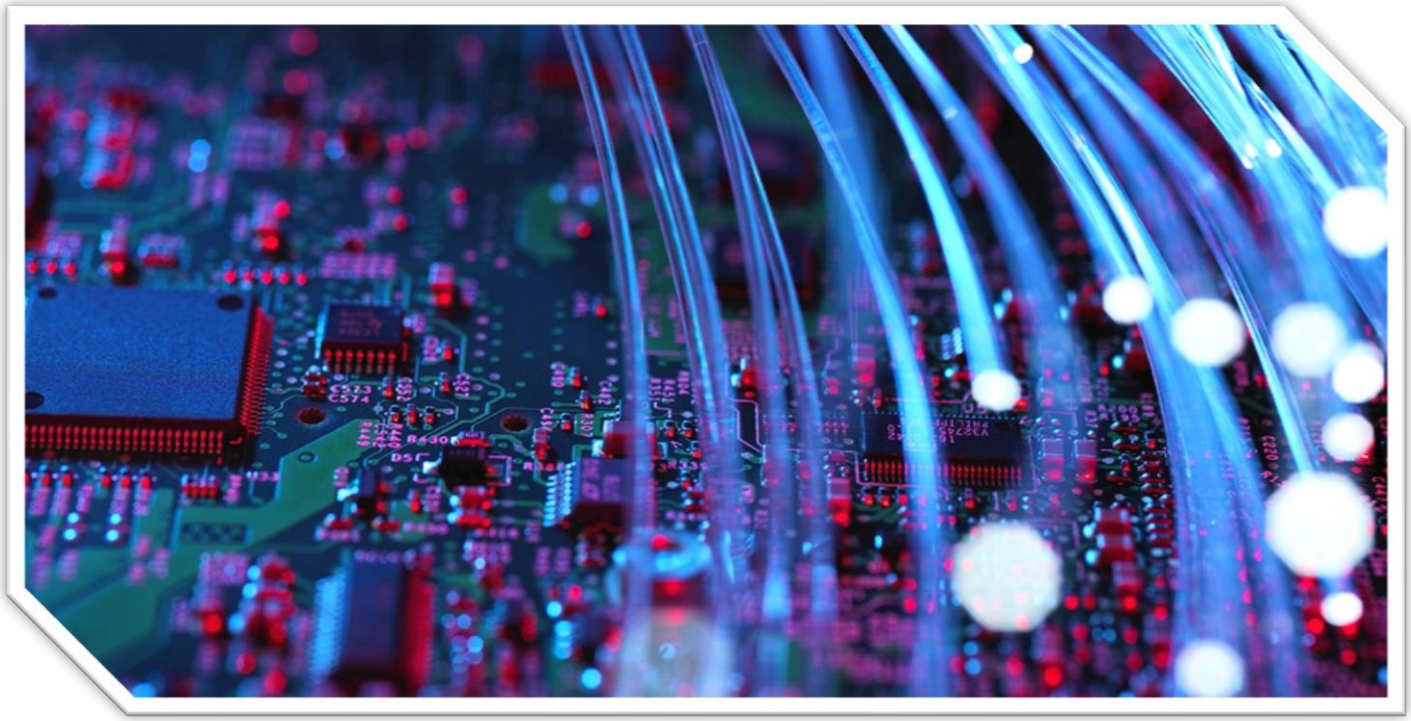
Analog and Digital assignment



[7]

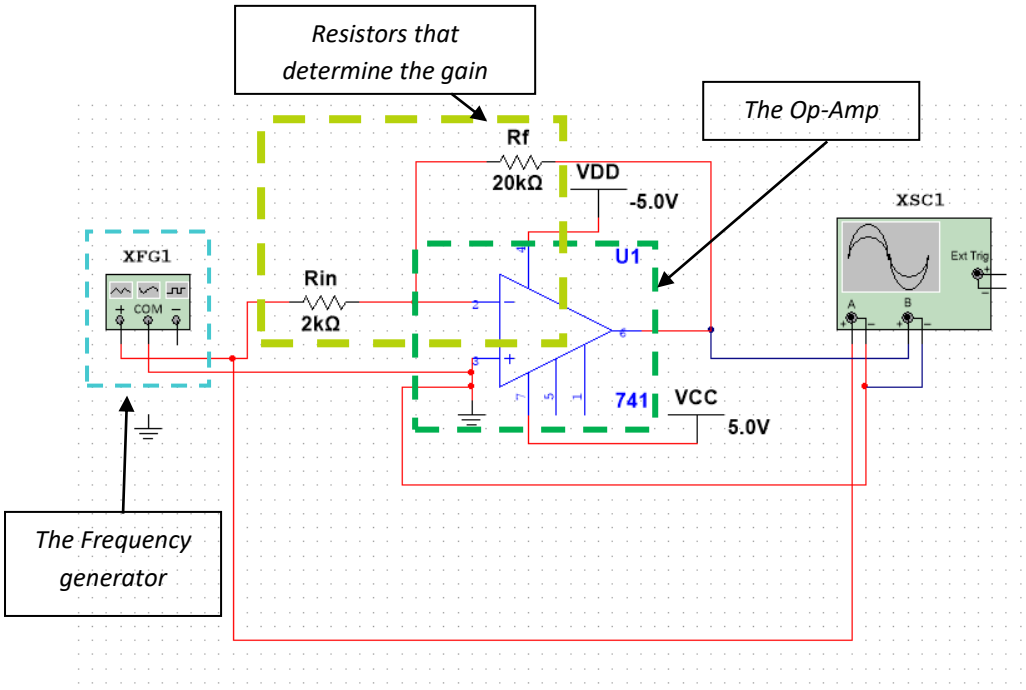
Written by : Ben Edwards

Student ID : 18026826

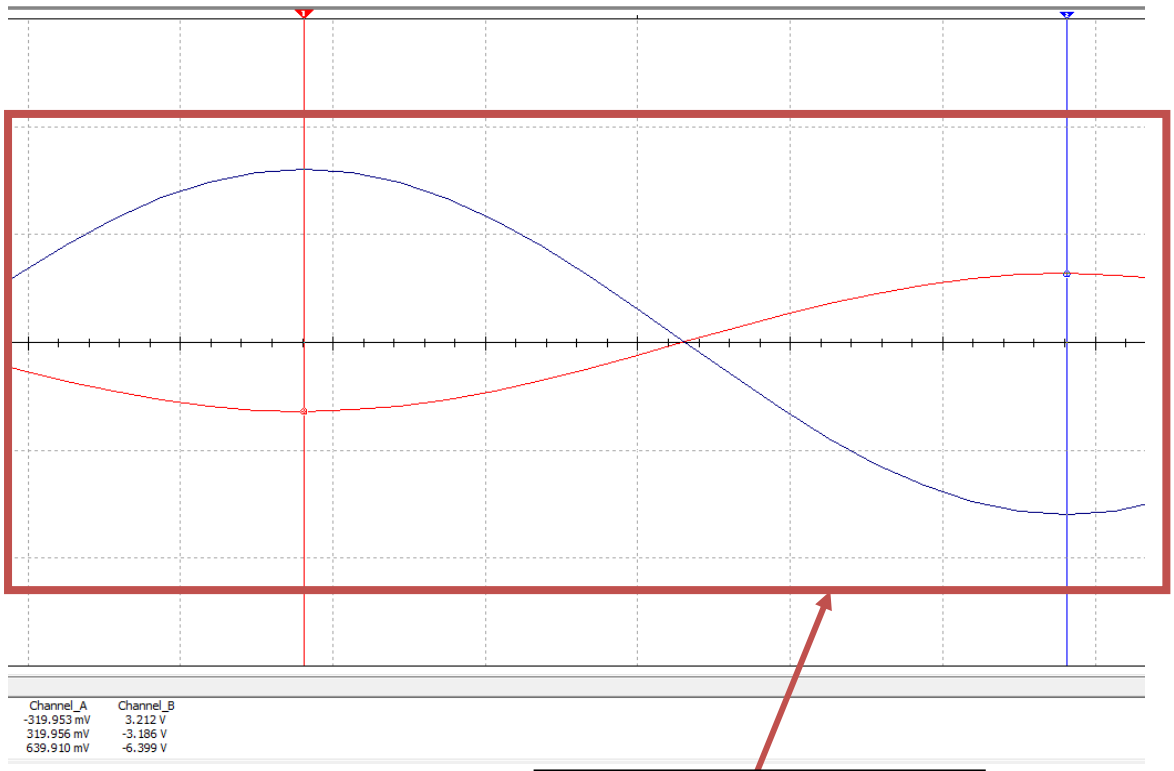
Table Of Contents

Figures 1

Fig(1.1): - Inverting Amp circuit design



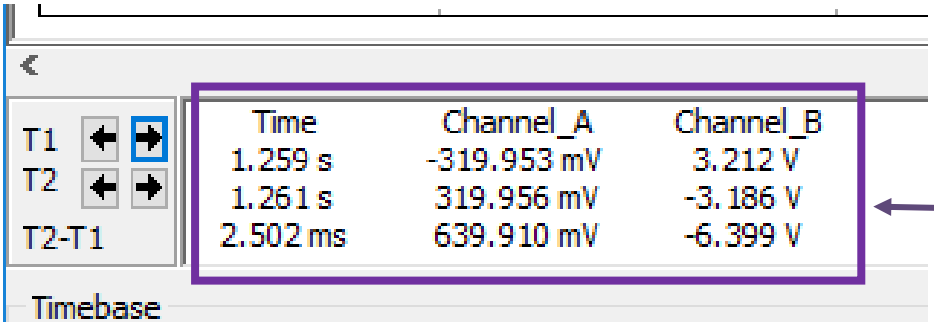
Fig(1.2)



Shows the values of the op-amp output at 320mV input, showing the values of…

	(Vin) Channel A	*(Vout) Channel B*
T1	*-319.95mV*	*3.212V*
T2	*319.956mV*	*-3.186V*
T2-T1	*639.910mV*	*-6.399V*

Fig(1.3)

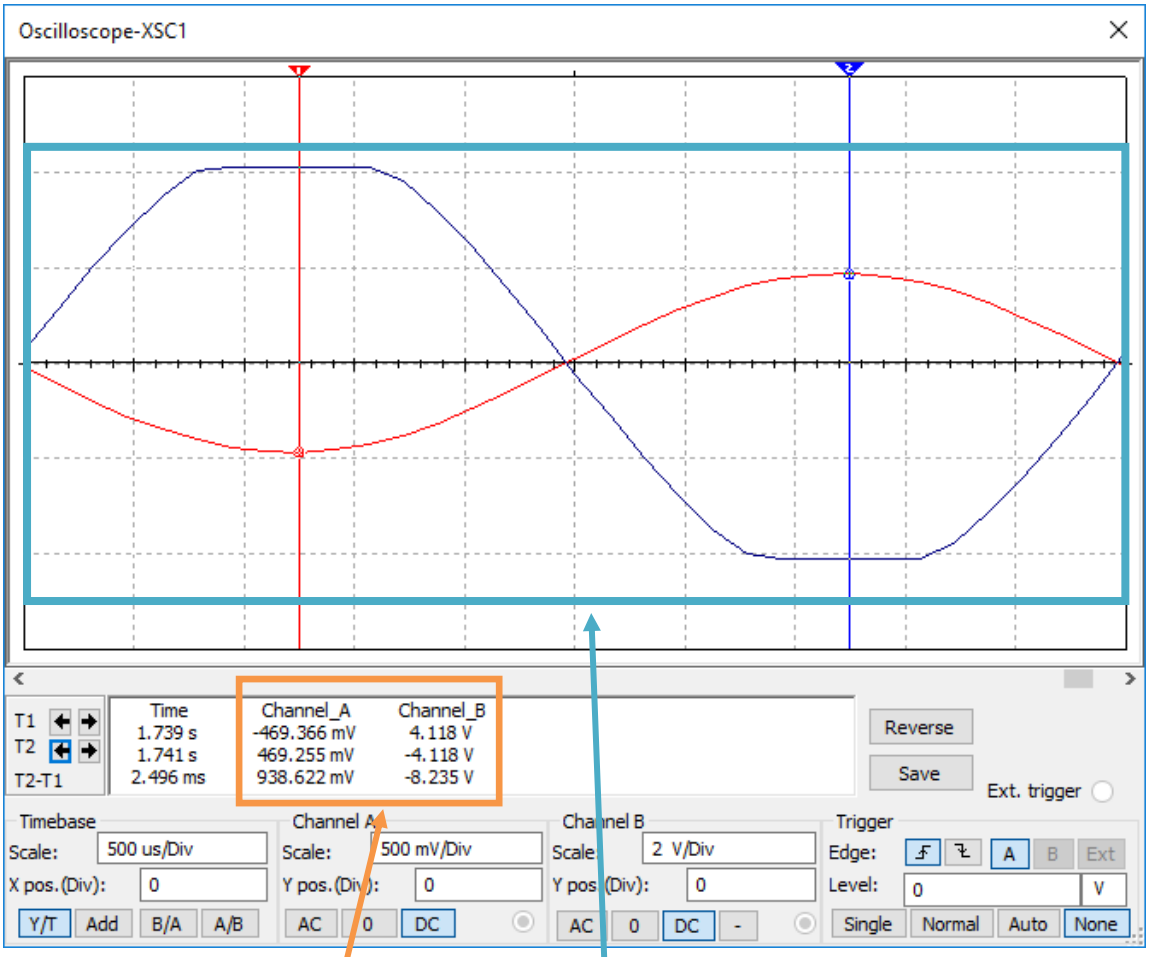


Shows the values of the op-amp output at 320mV input, showing the values of…

	(Vin) Channel A	*(Vout) Channel B*
T1	*-319.95mV*	*3.212V*
T2	*319.956mV*	*-3.186V*
T2-T1	*639.910mV*	*-6.399V*

Fig(1.4)

470mVpk:-

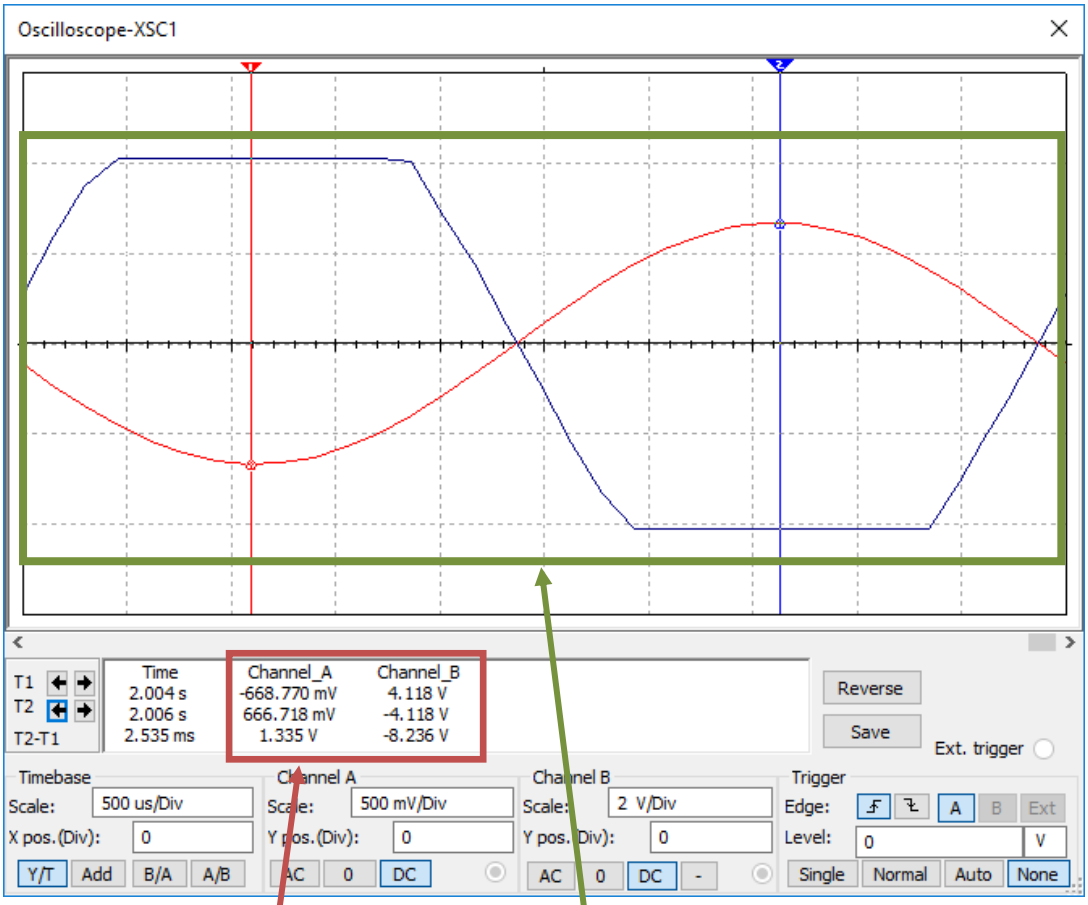


This shows the results and wave form for the 470mVpk input, inverting op-amp.

	(Vin) Channel A	*(Vout) Channel B*
T1	*-469.366mV*	*4.118V*
T2	*469.255mV*	*-4.118V*
T2-T1	*938.622mV*	*-8.235V*

Fig(1.5)

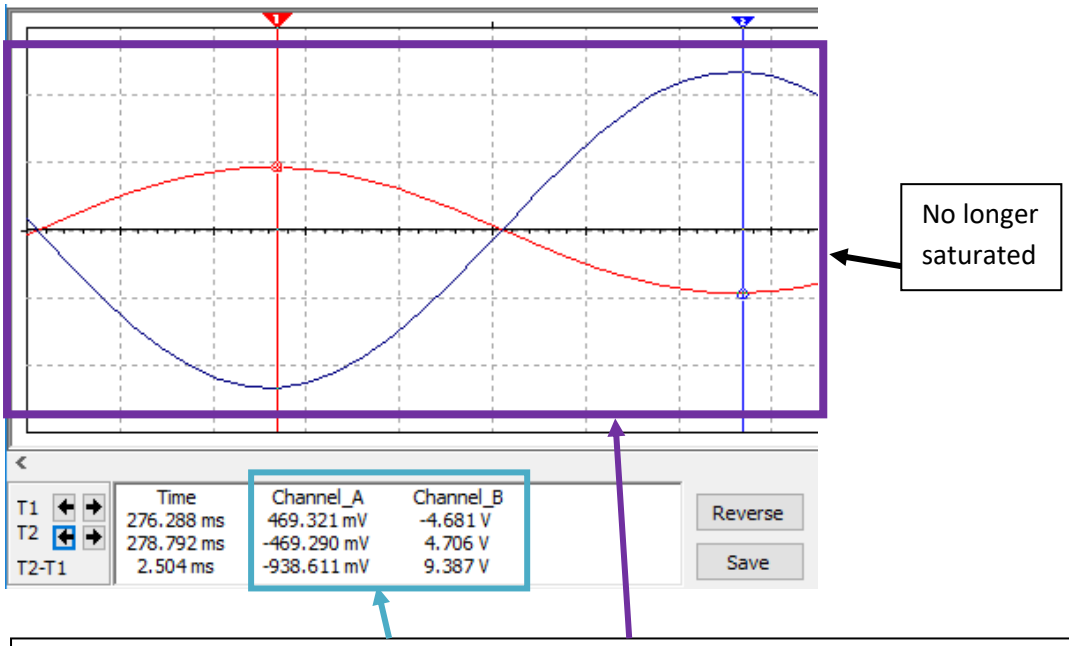
670mVpk:-



This shows the results and waveform for the 670mVpk input, inverting op-amp.

	(Vin) Channel A	*(Vout) Channel B*
T1	*-668.770mV*	*4.118V*
T2	*666.718mV*	*-4.118V*
T2-T1	*1.335V*	*-8.236V*

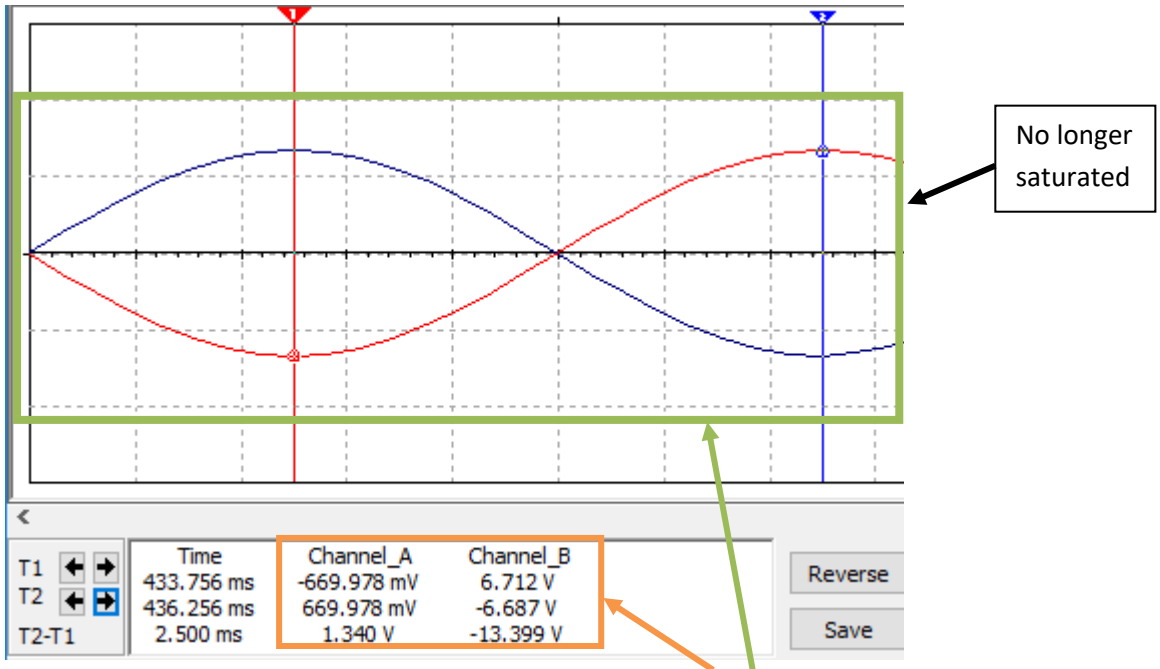
Fig(1.6)



This shows the values and the wave form of the output when the supply is a higher voltage, at the input value of 470mVpk.

	(Vin) Channel A	*(Vout) Channel B*
T1	*469.321mV*	*-4.681V*
T2	*-469.290mV*	*4.706V*
T2-T1	*-938.611mV*	*9.387V*

Fig(1.7)

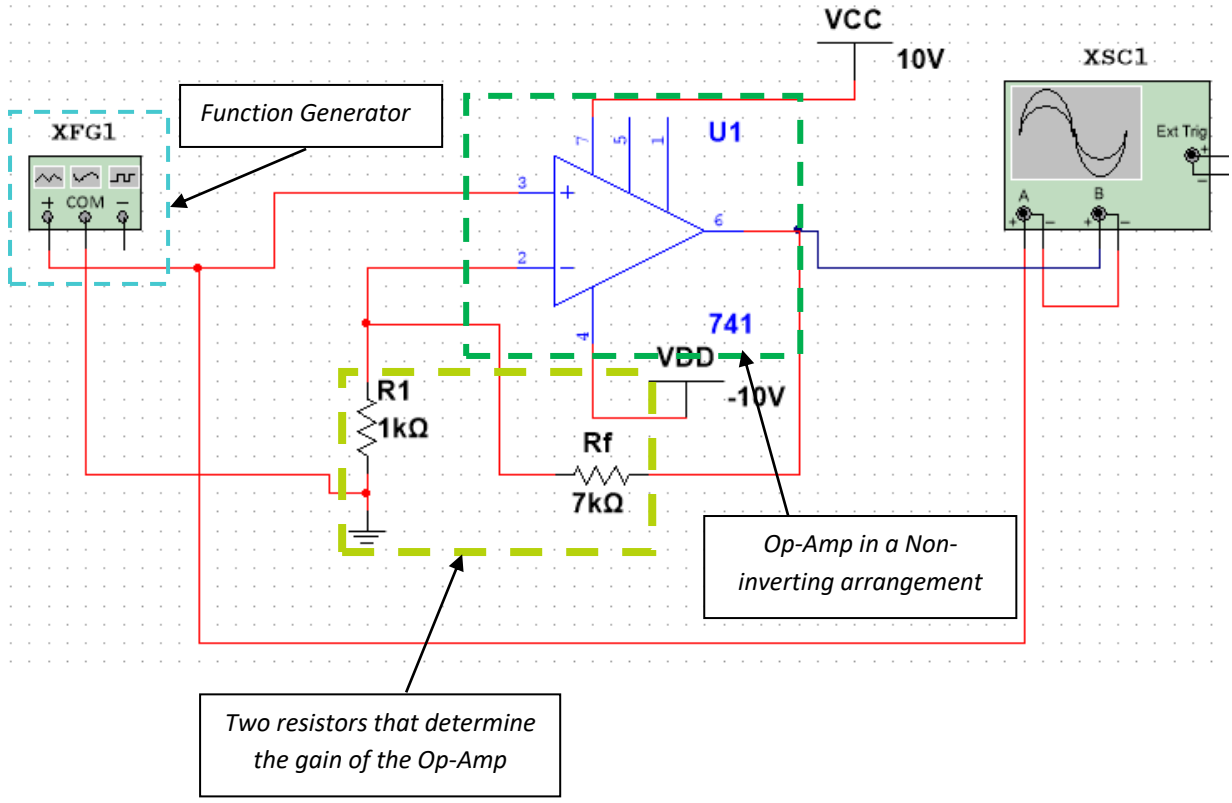


This shows the values and the wave form of the output when the supply is a higher voltage, at the input value of 670mVpk.

