Brunel University London

CHONGQING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

Assignment Report

TERM: Spring 2022

Module: EE1655 Digital Systems and Microprocessors

CLASS: 34092102

BRUNEL ID: 2161047

NAME: Xukang Liu

TUTOR: Zhengwen Huang

May 2022

Abstract

1 Aim

To review the digital system, we learned before. Further understand numerical representation of information, operation on binary data, combinational logic design and sequential logic fundamentals. In addition, use the knowledge learned in class to design some simple digital circuits.

2 Objectives

2.1 To review the contents of the first 5 chapters, which include converting between binary and other base, combinational logic system and sequential logic fundamentals.

2.2 To do some exercise and demonstrate what we have learned in this semester. It can help me further understand some knowledge points.

2.3 To design some simple digital circuits, which use some gates and triggers.

3 Theory

3.1 Numerical representation of information

- Binary coded decimal (BCD)
- Gray code
- Error-detecting codes (parity codes)
- Error-correcting codes

3.2 Combination logic design

- Boolean Algebra

De Morgan's Theorems:

$$\overline{A+B} = \bar{A}.\bar{B}$$
$$\overline{A.B} = \bar{A}+\bar{B}$$

- Logic Gates

AND, OR, NOT, NAND, NOR, EXCLUSIVE OR, EQUIVALENCE

- The 1st Canonical Form
- The 2nd Canonical Form
- The Karnaugh Map (minimal canonical form)
- Hazards in combinational logic

3.3 Sequential logic & Design

- Trigger

SRFF, JKFF, TFF (asynchronous, synchronous), DFF

- Design of sequential logic circuits

Asynchronous counter with output logic

Synchronous sequential logic

4 Methods and Solutions

4.1 Problem 1 (Number system)

My Brunel ID is 2161047. The last two digits is "47" and this number is greater than "05", so I do not need to make any changes.

4.1.1

a) The 4 least significant digits of my Brunel ID is "1047".

1st divide by 2: 1047/2=523 reminder 1 ∴b_0=1
2nd division: 523/2=261 reminder 1 ∴b_1=1
3rd division: 261/2=130 reminder 1 ∴b_2=1
4th division: 160/2=65 reminder 0 ∴b_3=0
5th division: 65/2=32 reminder 1 ∴b_4=1
6th division: 32/2=16 reminder 0 ∴b_5=0
7th division: 16/2=8 reminder 0 ∴b_6=0
8th division: 8/2=4 reminder 0 ∴b_7=0
9th division: 4/2=2 reminder 0 ∴b_8=0
10th division: 2/2=1 reminder 0 ∴b_9=0
11th division: 1/2=0 reminder 1 ∴b_{10}=1

So, the binary number is: 10000010111.

b) My binary number is 10000010111

i. Base 8

010 / 000 / 010 / 111
So, the number after conversion to octal is: 2027.

ii. Hexadecimal

0010 / 0001 / 0111
So, the number after conversion to hexadecimal is: 417.

iii. Base 12

Firstly, convert the binary number to decimal number 1047.

1st 1047/12=87 reminder 3 ∴b_0=3
2nd 87/12=7 reminder 3 ∴b_1=3
3rd 7/12=0 reminder 7 ∴b_2=7

So, the number after conversion to hexadecimal is: 733.

4.1.2

After converting my Brunel ID, it's 10.47.

1) For integer parts:
   10/5=5 reminder 0 $\therefore b_0=0$
   5/2=2 reminder 1 $\therefore b_1=1$
   2/2=1 reminder 0 $\therefore b_2=0$
   1/2=0 reminder 1 $\therefore b_3=1$

So, the binary number of the integer part is: 1010.

2) For fractional number parts:
   0.47*2=0.94 Fetch 0
   0.94*2=1.88 Fetch 1
   0.88*2=1.76 Fetch 1
   0.76*2=1.52 Fetch 1
   0.52*2=1.04 Fetch 1

So, the binary number of the fractional part is: 01111.
But the fractional part needs to accurate to 4 bits, so the fractional part is 1000.
So, the total number is 1010.1000.

4.1.3

The whole of my number is 2161047.

DECIMAL	8421	2316	74-2-1
0	0000	0000	0000
1	0001	0010	0111
2	0010	1000	0110
3	0011	0100	0101
4	0100	0110	0100
5	0101	1100	1010
6	0110	0001	1001
7	0111	0011	1000
8	1000	1001	1111
9	1001	0101	1110

Table1.1: Some weight of BCD codes

So, we can find that:

i. $0010\ 0001\ 0110\ 0001\ 0000\ 0100\ 0111_{8421}$
ii. $1000\ 0010\ 0001\ 0010\ 0000\ 0100\ 0011_{2316}$

iii. $0110\ 0111\ 1001\ 0111\ 0000\ 0100\ 1000_{74\text{-}2\text{-}1}$

4.1.4

0	0	12	1010	24	10100	36	110110
1	1	13	1011	25	10101	37	110111
2	11	14	1001	26	10111	38	110101
3	10	15	1000	27	10110	39	110100
4	110	16	11000	28	10010	40	111100
5	111	17	11001	29	10011	41	111101
6	101	18	11011	30	10001	42	111111
7	100	19	11010	31	10000	43	111110
8	1100	20	11110	32	110000	44	111010
9	1101	21	11111	33	110001	45	111011
10	1111	22	11101	34	110011	46	111001
11	1110	23	11100	35	110010	47	111000

Table 1.2: The Gray code

The 2 least significant of my number is 47. So, the corresponding Gary code is 111000.

4.1.5

The 8421BCD number from question 3 is $0010\ 0001\ 0110\ 0001\ 0000\ 0100\ 0111_{8421}$.
In this question, I use odd parity code.
Firstly, I send these codes with odd parity code.

0	0	0	1	0
0	0	0	0	1
1	0	1	1	0
0	0	0	0	1
1	0	0	0	0
0	0	1	0	0
0	0	1	1	1
0	**1**	**0**	**0**	**0**

Then the data has been transmitted to another digit system and in the process on bit of the data field has been corrupted.

The data block as received, is:

0	0	0	1	0
0	0	0	0	1

1	0	1	1	0
0	0	0	1	1
1	0	0	0	0
0	0	1	0	0
0	0	1	1	1
0	**1**	**0**	**0**	**0**

Generating the parity from the data alone we get:

0	0	0	1	0
0	0	0	0	1
1	0	1	1	0
***1**	0	0	1	1
1	0	0	0	0
0	0	1	0	0
0	0	1	1	1
0	**1**	**0**	***1**	**0**

The parity failures indicated by * can be identified and the intersection of the row and column locate the error.

The corrected data block is therefore:

0	0	0	1	0
0	0	0	0	1
1	0	1	1	0
0	0	0	0	1
1	0	0	0	0
0	0	1	0	0
0	0	1	1	1
0	**1**	**0**	**0**	**0**

Suppose that two errors are introduced into the data, the block received is:

0	0	0	1	0
0	0	0	0	1
1	0	1	1	0
0	1	1	0	1
1	0	0	0	0
0	0	1	0	0
0	0	1	1	1
0	**1**	**0**	**0**	**0**

Generating the parity from the data alone we get:

0	0	0	1	0
0	0	0	0	1

1	0	1	1	0
0	0	0	0	1
1	0	0	0	0
0	0	1	0	0
0	0	1	1	1
0	***0**	***1**	**0**	**0**

The data is error, but we can't find the specific error.

4.2 Problem 2 (Combinational Logic System)

My birthday is 20th October 2002. After adding, the result is 32. Convert it to 4 BASED number 200_4. Therefore, the F_0 is selected as my function.

4.2.1

Shorthand 1st canonical form equation:

F_0=f(ABCD)= ∑(2,5,7,8,10,12,13,15)

Full 1st canonical form Boolean equation:

$F_0=\bar{A}\bar{B}C\bar{D}+\bar{A}B\bar{C}\bar{D}+\bar{A}BCD+A\bar{B}\bar{C}\bar{D}+A\bar{B}C\bar{D}+AB\bar{C}\bar{D}+AB\bar{C}D+ABCD$

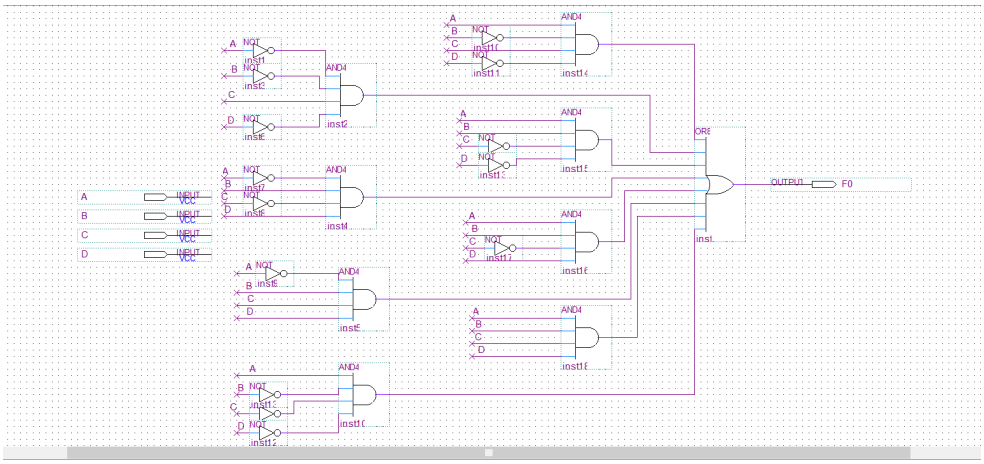


Figure 2.1

.

