2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE 10 YEAR ANNIVERSARY

Closed-Loop Model-First SoC Development With the Intel® Simics® Simulator

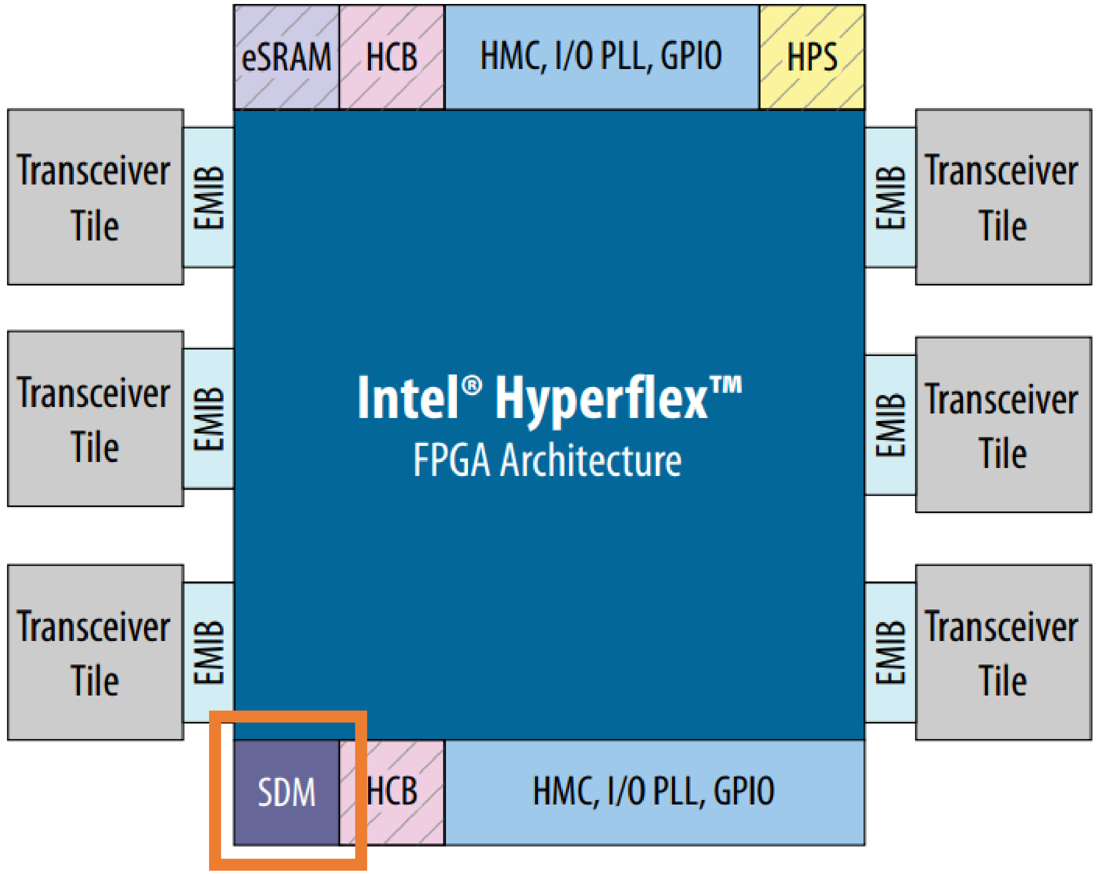
Kalen Brunham, Anthony Moore, Tobias Rozario, Wei Jun Yeap, and Jakob Engblom, Intel Corporation

intel®

accellera SYSTEMS INITIATIVE

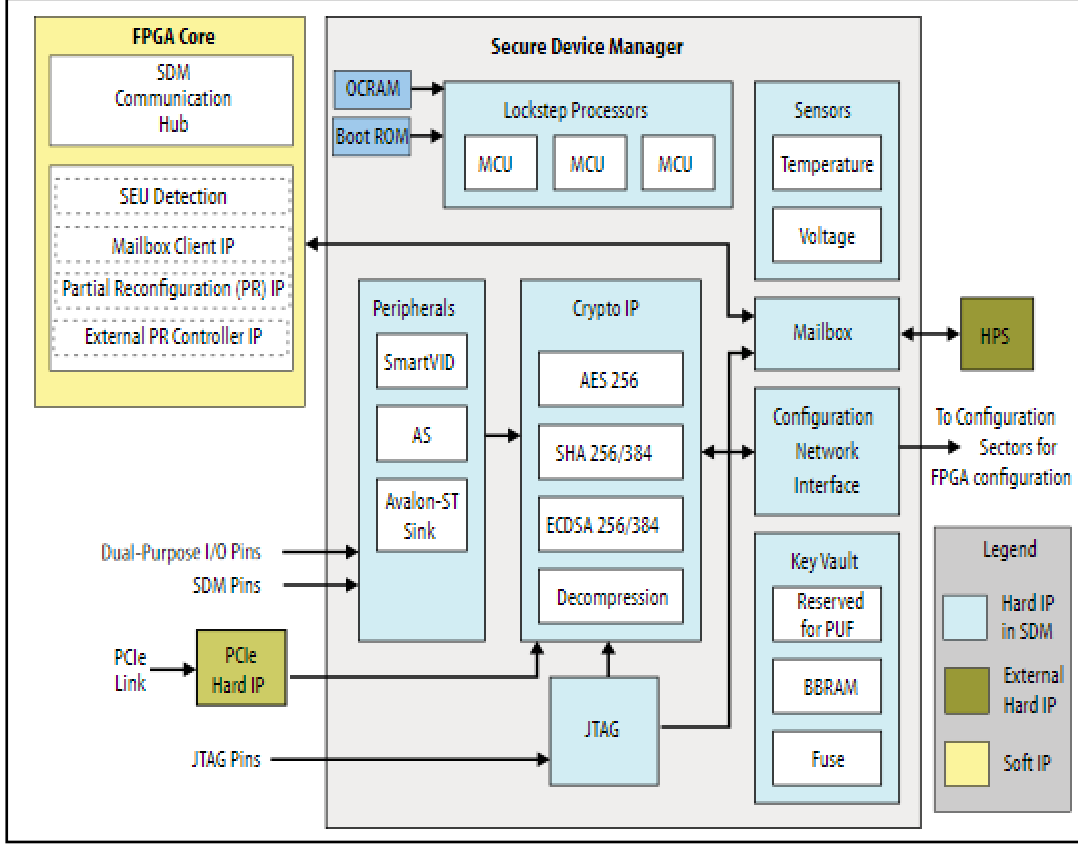
Let’s Start a New Project

- Hypothetically… Let’s create a next gen FPGA!



