Lab 8 Report - **Implement 3-bit full adder / subtracter, ALU, 4:2 priority encoder**

Josyula V N Taraka Abhishek - 200010021

February 24, 2022

Contents

List of Figures

1 Aim:

To implement 3-bit full adder, 3-bit full subtracter, ALU, 4:2 priority encoder using VHDL code and run the code on CPLD board.

2 Summary of the Experiment:

Write clear logic and implement the following codes as per the instructions given

1.)Write VHDL code for 3-bit adder/subtractor in structural modelling style. The adder/subtractor operation is controlled by signal 'm'.
2. Write VHDL code for ALU which performs following operation depending on selection lines
3. Write VHDL code for 4:2 priority encodes with active high enable pin

3 Procedure

1. Design and implement VHDL code in Quartus Software and compile.
2. Check the RTL and assign the pins, compile again.
3. Create .svf file for dumping to CPLD.
4. Open the terminal and type the commands sequentially.
5. sudo jtag.
6. cable ft2232 vid=0x0403 pid=0x6010.
7. detect.
8. svf ¡svf file location¿ progress.
9. Now test the CPLD board and record observations.

4 Components Used:

- Quartus web version software
- jtag cli
- CPLD board
- USB cable

5 VHDL Code and RTl:

5.1 3-bit full adder, full subtracter with control 'm'

```
--- Write VHDL code for 3-bit adder/subtractor in
--- structural modelling style. The adder/subtractor
--- operation is controlled by signal 'm'.

------------- ENTITY DECLARATION --> 1bit adder ---------

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity bit_adder is
port(A,B,Cin : in STD_LOGIC;

S,Cout : out STD_LOGIC
);
end bit_adder;

architecture dataflow of bit_adder is
Begin

S <= A xor B xor Cin;
Cout <= (A and B) or (B and Cin) or (Cin and A);

end dataflow;


-----------------------------------------------

--- ENTITY DECLARATION --> 3bitAdderSubtractor

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;--define the entity with ports
entity adsub_3bit is
port (a : in STD_LOGIC_VECTOR (2 downto 0);
b : in STD_LOGIC_VECTOR (2 downto 0);
m : in STD_LOGIC;
sum :out STD_LOGIC_VECTOR (2 downto 0);
carry : out STD_LOGIC);
end adsub_3bit;

architecture structural of adsub_3bit is

component bit_adder is
port(A,B,Cin : in std_LOGIC;
```

```
S , Cout : out std_LOGIC);
end component;

signal C1,C2,C3 : std_LOGIC;
signal G1,G2,G3 : std_LOGIC;

begin
G1 <= b(0) xor m;
G2 <= b(1) xor m;
G3 <= b(2) xor m;

u0: bit_adder port map (a(0),G1,m,sum(0),C1);
u1: bit_adder port map (a(1),G2,C1,sum(1),C2);
u2: bit_adder port map (a(2),G3,C2,sum(2),carry);

end structural;


----------------------------------------------------------------------------------------------------


----------------------------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity fulladder is
port(i1, i2, i3: in bit;
o1, o2 : out bit);
end fulladder;

architecture dataflow of fulladder is
Begin
o1 <= i1 xor i2 xor i3;
o2 <= (i1 and i2) or ((i1 xor i2) and i3);
end dataflow;
----------------------------------------------------------------------------------------------------
```

5.2 ALU

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use ieee.NUMERIC_STD.all;
entity ALU is
Port ( A, B : in STD_LOGIC_VECTOR(2 downto 0); -- 2 inputs 8-bit
ALU_Sel : in STD_LOGIC_VECTOR(1 downto 0);
ALU_Out : out STD_LOGIC_VECTOR(3 downto 0));
```

