Chapter 13

Asynchronous Sequential Network

13.0 Introduction

Asynchronous sequential circuit is specified by a timing sequence of inputs, outputs, and internal states. The change of internal state occurs in response to the synchronized clock pulse. Asynchronous sequential circuit does not use clock pulses. The memory element of the circuit is unclocking flip-flop or time-delay elements. The block diagram of an asynchronous sequential circuit is shown in Fig. 13.1.

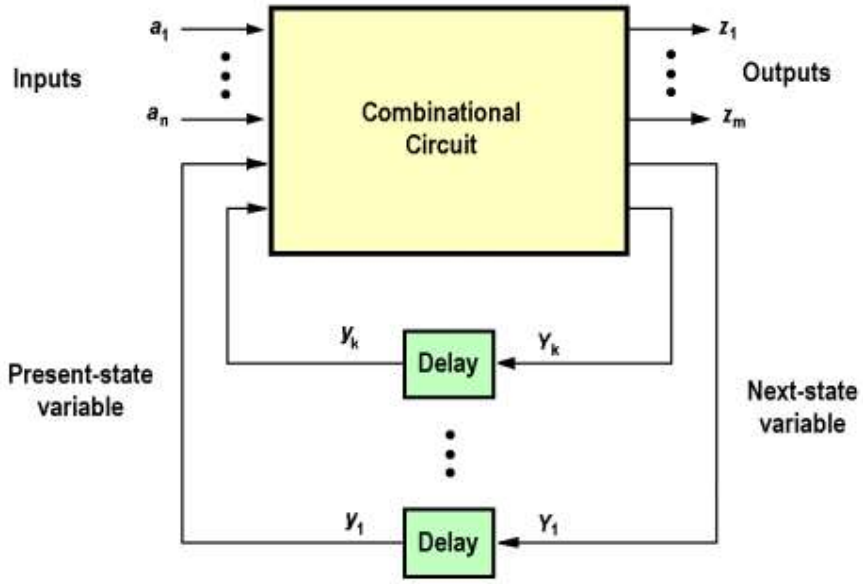


Figure 13.1: The block diagram of an asynchronous sequential circuit

The feedback would not have immediate response because of the propagation delay in the combinational circuit. After this time lapse, the output will only respond. Owing to the delay, there is unstable state associated with, race issue, and hazard issue, which shall be dealt with.

Asynchronous sequential circuit is faster and more difficult to design than synchronous sequential circuit. Asynchronous sequential circuit does not use clock pulses. The change of internal state occurs when there is a change in the input variables. The circuit is more difficult to design because of the timing problem involved in the feedback path.

Asynchronous sequential circuit is useful in a variety of application especially for the case where speed of operation is important. In the application where input signal can be changed at any time such as the communication between two units, the design must be done with asynchronous circuits.

The circuit is also less expensive to design because there is no requirement to design the clock pulse generation circuit.

13.1 Analysis Procedure

The analysis of asynchronous sequential circuits consists of obtaining a table or diagram that describes the sequence of internal states and outputs as a function of changes in the input variables. The analysis covers obtaining the transition table, flow table, race condition, and stability considerations.

13.1.1 Transition Table

The procedure for obtaining a transition table from an asynchronous circuit is as follow.

- Determine all feedback loops in the circuit.
- Designate the output of each feedback loop with variable Y_i and its corresponding input with y_i for i = 1, 2,….., k, where k is the number of feedback loops in the circuit.
- Derive the Boolean functions of all Y's as a function of the external inputs and the y's.
- Plot each Y function in a map using y variables for the row and the external inputs for the column.
- Combine all maps into one table showing the values of $Y = Y_1Y_2\ldots Y_k$ inside each square.
- Once the transition is ready, the behavior of the circuit can be analyzed by observation for the state transition as a function of changes in the input variables.

