

Problem 1

Primes: (0,2,8,10), (5,13), (9,13), (8,9), (5,7)

Essential Primes: (0,2,8,10)

There are 3 different ways to write down this boolean function:

i) $F = (0,2,8,10) + (5,13) + (9,13) = b'd' + bc'd + ac'd$

ii) $F = (0,2,8,10) + (5,13) + (8,9) = b'd' + bc'd + ab'c'$

iii) $F = (0,2,8,10) + (5,7) + (9,13) = b'd' + a'bd + ac'd$

Problem 2

The first step of the Quine-McCluskey method is to find all the prime implicants of the function. Here the first two steps leave us with 3 primes.

0: 000

1: 001 (0,1): 00-

2: 010 -> (0,2): 0-0

6: 110 (2,6): -10

7: 111

Note that both minterms 6 and 7 are don't cares, so the potential pairing (6,7) is not a prime implicant. So the algorithm doesn't take it in account for the next step.

Next, we construct the implication chart to figure out which of those we are going to keep:

	0	1	2
(0,1)	v	v	
(0,2)	v		v
(2,6)			v

	2
(0,2)	v
(2,6)	v

Of the primes only (0,1) is an essential one. Finally, there are two ways to write down the function:

i) $F = (0,1) + (0,2) = a'b' + a'c'$

ii) $F = (0,1) + (2,6) = a'b' + bc'$

Problem 3

i) $\{\text{NAND}, \text{NOR}\} = \text{YES}$

You can construct NOT by hardwiring either a NAND or a NOR with the same input. For an AND, you use a NAND with x and y as inputs followed by another NAND that has both its inputs to be the output of the first NAND. For an OR, you use a NOR with x and y as inputs followed by another NOR that has both its inputs to be the output of the first NOR.

ii) $\{\text{XOR}, \text{NOT}\} = \text{NO}$

There is no formal way to prove this, so instead we will try all the possible combinations of inputs, in order to show that we cannot construct neither an AND nor an OR gate.

The function for XOR is: $f(x,y) = xy' + x'y$

Using 0 and 1 as valid inputs, all we can get is:

$$f(x,1) = x', \quad f(x,0) = x, \quad f(x,x) = 0,$$

$$f(x, f(x,y)) = x \# (x \# y) = (x \# x) \# y = 0 \# y = y$$

where $\#$ means XOR.

iii) $\{f(x,y) = x'y\} = \text{YES}$

$$f(x,1) = x', \quad f(f(x,1),y) = xy,$$

$$f(f(x, f(y,1)),1) = f(f(x,y'),1) = f(x'y',1) = (x'y')' = x+y$$

iv) $\{f(x,y,z) = (x+y)z'\} = \text{YES}$

$$f(1,1,z) = z', \quad f(x,y,0) = x+y,$$

$$f(x,0, f(1,1,z)) = f(x,0,z') = (x+0)(z')' = xz$$

Problem 4

The first thing you need to do is to identify whether the state machine is a Moore or a Mealy one. This is derived from the excitation function of the output. In this case it is $y = Q_1 + Q_0'$, which means that this machine is a Moore one. Later I will change the output, making the machine a Mealy one, just to showcase the difference in drawing the state diagrams.

First we need the excitation functions and the next state functions for the two flip-flops.

$$T_0 = x'Q_1$$

$$T_1 = xQ_0'$$

And

$$Q_0^* = Q_0 \text{ XOR } T_0 = x'Q_1Q_0' + xQ_0 + Q_1'Q_0$$

$$Q_1^* = Q_1 \text{ XOR } T_1 = xQ_1'Q_0' + x'Q_1 + Q_1Q_0$$

The output for the Moore machine is: $y = Q_1 + Q_0'$

We also do a modification here and we define an additional output:

$$z = x'(Q_1 + Q_0')$$

that defines a Mealy machine.

The state transition table is the following:

Q1	Q0	x=0	x=1	y	x=0	x=1
0	0	00	10	1	1	0
0	1	01	01	0	0	0
1	0	11	00	1	1	0
1	1	10	11	1	1	0
		Q1Q0	Q1Q0		z	

The state diagrams for the Moore and Mealy machines are given in figures 1 and 2 respectively.

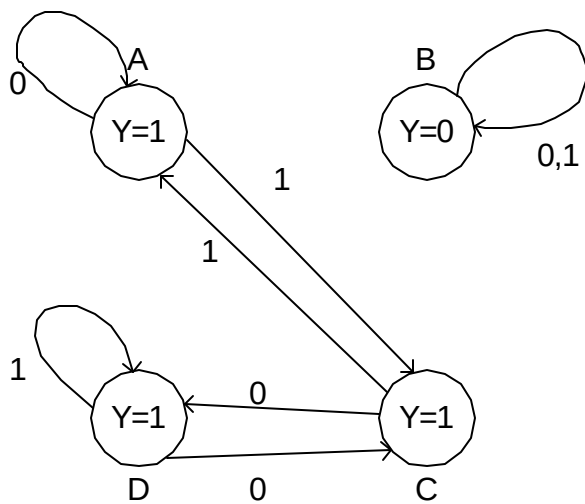


Figure 1

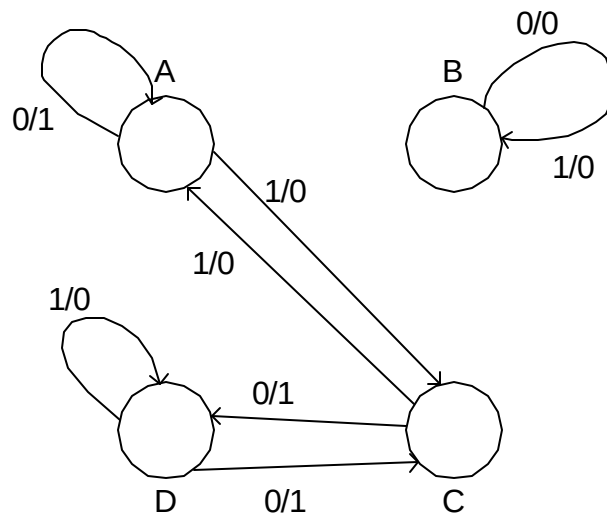


Figure 2

Problem 5

The state transition table for the flip-flop in design is the following:

J	K	Q	Q*	T
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	1

The excitation function for the input of the T flip-flop is:

$$T = JQ' + KQ$$

So we can build a JK flip-flop by using a T one, like this:

