Experiments in a Microprocessors and Microcontrollers Course

Ray Bachnak
Texas A&M University-Corpus Christi

Abstract

Laboratory experimentation is an essential component of a comprehensive learning experience in engineering technology programs. In fall semester 2000, we developed and taught Microprocessors and Microcontrollers, a required junior-level course in the Control Systems Engineering Technology program, by employing a set of software and hardware experiments. The purpose of the laboratory exercises is to introduce students to the practical aspects of microprocessors and microcontrollers and help with achieving the course objectives. This paper briefly describes the course, presents the experiments, and shows how students were exposed to a wide range of concepts and applications. Diagrams are shown and described, as necessary.

I. Introduction

Courses that teach microprocessors and microcontrollers are found in most engineering and engineering technology curricula. Typically, two or more courses are used to cover the various aspects of microprocessors and microcontrollers [1-7]. In the Control Systems Engineering Technology program at Texas A&M University-Corpus Christi (A&M-CC), however, the curriculum has only one course devoted to this area. Our goal is to introduce students to the practical aspects of microprocessors and microcontrollers through laboratory experimentation. A set of experiments that involve programming, hardware, interfacing, and applications was developed. The experiments range from simple software exercises that illustrate the use of the assembler and instruction set to more complex and practical applications such as interfacing and controlling external devices. This paper briefly describes the course, presents the experiments, and shows how students were exposed to a wide range of concepts and applications. The paper will also present initial course assessment and future plans.

II. Course Objectives

ENTC 4418 Microprocessors and Microcontrollers was developed and offered for the first time in fall semester 2000. It is a four-credit hour course with a weekly laboratory of three hours. Topics covered include hardware architecture, addressing modes, instruction set, programming, I/O operations, interrupts, timing, memory, peripheral interface devices, control of external devices, and applications. The prerequisites for the course are two courses in DC/AC circuit analysis, a course in digital logic, and a course in C language programming. Two textbooks were used for the course [8, 9]. The initial plan was to cover chapters 1-8 and 10–11 in the first book and Chapters 1-13 in the second book. Our goals were stated in the syllabus as follows. At successful completion of this course, students will be able to: (1) State the major features of the 8051, 8088/86, 80286, 80386, 80486, and Pentium; (2) Use the 8086/8051 addressing modes and instruction set; (3) Write assembly language programs; (4) Use DOS and BIOS interrupt services; (5) Create memory mapped I/O ports; (6) Read/write from/to input/output ports; (7) Program the 8051 for control applications; (8) Design microprocessor and microcontroller interface circuits. In our first attempt, we were partially successful in achieving the goals of the course and associated outcomes.

III. Laboratory Environment and Experiments

The three-hour weekly laboratory session gives students hands-on experience with both hardware and software. The 8086 microprocessor and 8051 microcontroller were used as practical examples. Two software packages that run on a PC were available: Macroassembler MASM 6.11 from Microsoft and PK51 Professional Development Kit from Keil. The microcontroller board has a C504 Siemens chip with four 8-bit I/O ports, 10-bit A/D converter, three timers, and an 8-bit USART. A total of 11 exercises were performed throughout the semester. The following paragraphs briefly describe the activities and learning objectives of each exercise.

Exercise 1: Students learn about microprocessors and their programming by going through an exercise that has more than 10 steps that illustrate the use of many of the DOS DEBUG commands.

Exercise 2: Students perform two major activities. First, they configure their computer to use MASM 6.11 by editing the config.sys and autoexec.bat files, and setting Programmer's WorkBench and CodeView options. Second, they create a template program, compile it, and then build the executable file. The template program, with all instructions removed, is shown in Fig. 1.

Exercise 3: Students learn about the addressing modes by tracing the execution of instructions and monitoring their effects. This is done in CodeView, where they also verify the segments referenced and the effective addresses used.

Exercise 4: Students examine the effects of executing many of the 8086/88 instructions by tracing the execution of a program in CodeView. The program includes a call to a procedure that converts from upper case to lower case letters.

Exercise 5: Students write their own program that reads a list of signed numbers and calculates the average value.

Exercise 6: Students write a program that multiplies two signed variables. The first variable is byte-size and the second is word-size. The result is stored in a double-word-size variable.

<u>Exercise 7:</u> Students practice with removing and replacing the PC Power Supply, floppy disk drive, and hard disk drive. Students write the steps they go through to perform their tasks.

```
;             Student Name
;             ENTC 4418 Microprocessors and Microcontrollers
;
              Page 60, 132
;*****************************************************
;*            Program description goes here
;*****************************************************
TITLE       LAB2.ASM
;--------------------------------------
;                   constants are defined here
;--------------------------------------
STACKSEG    SEGMENT PARA STACK ’STACK’
         DW     256 DUP(?)
STACKSEG    ENDS
;--------------------------------------
DATASEG     SEGMENT PARA PUBLIC ’DATA’
;                     variables are defined here
DATASEG     ENDS
;--------------------------------------
CODESEG      SEGMENT PARA PUBLIC ’CODE’
BEGIN          PROC FAR
         ASSUME SS:STACKSEG, CS:CODESEG, DS:DATASEG, ES:DATASEG
                  MOV   AX, DATASEG
                  MOV   DS, AX     ; Load DS for this program
; --------------------------------------------------
;                   code: instructions go here
; ----------------------------------------------------
                  MOV    AH, 4CH    ;return to
                   INT      21H                    ;DOS
BEGIN          ENDP
CODESEG      ENDS
END              BEGIN
```

Fig. 1 Template Program

<u>Exercise 8:</u> Students remove and replace the PC motherboard and learn how to locate and identify features of the microprocessor, RAM, and ROM chips. Students learn about other issues such as the importance of grounding the system unit by connecting it through a wire to ground, in order to reduce the possibility of damaging the chips due to static electricity.

<u>Exercise 9</u>: Students learn about the design and programming of a 6821 interface circuit. The circuit for this exercise is shown in Fig. 2. The address decoding section consists of a 74LS138 along with a 74LS260, and a 74LS00. Output Y3 of the ‘138 is asserted low only when any of the desired addresses (318H, 31AH, 31CH, or 31EH) appears on the address bus. Address bit A0 is used to select the low bank addresses when used with sixteen-bit bus, i.e. 8086 microprocessor. Along with the proper address bits, the PC bus signal AEN must be asserted low in order to properly assert the desired Y3 output of the

‘138. The reason for the inclusion of the AEN signal is to disable I/O port address decodes during DMA bus cycles.

