September 2003

Internet-Based Remote Control using a Microcontroller and an Embedded Ethernet Board[1]

by

Imran Ahmed, Hong Wong, and Vikram Kapila
Department of Mechanical, Aerospace, and Manufacturing Engineering
Polytechnic University, Brooklyn, NY 11201
Voice: (718) 260-[3791, 3791, 3161]
Fax: (718) 260-3532
Email: [iahmed03@utopia, hwong01@utopia, vkapila@duke].poly.edu

Abstract

In this paper, we describe a recently developed DC motor position control experimental setup that can be accessed via the Internet. The experiment consists of two primary elements communicating with each other: *i)* a *server* consisting of a low-cost microcontroller, Parallax's 40-pin Basic Stamp 2 (BS2P40), interfaced with an embedded ethernet IC, Cirrus Logic's Crystal CS8900A, and *ii)* a *client* computer. The client computer sends/receives data to/from the microcontroller using UDP packets. The client computer connects to the server using Java applets that allow the user to command the position of the motor via a graphical user interface. The interface includes a slider and text input box for commanding motor position and allows values from 0—360 degrees. A plot provides a visual display of current position of motor using real-time sensor data sent by the microcontroller. Our microcontroller-based remote control methodology can be readily applied to monitor and control other experimental hardware over the Internet.

Key Words: Basic Stamp 2, embedded ethernet, GUI, microcontroller, ethernet communication

Running Title: DC Motor Control using Ethernet Enabled Microcontroller

[1] This work was supported in part by the National Science Foundation under an RET Site Grant 0227479 and the NASA/NY Space Grant Consortium under Grant 39555-6519.

1. Introduction

Microcontrollers are low-cost embedded systems that control and monitor consumer appliances, robots, machinery, etc. Microcontrollers are widely used in applications that necessitate computing power delivered within a small form factor, e.g., cellular phones, calculators, digital wristwatches, etc. Because of space limitations, microcontrollers have limited connectivity options. For example, the type of connectivity provided for data communication to a personal computer (PC) user for data visualization and parameter adjustment "on-the-fly" is usually limited to a serial port interface. This interface is limited in that it allows only one user to control the microcontroller and restricts that user to be in close proximity of the microcontroller. Enabling a microcontroller to communicate to a ubiquitous data communication network, e.g., the ethernet network, will allow developers and end-users to monitor and control microcontroller operated devices with greater flexibility.

In recent years, the ethernet network protocol has been widely adopted as the choice method of data communication for personal computers and other digital devices. Its popularity is primarily due to the immense use of the Internet, an information exchange infrastructure that communicates data via the ethernet network, by the general public. Furthermore, ethernet communication is readily available on most of the currently deployed PCs. As a data communication protocol, the ethernet is efficient. In all ethernet networks, devices can easily communicate at speeds of about 10 megabits-per-second, with some of the most recent ethernet networks communicating even at data speeds of 1 gigabit-per-second.

In this paper, we address the issues of *i)* imparting ethernet capabilities to a Basic Stamp 2 (BS2) microcontroller by interfacing it with an embedded ethernet board (EEB) and *ii)* providing a Java-based graphical user interface to control and monitor the BS2. Capabilities of this integrated BS2 and EEB hardware are illustrated by using it to command and monitor a direct current (DC) motor test-bed remotely (see http://mechatronics.poly.edu/EEBPaper/). Although in this paper we focus on a 40-pin BS2 microcontroller, namely, BS2P40, our approach is applicable to any microcontroller possessing 16 or more digital input/output (I/O) pins.

This paper is organized as follows. First, in Sections 2 and 3, we describe the hardware environment and the software environment, respectively, used in this paper. Next, in Section 4, we present an illustrative example of the ethernet enabled BS2P40 controlling a DC motor. In section 5, we propose enhancements to be made to our hardware and software environment for the final version of this paper. Finally, some concluding remarks are given in Section 6.

