

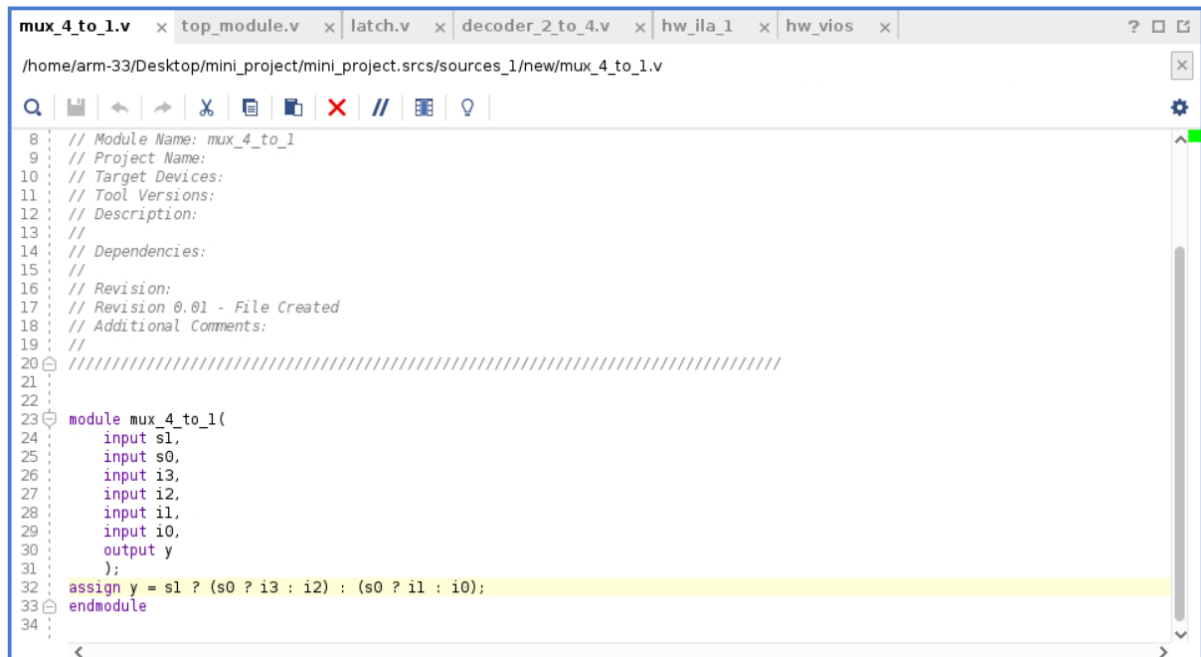
Mini Project

Submitted by:

Nivesh S

Verilog Codes

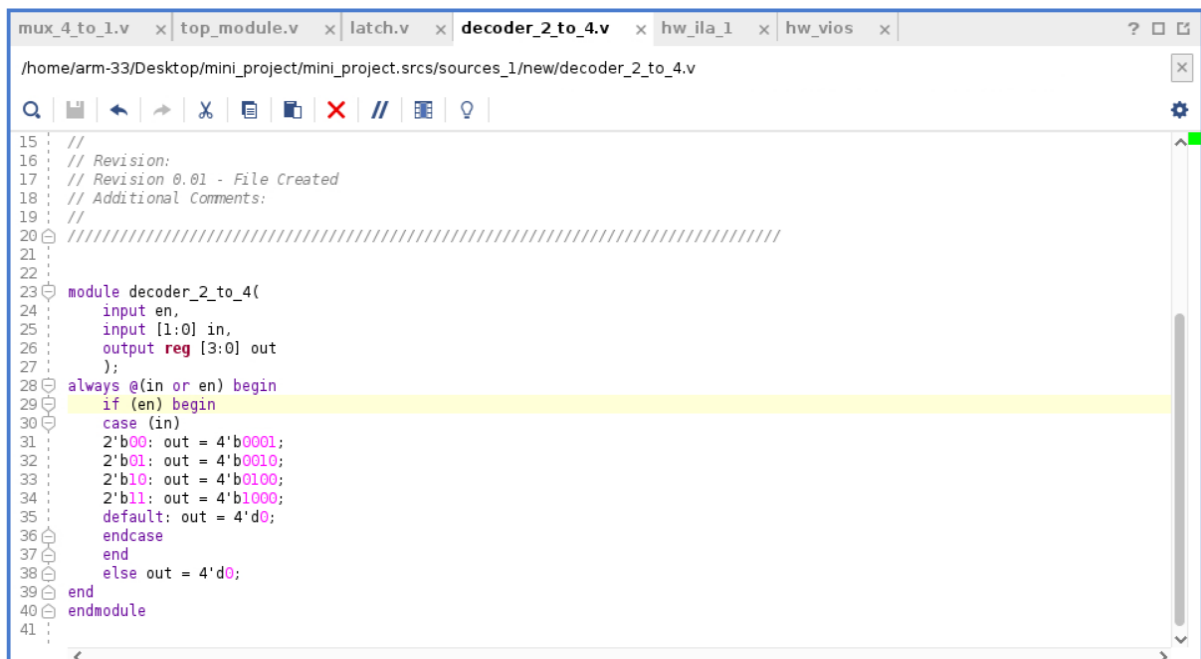
4 to 1 MUX using dataflow modelling



The screenshot shows a Verilog code editor with the file name `mux_4_to_1.v`. The code defines a module `mux_4_to_1` with four inputs (`s1`, `s0`, `i3`, `i2`), two inputs (`i1`, `i0`), and one output (`y`). The implementation uses dataflow modelling with an `assign` statement to select the output based on the select inputs `s1` and `s0`.

```
8 // Module Name: mux_4_to_1
9 // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module mux_4_to_1(
24     input s1,
25     input s0,
26     input i3,
27     input i2,
28     input i1,
29     input i0,
30     output y
31 );
32 assign y = s1 ? (s0 ? i3 : i2) : (s0 ? i1 : i0);
33 endmodule
34
```

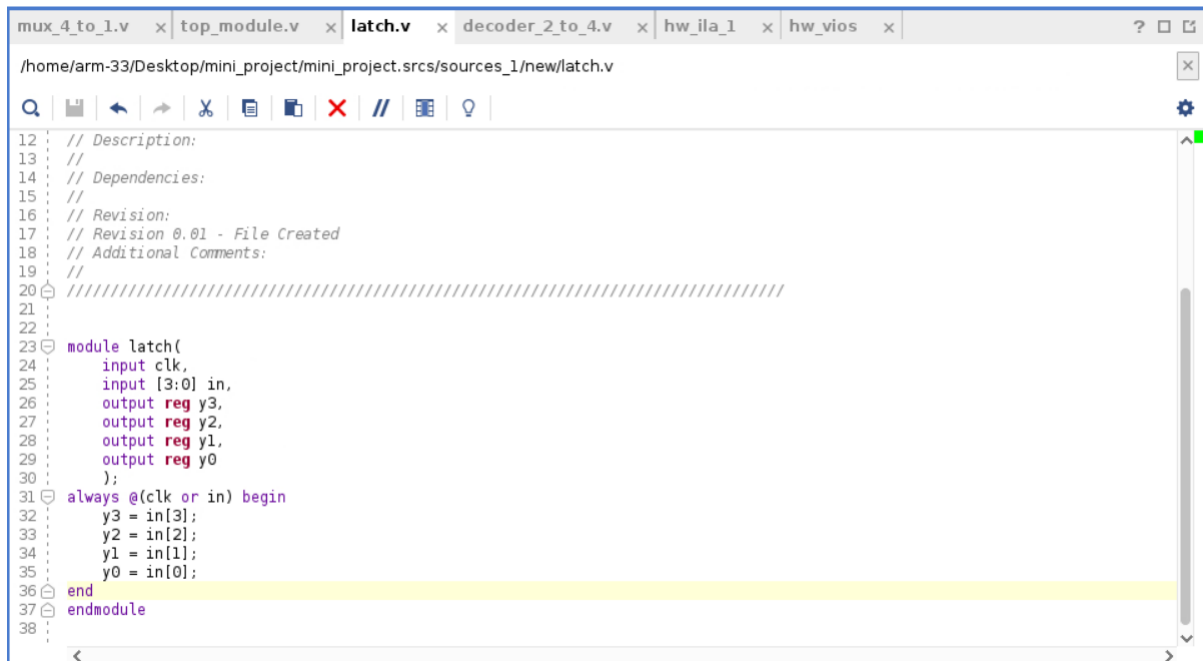
Decoder using behavioural modelling



The screenshot shows a Verilog code editor with the file name `decoder_2_to_4.v`. The code defines a module `decoder_2_to_4` with two inputs (`en`, `in`) and a 4-bit output register (`out`). The implementation uses behavioural modelling with an `always` block and a case statement to decode the input `in` based on the enable input `en`.

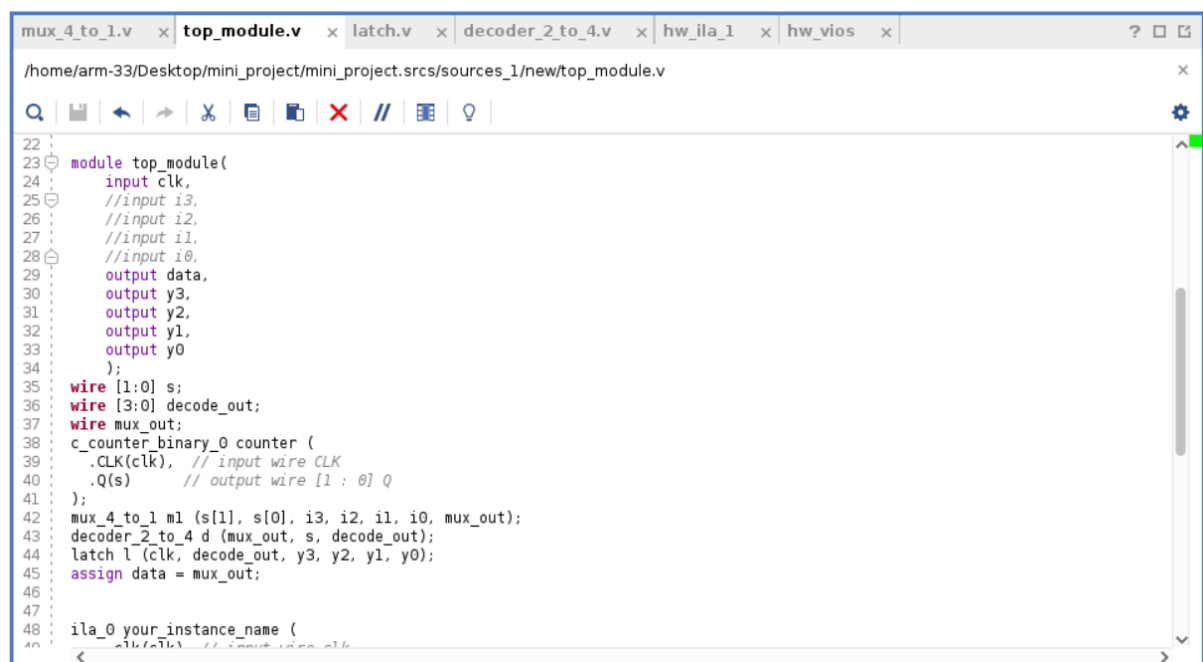
```
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module decoder_2_to_4(
24     input en,
25     input [1:0] in,
26     output reg [3:0] out
27 );
28 always @(in or en) begin
29     if (en) begin
30         case (in)
31             2'b00: out = 4'b0001;
32             2'b01: out = 4'b0010;
33             2'b10: out = 4'b0100;
34             2'b11: out = 4'b1000;
35             default: out = 4'd0;
36         endcase
37     end
38     else out = 4'd0;
39 end
40 endmodule
41
```

Latch using behavioural modelling



```
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module latch(
24     input clk,
25     input [3:0] in,
26     output reg y3,
27     output reg y2,
28     output reg y1,
29     output reg y0
30 );
31 always @(clk or in) begin
32     y3 = in[3];
33     y2 = in[2];
34     y1 = in[1];
35     y0 = in[0];
36 end
37 endmodule
38
```

Top level module along with counter IP Core

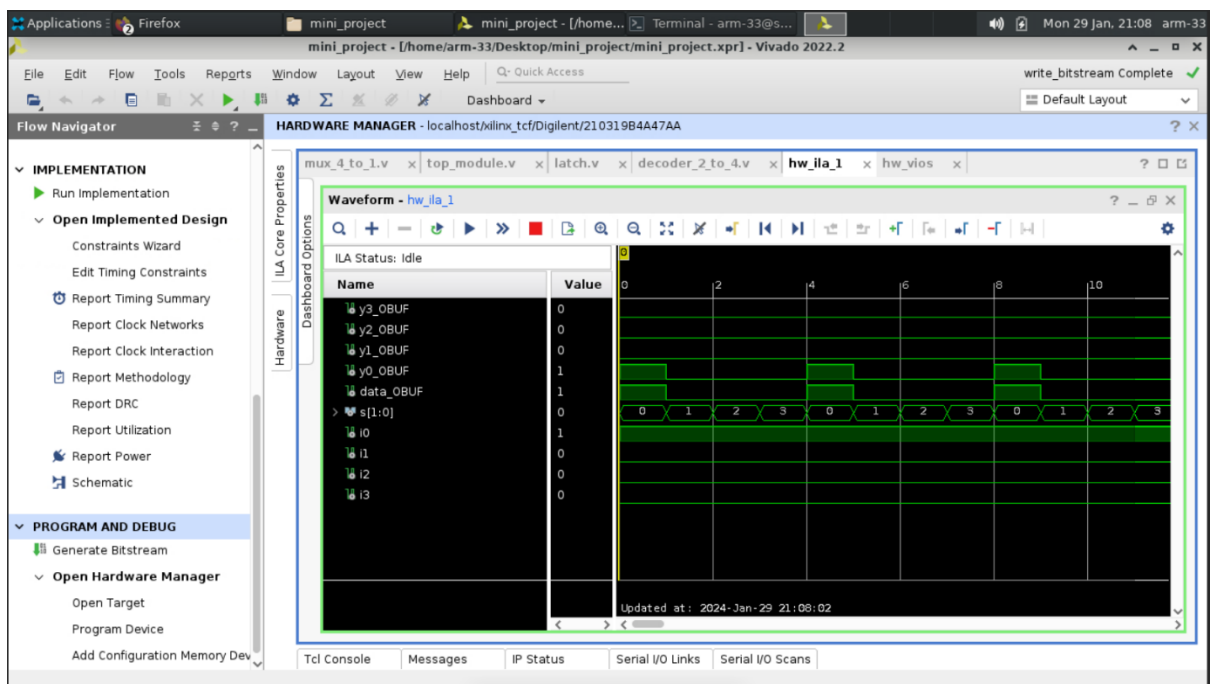


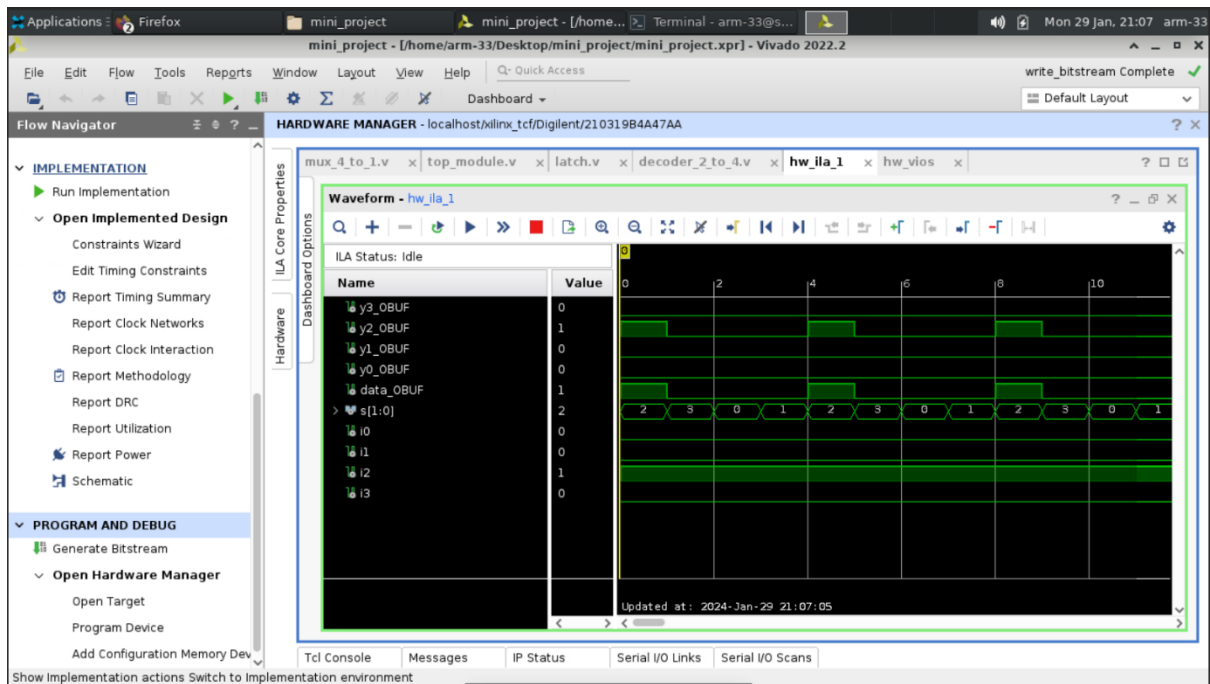
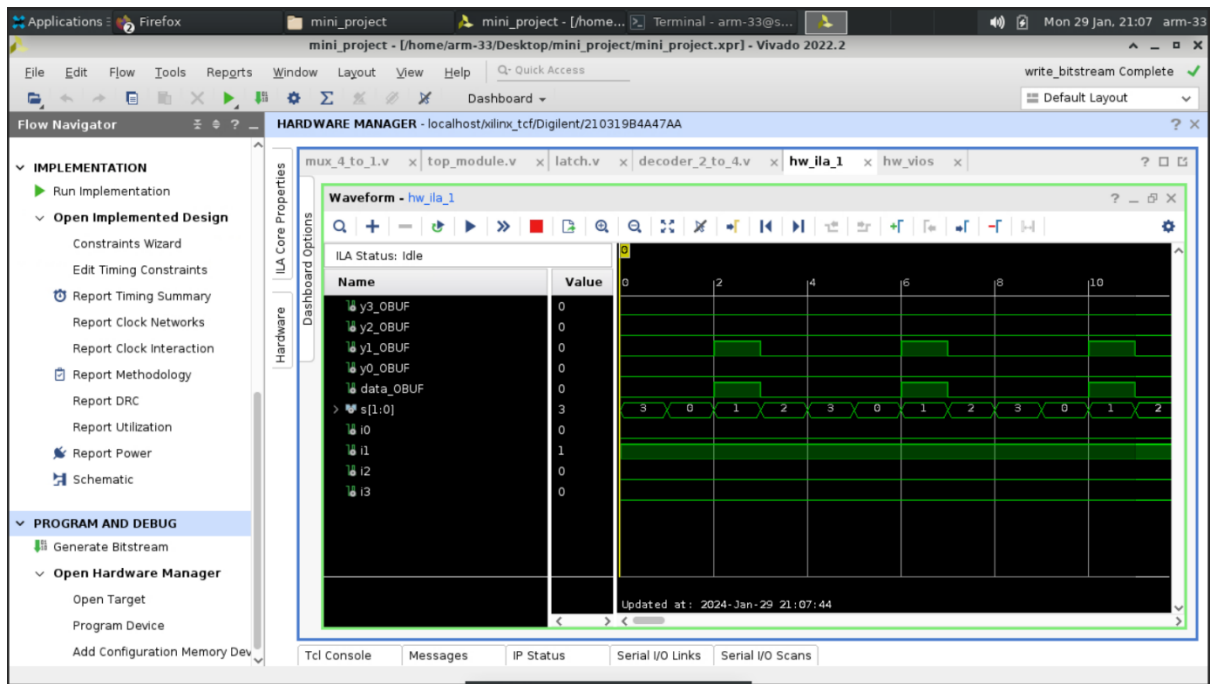
```
22
23 module top_module(
24     input clk,
25     //input i3,
26     //input i2,
27     //input i1,
28     //input i0,
29     output data,
30     output y3,
31     output y2,
32     output y1,
33     output y0
34 );
35 wire [1:0] s;
36 wire [3:0] decode_out;
37 wire mux_out;
38 c_counter_binary_0 counter (
39     .CLK(clk), // input wire CLK
40     .Q(s) // output wire [1 : 0] Q
41 );
42 mux_4_to_1 m1 (s[1], s[0], i3, i2, i1, i0, mux_out);
43 decoder_2_to_4 d (mux_out, s, decode_out);
44 latch l (clk, decode_out, y3, y2, y1, y0);
45 assign data = mux_out;
46
47
48 ila_0 your_instance_name (
49     clk, // input wire clk
50
```

```
mux_4_to_1.v x top_module.v x latch.v x decoder_2_to_4.v x hw_ila_1 x hw_vios x
/home/arm-33/Desktop/mini_project/mini_project.srscs/sources_1/new/top_module.v

49: .clk(clk), // input wire clk
50:
51:
52:     .probe0(y3), // input wire [0:0] probe0
53:     .probe1(y2), // input wire [0:0] probe1
54:     .probe2(y1), // input wire [0:0] probe2
55:     .probe3(y0), // input wire [0:0] probe3
56:     .probe4(data), // input wire [0:0] probe4
57:     .probe5(s), // input wire [1:0] probe5
58:     .probe6(i3), // input wire [0:0] probe6
59:     .probe7(i2), // input wire [0:0] probe7
60:     .probe8(i1), // input wire [0:0] probe8
61:     .probe9(i0) // input wire [0:0] probe9
62: );
63:
64: vio_0 vio (
65:     .clk(clk), // input wire clk
66:     .probe_out0(i3), // output wire [0 : 0] probe_out0
67:     .probe_out1(i2), // output wire [0 : 0] probe_out1
68:     .probe_out2(i1), // output wire [0 : 0] probe_out2
69:     .probe_out3(i0) // output wire [0 : 0] probe_out3
70: );
71:
72:
73:
74: endmodule
75:
```

