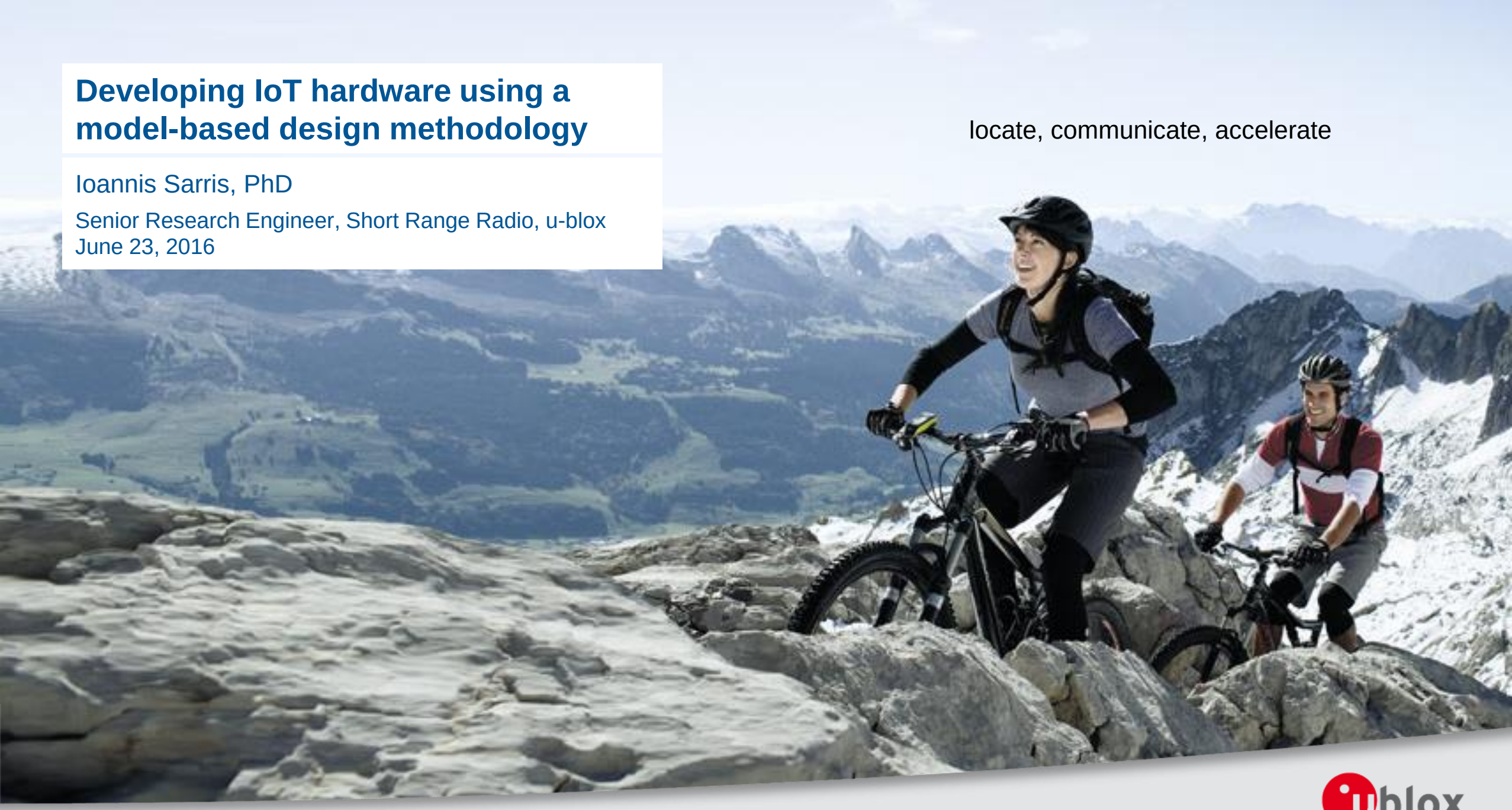
Developing IoT hardware using a model-based design methodology

Ioannis Sarris, PhD

Senior Research Engineer, Short Range Radio, u-blox

June 23, 2016

locate, communicate, accelerate



Key Takeaways

1. Model-based design offers
   - Quick time-to-market (initial prototypes in **weeks** instead of **months**)
   - Efficient use of resources (**up to 30%** less development & testing)
2. Good models outlast the life of a single product
   - Used throughout specification, implementation and validation phases
   - Reusable in similar-technology or next generation products
3. Accurate specification is crucial and requires
   - Careful planning
   - Previous experience

