

CSE140 Exercise Solutions

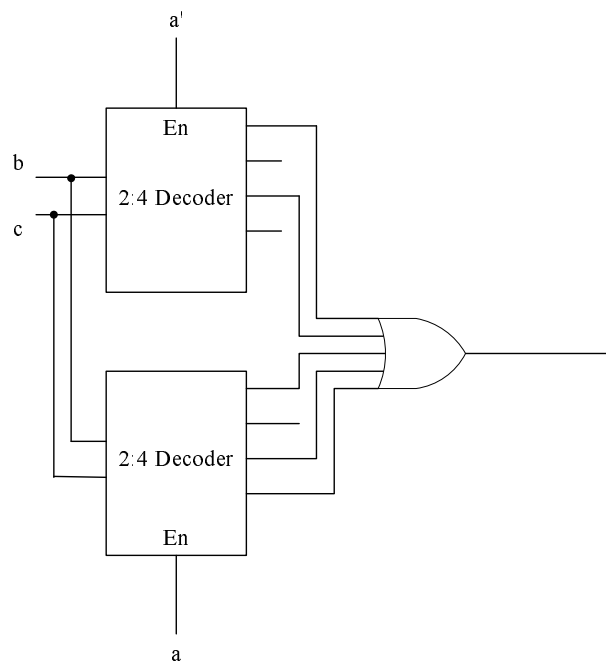
I. Given a three-input Boolean function $f(a, b, c) = \sum m(0, 2, 4, 6, 7) + \sum d(1)$.

a. Implement the function using a minimal network of 2:4 decoders and OR gates.

Standard solution:

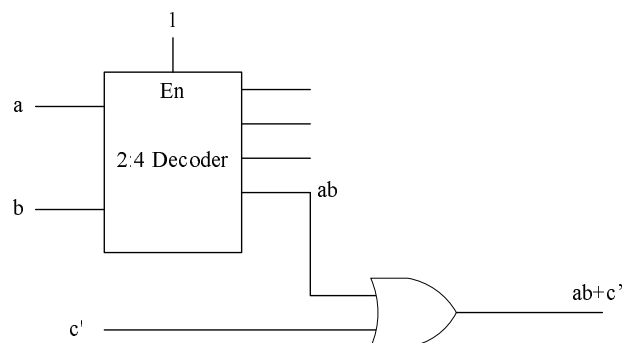
Use a as the enable signal, the truth table is:

a	bc			
	00	01	10	11
0	1	x	1	0
1	1	0	1	1



A more clever solution:

$f(a, b, c) = ab + c'$ (From K-Map)

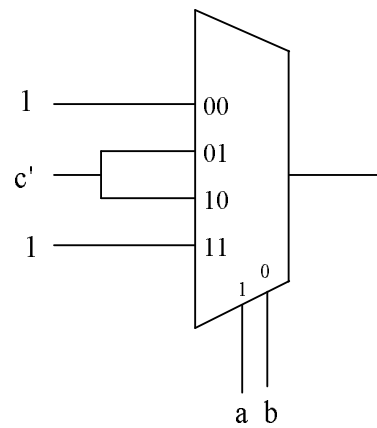


- b. Implement the function using a minimal network of 4:1 multiplexers.

The truth table using a, b as the selection signals:

a	b	$c = 0$	$c = 1$
0	0	1	x
0	1	1	0
1	0	1	0
1	1	1	1

$$f(a, b, c) = a'b' \cdot 1 + a'b \cdot c' + ab' \cdot c' + ab \cdot 1$$



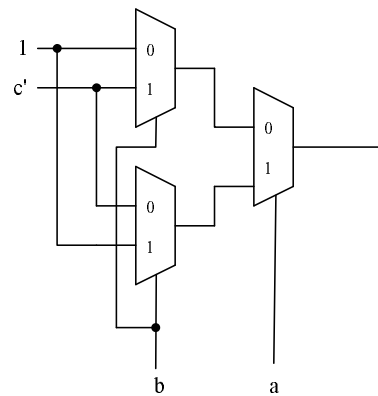
- c. Implement the function using a minimal network of 2:1 multiplexers.

