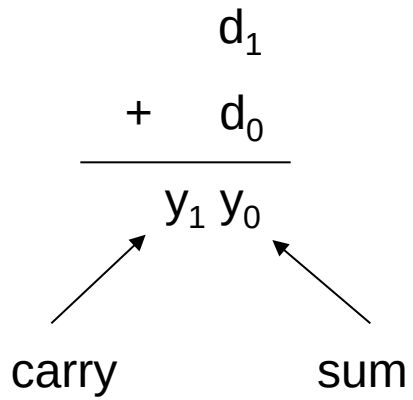
CHAPTER 11

DESIGN OF A SIMPLE SERIAL ARITHMETIC PROCESSOR

11.1 Introduction

Register transfer design
Data-path and control circuit.



11.2 Adder

Table 11.1 Truth table for a half adder.

d_1 d_0	y_1 y_0
0 0	0 0
0 1	0 1
1 0	0 1
1 1	1 0

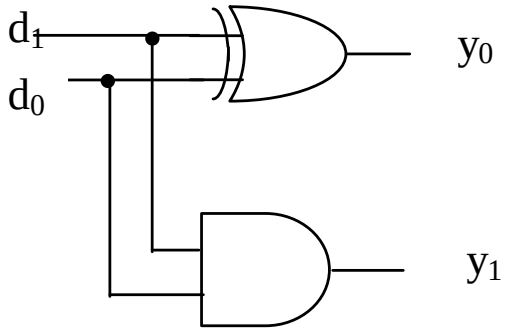


Figure 11.1 A half adder.

$c_3\ c_2\ c_1$ ← carries generated from addition

$a_3\ a_2\ a_1\ a_0$

$+\ \ b_3\ b_2\ b_1\ b_0$

$c_4\ S_3\ S_2\ S_1\ S_0$

c_i

a_i

$+\ \ b_i$

$c_{i+1}\ S_i$

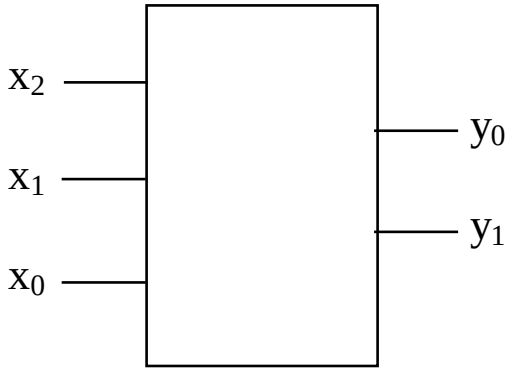


Figure 1.3 A 3-input, 2-output binary circuit.

Table 1.2 Truth table for the circuit in Figure 1.3.

x_2	x_1	x_0	y_1	y_0
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

Table 1.2

$y_0 = F(x_2, x_1, x_0) = \Sigma\, m(1, 2, 4, 7)$

$F(x_2, x_1, x_0) = x_2'x_1'x_0 + x_2'x_1x_0' + x_2x_1'x_0' + x_2x_1x_0$

$$\begin{aligned} F(x_2, x_1, x_0) &= (m_1 + m_2) + (m_4 + m_7) \\ &= (x_2'x_1'x_0 + x_2'x_1x_0') + (x_2x_1'x_0' + x_2x_1x_0) \\ &= x_2'(x_1'x_0 + x_1x_0') + x_2(x_1'x0' + x_1x_0) \\ &= x_2'(x_1 \oplus x_0) + x_2(x_1 \oplus x_0)' \\ &= x_2 \oplus x_1 \oplus x_0 \end{aligned}$$

x_0 \ x_2x_1	00	01	11	10
0		1		1
1	1		1	

Figure 5.27 K-map for y_0 in Table 1.2.

$$S_i = a_i \oplus b_i \oplus c_i$$

The canonical sum-of-products expression for the carry bit c_{i+1} is

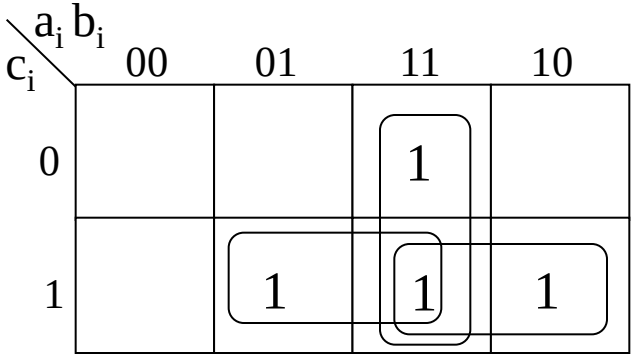
$$c_{i+1} = a_i' \, b_i \, c_i + a_i \, b_i' \, c_i + a_i \, b_i \, c_i' + a_i \, b_i \, c_i$$

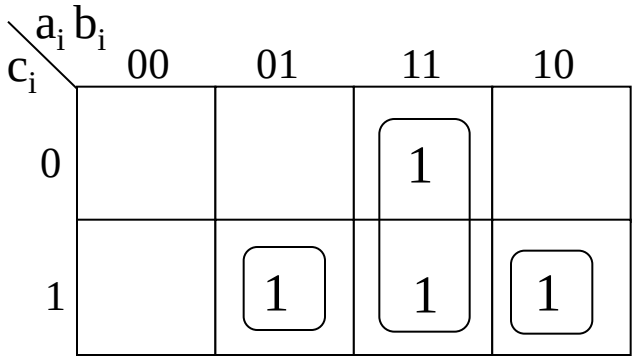
The simplest sum-of-products expression is

$$c_{i+1} = a_i \, b_i + b_i \, c_i + a_i \, c_i \qquad (11.1)$$

A different expression for the carry bit can be obtained as follows.

$$\begin{aligned} c_{i+1} &= a_i' \, b_i \, c_i + a_i \, b_i' \, c_i + a_i \, b_i \\ &= (a_i' \, b_i + a_i \, b_i')c_i + a_i \, b_i \\ &= (a_i \oplus b_i) \, c_i + a_i \, b_i \end{aligned} \qquad (11.2)$$





$S_i = a_i \oplus b_i \oplus c_i$

$c_{i+1} = (a_i \oplus b_i)\, c_i + a_i\, b_i$

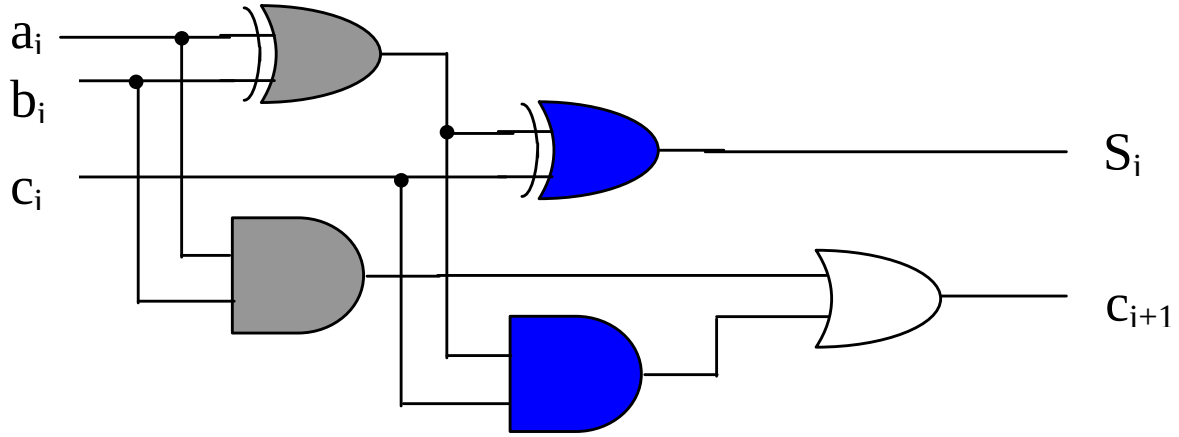


Figure 11.2 A full adder.

The addition of two n-bit numbers and an initial carry c_0 is given below.

$$
\begin{array}{rccccccc}
 & & & & & & & c_0 \\
 & a_{n-1} & a_{n-2} & \ldots\ldots\ldots\ldots & a_2 & a_1 & a_0 \\
+ & b_{n-1} & b_{n-2} & \ldots\ldots\ldots\ldots & b_2 & b_1 & b_0 \\
\hline
c_n & S_{n-1} & S_{n-2} & \ldots\ldots\ldots\ldots & S_2 & S_1 & S_0
\end{array}
$$

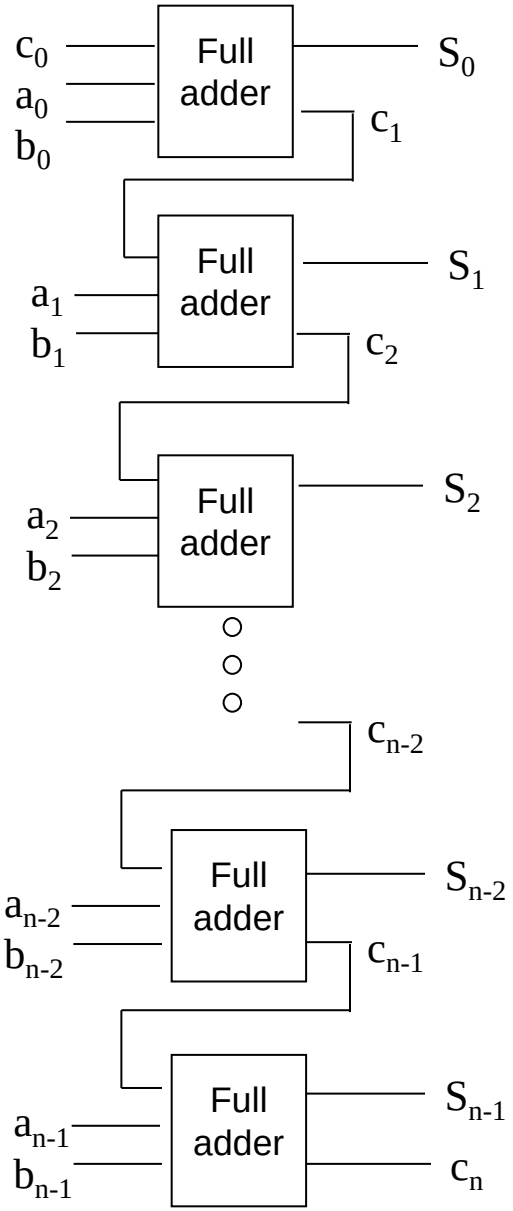


Figure 11.3 A ripple-carry adder.

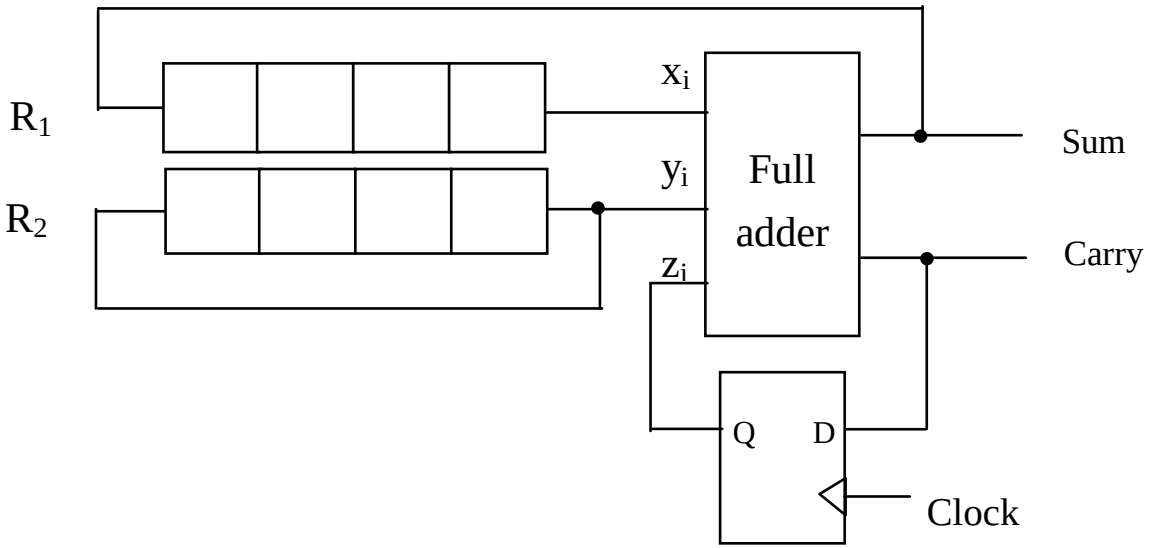
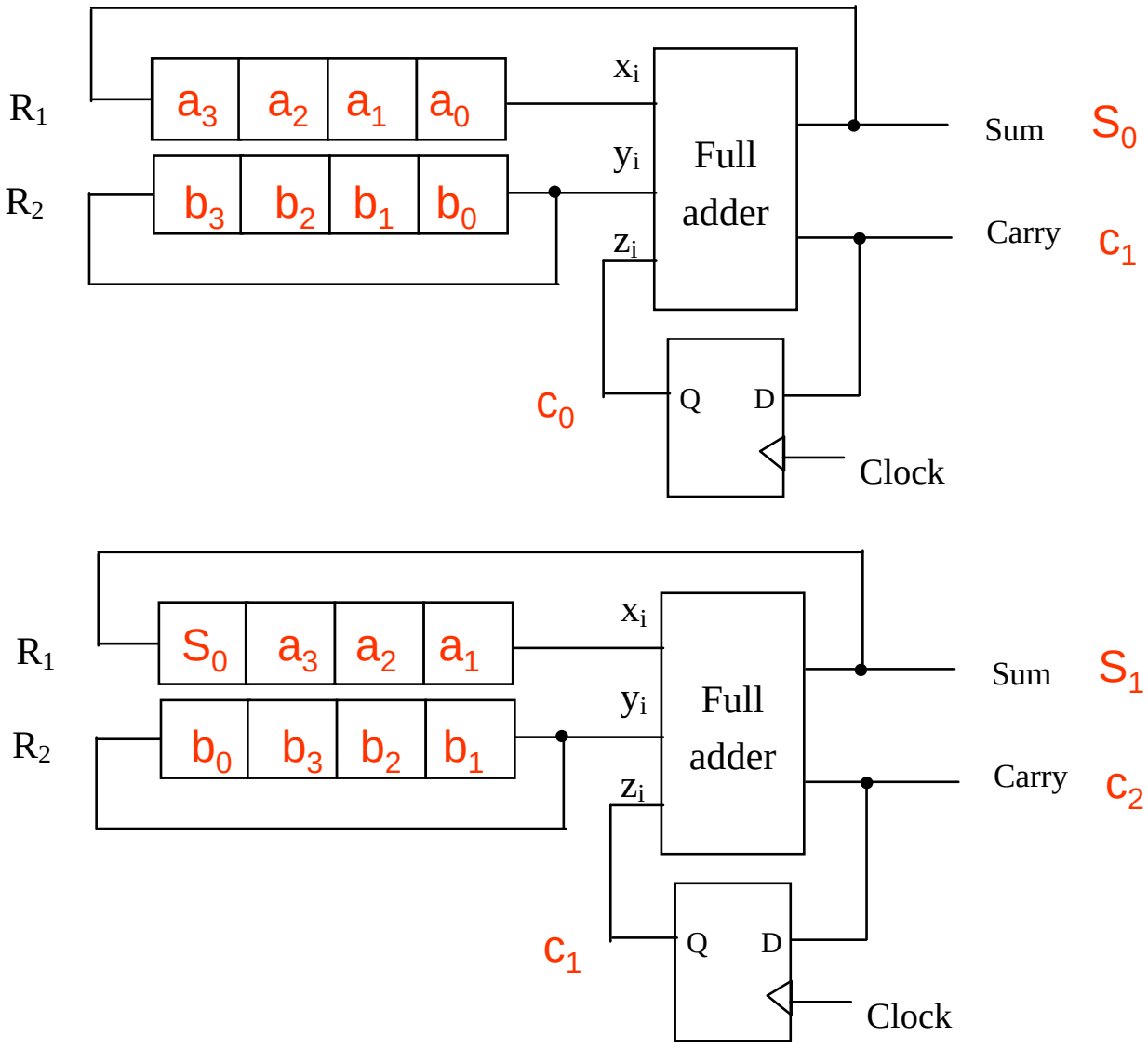
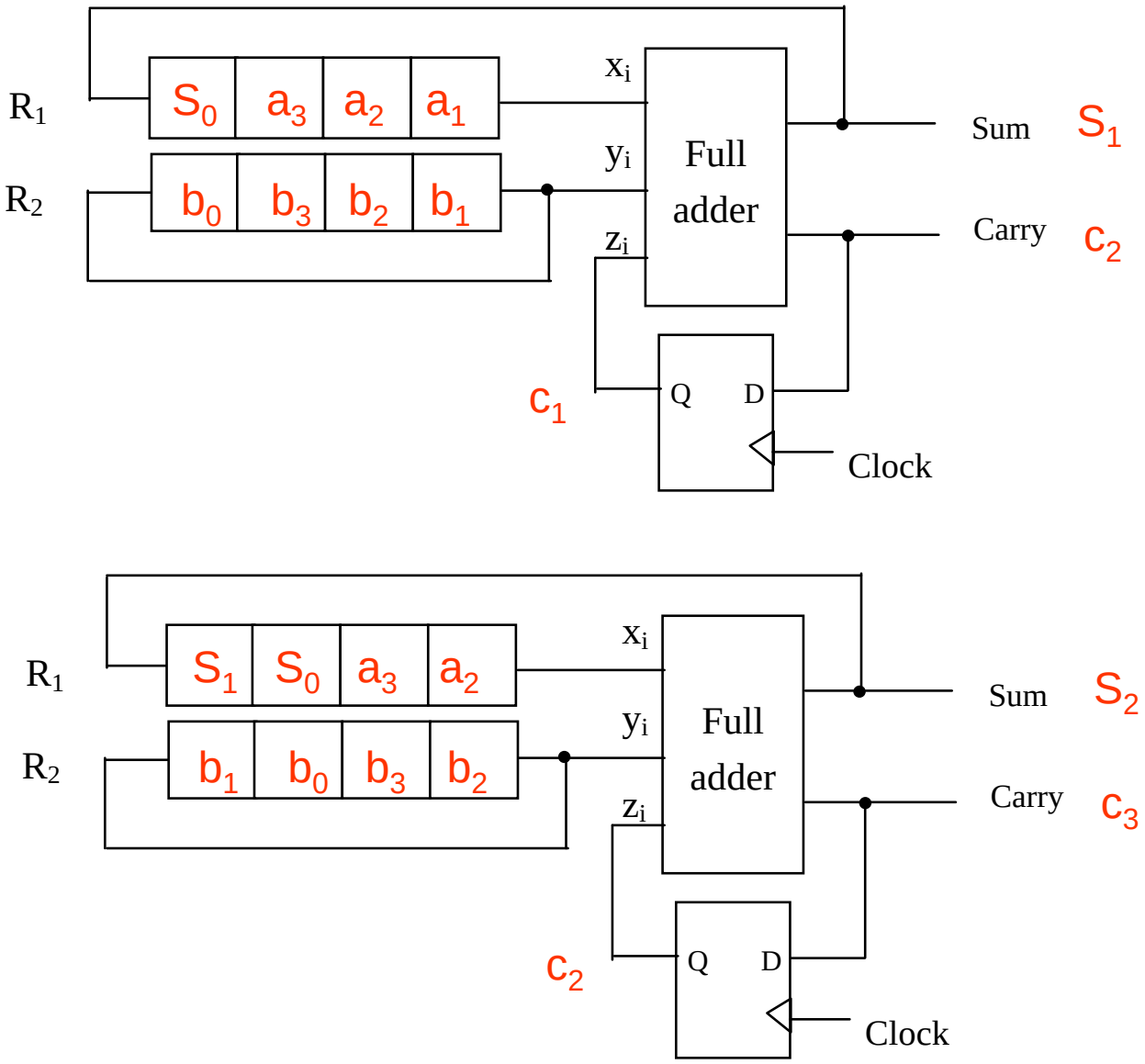


