AN INTEGRATED ENVIRONMENT FOR MODELING, SIMULATION, DIGITAL SIGNAL PROCESSING, AND CONTROL\textsuperscript{1}

Curtis C. Crane and Richard J. Kozick

Electrical Engineering Department
Bucknell University, Lewisburg, Pennsylvania 17837
kozick@bucknell.edu

ABSTRACT

An integrated laboratory for system modeling, simulation, real-time digital signal processing, and control is being developed at Bucknell University for undergraduate electrical engineering education. The laboratory bridges the gap between software simulation and testing of actual systems through a common visual programming interface. Students can explore the iterative process of developing a model and then refining the model until computer simulation results agree with experimental measurements. A liquid level control system is presented to illustrate the features of the laboratory environment. The key components of the laboratory are networked digital signal processing (DSP) hardware units and a simulation software package. The simulation software runs on workstations, and all of the laboratory equipment (including the DSP hardware) is connected to the Internet.

1. INTRODUCTION

Undergraduate electrical engineering laboratory exercises often include the steps of developing a mathematical model for a system or signal, simulating the system or signal, and finally performing an experiment involving the actual system or signal. These steps are typically performed at different times, in different physical locations, using different hardware/software tools, and with limited iteration of the modeling/simulation phase in response to experimental observations. An integrated environment that allows students to cycle through the modeling, simulation, and real-time stages of the experiment would help improve student understanding of system and signal theory.

An integrated laboratory of this type for digital signal processing and control applications is being developed at Bucknell University. The visual programming environment of Simulink provides a common interface for both computer simulations and real-time execution of algorithms on dSPACE digital signal processing hardware. The dSPACE/Simulink combination allows students to develop a model for a system, perform computer simulations of the system based on the model, and compare the simulation results with experimental measurements of an actual system. All of this can be done during the laboratory or class period using an integrated set of tools. The integrated environment facilitates iteration of modeling and design procedures in order to match simulation results with measured results and meet design specifications. The graphical nature of the Simulink interface removes the difficulty that is often associated with programming digital signal processors (DSPs). The merits of a graphical interface for DSP programming are described in [1].

The integrated environment has been used in several courses at Bucknell University, including the multidisciplinary “Exploring Engineering” course, as well as the electrical engineering courses in signals and systems, control systems, and communication systems. It will be used in several other courses, including digital signal processing and introductory electrical engineering courses. One interesting characteristic of the laboratory is that all of the components are connected to the Internet, making it possible to remotely share the laboratory and experiments with other universities [2,3].

This paper illustrates the operation and benefits of the dSPACE/Simulink environment by presenting a control system application. The laboratory hardware and software are described in Section 2, and the modeling, simulation, and real-time control of a liquid-level system are discussed in Section 3.

2. LABORATORY HARDWARE AND SOFTWARE

The main hardware components are ten dSPACE DS1102 Miniboxes and ten Sun workstations. The dSPACE Miniboxes and Sun workstations

\textsuperscript{1} This material is based upon work supported by the National Science Foundation under Grant No. DUE-9451618.
communicate with each other through the Internet. Each DS1102 Minibox contains a 40 MHz Texas Instruments TMS320C31 digital signal processor along with memory and input/output circuits, including four channels each for analog-to-digital and digital-to-analog conversion.

The dSPACE Miniboxes can be programmed in assembly language, C language, or through Simulink, which is a graphical interface to MATLAB. The Simulink interface is the simplest to use, since systems and algorithms are described by block diagrams. Simulink contains an extensive library of predefined blocks for signal and system analysis.

Once an algorithm or system is described by a block diagram in Simulink, it can either be simulated on the Sun workstation, or compiled, downloaded, and executed in real-time on a dSPACE Minibox. The simulation runs on the workstation only, while the real-time algorithm is executed on the dSPACE Minibox and is used to process external signals and connect to physical systems. The real-time version can be developed rapidly from the simulation version, since the Simulink block diagram descriptions are nearly identical for both cases.

An additional software tool called TRACE serves as a virtual oscilloscope for real-time data collection and display. TRACE can be used to transfer measured data directly to MATLAB for analysis. The combination of Simulink, MATLAB, and TRACE provides an integrated set of tools for simulation, real-time implementation, and testing of signal processing and control algorithms.

3. LIQUID LEVEL CONTROL SYSTEM

The integrated dSPACE/Simulink environment is applied to a liquid level control system to illustrate each step of the modeling, simulation, and real-time testing process. Some interesting pedagogical features of this approach include:

- Students develop a mathematical model for a real system and estimate parameters of the model, in contrast to the common textbook approach of "assuming" a model for a system.
- Students use the model to obtain computer simulations of the nonlinear liquid level system with Simulink.
- Students use TRACE to measure data on the actual system, and then compare the data with the simulation results.
- The entire process is performed using the dSPACE/Simulink environment. Mathematical modeling, computer simulations, and real-time implementation are performed at the same time and using the same tools. Students explore and understand differences between simulation and actual results, rather than simply observing the disagreement in a lab report.
- Students can document their results in a report using standard word processors on the workstations. Plots from Simulink and TRACE are easily imported into the reports.

