TLM-based Virtual Platforms at Ericsson Challenges and Experiences

Ola Dahl, Michael Lebert, Eric Frejd

Ericsson AB, Sweden

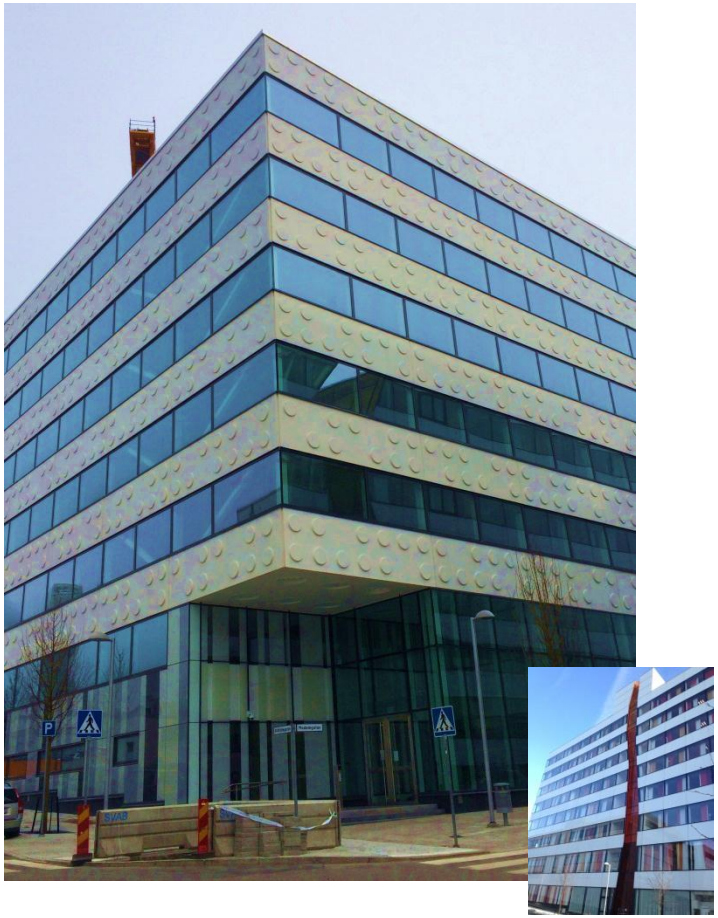
accellera SYSTEMS INITIATIVE

ERICSSON

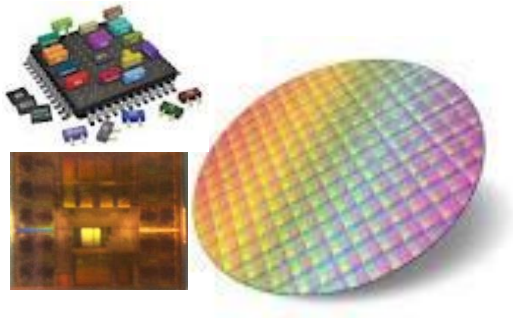
2016 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

Our Department at Ericsson

- 200+ people Design Center, Stockholm and Lund, Sweden
- Perform extensive benchmarking and evaluation of new and future silicon technologies
- Evaluate and develop our SoC architecture for many-core



Our Products



System-On-Chip

Baseband, Radio, and Control SoCs

System-On-Board

Radio Unit and Digital Unit

System-In-Cabinet

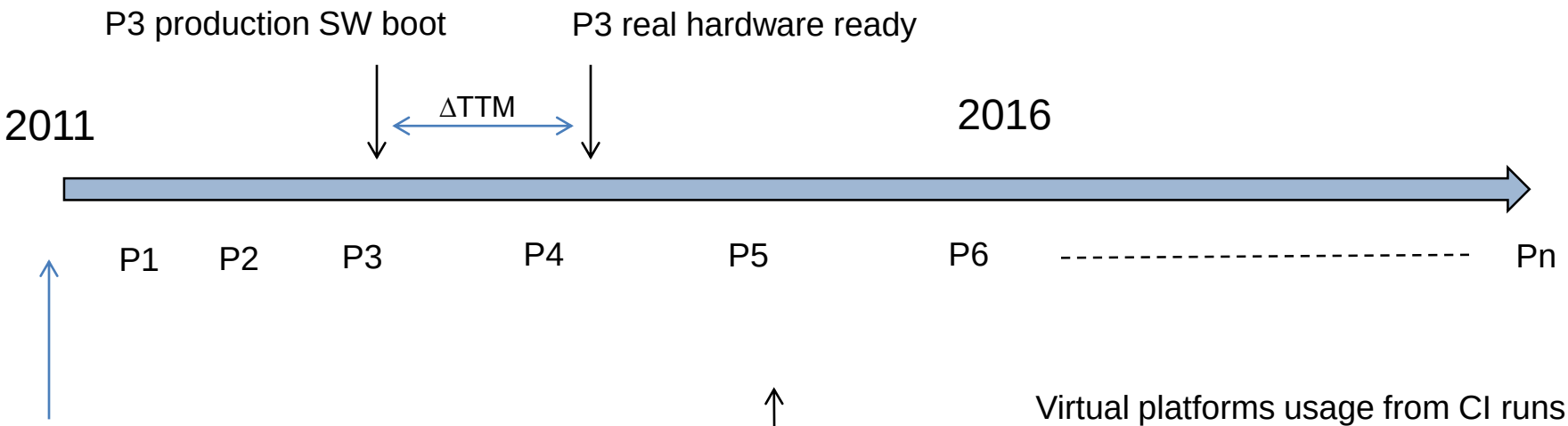
RBS 6000 series of multi-standard base stations

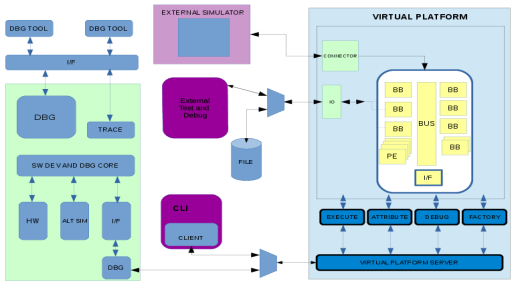
Contents

- From Prototype to Product*
- Requirements and Models
- Platform Assembly
- Usage Patterns
- SW Development and Virtual Platforms
- Conclusions

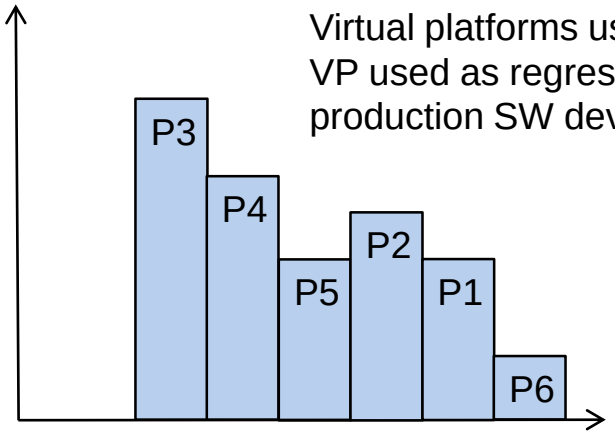
* Virtual platform (a.k.a. virtual prototype)

From Prototype to Product





VP architecture



Virtual platforms usage from CI runs
VP used as regression target in production SW development and test

User run stats 2016 (illustration)

