

MSC.Software VPD Conference | July 17-19, 2006 | Huntington Beach, California

An Alternative Approach to Testing Embedded Real-Time Software

Rich Mather
Jan Matlis
Draper Laboratory





Abstract

One of the major challenges facing the engineering community is integrating embedded software controller(s) into a complex electro-mechanical system. In the early phase of design this problem is typically addressed with proto-type software running on breadboards in an open loop environment. This leaves a limited number of software developers waiting for available hardware which is also evolving through the hardware design process. Later in the design process software engineers are provided a closed loop testing environment using real time hardware in the loop. Many functional integration issues are discovered at this time and may be intermittent and difficult to repeat. Secondly, the software debug environment will introduce some level of intrusion which may cause system behavior to diverge from the behavior of the system without the debug environment.

A simulation of software execution behavior can be provided virtually by using an Instruction Set Simulators (ISS) of the target processor. The Simulation Department at Draper Laboratory has invested several years of development into a simulation framework that can combine continuous time domain simulation and event driven simulation of digital electronics and Instruction Set Simulators. This framework provides software developers with a non-real time simulation environment to execute target code running closed loop on a dynamic system created from purely behavioral truth models. The advantages with this virtual system methodology are;

- Easily deployed and scalable to a large software development community geographically dispersed.
- Provide software development environment earlier.
- Provides a stable and repeatable software debug environment.
- Software debug capability without any intrusion.
- Reasonable ratio of simulated real time to wall clock time when compared to other purely virtual solutions.

The challenges we have recognized and addressed are not unique to Draper Laboratory. Draper will present this methodology, current results, and suggest a vision for the future. In the long term, Draper would like to see companies like MSC provide integrated solutions using Commercial off the Shelf (COTS) software products like the ISS.



Presentation Outline

- Vision – Task Charter
- Problem definition
- Ground rules for our design
- Problems we were able to solve
- Conclusion and future vision



Vision - Task Charter



Vision – A customer of Draper tasked the Lab to develop a collaborative Modeling and Simulation capability to support design/evaluation of future systems.

“No future system design shall be considered valid until it’s proven in the virtual system simulation.”

Task Charter - To develop and support a *framework* allowing for rigorous modeling and simulation of future systems.



Problem Definition

- **The Virtual System Simulation (VSSim) team was tasked to assemble a virtual simulation environment for testing real-time embedded software that...**
 - Can be easily deployed and scalable to a large software development community geographically dispersed.
 - Have checkpoint restart capability with operating point manipulation.
 - Execute with a reasonable ratio of simulated real time to wall clock time when compared to other purely virtual solutions.
 - Be available earlier than hardware breadboards.
 - Provide a stable and repeatable software testing environment.
 - Provide visibility with minimal intrusion



Design Ground Rules

- **The solution should be 100% virtual**
- **Should maximize use of Commercial Off The Shelf (COTS) software**
- **Should include Instruction Set Simulators (ISS) that are...**
 - Commercially available
 - Offer instruction accurate execution of target code
 - It was required to execute the actual software rather than a model of the software
 - Execute at a reasonable simulation speed
- **Should use variable step integration methods to solve the plant models**
 - Many of our validated plant models contain discontinuity in their behavior
 - Variable step integration methods provide performance advantages over a fixed step methods with a very small time step



