Development of a Universal Controller for Pedagogical Applications Involving Data Acquisition, Data Logging and Control

Nicholas Krouglicof
Union College, Mechanical Engineering Department

Abstract

This paper describes the development of a novel, low cost, microcontroller-based system that enables students to interface a variety of sensors and actuators to their laptop computers in a laboratory or studio classroom environment. The system could potentially find application throughout the engineering curriculum at Union College beginning with the freshman *Introduction to Engineering* course, a required course for engineering students in all majors. At the junior and senior levels it is designed to support laboratories in *Dynamics of Systems* and *Mechatronics Design*. Finally, as a stand-alone controller, it could potentially be applied to numerous mechatronic student projects including the International Virtual Design Studio (IVDS), the Robotics Club, the SAE Walking Machine Challenge as well as several capstone design projects each year.

The system is both technologically and pedagogically innovative. In terms of pedagogy, in provides a new avenue of application for laptop computers in the undergraduate curriculum. In terms of technological innovation, the system is unique in that it functions as a data acquisition system, stand-alone controller or data logger. It incorporates both a user programmable microcontroller and a user configurable Complex Programmable Logic Device (CPLD). The microcontroller supports high-level programming languages such as C and Basic as well as low-level assembly language. Finally, in terms of design philosophy, the system is based on an open architecture (i.e. all the firmware, source code and development tools are available to the student at no cost).

Introduction

Laptop computers are becoming increasingly "pervasive" in undergraduate engineering programs throughout the United States.[1] For example, under the IBM ThinkPad University program, freshmen are provided with the latest laptop computers and software at a substantially reduced cost. The possible pedagogical applications for these computers in the undergraduate engineering classroom are relatively obvious; word processing, spread sheets, Computer-Aided Design (CAD) packages, presentation software, mathematical utilities such as MATLAB, etc. In a laboratory environment, however, there are many impediments to the use of laptops as data acquisition systems or as real-time controllers. Most of the commercially available data acquisition hardware is designed for desktop computers (ISA or PCI bus) and cannot be readily interfaced to a laptop. Products that interface to the host computer via the USB port or by means of a PCMCIA cards can be used with laptops, however, the cost is prohibitive – on the order of \$1000.00 per system – and they offer only limited functionality. After a fairly exhaustive, unsuccessful search for a low-cost data acquisition package specifically for laptop computers, the decision was made to develop UC^2 (***U**nion **C**ollege **U**niversal **C**ontroller*) a system tailored to the needs of engineering students at Union College.

Design Requirements

Interface to Host Computer

Due to the unavailability of ISA or PCI bus expansion slots in laptop computer, the primary means of interfacing a laptop to a data acquisition system is either via a PCMCIA card[3] or the USB port. In general, the PCMCIA solution is costly and complex. Furthermore, since PCMCIA slots are generally not provided on desktop computers, this effectively limits the range of application to laptops. While the primary focus of this work is portable data acquisition, the system should also function in a more conventional laboratory environment consisting of fixed desktop computers.

The USB solution is viable[4,5] but still presents certain shortcomings. USB communications are not directly supported by certain older operating systems (notably Windows NT 4.0). Of greater significance is the lack of support under Visual Basic 6 which is gaining popularity in colleges and universities as a general-purpose programming language. Although the USB port is an effective means of transmitting large files from a scanner to a computer or from a computer to a printer, the overhead makes it is somewhat less efficient in transmitting small packets of information to or from a data acquisition system.

For these reasons, the decision was made to interface the data acquisition system to the host computer via the conventional serial port (RS-232). Serial communications are readily supported by Windows and by Visual Basic 6. In fact, data received via the serial port is automatically buffered by the Windows operating system and can read in an event driven manner under Visual Basic with no additional software components. Relatively high baud rates (typically above 115200) permit acquisition rates of several thousand samples per second which is adequate for most mechatronic applications. Serial ports remain prevalent on both laptop and desktop computers although certain manufacturers of laptop computers have completely eliminated the serial ports in certain products in favor of USB. This trend may continue. Consequently, in designing UC^2 certain provisions were made to facilitate the addition of a USB port in future revisions.

