EC-204

<LAB - 6>

NITK SURATHKAL

INBASEKARAN.P

201EC226

Prof: Sumam S

1) Design a synchronous modulo-5 counter using D FF

Solution:

Design and explanation

The counter design table for such counter shows the three flip-flop and their states also (0 to 5 states), as in table (a), the 6 inputs needed for the three flip-flops. The flip-flop inputs needed to step up the counter from the current to the next state have been worked out along with the assist of the excitation table illustrated in the table.

NOTE: A is the MSB and C is the LSB.

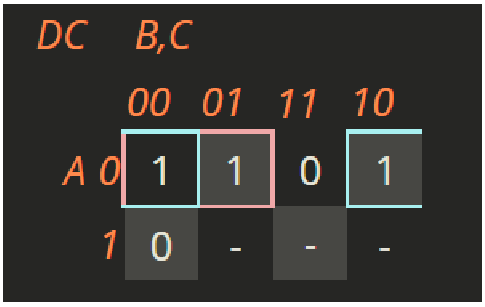
A	B	C	A+	B+	C+	DA	DB	DC
0	0	0	1	0	0	1	0	0
0	0	1	0	0	0	0	0	0
0	1	1	0	0	1	0	0	1
1	0	0	0	1	1	0	1	1

DA	B,C			
	00	01	11	10
A 0	0	0	1	0
1	0	-	-	-

DA (A, B, C) = BC

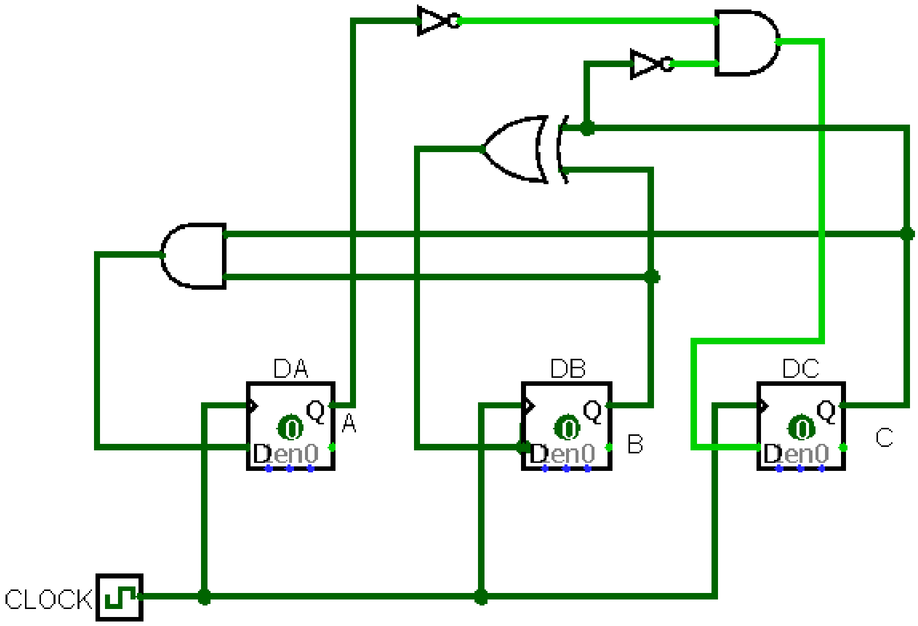
DB	B,C			
	00	01	11	10
A 0	0	1	0	1
1	0	-	-	-

DB(A, B, C) = B'C + BC'



DC(A, B, C) = A'B' + A'C'

CIRCUIT



RESULT OBTAINED

VIDEO LINKED

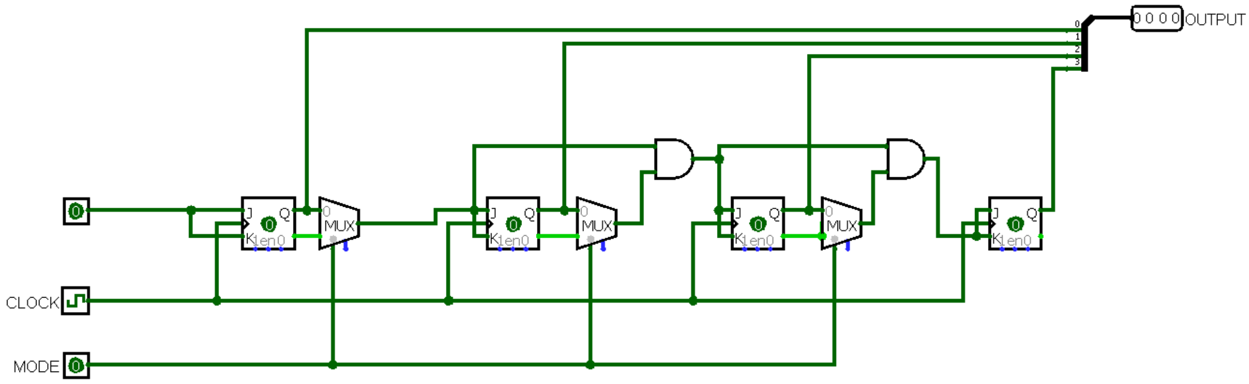
2) Design a synchronous 4 bit up-down counter using JK FF

Solution:

Design and explanation

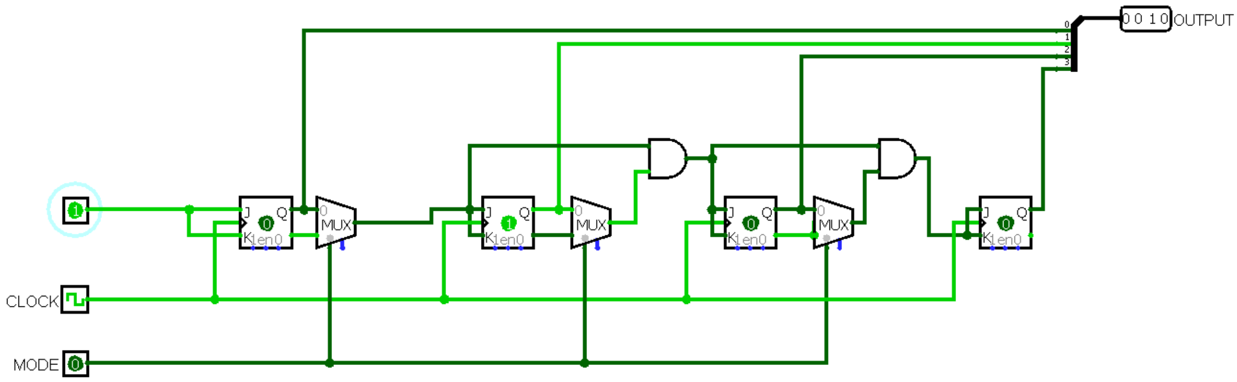
To design an up down counter we can use a Mode variable which when 0 does up counting and 1 does down counting. To perform down counting we just need to give the compliment of the clock to the next block, now the counter counts from backwards when Mode is equal to 1.