2. Hardware Environment

In this paper, a microcontroller interfaced with an EEB is used to control a DC motor. The microcontroller interfaces with the DC motor position sensor using an analog to digital converter (ADC) and with the DC motor using a digital to analog converter (DAC). Ethernet data communication between the microcontroller and a remote web-client is performed using the EEB. Specifically, the EEB receives reference commands from the remote web-client and communicates the same to the microcontroller. In addition, the EEB receives sensory data from the microcontroller and communicates the same to the remote web-client. See Figure 1 for a schematic of the hardware environment.

Figure 1: Hardware environment schematic

2.1. Microcontroller

In this paper, we focus on the BS2P40 microcontroller installed on a BS2P40 Demo Board development platform. These devices are manufactured by Parallax, Inc. [1]. The BS2P40 is a 40-pin Dual Inline Package (DIP) integrated circuit (IC) [2]. It is based on Ubicom Inc.'s SX48AC microcontroller. The BS2P40 is powered by a 6-14V direct current (VDC) power supply. A voltage regulator on the BS2P40 provides a steady 5VDC supply. The BS2P40 comes with ROM, 16KB Electronically Erasable Programmable ROM (EEPROM), and a small amount of RAM. The BS2P40 is programmed in PBasic language; the instruction set that is permanently stored on the BS2P40 ROM. The user-defined program is downloaded into the EEPROM from a PC through a DB-9 serial cable connection between the PC and the demo board. The excess EEPROM can be used for long-term data storage. The BS2P40 has 32 general-purpose digital I/O pins that are user defined. The high position on a digital I/O pin refers to a 5VDC and a low position on a digital I/O pin refers to a 0VDC (ground potential). Each pin can source (supply) or sink (draw) a maximum current of 30mA. The 32 I/O pins on the BS2P40 at any given time can source/sink a maximum of 60mA. See [2] for more details on BS2P40 hardware features.

2.2. Direct Current Motor Test-bed

The DC motor test-bed consists of an armature controlled DC motor, a continuous rotation potentiometer, a rotary optical encoder, a tachometer, and a power amplifier. This test-bed, shown in Figure 2, is manufactured by Quanser Consulting Inc. [3].

Both the potentiometer and the encoder sensors measure angular position of the DC motor. In this paper, only the potentiometer sensor is used to obtain angular position measurement of the DC motor. The potentiometer outputs a voltage signal relating to the DC motor angular position within the range of +/-5VDC. The BS2P40 controls the DC motor angular position by applying a controlled voltage signal.

2.3. Miscellaneous Electronics

The DC motor test-bed sends and receives analog signals from the microcontroller using an LTC1296 12-bit ADC IC and a MAX537 12-bit DAC IC, respectively. The two ICs are controlled by the BS2P40 via serial communication. The LTC1296, manufactured by Linear Technology Inc [4], is a 12-bit ADC (11-bit plus an additional sign bit) that has 8 single input channels, which can be used as 4 differential inputs, and requires a +/-5VDC power supply. The LTC1296 serves as a signal interface between the DC motor potentiometer and the BS2P40, converting voltage signal from the potentiometer to 12-bit data representation. The MAX537 IC, manufactured by Dallas Semiconductor Inc [5], is a 12-bit

DAC (11-bit plus an additional sign bit) that has 4 single output channels, which can be used as 2 differential outputs, and requires a +/-5VDC power supply. The MAX537 serves as a signal interface between the BS2P40 and the DC motor, converting a 12-bit data representation of the control voltage to a continuous voltage to the DC motor. A MAX764 DC-DC inverter, manufactured by Dallas Semiconductor Inc [6], powered by the BS2P40 demo board's +5VDC power supply, is used to obtain a +/-5VDC power supply for the LTC1296 and MAX537.

