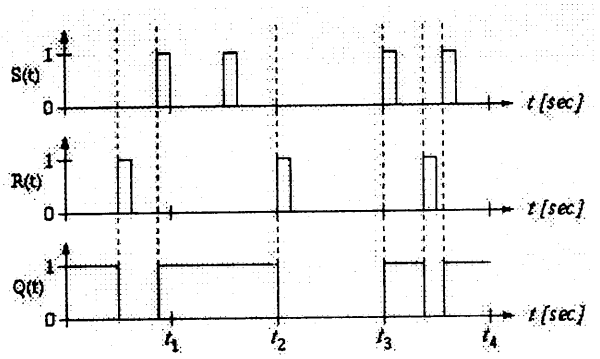
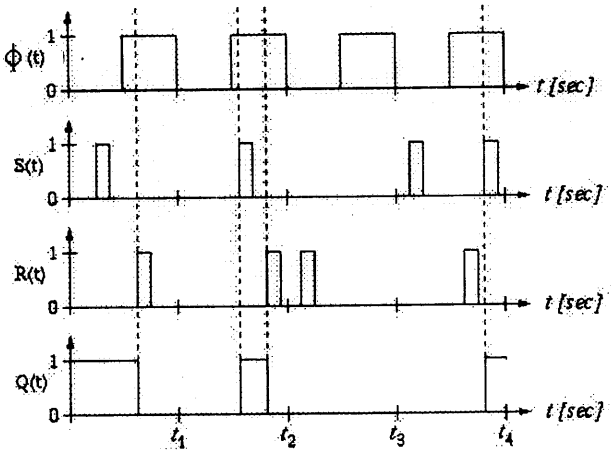
HOMEWORK SOLUTIONS #6 & 7

Chapter 9 Memory Elements and Arrays

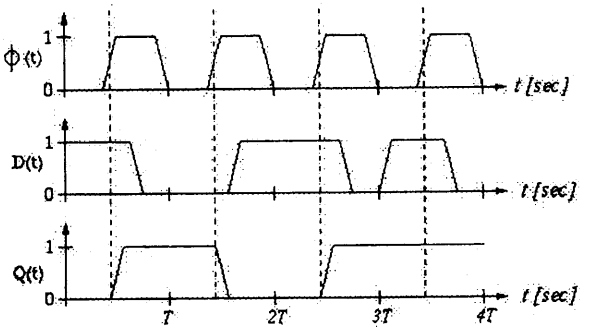
[9.1]



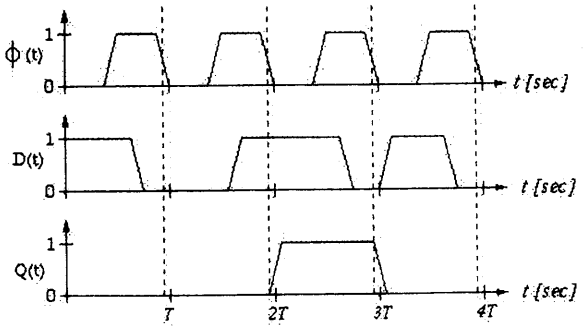
[9.2]



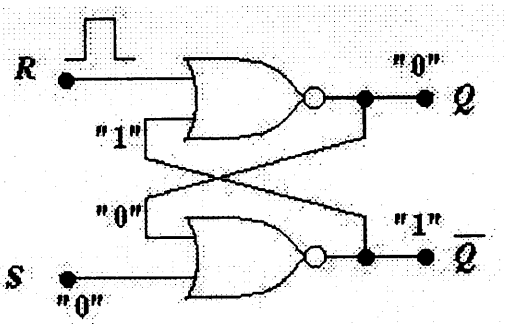
[9.3]



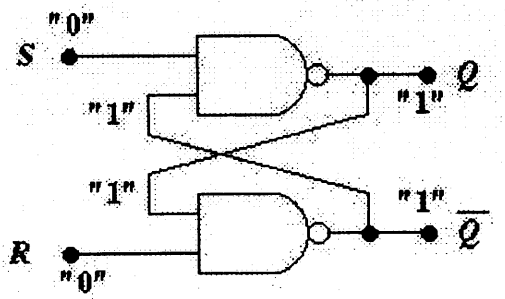
[9.4]



[9.5]



[9.6]



This case is "not used" because there is a conflict between Q and $\overline{Q}$, namely both of them cannot be "1" at the same time.

[9.7]

(a) 01011100 ⇒ SHR 2 ⇒ ***00***010111
(b) 01011100 ⇒ SHL 1 ⇒ 1011100***0***
(c) 01011100 ⇒ ROL 3 ⇒ 11100***010***
(d) 01011100 ⇒ ROR 2 ⇒ ***00***010111

[9.8] 86 = 01010110

(a) 01010110 ⇒ SHR 1 ⇒ ***0***0101011 ⇒ 43
(b) 01010110 ⇒ SHL 1 ⇒ 1010110***0*** ⇒ 172
(c) 01010110 ⇒ SHR 2 ⇒ ***00***010101 ⇒ 21
(d) 01010110 ⇒ ROR 2 ⇒ ***10***010101 ⇒ 149

[9.9] 0x4A35 ⇒ 0100101000110101 ⇒18,997

(a) 0100 1010 0011 0101 ⇒ SHR 6
⇒ ***0000 00***01 0010 1000 ⇒ 296
(b) 0100 1010 0011 0101 ⇒ SHR 3
⇒ ***000***0 1001 0100 0110 ⇒ 2,374
(c) 0100 1010 0011 0101 ⇒ SHL 8
⇒ 0011 0101 ***0000 0000*** ⇒ 13,568
(d) 0100 1010 0011 0101 ⇒ SHL 16
⇒ ***0000 0000 0000 0000*** ⇒ 0

[9.10]
From the question, every row has the same content which is 1010 1111 = 175

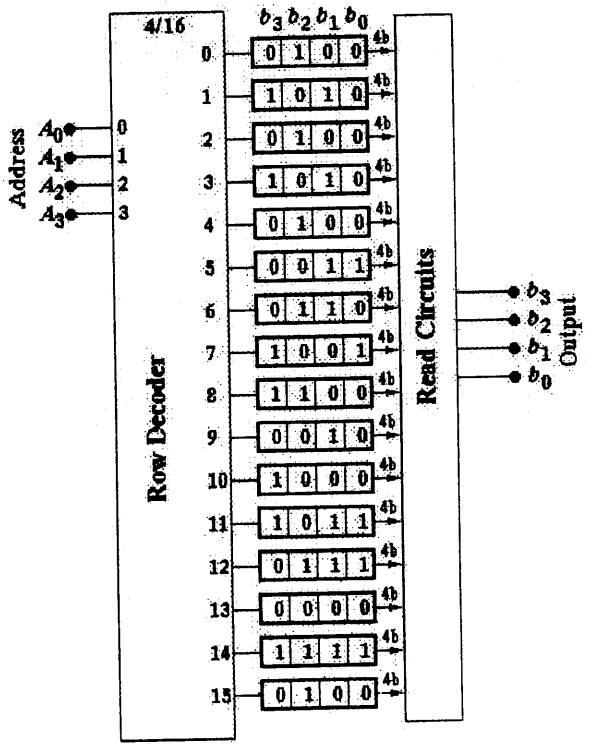
(a) $A_2A_1A_0 = 101 \Rightarrow$ Row 5: 175
(b) $A_2A_1A_0 = 011 \Rightarrow$ Row 3: 175
(c) $A_2A_1A_0 = 100 \Rightarrow$ Row 4: 175
(d) $A_2A_1A_0 = 110 \Rightarrow$ Row 6: 175

To make the question more interesting, let's consider the memory content in Figure 9.38 in the text:

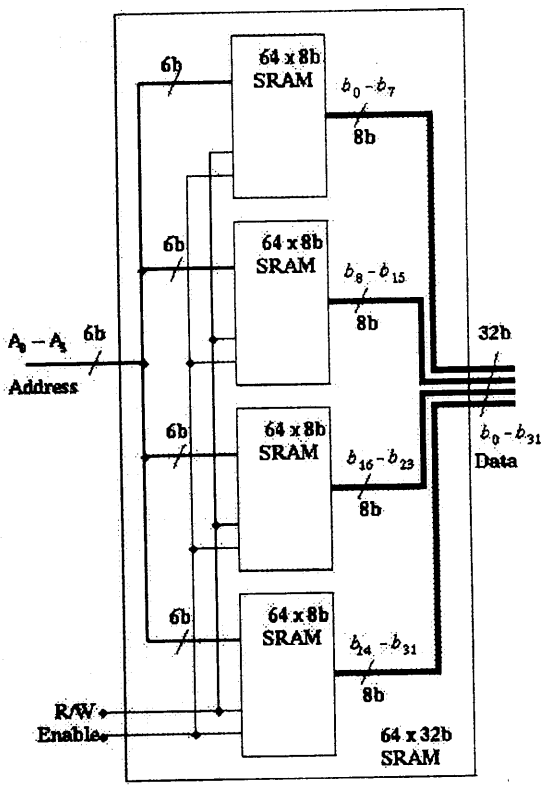
Row 0: 0110 0110 ⇒ 102
Row 1: 1111 0000 ⇒ 240
Row 2: 0000 1111 ⇒ 15
Row 3: 1010 1010 ⇒ 170
Row 4: 0101 0101 ⇒ 85
Row 5: 0000 1111 ⇒ 15
Row 6: 0011 1100 ⇒ 60
Row 7: 1100 0011 ⇒ 195

(a) $A_2A_1A_0 = 101 \Rightarrow$ Row 5: 15
(b) $A_2A_1A_0 = 011 \Rightarrow$ Row 3: 170
(c) $A_2A_1A_0 = 100 \Rightarrow$ Row 4: 85
(d) $A_2A_1A_0 = 110 \Rightarrow$ Row 6: 60

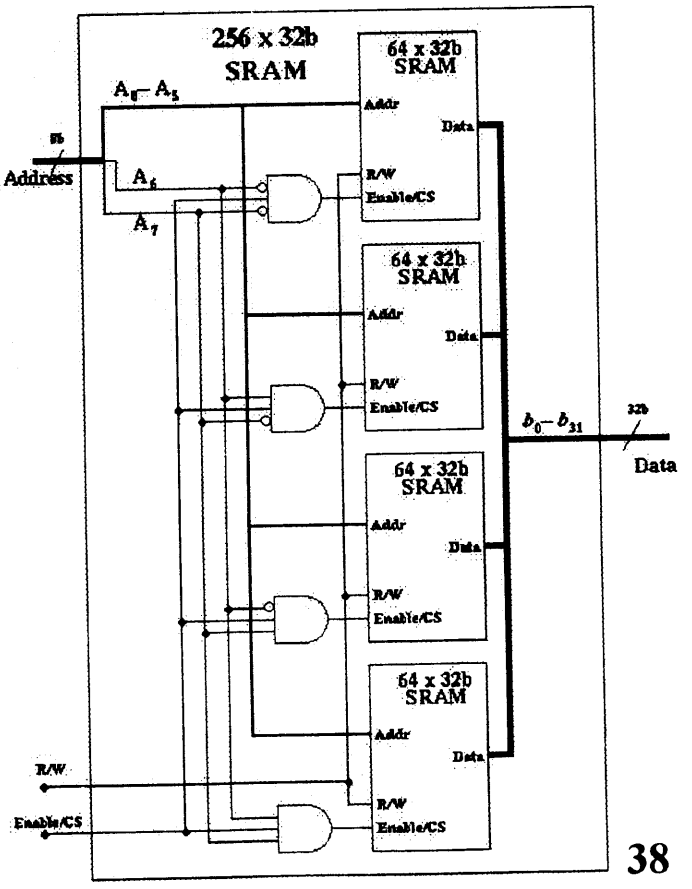
[9.11] The following diagram shows 16x4 SRAM:



[9.12]
Using a 64x8b SRAM as a basic block to construct a 64x32b SRAM as shown below:



[9.13]



Assume that we can construct a 64x32 SRAM from 64x8b SRAMs as shown in problem [9.12]. We then can use a 64x32b SRAM as a basic block diagram in this case to construct a 256x32b SRAM. Note that we can use the Enable signal as a Chip Select (CS) signal in the diagram above.

[9.14]
Four types of DRAM chips for a desktop personal computer are listed below:

5v EDO DIMMs (60 ns)

Size	Low	High	Avg.	Avg/MB
8MB	$17	$25	$21	$2.63
16MB	$25	$35	$31	$1.94
32MB	$53	$63	$58	$1.81
64MB	$112	$125	$116	$1.81
128MB	$215	$259	$237	$1.85

3.3v EDO DIMMs (60 ns)

Size	Low	High	Avg.	Avg/MB
16MB	$29	$35	$32	$2.00
32MB	$59	$90	$78	$2.44
64MB	$125	$169	$152	$2.38

3.3v SDRAM DIMMs

Size	Low	High	Avg.	Avg/MB
16MB	$22	$29	$25	$1.56
32MB	$35	$48	$38	$1.19
64MB	$59	$79	$64	$1.00
128MB	$109	$137	$118	$0.92

3.3v PC-100 SDRAM DIMMs

Size	Low	High	Avg.	Avg/MB
32MB	$35	$45	$37	$1.16
64MB	$49	$69	$60	$0.94
128MB	$99	$129	$112	$0.88

The price from SRAM chips can be obtained from the price of L2 cache which normally uses SRAM.

