Early Coverification of FW and HW to Speed Up Development of Embedded Systems

Jörg Bormann, Siemens EDA GmbH
joerg.bormann@siemens.com

VE-VIDES

funded by German Ministry of Education and Research

SIEMENS

Agenda

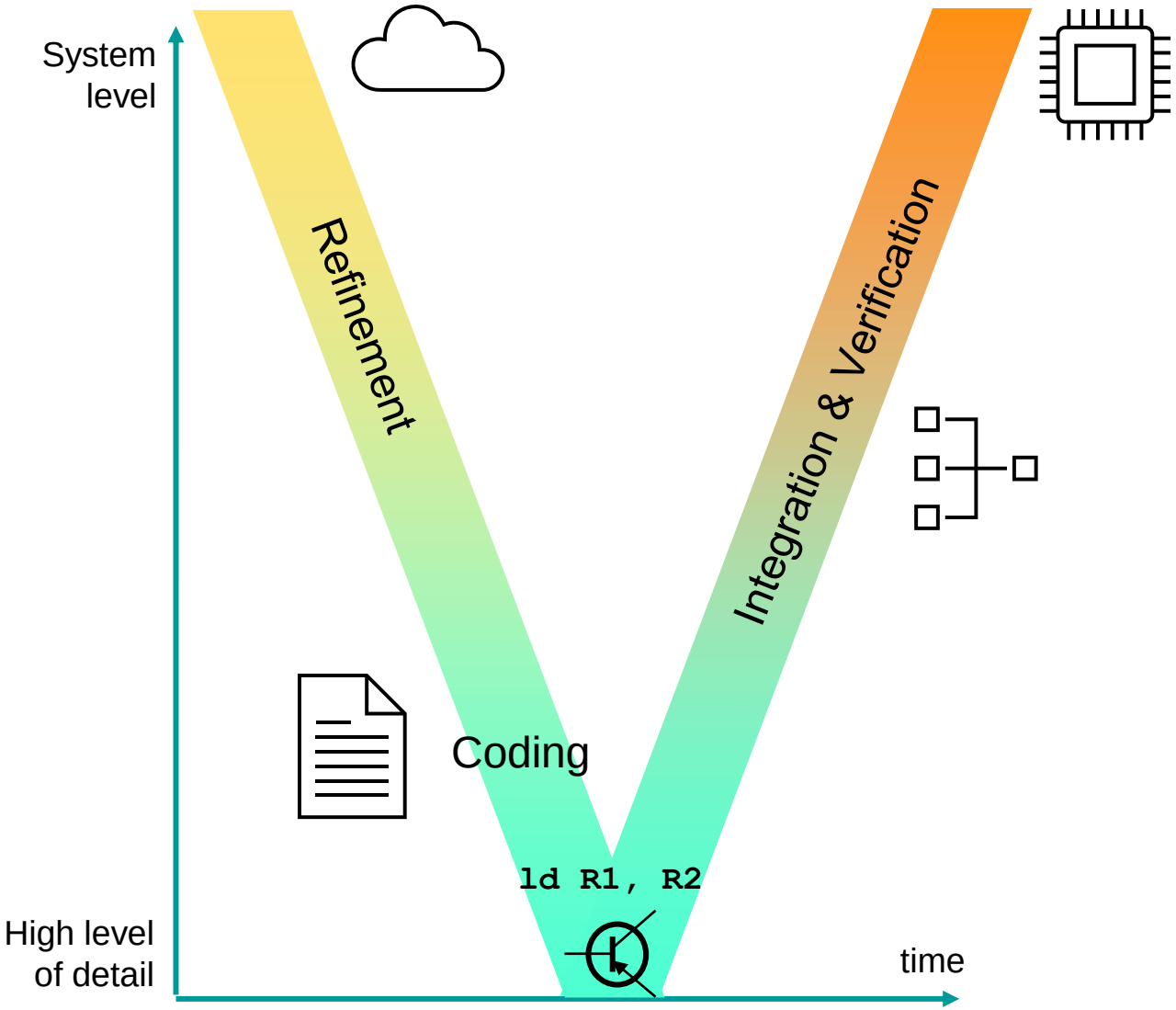
Disturbing Bugs in Verification of Embedded Systems

