**CSE3038 COMPUTER ORGANIZATION PROJECT II REPORT**

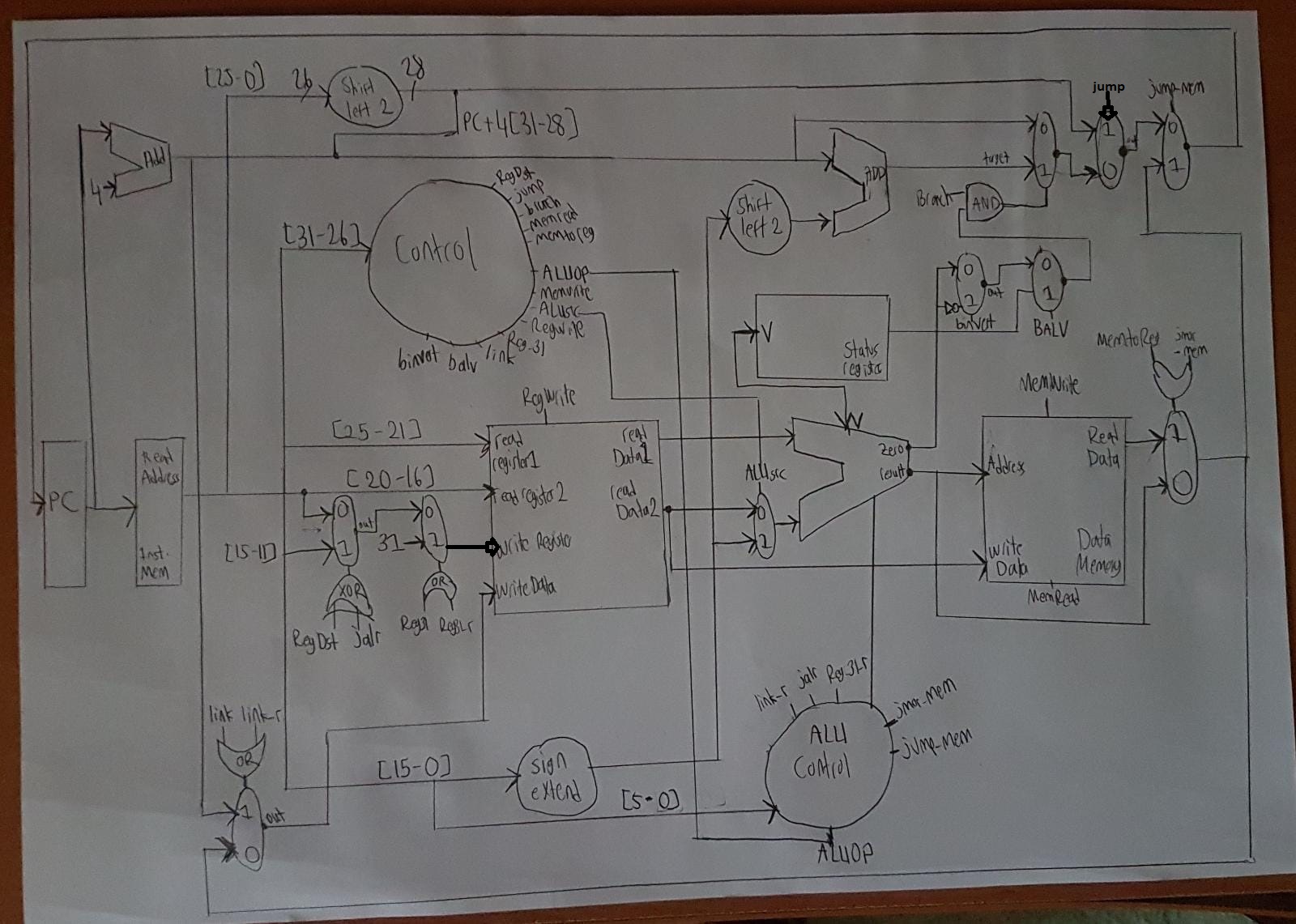
****

**KADİR HIZARCI – 150116004**

**BUĞRA AKDENİZ – 150116072**

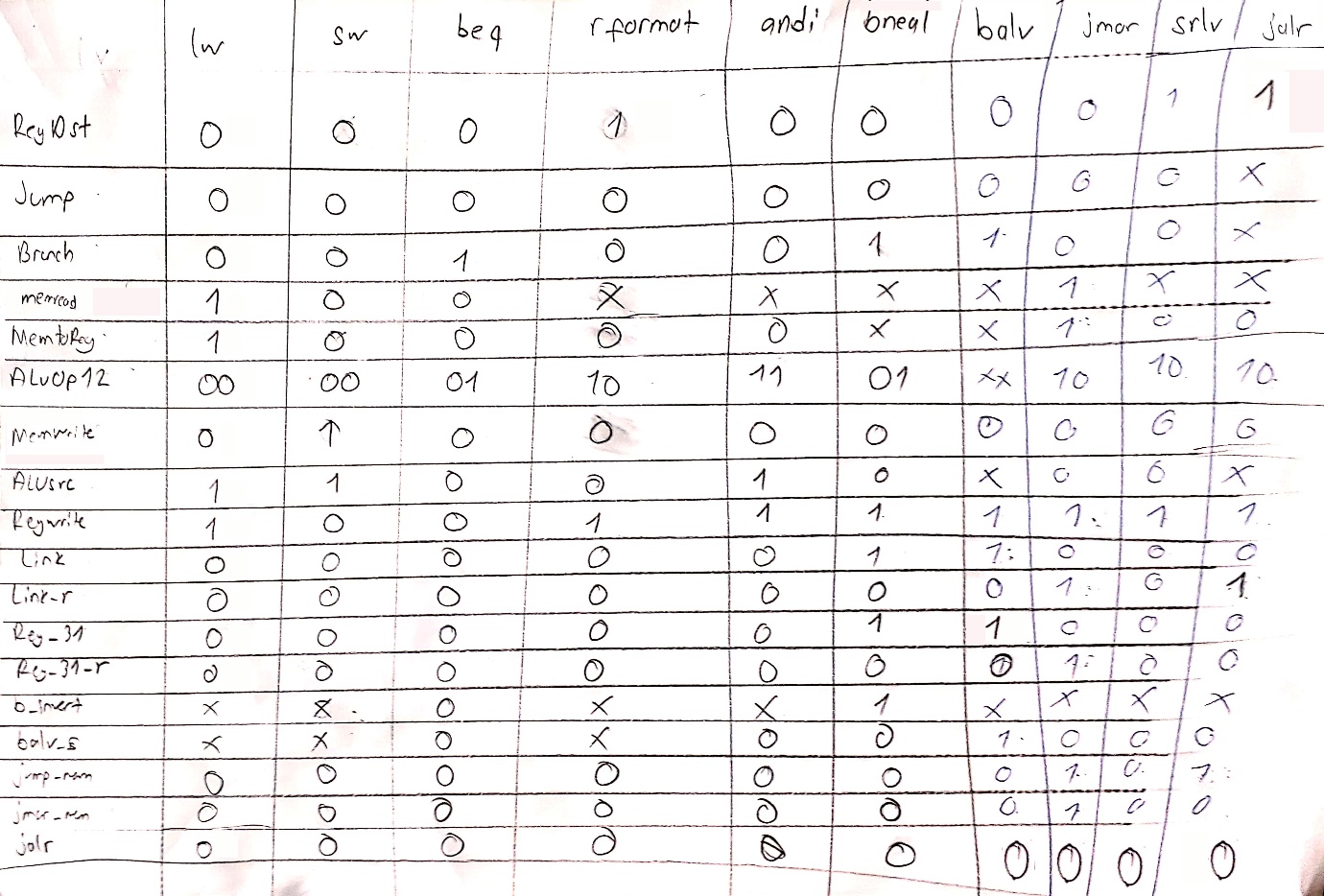
1. [**Datapath Design**](#data)

1. **[Verilog Implementation Overview](#book)**
2. **[Test Runs](#test)**
3. **Datapath Design**

Below is a drawing of our datapath design supporting new instructions ANDI, BNEAL, JMOR, JALR, BALV and SRLV as well as the default ones.

