Ain Shams University

Faculty of Engineering

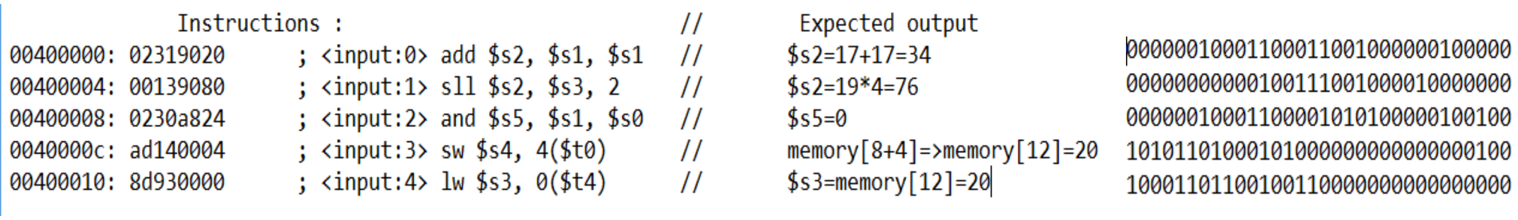
Computer and Systems Engineering Department

Pipelined-Mips Project

- The project is about implementing a pipelined MIPS processor using Verilog code
- Here are 2 programs we used to test our processor:
- Where the register files are loaded with values according to the following equation memory[i]=I .

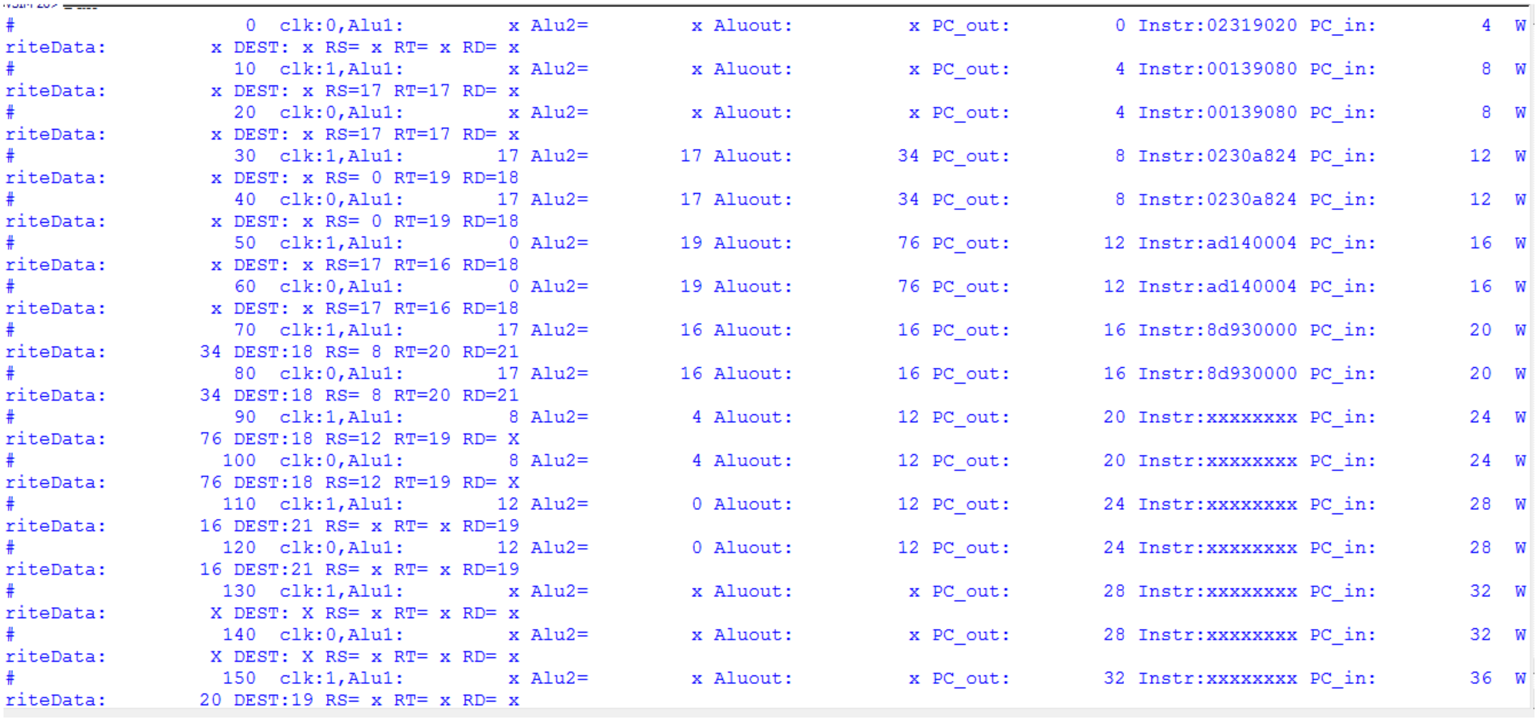
1)

