Here is a link to the github repo that I used to store the schematic and Verilog file generated for this lab: https://github.com/yuezhenglingluan/USC_EE533_lab3

Part 1:

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

I feel that these autos generated Verilog file makes sense, since the schematic is just components wired together, the Verilog file very precisely represents this by having wires defined to connect a bunch on modules.

However that sometimes the testbench would fail to run because the auto generated .vf files didn't includes the sub modules that it is using, and I have to manually add those includes, which could be annony.

And I feel that entering schematics is easier, since you can easily know the function of a schematic by knowing what does each modules do and how they are wired together. However, if I have to create something that can be done easily in Verilog (such as creating a mux), with just a few lines of verilog code and would need to have a lot of wires being connected on schematic if the mux input and output are very wide (for example, 64 bits).

Part 2:

a) Explain the pattern matching algorithm in the report

The Pattern matching algorithm in ids_sim.v is trying to detect specific patterns in a single packet, and it works as follows:

The pattern that the algorithm is trying to match is stored in two software registers: pattern_high and pattern_low, each of them are 32 bits, and together {pattern_high, pattern_low} they formed the 64-bit long pattern that we want to match.

The incoming data from in_data is buffered in a fallthrough_small_fifo called input_fifo.

The detect7B module matcher is responsible for checking the incoming packet to see if the packet data matches the pattern that we are trying to match.

matcher_ce enables the data flow to the detect7B;

matcher_en enables the matcher to actively scan the input data for matching pattern.

matcher_reset resets the matcher at the beginning of each new packet or when ids_cmd[0]/reset is set to high.

The dropfifo module called drop_fifo is responsible for outputting the packet data if a match is not detected by the detect7B module or drop the packet if a pattern match is detected by setting the valid_data output pin to active low.

There is also a state machine built into the file:

In state START:

- Detects the beginning of a packet when in_fifo_ctrl_p is not equal to 0
- Moves to HEADER state when a new packet starts

In state HEADER:

- Reads the first three control words, treating them as header information.
- Moves on to PAYLOAD state once the header is processed

In state PAYLOAD:

- Reads packet data.
- If matcher_match is high (indicating a pattern match), the match counter matches is incremented.
- When in_fifo_ctrl_p is not equal to 0, it indicates the end of a packet, and it would reset the matcher and return to START state.

b) Include the answers to your lab in this report.

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

It is found in the comparator module, and only when it's set to 7'b111_1111, it will allow the comparator to output the comparison results, else it will make the comparator to always indicate match (output always set to 1).

ii) What exactly does busmerge.v do?

It accepts two different input (one 48 bits long and the other 64 bits long) and merges them into a single 112 bits output

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

It takes two 8 bits inputs and uses XNOR gates and AND gates to output 1 (indicating that the inputs are equal) if the two 8 bits input are the same.

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

It works like a FIFO and is used to accept and store the packet that comes in from the reg9B module if the content in the packet doesn't match the pattern and output the packet.
If there is a match detected, the drop_pkt input will be 1, which will result in the DFF that is responsible for incrementing the write address of the mem to be always 0;

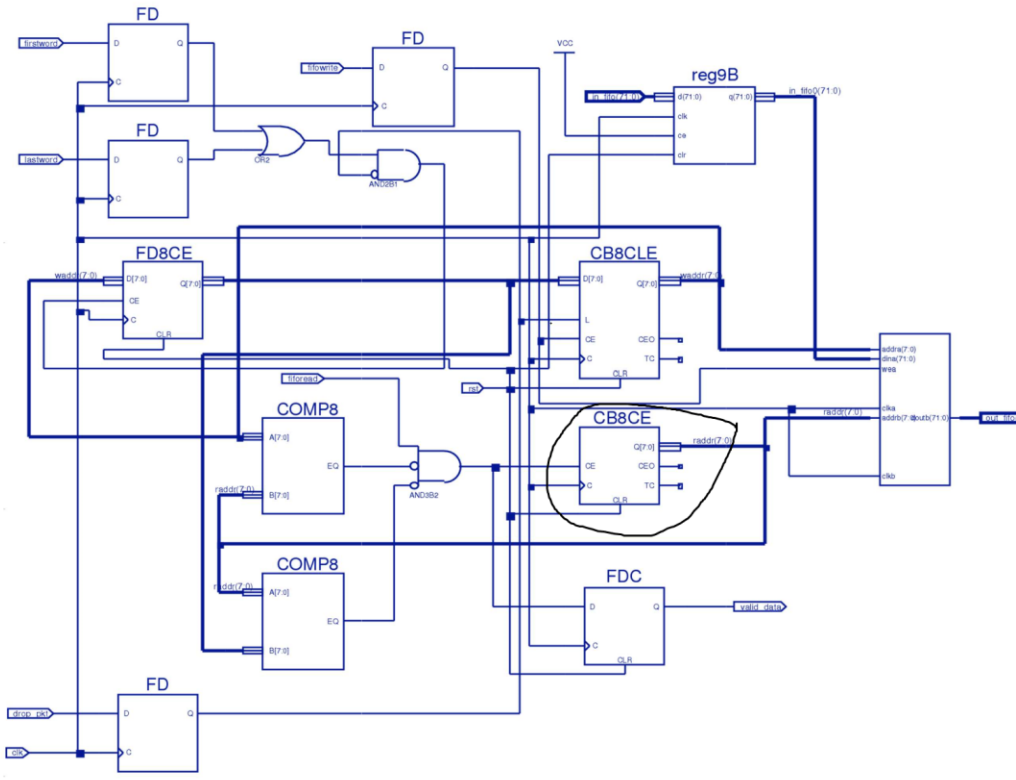
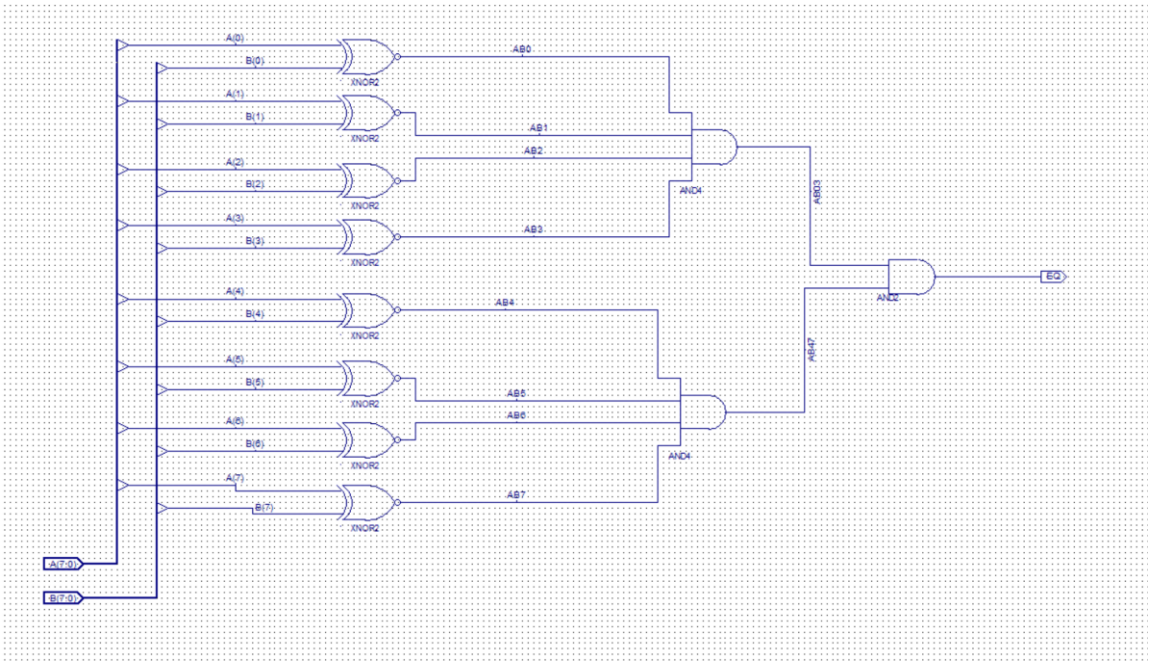


Figure 6: dropfifo.sch

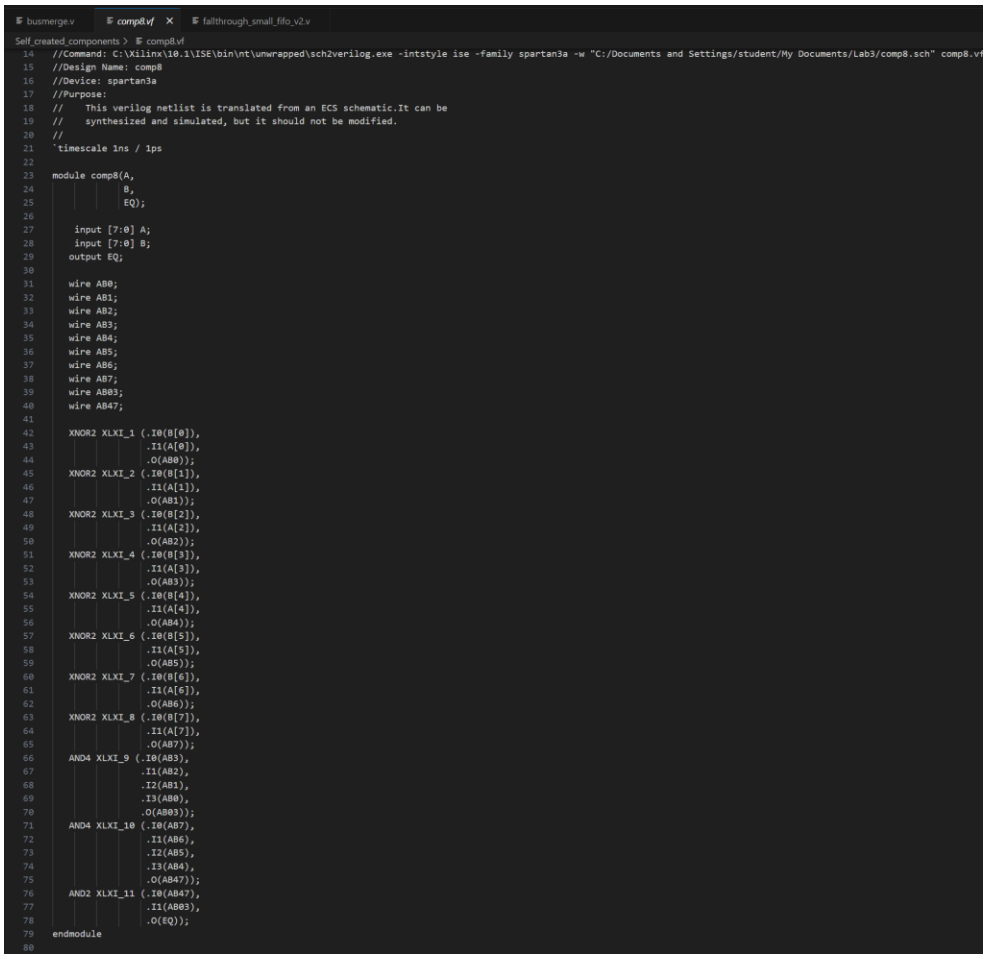
And the CB8CE in the circle will continue to increment the read address of the mem until the read address is equal to the write address, which means that the previous packet has been sent out successfully, and in this case, the two comp module will be outputting 0, which will be setting the valid_data output to 0, indicating that the output packet is not valid and needs to be dropped.

c) Please also include all of the screen capture of Schematics as well as generated Verilog

Here is the schematic of my comp8:



Here is the generated Verilog:

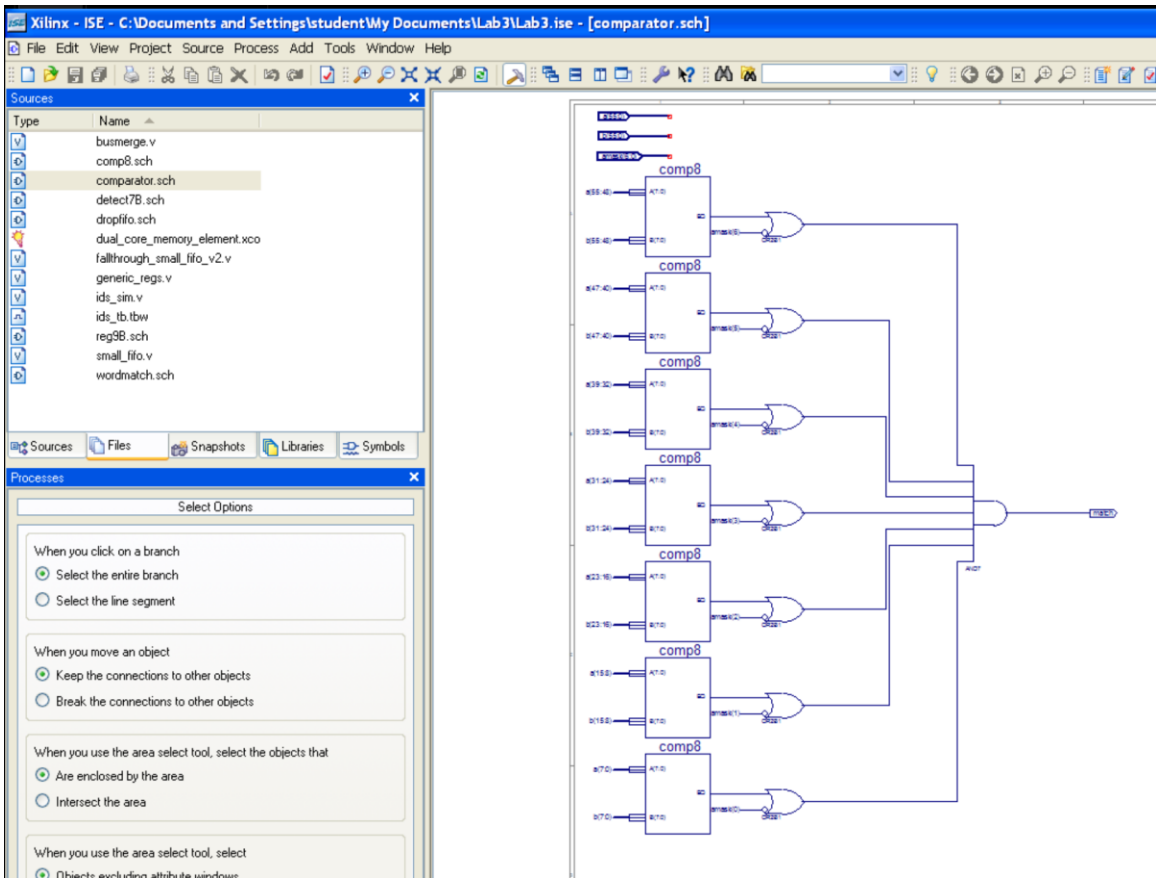


busmerge.v | comp8.vf | fallthrough_small_fifo_v2.v

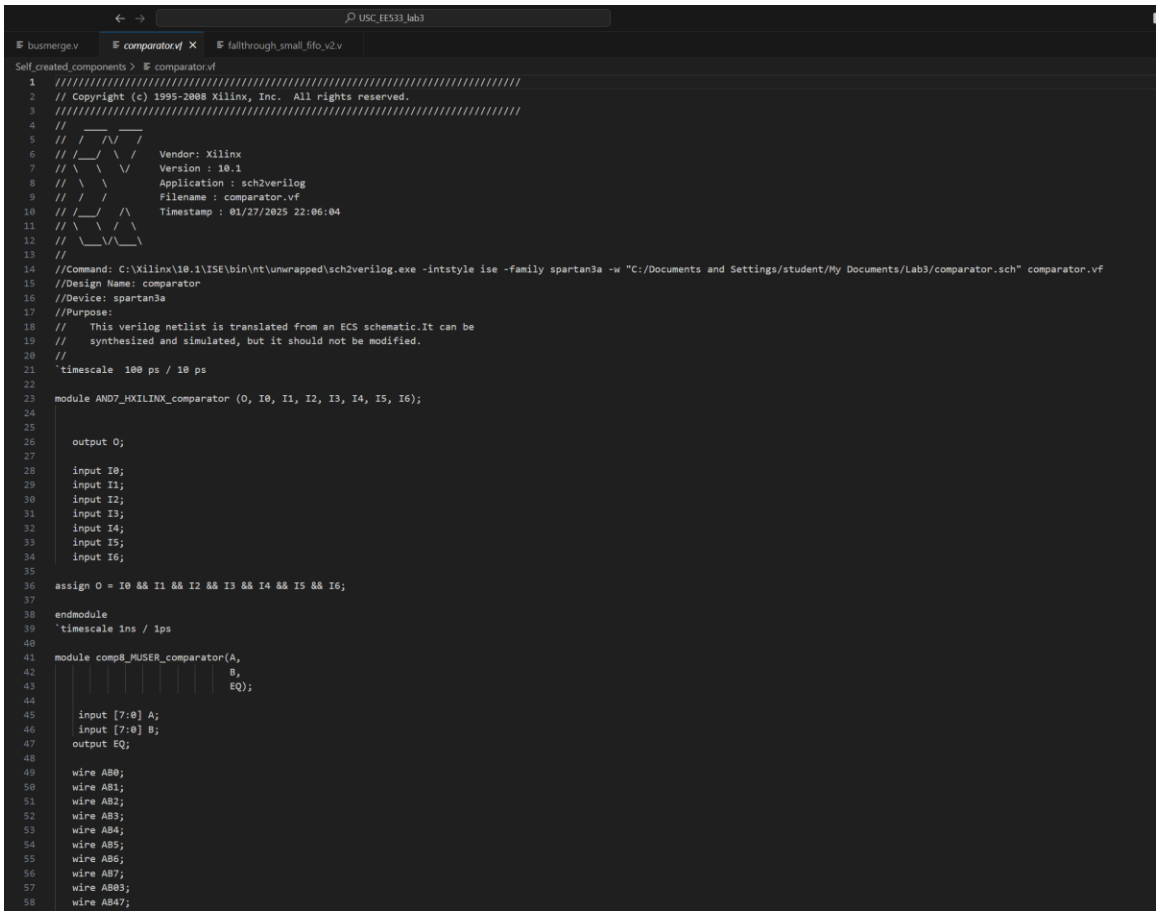
Self_created_components > comp8.vf

```
//Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family spartan3a -w "C:/Documents and Settings/student/My Documents/Lab3/comp8.sch" comp8.vf
//Design Name: comp8
//Device: spartan3a
//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 comp8(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;

   XNOR2 XLXI_1 (.I0(B[0]),
                 .I1(A[0]),
                 .O(AB0));
   XNOR2 XLXI_2 (.I0(B[1]),
                 .I1(A[1]),
                 .O(AB1));
   XNOR2 XLXI_3 (.I0(B[2]),
                 .I1(A[2]),
                 .O(AB2));
   XNOR2 XLXI_4 (.I0(B[3]),
                 .I1(A[3]),
                 .O(AB3));
   XNOR2 XLXI_5 (.I0(B[4]),
                 .I1(A[4]),
                 .O(AB4));
   XNOR2 XLXI_6 (.I0(B[5]),
                 .I1(A[5]),
                 .O(AB5));
   XNOR2 XLXI_7 (.I0(B[6]),
                 .I1(A[6]),
                 .O(AB6));
   XNOR2 XLXI_8 (.I0(B[7]),
                 .I1(A[7]),
                 .O(AB7));
   AND4 XLXI_9 (.I0(AB3),
                .I1(AB2),
                .I2(AB1),
                .I3(AB0),
                .O(AB03));
   AND4 XLXI_10 (.I0(AB7),
                 .I1(AB6),
                 .I2(AB5),
                 .I3(AB4),
                 .O(AB47));
   AND2 XLXI_11 (.I0(AB47),
                 .I1(AB03),
                 .O(EQ));
endmodule

```

Here is the schematic of my comparator:



And the generated Verilog code:



```
busmerge.v   comparator.vf   fallthrough_small_fifo_v2.v
Self_created_components > comparator.vf
```

```
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 : comparator.vf
10 // /___/   /\     Timestamp : 01/27/2025 22:06:04
11 // \   \  /  \
12 //  \___\/\___\
13 //
14 //Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family spartan3a -w "C:/Documents and Settings/student/My Documents/Lab3/comparator.sch" comparator.vf
15 //Design Name: comparator
16 //Device: spartan3a
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  100 ps / 10 ps
22
23 module AND7_HXILINX_comparator (O, I0, I1, I2, I3, I4, I5, I6);
24
25
26   output O;
27
28   input I0;
29   input I1;
30   input I2;
31   input I3;
32   input I4;
33   input I5;
34   input I6;
35
36 assign O = I0 && I1 && I2 && I3 && I4 && I5 && I6;
37
38 endmodule
39 `timescale 1ns / 1ps
40
41 module comp8_MUSER_comparator(A,
42                               B,
43                               EQ);
44
45   input [7:0] A;
46   input [7:0] B;
47   output EQ;
48
49   wire AB0;
50   wire AB1;
51   wire AB2;
52   wire AB3;
53   wire AB4;
54   wire AB5;
55   wire AB6;
56   wire AB7;
57   wire AB03;
58   wire AB47;
```

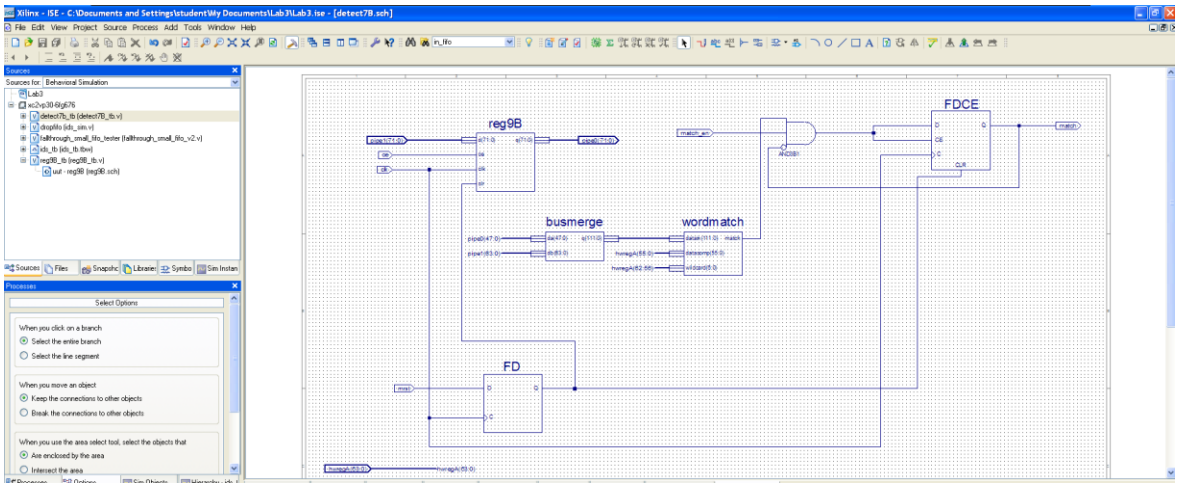
```
59
60     AND4 I_36_32 (.I0(AB7),
61                   .I1(AB6),
62                   .I2(AB5),
63                   .I3(AB4),
64                   .O(AB47));
65     XNOR2 I_36_33 (.I0(B[6]),
66                    .I1(A[6]),
67                    .O(AB6));
68     XNOR2 I_36_34 (.I0(B[7]),
69                    .I1(A[7]),
70                    .O(AB7));
71     XNOR2 I_36_35 (.I0(B[5]),
72                    .I1(A[5]),
73                    .O(AB5));
74     XNOR2 I_36_36 (.I0(B[4]),
75                    .I1(A[4]),
76                    .O(AB4));
77     AND4 I_36_41 (.I0(AB3),
78                   .I1(AB2),
79                   .I2(AB1),
80                   .I3(AB0),
81                   .O(AB03));
82     XNOR2 I_36_42 (.I0(B[2]),
83                    .I1(A[2]),
84                    .O(AB2));
85     XNOR2 I_36_43 (.I0(B[3]),
86                    .I1(A[3]),
87                    .O(AB3));
88     XNOR2 I_36_44 (.I0(B[1]),
89                    .I1(A[1]),
90                    .O(AB1));
91     XNOR2 I_36_45 (.I0(B[0]),
92                    .I1(A[0]),
93                    .O(AB0));
94     AND2 I_36_50 (.I0(AB47),
95                   .I1(AB03),
96                   .O(EQ));
97 endmodule
98 `timescale 1ns / 1ps
```

```
 99
100  module comparator(a,
101                    amask,
102                    b,
103                    match);
104
105     input [55:0] a;
106     input [6:0] amask;
107     input [55:0] b;
108    output match;
109
110     wire XLXN_20;
111     wire XLXN_21;
112     wire XLXN_22;
113     wire XLXN_23;
114     wire XLXN_28;
115     wire XLXN_30;
116     wire XLXN_32;
117     wire XLXN_40;
118     wire XLXN_41;
119     wire XLXN_42;
120     wire XLXN_43;
121     wire XLXN_44;
122     wire XLXN_45;
123     wire XLXN_46;
124
125     comp8_MUSER_comparator XLXI_1 (.A(a[55:48]),
126                                    .B(b[55:48]),
127                                    .EQ(XLXN_20));
128     comp8_MUSER_comparator XLXI_2 (.A(a[47:40]),
129                                    .B(b[47:40]),
130                                    .EQ(XLXN_21));
131     comp8_MUSER_comparator XLXI_3 (.A(a[39:32]),
132                                    .B(b[39:32]),
133                                    .EQ(XLXN_22));
134     comp8_MUSER_comparator XLXI_4 (.A(a[31:24]),
135                                    .B(b[31:24]),
136                                    .EQ(XLXN_23));
137     OR2B1 XLXI_13 (.I0(amask[5]),
138                    .I1(XLXN_21),
139                    .O(XLXN_45));
140     OR2B1 XLXI_14 (.I0(amask[4]),
141                    .I1(XLXN_22),
142                    .O(XLXN_44));
143     OR2B1 XLXI_15 (.I0(amask[3]),
144                    .I1(XLXN_23),
145                    .O(XLXN_40));
146     OR2B1 XLXI_16 (.I0(amask[6]),
147                    .I1(XLXN_20),
148                    .O(XLXN_46));
149     comp8_MUSER_comparator XLXI_17 (.A(a[23:16]),
```

```
150                                     .B(b[23:16]),
151                                     .EQ(XLXN_28));
152     OR2B1 XLXI_18 (.I0(amask[2]),
153                    .I1(XLXN_28),
154                    .O(XLXN_41));
155     comp8_MUSER_comparator XLXI_19 (.A(a[15:8]),
156                                     .B(b[15:8]),
157                                     .EQ(XLXN_30));
158     OR2B1 XLXI_20 (.I0(amask[1]),
159                    .I1(XLXN_30),
160                    .O(XLXN_42));
161     comp8_MUSER_comparator XLXI_21 (.A(a[7:0]),
162                                     .B(b[7:0]),
163                                     .EQ(XLXN_32));
164     OR2B1 XLXI_22 (.I0(amask[0]),
165                    .I1(XLXN_32),
166                    .O(XLXN_43));
167     AND7_HXILINX_comparator XLXI_23 (.I0(XLXN_43),
168                                      .I1(XLXN_42),
169                                      .I2(XLXN_41),
170                                      .I3(XLXN_40),
171                                      .I4(XLXN_44),
172                                      .I5(XLXN_45),
173                                      .I6(XLXN_46),
174                                      .O(match));
175     // synthesis attribute HU_SET of XLXI_23 is "XLXI_23_0"
176   endmodule
177
```

Here is the schematic of the detect7B :

I addes the pipe0 as outputs for debug purposes.



Here is the generated Verilog code:

I added these includes so the testbench could compile correctly:

```
21    `timescale 1ns / 1ps
22    `include "busmerge.v"
23    `include "wordmatch.vf"
24    module detect7B(ce,
```

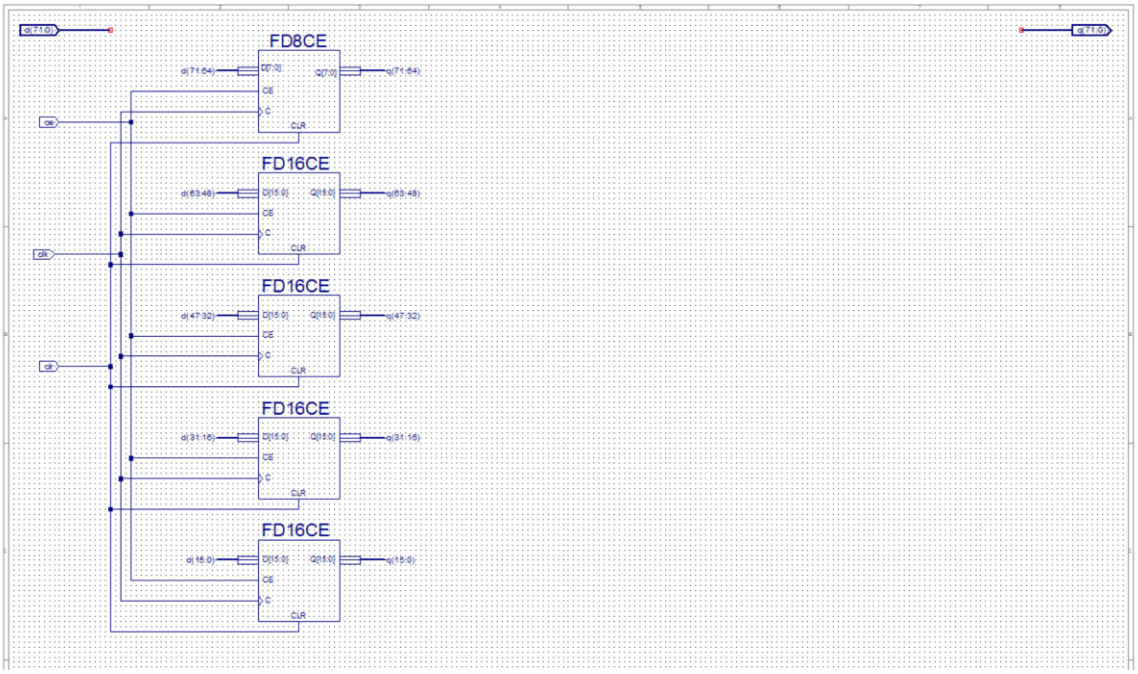
busmerge.v | detect7B.vf | fallthrough_small_fifo_v2.v

Self_created_components > detect7B.vf

```
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 : detect7B.vf
10  // /___/   /\     Timestamp : 01/28/2025 19:37:46
11  // \   \  /  \
12  //  \___\/\___\
13  //
14  //Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family spartan3a -w "C:/Documents and Settings/student/My Documents/Lab3/detect7B.sch" detect7B.vf
15  //Design Name: detect7B
16  //Device: spartan3a
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 detect7B(ce,
24                  clk,
25                  hwregA,
26                  match_en,
27                  mrst,
28                  pipe1,
29                  match);
30
31    input ce;
32    input clk;
33    input [63:0] hwregA;
34    input match_en;
35    input mrst;
36    input [71:0] pipe1;
37    output match;
38
39    wire [71:0] pipe0;
40    wire XLXN_18;
41    wire [111:0] XLXN_25;
42    wire XLXN_46;
43    wire XLXN_49;
44    wire match_DUMMY;
45
46    assign match = match_DUMMY;
47    reg9B XLXI_1 (.ce(ce),
48                  .clk(clk),
49                  .clr(XLXN_18),
50                  .d(pipe1[71:0]),
51                  .q(pipe0[71:0]));
52    busmerge XLXI_2 (.da(pipe0[47:0]),
53                     .db(pipe1[63:0]),
54                     .q(XLXN_25[111:0]));
55    wordmatch XLXI_3 (.datacomp(hwregA[55:0]),
56                      .datain(XLXN_25[111:0]),
57                      .wildcard(hwregA[62:56]),
58                      .match(XLXN_49));
59    FDCE XLXI_4 (.C(clk),
60                 .CE(XLXN_46),
61                 .CLR(XLXN_18),
62                 .D(XLXN_46),
63                 .Q(match_DUMMY));
64    defparam XLXI_4.INIT = 1'b0;
65    FD XLXI_5 (.C(clk),
66               .D(mrst),
67               .Q(XLXN_18));
```

```
68      defparam XLXI_5.INIT = 1'b0;
69      AND3B1 XLXI_15 (.I0(match_DUMMY),
70                      .I1(match_en),
71                      .I2(XLXN_49),
72                      .O(XLXN_46));
73  endmodule
74
```

Here is the schematic of the reg9B:



Here is the generated Verilog:

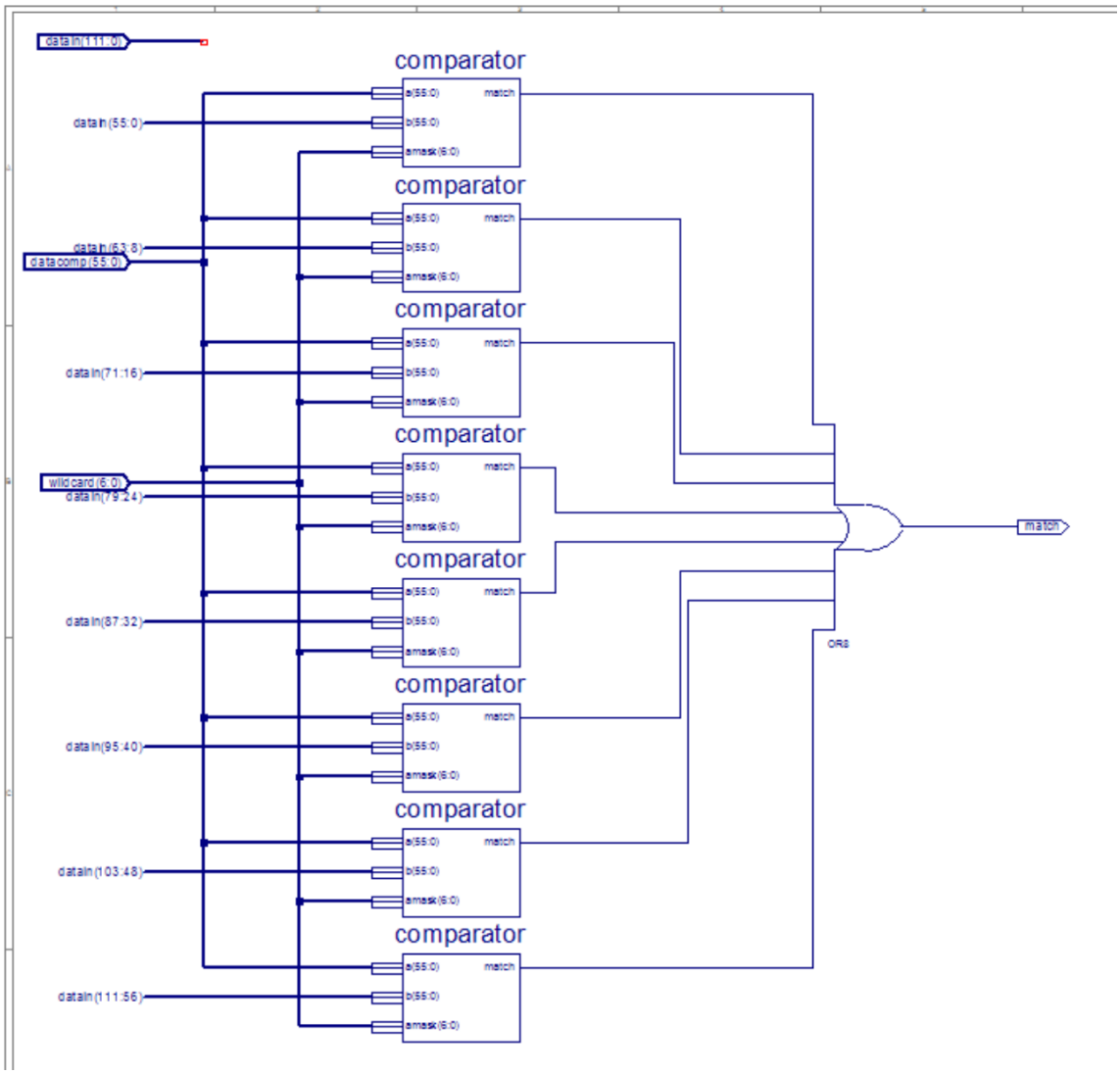
busmerge.v | reg9B.vf | fallthrough_small_fifo_v2.v

Self_created_components > reg9B.vf

```
//////////////////////////////////////////////////////////////////////////////////
// Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.
//////////////////////////////////////////////////////////////////////////////////
//   ____  ____
//  /   /\/   /
// /___/  \  /    Vendor: Xilinx
// \   \   \/     Version : 10.1
//  \   \         Application : sch2verilog
//  /   /         Filename : reg9B.vf
// /___/   /\     Timestamp : 01/28/2025 19:20:25
// \   \  /  \
//  \___\/\___\
//
//Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family spartan3a -w "C:/Documents and Settings/student/My Documents/Lab3/reg9B.sch" reg9B.vf
//Design Name: reg9B
//Device: spartan3a
//Purpose:
//    This verilog netlist is translated from an ECS schematic.It can be
//    synthesized and simulated, but it should not be modified.
//
`timescale 100 ps / 10 ps

module FD16CE_HXILINX_reg9B(Q, C, CE, CLR, D);


  output [15:0]     Q;

  input        C;
  input        CE;
  input        CLR;
  input  [15:0]     D;

  reg    [15:0]     Q;

  always @(posedge C or posedge CLR)
    begin
  if (CLR)
    Q <= 16'b0000_0000_0000_0000;
  else if (CE)
          Q <= D;
    end


endmodule
`timescale 100 ps / 10 ps

module FD8CE_HXILINX_reg9B(Q, C, CE, CLR, D);


  output [7:0]      Q;

  input        C;
  input        CE;
  input        CLR;
  input  [7:0]      D;

  reg    [7:0]      Q;

  always @(posedge C or posedge CLR)
    begin
  if (CLR)
    Q <= 8'b0000_0000;
  else if (CE)
          Q <= D;
    end


```

```
endmodule
`timescale 1ns / 1ps

module reg9B(ce,
             clk,
             clr,
             d,
             q);

    input ce;
    input clk;
    input clr;
    input [71:0] d;
   output [71:0] q;


   FD8CE_HXILINX_reg9B XLXI_1 (.C(clk),
                               .CE(ce),
                               .CLR(clr),
                               .D(d[71:64]),
                               .Q(q[71:64]));
   // synthesis attribute HU_SET of XLXI_1 is "XLXI_1_0"
   FD16CE_HXILINX_reg9B XLXI_2 (.C(clk),
                                .CE(ce),
                                .CLR(clr),
                                .D(d[63:48]),
                                .Q(q[63:48]));
   // synthesis attribute HU_SET of XLXI_2 is "XLXI_2_1"
   FD16CE_HXILINX_reg9B XLXI_3 (.C(clk),
                                .CE(ce),
                                .CLR(clr),
                                .D(d[47:32]),
                                .Q(q[47:32]));
   // synthesis attribute HU_SET of XLXI_3 is "XLXI_3_2"
   FD16CE_HXILINX_reg9B XLXI_4 (.C(clk),
                                .CE(ce),
                                .CLR(clr),
                                .D(d[31:16]),
                                .Q(q[31:16]));
   // synthesis attribute HU_SET of XLXI_4 is "XLXI_4_3"
   FD16CE_HXILINX_reg9B XLXI_5 (.C(clk),
                                .CE(ce),
                                .CLR(clr),
                                .D(d[15:0]),
                                .Q(q[15:0]));
   // synthesis attribute HU_SET of XLXI_5 is "XLXI_5_4"
endmodule

```

Here is the schematic of the wordmatch:



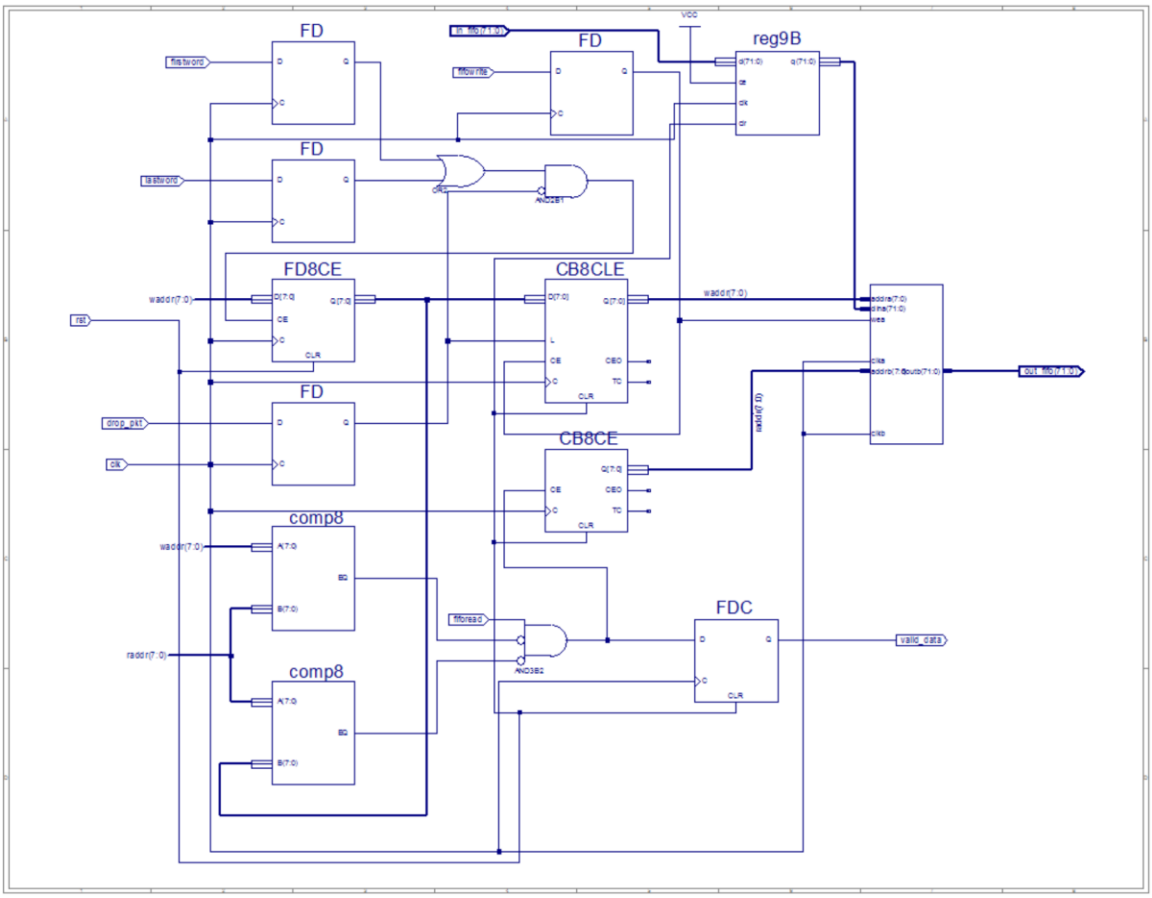
Here is the generated Verilog code:

busmerge.v | wordmatch.vf | fallthrough_small_fifo_v2.v

Self_created_components > wordmatch.vf

```
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 : wordmatch.vf
10 // /___/   /\     Timestamp : 01/27/2025 22:21:42
11 // \   \  /  \
12 //  \___\/\___\
13 //
14 //Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -family spartan3a -w "C:/Documents and Settings/student/My Documents/Lab3/wordmatch.sch" wordmatch.vf
15 //Design Name: wordmatch
16 //Device: spartan3a
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  100 ps / 10 ps
22
23 module OR8_HXILINX_wordmatch (O, I0, I1, I2, I3, I4, I5, I6, I7);
24
25
26    output O;
27
28    input I0;
29    input I1;
30    input I2;
31    input I3;
32    input I4;
33    input I5;
34    input I6;
35    input I7;
36
37 assign O = (I0 || I1 || I2 || I3 || I4 || I5 || I6 || I7);
38
39 endmodule
40 `timescale 1ns / 1ps
41
42 module wordmatch(datacomp,
43                  datain,
44                  wildcard,
45                  match);
46
47    input [55:0] datacomp;
48    input [111:0] datain;
49    input [6:0] wildcard;
50    output match;
51
52    wire XLXN_23;
53    wire XLXN_24;
54    wire XLXN_25;
55    wire XLXN_26;
56    wire XLXN_27;
57    wire XLXN_28;
58    wire XLXN_29;
59    wire XLXN_30;
60
61    comparator XLXI_1 (.a(datacomp[55:0]),
62                       .amask(wildcard[6:0]),
63                       .b(datain[55:0]),
64                       .match(XLXN_23));
65    comparator XLXI_2 (.a(datacomp[55:0]),
66                       .amask(wildcard[6:0]),
67                       .b(datain[63:8]),
68                       .match(XLXN_24));
69    comparator XLXI_3 (.a(datacomp[55:0]),
70                       .amask(wildcard[6:0]),
71                       .b(datain[71:16]),
72                       .match(XLXN_25));
73    comparator XLXI_4 (.a(datacomp[55:0]),
74                       .amask(wildcard[6:0]),
75                       .b(datain[79:24]),
76                       .match(XLXN_26));
77    comparator XLXI_5 (.a(datacomp[55:0]),
78                       .amask(wildcard[6:0]),
79                       .b(datain[87:32]),
80                       .match(XLXN_27));
81    comparator XLXI_6 (.a(datacomp[55:0]),
82                       .amask(wildcard[6:0]),
83                       .b(datain[95:40]),
84                       .match(XLXN_28));
85    comparator XLXI_7 (.a(datacomp[55:0]),
86                       .amask(wildcard[6:0]),
87                       .b(datain[103:48]),
88                       .match(XLXN_29));
89    comparator XLXI_8 (.a(datacomp[55:0]),
90                       .amask(wildcard[6:0]),
91                       .b(datain[111:56]),
92                       .match(XLXN_30));
93    OR8_HXILINX_wordmatch XLXI_9 (.I0(XLXN_30),
94                                  .I1(XLXN_29),
95                                  .I2(XLXN_28),
96                                  .I3(XLXN_27),
97                                  .I4(XLXN_26),
98                                  .I5(XLXN_25),
99                                  .I6(XLXN_24),
100                                 .I7(XLXN_23),
101                                 .O(match));
102   // synthesis attribute HU_SET of XLXI_9 is "XLXI_9_0"
103 endmodule
104
```

Here is the schematic of the dropfifo:



Here is the generated verilog code:

busmerge.v | dropfifo.vf | fallthrough_small_fifo_v2.v

Self_created_components > dropfifo.vf

```
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 : dropfifo.vf
10 // /___/   /\     Timestamp : 01/28/2025 21:37:20
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/My Documents/Lab3/dropfifo.sch" dropfifo.vf
15 //Design Name: dropfifo
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 FTCE_MXILINX_dropfifo(C,
24                              CE,
25                              CLR,
26                              T,
27                              Q);
28
29     input C;
30     input CE;
31     input CLR;
32     input T;
33     output Q;
34
35     wire TQ;
36     wire Q_DUMMY;
37
38     assign Q = Q_DUMMY;
39     XOR2 I_36_32 (.I0(T),
40                   .I1(Q_DUMMY),
41                   .O(TQ));
42     FDCE I_36_35 (.C(C),
43                   .CE(CE),
44                   .CLR(CLR),
45                   .D(TQ),
46                   .Q(Q_DUMMY));
47     // synthesis attribute RLOC of I_36_35 is "X0Y0"
48     defparam I_36_35.INIT = 1'b0;
49 endmodule
50 `timescale 1ns / 1ps
51
52 module CBBCE_MXILINX_dropfifo(C,
53                               CE,
54                               CLR,
55                               CEO,
56                               Q,
57                               TC);
58
59     input C;
60     input CE;
61     input CLR;
62     output CEO;
63     output [7:0] Q;
64     output TC;
65
66     wire T2;
67     wire T3;
```

```
    wire T4;
    wire T5;
    wire T6;
    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));
    AND2 I_36_24 (.I0(Q_DUMMY[1]),
                 .I1(Q_DUMMY[0]),
                 .O(T2));
    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_MUSER_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]),
```

```
192                  .O(AB7));
193     XNOR2 I_36_35 (.I0(B[5]),
194                    .I1(A[5]),
195                    .O(AB5));
196     XNOR2 I_36_36 (.I0(B[4]),
197                    .I1(A[4]),
198                    .O(AB4));
199     AND4 I_36_41 (.I0(AB3),
200                   .I1(AB2),
201                   .I2(AB1),
202                   .I3(AB0),
203                   .O(AB03));
204     XNOR2 I_36_42 (.I0(B[2]),
205                    .I1(A[2]),
206                    .O(AB2));
207     XNOR2 I_36_43 (.I0(B[3]),
208                    .I1(A[3]),
209                    .O(AB3));
210     XNOR2 I_36_44 (.I0(B[1]),
211                    .I1(A[1]),
212                    .O(AB1));
213     XNOR2 I_36_45 (.I0(B[0]),
214                    .I1(A[0]),
215                    .O(AB0));
216     AND2 I_36_50 (.I0(AB47),
217                   .I1(AB03),
218                   .O(EQ));
219   endmodule
220   `timescale 1ns / 1ps
221
222   module M2_1_MXILINX_dropfifo(D0,
223                                D1,
224                                S0,
225                                O);
226
227     input D0;
228     input D1;
229     input S0;
230     output O;
231
232     wire M0;
233     wire M1;
234
235     AND2B1 I_36_7 (.I0(S0),
236                    .I1(D0),
237                    .O(M0));
238     OR2 I_36_8 (.I0(M1),
239                 .I1(M0),
240                 .O(O));
241     AND2 I_36_9 (.I0(D1),
242                  .I1(S0),
243                  .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,
                               L,
                               CEO,
                               Q,
                               TC);

    input C;
    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),
```

```
                                .CE(OR_CE_L),
                                .CLR(CLR),
                                .D(D[5]),
                                .L(L),
                                .T(T5),
                                .Q(Q_DUMMY[5]));
    // synthesis attribute HU_SET of I_Q5 is "I_Q5_14"
    FTCLEX_MXILINX_dropfifo I_Q6 (.C(C),
                                .CE(OR_CE_L),
                                .CLR(CLR),
                                .D(D[6]),
                                .L(L),
                                .T(T6),
                                .Q(Q_DUMMY[6]));
    // synthesis attribute HU_SET of I_Q6 is "I_Q6_15"
    FTCLEX_MXILINX_dropfifo I_Q7 (.C(C),
                                .CE(OR_CE_L),
                                .CLR(CLR),
                                .D(D[7]),
                                .L(L),
                                .T(T7),
                                .Q(Q_DUMMY[7]));
    // synthesis attribute HU_SET of I_Q7 is "I_Q7_16"
    AND3 I_36_8 (.I0(Q_DUMMY[5]),
                 .I1(Q_DUMMY[4]),
                 .I2(T4),
                 .O(T6));
    AND2 I_36_11 (.I0(Q_DUMMY[4]),
                  .I1(T4),
                  .O(T5));
    VCC I_36_12 (.P(XLXN_1));
    AND2 I_36_19 (.I0(Q_DUMMY[1]),
                  .I1(Q_DUMMY[0]),
                  .O(T2));
    AND3 I_36_21 (.I0(Q_DUMMY[2]),
                  .I1(Q_DUMMY[1]),
                  .I2(Q_DUMMY[0]),
                  .O(T3));
    AND4 I_36_23 (.I0(Q_DUMMY[3]),
                  .I1(Q_DUMMY[2]),
                  .I2(Q_DUMMY[1]),
                  .I3(Q_DUMMY[0]),
                  .O(T4));
    AND4 I_36_25 (.I0(Q_DUMMY[6]),
                  .I1(Q_DUMMY[5]),
                  .I2(Q_DUMMY[4]),
                  .I3(T4),
                  .O(T7));
    AND5 I_36_29 (.I0(Q_DUMMY[7]),
                  .I1(Q_DUMMY[6]),
                  .I2(Q_DUMMY[5]),
                  .I3(Q_DUMMY[4]),
                  .I4(T4),
                  .O(TC_DUMMY));
    AND2 I_36_33 (.I0(CE),
                  .I1(TC_DUMMY),
                  .O(CEO));
    OR2 I_36_49 (.I0(CE),
                 .I1(L),
                 .O(OR_CE_L));
```

```
418  endmodule
419  `timescale 1ns / 1ps
420
421  module FD8CE_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
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),
```

```
                .D(D[7]),
                .Q(Q[7]));
    defparam I_Q7.INIT = 1'b0;
endmodule
`timescale 1ns / 1ps

module dropfifo(clk,
                drop_pkt,
                fiforead,
                fifowrite,
                firstword,
                in_fifo,
                lastword,
                rst,
                out_fifo,
                valid_data);

    input clk;
    input drop_pkt;
    input fiforead;
    input fifowrite;
    input firstword;
    input [71:0] in_fifo;
    input lastword;
    input rst;
   output [71:0] out_fifo;
   output valid_data;

   wire [7:0] raddr;
   wire [7:0] waddr;
   wire XLXN_23;
   wire XLXN_24;
   wire XLXN_26;
   wire XLXN_27;
   wire XLXN_31;
   wire [7:0] XLXN_34;
   wire XLXN_40;
   wire XLXN_41;
   wire XLXN_44;
   wire XLXN_47;
   wire [71:0] XLXN_48;
   wire XLXN_49;

   dual_core_memory_element XLXI_1 (.addra(waddr[7:0]),
                                    .addrb(raddr[7:0]),
                                    .clka(clk),
                                    .clkb(clk),
                                    .dina(XLXN_48[71:0]),
                                    .wea(XLXN_47),
                                    .doutb(out_fifo[71:0]));
   FD XLXI_2 (.C(clk),
              .D(firstword),
              .Q(XLXN_23));
   defparam XLXI_2.INIT = 1'b0;
   FD XLXI_3 (.C(clk),
              .D(lastword),
              .Q(XLXN_24));
```

```
    defparam XLXI_3.INIT = 1'b0;
    FD XLXI_4 (.C(clk),
               .D(fifowrite),
               .Q(XLXN_47));
    defparam XLXI_4.INIT = 1'b0;
    OR2 XLXI_5 (.I0(XLXN_24),
                .I1(XLXN_23),
                .O(XLXN_26));
    FD8CE_MXILINX_dropfifo XLXI_6 (.C(clk),
                                   .CE(XLXN_27),
                                   .CLR(rst),
                                   .D(waddr[7:0]),
                                   .Q(XLXN_34[7:0]));
    // synthesis attribute HU_SET of XLXI_6 is "XLXI_6_17"
    AND2B1 XLXI_7 (.I0(XLXN_31),
                   .I1(XLXN_26),
                   .O(XLXN_27));
    FD XLXI_8 (.C(clk),
               .D(drop_pkt),
               .Q(XLXN_31));
    defparam XLXI_8.INIT = 1'b0;
    CB8CLE_MXILINX_dropfifo XLXI_9 (.C(clk),
                                    .CE(XLXN_47),
                                    .CLR(rst),
                                    .D(XLXN_34[7:0]),
                                    .L(XLXN_31),
                                    .CEO(),
                                    .Q(waddr[7:0]),
                                    .TC());
    // synthesis attribute HU_SET of XLXI_9 is "XLXI_9_18"
    comp8_MUSER_dropfifo XLXI_10 (.A(waddr[7:0]),
                                  .B(raddr[7:0]),
                                  .EQ(XLXN_41));
    comp8_MUSER_dropfifo XLXI_11 (.A(raddr[7:0]),
                                  .B(XLXN_34[7:0]),
                                  .EQ(XLXN_40));
    CB8CE_MXILINX_dropfifo XLXI_12 (.C(clk),
                                    .CE(XLXN_44),
                                    .CLR(rst),
                                    .CEO(),
                                    .Q(raddr[7:0]),
                                    .TC());
    // synthesis attribute HU_SET of XLXI_12 is "XLXI_12_19"
    AND3B2 XLXI_13 (.I0(XLXN_40),
                    .I1(XLXN_41),
                    .I2(fiforead),
                    .O(XLXN_44));
    FDC XLXI_14 (.C(clk),
                 .CLR(rst),
                 .D(XLXN_44),
                 .Q(valid_data));
    defparam XLXI_14.INIT = 1'b0;
    reg9B XLXI_15 (.ce(XLXN_49),
                   .clk(clk),
                   .clr(rst),
                   .d(in_fifo[71:0]),
                   .q(XLXN_48[71:0]));
    VCC XLXI_16 (.P(XLXN_49));
```

```
endmodule

```