Digital Clock

Written By: Kevin Chen

Objective

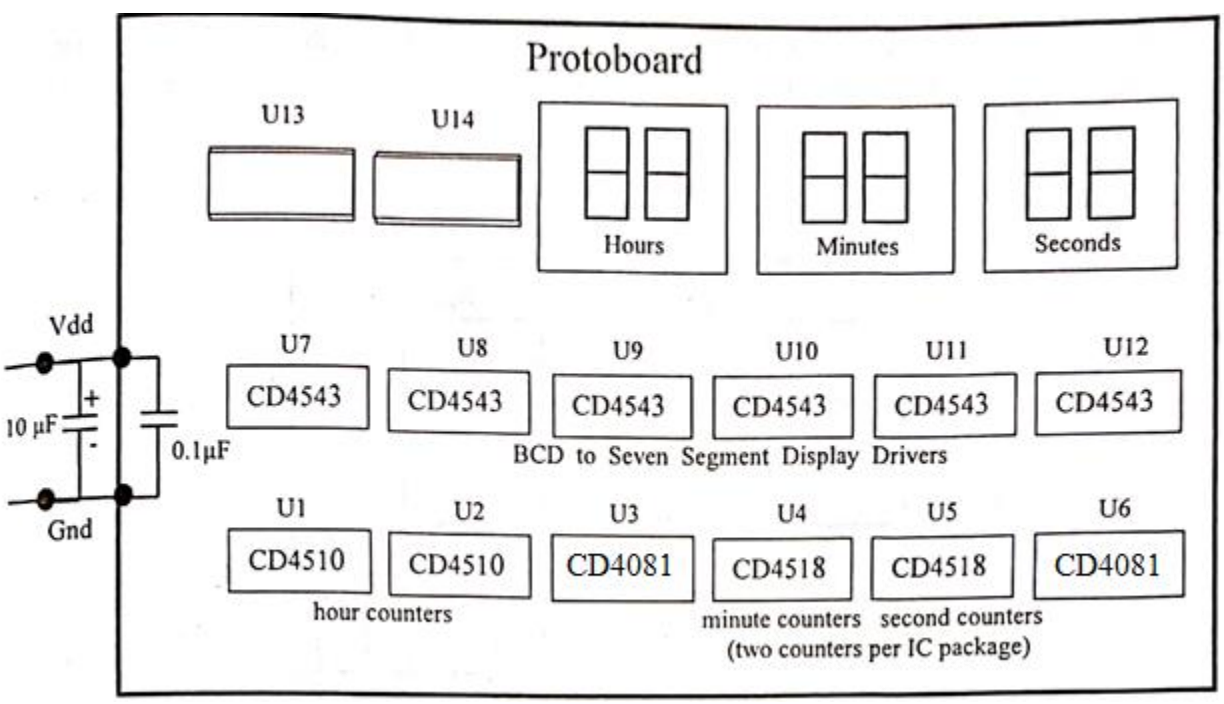
In this current experiment, Digital Clock, our goal is to learn how to use a seven-segment numerical display and various types of digital counters and gates to design a 12-hour digital clock. A seven-segment numerical display consist of 7 lines that will be used to display numbers depending on their inputs. The various types of digital counters and gates consist of a logical that runs inside of them to take in a certain input and outputs a desired output. To test theses devices, we're planning to experiment with them and understand what each part of the component does.

Introduction

To achieve our objective, we need a general knowledge of how circuits work, and knowing the purpose for each part used. We're going to create a 12-hour digital clock using five parts from our clock kit. The first parts are the three two-digit 7 segment display common anode, which are used to display the digits. The second parts are the six CD4543BE CMOS IC, which are the drivers for the seven segments. The third parts are the two CD4510BE CMOS IC, which are binary coded decimals for setting and counting the clock for the hours. The forth parts are the two CD4518BE CMOS IC, which are also binary coded decimals but for the minutes and seconds. The firth parts are the two CD4081BE CMOS IC, which consist of just AND gates. All five of these parts are essential in creating the 12-hour digital clock by how they'll be logically connected.