	(Vin) Channel A	*(Vout) Channel B*
T1	*-669.978mV*	*6.712V*
T2	*669.978mV*	*-6.687V*
T2-T1	*1.340V*	*-13.399V*

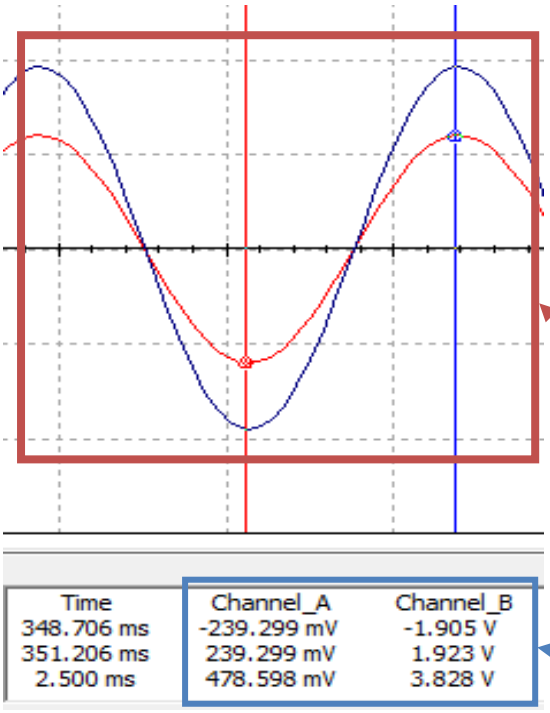
Figures 2

Fig (2.1)



Fig(2.2)

240mVpk: -



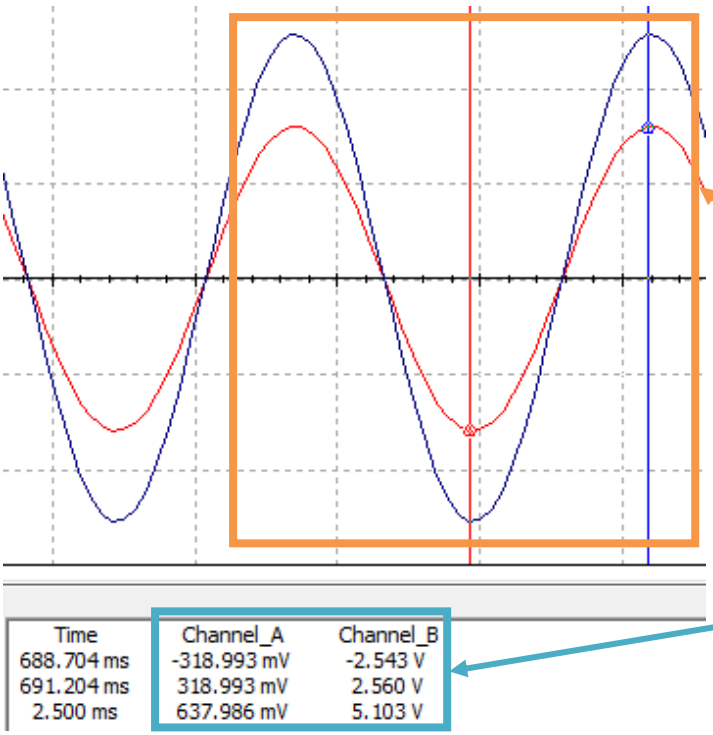
Time	Channel_A	Channel_B
348.706 ms	-239.299 mV	-1.905 V
351.206 ms	239.299 mV	1.923 V
2.500 ms	478.598 mV	3.828 V

This shows the values and wave form for the non-inverting amplifier at 240mVpk

	(Vin) Channel A	*(Vout) Channel B*
T1	*-239.299mV*	*-1.905V*
T2	*239.299mV*	*1.923V*
T2-T1	*478.598mV*	*3.828V*

Fig(2.3)

320mVpk: -

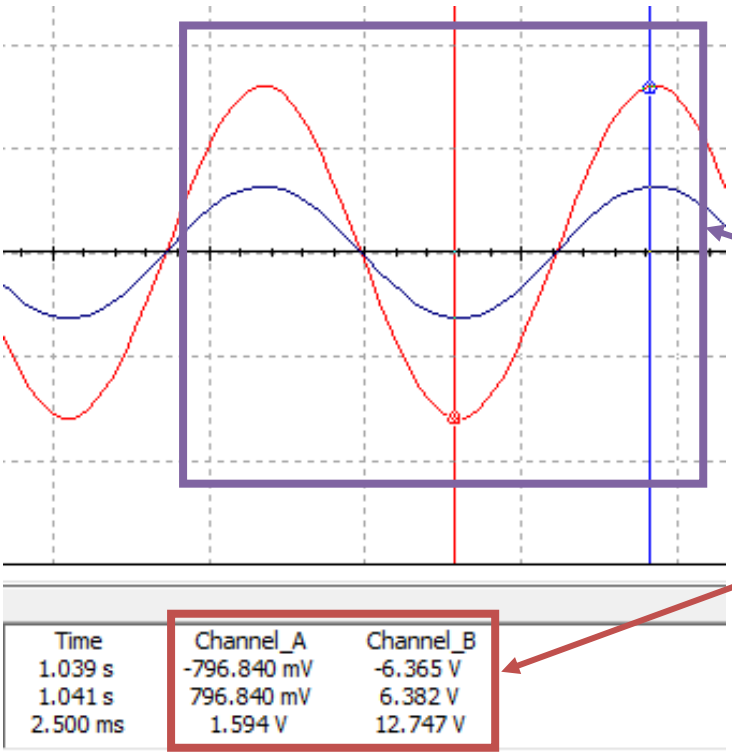


This shows the values and wave form for the non-inverting amplifier at 320mVpk

	(Vin) Channel A	*(Vout) Channel B*
T1	*-318.993mV*	*-2.543V*
T2	*318.993mV*	*2.560V*
T2-T1	*637.986mV*	*5.103V*

Fig(2.4)

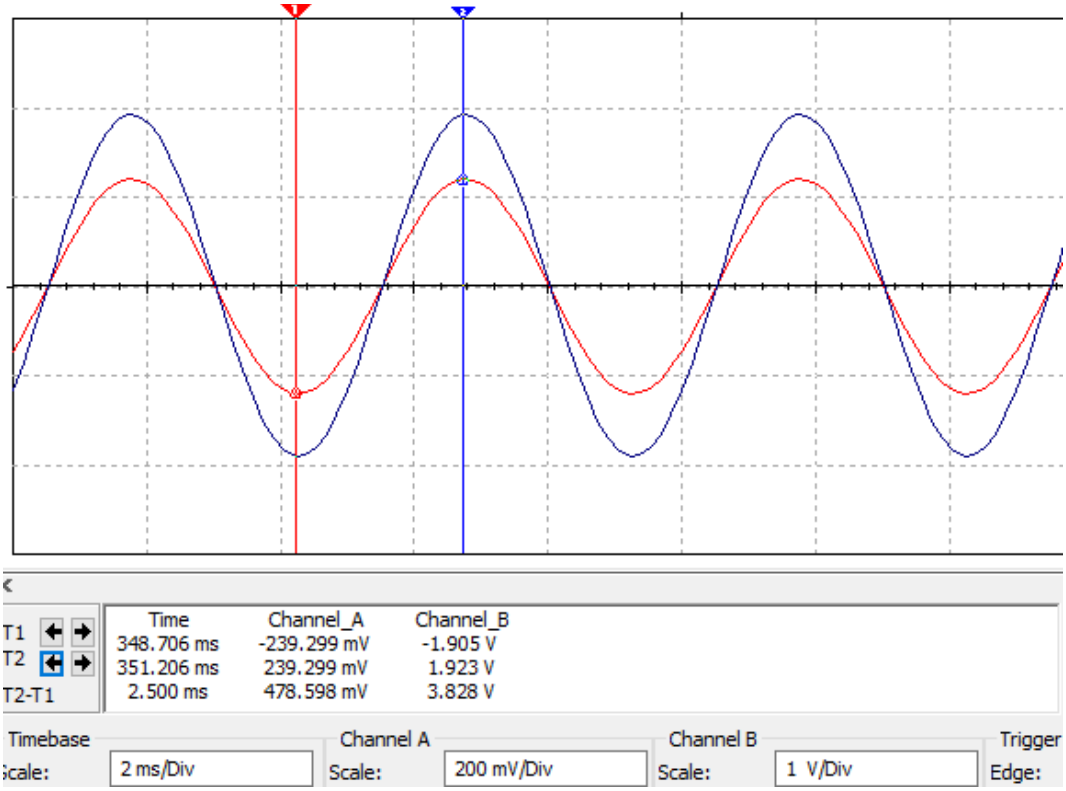
800mVpk: -



This shows the values and wave form for the non-inverting amplifier at 800mVpk

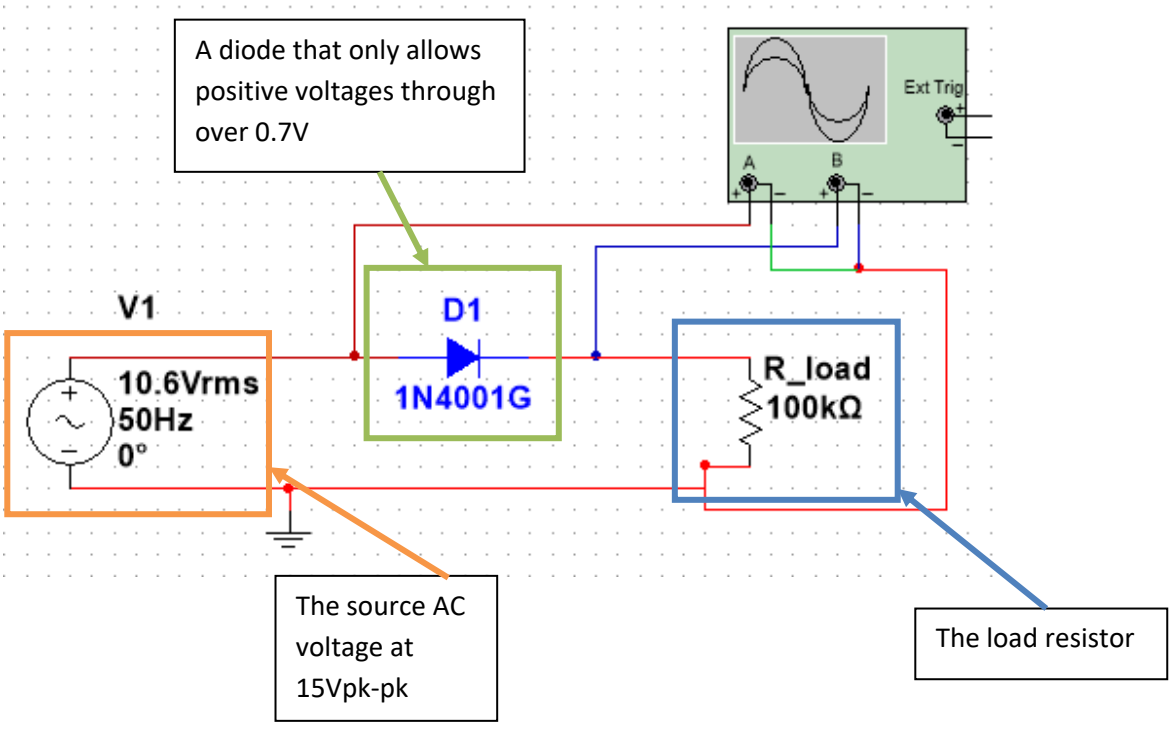
	(Vin) Channel A	*(Vout) Channel B*
T1	*-796.840mV*	*-6.365V*
T2	*796.840mV*	*6.382V*
T2-T1	*1.594V*	*12.747V*

Fig(2.5)

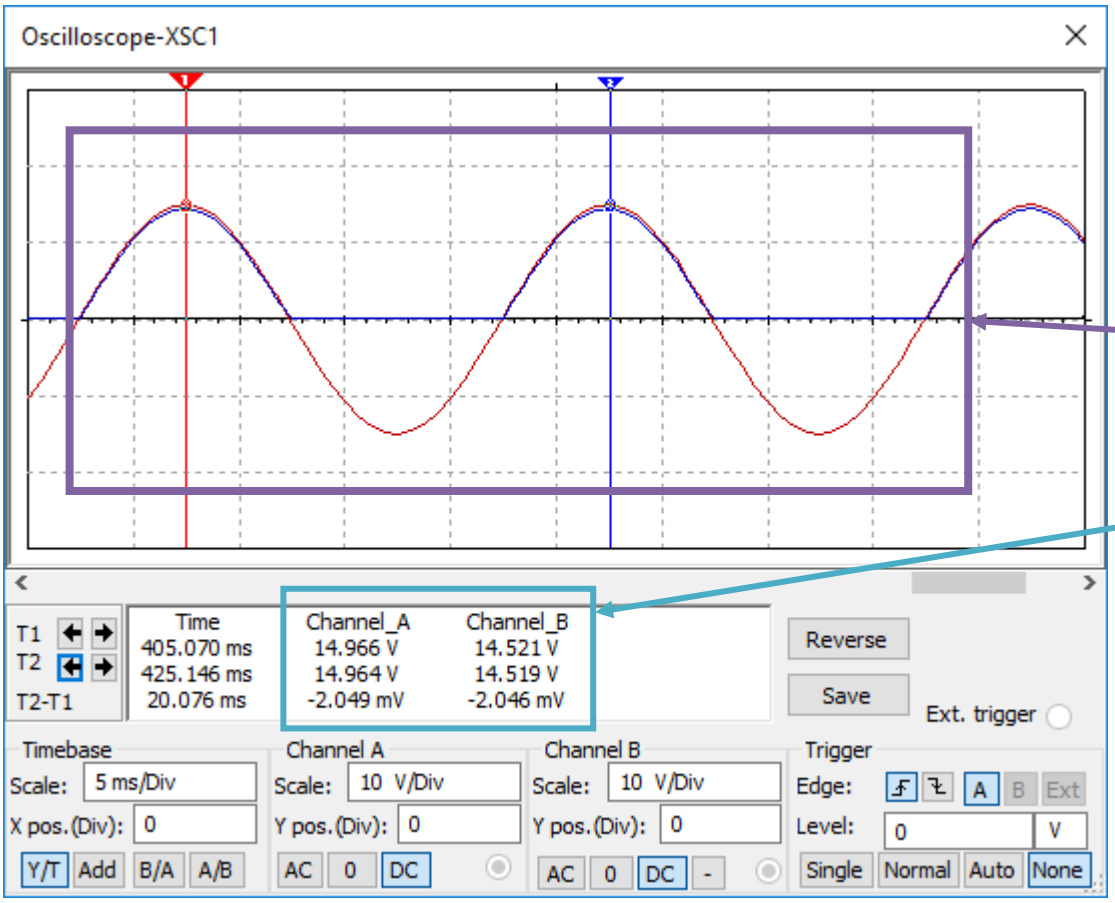


Figures 3

Fig(3.1)



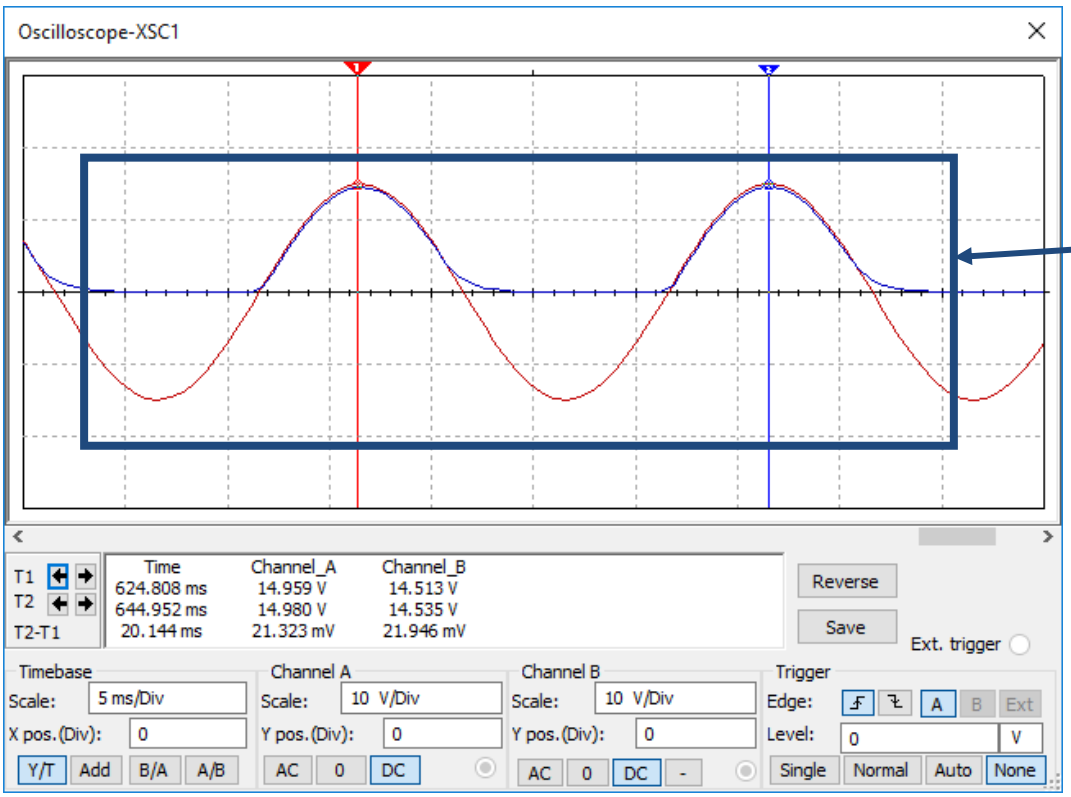
Fig(3.2)



The input and output of the half wave rectifier with a 445mV voltage drop, with no smoothing capacitor.

Fig(3.3)

With a smoothing capacitor of 10nF



10nF capacitor across the load

Fig (3.4)

100nF

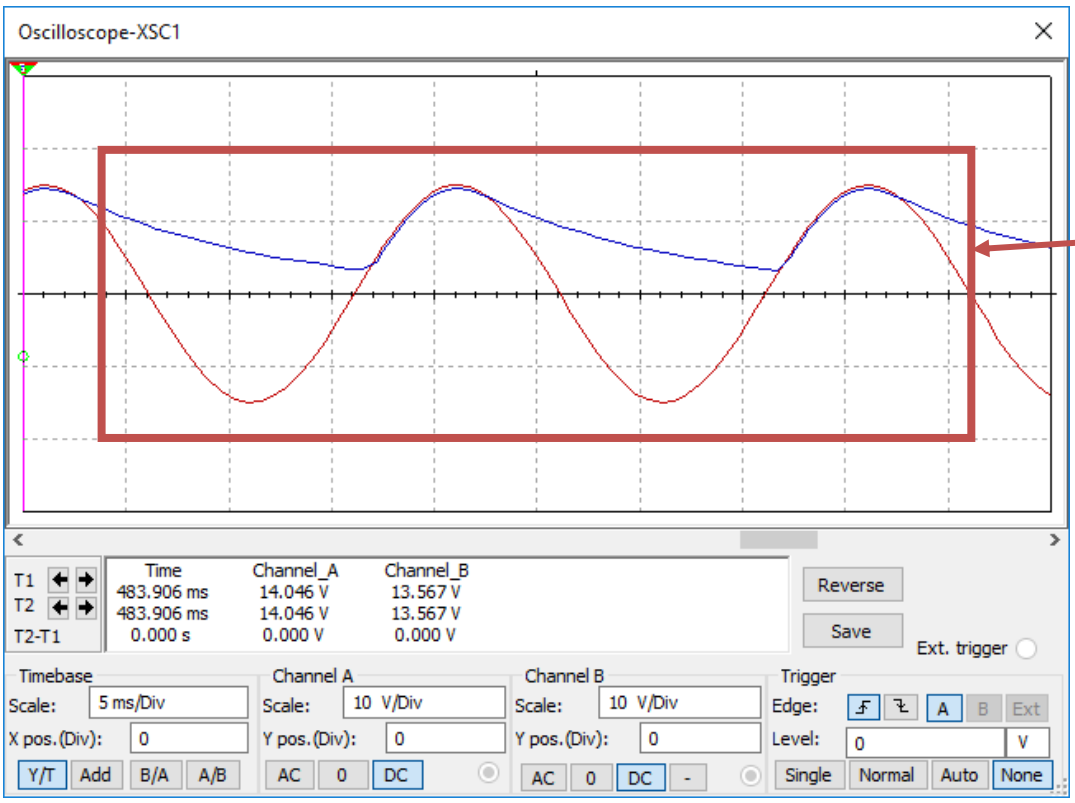


Fig (3.5)

500nF

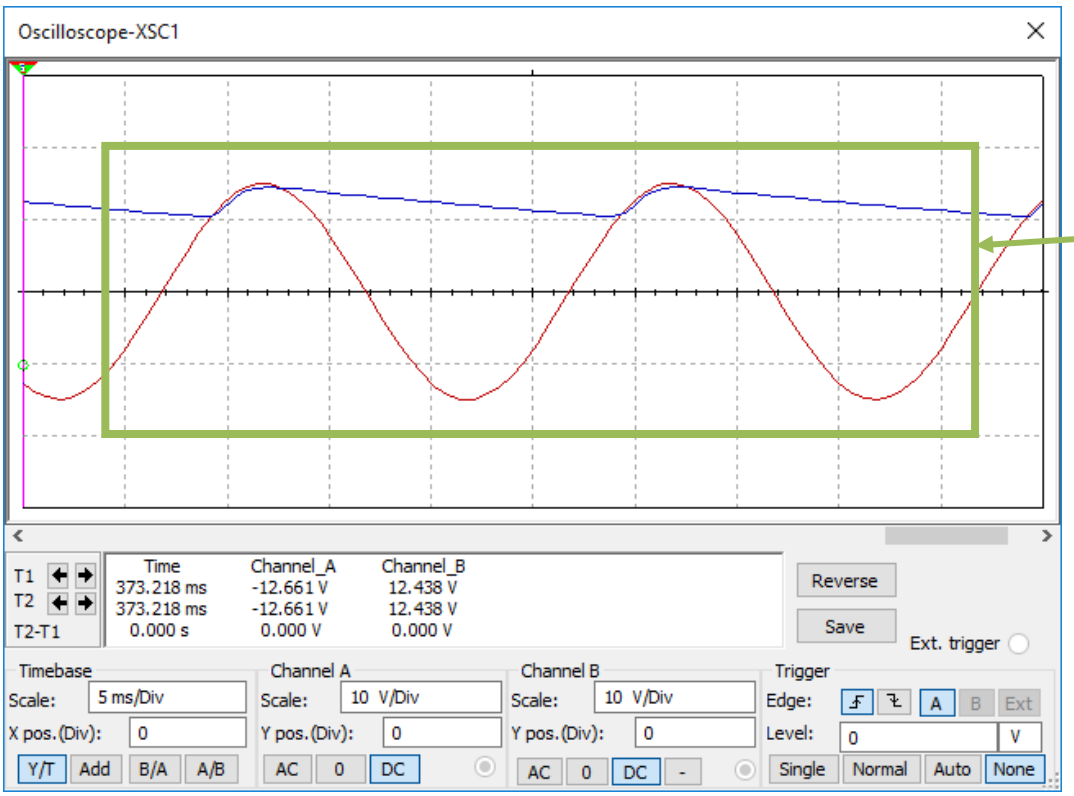
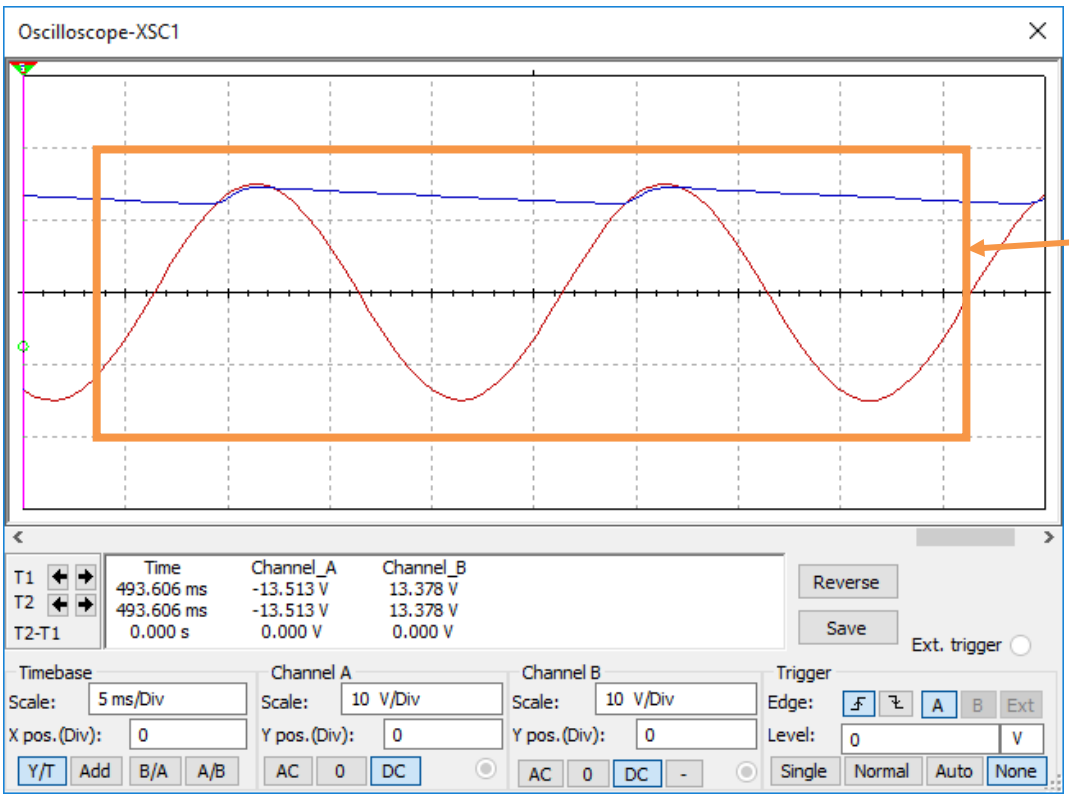


Fig (3.6)

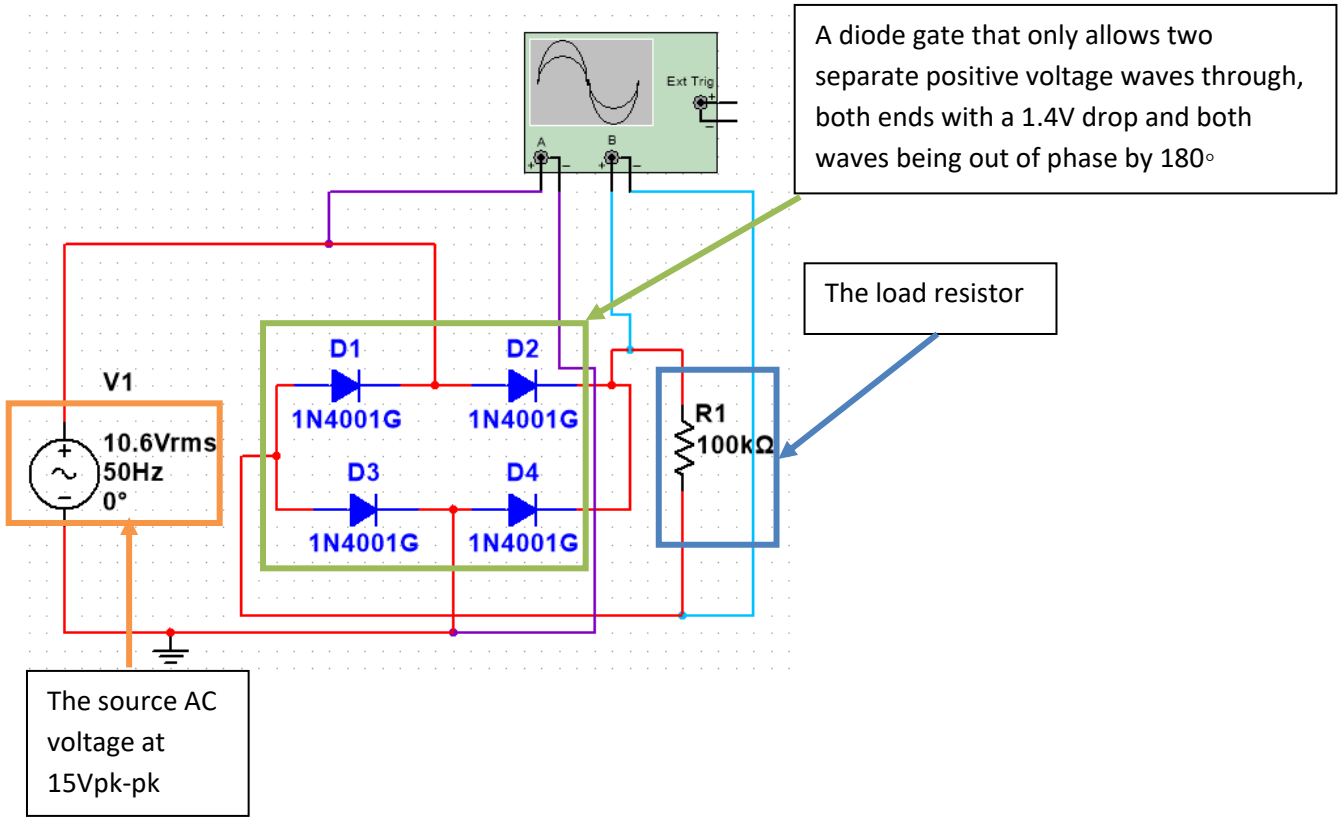
1uF



1uF capacitor across the load

Figures 4

Fig (4.1)



A diode gate that only allows two separate positive voltage waves through, both ends with a 1.4V drop and both waves being out of phase by 180◦

The load resistor

The source AC voltage at 15Vpk-pk

Fig (4.2)

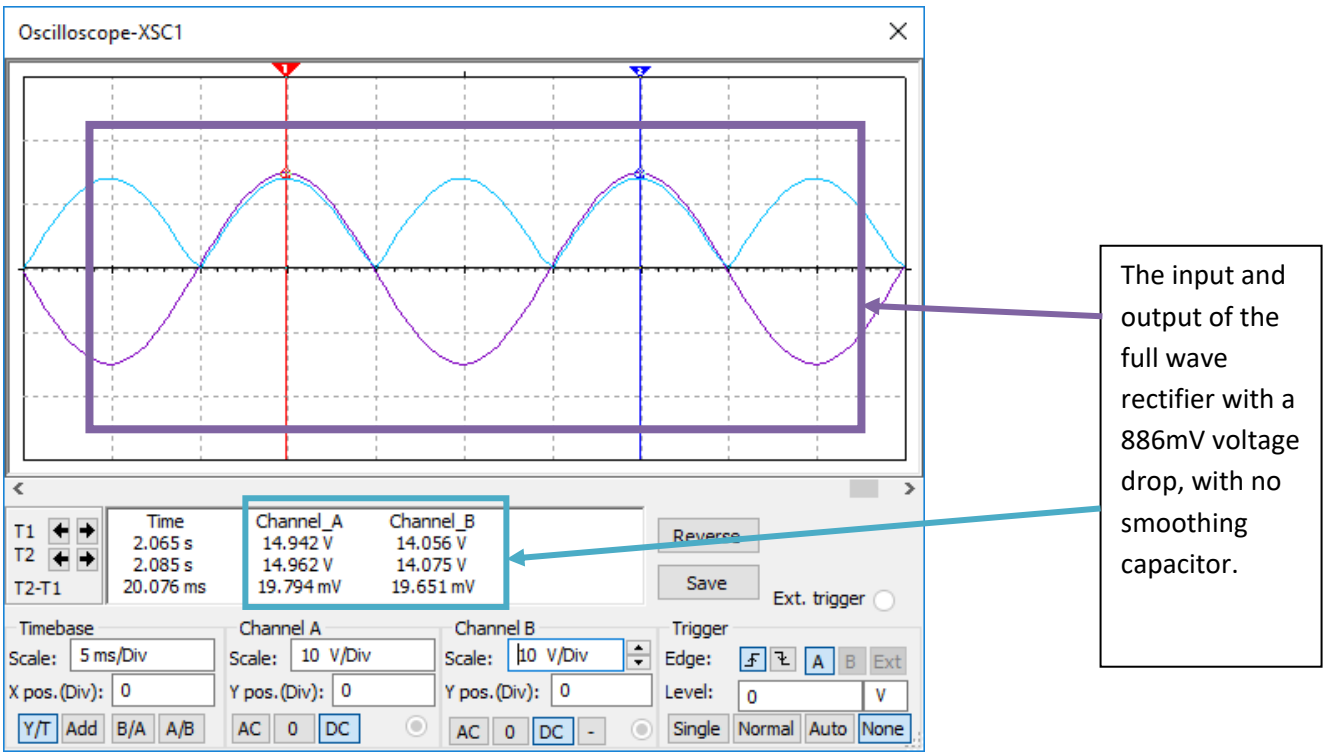


Fig (4.3)

10nF capacitor connected across resistor load.

