HomeWork #1

Test 3

a. The main code of project

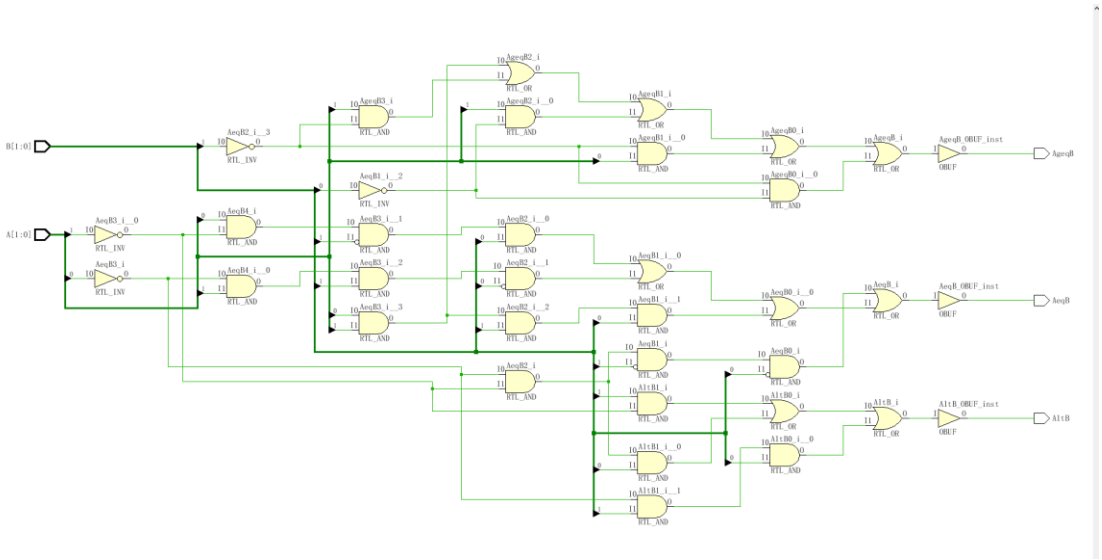
We can get the expression which I have written on handout from the K-maps. And then we can write code of comparatorExpression module and comparatorGates module.

```
1.  module comparatorExpression(A, B, AeqB, AgeqB, AltB );
2.  input [1:0] A, B;
3.  output  AeqB, AgeqB, AltB;
4.
5.  //assign AeqB = (A === B);
6.  //assign AgeqB = (A >= B);
7.  //assign AltB = (A < B);
8.
9.  assign AeqB = (~A[0] && ~A[1] && ~B[1] && ~B[0]) || (A[0] && ~A[1] && ~B[1]
    && B[0] ||  (~A[0] && A[1] && B[1] && ~B[0]) || (A[0] && A[1] && B[1] && B[0
    ]));
10. assign AgeqB = (A[0] && A[1]) || (A[1] && ~B[1]) ||  (A[1] && ~B[0]) || (~B[
    1] && A[0]) || (~B[1] && ~B[0]);
11. assign AltB = (B[1] && ~A[1]) || (~A[1] && ~A[0] && B[0]) ||  (~A[0] && B[1]
     && B[0]);
12.
13. endmodule
```

