Experiment 2: DeMUX

Debasish Panda 21D070021

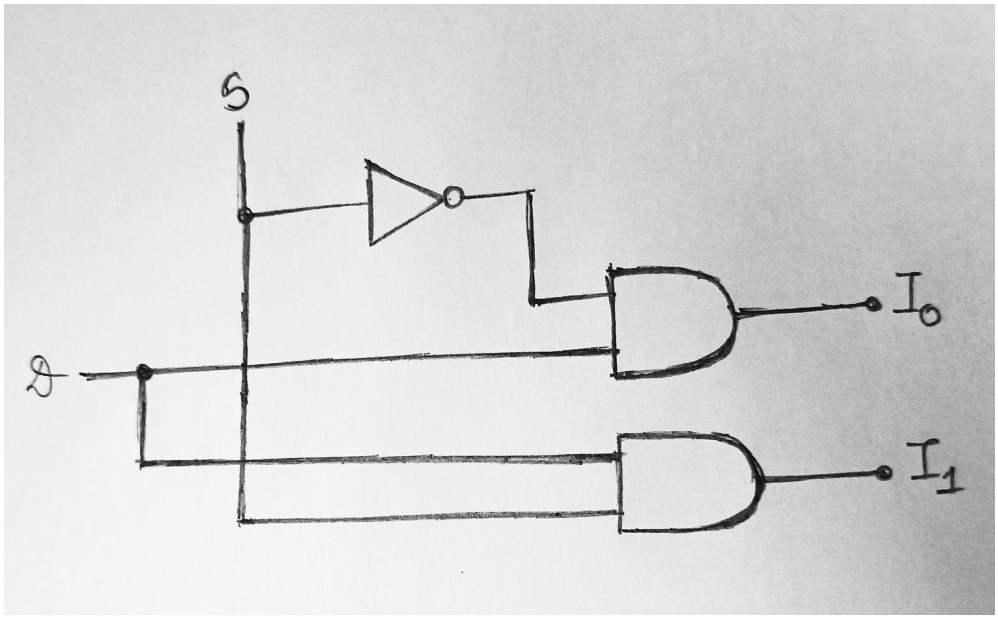
August 16, 2022

1 Overview of the experiment

The purpose of the experiment is to design a 1x2 DeMUX, a 1x4 DeMUx and a 4-bit 1x4 DeMUX using fundamental logic gates as well as describe them in VHDL using Structural Modelling.
In order to perform the experiment, first a 1x2 DeMUX is designed using AND gates and a NOT gate. A rough sketch of the digital circuit is drawn first. For designing the 1x4 DeMUX, we need 3 1x2 DeMUXes. Finally, the 4-bit 1x4 DeMUX is designed using the single-bit 1x4 DeMUXes. The appropriate code for Structural Modelling of these components is written using VHDL on Quartus Prime software.
In this report first we discuss the setup required for the experiment and the approach used by me to solve the assignment. Next, I put forward the observations of the experiment in the form of truth tables for the 1x2 DeMUX, 1x4 DeMUX and the 4-bit 1x4 DeMUX. Attached alongwith are the pen-and-paper diagrams and pictures showing the simulation output of the two designs.

