COMBINATIONL LOGIC CIRCUITS

OBJECTIVE

- To design, construct, and test four combinational logic circuits. The first two circuits are to be constructed with NAND gates, the third with XOR gates, and the fourth with a decoder and NAND gates.

CIRCUIT 1

To design a combinational circuit with four inputs— A, B, C, and D —and one output, F . F is to be equal to 1 when A = 1, provided that B = 0, or when B = 1, provided that either C or D is also equal to 1. Otherwise, the output is to be equal to 0.

Truth Table

A	B	C	D	F
0	0	0	0	**0**
0	0	0	1	**0**
0	0	1	0	**0**
0	0	1	1	**0**
0	1	0	0	**0**
0	1	0	1	**1**
0	1	1	0	**1**
0	1	1	1	**1**

A	B	C	D	F
1	0	0	0	**1**
1	0	0	1	**1**
1	0	1	0	**1**
1	0	1	1	**1**
1	1	0	0	**0**
1	1	0	1	**1**
1	1	1	0	**1**
1	1	1	1	**1**

F = AB' + BC +BD

Function using NAND gates

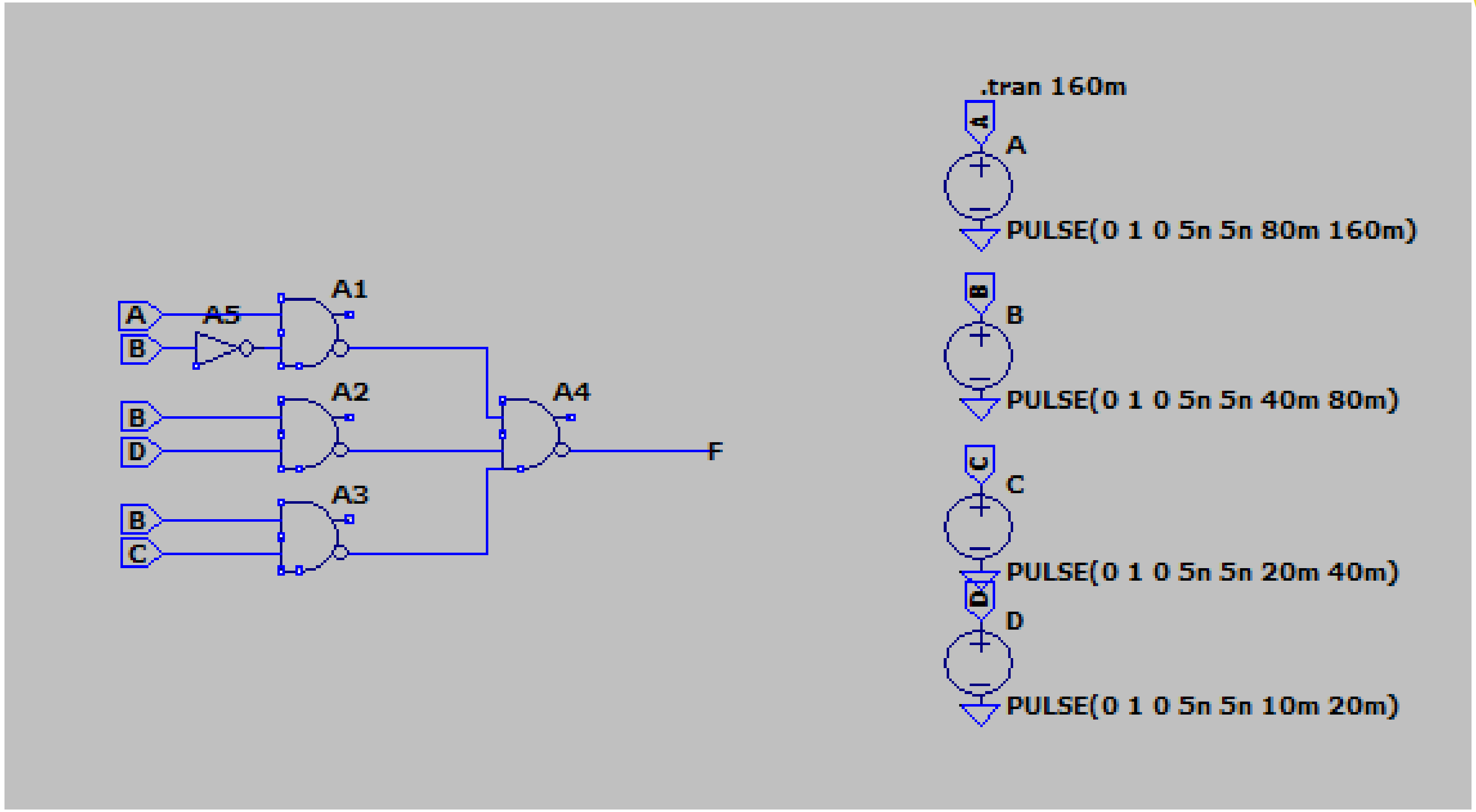
$F = AB' + BC + BD$

$F' = (AB' + BC + BD)'$

$(F')' = [(AB')'(BC)'(BD)']'$

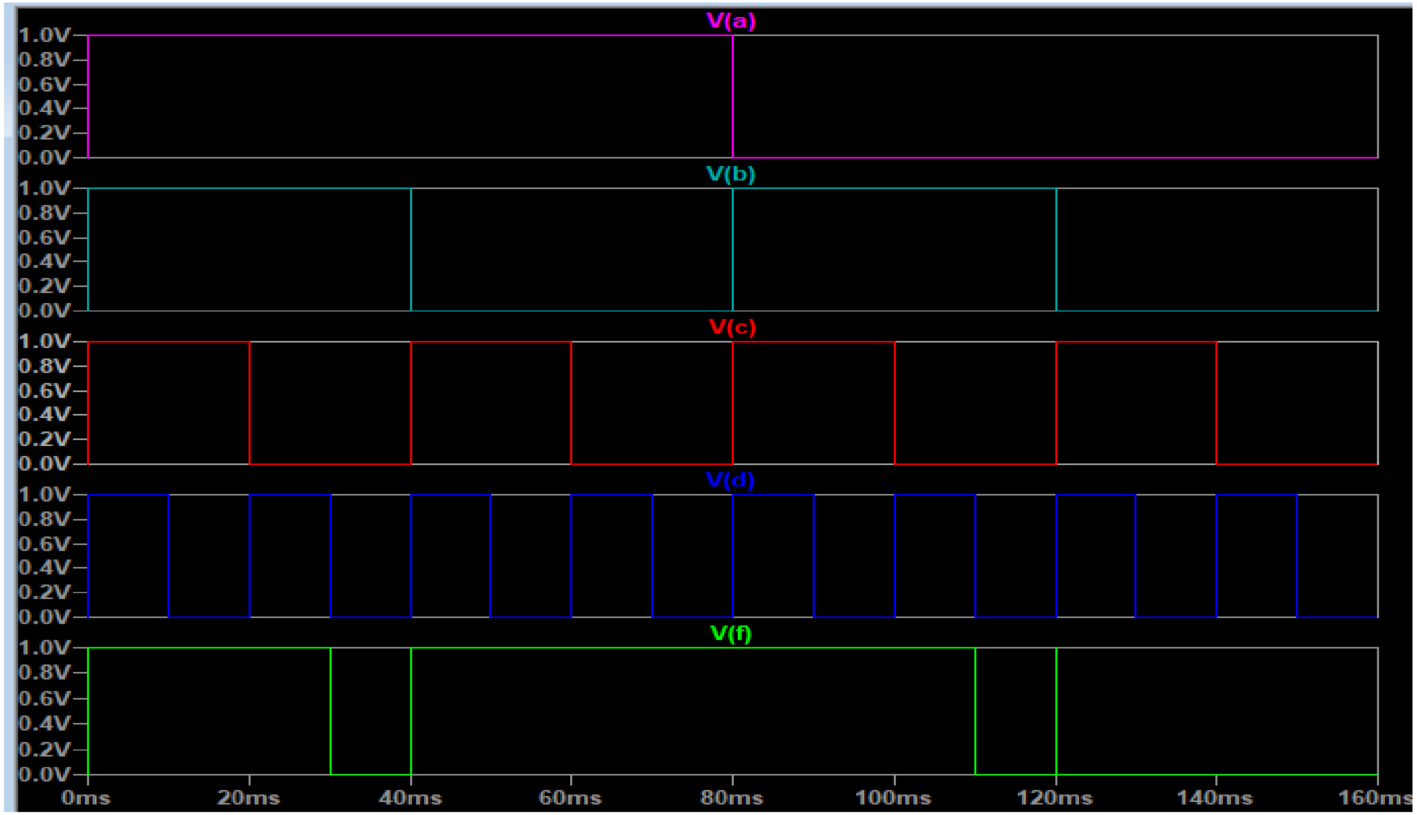
$F = [(AB')'(BC)'(BD)']'$

Logic Diagram in LTSpice



```
.tran 160m
PULSE(0 1 0 5n 5n 80m 160m)
PULSE(0 1 0 5n 5n 40m 80m)
PULSE(0 1 0 5n 5n 20m 40m)
PULSE(0 1 0 5n 5n 10m 20m)
```

Output from LTSpice



CIRCUIT 2 (Majority Logic)

A majority logic is a digital circuit whose output is equal to 1 if the majority of the inputs are 1's. The output is 0 otherwise. Design and test a three-input majority circuit using NAND gates with a minimum number of ICs.

Truth Table

A	B	C	F
0	0	0	**0**
0	0	1	**0**
0	1	0	**0**
0	1	1	**1**
1	0	0	**0**
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

F = AB + BC + CA

Function using NAND gates

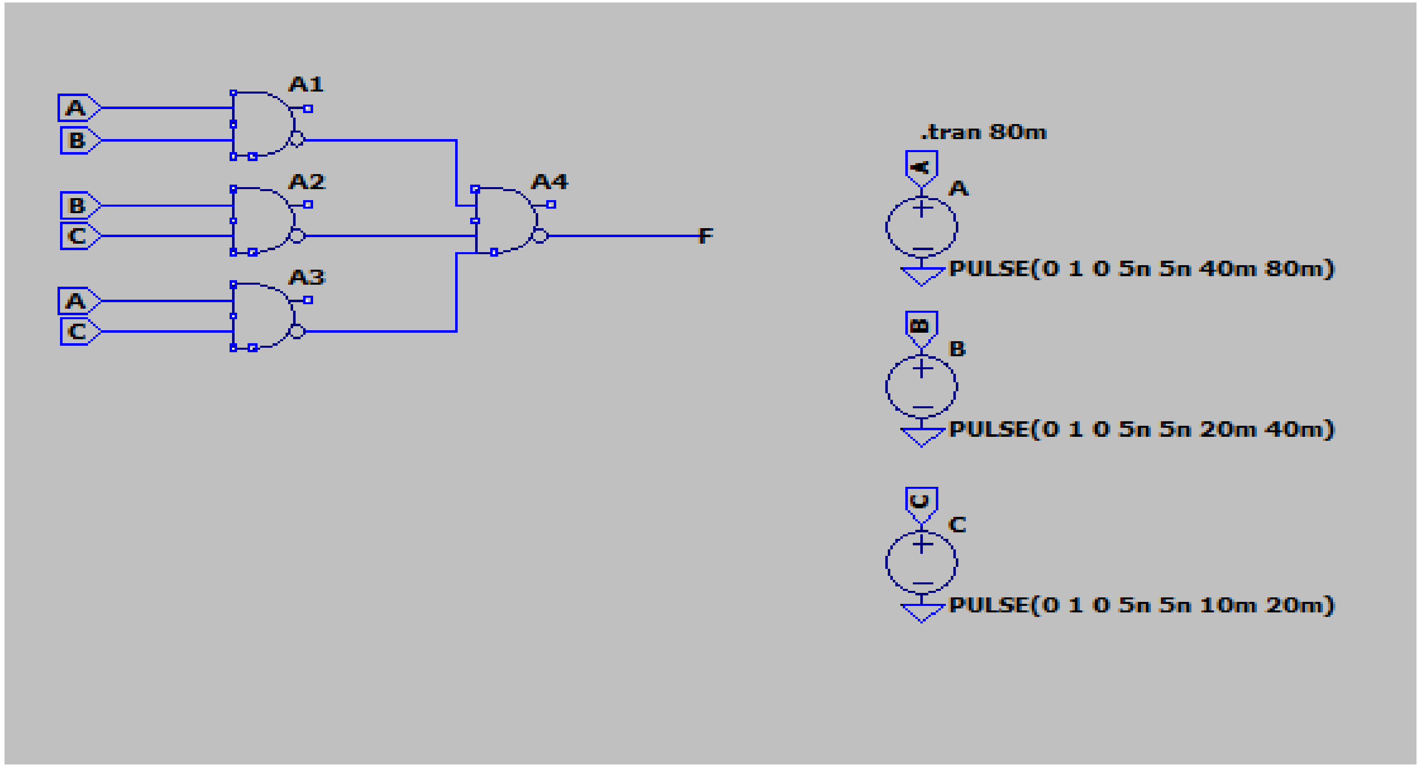
$F = AB + BC + CA$

$F' = (AB + BC + CA)'$

$(F')' = [(AB)'(BC)'(CA)']'$

$F = [(AB)'(BC)'(CA)']'$

Logic Diagram in LTSpice



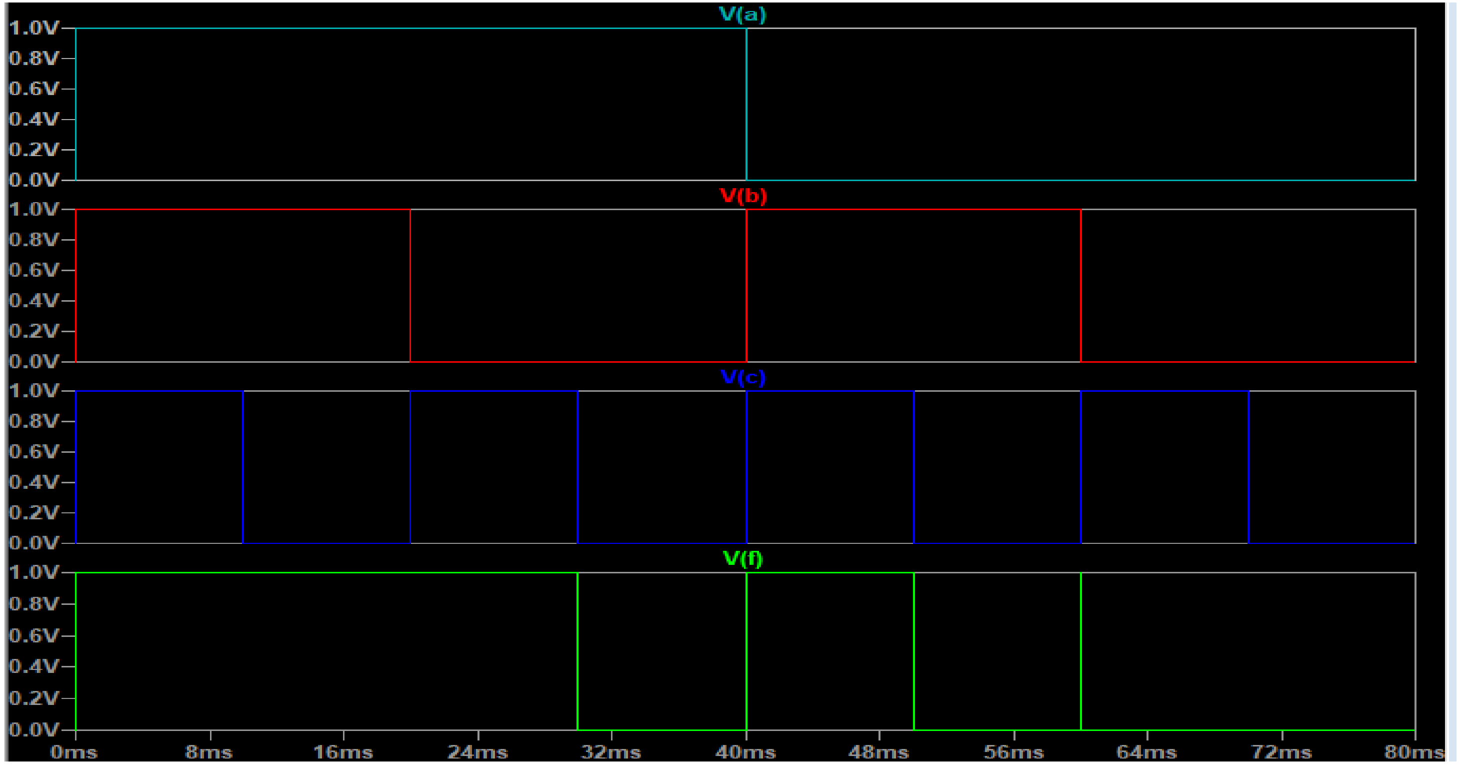
.tran 80m

PULSE(0 1 0 5n 5n 40m 80m)

PULSE(0 1 0 5n 5n 20m 40m)

PULSE(0 1 0 5n 5n 10m 20m)

Output from LTSpice



CIRCUIT 3(PARITY GENERATOR)

Design, construct, and test a circuit that generates an even parity bit from four message bits. Use XOR gates. Adding one more XOR gate, expand the circuit so that it generates an odd parity bit also.

Truth Table

A	B	C	D	**Even**	**Odd**
0	0	0	0	**0**	**1**
0	0	0	1	**1**	**0**
0	0	1	0	**1**	**0**
0	0	1	1	**0**	**1**
0	1	0	0	**1**	**0**
0	1	0	1	**0**	**1**
0	1	1	0	**0**	**1**
0	1	1	1	**1**	**0**

