

Name: Suraj Salian

ID:2018AAPS0253G

MIPS SINGLE CYCLE ARCHITECTURE

Modules Developed:

1. 32-bit Adder
2. ALU
3. Main Control Unit
4. Data Memory Unit
5. Instruction Memory Unit
6. Parameterized Mux
7. Program Counter Unit
8. Shift Left and Append (For Jump Instructions)
9. Register File
10. Shift Left By 2
11. Sign Extension Unit
12. MIPS Core
13. MIPS Core Testbench
14. Floating Point Adder

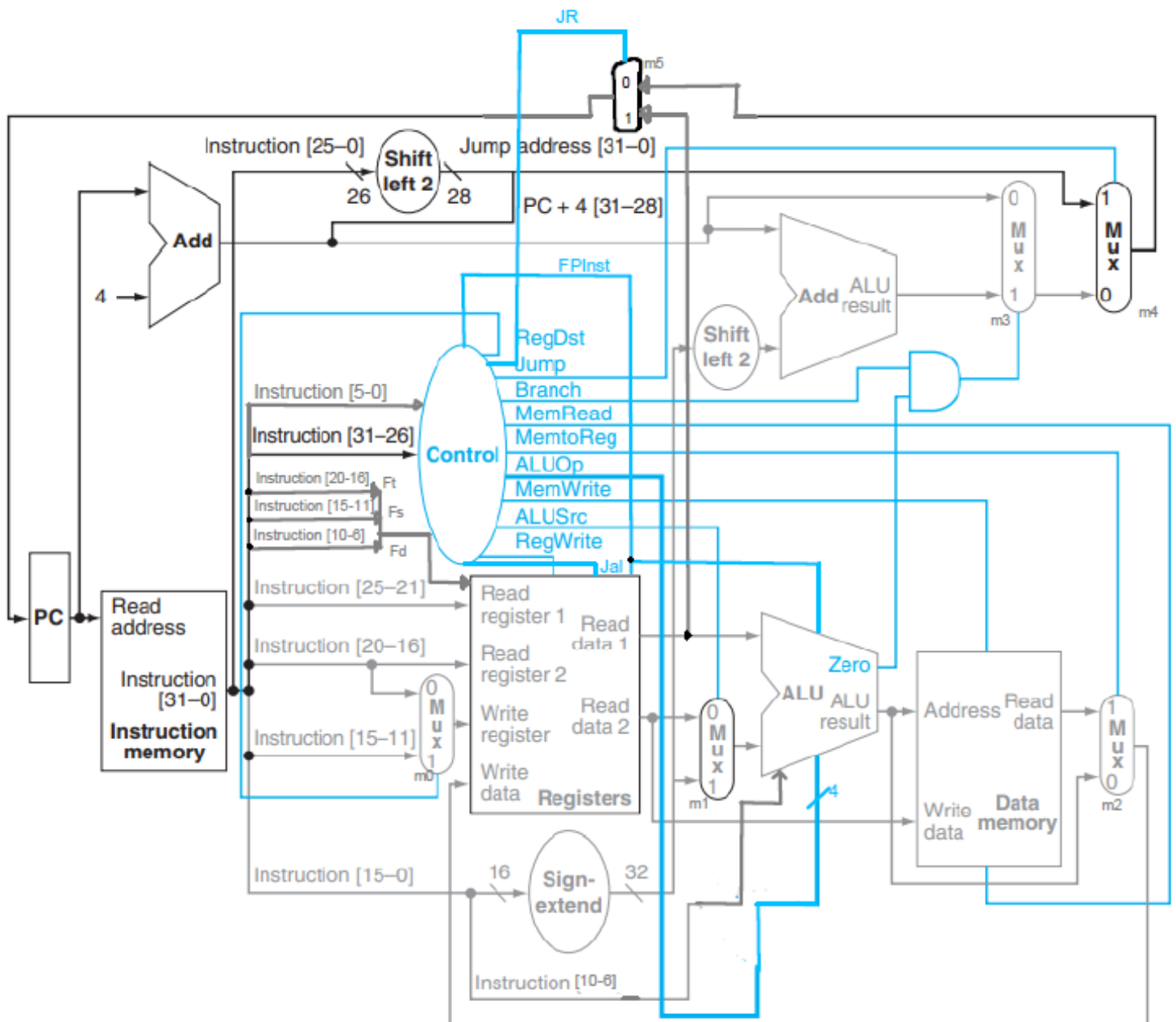
Assumptions:

1. Only 25 out of the complete MIPS ISA. These instructions include-
 - a. NOP
 - b. **R-Type:** *AND, OR, ADD, SUB, SLT, SLL, SRL, NOR, XOR, JR, LW, SW*
 - c. **I-Type:** *ADDI, ANDI, LUI, ORI, SLTI,*
 - d. **Branch:** *BEQ*
 - e. **J-Type:** *J, JAL*
 - f. **Float Type:** *ADD.S, SUB.S, NEG.S*
2. On boot, the Global Reset Signal is ON for one clock cycle to initialize the system (PC = 0).
3. Instruction Memory has 128 Memory Locations each memory location is 8-bit wide. Making the memory byte organized.
4. Data Memory has 512 Memory Locations each memory location is 8-bit wide.
5. Whenever the the Program Counter is out of bound i.e greater than 128 bytes. No Instruction fetching takes place, To go back to initial state Reset should be made ON.
6. Additional Control Signals like JR, FPinst, JAL are introduced in the architecture to support their respective instructions.
7. ***For running the simulation is necessary to place the 'Data.mem', 'REGINP.mem' and 'Instruction.mem' files in the project directory/ Simulation Directory.
8. **The results for register file, data memory are stored in files 'REGDATA.mem' , 'FPREGDATA.mem' and 'MEMDATA.mem'

Assumptions for Floating Point Operations:

1. Inputs to the given model are always in IEEE-754 Format, and inputs should be in normalized form.
i.e, Binary number Of Form: $1. < \dots \dots \dots \text{Fraction Part} \dots \dots > * 2^y$ (y – exponent)
Inputs: A, B (IEEE 754 single precision format- 32 Bits)
2. The input number cannot be NaN, it should be valid number of $\pm \infty$
3. The model outputs result in normalized IEEE-754 format.

