1. What is a decoder? What is the relationship between its inputs and outputs? Derive the function(s) that maps inputs and outputs for a 2-to-4 decoder. Write a VHDL code that implements the 2-to-4 decoder using case statement.
   1. A decoder is a popular combinational logic. It converts an input binary number into one high output.
   2. If there are n-inputs, there will be 2n outputs.
   3. Truth Table

|  |  |
| --- | --- |
| a | b |
| i1 i0 | d0 d1 d­­2 d3 |
| 00 | 1000 |
| 01 | 0100 |
| 10 | 0010 |
| 11 | 0001 |

* 1. Functions: and and and
  2. VHDL code with case statement

LIBRARY ieee ;

USE ieee.std\_logic\_1164.all ;

ENTITY dec2to4 IS

PORT (a : IN STD\_LOGIC\_VECTOR(1 DOWNTO 0) ;

b : OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0) ) ;

END dec2to4 ;

ARCHITECTURE Behavior OF dec2to4 IS

BEGIN

PROCESS(a)

BEGIN

CASE a IS

WHEN “00” => b <= “1000”;

WHEN “01” => b <= “0100”;

WHEN “10” => b <= “0010”;

WHEN “11” => b <= “0001”;

END CASE;

END PROCESS;

END Behavior ;

1. Implement the function using a 4-to-16 decoder.
   1. Truth Table

|  |  |
| --- | --- |
| ABCD |  |
| 0000 | 0 |
| 0001 | 0 |
| 0010 | 0 |
| 0011 | 0 |
| 0100 | 1 |
| 0101 | 1 |
| 0110 | 1 |
| 0111 | 1 |
| 1000 | 1 |
| 1001 | 1 |
| 1010 | 1 |
| 1011 | 1 |
| 1100 | 1 |
| 1101 | 1 |
| 1110 | 1 |
| 1111 | 0 |

* 1. Block diagram



1. What is an encoder? What is the relationship between its inputs and outputs? Derive the function(s) that maps inputs and outputs for a 4-to-2 encoder. Write a VHDL code that implements the 4-to-2 encoder using case statement.
   1. An encoder is used to reduce the number of bits required to represent given information. Exactly one input is high.
   2. When there are 2n inputs, there are n outputs.
   3. Truth Table

|  |  |
| --- | --- |
| a | b |
| w3 w2 w1 w0 | y1 y0 |
| 0001 | 00 |
| 0010 | 01 |
| 0100 | 10 |
| 1000 | 11 |

* 1. Functions:
  2. VHDL code with case statement

LIBRARY ieee ;

USE ieee.std\_logic\_1164.all ;

ENTITY enc4to2 IS

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

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

END enc4to2 ;

ARCHITECTURE Behavior OF enc4to2 IS

BEGIN

PROCESS(a)

BEGIN

CASE a IS

WHEN “0001” => b <= “00”;

WHEN “0010” => b <= “01”;

WHEN “0100” => b <= “10”;

WHEN “1000” => b <= “11”;

END CASE;

END PROCESS;

END Behavior ;

1. What is a multiplexer? What is the relationship between its inputs and outputs? Derive the function(s) that maps inputs and outputs for a 4-to-1 multiplexer. Write a VHDL code that implements the 4-to-1 multiplexer using case statement.
   1. A multiplexer is another popular combinational building block. It routes one of its N data inputs to its one output, based on the binary value of select inputs.
   2. With N inputs, there are select lines. There is always one output.
   3. Truth Table

|  |  |  |
| --- | --- | --- |
| ABCD | F |  |
| 0000 | 0 | F=D |
| 0001 | 1 | F=D |
| 0010 | 0 | F=D |
| 0011 | 1 | F=D |
| 0100 | 1 | F= |
| 0101 | 0 | F= |
| 0110 | 0 | F=0 |
| 0111 | 0 | F=0 |
| 1000 | 0 | F=0 |
| 1001 | 0 | F=0 |
| 1010 | 0 | F=D |
| 1011 | 1 | F=D |
| 1100 | 1 | F=1 |
| 1101 | 1 | F=1 |
| 1110 | 1 | F=1 |
| 1111 | 1 | F=1 |