```
end ALU;

architecture Behavioral of ALU is
Begin
c1: process (A,B,ALU_Sel)
Begin

case ALU_Sel is
when "00" =>
ALU_Out <= ('0'&A) + ('0'&B);
when "01" =>
ALU_Out <= ('0'&A) - ('0'&B);
when "10" =>
ALU_Out(3) <= '0';
ALU_Out(2) <= A(2) and B(2);
ALU_Out(1) <= A(1) and B(1);
ALU_Out(0) <= A(0) and B(0);
when "11" =>
ALU_Out(3) <= '0';
ALU_Out(2) <= A(2) xor B(2);
ALU_Out(1) <= A(1) xor B(1);
ALU_Out(0) <= A(0) xor B(0);
end case;
end process c1;
end Behavioral;
```

5.3 4:2 Priority encoder

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use ieee.NUMERIC_STD.all;

entity pe is
Port ( inp : in STD_LOGIC_VECTOR (3 downto 0);
enable : in STD_LOGIC ;
OUTPUT : out STD_LOGIC_VECTOR (1 downto 0));
end pe;



architecture Behavioral of pe is
begin
process(inp)
begin
if enable = '1' then
```

```
if (inp(3)='1') then
OUTPUT <= "11";
elsif (inp(2)='1') then
OUTPUT <= "10";
elsif (inp(1)='1') then
OUTPUT <= "01";
elsif (inp(0)='1') then
OUTPUT <= "00";
end if;
else
OUTPUT <= "00";
end if;
end process;
end Behavioral;
```