Single Cycle Implementation:



Modules:

- 1. 32-bit Adder:** Performs the Add Operation.
First Instance: 'a0' Used for incrementing the Program Counter by 4
Second Instance: 'a1' Used while executing branch instruction to calculate the Program Counter from the relative address.
- 2. ALU:** Performs all important task- R-type Instruction, I-Type Instructions, Float Type Instructions. This unit contains the floating point adder module which is used only for executing 'add.s' and 'sub.s'
- 3. Main Control Unit:** Generates the required control signals from the instruction received from the Instruction Memory. Control signals generated by this unit are- RegDst, Jump, Branch, ALUsrc, MemRead, MemWrite, MemtoReg, ALUOP, RegWrite, Jal, JR, FPinst.
- 4. Data Memory:** This Unit contains 512 bytes RAM for storing data during the operation of CPU. The memory is used for Load word and Store word Instructions. The ROM is byte organized. In this design data is written in "Data.mem" file which is used by the Data memory unit at the time of simulation.
- 5. Instruction Memory:** This unit contains 128 bytes ROM which contains the instruction to be executed by the CPU. In this design instructions are written in "Instruction.mem" file which is used by the Instruction memory unit at the time of simulation.
- 6. Parameterized Mux:** This single parameterized MUX is instantiated according to required width as per the block diagram.
- 7. Program Counter:** Global Clock and Global Reset are given to this unit. On reset the PC is made zero. Else on every clock edge the calculated correct value of Program Counter is written on to the PC.
- 8. Register File:** This unit contains two register files one for normal instruction execution and one register file for floating point instructions. Each Register file contains 32 registers. The Register file is used to read the registers and perform write operation on clock-edge. Depending on the control signal 'FPinst' either the normal Register file or floating-point register file is accessed for instruction execution.
- 9. Shift Left by 2:** This unit is used to shift the immediate data left by 2. (Used by the branch instructions)
- 10. Sign Extension Unit:** This unit sign extends the 16-bit immediate data which is the part of Instruction to 32 bit. (Used by LW, SW, Branch and other Immediate Instructions)
- 11. Shift Left and Append:** This module is used to shift left the 25-bit immediate data by 2. The
- 12. Floating Point Adder:** This unit used for executing the floating-point Addition and Subtract Operation. The floating data supported is single precision type.

13. MIPS Core: The main module of Single Cycle architecture. This unit completes the required connection with different blocks.

14. MIPS Core Testbench: This module provides the Clock of time period 100ns to the MIPS core. This also provides the global clock to the MIPS core.

Control Signal Analysis:

1. **RegDst:** Given as control input to MUX 'm0'. This signal is used to select the destination register from decoded instruction. Inputs to Mux 'm0' are 'Rt' and 'Rd' respectively.
2. **RegWrite:** This control signal is high only when the data is to be written in the destination register.
3. **Jump:** Given as control input to MUX 'm4'. This selects the value of PC for the next clock. This signal is active only during Jump Instruction.
4. **JR:** Given as control input to MUX 'm5'. This selects the value of PC for the next clock. This signal is active only during Jump Register Instruction.
5. **Branch:** Given as control input to MUX 'm3'. This selects the branch address from the immediate data given as instruction. This signal is active only during Branch Instruction.
6. **MemRead:** The given control signal is active only during memory access instruction (Memory Read Instruction) Eg- SW.
7. **MemWrite:** The given control signal is active only during memory write operation. (Eg- SW)
8. **MemoReg:** Given as control input to MUX 'm2'. This selects the data to be written in register file. This signal is active only during memory access Instruction. i.e When this signal is ON the data from memory is written onto the destination register
9. **ALUsrc:** Given as control input to MUX 'm1'. This selects the second input given to ALU. This signal is active only during Immediate, LW, SW Instruction.
10. **ALUOP:** 4-bit control signal given to ALU for given execution of Instructions.
ALUOP for given ALU instructions
 - a. AND 4'b0001
 - b. ADD 4'b0010
 - c. OR 4'b0011
 - d. LUI 4'b0101
 - e. SUB 4'b0110
 - f. SLT 4'b0111
 - g. SLL 4'b1000
 - h. SRL 4'b1001
 - i. FPNEG 4'b1010
 - j. FPSUB 4'b1011
 - k. NOR 4'b1100
 - l. XOR 4'b1110
 - m. FPADD 4'b1111
11. **FPinst:** This control signal is high only if the decoded instruction is Floating Point instruction.
12. **JaL:** The JaL instruction is high only during the Jump and Link instruction. When this control signal is ON the Reg \$31 = PC+4

Instruction Analysis:

1) NOP: Processor remains in the same state.

