Lecture 10:

Traffic Lights

A Design Example

Traffic Light Circuit Design

In this lecture we will work through a design example from problem statement to digital circuit. The design will start with a verbal description of the problem, after which:

The problem will be expressed in terms of a synchronous finite state machine.

The number of required states and the number of flip-flops will be determined.

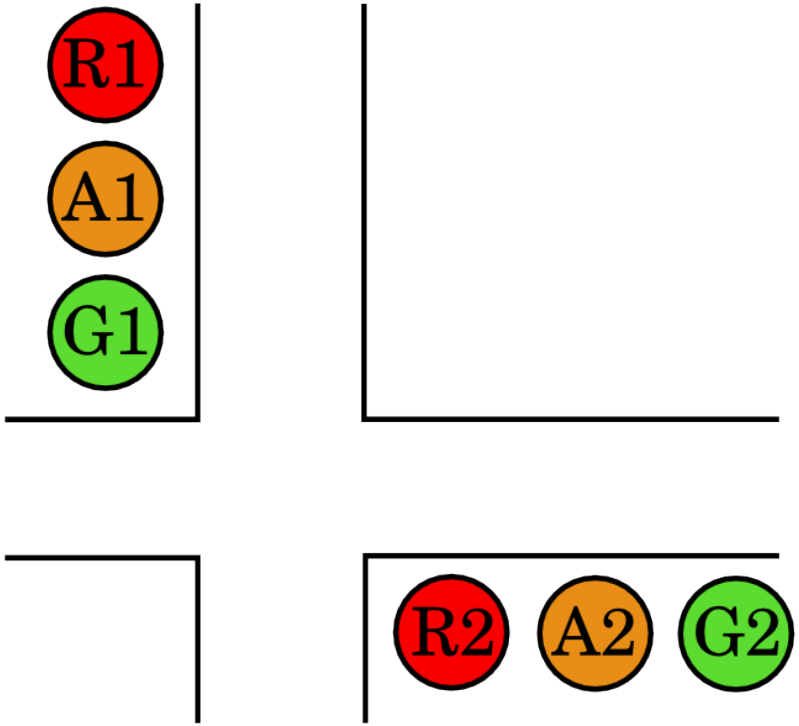
The state transition table (with states) will be constructed.

Flip-flop outputs assigned to states and K-maps drawn.

Circuits will be minimised and the system tested.

The Problem

The traffic department is trying out a new system of traffic lights. We have to design a synchronous digital system which operates this new type of traffic light at a road crossing.



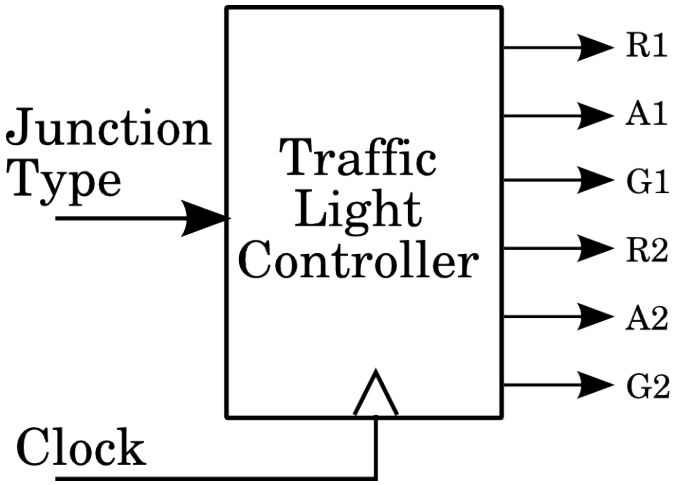
Quiet Junction

Red	Green
Red	Amber
Green	Red
Amber	Red

Busy Junction

Red	Green
Red	Amber
Red	Red
Green	Red
Amber	Red
Red	Red

The Problem (continued)



There are six lights controlled by 6 outputs:

R1, A1, G1 for the North/South direction

R2, A2, G2 for the East/West direction.

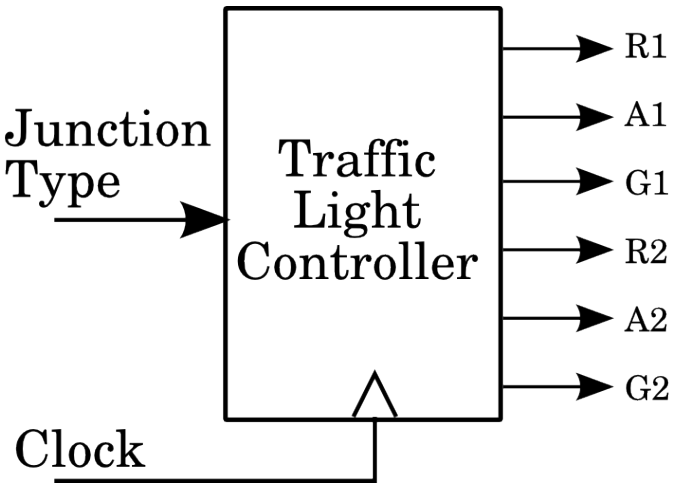
When the digital outputs are in the Logic-1 state they turn their respective lights on.

There is one input which is set to Logic-1 for a quiet junction and Logic-0 for a busy junction

Formalise the problem

Determine how many states are required?

What outputs are required for each state?