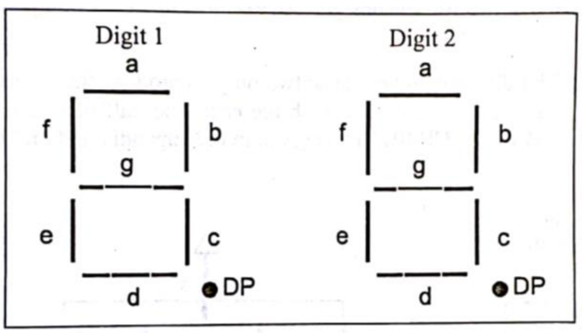
Analysis

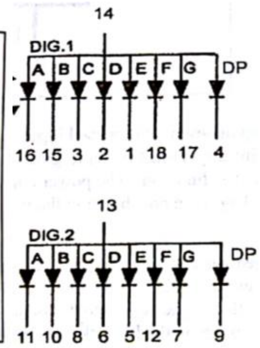
We start off by setting up the protoboard so that the parts of the clock are connected to the protoboard. The design for the setup for the protoboard is shown below:



NOTE: The pin assignments of any circuit part are counted from the bottom left to the bottom right to the top right to the top left in numerical sequence.

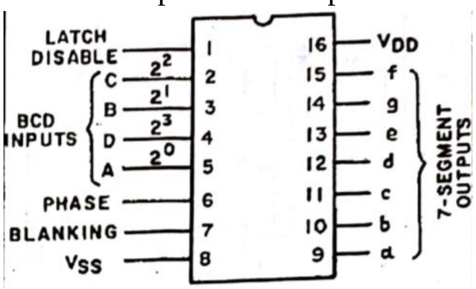
The first part of the experiment is to understand how the two-digit 7 segment display common anode works.





There is a total of 18 pins. Pins 13 and 14 should both be connected to the power source to supply all the segments power. You can make a certain segment light up by connecting the ground to any of the other pins.

The second part of the experiment is to understand how the CD4543BE CMOS IC works.



There is a total of 16 pins. We connect pin 9-15 to the corresponding 7-segments, making sure each letter is connected to each other. We connect pin 16, V_{DD}, to the power source and pin 8, V_{ss}, to the ground. In addition, we also connect pin 1 and 6, Latch Disable and Phase, to the power source, and pin 7, Blanking, to ground.

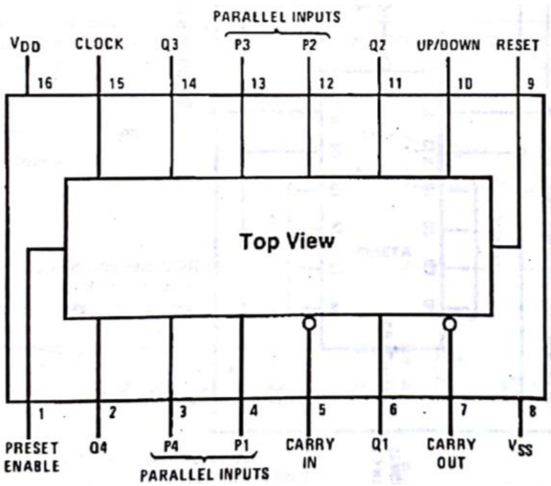
The reason for these connections are because of this truth table:

TRUTH TABLE FOR CD4543B

INPUT CODE							OUTPUT STATE							DISPLAY CHARACTER
LD	BI	Ph*	D	C	B	A	a	b	c	d	e	f	g	
X	1	0	X	X	X	X	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	1	1	1	1	1	1	0	0
1	0	0	0	0	0	1	0	1	1	0	0	0	0	1
1	0	0	0	0	1	0	1	1	0	1	1	0	1	2
1	0	0	0	0	1	1	1	1	1	1	0	0	1	3
1	0	0	0	1	0	0	0	1	1	0	0	1	1	4
1	0	0	0	1	0	1	1	0	1	1	0	1	1	5
1	0	0	0	1	1	0	1	0	1	1	1	1	1	6
1	0	0	0	1	1	1	1	1	1	0	0	0	0	7
1	0	0	1	0	0	0	1	1	1	1	1	1	1	8
1	0	0	1	0	0	1	1	1	1	1	0	1	1	9
1	0	0	1	0	1	0	0	0	0	0	0	0	0	Blank
1	0	0	1	0	1	1	0	0	0	0	0	0	0	Blank
1	0	0	1	1	0	0	0	0	0	0	0	0	0	Blank
1	0	0	1	1	0	1	0	0	0	0	0	0	0	Blank
1	0	0	1	1	1	0	0	0	0	0	0	0	0	Blank
1	0	0	1	1	1	1	0	0	0	0	0	0	0	Blank
0	0	0	X	X	X	X	**							**
†	†	1	†				Inverse of Output Combinations Above							Display as above