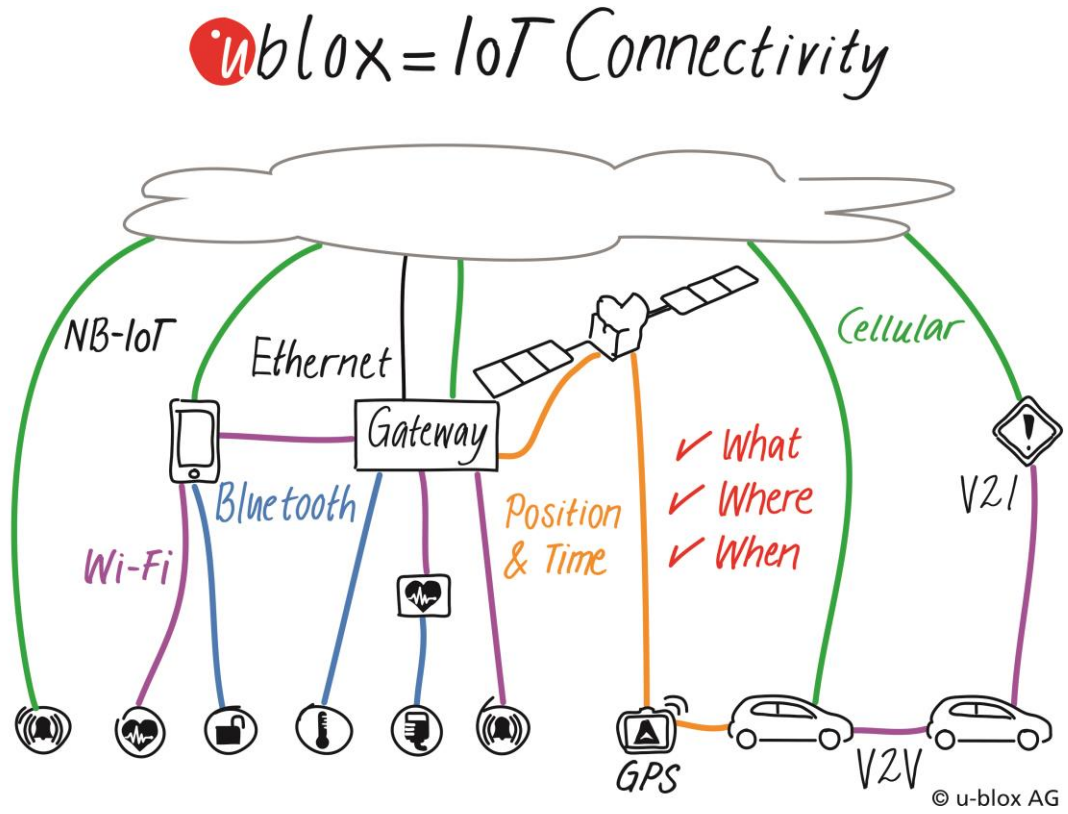
Introduction to u-blox

- Leading supplier of positioning and wireless semiconductors and modules
- Targets the automotive, industrial and consumer markets
- Our vision is to be the leading, industry-quality, supplier in the **Internet of Things that Really Matter**
- Headquarter is in Thalwil, Switzerland, Short-Range Modem development is based in Athens, Greece
- Focuses on Wi-Fi technology for IoT and vehicular communications

locate, communicate, accelerate

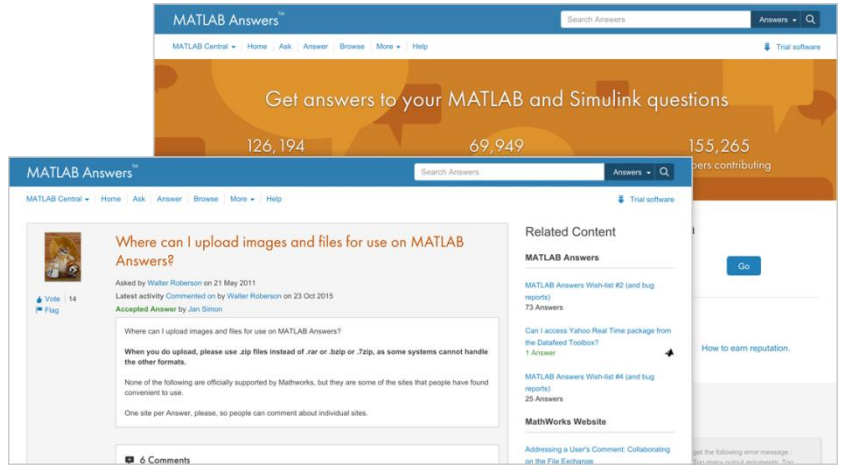
Innovation Challenges and Achievements

- A plethora of new connectivity technologies is rapidly emerging
- The key to success is to enable support with a competitive time-to-market
- A lot of common ground exists between these technologies, as well as many differences
- This is also true for different generations of same-technology products
- A model-based design methodology enabled us to leverage the experience from similar products



