UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

DEPARTMENT OF SPACE AND APPLICATION

COURSE: ELECTRONICS

LECTURER: NGUYEN TRAN THUAT

INSTRUCTOR: NGUYEN XUAN TRUONG

16 - Output Demultiplexer

Group:

Duong Thu Phuong - 22BI13362

Ha Bui Khoi Nguyen - 22BI13337

February 25th, 2024

Table of contents

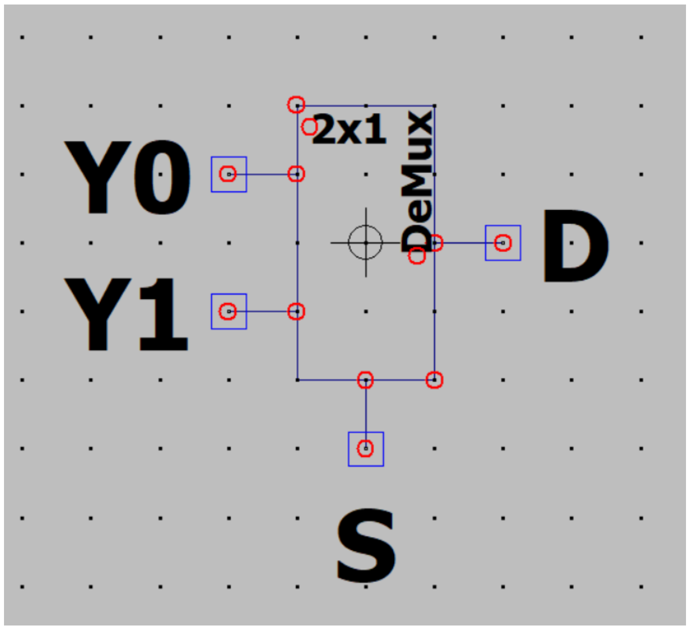
Introduction

The demultiplexer is also known as a **Demux**. It is a device with one input and multiple output lines and is used to send a signal to one of the many devices.

The Demux has a maximum of 2^n data outputs, n selection lines and a single input line. There are four types of demultiplexer:

- 1-2 demultiplexer: 2 outputs, 1 selection line, 1 input

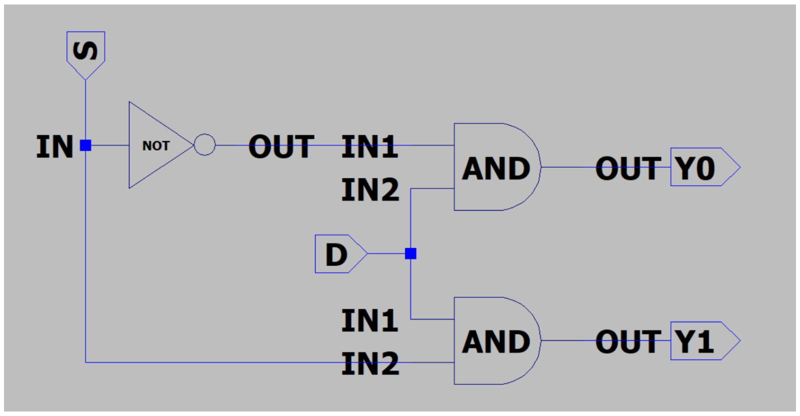
Block diagram:



Truth table:

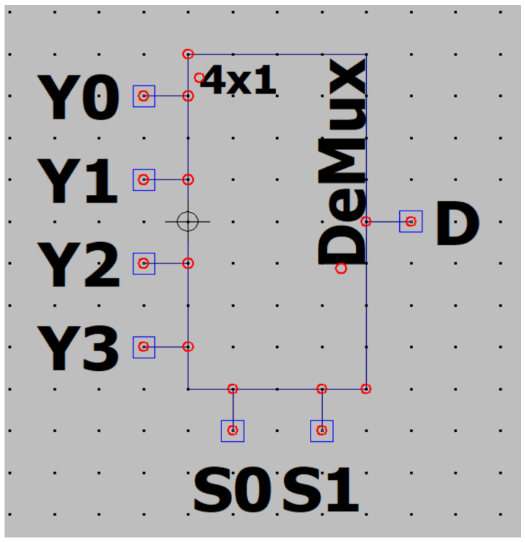
Select	Input	Outputs	
S	D	Y2	Y1
0	0	0	0
0	1	0	1
1	0	0	0
1	1	1	0

Logical circuit:



- 1-4 demultiplexer: 1 input line, 4 output lines, 2 selection lines

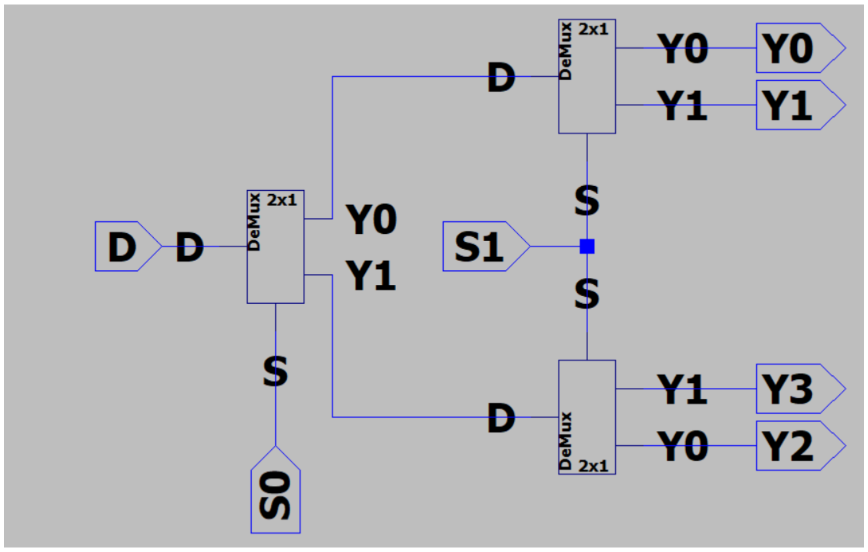
Block diagram:



Truth Table:

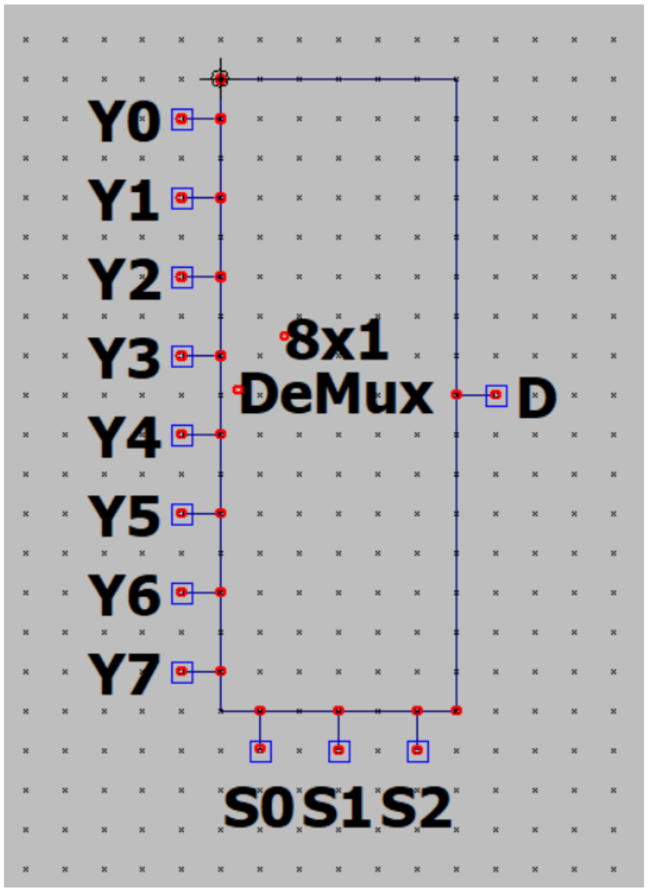
Input	Select Lines		Output Lines			
I	S_0	S_1	D_0	D_1	D_2	D_3
I	0	0	1	0	0	0
I	1	0	0	0	1	0
I	0	1	0	1	0	0
I	1	1	0	0	0	1

Logical circuit:



- 1-8 demultiplexer: 1 input line, 8 output lines, 3 select lines

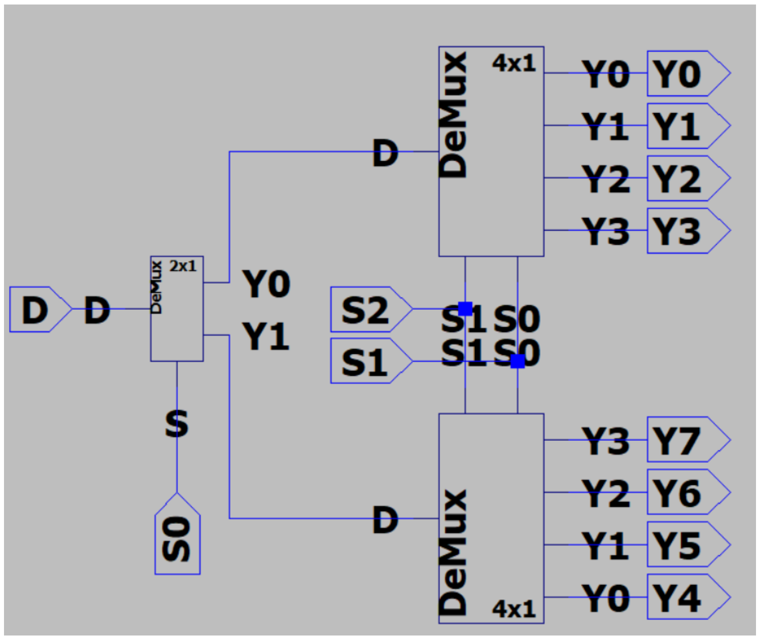
Block diagram:



Truth Table:

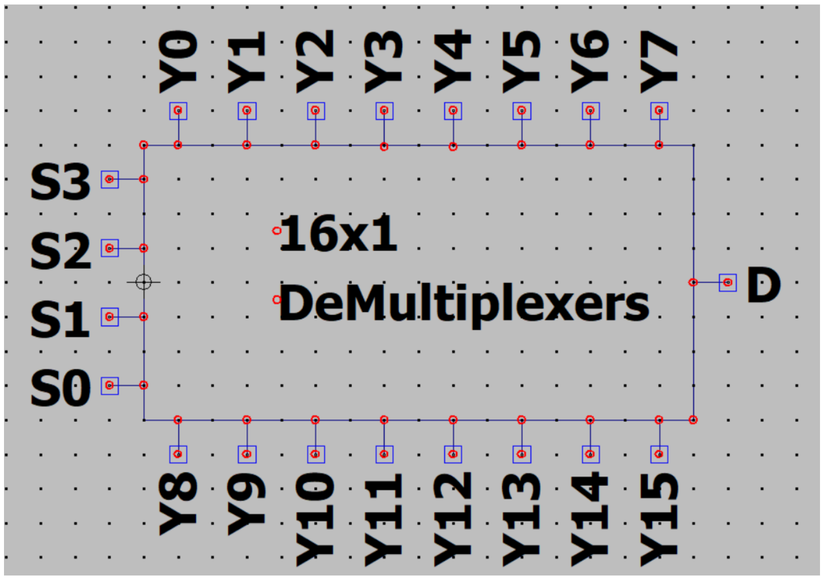
Input	Select Lines			Output Lines							
D	S_2	S_1	S_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
D	0	0	0	0	0	0	0	0	0	0	D
D	0	0	1	0	0	0	0	0	0	D	0
D	0	1	0	0	0	0	0	0	D	0	0
D	0	1	1	0	0	0	0	D	0	0	0
D	1	0	0	0	0	0	D	0	0	0	0
D	1	0	1	0	0	D	0	0	0	0	0
D	1	1	0	0	D	0	0	0	0	0	0
D	1	1	1	D	0	0	0	0	0	0	0

Logical circuit:

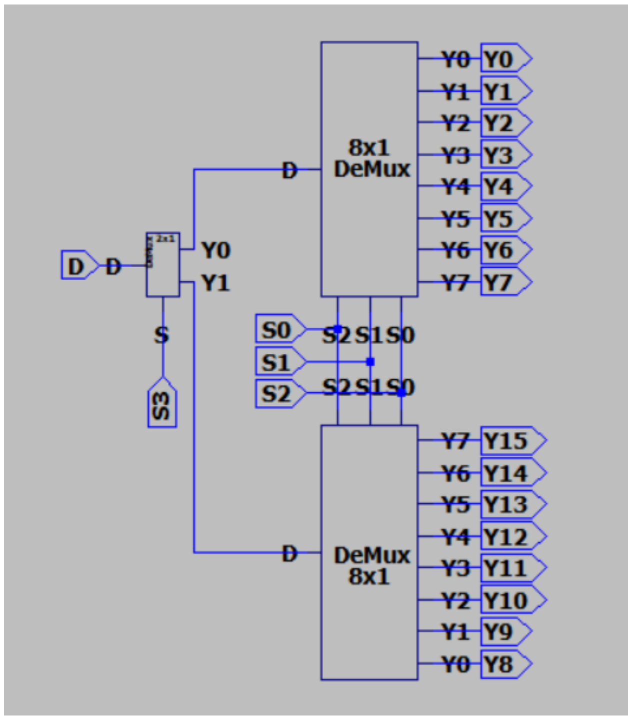


- 1-16 demultiplexer: 1 input, 16 outputs, 4 select lines

Block diagram:



Logical circuit:



Truth Table:

Inputs				Outputs															
S_3	S_2	S_1	S_0	Y_{15}	Y_{14}	Y_{13}	Y_{12}	Y_{11}	Y_{10}	Y_9	Y_8	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	0	0
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	A	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	A	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	0	A	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	A	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0	A	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	A	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	A	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	A	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	A	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	A	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	A	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	A	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

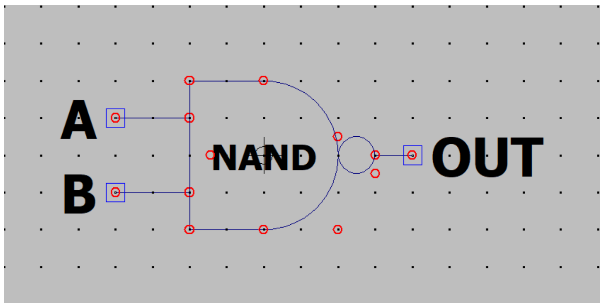
In a 16-output demultiplexer, the demultiplexer will take a single input signal and route it to one of the sixteen output channels based on control signals.