Consider an asynchronous circuit shown in Fig. 13.1, the Boolean functions of the output Y_1 and Y_2 are respectively equal to

$$Y_1 = x \cdot y_1 + \bar{x} \cdot y_2 \qquad (13.1)$$

$$Y_2 = x \cdot \overline{y_1} + \bar{x} \cdot y_2 \qquad (13.2)$$

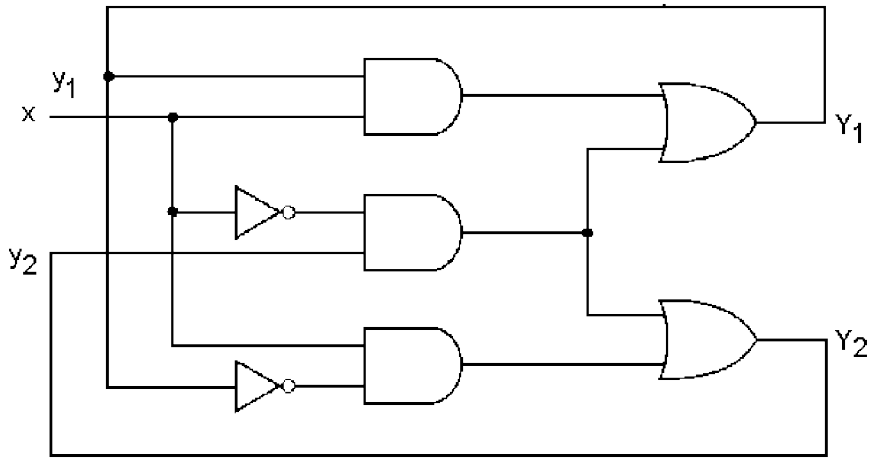


Figure 13.1: An asynchronous sequential circuit

The transition tables are shown in Fig. 13.2.

x / y_1y_2	0	1
00	0	0
01	1	0
11	1	1
10	0	1

(a) Map for $Y_1 = x \cdot y_1 + \bar{x} \cdot y_2$

x / y_1y_2	0	1
00	0	1
01	1	1
11	1	0
10	0	0

(b) Map for $Y_2 = x \cdot \bar{y_1} + \bar{x} \cdot y_2$

x / y_1y_2	0	1
00	(00)	01
01	11	(01)
11	(11)	10
10	00	(10)

(c) Transition table

Figure 13.2: Map and transition table of the circuit shown in Fig. 13.1

The transition table shown in Fig. 13.2(c) the value of $Y=Y_1Y_2$. The circled transitions are stable transition since the input y_1y_2 is the same. The non circled transition is not stable the output is not same as y_1y_2. The change of input from $=y_1y_2x = 000$ to $y_1y_2x = 001$ will yield output $Y = 01$, which is not 00. Thus, this transition is not stable.

The state table for the circuit is shown in Fig. 13.3.

Present State		Next State			
		x = 0		x = 1	
0	0	0	0	0	1
0	1	1	1	0	1
1	1	0	0	1	0
1	0	1	1	1	0

Figure 13.3: State table for circuit shown in Fig. 13.1

13.1.2 Flow Table

During the design of asynchronous sequential circuit, it is more convenient to name the states by letter symbols without making specific reference to their binary values. Thus, using this approach the flow table of Fig. 13.2(c) is shown in Fig. 13.4 after assign 00 = a, 01 = b. 11 = c, 10 = d.

y_1y_2 \ x	0	1
a	(a)	b
b	c	(b)
c	(c)	d
d	a	(d)

Figure 13.4: Flow table shown in letter symbol

Consider a two-state with two inputs and one output flow table shown in Fig. 13.5. The states shown in red color are the stable state.

y \ x_1,x_2	00	01	11	10
a	(a), 0	(a), 0	(a), 0	b, 0
b	a, 0	a, 0	(b), 1	(b), 0

Figure 13.5: A two-state with two inputs and one output flow table

The transition table and the map of the output are shown in Fig. 13.6.

y \ x_1,x_2	00	01	11	10
a	0	0	0	1
b	0	0	1	1

(a) Transition table for $Y = x_1 \cdot \overline{x_2} + x_1 \cdot y$

y \ x_1,x_2	00	01	11	10
0	0	0	0	0
1	0	0	1	0

(b) Map for $z = x_1 \cdot x_2 \cdot y$

Figure 13.6: The transition table and map of the output of flow table shown in Fig. 13.5

The circuit of the design based on the transition table and map shown in Fig. 13.7.