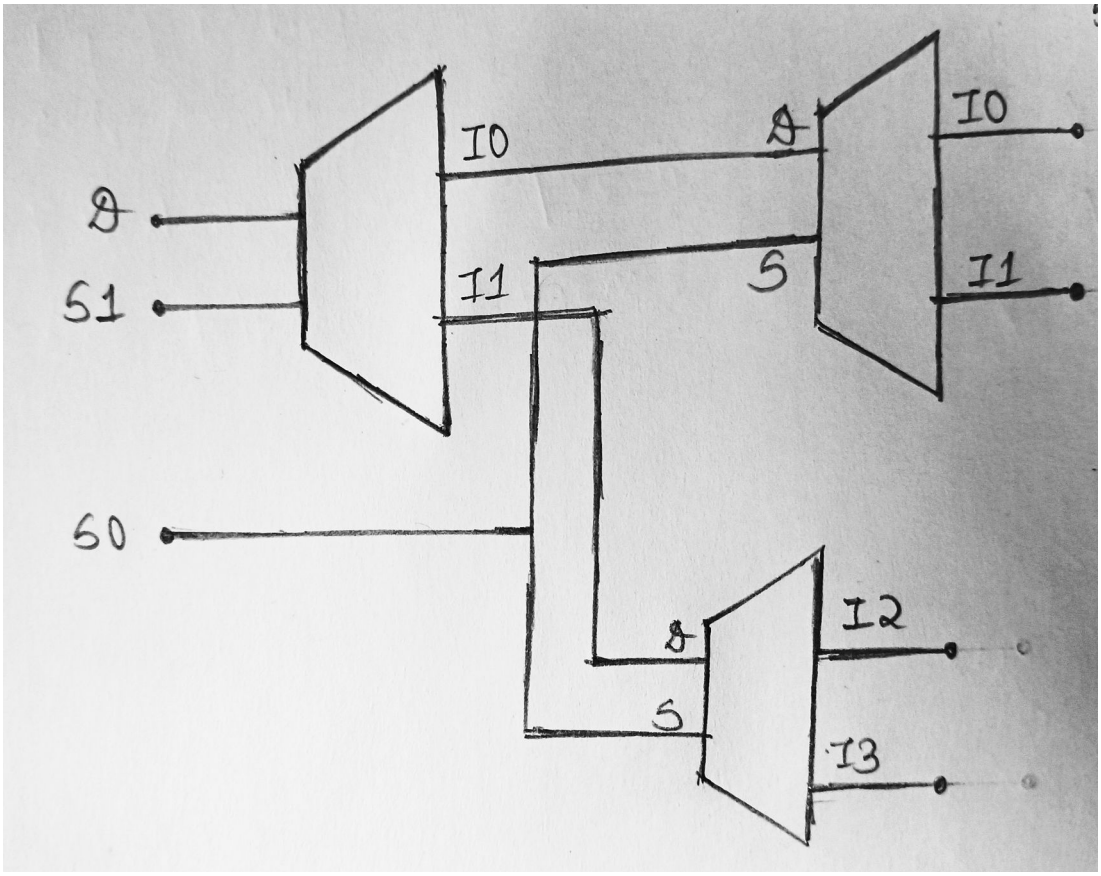
2 Experimental setup

First we need to design a 1x2 DeMUX gate using logic gates. The 1x2 DeMUX thus designed can be further used to design the 1x4 DeMUX. The pen-and-paper diagram of a 1x2 DeMUX is:



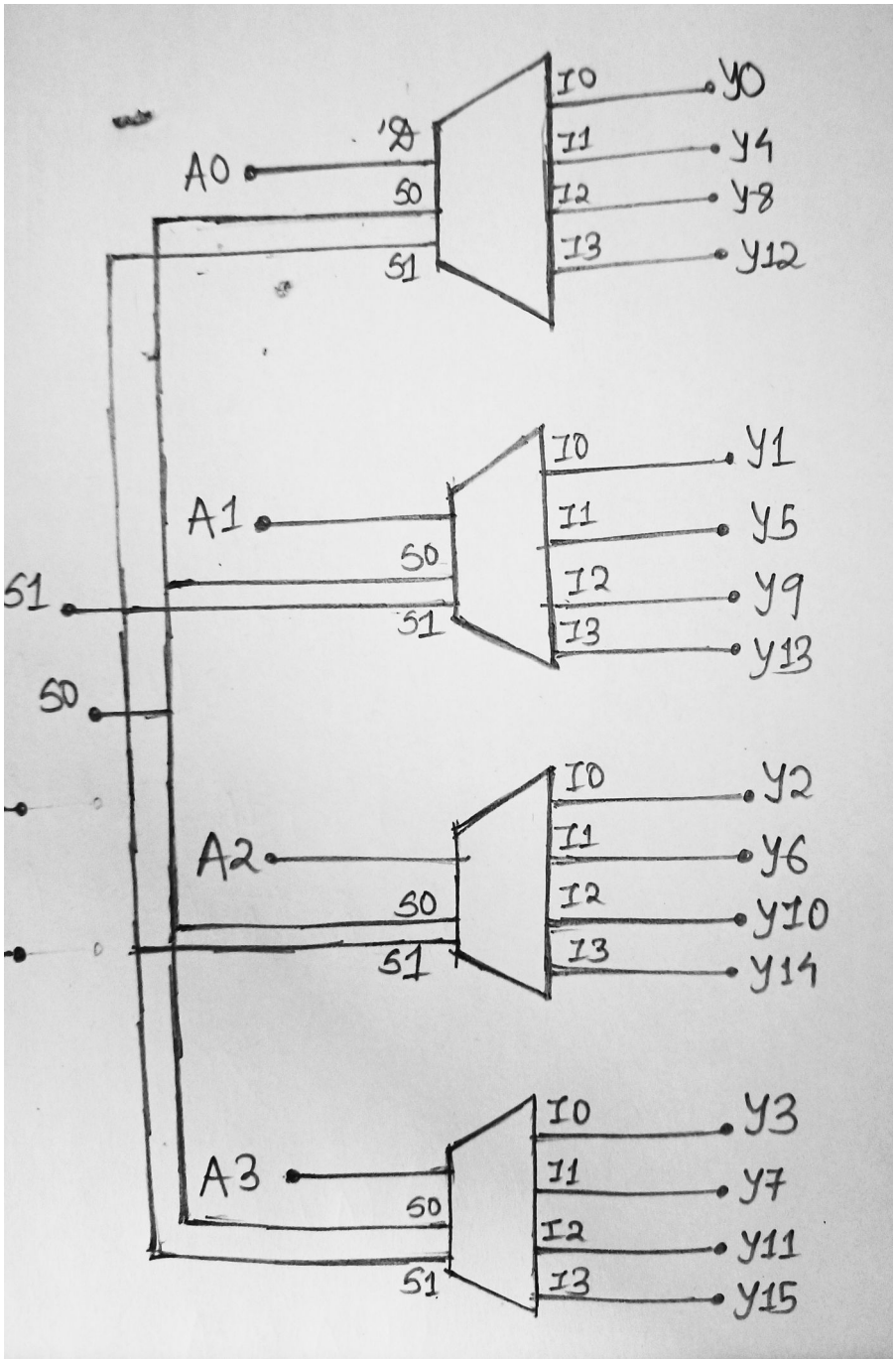
1x2 DeMUX

The pen-and-paper diagram of the 1x4 DeMUX is:



1x4 DeMUX

The pen-and-paper diagram of the 4-bit 1x4 DeMUX is:



4-bit 1x4 DeMUX

Now that we have the required circuits, we need to assign various component instances, architectures and signals to individual circuits as per VHDL syntax. The required code for the same is discussed in the next section.

2.1 Design Documentation and VHDL Code

The VHDL code for designing the entity-architecture pairs associated with the digital circuit is as follows:

2.1.1 1x2 DeMUX

```
library ieee;
use ieee.std_logic_1164.all;

library work;
use work.Gates.all;

entity DeMUX_2_1 is
  port(S,D: in std_logic; I0,I1: out std_logic);
end entity DeMUX_2_1;

architecture Simple of DeMUX_2_1 is
  signal S1: std_logic;

begin
  NOT1 : INVERTER port map (A => S, Y => S1);
  OR1 : AND_2 port map (A => S1 , B => D , Y => I0);
  OR2 : AND_2 port map (A => D , B => S, Y => I1);
end Simple;
```

2.1.2 1x4 DeMUX

```
library ieee;
use ieee.std_logic_1164.all;

library work;
use work.Gates.all;
```

```
entity DeMUX_2_1 is
  port(S,D: in std_logic; I0,I1: out std_logic);
end entity DeMUX_2_1;

architecture Simple of DeMUX_2_1 is
  signal S1: std_logic;

begin
  NOT1 : INVERTER port map (A => S, Y => S1);
  OR1 : AND_2 port map (A => S1 , B => D , Y => I0);
  OR2 : AND_2 port map (A => D , B => S, Y => I1);
end Simple;

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

library ieee;
use ieee.std_logic_1164.all;

library work;
use work.Gates.all;

entity DeMUX_4_1 is
  port(S1,S0,D: in std_logic; I3,I2,I1,I0: out std_logic);
end entity DeMUX_4_1;

architecture Unique of DeMUX_4_1 is
  signal P1,P2: std_logic;

component DeMUX_2_1 is
  port(S,D: in std_logic; I0,I1: out std_logic);
end component DeMUX_2_1;

begin
 DeMUX1 : DeMUX_2_1 port map (S => S1, D => D, I0 => P1, I1 => P2);
 DeMUX2 : DeMUX_2_1 port map (S => S0, D => P1, I0 => I0, I1 => I1);
 DeMUX3 : DeMUX_2_1 port map (S => S0, D => P2, I0 => I2, I1 => I3);
end Unique;
```