External L2 Cache DIMM

Size	Low	High	Avg.	Avg/MB
512KB	$59	$79	$70	$140
1MB	$115	$168	$132	$132

Note that these prices are the current prices obtained in May, 1999. As we show the price listed above, the average price per MB of SRAM chips is between 50-150 times that of DRAM chips in the retail market.

[9.15]
The major applications for SRAMs are in cache for personal computers, workstations, and embedded systems, and in main memory for supercomputers.

Another area of SRAM application is in communications and network applications. The major applications are in ATM switches and MPEG2 markets. Both uses small quantities of very fast SRAMs as buffers and storages.

[9.16]
(a) $p\ b_1b_2b_3 = 0\ 111$
$p \oplus b_1 \oplus b_2 \oplus b_3 = 1 \neq p$:Error occurs.

(b) $p\ b_1b_2b_3 = 1\ 101$
$p \oplus b_1 \oplus b_2 \oplus b_3 = 1 = p$: Meet parity check.

(c) $p\ b_1b_2b_3 = 0\ 100$
$p \oplus b_1 \oplus b_2 \oplus b_3 = 1 \neq p$: Error occurs.

(d) $p\ b_1b_2b_3 = 1\ 101$
$p \oplus b_1 \oplus b_2 \oplus b_3 = 1 = p$: Meet parity check.

(e) $p\ b_1b_2b_3 = 0\ 000$
$p \oplus b_1 \oplus b_2 \oplus b_3 = 0 = p$: Meet parity check.

[9.17]
Compact disks use a technique called **eight-to-fourteen modulation** (EFM) where 8b data word is expanded to a 14b modulated equivalent such that the modulated word does not violate the spacing rule between 1s. Thus, a 16b data word would result in 28b on the surface of a compact disk.

1024 16b data words $\Rightarrow$ 28,672 bits.

[9.18]
(a) Total user bytes = 333,000×2048
= 681,984,000 bytes

(b) Percentage overhead $= \dfrac{(2352 - 2048)}{2352}$
$= 13\%$

(c) 60 minutes of play time would result in fewer number of blocks:

$(\frac{60}{74})333{,}000 = 270{,}000$ blocks

Total user bytes = 270,000×2048
=552,960,000 bytes

[9.19]
BER = 10^{-10} would result in 0.01 bit/sec or 36 bits/hr for the data rate of 100 Mb/sec. The way to combat these error bits is to employ some sort of error detection and correction algorithm. This can reduce the error bit to virtually zero in practice.

[9.20]

$$t_H = \frac{C_s}{I_L}(V_1 - V_{\min}) = \frac{60\,fF}{0.2\,pA}(3V - 1.2V)$$
$$= 540\ ms.$$

[9.21]

(a) A DRAM cell can hold a logic 0 state forever because the leakage current will only have an effect on a logic 1 state where it will pull down the voltage on C_s to zero.

(b) $$t_H = \frac{C_s}{I_L}(V_1 - V_{\min}) = \frac{55\,fF}{0.14\,pA}(3V - 1V)$$
$$= 786\ ms.$$

[9.22]
The soft error in a DRAM can be quite a serious concern in the space radiation environment where the radiating particles in this environment consist primarily of high-energy electrons, proton, alpha particles, and cosmic rays.

Chapter 10 Sequential Logic Networks

[10.1]

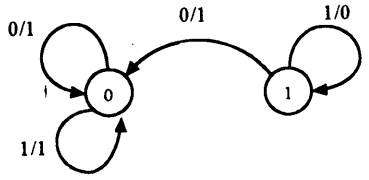
(a) $f(t) = \overline{p(t) \bullet A(t)}$

$A(t+1) = \overline{f(t)} = p(t) \bullet A(t)$

(b)

p(t)	A(t)	f(t)	A(t+1)
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

(c)

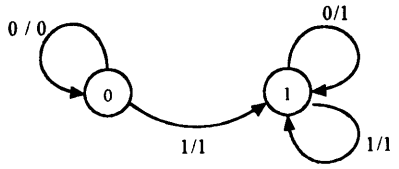


[10.2]

(a)