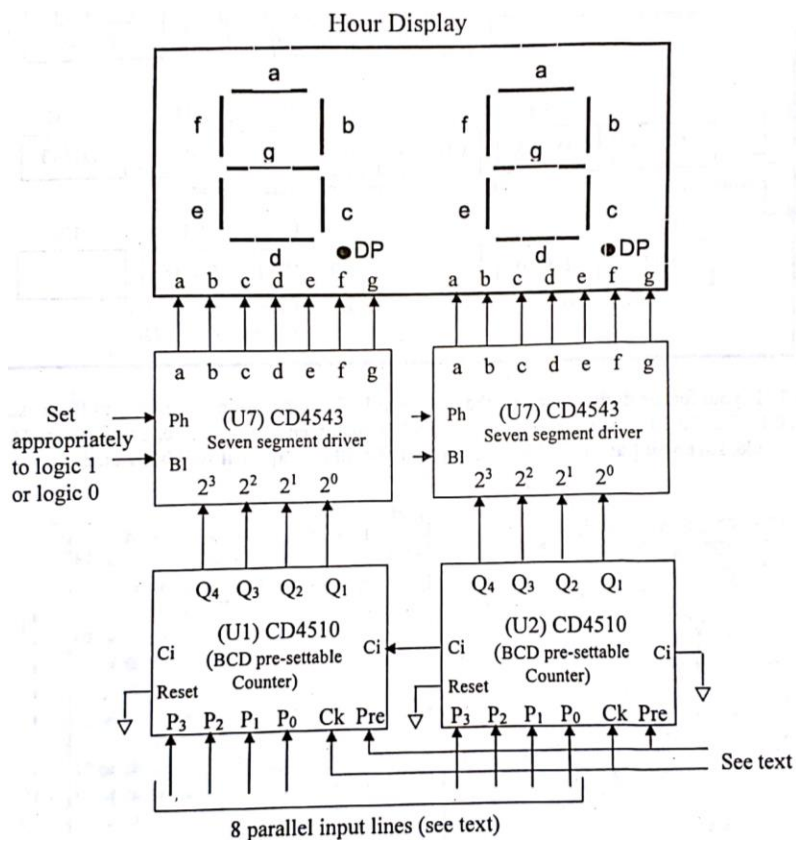
X=Don't care.
†=Above combinations.
*=For liquid-crystal readouts, apply a square wave to Ph.
For common cathode LED readouts, select Ph=0.
For common anode LED readouts, select Ph=1.
**=Depends upon the BCD code previously applied when LD=1.

The third part of the experiment is to understand how the CD4510BE CMOS IC works.

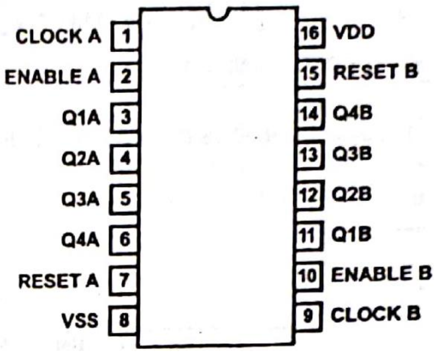


There is a total of 16 pins. This CMOS IC is specifically for the hour counter of the clock. We're going to need two of these, one for each digit. We start off by connecting pin 16 and 10, V_{DD} and Up/Down, to the power source to make sure power is supplied and it counts. We connect pin 8, V_{SS}, to ground. We then connect pin 2, 6, 11, and 14, the Q's, each to the corresponding BCD inputs on the CD4543BE CMOS IC. For example, Q1 (pin 6) goes to A (pin 5). The parallel inputs, pins 3, 4, 12, and 13, the P's, basically forces an input onto the Q's when pin 1, preset enable, is turned on.

Pins 5 and 7, carry in and carry out, are basically for transitioning the digits over; we're supposed to connect the first digit's carry in to the second digit's carry out. This way, when the second digits reaches 9 then back to 0, it carries out a 1 to add to the carry in of the first digit; this is how it counts. Since the first digit doesn't have a carry out, and the second digit doesn't have a carry in, we just set it to ground. For pin 15, clock, we connect them to the same generator that's able to change its value, this way the two CMOS IC follows the same signal to change values; they're used for the counting part. For pin 9, reset, it's used to reset the clock back to 0, when reset is 1. The design for connections should look like the image below:

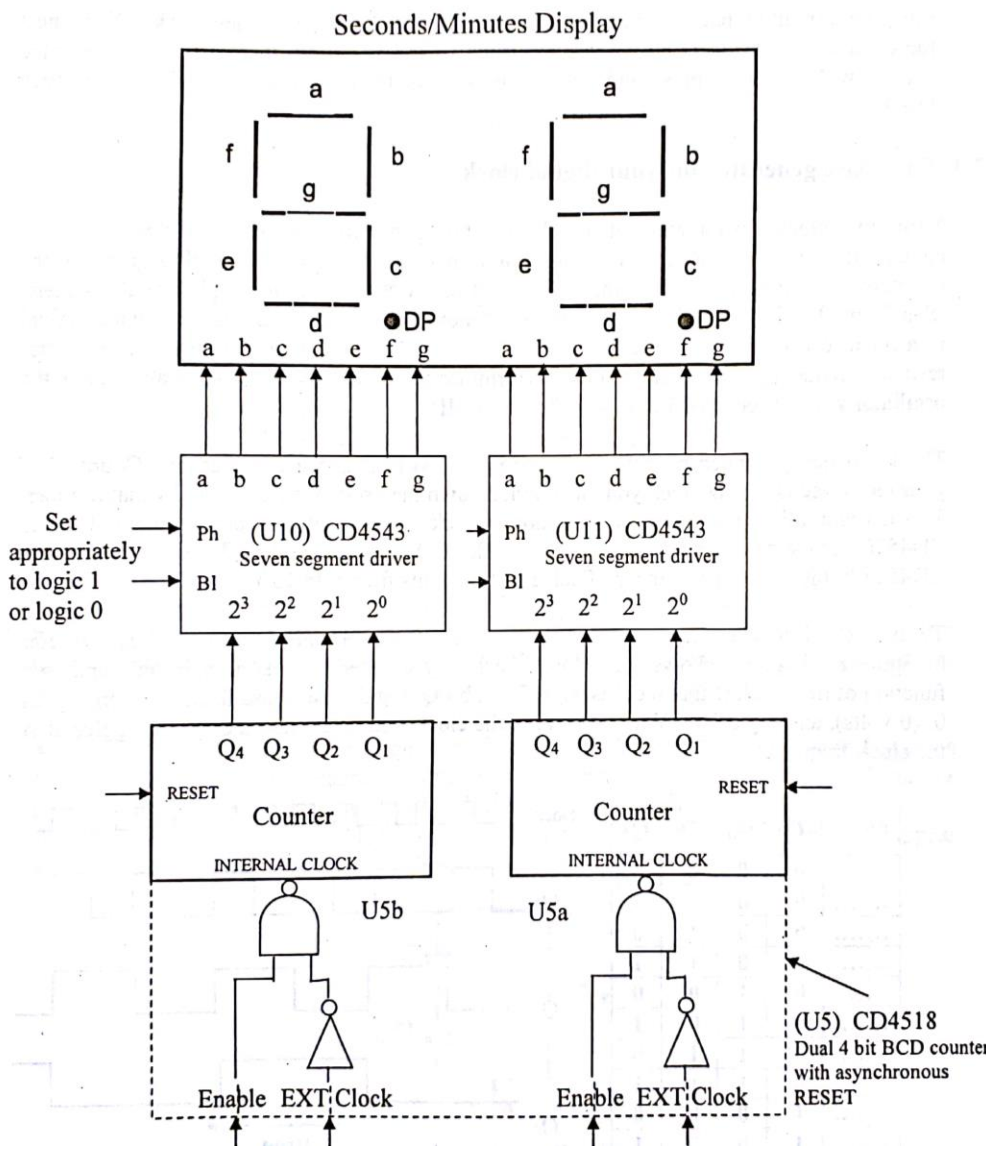


The fourth part of the experiment is to understand how the CD4518BE CMOS IC works.