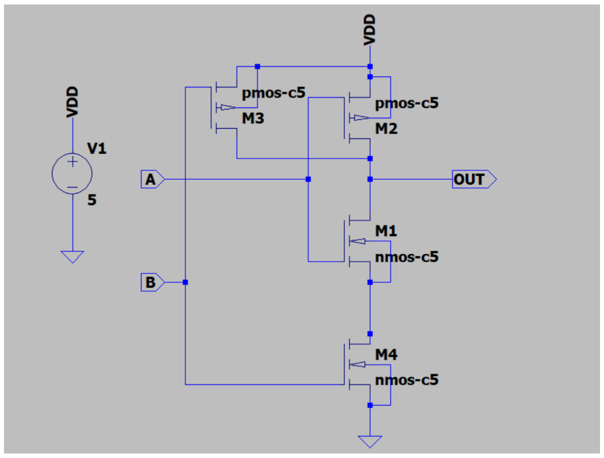
Simulation Process

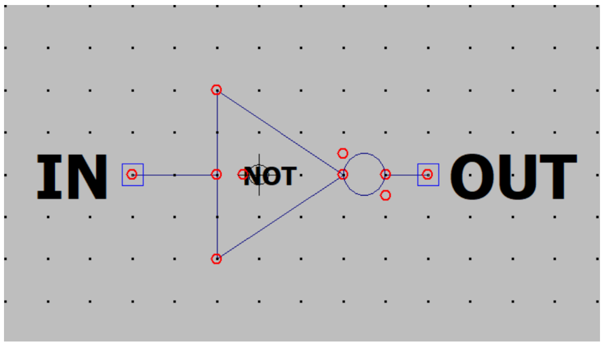
Step 1. Make Universal NAND gate and Inverter using the nmos and pmos.

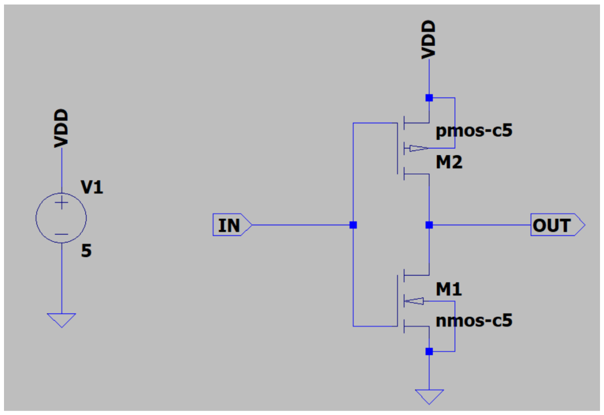
Symbol file

Schematic file



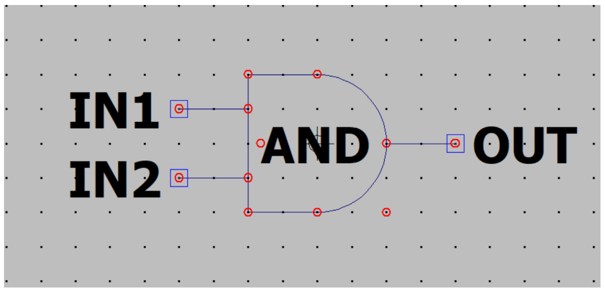


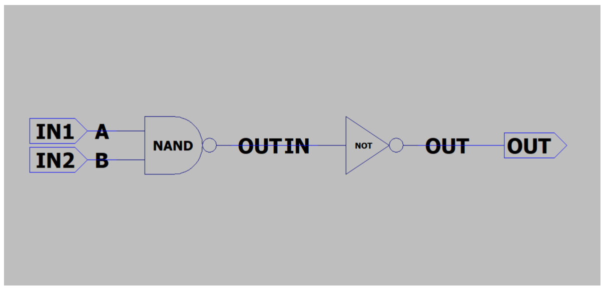


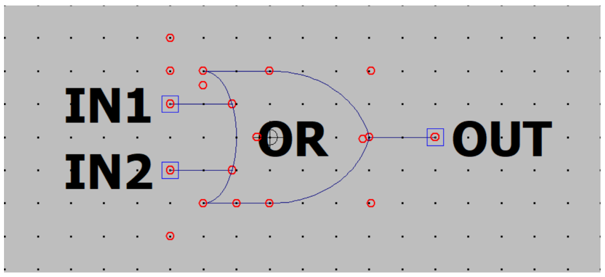


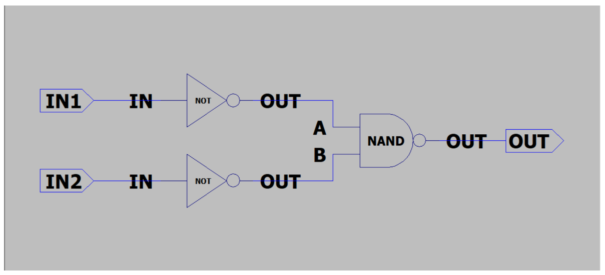
Step 2. Make other logic gate using NAND gate (including NAND gate, NOR gate).

Symbol file *Schematic file*



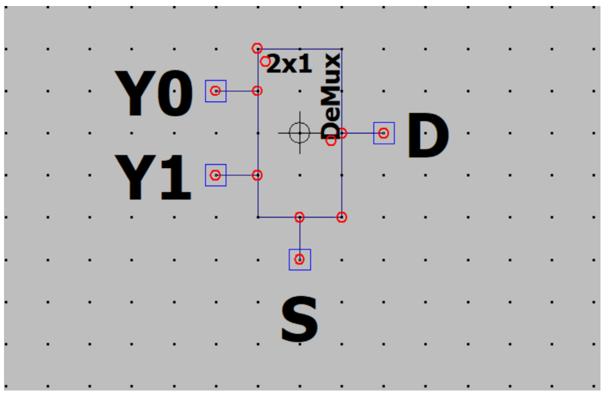


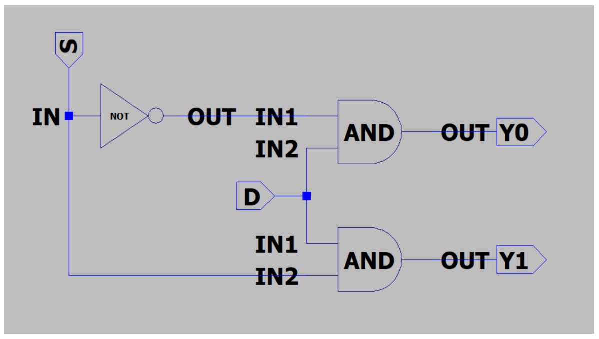




Step 3. Make 2-output demultiplexer

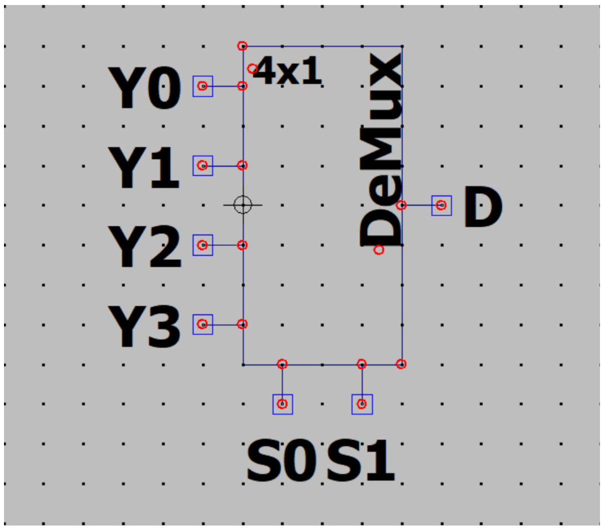
Symbol file *Schematic file*



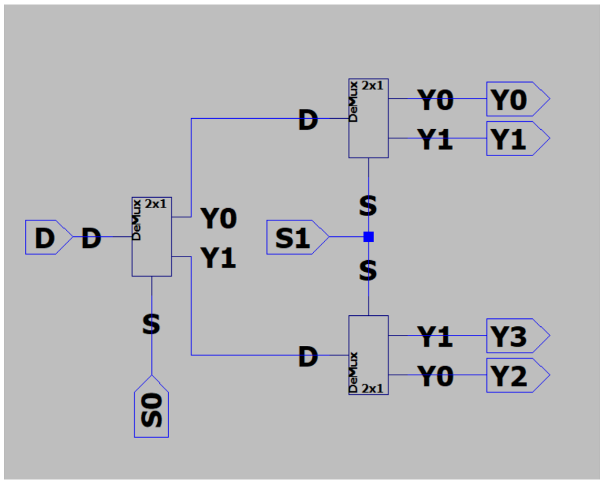


Step 4. Combine two 2-output demultiplexers to make a 4-output demultiplexer.

Symbol file

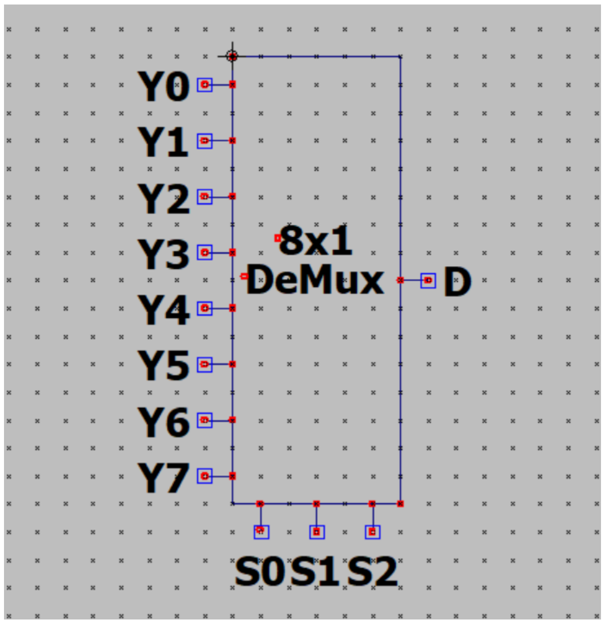


Schematic file

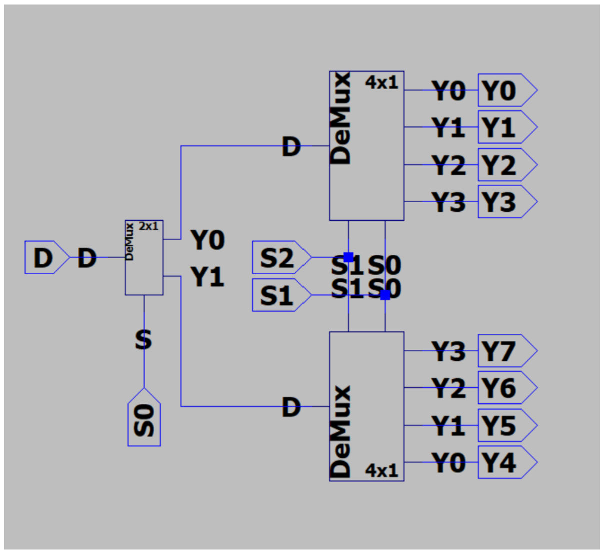


Step 5. Combine two 4-output demultiplexers to make an 8-output demultiplexer.

Symbol file



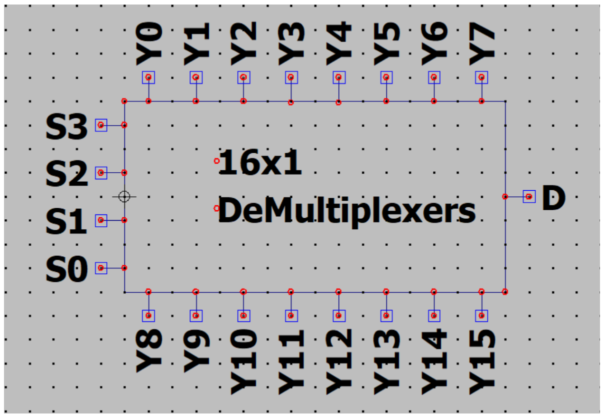
Schematic file

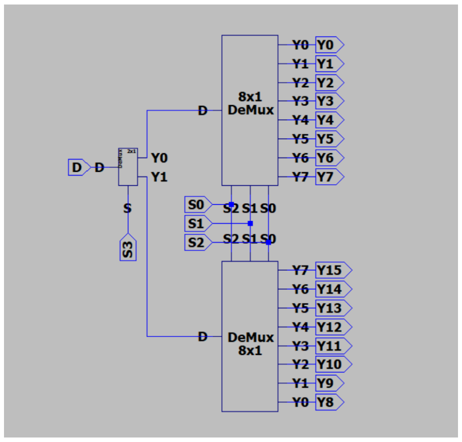


Step 6. Combine two 8-output demultiplexers to make a 16-output demultiplexer.

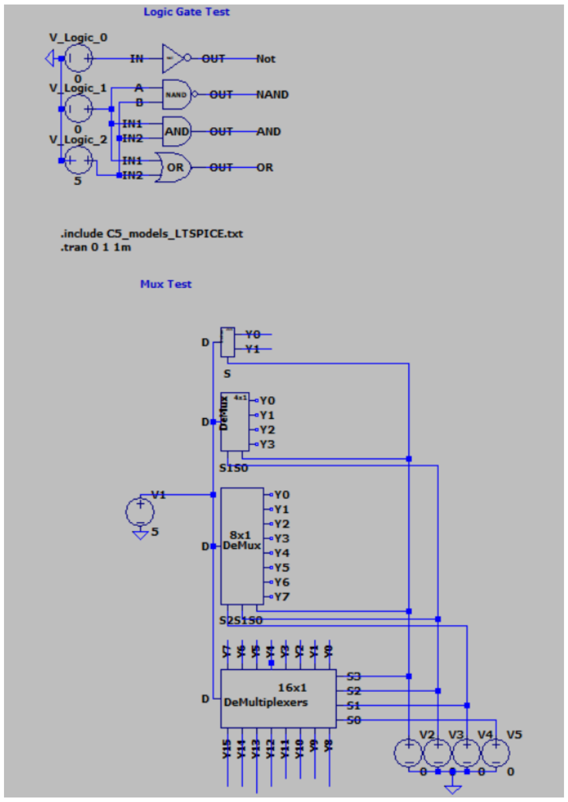
Symbol file

Schematic file





Step 7. Test and run simulation.



```
.include C5_models_LTSPICE.txt
.tran 0 1 1m
```

- Add all the SPICE directives that we need.
- Wire input signal into D pin, and the state into $S_0 \rightarrow S_3$ pin.
- Set the input signal to what we want and observe the output signal.
- Check correction with the theory truth table.

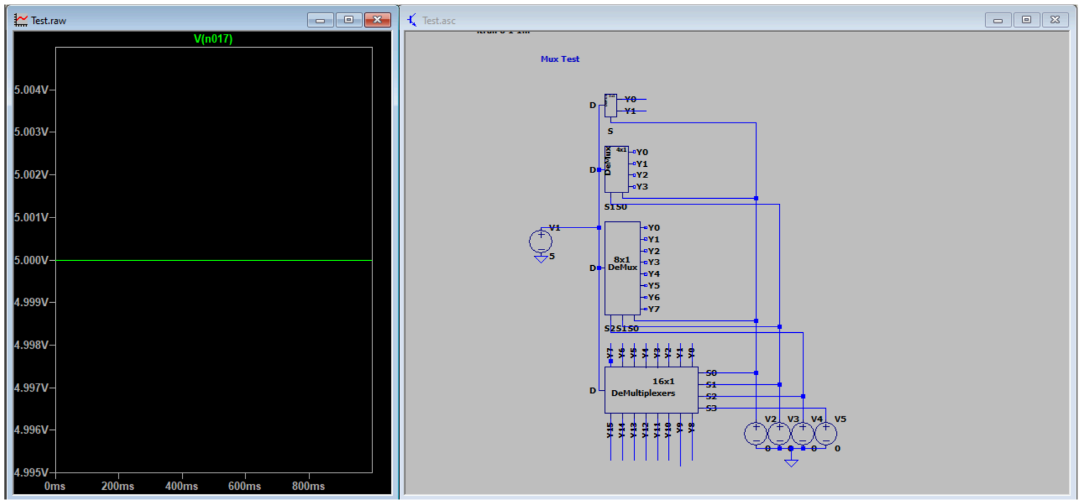
Note:

- The schematic and the symbol of one component should be named the same and saved in one directory.
- Because we use the third party mosfet (C5_models_LTSPICE.txt) so in the final test bench we need to add the SPICE directive ".include 5_models_LTSPICE.txt" with the .text file same into the same directory and we only need to include in the final test bench not every component.
- The above method is not the only way to make the Demux, we can make it directly from the logic gate or any factor of 2^x (e.g. 1x16 from 1x4 or 1x8).
- The signal greater than 3V then it is 1 in boolean algebra, otherwise it is zero (this is due to the data of the mosfet and the design), for this schematic we will use 5V.

Result

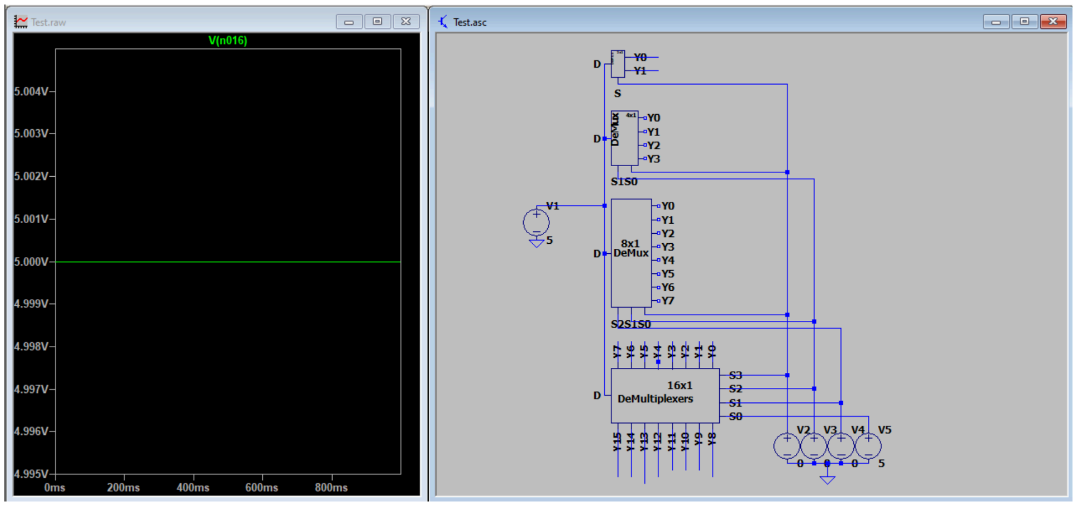
Case 1. Binary code: 0000 ⇒ Output taken from Y_0

S3 = 0V
S2 = 0V
S1 = 0V
S0 = 0V



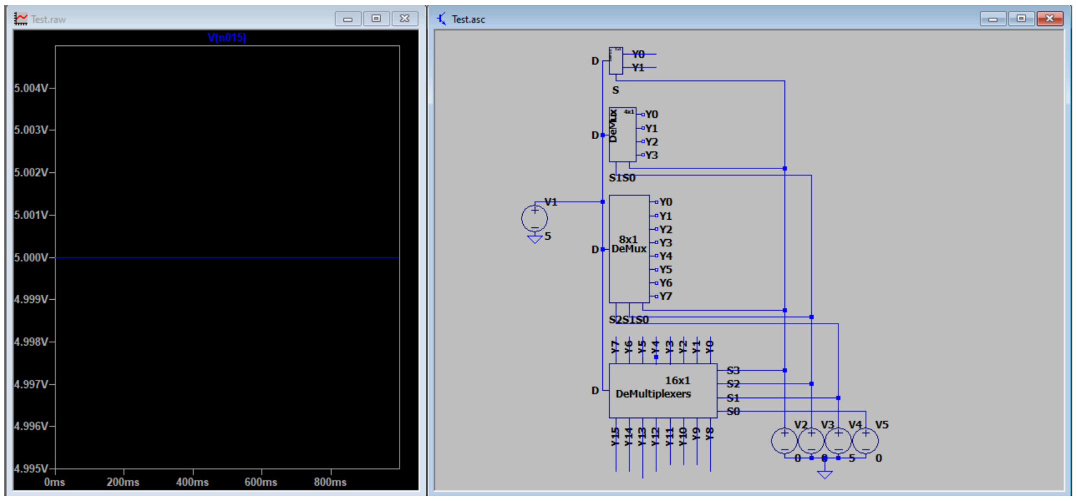
Case 2. Binary code: 0001 ⇒ Output taken from Y_1

S3 = 0V
S2 = 0V
S1 = 0V
S0 = 5V



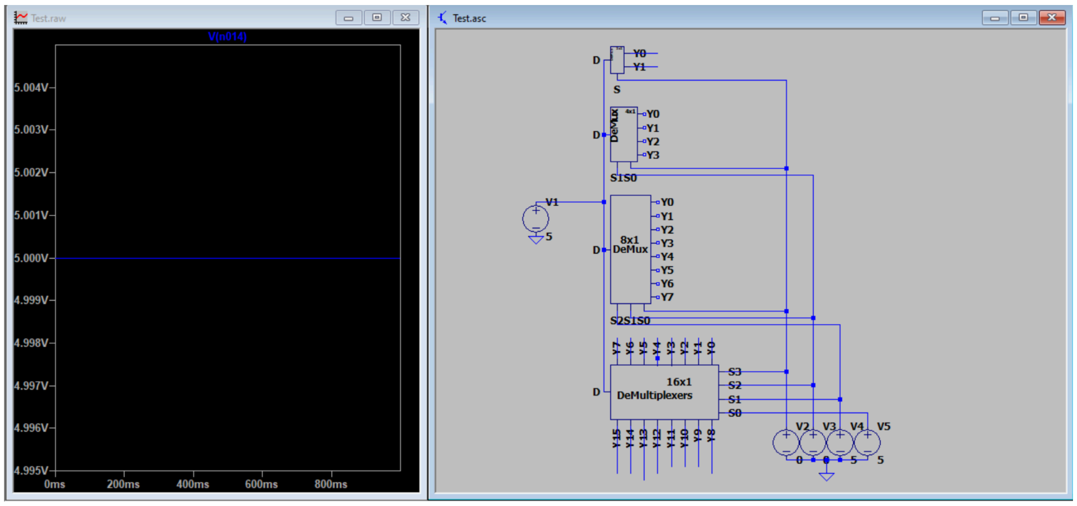
Case 3. Binary code: 0010 ⇒ Output taken from Y_2

S3 = 0V
S2 = 0V
S1 = 5V
S0 = 0V



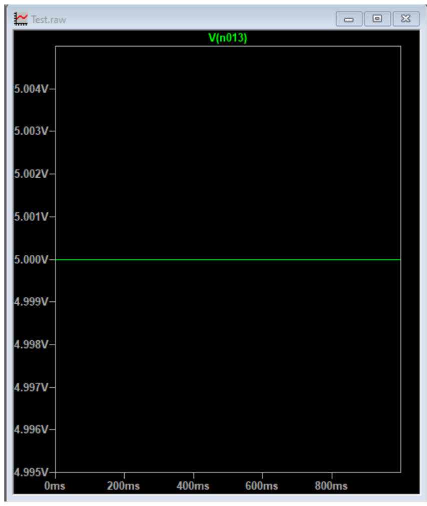
Case 4. Binary code: 0011 ⇒ Output taken from Y_3

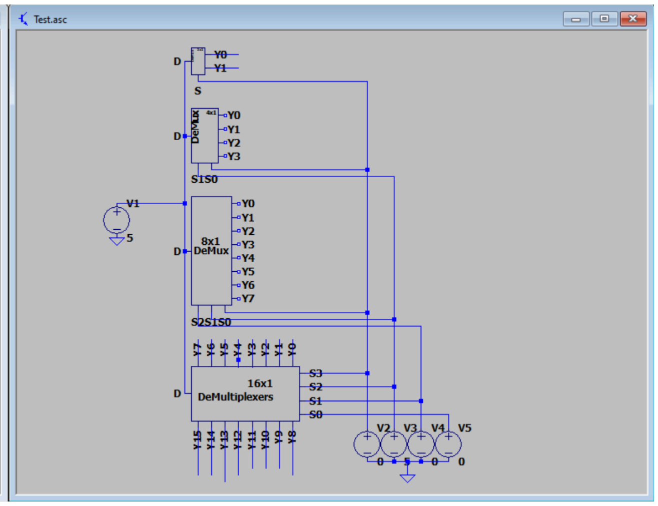
S3 = 0V
S2 = 0V
S1 = 5V
S0 = 5V



Case 5. Binary code: 0100 ⇒ Output taken from Y_4

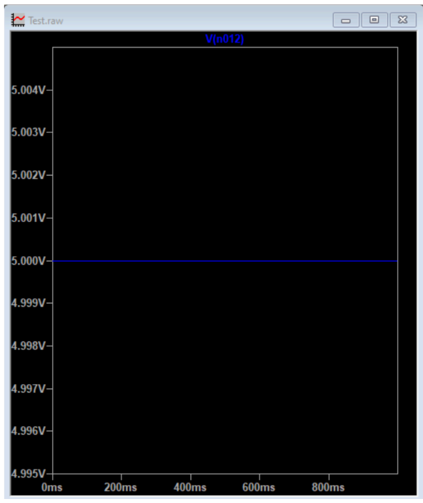
S3 = 0V
S2 = 5V
S1 = 0V
S0 = 0V

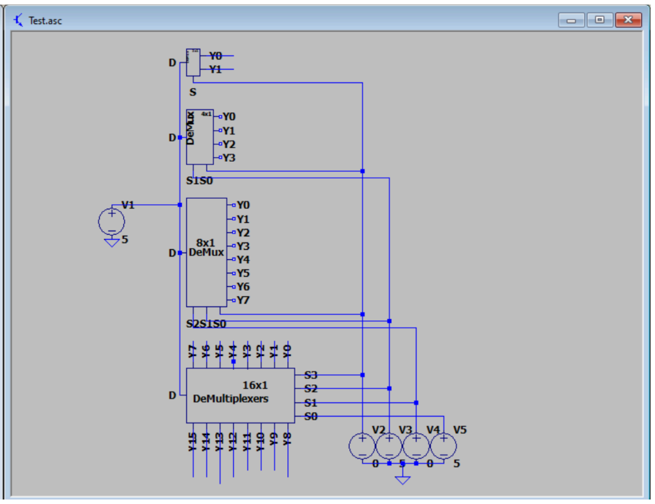




Case 6. Binary code: 0101 ⇒ Output taken from Y_5

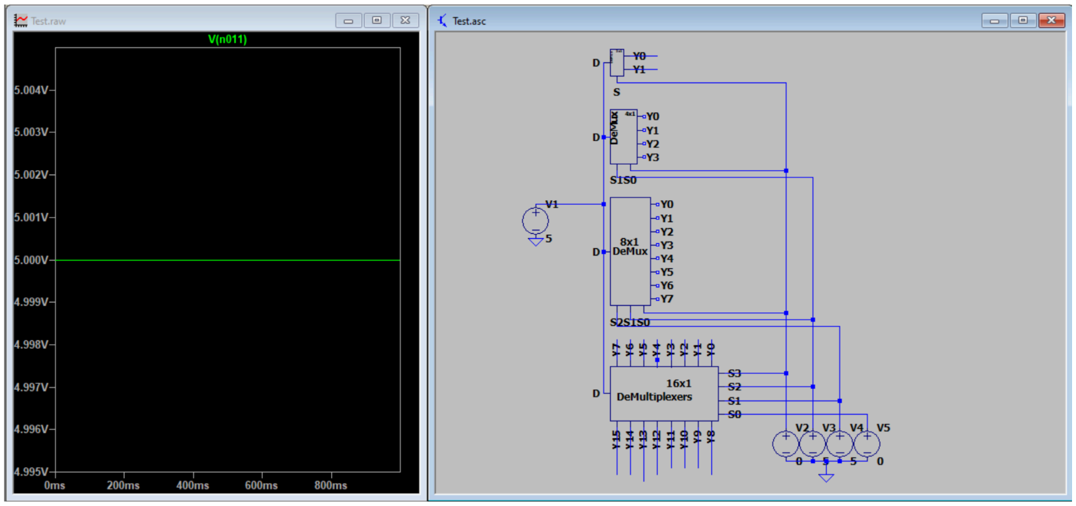
S3 = 0V
S2 = 5V
S1 = 0V
S0 = 5V





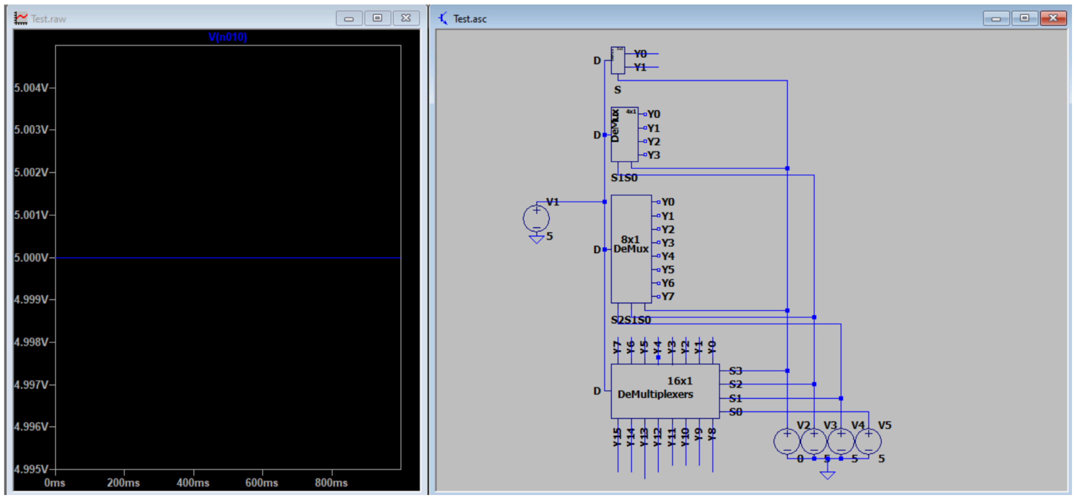
Case 7. Binary code: 0110 ⇒ Output taken from Y_6

S3 = 0V
S2 = 5V
S1 = 5V
S0 = 0V



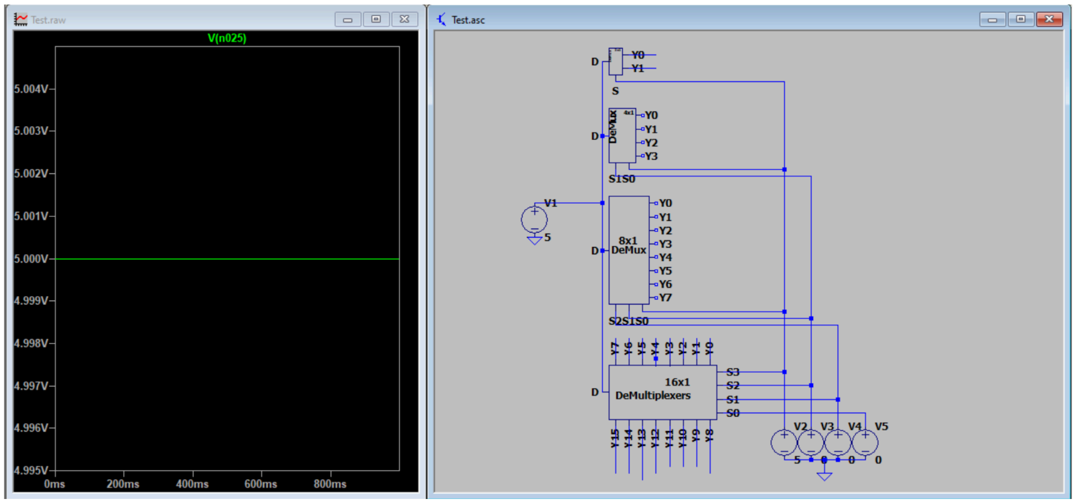
Case 8. Binary code: 0111 ⇒ Output taken from Y_7

S3 = 0V
S2 = 5V
S1 = 5V
S0 = 5V



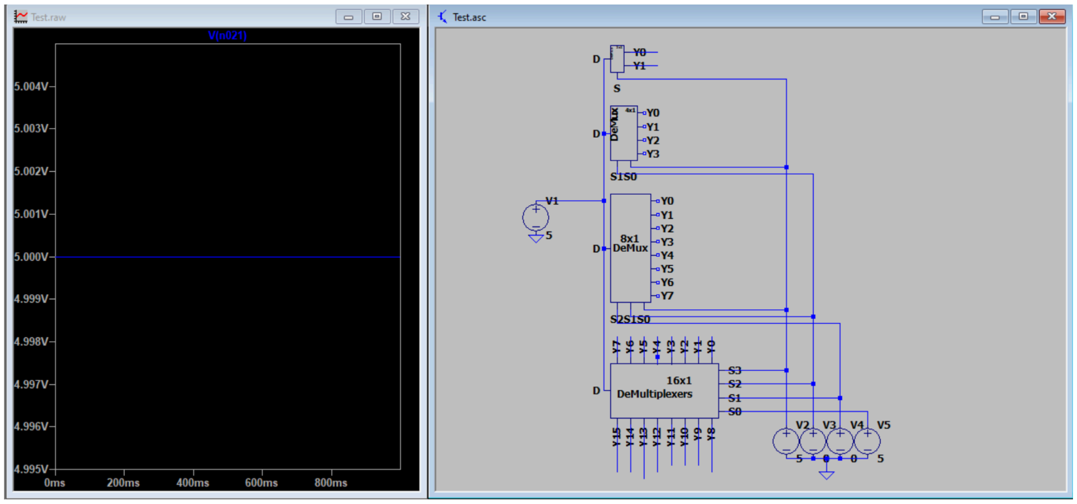
Case 9. Binary code: 1000 $\Rightarrow$ Output taken from Y_8

S3 = 5V
S2 = 0V
S1 = 0V
S0 = 0V



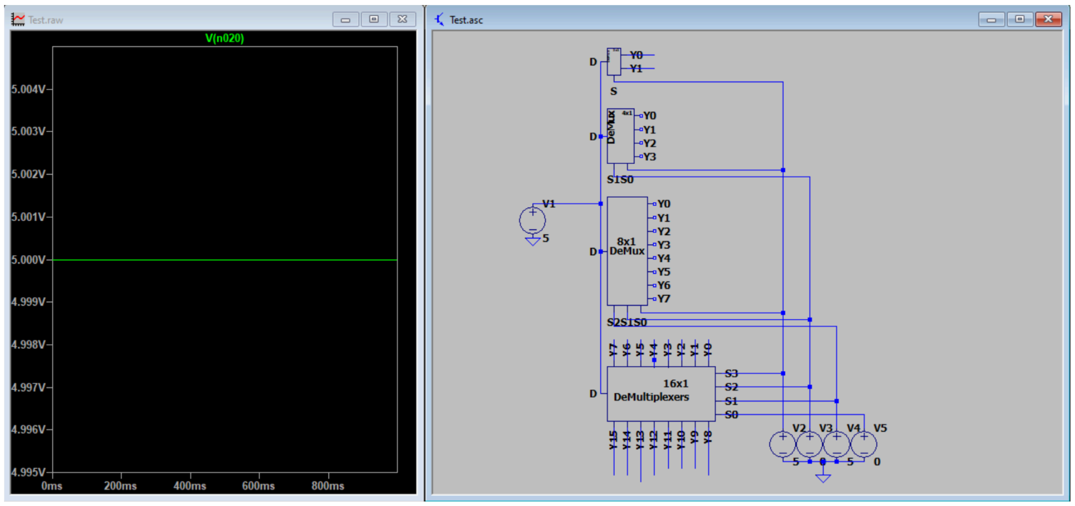
Case 10. Binary code: 1001 $\Rightarrow$ Output taken from Y_9

S3 = 5V
S2 = 0V
S1 = 0V
S0 = 5V



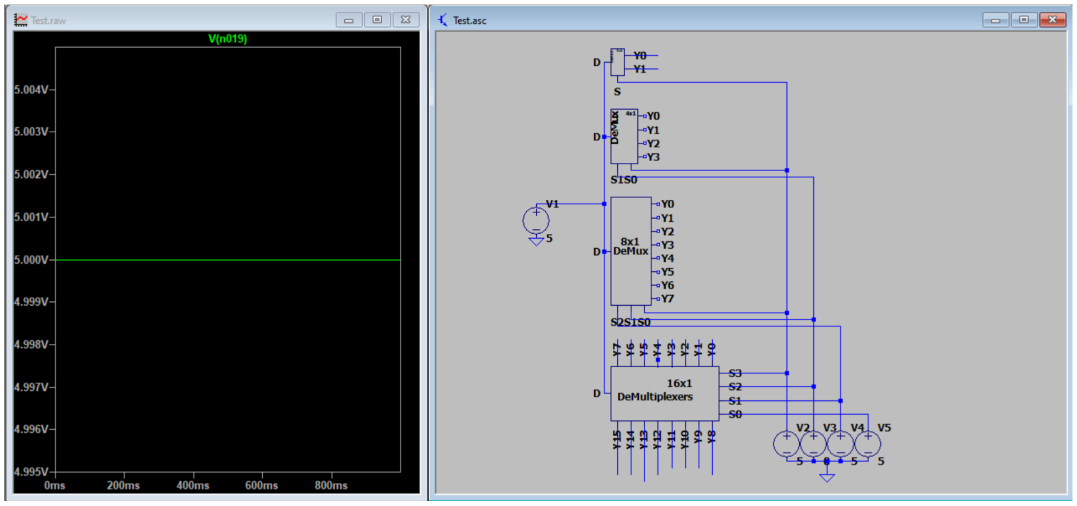
Case 11. Binary code: 1010 ⇒ Output taken from Y_{10}

S3 = 5V
S2 = 0V
S1 = 5V
S0 = 0V



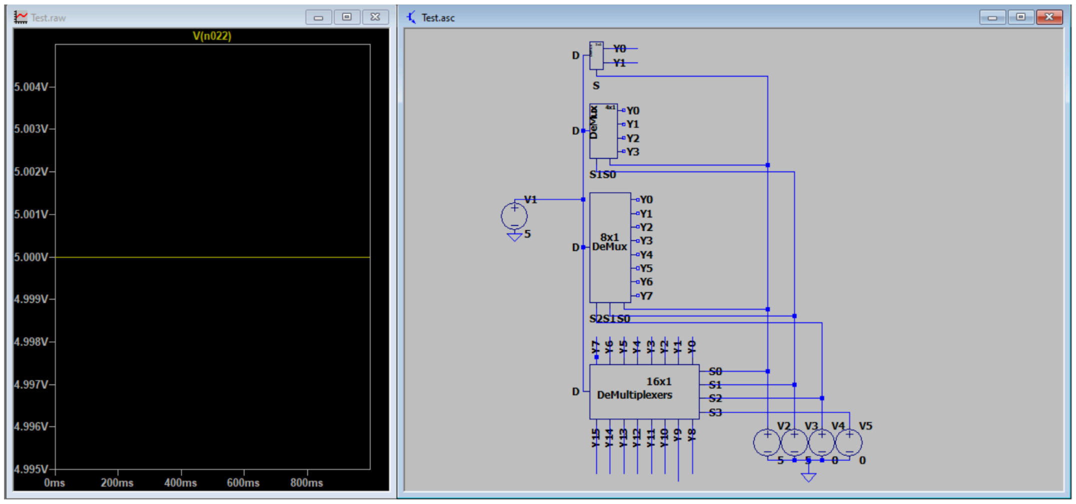
Case 12. Binary code: 1011 ⇒ Output taken from Y_{11}

S3 = 5V
S2 = 0V
S1 = 5V
S0 = 5V



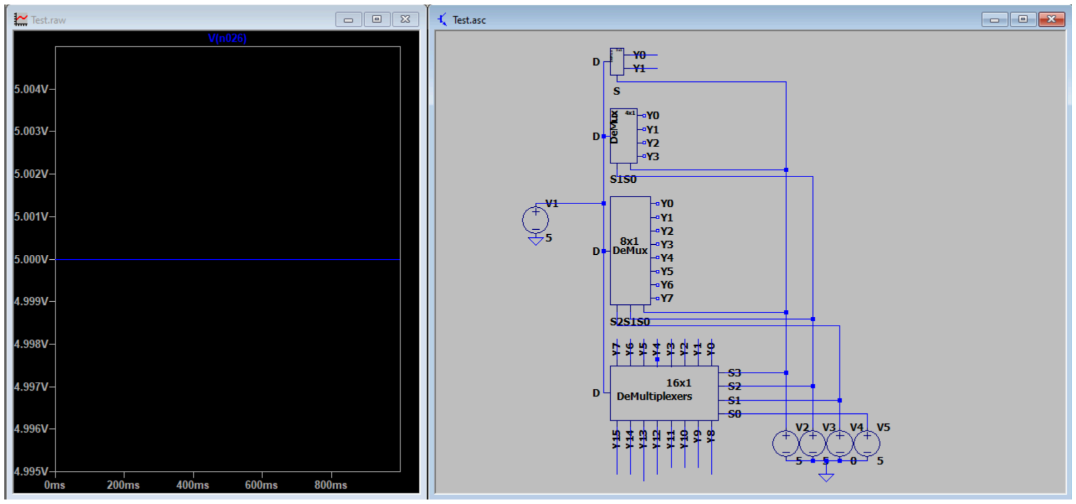
Case 13. Binary code: 1100 ⇒ Output taken from Y_{12}

S3 = 5V
S2 = 5V
S1 = 0V
S0 = 0V



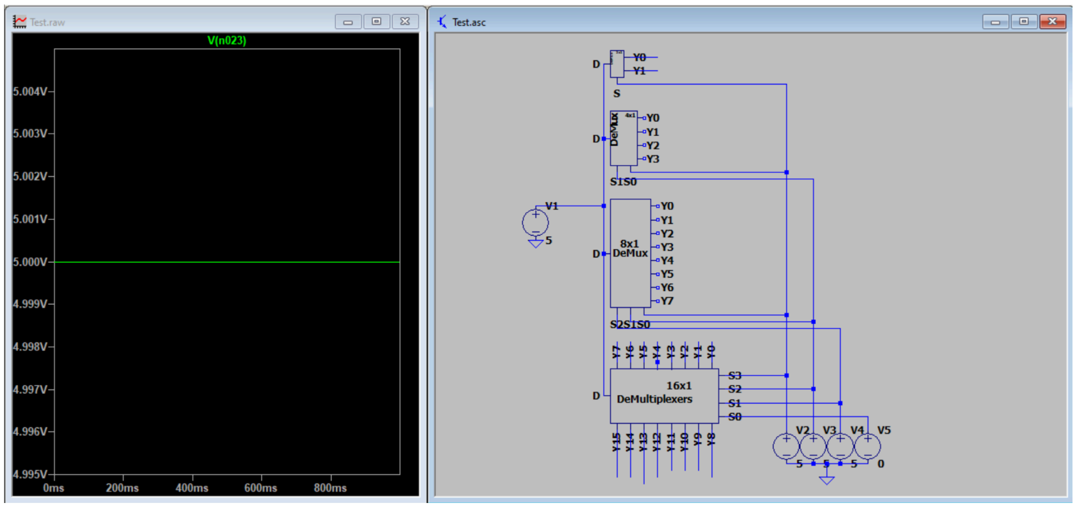
Case 14. Binary code: 1101 ⇒ Output taken from Y_{13}

S3 = 5V
S2 = 5V
S1 = 0V
S0 = 5V



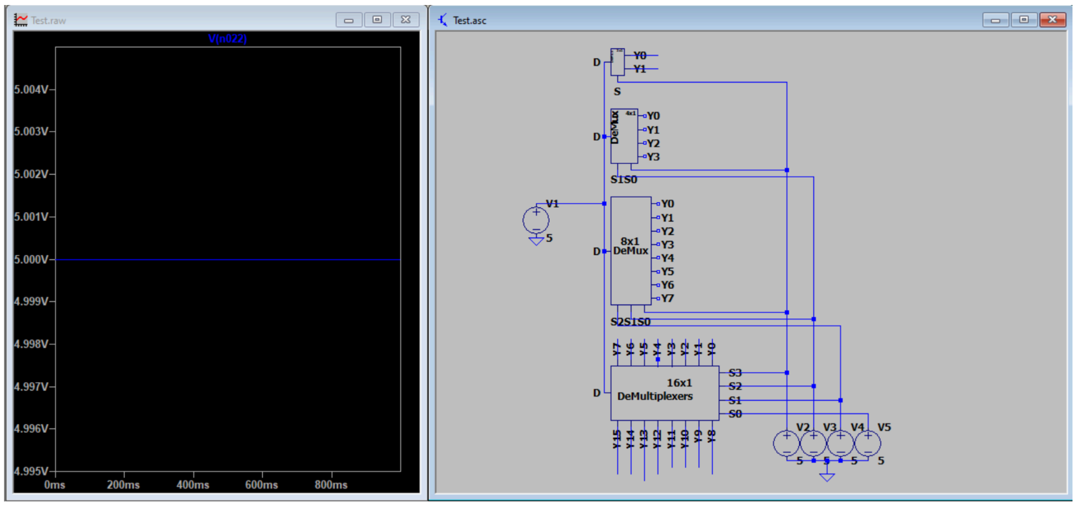
Case 15. Binary code: 1110 ⇒ Output taken from Y_{14}

S3 = 5V
S2 = 5V
S1 = 5V
S0 = 0V



Case 16. Binary code: 1111 ⇒ Output taken from Y_{15}

S3 = 5V
S2 = 5V
S1 = 5V
S0 = 5V



Manual Testing

Using the Test file that already wired

- Unzip and put all the file in to one folder (including all symbols, schematics and the C5_models_LTSPICE.txt)
- Open the Test.asc file.
- Here we can test all the components in this test file that are already wired.
- Change the input voltage to 0 or 5V (which is 0 or 1 in boolean algebra).
- Press Run button double click on the output wire (or any where you want to measure).
- Observe the signal, if it is 5V then it is 1 else if it is 0V (or few nV ~ 0V) then it is 0, this is due to the C5_ model mosfet data and wiring.
- Check it with the truth table to see if it is correct or not.
- Change the input signal and state and repeat the process

Note: We can change the in out signal to pulse and add some delay circuit to eliminate the repeating process, but for the ease in observation we will use DC signal and change the input separately.

Create and use it in new schematic

- Add every file into one directory.
- Create new LT spice.
- Add the needed Spice directive.
- Add the needed Component.
- Give and change the input combination.
- Observe the signal.