Stand-Alone Operation

Undergraduate students are increasingly involved in mechatronic design projects that call for small, microcontroller-based, stand-alone controllers. For example, student participating in the SAE Walking Machine Challenge must design an intelligent, autonomous system that must perform a variety of simple tasks without human intervention. Currently, the most commonly used controllers are the BASIC STAMP and HANDI-BOARD computer.

A less common but equally important application is data logging. Typically, sensory information is acquired in a remote or hostile environment over extended periods of time. Often the hostile environment or the unavailability of electrical power preclude the use of a desktop or even a laptop computer. Possible applications include remote weather stations, long term studies of building envelopes and structures, monitoring wind mills and solar panels, etc. In general, industrial data loggers are highly application dependant and relatively expensive.

The three functions described above, namely: portable data acquisition, autonomous control and remote data logging, share a certain commonality in terms of hardware. In recognition of this, the primary objective in developing UC^2 became to realize a simple, low-cost system that is pedagogically friendly and that can function as:

- a portable data acquisition system that can be readily interfaced to a laptop computer in a studio classroom environment,
- an autonomous, stand-alone controller in support of mechatronic design project,
- a data logger that can acquire and store data at regular intervals over extended periods of time.

Software
Cost and availability were the primary factors governing the choice of software tools for programming UC^2 as a stand-alone controller as well as for programming the graphical user interface on the host computer. The decision was made to rely exclusively on software that academic institutions already use (e.g. Visual Basic) or software that they can obtain at no cost.

System Description

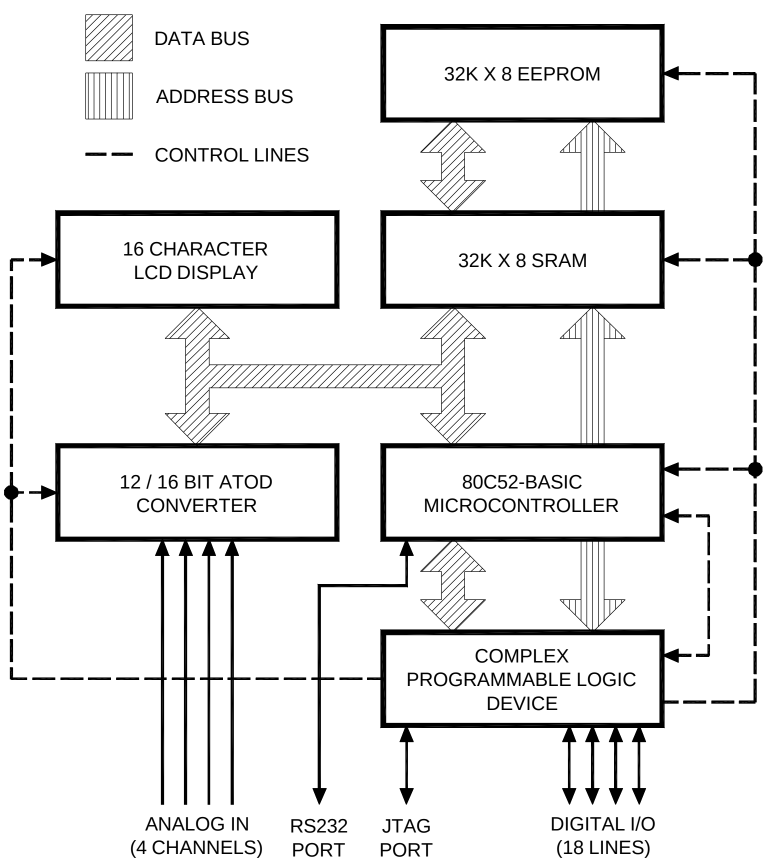

Figure 1: Block Diagram of System

Embedded Controller

In order to satisfy the design requirements related to autonomy of operation, UC^2 must incorporate an onboard embedded controller. The controller simplifies the management of the various onboard peripherals (Analog-to-Digital converter, LCD display, serial interface). To facilitate software development by the end user, the onboard controller must also support high-level programming languages such as BASIC or C, as well as low-level assembler.

The embedded controller selected for this application is the 80C52-BASIC. This device incorporates a powerful, imbedded BASIC interpreter that was originally developed by INTEL for process control applications. The source code for this interpreter is available as “freeware” and has been adopted by a number of other manufacturers of embedded controllers.