4.2.2

Shorthand 2nd canonical form equation:

F_0=f(ABCD)= Π(0,1,3,4,6,9,11,14)

Full 2nd canonical form Boolean equation

$F_0=(A+B+C+D).(A+B+C+\bar{D}).(A+B+\bar{C}+\bar{D}).(A+\bar{B}+C+D).(A+\bar{B}+\bar{C}+D).(\bar{A}+B+C+\bar{D}).(\bar{A}+B+\bar{C}+\bar{D}).(\bar{A}+\bar{B}+\bar{C}+D)$

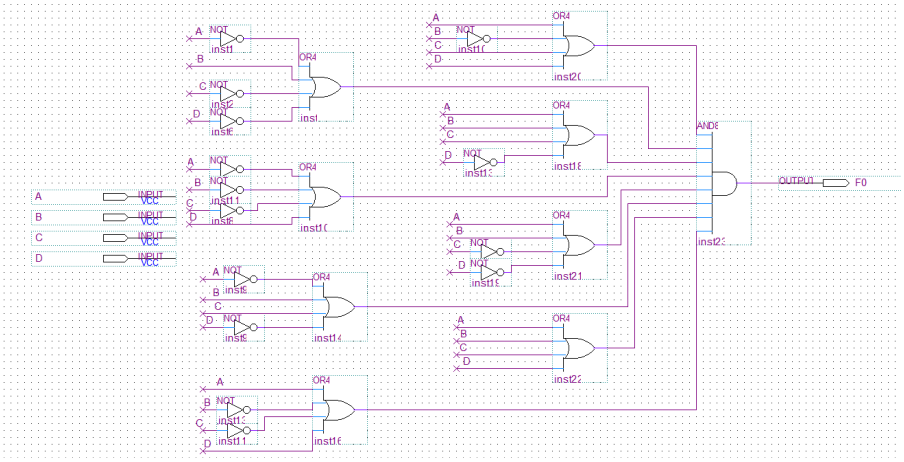


Figure 2.2

4.2.3

F=f(ABCD)= ∑(2,5,7,8,10,12,13,15)

F AB

CD		**00**	**01**	**11**	**10**
	00	0	0	1	1
	01	0	1	1	0
	11	0	1	1	0
	10	1	0	0	1

4.2.4

For the minimal 1st canonical form equation

F=$BD + A\bar{C}\bar{D} + \bar{B}C\bar{D}$

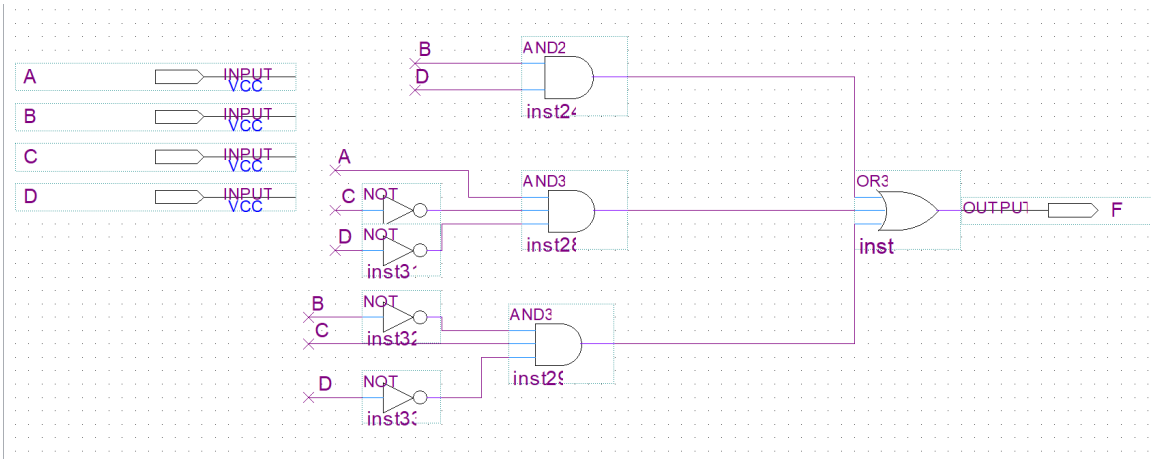


Figure 2.3

For the minimal 2nd canonical form equation

F=$(B+\bar{D}).(A+C+D).(\bar{B}+\bar{C}+D)$

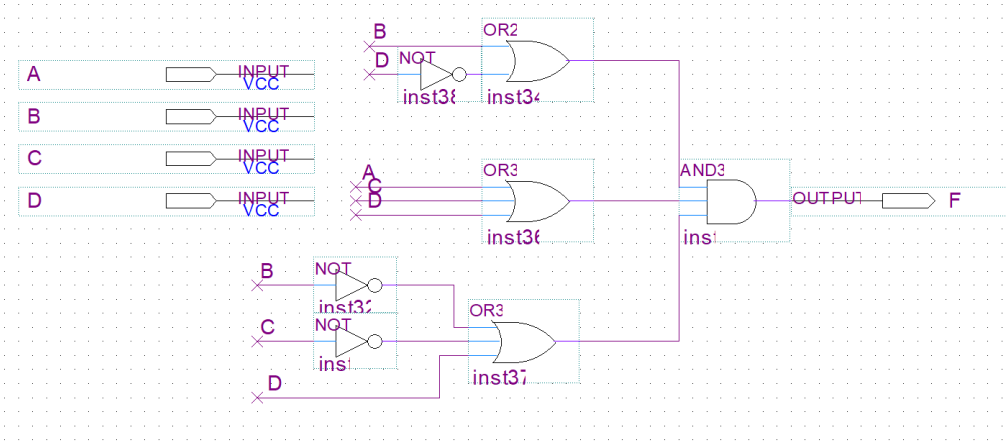


Figure 2.4

4.2.5

A	B	C	D	$\boldsymbol{BD+A\bar{C}\bar{D}+\bar{B}C\bar{D}}$	$\boldsymbol{(B+\bar{D}).(A+C+D).(\bar{B}+\bar{C}+D)}$
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	1	1
0	0	1	1	0	0
0	1	0	0	0	0
0	1	0	1	1	1
0	1	1	0	0	0
0	1	1	1	1	1
1	0	0	0	1	1
1	0	0	1	0	0
1	0	1	0	1	1
1	0	1	1	0	0
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	0	0
1	1	1	1	1	1

Table 2.1 Truth table

According to the truth table, it can be proved that the 1st canonical form and 2nd canonical form are equivalence.

4.2.6

a) To use NAND gates:

F= $BD+A\bar{C}\bar{D}+\bar{B}C\bar{D}$

$=\overline{\overline{BD+A\bar{C}\bar{D}+\bar{B}C\bar{D}}}$

$=\overline{\overline{B.D}.\overline{A.\bar{C}.\bar{D}}.\overline{\bar{B}.C.\bar{D}}}$

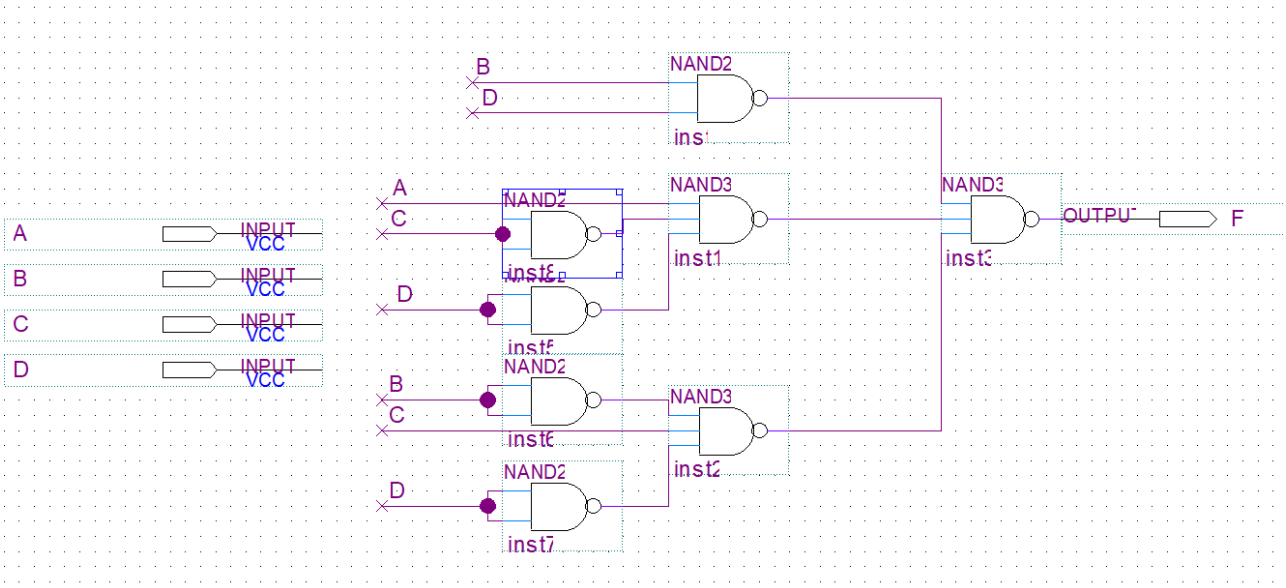


Figure 2.5

b) To use NOR gates:

$$F = \overline{(B + \bar{D}).(A + C + D).(\bar{B} + \bar{C} + D)}$$
$$= \overline{\overline{(B + \bar{D}).(A + C + D).(\bar{B} + \bar{C} + D)}}$$
$$= \overline{\overline{(B + \bar{D})} + \overline{(A + C + D)} + \overline{(\bar{B} + \bar{C} + D)}}$$