Standard Solution:

The truth table using a and b as the selection signals:

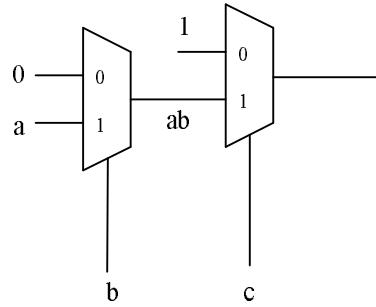
a	b	$c = 0$	$c = 1$
0	0	1	x
0	1	1	0
1	0	1	0
1	1	1	1

$$f(a, b, c) = a'(b' \cdot 1 + b \cdot c') + a(b' \cdot c' + b \cdot 1)$$



A more clever solution:

$$f(a, b, c) = ab + c' = c \cdot ab + c' \cdot 1 = c(b \cdot a + b' \cdot 0) + c' \cdot 1$$



II. Adders: Draw the logic diagram to show the following designs.

II.(1). Design a full adder with a minimal number of 2:1 multiplexers (no other gates). Draw the schematic diagram.

Truth table from a, b, c_i to s, c_o :

a	b	c_i	s	c_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth table for s , using a, b as the selection signals:

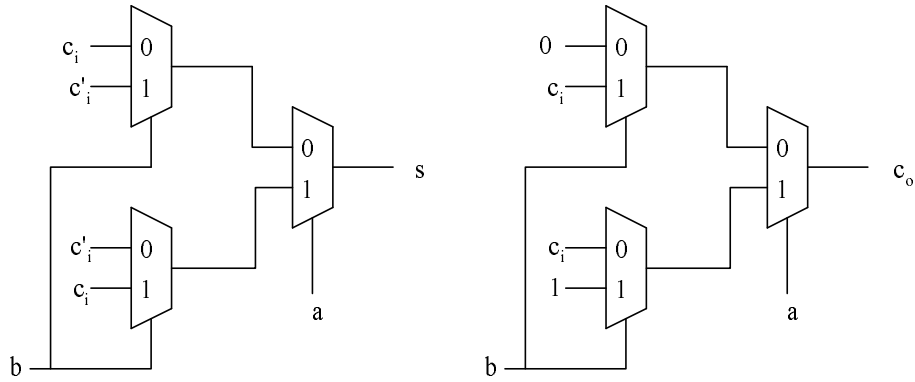
a	b	$c_i = 0$	$c_i = 1$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

$$s = a'b'c_i + a'bc_i + ab'c'_i + abc_i = a'(b'c_i + bc'_i) + a(b'c'_i + bc)$$

Truth table for c_o , using a, b as the selection signals:

a	b	$c_i = 0$	$c_i = 1$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

$$c_o = a'b' \cdot 0 + a'bc_i + ab'c_i + ab \cdot 1$$

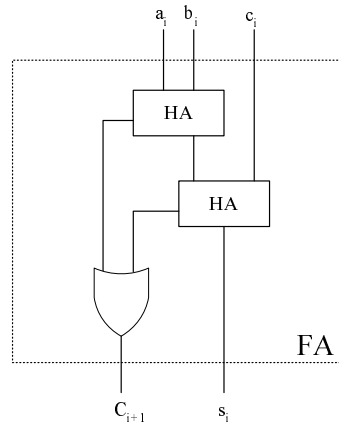


II.(2). A sequential three-at-a-time adder inputs a_i, b_i, e_i , the i 'th bit of three binary numbers in each clock cycle for $i = 0$ to $n-1$ and outputs the sum. Implement the adder with a minimal numbers of Half Adders, OR gates and two D flip-flops.