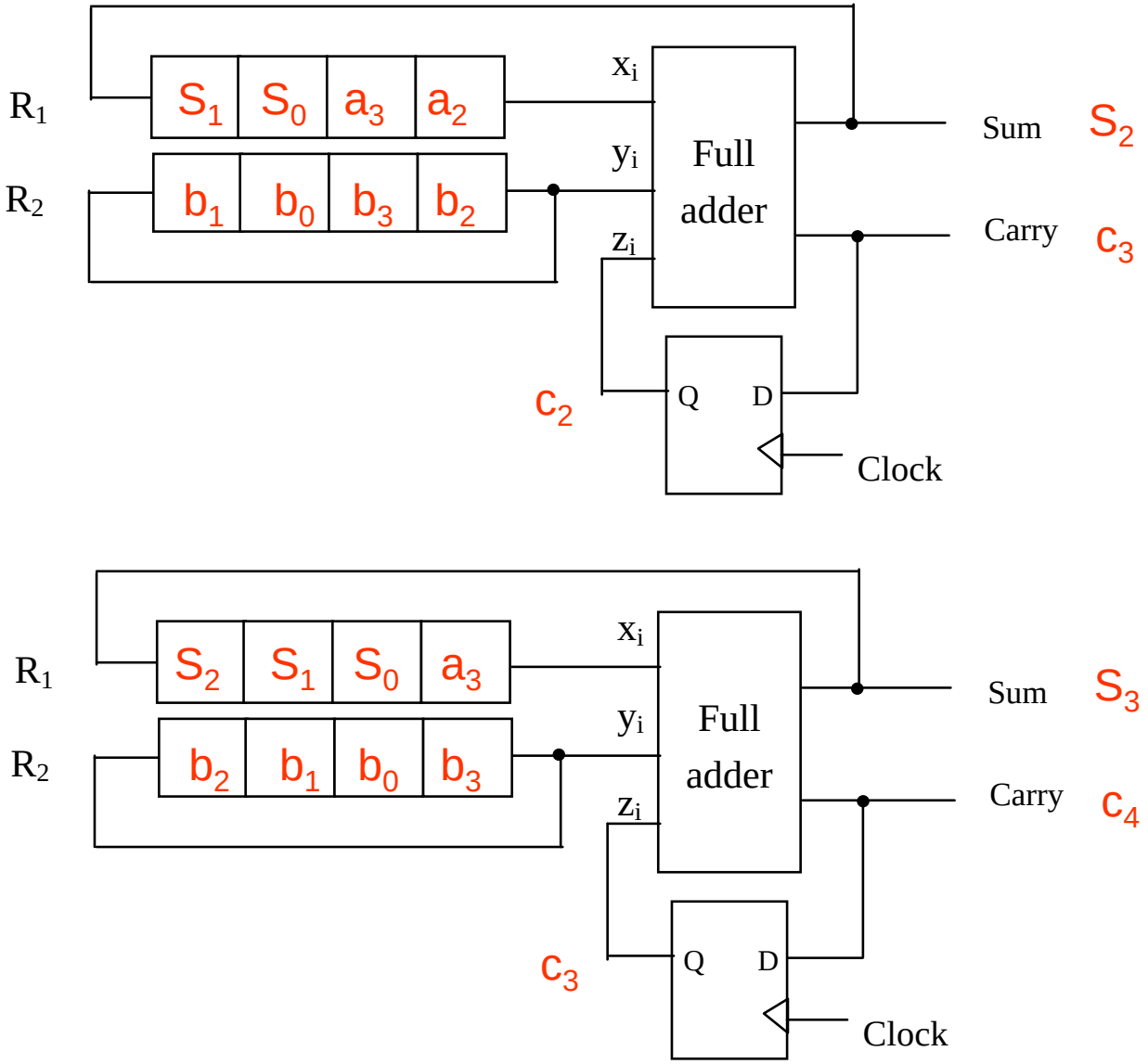
Figure 11.4 A serial adder.

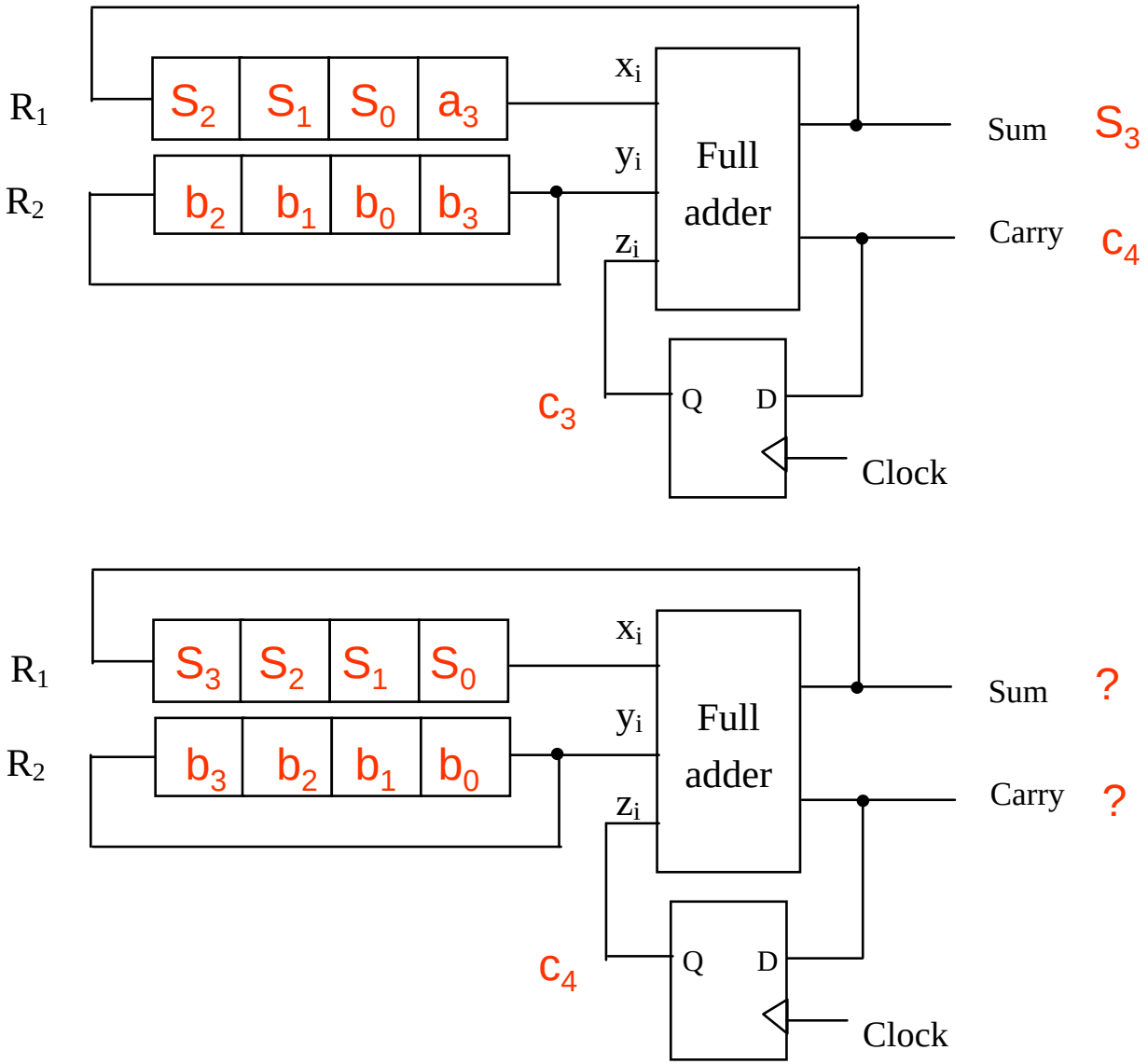
Table 11.2 Contents of the serial adder.

Clock cycle	R_1	R_2	x_i	y_i	z_i (Q)	Sum	Carry (D)
0	$a_3\ a_2\ a_1\ a_0$	$b_3\ b_2\ b_1\ b_0$	a_0	b_0	c_0	S_0	c_1
1	$S_0\ a_3\ a_2\ a_1$	$b_0\ b_3\ b_2\ b_1$	a_1	b_1	c_1	S_1	c_2
2	$S_1\ S_0\ a_3\ a_2$	$b_1\ b_0\ b_3\ b_2$	a_2	b_2	c_2	S_2	c_3
3	$S_2\ S_1\ S_0\ a_3$	$b_2\ b_1\ b_0\ b_3$	a_3	b_3	c_3	S_3	c_4
4	$S_3\ S_2\ S_1\ S_0$	$b_3\ b_2\ b_1\ b_0$	S_0	b_0	c_4	N/A	N/A


R1
a3 a2 a1 a0
xi
R2
b3 b2 b1 b0
yi
zi
Full adder
Sum
S0
Carry
c1
c0
Q
D
Clock
R1
S0 a3 a2 a1
xi
R2
b0 b3 b2 b1
yi
zi
Full adder
Sum
S1
Carry
c2
c1
Q
D
Clock


R1
S0 a3 a2 a1
R2
b0 b3 b2 b1
xi
yi
zi
Full adder
Sum
S1
Carry
c2
c1
Q
D
Clock
R1
S1 S0 a3 a2
R2
b1 b0 b3 b2
xi
yi
zi
Full adder
Sum
S2
Carry
c3
c2
Q
D
Clock


R1
R2
S1 S0 a3 a2
b1 b0 b3 b2
xi
yi
zi
Full adder
Sum
S2
Carry
c3
c2
Q
D
Clock
R1
R2
S2 S1 S0 a3
b2 b1 b0 b3
xi
yi
zi
Full adder
Sum
S3
Carry
c4
c3
Q
D
Clock


R1
S2 S1 S0 a3
R2
b2 b1 b0 b3
xi
yi
zi
Full adder
Sum
S3
Carry
c4
c3
Q
D
Clock
R1
S3 S2 S1 S0
R2
b3 b2 b1 b0
xi
yi
zi
Full adder
Sum
?
Carry
?
c4
Q
D
Clock

11.3 Signed Numbers

Sign-magnitude representation

$$-(2^{n-1}-1) \leq N \leq (2^{n-1}-1)$$

0 1 0 0 1 0 1 1	+ 75
0 1 1 1 1 1 1 1	+ 127
1 1 1 1 1 1 1 1	– 127
1 0 0 0 0 0 0 1	– 1
0 0 0 0 0 0 0 0	+ 0
1 0 0 0 0 0 0 0	– 0

2’s complement representation

$$-2^{n-1} \leq N \leq (2^{n-1} - 1)$$

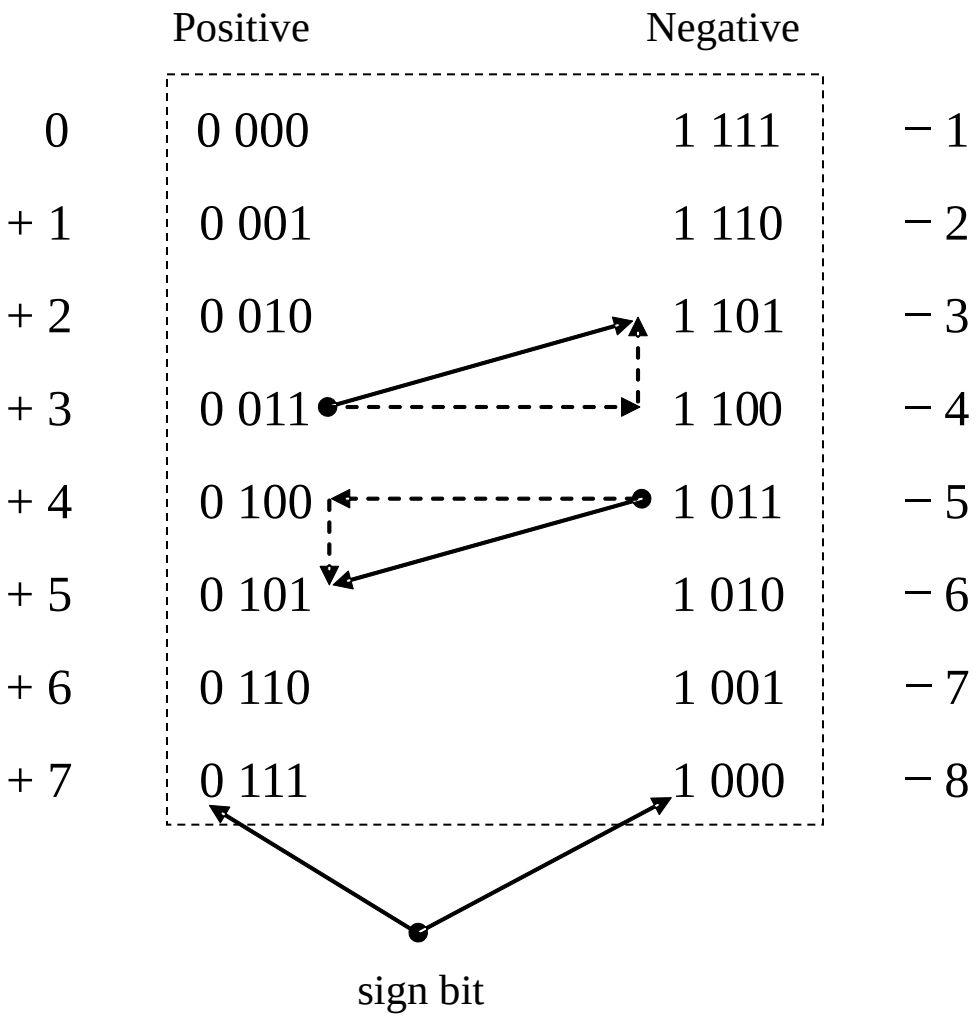


Figure 11.5 Two’s complement representation for 4-bit signed numbers.