A typical MCS BASIC-52 program is illustrated bellow.

MCS BASIC-52

```
        REM *** SELECT THE ANALOG INPUT CHANNEL

10      INPUT "Analog Input Channel ? ",CHAN
20      XBY(7FF2H)=CHAN

        REM *** CONVERT FROM 2'S COMPLEMENT FORMAT

30      MSB=XBY(7FF2H) : LSB=XBY(7FF3H)
40      HEXVAL=MSB*256+LSB
50      IF HEXVAL>32767 THEN HEXVAL=HEXVAL-65536

        REM *** CONVERT INPUT RANGE TO +/-10 VOLTS

60      VOLT=HEXVAL/3276.8

        REM *** DISPLAY RESULTS

70      PRINT USING(0),"A to D channel ",CHAN," = ",
80      PRINT USING(###.####),VOLT
90      GOTO 20
```

This program can be entered, edited and executed via Windows HyperTerminal or any other terminal emulation program. The program first prompts the user to enter an analog input channel. The channel number is written to the analog multiplexor and conversion is initiated on the Digital-to-Analog Converter (ADC). The program then reads the most significant byte (MSB) and the least significant byte (LSB) from the ADC. The results are converted from 2’s complement format and displayed in the terminal window.

The code is relatively self-explanatory with the exception of the XBY instruction. XBY is analogous to a pointer in the C language. The statement XBY(7FF2H)=CHAN stores the channel number in memory location 7FF2H (H signifies hexadecimal). The statement MSB=XBY(7FF2H) stores the contents of address location 7FF2H in variable MSB.

Note that peripheral devices in the UC^2 system are memory mapped. A WRITE operation to address 7FF2H or 7FF3H writes to the channel register of the analog multiplexor and initiates an analog conversion on the ADC. A READ operation from addresses 7FF2H and 7FF3H accesses the most significant byte and least significant byte of the ADC respectively.

MCS BASIC-52 employs a common 8 byte floating-point data type for all variables with the exception of strings. This is consistent with "standard" BASIC. The firmware on the 80C52-BASIC parses each line of code as it is received and stores it in a compact "tokenized" format. In the example shown above, the comments (statements beginning with REM) are not given line numbers so that they are interpreted but not stored in memory.

Memory
The UC^2 system incorporates 32K bytes of external SRAM (**S**tatic RAM) in the address space from 0 to 7FFFH. The last 16 bytes of RAM (7FFF0H to 7FFFH) are mapped to peripheral IO devices (ADC, LCD display, etc) and are not available to the user. In terms of non-volatile memory, an E^{2}PROM (**E**lectronically-**E**rasable-**P**rogrammable-**R**ead-**O**nly-**M**emory) occupies the address space from 8000H to FFFFH. The firmware for the BASIC interpreter is located within the 80C52-BASIC and does not impact external memory usage.

In general, programs are written and debugged in RAM and later transferred to E^{2}PROM (i.e. nonvolatile memory) for long-term storage. MCS BASIC-52 incorporates a simple file management system for editing, storing and retrieving multiple programs from either volatile or non-volatile memory. For stand-alone applications, the 80C52-BASIC device can be programmed to auto-execute a BASIC program stored in E^{2}PROM on power-up.

Both volatile and non-volatile memory in the range from 0000H to FFFFH can be accessed under program control via the XBY instruction. In data-logging applications, data is typically written to free nonvolatile memory for long term storage.

Analog-to-Digital Converter
A high-performance, 12 bit, 25 µs Analog-to-Digital Converter (ADC) with integrated sample-and-hold amplifier and 4 channel analog multiplexor is included in the UC^2 system. The bipolar input voltage range is set to ±10V which yields a resolution of 4.9 mV. For more exacting applications, a pin and software compatible 16 bit ADC is available which provides a theoretical resolution of 0.3 mV over the full ±10V input range. As illustrated in the software example shown above, the ADC is completely accessible through the MCS BASIC-52 programming language.

Complex Programmable Logic Device
Perhaps the most innovative feature of the UC^2 system is the incorporation of a user configurable Complex Programmable Logic Device (CPLD). The CPLD handles all the routine tasks normally associated with embedded controller applications; data latching, address decoding, and memory management. This effectively eliminates the need for any discrete logic on the board. In addition, approximately 75% of the internal resources and 18 IO pins on the CPLD are made available to the user for custom applications. The CPLD supports in-system programmability

(ISP) via the IEEE 1149.1 Joint Action Test Group (JTAG) test port. This permits the target device to be reprogrammed by the user without removing it from the host system. Code for the CPLD is developed in either in VHDL (**V**ery High Speed Integrated Circuit **H**ardware **D**escription **L**anguage) or AHDL (**A**ltera **H**ardware **D**escription **L**anguage) using the MAX+plus II 9.23 Baseline development environment from ALTERA. Educational institutions can obtain a MAX+plus II software license for the specific device used in the UC^2 system at no cost over the WEB.

Clearly reconfiguring the CPLD is beyond the capabilities of many students. In recognition of this, the 18 free IO pins on the device are predefined as 8 digital input lines and 8 digital output lines plus a free chip select (CS) and a PWM (**P**ulse-**W**idth-**M**odulation) signal. These IO signals are all accessible by the user through the MCS BASIC-52 programming language. On the other hand, advanced user can change the default settings to suit their application.

Examples of mechatronic applications envisioned for the CPLD on the UC^2 system include quadrature decoder/counter interfaces for optical encoders, stepper motors controllers, PWM motor drives, sequential controllers, event counters/timers, etc. Conventional tasks (e.g. address decoding) are not the primary reason for incorporating the CPLD, but can also be explored in a laboratory environment.

LCD Display
For stand-alone and data logging applications, the UC^2 system incorporates a 16 character LCD display which can be accessed directly via the MCS BASIC-52 programming language.

Power
The UC^2 system is typically powered by a standard, 9 VAC wall-mount adapter. For autonomous applications, a standard 9 volt battery can be substituted for the adapter. Since most of the components on the UC^2 system are CMOS devices, power consumption is reasonable - approximately 140 mA.

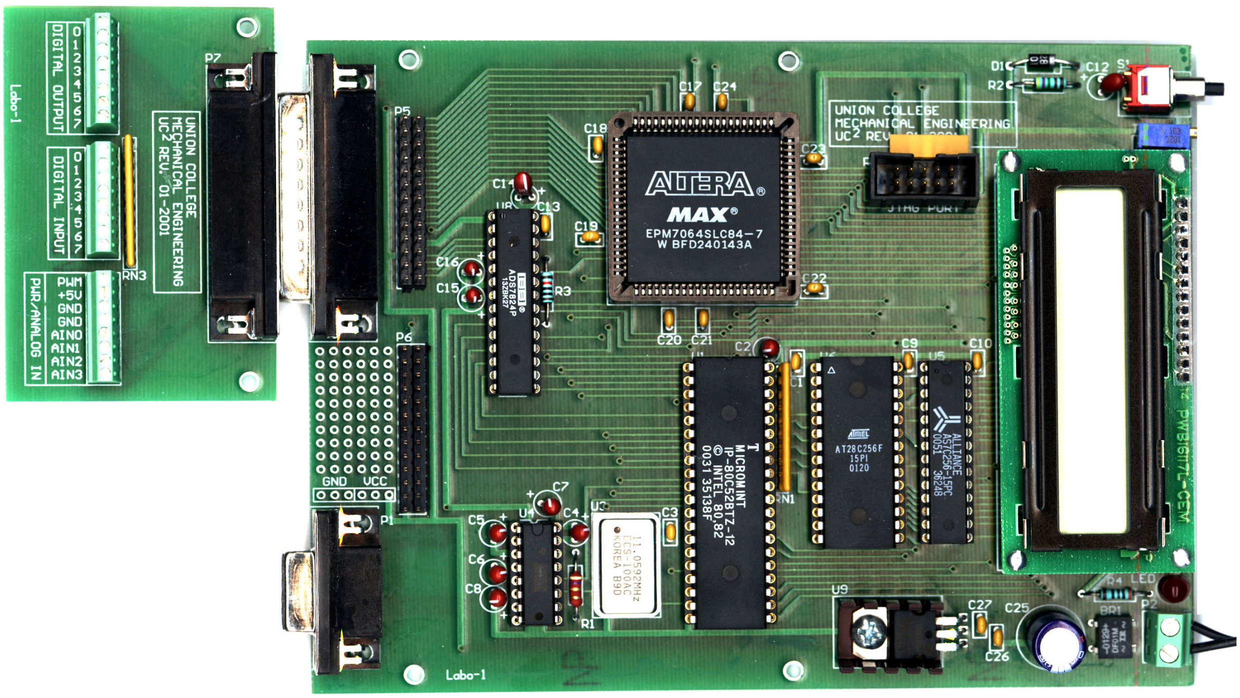

Figure 2: Image of populated printed circuit board for the UC^2 system.