6 Snapshots:

6.1 RTL Diagrams

Figure 1: Priority encoder

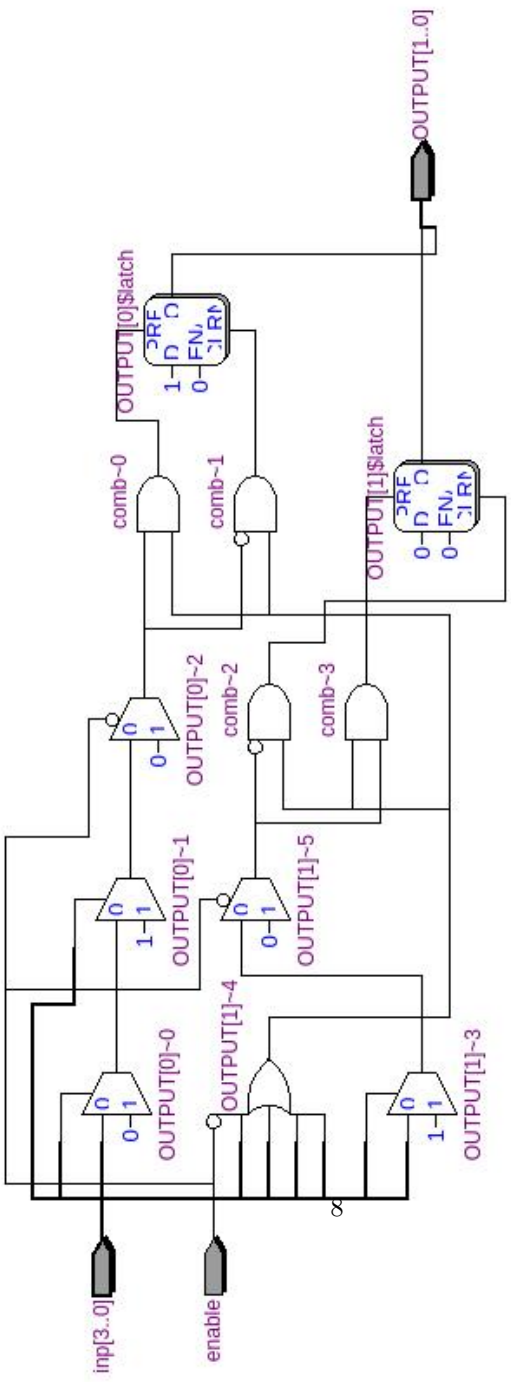


Figure 2: ALU

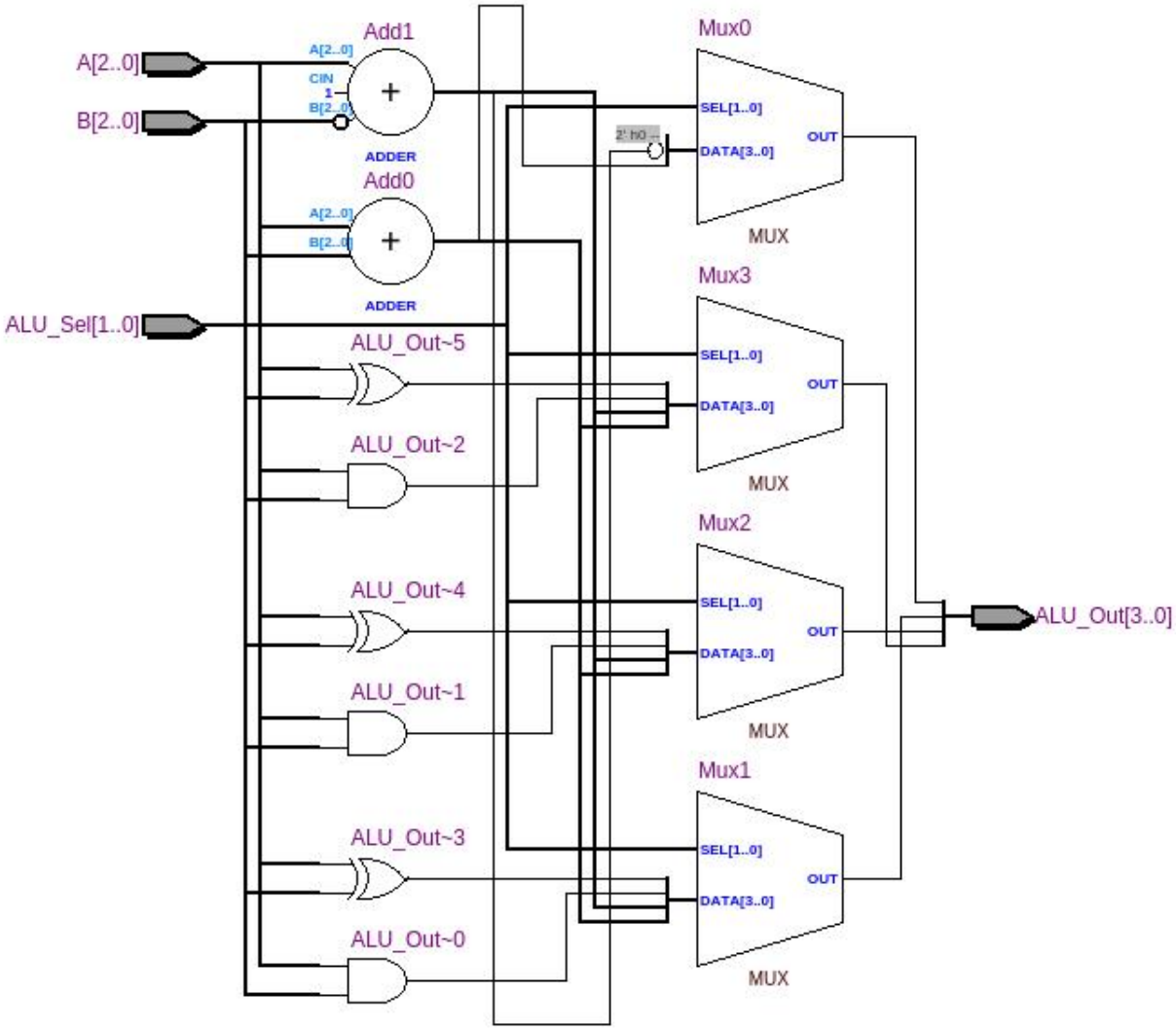
