

This is a take-home exam. You are allowed to check books, notes, and conduct an internet search. However, you are expected to work on the solution by yourself. No discussion with anyone else is allowed.

There are two parts to the exam. Part A weighs 90 points. Part B weighs 10 points. However, in Part B, the total weight of the problems is higher than 10 points. You are encouraged to earn up to 10 points. In other words, there is no penalty on wrong solutions in Part B, but the maximum points to earn are limited to 10 points.

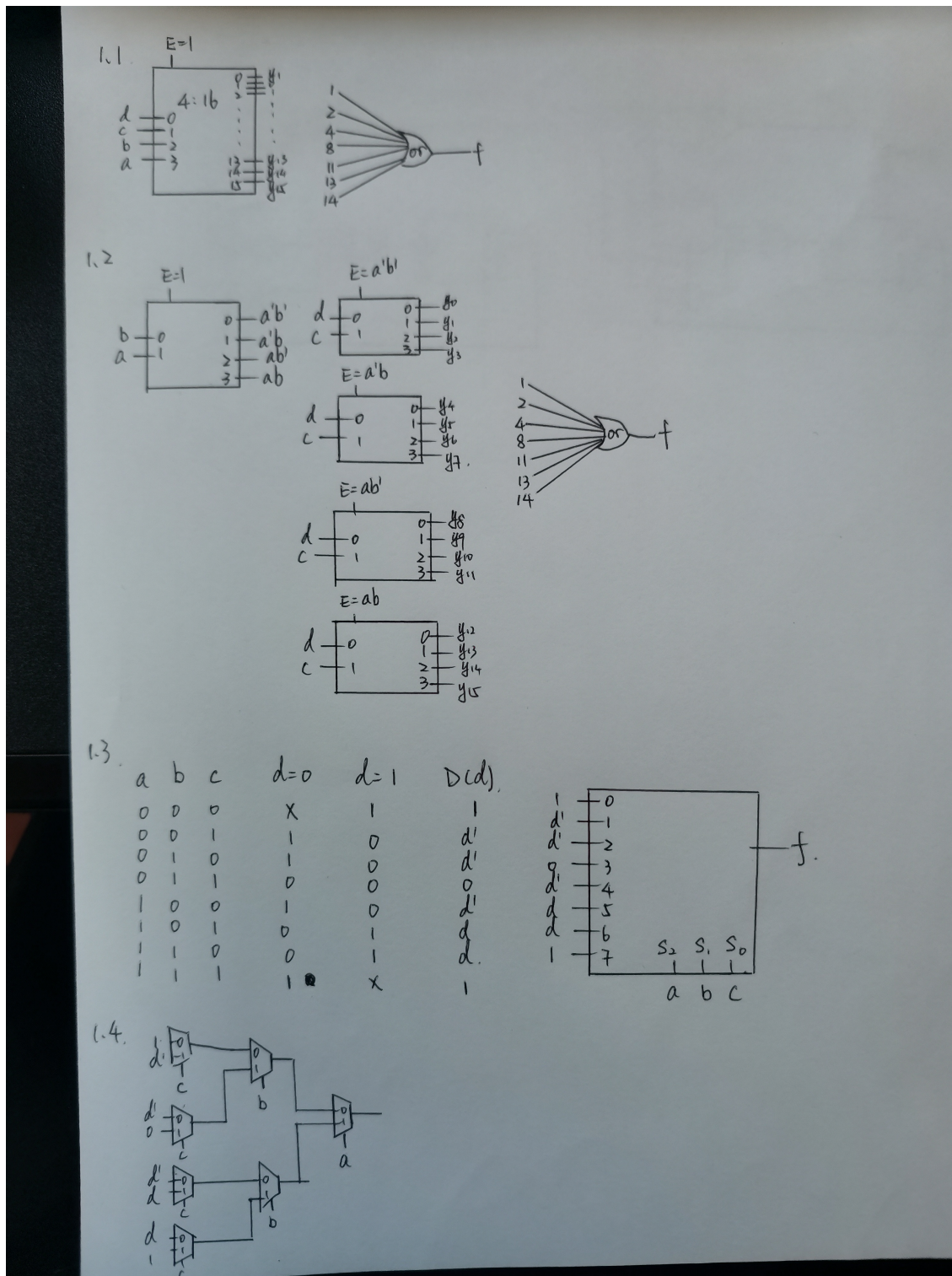
Please read the following instructions carefully. If you are unclear about any of the questions on the exam, make the most plausible assumption to answer the question. Instructors and proctors will not answer questions on the exam material. **All the Best!**

Part A: (90 pts)

A.1. (Decoders and Multiplexers) Given a four-input switching function

$$f(a, b, c, d) = \sum m(1, 2, 4, 8, 11, 13, 14) + \sum d(0, 15),$$

- 1.1. Implement the function using a minimal network of 4:16 decoders and OR gates.
- 1.2. Implement the function using a minimal network of 2:4 decoders and OR gates.
- 1.3. Implement the function using a minimal network of 8:1 multiplexers.
- 1.4. Implement the function using a minimal network of 2:1 multiplexers.