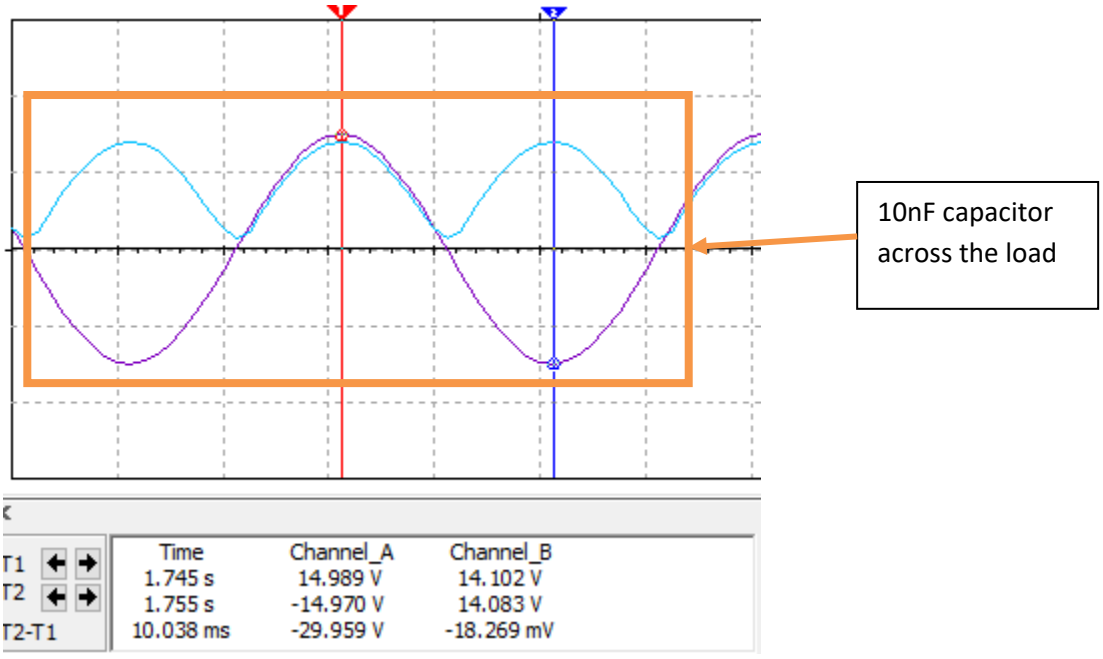


Fig (4.4)

100nF capacitor connected across resistor load.

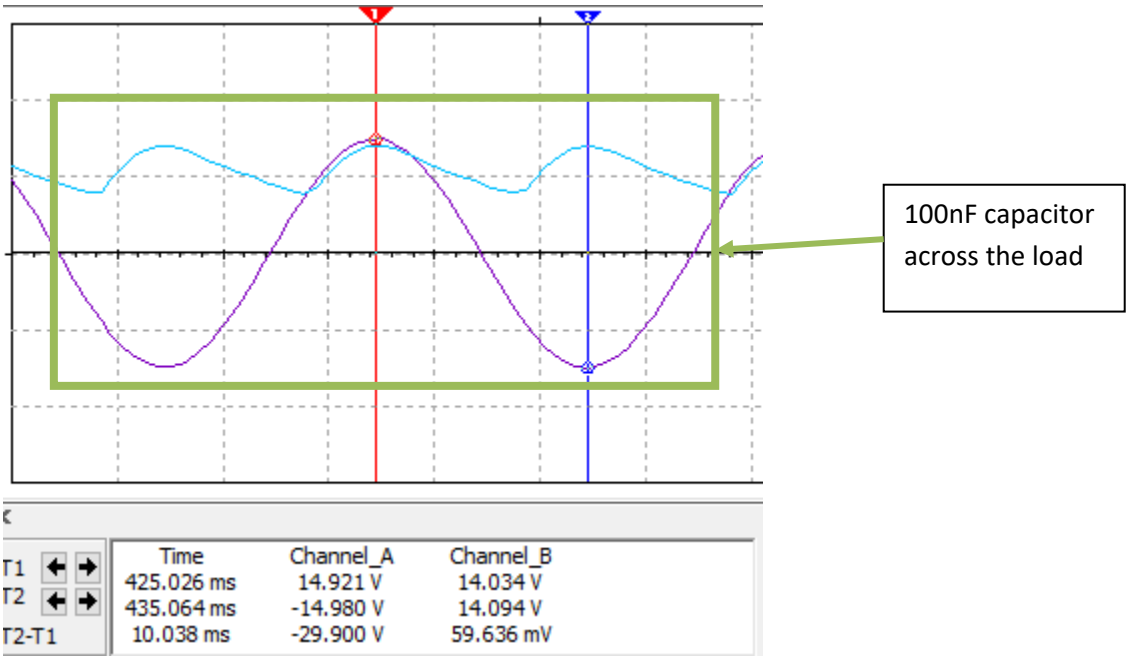


Fig (4.5)

500nF capacitor connected across resistor load.

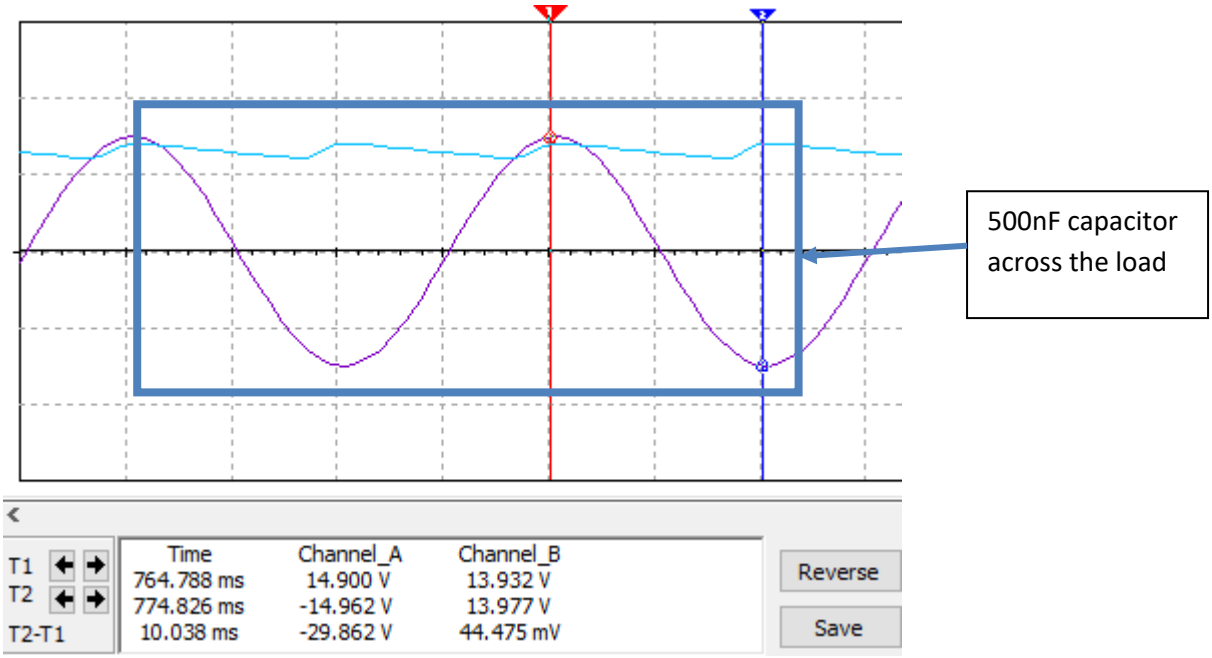
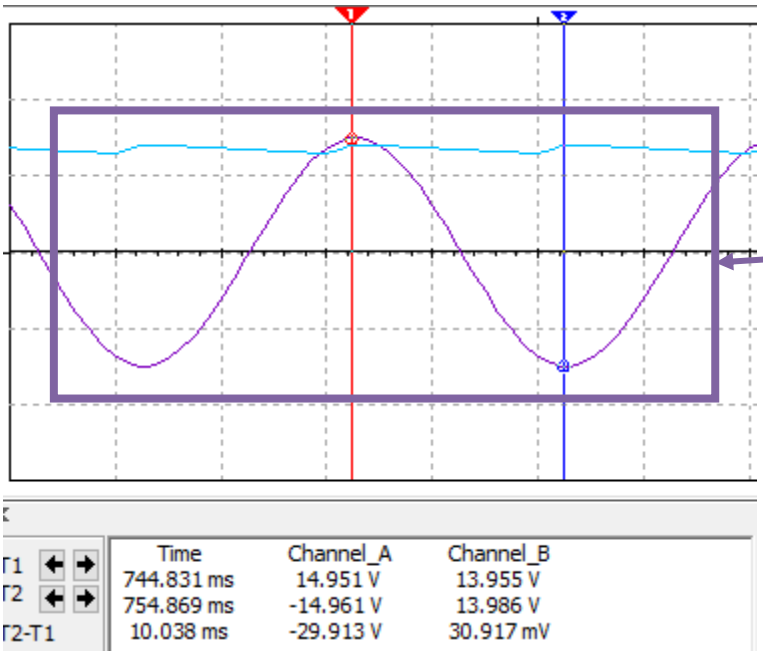


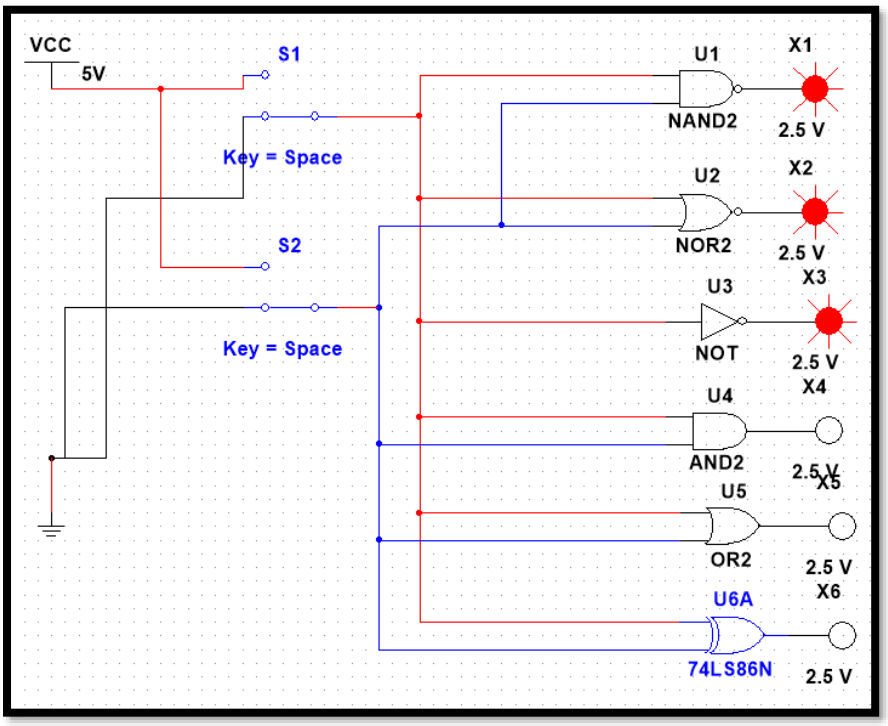
Fig (4.6)

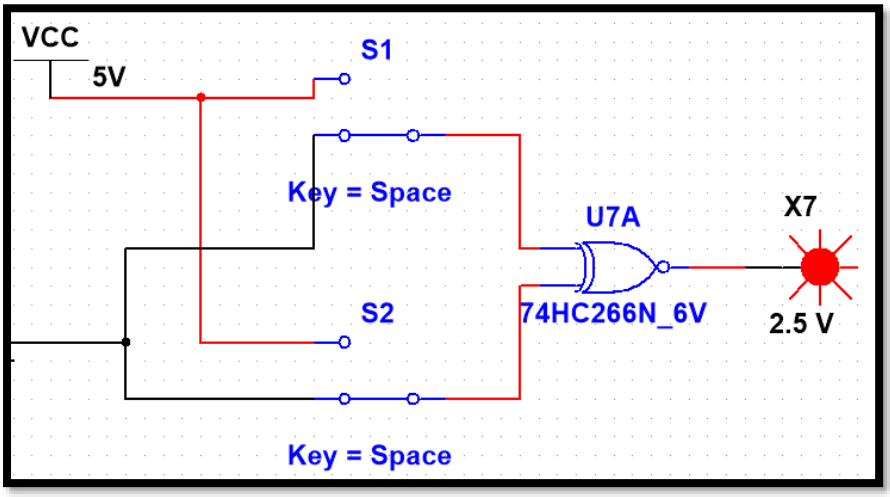
1uF capacitor connected across resistor load.



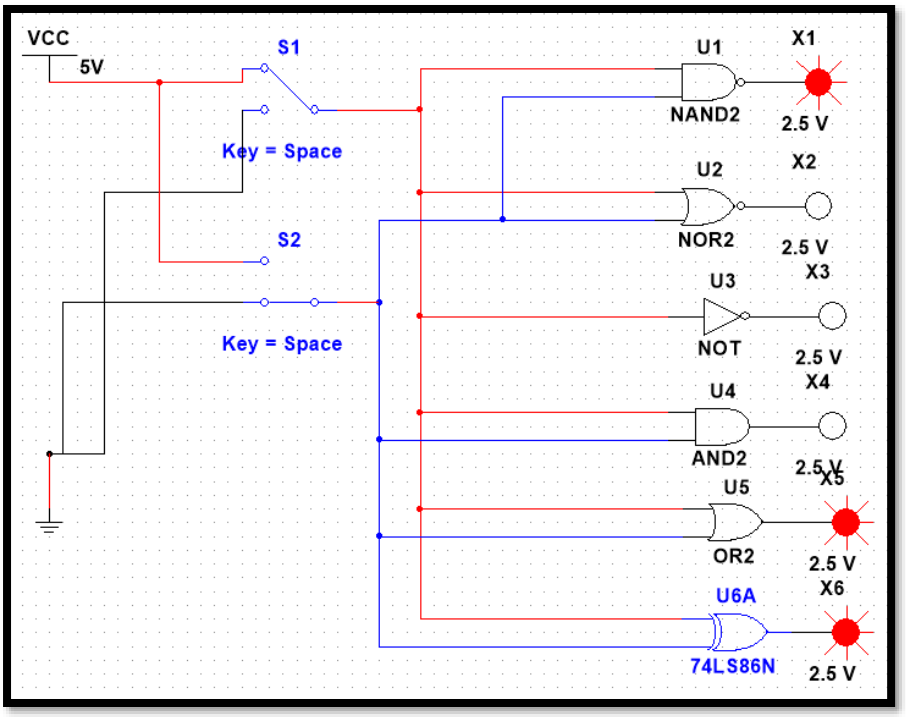
Figures 5

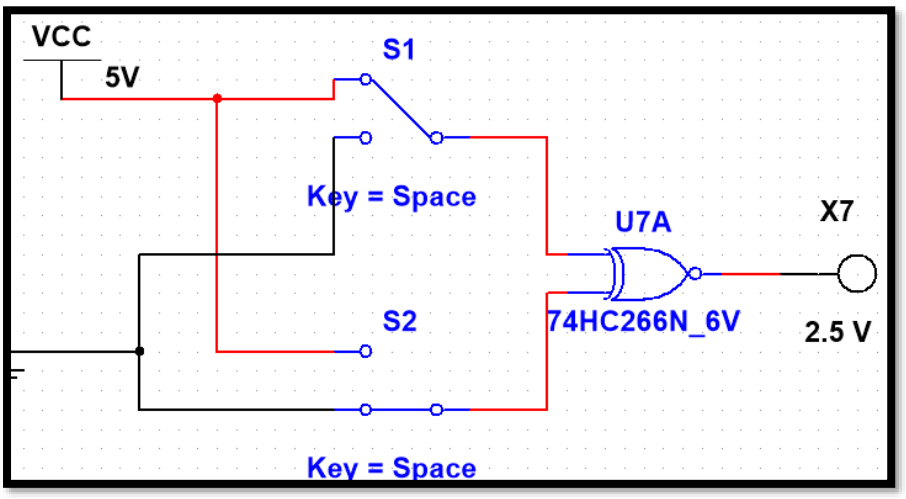
Fig(5.1)
Input A = 0
Input B = 0



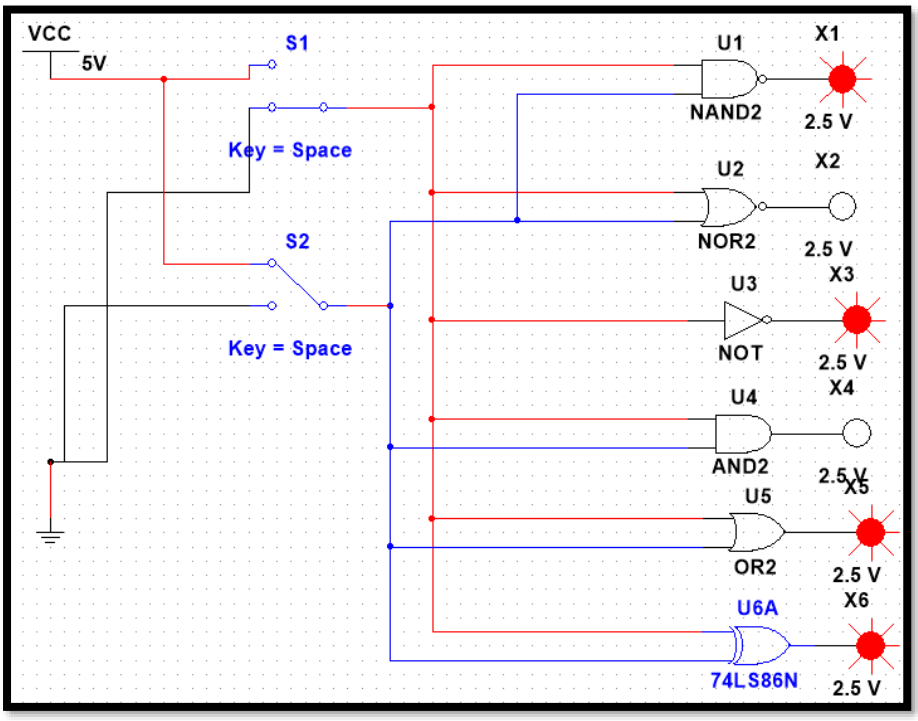


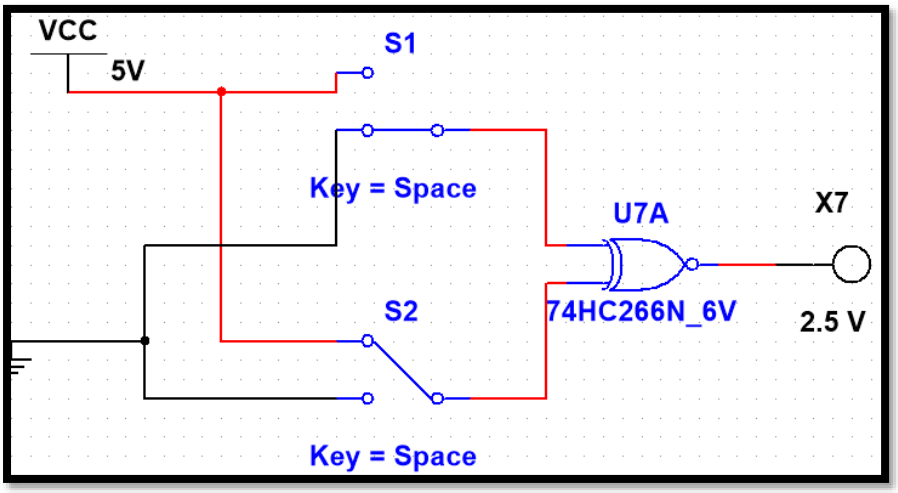
Fig(5.2)
Input A = 1
Input B = 0



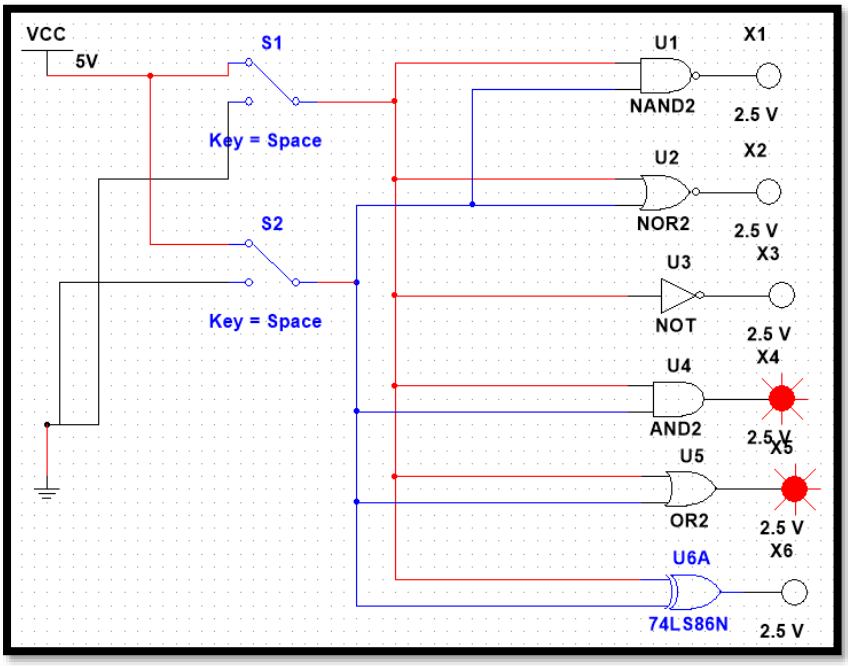


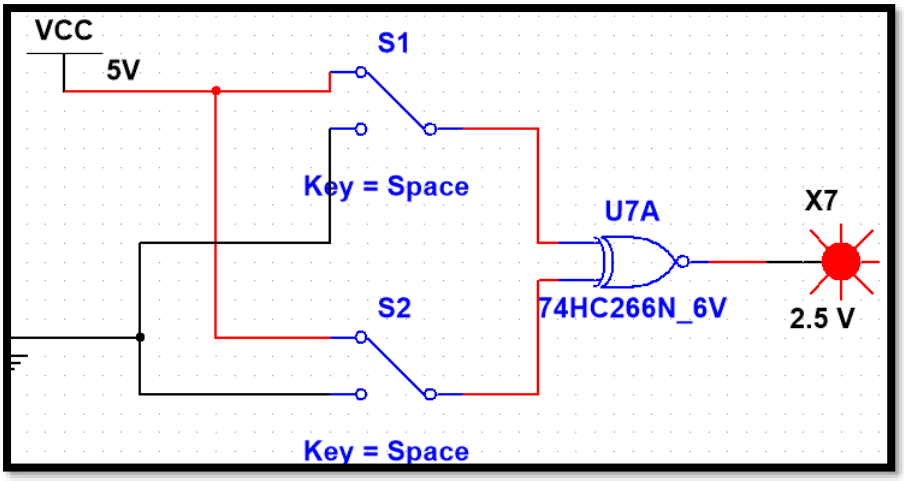
Fig(5.3)
Input A = 0
Input B = 1





Fig(5.4)
Input A = 1
Input B = 1





Fig(5.5)

OR Gate truth table [1]

Inputs		Output
B	A	Q
0	0	0
0	1	1
1	0	1
1	1	1

Fig(5.6)

NAND Gate truth table [1]

Inputs		Output
B	A	Q
0	0	1
0	1	1
1	0	1
1	1	0

Fig(5.7)

NOR Gate truth table [1]

Inputs		Output
B	A	Q
0	0	1
0	1	0
1	0	0
1	1	0

Fig(5.8)

EXOR Gate truth table [1]