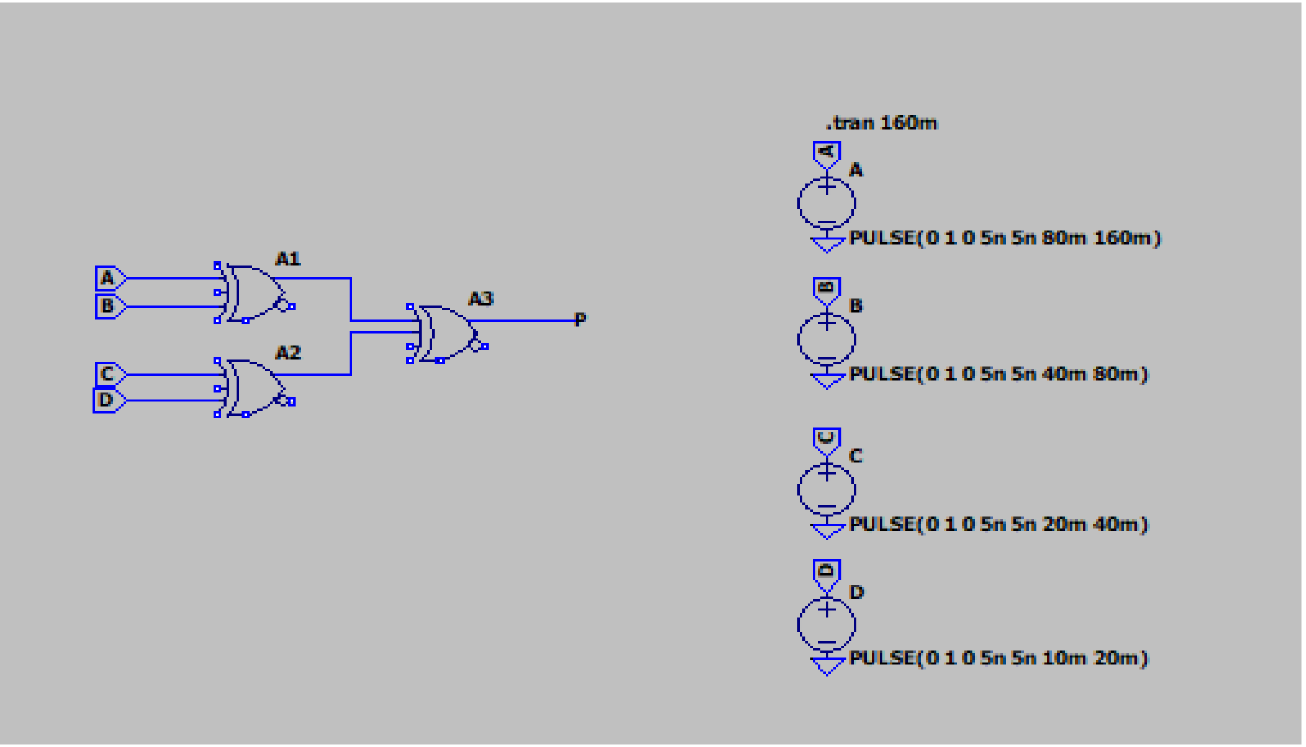
A	B	C	D	**Even**	**Odd**
1	0	0	0	**1**	**0**
1	0	0	1	**0**	**1**
1	0	1	0	**0**	**1**
1	0	1	1	**1**	**0**
1	1	0	0	**0**	**1**
1	1	0	1	**1**	**0**
1	1	1	0	**1**	**0**
1	1	1	1	**0**	**1**

Output Function

$EVEN = A \oplus B \oplus C \oplus D$
$EVEN = (A \oplus B) \oplus (C \oplus D)$

$ODD = EVEN'$
$ODD = EVEN \oplus 1$

Logic Diagram in LTSpice



PULSE(0 1 0 5n 5n 80m 160m)

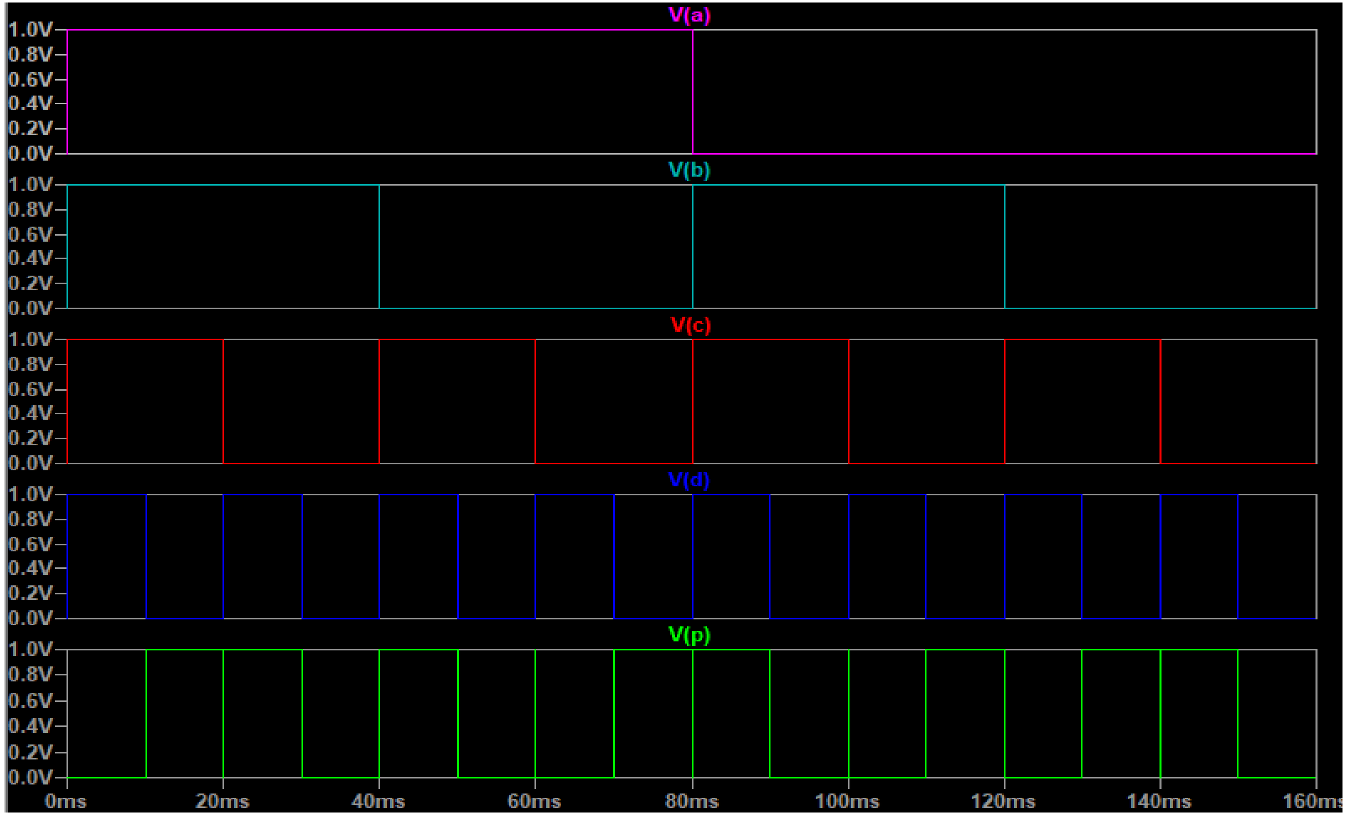
PULSE(0 1 0 5n 5n 40m 80m)

PULSE(0 1 0 5n 5n 20m 40m)