A.2. (Sequential Arithmetic Operations) Design a sequential operator that performs arithmetic operation $A + 2 \times B$ on two n -bit binary numbers (A, B) using D flip-flops and a minimal network of AND, OR, NOT gates. Assume that NOT gates are free. Show your design process and the logic diagram of the circuit.

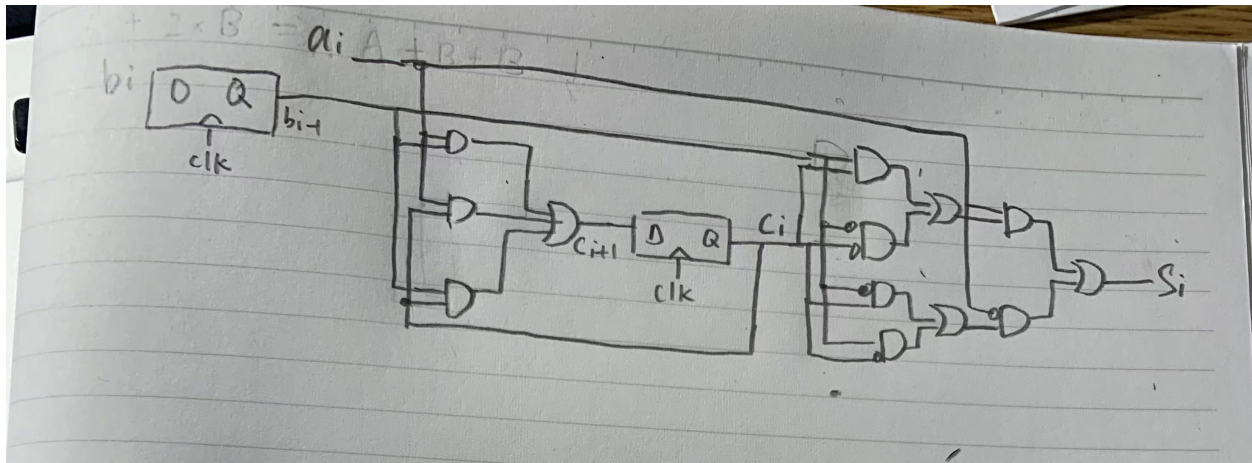


Figure 1: Example Diagram

Here $2 \cdot B$ is equivalent to the left shift one bit for B with $b(-1) = 0$.

A.3. (Counter) Use standard modulo counters to design various counters.