Inputs		Output
B	A	Q
0	0	0
0	1	1
1	0	1
1	1	0

Fig(5.9)

NOT Gate truth table [1]

Input	Output
A	Q
0	1
1	0

Fig(5.10)

AND Gate truth table [1]

Inputs		Output
B	A	Q
0	0	0
0	1	0
1	0	0
1	1	1

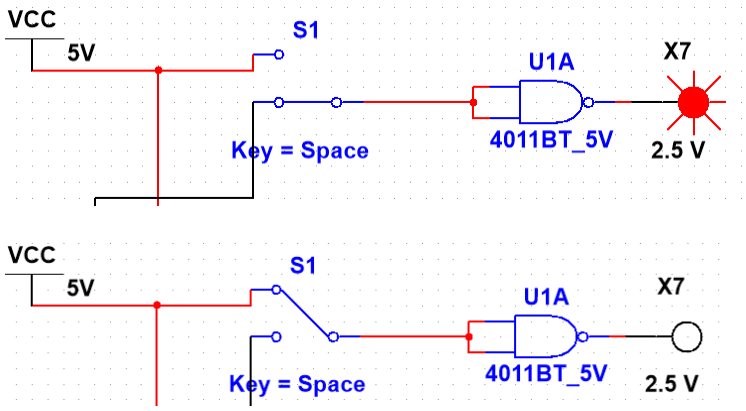
Fig(5.11)

EXNOR Gate truth table [1]

Inputs		Output
B	A	Q
0	0	1
0	1	0
1	0	0
1	1	1

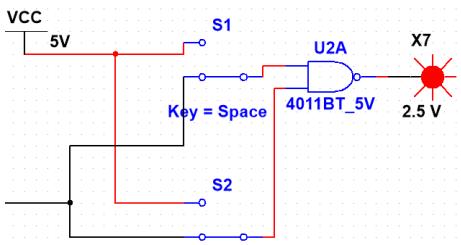
Fig(5.12)

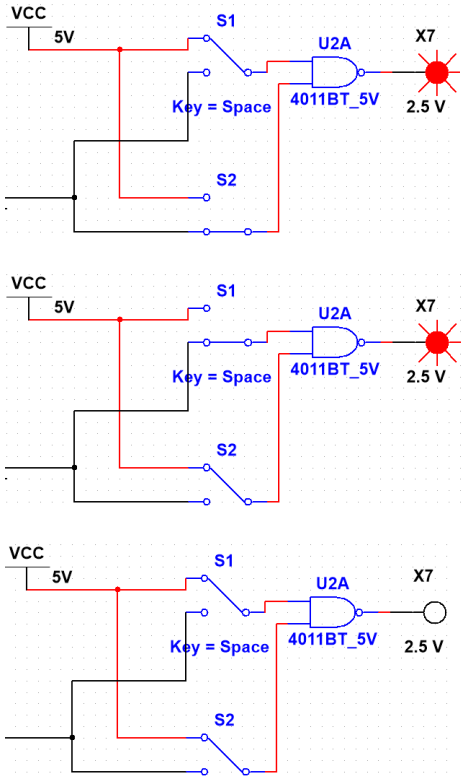
NOT gate equivalent [3]



Fig(5.13)

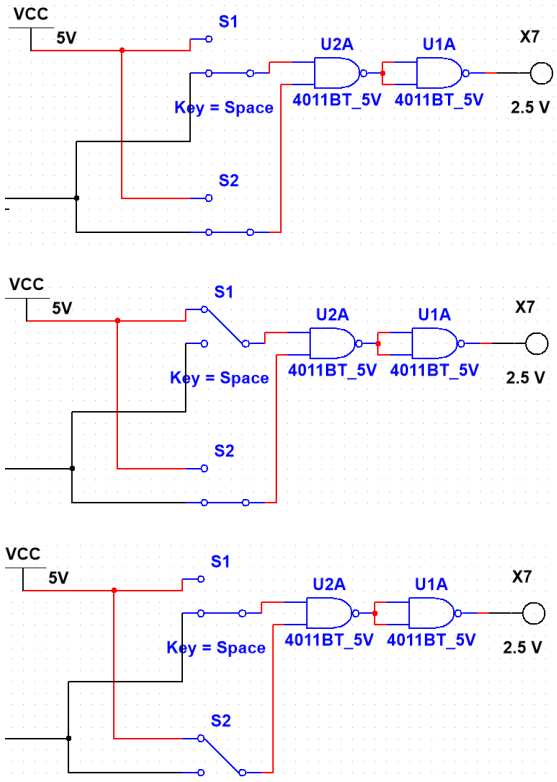
NAND gate equivalent [3]

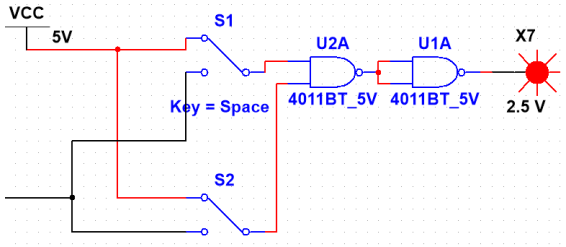




Fig(5.14)

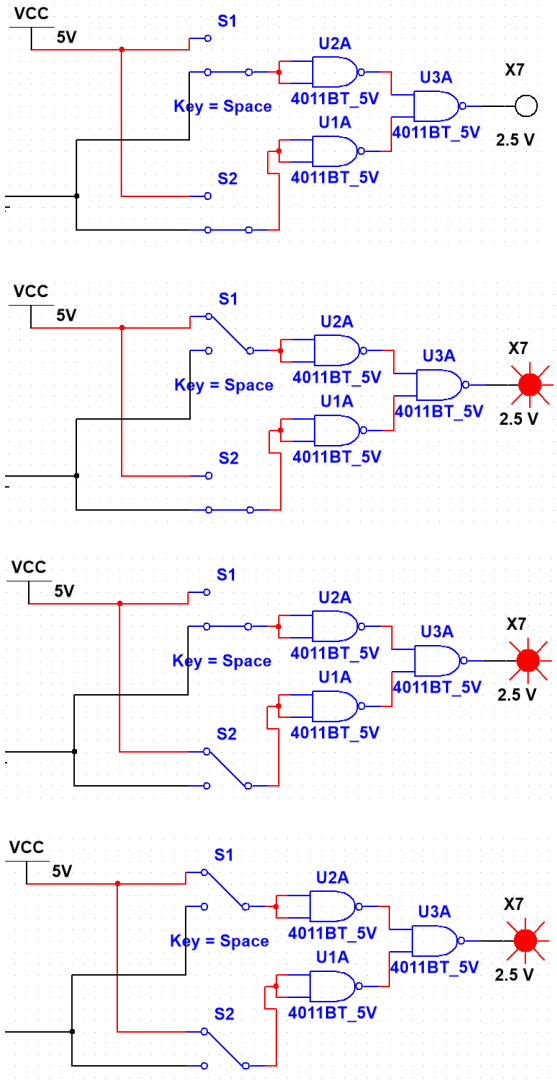
AND gate equivalent [3]





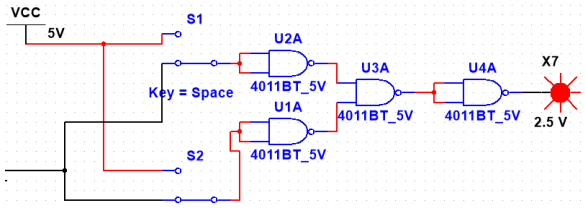
Fig(5.15)

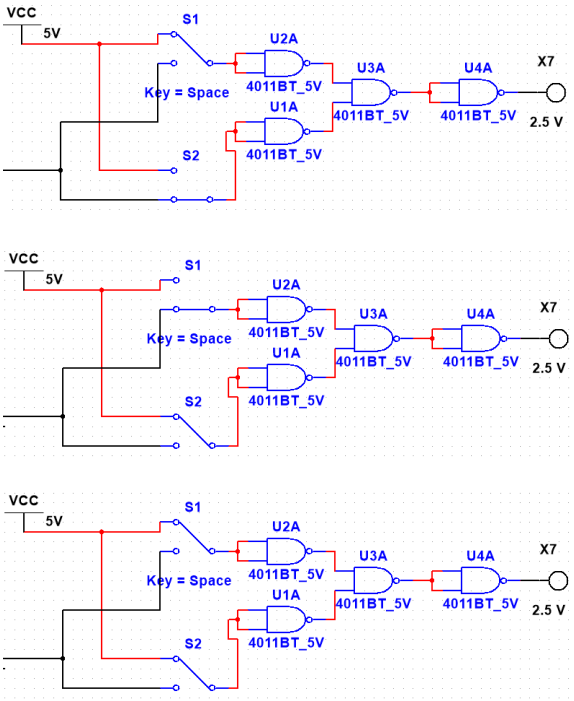
OR gate equivalent [3]



Fig(5.16)

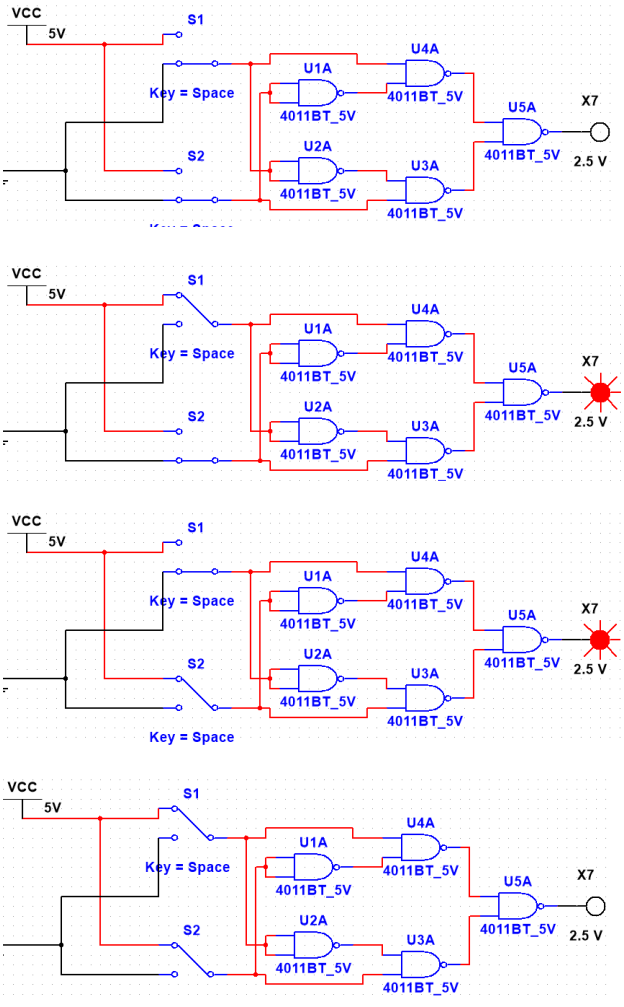
NOR gate equivalent [3]





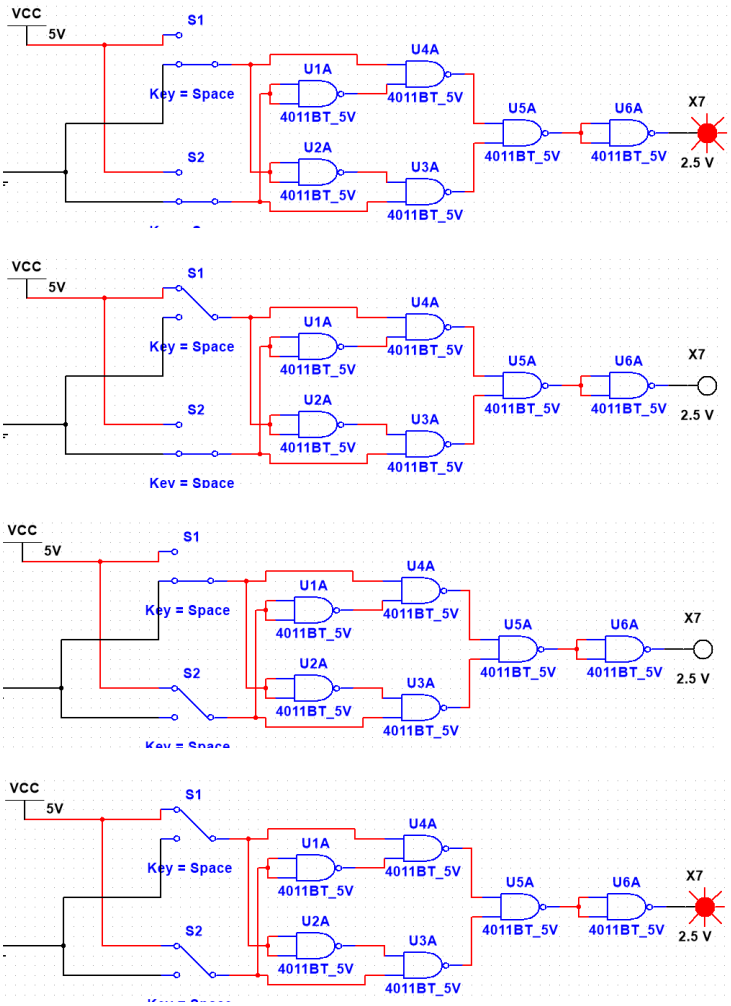
Fig(5.17)

EXOR gate equivalent [3]



Fig(5.18)

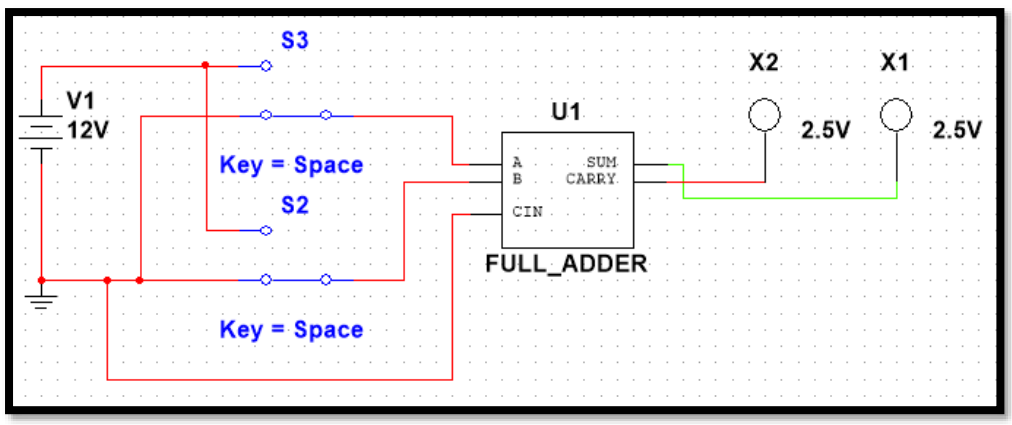
EXNOR gate equivalent [3]



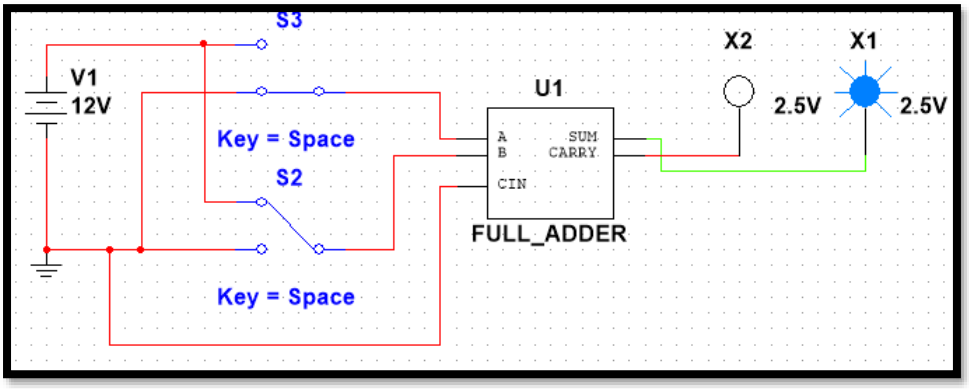
Figures 6

Fig(6.1.1)

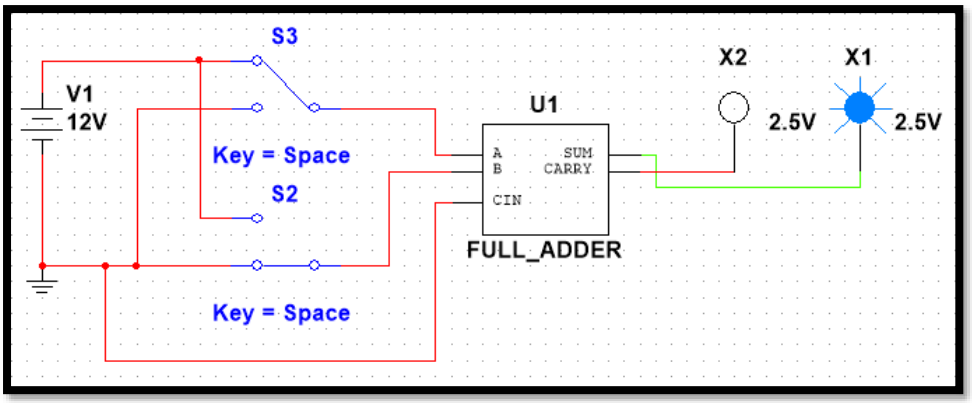
Single bit full adder



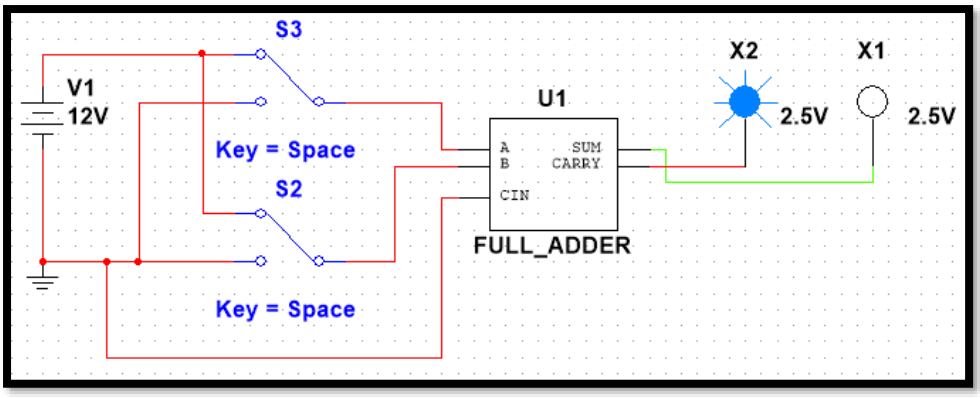
Fig(6.1.2)



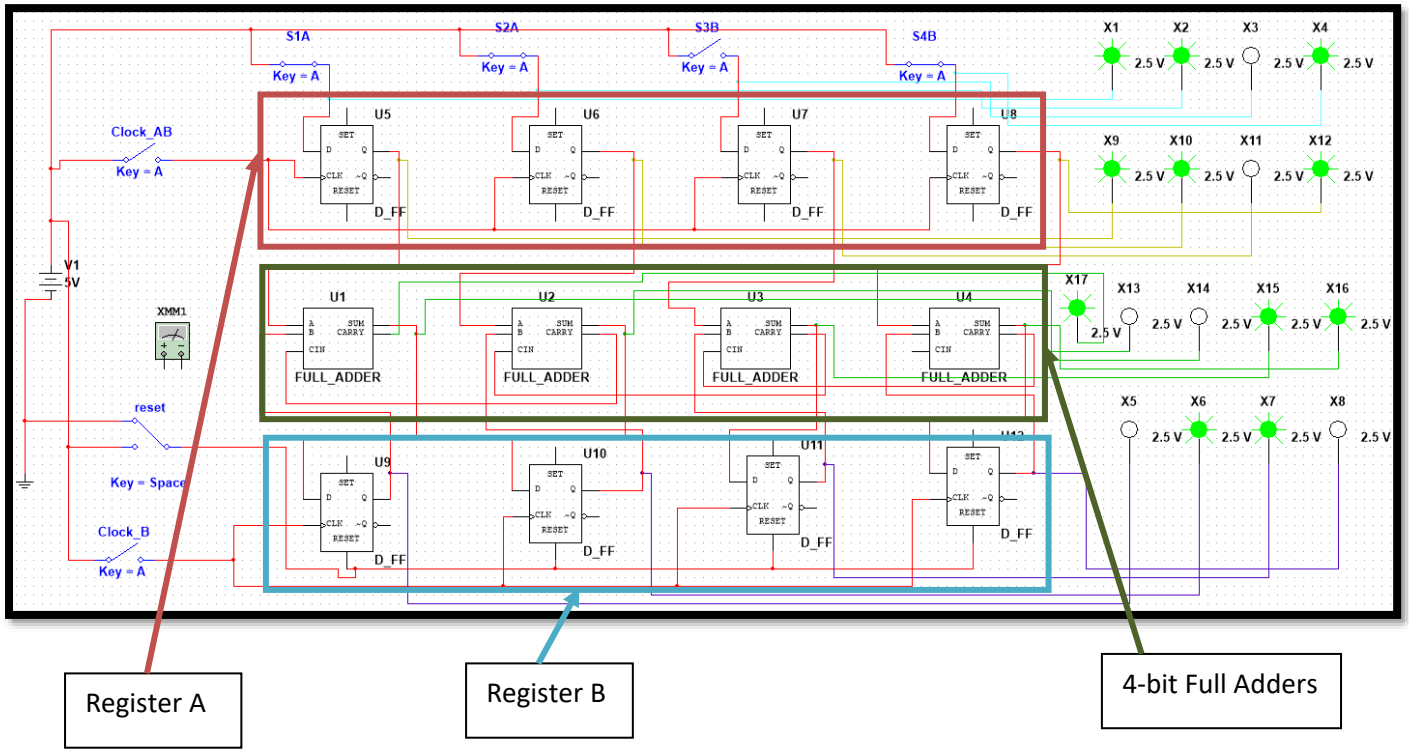
Fig(6.1.3)



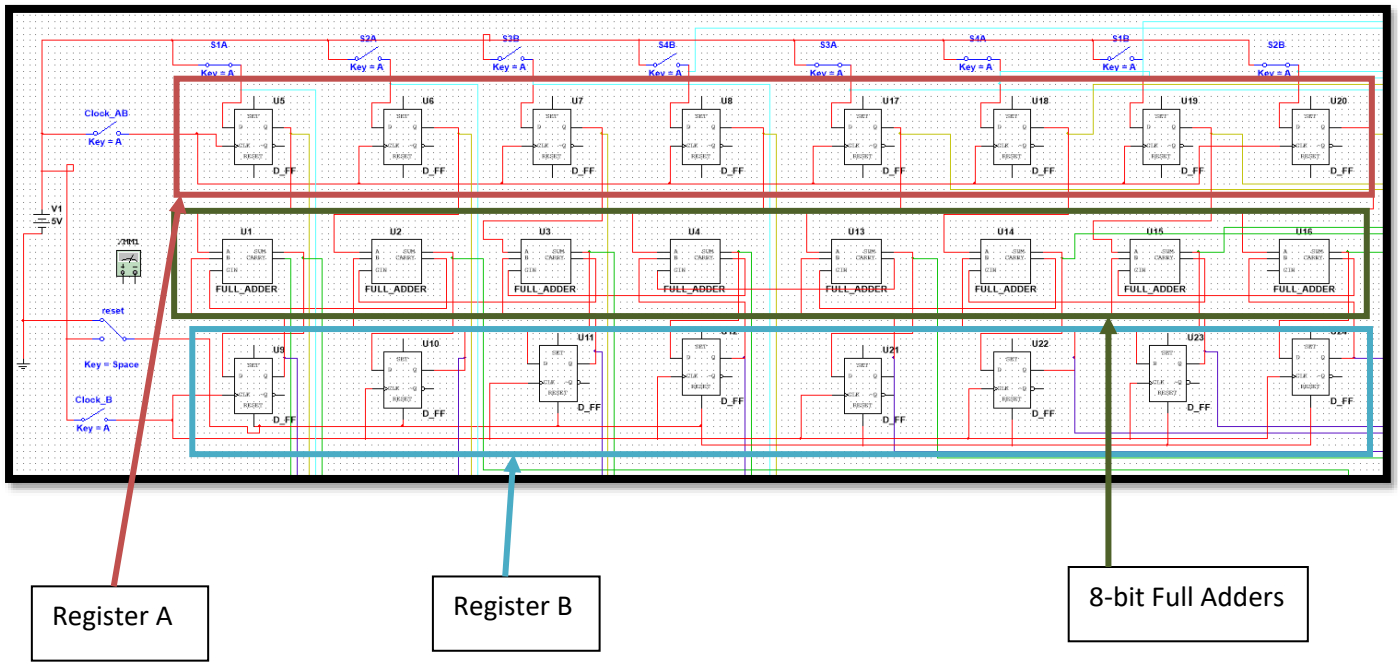
Fig(6.1.4)



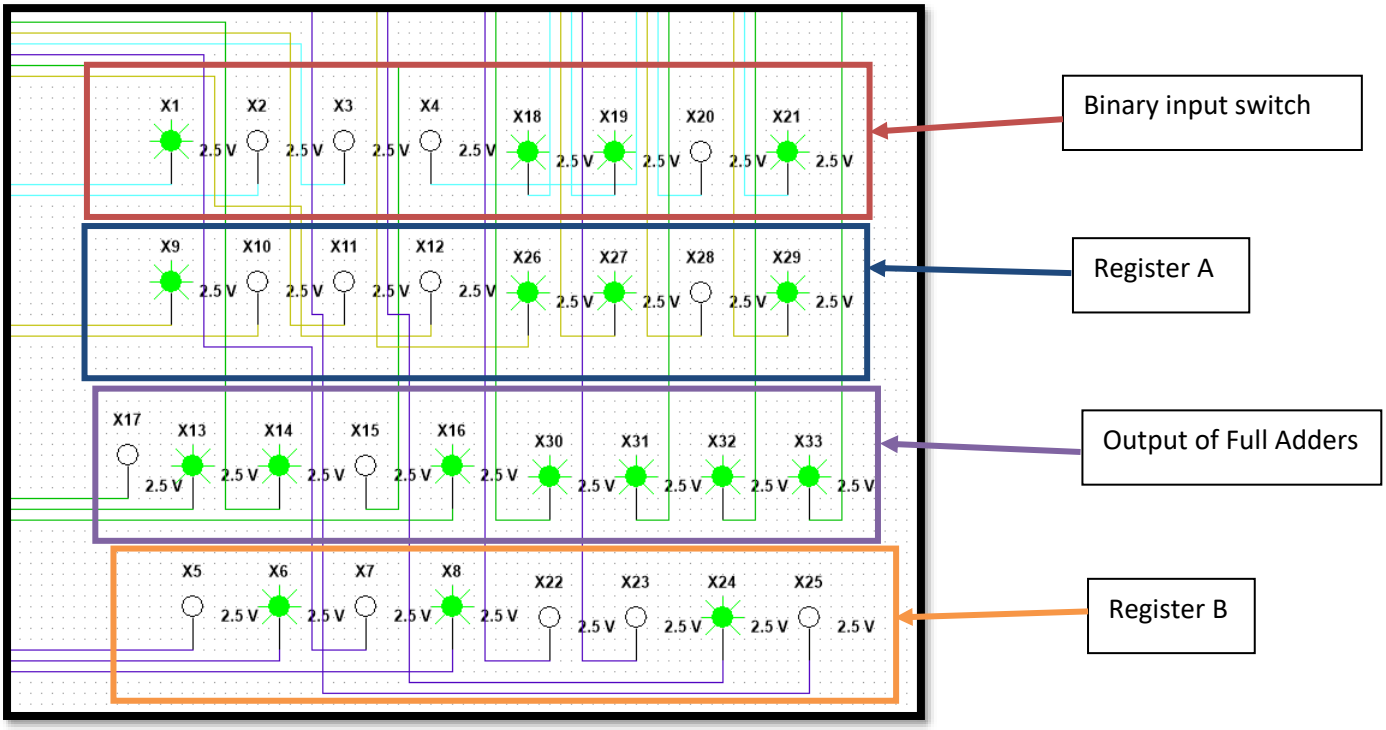
Fig(6.2)


