THE UNIVERSITY OF KANSAS

Assembly Instructions

EECS388 Fall 2022

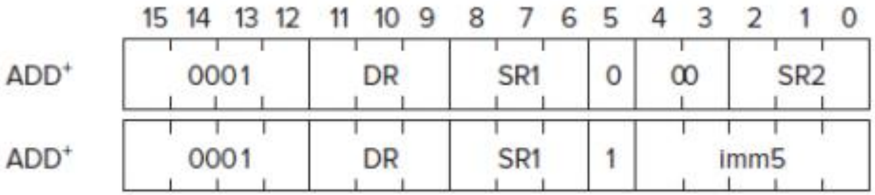
Context

- Recommended reading

Chapter 7 and Appendix A of “Introduction to Computing,” Patt, Patel

ADD

```
ADD DR, SR1, SR2
ADD DR, SR1, imm5
```



	15 14 13 12	11 10 9	8 7 6	5	4 3	2 1 0
ADD+	0001	DR	SR1	0	00	SR2
ADD+	0001	DR	SR1	1	imm5	

- Operation:

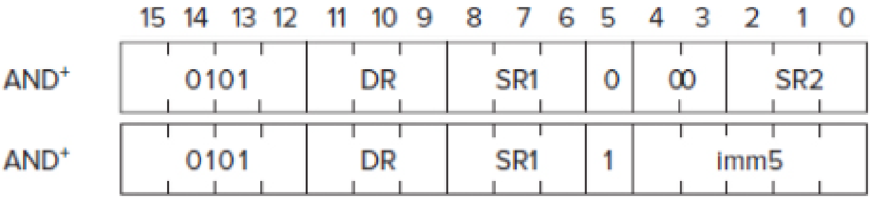
```
If (bit[5] == 0)
      DR = SR1 + SR2
Else
      DR = SR1 + SEXT(imm5)
setCC();
```

- Example:

```
ADD R1, R2, R3
ADD R2, R5, #7
```

AND

```
AND DR, SR1, SR2
AND DR, SR1, imm5
```



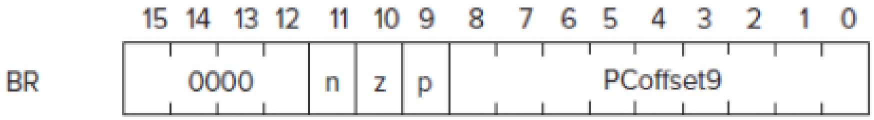
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AND^{+}	0101				DR			SR1			0	00		SR2		
AND^{+}	0101				DR			SR1			1	imm5				

- Operation:

```
If (bit[5] == 0)
     DR = SR1 & SR2
Else
     DR = SR1 & SEXT(imm5)
setCC();
```

- Example:

```
AND R1, R2, R3
AND R2, R5, #7
```



	15–12	11	10	9	8–0
BR	0000	n	z	p	PCoffset9

BR

```
BRn LABEL      BRz LABEL      BRp LABEL
BRnp LABEL     BRzp LABEL     BRnz LABEL
BRnzp LABEL
```

- Operation:

```
If ((n AND N)OR (p AND P) OR (z AND
Z))
    PC = PC + SEXT(PCoffset9)
```

- Example:

```
BRzp  LOOP
BRnzp TARGET
```

ORIG 0X0000

DS SR1 SR2

AND R1, R2, #0

0

CC

z n p

1 0 0

Taken

IR

BR nz LABEL

PC

ADD R1, R1, #1

LABEL ADD R1, R1, #2

2 0

0101 001 010 1 00000

0000 110

What is R1?

Mo -> 2

Manoj -> 1

Ezzy -> 2 X

Suhaan -> 3 ✓

Binary

PC = PC + SEXT(PCoff)

```
           .ORIG   0X3000
                         16 bits
0x0000 ->          AND     R1, R2, #0
0x0001 ->          BRnz    LABEL
0x0002 =>          ADD     R1, R1, #1
0x0003  LABEL      ADD     R1, R1, #2
        0x0003
```

```
short int VR1, VR2;
VR1 = VR2 & 0;
if ( VR1 > 0 )
        VR1 += 1; <=
VR1 += 2;
```

PC = 0X 0002

```
0101 001 010 1 00000     <= AND
0000 110 000000001
0001 001 001 1 00001
0001 001 001 1 00010
```

PC = PC + SEXT(offset)

