Hardware-in-the-Loop Simulation with the Common Simulation Framework

Judith D. Gardiner
Ohio Supercomputer Center
judithg@osc.edu

Objectives

This paper describes a project to demonstrate the effectiveness of using the Common Simulation Framework (CSF) for hardware-in-the-loop (HWIL) testing. Our objectives were twofold, to enhance the real-time support provided in CSF and to provide example code for a HWIL simulation using CSF. Similar work in the past has involved changes to CSF that were unique to the local operating system and hardware under test. Our goal was to create software that could be efficiently reused and adapted for other simulations. As a simple demonstration we used CSF to drive a motion simulator table through a preprogrammed trajectory.

Hardware and Software Environment

The test environment consists of a quad core computer running the simulation, a reflective memory network, a motion simulator table, and a table controller. The simulation computer runs the iHawk operating system, which is a real-time version of Linux.

The simulation sends trajectory information in real-time to the table controller via reflective memory. Access to reflective memory is handled through a locally developed program called RAP (Resource Allocation Program). Reflective memory is a real-time networking technology designed to mimic shared memory.

Common Simulation Framework

CSF is a simulation framework developed by AMRDEC (Army, Huntsville) that allows users to assemble component models into a complete missile simulation. It is intended to be general enough to be used in a variety of computational environments and to support a wide range of simulation domains.

The current version of CSF includes most of the features required for real-time operation. Multithreading is supported and is easy to use. There is a GUI for convenient development and a batch mode for fast execution. Custom real-time clocks can be added or a default real-time clock can be used.

In the course of our work we discovered a few gaps in CSF's real-time capabilities and made code modifications to correct them. Most of our development, however, was done within the framework without touching the CSF core code.

CSF Extensions for the HWIL Demo

For the demonstration we read trajectory points from a file and sent them to the motion table controller at the appropriate real-time rate. We developed a real-time monitor and a generic reflective memory interface for use in the demo.

When running a simulation with hardware in the loop, or in any other hard real-time environment, it is necessary to monitor for frame overruns. Updates must be sent to the hardware at regular intervals. If some portion of the simulation takes too long and an update is not ready at the specified time, the simulation results will not be valid. There may even be a risk of hardware damage if the simulation is allowed to continue. It is essential to provide the ability to terminate the simulation automatically and gracefully in this situation.

CSF does not have a real-time monitoring capability built in, but it does provide the hooks for adding such a capability as a plug-in library. We have developed a real time monitor that can be run as part of any real-time simulation. It can be run in a separate thread to eliminate any performance penalty. The monitor reports on frame overruns and can be configured to terminate the simulation if an overrun is detected.

For the interface to reflective memory we created a general-purpose reflective memory API to wrap the RAP interface that we were required to use. The wrapper will facilitate reuse of the code in different environments.

Conclusion

The goals of this project involve both real-time performance and generality of the software solution. Results will be presented relative to both goals. The project is not yet complete, but we expect to be able to drive the motion table at 1200 Hz using a simulation running under CSF. Furthermore, our software solution will be general enough to be of use to others in the DoD test and simulation community.

The extensions and modifications to CSF will be offered to the CSF steering committee for acceptance in the standard distribution. If accepted, the software will be available to a wide range of users across DoD's test and range facilities.

Acknowledgment

This publication was made possible through support provided by DoD HPCMP PET activities through Mississippi State University under contract No. GS04T01BFC0060. The opinions expressed herein are those of the author(s) and do not necessarily reflect the views of the DoD or Mississippi State University.