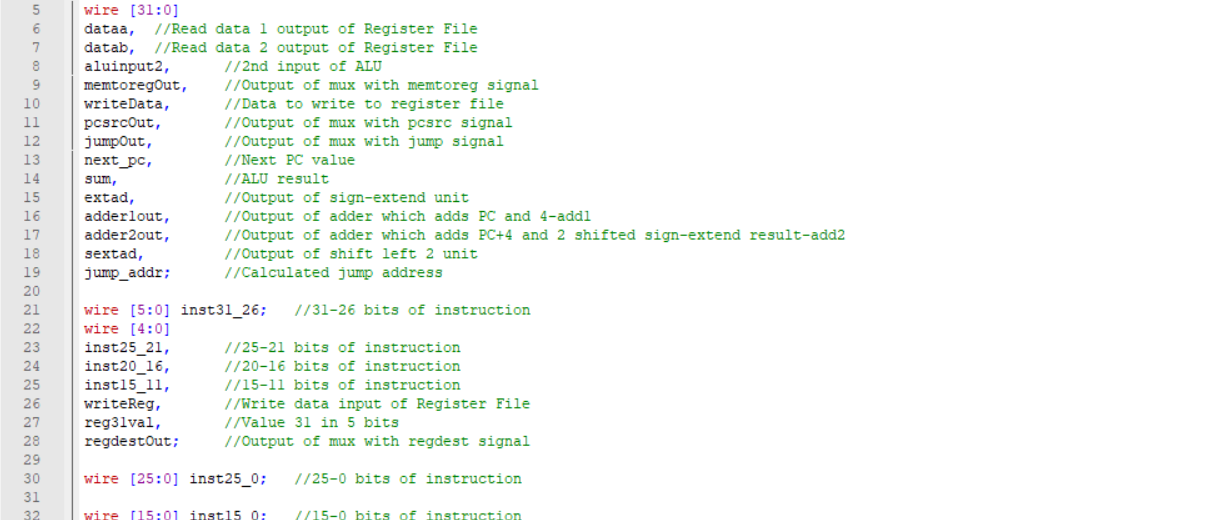
* There is a total of 8 new multiplexers.
* There is a total if 9 new signals, 5 of them is created from ALU whilst other 4 are created from Control Unit.
* There are also newly designed components, such as Status Register and Sign Extender for j type instructions.

Below is another image showing new datapath signals for each instruction.



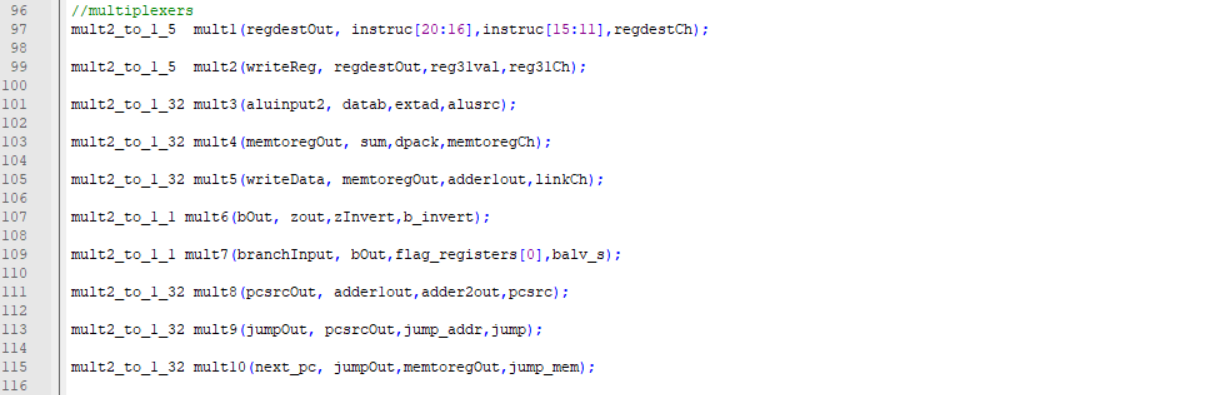
* There are no significant changes to default instruction’s signals.
* Link signal is 1 when PC address is needed to write to register.
* Link\_r does the same, but is created from ALU for r-type instructions.
* Reg\_31 is 1 when register to be written is register 31.
* Reg\_31\_r does the same, but is created from ALU for r-type instructions.
* B\_invert is 1 when zero output of ALU is needed to be inverted.
* Balv is 1 when status register is needed to decide whether to branch or not.
* Jump\_mem is 1 when jump address is need to be received from data memory or from ALU.
* Jmor\_mem is 1 when data memory is needed instead of ALU result.
* Jalr is 1 when register destination needs to be changed.