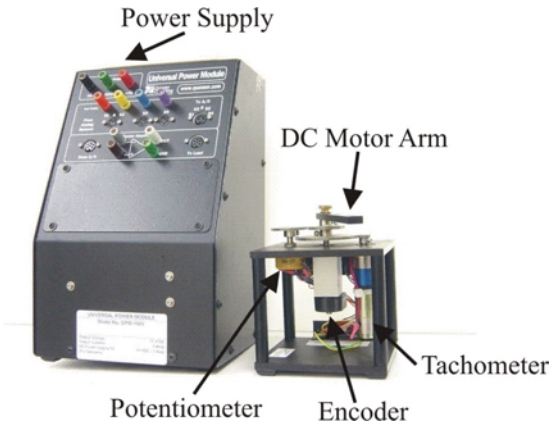

Figure 2: DC motor test-bed

2.4. Embedded Ethernet Board

A 10Base-T EEB, manufactured by Embedded Ethernet.com [7], provides bi-directional ethernet communication capability to the BS2P40. The EEB is an 18 pin DIP IC that can be interfaced with the BS2P40 using 15 of its 32 I/O pins. It is powered by a 5VDC power from the BS2P40 demo board and has an RJ-45 port for connection to an ethernet cable. At the core of the EEB is the Crystal CS8900A IC, manufactured by Cirrus Logic. The CS8900A on EEB provides interfaces for the 8-bit data bus (D0—D7), 4-bit address bus (A0—A3), and 4 control signals (AEN, WR, RD, IRQ). See Table 1 for the EEB pin assignments.

The EEB data bus allows for binary data communication to and from the microcontroller, where these basic send and receive functionalities are controlled by sending commands to the EEB read and write pins. In addition, the EEB address bus allows the microcontroller to control the memory of the CS8900A. Figure 3 shows a picture of the EEB.

Table 1: Pin assignments for the embedded ethernet board

Pin(s) #	Label	Signal Type	Functionality
1	Vcc	Input	5VDC
2	GND	Input	Ground
3—10	D7—D0	Input/Output	CS8900A Data Bus
11	/AEN	Input	CS8900A Address Enable (Active low)
12	/WR	Input	CS8900A Write control (Active low)
13	/RD	Input	CS8900A Read control (Active low)
14	IRQ	Output	Interrupt signaled by this pin going high
15—18	A3—A0	Input	CS8900A Address Bus

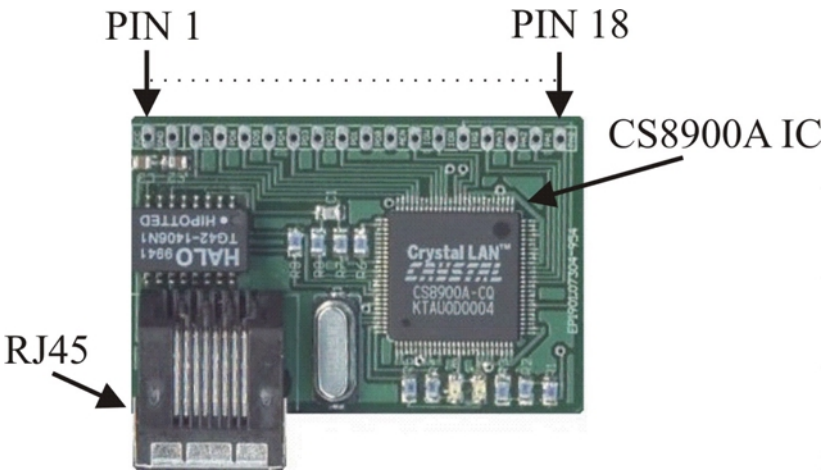

Figure 3: The embedded ethernet board

Figure 4 shows the pin connections between the EEB, the BS2P40 demo board, and the BS2P40.

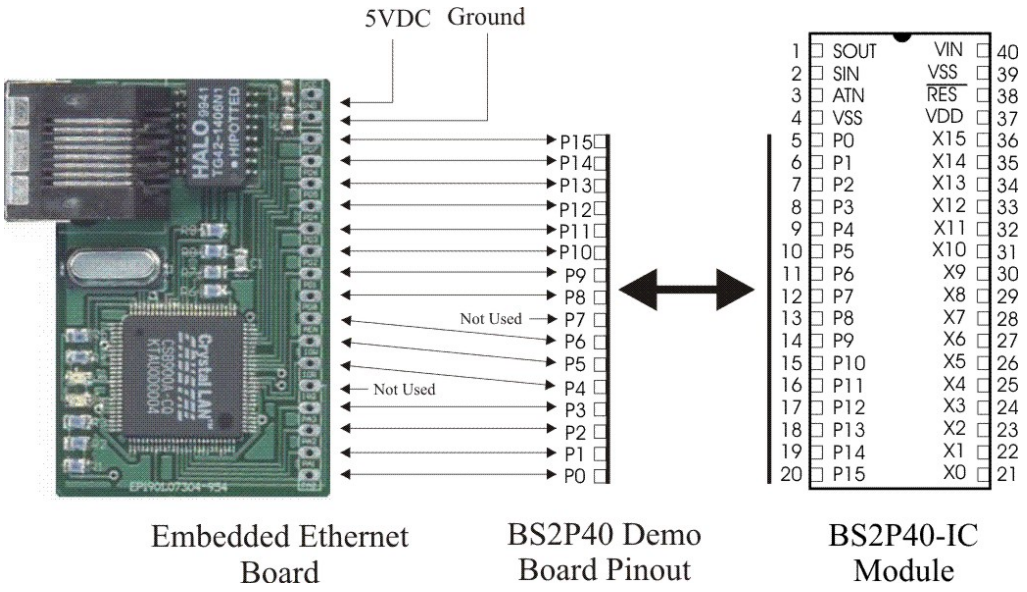

Figure 4: Pin connections between the EEB to BS2P40

3. Software Environment

The software environment of this paper consists of primarily PBasic and Java. Ethernet communication between the BS2P40 and PC is accomplished by implementing a User Datagram Protocol (UDP) in PBasic. The PBasic program initiates and performs all data exchange to and from a PC via the EEB including encoding of sensory data, which is to be sent to a remote client, in UDP format and decoding of UDP format position command, which is received from the remote client. See [8] for more details on network communication using the UDP. A feedback control algorithm for DC motor control is also implemented in PBasic. A Java applet, executing on a remote client, serves as a gateway to the BS2P40. Specifically, the Java applet presents a graphical user interface (GUI) to the remote client

through which the user can give DC motor angular position commands to the BS2P40 and visualize sensory data received from the BS2P40. Next, we describe various elements of software environment used in this paper.

3.1. Ethernet Data Communication

The BS2P40 microcontroller communicates with the remote client by creating, interpreting, sending, and receiving UDP datagram packets. A UDP datagram packet is a sequence of binary bits sent via the ethernet transmission wire. In the initial phase of the datagram packet, an IP and UDP header bit sequence is sent that contains information about the destination address, the packet origin address, and the data checksum. The final phase of the datagram packet contains the data bit sequence. Figure 5 shows a generic UDP packet. The UDP is a choice method of ethernet communication because of the datagram packet's compact size compared to other transmission protocols, e.g., Transmission Control Protocol (TCP).

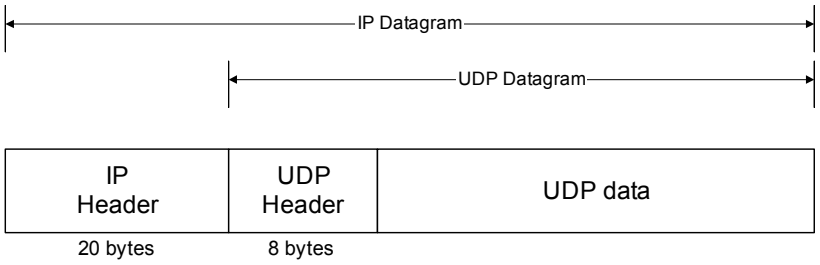

Figure 5: UDP encapsulation

3.2. PBasic Program

The BS2P40 is programmed using the PBasic programming language. It is a Basic-like language developed by Parallax, Inc. for the BS2 series microcontrollers. In addition to simple arithmetic, the BS2P40 executes certain task specific commands. See [9] for more details on PBasic programming language. We now provide a summary of our PBasic program.

