Lab 12 JHS & JLO
Name : Van Nguyen

CCSU - School of Engineering & Technology
CET 453 Microcomputers

Jump Instructions –
Higher or Same vs. Lower

Lab Goal

Understand the effect and use of jump instructions.

Instructions/Questions:

1) Open Workspace. Edit the asm.s43 file for this project.

Download file JHS JLO.zip From Lab 11 in Blackboard ,
Open file JHS JLO.eww
Edit file asm.s43

2) Study the code and explain the effect(s) of lines 18 and 19. Namely, what will happen to the carry bit and the Program Counter (PC) when R14 is 0 to 4 inclusive and when R14 is 5 to 65535 inclusive? (R14 will be incremented every time that an input goes low. This will happen automatically by the interrupt for Port 1.)

```
Line 18 :   CMP.W  #005, R14         ; Compare the number of times the input when low with 5 dec
Line 19 :   JHS    turnoffLED        ; If this happened 5 or more times, just turn off the led .
```

Explain :

Line 18 :

Using CMP instruction compare source and destination

CMP.W src, dst , dst + ! src +1 (binary)

OR (dst – src) (Hexa)

We have CMP.W #005, R14 R4 = 0b 0000 0000

# 005 = 0b 0000 0101 (value of Immediate Mode)

Compare dst and src, we see dst Lower source (dst < src)

⇨ **If src is greater than dst,** N = 1, **Z = 0, C = 0 ; dst < src**

Line 19 :

Using Instruction JHS Jump if higher or same (status bits not affected)

JHS label If C = 1 : Label => PC (Jump)

If C = 0 : Execute following instruction (not jump)

We have : JHS turnoffLED

Because dst < src , so not higher or same (C = 0) => Not Jump

Namely :

☆ When R14 from 0 to 4 then the value dst still lower source (# 5),
So dst < src and N = 1 => C = 0 . (Not Jump)

☆ When R14 from 5 to 65535 then the value dst >= src (C = 1) ,
So, Instruction JHS turnoffLED will execute Jump to " turnoffLED ". (Line 23)

3) Study the code and explain the effect(s) of lines 21 and 22. Namely, what will happen to the carry bit and the Program Counter (PC) when P1IN bit 3 is reset (0) and when P1IN bit 3 is set (1)?

```
Line 21    BIT.B   # 008h, &P1IN        ; test only p1.3 (bit 3) by AND'ing with 0b00001000
Line 22     JC   turnonLED              ; if C was set, bit 3 was set (p1.3 was high), if not set p1.3 was low
```

Line 21 :

Using instruction: BIT.B test bits in destination

BIT.B src, dst src AND dst

The result affected to **status bits**, **src** and **dst** are not affected .

```
We have:  BIT.B   # 008h, &P1IN     P1IN   =  0b 0000 1000        ( test only p1.3 - bit 3)
                                                           AND
                                   # 008h  =  0b 0000 1000        ( value of Immediate Mode)
                                  ________________________
                                    P1IN   =  0b 0000 1000         bit 3 is set  = 1  (high)
                                             (N = 0 => C = 1)
```

☆ Because P1IN (dst) bit 3 was set high, and Negative flag = 0 => C = 1

Line 22 :

Using Instruction JC Jump if carry was set (C = 1)

JC label If C = 1 : Label => PC (Jump)

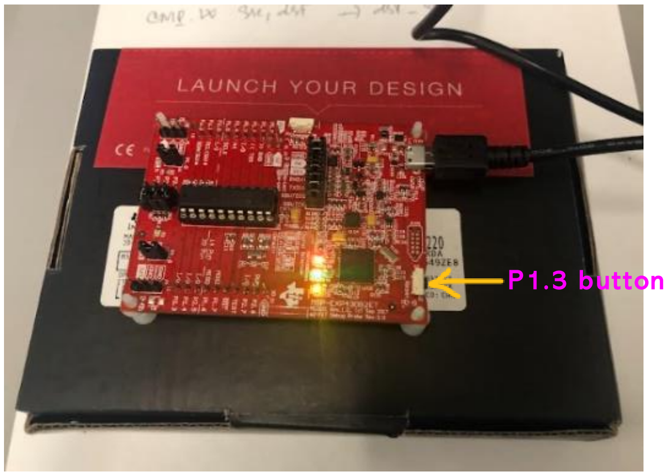
If C = 0 : Execute following instruction (not jump)

We have : JC turnonLED

Because Carry flag was set (C = 1) => Jump to turnonLED (line 26)

4) Connect the development tool to the PC.