Junction=0						Junction=1					
R1	A1	G1	R2	A2	G2	R1	A1	G1	R2	A2	G2
1	0	0	0	0	1	1	0	0	0	0	1
1	0	0	0	1	0	1	0	0	0	1	0
1	0	0	1	0	0						
0	0	1	1	0	0	0	0	1	1	0	0
0	1	0	1	0	0	0	1	0	1	0	0
1	0	0	1	0	0						

Six states are required

Construct the finite state machine

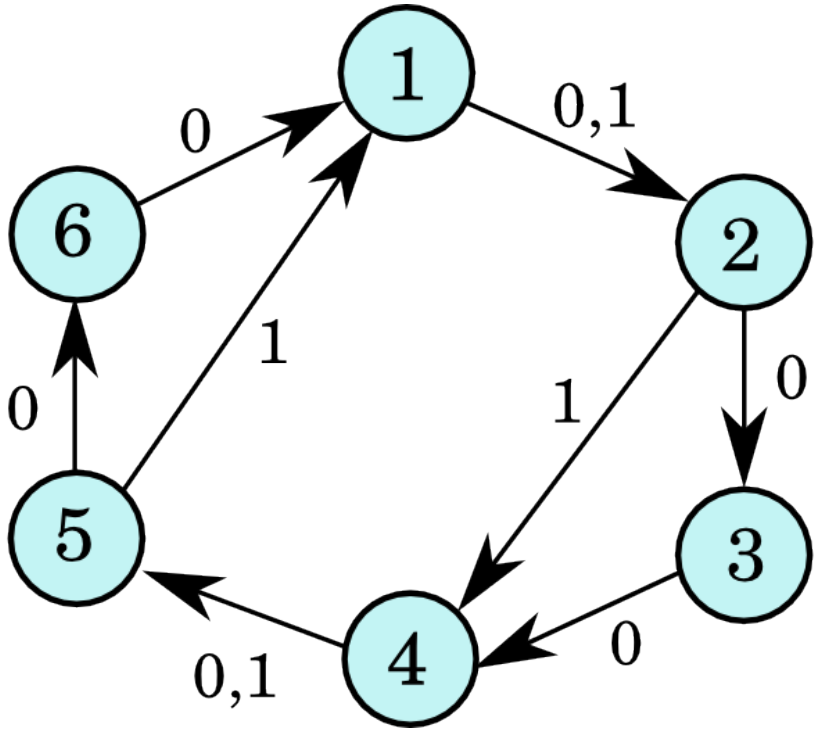
Junction=0						State	Junction=1					
R1	A1	G1	R2	A2	G2		R1	A1	G1	R2	A2	G2
1	0	0	0	0	1	1	1	0	0	0	0	1
1	0	0	0	1	0	2	1	0	0	0	1	0
1	0	0	1	0	0	3						
0	0	1	1	0	0	4	0	0	1	1	0	0
0	1	0	1	0	0	5	0	1	0	1	0	0
1	0	0	1	0	0	6						

Problem:

Two states (3 and 6) have exactly the same outputs, can they be merged as one state?

Construct the finite state machine

Junction=0						State	Junction=1					
R1	A1	G1	R2	A2	G2		R1	A1	G1	R2	A2	G2
1	0	0	0	0	1	1	1	0	0	0	0	1
1	0	0	0	1	0	2	1	0	0	0	1	0
1	0	0	1	0	0	3						
0	0	1	1	0	0	4	0	0	1	1	0	0
0	1	0	1	0	0	5	0	1	0	1	0	0
1	0	0	1	0	0	6						



Problem:

Two states (3 and 6) have exactly the same outputs, can they be merged as one state?

NO!!!

Select the type and number of flip-flops for the circuit.

Since the number of states is equal to six, the minimum number of flip-flops is three.

It would be possible to use is six flip-flops (one flip-flop per state). This might simplify the design of the circuit but it would be expensive in hardware.

For this design example (as normal) we will use three D-type flip-flops. There will be two unused states.

Assign flip-flop outputs to states and construct the transition table.

There are some heuristic rules for assigning states to flip-flop outputs, they are difficult to apply and do not guarantee a minimum circuit (nothing really does).

Assign flip-flops to states.

J	State (now)	State (next)
0	(1)	(2)
0	(2)	(3)
0	(3)	(4)
0	(4)	(5)
0	(5)	(6)
0	(6)	(1)
0	(7)	X
0	(8)	X
1	(1)	(2)
1	(2)	(4)
1	(3)	(X)
1	(4)	(5)
1	(5)	(1)
1	(6)	(X)
1	(7)	X
1	(8)	X

f-flops	State
010	(1)
011	(2)
100	(3)
101	(4)
110	(5)
111	(6)
000	(7)
001	(8)

where input J = 1 denotes a quiet junction

J	f-flops (now)	f-flops (next)
0	010	011
0	011	100
0	100	101
0	101	110
0	110	111
0	111	010
0	000	XXX
0	001	XXX
1	010	011
1	011	101
1	100	XXX
1	101	110
1	110	010
1	111	XXX
1	000	XXX
1	001	XXX

Find the minimum expressions

J	Q1	Q2	Q3	D1	D2	D3
0	0	0	0	X	X	X
0	0	0	1	X	X	X
0	0	1	1	1	0	0
0	0	1	0	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	1	0	1	0
0	1	1	0	1	1	1
1	1	0	0	X	X	X
1	1	0	1	1	1	0
1	1	1	1	X	X	X
1	1	1	0	0	1	0
1	0	0	0	X	X	X
1	0	0	1	X	X	X
1	0	1	1	1	0	1
1	0	1	0	0	1	1

