Lab #4: Elementary Programming & Adding a Keypad

Purpose

The purpose of this lab is to give students more practice in writing and debugging elementary assembly routines. In addition to this, students will learn how to add a simple keypad to their 6812 boards and how to scan it properly to eliminate switch bounce.

Part I. Writing Assembly - Sort Routine

You will be given a vector containing 2’s complement 8 bit numbers. You task is to write a routine that copies these numbers over to a new vector (in RAM) in order of magnitude. The largest number should be written first in the new vector and the smallest number should be last.

1. You are not allowed to corrupt the original vector that you are sorting. You are also not allowed to use the 68HC12 Max & Min functions. Write your own using conditional branches.

2. You should test your routine with a test vector of data (created by you).

3. In lab your TA will give you the address of the vector to be sorted (orig_addr), the length of this vector (orig_len) and the new sorted vector’s address (sorted_addr).

4. All code should be easily relocate-able in memory. You should write your sort routine as a subroutine and call it from your main program.

5. Pass parameters to your subroutine via the stack. Parameters should be placed on the stack (in the main program) in the following order:

   1. orig_addr
   2. orig_len
   3. sorted_addr

6. Your sort subroutine does not pass any values back to the main but instead creates the sorted vector in memory specified by “sorted_addr”.

7. Your code should be original and not match your classmates code (a data mining software will be used to examine all submitted code and check identical code). Copied Code will result in a zero for the lab.

Part II. Adding a Keypad

1. Connect the keypad given in your lab kit such that COL 4:1 are directly connected to Port P3:0. Note: It is best to place the keypad on top of the development board with the pins facing up nearest the port headers. See the diagram at the end of this lab documentation for more detail.

2. Connect the keypad ROW 4:1 signals to pull-up resistors and Port P7:4.

3. Using the Monitor, set DDRP such that Port P7:4 are inputs and Port P3:0 are outputs. We will use the lower nibble on Port P to send out a known pattern and then we will check for this pattern on the upper nibble (input) of Port P.

4. Try writing out a "E" (hex) on Port P and read the value in with no key pressed (should be "F").

5. Now press the "1" key and read the port. An "E" should be read back. Next, press "4" on the keypad and read the port. A "D" should be read back. The "E" output pattern places a "L" only on COL1 and so when we read this low in on a particular row, the pattern read in indicates whether it is "*", "7", "4" or "1" on the keypad.

********** Very Important Note **********

Only press one keypad button at a time. If you press two keypad buttons, it is possible to short two outputs together which may damage Port P and possibly destroy the 6812 IC.

6. Write a program that cycles a pattern of $7, $B, $D and $E over and over. Essentially, we are sending out only one "L" on a pin and cycling it through COL 4:1. Next, read in the port value that corresponds to keypad ROW 4:1 and using both pieces of information determine which key has been pressed on the keypad. Send this value to your binary display.
Lab #4: Elementary Programming & Adding a Keypad

**Note #1:** "*" pressed on the keypad should correspond to $E$ on the display and "#" pressed on the keypad should correspond to $F$ on the display.

**Note #2:** The above algorithm used to determine which key has been pressed is denoted as scanning a keypad and is also used on a PC keyboard. Every PC keyboard has an inexpensive microcontroller that scans for a pressed key and then serially transmits the information back to the PC.

**Part I. Pre-Lab Requirements (15%)**

1. Create a flow chart for your main and subroutines.

2. Code all blocks in the flow chart and assemble & test via SIMHC12.

3. Bring all above-mentioned materials to lab as hard copies and on diskette.

**Part II. Pre-Lab Requirements (15%)**

1. Clear concise hand drawn schematics showing the circuitry described in Part II. Show Port P pin numbers as well as those on the keypad.

2. Assembly code (assembled bug free) for the test routine described in Part II item #6 on diskette and on printouts (source and listing).

**In-Lab Requirements (70%)**

1. **Your TA will not allow you in unless you have the pre-lab materials specified above.**

2. After showing your TA the pre-lab materials, your TA will give you the address of a vector of 2's complement numbers, the length of that vector and the destination vector address. Re-assemble your program with the new information and then test/demo your results to your TA. You may need to show your source code to TA and answer specific questions (30%).

3. Next show your wiring and component placement corresponding to the work required in Part II (10%).

4. Demonstrate that pressing a particular key on your keypad lights up the corresponding binary pattern specified earlier on your LED display. You may need to show your source code to TA and answer specific questions (30%).

**Suggested Final Component Layout**

The ports we are connecting to are in the far upper left side of the wire-wrap area. Thus all components should be kept in this proximity.

![Diagram of UF 6812 Development Board with Keypad and Port P connections]