Design Report
1904971

Table of Contents

1. Abstract

Arithmetic is built into all computers, calculators, banking systems and other digital systems. In relation to banking, the BCD numbering system is often applied for financial transactions in order to minimise the potential of any error between binary to decimal numbers or vice-versa. This paper outlays the processing details needed in order to design a sub-system that can be utilized to multiply Binary Coded Decimal(BCD) numbers. Combinational Multiplier, Plus 3 Adder, Binary to BCD Converter, Seven Segment Driver, Clock Divider are the building blocks of a BCD Multiplier.

2. Introduction

BCD Multipliers, are to multiply BCD numbers. The combinational multiplier is to multiply two binary numbers to give a product. The combinational multiplier contains a 4 bit adder with sums up the partial products to give a final product. The results of a BCD multiplier should be in BCD therefore there is need for a Binary to BCD Converter. This converter includes the product input, Plus 3 Adder and the outputs. There is a clock input from the Clock Divider influences the input to the cathodes in the Seven Segment Driver. The simulation was executed using NI MultiSim 14.2 and implemented on a Field Programmable Gate Array (FPGA) Board.

3. Total Circuit

Figure A(10) (University of Pennsylvania, 2007)

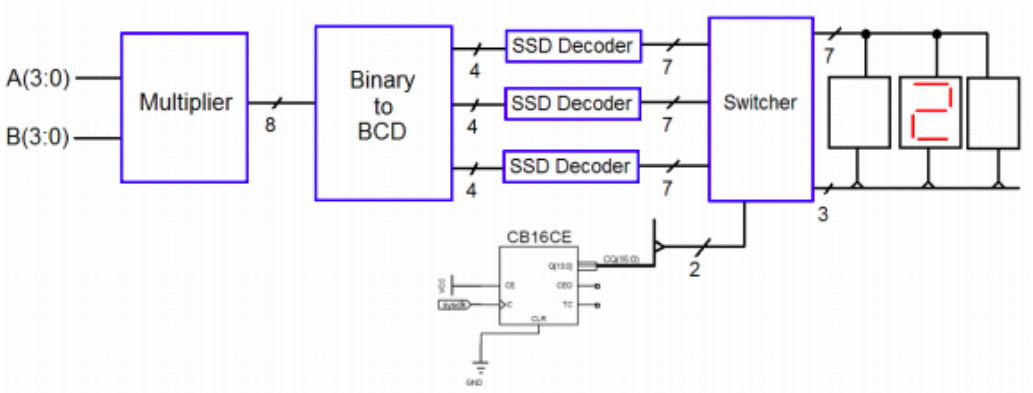


Figure A shows the idea of a Total circuit which consists of the two 4-bit binary numbers, the Combinational Multiplier, the 8-bit output from the multiplier, the Binary to BCD Converter, the Seven Segment Driver, and the Clock Divider. This circuit is done so the final output will be shown on the Seven Segment Display. The way for the values to show is for the digits to switch between each other so one will be displayed after the other. Depending on the clock input, inputs from the ones, tens or the hundreds will be taken and one of all those inputs will be passed to the output. There will be four outputs which will be inputs to the cathodes. The frequency flowing is 100MHz which is too fast therefore the clock divider system will get it reduced to 500Hz.

4. BCD Multiplier

4.1 Binary Multiplier to BCD Converter

Figure B(10) (University of Pennsylvania, 2007)

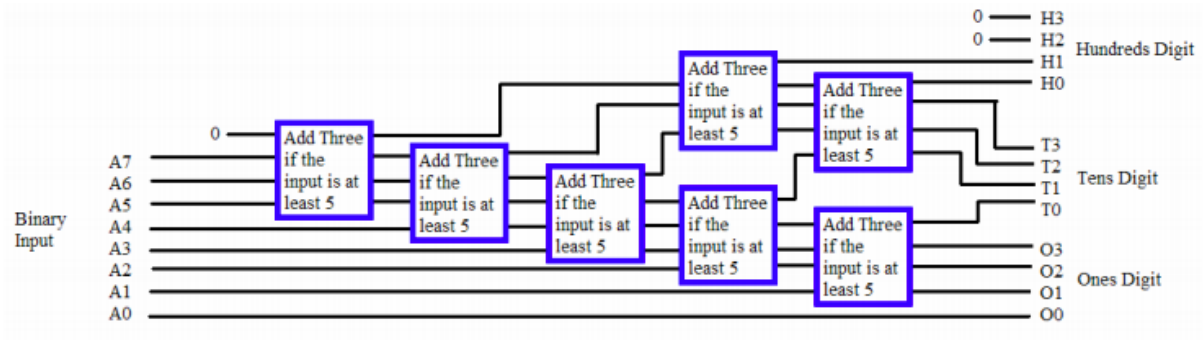


Figure B shows a Binary to BCD Converter. The binary input is a hierarchical block containing a combinational multiplier. The combinational multiplier multiplies two four bit binary numbers to produce an 8-bit binary output. The Least Significant Bit (A0) of the 8 bit-binary output is connected to an output(P0). The next Least Significant Bit(A1) output will be one of four inputs to one of the shift 3 adders which adds three to an input if input is greater than 4, then the output of that shift 3 adder is connected to an output (P1). The next Least Significant Bit(A2) output will be one of four inputs to another one of the shift 3 adders which adds three to an input if input is greater than 4, then the output of that shift 3 adder is connected to the first Plus 3 adder mentioned, then the output of that plus 3 adder is connected to an output (P2). This process goes on as shown in Figure B.

4.2 Combinational Multiplier

This combinational multiplier multiplies two 4-bit binary digits. One of the binary numbers is called multiplicand and the other is called a multiplier. The bit size of the product is the addition of the bit size of the multiplicand and multiplier. For example: two 4x4 bits multiplied will give a product with an 8 size bit. There are two steps to this multiplication and the first step is the multiplication of the two binary numbers which will give rise to a partial product and the second step is the addition of the partial products to give the final product with the use of a BCD adder. Figure 1 shows an example using numbers 1101 x 1110.

