Lab 4 Report

Software Team

Tong Ray Huang 52347257

David Pham 66811669

March 20, 2019

1 Introduction

The following report explains the functionality of a pipelined RISC-V processor improved upon a single cycle processor. The goal was to implement a set of 57 instructions of the following types:

- U-type
- S-type
- R-type
- B-type
- I-type
- J-type

The processor design also underwent synthesis to determine the critical path length, critical path slack and area and power report of the design.

2 Block Diagram

Refer to Figure 1.

3 Modules

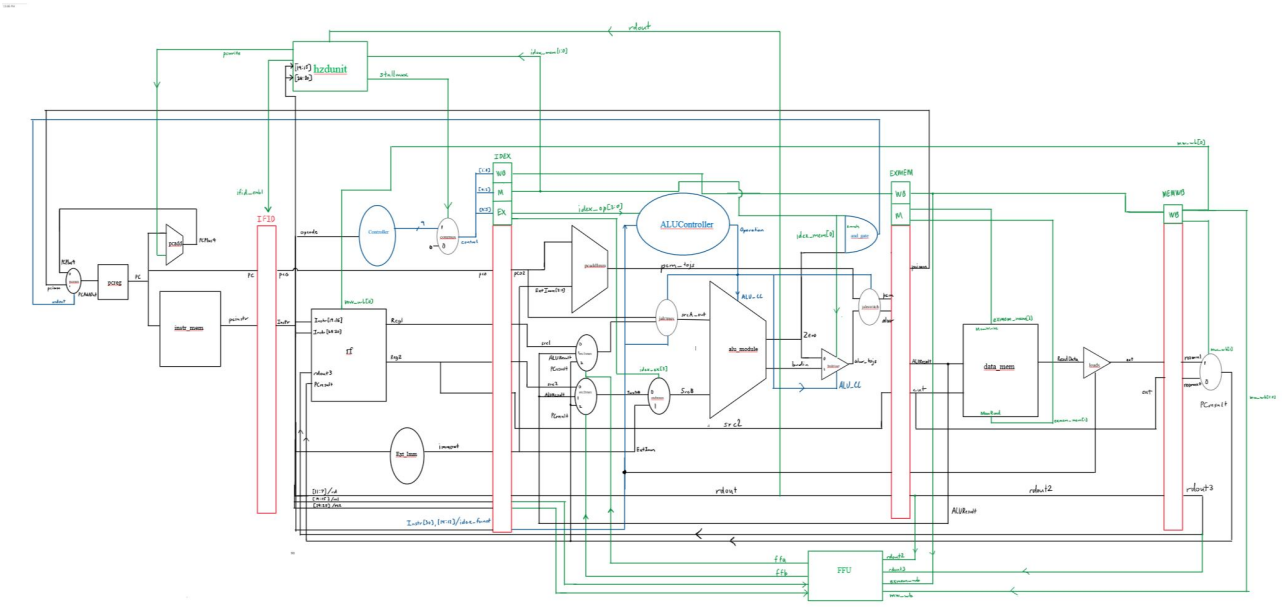
3.1 PC Register

The PC register keeps track of the addresses of instruction memories. It holds a 9 bit address counting value and is connected to instruction memory's address port and several adders to calculate its next value. It takes in a 9 bit value-either it's current address plus 4 or a modified jump address for branches and special instructions-to increment it's counting value every clock cycle (rising edge).

3.2 Instruction Memory

The instruction memory holds 128 32-bit instructions which feed its contents from the address selected by PC register into RegFile, controller, immediate generator, and sign extension modules.

Figure 1: Block Diagram of Pipelined RISC-V Processor



3.3 Controller

The controller manages register and alu modules to handle each instruction. Controller decodes the last 7-bits (OPCode) from every instruction, outputs a special 3-bit operation to ALU controller, and sets the appropriate enables for RegFile (RegWrite), multiplexers (ALUSrc and ResMux), branches (Branch), or data memory (MemRead, MemWrite).

3.4 RegFile

The RegFile houses 32 32-bit registers used for storing/using data from calculated from the instructions. RegFile encodes instruction memory to either poll data out of its two ports (Reg1 and Reg2) or write data into the desired register (RD), so long as it's write is enabled from the controller. Triggers on the falling edge of CLK.

3.5 Immediate Generator

The immediate generator takes in 20,12 or 5 bit immediates from instruction memory extends it a 32-bit value according it's instruction's last 7-bits (OPCode). For I-Type instructions, it feeds out to a multiplexer for ALU operations. For branch instructions, it links to a 9-bit adder to calculate an offset with PC's next value.

3.6 ALU Controller

The ALU controller generates a 4-bit operation code for alu operations using from 3 bits from controller (ALUOp) and 4 bits (Funct3 and 5th bit of Funct7) from instruction memory. It specifies codes for R-type, I-type, Branch, J-type, and U-type instructions, as well as enables to sign extension for load and S-type instructions.

3.7 ALU

The Arithmetic Logic Unit performs operations on data from it's two inputs (Reg1 and SrcB) based on the 4-bit operation code from ALUController. It calculates a 32-bit result used to write back to RegFile for I-type and R-type instructions or find an address in data memory. For branch operations, ALU matches conditions with a Zero flag to enable PCmux for a modified PC value. If the condition is met (always true for J-type instructions), it sets a Zero flag condition to confirm a branch (at the AND gate). ALU is also used to calculate modified offsets.

3.8 Sign Extension

The sign extension modules help truncate or extend for Load/Store halfword or byte operations. It needs a 3-bit code from ALU controller and instruction memory to decide it's output.

3.9 Data Memory

The Data memory holds 512x32 bits of memory used to store/load results for extended ALU operation. It finds a 32-bit address from the alu (ALUResult) and either writes or reads from that address based on enables (MemWrite, MemRead) from controller. Triggers on the falling edge of CLK.

3.10 Adder

Takes in two same width values and outputs the sum. Modified to accommodate 9-bits PC value and 32-bit offset.

3.11 Mux2

Acts as a two to one multiplexer. Selects an output based on an enable.

3.12 Mux3

Acts as a three to one multiplexer. Provide the inputs into the ALU module and are enabled by the forwarding unit.

3.13 Branch Driver

Acts as a two to one selector specific to determining writeback data. The module chooses between the ALU data output and Zero flag.

3.14 Hazard Unit

The hazard unit detects for conflicts in the access of memory at different stages of the pipeline. To fix data dependencies, the hazard unit stalls the pipelie by flushing the registers with zeroes for two clock cycles. As a result, the instruction is still in its decode stage while a NOP occupies the execution stage.

3.15 Forwarding Unit

The forwarding unit detects data dependencies of registers in the memory or writeback stages. For the situation where a register is decoded but passes through the memory or writeback stages, the forwarding unit drives the register to the three to one mux in the execution cycle instead of waiting until all stages of the processor are finished.