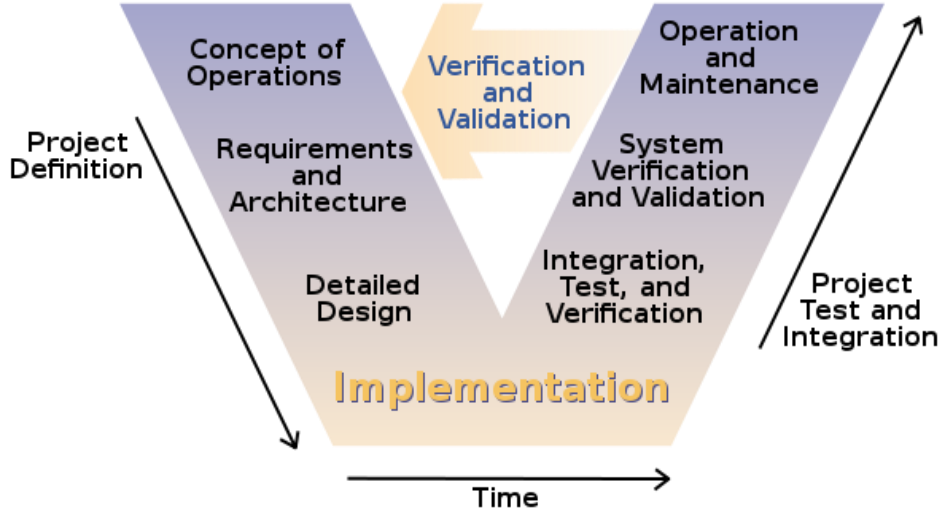
Involving MathWorks Products in our Development Workflow

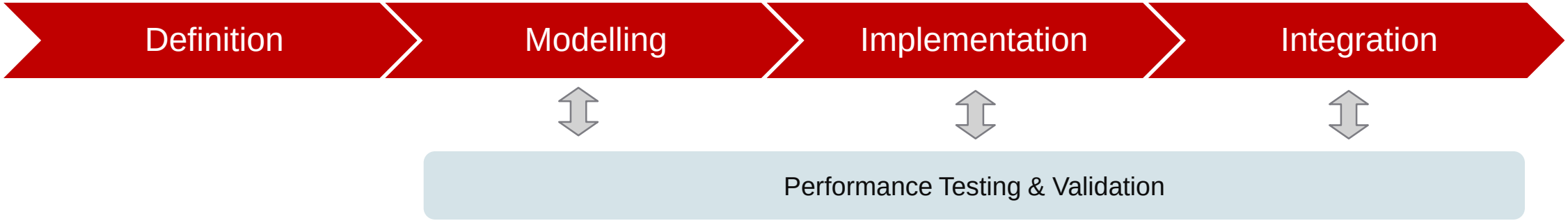
- MATLAB is used at the core of our model development
  - Engineers' familiarity with MATLAB
  - Rapid development time
  - Wealth of examples and community
- Additional training was required to use it optimally and meet the practical constraints of real hardware
- Once fully embraced, the use of a model-based design methodology allowed us to achieve exceptional time-to-market
- We were able to adopt different Wi-Fi "flavours" (from Wi-Fi 802.11n to 802.11ac, then to 802.11ah and 802.15.4 and now to 802.11p) each time, in a matter of weeks instead of months



Development Workflow

- Our development methodology is based on a 4-step process
- As in the V-model, progress is not linear, several re-visits to modelling might be needed
- Good definition ensures that these are limited in number and scale