Figure 1

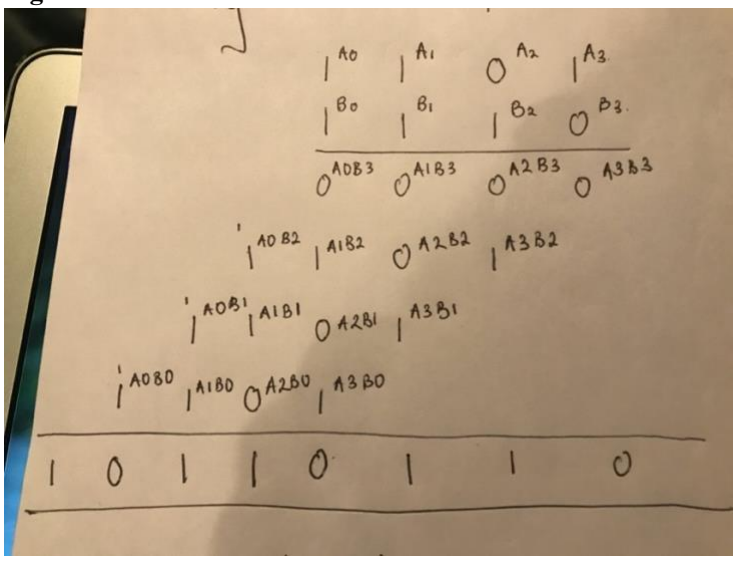
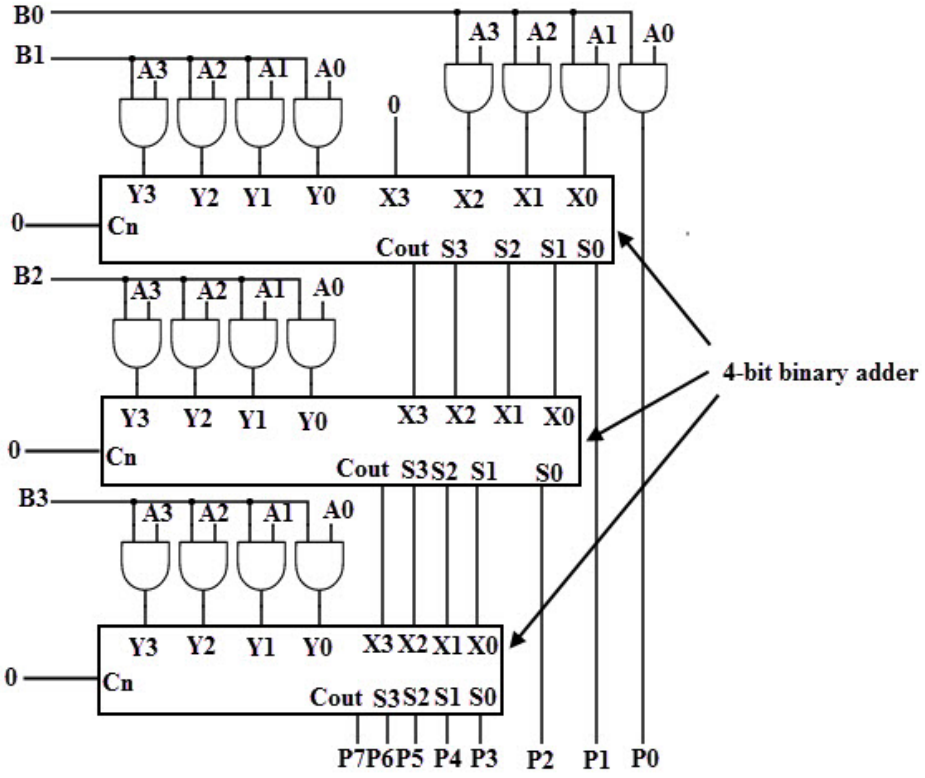


Figure 2 shows how the combinational multiplier is implemented into a schematic. With X(0) and Y(0) being the least bit inputs and P(0) being the least bit output. In the design implemented on NI Multism, A(0) and B(0) are the least bit inputs and P(0) is the least bit output as requested by the tutor.

Figure 2



(1) (Binary Multiplication Methods, 2015)

A Four bit Full adder with Fast Carry has the purpose of adding two 4-bit binary numbers. This adder was used to receive both 4-bit binary numbers (A0,A1,A2,A3,B0,B1,B2,B3) and the Carry input which will always be connected to a digital low. There will be four Sum Outputs from these inputs (S0,S1,S2,S3) and a Carry Output from the fourth bit. S0 is the least significant bit in the outputs and the Carry Output (co) is the most significant bit. Table 1 shows a function table in relation to Figure 2. A The Least Significant Bit(A0B0) of the first partial product is the Least Significant bit of the product so it goes directly to an output. The Least Significant Bit of the sum of the partial products from each adder is the sum of each adder is used as a bit connected to an output, and the other bits from the sum are added to the next partial products. (4) (Binary Multiplier - Types & Binary Multiplication Calculator, n.d.)
Numbers from 0 to 9 are taken into consideration so the 4-bit adder is manipulated to reduce the number of additions.

Table 1(2) (ADD_FULL4_FCR, 2017)

	Cn	X3	X2	X1	X0	Y3	Y2	Y1	Y0	S3	S2	S1	S0	Co
Logic Levels	L	L	H	L	H	H	L	L	H	H	H	L	L	H
Active High	0	0	1	0	1	1	0	0	1	1	1	0	0	1
Active Low	1	1	0	1	0	0	1	1	0	0	0	1	1	0

The time constraint for these Vivado simulations were ran at 3200ns. The usual 1000ns that is the default setting for simulations limits the frequency the circuit could run for due to the sub-systems such as the combinational circuit, adder and other circuits more time is needed. Too much time can be a problem too as the simulation will take too long to run.

VIVADO OUTPUT Figure 4

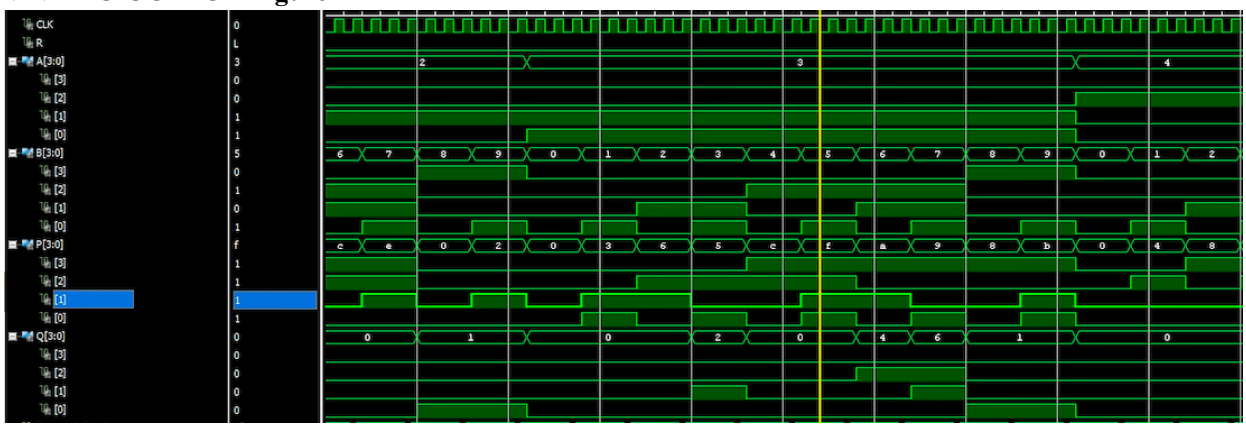


Figure 4 shows the Vivado output for the Combinational Multiplier. The input A is 0011 and input B is 0101 which when multiplied together give 1111 (F). In decimal this will be 3*5 = 15 which is correct. Reading the output, you read in order of Q3,Q2,Q1,Q0,P3,P2,P1,P0 as P0 is the least significant bit.

