

Efficient Engineering Through Simulation of CAN-Based Control Systems

By Johan Strandberg, Mikael Akerholm, Jörgen Hansson,
Mats Kjellberg, and Johan Bäckström, CC Systems AB

Developing applications for distributed CAN-based control systems is often a challenging task due to limited access to target hardware, especially in early phases of a project. Even with access to the target hardware, the process of testing and debugging can be highly inefficient and thereby expensive. The testing and debugging of the systems can be simplified by using simulation technology since black box functional testing is minimized. In this paper, the authors present how simulation technology may be used for embedded control systems and particularly how it has been used within their organization for several years with satisfactory results. They present how a set of tools, which includes a complete simulation core for communication buses, I/O handling, memory handling, etc., may be used to simulate a complete embedded control system in a single standard PC. It is also possible to use this simulation technique to emulate/simulate the surrounding target system like I/O signals, external communication buses, etc., by development of environmental models. Since this simulation also has support for National Instruments I/O boards as well as a number of CAN adapters, mixed simulation may be used for system and hardware tests. By using these techniques, higher efficiency can be obtained and the time to market can be greatly reduced.

Introduction

Development and testing of software for distributed embedded control systems is an area with great potential for improvements in terms of efficiency, quality and cost. One reason for this is that access to target hardware is often a limiting factor in many software development projects. This can cause unnecessary delays and costs.

The simulation technology described in this paper is a set of tools that can be used to improve and simplify development of software for embedded systems. This is facilitated by replacing all target hardware dependencies in the software with simulated equivalences—making a complete distributed embedded system executable and testable in a single PC. By using simulation, software can be developed without target hardware. Target hardware will instead be introduced into the testing procedure when the functionality is proven to work in a simulated environment. In addition, automated testing and fault injection can be used to guarantee the functional behavior of the software. This, in turn, simplifies the introduction of target hardware, since the correct functional behavior of the software is already validated.

Efficient Engineering Through Simulation

A Typical CoDeSys-Based CANopen System

In order to describe the context in which the simulation technology can be used, we outline some common and typical solutions and principles used in the design of a distributed CANopen-based system. The system architecture can be described as a set of computer nodes called Electronic Control Units (ECUs). The most common solution is to use one intel-

ligent node (PLC) as a CoDeSys programmable network master. This node does not only have the task of being the network master, but it will also serve as the application master since all functionality of the system will lie in this node. The other nodes are “dumb” CANopen slaves distributed over the machine/vehicle and contain a number of I/O signals per node. Furthermore, the nodes may be divided into multiple subnets, and there may be gateways and other communication buses, which will increase the complexity of the system even more.

In order to test the software functionality, the developer will

be delays in hardware delivery from the many manufacturers usually involved. If there is more than one software developer involved in the project, there is the need to set up more than one target system or the developers have to share the system with each other. Either way causes high costs and low efficiency. Another problem is that the first control system is usually built in a laboratory environment. In some cases, test racks with buttons, knobs and gauges are connected to the nodes in order to simulate them with signals and read outputs. This can turn out to be a very expensive setup and also cause delays before the test

in an easy way directly on the PC with tools designed for the purpose. Automated testing can easily be introduced and third-party programs for simulating physical behaviors can be integrated. This simulation-centric way of working has been used and developed by CC Systems for more than 10 years.

Simulation And Test Environments

Using this simulation, three conceptually different types of simulation environments (complete system simulation, mixed simulation and target hardware tests) become available to develop and test embedded control systems. Using these three test environments, combined with automated test tools and an environment model of the system to control, we get a complete test solution for embedded systems and give rise to a simulation-centric development process. This approach has proven successful in many different software projects.

Complete System Simulation

In a complete system simulation, all nodes, control panels and environment models are simulated in a single PC. This is the most commonly used test technique by CCS's simulation users, since no target hardware is needed.

The nodes communicate using a simulated field bus, e.g., CAN. I/O signals between the nodes, and the external devices (sensors, actuators, etc.) are simulated using I/O channels in the PC. The necessary internal target hardware devices of the nodes (EEPROM, Flash, PLD, Power monitor, etc.) are also simulated. These simulation devices (e.g. CAN, LIN, Flash Memories, RS232, etc.) are implemented as reusable software components. These components are at the heart of the

simulation technique and are implemented to support both internal communication between the simulated hardware nodes and communication with different devices to the PC. The technical issues of the simulation technique are further described later in this paper (see section “CC Simtech”). To make the simulation-centric development process more complete, we have integrated support for communication with third-party tools. These tools can, e.g., be used to build environment models or control panel simulations. Also, automated test tools, both for unit testing and for system tests, can be used together with the simulation. The environment models enable simulation of, e.g., mechanical, thermal and electrical devices, and these models are connected to the simulated hardware nodes using simulated I/O ports facilitating a complete system simulation. A system panel is typically used to control the start and stop of the simulated nodes.

Mixed Simulation

In a mixed simulation setting, some target hardware nodes are used together with the simulated system. This type of simulation is mainly used for two purposes: first, it enables integration of target hardware and hardware-related software in a controlled step-by-step procedure; second, it enables testing of the physical communication between nodes. Furthermore, testing of I/O ports is possible.

The environmental models and the control panels are the same as for complete system simulation. Also, if automated test tools are used, the same test scripts can be used regardless of whether complete system simulation or a mixed setting is used. Hence, hardware-in-the-loop tests are facilitated using the described approach for simu-

lating embedded systems. Real target nodes can be connected to the other nodes via, e.g., a PC CAN-card and I/O ports. A simulated CAN-bus then connects to the simulated nodes in the PC. Furthermore, IO-ports on the external nodes are connected to data acquisition hardware in the PC and connected to the simulated nodes and environmental models.

Target Hardware Tests

During the final phase of the simulation centric development process, all target hardware nodes are used. However, the control panel components can still be simulated on the PC and connected to the external nodes through data acquisition hardware in the PC. To facilitate hardware-in-the-loop testing, the target system hardware communicates with the environmental models describing the peripheral components and behavior in the system.

The environmental models and control panel simulations as used in fully simulated tests are reused for target system simulation.

Timeline Benefits

During an ordinary project without simulation, there is no way to do neither module tests nor system tests until the design, build and verification of prototypes are finished. Functional tests do not start until prototypes have been delivered. We divide the test into two parts: Functional Tests and System Tests. Functional tests include verifying the function of the individual module as well as the entire system. System test is running the complete system on final hardware in a lab and eventually on the actual machine/vehicle.

When simulation is used, the functional tests of each module, as well as the entire system,

may be performed in a simulated environment in parallel with the hardware development.

Note that integration of hardware and software is, of course, still required. However, the function of the system has already been verified. This integration takes longer time since some errors may originate from application errors not eliminated. The biggest difference is, however, in the functional testing phase since all functional testing will be on actual hardware with limited debug possibilities. Furthermore, the testing is initiated late in the project and if critical errors are found, even more time will be added to the schedule in contradiction with early found errors, which may be corrected much earlier in the project with less effort. In other words, if testing is initiated late in the project, the risk will increase significantly.

CC Simtech

As described, simulation enables control system software to be developed and tested without access to the target hardware. This is achieved by replacing all target hardware dependent operations (e.g. device driver and real-time operating system calls) with simulated equivalences. These equivalences are the core of this simulation technique. They are implemented as a set of software components, simulating the behavior of, e.g. CAN, LIN, I/O, RS232, EEPROM and Flash.

It is important to emphasize that the software executed on top of the simulation uses the same source code as the target hardware system does, except that the device drivers and the operating system are replaced. The application programmer's interface, used to communicate with the environment, is the exact same regardless of whether the control system is simulated or running on target hardware.

This means that the target source code can be tested and debugged in a very efficient way compared with black box testing on hardware. Furthermore, powerful tools for debugging, automated testing, fault injection and dynamic modeling of environmental models of the target machine, available for PCs, can be used to guarantee the functional behavior of the software. This package also offers time accurate simulation; this is among other things a sync feature between different simulated nodes in a system and is not described in this paper.

Conclusion And Future Work

In this paper, we describe an approach to improve software development for embedded distributed real-time systems. We describe the simulation technique and how it can be used to facilitate a simulation centric software development process. The technique can be used during all phases of embedded system software development, all the way from a system simulated in a single PC using automated test tools and environmental models to hardware integration tests. Plans for future work include improvements of performance issues when simulating a large system (typically in the range of 20-30 simulated nodes) and looking at extending the technique with a number of new software components (e.g. other communication buses).

For more information, contact info@cc-systems.com CC Systems can be found at www.cc-systems.com.

“Simulation enables control system software to be developed and tested without access to the target hardware.”

need access to the complete system and do all the debugging and testing directly on the target hardware. CoDeSys offers a simple simulation option, but it is very limited, especially in a system with a network since the CoDeSys simulation does not include communication with other nodes.

Motivation

There are a number of problems with the very common setup mentioned above. The developer is dependent of the target hardware in order to start debugging and testing the system. Many projects demand that new hardware modules are developed and hereby are not accessible during the first phases of the project. There can also