3.1 Given a modulo 16 counter, construct a 3 to 11 counter.

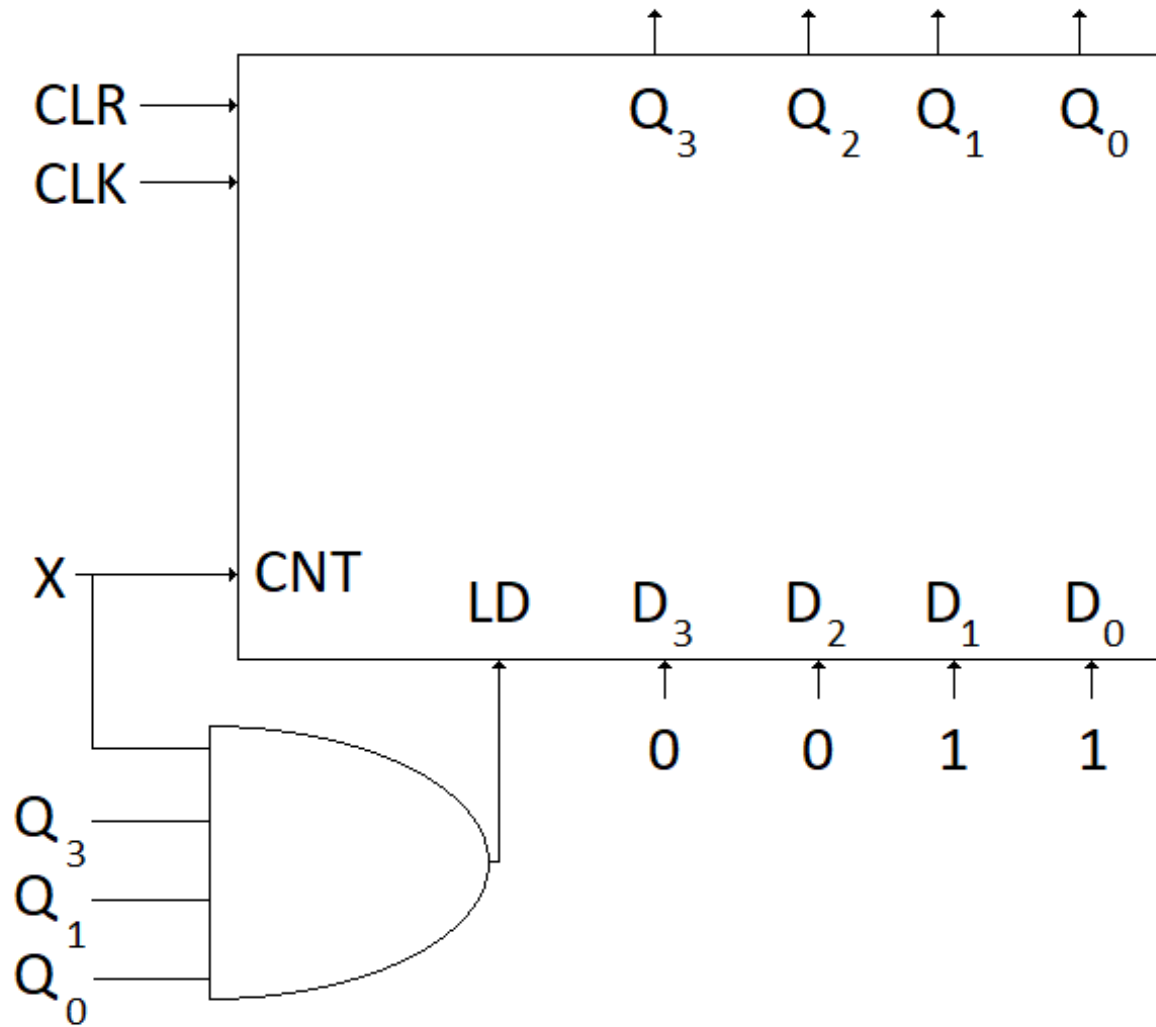


Figure 2: 3 to 11 counter

3.2 Given a modulo 8 counter, construct a counter that counts in a sequence 1,5,6,7,2,3 with a minimal network of AND, OR, NOT gates. Assume that NOT gates are free.

id	$Q_2Q_1Q_0$	LD	D_2	D_1	D_0
0	000	x	x	x	x
1	001	1	1	0	1
2	010	0	x	x	x
3	011	1	0	0	1
4	100	x	x	x	x
5	101	0	x	x	x
6	110	0	x	x	x
7	111	1	0	1	0

