Achieve software prototyping verification success with Veloce proFPGA CS

Marsheneil Koul
Field Application Engineer
Siemens EDA

Marsheneil Koul is a Field Application Engineer at Siemens EDA, where her technical focus is on FPGA prototyping. With close to a decade of experience in the semiconductor industry, Marsheneil has worked closely with engineering teams of leading design houses to optimize prototyping strategies, improve performance and accelerate time-to-silicon. Her role revolves around evaluating, deploying, and supporting advanced prototyping solutions across a broad range of customer environments, including but not limited to applications in AI, automotive and networking. Her close collaboration with R&D teams within Siemens helps to bridge the gap between tool development and real-world deployment.

At DVCon 2025, she will be presenting on "Achieve software prototyping verification success with Veloce proFPGA CS", bringing a practical, customer-focused and problem-solving perspective on real-world challenges faced by development teams and showing how the Veloce proFPGA CS addresses them.

Achieve software prototyping verification success with Veloce proFPGA CS

Marsheneil Koul
Field Application Engineer
Siemens EDA

WHY this matters?

Shrinking manufacturing technology	**Purpose-built SoCs and AI accelerator architectures**
Power and performance	**Software defines the product**

Massive software workloads
Complex design sizes
Competitive, fast-moving market

Need for **speed**
Need for **capacity**
Need for **productivity**

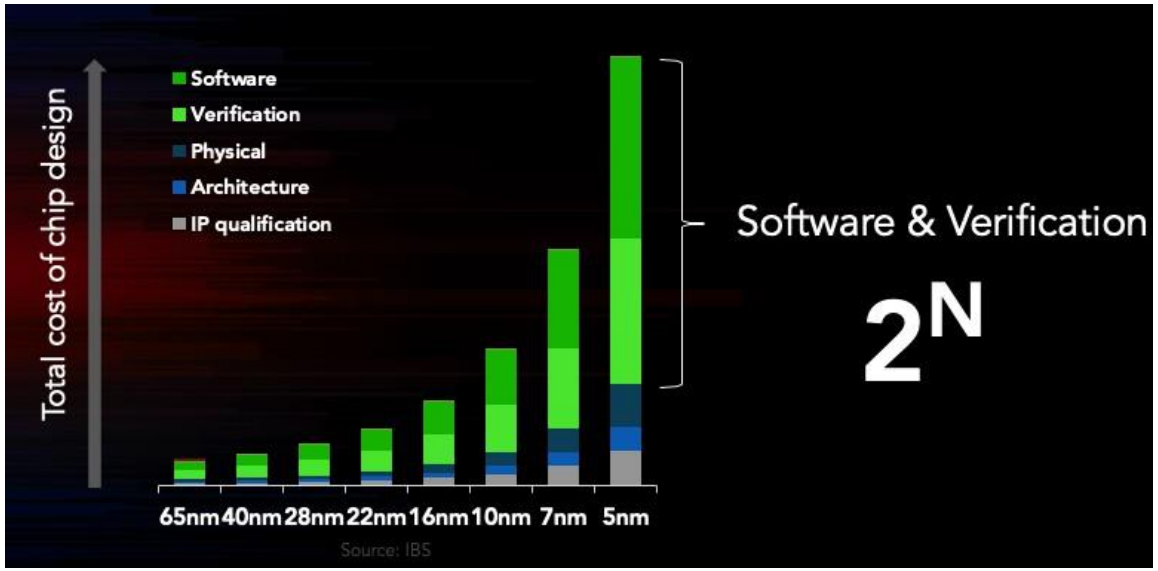
Emulation and prototyping are essential

SIEMENS

The path to product success leads through software

Hardware, the chip, is becoming a commodity

- Yes, re-spins are expensive
- Yes, implementation tools are expensive
- Yes, mask and fab cost are high



On the Other End

Software dominates design costs

- Software is a MUST HAVE to sell silicon
- Software is a differentiator
- Delays in software readiness costs money

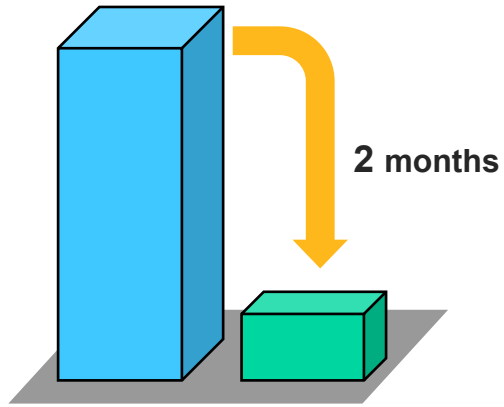
Software development & validation must start as early as possible!

SIEMENS

Enabling early Firmware & Software development and Validation

“Shift Left” resulting in faster ‘Time to Revenue’

- Let’s assume that your new product is expected to generate $50M per year (~$150k per day)
- Deploying a solution that is enabling to release your product 2 months early (or avoid being 2 months late)



$9M in additional revenue!