In the first step, the PBasic code declares various variables used in the program in addition to the locations of various registers of the Crystal CS8900A. These registers are used to read and write ethernet packets. In the second step, the PBasic code initializes the EEB so that the EEB is ready to send or receive UDP datagram packets. In order to send data of arbitrary size via the ethernet, UDP datagram packets are sent 8-bits at a time until all the data is transmitted. Our data to be transmitted by the BS2P40 is 4-bytes long and contains the current position of DC motor (2-bytes) and corresponding sample number (2-bytes). In addition, the data to be received by the BS2P40 is 2-bytes long and contains the commanded position of DC motor. In the third step, the microcontroller waits for an incoming packet. The fourth step of the PBasic code begins when the microcontroller receives a datagram packet. In this step, the microcontroller decodes the contents of the datagram packet information. The decoded information contains a user-defined 2-bytes long flag and an angular position command. The user-defined flag is used to ascertain the source of the datagram packet. In the fifth step, the PBasic code saves the received DC motor angular

position command and determines the current DC motor position from the potentiometer sensor. Next, an appropriate feedback control algorithm, e.g., proportional control, is used to compute the control voltage that needs to be applied on the DC motor. In the sixth step, the PBasic code transmits the control voltage to the DC motor via the DAC and power amplifier. In the seventh step, the PBasic code encodes and transmits a datagram packet containing the previously obtained DC motor position. Finally, the PBasic code loops back to the third step. See Figure 6 for a flow diagram of the PBasic code.