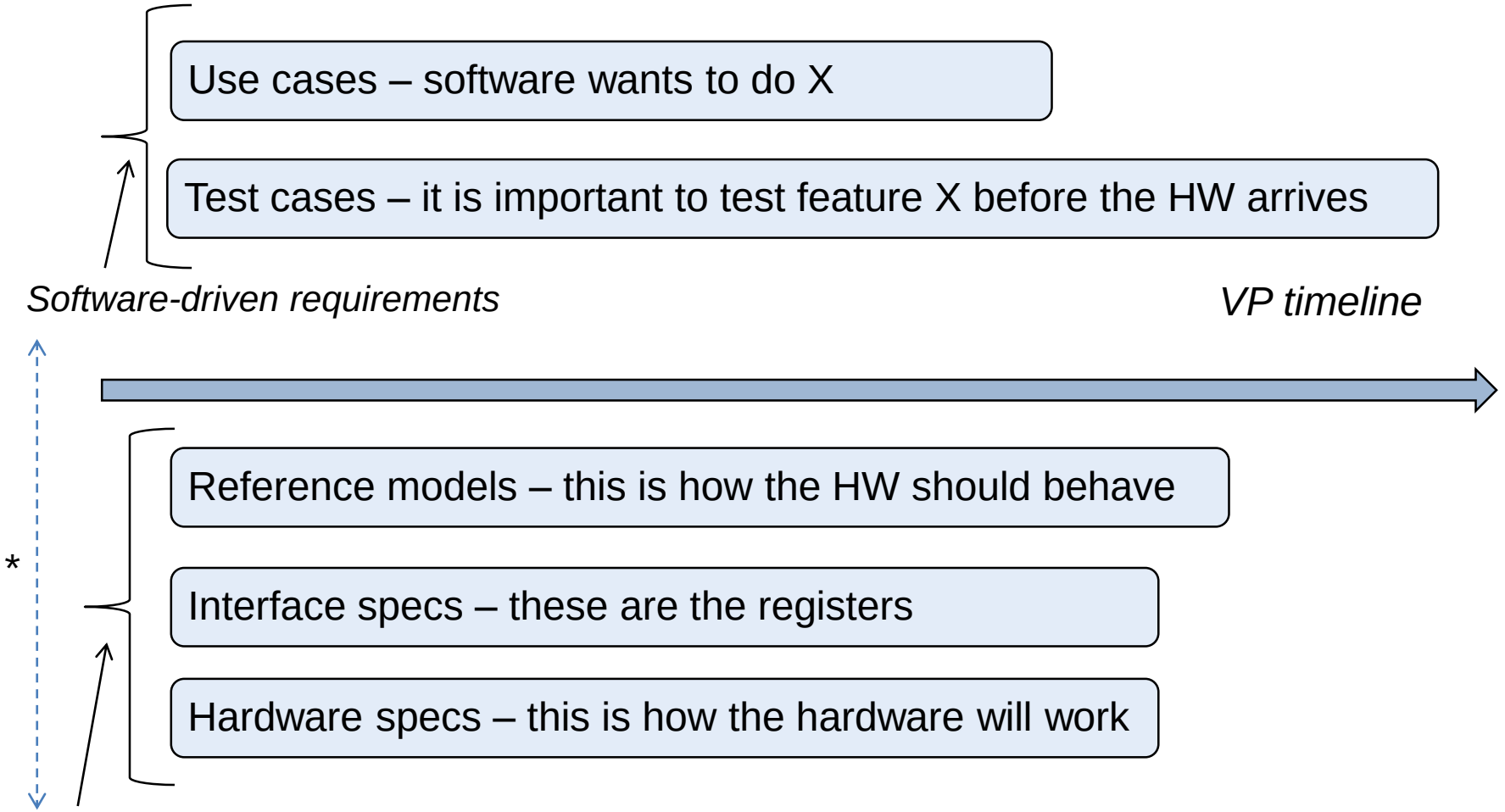
Challenges

- *Staffing – getting the right people onboard* – from RTL, via embedded, to OOAD and SW architecture
- *User community acceptance* – baseband and radio, users in different SW layers, ASIC simulation, board simulation, external tools interaction
- *Setting the requirements right* – from PoC to a more mature platform architecture, balancing hardware and software requirements, choosing the appropriate model accuracy (from registers, via functional models, to signal processing models)

Experiences

- *The first platform is still used* – long after the hardware is available, puts demands on our support organization
- *Model creation can (and should) be distributed* – our team has taken on the role of integrator, and we receive models from different sources, both in-house and external
- *CD/CI is a bliss* – daily builds and deliveries, well-defined baselines for external dependencies (e.g. target software from our users that we also run in our own regression testing)

Requirements and Models



* Find the best trade-off (varying over time and platform)

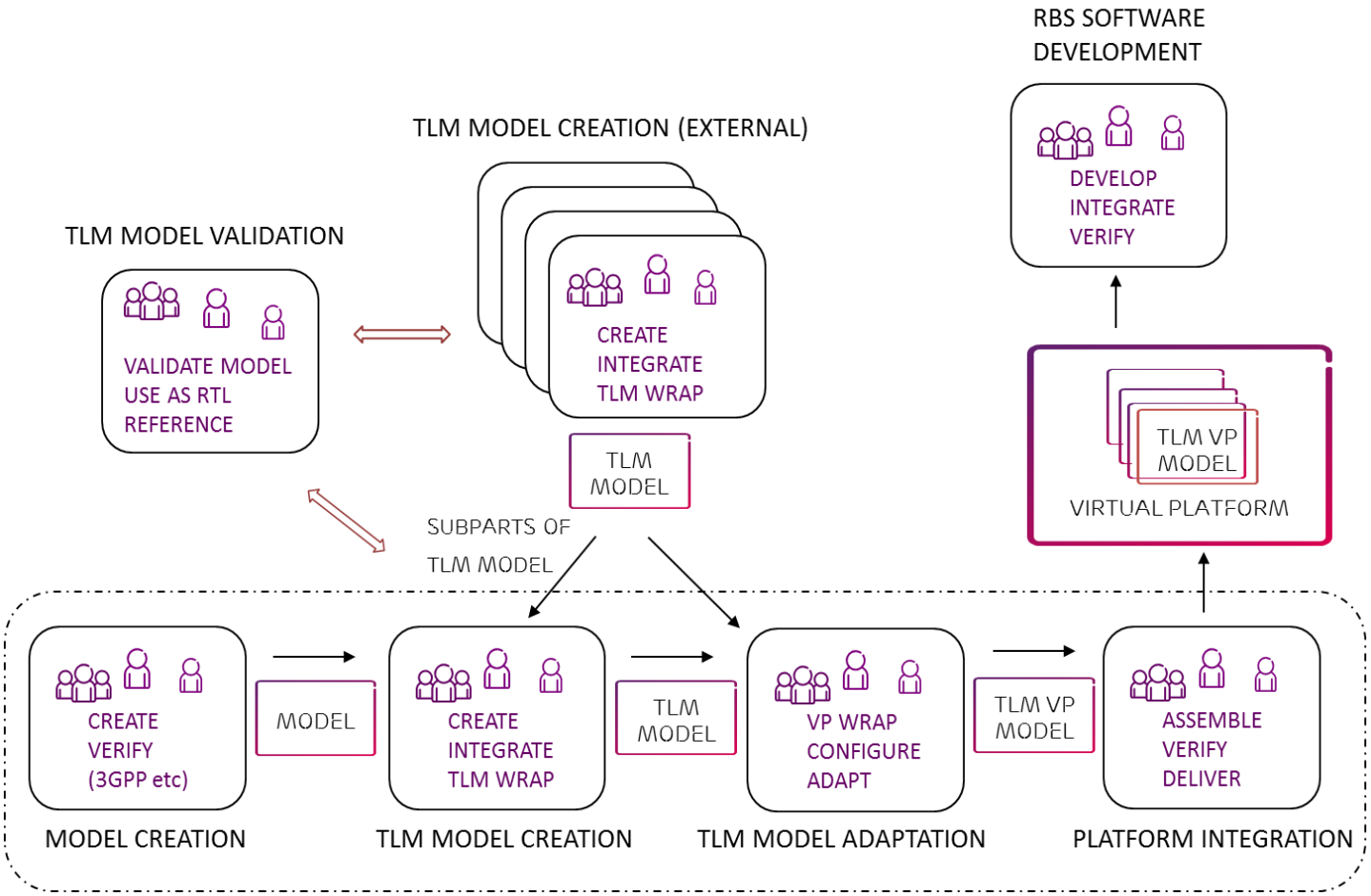
Challenges

- *Where can we find information?* – reading the (changing) specs (HW and SW), (organizational) networking, requests for information
- *When are we* done*?* – what is the use case? functional software verification (mostly), need also to handle new requirements (due to continuous VP usage), sometimes reflecting also timing
- *Who* validates *the golden reference?* – some models are used also for RTL verification, some models use reference code (e.g. signal processing)

