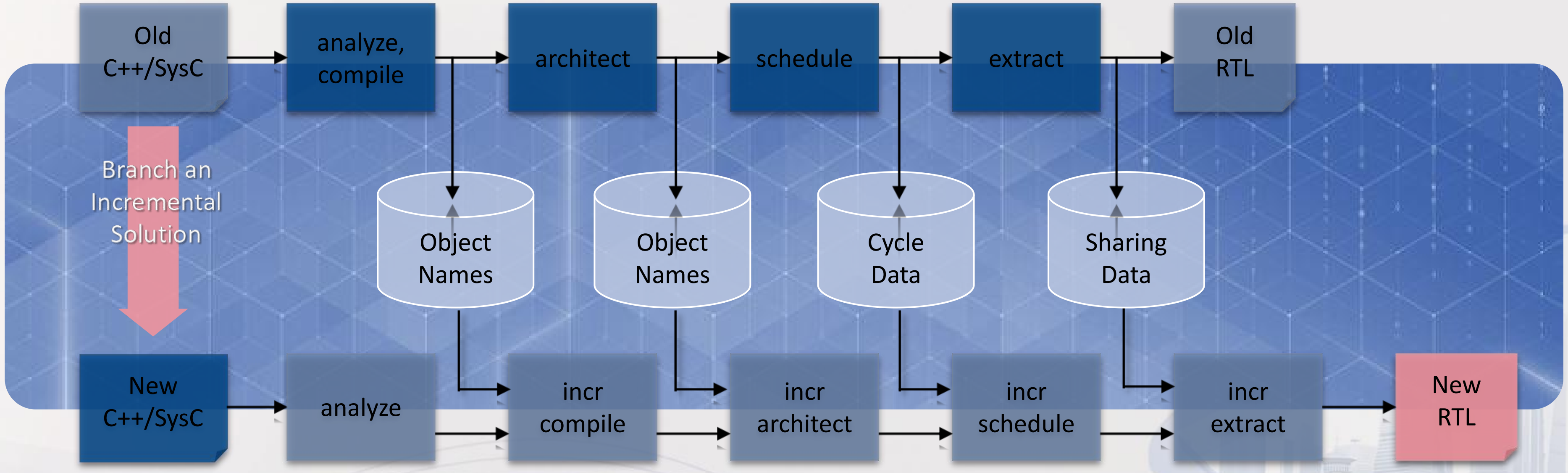


Incremental Synthesis & ECO Flow

Build an incremental solution from a reference solution

Apply reference data at appropriate HLS flow stages (automatic)

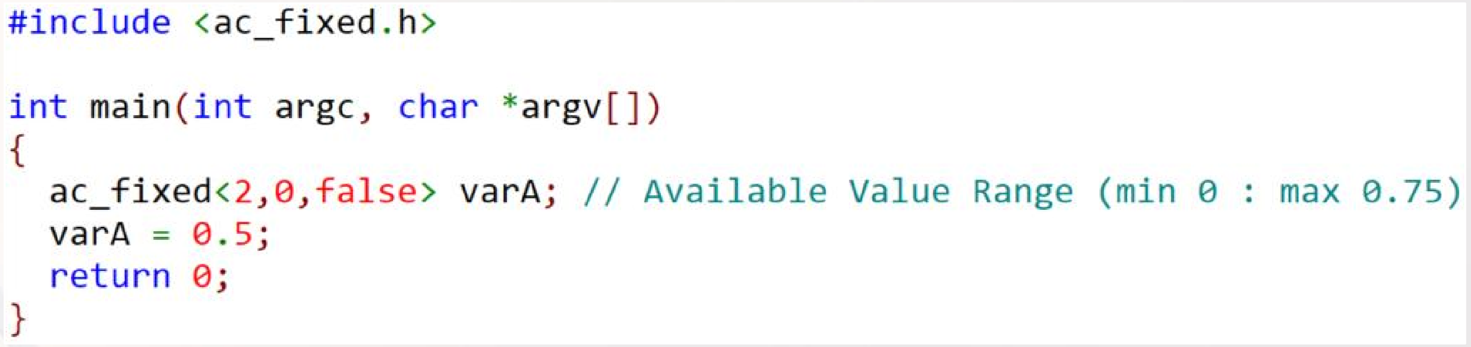
Continue with RTL ECO flow using new RTL



