Model Based Design approach for HVDC applications

MATLAB-EXPO, Munich 2014-07-09

Markus Engel, Siemens AG, E T TS PLM C&P
Co-Authors: Sebastian Behnisch, Dr. Christian Siegl, Christina Zenk, Marcos Pereira, Kevin Römer
Agenda

- Current situation
- Challenges
- Goals & Constraints
- Solution
- Conclusion
- Outlook
Current situation
Turnkey solution business & customers with high buying power
We have to deal with manifold contributions – see hawker’s tray of C&P

Control & Protection System

... and a lot more...
Challenges:
System & Software engineers work on parallel paths …

Electro-Magnetic Transients (EMT) simulation for development & optimization of
- control parameters & algorithms
- steady state & dynamic performance
- protection coordination

Software development in Step7/CFC (graphical & block-oriented language) & C/C++ / FPGA applications for
- closed loop control
- open loop control
- protection and measurement
Challenges: … with the same objects developed in different tools and they are dependent from each other!

EMT simulation needs
- Control & protection algorithms
- Communication delays
- …. 

Software development needs
- Parameters / structure for control algorithms
- …. 

Susie’s Simulation
- Electric network models & control algorithms
- Closed loop algorithms
- Algoirthms for...
- open-loop
- closed-loop
- Protection

Steven’s Software

Sebastian’s Software
Goal:
One platform independent Model Based Design tool for all targets

Constraints:
- Different automation platforms to be supported
- EMT simulation program PSCAD is state-of-the-art in HVDC business (often PSCAD models have to be delivered to customers)
- System & software engineers work in different departments
Which integration platform fits best?
MATLAB/Simulink seems to be a feasible approach for HVDC

<table>
<thead>
<tr>
<th>Criteria</th>
<th>PSCAD</th>
<th>CFC/TDC</th>
<th>MATLAB/Simulink</th>
</tr>
</thead>
<tbody>
<tr>
<td>integration in software development suite</td>
<td>-</td>
<td>-</td>
<td>++</td>
</tr>
<tr>
<td>rate of coverage of development lines</td>
<td>--</td>
<td>-</td>
<td>++</td>
</tr>
<tr>
<td>test/commissioning support</td>
<td>o</td>
<td>++</td>
<td>o</td>
</tr>
<tr>
<td>independency from automation platform</td>
<td>o</td>
<td>--</td>
<td>++</td>
</tr>
<tr>
<td>working in distributed teams</td>
<td>o</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>know-how protection</td>
<td>o</td>
<td>o</td>
<td>+</td>
</tr>
<tr>
<td>fulfillment of single-source principle</td>
<td>-</td>
<td>-</td>
<td>++</td>
</tr>
<tr>
<td>life cycle costs</td>
<td>o</td>
<td>o</td>
<td>+</td>
</tr>
</tbody>
</table>
Model Based Design approach using MATLAB/Simulink & code generation with Embedded Coder

Model-Based Design Workflow

- RESEARCH
- REQUIREMENTS
- DESIGN
- IMPLEMENTATION
- INTEGRATION
- TEST & VERIFICATION
- TEST SYSTEM

- PSCAD
  - OS1
- TDC
  - OS2
- MATLAB/Simulink
  - OS1
  - OS3
- DSP
  - OS3
- ...
  - OSx
MATLAB/Simulink & PSCAD
Approach is feasible, but some issues to be investigated!

Main approach:
- Unchanged use of C-code produced automatically by Simulink
- Generation of lib files to be used by PSCAD

Optimal workflow to be developed with PSCAD users:
- Accessibility of parameters
- Avoid jumps between Simulink and PSCAD as far as possible

- Fortran wrapper for interfacing with PSCAD necessary
- Benchmark: Comparison of simulation results of Simulink-based block and original PSCAD block without deviations
MATLAB/Simulink & Simatic TDC: Referenced models enable variant management, distributed engineering and efficient version control

One main model → link to multiple and hierarchical referenced models

- Implementation of different variants (e.g. different closed loop control structures) of one referenced model possible (same external interfaces)
- Integration with version control tools (e.g. SVN or GIT) possible
- Referenced models enable distributed engineering
- Clearness due to hierarchical structure
MATLAB/Simulink & Simatic TDC: Reuse of well-proven TDC library functions by C-macros and use of Legacy Code Tool

Integration of existing macros in Simulink:

C-Source file: hvdc_ccp_lib.c
H-Header file: hvdc_ccp_lib.h

Use of Legacy Code Tool:
- Generates the following files:
  * .mex file for simulation
  * .tlc file for Simulink Coder
- Generation of library for Simulink

Existing function blocks which should be re-used
MATLAB/Simulink & Simatic TDC:
Automatic C-code generation for use in TDC function block generator
Conventions defined and workflow tested

referenced model „CRPC“

Conventions:
- Block names Simulink = TDC connector names
- Signalnames in Simulink = comments in TDC
- Single task within one model
- Same datatypes

Additional connectors compared to original:
- Version of Simulink model
- Enable/disable input
- Failure output
- Internal variables output
MATLAB/Simulink & Simatic TDC:
Runtime of Simulink generated function blocks is at least 50% less than original TDC function blocks by applying feasible granularity of packaging.

Performance advantage of Simulink-generated function blocks can be reached by packaging of suitable number of function blocks within one Simulink model.

Stateflow suitable for modelling of plant sequences.

1:1 conversion of standard function blocks from Simatic TDC to Simulink not feasible.
MATLAB/Simulink & Simatic TDC:
Simulink concept of constant values and signal busses is useful for parameterization of functions

- All parameters are handled within separate referenced models, input e.g. via .csv file
- Distribution to „consumers“ via signal bus
- Existing procedures / tools for parameterizing remain as they are
MATLAB/Simulink & DSP
Benchmark against interface-identical software of customer project positive

Critical issues:
• Code efficiency (due to very fast control cycle within µs-area)
• Proof of accurate dynamic performance

Benchmark executed in the following steps:
1. Implementation and test of developed algorithms in MATLAB/Simulink
2. Benchmark between automatic generated Simulink-based software and (plain C) software of customer project
   a) in offline simulation environment (PSCAD)
   b) in realtime test environment

Benchmark Results:
- Control accuracy & dynamic performance nearly identical
- Runtime performance of Simulink-based control 14% better than original C-code
Conclusion

- Overall feasibility of approach is proven and shows significant advantages!
  - platform independency
  - performance
  - savings in project engineering

- Closed loop control algorithms migrated into Simulink models and tested

- „One click solution“ for generation of TDC- and PSCAD-lib’s
Outlook

- Migration and test of open-loop and protection algorithms
  - Use of stateflow
  - Simulink based (automatic) component test setup
  - Integration test in realtime simulation environment
  - Benchmark of runtime behaviour against original software

- Integration of different components' models
  - Enable PC based integration tests

- Evaluation of Simulink's EMT simulation capabilities
Thank you for your attention!

markus.engel@siemens.com