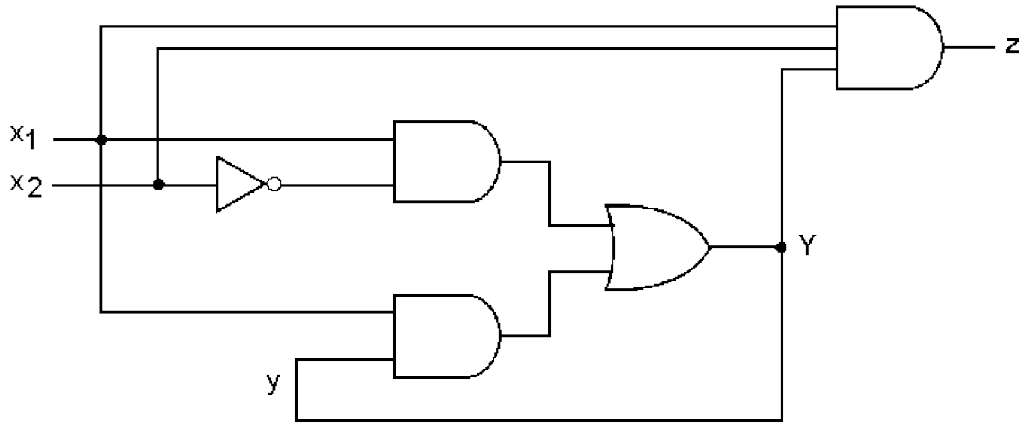


Figure 13.7: Circuit design based on the transition table and map shown in Fig. 13.6

13.1.3 Race Condition

A race condition is said to be exist in asynchronous sequential circuit when two or more binary state variables change value in response to change in an input variable. When unequal delays are encountered, a race condition may cause the state variables to change in an unpredictable manner. For an example state variables change from 00 to 11, due to the difference in delay, the change may take place in two sequences. If the first variable changes faster than the second variable then the sequence is from 00→10→11. If the second variable changes faster than the first one then the sequence is 00→01→11. If the final state does not depend on the sequence taken place then the race is called a *noncritical race*. If it is possible to end with more than one state depending on the order of the state change then it is a *critical race*, which must be avoided. The transition table shown in Fig. 13.8 is a noncritical type.

y_1y_2 \ x	0	1
00	(00)	11
01		11
11		(11)
10		11

Figure 13.8: A noncritical race transition

If the initial state is $y_1y_2x = 000$, by changing the external input from x = 0 to 1, the state will change to 111. Depending on the delay, the sequence of transition can be from 00→11, 00→01→11 or 00→10→11.

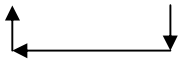
The transition table shown in Fig. 13.9 illustrates race condition. We start with state $y_1y_2x = 000$ and change the state to $y_1y_2x = 001$. It would mean the final state that would like to be is $y_1y_2 = 11$. Depending on the delay time, it can end with different state. If first variable is responding faster than the second variable, the state will end with $y_1y_2 = 10$, which is a stable state. If second variable is responding faster than the first variable, the state will end with $y_1y_2 = 01$, which is also a stable state. If both first and second variables respond simultaneously the state will end with $y_1y_2 = 11$, which is also a stable state. The conclusion means that the race condition can cause the final state to be ended with three possible states.

y_1y_2 \ x	0	1
00	(00)	11
01		(01)
11		(11)
10		(10)

Figure 13.9: A critical race transition

Race condition can be dealt with by proper binary assignment to the state variable. The state variables must be assigned binary number in such a way that only one state variable can be change at any one time when a state transition occurs in the flow table. We shall discuss the method in Section 13.5.

Figure 13.10 shows a cycle transition. We start with state $y_1y_2x = 000$ and change the state to $y_1y_2x = 001$. The state is a cycle transition from 00→01→11→10.



y_1y_2 \ x	0	1
00	(00)	01
01		11
11		10
10		01

Figure 13.10 A cycle transition

13.1.4 Stability Consideration

Owing to the feedback connection that exists in synchronous sequential circuits, care must be taken to ensure that circuit does not become unstable. An unstable condition will cause the circuit to oscillate between unstable states. The transition table method of analysis can be useful in detecting the occurrence of instability.

