CECS 285: Lab 5

OBJECTIVES:

- Gain familiarity with subroutine usage
- Use loops to generate time delays

ACTIVITY 1

Write a program that reads switch positions and uses the switch positions to set a register value which controls the number of times a nested loop iterates between toggling LEDs. Your program will have the following 3 sections:

Section 1: Main Program Loop

```
        jmp main        ;jump past interrupt vector

org 0x30
main:   ;--+ configuration section +--
        mov p1, #0x00   ;set p1 as output

        ;--+ main program section +--
main_loop:
        call    checkInput
        call    Delay
        mov     p1, #0xFF   ;LEDs ON
        call    checkInput
        call    Delay
        mov     p1, #0x00   ;LEDs OFF
        jmp     main_loop
```

Our programs will have this kind of structure from now on. The first instruction is a jump to the main program section which bypasses the ROM locations reserved for the interrupt vector table. Therefore our program instructions will be placed in rom beginning at HEX address 30. The main loop repeatedly goes through the following operations:

1. Read switches and set a r7 to a value that controls the length of delay that is generated
2. Create some delay using nested loops
3. Turn on LEDs
4. Read switches and set a r7 to a value that controls the length of delay that is generated
5. Create some delay using nested loops
6. Turn off LEDs

As it can be seen this program makes use of subroutines to read/process switch positions and create time delays. The source code for subroutines is listed on the following pages.

Section 2: Switch position read and processing subroutine:

```
        ;--+ input check subroutine +--
checkInput:
        mov     r7, p0      ;read switch positions into r7
        cjne    r7, #0x00, not0
        mov     r5, #1
        jmp     goback
not0:   cjne    r7, #0x01, not1
        mov     r5, #2
        jmp     goback
not1:   mov     r5, #3
goback: ret     ;return from subroutine
```

As it can be seen in the source code above, the switch positions in some combination of up and down will be converted to logic levels and will be represented in some 8-bit hex value. The hex value is moved into r7 where it is compared against other values using if/else style logic. Then a value is loaded into the r5 register and this value in the r5 register will be used to control the length of delay in the delay subroutine.

Section 3: Delay Generation Subroutine

```
        ;--+ delay subroutine +--
Delay:
again:  mov r0, #250
outer:  mov r1, #250
inner:  djnz    r1, inner
        djnz    r0, outer
        djnz    r5, again
        ret     ;return from subroutine
```

The delay subroutine listed above was tested on the lab pro 51 (classic 8051 architecture) generates about a (1/8 of a second second) 13mS time delay when r5 equals 1. If r5 equals 2 then the delay time is doubled to ¼ of a second, if r5 equals 3 then the time delay is 3/8 of a second and so forth.

The 8051 Trainer will need more loop nestings or larger r5 values to create an equivalent time delay, do calculations to determine the proper register values and for the nested loops. Delay times do not need to be precise, approximate time delays are acceptable within reasonable time variations.

Your task is to modify the program to use two switch positions to control the time delay generated between LED blinks according to the following table of operations:

switches	**time delay**
down, down	a quarter second
down, up	a half second
up, down	1 second
up, up	2 seconds

Test your program on hardware and when you are ready demonstrate proper functionality of your project for the instructor.

Deliverables:

- Your commented assembly language program source code