TI MSP430 Launch Pad Value Line Development Board connect to PC



5) Click "Download and Debug."
View the CPU registers.
From the menu select Debug -> Auto step. Enter a value of 500 for the delay and leave the level at Step Into.
Click Start.

- When selecting Debug --> Auto step => open a small box named delay, enter the value 500 press the Start button.
- The program will automatically run according to the code line.

6) Observe the flow of the code when you haven't pressed the P1.3 button. What happens to the program flow and register R14 and the carry flag?

- The program flow will automatically run in a continuous code flow without stopping.
- The value of register R14 remains unchanged. (0000h)
- The carry flag continuously changes from 0 (C = 0) to 1 (C =1).

7) Press and hold the P1.3 button. What happens to the program flow and register R14 and the carry flag? (Note R14)

When we press and hold the P1.3 button then :

- The program flow still run continuously on the flow of the code.
- The value of register 14 is increased more by 1 after one loop of the program flow.
- The carry flag still continues to change from 0 to 1 until R14 = # 04h

9) Let P1.3 button up and press it again. R14 should increment with each connection to ground (as long as the auto step function of the debugger can keep up). What happens and why after 5 occurrences of pressing P1.3?

Each time P1.3 button is pressed, register R14 will increase by 1, so after 5 times of pressing P1.3, the value of R14 will be 05 and the carry flag will always be one (C = 1) and there will be no 0.

Explain :

- Because when R14 = 05 and increments, the instruction of line 18 is:
- CMP.W #005, R14 R14 = 0b 0000 0101

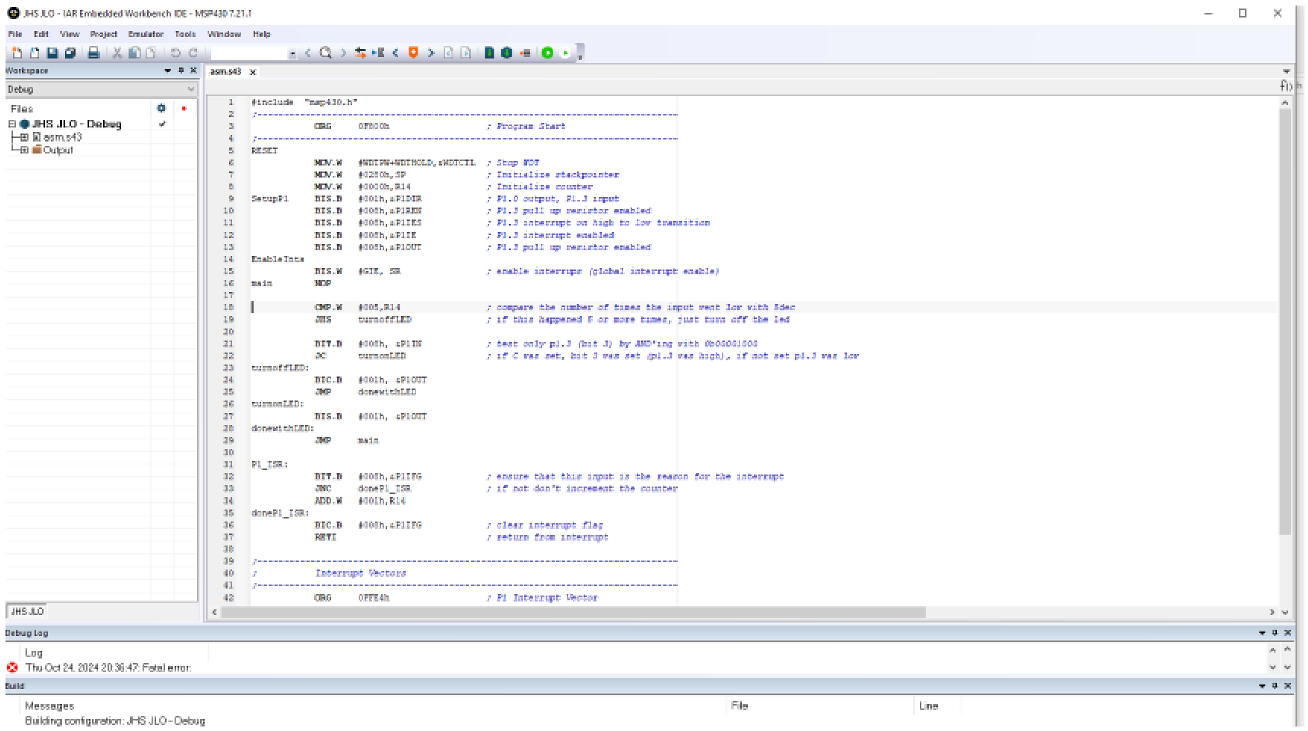
  # 005 = 0b 0000 0101 (value of Immediate Mode)