3.16 JALR Mux

Specific to the JALR instruction. The PC goes into the ALU module instead of a register.

3.17 JALR Switch

The output of the ALU module is the input of the PC flip flop to execute the jump and link. PC + 4 istored into ALU Result which is written in the destination register indicated by the instruction.

3.18 IFID Register

The pipeline register between the fetch stage and decode stage. The next instruction indicated by the program counter is pushed into the register as long as the hazard unit does not detect a data dependency and stalls are not needed.

3.19 IDEX Register

The pipeline register between the decode and execution stages. This register is the first to be flushed with zeroes when the hazard unit detects the need for a stall.

3.20 EXMEM Register

The pipeline register between execution and memory stages. Stores the PC + Immediate value, Register 2, destination register, pipelined operation code and ALU result.

3.21 MEMWB Register

Pipelined register between execution and writeback stages. Stores the output of data memory, destination register and source B from the register file.

4 Instructions

4.1 R-type

Reads two registers from RegFile into ALU and performs an operation based on a ALUOpcode from ALU controller. Output feeds to a multiplexer split between data memory output and ALU output and a sign extension module before going back to a result register in RegFile. Controller enables RegWrite.

4.2 I-type

Reads one register from RegFile and an extended immediate from immediate generator fed into SrcBMux into ALU and performs an operation based on a ALUOpcode from ALU controller. Output feeds to a multiplexer split between data memory output and ALU output and a sign extension module before going back to a result register in RegFile. Controller enables RegWrite and ALUSrc.

4.3 Load

Reads one register from RegFile and an extended immediate from immediate generator fed into SrcBMux into ALU. A data memory address is formed by adding the two together. Data memory read contents to a multiplexer split between data memory output and ALU output and a sign extension module before going back to a result register in RegFile. Controller enables RegWrite, MemRead, MemtoReg, and ALUSrc.

4.4 Store

Forms a data memory address by adding contents from one selected register (Reg1) in RegFile and an extended immediate from immediate generator fed into ALU. Data memory writes content from a sign extension linked to another register in RegFile (Reg2) for store half-word and byte operations. Controller enables MemWrite and ALUSrc.

4.5 Branch

A 32-bit shifted immediate is generated from 12-bits of instruction memory before linking data to the PC adder and ALU. Data from two registers from RegFile is compared with a condition in ALU. The ALU checks for the correct branch condition obtained from ALU controller and sets the Zero flag if the condition is met. PCmux is enabled through an AND gate to select the modified offset value from PC adder into PC. Controller enables branch and ALUSrc.

4.6 JAL

Stores the result from adding 20-bit immediate and PC's current value into a destination register. A 32-bit shifted immediate is generated from 20-bits of instruction memory before linking data to the PC adder and RegFile. Uses the same procedure for branch except the Zero flag condition is always set true. Controller enables ALUSrc, Branch, and RegWrite.

4.7 JALR

Uses a smaller 12-bit immediate which feeds adder result from PCmux (the next instruction) into another multiplexer connected to RegFile's write data. The offset address is calculated through another ALU module, which takes in an immediate value and PC's current value. Controller enables ALUSrc and RegWrite (Branch is not set to store the next instruction, PC+4, to RegFile instead of the offset address, which is fed to PC through the ALU).

4.8 U-type

Default instruction is LUI, which adds data from one register in RegFile (Reg1) and a shifted upper 20-bit immediate to a destination register back in RegFile. If the control Branch is set,

then an AUIPC instruction is called. The upper immediate is added to the current PC value via PC adder, which itself is added to PC after being selected from PCmux and goes through a sign extension module and a multiplexer split between resmux (the result from ALU or datamemory) before writing its value in RegFile. Controller enables RegWrite and ALUsrc.

5 Synthesis

Critical path length	Critical path slack
0.72	0.76

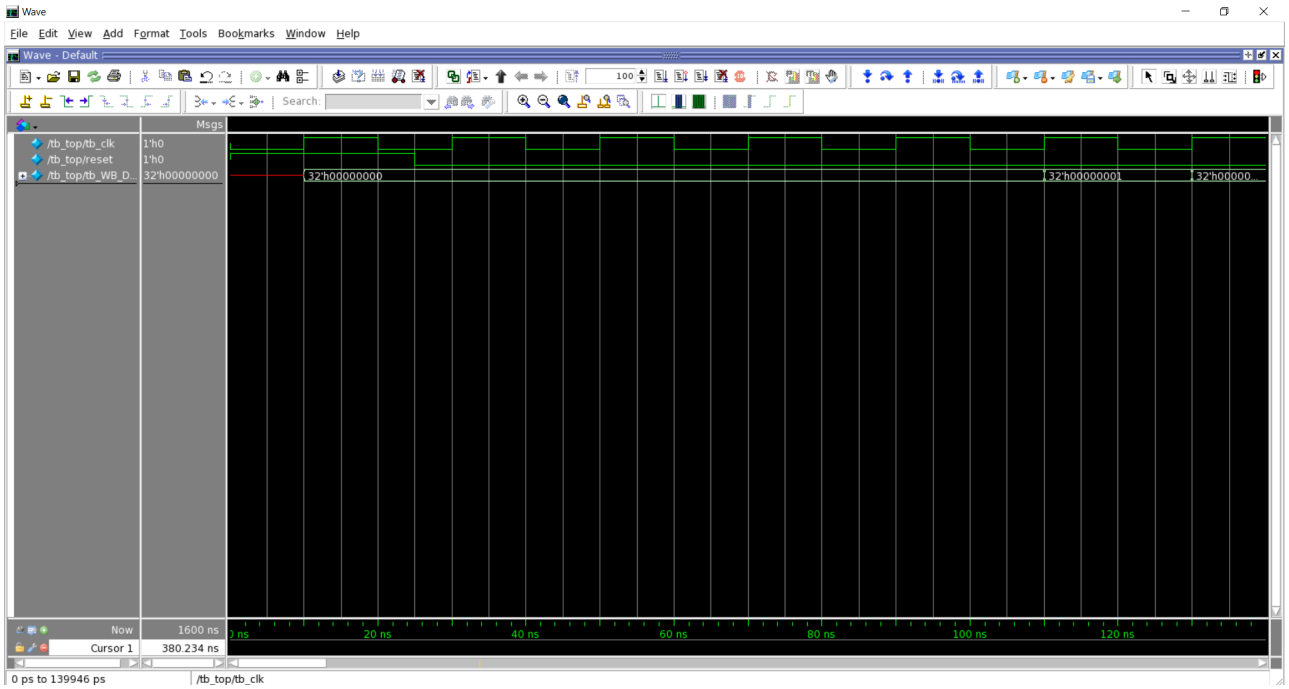
The clock was changed from 4 to 1.5 making the pipelined processor faster by 0.63 of the original run time. The slack is no longer negative.