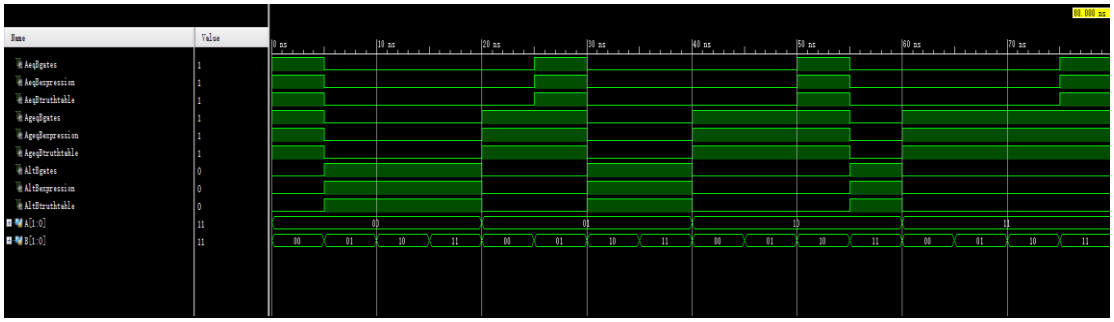
```
1.  module comparatorGates(A, B, AeqB, AgeqB, AltB );
2.  input [1:0] A, B;
3.  output  AeqB, AgeqB, AltB;
4.  wire aeqU1, aeqU2, aeqU3, aeqU4;
5.  wire agaeqU1, agaeqU2, agaeqU3, agaeqU4, ageqU5;
6.  wire altU1, altU2, altU3;
7.
8.  and aeqU1and(aeqU1,~A[0], ~A[1], ~B[1], ~B[0]);
9.  and aeqU2and(aeqU2,A[0], ~A[1], ~B[1], B[0]);
10. and aeqU3and(aeqU3,~A[0], A[1], B[1], ~B[0]);
11. and aeqU4and(aeqU4,A[0], A[1], B[1], B[0]);
```

```
12. or aeqOR(AeqB, aeqU1, aeqU2, aeqU3, aeqU4);
13.
14. and agaeqU1and(agaeqU1,~B[0], ~B[1]);
15. and agaeqU2and(agaeqU2,~B[1], A[1]);
16. and agaeqU3and(agaeqU3,A[1], A[0]);
17. and agaeqU4and(agaeqU4,A[1], ~B[0]);
18. and ageqU5and(ageqU5,~B[1], A[0]);
19. or ageqOR(AgeqB, agaeqU1, agaeqU2, agaeqU3, agaeqU4, ageqU5);
20.
21. and altU1and(altU1,B[1], ~A[1]);
22. and altU2and(altU2, ~A[1], ~A[0], B[0]);
23. and altU3and(altU3,~A[0], B[1], B[0]);
24. or altOR(AltB,altU1, altU2, altU3);
25.
26. endmodule
```

b. The RTL of project



c. The simulation of project



Test 4

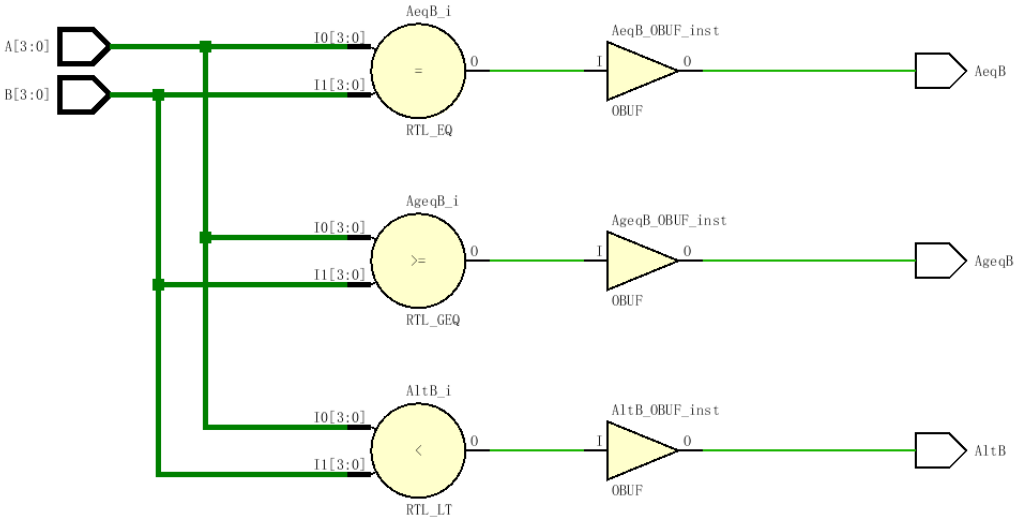
a. The main code of project

We can easily write the comparator4bit module below through reading test heading.

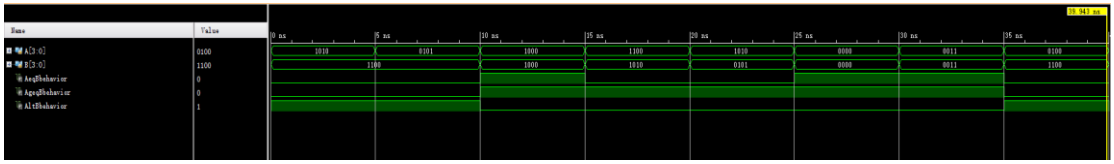
```
module comparator4bit(A, B, AeqB, AgeqB, AltB );
input [3:0] A, B;
output  AeqB, AgeqB, AltB;

assign AeqB = (A == B);
assign AgeqB = (A >= B);
assign AltB = (A < B);

endmodule
```

```
module comparitor4bits_TB;

reg[3:0] A,B;
wire AeqBbehavior,AgeqBbehavior,AltBbehavior;
comparator4bit UnitBehavior
(A,B,AeqBbehavior,AgeqBbehavior,AltBbehavior);

initial begin
    A=10;B=12;#5;
    A=5;B=12;#5;
    A=8;B=8;#5;
    A=12;B=10;#5;
    A=10;B=5;#5;
    A=0;B=0;#5;
    A=3;B=3;#5;
    A=20;B=12;#5;
    $stop;
end
endmodule
```

