Lab Overview

In this lab assignment, you will do the following:

- Add a serial EEPROM and an LCD to the hardware developed in Labs #1, #2 & #3.
- Write simple assembly and C programs to test EEPROM accesses.
- Write assembly and C programs to perform user output to the LCD.
- Continue learning how to use the MICRO-C compiler and makefiles.

This lab assignment is due by Saturday, November 15, 2003.
The deadline for this lab is Wednesday, November 19, 2003.

This lab is weighted as 18% of your course grade.

You should be working on your final project in parallel with this lab assignment.

**NOTE:** The quality of your user interfaces will impact your score on the lab. Your goal should be to ensure that the user has a successful and positive experience with your software. Your programs should handle error conditions gracefully (e.g. user input values outside the allowable range).

Lab Details

1. Read the data sheet for the Fairchild (National Semiconductor) NM24C16 (or NM24C04) Serial EEPROM. You may also want to read Fairchild Application Note AN-794.

2. [Optional, but recommended] Review Microchip app notes AN536, AN572, AN614 and AN709.

3. Review the data sheets for the Optrex DMC 20434 LCD and the Hitachi HD44780U LCD controller.

4. Refer to the EEPROM Guide and LCD Guide available on the course web site for further ideas and information on interfacing.

5. [Required Element\(^1\)] Design and implement your EEPROM circuit. Your EEPROM should be connected to two unused port pins on Port 1 or Port 3. Verify that you can write and read data from the EEPROM and verify the stored data after cycling power. Note that since you are connecting to the EEPROM using port pins, the EEPROM does not consume any 8051 address space.

   Demonstrate the following C functions (the underlying drivers may be in assembly):
   ```
   int eebytew(addr, data)  // returns status
   int eebyter(addr)        // returns data or status
   ```

**NOTE:** It is acceptable to use the MICRO-C eeread() and eewrite() functions instead of writing your own functions. If you use these MICRO-C functions, you must use the I/O pins defined in the MICRO-C I2C.ASM library code.

Write a C program which will allow the user to read and write EEPROM data from the PC terminal emulator screen. The C program must allow the user to specify the parameters for the operation. The program must allow any hex value from 0x00 to 0xFF to be programmed into any location in the EEPROM. The EEPROM address and the data value must be entered by the user in hex. Do not make the user type in "0x" before the address or data hex value.

6. [Optional] Use the I2C triggering program on the Agilent 54622D oscilloscope to trigger on a write or read frame on the bus. Display SCL and SDA on the oscilloscope screen and verify that the transaction is for the address you intended. Verify that your rise and fall times fall within the limits given in the I2C specification. Alternatively, use a logic analyzer to trigger on a bus transaction.
7. **[Required Element]** Design and implement your LCD circuit. Your LCD should be memory mapped in the 8KB of address space reserved for peripherals. The LCD contrast (V_{EE}) is typically grounded, but you can use a potentiometer or resistor divider to control the contrast if necessary. The LCD has 14 lines which must be connected to your board. One option is to use a 14-pin strip header or SIPP wire wrap socket. Another option is a 14-pin or 16-pin DIP socket connected to a ribbon cable. **(Note: LCDs with a backlight will have 16 pins, two of which control the backlight.)**

**NOTE:** It may take you a little time to devise and implement a good physical interface between the LCD and your board so don’t wait too long before getting started on this interface. If you come up with a good way to do the interface, bring it to the attention of the instructor.

Standoffs can be used to mount your LCD above circuitry on your board. A limited supply of #2 machine screws and standoffs are available from the instructor.

The eight data signals on the LCD must be connected to the data lines on Port 0 of the 8051.

Ensure that the E signal on the LCD is high **only** when you’re reading from or writing to the LCD.

8. **[Required Element]** Implement an LCD driver with the following C functions:

- // Name: lcdinit()
  // Description: Initializes the LCD (see Figure 25 on page 212 of the HD44780U data sheet).
  void lcdinit()

- // Name: lcdgotoaddr()
  // Description: Sets the cursor to the specified LCD DDRAM address.
  void lcdgotoaddr(unsigned char addr)

- // Name: lcdgotoxy()
  // Description: Sets the cursor to the LCD DDRAM address corresponding to the specified row and column.
  void lcdgotoxy(unsigned char row, unsigned char column)

- // Name: lcdputch()
  // Description: Writes the specified character to the current LCD cursor position.
  void lcdputch(char cc)

- // Name: lcdputstr()
  // Description: Writes the specified null-terminated string to the LCD starting at the current LCD cursor position and automatically wraps long strings to the next LCD line after the right edge of the display screen has been reached.
  void lcdputstr(char *ss)

Provide a well-designed menu on the PC terminal emulator screen which allows the user to:

- Demonstrate each of the C functions listed above.
- Store String: Enter and store three separate 32-character null-terminated ASCII user strings in EEPROM. Your program must prompt the user for a string number (1-3). As the user enters the string, echo the characters to the PC screen. Your program must support the backspace key, to allow the user to correct an input mistake. Be sure that you do not allow your string to overflow the array you allocated in your C code.
- Dump Strings: Display on the PC screen all the strings stored in the EEPROM.
- Erase String: Select a user string number (1-3) and erase its contents in the EEPROM.
- LCD Display: Select a user string number (1-3) and display the string on the LCD.