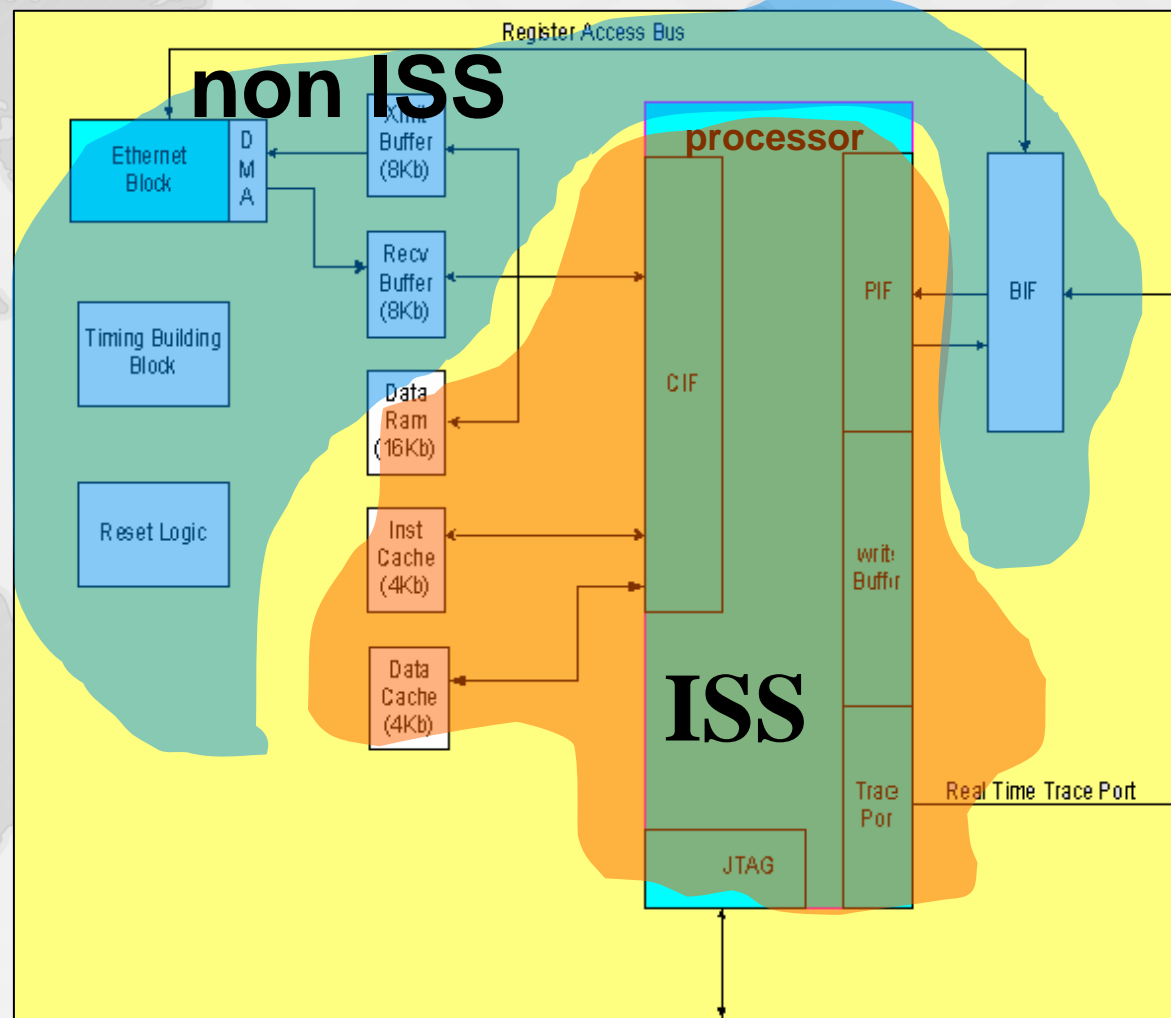
What our method addressed

- A methodology that enabled an event driven simulation like an ISS to synchronize with the variable step integrator of MSC Easy5 was developed
- **Because the electronics design process would not produce high level behavioral models that would execute within our simulation framework, it became the responsibility of the simulation task to produce models of digital electronics that would work with the ISS**
- It was necessary to create a simulation middleware that allowed us to link to an ISS running as a separate external process
- Because model development of plant dynamics and algorithms were done in the MATLAB Simulink environment, the simulation task needed to port these models into the Easy5 environment to be solved by a single central integrator
- It was necessary to utilize MSC Easy5 sorting capability to execute all of the pieces (Simulink models, Fortran code, C code, the ISS) in the correct order