Dynamic Variable Range Analysis

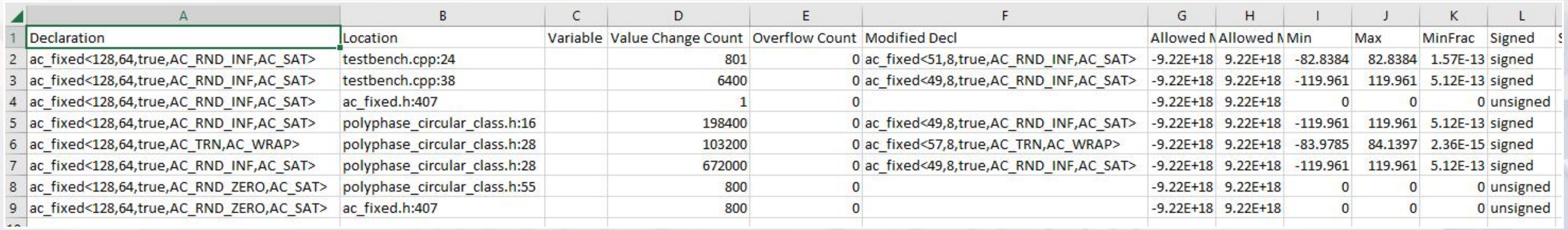
Quantization is a process to find the appropriate data accuracy, still satisfying the algorithm performance

AC datatype (ac_fixed) supports dynamic (simulation based) value range analysis

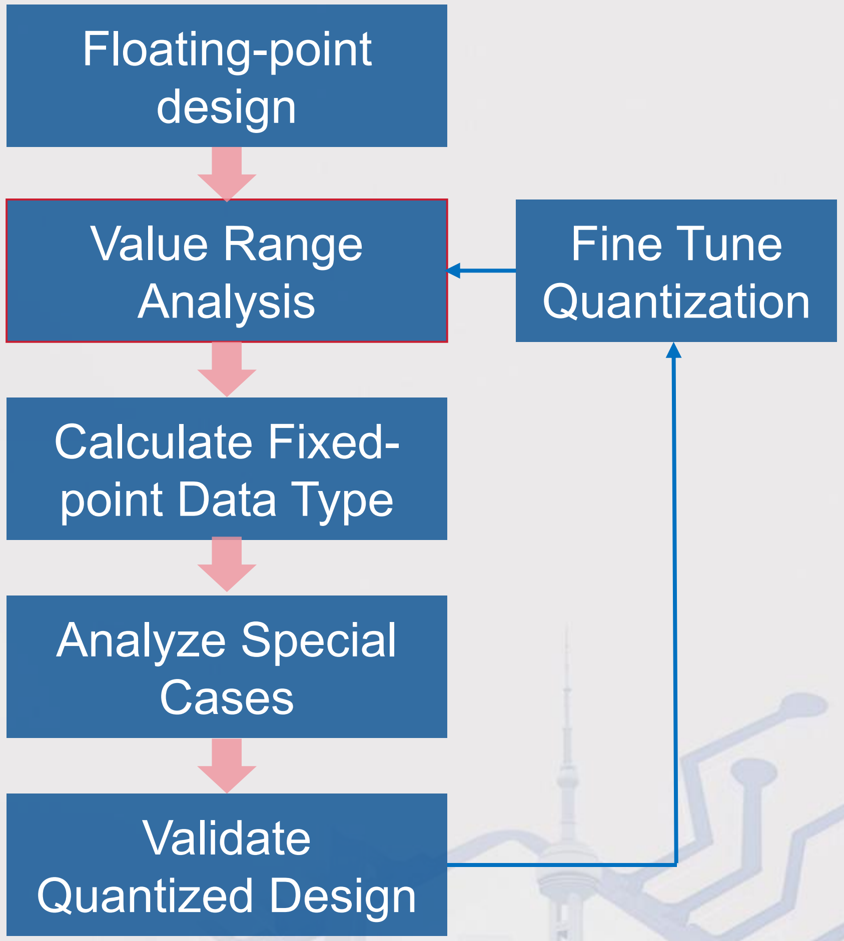


```
#include <ac_fixed.h>

int main(int argc, char *argv[])
{
  ac_fixed<2,0,false> varA; // Available Value Range (min 0 : max 0.75)
  varA = 0.5;
  return 0;
}
```