An unstable circuit is shown in Fig. 13.11. This circuit has function $Y = \overline{X_1} \cdot X_2 + X_2 \cdot y$.

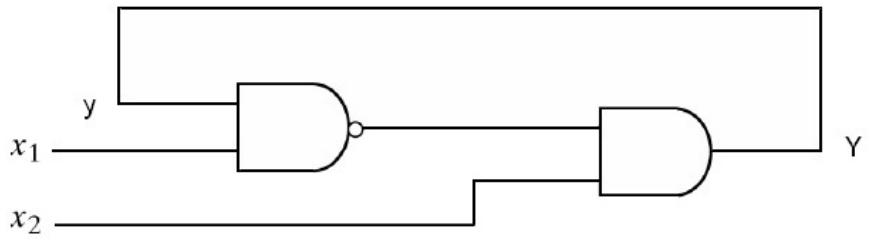


Figure 13.11: An unstable logic circuit

The transition table is shown in Fig. 13.12. The cycled values are value that Y = y. The uncycled values represent unstable state. Example of the unstable stable is when $yx_1x_2 = 111$ which should yield y = 0.

y \ x_1x_2	00	01	11	10
0	(0)	1	1	(0)
1	0	(1)	0	0

Figure 13.12: The transition table of circuit shown in Fig. 13.11

13.2 Analysis of Asynchronous Circuits

The most common asynchronous circuit is the SR flip-flop shown in Fig. 13.13.

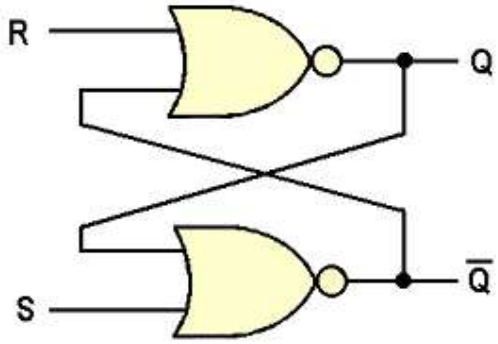


Figure 13.13: A SR flip-flop

If the SR flip-flop is redrawn according to the model shown in Fig. 13.13 i.e. including the propagation delay, the SR flip-flop shall be as shown in Fig. 13.14.

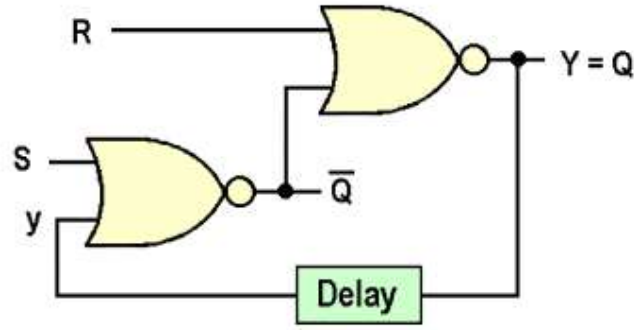


Figure 13.14: The model of an SR flip-flop with delay

The output Y is not immediately soon after the input y change because there is a propagation delay between two NOR gates. The Boolean function of output Y is

$$Y = \overline{\overline{S + y} + R} = (S + y) \bullet \overline{R} \qquad (13.3)$$

Based on equation (13.4), the flow table of the flip-flop is derived and shown in Fig. 13.15.

Present State	Next State				Output
	SR Value				
	00	01	10	11	
y	Y				Q
0	(0)	(0)	1	(0)	0
1	(1)	0	(1)	0	1

Figure 13.15: The flow table of SR flip-flop

Those next state values that marked in cycle are stable state. If the state y = 0 is assigned as A and y = 1 is assigned as B then the state assignment table of the SR flip-flop is shown in Fig. 13.16.

