1) I-type:

Bits range	Used to store
0-5	Immediate number
6-8	Destination register
9-11	Operand
12-15	Opcode

15 12 11 9 8 6 5 0

Opcode	Operand	Dest. register	Immediate Number

2) R-type:

Bits range	Used to store
0-2	Funct code
3-5	Destination register
6-8	Second Operand
9-11	First Operand
12-15	Opcode

15 12 11 9 8 6 5 3 0

Opcode	1st operand	2nd Operand	Dest. Register	Func code

3) J-type:

Bits range	Used to store
0-11	Immediate number
12-15	Opcode

Opcode	Immediate Number

15 12 11 0

li	li $r1, i 15-12 bits are for opcode (0010) 11-9 bits are empty. 8-6 bits are for $r1. 5-0 bits are for immediate i
add	add $r1, $r2, $r3 15-12 bits are for opcode (0000) 11-9 bits are for $r2. 8-6 bits are for $r3. 5-3 bits are for $r1. 2-0 bits are for funct code (001).
and	and $r1, $r2, $r3 15-12 bits are for opcode (0000) 11-9 bits are for $r2. 8-6 bits are for $r3. 5-3 bits are for $r1. 2-0 bits are for funct code (010).
or	or $r1, $r2, $r3 15-12 bits are for opcode (0000) 11-9 bits are for $r2. 8-6 bits are for $r3. 5-3 bits are for $r1. 2-0 bits are for funct code (000).
neg	neg $r1, $r2 15-12 bits are for opcode (0011) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are empty.
load	load $r1, $r2 15-12 bits are for opcode (0100) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are empty.
store	store $r1, $r2 15-12 bits are for opcode (0101) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are empty.
move	move $r1, $r2 15-12 bits are for opcode (0110) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are empty.
addi	addi $r1, $r2, i 15-12 bits are for opcode (0111) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are for immediate i

andi	andi $r1, $r2, i 15-12 bits are for opcode (1001) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are for immediate i
ori	ori $r1, $r2, i 15-12 bits are for opcode (1000) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are for immediate i.
ble	ble $r1, $r2, i 15-12 bits are for opcode (1010) 11-9 bits are for $r2. 8-6 bits are for $r1. 5-0 bits are for immediate i.
slt	slt $r1, $r2, $r3 15-12 bits are for opcode (0001) 11-9 bits are for $r2. 8-6 bits are for $r3. 5-3 bits are for $r1. 2-0 bits are for funct code (110).
lsl	lsl $r1, $r2, $r3 15-12 bits are for opcode (0000) 11-9 bits are for $r2. 8-6 bits are for $r3. 5-3 bits are for $r1. 2-0 bits are for funct code (100).
lsr	lsr $r1, $r2, $r3 15-12 bits are for opcode (0000) 11-9 bits are for $r2. 8-6 bits are for $r3. 5-3 bits are for $r1. 2-0 bits are for funct code (101).
jump	jump i 15-12 bits are for opcode (1011) 11-0 are for immediate i.
call	call i 15-12 bits are for opcode (1100) 11-0 are for immediate i.
rtn	rtn 15-12 bits are for opcode (1101) 11-0 are empty.

reboot	reboot 15-12 bits are for opcode (1110) 11-0 are empty.
halt	halt 15-12 bits are for opcode (1111) 11-0 are empty.

Test program 1:

instruction	machine code (binary)	machine code (hex)
li $r1, 1	0010 000 000 000001	2001
li $r2, 2	0010 000 001 000010	2042
li $r3, 10	0010 000 010 001010	208A
add $r2, $r1, $r2	0000 000 001 001 001	0049
ble $r2, $r3, -1	1010 001 010 111111	A2BF
slt $r4, $r3, $r2	0001 010 001 011 110	145E
halt	1111 000000000000	F000

Test program 2:

instruction	machine code (binary)	machine code (hex)
li $r1, 3	0010 000 000 000011	2003
li $r2, 5	0010 000 001 000101	2045
andi $r3, $r1, 3	1001 000 010 000011	9083
ori $r4, $r3, 8	1000 010 011 001000	84C8
neg $r5, $r4	0011 100 011 000000	38C0
lsl $r6, $r5, $r1	0000 100 000 101 100	082C
lsr $r7, $r5, $r2	0000 101 001 110 101	0A75
halt	1111 000000000000	F000

Test program 3:

instruction	machine code (binary)	machine code (hex)
li $r1, 6	0010 000 000 000110	2006
li $r2, 5	0010 000 001 000101	2045
and $r3, $r1, $r2	0000 000 001 010 010	0052
li $r8, 0	0010 000 111 000000	21C0
store $r3, $r8	0101 111 010 000000	5E80
or $r4, $r1, $r2	0000 000 001 011 000	0058
li $r8, 1	0010 000 111 000001	21C1
store $r4, $r8	0101 111 011 000000	5EC0
li $r8, 1	0010 000 111 000001	21C1
load $r7, $r8	0100 111 110 000000	4F80
reboot	1110 000000000000	E000
halt	1111 000000000000	F000

Test program 4:

instruction	machine code (binary)	machine code (hex)
li $r1, 6	0010 000 000 000110	2006
li $r2, 4	0010 000 001 000100	2044
call 7	1100 000000000111	C007
move $r4, $r3	0110 010 011 000000	64C0
li $r1, 7	0010 000 000 000111	2007
li $r2, 8	0010 000 001 001000	2048
call 3	1100 000000000011	C003

move $r5, $r3	0110 010 100 000000	6500
jump 3	1011 000000000011	B003
add $r3, $r1, $r2	0000 000 001 010 001	0051
rtn	1101 000000000000	D000
halt	1111 000000000000	F000