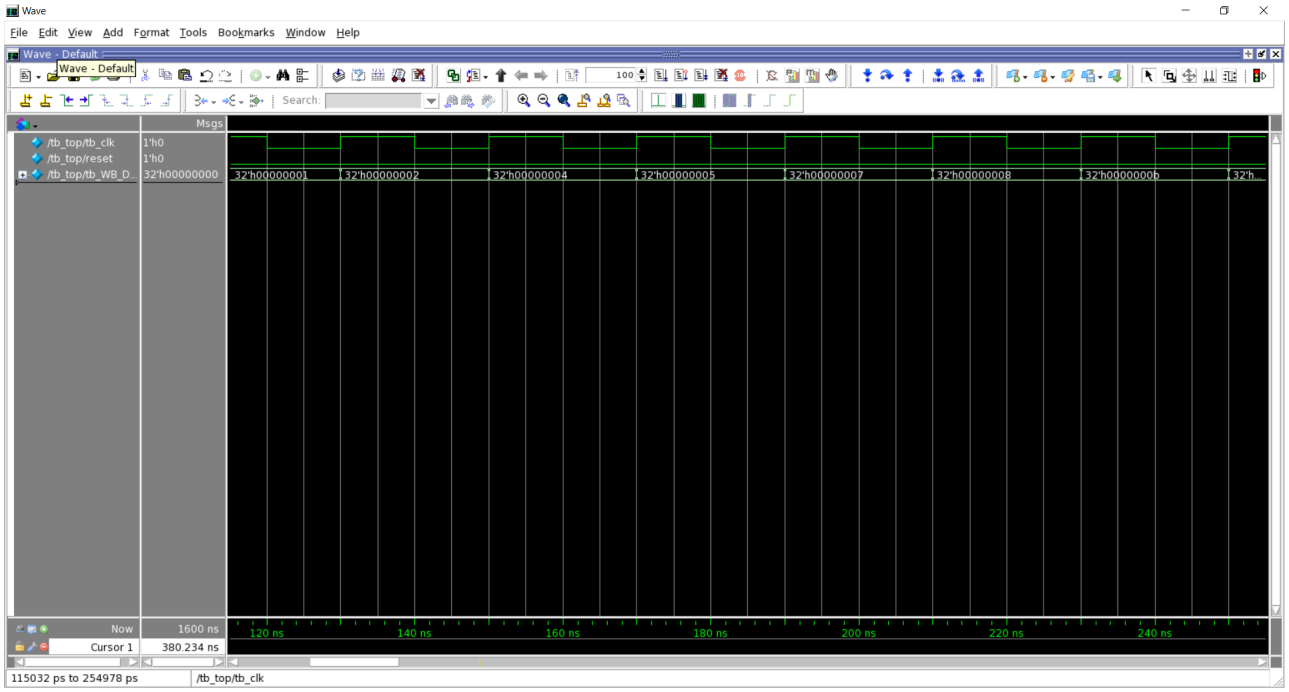
6 Waveforms

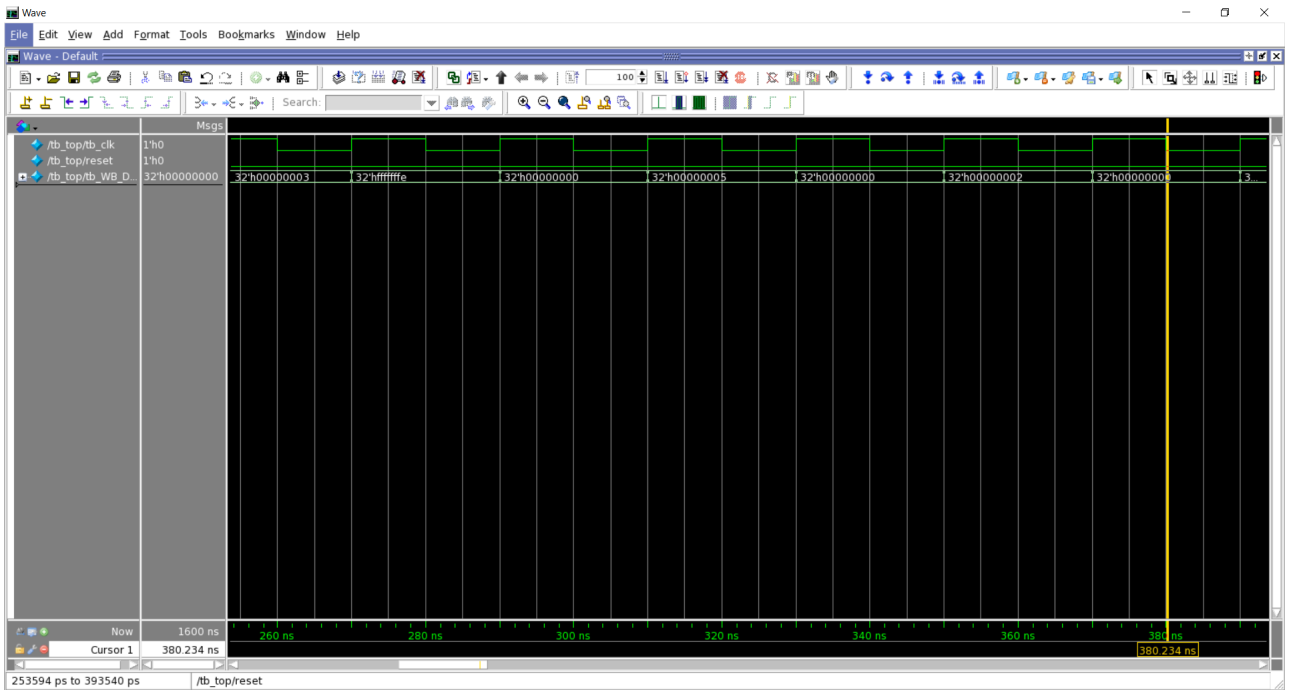
```
Area
--------------------------------
Combinational Area:     9493.294980
Noncombinational Area:  6856.297055
Buf/Inv Area:            348.177278
Total Buffer Area:            12.71
Total Inverter Area:         335.47
Macro/Black Box Area:  50667.683594
Net Area:               7126.455533
--------------------------------
Cell Area:             67017.275629
Design Area:           74143.731162


Design Rules
--------------------------------
Total Number of Nets:          5534
Nets With Violations:             1
Max Trans Violations:             0
Max Cap Violations:               1
--------------------------------


Hostname: zuma.eecs.uci.edu

Compile CPU Statistics
---------------------------------------
Resource Sharing:                   3.47
Logic Optimization:                 2.66
Mapping Optimization:              10.45
---------------------------------------
Overall Compile Time:              47.01
Overall Compile Wall Clock Time:  155.69

--------------------------------------------------------------------

Design  WNS: 0.00  TNS: 0.00  Number of Violating Paths: 0


Design (Hold)  WNS: 0.00  TNS: 0.00  Number of Violating Paths: 0

--------------------------------------------------------------------
```