Waveforms simulated and observed using VIO and ILA core IP





Applications: Firefox mini_project mini_project - [home... Terminal - arm-33@... Mon 29 Jan, 21:06 arm-33

mini_project - [home/arm-33/Desktop/mini_project/mini_project.xpr] - Vivado 2022.2

File Edit Flow Tools Reports Window Layout View Help Q: Quick Access write_bitstream Complete ✓

Flow Navigator HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/21031984447AA

IMPLEMENTATION

- Run Implementation
- Open Implemented Design
 - Constraints Wizard
 - Edit Timing Constraints
 - Report Timing Summary
 - Report Clock Networks
 - Report Clock Interaction
 - Report Methodology
 - Report DRC
 - Report Utilization
 - Report Power
 - Schematic
- PROGRAM AND DEBUG
 - Generate Bitstream
 - Open Hardware Manager
 - Open Target
 - Program Device
 - Add Configuration Memory Dev

mux_d4_to_1.v x top_module.v x latch.v x decoder_2_to_4.v x hw_ila_1 x hw_vios x

Waveform - hw_ila_1

ILA Status: Idle

Name	Value
y3_OBUF	0
y2_OBUF	0
y1_OBUF	0
y0_OBUF	0
data_OBUF	0
s[1:0]	0 0
i0	0
i1	0
i2	0
i3	1

Updated at: 2024-Jan-29 20:50:29

Tcl Console Messages IP Status Serial I/O Links Serial I/O Scans