For the first test program, we used instructions that don't involve forwarding or hazard cases:



```
              Instructions :                          //       Expected output
00400000: 02319020      ; <input:0> add $s2, $s1, $s1   //      $s2=17+17=34                   00000010001100011001000000100000
00400004: 00139080      ; <input:1> sll $s2, $s3, 2     //      $s2=19*4=76                    00000000000100111001000010000000
00400008: 0230a824      ; <input:2> and $s5, $s1, $s0   //      $s5=0                          00000010001100001010100000100100
0040000c: ad140004      ; <input:3> sw $s4, 4($t0)      //      memory[8+4]=>memory[12]=20     10101101000101000000000000000100
00400010: 8d930000      ; <input:4> lw $s3, 0($t4)      //      $s3=memory[12]=20              10001101100100110000000000000000
```

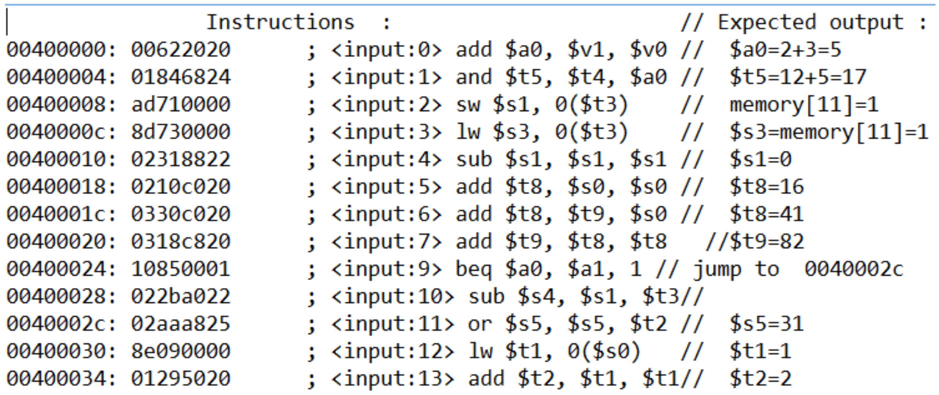
Output:



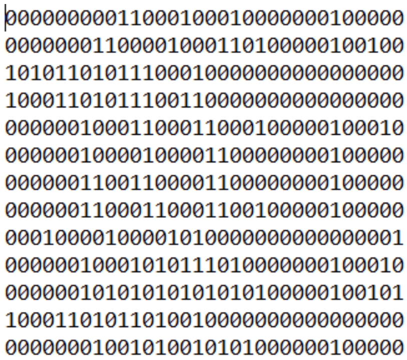
```
#             0  clk:0,Alu1:         x Alu2=         x Aluout:         x PC_out:         0 Instr:02319020 PC_in:         4  W
riteData:         x DEST: x RS= x RT= x RD= x
#            10  clk:1,Alu1:         x Alu2=         x Aluout:         x PC_out:         4 Instr:00139080 PC_in:         8  W
riteData:         x DEST: x RS=17 RT=17 RD= x
#            20  clk:0,Alu1:         x Alu2=         x Aluout:         x PC_out:         4 Instr:00139080 PC_in:         8  W
riteData:         x DEST: x RS=17 RT=17 RD= x
#            30  clk:1,Alu1:        17 Alu2=        17 Aluout:        34 PC_out:         8 Instr:0230a824 PC_in:        12  W
riteData:         x DEST: x RS= 0 RT=19 RD=18
#            40  clk:0,Alu1:        17 Alu2=        17 Aluout:        34 PC_out:         8 Instr:0230a824 PC_in:        12  W
riteData:         x DEST: x RS= 0 RT=19 RD=18
#            50  clk:1,Alu1:         0 Alu2=        19 Aluout:        76 PC_out:        12 Instr:ad140004 PC_in:        16  W
riteData:         x DEST: x RS=17 RT=16 RD=18
#            60  clk:0,Alu1:         0 Alu2=        19 Aluout:        76 PC_out:        12 Instr:ad140004 PC_in:        16  W
riteData:         x DEST: x RS=17 RT=16 RD=18
#            70  clk:1,Alu1:        17 Alu2=        16 Aluout:        16 PC_out:        16 Instr:8d930000 PC_in:        20  W
riteData:        34 DEST:18 RS= 8 RT=20 RD=21
#            80  clk:0,Alu1:        17 Alu2=        16 Aluout:        16 PC_out:        16 Instr:8d930000 PC_in:        20  W
riteData:        34 DEST:18 RS= 8 RT=20 RD=21
#            90  clk:1,Alu1:         8 Alu2=         4 Aluout:        12 PC_out:        20 Instr:xxxxxxxx PC_in:        24  W
riteData:        76 DEST:18 RS=12 RT=19 RD= X
#           100  clk:0,Alu1:         8 Alu2=         4 Aluout:        12 PC_out:        20 Instr:xxxxxxxx PC_in:        24  W
riteData:        76 DEST:18 RS=12 RT=19 RD= X
#           110  clk:1,Alu1:        12 Alu2=         0 Aluout:        12 PC_out:        24 Instr:xxxxxxxx PC_in:        28  W
riteData:        16 DEST:21 RS= x RT= x RD=19
#           120  clk:0,Alu1:        12 Alu2=         0 Aluout:        12 PC_out:        24 Instr:xxxxxxxx PC_in:        28  W
riteData:        16 DEST:21 RS= x RT= x RD=19
#           130  clk:1,Alu1:         x Alu2=         x Aluout:         x PC_out:        28 Instr:xxxxxxxx PC_in:        32  W
riteData:         X DEST: X RS= x RT= x RD= x
#           140  clk:0,Alu1:         x Alu2=         x Aluout:         x PC_out:        28 Instr:xxxxxxxx PC_in:        32  W
riteData:         X DEST: X RS= x RT= x RD= x
#           150  clk:1,Alu1:         x Alu2=         x Aluout:         x PC_out:        32 Instr:xxxxxxxx PC_in:        36  W
riteData:        20 DEST:19 RS= x RT= x RD= x
```

2)

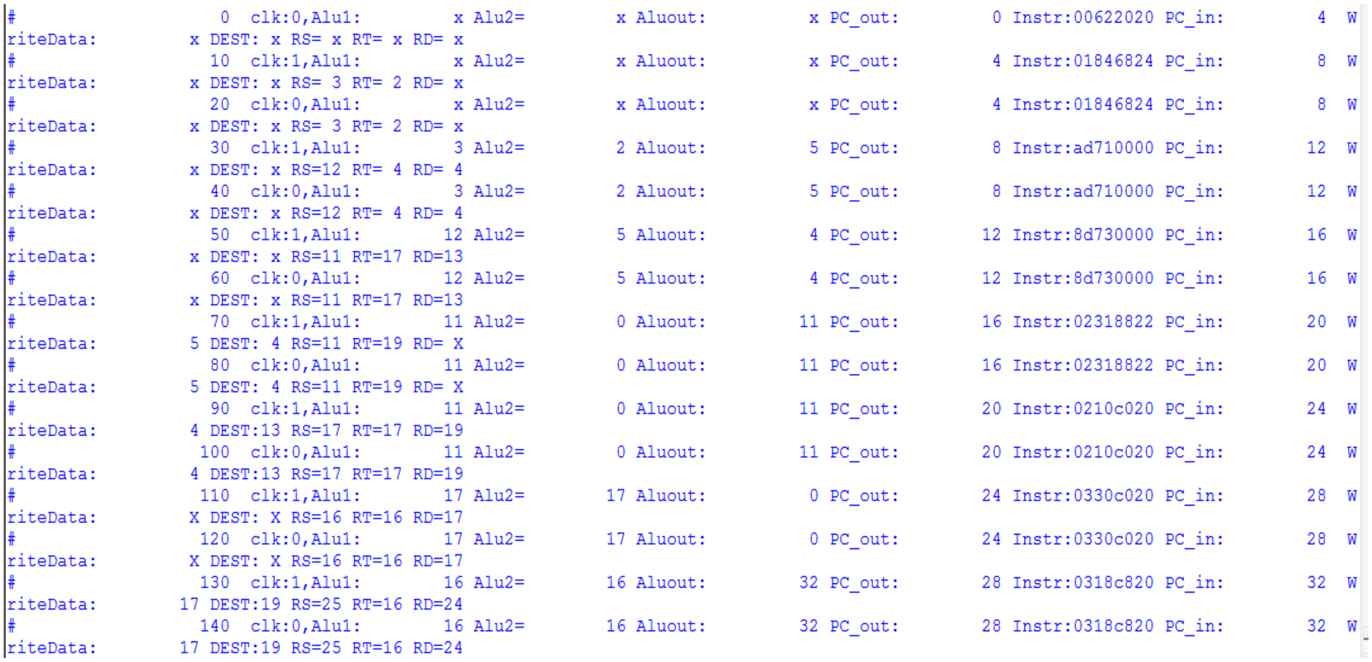
For the second test program, we used instructions that involve forwarding and hazard cases:



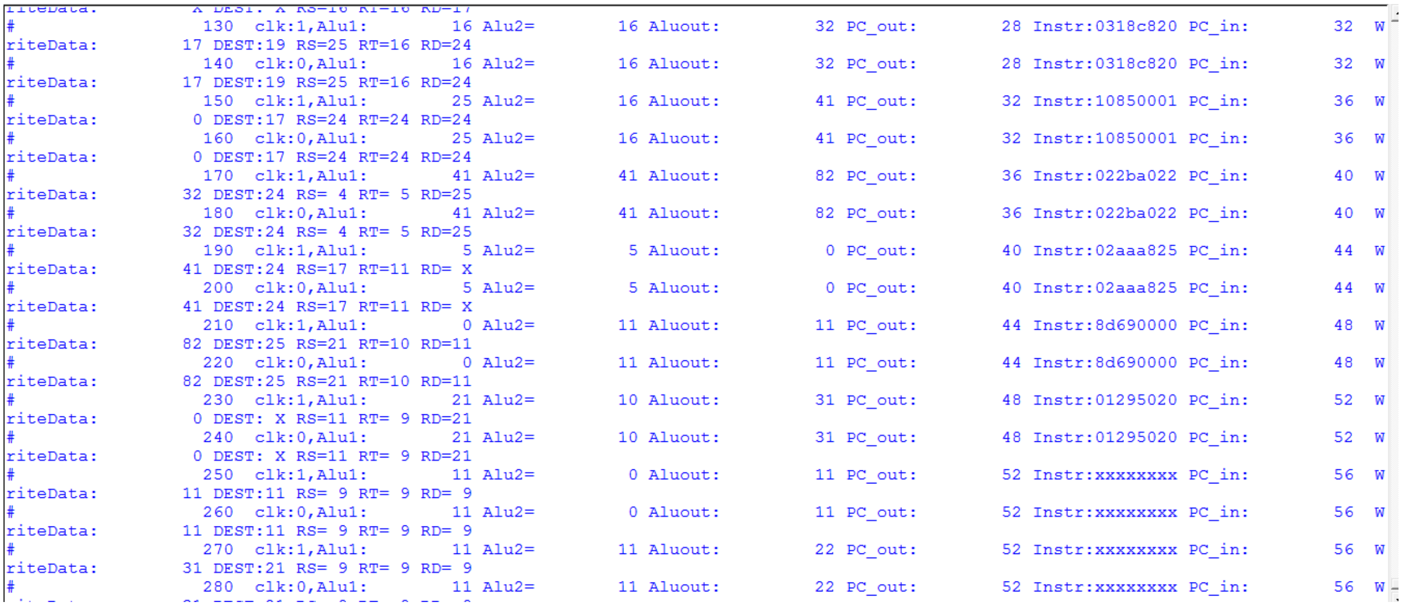
```
                Instructions  :                       // Expected output :
00400000: 00622020      ; <input:0> add $a0, $v1, $v0 //  $a0=2+3=5
00400004: 01846824      ; <input:1> and $t5, $t4, $a0 //  $t5=12+5=17
00400008: ad710000      ; <input:2> sw $s1, 0($t3)    //  memory[11]=1
0040000c: 8d730000      ; <input:3> lw $s3, 0($t3)    //  $s3=memory[11]=1
00400010: 02318822      ; <input:4> sub $s1, $s1, $s1 //  $s1=0
00400018: 0210c020      ; <input:5> add $t8, $s0, $s0 //  $t8=16
0040001c: 0330c020      ; <input:6> add $t8, $t9, $s0 //  $t8=41
00400020: 0318c820      ; <input:7> add $t9, $t8, $t8   //$t9=82
00400024: 10850001      ; <input:9> beq $a0, $a1, 1 // jump to  0040002c
00400028: 022ba022      ; <input:10> sub $s4, $s1, $t3//
0040002c: 02aaa825      ; <input:11> or $s5, $s5, $t2 //  $s5=31
00400030: 8e090000      ; <input:12> lw $t1, 0($s0)   //  $t1=1
00400034: 01295020      ; <input:13> add $t2, $t1, $t1//  $t2=2
```



```
00000000011000100010000000100000
00000001100001000110100000100100
10101101011100010000000000000000
10001101011100110000000000000000
00000010001100011000100000100010
00000010000100001100000000100000
00000011001100001100000000100000
00000011000110001100100000100000
00010000100001010000000000000001
00000010001010111010000000100010
00000010101010101010100000100101
10001101011010010000000000000000
00000001001010010101000000100000
```



```
#                    0  clk:0,Alu1:          x Alu2=          x Aluout:          x PC_out:          0 Instr:00622020 PC_in:          4  W
riteData:          x DEST: x RS= x RT= x RD= x
#                   10  clk:1,Alu1:          x Alu2=          x Aluout:          x PC_out:          4 Instr:01846824 PC_in:          8  W
riteData:          x DEST: x RS= 3 RT= 2 RD= x
#                   20  clk:0,Alu1:          x Alu2=          x Aluout:          x PC_out:          4 Instr:01846824 PC_in:          8  W
riteData:          x DEST: x RS= 3 RT= 2 RD= x
#                   30  clk:1,Alu1:          3 Alu2=          2 Aluout:          5 PC_out:          8 Instr:ad710000 PC_in:         12  W
riteData:          x DEST: x RS=12 RT= 4 RD= 4
#                   40  clk:0,Alu1:          3 Alu2=          2 Aluout:          5 PC_out:          8 Instr:ad710000 PC_in:         12  W
riteData:          x DEST: x RS=12 RT= 4 RD= 4
#                   50  clk:1,Alu1:         12 Alu2=          5 Aluout:          4 PC_out:         12 Instr:8d730000 PC_in:         16  W
riteData:          x DEST: x RS=11 RT=17 RD=13