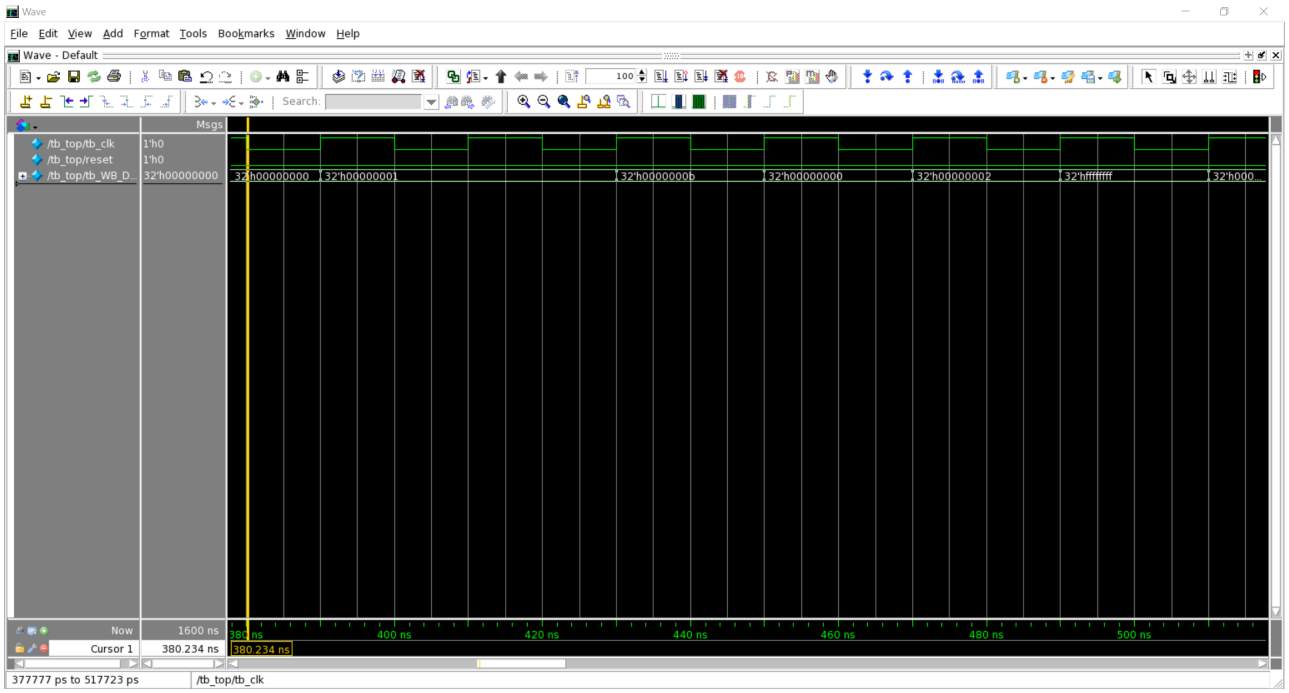
```
****************************************
Report : qor
Design : riscv
Version: J-2014.09-SP4
Date   : Wed Mar 20 20:58:03 2019
****************************************


  Timing Path Group 'clk'
  -----------------------------------
  Levels of Logic:              11.00
  Critical Path Length:          0.72
  Critical Path Slack:           0.76
  Critical Path Clk Period:      1.50
  Total Negative Slack:          0.00
  No. of Violating Paths:        0.00
  Worst Hold Violation:          0.00
  Total Hold Violation:          0.00
  No. of Hold Violations:        0.00
  -----------------------------------


  Cell Count
  -----------------------------------
  Hierarchical Cell Count:         13
  Hierarchical Port Count:        782
  Leaf Cell Count:               4889
  Buf/Inv Cell Count:             226
  Buf Cell Count:                  14
  Inv Cell Count:                 212
  CT Buf/Inv Cell Count:            0
  Combinational Cell Count:      3850
  Sequential Cell Count:         1039
  Macro Count:                      0
  -----------------------------------
```

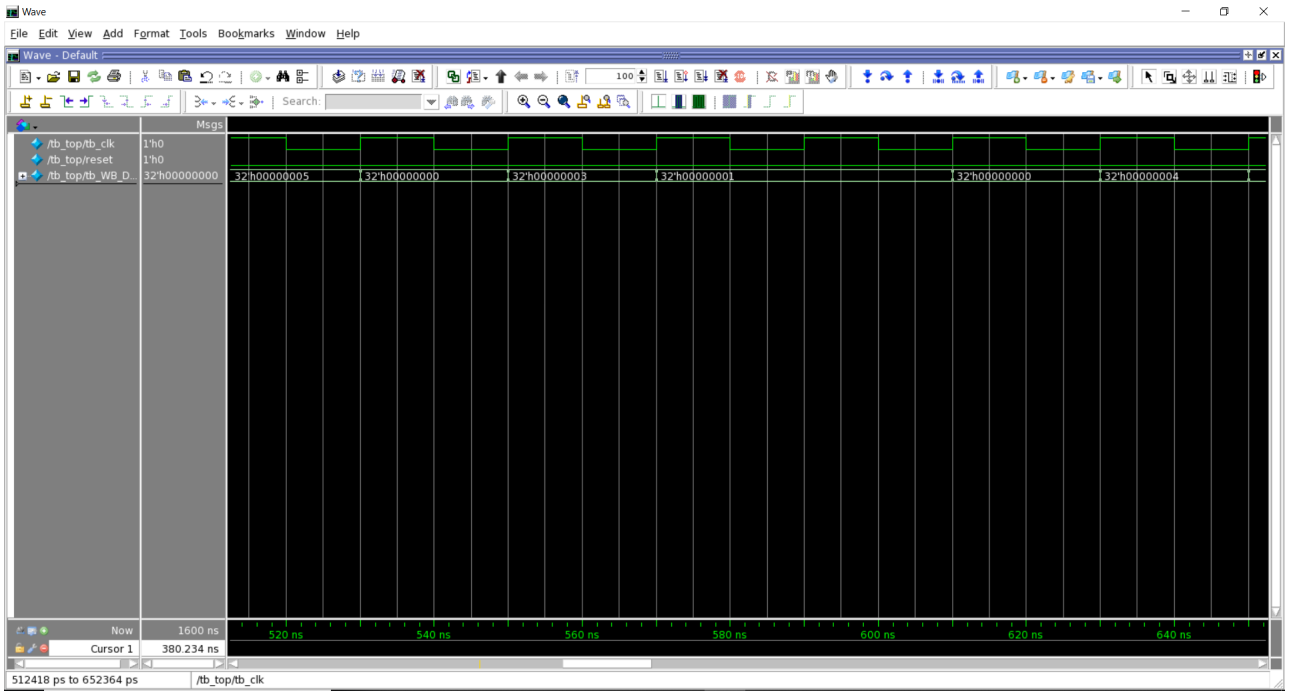
```
Global Operating Voltage = 1.05
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1pW


--------------------------------------------------------------------------------
                                    Switch   Int      Leak     Total
Hierarchy                           Power    Power    Power    Power    %
--------------------------------------------------------------------------------
riscv                              1.26e+03 5.25e+03 1.23e+10 1.88e+04 100.0
  dp (Datapath)                    2.26e+03 5.24e+03 1.22e+10 1.97e+04 104.8
    data_mem (datamemory)             1.057  628.235 1.48e+04  629.306   3.3
    alu_module (alu)                 76.007  141.360 1.20e+09 1.41e+03   7.5
    srcbmux (mux2_WIDTH32_1)         13.077    5.521 5.65e+07   75.091   0.4
    Ext_Imm (imm_Gen)                 6.463    5.357 3.55e+07   47.319   0.3
    resmux (mux2_WIDTH32_0)           7.527    4.061 5.32e+07   64.812   0.3
    conmux (mux2_WIDTH9_1)            0.387    0.873 4.22e+06    5.478   0.0
    rf (RegFile)                        N/A 4.42e+03 1.07e+10 1.49e+04  79.1
    instr_mem (instructionmemory)       N/A    0.000    0.000      N/A   N/A
      C8 (*MUX_OP_128_7_30)           0.132    0.000    0.000    0.132   0.0
    pcmux (mux2_WIDTH9_0)             1.597    2.466 1.49e+07   18.964   0.1
    pcaddImm (adder_WIDTH9_1)         2.760    4.089 3.50e+07   41.878   0.2
    pcreg (flopr_WIDTH9)                N/A   32.429 7.52e+07      N/A   N/A
    pcadd (adder_WIDTH9_0)            1.585    4.062 3.50e+07   40.682   0.2
1
```

