# Instruction Set Architecture

For instructions: ADD, SUB, AND, OR, XOR, NOT.
i.e.… register addressing ALU performed NON-SHIFT operations.

| OP-Code | RS | RD(destination) | RT | Don't care | Function-Code |
|---|---|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-11 | 10-6 | 5-0 |

Operations supported:
ADD R1, R2, R3
SUB R3, R4, R5
AND R5, R4, R2
SLA R4, R5, R3 //R4 <= R5<<R3 [0]

For instructions: ADDI, SUBI, ANDI, ORI, XORI, NOTI.
i.e.… immediate addressing ALU performed NON-SHIFT operations.

| OP-Code | RS | RD(destination) | Immediate addressed Operand- 2 |
|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-0 |

Operations supported:
ADDI R3 #25 // R3 = R3 + 25
SUBI R5 #20 // R5 = R3 – 20
ANDI R3 R5 #250 // R5 = R3 & $(250)_2$

For instructions: SLA, SRA, and SRL.
i.e.… register addressing ALU performed SHIFT operations.

| OP-Code | RS | RD(destination) | RT(RT[0]used as shamt ) | Don't care | Function-Code |
|---|---|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-11 | 10-6 | 5-0 |

Operations supported:
SLA R4, R5, R3 //R4 <= R5<<R3 [0]
SRA R4, R5, R3 //R4 <= R5<<R3 [0]
SRL R4, R5, R3 //R4 <= R5<<R3 [0]

For instructions: NOT, MOVE
i.e.…register addressing ALU performed unary operations.
For MOVE

| OP-Code | RS | RD(destination) | Don't care |
|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-0 |

For NOT

| OP-Code | RS | RD(destination) | Don't care | Function-Code |
|---|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-6 | 5-0 |

Operations supported:
NOT R3, R4
MOVE R5, R4
MOVE SP, R5

For instructions: LOAD, STORE, and LDSP
i.e….. register addressing and interacting with main memory.
For LDSP we use similar syntax but RD is always Stack Pointer.

| OP-Code | RS | RD | Immediate addressed OFFSET |
|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-0 |

Operations supported:
LD R2, 10(R6) // R2 <= Mem [R6+10]
ST R2,-2(R11) // Mem [R11-2] <= R2
LDSP SP, 0(R2) // SP <= Mem [R2+0]
STSP SP, 100(R7) // Mem [R7+100] <= SP

For instructions: BMI, BR, BPL, BZ.
i.e.…register addressing is used, and for branching instructions.
For BMI, BPL, BZ.

| OP-Code | RS | RD(test register) | Immediate addressed OFFSET |
|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-0 |

For BR.

| OP-Code | RS | Don't care | Immediate addressed OFFSET |
|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-0 |

Operations supported:
BR 10(R2) // PC = Mem [R2+10]
BMI R5, 10(R2) // PC = Mem [R2+10] if R5<0
BPL R5, 10(R2) // PC = Mem [R2+10] if R5>0
BZ R5, 10(R2) // PC = Mem [R2+10] if R5=0

For instructions: PUSH, POP, CALL, and RET.
i.e.…for all Stack Operations.

For Push

| OP-Code | RS | Don't care |
|---|---|---|
| 31-26 | 25-21 | 20-0 |

For POP

| OP-Code | Don't Care | RD | Don't care |
|---|---|---|---|
| 31-26 | 25-21 | 20-16 | 15-0 |

Operations supported:
PUSH R6 // Push R6 in the stack
POP R10 // Pop from stack and store in R10
CALL #1000 // SP <= SP – 4; Mem [SP] <= PC + 4; //PC <= PC + 1000
RET // PC = Mem [PC]; SP <= SP + 4

For instructions: HALT, NOP
i.e... For instructions where ALU has no work to do or process is terminated.

| OP-Code | Don't care |
|---|---|
| 31-26 | 25-0 |

Operations supported:
HALT
NOP