* 1. Functions: F=D and F= and F=0 and F=1
  2. VHDL code with Case statement

LIBRARY ieee ;

USE ieee.std\_logic\_1164.all ;

ENTITY mux4to1 IS

PORT (a, b, c, d : IN STD\_LOGIC;

selection: IN STD\_LOGIC\_VECTOR(1 DOWNTO 0);

f : OUT STD\_LOGIC) ;

END mux4to1 ;

ARCHITECTURE Behavior OF mux4to1 IS

BEGIN

PROCESS(a, b, c, d, selection)

BEGIN

CASE selection IS

WHEN “00” => f <= a;

WHEN “01” => f <= b;

WHEN “10” => f <= c;

WHEN OTHERS => f <= d;

END CASE;

END PROCESS;

END Behavior ;

1. Implement the function using an 8-to-1 multiplexer.
   1. Truth Table

|  |  |  |
| --- | --- | --- |
| ABCD |  |  |
| 0000 | 1 | F= |
| 0001 | 0 |
| 0010 | 0 | F=0 |
| 0011 | 0 |
| 0100 | 0 | F=D |
| 0101 | 1 |
| 0110 | 0 | F=D |
| 0111 | 1 |
| 1000 | 0 | F=D |
| 1001 | 1 |
| 1010 | 1 | F=1 |
| 1011 | 1 |
| 1100 | 0 | F=D |
| 1101 | 1 |
| 1110 | 0 | F=D |
| 1111 | 1 |

* 1. Use ABC as select lines. Function output varies between 1, 0, D, and .
  2. Block diagram



1. What is a demultiplexer? What is the relationship between its inputs and outputs? Derive the function(s) that maps inputs and outputs for a 1-to-4 demultiplexer.
   1. A demultiplexer routes a single input (if the enable is used as the input) to one of many outputs. Functions the same as a decoder.
   2. With N outputs, there are select lines. There is always one input.
   3. Truth Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Data input | Select Inputs | | Outputs | | | |
| D | S1 | S0 | Y3 | Y2 | Y1 | Y0 |
| D | 0 | 0 | 0 | 0 | 0 | D |
| D | 0 | 1 | 0 | 0 | D | 0 |
| D | 1 | 0 | 0 | D | 0 | 0 |
| D | 1 | 1 | D | 0 | 0 | 0 |

* 1. Functions: and and and

1. Derive the truth table for half-subtractor and full-subtractor. Derive the input-output function from the truth table.
   1. Truth Table half-subtractor

|  |  |
| --- | --- |
| Inputs | Outputs |
| A B | Di(Difference) B0(Borrow) |
| 0 0 | 0 0 |
| 0 1 | 1 1 |
| 1 0 | 1 0 |
| 1 1 | 0 0 |

* 1. Half-subtractor input-output functions: and
  2. Truth Table full-subtractor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | BORROW | DIFFERENCE |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

* 1. Full-subtractor input-output functions: and

1. What is a multifunction register? Using block diagrams, implement a 3-bit multifunction register using D flip-flops and multiplexers that does parallel load, inverted parallel load, shift right, and shift left operations.
   1. A multifunction register takes in parallel and serial input and outputs the operation done. Inputs must be specified to select which function will be used.
   2. There will be 8 flip flops, but 3 unused.
   3. Truth Table

|  |  |  |  |
| --- | --- | --- | --- |
| S2 | S1 | S0 | Operation |
| 0 | 0 | 0 | Hold |
| 0 | 0 | 1 | Shift right |
| 0 | 1 | 0 | Shift left |
| 0 | 1 | 1 | Parallel load |
| 1 | 0 | 0 | Inverted parallel load |
| 1 | 0 | 1 |  |
| 1 | 1 | 0 |  |
| 1 | 1 | 1 |  |

* 1. Block diagram



1. Write a behavioral VHDL code that implements a 4-bit multifunction register that does parallel load, shift right, shift left, and rotate right operations. Write a VHDL test bench to test the operations with at least one test vector per operation.
   1. VHDL code

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

ENTITY multfr IS

PORT(Clock, shr\_in, shl\_in : IN STD\_LOGIC;

s : IN STD\_LOGIC\_VECTOR(1 DOWNTO 0);

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

Q : OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0));

END multfr;

ARCHITECTURE behav OF multfr IS

SIGNAL tmp: std\_logic\_vector(3 downto 0); -- 4-bit flip-flop

BEGIN

PROCESS(Clock)

BEGIN

IF (clock’ event and clock = ‘1’) THEN

CASE s IS

when “00” => -- do nothing;

when “01” => tmp <= I;

when “10” =>

tmp(0) <= tmp(1);

tmp(1) <= tmp(2);

tmp(2) <= tmp(3);

tmp(3) <= shr\_in;

when “11” =>

tmp(0) <= shl\_in;

tmp(1) <= tmp(0);

tmp(2) <= tmp(1);

tmp(3) <= tmp(2);

END CASE;

END IF;

END PROCESS;

Q <= tmp;

END behav;

* 1. VHDL test bench

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.numeric\_std.ALL;

LIBRARY UNISIM;

USE UNISIM.Vcomponents.ALL;

ENTITY problem9\_tb IS

END problem9\_tb;

ARCHITECTURE behavioral OF problem9\_tb IS

COMPONENT f2

PORT( a, b, c : IN STD\_LOGIC;

f2 : OUT STD\_LOGIC);

END COMPONENT;

SIGNAL a, b, c, f2 : STD\_LOGIC;

BEGIN

UUT: f2 PORT MAP(

a=>a,

b => b,

c => c,

f2 => f2);

-- \*\*\* Test Bench - User Defined Section \*\*\*

tb : PROCESS

BEGIN

a<=’0’; b=’0’; c=’0’;

wait for 10 ms;

a<=’0’; b=’0’; c=’1’;

wait for 10 ms;

a<=’0’; b=’1’; c=’0’;

wait for 10 ms;

a<=’0’; b=’1’; c=’1’;

wait for 10 ms;

a<=’1’; b=’0’; c=’0’;

wait for 10 ms;

a<=’1’; b=’0’; c=’1’;

wait for 10 ms;

a<=’1’; b=’1’; c=’0’;

wait for 10 ms;

a<=’1’; b=’1’; c=’1’;

wait for 10 ms;

WAIT; -- will wait forever

END PROCESS;

-- \*\*\* End Test Bench - User Defined Section \*\*\*

END behavioral;

1. What is a barrel shifter? Demonstrate how left shift by 9 positions is done using a 12- bit barrel shifter?
   1. A barrel shifter is a shifter that can shift by any amount.
   2. A 12-bit barrel shifter can shift left by 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, or 11 positions
   3. Chain four shifters: 8, 4, 2, and 1
   4. Can achieve any shift of 0 to 11 by enabling the correct combination of those four shifters, because the shifts should sum to the desired amount
   5. WXYZ = 1001 will do a shift left by 9 because it uses the 8 shifter and the 1 shifter.

|  |  |  |
| --- | --- | --- |
| Chain carries the 12 bits to next shifter: 0000 0011 0100 0111 | | |
| W 🡪1 | Sh << 8 in | 🡨 0 |
| Chain carries the 12 bits to next shifter: 0100 0111 0000 0000 (by 8) | | |
| X 🡪0 | Sh << 4 in | 🡨 0 |
| Chain carries the 12 bits to next shifter: 0100 0111 0000 0000 | | |
| Y 🡪0 | Sh << 2 in | 🡨 0 |
| Chain carries the 12 bits to next shifter: 0100 0111 0000 0000 | | |
| Z 🡪1 | Sh << 1 in | 🡨 0 |
| Chain carries the 12 bits to next shifter: 1000 1110 0000 0000 (by 1) | | |
| Result: shift by 9: 1000 1110 0000 0000 | | |

1. Differentiate (at least two differences) between a serial and an array multiplier.
   1. Serial Multiplier
      1. Power used is minimal
      2. Area used is minimal
      3. Delay is longer
   2. Array Multiplier
      1. An efficient layout of a combinational multiplier
      2. Uses n-1 adders, eliminates registers
      3. May be pipelined to decrease clock period at the expense of latency
      4. Consumes a high amount of power
      5. Takes up a lot of area