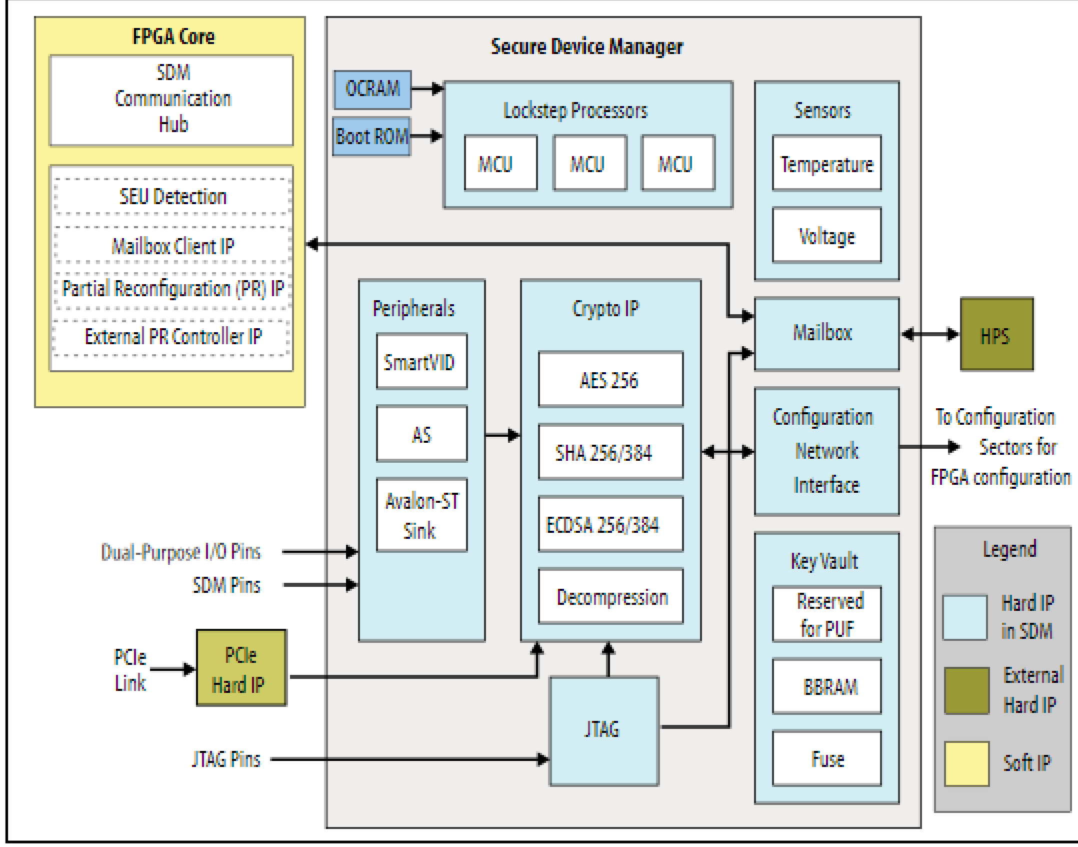
Let's Start a New Project

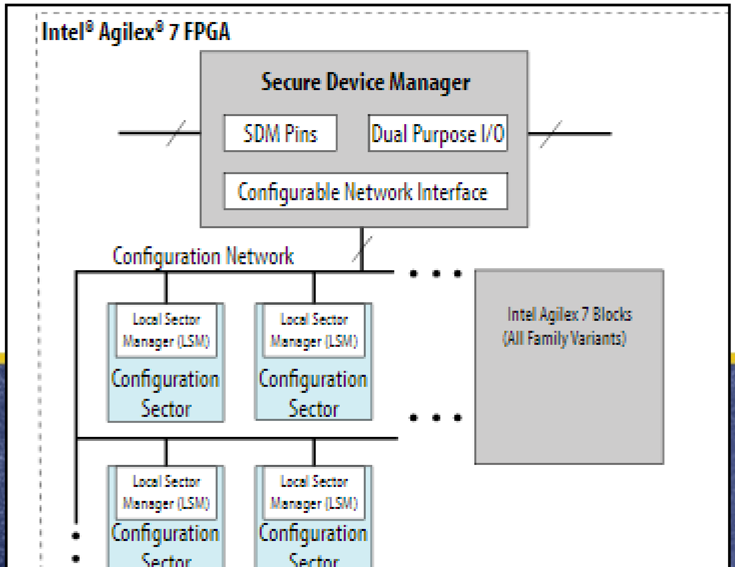
- Let's use processors to manage the configuration of the device
- This new system uses a combination of 3rd party IP and custom IP
  - A 3rd party NoC is used to connect the peripherals and enforce access



Let’s Start a New Project

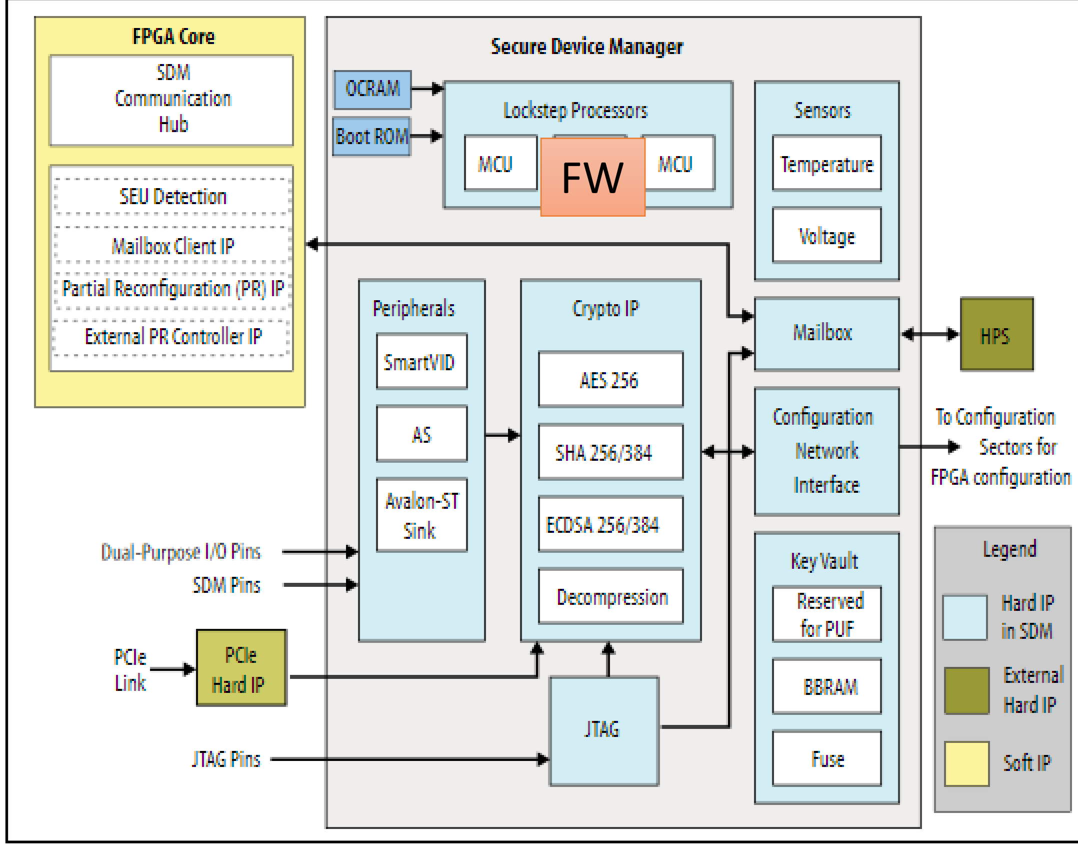
- Let’s use processors to manage the configuration of the device
- This new system uses a combination of 3rd party IP and custom IP
  - A 3rd party NoC is used to connect the peripherals and enforce access
- The system has ~100 distributed processors all talking to each other

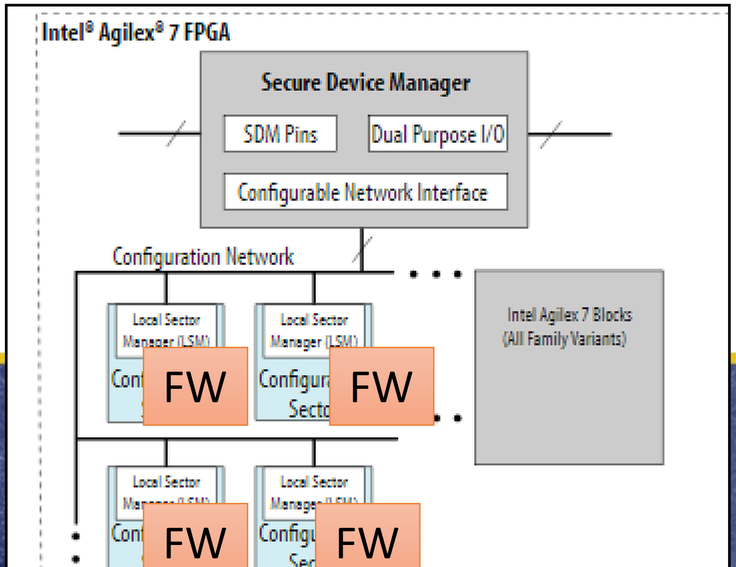



Intel® Agilex® 7 FPGA
Secure Device Manager
SDM Pins
Dual Purpose I/O
Configurable Network Interface
Configuration Network
Local Sector Manager (LSM)
Configuration Sector
Local Sector Manager (LSM)
Configuration Sector
Intel Agilex 7 Blocks (All Family Variants)
Local Sector Manager (LSM)
Configuration Sector
Local Sector Manager (LSM)
Configuration Sector

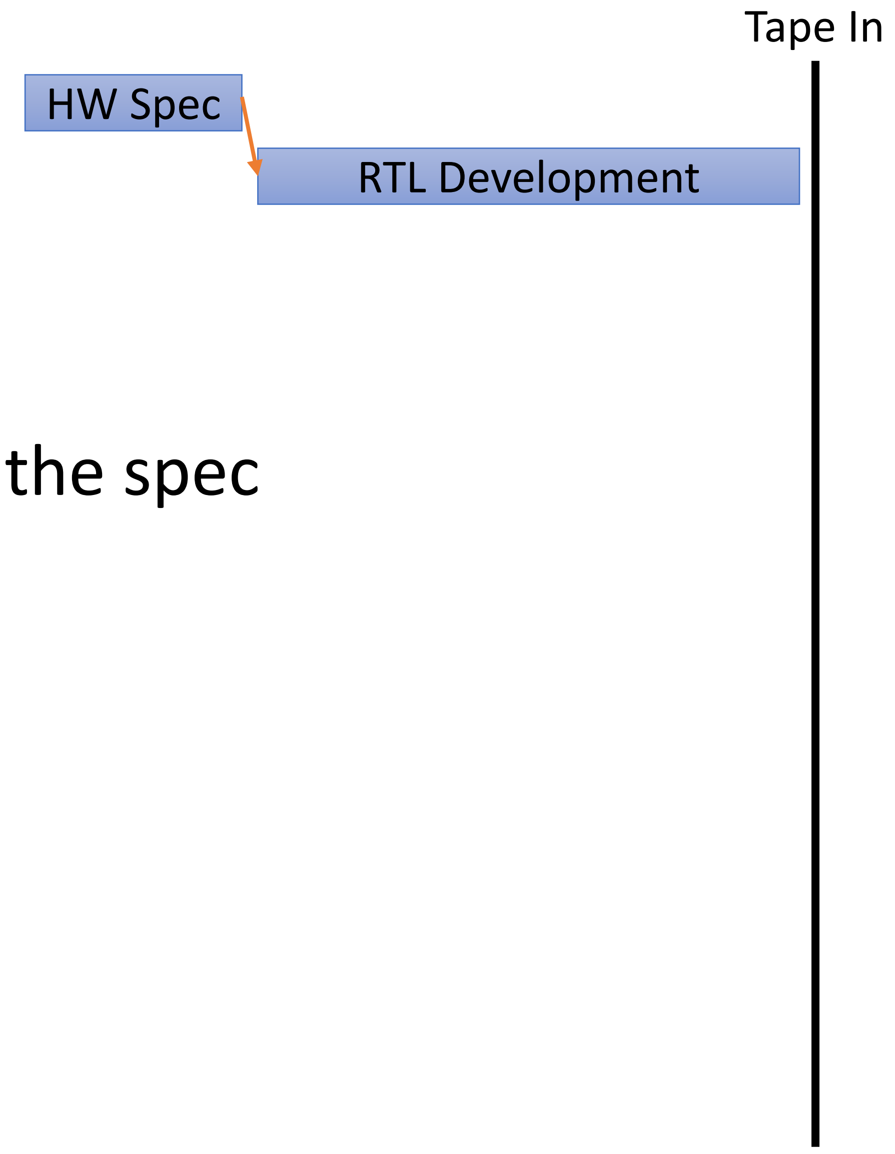
Let’s Start a New Project

- Let’s use processors to manage the configuration of the device
- This new system uses a combination of 3rd party IP and custom IP
  - A 3rd party NoC is used to connect the peripherals and enforce access
- The system has ~100 distributed processors all talking to each other
  - These processors all run FW that is needed to power-on the device





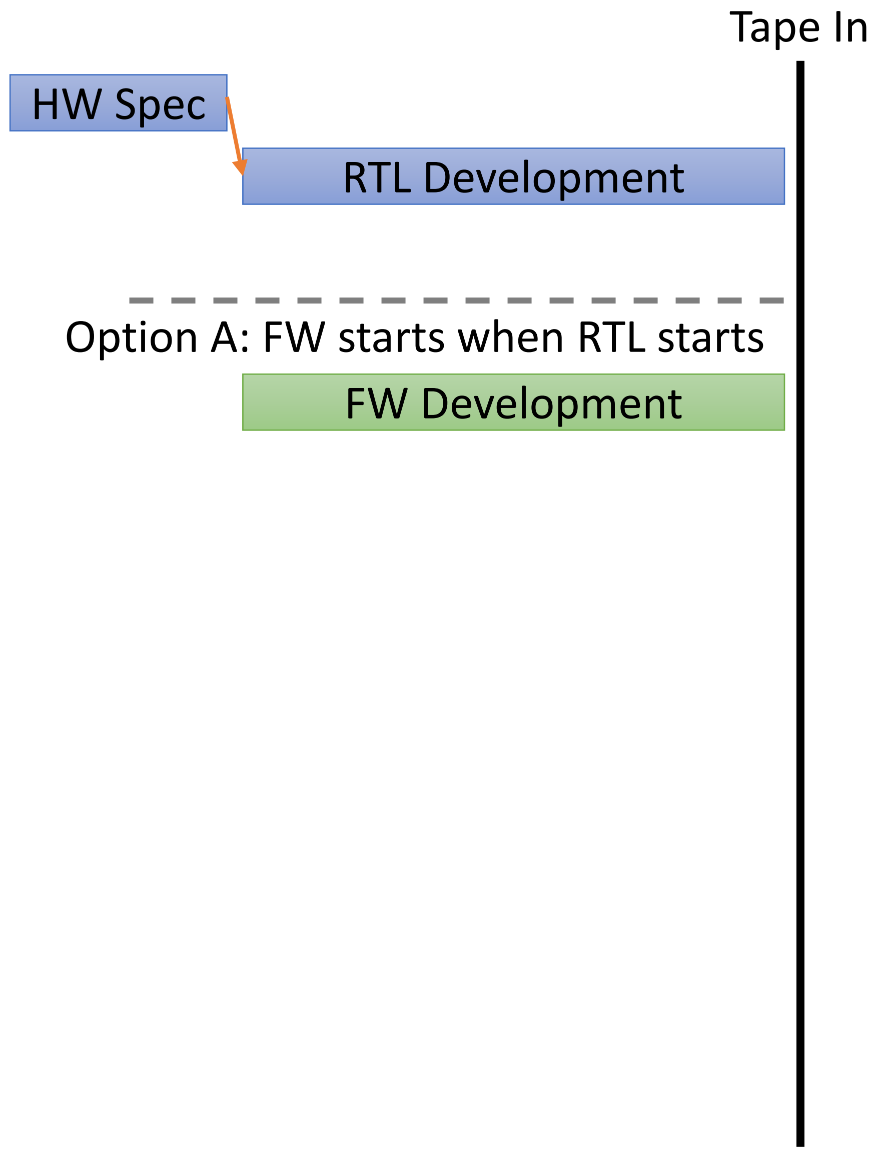
When to Start?



- HW team needs the spec to write the RTL
- DV team needs the RTL to validate it against the spec

When to Start Software?