The bit inversion of an n-bit number $Y = y_{n-1}\, y_{n-2}\, \ldots\ldots\, y_2 y_1 y_0$ is equivalent to the subtraction of an n-bit number $2^n - 1$ by Y, which is shown below.

$$\begin{array}{ccc} 1 & 1 & 1 \\ -\ 0 & -\ 1 & -\ y_i \\ \hline 1 & 0 & y_i' \end{array}$$

$$\begin{array}{lll} 1\ \ 1\ \ \ldots\ldots\ 1\ \ 1\ \ 1 & \leftrightarrow & (2^n - 1) \\ -\ \ y_{n-1}\ y_{n-2}\ \ \ldots\ldots\ y_2\ \ y_1\ \ y_0 & \leftrightarrow & Y \\ \hline y_{n-1}'\ y_{n-2}'\ \ldots\ldots\ y_2'\ y_1'\ y_0' & \leftrightarrow & {}^1Y \text{ (1's complement of Y)} \end{array}$$

${}^2Y$, the 2's complement of Y, is defined as

$${}^2Y = {}^1Y + 1$$

Thus the 2's complement of Y can also be obtained by subtracting Y from 2^n because

$${}^2Y = {}^1Y + 1 = (2^n - 1) - Y + 1 = 2^n - Y$$

Subtraction of two n-bit signed numbers: $A - B$

$$A - B = A + (-B) \longrightarrow A + {}^2B = A + (2^n - B) = (A - B) + 2^n$$

Decimal : 5 – 2 = 5 + (–2) = +3

Binary : 0101 – 0010 = 0101 + (– 0010)

1101	0101
+ 1	+ 1110
1110	1 0011

conversion to 2's complement

0101 + 1110 = 1 0011

discard of extra bit

0011

Decimal : 2 – 5 = 2 + (–5) = –3

Binary : 0010 – 0101 = 0010 + (–0101)

1010	0010
+ 1	+ 1011
1011	1101

conversion to 2's complement

0010 + 1011 = 0 1101

discard of extra bit

1101

The result is a negative number. The 2's complement of this number is 0011.

Decimal : 5 + 4 = +9

Binary : 0101 + 0100 = 1001

Positive Positive Negative

```
  0101
+ 1110
------
  1001
```

Decimal : –5 – 6 = – 5 + (– 6) = – 11

Binary : – (0101) + (– 0110)

(1011) + (1010) = 1 (0101)

Negative Negative Positive

```
  1011
+ 1010
------
 10101
```

Addition and Subtraction Methods for 2's Complement Signed Number Conversion

The conversions for positive signed numbers between decimal and binary are no difference from what have been done in Chapter 2.

The conversions for negative signed numbers between decimal and .

	a_{n-1}	a_{n-2}	a_{n-3}	..... a_3	a_2	a_1	a_0
Weight	-2^{n-1}	2^{n-2}	2^{n-3}	..... 2^3	2^2	2^1	2^0
	-2^{n-1}	2^{n-2}	2^{n-3}	 8	4	2	1

Note: The weight of signed bit is negative.

8-bit negative signed number 10011010

Binary number	1	0	0	1	1	0	1	0
Weight	–128	+64	+32	+16	+8	+4	+2	+1

(Add weights of all 1-bits)

Decimal number	–128			+16	+8		+2	= –102

Convert –79 to a 2's complement 8-bit signed number using the subtraction method

Weight		decimal number N
$-2^7 = -128$	$-79 - (-128)$	
	49	difference
$2^5 = 32$	$- 32$	
	17	difference
$2^4 = 16$	$- 16$	
	1	difference
$2^0 = 1$	$- 1$	
	0	difference (stop)

$$a_7 = a_5 = a_4 = a_0 = 1,\ a_6 = a_3 = a_2 = a_1 = 0$$

$$(-79)_{10} = (a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0)_2 = (10110001)_2$$

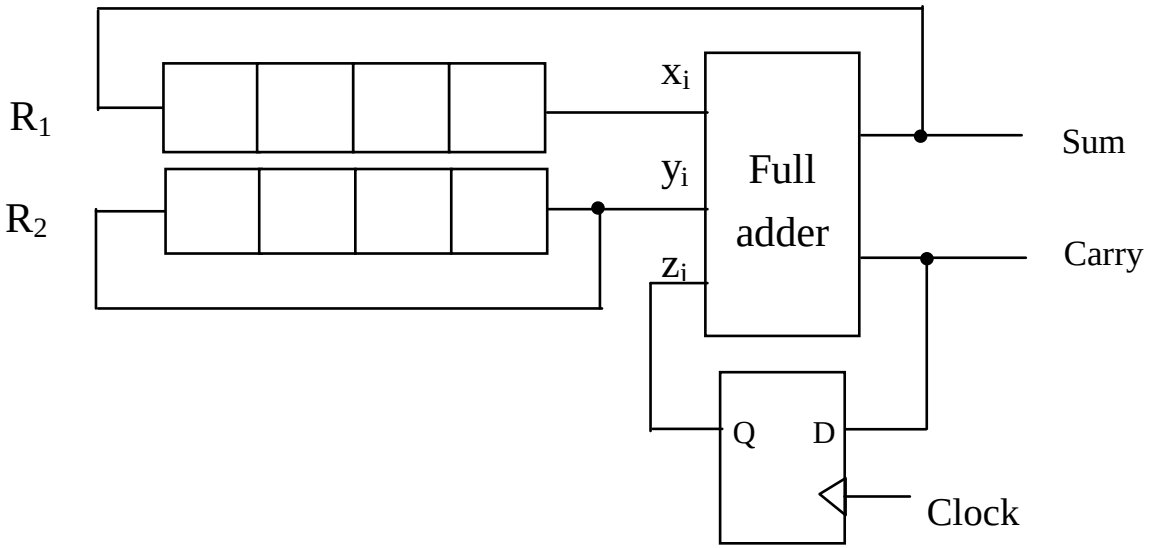


Figure 11.4 A serial adder.

Table 11.2 Contents of the serial adder.

Clock cycle	R_1	R_2	x_i	y_i	z_i (Q)	Sum	Carry (D)
0	$a_3\ a_2\ a_1\ a_0$	$b_3\ b_2\ b_1\ b_0$	a_0	b_0	c_0	S_0	c_1
1	$S_0\ a_3\ a_2\ a_1$	$b_0\ b_3\ b_2\ b_1$	a_1	b_1	c_1	S_1	c_2
2	$S_1\ S_0\ a_3\ a_2$	$b_1\ b_0\ b_3\ b_2$	a_2	b_2	c_2	S_2	c_3
3	$S_2\ S_1\ S_0\ a_3$	$b_2\ b_1\ b_0\ b_3$	a_3	b_3	c_3	S_3	c_4
4	$S_3\ S_2\ S_1\ S_0$	$b_3\ b_2\ b_1\ b_0$	S_0	b_0	c_4	N/A	N/A

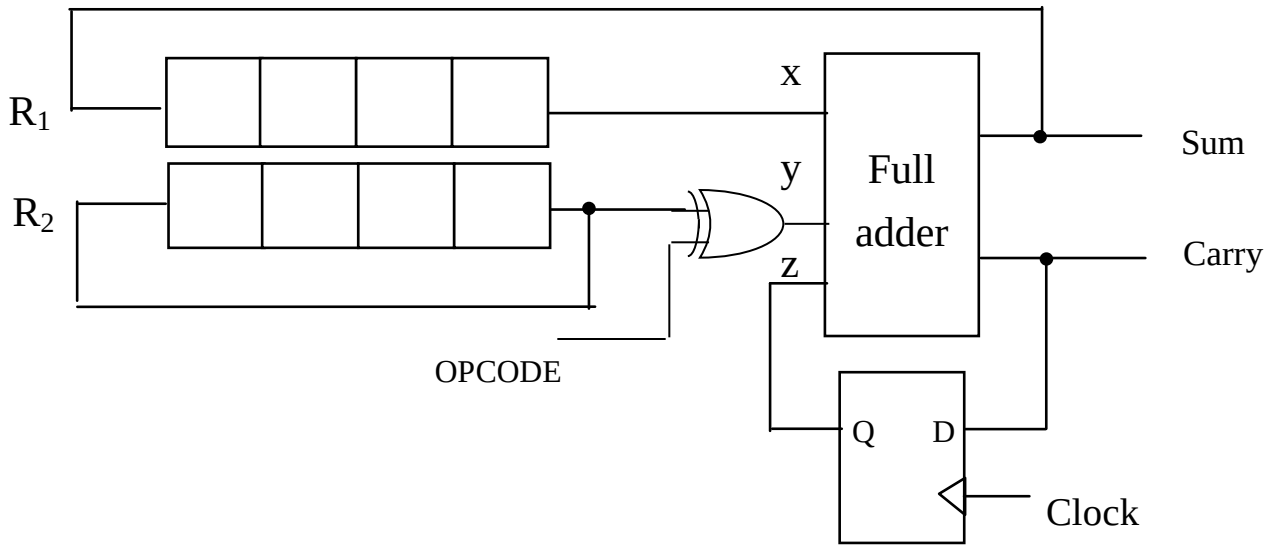


Figure 11.4 A serial adder.

OPCODE = 0	OPCODE = 1
Addition	Subtraction
0	1
$a_3\ a_2\ a_1\ a_0$	$a_3\ \ a_2\ \ a_1\ \ a_0$
$+\ b_3\ b_2\ b_1\ b_0$	$+\ b_3'\ b_2'\ b_1'\ b_0'$
A + B	A − B

11.4 Algorithmic State Machine (ASM) Chart

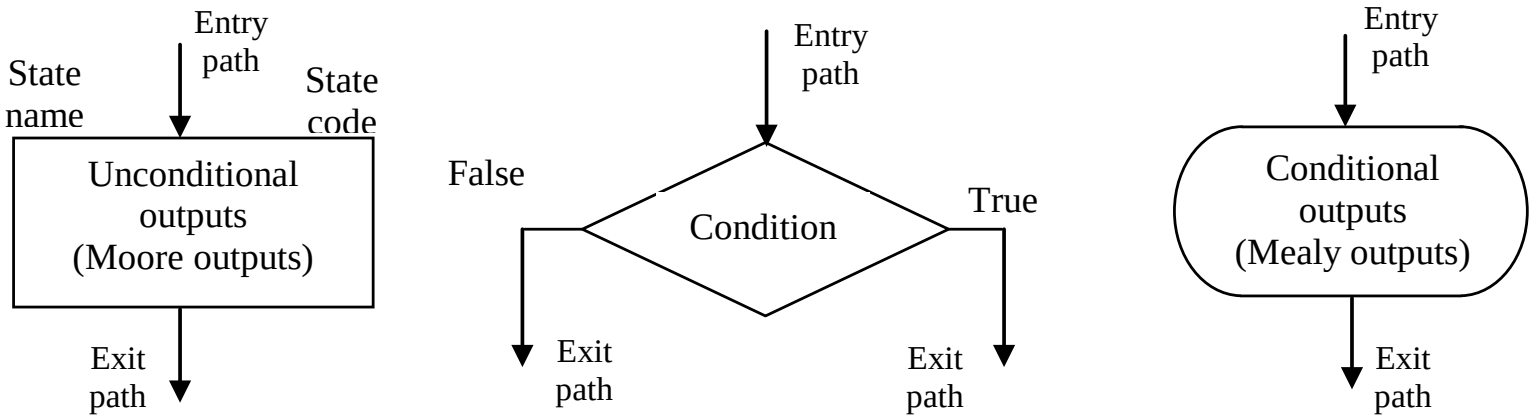
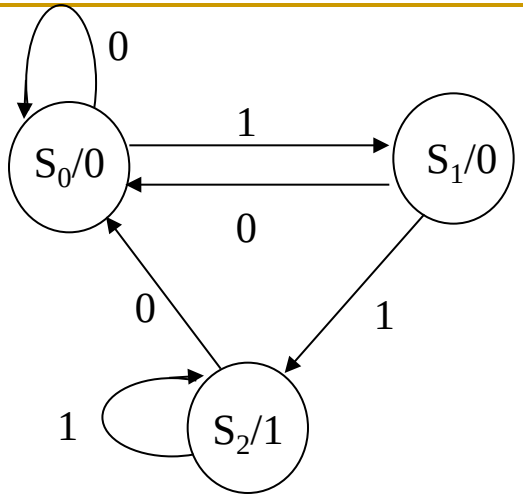


Figure 11.6 Basic elements of ASM charts. (a) State box. (b) Decision box. (c) Conditional output box.