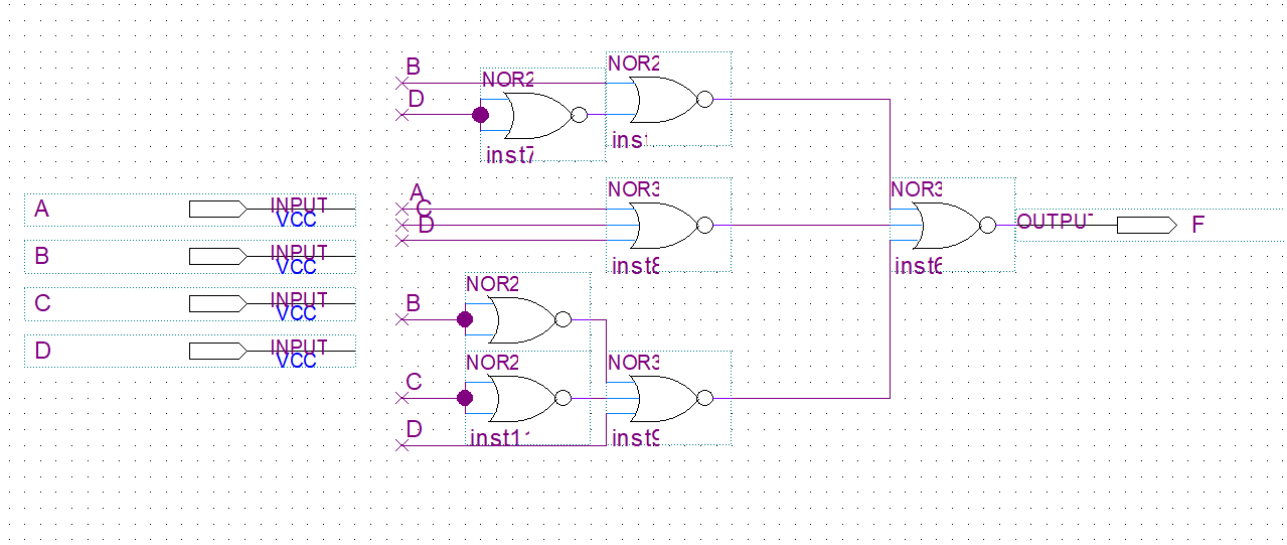


Figure 2.6

4.2.7

Add 4 “don’t care” states in my Karnaugh map. I choose “3,4,6,14” as “don’t care” in the K-map.

F	*AB*				
		00	**01**	**11**	**10**
CD	**00**	0	d	1	1
	01	0	1	1	0
	11	d	1	1	0
	10	1	d	d	1

For the 1st canonical form equation:
$F = B + C\bar{D} + A\bar{B}\bar{D}$

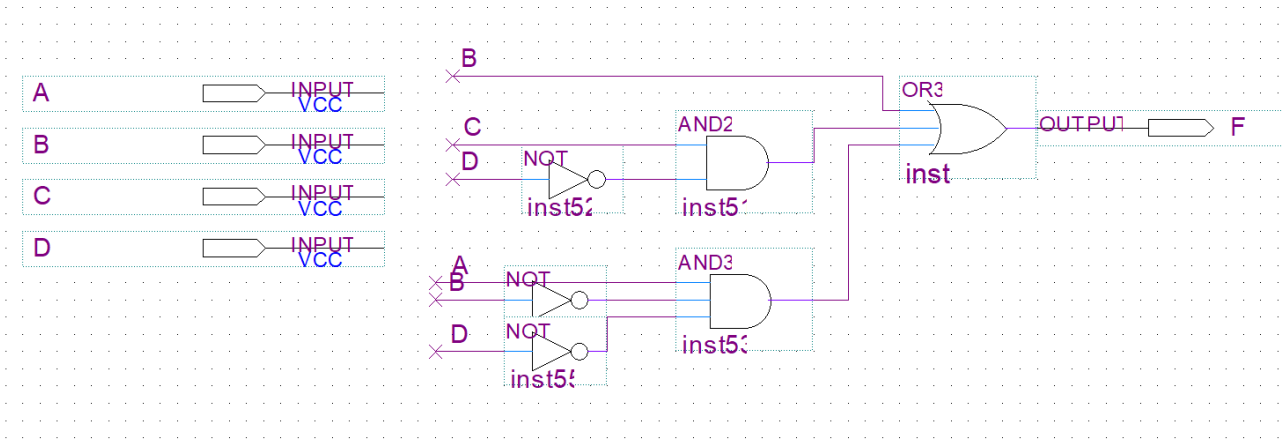


Figure 2.7

For the 2nd canonical form equation:
$F = (\bar{A} + B + \bar{D}).(A + B + C)$

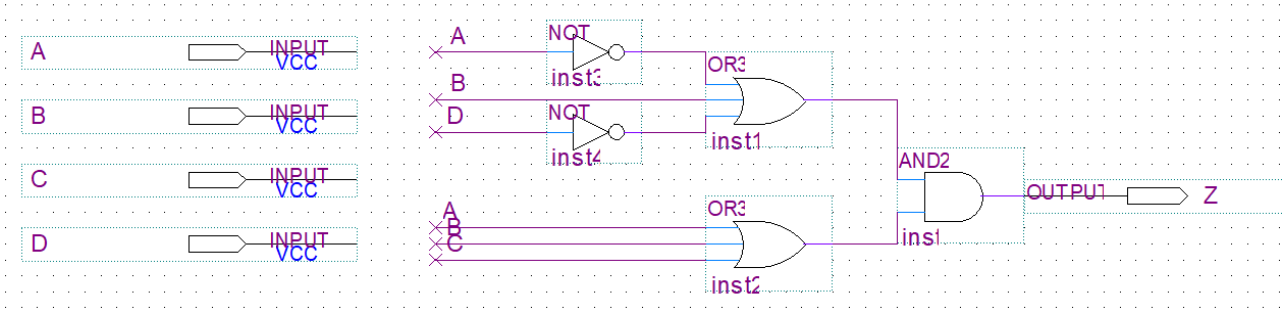


Figure 2.8

4.2.8

a) To use NAND gates:
$F = B + C\bar{D} + A\bar{B}\bar{D}$

$$= \overline{\overline{B + C\bar{D} + A\bar{B}\bar{D}}}$$
$$= \bar{B}.\overline{C\bar{D}}.\overline{A\bar{B}D}$$

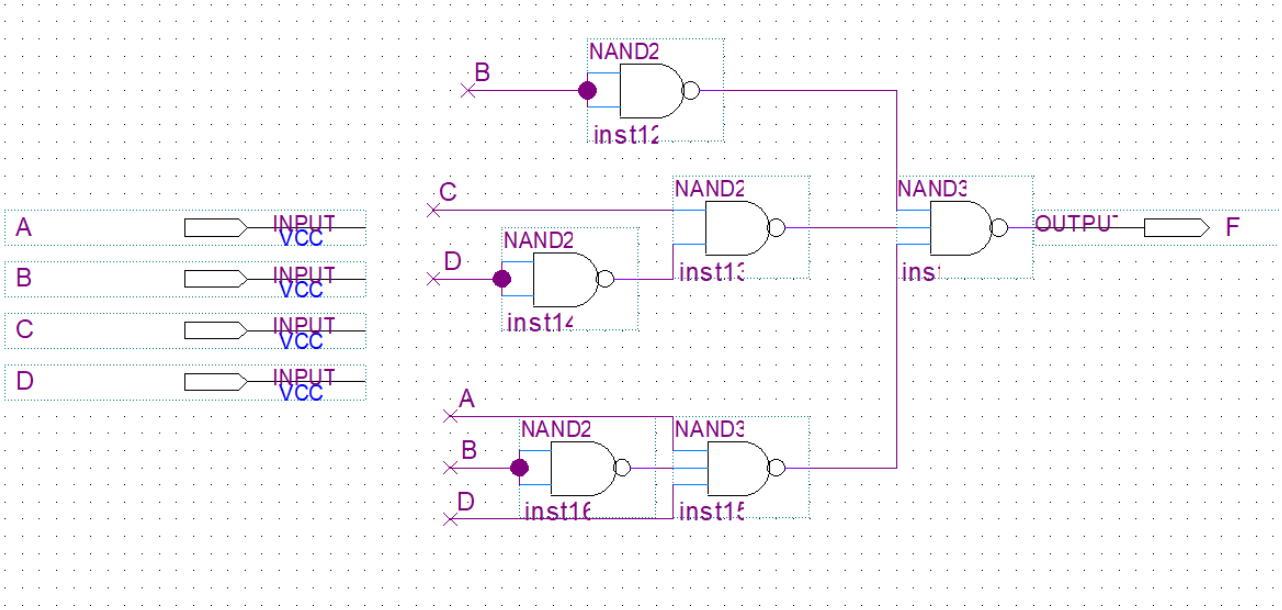


Figure 2.9

b) To use NOR gates:

$$F = (\bar{A} + B + \bar{D}).(A + B + C)$$
$$= \overline{\overline{(\bar{A} + B + \bar{D}).(A + B + C)}}$$
$$= \overline{\overline{(\bar{A} + B + \bar{D})} + \overline{(A + B + C)}}$$