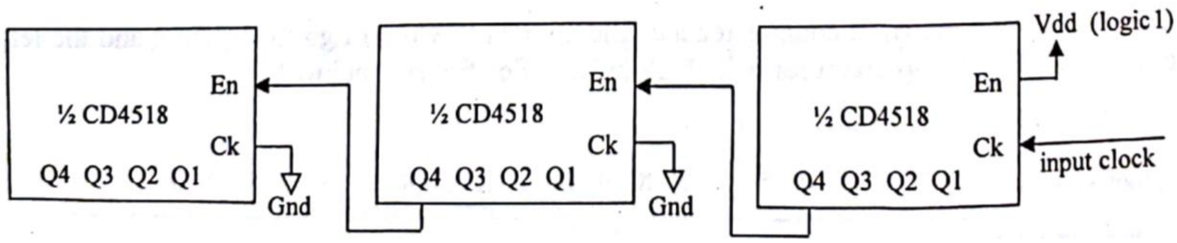


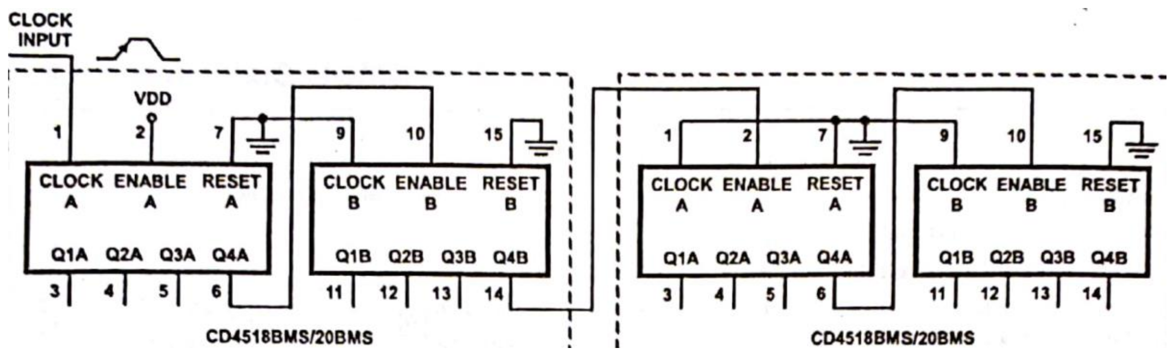
There is a total of 16 pins. This CMOS IC is specifically for the minutes/seconds counter of the clock. We're going to need two of these, one for each two digits. We start off by connecting pin 16, V_{DD}, to the power source to make sure power is supplied. We connect pin 8, V_{SS}, to ground. We then connect pins 3, 4, 5, 6, the Q's, each to the corresponding BCD inputs on the CD4543BE CMOS IC. We do the same for pins 11, 12, 13, and 14, the Q's.

Pins 1, and 9, clocks, are used for counting and their inputs depends on what it's being used for. Pins 2, and 10, enables, are used to turn on or trigger an input. Pins 7, and 15, resets, are used to reset the clock back to 0, when reset is 1. The design for connections should look like the image below:

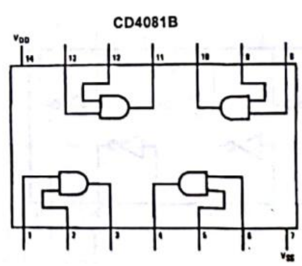


The design for the connection between the CMOS IC are shown below:



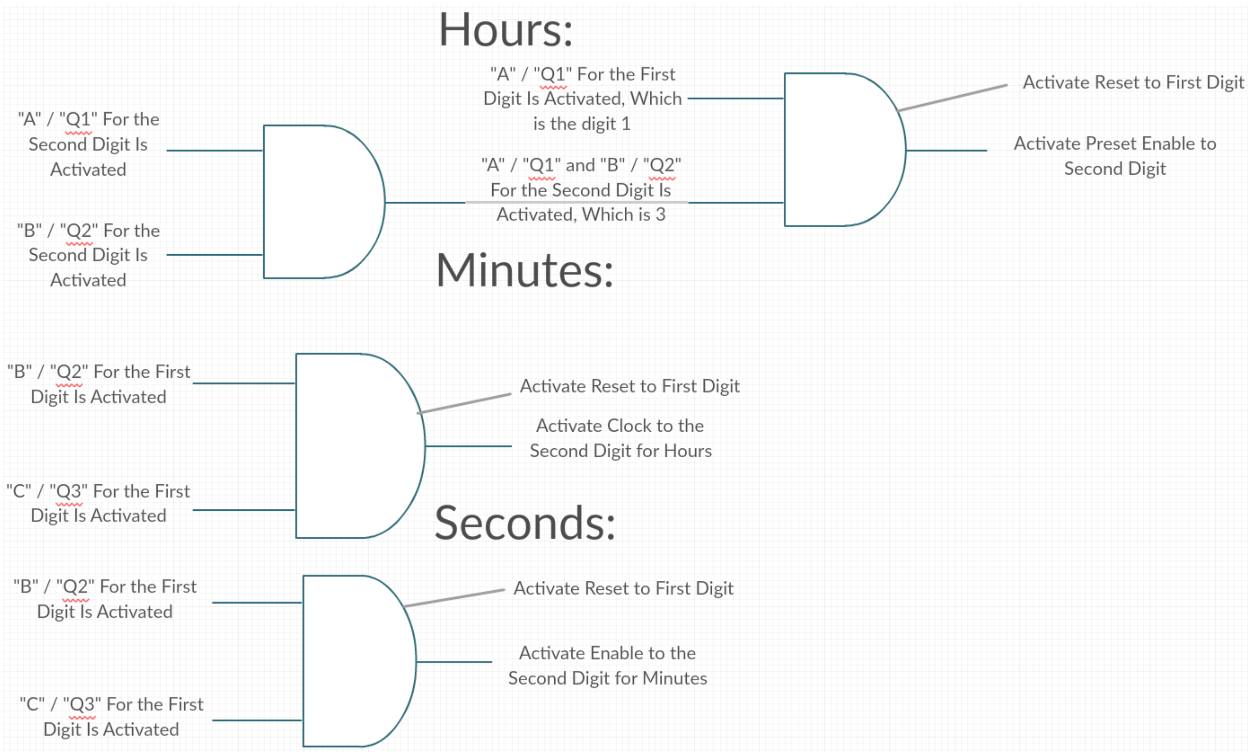


The fifth part of the experiment is to understand how the CD4081BE CMOS IC works.



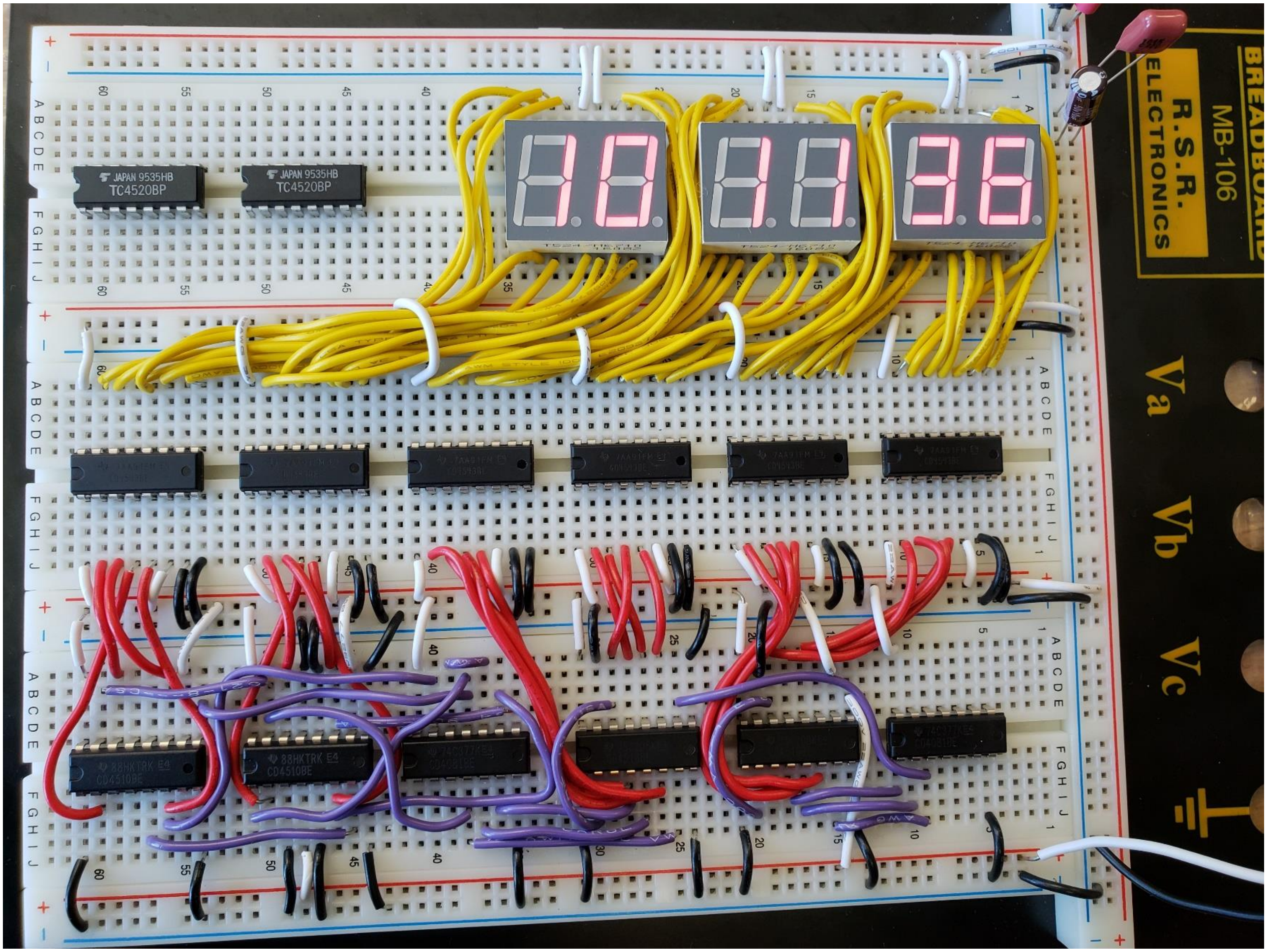
There is a total of 14 pins. We start off by connecting pin 16, V_{DD}, to the power source to make sure power is supplied. We connect pin 8, V_{SS}, to ground. Pins 1, 2, 5, 6, 8, 9, 12, and 13, inputs, all work the same way. They take the input and logically display an output. The outputs are pins 3, 4, 10, and 11. The output will only be 1 if both inputs are 1. This is important for the logical for the resets.