Table 1: State table

$Q_2Q_1 \setminus Q_0$	0	1
00	x	1
01	0	1
11	0	1
10	x	0

Table 2: $LD = Q_1Q_0 + Q'_2Q_0 = Q_0(Q_1 + Q'_2)$

$Q_2Q_1 \setminus Q_0$	0	1
00	x	1
01	x	0
11	x	0
10	x	x

Table 3: $D_2 = Q'_1$

$Q_2Q_1 \setminus Q_0$	0	1
00	x	0
01	x	0
11	x	1
10	x	x

Table 4: $D_1 = Q_2$

$Q_2Q_1 \setminus Q_0$	0	1
00	x	1
01	x	1
11	x	0
10	x	x

Table 5: $D_0 = Q'_2$

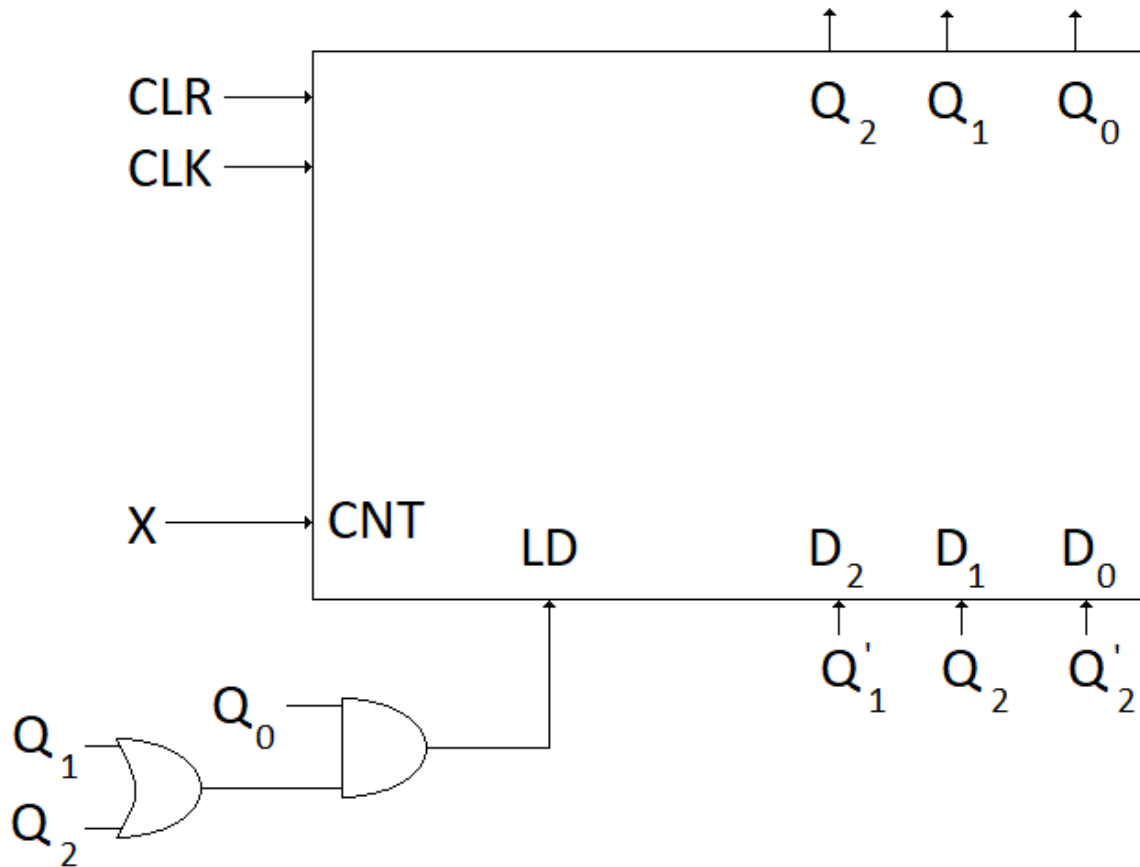


Figure 3: Counter that counts in sequence 1, 5, 6, 7, 2, 3

3.3 Construct a modulo 128 counter using modulo 4 counters.

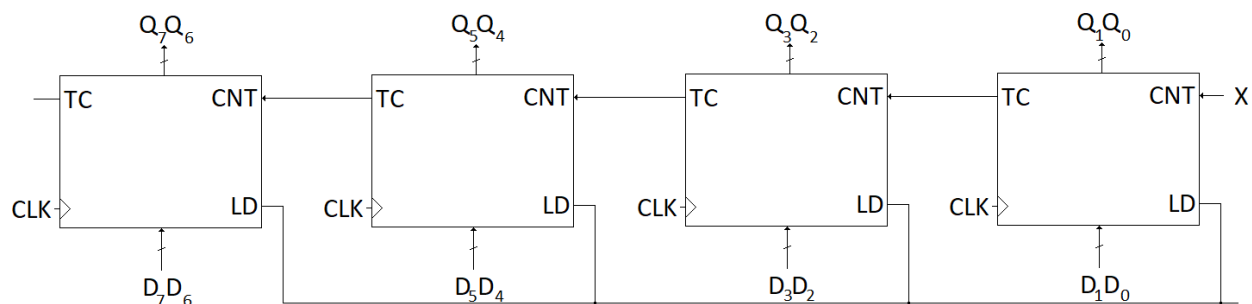


Figure 4: Modulo 128 counter

A.4. 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];

```

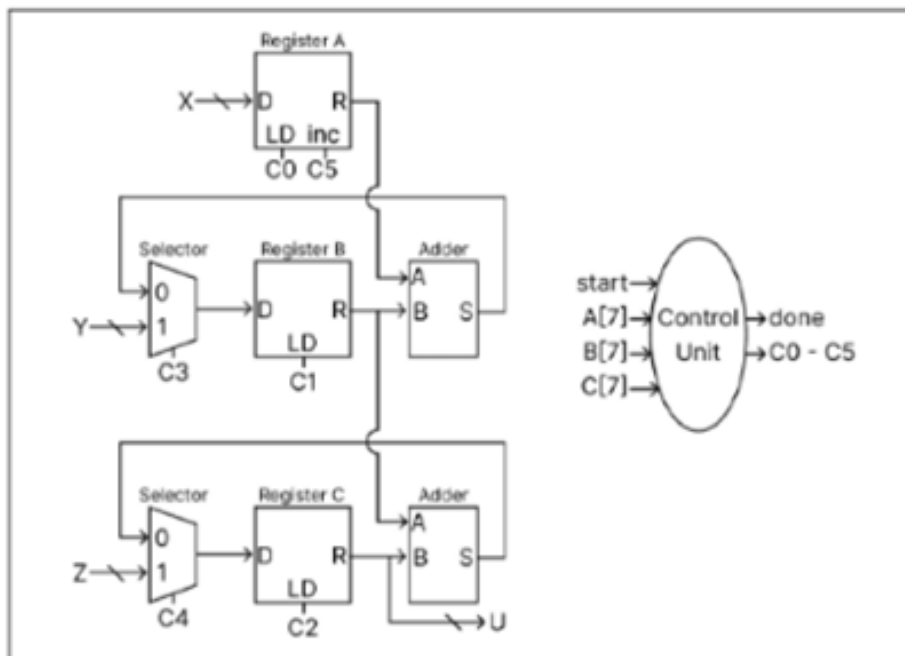
```
S1: If start' goto S1 || done  $\leftarrow$  1;
S2: done  $\leftarrow$  0 || A  $\leftarrow$  X || B  $\leftarrow$  Y || C  $\leftarrow$  Z;
S3: B  $\leftarrow$  Add(A,B) || C  $\leftarrow$  Add(B,C);
S4: If A[7] goto S6 || A  $\leftarrow$  INC(A);
S5: If C[7] goto S4 || C  $\leftarrow$  Add(B,C);
S6: If B[7] goto S5 || B  $\leftarrow$  Add(A,B);
S7: U  $\leftarrow$  C || goto S1;
End Alg
```

- 4.1. Design a data subsystem that is adequate to execute the algorithm by answering the following questions.