Stage 1 - Definition

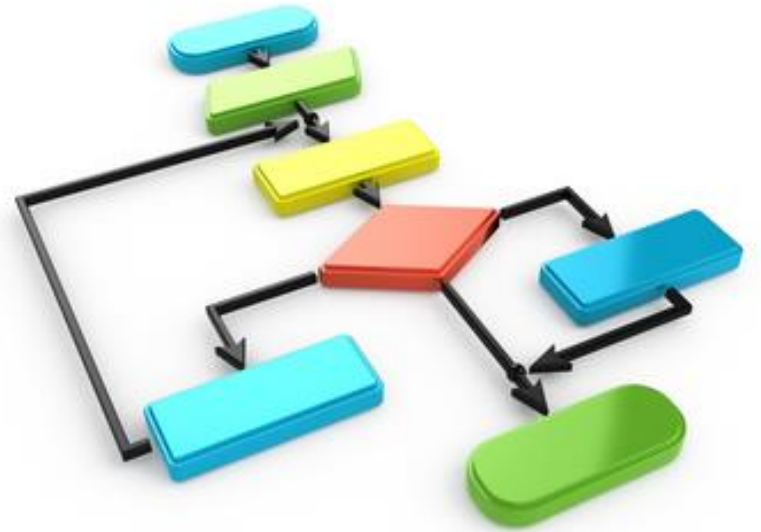
- Affects both the efficiency of the whole project and the quality of the final product
- The structure and operation of the system should be clearly determined by all teams involved in the development
- Sufficient time and effort should be spent to define the high and low-level architecture in as much detail as possible
- Performance targets and methods of validation should also be defined here

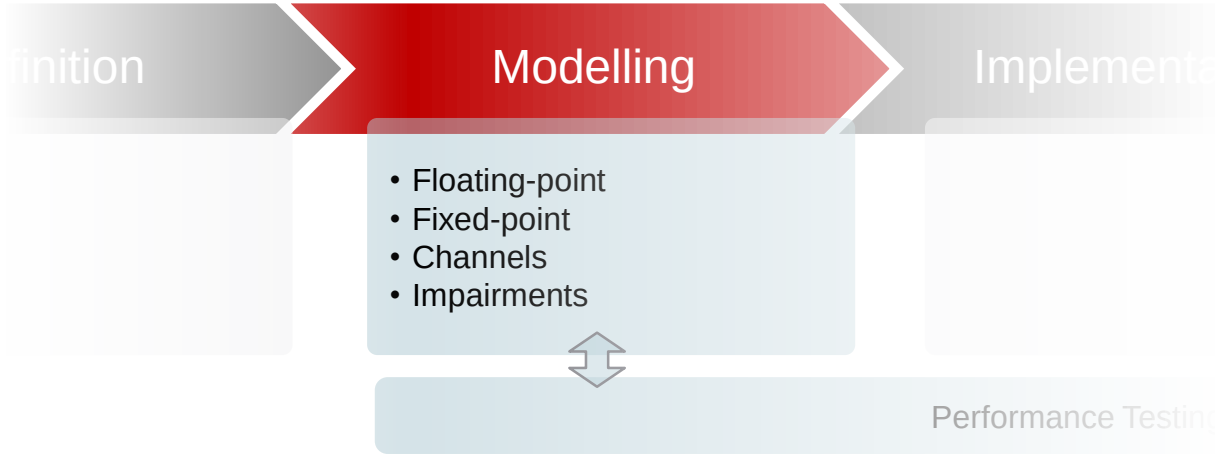
Definition | Modelling

- Requirements
- Protocol specifications
- Algorithm definition
- Feasibility study
- HW/SW partitioning

Stage 2 - Modelling

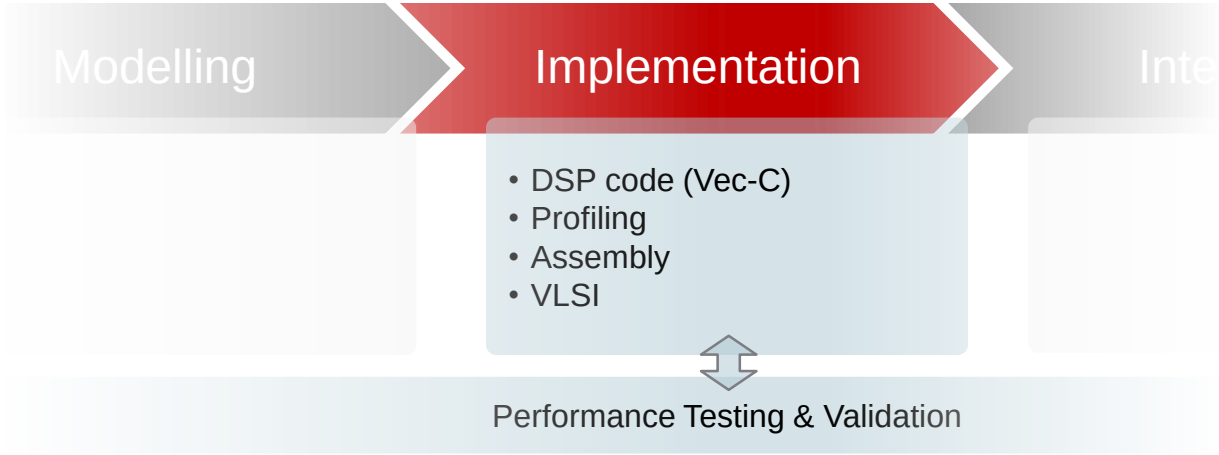
- Involves using MATLAB to implement efficient models which offer the required functionality
- Wherever possible, the use of existing Toolboxes alleviates the need for in-house development of non-core functionality
- Implementation of regression tests is necessary to ensure that performance targets are met
- Validation against 3rd party models is needed to ensure compatibility