How to Get Rid of Disturbing Bugs

Early Removal of Functional FW/HW Integration Issues

Disturbing Bugs in Verification of Embedded Systems

V-Model

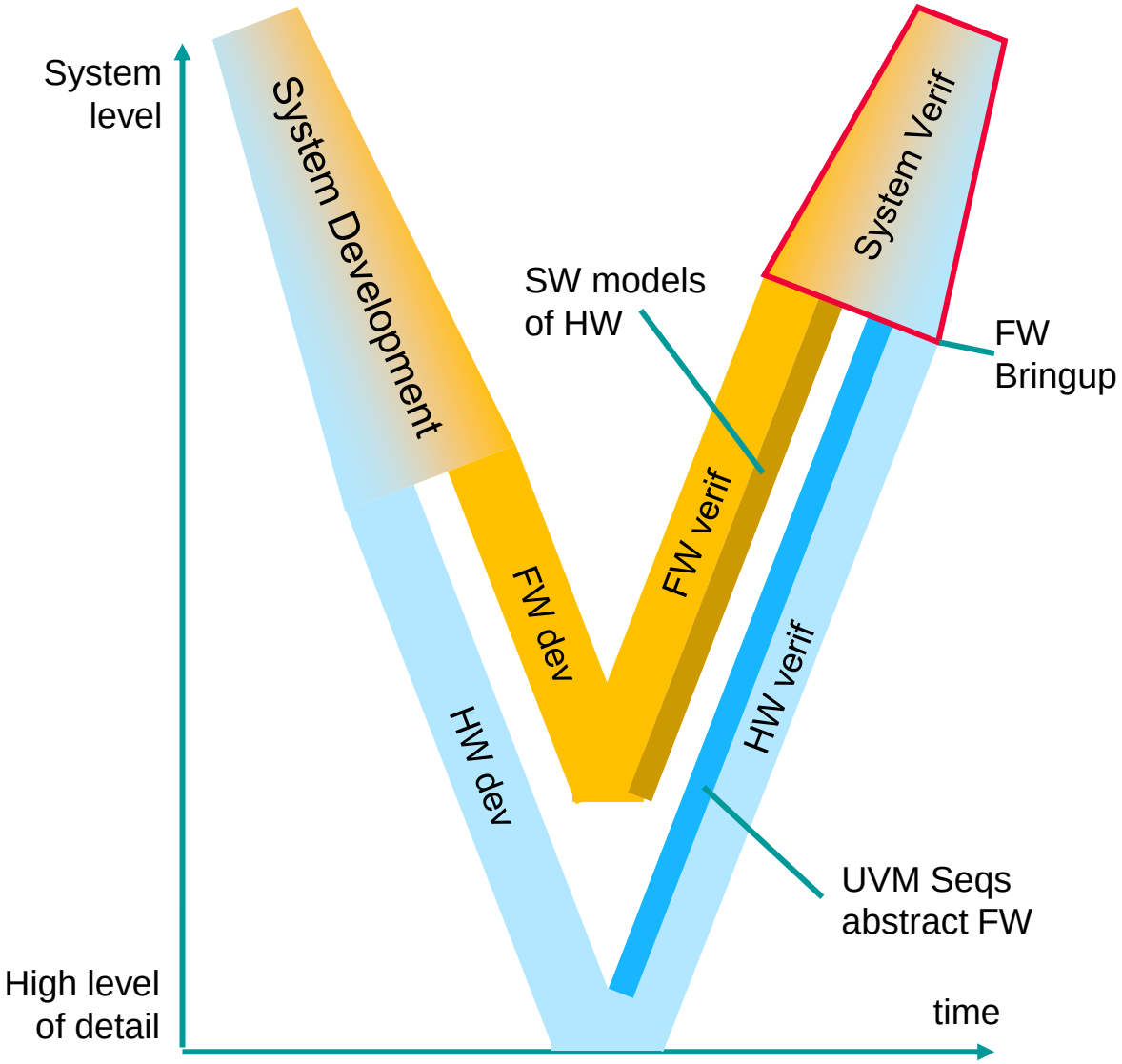


V-Model informally describes system design process from top level requirements to final system.

- Refinement to more and more level of detail
- Coding (RTL, C-Code)
- Successive integration and verification up to the full system

Qualitative description.

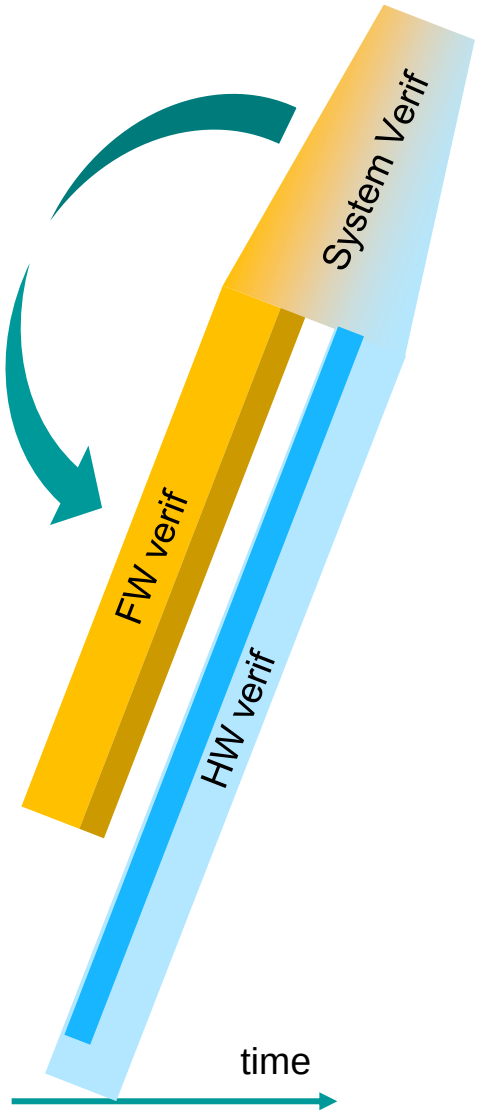
Error Classes in System Verification of Embedded Systems