	A	B	C	D	E	F	G	H	I	J	K	L
1	Declaration	Location	Variable	Value Change Count	Overflow Count	Modified Decl	Allowed M	Allowed M	Min	Max	MinFrac	Signed
2	ac_fixed<128,64,true,AC_RND_INF,AC_SAT>	testbench.cpp:24		801	0	ac_fixed<51,8,true,AC_RND_INF,AC_SAT>	-9.22E+18	9.22E+18	-82.8384	82.8384	1.57E-13	signed
3	ac_fixed<128,64,true,AC_RND_INF,AC_SAT>	testbench.cpp:38		6400	0	ac_fixed<49,8,true,AC_RND_INF,AC_SAT>	-9.22E+18	9.22E+18	-119.961	119.961	5.12E-13	signed
4	ac_fixed<128,64,true,AC_RND_INF,AC_SAT>	ac_fixed.h:407		1	0		-9.22E+18	9.22E+18	0	0	0	unsigned
5	ac_fixed<128,64,true,AC_RND_INF,AC_SAT>	polyphase_circular_class.h:16		198400	0	ac_fixed<49,8,true,AC_RND_INF,AC_SAT>	-9.22E+18	9.22E+18	-119.961	119.961	5.12E-13	signed
6	ac_fixed<128,64,true,AC_TRN,AC_WRAP>	polyphase_circular_class.h:28		103200	0	ac_fixed<57,8,true,AC_TRN,AC_WRAP>	-9.22E+18	9.22E+18	-83.9785	84.1397	2.36E-15	signed
7	ac_fixed<128,64,true,AC_RND_INF,AC_SAT>	polyphase_circular_class.h:28		672000	0	ac_fixed<49,8,true,AC_RND_INF,AC_SAT>	-9.22E+18	9.22E+18	-119.961	119.961	5.12E-13	signed
8	ac_fixed<128,64,true,AC_RND_ZERO,AC_SAT>	polyphase_circular_class.h:55		800	0		-9.22E+18	9.22E+18	0	0	0	unsigned
9	ac_fixed<128,64,true,AC_RND_ZERO,AC_SAT>	ac_fixed.h:407		800	0		-9.22E+18	9.22E+18	0	0	0	unsigned



NVIDIA Research
- Catapult HLS Key to Optimize AI Inferencing for Performance/Watt

AI/ML Inference SoC implemented entirely in SystemC with HLS and Catapult

Enabled full SoC-level performance verification

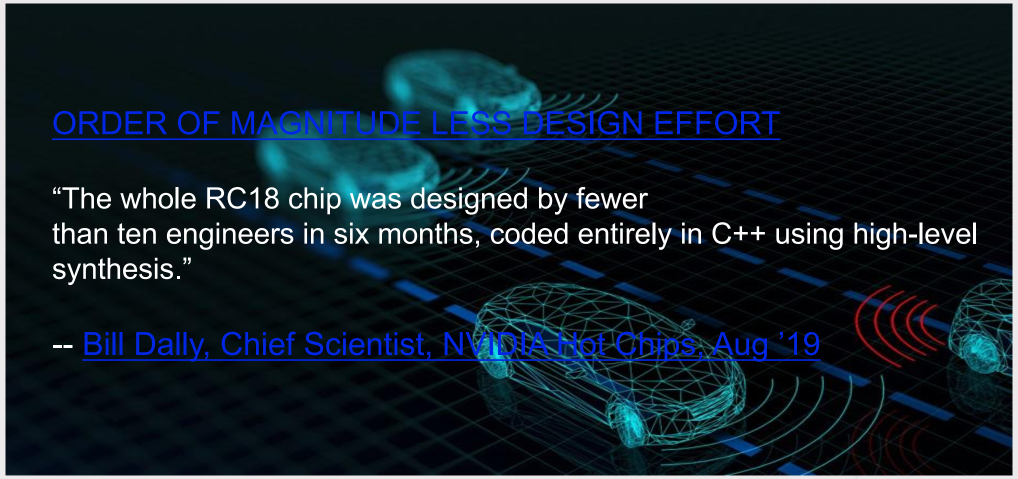
- 30X RTL, <2.6% difference from RTL in cycle count

