CPU implemented by Verilog

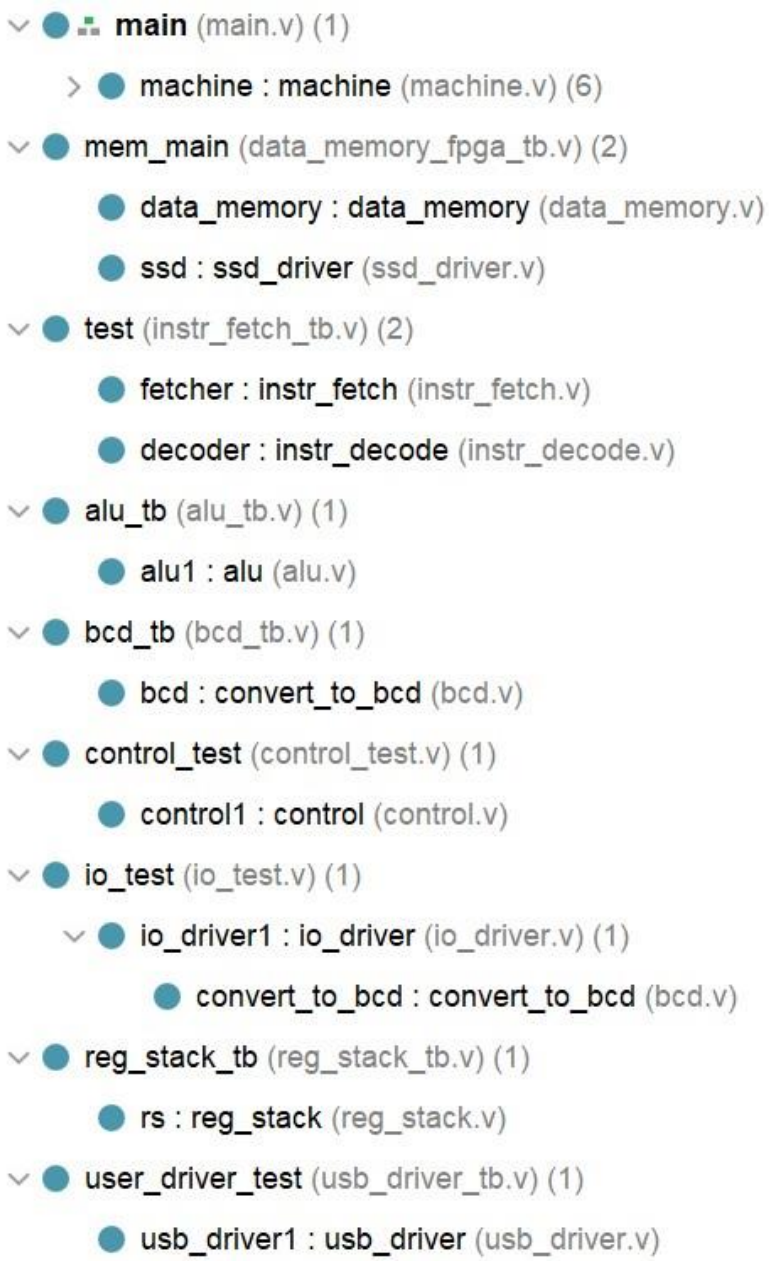
email: LdTenacity666@163.com

In this experiment, I realize a simple cpu as a total machine by Verilog. The code can be organized as the modules shown below: **alu module, control module, io module, memory module, reg_stack module, usb_driver module, bcd module**. In the end, we organize those modules together as **a total machine**.

code: code of all modules in `**code**` directory screen:

screen of all modules in `**screen**` directory the code

is organized as below structure:

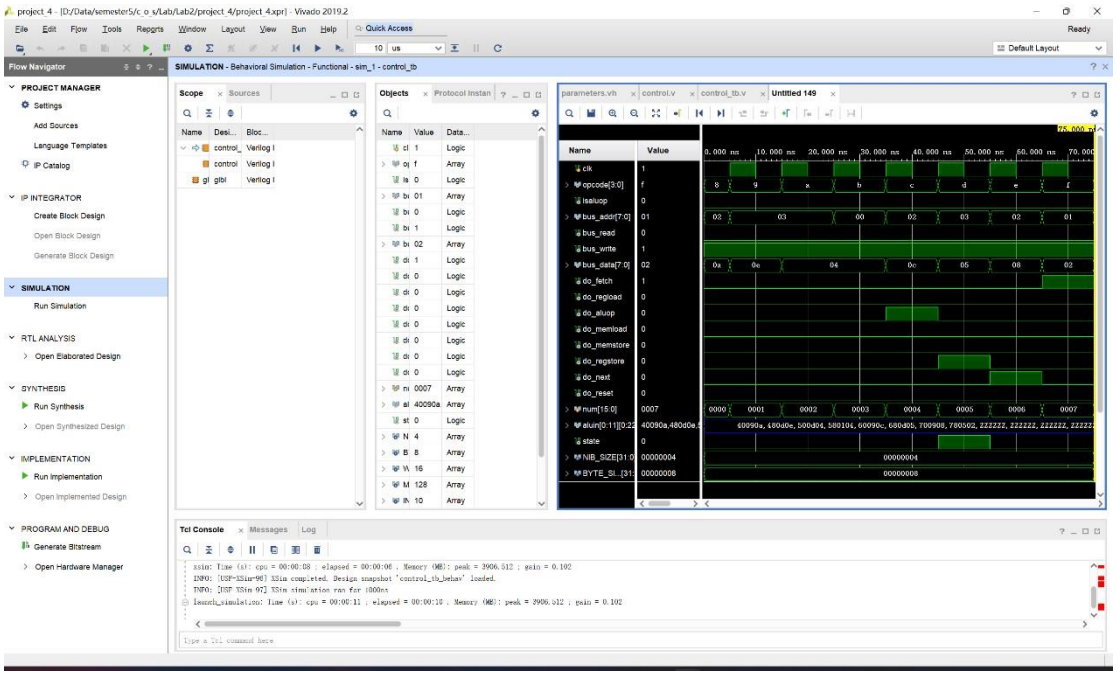


1. **Alu module**

   This module is mentioned in Lab1, and you can check the result snapshot in screen directory and its code in code directory.

2. **Control module** (below shows the test data and the result)

```
assign testdata[0] = {`OP_LOAD, 1'd0, 8'h02, 1'd0, 1'd1, 8'd10};
assign testdata[1] = {`OP_STORE, 1'd0, 8'h03, 1'd0, 1'd1, 8'd14};
assign testdata[2] = {`OP_IN, 1'd0, 8'h03, 1'd0,1'd1,8'd4};
assign testdata[3] = {`OP_OUT, 1'd0, 8'h00, 1'd0,1'd1,8'd4};
assign testdata[4] = {`OP_JMP, 1'd0, 8'h02, 1'd0,1'd1,8'd12};
assign testdata[5] = {`OP_BR, 1'd0, 8'h03, 1'd0,1'd1,8'd5};
assign testdata[6] = {`OP_LOADLO, 1'd0, 8'h02, 1'd0,1'd1,8'd8};
assign testdata[7] = {`OP_LOADHI, 1'd0, 8'h01, 1'd0,1'd1,8'd2};
```

