EE lab3 report： mini-IDS

Name：Chenyang Zhang

GitHub Link：https://github.com/zhangc74/ee533-lab3

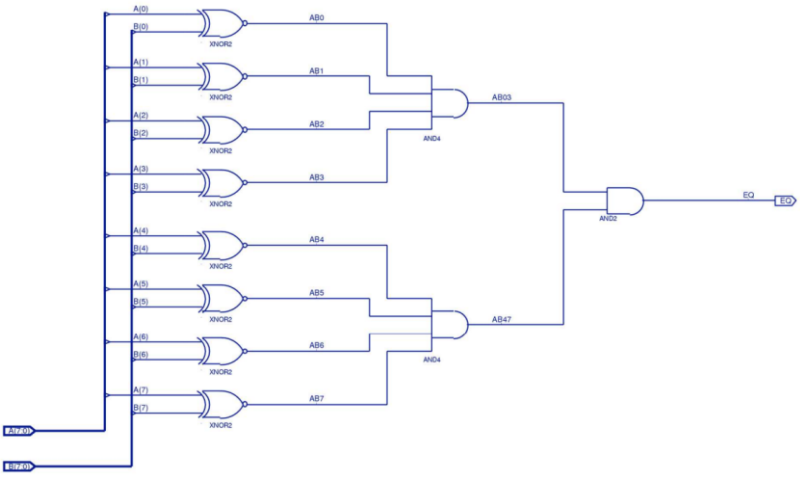
For this lab, I will create an intrusion detection system (mini-IDS) using schematics, IP Cores, and verilog.

Schematic Capture

1. Create busmerge.v
   Used to spell 48-bit + 64-bit into 112-bit, it is a "splicer" for the subsequent wordmatch

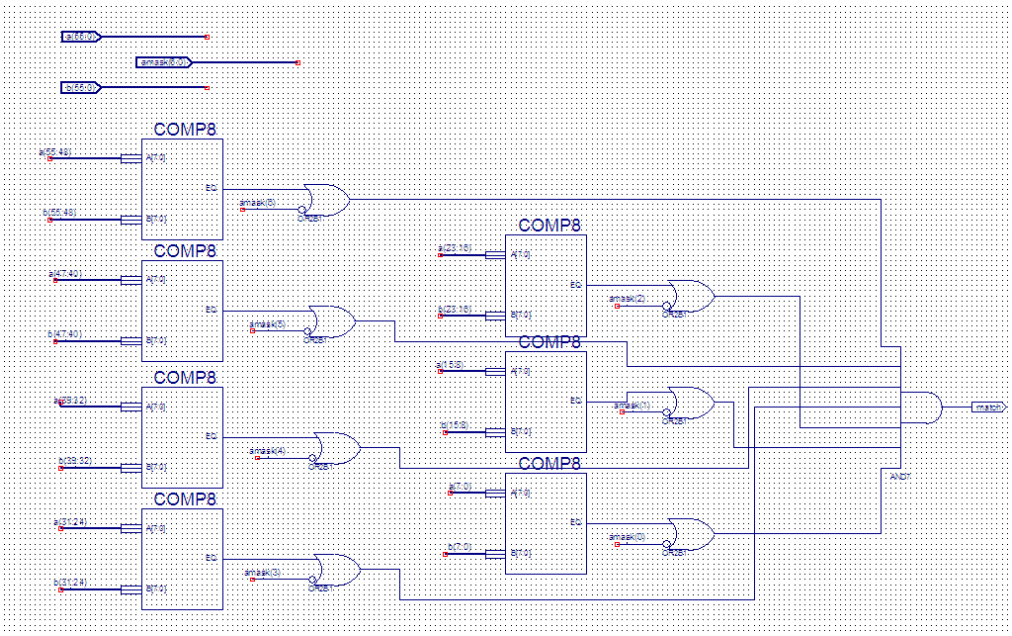
```
1  module busmerge(da, db, q);
2
3      input [47:0] da;
4      input [63:0] db;
5      output [111:0] q;
6      assign q = {da,db};
7
8  endmodule
9
```

2. Draw comp8.sch
   The 8-bit comparator is a built-in Xilinx component and is included for reference, so I do not need to re-implement it.　　input：A[7:0], B[7:0]　　output：EQ（equal = 1）



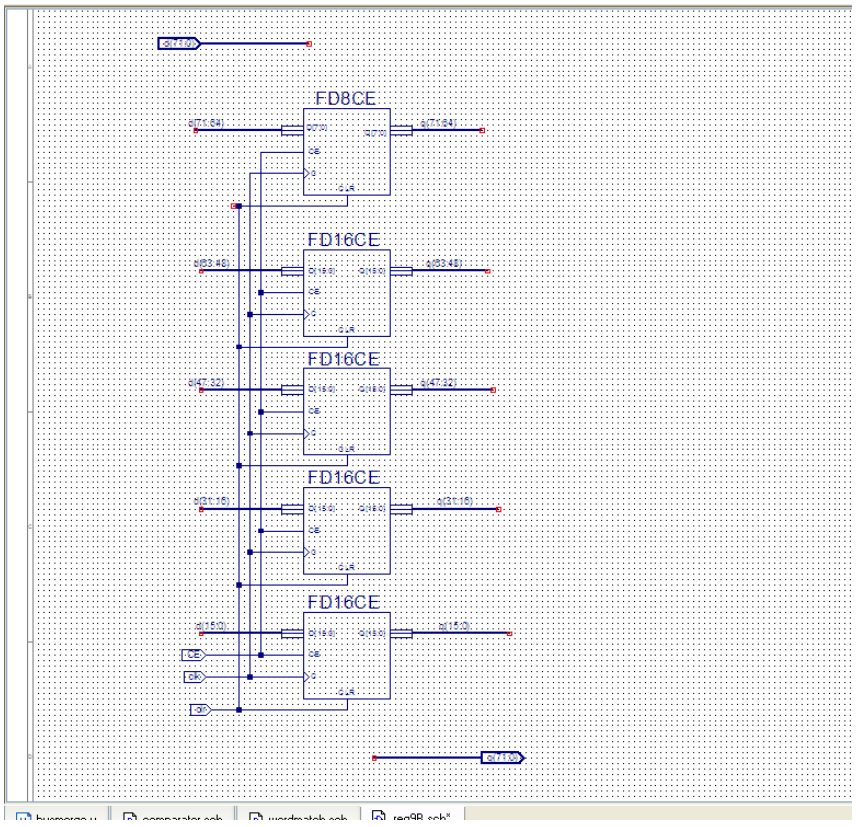
3. Draw comparator.sch
   Used to compare a 56-bit input. It is built using seven 8-bit comparators, where each comparator checks one byte of the data. The AMASK[6:0] signal allows some bytes to be ignored during comparison. The output match becomes 1 only when the required bytes

match the pattern.



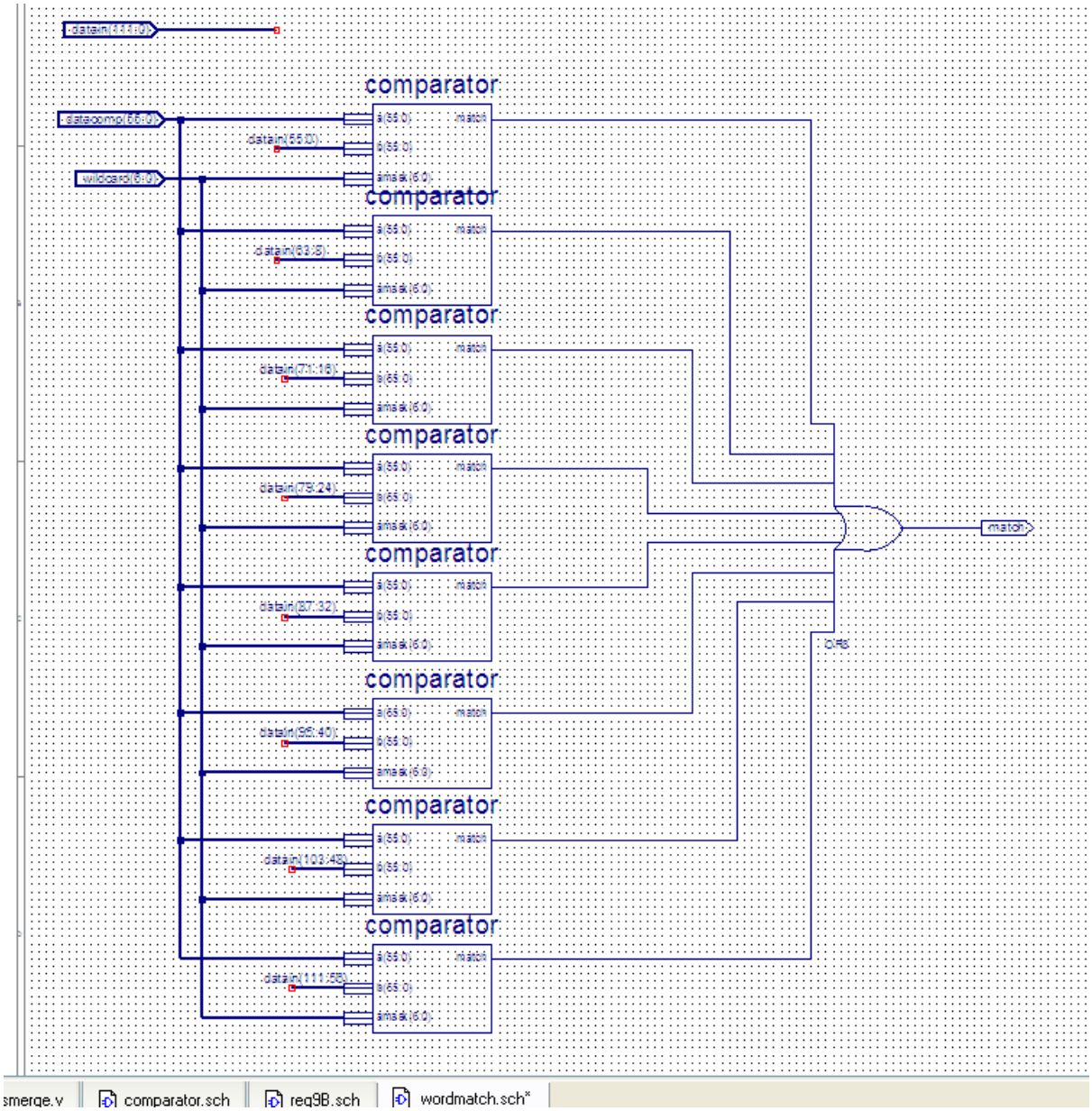
4. Draw reg9B.sch

   This is a 9-byte (72-bit) register used to store packet data. It is built using one 8-bit register and four 16-bit registers connected together. All registers share the same clock, enable, and clear signals so that the entire 72-bit value is updated at the same time. This module is used to temporarily hold data before it is processed by other modules.



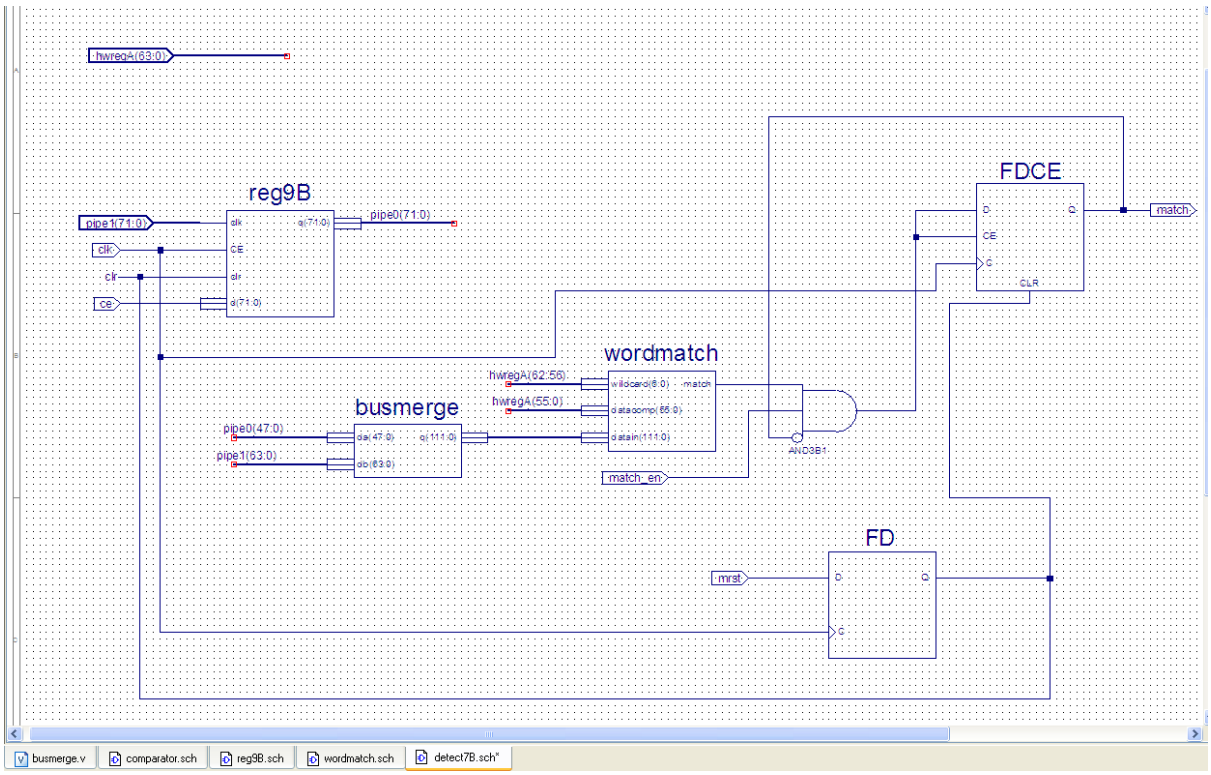
5. Draw wordmatch.sch

   Used to scan incoming data for a matching pattern. It takes a 112-bit input and splits it into several overlapping 56-bit segments. Each segment is compared with the target pattern using the comparator module. The match results from all comparators are combined using OR logic, so the output match becomes 1 if any segment matches the pattern.