Simulation of Digital Electronics

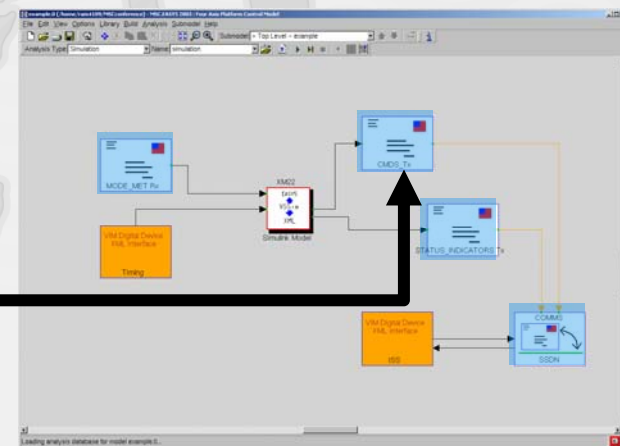
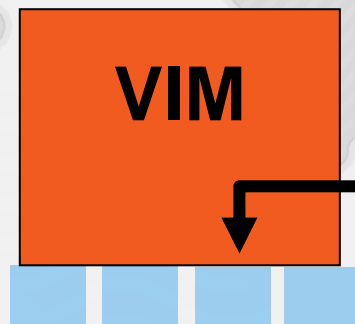
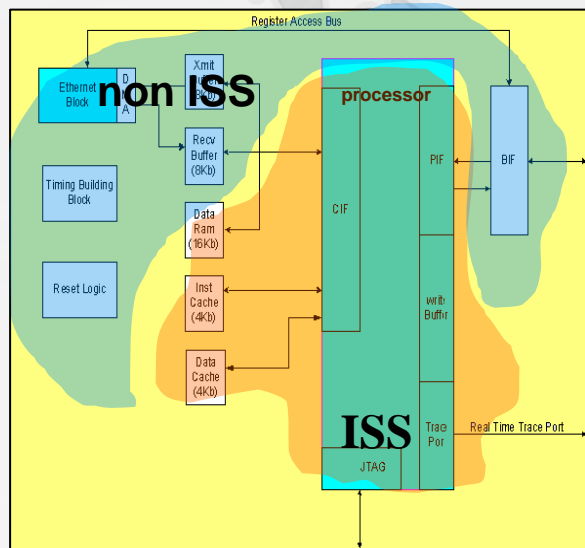
- The electronic design process would not produce high level behavioral models that would execute within our simulation framework
- The ISS out of the box supplied a processor and memory models
- The simulation task produced models of digital electronics that would work with the ISS





Simulation of Digital Electronics

- The simulation task created individual custom models of digital electronic components (timing, communications, messages)
- Interfaces exist as graphical blocks in Easy5 that could be interconnected
- The VSSim Infrastructure Middleware (VIM) interfaces Easy5 with the custom models of digital electronics





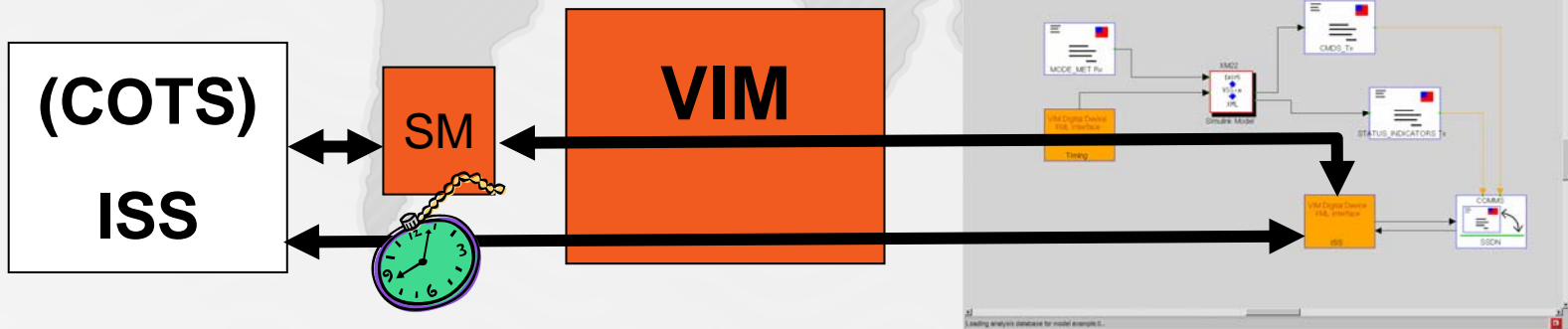
What our method addressed

- A methodology that enabled an event driven simulation like an ISS to synchronize with the variable step integrator of MSC Easy5 was developed
- Because the electronics design process would not produce high level behavioral models that would execute within our simulation framework, it became the responsibility of the simulation task to produce models of digital electronics that would work with the ISS
- **It was necessary to create a simulation middleware that allowed us to link to an ISS running as a separate external process**
- Because model development of plant dynamics and algorithms were done in the MATLAB Simulink environment, the simulation task needed to port these models into the Easy5 environment to be solved by a single central integrator
- It was necessary to utilize MSC Easy5 sorting capability to execute all of the pieces (Simulink models, Fortran code, C code, the ISS) in the correct order



Incorporating the ISS

- ISS is Commercial Off the Shelf (COTS) software
- The VSSim Infrastructure Middleware (VIM) interfaces Easy5 with the ISS
- A shared memory block allows data to be passed between Easy5 and the ISS
- Semaphores are used to synchronize time between Easy5 and the ISS



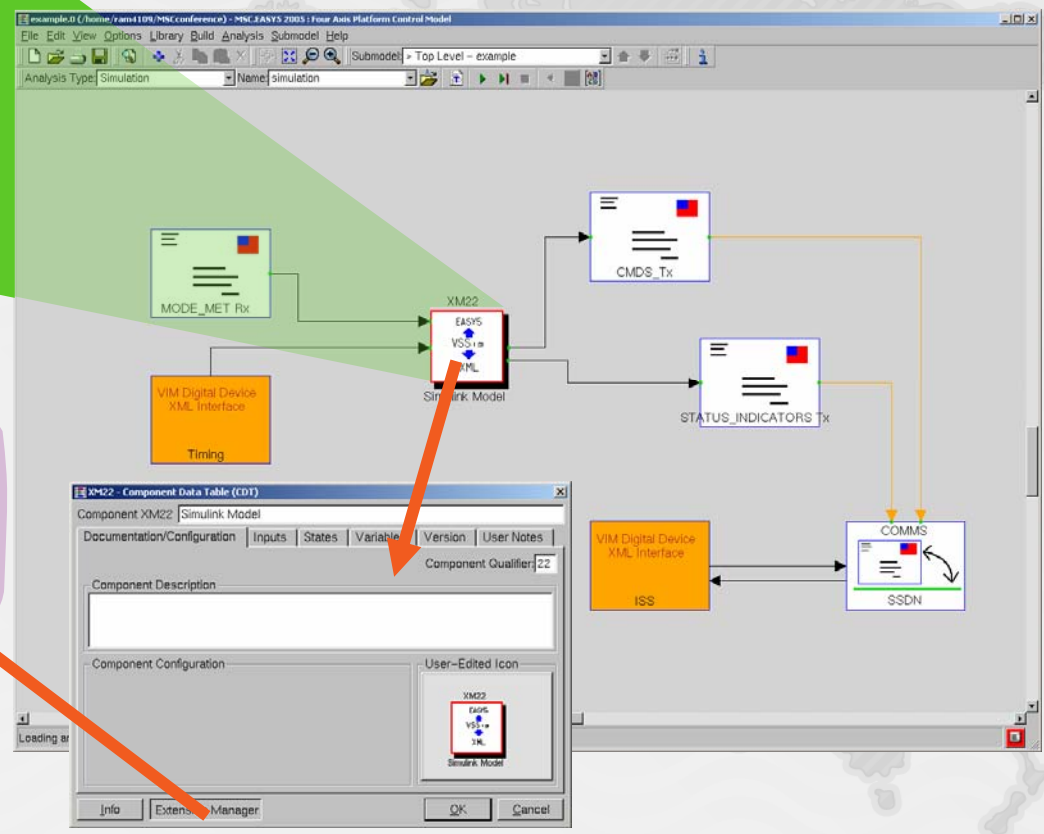
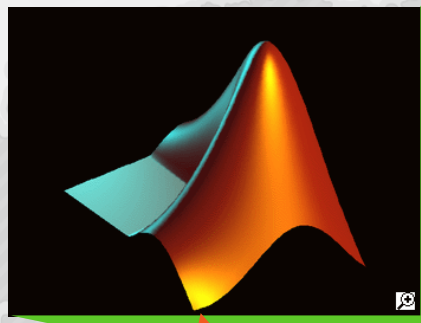


What our method addressed

- A methodology that enabled an event driven simulation like an ISS to synchronize with the variable step integrator of MSC Easy5 was developed
- Because the electronics design process would not produce high level behavioral models that would execute within our simulation framework, it became the responsibility of the simulation task to produce models of digital electronics that would work with the ISS
- It was necessary to create a simulation middleware that allowed us to link to an ISS running as a separate external process
- **Because model development of plant dynamics and algorithms were done in the MATLAB Simulink environment, the simulation task needed to port these models into the Easy5 environment to be solved by a single central integrator**
- It was necessary to utilize MSC Easy5 sorting capability to execute all of the pieces (Simulink models, Fortran code, C code, the ISS) in the correct order



Porting Models from Simulink



Extension Manager

Selected Extension Type: Simulink

Selected ICD: [Browse... Edit...]

Selected Simulink Model: [Browse... Edit...]

Compile Options:
 Compile with Debug

Accept Update Component Close

Extension Manager started successfully Message Log...

Simulation infrastructure produced a linkable version of the Simulink model
 MSC Easy5 variable step integrators provided central integration of all models

Conclusions and our Future Vision for MSC Easy5



- The challenges we have recognized and addressed are not unique to Draper Laboratory.
- We believe this methodology is a viable commercial product for MSC to develop and market
- Should partner with a provider of Instruction Set Simulators
- Unlike our approach the methodology should allow digital models to be developed with an Electronic Design Automation (EDA) industry standard such as SystemC™ to execute within the simulation framework