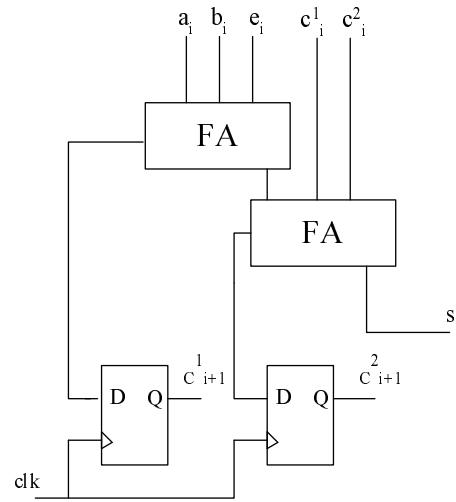
Note: Adding 3 digits a_i, b_i, e_i and carry-in c_i can generate a carry-out up to 2 bits.

Solution 1: Represent the 2-bit carry out of the current cycle in the same weight, and add them with the three bits input.

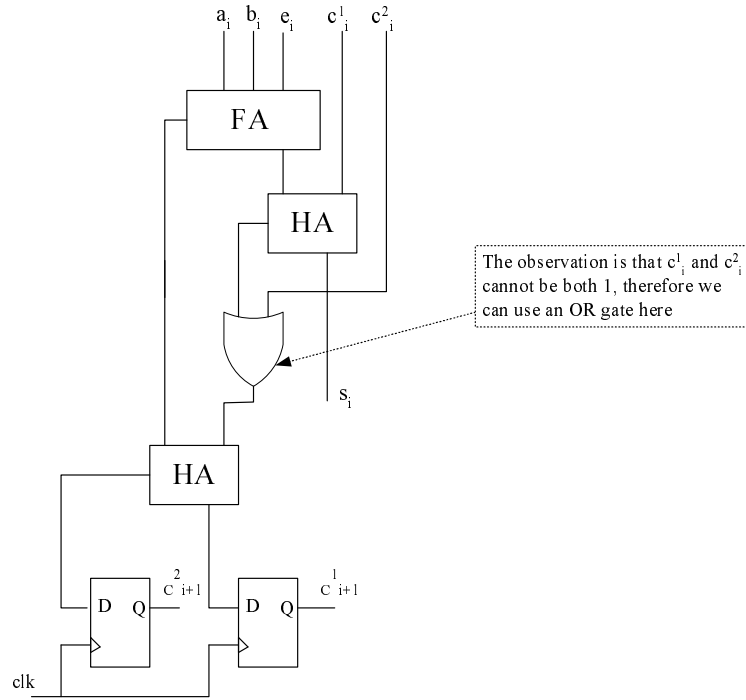
We know that a full adder (FA) can be built using two half adders (HA):



Using two full adders we can build the three-at-a-time adder:



Solution 2: Represent the 2-bit carry as a binary number. Then we have 3 possible cases: $c_i^2 c_i^1 = 00, 01, 10$. We add c_i^1 together with a_i, b_i, e_i and add c_i^2 to the carry.



III. A sequential adder inputs a_i, b_i , the i 'th bit of two binary numbers in each clock cycle for $i = 0$ to $n - 1$ and outputs the sum. Implement the adder with a JK flip-flop, and a minimal AND-OR-NOT network (if the network is needed). Draw the schematic diagram.

The carry-in signal is from the flip-flop and can be considered as $Q(i)$, and the carry-out is therefore $Q(i + 1)$. The truth table can be written as follows:

$Q(i)$	ai	bi	si	$Q(i+1)$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

The excitation table for a JK flip-flop is:

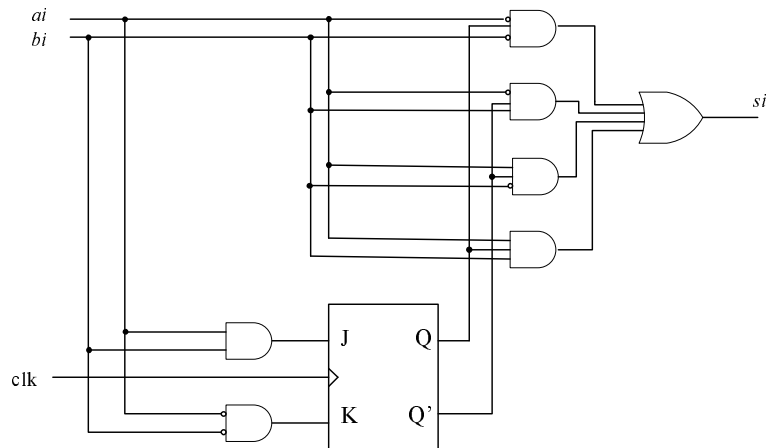
$Q(i)$	$Q(i+1)$	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

The truth table from $Q(i)$, ai , bi to J , K is:

$Q(i)$	ai	bi	J	K
0	0	0	0	x
0	0	1	0	x
0	1	0	0	x
0	1	1	1	x
1	0	0	x	1
1	0	1	x	0
1	1	0	x	0
1	1	1	x	0

Therefore, $J = ai \cdot b_i$, $K = ai' \cdot bi'$,

$$\begin{aligned}
 si &= Q(i) \oplus ai \oplus bi = Q(i) \oplus (ai'bi + aibi') = Q(i)'(ai'bi + aibi') + Q(i)(ai'bi' + aibi) \\
 &= Q(i)'ai'bi + Q(i)'aibi' + Q(i)ai'bi' + Q(i)aibi
 \end{aligned}$$



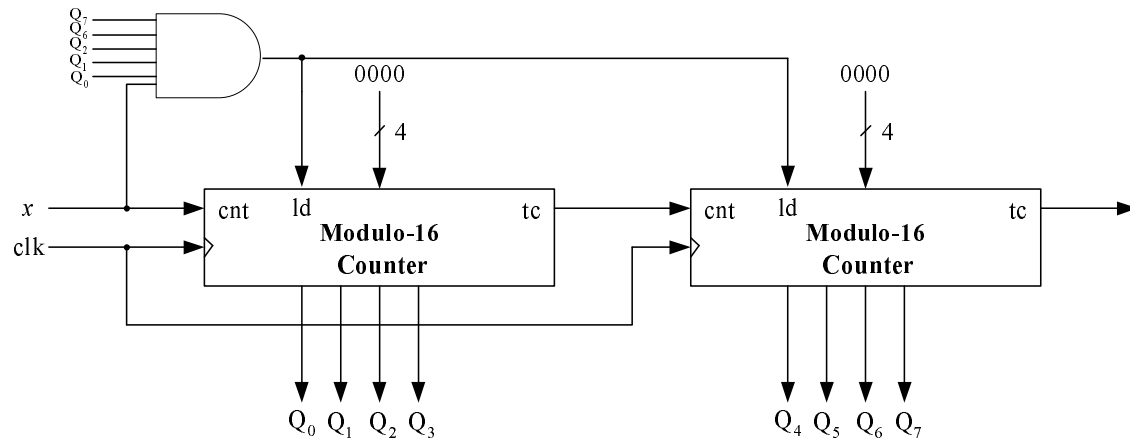
IV. Given modulo-16 counters, draw the logic diagram to show the following designs.

IV(1). Design a modulo-200 counter with a repeated output.

We know that $2^7 < 200 < 2^8$, so we need 8 bits to represent 0 – 199. A modulo-16 counter have 4 bits output, so two such counters are enough, with one representing the least significant 4 bits, and the other representing the most significant 4 bits.

The terminal count (tc) signal of the first counter will be sent to the enable port (cnt) of the second counter; consequently, the second counter will count once when the first counter reaches 1111. Thus the carry could be perform correctly in this way.

Since we want to count the numbers from 0 – 199, the load (ld) signal should be sent to both counters to load 0000 when the number counts to 199, which is 1100 0111. So we should use an AND gate which takes the signal of Q_7, Q_6, Q_2, Q_1, Q_0 together with the input signal x to generate the load signal for both counters.