#                   60  clk:0,Alu1:         12 Alu2=          5 Aluout:          4 PC_out:         12 Instr:8d730000 PC_in:         16  W
riteData:          x DEST: x RS=11 RT=17 RD=13
#                   70  clk:1,Alu1:         11 Alu2=          0 Aluout:         11 PC_out:         16 Instr:02318822 PC_in:         20  W
riteData:          5 DEST: 4 RS=11 RT=19 RD= X
#                   80  clk:0,Alu1:         11 Alu2=          0 Aluout:         11 PC_out:         16 Instr:02318822 PC_in:         20  W
riteData:          5 DEST: 4 RS=11 RT=19 RD= X
#                   90  clk:1,Alu1:         11 Alu2=          0 Aluout:         11 PC_out:         20 Instr:0210c020 PC_in:         24  W
riteData:          4 DEST:13 RS=17 RT=17 RD=19
#                  100  clk:0,Alu1:         11 Alu2=          0 Aluout:         11 PC_out:         20 Instr:0210c020 PC_in:         24  W
riteData:          4 DEST:13 RS=17 RT=17 RD=19
#                  110  clk:1,Alu1:         17 Alu2=         17 Aluout:          0 PC_out:         24 Instr:0330c020 PC_in:         28  W
riteData:          X DEST: X RS=16 RT=16 RD=17
#                  120  clk:0,Alu1:         17 Alu2=         17 Aluout:          0 PC_out:         24 Instr:0330c020 PC_in:         28  W
riteData:          X DEST: X RS=16 RT=16 RD=17
#                  130  clk:1,Alu1:         16 Alu2=         16 Aluout:         32 PC_out:         28 Instr:0318c820 PC_in:         32  W
riteData:         17 DEST:19 RS=25 RT=16 RD=24
#                  140  clk:0,Alu1:         16 Alu2=         16 Aluout:         32 PC_out:         28 Instr:0318c820 PC_in:         32  W
riteData:         17 DEST:19 RS=25 RT=16 RD=24
```



```
riteData:          X DEST: X RS=16 RT=16 RD=17
#                  130  clk:1,Alu1:         16 Alu2=         16 Aluout:         32 PC_out:         28 Instr:0318c820 PC_in:         32  W
riteData:         17 DEST:19 RS=25 RT=16 RD=24
#                  140  clk:0,Alu1:         16 Alu2=         16 Aluout:         32 PC_out:         28 Instr:0318c820 PC_in:         32  W
riteData:         17 DEST:19 RS=25 RT=16 RD=24
#                  150  clk:1,Alu1:         25 Alu2=         16 Aluout:         41 PC_out:         32 Instr:10850001 PC_in:         36  W
riteData:          0 DEST:17 RS=24 RT=24 RD=24
#                  160  clk:0,Alu1:         25 Alu2=         16 Aluout:         41 PC_out:         32 Instr:10850001 PC_in:         36  W
riteData:          0 DEST:17 RS=24 RT=24 RD=24