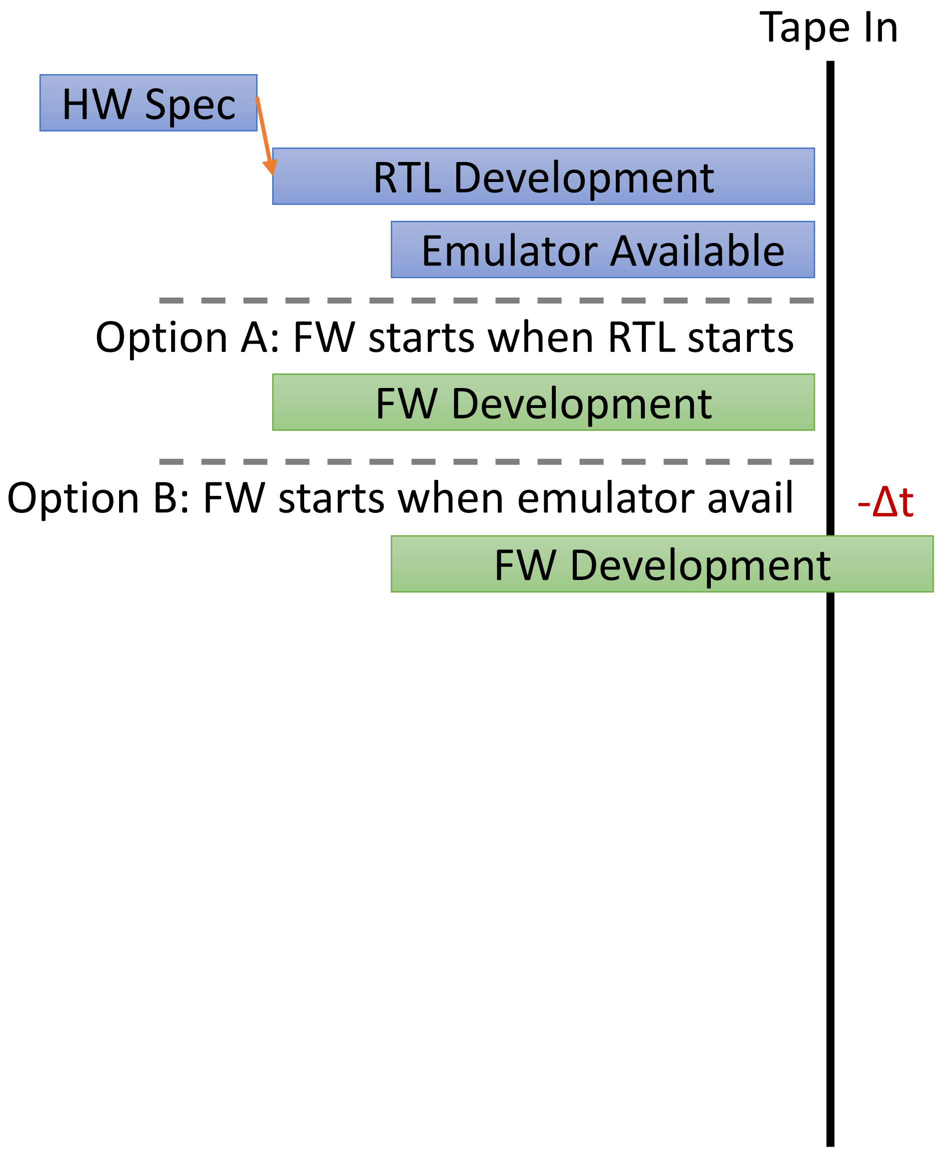
- Developing the SW for an SoC takes a defined amount of time
  A. Without any pre-silicon vehicle SW development could start when RTL starts but has no guarantee to work on HW



Some pre-silicon system or model is needed to ensure the HW and SW works together

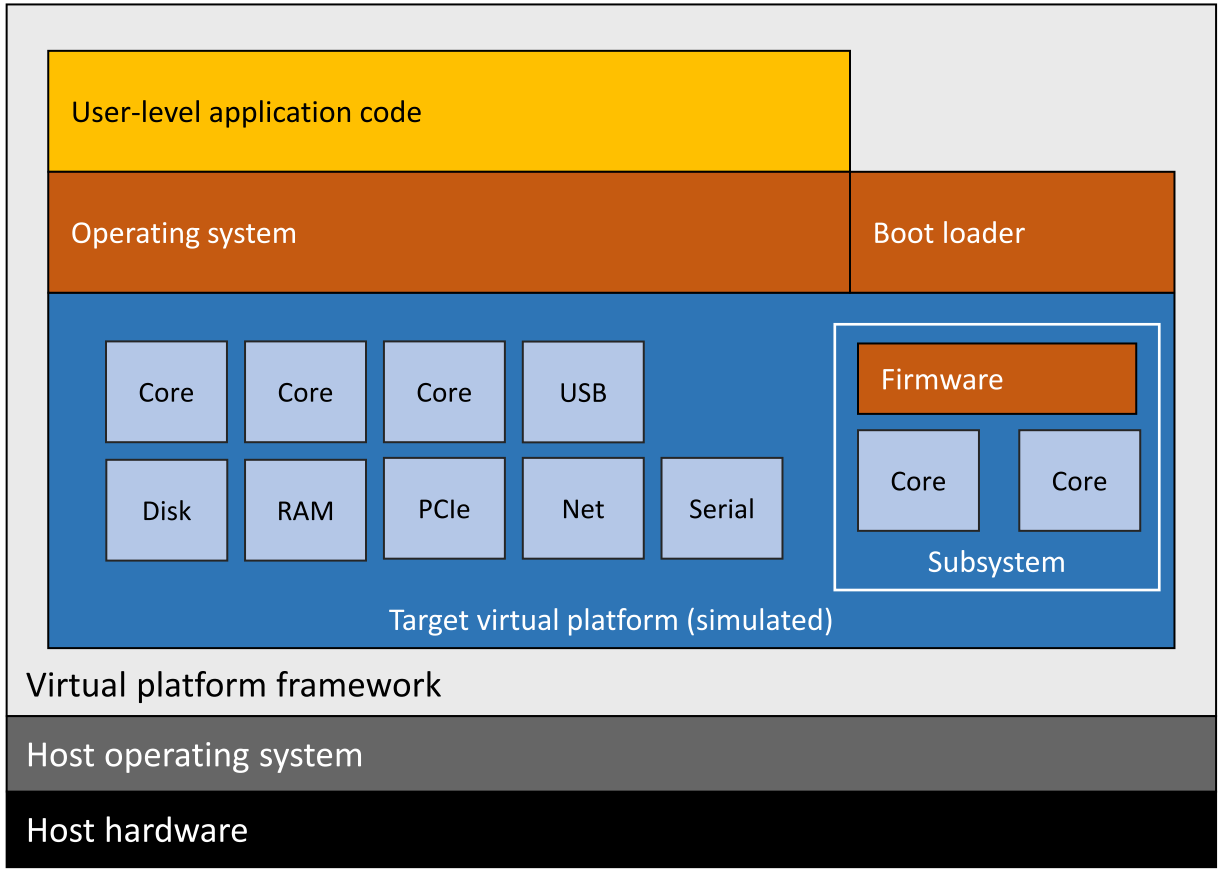
When to Start Software?

- Developing the SW for an SoC takes a defined amount of time
  - A. Without any pre-silicon vehicle SW development could start when RTL starts but has no guarantee to work on HW
  - B. Waiting for the first emulator then depends on the RTL being available delaying SW completion



FW development shift left **not possible** when the FW is tested using the emulator

Virtual Platforms? Why and What?