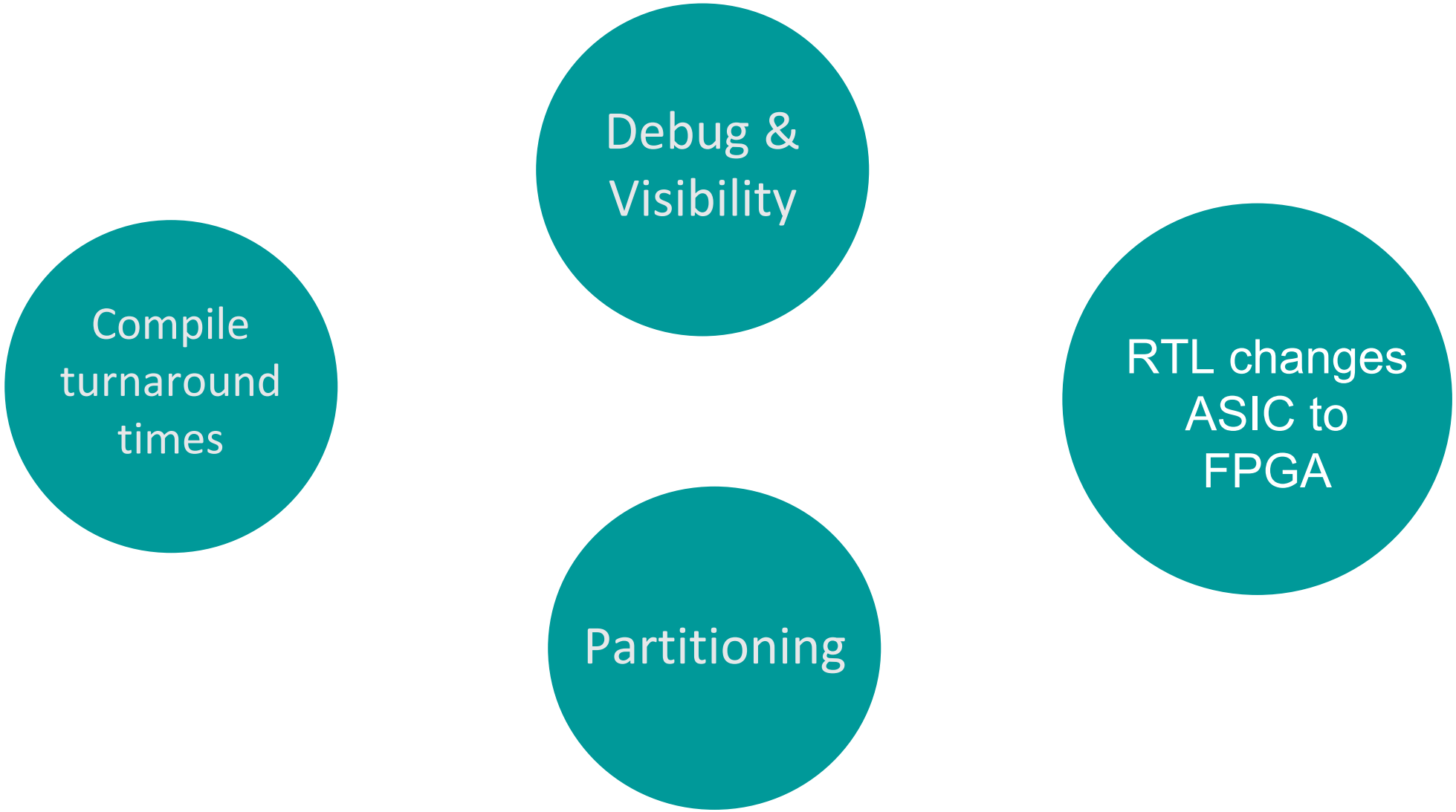
The key ingrediencies for early FW/SW validation

- **Performance** fast enough to boot firmware and operating system
- **Accuracy** cycle-accurate representation of the IP or chip hardware
- **Productivity** software-centric features like memory-backdoor, SW debug, etc.
- **Connectivity** real-world interfaces like PCIe, sensors, cameras, etc.