Present State	Next State				Output
	SR Value				
	00	01	10	11	
y	Y				Q
A	(A)	(A)	B	(A)	0
B	(B)	A	(B)	A	1

Figure 13.16: State assignment table of SR flip-flop

At y = 0 state, changing SR value 11 to 10 will cause Y = B. After the propagation delay, the input shall be ySR = 110, in which the output Y shall be settled at B. At y =1 state, changing SR value from 10 to 11 will cause Y = A. After the propagation delay, the input shall be ySR = 011.

Based on the flow table, the flow diagram of SR flip-flip is shown in Fig. 13.17.

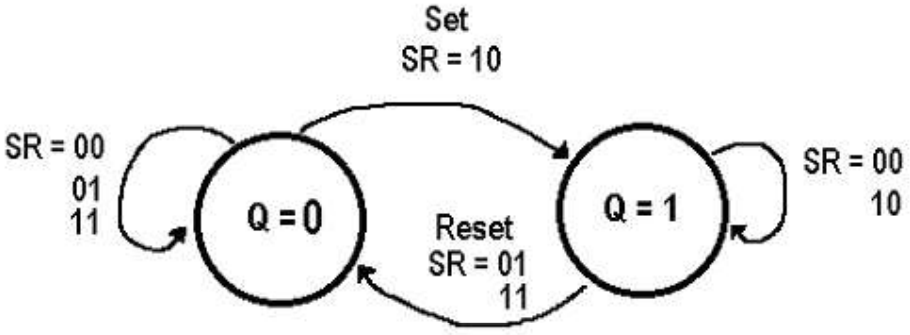


Figure 13.17: The flow diagram of SR flip-flop

Based on the results of state table shown in Fig. 13.16, the Mealy model flow diagram is shown in Fig. 13.18.

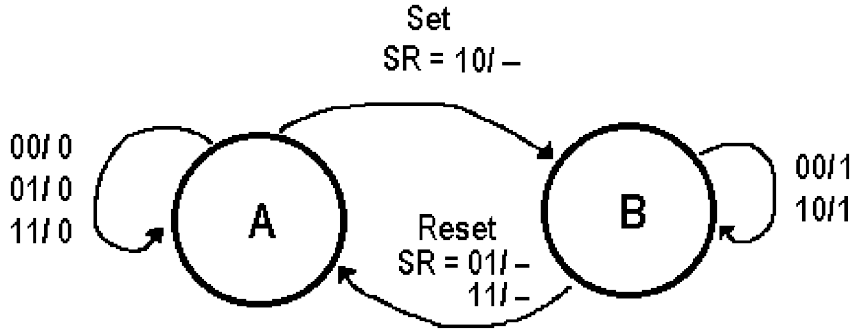


Figure 13.18: Mealy model flow diagram of SR flip-flop

Let's look at a gated D flip-flop shown in Fig. 13.18. The Boolean function of the output Y is $Y = (C \bullet D) \bullet ((C \bullet \overline{D}) \bullet y) = CD + \overline{C} \bullet y + D \bullet y = CD + \overline{C} \bullet y$, after delete the redundant D•y that solve race condition known as *hazard* that will be dealt later on. Two gated-D flip-flop is used to implement the master-slave D flip-flop. Thus, its circuit diagram is shown in Fig. 13.20.