- 4.1.1. Write the table that lists the instructions and the corresponding components that should be used in the data path subsystem.
- 4.1.2. Draw the schematic diagram to show the data path subsystem. Label the inputs, outputs, and control signals of all components.
- 4.2. Design the control subsystem by answering the following questions.
- 4.2.1. Write the table that lists the value of control signals for every state.
- 4.2.2. Draw the state diagram.
- 4.2.3. Implement the control subsystem using a one-hot encoding design. Draw the logic diagram.

4.1.1 Answer:

	State	Instruction
S1: If start' goto S1 done = 1;		
S2: done = 0 A ← X B ← Y C ← Z;	S2	A ← Load(X)
S3: B ← Add(A,B) C ← Add(B,C);	S2	B ← Load(Y)
S4: If A[7] goto S6 A ← INC(A);	S2	C ← Load(Z)
S5: If C[7] goto S4 C ← Add(B,C);	S3, S6	B ← Add(A,B)
S6: If B[7] goto S5 B ← Add(A,B);	S3, S5	C ← Add(B,C)
S7: U ← C goto S1;	S4	A ← INC(A)
	S7	U ← C, wires

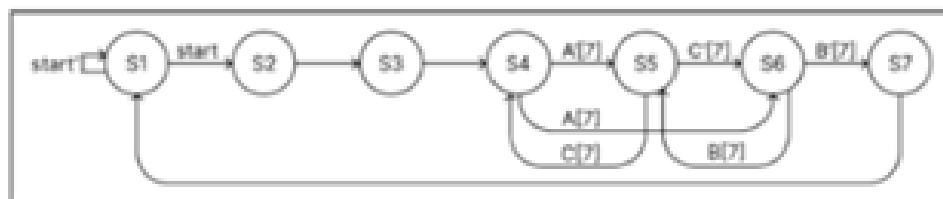
4.1.2 Answer:



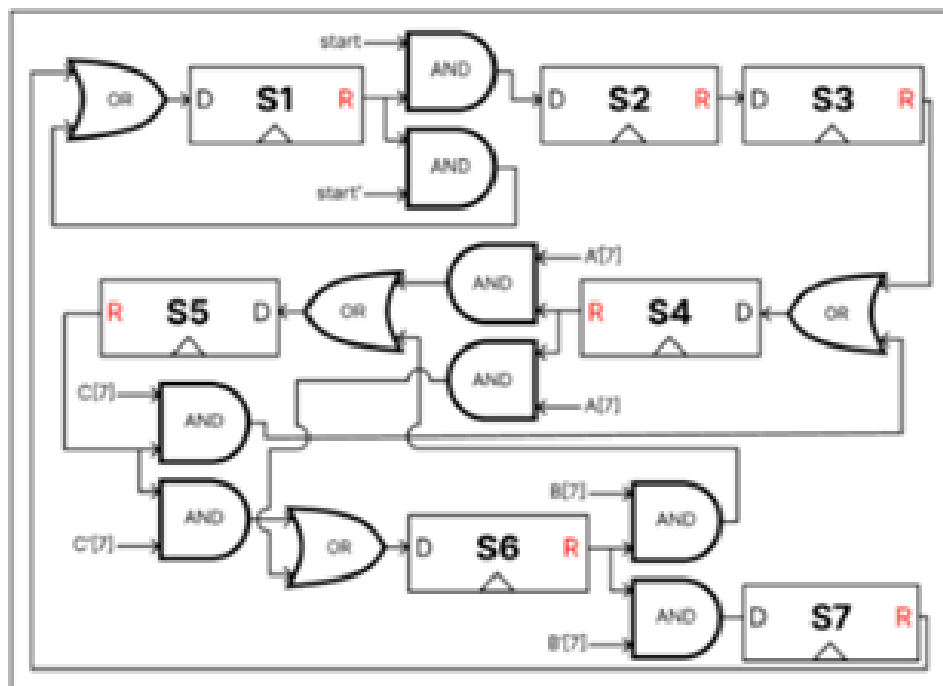
4.2.1 Answer:

State	C0	C1	C2	C3	C4	C5	done
S1	0	0	0	-	-	-	1
S2	1	1	1	0	0	0	0
S3	0	0	0	1	1	-	0
S4	1	0	0	-	-	1	0
S5	0	0	1	-	1	-	0
S6	0	1	0	1	-	-	0
S7	0	0	0	-	-	-	0

4.2.2 Answer:



4.2.3 Answer:

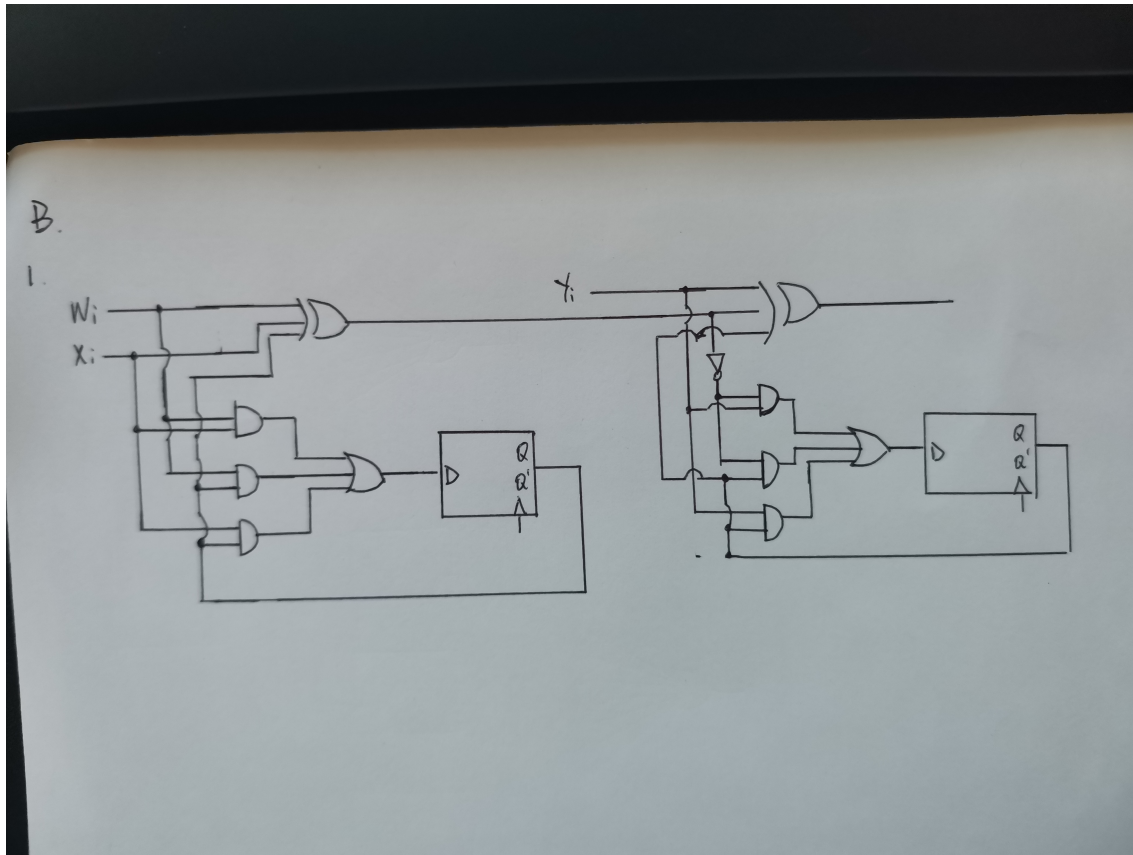


Part B: (10 pts)

