Embedded Hardware/Software Verification and Validation using Hardware-In-the-Loop Simulation

Syed Muslim Shah  
University of Engineering & Technology  
Taxila, Pakistan.  
syedmuslimshah@hotmail.com

Muhammad Irfan  
University of Engineering & Technology  
Taxila, Pakistan.  
irfan29wah@yahoo.com

Abstract: Embedded systems verification and validation practice emphasis on exhaustive worst case analysis. In this paper the authors have described procedures of an extended Hardware-In-the-Loop (HIL) simulation setup testing process for the verification and validation of Embedded Hardware and Software. Extended HIL setup architecture including real time system, controller hardware and data acquisition setup for real time data receiving has been focused. The paper presents time synchronization of real time system running simulation models and the controlling computer executing control logic. Real-time control parameters extraction procedures have been covered which is used for analysis and comparison so as to verify control logic implementation. FLASH based method for hardware and software verification is explored and the use of hardware in loop technology for testing and verification of controller interfaces is presented.

Keywords: hardware-in-loop, control logic, simulation, time synchronization, real time, parameters extraction.

1. Introduction

Safety critical systems mostly employ embedded computers as central controlling part due to its programmability which is desired for keeping these systems intelligent. Correct verification and validation analysis is highly required to ensure the healthy performance of the system and also to assist in finding out errors and to determine future trends that would indicate a possible failure.

HIL allows to test embedded software or attach prototypes or final electronic components, to test design against a real time dynamic environment without the expense of testing using the real plant[1]. HIL enables embedded software testing at much earlier stage of the development process when errors are easier and less expensive to correct. It also reduces the risk of testing prototype controllers on expensive or dangerous system hardware such as aircraft or satellite systems[1].

System functions are developed and implemented using modeling (which is mathematical representation of plant systems), simulation and code generation software packages and development tools like MATLAB /SIMULINK/STATEFLOW [2] and Microsoft Visual C. These tools provide a graphical way for describing functions and systems. Using automatic code generation from models, the build code is downloaded on a powerful real time system. This results in reduction of testing time without any compromise on quality [1]. These real time systems have available special input and output (I/O) interfaces which are connected to the actual plant for commands actuation and sensor input [3].

The generated code is implemented with system function builder which is further optimized so as the code must be highly efficient and error free. The embedded software is tested from desktop simulation to over software in the loop down to hardware in the loop; the generated software components should have identical behavior at all steps. The first step of the verification process is called model in-the-loop simulation (MIL). It captures the specified behavior of the model by recording block output and block state data. Software-in-the-loop simulation (SIL) is the next step. Code is generated and compiled with a host compiler and executed in the same simulation environment. Code that runs correctly on the PC can still cause trouble on the target processor. Therefore, the final checks need to be done with hardware-in-the-loop simulation (HIL).

The testing procedure helps in verification of low level platform software and the proper functioning of hardware [4]. System failure conditions are tested by inserting various disturbances and changing the input parameters. Human Machine Interface (HMI) software in the form of Graphical User Interface (GUI) is developed which is used for diagnosing the controller hardware, its interrupts, I/O and clock. Various profiles are given to the controller and the output signals from the controller are acquired and analyzed.

This paper is organized as follows: Section 2 gives more about the validation process. An extended HIL setup architecture has been described in section 3. Hardware testing with HIL is discussed in section 4 while section 5 is focused on real time system and controller hardware synchronization. Section 6 describes real time parameter extraction required for control software verification and analysis. Non-model based verification is the topic of section 7 while concluding remarks are drawn in section 8.

2. Validation process

Validation means a provably correct, efficient procedure or exhaustive simulation and testing demonstration [5]. Embedded system design requires functional specifications, hardware & software architectures, implementation of hardware/software components and finally the validation process. The Control system algorithm development follows the process of system dynamics models building, design, analysis and
Algorithm development consists of modeling software components integrated with handwritten software modules. After the logic is verified in a desktop computer simulation, the corresponding embedded control algorithm code is ported on the proposed platform and downloaded on the microcontroller. Then a Hardware-In-the-Loop is performed to verify the control logic and embedded system hardware interfaces [6]. The controller function is implemented as a system function of the model and the software is developed in C language. The model with controller function is integrated into one model and the software is tested in non-real time. The software is then prepared for real time according to real time system requirements. The final code is compiled, build and downloaded on the real time system and the simulation is run in real time (software in the loop).

In parallel the controller functionality is implemented for the actual embedded controller with low level coding for interrupts setting, serial communication and input/output interfaces. As the controller application program has to communicate with the external environment in a timely fashion with hard deadlines on the execution time of such software, it is ensured that all real time constraints are satisfied. This requires worst case execution time (WCET) analysis of software [7]. Finally the controller function is disconnected in model and its inputs are given to actual controller system running standalone and the outputs are fed back to the real time system thus creating hardware in loop setup.