S1A
S2A
S3B
S4B
Key = A
Clock_AB
Key = A
V1
5V
XMM1
reset
Key = Space
Clock_B
Key = A
U5
U6
U7
U8
D_FF
U1
U2
U3
U4
FULL_ADDER
U9
U10
U11
U12
D_FF
X1
X2
X3
X4
X5
X6
X7
X8
X9
X10
X11
X12
X13
X14
X15
X16
X17
2.5 V
Register A
Register B
4-bit Full Adders

Fig(6.3)


S1A
S2A
S3B
S4B
S3A
S4A
S1B
S2B
Key = A
Clock_AB
Key = A
V1
5V
reset
Key = Space
Clock_B
Key = A
U5
U6
U7
U8
U17
U18
U19
U20
D_FF
U1
U2
U3
U4
U13
U14
U15
U16
FULL_ADDER
U9
U10
U11
U21
U22
U23
Register A
Register B
8-bit Full Adders

Fig(6.4)



Figures 7

Fig(7.1)

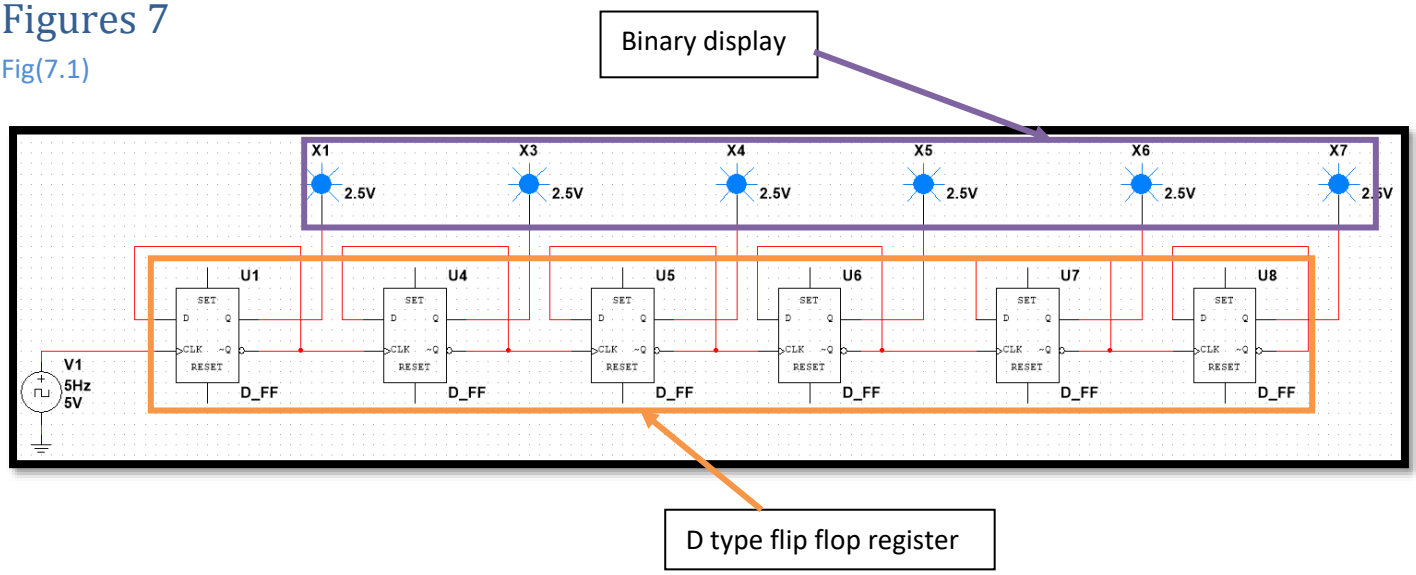
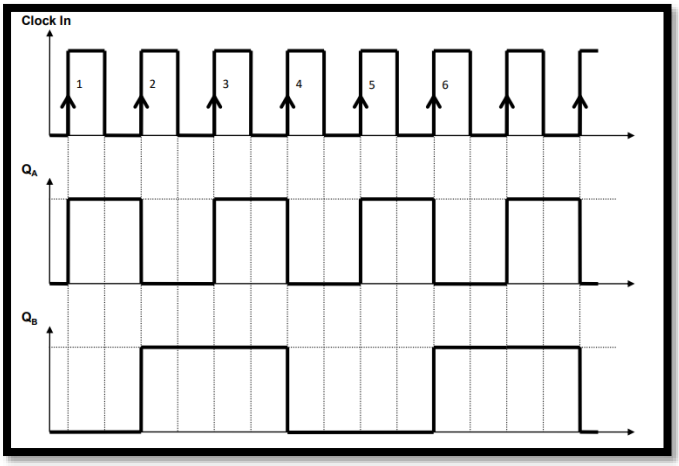
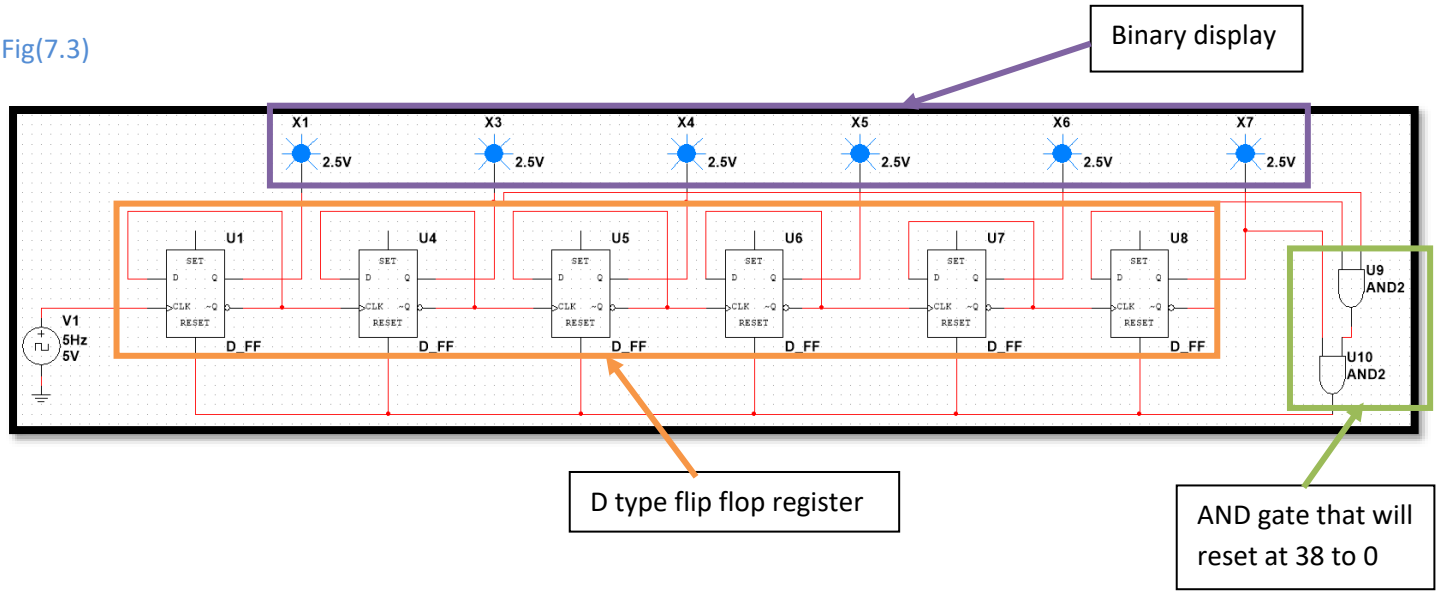


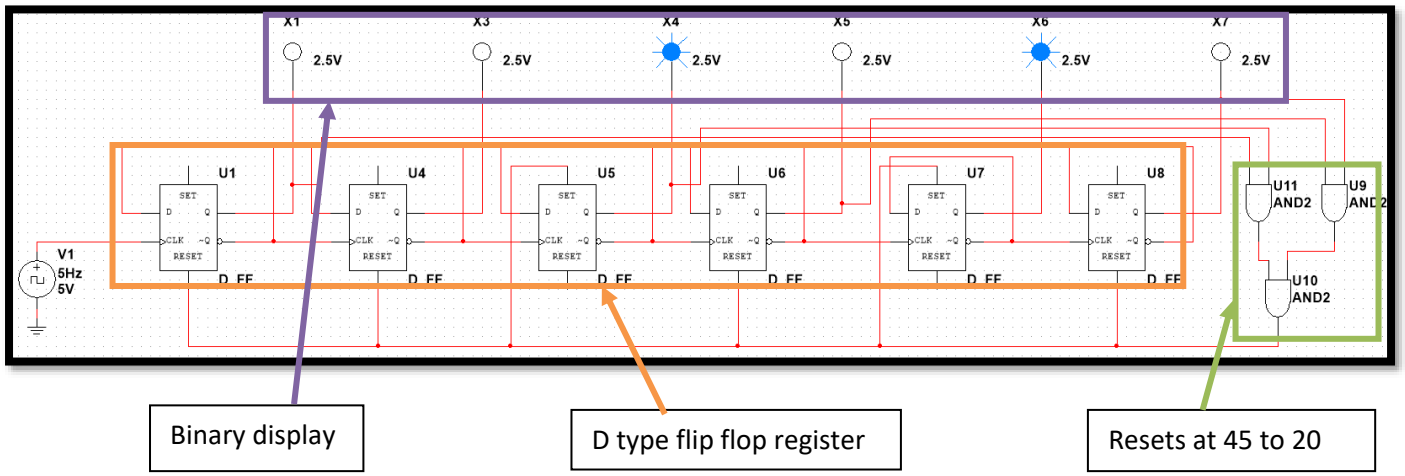
Fig (7.2)



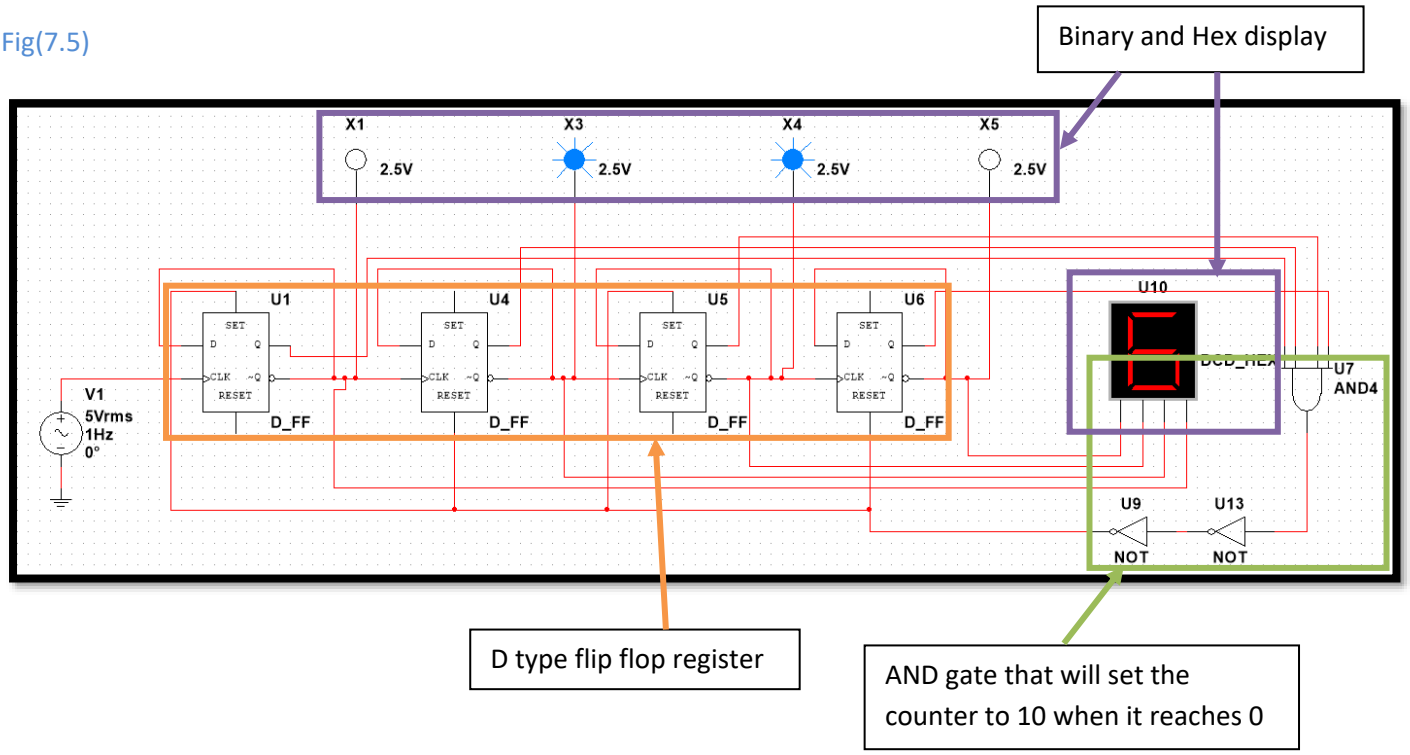
Fig(7.3)


Binary display
X1
X3
X4
X5
X6
X7
2.5V
U1
U4
U5
U6
U7
U8
D_FF
V1
5Hz
5V
U9
AND2
U10
AND2
D type flip flop register
AND gate that will reset at 38 to 0

Fig(7.4)


X1
X3
X4
X5
X6
X7
2.5V
U1
U4
U5
U6
U7
U8
D_FF
V1
5Hz
5V
U11
AND2
U9
AND2
U10
AND2
Binary display
D type flip flop register
Resets at 45 to 20

Fig(7.5)



Tables: -

1)

Inputs/Outputs	Base 2	Base 10
Register A	0 1 1 0	6
Register B	0 0 0 0	0
Output	0 1 1 0	6

Table 1 [4]

Showing the first output of the full adder when register B is clear and Register A has a binary number stored inside it.

2)

Inputs/Outputs	Base 2	Base 10
Register A	1 1 0 1	13
Register B	0 1 1 0	6
Sum	0 0 1 1	-
Carry	1 1 0 0	-
Output	1 0 0 1 1	19

Table 2 [4]

Showing the results of the 4-bit full adder when binary values are in both register A and B being added together.

3)

Inputs/Outputs	Base 2	Base 10
Register A	1 0 0 0 1 1 0 1	141
Register B	0 1 0 1 0 0 1 0	82
Sum	1 1 0 1 1 1 1 1	-
Carry	0 0 0 0 0 0 0 0	-
Output	0 1 1 0 1 1 1 1	223

Tabel 3 [4]

Showing the results for the 8-bit full adder circuit when binary values are in both register A and B being added together

Assignment Introduction: -

For This assignment, there is a task using descrete components to build and understand Op-Amps and wave Rectifier circuits using a simulation software called multisim. This will be for the first part of the assignment, as the second part will require me to simulate the existing logic gates, with their NOT gate equivalents, and a series of synchronous counters, using the same simulation software. The main goal of this assignment is to succesfully simulate a multitude of basic, common circuits that is widely utilised in the designing of other larger, complete circuits. Helping to teach us basic theory on how these circuits work.

The first part of the assignment requires me to construct an inverting and non-inverting Op-Amp. I would have to be able to calculate the nesicery resistor values needed to calculate the gain of the Op-Amp. The gain being specified to be a gain of 10 for the Inverting, and a gain of 8 for the non-inverting Op-Amp.

The next section requires me to construct a half wave rectifier and a full wave rectifire circuit in multisim, recording the results at the output for each circuit when changing the values of the smoothing capacitor. From doing this, we will be able to visualise the output by using a digital osciliscope tool that comes with the multisim software.

For the next part of the assignment, The different types of logic gates must be simulated showing the different combinations and outcomes of each gate. This must be done before replacing the gates with its equivalent NOT gate variant. Showing the design and giving evidance of its outputs for each input combination.

The next task asks for a simulation of a full adder, a 4 bit full adder and an 8 bit full adder circuit. For this there must be a theoretical explaination for how this type of circuit functions. This theoretical explaination must also be backed up by a test showing the result from this simulated circuit, giving the steps for the whole process.

The final task is to simulate a series of counters that count up to 6-bit binary numbers. Each task requesting that the counter would start and stop counting from certain points, counting either up or down depending on the request. This circuit would require the use of D type flip flops to build the appropriate circuit that would have the ability to count up or down in binary, depending on which of the outputs of the flip flops was being used at that time.

After completing these tasks, I will be able to…

- Become compitant with using the multisim software
- Be able to build an inverting or non-invertingop-amp while also being able to calculate which resistor values I would need for my required gain.
- Be able to build a half and full wave rectifier circuit
- Be able to show the output for each logic gate and show each NOT gate equivalent.
- Be able to construct a multiple bit, full adder circuit
- And to be able to create a incrementing and decrementing counter circuit that start and reset when it reaches certain binary values.

Part 1

Task 1: - Inverting Op Amp

1) Construction of an inverting Op-Amp

Using Multi-sim 14.1, looking at *fig (1.1)*, I have constructed an inverting operation amplifier that can be successfully simulated. The first part that is highlighted is the frequency generator, which will give out an AC signal

at any peak voltage and frequency that you set the output to. The Other parts that make up the inverting Op Amp are the resistors, and the Op amp IC.

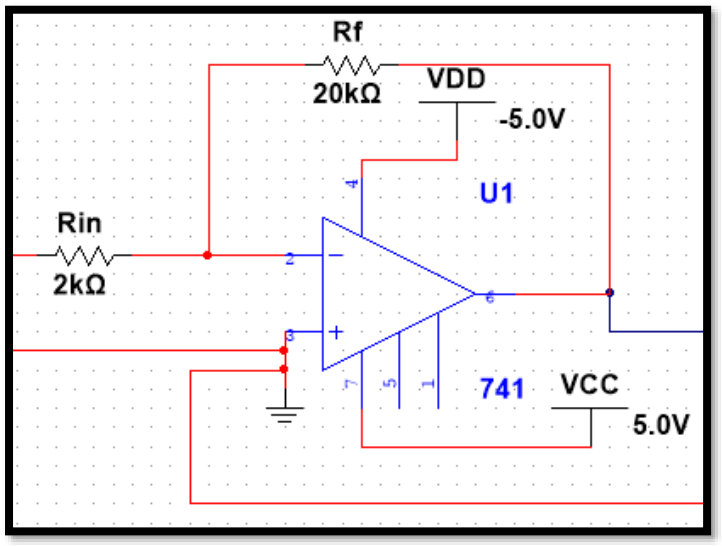


Fig 1.1

The resistors Rf and Rin, determin the gain of an inverting or non inverting Op amp. For these two resistors I have selected *Rin = 2KΩ* and *Rf = 20KΩ*. the formula for an inverting amplifier gain is (*-Rf/Rin = gain*)[2], so if I put my values in this formula I will get (*-20KΩ/2KΩ = -10*). By looking at this value of the gain of -10, I can now predict that the output of the Op amp will have an inverted and amplified sine wave, compared to the input of the Op amp.

The Op amp IC is the center piece for this Amplifier circuit and has an 8 pin configuration. This IC requires a negative and posotive supply rail, so for my circuit I gave it a 5V to -5V DC supply at the 4th and 7th pin, which will dictate what the point of saturation will be for any amplified signals. This means that any signal amplified over 5V or -5V will saturate at these voltages, never to exceed them. Pins 2 and 3 are the negative input and the positive input respectively, the input that I need to use for the inverting amp is the negative input, pin 2, while the positive input is connected directly to ground.

The output of the Op amp can be shown using the osciliscope simulation function and connecting channel A and B to the output of the function generator and the output of the op amp respectively. The results of the input voltage of 320mV can be shown below.

2) Results of an inverting Op-Amp

320mVpk, Inverting Op-Amp: -

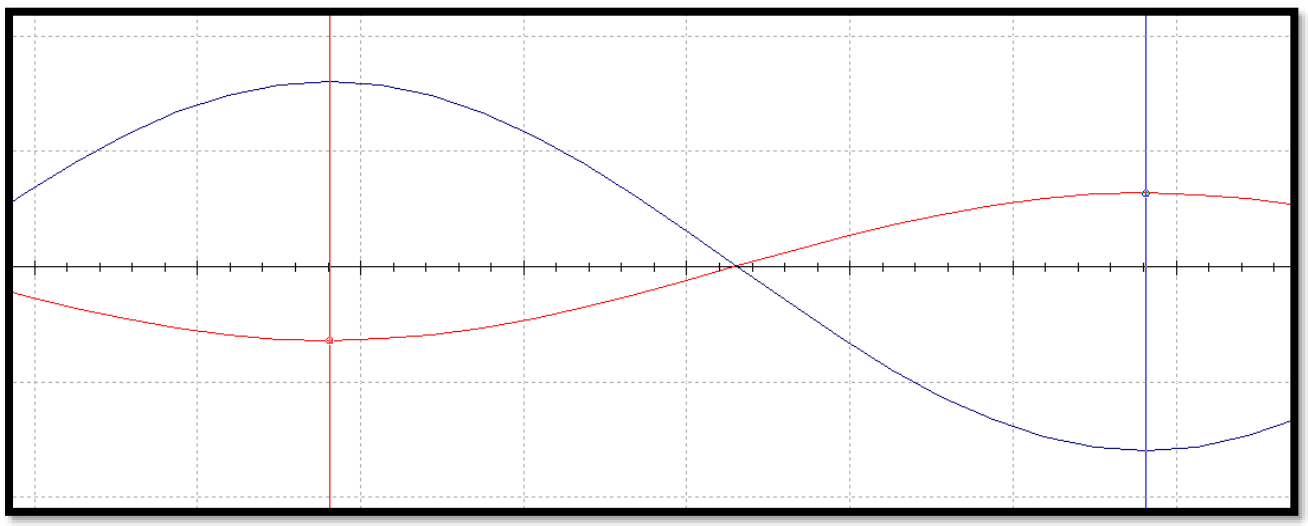


Fig (1.2)

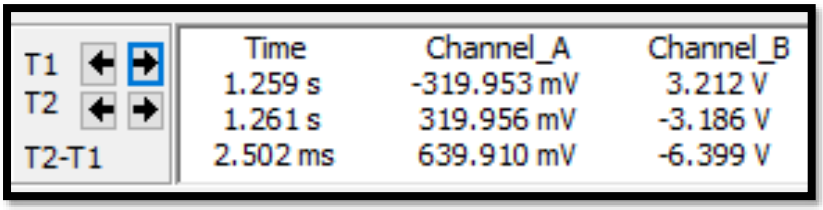


Fig (1.3)

470mVpk, Inverting Op-Amp: -

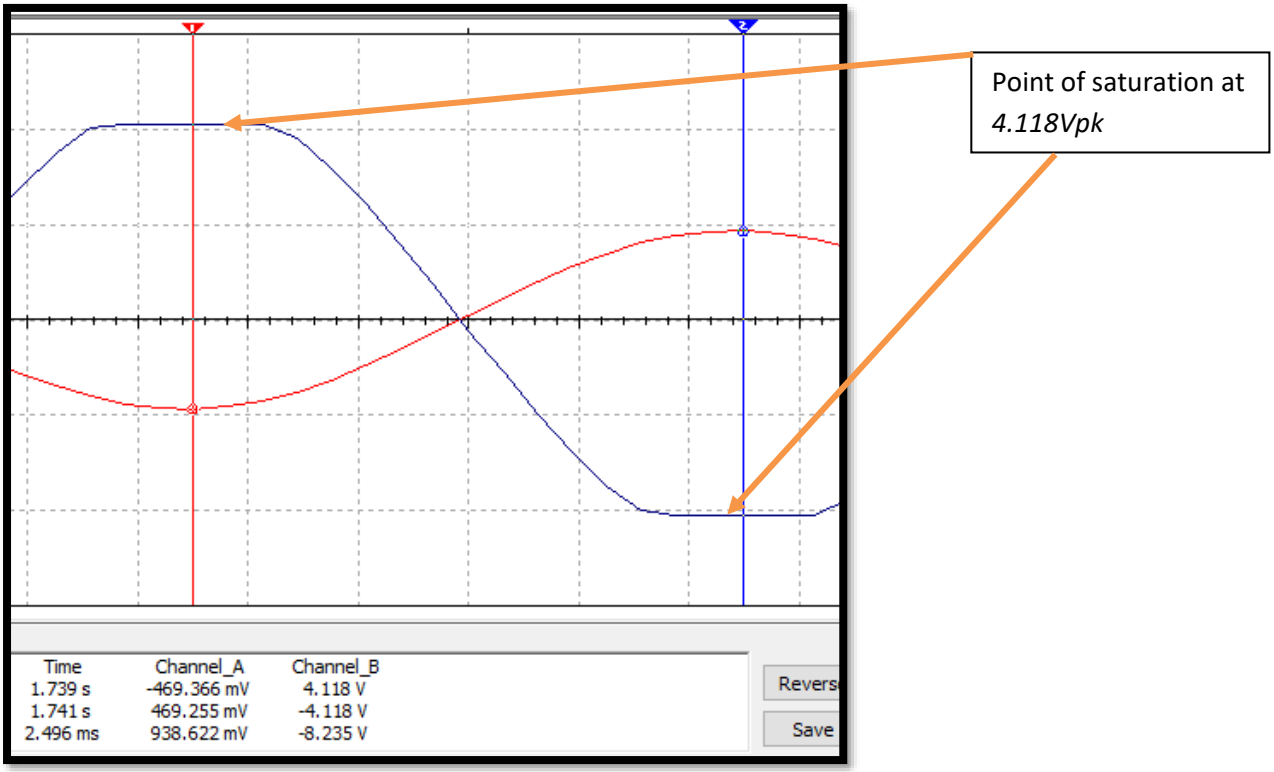


Fig (1.4)

670mVpk, Inverting Op-Amp: -

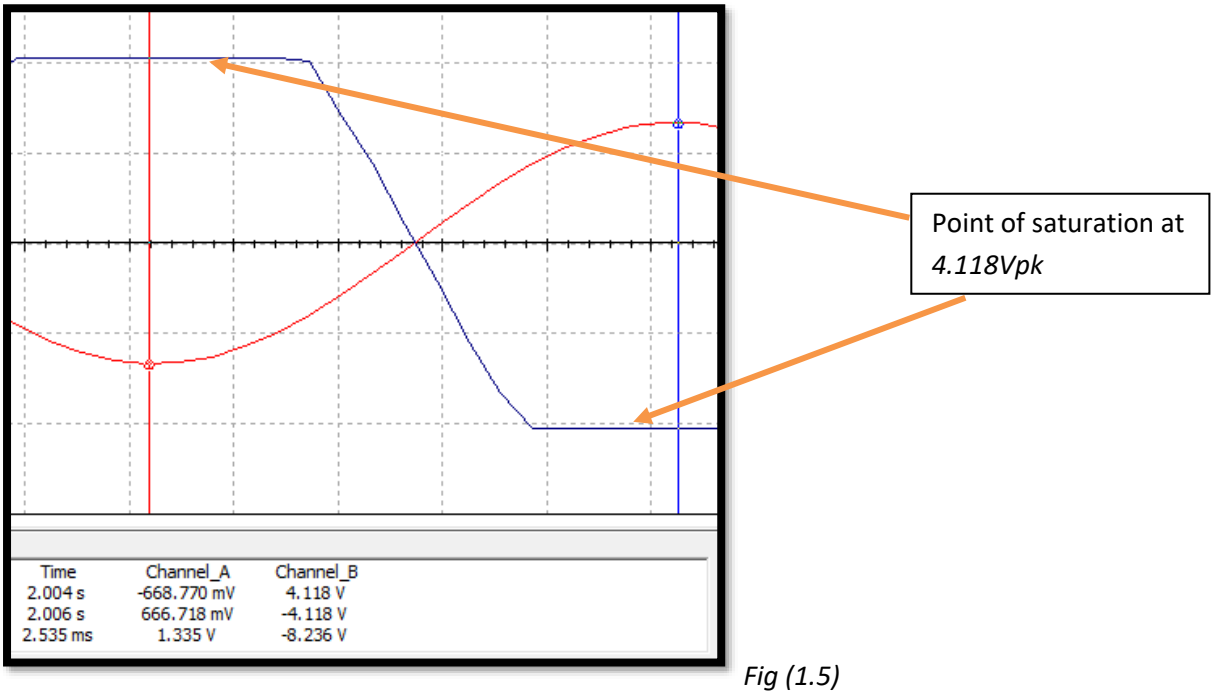


Fig (1.5)

3)

This is the results shown from channel A and B in *fig (1.2)*. A being the red sine wave, the input, and B being the blue sine wave, the output. I could then move the measuring probes, labled T1 and T2, to find the amplitude voltage values for both sine waves. The input voltage being amplified is 320mVpk at the input with a gain of -10.