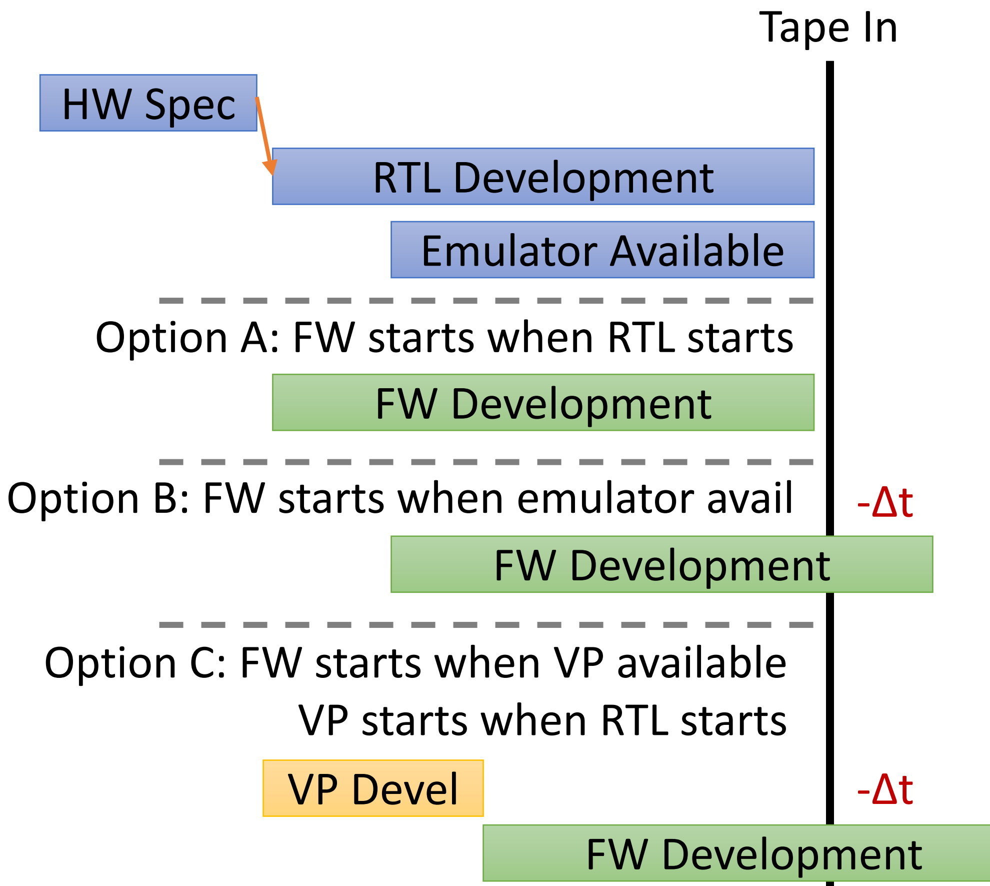


- Technology
  - Software model of hardware
  - Run **the same software** as the hardware
  - In our case, fast transaction-based models (TLM)
- Use case examples
  - Explore system architecture
  - Develop software early
  - Continuous integration of software and hardware
  - Debug and test software

SW “shift-left”

When to Start Software?

- Developing the SW for an SoC takes a defined amount of time
  A. Without any pre-silicon vehicle SW development could start when RTL starts but has no guarantee to work on HW
  B. Waiting for the first emulator then depends on the RTL being available delaying SW completion
  C. A VP developed from the same specs as required to enable RTL development also delays SW development completion

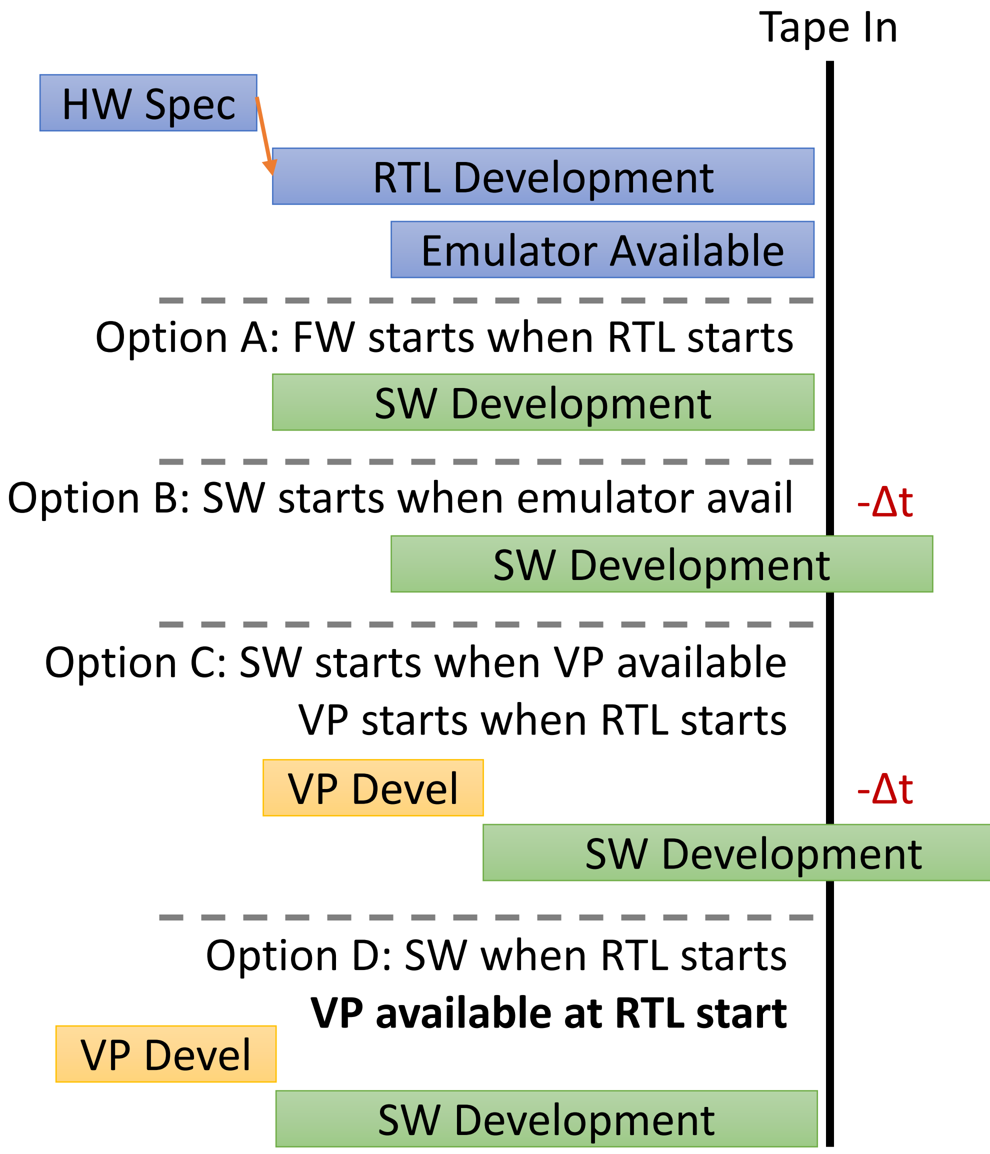


Tape In
HW Spec
RTL Development
Emulator Available
Option A: FW starts when RTL starts
FW Development
Option B: FW starts when emulator avail
-Δt
FW Development
Option C: FW starts when VP available
VP starts when RTL starts
VP Devel
-Δt
FW Development

FW development shift left **not possible** when the VP model is created from RTL spec

How to Start Software?

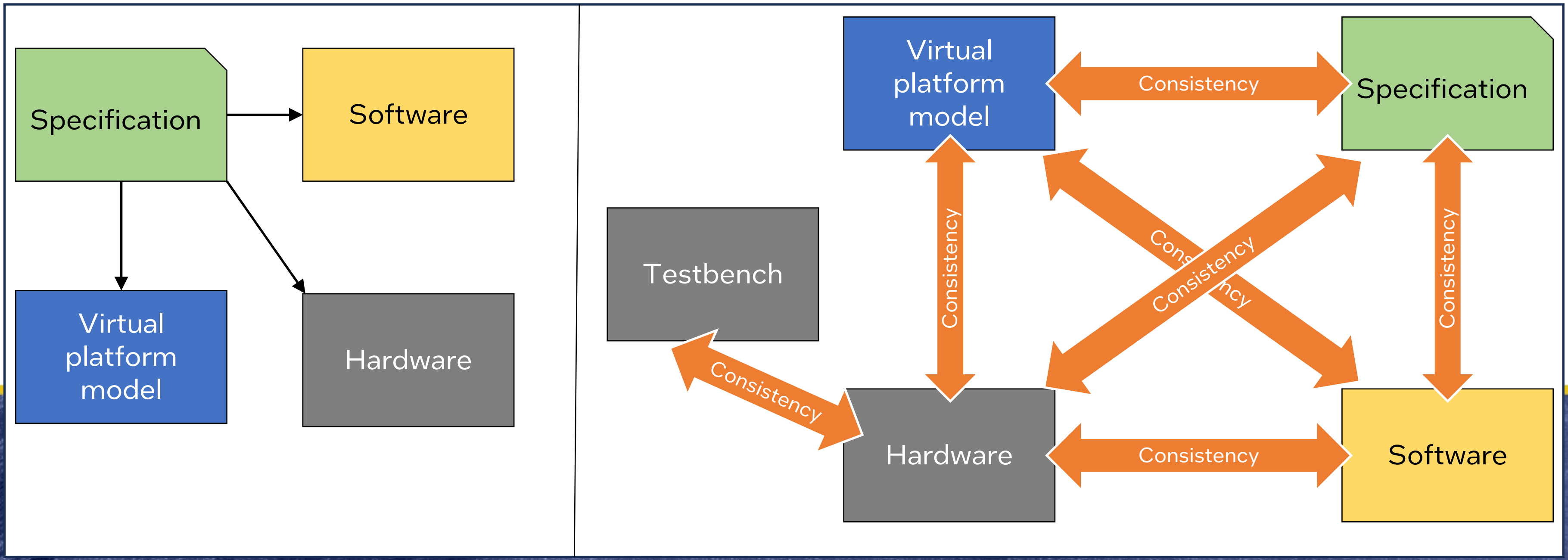
- Ideally RTL and SW should be able to start at the same time
  - Necessitates having a VP available for SW development
  - Necessitates starting VP development prior to specifications being available
  - Necessitates starting VP development without HW design collateral (IP-XACT)


Tape In
HW Spec
RTL Development
Emulator Available
Option A: FW starts when RTL starts
SW Development
Option B: SW starts when emulator avail
-Δt
SW Development
Option C: SW starts when VP available
VP starts when RTL starts
VP Devel
-Δt
SW Development
Option D: SW when RTL starts
VP available at RTL start
VP Devel
SW Development

Pre-SI value for a VP is to maximize time to emulator and time to tape in.

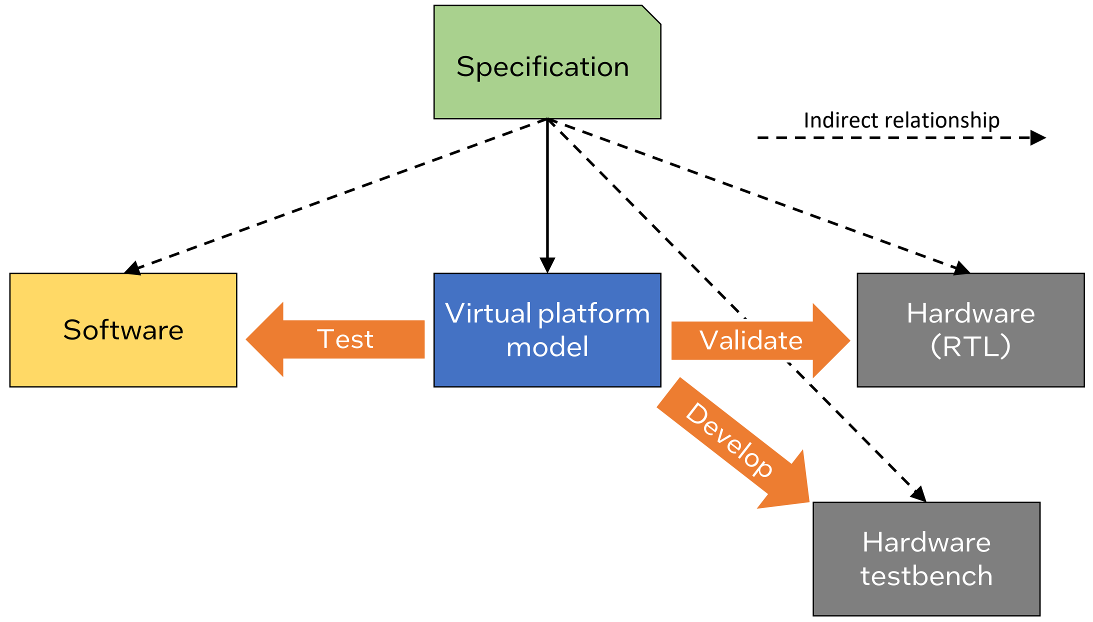
Where is the Spec? To Enable “Shift-Left”

- For **new HW blocks**, many teams need artifacts of the spec
    - Conceptually a “**master specification**” which all teams work from
- To ensure that the SW running on the VP will work on HW
    - The SW, VP, and RTL/HW must be consistent with the same spec



Our Approach to RTL-VP Model Consistency

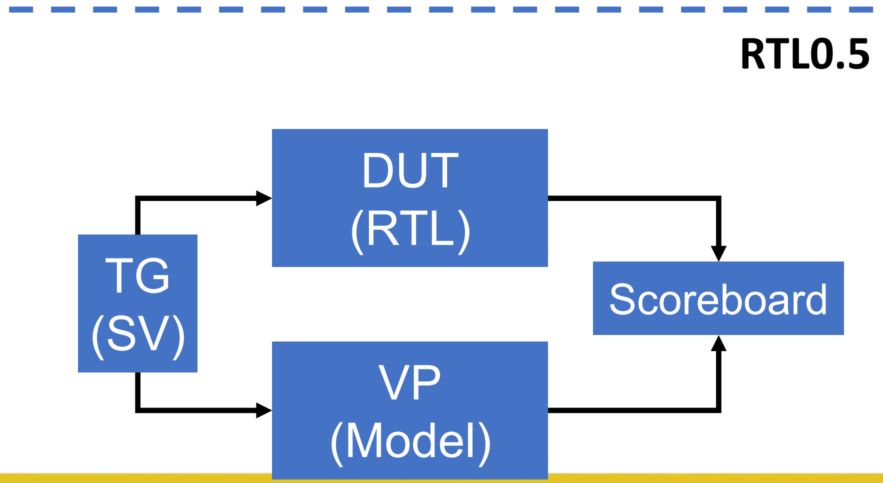
- Work "**model first**" and make the VP the executable specification
  - Validate the RTL/HW and SW based on the VP models
  - Make models accurate at the transaction level
- Keep the specification and model in sync as spec evolves
  - VP model is not frozen until the HW design is frozen



Need to ensure the RTL is validated against the model (i.e. executable spec)

