## CS2323: HOMEWORK-3

## CS 2323: Home Work - 3

Da. addi 215, 222, -45.

Imm	rs1	Bunct 3	rd	opcode
12	5	3	5	7
-45	22	0	15	0010011
1	<u></u>	1	5	
1111 1101	0011 101	10 000	01111	0010011
1111 1101	0011 1011	0000	0111 100	0011

0xFD3B0793

b. and 223, 28,29.

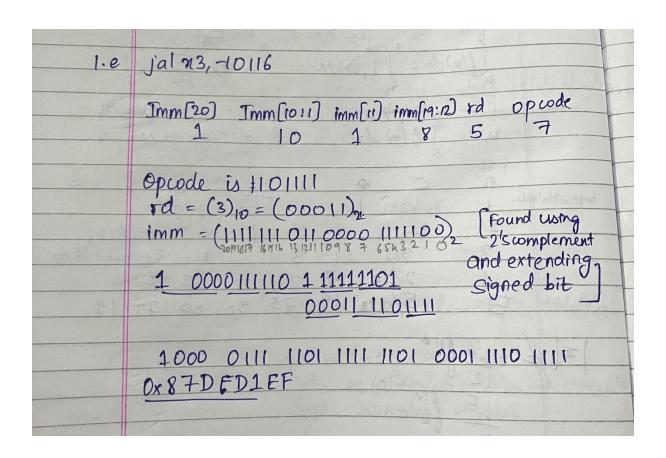
Bunot7	rs2	<b>151</b>	Bunet 3	od	opcode
7	2	5	3	5	7
0	9	8	7	23	0110011
1	i	1	1	1	1
0000000	01001	01000	111	10111	0110011
0000 000	1001 00	0100 0	111 1011		

0x00947BB3

```
imm (12) imm(10:5) rs2 rs1 Bunct3 imm(4:4) imm(11)

1 6 5 5 3 4 1
C. bH x2, x11, 240
 Opcode = 1100011
  Bund3 = (4)10= (100)2
  imm = (240) 10 = (00000 1111 0000) 2
  132=(11)10= (01011)2
  HS1= (2)10= (00010)2
  So, finally
    0 000111 01011 00010 100 1000 0 1100011
    0000 1110 1011 0001 0100 1000 0110
 Ox 0EB14863
d. Sd x19, -54(x1)
  imm[11:5] rs:2 rs:1 funct3 imm[4:0] opcode
7 5 5 3 5 7
 Opcode = 0100011
 funct 3 = (3)_{10} = (011)_2
  752 = (19)_{10} = (10011)_2

751 = (1)_{10} = (00001)_2
   (mm = (54) 0= (1111 1100 1010) 2.
 So, finally.
       1111110 10011 00001 011 01010 D100011
      1111 1101 0011 0000 1011 0101 0010 0011
 Or F030 B523
```



2) a. lixs, OxFFFFFFF

The value OxFFFFFFFF represents -1 in signed 32 bit représentation. Since Risc-V uses 2's complement for negative numbers, the assembler can use '-1' as immediate value instead of the original unsigned representation. The addi instruction with 'no' allows to effectively load \* '-1' into 'x5' by performing the operation O+(-1) as this is a more valid way to & represent the So, the assembler translates this into 'addi x5,x0,-1'. value to be stored. to effectively load signed value into the register.

b. li 25,132.

the value 132 fits within the immediate value range Of addi and thus can use this value directly. The addi instruction adds the immediate value 132 to the XO. This effectively loads the value 132' to register x5 as '0+132', thuis a more efficient way to store load values. So, it translates to addi 25,20,132.

2) (c) li x5, 2134 As 2134 is outside the range of -2018 to 2017,
if is actually translating this immediate value fooding
into Storing the upper part of the immediate value
first into xs. and then adding the remaining
lower part with matrix resulting to the desired
value of 2131. value of 2134. So, it translates into lui x5,0x1 and then addiw 25,25,-1962. (d) li 25, Orzzusabed The immediate value to be loaded into 25 is very large, and larger than the immediate value allowable range [-2048 to 2047]. So, it is translating into multiple instructions. The translation involves: lui 25,0x2345b → loading the upper 20 bits into addiw 25,25,-1075 of which is 0x2345b000x5 this instruction then adds Oxabed to as typically adding loading the lower 12 bits. Oxabed in signed decimal form is -1075.

(b)  $0 \times 06B4D 763$   $0 \times 06B4D 763$ The Department of the line of the line

C.  $O_{x}$ 0169CF9 3

O I 6 9 C F 91013) = 0(11) = 221

0000 0001 0110 [001 1100 1111 1001 0011]

The last 7 bits [0010011] represents Informat instruction and funct3 [14:12]  $\hat{y}$  (00)2  $\hat{y}$  (00)2  $\hat{y}$  (4) 101 which represents the xori instruction of 101 0111 0000

imm = (0000 0001 0110) 2 = (22) 10 20 21830 x 0

rs1 = (10011) 2 = (19) 10. , rd = (11111) 2 = (31) 10

So, the instruction is 2000 x 221 221 1000 x 231 x