**NOTE:** The MICRO-C compiler accepts two types of comment syntax: Traditional K&R C style comments, such as ‘/*’ and ‘*/’, and C++ style comments, such as ‘//’.
9. **[Supplemental Element\(^1\), 7 points max]**:

Modify your previous C program to do the following additional things:

- In the bottom right corner of the LCD, continuously display the elapsed time since your program started running using the format "MM:SS", where MM is the number of elapsed minutes and SS is the number of elapsed seconds. For example, 64 seconds would be displayed as “01:04”.

- Provide additional menu options to stop the elapsed time clock, to restart the clock, to reset the clock back to "00:00", and to change the direction of counting (up/down). If the clock is counting down, it must stop when it reaches "00:00".

**NOTE:** Make sure that the cursor location is correctly stored before and restored after any ISRs.

**NOTE:** Remember not to use any local variables from within the context of an ISR (a MICRO-C constraint). This includes any functions that your ISR calls.

**NOTE:** This supplemental element is an addition to the previous required element. The required and supplemental code must be integrated together. The elapsed timer must work correctly while simultaneously allowing all the menu options in the previous C program to work correctly.

10. **[Supplemental Element\(^1\), 7 points max]**:

**NOTE:** The following routines can be implemented as a stand-alone program, or can be integrated into the previous C programs above. Either way is acceptable.

Design and implement C routines which allow the user to create custom LCD characters using CGRAM. Explore other features of the HD44780 LCD controller.

- Implement the following function:

  ```c
  void lcdcreatechar(unsigned char ccode, unsigned char row_vals[])
  ```

- Write a program to provide a way for the user to interactively input (in hex) a custom character from the PC terminal emulator and to display the character on the LCD. Provide a way for custom characters to be stored in EEPROM and to be displayed after a power cycle of your board.

- In the same program, continuously animate a spinner arrow symbol on the LCD using four characters: '↑', '→', '↓', '←'. By sequentially displaying each of these characters in the same position on the screen, you can make these look like a spinning arrow. You will have to create some of the characters, since they are not all built into the LCD. You may want to use a timer interrupt to control the spin rate. Also note that MICRO-C provides a `delay()` function.

- In the same program, implement other features of the HD44780 LCD controller, including display shifting and cursor shifting and blinking. You may choose to explore the controller’s 5x10 pixel character capability. You may also choose to implement some other special effect, such as having each character of the string to be displayed ‘slide’ in from one side of the screen.

11. Demonstrate your hardware/software and get your lab signoff sheet signed by the TA or instructor.

---

\(^1\) Required elements are necessary in order to meet the requirements for the lab. Supplemental elements of the lab assignment may be completed by the student to qualify for a higher grade, but they do not have to be completed to successfully meet the requirements for the lab. The highest possible grade an ECEN 5613 student can earn on this assignment without completing any of the supplemental elements is an '86' (out of 100). The highest possible grade an ECEN 4613 student can earn on this assignment without completing either of the supplemental elements is a '93' (out of 100). ECEN 4613 students can earn full credit for this lab assignment by completing the required elements and only one supplemental element (either one, student's choice).

© 2003 Linden H. McClure, Ph.D. – 3 – University of Colorado
You will need to obtain the signature of your instructor or TA on the following items in order to receive credit for your lab assignment. This assignment is due by Saturday, November 15, 2003. Labs completed after the due date will receive grade reductions.

Print your name below, sign the honor code pledge, and then demonstrate your working hardware & firmware in order to obtain the necessary signatures. All items must be completed to get a signature, but partial credit is given for incomplete labs. Separate this sheet from the rest of the lab and turn in this signed form, a full copy of your updated schematic, and a printout of your fully and neatly commented source code (not .LST listing files) in order to receive credit for your work.

Make sure your name is on each item and staple the items together, with this signoff sheet as the top item.

Student Name: ______________________________________  4613 or 5613

Honor Code Pledge: "On my honor, as a University of Colorado student, I have neither given nor received unauthorized assistance on this work."

Student Signature: ____________________________________________

Checklist

Required Elements

☐ Schematic of acceptable quality (all components shown):
☐ Pins and signals labeled and decoupling capacitors present on board:
☐ All code clearly commented:
☐ Serial EEPROM functional, contents present after power cycle:
☐ C code for EEPROM hex byte reads and writes functional:
☐ LCD functional:
☐ C code for basic LCD routines functional:
☐ EEPROM user string storage and display:

Instructor/TA signature and date

Supplemental Element (Qualifies students for higher grade.)

☐ Elapsed time display (accurate 1 second resolution):
☐ Good integration with previous code, all functions work with no irregularities:
☐ Elapsed time stop, restart, reset to "00:00", up/down:

Instructor/TA Comments (e.g. user interface quality/issues):