3. System architecture

HIL is a very popular technology for verification of embedded systems. We have extended the standard HIL procedures and have proposed real time parameters extraction. The procedure helps in analysis of control variables during execution time. The embedded software test environment consists of a powerful real time system machine. This machine can be either the host PC or some other stand alone dedicated real time system communicating with the host PC through some high speed communication link like fiber optic or network cable. These real time systems are capable to run the model simulation software at required sample time.

The system function algorithm is executed on a C/C++ programmable 100 MHz 32-bit microcontroller with floating point unit, 19 Analog to Digital Converters (ADC) and 8 Digital to Analog Converters (DAC). It has 32 programmable pins which can be configured as input, output or for external interrupts. There is one programmable interval timer (PIT) which supports timing or counting external events. The development environment is quite powerful with a rich library of system I/O functions and has powerful debugging capabilities. The setup is as shown in the figure 2.

The whole HIL setup system is synchronized by a stable clock generated by the real time system (in actual case this clock comes from input sensor having highly stable clock) and its ticks are used by the micro-controller to generate the tick interrupt for main service routine. The real time systems is either xPC target having dedicated powerful machine or the host PC running MATLAB with Real Time Windows Target (RTWT).

Input sensor data is generated either in the model transferred to the controller through a digital output card in real time systems or this block is disconnected from the model if the actual sensor is present. This input data is sampled by the controller, which after executing the control law algorithm issues the appropriate output signal to the actuators. A number of techniques have been developed to recover the output data, input data from the

Figure 1: Validation Process

Figure 1 shows the flow chart for the validation process. The forward and backward paths corresponds to situation when need arises or some bug is to be fixed at integration time, the processes is repeated. The five steps process is performed when there is a change in controller software while the direct path is followed when no change in controller software is required.

Figure 2: HIL Setup
microcontroller during testing. This input data is then given as input to the non-real time simulation software, run a simulation (non-real time or Open Loop) and the difference of the open loop and closed loop simulation data is plotted which is of the order $10^{-6}$ as the recovered data is in floating point single precision format and simulation is executed with all data types in double precision format.

The system block diagram is like the one shown in Figure 3 [1]. There are two ways to test the embedded software. In one way the plant portion of the system is modeled and downloaded on the real time system. In this case sensor and actuators are part of the model. Physical system represents the dynamic model running in continuous time.

![Figure 3: System Block Diagram](image)

In an other method actuator blocks are independent hardware blocks, only the physical system is modeled and downloaded and executed on the real time system. Actuator systems are connected to the controller (executing control logic) and real time system through I/O cards fitted in real time computer system and controller rack. The later procedure is called actuators in loop.

The control algorithm execution is initiated from a computer communicating with the micro-controller through a serial port using GUI. The setup is as shown in the figure 2. Various modes have been developed in the controller software which tests the controller hardware, the input sensor behavior and the command output. The GUI is also used to transfer various constants to the microcontroller required as controller coefficients and sensor error constants for flexibility.

Following plots show profiles given to actuators from simulation, command signals to the actuators from the microcontroller and the feedback from actuators. The third plot shows a zoomed view of the command and feedback signals. It can be seen that there is very small negligible error which usually of the order $10^{-12}$ or less.

![Figure 4: Simulation Profile](image)

![Figure 5: Data Acquired at rate of 2k Samples/ Sec](image)

![Figure 6: Zoomed view of Command and Feedback](image)

4. Hardware testing with HIL

HIL also helps in testing the hardware interfaces including actuators, input sensors, digital I/O and system clock used for periodic interrupt generation. A single data acquisition model is developed, downloaded on the real time system. The onboard software has certain modes which are conditionally selected from desktop PC communicating with micro-controller through a serial port. Certain profiles are given to the micro-controller from the desktop, then actuator behavior according to those profiles is checked and the data is acquired by the real time system which is further analyzed. The input sensor card is checked by giving input from the model and saving data for some time. If the input card is working properly, sensor is attached to the card and data is taken and sent to the desktop computer for analysis. Various switch boxes have been designed to test the functionality of digital I/O. the system clock is checked by periodically issuing a toggling output on particular output pin of the I/O card (built onboard) which is captured using some fast data acquisition instrument.

5. Controller software and real time system synchronization

Time synchronization of real time system running simulation models and the controlling computer executing control has not been outlined so far. This synchronization is required to correctly model the hardware system tick and other timed hardware operations.
An embedded system contains at least one processor running application specific software that communicates with the external environment in a timely fashion. There are hard deadlines on the execution of such software. More over many embedded systems are safety critical, such systems must ensure to satisfy the real time constraint [6]. One of the main issues in hardware in loop setup is how to synchronize the real time system software with the embedded software i.e. when to start model execution to generate proper input at proper sample time for the embedded controller. In our case the simulation is started when the controller issues a command which brings one of its DIO pin to active high state. After issuing the command, the controller loads the count value to the programmable interval timer (PIT), the output of which is mapped to one of the input pins of the programmable interrupt controller (PIC). Thus periodic interrupts are started at embedded controller while on the other side the model execution on the real time is started in response to the command issued by the controller. The controller elapsed time at particular sample is sent to the slave controller. As discussed previously the input data is also saved, which is then given as input to software simulation model and the parameters are again saved at the host PC. The results are compared and plotted for any difference. Figure 7 shows the timing and the difference after open loop simulation.

![Figure 7: Timing Comparison](image)

### 6. Algorithm parameters extraction in real time

The onboard software consists of an infinite loop with foreground and background tasks. There is no real time kernel or Real Time Operating System (RTOS) installed on the microcontroller as there is no need for RTOS in our application and we don’t want to have any overhead from kernel scheduling algorithm because of the small memory constraints and hard timing requirements of the system software. The foreground processing consists of an interrupt service routine which calls functions and subroutines. In background when the microcontroller does not execute the system algorithm, it sends certain parameters to another microcontroller (slave) coupled through General Purpose Interface Bus (GPIB). These parameters are typecast to single precision from double so as to send as much parameters as possible. At the master end, these data streams can contain 100 or more parameters. A single parameter is a frame consisting of five Bytes. The limitation of frame size is a predefined process used in our proposed scheme. Each parameter is attached with an identity byte (ID) which is used for extracting corresponding data. The first four bytes are data bytes, which can be merged together as float, integer or short to acquire data. The ID’s can then be further divided into two groups of Even ID’s and Odd ID’s.

The slave then frames the parallel data into two serial channels which is received by desktop PC. Pattern recognition algorithms which use two types of filtering techniques including Filtering using time stamp and Filtering using the band of known magnitude for a given parameter have been applied so as to extract as much data as possible. Provisions have been provided in slave software which saves data when there are lots of expected disturbances in the actual test scenario. Figure 8 shows plot of three parameters and the difference when data is extracted from slave.

![Figure 8: Parameter Values and Difference](image)

### 7. Non-model based testing

The micro-controller which is based on Pentium based processor has the capability to address FLASH memory up to 1GB which is enough to save some variables during execution time. This FLASH memory is written at the desktop computer (FAT32 File type) to generate input data. The FLASH is then inserted into slot on the control computer. As the input data is generated from the model in non-real time, there is no more need for real time system and the behavior of the controller is tested by the sequence of timing commands, actuator output data and other parameters as part of the control algorithm. The FLASH is also used to be written the control algorithm parameters, input sensor data and output actuator data for later analysis purpose, but this is not...
recommended in actual system since it takes larger time for writing to the FLASH with FAT32 file system reading/writing algorithm taking about 80% of the tick time. The flash is then removed from the controller and inserted into a desktop/laptop. The data is extracted and plotted for analysis.

This technique is very economical if one has not to use any real time systems and testing the functionality of the embedded systems software, hardware but it depends upon sampling rate and the parameters which are written. This technique also helps in testing the controller software of any high sample rate at low sample rate as the inputs are readily available in the flash memory and there is no need to execute the algorithm at higher sample rate (can’t be used for output checking because of system response at low frequencies). The output can be saved but not given to the actual plant as the frequency will be slower. The technique is useful in applications where the total time software execution time is smaller as data can’t be saved for greater time due to limitations of memory size.

8. Conclusions

HIL provides many benefits to the user and can be applied for testing and validation of safety critical embedded systems (hardware & software). Commercial–Of-The-Shelf (COTS) reliable real time systems are available for this technology which significantly reduces testing time of system software verification and validation processes. Software quality is improved by identifying potential errors and to determine if there are any future trends that would indicate a possible future problem. This has improved our future design and has reduced modeling and implementation errors. Using the technique, we have achieved significant accuracy and the error is of the order $10^{-12}$ in software in loop while $10^{-8}$ with processor in loop sending data in single precision.

9. References


11. Acronyms

ADC Analog to Digital Converter  
COTS Commercial-Of-The-Shelf  
DAC Digital to Analog Converter  
DIO Digital Input and Output  
GPIB General Purpose Interface Bus  
GUI Graphical User Interface  
HIL Hardware-In-The-Loop  
HMI Human Machine Interface  
I/O Input/Output  
PC Personal Computer  
PIC Programmable Interrupt Controller  
RTOS Real Time Operating System  
WCET Worst Case Execution Time  

Acknowledgment

The authors express their gratitude to Dr. Tariq M Jadoon, Dr. Inayat Ullah Baber and Dr. Fida M. Khan for guidance, thorough review and giving useful comments.

About the authors

Syed Muslim Shah
BS in Electrical & Electronics Engineering from NWFP, UET, Peshawar, Pakistan and student of MS Computer Engineering at UET, Taxila. He has more than three years experience in design, development and testing of Embedded Systems Software, experience of real time software preparation and user interface applications development for testing embedded hardware and software. He is employed as engineer in National Engineering and Scientific Commission, Islamabad, Pakistan.

Email: syedmuslimshah@hotmail.com

Muhammad Irfan
BS and MS in Electrical & Electronic Engineering from University of Engineering and Technology Taxila, Pakistan. He has more than three years experience of embedded hardware and software testing using HILS. He is employed as engineer in National Engineering and Scientific Commission, Islamabad, Pakistan.

Email: irfan29wah@yahoo.com