<u>Integration in the Union College Engineering Curriculum</u>

Introduction to Engineering Course

All freshman engineering students at Union College, regardless of their major, are required to take *Introduction to Engineering*. This course includes an important laboratory component covering design, engineering communications and an introduction to "intelligent" systems (i.e. system that are computer controlled). The goal is to introduce the UC^2 system into the laboratory component of the course. A typical preliminary laboratory exercise would involve interfacing a number of sensors to the UC^2 system and acquiring the data on a laptop computer. Under this scenario, students would be provided with a user interface (written in Visual Basic) that would acquire, display, and store the sensory data in the appropriate format. A typical graphical user interface is shown in Figure 3. The actual laboratory exercise would consist, for example, of calibrating a pressure transducer using a manometer or dead weight tester.

It is worth noting that under this scenario, students are not required to familiarize themselves with the detailed, internal operation of the UC^2 system. The Visual Basic application takes control of the system, downloads the appropriate code to the onboard embedded controller and acts as an interface between the user and the system.

A more advanced laboratory would consist of using the UC^2 system in stand-alone mode to control a simple process, for example, temperature control of an enclosure. Student would implement the control algorithm in MCS Basic-52 and evaluate the performance of the system by logging the temperature as a function of time.

As a final exercise, we are currently considering having the students design a simple autonomous vehicle that would sense the environment, navigate around obstacles, and realize a simple task.

Introduction to Programming

The Mechanical Engineering Department at Union College is currently considering modifications to its *Introduction to Programming* course. Instead of a classic computer programming course that focuses on the syntax of a specific high level language such as C++ or Java, the new course would take a "macro-programming" approach using a combination of EXCEL and VISUAL BASIC to solve elementary engineering problems. The emphasis would be on algorithm development and implementation as well as the creation of appropriate user interfaces for data entry, manipulation and visualization. The UC^2 system would serve as a tool in support of such a course.

Mechatronics Design

Mechatronics can be defined as a design philosophy which encourages engineers to integrate precision mechanical engineering, digital and analog electronics, control theory and computer engineering in the design of "intelligent" products, systems and processes rather than engineering each set or requirements separately. The advantages of the mechatronics approach to design are shorter design cycles, lower costs, and elegant solutions to design problems that can not easily be solved by staying within the bounds of the traditional engineering disciplines.

