CPTR-215
HW # 16 - Due Wednesday

The goals of this home work are to gain experience with:

- reading data from an I/O port
- writing data to an I/O port
- manipulating bits to perform a desired operation
- using subroutines

Write a program for the ARM processor on an Embedded Artists LPC2148 board that will repeatedly read the status of the joystick and display a binary number on the LEDs coded per the direction that the joystick is pressed as follows:

1 - left
2 - right
3 - up
4 - down
5 - center
6 - a combination of directions

If the joystick is not pressed then no LEDs will light.

The number corresponding to how the joystick is pressed is to be displayed in binary on the LEDs using bits P0.12, P0.11, P0.10 with P0.10 being the least significant digit of the binary number.

This program is to be written with a main part and a subroutine. The subroutine will read the joystick, determine if the joystick is being pressed, figure out the direction it is pressed, and return a value as defined above (1 thru 6). Also, if the joystick is pressed the subroutine will cause the zero flag to be set but if the joystick is not pressed the zero flag will be cleared. When writing the subroutine use standard practice, i.e. registers r0, r1, r2, and r3 do not have to be preserved during subroutine execution, i.e. upon return from a subroutine these 4 registers will be assumed changed. Registers r5 up to r12 will be assumed unchanged by a subroutine. Thus if a subroutine needs to use more than 4 registers it needs to save others on the stack and then restore them prior to return from the subroutine.

The main part of the program will have a loop that calls the subroutine to see if the joystick is pressed and, if it is, will display the direction code on the LEDs as defined above. .

Before writing assembly code design the program. Create NS diagrams or flow charts that show program operation. Then code and test. Testing can be done in simulation using the debugger but you need to also test the program on the hardware board (where you also can use the debugger). Note that the subroutine will be written in the same assembly file (.s file) as the main part of the program.

Turn in: hard copy of your design documentation and a program listing. On the program listing

write a note describing the success (or failure) of your program and any particular issues you had in creating it. E-mail me a copy of your source file.

I/O Documentation

Details of how the switches are connected I/O port zero (P0) are documented in the users manual for the Embedded-Artists board that we are using. The choice of which port and which bits are used was made by the designer of the board and not the designer of the CPU chip. On the class website you will find the complete users guide for the board. Below is an excerpt that shows the details of the joystick:

2.1.8 Page 3: Joystick-switch

There is a joystick-switch on the board and *Figure 14* below illustrates this part of the design. The switch has five internal switches, one for the four directions and one center, push-down switch. All of the input pins (P0.16 – P0.20) can be programmed as interrupt inputs, either directly as an EINTx-pin or via a CAPture-pin, which in turn can generate an interrupt.

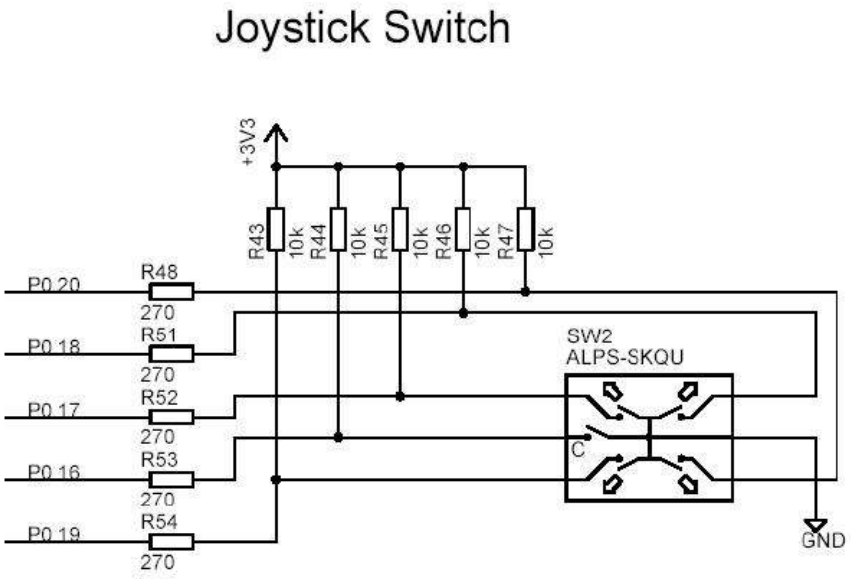

Figure 14 - LPC2148 Education Board Schematic, page 3: Joystick-switch

We are not using interrupts so ignore the comments about that. (The page 3 referred to in the figure is the schematic page that this part of the circuit comes from).

In summary, when a switch is not pressed a logic 1 will be read from a port pin. When a switch is pressed, a logic 0 will be read. If the center switch is activated, none of the other four switches will be. If the joystick is pushed straight up or down or left or right then one switch will be activated (closed). If the joystick is pushed diagonally two switches may be activated and thus two bits of the five may be logic 0.