D1

J Q1 \ Q2 Q3	00	01	11	10
00	X	X	1	0
01	1	1	0	1
11	X	1	X	0
10	X	X	1	0

D2

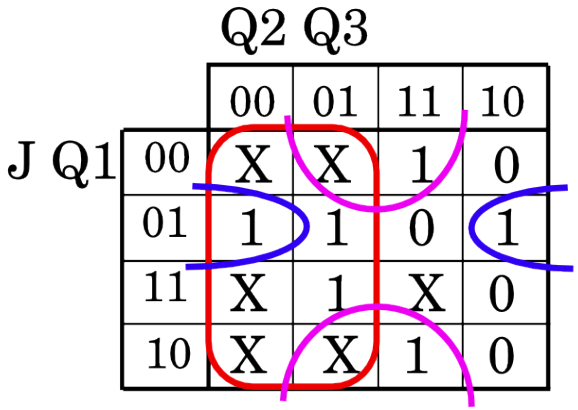
J Q1 \ Q2 Q3	00	01	11	10
00	X	X	0	1
01	0	1	1	1
11	X	1	X	1
10	X	X	0	1

D3

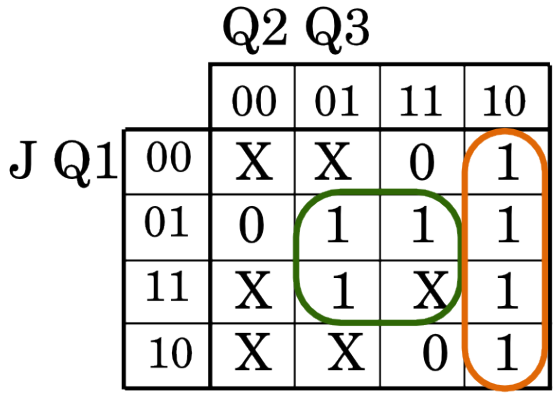
J Q1 \ Q2 Q3	00	01	11	10
00	X	X	0	1
01	1	0	0	1
11	X	0	X	0
10	X	X	1	1

Find the minimum expressions

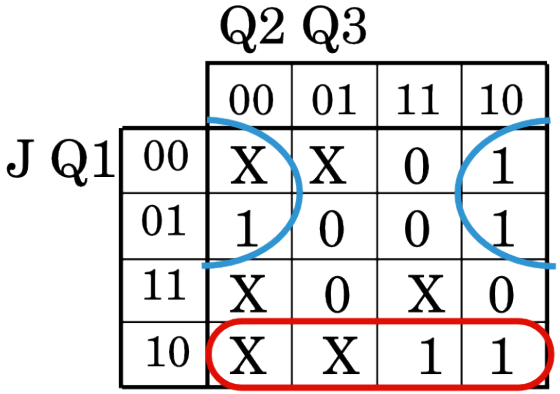
J	Q1	Q2	Q3	D1	D2	D3
0	0	0	0	X	X	X
0	0	0	1	X	X	X
0	0	1	1	1	0	0
0	0	1	0	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	1	0	1	0
0	1	1	0	1	1	1
1	1	0	0	X	X	X
1	1	0	1	1	1	0
1	1	1	1	X	X	X
1	1	1	0	0	1	0
1	0	0	0	X	X	X
1	0	0	1	X	X	X
1	0	1	1	1	0	1
1	0	1	0	0	1	1



D1 = Q2' + Q3Q1' + J'Q1Q3'



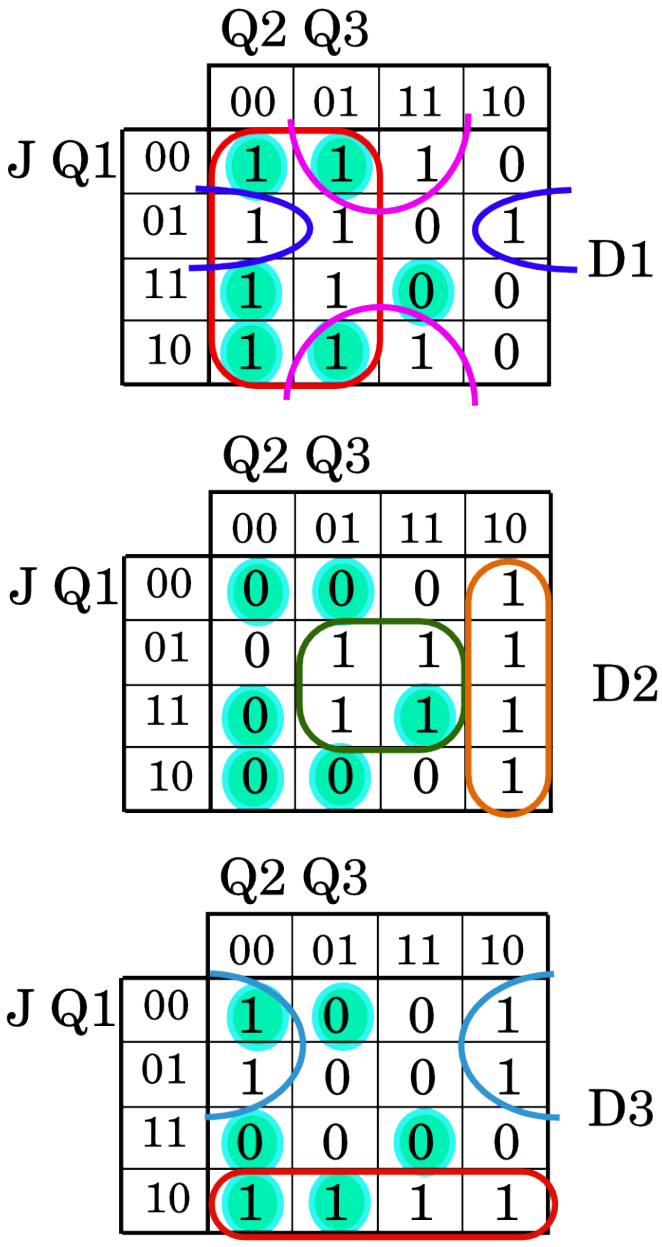
D2 = Q1Q3 + Q2Q3'



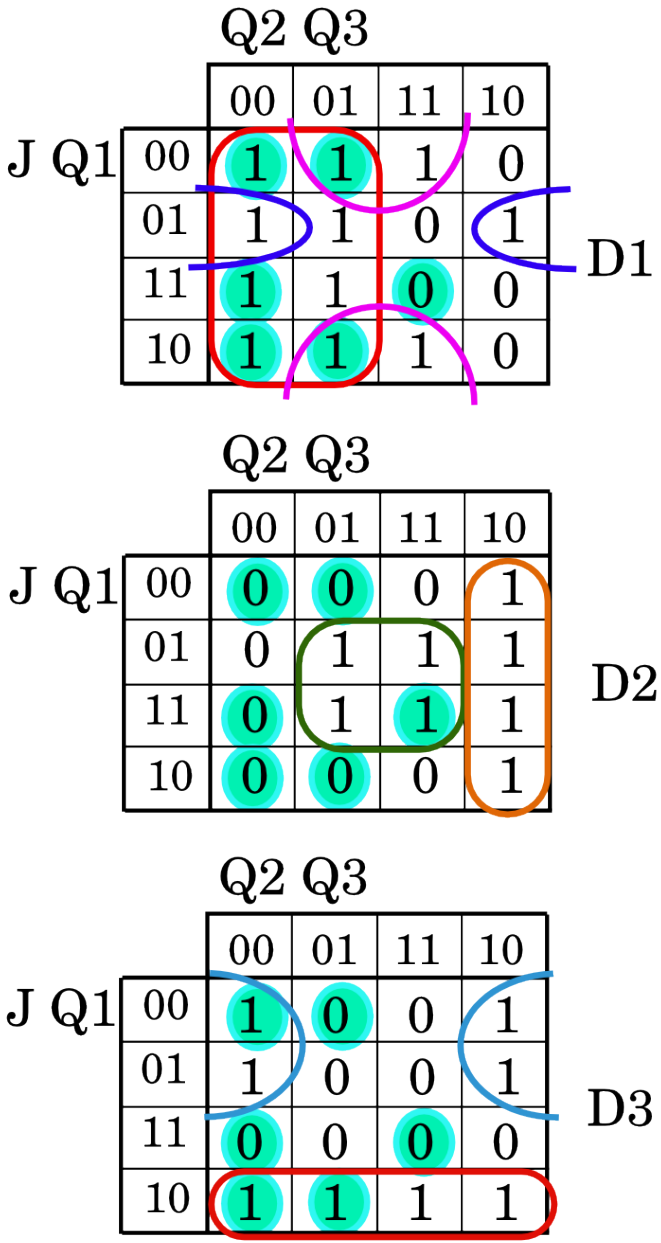
D3 = J'Q3' + J Q1'

Check the unused states

Once the circles have been decided we can deduce whether the don't cares have been made 1s or 0s. Any don't care in a circle becomes a 1 those outside circles are zeros. All eight states can now be included in the transition diagram, and the complete circuit behaviour can be found.

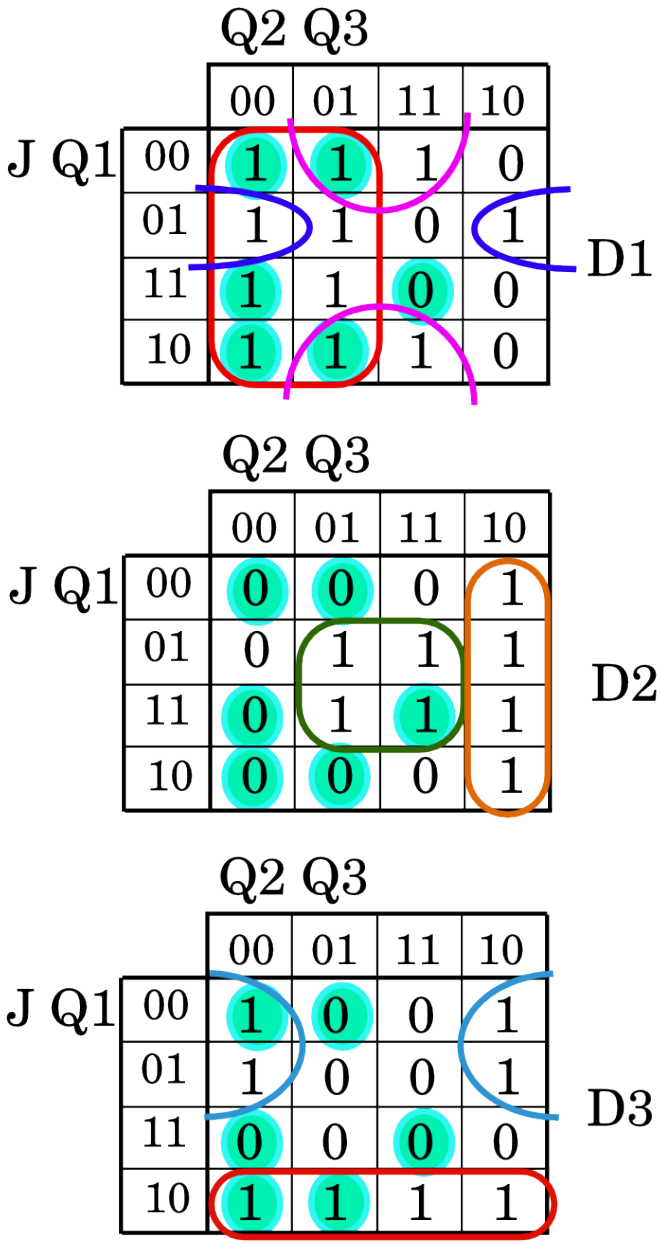


Check the unused states



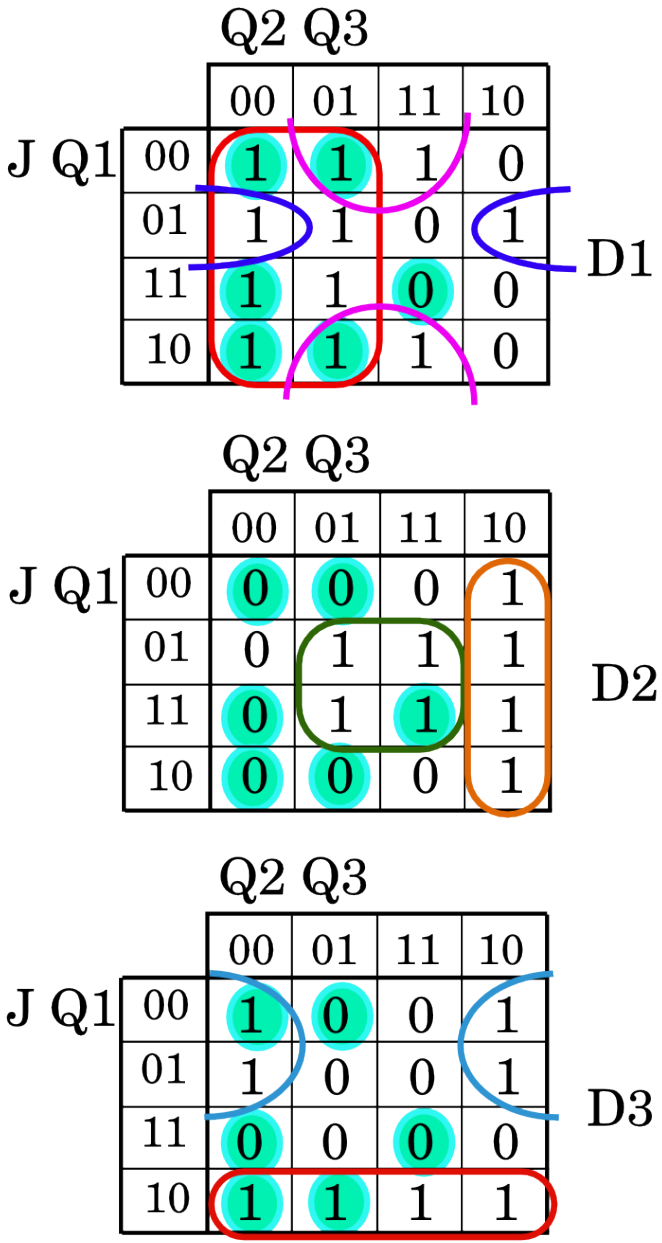
S	f-flops State (now)	State (now)	f-flops State (next)
	(next)		
0	000	(7)	(X) xxx
0	001	(8)	(X) xxx
0	010	(1)	(2) 011
0	011	(2)	(3) 100
0	100	(3)	(4) 101
0	101	(4)	(5) 110
0	110	(5)	(6) 111
0	111	(6)	(1) 010
1	000	(7)	(X) xxx
1	001	(8)	(X) xxx
1	010	(1)	(2) 011
1	011	(2)	(4) 101
1	100	(3)	(X) xxx
1	101	(4)	(5) 110
1	110	(5)	(1) 010
1	111	(6)	(X) xxx

Check the unused states



S	f-flops State (now)	State (now)	f-flops (next)
	(next)		
0	000	(7)	(4) 101
0	001	(8)	(3) 100
0	010	(1)	(2) 011
0	011	(2)	(3) 100
0	100	(3)	(4) 101
0	101	(4)	(5) 110
0	110	(5)	(6) 111
0	111	(6)	(1) 010
1	000	(7)	(4) 101
1	001	(8)	(4) 101
1	010	(1)	(2) 011
1	011	(2)	(4) 101
1	100	(3)	(3) 100
1	101	(4)	(5) 110
1	110	(5)	(1) 010
1	111	(6)	(1) 010