IV(2). Design a counter with a repeated output sequence 15, 0, 1, 2, 8, 9, 10, 6, 7, with a modulo-16 counter and a minimal combinational network. Write the Boolean expression and draw the schematic diagram.

From the given the sequence, we can determine the load bit L and the values I_3, I_2, I_1, I_0 that will be loaded to the counter, according to the current output Q_3, Q_2, Q_1, Q_0 . The truth table is as follows:

ID	Q_3	Q_2	Q_1	Q_0	L	I_3	I_2	I_1	I_0
0	0	0	0	0	0	x	x	x	x
1	0	0	0	1	0	x	x	x	x
2	0	0	1	0	1	1	0	0	0
3	0	0	1	1	x	x	x	x	x
4	0	1	0	0	x	x	x	x	x
5	0	1	0	1	x	x	x	x	x
6	0	1	1	0	0	x	x	x	x
7	0	1	1	1	1	1	1	1	1
8	1	0	0	0	0	x	x	x	x
9	1	0	0	1	0	x	x	x	x
10	1	0	1	0	1	0	1	1	0
11	1	0	1	1	x	x	x	x	x
12	1	1	0	0	x	x	x	x	x
13	1	1	0	1	x	x	x	x	x
14	1	1	1	0	x	x	x	x	x
15	1	1	1	1	0	x	x	x	x

Hence, $L = Q_2'Q_1 + Q_3'Q_1Q_0$, $I_3 = Q_3'$, $I_2 = I_1 = Q_3 + Q_2$, $I_0 = Q_2$. The K-maps are shown as follows.