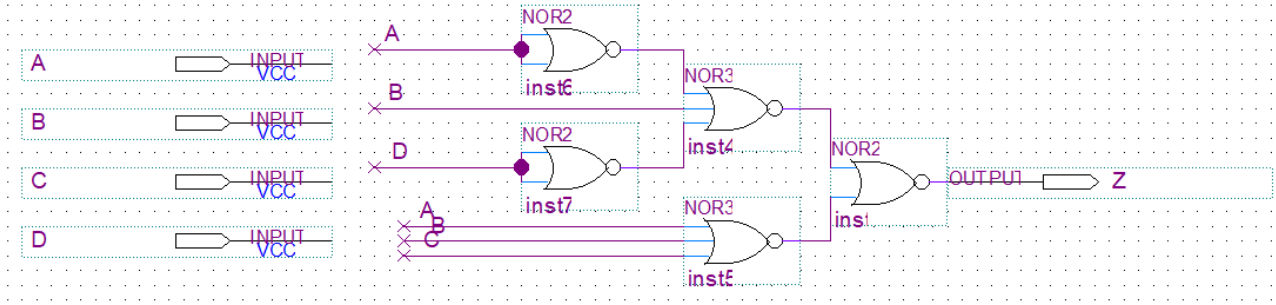


Figure 2.10

4.2.9

The circuit in question 1 and question 2 is too complicated, so it can't be mass produced. The circuit in question 4 and question 7 is be minimized, it can be used for mass production, but it needs a variety of gates. The circuit in question 6 and question 8 only need one kind of gate, so it is very suitable for mass production.

4.2.10

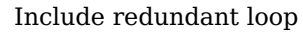
Practical electronic logic circuits are not ideal devices. They require a finite time to operate and consequently introduce delays into the propagation of information. In question 4, some NAND doors have NOT doors in front of them, while others do not. If the inputs are switched from 0 to 1

or 1 to 0, NOT gate need a little time to process this information. But it can cause delay and eventually lead to hazards. Therefore, we need some solutions to deal with the hazards.

For 1st canonical form equation:

F *AB*

CD	**00**	**01**	**11**	**10**
00	0	0	1	1
01	0	1	1	0
11	0	1	1	0
10	1	0	0	1



After modifying the circuits.
$F = BD + A\bar{C}\bar{D} + \bar{B}C\bar{D} + A\bar{B}\bar{D} + AB\bar{C}$
$A\bar{B}\bar{D}$ and $AB\bar{C}$ are redundant terms.

For 2nd canonical form equation:

F *AB*

CD	**00**	**01**	**11**	**10**
00	0	0	1	1
01	0	1	1	0
11	0	1	1	0
10	1	0	0	1

After modifying the circuits.
$F = (B + \bar{D}).(A + C + D).(\bar{B} + \bar{C} + D).(A + B + C).(A + \bar{B} + D)$
$(A + B + C)$ and $(A + \bar{B} + D)$ are redundant terms.

4.3 Problem 3 (Sequential Logic system)

My birthday number adds up to 32, then convert it to 4 BASED number 200_4, the last digit is '0'. Therefore, the first number set (0) {0,3,8,4,5,1,9,7} is selected as my number set.

4.3.1

In this question, I need use 3 asynchronous TFFs. Output logic is very important in the design.

Here is the truth table:

NO.	Q_2	Q_1	Q_0	Z_3	Z_2	Z_1	Z_0
0	0	0	0	0	0	0	0
1	0	0	1	0	0	1	1
2	0	1	0	1	0	0	0
3	0	1	1	0	1	0	0
4	1	0	0	0	1	0	1
5	1	0	1	0	0	0	1
6	1	1	0	1	0	0	1
7	1	1	1	0	1	1	1

Table 3.1 Truth table of output logic

Z_3, Z_2, Z_1, Z_0 are all functions and can be minimized on K-maps.

Z_3 **Q_2Q_1**

		00	**01**	**11**	**10**
Q_0	**00**	0	1	1	0
	01	0	0	0	0

$Z_3 = B\bar{C}$

Z_2 **Q_2Q_1**

		00	**01**	**11**	**10**
Q_0	**00**	0	0	0	1
	01	0	1	1	0

$Z_2 = BC + A\bar{B}\bar{C}$

Z_1 **Q_2Q_1**

		00	**01**	**11**	**10**
Q_0	**00**	0	0	0	0
	01	1	0	1	0

$Z_1 = \bar{A}\bar{B}C + ABC$

Z_0 **Q_2Q_1**

		00	01	11	10
Q_0	**00**	0	0	1	1
	01	1	0	1	1

$$Z_0 = A + \bar{B}C$$

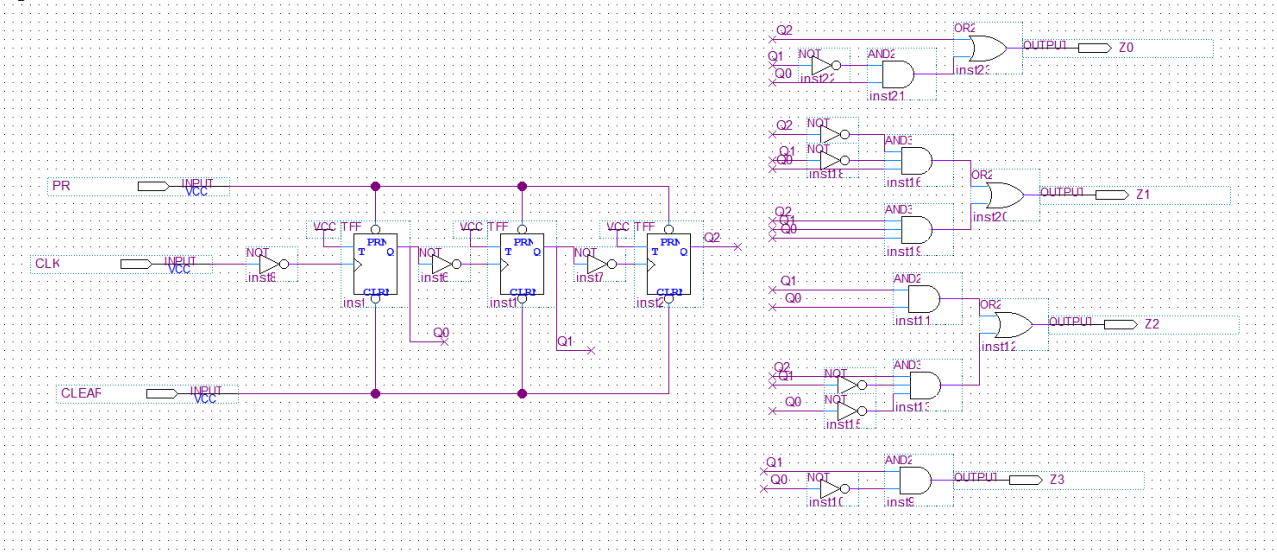


Figure 3.1

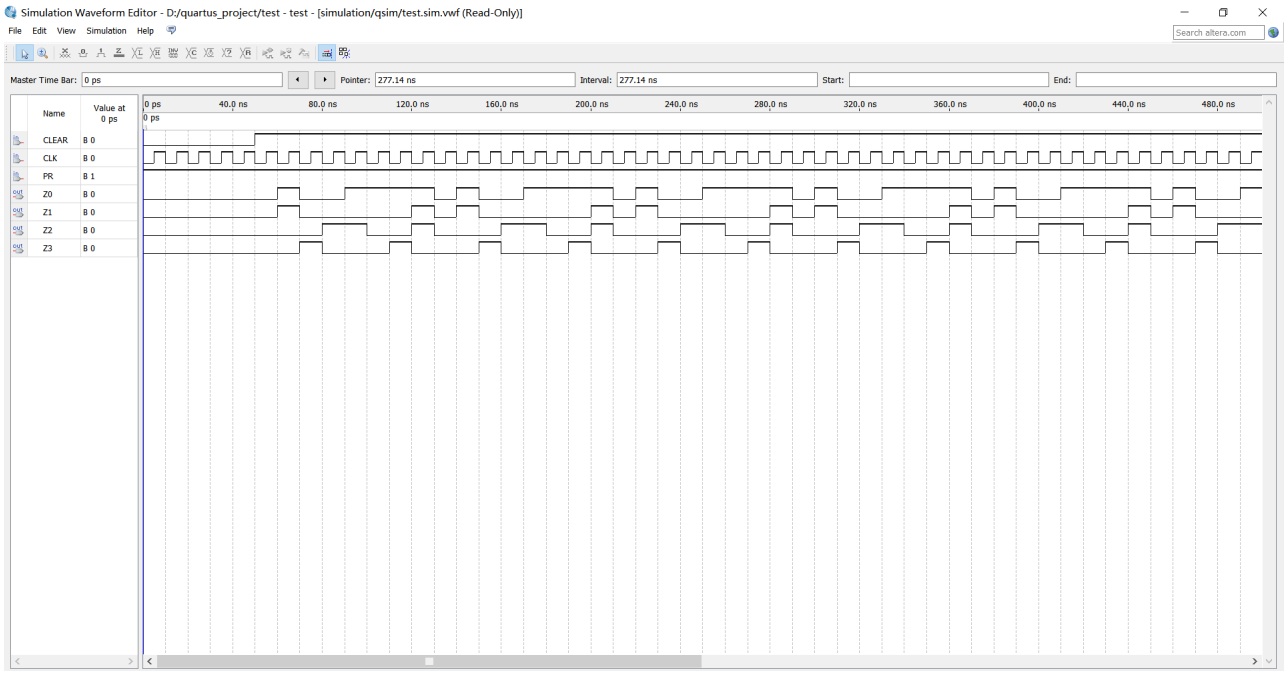


Figure 3.2 simulation

This is my simulation result, after checking the truth table, it is found that the result is correct.

