CPU

Instruction Types: RISUBJ

- R → (Register Type) → Arithmetic / logical operations on two registers. (0110011) (0x33) and/or/xor (all operations in the ALU)
- I (Immediate) → One operand is a register and another is an immediate (a constant) embedded in the instruction.
  All arithmetic operations can be done with one register and one immediate operand.
  Other than that, lw → load word from memory
  Jalr → Jump and link register
- S (store) → Taking a value from register and storing it at a memory address.

	funct3	
sb → store byte (8 bit)	000	(0100011) (0x
sh → store half word (16 bit)	001	
sw → store word (32 bit)	010	

rs1 → base address.
rs2 → holds value to be stored.
imm → offset added to rs1.
funct3 → store type (sb, sh, sw).
opcod → identify as store instruction.

imm	rs2	rs1	funct3	imm	opcode
7	5	5	3	5	7

R-type format

funct7	rs2	rs1	funct3	rd	opcode
7	5	5	3	5	7.

	funct7	funct3	
add	0000000	000	SLL
sub	0100000	000	SRL
			SRA
sll	0000000	001	SLT
slt	"	010	SLTU
sltu	"	011	
XOR	"	100	
SRL	0000000	101	
SRA	0100000	101	
OR	"	110	
AND	"	111	

⊙ U type (upper)

imm | rd | opcode
20 5 7

1) lui (0110111)

↳ Add 12 trailing zeroes.

Fill the destination register with immediate.

2) auipc (0010111)

Adds upper immediate to current program counter.

x5 ← PC + (0x10000 << 12) (get PC relative address for Jumps)

⊙ B-Type (Branch) (1100011)

→ Format

im[12]	imm[10:5]	rs2	rs1	funct3	imm[4:1]	imm[11]	opcode
1	6	5	5	3	4	1	7

imm → 13 bit signed offset, tell ×2, tells how far to jump.

	funct3	
beq	000	=
bne	001	!=
blt	100	<
bge	101	>=
bltu	110	< (u)
bgeu	111	>= (u)

⊙ J-type (1101111) (Jump)

imm[20] | imm[10:1] | imm[11] | imm[19:12] | rd | opcode

Jal → rd = PC+4 // saves the next instruction.

PC = pc+offset // Jumps

offset = [imm << 1]

Immediate codes

1) For ALU i same funct3 and funct7, Just the opcode is 0010011 (0x63)

2) For Load types, opcode is 0000011 (0x03)
(Similar to store types).

	funct3.
lb	000
lh	001
lw	010
lbu	100
lhu	101

3) For Jump (Jalr) opcode 1100111 funct3 000.

4) Environment calls. opcode (1110011)

imm[11:0]

000....000	ecall	000
000....001	ebreak	000

> They differ by immediate field.

Overall Same format | imm[11:0] | rs1 | funct3 | rd | opcode |

12 5 3 5 7

Control Unit (Main)

	opcode	control signal → Requisite
R-type	0110011 (33)	10000010
I (ALU)	0010011 (13)	11000011
I (LOAD)	0000011 (03)	11101000
I (JALR)	1100111 (67)	11001000
I (System)	1110011 (73)	0000x000
S-type	0100011 (23)	0101x000
U (lui)	0110111 (37)	11000000
U (auipc)	0010111 (27)	11000000
B-type	1100011 (63)	0000x101
J (JAL)	1101111 (6F)	1x001000

ALUsrc
MemRead
MemWrite
MemtoReg
Branch
$ALUOp_2$

Immediate Generator

Ins

We use a 8→1 mux. & 7→3 ROM

I = 000
S = 001
B = 010
U = 011
J = 100

ALU Control

ADD (0000)
SUB (0001)
AND (0010)
OR (0011)
XOR (0100)
SLL (0101)
SRL (0110)
SRA (0111)
SLT (1000)
SLTU (1001)

Control Bits
$ALUOP_2$ funct3 funct7
↓
10 → Normal operations.