In Part B, the total weight of the problems is higher than 10 points. You are encouraged to earn up to 10 points. In other words, there is no penalty on wrong solutions, but the maximum points to earn are limited to 10 points. Note that some problems are open, i.e. exact (perfect) solutions may be challenging to derive. Nonetheless, the principles taught in the class can be used to start the exploration.

B.1. Sequential Arithmetic Operation: Design a sequential arithmetic operator that performs arithmetic op-

eration $W + X - Y$ on three n -bit binary numbers, W , X , and Y , in n clock cycles using D flip-flops and a minimal network of AND, OR, NOT gates. Assume that NOT gates are free. Show your design process and the logic diagram of the circuit. (5pts)



B.2. Multistate Networks: With reference to Benes and Omega networks shown in slide 21 of Lecture 16 (System Design I), we label the left side ports from top to bottom (0, 1, 2, 3, 4, 5, 6, 7) and the right side ports from top to bottom (5, 6, 7, 0, 1, 2, 3, 4). We connect the ports of the same labels between the right and left ports. (5pts)

1. Draw the routes using the Benes network (top figure in slide 21).

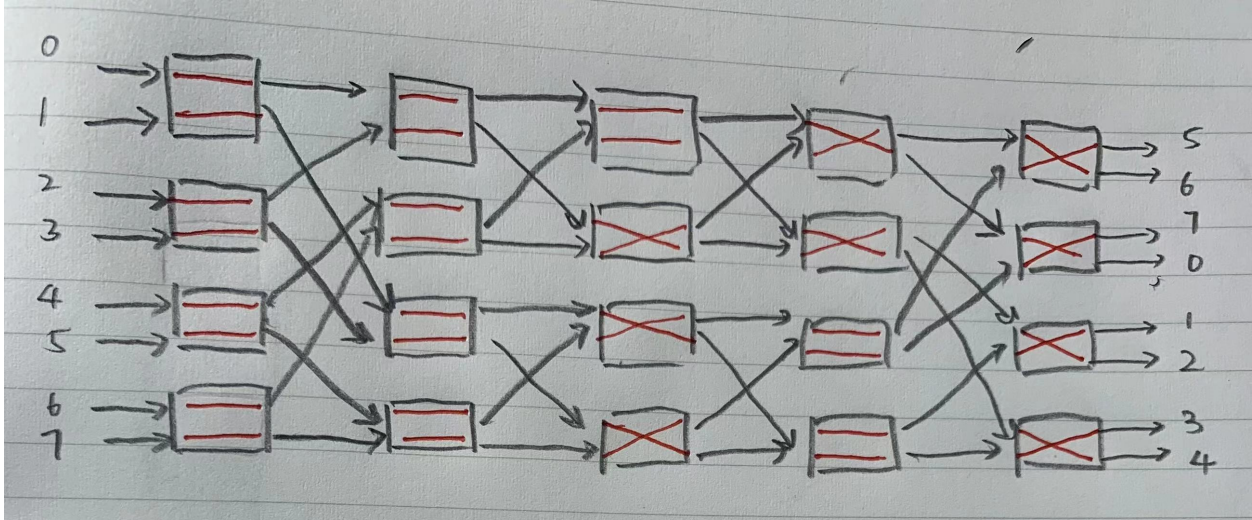


Figure 5: Benes network

2. Draw the routes using the Omega network (bottom figure in slide 21).