The clock signal in this software is triggered at rising edge, so I need add some NOT gates to convert it to falling edge.

Asynchronous counter creates time problems:

Because these triggers are not synchronous, so it will cause some delay. Hazards arise from unequal propagation paths in combination logic. A delay of 5 nanoseconds to 1 microsecond depending on the type of logic being used is required for the hazards to pass.

For example, "Q_0" has two branches, one is connect with output logic and the other is TFF_1. "Q_1" also has two branches, one is connect with output logic, and the other is TFF_2. The output "Z_3" is the function of Q_1 and Q_0. However, due to the NOT gate, the two lines do not reach the "AND" gate at the same time, it will produce wrong output. We need to wait until the output is right.

4.3.2

i. Asynchronous TFFs with output logic

For synchronous trigger flip-flops.

Present state			Next state		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Table 3.2

TFF transition table:

T	$Q_r \rightarrow Q_{r+1}$
0	$0 \rightarrow 0$
1	$0 \rightarrow 1$
0	$1 \rightarrow 1$
1	$1 \rightarrow 0$

T_0 **Q_2Q_1**

		00	**01**	**11**	**10**
Q_0	**00**	1	1	1	1
	01	1	1	1	1

$T_0 = 1$

T_1 **Q_2Q_1**

		00	**01**	**11**	**10**
Q_0	**00**	0	0	0	0

	01	1	1	1	1

$T_1 = Q_0$

T₂	**Q₂Q₁**				
		00	**01**	**11**	**10**
Q₀	**00**	0	0	0	0
	01	0	1	1	0

$T_2 = Q_1 Q_0$

The following steps are the same as 4.3.1. Draw a truth table and obtain the K-map.

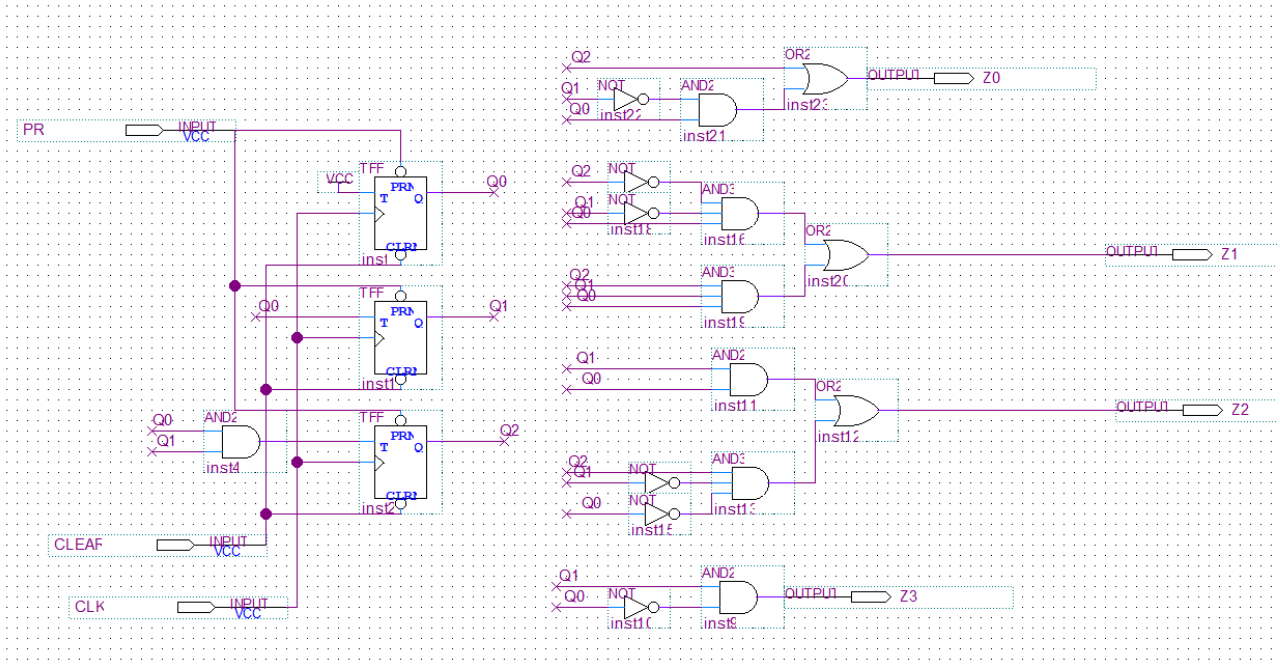


Figure 3.3

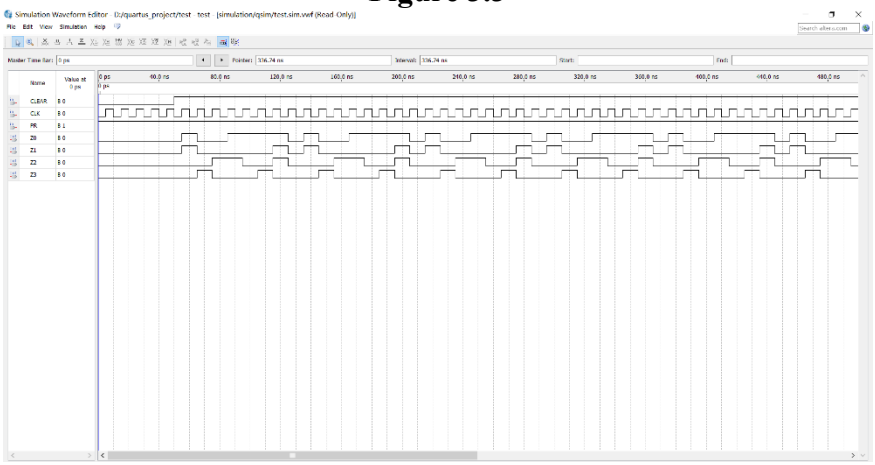


Figure 3.4 simulation

After simulation, I got a waveform. After checking against the truth table, the result is right.

ii. Asynchronous TFFs without output logic

	Present state				*Next state*			
No.	**Q_3**	**Q_2**	**Q_1**	**Q_0**	**Q_3**	**Q_2**	**Q_1**	**Q_0**
0	0	0	0	0	0	0	1	1
1	0	0	0	1	1	0	0	1
2	0	0	1	0	d	d	d	d
3	0	0	1	1	1	0	0	0
4	0	1	0	0	0	1	0	1
5	0	1	0	1	0	0	0	1
6	0	1	1	0	d	d	d	d
7	0	1	1	1	d	d	d	d
8	1	0	0	0	0	1	0	0
9	1	0	0	1	0	1	1	1
10	1	0	1	0	d	d	d	d
11	1	0	1	1	d	d	d	d
12	1	1	0	0	d	d	d	d
13	1	1	0	1	d	d	d	d
14	1	1	1	0	d	d	d	d
15	1	1	1	1	d	d	d	d

Table 3.3

I need 4 TFFs in this diagram:

T_0 Q_3Q_2

Q_1Q_0	**00**	**01**	**11**	**10**
00	1	1	d	0
01	0	0	d	0
11	1	d	d	d
10	d	d	d	d

$T_0 = \overline{Q_3}.\overline{Q_0} + Q_1$

T_1 Q_3Q_2

Q_1Q_0	**00**	**01**	**11**	**10**
00	1	0	d	0
01	0	0	d	1
11	1	d	d	d
10	d	d	d	d

$T_1 = \overline{Q_3}.\overline{Q_2}.\overline{Q_0} + Q_1 + Q_3Q_0$

T_2 Q_3Q_2

Q_1Q_0		**00**	**01**	**11**	**10**
	00	0	0	d	1
	01	0	1	d	1
	11	0	d	d	d
	10	d	d	d	d

$T_2 = Q_2Q_0 + Q_3$

T_3 Q_3Q_2

Q_1Q_0		**00**	**01**	**11**	**10**
	00	0	0	d	1
	01	1	0	d	1
	11	1	d	d	d
	10	d	d	d	d