Figure 5

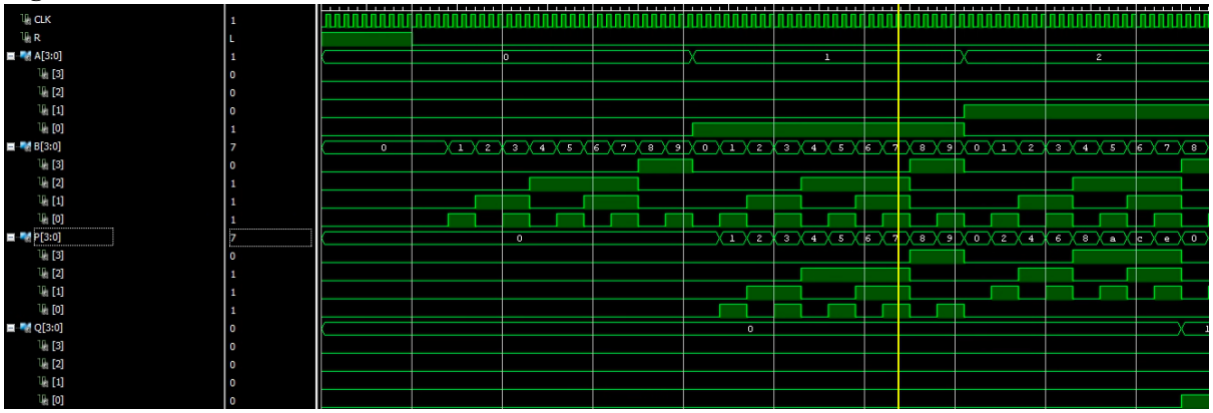


Figure 5 shows the Vivado output for the Combinational Multiplier. The input A is 0001 and input B is 0111 which when multiplied together give 0111. In decimal this will be 1*7 = 7 which is correct. Figures 4 and 5 showing that the combinational multipliers work when cursor placed on a random point on the page.

4.3 Flip-Flop

A Flip-Flop stores a single bit of data. It is a circuit that changes state based on signals from an input. This result to an output that depends on it's present input and present state. (9) (WHAT ARE FLIP FLOPS?, n.d.)

This Flip-Flop changes it's signals on the rising edge of the clock. The Load and Valid signals are the signals. The Flip-Flop takes load and provides a valid signal for the circuit.

Figure 6

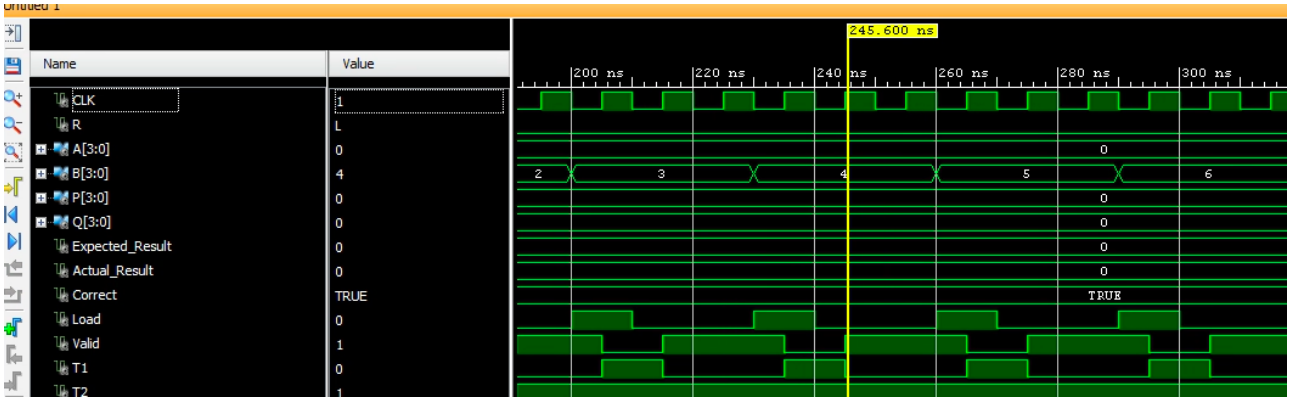


Figure 6 shows that at 245.6ns, the rising edge of the clock, the Load signal changed to a 1 and the Valid signal changed to a 0 indicating that this Flip-Flop works.

4.4 Plus 3 Adder

Figure 7

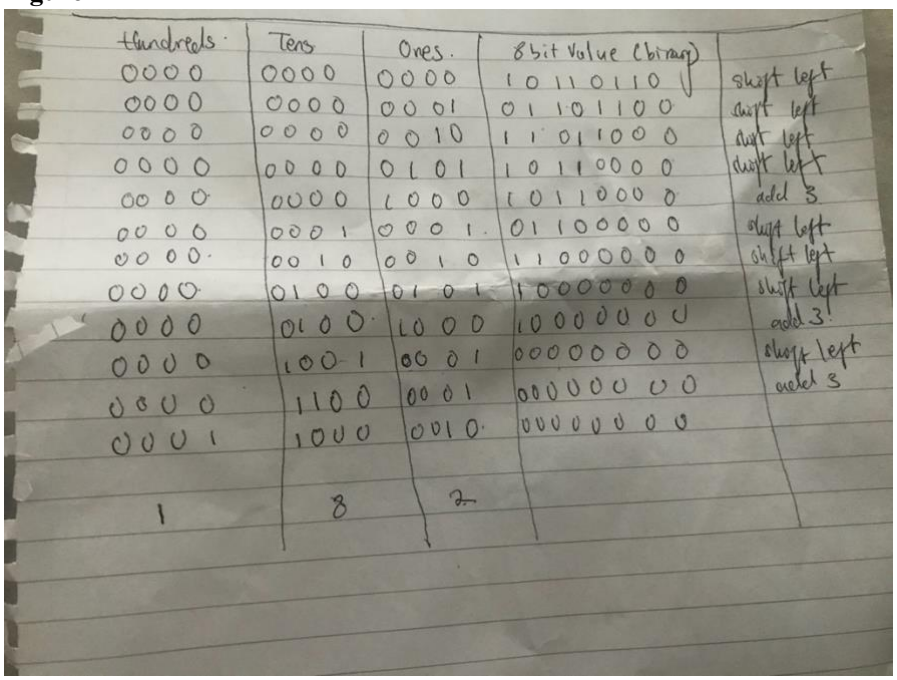


Figure 7 above shows the process of a Plus 3 adder. Explaining the shift register processing: There is an 8 bit binary value and there sections for “ones”, the “tens” and the “hundreds”. The 8 bit binary value represents the value of 182 in decimal and this value is put into a shift register from 19 through 0 which is a 20 bit wide register. The aim per line is to check if one of the hundreds, tens, or ones is greater than or equal to 5, if it is add 3 and shift left. At the start the ones, tens, and hundreds are all initialised to 0. The left shift process happens by adding a 0 to the right side of the 8 bit value. After that there row three more left shifts follow. The fourth line gives 101 in the ones section. In decimal 101 is five which is great than 4 and has met the condition to add 3. Plus 3 is added to 101(5 in decimal) which gives 1000(8 in decimal). The other sections are left untouched when doing this. There is a left shift 3 more times, and the ones section gives 101(5 in decimal), plus 3 is added to 101(5 in decimal) which gives 1000(8 in decimal). There is a left shift in the next line which gives 1001(9 in decimal) and plus 3 is added which gives 1100(12 in decimal). Then a left shift happens one more time because this is the shift of the last digit in the 8 bit value. In the “Hundreds” section the final value is 1(binary) = 1 (in decimal), in the “Tens” section the final value is 1000(binary) = 8(in decimal), in the “ones” section the final value is 10(binary) = 2 (in decimal). In conclusion the result is 182 and this answer corresponds with the decimal value which is 182.