6. Draw detect7B.sch

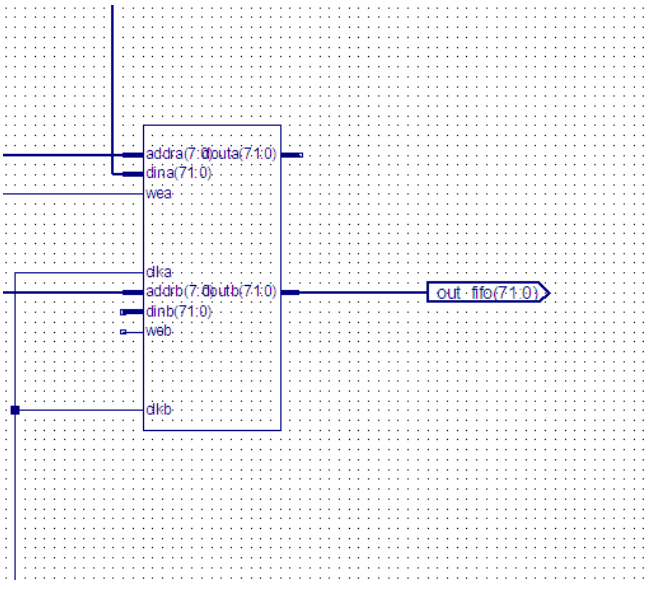
   Used to connect the pattern matching logic with the data flow control. It takes packet data, combines different data fields using busmerge, and sends the result to the wordmatch module for comparison. When a match is detected, control signals are generated and stored using flip-flops to indicate that a suspicious pattern has been found. This module acts as the main detection unit of the mini intrusion detection system.

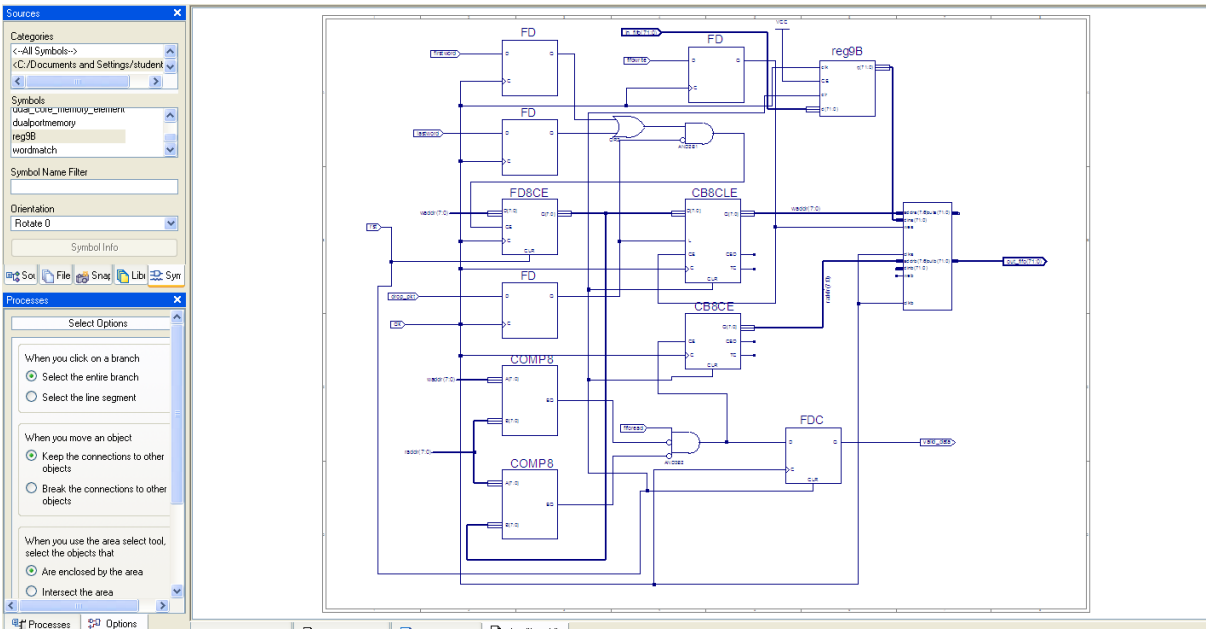


7. Draw dropfifo.sch

   Used to control whether packets are forwarded or dropped. It uses a FIFO memory to buffer packet data and comparison results. When the detection logic indicates a match, the module prevents the corresponding packet from being forwarded. This module simulates how an intrusion detection system can drop suspicious packets in a real network environment.

✓ Add a new IP Core component for a 9-byte (72 bit) wide synchronous dual port memory (1 read, 1 write port).





8. Convert all of my schematics to Verilog (ISE provides this feature).

1) Take a look at the created Verilog. Do they make sense? Which do you think easier: entering the schematics or writing Verilog? Why? In which cases might you do the other?

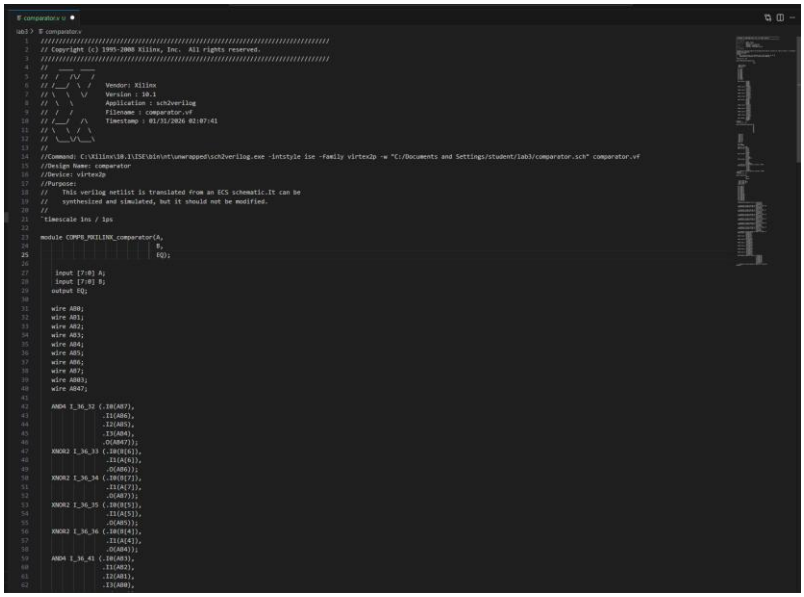
I think the generated Verilog files make sense.

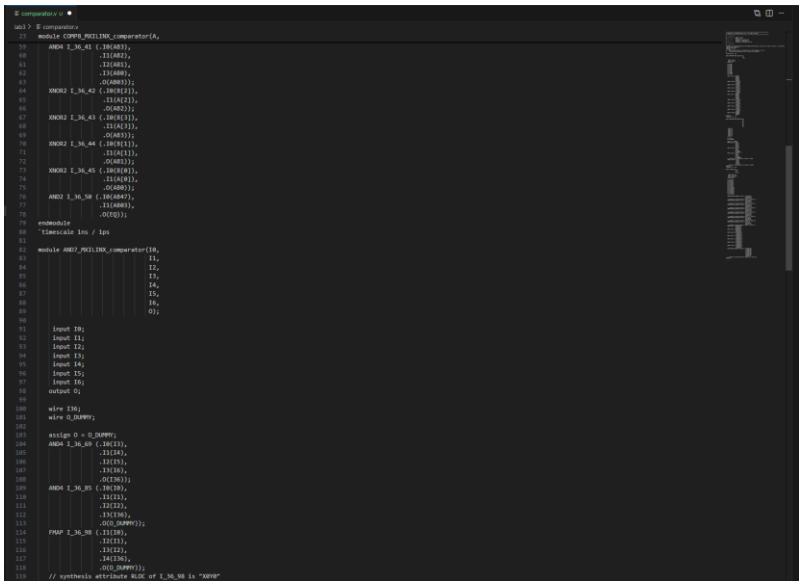
I think drawing schematics is easier because it is more visual and easier to understand how the system works.

However, for simple logic like a multiplexer, writing Verilog is faster and cleaner, especially when the data width is large, such as 64 bits.

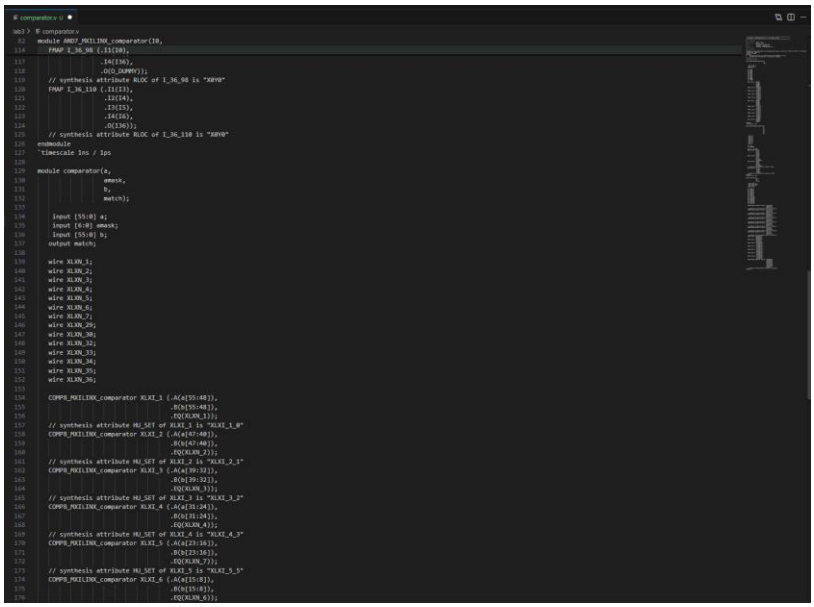
Generated Verilog files are following:

i) comparator.v

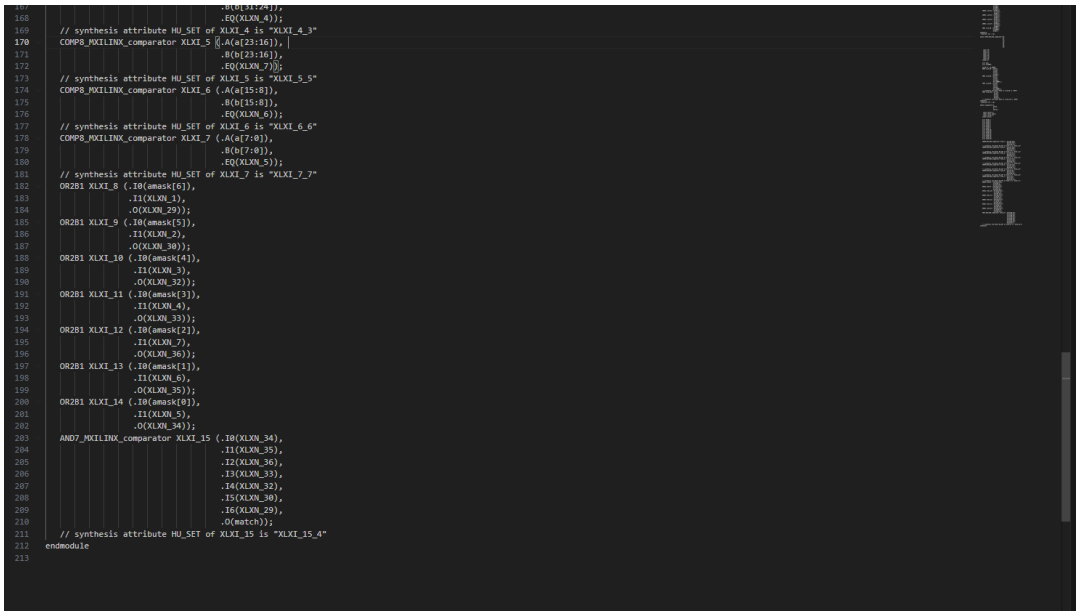




```
comparator.v
sob3 > comparator.v
  module COMP8_MXILINX_comparator(A,
    AND4 I_36_41 (.I0(A03),
                  .I1(A02),
                  .I2(A01),
                  .I3(A00),
                  .O(A003));
    XNOR2 I_36_42 (.I0(B[2]),
                   .I1(A[2]),
                   .O(A02));
    XNOR2 I_36_43 (.I0(B[3]),
                   .I1(A[3]),
                   .O(A03));
    XNOR2 I_36_44 (.I0(B[1]),
                   .I1(A[1]),
                   .O(A01));
    XNOR2 I_36_45 (.I0(B[0]),
                   .I1(A[0]),
                   .O(A00));
    AND2 I_36_50 (.I0(A047),
                  .I1(A003),
                  .O(EQ));
  endmodule
  `timescale 1ns / 1ps

  module AND7_MXILINX_comparator(I0,
                                 I1,
                                 I2,
                                 I3,
                                 I4,
                                 I5,
                                 I6,
                                 O);

    input I0;
    input I1;
    input I2;
    input I3;
    input I4;
    input I5;
    input I6;
    output O;

    wire I36;
    wire O_DUMMY;

    assign O = O_DUMMY;
    AND4 I_36_69 (.I0(I3),
                  .I1(I4),
                  .I2(I5),
                  .I3(I6),
                  .O(I36));
    AND4 I_36_85 (.I0(I0),
                  .I1(I1),
                  .I2(I2),
                  .I3(I36),
                  .O(O_DUMMY));
    FMAP I_36_98 (.I1(I0),
                  .I2(I1),
                  .I3(I2),
                  .I4(I36),
                  .O(O_DUMMY));
    // synthesis attribute RLOC of I_36_98 is "X0Y0"
```