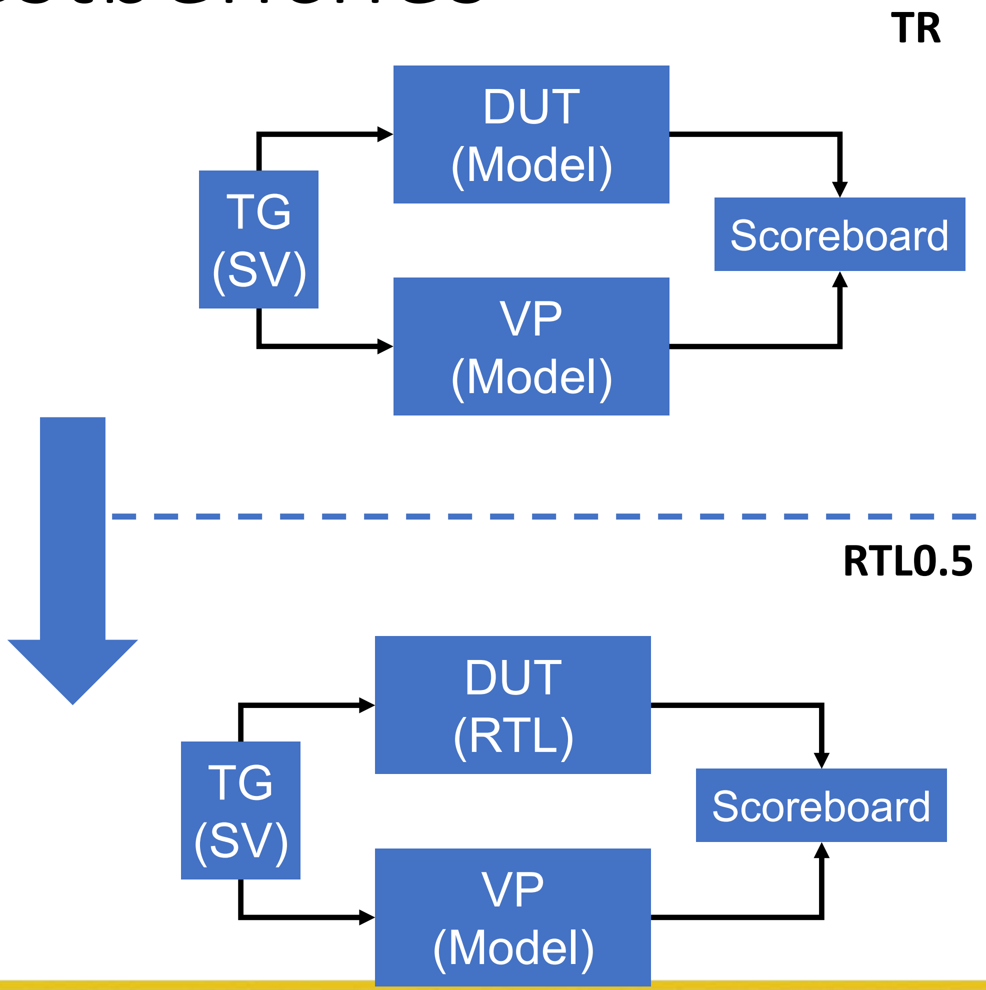
Closing the Model-RTL Loop

- Predictor in UVM scoreboard
  - Comparing VP model with RTL, at transaction level
  - Closes loop between RTL and model to enable correctness for SW shift-left
  - Replaces activity of using SystemVerilog Testbench (or other) for predictor

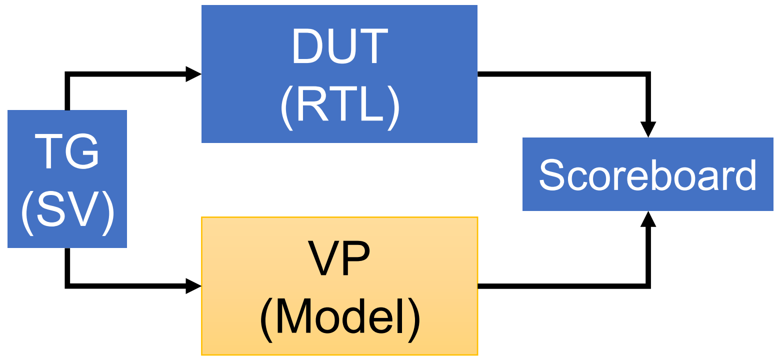


Uses for VP Models In RTL Testbenches

- Predictor in UVM scoreboard
  - Comparing VP model with RTL, at transaction level
  - Closes loop between RTL and model to enable correctness for SW shift-left
  - Replaces activity of using SystemVerilog Testbench (or other) for predictor
- Replacing an RTL block
  - Enable testbench development when RTL not yet available



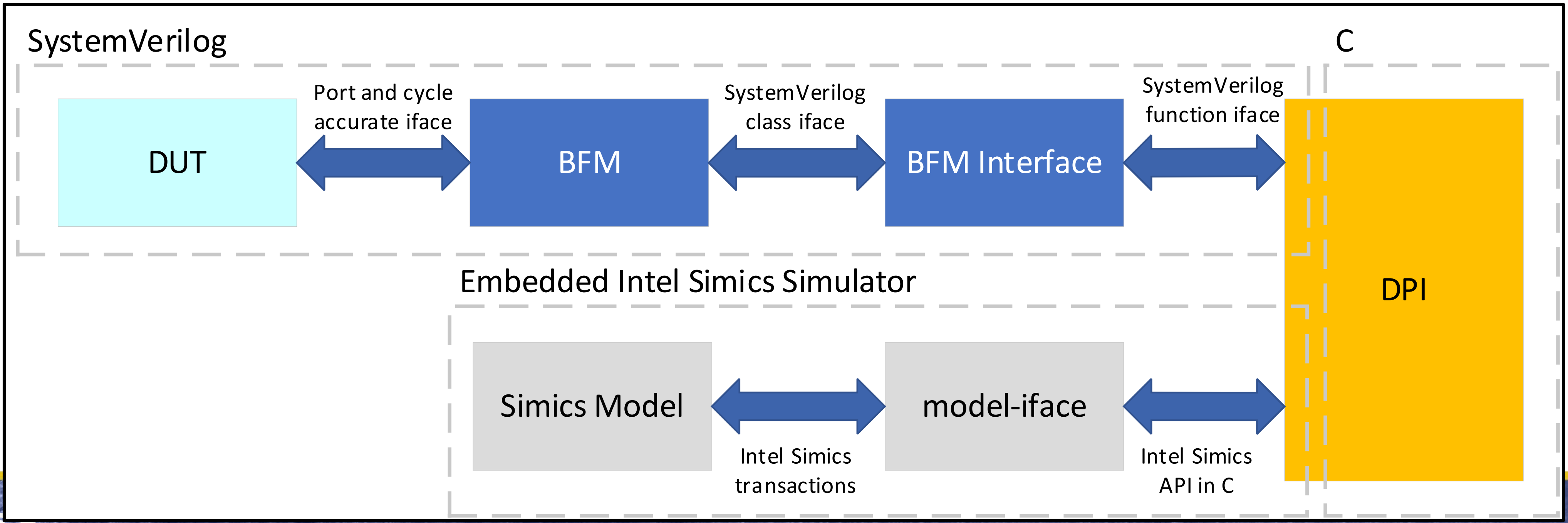
Embedding Simics in RTL



- Using Simics models requires the Simics simulator to execute the models
- Several topologies possible
  - Embedded Simics in RTL – RTL Simulator is top
  - RTL embedded in Simics – Simics is top
  - Simics and RTL simulator as siblings – connected processes
- We chose embedding Simics in the RTL simulator via DPI
  - Aligns to current use model for validating RTL
  - Embedding Simics in C is a standard feature of Simics