Performance/Power and hits the mark

- 9.5 TOPS/watt in vanilla TSMC 16nm
- Scales to 128TOPS

10X Productivity over manual RTL

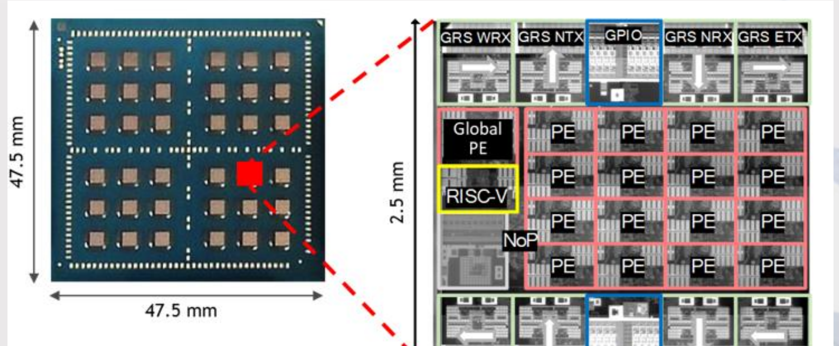
- Spec-to-Tapeout in 6 months with < 10 engineers



ORDER OF MAGNITUDE LESS DESIGN EFFORT

"The whole RC18 chip was designed by fewer than ten engineers in six months, coded entirely in C++ using high-level synthesis."

-- Bill Dally, Chief Scientist, NVIDIA Hot Chips, Aug '19



Core area	111.6 mm²
Voltage	0.52-1.1 V
Frequency	0.48-1.8 GHz

Catapult HLS – More Customer Success Stories

NVIDIA - Catapult HLS Key to Optimize AI Inferencing for Performance/Watt

SIEMENS
Ingenuity for life

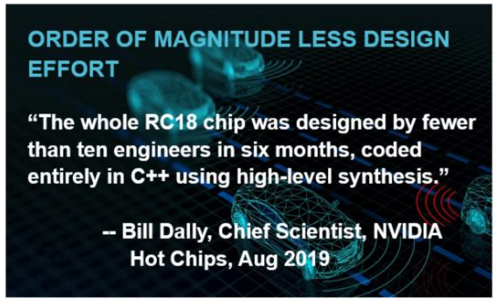
AI/ML Inference SoC implemented entirely with HLS and Catapult

Performance/Power hits the mark
- 9.5 TOPS/watt in vanilla TSMC 16nm
- Scales to 128TOPS

10X Productivity over manual RTL
- Design and Verification done in C++

Growth from 12-19 seats Catapult

Partnering with Mentor for productization
- Releasing libraries into open-source



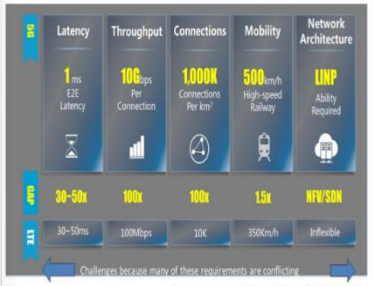
ORDER OF MAGNITUDE LESS DESIGN EFFORT

"The whole RC18 chip was designed by fewer than ten engineers in six months, coded entirely in C++ using high-level synthesis."