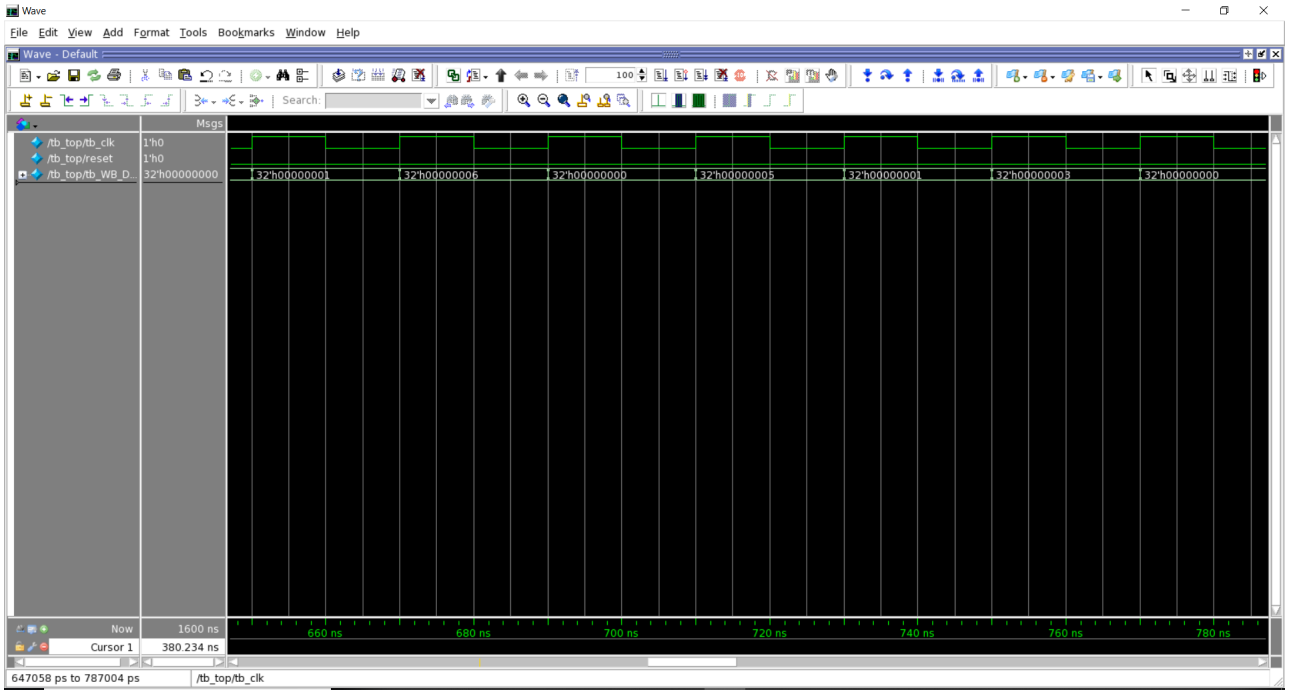

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h00000000
32'h00000001
Now
1600 ns
Cursor 1
380.234 ns
20 ns
40 ns
60 ns
80 ns
100 ns
120 ns
0 ps to 139946 ps
/tb_top/tb_clk

