Assignment 2

Tanmay
2020CS10399

- ALUCell:

  It is an ALU cell similar as described in Lecture 8 slide 22.

  It takes input $A|i$, B_i , C_i as Ain, Bin and Cin

  It takes input control as std logic vector(2 downto 0)

  It take operation as logic vector(1 downto 0);

  It outputs C_{i+1} as Cout and S i as S

  Effective a to take part in operations

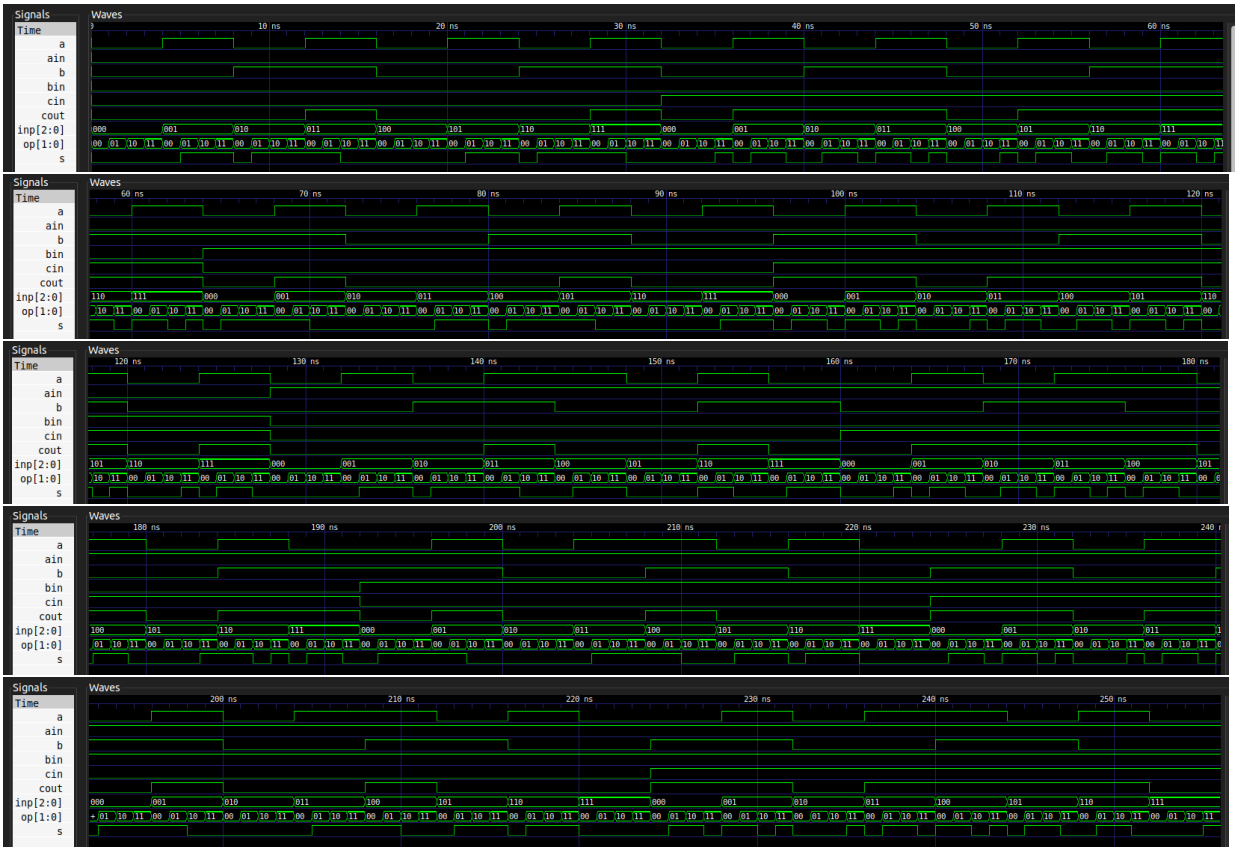
  a $\Leftarrow$ (Ain and Inp(2)) xor Inp(0);

  Effective b to take part in operations

  b $\Leftarrow$ Bin xor Inp(1);

  Testbench:

  As there are only 256 possible combinations of input in ALUcell, EPWave signals consists all possible input and their corresponding output.



  Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ******************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ******************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------------
# Info: IOs                               10      210       4.76%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              3       63400     0.00%
# Info: CLB Slices                        1       15850     0.01%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------------
# Info: ******************************************************
# Info: Library: work    Cell: ALUcell    View: arc
# Info: ******************************************************
# Info:  Number of ports :                            10
# Info:  Number of nets :                             21
# Info:  Number of instances :                        13
# Info:  Number of references to this view :           0
# Info: Total accumulated area :
# Info:  Number of LUTs :                              3
# Info:  Number of Primitive LUTs :                    3
# Info:  Number of accumulated instances :            13
# Info: ******************************
```

- ALU:

ALU is designed similar as described in lecture 8 slide 33

Only difference is that it takes input dp instruction (std logic vector(3 downto 0)) instead of inp(std logic vector(2 downto 0)) and op(std logic vector(1 downto 0))

ALU takes input Ain, Bin as std logic vector(31 downto 0)

It takes DP instructions as inst as std logic vector(3 downto 0)

It takes Cflag input as std logic

It outputs result Sout as std logic vector(31 downto 0)

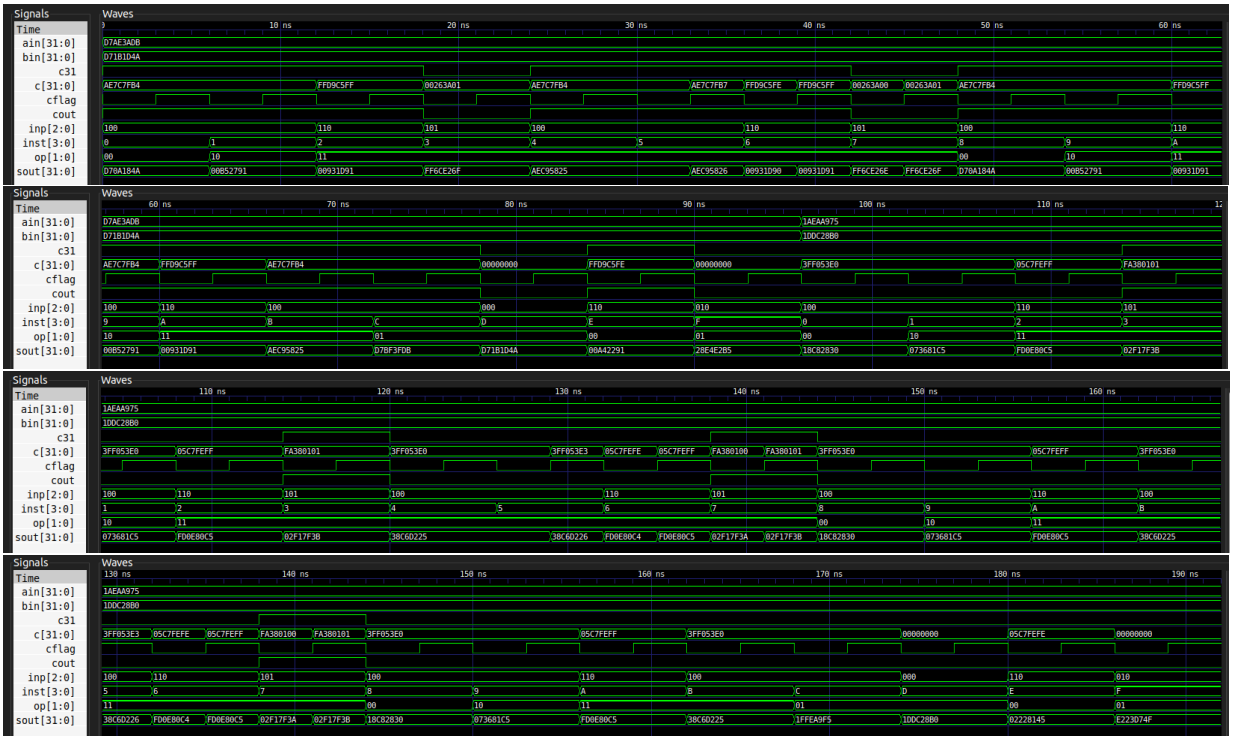
It outputs carry 31 as C31 and carry out as std logic

Testbench:

Test bench consists of two 32 bit integer and applying all possible DP instructions on them.

Next it consists of two 32 bit integer with some bits missing and applied all possible DP instructions on them.

Following shows EPWave signal for ALU tb



Device Utilization for 7A100TCSG324 is as follows:

```
# Info: Device Utilization for 7A100TCSG324
# Info: ***************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ---------------------------------------------------------------
# Info: IOs                               103     210      49.05%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              123     63400     0.19%
# Info: CLB Slices                        12      15850     0.08%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ---------------------------------------------------------------
# Info: *************************************************
# Info: Library: work    Cell: ALU    View: arc
# Info: *************************************************
# Info:  Number of ports :                          103
# Info:  Number of nets :                           320
# Info:  Number of instances :                      251
# Info:  Number of references to this view :          0
# Info: Total accumulated area :
# Info:  Number of LUTs :                           123
# Info:  Number of Primitive LUTs :                 148
# Info:  Number of LUTs with LUTNM/HLUTNM :          50
# Info:  Number of accumulated instances :          251
# Info: ******************************
```

- Register File:

  It takes input 2 read address and 1 write address as rdAddr1, rdAddr2 and wrAddr as std logic vector(3

downto 0)

It takes input a clock as clk as std logic

It takes input whether to write data or to read data as regWrite as std logic

It takes input write data as wrData as std logic(31 downto 0)

It outputs 2 read data as rdData1, rdData2 as std logic vector(31 downto 0)

Grid is defined as array of std logic

Registers are defined as grid

indexWr, indexRd1, indexRd2 to convert address to integer

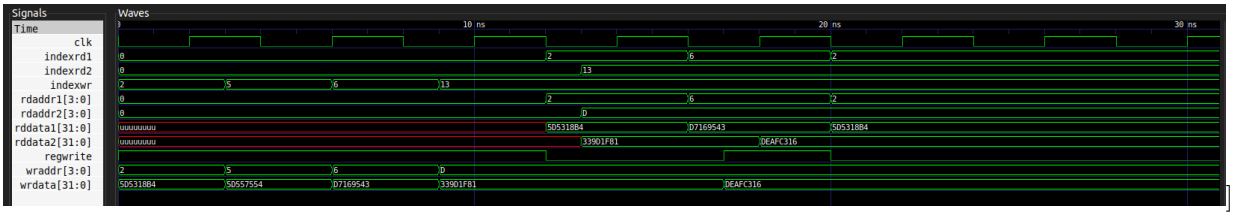
Reads data when regWrite = 0

Writes data with clock and regWrite = 1

Testbench:

Wrote some data in some register and read the data from some register

Following shows EPWave signal for registerFile tb



Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------
# Info: IOs                               110     210      52.38%
# Info: Global Buffers                    1       32        3.12%
# Info: LUTs                              272     63400     0.43%
# Info: CLB Slices                        68      15850     0.43%
# Info: Dffs or Latches                   512     126800    0.40%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------
# Info: ************************************************************
# Info: Library: work    Cell: registerFile    View: arc
# Info: ************************************************************
# Info:  Number of ports :                          110
# Info:  Number of nets :                           218
# Info:  Number of instances :                      110
# Info:  Number of references to this view :          0
# Info: Total accumulated area :
# Info:  Number of Dffs or Latches :                512
# Info:  Number of LUTs :                           272
# Info:  Number of Primitive LUTs :                 272
# Info:  Number of MUXF7 :                          128
# Info:  Number of MUXF8 :                           64
# Info:  Number of accumulated instances :         1086
# Info: ******************************
```

- Memory:

  It is an memory similar as described in Lecture 9 slide 23.

  Address is input as addr as std logic vector(8 downto 0);

  Datain and Dataout is std logic vector(31 downto 0);

  Read - Write signal is declared as Signal as std logic;

  byteSignal gives the power to read-write at byte level.

  If byte signal is "1001" then this implies to read/write std logic vector(31 downto 24) and std logic vector(7 downto 0) only.

  clock is input as clk as std logic

  Wrote some data in some location of memory and read the data from some location of memory.

  Following shows EPWave signal for dataMemory tb

  Testbench:

Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ******************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ******************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------------
# Info: IOs                               79      210      37.62%
# Info: Global Buffers                    4       32       12.50%
# Info: LUTs                              2040    63400     3.22%
# Info: CLB Slices                        512     15850     3.23%
# Info: Dffs or Latches                   4096    126800    3.23%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------------
# Info: ****************************************************
# Info: Library: work    Cell: memory    View: arc
# Info: ****************************************************
# Info:  Number of ports :                            79
# Info:  Number of nets :                            328
# Info:  Number of instances :                       254
# Info:  Number of references to this view :           0
# Info: Total accumulated area :
# Info:  Number of Dffs or Latches :                4096
# Info:  Number of LUTs :                           2040
# Info:  Number of LUTs with LUTNM/HLUTNM :           54
# Info:  Number of MUXF7 :                            32
# Info:  Number of accumulated instances :          6254
# Info: ******************************
```

- Program Counter:

It takes input clock as std logic

It takes input offset as std logic vector(23 downto 0) which is directly retrieved from instruction of memory

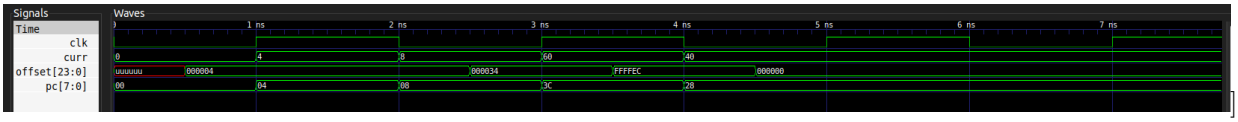
It return pointer to next program address

It maintains current pointer to program address

One every clock cycle, it adds offset to current pointer and returns pc

Testbench:

Following shows EPWave signal for programCounter tb



Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ******************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ******************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------------
# Info: IOs                               33      210      15.71%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              17      63400     0.03%
# Info: CLB Slices                        2       15850     0.01%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------------
# Info: ****************************************************************
# Info: Library: work    Cell: programCounter    View: arc
# Info: ****************************************************************
# Info:  Number of ports :                          33
# Info:  Number of nets :                          124
# Info:  Number of instances :                     100
# Info:  Number of references to this view :         0
# Info: Total accumulated area :
# Info:  Number of LUTs :                           17
# Info:  Number of Primitive LUTs :                 21
# Info:  Number of LUTs with LUTNM/HLUTNM :          8
# Info:  Number of MUX CARRYs :                     23
# Info:  Number of accumulated instances :         100
# Info: ******************************
```

- Flags:

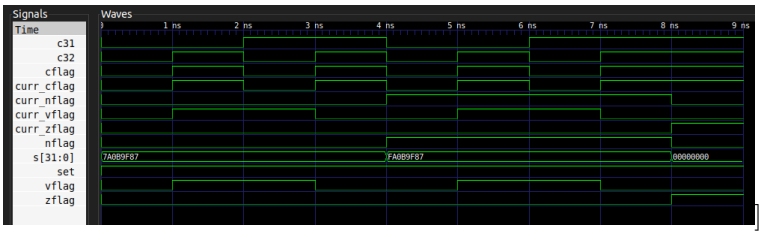
  Takes input result of ALU as S as std logic vector(31 downto 0);

  Takes input Last two carry values C31 and C32 as std logic

  Output updated flags N, Z, C and V as std logic if set = 1

  Testbench:

  Following shows EPWave signal for flags tb



  Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------
# Info: IOs                               39      210      18.57%
# Info: Global Buffers                    1       32        3.12%
# Info: LUTs                              8       63400     0.01%
# Info: CLB Slices                        1       15850     0.01%
# Info: Dffs or Latches                   4       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------
# Info: ****************************************************
# Info: Library: work    Cell: flags    View: arc
# Info: ****************************************************
# Info:  Number of ports :                          39
# Info:  Number of nets :                           88
# Info:  Number of instances :                      53
# Info:  Number of references to this view :         0
# Info: Total accumulated area :
# Info:  Number of Dffs or Latches :                 4
# Info:  Number of LUTs :                            8
# Info:  Number of LUTs with LUTNM/HLUTNM :          2
# Info:  Number of accumulated instances :          53
# Info: ******************************
```

- Condition Checker (Predicate):

  It checks the conditions like EQ, NE, GE, LE, GT, LT etc.

  It takes input condition field as std logic vector(3 downto 0);

  It takes input all flags namely N, Z, C and V as std logic;

  It outputs '1' if condition is true or '0' if condition is false as result as std logic

  Testbench:

  As there are only 256 combinations possible therefore we can try checking all of them.

  Following shows EPWave signal for predicate tb

Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ******************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ******************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------------
# Info: IOs                               9       210       4.29%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              4       63400     0.01%
# Info: CLB Slices                        1       15850     0.01%
# Info: Dffs or Latches                   1       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------------
# Info: ********************************************************
# Info: Library: work    Cell: predicate    View: arc
# Info: ********************************************************
# Info:  Number of ports :                          9
# Info:  Number of nets :                          24
# Info:  Number of instances :                     16
# Info:  Number of references to this view :        0
# Info: Total accumulated area :
# Info:  Number of Dffs or Latches :                1
# Info:  Number of LUTs :                           4
# Info:  Number of Primitive LUTs :                 4
# Info:  Number of accumulated instances :         16
# Info: ******************************
```

- Data Register:

  It is a basic register which takes in clk as std logic;

  It takes in write Signal as wrSignal as std logic;

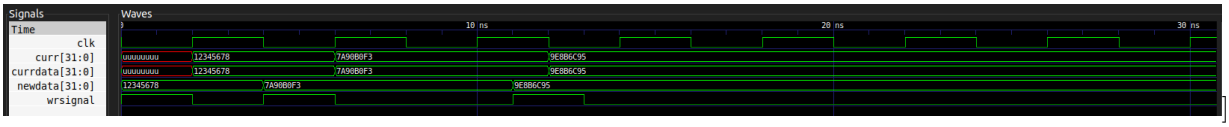
  It takes in new Data as std logic vector(31 downto 0);

  It outputs curr Data stored as std logic vector(31 downto 0);

  Testbench:

  Wrote some data into the register and accessed it.

Following shows EPWave signal for dataRegister tb



Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ******************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ******************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------------
# Info: IOs                               66      210      31.43%
# Info: Global Buffers                    1       32        3.12%
# Info: LUTs                              0       63400     0.00%
# Info: CLB Slices                        4       15850     0.03%
# Info: Dffs or Latches                   32      126800    0.03%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------------
# Info: ************************************************************
# Info: Library: work    Cell: dataRegister    View: arc
# Info: ************************************************************
# Info:  Number of ports :                          66
# Info:  Number of nets :                          132
# Info:  Number of instances :                      99
# Info:  Number of references to this view :         0
# Info: Total accumulated area :
# Info:  Number of Dffs or Latches :                32
# Info:  Number of gates :                           0
# Info:  Number of accumulated instances :          99
# Info: ******************************
```

- Bit Operator:

It processes all bit operations like left shift, right shift, arithmetic right shift and right rotate;

It performs basic bit operations of size 1, 2, 4, 8 and 16 and combines them to perform bit operations of any size between 0 and 31.

It shifts/rotate input by given amount according to the given operation

takes input a as std logic vector(31 downto 0);

it takes operation as op as std logic vector(1 downto 0);

op = ”00” ⇒ LSL

op = ”01” ⇒ LSR

op = ”10” ⇒ ASR

op = ”11” ⇒ ROR

shift = 1 ⇒ we have to return shifted output

shift = 0 ⇒ we have to return output same as input

It output cout as std logic

Device Utilization for 7A100TCSG324 is as follows:

- # Multiplier:

  It takes a, b, as std logic vector(31 downto 0),

  It takes c as std logic vector(63 downto 0)

  It also takes allow, sgn and acc input as signed

  It outputs result as res as std logic vector(63 downto 0)

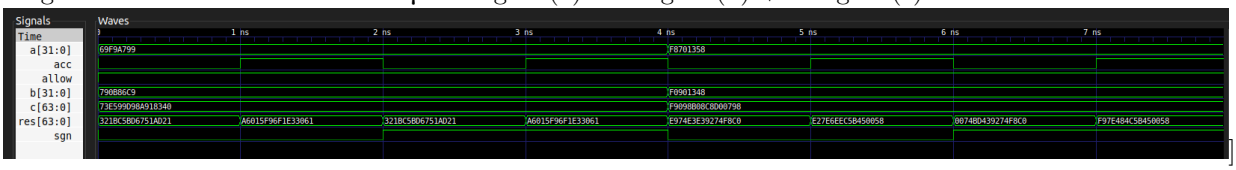
  It multiplies only when allow is '1'. This is done to prevent any kind of undefined signal as it can lead to errors

  If sgn = '1' and acc = '0' then res ¡= signed(a) * signed(b)

  If sgn = '0' and acc = '0' then res ¡= unsigned(a) * unsigned(b)

  If sgn = '1' and acc = '1' then res ¡= signed(a) * signed(b) + signed(c)

  If sgn = '0' and acc = '1' then res ¡= unsigned(a) * unsigned(b) + unsigned(c)



Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ******************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ******************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------------
# Info: IOs                               195     210      92.86%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              128     63400     0.20%
# Info: CLB Slices                        32      15850     0.20%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          8       240       3.33%
# Info: ------------------------------------------------------------------
# Info: ********************************************************
# Info: Library: work    Cell: multiplier    View: arc
# Info: ********************************************************
# Info:  Number of ports :                          195
# Info:  Number of nets :                          1071
# Info:  Number of instances :                      460
# Info:  Number of references to this view :          0
# Info: Total accumulated area :
# Info:  Number of DSP48E1s :                         8
# Info:  Number of LUTs :                           128
# Info:  Number of Primitive LUTs :                 128
# Info:  Number of MUX CARRYs :                      63
# Info:  Number of accumulated instances :          460
# Info: ******************************
```

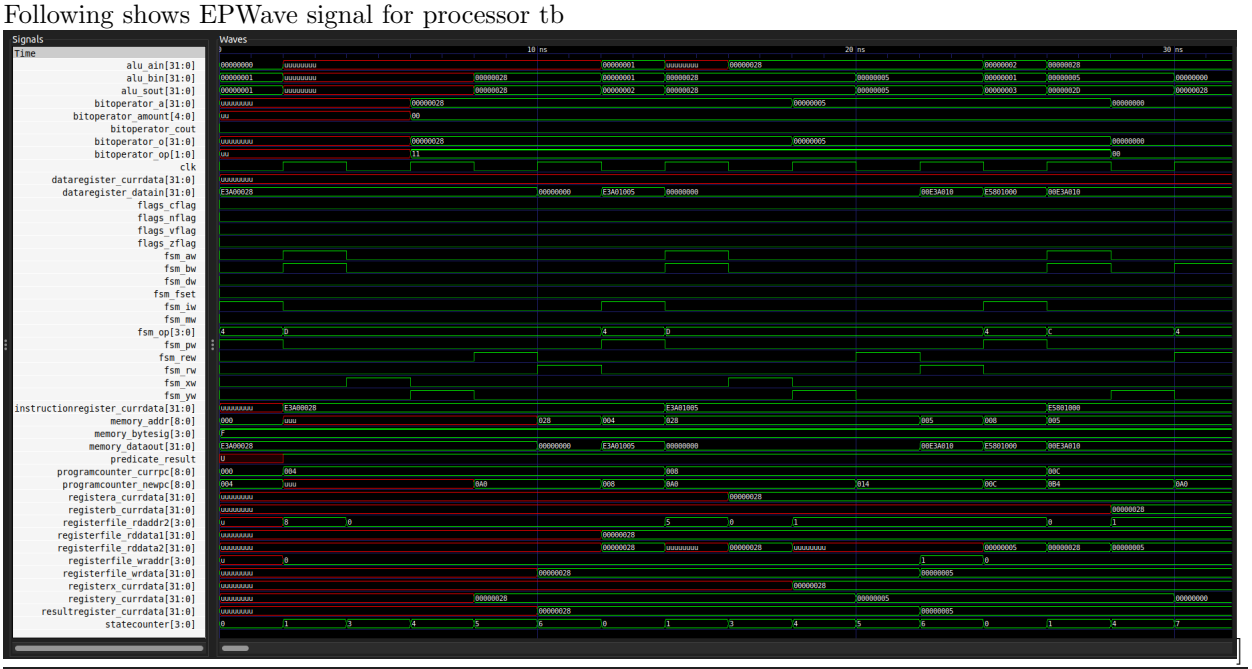
- Processor:

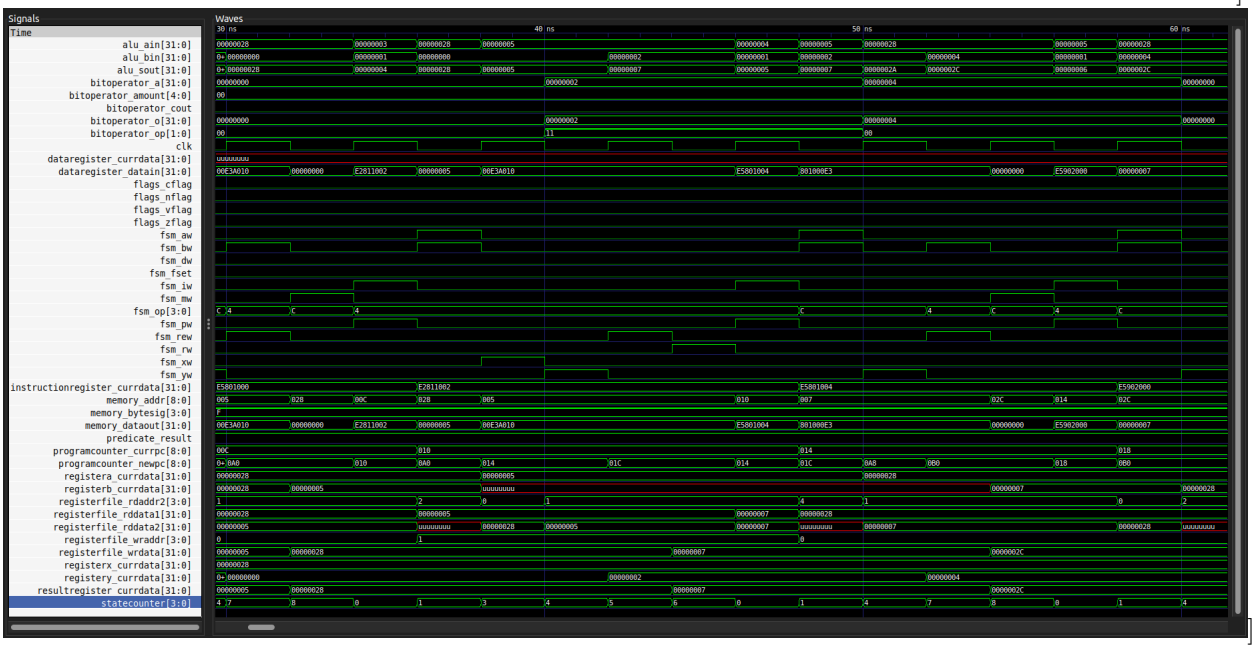
It only takes input clock as clk as std logic
It assembles all the components required in a processor
Program Counter
It maintains program counter and it's offset depends on the instruction.
If instruction is branched and condition is satisfied then it changes offset to the offset provided in instruction
Memory
It output instruction set and Allows read or write of memory from register
Register File
It reads and writes the data from memory to registerFile or vice versa or directly from instruction set
ALU
Take two inputs from registers or one from register and another from instruction set and performs arithmetic or logical operation on them. It also provides carry out
Flags
Updates flag whenever it is dp instruction and flag set condition is turned on
Cycle Counter
Changes the cycle number on every clock cycle
According to different instruction and cycleCounter, FSM controls are changed
Changes done by FSM controls:
updates programCounter newPc
specifies from where to get memory addr (i.e., from programCounter or from result Register)
specifies from where to get registerFile address 2 (from instruction register (3 downto 0) or from instruction Register (15 downto 12))
specifies from where to get registerFile wrData (from result Register or from data Register)
specifies from where to get the operands of ALU, operand 1 (from register A or programCounter) and

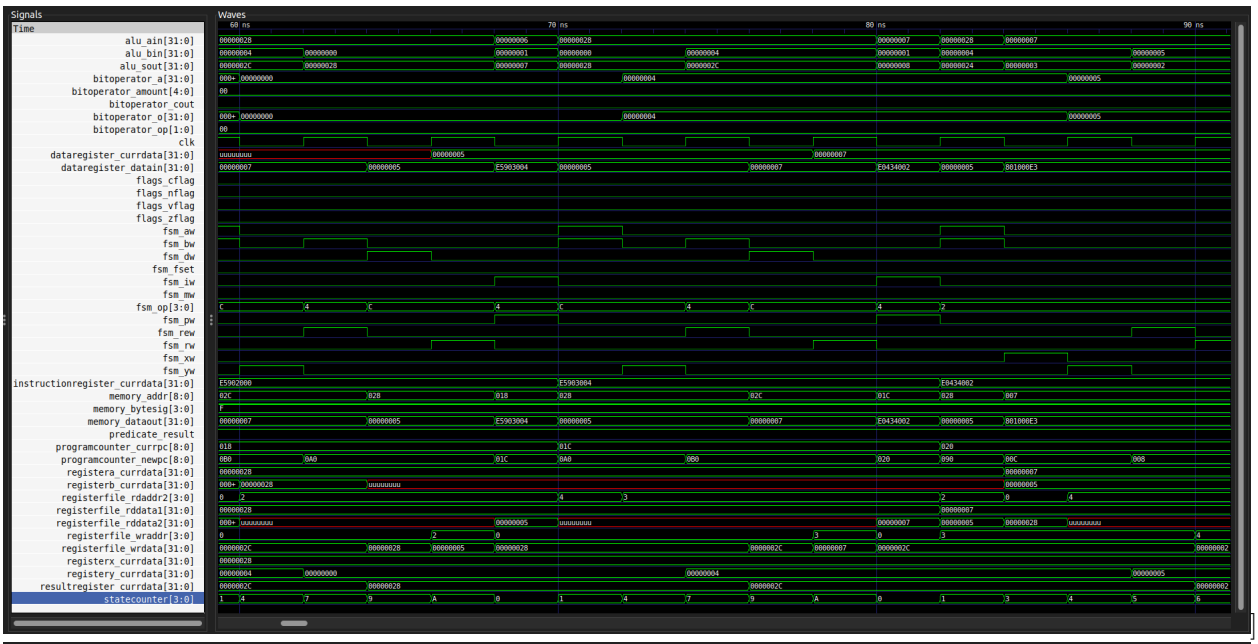
operand 2 (from register B, constant, or instruction register)

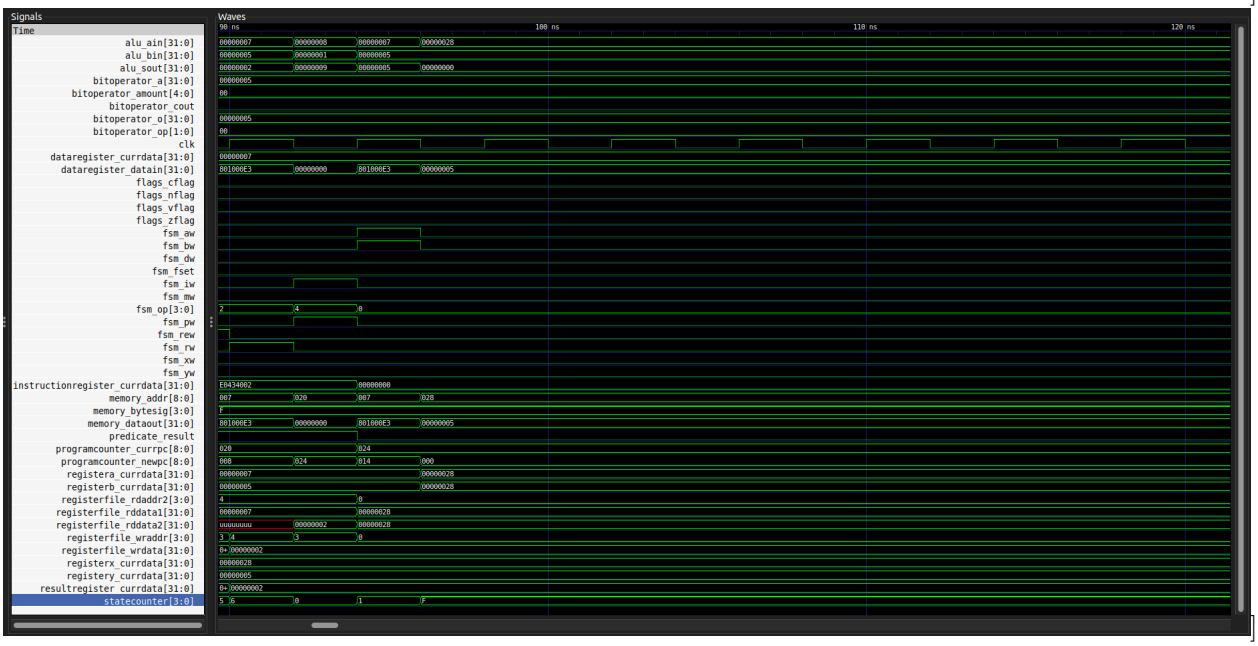
Testbench:
PROGRAM 1:
signal mem: grid :=(
0 ⇒ X"E3A00028", – mov r0, #40
1 ⇒ X"E3A01005", – mov r1, #5
2 ⇒ X"E5801000", – str r1, [r0]
3 ⇒ X"E2811002", – add r1, #2
4 ⇒ X"E5801004", – str r1, [r0, #4]
5 ⇒ X"E5902000", – ldr r2, [r0]
6 ⇒ X"E5903004", – ldr r3, [r0, #4]
7 ⇒ X"E0434002", – sub r4, r3, r2
others ⇒ X"00000000" – .end
);

Here program accurately stores and loads the data in and out of memory as well as registers.
And as the program ends, r3 contains 7, r2 contains 5 and r4 contains 7-5=2
Following shows EPWave signal for processor tb



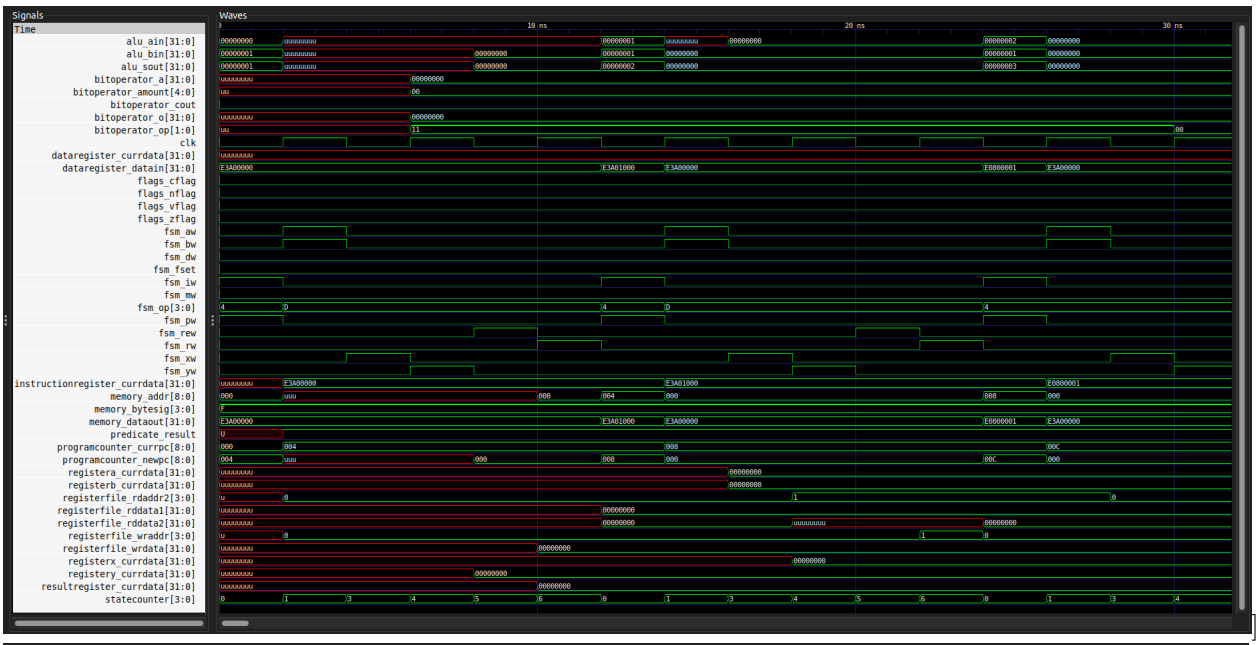


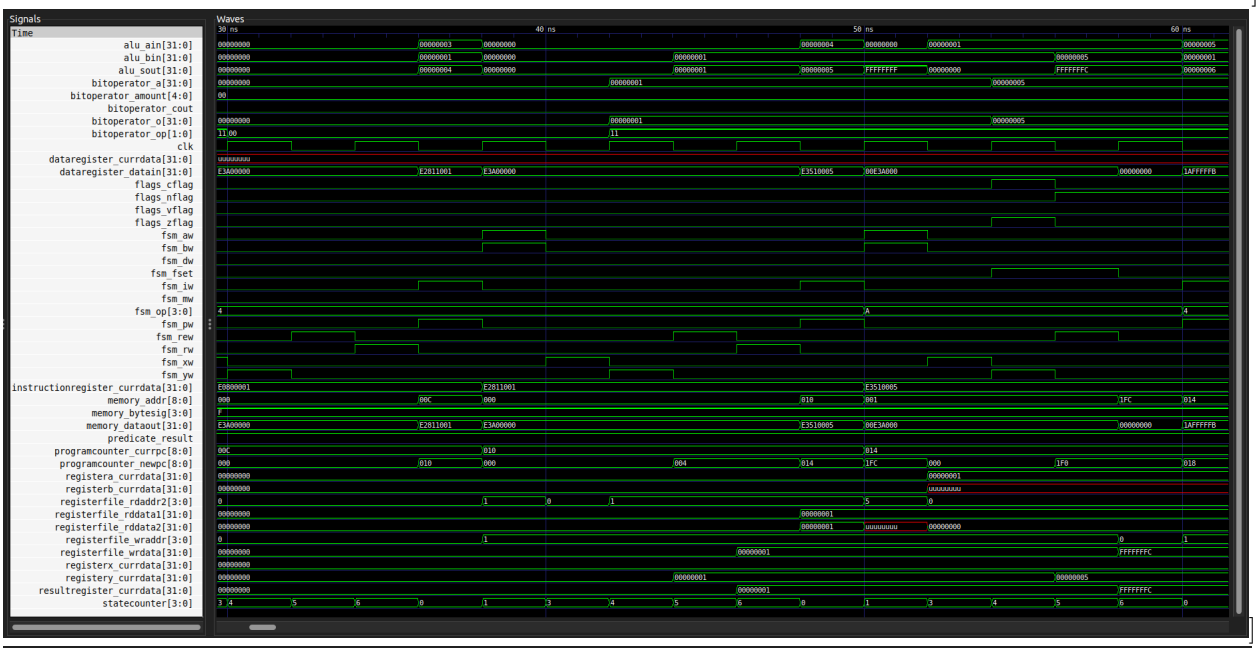


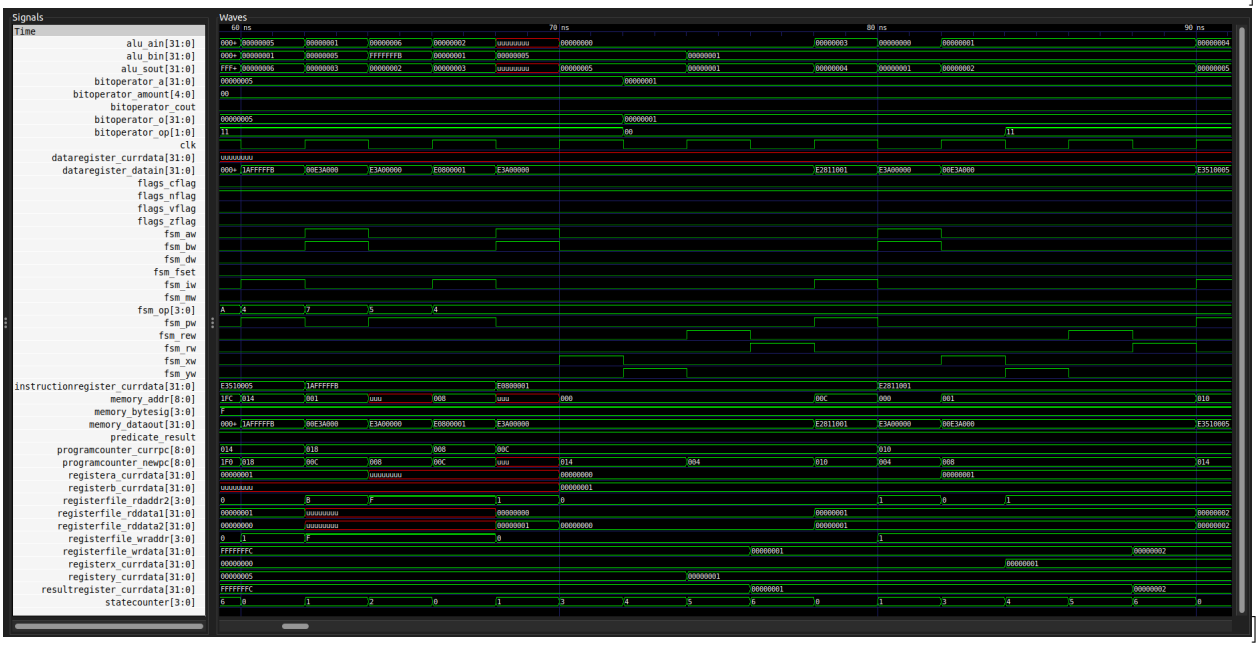


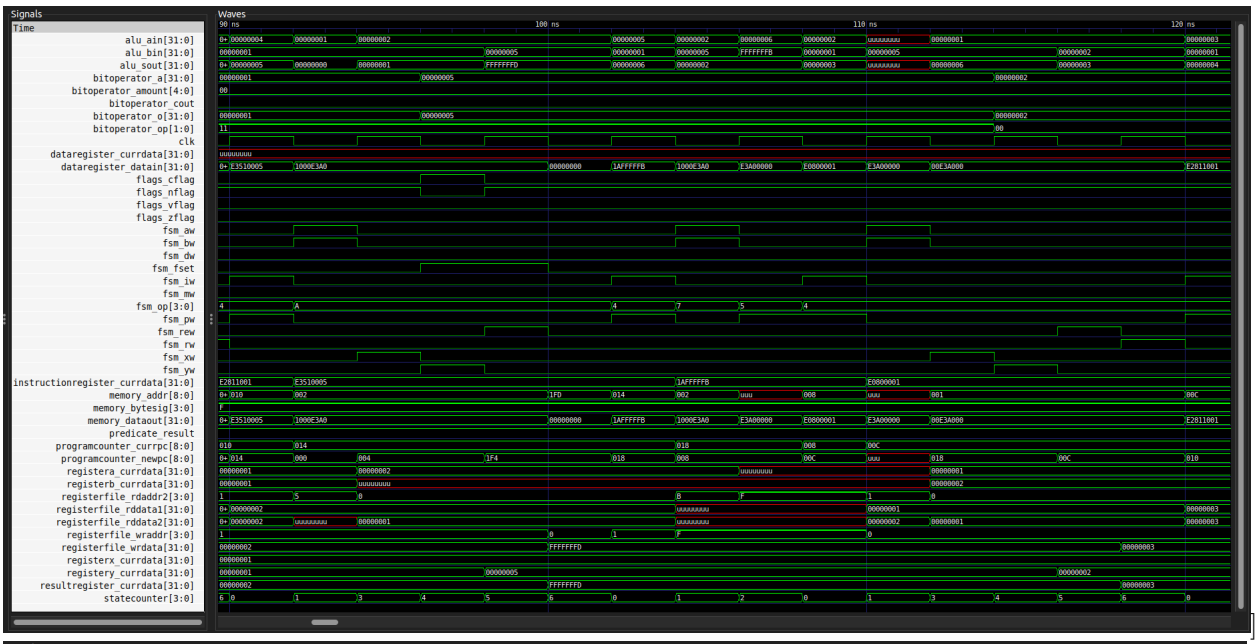
PROGRAM 2:
signal mem: grid :=(
0 ⇒ X”E3A00000”, – mov r0, #0
1 ⇒ X”E3A01000”, – mov r1, #0
2 ⇒ X”E0800001”, – Loop: add r0, r0, r1
3 ⇒ X”E2811001”, – add r0, r0, #1
4 ⇒ X”E3510005”, – cmp r1, #5
5 ⇒ X”1AFFFFFB”, – bne Loop
others ⇒ X”00000000” – .end
);

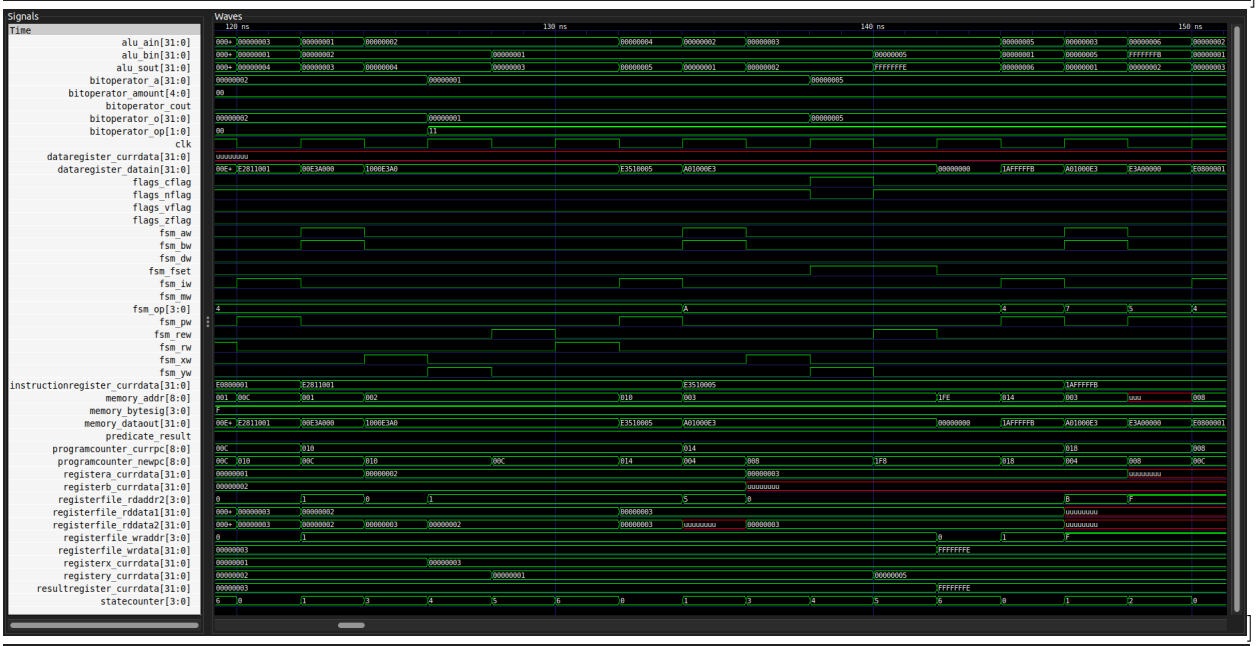
Here as the program ends, r0 contains 10 and r1 contains 5 which is indeed valid.
Following shows EPWave signal for processor tb

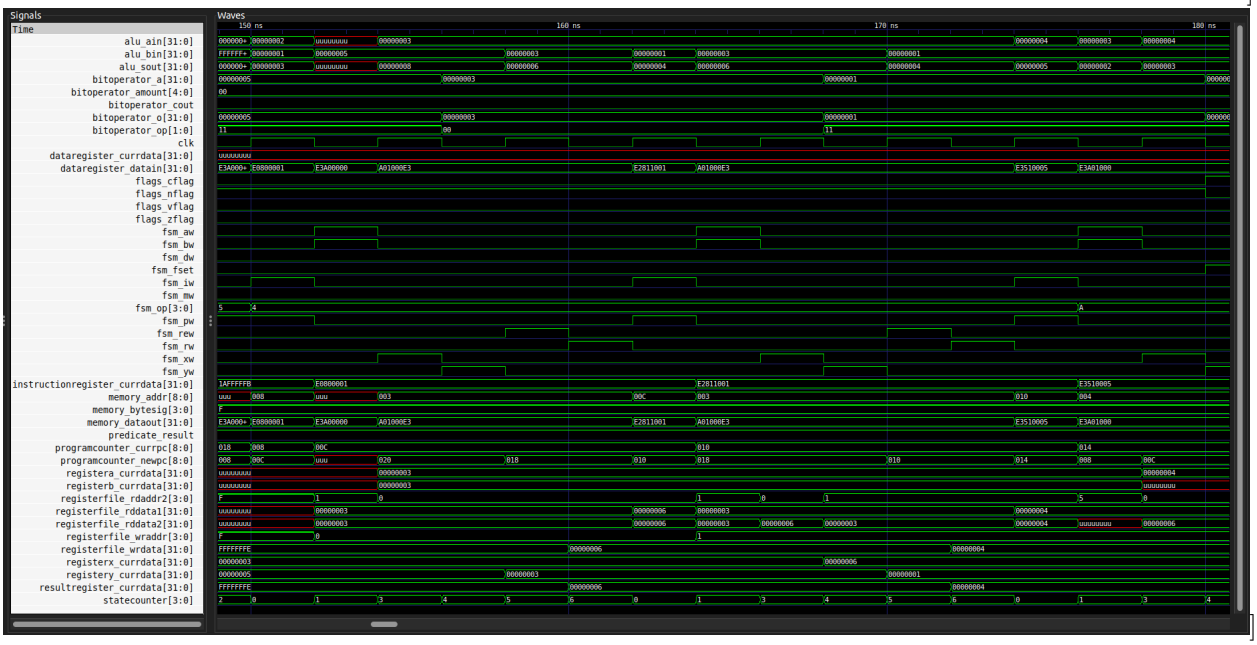

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
10 ns
20 ns
30 ns

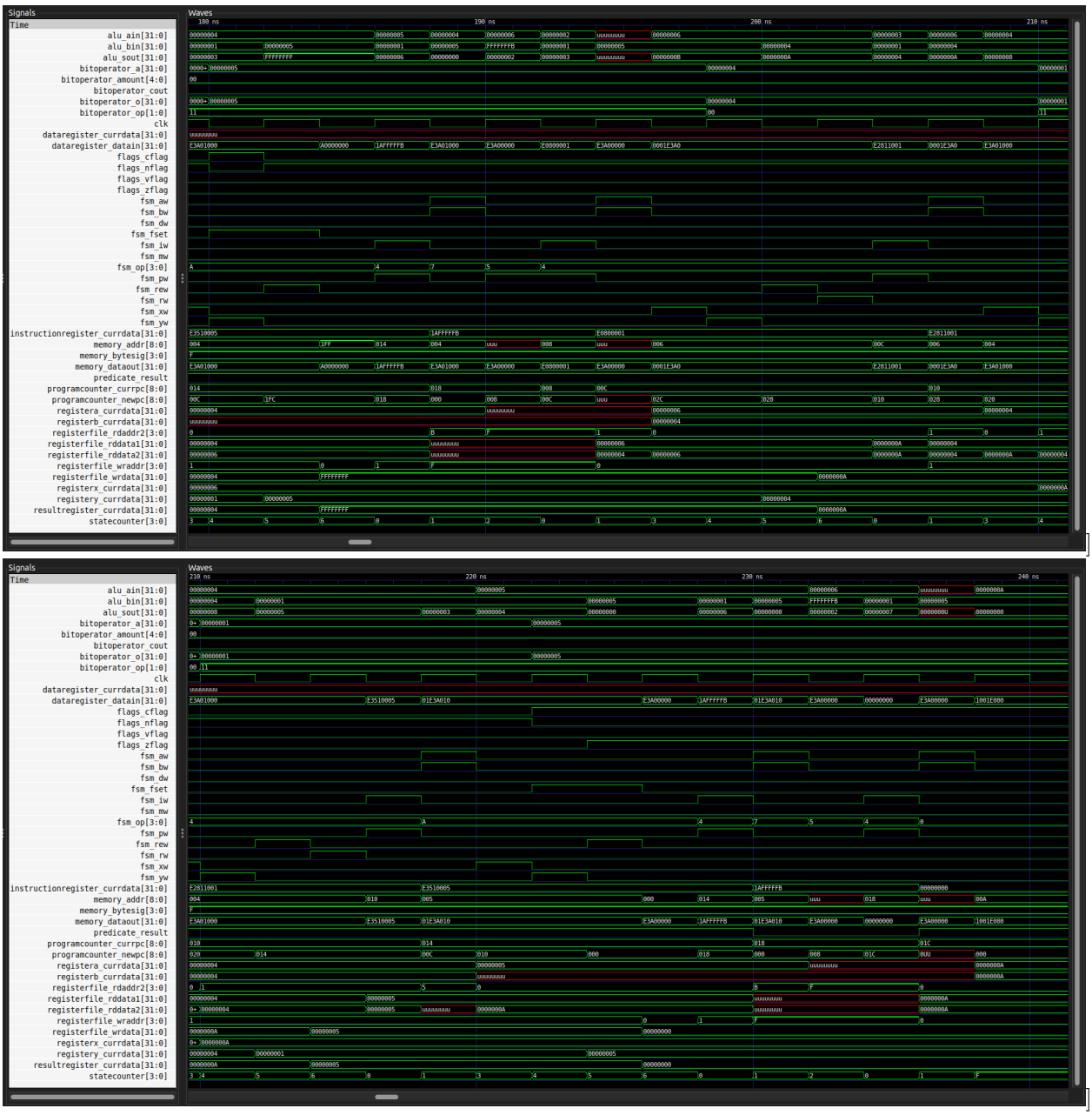

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
30 ns
40 ns
50 ns
60 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
60 ns
70 ns
80 ns
90 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
90 ns
100 ns
110 ns
120 ns

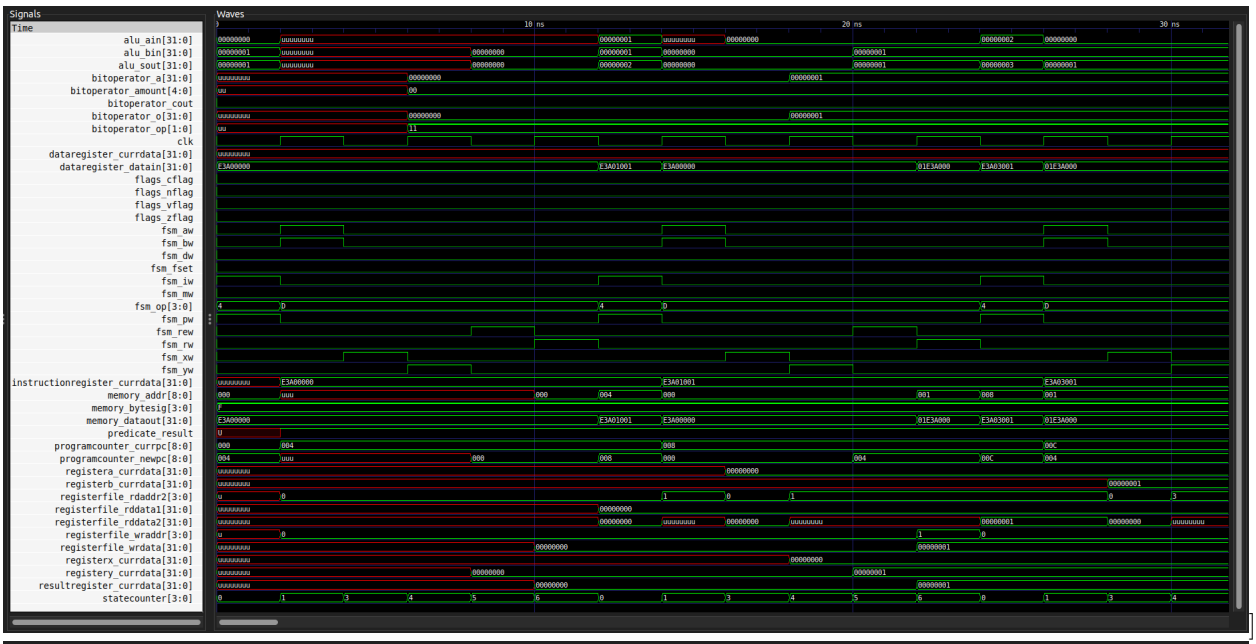

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
120 ns
130 ns
140 ns
150 ns

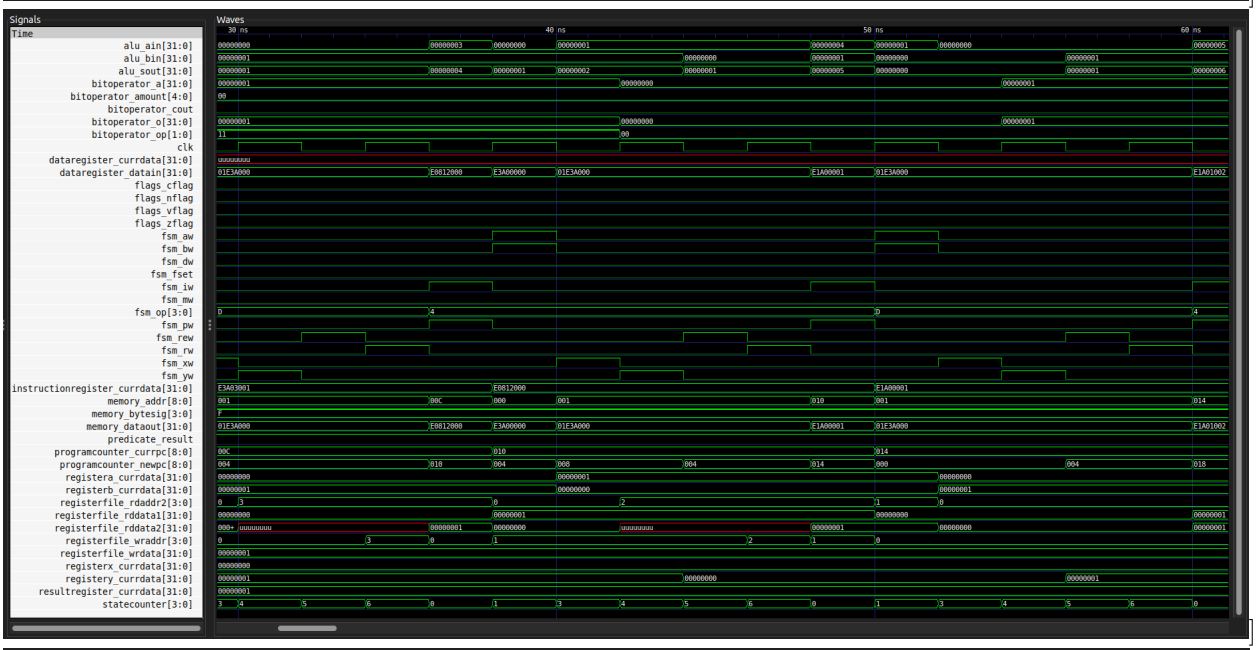

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
150 ns
160 ns
170 ns
180 ns

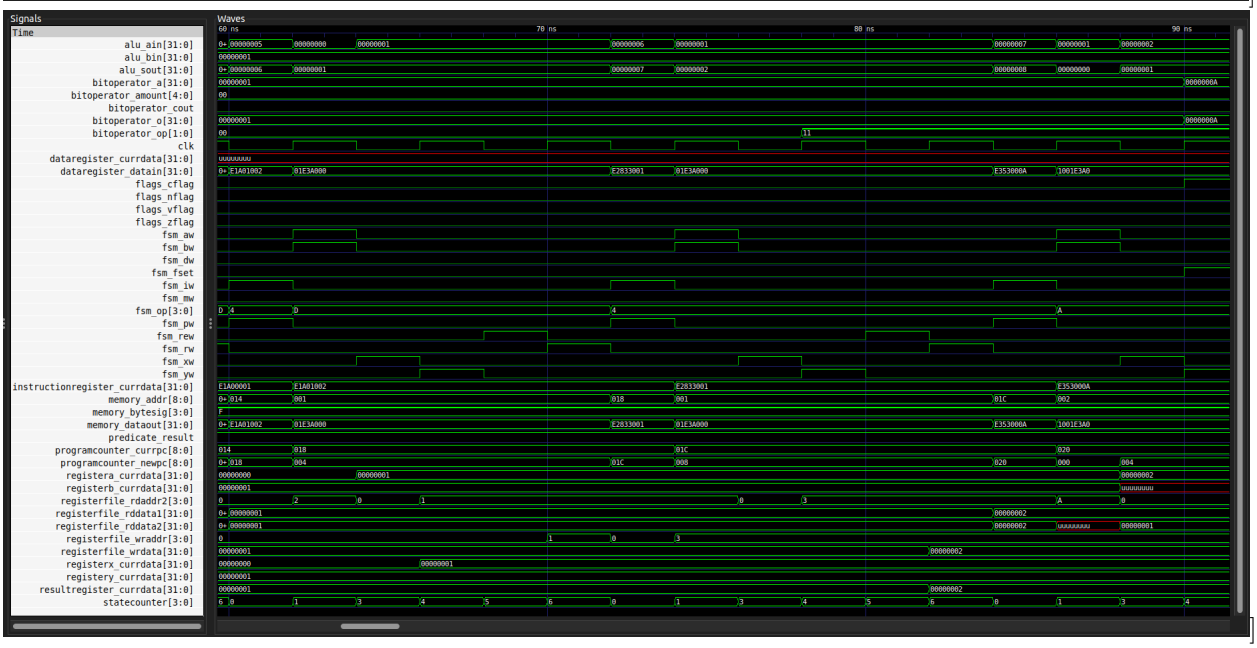


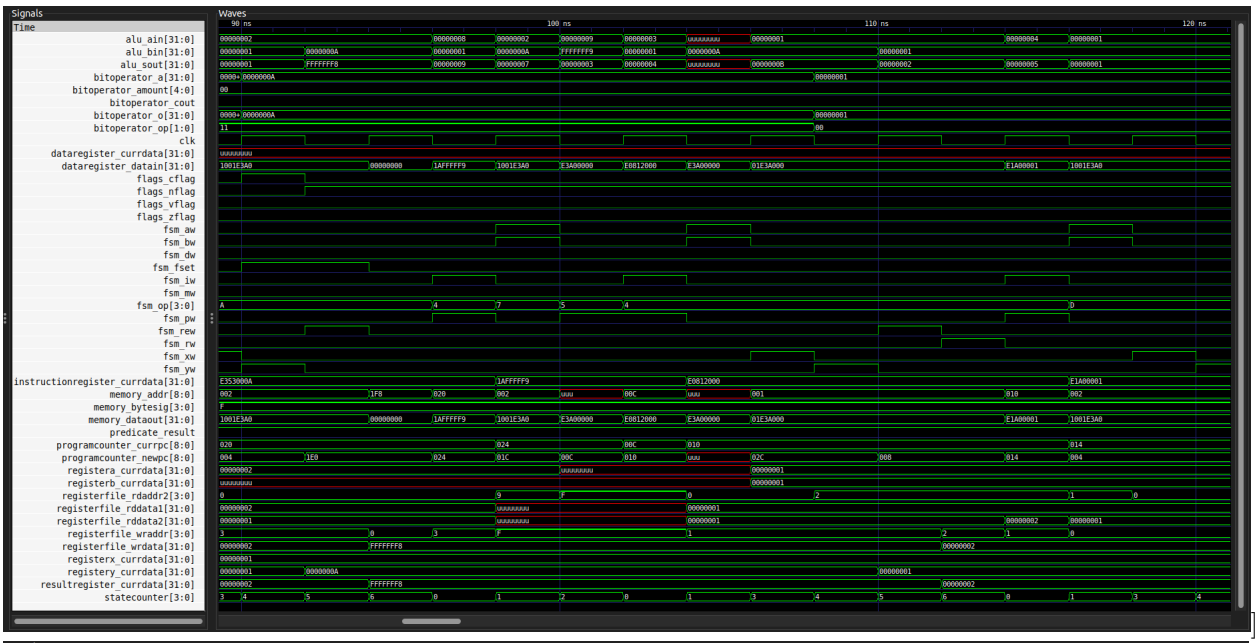
PROGRAM 3:
signal mem: grid :=(
0 ⇒ X"E3A00000", – mov r0, #0
1 ⇒ X"E3A01001", – mov r1, #1
2 ⇒ X"E3A03001", – mov r3, #1
3 ⇒ X"E0812000", – Loop: add r2, r1, r0
4 ⇒ X"E1A00001", – mov r0, r1
5 ⇒ X"E1A01002", – mov r1, r2
6 ⇒ X"E2833001", – add r3, r3, #1
7 ⇒ X"E353000A", – cmp r3, #10
8 ⇒ X"1AFFFFF9", – bne Loop
others ⇒ X"00000000" – .end
);

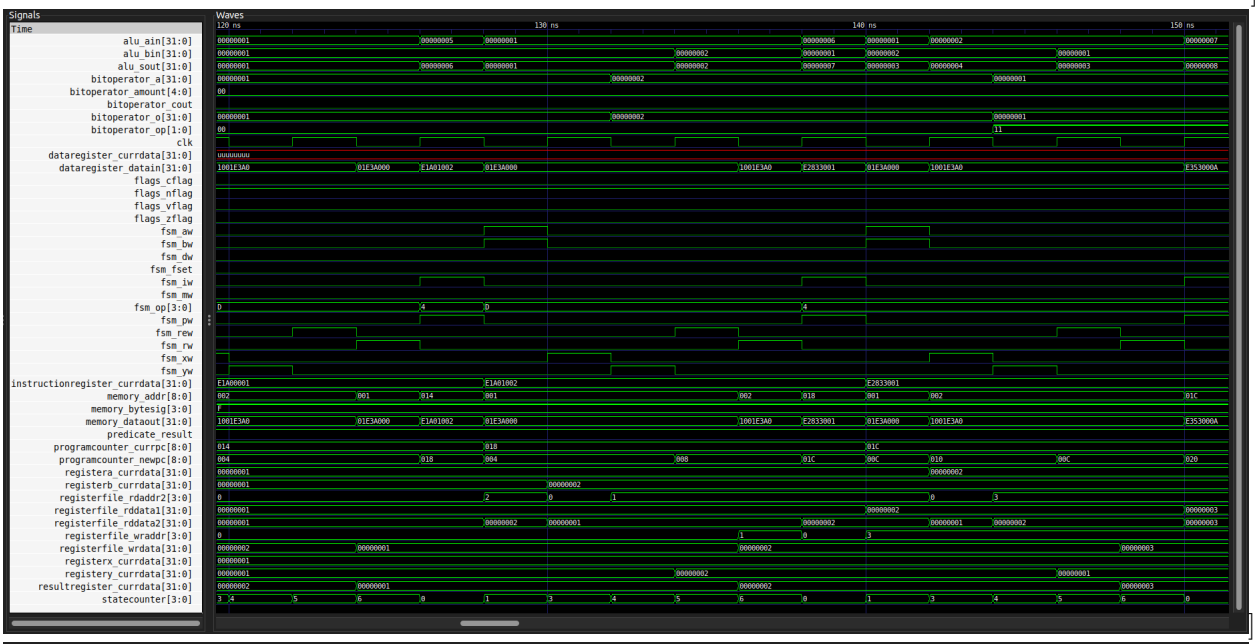
Here as the program ends, r3 contains 10, r1 contains 55 and r0 contain 34 i.e., r1 is 10^{th} Fibonacci number and r0 is 9^{th} Fibonacci number
Following shows EPWave signal for processor tb

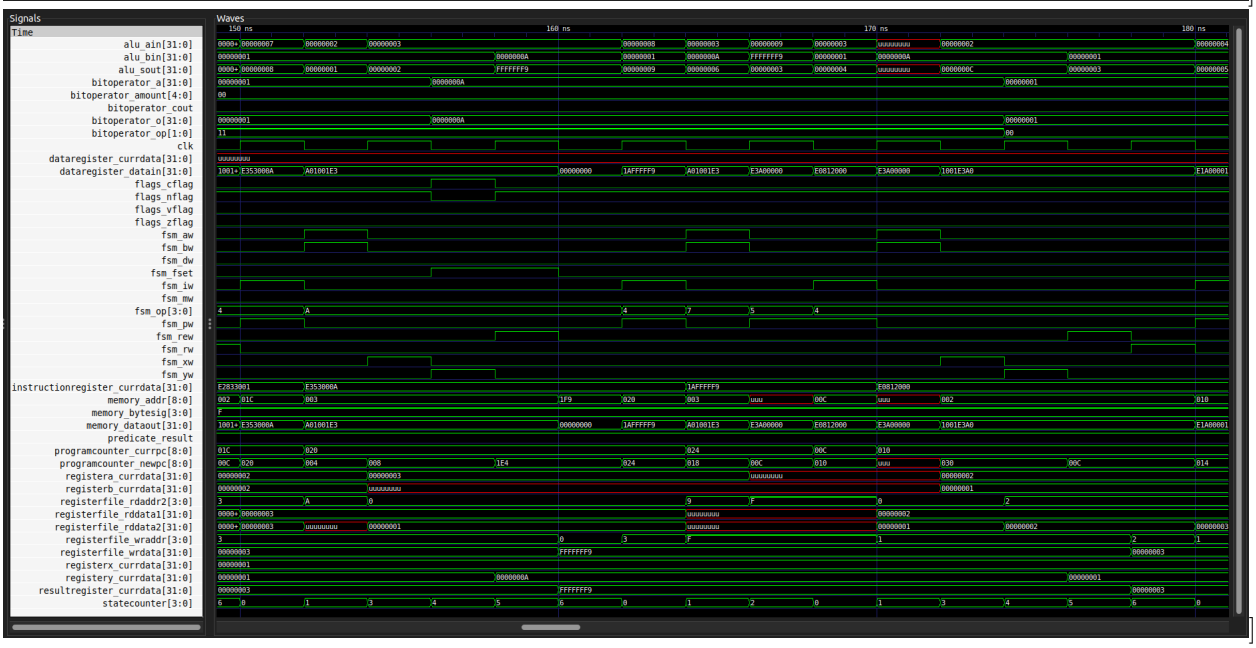

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
10 ns
20 ns
30 ns

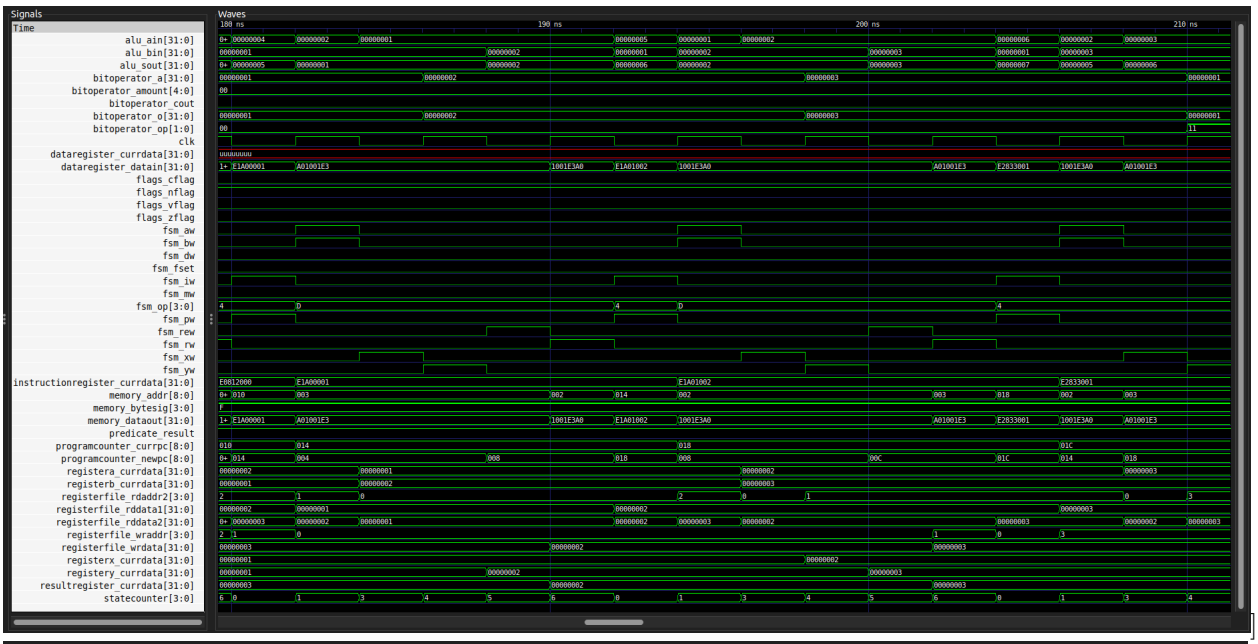

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
30 ns
40 ns
50 ns
60 ns

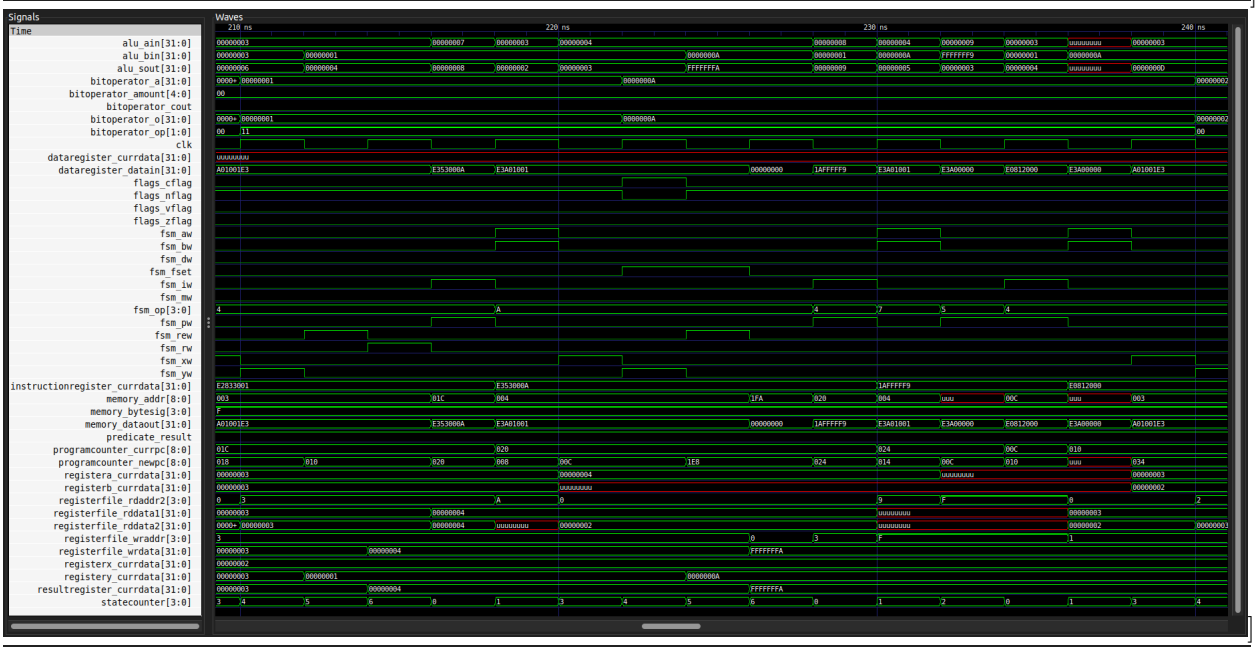

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
60 ns
70 ns
80 ns
90 ns

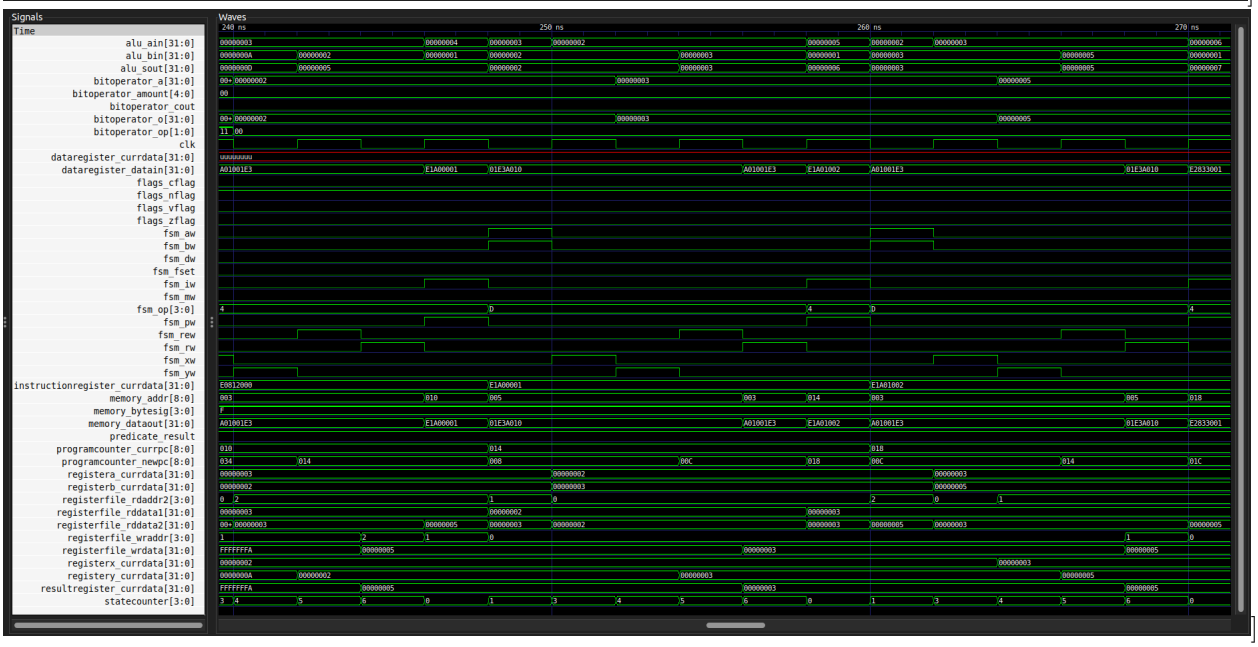

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
90 ns
100 ns
110 ns
120 ns

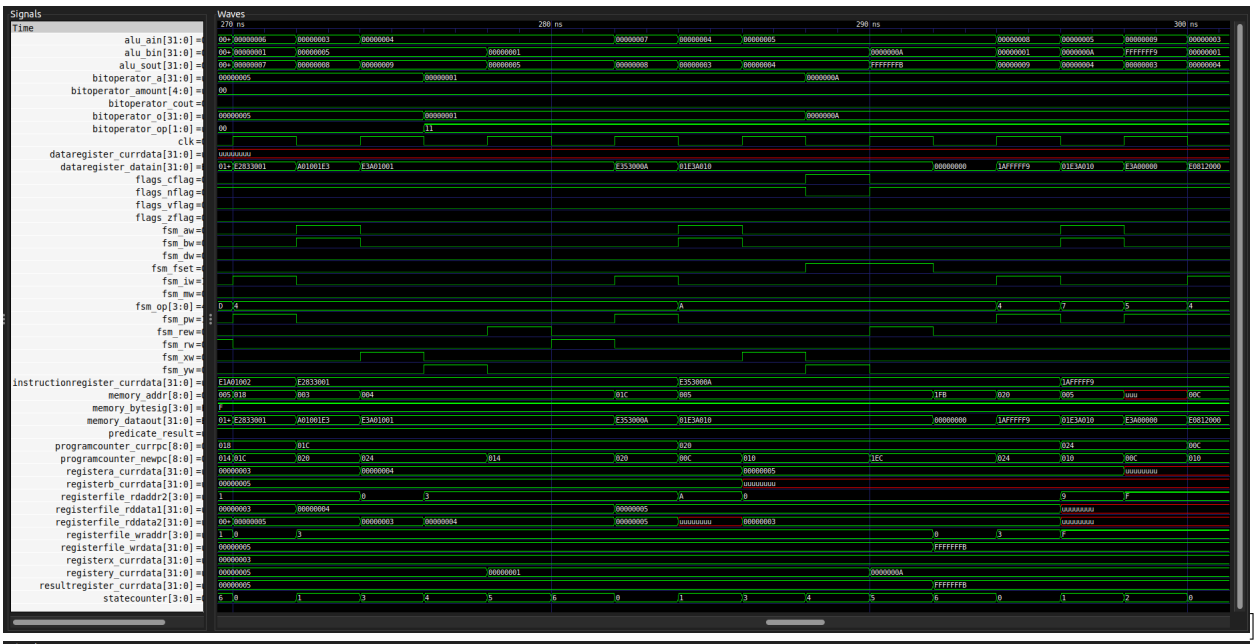

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
120 ns
130 ns
140 ns
150 ns

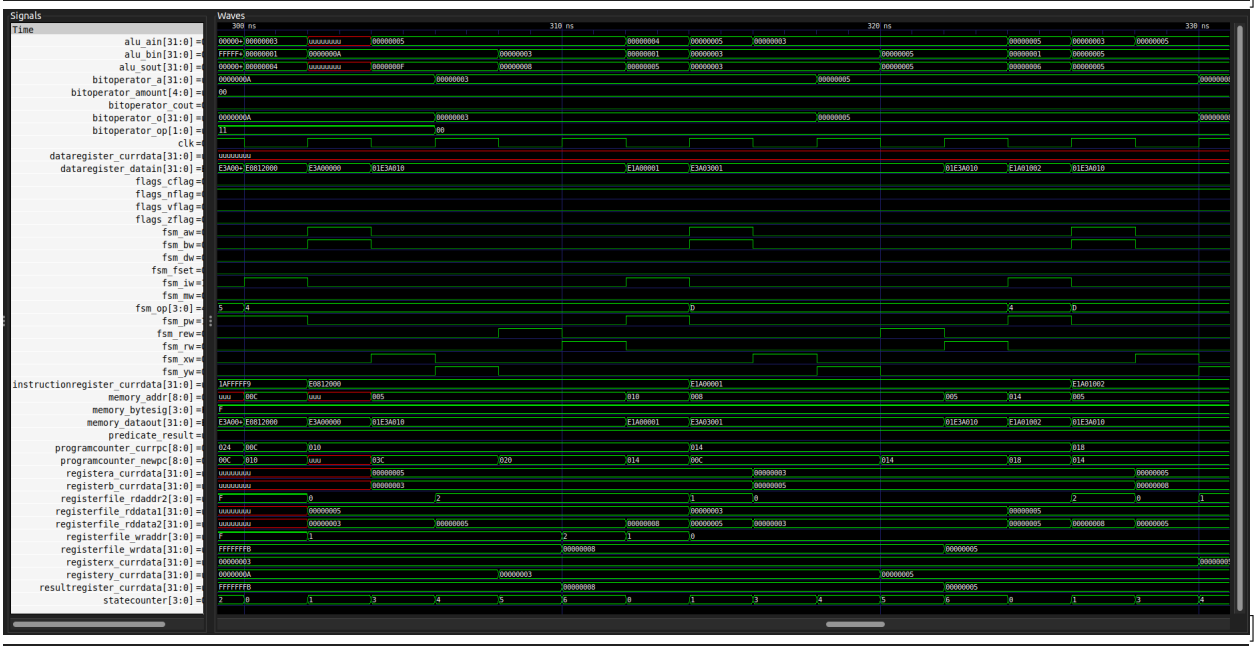

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
150 ns
160 ns
170 ns
180 ns

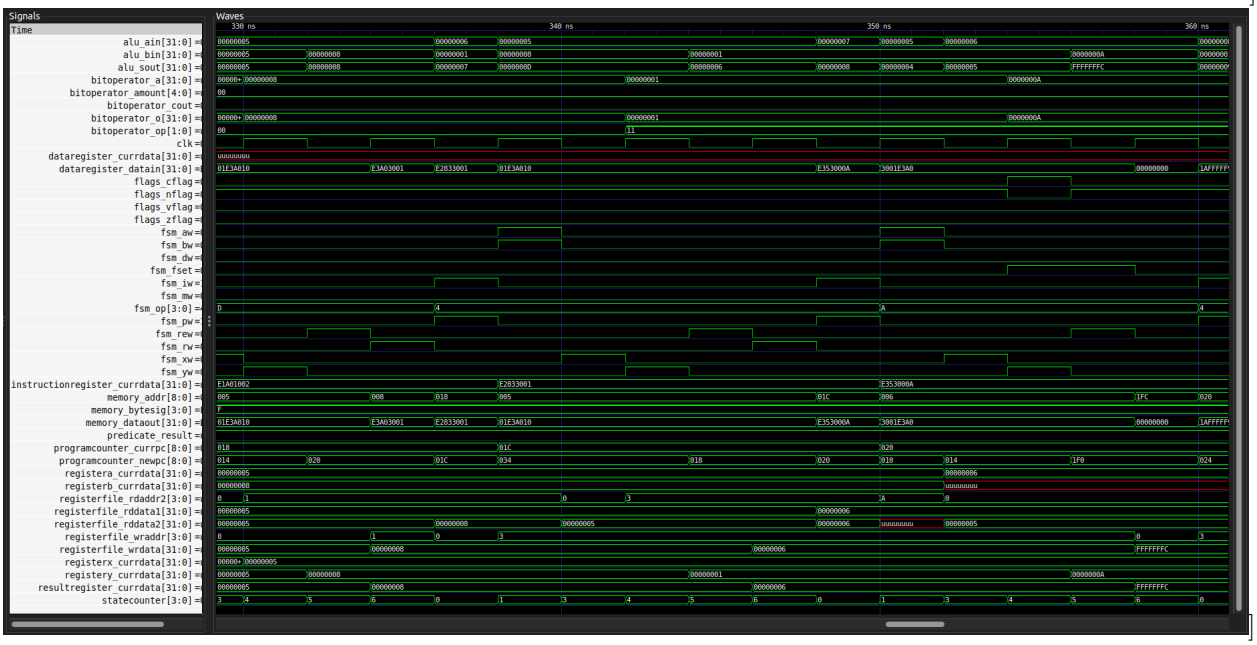

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
100 ns
190 ns
200 ns
210 ns

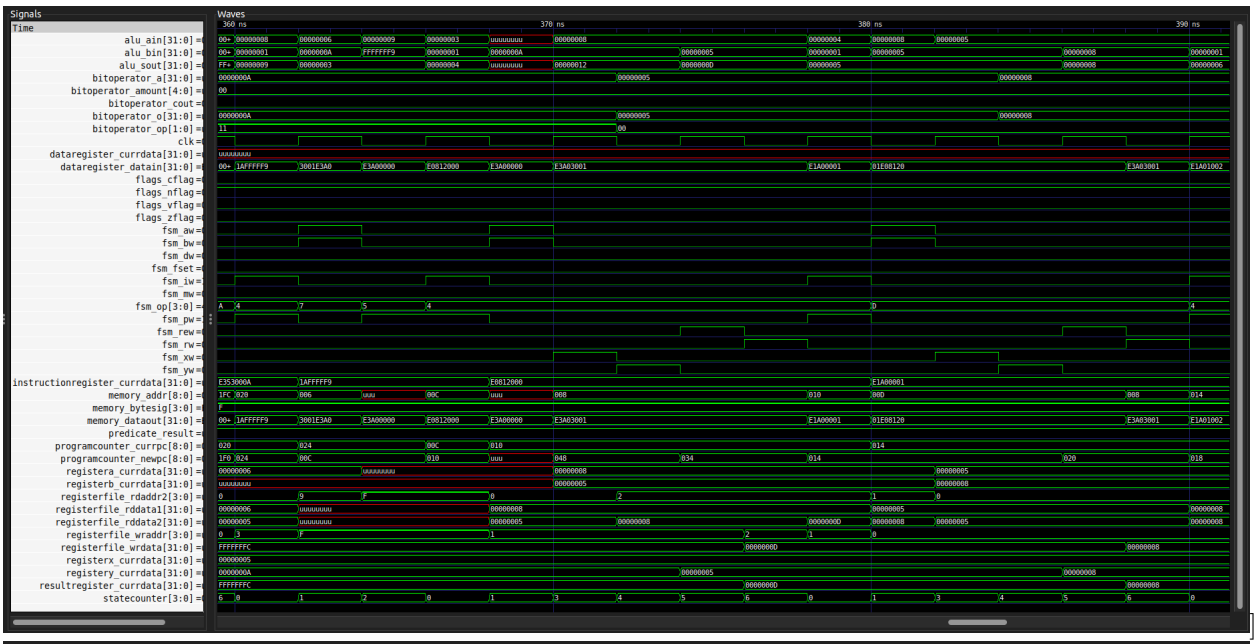

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
210 ns
220 ns
230 ns
240 ns

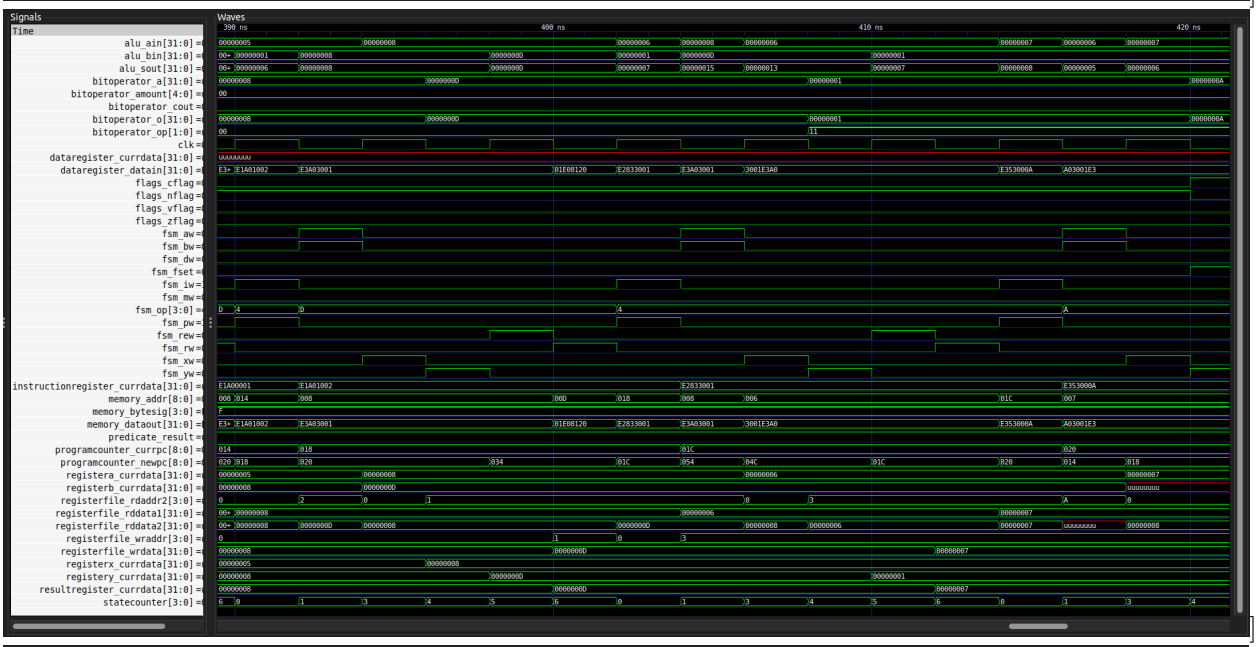

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
240 ns
250 ns
260 ns
270 ns

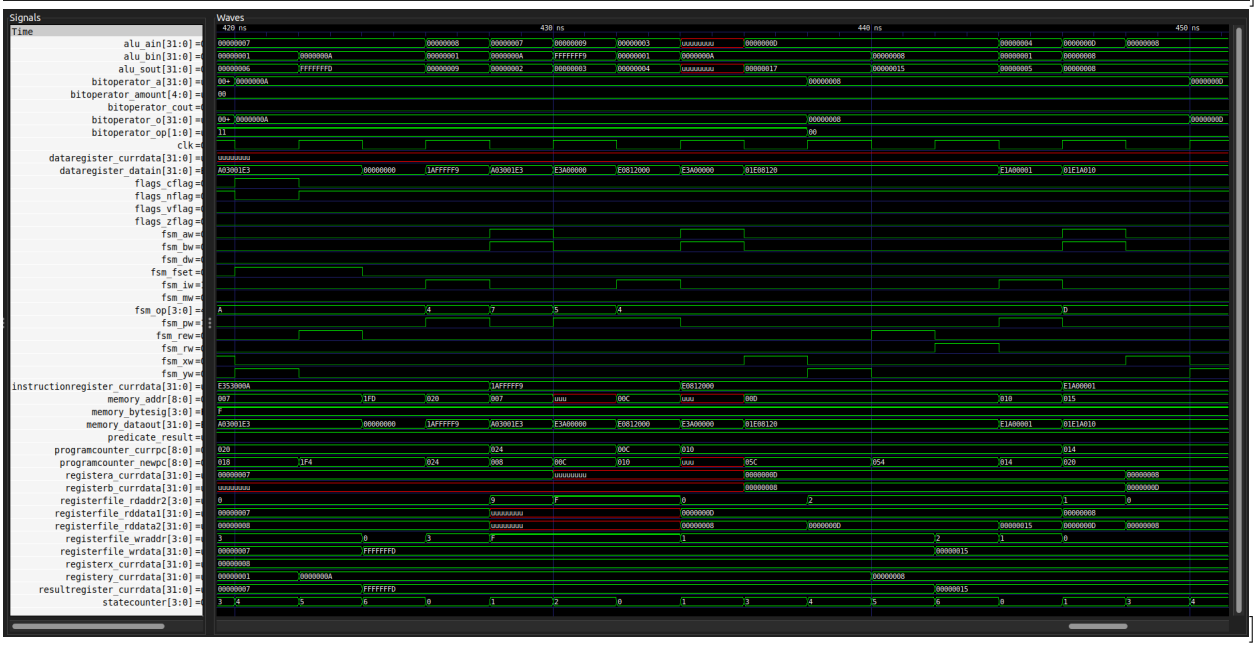

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
270 ns
280 ns
290 ns
300 ns

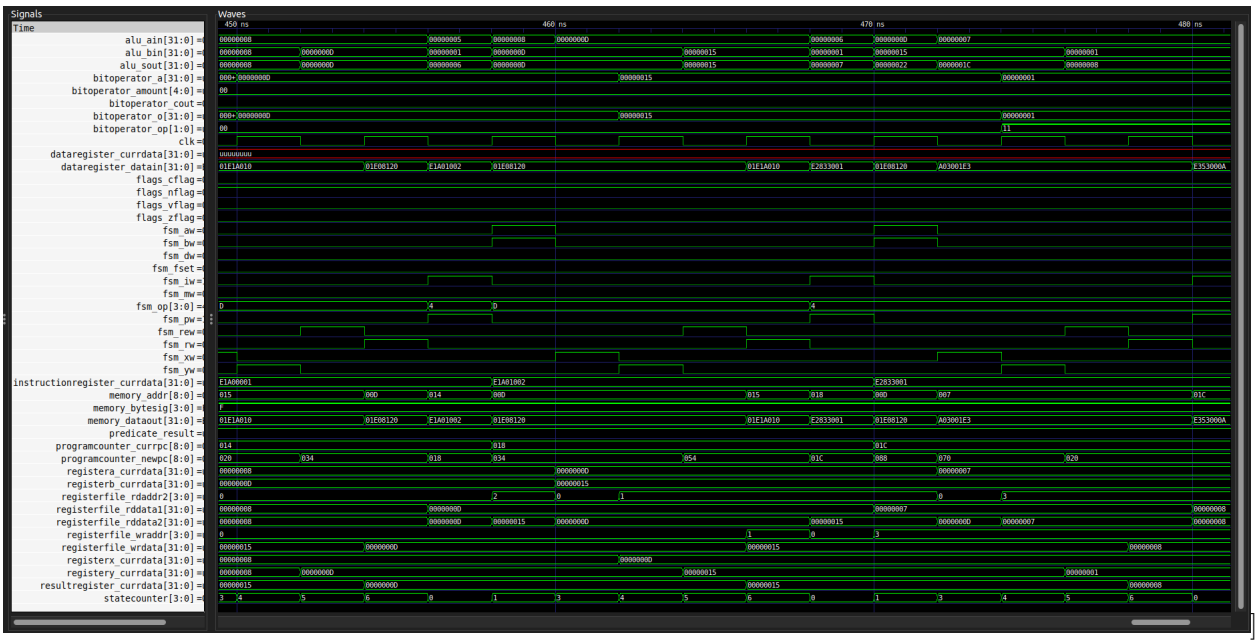

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
300 ns
310 ns
320 ns
330 ns

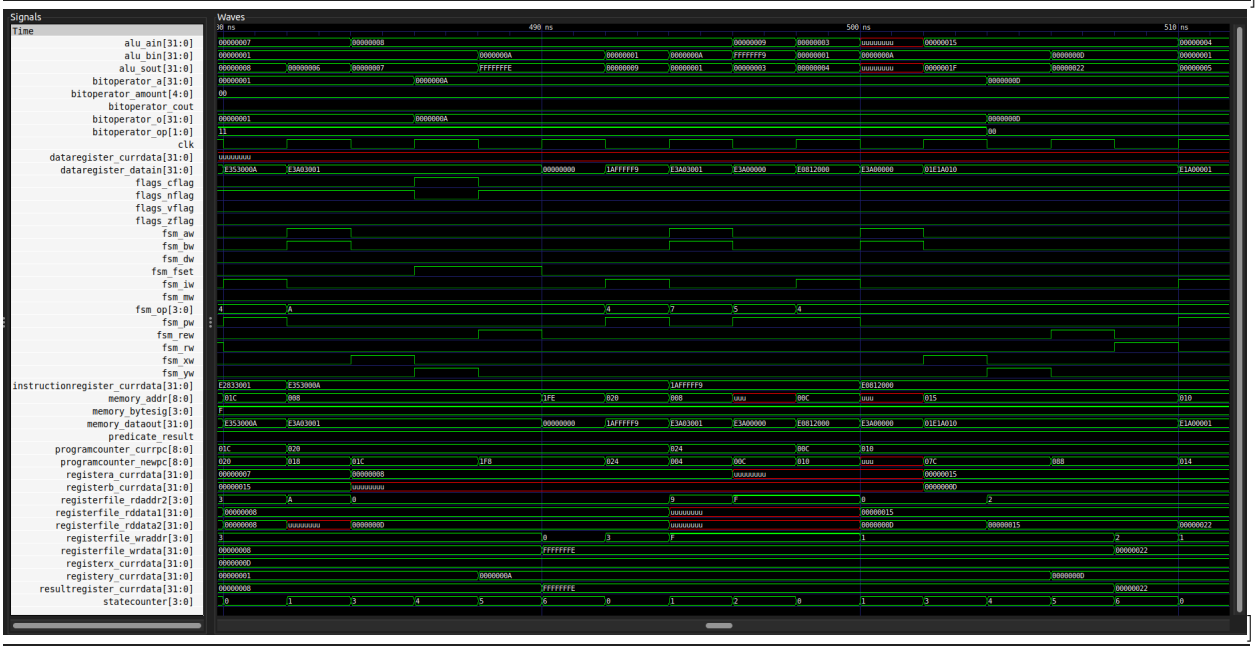

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
330 ns
340 ns
350 ns
360 ns

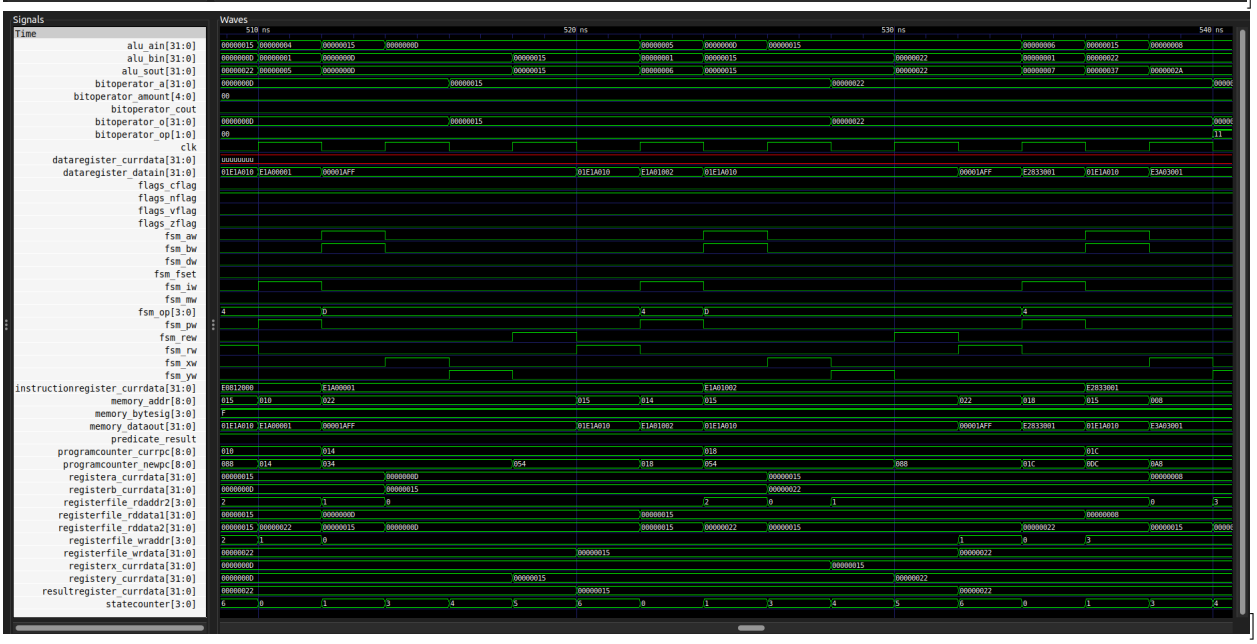

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
360 ns
370 ns
380 ns
390 ns

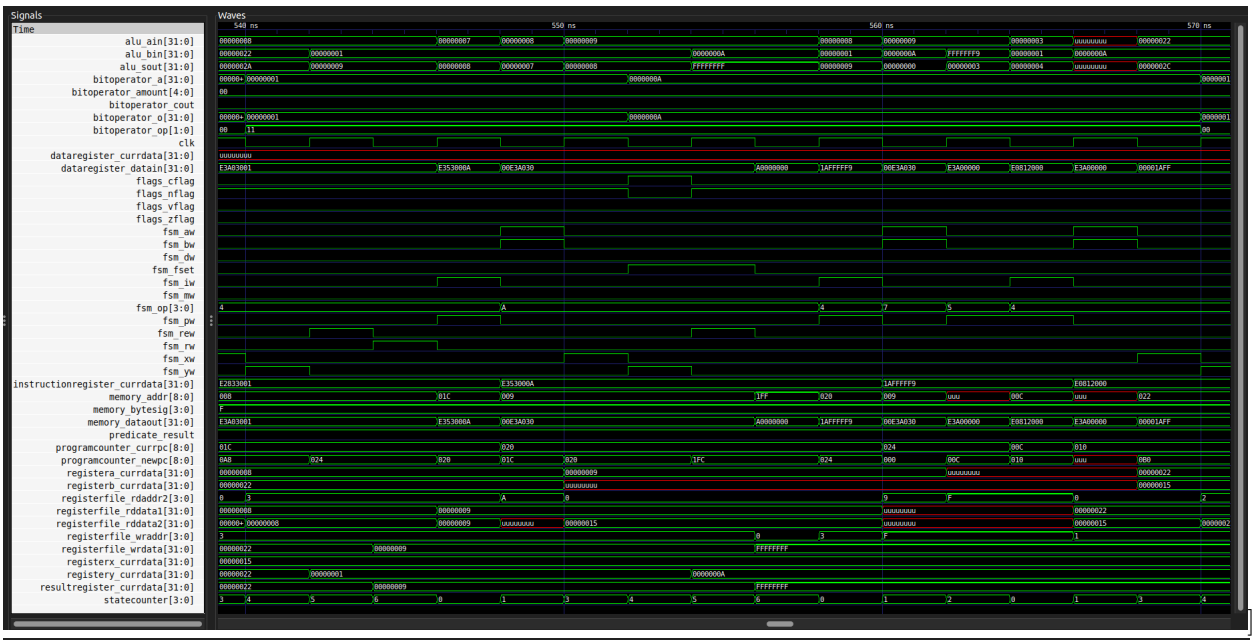

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
390 ns
400 ns
410 ns
420 ns

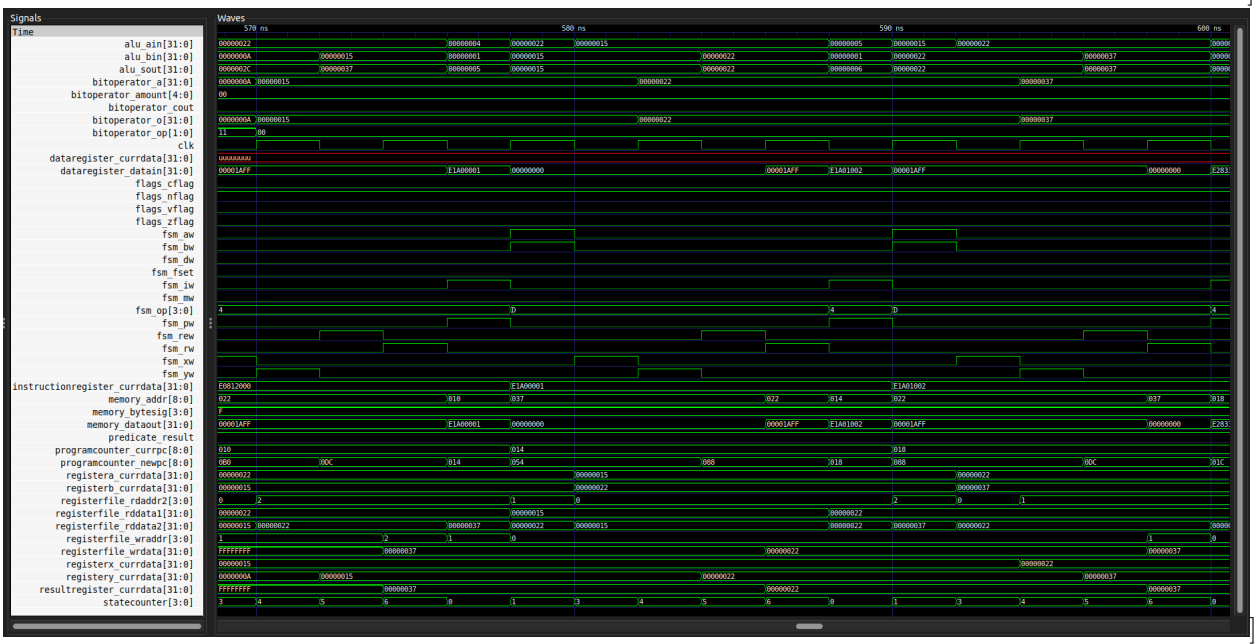

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
420 ns
430 ns
440 ns
450 ns

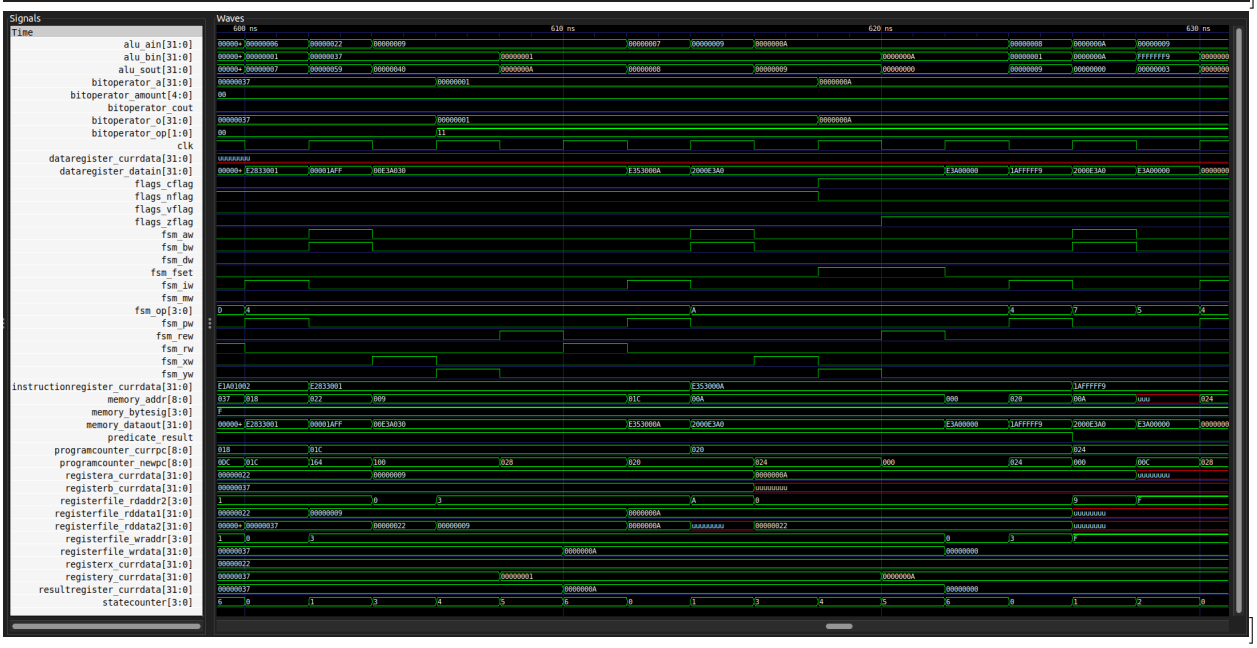

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
450 ns
460 ns
470 ns
480 ns

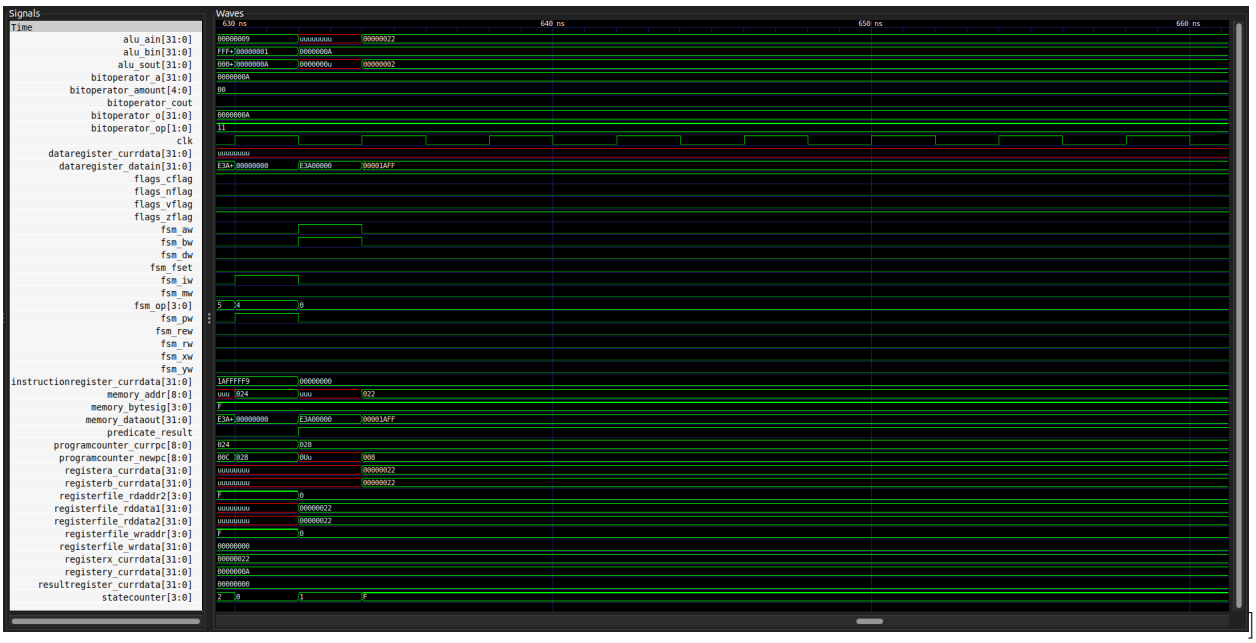

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
490 ns
500 ns
510 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
510 ns
520 ns
530 ns
540 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
540 ns
550 ns
560 ns
570 ns

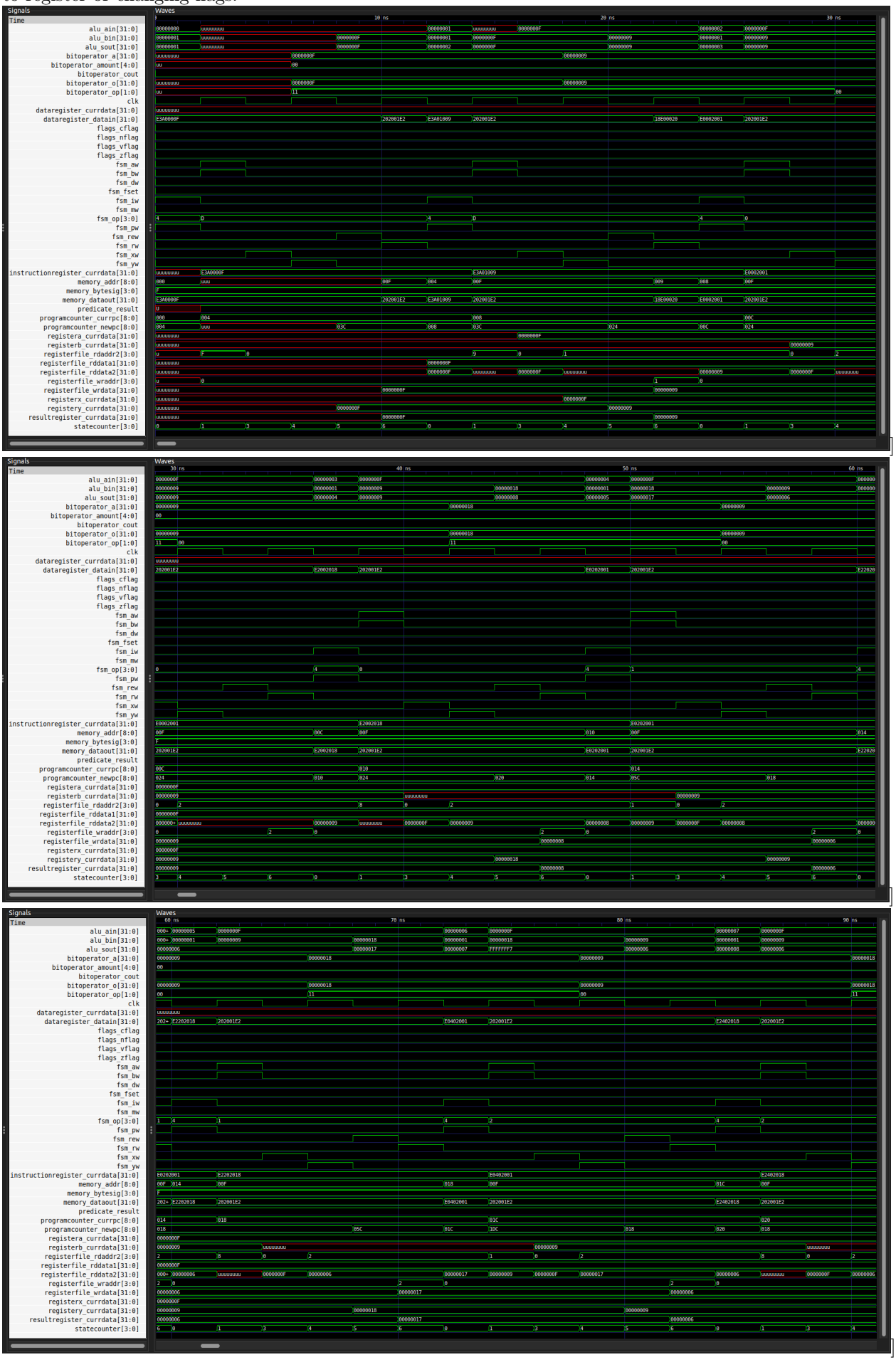

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
570 ns
580 ns
590 ns
600 ns

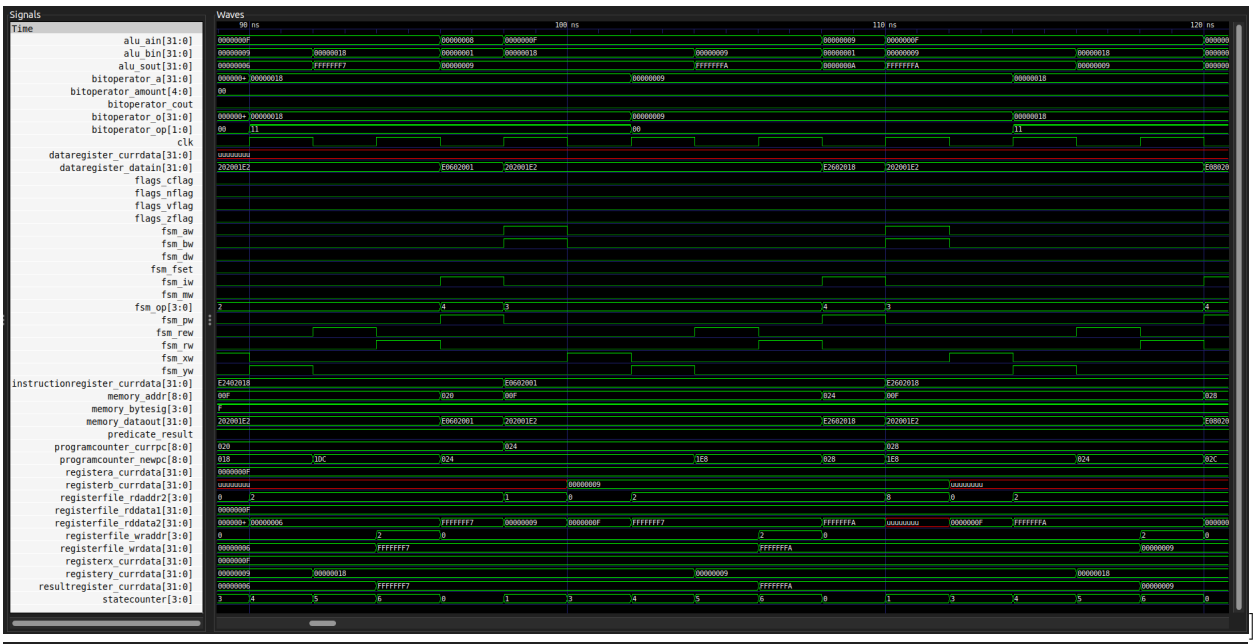

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
600 ns
610 ns
620 ns
630 ns

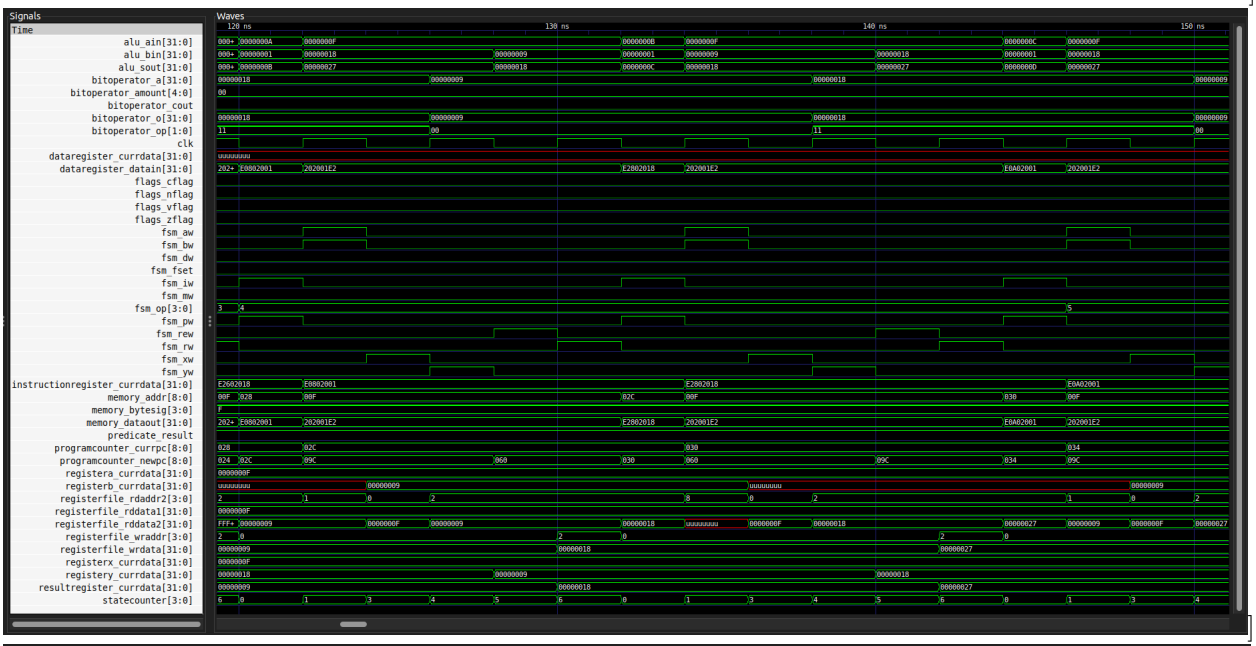


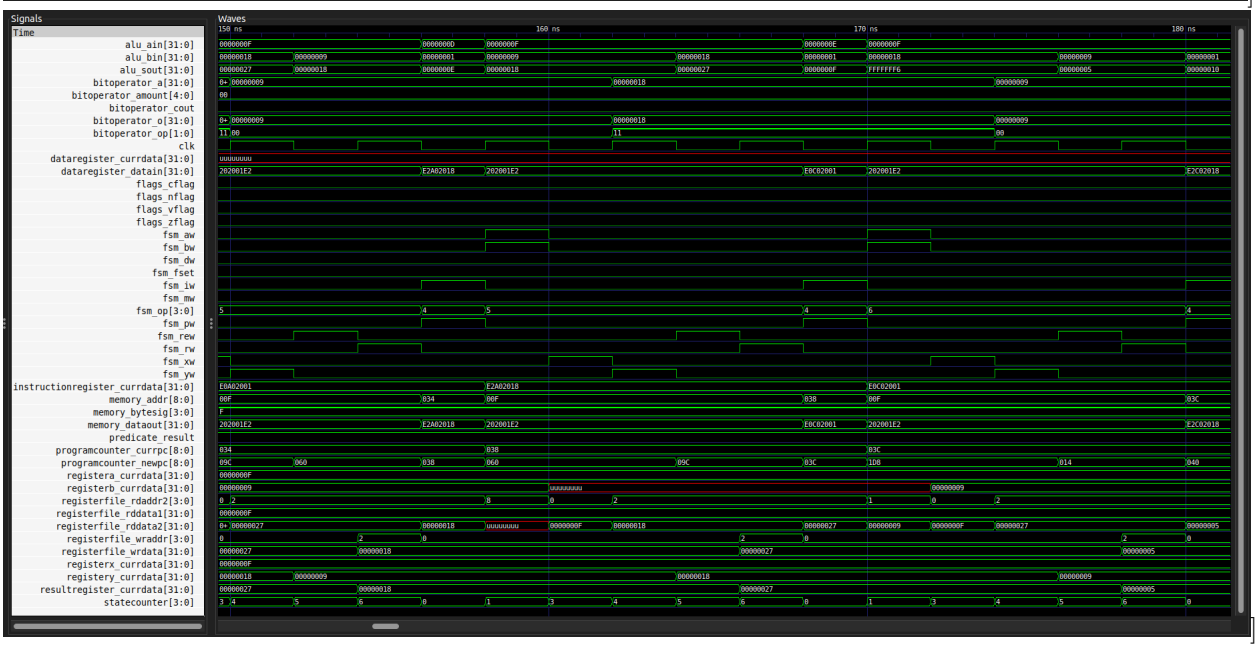
PROGRAM 4 to check all dp commands
signal mem: grid :=(
0 ⇒ X"E3A0000F", — mov r0, #15
1 ⇒ X"E3A01009", — mov r1, #9
2 ⇒ X"E0002001", — and r2, r0, r1 ⇒ r2 = 9
3 ⇒ X"E2002018", — and r2, r0, #24 ⇒ r2 = 8
4 ⇒ X"E0202001", — eor r2, r0, r1 ⇒ r2 = 6
5 ⇒ X"E2202018", — eor r2, r0, #24 ⇒ r2 = 23
6 ⇒ X"E0402001", — sub r2, r0, r1 ⇒ r2 = 6
7 ⇒ X"E2402018", — sub r2, r0, #24 ⇒ r2 = -9
8 ⇒ X"E0602001", — rsb r2, r0, r1 ⇒ r2 = -6
9 ⇒ X"E2602018", — rsb r2, r0, #24 ⇒ r2 = 9
10 ⇒ X"E0802001", — add r2, r0, r1 ⇒ r2 = 24
11 ⇒ X"E2802018", — add r2, r0, #24 ⇒ r2 = 39
12 ⇒ X"E0A02001", — adc r2, r0, r1 ⇒ r2 = 24
13 ⇒ X"E2A02018", — adc r2, r0, #24 ⇒ r2 = 39
14 ⇒ X"E0C02001", — sbc r2, r0, r1 ⇒ r2 = 5
15 ⇒ X"E2C02018", — sbc r2, r0, #24 ⇒ r2 = -10
16 ⇒ X"E0E02001", — rsc r2, r0, r1 ⇒ r2 = -7
17 ⇒ X"E2E02018", — rsc r2, r0, #24 ⇒ r2 = 8
18 ⇒ X"E1102001", — tst r0, r1
19 ⇒ X"E3102018", — tst r0, #24
20 ⇒ X"E1302001", — teq r0, r1
21 ⇒ X"E3302018", — teq r0, #24
22 ⇒ X"E1502001", — cmp r0, r1
23 ⇒ X"E3502018", — cmp r0, #24
24 ⇒ X"E1702001", — cmn r0, r1
25 ⇒ X"E3702018", — cmn r0, #24
26 ⇒ X"E1802001", — orr r2, r0, r1 ⇒ r2 = 15
27 ⇒ X"E3802018", — orr r2, r0, #24 ⇒ r2 = 31
28 ⇒ X"E1A02001", — mov r2, r1 ⇒ r2 = 9
29 ⇒ X"E3A02018", — mov r2, #24 ⇒ r2 = 24
30 ⇒ X"E1C02001", — bic r2, r0, r1 ⇒ r2 = 6
31 ⇒ X"E3C02018", — bic r2, r0, #24 ⇒ r2 = 31
32 ⇒ X"E1E02001", — mvn r2, r1 ⇒ r2 = -10
33 ⇒ X"E3E02018", — mvn r2, #24 ⇒ r2 = -24
others ⇒ X"00000000" — end
);
This program is meant to check whether all dp instructions are working properly or not
And Following EPWave shows everything is working properly being it operation or access values from or

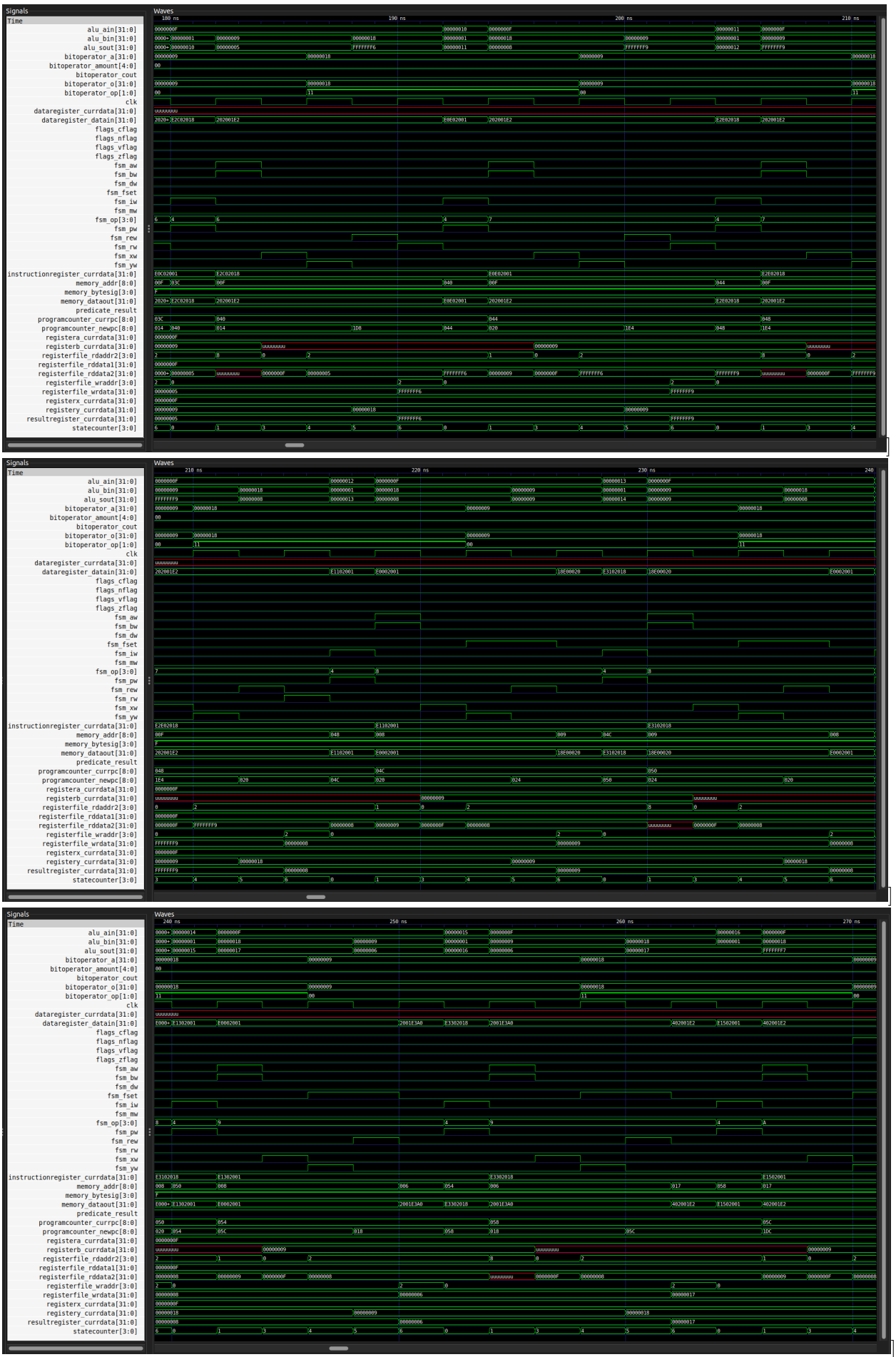
to register or changing flags.

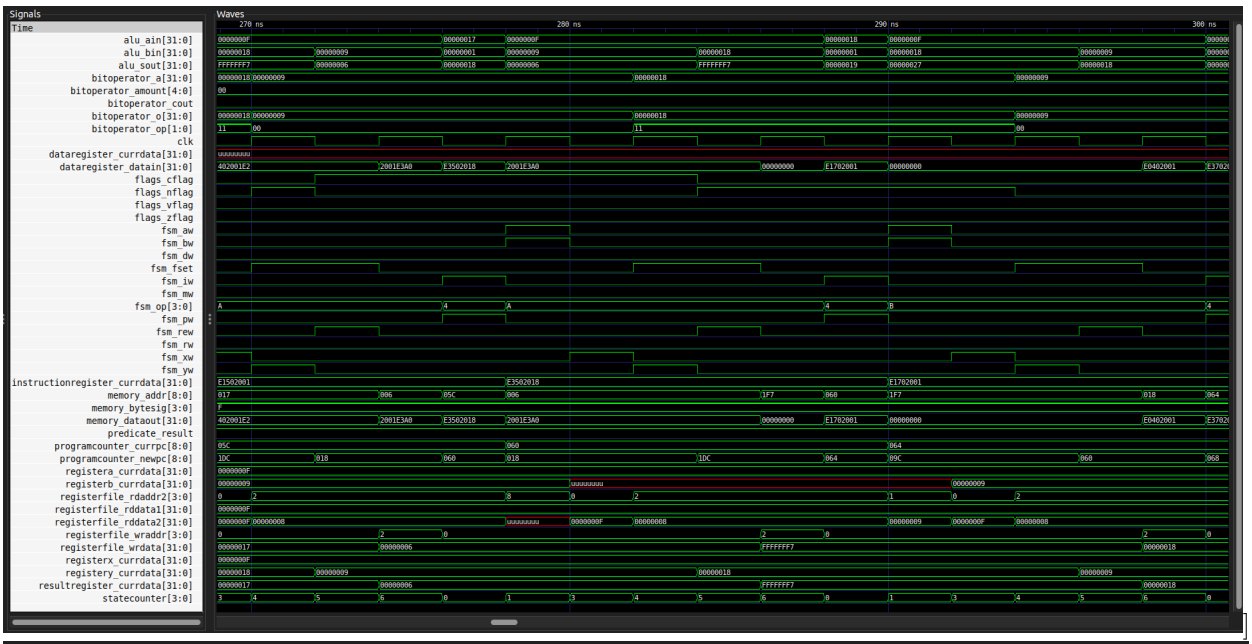


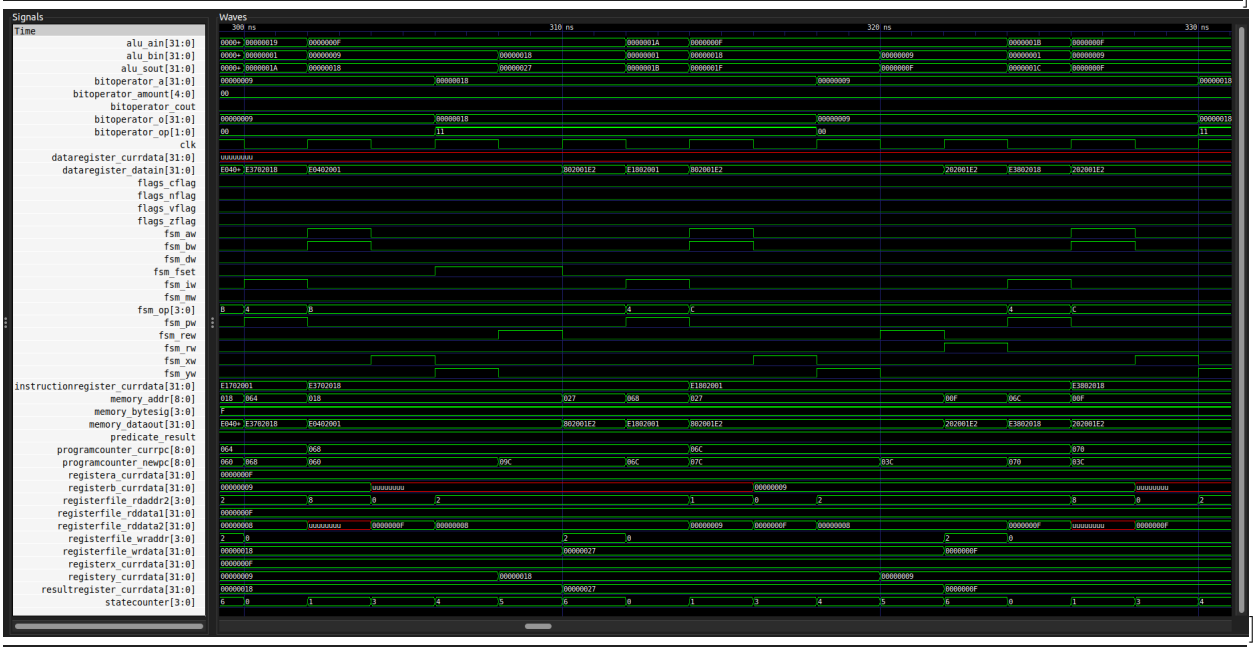

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
90 ns
100 ns
110 ns
120 ns

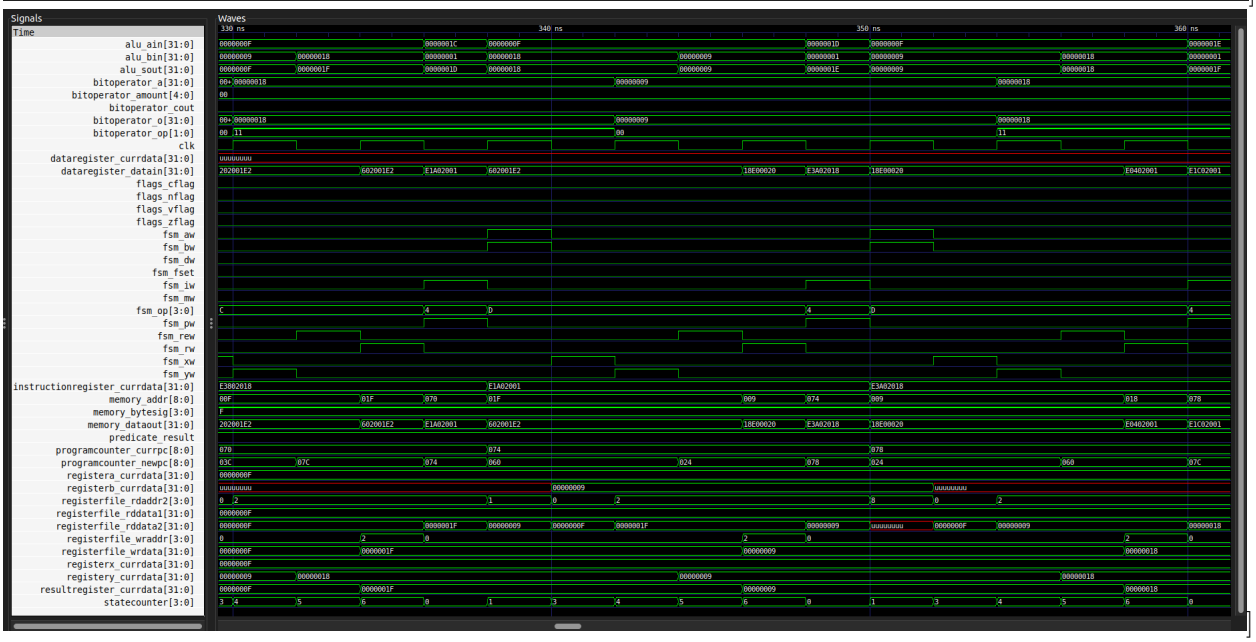

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
120 ns
130 ns
140 ns
150 ns

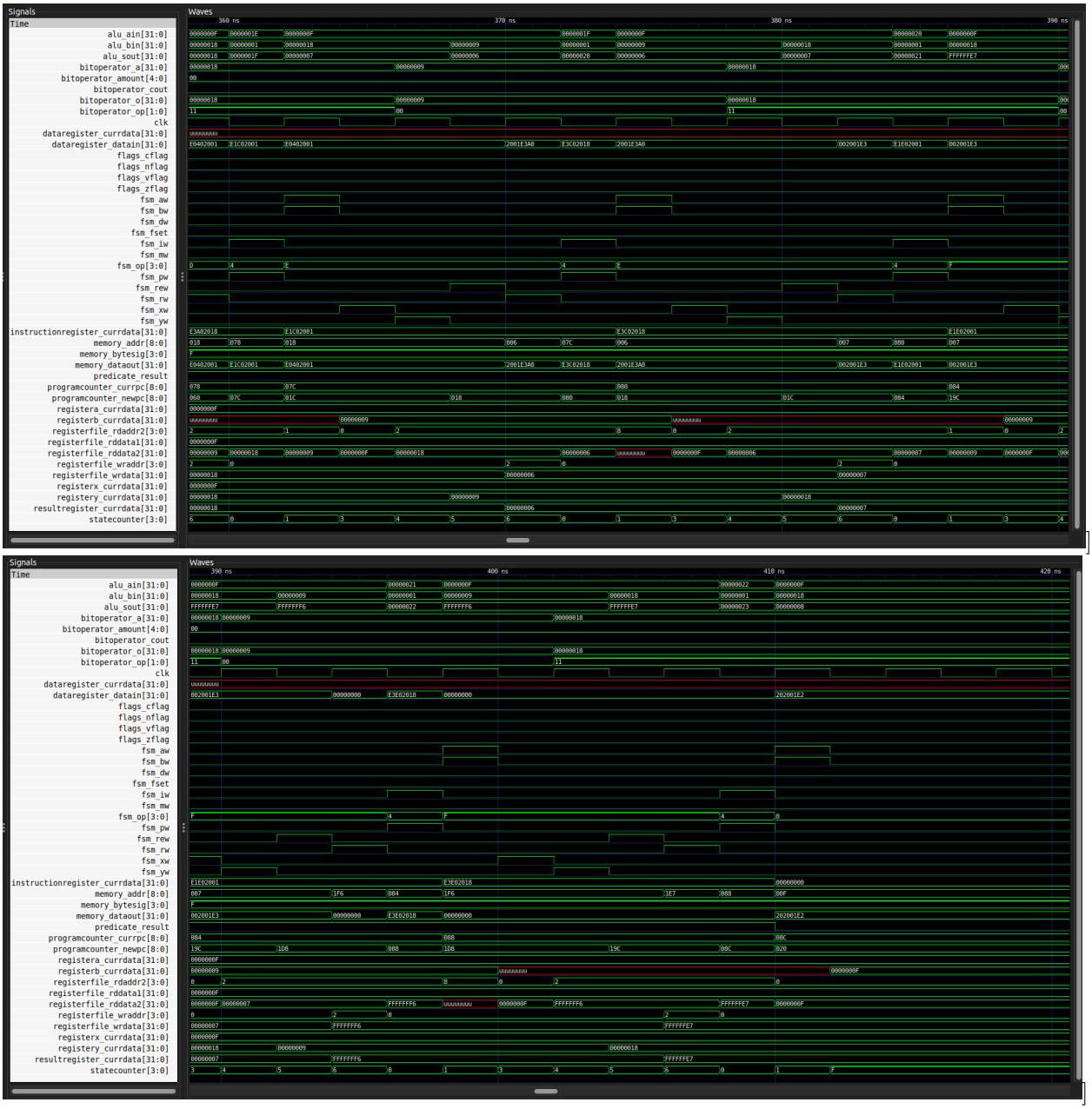

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
150 ns
160 ns
170 ns
180 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
180 ns
190 ns
200 ns
210 ns
220 ns
230 ns
240
240 ns
250 ns
260 ns
270 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
270 ns
280 ns
290 ns
300 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
300 ns
310 ns
320 ns
330 ns

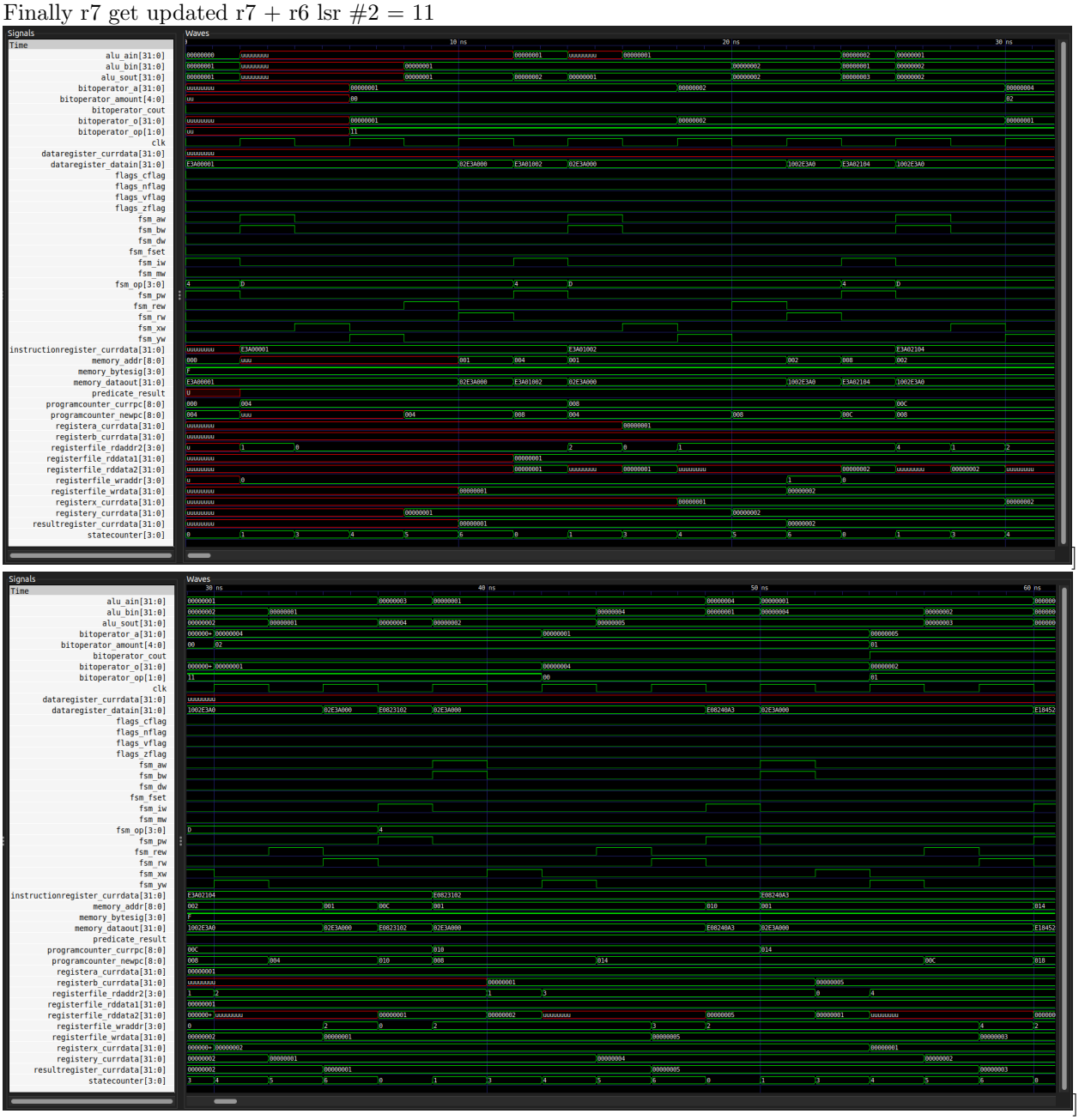

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
330 ns
340 ns
350 ns
360 ns

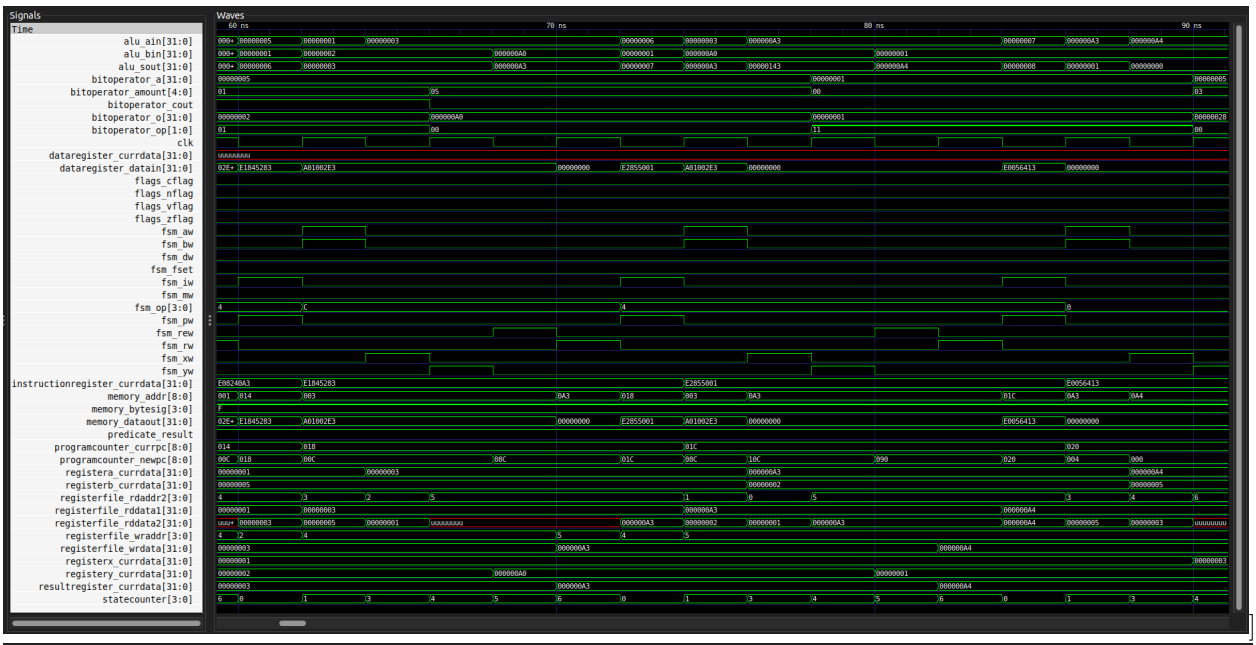


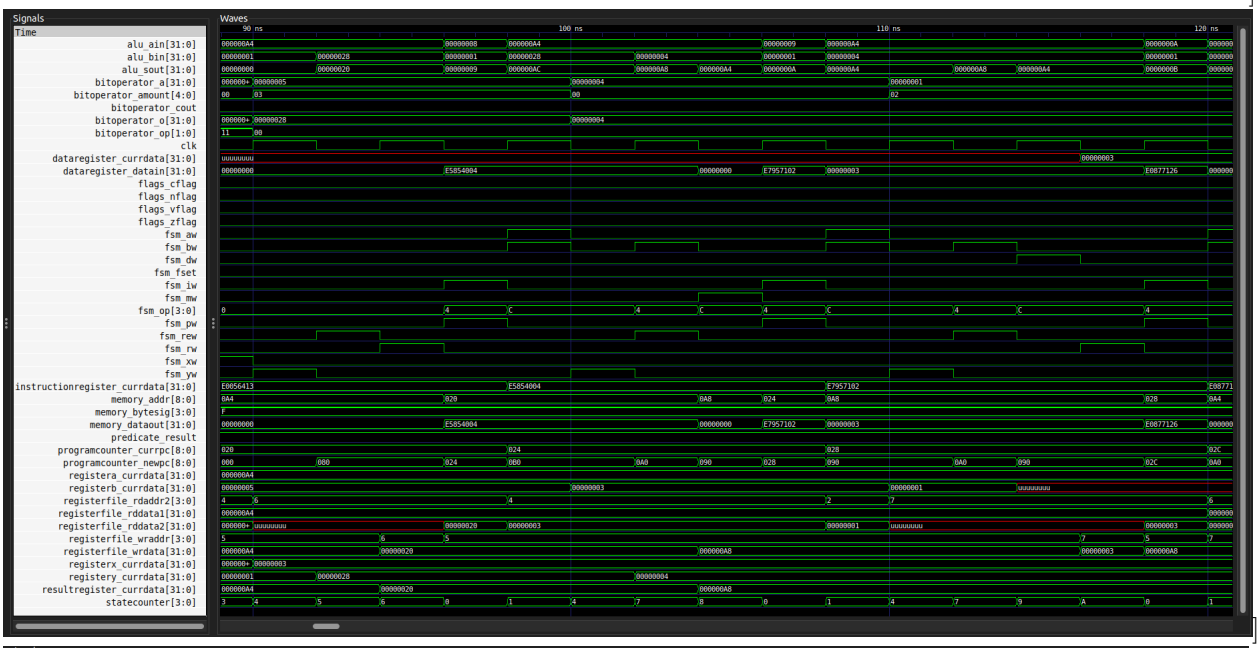
PROGRAM 5:
signal mem: grid := (
0 ⇒ X"E3A00001", – mov r0, #1 ⇒ r0 = 1
1 ⇒ X"E3A01002", – mov r1, #2 ⇒ r1 = 2
2 ⇒ X"E3A02104", – mov r2, #4, #2 ⇒ r2 = 1
3 ⇒ X"E0823102", – add r3, r2, r2, lsl #2 ⇒ r3 = 5
4 ⇒ X"E08240A3", – add r4, r2, r3, lsr #1 ⇒ r4 = 3
5 ⇒ X"E1845283", – orr r5, r4, r3, lsl #5 ⇒ r5 = 163
6 ⇒ X"E2855001", – add r5, r5, #1 ⇒ r5 = 164
7 ⇒ X"E0056413", – and r6, r5, r3, lsl r4 ⇒ r6 = 32
8 ⇒ X"E5854004", – str r4, [r5, #4] ⇒ mem[168] = 3
9 ⇒ X"E7957102", – ldr r7, [r5, r2, lsl #2] ⇒ r7 = mem[168] = 3
10 ⇒ X"E0877126", – add r7, r7, r6, lsr #2 ⇒ r7 = 11
others ⇒ X"00000000" – .end
);

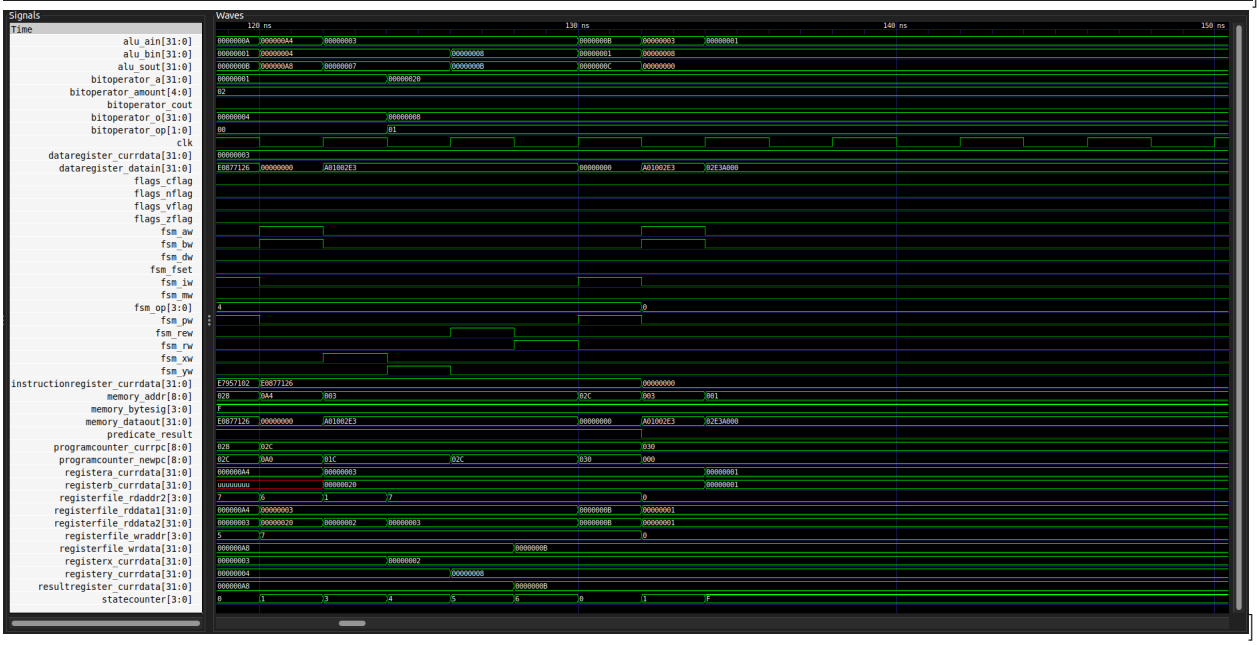
Following shows EPWave signal for processor tb as it correctly outputs the result at given memory location and by different shifting.
r0 contains 1, r1 contains 2, r2 contains 1
r3 get 5, r4 gets 3, r5 is assigned 163 and the updated to 164, r6 is assigned 32.

Then data present at r4 = 4 is stored in mem[r5 + 4] = mem[168] ⇐ 3
Then data from mem[168] is retrieved in r7 using left shift
Finally r7 get updated r7 + r6 lsr #2 = 11

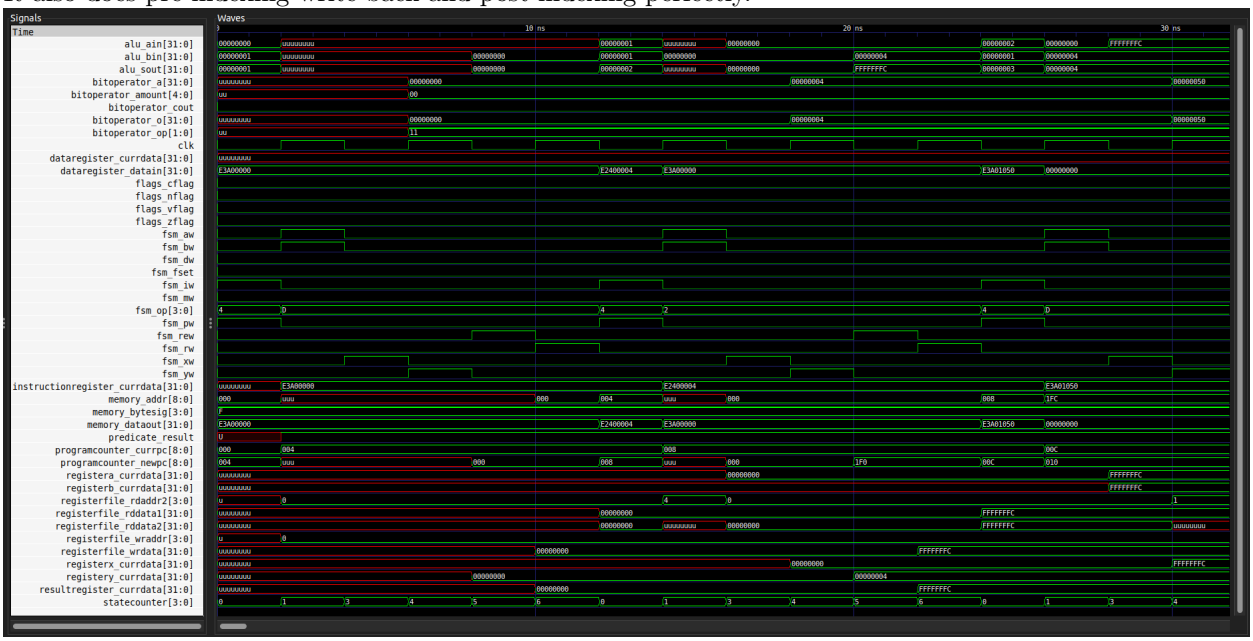


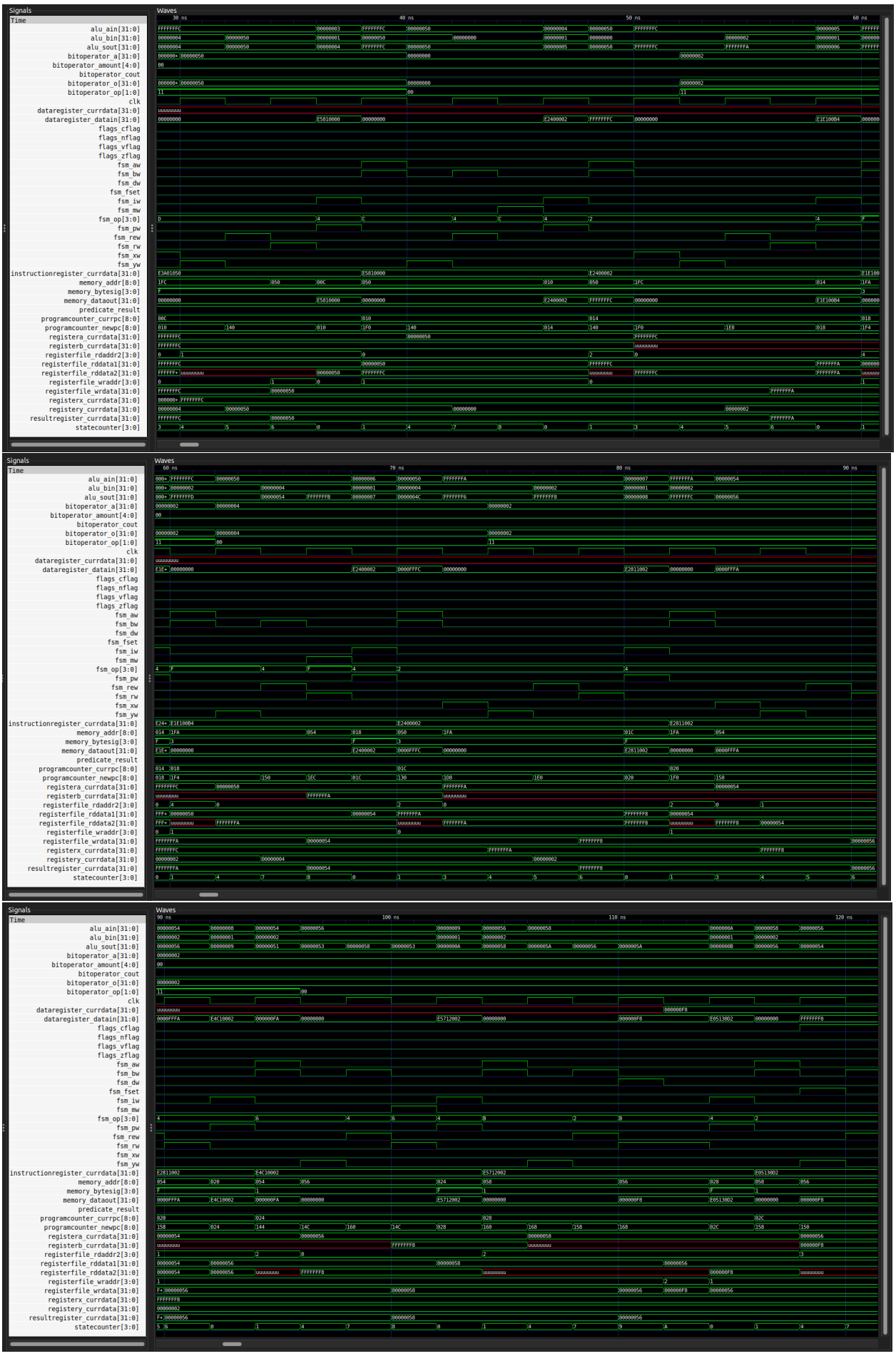

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
60 ns
70 ns
80 ns
90 ns

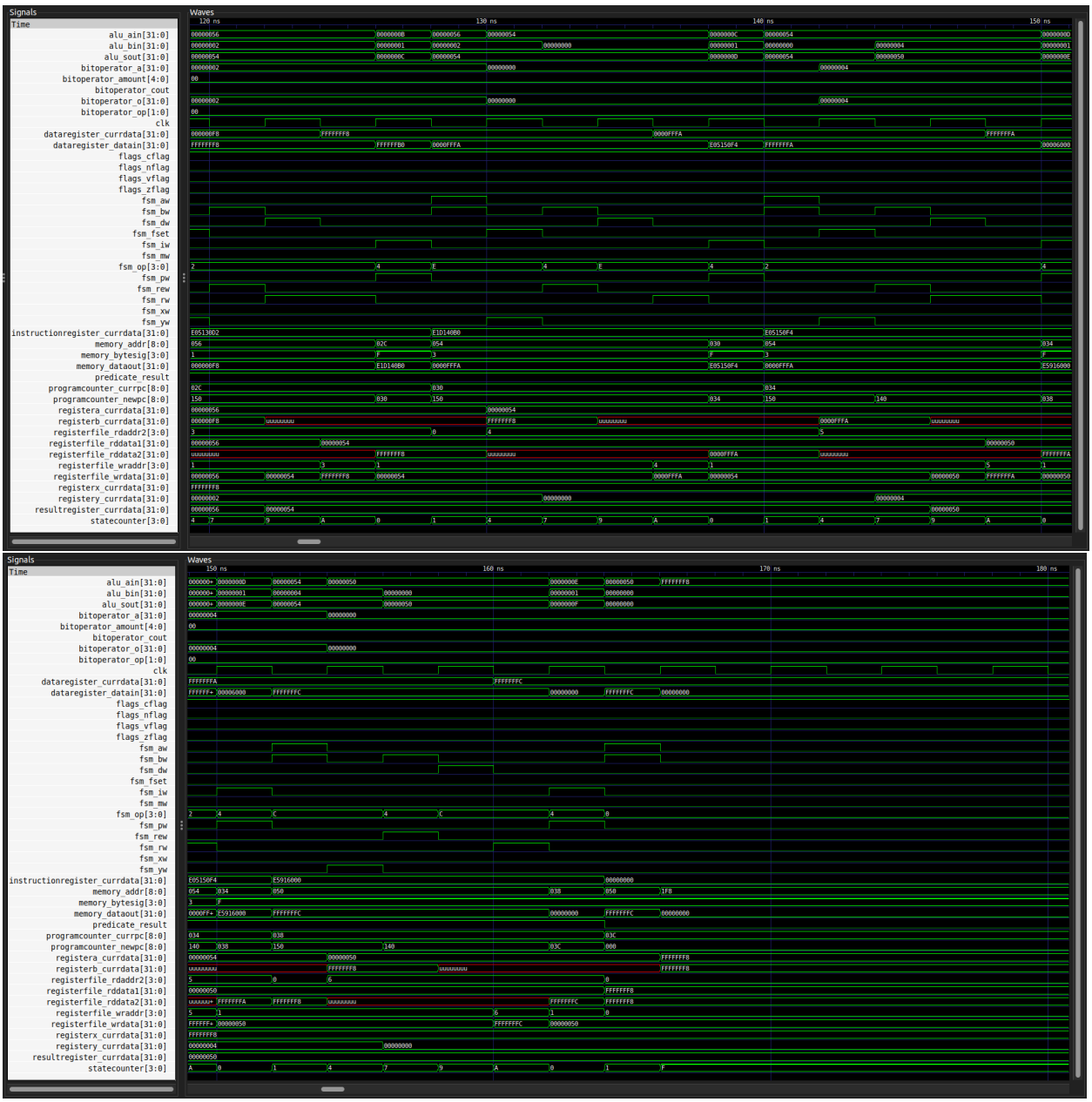

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
90 ns
100 ns
110 ns
120 ns


Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
120 ns
130 ns
140 ns
150 ns

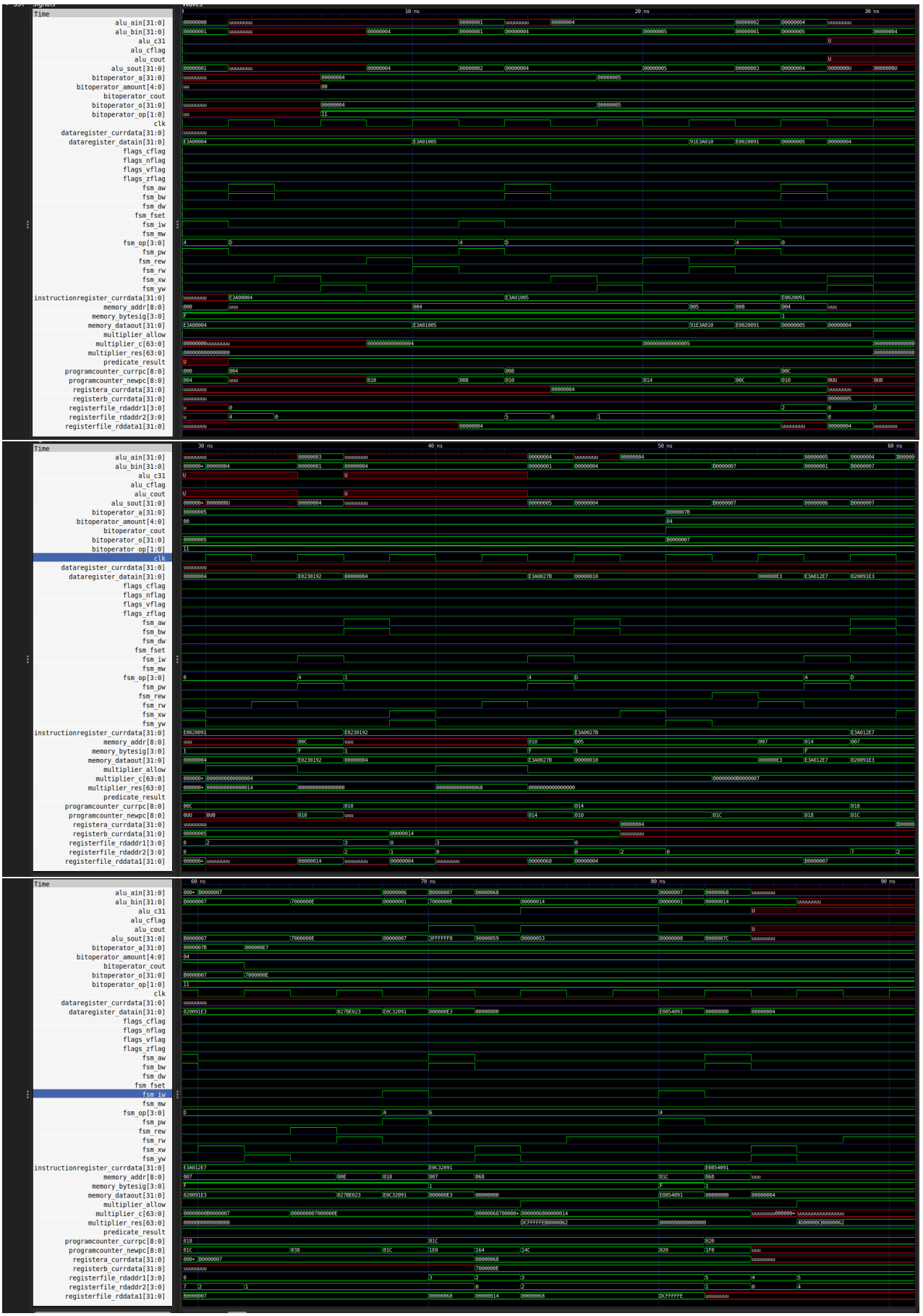
PROGRAM 6:
signal mem: grid := (
0 ⇒ X”E3A00000”, – mov r0, #0 ⇒ r0 = 0
1 ⇒ X”E2400004”, – sub r0, r0, #4 ⇒ r0 = -4
2 ⇒ X”E3A01050”, – mov r1, #80 ⇒ r1 = 80
3 ⇒ X”E5810000”, – str r0, [r1] ⇒ mem[83 downto 80] = X”FFFFFFFC”
4 ⇒ X”E2400002”, – sub r0, r0, #2 ⇒ r0 = -6
5 ⇒ X”E1E100B4”, – strh r0, [r1, #4]! ⇒ r1 = 84, mem[85 downto 84] = X”FFFA”
6 ⇒ X”E2400002”, – sub r0, r0, #2 ⇒ r0 = -8
7 ⇒ X”E2811002”, – add r1, r1, #2 ⇒ r1 = 86
8 ⇒ X”E4C10002”, – strb r0, [r1], #2 ⇒ mem[86] = X”F8”, r1 = 88
9 ⇒ X”E5712002”, – ldrb r2, [r1, #-2]! ⇒ r1 = 86, r2 = mem[86] =¿ r2 = X”000000F8”
10 ⇒ X”E05130D2”, – ldrsb r3, [r1], #-2 ⇒ r2 = mem[86] =¿ r2 = X”FFFFFFF8”, r1 = 84
11 ⇒ X”E1D140B0”, – ldrh r4, [r1] ⇒ r4 = mem[85 downto 84] = X”0000FFFA”
12 ⇒ X”E05150F4”, – ldrsh r5, [r1], #-4 ⇒ r5 = mem[85 downto 84] = X”FFFFFFFA”, r1 = 80
13 ⇒ X”E5916000”, – ldr r6, [r1] ⇒ r6 = mem[83 downto 80] = X”FFFFFFFC”
others ⇒ X”00000000”
);
Following show EPWave signal for the processor tb as it correctly loads word(X”FFFFFFFC”), half-word(”X”FFFA”) and byte(X”F8”) at correct location of memory and the correctly retrieves byte(X”000000F8”), signed byte(X”FFFFFFF8”), half-word(X”0000FFFA”), signed half-word(X”FFFFFFFA”), word(X”FFFFFFFC”). It also does pre-indexing write back and post indexing perfectly.

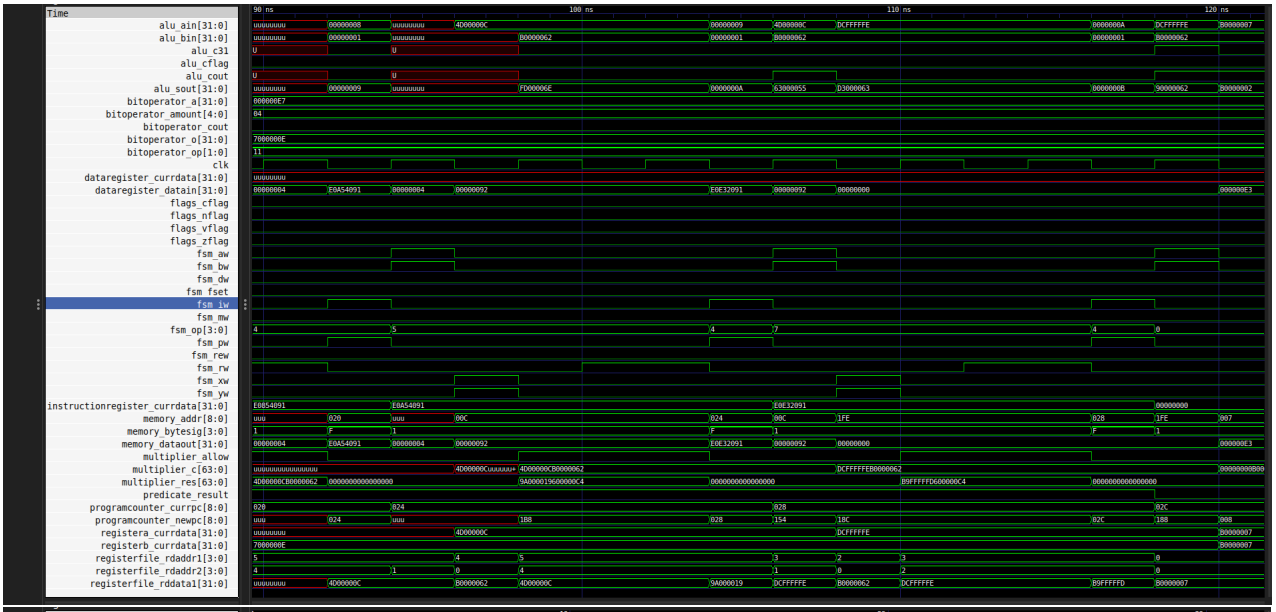


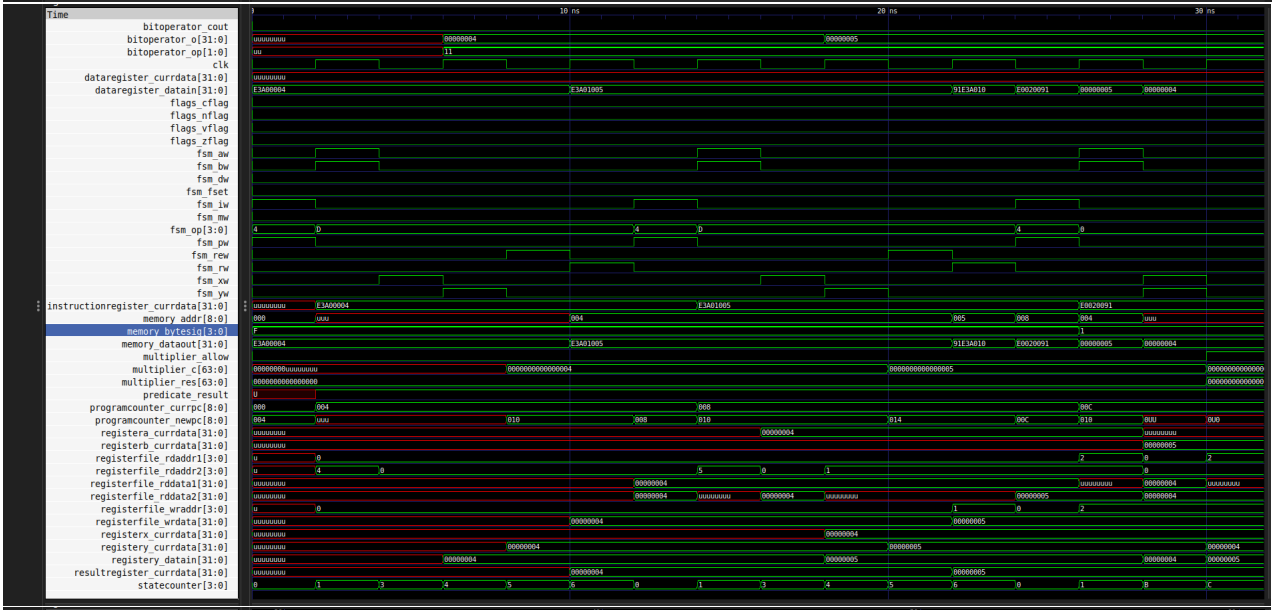

Signals
Time
alu_ain[31:0]
alu_bin[31:0]
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
Waves
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns

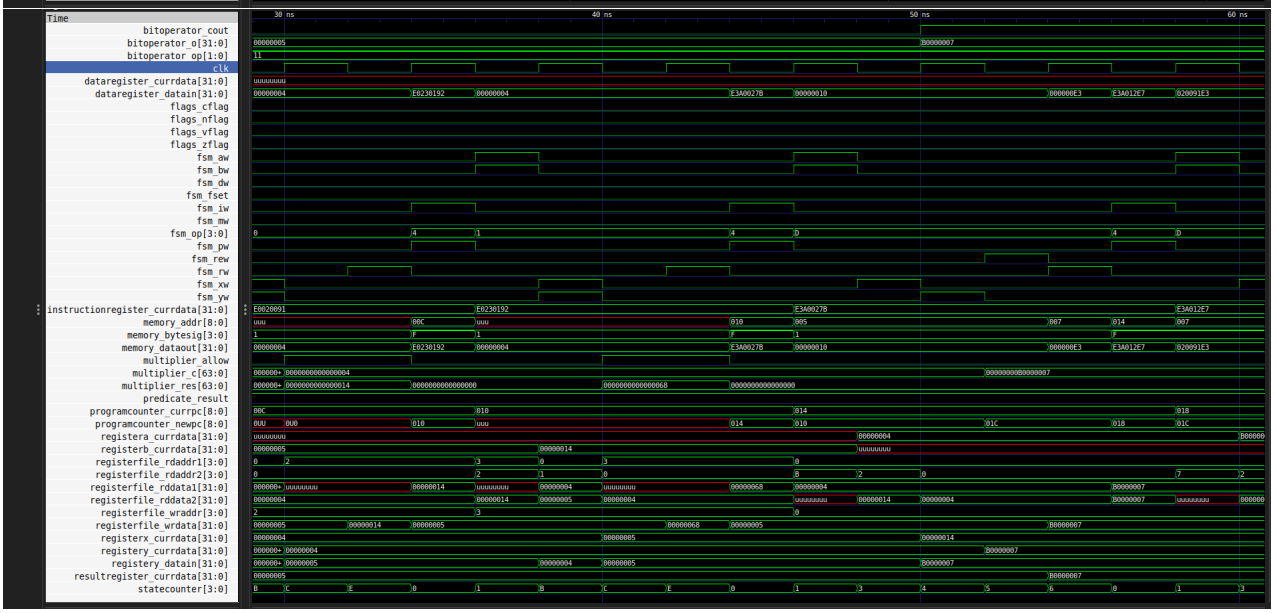


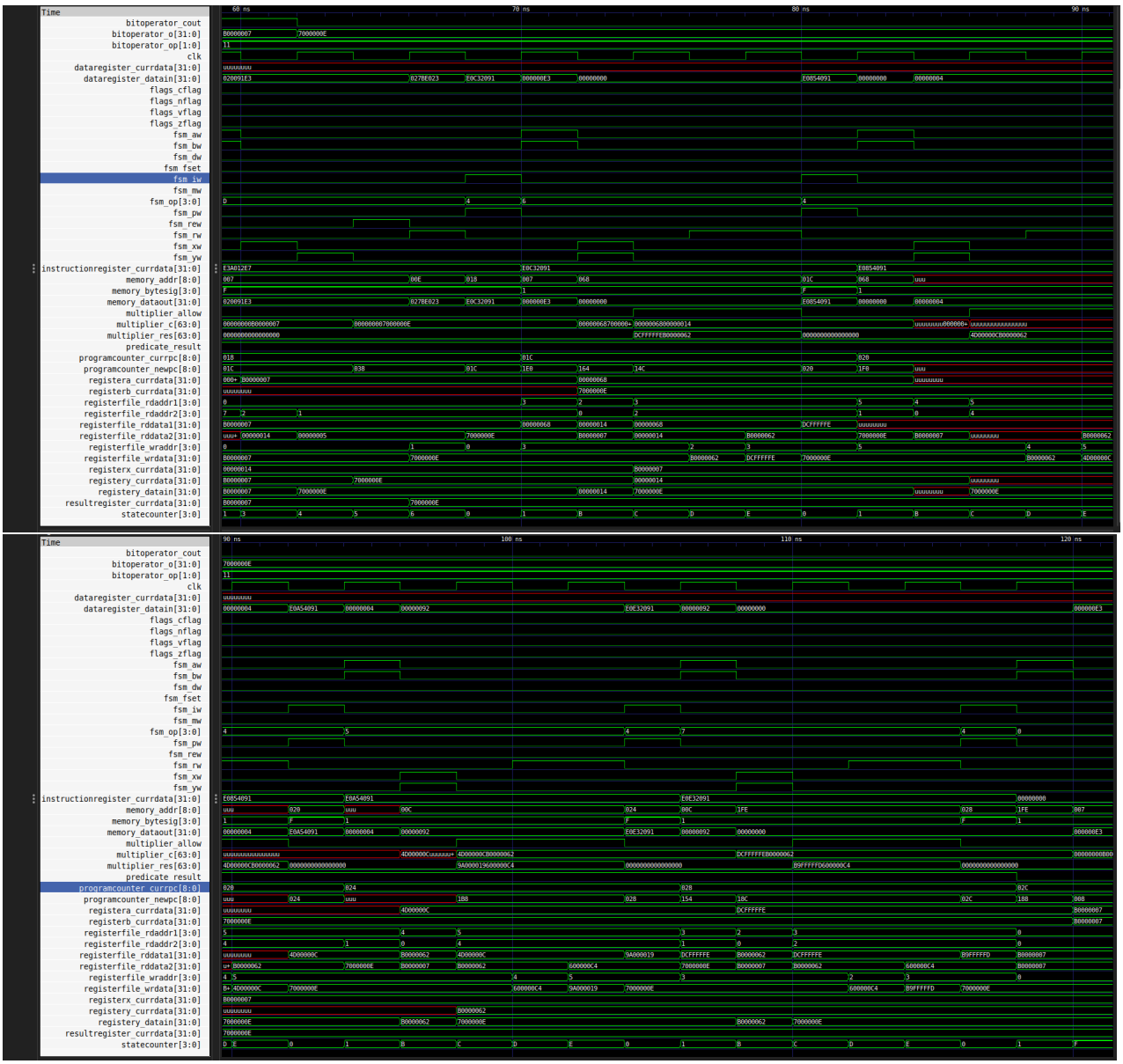
– PROGRAM 7
signal mem: grid := (
0 ⇒ X”E3A00004”, – mov r0, #4 ⇒ r0 = 4
1 ⇒ X”E3A01005”, – mov r1, #5 ⇒ r5 = 5
2 ⇒ X”E0020091”, – mul r2, r1, r0 ⇒ r2 = 4 * 5 = 20
3 ⇒ X”E0230192”, – mla r3, r2, r1, r0 ⇒ r3 = 20 * 5 + 4 = 104
4 ⇒ X”E3A0027B”, – mov r0, #123, #4 ⇒ r0 = X”B0000007”
5 ⇒ X”E3A012E7”, – mov r1, #231, #4 ⇒ r1 = X”7000000E”
6 ⇒ X”E0C32091”, – smull r2, r3, r1, r0 ⇒ r3r2 = signed(r1) * signed(r0) = X”DCFFFFFEB0000062”
7 ⇒ X”E0854091”, – umull r4, r5, r1, r0 ⇒ r5r4 = unsigned(r1) * unsigned(r0) = X”4D00000CB0000062”
8 ⇒ X”E0A54091”, – umlal r4, r5, r1, r0 ⇒ r5r4 = unsigned(r1) * unsigned(r0) + unsigned(r5r4) = X”9A000019600000C4”
9 ⇒ X”E0E32091”, – smlal r2, r3, r1, r0 ⇒ r3r2 = signed(r1) * signed(r0) + signed(r3r2) = X”89FFFFFD600000C4
others ⇒ X”00000000”
);
Following show EPWave signal for the processor tb as it correctly does small multiplication, small multiplication with accumulation, long signed and unsigned multiplication with as well as without accumulation.


Time
alu_ain[31:0]
alu_bin[31:0]
alu_c31
alu_cflag
alu_cout
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
multiplier_allow
multiplier_c[63:0]
multiplier_res[63:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr1[3:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns


Time
alu_ain[31:0]
alu_bin[31:0]
alu_c31
alu_cflag
alu_cout
alu_sout[31:0]
bitoperator_a[31:0]
bitoperator_amount[4:0]
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
multiplier_allow
multiplier_c[63:0]
multiplier_res[63:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr1[3:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
90 ns
100 ns
110 ns
120 ns


Time
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
multiplier_allow
multiplier_c[63:0]
multiplier_res[63:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr1[3:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
registery_datain[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
10 ns
20 ns
30 ns


Time
bitoperator_cout
bitoperator_o[31:0]
bitoperator_op[1:0]
clk
dataregister_currdata[31:0]
dataregister_datain[31:0]
flags_cflag
flags_nflag
flags_vflag
flags_zflag
fsm_aw
fsm_bw
fsm_dw
fsm_fset
fsm_iw
fsm_mw
fsm_op[3:0]
fsm_pw
fsm_rew
fsm_rw
fsm_xw
fsm_yw
instructionregister_currdata[31:0]
memory_addr[8:0]
memory_bytesig[3:0]
memory_dataout[31:0]
multiplier_allow
multiplier_c[63:0]
multiplier_res[63:0]
predicate_result
programcounter_currpc[8:0]
programcounter_newpc[8:0]
registera_currdata[31:0]
registerb_currdata[31:0]
registerfile_rdaddr1[3:0]
registerfile_rdaddr2[3:0]
registerfile_rddata1[31:0]
registerfile_rddata2[31:0]
registerfile_wraddr[3:0]
registerfile_wrdata[31:0]
registerx_currdata[31:0]
registery_currdata[31:0]
registery_datain[31:0]
resultregister_currdata[31:0]
statecounter[3:0]
30 ns
40 ns
50 ns
60 ns



Device Utilization for 7A100TCSG324 is as follows:

```
# Info: ********************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ********************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: --------------------------------------------------------------------
# Info: IOs                               1       210       0.48%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              0       63400     0.00%
# Info: CLB Slices                        0       15850     0.00%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: --------------------------------------------------------------------
# Info: ************************************************************
# Info: Library: work    Cell: processor    View: arc
# Info: ************************************************************
# Info:  Number of ports :                            1
# Info:  Number of nets :                             0
# Info:  Number of instances :                        0
# Info:  Number of references to this view :          0
# Info: Total accumulated area :
# Info:  Number of gates :                            0
# Info:  Number of accumulated instances :            0
# Info: ******************************
```