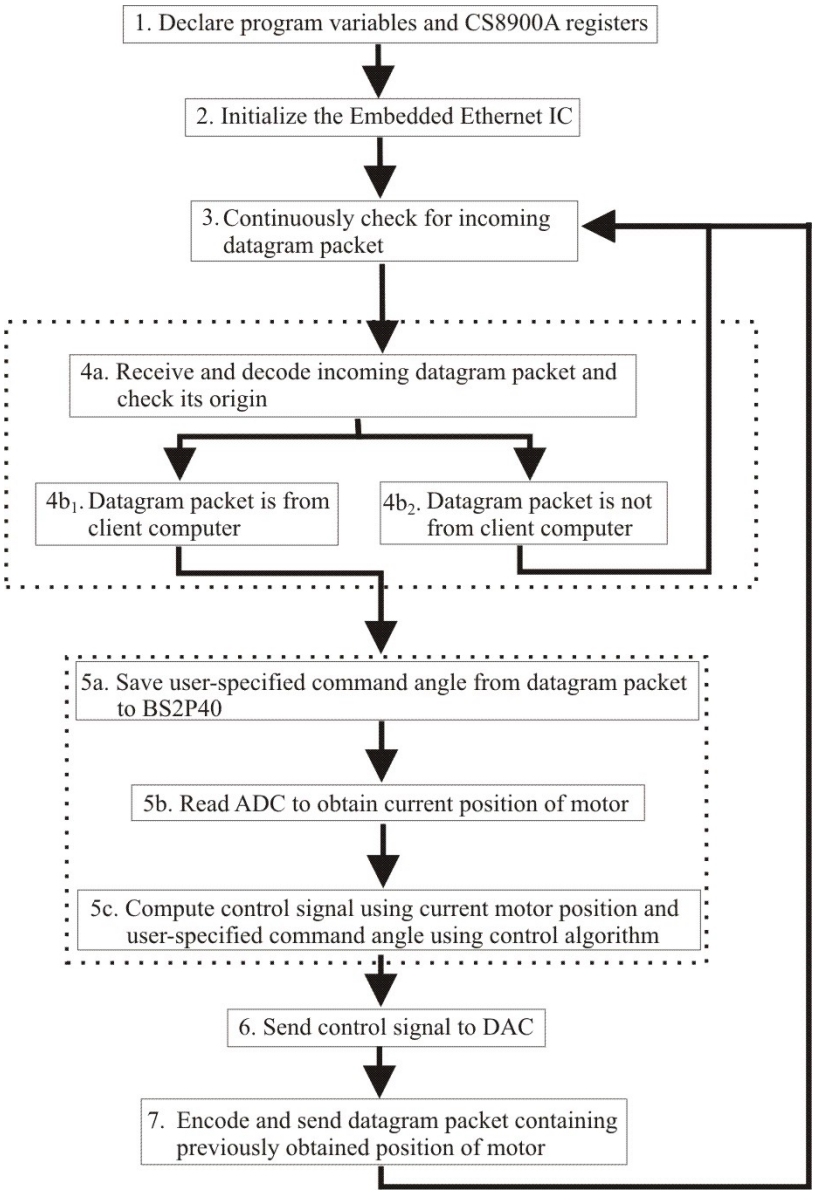

Figure 6: Flow of events in the PBasic program

3.3. Java Interface

A GUI, which runs on a remote client computer, has been created using the Java applet technology [10]. This GUI allows remote clients using any Java compatible operating system to interact with our experiment test-bed. The GUI consists of three Java applet libraries and one main Java program.

The first Java applet library includes functionality for a horizontal slider bar. This object is dragged left or right to specify 0º—360º command for DC motor angular position. The user-selected angle on the slider bar is written to a text file. The second Java applet library includes functionality for a text box to receive user input. The user enters 0º—360º command for DC motor angular position in the text box and then clicks the "Send" button on GUI to submit his/her input. Similar to the slider bar object, the user-selected angle from the text box object is written to a text file. The third Java applet library is the PlotLive component [11]. This object reads the current position of DC motor obtained from the BS2P40 and plots it against the corresponding sample number.

The main Java program integrates the aforementioned three Java applet libraries for data manipulation and display. In addition, using Java's network data communication library, the main program transmits and receives the commanded DC motor position data and current position of DC motor, respectively, to/from the BS2P40. See Figure 7 for a flow diagram of the main Java program.

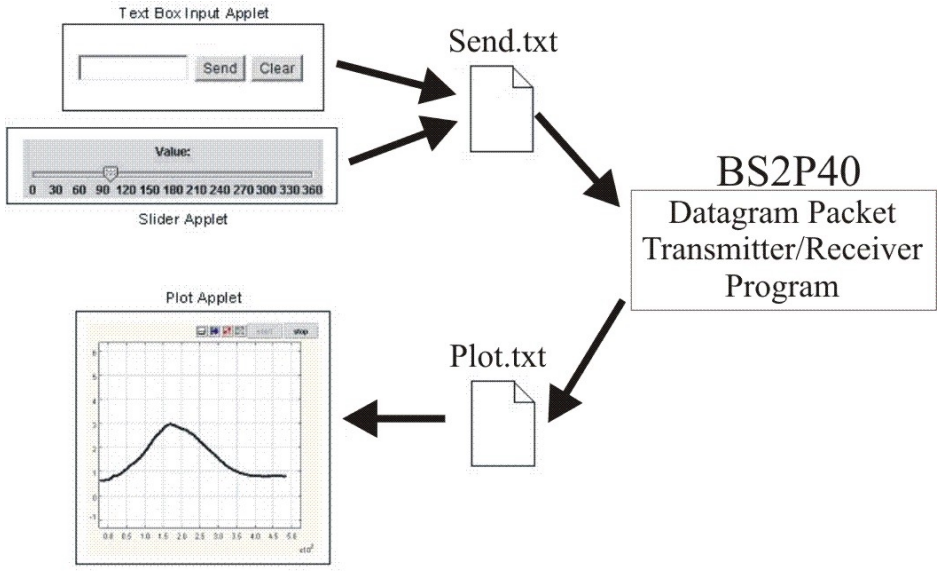

Figure 7: Java program interaction

4. Illustrative Example

In this example, a step command, provided by a remote web-client running a Java applet GUI, changes the command angle of the DC motor arm. First, the user commands the DC motor to maneuver from 50 degrees to 100 degrees using the horizontal slider on the GUI. Next, the user commands the DC motor to maneuver from 100 degrees to 250 degrees using the text box on the GUI. Figure 8 shows the results of the GUI interface after performing the specified command angle changes. The reader can

remotely access and evaluate current version of our experimental setup by accessing and following the instructions on http://mechatronics.poly.edu/EEBPaper/.