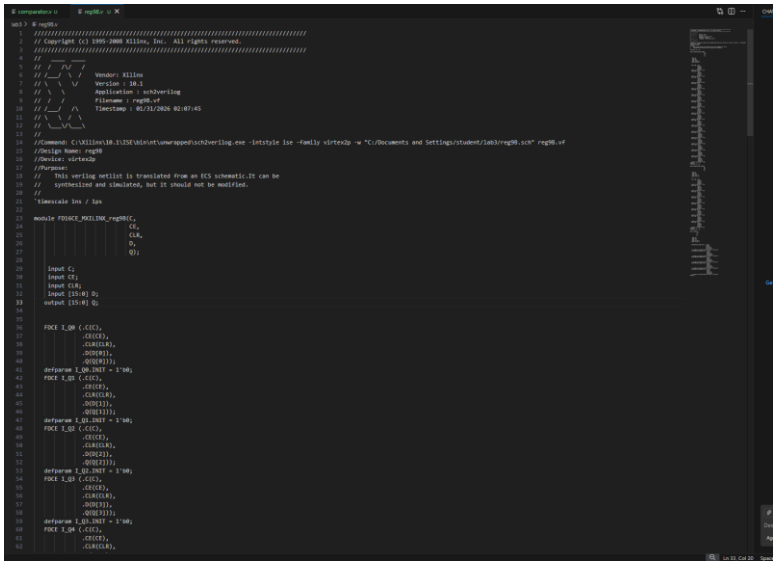


```
comparator.v
sob3 > comparator.v
  module AND7_MXILINX_comparator(I0,
    FMAP I_36_98 (.I1(I0),
                  .I4(I36),
                  .O(O_DUMMY));
    // synthesis attribute RLOC of I_36_98 is "X0Y0"
    FMAP I_36_110 (.I1(I3),
                   .I2(I4),
                   .I3(I5),
                   .I4(I6),
                   .O(I36));
    // synthesis attribute RLOC of I_36_110 is "X0Y0"
  endmodule
  `timescale 1ns / 1ps

  module comparator(a,
                    amask,
                    b,
                    match);

    input [55:0] a;
    input [6:0] amask;
    input [55:0] b;
    output match;

    wire XLXN_1;
    wire XLXN_2;
    wire XLXN_3;
    wire XLXN_4;
    wire XLXN_5;
    wire XLXN_6;
    wire XLXN_7;
    wire XLXN_29;
    wire XLXN_30;
    wire XLXN_32;
    wire XLXN_33;
    wire XLXN_34;
    wire XLXN_35;
    wire XLXN_36;

    COMP8_MXILINX_comparator XLXI_1 (.A(a[55:48]),
                                     .B(b[55:48]),
                                     .EQ(XLXN_1));
    // synthesis attribute HU_SET of XLXI_1 is "XLXI_1_0"
    COMP8_MXILINX_comparator XLXI_2 (.A(a[47:40]),
                                     .B(b[47:40]),
                                     .EQ(XLXN_2));
    // synthesis attribute HU_SET of XLXI_2 is "XLXI_2_1"
    COMP8_MXILINX_comparator XLXI_3 (.A(a[39:32]),
                                     .B(b[39:32]),
                                     .EQ(XLXN_3));
    // synthesis attribute HU_SET of XLXI_3 is "XLXI_3_2"
    COMP8_MXILINX_comparator XLXI_4 (.A(a[31:24]),
                                     .B(b[31:24]),
                                     .EQ(XLXN_4));
    // synthesis attribute HU_SET of XLXI_4 is "XLXI_4_3"
    COMP8_MXILINX_comparator XLXI_5 (.A(a[23:16]),
                                     .B(b[23:16]),
                                     .EQ(XLXN_7));
    // synthesis attribute HU_SET of XLXI_5 is "XLXI_5_5"
    COMP8_MXILINX_comparator XLXI_6 (.A(a[15:8]),
                                     .B(b[15:8]),
                                     .EQ(XLXN_6));
```

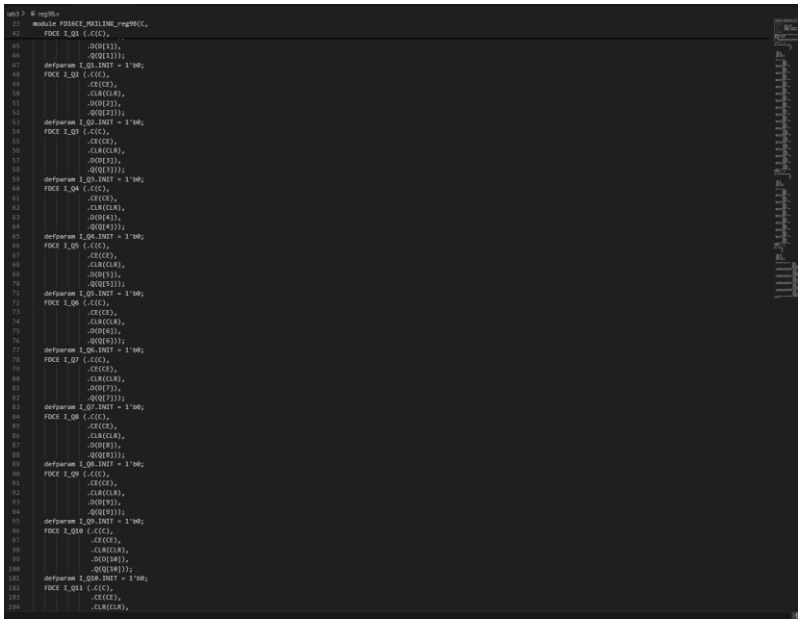


```
                                     .B(b[31:24]),
                                     .EQ(XLXN_4));
    // synthesis attribute HU_SET of XLXI_4 is "XLXI_4_3"
    COMP8_MXILINX_comparator XLXI_5 (.A(a[23:16]),
                                     .B(b[23:16]),
                                     .EQ(XLXN_7));
    // synthesis attribute HU_SET of XLXI_5 is "XLXI_5_5"
    COMP8_MXILINX_comparator XLXI_6 (.A(a[15:8]),
                                     .B(b[15:8]),
                                     .EQ(XLXN_6));
    // synthesis attribute HU_SET of XLXI_6 is "XLXI_6_6"
    COMP8_MXILINX_comparator XLXI_7 (.A(a[7:0]),
                                     .B(b[7:0]),
                                     .EQ(XLXN_5));
    // synthesis attribute HU_SET of XLXI_7 is "XLXI_7_7"
    OR2B1 XLXI_8 (.I0(amask[6]),
                  .I1(XLXN_1),
                  .O(XLXN_29));
    OR2B1 XLXI_9 (.I0(amask[5]),
                  .I1(XLXN_2),
                  .O(XLXN_30));
    OR2B1 XLXI_10 (.I0(amask[4]),
                   .I1(XLXN_3),
                   .O(XLXN_32));
    OR2B1 XLXI_11 (.I0(amask[3]),
                   .I1(XLXN_4),
                   .O(XLXN_33));
    OR2B1 XLXI_12 (.I0(amask[2]),
                   .I1(XLXN_7),
                   .O(XLXN_36));
    OR2B1 XLXI_13 (.I0(amask[1]),
                   .I1(XLXN_6),
                   .O(XLXN_35));
    OR2B1 XLXI_14 (.I0(amask[0]),
                   .I1(XLXN_5),
                   .O(XLXN_34));
    AND7_MXILINX_comparator XLXI_15 (.I0(XLXN_34),
                                     .I1(XLXN_35),
                                     .I2(XLXN_36),
                                     .I3(XLXN_33),
                                     .I4(XLXN_32),
                                     .I5(XLXN_30),
                                     .I6(XLXN_29),
                                     .O(match));
    // synthesis attribute HU_SET of XLXI_15 is "XLXI_15_4"
  endmodule
```

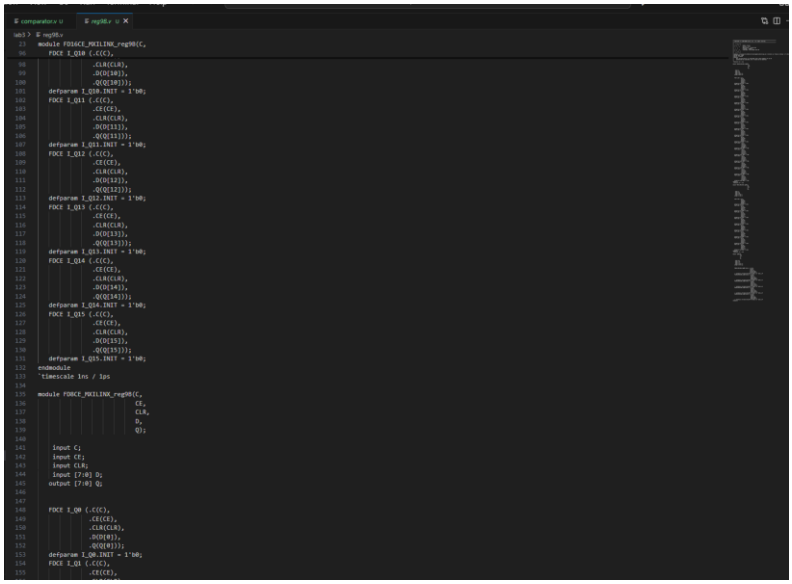
ii) reg9B.v



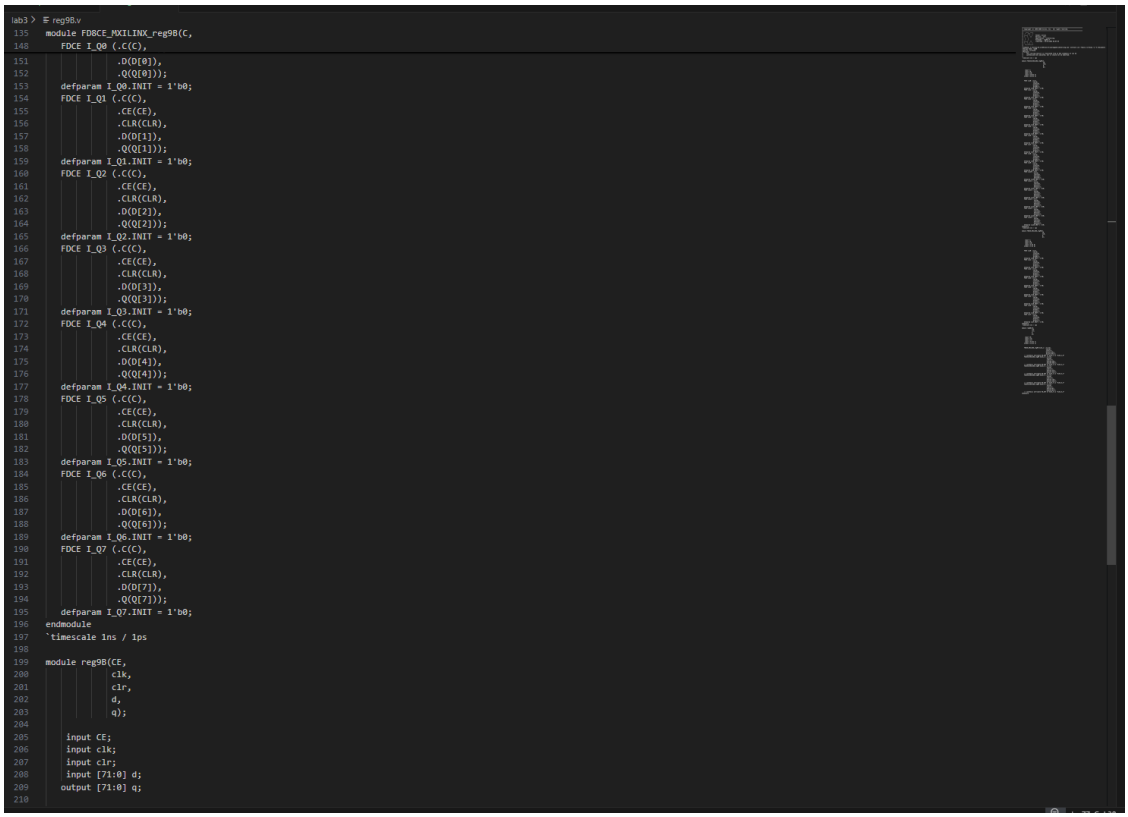
```
lab3 > reg98.v
1   //////////////////////////////////////////////////////////////////////////////////
2   // Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.
3   //////////////////////////////////////////////////////////////////////////////////
4   //   ____  ____
5   //  /   /\/   /
6   // /___/  \  /    Vendor: Xilinx
7   // \   \   \/     Version : 10.1
8   //  \   \         Application : sch2verilog
9   //  /   /         Filename : reg98.vf
10  // /___/   /\     Timestamp : 01/31/2026 02:07:45
11  // \   \  /  \
12  //  \___\/\___\
13  //
14  //Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family virtex2p -w "C:/Documents and Settings/student/lab3/reg98.sch" reg98.vf
15  //Design Name: reg98
16  //Device: virtex2p
17  //Purpose:
18  //    This verilog netlist is translated from an ECS schematic.It can be
19  //    synthesized and simulated, but it should not be modified.
20  //
21  `timescale 1ns / 1ps
22
23  module FD16CE_MXILINX_reg98(C,
24                              CE,
25                              CLR,
26                              D,
27                              Q);
28
29    input C;
30    input CE;
31    input CLR;
32    input [15:0] D;
33    output [15:0] Q;
34
35
36    FDCE I_Q0 (.C(C),
37               .CE(CE),
38               .CLR(CLR),
39               .D(D[0]),
40               .Q(Q[0]));
41    defparam I_Q0.INIT = 1'b0;
42    FDCE I_Q1 (.C(C),
43               .CE(CE),
44               .CLR(CLR),
45               .D(D[1]),
46               .Q(Q[1]));
47    defparam I_Q1.INIT = 1'b0;
48    FDCE I_Q2 (.C(C),
49               .CE(CE),
50               .CLR(CLR),
51               .D(D[2]),
52               .Q(Q[2]));
53    defparam I_Q2.INIT = 1'b0;
54    FDCE I_Q3 (.C(C),
55               .CE(CE),
56               .CLR(CLR),
57               .D(D[3]),
58               .Q(Q[3]));
59    defparam I_Q3.INIT = 1'b0;
60    FDCE I_Q4 (.C(C),
61               .CE(CE),
62               .CLR(CLR),
63               .D(D[4]),
64               .Q(Q[4]));
65    defparam I_Q4.INIT = 1'b0;
66    FDCE I_Q5 (.C(C),
67               .CE(CE),
68               .CLR(CLR),
69               .D(D[5]),
70               .Q(Q[5]));
71    defparam I_Q5.INIT = 1'b0;
72    FDCE I_Q6 (.C(C),
73               .CE(CE),
74               .CLR(CLR),
75               .D(D[6]),
76               .Q(Q[6]));
77    defparam I_Q6.INIT = 1'b0;
78    FDCE I_Q7 (.C(C),
79               .CE(CE),
80               .CLR(CLR),
81               .D(D[7]),
82               .Q(Q[7]));
83    defparam I_Q7.INIT = 1'b0;
84    FDCE I_Q8 (.C(C),
85               .CE(CE),
86               .CLR(CLR),
87               .D(D[8]),
88               .Q(Q[8]));
89    defparam I_Q8.INIT = 1'b0;
90    FDCE I_Q9 (.C(C),
91               .CE(CE),
92               .CLR(CLR),
93               .D(D[9]),
94               .Q(Q[9]));
95    defparam I_Q9.INIT = 1'b0;
96    FDCE I_Q10 (.C(C),
97                .CE(CE),
98                .CLR(CLR),
99                .D(D[10]),
100               .Q(Q[10]));
101   defparam I_Q10.INIT = 1'b0;
102   FDCE I_Q11 (.C(C),
103               .CE(CE),
104               .CLR(CLR),
```



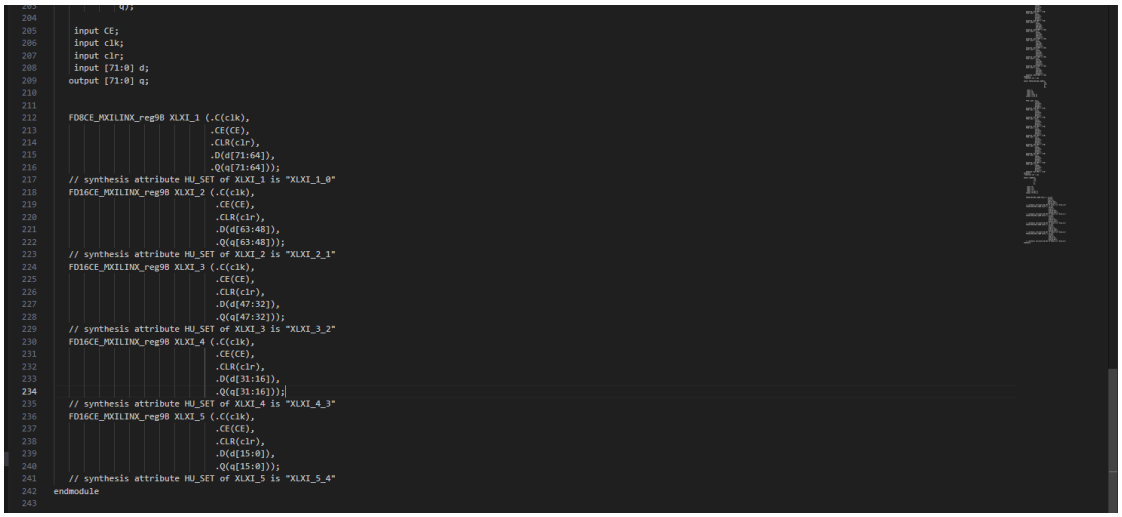
```
lab3 > reg98.v
23  module FD16CE_MXILINX_reg98(C,
42    FDCE I_Q1 (.C(C),
45               .D(D[1]),
46               .Q(Q[1]));
47    defparam I_Q1.INIT = 1'b0;
48    FDCE I_Q2 (.C(C),
49               .CE(CE),
50               .CLR(CLR),
51               .D(D[2]),
52               .Q(Q[2]));
53    defparam I_Q2.INIT = 1'b0;
54    FDCE I_Q3 (.C(C),
55               .CE(CE),
56               .CLR(CLR),
57               .D(D[3]),
58               .Q(Q[3]));
59    defparam I_Q3.INIT = 1'b0;
60    FDCE I_Q4 (.C(C),
61               .CE(CE),
62               .CLR(CLR),
63               .D(D[4]),
64               .Q(Q[4]));
65    defparam I_Q4.INIT = 1'b0;
66    FDCE I_Q5 (.C(C),
67               .CE(CE),
68               .CLR(CLR),
69               .D(D[5]),
70               .Q(Q[5]));
71    defparam I_Q5.INIT = 1'b0;
72    FDCE I_Q6 (.C(C),
73               .CE(CE),
74               .CLR(CLR),
75               .D(D[6]),
76               .Q(Q[6]));
77    defparam I_Q6.INIT = 1'b0;
78    FDCE I_Q7 (.C(C),
79               .CE(CE),
80               .CLR(CLR),
81               .D(D[7]),
82               .Q(Q[7]));
83    defparam I_Q7.INIT = 1'b0;
84    FDCE I_Q8 (.C(C),
85               .CE(CE),
86               .CLR(CLR),
87               .D(D[8]),
88               .Q(Q[8]));
89    defparam I_Q8.INIT = 1'b0;
90    FDCE I_Q9 (.C(C),
91               .CE(CE),
92               .CLR(CLR),
93               .D(D[9]),
94               .Q(Q[9]));
95    defparam I_Q9.INIT = 1'b0;
96    FDCE I_Q10 (.C(C),
97                .CE(CE),
98                .CLR(CLR),
99                .D(D[10]),
100               .Q(Q[10]));
101   defparam I_Q10.INIT = 1'b0;
102   FDCE I_Q11 (.C(C),
103               .CE(CE),
104               .CLR(CLR),
```



```
23  module FD16CE_MXILINX_reg98(C,
96    FDCE I_Q10 (.C(C),
98                .CLR(CLR),
99                .D(D[10]),
100               .Q(Q[10]));
101   defparam I_Q10.INIT = 1'b0;
102   FDCE I_Q11 (.C(C),
103               .CE(CE),
104               .CLR(CLR),
105               .D(D[11]),
106               .Q(Q[11]));
107   defparam I_Q11.INIT = 1'b0;
108   FDCE I_Q12 (.C(C),
109               .CE(CE),
110               .CLR(CLR),
111               .D(D[12]),
112               .Q(Q[12]));
113   defparam I_Q12.INIT = 1'b0;
114   FDCE I_Q13 (.C(C),
115               .CE(CE),
116               .CLR(CLR),
117               .D(D[13]),
118               .Q(Q[13]));
119   defparam I_Q13.INIT = 1'b0;
120   FDCE I_Q14 (.C(C),
121               .CE(CE),
122               .CLR(CLR),
123               .D(D[14]),
124               .Q(Q[14]));
125   defparam I_Q14.INIT = 1'b0;
126   FDCE I_Q15 (.C(C),
127               .CE(CE),
128               .CLR(CLR),
129               .D(D[15]),
130               .Q(Q[15]));
131   defparam I_Q15.INIT = 1'b0;
132 endmodule
133 `timescale 1ns / 1ps
134
135 module FD8CE_MXILINX_reg98(C,
136                            CE,
137                            CLR,
138                            D,
139                            Q);
140
141   input C;
142   input CE;
143   input CLR;
144   input [7:0] D;
145   output [7:0] Q;
146
147
148   FDCE I_Q0 (.C(C),
149              .CE(CE),
150              .CLR(CLR),
151              .D(D[0]),
152              .Q(Q[0]));
153   defparam I_Q0.INIT = 1'b0;
154   FDCE I_Q1 (.C(C),
155              .CE(CE),
```

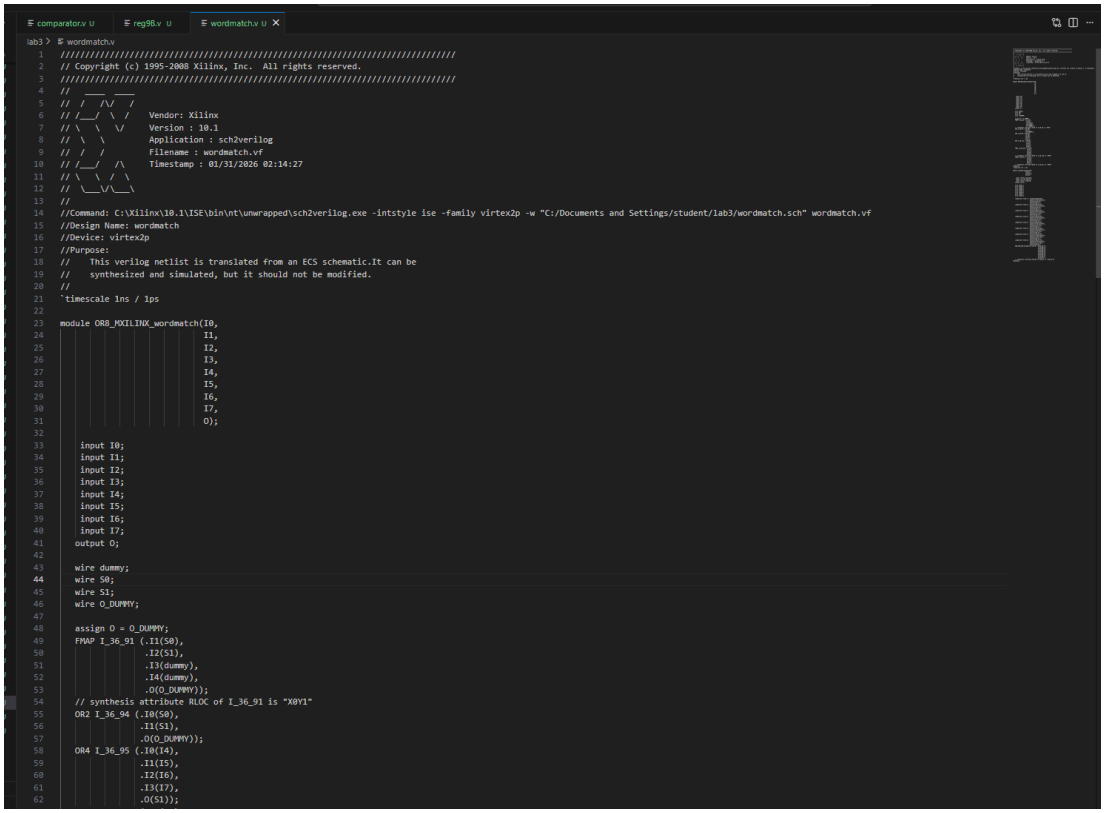


```
lab3 > ≡ reg98.v
135   module FD8CE_MXILINX_reg98(C,
148     FDCE I_Q0 (.C(C),
151               .D(D[0]),
152               .Q(Q[0]));
153     defparam I_Q0.INIT = 1'b0;
154     FDCE I_Q1 (.C(C),
155               .CE(CE),
156               .CLR(CLR),
157               .D(D[1]),
158               .Q(Q[1]));
159     defparam I_Q1.INIT = 1'b0;
160     FDCE I_Q2 (.C(C),
161               .CE(CE),
162               .CLR(CLR),
163               .D(D[2]),
164               .Q(Q[2]));
165     defparam I_Q2.INIT = 1'b0;
166     FDCE I_Q3 (.C(C),
167               .CE(CE),
168               .CLR(CLR),
169               .D(D[3]),
170               .Q(Q[3]));
171     defparam I_Q3.INIT = 1'b0;
172     FDCE I_Q4 (.C(C),
173               .CE(CE),
174               .CLR(CLR),
175               .D(D[4]),
176               .Q(Q[4]));
177     defparam I_Q4.INIT = 1'b0;
178     FDCE I_Q5 (.C(C),
179               .CE(CE),
180               .CLR(CLR),
181               .D(D[5]),
182               .Q(Q[5]));
183     defparam I_Q5.INIT = 1'b0;
184     FDCE I_Q6 (.C(C),
185               .CE(CE),
186               .CLR(CLR),
187               .D(D[6]),
188               .Q(Q[6]));
189     defparam I_Q6.INIT = 1'b0;
190     FDCE I_Q7 (.C(C),
191               .CE(CE),
192               .CLR(CLR),
193               .D(D[7]),
194               .Q(Q[7]));
195     defparam I_Q7.INIT = 1'b0;
196   endmodule
197   `timescale 1ns / 1ps
198
199   module reg98(CE,
200              clk,
201              clr,
202              d,
203              q);
204
205     input CE;
206     input clk;
207     input clr;
208     input [71:0] d;
209     output [71:0] q;
210
```

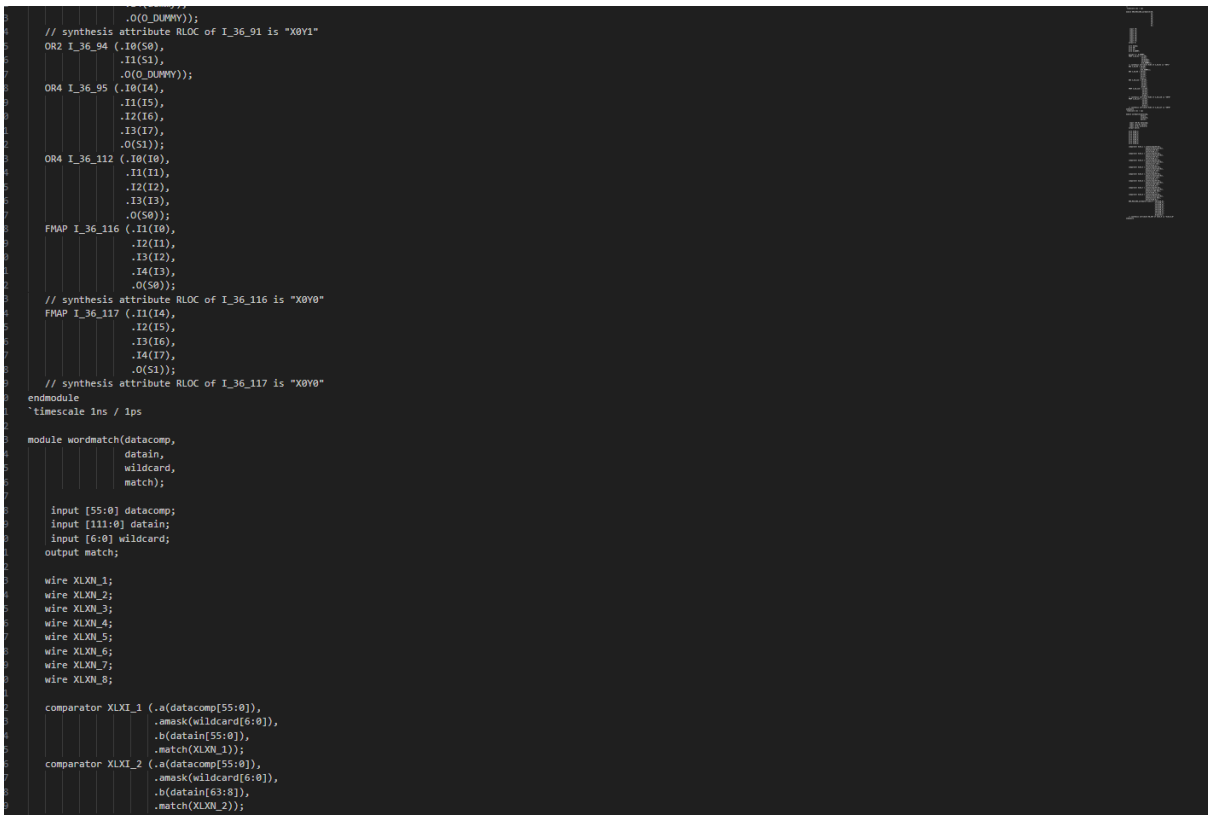


```
204
205     input CE;
206     input clk;
207     input clr;
208     input [71:0] d;
209     output [71:0] q;
210
211
212     FD8CE_MXILINX_reg98 XLXI_1 (.C(clk),
213                                 .CE(CE),
214                                 .CLR(clr),
215                                 .D(d[71:64]),
216                                 .Q(q[71:64]));
217     // synthesis attribute HU_SET of XLXI_1 is "XLXI_1_0"
218     FD16CE_MXILINX_reg98 XLXI_2 (.C(clk),
219                                  .CE(CE),
220                                  .CLR(clr),
221                                  .D(d[63:48]),
222                                  .Q(q[63:48]));
223     // synthesis attribute HU_SET of XLXI_2 is "XLXI_2_1"
224     FD16CE_MXILINX_reg98 XLXI_3 (.C(clk),
225                                  .CE(CE),
226                                  .CLR(clr),
227                                  .D(d[47:32]),
228                                  .Q(q[47:32]));
229     // synthesis attribute HU_SET of XLXI_3 is "XLXI_3_2"
230     FD16CE_MXILINX_reg98 XLXI_4 (.C(clk),
231                                  .CE(CE),
232                                  .CLR(clr),
233                                  .D(d[31:16]),
234                                  .Q(q[31:16]));
235     // synthesis attribute HU_SET of XLXI_4 is "XLXI_4_3"
236     FD16CE_MXILINX_reg98 XLXI_5 (.C(clk),
237                                  .CE(CE),
238                                  .CLR(clr),
239                                  .D(d[15:0]),
240                                  .Q(q[15:0]));
241     // synthesis attribute HU_SET of XLXI_5 is "XLXI_5_4"
242   endmodule
243
```

iii) wordmatch.v



```
comparator.v U   reg98.v U   wordmatch.v U
lab3 > wordmatch.v
//////////////////////////////////////////////////////////////////////////////////
// Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.
//////////////////////////////////////////////////////////////////////////////////
//   ____  ____
//  /   /\/   /
// /___/  \  /    Vendor: Xilinx
// \   \   \/     Version : 10.1
//  \   \         Application : sch2verilog
//  /   /         Filename : wordmatch.vf
// /___/   /\     Timestamp : 01/31/2026 02:14:27
// \   \  /  \
//  \___\/\___\
//
//Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family virtex2p -w "C:/Documents and Settings/student/lab3/wordmatch.sch" wordmatch.vf
//Design Name: wordmatch
//Device: virtex2p
//Purpose:
//    This verilog netlist is translated from an ECS schematic.It can be
//    synthesized and simulated, but it should not be modified.
//
`timescale 1ns / 1ps

module OR8_MXILINX_wordmatch(I0,
                             I1,
                             I2,
                             I3,
                             I4,
                             I5,
                             I6,
                             I7,
                             O);

    input I0;
    input I1;
    input I2;
    input I3;
    input I4;
    input I5;
    input I6;
    input I7;
    output O;

    wire dummy;
    wire S0;
    wire S1;
    wire O_DUMMY;

    assign O = O_DUMMY;
    FMAP I_36_91 (.I1(S0),
                  .I2(S1),
                  .I3(dummy),
                  .I4(dummy),
                  .O(O_DUMMY));
    // synthesis attribute RLOC of I_36_91 is "X0Y1"
    OR2 I_36_94 (.I0(S0),
                 .I1(S1),
                 .O(O_DUMMY));
    OR4 I_36_95 (.I0(I4),
                 .I1(I5),
                 .I2(I6),
                 .I3(I7),
                 .O(S1));
```



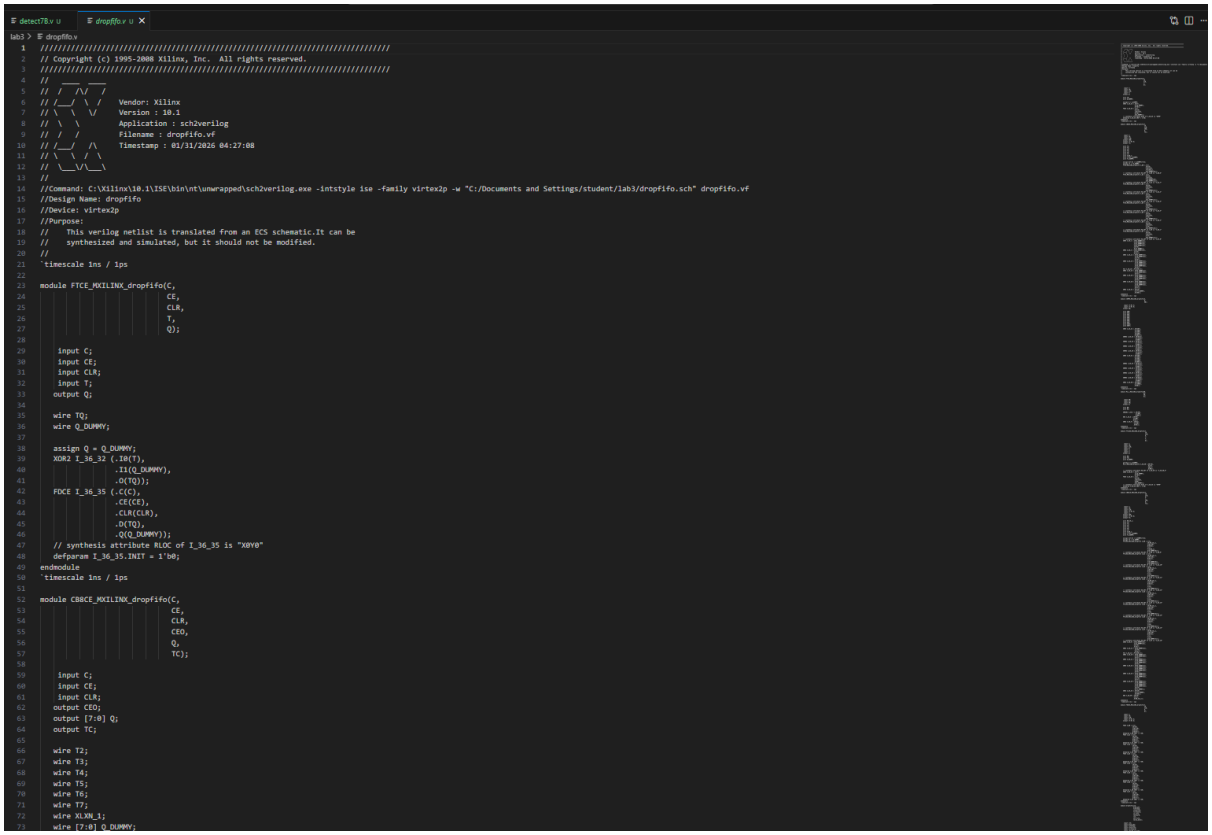
```
                 .O(O_DUMMY));
    // synthesis attribute RLOC of I_36_91 is "X0Y1"
    OR2 I_36_94 (.I0(S0),
                 .I1(S1),
                 .O(O_DUMMY));
    OR4 I_36_95 (.I0(I4),
                 .I1(I5),
                 .I2(I6),
                 .I3(I7),
                 .O(S1));
    OR4 I_36_112 (.I0(I0),
                  .I1(I1),
                  .I2(I2),
                  .I3(I3),
                  .O(S0));
    FMAP I_36_116 (.I1(I0),
                   .I2(I1),
                   .I3(I2),
                   .I4(I3),
                   .O(S0));
    // synthesis attribute RLOC of I_36_116 is "X0Y0"
    FMAP I_36_117 (.I1(I4),
                   .I2(I5),
                   .I3(I6),
                   .I4(I7),
                   .O(S1));
    // synthesis attribute RLOC of I_36_117 is "X0Y0"
endmodule
`timescale 1ns / 1ps

module wordmatch(datacomp,
                 datain,
                 wildcard,
                 match);

    input [55:0] datacomp;
    input [111:0] datain;
    input [6:0] wildcard;
    output match;

    wire XLXN_1;
    wire XLXN_2;
    wire XLXN_3;
    wire XLXN_4;
    wire XLXN_5;
    wire XLXN_6;
    wire XLXN_7;
    wire XLXN_8;

    comparator XLXI_1 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[55:0]),
                       .match(XLXN_1));
    comparator XLXI_2 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[63:8]),
                       .match(XLXN_2));
```

```
                    .match(XLXN_2));
    comparator XLXI_3 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[71:16]),
                       .match(XLXN_3));
    comparator XLXI_4 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[79:24]),
                       .match(XLXN_4));
    comparator XLXI_5 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[87:32]),
                       .match(XLXN_5));
    comparator XLXI_6 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[95:40]),
                       .match(XLXN_6));
    comparator XLXI_7 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[103:48]),
                       .match(XLXN_7));
    comparator XLXI_8 (.a(datacomp[55:0]),
                       .amask(wildcard[6:0]),
                       .b(datain[111:56]),
                       .match(XLXN_8));
    OR8_MXILINX_wordmatch XLXI_9 (.I0(XLXN_8),
                                  .I1(XLXN_7),
                                  .I2(XLXN_6),
                                  .I3(XLXN_5),
                                  .I4(XLXN_4),
                                  .I5(XLXN_3),
                                  .I6(XLXN_2),
                                  .I7(XLXN_1),
                                  .O(match));
    // synthesis attribute HU_SET of XLXI_9 is "XLXI_9_0"
endmodule

```

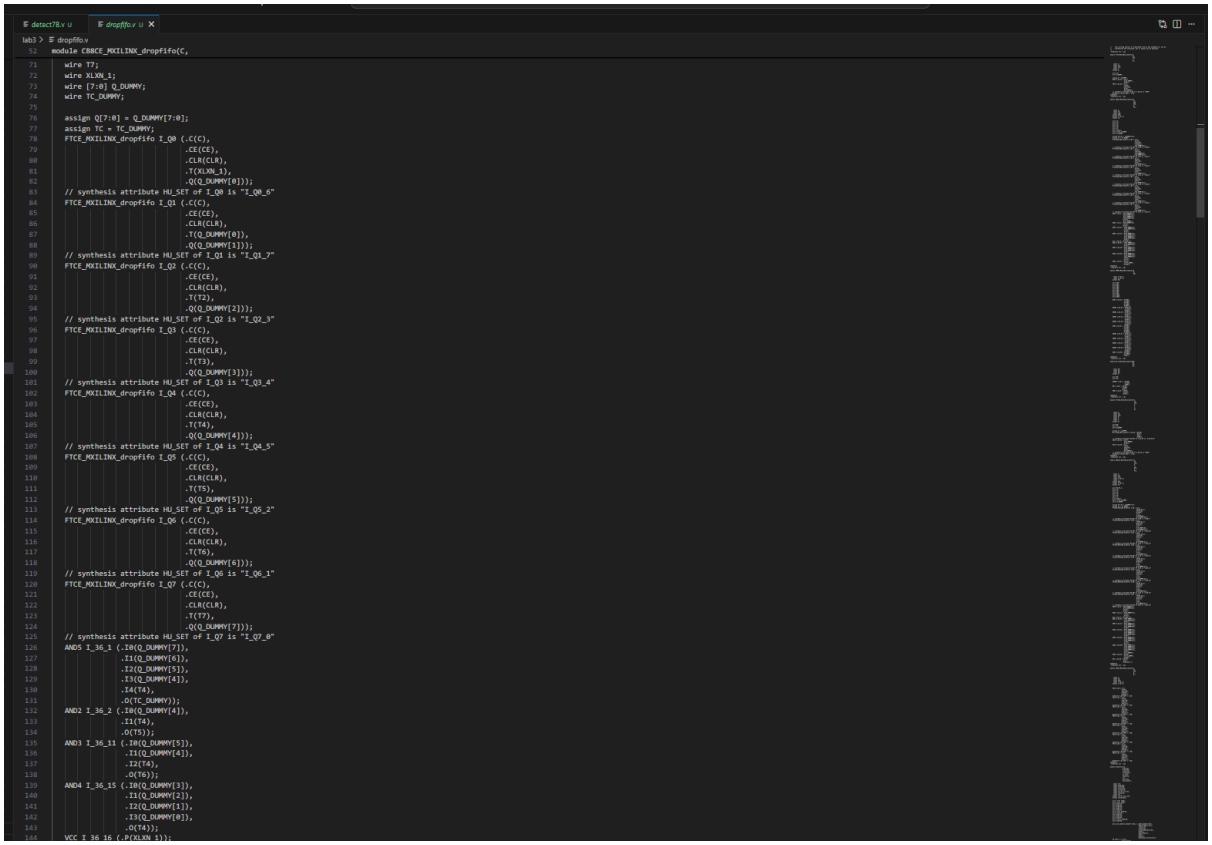
iv) detect7B.v

```
detect7B.v U
lab3 > detect7B.v
  //Device: virtex2p
  //Purpose:
  //    This verilog netlist is translated from an ECS schematic.It can be
  //    synthesized and simulated, but it should not be modified.
  //
  `timescale 1ns / 1ps
  `include "busmerge.v"
  `include "wordmatch.v"



  module detect7B(ce,
                  clk,
                  hwregA,
                  match_en,
                  mrst,
                  pipe1,
                  match);

    input ce;
    input clk;
    input [63:0] hwregA;
    input match_en;
    input mrst;
    input [71:0] pipe1;
    output match;

    wire clr;
    wire [71:0] pipe0;
    wire [111:0] XLXN_1;
    wire XLXN_18;
    wire XLXN_23;
    wire match_DUMMY;

    assign match = match_DUMMY;
    busmerge XLXI_1 (.da(pipe0[47:0]),
                     .db(pipe1[63:0]),
                     .q(XLXN_1[111:0]));
    wordmatch XLXI_2 (.datacomp(hwregA[55:0]),
                      .datain(XLXN_1[111:0]),
                      .wildcard(hwregA[62:56]),
                      .match(XLXN_18));
    FD XLXI_4 (.C(clk),
               .D(mrst),
               .Q(clr));
    defparam XLXI_4.INIT = 1'b0;
    FDCE XLXI_5 (.C(clk),
                 .CE(XLXN_23),
                 .CLR(clr),
                 .D(XLXN_23),
                 .Q(match_DUMMY));
    defparam XLXI_5.INIT = 1'b0;
    AND3B1 XLXI_6 (.I0(match_DUMMY),
                   .I1(match_en),
                   .I2(XLXN_18),
                   .O(XLXN_23));
    reg9B XLXI_7 (.CE(ce),
                  .clk(clk),
                  .clr(clr),
                  .d(pipe1[71:0]),
                  .q(pipe0[71:0]));
  endmodule
```