#                  170  clk:1,Alu1:         41 Alu2=         41 Aluout:         82 PC_out:         36 Instr:022ba022 PC_in:         40  W
riteData:         32 DEST:24 RS= 4 RT= 5 RD=25
#                  180  clk:0,Alu1:         41 Alu2=         41 Aluout:         82 PC_out:         36 Instr:022ba022 PC_in:         40  W
riteData:         32 DEST:24 RS= 4 RT= 5 RD=25
#                  190  clk:1,Alu1:          5 Alu2=          5 Aluout:          0 PC_out:         40 Instr:02aaa825 PC_in:         44  W
riteData:         41 DEST:24 RS=17 RT=11 RD= X
#                  200  clk:0,Alu1:          5 Alu2=          5 Aluout:          0 PC_out:         40 Instr:02aaa825 PC_in:         44  W
riteData:         41 DEST:24 RS=17 RT=11 RD= X
#                  210  clk:1,Alu1:          0 Alu2=         11 Aluout:         11 PC_out:         44 Instr:8d690000 PC_in:         48  W
riteData:         82 DEST:25 RS=21 RT=10 RD=11
#                  220  clk:0,Alu1:          0 Alu2=         11 Aluout:         11 PC_out:         44 Instr:8d690000 PC_in:         48  W
riteData:         82 DEST:25 RS=21 RT=10 RD=11
#                  230  clk:1,Alu1:         21 Alu2=         10 Aluout:         31 PC_out:         48 Instr:01295020 PC_in:         52  W
riteData:          0 DEST: X RS=11 RT= 9 RD=21
#                  240  clk:0,Alu1:         21 Alu2=         10 Aluout:         31 PC_out:         48 Instr:01295020 PC_in:         52  W
riteData:          0 DEST: X RS=11 RT= 9 RD=21
#                  250  clk:1,Alu1:         11 Alu2=          0 Aluout:         11 PC_out:         52 Instr:xxxxxxxx PC_in:         56  W
riteData:         11 DEST:11 RS= 9 RT= 9 RD= 9
#                  260  clk:0,Alu1:         11 Alu2=          0 Aluout:         11 PC_out:         52 Instr:xxxxxxxx PC_in:         56  W
riteData:         11 DEST:11 RS= 9 RT= 9 RD= 9
#                  270  clk:1,Alu1:         11 Alu2=         11 Aluout:         22 PC_out:         52 Instr:xxxxxxxx PC_in:         56  W
riteData:         31 DEST:21 RS= 9 RT= 9 RD= 9
#                  280  clk:0,Alu1:         11 Alu2=         11 Aluout:         22 PC_out:         52 Instr:xxxxxxxx PC_in:         56  W
riteData:         31 DEST:21 RS= 9 RT= 9 RD= 9
```

Note:

To try a new test program of your own choice:

1-Create a text file containing the instructions in binary.

2-Rename the file to “Binary.txt”.

3-Move the file to the folder where your project is saved for example:
C:/Modelsim/examples.