Integrating Intel Simics Models in RTL TB

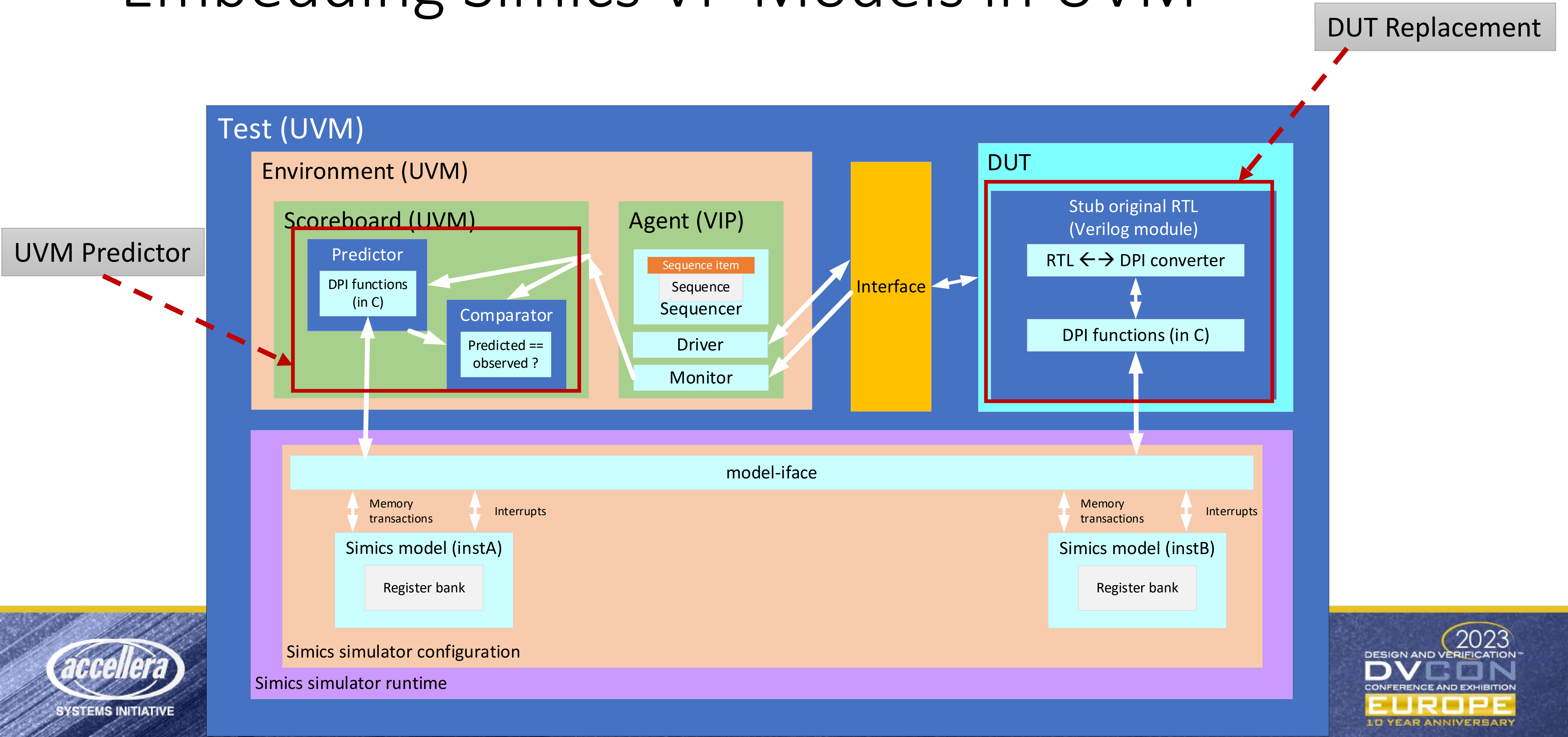
- Use BFMs as transactors between classes and cycle accurate interface
- Use SystemVerilog and DPI to translate into C library
- C library interacts with embedded Intel Simics simulator



SystemVerilog
C
DUT
Port and cycle accurate iface
BFM
SystemVerilog class iface
BFM Interface
SystemVerilog function iface
DPI
Embedded Intel Simics Simulator
Simics Model
Intel Simics transactions
model-iface
Intel Simics API in C

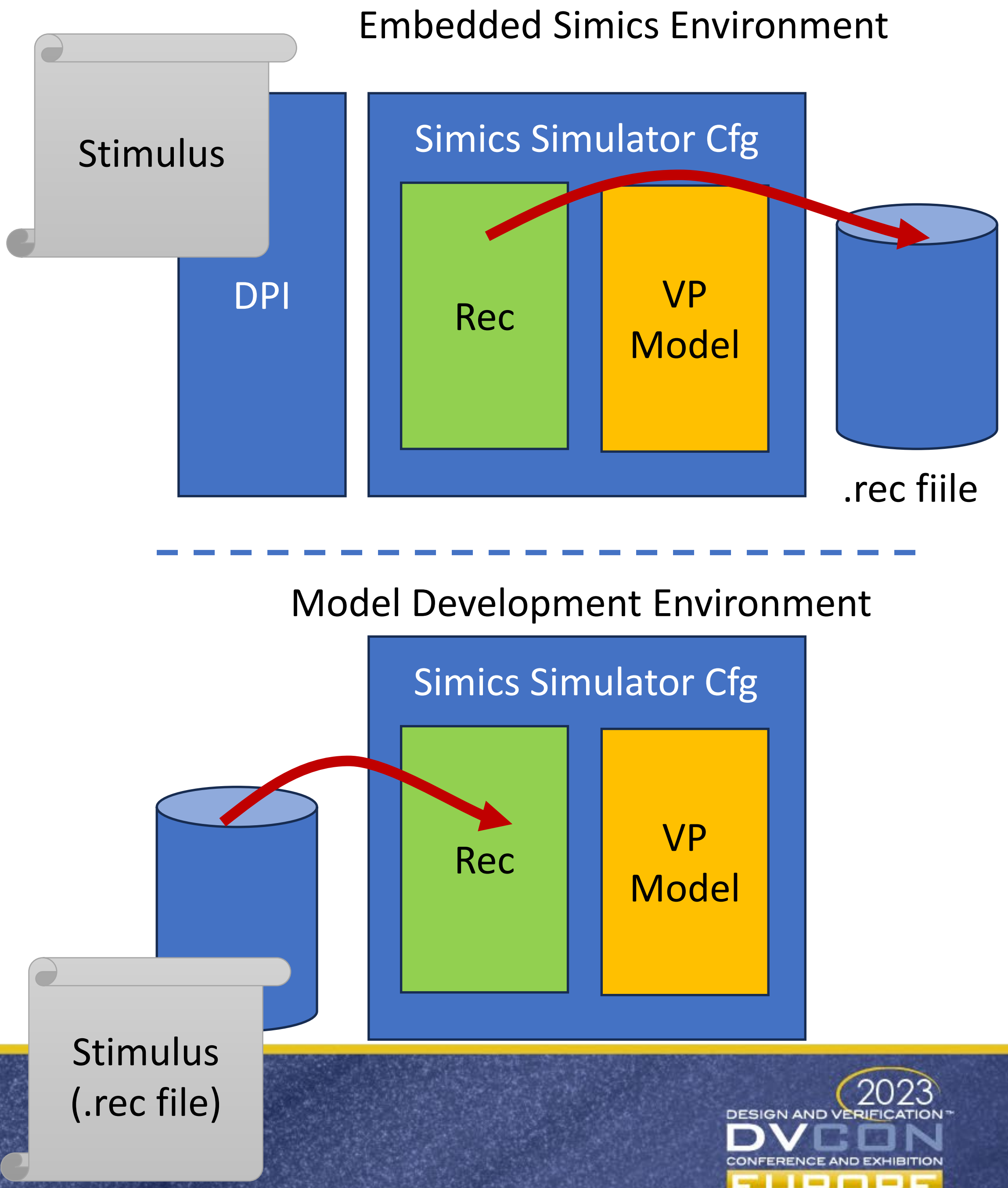
Enables supports RTL and Simics initiated reads

Embedding Simics VP Models in UVM



Debug: Record-Playback

- Expect spec interpretation differences between model and RTL
- Method to debug suspected VP model bugs outside of RTL sim
  - Record all transactions for playback in stand-alone Simics debug environment
- Enable model developers to debug in their enviornment



Technical and Non-Technical Changes

- Writing a constantly updated VP model is different
  - Write models as the spec itself is being written and updated
  - Achieved by integrating modelers into hardware team
- Requires a mindset-shift on modeling and usage of models
  - Total modeling effort now larger, but models available earlier
    - Enables feedback to architects earlier too
  - Users of models must understand model can change significantly throughout lifetime
  - Must be able to write models faster, sometimes at the expense of performance

Results - Benefits

- Use of the VP as the golden reference for software and RTL development and testing
  - Embeds the Intel Simics VP model into an RTL-level simulator
  - Use the VP model as the UVM predictor when testing RTL, closing the RTL-model loop
- Earlier testbench development
  - Embedding the VP model as an RTL replacement shifting left testbench development
- VP model and software debug in a stand-alone environment
  - Recorder enables debug of traces in a stand-alone VP simulation

Results - Limitations

- RTL developers and testers have limited VP model visibility
  - No inspection of internal states and signals possible
  - DV engineers cannot independently debug suspected testbench/model issues
    - Model developers can add extra hooks for some internal visibility
- Abstraction-level limitations
  - The TLM VP models are unable to mimic certain hardware behavior such as HW error response signals
  - TLM models are transactionally accurate but have different timing vs. RTL impacting some scoreboards

2023
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
EUROPE
10 YEAR ANNIVERSARY

Questions?

accellera
SYSTEMS INITIATIVE