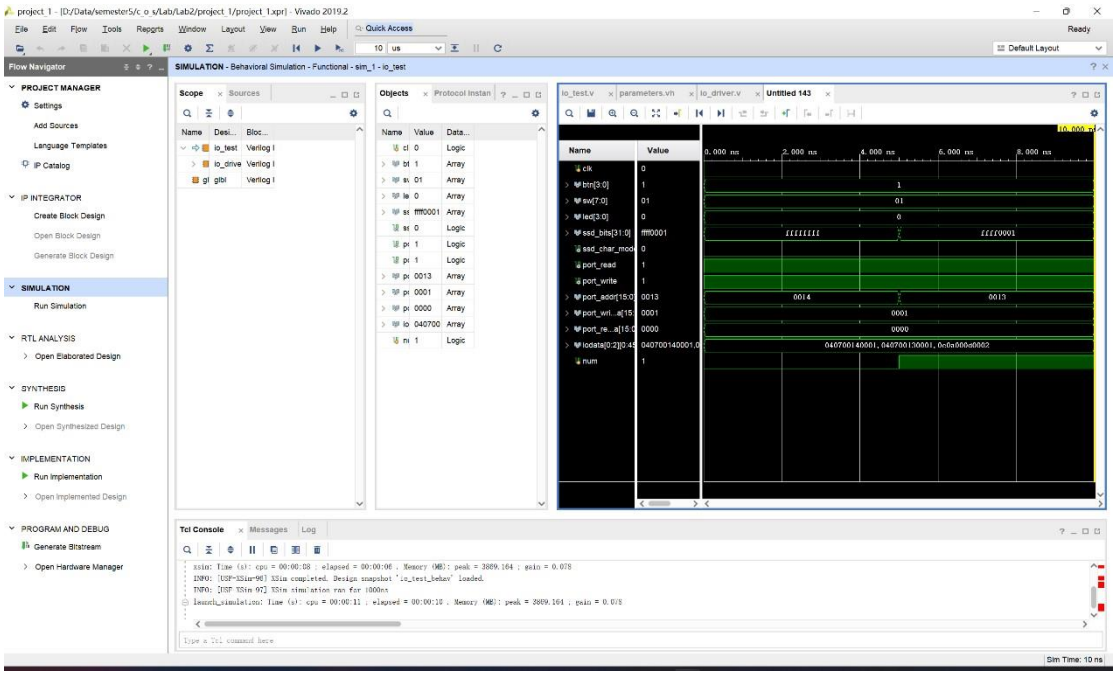


```
# set curr_wave [current_wave_config]
# if { [string length $curr_wave] == 0 } {
#   if { [llength [get_objects]] > 0} {
#     add_wave /
#     set_property needs_save false [current_wave_config]
#   } else {
#     send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave window. If you want to open a wave window go to 'File->New Waveform Configuration' or type 'create_wave_config' in the TCL cons
#   }
# }
# run 1000ns
opcode =  8, isaluop = 0, bus_addr =  2, bus_read = 0, bus_write = 1, bus_data =  10
state =  0, opcode =  8
opcode =  9, isaluop = 0, bus_addr =  3, bus_read = 0, bus_write = 1, bus_data =  14
state =  0, opcode =  9
opcode = 10, isaluop = 0, bus_addr =  3, bus_read = 0, bus_write = 1, bus_data =  4
state =  0, opcode = 10
opcode = 11, isaluop = 0, bus_addr =  0, bus_read = 0, bus_write = 1, bus_data =  4
state =  0, opcode = 11
opcode = 12, isaluop = 0, bus_addr =  2, bus_read = 0, bus_write = 1, bus_data =  12
state =  4, opcode = 12
opcode = 13, isaluop = 0, bus_addr =  3, bus_read = 0, bus_write = 1, bus_data =  5
state =  7, opcode = 13
opcode = 14, isaluop = 0, bus_addr =  2, bus_read = 0, bus_write = 1, bus_data =  8
state =  8, opcode = 14
$stop called at time : 75 ns : File "D:/Data/semester5/c_o_s/Lab/Lab2/project_4/project_4.srcs/sim_1/new/control_tb.v" Line 49
xsim: Time (s): cpu = 00:00:09 ; elapsed = 00:00:06 . Memory (MB): peak = 3927.090 ; gain = 0.000
INFO: [USF-XSim-96] XSim completed. Design snapshot 'control_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:12 ; elapsed = 00:00:10 . Memory (MB): peak = 3927.090 ; gain = 0.000
```

3. **IO module** (below shows the test data and the result **when the port is set as PORT_TO_BITS**, there are **7 ports type**, and **you can check all 7 results when the ports choose different types in the screen directory**)

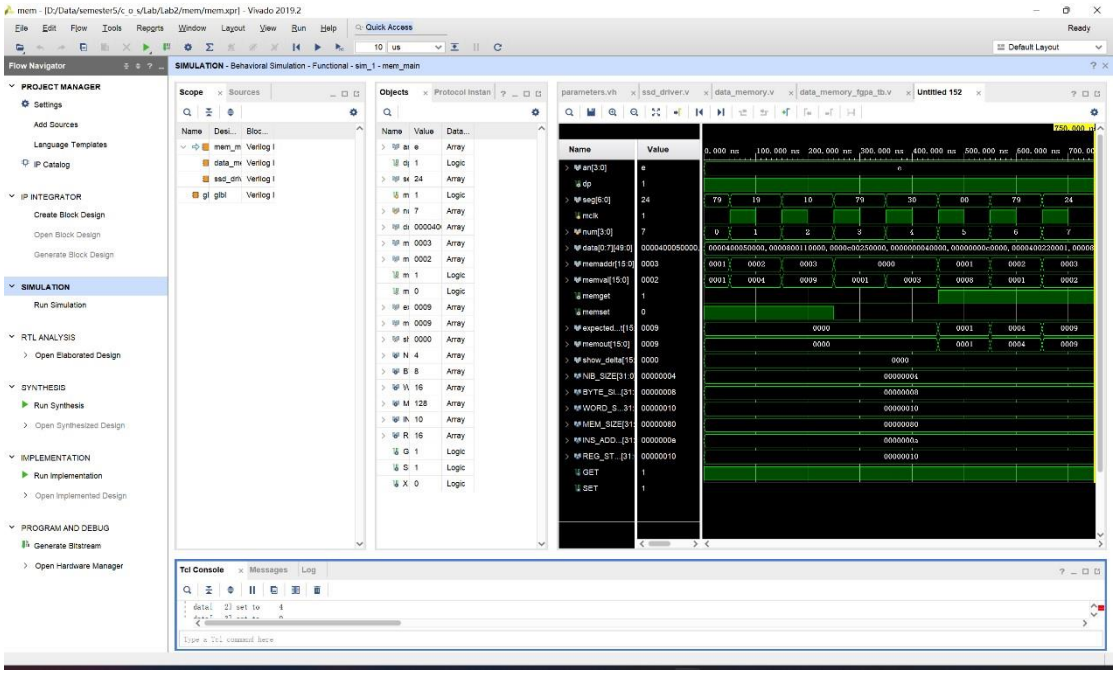
```
assign iodata[0] = {4'b1, 8'b00000001, 1'b1, 1'b1, `PORT_IO_LED, 16'h0001};
assign iodata[1] = {4'b1, 8'b00000001, 1'b1, 1'b1, `PORT_IO_HEX, 16'h0001};
assign iodata[2] = {4'b1, 8'b00000001, 1'b1, 1'b1, `PORT_IO_DEC, 16'h0001};
assign iodata[3] = {4'b1, 8'b00000001, 1'b1, 1'b1, `PORT_IO_CHAR, 16'h0001};
assign iodata[4] = {4'b1, 8'b00000001, 1'b1, 1'b1, `PORT_IO_BITS, 16'h0001};
assign iodata[5] = {4'b1, 8'b00000001, 1'b1, 1'b1, `PORT_IO_SWITCH, 16'h0001};
assign iodata[6] = {4'b1, 8'b00000001, 1'b1, 1'b1, `PORT_IO_BUTTON, 16'h0001};
```