Now that we understand how the parts work. We must understand how the logical of the clock works. The logic that uses the CD4081BE CMOS IC would be to reset and to count when the seconds and minutes reaches 60 seconds/minutes. In addition, when the hour reaches 13 hours, it would also reset it, so it displays 01 hours. The 1 digit, is set by using the preset part of the CD4510BE CMOS IC and it's to enable it when it reaches 13 hours. The logic of the "AND" gates are shown below:



After knowing how the logic of the clock works, just connecting the wires to it's appropriate location and turning on the DC power supply and function generator would allow the 12-hour digital clock to start working. Make sure the DC power supply supplies at least 5 V. In addition, the settings for the function generator should be in a square wave, with a frequency of 1 Hz, for it to count in seconds, and having a maximum of 5 V and a minimum of 0 V.

Pictures of the 12-hour digital clock is shown below:



Key:

The yellow wires represent the 7 segments connections.

The red wires represent the BCD connections.

The white wires represent the connection to voltage.

The black wires represent the connection to ground.

The purple wires represent the connection to the logic gates and to the clock operations.

Questions

1. *An explanation of how you made the counters count to the correct numbers.*

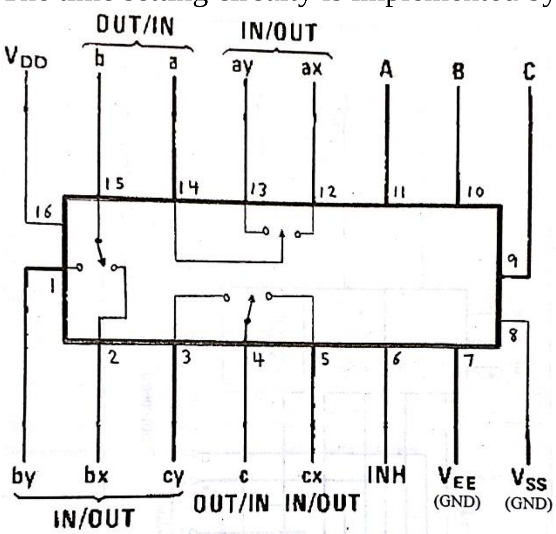
I made the counters count to the correct numbers by the connections from the BCD inputs to the 7-segments and the clock signal only going through the seconds clock. In addition, to get the correct numbers I want it, I used the “AND” gates, so that it would activate the reset and the preset when the conditions were met. The conditions were that if the seconds or minutes reached 60, it would reset it back to 00 and add a value to the next one over. The hours conditions were that if it reached 13, it would reset the 1 to 0 and it would preset the 3 into 1 so it would become 01.

2. *You will describe the display and how it works.*

The display can be described as 7 segments that are lit up based on which pins are active. When connecting a voltage source to pin 13 and 14 and connecting the ground to any of the other pins, will light up a certain segment and how’s how the display works. Using a BCD and connecting it to the 7 segments would follow a code that’s able to light up multiple segments depending on their inputs.

3. *Include a description of the time setting circuitry. Discuss the switch bounce problem and possible solutions.*

The time setting circuity is implemented by using a CD4053BE CMOS IC as shown below:



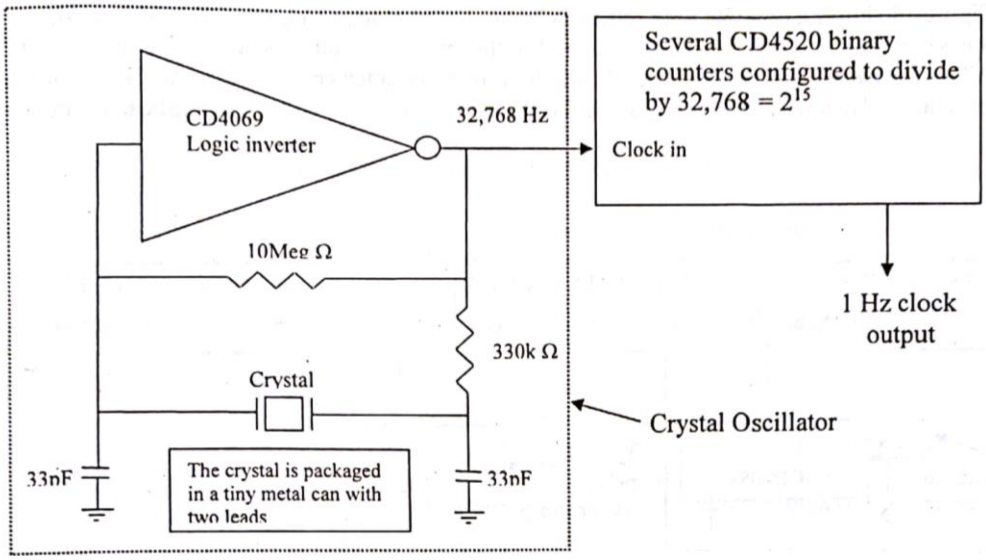
It uses a triple 2-channel analog multiplexer and the truth table is shown below. Its purpose is to set the time for the hours, minutes and seconds by increasing its increment when the button is pressed.

The switch bounce problem is when the clock receives multiple counts as inputs as opposed to one. A possible solution to this problem is by adding in resistors and capacitors to prevent the additional counts from interfering.

INPUT STATES				"ON" CHANNELS		
INHIBIT	C	B	A	CD4051B	CD4052B	CD4053B
0	0	0	0			cx, bx, ax
0	0	0	1			cx, bx, ay
0	0	1	0			cx, by, ax
0	0	1	1			cx, by, ay
0	1	0	0			cy, bx, ax
0	1	0	1			cy, bx, ay
0	1	1	0			cy, by, ax
0	1	1	1			cy, by, ay
1	•	•	•			NONE

4. *Describe the time base generator.*

The time base generator is described by the image below:



It uses capacitors, crystal, resistors, and a logic inverter so that the output will be a 1 Hz clock. How it works is that the voltage will go in and it will convert the 32,768 Hz that the crystal must 1 Hz by the logic inverter and that will supply the clock to count seconds. It acts as a function generator, so batteries could power the clock.

5. *Tell your reader how long a 9 V battery or 6 AA batteries will run your clock. Discuss your findings about how much current the logic portion of the clock draws compared to the displays. Do you know of another type of display that draws much less current then the display that you used?*

The 12-hour digital clock that was constructed needed about 5 volts that needs 0.1 amps to be supplied to stay lit up. A 9-volt battery supply will only last about 5 hours because a typical 9-volt battery has a capacity of about 500 mAh and the digital clock needs about 100 mA so $\frac{500}{100} = 5\ hours$. 6 AA batteries will last about 144 hours because 1 AA batteries has a capacity of about 2400 mAh so 6 of those will have 14400 mAh and the digital clock needs about 100 mA so $\frac{14400}{100} = 144\ hours$. The logic portion of the clock draws a lot less current compared to the displays because of how the display needs more current to show the brightness of the segments. We are unaware of another type of display that draws much less current than the display we used.

Conclusion

The goal of this experiment to learn how to use a seven-segment numerical display and various types of digital counters and gates to design a 12-hour digital clock. We plan to achieve this goal by understanding the purpose for each of the parts and how to logically connect them with each other. There was a total of five parts that was used, two-digit 7 segment display common anode, CD4543BE CMOS IC, CD4510BE CMOS IC, CD4518BE CMOS IC, and CD4081BE CMOS IC; these five parts are all essential in making the 12-hour digital clock operational. We started by attaching the parts onto the protoboard in a specific order so that they're easier to work with. We later understood how the pins of the parts work and connected them. We then used the logic gates to configure the clock to reset at 60 seconds/minutes and presetting the 13 hours to 01 hours when it reaches past 12 hours. The outcome came out to show that the clock was fully functional and was able to count correctly like how a 12-hour digital clock supposed to be. The efficacy of the procedures used to obtain our clock was a bit confusing and wasn't as detailed. The procedures from the lab manual showed some unnecessary information and didn't show how each part connects or description of what each part works. There wasn't any limitation in this experiment because the clock was fully functional without any problems, and none of the parts were defected. A possible method of reperforming this experiment would be to reconstruct the clock and see if there's any possible ways to improve it. For example, reconfiguring some of the wires to save space or just using fewer gates.