		Q_1Q_0			
		00	01	11	10
Q_3Q_2	00	0	0	x	1
	01	x	x	1	0
	11	x	x	0	x
	10	0	0	x	1

L

		Q_1Q_0			
		00	01	11	10
Q_3Q_2	00	x	x	x	1
	01	x	x	1	x
	11	x	x	x	x
	10	x	x	x	0

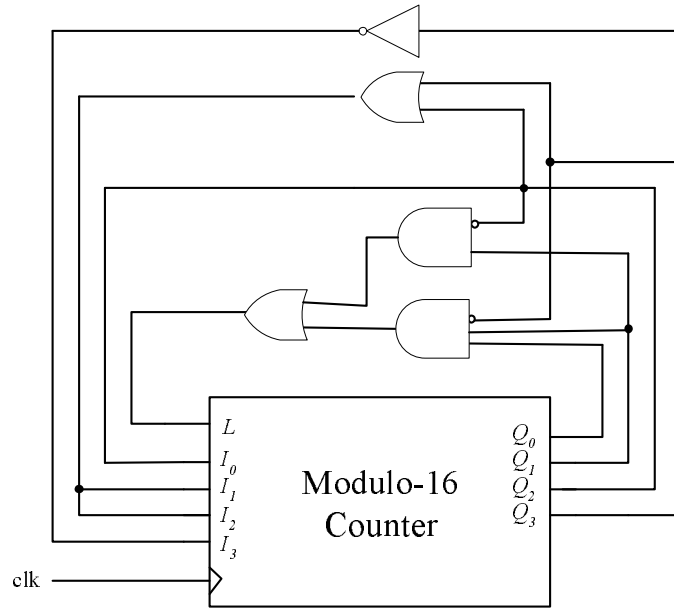
I_3

		Q_1Q_0			
		00	01	11	10
Q_3Q_2	00	x	x	x	0
	01	x	x	1	x
	11	x	x	x	x
	10	x	x	x	1

I_2, I_1

		Q_1Q_0			
		00	01	11	10
Q_3Q_2	00	x	x	x	0
	01	x	x	1	x
	11	x	x	x	x
	10	x	x	x	0

I_0

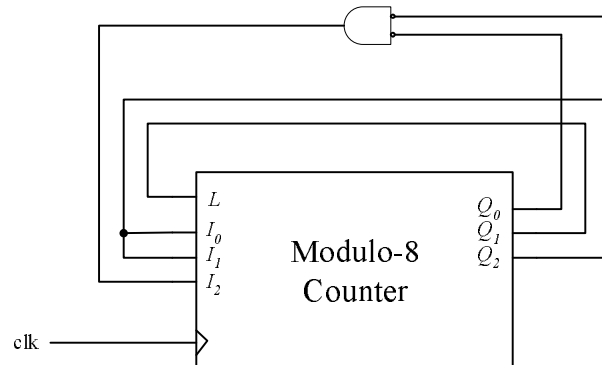


V. Design a counter with a repeated output sequence 0, 1, 2, 4, 5, 6, 3, with a modulo-8 counter and a minimal AND-OR-NOT network. Write the Boolean expression and draw the schematic diagram.

From the given the sequence, we can determine the load bit L and the values I_2, I_1, I_0 that will be loaded to the counter, according to the current output Q_2, Q_1, Q_0 . The truth table is as follows:

ID	Q_2	Q_1	Q_0	L	I_2	I_1	I_0
0	0	0	0	0	x	x	x
1	0	0	1	0	x	x	x
2	0	1	0	1	1	0	0
3	0	1	1	1	0	0	0
4	1	0	0	0	x	x	x
5	1	0	1	0	x	x	x
6	1	1	0	1	0	1	1
7	1	1	1	x	x	x	x

Hence $L = Q_1$, $I_2 = Q_2'Q_0'$, $I_1 = Q_2$, $I_0 = Q_2$ (from K-Maps).



VI. System Designs: Implement the following algorithm:

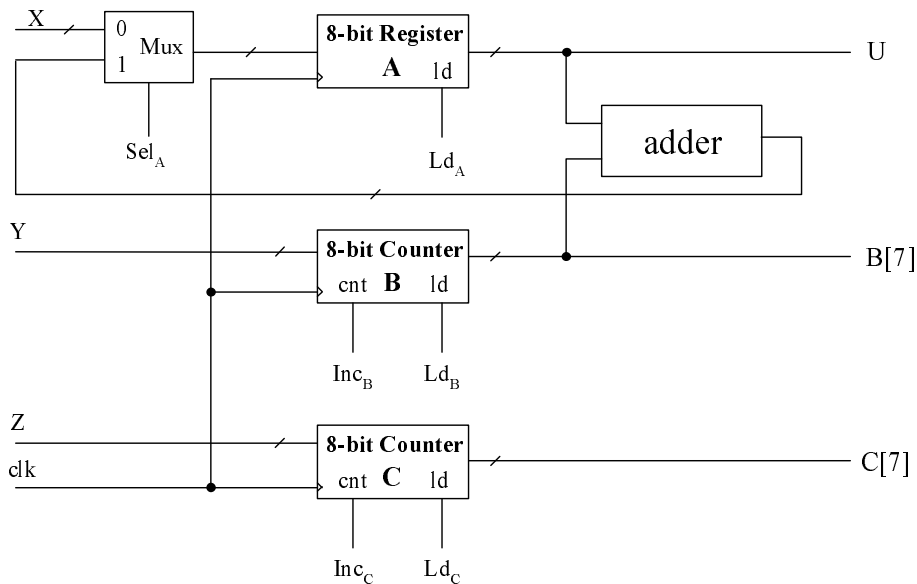
```

Alg(X,Y,Z,start,U,done);
Input X[7:0], Y[7:0], Z[7:0], start;
Output U[7:0], done;
Local-object A[7:0], B[7:0], C[7:0];
S1: If start' goto S1;
S2: done <= 0 || A <= X || B <= Y || C <= Z;
S3: A <= Add(A,B);
S4: If B'[7] goto S3 || B <= Inc(B);
S5: If C'[7] goto S3 || C <= Inc(C);
S6: U <= A || done <= 1 || goto S1;
End Alg