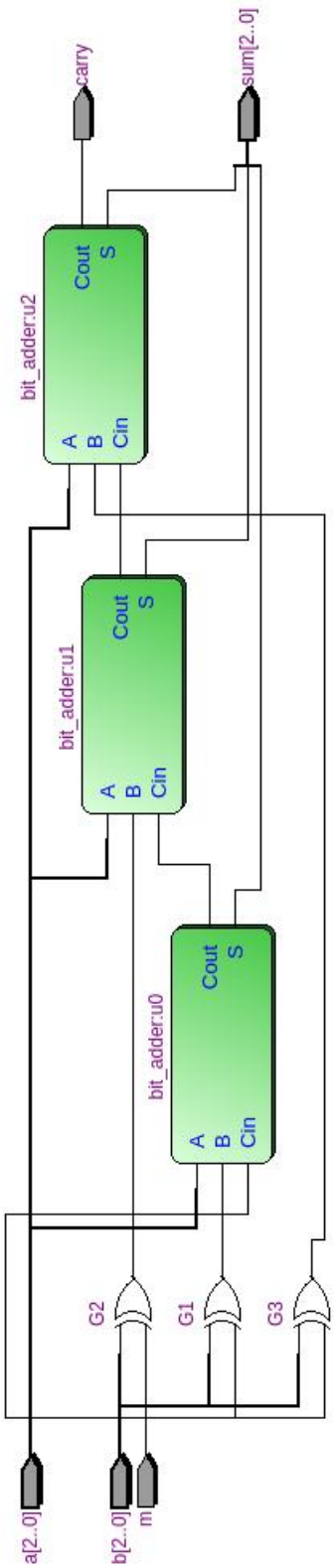
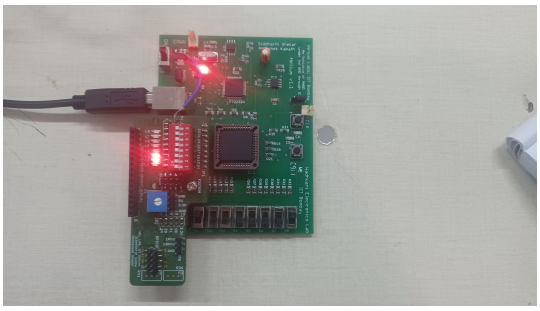
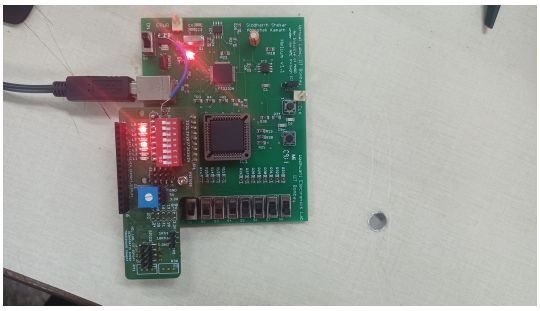
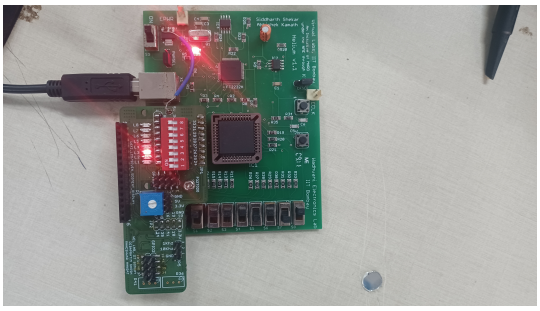

