Name : Van Nguyen

CCSU - School of Engineering & Technology
CET 453 : Microcomputers

Jump Instructions - equal, not equal

Lab Goal :

Understand the effect and use of jump instructions.

Instructions/Questions:

1) Use your notes to study the following "program". Based on the values given, explain what will happen. Show Work by taking a photo of your work.

Changes to registers should be detailed as well as what line number would be executed if the normal progression is changed.

If a line will not be executed during a loop indicate this with "–". Indicate status register flags Z = x,… if there is a change.

Line	Instruction	Effect during loop 1	Effect during loop 2
1	Main:	n/a	n/a
2	CMP.B #0Ah, R4	R4 = src = 0x0A (N = 0, Z = 1, C = 1)	R4 = 0x06 < src (N = 1, Z = 0 , C = 0)
3	JNE compare8	Equal , (src = dst) =>not Jump	Not equal (src != dst) => jump compare8 (*line 6)*
4	MOV.B #06h, R4	R4 = 0x06	-
5	JMP CompareDone	Jump CompareDone (*line 15*)	-
6	Compare8:		
7	CMP.B #08h, R4	-	R4 = 0x06 < src (N = 1, Z = 0 , C = 0)
8	JNZ Compare6	-	Not equal (src != dst) => jump compare6 (*line 11)*
9	MOV.B #06h, R4	-	-
10	JMP CompareDone	-	-
11	Compare6:		-
12	CMP.B #06h, R4	-	R4 = src = 0x0A (N = 0, Z = 1, C = 1)
13	JNE CompareDone	-	Equal , (src = dst) => not Jump
14	MOV.B #07h, R4	-	R4 = 0x07
15	Comparedone:		
16	NOP	No Operation	No Operation
17	JMP Main	Jump Main (*line 1 – loop 2*)	-

2) Use your notes to create a program which will continuously read port2 inputs to see when both P2.0 and P2.1 are high. When both are high the LED should be turned on by setting P1.0 high, in every other case P1.0 should be low.

Execute the code in Embedded Workbench and simulate the 4 possible combinations of P2.0, P2.1 using the Registers Window – Port 1/2 – P2IN

Paste the main loop of the code below and submit only this word document.

P2IN								Case
P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	
x	x	x	x	x	x	1	1	Turn LED On
x	x	x	x	x	x	0	0	Turn LED Off
x	x	x	x	x	x	0	1	Turn LED Off
x	x	x	x	x	x	1	0	Turn LED Off

Useful instructions:

Turn on P1.0 (LED)	BIS.B #001h,&P1OUT
Turn off P1.0 (LED)	BIC.B #001h,&P1OUT
Copy Port2 input status into R4	MOV.B &P2IN, R4

CODE :

```
Main:
      MOV.B &P2IN, R4          ; Read port 2 input into R4
      CMP.B #03h, R4           ; Check if P2.0 and P2.1 are high
      JNE LowLED_Off           ; Jump if not Equal ( both not high )

HighLED_On:
      BIS.B #01h, &P1OUT       ;  Set P1.0  high and turn on LED
      JMP Main                 ;  Go back Loop check again

LowLED_Off:
      BIC.B #01h, &P1OUT       ;  P1.0 to Low and turn Off LED
      JMP Main                 ;  Go back Loop check again
```

Explain :

1. Main : (*MainLoop*)

2. Used Instruction **CMP.B** (Compare: source & destination)
   - Operation : dst – src
     Status bits :
     - If src is equal to dst, N = 0, Z = 1, C = 1 ; dst = src
     - If src is less than dst, N = 0, Z = 0, C = 1 ; dst > src
     - If src is greater than dst, N = 1, Z = 0, C = 0 ; dst < src

   - We have : CMP.B #0Ah , R4
     Compare :
     - Source : 0x0A = $(0000\ 1010)_2$
     - R4 (dst) : 0x0A = $(0000\ 1010)_2$
       0x00 **$(0000\ 0000)_2$**

   ★ **Src is equal to dst,** N = 0**, Z = 1, C = 1 ; dst = src**

3. Used Instruction **JNE Label** (Jump if not equal , src != dst)
   - Operation : If src != dst (Z=0) Value → PC
     If src = dst (Z=1) jump to Label
     Status bits : *not affected*

   - We have : JNE compare8

   ★ Because in line 2 **(dst = src)** so **Not Jump** and *continuous next line* (line 4)