Interesting Bugs in System Verification
Algorithmic complement of HW and FW
Bandwidth & latency
Power consumption
System Security & Functional Safety

Disturbing Bugs in System Verification
FW verification escapes
HW verification escapes
functional FW/HW integration issues

Cost of the Disturbing Bugs in System Verification



Analysis

- On System Level more difficult than on lower hierarchy levels
  - Larger DUT => large search space => slow
  - System verification machines provide only limited visibility into the embedded system
- Involves rare and expensive system experts
  - Expertise: HW, FW, intended interaction, HW accelerated system verification machines
  - Experts may be blocked until a workaround or fix is found
  - Delayed product rollout

Fix

- Root cause in FW: FW fix is straight forward
- Root cause in HW
  - RTL fixes interfere with physical design – very expensive
  - FW workaround: Development + verification effort, additional risk
  - Feature cancellation
  - ECOs

Total cost indicates if HW or FW flows need to change.

Number of Disturbing Bugs allows to predict benefit of HW or FW flow changes.

SIEMENS

How to Get Rid of Disturbing Bugs

Options to Improve HW Flow

Incremental

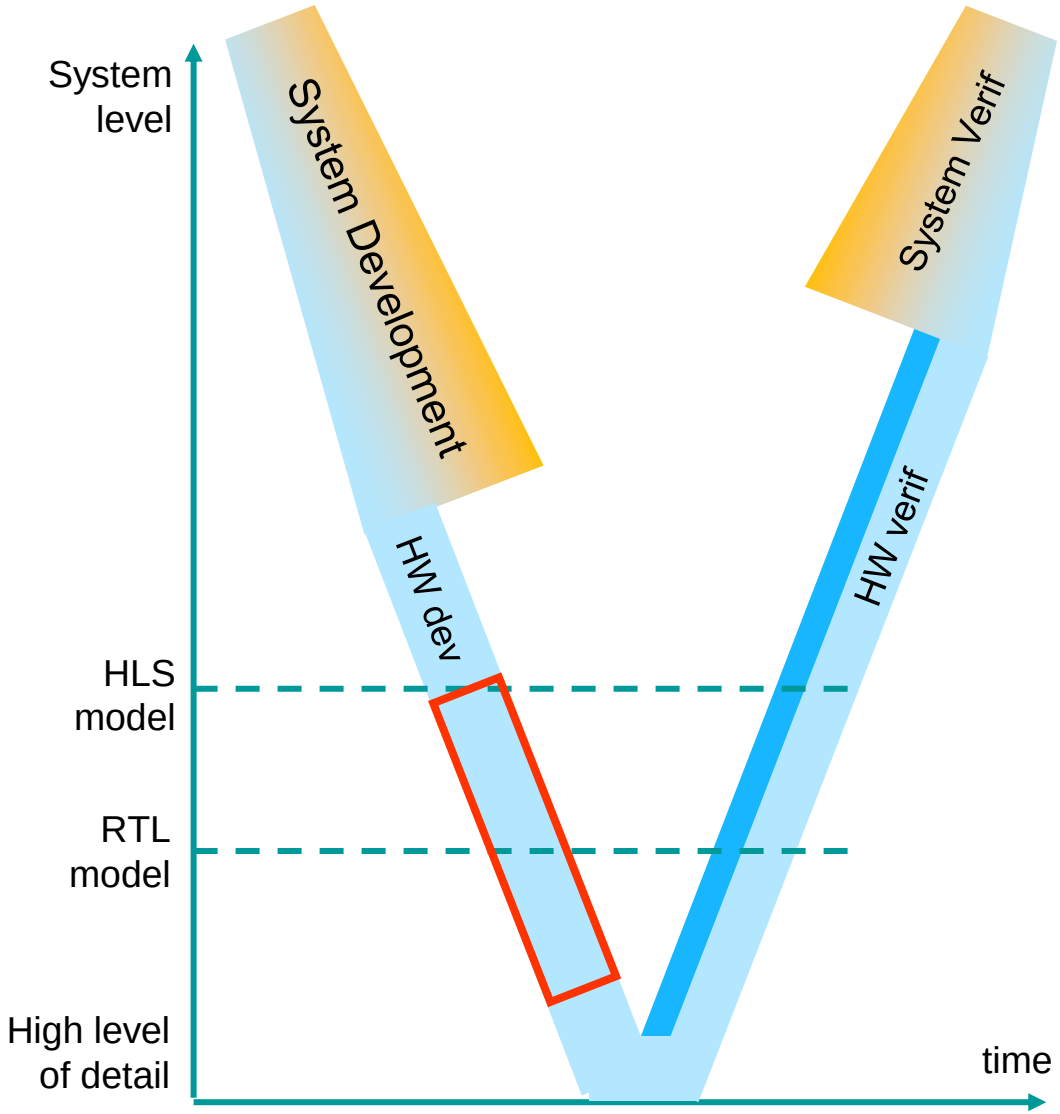
- More effort for HW development and verification
- Process optimizations
- Tools to better support the existing flow

