

Virtual Engine System Prototyping with High-Resolution FFT for Digital Knock Detection Using CPU Model-Based Hardware/Software Co-simulation

Yasuo Sugure and Shigeru Oho
Hitachi, Ltd.

Sujit S. Phatak and George Saikal
Hitachi America, Ltd.

Copyright © 2009 SAE International

ABSTRACT

We have developed a full virtual engine system prototyping platform with 4-cylinder engine plant model, SH-2A CPU hardware model, and object code level software including OSEK OS. The virtual engine system prototyping platform can run simulation of an engine control system and digital knock detection system including 64-pt FFT computations that provide required high-resolution DSP capability for detection and control. To help the system design, debugging, and evaluation, the virtual system prototyping consists of behavior analyzer which can provide the visualization of useful CPU internal information for control algorithm tuning, RTOS optimization, and CPU architecture development. Thus the co-simulation enables time and cost saving at validation stage as validation can be performed at the design stage before production of actual components.

INTRODUCTION

Mechatronics control systems as typified by automotive control systems are becoming increasingly complicated due to requirement of low fuel consumption and a strict regulation of emission. Some recent automotive control systems, such as engine control, throttle control, transmission control, and brake control have their own microcontroller modules, that communicate with each other via vehicle network such as CAN [1]. In developing complicated automotive control system, the system

developers have issues related to increasing validation time and costs.

One of the solutions for this issue is to use model-based development method without using actual components. In the system design level, many companies have used models developed using MATLAB®/Simulink® in earlier design stage to verify the plant design and control algorithms. However, the controller model is treated as a "black box" because it is highly abstracted to behavioral level. In an actual controller module, embedded microcontroller chips and the peripheral IC chips operate to execute object codes including real-time operating system such as OSEK OS and device drivers to handle I/O functions. In order to verify automotive control systems close to more realistic system, verification in the detail level by using object code is required.

For the verification in the detail level implementation, there are many approaches to use hardware-software co-simulation [2, 3]. However, we have proposed to use a detailed CPU-model into model-based mechatronics design process [4-6] in earlier development stage. By using this approach, system designers can do concurrent development to validate the software programs including OS and device driver as well as the mechatronics design. Another benefit for using CPU model is the capability of non-intrusive controller behavior analysis. HILS and emulator with actual evaluation board have less capability of monitoring CPU internal information due to a limitation of output pins for debugging compared to use CPU model. Moreover, our approach enables to change

a configuration of microcontroller easily. The system designers can investigate the microcontroller performance as well.

In this paper, we have developed a virtual full engine system prototyping with digital knock control that provided a heavy processing load to the CPU-model and evaluated CPU utilization by using the CPU behavior analysis tool.

POSITION OF CPU MODEL-BASED

The typical V-cycle development process and our CPU model-based development concept are shown in Figure 1. The vertical and horizontal axes show the abstraction level of the design object and time, respectively.

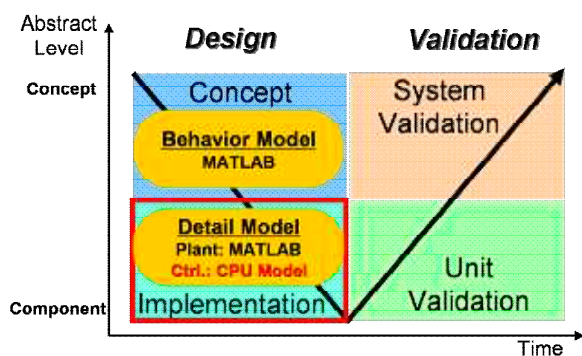


Figure 1 V-cycle process

It can be roughly divided into four stages:

- Concept design
- Implementation design
- Unit validation
- System validation.

The first stage is concept design, which is the domain of application developers. In this stage, they design system functional requirements working on simulation with higher behavioral level models. The second stage is implementation design, where the design implementation is done at the architecture level. In this stage, the software developers map the functions developed in the previous stage into real-time operation as object code generation from behavioral level control model. After two design stages, validation is started. At the unit validation and at system validation stage, developers work on component level verification and system level verification. To save time and cost, validation using models is required at design stage. Today, many companies have validated system behavior by using MATLAB®/Simulink® etc.

Our proposal is focused on the "Implementation design" stage, where every component of the system is designed with the modeling tools in detail level. This includes device driver for peripheral modules in microcontrollers and RTOS configuration. The validation of their behavior

is important as well. Software programs except for application which is enabled to validate at concept design stage, have a possibility of containing some bugs. Moreover, developers can validate whether timing for real-time control meets the design requirements at implementation stage itself. If timing critical issue occurs, this design and verification may be iterated many times to find out the optimal design. To complete validation at design stage, detailed models are required. CPU Model-based design method plays a very important role in this validation process.

MICROCONTROLLER MODELING

We developed a co-simulation environment, which can operate virtual full engine system using detailed SH-2A microcontroller model. In this section, the microcontroller modeling for SH-2A is described.

SH-2A is the latest 32-bit microcontroller in SuperH RISC engine family from Renesas Technology. The SH-2A has a dual-issue superscalar structure with pipelines at three- or five-stage, for instruction processing with low latency, 2MB internal flash memory, 128KB internal SRAM, 37channel 12bit resolution A/D converter, advanced timer unit (ATU), CAN, and SPI (Serial Peripheral Interface) for external communication. It can run at 200MHz. This controller chip is a targeted high-end microcontroller for automotive powertrain control.