Check the unused states

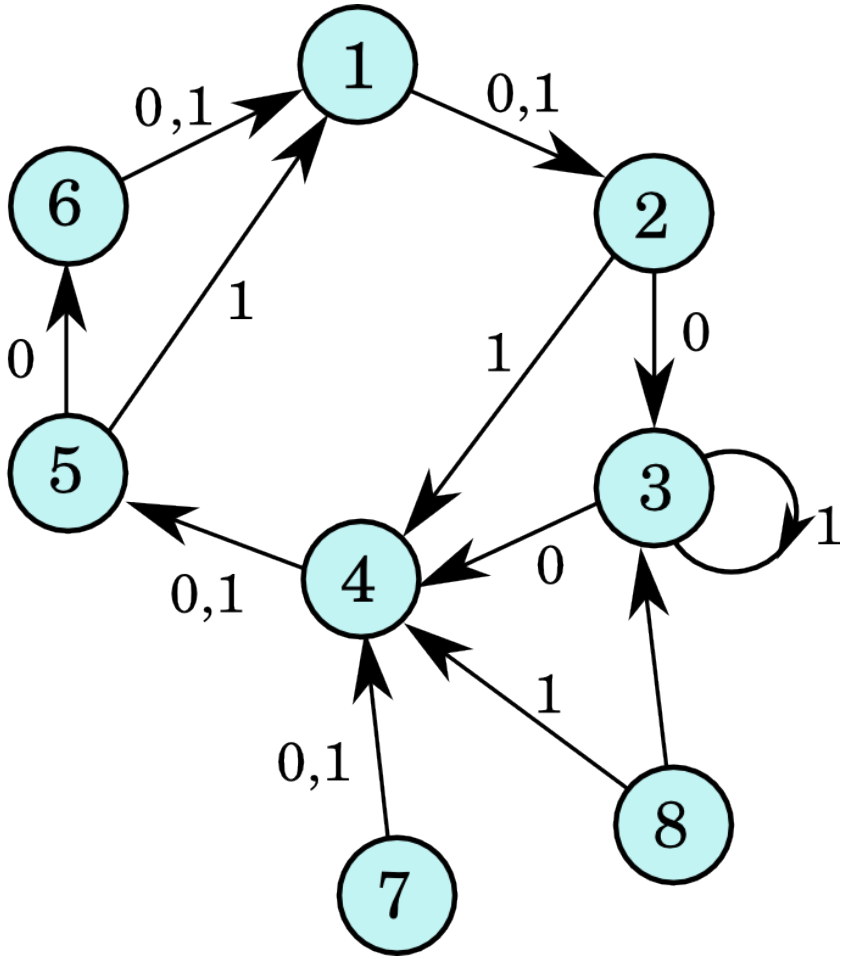


S	f-flops (now)	State (now)	State (next)	f-flops (next)
0	000	(7)	(4)	101
0	001	(8)	(3)	100
0	010	(1)	(2)	011
0	011	(2)	(3)	100
0	100	(3)	(4)	101
0	101	(4)	(5)	110
0	110	(5)	(6)	111
0	111	(6)	(1)	010
1	000	(7)	(4)	101
1	001	(8)	(4)	101
1	010	(1)	(2)	011
1	011	(2)	(4)	101
1	100	(3)	(3)	100
1	101	(4)	(5)	110
1	110	(5)	(1)	010
1	111	(6)	(1)	010

Disaster - the system gets stuck in State 3.!

Construct the Complete Transition Diagram

S	$State_t$	$State_{t+1}$
0	(1)	(2)
0	(2)	(3)
0	(3)	(4)
0	(4)	(5)
0	(5)	(6)
0	(6)	(1)
0	(7)	(4)
0	(8)	(3)
1	(1)	(2)
1	(2)	(4)
1	(3)	(3)
1	(4)	(5)
1	(5)	(1)
1	(6)	(1)
1	(7)	(4)
1	(8)	(4)

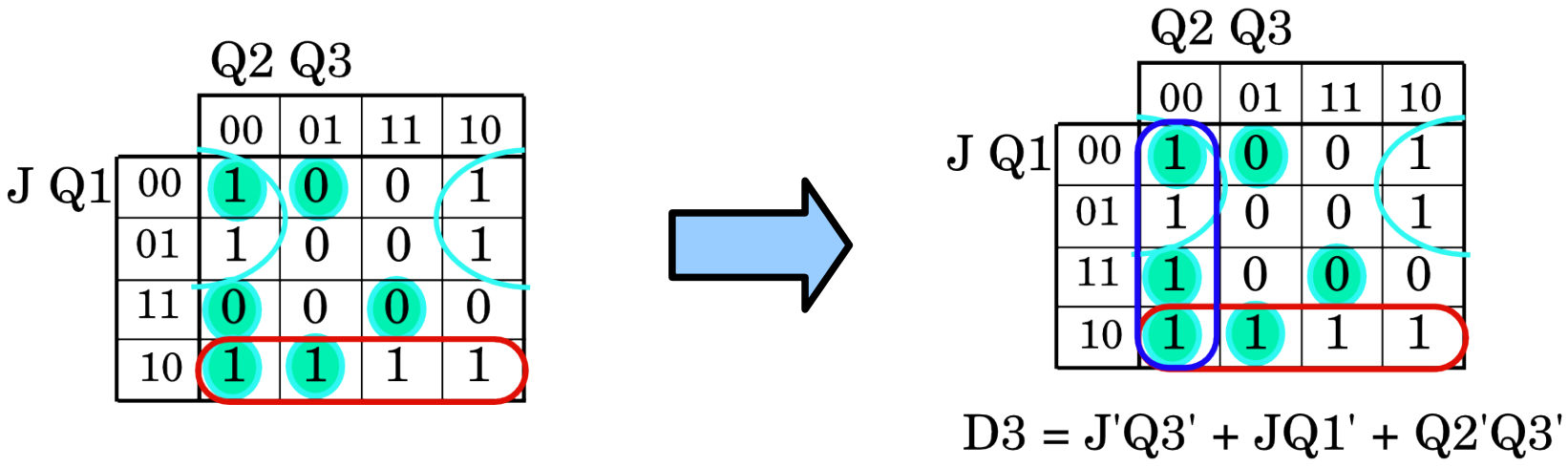


Test and Repair if Necessary

The problem occurs at one line in the transition table

J	Q1	Q2	Q3	D1	D2	D3
1	1	0	0	1	0	0

We could solve it by changing D3 to a 1. This could be done with one additional circle on the D3 Karnaugh map.



Construct the Output Circuits Truth Table

There are six such circuits and they have three inputs only. Their truth tables can be filled out by the states of the lights which are either ON or OFF for each given system state.