More powerful alternatives

- Design entry on higher level
  - CoreDSL, Chisel, ...
  - High Level Synthesis

- Full exploitation of the capabilities of formal
  - Theorem provers (e.g., Isabelle, Coq), Lubis EDA, GapFree

High Level Synthesis



For algorithmic designs

High quality of HLS model

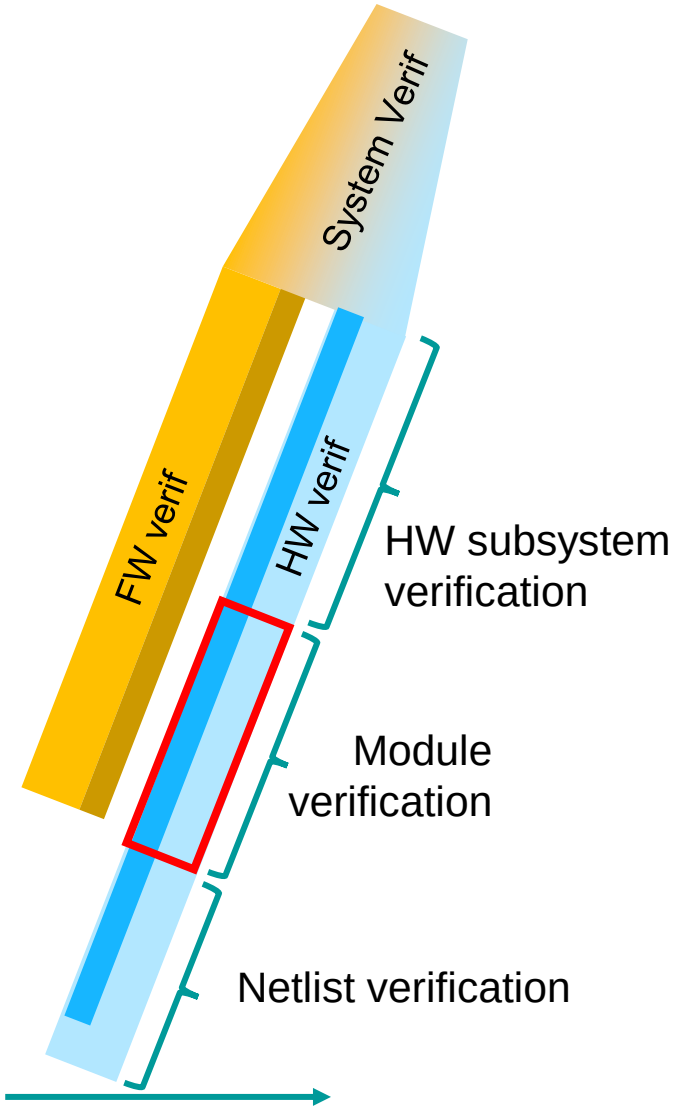
- ≤ 1/10 LoC of a similar handwritten RTL model
- HLS model ~ system development models.

Faster development of HLS model than RTL

Extensive design exploration => efficient circuit

Automated flow up to netlist

GapFree Verification



A variant of assertion based formal verification to verify all module behavior

- Assertions about transactions of the DUT
- Developed during the verification
- Until the assertions contain a transaction level model of the DUT.
- Completeness checker signs off the transaction level model.
- Clear termination criterion

KPIs (from OneSpins consulting projects)

- Progress: av. ~ 3000 LoC[1] / PM
- Escape rate: 1 bug per 30 000 LoC

[1]Lines of handwritten RTL code

SIEMENS

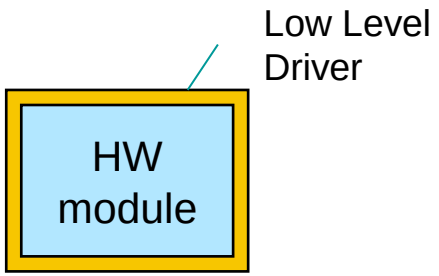
Early Removal of Functional FW/HW Integration Issues

Functional FW / HW Integration Errors

Error Class	Example
FW writes „reserved“ value into config reg	DMA configured with address increment = 0.
Encoding mismatch	HW and FW use different endianness HW encoding allows gate count savings.
Synchronization issue	FW forgets polling loop / HW does not block transaction / ISR reacts wrongly on interrupt
Wrong transaction sequence	FW activates a HW accelerator, then configures it.

If these bugs occured between two HW modules instead of FW & HW

- Functional bugs
- System verification is inappropriate flow phase for their detection.