There are ten decimal digits 0-9 and they have are ten different binary representations. Four bits can represent sixteen individual codes, so six codes are not used.

Table 2

A	B	C	D	Output 3	Output 2	Output 1	Output 0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	NA	NA	NA	NA
1	0	1	1	NA	NA	NA	NA
1	1	0	0	NA	NA	NA	NA
1	1	0	1	NA	NA	NA	NA
1	1	1	0	NA	NA	NA	NA
1	1	1	1	NA	NA	NA	NA

This table 2 shows that when the input is more than 4, Plus 3 should be added.

Figure 8

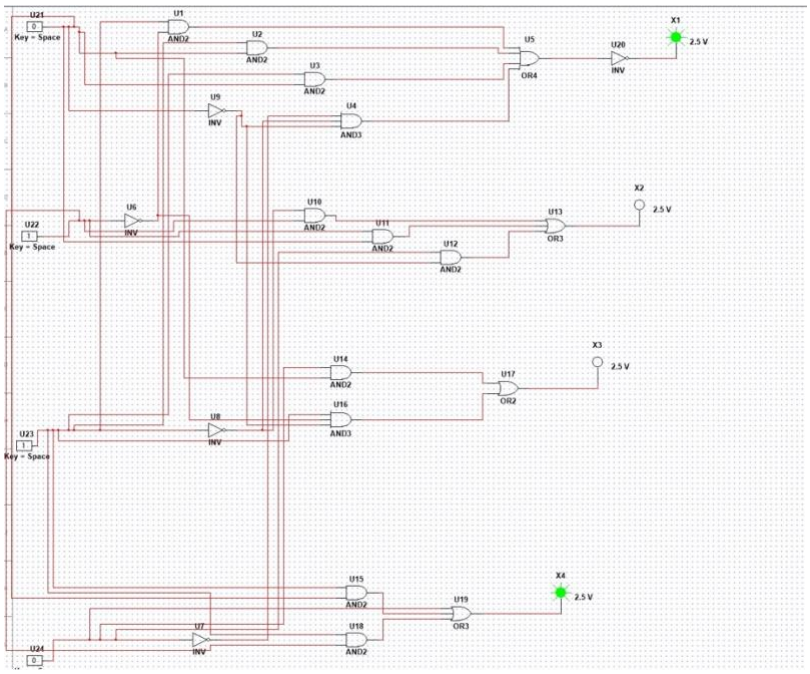


Figure 9

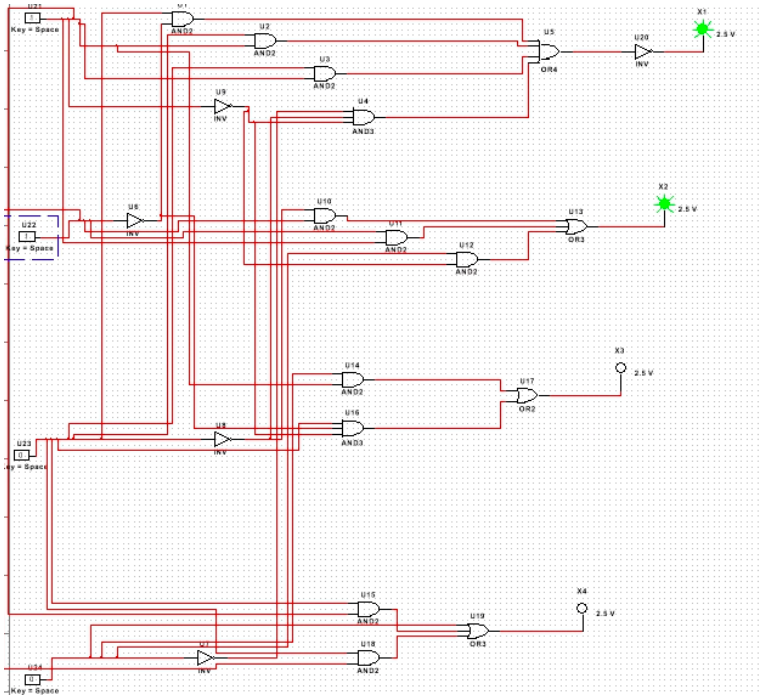


Figure 8 and Figure 9 show the adder schematic done in NI Multism. These pictures show that the adder is working correctly and the probes light up when output is 1. Figure 8 shows the output when the input value is 0110 which gives an output of 1001 as the first and fourth bulb light up. Figure 9 shows the output when the input value is 0011 which gives the same output of 0011 as 0011 is not greater than four. The first and second bulb light up.

4.5 Seven Segment Driver

LEDs are diodes that emit light when current is passed through it in form of photons. Seven LEDs which are each referred to as segments are used in the Seven Segment Display. These segments are labelled from letters a to g and it looks like the figure 11 below

Figure 11(6) (Arduino and Seven Segment Display-Interfacing, n.d.)

The BCD to Seven Segment Decoder is a circuit that converts from one digital format to another. It will be converting a BCD to Binary in so that the segments will be selected and display a digit in decimal accordingly. If the BCD code is 0010, a,b,g,e,d will illuminate and the display shows 2. If the BCD code is 0011 a,b,g,c,d will illuminate and the display shows 3. The Decoder used is the DEC_BCD_7. The data input are 4 BCD values (A, B, C, D) which come from the switches on the board and they are 16. There will be four multiplexers and each multiplexer has 4 inputs. One digit will be driven at a time as the multiplexer chooses 1 output from 4 multiplexer inputs. There are four multiplexers therefore four outputs which are the inputs (A, B, C, D) to the decoder. These inputs feed the seven outputs which pass current to the segments and light up the correct segments. A counter produces a BCD output code from 0000 to 1001 and rests itself back to 0000. The counter repeats two bits to give four combinations.(8) (Laboratory Stage 1.pdf, 2020)

The seven LEDs all have its particular position as shown in Figure 10 above. Each of the seven LEDs are outputs from the DEC_BCD_7.
There are two types of LED Segment Display

1. The Common Cathode
2. The Common Anode

The Common Cathode is when the cathode connections are connected to digital low. The individual segments will light up if the signal is a 1.
The Common Anode is the opposite of the Common Cathode as the anode connections are connected to digital high. The individual segments will light up if the signal is a 0.

For this signal to be a 1,current has to be forward biased and the light emitted will be proportional to that current. Light intensity is directly proportional to the current, therefore, if there is too much current flowing there could be damage to the segments. A solution to this will be to add a resistor for controlling purposes. (3) (7-segment Display and Driving a 7-segment Display, n.d.)

To show how Values 0-9 will be outputted from the Seven Segment Driver, Figures 12a and 12b are attached below
Figure(12a) (7) (How Seven Segment Display Works & Interface it with Arduino, n.d.)