```

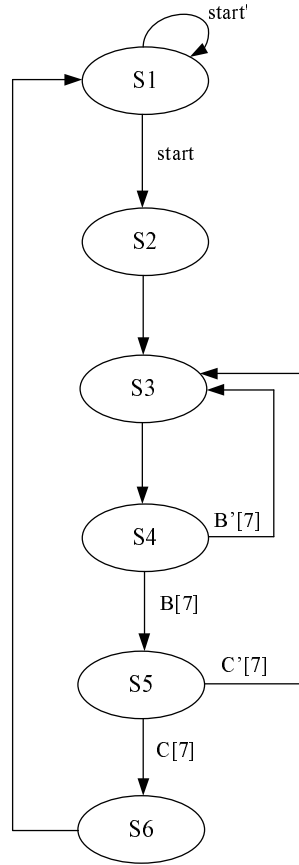
VI(1). Design a data subsystem that is adequate to execute the algorithm. Draw the schematic diagram to show the design.

First we identify the essential elements in the data subsystem: one 8-bit register to store A; an adder to perform the addition $\text{Add}(A,B)$; two 8-bit counters to perform the self-increment (Inc operation) for B and C. In addition, since the input of the 8-bit register could be either from the input X or the adder results, we need a 2-to-1 multiplexer to select the input. Thus, the inputs of the data subsystem include three signals Ld_A , Ld_B and Ld_C that are used to load the inputs to the register and two counters, a selection signal Sel_A to select the input of the register, two operational signals Inc_B and Inc_C to indicate whether we need to increase the value of B or C by 1, and the data inputs X, Y, Z. The outputs of the data subsystem are U, B[7], and C[7] (U is the required output and B[7] and C[7] will be used in the control subsystem). The schematic diagram of the data subsystem is shown as follows:



VI(2). Design the control subsystem

(i) draw the state diagram;



(ii) draw the logic diagram that implements the control subsystem with a one hot encoding design

The state table is as follows:

	$Q_6Q_5Q_4Q_3Q_2Q_1$	start	B[7]	C[7]	$N_6N_5N_4N_3N_2N_1$	done	sel_A	Ld_A	Inc_B	Ld_B	Inc_C	Ld_C
S1	0 0 0 0 0 1	0	x	x	0 0 0 0 0 1	1	x	0	0	0	0	0
S1	0 0 0 0 0 1	1	x	x	0 0 0 0 1 0	1	x	0	0	0	0	0
S2	0 0 0 0 1 0	x	x	x	0 0 0 1 0 0	0	0	1	0	1	0	1
S3	0 0 0 1 0 0	x	x	x	0 0 1 0 0 0	0	1	1	0	0	0	0
S4	0 0 1 0 0 0	x	0	x	0 0 0 1 0 0	0	x	0	1	0	0	0
S4	0 0 1 0 0 0	x	1	x	0 1 0 0 0 0	0	x	0	1	0	0	0
S5	0 1 0 0 0 0	x	x	0	0 0 0 1 0 0	0	x	0	0	0	1	0
S5	0 1 0 0 0 0	x	x	1	1 0 0 0 0 0	0	x	0	0	0	1	0
S6	1 0 0 0 0 0	x	x	x	0 0 0 0 0 1	1	x	0	0	0	0	0

$$\begin{aligned}
 done &= Q_1 + Q_6 & N_1 &= Q_6 + Q_1 \cdot start' \\
 sel_A &= Q_2' \text{ (or } Q_3) & N_2 &= Q_1 \cdot start \\
 Ld_A &= Q_2 + Q_3 & N_3 &= Q_2 + Q_4B'[7] + Q_5C'[7] \\
 Inc_B &= Q_4 & N_4 &= Q_3 \\
 Ld_B &= Q_2 & N_5 &= Q_4B[7] \\
 Inc_C &= Q_5 & N_6 &= Q_5C[7] \\
 Ld_C &= Q_2
 \end{aligned}$$

The logic diagram is shown as follows:

