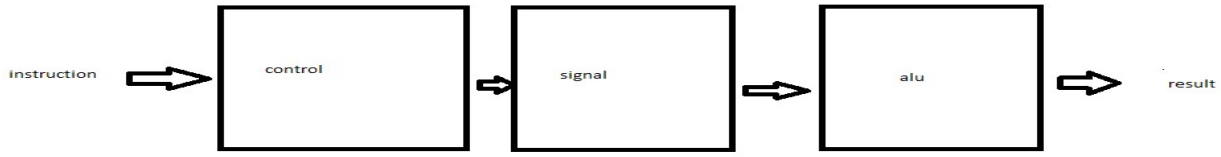
CSE 331 Computer Organization
Homework 4

Hamza Yoğurtcuoğlu - 171044086

4 January , 2019

- Top Level Design File: mips32_single_cycle
- Testbench File: mips32_single_cycle_testbench
- registers.mem for initialization of registers
- instruction.mem for instruction memory initialization
- data.mem for data memory initialization
-nextPC.v is calculate next instruction location in instruction.mem
-zero_or_sing.v is putting 0 or 1 immediate according to instruction



What I am trying to do is shown in the figure above.

1.1 Life cycle of 1 instruction

After instruction is stored in instruction, I sent it to the control unit. There it is decided what type of instruction it is. Signals follow it. Then alu executes according to the previous step.

2. Methods

2.1 Step 1 Control

In this module, According to the opcode and function cod, I find out what type of instructions I am dealing with

2.2 Step 2 Signal

According to the instruction types , I set the signals in this module

2.3 Step 3 ALU

Above 2 modules will be considered when alu calculations are happening

NOT : EACH INSTRUCTION HAS TWO CLOCK . 1 AND 0 !

Test Result - 1

time = 0, instruction =00000001110011010000100000100000
# rs = 01110 ,rt = 01101 , rd=00001
result=00000000000000000000000000011011
→ **RS REGİSTER ADD RT REGISTER ,STORE RD**
# time = 50, instruction =00000000001000010001000000100001
# rs = 00001 ,rt = 00001 , rd=00010
result=00000000000000000000000000110100
→ **RS REGİSTER ADDI RT REGISTER , STORE RD**
# time = 150, instruction =00110000010001011111111111111111
# rs = 00001 ,rt = 00010 , rd=11111
result=00000000000000000000000000011010
→ **RS REGİSTER ANDI IMMEDIATE , STORE RT**

# time = 250, instruction =00110100010000110000000000000001
# rs = 00010 ,rt = 00011 , rd=00000
result=00000000000000000000000000011001
→ **RS REGİSTER ORI IMMEDIATE , STORE RT**
# time = 350, instruction =10101100000000110000000000000001
# rs = 00000 ,rt = 00011 , rd=00000 //ZERO REGISTER RS!
result=00000000000000000000000000000001
→ **RS REGİSTER ANDI IMMEDIATE , STORE RT**
# time = 450, instruction =01011100001000010000000000000001
# rs = 00001 ,rt = 00001 , rd=00000
result=00000000000000000000000000011011
→ **MEMORY(RS REGİSTER + IMMEDIATE) , STORE RT**
time = 550, instruction =00100101110011010000000000000011

# rs = 01110 ,rt = 01101 , rd=00000
result=00000000000000000000000000001111
→ RT CONTENT , STORE RS REGİSTER + IMMEDIATE
# time = 650, instruction =00010001110011010000000000000011
# rs = 01110 ,rt = 01101 , rd=00000
result=11111111111111111111111110111
RS REGİSTER == RT REGISTER BRANCH NOT EQUAL
# time = 750, instruction =00001000000000000000000000001000
# rs = 00000 ,rt = 00000 , rd=00000
result=00000000000000000000000000000011
JUMP TO LAST INSTRUCTION
# time = 850, instruction =00000000010000010001100000100010
# rs = 00010 ,rt = 00001 , rd=00011
result=00000000000000000000000000000001
LAST SUB INSTRUCTION BECAUSE JUMP TO HERE.