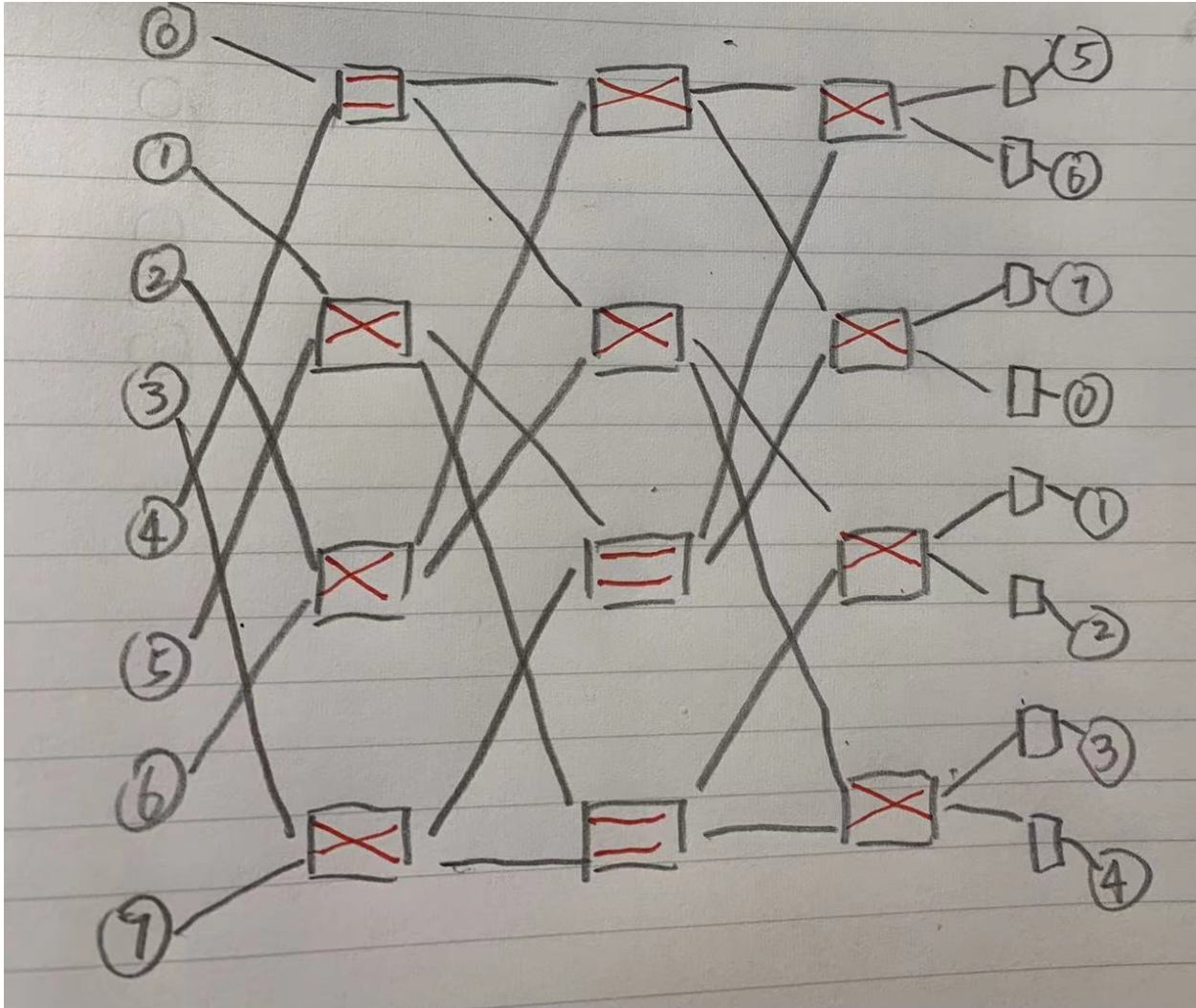


Figure 6: Omega network

B.3. System Designs: Description and conversion of handshake protocols. (5pts)

1. Define the 4-phase handshake protocol. Show that the algorithm stated in Problem A.4 follows the 4-phase handshake protocol.

In 4-phase handshake there should be a sender (can send "req" to receiver) and receiver (can send "ack" signal to sender). The handshake requires 4 steps, starting from an initial state where $req=0$ and $ack=0$:

1. Sender sets $req=1$
2. Receiver detects the change in req and sets $ack=1$
3. Sender sees $ack=1$ and sets $req=0$
4. Receiver sees $req=0$ and sets $ack=0$

At the end of the 4th step, we are back in the initial state and can repeat the process, which initiates a new handshake.

The system in A4 can be modeled as a receiver where done is the opposite of ack (done=0 is equivalent to ack=1 and vice versa) and start is equivalent to req. The input to the system is initially start=0 and output is initially done=1, which follows the 4-phase handshake protocol where req=0 and ack=0 in the initial state.

1. The system receives start=1 (req=1)
 2. The system sets done=0 (ack=1) and then executes the rest of the algorithm
 3. Start is set back to 0 (req=0)