Below is shown the result when port is set as PORT_TO_BITS



4. **Memory module** (below shows the test data set, the result and the message in the console)

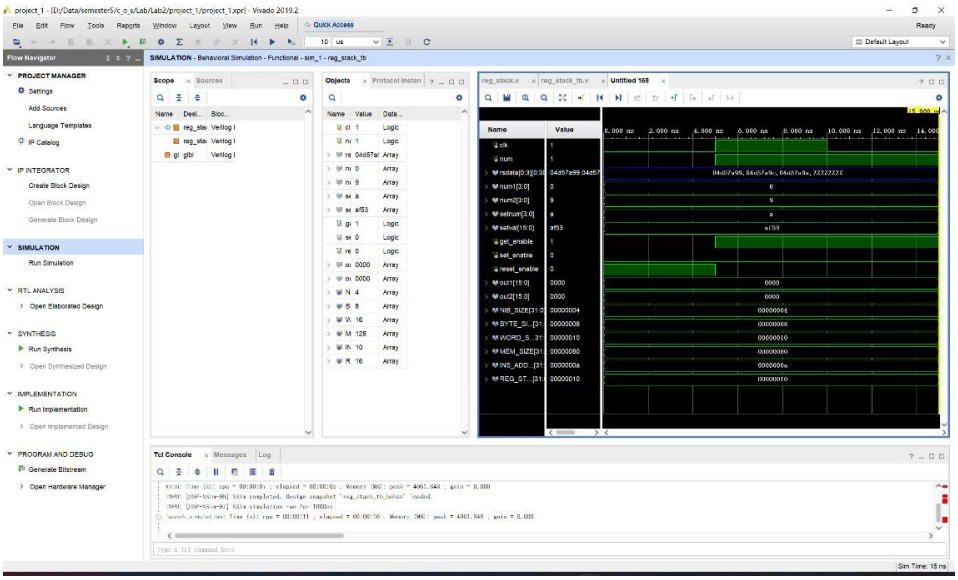
```
assign data[0] = { 16'h1, 16'h1, X, SET, 16'h0 };
assign data[1] = { 16'h2, 16'h4, X, SET, 16'h0 };
assign data[2] = { 16'h3, 16'h9, X, SET, 16'h0 };
assign data[3] = { 16'h0, 16'h1, X, X, 16'h0 };
assign data[4] = { 16'h0, 16'h3, X, X, 16'h0 };
assign data[5] = { 16'h1, 16'h8, GET, X, 16'h1 };
assign data[6] = { 16'h2, 16'h1, GET, X, 16'h4 };
assign data[7] = { 16'h3, 16'h2, GET, X, 16'h9 };
```

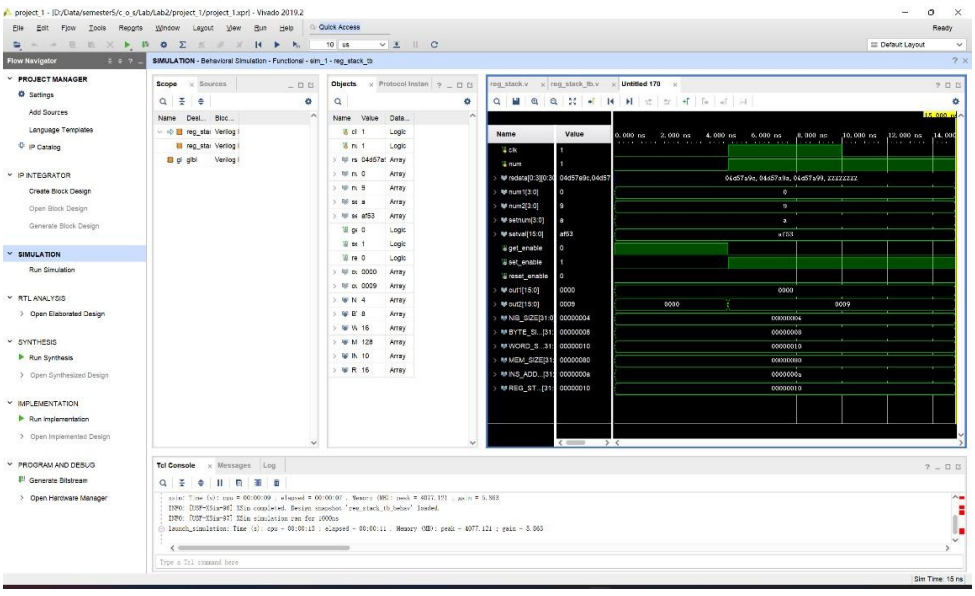


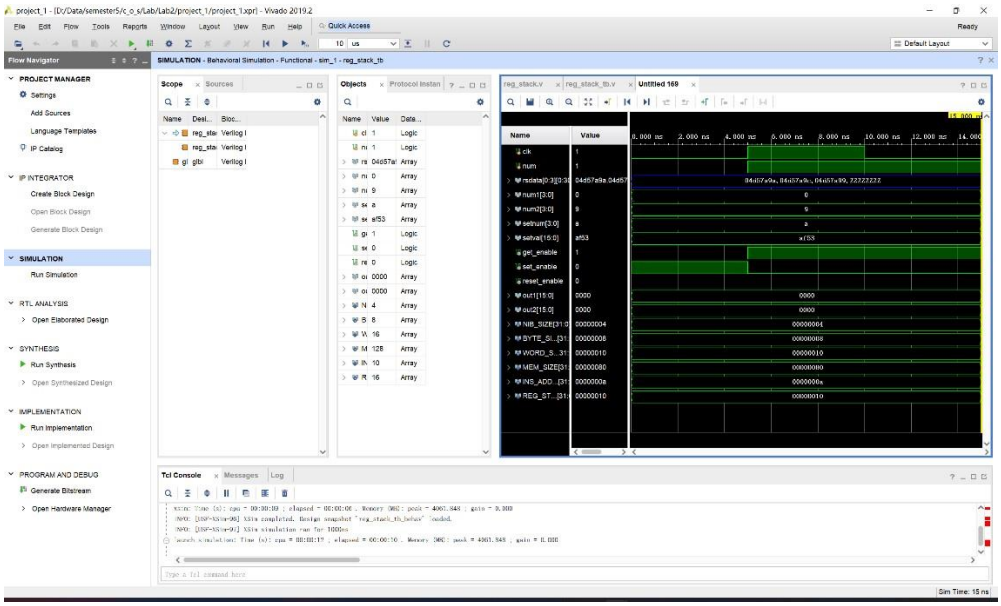
```
source mem_main.tcl
# set curr_wave [current_wave_config]
# if { [string length $curr_wave] == 0 } {
#   if { [llength [get_objects]] > 0} {
#     add_wave /
#     set_property needs_save false [current_wave_config]
#   } else {
#      send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave window. If you want to open a wave window go to 'File->New Waveform Configuration' or type 'create_wave_config' in the TCL cons
#   }
# }
# run 1000ns
data[   1] set to    1
data[   1] set to    1
data[   2] set to    4
data[   2] set to    4
data[   3] set to    9
data[   3] set to    9
   1 read from data[   1]
   4 read from data[   2]
   9 read from data[   3]
$stop called at time : 750 ns : File "D:/Data/semester5/c_o_s/Lab/Lab2/mem/mem.srcs/sim_1/new/data_memory_fpga_tb.v" Line 33
xsim: Time (s): cpu = 00:00:10 ; elapsed = 00:00:07 . Memory (MB): peak = 3933.715 ; gain = 0.000
INFO: [USF-XSim-96] XSim completed. Design snapshot 'mem_main_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:13 ; elapsed = 00:00:11 . Memory (MB): peak = 3933.715 ; gain = 0.000
```

5. **Register-stack module** (there are three types in the register-stack module, they are get, reset and set, the test data is shown below contains three types, and outputs of those three types are also shown in the below)

```
assign rsdata[0] = {4'b1001, 4'b1010, 16'b1010111101010011, 1'b1, 1'b0, 1'b0};
assign rsdata[1] = {4'b1001, 4'b1010, 16'b1010111101010011, 1'b0, 1'b1, 1'b0};
assign rsdata[2] = {4'b1001, 4'b1010, 16'b1010111101010011, 1'b0, 1'b0, 1'b1};
```