2.1.3 4-bit 1x4 DeMUX

```
library ieee;
use ieee.std_logic_1164.all;

library work;
use work.Gates.all;

entity DeMUX_2_1 is
  port(S,D: in std_logic; I0,I1: out std_logic);
end entity DeMUX_2_1;

architecture Simple of DeMUX_2_1 is
  signal S1: std_logic;

begin
  NOT1 : INVERTER port map (A => S, Y => S1);
  OR1 : AND_2 port map (A => S1 , B => D , Y => I0);
  OR2 : AND_2 port map (A => D , B => S, Y => I1);
end Simple;

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

library ieee;
use ieee.std_logic_1164.all;

library work;
use work.Gates.all;

entity DeMUX_4_1 is
  port(S1,S0,D: in std_logic; I3,I2,I1,I0: out std_logic);
end entity DeMUX_4_1;

architecture Unique of DeMUX_4_1 is
  signal P1,P2: std_logic;

component DeMUX_2_1 is
  port(S,D: in std_logic; I0,I1: out std_logic);
```

```
end component DeMUX_2_1;

begin
 DeMUX1 : DeMUX_2_1 port map (S => S1, D => D, I0 => P1, I1 => P2);
 DeMUX2 : DeMUX_2_1 port map (S => S0, D => P1, I0 => I0, I1 => I1);
 DeMUX3 : DeMUX_2_1 port map (S => S0, D => P2, I0 => I2, I1 => I3);
end Unique;

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

library ieee;
use ieee.std_logic_1164.all;

library work;
use work.Gates.all;

entity DeMUX_3 is
 port(A3,A2,A1,A0,S1,S0: in std_logic; Y15,Y14,Y13,Y12,Y11,Y10,Y9,Y8,Y7,Y6,Y5,Y4,Y
end entity DeMUx_3;

architecture Complex of DeMUX_3 is

component DeMUX_4_1 is
  port(S1,S0,D: in std_logic; I0,I1,I2,I3: out std_logic);
end component DeMUX_4_1;

begin
 DeMUX_1 : DeMUX_4_1 port map (S0 => S0, S1 => S1, D => A0, I0 => Y0, I1 => Y4, I2
 DeMUX_2 : DeMUX_4_1 port map (S0 => S0, S1 => S1, D => A1, I0 => Y1, I1 => Y5, I2
 DeMUX_3 : DeMUX_4_1 port map (S0 => S0, S1 => S1, D => A2, I0 => Y2, I1 => Y6, I2
 DeMUX_4 : DeMUX_4_1 port map (S0 => S0, S1 => S1, D => A3, I0 => Y3, I1 => Y7, I2
end Complex;
```

3 Observations

The experiment was conducted successfully, and the expected output was obtained through the RTL Simulation. The expected output of the 1x2

DeMUX, 1x4 DeMUX and the 4-bit 1x4 DeMUX is shown in the form of the truth table below, alongwith the final RTL Simulation results of the three designs.

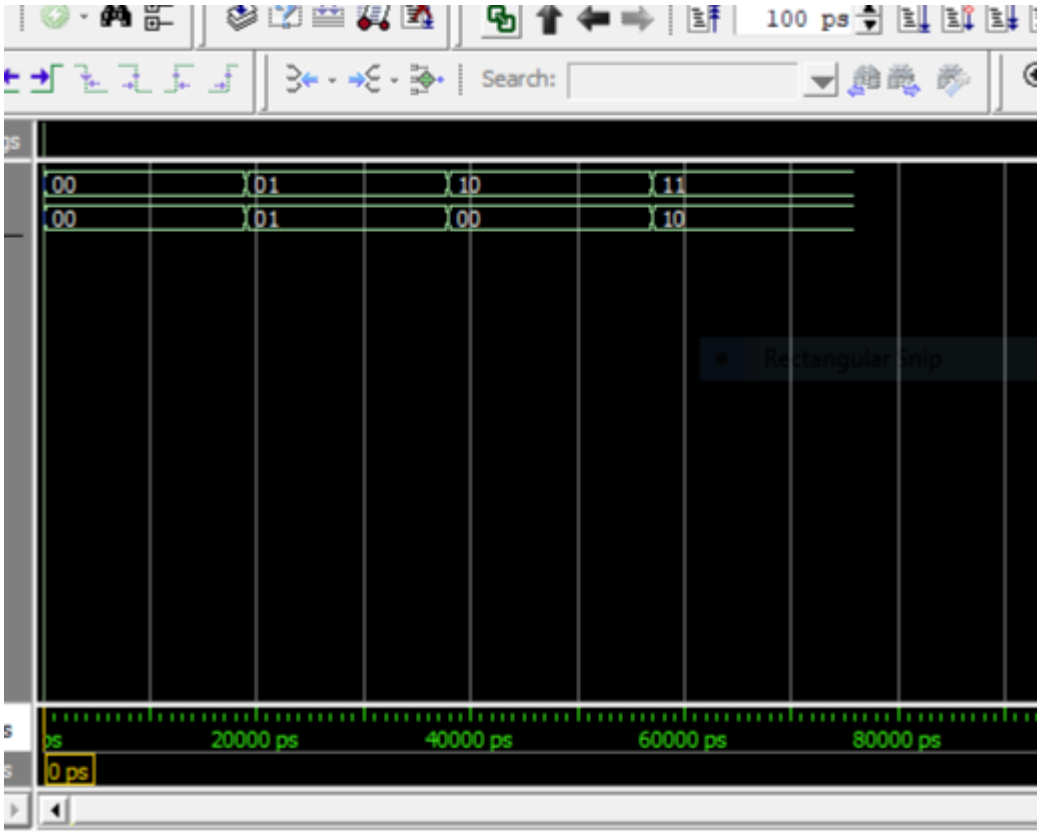
S	D	Y1	Y0
0	0	0	0
0	1	0	1
1	0	0	0
1	1	1	0

Table 1: 1x2 DeMUX truth table

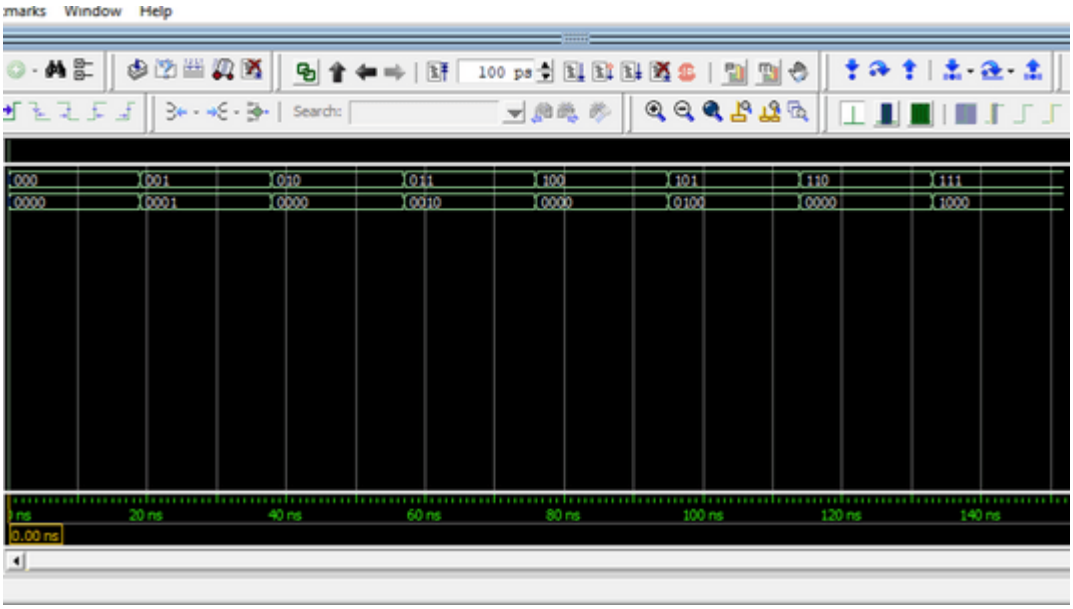
S2	S1	In0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	0	0
0	1	1	0	0	1	0
1	0	0	0	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	0	0
1	1	1	1	0	0	0

Table 2: 1x4 DeMUX truth table

4



RTL Simulation of 1x2 DeMUX



RTL Simulation of 1x4 DeMUX

RTL Simulation of 4-bit 1x4 DeMUX

4 References

[1] EE214 Github page

[2] Overleaf LaTeX tutorial for beginners