Test Result – 2

TEST INSTRUCTION

add	00000001110011010000100000100000
addiu	00100100001000010001000000100001
andi	00110000001000101111111111111111
ori	00110100010000110000000000000001
sw	10101100000000110000000000000001
sw	10101100000000110000000000000001
lw	10001100000111110000000000000001
beq	00010000001000001111111111111101
j	00001000000000000000000000000000
...	00000001110011010000100000100000
...	00000000001000010001000000100001

There is 10 instruction in there
This program will continue forever if we get clock.

```
# time =  0, instruction =00000001110011010000100000100000
# rs = 01110 ,rt = 01101 , rd=00001 result=00000000000000000000000000011011 clock=1
# time = 50, instruction =00000001110011010000100000100000
# rs = 01110 ,rt = 01101 , rd=00001 result=00000000000000000000000000011011 clock=0
# time = 100, instruction =00000001110011010000100000100000
# rs = 01110 ,rt = 01101 , rd=00001 result=00000000000000000000000000011011 clock=1
# time = 150, instruction =00100100001000010001000000100001
# rs = 00001 ,rt = 00001 , rd=00010 result=00000000000000000001000000111100 clock=0
# time = 200, instruction =00100100001000010001000000100001
# rs = 00001 ,rt = 00001 , rd=00010 result=00000000000000000001000000111100 clock=1
# time = 250, instruction =00110000001000101111111111111111
# rs = 00001 ,rt = 00010 , rd=11111 result=00000000000000000000000000011011 clock=0
# time = 300, instruction =00110000001000101111111111111111
# rs = 00001 ,rt = 00010 , rd=11111 result=00000000000000000000000000011011 clock=1
# time = 350, instruction =00110100010000110000000000000001
# rs = 00010 ,rt = 00011 , rd=00000 result=00000000000000000000000000011011 clock=0
# time = 400, instruction =00110100010000110000000000000001
# rs = 00010 ,rt = 00011 , rd=00000 result=00000000000000000000000000011011 clock=1
# time = 450, instruction =10101100000000110000000000000001
# rs = 00000 ,rt = 00011 , rd=00000 result=00000000000000000000000000000001 clock=0


# time = 500, instruction =10101100000000110000000000000001
# rs = 00000 ,rt = 00011 , rd=00000 result=00000000000000000000000000000001 clock=1
# time = 550, instruction =10101100000000110000000000000001
# rs = 00000 ,rt = 00011 , rd=00000 result=00000000000000000000000000000001 clock=0
# time = 600, instruction =10101100000000110000000000000001
# rs = 00000 ,rt = 00011 , rd=00000 result=00000000000000000000000000000001 clock=1
# time = 650, instruction =10001100000111110000000000000001
# rs = 00000 ,rt = 11111 , rd=00000 result=00000000000000000000000000000001 clock=0
# time = 700, instruction =10001100000111110000000000000001
# rs = 00000 ,rt = 11111 , rd=00000 result=00000000000000000000000000000001 clock=1
# time = 750, instruction =00010000001000001111111111111101
# rs = 00001 ,rt = 00000 , rd=11111 result=00000000000000000000000000011110 clock=0
# time = 800, instruction =00010000001000001111111111111101
# rs = 00001 ,rt = 00000 , rd=11111 result=00000000000000000000000000011110 clock=1
# time = 850, instruction =00001000000000000000000000000000
# rs = 00000 ,rt = 00000 , rd=00000 result=00000000000000000000000000000000 clock=0
# time = 900, instruction =00001000000000000000000000000000
# rs = 00000 ,rt = 00000 , rd=00000 result=00000000000000000000000000000000 clock=1
# time = 950, instruction =00000000001000010001000000100001
# rs = 00001 ,rt = 00001 , rd=00010 result=00000000000000000000000000110110 clock=0
```