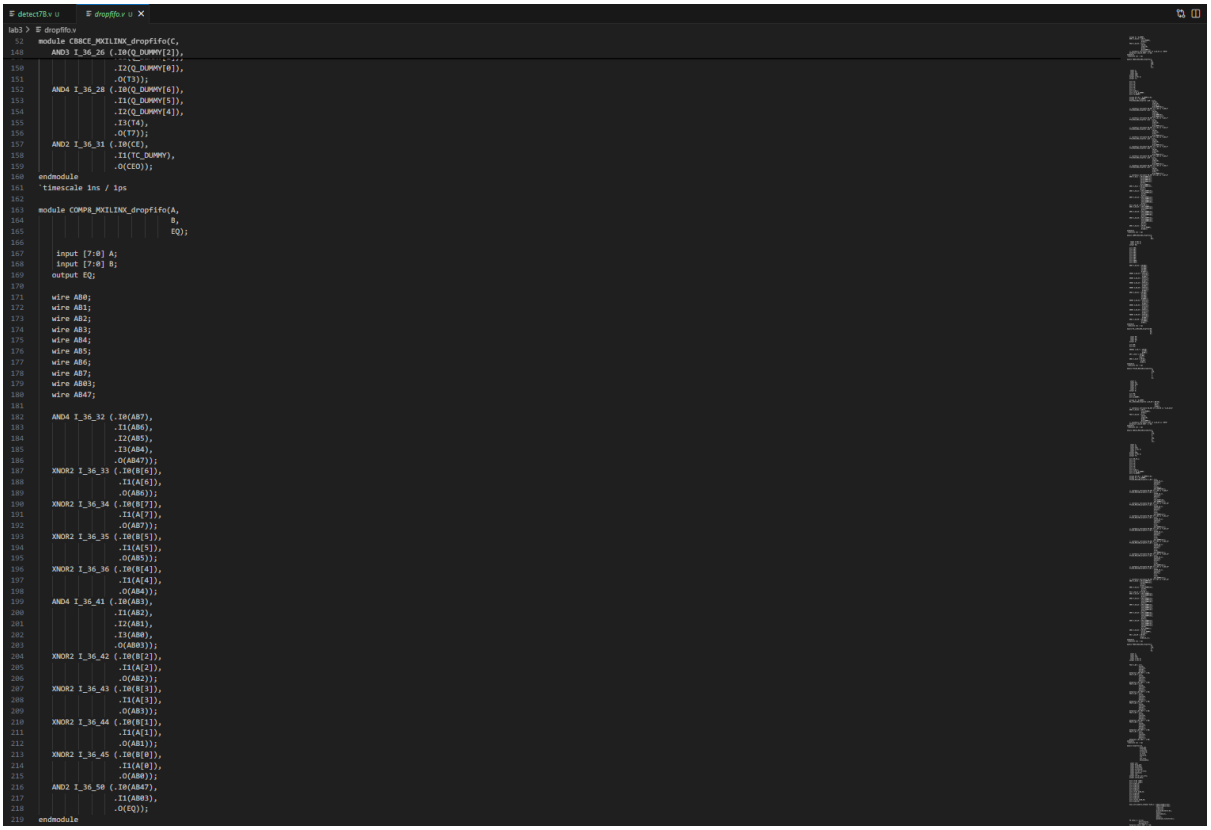
v) dropfifo.v



```
detect78.v U    dropfifo.v U
lab3 > dropfifo.v
////////////////////////////////////////////////////////////////////////////////
// Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.
////////////////////////////////////////////////////////////////////////////////
//   ____  ____
//  /   /\/   /
// /___/  \  /    Vendor: Xilinx
// \   \   \/     Version : 10.1
//  \   \         Application : sch2verilog
//  /   /         Filename : dropfifo.vf
// /___/   /\     Timestamp : 01/31/2026 04:27:08
// \   \  /  \
//  \___\/\___\
//
//Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family virtex2p -w "C:/Documents and Settings/student/lab3/dropfifo.sch" dropfifo.vf
//Design Name: dropfifo
//Device: virtex2p
//Purpose:
//    This verilog netlist is translated from an ECS schematic.It can be
//    synthesized and simulated, but it should not be modified.
//
`timescale 1ns / 1ps

module FTCE_MXILINX_dropfifo(C,
                             CE,
                             CLR,
                             T,
                             Q);

    input C;
    input CE;
    input CLR;
    input T;
    output Q;

    wire TQ;
    wire Q_DUMMY;

    assign Q = Q_DUMMY;
    XOR2 I_36_32 (.I0(T),
                  .I1(Q_DUMMY),
                  .O(TQ));
    FDCE I_36_35 (.C(C),
                  .CE(CE),
                  .CLR(CLR),
                  .D(TQ),
                  .Q(Q_DUMMY));
    // synthesis attribute RLOC of I_36_35 is "X0Y0"
    defparam I_36_35.INIT = 1'b0;
endmodule
`timescale 1ns / 1ps

module CB8CE_MXILINX_dropfifo(C,
                              CE,
                              CLR,
                              CEO,
                              Q,
                              TC);

    input C;
    input CE;
    input CLR;
    output CEO;
    output [7:0] Q;
    output TC;

    wire T2;
    wire T3;
    wire T4;
    wire T5;
    wire T6;
    wire T7;
    wire XLXN_1;
    wire [7:0] Q_DUMMY;
```

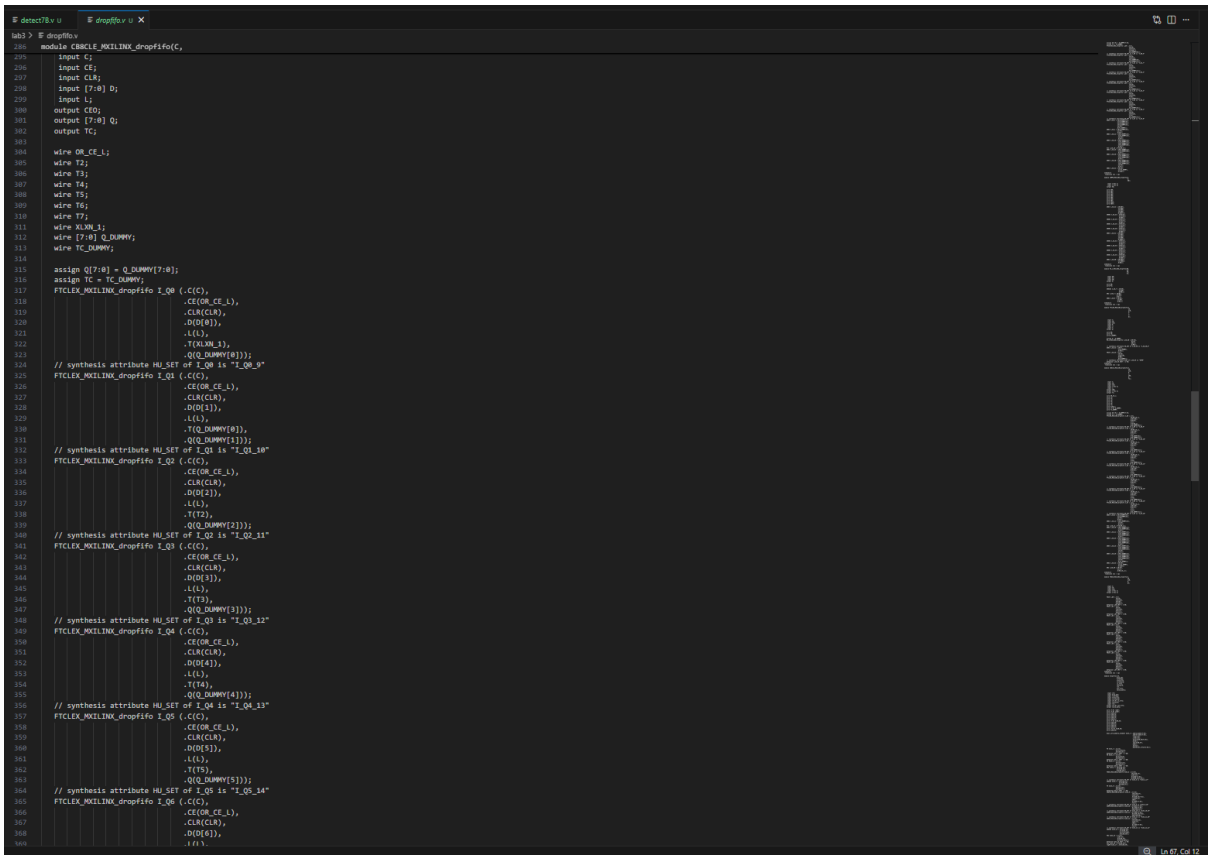


```
detect78.v U    dropfifo.v U
lab3 > dropfifo.v
module CB8CE_MXILINX_dropfifo(C,
    wire T7;
    wire XLXN_1;
    wire [7:0] Q_DUMMY;
    wire TC_DUMMY;

    assign Q[7:0] = Q_DUMMY[7:0];
    assign TC = TC_DUMMY;
    FTCE_MXILINX_dropfifo I_Q0 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(XLXN_1),
                                .Q(Q_DUMMY[0]));
    // synthesis attribute HU_SET of I_Q0 is "I_Q0_6"
    FTCE_MXILINX_dropfifo I_Q1 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(Q_DUMMY[0]),
                                .Q(Q_DUMMY[1]));
    // synthesis attribute HU_SET of I_Q1 is "I_Q1_7"
    FTCE_MXILINX_dropfifo I_Q2 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(T2),
                                .Q(Q_DUMMY[2]));
    // synthesis attribute HU_SET of I_Q2 is "I_Q2_3"
    FTCE_MXILINX_dropfifo I_Q3 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(T3),
                                .Q(Q_DUMMY[3]));
    // synthesis attribute HU_SET of I_Q3 is "I_Q3_4"
    FTCE_MXILINX_dropfifo I_Q4 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(T4),
                                .Q(Q_DUMMY[4]));
    // synthesis attribute HU_SET of I_Q4 is "I_Q4_5"
    FTCE_MXILINX_dropfifo I_Q5 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(T5),
                                .Q(Q_DUMMY[5]));
    // synthesis attribute HU_SET of I_Q5 is "I_Q5_2"
    FTCE_MXILINX_dropfifo I_Q6 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(T6),
                                .Q(Q_DUMMY[6]));
    // synthesis attribute HU_SET of I_Q6 is "I_Q6_1"
    FTCE_MXILINX_dropfifo I_Q7 (.C(C),
                                .CE(CE),
                                .CLR(CLR),
                                .T(T7),
                                .Q(Q_DUMMY[7]));
    // synthesis attribute HU_SET of I_Q7 is "I_Q7_0"
    AND5 I_36_1 (.I0(Q_DUMMY[7]),
                 .I1(Q_DUMMY[6]),
                 .I2(Q_DUMMY[5]),
                 .I3(Q_DUMMY[4]),
                 .I4(T4),
                 .O(TC_DUMMY));
    AND2 I_36_2 (.I0(Q_DUMMY[4]),
                 .I1(T4),
                 .O(T5));
    AND3 I_36_11 (.I0(Q_DUMMY[5]),
                  .I1(Q_DUMMY[4]),
                  .I2(T4),
                  .O(T6));
    AND4 I_36_15 (.I0(Q_DUMMY[3]),
                  .I1(Q_DUMMY[2]),
                  .I2(Q_DUMMY[1]),
                  .I3(Q_DUMMY[0]),
                  .O(T4));
    VCC I_36_16 (.P(XLXN_1));
```



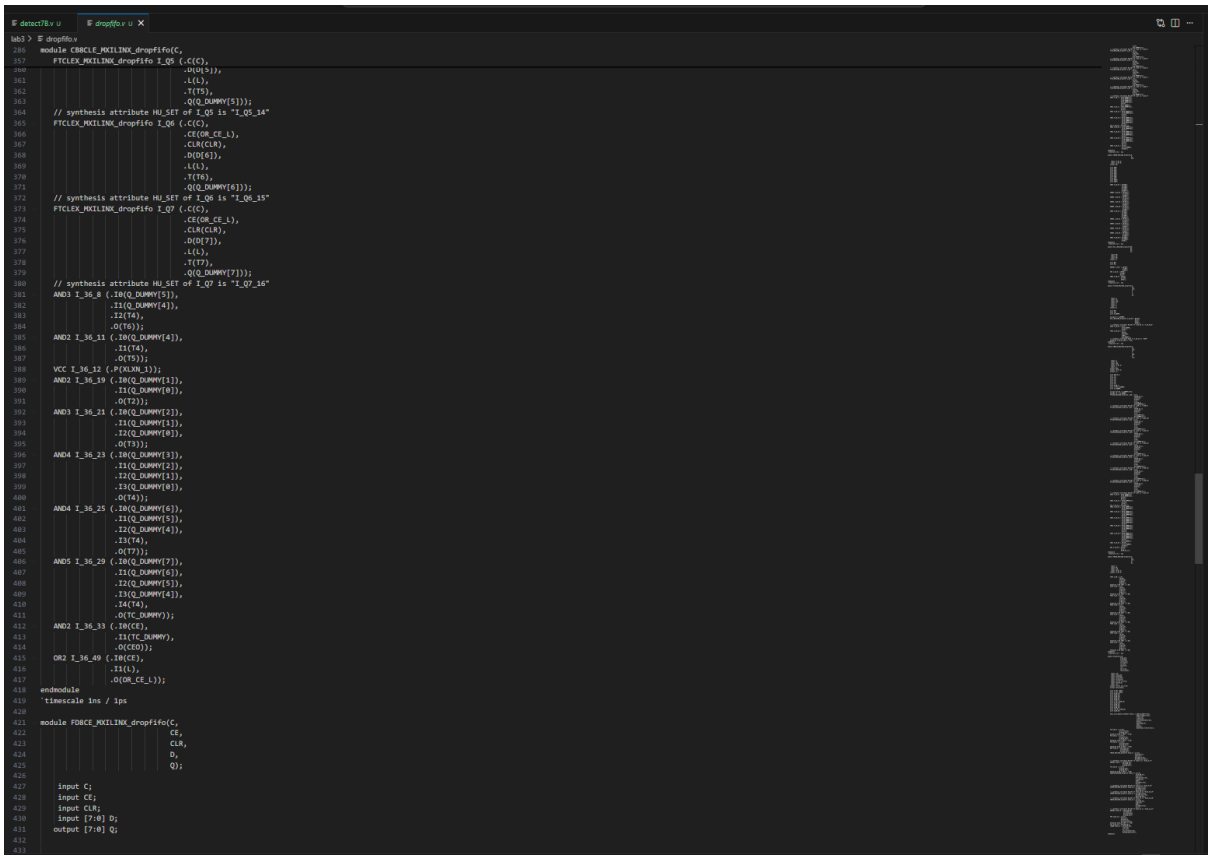
```
detect78.v U    dropfifo.v U
lab3 > dropfifo.v
  module CB8CE_MXILINX_dropfifo(C,
     AND3 I_36_26 (.I0(Q_DUMMY[2]),
                   .I1(Q_DUMMY[1]),
                   .I2(Q_DUMMY[0]),
                   .O(T3));
     AND4 I_36_28 (.I0(Q_DUMMY[6]),
                   .I1(Q_DUMMY[5]),
                   .I2(Q_DUMMY[4]),
                   .I3(T4),
                   .O(T7));
     AND2 I_36_31 (.I0(CE),
                   .I1(TC_DUMMY),
                   .O(CEO));
  endmodule
  `timescale 1ns / 1ps

  module COMP8_MXILINX_dropfifo(A,
                                B,
                                EQ);

    input [7:0] A;
    input [7:0] B;
    output EQ;

    wire AB0;
    wire AB1;
    wire AB2;
    wire AB3;
    wire AB4;
    wire AB5;
    wire AB6;
    wire AB7;
    wire AB03;
    wire AB47;

    AND4 I_36_32 (.I0(AB7),
                  .I1(AB6),
                  .I2(AB5),
                  .I3(AB4),
                  .O(AB47));
    XNOR2 I_36_33 (.I0(B[6]),
                   .I1(A[6]),
                   .O(AB6));
    XNOR2 I_36_34 (.I0(B[7]),
                   .I1(A[7]),
                   .O(AB7));
    XNOR2 I_36_35 (.I0(B[5]),
                   .I1(A[5]),
                   .O(AB5));
    XNOR2 I_36_36 (.I0(B[4]),
                   .I1(A[4]),
                   .O(AB4));
    AND4 I_36_41 (.I0(AB3),
                  .I1(AB2),
                  .I2(AB1),
                  .I3(AB0),
                  .O(AB03));
    XNOR2 I_36_42 (.I0(B[2]),
                   .I1(A[2]),
                   .O(AB2));
    XNOR2 I_36_43 (.I0(B[3]),
                   .I1(A[3]),
                   .O(AB3));
    XNOR2 I_36_44 (.I0(B[1]),
                   .I1(A[1]),
                   .O(AB1));
    XNOR2 I_36_45 (.I0(B[0]),
                   .I1(A[0]),
                   .O(AB0));
    AND2 I_36_50 (.I0(AB47),
                  .I1(AB03),
                  .O(EQ));
  endmodule
```

```
detect78.v U    dropfifo.v U
lab3 > dropfifo.v
  module COMP8_MXILINX_dropfifo(A,
     AND2 I_36_50 (.I0(AB47),
                   .I1(EQ));
  endmodule
  `timescale 1ns / 1ps

  module M2_1_MXILINX_dropfifo(D0,
                               D1,
                               S0,
                               O);

    input D0;
    input D1;
    input S0;
    output O;

    wire M0;
    wire M1;

    AND2B1 I_36_7 (.I0(S0),
                   .I1(D0),
                   .O(M0));
    OR2 I_36_8 (.I0(M1),
                .I1(M0),
                .O(O));
    AND2 I_36_9 (.I0(D1),
                 .I1(S0),
                 .O(M1));
  endmodule
  `timescale 1ns / 1ps

  module FTCLEX_MXILINX_dropfifo(C,
                                 CE,
                                 CLR,
                                 D,
                                 L,
                                 T,
                                 Q);

    input C;
    input CE;
    input CLR;
    input D;
    input L;
    input T;
    output Q;

    wire MD;
    wire TQ;
    wire Q_DUMMY;

    assign Q = Q_DUMMY;
    M2_1_MXILINX_dropfifo I_36_30 (.D0(TQ),
                                   .D1(D),
                                   .S0(L),
                                   .O(MD));
    // synthesis attribute HU_SET of I_36_30 is "I_36_30_8"
    XOR2 I_36_32 (.I0(T),
                  .I1(Q_DUMMY),
                  .O(TQ));
    FDCE I_36_35 (.C(C),
                  .CE(CE),
                  .CLR(CLR),
                  .D(MD),
                  .Q(Q_DUMMY));
    // synthesis attribute RLOC of I_36_35 is "X0Y0"
    defparam I_36_35.INIT = 1'b0;
  endmodule
  `timescale 1ns / 1ps

  module CB8CLE_MXILINX_dropfifo(C,
                                 CE,
                                 CLR,
                                 D,
```

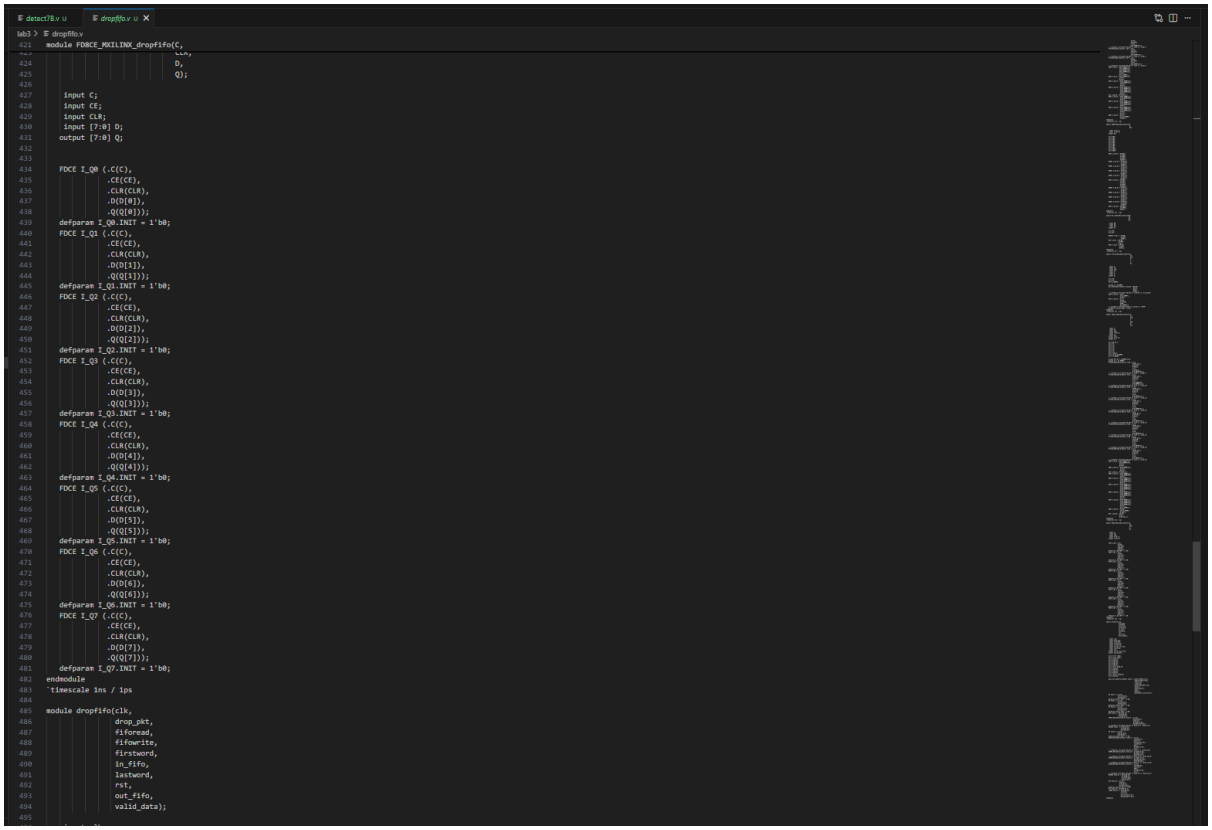


```
detect78.v U    dropfifo.v U
lab3 > dropfifo.v
286  module CBBCLE_MXILINX_dropfifo(C,
295      input C;
296      input CE;
297      input CLR;
298      input [7:0] D;
299      input L;
300      output CEO;
301      output [7:0] Q;
302      output TC;
303
304      wire OR_CE_L;
305      wire T2;
306      wire T3;
307      wire T4;
308      wire T5;
309      wire T6;
310      wire T7;
311      wire XLXN_1;
312      wire [7:0] Q_DUMMY;
313      wire TC_DUMMY;
314
315      assign Q[7:0] = Q_DUMMY[7:0];
316      assign TC = TC_DUMMY;
317      FTCLEX_MXILINX_dropfifo I_Q0 (.C(C),
318                                    .CE(OR_CE_L),
319                                    .CLR(CLR),
320                                    .D(D[0]),
321                                    .L(L),
322                                    .T(XLXN_1),
323                                    .Q(Q_DUMMY[0]));
324      // synthesis attribute HU_SET of I_Q0 is "I_Q0_9"
325      FTCLEX_MXILINX_dropfifo I_Q1 (.C(C),
326                                    .CE(OR_CE_L),
327                                    .CLR(CLR),
328                                    .D(D[1]),
329                                    .L(L),
330                                    .T(Q_DUMMY[0]),
331                                    .Q(Q_DUMMY[1]));
332      // synthesis attribute HU_SET of I_Q1 is "I_Q1_10"
333      FTCLEX_MXILINX_dropfifo I_Q2 (.C(C),
334                                    .CE(OR_CE_L),
335                                    .CLR(CLR),
336                                    .D(D[2]),
337                                    .L(L),
338                                    .T(T2),
339                                    .Q(Q_DUMMY[2]));
340      // synthesis attribute HU_SET of I_Q2 is "I_Q2_11"
341      FTCLEX_MXILINX_dropfifo I_Q3 (.C(C),
342                                    .CE(OR_CE_L),
343                                    .CLR(CLR),
344                                    .D(D[3]),
345                                    .L(L),
346                                    .T(T3),
347                                    .Q(Q_DUMMY[3]));
348      // synthesis attribute HU_SET of I_Q3 is "I_Q3_12"
349      FTCLEX_MXILINX_dropfifo I_Q4 (.C(C),
350                                    .CE(OR_CE_L),
351                                    .CLR(CLR),
352                                    .D(D[4]),
353                                    .L(L),
354                                    .T(T4),
355                                    .Q(Q_DUMMY[4]));
356      // synthesis attribute HU_SET of I_Q4 is "I_Q4_13"
357      FTCLEX_MXILINX_dropfifo I_Q5 (.C(C),
358                                    .CE(OR_CE_L),
359                                    .CLR(CLR),
360                                    .D(D[5]),
361                                    .L(L),
362                                    .T(T5),
363                                    .Q(Q_DUMMY[5]));
364      // synthesis attribute HU_SET of I_Q5 is "I_Q5_14"
365      FTCLEX_MXILINX_dropfifo I_Q6 (.C(C),
366                                    .CE(OR_CE_L),
367                                    .CLR(CLR),
368                                    .D(D[6]),
369                                    .L(L),
370                                    .T(T6),
371                                    .Q(Q_DUMMY[6]));
372      // synthesis attribute HU_SET of I_Q6 is "I_Q6_15"
373      FTCLEX_MXILINX_dropfifo I_Q7 (.C(C),
374                                    .CE(OR_CE_L),
375                                    .CLR(CLR),
376                                    .D(D[7]),
377                                    .L(L),
378                                    .T(T7),
379                                    .Q(Q_DUMMY[7]));
380      // synthesis attribute HU_SET of I_Q7 is "I_Q7_16"
381      AND3 I_36_8 (.I0(Q_DUMMY[5]),
382                   .I1(Q_DUMMY[4]),
383                   .I2(T4),
384                   .O(T6));
385      AND2 I_36_11 (.I0(Q_DUMMY[4]),
386                    .I1(T4),
387                    .O(T5));
388      VCC I_36_12 (.P(XLXN_1));
389      AND2 I_36_19 (.I0(Q_DUMMY[1]),
390                    .I1(Q_DUMMY[0]),
391                    .O(T2));
392      AND3 I_36_21 (.I0(Q_DUMMY[2]),
393                    .I1(Q_DUMMY[1]),
394                    .I2(Q_DUMMY[0]),
395                    .O(T3));
396      AND4 I_36_23 (.I0(Q_DUMMY[3]),
397                    .I1(Q_DUMMY[2]),
398                    .I2(Q_DUMMY[1]),
399                    .I3(Q_DUMMY[0]),
400                    .O(T4));
401      AND4 I_36_25 (.I0(Q_DUMMY[6]),
402                    .I1(Q_DUMMY[5]),
403                    .I2(Q_DUMMY[4]),
404                    .I3(T4),
405                    .O(T7));
406      AND5 I_36_29 (.I0(Q_DUMMY[7]),
407                    .I1(Q_DUMMY[6]),
408                    .I2(Q_DUMMY[5]),
409                    .I3(Q_DUMMY[4]),
410                    .I4(T4),
411                    .O(TC_DUMMY));
412      AND2 I_36_33 (.I0(CE),
413                    .I1(TC_DUMMY),
414                    .O(CEO));
415      OR2 I_36_49 (.I0(CE),
416                   .I1(L),
417                   .O(OR_CE_L));
418  endmodule
419  `timescale 1ns / 1ps
420
421  module FDBCE_MXILINX_dropfifo(C,
422                                CE,
423                                CLR,
424                                D,
425                                Q);
426
427      input C;
428      input CE;
429      input CLR;
430      input [7:0] D;
431      output [7:0] Q;
432
433
```