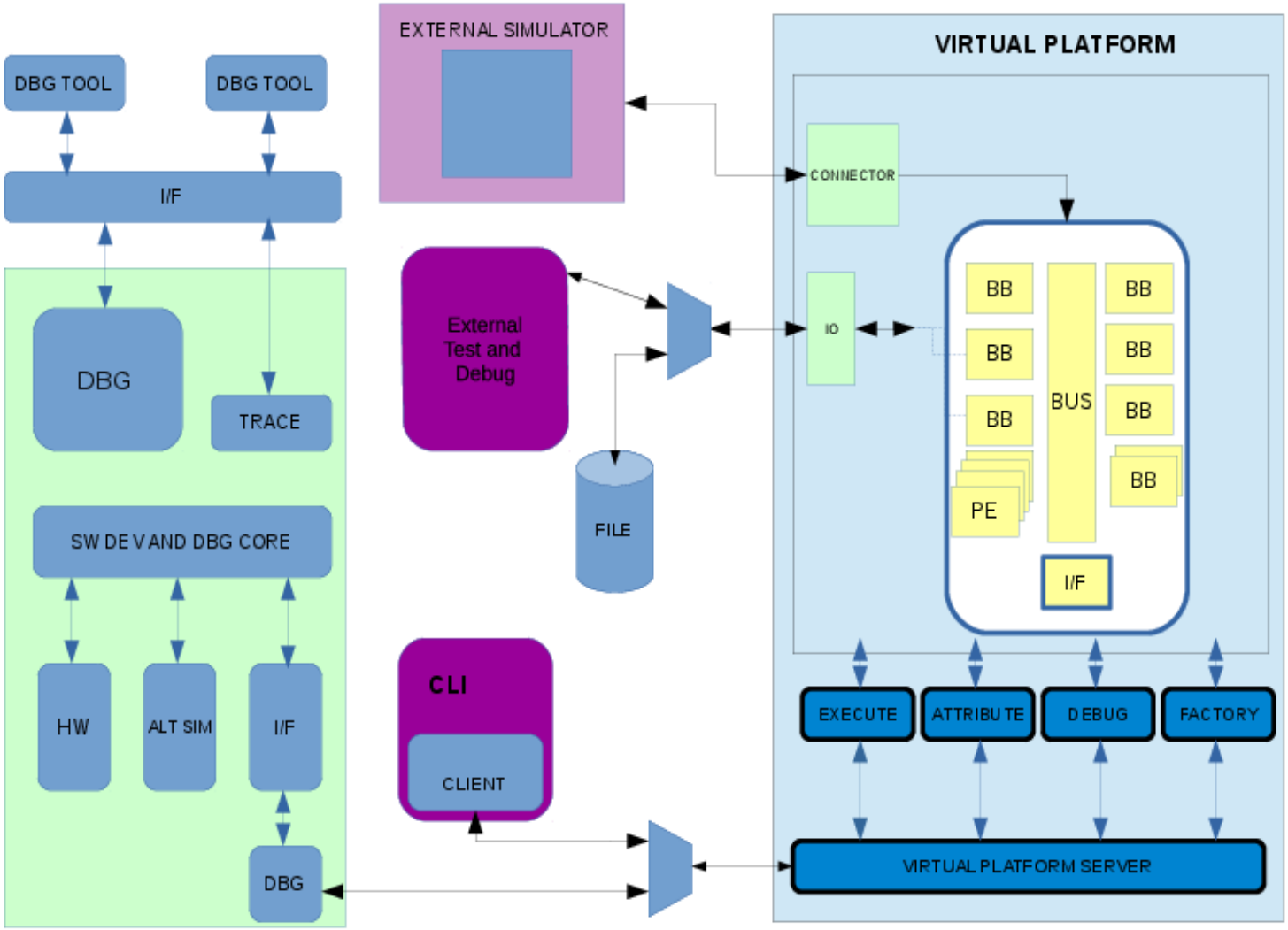
Experiences

- *HW specs are good, but SW User stories are also very useful* – finding the trade-off between hardware specifications and software user stories
- *Sometimes a register model with a selection of functional behavior is good enough, and sometimes we need full functionality (e.g. a complete signal processing algorithm)* – we use both kinds of models, in different use cases
- Some *models are used also in ASIC verification, others are not* – model updates must be compliant with requirements from SW and HW organizations

