VSDSquadron Mini Research Internship
Task-3

1. RISC-V:

RISC-V (pronounced "risk-five") is a new instruction-set architecture (ISA) that is freely available to academia and industry. It supports both 32-bit and 64-bit address space variants for applications, operating system kernels, and hardware implementations. The system-level organization of a RISC-V hardware platform can range from a single-core microcontroller to a many-thousand-node cluster of shared-memory manycore server nodes. Even small systems-on-a-chip might be structured as a hierarchy of multicomputer and/or multiprocessors to modularize development effort or to provide secure isolation between subsystems.

2. Instruction Formats:

An instruction format defines the structure of a machine-level instruction, specifying how different fields (like opcode, operands, and immediate values) are organized. It ensures compatibility and efficient decoding by the processor.

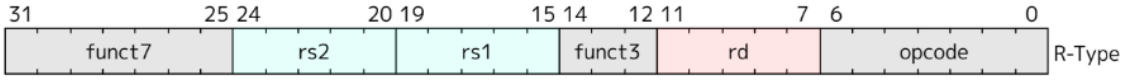
There are 6 instruction formats:

- R-format
- I-format
- S-format
- B-format
- U-format
- J-format

2.1. R-format Instruction:

Used for arithmetic and logical operations that operate entirely within registers.
Fields: Opcode, rs1 (source register 1), rs2 (source register 2), rd (destination register), funct3, funct7.

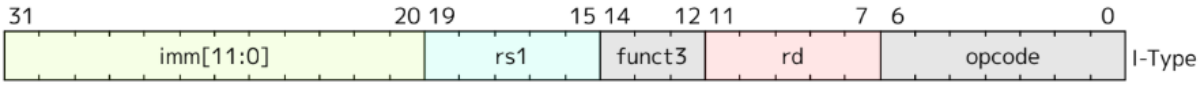
rs1 and rs2 will serve as operands, and the operation defined by func3 and func7 will be performed between rs1 and rs2, with the result being stored in rd and opcode defines the type.



2.2. I-format Instruction:

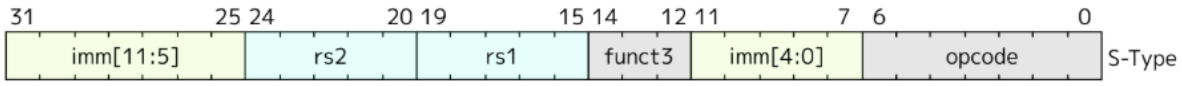
Used for operations involving immediate values (e.g., load instructions, arithmetic with an immediate).
Fields: Opcode, rs1, rd, funct3, and a 12-bit immediate value.

The operation will be performed between the source register *rs1* and a 12-bit constant, with *func3* defining the operation to be executed, and the result will be stored in *rd*.



2.3. S-format Instructions

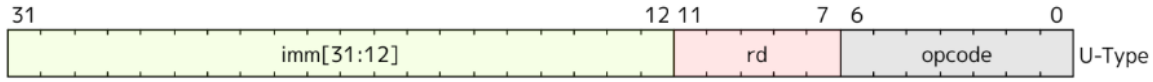
Used for store operations where data is saved from a register to memory.
Fields: Opcode, rs1, rs2, funct3, and a split immediate (7-bit + 5-bit).



2.4. U-format Instruction:

Used for loading 20-bit upper immediate values into the destination register.

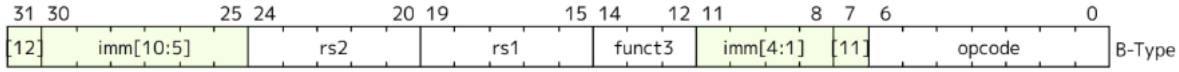
Fields: Opcode, rd, and a 20-bit immediate.



2.5. B-format Instruction:

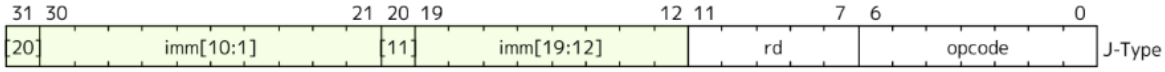
Used for conditional branching based on register comparisons.

Fields: Opcode, rs1, rs2, funct3, and a split immediate for the branch target.



2.6. J-format Instruction:

Used for jump instructions, specifying a target address for control flow.
Fields: Opcode, rd, and a 20-bit split immediate.



3. Application code: 4 x 1 Mux:

```
0000000000010184 <main>:
   10184:       ff010113                addi    sp,sp,-16
   10188:       00113423                sd      ra,8(sp)
   1018c:       00100713                li      a4,1
   10190:       00000693                li      a3,0
   10194:       00100613                li      a2,1
   10198:       00000593                li      a1,0
   1019c:       00021537                lui     a0,0x21
   101a0:       1a050513                addi    a0,a0,416 # 211a0 <__clzdi2+0x3c>
   101a4:       290000ef                jal     ra,10434 <printf>
   101a8:       00000613                li      a2,0
   101ac:       00100593                li      a1,1
   101b0:       00021537                lui     a0,0x21
   101b4:       1c850513                addi    a0,a0,456 # 211c8 <__clzdi2+0x64>
   101b8:       27c000ef                jal     ra,10434 <printf>
   101bc:       00100593                li      a1,1
   101c0:       00021537                lui     a0,0x21
   101c4:       1e850513                addi    a0,a0,488 # 211e8 <__clzdi2+0x84>
   101c8:       26c000ef                jal     ra,10434 <printf>
   101cc:       00000513                li      a0,0
   101d0:       00813083                ld      ra,8(sp)
   101d4:       01010113                addi    sp,sp,16
   101d8:       00008067                ret
```