Figure(12b) (7) (How Seven Segment Display Works & Interface it with Arduino, n.d.)

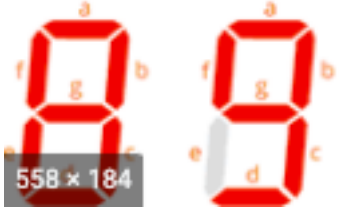


Figure 13

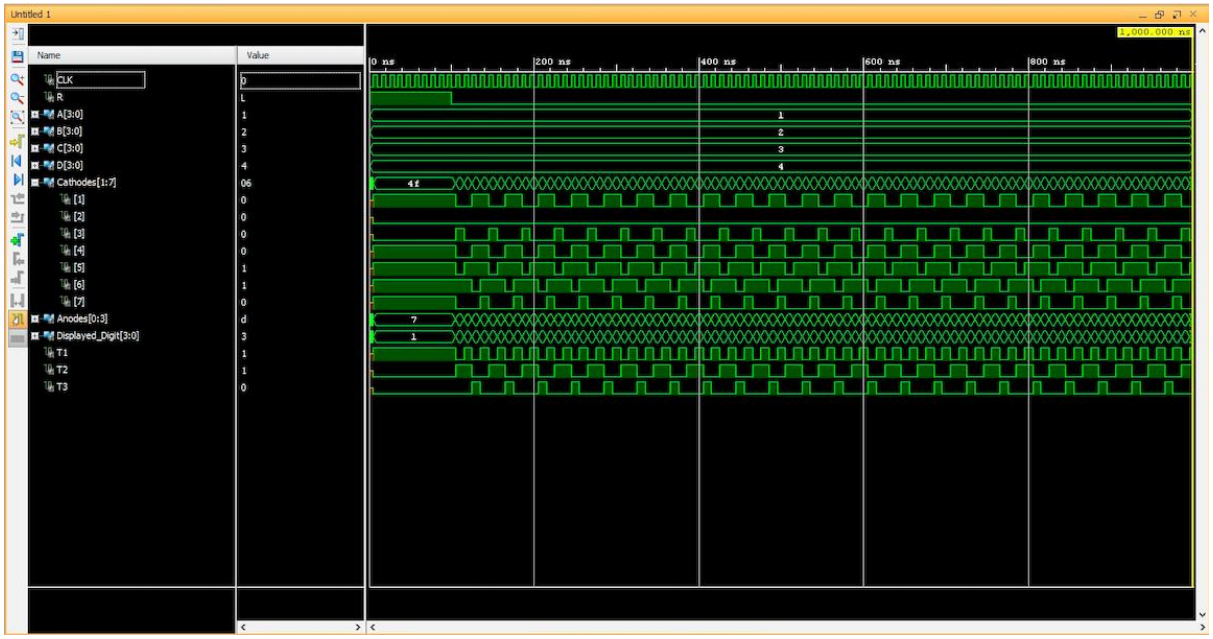


Figure 14 (5) (74HC4511; 74HCT4511 BCD to 7-segment latch/decoder/driver, 2016)

Inputs							Outputs							Display
$\overline{LE}$	$\overline{BI}$	$\overline{LT}$	D	C	B	A	a	b	c	d	e	f	g	
X	X	L	X	X	X	X	H	H	H	H	H	H	H	8
X	L	H	X	X	X	X	L	L	L	L	L	L	L	blank
L	H	H	L	L	L	L	H	H	H	H	H	H	L	0
L	H	H	L	L	L	H	L	H	H	L	L	L	L	1
L	H	H	L	L	H	L	H	H	L	H	H	L	H	2
L	H	H	L	L	H	H	H	H	H	H	L	L	H	3
L	H	H	L	H	L	L	L	H	H	L	L	H	H	4
L	H	H	L	H	L	H	H	L	H	H	L	H	H	5
L	H	H	L	H	H	L	L	L	H	H	H	H	H	6
L	H	H	L	H	H	H	H	H	H	L	L	L	L	7
L	H	H	H	L	L	L	H	H	H	H	H	H	H	8
L	H	H	H	L	L	H	H	H	H	L	L	H	H	9
L	H	H	H	L	H	L	L	L	L	L	L	L	L	blank
L	H	H	H	L	H	H	L	L	L	L	L	L	L	blank
L	H	H	H	H	L	L	L	L	L	L	L	L	L	blank
L	H	H	H	H	L	H	L	L	L	L	L	L	L	blank
L	H	H	H	H	H	L	L	L	L	L	L	L	L	blank
L	H	H	H	H	H	H	L	L	L	L	L	L	L	blank
H	H	H	X	X	X	X	[2]							[2]

Looking at Figure 13 the cathode value is 6 (0000110) and the displayed digit is 3. Using Figure 14, in the display column, the value 3 is H, H, H, H, L, L, H which corresponds to (0000110). Therefore, the Seven Segment Driver works.

4.6 Clock Divider

A clock divider takes input signals of a frequency and generates an output signal of a frequency. In the schematic, the digital low is a component so that there is no additional value for the circuit. 100MHz is the frequency already flowing through. This clock input is too fast and will not be compatible with the whole circuit. 500Hz is the proposed value for this clock input so the clock divider is used to reduce the frequency to this. Each counter used divides the previous frequency by 10 and a flip-flop is needed to divide the previous frequency by 2.

100MHz/10 = 10MHz

10MHz/10 = 1MHz

1MHz/10 = 100KHz

100KHz/10 =10KHz

10KHz/10 = 1KHz

Hence 5 Counters are needed and a flip flop to divide 1KHz to 500Hz.

The Vivado Output shows the distance from one peak to another peak on the A output is 10micro seconds. To find the frequency 1/10*10^-6 is done which gives 100KHz meaning it was divided by three counters.

Figure 15

5 Why BCD Multipliers are used

The interest in the hardware design of decimal arithmetic processes has increased exponentially over the years as execution time is short compared to that of software. FPGA is a hardware circuit for the implementation of logical operations by using parallel processing techniques. Multiplication is a complicated arithmetic operation with high propagation delay and high resource usage. BCD multipliers were introduced to dismiss this problem. The 1x1 digits multipliers is the foundation for all multipliers that incorporates the binary-BCD conversion system, however, owing to the fact that the results displayed were in binary they had to be converted to BCD by employing the shift add 3 approach. When decimal numbers are converted to binary numbers, there is a consistently minor conversion error. This is due to the fact that the base for decimal is base 10 and the base for binary numbers is base 2, and as a consequence of this, the change won't be 100% accurate. This leads to another reason why BCD has been introduced. BCD does a 4-bit binary code representation for each base 10 digits in a number, i.e 0001-1000- 0010 in BCD as 182 in decimal.