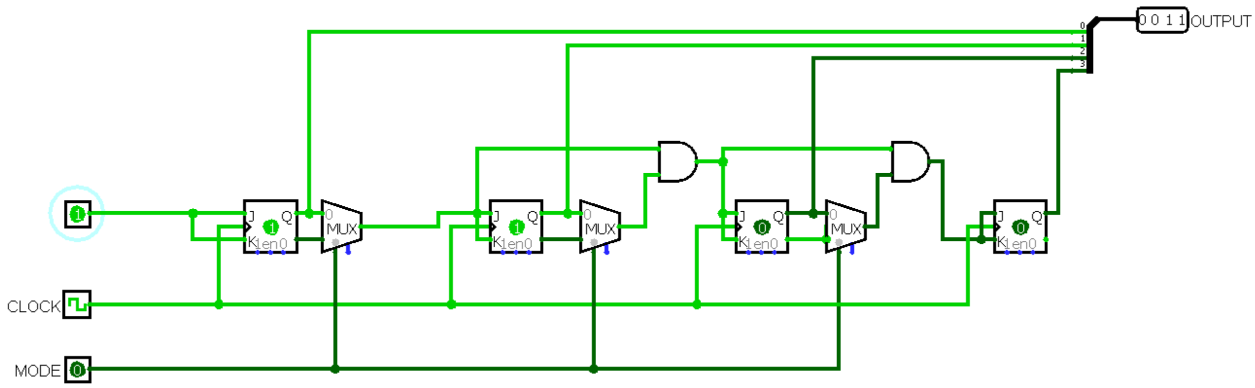
CIRCUIT



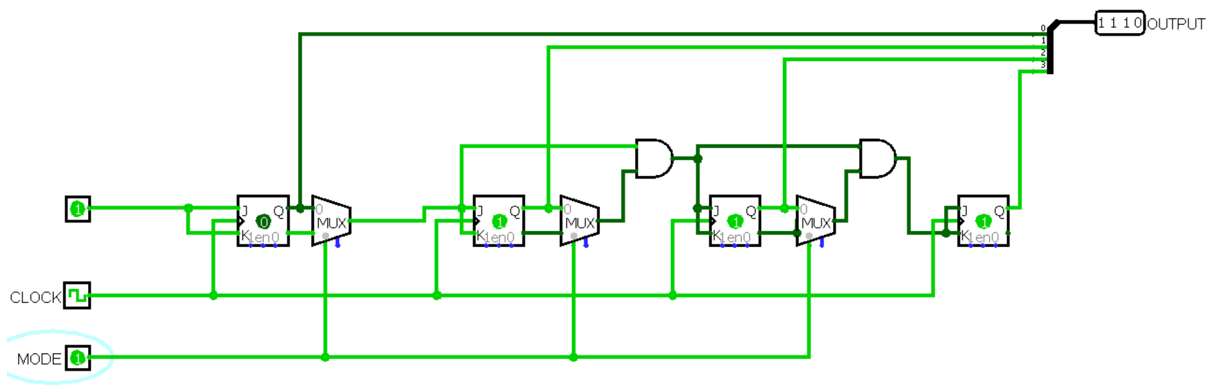
RESULT OBTAINED

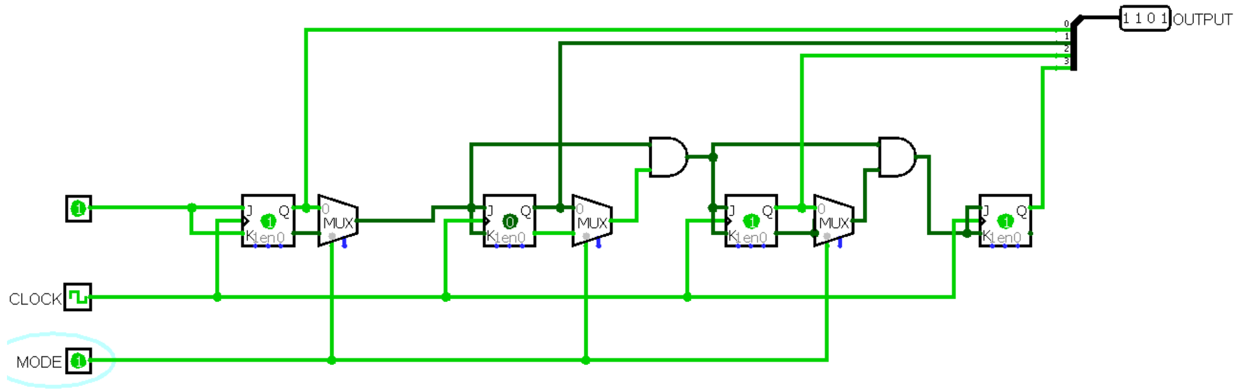
UP Count





Down Count





3) Design a circuit that generates the sequence 0-8-12-6-13-11-7-3-1-0 using DFF.

Solution:

Design and explanation

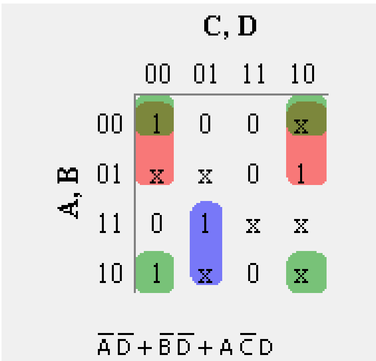
The sequence generator block diagram using a counter is illustrated below. Here, the combinational circuit is the next state decoder. The input of this state decoder can be obtained from the outputs of the FFs. Similarly, the outputs of this state decoder are given as inputs to the flip-flops. Based on the number of FFs, the required sequence like 0's or 1's can be generated.

Characteristics table

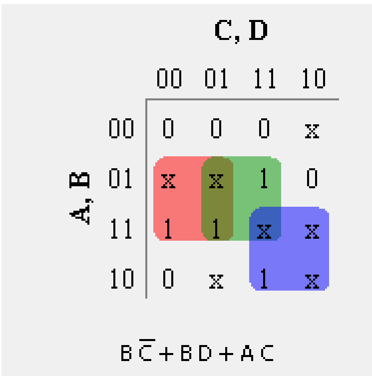
A	B	C	D	DA	DB	DC	DD
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	x	x	x	x
0	0	1	1	0	0	0	1
0	1	0	0	x	x	x	x
0	1	0	1	x	x	x	x
0	1	1	0	1	1	0	1
0	1	1	1	0	0	1	1
1	0	0	0	1	1	0	0
1	0	0	1	x	x	x	x
1	0	1	0	x	x	x	x
1	0	1	1	0	1	1	1
1	1	0	0	0	1	1	0
1	1	0	1	1	0	1	1
1	1	1	0	x	x	x	x
1	1	1	1	x	x	x	x