1. **Verilog Implementation Overview**

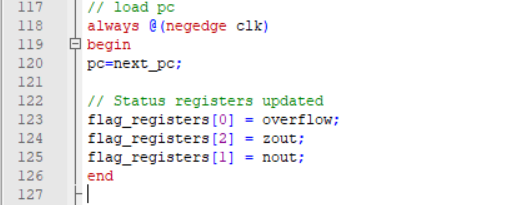


**processor.v**

* Above image shows mostly 1-bit variables used to control muxes.

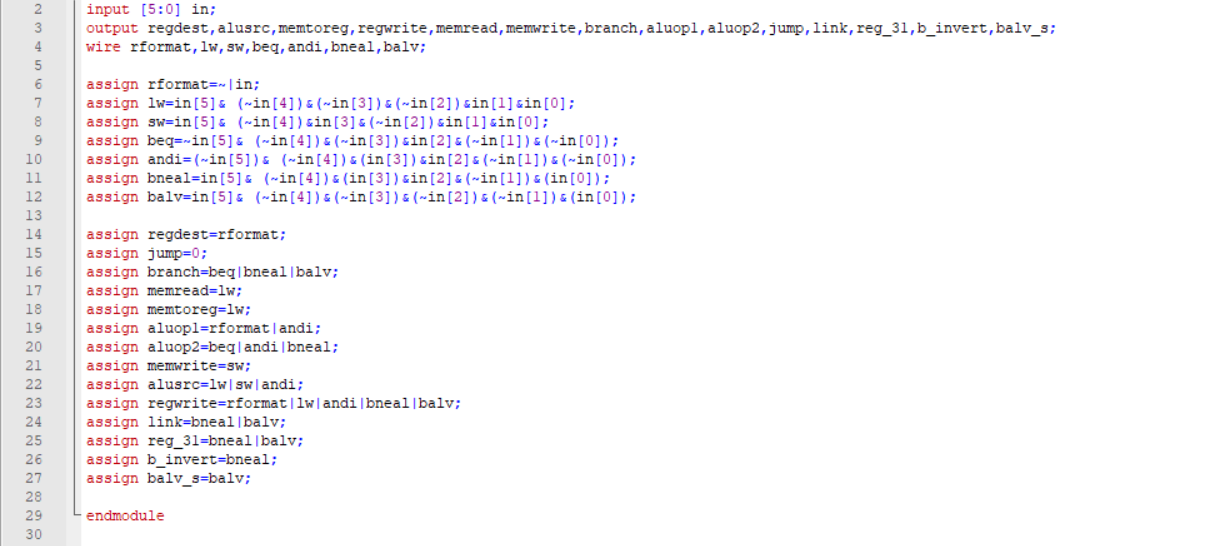


* Above image shows new and default muxes updated with their new signals and variables as inputs.



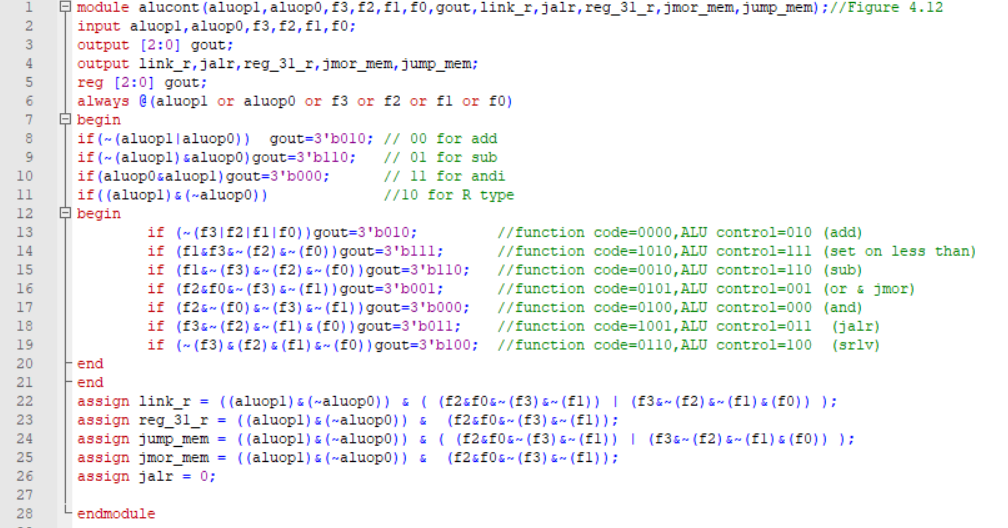
* Status Register is updated on each negative edge of the clock.

