euspen's 16th International Conference & Exhibition, Nottingham, UK, May 2016
www.euspen.eu

Enhancement of the Mechatronic Development Process with Software in the loop Simulation

Karin Fåhraeus, Jonas Walther

Mycronic AB, R&D Mechatronics, Sweden

karin.fahraeus@mycronic.com, jonas.walther@mycronic.com

Abstract
This paper describes a concept for working with a model based approach in the development of embedded mechatronic control systems. The focus has been to find an independent method with low overhead and low maintenance that enables modeling and simulation in multiple phases during the development cycle. The work started as a Master Thesis [1] and the concept of a software in the loop simulation has been implemented and evaluated as a proof-of-concept in a pick-and-place system.

Software in the loop, Mechatronics, Model-based development, embedded system, simulation

1. Introduction

Model-based development (MBD) is very common in embedded system projects for control applications and refers to control- and plant models and simulations [6]. The methodology of model based development usually starts with Model in the loop (MIL), then Software in the loop (SIL), Processor in the loop (PIL) and finally Hardware in the loop (HIL) [2,3,4]. Today there are a lot of different tools for implementing MBD, but they also have some drawbacks. It usually requires a different software development environment than the existing one that may introduce a high learning threshold. They can be target specific meaning that they only support some processors and one becomes dependent of that software development environment. Sometimes extra hardware is also needed.

A PIL simulation is implemented for the embedded system on a pick and place machine, where the control system code runs on the actual target processor and the plant model is represented by mathematical equations in a C-function. The plant model refers to the dynamics of the system and is determined by the physical properties of the system. This simulation mode is dependent on target hardware, it is hard to debug and it takes a long time to compile and download the code to the target.

For software in the loop simulations, the plant is modeled but the control is executed in a low level language, such as C, where the simulation runs on a desktop computer [2]. An analysis of how to implement an independent SIL solution has been conducted and the investigation resulted in a prototype SIL simulator. This SIL solution has been developed as a proof of concept for one of the axis in a pick and place machine. The aim is to examine the advantages and opportunities this solution brings to the development process of an embedded system.

2. SIL solution as proof of concept

The development cycle of a mechatronic system for a pick and place machine has been evaluated and currently, Model in the loop and Processor in the loop are implemented. To enhance the development process a SIL solution prototype has been developed.

2.1 MIL and PIL

Model in the loop is used in the beginning of the development, where both the control system and the plant are modeled in Simulink. At Mycronic, a lot of the code is reused during development where in that case, MIL can be excessive to implement. The embedded code can also be executed and tested in a PIL simulation that has made the testing of the code partly independent of the hardware and given a chance to test the code before running it on the real machine.

2.2 Implementation of SIL

The control systems code running on the embedded system is modified so that it can be compiled with a gcc compiler and is executed on a desktop computer. This is accomplished by identifying target specific code and in some cases replacing the code with corresponding functionalities that are available on a desktop computer. In other cases the functionalities are simulated. The interface to the plant model is implemented with TCP socket communication. The plant model is implemented as a mathematical representation in a C-function or in Simulink. However, the plant can easily be replaced by a model in another program giving a modular and flexible solution. The overview of the SIL solution compared to the real machine can be seen in Figure 1. Here the SIL environment can be seen with the control software running on a desktop computer, sending the control signal of current to the plant model which will return the calculated encoder position through TCP socket communication. For the real system the system software is running on a Linux computer communicating with the embedded system through a CAN network. The embedded system controls the real plant through hardware IO. In Figure 2 the Simulink part is shown in more detail with an S-function handling the socket communication retrieving the control signal and then sending back the position.

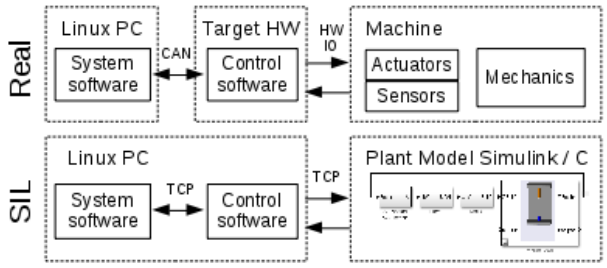

Figure 1. The real system compared to the implemented software in the loop simulation.

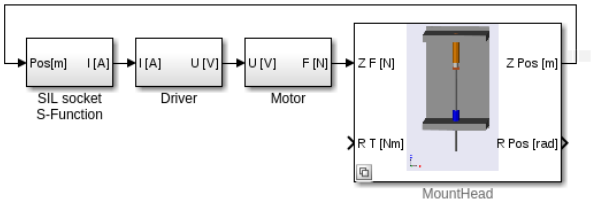

Figure 2. The plant model in Simulink with a SIL socket in an S-function that handles the communication with the control application code.

2.3 Different ways of running the code

The plant model can be switched between variants depending on the situation. During software functionality development a fast executable and lightweight model can be used giving a fast debugging turnaround. During control system development a more realistic model can be used. When doing tests and development, there are now 5 different ways of running the code according to:

1. The MIL simulation implemented in Simulink/ SimMechanics.
2. SIL simulation with the plant model as a mathematical representation in a C-function.
3. SIL simulation with the plant model in Simulink/SimMechanics.
4. The PIL simulation with the plant model as a mathematical representation in a C-function.
5. The real machine or real hardware prototypes.

This gives a high flexibility to select the right tool for different situations.

3. Result

This SIL environment can be used when developing the code for the control system and the code can be tested without the real plant hardware. One of the big advantages with SIL is that it is not dependent of either electronics or mechanics and it gives an opportunity to execute and test the code and the control before it is integrated with the target processor, e.g. in early design phases. It is also useful in sustaining development when hardware in not available.

An advantage with using the SIL compared to MIL is that it is the real code that is running, which makes it closer to reality. It is common for MBD to auto-generate code from MIL to SIL [3, 4]. This might be good when developing systems from the beginning, but not as good in incremental development when a lot of code is reused. It is more efficient to use the existing and reusable code in a SIL environment.

The SIL provides the possibility to check whether the controller design is well executed and satisfying. Problems and errors that might arise can be caught early with the SIL simulation. The detection and elimination of errors early will contribute to have the code more thoroughly tested.

Mechatronics is a synergistic combination of mechanical and electrical engineering, computer science and control engineering [5]. In designing of a mechatronic system it is very important to include these interactions and consider their impact on each other. The SIL simulation includes the interaction of the control design, mechanical design, electrical design and software design which is a great advantage.

The SIL simulation contributes with an opening to run other system software together with simulations of the embedded system on a desktop computer. This facilitates development of the system software and enables more complete regression testing.

The SIL simulation will be useful throughout the whole development cycle. It enables the use of standard software development tools like debugger, profiler, code coverage and test tools. This reduces the dependency on target specific tools that can be limited or have a high learning threshold. It also makes it easier to migrate to other target hardware. If the embedded software is designed for this version of SIL with an abstraction of the target specific code and an abstraction of other hardware accesses the implementation and maintenance requires a relatively small amount of work.

4. Conclusion and future work

A SIL simulation concept has been developed and evaluated as an enhancement to the development process for complex mechatronic control systems. A benefit from using MBD and SIL is that the systems behavior can be predicted and the interaction with mechanical, control, electrical and software design can be evaluated early. The concept will improve the work flow and quality in all phases of the development cycle with low overhead work effort once it is up and running. It enables the use of open source software tools and it is flexible so that it can be used regardless of target system. Another advantage is that it is independent of a specific software development environment and therefor also suitable for smaller companies and organizations. With this approach of using the new SIL simulation the code will be more thoroughly tested and errors can be found in the beginning of development. The SIL simulation contributes with faster development.

The next ongoing step is to implement the SIL simulation in full scale for a complete system. As the implementation is today, the SIL does not include any of the target specifics, for example numerical properties or timing. The possibility to include target specifics in the SIL simulation will be examined in the next step. There is an ongoing work to implement the same concept of SIL simulation for Mycronic's high precision lithographic mask writers. An evaluation of the SIL simulation concept used in a development project needs to be executed.

References

[1] Fåhraeus, K 2015 Enhancement of the Mechatronic Development Process with Software in the loop Simulation, An embedded control case study

[2] Bouissou, O Chapoutotm A 2012 An Operational Semantics for Simulink's Simulation Engine *Proceedings of the 13th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, Tools and Theory for Embedded Systems (LCTES 2012)*

[3] Shokry, H Hinchey, M 2009 Model-based verification of embedded software *Computer* 42

[4] Jaikamal, V 2009 Model-based ECU development – An integrated MiL-SiL-HiL Approach *SAE Technical Paper*

[5] Fotsom, A.B Wasgint, R Rettberg, A 2012 State of the Art for Mechatronic Design Concepts *International Conference on Mechatronics and Embedded Systems and Applications (MESA)*

[6] Bhatt, D Hall, B Dajani-Brown, S Hickman, S Paulitsch, M 2005 Model-based development and the implications to design assurance and certification *Digital Avionics Systems Conference (DASC)*