Combinational logic expression

DA



DB



DC

A, B \ C, D	00	01	11	10
00	0	0	0	x
01	x	x	0	1
11	1	0	x	x
10	1	x	1	x

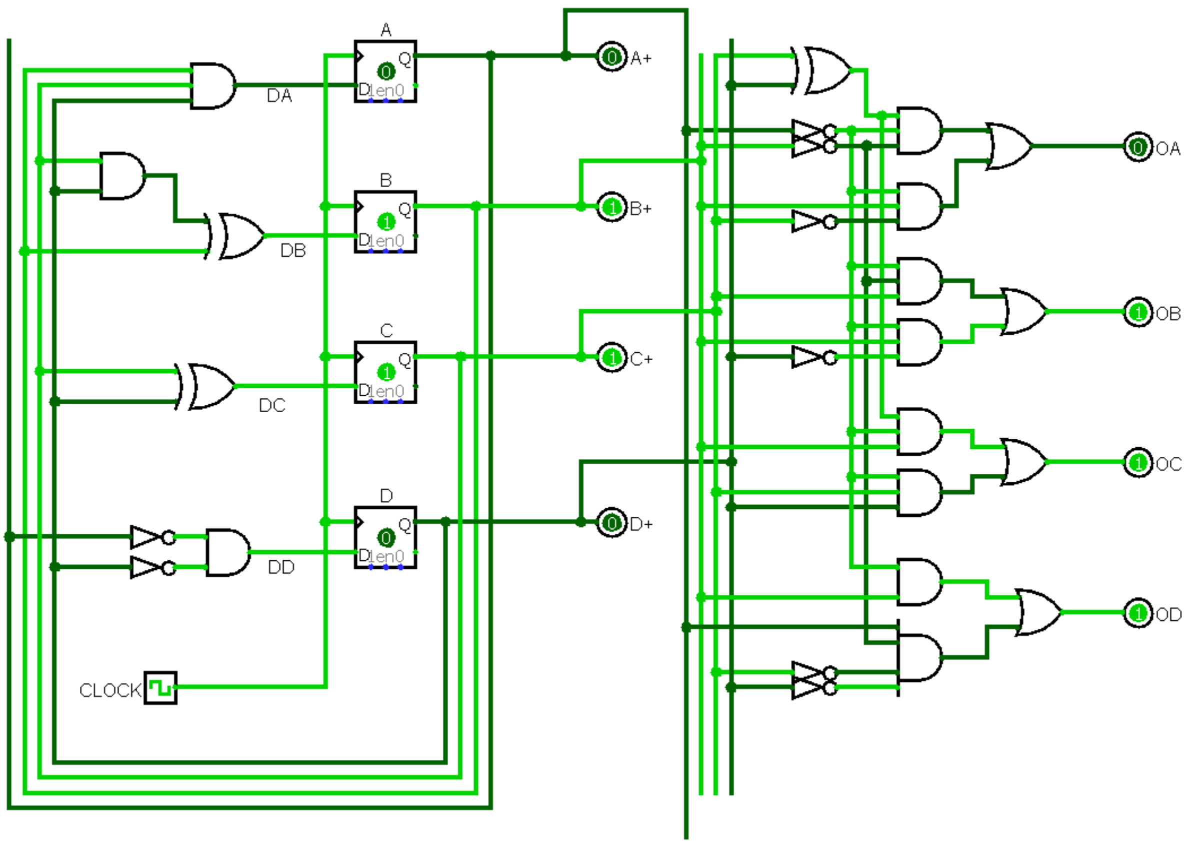
$B\overline{D} + A\overline{B}$

DD

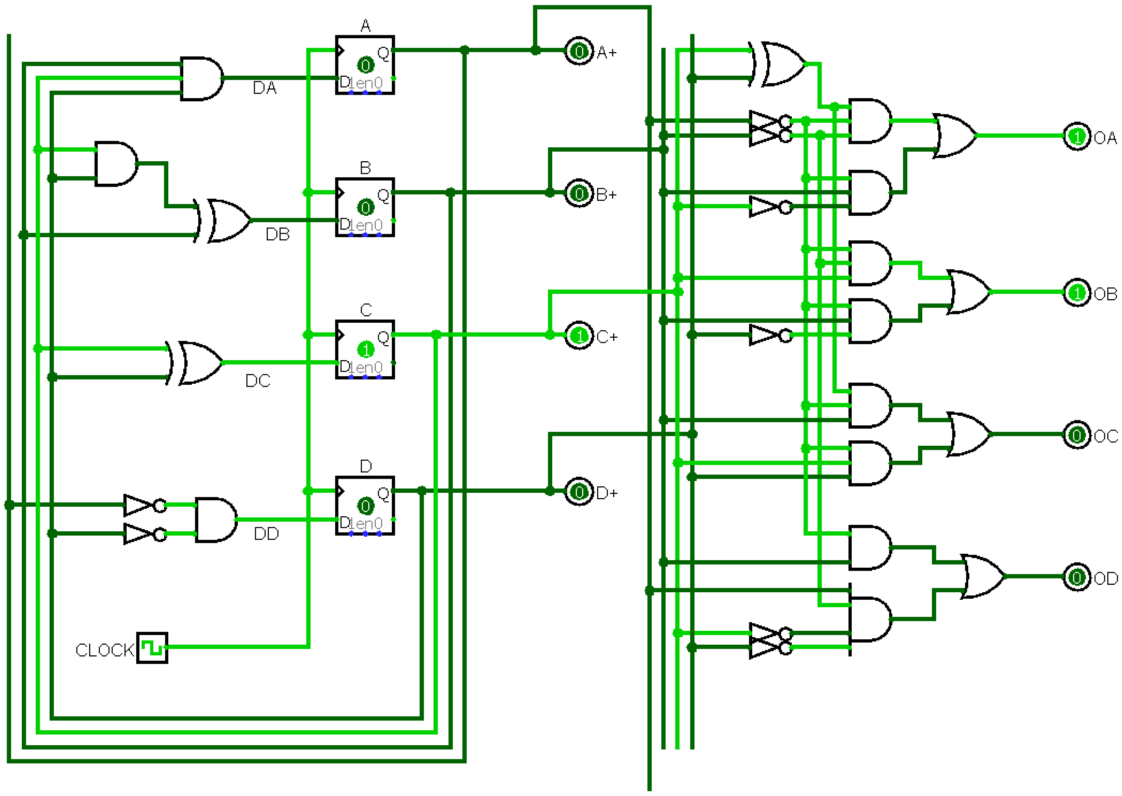
A, B \ C, D	00	01	11	10
00	0	0	1	x
01	x	x	1	1
11	0	1	x	x
10	0	x	1	x

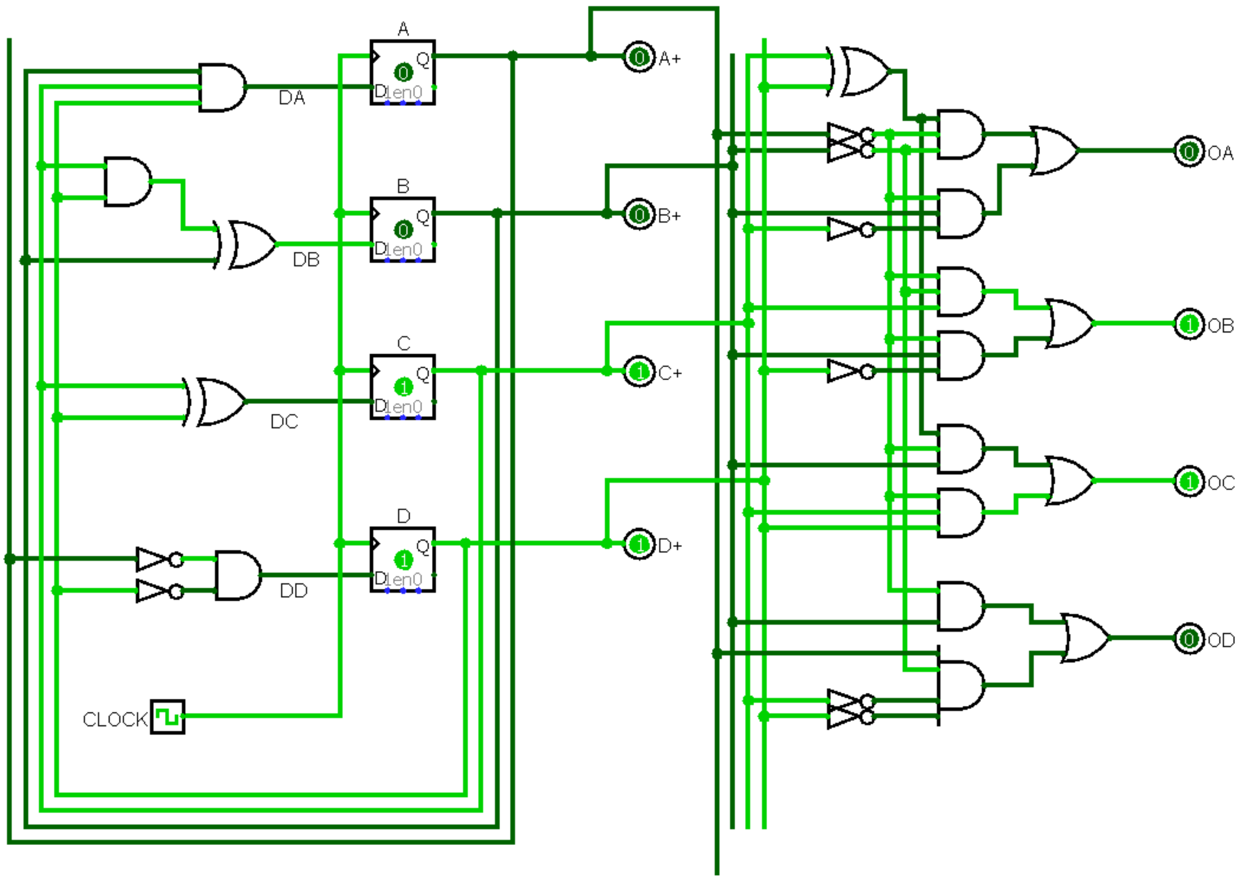
$C + AD$

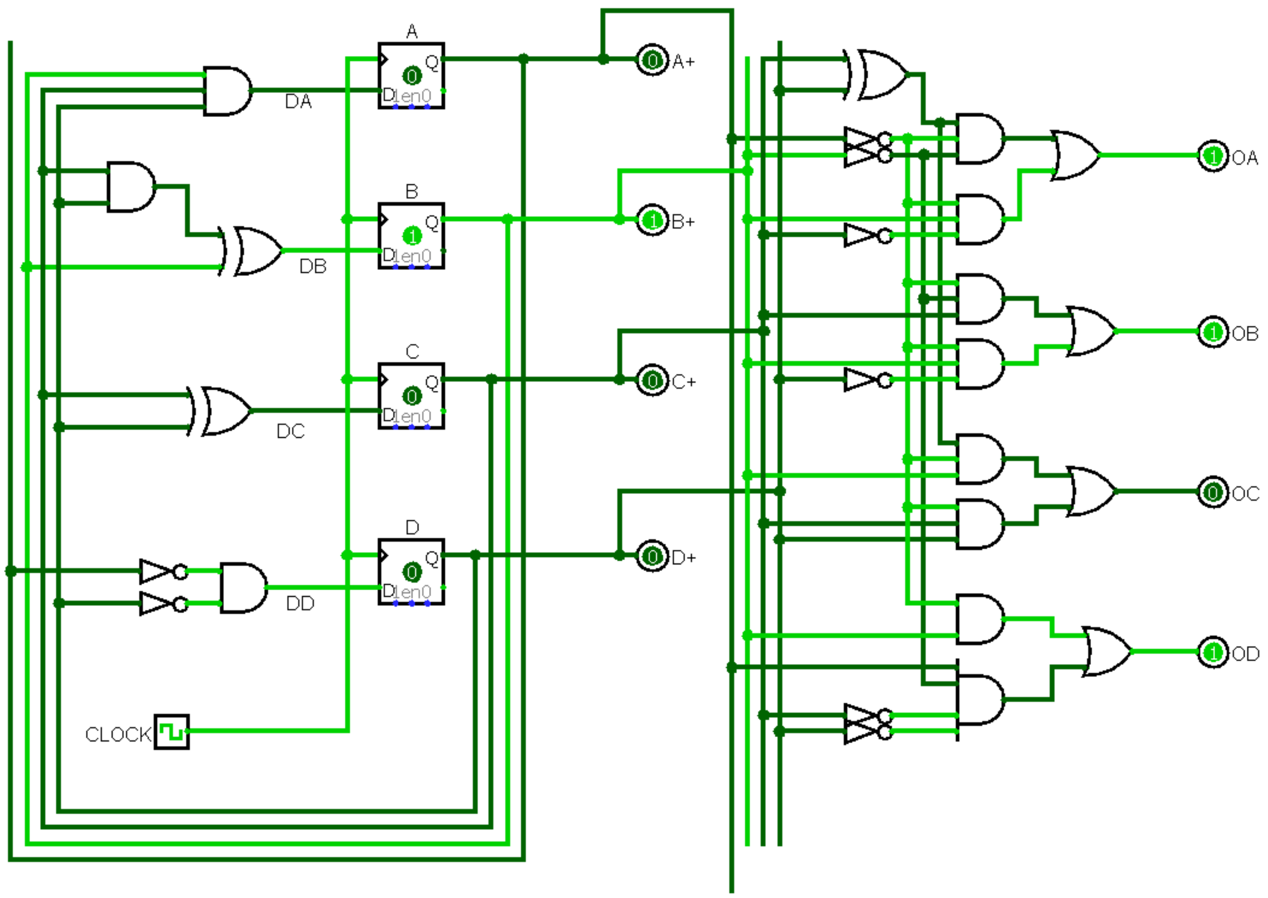
CIRCUIT

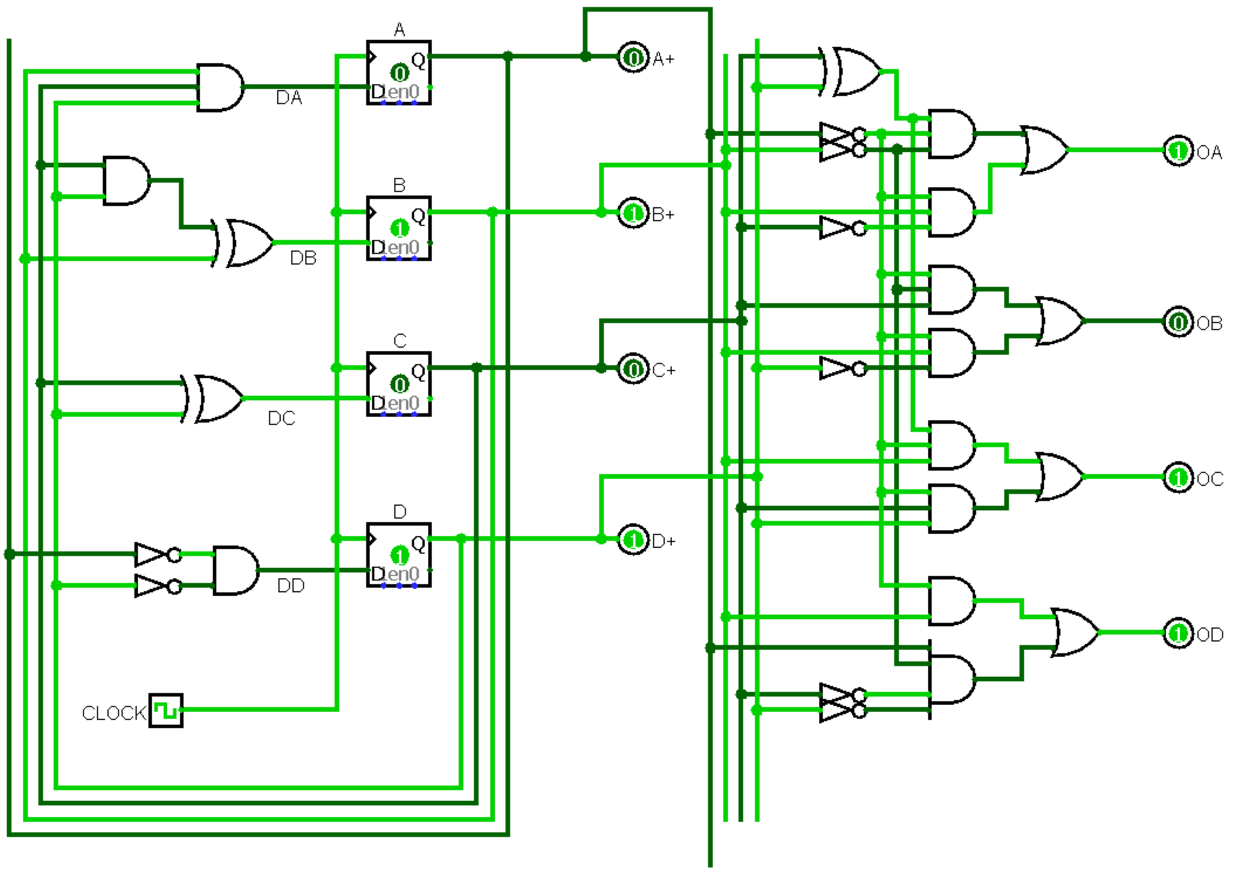


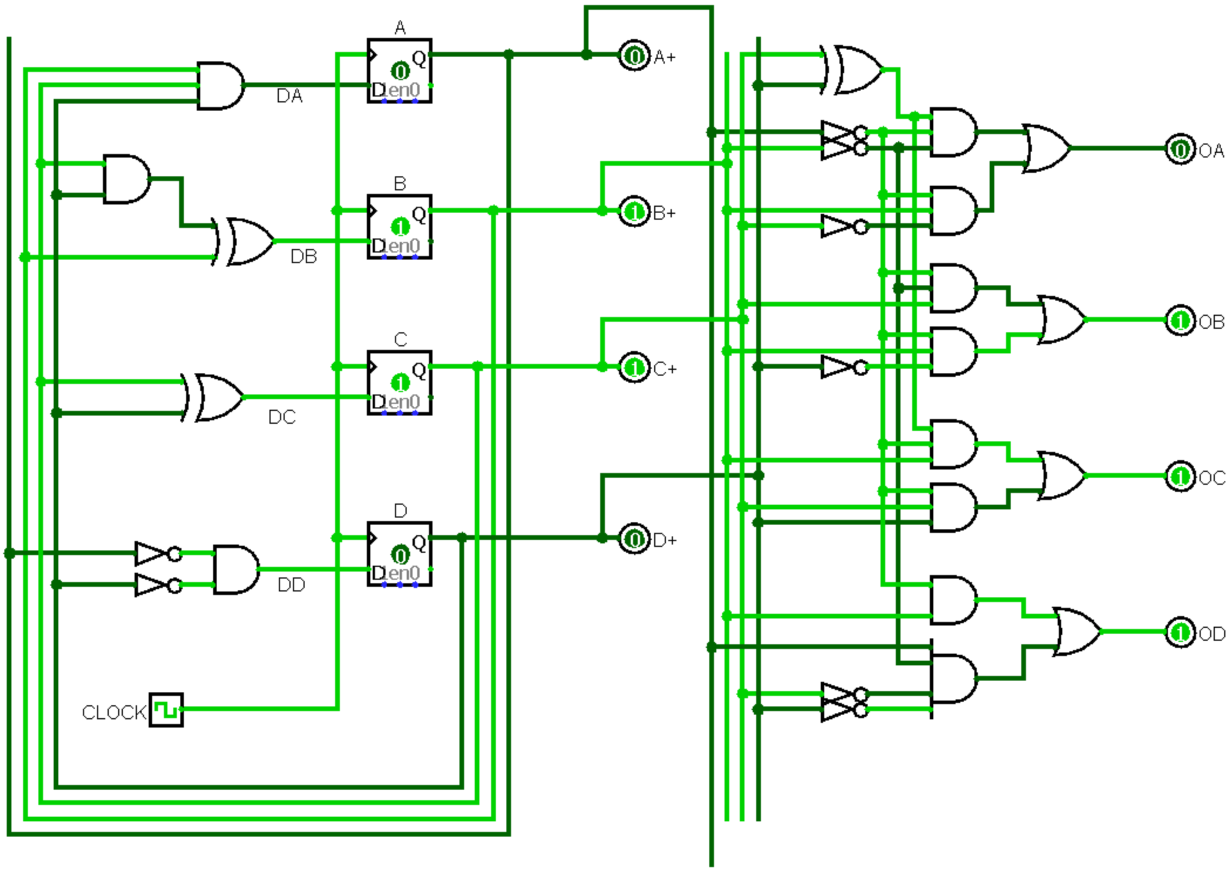
RESULT OBTAINED






A
B
C
D
DA
DB
DC
DD
CLOCK
A+
B+
C+
D+
OA
OB
OC
OD


A
B
C
D
DA
DB
DC
DD
CLOCK
A+
B+
C+
D+
OA
OB
OC
OD