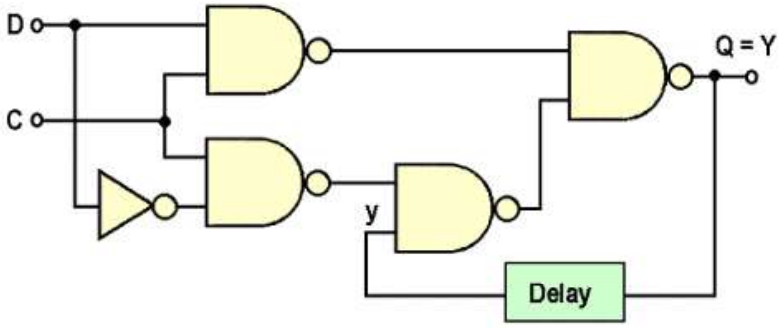


Figure 13.19: D flip-flop

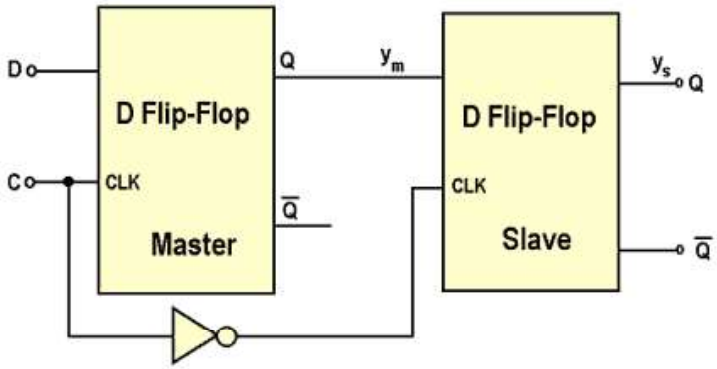


Figure 13.20: A master-slave D flip-flop

Boolean function of the master-slave D flip-flop shall be

$$Y_m = C \bullet D + \overline{C} \bullet y_m \tag{13.4}$$

and

$$Y_s = \overline{C} \bullet y_m + \overline{C} \bullet y_s \tag{13.5}$$

The flow table of master-slave D flip-flop is shown in Fig. 13.21.

Present State	Next State				Output
	CD Value				
	00	01	10	11	
$y_m y_s$	$Y_m Y_s$				Q
00	(00)	(00)	(00)	10	0
01	00	00	(01)	11	1
10	11	11	00	(10)	0
11	(11)	(11)	01	(11)	1

Figure 13.21: The flow table of master-slave D flip-flop

If state assignment is made such that S1 = 00, S2 = 01, S3 = 10, and S4 = 11, the state assignment table including the unspecified entries shown as "-" shown in Fig. 13.22.

Present State	Next State				Output
	CD Value				
	00	01	10	11	
$y_m y_s$	$Y_m Y_s$				Q
S1	(S1)	(S1)	(S1)	S3	0
S2	S1	-	(S2)	S4	1
S3	-	S4	S1	(S3)	0
S4	(S4)	(S4)	S2	(S4)	1

Figure 13.22: State assignment table including unspecified entries of master-slave D flip-flop

The unspecified entries are derived based on the fact that the input must change one at a time. Thus, the input changes from CD = 10 to CD = 01 for $y_m y_s$ = 01 is unspecified. Similarly, the input changes from CD = 11 to CD = 01 for $y_m y_s$ = 10 is unspecified.

13.3 Designing Asynchronous Network Circuits

The way to design the asynchronous network circuit is same as how the synchronous network circuit is designed. The procedure listed below shall be used when designing the asynchronous network circuit.

- Devise a state diagram for an FSM that realizes the required functional behavior.
- Derive the flow table and reduce the number of possible state.
- Perform the state assignment and derive the excitation table.
- Obtain the next state and output expressions.
- Construct a circuit that implements these expressions.

Let's use the design of an arbiter illustration. An arbiter is a hardware that would decide the request from the digital system to utilize the shared resource such as a printer connected to two computer systems. If there is a request from computer 1 to use the printer, the arbiter will process the request and acknowledge the request if the printer is free. If the printer is busy printing the computer 2's request, then the arbiter would not assign the printer to computer 1 until printer has printed computer 2's request and computer 2 returns the end of

print signal to the arbiter. The flow diagram of the arbiter is shown in Fig. 13.23.

