Experiment 0 : Combinational Circuits - 1

Mayur Ware, 19D070070
EE-214, WEL, IIT Bombay

January 28, 2021

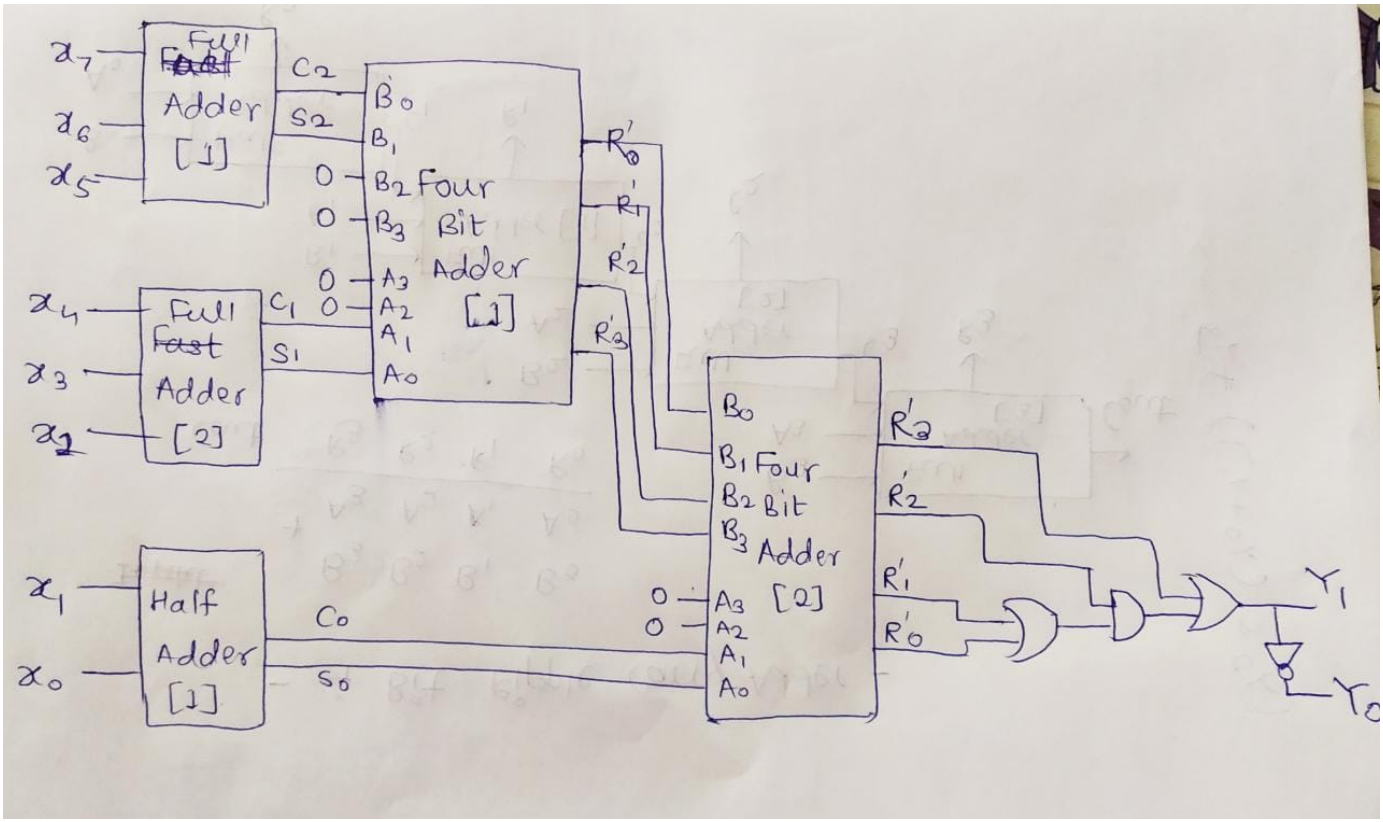
Overview of the experiment

- The purpose of the experiment is to make a digital device which implements the given combinational Circuit.
  Combinational Circuit : The circuit has 8 inputs x7, x6, x5, x4,.. x0 and two outputs y1, y0. The output bit y1 is 1 if and only if the number ofinput bits that are 1 is greater than the number of input bits that are 0. The output bit y0 is the complement of y1.
- I used Full_Adder for the calculation of sum and carry in the input for [x7, x6, x5] and [x4, x3, x2] set of bits and Half_Adder for [x1, x0]. Then I used Four_Bit_Adder to find the exact No. of 1s in the input.
- The report contains a handmade diagram which describes my approach, some important extracts of VHDL code and the output waveforms.