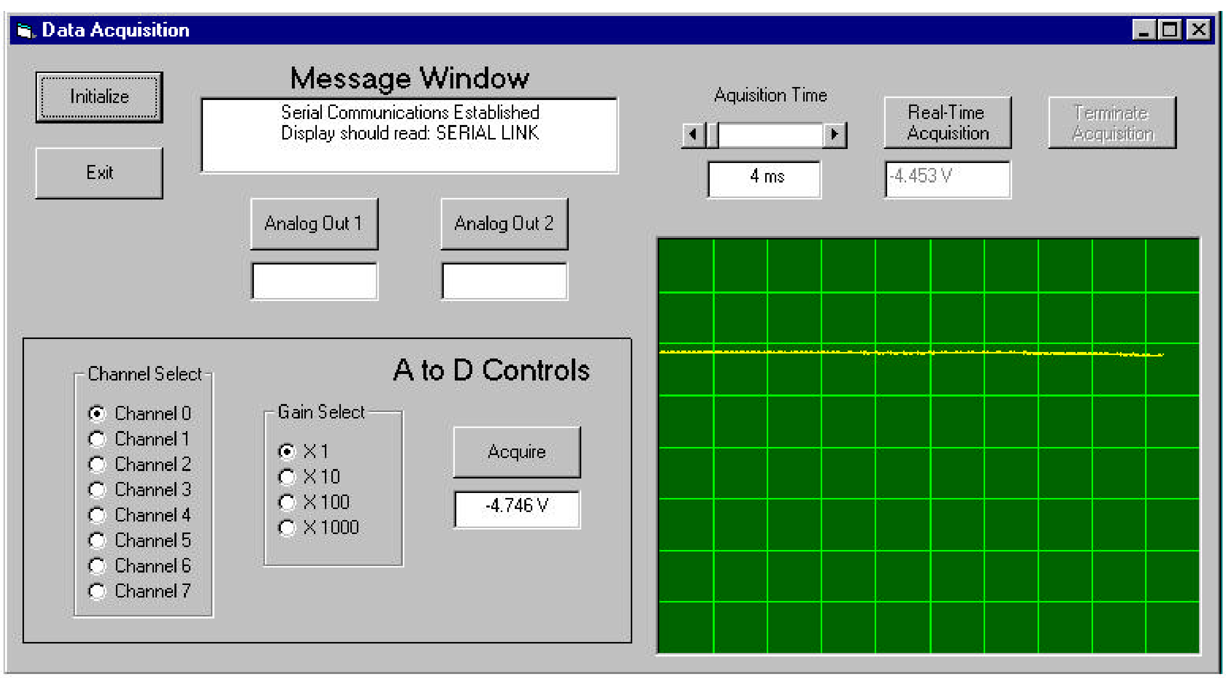

Figure 3: Typical Visual Basic interface to the UC^2 system.

With an underlying focus on integration, the new mechatronic course at Union College emphasizes the fundamental technologies on which contemporary mechatronic designs are based; sensors and actuators, system dynamics and control, analog and digital electronics, microcontroller technology, interface electronics and real-time programming.

The laboratory sessions focus on small, hands-on interdisciplinary design projects in which small teams of students configure, design, and implement a succession of mechatronic subsystems, leading to system integration in a final project. For example, as an introduction to digital design, students apply the fundamental principals of combinatorial and sequential logic to the design of a dedicated digital circuit to interface a quadrature incremental encoder to a microcontroller. The design is implemented and tested using the CPLD on the UC^2 system and the associated software development tools. As part of the final design project, the interface circuit and encoder are integrated with a DC servomotor / lead-screw assembly to construct a servomechanism which controls one axis of a simple machine tool.

Conclusions

The UC^2 system is both technologically and pedagogically innovative. In terms of pedagogy, in provides a new avenue of application for laptop computers in the undergraduate curriculum. In terms of technological innovation, the system is unique in that it functions as a data acquisition system, stand-alone controller or data logger. It incorporates both a user programmable microcontroller and a user configurable Complex Programmable Logic Device (CPLD).

The system was specifically tailored to meet the needs of engineering students at Union College. It is envisioned that it will be used throughout the engineering curriculum, beginning with the freshman *Introduction to Engineering* course. At the junior and senior levels it is designed to support laboratories in *Dynamics of Systems* and *Mechatronics Design*. Finally, as a stand-alone controller, it could potentially be applied to numerous mechatronic student projects including the International Virtual Design Studio (IVDS), the Robotics Club, the SAE Walking Machine Challenge as well as several capstone design projects each year.

Bibliographical Information

1. Crynes, B.L., "Universal Student Computer Access – Requiring Engineering Students to Own Computers," *Journal of Engineering Education*, October 1997, pp. 301-304.

2. Murray, W.R., and Garbini, J.L., *Embedded Computing in the Mechanical Engineering Curriculum: A Course Featuring Structured Laboratory Exercises,"* July 1997, pp. 301-304.

3. IOtech PC-Cards: A/D Cards for Notebook PCs: http://www.iotech.com/catalog/daq/catpcm1.html.

4. IOtech Personal Daq USB Data Acquisition Modules: http://www.iotech.com/catalog/daq/persdaq.html.

5. National Instruments E Series Multifunction DAQ: http://ww.ni.com.

Biographical Information

NICHOLAS KROUGLICOF joined the Mechanical Engineering Department at Union College in September 2000. Previously, he was a faculty member at the École de technologie supérieure in Montreal. He has taught and developed laboratories for a number of undergraduate courses relating to system dynamics and control, mechatronics, automation, and CAD. His research interests are in the areas of machine vision, intelligent sensors, and mechatronics.