Registers 0-15

$zero	0000
$v0	0001
$PC	0010
$sp	0011
$ra	0100
$t0	0101
$t1	0110
$t2	0111
$t3	1000
$a0	1001
$a1	1010
$a2	1011
$s0	1100
$s1	1101
$s2	1110
$s3	1111

Instructions 0-15

add	0000
addi	0001
sub	0010
lw	0011
sw	0100
and	0101
xor	0110
j	0111
jal	1000
jr	1001
nlj	1010
rj	1011
lins	1100
nor	1101
slt	1110
beq	1111

Register Operation: add, sub, and, xor, nor, slt, beq

0 3	4 7	8 11	12 15
opcode	**rd**	**rt**	**rs**

operation destination(rd), register1(rt), register2(rs)

Normal register arithmetic/logical operations.

Load and Store: lw, sw

operation value(rd), register1(rt), register2(rs)

Register1 is the offset and Register2 base memory address.

Jump Operation: j, jal, jr, nlj, rj

0 3	4 15
opcode	**address**

operation address

Exception for Jump Register (jr) address of register: 4-7.
Supporting all possible (may be needed in our architecture) jump operations.

Load, Increment and Store Operation: lins

0 3	4 7	8 11	12 15
opcode	**rd**	**rt**	**0**

operation destinationRegister(rd), memoryLocation(rt)

Loading a value, incrementing it by one and finally storing it back.

Immediate Operation: addi

0 3	4 7	8 15
opcode	rd	immediate

operation destination(rd), immediate

Immediate adds/loads and stores with a value and a register to be applied to.

Architecture:

Von Neumann Instruction Set Architecture.
Word addressable (which is 2 bytes).
4 bits for registers are used (16 registers).
4 bits for instructions are used (16 instructions).
Special purpose (instruction register) and general purpose registers are used.
Arithmetic, logical and jump operations are included.
Definite use of ALU, control unit and memory in the architecture.
Multiplexers are used to help identify most of the wires to be used.
Sign-extend is used to help compensate shortage of bits in some operations.