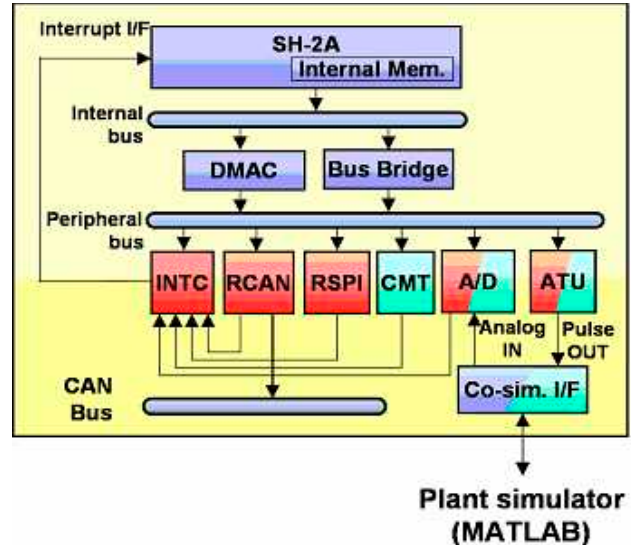


Figure 2 SH-2A model for CoMET™

The SH-2A controller model was built on the CoMET™ simulator platform. Figure 2 shows block diagram of the CoMET™ SH-2A model. CPU core and buses were provided from VaST Systems Technology, and the peripherals were developed later in accordance with our software architecture.

CPU BEHAVIOR ANALYZER

The primary objective to apply the CPU model into the model-based design is to analyze and validate the microcontroller hardware and software. We have developed CPU behavior analyzer to visualize CPU internal information. This section describes CPU behavior analysis feature integrated in the VaST CoMET™ environment for SH-2A.

CoMET™ has an optional interface to obtain the trace data from the models called Metrix as shown in right side of Figure 3. Metrix includes information about instruction execution, CPU register read/write, memory load/store, and every signal in peripheral modules. However, Metrix information appears to be too much from the perspective of the control system designer. Therefore we developed the CPU behavior analysis tool to validate CPU behavior easily. This tool analyzes the Metrix stream and displays the results during the simulation simultaneously. The tool consists of function analysis, interrupt analysis, PC trace, and value monitoring features.

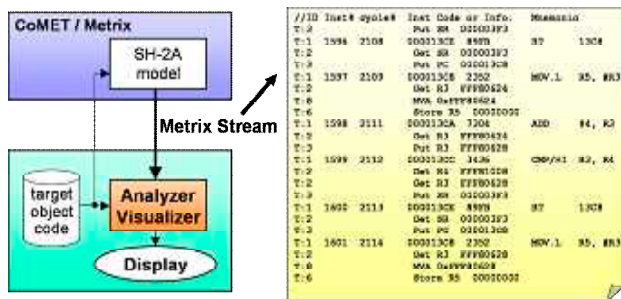


Figure 3 Metrix and CPU behavior analysis

To investigate CPU utilization, we use the task/function analysis which is one of the tool functions as shown in Figure 4. The result from task/function analysis includes calling times, instruction execution numbers, cycle, and CPU load ratio in every task. We can analyze the cycle performance and CPU utilization in complex control systems such as engine control, electronic throttle body (ETB) control, and knock detection and control based on the results obtained from function analysis.

Task/Function Analysis

Performance info. in each task
- # of call, cycle, & instruction
- load ratio



Cycle performance

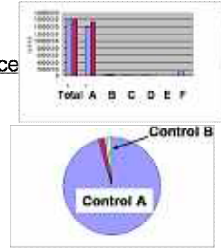


Figure 4 Results from Task/Function analysis

VIRTUAL FULL ENGINE SYSTEM

The virtual full engine system implemented in this activity is shown in Figure 5. As observed it consists of four major model blocks that were implemented in MATLAB®/Simulink® platform. They include the 4-cylinder engine model, the sensor model, the knock generation model, and the microcontroller model. The 4-cylinder engine model and the sensor model together are called 'plant model'. The microcontroller model is implemented as an S-function block into MATLAB®/Simulink® platform.

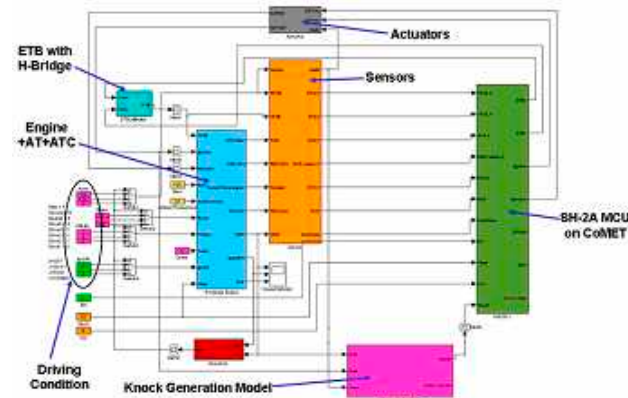


Figure 5 Virtual full engine with knock generation and detection system

PLANT MODEL

Engine Model

This block is labeled as the 'Engine+AT+ATC' in Figure 5. A 4-cylinder engine model was built on MATLAB®/Simulink® platform. It also included the Automatic Transmission (AT) controller. It was implemented as a generic model that could support up to 8 cylinders and its ports were defined by typical numerical I/Os and not by actual engine specification. For example, the injection pulse width was specified as a millisecond value. In the similar fashion AT controller did not generate torque but RPM and vehicle speed depending on the road condition and AT shift position signals.