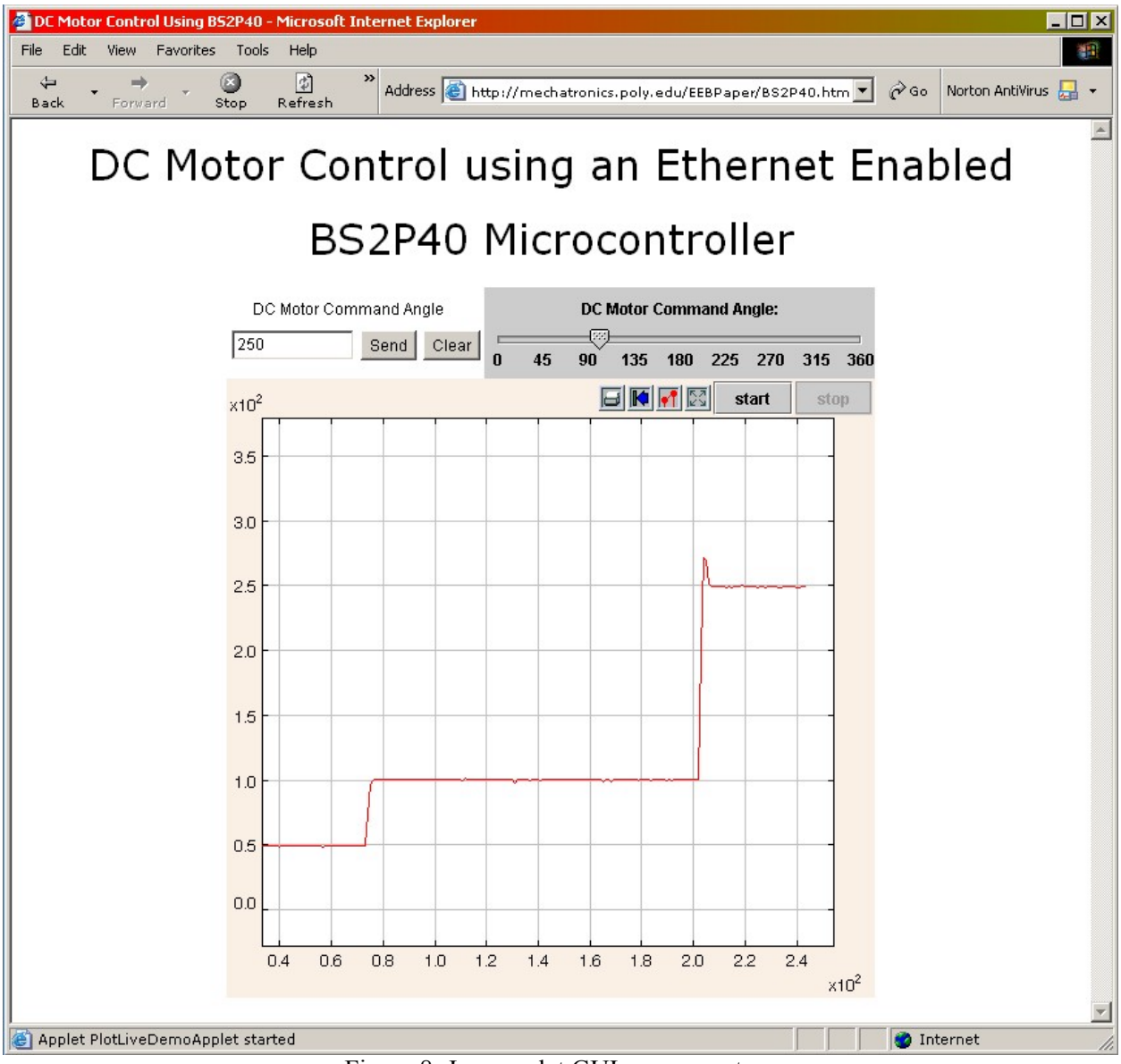

Figure 8: Java applet GUI screen capture

5. Planned Enhancements

The following hardware and software enhancements are planned to extend the current capabilities of our ethernet-enabled microcontroller environment and will be reported in the final version of this paper.

1. Design and implement a PD control algorithm for DC motor control.
2. Allow the remote user to input PD control gains via the Java GUI.

3. Enhance the Java GUI to host additional user-interface and data visualization tools e.g., 3-D plots, miscellaneous controls (e.g., switches, knobs, etc.), and VRML-based 3-D animations.
4. Implement a scheme for user prioritization/queue.
5. Enable software-based switching of control architectures (PD, PID, LQR, etc.).
6. Integrate a low-cost embedded web server with our current hardware to provide web-hosting capabilities.
7. Provide live streaming video of online experiments in motion.

6. Conclusion

In this paper, we exploited an EEB for ethernet data communication between a BS2P40 and a remote web-client PC. Utilizing UDP datagram packets, DC motor position sensor measurements were transmitted from the BS2P40 to the remote web-client PC while command angles were transmitted from the remote web-client PC to the BS2P40. In addition, a proportional controller was implemented on the BS2P40 to control the DC motor arm. On the remote web-client PC, a Java applet GUI was provided for data visualization and command adjustment "on-the-fly." Specifically, a generic Java plotting library was used to plot the DC motor position history and generic Java horizontal slider and text box libraries were used to send command angles to the proportional controller on the BS2P40. This experiment illustrates the Internet capabilities imparted to a microcontroller by the use of embedded ethernet for data communication and Java for GUI functionality.