$T_3 = \overline{Q_2}Q_0 + Q_3\backslash$

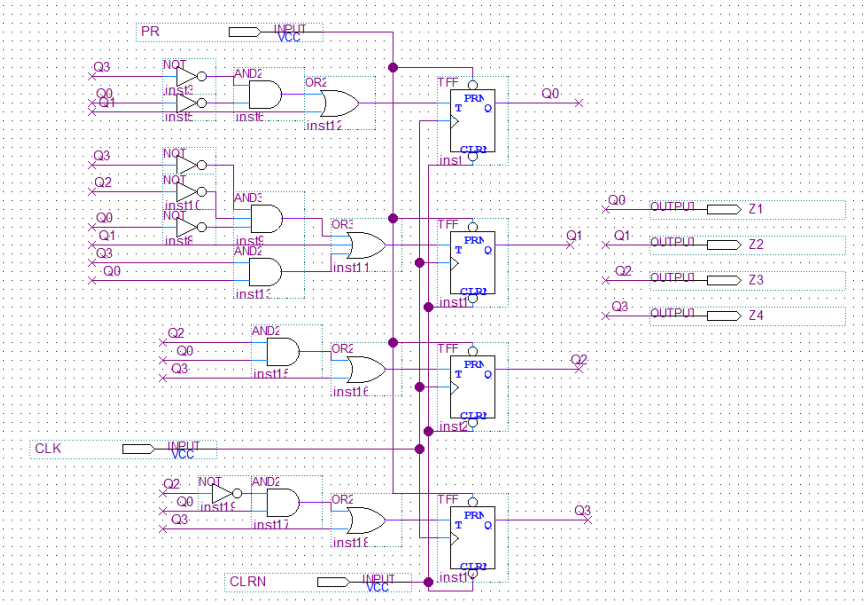


Figure 3.5

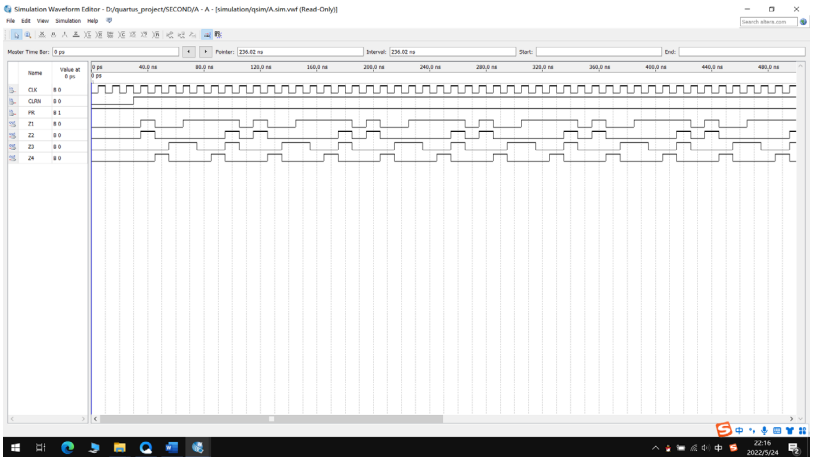


Figure 3.6 simulation

After simulation, I got a waveform. After checking against the truth table, the result is right.

In question 1, I use some asynchronous TFFs to combine the circuit. The specific number is 3 TFFs, 6 AND gates, 3 OR gates, 6 NOT gates. Compared with the first question, in the second question, I use some synchronous TFFs to combine the circuit. The number of hardware used in the output logic is the same as that in question 1, but some gates need to be used in the next state logic.

Therefore, compared with the first question, the second question is not as good as question 1 in the terms of the amount of hardware required and ease of design.

However, because of question 2 use synchronous TFFs, it does not need to worry about timing problems. Multiple synchronous TFFs are controlled by one clock signal, so the system runs synchronously.

4.3.3

After converting my original number set, the new number set A is “3,8,4,5,5,1,9”.

Present state			Next state		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	0	0	0

Table 3.4

No.	Q			Z			
	Q_2	Q_1	Q_0	Z_3	Z_2	Z_1	Z_0
0	0	0	0	0	0	1	1
1	0	0	1	1	0	0	0
2	0	1	0	0	1	0	0
3	0	1	1	0	1	0	1
4	1	0	0	0	1	0	1
5	1	0	1	0	0	0	1
6	1	1	0	1	0	0	1

Table 3.5

J	K	$Q_r \rightarrow Q_{r+1}$

0	d	0 → 0
1	d	0 → 1
d	0	1 → 1
d	1	1 → 0

J_0 **Q_2Q_1**

		00	01	11	10
Q_0	**00**	1	1	0	1
	01	d	d	d	d

$J_0 = \overline{Q_2} + \overline{Q_1}$

K_0 **Q_2Q_1**

		00	01	11	10
Q_0	**00**	d	d	d	d
	01	1	1	d	1

$K_0 = 1$

J_1 **Q_2Q_1**

		00	01	11	10
Q_0	**00**	0	d	d	0
	01	1	d	d	1

$J_1 = Q_0$

K_1 **Q_2Q_1**

		00	01	11	10
Q_0	**00**	d	0	1	d
	01	d	1	d	d

$K_1 = Q_0 + Q_2$

J_2 **Q_2Q_1**

		00	01	11	10
Q_0	**00**	0	0	d	d
	01	0	1	d	d

$J_1 = Q_1Q_0$

K_2 **Q_2Q_1**

		00	01	11	10
Q_0	**00**	d	d	1	0
	01	d	d	d	0

$K_2 = Q_1$

Z_3, Z_2, Z_1, Z_0 are all functions and can be minimized on K-maps.

Z_3 **Q_2Q_1**

Q_0		00	01	11	10
	00	0	0	1	0
	01	1	0	d	0

$Z_3 = (\overline{Q_2} + Q_1).(Q_2 + Q_0).(Q_2 + \overline{Q_1})$

Z_2 **Q_2Q_1**

Q_0		00	01	11	10
	00	0	1	0	1
	01	0	1	d	0

$Z_2 = (Q_2 + Q_1).(\overline{Q_2} + \overline{Q_1}).(\overline{Q_2} + \overline{Q_0})$

Z_1 **Q_2Q_1**

Q_0		00	01	11	10
	00	1	0	0	0
	01	0	0	d	0

$Z_1 = \overline{Q_2}.\overline{Q_1}.\overline{Q_0}$

Z_0 **Q_2Q_1**

Q_0		00	01	11	10
	00	1	0	1	1
	01	0	1	d	1

$Z_0 = (Q_2 + Q_1 + \overline{Q_0}).(Q_2 + \overline{Q_1} + Q_0)$

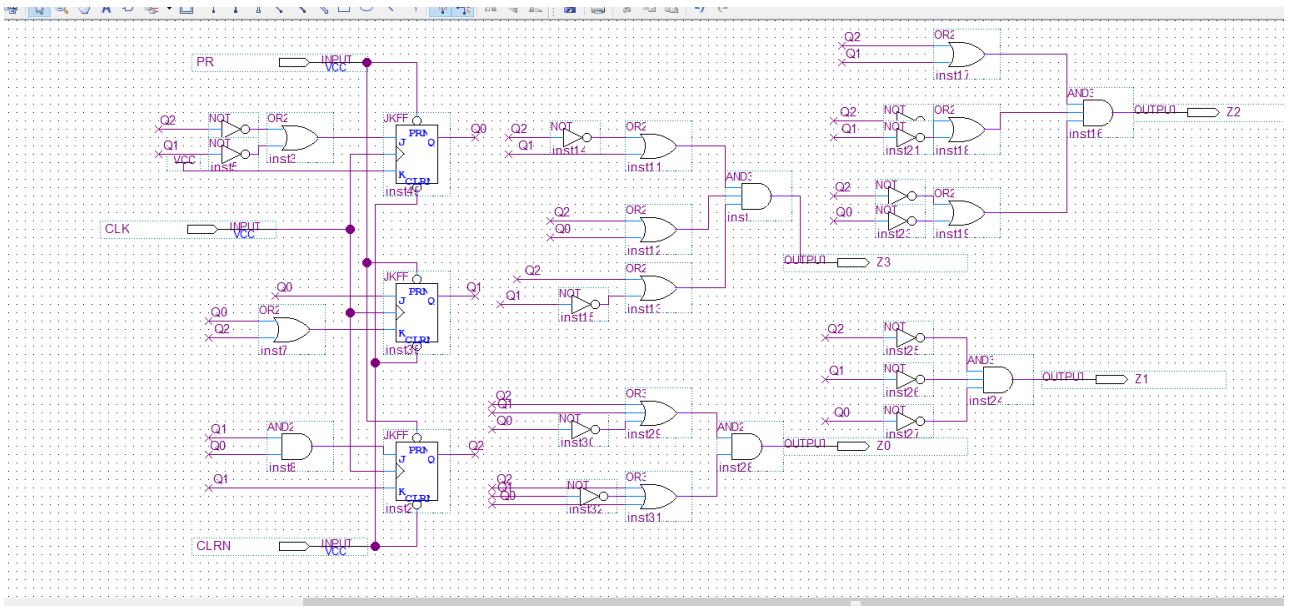


Figure 3.7

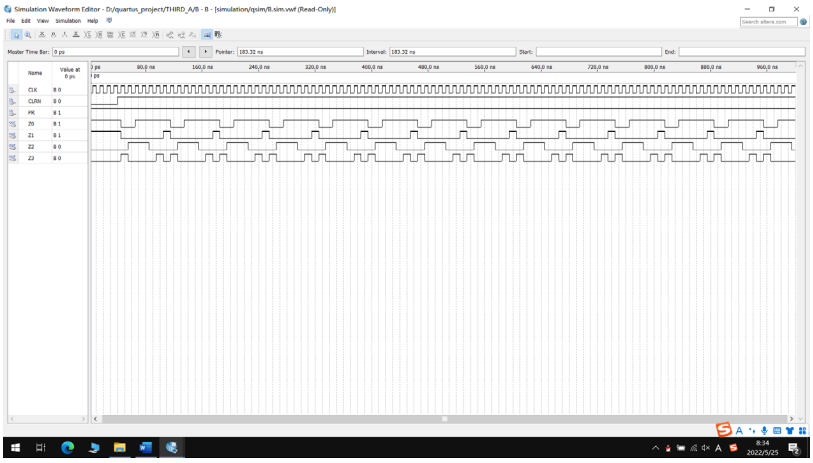


Figure 3.8 simulation

After simulation, I got a waveform. After checking against the truth table, the result is right.

Then modify the system so that the sequence stops at the end of my new number set, after having output all the digit once. I want to add a state logic, set the present state and next state to “1 1 1”, so that the system cycle in this state all the time, and the output set to “0 0 0 0”.