(a)

State name	State code Q_1Q_0
S_0	0 0
S_1	0 1
S_2	1 1

(b)

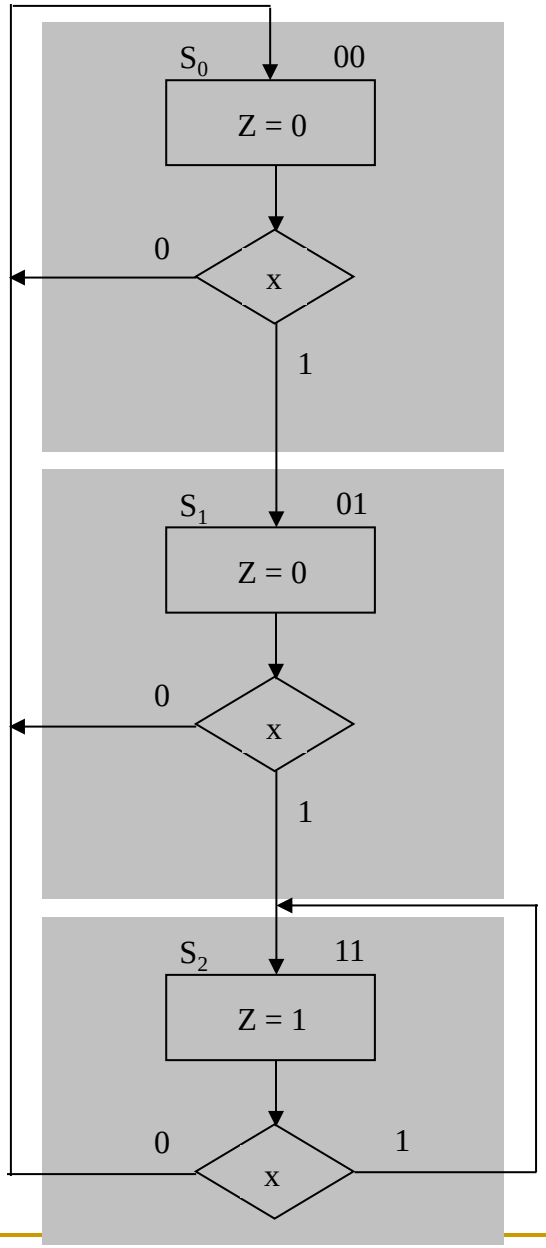
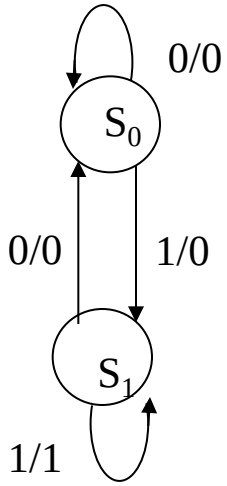


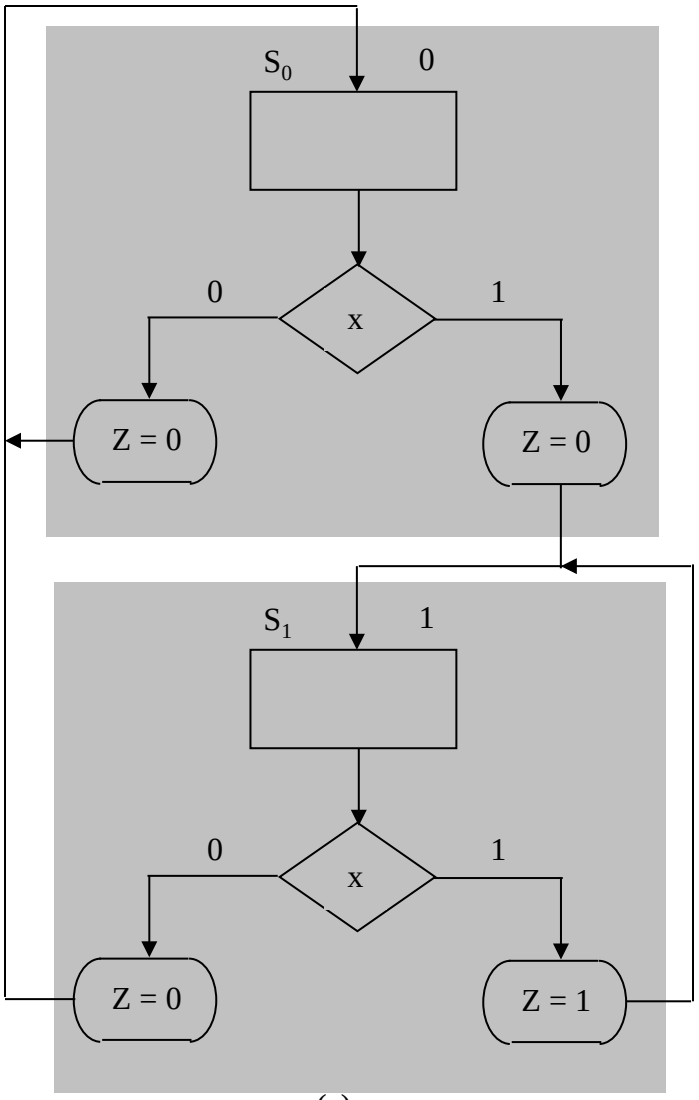
Figure 11.7 Conversion of a Moore state diagram to ASM chart. (a) State diagram. (b) State assignment. (c) ASM chart.



(a)

State name	State code Q
S_0	0
S_1	1

(b)



(c)

Figure 11.8 Conversion of a Mealy state diagram to ASM chart. (a) State diagram. (b) State assignment. (c) ASM chart.

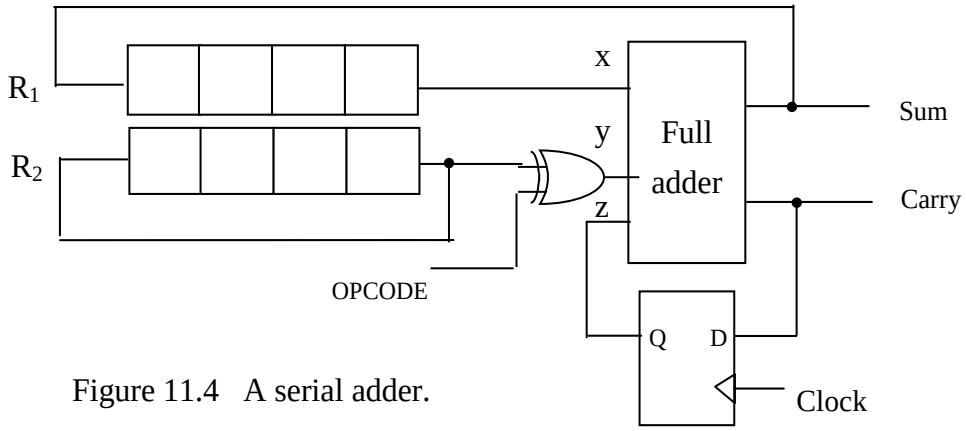


Figure 11.4 A serial adder.

$$y = OPCODE \oplus b_i$$

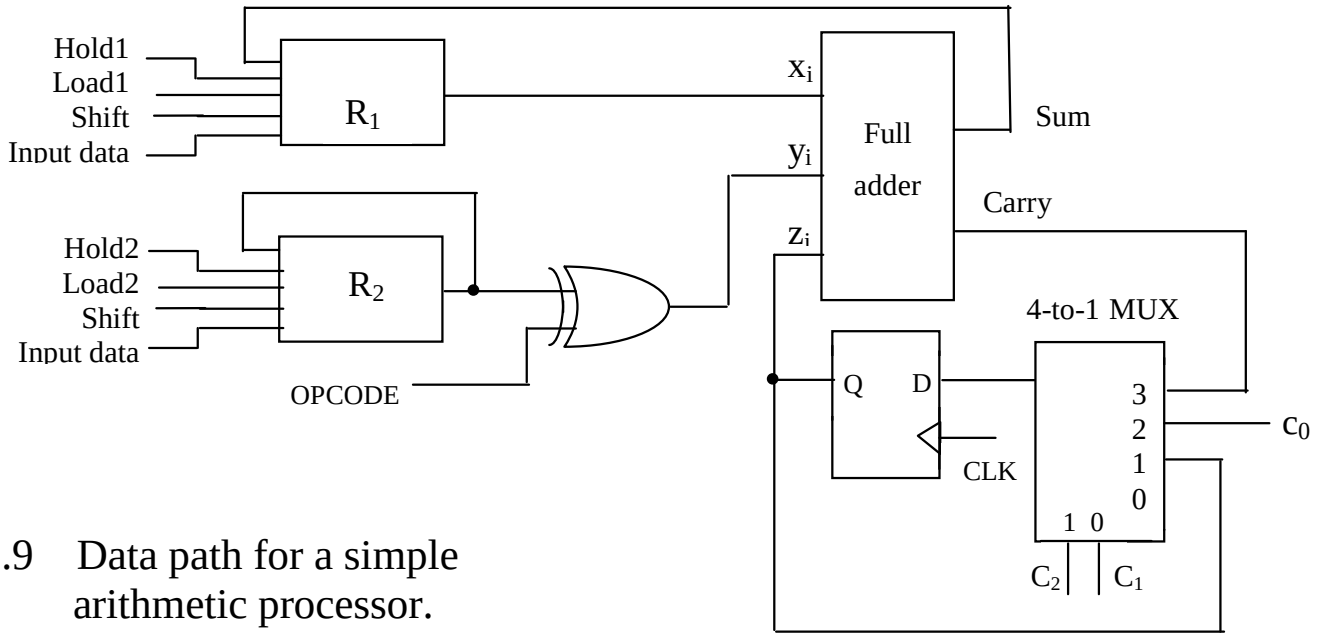


Figure 11.9 Data path for a simple arithmetic processor.

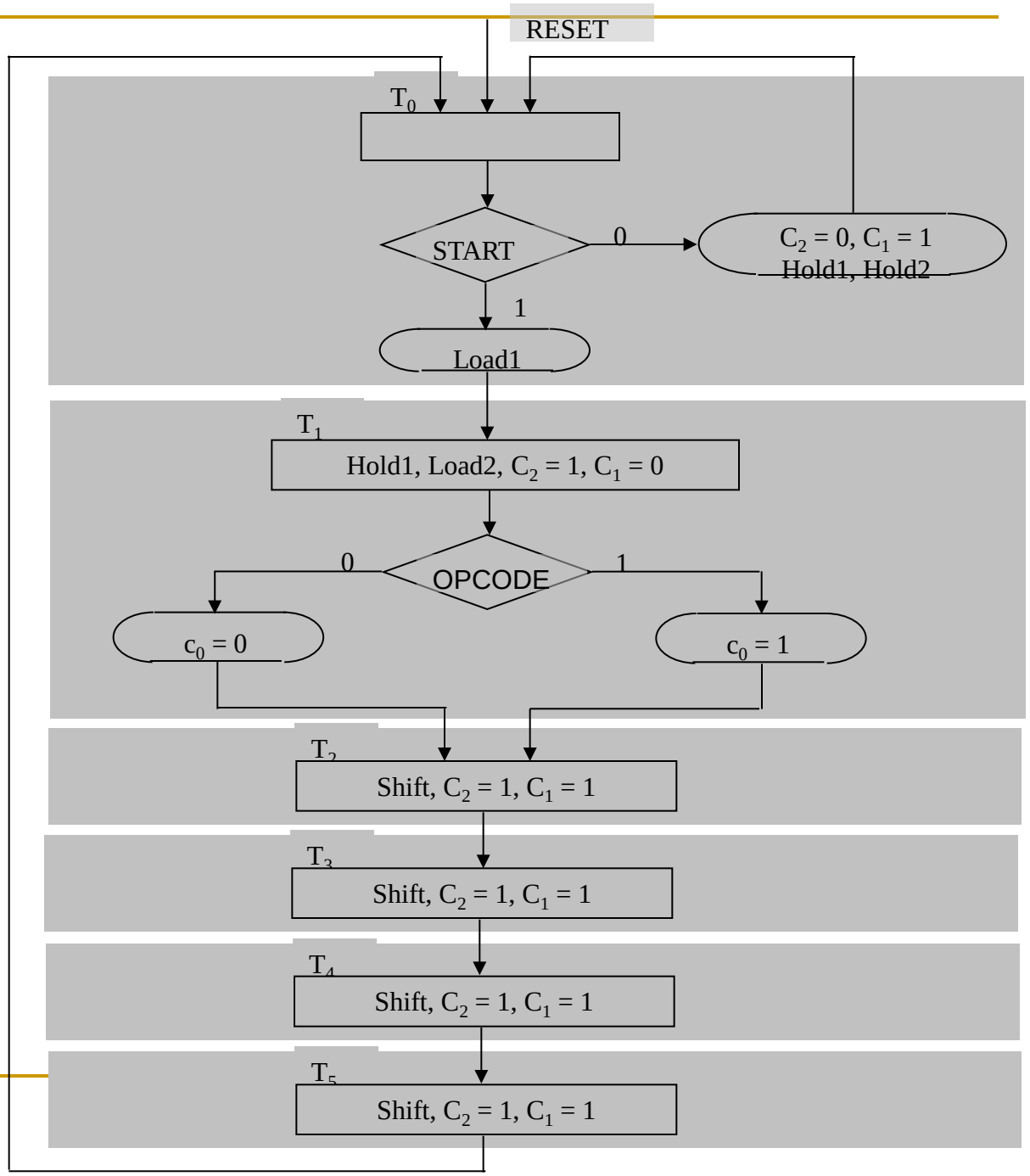


Figure 11.10 ASM chart for the arithmetic processor.

Ring Counter

Table 10.2 State assignment table for a 4-state ring counter

State	Q_0	Q_1	Q_2	Q_3
T_0	1	0	0	0
T_1	0	1	0	0
T_2	0	0	1	0
T_3	0	0	0	1

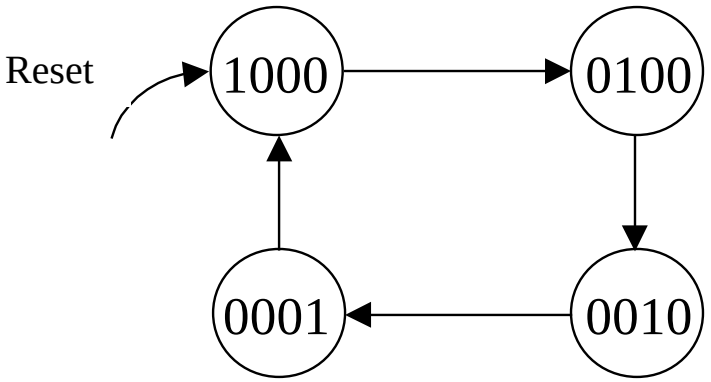


Figure 10.6 State diagram for a 4-state ring counter.

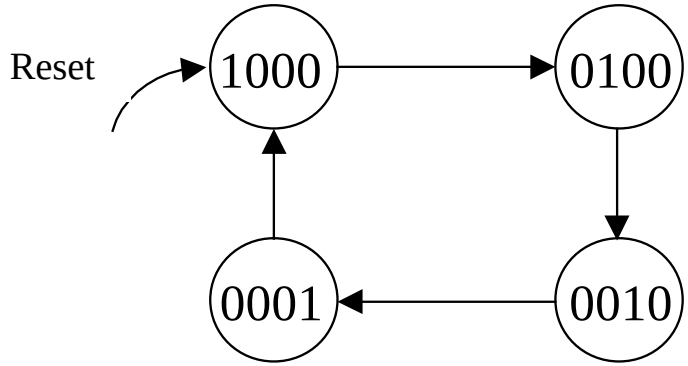


Figure 10.6 State diagram for a 4-state ring counter.

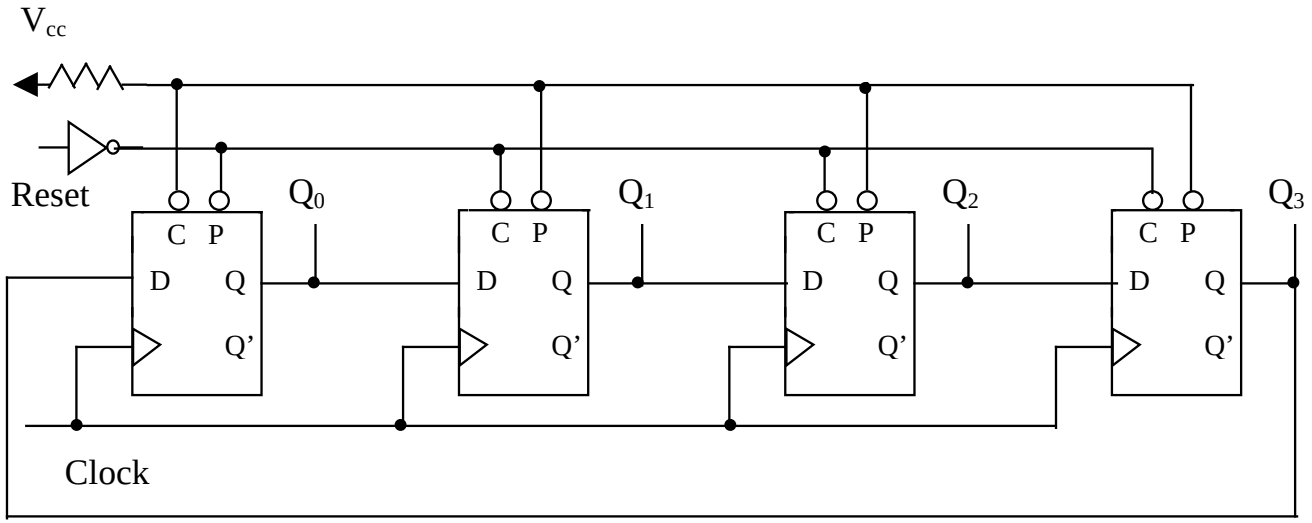
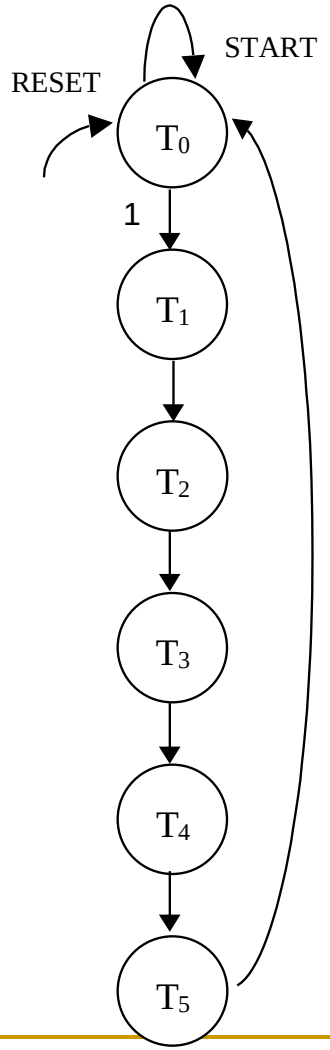
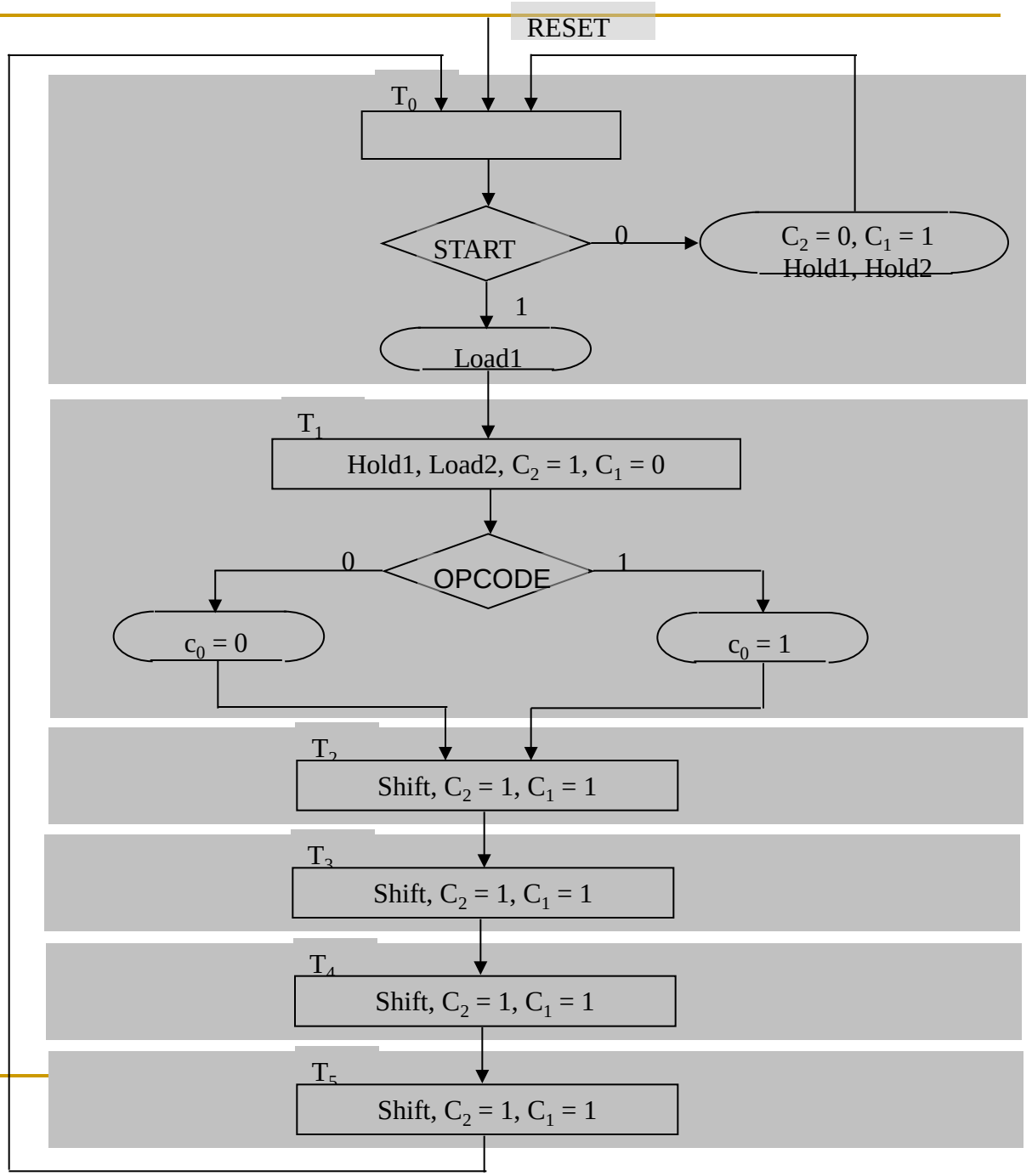


Figure 10.7 Circuit diagram for a 4-bit ring counter.

Figure 11.10 ASM chart for the arithmetic processor.





Design of State Generator

Table 11.3 State assignment.

State	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5
T_0	1	0	0	0	0	0
T_1	0	1	0	0	0	0
T_2	0	0	1	0	0	0
T_3	0	0	0	1	0	0
T_4	0	0	0	0	1	0
T_5	0	0	0	0	0	1

Table 11.4 State table for state generator.

Present state	START	Next state
T_0	0	T_0
T_0	1	T_1
T_1	d	T_2
T_2	d	T_3
T_3	d	T_4
T_4	d	T_5
T_5	d	T_0

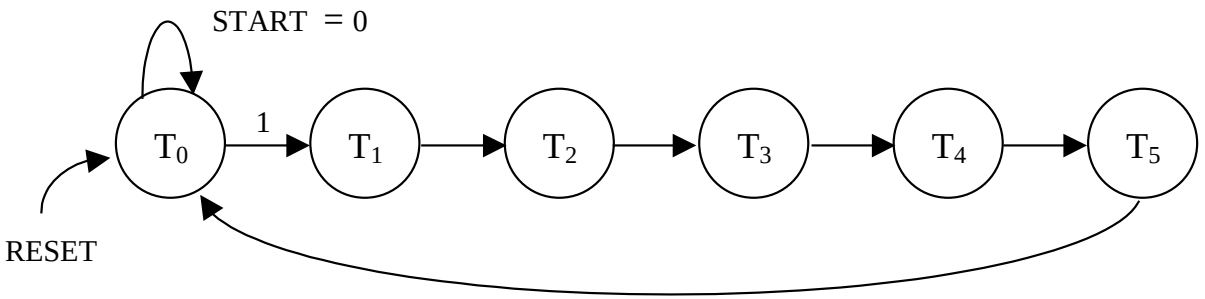


Figure 11.11 State diagram for state generator.

$$Q_0^+ = Q_0 \bullet START' + Q_5$$
$$Q_1^+ = Q_0 \bullet START$$
$$Q_2^+ = Q_1$$
$$Q_3^+ = Q_2$$
$$Q_4^+ = Q_3$$
$$Q_5^+ = Q_4$$

$$D_0 = Q_0^+ = Q_0 \bullet START' + Q_5$$
$$D_1 = Q_1^+ = Q_0 \bullet START$$
$$D_2 = Q_2^+ = Q_1$$
$$D_3 = Q_3^+ = Q_2$$
$$D_4 = Q_4^+ = Q_3$$
$$D_5 = Q_5^+ = Q_4$$

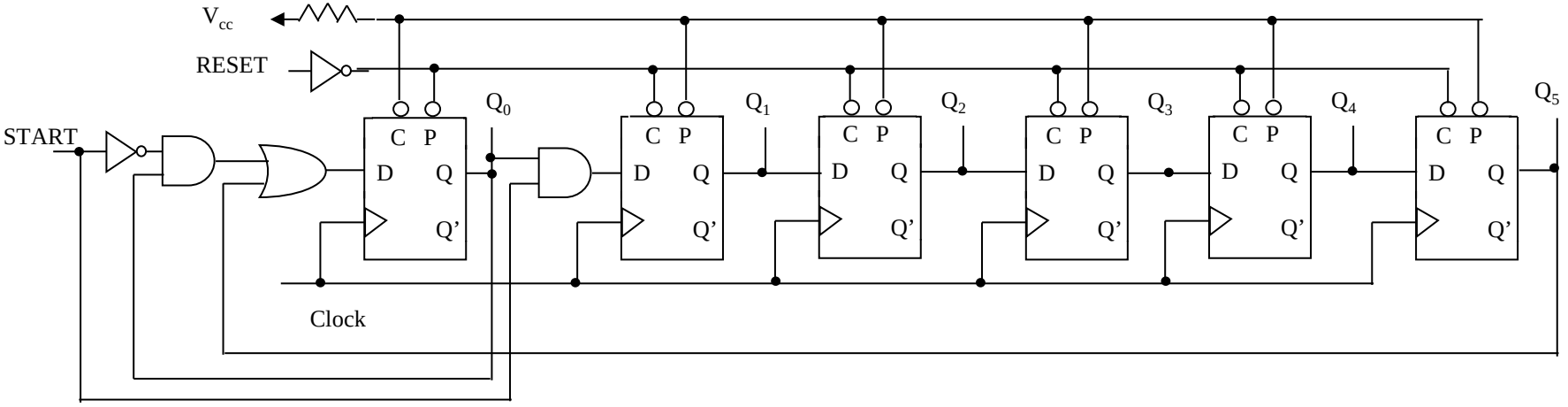


Figure 11.12 State generator.

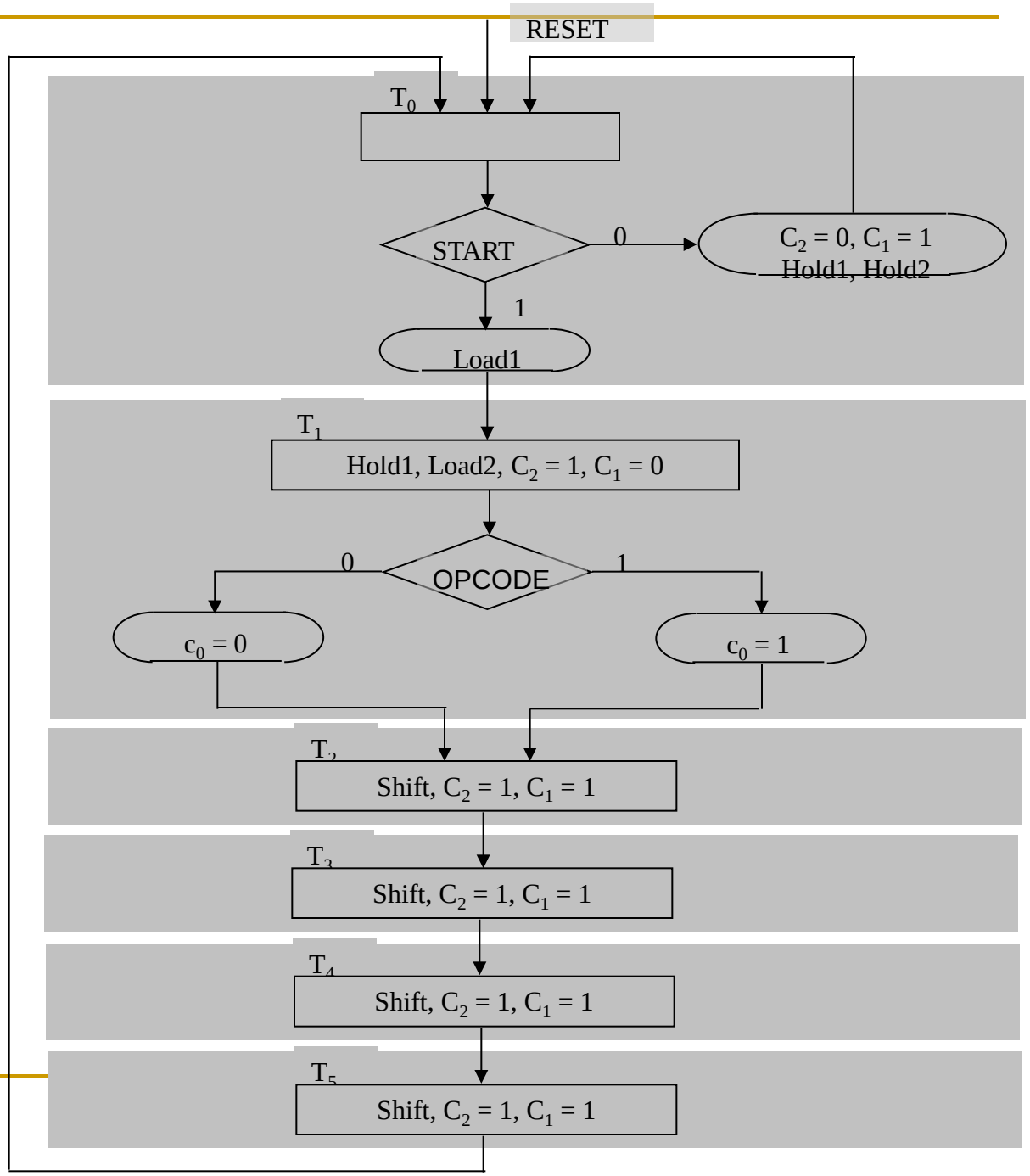


Figure 11.10 ASM chart for the arithmetic processor.

Table 10.1 Function table for a 4-bit universal shift register.

Function	s_1 s_0	Contents			
		Bit position			
		3	2	1	0
Hold	0 0	Q_3	Q_2	Q_1	Q_0
Shift right	0 1	SR_{in}	Q_3	Q_2	Q_1
Shift left	1 0	Q_2	Q_1	Q_0	SL_{in}
Parallel load	1 1	a_3	a_2	a_1	a_0

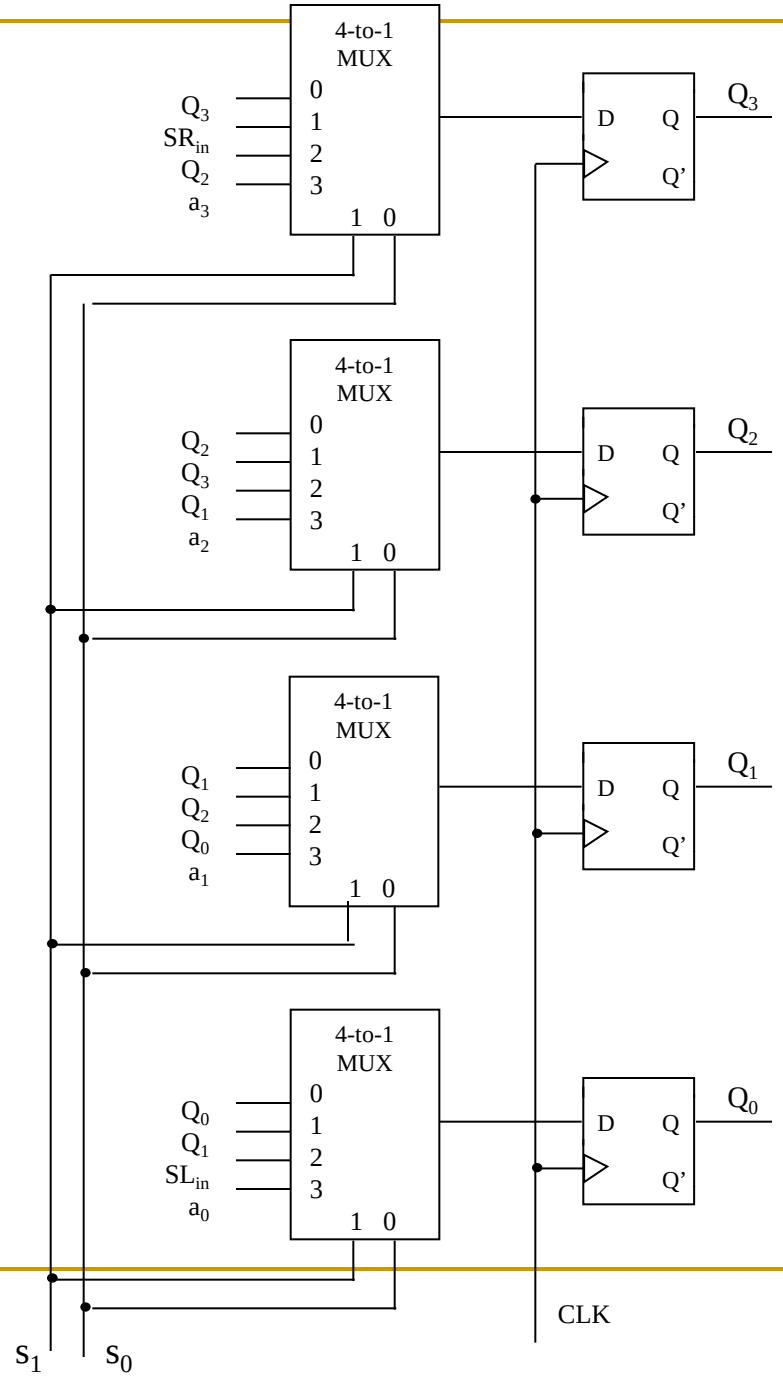


Figure 10.3 Design of a 4-bit universal shift register.

Design of Control Circuit

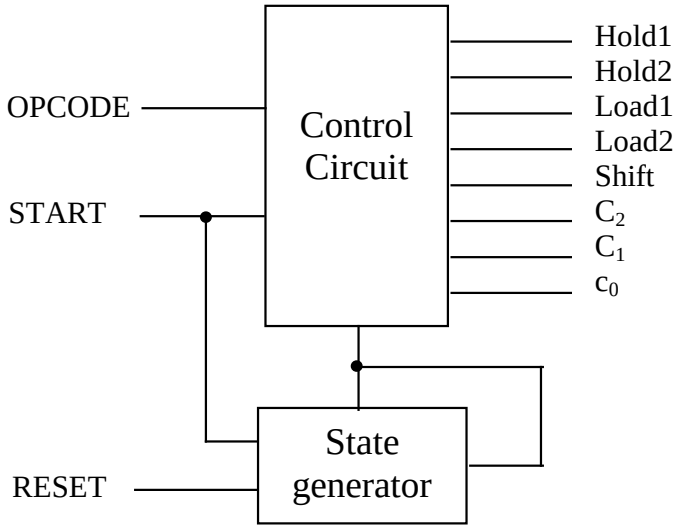
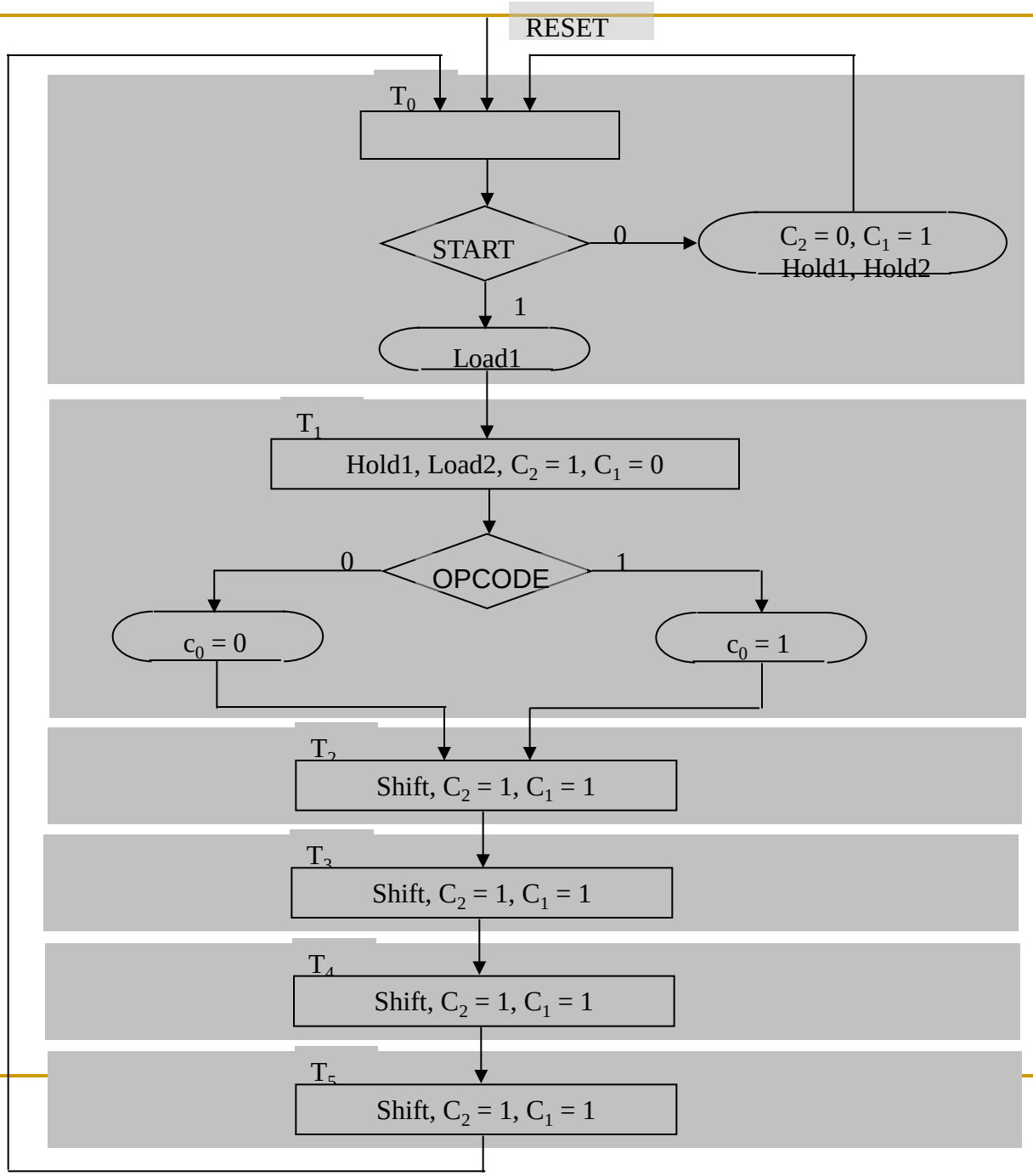
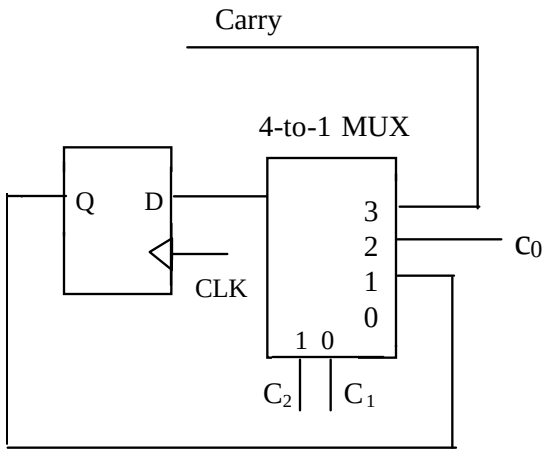


Figure 11.13 Block diagram for control circuit.

Table 11.5 Conversion of asserted signals to selection signals for shift register.

Asserted signal	$(s_1s_0)_{R1}$	$(s_1s_0)_{R2}$
Hold1	0 0	N/A
Hold2	N/A	0 0
Load1	1 1	N/A
Load2	N/A	1 1
Shift	0 1	0 1


RESET
T0
START
0
C2 = 0, C1 = 1
Hold1, Hold2
1
Load1
T1
Hold1, Load2, C2 = 1, C1 = 0
0
OPCODE
1
c0 = 0
c0 = 1
T2
Shift, C2 = 1, C1 = 1
T3
Shift, C2 = 1, C1 = 1
T4
Shift, C2 = 1, C1 = 1
T5
Shift, C2 = 1, C1 = 1



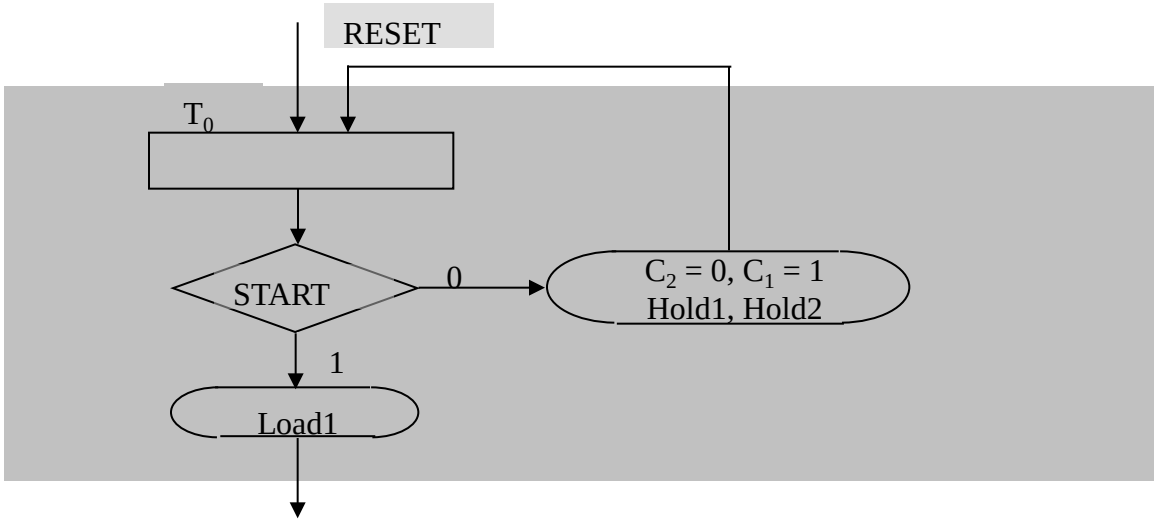
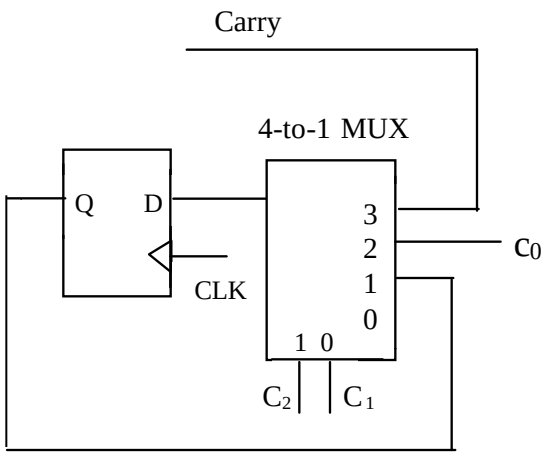


Table 11.6 Truth table for the control circuit.

State	START	$(s_1s_0)_{R1}$	$(s_1s_0)_{R2}$	C_2	C_1	c_0
T_0	0	0 0	0 0	0	1	d
T_0	1	1 1	d d	d	d	d



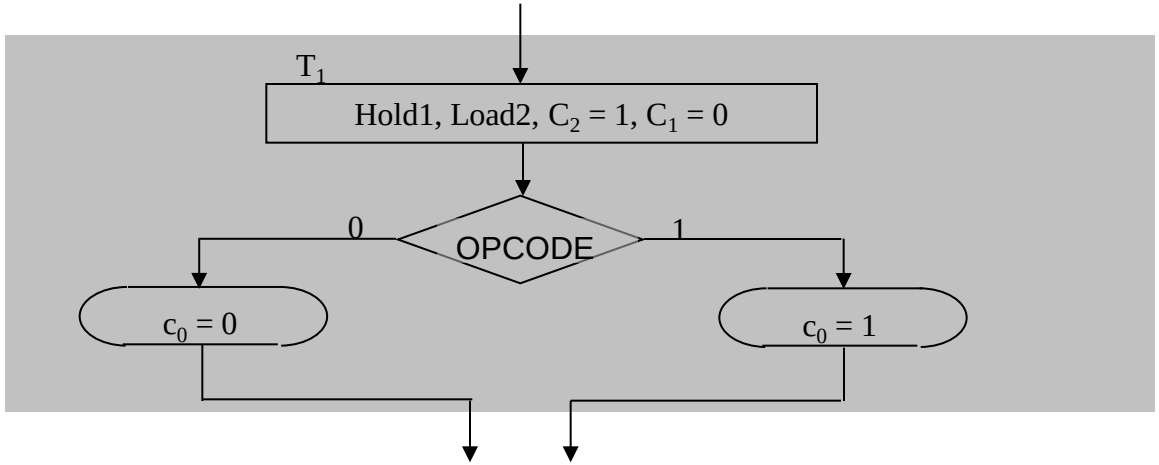
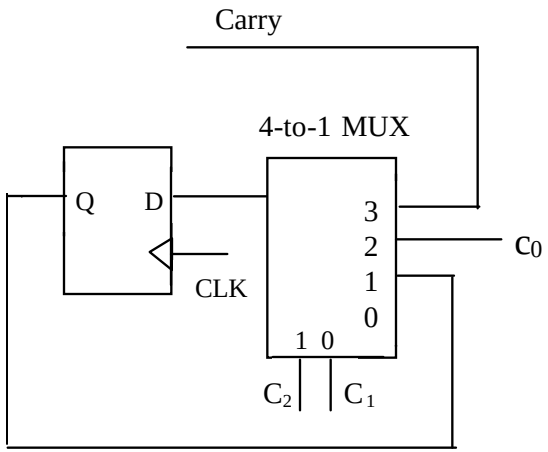


Table 11.6 Truth table for the control circuit.

State	START	$(s_1s_0)_{R1}$	$(s_1s_0)_{R2}$	C_2	C_1	c_0
T_0	0	0 0	0 0	0	1	d
T_0	1	1 1	d d	d	d	d
T_1	d	0 0	1 1	1	0	OPCODE



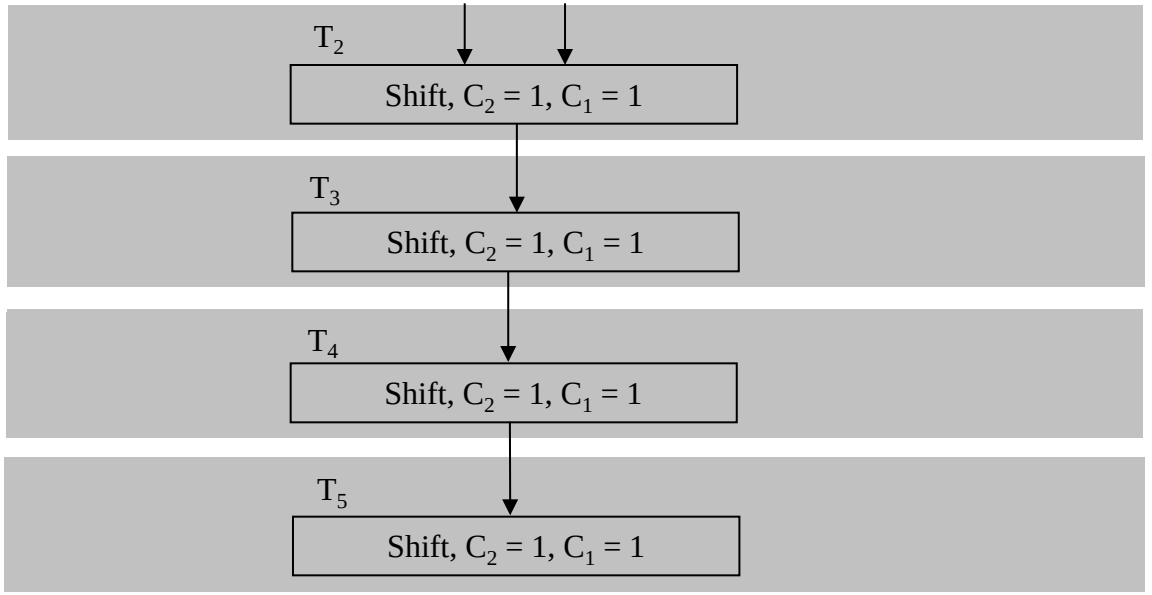


Table 11.6 Truth table for the control circuit.

State	START	$(s_1s_0)_{R1}$	$(s_1s_0)_{R2}$	C_2	C_1	c_0
T_0	0	0 0	0 0	0	1	d
T_0	1	1 1	d d	d	d	d
T_1	d	0 0	1 1	1	0	OPCODE
T_2	d	0 1	0 1	1	1	d
T_3	d	0 1	0 1	1	1	d
T_4	d	0 1	0 1	1	1	d
T_5	d	0 1	0 1	1	1	d

Table 11.6 Truth table for the control circuit.

State	START	$(s_1s_0)_{R1}$	$(s_1s_0)_{R2}$	C_2	C_1	c_0
T_0	0	0 0	0 0	0	1	d
T_0	1	1 1	d d	d	d	d
T_1	d	0 0	1 1	1	0	OPCODE
T_2	d	0 1	0 1	1	1	d
T_3	d	0 1	0 1	1	1	d
T_4	d	0 1	0 1	1	1	d
T_5	d	0 1	0 1	1	1	d

$(s_1)_{R1} = T_0 \bullet START$

$(s_0)_{R1} = T_0 \bullet START + T_2 + T_3 + T_4 + T_5$

$(s_1)_{R2} = T_1$

$(s_0)_{R2} = T_1 + T_2 + T_3 + T_4 + T_5$

$C_2 = T_1 + T_2 + T_3 + T_4 + T_5$

$C_1 = T_0 + T_2 + T_3 + T_4 + T_5$

$c_0 = OPCODE$

Table 11.7 Truth table for s_0', C_2', and C_1'

State	START	$(s_0')_{R1}$	$(s_0')_{R2}$	C_2'	C_1'
T_0	0	1	1	1	0
T_0	1	0	d	d	d
T_1	d	1	0	0	1
T_2	d	0	0	0	0
T_3	d	0	0	0	0
T_4	d	0	0	0	0
T_5	d	0	0	0	0

$(s_1)_{R1} = T_0 \bullet START$

$(s_0)_{R1} = T_0 \bullet START + T_2 + T_3 + T_4 + T_5$

$(s_1)_{R2} = T_1$

$(s_0)_{R2} = C_2 = T_1 + T_2 + T_3 + T_4 + T_5$

$C_1 = T_0 + T_2 + T_3 + T_4 + T_5$

c_0 = OPCODE

From Table 11.7,

$(s_0')_{R1} = T_0 \bullet START' + T_1$

$(s_0')_{R2} = C_2' = T_0$

$C_1' = T_1$

$(s_0)_{R1} = (T_0 \bullet START' + T_1)'$

$(s_0)_{R2} = C_2 = T_0'$

$C_1 = T_1'$

11.6 Revisit of Arithmetic Processor

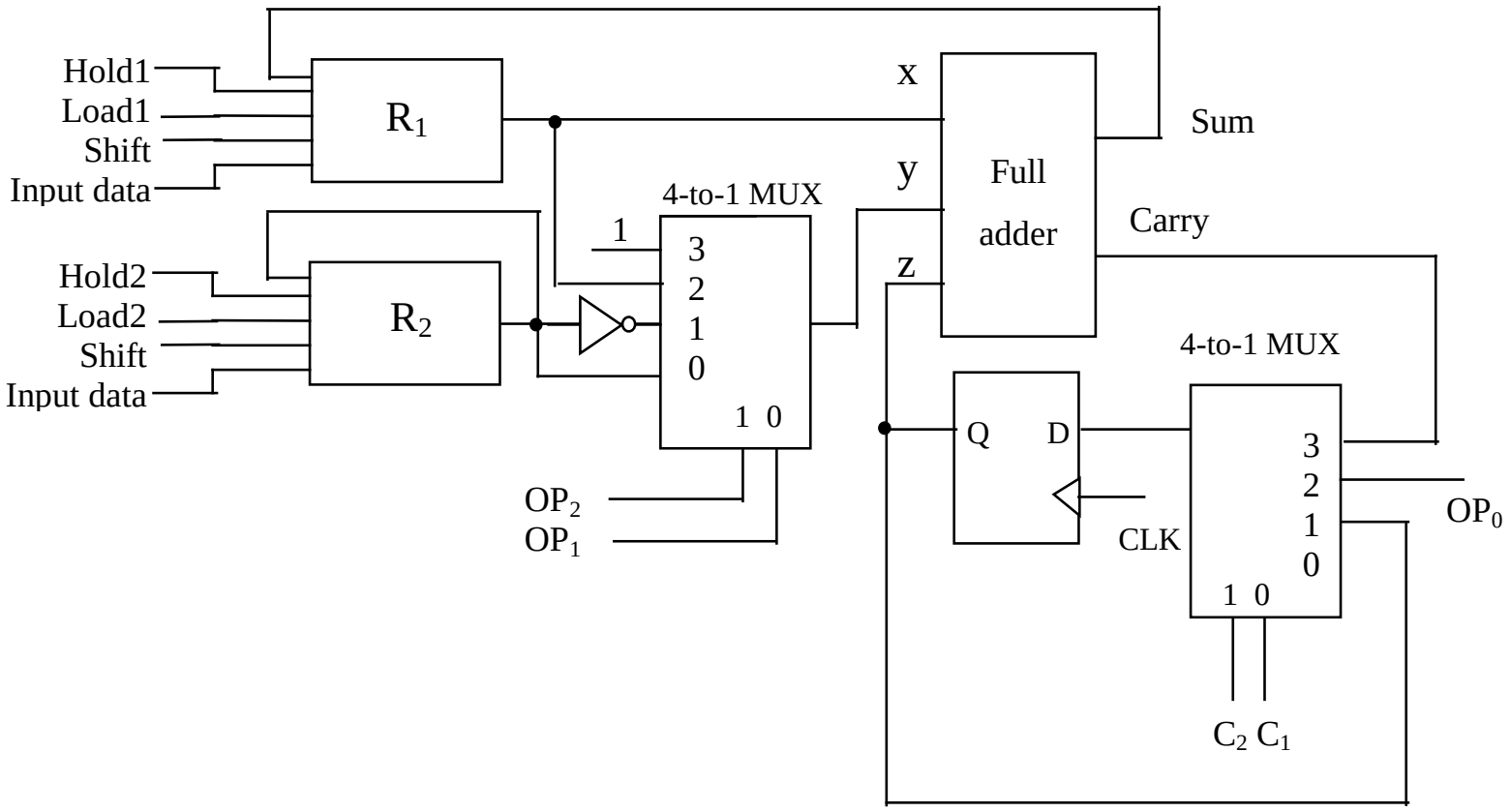
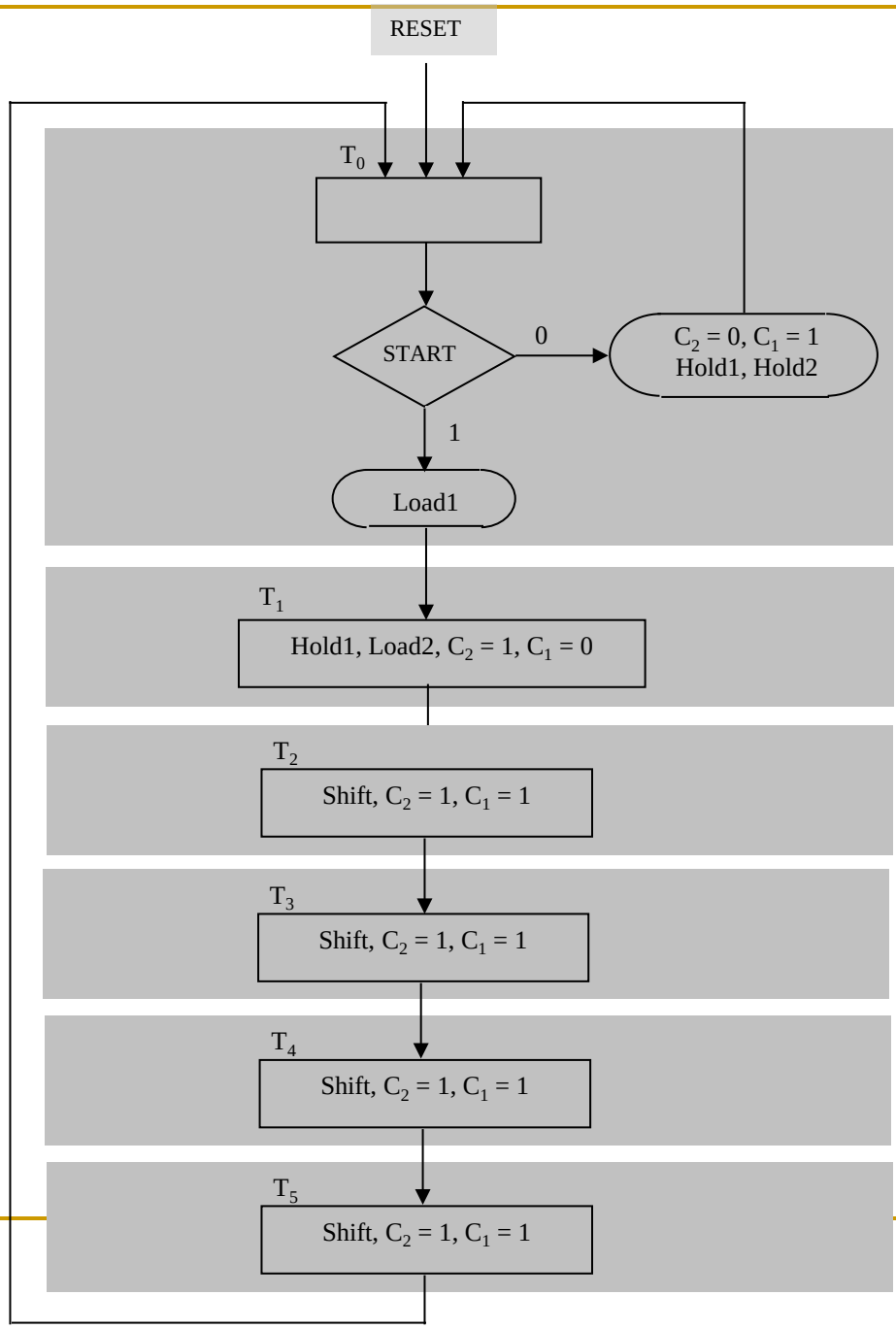
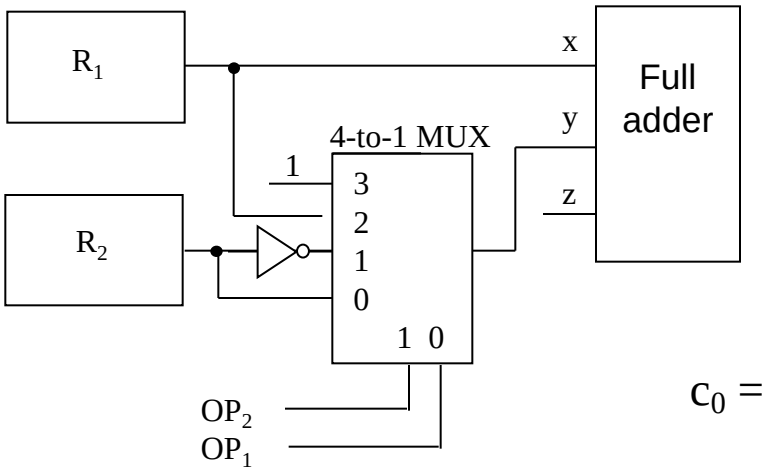


Figure 11.14 A processor for eight arithmetic functions.

Figure 11.15 ASM chart for the arithmetic processor in Figure 11.14.





$c_0 = OP_0$

Table 11.8 Arithmetic functions for the processor in Figure 11.14.

OP_2	OP_1	OP_0	$x_3x_2x_1x_0 + y_3y_2y_1y_0 + c_0$	Arithmetic function
0	0	0	$a_3a_2a_1a_0 + b_3b_2b_1b_0 + 0$	A + B
0	0	1	$a_3a_2a_1a_0 + b_3b_2b_1b_0 + 1$	A + B + 1
0	1	0	$a_3a_2a_1a_0 + b_3'b_2'b_1'b_0' + 0$	A – B – 1
0	1	1	$a_3a_2a_1a_0 + b_3'b_2'b_1'b_0' + 1$	A – B
1	0	0	$a_3a_2a_1a_0 + a_3a_2a_1a_0 + 0$	2A
1	0	1	$a_3a_2a_1a_0 + a_3a_2a_1a_0 + 1$	2A + 1
1	1	0	$a_3a_2a_1a_0 + 1111 + 0$	A – 1
1	1	1	$a_3a_2a_1a_0 + 1111 + 1$	A

Experiment 5 Arithmetic Processor

1. Sequence assignment

$Op_2\ Op_1$	Arithmetic Function
0 0	
0 1	
1 0	
1 1	

2. Processor Design

Construct the truth table for the input processor. (Use a_i, b_i, 0, 1)

$Op_2\ Op_1$	x_i	y_i	c_0
00			
01			
10			
11			

Experiment 5 Arithmetic Processor

1. Sequence assignment

$Op_2\ Op_1$	Arithmetic Function
0 0	2B + 1
0 1	A - B
1 0	A + B
1 1	A - 1

2. Processor Design

Construct the truth table for the input processor. (Use a_i, b_i, 0, 1)

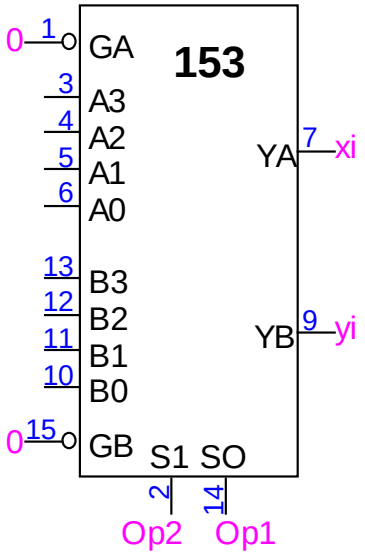
$Op_2\ Op_1$	x_i	y_i	c_0
00	b_i	b_i	1
01	a_i	b_i'	1
10	a_i	b_i	0
11	a_i	1	0

2. Processor Design

Construct the truth table for the input processor. (Use a_i, b_i, 0, 1)

Op_2 Op_1	x_i	y_i	c_0
00	b_i	b_i	1
01	a_i	b_i'	1
10	a_i	b_i	0
11	a_i	1	0

Label the data inputs of the two 4-to-1 multiplexers given below for the realization of x_i, y_i.



Express the initial carry c_0 as a function of Op_2 and Op_1.

c_0 =

2. Processor Design

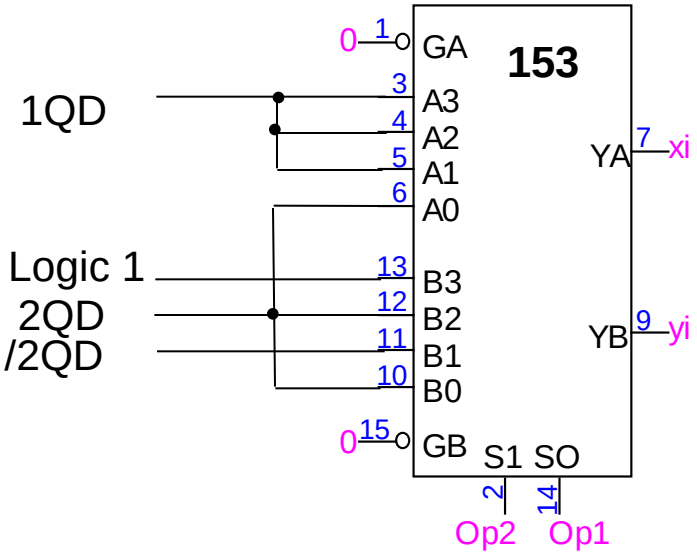
Construct the truth table for the input processor. (Use a_i, b_i, 0, 1)

Op_2 Op_1	x_i	y_i	c_0
00	b_i	b_i	1
01	a_i	b_i'	1
10	a_i	b_i	0
11	a_i	1	0

Label the data inputs of the two 4-to-1 multiplexers given below for the realization of x_i, y_i.

Use the following signal names for input processor 1QD for ai, 2QD for bi, Op2, Op1, xi, yi, zi

Use the following signal names for control signals 1s1, 1s0, 2s1, 2s0, c0, C2, C1



Express the initial carry c_0 as a function of Op_2 and Op_1.

$c_0 = Op_2'$

Construct the truth table for the control circuit.