-- Bill Dally, Chief Scientist, NVIDIA Hot Chips, Aug 2019

Nokia - Catapult Helps Deliver 5G Faster to Multiple Carriers/Countries

SIEMENS
Ingenuity for life


5G
Latency
Throughput
Connections
Mobility
Network Architecture
1ms E2E Latency
10Gbps Per Connection
1,000K Connections Per km²
500km/h High-speed Railway
LINP Ability Required
GAP
30~50x
100x
100x
1.5x
NFV/SDN
LTE
30~50ms
100Mbps
10K
350Km/h
Inflexible
Challenges because many of these requirements are conflicting

Source: NGMN 5G White Paper

5G is easily 10X the complexity of 4G

Specifications is still changing – Huge race to win!

Trained over 150 Nokia engineers in HLS all over the world

HLS enabled them to create **new derivative FPGA designs** – Finland to China, US, Asia – MUCH Faster than RTL

ASIC is coming next and FPGA HLS will be re-used

Resulting in **$9-11.5M TCV for Catapult with ~2X growth**

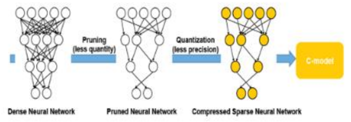
Interlock with TSS and Division key to growth

Chips and Media for AI - First Time for AI Design and HLS

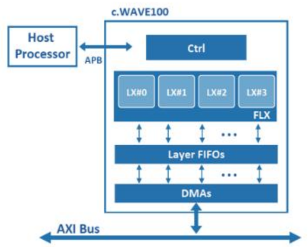
SIEMENS
Ingenuity for life

Successfully delivered Deep Learning IP on first project with HLS

Evaluation Process
- Side by side teams; one for HLS, one for RTL
- HLS team experienced designers but no HLS experience
- **Catapult Korea AE support throughout was key partnered with senior designer**



Evaluation Results
- HLS cut the block/IP design and verification time in half
- Delivered critical FPGA customer demonstrator early
- HLS found optimal power/performance not possible with RTL
- RTL teams understand and are convinced of HLS



What's Next
- HLS will be used on all new CV/DNN IP going forward
- New IP with Deep Learning for 4K to 8K upscaling

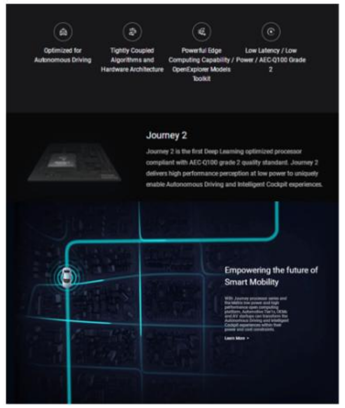
Horizon Robotics - Automotive-grade AI Processor

SIEMENS
Ingenuity for life

BPU AI Core Optimization With PowerPro, OFE by Catapult HLS

Evaluation Process
- Optical Flow Engine Design/Architecture Consultation
  - C++ Design 1st, System C Version Followed for System Validation Purpose
- BPU PowerPro RTL Power Optimization throughout Design Cycles

Evaluation Results
- OFE Design & Verification Time cut by Half
  - Enable Full System Validation based on SystemC TLM Methodology
- BPU Power Reduced by at least 10%
  - Easy Iterations of RTL Modification based on Accurate and Highly Implementable PowerPro Guidance
- Beat Stratus on SystemC HLS & PowerArtist on RTL Power in Single Campaign


Optimized for Autonomous Driving
Tightly Coupled Algorithms and Hardware Architecture
Powerful Edge Computing Capability / OpenExplorer Models Toolkit
Low Latency / Low Power / AEC-Q100 Grade 2
Learn More >

Journey 2

Journey 2 is the first Deep Learning optimized processor compliant with AEC-Q100 grade 2 quality standard. Journey 2 delivers high performance perception at low power to uniquely enable Autonomous Driving and Intelligent Cockpit experiences.

Empowering the future of Smart Mobility

Contact

Published by Siemens Digital Industries Software

Wenbo Zheng

Front End Design Solution Group

Beijing

China

Mobile +86 155 0102 7338

E-mail wenbo.zheng@siemens.com