Instruction: 32'h0

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 0	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 0	ALUOP = 0000

2) AND:

	Opcode	Rs	Rt	Rd	Shamt	Function
and rd, rs, rt	0	rs	rt	rd	0	0x24
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUsrc = 0	ALUOP = 0001

Eg: Hex- 00A63824 // Instructions - AND \$a3 \$a1 \$a2

\$a1 = 0x05 \$a2 = 0x06

\$a3 = ALUResult = 0x5 & 0x6 = 0x4

/MIPS_CORETB/uut/ALUResult	32h00000004	32h00000004
/MIPS_CORETB/uut/ReadData2	32h00000006	32h00000006
/MIPS_CORETB/uut/ReadData1	32h00000005	32h00000005
/MIPS_CORETB/uut/Instruction	32h00a63824	32h00a63824
/MIPS_CORETB/uut/Opcode	6h00	6h00
/MIPS_CORETB/uut/Funct	6h24	6h24
/MIPS_CORETB/uut/ReadReg1	5h05	5h05
/MIPS_CORETB/uut/ReadReg2	5h06	5h06
/MIPS_CORETB/uut/Jal	1h0	
/MIPS_CORETB/uut/Jr	1h0	
/MIPS_CORETB/uut/Jump	1h0	
/MIPS_CORETB/uut/MemRead	1h0	
/MIPS_CORETB/uut/MemWrite	1h0	
/MIPS_CORETB/uut/RegDst	1h1	
/MIPS_CORETB/uut/RegWrite	1h1	
/MIPS_CORETB/uut/ALUop	4h1	4h1
/MIPS_CORETB/uut/ALUsrc	1h0	
/MIPS_CORETB/uut/Branch	1h0	
/MIPS_CORETB/uut/FPinst	1h0	
/MIPS_CORETB/uut/MemtoReg	1h0	
/MIPS_CORETB/uut/ZERO	1h0	

3) OR:

	Opcode	Rs	Rt	Rd	Shamt	Function
or rd, rs, rt	0	rs	rt	rd	0	0x25
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUsrc = 0	ALUOP = 0011

Eg: Hex- 00E84825 // Instructions - OR \$t1 \$a3 \$t0

\$a3 = 0x3 \$t0 = 0xf \$t1 = ALUResult = 0xf

+	/MIPS_CORETB/uut/ALUResult	32h0000000f	32h0000000f
+	/MIPS_CORETB/uut/ReadData2	32h0000000f	32h0000000f
+	/MIPS_CORETB/uut/ReadData1	32h00000003	32h00000003
+	/MIPS_CORETB/uut/Instruction	32h00e84825	32h00e84825
+	/MIPS_CORETB/uut/Opcode	6h00	6h00
+	/MIPS_CORETB/uut/Funct	6h25	6h25
+	/MIPS_CORETB/uut/ReadReg1	5h07	5h07
+	/MIPS_CORETB/uut/ReadReg2	5h08	5h08
	/MIPS_CORETB/uut/Jal	1h0	
	/MIPS_CORETB/uut/Jr	1h0	
	/MIPS_CORETB/uut/Jump	1h0	
	/MIPS_CORETB/uut/MemRead	1h0	
	/MIPS_CORETB/uut/MemWrite	1h0	
	/MIPS_CORETB/uut/RegDst	1h1	
	/MIPS_CORETB/uut/RegWrite	1h1	
+	/MIPS_CORETB/uut/ALUop	4h3	4h3
	/MIPS_CORETB/uut/ALUsrc	1h0	
	/MIPS_CORETB/uut/Branch	1h0	
	/MIPS_CORETB/uut/FPinst	1h0	
	/MIPS_CORETB/uut/MemtoReg	1h0	
	/MIPS_CORETB/uut/ZERO	1h0	

4) ADD:

	Opcode	Rs	Rt	Rd	Shamt	Function
add rd, rs, rt	0	rs	rt	rd	0	0x20
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUsrc = 0	ALUOP = 0010

Eg: Hex- 02138820 // Instructions - ADD \$s1 \$s0 \$s3

\$s0 = 0x10 \$s3 = 0x80000013 \$s1 =ALUResult= 0x10 + 0x80000013= 0x80000023

+	/MIPS_CORETB/uut/ALUResult	32h80000023	32h80000023
+	/MIPS_CORETB/uut/ReadData2	32h80000013	32h80000013
+	/MIPS_CORETB/uut/ReadData1	32h00000010	32h00000010
+	/MIPS_CORETB/uut/Instruction	32h02138820	32h02138820
+	/MIPS_CORETB/uut/Opcode	6h00	6h00
+	/MIPS_CORETB/uut/Funct	6h20	6h20
+	/MIPS_CORETB/uut/ReadReg1	5h10	5h10
+	/MIPS_CORETB/uut/ReadReg2	5h13	5h13
	/MIPS_CORETB/uut/Jal	1h0	
	/MIPS_CORETB/uut/Jr	1h0	
	/MIPS_CORETB/uut/Jump	1h0	
	/MIPS_CORETB/uut/MemRead	1h0	
	/MIPS_CORETB/uut/MemWrite	1h0	
	/MIPS_CORETB/uut/RegDst	1h1	
	/MIPS_CORETB/uut/RegWrite	1h1	
+	/MIPS_CORETB/uut/ALUop	4h2	4h2
	/MIPS_CORETB/uut/ALUsrc	1h0	
	/MIPS_CORETB/uut/Branch	1h0	
	/MIPS_CORETB/uut/FPinst	1h0	
	/MIPS_CORETB/uut/MemtoReg	1h0	
	/MIPS_CORETB/uut/ZERO	1h0	

5) SUB:

	Opcode	Rs	Rt	Rd	Shamt	Function
<code>sub rd, rs, rt</code>	0	rs	rt	rd	0	0x22
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUSrc = 0	ALUOP = 0110