Idea: Small FW wrapper (= „Low Level Driver“) around HW module

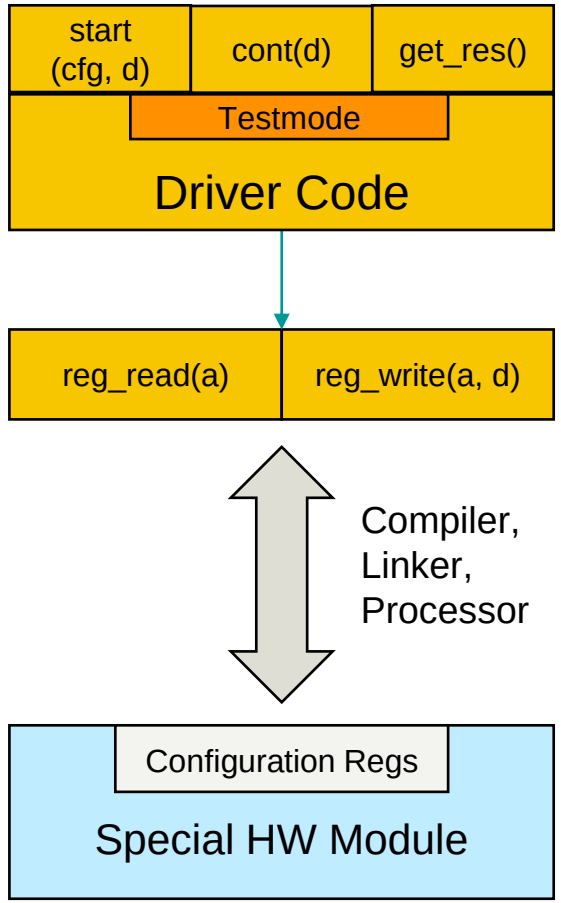
- **Developed during HW module development**
- **Shall capture HW specifics and expose HW functionality like a SW lib.**

SIEMENS

Contents of Low Level Drivers

Upper API: C/C++ funs that concisely expose the HW module functionality

Lower API: HW register access functions



Functionality

- Configures HW module (with transactions in supported order)
- Converts parameters into HW encoding, if necessary
- Synchronization with HW module
- Read result, change encoding, and return to higher FW layers

Testmode (to be removed for product version)

- Assert statements to detect wrong usage of driver functions
- E.g., checks parameters & sequence of driver calls

LL driver code is similar to logic descriptions in System Verilog.

SIEMENS

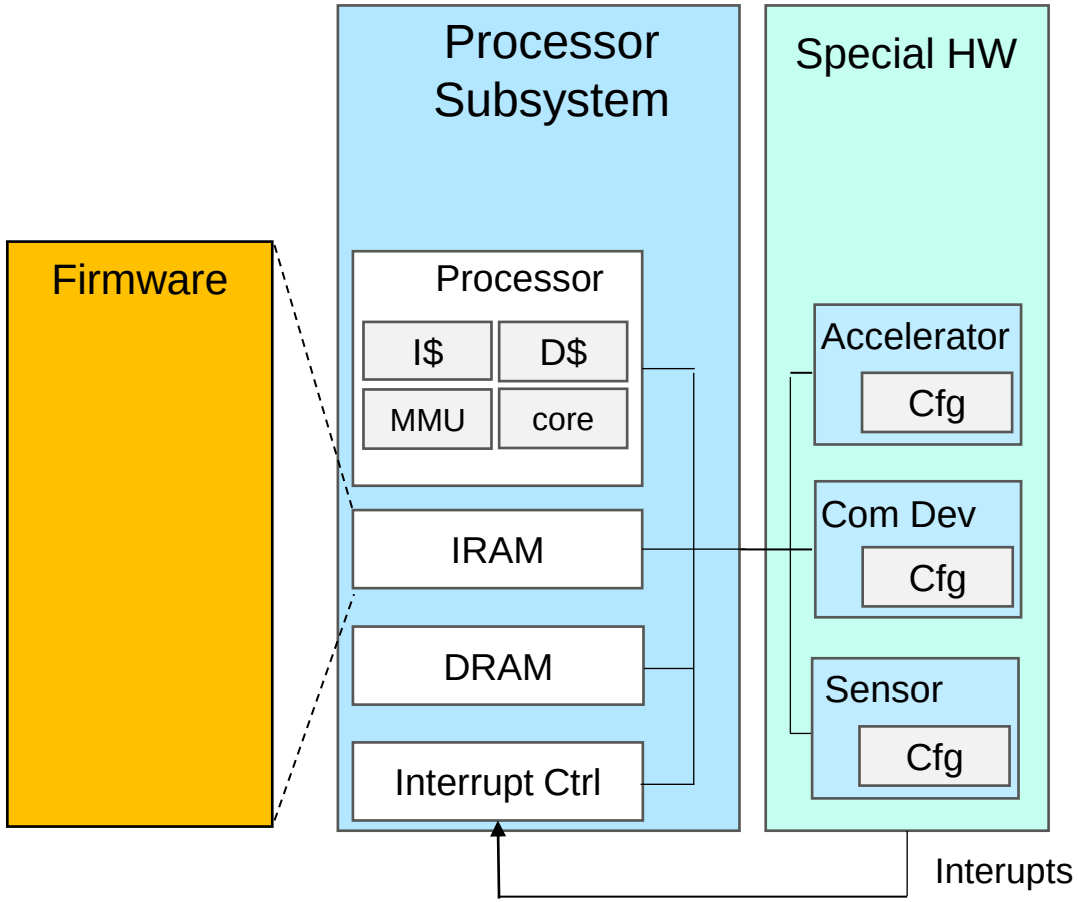
Verification of FW/HW Integration – Current Status and Convictions

Observations:

- FW Developers must learn HW peculiarities to properly use the configuration registers via the signal interface.
- VPs for special HW modules are custom developments, with little ROI.