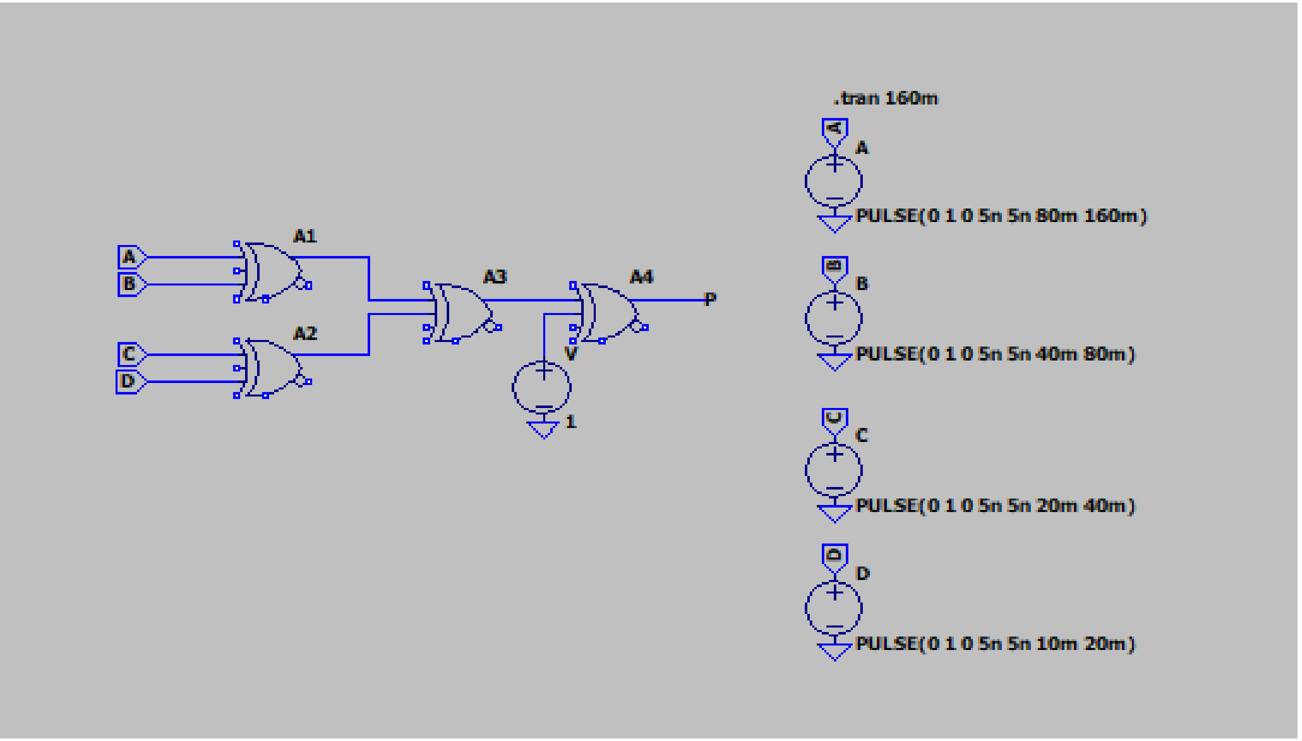
PULSE(0 1 0 5n 5n 10m 20m)

For Even Parity Checker

Output from LTSpice

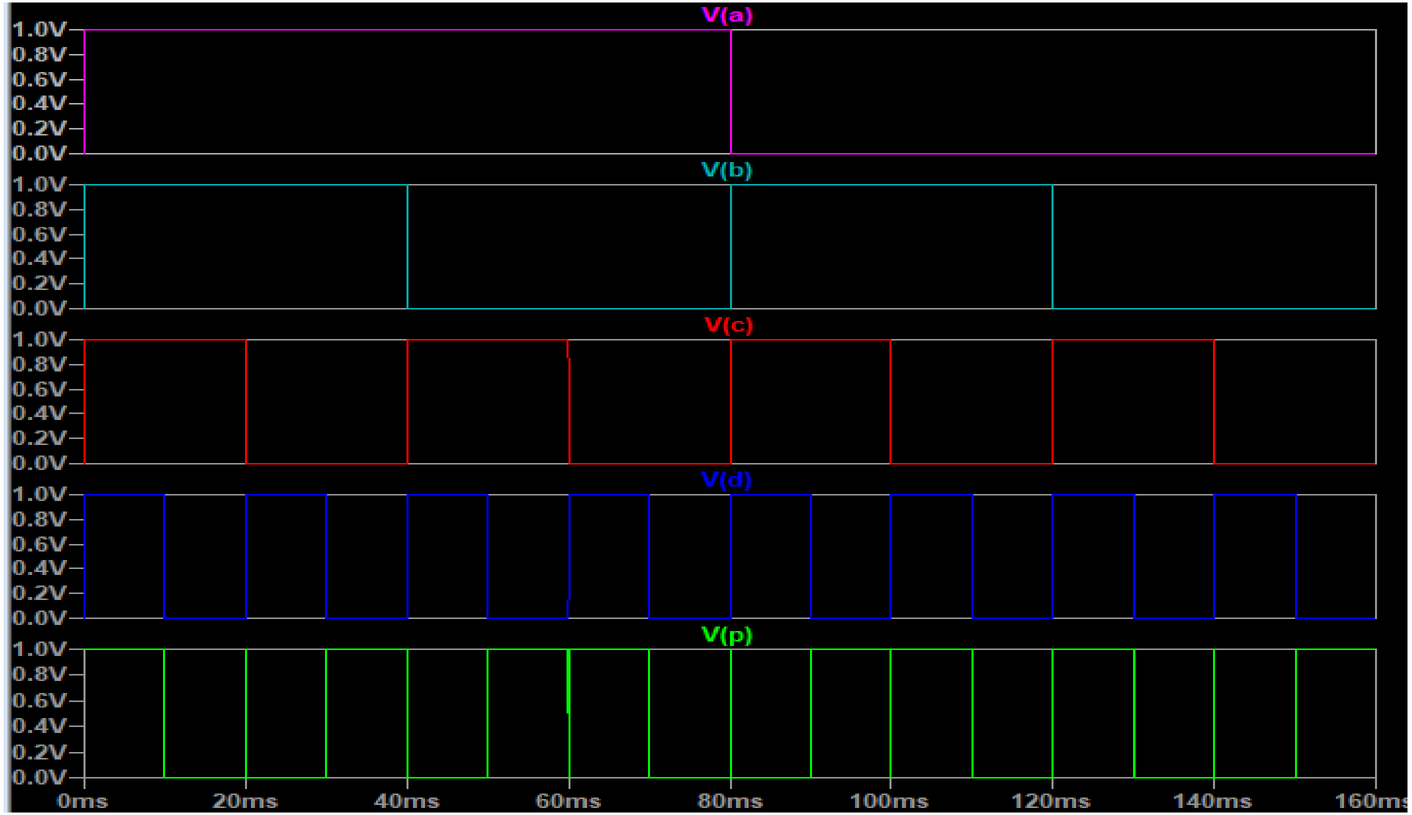


Logic Diagram in LTSpice


.tran 160m
A
B
C
D
A1
A2
A3
A4
P
V
1
PULSE(0 1 0 5n 5n 80m 160m)
PULSE(0 1 0 5n 5n 40m 80m)
PULSE(0 1 0 5n 5n 20m 40m)
PULSE(0 1 0 5n 5n 10m 20m)