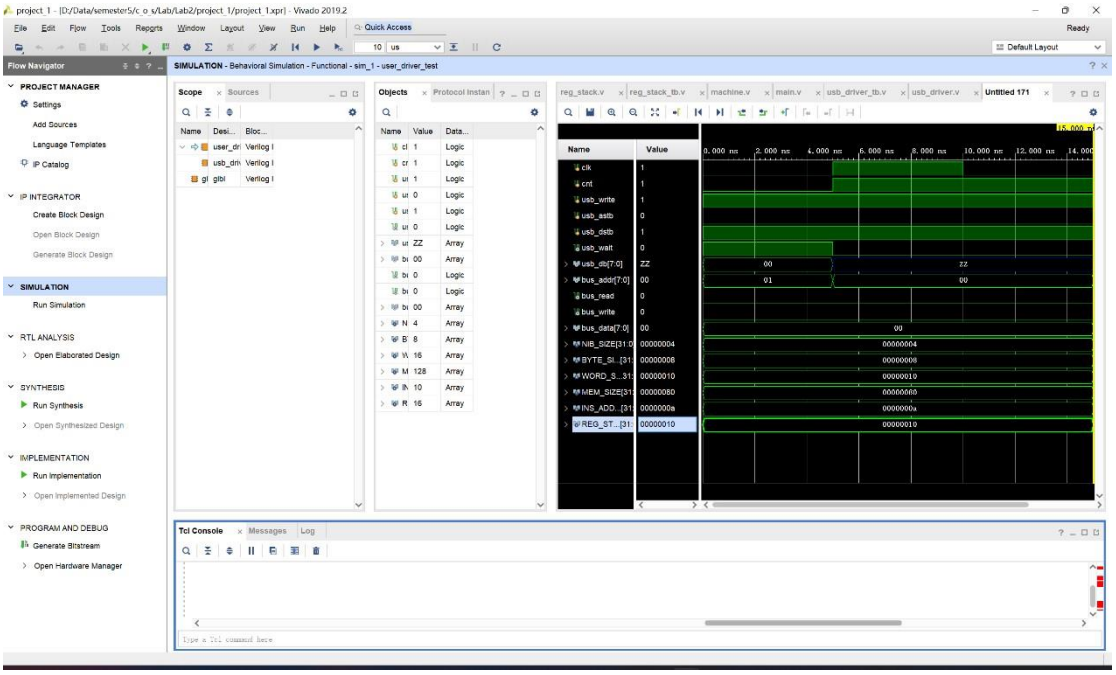






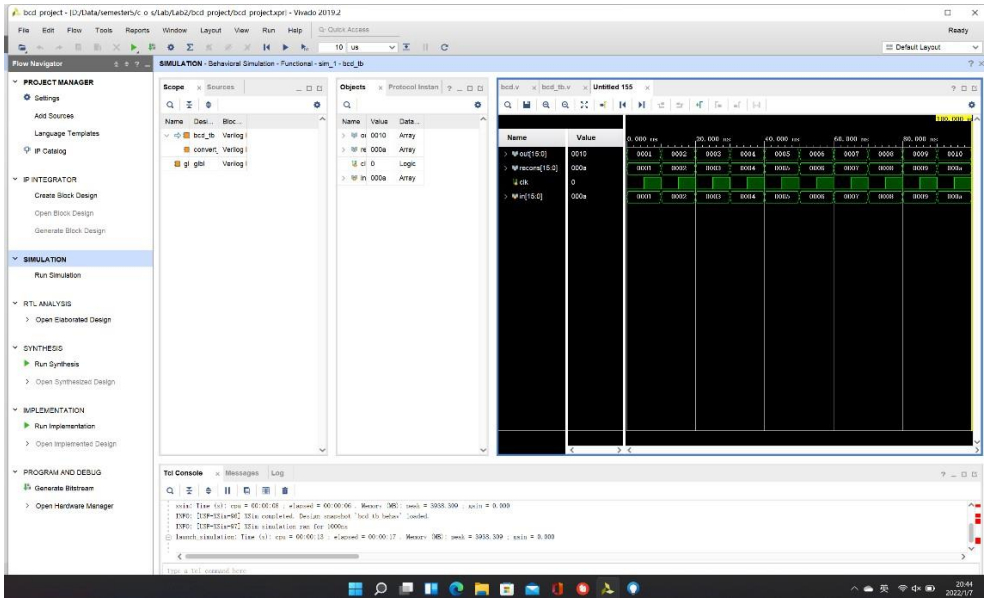
6. **USB driver module** (test data and result)

```
reg usb_write = 1;
reg usb_astb = 0;
reg usb_dstb = 1;
```



7. **bcd module** (message in the console and the result, as you can see the in==recons which means the bcd module runs successfully)

```
in =     1, out = 0001, recons =     1, ok = 1
in =     2, out = 0002, recons =     2, ok = 1
in =     3, out = 0003, recons =     3, ok = 1
in =     4, out = 0004, recons =     4, ok = 1
in =     5, out = 0005, recons =     5, ok = 1
in =     6, out = 0006, recons =     6, ok = 1
in =     7, out = 0007, recons =     7, ok = 1
in =     8, out = 0008, recons =     8, ok = 1
in =     9, out = 0009, recons =     9, ok = 1
```



8. **total machine** (test data set and the result)

```
assign machdb[0]={8'b00000001,4'b0000,1'b1,1'b1,1'b1};
assign machdb[1]={8'b00000010,4'b0001,1'b1,1'b0,1'b0};
assign machdb[2]={8'b00000011,4'b0010,1'b0,1'b1,1'b0};
assign machdb[3]={8'b00000100,4'b0011,1'b0,1'b0,1'b1};
```