Example: Branches and Conditional statements

Program stored in memory – current address 3000.

`if R1 > R2 then branch to address x3007`

Address **memory contents**

X3000: 1001 011 010 111111 R3= NOT(R2) 1001 is NOT opcode

X3001: 0001 011 011 1 00001 R3= R3 +1 *2's complement, so R3= -R2*

using ADD immediate value (rightmost 5 bits in 2C)

X3002: 0001 100 011 000 001 R4= R1 +R3 *(R4= R1-R2)*

If R1 > R2 then R4 >0. Therefore CC register P=1 is set by output of ALU

X3003: 0000 001 000000011 BR to PC+3 if R1>R2

branch

PC already incremented to 3004 after instruction fetched

NZP

Offset to target

69

Programming the LC3 – machine language

Example: we want to program this C code on LC3

```
if (x+3) < 5 y=x+3;
          else y = 5;
```

Assume:

1. Program starts at x3000
2. x is stored at x3010 and y at x3012

Solution:

1. Load x into register R1 – *use LD instruction*
2. Add 3 to register R1 - *use Add immediate mode*
3. Subtract 5 from R1 - *use Add immediate value -5 in 2C*
4. if (R1-5)< 0 then store x+3 into x3012 – *branch if N=1*
5. Else Store 5 in x3012

Note hex representation: x12 = #18 (decimal)

70

The C code implemented on LC3

```
X3000: 0010 001 000001111          Load (LD) from x3010 to R1
                                          15 places from x3001
X3001: 0001 001 001 1 00011        Add (immediate) 3 to R1 to get R1=x+3
X3002: 0001 010 001 1 11011        Subtract 5 from R1:  R2 = (x+3)-5
                          note in 2C: 11011 = -5    if R2 <0 then CC N=1
X3003: 0000 100 000000100          If N=1 goto  x3008 to do y=x+3 offset=4
X3004: 0101 011 011 1 00000        Else  -  Set R3=0; 0 AND anything=0
X3005: 0001 011 011 1 00101        Add 5 to R3 to get R3=5
X3006: 0011 011 000001011          Store R3 to x3012 (address of y) offset=11
X3007: 0000 111 ?????????          goto statement after IF statement
X3008: 0011 001 000000011          Store (x+3) to x3012 (address of y) offset=3
X3009: ???? ???? ???? ????         instruction after IF statement
```

71

71

In-Class Exercise: Tracing program execution (machine code)

Consider the following LC3 program stored in memory. The leftmost column is the address, and the other columns specify the 16 bit value stored at that address; for example, address x3005 stores the 16-bit value 0101 010 010 1 00010 (the figure shows the value for each of the bits from bit 15 to bit 0) – this is an encoding of the instruction R2= R2 AND #2 (00…0010 in binary). Assume that at the start of the program R4 contains #6 and R2 contains #0.
What do the instructions at addresses x3000, x3001 and x3002 do ? What is the outcome of executing these instructions – i.e., what happens during execution of this program?

Address	**15**	**14**	**13**	**12**	11	10	9	8	7	6	5	4	3	2	1	0
x3000	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	1
x3001	0	0	0	1	0	1	1	0	1	0	1	1	1	0	0	0
x3002	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0
x3003	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1
x3004	0	0	0	0	1	1	1	1	1	1	1	1	1	0	1	1
x3005	0	1	0	1	0	1	0	0	1	0	1	0	0	0	1	0
x3006	0	0	0	0	0	1	0	1	1	1	1	1	1	1	0	1
X3007	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Solutions:
Program has an infinite loop.

1. Address x3000: LD instruction, with offset 3, and PC would have been incremented by 1 after fetching instruction at x3000 – therefore PC is now x3001. So instruction is: Load from PC+3 into Register R2 = load from memory address x3004. R2= 0x0FFB (4091).
2. Address x3001: ADD (immediate) instruction since bit5=1. Therefore Add R2 and immediate value 11000 = -8 and store into R3 ; therefore R3= 4091-8.
3. Address x3002: Branch instruction if result Neg or Zero. Result is positive therefore branch not taken and next instruction at x3003 is executed.
4. Address x3003: ST instruction with offset 3 and PC is now x3004. Store contents of R4 to address PC+3 = x3007. Stores value #6 to address x3007
5. Address x3004: Branch on N or Z or P – branch always taken to PC+ (11..11011) = PC -5 = x3000…loop back to start (program is in an infinite loop)

For reference, suppose contents at memory address x3004 were x0005 (#5). Then:

1. Address x3000: LD instruction, with offset 3, and PC would have been incremented by 1 after fetching instruction at x3000 – therefore PC is now x3001. So instruction is: Load from PC+3 into Register R2 = load from memory address x3004. Therefore R2= 5 after instruction executed.
2. Address x3001: Add R2 and immediate 11000 = -8 and store into R3 ; therefore R3= 5-8 = -3 (111101 binary)
3. Address x3002: Branch to PC+2 if result Neg or Zero= branch taken to PC+2 (since R3= -3) so next instruction to be fetched and executed is at x3005
4. Address x3005: AND (immediate) R2 = R2 AND 000010 (#2) = 00…0101 AND 00…0010 = 0000 = #0

5 Address x3006: Branch on Zero to PC-7 = branch taken (since R2=0) to PC-7 = x3000 (program loops again – infinite loop)

Assembly language equivalent

```
Start   LD R2, #3
        ADD R3, R2, # -8
        BRnz Goto
        ST R4, #3
        BRnzp Start
Goto    AND R2, R2, #2
        BRz # Start
```