LC-3 INSTRUCTION SET – PART II

Control instructions

They change the sequence of execution of instructions by manipulating the value of the PC register.

* JMP (unconditional jump): load PC with content of register

\[ PC \leftarrow \text{BaseR} \]

Example: \[ 1100 \ 000 \ 011 \ 000000 \]

Opcodes (JMP)
BR (conditional branch): if any of the relevant condition codes occur then increment PC with $PC_{offset9}$

Notation:
- $n, z, p$: bits in instruction (lower case)
- $N, Z, P$: bits in condition codes (uppercase)

If $(n \text{ AND } N) \text{ OR } (z \text{ AND } Z) \text{ OR } (p \text{ AND } P)$
then $PC \leftarrow PC + \text{SEXT}(PC_{offset9})$
Example:

```
0000 011011001
```

Oprade \( n \) \( z \) \( p \) \( \text{PC offset+9} \)

(BR)

Question: what happens if in the instruction \( n = z = p = 1 \)?
* TRAP: sequence of instruction execution goes to 
Operative System (OS) service call

\[ PC \leftarrow M[\text{TEXT(trapvec} + 8)] \]

There are 3 trap vectors:
x21: outputs character to the monitor
x23: input character from keyboard
x25: halts program

Example: 1111 0000 0010 0101

Opcode
(TRAP)
Introduction to programming in machine code

* Notation:

    1111 0000 0010 0101 ; Comment sentence

Machine code, use spaces for clarity

Comments are preceded by ;
* Starting/stopping program:
First line of program is initial address, not an instruction.
In LC-3, programs must be stored after x2FFF address
0011 0000 0000 0000 ; Load program starting in x3000
Last line of program must stop computer
1111 0000 0010 0101 ; HALT instruction (TRAP x25)
* Working with registers:

1) Clear a register

Example: \( R6 \leftarrow 0 \)
2) Copy register into another

Example: R6 ← R1
3) Increment register by one
   Example: \( R6 \leftarrow R6 + 1 \)

4) Decrement register by one
   Example: \( R6 \leftarrow R6 - 1 \)
5) Subtract two registers

Example: R7 ← R5 - R6

RTL:

Machine language:
6) Bitwise OR

Example: \( R7 \leftarrow R5 \text{ or } R6 \)

Approach:

RTL: