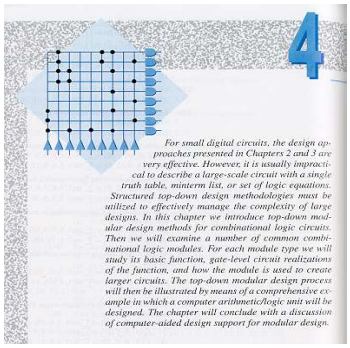
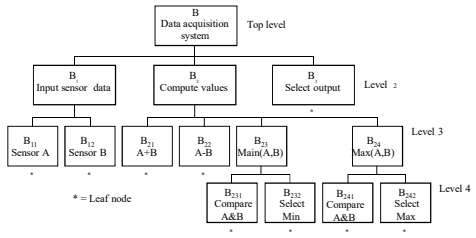
Chapter 4 -- Modular Combinational Logic



For small digital circuits, the design approaches presented in Chapters 2 and 3 are very effective. However, it is usually impractical to describe a large-scale circuit with a single truth table, minterm list, or set of logic equations. Structured top-down design methodologies must be utilized to effectively manage the complexity of large designs. In this chapter we introduce top-down modular design methods for combinational logic circuits. Then we will examine a number of common combinational logic modules. For each module type we will study its basic function, gate-level circuit realizations of the function, and how the module is used to create larger circuits. The top-down modular design process will then be illustrated by means of a comprehensive example in which a computer arithmetic/logic unit will be designed. The chapter will conclude with a discussion of computer-aided design support for modular design.

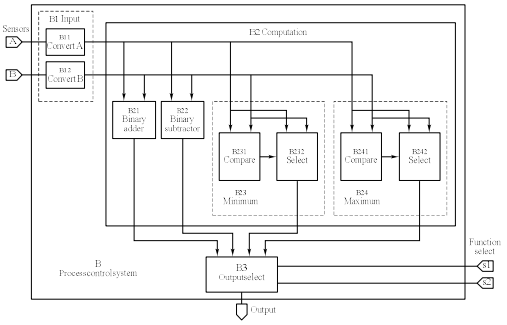
1

Top-down design of a function.



Hierarchical design tree.

2



Hierarchical block diagram

3

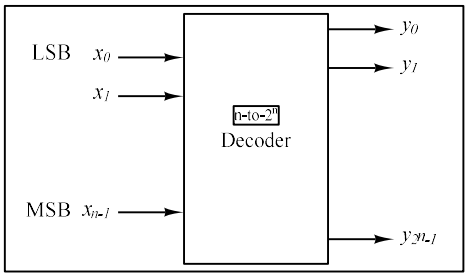
Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

4

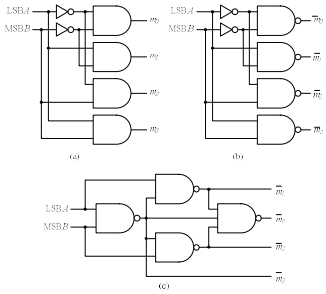
Decoders

$n - to - 2^n$ **decoder module**

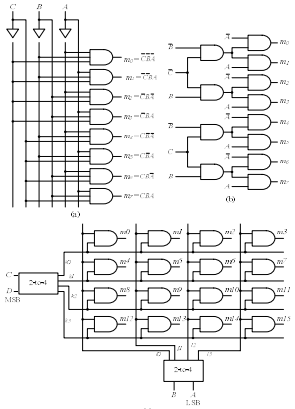


ASCII, BCD, Gray, Excess-3

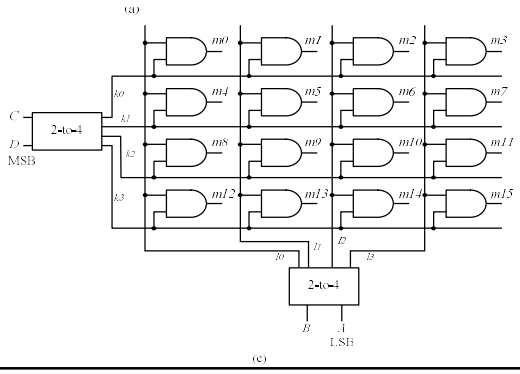
Decoder Realization



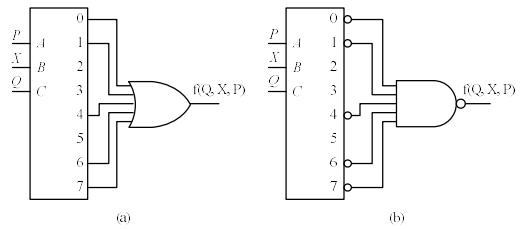
More complex decoders



More complex decoders

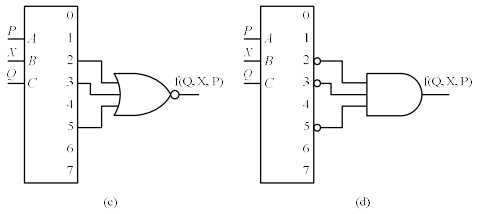


Example 4.1 -- Realize
$f(Q,X,P) = \sum m(0,1,4,6,7) = \prod M(2,3,5)$



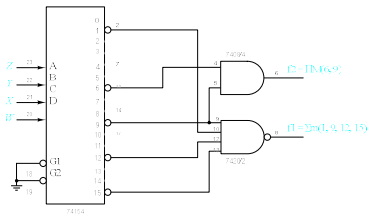
9

Example 4.1 (concluded)



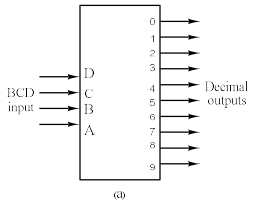
10

Realization of switching functions with a decoder



11

BCD to decimal decoder.



BCD code DCBA	Decimal digits
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

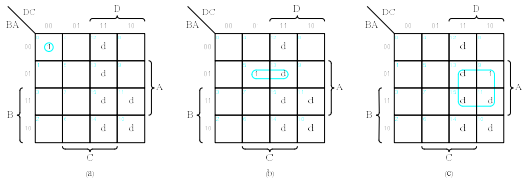
(b)

(a) Logic symbol.

(b) BCD codes and decimal digits

12

K-maps for outputs 0, 5 and 9 of a BCD to decimal code converter

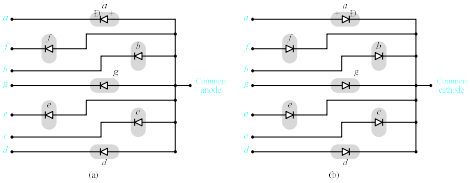


(a)Decimal 0 = $\overline{D}\overline{C}\overline{B}\overline{A}$

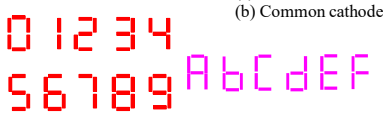
(b) Decimal 5 = $C\overline{B}A$

(c) Decimal 9 = DA

7-Segment display elements

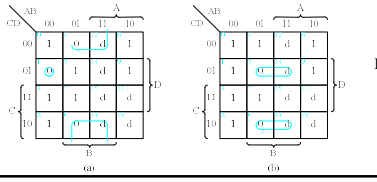


(a) Common anode
(b) Common cathode



Decimal digits displayed on 7-segment display element

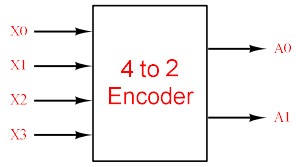
Decimal Digit	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
5	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1



K-maps for a BCD to 7-segment code converter segment a,b.

Encoders

$2^s \geq n$ หรือ $s \geq \log_2 n$

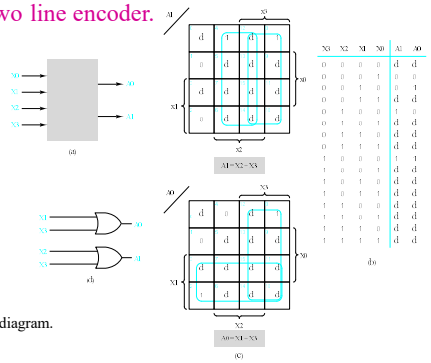


(b) Truth table

X3	X2	X1	X0	A1	A0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

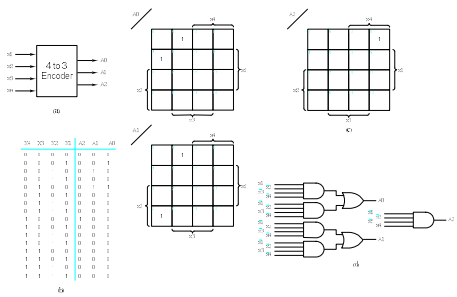
(b)

Four-to-two line encoder.



(a) Functional diagram.
(b) Truth table
(c) K-maps
(d) Logic diagram

Four-to-three line encoder



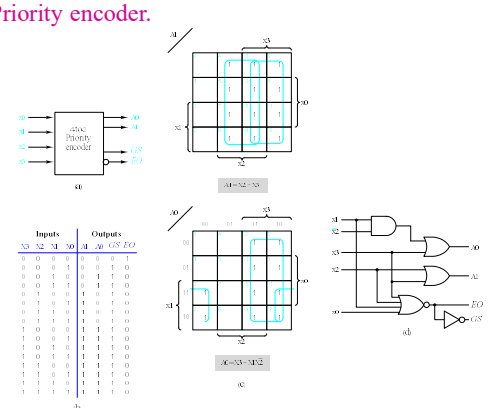
(a) Functional diagram (b) Truth table
(c) K-maps (d) Logic diagram

Truth table

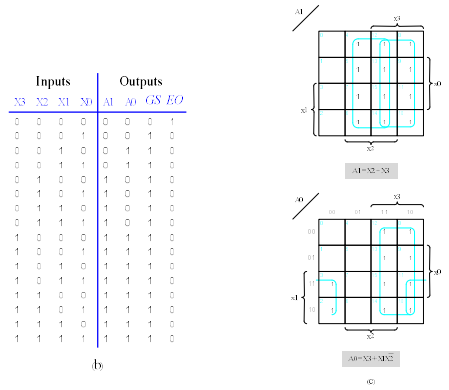
X4	X3	X2	X1	A2	A1	A0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	0	0
0	1	0	0	0	1	1
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	0	0	0
1	0	0	0	1	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	0	0	0
1	1	0	0	0	0	0
1	1	0	1	0	0	0
1	1	1	0	0	0	0
1	1	1	1	0	0	0

(b)

Priority encoder.

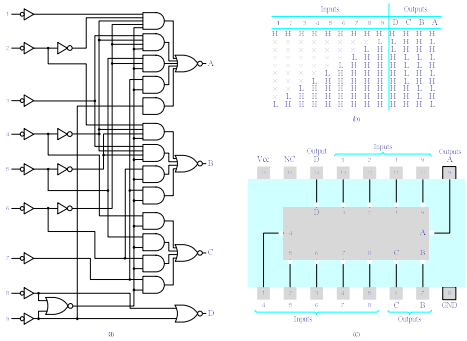


21



22

74147 priority encoder module



(a) Logic diagram (b) Function table
(c) Package configuration

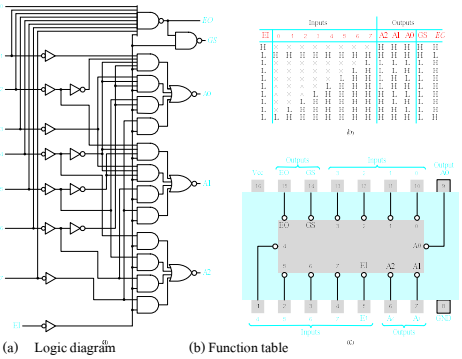
23

74147 priority encoder module function table

Inputs									Outputs			
1	2	3	4	5	6	7	8	9	D	C	B	A
H	H	H	H	H	H	H	H	H	H	H	H	H
×	×	×	×	×	×	×	×	L	L	H	H	L
×	×	×	×	×	×	×	L	H	L	H	H	H
×	×	×	×	×	×	L	H	H	H	L	L	L
×	×	×	×	×	L	H	H	H	H	L	L	H
×	×	×	×	L	H	H	H	H	H	L	H	L
×	×	×	L	H	H	H	H	H	H	L	H	H
×	×	L	H	H	H	H	H	H	H	H	L	L
×	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

24

74148 priority encoder module



(a) Logic diagram (b) Function table

(c) Package configuration

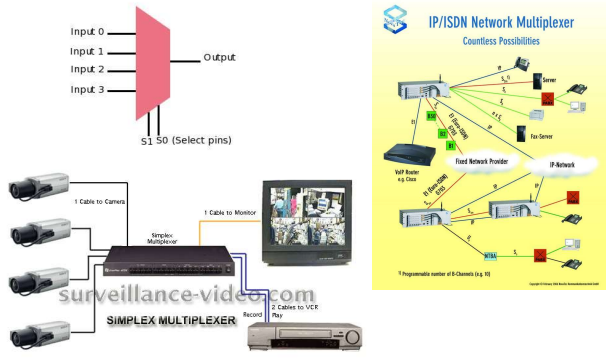
Source: The TTL Data Book Volume 2, Texas Instruments Inc., 1985

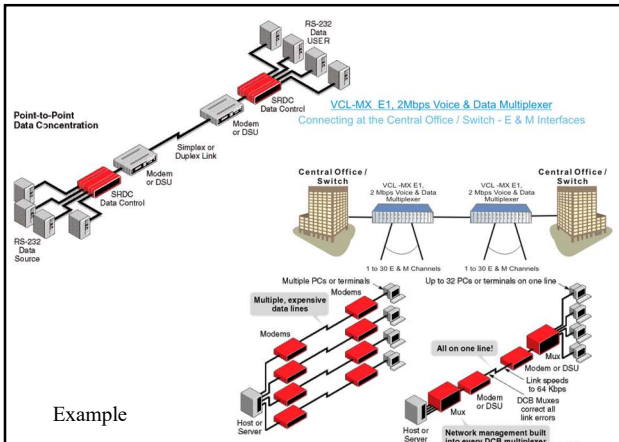
74148 priority encoder module

Function table

	Inputs								Outputs				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	*EO*
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

K-Channel multiplexing/demultiplexing





Example

K-Channel multiplexing/demultiplexing

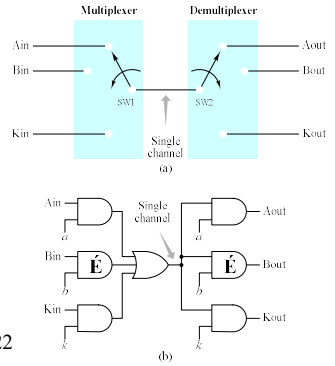
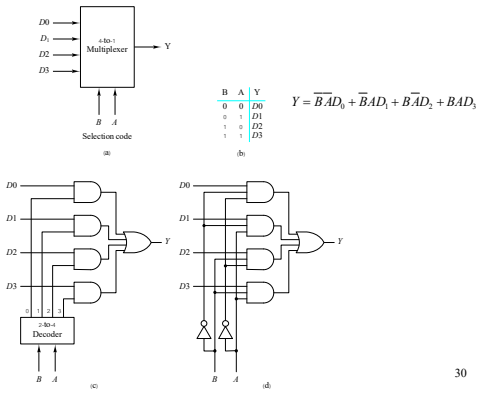


Figure 4.22

Four-to-one multiplexer design

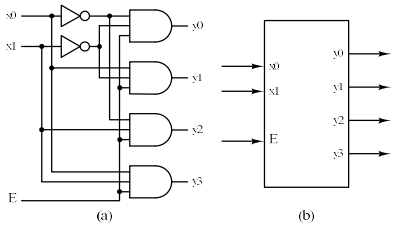


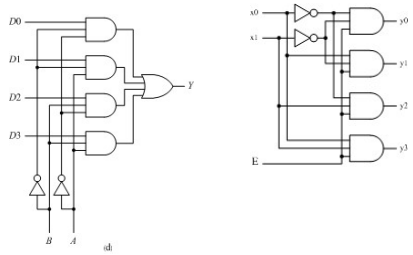
X4	X3	X2	X1	A2	A1	A0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	0	0
0	1	0	0	0	1	1
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	0	0	0
1	0	0	0	1	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	0	0	0
1	1	0	0	0	0	0
1	1	0	1	0	0	0
1	1	1	0	0	0	0
1	1	1	1	0	0	0

(b)

one-to-four demultiplexer design

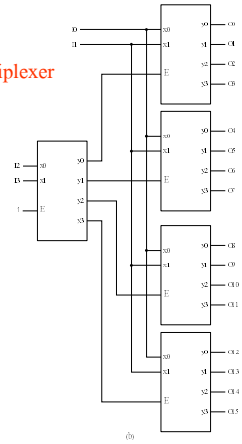
(a) Functional diagram
(b) Truth table



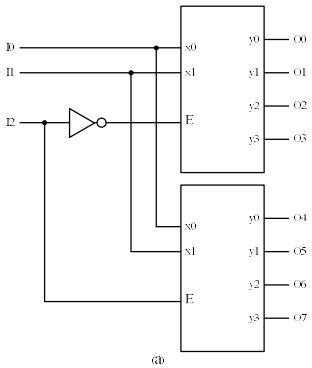


33

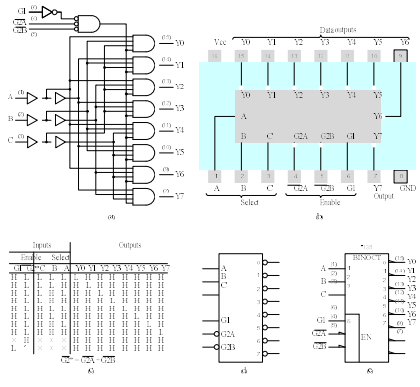
1-to-16 Demultiplexer



3-to-8 Decoder



35



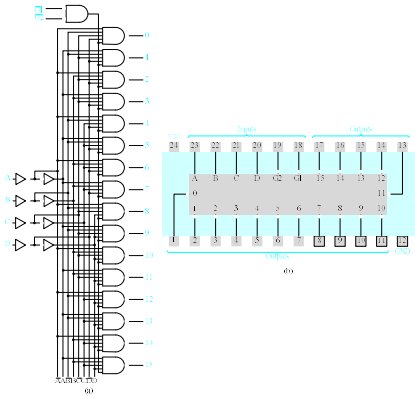
36

Inputs					Outputs							
Enable			Select									
G1	G2*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L
×	H	×	×	×	H	H	H	H	H	H	H	H
L	´	×	×	×	H	H	H	H	H	H	H	H

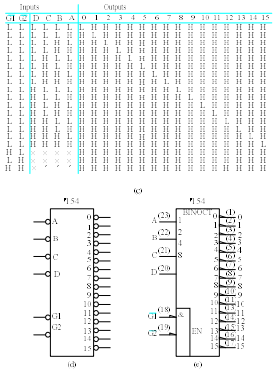
$\overline{G2^*} = \overline{G2A} + \overline{G2B}$

(c)

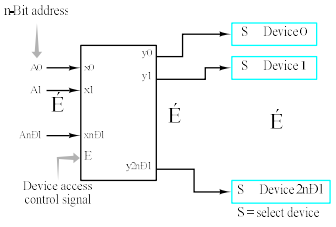
37



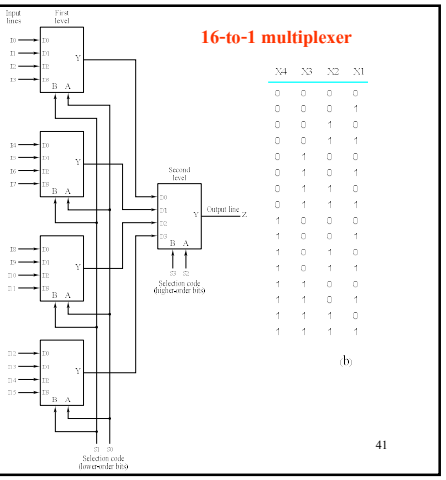
38



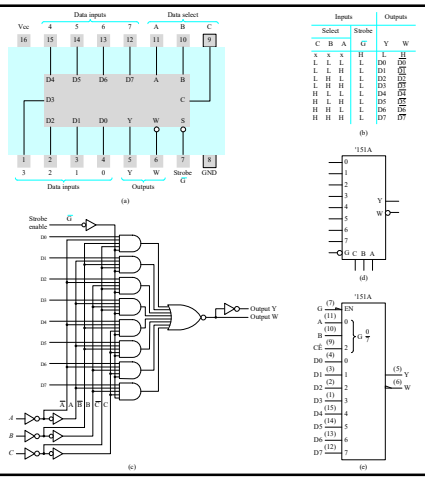
39



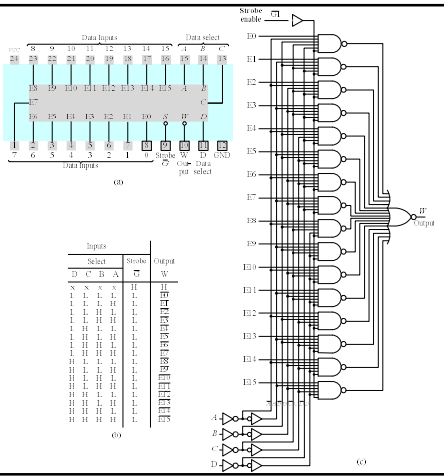
40

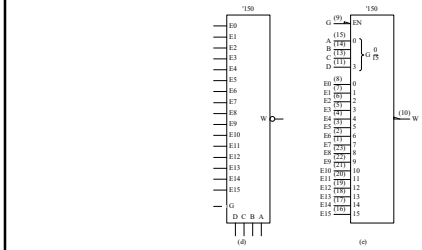

16-to-1 multiplexer
Input lines
First level
Second level
Output line
Selection code
(higher-order bits)
Selection code
(lower-order bits)
(b)

41

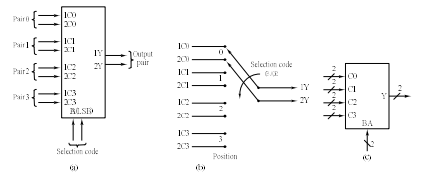

Data inputs
Data select
Outputs
Strobe
(a)
(b)
'151A
(d)
Output Y
Output W
(c)
(e)

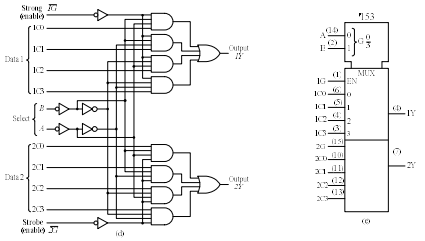
42


Strobe enable
Data Inputs
Data select
Output
(a)
(b)
(c)


'150
(d)
(e)

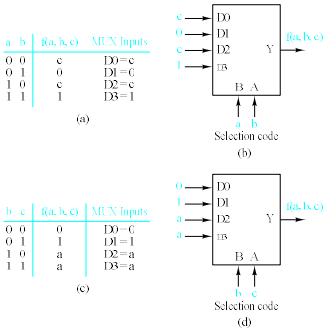
44


Pair 0
Pair 1
Pair 2
Pair 3
Output pair
MSB
LSB
Selection code
(a)
Selection code
Position
(b)
(c)

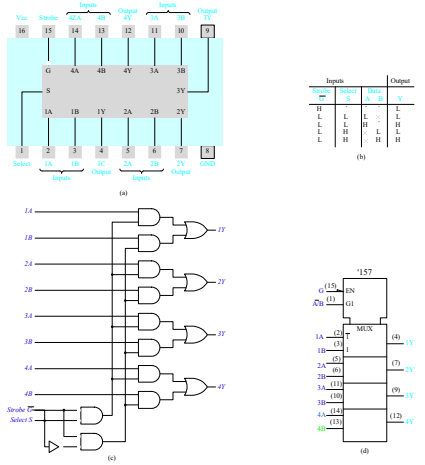

Strobe
Data 1
Data 2
Select
Output
Strobe
(d)

'153
MUX
(e)

45

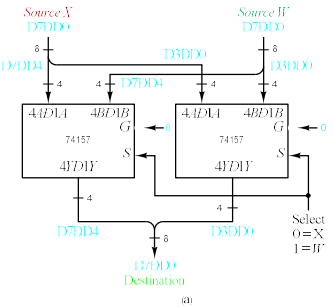

a b f(a, b, c) MUX Inputs
0 0 c D0 = c
0 1 0 D1 = 0
1 0 c D2 = c
1 1 1 D3 = 1
(a)
Selection code
(b)
b c f(a, b, c) MUX Inputs
0 0 0 D0 = 0
0 1 1 D1 = 1
1 0 a D2 = a
1 1 a D3 = a
(c)
Selection code
(d)

46

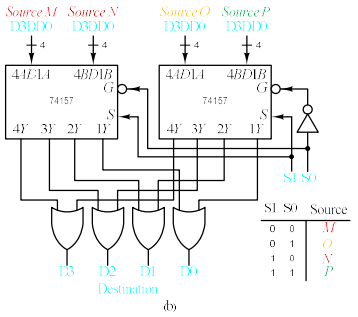

Vcc
Strobe
Inputs
Output
Select
Data
Output
(a)
(b)
(c)

'157
MUX
(d)

47


Source X
D7DD0
Source W
D7DD0
D3DD0
D7DD4
D3DD0
4ADLA
4BDLB
G
74157
S
4YDY
Select
0 = X
1 = W
D7DD4
D3DD0
D7DD0
Destination
(a)

48



49

Use a 74151A multiplexer to Realize
$f(x_1,x_2,x_3) = \sum m(0,2,3,5)$

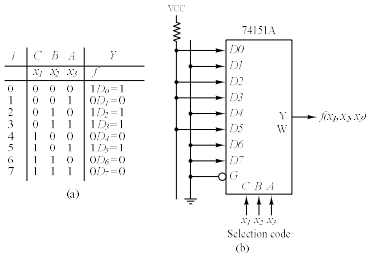


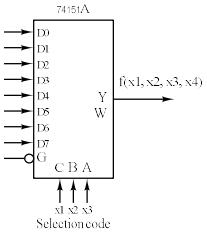
Figure 4.30

50

Use a 74151A multiplexer to Realize
$f(x_1,x_2,x_3,x_4) = \sum m(0,1,2,3,4,9,13,14,15)$

i	C X1	B X2	A X3	X4	f
0	0	0	0	0	1
	0	0	0	1	1
1	0	0	1	0	1
	0	0	1	1	1
2	0	1	0	0	1
	0	1	0	1	0
3	0	1	1	0	0
	0	1	1	1	0
4	1	0	0	0	0
	1	0	0	1	1
5	1	0	1	0	0
	1	0	1	1	0
6	1	1	0	0	0
	1	1	0	1	1
7	1	1	1	0	1
	1	1	1	1	1

(a)

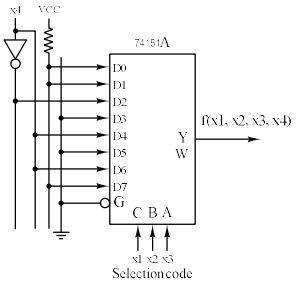


51

Use a 74151A multiplexer to Realize
$f(x_1,x_2,x_3,x_4) = \sum m(0,1,2,3,4,9,13,14,15)$

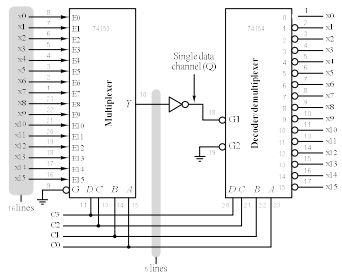
i	C X1	B X2	A X3	X4	f	f	Y
0	0	0	0	0	1		
	0	0	0	1	1	1	D0 = 1
1	0	0	1	0	1		
	0	0	1	1	1	1	D1 = 1
2	0	1	0	0	1		
	0	1	0	1	0	$\bar{X4}$	D2 = $\bar{X4}$
3	0	1	1	0	0		
	0	1	1	1	0	0	D3 = 0
4	1	0	0	0	0		
	1	0	0	1	1	X4	D4 = X4
5	1	0	1	0	0		
	1	0	1	1	0	0	D5 = 0
6	1	1	0	0	0		
	1	1	0	1	1	X4	D6 = X4
7	1	1	1	0	1		
	1	1	1	1	1	1	D7 = 1

(a)

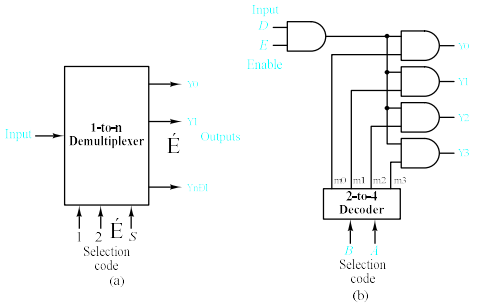


(b)

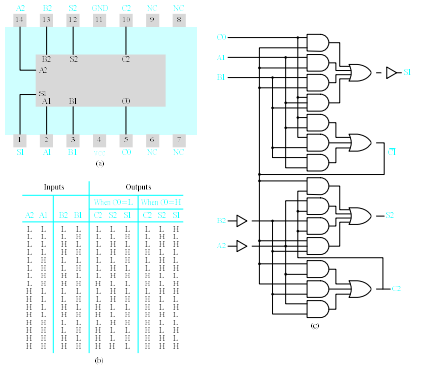
52


Multiplexer
Decoder demultiplexer
Single data channel (Q)
4 lines
4 lines

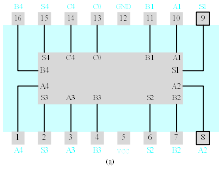
53


1-to-n Demultiplexer
Input
Outputs
Selection code
(a)
Input
Enable
2-to-4 Decoder
Selection code
(b)

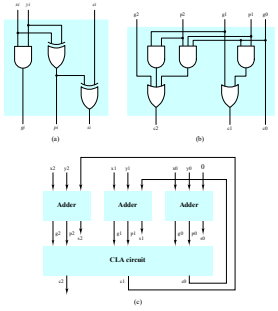
54



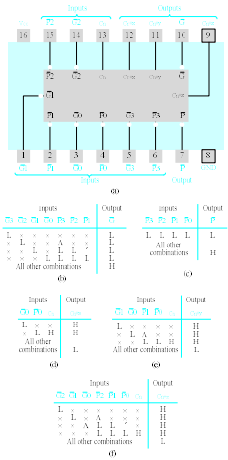
55



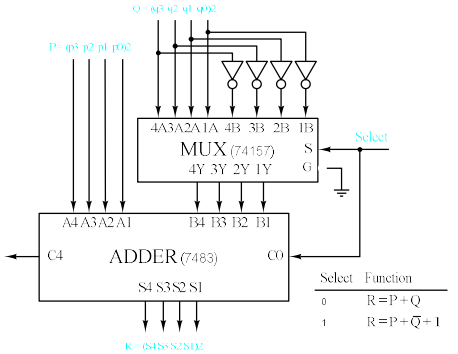
56


Adder
Adder
Adder
CLA circuit

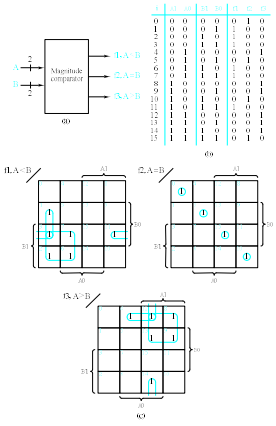
57



58


MUX (74157)
ADDER (7483)
Select
Select Function
0 R = P + Q
1 R = P + Q̄ + 1

59

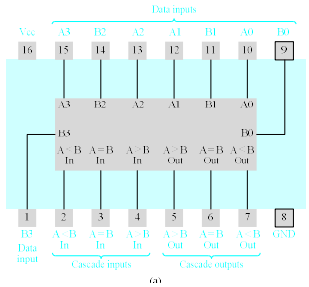

Magnitude comparator

60

Comparing inputs				Cascading inputs			Outputs		
A3, B3	*A2, B2*	*A1, B1*	*A0, B0*	*A > B*	*A < B*	*A = B*	*A > B*	*A < B*	*A = B*
A3 > B3	×	×	×	×	×	×	H	L	L
A3 < B3	×	×	×	×	×	×	L	H	L
A3 = B3	*A2 > B2*	×	×	×	×	×	H	L	L
A3 = B3	*A2 < B2*	×	×	×	×	×	L	H	L
A3 = B3	*A2 = B2*	*A1 > B1*	×	×	×	×	H	L	L
A3 = B3	*A2 = B2*	*A1 < B1*	×	×	×	×	L	H	L
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 > B0*	×	×	×	H	L	L
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 < B0*	×	×	×	L	H	L
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 = B0*	H	L	L	H	L	L
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 = B0*	L	H	L	L	H	L
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 = B0*	L	L	H	L	L	H
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 = B0*	×	×	H	L	L	H
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 = B0*	H	H	L	L	L	L
A3 = B3	*A2 = B2*	*A1 = B1*	*A0 = B0*	L	L	L	H	H	L

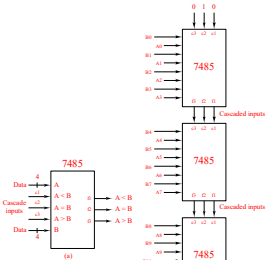
(b)

61



(a)

62



(a)

(b)

63

Half Adders

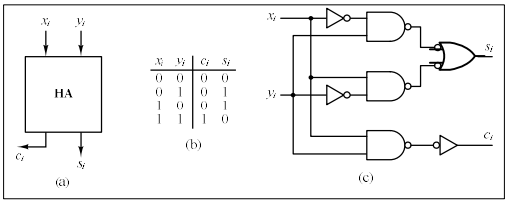


Figure 4.35 (a) -- (c)

64

Full Adders

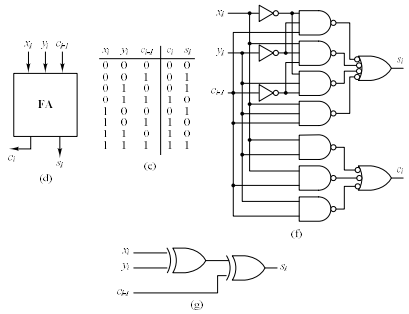


Figure 4.35 (d) -- (g)

Ripple Carry Adder

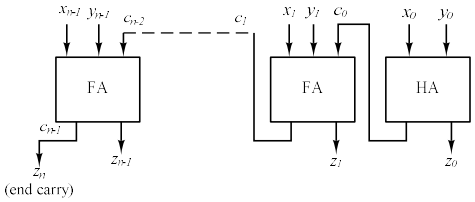


Figure 4.36

Addition Time for a Basic Ripple-Carry Adder

Let t_{gate} = the propogation delay through a typical logic gate

Half adder propagation delays

$t_{add} = 3\ t_{gate}$

$t_{carry} = 2\ t_{gate}$

Full adder propagation delays

$t_{add} = 3\ t_{gate}$

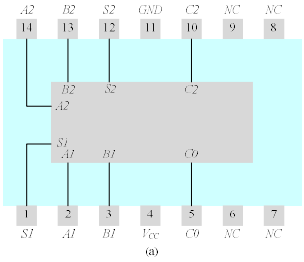
$t_{carry} = 2\ t_{gate}$

Ripple-Carry Adder (n-bits)

$t_{add} = (n - 1)2\ t_{gate} + 3\ t_{gate}$

$= (2n + 1)\ t_{gate}$

SN7482 Two-Bit Pseudo Parallel Adder Module



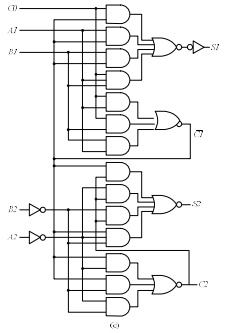
Package Pin Configuration

SN7482 Pseudo Parallel Adder -- Truth Table

Inputs				Outputs					
				When $C0$ = L			When $C0$ = H		
$A2$	$A1$	$B2$	$B1$	$C2$	$S2$	$S1$	$C2$	$S2$	$S1$
L	L	L	L	L	L	L	L	H	H
L	L	L	H	L	L	H	L	H	L
L	L	H	L	L	H	L	L	H	H
L	L	H	H	L	H	H	H	L	L
L	H	L	L	L	L	H	L	H	L
L	H	L	H	L	H	L	L	H	H
L	H	H	L	L	H	H	H	L	L
L	H	H	H	H	L	L	H	L	H
H	L	L	L	L	H	H	H	L	L
H	L	L	H	H	L	L	H	L	H
H	L	H	L	H	L	H	H	H	L
H	H	L	L	L	H	H	H	L	L
H	H	L	H	H	L	L	H	L	H
H	H	H	L	H	L	H	H	H	L
H	H	H	H	H	H	L	H	H	H

(b)

SN7482 Pseudo Parallel Adder -- Logic Diagram



SN7482 Two-Bit Adder -- Logic Equations

$$C1 = C0A1 + C0B1 + A1B1 \tag{4.20}$$

$$\begin{aligned}\sum 1 &= C0C1' + A1C1' + B1C1' + A1B1C0 \\ &= C1'(C0 + A1 + B1) + A1B1C0 \\ &= (C0'+A1')(C0'+B1')(A1'+B1')\,(C0 +A1+B1) +A1B1C0 \\ &= (C0'+ A1'B1')(A1'+B1')(C0 +A1+B1) +A1B1C0 \\ &= [C0'(A1+B1)+ C0A1'B1'](A1'+B1')+A1B1C0 \\ &= C0'A1B1'+C0'A1'B1+C0A1'B1'+A1B1C0 \\ &= C0 \oplus A1 \oplus B1\end{aligned} \tag{4.21}$$

Similarly

$$C2 = C1A2 + C1B2 + A2B2 \tag{4.22}$$

$$\sum 2 = C1 \oplus A2 \oplus B2$$

Add Time for SN7482 Adder Circuits

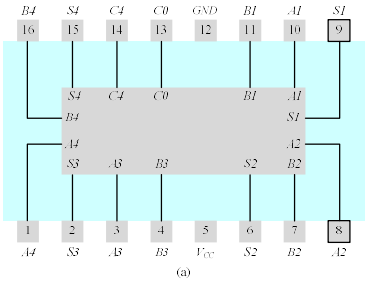
SN7482 propagation delays

$$t_{\Sigma 1} = 5\, t_{\text{gate}}$$
$$t_{C1} = 2\, t_{\text{gate}}$$
$$t_{\Sigma 2} = 6\, t_{\text{gate}}$$
$$t_{C2} = 4\, t_{\text{gate}}$$

SN7482-based ripple-carry adder (n-bits)

$$t_{\text{add}} = (2n + 2)\, t_{\text{gate}}$$

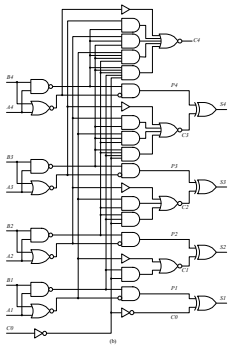
SN7483 Four-Bit Adder Module



(a)

Package Pin Configuration

SN7483 Four-Bit Adder Module -- Logic Diagram



SN7483 Four-Bit Adder -- Logic Equations

$$Pi = (BiAi)'(Ai + Bi)$$
$$= (Ai' + Bi')(Ai + Bi)$$
$$= Ai \oplus Bi \qquad (4.24)$$

$$\sum i = Pi \oplus Ci\text{-}1$$
$$= Ai \oplus Bi \oplus Ci\text{-}1 \qquad (4.25)$$

$$C1 = [C0'(A1B1)' + (A1 + B1)']'$$
$$= [C0'(A1B1)']'(A1 + B1)$$
$$= (C0+(A1B1))(A1 + B1)$$
$$= C0A1 + C0B1 + A1B1 \qquad (4.26)$$

Similarly

$$Ci = Ci\text{-}1Ai + Ci\text{-}1Bi + AiBi$$

Add Times for SN7483 Adder Circuits

SN7483 propagation delays

$$t_{\Sigma 1} = 3\ t_{gate}$$
$$t_{\Sigma 2} = t_{\Sigma 3} = t_{\Sigma 4} = 4\ t_{gate}$$
$$t_{C1} = t_{C2} = t_{C3} = t_{C4} = 3\ t_{gate}$$

SN7483-based Ripple-Carry Adder (n-bits)

$$t_{add} = (3m + 1)\, t_{gate}$$

where $m = \lceil n/4 \rceil$.

Fully Parallel Three-Bit Adder

$$c_0 = x_0 y_0 \qquad (4.30)$$
$$s_0 = x_0 \oplus y_0$$

$$\begin{aligned} c_1 &= x_1 y_1 c_0' + x_1 y_1 c_0 + x_1 y_1' c_0 + x_1' y_1 c_0 \\ &= x_1 y_1 + (x_1 \oplus y_1) c_0 \\ &= x_1 y_1 + (x_1 \oplus y_1)(x_0 y_0) \end{aligned} \qquad (4.31)$$
$$\begin{aligned} s_1 &= x_1 \oplus y_1 \oplus c_0 \\ &= x_1 \oplus y_1 \oplus x_0 y_0 \end{aligned}$$

$$\begin{aligned} c_2 &= x_2 y_2 + (x_2 \oplus y_2) c_1 \\ &= x_2 y_2 + (x_2 \oplus y_2)[x_1 y_1 + (x_1 \oplus y_1)(x_0 y_0)] \\ &= x_2 y_2 + (x_2 \oplus y_2)(x_1 y_1) + (x_2 \oplus y_2)(x_1 \oplus y_1)(x_0 y_0) \end{aligned} \qquad (4.32)$$
$$\begin{aligned} s_2 &= x_2 \oplus y_2 \oplus c_1 \\ &= x_2 \oplus y_2 \oplus [x_1 y_1 + (x_1 \oplus y_1)(x_0 y_0)] \end{aligned}$$

77

Add Time for a Fully Parallel Adder

Assuming a three-level realization

$$t_{add} = 3\, t_{gate}$$

However, the fan in requirements become impractical as n increases.

78

Carry Look-Ahead Adders -- Basic Idea

Recall that
$$\begin{aligned} c_i &= x_i y_i + x_i c_{i-1} + y_i c_{i-1} \\ &= x_i y_i + x_i y_i c_{i-1} + x_i y_i' c_{i-1} + x_i y_i c_{i-1} + x_i' y_i c_{i-1} \\ &= x_i y_i + x_i y_i' c_{i-1} + x_i' y_i c_{i-1} \\ &= x_i y_i + (x_i y_i' + x_i' y_i) c_{i-1} \\ &= x_i y_i + (x_i \oplus y_i) c_{i-1} \end{aligned}$$

Let
$$g_i = x_i y_i \quad \text{[carry generate]} \qquad (4.33)$$
$$p_i = x_i \oplus y_i \quad \text{[carry propagate]} \qquad (4.34)$$

Then
$$c_i = g_i + p_i c_{i-1}$$
$$s_i = p_i \oplus c_{i-1} \qquad (4.38)$$

79

Carry Look-Ahead Adders -- Three-Bit Example

$$c_0 = g_0 \qquad (4.35)$$
$$s_0 = p_0$$

$$\begin{aligned} c_1 &= g_1 + p_1 c_0 \\ &= g_1 + p_1 g_0 \end{aligned} \qquad (4.36)$$
$$s_1 = p_1 \oplus c_0$$

$$\begin{aligned} c_2 &= g_2 + p_2 c_1 \\ &= g_2 + p_2(g_1 + p_1 g_0) \\ &= g_2 + p_2 g_1 + p_2 p_1 g_0 \end{aligned} \qquad (4.37)$$
$$s_2 = p_2 \oplus c_1$$

80

Carry Look-Ahead Adder Design

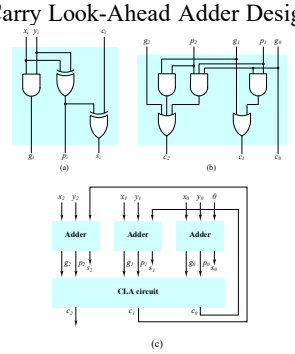


Figure 4.39

Add Times for Carry Look-Ahead Adders

Adder modules

$$t_g = t_p = t_s = t_{gate}$$

CLA module

$$t_c = 2\ t_{gate}$$

Overall

$$t_{add} = t_{gate} + 2\ t_{gate} + t_{gate}$$

$$= 4\ t_{gate}$$

Binary Subtraction Circuits

Recall that $(R)_2 = (P)_2 - (Q)_2$

$= (P)_2 + (-Q)_2$

$= (P)_2 + [Q]_2$

$= (P)_2 + (Q)_2' + 1$

For an SN7483 adder

$$(\Sigma)_2 = (A)_2 + (B)_2 + (C0)_2 \qquad (4.39)$$

where $\Sigma = \Sigma 4 \Sigma 3 \Sigma 2 \Sigma 1$, $A = A4A3A2A1$, and $B = B4B3B2B1$

If $C0 = 0$, $A = P$, and $B = Q$, then $(\Sigma)_2 = (P)_2 + (Q)_2$.

If $C0 = 1$, $A = P$, and $B = Q'$, then $(\Sigma)_2 = (P)_2 - (Q)_2$.

Two's Complement Adder/Subtracter

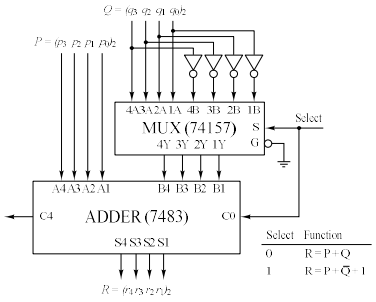


Figure 4.41

Arithmetic Overflow Detection

a_{n-1}	b_{n-1}	c_{n-2}	c_{n-1}	s_{n-1}	V
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	1	1	0

85

Overflow Detection Circuits

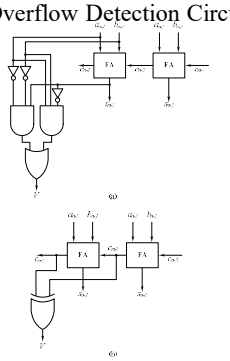
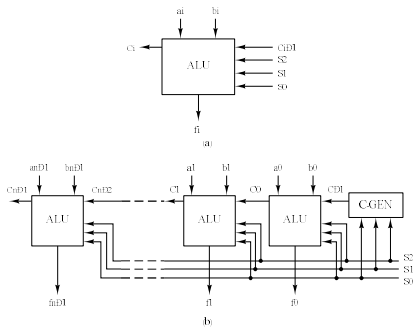
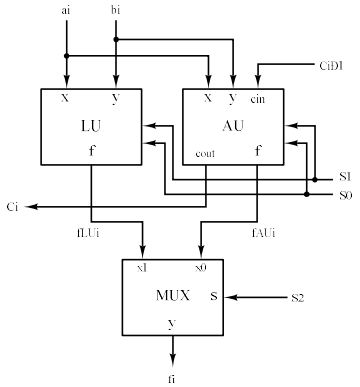


Figure 4.42

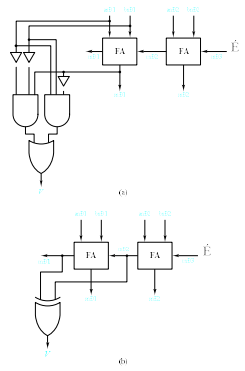
86



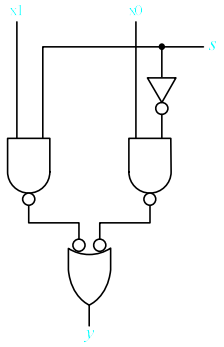
87



88



89

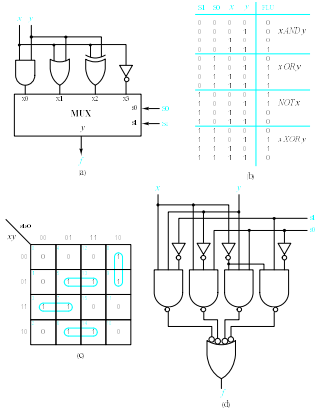


90

S1	S0	x	y	FLU	
0	0	0	0	0	
0	0	0	1	0	$x AND y$
0	0	1	0	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	1	1	$x OR y$
0	1	1	0	1	
0	1	1	1	1	
1	0	0	0	1	
1	0	0	1	1	$NOT x$
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	1	$x XOR y$
1	1	1	0	1	
1	1	1	1	0	

(b)

91



92

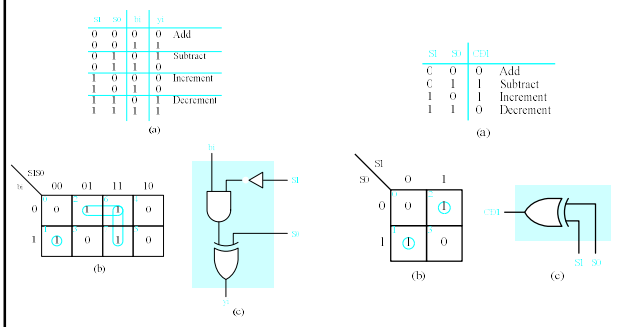
S1	S0	x	y	FLU	
0	0	0	0	0	
0	0	0	1	0	$x AND y$
0	0	1	0	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	1	1	$x OR y$
0	1	1	0	1	
0	1	1	1	1	
1	0	0	0	1	
1	0	0	1	1	$NOT x$
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	1	$x XOR y$
1	1	1	0	1	
1	1	1	1	0	

93

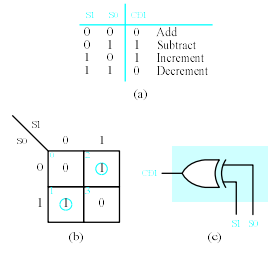
S1	S0	bi	xi	
0	0	0	0	Add
0	0	1	1	
0	1	0	1	Subtract
0	1	1	0	
1	0	0	0	Increment
1	0	1	0	
1	1	0	1	Decrement
1	1	1	1	

(a)

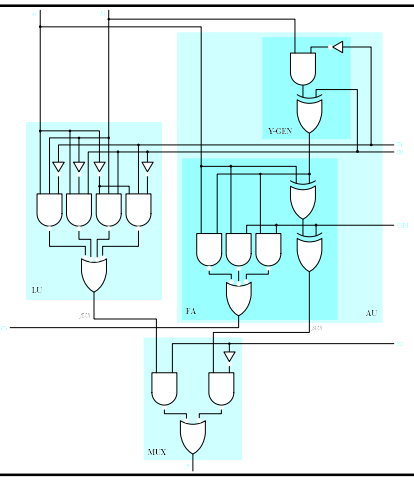
94



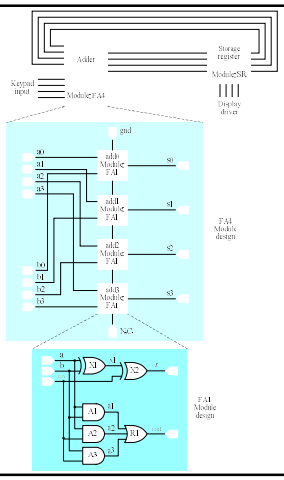
95



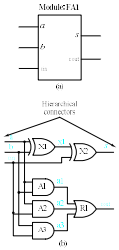
96


Y-GEN
LU
FA
AU
MUX

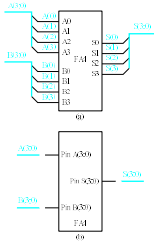
97


Adder
Storage register
ModuleSR
Keypad input
ModuleFA4
Display driver
gnd
FA4 Module design
NAG
FA1 Module design

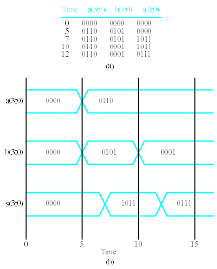
98


ModuleFA1
Hierarchical connectors

99


FA4

100


Time a(3:0) b(3:0) s(3:0)
0 0000 0000 0000
5 0110 0101 0000
7 0110 0101 1001
10 0110 0001 1001
12 0110 0001 0111
a(3:0)
b(3:0)
s(3:0)
0000
0110
0000
0101
0001
0000
1001
0111
0
5
10
15
Time

101