b. The RTL of project



c. The simulation of project



Test 8

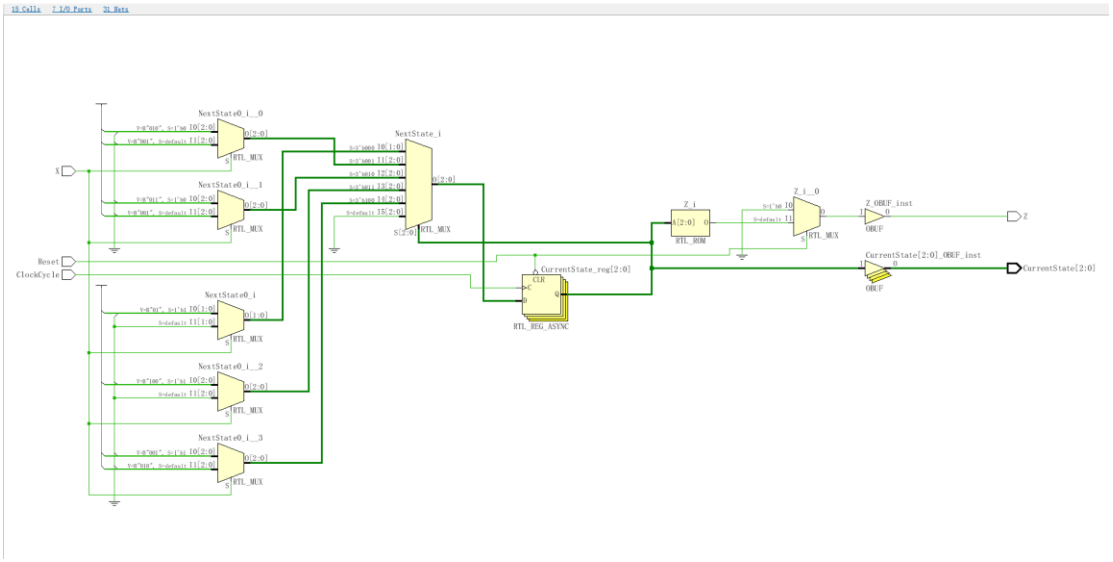
a. The main code of project

We can know the state machine by the heading and the sample timing sequence, and then we could write a state machine module. Also from that the RTL graph and simulation will come out.

```
1.  module statemachine(X, Z, CurrentState, Reset, ClockCycle);
2.  input X, Reset, ClockCycle;
3.  output reg Z;
4.  output reg [2:0] CurrentState;
5.  reg [2:0] NextState;
6.
7.  parameter Initial = 3'b000, S1 = 3'b001, S10 = 3'b010, S100 = 3'b011, S1001
    = 3'b100;
8.
9.  always @(negedge Reset or posedge  ClockCycle)begin
10.         if(Reset == 0)
11.             CurrentState <= Initial;
12.         else
13.             CurrentState <= NextState;
14.     end
15.
16. always @ (Reset or CurrentState)begin
17.         if(Reset == 0)
18.             Z = 0;
19.         else
20.             case(CurrentState)
21.                 Initial: Z = 0;
22.                 S1: Z = 0;
23.                 S10: Z = 0;
24.                 S100: Z = 0;
25.                 S1001: Z = 1;
26.                 default: Z = 0;
27.         endcase
28.     end
29.
30. always @(X or CurrentState)
31.     begin
```

```
32.         case (CurrentState)
33.             Initial:    NextState = (X==1)?S1:Initial;
34.             S1:     NextState = (X==0)?S10:S1;
35.             S10:     NextState = (X==0)?S100:S1;
36.             S100:     NextState = (X==1)?S1001:Initial;
37.             S1001:     NextState = (X==1)?S1:S10;
38.             default:    NextState = Initial;
39.         endcase
40.     end
41.
42. endmodule
```

b.The RTL of project



c. The simulation of project