Platform Assembly



Platform Assembly



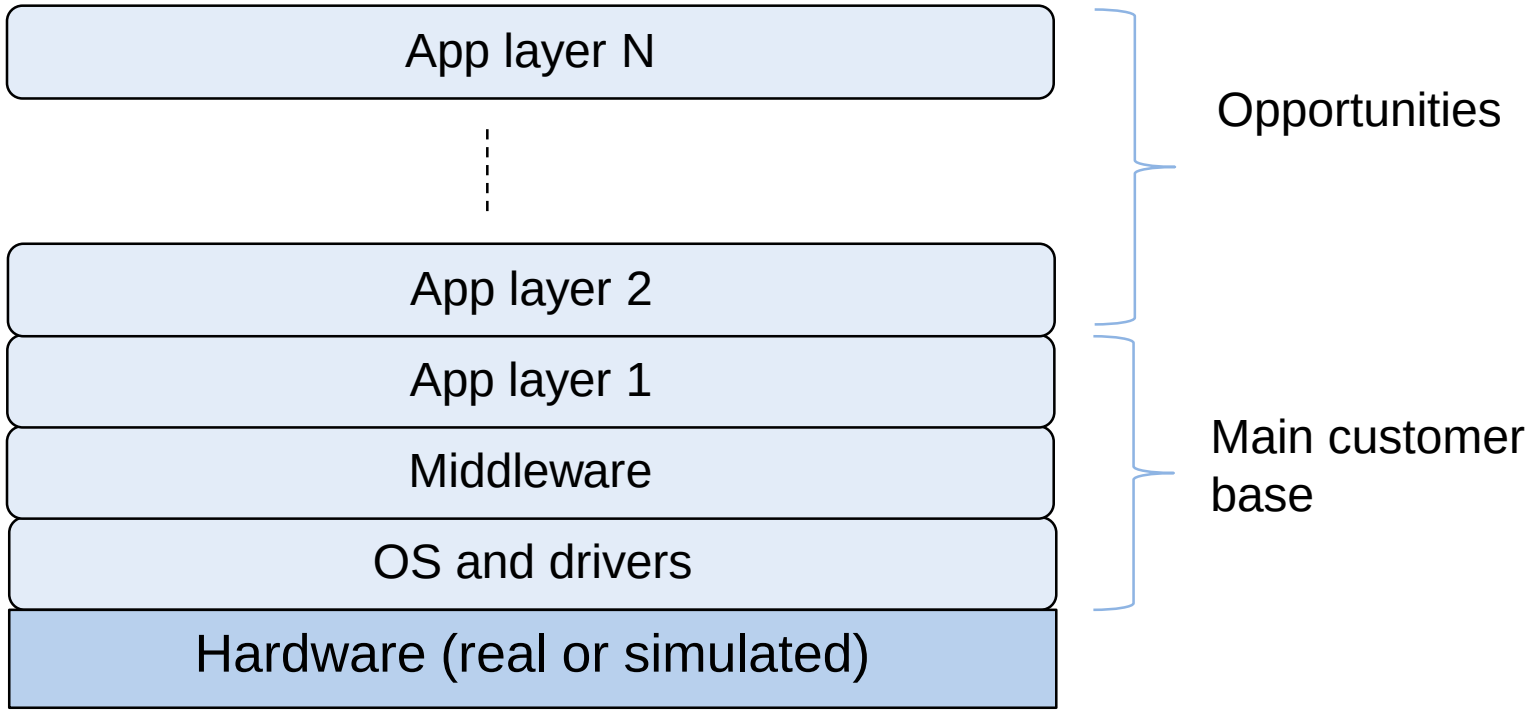
Challenges

- *We need to support several platform variants* – dynamic platform configuration (and parametrization)
- *How can we formulate modeling requirements for model makers?* – modeling guidelines, distributed to vendors, both in-house and external
- *We* need *to put the platform together, but we also need to interact with it (in many ways)* – architecture framework with well-defined interfaces for attaching software debuggers and other tools, e.g. test tools that are also used together with real hardware

Experiences

- *Dynamic platform building is useful* – we can create new platform variants quickly
- *A custom module class and associated configuration and control classes have helped a lot (and CCI will make it even better)* – we are active in the CCI working group
- *A common SW architecture was created, and it is still used* – this has been a fruitful investment

Usage patterns



Challenge: Coordinated multi-layer Virtual platform enablement

Challenges

- *Supporting Control plane and Data plane – on time and on budget* – we often start with control plane modeling, and add data plane models as the project proceeds
- *How to best prepare for ASIC bring-up and Board bring-up* – tight interaction with software organizations, find test cases that are important to run in a simulated environment
- *How to support SW development and integration also after the hardware has arrived* – supporting several software layers, inter-layer dependencies (in time and in functionality), adding VP functionality, increasing our test coverage

