THIẾT KẾ HỆ THỐNG SỐ VỚI HDL

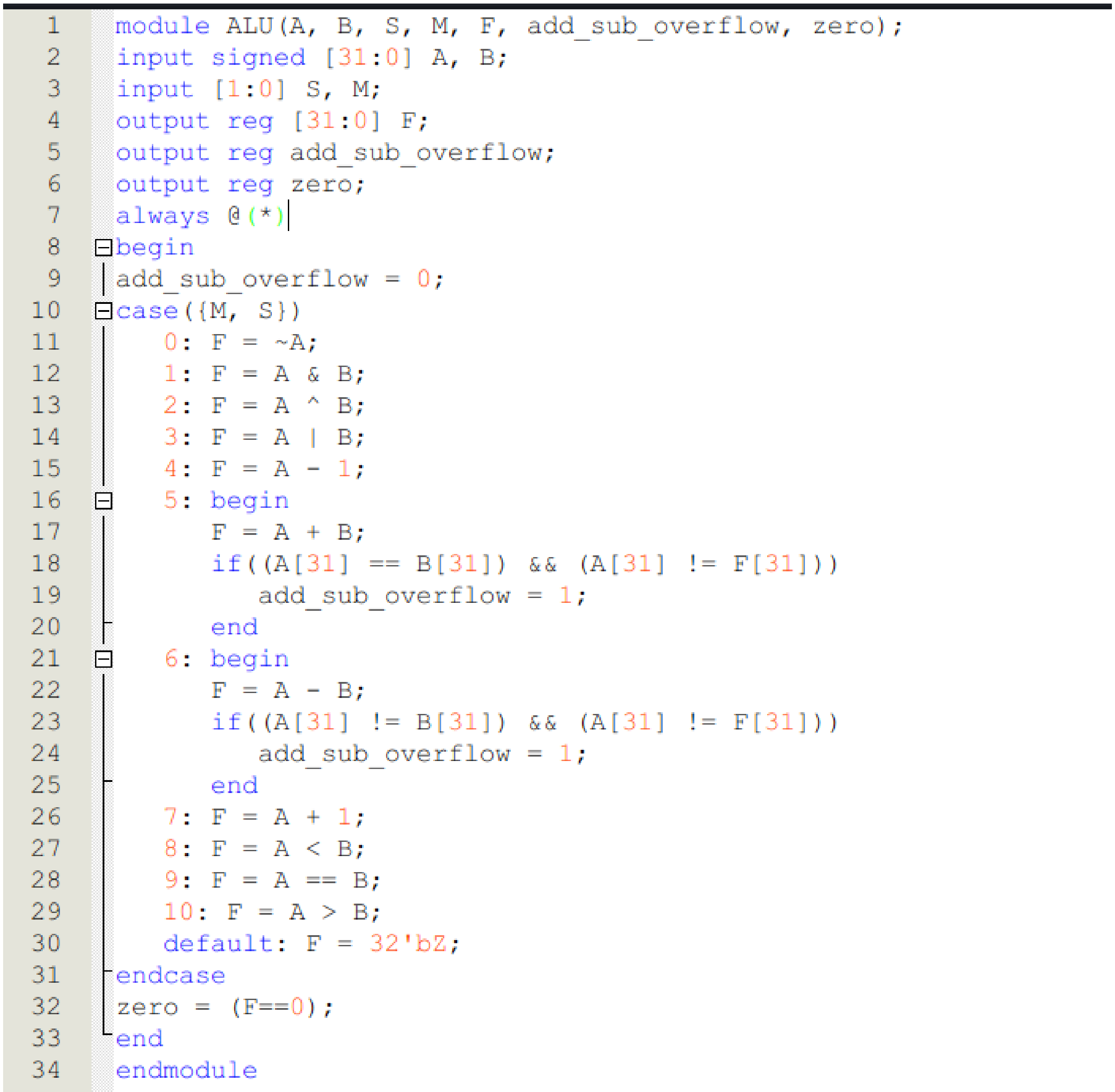
BÀI THỰC HÀNH 4

GVHD: Tạ Trí Đức