State	START	$(s_1)_{R1}$	$(s_0)_{R1}$	$(s_1)_{R2}$	$(s_0)_{R2}$	C_2	C_1
T_0	0						
T_0	1						
T_1	d						
T_2	d						
T_3	d						
T_4	d						
T_5	d						

$(s_1)_{R1}$ =
$(s_0)_{R1}$ =
$(s_1)_{R2}$ =
$(s_0)_{R2}$ =
C_2 =
C_1 =

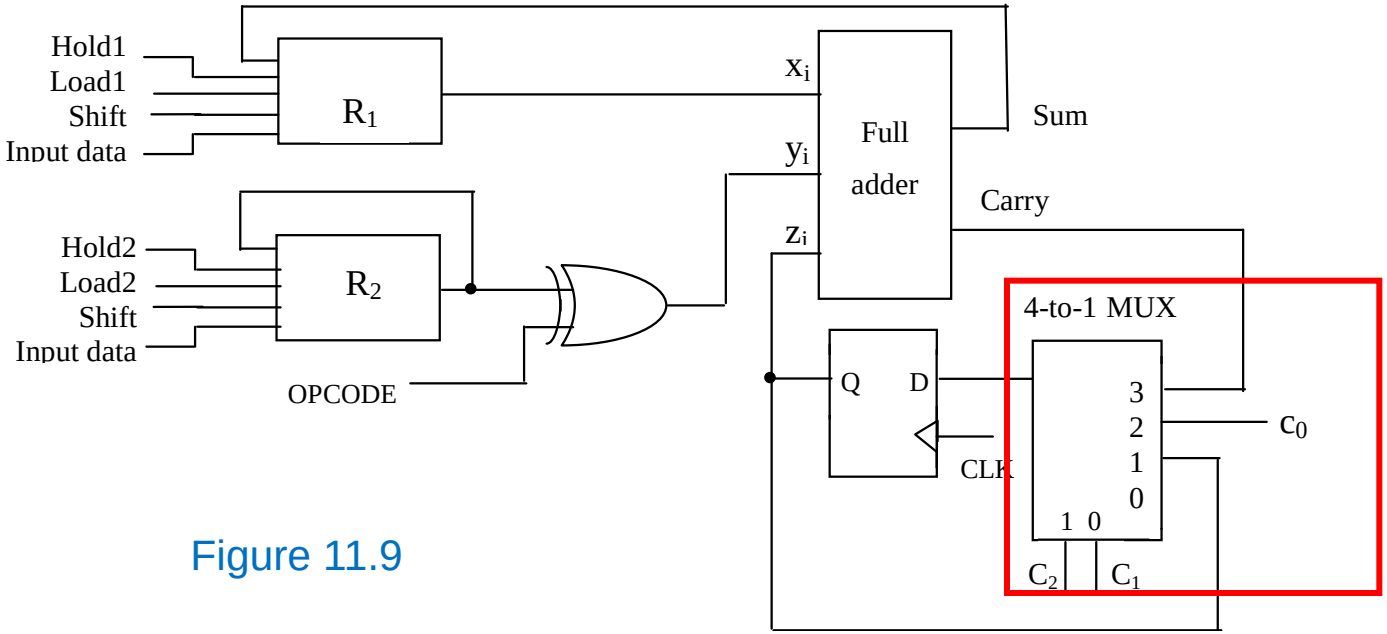
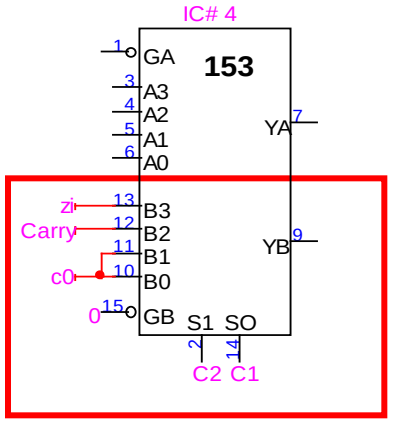
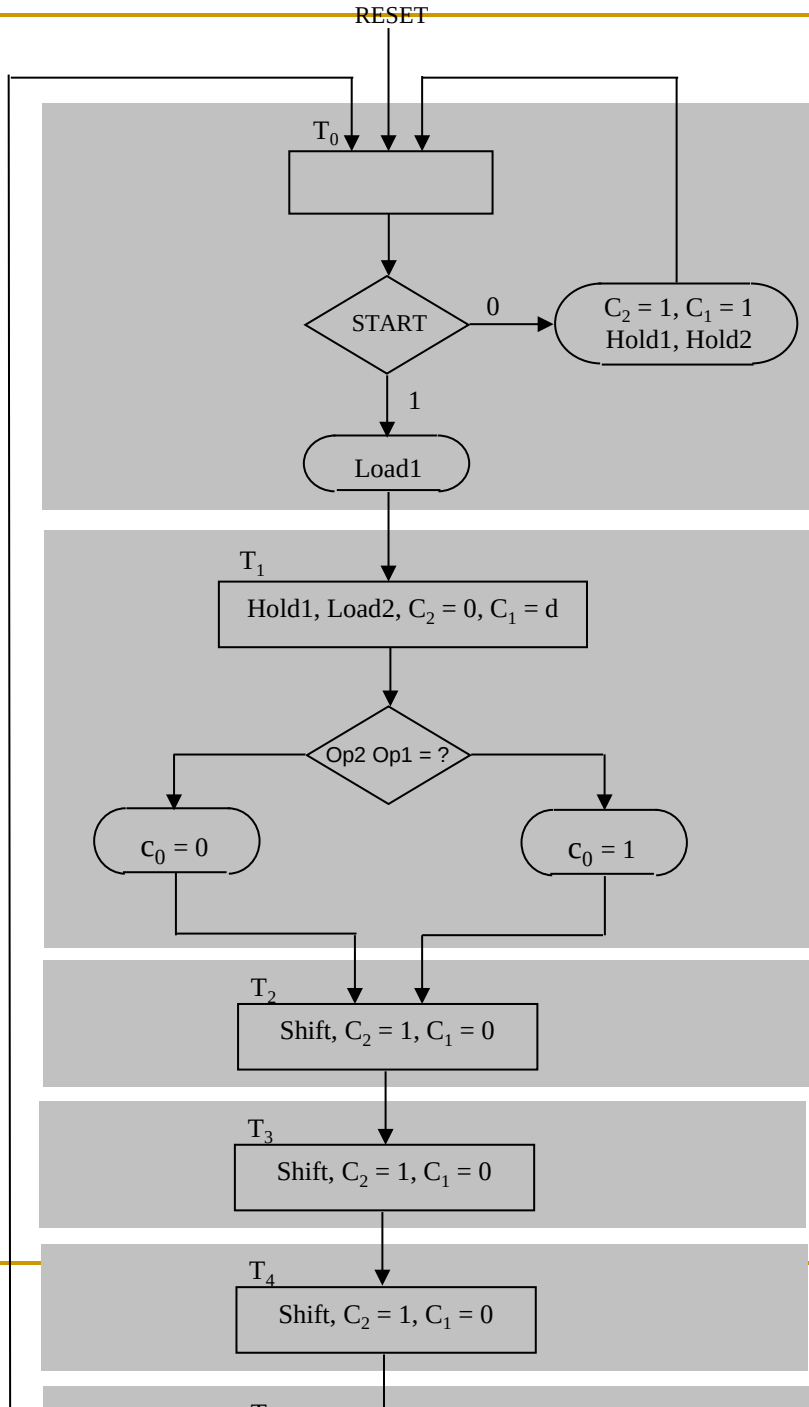


Figure 11.9

Note the difference in the data inputs of the two multiplexers.

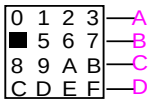
Lab 5 schematic diagram

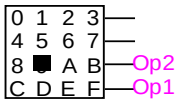


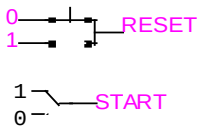

RESET
T0
START
0
C2 = 1, C1 = 1
Hold1, Hold2
1
Load1
T1
Hold1, Load2, C2 = 0, C1 = d
Op2 Op1 = ?
c0 = 0
c0 = 1
T2
Shift, C2 = 1, C1 = 0
T3
Shift, C2 = 1, C1 = 0
T4
Shift, C2 = 1, C1 = 0

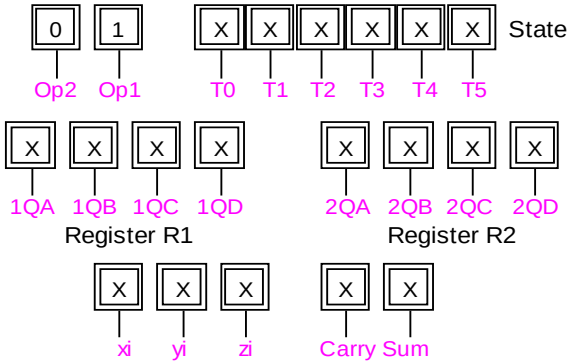
16.265 Logic Design
Experiment 5: Arithmetic Processor
Name:
Student Logic Number:
Function Set Number:

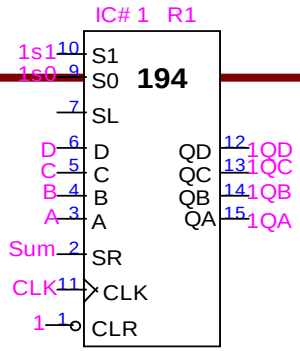
4-bit data
inpput
ABCD

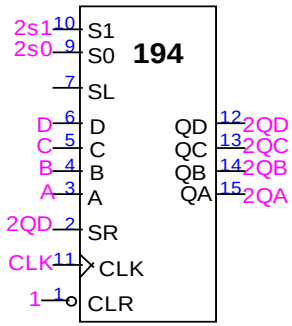


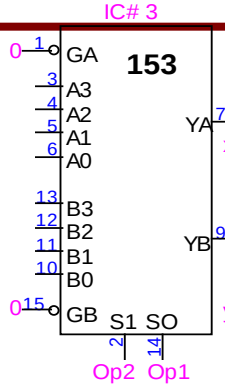


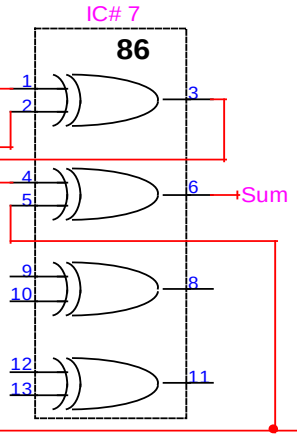


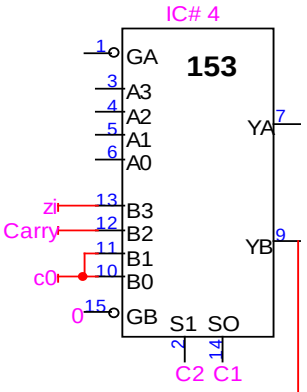


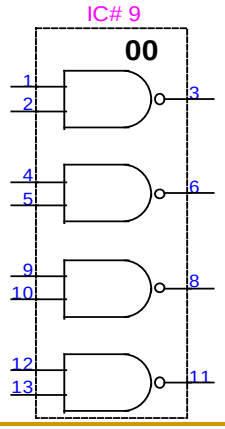


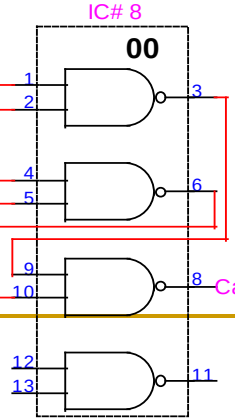


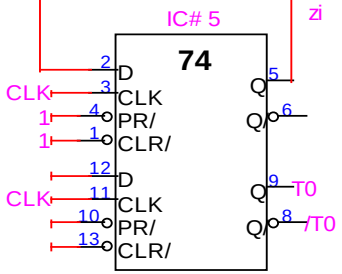






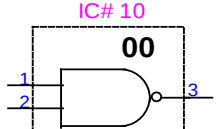


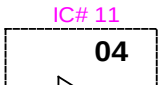


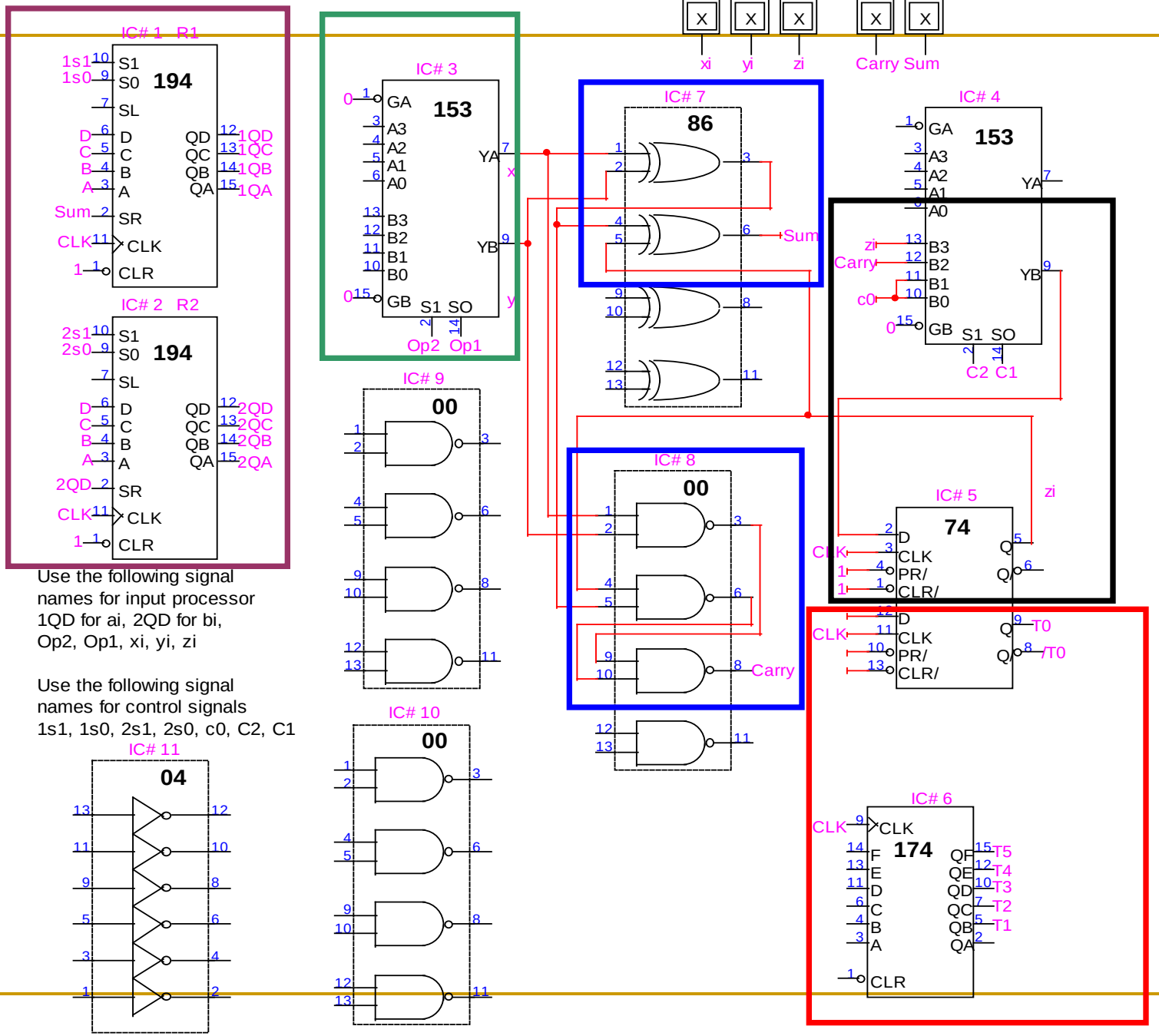


Use the following signal names for input processor 1QD for ai, 2QD for bi, Op2, Op1, xi, yi, zi

Use the following signal names for control signals 1s1, 1s0, 2s1, 2s0, c0, C2, C1






IC# 1 R1
194
1s1
1s0
S1
S0
SL
D
C
B
A
SR
CLK
CLR
Sum
1QD
1QC
1QB
1QA
IC# 2 R2
194
2s1
2s0
2QD
2QC
2QB
2QA
IC# 3
153
GA
A3
A2
A1
A0
B3
B2
B1
B0
GB
YA
YB
S1 SO
Op2 Op1
x
y
xi
yi
zi
Carry
Sum
IC# 7
86
Sum
IC# 4
153
C2 C1
c0
IC# 9
00
IC# 8
00
Carry
IC# 10
00
IC# 11
04
IC# 5
74
IC# 6
174
T0
/T0
T1
T2
T3
T4
T5
Use the following signal names for input processor 1QD for ai, 2QD for bi, Op2, Op1, xi, yi, zi
Use the following signal names for control signals 1s1, 1s0, 2s1, 2s0, c0, C2, C1