Approach to the experiment

I used 2 Full Adders and 1 Half Adder in the design which give the sum and carry of the given sets of input bits. This gives output in the form of carry bits C2, C1 and C0 and sum bits S2, S1 and S0. Then I Added these outputs in the form of 0 0 C2 S2 + 0 0 C1 S1 + 0 0 C0 S0 with the help of Four_Bit_Adder which gives output R'3 R'2 R'1 R'0 which is the exact No. of 1s in the input bits.
Now, when the No. of 1s will exceed 4, it will be greater than the No. of 0s, then Y1 will be 1. This implies, for 0101 0110 0111 1000, Y1 will be 1. By using a K-MAP and setting values exceeding 1000 as Don't Cares. We get the following expression for Y1.

Y1 = R'3 + (R'2 & (R'1 + R'0))

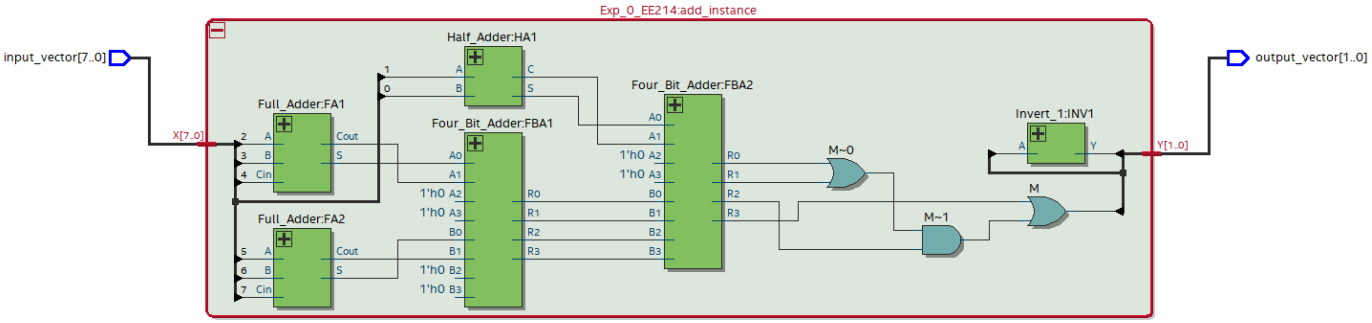


Design document and VHDL code if relevant

Full_Adder :

```
entity Exp_0_EE214 is
port (X : in std_logic_vector(7 downto 0);
Y : out std_logic_vector(1 downto 0));
end entity Exp_0_EE214;
architecture Struct of Exp_0_EE214 is
signal C2, C1, C0, S2, S1, S0, M: std_logic;
signal F4, F3, F2 : std_logic_vector(3 downto 0);
signal F1, F0 : std_logic_vector(4 downto 0);
begin
HA1: Half_Adder port map (A => X(1), B => X(0), S => S0, C => C0);
FA1: Full_Adder port map (A => X(2) , B => X(3), Cin => X(4), S => S1, Cout => C1);
FA2: Full_Adder port map (A => X(5) , B => X(6), Cin => X(7), S => S2, Cout => C2);
F4(3) <= '0';
F4(2) <= '0';
F4(1) <= C2;
F4(0) <= S2;
F3(3) <= '0';
F3(2) <= '0';
F3(1) <= C1;
F3(0) <= S1;
F2(3) <= '0';
F2(2) <= '0';
F2(1) <= C0;
F2(0) <= S0;          – Assigning Appropriate bits to the Four_Bit_Adders
FBA1: Four_Bit_Adder
port map (B3 => F4(3), A3 => F3(3), B2 => F4(2), A2 => F3(2), B1 => F4(1), A1 => F3(1), B0 => F4(0), A0 =>
F3(0), R3 => F1(3), R2 => F1(2), R1 => F1(1), R0 => F1(0), Cout => F1(4));
FBA2: Four_Bit_Adder
port map (B3 => F1(3), A3 => F2(3), B2 => F1(2), A2 => F2(2), B1 => F1(1), A1 => F2(1), B0 => F1(0), A0 =>
F2(0), R3 => F0(3), R2 => F0(2), R1 => F0(1), R0 => F0(0), Cout => F0(4));
M <= F0(3) or (F0(2) and (F0(0) or F0(1)));          –Creating Signal M
Y(1) <= M;                                            – Y1 = M
INV1: Invert_1 port map ( A => M, Y => Y(0));          – Y0 =  Y1
end Struct;
```