Present state			Next state		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	1	1	1

Table 3.7

No.	Q			Z			
	Q_2	Q_1	Q_0	Z_3	Z_2	Z_1	Z_0
0	0	0	0	0	0	1	1
1	0	0	1	1	0	0	0
2	0	1	0	0	1	0	0
3	0	1	1	0	1	0	1
4	1	0	0	0	1	0	1
5	1	0	1	0	0	0	1
6	1	1	0	1	0	0	1
7	1	1	1	0	0	0	0

Table 3.8

J_0 **Q_2Q_1**

Q_0	00	01	11	10
00	1	1	0	1
01	d	d	d	d

$J_0 = \overline{Q_2} + \overline{Q_1}$

K_0 **Q_2Q_1**

Q_0	00	01	11	10
00	d	d	d	d
01	1	1	0	1

$K_0 = \overline{Q_2} + \overline{Q_1}$

J_1 **Q_2Q_1**

Q_0	00	01	11	10
00	0	d	d	0
01	1	d	d	1

$J_1 = Q_0$

K_1 **Q_2Q_1**

Q_0	00	01	11	10
00	d	0	1	d
01	d	1	0	d

$K_1 = (Q_0 + Q_2).(\overline{Q_2} + \overline{Q_1})$

J_2 **Q_2Q_1**

Q_0	00	01	11	10
00	0	0	d	d
01	0	1	d	d

$J_1 = Q_1Q_0$

K_2 **Q_2Q_1**

Q_0	00	01	11	10
00	d	d	1	0
01	d	d	0	0

$K_2 = Q_1\overline{Q_0}$

Z_3, Z_2, Z_1, Z_0 are all functions and can be minimized on K-maps.

Z_3 **Q_2Q_1**

Q_0	00	01	11	10
00	0	0	1	0
01	1	0	0	0

$Z_3 = (\overline{Q_2} + Q_1).(Q_2 + Q_0).(Q_2 + \overline{Q_1}).(\overline{Q_2} + \overline{Q_0})$

Z_2 **Q_2Q_1**

Q_0	00	01	11	10
00	0	1	0	1
01	0	1	0	0

$Z_2 = (Q_2 + Q_1).(\overline{Q_2} + \overline{Q_1}).(\overline{Q_2} + \overline{Q_0})$

Z_1		Q_2Q_1			
		00	**01**	**11**	**10**
Q_0	**00**	1	0	0	0
	01	0	0	0	0

$Z_1 = \overline{Q_2}.\overline{Q_1}.\overline{Q_0}$

Z_0		Q_2Q_1			
		00	**01**	**11**	**10**
Q_0	**00**	1	0	1	1
	01	0	1	0	1

$Z_0 = (Q_2 + Q_1 + \overline{Q_0}).(Q_2 + \overline{Q_1} + Q_0).(\overline{Q_2} + \overline{Q_1} + \overline{Q_0})$

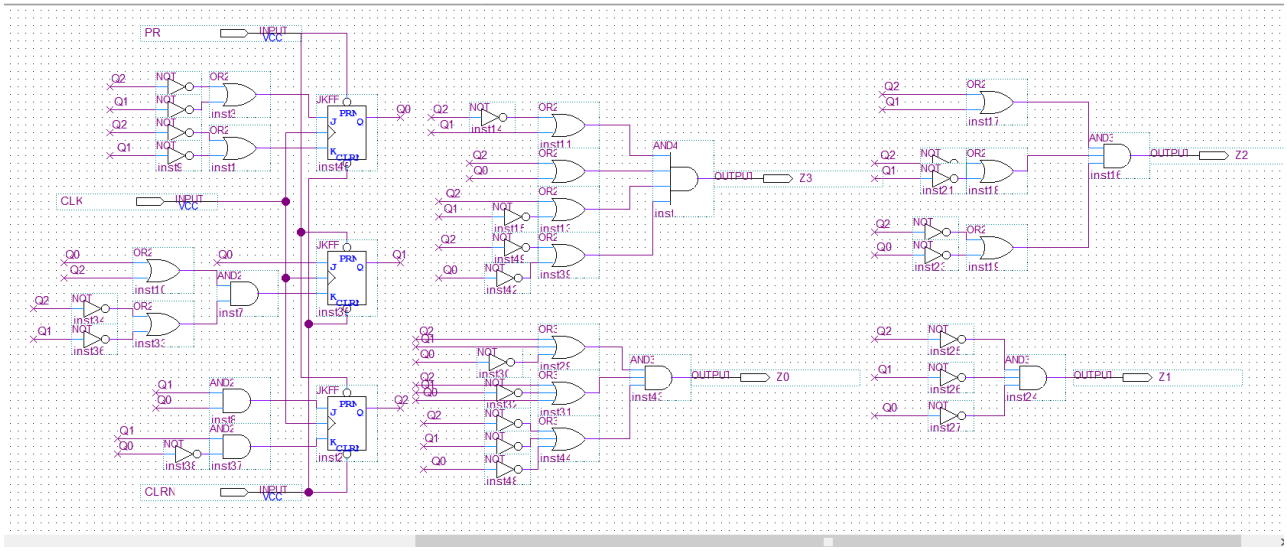


Figure 3.9

Therefore, in these K-maps and Boolean functions, I can find that some of the hardware of the circuit need to be replaced. In the next state logic part, the logic of all "J" interfaces does not need to be modified, but the logic of all "K" interfaces need to be modified.

In the output logic part, the logic of Z_3 and Z_0 need to be modified.

4.3.4

				Next state					
				I=0			I=1		
Present	0	0	0	1	1	0	0	0	1
State	0	0	1	0	0	0	0	1	0
$Q_2Q_1Q_0$	0	1	0	0	0	1	0	1	1
	0	1	1	0	1	0	1	0	0

1	0	0	0	1	1	1	0	1
1	0	1	1	0	0	1	1	0
1	1	0	1	0	1	0	0	0

Table 3.9

No.	Q			Z			
	Q_2	Q_1	Q_0	Z_3	Z_2	Z_1	Z_0
0	0	0	0	0	0	1	1
1	0	0	1	1	0	0	0
2	0	1	0	0	1	0	0
3	0	1	1	0	1	0	1
4	1	0	0	0	1	0	1
5	1	0	1	0	0	0	1
6	1	1	0	1	0	0	1

Table 3.10

D_0 Q_3Q_2

Q_1Q_0	**00**	**01**	**11**	**10**
00	0	1	1	1
01	0	0	0	0
11	0	d	d	0
10	1	1	0	1

$D_0 = \bar{I}Q_1\overline{Q_0} + \bar{I}Q_2\overline{Q_0} + I\overline{Q_1}.\overline{Q_0} + I\overline{Q_2}.\overline{Q_0}$

D_1 Q_3Q_2

Q_1Q_0	**00**	**01**	**11**	**10**
00	0	1	0	0
01	1	1	1	1
11	1	d	d	1
10	0	0	0	0

$D_1 = \bar{I}Q_2\overline{Q_1} + Q_0$

D_2 Q_3Q_2

Q_1Q_0	00	01	11	10
00	1	0	1	0
01	0	1	1	0
11	0	d	d	1
10	0	1	0	0

$D_2= \overline{I\overline{Q_2}}.\overline{Q_1}.\overline{Q_0} + IQ_2\overline{Q_1} + IQ_1Q_0 + \bar{I}Q_2Q_1 + Q_2Q_0$

Z_3, Z_2, Z_1, Z_0 are all functions and can be minimized on K-maps.

Z_0 **Q_2Q_1**

Q_0	00	01	11	10
00	1	0	1	1
01	0	1	d	1

$Z_0= \overline{Q_2}Q_1Q_0 + Q_2\overline{Q_0} + Q_2\overline{Q_1} + \overline{Q_1}.\overline{Q_0}$

Z_1 **Q_2Q_1**

Q_0	00	01	11	10
00	1	0	0	0
01	0	0	d	0

$Z_1= \overline{Q_2}.\overline{Q_1}.\overline{Q_0}$

Z_2 **Q_2Q_1**

Q_0	00	01	11	10
00	0	1	0	1
01	0	1	d	0

$Z_2= \overline{Q_2}Q_1 + Q_2\overline{Q_1}.\overline{Q_0}$

Z_3 **Q_2Q_1**

Q_0	00	01	11	10
00	0	0	1	0
01	1	0	d	0

$Z_3= Q_2Q_1\overline{Q_0} + \overline{Q_2}.\overline{Q_1}Q_0$

Convert these Boolean equations to NAND form.