FPGA Prototyping: The insurance policy we all need

- Early Design Validation – Prevent system flaws early
- Avoid Costly Re-Spins – Save $$$ on mask sets
- Parallel Development – HW + SW in sync
- Accelerate Time-to-Market – Deliver faster

The Problem



The Solution

Partitioning

- **Automatic partitioning**
- **Hybrid/guided partitioning**
- **Netlist level partitioning (more precise resource calculation)**

The Solution

Debug & Visibility

- **Early probe (RTL) and Late probe (Netlist level) flows**
- **Probe more signals without worrying about resource constraints**
- **Full visibility debug**

The Solution

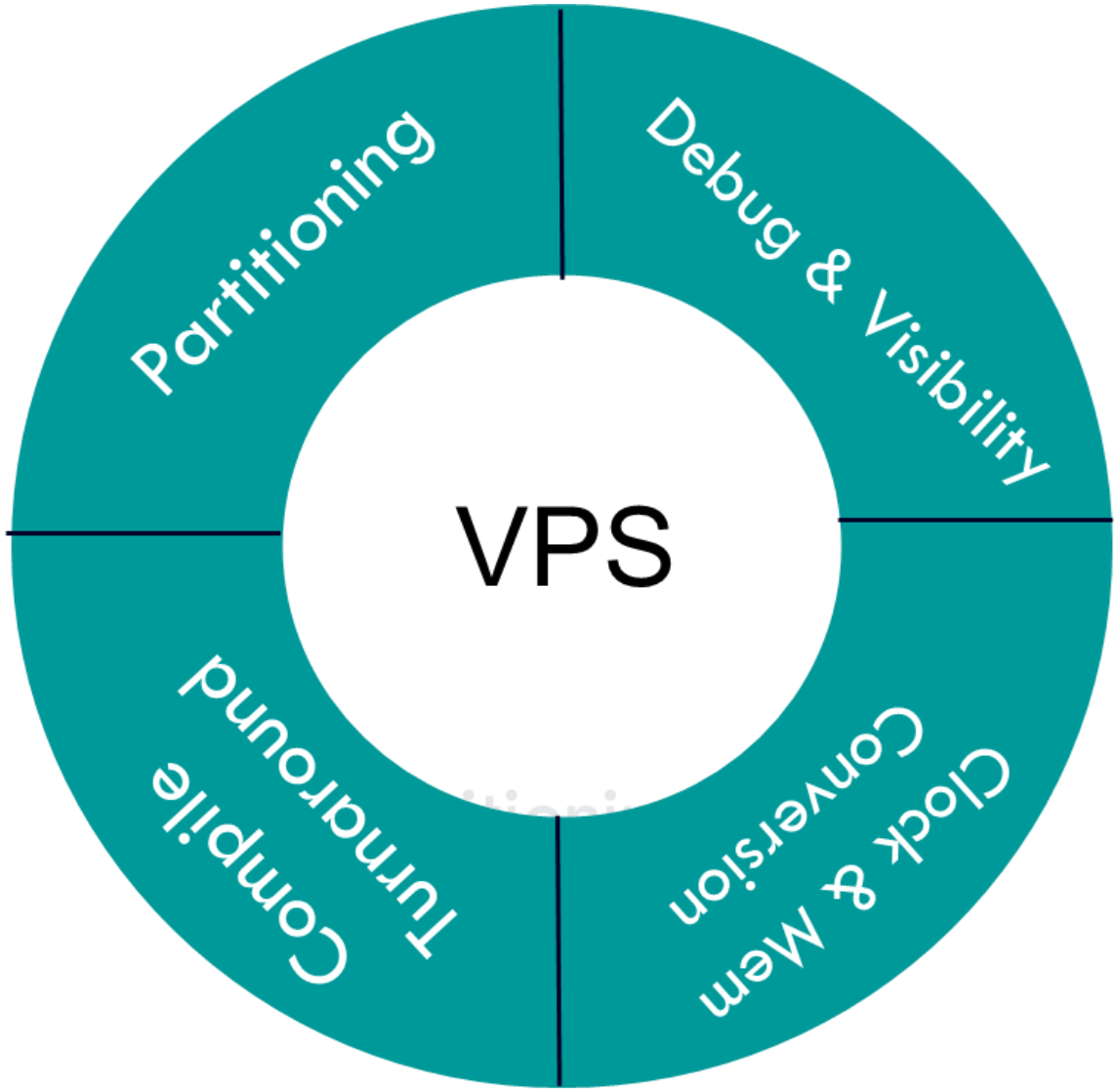
Compile turnaround times

- **Incremental Compile**
- **Late probe insertion**
- **Reusing partitioning results**

The Solution

RTL changes
ASIC to FPGA

Automatic conversion from ASIC to FPGA friendly clocks and memories



VeloceOS for prototyping (VPS) SW accelerates design bring-up

Automatic prototype build without manual RTL modifications

- DSP and Memory inference as FPGA primitives
- Automated handling of multi driver, XMR and tristate
- Gated and generated clock conversion
- Logic analysis and optimization

Automated partitioning

- Optimized cuts between FPGAs
- Respect user constraints (filling rate, grouping, flattening)
- FPGA and die level partitioning

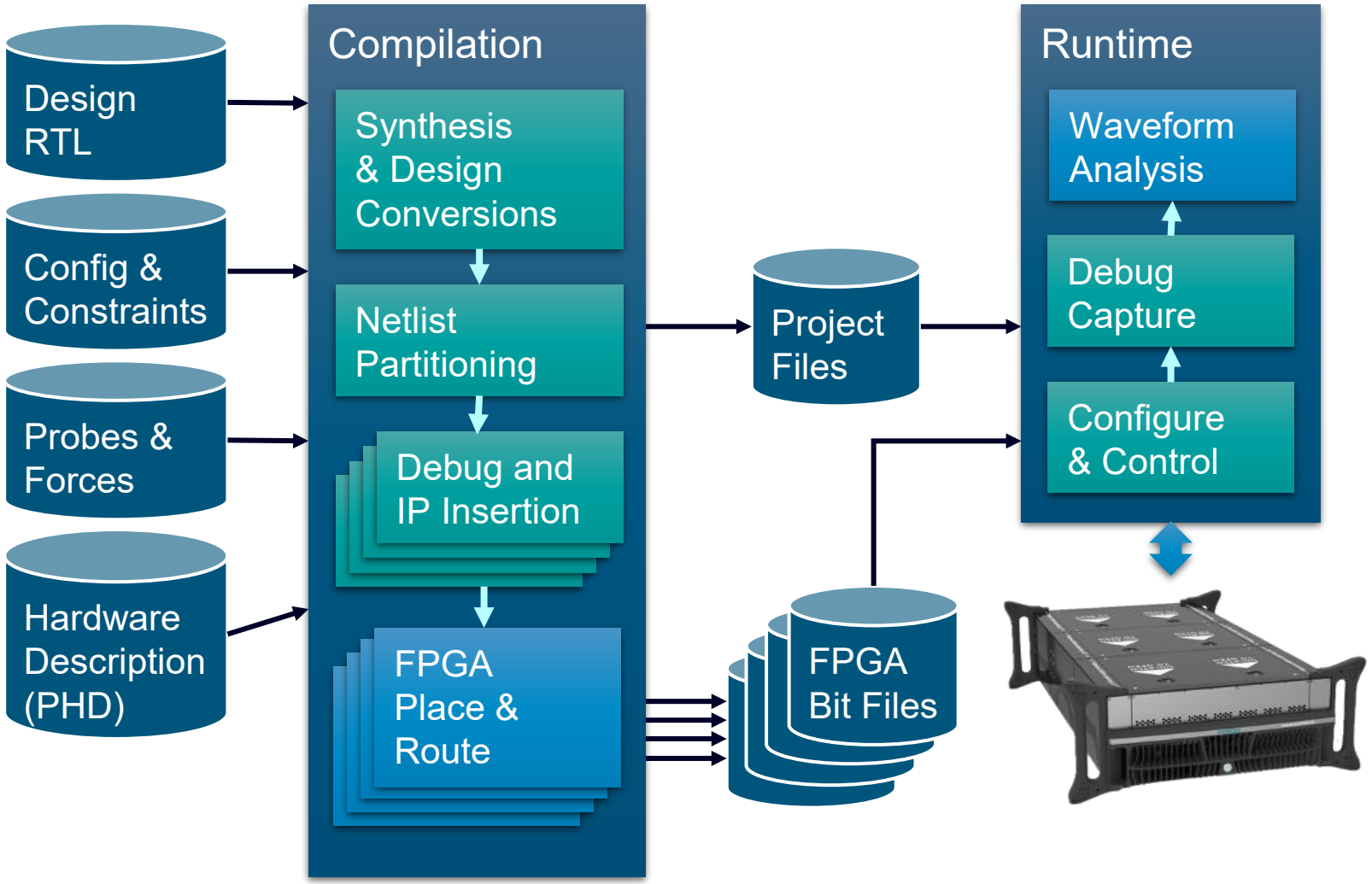
Timing-driven for best performance

- Optimized signals routing between FPGAs
- Clock frequency calculation
- Mux IP insertion and wireset management

Test drive

SIEMENS

VPS (Veloce Prototyping Software): compilation flow



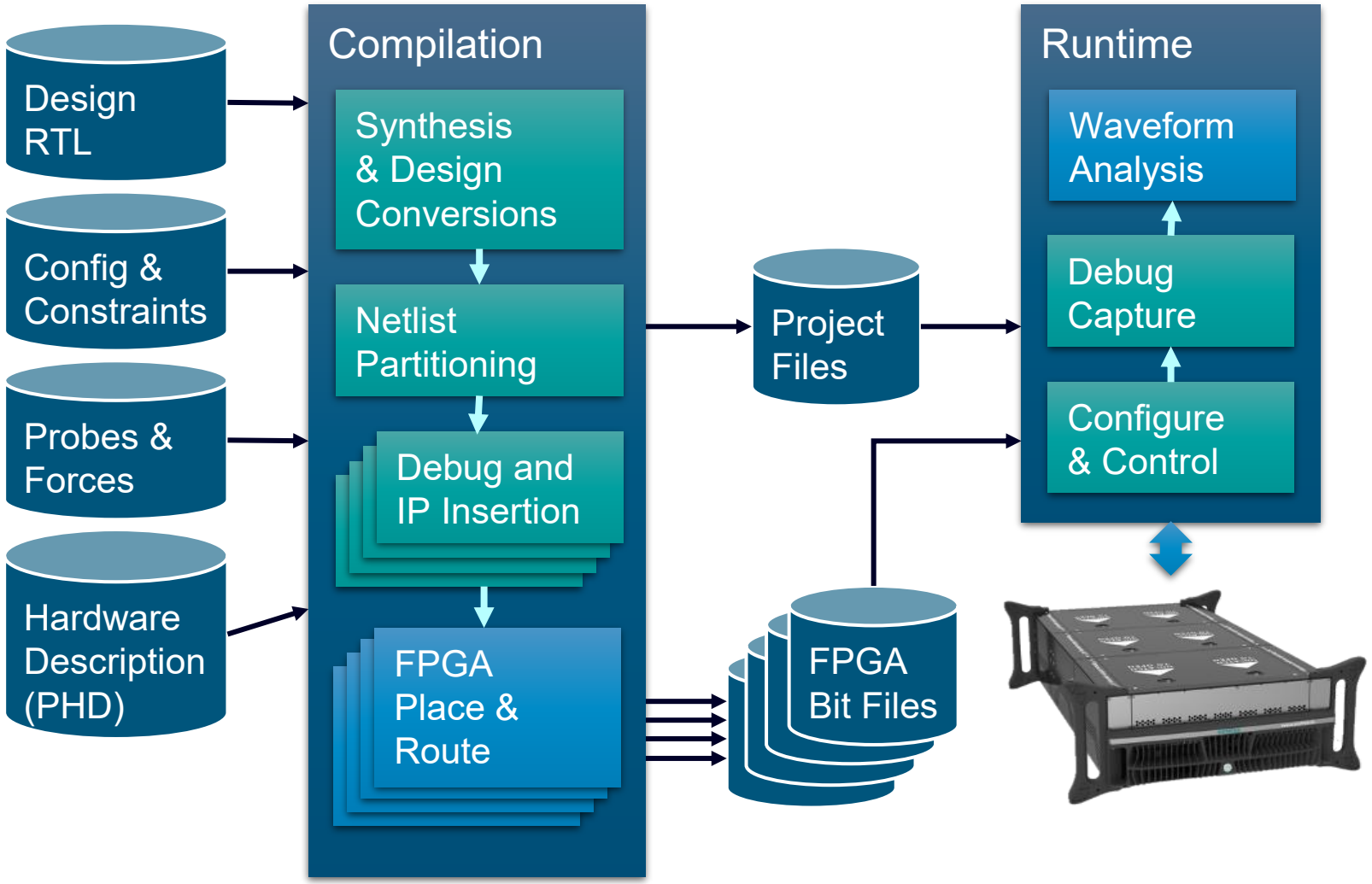
Compilation

- Design is synthesized for FPGA
- ASIC constructs are remodeled in FPGA friendly way, as needed
- Netlist is partitioned into FPGAs
- Debug and other IP instantiated into post-partition netlist
- Control & management of Vivado place and route tool

Interface

- TCL based configuration files
- GUI design and netlist visualization
- Live compile cockpit (Build Assistant)

VPS (Veloce Prototyping Software): runtime



Runtime

- Bit files are loaded onto target HW
- Memories loaded
- Clocks and reset are configured
- Debug capture is configured
- Tests/payload is run
- Waveforms are captured and analyzed

Interface

- Tcl or C-API for debug, runtime and testbench control
- GUI for debug control
- GUI for design and waveform viewing

Synthesis and design conversions

- Synthesizes the design
- Netlist flattening and design optimization
- Output netlist of FPGA primitives
- Preserved debug probes and constraint mentions
- Identifies memories, DSPs and multiple drivers for downflow processing

- Additional design optimization
- Constant value propagation
- Dead logic removal
- Multi-driver resolution

- Clock and reset analysis
- Gated and generated clock conversion
- Probe clock association
- ASIC multi port memory conversion to FPGA friendly memories

Incremental Compile: frontend

No optimization Mode

- Fastest
- Least optimizations

Aggressive optimization Mode

- Capacity, performance or congestion mode optimizations
- Can be global or per hierarchy

Guided optimization Mode

- Based on information feedback
- Limits optimization for modified RTL

VPS partitioning methodology

VPS partitions the design

Parallel trials, different seeds, fill-rate exploration, hierarchical exploration

Autoreplication and Autocabling usually utilized in this mode

Full control over partitioning

All user directives will be honored

User design knowledge applied to coarse-grain partitioning & VPS partitions within those constraint limits

User design knowledge & information from VPS used to further optimize prototype

Incremental Compile: partitioning and cabling

Reproducing partitioning solution

Partitioning can be nondeterministic, but there is a way to preserve a golden partitioning solution from previous runs. This can save both exploration effort and compile time. All that is needed is to source the output tcl from a previous run

Reproducing cabling solution

Cabling exploration can also take time and several iterations. Each cabling iteration output can be sourced as input to a new compile to use it as a starting point and build additional cabling on top of it

Debug and IP Insertion

Time it up!!! Last minute edits!!! It's a wrap!!!

- Calculates inter-FPGA signal routing
- Bundles signals into wiresets
- Calculates clock speeds

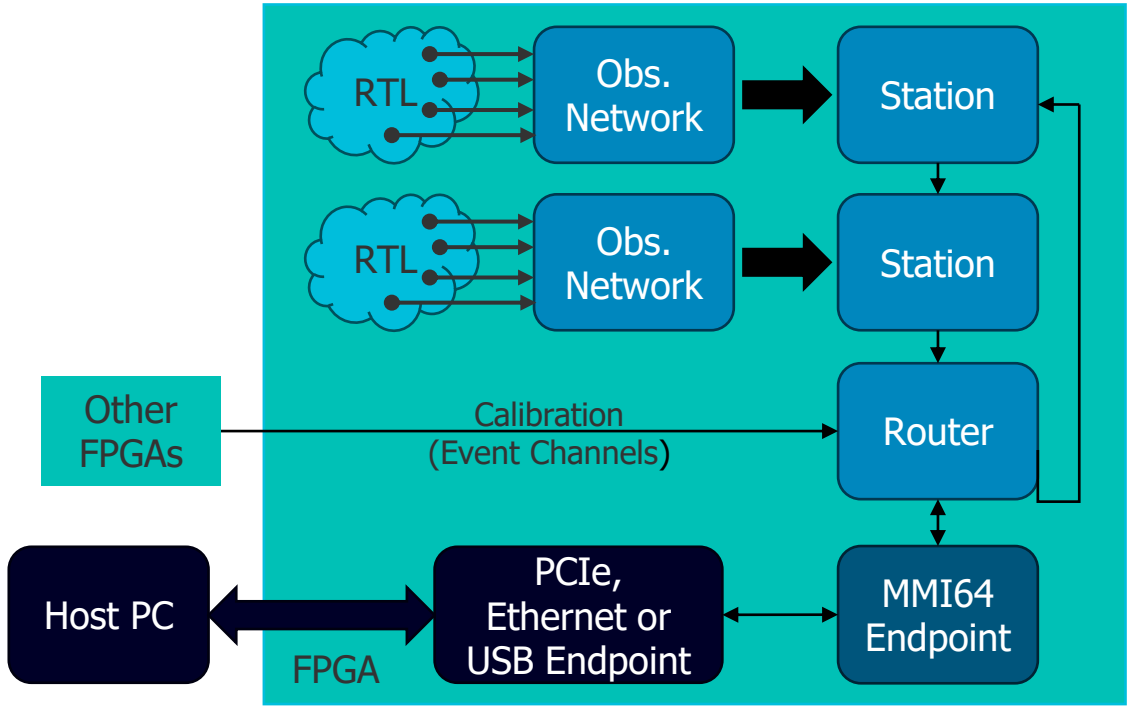
- Makes modifications to individual FPGA partitions
- Late stage probe insertion
- Late stage netlist edits (ECOs)

- Generates top level wrapper
- Performs debug insertion
- Instantiates
  - Design partition
  - Wire sharing IP
  - Clock/reset generators
  - Softmodels
  - External interfaces
  - Inferred memories
- Generates for backend (Vivado)
  - Projects
  - Constraints

Advanced Probe-based debug

Probed Debugging

- Signals referred to by original RTL names
- Probed signals traced at full speed
- Runtime choice to change traced signals
- Arming triggers with series of conditions
- Early and late probes flow available at RTL and netlist levels respectively



SIEMENS

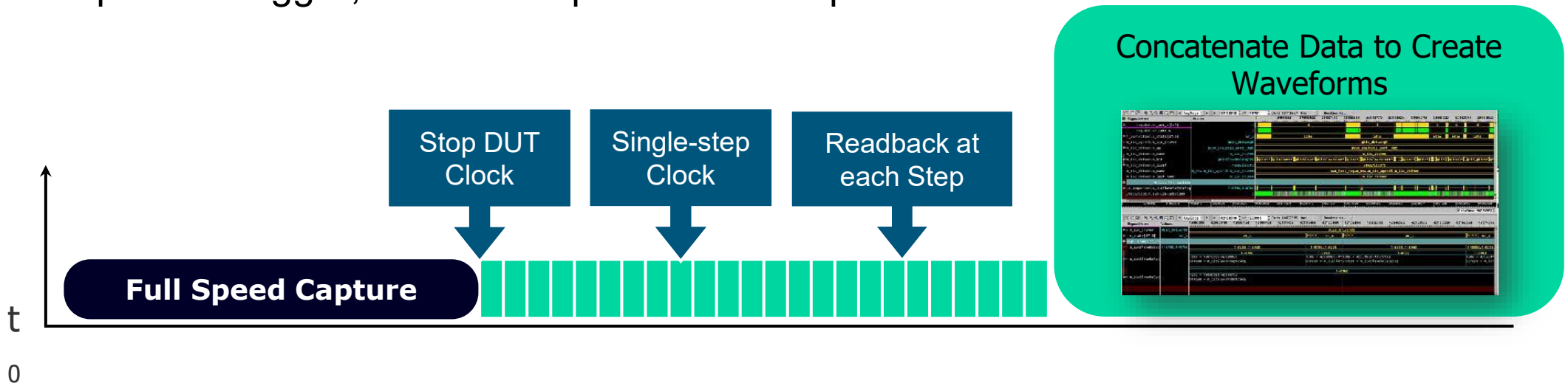
Emulation-like debug productivity in VPS

Probeless Debugging

- Full-State capture at slower speeds using readback
- Full-Visibility for combinational logic using reconstruction

Hardware Breakpoint

- Full-speed to trigger, then slow-speed for full capture

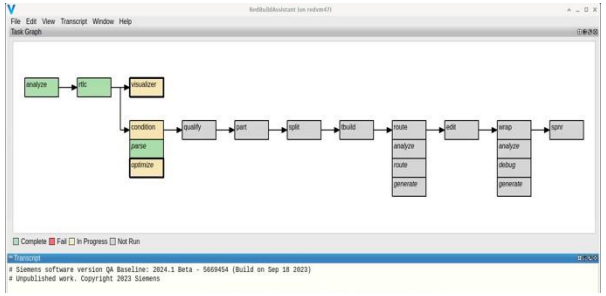


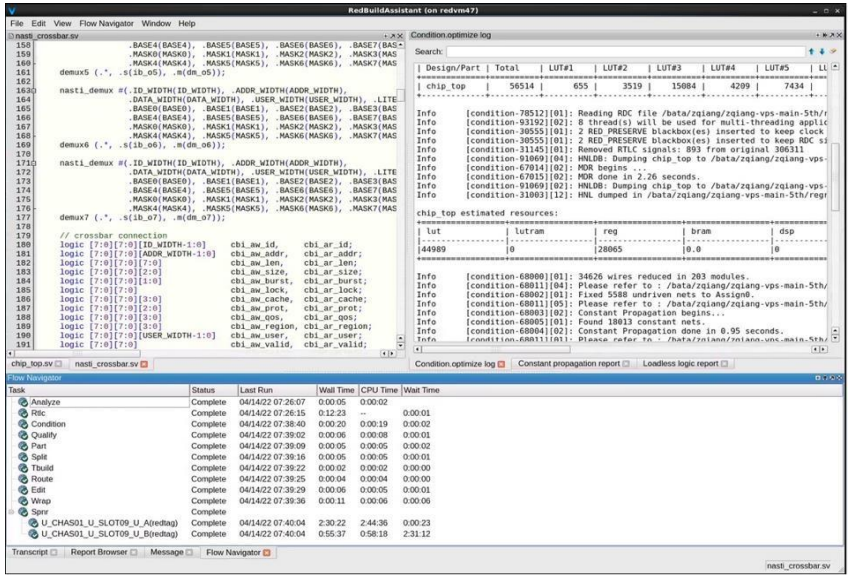
SIEMENS

Useful Interface for efficient prototyping compile & debug

Build Assistant

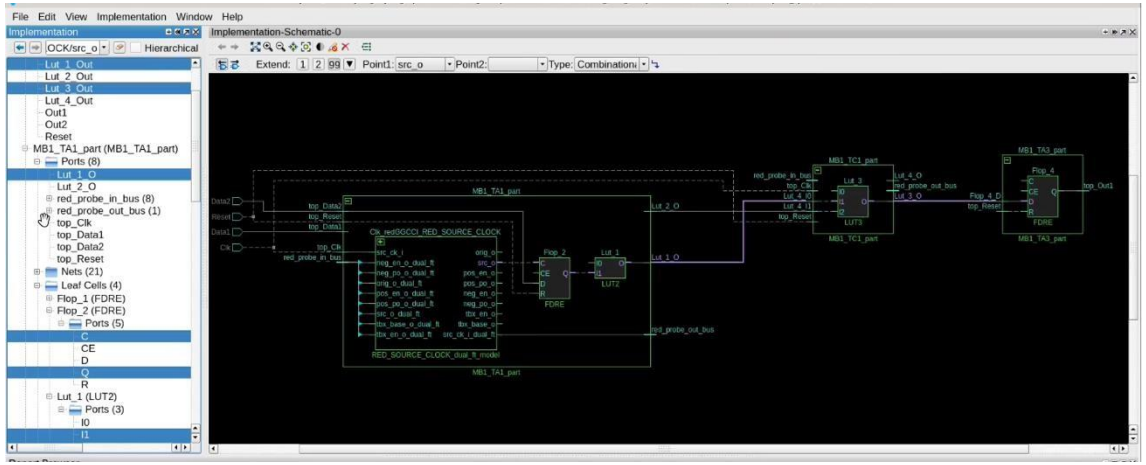
- Review compilation steps
- Report browser
- Compile checker
- Visualize task status in real time