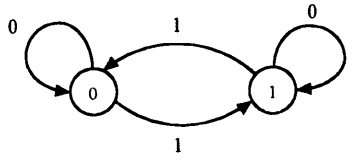
c(t)	X(t)	T(t)	X(t+1)
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1

(b)



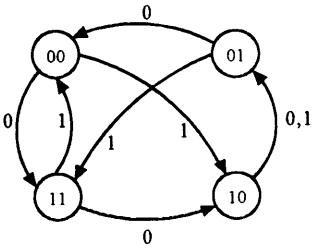
[10.3]

A(t)	x(t)	A(t+1)
0	0	0
0	1	1
1	0	1
1	1	0



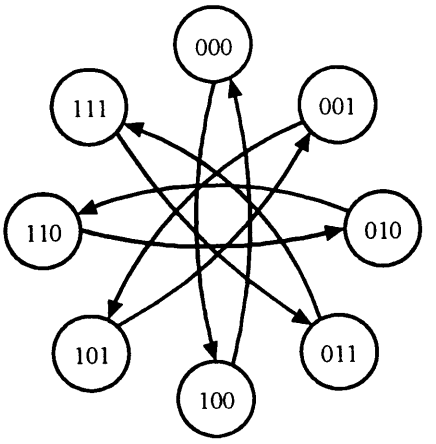
[10.4]

x(t)	A(t)	B(t)	A(t+1)	B(t+1)
0	0	1	0	0
0	0	0	1	1
0	1	1	1	0
0	1	0	0	1
1	0	1	1	1
1	0	0	1	0
1	1	1	0	1
1	1	0	0	0



[10.5]

Present			Next		
a_2	a_1	a_0	a_2	a_1	a_0
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	0	0	0
1	0	1	0	0	1
1	1	0	0	1	0
1	1	1	0	1	1



Note that the state diagram shows that this state machine will only toggle between two states depending on the initial state.

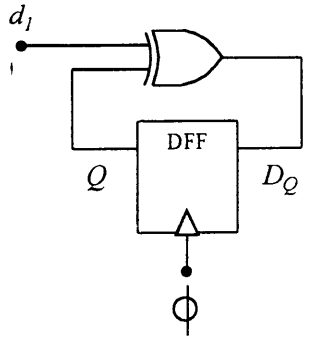
[10.6]

(a)

Present			Next
$Q(t)$	d_1	d_0	$Q(t+1)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

(b)

$Q(t+1) = d_1 \oplus Q(t)$



Note that $Q(t+1)$ does not at all depend on d_2.

[10.7]

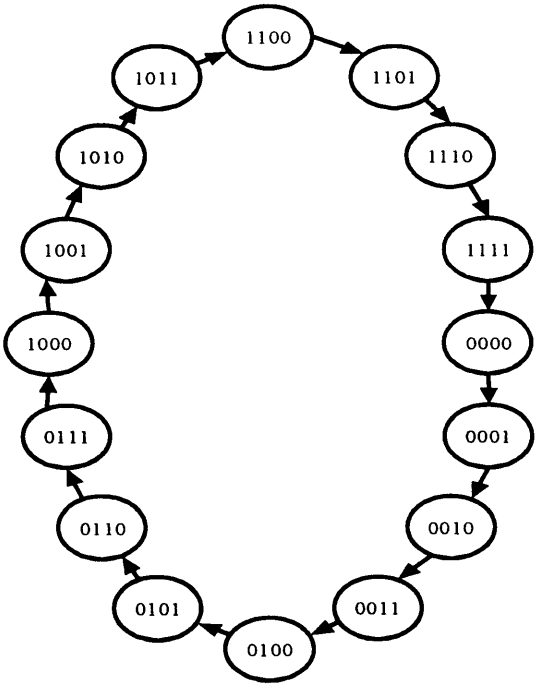
Present			Next	
S_1	S_0	x	S_1	S_0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	1
0	1	1	1	1
1	0	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	1	0

[10.8]

Present			Next	
A_1	A_0	X	A_1	A_0
0	0	0	1	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	0

[10.9]

(a)



(b)

A_3	A_2	A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

(c)

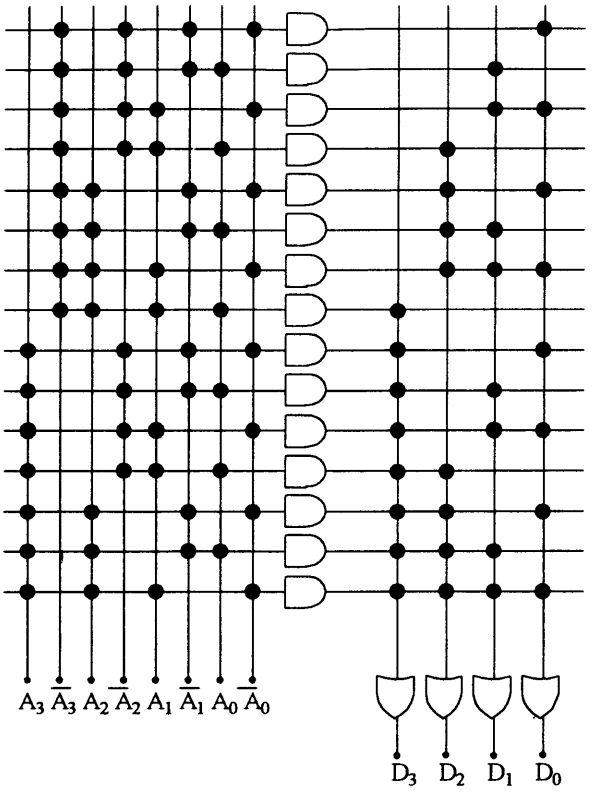
$D_0 = \overline{A_0}$

$D_1 = A_1 \oplus A_0$

$D_2 = A_2 \oplus (A_1 \bullet A_0)$

$D_3 = A_3 \oplus (A_2 \bullet A_1 \bullet A_0)$

(d) Using the simplified wiring notation,

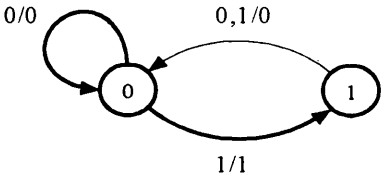


[10.10]

(a)

A(t)	*x(t)*	*F(t)*	*x(t+1)*
0	0	0	0
0	1	0	0
1	0	1	1
1	1	0	0

(b)

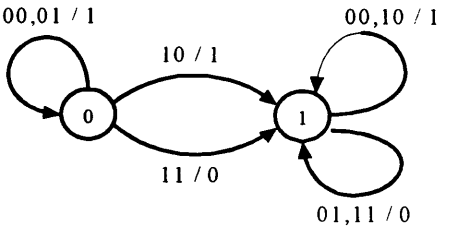


[10.11]

(a)

A(t)	*B(t)*	*x(t)*	*R(t)*	*x(t+1)*
0	0	0	1	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	1
1	1	0	0	1
1	1	1	0	1

(b)

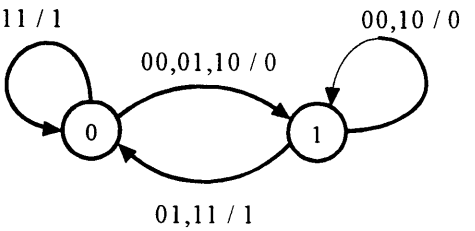


[10.12]

(a)

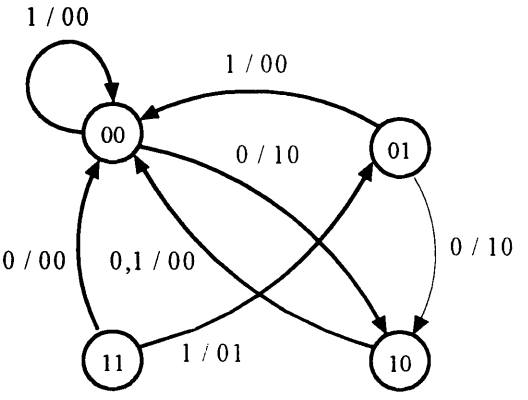
a(t)	*b(t)*	*x(t)*	*f(t)*	*x(t+1)*
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	1	0
1	1	1	1	0

(b)



[10.13]

w(t)	*x(t)*	*y(t)*	$f_x(t)$	$f_y(t)$	*x(t+1)*	*y(t+1)*
0	0	0	1	0	1	0
0	0	1	1	0	1	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	0	1	0	1



[10.14] Let x and y be the state variables.

(a)

$d(t)$	$x(t)$	$y(t)$	$F(t)$	$x(t+1)$	$y(t+1)$
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
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	1	1

(b)

The sequence of bits that the network detects is 1111. We will know that the circuit detects that sequence by observing the output F. The output F will be 1 when the circuit has detected a bit sequence 1111 and it will stay at 1 until there is 0 appearing in between. The circuit then has to start detecting a bit sequence 1111 again.

(c)