STATE	Q1	Q2	Q3	R1	A1	G1	R2	A2	G2
1	0	1	0	1	0	0	0	0	1
2	0	1	1	1	0	0	0	1	0
3	1	0	0	1	0	0	1	0	0
4	1	0	1	0	0	1	1	0	0
5	1	1	0	0	1	0	1	0	0
6	1	1	1	1	0	0	1	0	0

Construct the Output Circuits K-maps

Q1 \ Q2 Q3	00	01	11	10
0	X	X	1	1
1	1	0	1	0

R1

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	0	0	0	1

A1

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	0	1	0	0

G1

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	1	1	1	1

R2

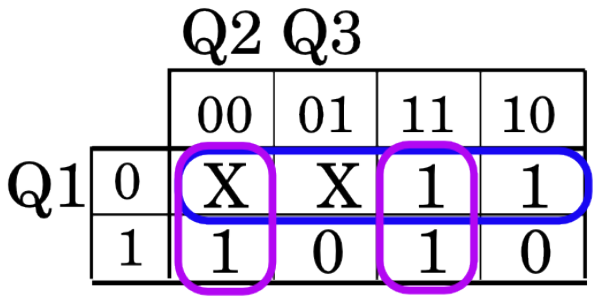
Q1 \ Q2 Q3	00	01	11	10
0	X	X	1	0
1	0	0	0	0

A2

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	1
1	0	0	0	0

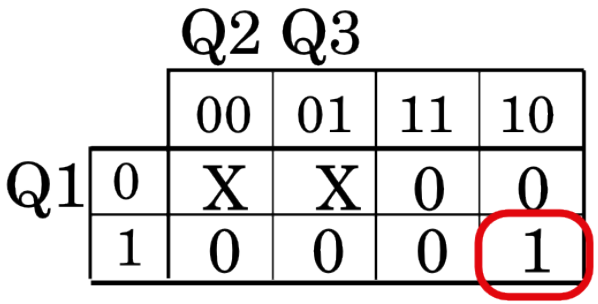
G2

Construct the Output Circuits K-maps



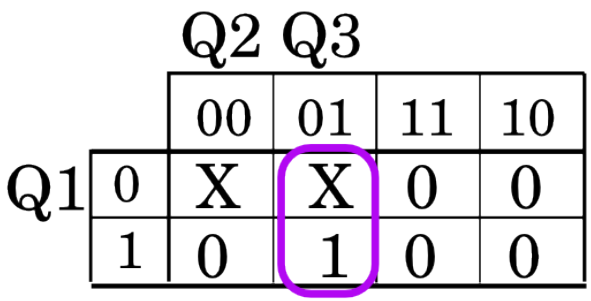
Q1 \ Q2 Q3	00	01	11	10
0	X	X	1	1
1	1	0	1	0

R1 = Q1'+Q2'Q3'+Q2Q3



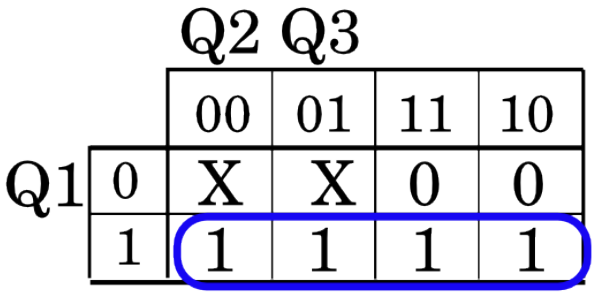
Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	0	0	0	1

A1 = Q1Q2Q3'



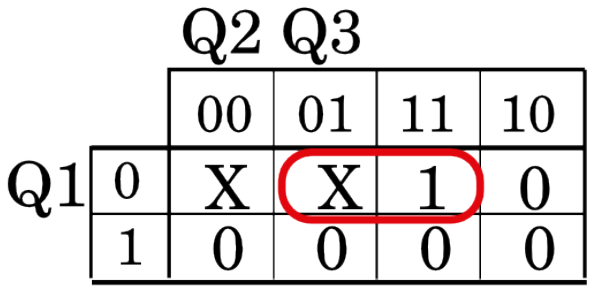
Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	0	1	0	0

G1 = Q2'Q3



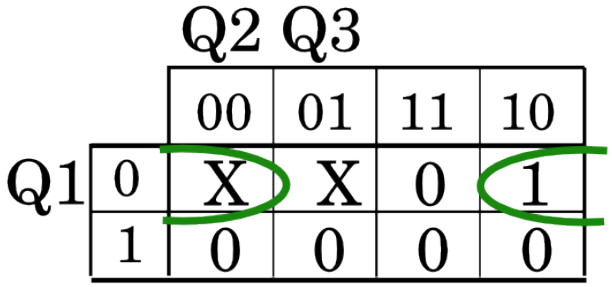
Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	1	1	1	1

R2 = Q1



Q1 \ Q2 Q3	00	01	11	10
0	X	X	1	0
1	0	0	0	0

A2 = Q1'Q3



Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	1
1	0	0	0	0

G2 = Q1'Q3'

We have the following circuits to build:

D1 = Q2' + J'•Q1•Q3' + Q1'•Q3

D2 = Q1•Q3 + Q2•Q3'

D3 = Q2'•Q3' + J'•Q3' + J•Q1'

R1 = Q1' + Q2'•Q3' + Q2•Q3

A1 = Q1•Q2•Q3' or Q2•Q1•Q3'

G1 = Q2'•Q3

R2 = Q1

A2 = Q1'•Q3

G2 = Q1'•Q3'

Terms that appear more than once are underlined

Try A Different State Assignment

If we want to try to find a simpler overall circuit, we may try different flip-flop assignments for the states. One idea is to minimise the output circuitry.

We could, for example, try setting Q1=R1 and Q2=R2, to see if these simple assignments will give us a correct complete state assignment.

STATE	Q1	Q2	Q3	R1	A1	G1	R2	A2	G2
1	1	0	0	1	0	0	0	0	1
2	1	0	1	1	0	0	0	1	0
3	1	1	1	1	0	0	1	0	0
4	0	1	1	0	0	1	1	0	0
5	0	1	0	0	1	0	1	0	0
6	1	1	0	1	0	0	1	0	0
7	0	0	0	X	X	X	X	X	X
8	0	0	1	X	X	X	X	X	X

A Different State Assignment (The output logic is simplified)

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	1	1	1	1

R1 = Q1

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	1
1	0	0	0	0

A1 = Q1'Q3'

Q1 \ Q2 Q3	00	01	11	10
0	X	X	1	0
1	0	0	0	0

G1 = Q1'Q3

Q1 \ Q2 Q3	00	01	11	10
0	X	X	1	1
1	0	0	1	1

R2 = Q2

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	0	1	0	0

A2 = Q2'Q3

Q1 \ Q2 Q3	00	01	11	10
0	X	X	0	0
1	1	0	0	0

G2 = Q2'Q3'

A Different State Assignment (The state sequencing logic looks simpler)

J Q1 \ Q2 Q3	00	01	11	10
00	X	X	0	1
01	1	1	0	1
11	1	0	X	X
10	X	X	0	1

D1 = J'Q2' + Q3'

J Q1 \ Q2 Q3	00	01	11	10
00	X	X	1	1
01	0	1	1	0
11	0	1	X	X
10	X	X	1	0

D2 = J'Q1' + Q3

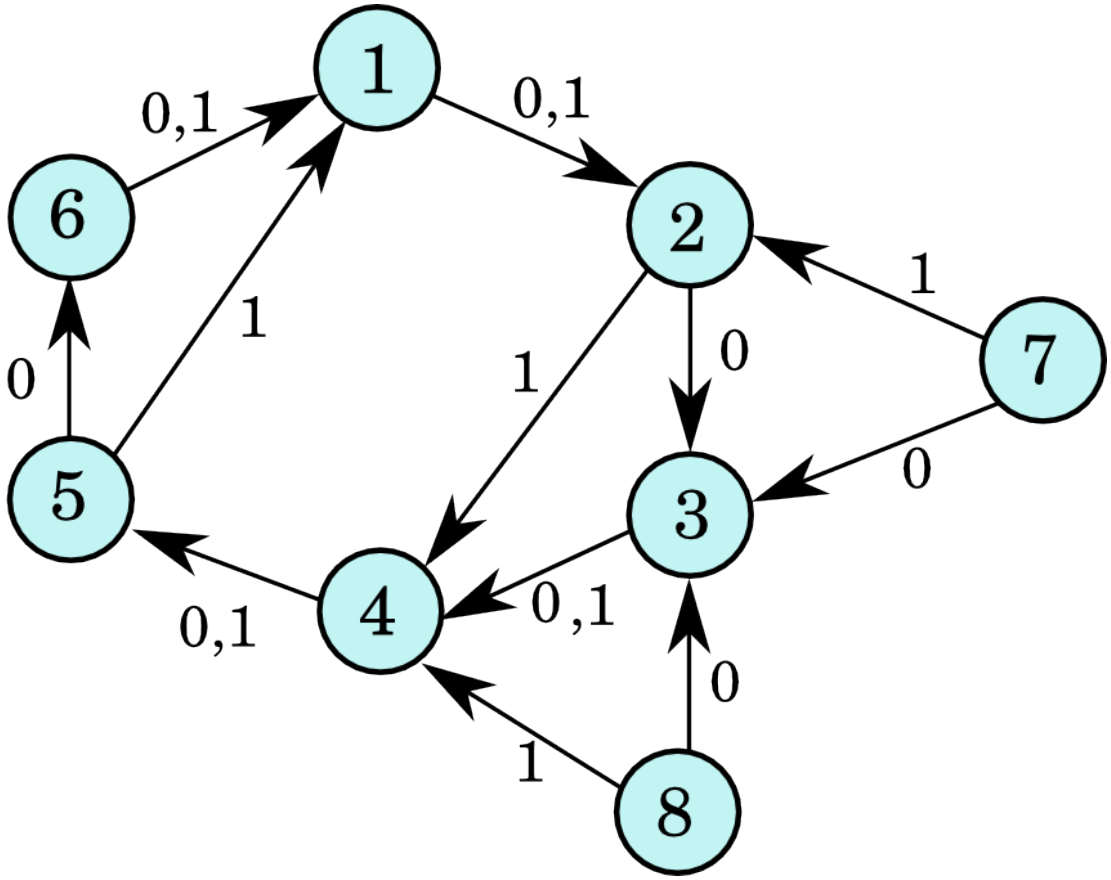
J Q1 \ Q2 Q3	00	01	11	10
00	X	X	0	0
01	1	1	1	0
11	1	1	X	X
10	X	X	0	0

D3 = Q2' + Q1Q3

This circuit seems to be simpler than the first, but whether it is cheaper or not can be established only after the circuit is designed and costed.

A Different State Assignment (Transition diagram)

S	$State_t$	$State_{t+1}$
0	(1)	(2)
0	(2)	(3)
0	(3)	(4)
0	(4)	(5)
0	(5)	(6)
0	(6)	(1)
0	(7)	(3)
0	(8)	(3)
1	(1)	(2)
1	(2)	(4)
1	(3)	(4)
1	(4)	(5)
1	(5)	(1)
1	(6)	(1)
1	(7)	(2)
1	(8)	(4)



The circuit looks safe enough

A Different State Assignment (The final circuit)

D1 = J'•Q2'+Q3’
D2 = J'•Q1'+Q3
D3 = Q2'+Q1•Q3
R1 = Q1
A1 = Q1'•Q3'
G1 = Q1'•Q3
R2 = Q2
A2 = Q2'•Q3
G2 = Q2'•Q3'