Computer Architecture (CS2323) - Assignment 2

CO22BTECH11006
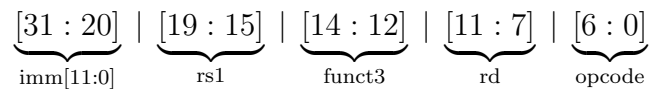
OM DAVE

Question 1

a) `addi x15, x22, -45`

Format: I-type
Format layout (bits):

$$\underbrace{[31:20]}_{\text{imm[11:0]}} \mid \underbrace{[19:15]}_{\text{rs1}} \mid \underbrace{[14:12]}_{\text{funct3}} \mid \underbrace{[11:7]}_{\text{rd}} \mid \underbrace{[6:0]}_{\text{opcode}}$$

Fields:

imm (decimal)	-45
imm (12-bit two's complement)	1111 1101 0011
rs1 = x22	10110
funct3 (addi)	000
rd = x15	01111
opcode (OP-IMM)	0010011

Assembled (fields in order imm[11:0] — rs1 — funct3 — rd — opcode):

111111010011 10110 000 01111 0010011
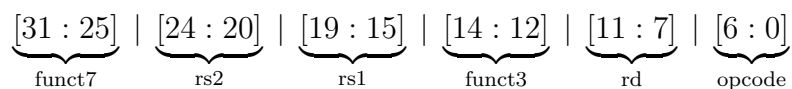
1111 1101 0011 1011 0000 0111 1001 0011

Hex (32-bit): 0xFD3B0793

---

b) `and x23, x8, x9`

Format: R-type
Format layout (bits):

$$\underbrace{[31:25]}_{\text{funct7}} \mid \underbrace{[24:20]}_{\text{rs2}} \mid \underbrace{[19:15]}_{\text{rs1}} \mid \underbrace{[14:12]}_{\text{funct3}} \mid \underbrace{[11:7]}_{\text{rd}} \mid \underbrace{[6:0]}_{\text{opcode}}$$

Fields:

$$
\begin{array}{rl}
\text{funct7 (and)} & \texttt{0000000} \\
\text{rs2} = \text{x9} & \texttt{01001} \\
\text{rs1} = \text{x8} & \texttt{01000} \\
\text{funct3 (and)} & \texttt{111} \\
\text{rd} = \text{x23} & \texttt{10111} \\
\text{opcode (OP / R-type)} & \texttt{0110011}
\end{array}
$$

Assembled (funct7 — rs2 — rs1 — funct3 — rd — opcode):

0000000 01001 01000 111 10111 0110011

0000 0000 1001 0100 0111 1011 1011 0011

Hex (32-bit): 0x00947BB3

---

c) `blt x2, x11, 240`

Format: B-type (branch)
B-type layout:

$$
\underbrace{[31]}_{\text{imm[12]}} \mid \underbrace{[30:25]}_{\text{imm[10:5]}} \mid \underbrace{[24:20]}_{\text{rs2}} \mid \underbrace{[19:15]}_{\text{rs1}} \mid \underbrace{[14:12]}_{\text{funct3}} \mid \underbrace{[11:8]}_{\text{imm[4:1]}} \mid \underbrace{[7]}_{\text{imm[11]}} \mid \underbrace{[6:0]}_{\text{opcode}}
$$

Immediate: $+240$ (decimal):

$$240_{10} = 00\,0011\,1100\,00_2 \text{ (13 bits: } 000011110000)$$

Breaking to encoded pieces:

- imm[12] = 0

- imm[11] = 0

- imm[10:5] = 000111 (6 bits)

- imm[4:1] = 1000 (4 bits)

- imm[0] = 0 (implicit / must be zero)

Other fields:

$$
\begin{array}{rl}
\text{rs1} = \text{x2} & \texttt{00010} \\
\text{rs2} = \text{x11} & \texttt{01011} \\
\text{funct3 (blt)} & \texttt{100} \\
\text{opcode (BRANCH)} & \texttt{1100011}
\end{array}
$$

Assembled (imm[12] — imm[10:5] — rs2 — rs1 — funct3 — imm[4:1] — imm[11] — opcode):

