VHDL for Combinational Circuits and Storage Elements

COE 328-022 Pei Yang

Nov 1 2022

Introduction

The objective of this lab was to construct combinational circuits and circuits with basic storage, using VHDL.

Experiment

The first task was to implement a 4:1 mux using 2 3:1 muxes, and a 3:8 decoder with 2 2:4 decoders. To do this, blocks were created using the mux and decode VHDL codes explored in the pre-lab.

**Mux**

To create the output of a 4:1, 2 layers of 3:1 muxes are used. The output of each 3:1 mux in layer 1 are connected to the input of the 3:1 mux in layer 2, with layer 1 muxes sharing an s input. When s of layer 2 (s2) = 0, the mux outputs the first input. In this case, the first input leads to the first mux of layer 1. When s2 =1, the mux will output the second input, or the second mux of layer 1. The same logic is applied to the muxes in layer 1. For example, in mux1 of layer 1, if the s of layer 1 (s1) = 0, the output will match the first input, w0.

**Decode**

To create the 3:8 decoder with 2 2:4 decodes, 2 2:4 decoders are used in tangent with a NOT and 2 AND gates. The inputs of each gate are connected to the matching inputs of the same name. The En input of decode1 = !w2 AND En. The En input of decode2 = w2 AND En. The w2 value controls which decode block turns on at a time. When w2=0, the decode1 outputs its values. When w2=1, the decode1 shuts off and decode2 outputs its values. The output of each decode follows the VHDL code precisely, using En, w1, w0 as the inputs (in that order), and y0, y1, y2, y3 as the outputs.

Encode VHDL and edited johns counter code have been attached below. Johns code was run outside of the allotted lab time.

Conclusion

While conducting the decode part of the lab, there was confusion trying to understand the waveform, which was created due to the change in order of the w1, w0 on the block diagram compared to the waveform. On the waveform, the inputs are w0, w1, but on the diagram they are w1, w0.

**muxModified**

|  |
| --- |
| LIBRARY ieee; USE ieee.std\_logic\_1164.all;      ENTITY mux3to1 IS         PORT ( w0, w1, s: IN STD\_LOGIC ;         f : OUT STD\_LOGIC ) ; END mux3to1 ;  ARCHITECTURE Behavior OF mux3to1 IS BEGIN     WITH s SELECT         f <= w0 WHEN '0',               w1 WHEN OTHERS ; END Behavior; |

Chart, box and whisker chart

Description automatically generated

A picture containing table

Description automatically generated

**decodModified**

|  |
| --- |
| LIBRARY ieee ; USE ieee.std\_logic\_1164.all ;  ENTITY decode2to4 IS     PORT ( w : IN STD\_LOGIC\_VECTOR(1 DOWNTO 0) ;             En : IN STD\_LOGIC ;              y : OUT STD\_LOGIC\_VECTOR(0 TO 3) ) ; END decode2to4 ;  ARCHITECTURE Behavior OF decode2to4 IS     SIGNAL Enw : STD\_LOGIC\_VECTOR(2 DOWNTO 0) ; BEGIN     Enw <= En & w ;     WITH Enw SELECT     y <= "1000" WHEN "100",             "0100" WHEN "101",             "0010" WHEN "110",             "0001" WHEN "111",             "0000" WHEN OTHERS ; END Behavior ; |

A screenshot of a computer

Description automatically generated  
Graphical user interface, application, table

Description automatically generated

**Encod:**

LIBRARY ieee ;

USE ieee.std\_logic\_1164.all ;

ENTITY encod IS

PORT ( w : IN STD\_LOGIC\_VECTOR(3 DOWNTO 0) ;

y : OUT STD\_LOGIC\_VECTOR(1 DOWNTO 0) ;

z : OUT STD\_LOGIC ) ;

END encod ;

ARCHITECTURE Behavior OF encod IS

BEGIN

PROCESS ( w )

BEGIN

y <= "00" ;

IF w(1) = '1' THEN y <= "01" ; END IF ;

IF w(2) = '1' THEN y <= "10" ; END IF ;

IF w(3) = '1' THEN y <= "11" ; END IF ;

z <= '1' ;

IF w = "0000" THEN z <= '0' ; END IF ;

END PROCESS ;

END Behavior ;

**Johnsons code:**

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

ENTITY johns IS

PORT (Clrn, E, Clkn : IN STD\_LOGIC; --clrn is your reset button

STUDENT\_ID : out std\_logic\_vector(3 downto 0);

Q : OUT STD\_LOGIC\_VECTOR (0 TO 2));

END johns;

ARCHITECTURE Behavior OF johns IS

signal Qreg : STD\_LOGIC\_VECTOR (0 TO 2);

BEGIN

PROCESS (Clrn, Clkn)

BEGIN

IF Clrn = '0' THEN

Qreg <= "000";

ELSIF (Clkn'EVENT AND Clkn = '0') THEN

IF E = '1' THEN

Qreg(0) <= NOT(Qreg(2));

Qreg(1) <= Qreg(0);

Qreg(2) <= Qreg(1);

ELSE

Qreg <= Qreg;

END IF;

END IF;

-- STUDENT\_ID variable represents the last 6 digits of your student ID

--hence d4 is the fourth digit of your

--student ID in four bits, d5 is the fifth and so on. For example, for

--Student ID 500435429,

--d4 is 0100, d5 is 0011 and so on

--my student number is 500 435429

CASE Qreg IS

WHEN "000" =>

STUDENT\_ID <= "0001"; --d1

WHEN "100" =>

STUDENT\_ID <= "0011"; --d2

WHEN "110" =>

STUDENT\_ID <= "0111";--d3

WHEN "111" =>

STUDENT\_ID <= "0110";--d4

WHEN "011" =>

STUDENT\_ID <= "0101";--d5

WHEN "001" =>

STUDENT\_ID <= "1001";--d6

WHEN OTHERS => STUDENT\_ID <= "----";--error

END CASE;

END PROCESS;

Q <= Qreg;

END Behavior;

Graphical user interface, application, table, Excel

Description automatically generated