CS 4341: Digital Logic and Computer Architecture

Homework 4 Guide

Question 3.24

Outline

The steps to solve this type of problem are as follows:

1. Create the state table
2. Determine from the number of states the number of memory elements you are going to use.
3. For each memory element and each output, create a k-map.
4. Build the resulting circuit.

State Table

The sequence of states is given in the question; it is an 8 number gray code. The state table can be created from this, see the table below.

Inputs	Current State			Next State		
RST	A	B	C	An	Bn	Cn
0	0	0	0	0	0	1
0	0	0	1	0	1	1
0	0	1	1	0	1	0
0	0	1	0	1	1	0
0	1	1	0	1	1	1
0	1	1	1	1	0	1
0	1	0	1	1	0	0
0	1	0	0	0	0	0
1	X	X	X	0	0	0

Note that in this table X means “Don’t care” even when it occurs on the “current state” side of the table. This convention isn’t universal but I hope that the meaning is intuitively clear.

Memory Design

The state(s) where the reset input is high can be handled using the reset inputs of the memory elements, so they don’t factor in to the number of states the circuit must be able to remember.

There are 8 such states that must be remembered, which can be expressed using 3 bits of memory since $2^3=8$. There are also 3 outputs from the circuit, the digits of the current gray code number. In this case it is very effective to have the circuit simply output its current state, but note that this won't always be possible.

I'm going to use D flip flops. Each memory element needs to be set to the proper value for the circuit's next state based on the inputs (in this case, none which will be considered) and the circuit's current state. This can be accomplished by constructing combinational circuits that use the output (current state) of the memory elements as input and produce as output the next state of each memory element. There are 3 memory elements so there will be 3 combinational blocks.

These blocks can (and should) be minimized using K-maps. Example K-maps can be seen below.

Minimization

An (Next state of A)

		B,C			
		00	**01**	**11**	**10**
A	**0**	0	0	0	1
	1	0	1	1	1

B !C + A C

Bn (Next state of B)

		B,C			
		00	**01**	**11**	**10**
A	**0**	0	1	1	1
	1	0	0	0	1

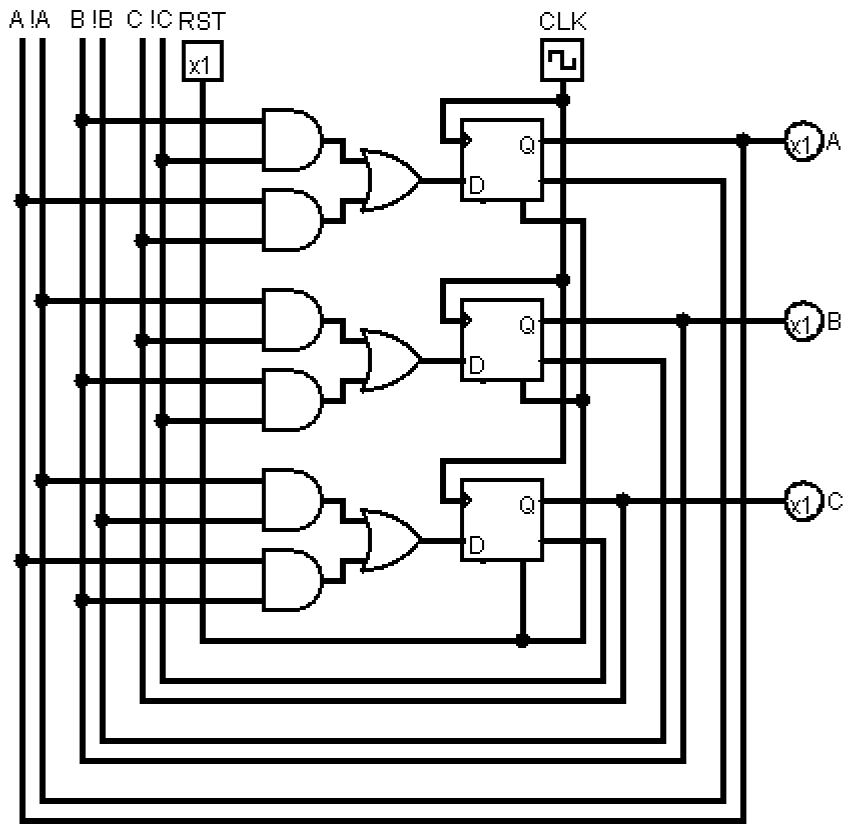
!A C + B !C

Cn (Next state of C)

		B,C			
		00	**01**	**11**	**10**
A	**0**	1	1	0	0
	1	0	0	1	1

!A !B + A B

Implementation



Note that the reset input of these flip-flops is the right side of the lower edge, and that reset is an active-high input for them. That is, the flip flops are pinned to 0 when their reset input is 1.

Make sure that you can identify the combinational blocks corresponding to the K-maps above.

Question 3.25

State Table

Inputs		Current State			Next State		
RST	Up/ Down	A	B	C	An	Bn	Cn
0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0
0	0	0	1	1	0	0	1
0	0	0	1	0	0	1	1
0	0	1	1	0	0	1	0
0	0	1	1	1	1	1	0
0	0	1	0	1	1	1	1
0	0	1	0	0	1	0	1
0	1	0	0	0	0	0	1
0	1	0	0	1	0	1	1
0	1	0	1	1	0	1	0
0	1	0	1	0	1	1	0
0	1	1	1	0	1	1	1
0	1	1	1	1	1	0	1
0	1	1	0	1	1	0	0
0	1	1	0	0	0	0	0
1	X	X	X	X	0	0	0

Memory Design

There is no need to create additional states, and therefore no need to change the number of memory elements. The new input is simply another factor in determining the next state.

Minimization

An (Next state of A)

		C,D			
		00	**01**	**11**	**10**
A,B	**00**	1	0	0	0
	01	1	1	1	0
	11	0	1	1	1
	10	0	0	0	1

!D !B !C + A C + D B !C

Bn (Next state of B)

		C,D			
		00	**01**	**11**	**10**
A,B	**00**	0	0	0	1
	01	0	1	1	1
	11	0	0	0	1
	10	0	1	1	1

B !C + !D A C + D !A C

Cn (Next state of C)

		C,D			
		00	**01**	**11**	**10**
A,B	**00**	0	0	1	1
	01	1	1	0	0
	11	0	0	1	1
	10	1	1	0	0

!D !A B + !D A !B + D !A !B + D A B

Implementation