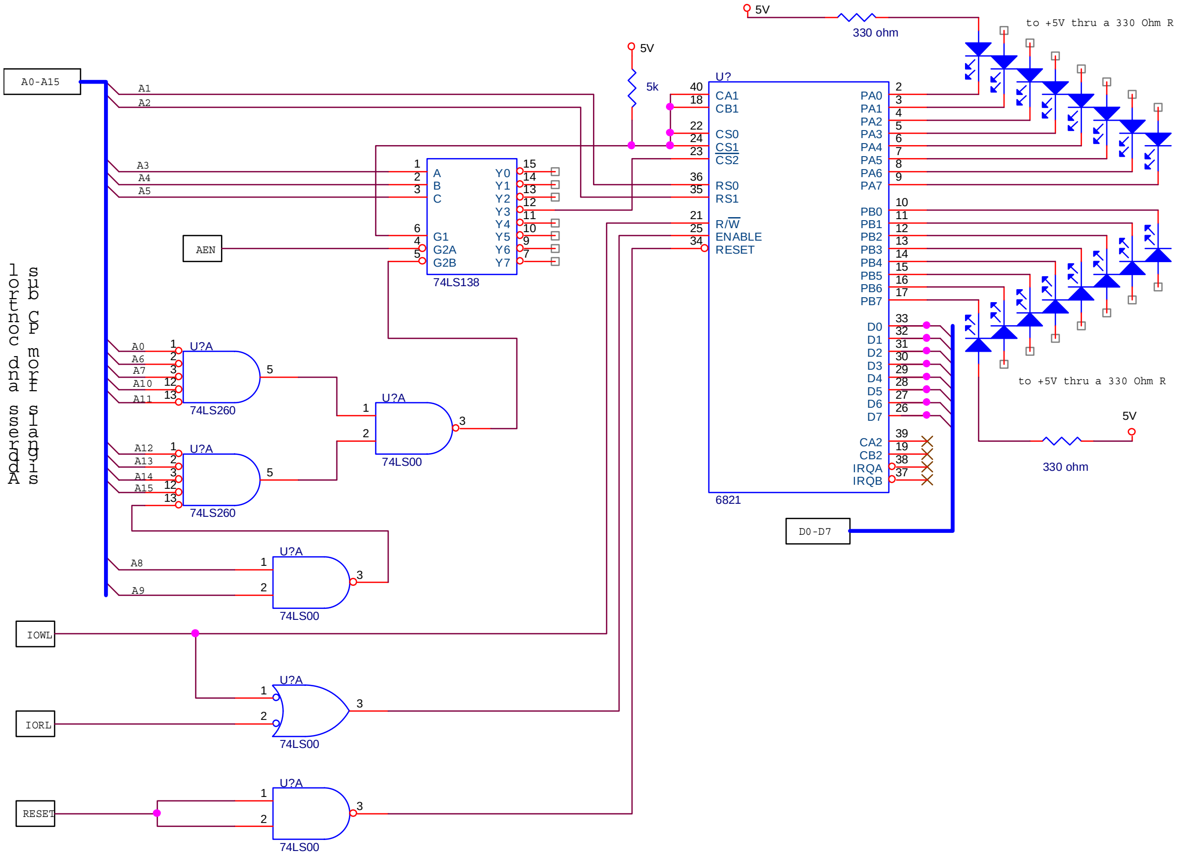

Fig. 2 Designing and Programming a Peripheral Interface Device Circuit

The MC6821 is selected through pin Y3 of the 74LS138 decoder that asserts the CS2 pin of the MC6821. PC-Bus signal IOW is inverted and applied to the ENABLE pin of the MC6821. The reason for this is that the I/O port is used as a write only operation since we do not wish to perform read operations. All other MC6821 chip select pins are not used and are tied high. All data bits, PA0-PA7 and PB0-PB7, are monitored by the use of LEDs. A logic 0 on one of these pins will illuminate the LED, while a logic 1 will turn off the LED. Address bits A1 and A2 are used to select the internal registers of the MC6821. The control register bits, CRA2 and CRB2, are used in conjunction with RS1 and RS0 to program the MC6821 and select the desired internal registers. The board has the following address assignments: PORTA=318H, CRA=31AH, PORTB=31CH, CRB=31EH. To verify the operation of the circuit, students write a program that accomplishes the following: (1) Configures all bits of PORTA and PORTB to be output lines; (2) Sends to PORTA and PORTB various patterns of 0s and 1s (a "0" turns an LED on and a "1" turns it off). The patterns are sent out one at a time with delays inserted between them to allow observing the status of the LEDs.

Exercise 10: Students complete a program that displays on the screen a menu of three items and waits for the user to enter a selection. The program reads the user's input, and displays corresponding information. Several INT 10H and INT 21H software interrupt functions (setting cursor, display to screen, reading from keyboard, clearing screen, etc.) are used to accomplish various tasks.

Exercise 11: Students learn how to access the printer parallel port on a PC. Fig. 3 shows the circuit used for this exercise. Students are asked to write a program that outputs data to the parallel port and test the program using the following data patterns: FFH, 55H, 00H. The patterns are sent out one at a time with delays inserted between them to allow observing the status of the LEDs and the fan. The parallel port is accessed in two ways [10]. By directly outputting to address 0378H as follows.

```
MOV DX, 378H        ;move printer address into DX
MOV AL, 55H         ;move data into AL
OUT DX, AL          ;output data to port 0378H
```

or by using the BIOS service routine as follows.

```
MOV DX, 0           ;specify printer number
MOV AH, 0           ;specify BIOS function
MOV AL, 55H         ;move data into AL
INT 17H             ;execute BIOS function
```

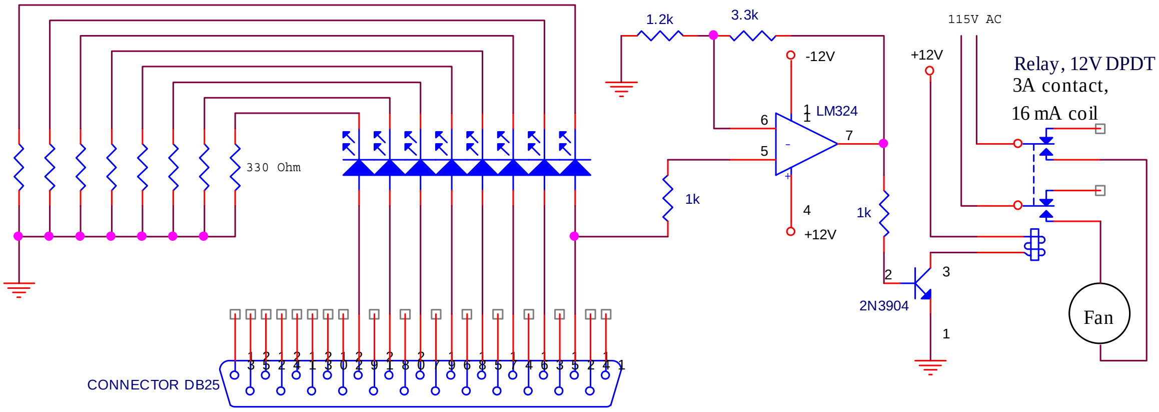

Fig. 3 Designing and Programming a Parallel Printer Port Circuit

Due to a lack of time, students did not get a chance to experiment with the C504 microcontroller board. However, students learned about its features and capabilities. Chapters 1-8 in the textbook [9] were covered and students wrote code that accesses the 8051 I/O ports. A simple circuit that illustrates the use of microcontrollers in interfacing applications is shown in Fig. 4. This circuit simply reads the status of the switch and turns the motor on or off, accordingly. Here is the code that accomplishes the operation.

```
BACK:    SETB  P1.6          ; make P1.6 input
         MOV  A, P1          ; read contents of Port1 into ACC
         JBC    ACC.6, NEXT  ; jump if ACC.6 is set
         CLR P1.7            ; clear P1.7
         SJMP BACK           ; keep reading
 NEXT:   SETB P1.7           ; set P1.7
         SJMP BACK           ; keep reading
```

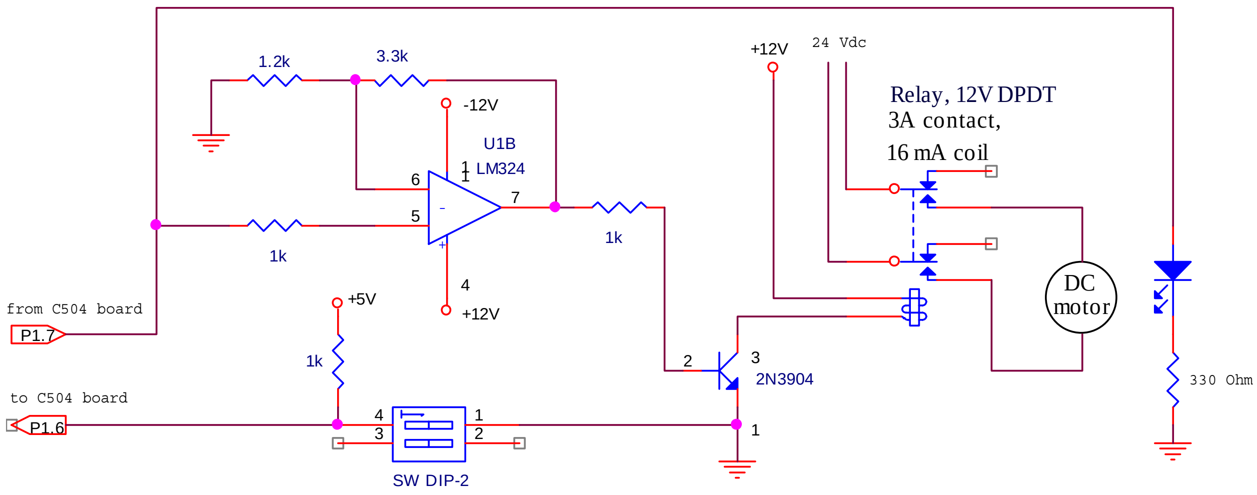

Fig. 4 A Simple I/O Microcontroller Circuit

IV. Students' Feedback and Future Plans

A major challenge that the instructor encountered while teaching this course was covering a wide range of topics at a level acceptable for a senior level course. Students started complaining about too much work around the fourth week of the semester. Realizing that it was impossible to adequately teach both microprocessors and microcontrollers in one course, adjustments were made so that while microcontrollers were taught, the laboratory experiments dealt mainly with the 8086 microprocessor.

A student survey was conducted at the end of the semester. Feedback was in general positive but most students felt that the course addressed too many issues. Given the complexity of microprocessors and microcontrollers, it is clear that a redesign of the course is necessary. A possibility is to rearrange the weekly schedule so that microcontrollers are emphasized and students have proper hands-on experience in their use. This seems to be desirable due to the nature of the CSET program and the popularity of microcontrollers in many areas such as consumer electronics, industrial control, and communication. The weekly schedule may be revised as the follows.

- The first two weeks of the semester are devoted to discussing the basic fundamentals and programming of microprocessors and microcontrollers.
- The next 7 weeks of the semester are devoted to teaching microcontrollers, with emphasis on laboratory experimentation.

- The last 6 weeks of the semester will be devoted to teaching microprocessors, without lab exercises.

The instructor is also considering a project-based approach for the course. Projects drawn from real-world industrial control applications will allow students to improve many of the skills they need after graduation.

V. Conclusion

Laboratory experimentation is an essential component of a comprehensive learning experience in engineering technology programs. This paper describes a set of software and hardware experiments for a new microprocessor and microcontroller course. The exercises range from simple software exercises that illustrated the use of the assembler and instruction set to more complex and practical applications such as interfacing and controlling external devices. In our first attempt, we were partially successful in achieving the course objectives. Future plans include revising the course so that microcontrollers are emphasized.

Bibliography

1. R. Helps and D. Phillips, "Teaching Embedded Systems Using Multiple Microcontrollers," 1999 ASEE Annual Conference Proceedings, Session 3547, 8 pages.
2. R. Stemprok, "Teaching and curriculum development of microprocessor classes," 2000 ASEE Annual Conference Proceedings, Session 2559, 8 pages.
3. D. Smith and R. Douglas, "A revised assembly language programming course for a computer engineering technology program," 1999 ASEE Annual Conference Proceedings, Session 2548, 7 pages.
4. A. Avila and J. Hinojosa, "Integrating active and cooperative learning strategies to a redesigned microprocessor based system design course," 1999 ASEE Annual Conference Proceedings, Session 2532, 10 pages.
5. P. Neal and G. York, "MC68HC11 portable lab unit – a flexible tool for teaching microprocessor concepts," 1996 ASEE Annual Conference Proceedings, Session 3532, 6 pages.
6. R. Rockland, "Use of problem solving skills in an introductory microprocessor course," 1999 ASEE Annual Conference Proceedings, Session 3147, 9 pages.
7. Y. Liu, "Development system for undergraduate microprocessor course," 1996 ASEE Annual Conference Proceedings, Session 1626, 4 pages.
8. Triebel and Singh, The 8088 and 8086 Microprocessors: Programming, Interfacing, software, hardware, and Applications, Third Edition, Prentice Hall, 2000.
9. Mazidi and Mazidi, The 8051 Microcontroller and Embedded Systems, Prentice Hall, 2000.
10. Kenneth Ayala, The 8086 Microprocessor: Programming and Interfacing the PC, Delmar, 1995.

Biography

RAFIC BACHNAK
Rafic (Ray) Bachnak is an Associate Professor of Engineering Technology at Texas A&M University-Corpus Christi. He received his B.S., M.S., and Ph.D. degrees in Electrical and Computer Engineering from Ohio University in 1983, 1984, and 1989, respectively. Dr. Bachnak was previously on the faculty of Franklin University and Northwestern State University.