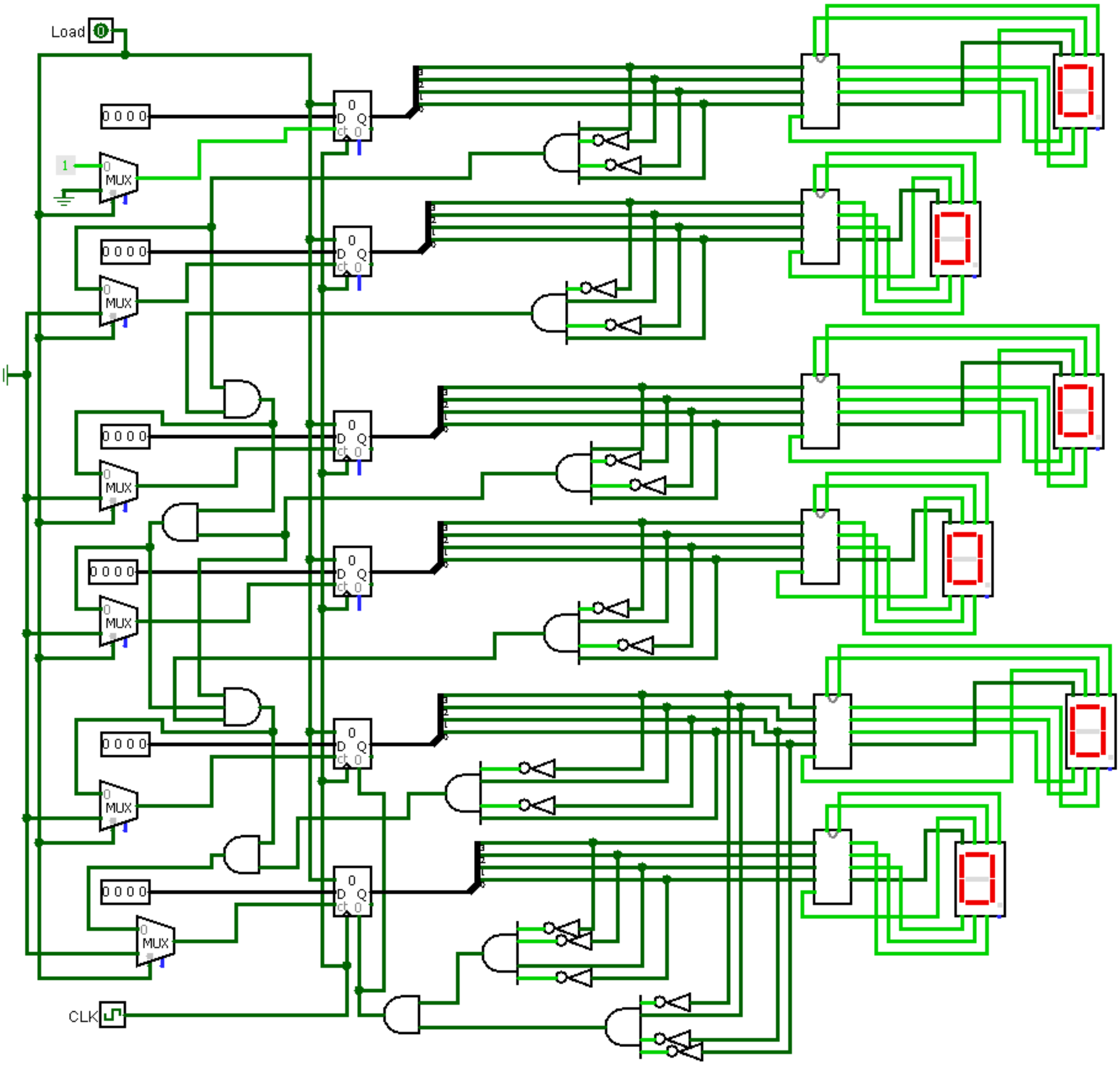
4) Design a Real time clock HRS: MIN:SEC using the counter available in the logisim library. Display the results using seven segment displays. There should be provision to set the time (using load). Optional – Add provision for setting an alarm and displaying it

Solution:

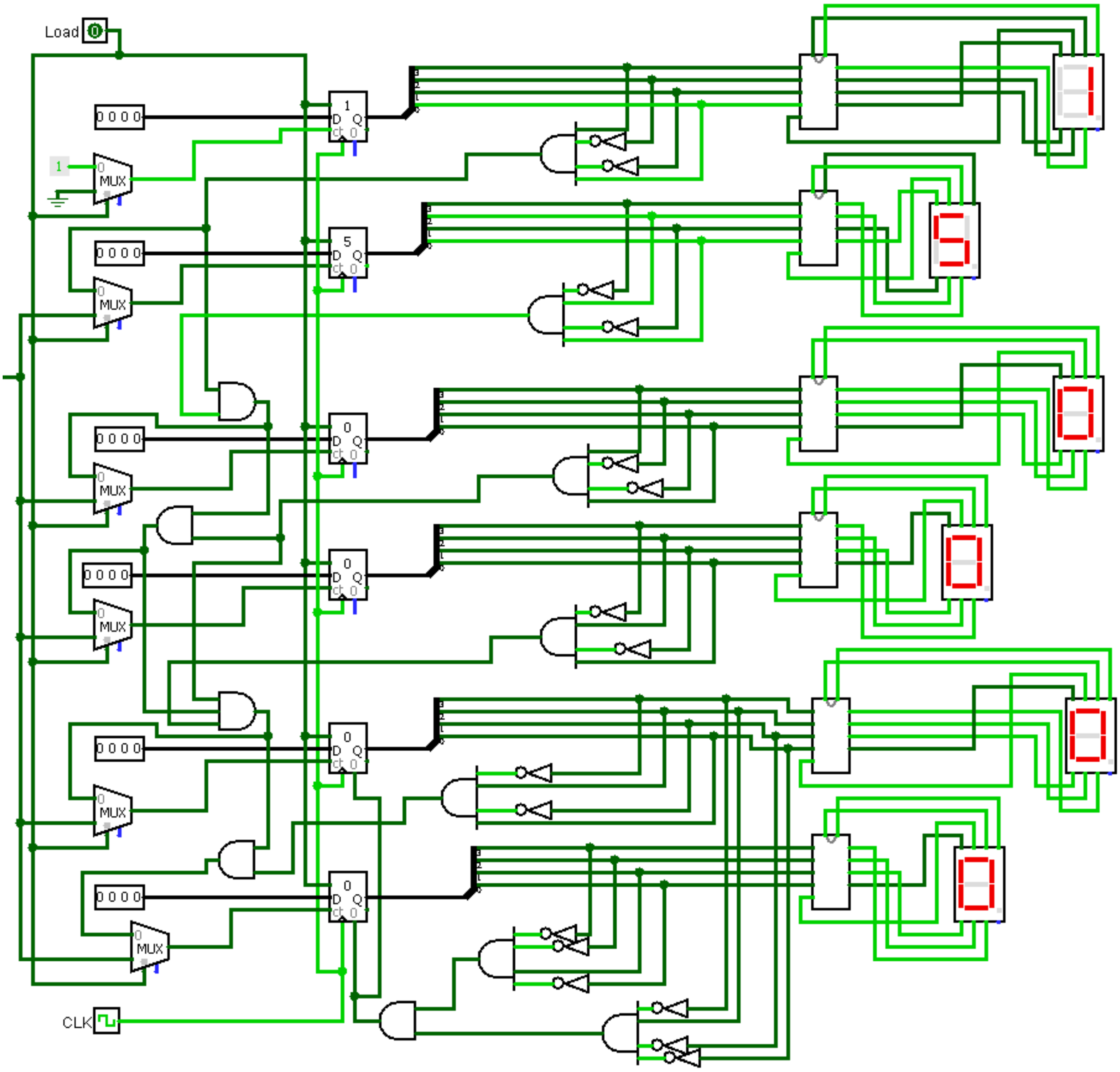
Design and explanation

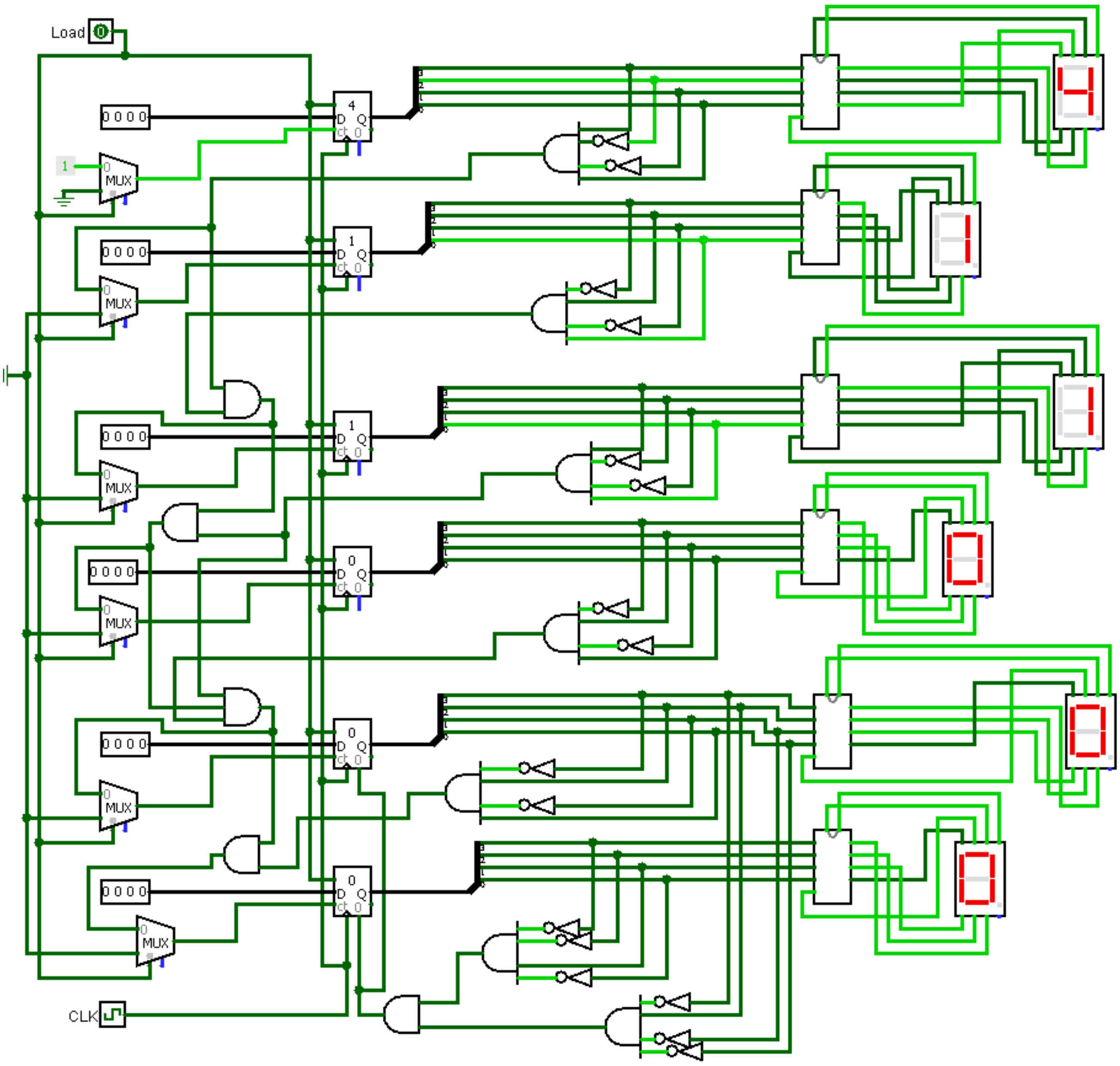
For the counters in the ones place of hours, minutes and seconds their maximum value before reset is 9, for the tens place in minutes and seconds maximum is 5 and for the hour's tens place it is 2, so the specifications are made accordingly. For the counter in one's place of seconds it should count for every clock pulse and for the one in the tens place it should count when the previous counter changes from 9 to 0, so using an AND gate it can be implemented. Similarly, the minutes can be designed but the minutes should count when the seconds changes from 59 to 00, it can also be made to do so using an AND gate. For the hours position it starts counting only when minutes and seconds change simultaneously from 59:59 to 00:00, such implementation can be done in a similar way as mentioned before. For the hours position it should reset when the count is 24 so a reset mechanism can be implemented to do so. As per inserting the data (setting the time) 2:1 MUXs can be used such that when the set input is 1 all the counters are disabled and load input will active and data is loaded for the next clock pulse.

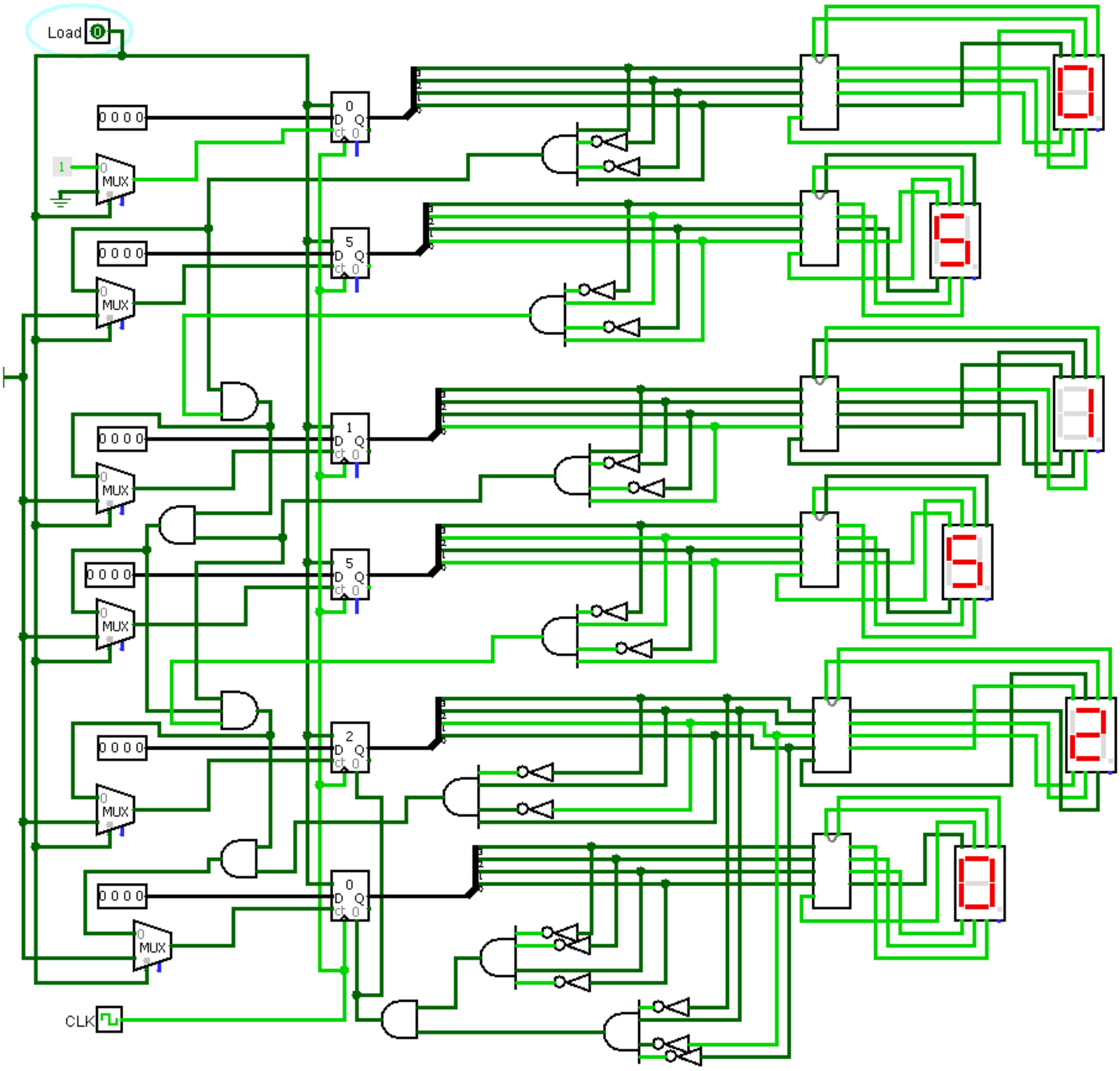
CIRCUIT



RESULT OBTAINED

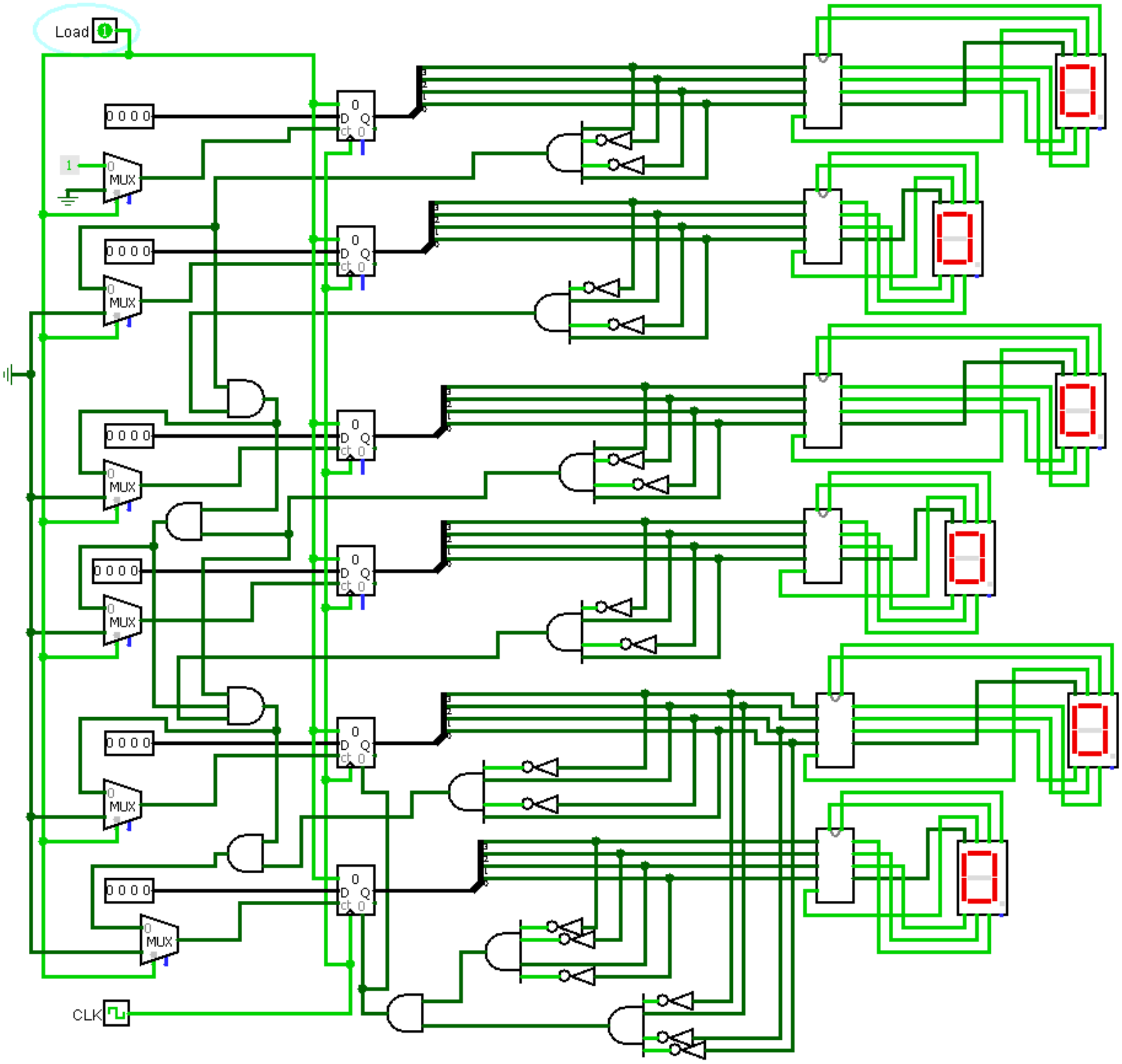



Load
0 0 0 0
MUX
CLK

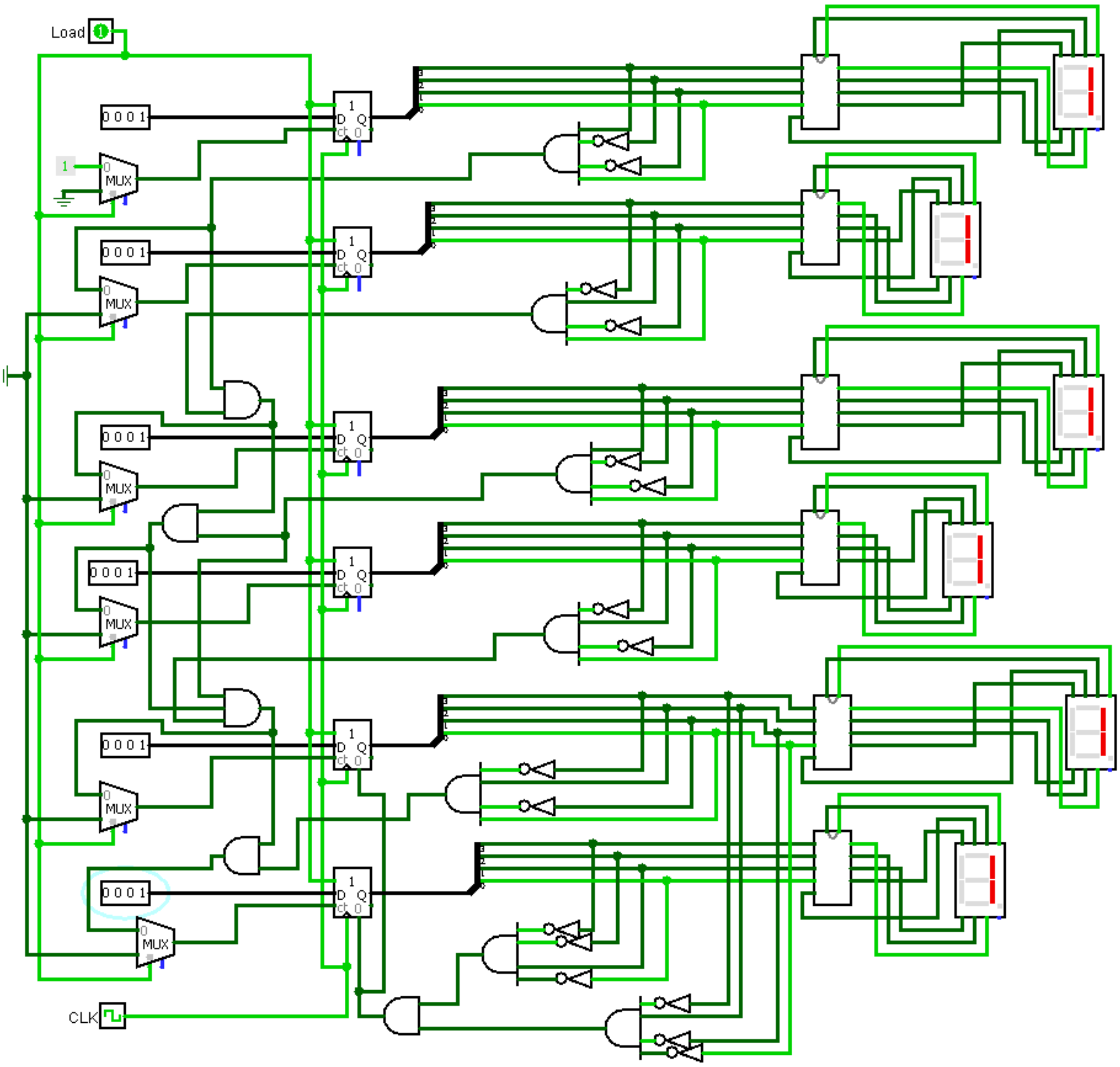


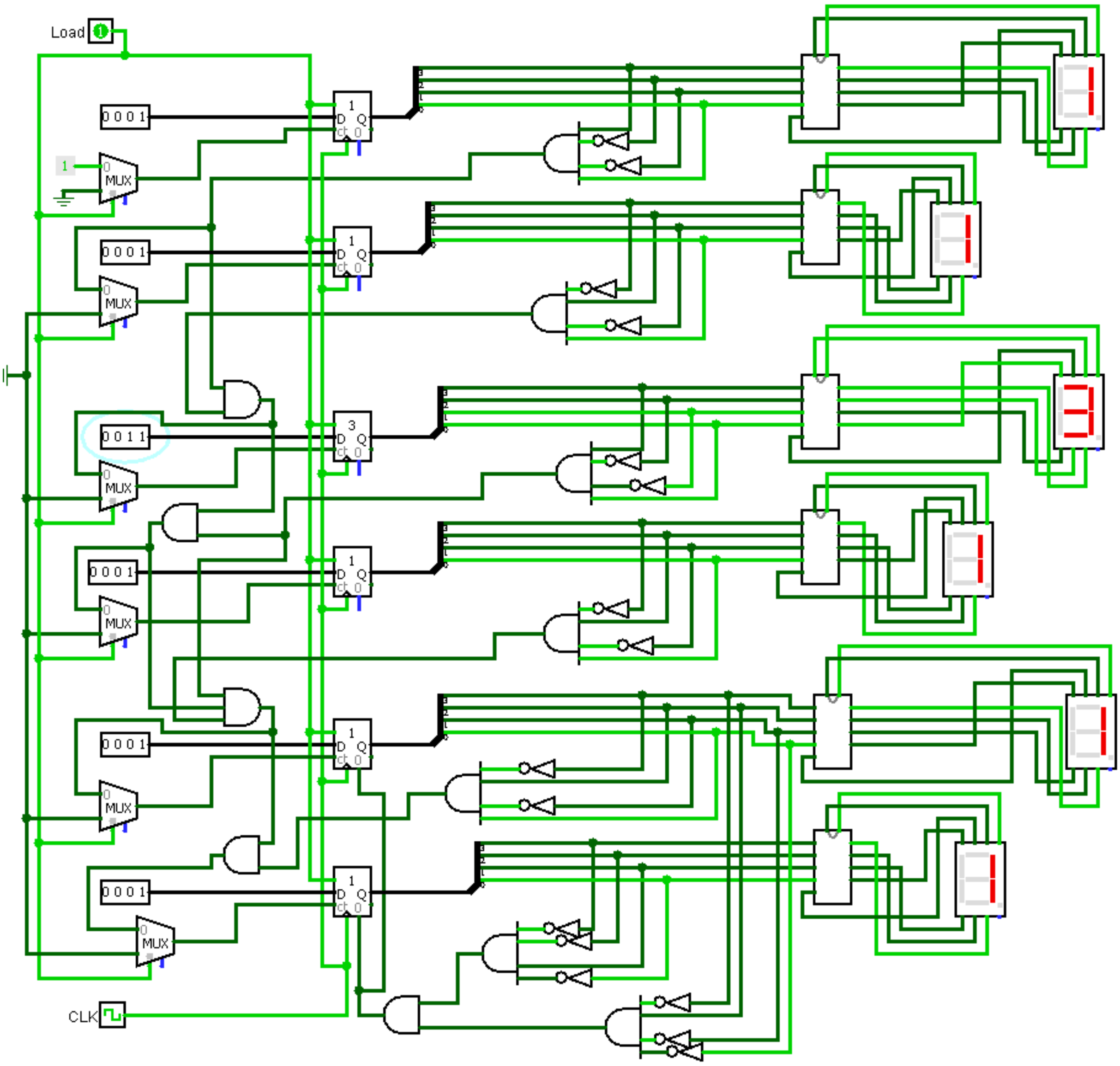
Setting time

In order to set the time, the set input is made 1 and the time is entered in binary equivalent in the inputs given, here time is entered as 23:59:59 in the inputs, so for the next pulse the date will be inserted into the counters and will be displayed as shown. After the time the Set input should be made 0 so that it starts work as a clock.



Load
CLK
MUX
0 0 0 0
D Q


Load
0 0 0 1
MUX
0 0 0 1
MUX
0 0 0 1
MUX
0 0 0 1
MUX
0 0 0 1
MUX
0 0 0 1
MUX
CLK


Load
0 0 0 1
1
MUX
0 0 0 1
MUX
0 0 1 1
MUX
0 0 0 1
MUX
0 0 0 1
MUX
0 0 0 1
MUX
CLK