**CSE3038**

**PROJECT 2 REPORT**

**Group Members**

**Muhammet ŞERAMET – 150115069**

**Tolga SAVAŞ - 150115004**

In this project we implemented instructions 1, 7, 11, 15, 21 and 24. We have implemented I type and J type instructions in control unit. However, we implemented R type instructions in ALU control. Also we opened a new control unit name as brjmpcont. We used it to control brz, bmz, jmadd and jrsal instructions. Also we defined n and z in status register in processor unit. We didn’t defined v bit because our instructions does not use v bit.

**BMZ Instruction:**

It is immediate type; so we implemented it’s opcode in control unit. Also we made AluSrc signal is 1 and x for others. Also we made brjmpcont signal 3'b001 and controlled it in brjmpcont unit with status z = 1. Then we send the new address to the processor file.

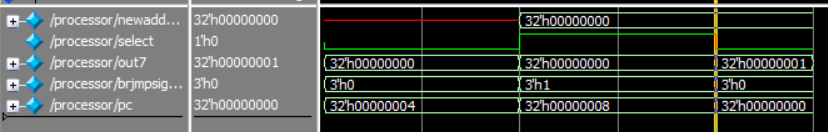
Test Case:

sub $s1, $s1, $s1

bmz 1($s1)

sw $t0, 4($s0)

Sample Output:



**BRZ Instruction:**

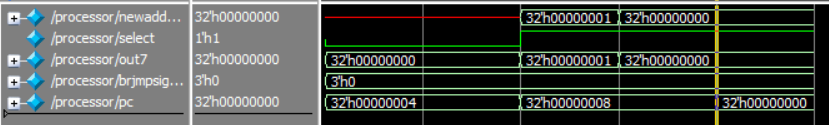
It is R-type instruction so we implemented it’s opcode in Alu Control unit. Then we specified that it is brz instruction with rtypeout signal and send it to brjmpcont unit. We controlled it with status z = 1 and rtypeout signal in here. Then we send the new address to the processor file.

Test Case:

sub $s2, $s1, $s1

brz $s2

Sample Output:



**JMADD Instruction:**

It is R-type instruction so we implemented it’s opcode in ALU Control unit. Then we specified that it is brz instruction with rtypeout signal and send it to brjmpcont unit. We controlled it with rtypeout signal in here. The main thing is this instruction links the address. We did it with combining this instruction with jump operation. (mux8 in processor)

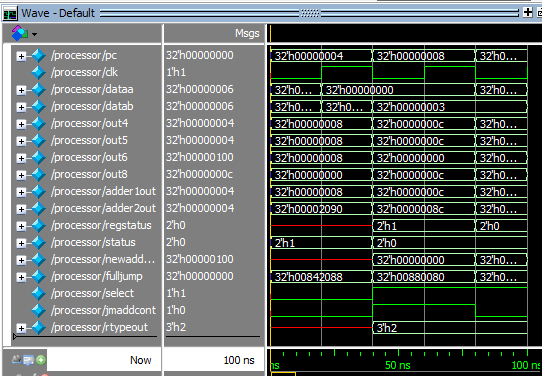
Test Case:

sub $s1, $s1, $s1

jmadd $s1, $s2

sw $t0, 4($s0)

Sample Output:



**BN Instruction:**

It is Jump type instruction so we implemented it’s opcode in control unit. This instruction branches to pseudo-direct address (formed as j does). So we didn’t implement its branch operation in brjmpcont. In processor, we put bnwire into AND gate in MIPS datapath (From branch signal). And we control status signal (n=1) in here to.

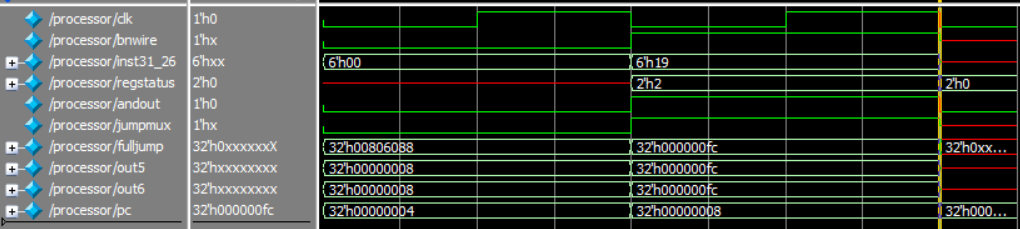
Test Case:

sub $s3, $ss1, $s0

bn 0x0000011

sw $s0, 4($s0)

Sample Output:



**JRSAL Instruction:**

It is immediate type; so we implemented it’s opcode in control unit. Also we made brjmpcont signal 3'b100 and controlled it in brjmpcont unit. Then we assign new addressout to use it in processor.

In the processor, we used jrsaljmaddcont wire to distinguish them. After that we used it as selector for multiplexer9 which links the register address to pc+4.

Test Case:

sub $s3,$s1,$s0

jrsal $s7

Sample Output:

**SLLV Instruction:**

It is R-type instruction so we implemented it’s opcode in ALU Control unit. After that send a shift signal to ALU from ALU Control unit. If this signal is caught, then shift operation is available. We did it with simple << operator in ALU.

Test Case:

sllv $s0, $s1, $s2

sw $s0, 4($s0)

Sample Output:

