ÇANKAYA UNIVERSITY
COMPUTER ENGINEERING DEPARTMENT

Microprocessors Course (CENG329) Laboratory Manual

Experiment 4-B
ARITHMETIC AND LOGICAL INSTRUCTIONS

OBJECTIVE
Learn about the usage of arithmetic and logical instructions available to eZ8 CPU and experiment on each of them.

1. Arithmetic Instructions
The following arithmetic instructions are available for eZ8 CPU:

Mnemonic	Operands	Instruction
ADC	dst, src	Add with Carry
ADCX	dst, src	Add with Carry using Extended Addressing
ADD	dst, src	Add
ADDX	dst, src	Add using Extended Addressing
CP	dst, src	Compare
CPC	dst, src	Compare with Carry
CPCX	dst, src	Compare with Carry using Extended Addressing
CPX	dst, src	Compare using Extended Addressing
DA	dst	Decimal Adjust
DEC	dst	Decrement
DECW	dst	Decrement Word
INC	dst	Increment
INCW	dst	Increment Word
MULT	dst	Multiply
SBC	dst, src	Subtract with Carry
SBCX	dst, src	Subtract with Carry using Extended Addressing
SUB	dst, src	Subtract
SUBX	dst, src	Subtract using Extended Addressing

Study the following code:

```
vector reset = startup
org %1000

startup:
srp #00

ld r0, #07h
loop:           ; A loop for
    clr @r0     ; clearing the
    dec r0      ; first ???
    jr nz, loop ; registers

ld r0, #1h      ; Assign ??'s value to ??
ld r1, #5h      ; Assign ??'s value to ??

inc r2          ; r2 = r2 + 1h
add r2, r0      ; r? = ??
add r2, r1      ; r? = ??

dec r1          ; r1 = ??
sub r1, #2h     ; r1 = ??

ld r3, #4h      ; Assign ??'s value to ??
mult rr2        ; rr2 = ??
```

2. Logical Instructions

The following logical instructions are available for eZ8 CPU:

Mnemonic	Operands	Instruction
AND	dst, src	Logical AND
ANDX	dst, src	Logical AND using Extended Addressing
COM	dst	Complement
OR	dst, src	Logical OR
ORX	dst, src	Logical OR using Extended Addressing
XOR	dst, src	Logical Exclusive OR
XORX	dst, src	Logical Exclusive OR using Extended Addressing

Study the following sample code:

```
vector reset = startup
org %1000

startup:
srp #00

ld r0, #03h
loop:           ; A loop for
    clr @r0     ; clearing the
    dec r0      ; first ???
    jr nz, loop ; registers

ld r0, #10b
ld r1, #1010b

and r1, r0      ; = 0000 0010
or  r1, #111b   ; = ???? ????
xor r1, #110b   ; = ???? ????
com r1          ; = ???? ????
```

EXPERIMENTAL WORK

1. Answer the following questions:
   a) In the first sample code; what is RR2? What is being multiplied in the last line?
   b) In the second sample code; what is '**b**'? Why isn't it '**h**', but '**b**'? In how many other ways these values ending with '**b**' can be written?
2. By examining how each instruction affects the values in memory, try to fill each blank space in example code comments.