3.1. Liquid Level System (LLS)

The LLS hardware is depicted in Figure 1. It consists of a column with a small hole in the bottom from which water exits into a reservoir. Water enters the column at the top by means of a variable speed pump. A pressure transducer is located at the bottom of the column, and the measured pressure is related to the height of water through a calibration process. The objective of the Liquid Level Control System (LLCS) is to maintain a specified height of water in the column. This system and several other low-cost experiments are available at Bucknell University and can be developed at other universities [3,4].

A block diagram of the liquid level control system (LLCS) is shown in Figure 2. Based on the error between the desired height of water and the measured height of water, the controller applies a voltage to the pump to produce a change in the water height. The Simulink description of the LLCS for simulation purposes is shown in Figure 3. Only a
slight modification of the Simulink diagram in Figure 3 is needed to convert it into a real-time control algorithm that executes on the dSPACE hardware. The real-time algorithm performs the operations enclosed by the dotted line in Figure 2.

3.2. Modeling the LLS

A model can be developed for the water column and pump dynamics by balancing the rate of change in height with the flow rates into and out of the column. A first attempt at the model might be the following nonlinear differential equation:

\[
\frac{dh}{dt} = -C_a \sqrt{h} + C_b V_{in}
\]  

(1)

where

- \( h \) = height, measured from the orifice
- \( V_{in} \) = voltage applied to the pump
- \( C_a \) = constant relating height to flow rate out
- \( C_b \) = constant relating \( V_{in} \) to flow rate in

The model in (1) is quite simple, in that the flow rate into the column is assumed to be a linear function of the voltage applied to the pump. The parameters \( C_a \) and \( C_b \) in the model can be determined from two measurements:

• **Draining time** -- If the pump is turned off, i.e. \( V_{in} = 0 \), then (1) can be integrated to yield the following relation between the time \( t_{final} \) required for the column to drain from starting height \( h(0) \) to ending height \( h(t_{final}) \):

\[
\sqrt{h(0)} = C_a \frac{t_{final}}{2} + \sqrt{h(t_{final})} \tag{2}
\]

\( C_a \) is then the slope of the line when \( \sqrt{h(0)} \) is plotted as a function of \( \frac{t_{final}}{2} \).

• **Steady-state** -- When the liquid height is constant, i.e. \( \frac{dh}{dt} = 0 \), then \( C_b \) can be found directly from

\[
C_b = C_a \frac{\sqrt{h}}{V_{in_{ss}}}, \text{ at steady state.} \tag{3}
\]

These measurements are easily acquired with a ruler and stopwatch. Draining time can be measured for several initial heights \( h(0) \), and the steady-state height can be measured for several pump voltages \( V_{in} \). When this data is plotted with MATLAB, two facts become apparent [5]. First, the draining time data closely matches the model in (2), so a least-squares procedure can be used to estimate \( C_a \) from the slope of the graph. (MATLAB contains built-in functions for the least-squares estimate.) Second, the steady-state measurements do not match the model in (3). The problem is that the pump does not turn on until the voltage \( V_{in} \) exceeds a threshold.

The model in (1) can now be refined in light of the measured data. The solution is to include a nonlinear function of the pump voltage, \( f(V_{in}) \):

\[
\frac{dh}{dt} = C_a \sqrt{h} + f(V_{in}) \tag{4}
\]

Additional measurements and subsequent least-squares analysis with MATLAB produces the results \( C_a = -0.40 \) and

\[
f(V_{in}) = \begin{cases} 
0.81 V_{in} - 0.58 & \text{if } V_{in} \geq 0.75 \\
0 & \text{if } V_{in} < 0.75 
\end{cases} \tag{5}
\]

The process of estimating parameters in the model through simple measurements and calculations is a valuable learning exercise for students. Deciding how to revise the model in light of measured data is another important lesson. The model in (4), (5) is used in the following subsection to simulate the LLCS. The simulation results are then compared with measurements of the actual system behavior.
3.3. Simulation and Implementation

Results

A Simulink model for the LLS is easily developed from (4) and (5) and is encapsulated into the “LLS Plant” block in Figure 3. (The complete diagram for the LLS Plant can be found in [5].) Note that Simulink has allowed us to describe the LLS with a block diagram, and this block diagram can now be used to perform a variety of simulations. Various control algorithms can be simulated, and the effects of parameter variations can be investigated. An important payoff from the effort in constructing the simulation block diagram is that the real-time implementation with dSPACE requires only a few more keystrokes!

Figure 4 displays the result of Simulink simulations and real-time control of the LLS using proportional-plus-integral (PI) control. Three curves are shown: one for simulation, one for real-time control with a C program, and one for real-time control based on the Simulink block diagram. The simulation curve begins at height -4 inches while the measured height is never less than 3 inches because the pressure transducer is located 3 inches from the bottom of the column. The simulation results agree closely with the measured response of the system. Similar agreement between simulation results and measured results were obtained for proportional (P) and integral (I) controllers [5]. The refined model in (4), (5) was found to be important in order for simulation results to agree with actual system results.

REFERENCES