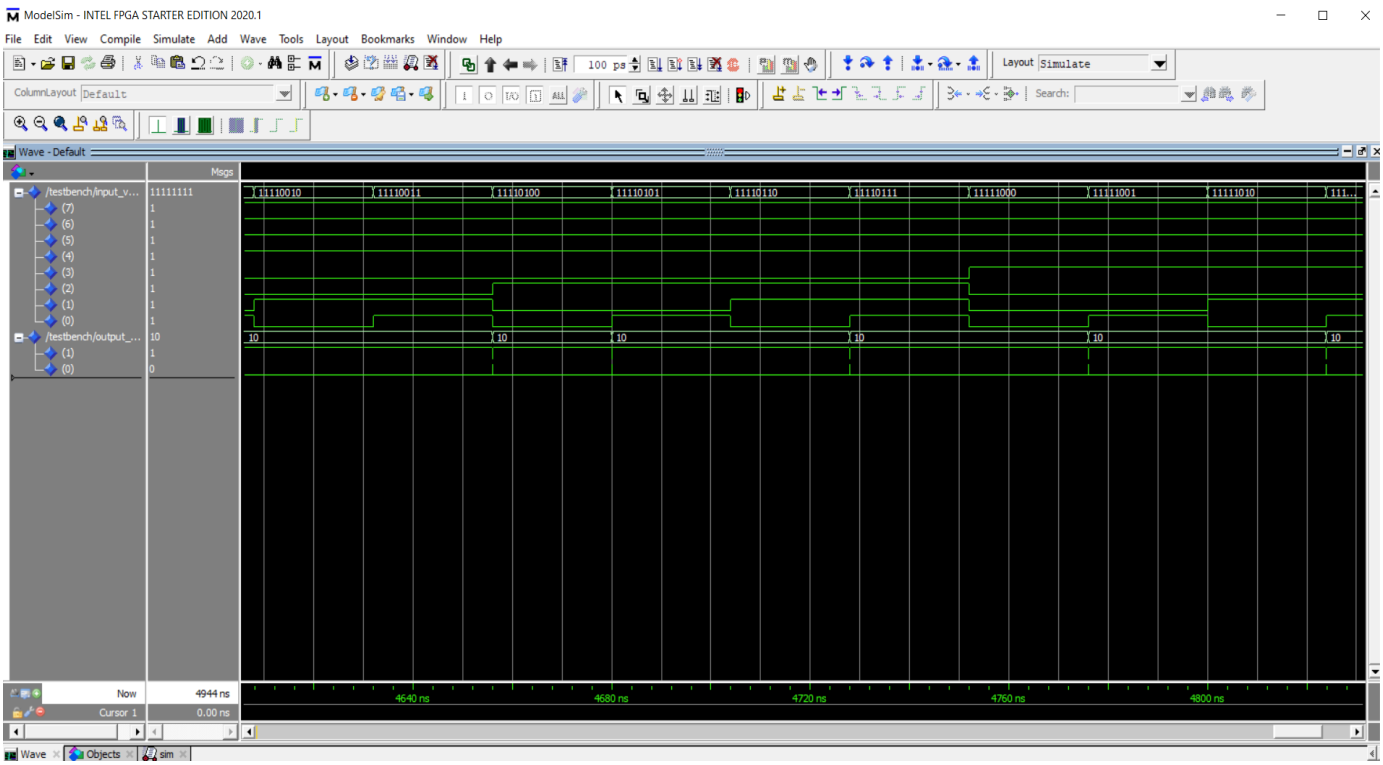
RTL View



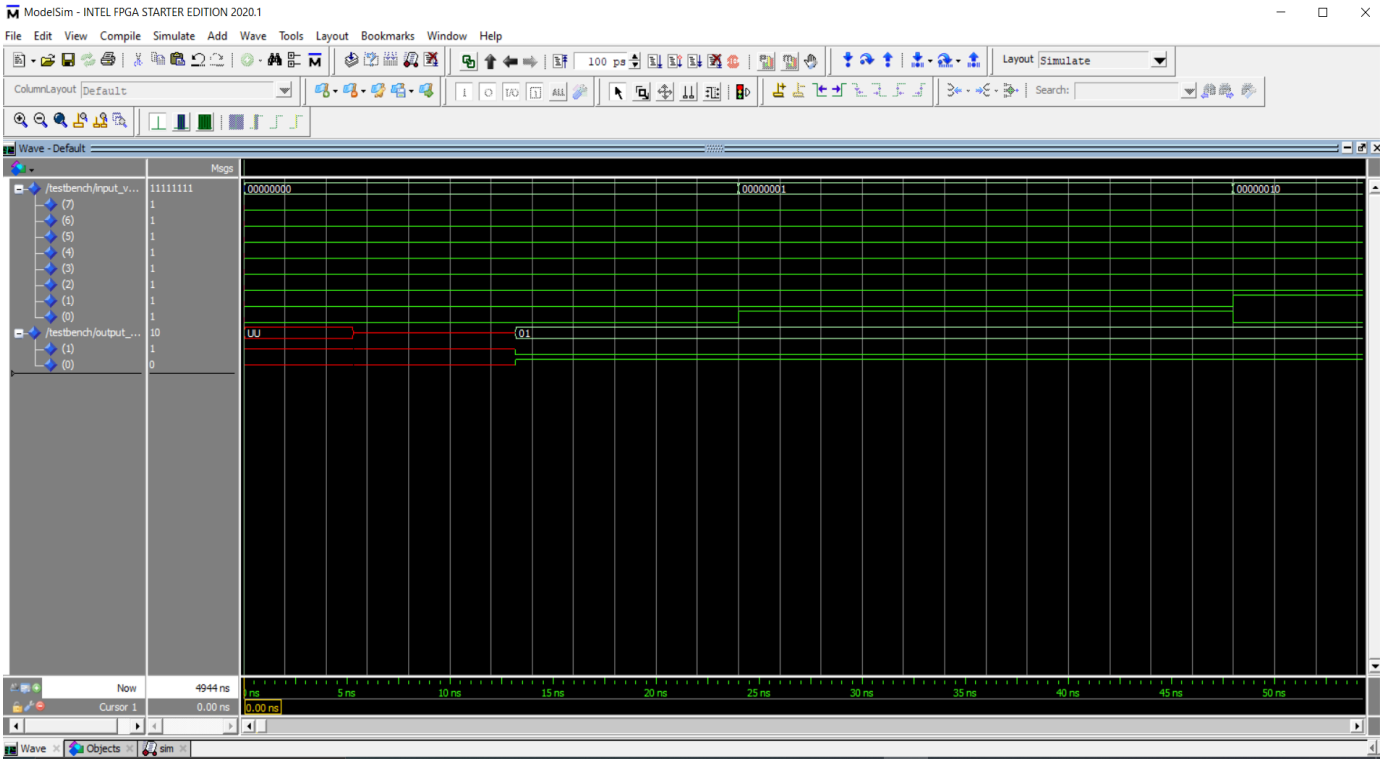
DUT Input/Output Format

```
X : in std_logic_vector(7 downto 0);          –Input Bit Vector
Y : out std_logic_vector(1 downto 0));        –Output Bit Vector
11101111      10      11                      –Input Bit Vector, Output Bit Vector, Mask Bits
```

RTL Simulation



Gate-level Simulation



Krypton board*

Map the logic circuit to the Krypton board and attach the images of the pin assignment and output observed on the board (switches/LEDs).

Observations*

You must summarize your observations, either in words, using figures and/or tables.

References

None

* To be submitted after the tutorial on "Using Krypton.