 4. The system returns to S1 from S7, setting done=1 (ack=0)
2. Define the 2-phase handshake protocol.

In 2-phase handshake, the same steps from the 4-phase handshake are performed, but a handshake is completed at the end of both steps 2 and 4, when the ack signal matches the req signal. Thus the system may begin and end in either req=0 and ack=0 state or in req=1 and ack=1 state.

3. Rewrite the algorithm stated in Problem A.4 using 2-phase handshake protocol.

```

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], L;
S1: L  $\leftarrow$  0 || done  $\leftarrow$  1;
S2: If start == L goto S2;
S3: done  $\leftarrow$  done' || L  $\leftarrow$  L' || A  $\leftarrow$  X || B  $\leftarrow$  Y || C  $\leftarrow$  Z;
S4: B  $\leftarrow$  Add(A,B) || C  $\leftarrow$  Add(B,C);
S5: If A[7] goto S7 || A  $\leftarrow$  INC(A);
S6: If C[7] goto S5 || C  $\leftarrow$  Add(B,C);
S7: If B[7] goto S6 || B  $\leftarrow$  Add(A,B);
S8: U  $\leftarrow$  C || goto S2;
End Alg

```

B.4. System Designs: For the System Designs of Problem 5 in HW4, suppose that the resources of the arithmetic operators are limited: we have no more than one Add operator and no more than one Inc operator. Show your revised solution through subproblems 5.1, 5.2 and 5.3. (5 pts)