Sinh viên thực hiện: Phạm Quốc Tiến – 22521472

1) Thiết kế ALU

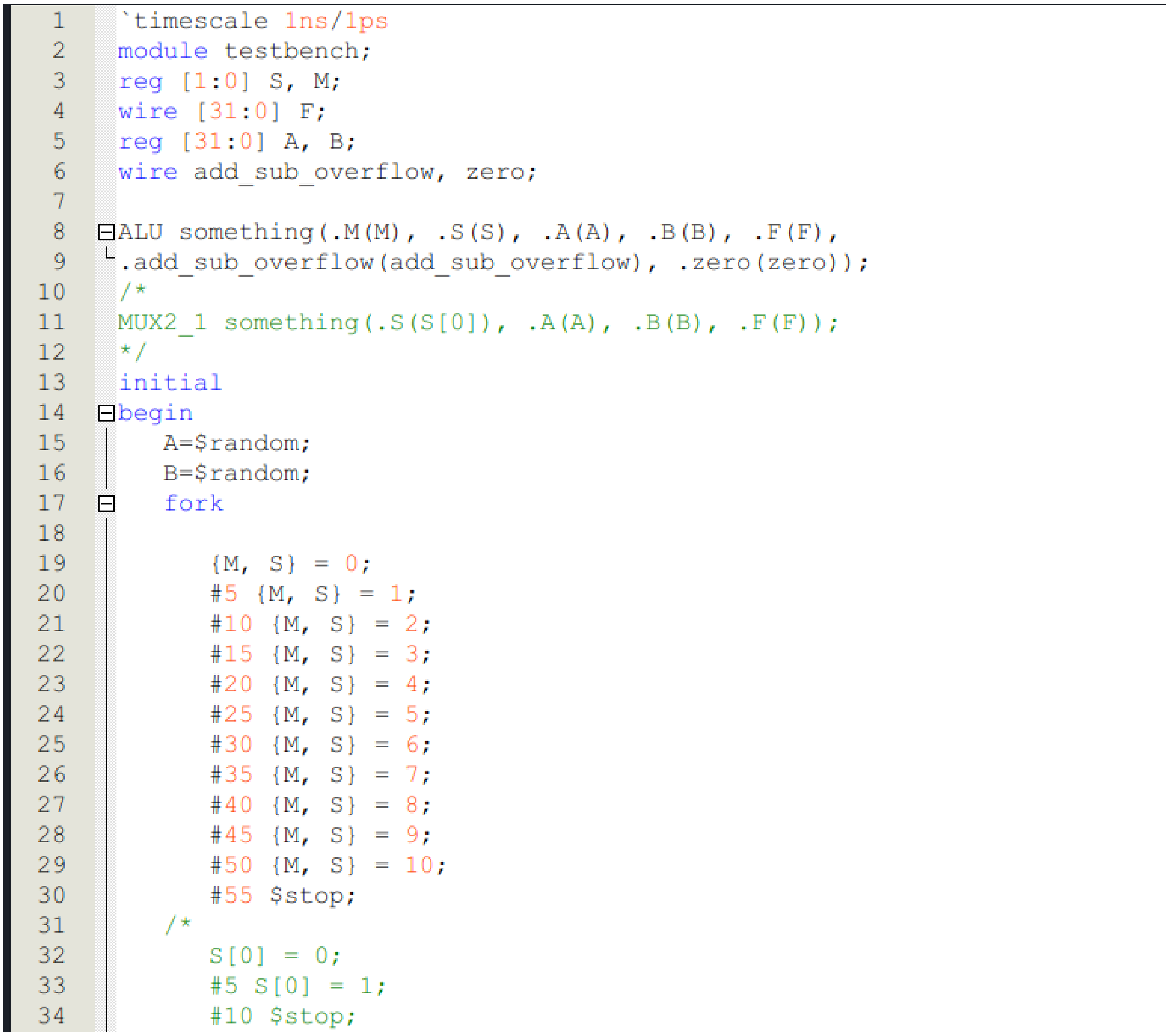
Code:



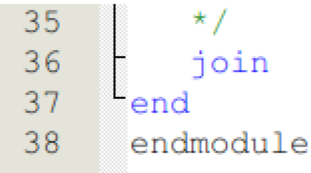
```
module ALU(A, B, S, M, F, add_sub_overflow, zero);
input signed [31:0] A, B;
input [1:0] S, M;
output reg [31:0] F;
output reg add_sub_overflow;
output reg zero;
always @(*)
begin
add_sub_overflow = 0;
case({M, S})
   0: F = ~A;
   1: F = A & B;
   2: F = A ^ B;
   3: F = A | B;
   4: F = A - 1;
   5: begin
      F = A + B;
      if((A[31] == B[31]) && (A[31] != F[31]))
         add_sub_overflow = 1;
      end
   6: begin
      F = A - B;
      if((A[31] != B[31]) && (A[31] != F[31]))
         add_sub_overflow = 1;
      end
   7: F = A + 1;
   8: F = A < B;
   9: F = A == B;
   10: F = A > B;
   default: F = 32'bZ;
endcase
zero = (F==0);
end
endmodule
```

RTL:

Testbench:

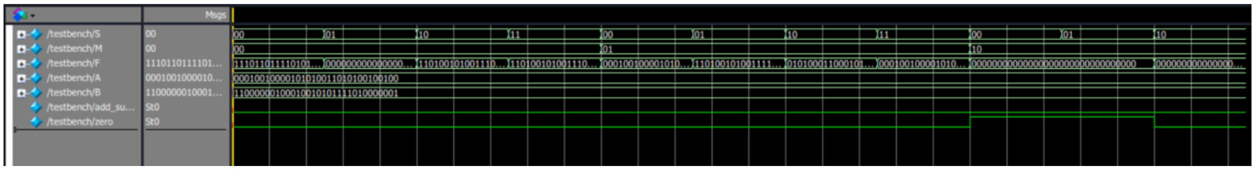


```
`timescale 1ns/1ps
module testbench;
reg [1:0] S, M;
wire [31:0] F;
reg [31:0] A, B;
wire add_sub_overflow, zero;

ALU something(.M(M), .S(S), .A(A), .B(B), .F(F),
.add_sub_overflow(add_sub_overflow), .zero(zero));
/*
MUX2_1 something(.S(S[0]), .A(A), .B(B), .F(F));
*/
initial
begin
   A=$random;
   B=$random;
   fork

      {M, S} = 0;
      #5 {M, S} = 1;
      #10 {M, S} = 2;
      #15 {M, S} = 3;
      #20 {M, S} = 4;
      #25 {M, S} = 5;
      #30 {M, S} = 6;
      #35 {M, S} = 7;
      #40 {M, S} = 8;
      #45 {M, S} = 9;
      #50 {M, S} = 10;
      #55 $stop;
   /*
      S[0] = 0;
      #5 S[0] = 1;
      #10 $stop;
```



```
      */
   join
end
endmodule
```

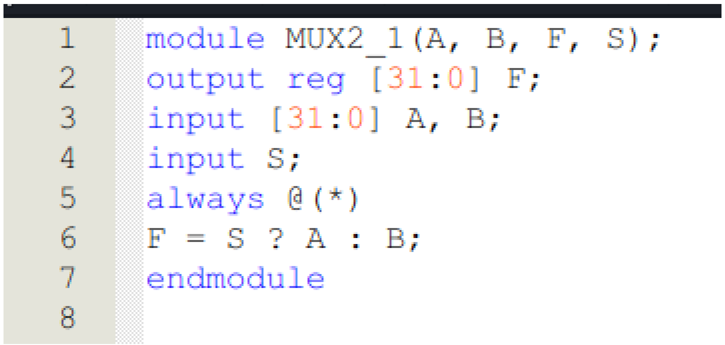
Sóng mô phỏng:



```
/testbench/S          00              00 | 01 | 10 | 11 | 00 | 01 | 10 | 11 | 00 | 01 | 10
/testbench/M          00              00 | 01 | 10
/testbench/F          1110110111101...  1110110111110101... | 0000000000000000... | 1101001010011110... | 1101001010011110... | 0001001000001010... | 1101001010011110... | 0101000110001010... | 0001001000001010... | 0000000000000000... | 00000000000000000000000000000000 | 0000000000000000...
/testbench/A          0001001000010...  00010010000101010011010100100100
/testbench/B          1100000010001...  11000000100010010101111010000001
/testbench/add_su...  St0
/testbench/zero       St0
```