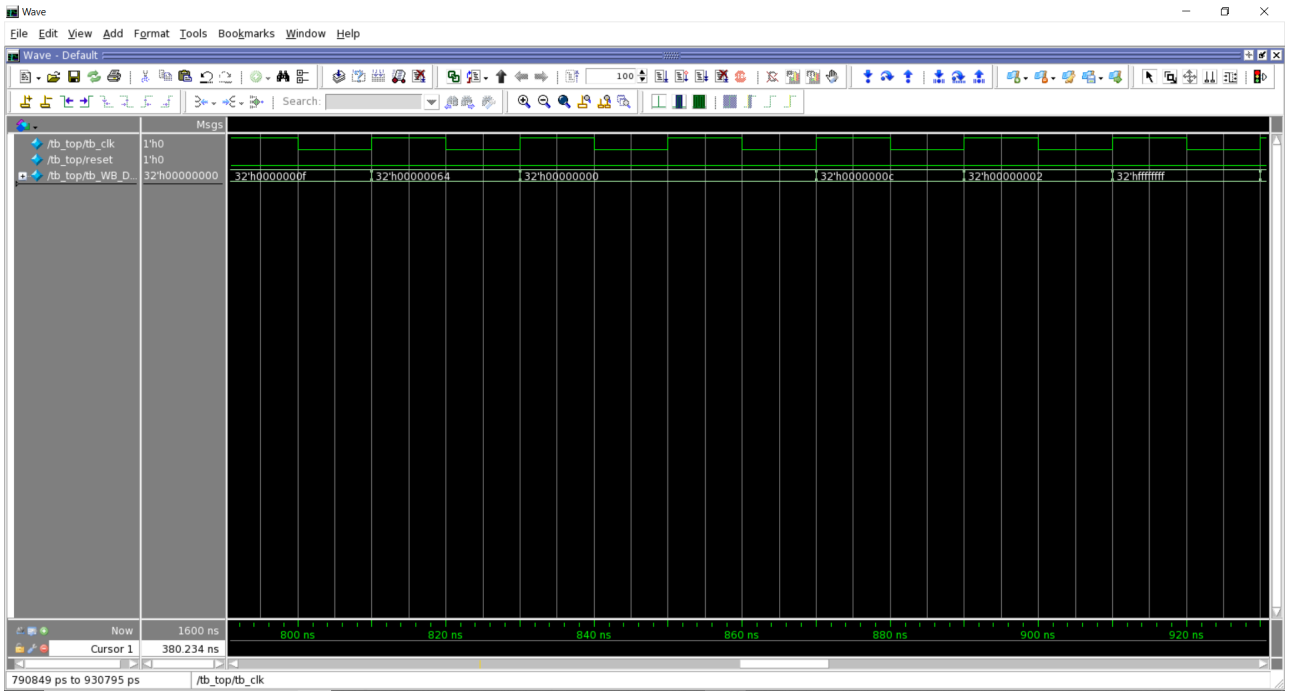

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h00000001
32'h00000002
32'h00000004
32'h00000005
32'h00000007
32'h00000008
32'h0000000b
Now
1600 ns
Cursor 1
380.234 ns
120 ns
140 ns
160 ns
180 ns
200 ns
220 ns
240 ns
115032 ps to 254978 ps
/tb_top/tb_clk

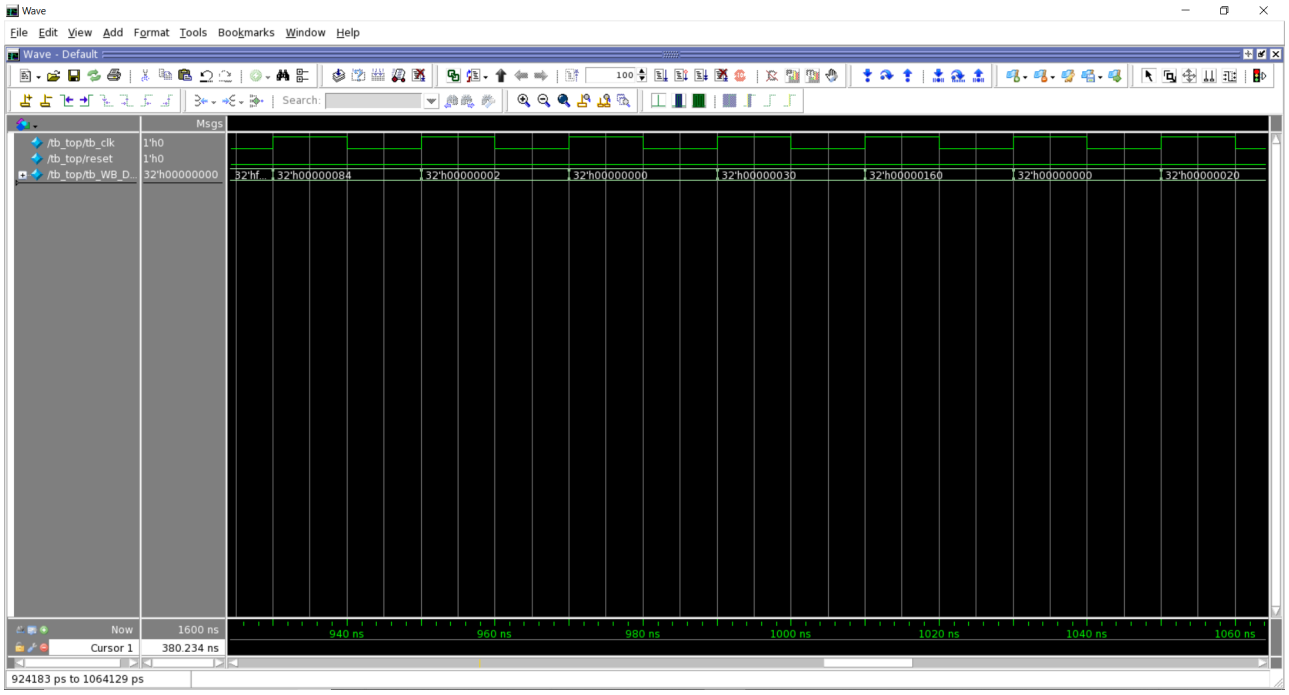

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h00000003
32'hfffffffe
32'h00000000
32'h00000005
32'h00000000
32'h00000002
32'h00000000
Now
1600 ns
Cursor 1
380.234 ns
260 ns
280 ns
300 ns
320 ns
340 ns
360 ns
380 ns
380.234 ns
253594 ps to 393540 ps
/tb_top/reset