**control.v**

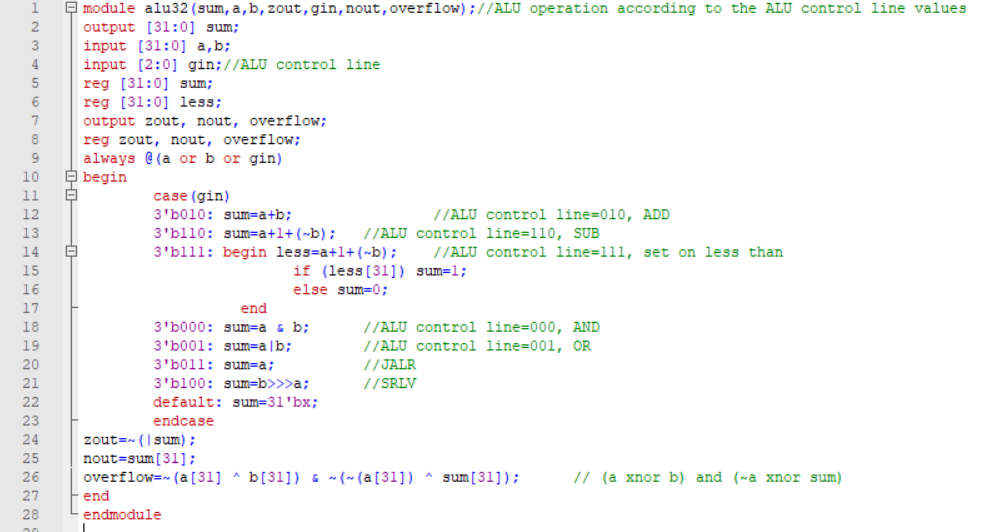


* New instructions and and new signals are added to the control unit.

**alucont.v**



* ALU control is modified to create r-type specific signals from the received funct code.

****

**alu32.v**

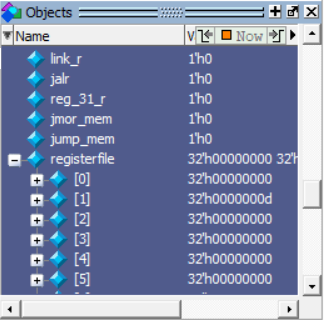
* ALU is modified to set untouched ALU input as result for JALR instruction.
* Added shift operation for SRLV instruction.
* Added operations for status register components such as overflow and nout.

1. **Test Runs**

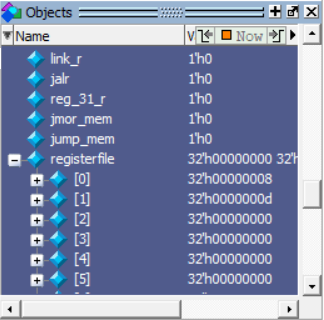
* **andi**

andi $0, $1, 8

Initial values:



After instruction:

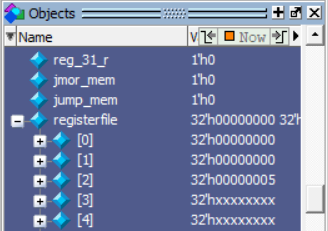


**-bneal**

bneal $0, $1, 8

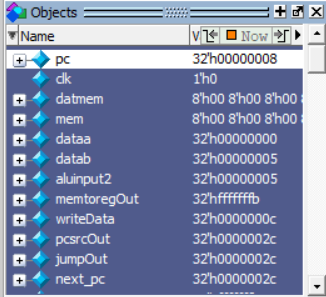
bneal $1, $2, 8

Initial values:

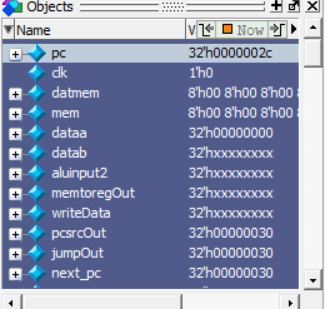


* After first instruction, do not jump (since $0=$1).

After first instruction:



After second one:

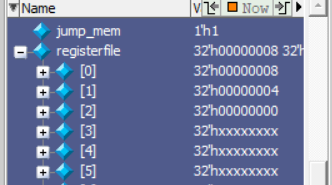


Jumps succesfully.

* **jmor**

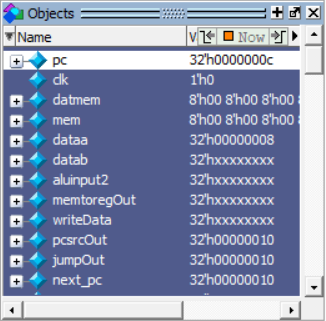
jmor $0, $1

Initial Values:



Data memory: 12-16 -> 00 00 00 0c

* + After instruction:



**-jalr**

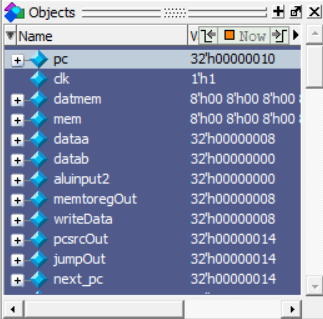
jalr $0, $1

Initial values:

$0=10

$1=10

After instruction:



**-balv:**

add $2, $1, $0

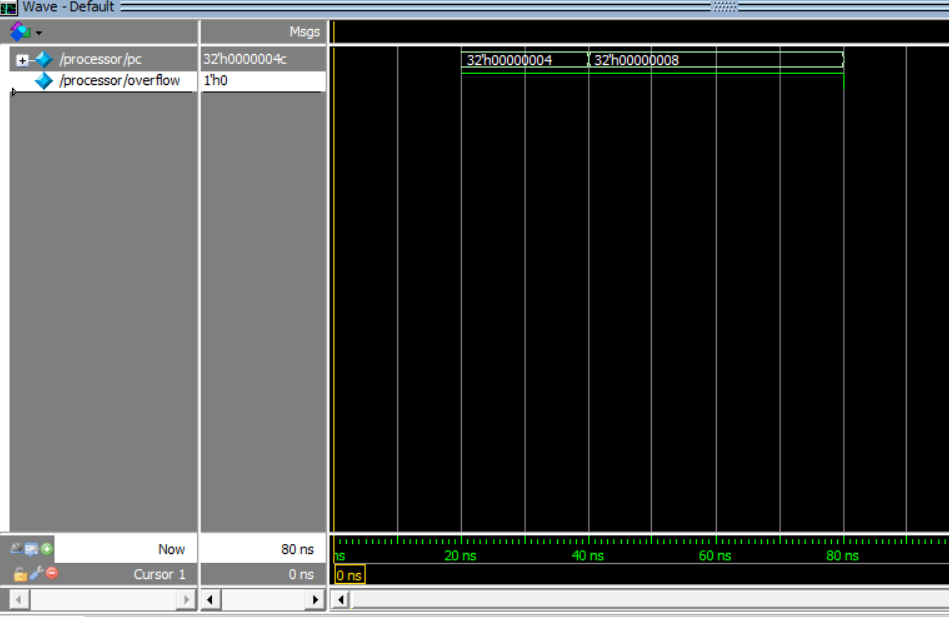
balv 16

$0= 7ff...f

$1= 1

$2 initially 0

After operation:



**-srlv:**

srlv $2, $0, $1

$0=8

$1=2

$2 initially 0, should be 2 after the operation

After instruction:

