DIGITAL CMOS INTEGRATION OF QUAD-COMPUTATIONAL UNIT LOGIC PROCESSOR

EENG-5540

DIGITAL IC DESIGN

Project report

HANUMAN SAGAR BATHULA

Due : 12/06/2023

Under The Guidance of Instructor:

Dr. Gayatri Mehta

Abstract:

This project details the design and validation of a digital CMOS computational unit (CU) optimized for a variety of arithmetic and logical operations including addition, multiplication, comparison, and arithmetic shift right. The design focuses on a 2x2 matrix structure with a decoder, utilizing a 3-bit interface for input and output operations. The computational unit's functionality is extended with a 4:1 multiplexer that selects the output based on specified control signals. Detailed circuit schematics, symbol drawings, and selection tables are provided to illustrate the design process. Test circuits validate the CU's operations, ensuring that the outputs match the expected results for various input conditions. This verification confirms the CU's reliability and accuracy for potential application in more complex digital systems.

Table of Contents

PART 1 CU WITHOUT DECODER: 5
Design Explanation: 5
CU circuit drawing: 6
CU symbol schematic: 7
CU symbol drawing: 7
Selection Table: 8
Test Calculation Table: 8
CU test circuits: 9
Conclusion: 12
PART 2 OVERALL DEVICE: 13
Design Explanation: 13
Overall device circuit drawing: 14
Overall device symbol schematic: 15
Overall device symbol drawing: 15
Selection Table: 16
Test Calculation Table: 16
Overall device test circuits: 17
Conclusion: 23
PART 2 CU WITH DECODER: 24
Design Explanation: 24
Circuit drawing: 25
Symbol schematic: 26
Symbol drawing: 26
Selection Table: 27
Test Calculation Table: 27
Test circuits: 27
Conclusion: 34
SUBCOMPONENTS (OPERATION CIRCUITS): 35
ADDER: 35
MULTIPLIER: 38
GREATER THAN: 41
ARITHMETIC SHIFT RIGHT: 46
ADDER ENABLE: 50
MULTIPLIER ENABLE: 54

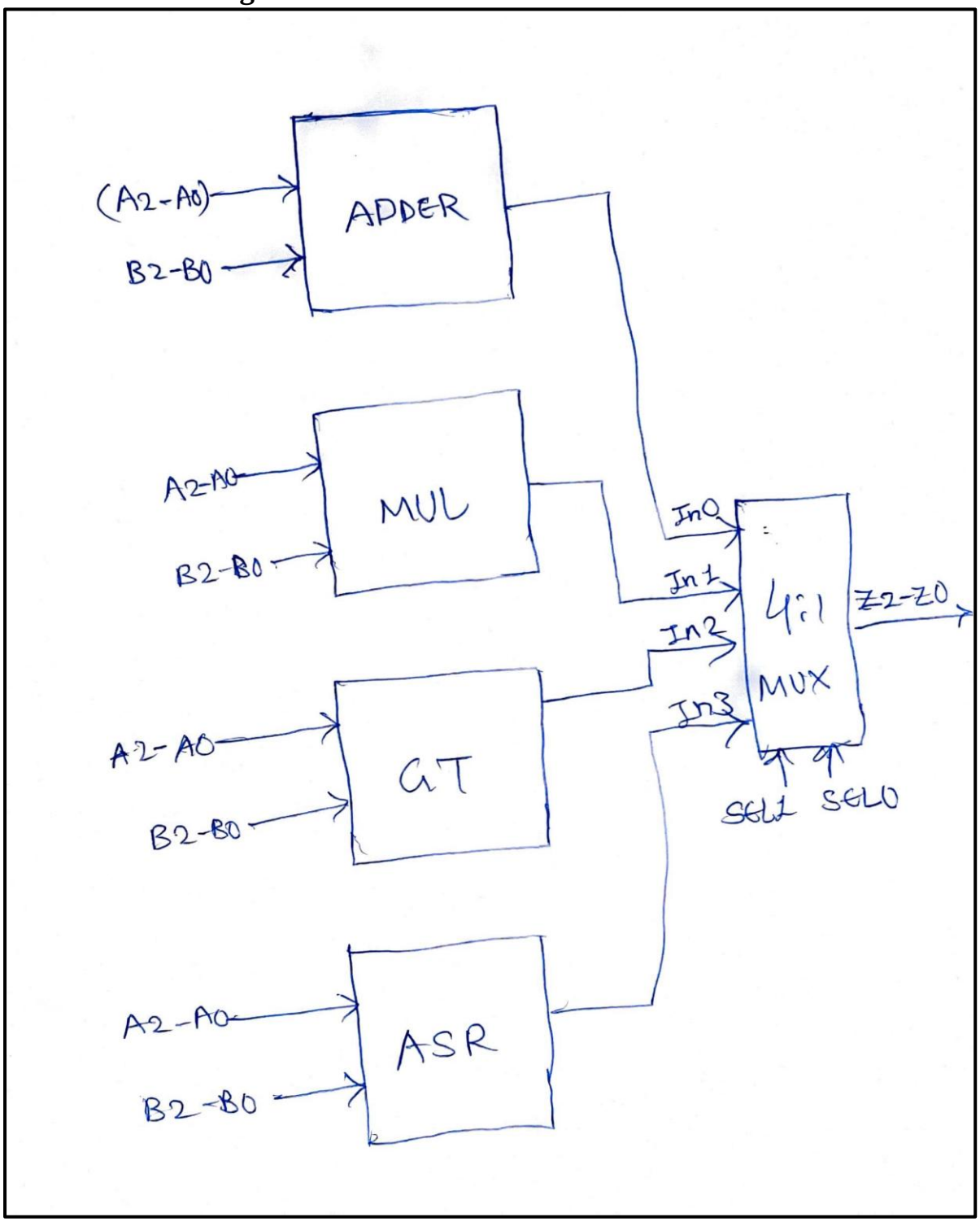
GREATER THAN ENABLE: 57
ARITHMETIC SHIFT REGISTER ENABLE: 62
GATES AND SUBCOMPONENTS: 66
NOT GATE: 66
AND GATE: 69
OR GATE: 72
XOR GATE: 75
XNOR GATE: 77
HALF ADDER: 79
FULL ADDER: 82
MUX 2:1: 85
MUX 4:1: 88
3-BIT MUX 4:1: 91
MUX 8:1: 95
DECODER 1:2: 98
DECODER2:4: 101

PART 1 CU WITHOUT DECODER:

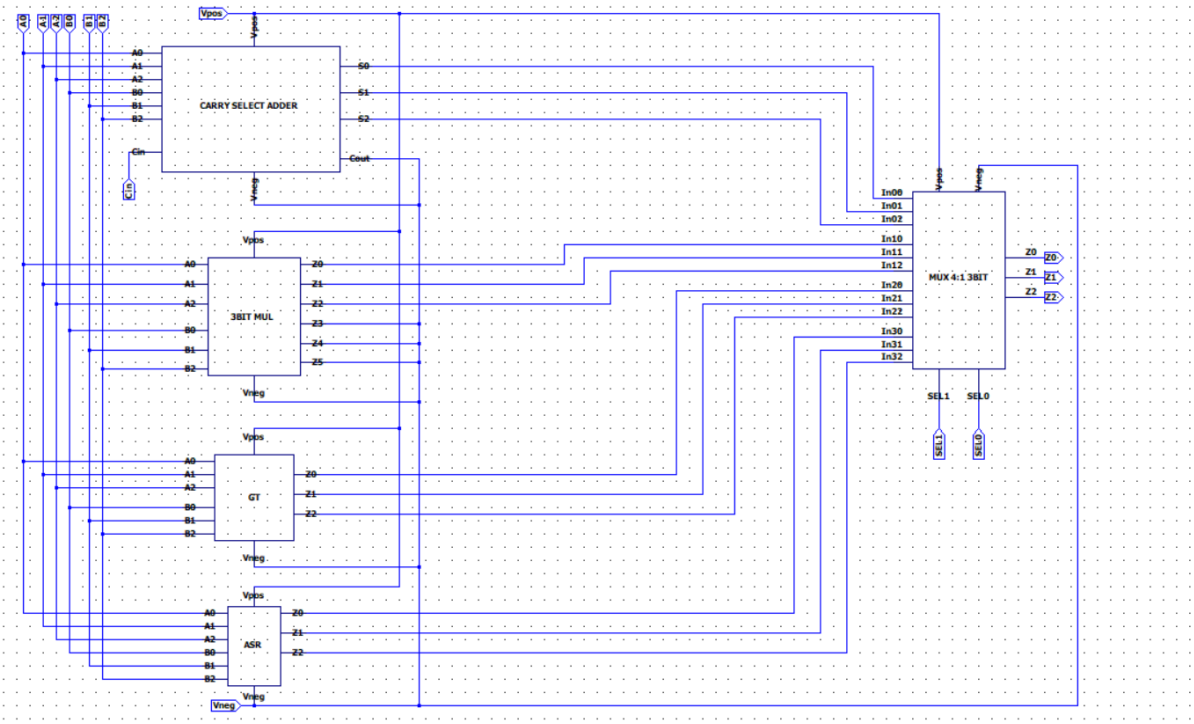
Design Explanation:

1. The below designed Computational Unit (CU) can perform 3bit Addition(ADD), Multiplication(MUL), Greater than(GT) and Arithmetic Shift Right(ASR) Operations.
2. A(A2-A0) and B(B2-B0) are two three-bit inputs for the for the four above mentioned operations.
3. The 3bit output from every operation is fed to the inputs for the 3bit 4:1 Multiplexer (MUX).
4. The Carry-in taken to the 3bit Adder(Carry Select Adder) and the Cout is grounded. SUM (MSB.S2-S0) is given to (MSB.In02-In00) MUX input.
5. The three MSB bits of MUL is grounded (Z5-Z3) and Z2 as MSB (Z2-Z0) is given to the (MSB.In12-In10) MUX input.
6. The designed 3bit greater than (GT) gives 3bits High output(111) when A<=B otherwise the output is A(A2-A0). The output (Z2-Z0) from GT is given to the (MSB.In22-In20) MUX input.
7. The ASR is used to arithmetic shift right 3bit input A by 3bit input shift B. The output of ASR (MSB.Z2-Z0) is given to the (MSB.In32-In30) MUX input.
8. SEL1(MSB) and SEL0 are the selection inputs for the 4:1 MUX to select which operation output is to take at the MUX output (MSB.Z2-Z0)
9. Output from 3bit 4:1 MUX is final output of the below CU.

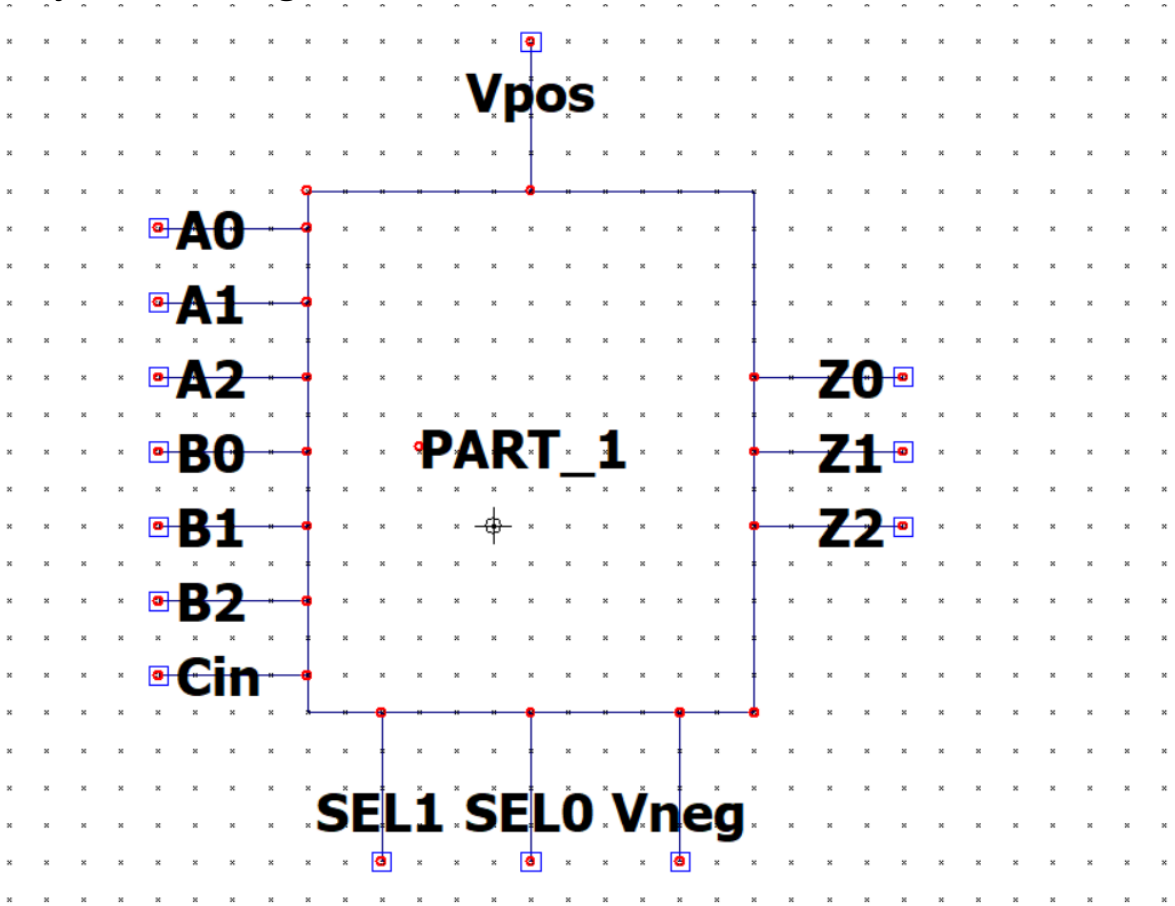
CU circuit drawing:



CU symbol schematic:



CU symbol drawing:



Selection Table:

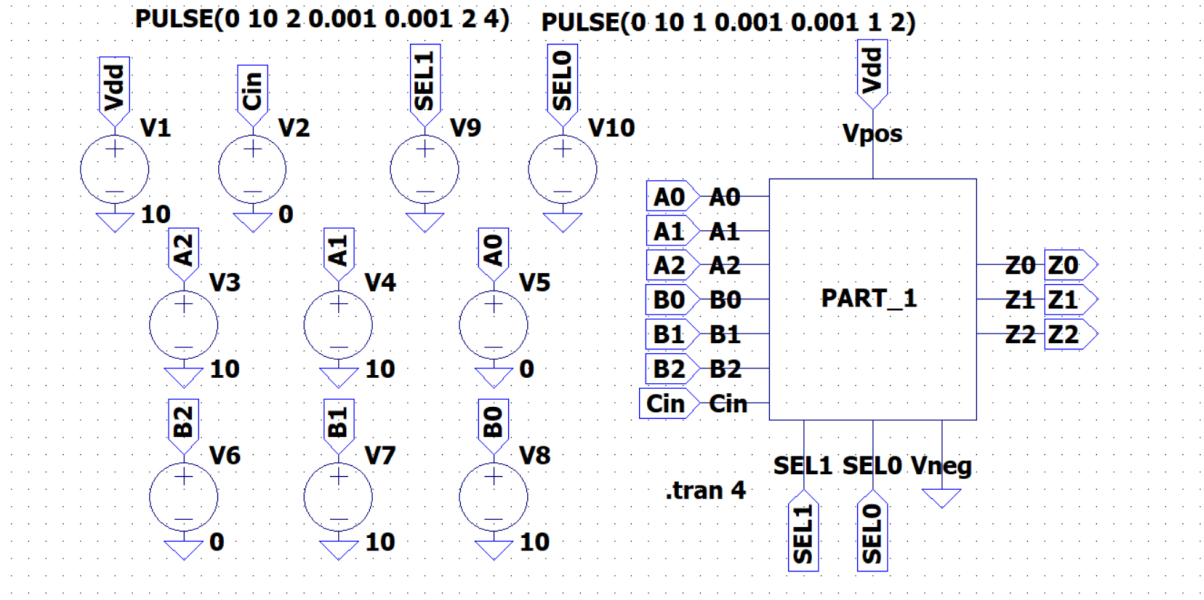
S.no	SEL1 MSB	SEL0 LSB	Output from Operation
1	0	0	ADD
2	0	1	MUL
3	1	0	GT
4	1	1	ASR

Test Calculation Table:

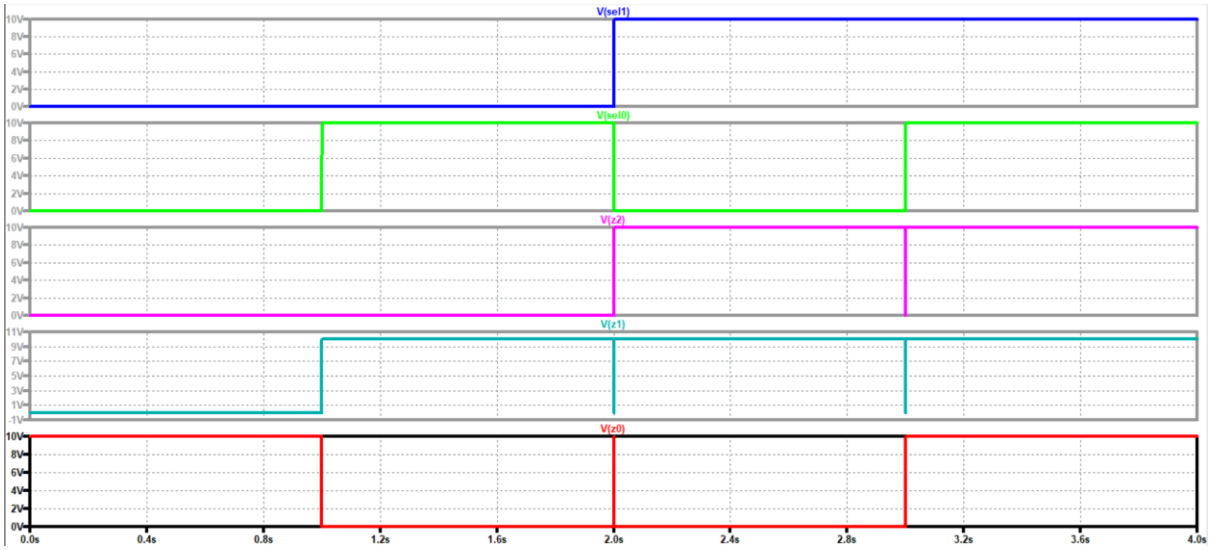
S.no	A	B	SEL1 MSB	SEL0 LSB	Operation	Output (Z2-Z0)
1	110	011	0	0	ADD	001
	110	011	0	1	MUL	010
	110	011	1	0	GT	110
	110	011	1	1	ASR	111
2	011	110	0	0	ADD	001
	011	110	0	1	MUL	010
	011	110	1	0	GT	111
	011	110	1	1	ASR	000
3	101	001	0	0	ADD	110
	101	001	0	1	MUL	101
	101	001	1	0	GT	101
	101	001	1	1	ASR	110
4	110	110	0	0	ADD	100
	110	110	0	1	MUL	100
	110	110	1	0	GT	111
	110	110	1	1	ASR	111

CU test circuits:

Test-1: A(A2-A0)=110; B(B2-B0)=011;

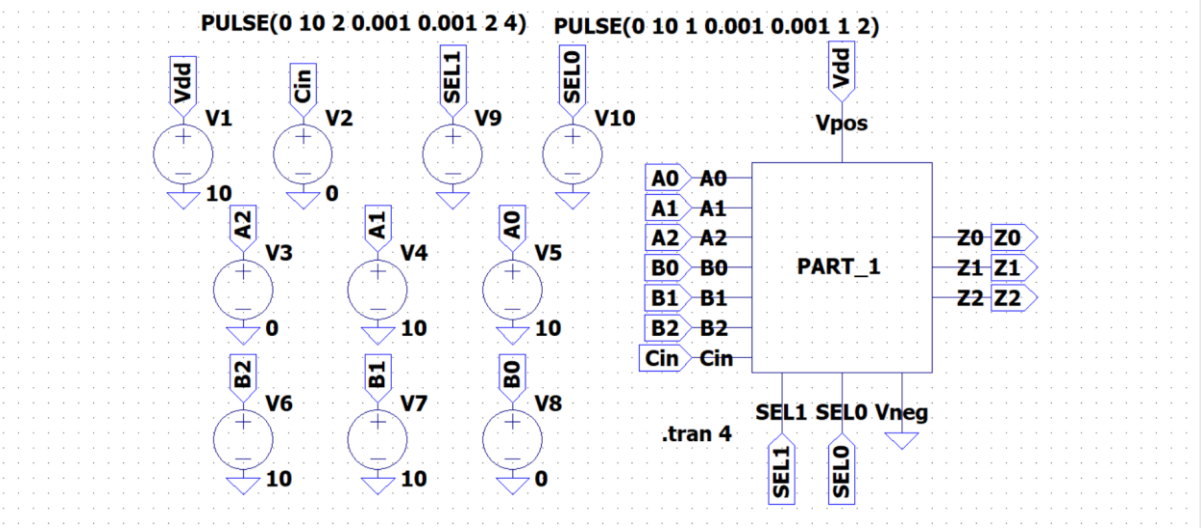


Test-1 Waveform:

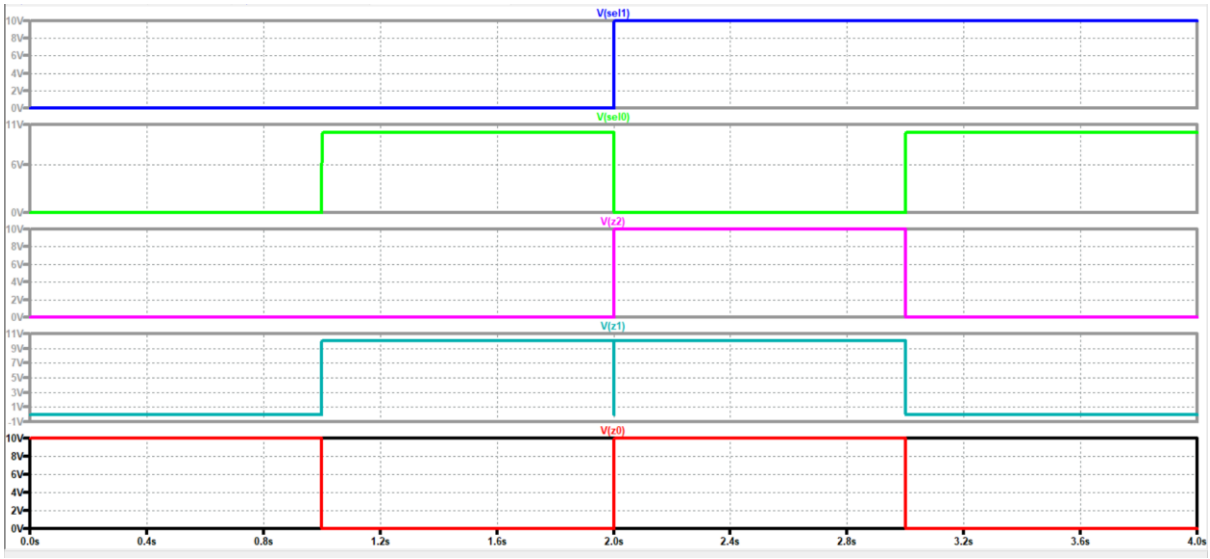


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	0	0	ADD	001	001
	110	011	0	1	MUL	010	010
	110	011	1	0	GT	110	110
	110	011	1	1	ASR	111	111

Test-2: A(A2-A0)=011; B(B2-B0)=11

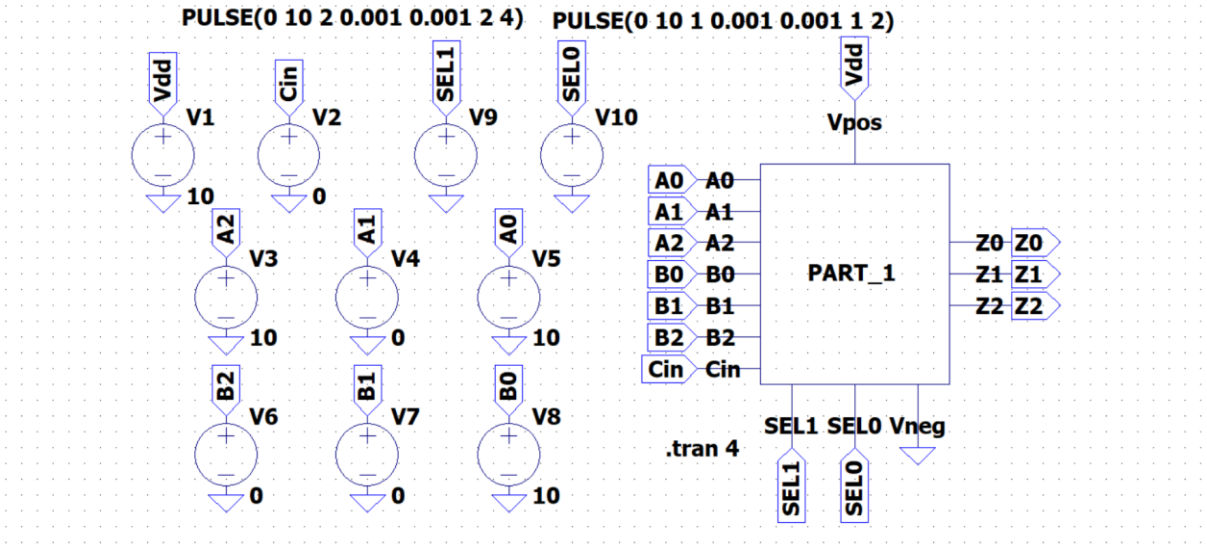


Test-2 Waveform:

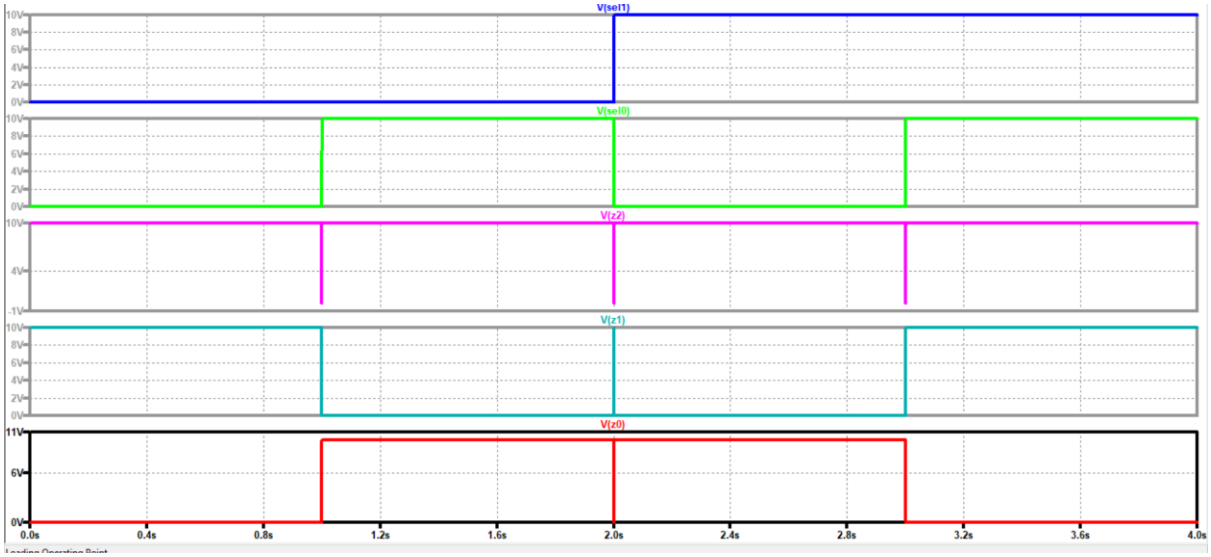


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
2	011	110	0	0	ADD	001	001
	011	110	0	1	MUL	010	010
	011	110	1	0	GT	111	111
	011	110	1	1	ASR	000	000

Test-3: A(A2-A0)=101; B(B2-B0)=001

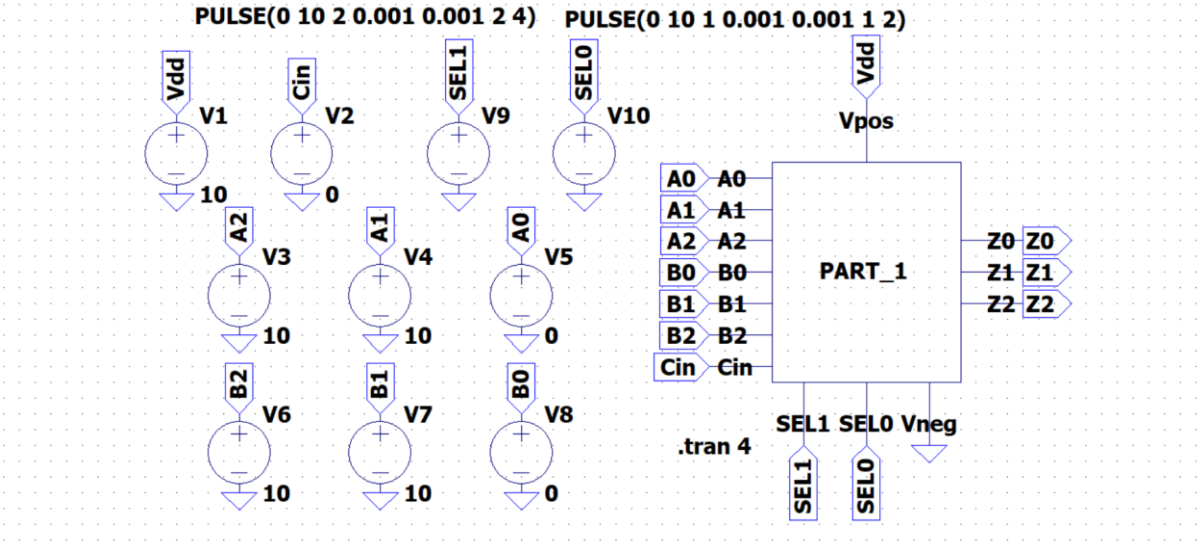


Test-3 Waveform:

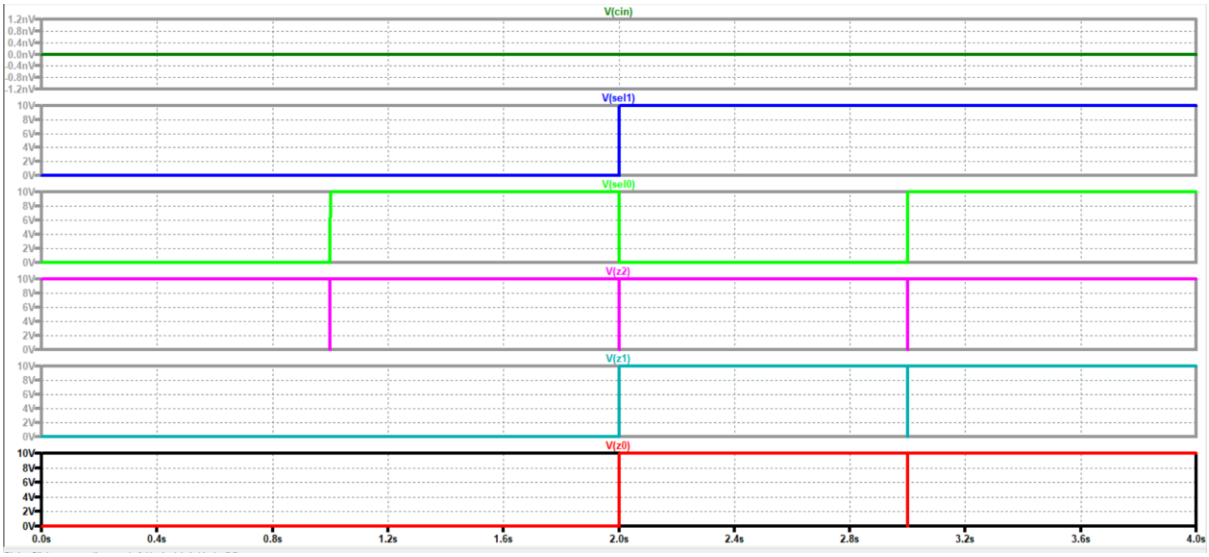


S.no	A (A2-A0)	B (B2-B0)	SEL11 MSB	SEL0 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
3	101	001	0	0	ADD	110	110
	101	001	0	1	MUL	101	101
	101	001	1	0	GT	101	101
	101	001	1	1	ASR	110	110

Test-4: A(A2-A0)=110; B(B2-B0)=110



Test-4 Waveform:



S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
3	110	110	0	0	ADD	100	100
	110	110	0	1	MUL	100	100
	110	110	1	0	GT	111	111
	110	110	1	1	ASR	111	111

Conclusion:

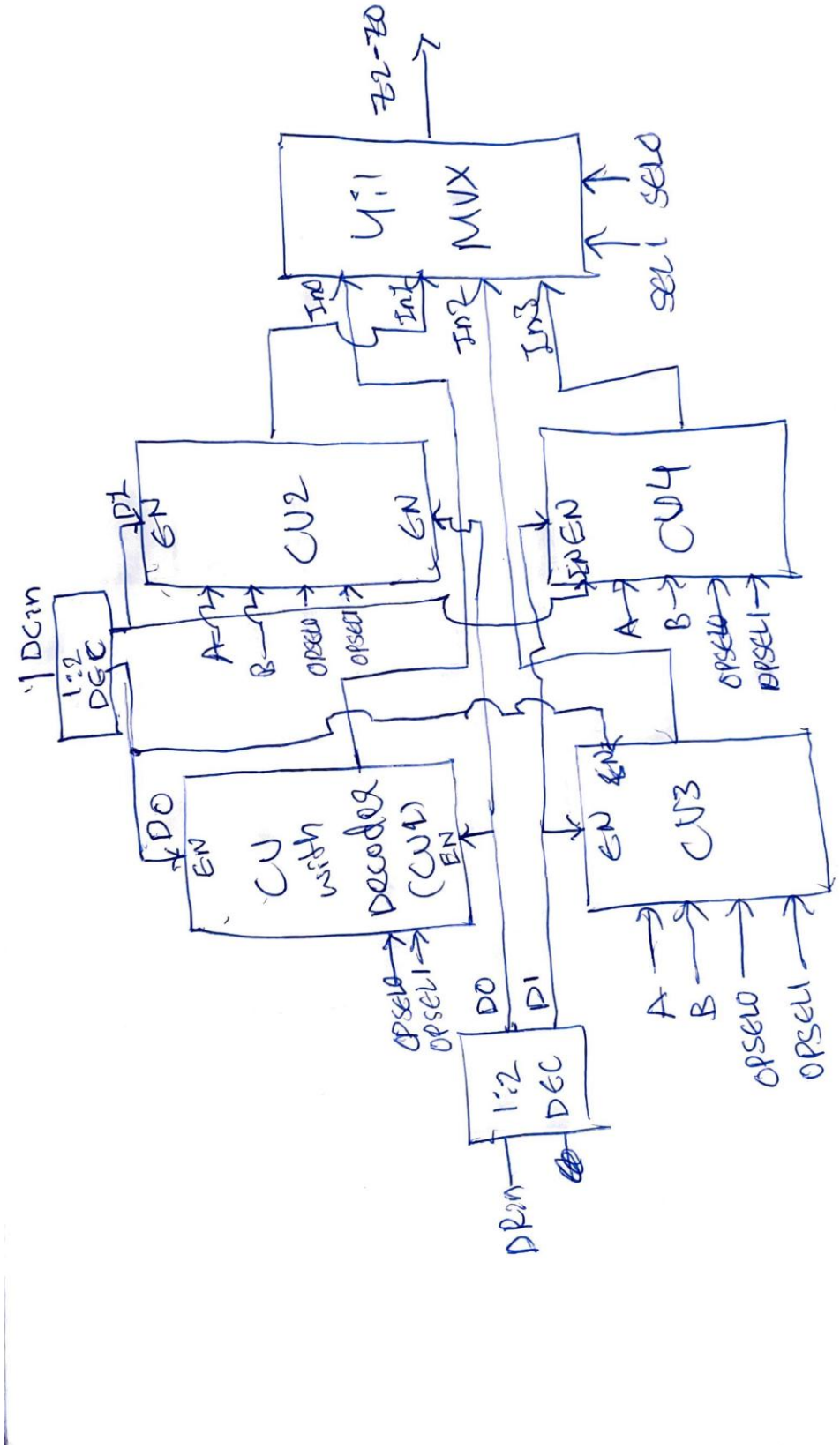
The CU without decoder calculation outputs and the output waveforms are matching exactly from the comparison table for all the operations. Therefore, the designed CU with four Operation (ADD,MUL,GT and ASR) is working properly and verified.

PART 2 OVERALL DEVICE:

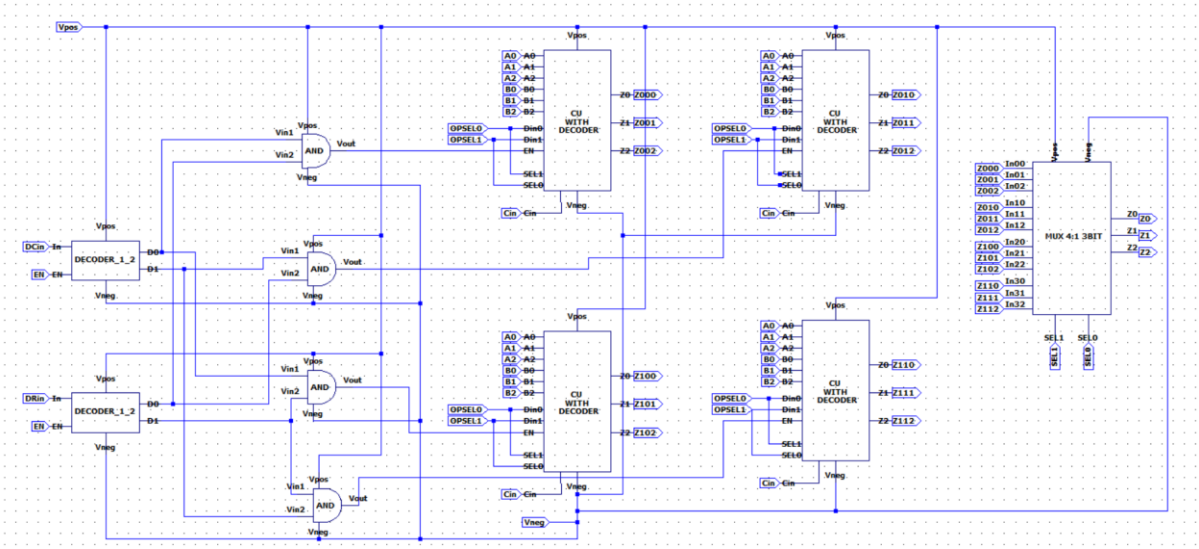
Design Explanation:

1. The Overall design has four CUs each CU can perform 3bit Addition(ADD), Multiplication(MUL), Greater than(GT) and Arithmetic Shift Right(ASR) Operation.
2. All the above CUs have HIGH enable pin to operate.
3. A(A2-A0) and B(B2-B0) are two three-bit inputs for all four CUs.
4. The design has 2 1:2 decoder to select one of the CU at time to perform a particular operation. And the output of each CU is given to the 3bit 4:1 MUX. DRin and DCin are decoder inputs to select the CU. The Selection of CUs and the output is given below in the selection table section.
5. Output from CU1 MSB.Z002-Z001 is given to the MSB.In02-In00, output from CU2 MSB.Z002-Z001 is given to the MSB.In12-In10, output from CU3 MSB.Z002-Z001 is given to the MSB.In22-In20, output from CU1 MSB.Z002-Z001 is given to the MSB.In32-In30. SEL1(MSB) and SEL0 used to select the output from the CUs to take the final output.
6. In each CU there is a 2:4 Decoder to enable one of the operations at a time and disable all the other operations. D0,D1,D2 and D3 are connected to the ADD, MUL, GT and ASR respectively in CU to enable one of the operations according to the 2:4 decoder truth table.
7. Here OPSEL0(MSB) and OPSEL1 are used to select the operations in the CUs. OPSEL0(MSB) and OPSEL1 is also connected SEL1(MSB) and SEL0 respectively to the CUs internal 3bit 4:1 MUXes to select the output from the operation. So, I have used same selection for the 2:4 decoder and 4:1 MUX.

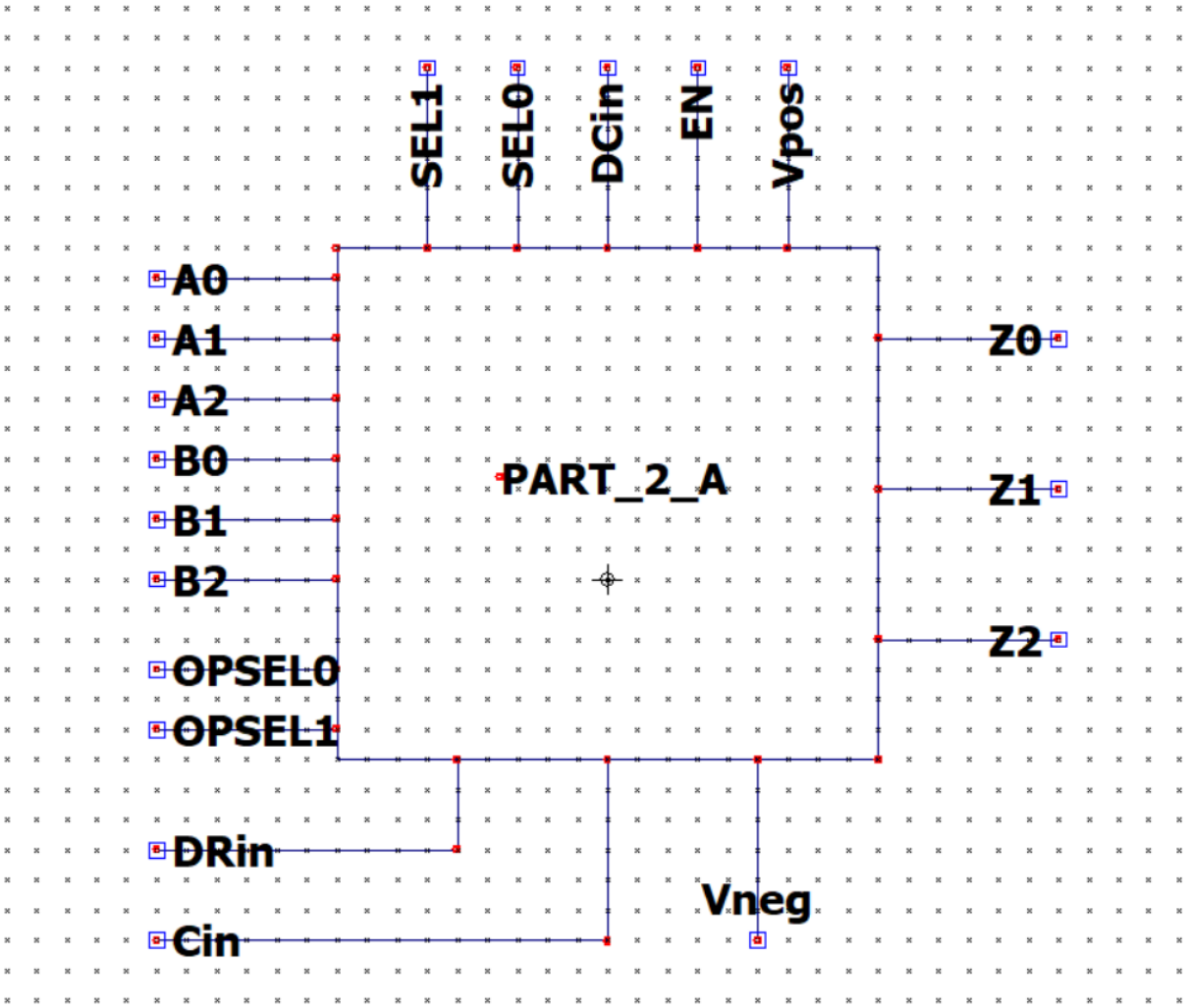
Overall device circuit drawing:



Overall device symbol schematic:



Overall device symbol drawing:



Selection Table:

S.no	DRin MSB	DCin LSB	CU Selection
1	0	0	CU1
2	0	1	CU2
3	1	0	CU3
4	1	1	CU4

S.no	OPSEL0 MSB	OPSEL1 LSB	Operation Selection
1	0	0	ADD
2	0	1	MUL
3	1	0	GT
4	1	1	ASR

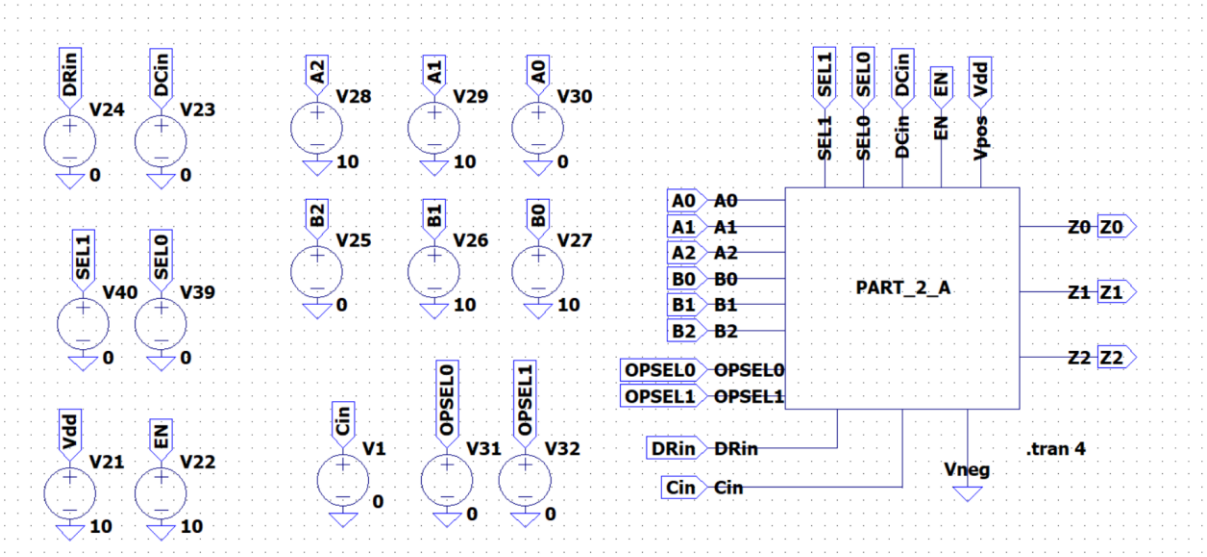
S.no	SEL1 MSB	SEL0 LSB	Output Selection from
1	0	0	CU1
2	0	1	CU2
3	1	0	CU3
4	1	1	CU4

Test Calculation Table:

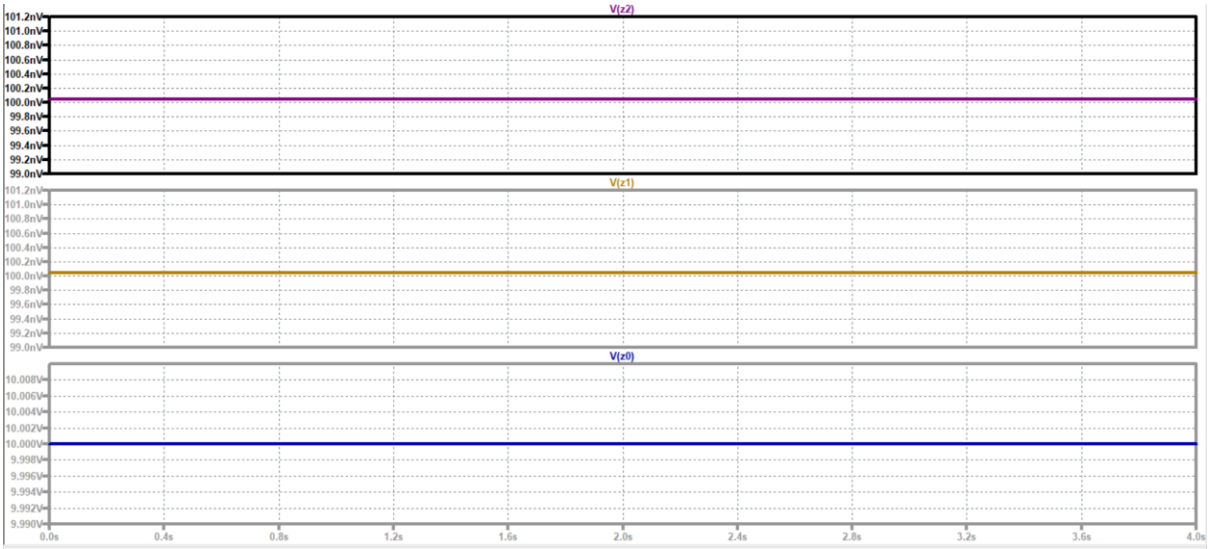
S.no	A (A2-A0)	B (B2-B0)	DRin MSB	DCin LSB	SEL1 MSB	SEL0 LSB	CU	OPSEL0 MSB	OPSEL1 LSB	Operation	Output (Z2-Z0)
1	110	011	0	0	0	0	CU1	0	0	ADD	001
2	110	011	0	1	0	1	CU2	0	1	MUL	010
3	110	011	1	0	1	0	CU3	1	0	GT	110
4	110	011	1	1	1	1	CU4	1	1	ASR	111
5	011	110	0	0	0	0	CU1	1	0	GT	111
6	101	001	0	1	0	1	CU2	1	1	ASR	110
7	011	011	1	1	1	1	CU3	1	0	GT	111

Overall device test circuits:

Test-1: A=110; B=011; DRin=0; DCin=0; SEL1=0; SEL0=0; OPSEL0=0; OPSEL1=0.

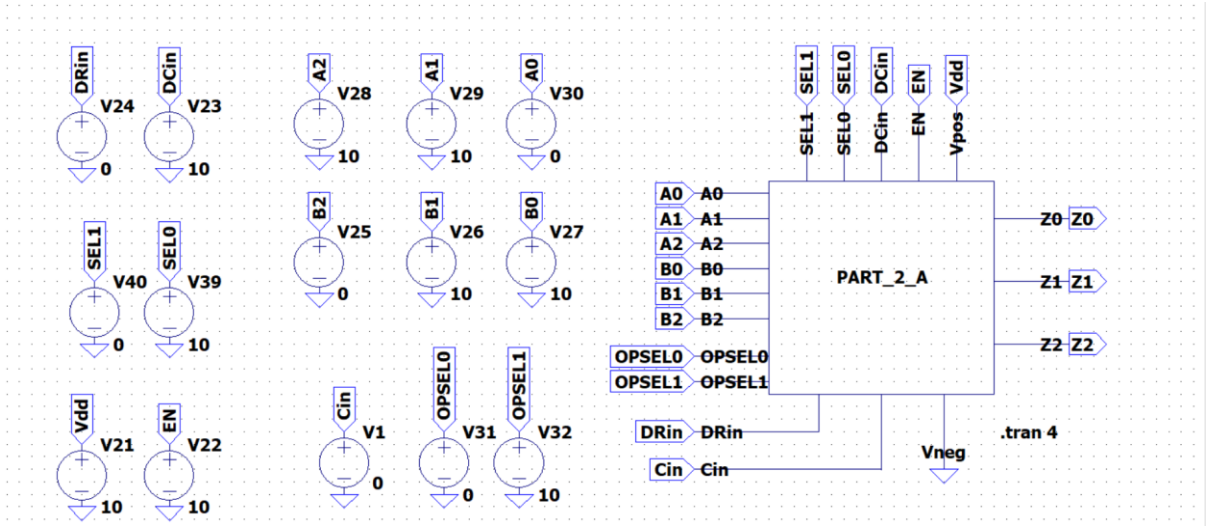


Test-1 Waveforms:

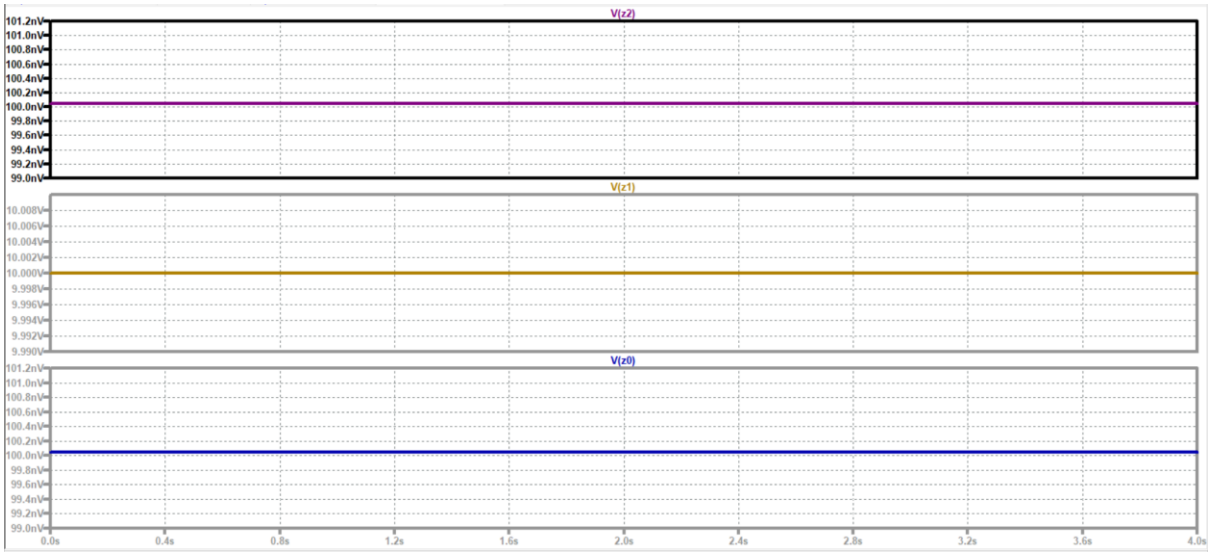


S.no	A (A2-A0)	B (B2-B0)	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	ADD	001	001

Test-2: A=110; B=011; DRin=0; DCin=1; SEL1=0; SEL0=1; OPSEL0=0; OPSEL1=1.

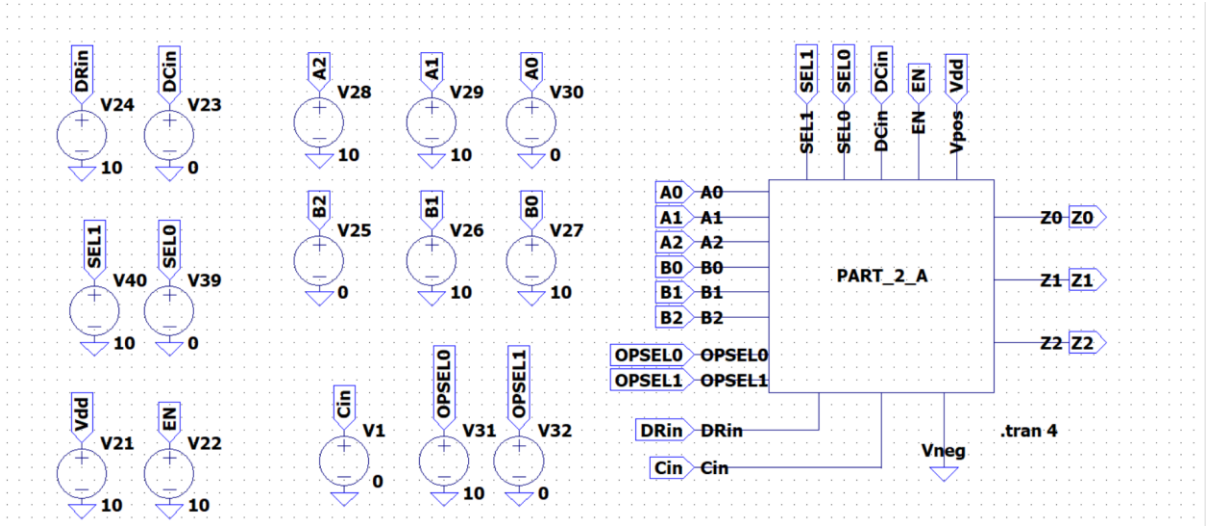


Test-2 Waveforms:

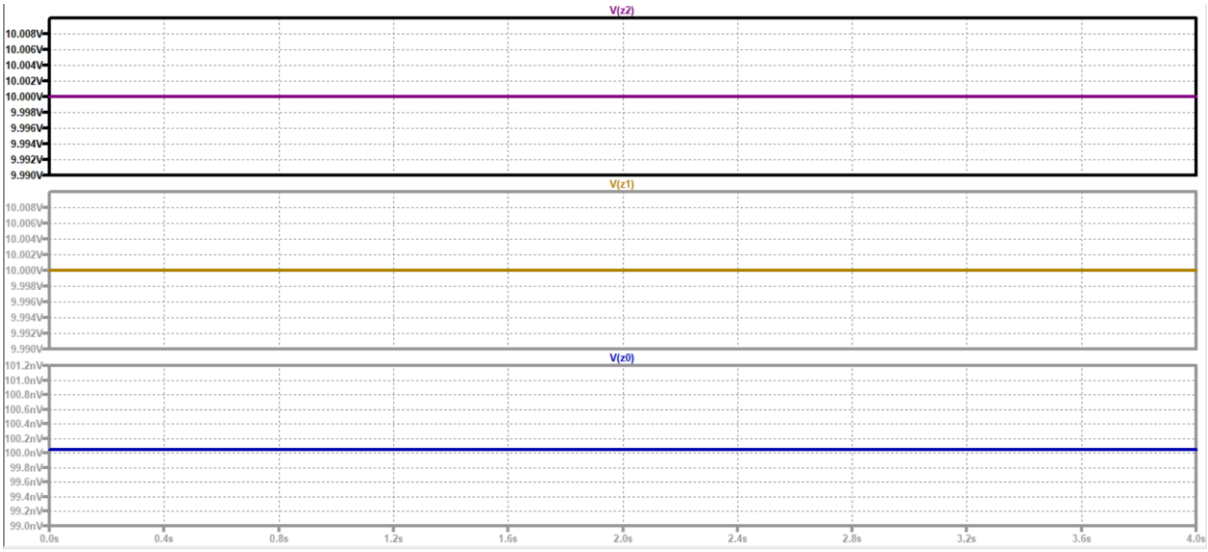


S.no	A (A2-A0)	B (B2-B0)	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	MUL	010	010

Test-3: A=110; B=011; DRin=1; DCin=0; SEL1=1; SEL0=0; OPSEL0=1; OPSEL1=0.

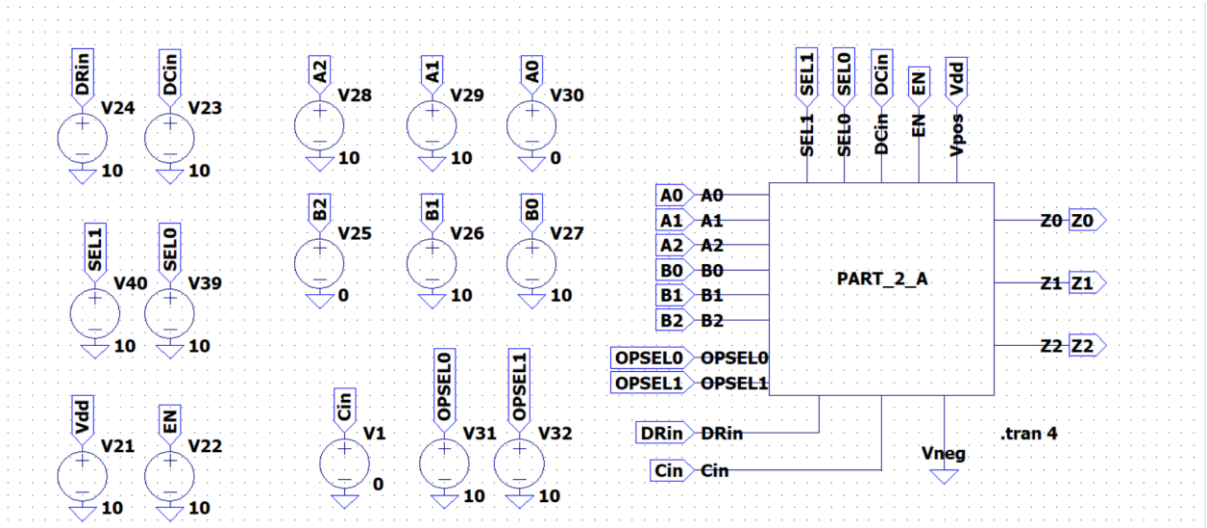


Test-3 Waveforms:

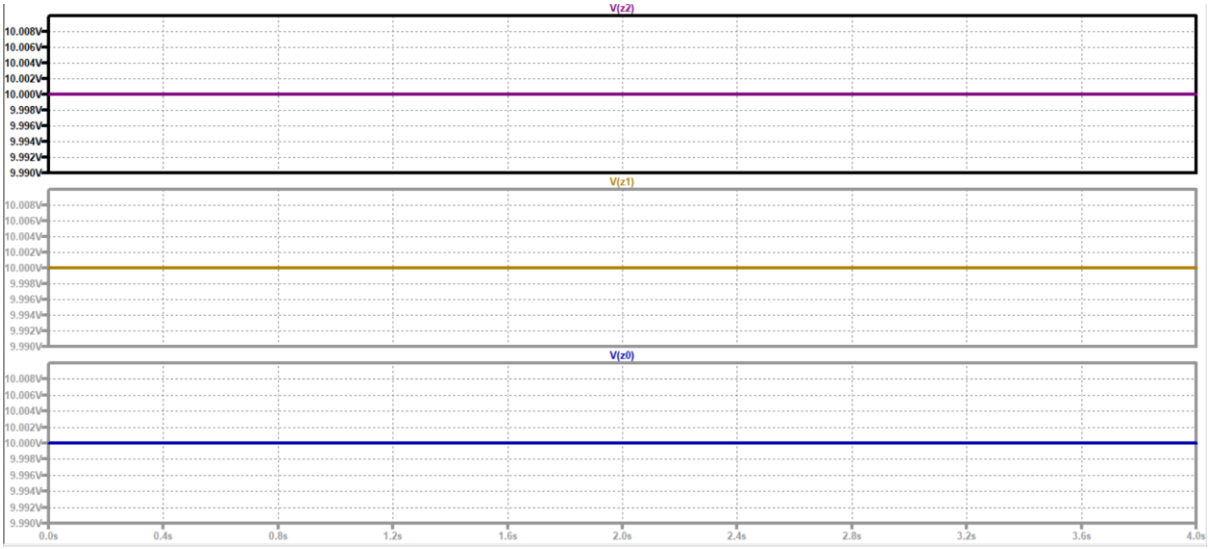


S.no	A (A2-A0)	B (B2-B0)	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	GT	110	110

Test-4: A=110; B=011; DRin=1; DCin=1; SEL1=1; SEL0=1; OPSEL0=1; OPSEL1=1.

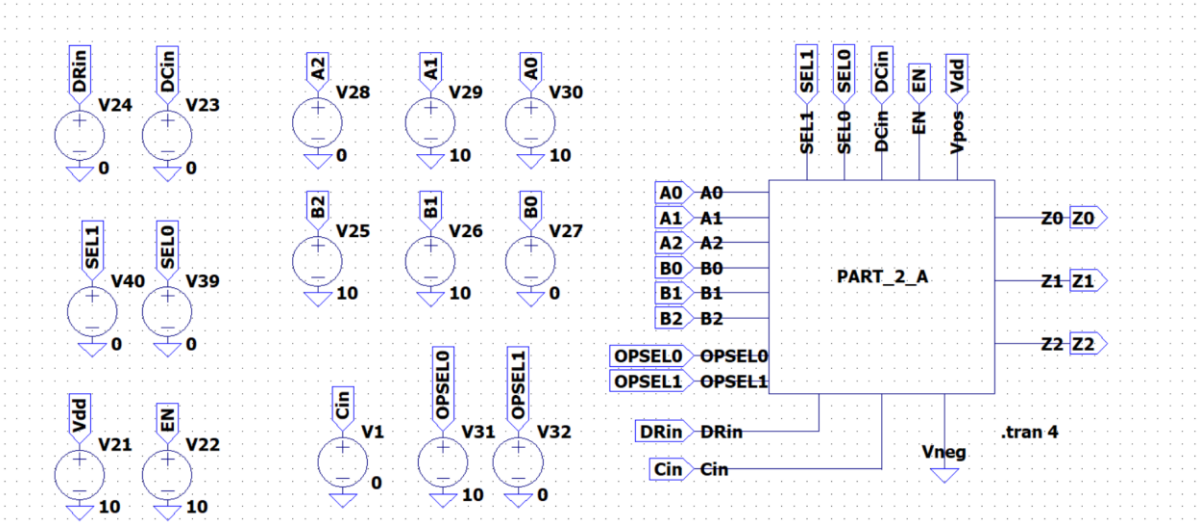


Test-4 Waveforms:

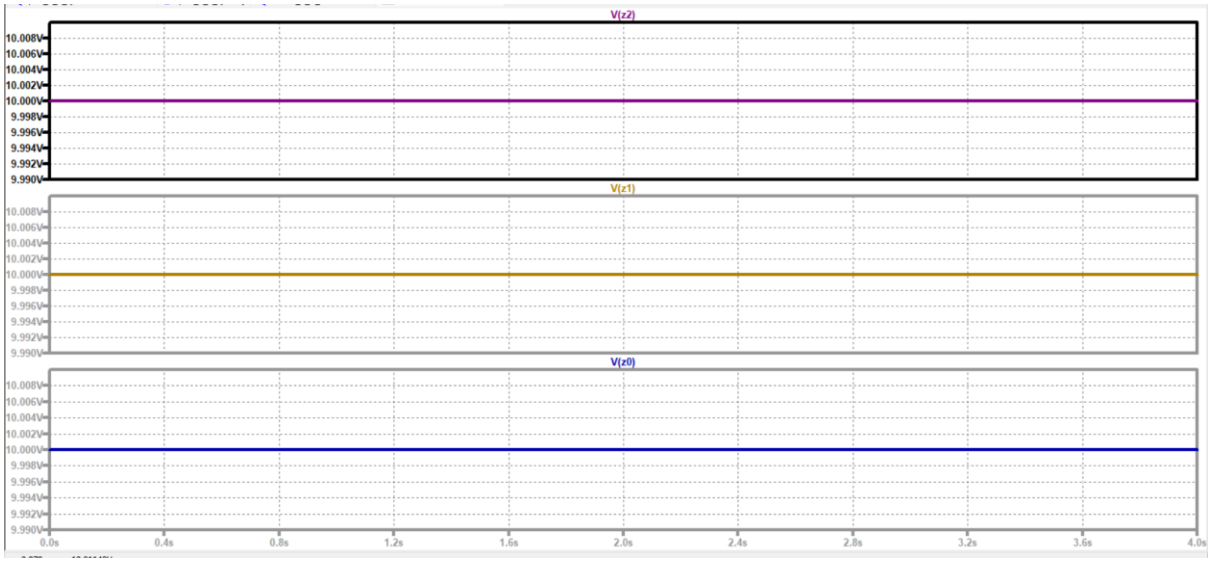


S.no	A (A2-A0)	B (B2-B0)	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	ASR	111	111

Test-5: A=011; B=110; Drin=0; DCin=0; SEL1=0; SEL0=0; OPSEL0=1; OPSEL1=0.

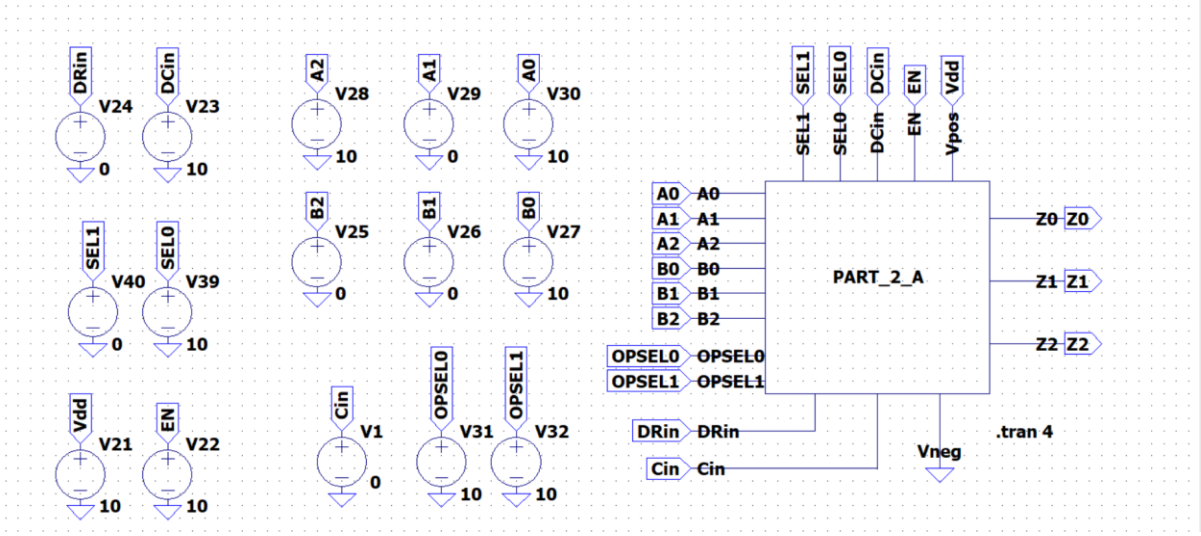


Test-5 Waveforms:

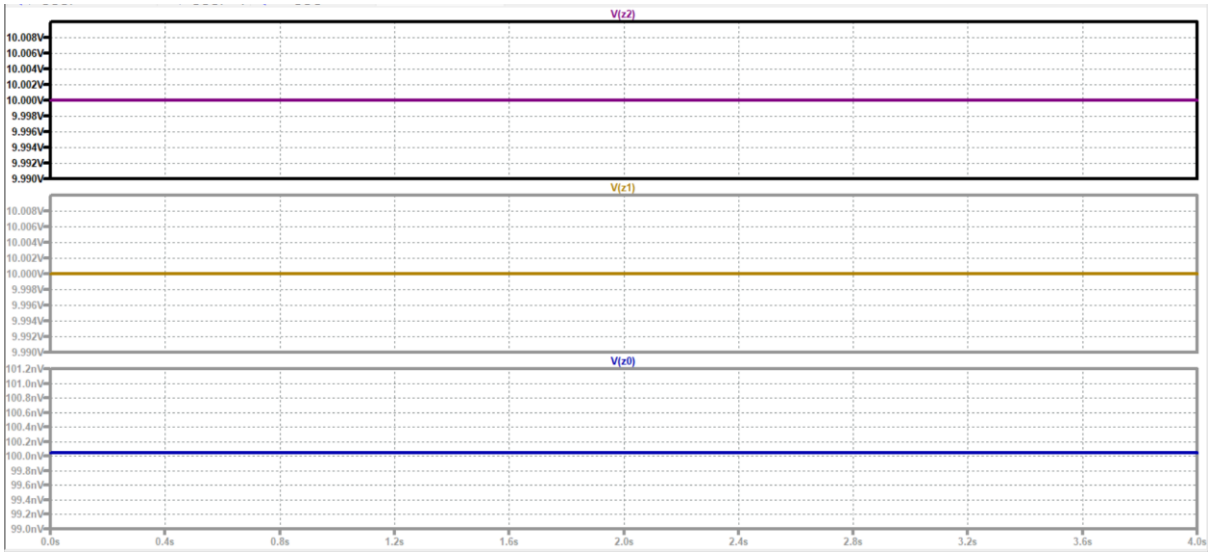


S.no	A (A2-A0)	B (B2-B0)	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	011	110	GT	111	111

Test-6: A=101; B=001; Drin=0; DCin=1; SEL1=0; SEL0=1; OPSEL0=1; OPSEL1=1.

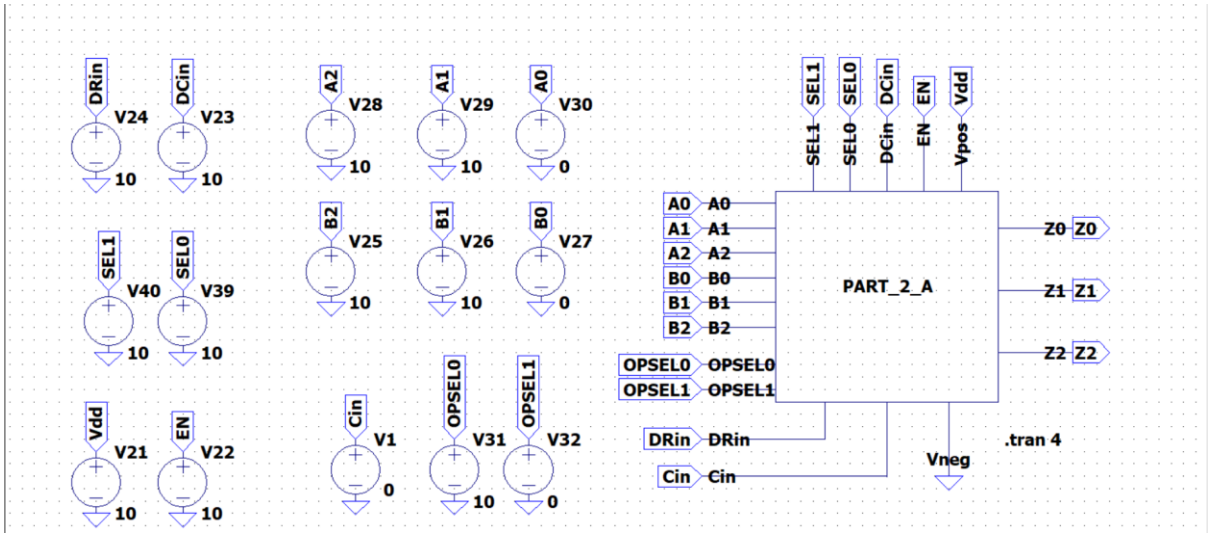


Test-6 Waveforms:

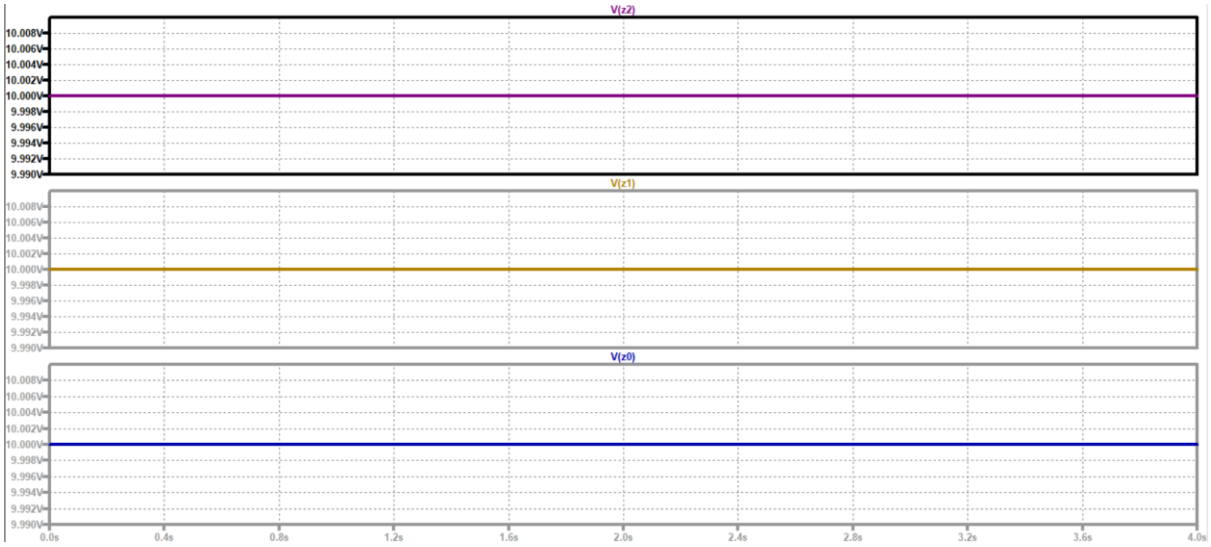


S.no	A (A2-A0)	B (B2-B0)	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	101	001	ASR	110	110

Test-7: A=110; B=110; DRin=1; DCin=1; SEL1=1; SEL0=1; OPSEL0=1; OPSEL1=0.



Test-7 Waveforms:



S.no	A (A2-A0)	B (B2-B0)	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	110	GT	111	111

Conclusion:

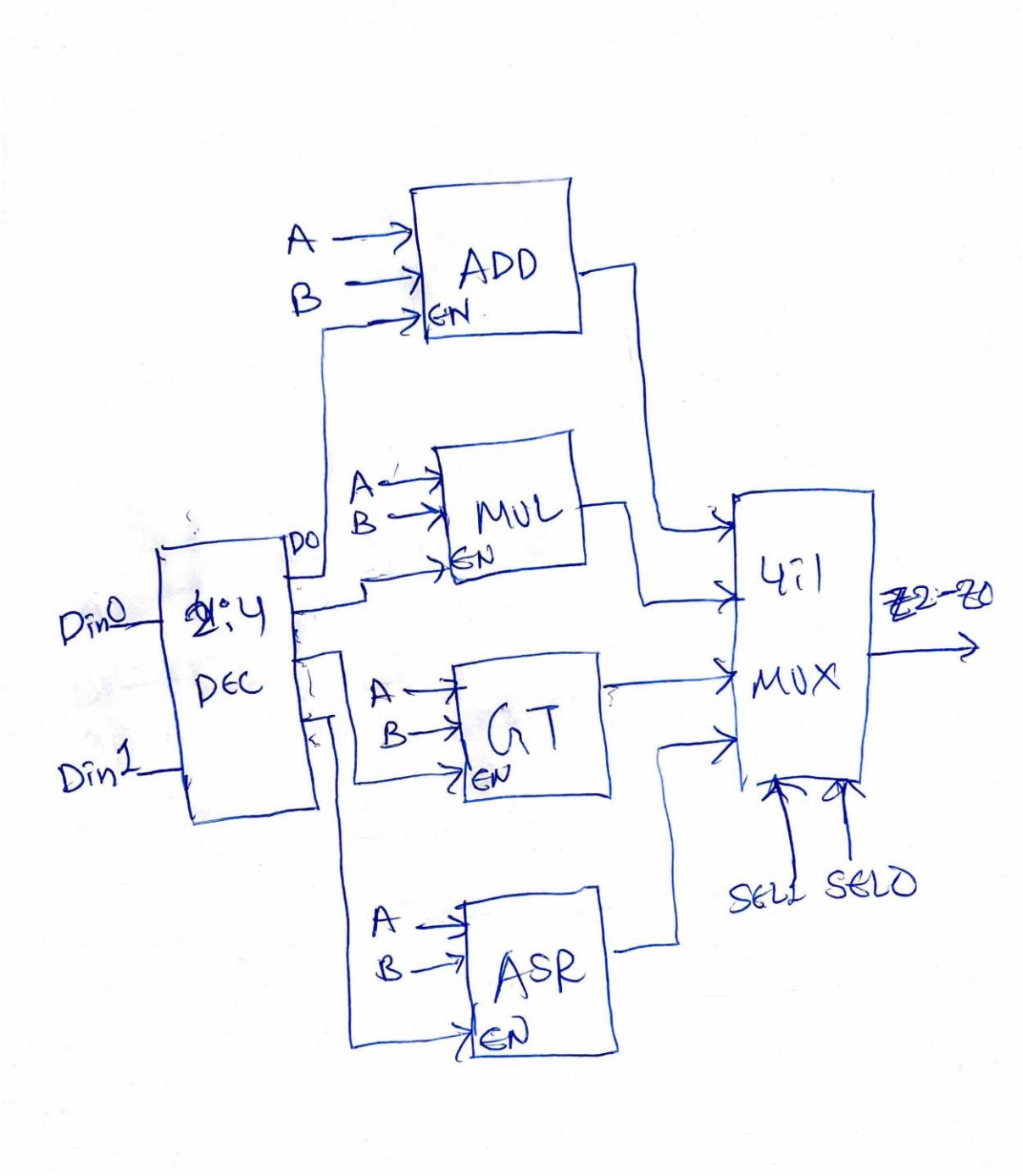
The Overall Circuit Calculations and the waveform outputs are matched exactly for all the above test cases from all the CUs and for all the four operations. Hence, the overall designed circuit is working properly for all the operations in CUs.

PART 2 CU WITH DECODER:

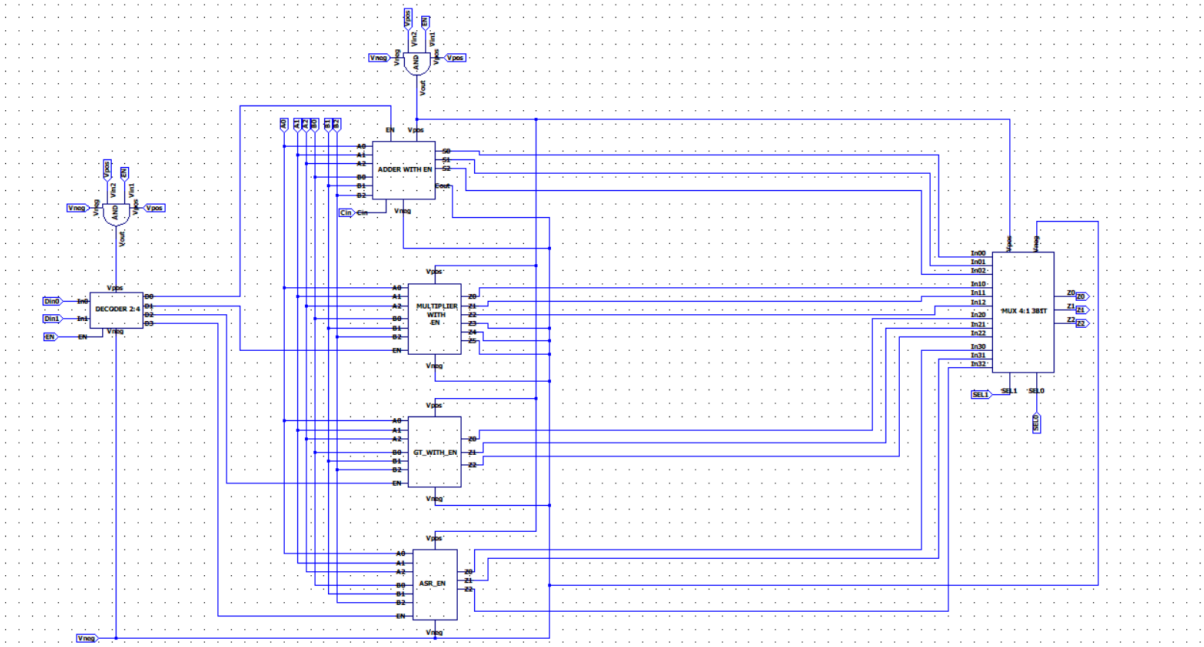
Design Explanation:

8. The below designed Computational Unit (CU) can perform 3bit Addition(ADD), Multiplication(MUL), Greater than(GT) and Arithmetic Shift Right(ASR) Operation. All the above operations have HIGH enable pin to operate.
9. In the below design 2:4 Decoder is used to enable one of the operations at a time and disable all the other three operations. D0,D1,D2 and D3 are connected to the ADD, MUL, GT and ASR respectively to enable one of the operations according to the 2:4 decoder truth table output. If D0,D1,D2 and D3 are HIGH the ADD,MUL,GT and ASR operations will perform respectively.
10. A(A2-A0) and B(B2-B0) are two three-bit inputs for all four above mentioned operations.
11. The 3bit output from every operation is fed to the inputs for the 3bit 4:1 Multiplexer (MUX).
12. The Carry-in taken to the 3bit Adder(Carry Select Adder) and the Cout is grounded. SUM (MSB.S2-S0) is given to (MSB.In02-In00) MUX input.
13. The three MSB bits of MUL is grounded (Z5-Z3) and Z2 as MSB (Z2-Z0) is given to the (MSB.In12-In10) MUX input.
14. The designed 3bit greater than (GT) gives 3bits High output(111) when A<=B otherwise the output is A(A2-A0). The output(Z2-Z0) from GT is given to the (MSB.In22-In20) MUX input.
15. The ASR is used to arithmetic shift right 3bit input A by 3bit input shift B. (MSB.Z2-Z0) is given to the (MSB.In32-In30) MUX input.
16. SEL1(MSB) and SEL0 are the selection inputs for the 4:1 MUX to select which operation output is to take at the MUX output (MSB.Z2-Z0)
17. Output from 3bit 4:1 MUX is final output of the below CU.

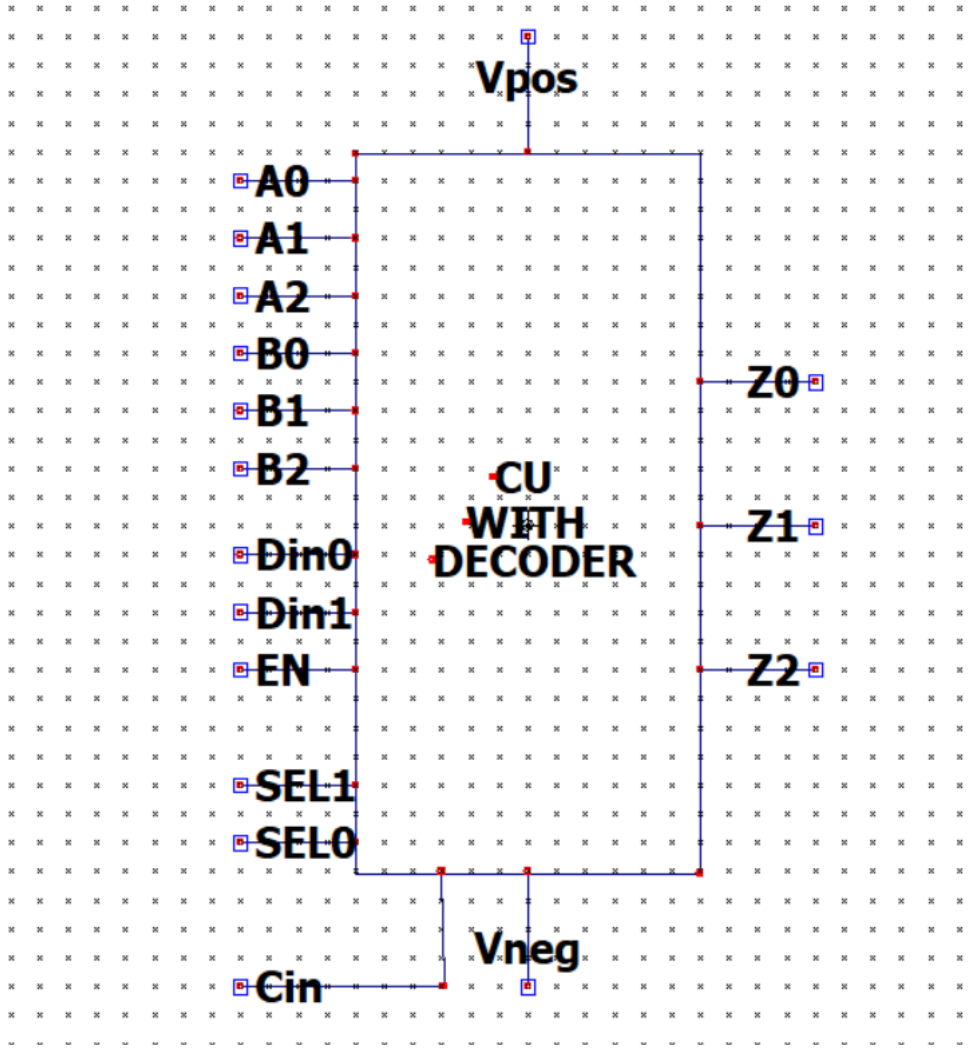
Circuit drawing:



Symbol schematic:



Symbol drawing:



Selection Table:

S.no	Din0 MSB	Din1 LSB	Operation Selection
1	0	0	ADD
2	0	1	MUL
3	1	0	GT
4	1	1	ASR

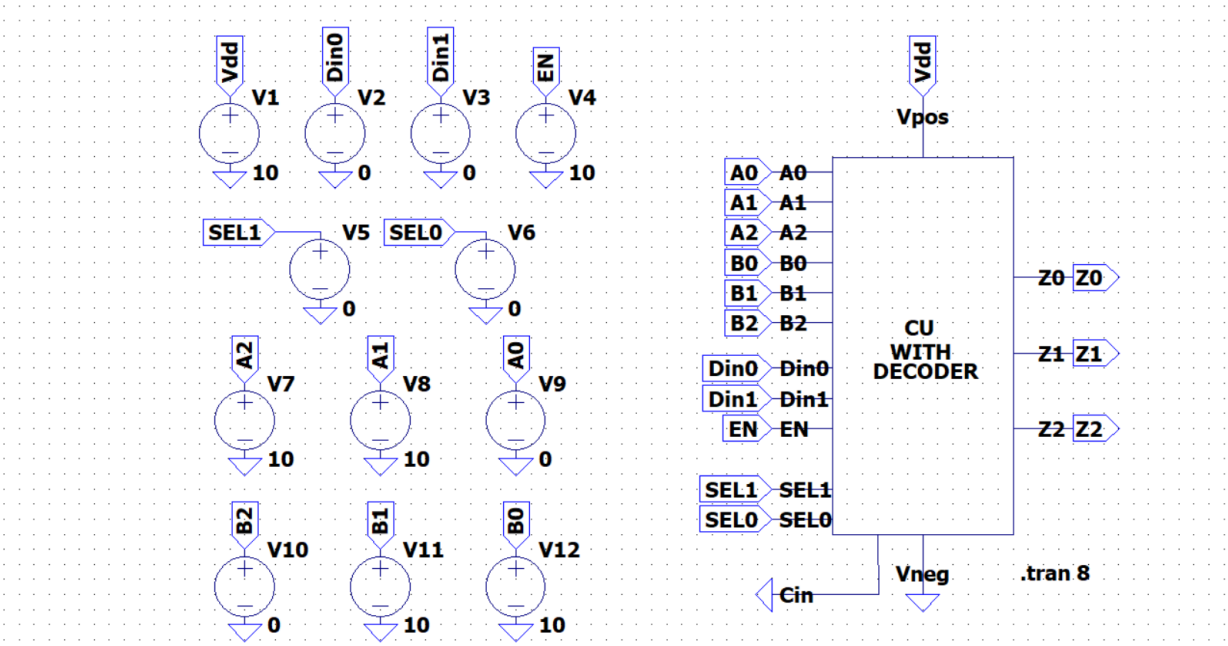
S.no	SEL1 MSB	SEL0 LSB	Output Selection
1	0	0	ADD
2	0	1	MUL
3	1	0	GT
4	1	1	ASR

Test Calculation Table:

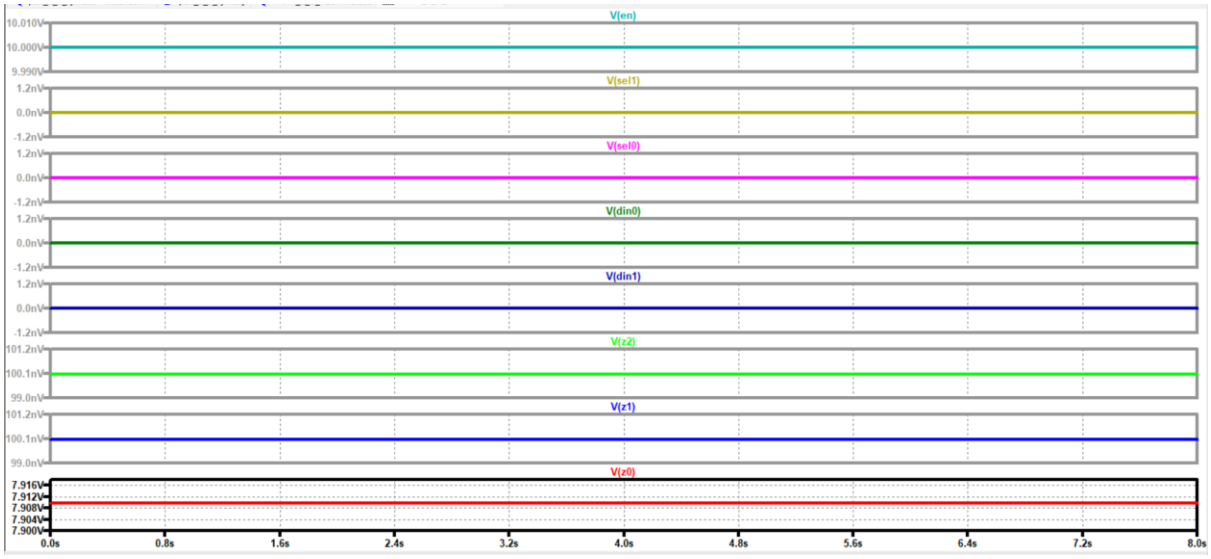
S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)
1	110	011	0	0	0	0	ADD	001
2	110	011	0	1	0	1	MUL	010
3	110	011	1	0	1	0	GT	111
4	110	011	1	1	1	1	ASR	000
5	011	110	1	0	1	0	GT	111
6	101	001	1	1	1	1	ASR	110
7	011	011	1	0	1	0	GT	111

Test circuits:

Test1: A=110; B=011; SEL1=0; SEL0=0; Din0=0; Din1=0.

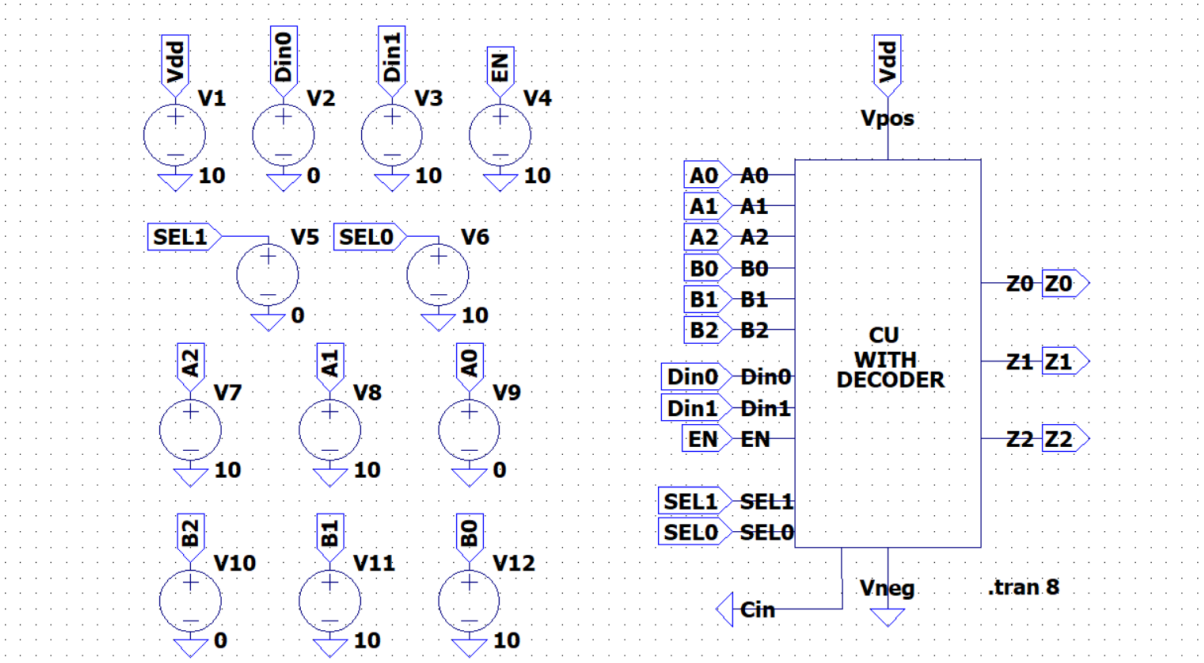


Test-1 Waveforms:

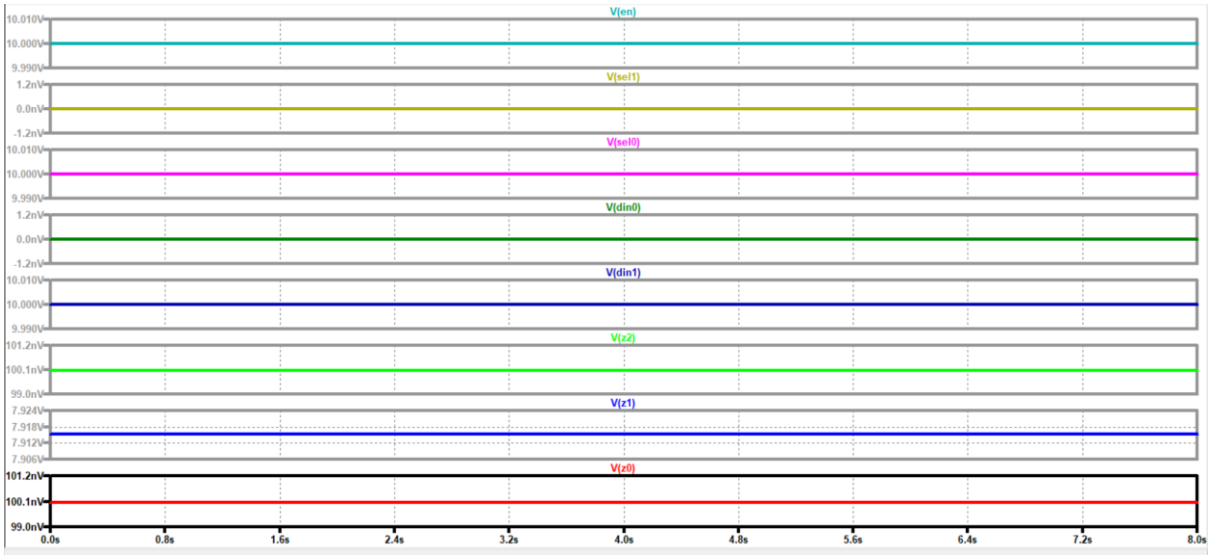


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	0	0	0	0	ADD	001	001

Test2: A=110; B=011; SEL1=0; SEL0=1; Din0=0; Din1=1.

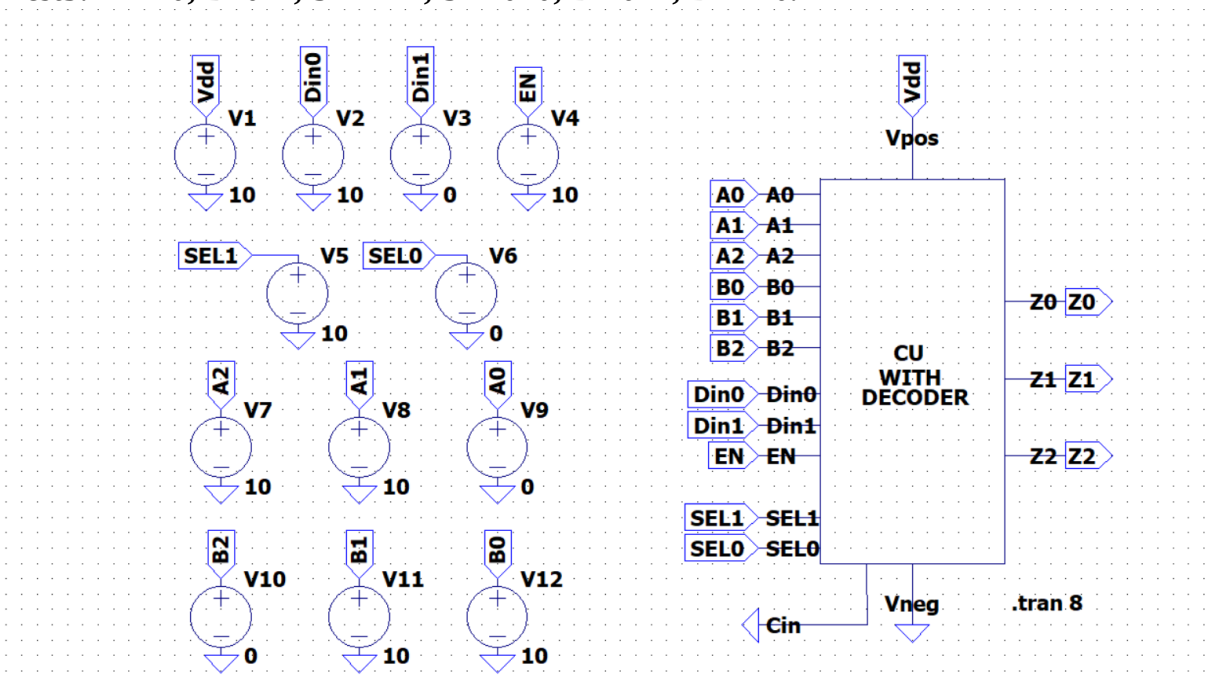


Test-2 Waveforms:

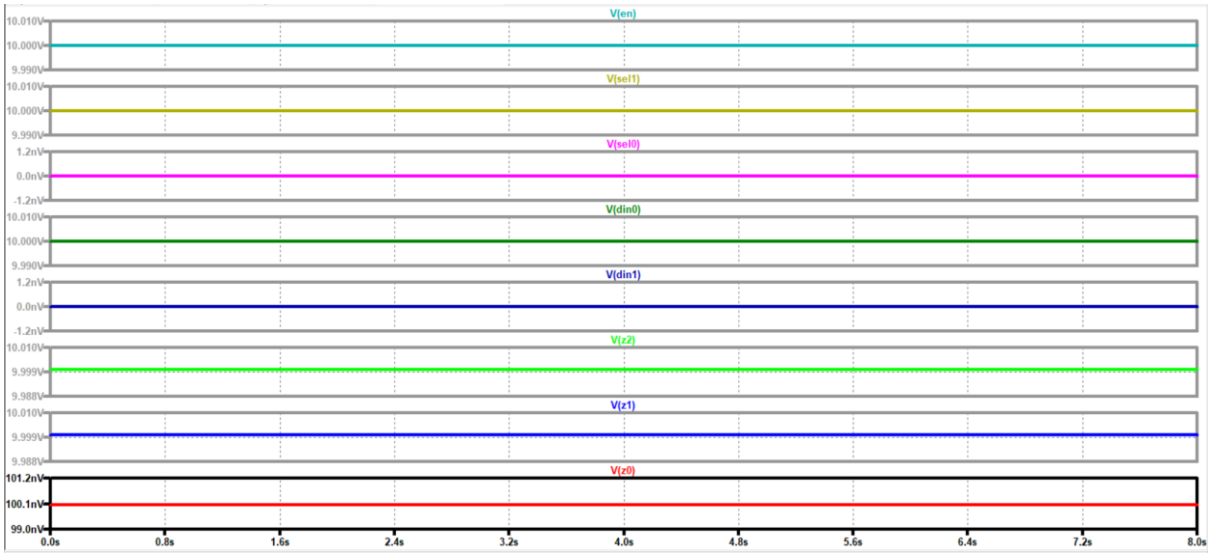


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	0	1	0	1	MUL	010	010

Test3: A=110; B=011; SEL1=1; SEL0=0; Din0=1; Din1=0.

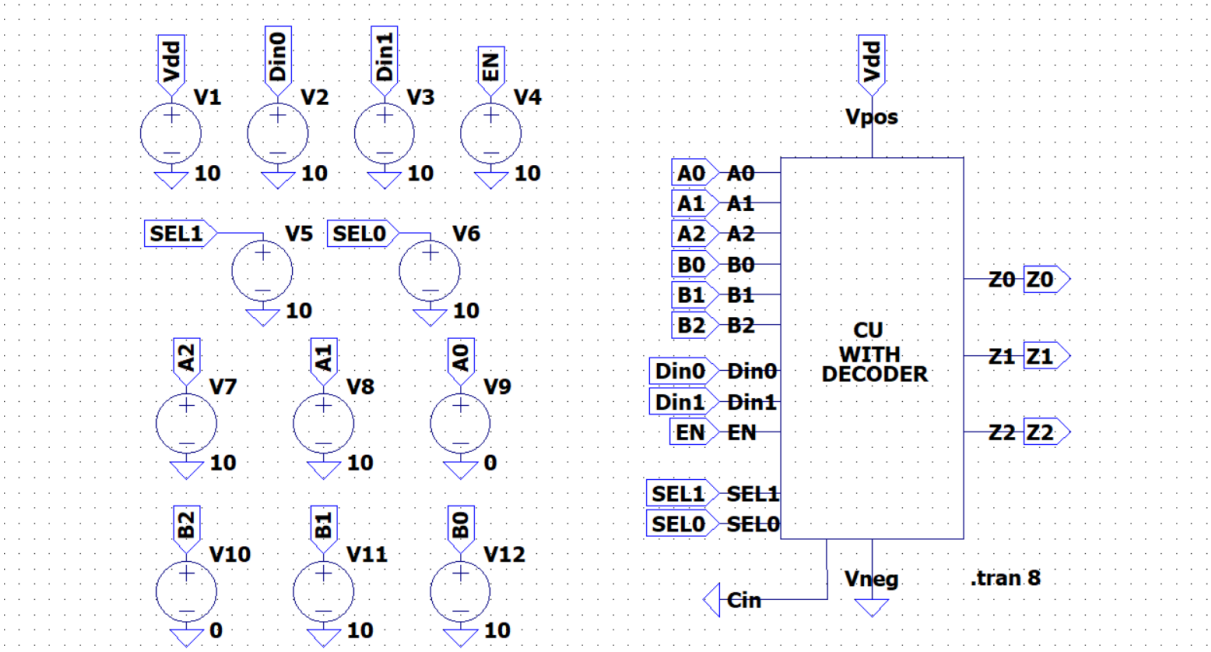


Test-3 Waveforms:

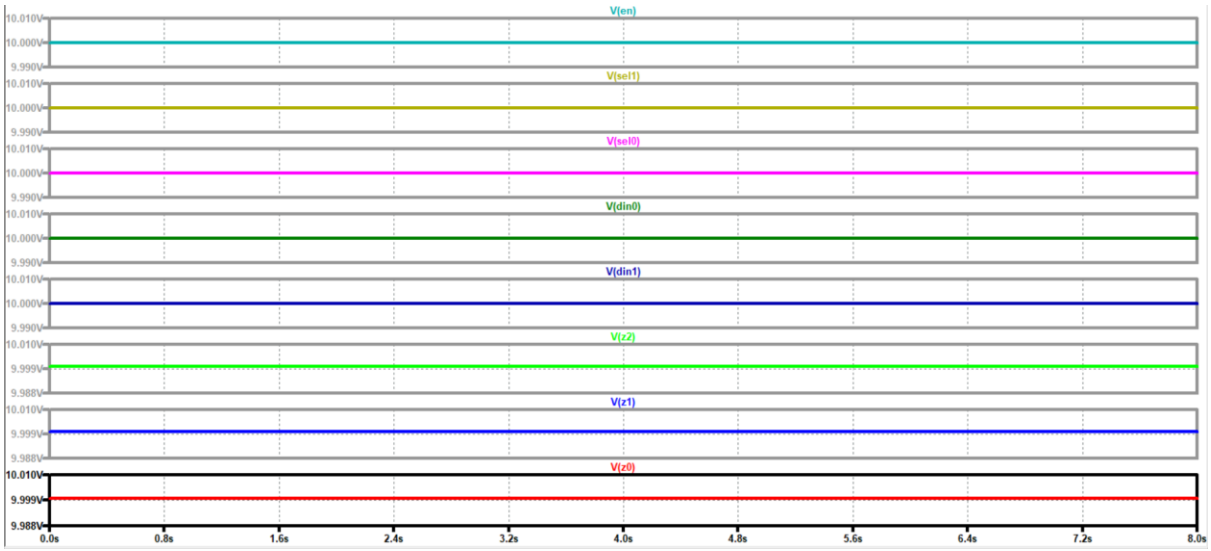


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	1	0	1	0	GT	110	110

Test4: A=110; B=011; SEL1=1; SEL0=1; Din0=1; Din1=1.

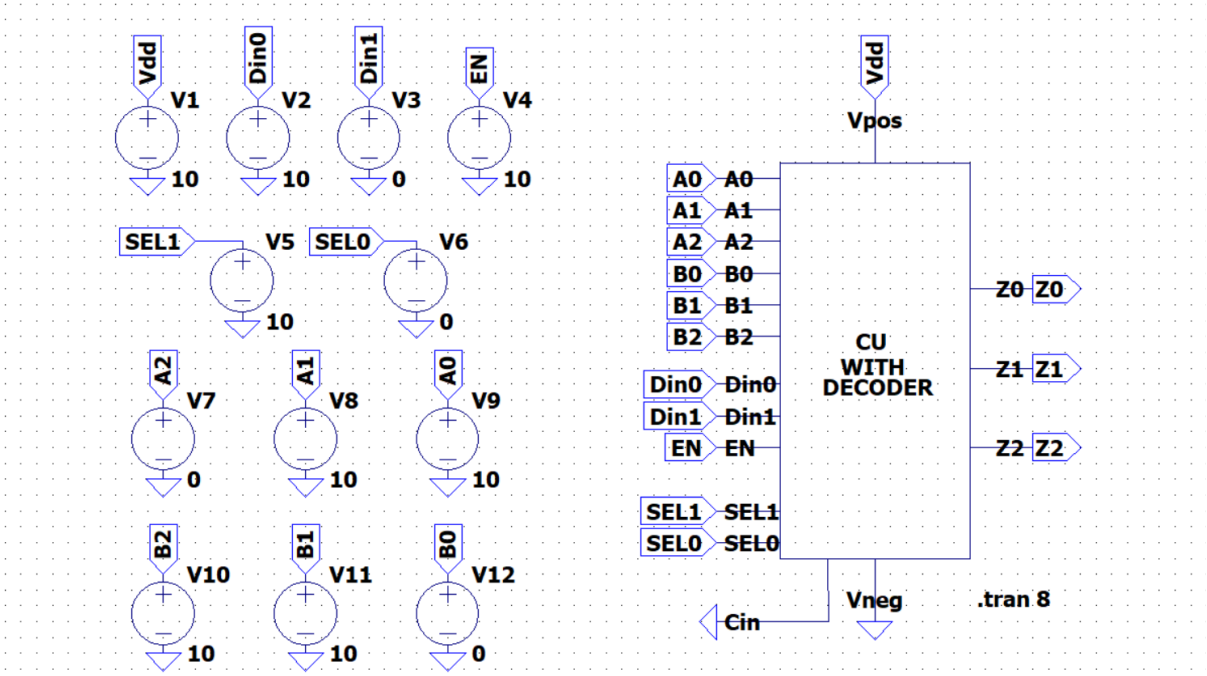


Test-4 Waveforms:

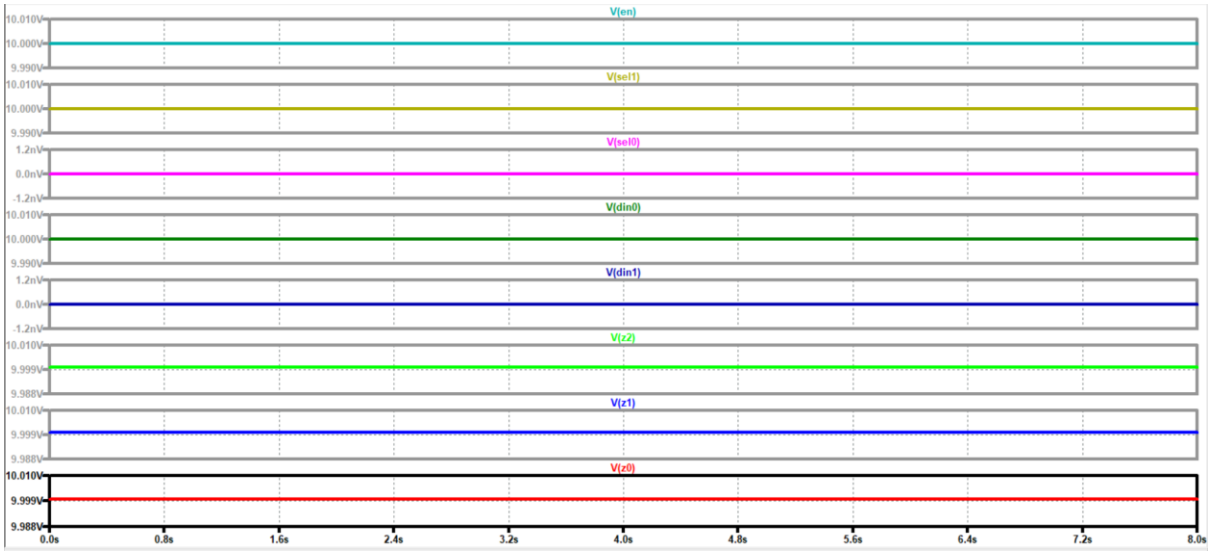


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	011	1	1	1	1	ASR	111	111

Test5: A=011; B=110; SEL1=1; SEL0=0; Din0=1; Din1=0.

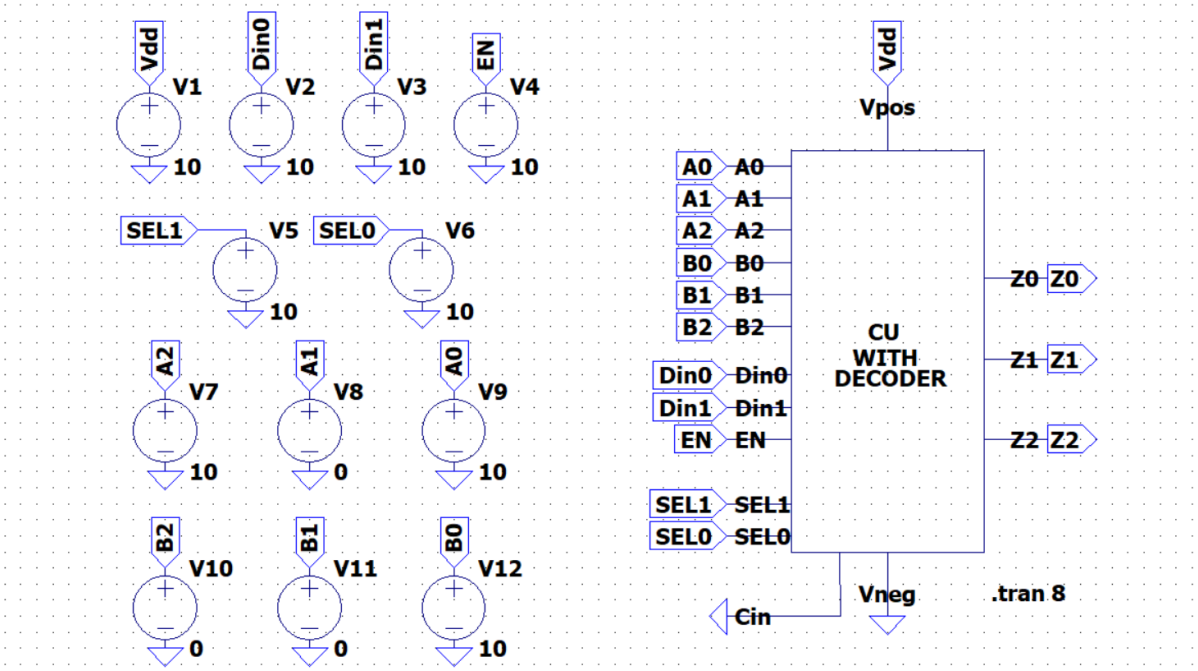


Test-5 Waveforms:

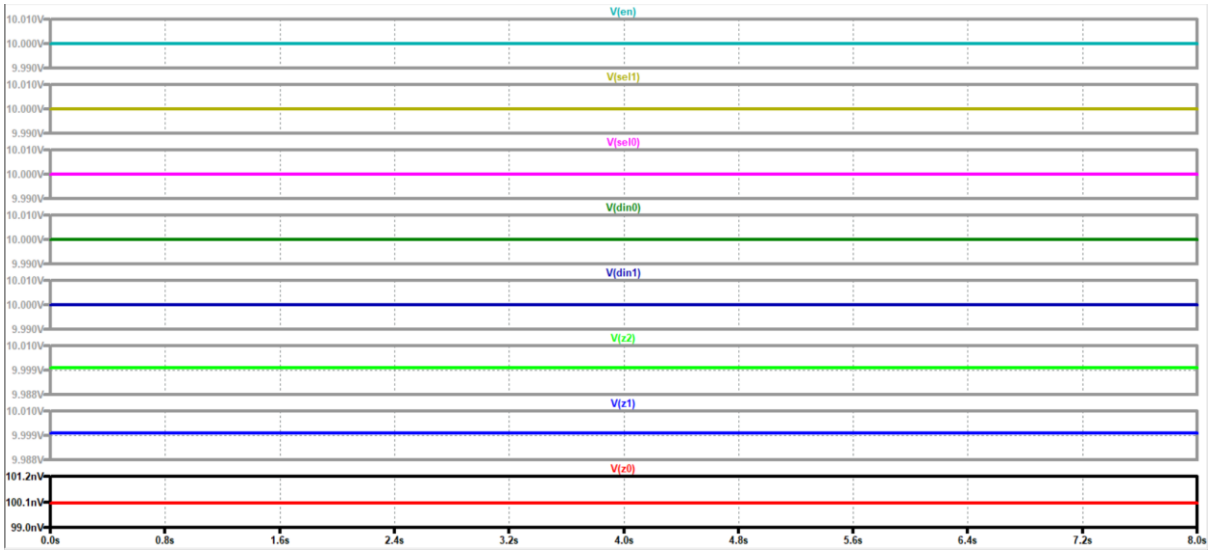


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	011	110	1	0	1	0	GT	111	111

Test-6: A=101; B=001; SEL1=1; SEL0=1; Din0=1; Din1=1.

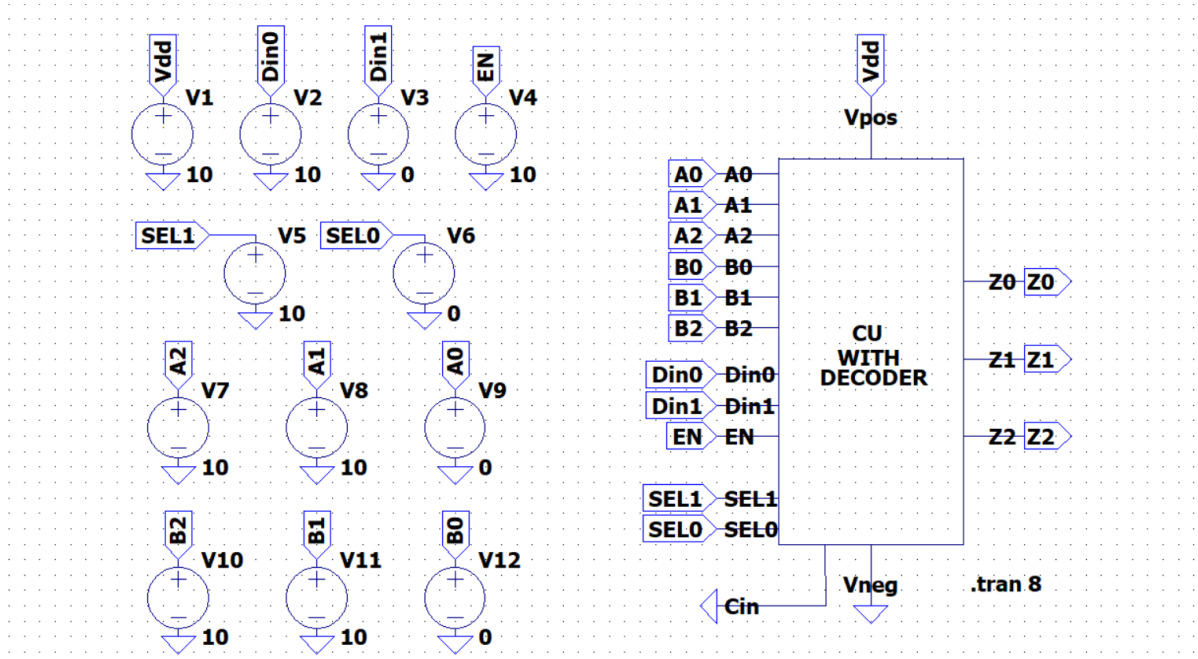


Test-6 Waveforms:

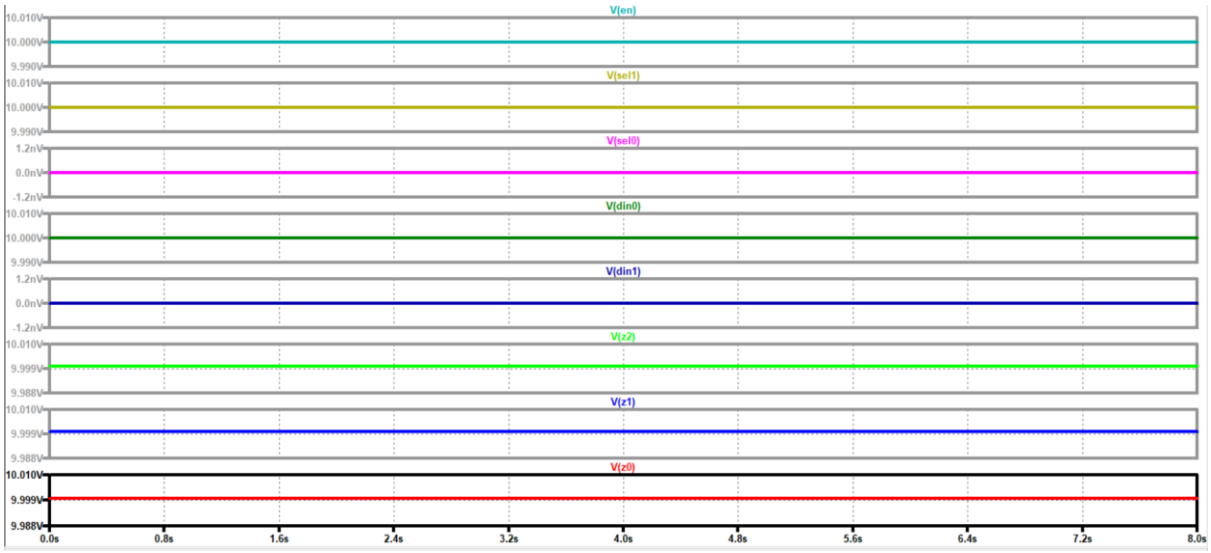


S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	101	001	1	1	1	1	ASR	110	110

Test-7: A=110; B=110; SEL1=1; SEL0=0; Din0=1; Din1=0.



Test-7 Waveforms:



S.no	A (A2-A0)	B (B2-B0)	SEL1 MSB	SEL0 LSB	Din0 MSB	Din1 LSB	Operation	Output (Z2-Z0)	Waveform (Z2-Z0)
1	110	110	1	0	1	0	GT	111	111

Conclusion:

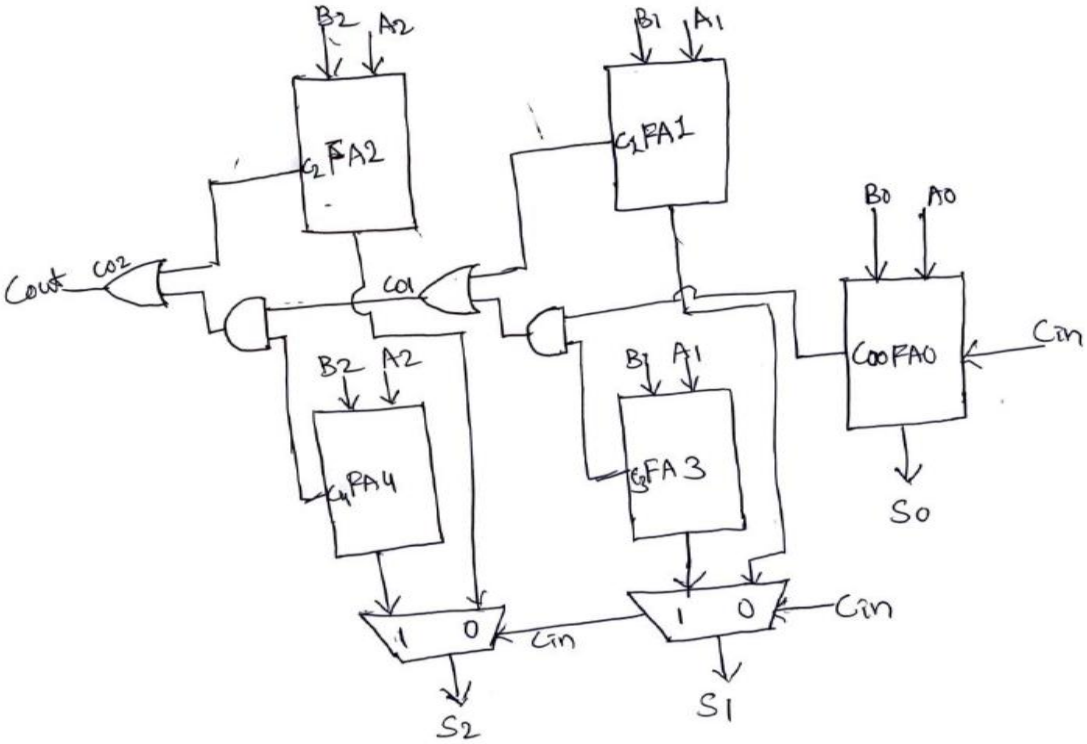
The CU with decoder calculation outputs and the output waveforms are matching exactly from the comparison table for all the test cases. Therefore, the designed CU with decoder of four Operation (ADD,MUL,GT and ASR) is working properly and verified.

SUBCOMPONENTS (OPERATION CIRCUITS):

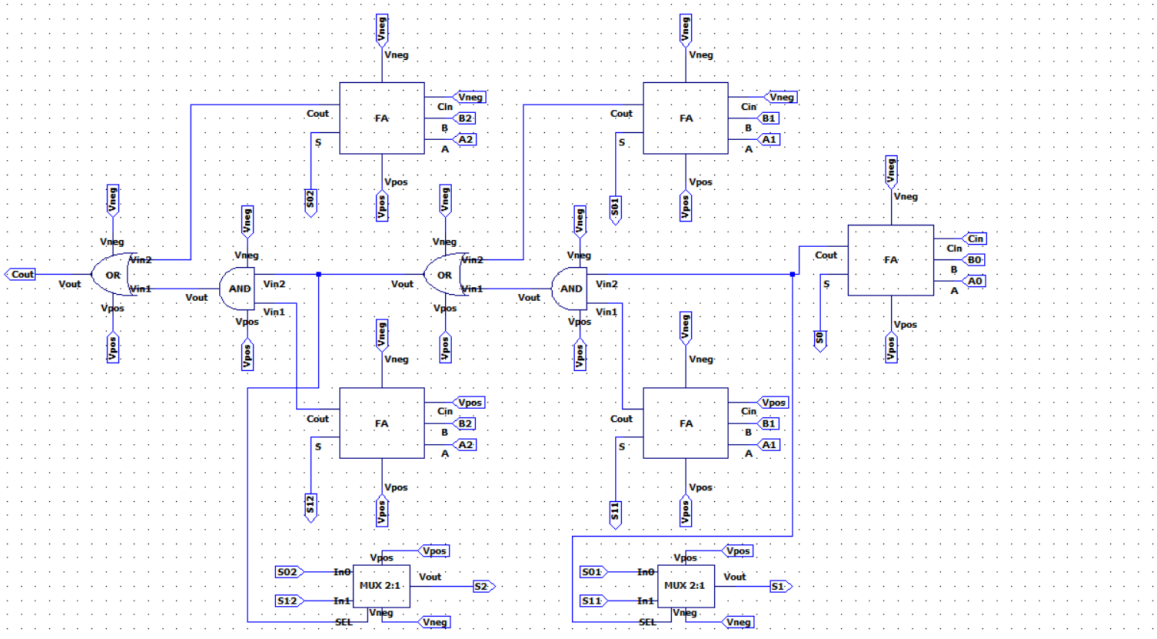
ADDER:

For this 3bit adder I have used the previous developed 3bit Carry Select Adder.

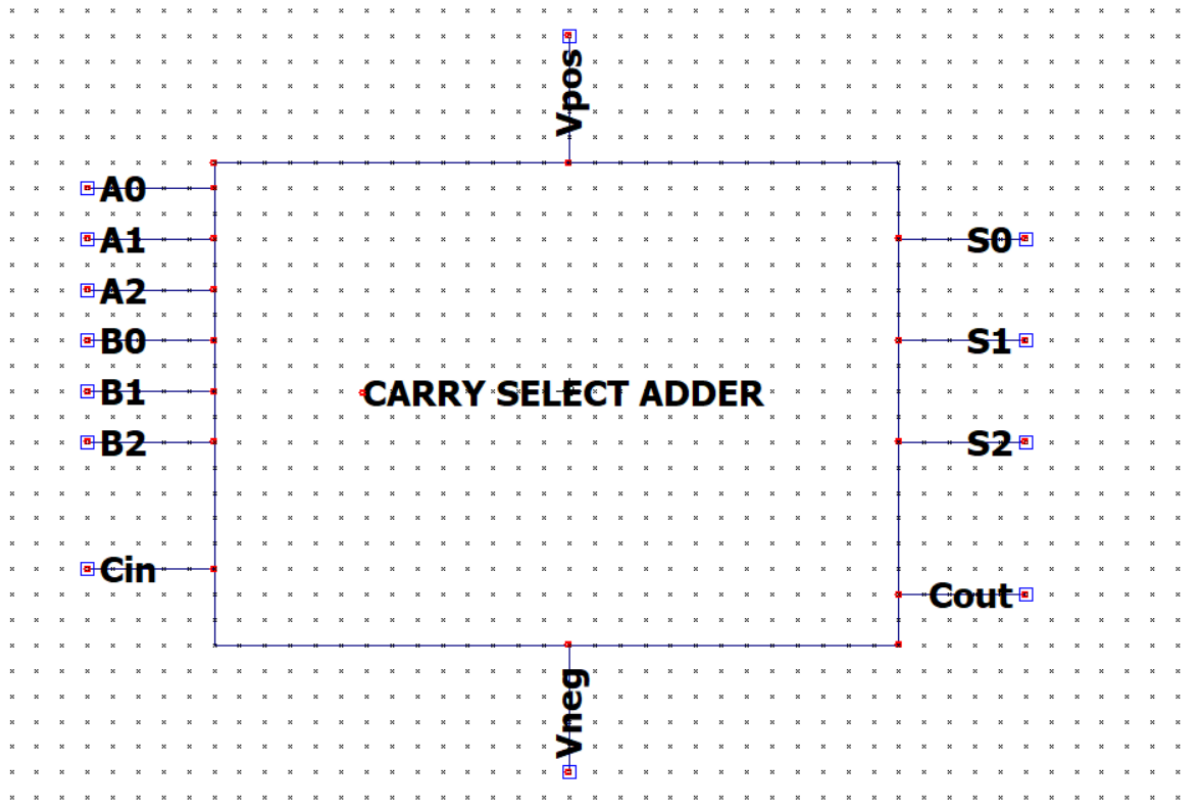
Gate-level drawing:



Symbol schematic:



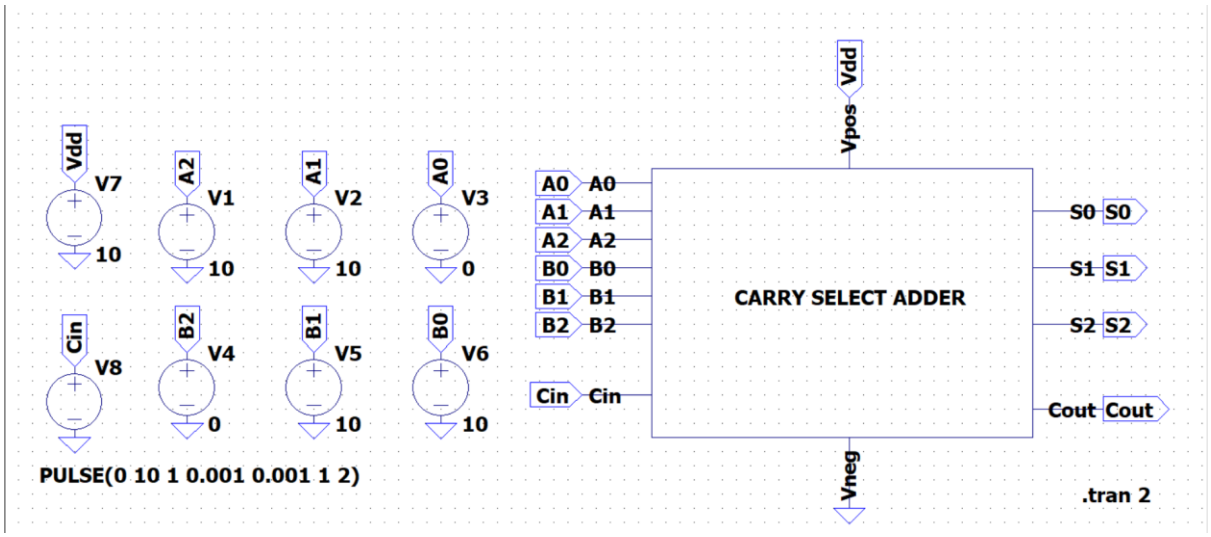
Symbol drawing:



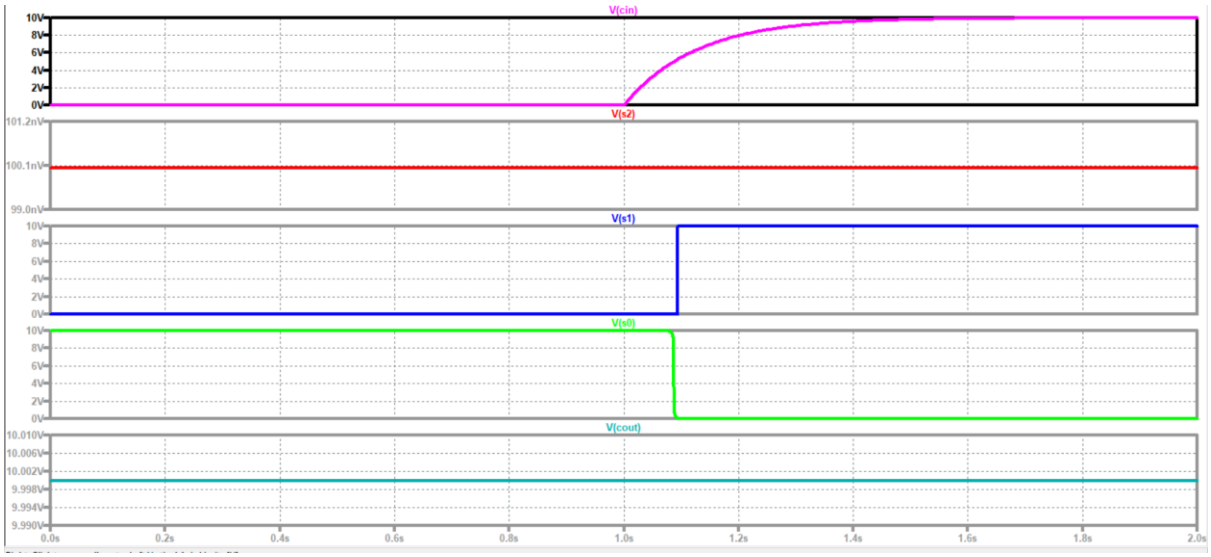
Test Calculation Table:

Test Case	A(A2-A0)	B(B2-B0)	Cin	SUM(S2-S0)	Cout
1	110	011	0	001	1
2	110	011	1	010	1

Test circuit:
I have added Parasitic properties to the Cin.



Test circuit waveforms:



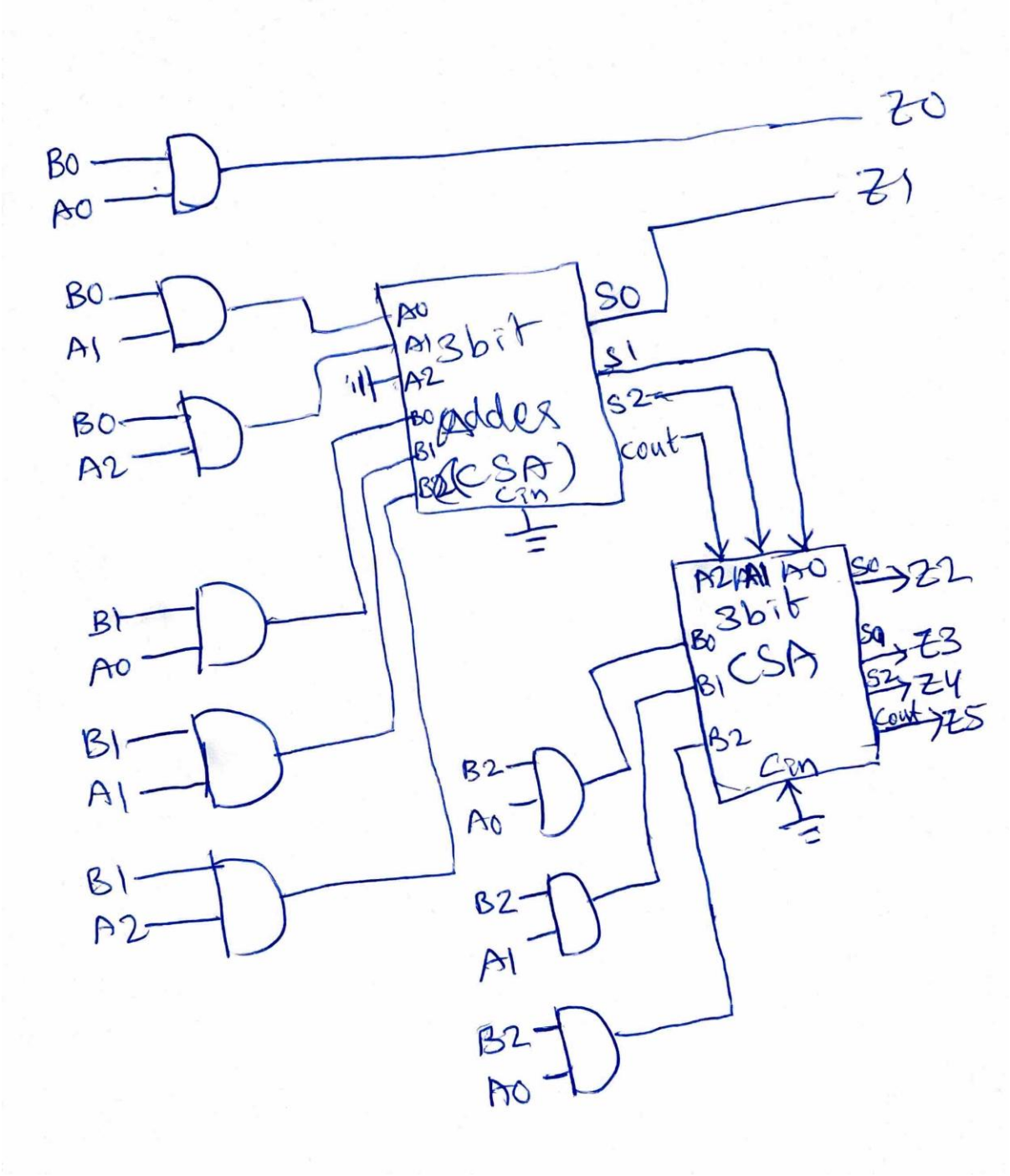
Test Case	A (A2-A0)	B (B2-B0)	Cin	Calculations		Waveform	
				SUM (S2-S0)	Cout	SUM (S2-S0)	Cout
1	110	011	0	001	1	001	1
2	110	011	1	010	1	010	1

Verification statement:

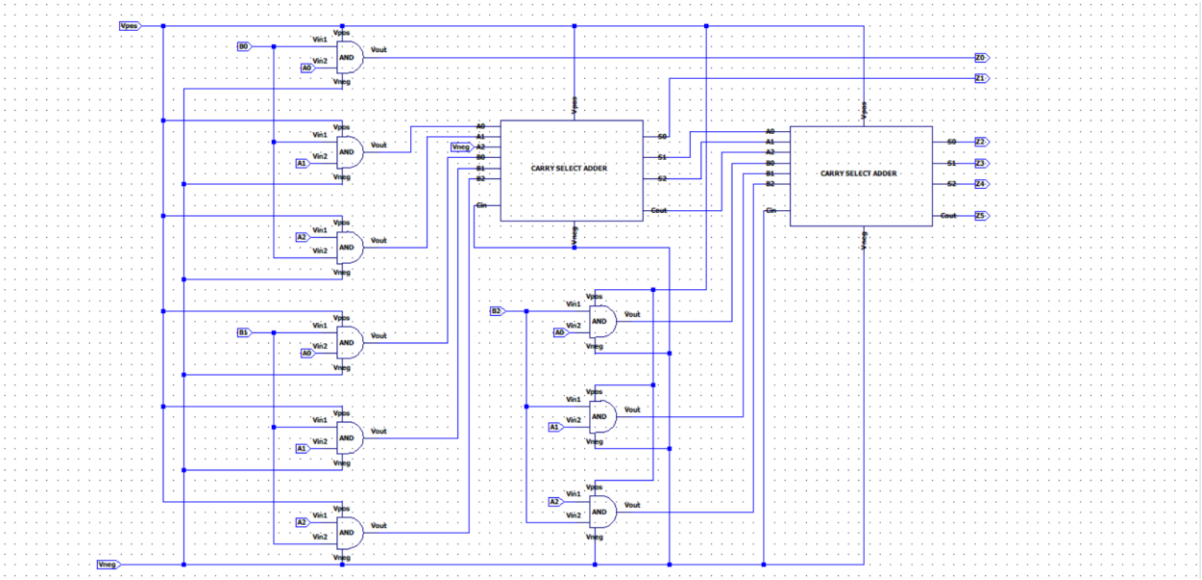
The test calculations and output waveforms are matched exactly from the comparison table. So, the designed 3-bit Carry Select Adder is working properly.

MULTIPLIER:

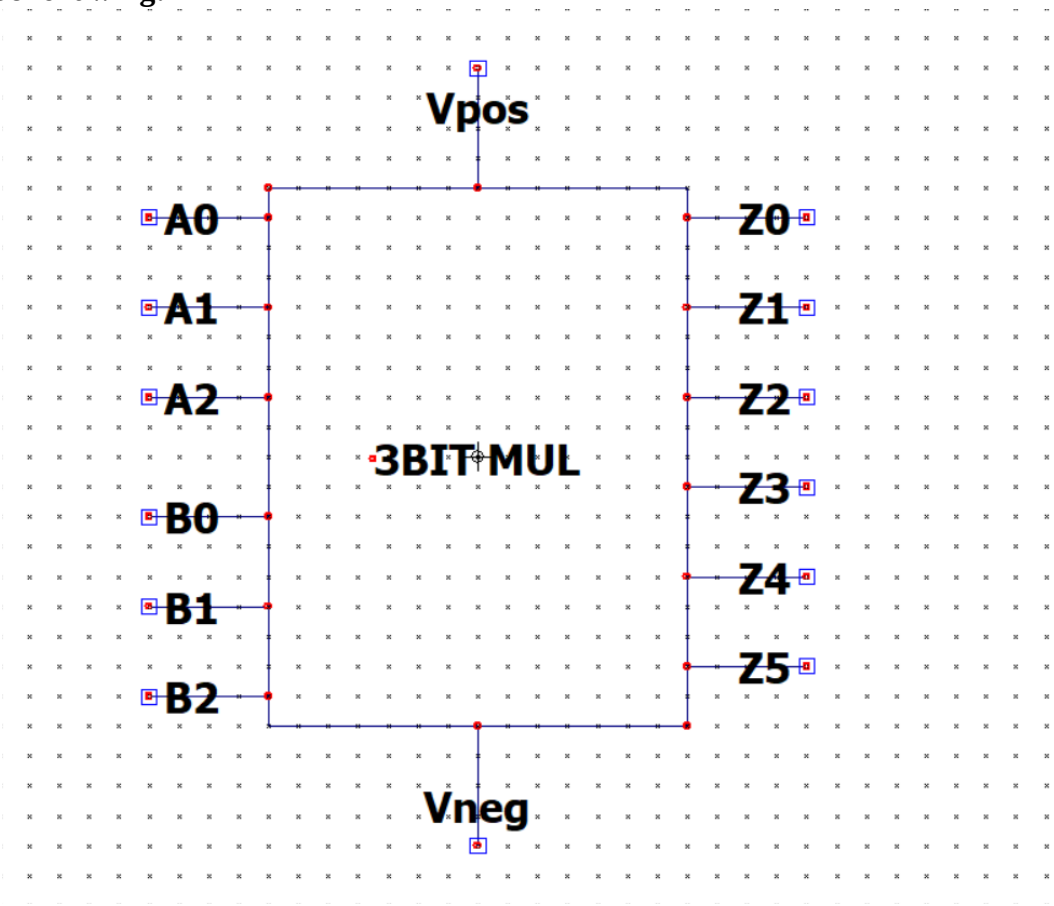
Gate-level drawing:



Symbol schematic:



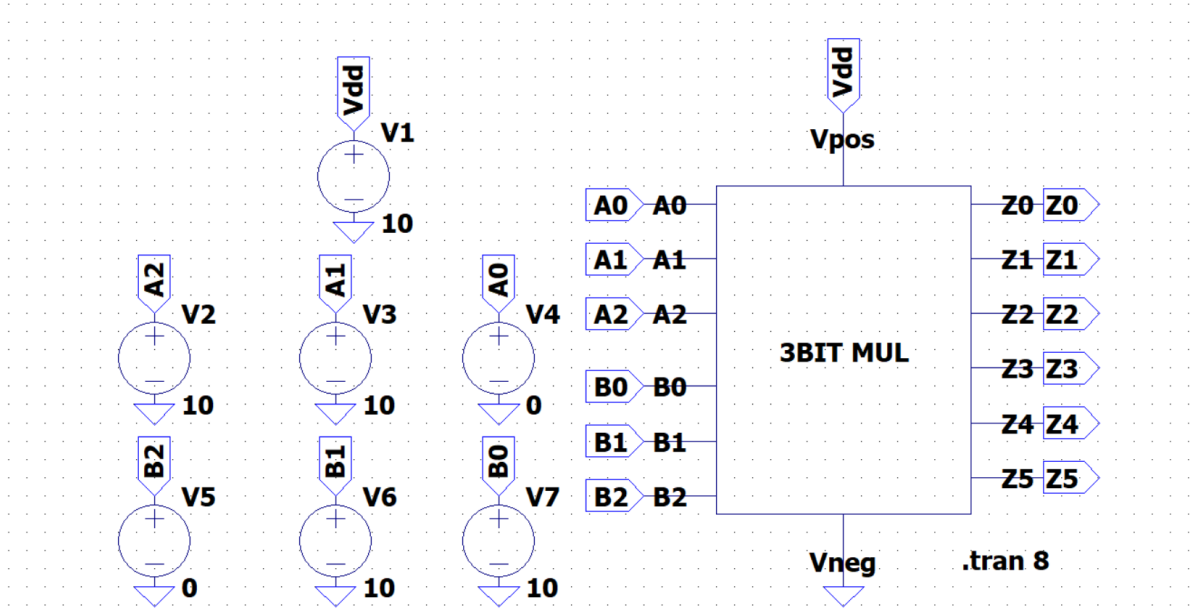
Symbol drawing:



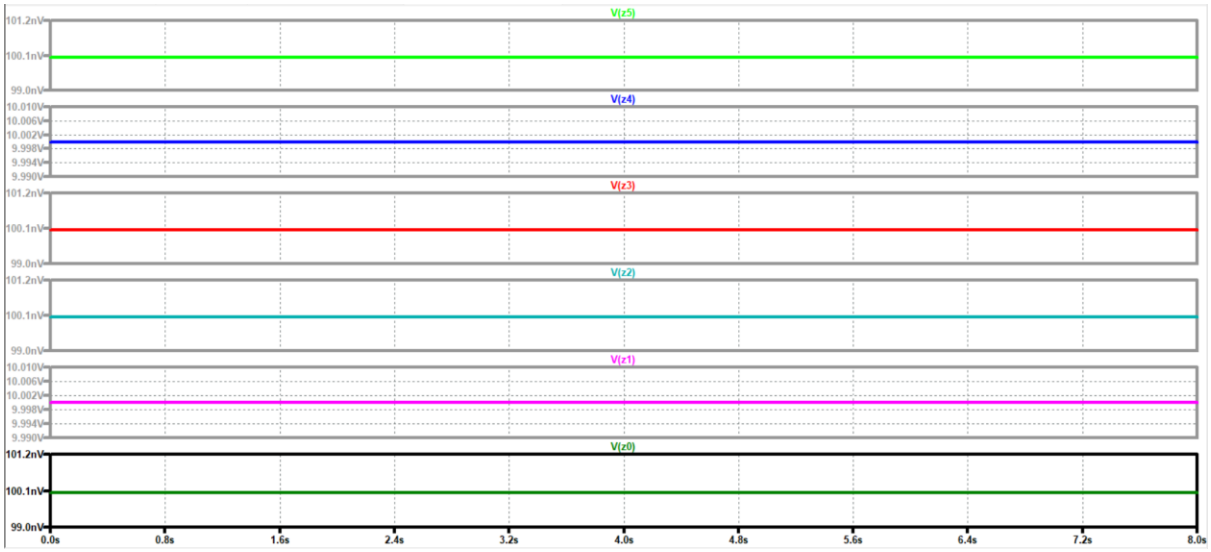
Test Calculation Table:

Test Case	A(A2-A0)	B(B2-B0)	Output Z5-Z0
1	110	011	010010

Test circuit:



Test circuit waveforms:



Test Case	A(A2-A0)	B(B2-B0)	Calculation Z5-Z0	Waveform Z5-Z0
1	110	011	010010	010010

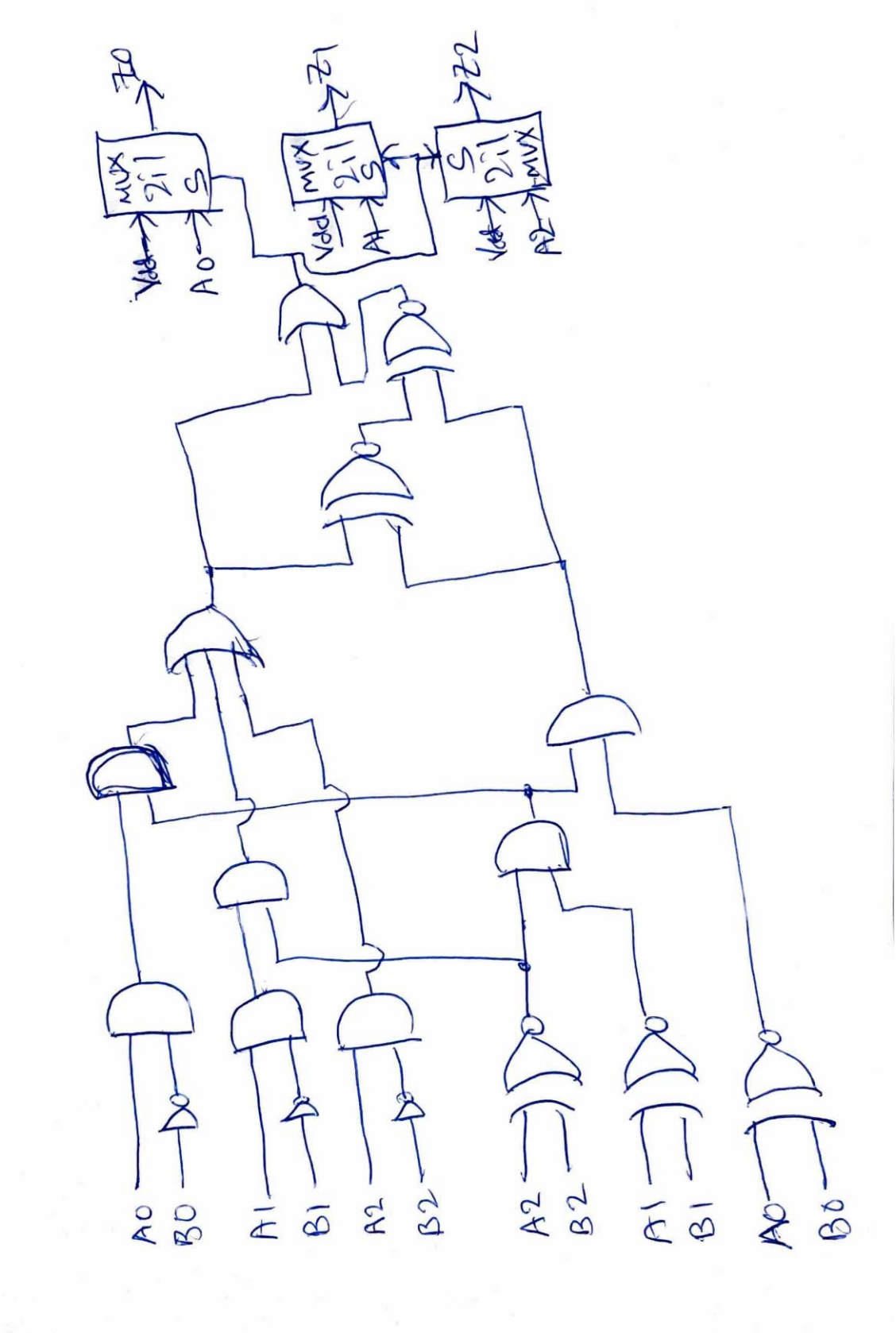
Verification statement:

The test calculations and output waveforms are matched exactly from the comparison table. So, the designed 3-bit Multiplier is working properly.

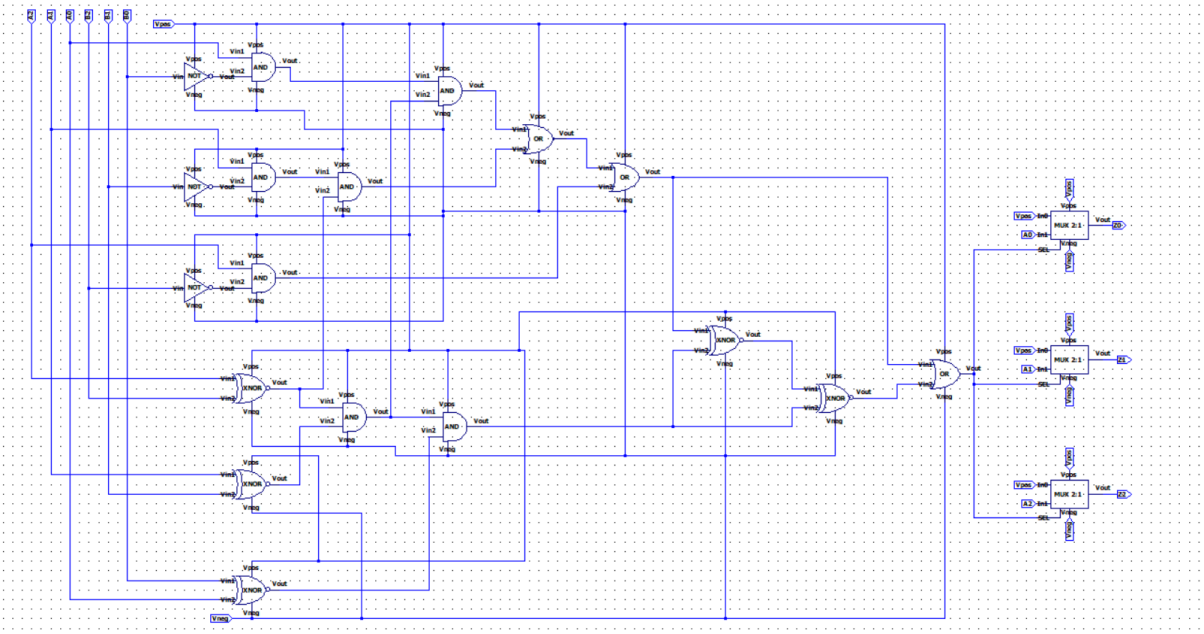
GREATER THAN:

The output is A when A>B, If A<=B the output is 3bit High(111).

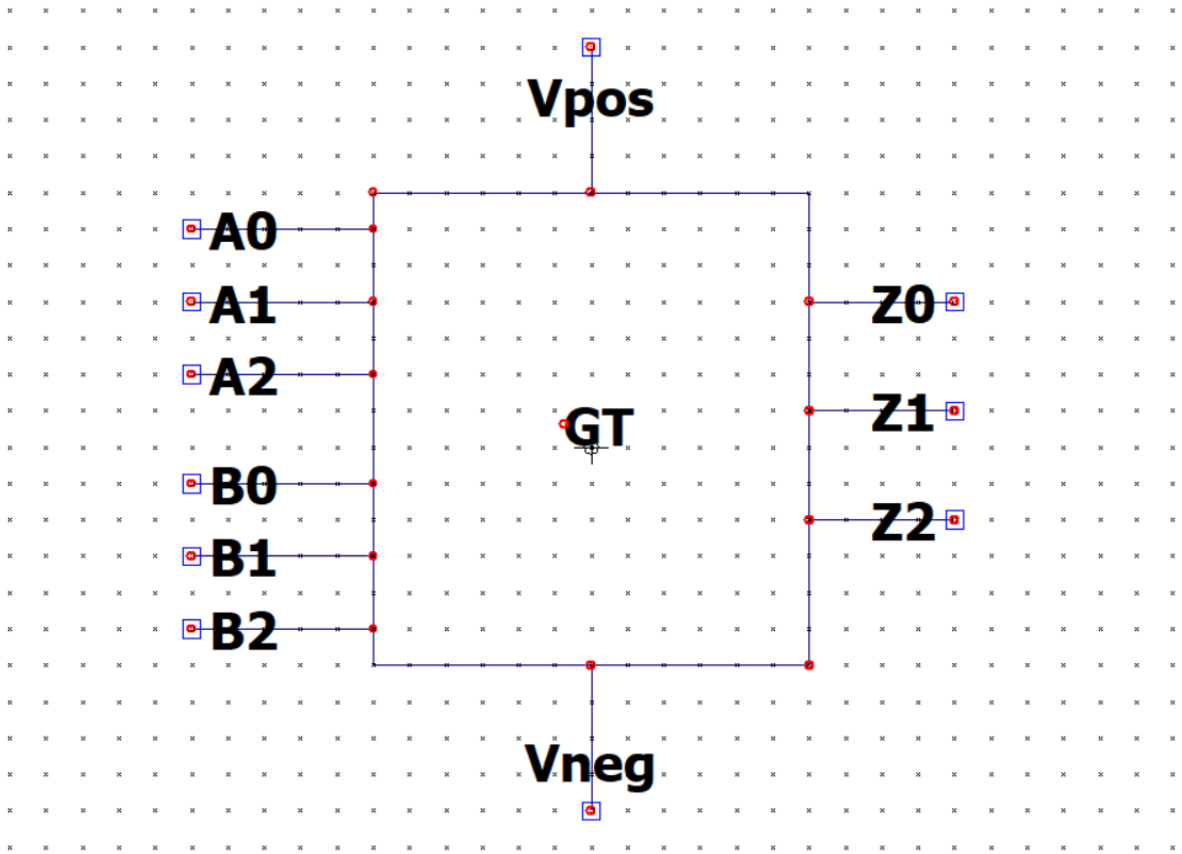
Gate-level drawing:



Symbol schematic:



Symbol drawing:

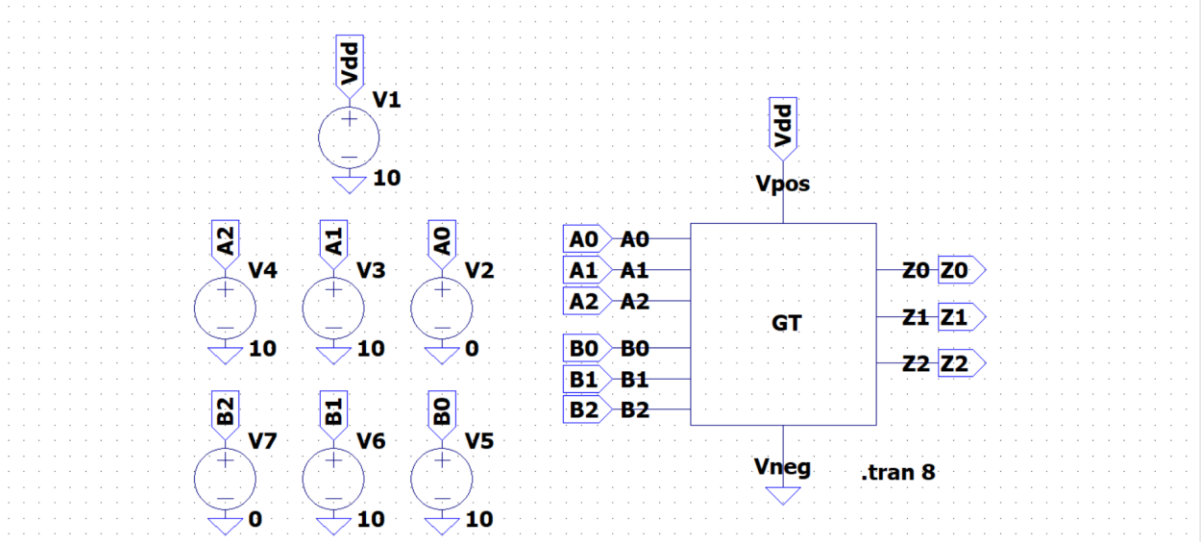


Test Calculation Table:

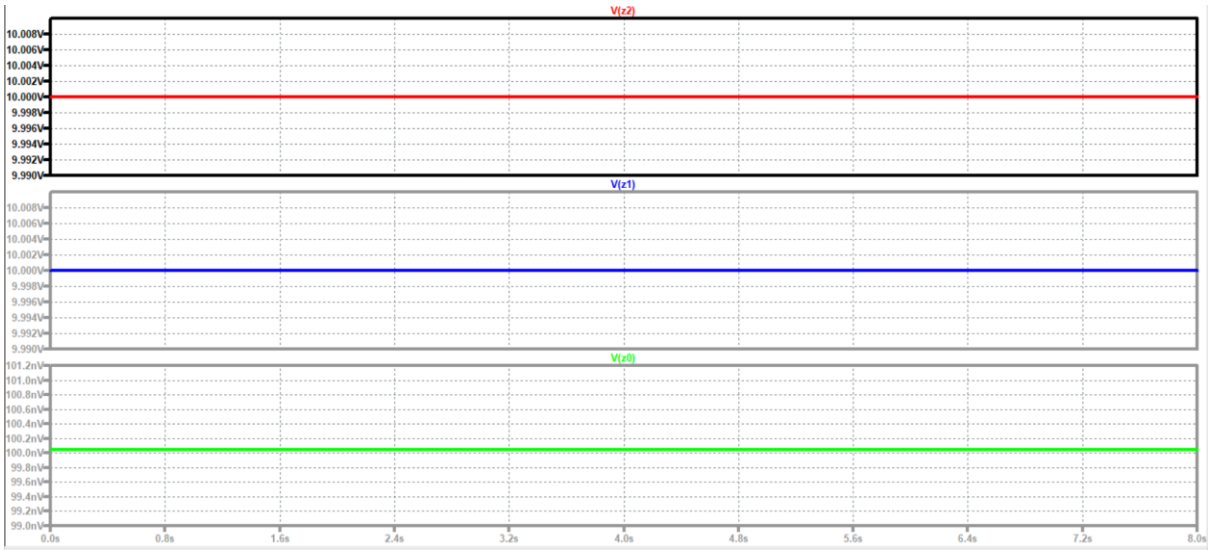
Test Case	A(A2-A0)	B(B2-B0)	Output Z2-Z0
1	110	011	110(A>B)
2	011	110	111(A<B)
3	110	110	111(A=B)

Test circuits:

Test-1:A=110 and B=011

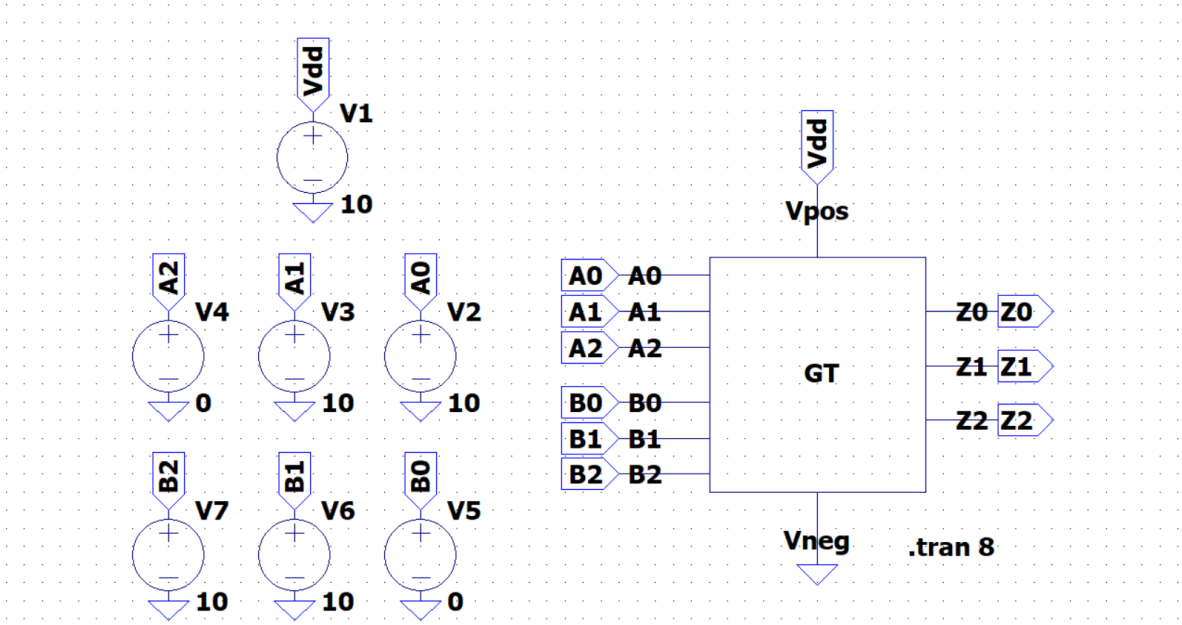


Test-1 circuit waveforms:

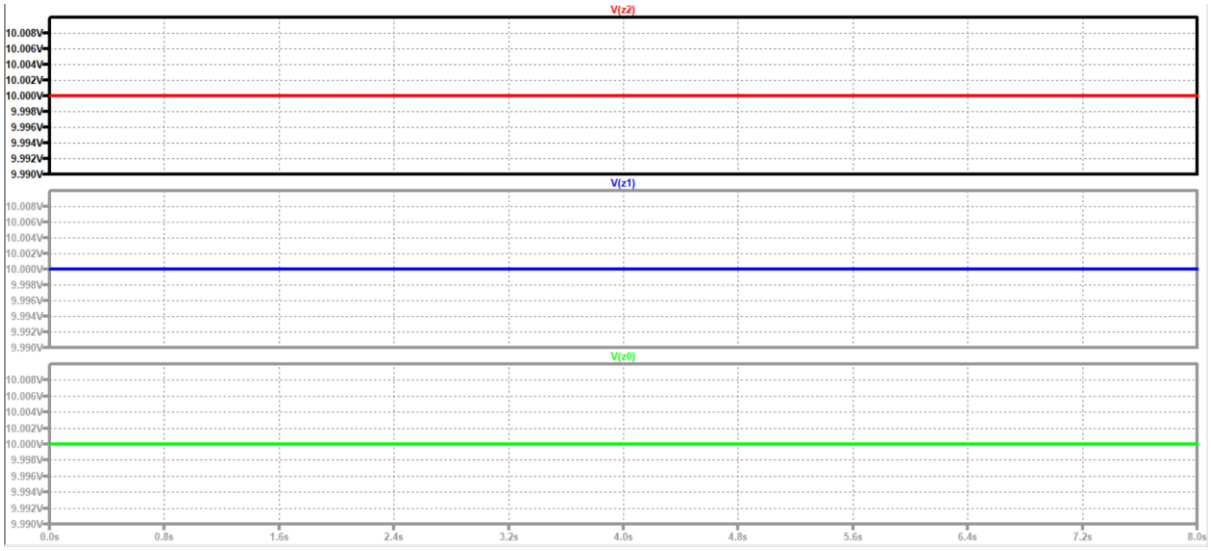


Test Case	A(A2-A0)	B(B2-B0)	Calculation Z2-Z0	Waveform Z2-Z0
1	110	011	110	110

Test-2: A=011 and B=110

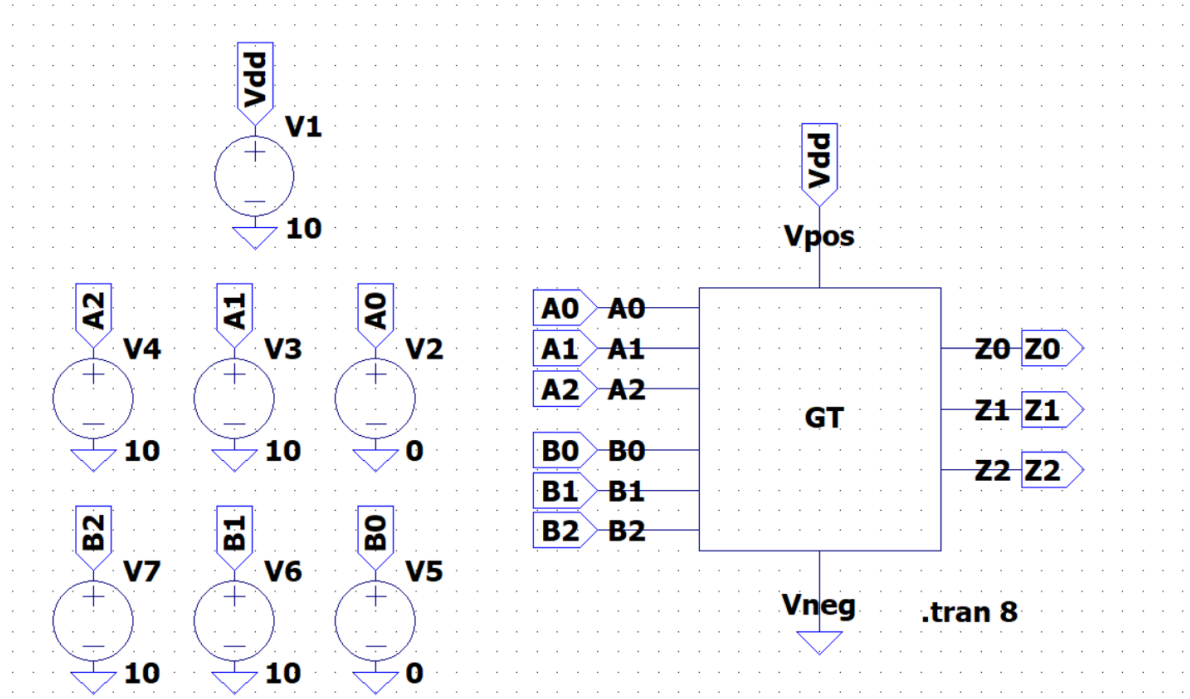


Test-2 Waveforms:

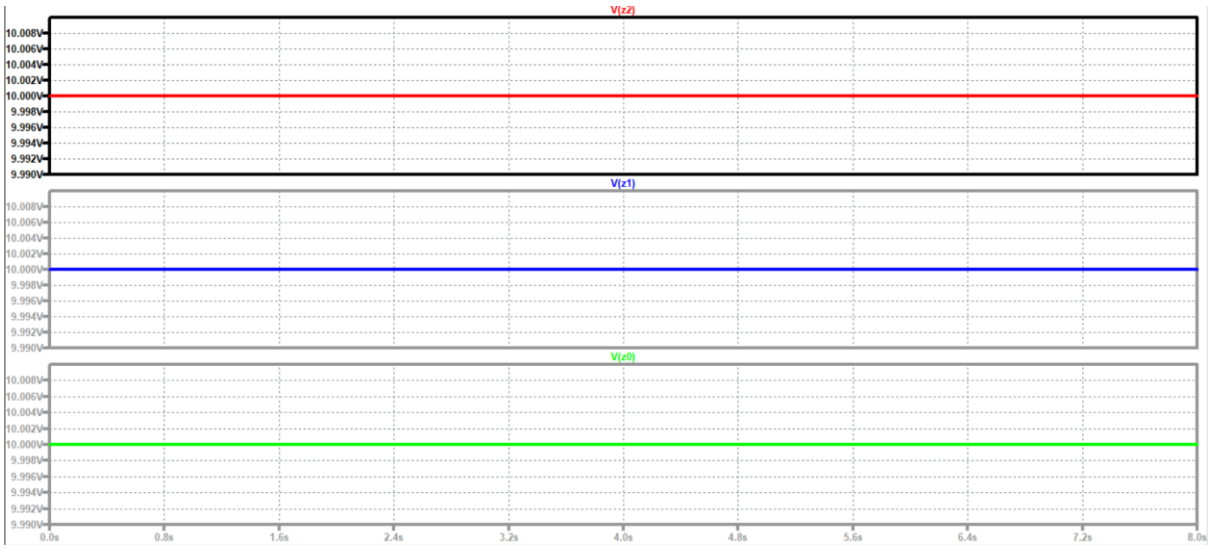


Test Case	A(A2-A0)	B(B2-B0)	Calculation Z2-Z0	Waveform Z2-Z0
1	011	110	111	111

Test-3: A=110 and B=110



Test-3 Waveforms:



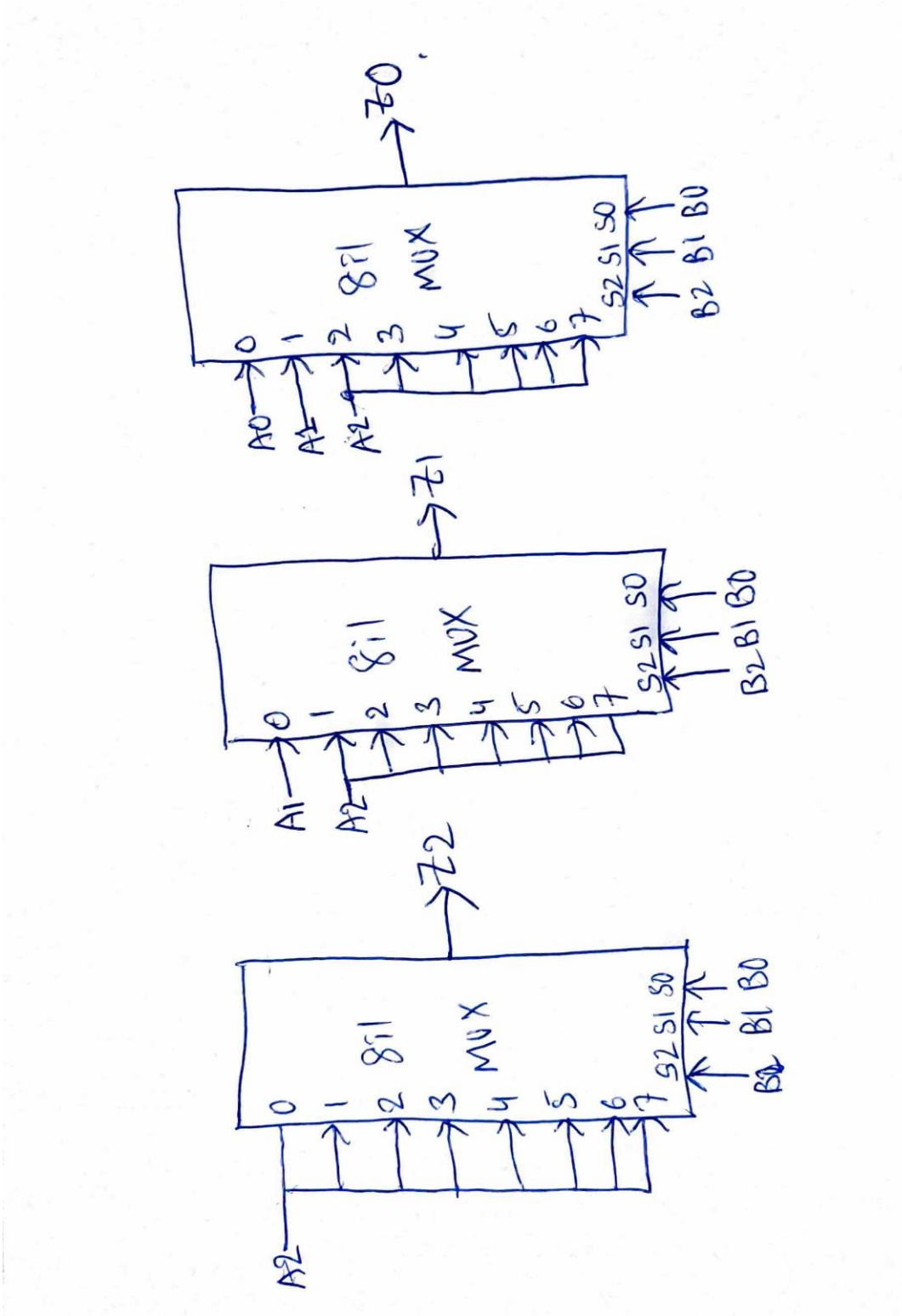
Test Case	A(A2-A0)	B(B2-B0)	Calculation Z2-Z0	Waveform Z2-Z0
1	110	110	111	111

Verification statement:

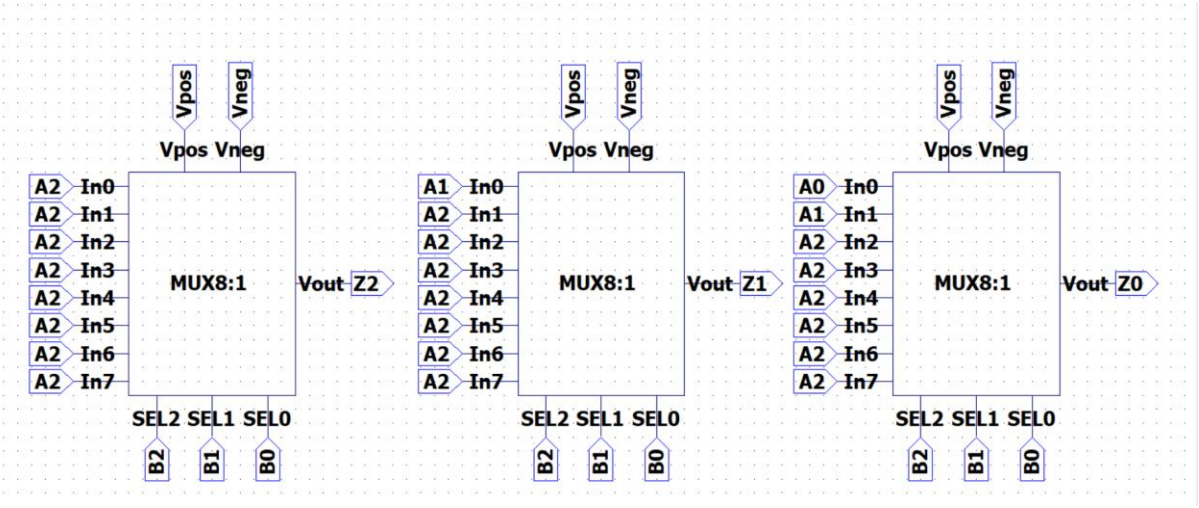
The test calculations and output waveforms are matched exactly from the comparison table. For A>B, A<B and A=B. So, the designed 3-bit Greater than circuit is working properly.

ARITHMETIC SHIFT RIGHT:

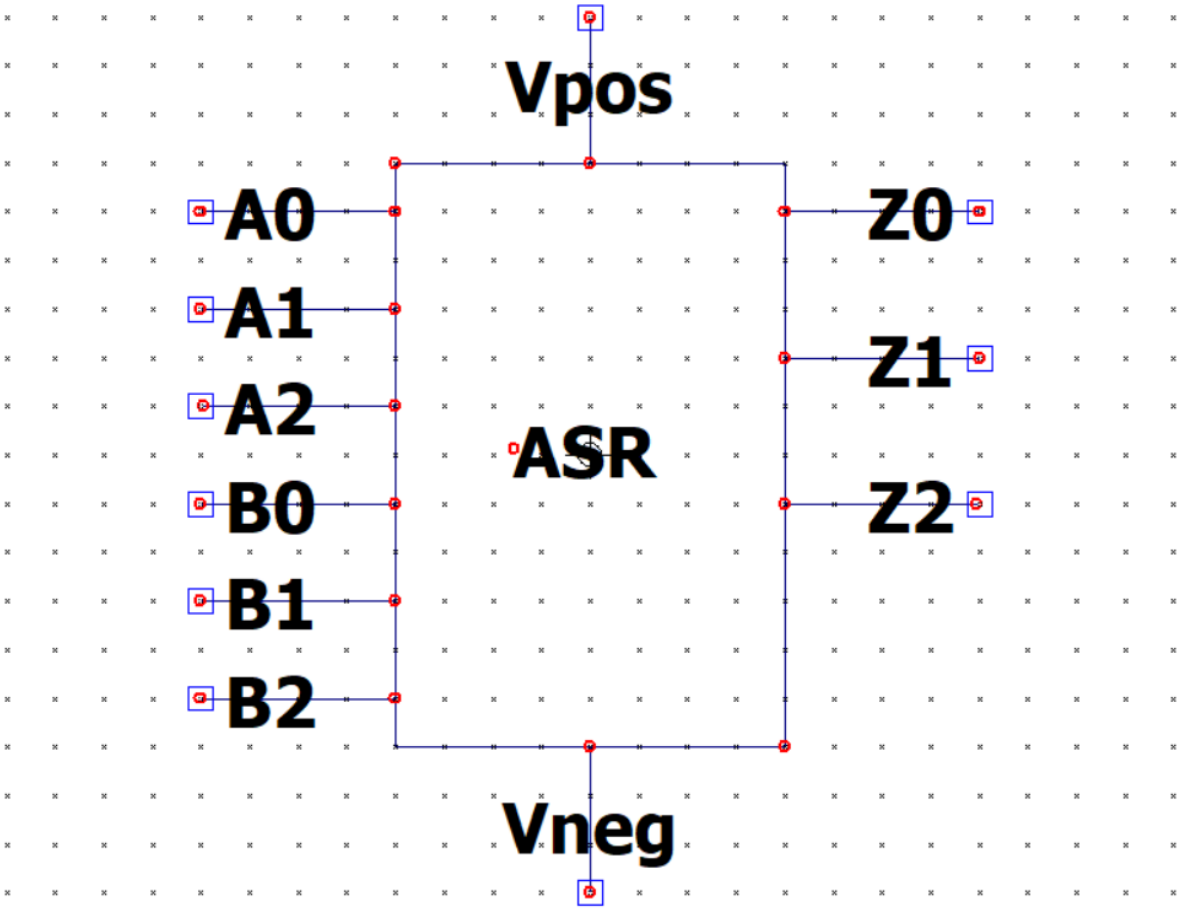
Gate-level drawing:



Symbol schematic:



Symbol drawing:

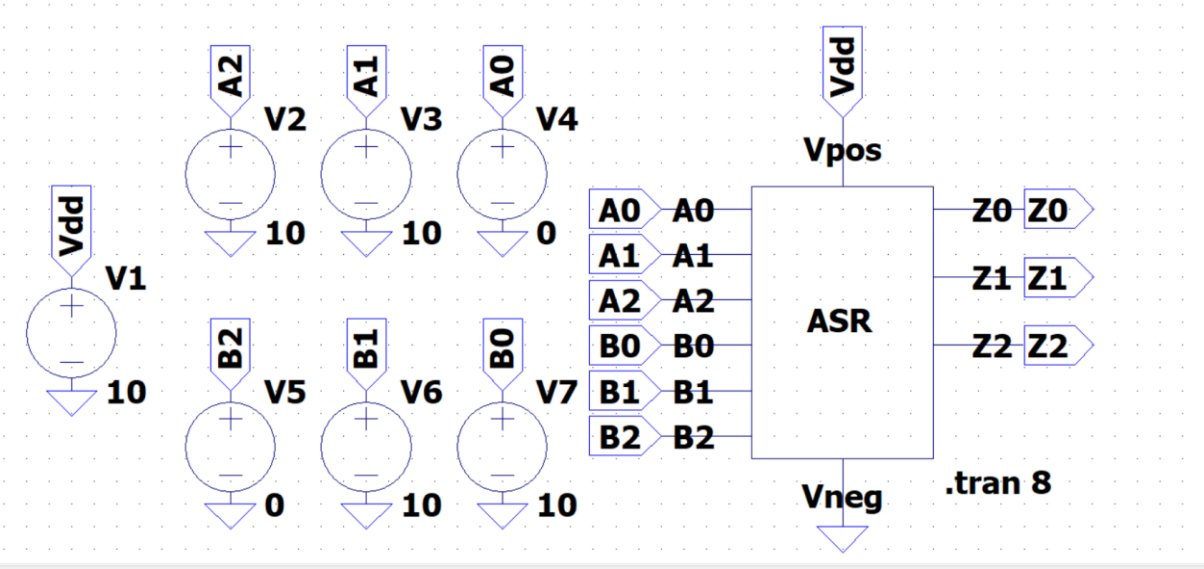


Test Calculation Table:

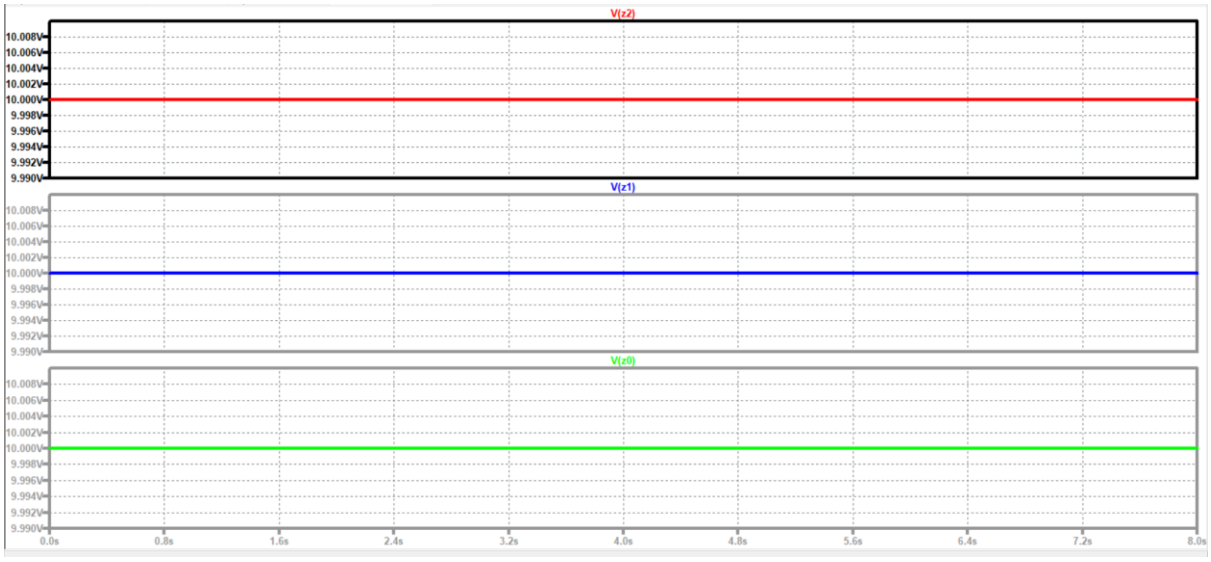
Test Case	A(A2-A0)	B(B2-B0)	Output Z2-Z0
1	110	011	111
2	101	001	110

Test circuits:

Test-1:

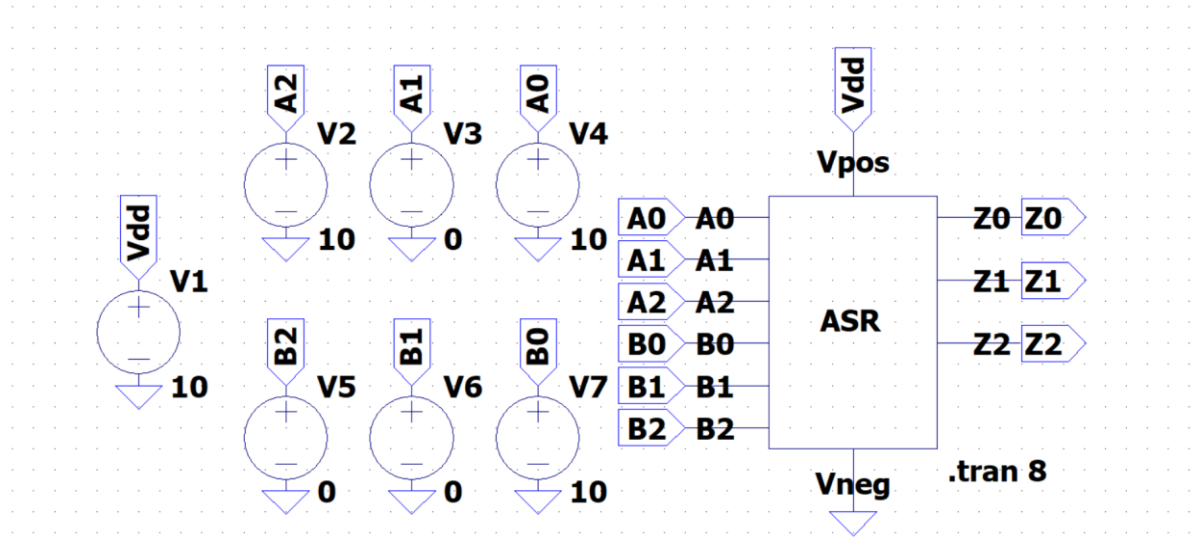


Test-1 circuit waveforms:

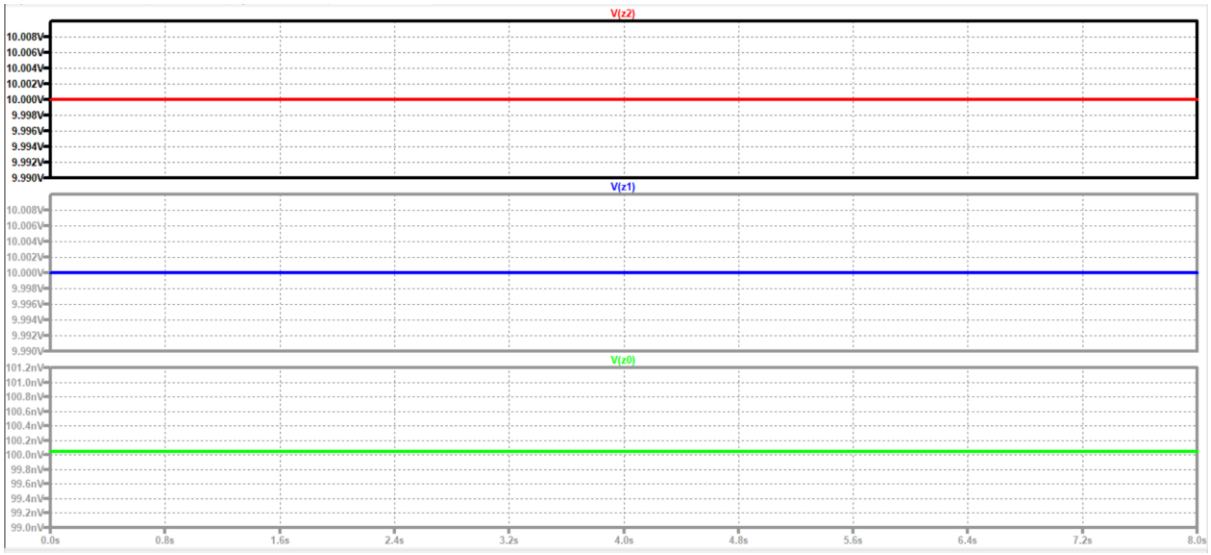


Test Case	A(A2-A0)	B(B2-B0)	Calculation Z2-Z0	Waveform Z2-Z0
1	110	011	111	111

Test-2:



Test-2 circuit waveforms:



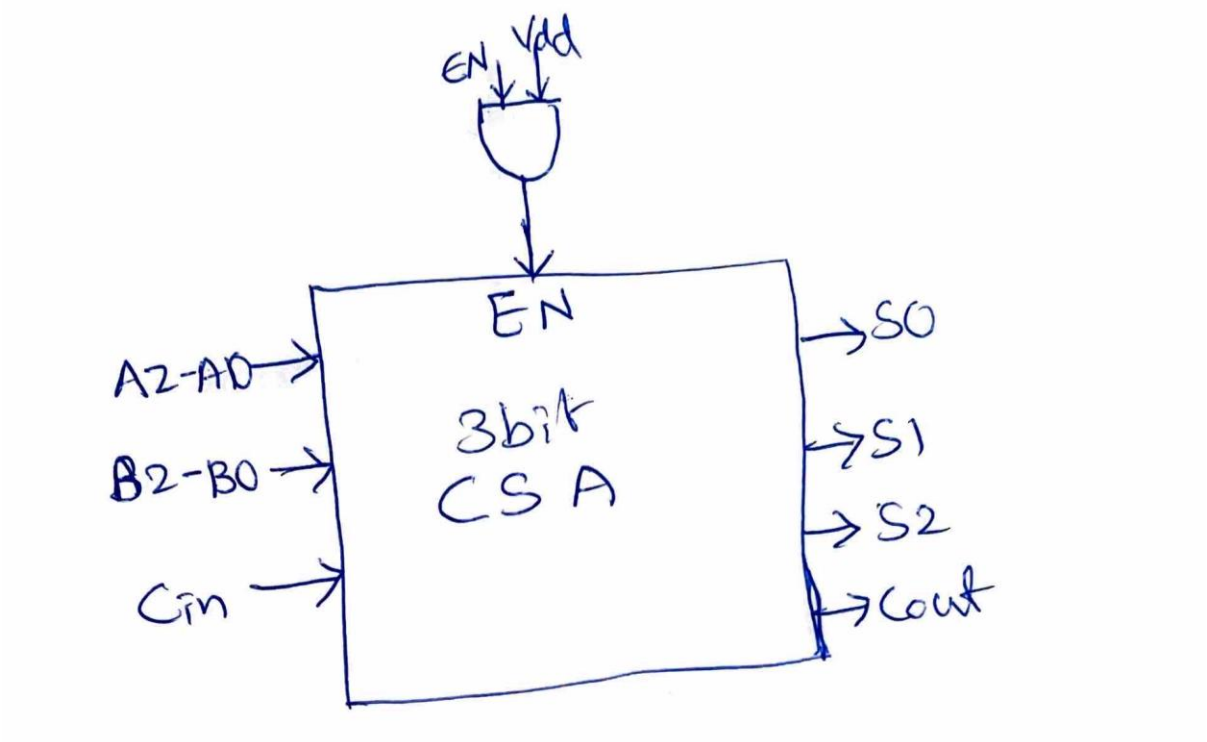
Test Case	A(A2-A0)	B(B2-B0)	Calculation Z2-Z0	Waveform Z2-Z0
1	101	001	110	110

Verification statement:

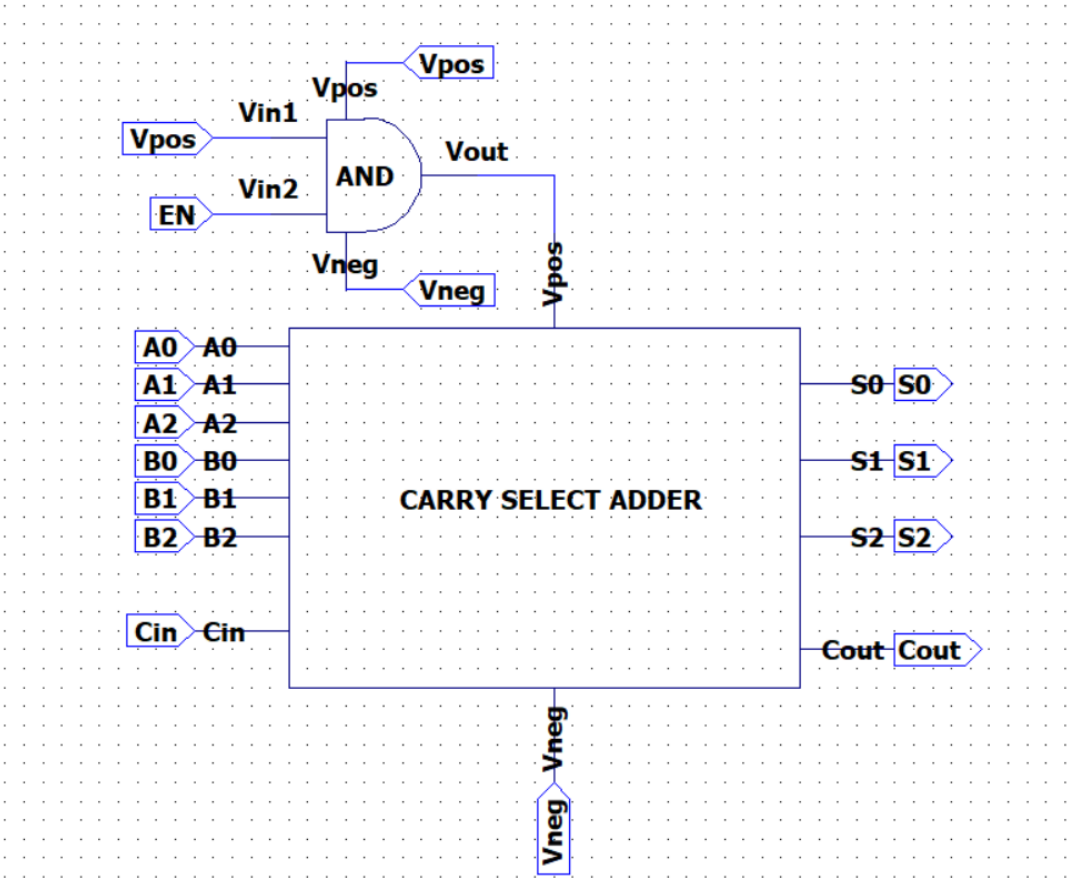
The test calculations and output waveforms are matched exactly from the comparison table. Hence, the designed 3-bit Arithmetic Shift Right Register circuit is working properly.

ADDER ENABLE:

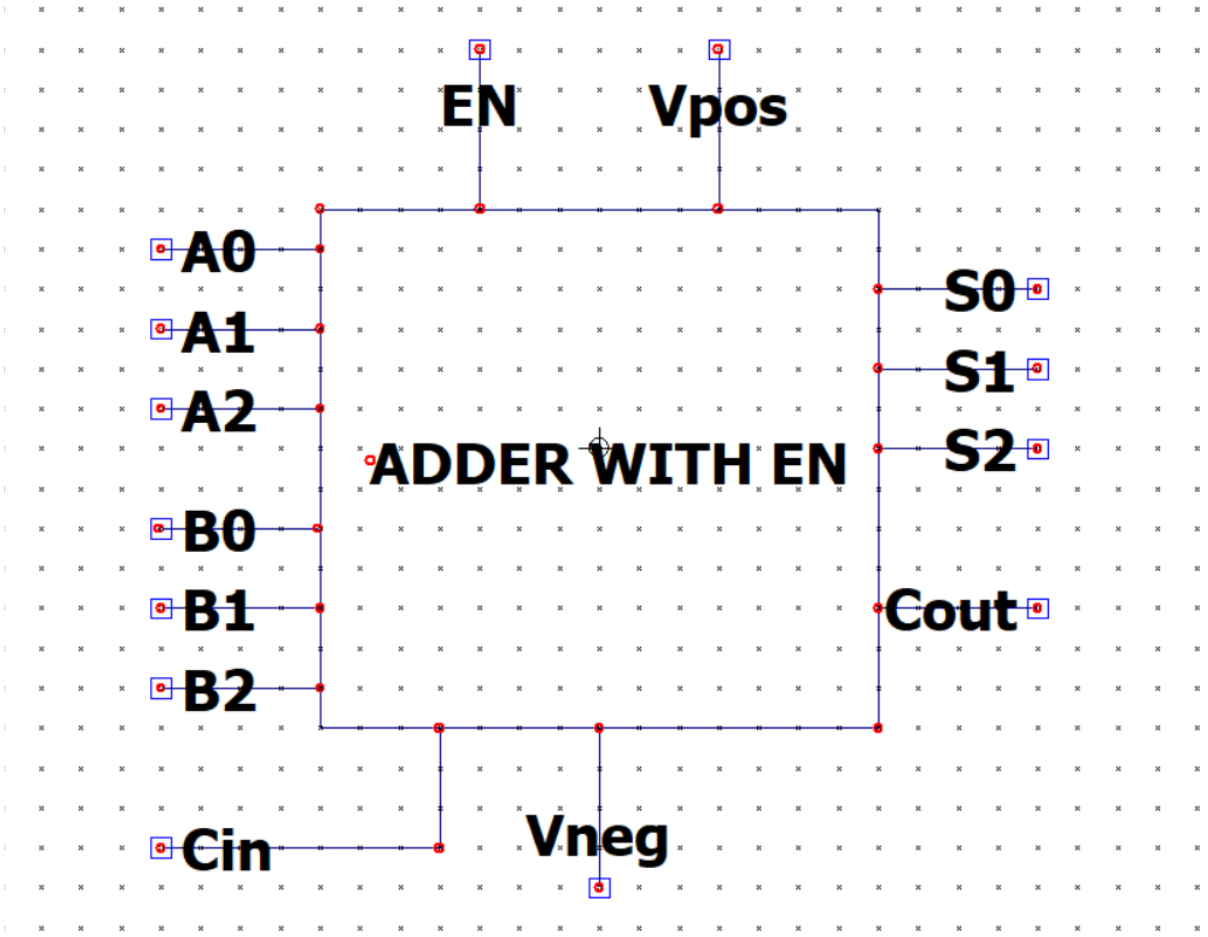
Gate-level drawing:



Symbol schematic:



Symbol drawing:

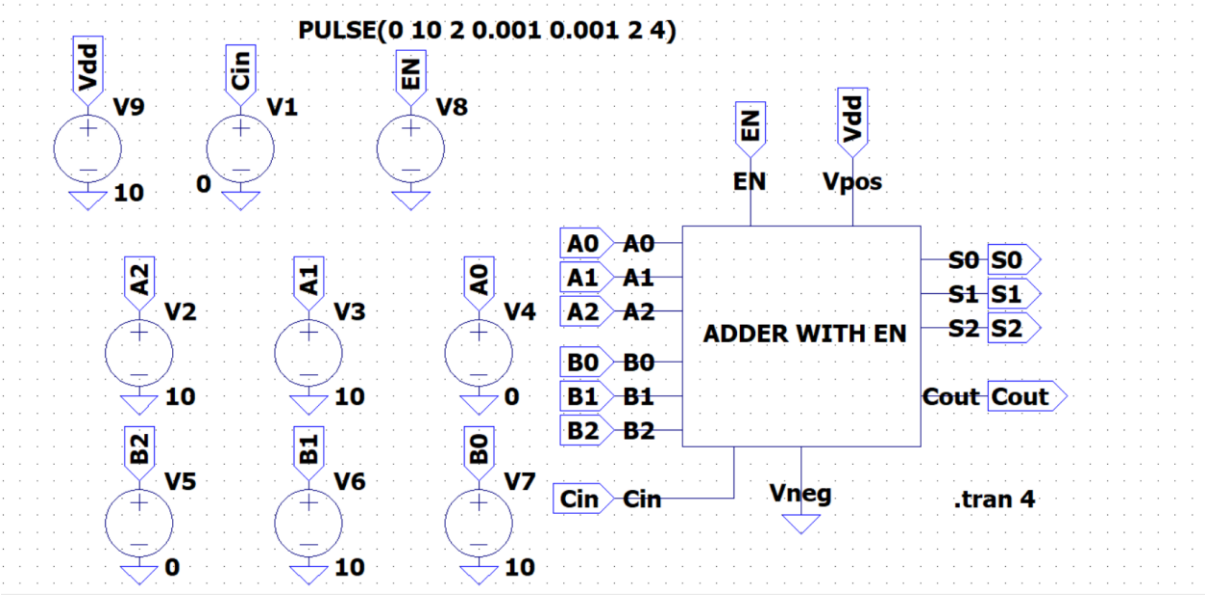


Test Calculation Table:

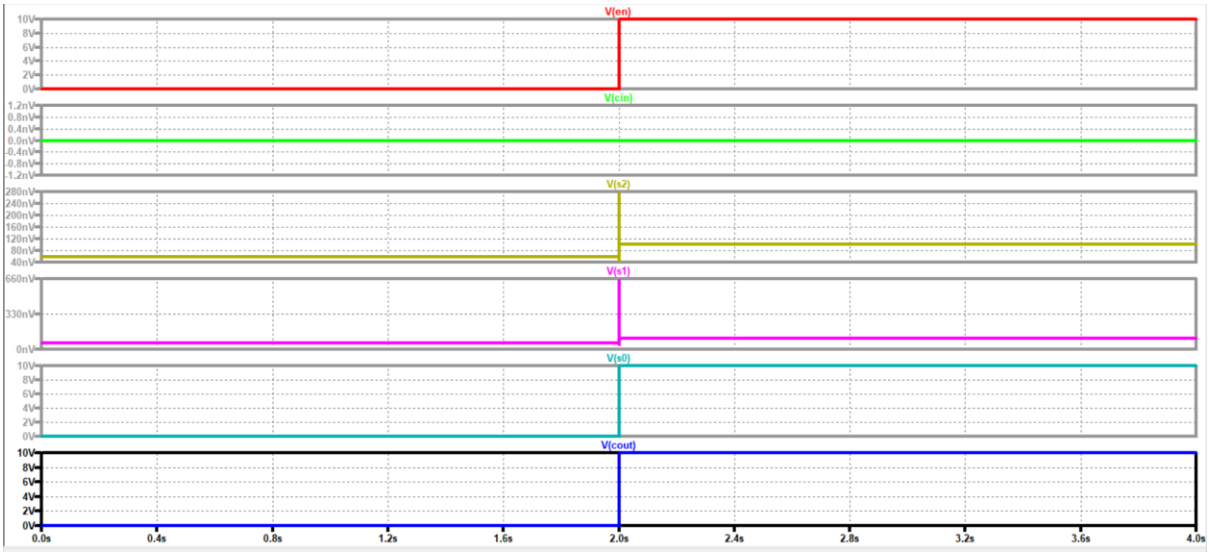
Test Case	A(A2-A0)	B(B2-B0)	Cin	Enable (En)	SUM(S2-S0)	Cout
1	110	011	0	0	000	0
	110	011	0	1	001	1
2	110	011	1	0	000	0
	110	011	1	1	010	1

Test Circuits:

Test circuit-1: A=110; B-011 and Cin=0

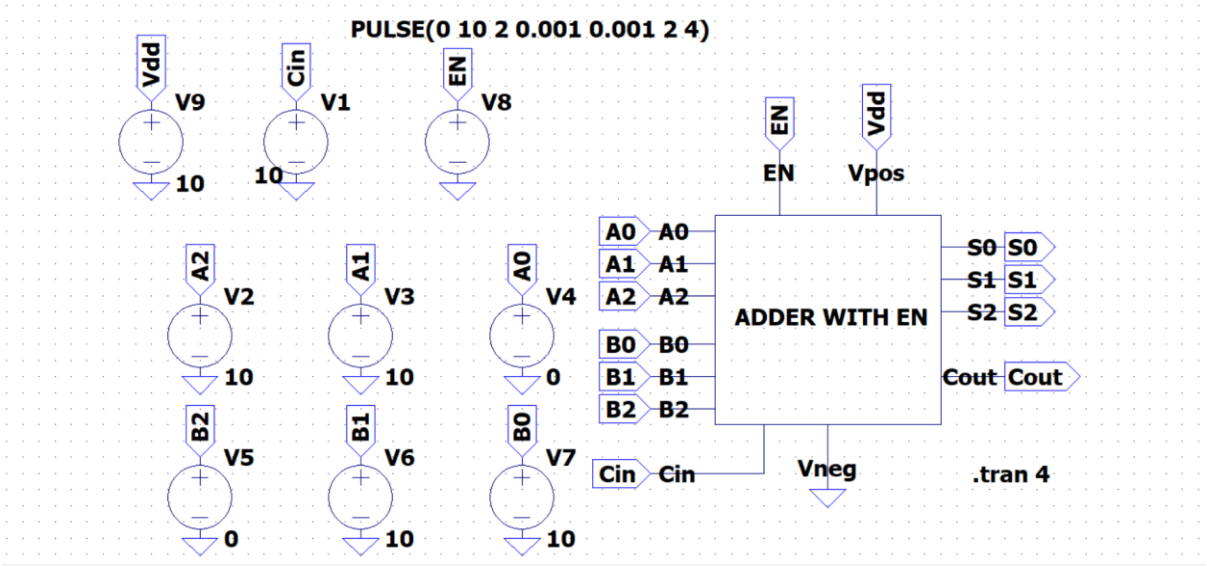


Test-1 circuit waveforms:

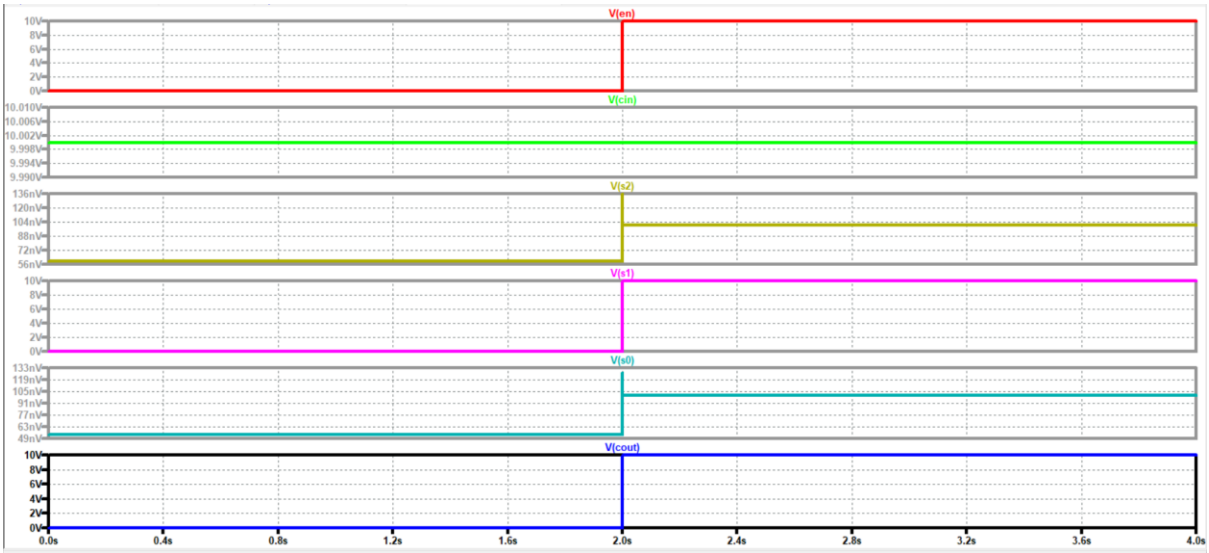


Test Case	A (A2-A0)	B (B2-B0)	Cin	Enable (EN)	Calculations		Waveform	
					SUM (S2-S0)	Cout	SUM (S2-S0)	Cout
1	110	011	0	0	000	0	000	0
	110	011	0	1	001	1	001	1

Test circuit-2: A=110; B-011 and Cin=1



Test-2 circuit waveforms:



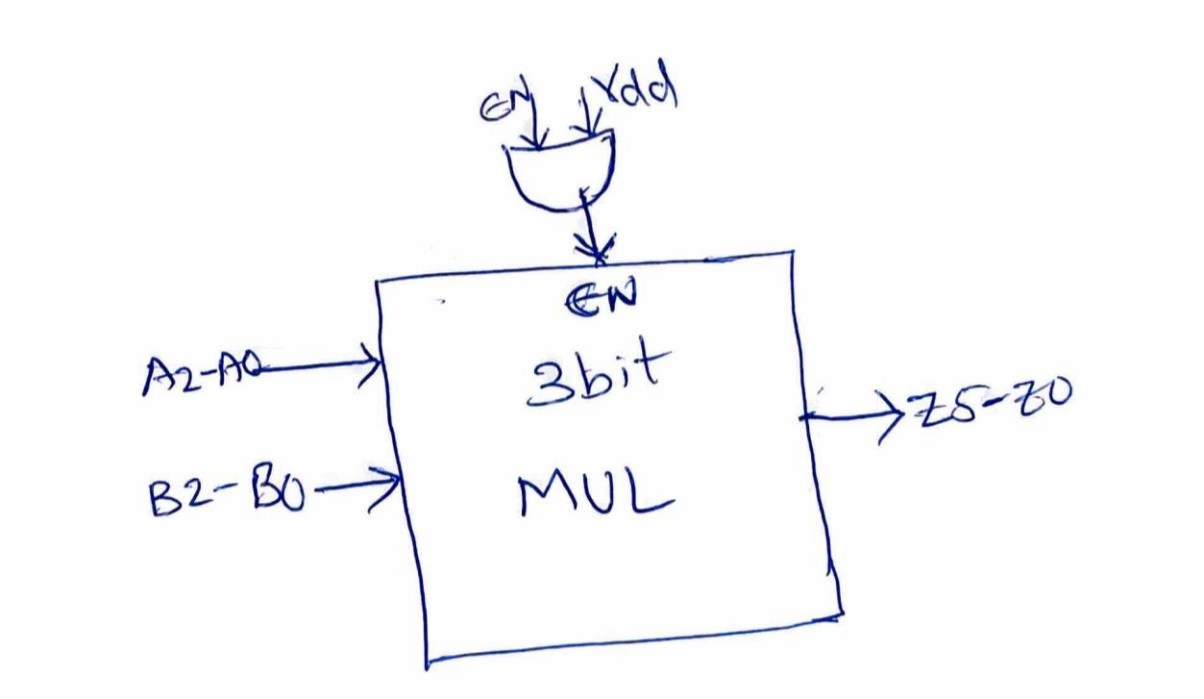
Test Case	A (A2-A0)	B (B2-B0)	Cin	Enable (EN)	Calculations		Waveform	
					SUM (S2-S0)	Cout	SUM (S2-S0)	Cout
1	110	011	1	0	000	0	000	0
	110	011	1	1	010	1	010	1

Verification statement:

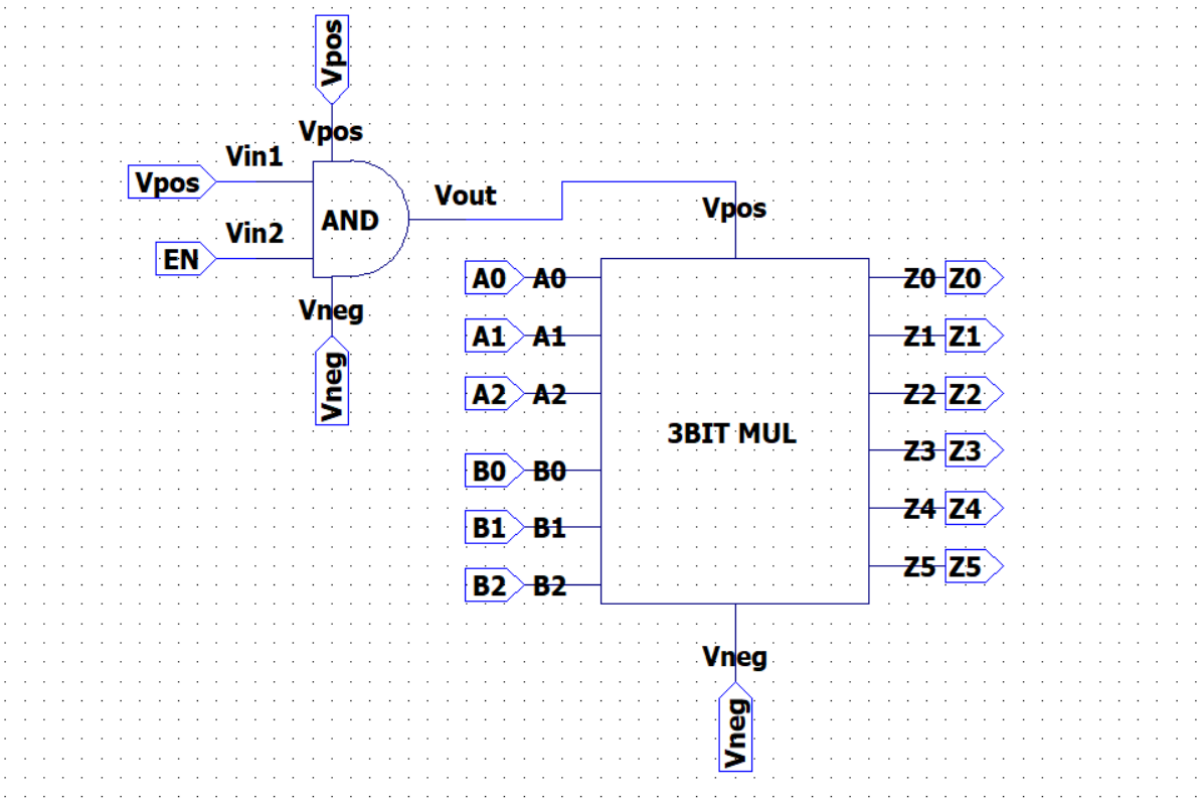
The test calculations and output waveforms are matched exactly from the comparison table. So, the designed 3-bit Carry Select Adder with enable is working properly.

MULTIPLIER ENABLE:

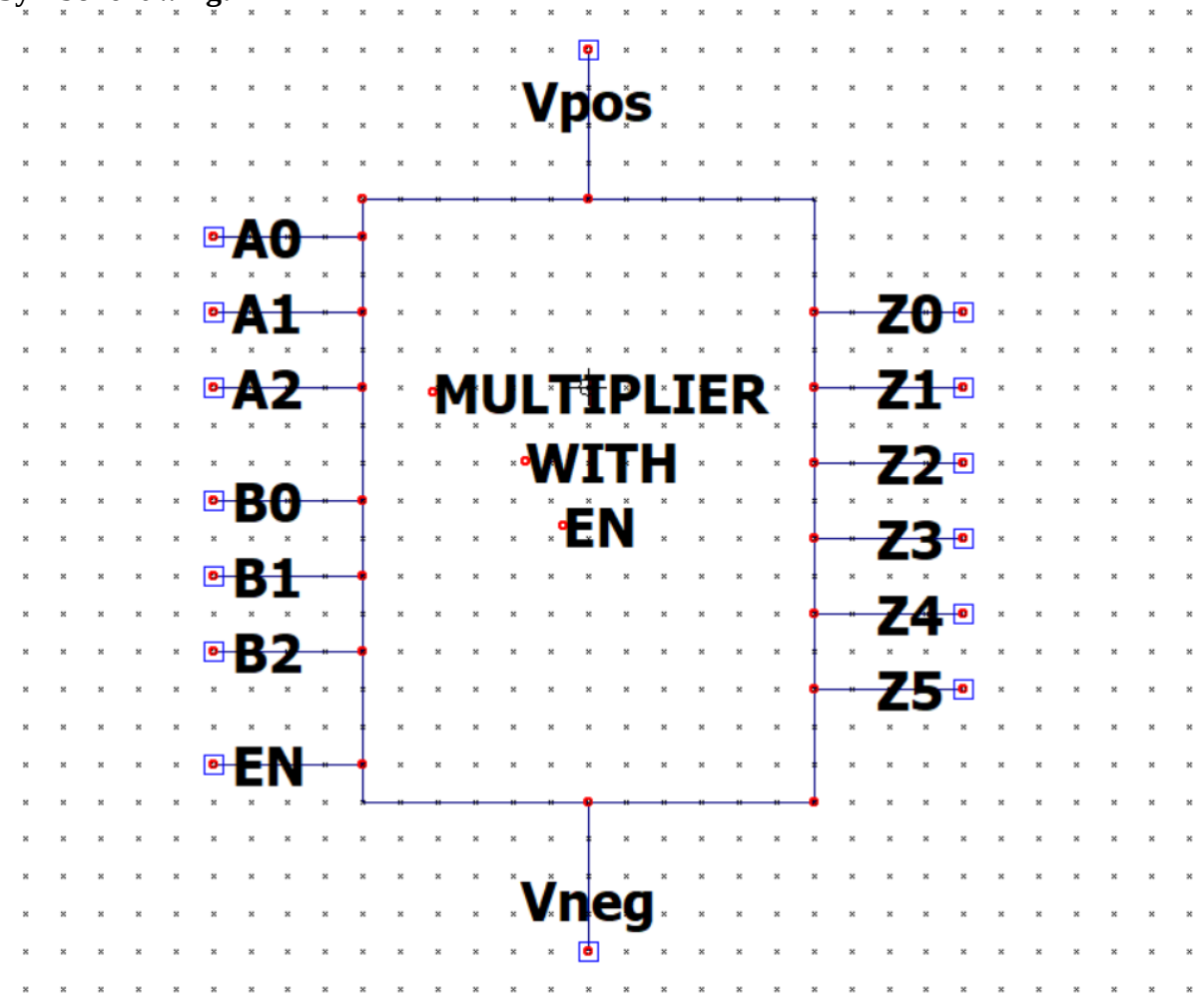
Gate-level drawing:



Symbol schematic:



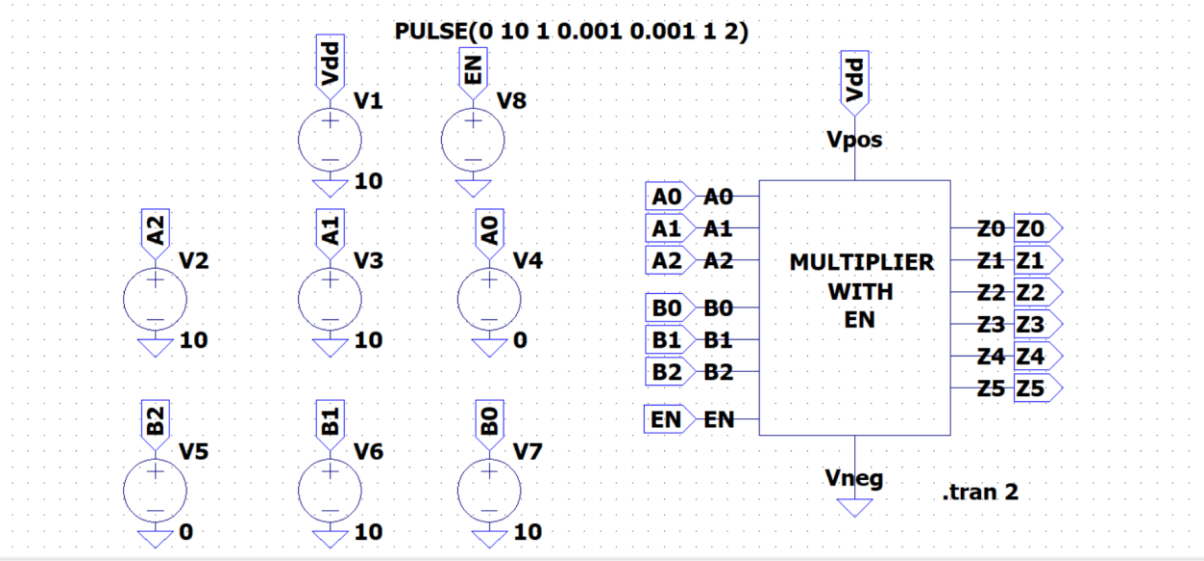
Symbol drawing:



Test Calculation Table:

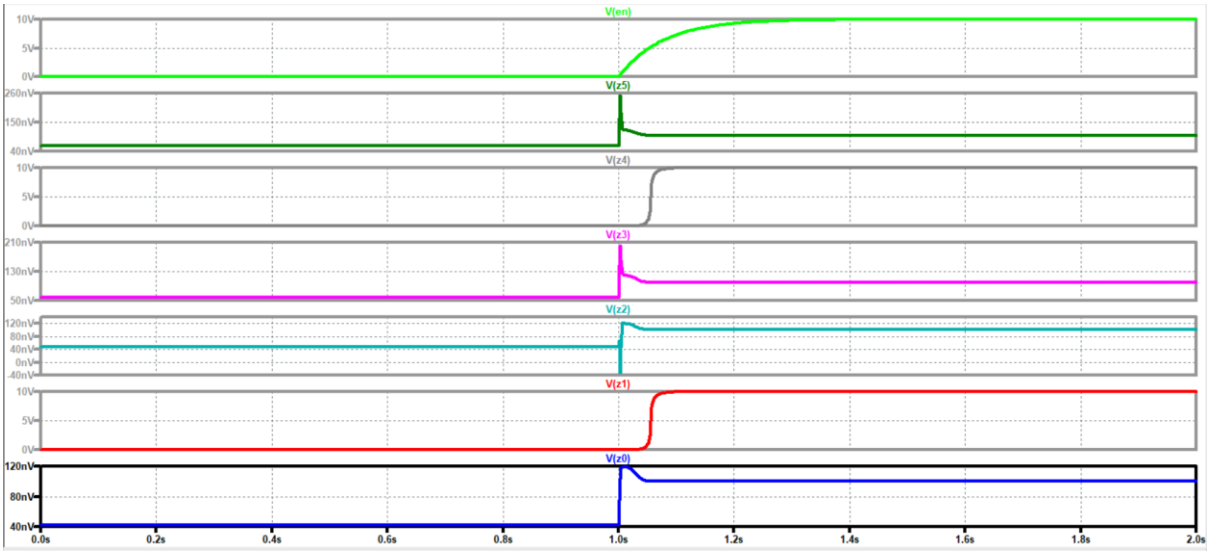
Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Output Z5-Z0
1	110	011	0	000000
	110	011	1	010010

Test circuit:



Test circuit waveforms:

I have added parasitic properties to the Enable pulse.



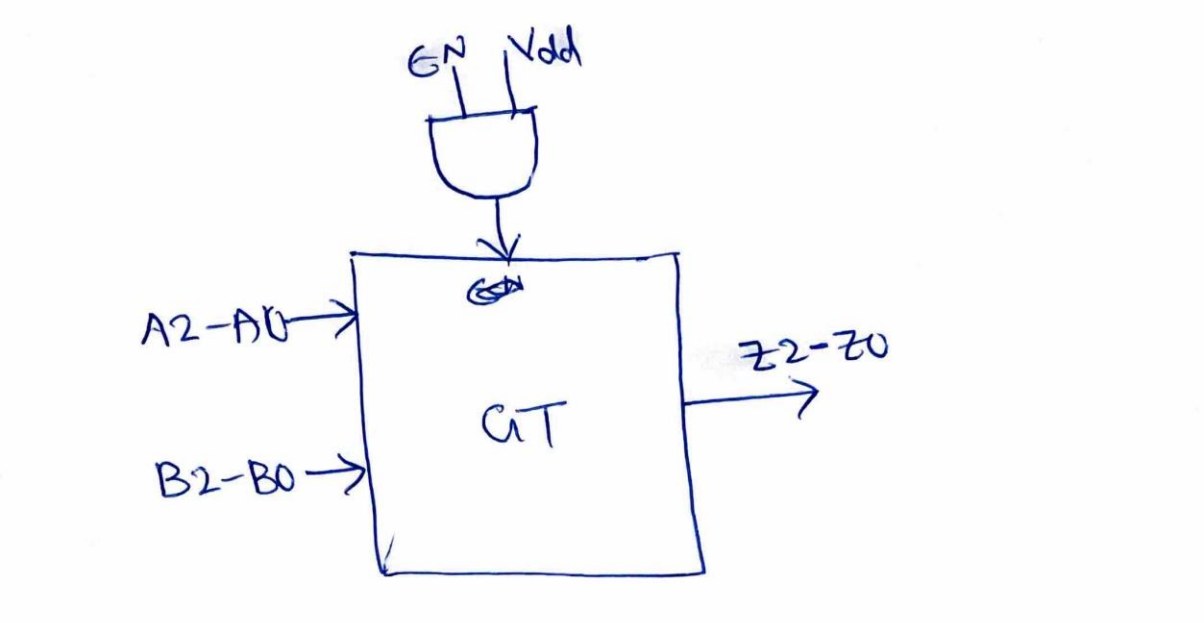
Test Case	A(A2-A0)	B(B2-B0)	Enable	Calculation Z5-Z0	Waveform Z5-Z0
1	110	011	0	000000	000000
	110	011	1	010010	010010

Verification statement:

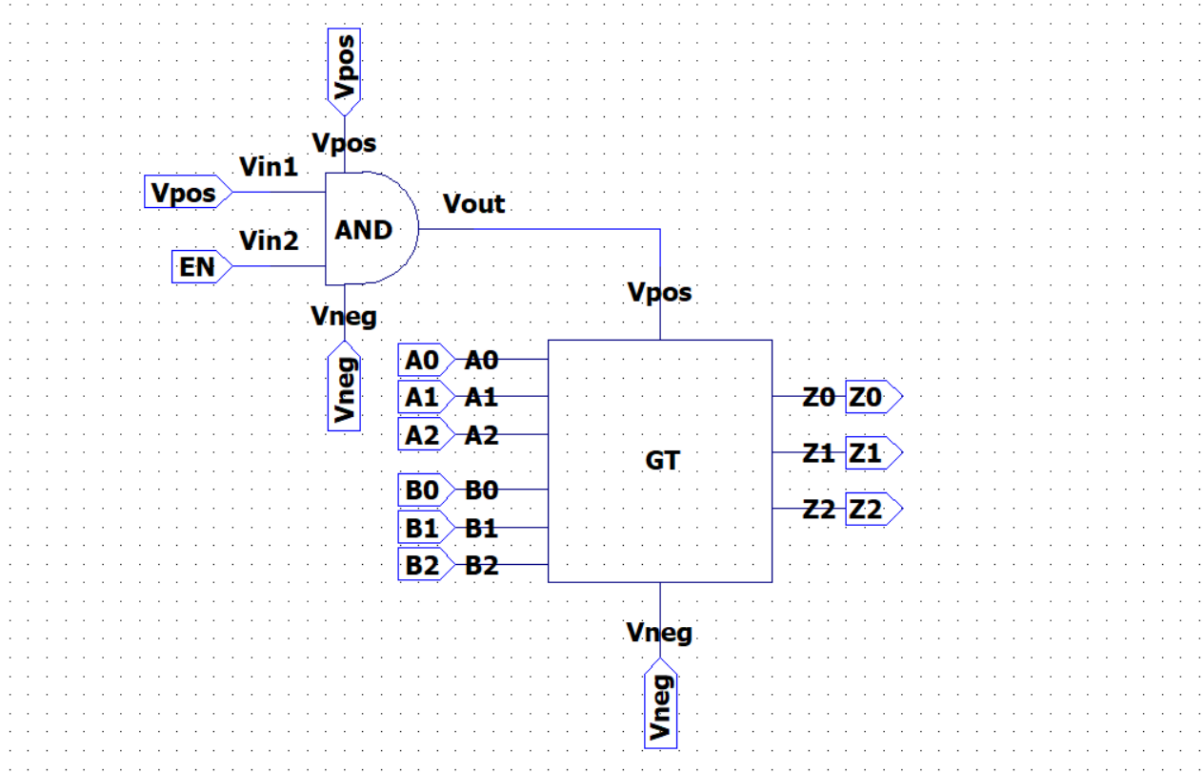
The test calculations and output waveforms are matched exactly from the comparison table. So, the designed 3-bit Multiplier enable is working properly.

GREATER THAN ENABLE:

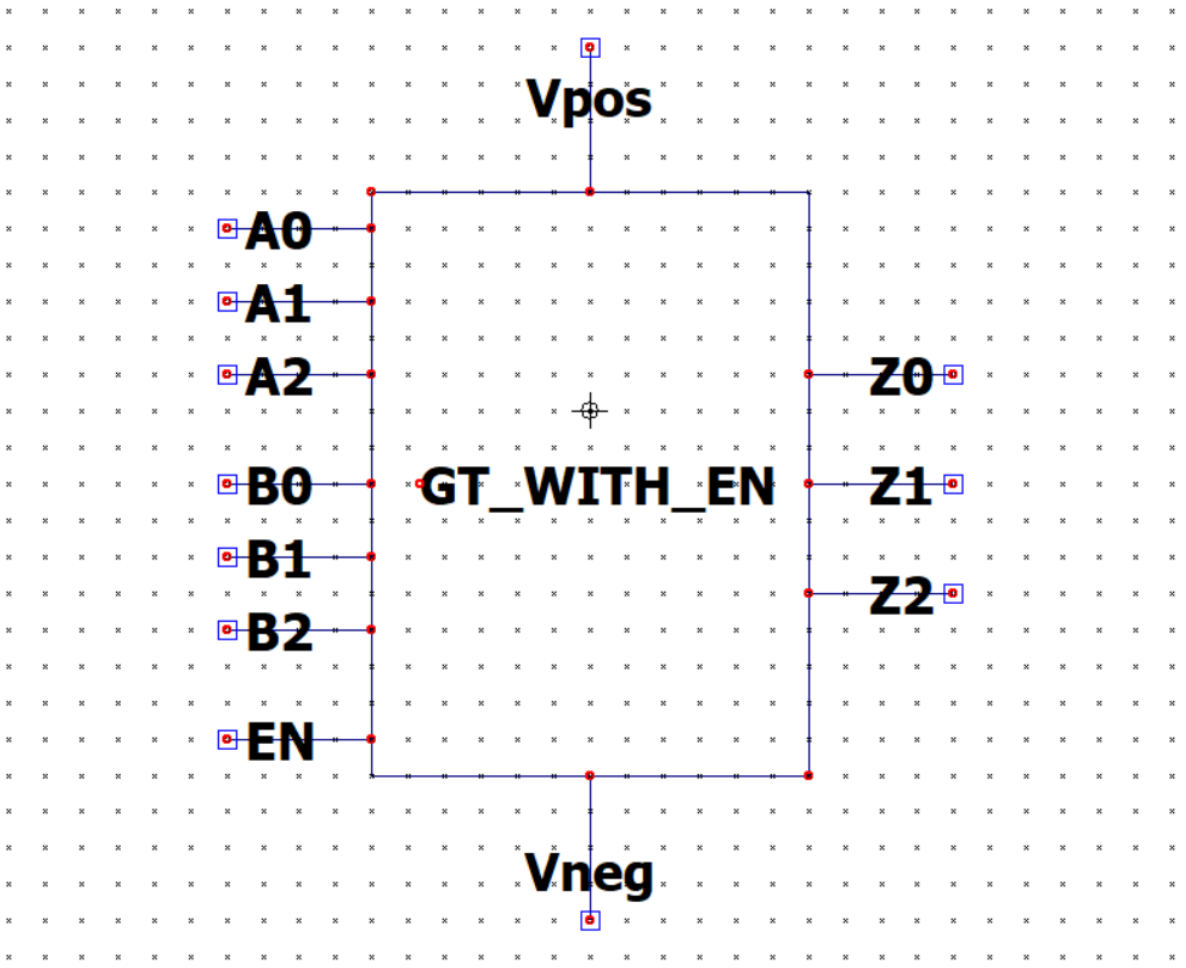
Gate-level drawing:



Symbol schematic:



Symbol drawing:

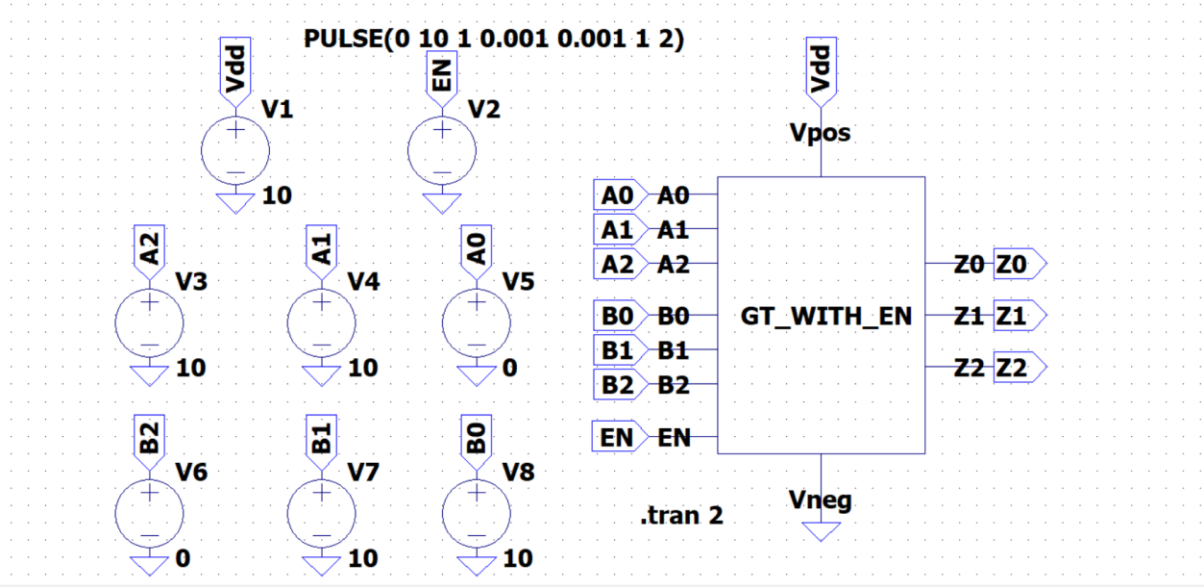


Test Calculation Table:

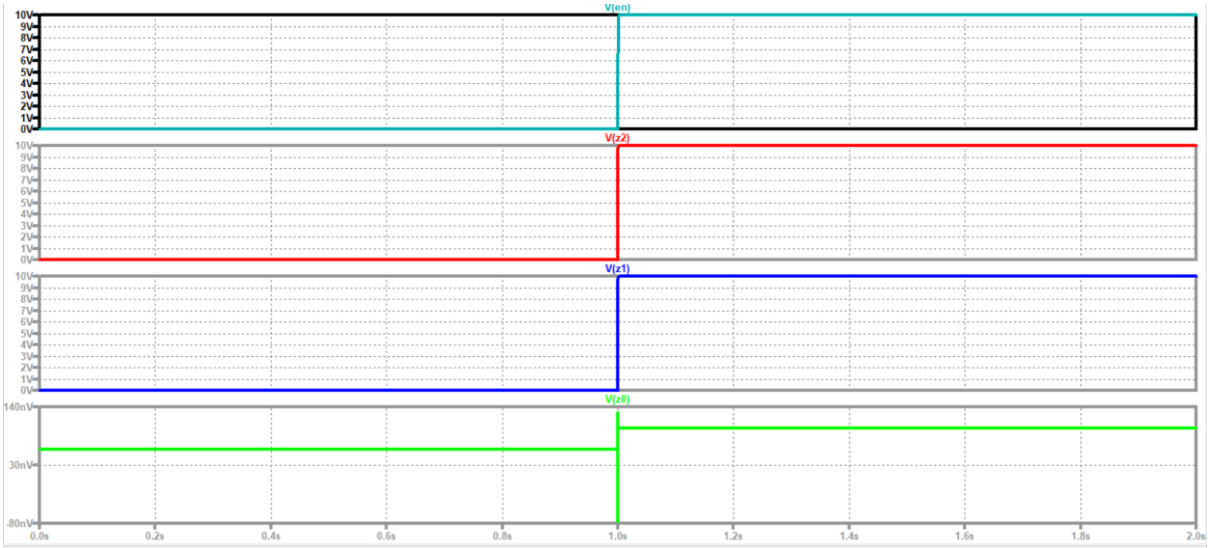
Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Output Z2-Z0
1	110	011	0	000
	110	011	1	110(A>B)
2	011	110	0	000
	011	110	1	111(A<B)
3	110	110	0	000
	110	110	1	111(A=B)

Test circuits:

Test-1:A=110 and B=011

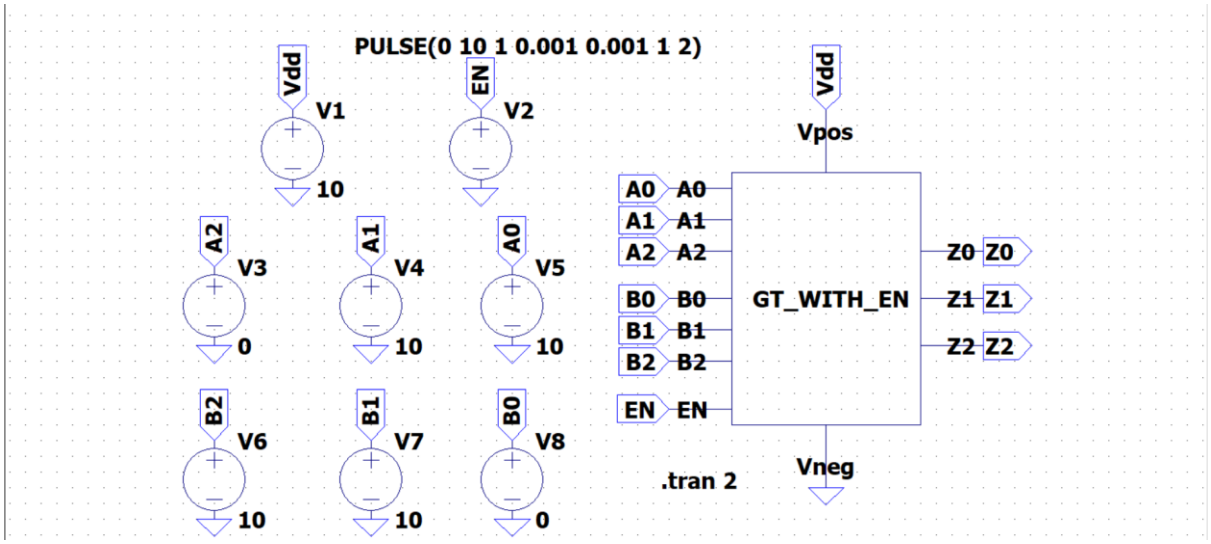


Test-1 circuit waveforms:

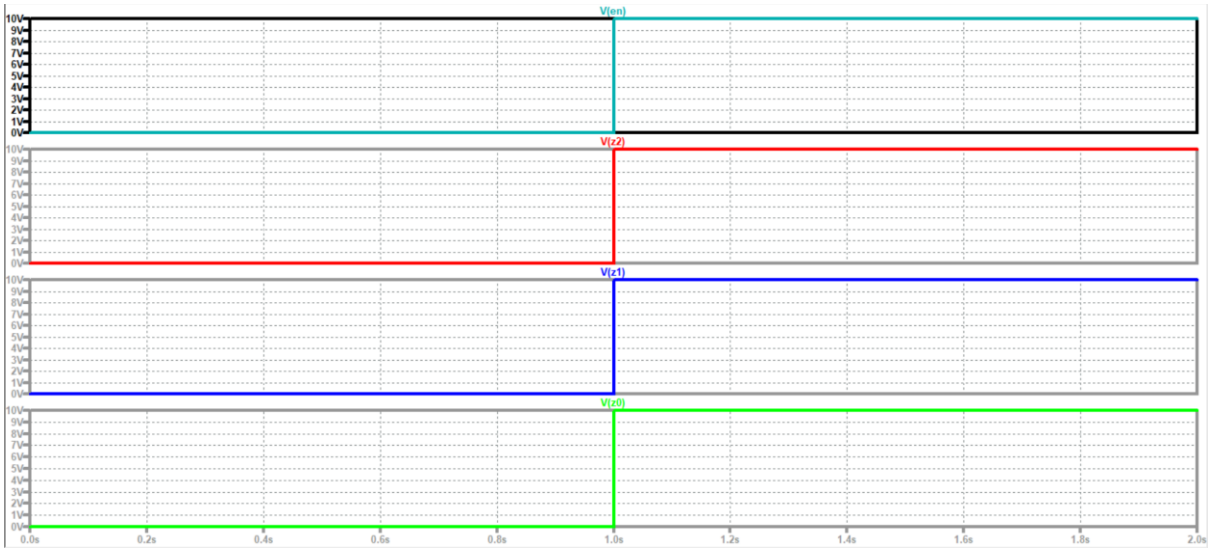


Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Calculation Z2-Z0	Waveform Z2-Z0
1	110	011	0	000	000
	110	011	1	110	110

Test-2: A=011 and B=110

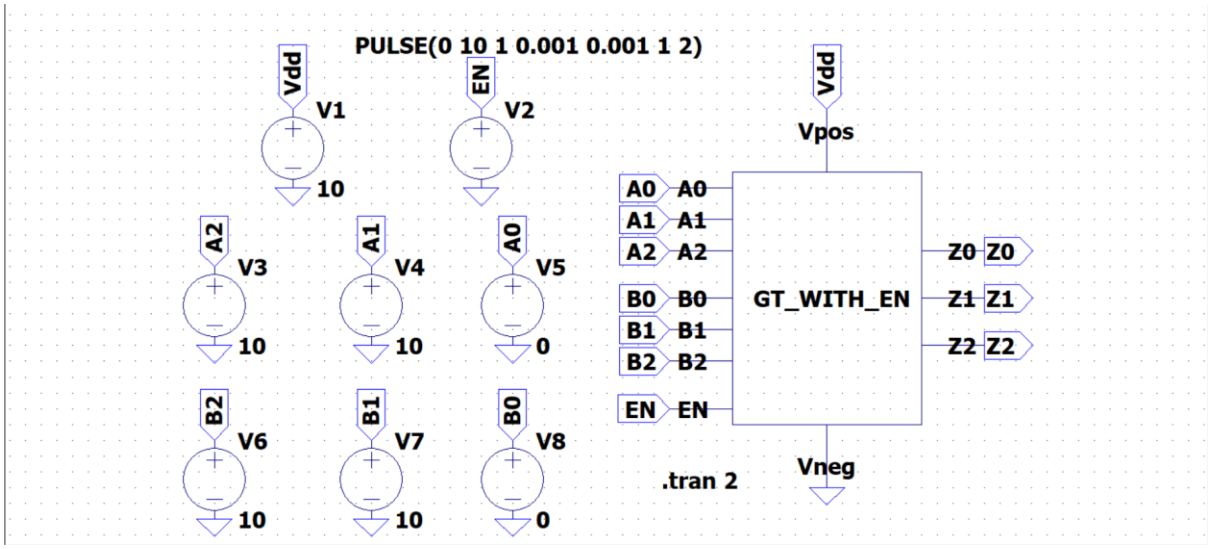


Test-2 Waveforms:

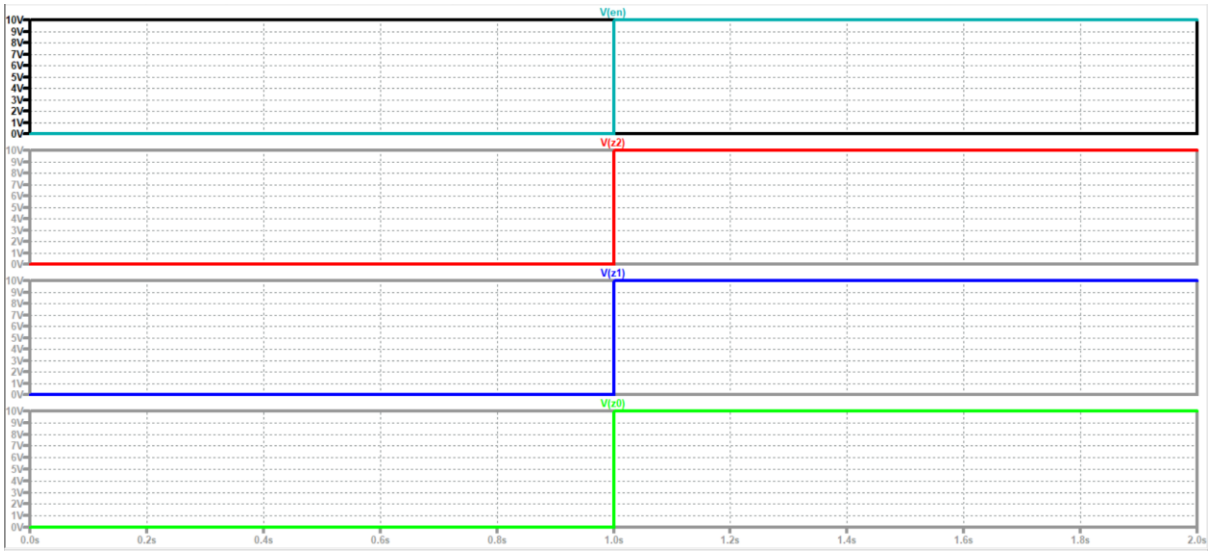


Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Calculation Z2-Z0	Waveform Z2-Z0
1	011	110	0	000	000
	011	110	1	111	111

Test-3: A=110 and B=110



Test-3 Waveforms:



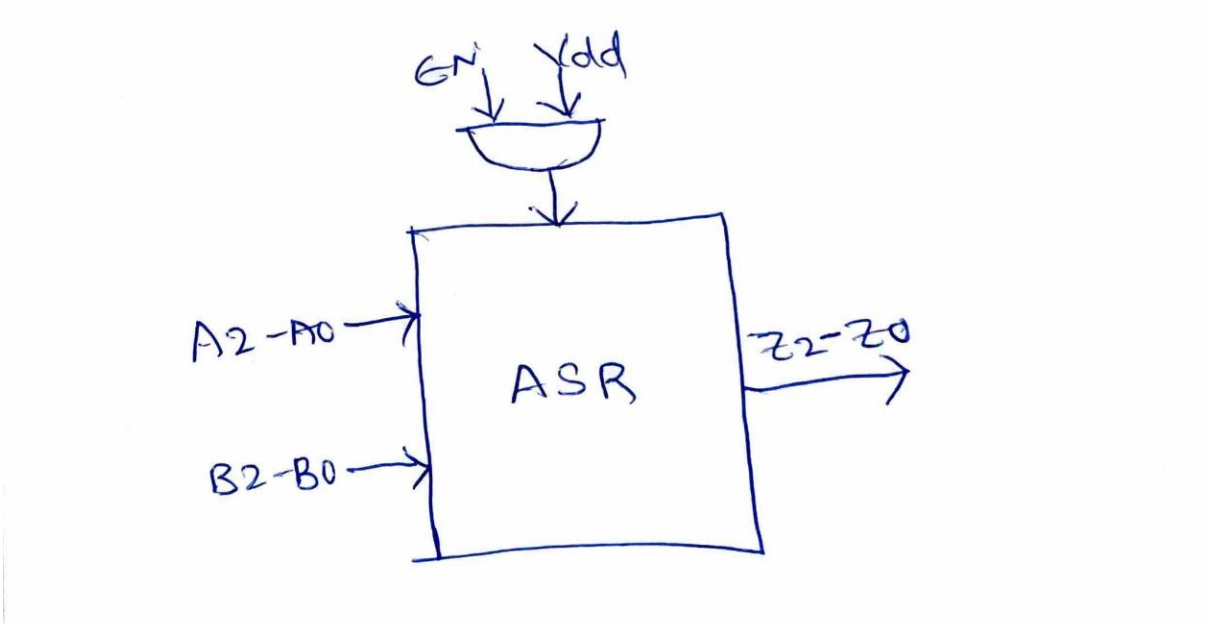
Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Calculation Z2-Z0	Waveform Z2-Z0
1	110	110	0	000	000
	110	110	1	111	111

Verification statement:

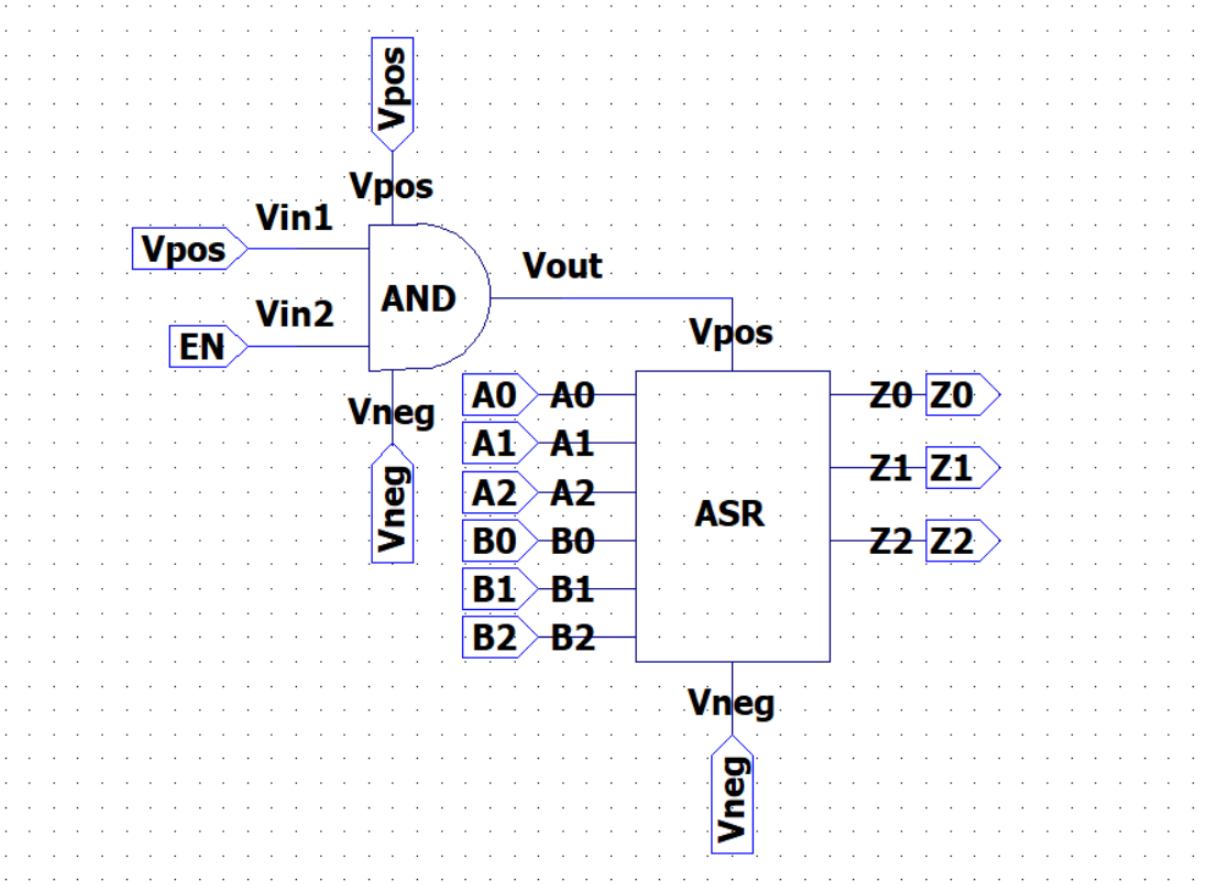
The test calculations and output waveforms are matched exactly from the comparison table. For A>B, A<B and A=B. So, the designed 3-bit Greater than enable circuit is working properly.

ARITHMETIC SHIFT REGISTER ENABLE:

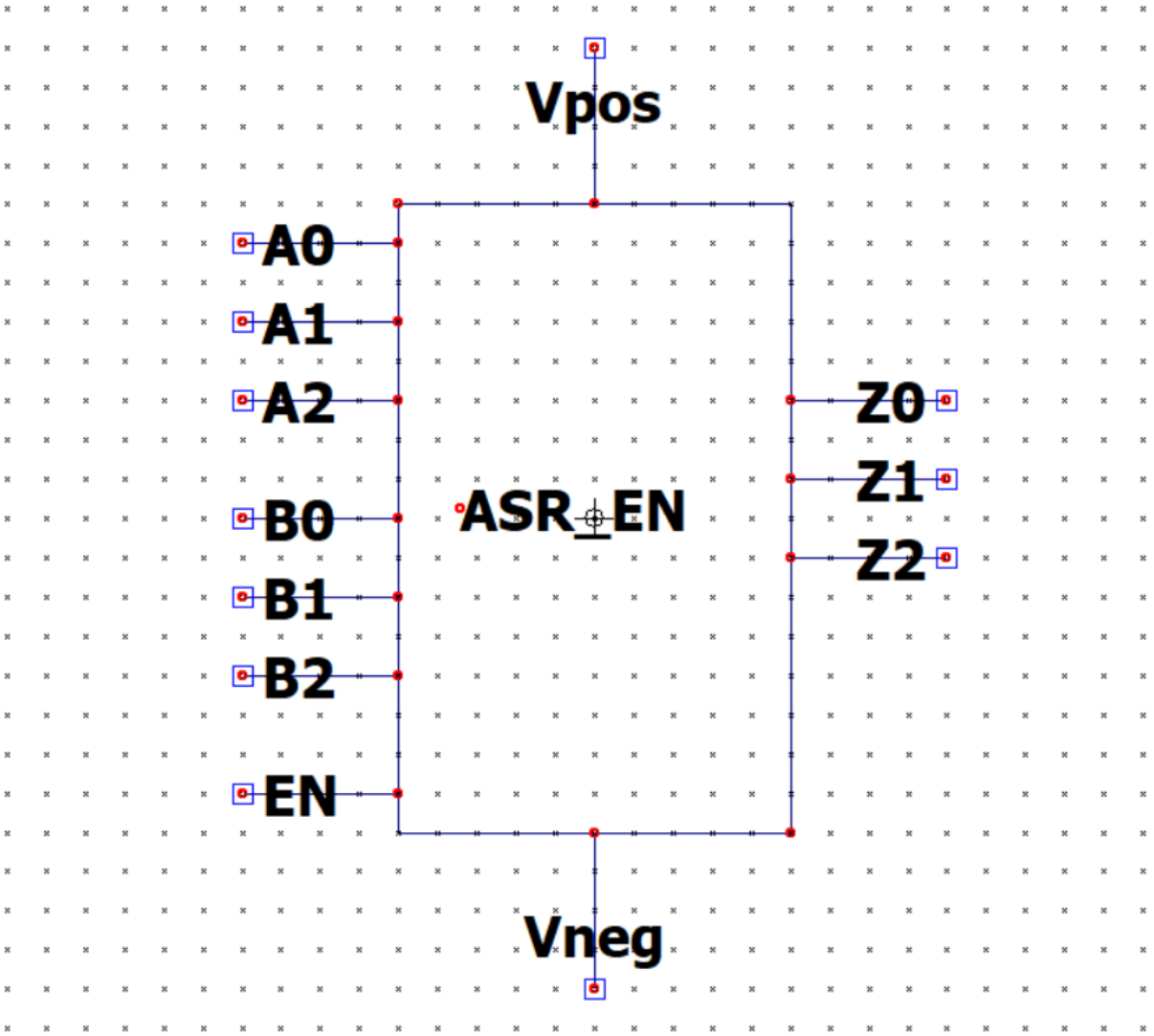
Gate-level drawing:



Symbol schematic:



Symbol drawing:

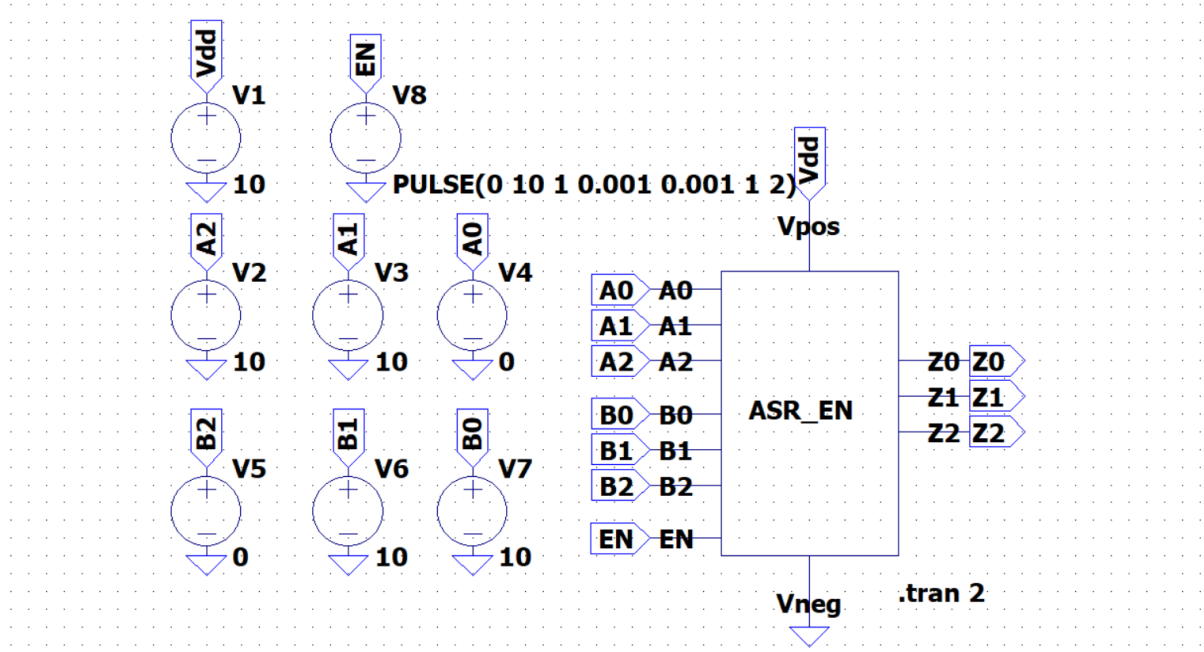


Test Calculation Table:

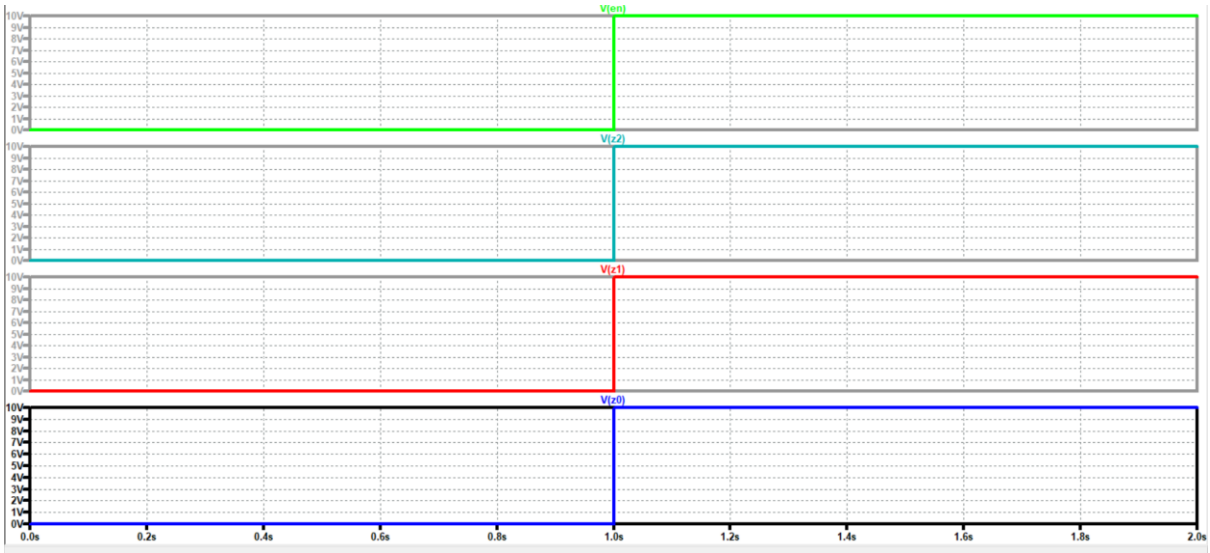
Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Output Z2-Z0
1	110	011	0	000
	110	011	1	111
2	101	001	0	000
	101	001	1	110

Test circuits:

Test-1:

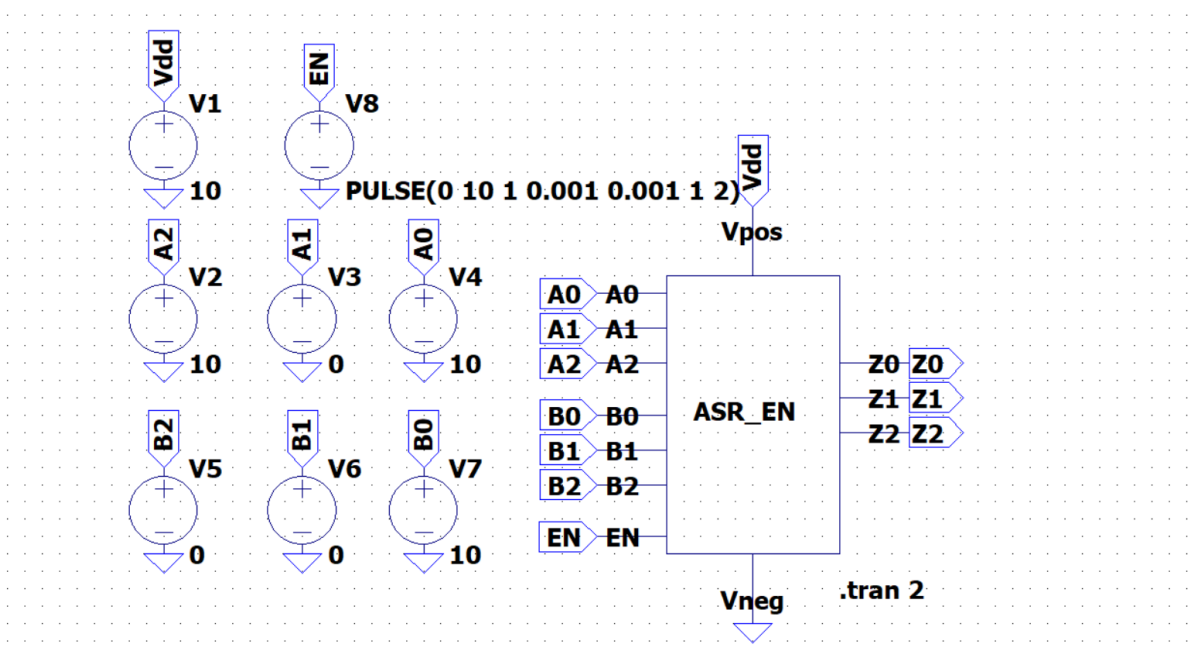


Test-1 circuit waveforms:

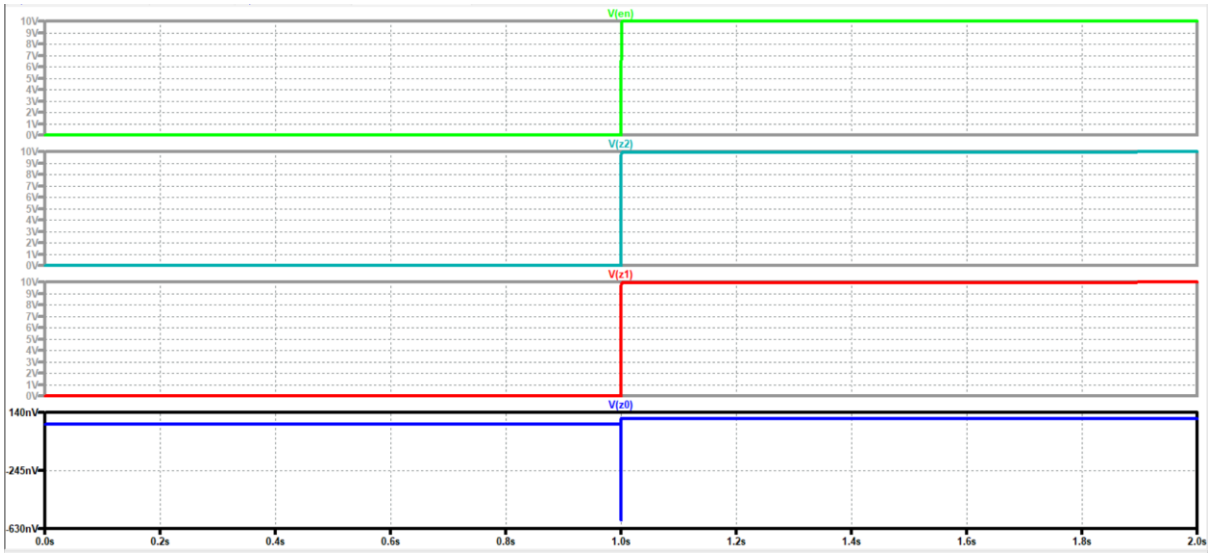


Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Calculation Z2-Z0	Waveform Z2-Z0
1	110	011	0	000	000
	110	011	1	111	111

Test-2:



Test-2 circuit waveforms:



Test Case	A(A2-A0)	B(B2-B0)	Enable (EN)	Calculation Z2-Z0	Waveform Z2-Z0
1	101	001	0	000	000
	101	001	1	110	110

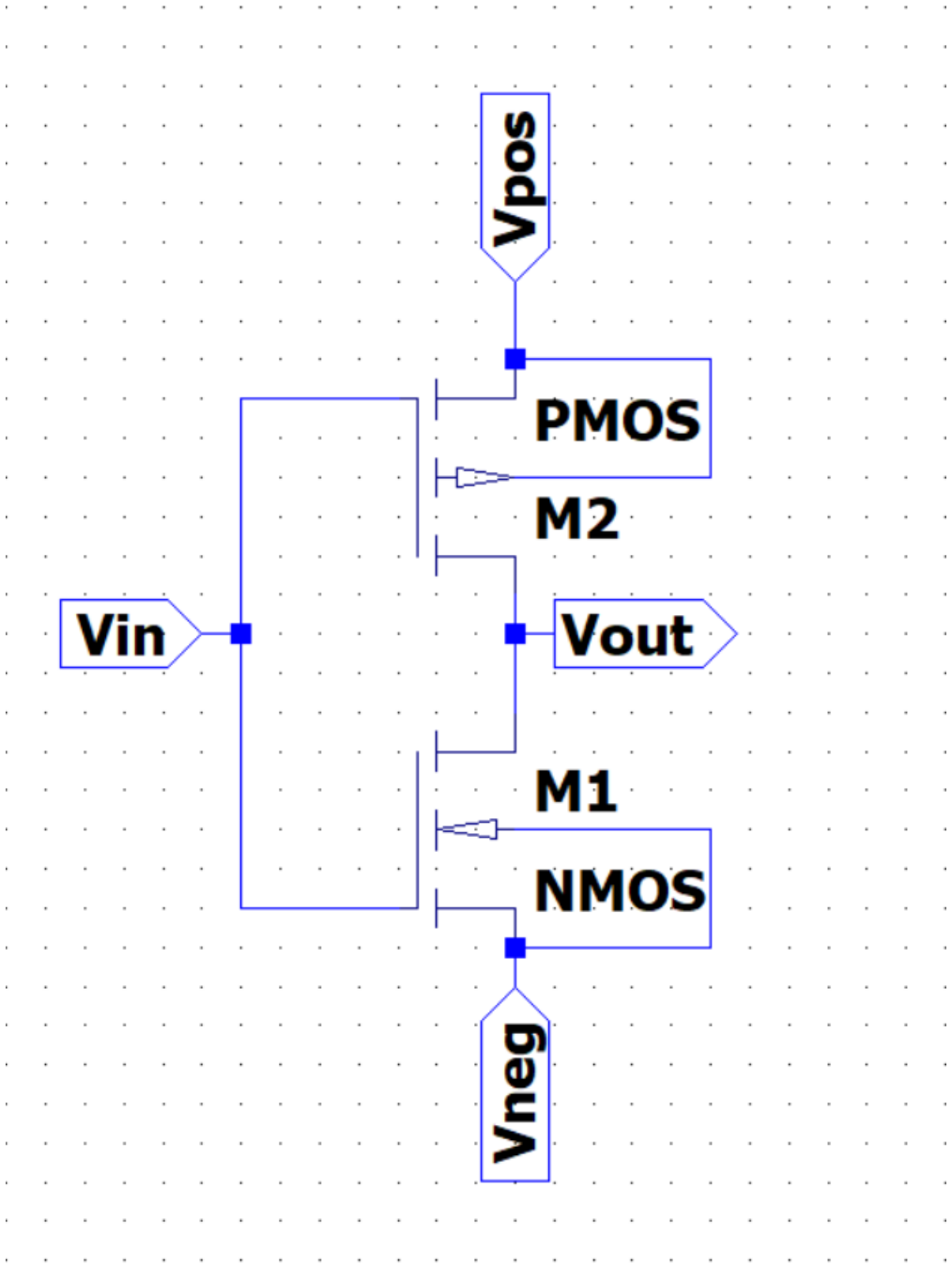
Verification statement:

The test calculations and output waveforms are matched exactly from the comparison table. Hence, the designed 3-bit Arithmetic Shift Right Register circuit is working properly.

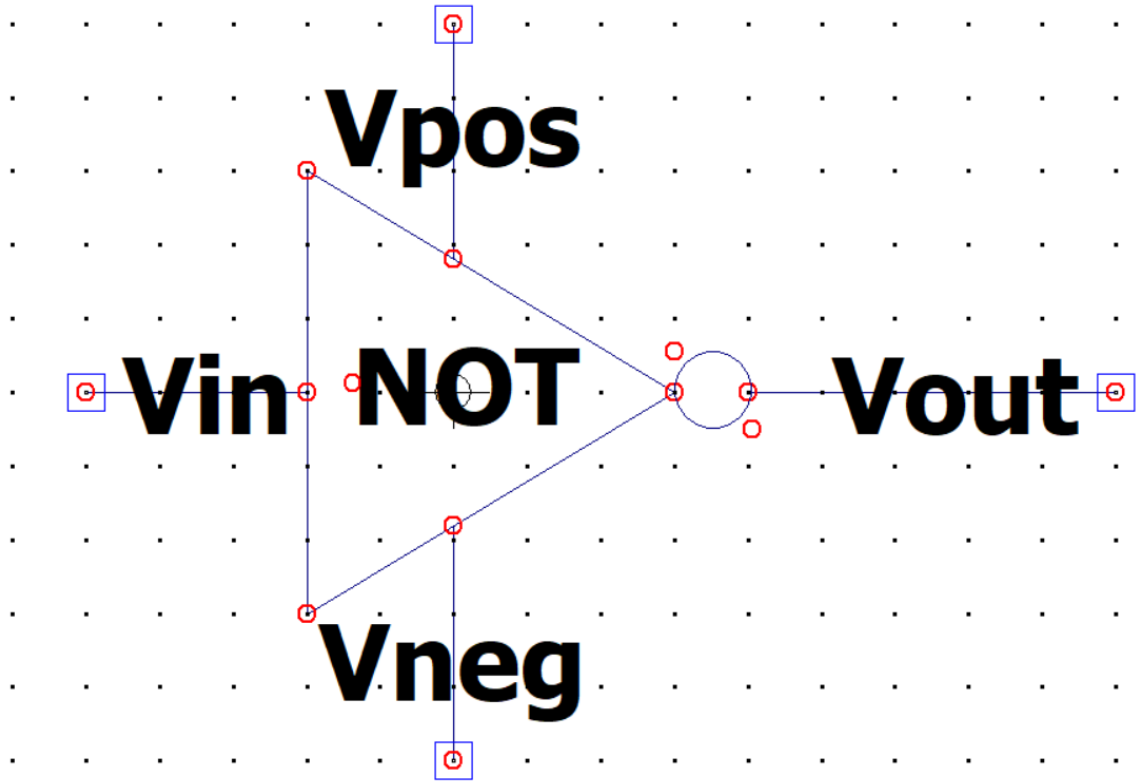
GATES AND SUBCOMPONENTS:

NOT GATE:

Symbol schematic:



Symbol drawing:



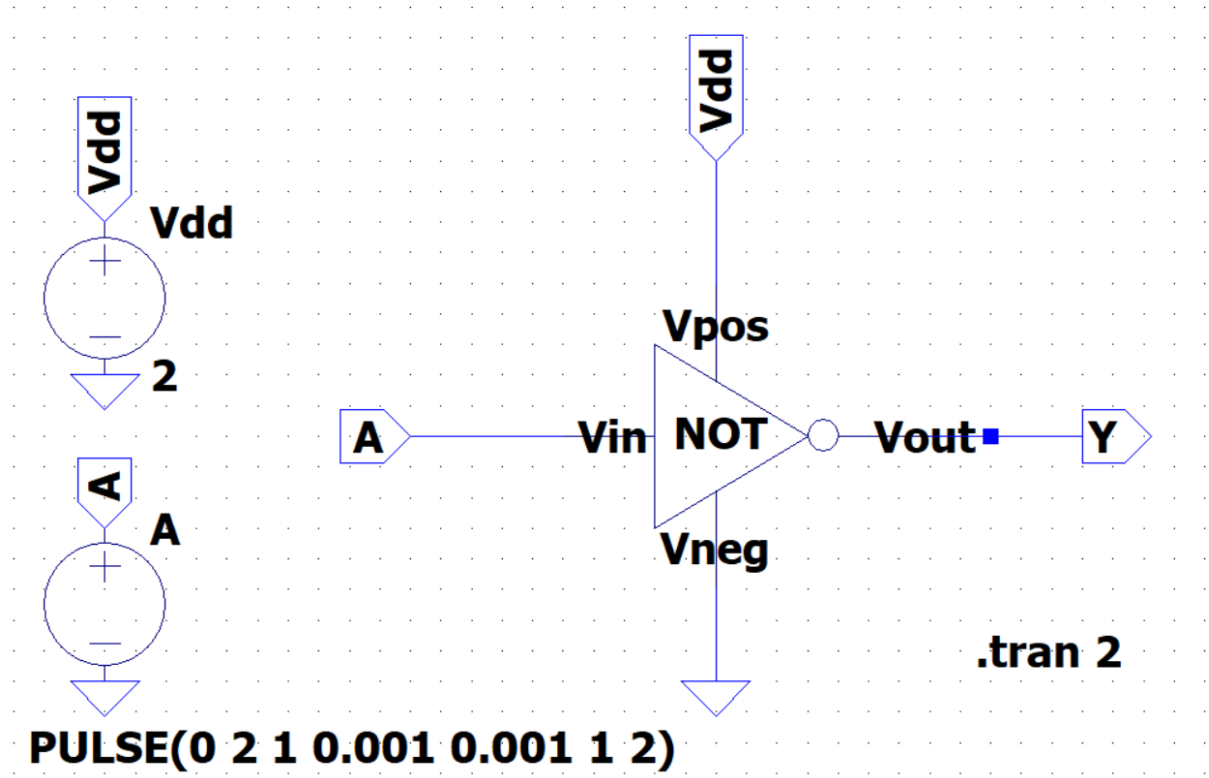
Truth Table:

S.no	A	Output (Y)
0	0	1
1	1	0

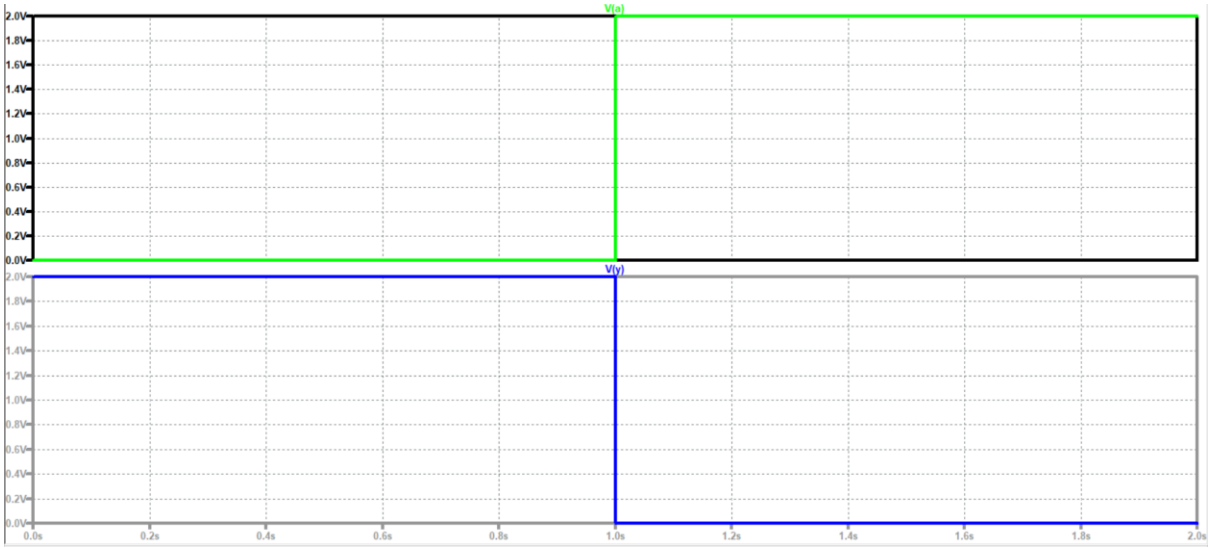
$Y=\pi M(1)$

$Y=\sum m(0)$

Test circuit:



Test circuit waveforms:



Y=πM(1)

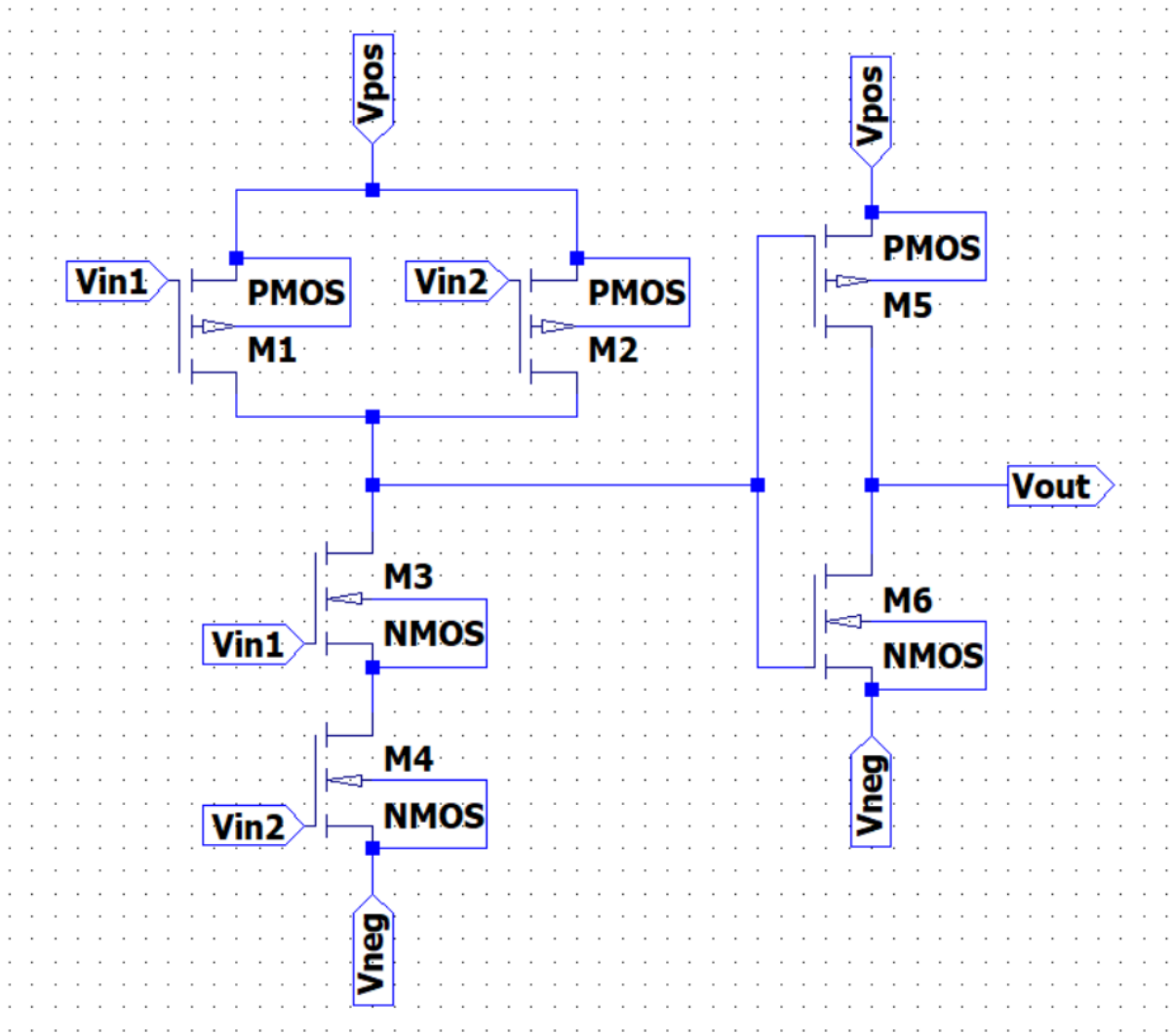
Y=∑m(0)

Verification statement:

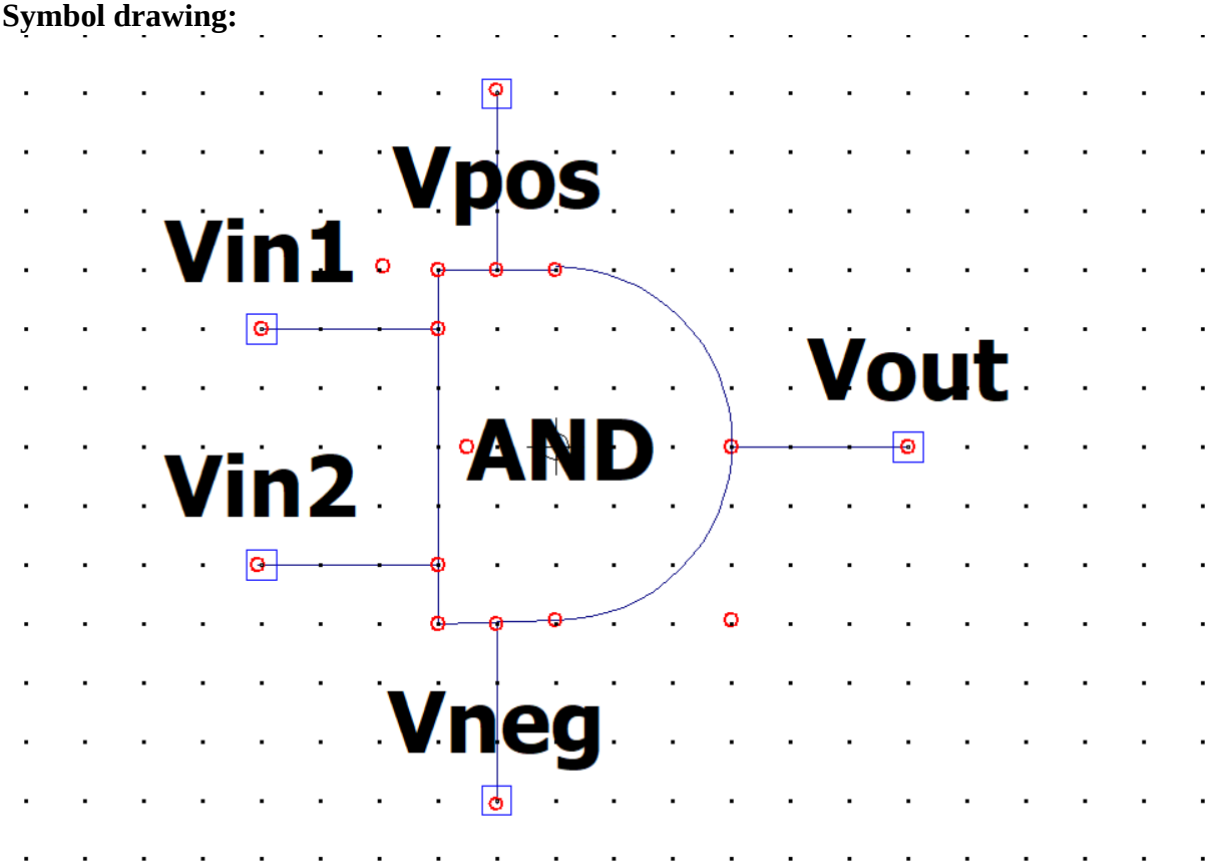
The min terms and max terms of the output from the table is matched with the output waveform, So the designed NOT circuit is verified and working properly for all conditions.

AND GATE:

Symbol schematic:



Symbol drawing:



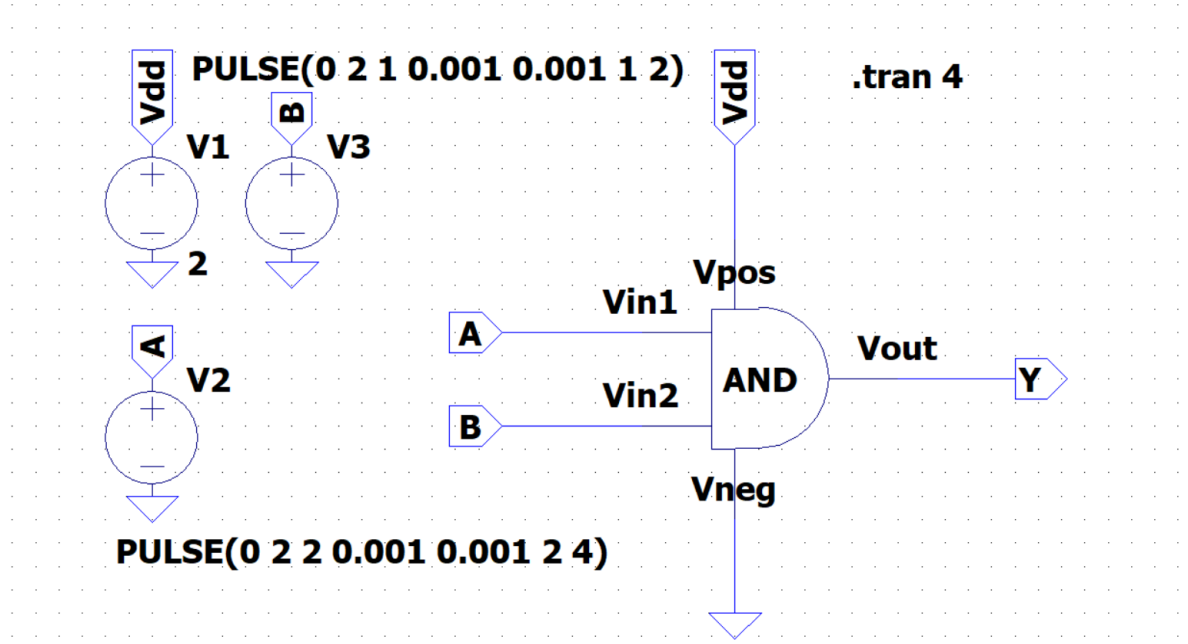
Truth Table:

S.no	A	B	Output (Y)
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

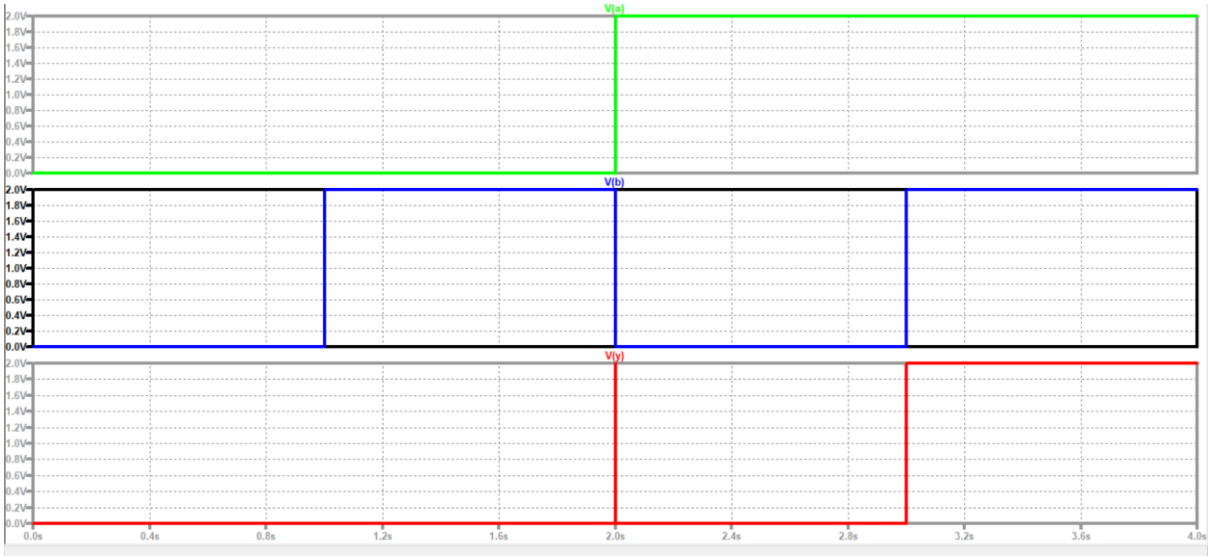
$Y=\pi M(0,1,2)$

$Y=\sum m(3)$

Test circuit:



Test circuit waveforms:



Y=πM(0,1,2)

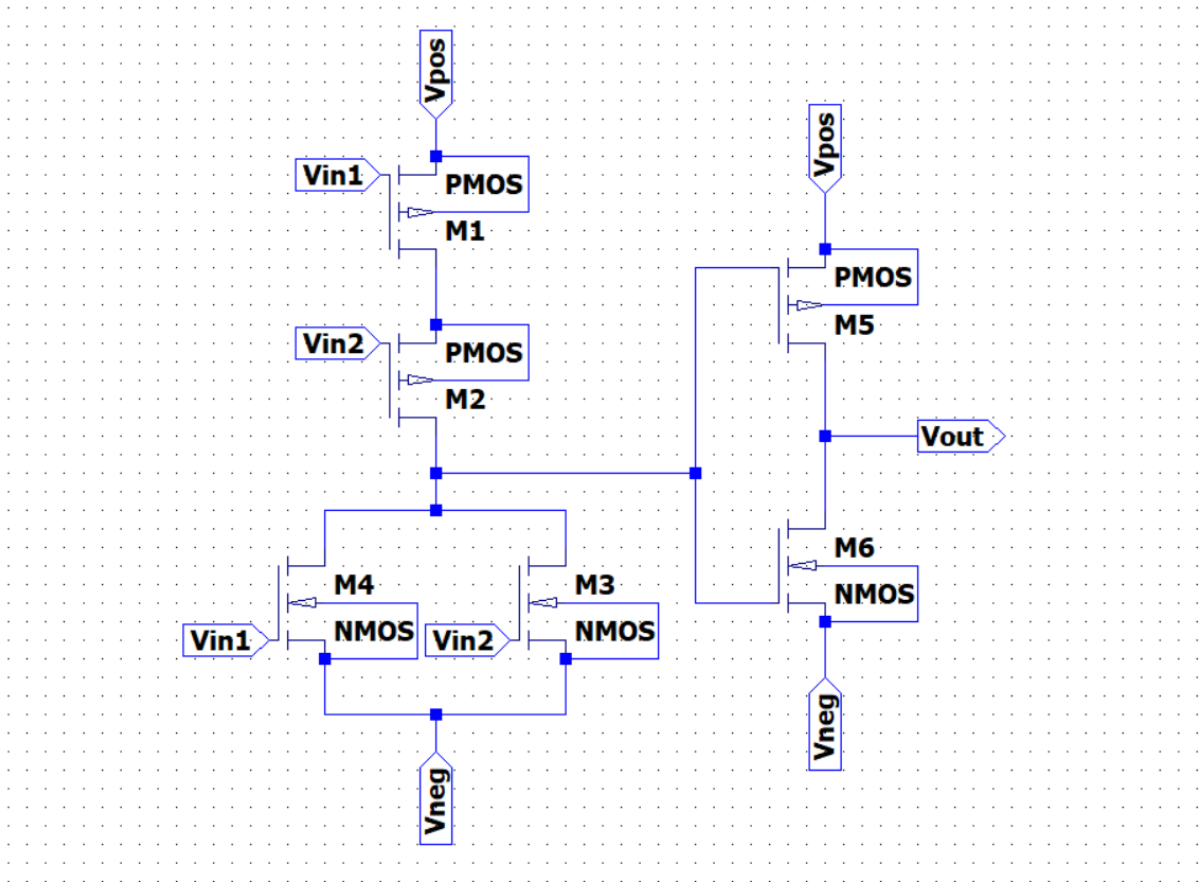
Y=∑m(3)

Verification statement:

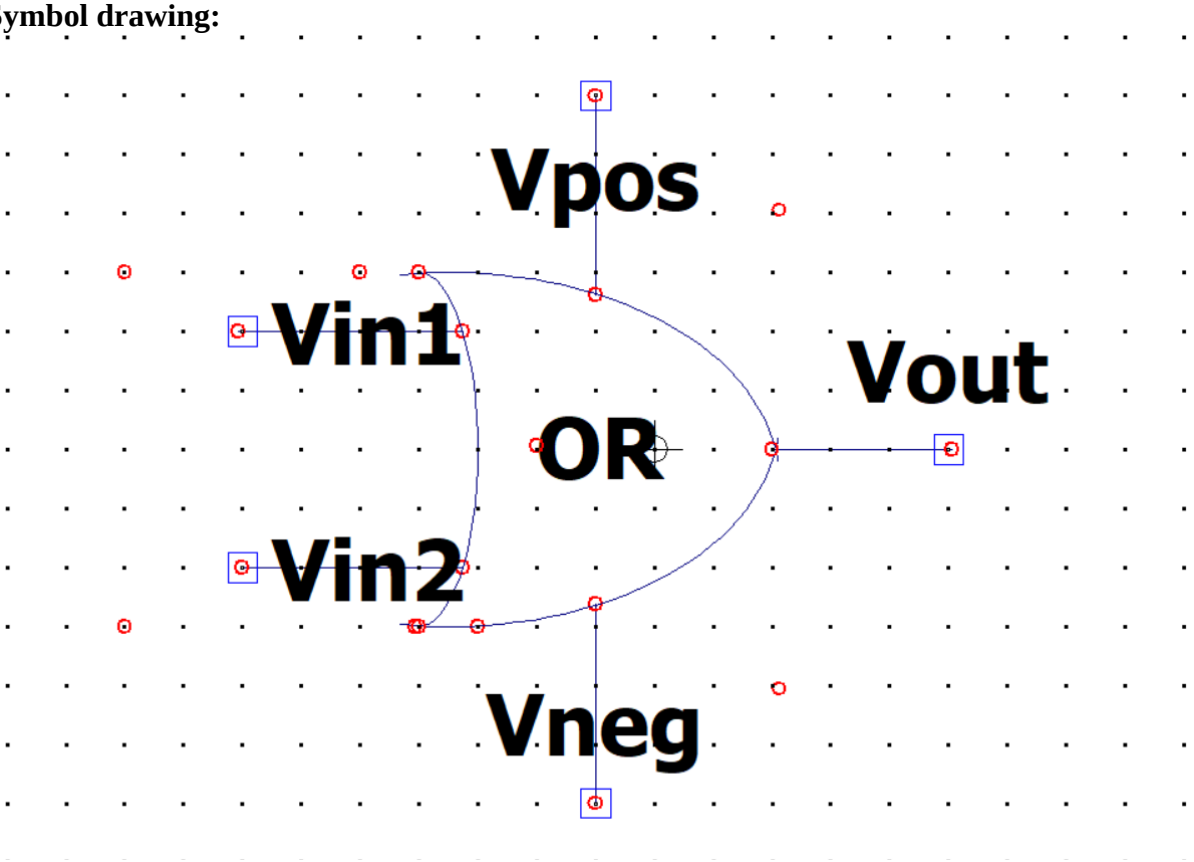
The min terms and max terms of the output from the table is matched with the output waveform, So the designed AND circuit is verified and working properly for all conditions.

OR GATE:

Symbol schematic:



Symbol drawing:



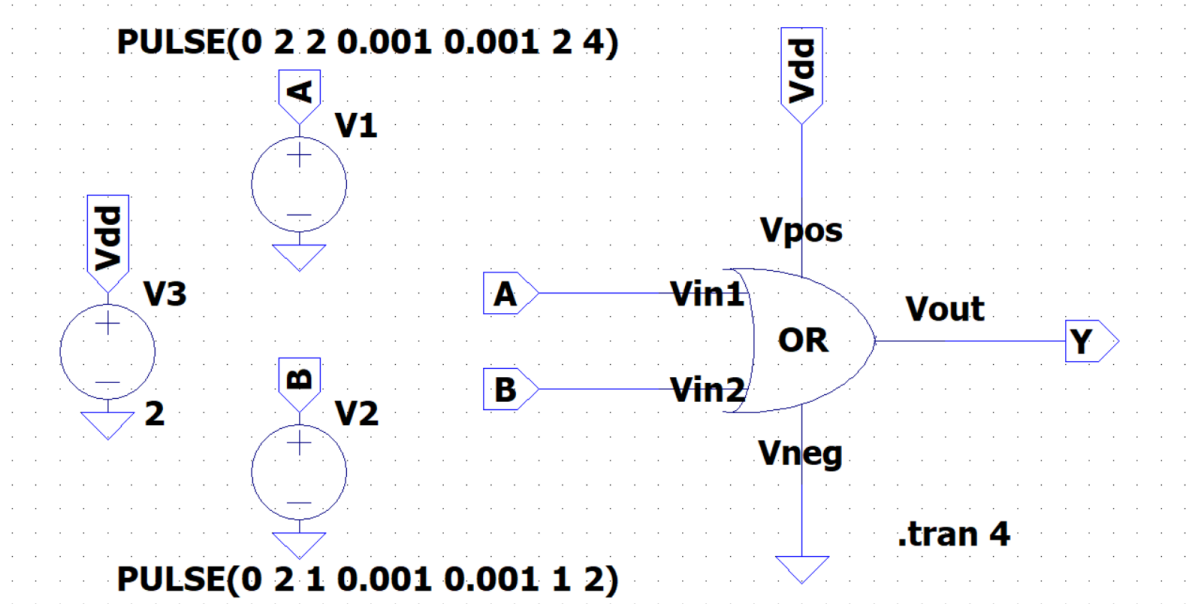
Truth Table:

S.no	A	B	Output (Y)
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

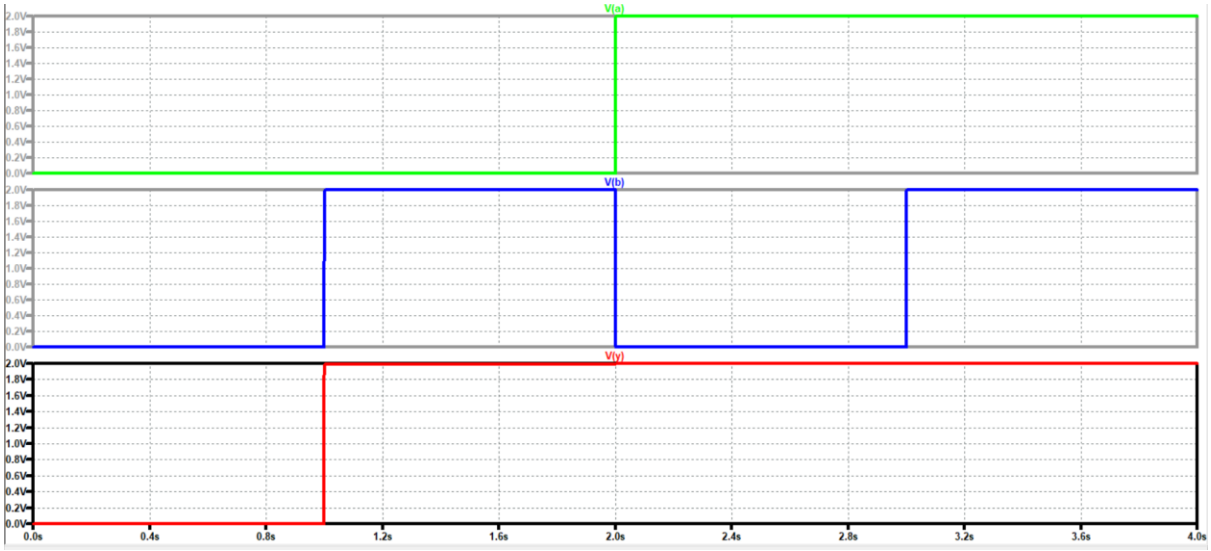
$Y=\pi M(0)$

$Y=\sum m(1,2,3)$

Test circuit:



Test circuit waveforms:



Y=πM(0)

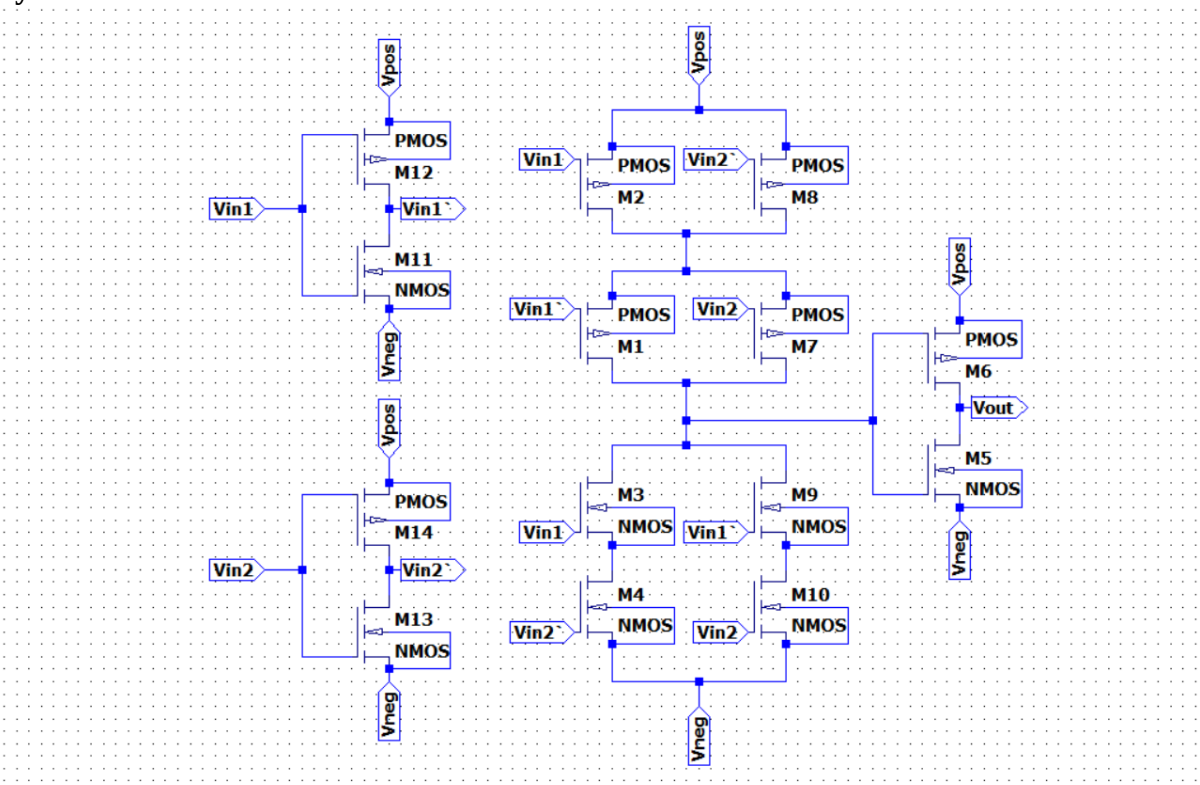
Y=∑m(1,2,3)

Verification statement:

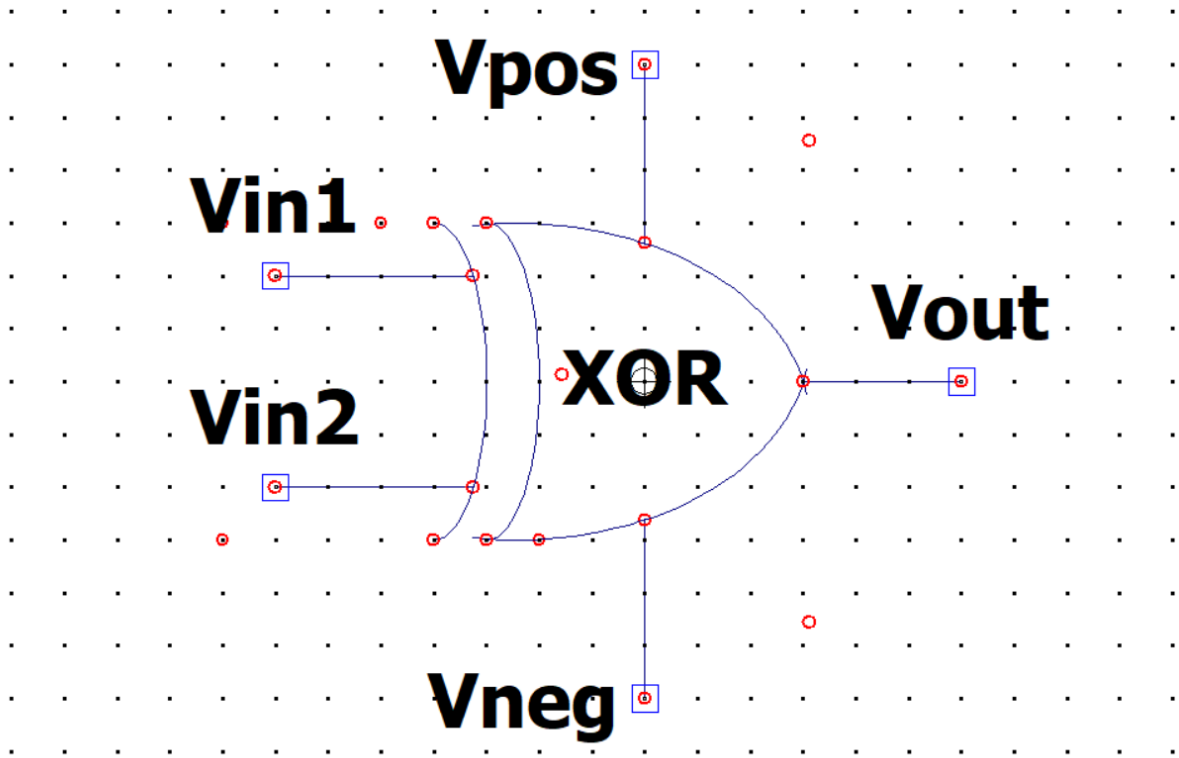
The min terms and max terms of the output from the table is matched with the output waveform, So the designed OR circuit is verified and working properly for all conditions.

XOR GATE:

Symbol schematic:



Symbol drawing:

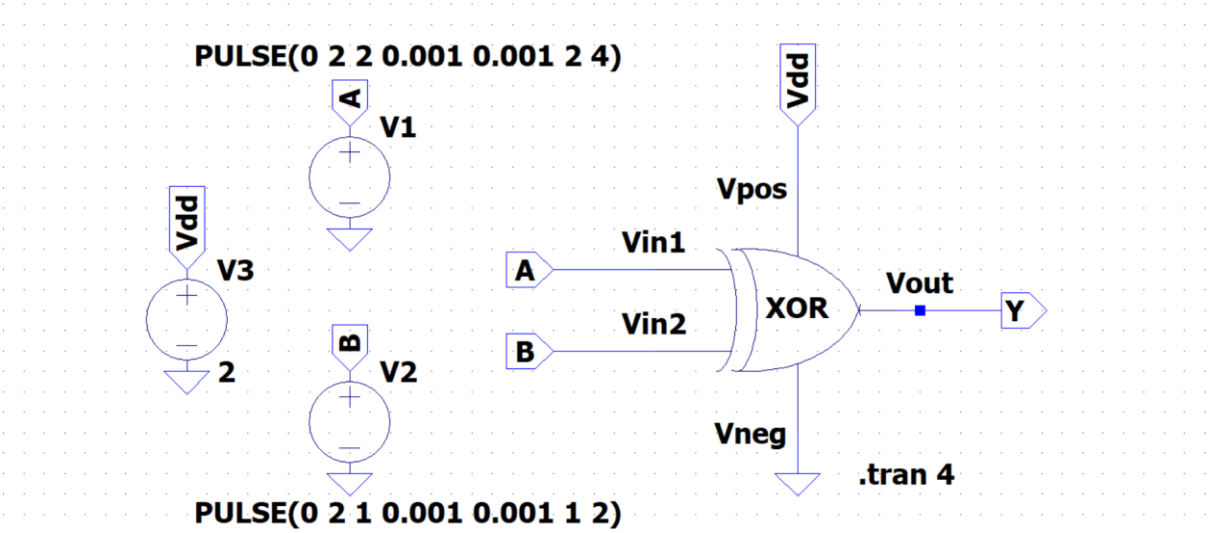


Truth Table:

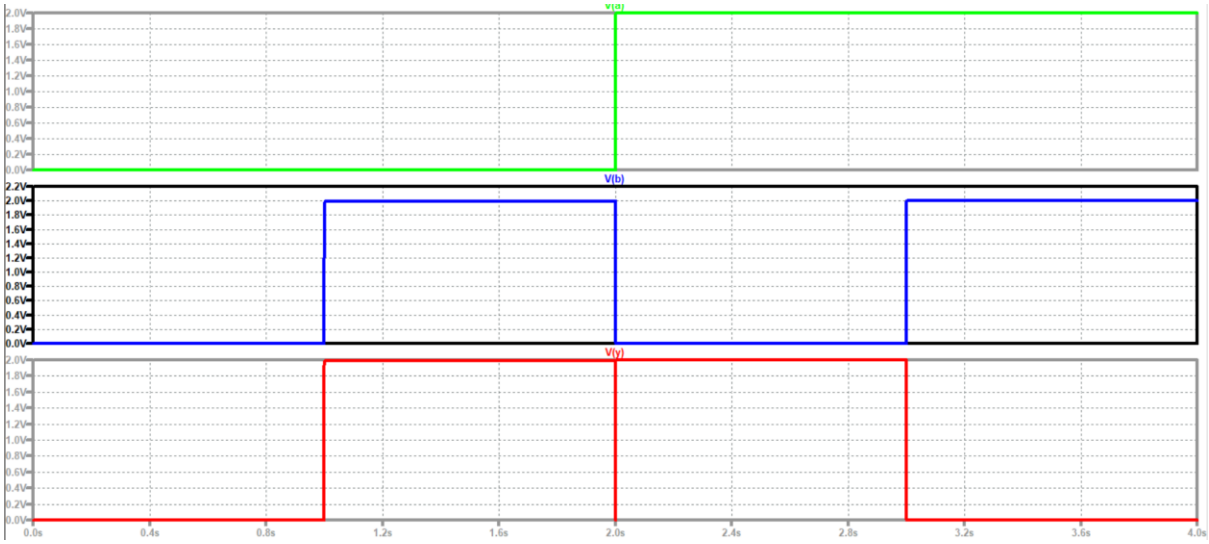
S.no	A	B	Output (Y)
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

Y=πM(0,3); Y=∑m(1,2)

Test circuit:



Test circuit waveforms:



Y=πM(0,3)

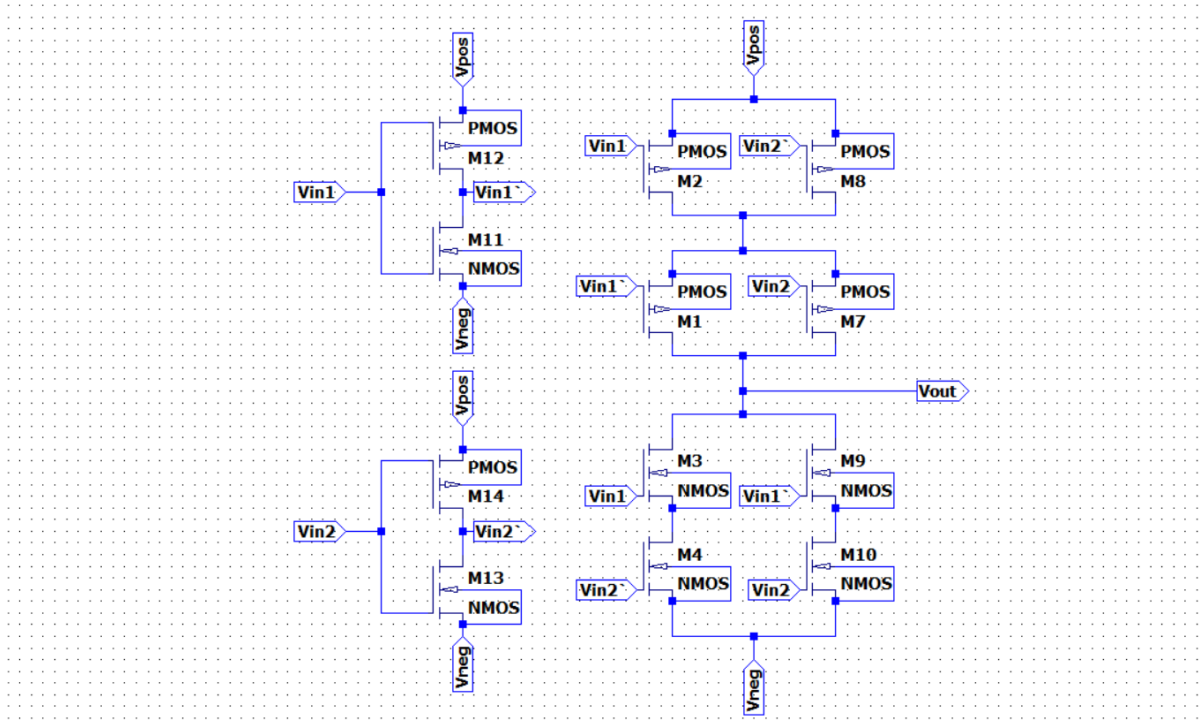
Y=∑m(1,2)

Verification statement:

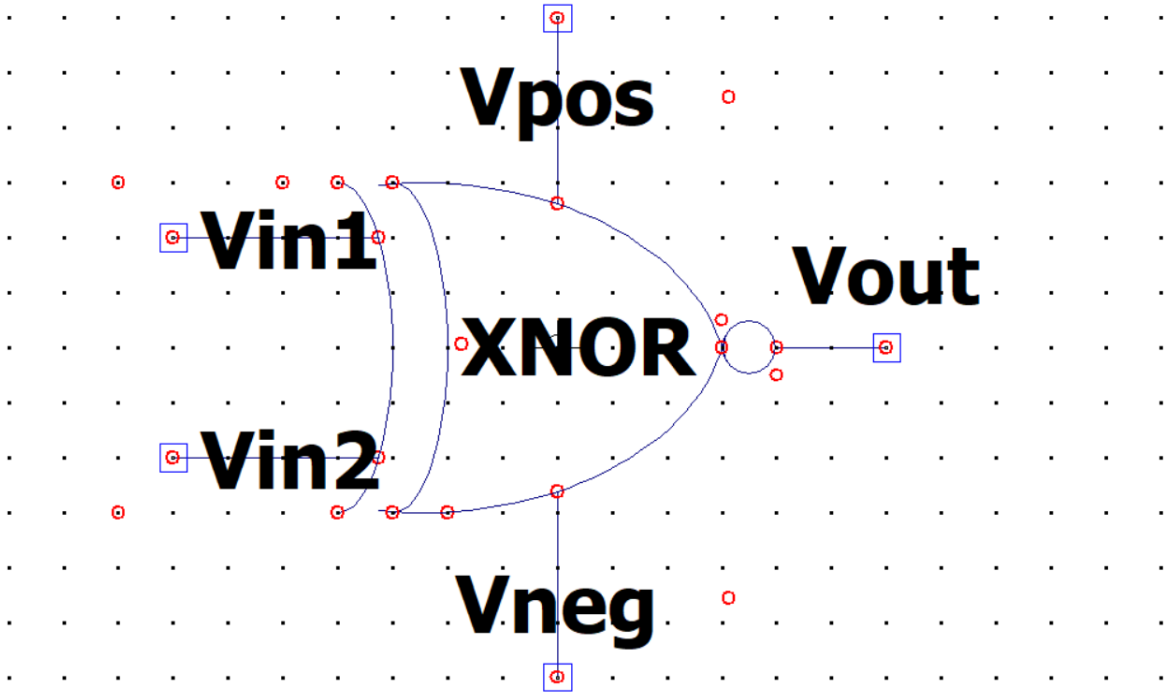
The min terms and max terms of the output from the table is matched with the output waveform, So the designed XOR circuit is verified and working properly for all conditions.

XNOR GATE:

Symbol schematic:



Symbol drawing:

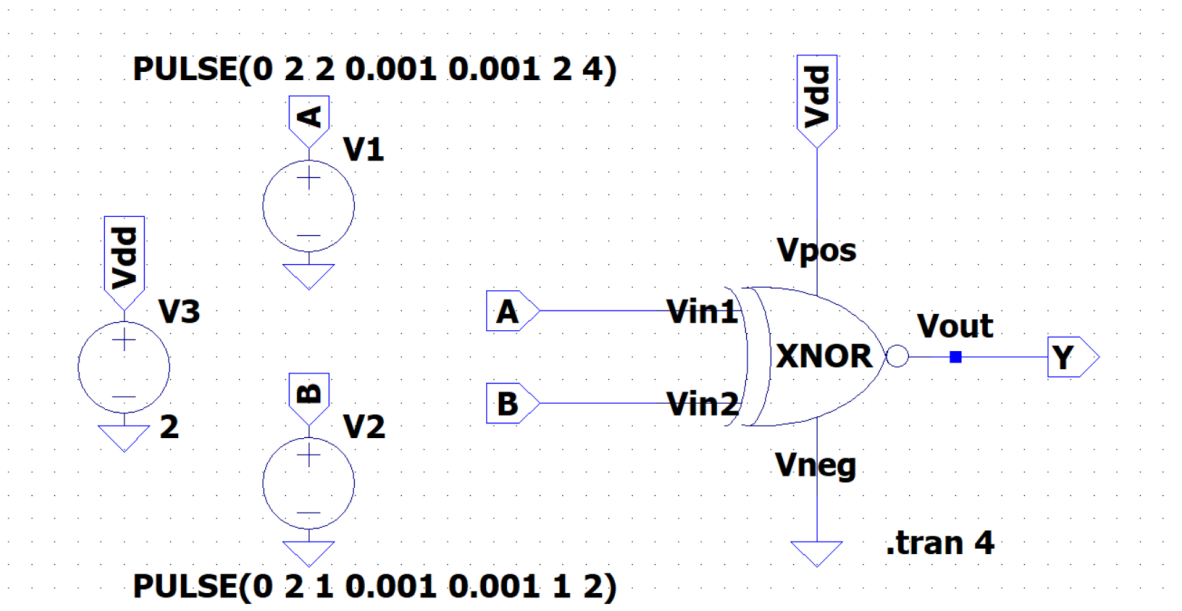


Truth Table:

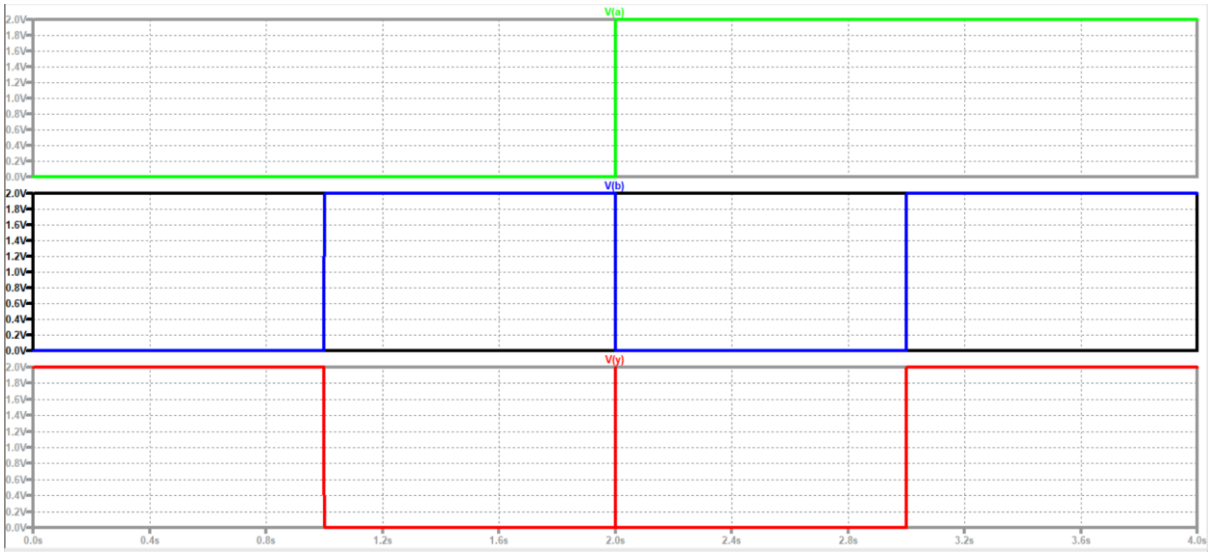
S.no	A	B	Output (Y)
1	0	0	1
2	0	1	0
3	1	0	0
4	1	1	1

Y=πM(1,2); Y=∑m(0,3)

Test circuit:



Test circuit waveforms:



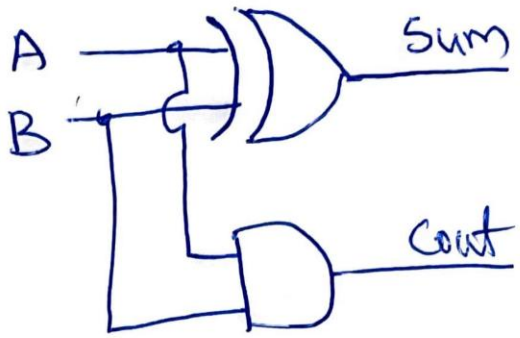
Y=πM(1,2); Y=∑m(0,3)

Verification statement:

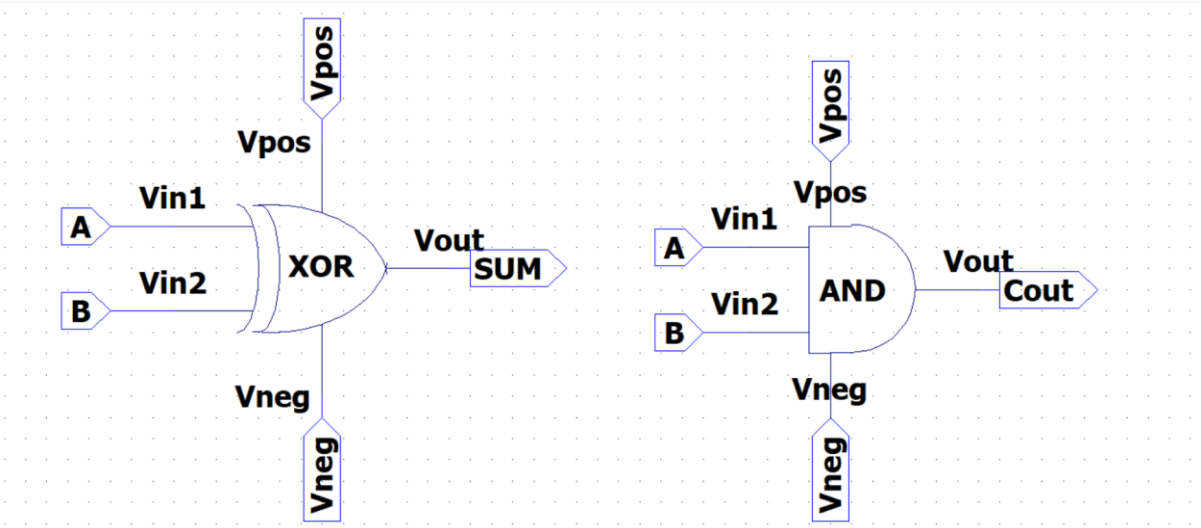
The min terms and max terms of the output from the table is matched with the output waveform, So the designed XNOR circuit is verified and working properly for all conditions.

HALF ADDER:

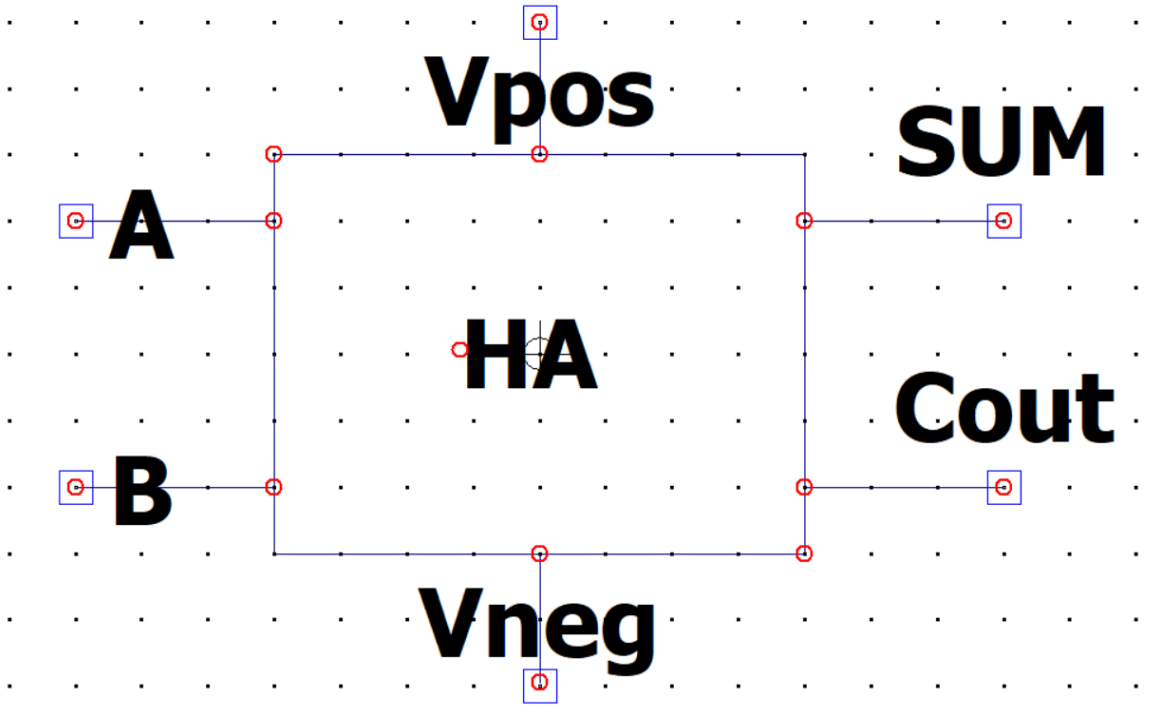
Gate-level drawing:



Symbol schematic:



Symbol drawing:



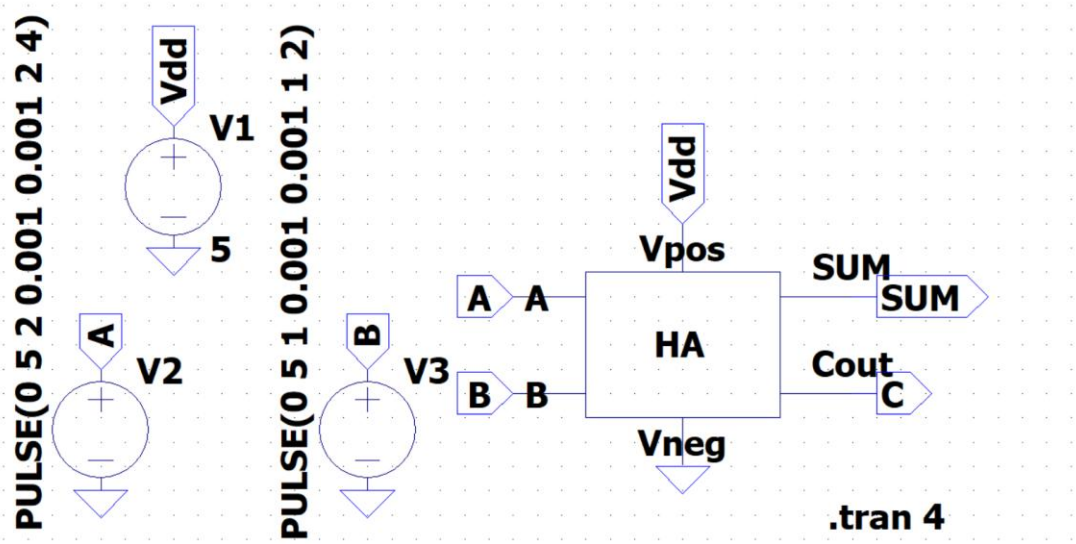
Truth table:

S.no	A	B	SUM	Cout
0	0	0	0	0
1	0	1	1	0
2	1	0	1	0
3	1	1	0	1

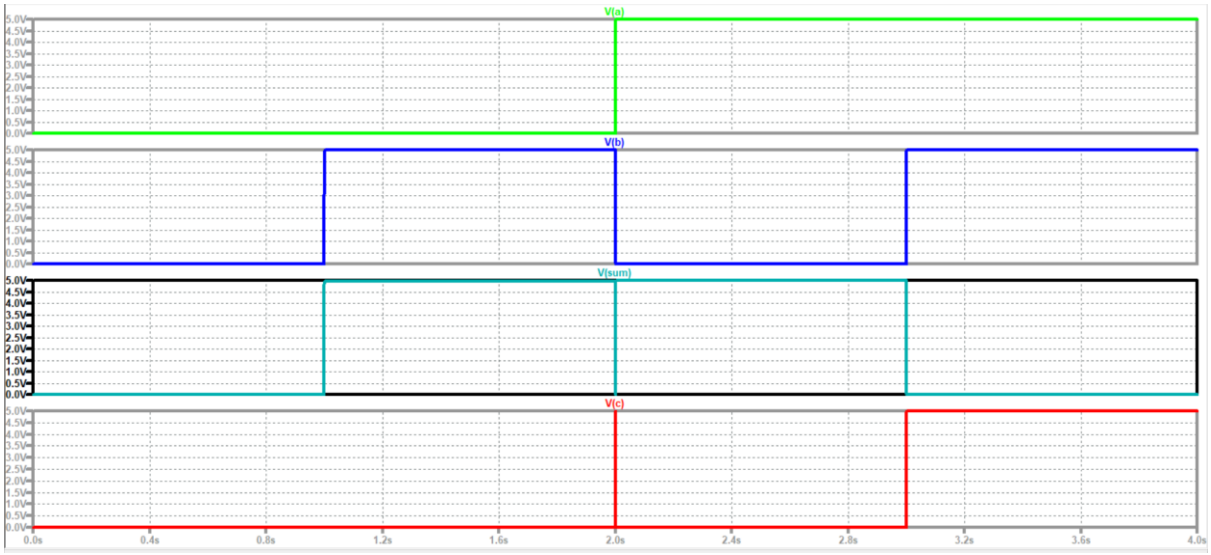
SUM=πM(0,3), Cout= πM(0,1,2),

SUM=∑m(1,2), Cout=∑m(3)

Test circuit:



Test circuit waveforms:



SUM=πM(0,3), Cout= πM(0,1,2),

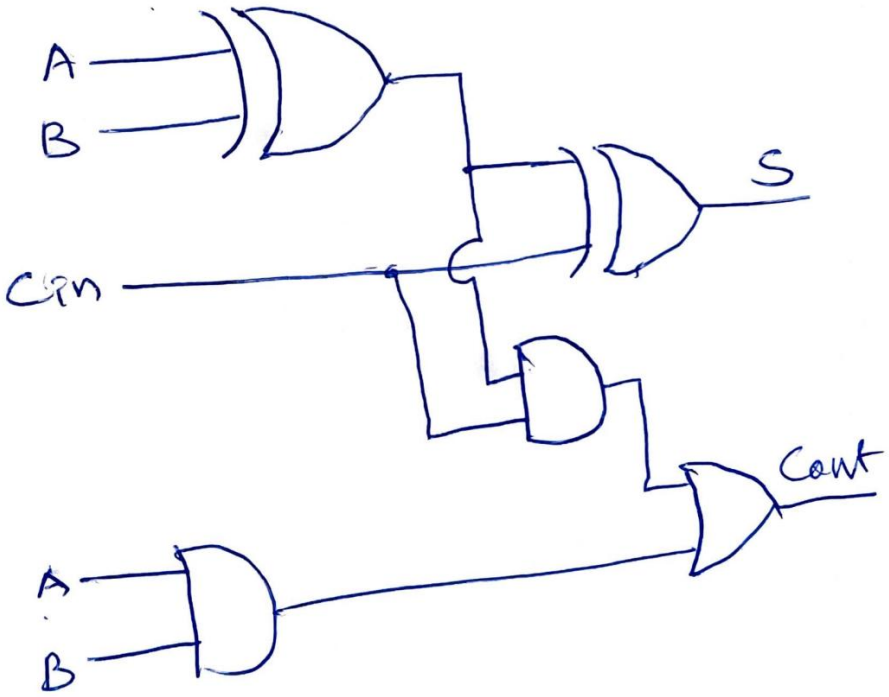
SUM=∑m(1,2), Cout=∑m(3)

Verification statement:

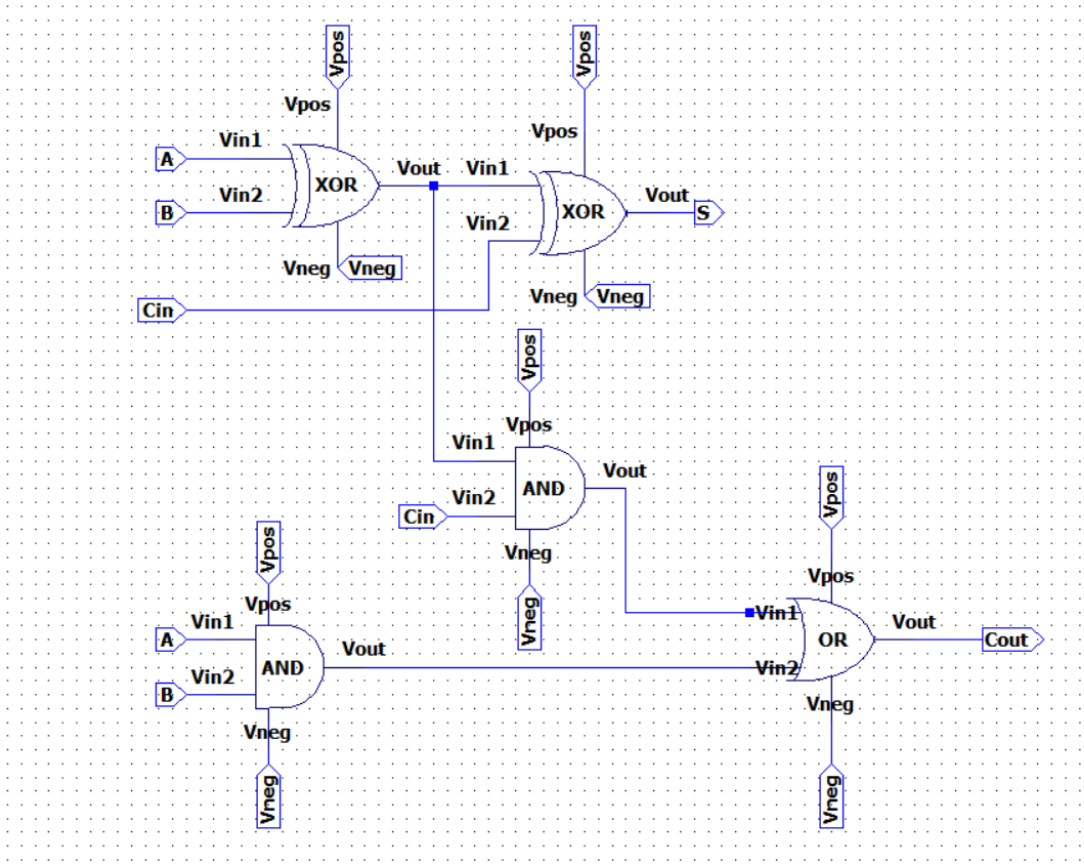
The min terms and max terms of the output from the table is matched with the output waveform, So the designed HALF ADDER circuit is verified and working properly for all conditions.

FULL ADDER:

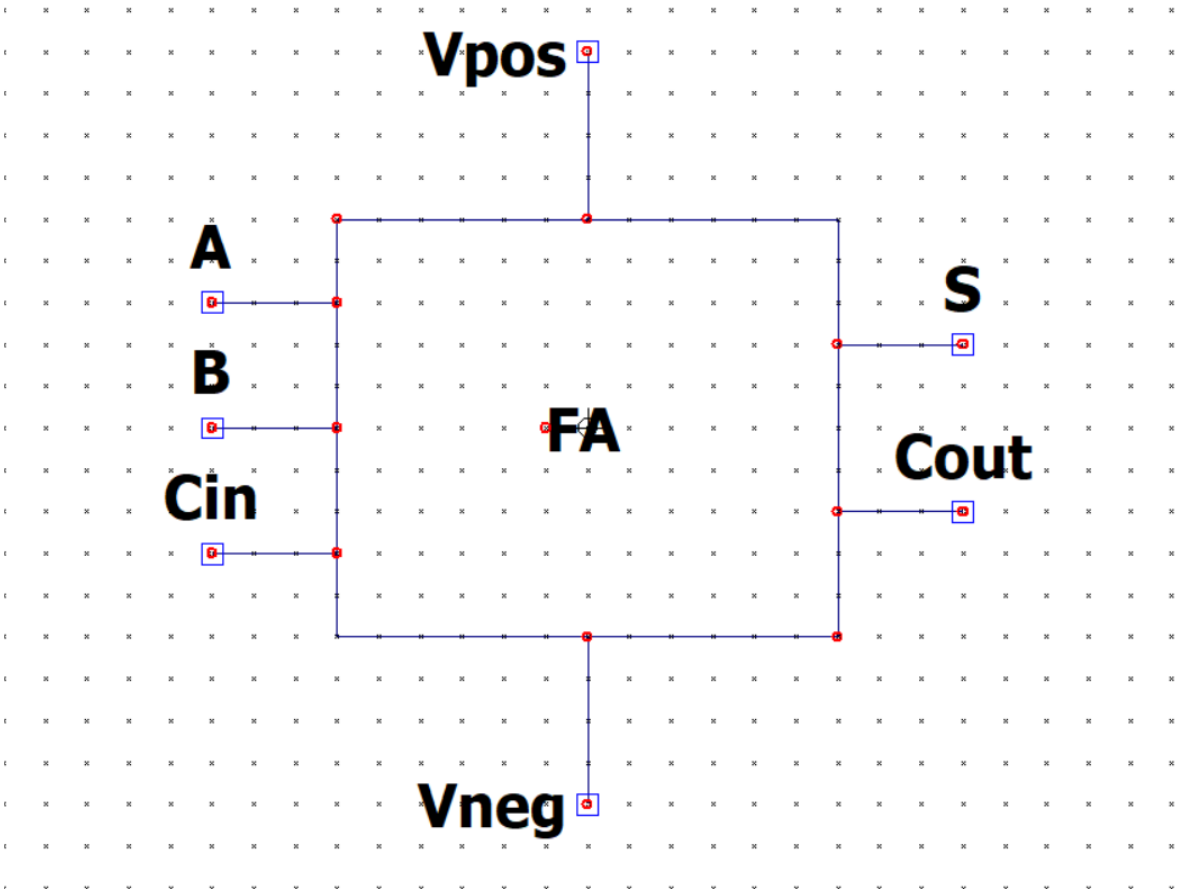
Gate-level drawing:



Symbol schematic:



Symbol drawing:



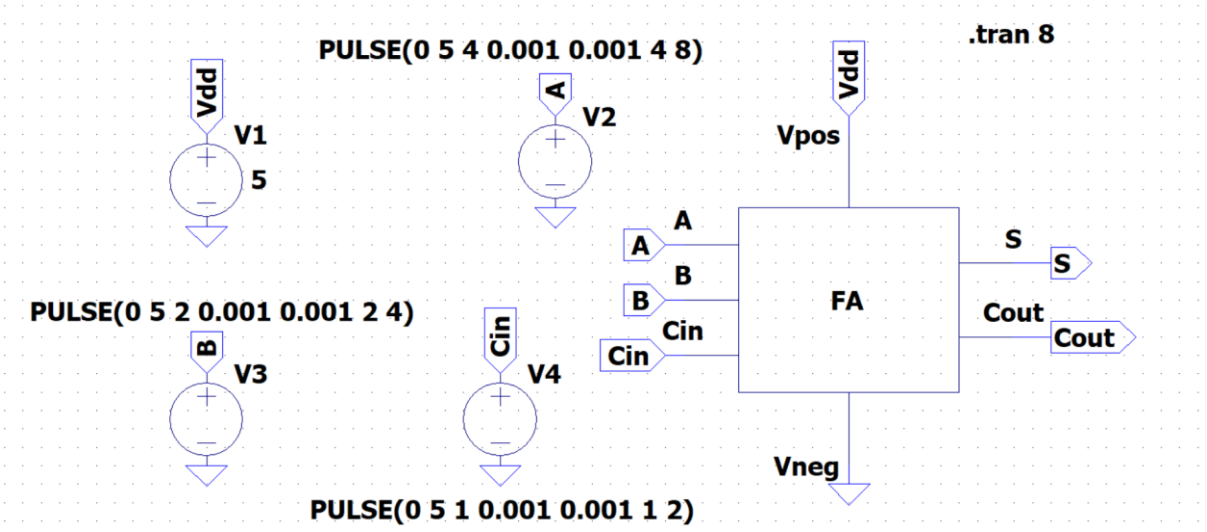
Truth table:

S.No.:	A	B	Cin	SUM	Cout
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

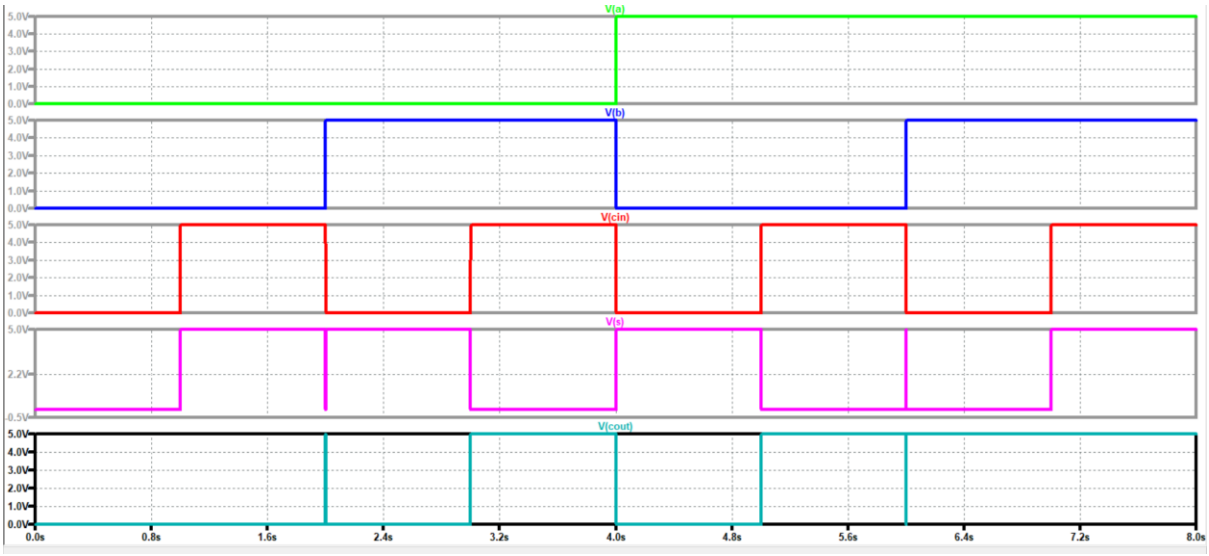
SUM=πM(0,3,5,6), Cout= πM(0,1,2,4),

SUM=∑m(1,2,4,7), Cout=∑m(3,5,6,7)

Test circuit:



Test circuit waveforms:



SUM=πM(0,3,5,6), Cout= πM(0,1,2,4),

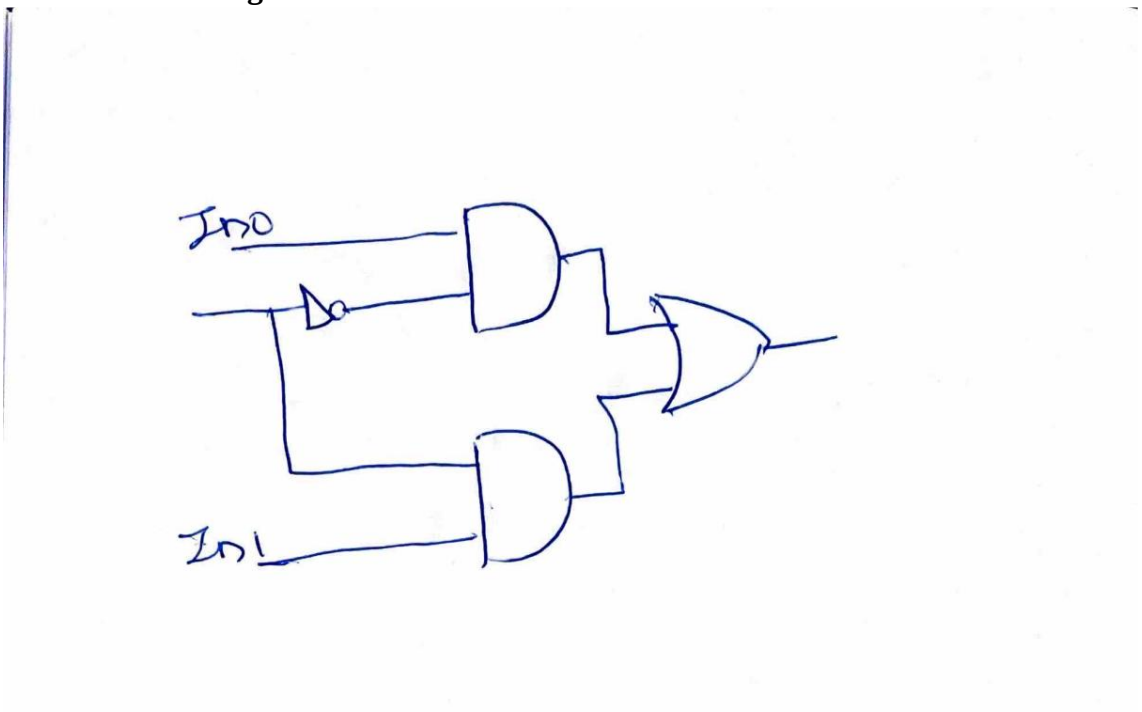
SUM=∑m(1,24,7), Cout=∑m(3,5,6,7)

Verification statement:

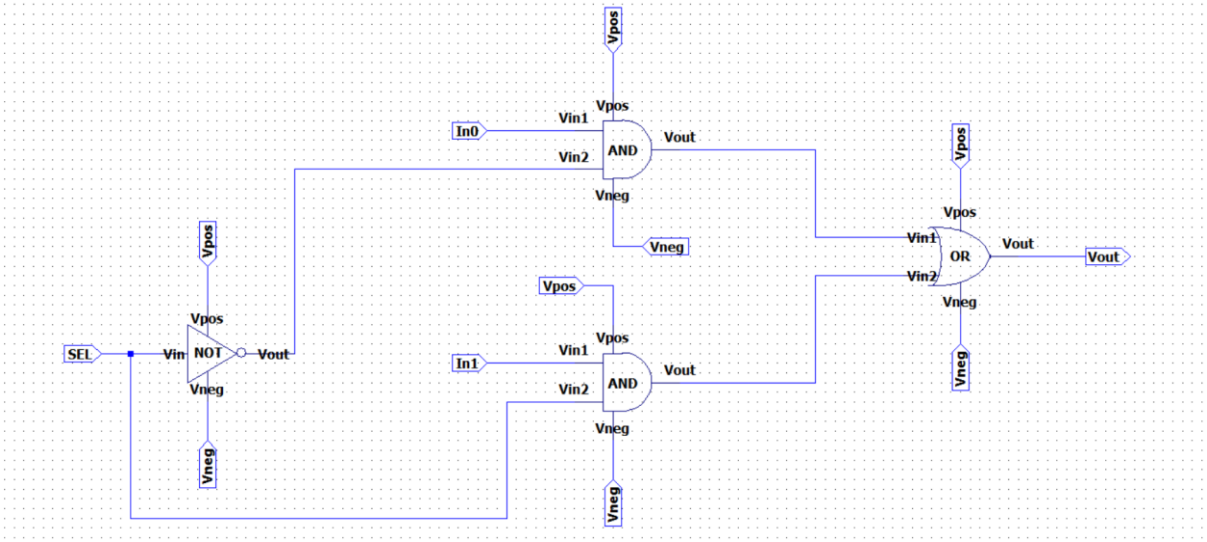
The min terms and max terms of the output from the table is matched with the output waveform, So the designed FULL ADDER circuit is verified and working properly for all conditions.

MUX 2:1:

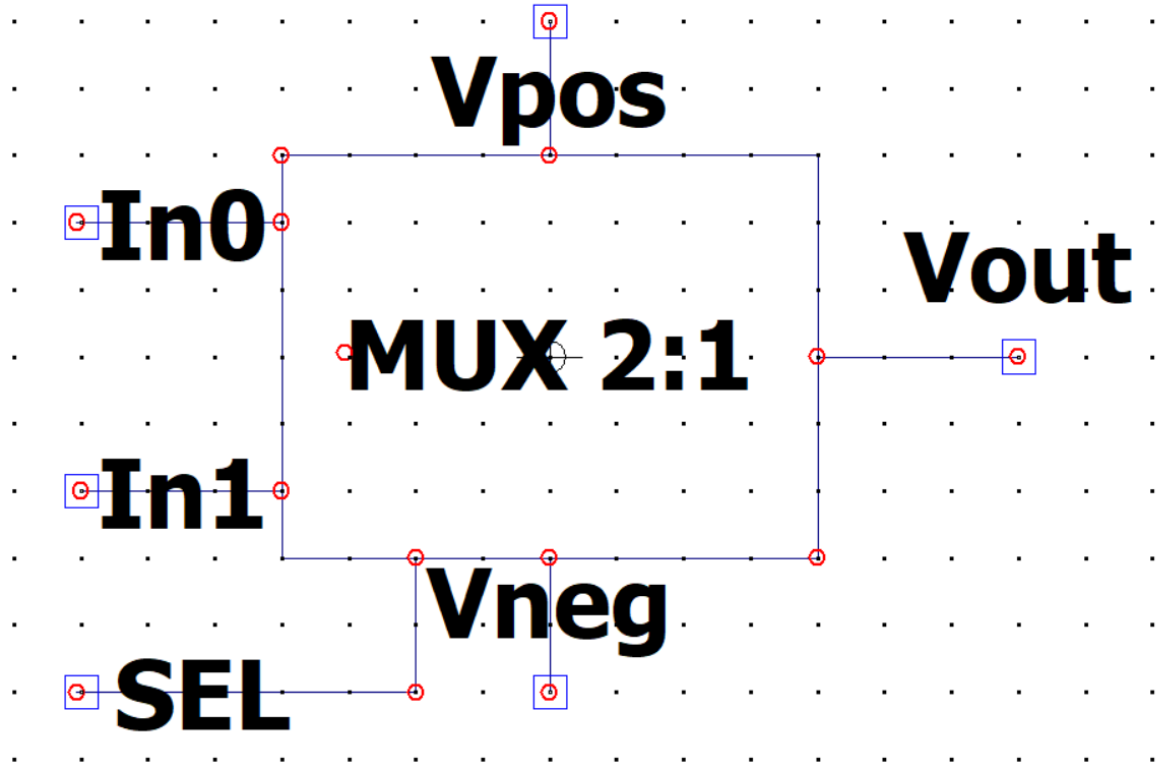
Gate-level drawing:



Symbol schematic:



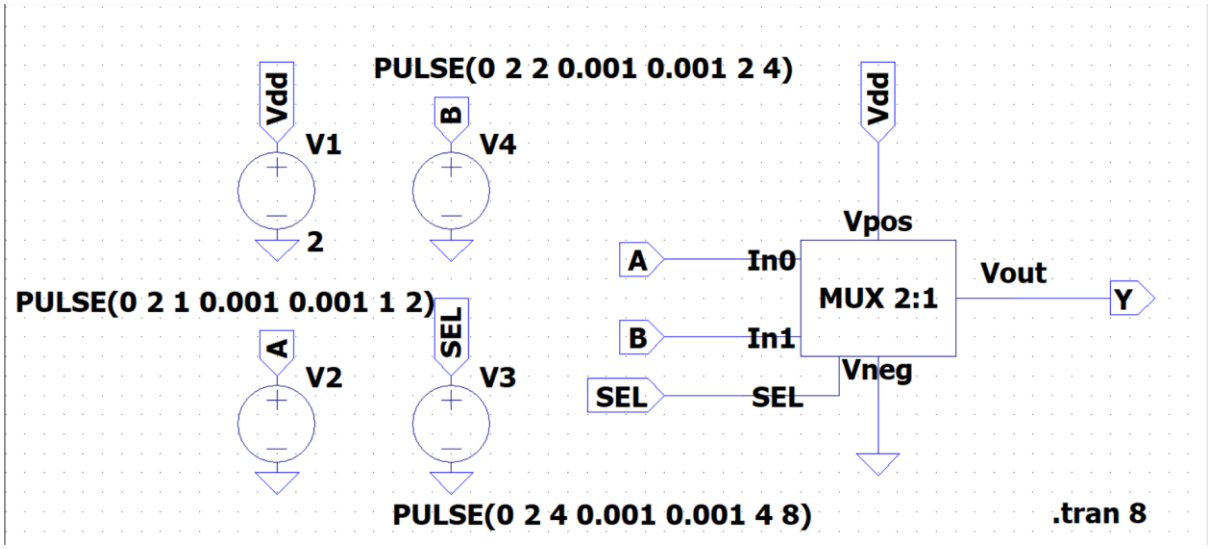
Symbol drawing:



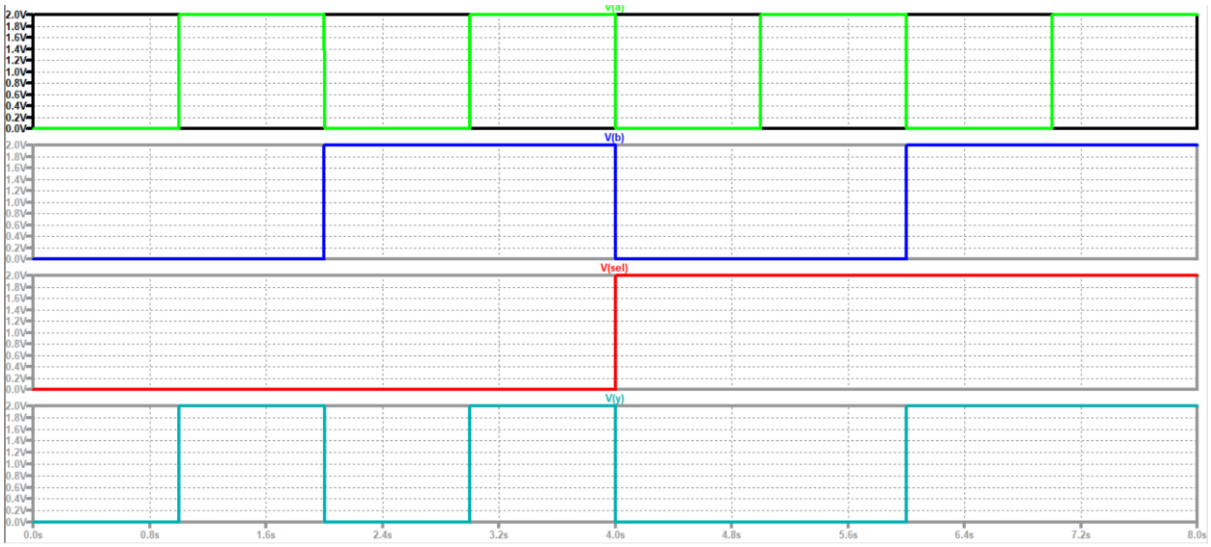
Truth table:

S.no	Data Inputs		SEL	Vout
	In0	In1		
0	1	0	0	1(In0)
1	0	1	1	1(In1)

Test circuit:



Test circuit waveforms:

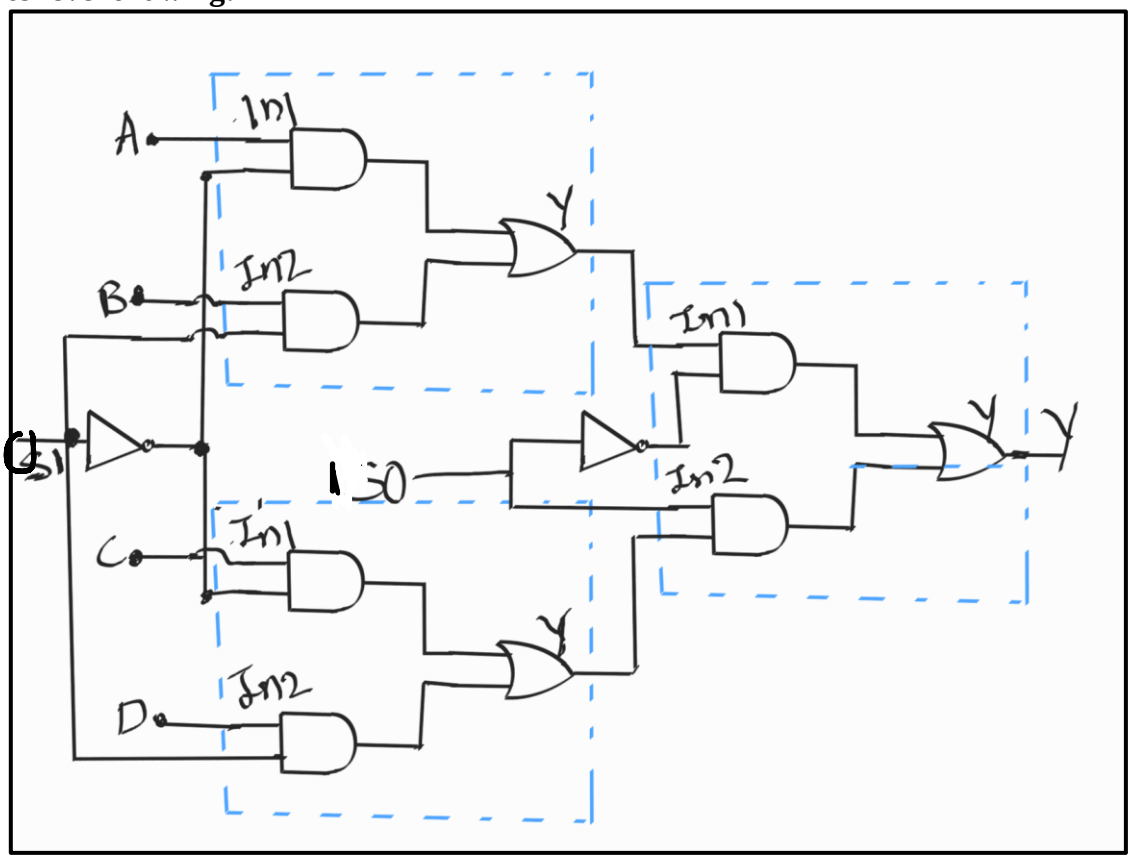


Verification statement:

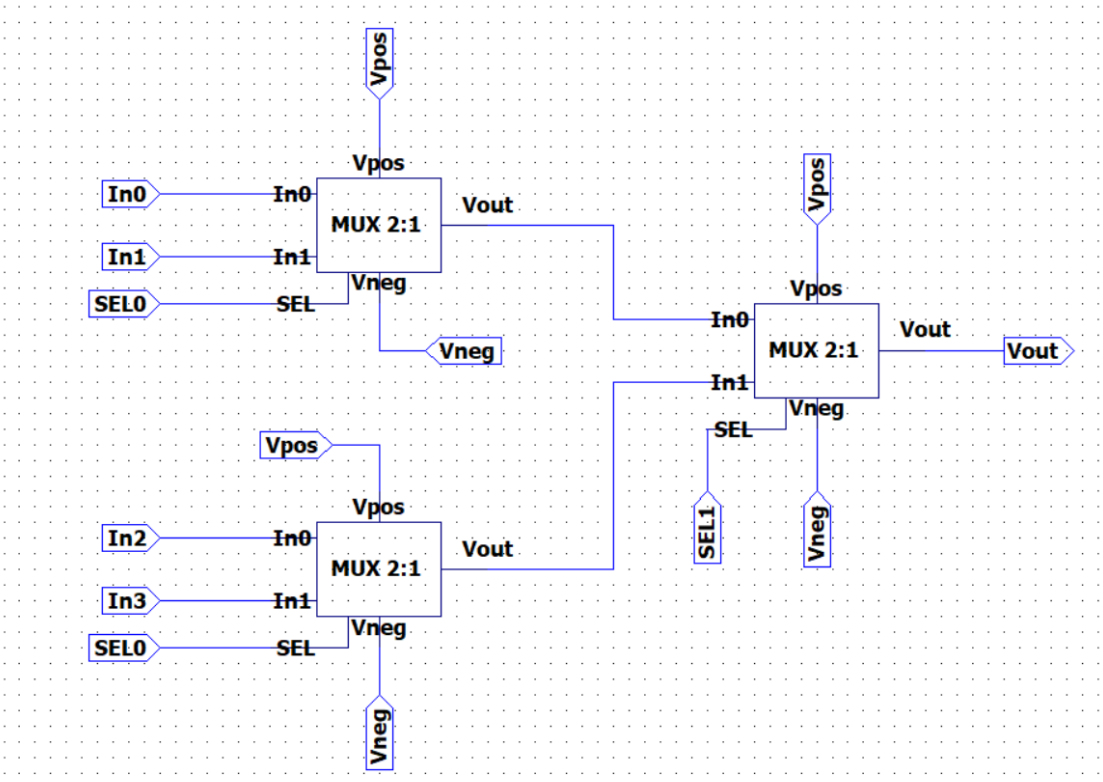
The 2:1 MUX is designed and verified with output waveforms to the truth table outputs data. Hence, the 2:1 MUX is working properly for the given data inputs.

MUX 4:1:

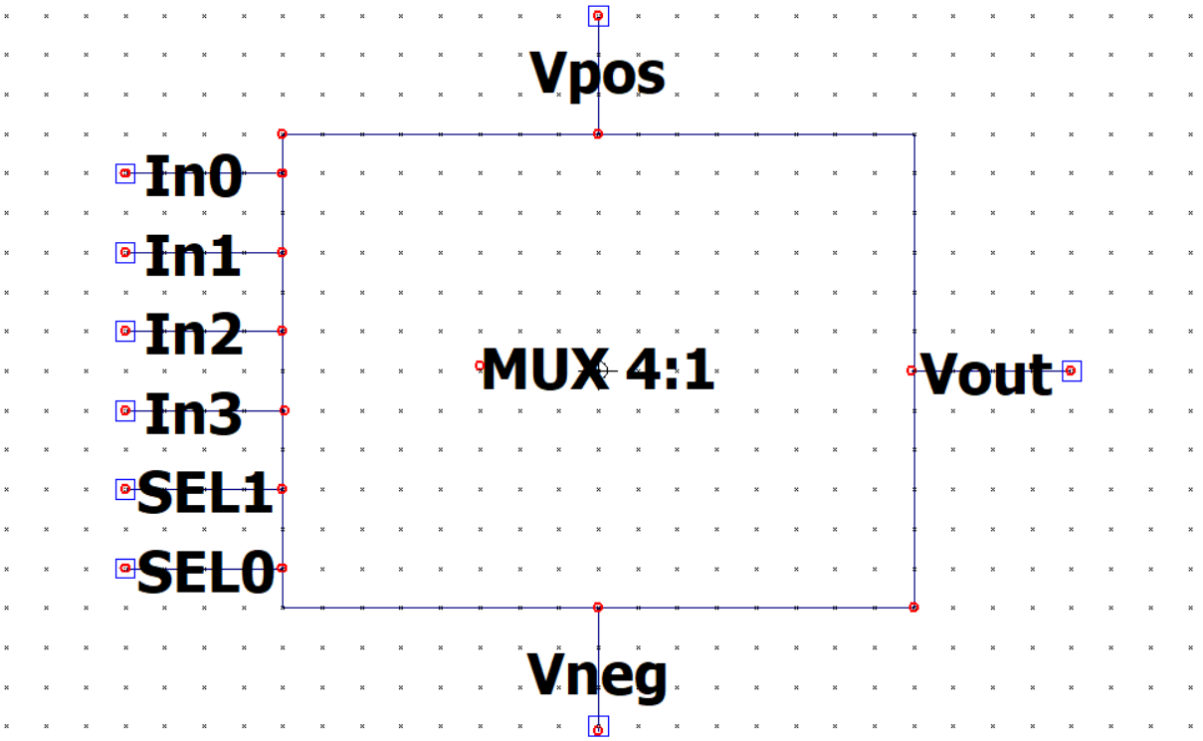
Gate-level drawing:



Symbol schematic:



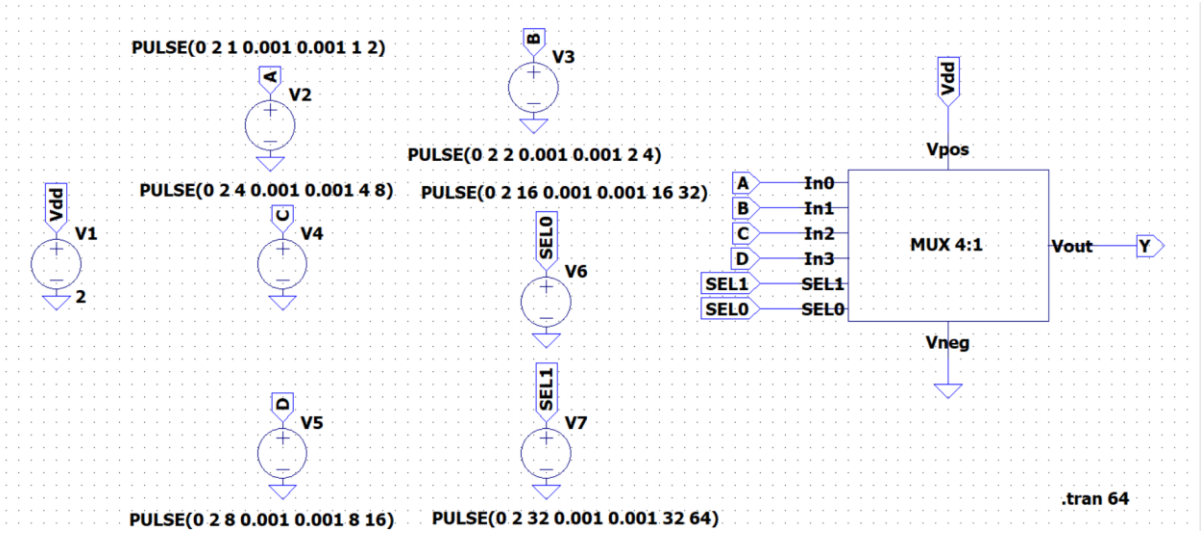
Symbol drawing:



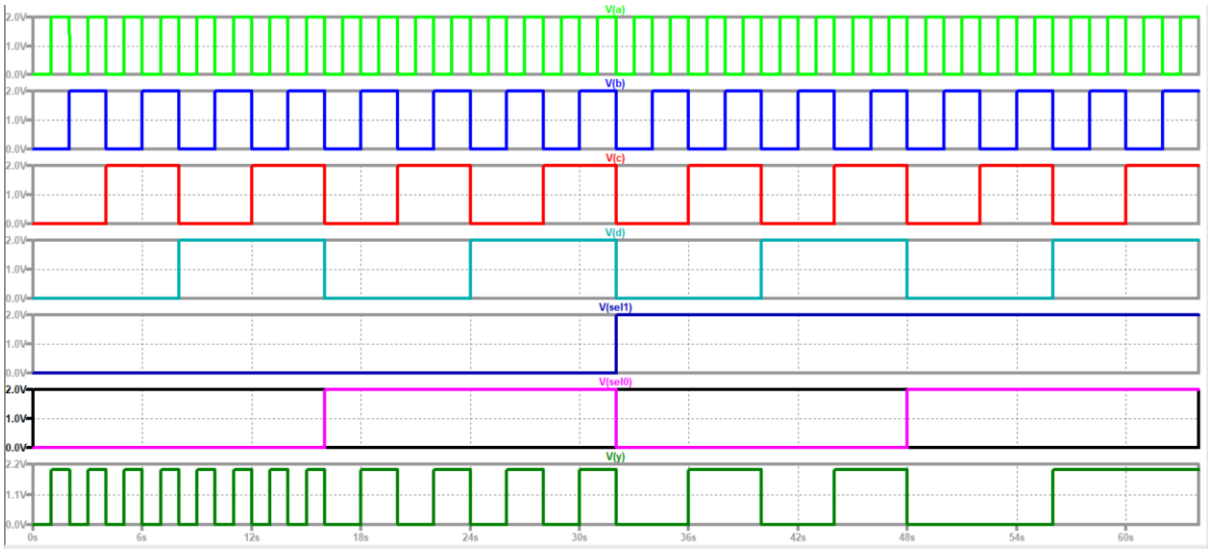
Truth table:

S.no	Data Inputs				SEL1	SEL0	Output (Vout)
	In0	In1	In2	In3			
0	1	0	0	0	0	0	1 (In0)
1	0	1	0	0	0	1	1(In1)
2	0	0	1	0	1	0	1(In2)
3	0	0	0	1	1	1	1(In3)

Test circuit:



Test circuit waveforms:

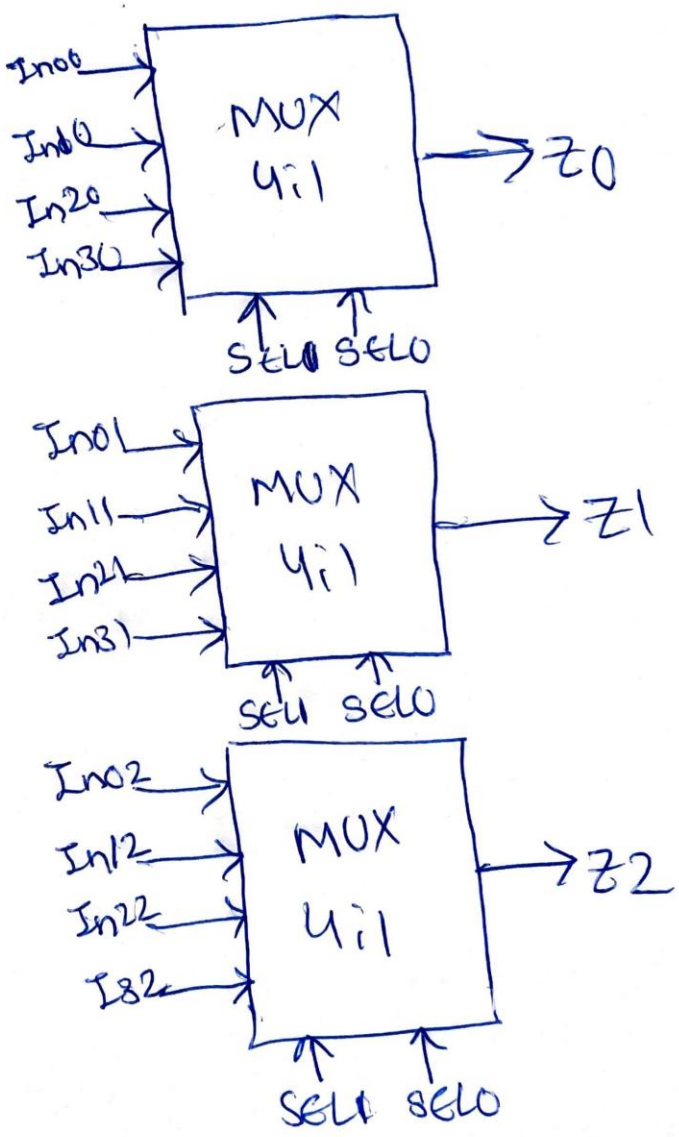


Verification statement:

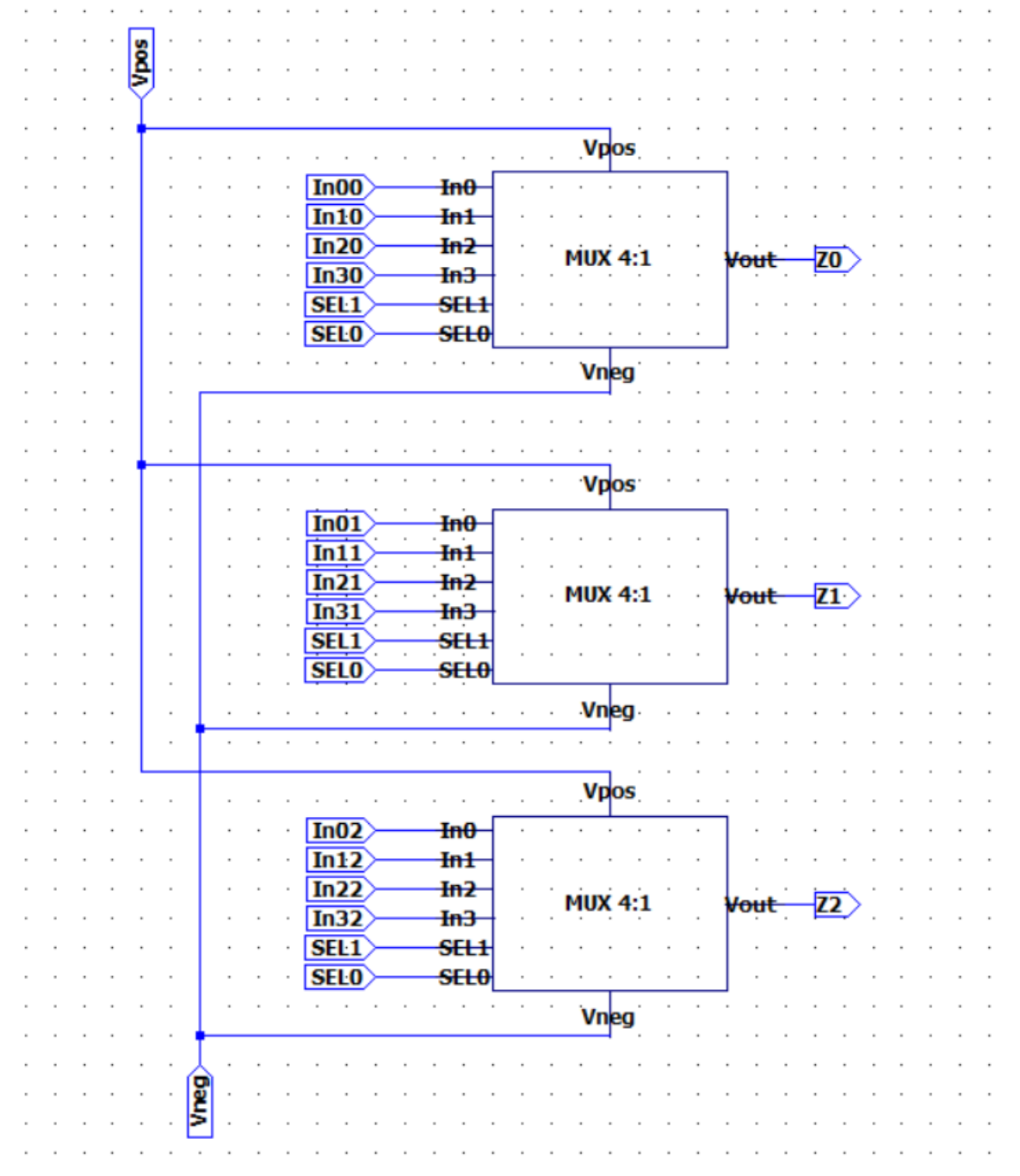
The 4:1 MUX is designed and verified with output waveforms to the truth table outputs data. Hence, the 4:1 MUX is working properly for the given data inputs.

3-BIT MUX 4:1:

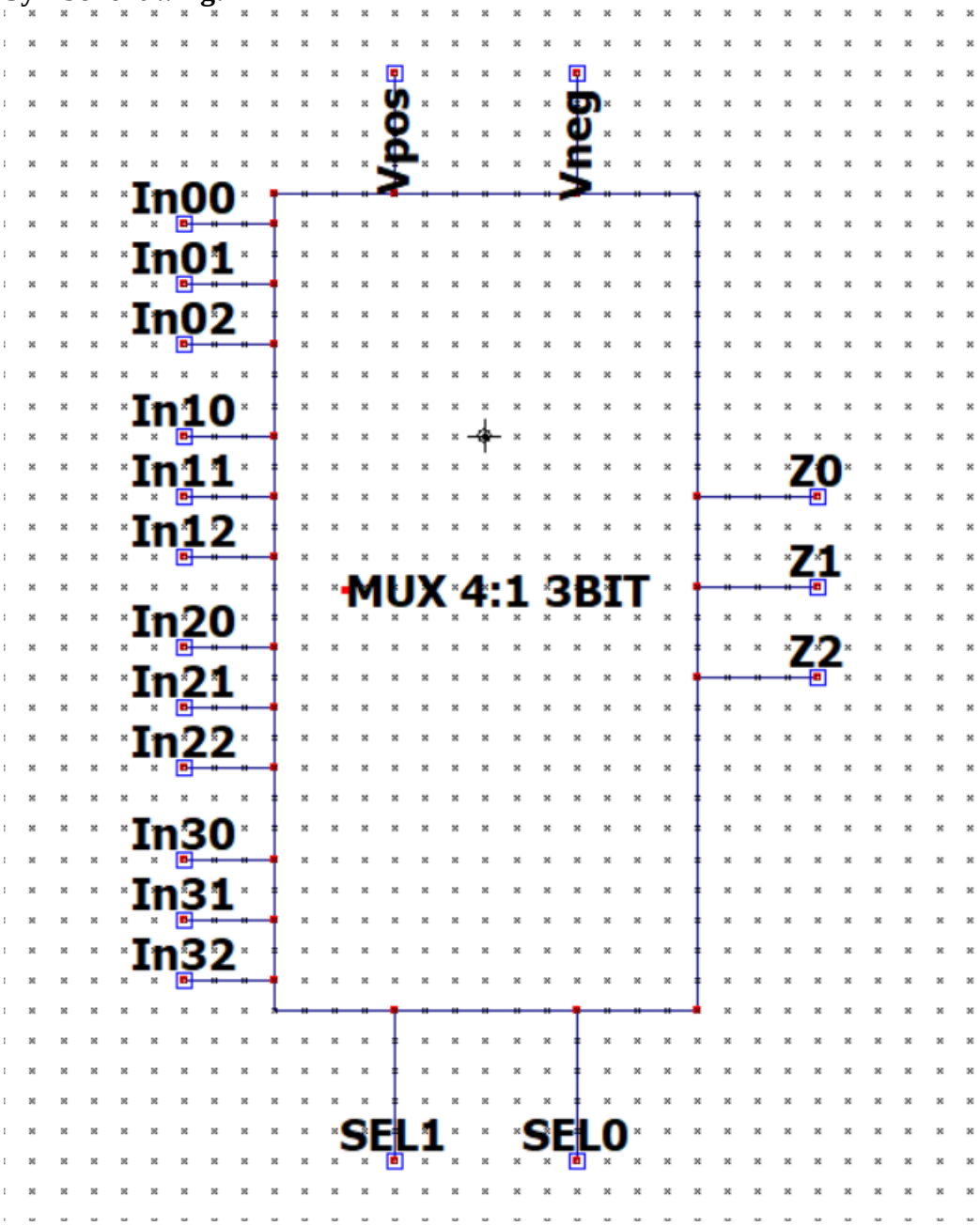
Gate-level drawing:



Symbol schematic:



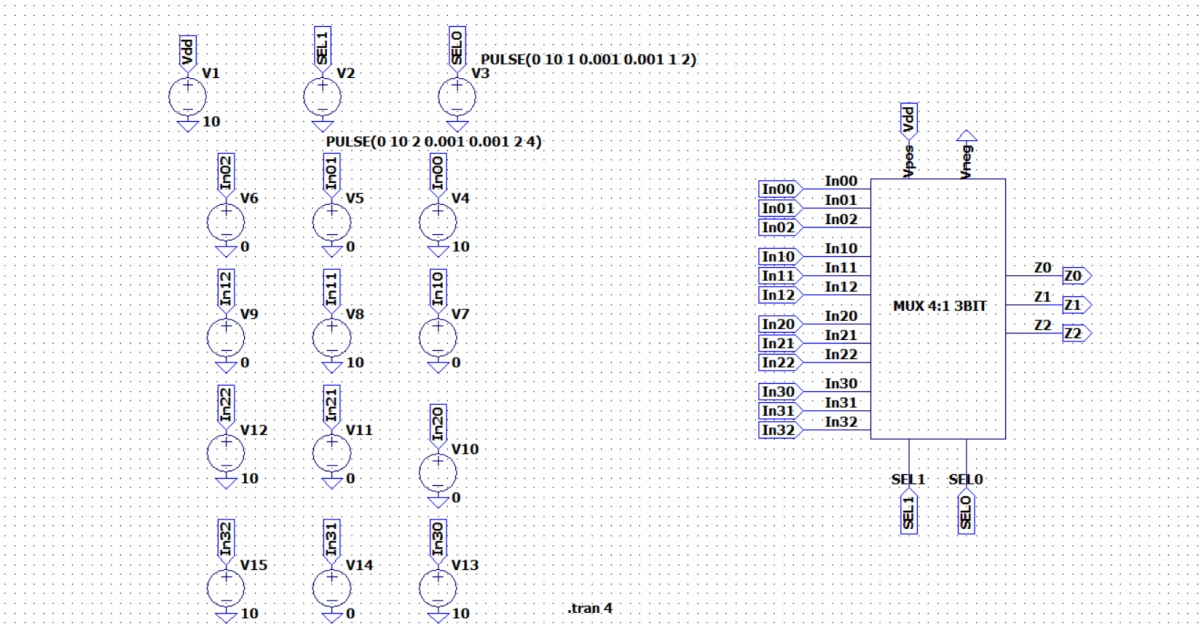
Symbol drawing:



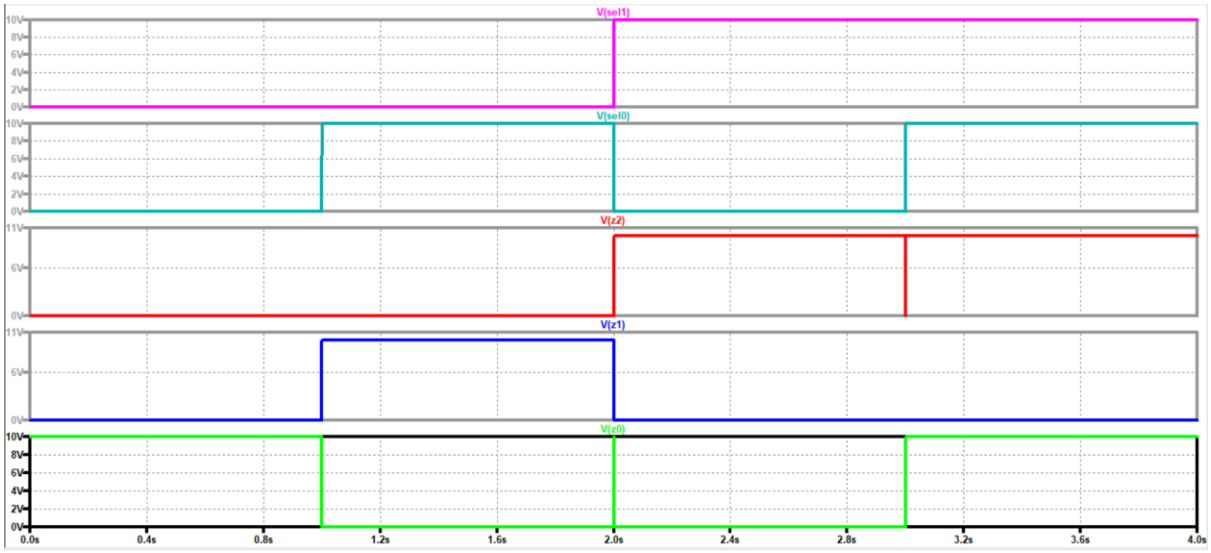
Truth table:

S.no	Data Inputs				SEL1 MSB	SEL0 LSB	Output (Z2-Z0)
	In02-In01(A)	In10-In12(B)	In20-In01(C)	In30-In32(D)			
0	001	010	100	101	0	0	001(A)
1	001	010	100	101	0	1	010(B)
2	001	010	100	101	1	0	100(C)
3	001	010	100	101	1	1	101(D)

Test circuit:



Test circuit waveforms:

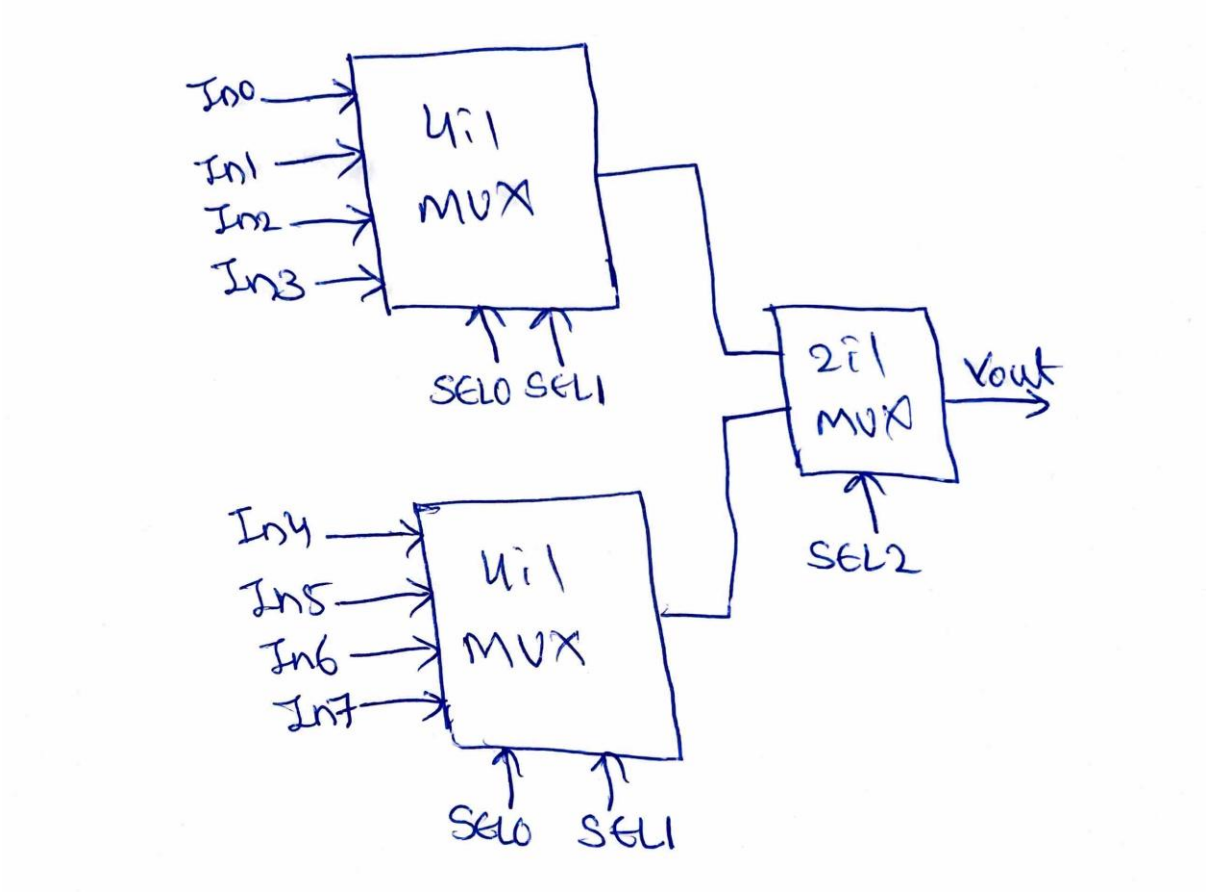


Verification statement:

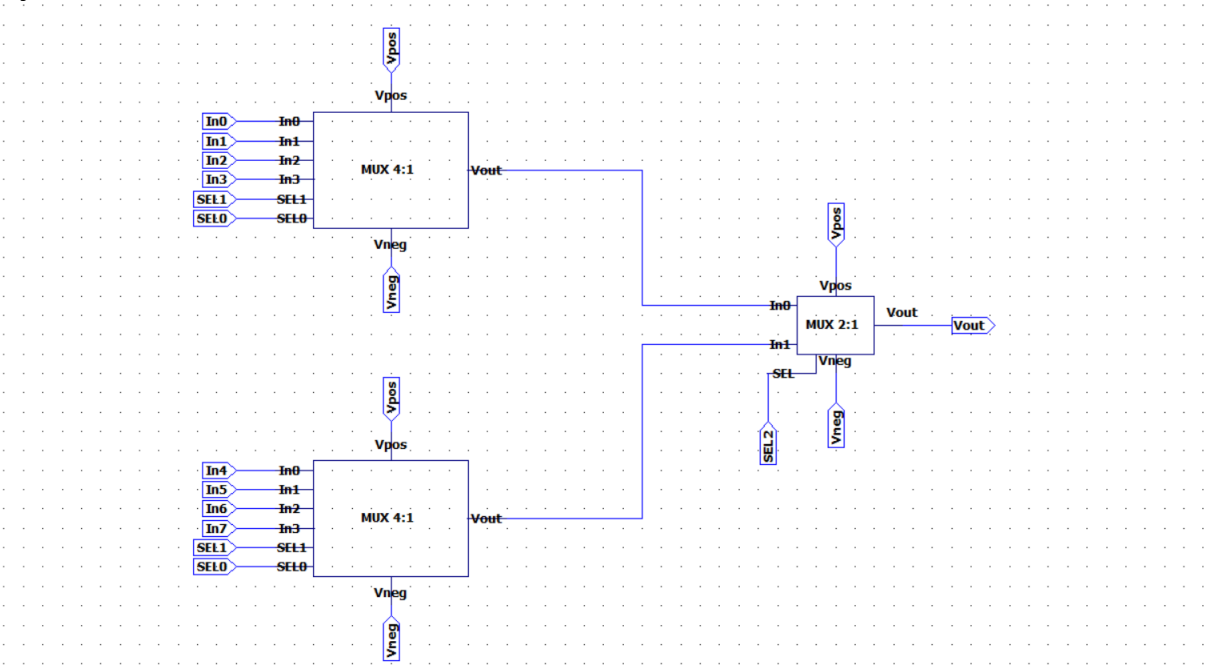
The 3bit 4:1 MUX is designed and verified with output waveforms to the truth table outputs data. Hence, the 3bit 4:1 MUX is working properly for the given data inputs.

MUX 8:1:

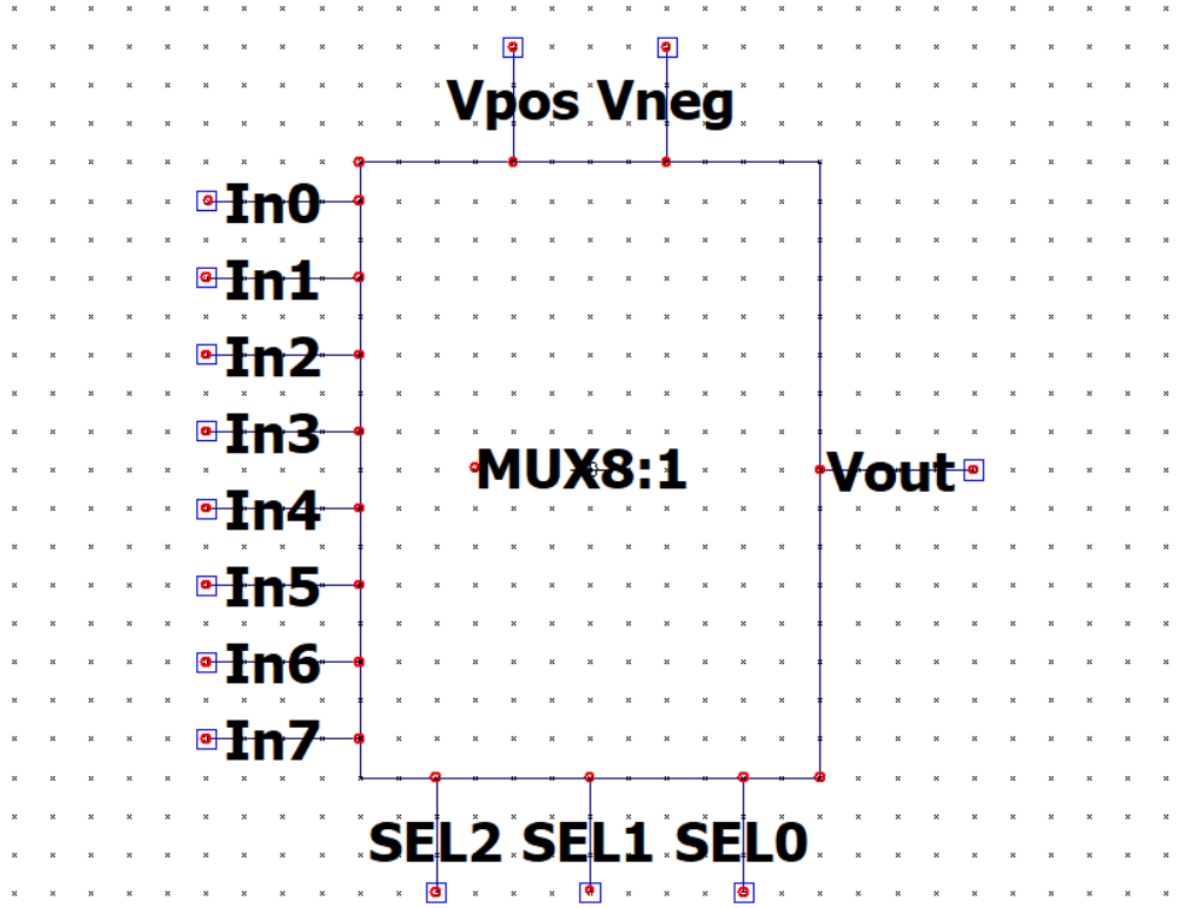
Gate-level drawing:



Symbol schematic:



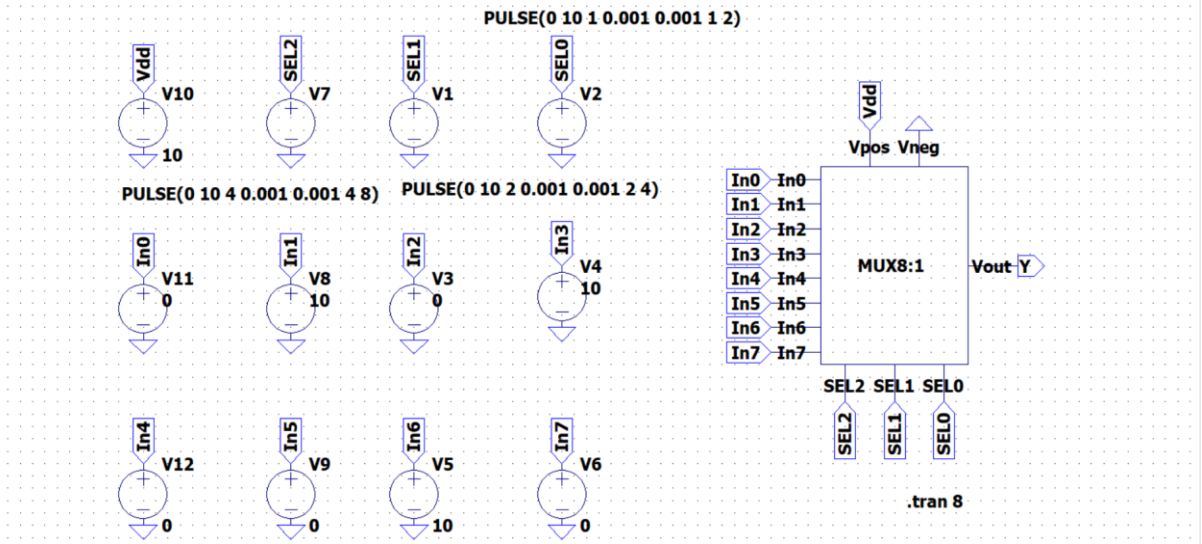
Symbol drawing:



Truth table:

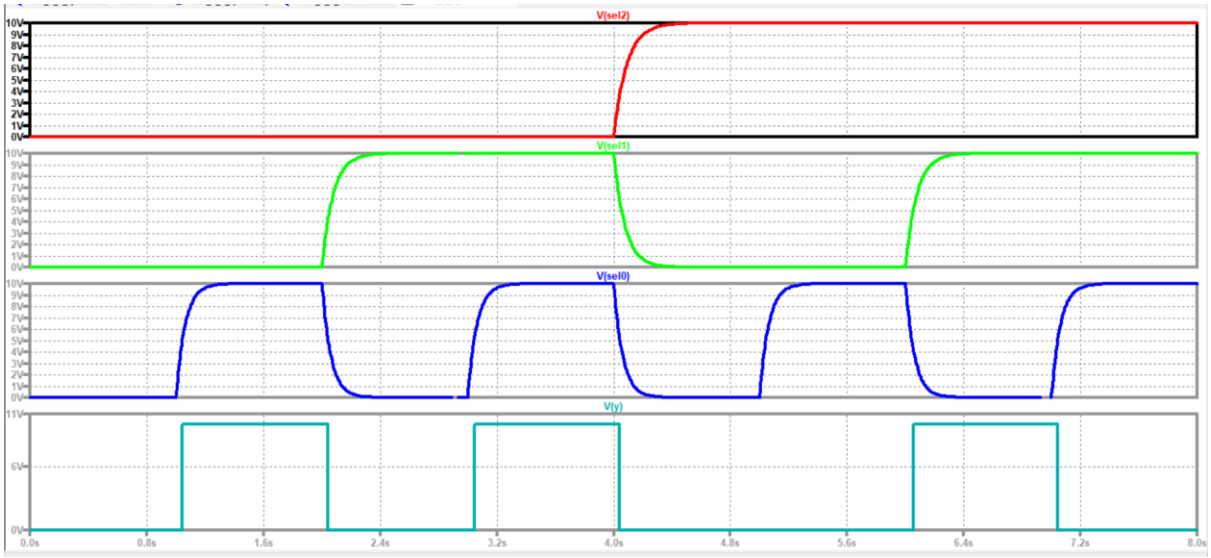
S.no	Data Inputs								SEL2 MSB	SEL1	SEL0 LSB	Output (Y)
	In0	In1	In2	In3	In4	In5	In6	In7				
0	0	1	0	1	0	0	1	0	0	0	0	0(In0)
1	0	1	0	1	0	0	1	0	0	0	1	1(In1)
2	0	1	0	1	0	0	1	0	0	1	0	0(In2)
3	0	1	0	1	0	0	1	0	0	1	1	1(In3)
4	0	1	0	1	0	0	1	0	1	0	0	0(In4)
5	0	1	0	1	0	0	1	0	1	0	1	0(In5)
6	0	1	0	1	0	0	1	0	1	1	0	1(In6)
7	0	1	0	1	0	0	1	0	1	1	1	0(In7)

Test circuit:



Test circuit waveforms:

Used parasitic properties for the selection inputs.

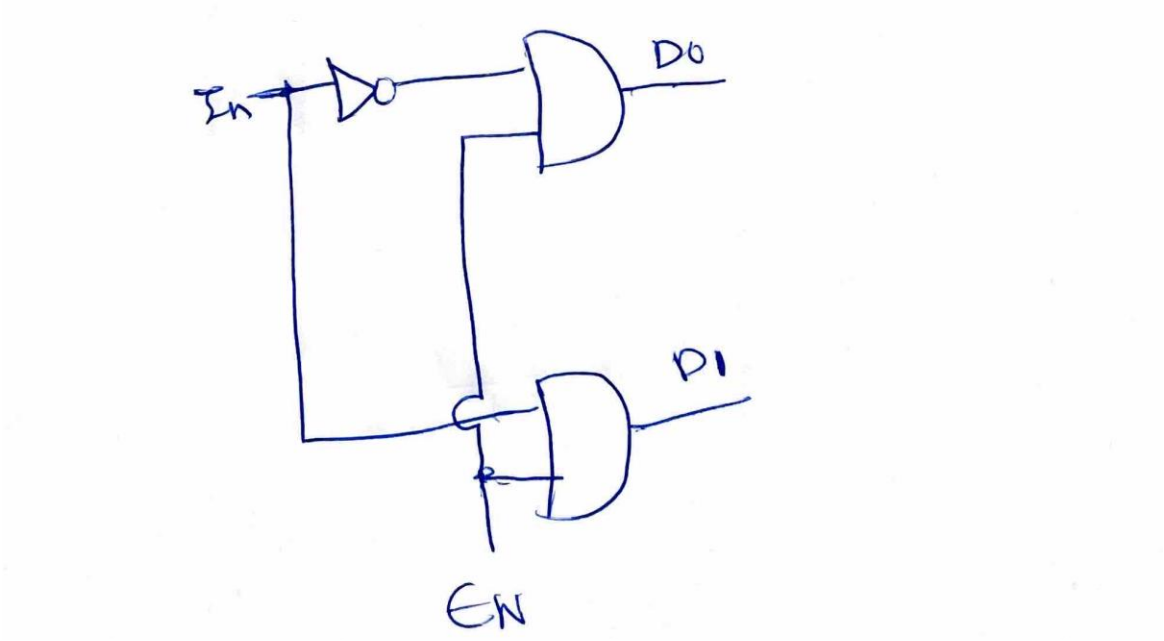


Verification statement:

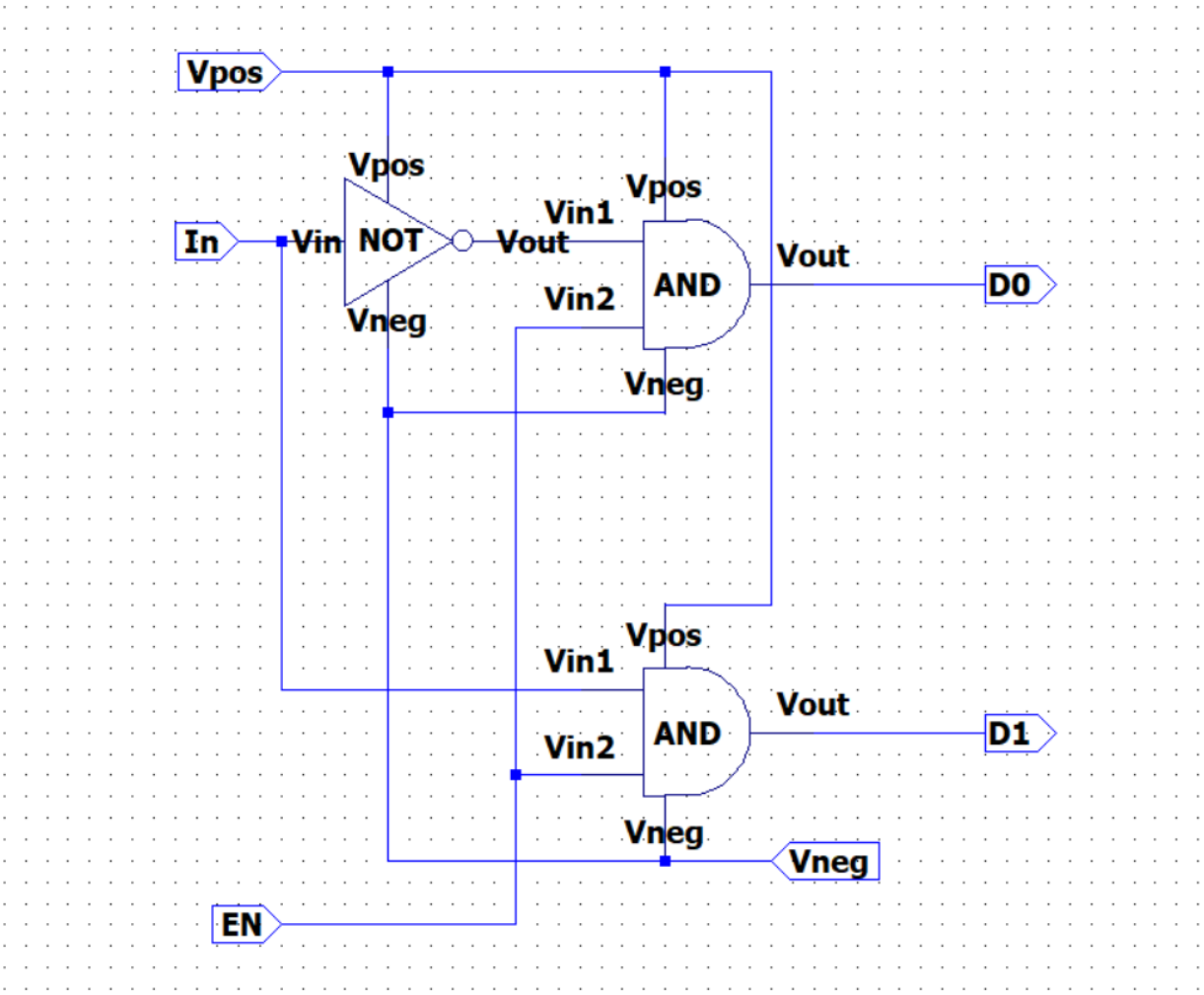
The 8:1 MUX is designed and verified with output waveforms to the truth table outputs data. Hence, the 8:1 MUX is working properly for the given data inputs.

DECODER 1:2:

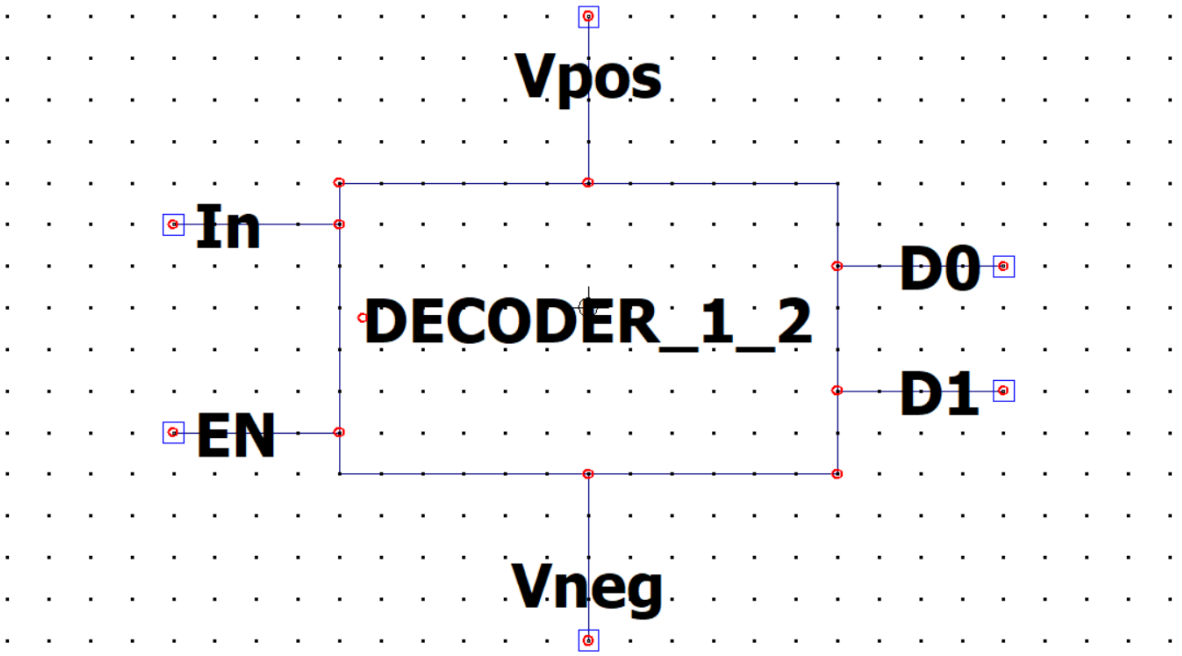
Gate-level drawing:



Symbol schematic:



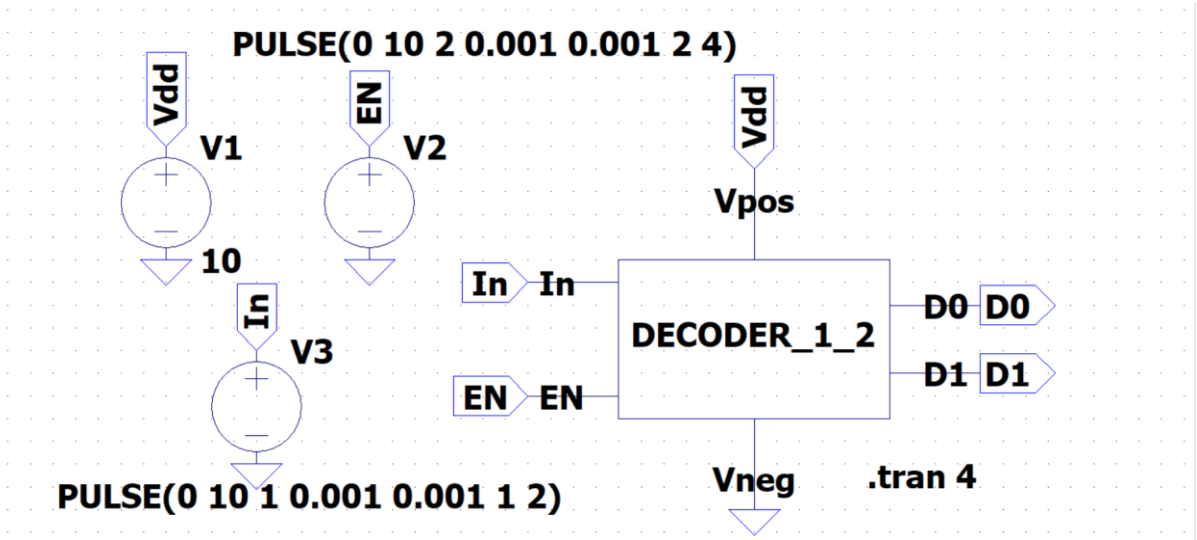
Symbol drawing:



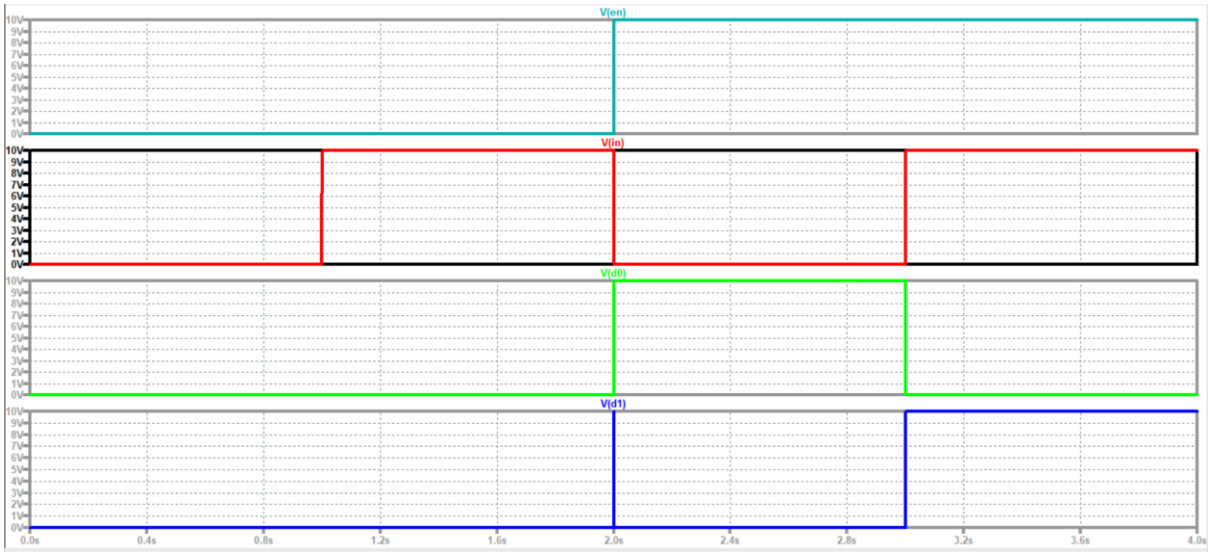
Truth table:

yS.no	Data Inputs	Outputs	
	IN	D0	D1
0	0	1	0
1	1	0	1

Test circuit:



Test circuit waveforms:

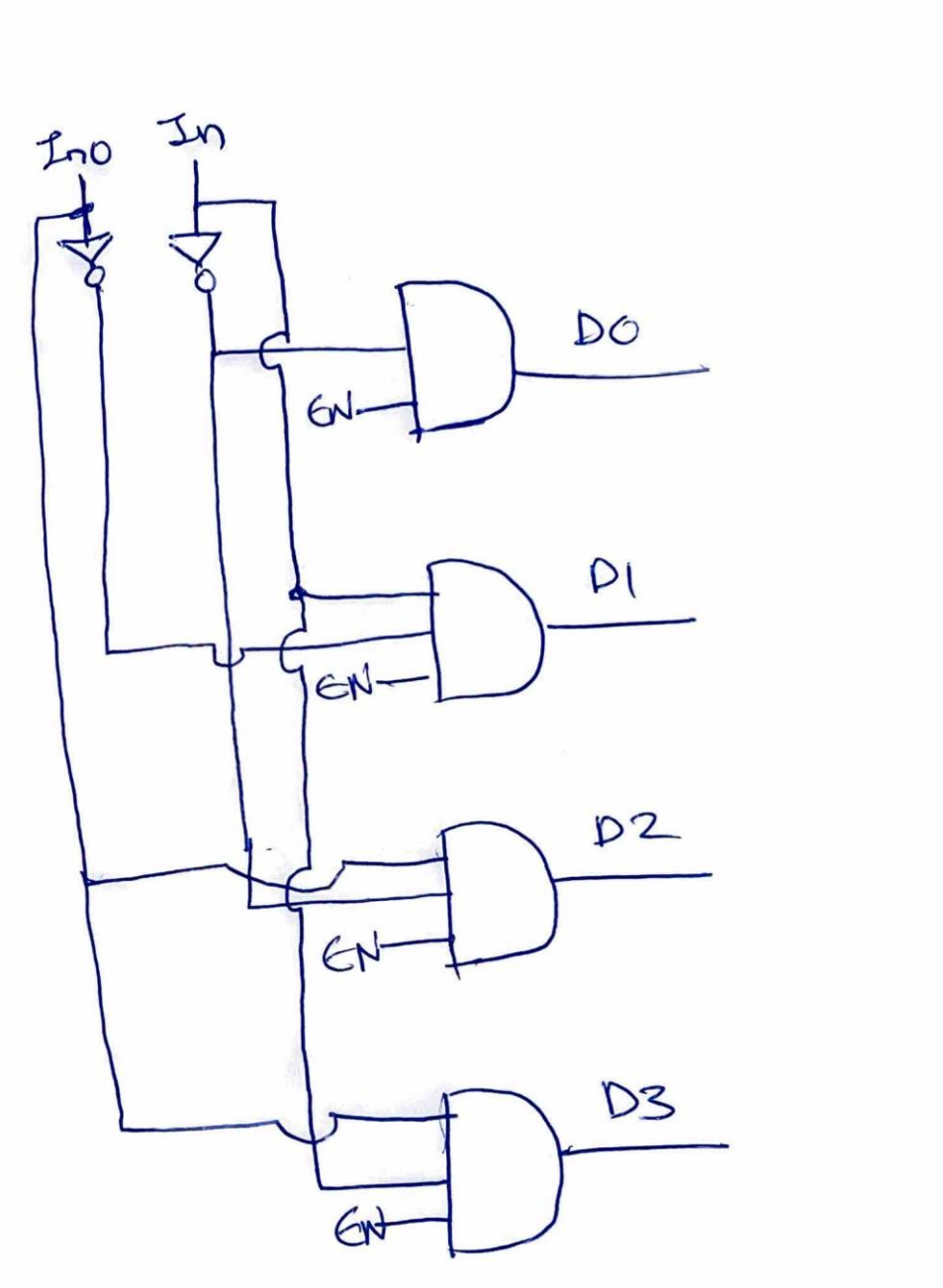


Verification statement:

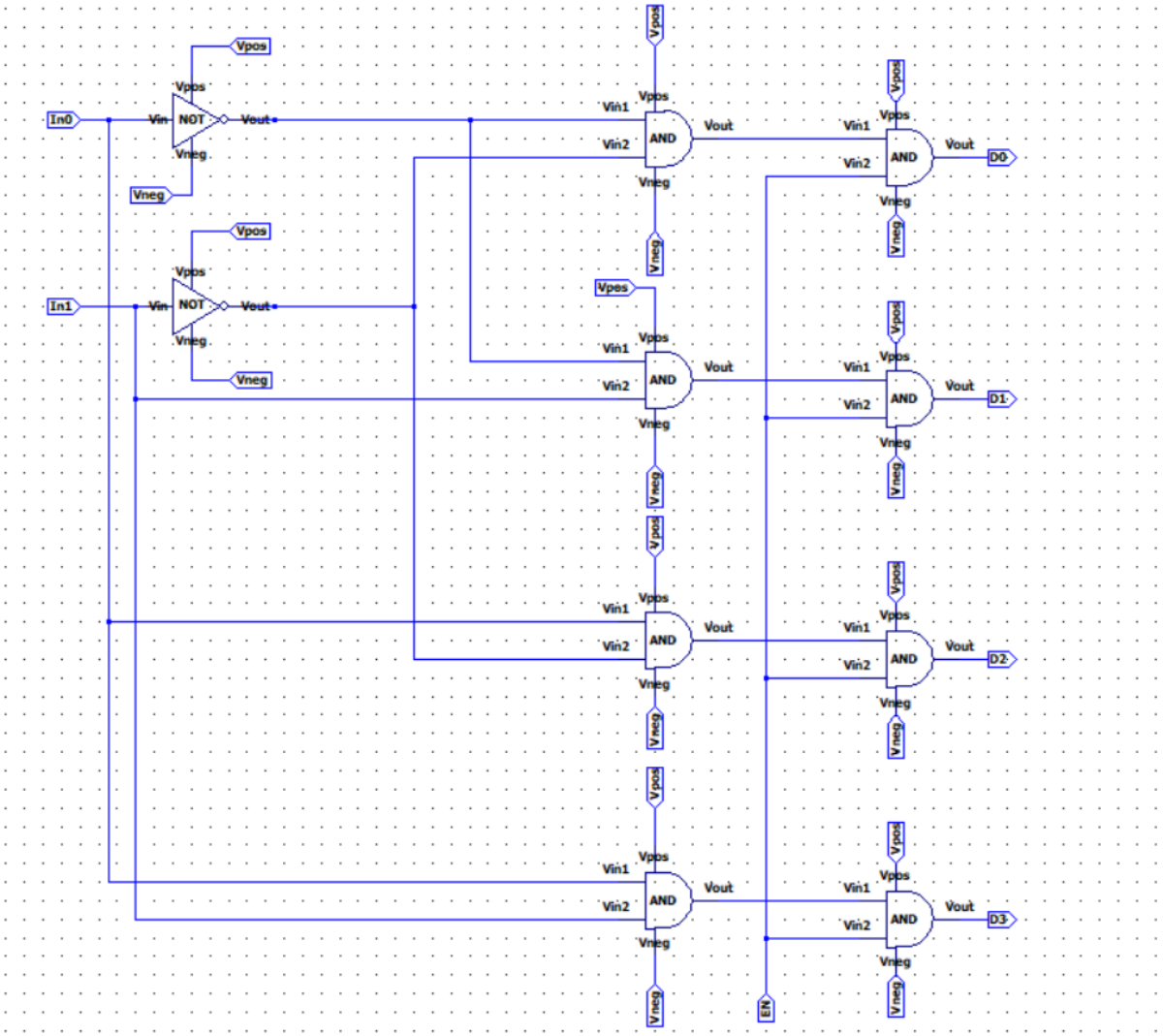
The 1:2 Decoder output waveforms are exactly matched with the truth table output data. Therefore, the designed 1:2 Decoder has been verified successfully and perfectly working for the given input data signals.

DECODER2:4:

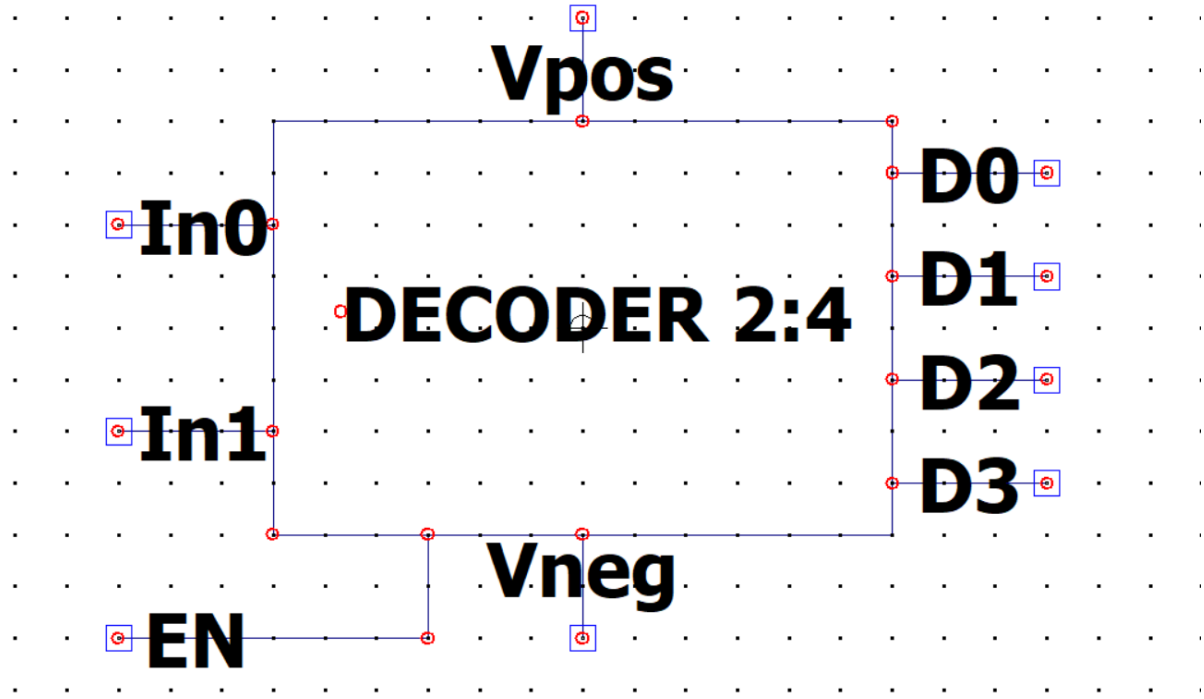
Gate-level drawing:



Symbol schematic:



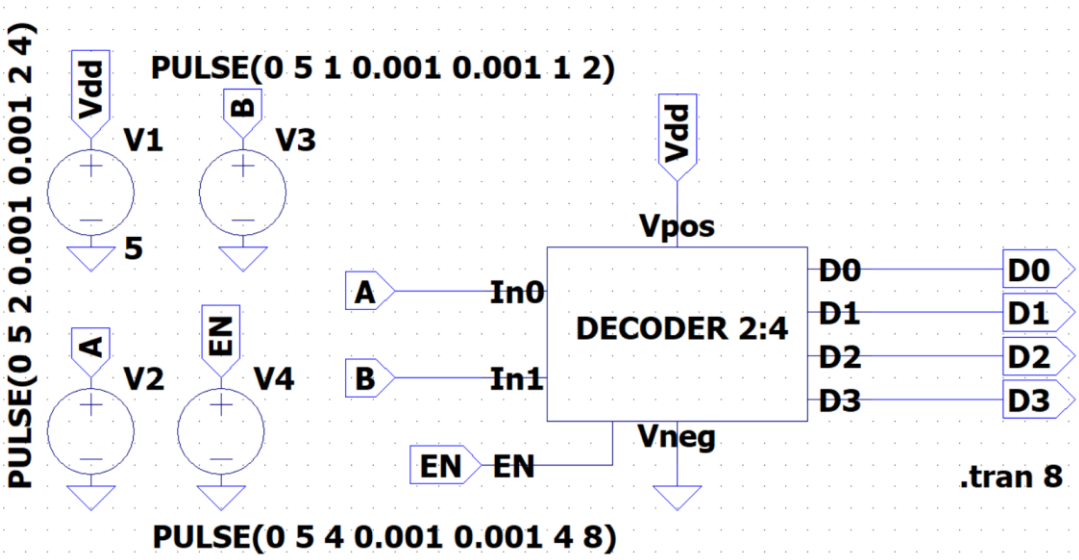
Symbol drawing:



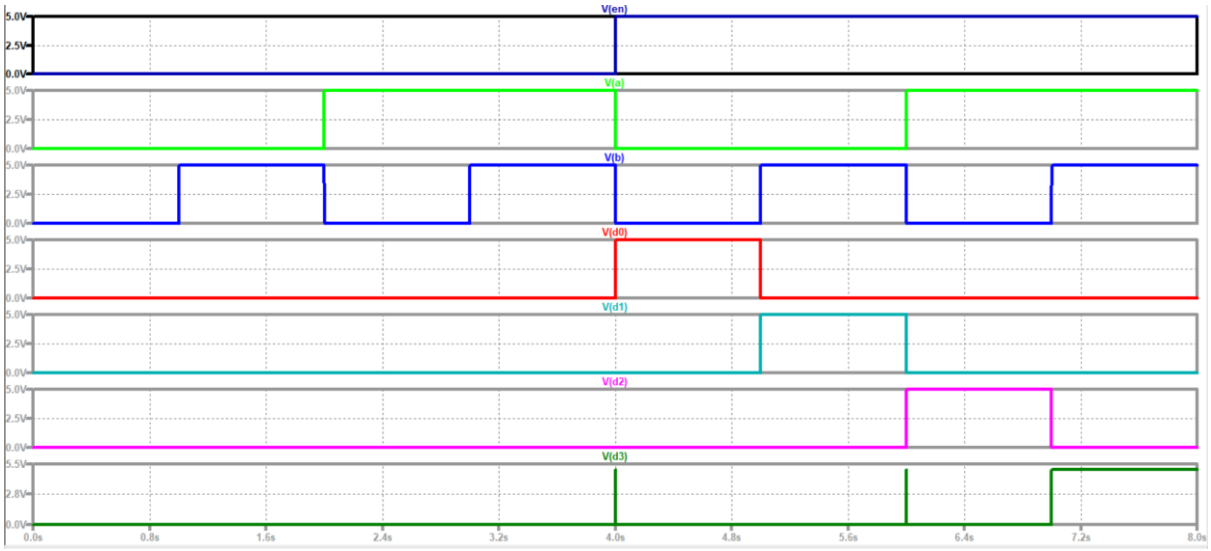
Truth table:

S.no	Data Inputs		Outputs			
	IN0 MSB	IN1 LSB	D0	D1	D2	D3
0	0	0	1	0	0	0
1	0	1	0	1	0	0
2	1	0	0	0	1	0
3	1	1	0	0	0	1

Test circuit:



Test circuit waveforms:



Verification statement:

The 2:4 Decoder output waveforms are exactly matched with the truth table output data. Therefore, the designed 2:4 Decoder has been verified successfully and perfectly working for the given input data signals.