Experiences

- *Control plane is common, but well-defined data plane scenarios can be real game changers* – it is costly to simulate all functionality, and we have selected specific use cases where we do full data plane simulation
- *Many virtual platforms tend to stay around, also long after hardware is available*
- *A virtual platform that is integrated into the SW development flow can run and test new software, but it can also serve as gatekeeper for deliveries (tests must pass on the virtual platform before delivery is allowed)*

SW Development and Virtual Platforms

Debugging scenarios may involve software as well as (the simulated) hardware

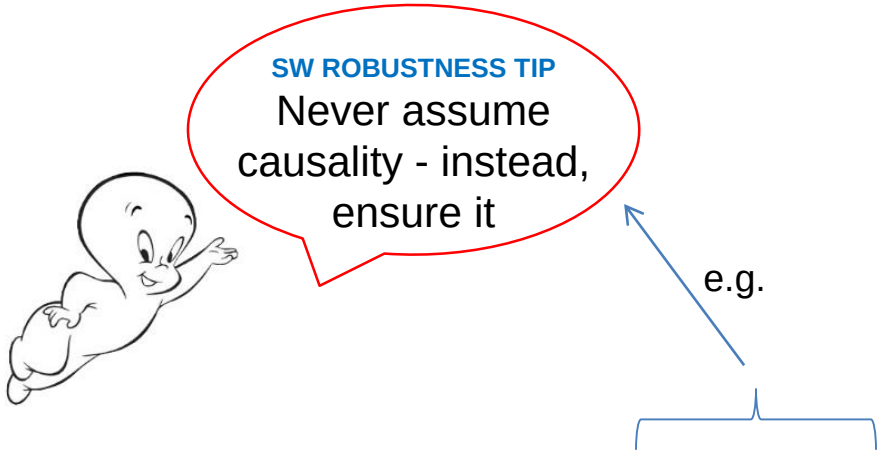
Was it a SW crash or a VP crash?

- Who misbehaved? SW or HW (i.e. the virtual platform)
- We have – over the years – gained significant experience in debugging, based on error reports from customers
- In many cases, we have found errors in software (and in several situations, the software has then been updated accordingly)

SW Development and Virtual Platforms

We have learned* that *software that is well-written from a concurrency perspective (e.g. do not assume that you know the relative speed of concurrent processes, do not rely on the precise duration of an activity)* – tends to work well on the virtual platform (and, of course, on the hardware)

In this sense, the virtual platform becomes a detector of unrobust ** software



* Discovered by VP team, and now communicated as design advice (from SW managers to SW developers)

** Software that most likely will fail when run on a hardware that is (sometimes only slightly) different

Challenges

- *Driver development is straightforward, but how can we support higher layers in the software stack*?
- *My software works on hardware but not on the virtual platform – what to do next?* – we have had many debugging scenarios where it has been difficult to see if a symptom is due to an error in software or in the virtual platform (or both) – and this is especially challenging if the software works fine on real hardware
- *I want to check my software performance, can I do that?* – we have a focus on functional modeling, but using time annotation in LT models, some aspects of performance measurements can be done

Experiences

- *Virtual platform developers often need to connect the dots – between the software layers and down to the hardware – especially when solving problems (is it a software crash or a virtual platform crash?)* – we experience this in our daily work with debugging
- *Timing-sensitive software does not run well on a virtual platform (it may run well on one specific hardware, but not on all hardware) – in this way, the presence of a virtual platform encourages robustification of software, which in the end should lead to higher software quality* – we see this as an additional selling point for a virtual platform
- *Timing annotation can give indications of performance, but cycle-accurate models and/or RTL is needed for more precise figures*

Conclusions

- From prototype in 2011 to multi-department production usage in 2016
- Following the TLM standard, active in CCI WG
- Virtual platforms for baseband and radio – approx 100K simulator starts daily
- ASIC simulators, Board simulators
- Early start of software development – bring-up of boot and OS approx. one year ahead of silicon
- Long-lived virtual platforms (several years after HW arrives) – used in regular SW development CI processes

Questions