HW2 CS2323

Name :- Darpan Gaur
Roll No :- CO21BTECH11004

DATE: / /
PAGE:

Que 1 a) addi x15, x22, -45

addi → I type

31 … 20	19 … 15	14 … 12	11 … 7	6 … 0
imm[11:0]	rs1	funct3	rd	opcode
		000		0010011

(from RISCV Card)

x15 → rd ⇒ 01111 → (15 in binary)

x22 → rs1 ⇒ 10110 → (22 in binary)

-45 → imm ⇒ 2's complement of 45

45 ⇒ 0000 0010 1101

2's complement of 45 ⇒ 1111 1101 0010 + 1

⇒ 1111 1101 0011 → imm

∴ addi x15, x22, -45

Binary ⇒ 1111 1101 0011 (imm) 10110 (rs1) 000 (funct3) 01111 (rd) 0010011 (opcode)

In hex ⇒ $\boxed{0xfd3b0793}$

b) and x23, x8, x9

and → R type

31 … 25	24 … 20	19 … 15	14 … 12	11 … 7	6 … 0
funct7	rs2	rs1	funct3	rd	opcode
0000000			111		0110011

(from RISCV Card)

x23 → rd ⇒ 10111 → (23 in binary)

x8 → rs1 ⇒ 01000 → (8 in binary)

x9 → rs2 ⇒ 01001 → (9 in binary)

Assembly ⇒ and x23, x8, x9

Binary ⇒ 0000 000 (funct7) 01001 (rs2) 01000 (rs1) 111 (funct3) 10111 (rd) 0110011 (opcode)

In Hexadecimal ⇒ $\boxed{0x00947bb3}$

DATE: / /
PAGE:

c) blt x2, x11, 240

blt → B-type

31	25 24	20 19	15 14	12 11	7 6	0
imm[12] imm[10:5]	rs2	rs1	funct3	imm[4:1] imm[11]	opcode	
			100		1100011	

(from RISCV card)

x2 → rs1 ⇒ 000010 (1 in binary)

x11 → rs2 ⇒ 01011 (11 in binary)

240 → imm ⇒ 0000 1111 0000 (240 in binary)

imm[12] imm[10:5] imm[4:1]

Assembly ⇒ blt x2, x11, 240

imm[4:1] imm[11]

Binary ⇒ 0000 1110 1011 0001 0100 1000 0110 0011

imm[12] imm[10:5]

imm rs2 rs1 funct3 imm opcode

In Hexadecimal ⇒ $\boxed{0x0eb14863}$

d) sd x19, -54(x1)

sd → S-type

31	25 24	20 19	15 14	12 11	7 6	0
imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode	
			011		0100011	

(from RISCV card)

x19 → rs2 ⇒ 10011 (19 in binary)

x1 → rs1 ⇒ 00001 (1 in binary)

-54 → imm ⇒ 2's complement of 54

54 ⇒ 0000 0011 0110

2's complement of 54 ⇒ 1111 1100 1001 + 1

⇒ 1111 1100 1010

imm[11:5] imm[4:0]

Assembly ⇒ sd x19, -54(x1)

~~Binary ⇒ 1111 1101 0011 0000 1011 0010 0012~~

~~imm opcode~~

Binary ⇒ 1111 1101 0011 0000 1011 0101 0010 0011

imm rs2 rs1 funct3 imm opcode

In Hexadecimal ⇒ $\boxed{0xfd30b523}$

DATE: / /
PAGE:

c) jal x3, -10116

jal → J-Type

31	12 11	7 6	0
imm[20|10:1|11|19:12]	rd	opcode	
		1101111	

(from RISCV card)

x3 → rd ⇒ 00011 (3 in binary)

-10116 → imm ⇒ 2's complement of 10116

10116 ⇒ 0000 00100111 2 1000 0100

2's complement of 10116 ⇒ 1111 1101 1000 0111 1011 + 1

⇒ $\underbrace{1111\ 1101}_{imm[19:12]}\ \underset{imm[11]}{1}\ \underbrace{000\ 0111\ 2100}_{imm[10:1]}$

~~Bomo~~ Assembly ⇒ jal x3, -10116

Binary $\underset{imm[20]}{1}\ \underbrace{000\ 0111\ 1101}_{imm[10:1]}\ \underset{imm[11]}{1}\ \underbrace{1111\ 1101}_{imm[19:12]}\ \underbrace{0001\ 1}_{rd}\ \underbrace{110\ 1112}_{opcode}$

In Hexadecimal ⇒ ~~0x0b14863~~ 0x87dfd1ef

DATE: / /
PAGE:

Que 2 a) li x5, -1 → x5 = x0 - 1 = 0 - 1 = -1

Disassembled code :- addi x5, x0, -1

- Here we want to load x5 with -1.
- addi → I type → Imm → 12 bits
- So we can load values with x0 in range [-2048, 2047] using addi by adding. -1 is in this range.

b) li x5, 0xFFFFFFFF

Disassembled code :- addi x5, x0, -1

(0xFFFFFFFF)Hex → (-1)decimal [2's complement Representation]

- Load -1 to x5
- Use addi → I-type → Imm → 12 bit, so can load values in range [-2048, 2047] by adding with x0.

c) li x5, 132

Disassembled code :- addi x5, x0, 132

- Here we want to load 132 to x5.
- Use addi → I type → Imm → 12 bits, so can load values in range [-2048, 2047] by adding with x0.

d) li x5, 2134

Disassembled code :- lui x5 0x1
addiw x5 x5 -1962

- 2134 not in range [-2048, 2047] so single addi can't be used.
- lui x5 0x1 → sets first 20 bits
  ↳ U type → Imm → 20 bits → (Imm << 12)
  Here, Imm = 1 so, x5 = (1 << 12) = 4096
- Now addiw x5 x5 -1962

add immediate word ✓ ↳ adds sign extended 12-bit to produce sign extended 32-bit

addi x5 x5 -1962 ⇒ x5 = x5 - 1962 → 4096 - 1962 = 2134

- 2134 is loaded in x5

DATE: / /
PAGE:

e) li x5, 0x2345abcd

Disassembled code:- lui x5 0x2345b
addiw x5 x5 -1075

- First set first 20 bits with 0x 2345b (greater than 0x2345a), using lui
- Then use addiw, subtract 1075, to get 0x 2345abcd.
- Similar to que d.

Que3 (1) lhu x3, 0(x1)

$\boxed{0x\ 0000\ 0000\ 0000\ 3939}$

- lhu → load half word (unsigned) → zero extends
So, 16 bits from 0(x1) & extend 0.
↳ Byte 0

(2) lh x3, 0(x1)

$\boxed{0x\ 0000\ 0000\ 0000\ 3939}$

- lh → load half word (16 bit) & extend sign-bit
here 3939
↓
extend 0 ← 0011
for rest 48 bits

(3) lh x3 2(x1)

$\boxed{0x\ ffff\ ffff\ ffff\ 9393}$

- lh → load half word (16 bit) & extend sign bit.
2(x1) → load from Byte 2 of from (x1)
So, 9393
Here 9393
↓
1001
↓
signbit so extend 1 for rest 48 bits [1111 → f]

DATE: / /
PAGE:

(4) ld x3, 0(x1) → ~~0xa55a a5a5 9393~~ 0x10000

- ld → load double word (64 bits) from x1 (0(x1))

$\boxed{0x\ a55a\ a5a5\ 9393\ 3939}$

(5) lw x3, 12(x1)

$\boxed{0x\ 0000\ 0000\ 3993\ 3939}$

- lw → load word (32 bits) & extend sign bit

12 (x1) → Byte 12 from 0(x1).

Here for sign-bit 3993

sign bit → 0011

∴ Extend 0 for 32 bits.

(6) lbu x3, 7(x1)

$\boxed{0x\ 0000\ 0000\ 0000\ 0a5}$

- lbu → load byte (8 bits) & extend zero

7(x1) → first 8 bits from MSB in first word

∴ a5

then extend 0 for rest 56 bits.

(7) lb x3, 7(x1)

$\boxed{0x\ ffff\ ffff\ ffff\ ffa5}$

- lb → load byte (8 bits) then extend sign bit.

7(x1) → first 8 bits from MSB in first word

∴ a 5

1010

sign bit, so extend 1

for rest 56 bits [1111 → f]

DATE: / /
PAGE:

(8) lb x3, 6(x1)

$\boxed{0x0000\ 0000\ 0000\ 005a}$

- lb → load byte (8 bits) then extend sign bit

6 (x1) → Byte 6 ~~Byte 0 = 0(x1)~~

0x a5 5a a5 a5 93 93 39 69

Byte 6 5 4 3 2 1 0

5a

0 1 0 1

sign-bit so extend 0 for rest 56 bits.

Que 4

- .data section in our code maps to address 0x10000000.
- It is not fixed address, it is by default in the simulator settings.
- Yes, it is configurable.

To change it, follow the following steps: (in simulator)

- Click Edit → click settings
- Select compiler
- Scroll down.
- In Assembler section, change the address for ".data section start address"
- Click ok.