My program can compile on verilog but in modelsim, I can't get the results of the instruction. So that,I will show the testbench results of each module.s

1- Control Unit Module

```
# time =  0, opcode =100011, RegWrite=1, AluSrc=1, RegDst=0, MemToReg=1, MemRead=1, MemWrite=0, Bra
nch=0, AluOp1=0, AluOp0=0
# time = 20, opcode =000100, RegWrite=0, AluSrc=0, RegDst=0, MemToReg=0, MemRead=0, MemWrite=0, Bra
nch=1, AluOp1=0, AluOp0=1
```

First instruction opcode is load word and second instruction opcode is branch equal. These two instructions' output signals are true.

2- ALU Control

```
# time =  0, AluOp =10, func=100000, AluCtr=010
# time = 20, AluOp =10, func=100010, AluCtr=110
```

Two instructions' AluOp is "10" and these are R type instructions. First instruction's function part says that this instruction will make add operation at the ALU. So that this instruction's AluCtr is 010.

Second instruction's function part says that this instruction will make subtract operation at the ALU. So that this instruction's AluCtr is 110.

3- ALU Result

```
# time =  0, aluctr= 000, a= 00000000000000000000000011111111, b= 00000000000000001111111111111111,
 res= 00000000000000000000000011111111, c_out= 0, v= 0, z= 0
# time = 20, aluctr= 010, a= 00000000000000000000000000000011, b= 00000000000000000000000000000111,
 res= 00000000000000000000000000001010, c_out= 0, v= 0, z= 0
```

First instruction's AluCtr value is 000. This value says that ALU will make a and operation. So that, ALU make a and operation in the first instruction.

Second instruction's AluCtr value is 010. This value says that ALU will make a add operation. So that, ALU make a add operation in the second instruction.

4-Alu Src Mux

```
# time =  0, reg_data =00000000000000001111111111111111, sign_ext_data =11111111111111110000000000
00000, dest_data =00000000000000001111111111111111, AluSrc =0
# time = 20, reg_data =00000000000000000000000000000000, sign_ext_data =11111111111111111111111111
11111, dest_data =11111111111111111111111111111111, AluSrc =1
```

This mux module determines that ALU will take a content of register (rt) or sign extension of immediate part. If AluSrc signal is 0, ALU will take a content of register (rt). If AluSrc is 1, ALU will take sign extension of immediate part. At the below image, you can see the example of this.

5-Reg Dst Mux

```
# time =  0, rt_add =00011, rd_add =00011, dest_add =00011, RegDst =0
# time = 20, rt_add =00000, rd_add =11111, dest_add =11111, RegDst =1
```

This mux module determines that data will written to rt address of instruction or rd address of instruction. If RegDst signal is 0, data will written to rt address of instruction. If RegDst signal is 1, data will written to rd address of instruction. At the below image, you can see the example of this.

6- Sign Extend

```
# time =  0, a =0000000000001111, b=00000000000000000000000000001111
# time = 20, a =1111111111111000, b=11111111111111111111111111111000
```

This module completes the 16 bit input to 32 bits according to the sign bit of the 16 bit input. In our program, immediate part of instruction will enter this module in order to get sign extended immediate. This module is used by load word, store word etc… instructions.

7- Shift Left 2

```
# time =  0, a =11111111111111111111111111111111, b=11111111111111111111111111111100
# time = 20, a =00000000000000000000000000000011, b=00000000000000000000000000001100
```

This module applies 2 bit shifting to its input. In our program, we apply 2 bit shifting to sign extended immediate. So that, this module's input will be sign extended immediate.

Mehmet Acar

1801042095