Group 17 Report

Encodings

- **1) Arithmetic instructions**

1.1 R-type

OP code	rs(source register)	rt(destination register)	Don't care	func
6 bits	5 bits	5 bits	10 bits	6 bits

Instruction	OP code	func
add rs,rt	000000	000001
Comp rs,rt	000000	000010
and rs,rt	000000	000011
xor rs,rt	000000	000100
diff rs,rt	000000	000101

1.2 I-type

OP code	rs	rs	Immediate value
6 bits	5 bits	5 bits	16 bits

Instruction	OP code
addi rs,imm	000001
compi rs,imm	000010

2) Shift

2.1

OP code	rs(source register)	rt(destination register)	Don’t care	func
6 bits	5 bits	5 bits	10 bits	6 bits

Instruction	OP code	func
shllv rs,rt	000000	000110
shrl rs,rt	000000	000111
shrav rs,rt	000000	001000

2.2

OP code	rs	rs	Immediate value
6 bits	5 bits	5 bits	16 bits

shll rs,sh	000011
shrl rs,sh	000100
shra rs,sh	000101

2) Memory instructions

OP code	rt(destination register)	rs(source register)	Immediate value
6 bits	5 bits	5 bits	16 bits

instructions	OP code

lw rt,imm(rs)	000110
sw rt,imm(rs)	000111

3) Branch Instructions

3.1 Conditional jump

OP code	Don’t care	rs	L
6 bits	5 bits	5 bits	16 bits

instructions	OP code
bltz rs,L	001000
bz rs,L	001001
bnz rs,L	001010

3.2 Unconditional jump

3.2.1 Based on register

OP code	Don’t care	rs(source register)	Don’t care
6 bits	5 bits	5 bits	16 bits

instructions	OP code
br rs	001011

3.2.2 Based on L

OP code	Don’t care	L
6 bits	10 bits	16 bits

instructions	OP code
b, L	001100
bcy, L	001101
bncy, L	001110

3.2.3 Jump and Link (for subroutines)

OP code	Don’t care	L
6 bits	10 bits	16 bits

instructions	OP code
bl L	001111

4) Halt instruction
This instruction stops the program counter from incrementing, and the processor comes to a halt

OP code	Don’t care
111111	26 bits

5) NOP instruction
Does nothing

OP code	Don’t care
111110	26 bits

Note:- We have added halt to stop the program counter and nop for no operations.

2) Controls Signal

2.1)

Inst.	halt	Datapcsel	Regsel	ALUinsel	Regwrite	Memread	Memwrite	Memtoreg	Adsel	uncond.	ALU	Conditional
add rs,rt	0	0	0	00	1	0	0	1	0	0	111	000
comp rs,rt	0	0	0	00	1	0	0	1	0	0	111	000
addi rs,imm	0	0	0	10	1	0	0	1	0	0	000	000
compi rs,imm	0	0	0	10	1	0	0	1	0	0	100	000
and rs,rt	0	0	0	00	1	0	0	1	0	0	111	000
xor rs,rt	0	0	0	00	1	0	0	1	0	0	111	000
shllv rs,rt	0	0	0	00	1	0	0	1	0	0	111	000
shrl rs,rt	0	0	0	00	1	0	0	1	0	0	111	000
shrav rs,rt	0	0	0	00	1	0	0	1	0	0	111	000
shll rs,sh	0	0	0	10	1	0	0	1	0	0	001	000
shrl rs,sh	0	0	0	10	1	0	0	1	0	0	010	000
shra rs,sh	0	0	0	10	1	0	0	1	0	0	011	000
lw rt,imm(rs)	0	0	0	10	1	1	0	0	0	0	000	000
sw rt,imm(rs)	0	0	0	10	0	0	1	0	0	0	000	000
b L	0	0	0	00	0	0	0	0	0	1	000	000
br rs	0	0	0	01	0	0	0	0	0	1	000	000
bltz rs,L	0	0	0	01	0	0	0	0	1	1	000	001
bz rs,L	0	0	0	01	0	0	0	0	0	0	000	010
bnz rs,L	0	0	0	01	0	0	0	0	0	0	000	011
bl L	0	1	1	00	1	0	0	0	0	1	000	000
bly L	0	0	0	00	0	0	0	0	0	0	000	100
bncy L	0	0	0	00	0	0	0	0	0	0	000	101

diff rs,rt	0	0	0	00	1	0	0	1	0	0	111	000

2.2) ALU control

instr	ALU	funct	out
add rs,rt	111	000001	00000
comp rs,rt	111	000010	01100
and rs,rt	111	000011	00001
xor rs,rt	111	000100	000010
diff rs,rt	111	000101	10000
shllv rs,rt	111	000110	00011
shrl rs,rt	111	000111	00111
shrav rs,rt	111	001000	01111
default	000	xxxxxx	00000
shll rs,sh	001	xxxxxx	00011
shrl rs,sh	010	xxxxxx	00111
shra rs,rsh	011	xxxxxx	01111
compi rs,imm	100	xxxxxx	01100
add rs,rt	111	000001	00000

2.3) JUMP CONTROL

Instruction	conditional	flags	out
default	000	xxx	0
bltz rs,L	001	xx1	1
bz rs,L	010	x1x	1
bnz rs,L	011	x0x	1
bcy L	100	1xx	1
bncy L	101	0xx	1
beq rs,rt,L	110	xx1	1

4) Data Path and ALU architecture