A[2..0]
B[2..0]
ALU_Sel[1..0]
Add1
A[2..0]
CIN
1
B[2..0]
ADDER
Add0
A[2..0]
B[2..0]
ADDER
ALU_Out~5
ALU_Out~2
ALU_Out~4
ALU_Out~1
ALU_Out~3
ALU_Out~0
Mux0
SEL[1..0]
DATA[3..0]
OUT
MUX
Mux3
SEL[1..0]
DATA[3..0]
OUT
MUX
Mux2
SEL[1..0]
DATA[3..0]
OUT
MUX
Mux1
SEL[1..0]
DATA[3..0]
OUT
MUX
ALU_Out[3..0]

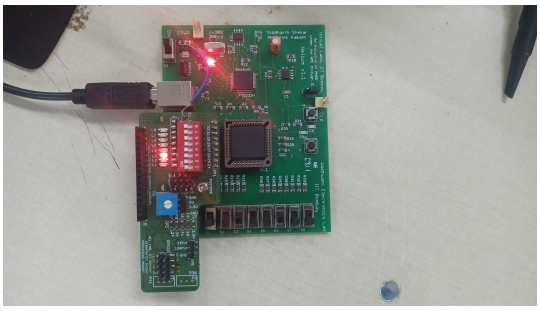
Figure 3: adsub 3bit

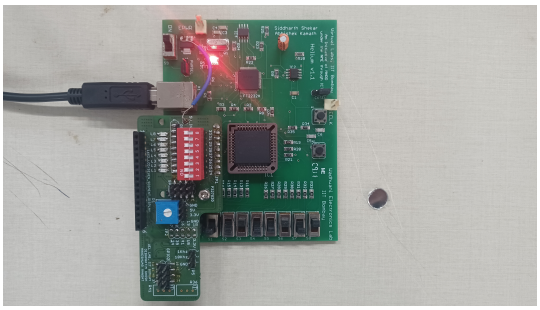


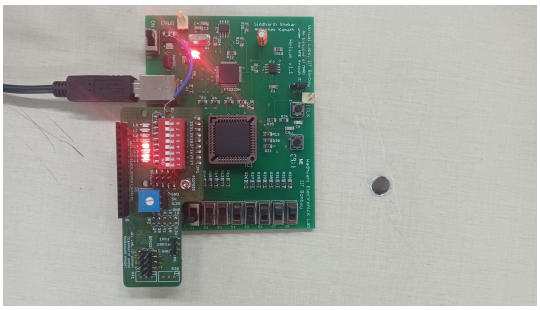


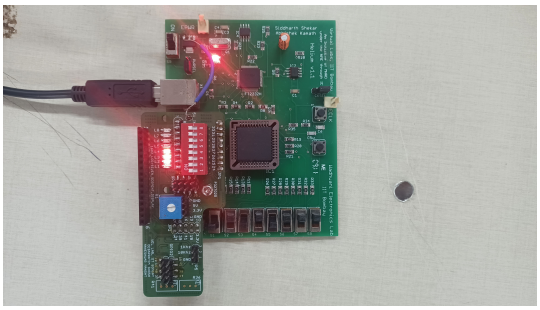


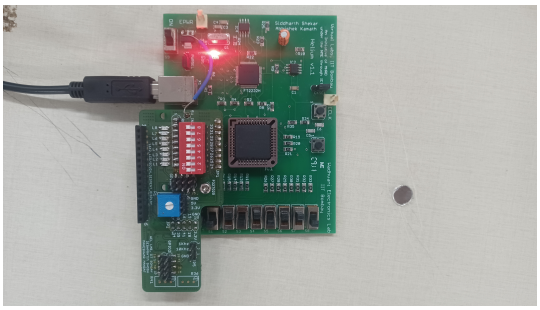


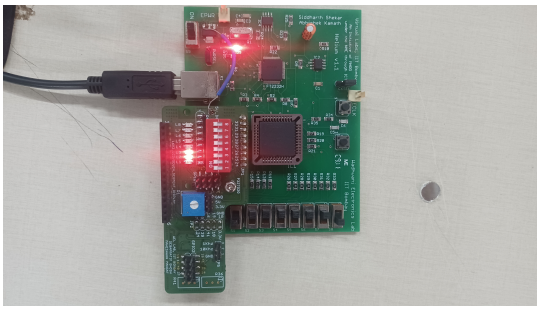












7 Results and Discussion:

1. The CPLD board functioned as designed
2. From this lab experiment I have understood the functionality of CPLD Board and how to write VHDL code for priority encoder, ALU, full adder and subtracter.

8 Conclusion:

In this lab, we implemented full adder, full subtracter, ALU and 4:2 proirity encoder by writing programs in VHDL and executed the working of the circuit on the CPLD board by powering it up with a PC with help of Quartus II web version software. We also generated RTLs for these using VHDL. We also used jtag shell to run VHDL code in CPLD board and noted all observations.

Team Members:

- Josyula V N Taraka Abhishek - 200010021
- M V Karthik - 200010030