Convictions of practitioners:

- Early FW/HW coverification would require additional system engineering resources, which are rare.
- Processor subsystem necessary, hence we can include the whole system.
- Register descriptions like IP-XAct ensure FW/HW integration at an early stage.



SIEMENS

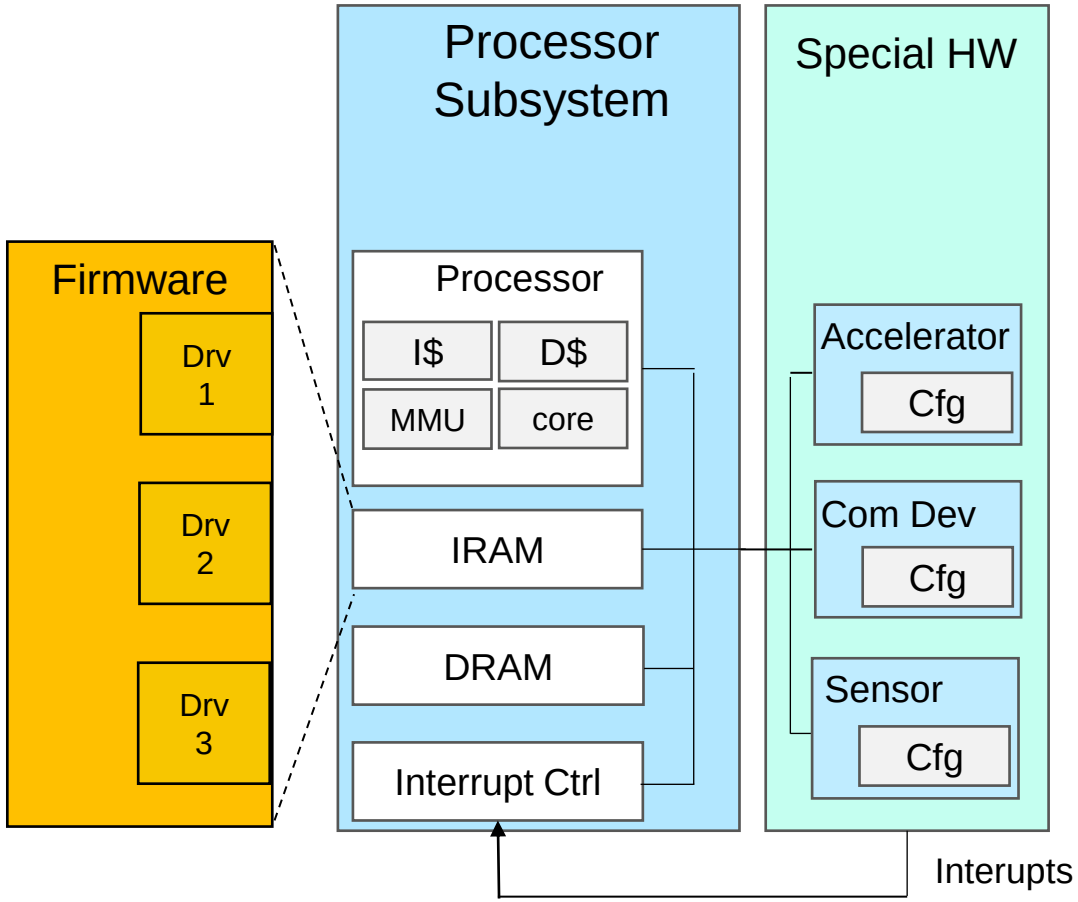
Early Verification of FW/HW Integration

Differences to current process:

- FW engs need to learn less, HW functionality is exposed like a SW library.
- HW engs need to learn LL Drv design. Only a one time effort. Similarity with logic descriptions in System Verilog.

Contrary to the convictions of practitioners:

- HW eng can verify LLDrv + HW module.
- C/C++ semantics + timing variations can replace processor.
- LL Drv + HW module should be verified for many processors.
- Each pair of LLDrv + HW module can be verified separately.
- LLDrv complement IP-Xact to avoid functional integration issues.



SIEMENS

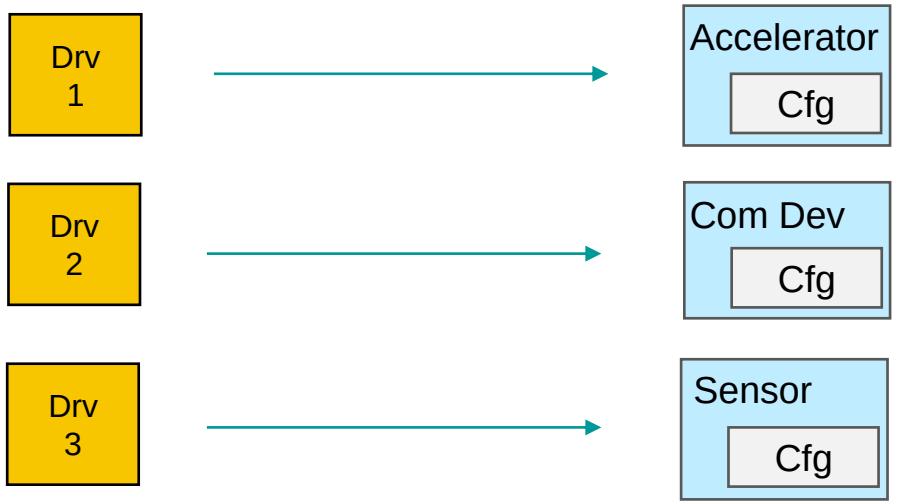
Early Verification of FW/HW Integration

Differences to current process:

- FW engs need to learn less, HW functionality is exposed like a SW library.
- HW engs need to learn LL Drv design. Only a one time effort. Similarity with logic descriptions in System Verilog.
- For FW verification, insert LLDrv between FW and VP to check FW with testmode.

Contrary to the convictions of practitioners:

- HW eng can verify LLDrv + HW module.
- C/C++ semantics + timing variations can replace processor.
- LL Drv + HW module should be verified for many processors.
- Each pair of LLDrv + HW module can be verified separately.
- LLDrv complement IP-Xact to avoid functional integration issues.



SIEMENS

Formal Verification of LL Drivers and HW

Verify LL Driver and HW module together

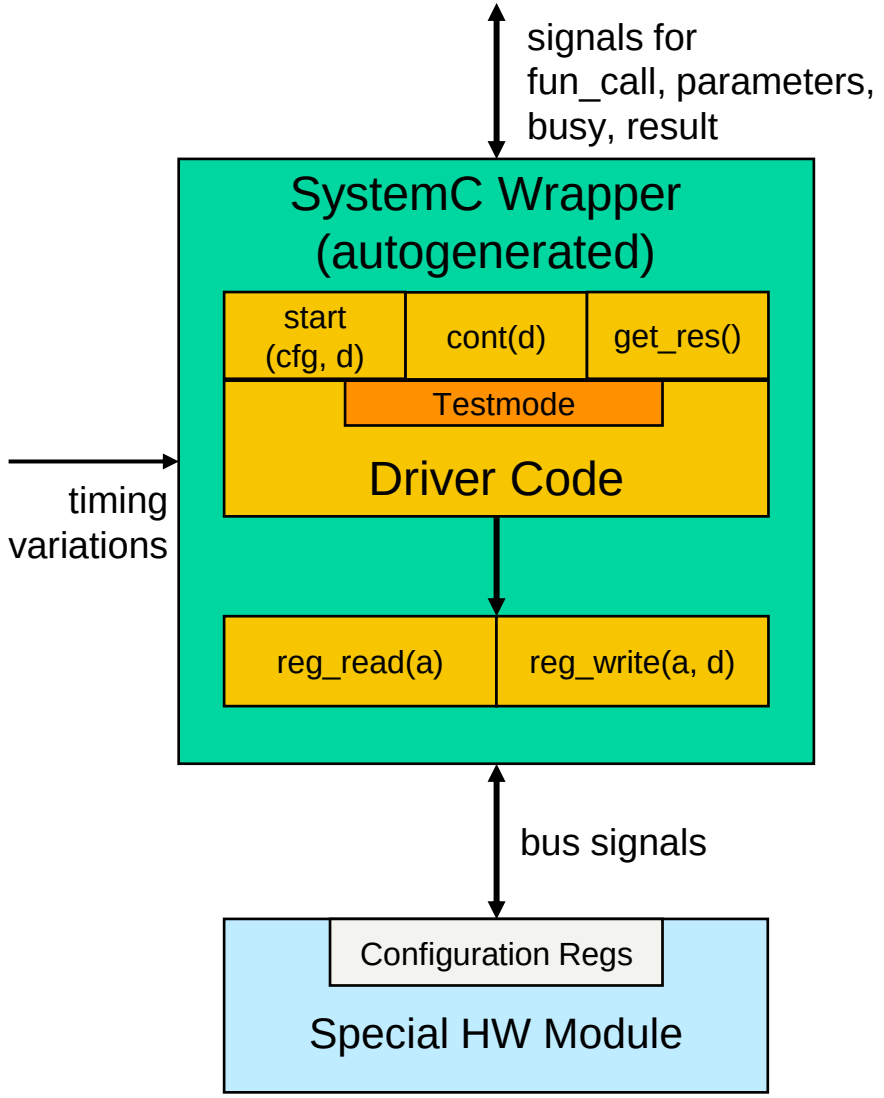
- No separate HW module verification

Model for verification: Embed LL Drv in SystemC =>

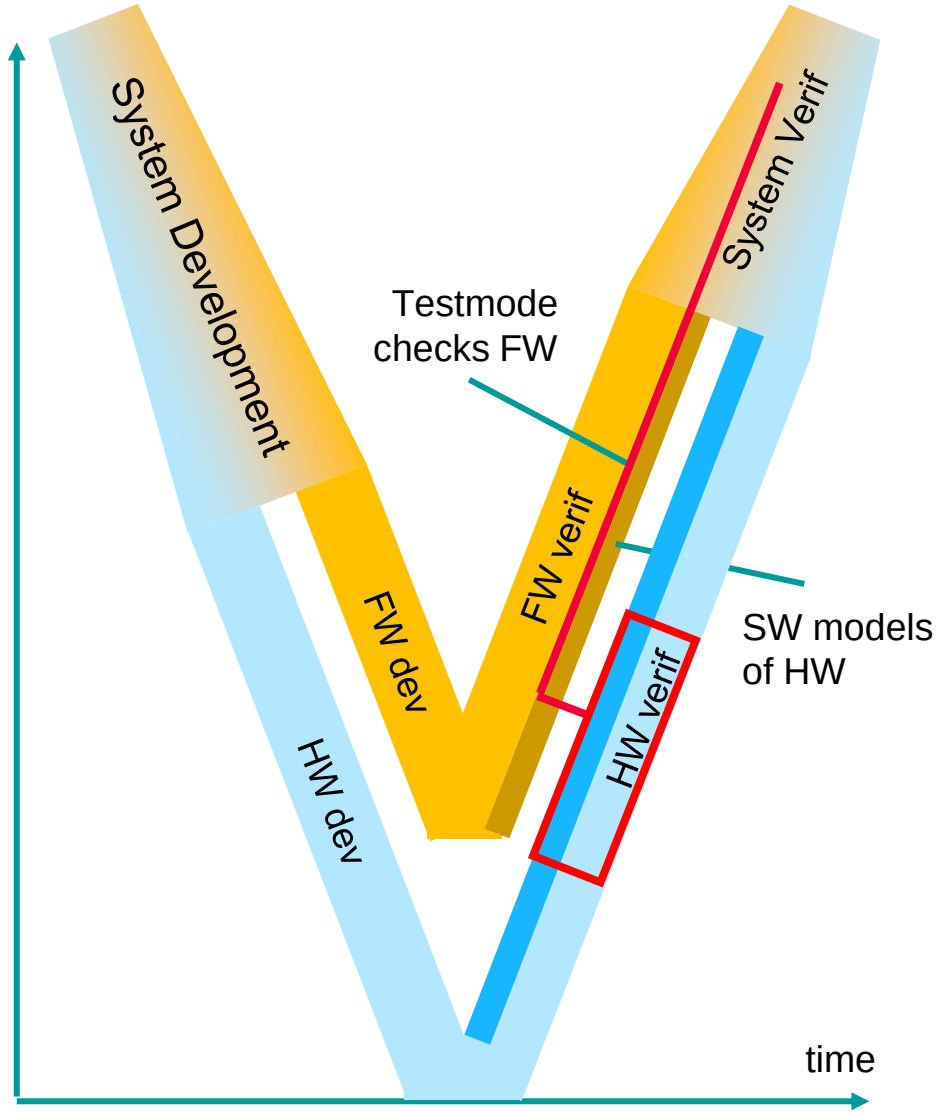
- Signals to control and examine the driver and
- Signals to interact with the Special HW Module.
- Extra inputs to create timing variants

Verification

- for each sequence of LL Drv function calls and
- for all parameter values
- assume: The testmode does not complain
- assert: LLDrv + HW provide the expected E2E functionality.
- This verification shows that testmode guarantees proper HW configuration.



Advantages



Many functional integration bugs avoided by design flow

- Because HW engineers are involved in LL Drv design.

Functional integration bugs are found during module development

- smaller DUT => smaller analysis effort per bug
- Less config use cases to be verified, but more timing alternatives
- RTL changes possible and cheap
- Less disturbing functional FW/HW integration problems

Low complexity of the verification problem (no processor subsystem)

- Enables Assertion based formal, GapFree, RTL sim.

Benefits for FW design

- FW engineers need not learn signal I/F of HW => faster, less buggy.
- FW deploys HW like a SW library => faster, less error prone
- Testmode checks FW early

Wrap Up

Disturbing Bugs in Verification of Embedded Systems

- Proposal to count them and measure related effort to objectivate need for change

How to Get Rid of Disturbing Bugs

- Consider GapFree or High Level Synthesis

Early Removal of Functional FW/HW Integration Issues

- Proposal for a flow change: LL Drv developed and verified in the HW module verification flow

Disclaimer

Subject to changes and errors. The information given in this document only contains general descriptions and/or performance features which may not always specifically reflect those described, or which may undergo modification in the course of further development of the products. The requested performance features are binding only when they are expressly agreed upon in the concluded contract.

Contact

Published by Siemens EDA

Jörg Bormann
Program Manager Advanced Verification
DI SW ICS ICV OS
Nymphenburger Straße 20a
80335 Munich
Germany

Phone +49 1577 356 4108

E-mail joerg.bormann@siemens.com

SIEMENS