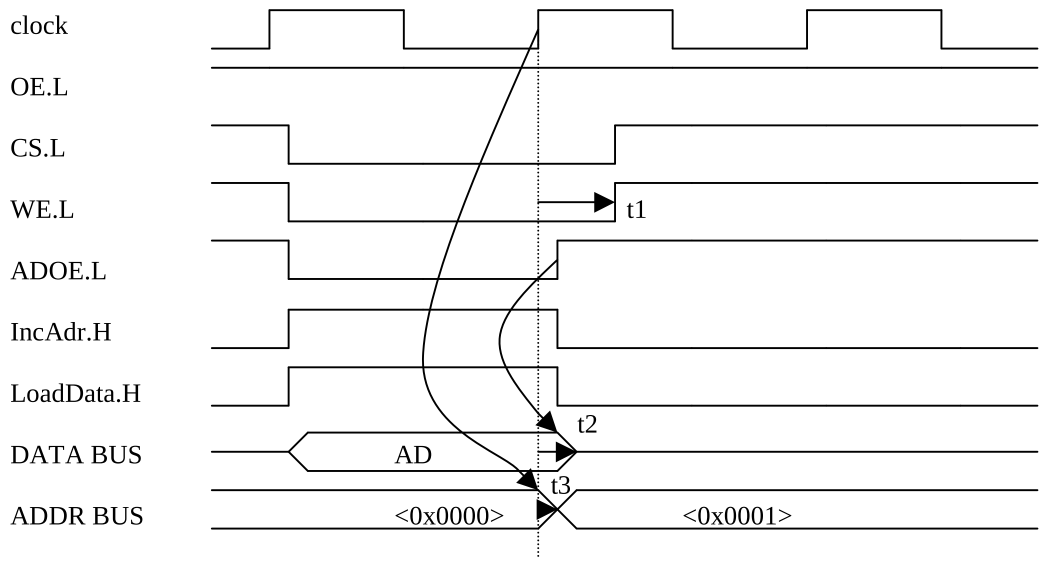
1: RAM Timing (review)

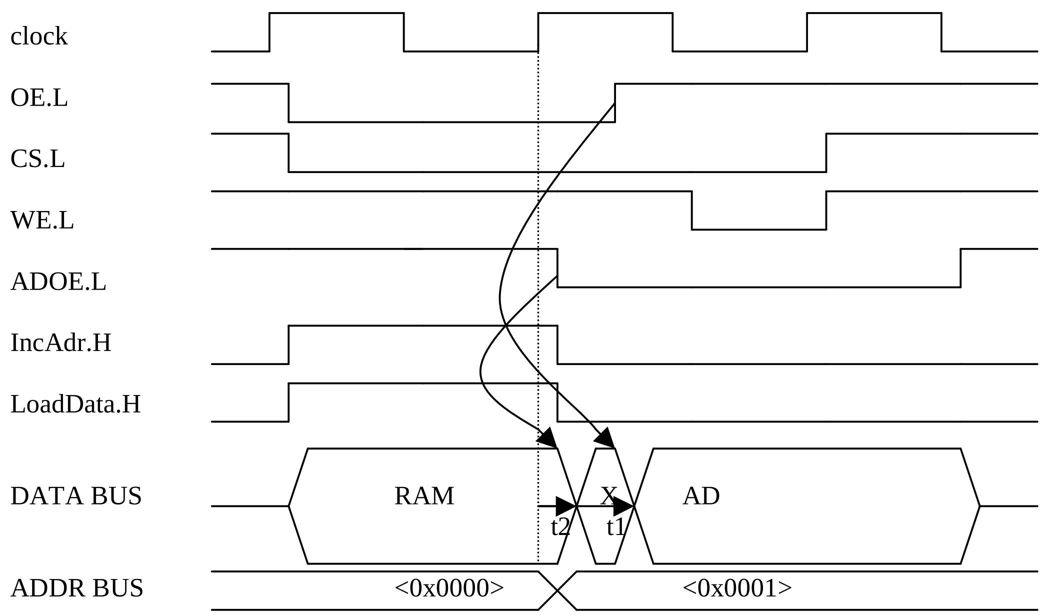
1)



t1: time for WE to be deasserted = tfsm + 20ns
t2: time for AD to disable outputs = tfsm + tADOEZ
t3: time for address to increment = tcount

t2 < t1: AD disables outputs before RAM stops writing (invalid data written to RAM).
t3 < t1: address changes before RAM stops writing (data written to wrong address).

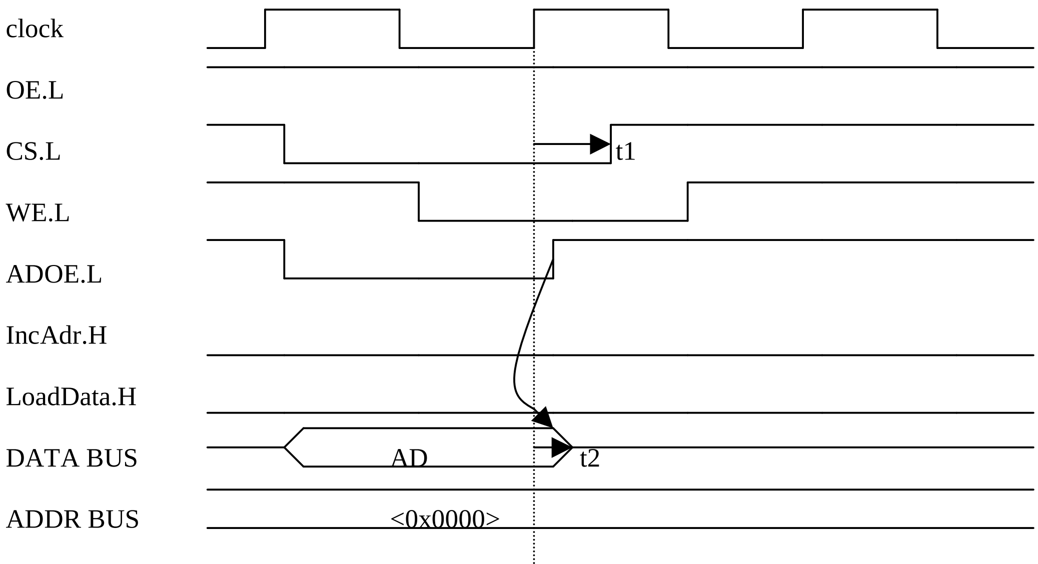
2)



t1: time for RAM to disable outputs = tfsm + 20ns + tOEZ
t2: time for AD to enable outputs = tfsm + tADOEA

t2 < t1: AD enables outputs before RAM disables outputs (bus contention).

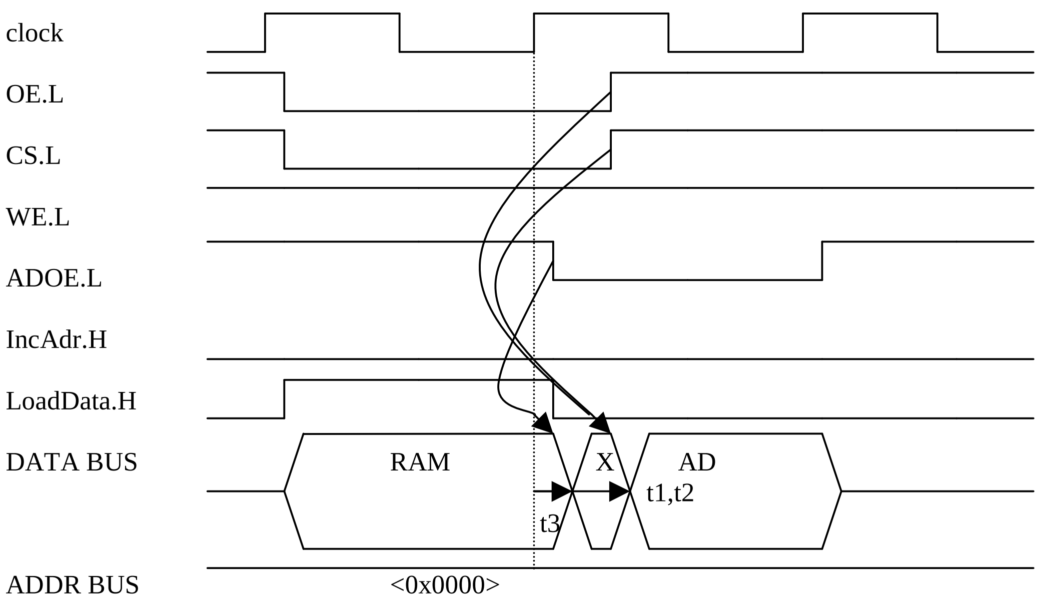
3)



t1: time to disable RAM to (and writing) = tfsm + 20ns
t2: time for AD to disable outputs = tfsm + tADOEZ

t2 < t1: AD disables outputs before RAM stops writing (invalid data written into RAM).

4)



t1: time for RAM to disable outputs = tfsm + 20ns + tOEZ
t2: time to disable RAM = tfsm + 20ns + tdis
t3: time for AD to enable outputs = tfsm + tADOEA

t3 < t1 & t2: AD enables outputs before ram disables outputs (bus contention).

2: K-Maps

- Truth Table

A2	A1	A0	F2	F1	F0
0	0	0	1	0	1
0	0	1	1	1	0
0	1	0	1	1	1
0	1	1	0	0	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	0	1	1
1	1	1	1	0	0

- Show all Karnaugh maps for S.O.P. and P.O.S.

*Sum of Products

A1 A2 A0

F2

A0\A2A1	00	01	11	10
0	1	1	0	0
1	1	0	1	0

F1

A0\A2A1	00	01	11	10
0	0	1	1	0
1	1	0	0	1

F0

A0\A2A1	00	01	11	10
0	1	1	1	1
1	0	0	0	0

*Products of Sums

F2

A0\A2A1	00	01	11	10
0	1	1	0	0
1	1	0	1	0

F1

A0\A2A1	00	01	11	10
0	0	1	1	0
1	1	0	0	1

F0

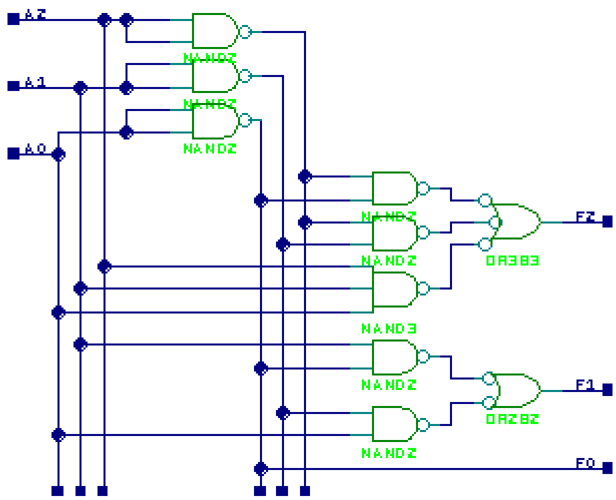
A0\A2A1	00	01	11	10
0	1	1	1	1
1	0	0	0	0

- Draw the logic diagram using NAND gates only for minimized S.O.P.

$$F2 = \overline{A2}\bullet\overline{A0} + \overline{A2}\bullet\overline{A1} + A2\bullet A1\bullet A0$$

$$F1 = A1\bullet\overline{A0} + \overline{A1}\bullet A0$$

$$F0 = \overline{A0}$$

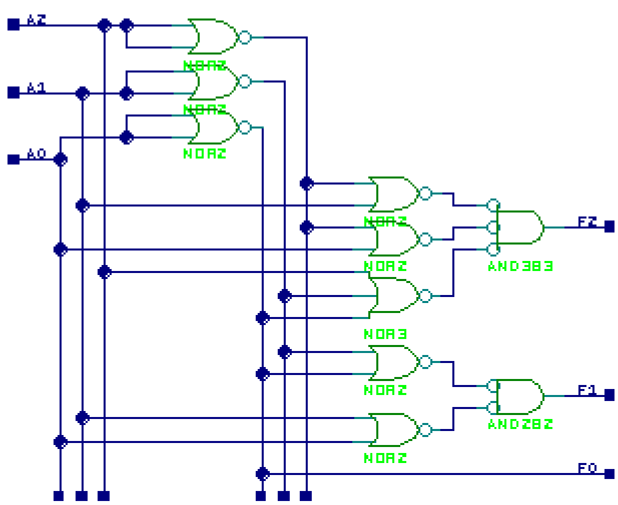


- Draw the logic diagram using NOR gates only for minimized P.O.S.

$$F2 = (\overline{A2} + A1)\bullet(\overline{A2} + A0)\bullet(A2 + \overline{A1} + \overline{A0})$$

$$F1 = (\overline{A1} + \overline{A0})\ (A1 + A0)$$

$$F0 = \overline{A0}$$



3: State Minimization

State Transition Table

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
S0	S1	S4	0	0
S1	S2	S1	0	0
S2	S1	S6	0	0
S3	S1	S3	0	0
S4	S5	S4	0	0
S5	S2	S1	0	0
S6	S5	S3	0	1

First Pass: Check Outputs

S1	S2-S1 S1-S4					
S2	S1-S1 S6-S4	S1-S2 S6-S1				
S3	S1-S1 S3-S4	S1-S2 S3-S1	S1-S1 S3-S6			
S4	S5-S1 S4-S4	S5-S2 S4-S1	S5-S1 S4-S6	S5-S1 S4-S3		
S5	S2-S1 S1-S4	S2-S2 S1-S1	S2-S1 S1-S6	S2-S1 S1-S3	S2-S5 S1-S4	
S6	X X	X X	X X	X X	X X	X X
	S0	S1	S2	S3	S4	S5

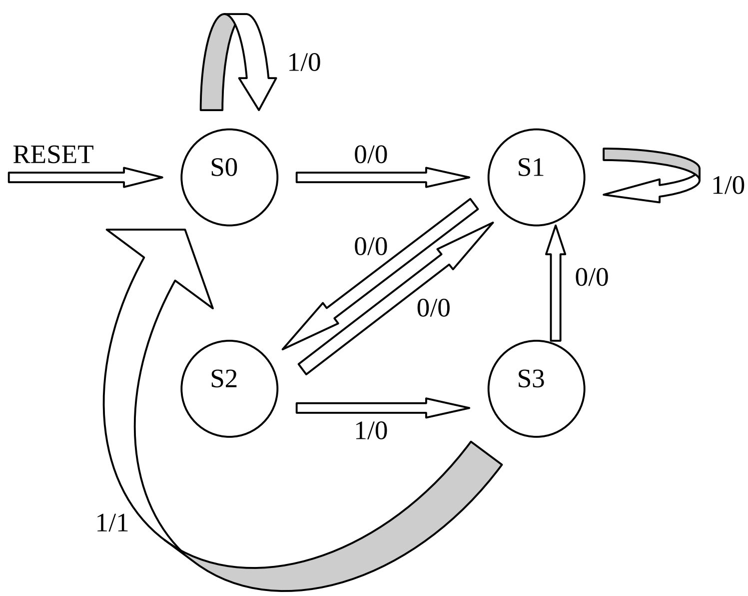
Second Pass: Eliminate S6's

S1	S2-S1 S1-S4					
S2	~~S1-S1~~ ~~S6-S4~~	~~S1-S2~~ ~~S6-S1~~				
S3	S1-S1 S3-S4	S1-S2 S3-S1	~~S1-S1~~ ~~S3-S6~~			
S4	S5-S1 S4-S4	S5-S2 S4-S1	~~S5-S1~~ ~~S4-S6~~	S5-S1 S4-S3		
S5	S2-S1 S1-S4	S2-S2 S1-S1	~~S2-S1~~ ~~S1-S6~~	S2-S1 S1-S3	S2-S5 S1-S4	
S6	X X	X X	X X	X X	X X	X X
	S0	S1	S2	S3	S4	S5

Third Pass: Eliminate S2’s and so on…

S1	~~S2-S1 S1-S4~~					
S2	~~S1-S1 S6-S4~~	~~S1-S2 S6-S1~~				
S3	S1-S1 S3-S4	~~S1-S2 S3-S1~~	~~S1-S1 S3-S6~~			
S4	S5-S1 S4-S4	~~S5-S2 S4-S1~~	~~S5-S1 S4-S6~~	S5-S1 S4-S3		
S5	~~S2-S1 S1-S4~~	S2-S2 S1-S1	~~S2-S1 S1-S6~~	~~S2-S1 S1-S3~~	~~S2-S5 S1-S4~~	
S6	X X	X X	X X	X X	X X	X X
	S0	S1	S2	S3	S4	S5

Therefore, the following are equivalent: S4 & S3 & S0, S5 & S1. By eliminating S3, S4, and S5, and replacing S6 with S3, the new reduced state diagram looks like this.



The recognizer will output a 1 when the input string is an even number of 0’s followed by two 1’s.