6 My Result

Figure X

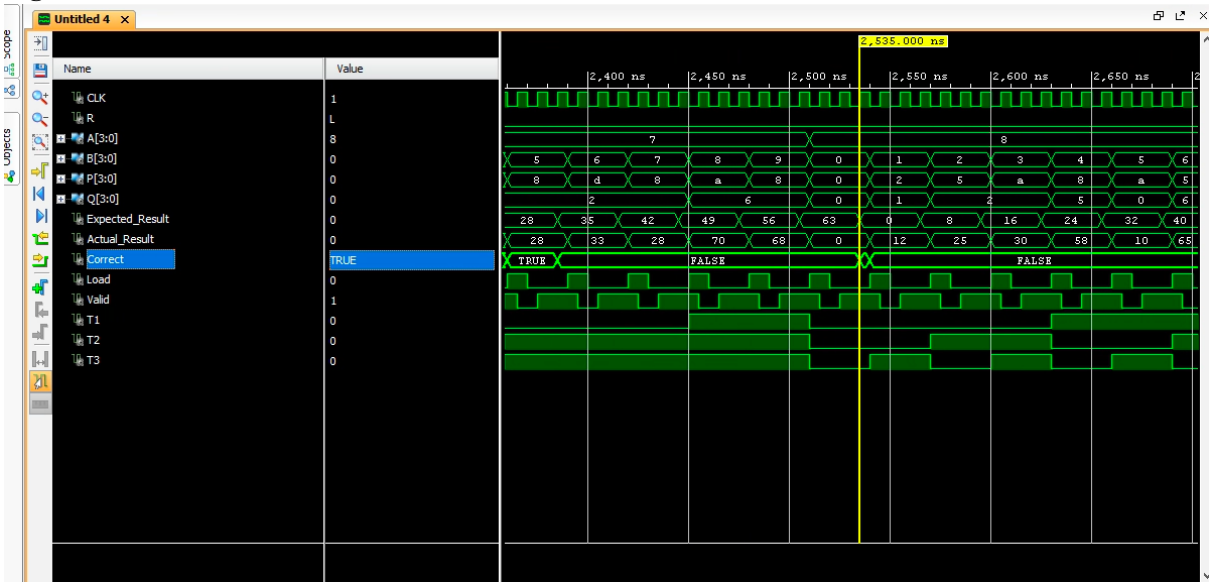


Figure X shows a part of this Vivado output where the "correct" is true. The inputs are 8 and 0 therefore 8 multiplied by 0 gives an Expected and Actual Result as 0.

Figure Y

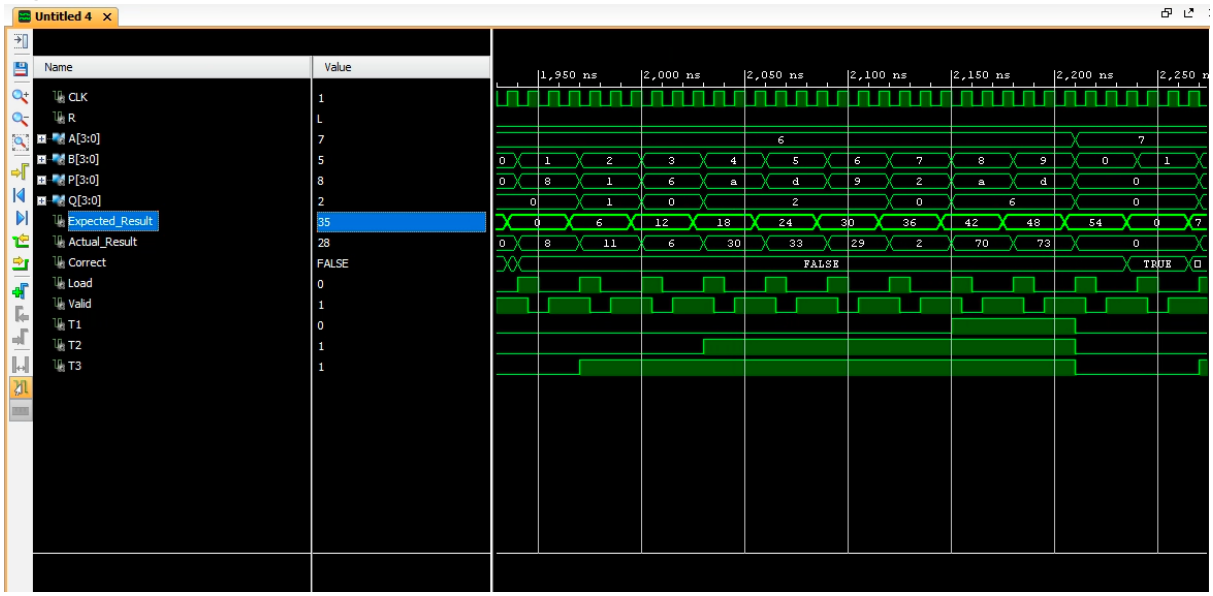


Figure Y shows a part of this Vivado output where the “correct” is false. The inputs are 7 and 5 therefore 7 multiplied by 5 gives an Expected Result as 35 and the Actual Result gotten is 28. This shows that the Binary to BCD converter doesn’t work correctly and there is an error somewhere. The previous circuits were tested step by step to figure out the error but they were all gave the expected simulation and results.

7 Expected Result

The expected result is get a true all through the “correct” column. The expected result is to get the same actual result and expected result when pointer is on a random number. The Binary to BCD Converter should have the correct values corresponding to each other.

8 Reflective Statement

A combinational multiplier using a sequential test bench was a brave move on my part. It could’ve worked with a combinational multiplier but it did not work in this case. The combinational multiplier was tested and it worked but I don’t think it is compatible for the whole BCD Multiplier when everything else is included. I made different structures of combinational multipliers and by the time I realised I thought of a sequential multiplier possibility it was too late. The Clock Divider should’ve had more counters and a working flip-flop. I tried adding two more counter and a flip-flop to the circuit but I got undefined values so I decided to stick with using three cause it was at least divided. I had a mindset of determination doing this as I wanted everything to work and put in as many hours possible. It was an interesting project and doing this I had to show a lot of patience.

9 Conclusion

In Conclusion the total circuit did not work. The Combinational Multiplier worked, the Flip-Flop worked, the adder worked, the Seven Segment Driver worked. The Binary to BCD Converter however did not work so as the clock divider.

10 Bibliography

(1) electronicshub.org. 2015. Binary Multiplication Methods. [online] Available at: <https://www.electronicshub.org/binary-multiplication/>

(2) zone.ni.com. 2017. ADD_FULL4_FCR. [online] Available at: <https://zone.ni.com/reference/en-XX/help/375482B-01/multisim/addfull4fcr/>

(3) Electronics Tutorial. n.d. 7-segment Display and Driving a 7-segment Display. [online] Available at: <https://www.electronics-tutorials.ws/blog/7-segment-display-tutorial.html>

(4) electricaltechnology.org. n.d. Binary Multiplier - Types & Binary Multiplication Calculator. [online] Available at: <https://www.electricaltechnology.org/2018/05/binary-multiplier-types-binary-multiplication-calculator.html>

(5) https://assets.nexperia.com. 2016. 74HC4511; 74HCT4511 BCD to 7-segment latch/decoder/driver. [online] Available at: <https://assets.nexperia.com/documents/data-sheet/74HC_HCT4511.pdf>

(6) circuitstoday.com. n.d. Arduino and Seven Segment Display-Interfacing. [online] Available at: <https://www.circuitstoday.com/arduino-and-7-segment-display> [Accessed 5 February 2021].