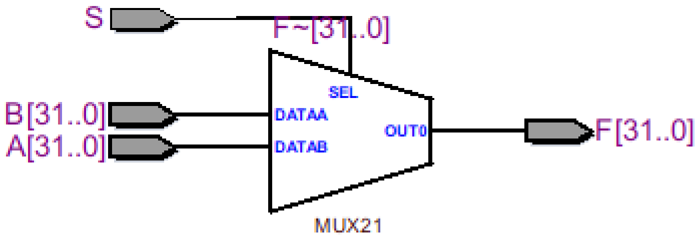
2) Thiết kế Mux2_1 32bits

Code:

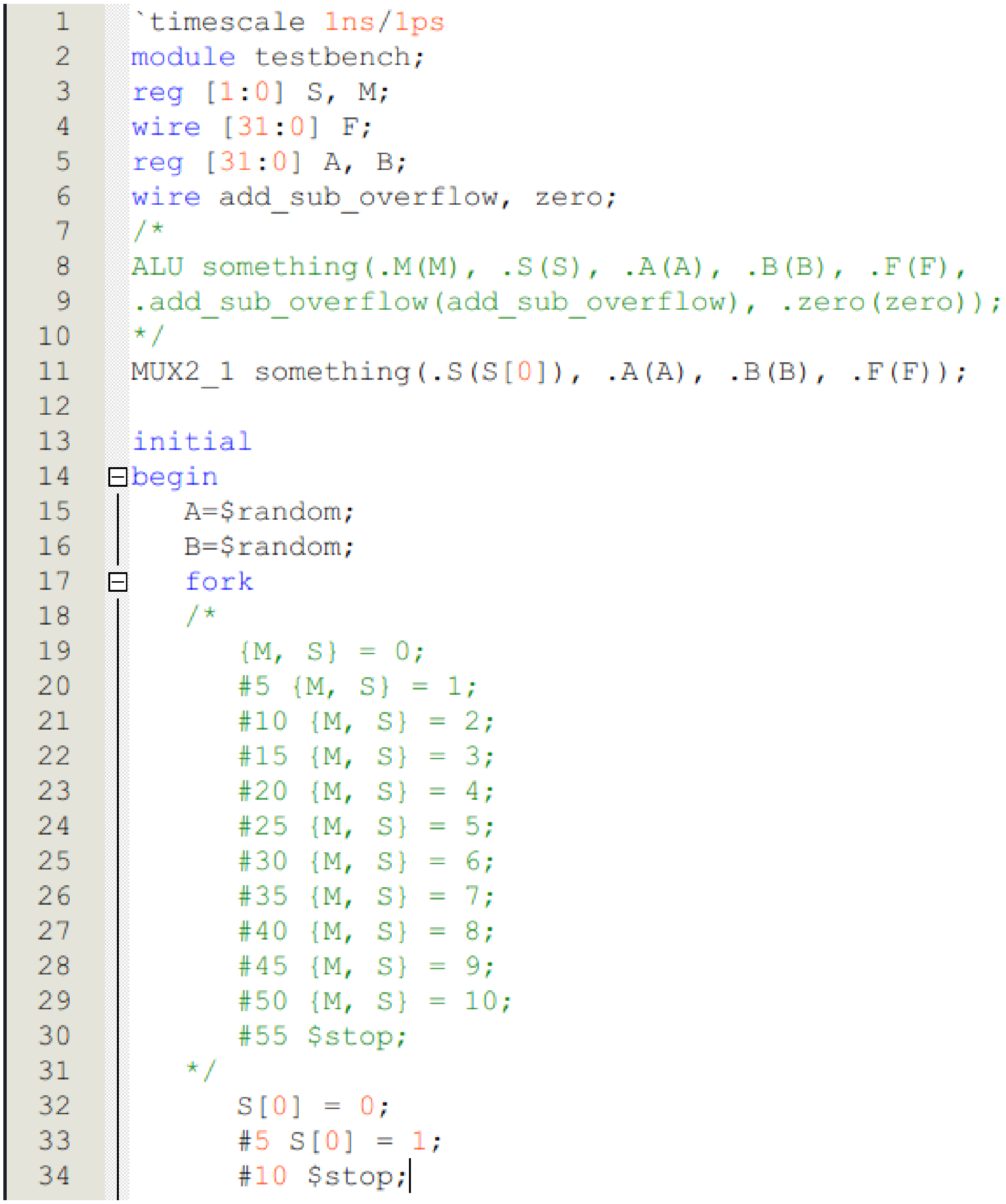


```
module MUX2_1(A, B, F, S);
output reg [31:0] F;
input [31:0] A, B;
input S;
always @(*)
F = S ? A : B;
endmodule
```

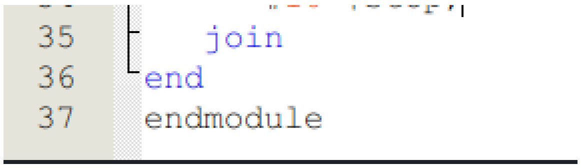
RTL:



Testbench:

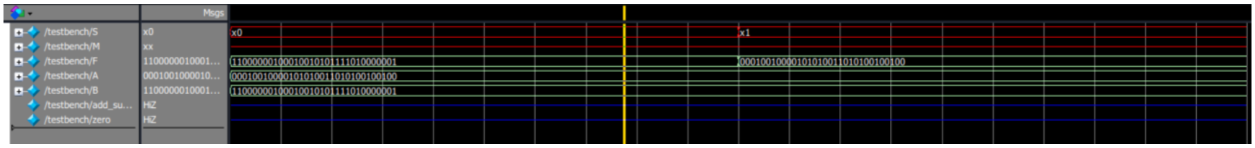


```
`timescale 1ns/1ps
module testbench;
reg [1:0] S, M;
wire [31:0] F;
reg [31:0] A, B;
wire add_sub_overflow, zero;
/*
ALU something(.M(M), .S(S), .A(A), .B(B), .F(F),
.add_sub_overflow(add_sub_overflow), .zero(zero));
*/
MUX2_1 something(.S(S[0]), .A(A), .B(B), .F(F));

initial
begin
    A=$random;
    B=$random;
    fork
    /*
        {M, S} = 0;
        #5 {M, S} = 1;
        #10 {M, S} = 2;
        #15 {M, S} = 3;
        #20 {M, S} = 4;
        #25 {M, S} = 5;
        #30 {M, S} = 6;
        #35 {M, S} = 7;
        #40 {M, S} = 8;
        #45 {M, S} = 9;
        #50 {M, S} = 10;
        #55 $stop;
    */
        S[0] = 0;
        #5 S[0] = 1;
        #10 $stop;
```



```
    join
end
endmodule
```

Sóng mô phỏng:



```
/testbench/S          x0               x0                                 x1
/testbench/M          xx
/testbench/F          1100000010001...  11000000100010010101111010000001   00010010000101010011010100100100
/testbench/A          0001001000010...  00010010000101010011010100100100
/testbench/B          1100000010001...  11000000100010010101111010000001
/testbench/add_su...  HiZ
/testbench/zero       HiZ
```