0 000111 01011 00010 100 1000 0 1100011
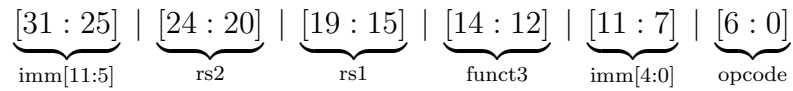
0000 1110 1011 0001 0100 1000 0110 0011

Hex (32-bit): 0x0EB14863

---

d) `sd x19, -54(x1)`

Format: S-type (store)
S-type layout:

$$\underbrace{[31:25]}_{\text{imm[11:5]}} \mid \underbrace{[24:20]}_{\text{rs2}} \mid \underbrace{[19:15]}_{\text{rs1}} \mid \underbrace{[14:12]}_{\text{funct3}} \mid \underbrace{[11:7]}_{\text{imm[4:0]}} \mid \underbrace{[6:0]}_{\text{opcode}}$$

Immediate: -54. Represent as 12-bit two's complement:

$$-54 \xrightarrow{\text{12-bit two's comp}} 1111\,1100\,1010$$

Split into:

- imm[11:5] = `1111110` (bits 31:25)

- imm[4:0] = `01010` (bits 11:7)

Other fields:

$$\begin{aligned}
\text{rs2} = \text{x19 (value to store)} &\quad \texttt{10011} \\
\text{rs1} = \text{x1 (base)} &\quad \texttt{00001} \\
\text{funct3 (sd)} &\quad \texttt{011} \\
\text{opcode (STORE)} &\quad \texttt{0100011}
\end{aligned}$$

Assembled (imm[11:5] — rs2 — rs1 — funct3 — imm[4:0] — opcode):

`1111110 10011 00001 011 01010 0100011`
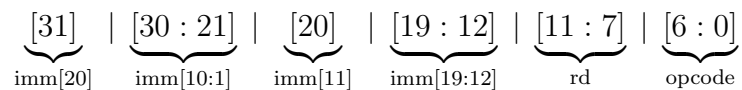
`1111 1101 0011 0000 1011 0101 0010 0011`

Hex (32-bit): `0xFD30B523`

---

e) `jal x3, -10116`

Format: J-type (JAL)
J-type layout:

$$\underbrace{[31]}_{\text{imm[20]}} \mid \underbrace{[30:21]}_{\text{imm[10:1]}} \mid \underbrace{[20]}_{\text{imm[11]}} \mid \underbrace{[19:12]}_{\text{imm[19:12]}} \mid \underbrace{[11:7]}_{\text{rd}} \mid \underbrace{[6:0]}_{\text{opcode}}$$

Immediate: -10116. Represent as 21-bit signed immediate (imm[20:0]):

$$-10116 \xrightarrow{\text{21-bit two's comp}} 1\,1111101\,1\,00001111100\,0$$

- imm[20] = `1`

- imm[10:1] = `0000111110`   (10 bits)

- $\text{imm}[11] = 1$

- $\text{imm}[19{:}12] = 11111101$ (8 bits)

Other fields:

$$\text{rd} = \text{x3} \quad \texttt{00011}$$
$$\text{opcode (JAL)} \quad \texttt{1101111}$$

Assembled (imm[20] — imm[10:1] — imm[11] — imm[19:12] — rd — opcode):

`1 0000111110 1 11111101 00011 1101111`

`1000 0111 1101 1111 1101 0001 1110 1111`

Hex (32-bit): `0x87DFD1EF`

Question 2

(a) `li x5, 0xFFFFFFFFFFFFFFFF` (i.e. -1 in two's complement)

> **Reason:** -1 fits in signed 12-bit immediate ($-1 \in [-2048, 2047]$).
> **Expansion (1 real instruction):**
>
>   `addi x5, x0, -1`

(b) `li x5, 132` ($132_{10} = 0x84$)

> **Reason:** 132 lies in signed 12-bit range.
> **Expansion (1 real instruction):**
>
>   `addi x5, x0, 132`

(c) `li x5, 2134` ($2134_{10} = 0x856$)

> **32-bit binary of decimal:** $2134_{10} = $ 0000 0000 0000 0000 0000 1000 0101 0110$_2$.
> **split 20—12:** top20 $[31:12] = $ 0000 0000 0000 0000 0000, low12 $[11:0] = $ 1000 0101 0110 $= 0x856$.
> **Rounding rule for `lui/addi`:** low12 $\geq 0x800 \Rightarrow$ carry 1 into top20. Thus upper20 $= 0x00001 \Rightarrow$ `lui` loads $0x00001 \ll 12 = 0x1000$.
> **Signed low12 for `addi`:** lower $= 2134 - 0x1000 = -1962$, 12-bit two's comp $= -1962$.
> **Expansion (2 real instructions):**
>
>   `lui  x5, 0x00001`
>   `addi x5, x5, -1962`

(d) `li x5, 0x000000002345abcd`

> **32-bit binary of Hexadecimal:** 0010 0011 0100 0101 1010 1011 1100 1101_2.
> **split 20—12:** top20 $[31 : 12] = $ 0010 0011 0100 0101 1010 $= 0x2345A$, low12
> $[11 : 0] = $ 1011 1100 1101 $= 0xBCD$.
> **Rounding rule:** low12 $\geq 0x800 \Rightarrow$ carry 1 into top20. Thus `upper20` $= 0x2345B$
> $\Rightarrow$ `lui` loads $0x2345B \ll 12 = 0x2345B000$.
> **Signed low12 for** `addi:` `lower` $= 0x2345ABCD - 0x2345B000 = -1075$, 12-bit
> two's comp $= -1075$.
> **Expansion (2 real instructions):**
>
> ```
>   lui  x5, 0x2345B
>   addi x5, x5, -1075
> ```

Question 3

a) `0x0019F233`

Hex to binary:

$$0x0019F233 = 0000\ 0000\ 0001\ 1001\ 1111\ 0010\ 0011\ 0011_2$$

Format: R-type

$$\underbrace{[31:25]}_{\text{funct7}} \mid \underbrace{[24:20]}_{\text{rs2}} \mid \underbrace{[19:15]}_{\text{rs1}} \mid \underbrace{[14:12]}_{\text{funct3}} \mid \underbrace{[11:7]}_{\text{rd}} \mid \underbrace{[6:0]}_{\text{opcode}}$$

Fields:

opcode	0110011 (R-type)
funct7	0000000
rs2	00001 = x1
rs1	10011 = x19
funct3	111 → AND
rd	00100 = x4

Assembly: `and x4, x19, x1`

---

b) `0x06B4D763`

Hex to binary:

$$0x06B4D763 = 0000\ 0110\ 1011\ 0100\ 1101\ 0111\ 0110\ 0011_2$$

$$0x06B4D763 = 0\ 000011\ 01011\ 01001\ 101\ 0111\ 0\ 1100011_2$$

Format: B-type (branch)

$$\underbrace{[31]}_{\text{imm[12]}} \mid \underbrace{[30:25]}_{\text{imm[10:5]}} \mid \underbrace{[24:20]}_{\text{rs2}} \mid \underbrace{[19:15]}_{\text{rs1}} \mid \underbrace{[14:12]}_{\text{funct3}} \mid \underbrace{[11:8]}_{\text{imm[4:1]}} \mid \underbrace{[7]}_{\text{imm[11]}} \mid \underbrace{[6:0]}_{\text{opcode}}$$

Fields:

opcode	1100011 (branch)
funct3	101 → BGE
rs1	01001 = x9
rs2	01011 = x11
imm	from bits → 0000001101110 = 110 (decimal)

Assembly: `bge x9, x11, 110`

---

c) `0x0169CF93`

Hex to binary:

$$0x0169CF93 = 0000\ 0001\ 0110\ 1001\ 1100\ 1111\ 1001\ 0011_2$$

Format: I-type

$$\underbrace{[31:20]}_{\text{imm[11:0]}} \mid \underbrace{[19:15]}_{\text{rs1}} \mid \underbrace{[14:12]}_{\text{funct3}} \mid \underbrace{[11:7]}_{\text{rd}} \mid \underbrace{[6:0]}_{\text{opcode}}$$

Fields:

opcode	0010011 (I-type)
funct3	100 → XORI
rs1	10011 = x19
rd	11111 = x31
imm	000000010110 = 22 (decimal)

Assembly: `xori x31, x19, 22`