(7) lastminuteengineers.com. n.d. How Seven Segment Display Works & Interface it with Arduino. [online] Available at: <https://lastminuteengineers.com/seven-segment-arduino-tutorial/>

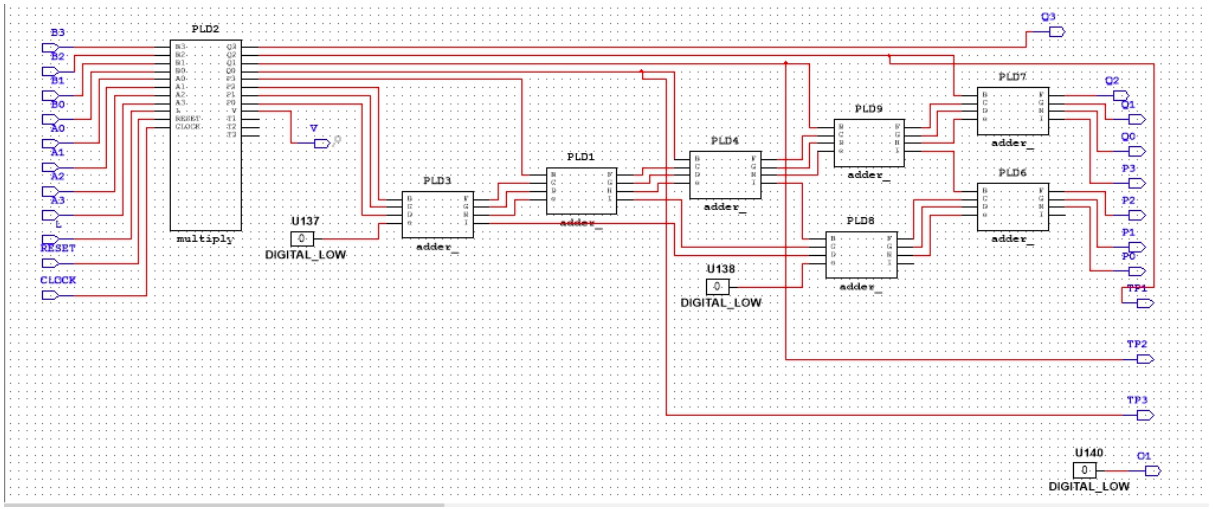
(8) moodle.essex.ac.uk. 2020. Laboratory Stage 1.pdf. [online] Available at: <https://moodle.essex.ac.uk/pluginfile.php/471733/mod_resource/content/12/Laboratory%20stage%201.pdf>

(9) circuitbasics.com. n.d. WHAT ARE FLIP FLOPS?. [online] Available at: <https://www.circuitbasics.com/what-are-flip-flops/>

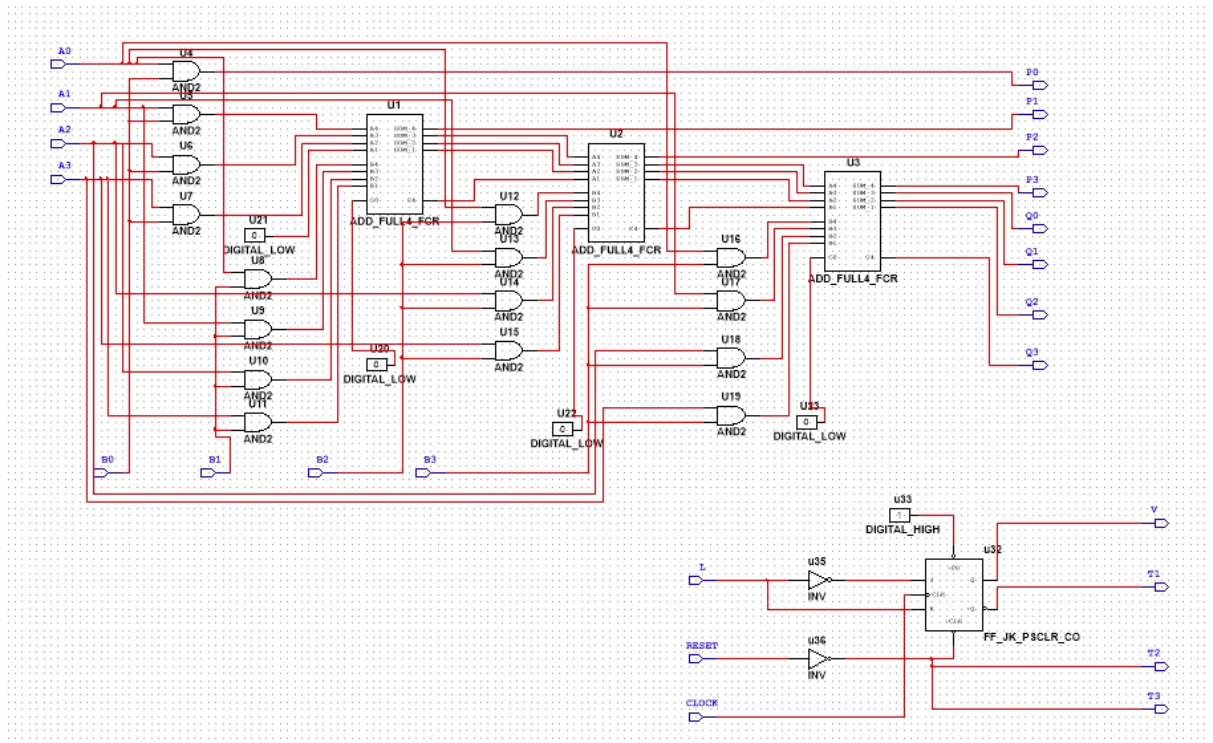
(10) seas.upenn.edu. 2007. *University of Pennsylvania*. [online] Available at: <https://www.seas.upenn.edu/~ese171/labise/Lab03_CombMultiplier.pdf>