Eg: Hex- 01108822 // Instructions - sub \$s1 \$t0 \$s0

\$t0 = 0x0F \$s0 = 0x10 \$s1 =ALUResult= 0x0F + 0x10 = 0xFFFFFFFFFFFFFFFF

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'hffffffff	32'hffffffff
/MIPS_CORETB/uut/ReadData2	32'h00000010	32'h00000010
/MIPS_CORETB/uut/ReadData1	32'h0000000f	32'h0000000f
/MIPS_CORETB/uut/Instruction	32'h01108822	32'h01108822
/MIPS_CORETB/uut/Opcode	6'h00	6'h00
/MIPS_CORETB/uut/Funct	6'h22	6'h22
/MIPS_CORETB/uut/ReadReg1	5'h08	5'h08
/MIPS_CORETB/uut/ReadReg2	5'h10	5'h10
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h1	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h6	4'h6
/MIPS_CORETB/uut/ALUSrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

6) SLT:

	Opcode	Rs	Rt	Rd	Shamt	Function
<code>slt rd, rs, rt</code>	0	rs	rt	rd	0	0x2a
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUSrc = 0	ALUOP = 0111

Eg: Hex- 0253A02A // Instructions - slt \$s4 \$s2 \$s3

\$s2 = 0x12 \$s3 = 0x80000013 \$s4 =ALUResult= 0x0F < 0x 80000013 ? = 0x00

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'h00000000	32'h00000000
/MIPS_CORETB/uut/ReadData2	32'h80000013	32'h80000013
/MIPS_CORETB/uut/ReadData1	32'h00000012	32'h00000012
/MIPS_CORETB/uut/Instruction	32'h0253a02a	32'h0253a02a
/MIPS_CORETB/uut/Opcode	6'h00	6'h00
/MIPS_CORETB/uut/Funct	6'h2a	6'h2a
/MIPS_CORETB/uut/ReadReg1	5'h12	5'h12
/MIPS_CORETB/uut/ReadReg2	5'h13	5'h13
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h1	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h7	4'h7
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h1	

7) SLL:

	Opcode	Rs	Rt	Rd	Shamt	Function
sll rd, rt, shamt	0	rs	rt	rd	shamt	0
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUsrc = 0	ALUOP = 1000

Eg: Hex- 000C6900 // Instructions - sll \$t5 \$t4 0x4

\$t4 = 0x20 \$t5 = ALUResult = 0x20<<4 = 0x200

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'h00000200	32'h00000200
/MIPS_CORETB/uut/ReadData2	32'h00000020	32'h00000020
/MIPS_CORETB/uut/ReadData1	32'h00000000	32'h00000000
/MIPS_CORETB/uut/Instruction	32'h000c6900	32'h000c6900
/MIPS_CORETB/uut/Opcode	6'h00	6'h00
/MIPS_CORETB/uut/Funct	6'h00	6'h00
/MIPS_CORETB/uut/ReadReg1	5'h00	5'h00
/MIPS_CORETB/uut/ReadReg2	5'h0c	5'h0c
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h1	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h8	4'h8
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

8) SRL:

	Opcode	Rs	Rt	Rd	Shamt	Function
<code>srl rd, rt, shamt</code>	0	rs	rt	rd	shamt	2
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUSrc = 0	ALUOP = 1001

Eg: Hex- 000E7882 // Instructions - srl \$t7 \$t6 0x02
 \$t6 = 0x0C \$t7 = ALUResult = 0x20 >> 2 = 0x03

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'h00000003	32'h00000003
/MIPS_CORETB/uut/ReadData2	32'h0000000c	32'h0000000c
/MIPS_CORETB/uut/ReadData1	32'h00000000	32'h00000000
/MIPS_CORETB/uut/Instruction	32'h000e7882	32'h000e7882
/MIPS_CORETB/uut/Opcode	6'h00	6'h00
/MIPS_CORETB/uut/Funct	6'h02	6'h02
/MIPS_CORETB/uut/ReadReg1	5'h00	5'h00
/MIPS_CORETB/uut/ReadReg2	5'h0e	5'h0e
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h1	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h9	4'h9
/MIPS_CORETB/uut/ALUSrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

9) NOR:

	Opcode	Rs	Rt	Rd	Shamt	Function
<code>nor rd, rs, rt</code>	0	rs	rt	rd	0	0x27
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUSrc = 0	ALUOP = 1100

Eg: Hex- 01495827 // Instructions - NOR \$t3 \$t2 \$t1

\$t2 = 0x F00 \$t1 = 0x F0 \$t3 = ALUResult = 0x FFFFFFFF00F

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'hffff00f	32'hffff00f
/MIPS_CORETB/uut/ReadData2	32'h000000f0	32'h000000f0
/MIPS_CORETB/uut/ReadData1	32'h000000f0	32'h000000f0
/MIPS_CORETB/uut/Instruction	32'h01495827	32'h01495827
/MIPS_CORETB/uut/Opcode	6'h00	6'h00
/MIPS_CORETB/uut/Funct	6'h27	6'h27
/MIPS_CORETB/uut/ReadReg1	5'h0a	5'h0a
/MIPS_CORETB/uut/ReadReg2	5'h09	5'h09
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h1	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'hc	4'hc
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

10) XOR:

	Opcode	Rs	Rt	Rd	Shamt	Function
xor rd, rs, rt	0	rs	rt	rd	0	0x26
	6	5	5	5	5	6

Control Signals Generated:

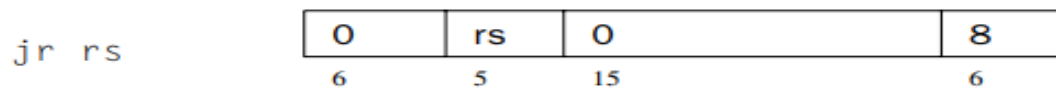
Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 1	FPinst = 0
ALUsrc = 0	ALUOP = 1110

Eg: Hex- 00E84826 // Instructions - XOR \$t1 \$a3 \$t0

\$a3 = 0x3 \$t0 = 0xf \$t1 = ALUResult = 0x0C

	Msgs	
/MIPS_CORETB/uut/rf0/WriteData	32'h0000000c	32'h0000000c
/MIPS_CORETB/uut/ALUResult	32'h0000000c	32'h0000000c
/MIPS_CORETB/uut/ReadData2	32'h0000000f	32'h0000000f
/MIPS_CORETB/uut/ALUInput	32'h0000000f	32'h0000000f
/MIPS_CORETB/uut/ReadData1	32'h00000003	32'h00000003
/MIPS_CORETB/uut/Instruction	32'h00e84826	32'h00e84826
/MIPS_CORETB/uut/Opcode	6'h00	6'h00
/MIPS_CORETB/uut/Funct	6'h26	6'h26
/MIPS_CORETB/uut/ReadReg1	5'h07	5'h07
/MIPS_CORETB/uut/ReadReg2	5'h08	5'h08
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h1	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'he	4'he
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

11) JR:



Control Signals Generated:

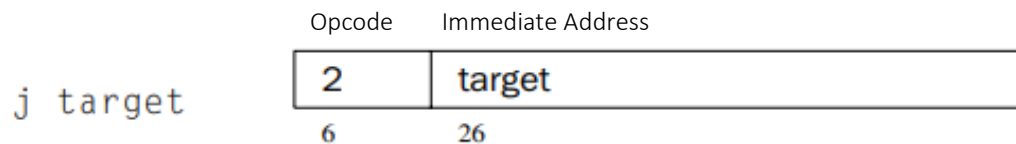
Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 0	Jr = 1
RegDst = 0	FPinst = 0
ALUSrc = 0	ALUOP = 0000

Eg: Hex: 01000008 // Instructions -JR \$t0

\$t0 = 0x0F PCin = 0x0F

	Msgs	
/MIPS_CORETB/uut/pc0/PCin	32'h0000000f	32'h0000000f
/MIPS_CORETB/uut/rf0/WriteData	32'hxxxxxxxx	
/MIPS_CORETB/uut/ALUResult	32'hxxxxxxxx	
/MIPS_CORETB/uut/ReadData2	32'h00000000	32'h00000000
/MIPS_CORETB/uut/ALUInput	32'h00000000	32'h00000000
/MIPS_CORETB/uut/ReadData1	32'h0000000f	32'h0000000f
/MIPS_CORETB/uut/Instruction	32'h01000008	32'h01000008
/MIPS_CORETB/uut/Opcode	6'h00	6'h00
/MIPS_CORETB/uut/Funct	6'h08	6'h08
/MIPS_CORETB/uut/ReadReg1	5'h08	5'h08
/MIPS_CORETB/uut/ReadReg2	5'h00	5'h00
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h1	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h0	
/MIPS_CORETB/uut/ALUOp	4'h0	4'h0
/MIPS_CORETB/uut/ALUSrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'hx	

12) JUMP:



Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 1
MemtoReg = 0	Jal = 0
RegWrite = 0	Jr = 0
RegDst = 0	FPinst = 0
ALUSrc = 0	ALUOP = 0000

Eg: Hex- 0x08000001 // Instructions - J 0x0000001
PCin = {00,(0x01 <<2)} = 0x04

	Msgs	
/MIPS_CORETB/uut/pc0/PCin	32'h00000004	32'h00000004
/MIPS_CORETB/uut/rf0/WriteData	32'hxxxxxxxx	
/MIPS_CORETB/uut/ALUResult	32'hxxxxxxxx	
/MIPS_CORETB/uut/ReadData2	32'h00000000	32'h00000000
/MIPS_CORETB/uut/ALUInput	32'h00000000	32'h00000000
/MIPS_CORETB/uut/ReadData1	32'h00000000	32'h00000000
/MIPS_CORETB/uut/Instruction	32'h08000001	32'h08000001
/MIPS_CORETB/uut/Opcod	6'h02	6'h02
/MIPS_CORETB/uut/Funct	6'h01	6'h01
/MIPS_CORETB/uut/ReadReg1	5'h00	5'h00
/MIPS_CORETB/uut/ReadReg2	5'h00	5'h00
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h1	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h0	
/MIPS_CORETB/uut/ALUop	4'h0	4'h0
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'hx	

13) JAL:

	Opcode	Immediate Address
jal target	3	target
	6	26

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 1
MemtoReg = 0	Jal = 1
RegWrite = 0	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 0	ALUOP = 0000

Eg: Hex- 0x1042001D // Instructions - JAL 0x000001D
PCin = {00,(0x1D <<2)} = 0x74
\$ra = 0x04

	Msgs
/MIPS_CORETB/uut/pc0/PCin	32'h00000074
/MIPS_CORETB/uut/rf0/WriteData	32'h00000000
/MIPS_CORETB/uut/ALUResult	32'h00000000
/MIPS_CORETB/uut/ReadData2	32'h00000000
/MIPS_CORETB/uut/ALUInput	32'h00000000
/MIPS_CORETB/uut/ReadData1	32'h00000000
/MIPS_CORETB/uut/Instruction	32'h00000000
/MIPS_CORETB/uut/Opcod	6'h00
/MIPS_CORETB/uut/Funct	6'h00
/MIPS_CORETB/uut/ReadReg1	5'h00
/MIPS_CORETB/uut/ReadReg2	5'h00
/MIPS_CORETB/uut/Jal	1'h0
/MIPS_CORETB/uut/Jr	1'h0
/MIPS_CORETB/uut/Jump	1'h0
/MIPS_CORETB/uut/MemRead	1'h0
/MIPS_CORETB/uut/MemWrite	1'h0
/MIPS_CORETB/uut/RegDst	1'h1
/MIPS_CORETB/uut/RegWrite	1'h1
/MIPS_CORETB/uut/ALUop	4'h8
/MIPS_CORETB/uut/ALUsrc	1'h0
/MIPS_CORETB/uut/Branch	1'h0
/MIPS_CORETB/uut/FPinst	1'h0
/MIPS_CORETB/uut/MemtoReg	1'h0
/MIPS_CORETB/uut/ZERO	1'h1
/MIPS_CORETB/uut/rf0/REG[31]	32'h00000004

14) ADDI:

	Opcode	Rs	Rt	Immediate
<code>addi rt, rs, imm</code>	8	rs	rt	imm
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 1	ALUOP = 0010

Eg: Hex- 22B70012 // Instructions - ADDI \$s7 \$s5 0x0012

\$s5 = 0x15 \$s7 = ALUResult = 0x15+0x12 = 0x27

	Msgs	
/MIPS_CORETB/uut/ALUResult	32h00000027	32h00000027
/MIPS_CORETB/uut/ReadData2	32h00000017	32h00000017
/MIPS_CORETB/uut/ReadData1	32h00000015	32h00000015
/MIPS_CORETB/uut/Instruction	32h22b70012	32h22b70012
/MIPS_CORETB/uut/Opcode	6h08	6h08
/MIPS_CORETB/uut/Funct	6h12	6h12
/MIPS_CORETB/uut/ReadReg1	5h15	5h15
/MIPS_CORETB/uut/ReadReg2	5h17	5h17
/MIPS_CORETB/uut/Jal	1h0	
/MIPS_CORETB/uut/Jr	1h0	
/MIPS_CORETB/uut/Jump	1h0	
/MIPS_CORETB/uut/MemRead	1h0	
/MIPS_CORETB/uut/MemWrite	1h0	
/MIPS_CORETB/uut/RegDst	1h0	
/MIPS_CORETB/uut/RegWrite	1h1	
/MIPS_CORETB/uut/ALUOp	4h2	4h2
/MIPS_CORETB/uut/ALUsrc	1h1	
/MIPS_CORETB/uut/Branch	1h0	
/MIPS_CORETB/uut/FPinst	1h0	
/MIPS_CORETB/uut/MemtoReg	1h0	
/MIPS_CORETB/uut/ZERO	1h0	

15) XORI:

	Opcode	Rs	Rt	Immediate
<code>xori rt, rs, imm</code>	0xe	rs	rt	Imm
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 1	ALUOP = 1110

Eg: Hex- 0x3AF9001B // Instructions - XORI \$t9 \$s7 0x001B
 \$s7 = 0x17 \$t9 = ALUResult = $0x17 \wedge 0x1B = 0x0C$

	Msgs	
/MIPS_CORETB/uut/rf0/WriteData	32'h0000000c	32'h0000000c
/MIPS_CORETB/uut/ALUResult	32'h0000000c	32'h0000000c
/MIPS_CORETB/uut/ReadData2	32'h00000019	32'h00000019
/MIPS_CORETB/uut/ALUInput	32'h0000001b	32'h0000001b
/MIPS_CORETB/uut/ReadData1	32'h00000017	32'h00000017
/MIPS_CORETB/uut/Instruction	32'h3af9001b	32'h3af9001b
/MIPS_CORETB/uut/Opcode	6'h0e	6'h0e
/MIPS_CORETB/uut/Funct	6'h1b	6'h1b
/MIPS_CORETB/uut/ReadReg1	5'h17	5'h17
/MIPS_CORETB/uut/ReadReg2	5'h19	5'h19
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'he	
/MIPS_CORETB/uut/ALUsrc	1'h1	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

16) ANDI:

	Opcode	Rs	Rt	Immediate
andi rt, rs, imm	0xc	rs	rt	imm
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 1	ALUOP = 0001

Eg: Hex- 32F9001B // Instructions - ANDI \$t9 \$s7 0x001B
 \$s7 = 0x17 \$t9 = ALUResult = $0x17 \& 0x1B = 0x13$

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'h00000013	32'h00000013
/MIPS_CORETB/uut/ReadData2	32'h00000019	32'h00000019
/MIPS_CORETB/uut/ReadData1	32'h00000017	32'h00000017
/MIPS_CORETB/uut/Instruction	32'h32f9001b	32'h32f9001b
/MIPS_CORETB/uut/Opcode	6'h0c	6'h0c
/MIPS_CORETB/uut/Funct	6'h1b	6'h1b
/MIPS_CORETB/uut/ReadReg1	5'h17	5'h17
/MIPS_CORETB/uut/ReadReg2	5'h19	5'h19
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h1	4'h1
/MIPS_CORETB/uut/ALUsrc	1'h1	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

17) BEQ:

<code>beq rs, rt, label</code>	4	rs	rt	Offset
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUSrc = 1	ALUOP = 0001

Eg: Hex- 10670016 // Instructions - BEQ \$v1 \$a3 0x0016
 \$a3 = 0x3 \$v1=0x3 PCin = 0x00+0x4+(0x16 << 2)= 0x5C

	Msgs	
/MIPS_CORETB/uut/pc0/PCin	32'h0000005c	32'h0000005c
/MIPS_CORETB/uut/rf0/WriteData	32'h00000000	32'h00000000
/MIPS_CORETB/uut/ALUResult	32'h00000000	32'h00000000
/MIPS_CORETB/uut/ReadData2	32'h00000003	32'h00000003
/MIPS_CORETB/uut/ALUInput	32'h00000003	32'h00000003
/MIPS_CORETB/uut/ReadData1	32'h00000003	32'h00000003
/MIPS_CORETB/uut/Instruction	32'h10670016	32'h10670016
/MIPS_CORETB/uut/Opcode	6'h04	6'h04
/MIPS_CORETB/uut/Funct	6'h16	6'h16
/MIPS_CORETB/uut/ReadReg1	5'h03	5'h03
/MIPS_CORETB/uut/ReadReg2	5'h07	5'h07
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h0	
/MIPS_CORETB/uut/ALUop	4'h6	4'h6
/MIPS_CORETB/uut/ALUSrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h1	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h1	

18) LUI:

<code>lui rt, imm</code>	0xf	0	rt	imm
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUSrc = 1	ALUOP = 0101

Eg: Hex- 3D7D007F // Instructions - LUI \$sp 0x007F
WriteData = \$sp = 0x007F0000

	Msgs	
/MIPS_CORETB/uut/rf0/WriteData	32'h007f0000	32'h007f0000
/MIPS_CORETB/uut/ALUResult	32'h007f0000	32'h007f0000
/MIPS_CORETB/uut/ReadData2	32'h0000001d	32'h0000001d
/MIPS_CORETB/uut/ALUInput	32'h0000007f	32'h0000007f
/MIPS_CORETB/uut/ReadData1	32'h00000800	32'h00000800
/MIPS_CORETB/uut/Instruction	32'h3d7d007f	32'h3d7d007f
/MIPS_CORETB/uut/Opcode	6'h0f	6'h0f
/MIPS_CORETB/uut/Funct	6'h3f	6'h3f
/MIPS_CORETB/uut/ReadReg1	5'h0b	5'h0b
/MIPS_CORETB/uut/ReadReg2	5'h1d	5'h1d
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h5	4'h5
/MIPS_CORETB/uut/ALUsrc	1'h1	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

19) LW:

lw rt, address	0x23	rs	rt	Offset
	6	5	5	16

Control Signals Generated:

Memread = 1	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 1	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 1	ALUOP = 0010

Eg: Hex- 8C620020 // Instructions - lw \$v0 0x0020 \$v1
\$v1 = 0x3, ; Load Address= ALU Result = 0x3 + 0x20 = 0x23
Data @ 0x23 = 0x900
MemReadData= WriteData: 0x900

	Msgs	
/MIPS_CORETB/uut/rf0/WriteData	32'h00000900	32'h00000900
/MIPS_CORETB/uut/ALUResult	32'h00000023	32'h00000023
/MIPS_CORETB/uut/ReadData2	32'h00000002	32'h00000002
/MIPS_CORETB/uut/ALUInput	32'h00000020	32'h00000020
/MIPS_CORETB/uut/ReadData1	32'h00000003	32'h00000003
/MIPS_CORETB/uut/Instruction	32'h8c620020	32'h8c620020
/MIPS_CORETB/uut/Opcode	6'h23	6'h23
/MIPS_CORETB/uut/Funct	6'h20	6'h20
/MIPS_CORETB/uut/ReadReg1	5'h03	5'h03
/MIPS_CORETB/uut/ReadReg2	5'h02	5'h02
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h1	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h2	4'h2
/MIPS_CORETB/uut/ALUsrc	1'h1	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h1	
/MIPS_CORETB/uut/ZERO	1'h0	

20) SW:

<code>sw rt, address</code>	0x2b	rs	rt	Offset
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 1	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 0	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 1	ALUOP = 0010

Eg: Hex- AFEF0004 Instruction - SW \$t7 0x0004 \$ra

\$ra = 0x1F, \$t7 = 0x60 ; Store Address= ALU Result = 0x1F + 0x04 = 0x23

Read Data2= Write Data: 0x60

/MIPS_CORETB/uut/ALUResult	32'h00000023	32'h00000023	
/MIPS_CORETB/uut/ReadData2	32'h00000060	32'h00000060	
/MIPS_CORETB/uut/ReadData1	32'h0000001f	32'h0000001f	
/MIPS_CORETB/uut/Instruction	32'hafef0004	32'hafef0004	
/MIPS_CORETB/uut/Opcode	6'h2b	6'h2b	
/MIPS_CORETB/uut/Funct	6'h04	6'h04	
/MIPS_CORETB/uut/ReadReg1	5'h1f	5'h1f	
/MIPS_CORETB/uut/ReadReg2	5'h0f	5'h0f	
/MIPS_CORETB/uut/Jal	1'h0		
/MIPS_CORETB/uut/Jr	1'h0		
/MIPS_CORETB/uut/Jump	1'h0		
/MIPS_CORETB/uut/MemRead	1'h0		
/MIPS_CORETB/uut/MemWrite	1'h1		
/MIPS_CORETB/uut/RegDst	1'h0		
/MIPS_CORETB/uut/RegWrite	1'h0		
/MIPS_CORETB/uut/ALUop	4'h2	4'h2	
/MIPS_CORETB/uut/ALUsrc	1'h1		
/MIPS_CORETB/uut/Branch	1'h0		
/MIPS_CORETB/uut/FPinst	1'h0		
/MIPS_CORETB/uut/MemtoReg	1'h0		
/MIPS_CORETB/uut/ZERO	1'h0		

21) ORI:

<code>ori rt, rs, imm</code>	0xd	rs	rt	imm
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 1	ALUOP = 0011

Eg: Hex- 375B0019 // Instructions - ORI \$k1 \$k0 0x0019
 \$k0 = 0x1A \$k1 = ALUResult = 0x1A | 0x19 = 0x1B

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'h0000001b	32'h0000001b
/MIPS_CORETB/uut/ReadData2	32'h0000001b	32'h0000001b
/MIPS_CORETB/uut/ReadData1	32'h0000001a	32'h0000001a
/MIPS_CORETB/uut/Instruction	32'h375b0019	32'h375b0019
/MIPS_CORETB/uut/Opcode	6'h0d	6'h0d
/MIPS_CORETB/uut/Funct	6'h19	6'h19
/MIPS_CORETB/uut/ReadReg1	5'h1a	5'h1a
/MIPS_CORETB/uut/ReadReg2	5'h1b	5'h1b
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h3	4'h3
/MIPS_CORETB/uut/ALUsrc	1'h1	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

22) SLTI:

slti rt, rs, imm	Oxa	rs	rt	imm
	6	5	5	16

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 0
ALUsrc = 1	ALUOP = 0111

Eg: Hex- 2910001D // Instructions - slti \$s0 \$t0 0x001d
 \$t0 = 0xF \$s0 = ALUResult = 0xF < 0x1d ?= 0x01

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'h00000001	32'h00000001
/MIPS_CORETB/uut/ReadData2	32'h00000010	32'h00000010
/MIPS_CORETB/uut/ALUInput	32'h0000001d	32'h0000001d
/MIPS_CORETB/uut/ReadData1	32'h0000000f	32'h0000000f
/MIPS_CORETB/uut/Instruction	32'h2910001d	32'h2910001d
/MIPS_CORETB/uut/Opcode	6'h0a	6'h0a
/MIPS_CORETB/uut/Funct	6'h1d	6'h1d
/MIPS_CORETB/uut/ReadReg1	5'h08	5'h08
/MIPS_CORETB/uut/ReadReg2	5'h10	5'h10
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUOp	4'h7	4'h7
/MIPS_CORETB/uut/ALUsrc	1'h1	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h0	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

23) ADD.S:

<code>add.s fd, fs, ft</code>	0x11	0x10	ft	fs	fd	0
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 1
ALUsrc = 0	ALUOP = 1111

Eg: Hex- 46010080 // Instructions -add.s \$f2 \$f0 \$f1

\$f0 = 0x 40300000 \$f1 = 0x 3FC00000 \$f2=ALUResult= 0x40880000

	Msgs	
/MIPS_CORETB/uut/ALUResult	32'h40880000	32'h40880000
/MIPS_CORETB/uut/ReadData2	32'h3fc00000	32'h3fc00000
/MIPS_CORETB/uut/ReadData1	32'h40300000	32'h40300000
/MIPS_CORETB/uut/Instruction	32'h46010080	32'h46010080
/MIPS_CORETB/uut/Opcode	6'h11	6'h11
/MIPS_CORETB/uut/Funct	6'h00	6'h00
/MIPS_CORETB/uut/ReadReg1	5'h10	5'h10
/MIPS_CORETB/uut/ReadReg2	5'h01	5'h01
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
/MIPS_CORETB/uut/ALUop	4'hf	4'hf
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h1	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

24) SUB.S:

<code>sub.s fd, fs, ft</code>	0x11	0x10	ft	fs	fd	1
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 1
ALUsrc = 0	ALUOP = 1011

Eg: Hex- 46010081 // Instructions -sub.s \$f2 \$f0 \$f1

\$f0 = 0x40300000 \$f1 = 0x3FC00000 \$f2=ALUResult= 0x3fa00000

	Msgs	
+ /MIPS_CORETB/uut/ALUResult	32'h3fa00000	32'h3fa00000
+ /MIPS_CORETB/uut/ReadData2	32'h3fc00000	32'h3fc00000
+ /MIPS_CORETB/uut/ReadData1	32'h40300000	32'h40300000
+ /MIPS_CORETB/uut/Instruction	32'h46010081	32'h46010081
+ /MIPS_CORETB/uut/Opcode	6'h11	6'h11
+ /MIPS_CORETB/uut/Funct	6'h01	6'h01
+ /MIPS_CORETB/uut/ReadReg1	5'h10	5'h10
+ /MIPS_CORETB/uut/ReadReg2	5'h01	5'h01
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
+ /MIPS_CORETB/uut/ALUOp	4'hb	4'hb
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h1	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	

25) NEG.S:

neg.s fd, fs	0x11	0x10	0	fs	fd	7
	6	5	5	5	5	6

Control Signals Generated:

Memread = 0	Branch = 0
Memwrite = 0	Jump = 0
MemtoReg = 0	Jal = 0
RegWrite = 1	Jr = 0
RegDst = 0	FPinst = 1
ALUsrc = 0	ALUOP = 1011

Eg: Hex- 46000087// Instructions - neg.s \$f2 \$f0

\$f0 = 0x40300000 \$f2=ALUResult= 0xc0300000

	Msgs	
+ /MIPS_CORETB/uut/ALUResult	32'hc0300000	32'hc0300000
+ /MIPS_CORETB/uut/ReadData2	32'h40300000	32'h40300000
+ /MIPS_CORETB/uut/ReadData1	32'h40300000	32'h40300000
+ /MIPS_CORETB/uut/Instruction	32'h46000087	32'h46000087
+ /MIPS_CORETB/uut/Opcode	6'h11	6'h11
+ /MIPS_CORETB/uut/Funct	6'h07	6'h07
+ /MIPS_CORETB/uut/ReadReg1	5'h10	5'h10
+ /MIPS_CORETB/uut/ReadReg2	5'h00	5'h00
/MIPS_CORETB/uut/Jal	1'h0	
/MIPS_CORETB/uut/Jr	1'h0	
/MIPS_CORETB/uut/Jump	1'h0	
/MIPS_CORETB/uut/MemRead	1'h0	
/MIPS_CORETB/uut/MemWrite	1'h0	
/MIPS_CORETB/uut/RegDst	1'h0	
/MIPS_CORETB/uut/RegWrite	1'h1	
+ /MIPS_CORETB/uut/ALUOp	4'ha	4'ha
/MIPS_CORETB/uut/ALUsrc	1'h0	
/MIPS_CORETB/uut/Branch	1'h0	
/MIPS_CORETB/uut/FPinst	1'h1	
/MIPS_CORETB/uut/MemtoReg	1'h0	
/MIPS_CORETB/uut/ZERO	1'h0	