References

[1] Online: http://www.parallax.com/, website of Parallax, Inc.

[2] Online: http://www.parallax.com/detail.asp?product_id=BS2P40-IC, website of Parallax, Inc. developer and distributor of 40 pin Basic Stamp 2 (BS2P40) microcontroller (access link for BS2P40 product information).

[3] Online: http://www.quanser.com/english/html/products/fs_product_challenge.asp?lang_code=english&pcat_code=exp-rot&prod_code=R1-posserv, website of Quanser Consulting Inc. developer and distributor of the DC motor test-bed (access link for product information).

[4] Online: http://www.linear.com/prod/datasheet.html?datasheet=324, website of Linear Technology Inc. developer and distributor of the LTC1296 ADC (access link for product information).

[5] Online: http://www.maxim-ic.com/quick_view2.cfm?qv_pk=1125, website of Dallas Semiconductor Inc. developer and distributor of the MAX537 DAC (access link for product information).

[6] Online: http://www.maxim-ic.com/quick_view2.cfm/qv_pk/1171/ln/en, website of Dallas Semiconductor Inc. developer and distributor of the MAX764 DC-DC invertor (access link for product information).

[7] Online: http://www.embeddedethernet.com, website of Embedded Ethernet.com developer and distributor of the Embedded Ethernet Board (access link for product information).

[8] W. R. Stevens, *TCP/IP Illustrated Volume 1*, Addison-Wesley, Boston, MA (1994).

[9] *Basic Stamp Programming Manual*, v2.0c, Parallax, available at http://www.parallax.com/dl/docs/prod/stamps/basic%20stamp%20manual.pdf.

[10] Online: http://java.sun.com/, website of Java.

[11] Online: http://ptolemy.eecs.berkeley.edu/ptolemyII/, website of Ptolemy II a Java package containing plotting libraries (access link for product information).