11 Appendix

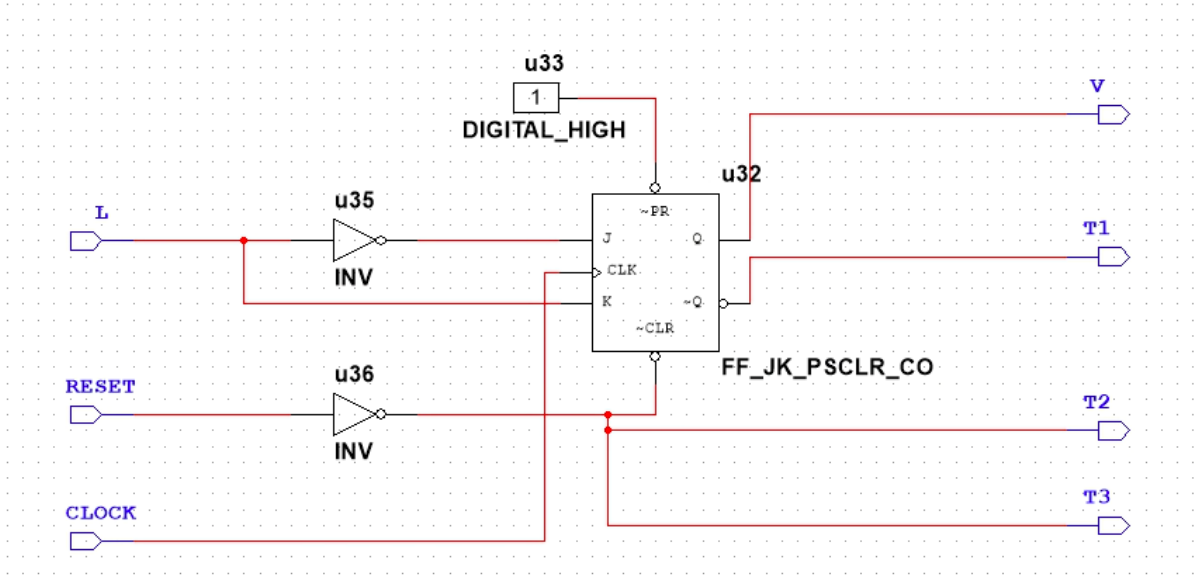
BCD MULTIPLIER (BINARY MULTIPLIER TO BCD CONVERTER)



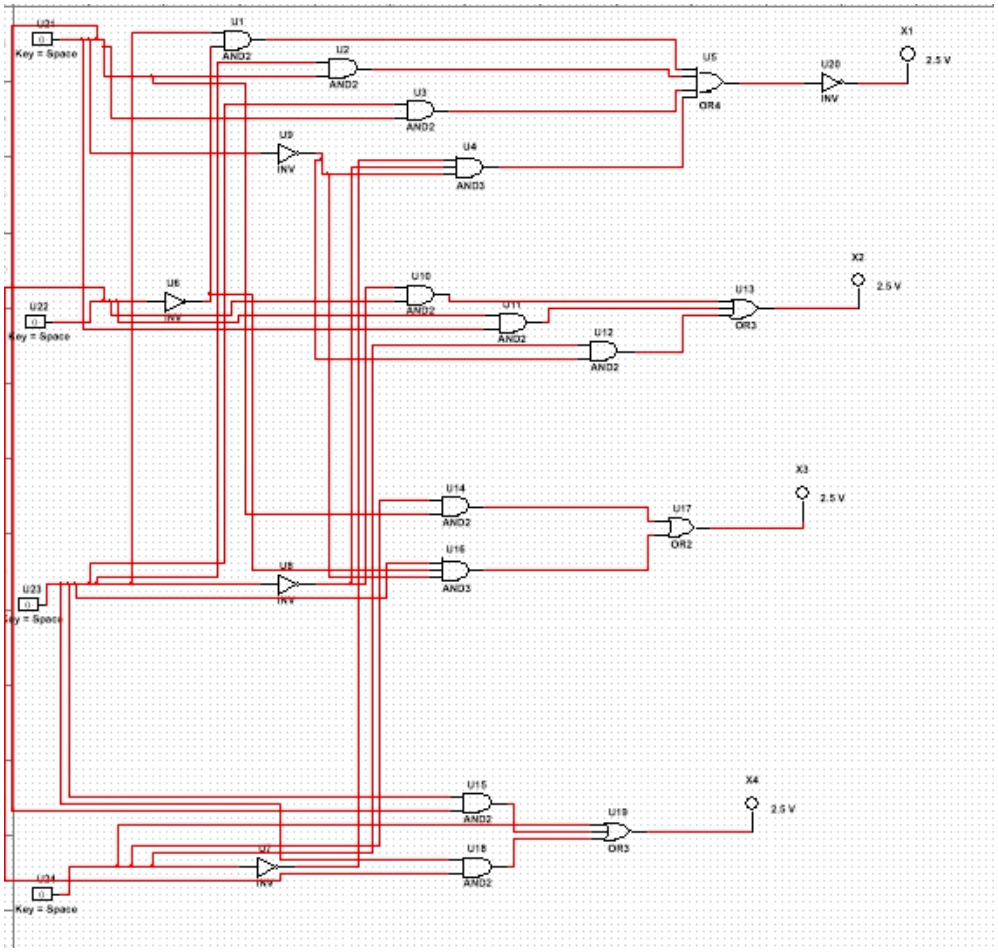
COMBINATIONAL MULTIPLIER



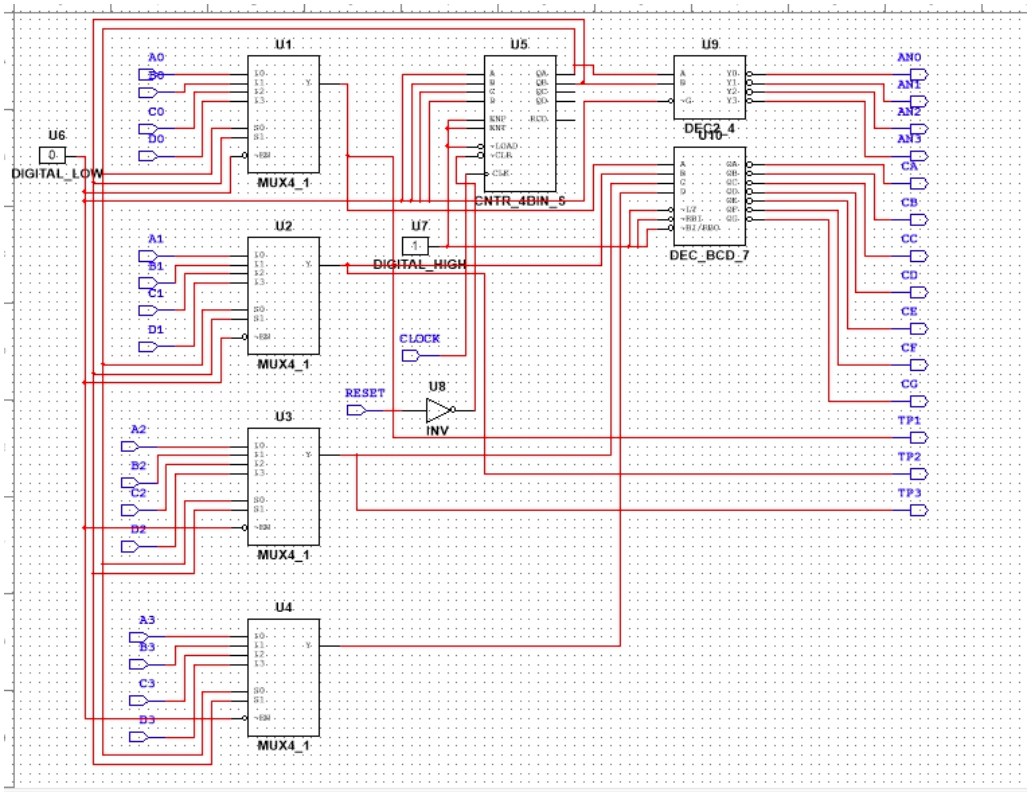
FLIP-FLOP



PLUS 3 ADDER



SEVEN SEGMENT DRIVER



CLOCK DIVIDER