$D_0= \overline{\overline{I\overline{Q_1}\overline{Q_0} + I\overline{Q_2}\overline{Q_0} + I\overline{Q_1}.\overline{Q_0} + I\overline{Q_2}.\overline{Q_0}}}$

$= \overline{\overline{I\overline{Q_1}\overline{Q_0}}.\overline{I\overline{Q_2}\overline{Q_0}}.\overline{I\overline{Q_1}.\overline{Q_0}}.\overline{I\overline{Q_2}.\overline{Q_0}}}$

$D_1= \overline{\overline{I\overline{Q_2}\overline{Q_1} + Q_0}}$

$= \overline{\overline{I\overline{Q_2}\overline{Q_1}}.\overline{Q_0}}$

$D_2= \overline{\overline{I\overline{Q_2}.\overline{Q_1}.\overline{Q_0} + IQ_2\overline{Q_1} + IQ_1Q_0 + \bar{I}Q_2Q_1 + Q_2Q_0}}$

$$= \overline{\overline{I\overline{Q_2}}.\overline{Q_1}.\overline{Q_0}}.\overline{I\overline{Q_2}\overline{Q_1}}.\overline{I\overline{Q_1}\overline{Q_0}}.\overline{\overline{I}\overline{Q_2}\overline{Q_1}}.\overline{\overline{Q_2}\overline{Q_0}}$$

$$Z_0 = \overline{\overline{Q_2}Q_1Q_0 + Q_2\overline{Q_0} + Q_2\overline{Q_1} + \overline{Q_1}.\overline{Q_0}}$$

$$= \overline{\overline{\overline{Q_2}Q_1Q_0}.\overline{Q_2\overline{Q_0}}.\overline{Q_2\overline{Q_1}}.\overline{\overline{Q_1}.\overline{Q_0}}}$$

$$Z_1 = \overline{\overline{Q_2}.\overline{Q_1}.\overline{Q_0}}$$

$$Z_2 = \overline{\overline{Q_2}Q_1 + Q_2\overline{Q_1}.\overline{Q_0}}$$

$$= \overline{\overline{\overline{Q_2}Q_1}.\overline{Q_2\overline{Q_1}}.\overline{Q_0}}$$

$$Z_3 = \overline{Q_2Q_1\overline{Q_0} + \overline{Q_2}.\overline{Q_1}Q_0}$$

$$= \overline{\overline{Q_2Q_1\overline{Q_0}}.\overline{\overline{Q_2}.\overline{Q_1}Q_0}}$$

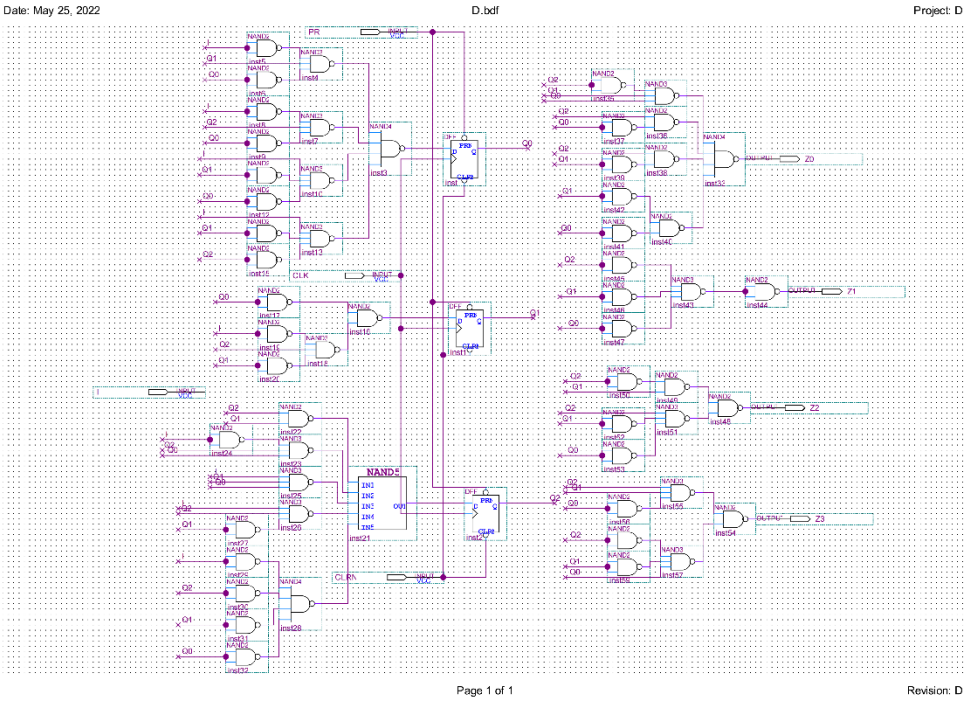


Figure 3.10

4.3.5

My original number set is “0,3,8,4,5,1,9,7”, so after converting, the new number set is “3,8,4,5,1”, the 5-bit binary number is “10011”.

State transition diagram:

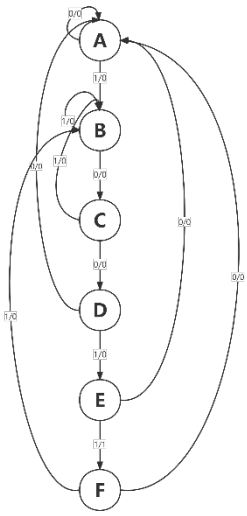


Figure 3.11 State transition diagram

The present/next table is:

		Next state	
		I=0	I=1
Present state	A	A	B
	B	C	B
	C	D	B
	D	A	E
	E	A	F
	F	A	B

Table 3.11

Assigning binary values to the internal states. (let A=000, B=001, C=010, D=011, E=100, F=101).

We obtain:

		Next state	
		I=0	I=1
Present state	000	000	001
	001	010	001
	010	011	001
	011	000	100
	100	000	101
	101	000	001

Table 3.12

The next state logic requires 3DFFs and the driving functions may be derived from K-maps as shown below:

D_2 IQ_2

Q_1Q_0	**00**	**01**	**11**	**10**
00	0	0	1	0
01	0	0	0	0
11	0	d	d	1
10	0	d	d	0

$D_2 = I.(\bar{I} + Q_1 + \overline{Q_0}).(\bar{I} + Q_2 + Q_0)$

D_1 IQ_2

Q_1Q_0	**00**	**01**	**11**	**10**
00	0	0	0	0
01	1	0	0	0
11	0	d	d	0
10	1	d	d	0

$D_1 = \bar{I}.\overline{Q_2}.(Q_1 + Q_0).(\overline{Q_1} + \overline{Q_0})$

D_1 IQ_2

Q_1Q_0	**00**	**01**	**11**	**10**
00	0	0	1	1
01	0	0	1	1
11	0	d	d	0
10	1	d	d	1

$D_0 = (I + Q_1).(\overline{Q_1} + \overline{Q_0})$

The output function is:

$Z=IQ_2\overline{Q_1}Q_0$

Convert it to NOR form:

$D_2=\overline{\overline{I}.(\overline{I}+Q_1+\overline{Q_0}).(\overline{I}+Q_2+Q_0)}$

$=\overline{\overline{I}+\overline{(\overline{I}+Q_1+\overline{Q_0})}+\overline{(\overline{I}+Q_2+Q_0)}}$

$D_1=\overline{\overline{I}.\overline{Q_2}.(Q_1+Q_0).(\overline{Q_1}+\overline{Q_0})}$

$=\overline{I+Q_2+\overline{(Q_1+Q_0)}+\overline{(\overline{Q_1}+\overline{Q_0})}}$

$D_0=\overline{\overline{(I+Q_1).(\overline{Q_1}+\overline{Q_0})}}$

$=\overline{\overline{(I+Q_1)}+\overline{(\overline{Q_1}+\overline{Q_0})}}$

$Z=\overline{\overline{IQ_2\overline{Q_1}Q_0}}$

$=\overline{\overline{I}+\overline{Q_2}+Q_1+\overline{Q_0}}$

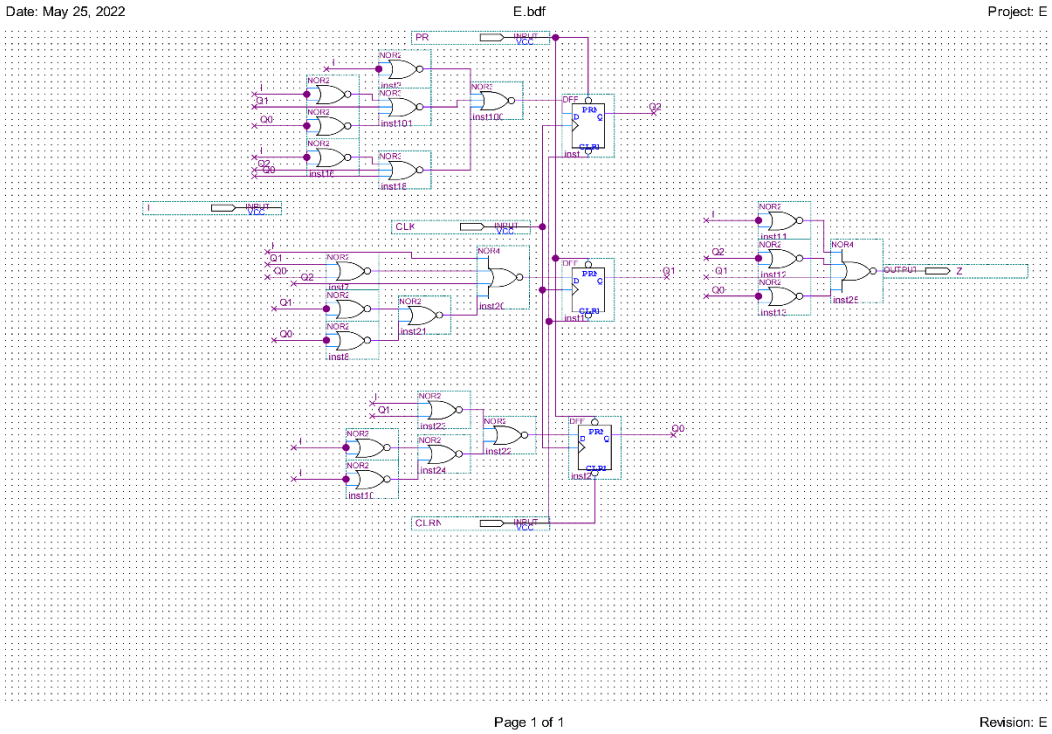


Figure 3.12

References

[1] T.J.Sthonham. *Digital Logic Techniques – principles and practice (Second edition)*. Published by Chapman & Hall, 1987.

[2] Zou Hong. *Digital system and logic design (Second edition)*. Posts & Telecom press, 2017.