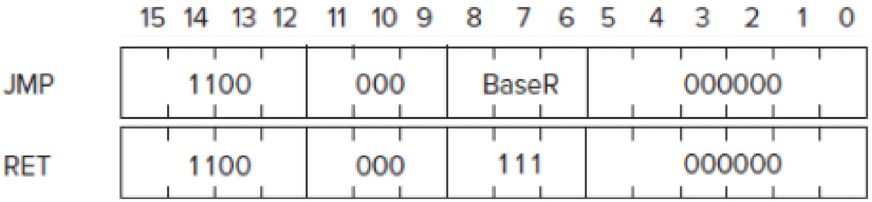
3 = 2 + SEXT(offset)

JMP Label

JMP (Jump), RET (Return from Subroutine)

JMP BaseR

RET



	15 14 13 12	11 10 9	8 7 6	5 4 3 2 1 0
JMP	1100	000	BaseR	000000
RET	1100	000	111	000000

Operation:

JMP: PC = BaseR
RET: PC = $\boxed{\text{R7}}$

- Example:

JMP R1

RET

ORIG.0x3000

0x3000 ADD

0x3001 AND

0x3002 → call fun

0x3003 BR

0x3004 NOT

void fun ( ) {
/* */
return ;
}

⇓

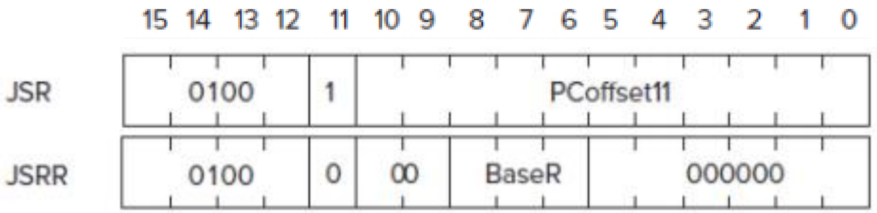
RET

JSR, JSRR (Jump to Subroutine)

```
JSR  LABEL
JSRR BaseR
```

- Operation:

```
Temp = PC;
if (bit[11] == 0)
        PC = BaseR
else
        PC = PC + SEXT(PCoffset11)
R7 = TEMP
```



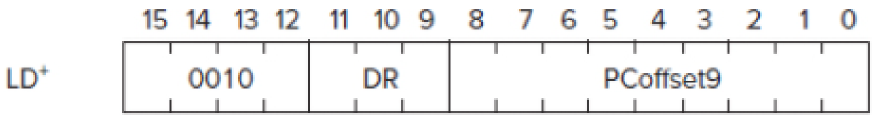
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JSR	0100				1	PCoffset11										
JSRR	0100				0	00		BaseR			000000					

- Example:

```
JSR  QUEUE
JSRR R3
```

LD (Load)

```
LD   DR, LABEL
```



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LD$^+$	0010				DR			PCoffset9								

- Operation:

$$DR = Mem[\underbrace{PC + SEXT(PCoffset9)}_{\text{ADDRESS}}]$$

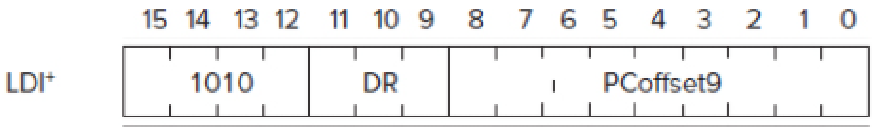
```
setCC();
```

- Example:

```
LD R4, VALUE
```

LDI (Load Indirect)

```
LDI  DR, LABEL
```



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LDI⁺	1010				DR			PCoffset9								

- Operation:

```
DR = Mem[Mem[PC + SEXT(PCoffset9)]]
setCC();
```

ADDRESS

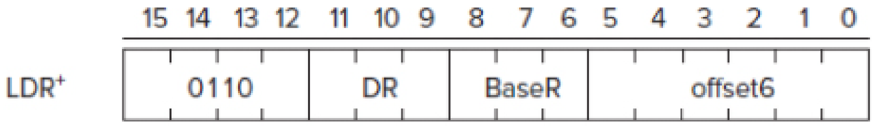
DATA

- Example:

```
LDI R4, InDirADDRESS
```

LDR (Load Base+Offset)

```
LDR  DR,BaseR,Offset6
```



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LDR+	0110				DR			BaseR			offset6					

- Operation:

```
DR = Mem[BaseR + SEXT(offset6)]
setCC();
```

- Example:

```
LDR R4,R2,#-5
```

LEA (Load Effective Address)

```
LEA  DR,LABEL
```

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEA	1110				DR			PCoffset9								

- Operation:

```
DR = PC + SEXT(PCoffset9)
setCC();
```

- Example:

```
LEA R4,TARGET
```

TARGET

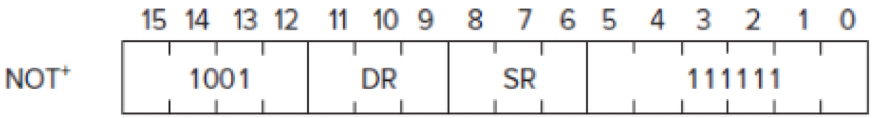
```
        .ORIG   0x3000
0x3000  LEA     R4, L1
0x3001  L1  ADD R5, R4, #1
```

=> R4 ← R2 + 1

R4 ← $\boxed{0x3001}$ ←

NOT (Bitwise complement)

```
NOT  DR,SR
```



	15 14 13 12	11 10 9	8 7 6	5 4 3 2 1 0
NOT+	1001	DR	SR	111111

- Operation:

```
DR = NOT(SR)
setCC();
```

- Example:

0x0000

```
NOT R4,R2
```

N=1 , Z=0 , P=0

R4 = NOT(0x0000)

0xFFFF

2'comp

0b 111111

-1

0x F

0xFFFF...F

RTI (Return from Interrupt)

- Skip for now.

ST (Store)

```
ST   SR, LABEL
```

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST	0011				SR			PCoffset9								

- Operation:

```
Mem[PC + SEXT(PCoffset9)] = SR
```

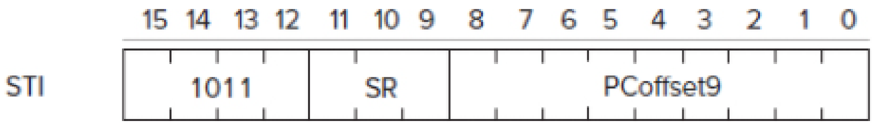
- Example:

```
ST R4, HERE
```

LDI

STI (Store Indirect)

```
STI  SR, LABEL
```



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STI	1011				SR			PCoffset9								

- Operation:

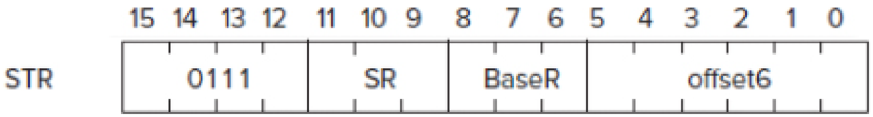
$$\texttt{Mem[}\underbrace{\texttt{Mem[PC + SEXT(PCoffset9)]}}_{\text{Load}}\texttt{] = SR}\ ✓$$

- Example:

```
STI R4, InDirADDRESS
```

STR (Store Base+Offset)

STR SR,BaseR,Offset6



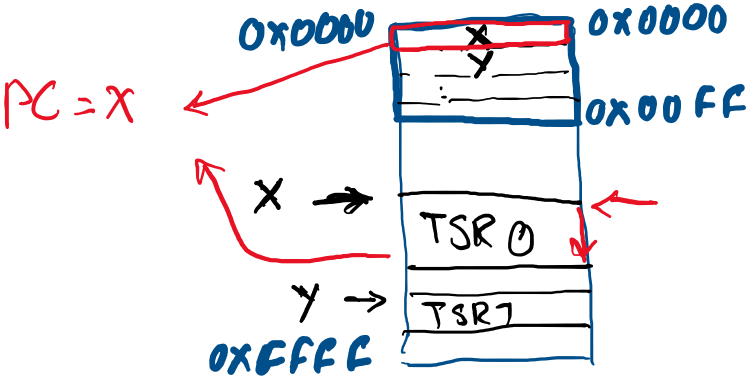
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STR	0111				SR			BaseR			offset6					

- **Operation:**

Mem[BaseR + SEXT(offset6)]= SR

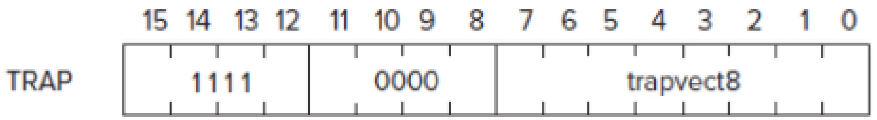
- **Example:**

STR R4,R2,#-5



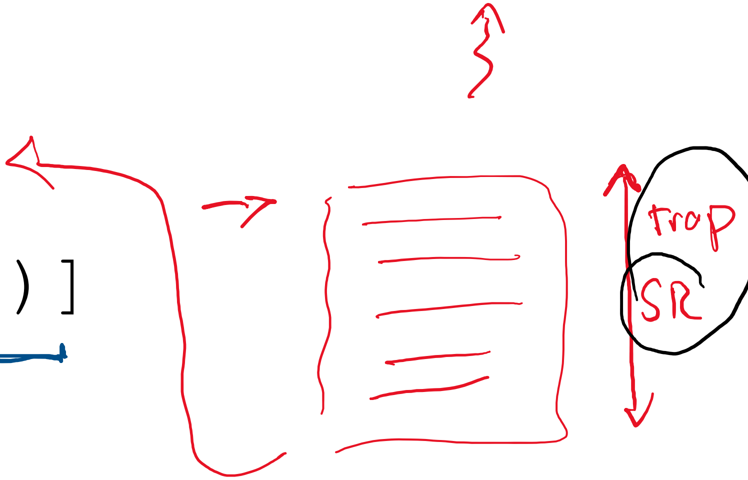
TRAP (System Call)

```
TRAP  trapvect8
```



- Operation:

```
R7 = PC
PC = mem[ZEXT(trapvect8)]
```



- Example:

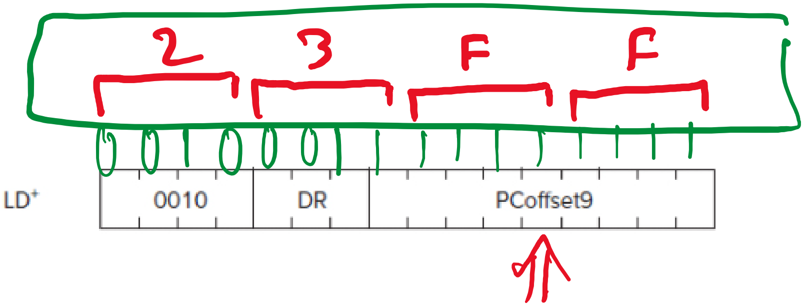
```
TRAP  x23
```

*memory location 0x0000 – 0x00FF implement Trap Vector Table

Trap 0x00

- Example Question : What would be the content of register file after executing the following code? Assume that all registers are initialized to zero.

```
              .ORIG   x1234
x1234  LABEL   LD      R1,LABEL  <=
1235
```



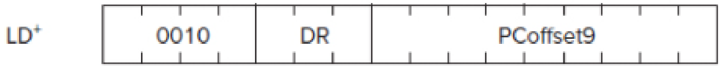
$$\underset{1234}{LABEL} = \underset{1235}{PC} \overset{-1}{+} SEXT(\underset{-1}{offset})$$

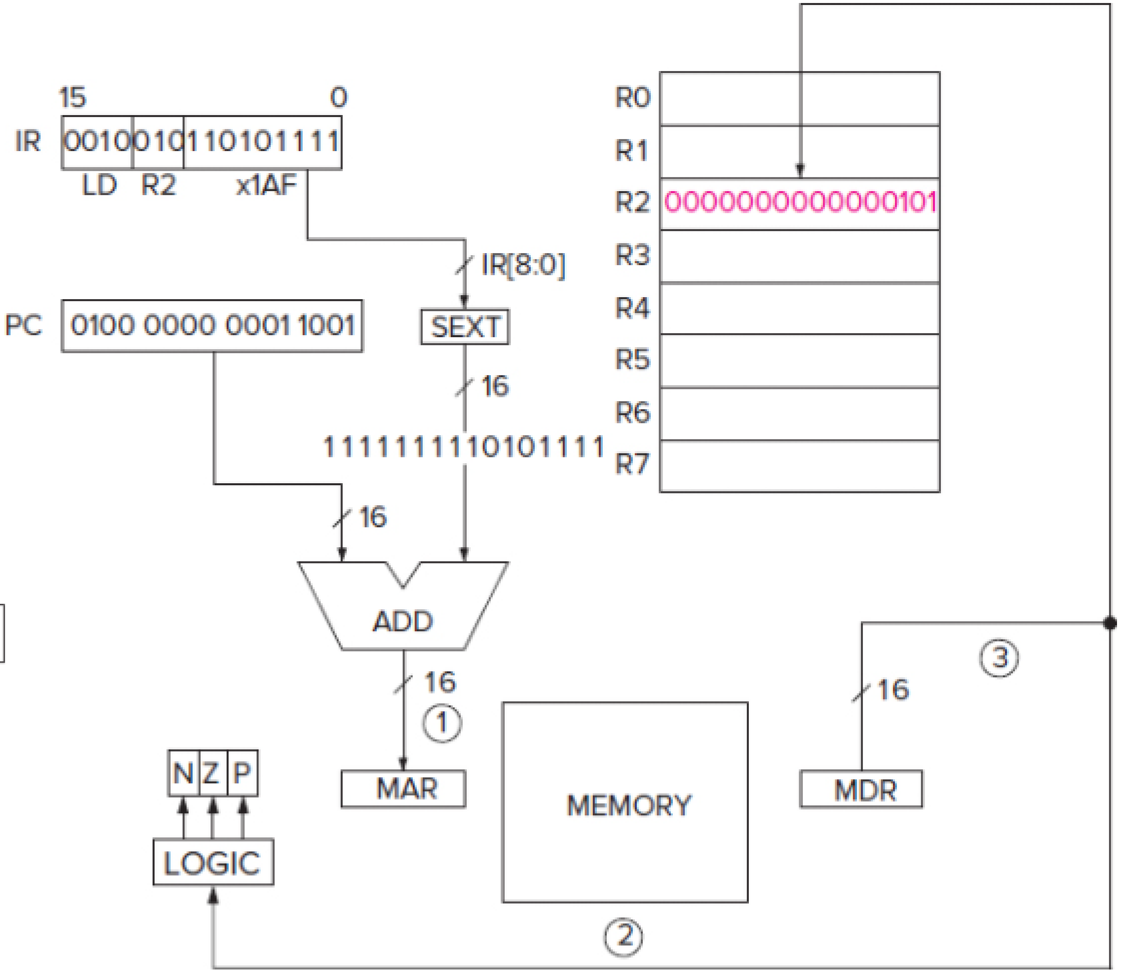
R0	0x0000
R1	~~0x0000~~ X23FF
R2	0x0000
R3	0x0000
R4	0x0000
R5	0x0000
R6	0x0000
R7	0x0000

Review of PC-relative addressing mode

- LD and ST
- Address:

SEXT(bits[8:0]) + PC





More Examples

value of ADDRESS ? x3001

- What is the value of R1 after executing the following LC-3 assembly with the following memory content?

~~1~~ 0

x3000

x3001 ADDRESS

```
.ORIG  x3000
LD      R1, ADDRESS
.BLKW  1
```

R1 ← Mem[ADDESS]

0x3001

```
     .ORIG  x3000
3000 LD     R1, L1
3001 ADD    R3, R3, #1
3002 ADD    R5, R5, #1
3003
L1   .BLKW  1
```

Address	Data (2 bytes)
0x3000	0x2200
0x3001	0x3005
0x3002	0x0001
0x3003	0x0002
0x3004	0x0003
0x3005	0x0004
0x3006	0x0005

R1 ← Mem[L1]

3003

- What is the value of R1 after executing the following LC-3 assembly with the following memory content?

```
                 .ORIG   x3000
x3000            LD      R1, ADDRESS      PC ← x3001
x3001  ADDRESS   .BLKW   1       ADD R2, R3, #1
```

Address	Data (2 bytes)
0x3000	0x2200
0x3001	0x3005
0x3002	0x0001
0x3003	0x0002
0x3004	0x0003
0x3005	0x0004
0x3006	0x0005

$$\underbrace{0010}_{2}\ \underbrace{001}_{2}\ \underbrace{000}_{0}\underbrace{000000}_{0}$$

$$\underbrace{ADDRESS}_{x3001} = \underbrace{PC}_{x3001} + \underbrace{SEXT(Offset)}_{0}$$

x3002   x3001   1

0x2201

- What is the value of R1 after executing the following LC-3 assembly with the following memory content. Assume that the initial value of R2 is 0x3002

```
        .ORIG        x3000
                          -3
        LDR          R1,R2, #-2
```

Base Register

$$R1 = Mem\left[\underbrace{\overset{x3002}{R2}+(-2)}_{x3000}\right]$$

$$\underbrace{0110}_{6}\ \underbrace{001\ 0}_{2}\underbrace{10\ 11}_{B}\ \underbrace{111\ 0}_{E}$$

$$-2 \Downarrow \underbrace{111110}_{-2}$$

Address	Data (2 bytes)
0x3000	0x62BE
0x3001	0x3005
0x3002	0x0001
0x3003	0x0002
0x3004	0x0003
0x3005	0x0004
0x3006	0x0005