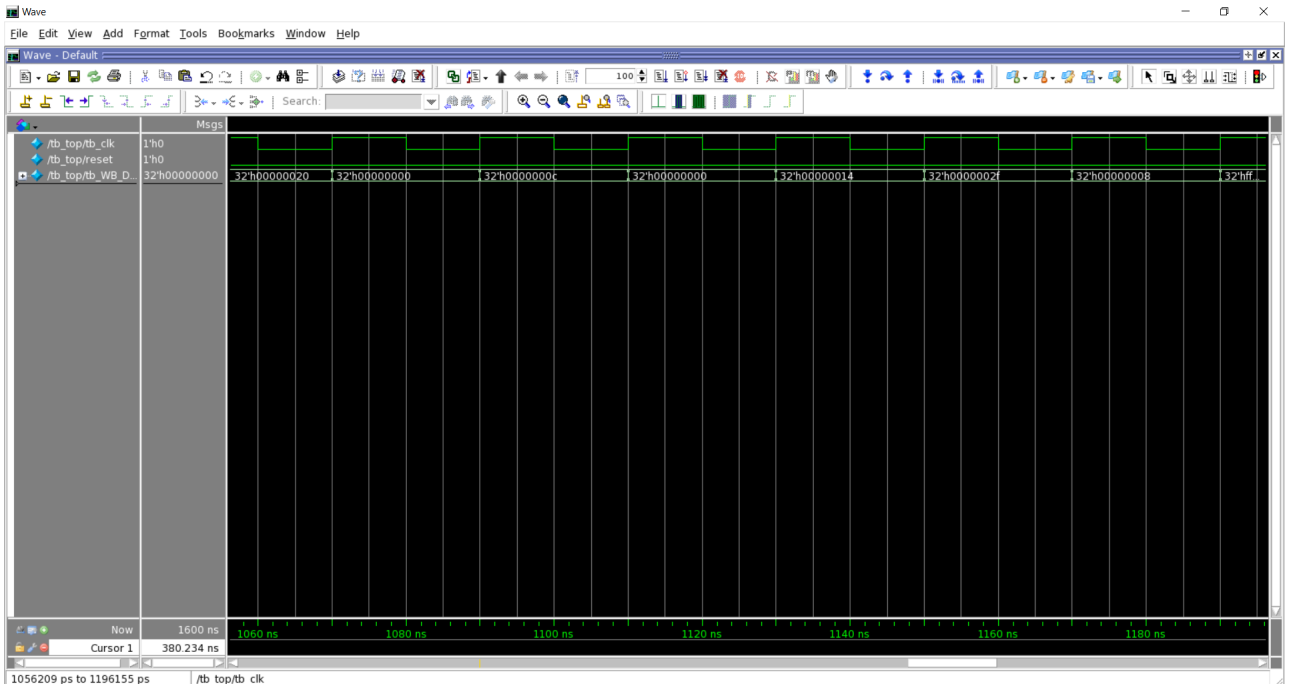

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h00000000
32'h00000001
32'h0000000b
32'h00000000
32'h00000002
32'hffffffff
Now
1600 ns
Cursor 1
380.234 ns
380 ns
400 ns
420 ns
440 ns
460 ns
480 ns
500 ns
380.234 ns
377777 ps to 517723 ps
/tb_top/tb_clk

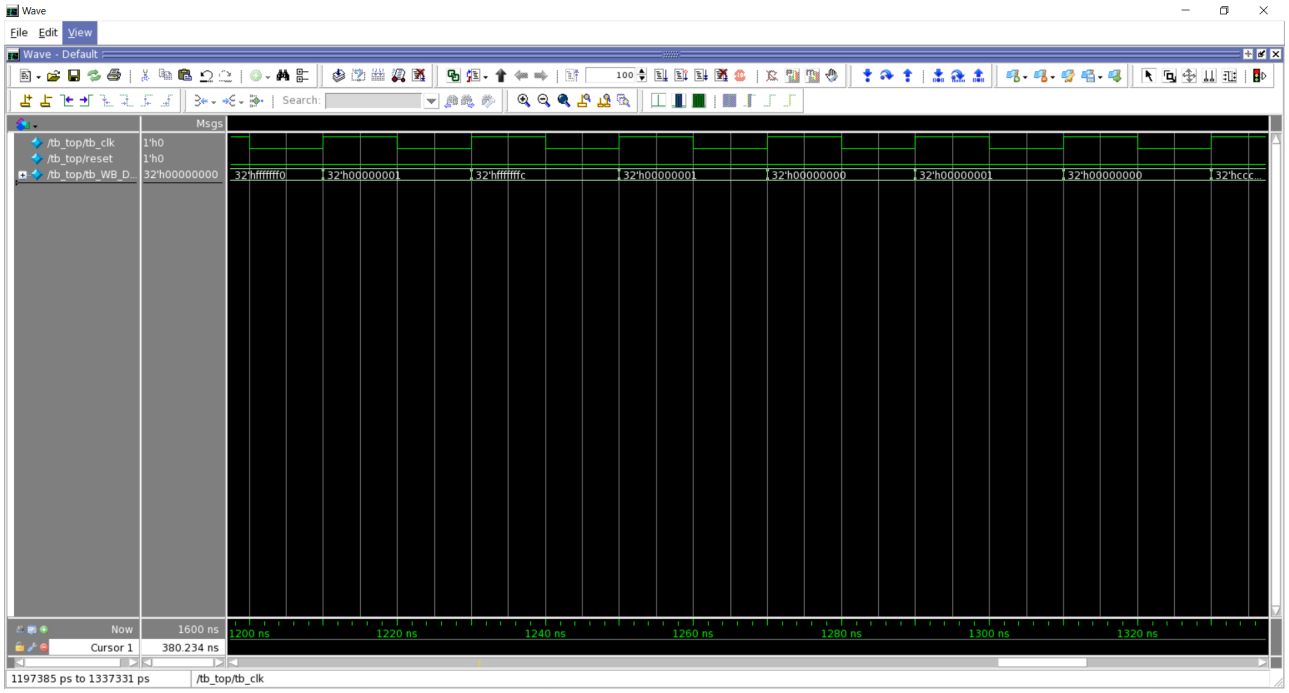

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h00000005
32'h00000000
32'h00000003
32'h00000001
32'h00000000
32'h00000004
Now
1600 ns
Cursor 1
380.234 ns
520 ns
540 ns
560 ns
580 ns
600 ns
620 ns
640 ns
512418 ps to 652364 ps
/tb_top/tb_clk

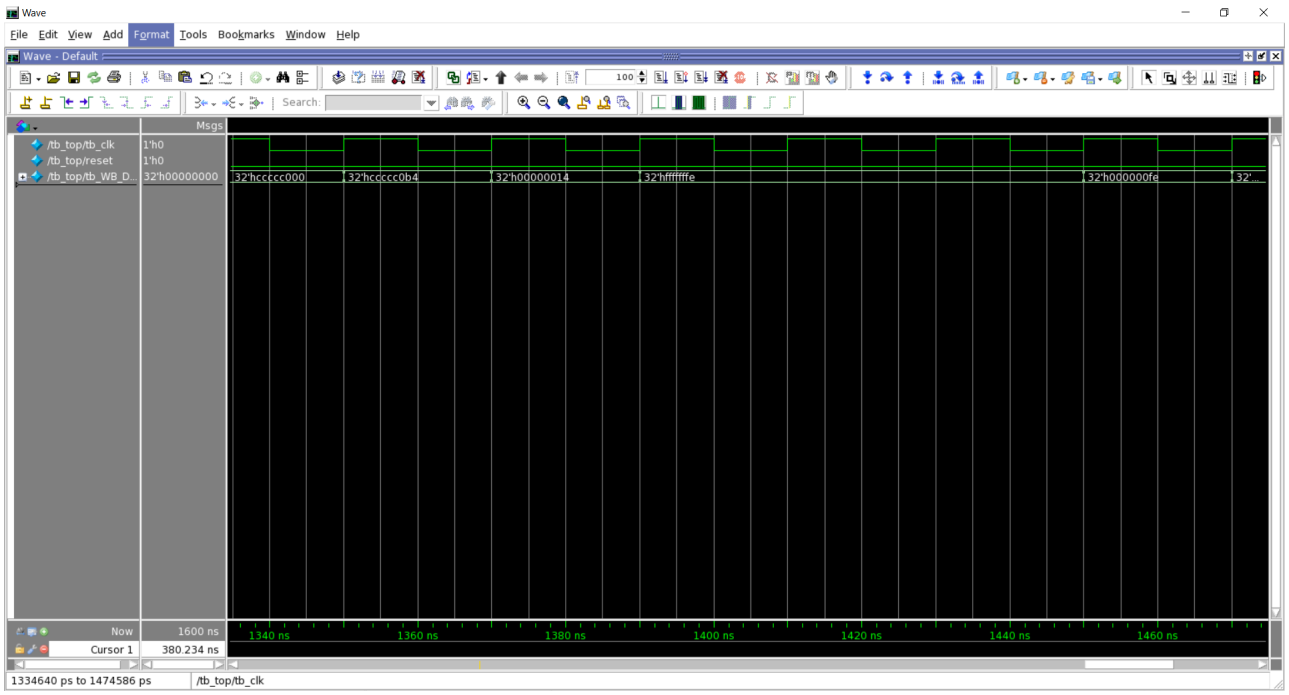

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h00000001
32'h00000006
32'h00000000
32'h00000005
32'h00000001
32'h00000003
32'h00000000
Now
1600 ns
Cursor 1
380.234 ns
660 ns
680 ns
700 ns
720 ns
740 ns
760 ns
780 ns
647058 ps to 787004 ps
/tb_top/tb_clk

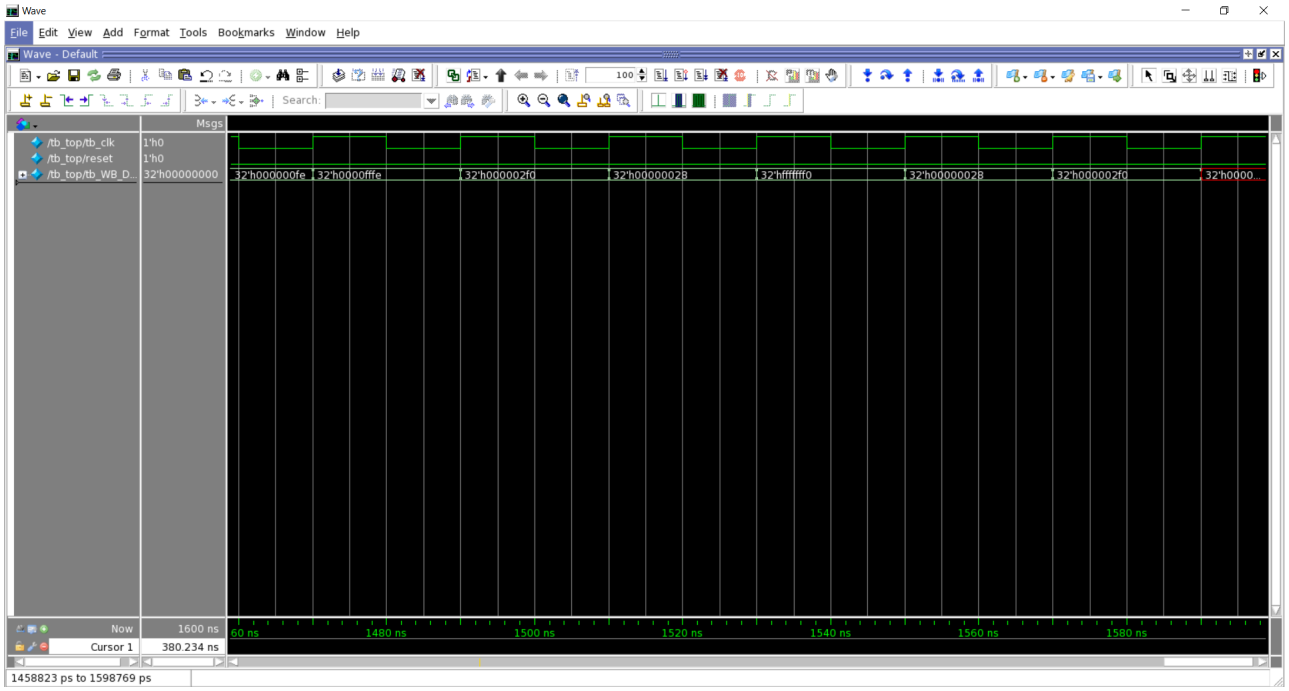

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h0000000f
32'h00000064
32'h00000000
32'h0000000c
32'h00000002
32'hffffffff
Now
1600 ns
Cursor 1
380.234 ns
800 ns
820 ns
840 ns
860 ns
880 ns
900 ns
920 ns
790849 ps to 930795 ps
/tb_top/tb_clk


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'hf...
32'h00000084
32'h00000002
32'h00000000
32'h00000030
32'h00000160
32'h00000000
32'h00000020
Now
1600 ns
Cursor 1
380.234 ns
940 ns
960 ns
980 ns
1000 ns
1020 ns
1040 ns
1060 ns
924183 ps to 1064129 ps


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h00000020
32'h00000000
32'h0000000c
32'h00000000
32'h00000014
32'h0000002f
32'h00000008
32'hff...
Now
1600 ns
Cursor 1
380.234 ns
1060 ns
1080 ns
1100 ns
1120 ns
1140 ns
1160 ns
1180 ns
1056209 ps to 1196155 ps
/tb top/tb clk


Wave
File Edit View
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'hfffffff0
32'h00000001
32'hfffffffc
32'h00000001
32'h00000000
32'h00000001
32'h00000000
32'hccc...
Now
1600 ns
Cursor 1
380.234 ns
1200 ns
1220 ns
1240 ns
1260 ns
1280 ns
1300 ns
1320 ns
1197385 ps to 1337331 ps
/tb_top/tb_clk


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'hcccccc000
32'hcccccc0b4
32'h00000014
32'hffffffe
32'h000000fe
Now
1600 ns
Cursor 1
380.234 ns
1340 ns
1360 ns
1380 ns
1400 ns
1420 ns
1440 ns
1460 ns
1334640 ps to 1474586 ps
/tb_top/tb_clk


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_top/tb_clk
/tb_top/reset
/tb_top/tb_WB_D...
1'h0
1'h0
32'h00000000
32'h000000fe
32'h0000fffe
32'h000002f0
32'h0000002B
32'hffffffff0
32'h0000002B
32'h000002f0
32'h0000...
Now
1600 ns
Cursor 1
380.234 ns
60 ns
1480 ns
1500 ns
1520 ns
1540 ns
1560 ns
1580 ns
1458823 ps to 1598769 ps