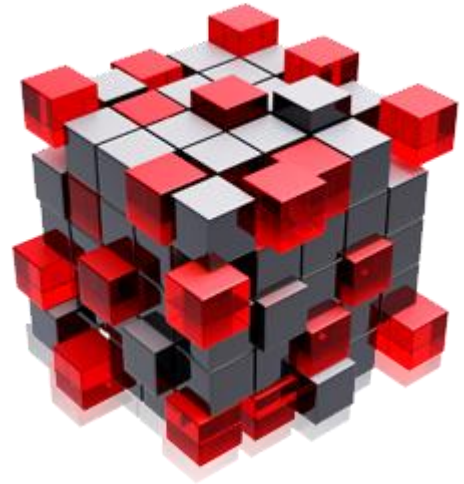
Stage 3 - Implementation

- The embedded software / hardware design teams use the MATLAB model as an **executable specification** for their implementation
- Matching the model with the implemented systems requires that the two are bit-exact under all conditions
- Co-simulation with HDL / DSP simulation makes it easy to pinpoint potential mismatches



Stage 4 - Integration

- The system components are integrated into a test platform
- Multiple test cases are triggered using a PC-controlled test environment
- Real-time tests are also performed with measurement devices (Signal Generators, Spectrum Analyzers, VSA software)
- Interoperability testing is done against 3rd party equipment



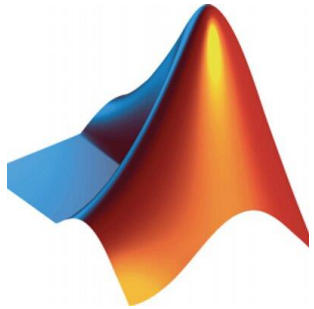
Implementation | Integration

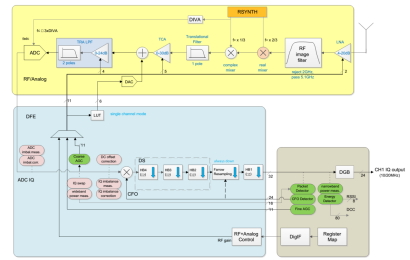
- Platform Integration (DSP / HWA / RF / Interconnections)
- Testing / Validation

Performance Testing & Validation

MATLAB Use Cases

- MATLAB is used in multiple aspects of our development
- A number of use cases are shown next
- These focus on
  - Core modem development
  - Testing / verification
  - Demonstration
- The practical challenges of each case and the solutions that we have applied are presented





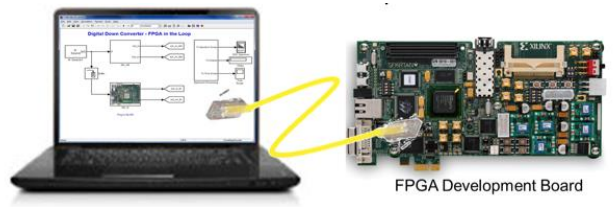
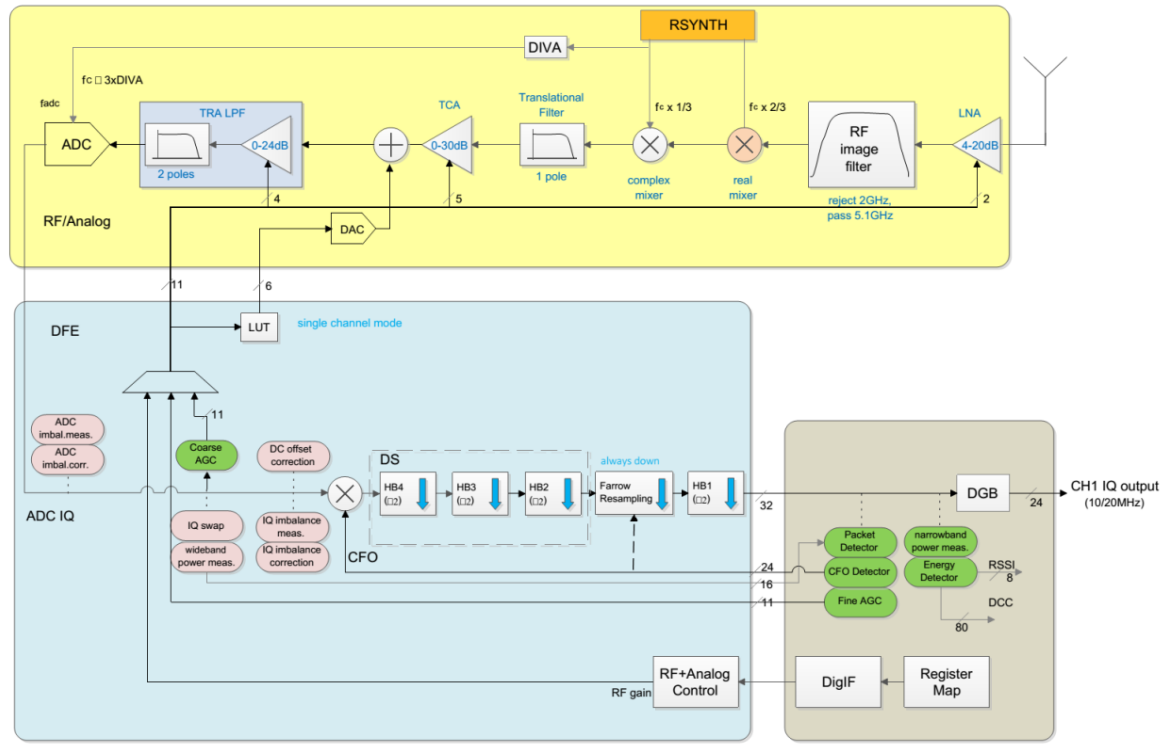


Figure 10. FPGA-in-the-Loop simulation using Simulink and FPGA hardware

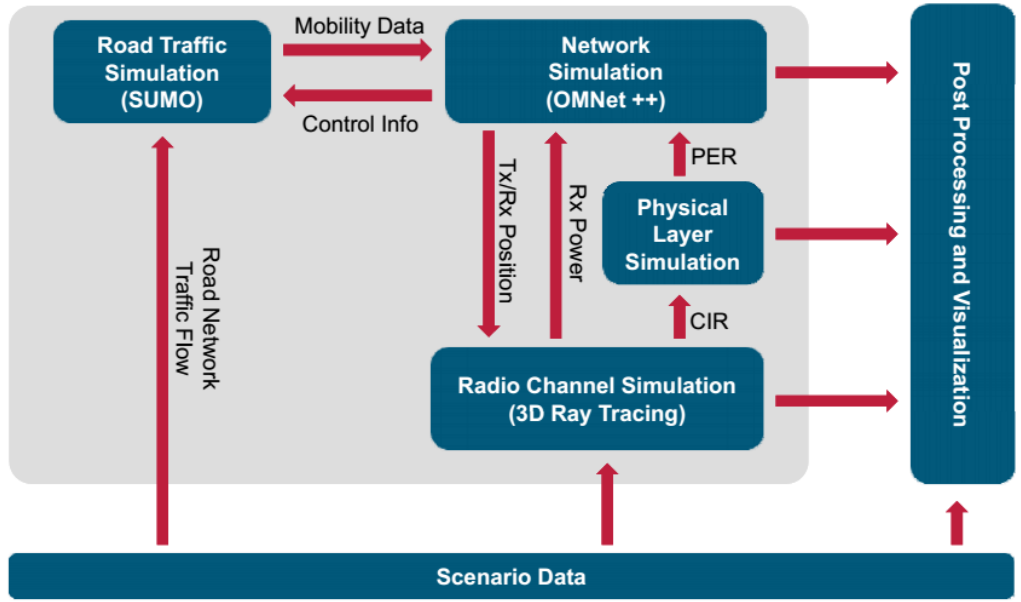
Case 1 - Sample-Based Processing

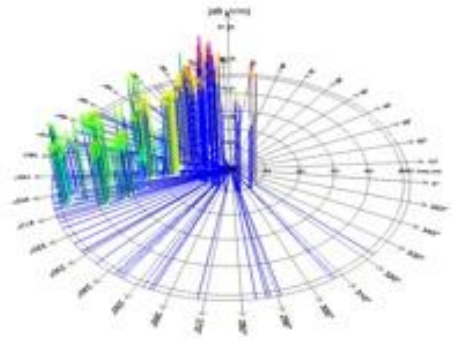
- Frame-based processing utilizes optimized routines for vector / matrix manipulations
- A serial communication processing chain cannot be fully modelled with frame-based methodology
- We have implemented a sample-based mechanism in MATLAB
- This does not utilize MATLAB's strengths in vector/matrix processing resulting in very slow simulations
- Code-generation was used to compile MATLAB code into MEX files
- Resulting speed increase of the MEX code was 100x faster than MATLAB code and even 4x faster than vector-based code

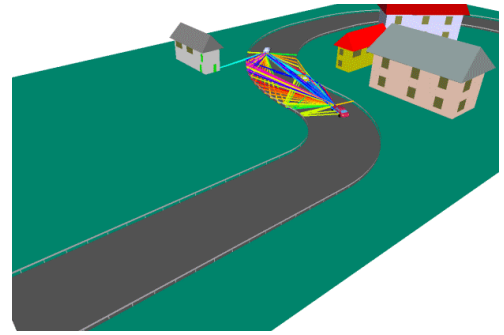


Case 2 - Co-Simulation

- Integrating multiple aspects of simulation recreates a more complete testing environment and gives better insight on “real-world” performance
- In the past, we have used co-simulation of
  - Dynamic channel modelling (ray-tracing)
  - Vehicular Traffic simulation
  - HDL Simulation
  - Other Golden Reference model
- Conventional approach involves offline and independent processing in each simulator
- MATLAB enables concurrent operation via built-in mechanisms

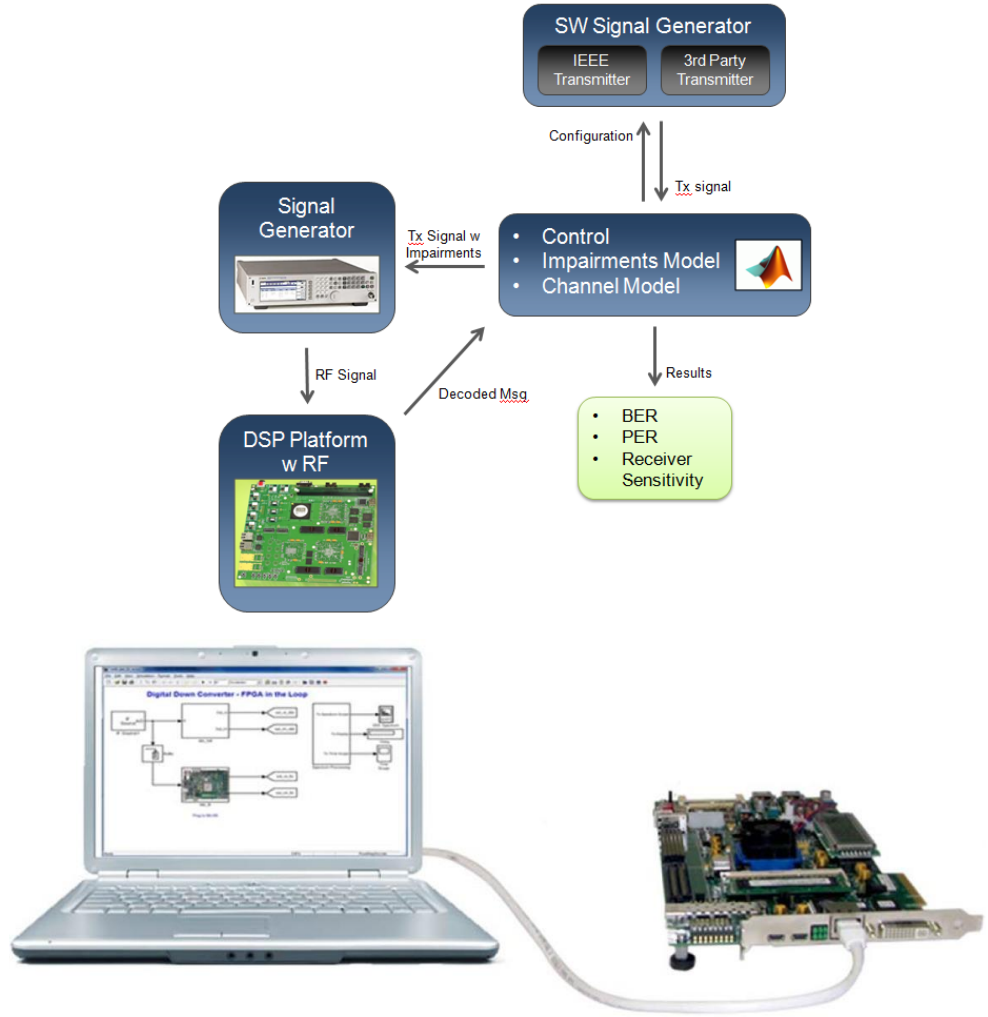






Case 3 - System Emulation

- Moving a step closer to the real system involves using real hardware
- This can be test equipment (oscilloscopes, signal generators, etc)
- It can also be hardware platforms (eg FPGA) which run the developed system
- The complexity of coordination of this equipment can be reduced using MATLAB's tools such as Instrument Control and Hardware-in-the-Loop
- System emulation gives maximum confidence on the developed solution