If dst is equal to src,	N = 0, Z = 1, C = 1	; dst = src
If dst is most than src,	N = 0, Z = 0, C = 1	; dst > src

Compare dst and src, in the two cases above, we see that dst is Higher or same source (dst >= src)

⇨ The carry flag always is one **C = 1** .

The program flow and the flow of the code in file asm.s43 of project



```
#include  "msp430.h"
;-------------------------------------------------------------------------------
            ORG     0F800h                  ; Program Start
;-------------------------------------------------------------------------------
RESET
            MOV.W   #WDTPW+WDTHOLD,&WDTCTL  ; Stop WDT
            MOV.W   #0280h,SP               ; Initialize stackpointer
            MOV.W   #0000h,R14              ; Initialize counter
SetupP1     BIS.B   #001h,&P1DIR            ; P1.0 output, P1.3 input
            BIS.B   #008h,&P1REN            ; P1.3 pull up resistor enabled
            BIS.B   #008h,&P1IES            ; P1.3 interrupt on high to low transition
            BIS.B   #008h,&P1IE             ; P1.3 interrupt enabled
            BIS.B   #008h,&P1OUT            ; P1.3 pull up resistor enabled
EnableInts
            BIS.W   #GIE, SR                ; enable interrups (global interrupt enable)
main        NOP

            CMP.W   #005,R14                ; compare the number of times the input went low with 5dec
            JHS     turnoffLED              ; if this happened 5 or more times, just turn off the led

            BIT.B   #008h, &P1IN            ; test only p1.3 (bit 3) by AND'ing with 0b00001000
            JC      turnonLED               ; if C was set, bit 3 was set (p1.3 was high), if not set p1.3 was low
turnoffLED:
            BIC.B   #001h, &P1OUT
            JMP     donewithLED
turnonLED:
            BIS.B   #001h, &P1OUT
donewithLED:
            JMP     main

P1_ISR:
            BIT.B   #008h,&P1IFG            ; ensure that this input is the reason for the interrupt
            JNC     doneP1_ISR              ; if not don't increment the counter
            ADD.W   #001h,R14
doneP1_ISR:
            BIC.B   #008h,&P1IFG            ; clear interrupt flag
            RETI                            ; return from interrupt

;-------------------------------------------------------------------------------
;           Interrupt Vectors
;-------------------------------------------------------------------------------
            ORG     0FFE4h                  ; P1 Interrupt Vector
```

```
Log
Thu Oct 24, 2024 20:36:47: Fatal error:
```

```
Messages                                                    File            Line
Building configuration: JHS JLO - Debug
```