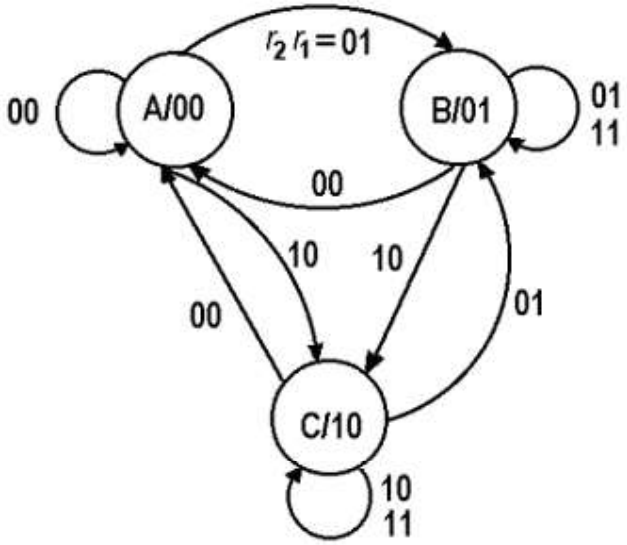


Figure 13.23: The flow diagram of an arbiter

r_1r_2 = 00 shall mean there is no request. r_1r_2 = 01 shall mean B is requesting and r_1r_2 = 10 shall mean C is requesting. Output g_2g_1 are the acknowledged codes of the arbiter.

The state assignment table of the arbiter is shown in Fig. 13.24.

Present State	Next State				Output
	r_1r_2 Value				
	00	01	10	11	
y_2y_1	Y_2Y_1				g_1g_2
A	(A)	B	C	-	00
B	A	(B)	C	(B)	11
C	A	B	(C)	(C)	10
D	-	B	C	-	*dd*

Figure 13.24: The state assignment table of the arbiter

For circuit stabilized at B, changing from r_2r_1 from 11 to 10 should end with Y_2Y_1 = C, which is also meant that the value of y_2y_1 should be changing to 10. If y_2 changes faster y_1, then y_2y_1 should be 11, which is the D state. Since D state has value Y_2Y_1 is 10, it will move to the stable C state. If y_1 changes faster than y_2, y_2y_1 shall end with 00 state, which is a stable A state. Thus, there is race issue here, which is a *critical race* since its final state depends on propagation delay. Similarly, from stable C state change r_1r_2 from 11 to 01 will have the

same race issue like the earlier case. In order to avoid these critical races, one needs to change the states to state A marked in red to avoiding critical racing. The modified state assignment table is shown in Fig. 13.25.

Present State	Next State				Output
	r_1r_2 Value				
	00	01	10	11	
y_2y_1	Y_2Y_1				g_1g_2
A	(A)	B	C	-	00
B	A	(B)	A	(B)	11
C	A	A	(C)	(C)	10
D	-	B	C	-	*dd*

Figure 13.25: State assignment table of an arbiter after avoiding critical race

The next state Boolean function for Y_1 and Y_2 are $Y_1 = r_1 \bullet \overline{y_2}$ and $Y_2 = \overline{r_1} \bullet r_2 \bullet \overline{y_1} + r_2 \bullet y_2$. The logic circuit of the arbiter is shown in Fig. 13.26. This circuit is simpler than the circuit designed with flip-flop and clock pulse circuitry. The circuit can respond almost instantly and does not wait for clocking pulse to respond like the case where it is designed using synchronous approach.

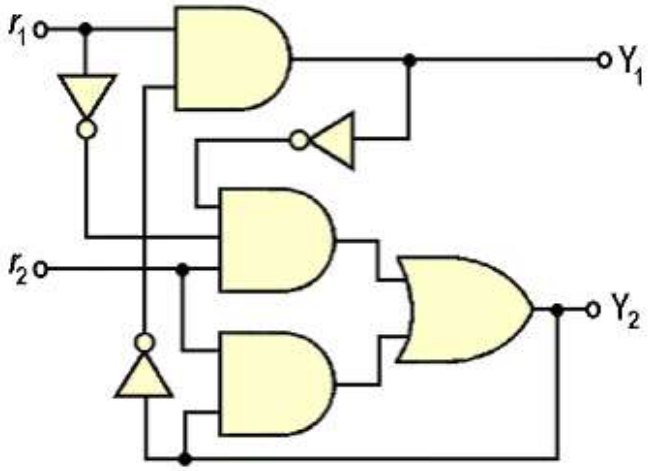


Figure 13.26: The logic circuit of an arbiter

13.4 State Reduction

The procedure for reducing the number of internal states in an asynchronous sequential network resembles the procedure that used for synchronous network