Review FPGA implementation

- Timing Analysis
- Dead logic annotation
- Probe selection
- Waveform side by side with implementation netlist & schematics
- Combi loop & Logic depth widgets



SIEMENS

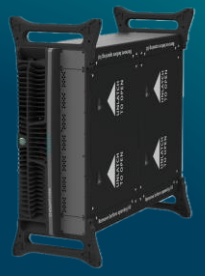
Veloce proFPGA CS: Single FPGA desktop to large multi-blade system

UNO Desktop

Single FPGA board

Up to 400MHz

QUAD Desktop HEXA Desktop



Multi-FPGA designs

Multi-user setups

HEXA Blade

Data-center standard

Remote management

Multi-blade System

Billions gates designs

Enterprise solution

Same functionalities across platforms

Advanced clock management	FPGA Granularity	Ease of use SW
Runtime control via USB, Ethernet, PCIe	Manual or Automated compile	At speed target
Best Performances	No HW development costs	100% Flexible cabling

SIEMENS

Customer use cases: Our platform with AMD's breakthroughs

Case Studies > Siemens Digital Industries Software

Siemens Veloce proFPGA Case Study

Siemens' Veloce proFPGA CS for software prototyping – Enabled by AMD Versal™ Premium Adaptive SoC

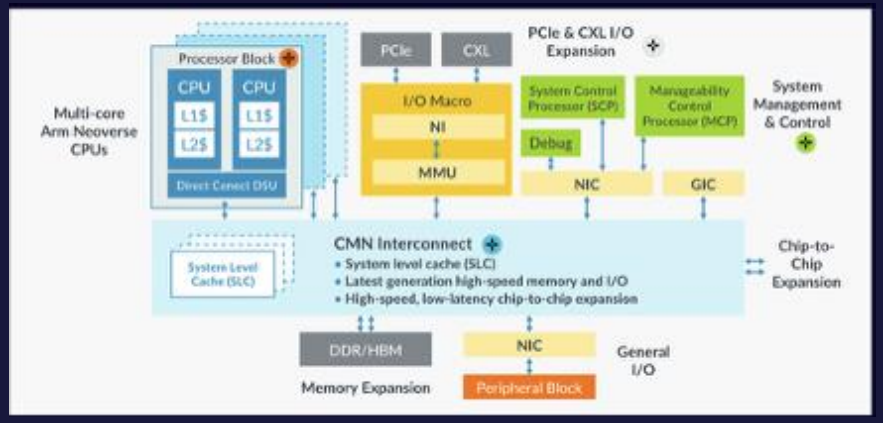
Siemens' Veloce proFPGA CS is a prototyping system to quickly perform pre-silicon validation of ICs and software, enabled by an AMD Versal™ Premium adaptive SoC.

Feedback

SIEMENS

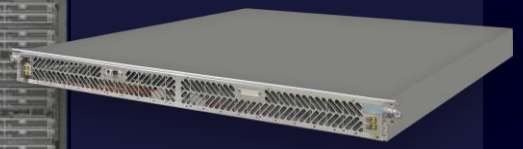
Press release for July 14, 2025

Veloce CS used by Arm for Neoverse CSS verification



Veloce Strato CS
4 Towers, 11 BG

Veloce Strato CS blade



Veloce proFPGA CS blade

Veloce proFPGA CS
10 blades

Veloce CS enables faster time-to-market for Arm's partner ecosystem through pre-validation and verification

Veloce Strato CS

- High-performance emulation scaling from 40MG to 48BG
- Full visibility debug capability
- PCIe composite device for Arm Compliance Suite integration

Veloce proFPGA CS

- Fast software prototyping scaling from single to hundreds of FPGAs
- Accelerated firmware, OS, and application development

"A core component of Arm Neoverse CSS is the pre-validation and verification, made possible by adopting innovative new tools like Siemens Veloce CS system, so that our partners can get their silicon solutions to market faster."

Karima Dridi, Head of Productivity Engineering, Arm

SIEMENS

Ericsson case study

Take away

✓ Homebrew vs Commercial solution

✓ Small design size (<4FPGAs)

Performance vs Time to prototype

✓ When design size increase -> no other options

Thank You

Marsheneil Koul
Field Application Engineer
Hardware Assisted Verification (HAV)
E-mail : Marsheneil.koul@siemens.com