Model-Based Design as an Enabler for Supply Chain Collaboration

Richard Mijnheer, CEO, 3T
Stephan van Beek, Technical Manager, MathWorks
Richard Mijnheer (1970)

- CEO and co-owner of 3T
- 3T (2014-now)
Founded in 1982, 3T since 1994
Management buy-out in October 2014
Development, manufacturing and support of **customer specific electronics and embedded systems**
40-45 employees (mainly MSc/BSc)
Offices in Enschede (HQ) and Eindhoven
Our mission

We continuously invest in our expertise of electronics and embedded systems to enable customers to supply perfect products.
Why invest in Model-Based Design?

- Systems are becoming more intelligent, more complex and have more and more electronics and software
- We believe Model-Based Design is a way to deal with this
Why partner with MathWorks?

- A lot of companies are already using MATLAB and Simulink e.g.
  - For algorithm development
  - For modeling dynamic mechanical behavior

- Has code generation in place for c/c++ and VHDL

- Is willing to collaborate with us to help our customers become successful
Benefits of Model-Based Design

- Faster innovation
  - Design errors are visible in an early stage
  - Continuous verification in a model-based simulation including hardware-in-the-loop
  - No programming errors due to code generation
  - Hardware choices can be delayed
  - Risk mitigation due to fast iterations
  - Higher quality, flexibility and shorter time-to-market

- Better collaboration
  - The model is the specification
  - Impact of requirements are clear very early
  - Change requests can be implemented quickly
Why collaboration leads to innovation in the supply chain?

- We are not the domain expert: the customer is
- We are not the tooling expert: MathWorks is
- The customer and MathWorks are not the electronics and embedded software expert: we are
- Model-Based Design helps to bridge the gap between these worlds resulting in better collaboration and innovation
Brake box for wafer handler robot
Problem definition:

Design a braking system that accurately and safely stops the robot in case ANY of the control components fails to avoid major damage to the machine.
SCARA robot system

- High power robot control
  - 2 high power (>250V hazardous) 3 phase amplifiers
  - Complex motion control platform
  - High resolution encoders
  - A number of low and high voltage power supplies
  - Many interconnects any communication interfaces between these parts

- Any component failure could cause a lot of damage

Challenge:
- Design a braking system that accurately and safely stop the robot in case ANY of the control components fails.
- Within ~mm accuracy (without encoders >> sensor less)
- At ~m/s speeds
- Within a < 0.5 s timeframe
- No external additional support for measurements
Design approach

- Dynamic robot Simulink model was already available
  - Very complex on the inside
  - Very usable by the other engineers on the outside!

- High level model of electronics added
  - 3 phase motors
  - Conceptual actuators and sensors

- Control (closing the loop)
  - Perfect to test different control algorithms
  - Effective control algorithm quickly emerged

- Decided to go for a FPGA solution using HDL code generation for the control
Results

- Final design functions with little tuning needed.
- But……
  - New requirements pops up:
    - Maximum allowed deceleration is dramatically decreased
    - Hardware hits its limits, causing some critical tests to fail. No easy hardware fixes available.
- Now what?
  - After initial cheers project heads towards failure!
  - Hardware redesign is costly in this phase
Results

- PWM style switching was implemented in “golden reference” to mitigate hardware limitation.
- Code generation and testing was highly automated and proceeded very quickly.
- The new requirements met by a very fast design iteration of the HDL logic.

Lesson learned: Fast iterations can be a life safer!
Project example

Radar tracking module
Problem definition:

Develop a new radar tracking module which can

- Track 6 lanes instead of 3
- Discriminate vehicle types better
- Adopt changes in the algorithms due to new circumstances
Design approach

- SensysGatso designed the radar tracking model in MATLAB
  
  ![Diagram](image)

  - Radar Tracking Model (MATLAB)
  - Radar Model
  - 2D Traffic Data
  - Dec.
  - Image Processing Model
  - FFT 1
  - FFT 2
  - Tracking Algorithm
  - Intermediate Result Files

- MathWorks helped to optimize model for code generation and advised on decomposition in FPGA vs CPU
- 3T developed the electronics, FPGA firmware and a framework for the generated code
Radar tracking module

- High performance analog front-end for signal conditioning radar signals
- System on Module with Xilinx Zynq Z-7020
- C/C++ code for radar tracking algorithms generated via Model-Based Design using MatLab and Simulink
- Digital Signal Processing in Zynq FPGA fabric
- eCos RTOS on one Zynq ARM core

See: [http://3t.nl/algemeen/soc/](http://3t.nl/algemeen/soc/)
Results

Marco Siebeling, Manager Research, SensysGatso:

- The lead time of the RT4 to a workable product was significantly shorter than with previous radars, roughly 50%
- We have now the possibility to record raw data and to play it back into the model which reduces test time significantly
- From MATLAB algorithm to an integrated RT4 test version takes less than half an hour
- Model based design helps the impact analysis of change requests
- Model bases design helps to efficiently localize problems
Conclusions after using MBD
Conclusions

- The project results show that Model-Based Design helps to

- Enhance innovation:
  - Shortens lead time
  - Helps to adapt to changing requirements through faster iterations
  - Improves the quality by early simulation using hardware-in-the-loop results

- Enhance collaboration
  - Improves communication between different disciplines using the model as the specification
  - Faster impact analysis of changes
  - Helped us to get very satisfied customers