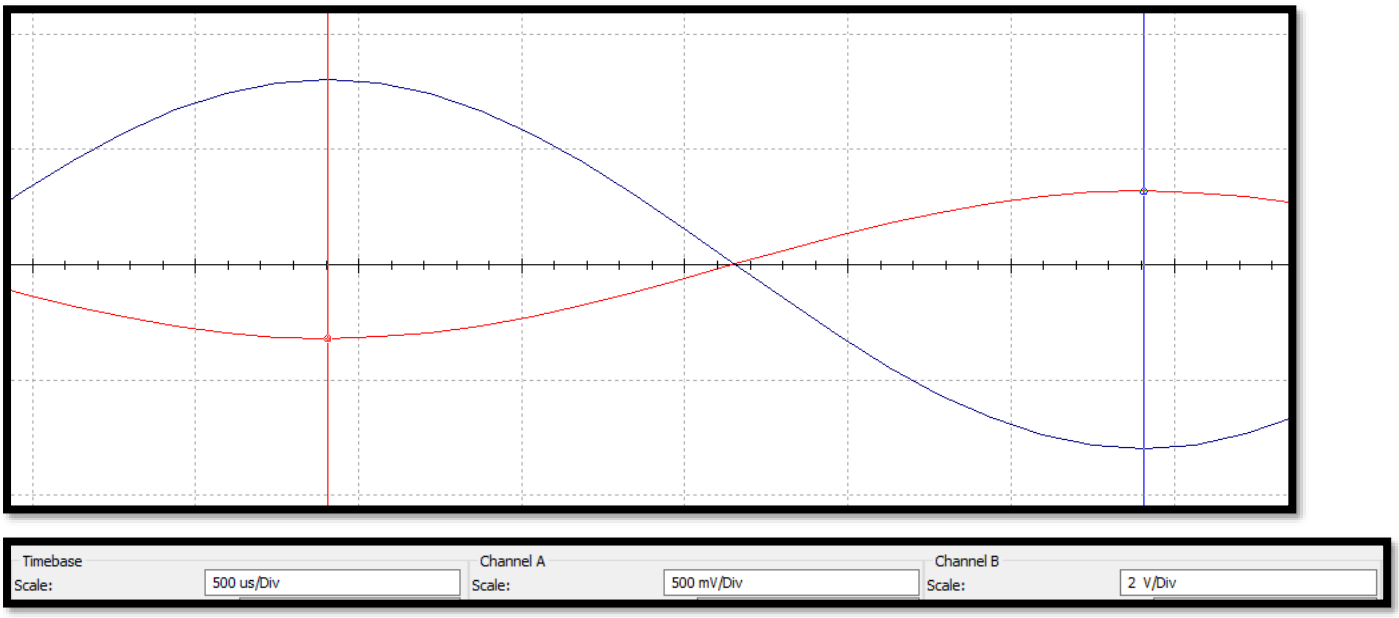


Fig (1.2)

Here in *fig (1.3)*, we can confirm that the input is *319.9Vpk* and a pk-pk voltage of *639.9mVpk-pk*. The voltages are not exact because I had to manually move each probe which I could not make exact, which isn't an issue because I can clearly see that the results are basically the same because of how close the values get to the intended value.

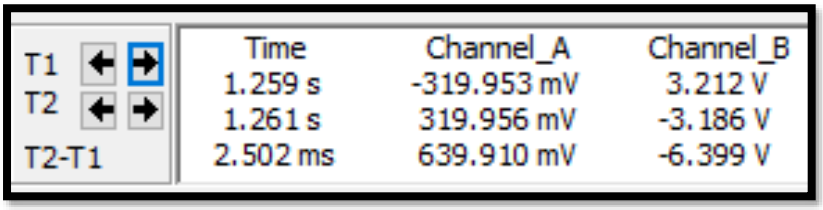


Fig (1.3)

I can then see that the amplified signal is *-3.186Vpk* and *3.212Vpk*, coming to a total of *-6.399Vpk-pk*, at the output of the Op amp.

4)

The calculation for each amplified voltage signal is in the formula (***Voltage * gain = amp voltage***).

Using this formula, we can calculate the voltage for *fig (1.3), fig (1.4)* and *fig (1.5)* before reading the simulated results.

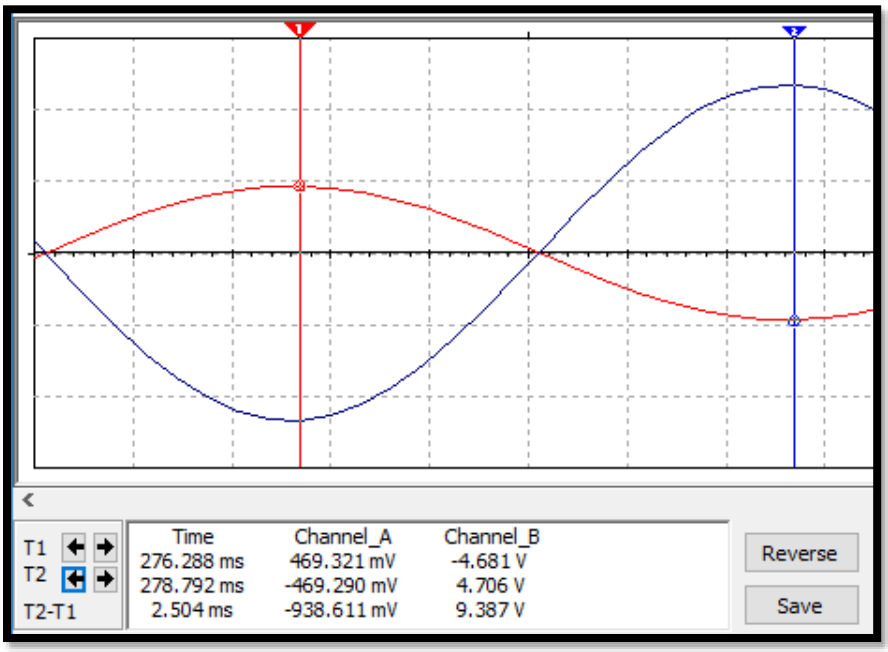
i) ***320*10^-3 * -10 = -3.2V***

Looking at this result, I can confirm that the amplified signal in our calculation is the same as the amplified signal recorded from my circuit in *fig (1.3)*. The recorded value being *-3.2Vpk*.

ii) ***470*10^-3 * -10 = -4.7V***

Looking at the result for this calculation and looking at the results from *fig (1.4)*, I can see that the voltage cuts off at a certain value before the calculated value has been achieved. This is due to the saturation of the circuit which can be amended by increasing the voltage being supplied to the Op amp IC.

Fig (1.6) is the results from the op amp if the power supplied to the IC was increased to *-10V* and *10V DC*. We can see that the saturation has stopped and that the recorded values are similar to my calculated value. The recorded value for this circuit is *-4.681Vpk, 4.706Vpk* and *9.387Vpk-pk.*



iii) ***670*10^-3 * -10 = -6.7V***

From this calculation and seeing the results from my simulation test, *fig (1.5)*, We can also see that the amplified signal is also being saturated with the *-5V* and *5V DC* supply connected. The saturation occurs at *4.1V* it seems. But can also change the DC power supply rails to *-10V* and *10V DC* to fix this issue.

Fig (1.7) is the results from that supply, with the recorded values reaching *-6.68Vpk* which is also similar to my calculated result of *-6.7Vpk*, so my calculations have been accurate for predicting the value of the amplified signal.

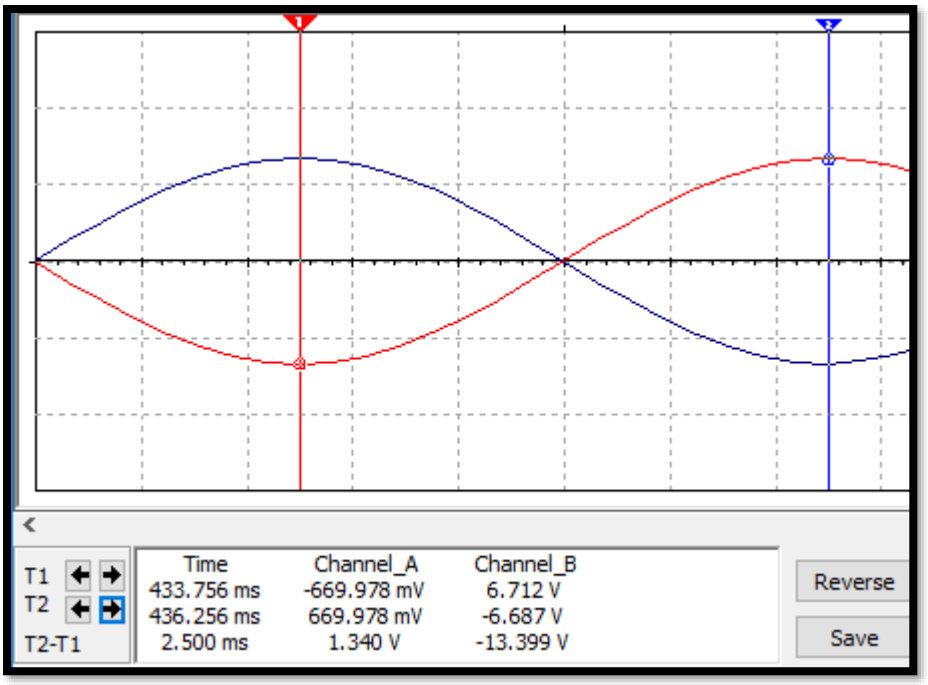


Fig (1.7)

Task 2: - Non-Inverting Op Amp

5)

Looking at *fig 2.1*, An non-inverting op-amp works in a similar way to an inverting op-amp in the sense that they use the same princable of using the two resistors to calculate the gain of the amplifier. This is put into the formula **((Rf/R1) +1)[2]**. However, there is a difference between the two circuits, in the way they are aranged and the effect it has on the signal wave at the output. This circuit will require the calculated gain of 8 for this task as specified. So, the values ive chosen were 1KΩ for R1 and 7KΩ for Rf. This calculates to…

- ***Voltage Gain = (Rf/R1) +1 ➔ (7K/1K) +1 = 8***

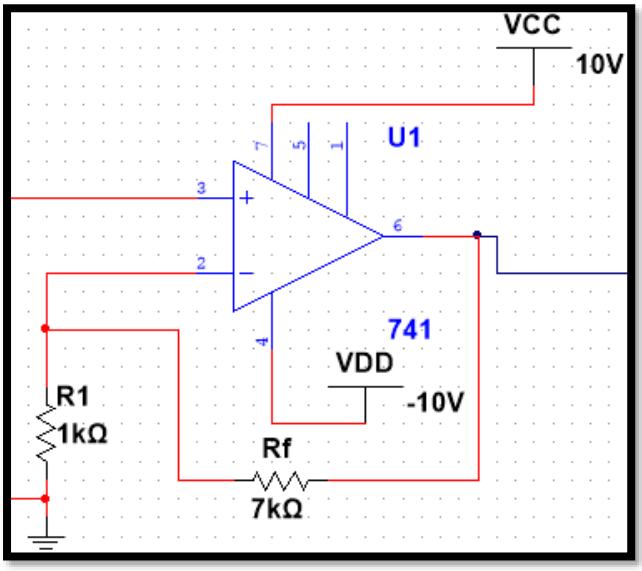


Fig (2.1)

While the inverting amplifier will invert the signal from the input at the output, the non-inverting does the opposite. Not inverting the signal at the output, essentualy keeping the input and output in phase with each other.

I can also see that the signal source is being directly fed into the positive input of the op amp without any resistors, while the negative input is beeing used as the feedback with the resistors connected to determin the gain of the amplifier.

6)

240mVpk: -

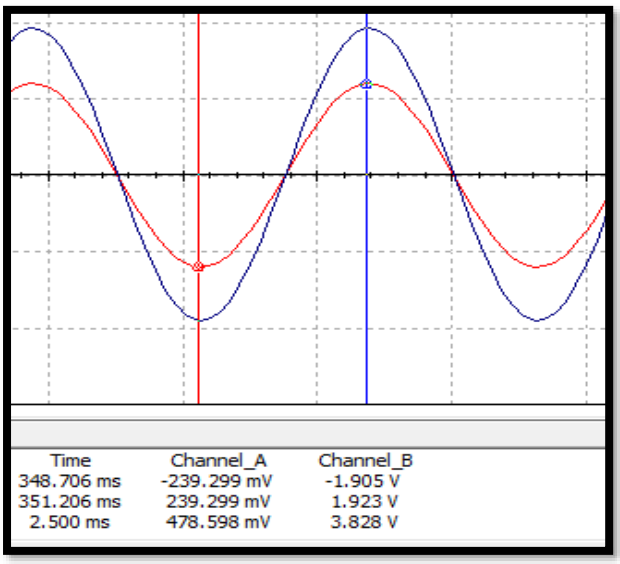


Fig (2.2)

320mVpk: -

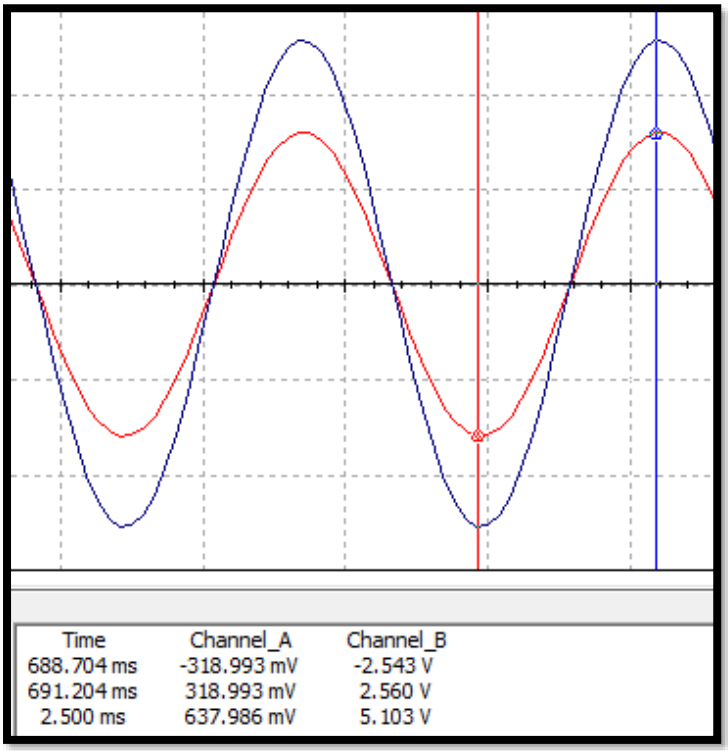


Fig (2.3)

800mVpk: -

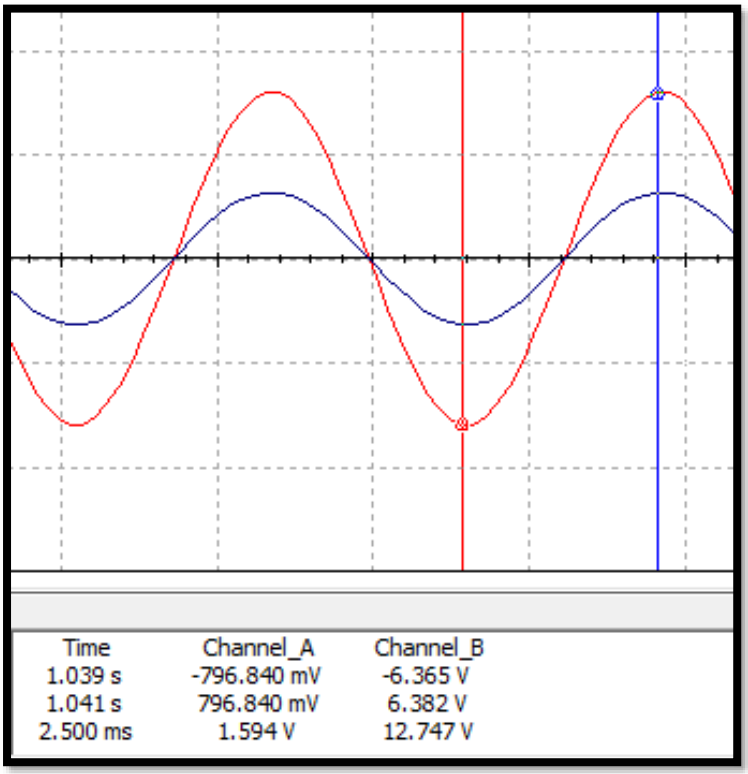
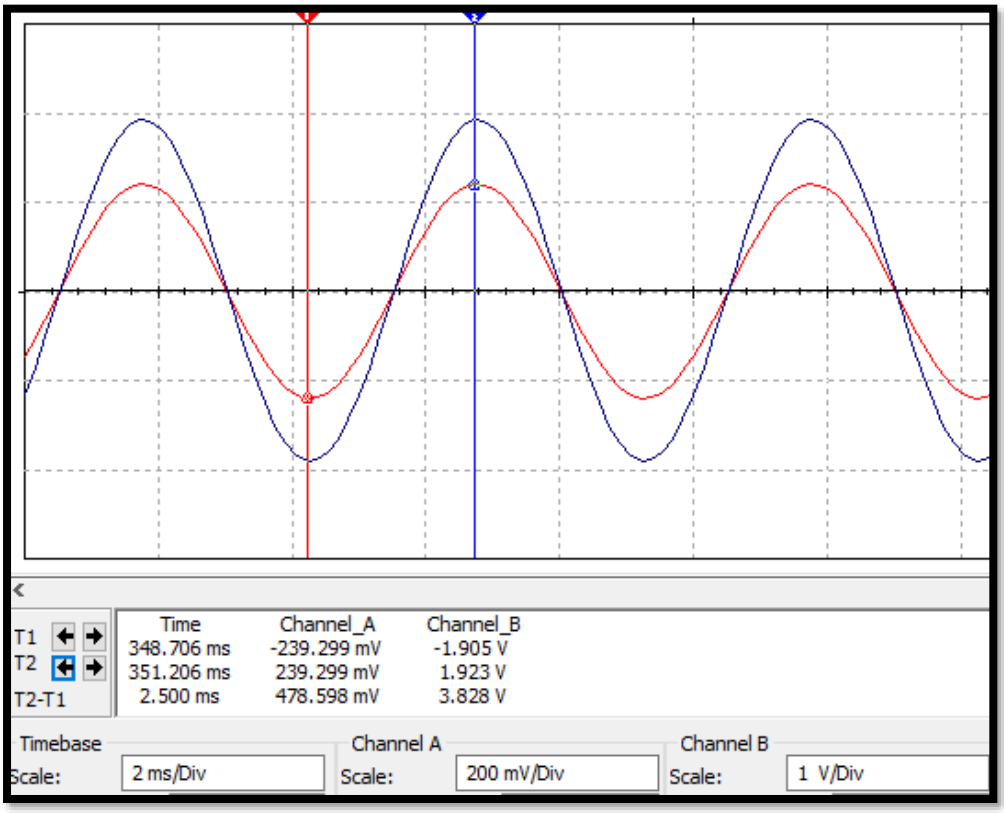


Fig (2.4)

7)

Fig 2.5 shows the input and output signals while also giving an indication for the scaling



8)

i) ***240mVpk*8 = 1.94Vpk***

in *fig 2.2* we can see that the recorded value is *1.905Vpk, 1.923Vpk* and *3.828Vpk-pk*, so the recorded value is similar to the calculated value of *1.94Vpk*

ii) ***320mVpk*8 = 2.56Vpk***

in *fig 2.3* we can see that the recorded value is *2.56Vpk, -2.54Vpk* and *5.10Vpk-pk,* so the recorded value is similar to the calculated value of *2.56Vpk* up to two decimal places.

iii) ***800mVpk*8 = 6.4Vpk***

in *fig 2.4* we can see that the recorded value is *6.382Vpk, -6.365Vpk* and *12.747Vpk-pk*, which is also accurate to my calculation of *6.4Vpk* using the formula of, ***(vin * gain).***

Task 3: - Half Wave Rectifier

9)

Looking at *fig 3.1* we can see that there are two parts to this half wave rectifier. The *diode* to block all negative voltages and only allow positive voltages through, minus the *0.7V* voltage drop needed to power the diode. And the resistor which acts as the load of *100kΩ* for the circuit.

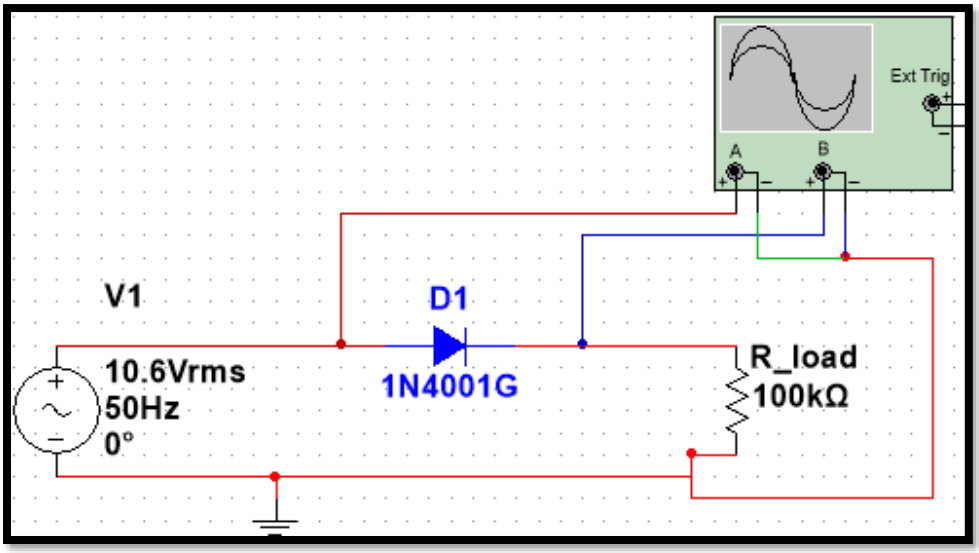


Fig (3.1) [6]

The diode *1N4001G* has a voltage drop of *0.7V* which is needed to power the diode to allow the current to flow. But because of this trigger voltage, the diode will only allow a voltage through if it is over *0.7V*, therefore blocking any negative voltages from flowing, only allowing the positive voltages through that exceed the *0.7V* range [6]. Because of this the voltage wave will have gaps between each arch where the negative voltages were originally.
The resistor is used to act as a load for the circuit, to complete the flow of the circuit and to draw power through the circuit.

10)

If we look at the results from the simulation for the half wave rectifier, the input voltage was roughly *15Vpk* while the output voltage is *14.5Vpk*, which has a smaller voltage drop than I expected. The voltage drop being only *0.5V* instead of *0.7V*. This can be because of the resitstance value of the load though because when the value has been lowered, the voltage drop has gone back to the *0.7V* that was expected.

Either way, as we can see in *fig 3.2* the negative half of the wave gets cut off while the positive half has a voltage drop of *0.5V* applied to the output rectified half-wave.

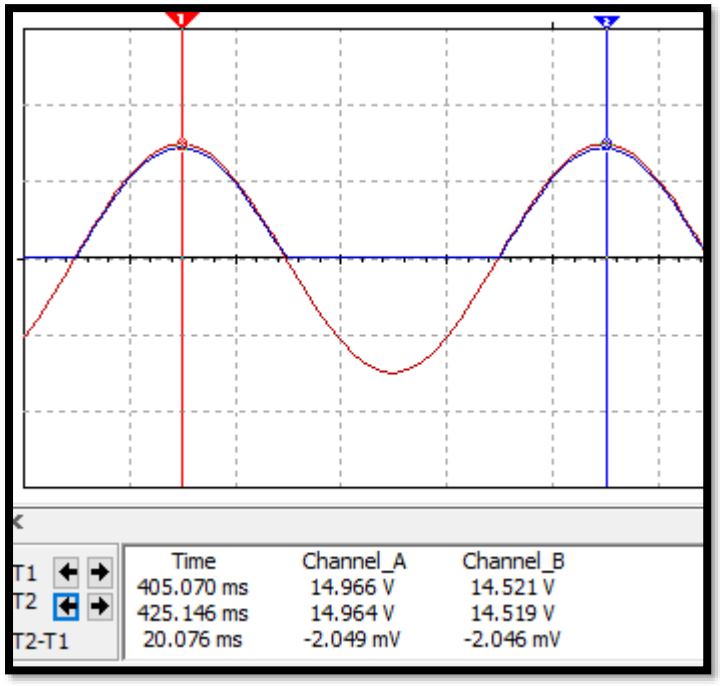


Fig (3.2)

11)

Fig 3.3 show the rectified half wave when a smoothing capacitor of 10nF is applied to the circuit parallel to the resistor.

