

CSC 230 Assignment 3

Fall 2015

Submissions due on November 5, 2015 and November 16, 2015.

Demonstrations the week of November 17-20, 2015.

This assignment has three parts:

- 1) **Written Part:** A small written part, will be submitted by uploading a pdf document to the Assignments link of the Connex site. (No use of Tests&Quizzes this time!) **Submit it on connex before 9am on November 5, 2015.**
 - 2) **Programming Part:** Will be submitted via the Assignments link on the Connex site. **Submit it on connex before 9am on November 16, 2015.**
 - 3) **Demonstration:** You will be required to demonstrate the functioning of the programming part of your assignment to a member of the teaching team. **Demo during the week of November 17-20, 2015.**
-

Objectives

- Gain confidence with assembly language programming.
- Create effective subroutines/functions.
- Use peripherals: the LCD
- Become familiar with the use of busy-waiting

Academic Integrity

Prior to submitting your assignment, carefully read the University policy on Academic Integrity:

<http://web.uvic.ca/calendar2014/FACS/UnIn/UARe/PoAcl.html>

If you do not understand the meaning of anything in the document, please ask. A plagiarism detection tool is used on assignment submissions.

Assignment 3: Written Part [10]

1. [1] If an AVR processor is running with a clock speed of 16MHz, what is the maximum number of instructions it can execute in a single second?
2. [4] Design a loop using a conditional branch instruction that will repeat 255 times. Include two nop instructions in the body of the loop. As the name indicates, a nop (no-operation) instruction doesn't perform any operations other than consuming a CPU cycle. Loops are a very simple way to produce delays (e.g., waiting for something to happen). Knowing the CPU frequency, number of clock cycles each instruction consumes and the number of instructions executed in a loop, one can precisely calculate the loop delay in seconds.
 - a. If the microcontroller is running at a clock speed of 1MHz, how long will the code take to execute 256 iterations? Provide your answer in milliseconds.

- b. How long would it take if CPU is running at 16 MHz? Provide your answer in milliseconds.
3. [5] Lookup the addresses of the I/O register in the ATMEGA2560 for the digital I/O port A. Write assembler directives that define the symbols PORTA, PINA, and DDRA. Next, write statements that configure this port so that even numbered pins are inputs and the odd numbered pins are outputs.

Assignment 3: Programming Part [50]

The LCD

The boards we are using have a Hitachi HD44780 compatible LCD display that has its own (simple) processor. To communicate with that processor a specific protocol is used that initializes and controls the LCD screen. Learn more about how the LCD controller works by looking at the data sheet: <https://www.sparkfun.com/datasheets/LCD/HD44780.pdf>

A previous CSC 230 student has written a library of subroutines for this assignment. The table below lists the subroutines, their parameters and what they do.

Subroutine	Parameters (passed on the stack)	Description
<code>lcd_init</code>	None	Initializes the LCD screen. This subroutine must be called before any other subroutine in the LCD library is used.
<code>lcd_gotoxy</code>	X – 1 byte Y – 1 byte	Move the LCD cursor to position (x,y). The first line on the LCD is line 0, the second line is line 1.
<code>lcd_puts</code>	Address of C-String in data memory – 2 bytes	Display the null terminated string at the current cursor position. This routine does no length checking. It is up to you to make sure the string isn't longer than the available space on the LCD.
<code>lcd_clr</code>	None	Clear the LCD screen.
<code>str_init</code>	Address of source string in Program Memory – 2 bytes Address of destination string in Data Memory – 2 bytes	Copy a C string from program memory into data memory.

There is an example program, that shows how to use the LCD functions in the file:

`lcd_example.asm`.

Review the files, searching for a way to set the number of rows on the LCD to 2 and the number of columns to 20.

LCD Moving Message Sign

LCD advertising signs often move written messages on a screen and flash. The movement and the flashing are very effective at attracting attention. The goal of this programming task will be to display alternately two written messages, gradually moving them down and across the screen, and to display and flash both messages.

In particular, the program will need to:

- Create two messages that will be displayed on the screen. For example, assume:
 msg1 = "Fiona H. Albioni"
 msg2 = "CSC 230 Student!"
- When the program starts, the LCD screen will contain: Fiona H. Albioni with the F in the first column of the first line of the screen. After approximately $\frac{1}{2}$ a second, the LCD screen will be updated and the first message replaced with: CSC 230 Student!
- After another $\frac{1}{2}$ a second, the LCD screen will be cleared then will be set to contain: Fiona H. Albioni with the F in the second column and on the second line of the screen. After approximately $\frac{1}{2}$ a second, the LCD screen will be updated and the previous message replaced (in the same location) with: CSC 230 Student!
- After the message has been displayed clear the screen, wait $\frac{1}{2}$ a second, then repeat all of the above steps 2 more times.
- Finally, clear the screen, then at the middle of the screen flash (turn on and off) 3 times a single character. For example, the character 'A'.

Implementation Help

The instructor's solution has the following in the data segment:

```
; sample strings
; These are in program memory
msg1_p: .db "Fiona H. Albioni", 0
msg2_p: .db "CSC 230 Student!", 0

.dseg
;

; The program copies the strings from program memory
; into data memory.
;
msg1: .byte 200
msg2: .byte 200
; These strings contain the 16 characters to be displayed on the LCD
; Each time through, the 16 characters are copied into these memory locations
line1: .byte 17
line2: .byte 17
```

With these data definitions, the main loop of the application looks like:

```

initialize the lcd
clear the lcd
copy the strings from program to data memory
do 3 times:
    set row counter to 0 and column counter to 0
    do 2 times:
        clear the lcd
        display line1
        delay 0.5 second
        display line2
        delay 0.5 second
    increment row and column counters
clear the lcd
flash a character at centre of screen

```

You should make extensive use of subroutines in your code. In fact, each line above is its own subroutine. Including the delay code (which you can borrow from your lab) the solution is ~300 lines of assembly language. Start now!

Extending the Moving Message Sign

Now it is time to add some features:

- a. Extend the application so that you can press the “UP” button to stop the display. If stopped, pressing the “DOWN” button will restart the sign.
- b. Add another improvement, here are some suggestions:
 1. Allow button presses to increase and decrease the sign change speed
 2. Allow the user to select different messages

In order to handle button presses gracefully you will have to do something other than busy loop waiting, either by checking the buttons in your delay subroutine or by using timer interrupts. Your instructor will be covering timer interrupts in the next week or so.

Grading note:

- a. If you submit a program that does not assemble you will receive 0 for that part of the assignment.
- b. If you do not complete a demonstration of your code during the posted demonstration times, you will not receive a grade for this assignment.
- c. The basic moving message sign (as described under the title LCD Moving Message Sign) is worth 90% of the grade for the programming part of the assignment, while the extension (as described under the title Extending the Moving Message Sign) is worth 10%.

Assignment 3: Demonstration Part

At the demo, you will explain how your code works and be asked to make a small change in it. You will show that you can use the AVR Studio 4 and upload code to the board.

1. Sign up for a demonstration time on the sign-up sheet that will be posted on the door to the lab. (Demonstrations are likely take place outside lecture and lab times.)
2. Be prepared to demonstration the functionality of your assignment, to explain how your code works and be asked to make a small change in it. You will need to show competence with AVR Studio 4 and uploading the code to the board.