4. Used Instruction **MOV.B** (move the contents of source to destination)
   - Operation : src → dst
     Status bits : *not affected*

   - We have : MOV.B #06, R4
     - Source : 0x06 = $(0000\ 0110)_2$
     - Destination : R4

   ★ **Move src (0x06) to dst (R4) ,** Now **R4 = 0x06**

5. Used Instruction **JUMP label** (*unconditionally*)
   - Operation : Jump → label value (PC)
   - Status bits : *not affected*

   - We have : JUMP CompareDone

★ **Jump to label CompareDone** (line 15) skip from line 6-14

16. We have : NOP (*Not Operation*)

★ **Don't do anything** continue next line (line 17)

17. Used Instruction **JUMP label** (*unconditionally*)
   - Operation : Jump → label value (PC)
   - Status bits : *not affected*

   - We have : JUMP main

★ **Jump to label main** (line 1_ Loop 2)

Loop 2 :

2. Used Instruction **CMP.B** (Compare: source & destination)
   - Operation : dst – src
     Status bits :
     - If src is equal to dst, N = 0, Z = 1, C = 1 ; dst = src
     - If src is less than dst, N = 0, Z = 0, C = 1 ; dst > src
     - If src is greater than dst, N = 1, Z = 0, C = 0 ; dst < src

   - We have : CMP.B #0Ah , R4
     Compare :
     - Source : 0x0A = $(0000\ 1010)_2$
     - R4 (dst) : 0x06 = $(0000\ 0110)_2$

★ **Src** *is greater than* **dst, N = 1, Z = 0, C = 0 ; dst < src**

3. Used Instruction **JNE Label** (Jump if not equal , src != dst)

- Operation : If src != dst (Z=0) Value → PC
  If src = dst (Z=1) jump to Label
  Status bits : *not affected*

- We have : JNE compare8

★ Because in line 2 **(src != dst)** so **Jump** to Compare8 (line 6) *skip line 4 and 5*

7. Used Instruction **CMP.B** (Compare: source & destination)

- Operation : dst – src
  Status bits :
  - If src is equal to dst, N = 0, Z = 1, C = 1 ; dst = src
  - If src is less than dst, N = 0, Z = 0, C = 1 ; dst > src
  - If src is greater than dst, N = 1, Z = 0, C = 0 ; dst < src

- We have : CMP.B #08h , R4
  Compare :
  - Source : 0x08 = $(0000\ 1000)_2$
  - R4 (dst) : 0x06 = $(0000\ 0110)_2$

★ **Src** *is greater than* **dst, N = 1, Z = 0, C = 0 ; dst < src**

8. Used Instruction **JNE Label** (Jump if not equal , src != dst)

- Operation : If src != dst (Z=0) Value → PC
  If src = dst (Z=1) jump to Label
  Status bits : *not affected*

- We have : JNE compare6

★ Because in line 7 **(src != dst)** so **Jump** to Compare6 (line 11) *skip line 9 and 10*

12. Used Instruction **CMP.B** (Compare: source & destination)

- Operation : dst – src
  Status bits :
  - If src is equal to dst, N = 0, Z = 1, C = 1 ; dst = src
  - If src is less than dst, N = 0, Z = 0, C = 1 ; dst > src
  - If src is greater than dst, N = 1, Z = 0, C = 0 ; dst < src

- We have : CMP.B #06h , R4
  Compare :
  - Source : 0x06 = $(0000\ 0110)_2$
  - R4 (dst) : 0x06 = $(0000\ 0110)_2$
    0x00 **$(0000\ 0000)_2$**

★ **Src** is equal to **dst,** N = 0**, Z = 1, C = 1 ; dst = src**

13. Used Instruction **JNE Label** (Jump if not equal , src != dst)

- Operation : If src != dst (Z=0) Value → PC
  If src = dst (Z=1) jump to Label
  Status bits : *not affected*

- We have : JNE compareDone

★ Because in line 12 **(dst = src)** so **Not Jump** and *continuous next line* (line 14)

14. Used Instruction **MOV.B** (move the contents of source to destination)

- Operation : src → dst
  Status bits : *not affected*

- We have : MOV.B #07, R4
  - Source : 0x07 = $(0000\ 0111)_2$
  - Destination : R4

★ **Move src (**0x07**) to dst (**R4**) ,** Now **R4 = 0x07**

The end Loop !