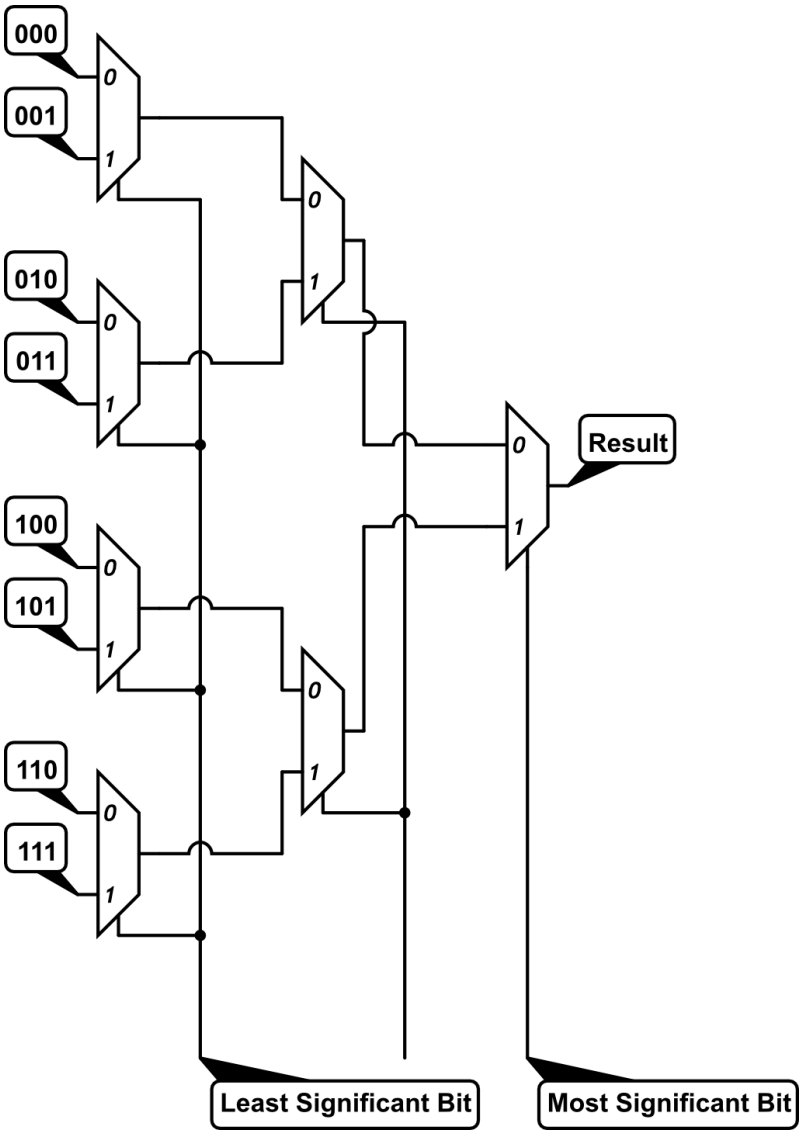
CST Part IA: Digital Design, SV 2
Joe Yan
2016-10-17

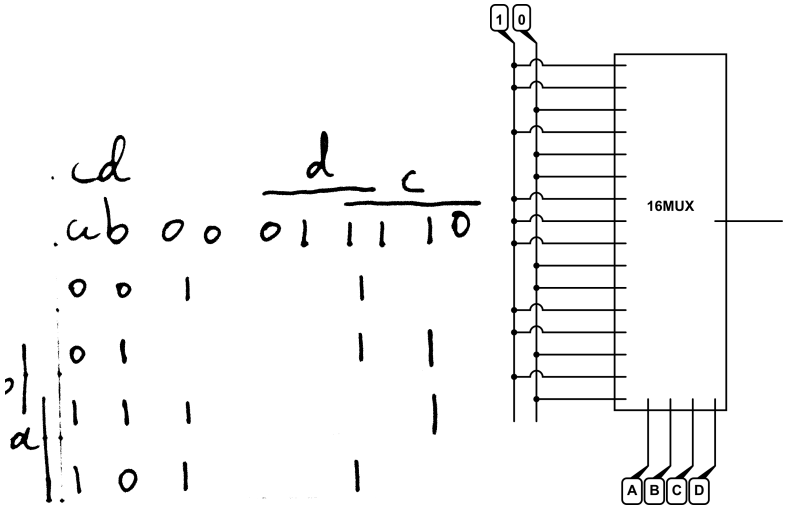
Exercise 1



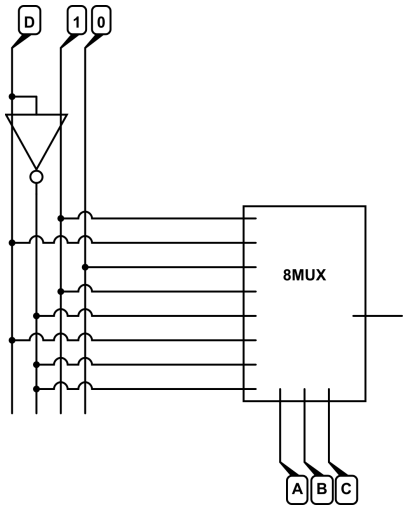
The idea is once a control input of a "higher level" multiplexor is chosen either 0 or 1 then it is actually looking up the corresponding input from a "lower level" multiplexor output. And this idea goes through the whole circuit until get the lowest level multiplexor and take the 1 or 0 from the input. In this way,the control input of the highest level multiplexor is controlling the most significant bit of the truth table and the lowest one is controlling the least significant bit.
And the picture shows how it is built.

Exercise 2

(a)



(b)



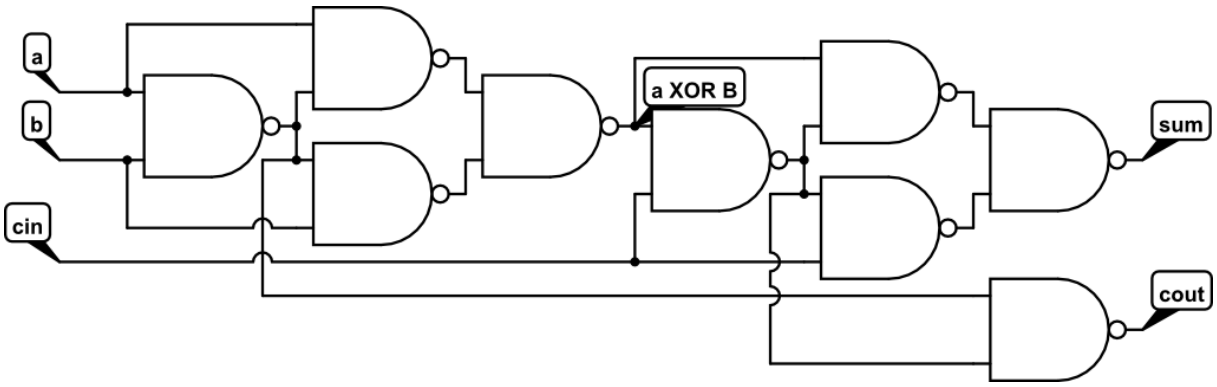
A	B	C	D	result	
0	0	0	0	1	1
0	0	0	1	1	
0	0	1	0	0	D
0	0	1	1	1	
0	1	0	0	0	0
0	1	0	1	0	
0	1	1	0	1	1
0	1	1	1	1	
1	0	0	0	1	$\bar{D}$
1	0	0	1	0	
1	0	1	0	0	D
1	0	1	1	1	
1	1	0	0	1	$\bar{D}$
1	1	0	1	0	
1	1	1	0	1	$\bar{D}$
1	1	1	1	0	

* n means not related with D.
Because for specific A,B,C the result is either based on D or $\bar{D}$ or unrelated to D and constrained to 1 or 0. So the D or $\bar{D}$ can be an input instead of control input.

Exercise 3

c_{in}	a	b	c_{out}	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$sum = a \oplus b \oplus c$
$c_{out} = a.b + c_{in}(a \oplus b)$
$a \oplus b$
$= \bar{a}b + a\bar{b}$
$= \overline{\overline{b\overline{ab}} + \overline{a\overline{ab}}}$
$= \overline{\overline{b\overline{ab}} + a\overline{ab}}(*)$
$= \overline{\overline{b\overline{ab}}.\overline{a\overline{ab}}}$
c_{out} can also use the idea of (*).
$c_{out} = a.b + c_{in}(a \oplus b)$
$= \overline{\overline{a.b} + \overline{c_{in}(a \oplus b)}}$
$= \overline{\overline{a.b}.\overline{c_{in}(a \oplus b)}}$
Because $a \oplus b$ is defined before is no need to worry.



Exercise 4

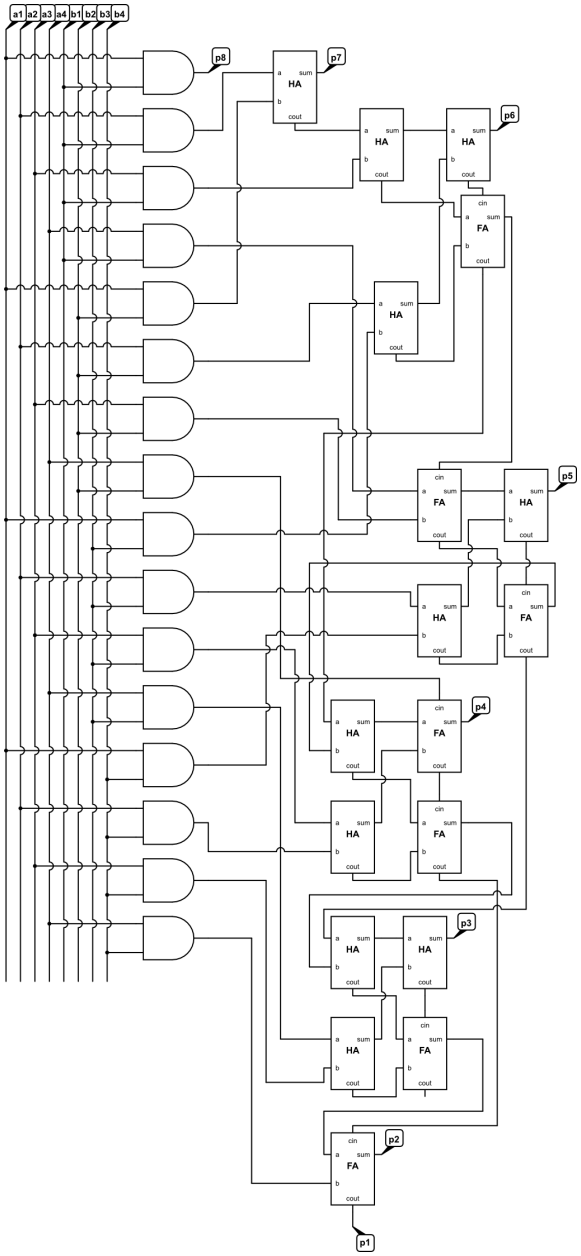
(a) $\bar{B} + C$

(b) $ABC + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C} + A\bar{B}C$

(c) $\bar{B} + C$

(d)

Exercise 5

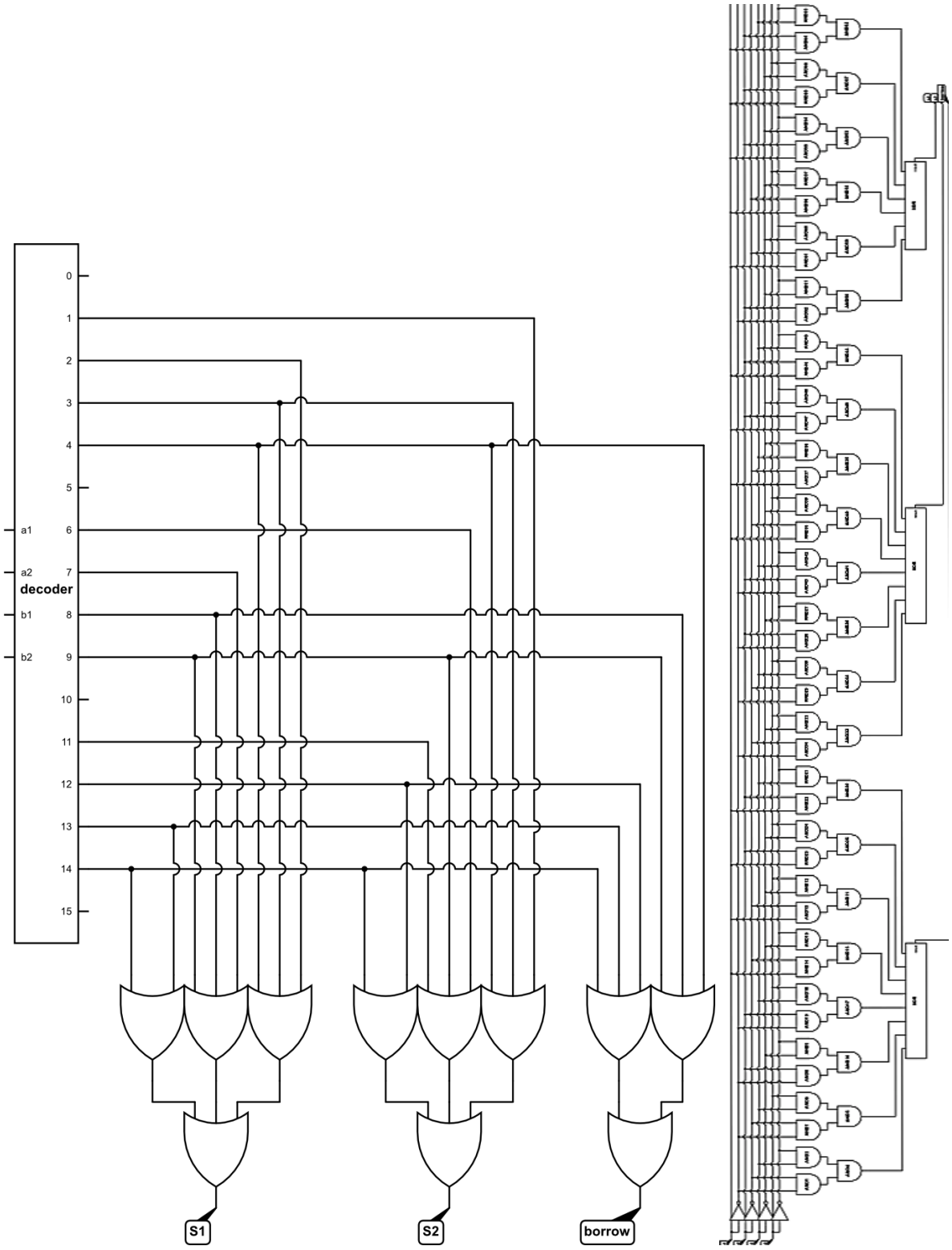


*x_n means the nth number from most significant to least significant of binary number x.
This idea comes from the basic product operation by human. And notice the one of the full adder close to bottom is impossible to have a c_{out} as one.

Exercise 6

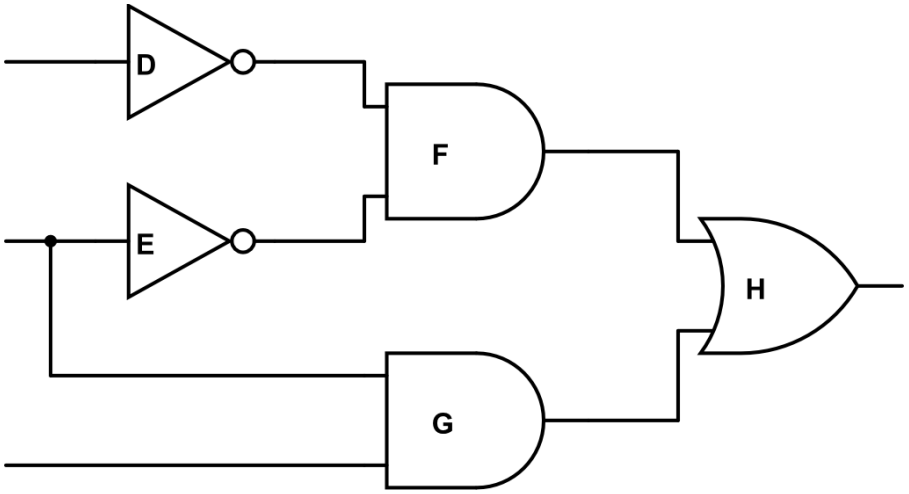
a_1	a_2	b_1	b_2	s_1	s_2	$borrow$
0	0	0	0	0	0	0
0	0	0	1	0	1	0
0	0	1	0	1	0	0
0	0	1	1	1	1	0
0	1	0	0	1	1	1
0	1	0	1	0	0	0
0	1	1	0	0	1	0
0	1	1	1	1	0	0
1	0	0	0	1	0	1
1	0	0	1	1	1	1
1	0	1	0	0	0	0
1	0	1	1	0	1	0
1	1	0	0	0	1	1
1	1	0	1	1	0	1
1	1	1	0	1	1	1
1	1	1	1	0	0	0

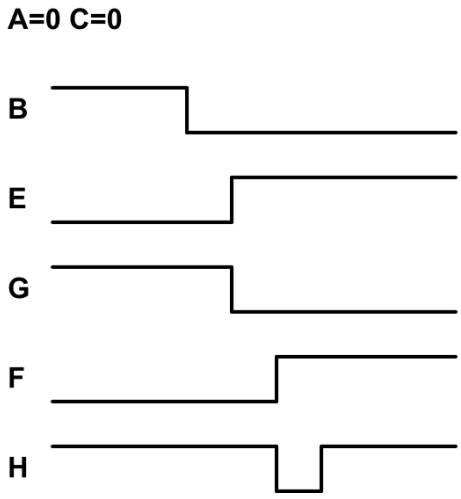
m means the number of bits in each address.
n means the number of input for address looking-up.(So there are 2^n different address)
Number of bit stored: $m \times 2^n$
The ROM is basically a n-input looking-up table for m-bit words. Each output of the decoder represents a minterm. And combining these minterm gives a SOP boolean formula hence to represent a looking-up table.
The implement for PAL is also by a SOP formula, the complement or uncomplement of each variable will join in AND gates to represent a minterm and then the output of these AND gates will be input to OR gates again to give an output as SOP formula.Also a PAL is written by physically break the connection between gates.



* Rectangle represents the n-input OR gate.

Exercise 7



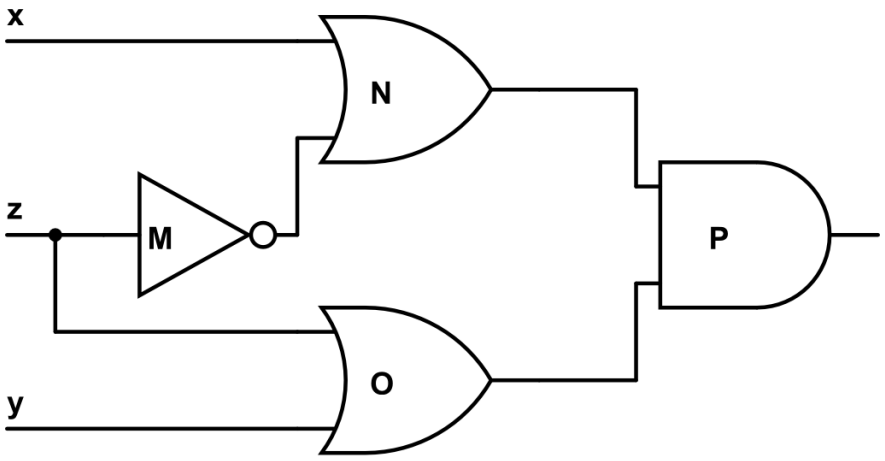


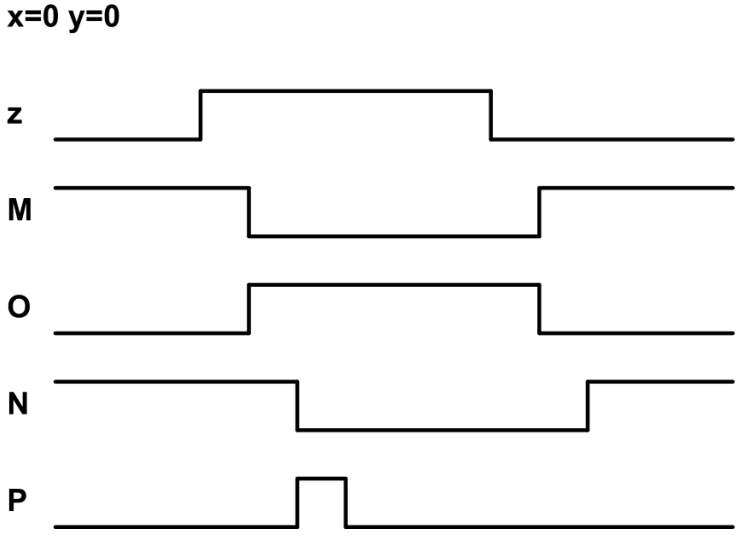
(i)

This will have a 1 static hazard. Propagation delay: $3T$
Contamination delay: $2T$
FIX:
$f = \bar{A}\bar{B} + BC$

C AB	00	01	11	10
0	1	0	0	0
1	1	1	1	0

To remove the 1 static hazard, 1 more term is need on the K-Map to connect the previous 1 terms. It is $\bar{A}C$.
Now,$f = \bar{A}\bar{B} + BC + \bar{A}C$





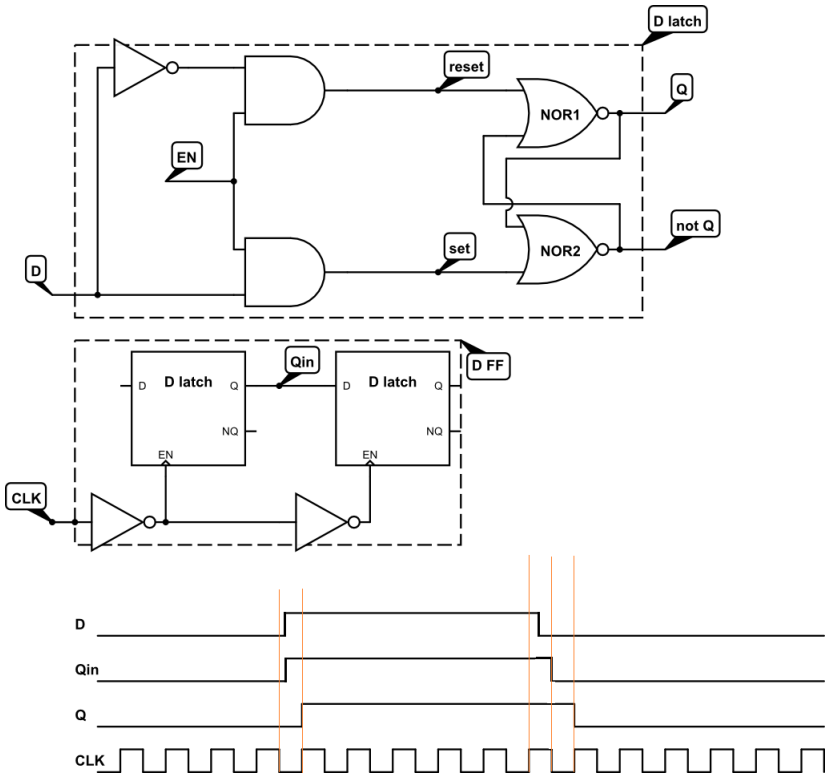
(ii) This will have a 0 static hazard. Propagation delay: $3T$
Contamination delay: $2T$
g is a 2-multiplexor.
FIX:
$f = Y\bar{Z} + XZ$
$\bar{f} = \bar{X}Z + \bar{Y}\bar{Z}$

C AB	00	01	11	10
0	0	1	1	0
1	0	0	1	1

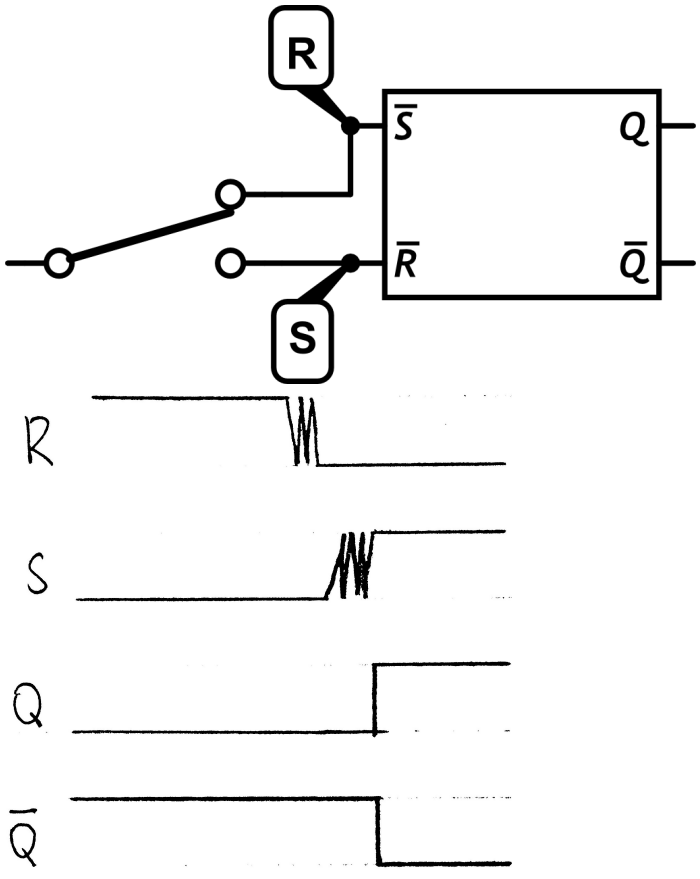
To remove the 0 static hazard, 1 more term is need on the K-Map to connect the previous 0 terms. It is $\bar{X}\bar{Y}$. Now $\bar{f} = \bar{X}Z + \bar{Y}\bar{Z} + \bar{X}\bar{Y}$
and $f = (X + \bar{Z})(Y + Z)(X + Y)$

Exercise 8

(a) The flip-flop will give a synchronous output but a latch will not if the input is asynchronous.
To make a flip-flop from a latch, a clock is needed to be as an active input (EN) to enable the state of the latch to change when EN is high or low.

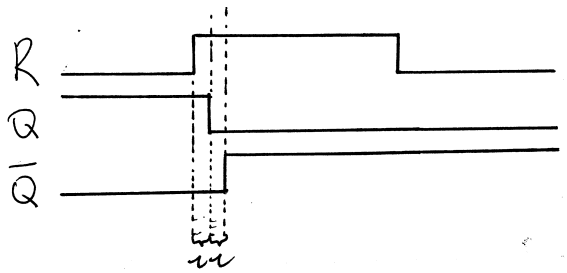

D latch
reset
Q
NOR1
EN
not Q
set
NOR2
D
Qin
D FF
D latch
D latch
D
Q
NQ
EN
CLK
D
Qin
Q
CLK

(b)


R
S
Q
R
S
Q
Q

I assume the bounce time will change between 0 and 1 very quickly and each time period keeping at 0 or 1 will be too short comparing with propagation delay and not enough to change the state of the latch. Only after the bouncing is finished then the state of latch will be able to change.

(c)



The state change minimum time: $t > 2\tau$
When the R is switched back to 0 as far as $\bar{Q}$ has already changed to one then Q will remain on changed 0 state otherwise Q will drop back to its previous state which is 1.
So when $2\tau > t$ is true, the state change will fail. On other words, the minimum t will be 2τ.

(d) If the setup time is violated, the actual time change of Q might be slightly delayed to make the output asynchronous.
if the hold time is violated, it will be risky that the change of states fails as (c) declared.
These situation may occur when the other part of the circuit are not obeying the same standard regularity.

(e) If the setup time is eliminated and the propagation time is very short, then when the FF output changes this may lead to the next FF state also changes immediately instead of waiting until next clock cycle.

Exercise 9

Tristate buffer accepts inputs and a output enable signal to decide whether the tristate is electrically connected to a bus line. Because there may be numbers of circuit connected to one bus line so it is necessary to ensure only output is actually connected to the bus line to avoid messing the signal.

Exercise 10

The list is generated by c++.
The column from left to right means number of time cycle, BIN, DEC, OCT and HEX.
Because there are 5 bits after at most $2^5 - 1$ state changes, the state must be repeat as one previous appeared so pattern repeats again.

0	11111	31	37	1f
1	01111	15	17	f
2	00111	7	7	7
3	00011	3	3	3
4	10001	17	21	11
5	11000	24	30	18
6	01100	12	14	c
7	10110	22	26	16
8	11011	27	33	1b
9	11101	29	35	1d
10	01110	14	16	e
11	10111	23	27	17
12	01011	11	13	b
13	10101	21	25	15
14	01010	10	12	a
15	00101	5	5	5
16	00010	2	2	2
17	00001	1	1	1
18	10000	16	20	10
19	01000	8	10	8
20	00100	4	4	4
21	10010	18	22	12
22	01001	9	11	9
23	10100	20	24	14
24	11010	26	32	1a
25	01101	13	15	d
26	00110	6	6	6
27	10011	19	23	13
28	11001	25	31	19
29	11100	28	34	1c
30	11110	30	36	1e
31	11111	31	37	1f

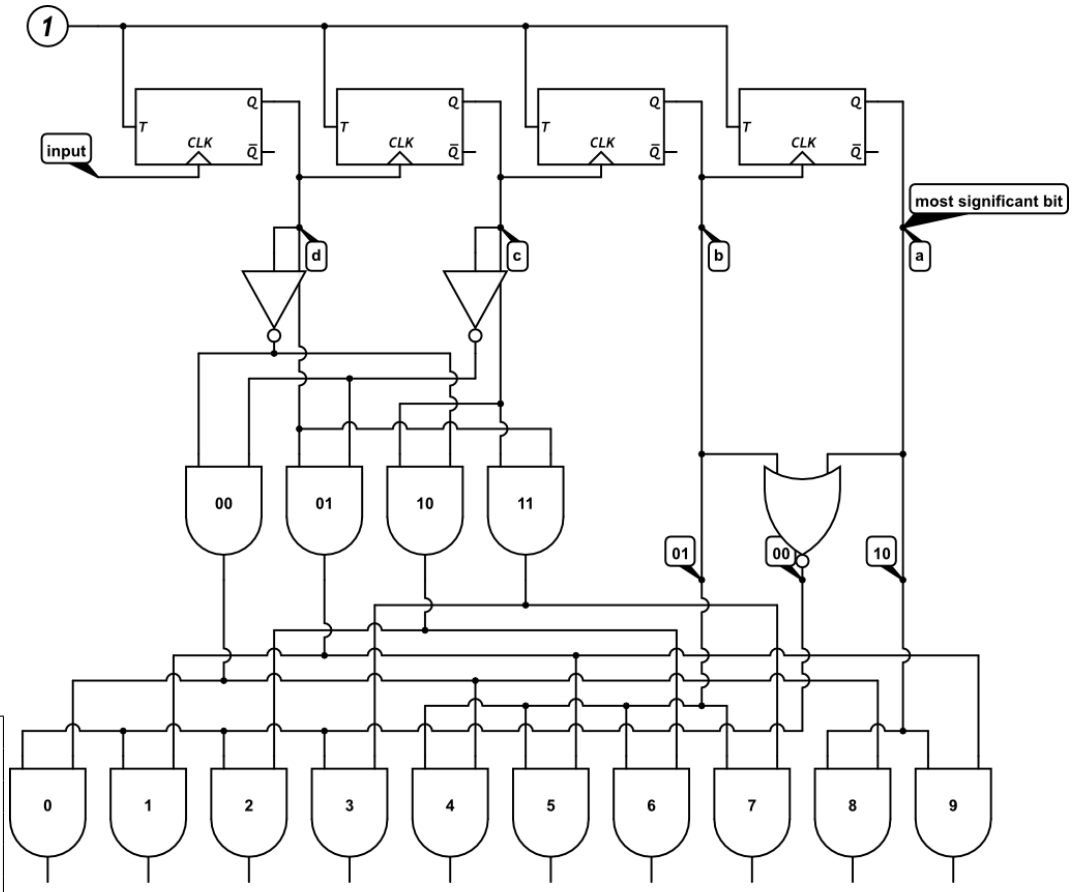
If all five states are set 0 then it will always be all 0.

Finally, I can say all 31 combinations of the example is kind of a closed set under the operation of this circuit as well as 00000.

Exercise 11

The idea is to ripple counter to convert the series of pulses to a binary number and then to a decimal number.
Define the output of the counter from most significant bit as a, b, c, d.

$ab\ cd$	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	x	x	x	x
10	8	9	x	x



0	$\bar{a}b\bar{c}d$
1	$\bar{a}\bar{b}\bar{c}d$
2	$\bar{a}\bar{b}cd\bar{}$
3	$\bar{a}\bar{b}cd$
4	$b\bar{c}\bar{d}$
5	$b\bar{c}d$
6	$bcd\bar{}$
7	bcd
8	$a\bar{c}\bar{d}$
9	$a\bar{c}d$

QUESTION:Is there actually a systematic way to find the minimum numbers of required gates required in a specific circuit such as this or Q3?

Exercise 12

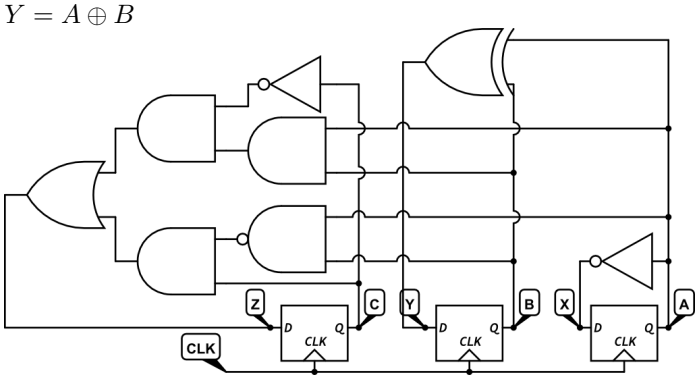
15. The idea is to use D-FF to generate next signal and store it for a clock period.

CBA	ZYX
000	001
001	010
010	011
011	100
100	101
101	110
110	111
111	000

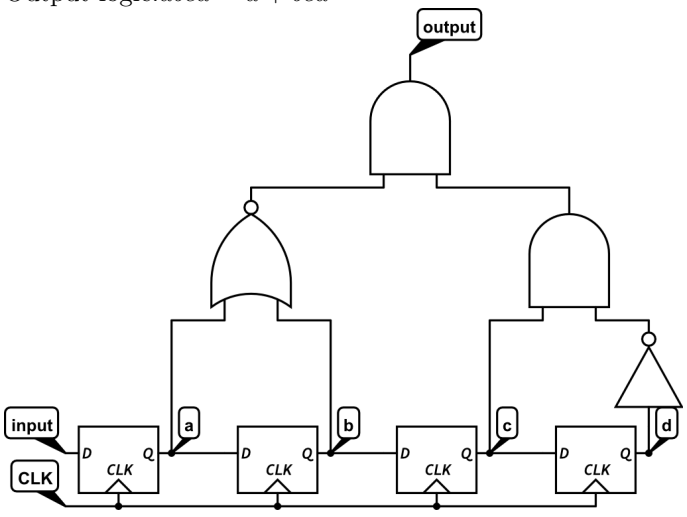
K-Map for Z.

C BA	00	01	11	10
0			1	
1	1	1		1

From table:
$Z = AB\bar{C} + \bar{B}C + \bar{A}C = \overline{AB}C + AB\bar{C}$
Also X,Y is:
$X = \bar{A}$
$Y = A \oplus B$



17. The idea is to use shift register to get new signal and then move them forward and check the output.
Output logic:$\bar{a}\bar{b}c\bar{d} = \overline{a + b c \bar{d}}$



Exercise 13

The idea is to use D-FF to store the previous signal and generate the next stage.
Define the output to be A,B,C and the next output to be $A'B'C'$.

cycle 1
000
011
110
100
001
000

cycle 2
010
101
111
010

current	next	
000	011	circle 1
011	110	
110	100	
100	001	
001	000	
010	101	circle 2
101	111	
111	010	

The K-Map for A':

C AB	00	01	11	10
0		1	1	
1		1		1

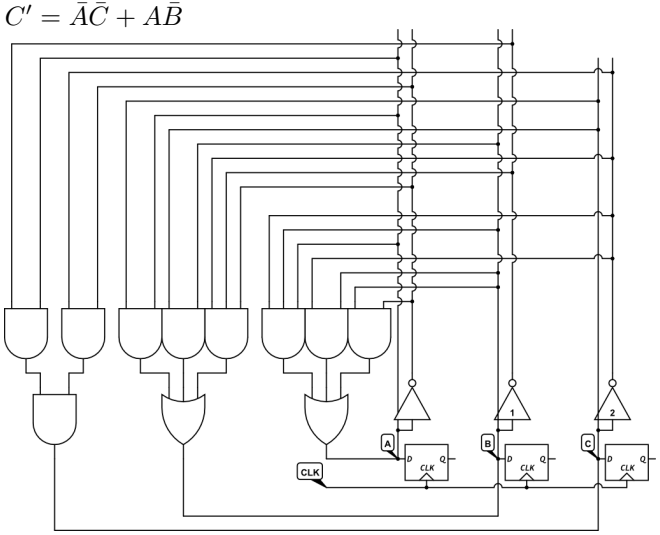
The K-Map for B':

C AB	00	01	11	10
0	1			
1		1	1	1

The K-Map for C':

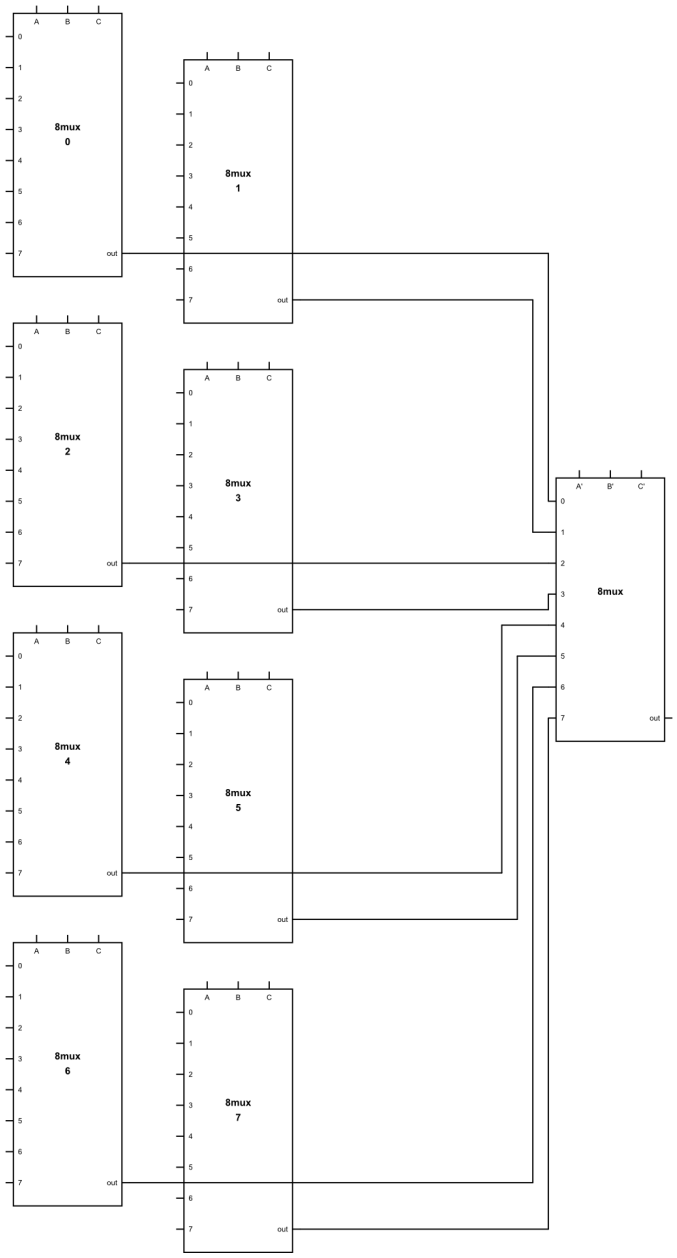
C AB	00	01	11	10
0	1	1		1
1				1

$A' = \bar{A}B + B\bar{C} + A\bar{B}C$
$B' = \bar{A}\bar{B}\bar{C} + BC + AC$
$C' = \bar{A}\bar{C} + A\bar{B}$

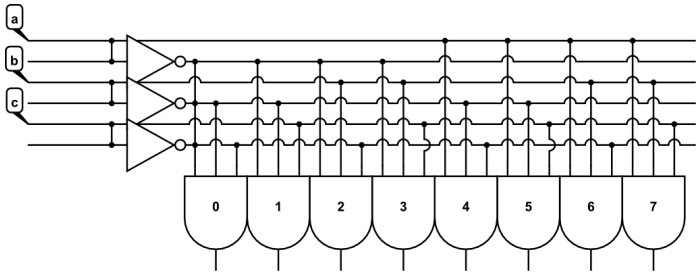


Exercise 16

1. A,B,C are the control inputs and significance from most to least.
   I_n $n = 1, 2, ..7$ are inputs.
   Equation for true output: $\bar{A}\bar{B}\bar{C}I_0 + \bar{A}\bar{B}CI_1 + \bar{A}B\bar{C}I_2 + \bar{A}BCI_3 + A\bar{B}\bar{C}I_4 + A\bar{B}CI_5 + AB\bar{C}I_6 + ABCI_7$

2. 

The control input of all left hand side 8 MUXes should be connect together such all control input have the same name should have the same input. And so the control signal should be $A'B'C'ABC$ (significance from most to least). And all 64 inputs will build up the truth table for the 64-MUX.

3. 

4. Each output of the decoder is actually representing a minterm of the input. So we can

use multiple OR gates to connect some or all of the output to build up the SOP formula for output. Then it is using the input address to take out the data previously stored in the circuit.

Exercise 17

(a) 4

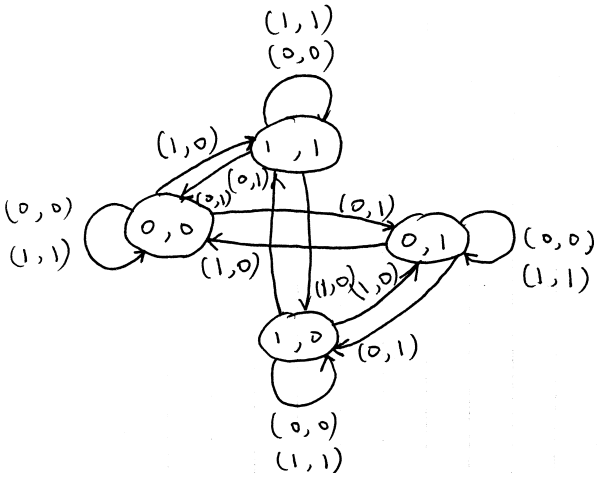
(b) 16

(C)

X	Y	A	B	X'	Y'
0	0	0	0	H	H
0	0	0	1	H	T
0	0	1	0	T	T
0	0	1	1	H	H
0	1	0	0	H	H
0	1	0	1	T	T
0	1	1	0	H	T
0	1	1	1	H	H
1	0	0	0	H	H
1	0	0	1	H	T
1	0	1	0	T	T
1	0	1	1	H	H
1	1	0	0	H	H
1	1	0	1	T	T
1	1	1	0	H	T
1	1	1	1	H	H

* H is HOLD. T is TOGGLE.

(d)



(e) When the input A,B is the same, the state remains.
when the input A,B is different, it either change the state of Y and remain X or change the states of both X and Y.