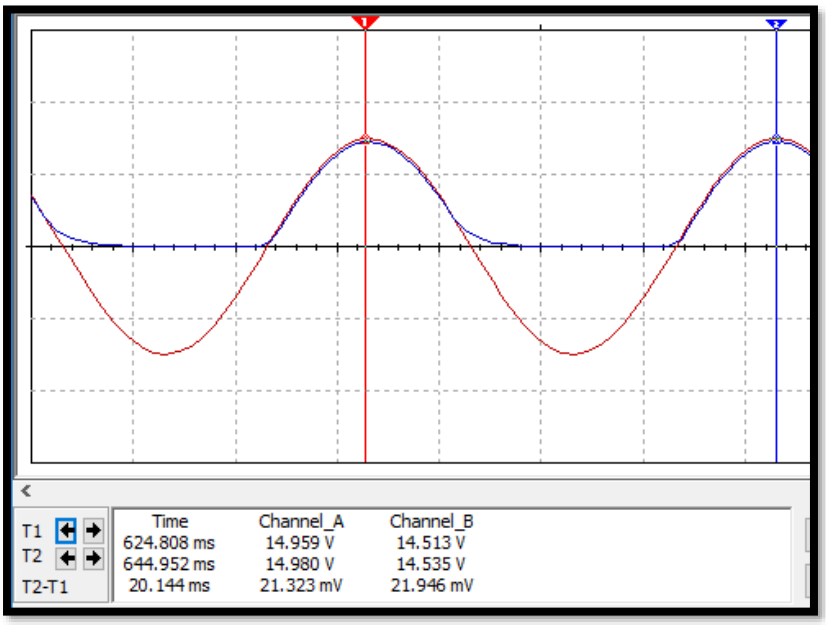


Fig (3.3)

Fig 3.4 shows a smoothing capacitor of 100nF

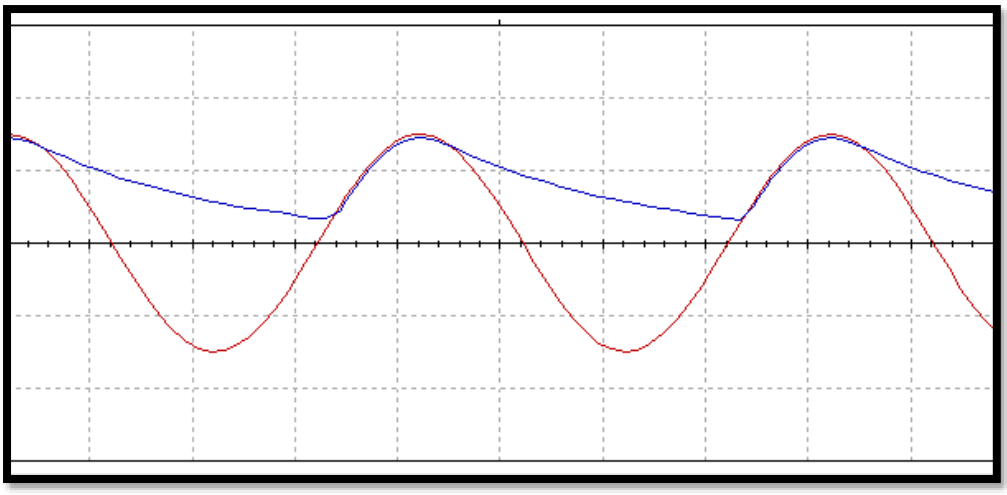


Fig (3.4)

Fig 3.5 shows a smoothing capacitor of 500nF

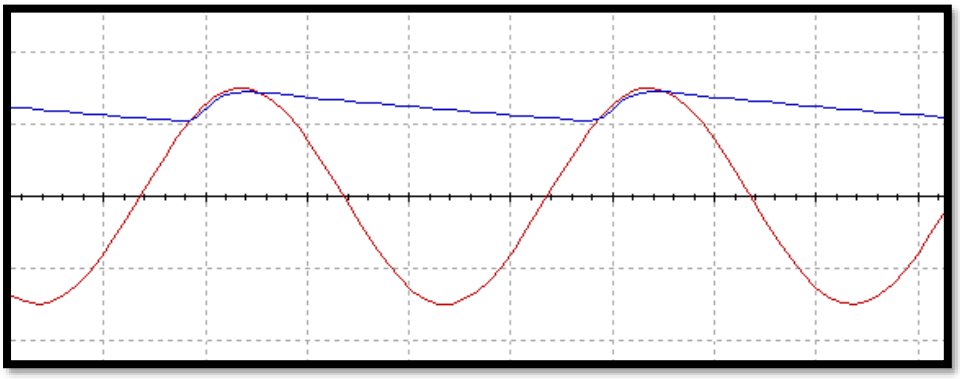


Fig (3.5)

Fig 3.6 shows a smoothing capacitor of 1uF

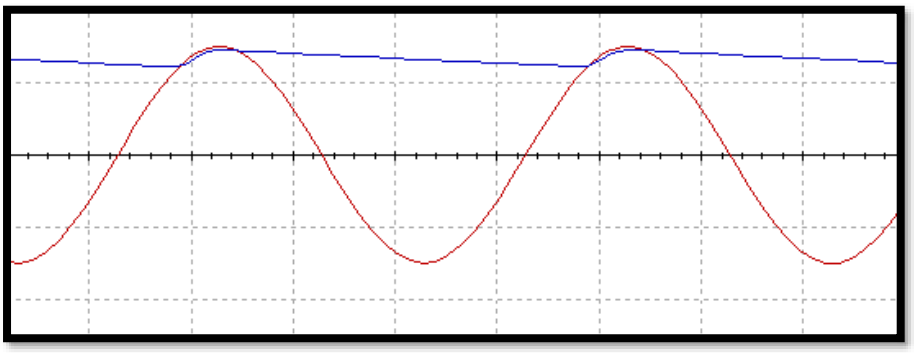


Fig (3.6)

12)

By looking at the results shown by *fig 3.3 – fig 3.6*, we can observe that the effect of the smoothing capacitor smooths the decline of the voltage more efficiently when a bigger capacitor is used in parallel to the load resistor. The reason for this is because a larger capacitor will have a slower discharge time in comparison to a smaller capacitor, creating this smoothing effect that will more accurately recreate a DC voltage.

Task 4: - Full Wave Rectifier

13)

In *fig 4.1* we can see that there are now four diodes connected to make a full wave rectifier compared to the half wave rectifier. This is now because, using this diode gate method, there will be no gaps in the voltage wave where the negative voltage should be. Instead, replacing it with a positive wave to close the distance between the positive waves. This is done by each half of the diode gate acting as a half wave rectifier and a way to block the other voltage from the other half from leaking into the other half wave rectifier. Because of this, we can take the output of each half wave rectifier and add them together to create a full wave rectified voltage. This is done because this diode arrangement allows for current to flow in both half cycles of the AC input. [6]

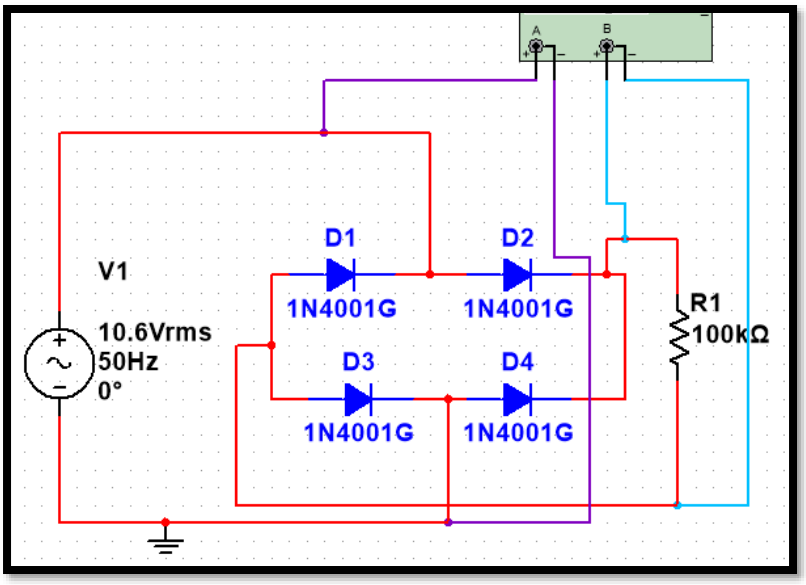


Fig (4.1) [6]

The reason it does this instead of overlapping each other without any change is because each wave is out of phase with each other by 180◦ degrees, so they will add together to make a rectified full wave voltage because of the phase difference.

14)

This is the result of the full wave rectifier when *15Vpk-pk* is applied to the input of the circuit, as shown in *Fig (4.2)*. looking at the results of the test, we can observe that the voltage has a drop of around *886mV*, which is odd since the expectation was that the voltage drop would be around *1.4V*, because of the voltage having to pass through two diodes, meaning a voltage drop of *0.7V* twice.

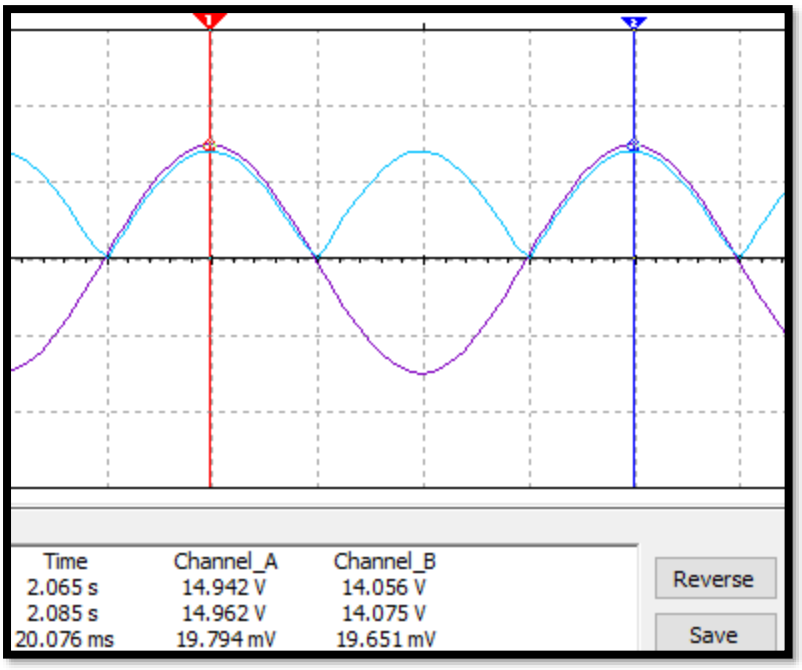


Fig (4.2)

15)

10nF capacitor connected parallel to the output of the full wave rectifier:

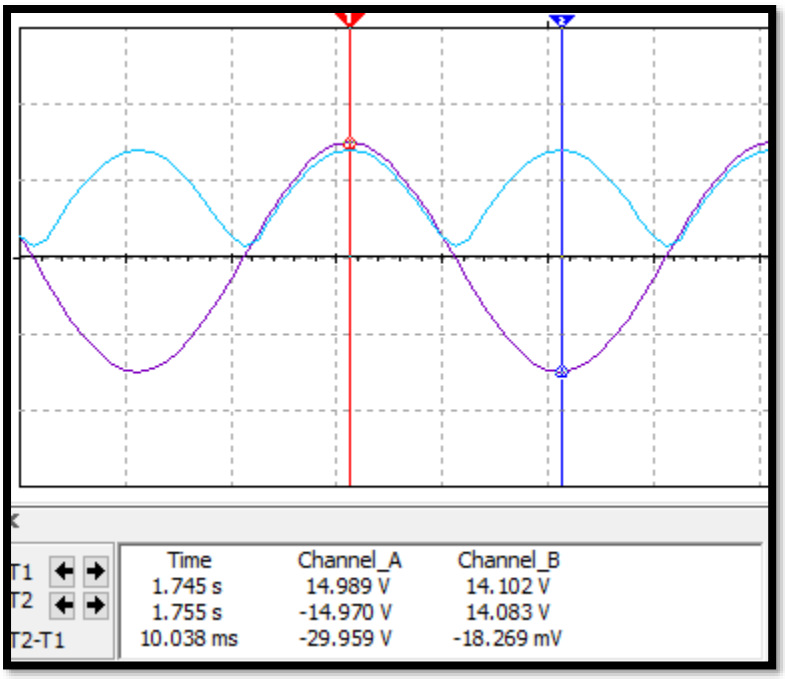


Fig (4.3)

100nF capacitor connected parallel to the output of the full wave rectifier:

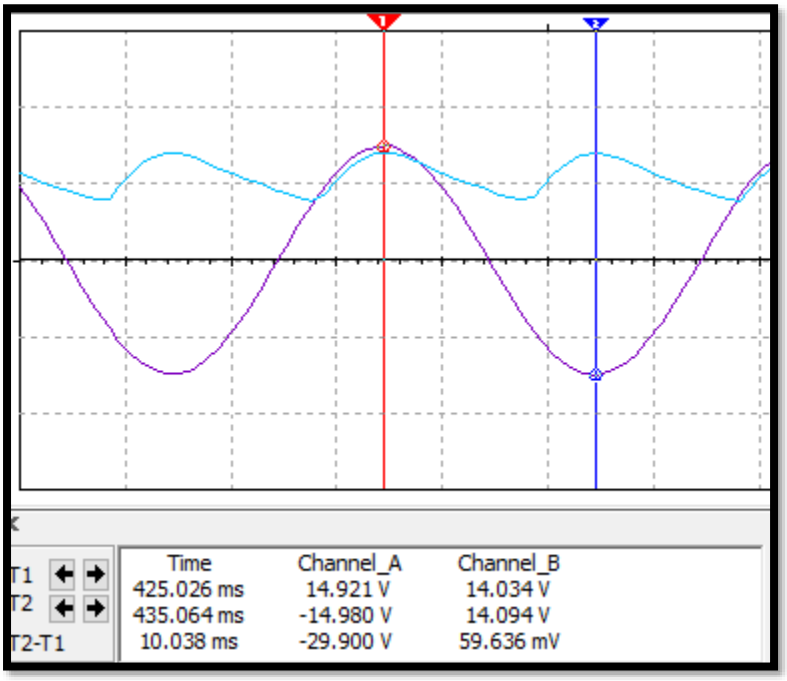


Fig (4.4)

500nF capacitor connected parallel to the output of the full wave rectifier:

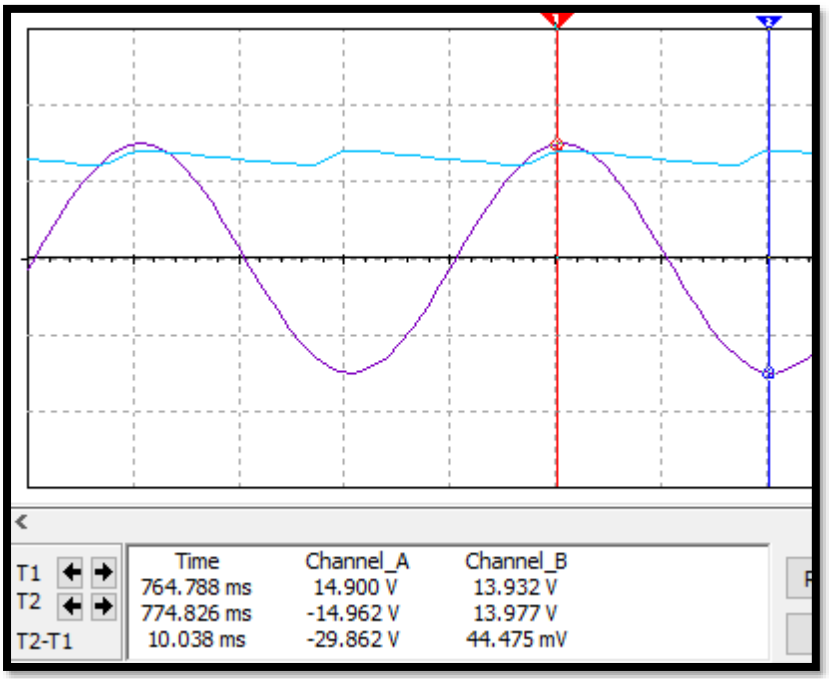


Fig (4.5)

1uF capacitor connected parallel to the output of the full wave rectifier:

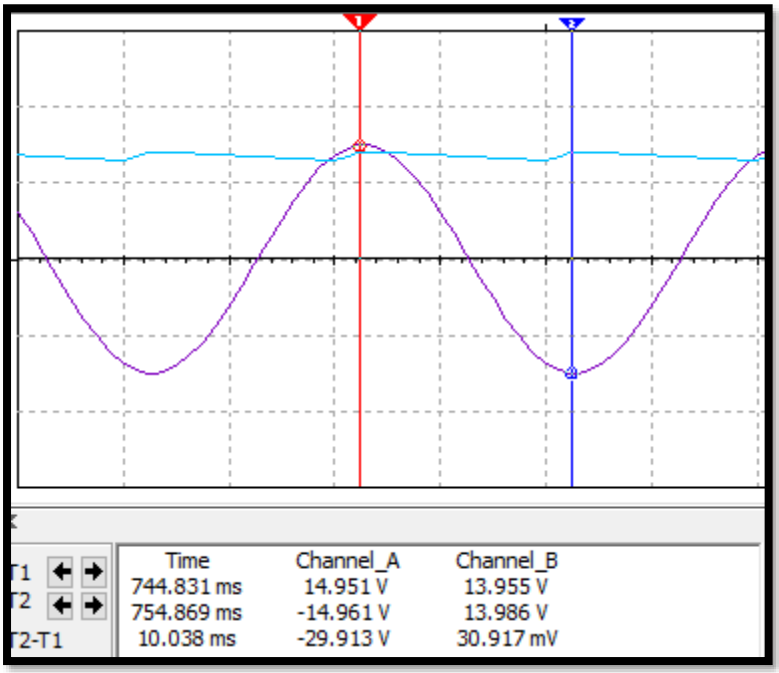


Fig (4.6)

16)

Looking at the results of the full wave rectifier from *fig (4.3)* to *fig (4.6)*, we can see that the charging and discharging of the capacitor has smoothed the positive signal waves. This shows that, similarly to the half wave rectifier task, the bigger the capacitor, the smoother the signal being generated.
This method is seen as being more efficient than the half wave rectifier circuit since it can use twice the number of waves with the diode gate method. Therefore, making it easier to get a more consistent, flat line, DC voltage at the output.

Task 5: - Logic Gate Simulation

17)

Looking at the *figure 5.1, 5.2, 5.3 and 5.4*, we can see the results for every logic gate when the logic inputs change. We can see the change by looking at the probe connected at each output of the gates. By looking at the logic outputs of each gate via a truth table then we can predict what the next logic output would be for each logic gate.

Starting with the OR gate, the truth table show that the gate should have a logic output of 1 when either input is logic 1 or if both inputs are at logic level 1. The truth table can be seen in *Fig 5.5*.

The prediction can then be put to the test when trying to simulate this in multisim as shown in *figures 5.1, 5.2, 5.3 and 5.4* with each logic input in chronilogical order. As it can be seen when simulated in multisim, the probe only shows a positive output of logic level 1, when a logic level 1 is introduced into either inputs of the gate.

The opposit of an OR gate would be its sibling, the NOR gate (*shown in fig 5.7*). The NOR gate has the exact opposite logic combinations as the OR gate, only having an output at logic level 1 when none of the inputs have logic 1 inputs.

We can now move onto the at the AND and NAND gates. The AND gate allows a logic output of 1 only when both inputs have a logic input of 1, shown in *fig 5.10*. This is effective for when you need two positive inputs at the same time to trigger an event within the circuit.

The opposite of an AND gate is a NAND gate, which acts in an opposite way to the AND gate as it only allows a logic output of 1 when both inputs are anything but a combination of two logic 1 inputs, as shown in *fig 5.6*. Here we can see a theme occuring from the fact that a 'NOT' AND and a 'NOT' OR, (or NAND plus NOR), act as a mirror image of their respective counterparts in logical function.

The next pair pair of gates to be introduced is the EXOR and EXNOR gates. The EXOR gates acts similarly to an OR gate, but only gives an output of 1 when one or the other of the inputs are at logic level one, however, not when both inputs are at logic level 1 or logic level 0. So this gate only allows one or the other through, making it an EXCLUSIVE OR function, as can be shown in *fig 5.8*.

The EXNOR is its opposite by only allowing either both logic 1's or logic 0's through to the conclusion of an output of logic 1, backed up by *fig 5.11* looking at the truth table for the EXNOR gate.

And finaly, the last gate to discuss is the NOT gate which, to put it bluntly, inverts the input of the gate at the output. This can also explain the other mirrored halfs of the other gates because a not gate can be thought to be used to invert the output of an AND, OR and EXOR gate into an NAND, NOR and EXNOR gate. The truth table for this gate can be shown in *fig 5.9*.

18)

Looking at *Figures 5.12* through to *5.18*, we can see the equivalent NAND gate arangement, that fits the truth table for the seven logical gates listed above.

- OR Gate [3]

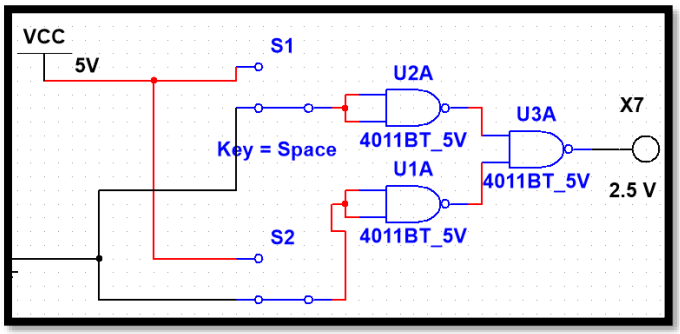


Fig (5.15)

- AND Gate [3]

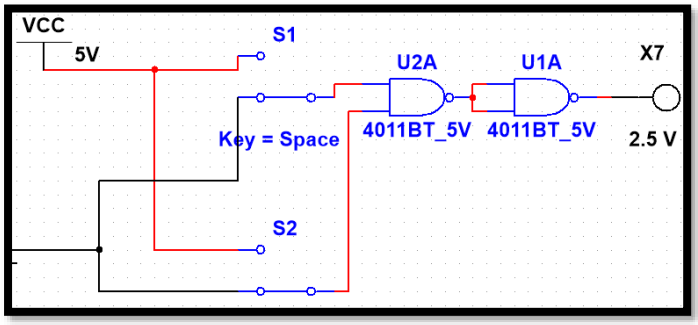


Fig (5.14)

- NOT Gate [3]

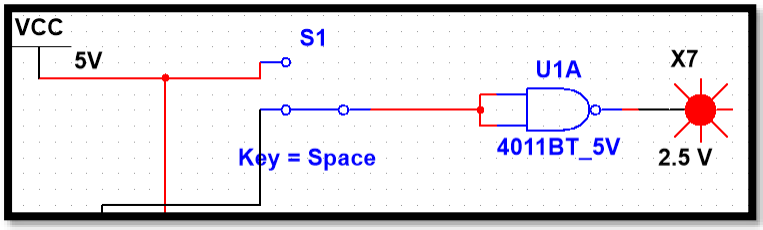


Fig (5.12)

- NAND Gate [3]

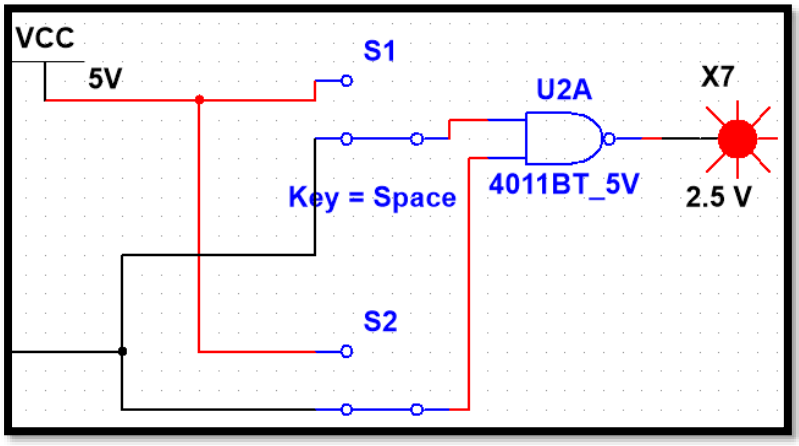


Fig (5.13)

- NOR Gate [3]

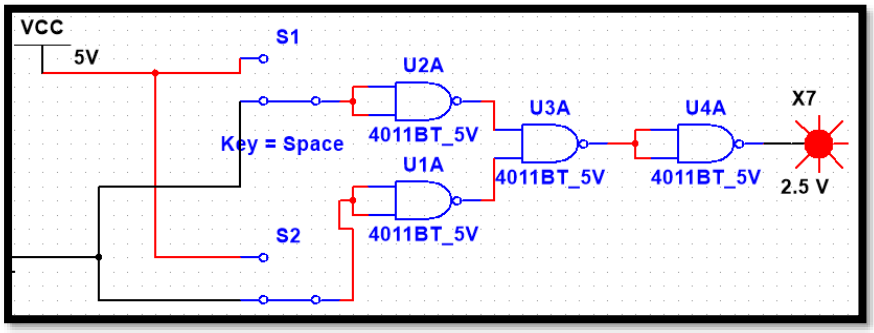


Fig (5.16)

- EXOR Gate [3]

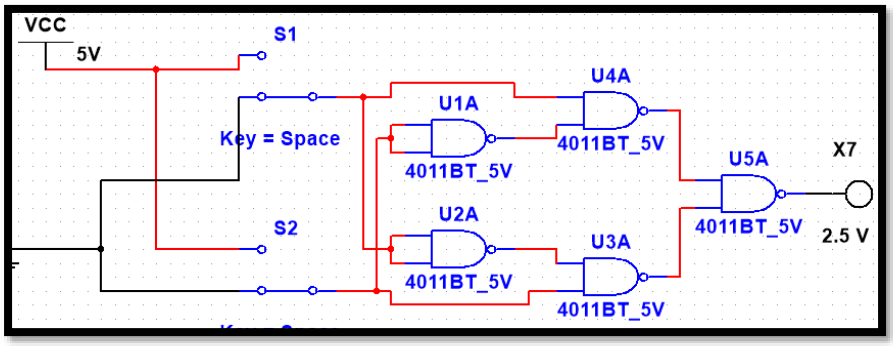


Fig (5.17)

- EXNOR Gate [3]

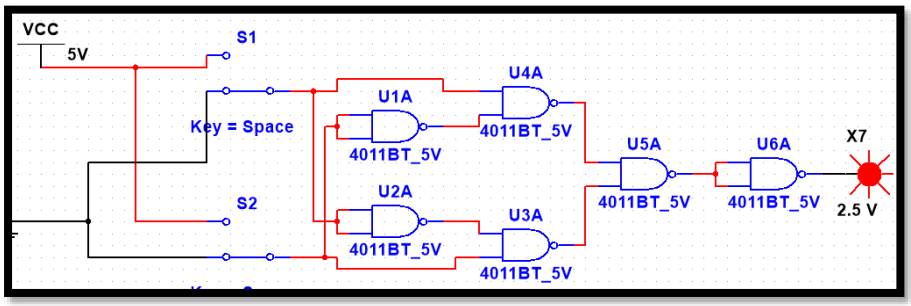


Fig (5.18)

Task 6: - Full Adder Circuits

19)

This is the circuit design for a single bit full adder circuit, as shown in *Fig (6.1)*.

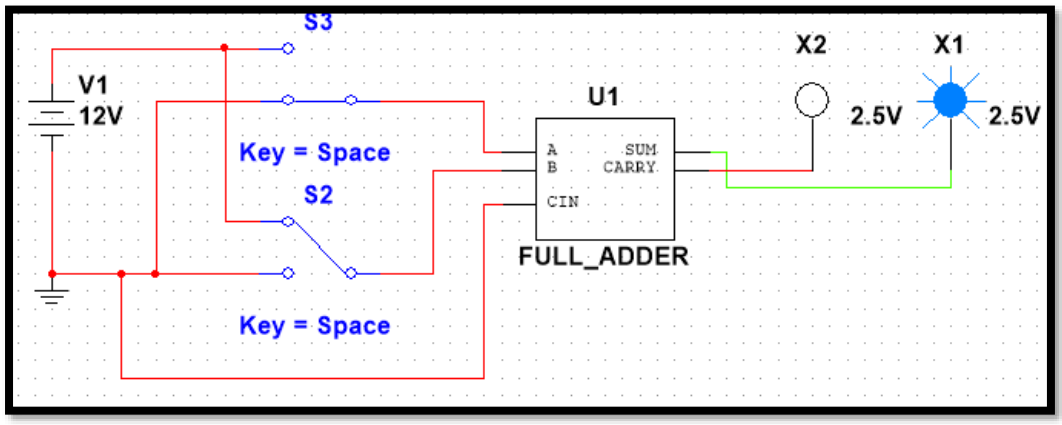


Fig (6.1.2)

The full adder would first take the singular bit being placed in the B and A register, comparing the binary bits together to determine the output. So, in the scenario above, the input bits are “Register A = 0 and Register B = 1”. From this, the result is shown to be…

0 and 1 = 1

From this we can see that the output of the sum is 1. In a more basic term, this means that *1 + 0 = 1 [4]*. However, when an input of “Register A = 1 and Register B = 1”, a different result is shown.

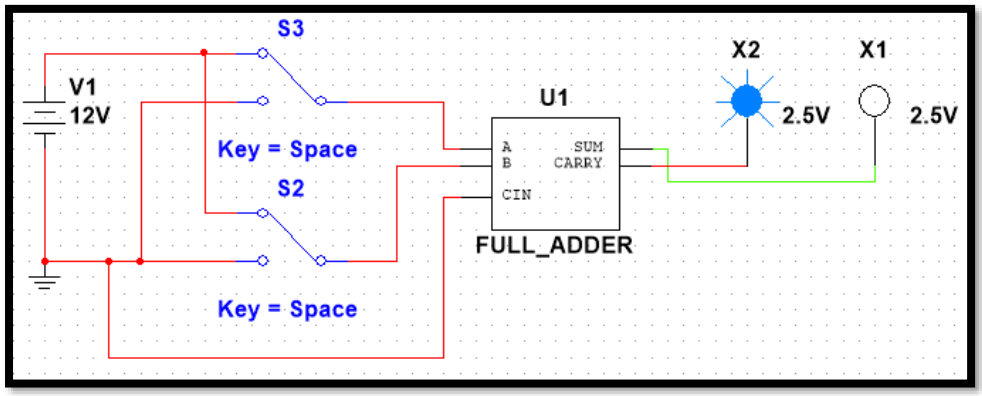


Fig (6.1.4)

1 and 1 = 1 0

Because both registers had a logic level of 1, the output would be a 0 at the pin labelled ‘sum’, while the pin ‘carry’ would carry over the value ‘1’, when both Registers A and B are equal to 1. The carry bit being the most significant bit. So, this would be in a numerical equation look like, *‘1 + 1 = 2’*.

20)

Here is a screen shot of a 4-bit, full adder circuit that was constructed and simulated in Multisim. For this example, the binary values of…

1101 = 13

and

0110 = 6

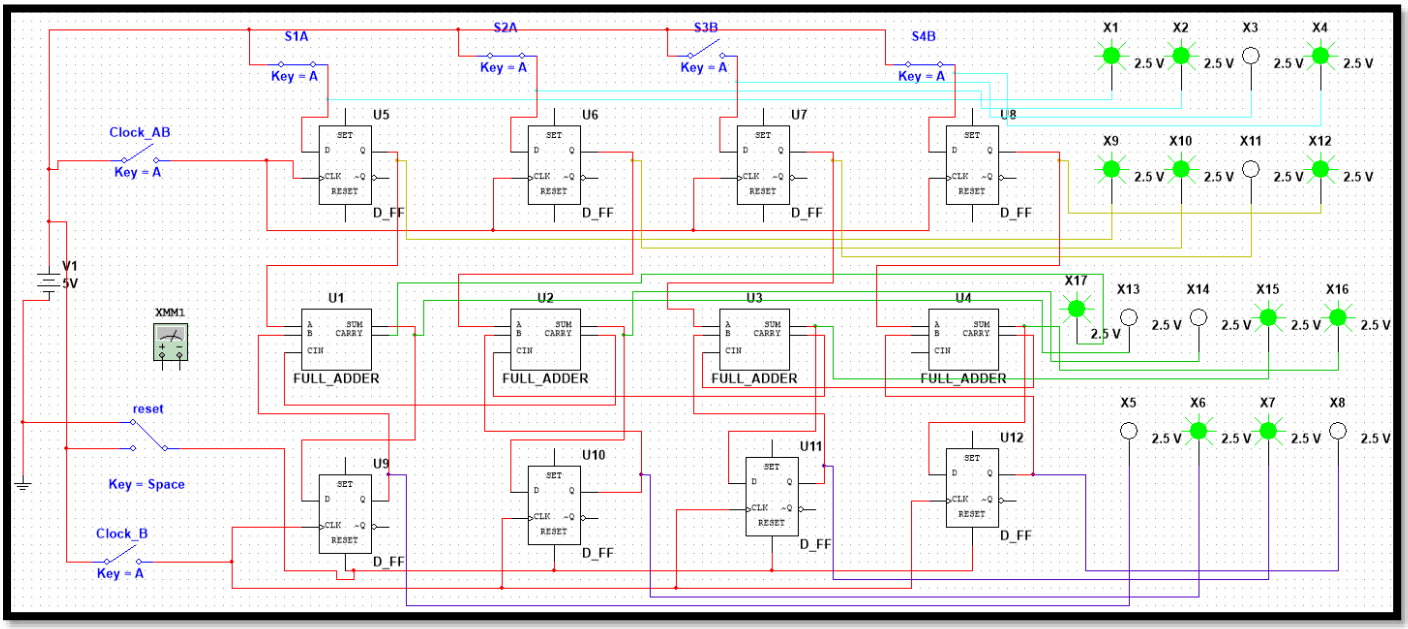


Fig (6.2) [4]

The first step for operating this circuit is to first reset the B register by setting the reset switch to logic level 1. Clearing all the outputs of the bottom B register d type flip flops to equal the binary number for 0. This is so when the contents of register A gets stored in register B, there won’t be any errors in the process.

The binary number must then be decided and be stored in the A register ready. This is done by triggering the clock pulse for the A register ‘D flip flops’, storing the binary input into the register. During this process it is possible to overwrite this binary number, so the clock pulse for register B must be more than twice as long as that of register A to prevent this error.

The next step is to trigger the clock pulse for the B register to store the contents of the full adders’ outputs. The outputs of the full adders at this time would be…

Inputs/Outputs	Base 2	Base 10
Register A	0 1 1 0	6
Register B	0 0 0 0	0
Output	0 1 1 0	6

Table 1 [4]

So, when the B register clock gets triggered the contents of the sum outputs will be placed and stored in the B register.

Now that the B register has its contents stored it is now time to give register A, a new value. Once the value has been decided and set, the clock pulse can then be triggered high again. Storing the binary value in the contents of register A.

Once this has been done, the full adders will automatically add the two binary numbers in register A and B together. Giving the value as a binary number at its C-out pins and the most significant bit being the Carry bit on the far-right side, or the most significant bit side of the circuit. Being the largest bit for the Adder.

The results for the solution are shown in the table below…

Inputs/Outputs	Base 2	Base 10
Register A	1 1 0 1	13
Register B	0 1 1 0	6
Sum	0 0 1 1	-
Carry	1 1 0 0	-
Output	1 0 0 1 1	19

Table 2 [4]

21)

The next task requires that an 8-bit full adder was to be constructed and tested. This circuit can be shown below with its results. *Fig (6.3)* and *Fig (6.4).*

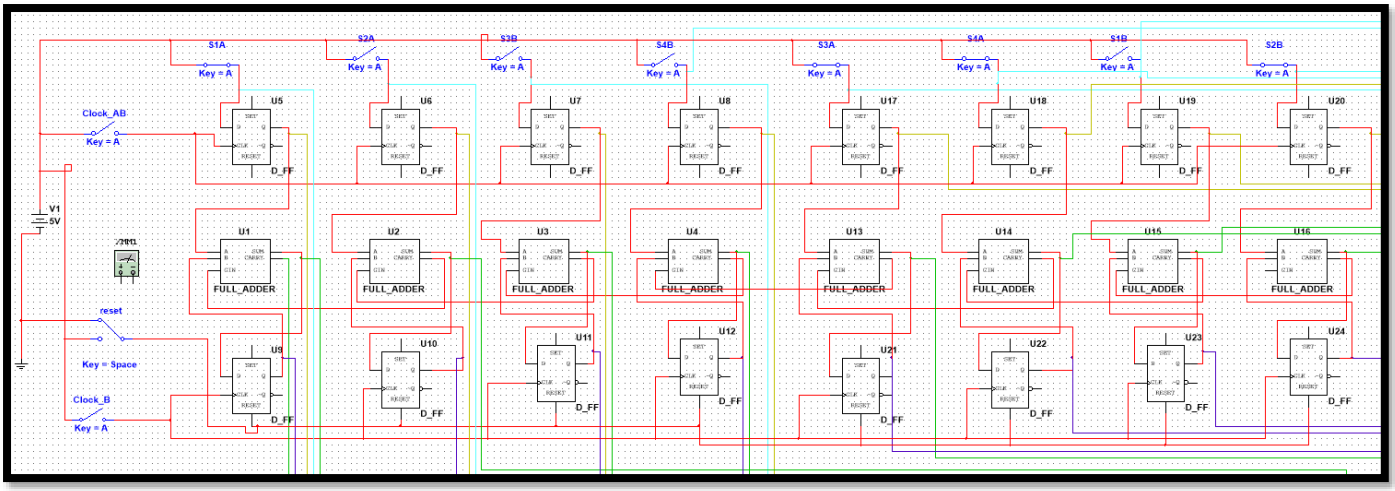


Fig (6.3) [4]

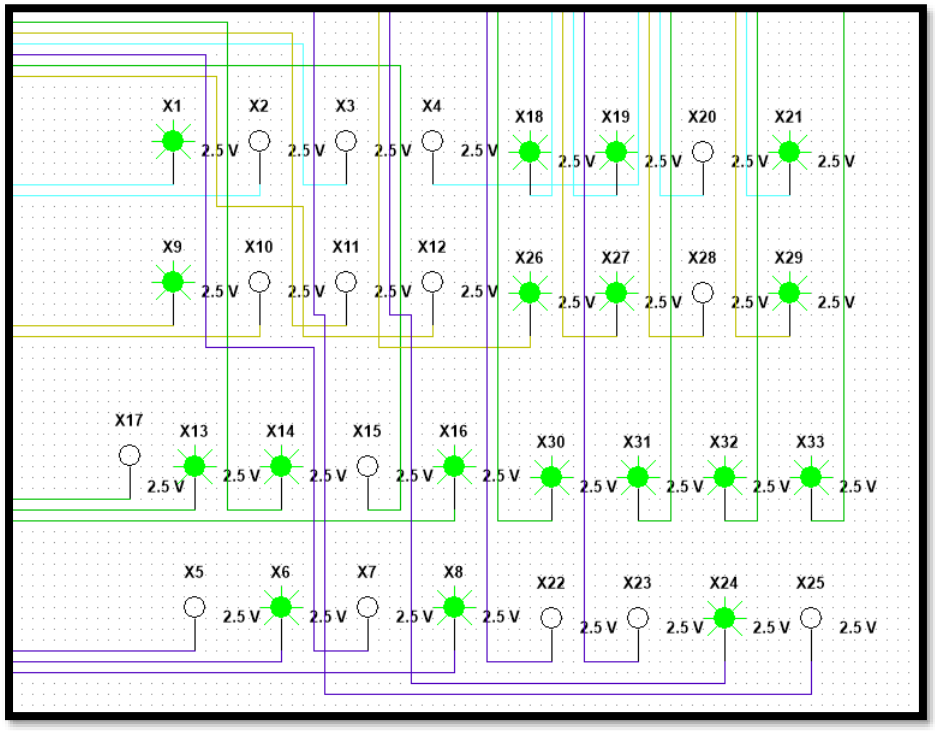


Fig (6.4)

As it can be observed, the 8-bit circuit is just an extension of the 4-bit variation, leading to a much larger circuit that was needed to be split into two screenshots. For this theiry example I will be using the values of…

1 0 0 0 1 1 0 1b = 141
And
0 1 0 1 0 0 1 0b = 82

So if these values were to be added together using the full adders, the results should be…

Inputs/Outputs	Base 2	Base 10
Register A	1 0 0 0 1 1 0 1	141
Register B	0 1 0 1 0 0 1 0	82
Sum	1 1 0 1 1 1 1 1	-
Carry	0 0 0 0 0 0 0 0	-
Output	0 1 1 0 1 1 1 1	223

Tabel 3 [4]

So, the output result would be 223 and '0 1 1 0 1 1 1 1 1b' in its binary value. This can also be proved by adding the two number in the registers together and see if the result is the same.

141 + 82 = 223

These results can also be seen at the output of the circuit, which can be shown above in *Fig (6.3)* and *Fig (6.4)*.

Task 7: - Synchronous Counter

22)

For this task, a simple 6-bit binary counter had to be constructed using D-type flip flops that can increment up to the maximum number of 63 before resetting back down to 0. Ready to restart the count again. The circuit for this can be seen in *Fig (7.1)*

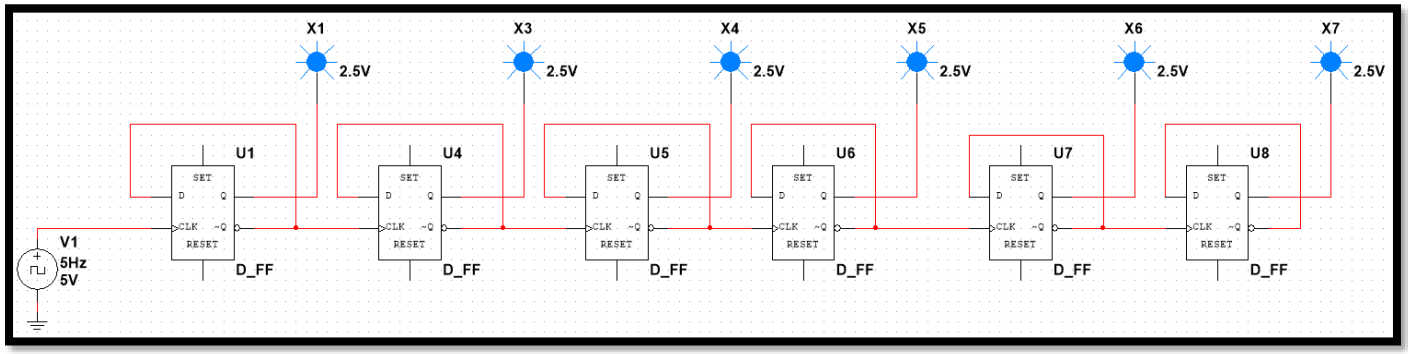


Fig (7.1)

The Theory behind this is that Q-bar will become the input for the first data input, which would then trigger the output of Q to be equal to the value of Q-bar when the clock pulses high. When the clock pulses high the next time, the Q-bar output will rise to logic high again, triggering the clock pulse for the next D-type flip flop. This essentially doubles the clock time for each flip flop, next in the line.

This effectively makes it so the output of each Q, when put in order, can count in a sequence that mimics the counting of a binary number.

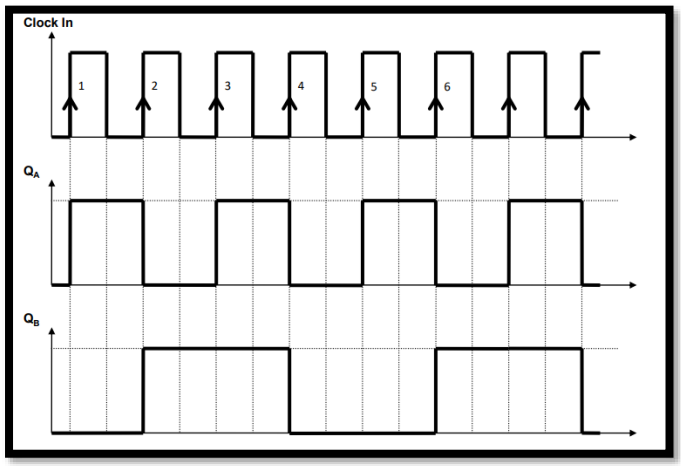


Fig (7.2)

This diagram shows the pulses at the Q output after each clock pulse. As it can be seen, the pulses double in size due to the process of waiting for the next clock pulse to reach that particular flip flop component. [5]

23)

The following task now requires a counter that will reset the circuit and count, when the value of 38 is being displayed at the output. The circuit for this can be seen in *Fig (7.3)*

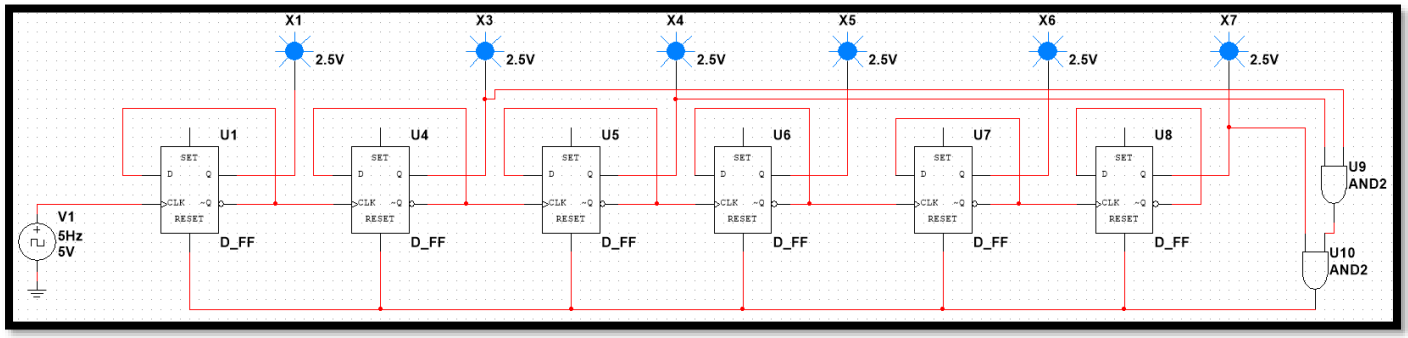


Fig (7.3)

The way to reset the circuit under this condition is to use a series of logical gates that will trigger when the binary value for 38 is being displayed at the output. When this occurs, the Q outputs that are only on at the same time when the count reaches 38, will trigger the logic gates to have a logic output of 1, which will then activate the reset pins for all the flip flops in the circuit. This will return the count back to 0 when the value 38 is displayed. Essentially making 38 equals to 0 because the reset will happen so fast that it won't be noticeable to the naked eye.

24)

This task asks that the count-up circuit resets at 45 to the binary value for 20. This circuit can be seen in *Fig (7.4)* shown below.

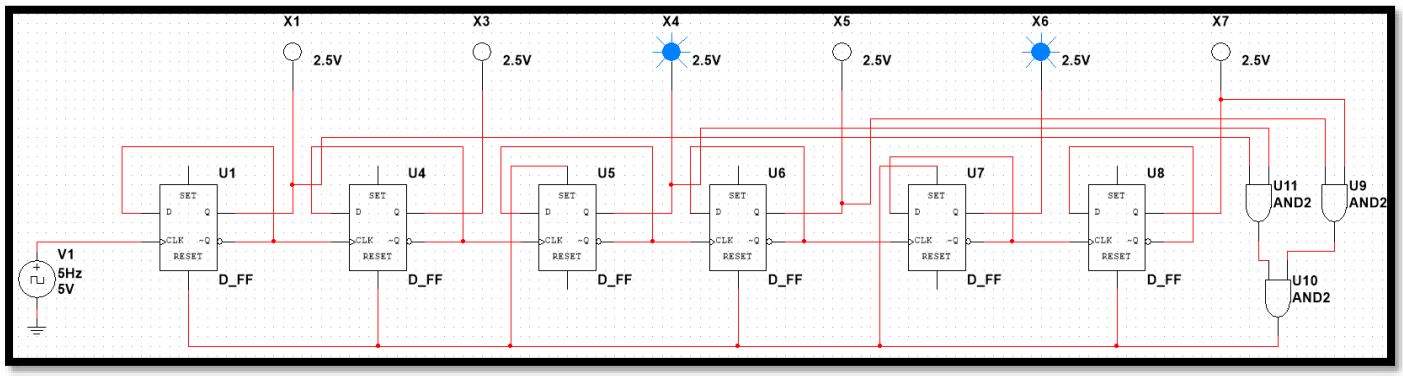


Fig (7.4)

In this circuit, when the value for 45 is displayed in binary, the bits that are only on at the same time when the value of 45 is being displayed. Those bits will lead into a 4 input AND gate which will reset AND set certain flip flop components that will display the value of 20 when the reset occurs, continuing the count again until it reaches the next point that the circuit will reset once more.

25)

The final task requires that the D-type flip flop will count down. Starting from the value of 10, counting down to the binary value for 1. The circuit will reset to the value of 10 every cycle. The circuit for this can be seen below in *Fig (7.5)*

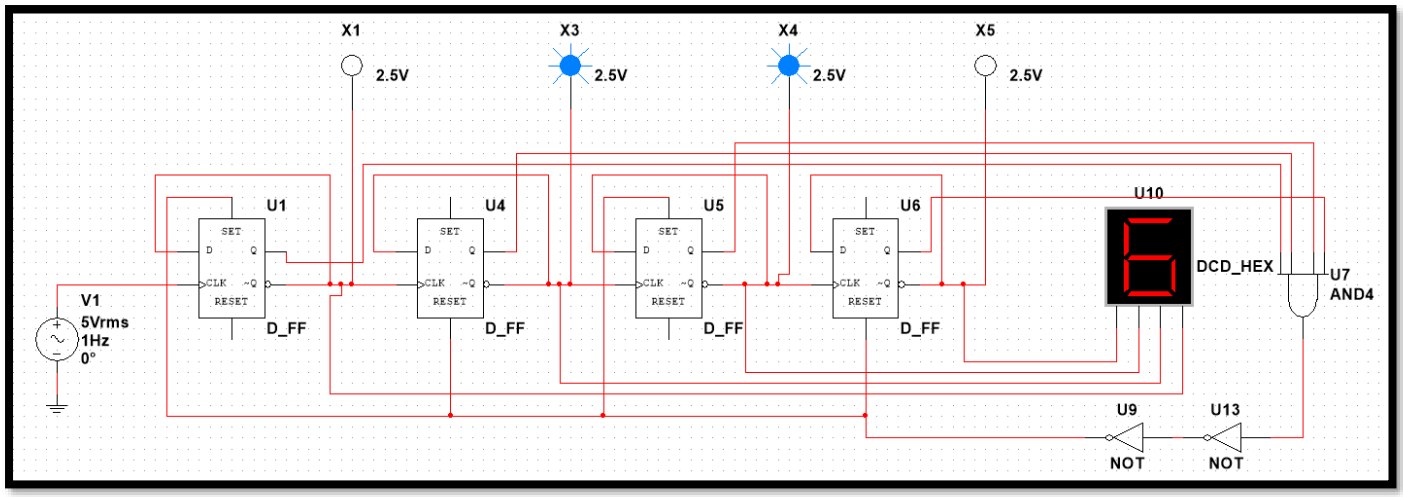


Fig (7.5)

The difference between the count up and the count down circuit is that the value of Q-bar is being displayed instead of Q. This is because since Q-bar is the oposit of Q, this means that the output of Q-bar must be counting down, starting from the highest number. When the value of the count down reaches 0 the 4 input AND gate will check the Q outputs to see if the value of the circuit is equal to the maximum number of 15. This would be the

value of 0 in the Q-bar output. When the gate has been triggered, the flip flop components will be set and reset to so the binary output after the reset will be equal to 10, continuing the cycle.

Conclusion: -

I have been able to successfully simulate these multitude of basic, common circuits. Being able to explain their function and how I was able to manipulate them using a multitude of methods and giving my reasons for doing so. Over this time of building and simulating different circuits, I have been able to become accommodated with the Multisim simulation program, being able to navigate the user interface while also searching for the specific component that I required at the time. I have also been able to effectively use the simulation feature with ease, setting up the circuit so that I can either manually control it using switches, or make it automatically run by itself.

For the first part of the assignment, I was able to calculate the gain specified and requested of me by using the formula for each Op-Amp and using it to decide on the value of the resistors in each circuit. Both circuits worked as intended, amplifying the circuit by the gain specified and inverting the signal when using the inverting variation.

The second task asked for a construction of a half wave and full wave rectifier circuit, which had been completed and screenshots of the results were shown using the Multisim oscilloscope simulation feature. Using this feature, the before and after of both variants of the rectifier, with and without the smoothing capacitors, had been recorded showing the effects that the size of the capacitors had. The larger the capacitor, the smoother the signal wave to put it in simple terms, but when it came to full wave vs half wave, the full wave would come out on top as the most efficient method between the two by using the most that the power supply had to offer, which also could result in the circuit not needing as big of a capacitor as the half wave variation.

The third task requires the testing of different logic gate variations and to display the results. This has been done, showing all the truth tables for them and being able to simulate and display the NAND gate variation of each logic gate, to show that they could be easily constructed using a multitude of NAND gates.

The fourth task requires a simulation of full adder circuits and to show how they operate and make their calculations. I have been able to describe this in full detail how the calculations are made, using binary numbers as examples to make the steps that the circuit would take to make the calculation. I have also been able to back up my theory by simulating a 1-bit, 4-bit and 8-bit adders in Multisim.

The final task requires a simulation of a D-type flip flop counter circuit that will increment up and down in values, while also using logic gates to manipulate the circuit into resetting at certain values and resetting to certain values during its process. The simulations were successful in accomplishing these requirements, allowing for the manipulation of each counter circuit that had been constructed.

In conclusion, I have been able to successfully simulate and run a multitude of circuits using the Multisim software, becoming competent with the software over a short period of time, while also being able to construct simple and common circuits that are used in the construction of everyday electronics. I am also able to give a detailed explanation on how theses circuits operate under certain conditions, while also using these conditions to make the circuit preform in the way that I require it to within reason.

References: -

[1] Fig 5.5-5.11 shows the truth tables for each logic gates – from the lecture 8 and 9 power points (“Lecture 8_NG1S902_2018_2019_Op_Amp_2 and Gates” & “Lecture 9_NG1S902_2018_2019_Logic devices and truth tables”) found in the learning materials for analog and digital on blackboard.

[2] Shows the formula for the gain of the inverting and non-inverting Op-Amp – From the lecture 8 power point (“Lecture 8_NG1S902_2018_2019_ Op_Amp_2 and Gates”) found in the learning materials for analog and digital on blackboard.

[3] collected information about the NAND gate equivalents (Wordpresscom. 2012. Computer Organisation and Architecture. [Online]. [12 April 2019]. Available from: https://lings2mi.wordpress.com/2012/10/22/integrated-circuits/)

[4] Shows the method of how a full adder will calculate adding two numbers together in binary and the construction of a multi bit full adder circuit. (“Lecture_17_NG1S902_2018_2019_Adders.pdf”) found in the learning materials for analog and digital on blackboard.

[5] Shows how flip flop counter circuit displays its values in binary. (“Lecture_16_NG1S902_2018_2019_Counters.pdf”) found in the learning materials for analog and digital on blackboard.

[6] Shows the construction of the half and full wave rectifier circuit and how it operates. (“Lecture 6 _NG1S902_2018_2019_ Rectification.pdf”) found in the learning materials for analog and digital on blackboard.

[7] used an image for the front cover of the assignment. Altrancom. 2019. Altrancom. [Online]. [12 April 2019]. Available from: https://www.altran.com/us/en/industries/industrials-electronics/ & “https://www.altran.com/as-content/uploads/2017/05/2-6_industrials-electronics_v2_share_1200x600.jpg ”