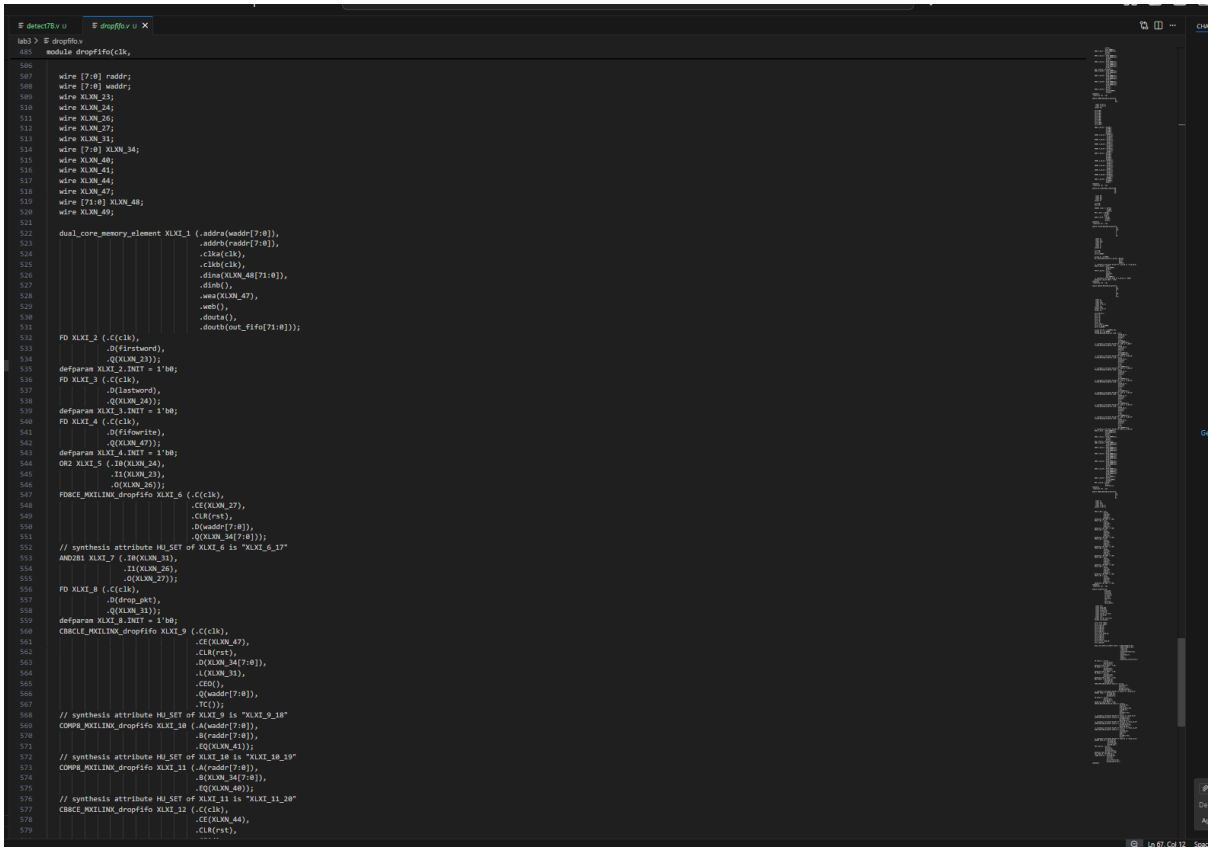
Ln 67, Col 12



```
detect78.v U    dropfifo.v U
lab3 > dropfifo.v
286  module CBBCLE_MXILINX_dropfifo(C,
357      FTCLEX_MXILINX_dropfifo I_Q5 (.C(C),
360                                    .D(D[5]),
361                                    .L(L),
362                                    .T(T5),
363                                    .Q(Q_DUMMY[5]));
364      // synthesis attribute HU_SET of I_Q5 is "I_Q5_14"
365      FTCLEX_MXILINX_dropfifo I_Q6 (.C(C),
366                                    .CE(OR_CE_L),
367                                    .CLR(CLR),
368                                    .D(D[6]),
369                                    .L(L),
370                                    .T(T6),
371                                    .Q(Q_DUMMY[6]));
372      // synthesis attribute HU_SET of I_Q6 is "I_Q6_15"
373      FTCLEX_MXILINX_dropfifo I_Q7 (.C(C),
374                                    .CE(OR_CE_L),
375                                    .CLR(CLR),
376                                    .D(D[7]),
377                                    .L(L),
378                                    .T(T7),
379                                    .Q(Q_DUMMY[7]));
380      // synthesis attribute HU_SET of I_Q7 is "I_Q7_16"
381      AND3 I_36_8 (.I0(Q_DUMMY[5]),
382                   .I1(Q_DUMMY[4]),
383                   .I2(T4),
384                   .O(T6));
385      AND2 I_36_11 (.I0(Q_DUMMY[4]),
386                    .I1(T4),
387                    .O(T5));
388      VCC I_36_12 (.P(XLXN_1));
389      AND2 I_36_19 (.I0(Q_DUMMY[1]),
390                    .I1(Q_DUMMY[0]),
391                    .O(T2));
392      AND3 I_36_21 (.I0(Q_DUMMY[2]),
393                    .I1(Q_DUMMY[1]),
394                    .I2(Q_DUMMY[0]),
395                    .O(T3));
396      AND4 I_36_23 (.I0(Q_DUMMY[3]),
397                    .I1(Q_DUMMY[2]),
398                    .I2(Q_DUMMY[1]),
399                    .I3(Q_DUMMY[0]),
400                    .O(T4));
401      AND4 I_36_25 (.I0(Q_DUMMY[6]),
402                    .I1(Q_DUMMY[5]),
403                    .I2(Q_DUMMY[4]),
404                    .I3(T4),
405                    .O(T7));
406      AND5 I_36_29 (.I0(Q_DUMMY[7]),
407                    .I1(Q_DUMMY[6]),
408                    .I2(Q_DUMMY[5]),
409                    .I3(Q_DUMMY[4]),
410                    .I4(T4),
411                    .O(TC_DUMMY));
412      AND2 I_36_33 (.I0(CE),
413                    .I1(TC_DUMMY),
414                    .O(CEO));
415      OR2 I_36_49 (.I0(CE),
416                   .I1(L),
417                   .O(OR_CE_L));
418  endmodule
419  `timescale 1ns / 1ps
420
421  module FDBCE_MXILINX_dropfifo(C,
422                                CE,
423                                CLR,
424                                D,
425                                Q);
426
427      input C;
428      input CE;
429      input CLR;
430      input [7:0] D;
431      output [7:0] Q;
432
433
```

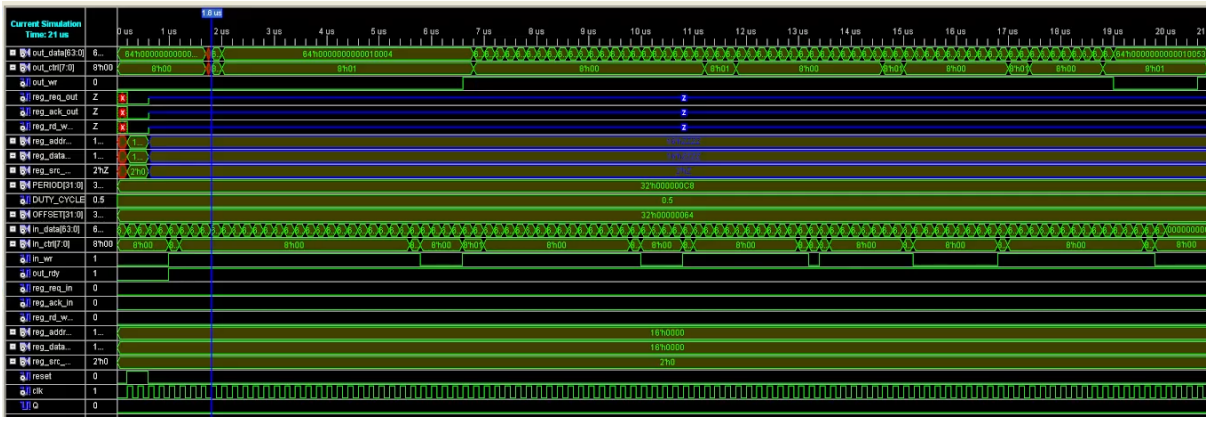


```
detect78.v U   dropfifo.v U
lab3 > dropfifo.v
421  module FDBCE_MXILINX_dropfifo(C,
424                                D,
425                                Q);
426
427    input C;
428    input CE;
429    input CLR;
430    input [7:0] D;
431    output [7:0] Q;
432
433
434    FDCE I_Q0 (.C(C),
435               .CE(CE),
436               .CLR(CLR),
437               .D(D[0]),
438               .Q(Q[0]));
439    defparam I_Q0.INIT = 1'b0;
440    FDCE I_Q1 (.C(C),
441               .CE(CE),
442               .CLR(CLR),
443               .D(D[1]),
444               .Q(Q[1]));
445    defparam I_Q1.INIT = 1'b0;
446    FDCE I_Q2 (.C(C),
447               .CE(CE),
448               .CLR(CLR),
449               .D(D[2]),
450               .Q(Q[2]));
451    defparam I_Q2.INIT = 1'b0;
452    FDCE I_Q3 (.C(C),
453               .CE(CE),
454               .CLR(CLR),
455               .D(D[3]),
456               .Q(Q[3]));
457    defparam I_Q3.INIT = 1'b0;
458    FDCE I_Q4 (.C(C),
459               .CE(CE),
460               .CLR(CLR),
461               .D(D[4]),
462               .Q(Q[4]));
463    defparam I_Q4.INIT = 1'b0;
464    FDCE I_Q5 (.C(C),
465               .CE(CE),
466               .CLR(CLR),
467               .D(D[5]),
468               .Q(Q[5]));
469    defparam I_Q5.INIT = 1'b0;
470    FDCE I_Q6 (.C(C),
471               .CE(CE),
472               .CLR(CLR),
473               .D(D[6]),
474               .Q(Q[6]));
475    defparam I_Q6.INIT = 1'b0;
476    FDCE I_Q7 (.C(C),
477               .CE(CE),
478               .CLR(CLR),
479               .D(D[7]),
480               .Q(Q[7]));
481    defparam I_Q7.INIT = 1'b0;
482  endmodule
483  `timescale 1ns / 1ps
484
485  module dropfifo(clk,
486                  drop_pkt,
487                  fiforead,
488                  fifowrite,
489                  firstword,
490                  in_fifo,
491                  lastword,
492                  rst,
493                  out_fifo,
494                  valid_data);
495
```



```
detect78.v U   dropfifo.v U
lab3 > dropfifo.v
485  module dropfifo(clk,
506
507    wire [7:0] raddr;
508    wire [7:0] waddr;
509    wire XLXN_23;
510    wire XLXN_24;
511    wire XLXN_26;
512    wire XLXN_27;
513    wire XLXN_31;
514    wire [7:0] XLXN_34;
515    wire XLXN_40;
516    wire XLXN_41;
517    wire XLXN_44;
518    wire XLXN_47;
519    wire [71:0] XLXN_48;
520    wire XLXN_49;
521
522    dual_core_memory_element XLXI_1 (.addra(waddr[7:0]),
523                                     .addrb(raddr[7:0]),
524                                     .clka(clk),
525                                     .clkb(clk),
526                                     .dina(XLXN_48[71:0]),
527                                     .dinb(),
528                                     .wea(XLXN_47),
529                                     .web(),
530                                     .douta(),
531                                     .doutb(out_fifo[71:0]));
532    FD XLXI_2 (.C(clk),
533               .D(firstword),
534               .Q(XLXN_23));
535    defparam XLXI_2.INIT = 1'b0;
536    FD XLXI_3 (.C(clk),
537               .D(lastword),
538               .Q(XLXN_24));
539    defparam XLXI_3.INIT = 1'b0;
540    FD XLXI_4 (.C(clk),
541               .D(fifowrite),
542               .Q(XLXN_47));
543    defparam XLXI_4.INIT = 1'b0;
544    OR2 XLXI_5 (.I0(XLXN_24),
545                .I1(XLXN_23),
546                .O(XLXN_26));
547    FDBCE_MXILINX_dropfifo XLXI_6 (.C(clk),
548                                   .CE(XLXN_27),
549                                   .CLR(rst),
550                                   .D(waddr[7:0]),
551                                   .Q(XLXN_34[7:0]));
552    // synthesis attribute HU_SET of XLXI_6 is "XLXI_6_17"
553    AND2B1 XLXI_7 (.I0(XLXN_31),
554                   .I1(XLXN_26),
555                   .O(XLXN_27));
556    FD XLXI_8 (.C(clk),
557               .D(drop_pkt),
558               .Q(XLXN_31));
559    defparam XLXI_8.INIT = 1'b0;
560    CB8CLE_MXILINX_dropfifo XLXI_9 (.C(clk),
561                                    .CE(XLXN_47),
562                                    .CLR(rst),
563                                    .D(XLXN_34[7:0]),
564                                    .L(XLXN_31),
565                                    .CEO(),
566                                    .Q(waddr[7:0]),
567                                    .TC());
568    // synthesis attribute HU_SET of XLXI_9 is "XLXI_9_18"
569    COMP8_MXILINX_dropfifo XLXI_10 (.A(waddr[7:0]),
570                                    .B(raddr[7:0]),
571                                    .EQ(XLXN_41));
572    // synthesis attribute HU_SET of XLXI_10 is "XLXI_10_19"
573    COMP8_MXILINX_dropfifo XLXI_11 (.A(raddr[7:0]),
574                                    .B(XLXN_34[7:0]),
575                                    .EQ(XLXN_40));
576    // synthesis attribute HU_SET of XLXI_11 is "XLXI_11_20"
577    CB8CE_MXILINX_dropfifo XLXI_12 (.C(clk),
578                                    .CE(XLXN_44),
579                                    .CLR(rst),
```

9. Simulate the mini-IDS using the ids_tb.tbw testbench. Run the testbench and take a screen shot. Describe what the testbench does and how it shows that the mini-IDS is functioning.



The testbench sends packets into the mini-IDS and watches how the outputs change. It provides the clock and reset signals and then drives input data into the system, just like packets arriving from a network.

Here I show the simulation waveform. We can see that when input data is written and accepted, the IDS processes it correctly. When no attack pattern is present, the output data is forwarded normally. When a match signal is triggered, the corresponding packet is dropped, which confirms that the IDS logic is working as intended.

10. Explain the pattern matching algorithm in the report

    The mini-IDS checks incoming data to see if a specific pattern appears. Data arrives continuously, so it is first stored and then checked in small sections. A 7-byte window moves across the data, and each window is compared with the pattern. If any window matches, the system reports a detection. if an attack pattern is detected, the entire packet is dropped. Otherwise, the packet is forwarded normally.

11. Include the answers to your lab in this report.

    i) What is the purpose of AMASK[6:0]?

AMASK [6:0] is used to decide which bytes need to be compared. Each bit in AMASK controls one byte. If a bit is set, that byte is ignored during comparison. This makes the pattern matching more flexible.

ii) What exactly does busmerge.v do?

The busmerge module joins two data buses into one larger bus. It combines old data and new data so they can be checked together. This helps the system compare longer data sequences.

iii) What do the comp8 modules do in this schematic?

The comp8 modules compare two 8-bit values. Each comp8 checks whether one byte of input data matches the pattern. Several comp8 modules are used together to compare multiple bytes.

iv) What is the purpose of dual9Bmem in dropfifo.sch?

The dual9Bmem module is a 9-byte memory used as a FIFO buffer. It temporarily stores packet data while allowing read and write at the same time. When a match is detected, the system can stop the packet from being forwarded.