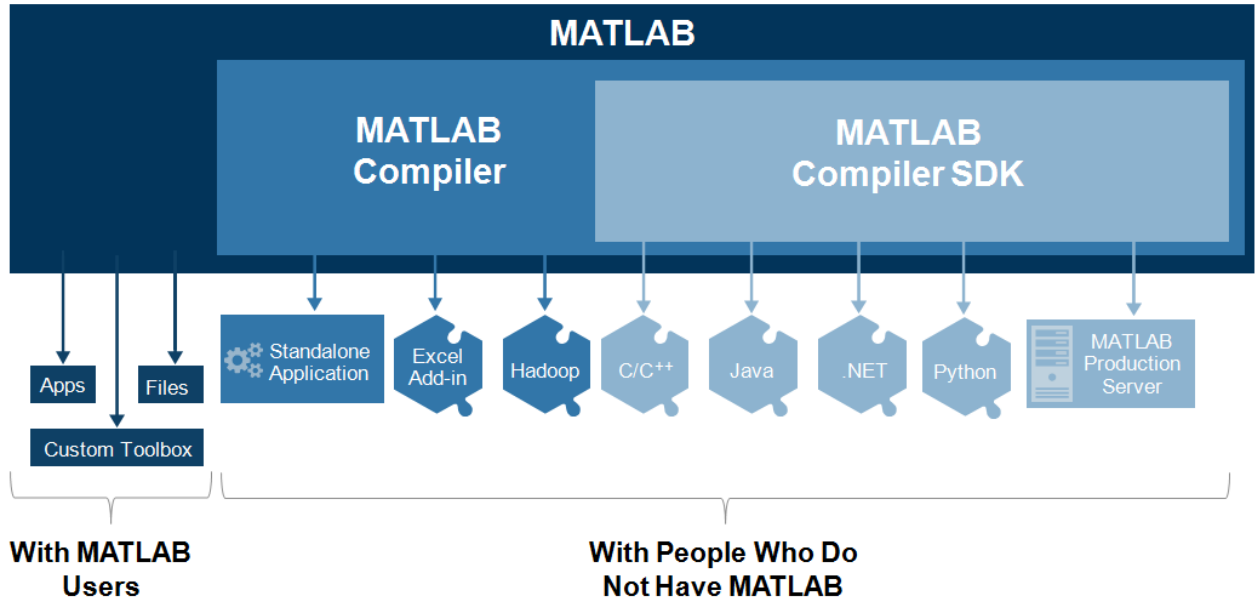


Case 4 - System Demo

- Real-time demos of our hardware could be easily implemented
- MATLAB is used for the control and visualization
- Hardware-in-the-Loop simulation is used to transmit and receive data
- Demo can be compiled for target PCs so that MATLAB installation is not necessary

Case 5 - Batch Simulations

- Sometimes exhaustive simulations are needed which require long simulation time
- A speed-optimized model is a good starting point
- The next step is to utilize the multi-core capabilities of recent PCs and the use of a pool of PCs
- An in-house solution was developed
  - A pre-processing script generates the starting conditions of each simulation
  - Simulations are distributed to multiple PCs
  - A post-processing script gathers all results
  - Multiple MATLAB instances could be used, even without a MATLAB license, by compiling the code using MATLAB Compiler



Concluding Remarks

- Model based design can help saving significant development time and effort (**up to 30%**)
- Compiled code resulted in a **100x execution speed up**
- Requires considerable effort in defining detailed specifications
- New tools/methodologies constantly emerge. A balance between following the bleeding-edge advances and focusing on core functionality should be found
- There is no “silver bullet” development workflow, the available options should be evaluated and choices should be tailored for the target application, team skills etc

Developing IoT hardware using a model-based design methodology

Ioannis Sarris, Senior Research Engineer, Short Range Radio, u-blox
ioannis.sarris@u-blox.com

June 23, 2016

locate, communicate, accelerate

Thank you!