For Odd Parity Checker

Output from LTSpice



CIRCUIT 4(DECODER IMPLEMENTATION)

A combinational circuit has three inputs— x, y, and z —and three outputs— F1, F2, and F3. The simplified Boolean functions for the circuit are

$$F1 = xz + xyz$$
$$F2 = xy + xyz$$
$$F3 = xy + xyz$$

Implement and test the combinational circuit, using a 74155 decoder IC and external NAND gates,

Truth Table

X	Y	Z	**F1**	**F2**	**F3**
0	0	0	**1**	**0**	**0**
0	0	1	**0**	**0**	**1**
0	1	0	**0**	**1**	**0**
0	1	1	**0**	**1**	**0**
1	0	0	**0**	**1**	**0**
1	0	1	**1**	**0**	**0**
1	1	0	**0**	**0**	**1**
1	1	1	**1**	**0**	**1**

F1 = XZ + X’Y’Z’

F2 = X’Y + XY’Z’

F3 = XY + X’Y’Z

Functions using NAND Gates

$F1 = XZ + X'Y'Z'$

$F1 = XZ(Y + Y') + X'Y'Z'$

$F1 = XYZ + XY'Z + X'Y'Z'$

$(F1)' = [(XYZ)'(XY'Z)'(X'Y'Z')]'$

$F1 = [(XYZ)'(XY'Z)'(X'Y'Z')]'$

$F2 = X'Y + XY'Z'$

$F2 = X'Y(Z + Z') + XY'Z'$

$F2 = X'YZ + X'YZ' + XY'Z'$

$(F2)' = [(X'YZ)'(X'YZ')'(XY'Z')]'$

$F2 = [(X'YZ)'(X'YZ')'(XY'Z')]'$

$F3 = XY + X'Y'Z$

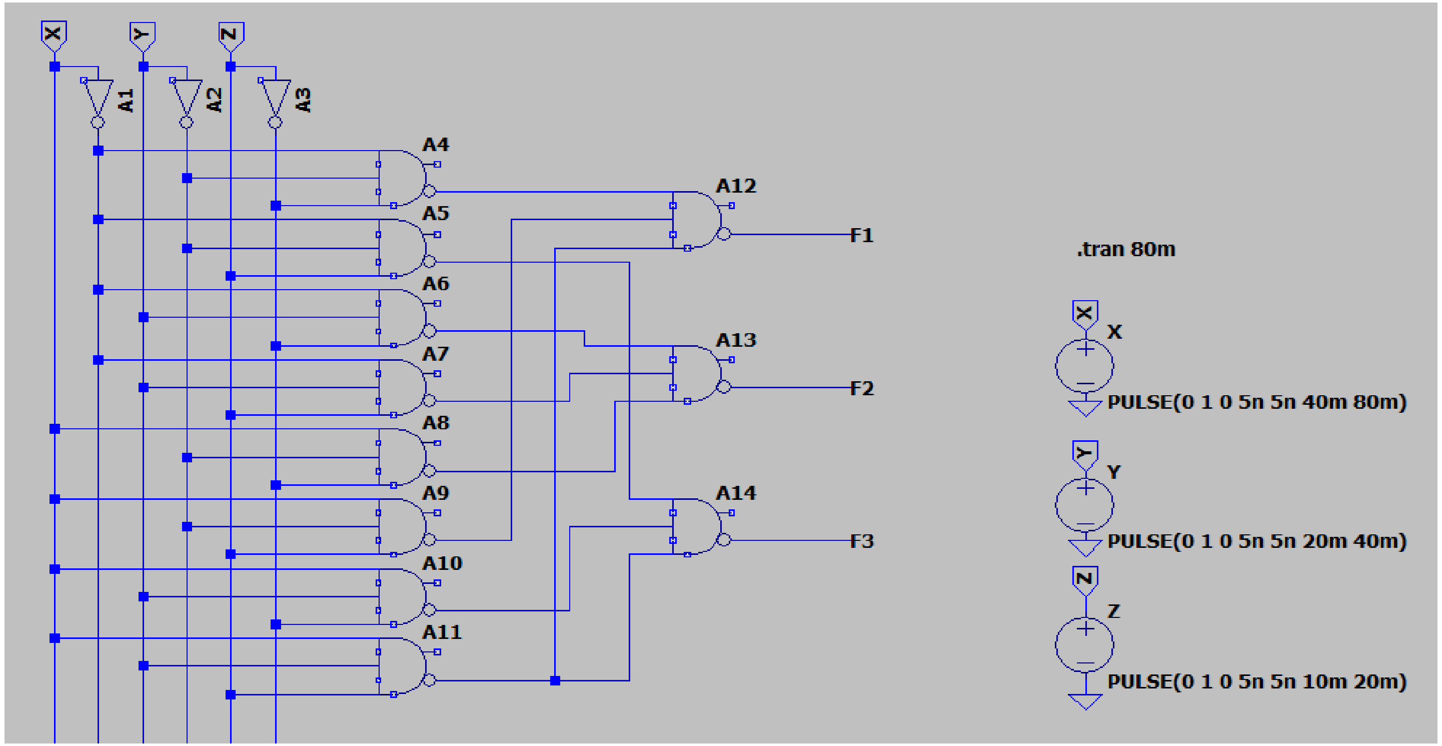
$F3 = XY(Z + Z') + X'Y'Z$

$F3 = XYZ + XYZ' + X'Y'Z$

$(F3)' = [(XYZ)'(XYZ')'(X'Y'Z)]'$

$F3 = [(XYZ)'(XYZ')'(X'Y'Z)]'$

Logic Diagram in LTSpice



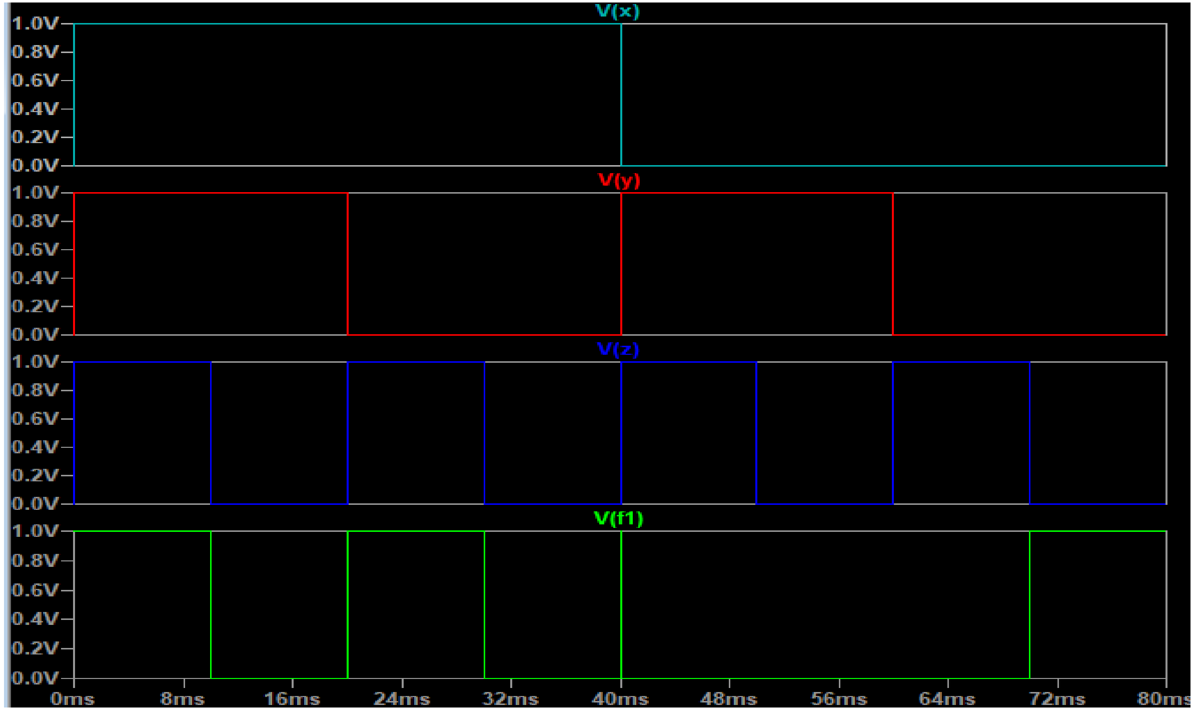
.tran 80m

PULSE(0 1 0 5n 5n 40m 80m)

PULSE(0 1 0 5n 5n 20m 40m)

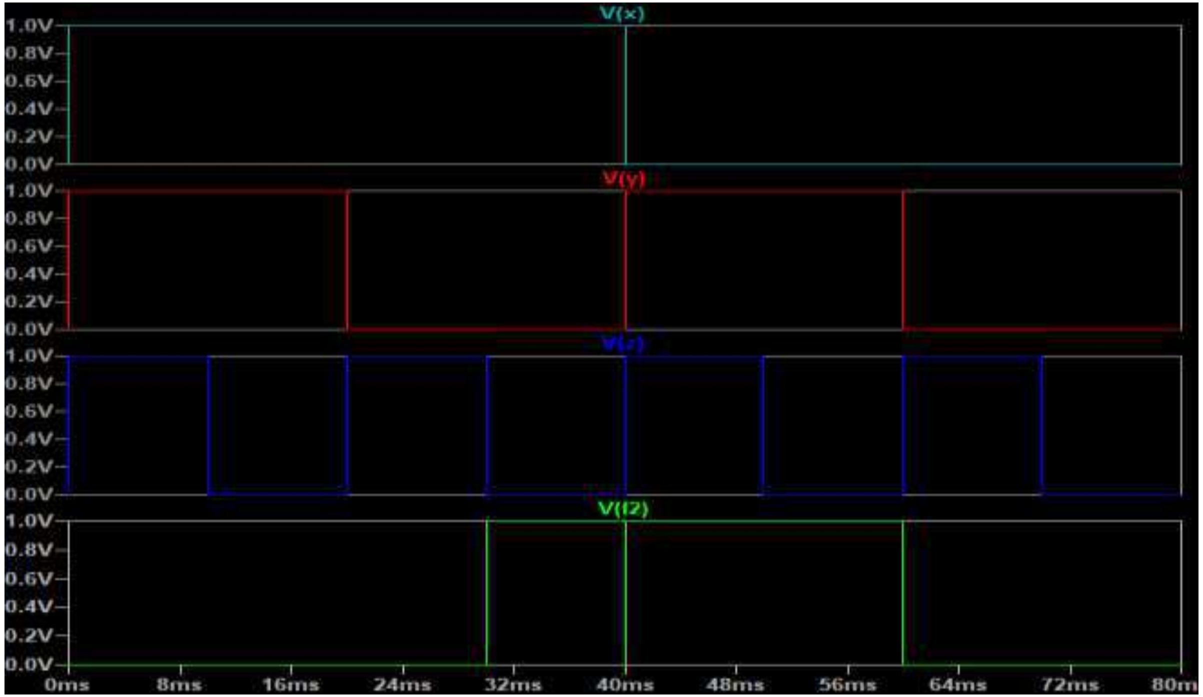
PULSE(0 1 0 5n 5n 10m 20m)

Output from LTSpice



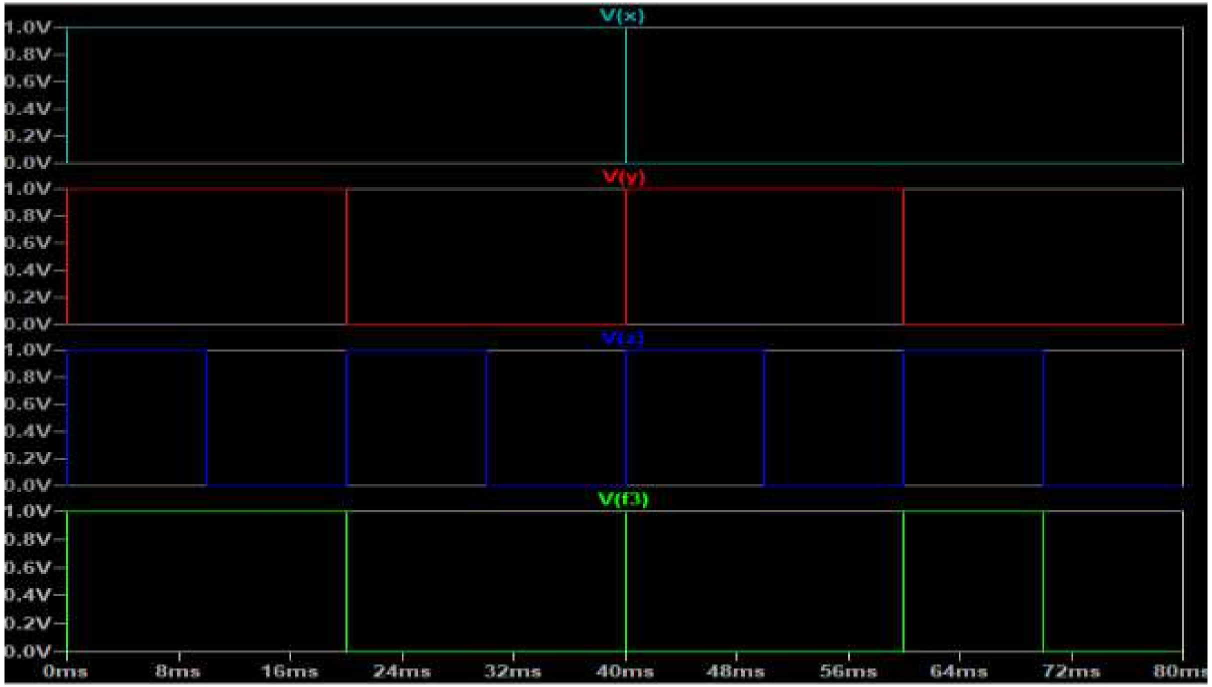
For F1

Output from LTSpice



For F2

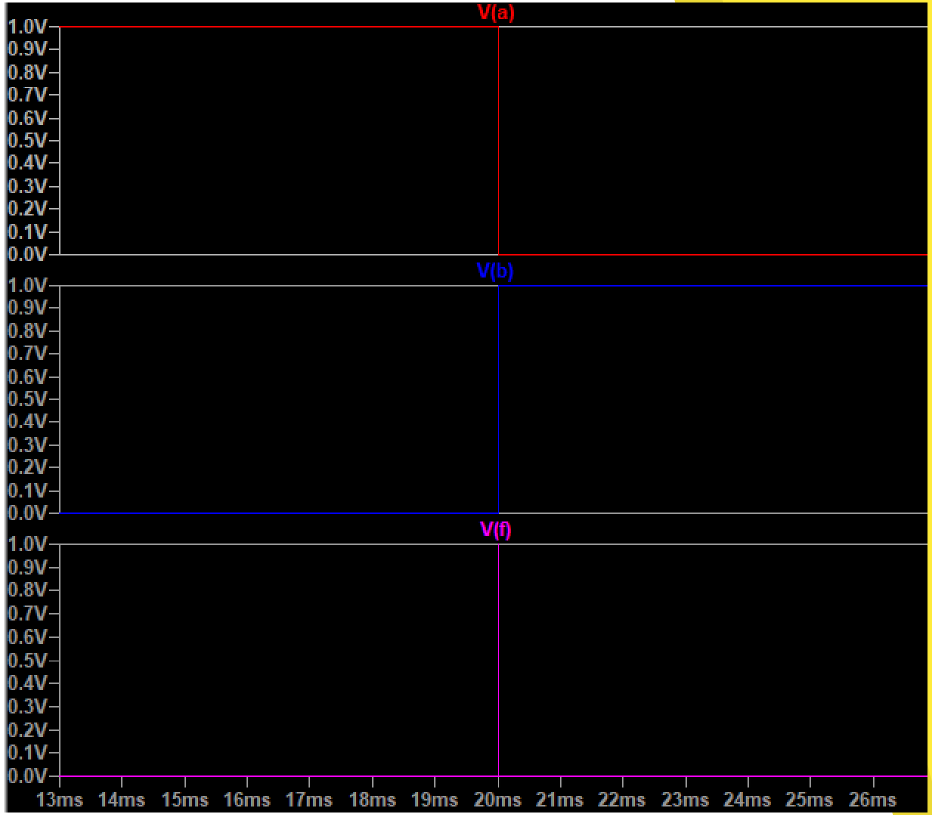
Output from LTSpice



For F3

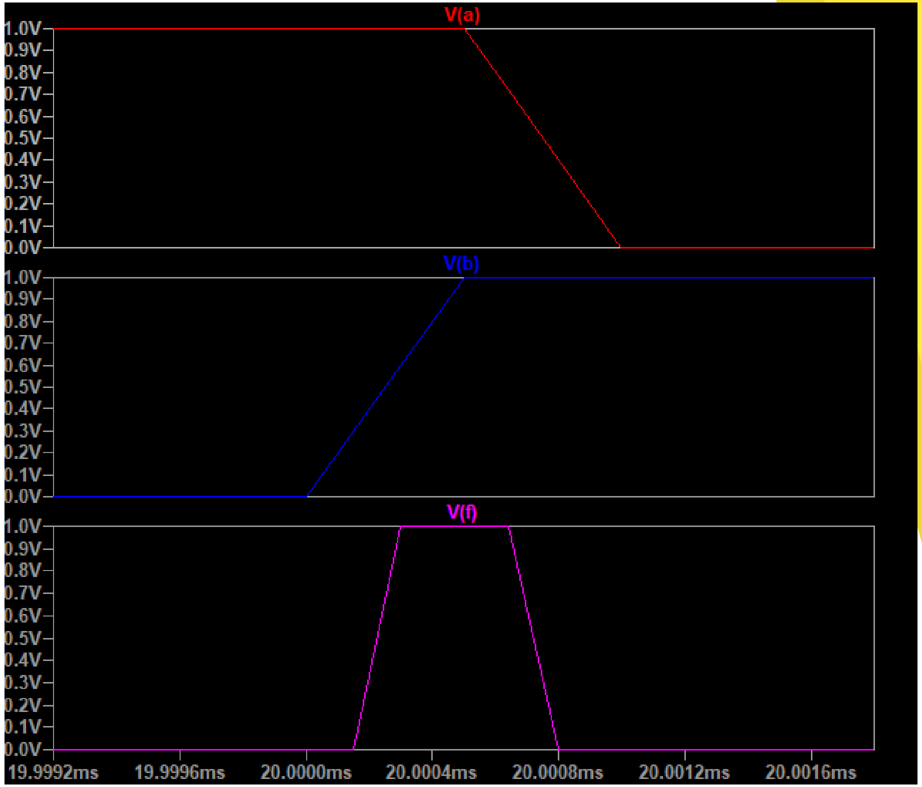
Spikes in the Output

The waveform shown by LTspice is not exactly ideal. We set the voltage to instantly turn from 0 to 1V, but the program executes this by ramping up in a short amount of time. This creates spikes in the output waveform.



Zooming the spike

The diagram shows one such transition and how it changes the output waveform.


V(a)
V(b)
V(f)
1.0V
0.9V
0.8V
0.7V
0.6V
0.5V
0.4V
0.3V
0.2V
0.1V
0.0V
19.9992ms
19.9996ms
20.0000ms
20.0004ms
20.0008ms
20.0012ms
20.0016ms