2’s comp integers only represent negative #s-False

The set AND OR NOT gates is logically complete-True

In a Von Neumann machine data & instructions both reside in memory-True

The program counter contains the address of the next instruction to be executed-True

The addressing mode is a mechanism for specifying where the operand is located-True

The addressing mode of LDI is pc-relative-False

LEA modifies condition code bits-True

.END is an assembly language instruction-False

.HALT is a trap instruction-True

Using operate type instructions place 45 in R1-AND R1, R1,#0..ADD R1,R1,#15.. ADD R1,R1,#15.. ADD R1,R1,#15

How can you use a single instruction to move contents of R2 into R3?-ADD R3,R2,#0

Lc3 instruction to clear R1?-AND R1, R1, #0

Opcode for any trap?-1111

Decimal value for 2’s compliment of 111111110001::000000001111=-15

119>base5:::119/5=23+**4**:::23/5=4+**3**:::4/5=0+**4=(434)base5**

What is the range of values that can be represented with 16 bits in 2s compliment format?::n\_bits= max: 2^n-1\_-1 min: -2^n-1

Max 2^16-1 -1=**2^15 \_-1** Min:-2^16-1=**-2^15**

If the program is executed, what is the value in R2 at the end of execution?

X30FF 1110 0010 0000 0001 ::: LEA R1, x1

X3100 0110 0100 0100 0010::: LDR R2,R1,x2

X3101 111 000 0010 0101:: TRAP x25 (HALT)

Anything after HALT is DATA

X3102 0001 0100 0100 0001:: x1441

X3103 0001 0100 1000 0010 ::x1482

Contents of R2:x3101 R1:x1482

**Addressing Modes**: LEA=immediate LDR=Base+offset

LD=Pc-relative LDI=indirect ST=Pc-relative STR=base+offset STI=indirect

Operate instructions: ADD,AND,NOT

Data Movement:LD,LDR,LDI,ST,STR,STI

Control Instructions:BR,JMPM,JSR,JSRR

Assembler Directives:.ORIG,.END(anyting with a .)

ADD R1,R1,#0:::sets condition code bits

Highest integer you can ADD is #15

Lowest integer you can ADD is #-16

Hex Dec Hex Dec

0 0000 8 1000

1 0001 9 1001

2 0010 A 1010

3 0011 B 1011

4 0100 C 1100

5 0101 D 1101

6 0110 E 1110

7 0111 F 1111

25.125->binary 25/2=12+**1** .125x2=**0**.25

12/2= 6+**0** 0.25x2=**0**.5

6/2= 3+**0**  0.5x2= **1**.0

3/2=1+**1 =(11001.001)base2**

1/2=0+**1**

IEEE 754 32-bit floating point

|  |  |  |
| --- | --- | --- |
| 1 bit | 8 bits | 23 bits |
| S | Exponent | Fraction |

N=(-1)^s x 1.Fraction x2^(E-127)