Logic design — 4

Causes of malfunction in event-driven circuits

by B. Holdsworth* and D. Zissos†

*Chelsea College, University of London †Department of Computing Science, University of Calgary, Canada.

In the last article, the procedure needed for the design of event-driven logic circuits was discussed. This second half of that article goes on to describe the causes of misoperation in such circuits and concludes with some examples of design. It is unfortunate that some of the diagrams concerned with this half of the article appeared in the first half — for this, we apologise.

Races between primary signals. The circuit shown in Fig. 11 is required to operate three lamps L_1, L_2, and L_3, according to the following specifications.

(1) Lamp L_1 is to turn-on when both X and Y are operated, but only if switch X is operated before switch Y.

(2) Lamp L_2 is to turn-on when both input switches are operated simultaneously.

(3) Lamp L_3 is to turn-on when both X and Y are operated, but only if switch Y is operated first.

In practice, a logic circuit responds with different speeds to changes in the input signals. Hence the response time of the circuit to a change in the input signal X must be assumed to be different from the response time to a change in Y. As a consequence the circuit, instead of assuming state S_3 on leaving state S_0, either assumes state S_2, if the circuit responds to the change in X first, or alternatively it enters state S_5, if the circuit responds to a change in Y first. In both cases the circuit operation is not according to specification.

Since there is no remedy to this problem the circuit constraint applied is that only one input signal is allowed to change at a time.

Races between secondary signals. In the internal state diagram shown in Fig. 12(a), the coding of the internal states is such that circuit transitions S_0 to S_1 and S_2 to S_3 involve the change of more than one secondary signal. In practice because of variations in the response times of the two secondary signals to a change in the input signal X from 0 to 1, either A or B will change first.

Assuming that A changes first the circuit, when it leaves S_0, first enters S_2. From state S_2, because X = 1, the circuit assumes state S_3 instead of S_1, and this a stable state for X = 1. This is clearly incorrect operation of the circuit. Obviously a similar analysis of the circuit operation can be performed for the case when B changes faster than A.

The solution to this problem is to ensure that each circuit transition involves the change of one secondary signal only and a race-free assignment of the state variables should be used as described earlier in this article and as shown in Fig. 12(b).

Races between primary and secondary signals. A circuit implementation of Fig. 12(b) is shown in block schematic form in Fig. 13. The letters a and b are assigned to the two sections of the circuit which generate the secondary signals A and B.

Consider the transition from S_0 to S_1 in Fig. 12(b). This transition will take place in the time t_s, which it takes to turn-on the secondary signal B. It will also be assumed that the time taken to invert the primary signal X is t_p. If $t_p > t_s$ the following sequence of events will take place.

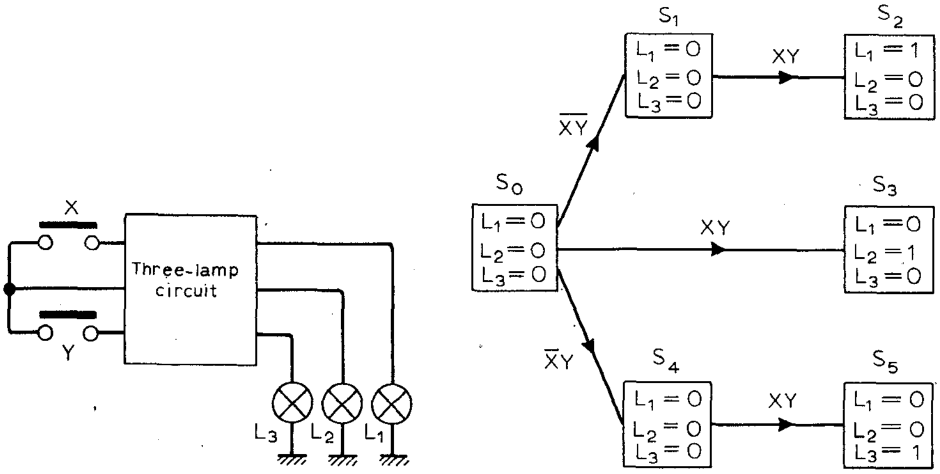
(1) At time t_s, B changes to 1 and the circuit assumes state S_1.

(2) Since $t_p > t_s$, $\overline{X} = 1$, and the condition for turning A on exists.

(3) A turns on causing the circuit to move to state S_2.

(4) On assuming state S_2, the circuit

Fig. 11. Three-lamp circuit and its state diagram.



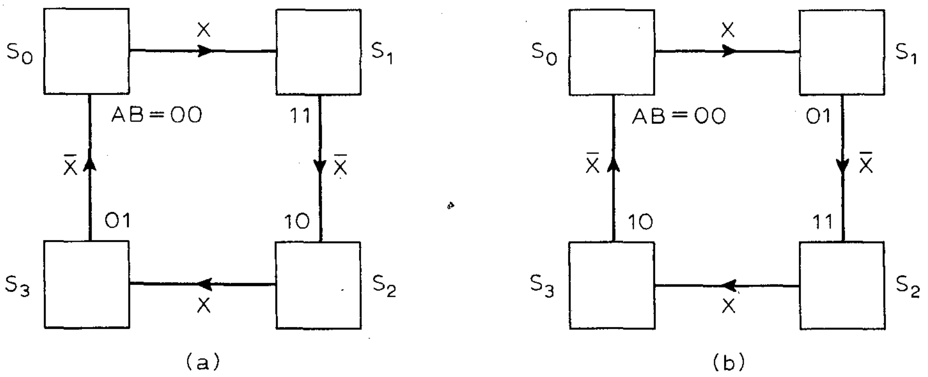


Fig. 12. Elimination of races between secondary signals.

moves to state S_3, since $X=1$.

If $t_p<t_s$ on assuming state S_1 the input signal to section a has already changed, i.e. $\overline{X}=0$, and the circuit remains in state S_1.

Unlike the previous two cases, elimination of races between primary and secondary signals cannot be achieved, since a change in a primary signal initiates a change in a secondary signal. Therefore to avoid circuit misoperation it is necessary to ensure that $t_p \leq t_s$. It follows that incorrect circuit behaviour will not occur if the maximum delay associated with a primary signal t_{pmax}, is less than the minimum delay associated with a secondary signal t_{smin}.

Hence

$$\frac{t_{pmax}}{t_{smin}} \leq 1$$

The 33⅓% property

The sequential circuits designed with the aid of the sequential equations are hazard-free when implemented with gates whose maximum speed tolerance is ±33⅓%. The justification for this statement is as follows.

The maximum delay by which a primary signal in primitive sequential circuits can be delayed is one gate delay, t_g, when it has to be inverted. Allowing

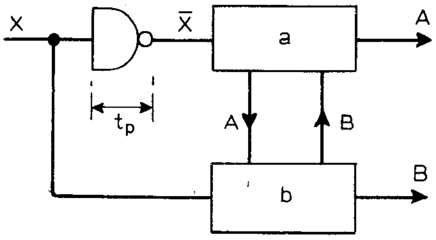


Fig. 13. Races between primary and secondary signals.

x% variation due to production spread, loading etc. $t_{pmax=t_g}(1+x)$.

The minimum delay associated with a secondary signal is $2t_g$, since at least two levels of switching are involved, as an examination of the NAND sequential equation $Q=S+\overline{R}Q$ will show. Allowing x% variation, $t_{smin}=2t_g(1-x)$.

Substituting these values in the equation developed in the last section gives $t_g(1+x)/2t_g(1-x) \leq 1$ for correct circuit behaviour. The reader should observe that this property is valid for circuits in which the sequential equations are implemented in their primitive form. Algebraic manipulation of the sequential equations will lead to a modification of the relative delays of the primary and secondary signals and therefore invalidate the 33⅓% property. Hence, processing of the sequential equations is not advised.

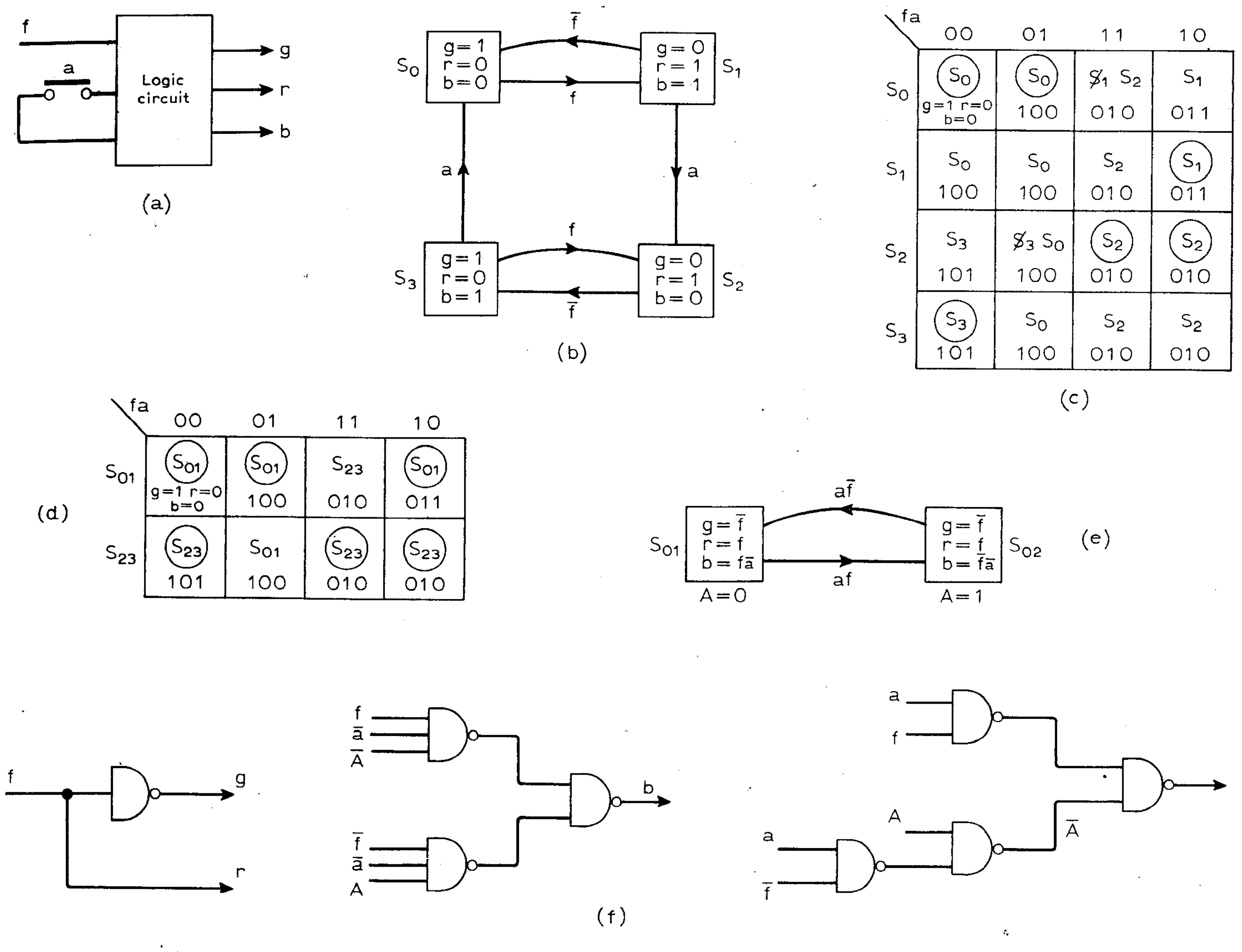
Design steps

Step 1. Draw a block diagram showing the available input signals and the required output signals.

Step 2. Draw a state diagram describing the internal performance of the circuit.

Step 3. This step is optional and can be omitted. Its purpose is to provide the designer with a means of reducing the number of internal states obtained in Step 2, if such a reduction is possible or desirable.

Step 4. With the aid of a race-free diagram if necessary, each internal state is given a unique code. From the coded state diagram the turn-on and turn-off sets for the secondary signals are obtained and these are used to derive the primitive sequential equations. Expressions are also obtained for the output signals. The implementation of these equations is the required circuit.



(c)

fa	00	01	11	10
S_0	S_0 g=1 r=0 b=0	S_0 100	$\cancel{S_1}$ S_2 010	S_1 011
S_1	S_0 100	S_0 100	S_2 010	S_1 011
S_2	S_3 101	$\cancel{S_3}$ S_0 100	S_2 010	S_2 010
S_3	S_3 101	S_0 100	S_2 010	S_2 010

(d)

fa	00	01	11	10
S_{01}	S_{01} g=1 r=0 b=0	S_{01} 100	S_{23} 010	S_{01} 011
S_{23}	S_{23} 101	S_{01} 100	S_{23} 010	S_{23} 010

Fig. 14. Function to be realized in Example 1 is at (a) and its state diagram is at (b), while the state table is shown in (c) and in merged form at (d). Initial state diagram based on (d) is shown at (e) and realization of the circuit is (f) Output of r.h. circuit is a.

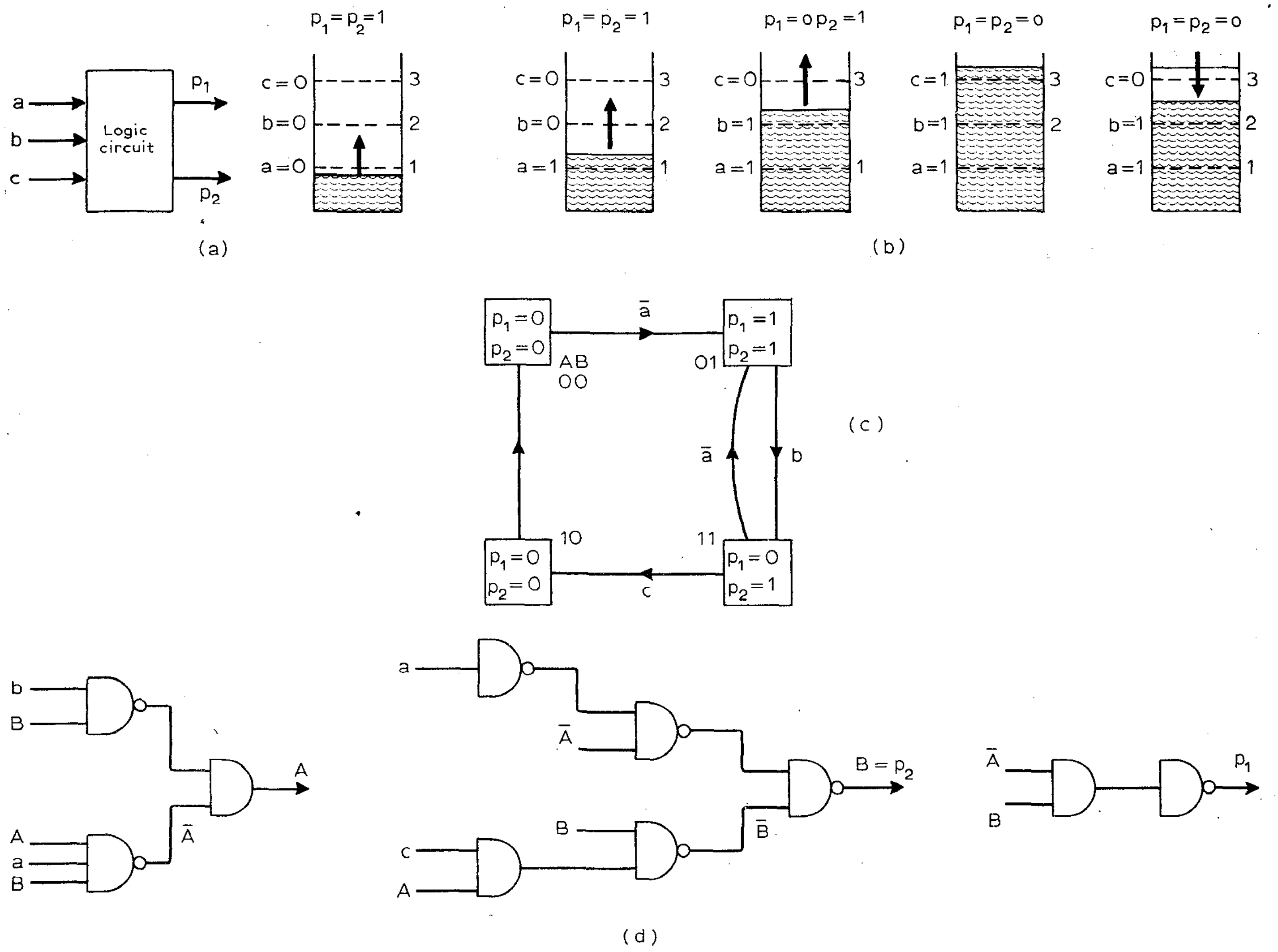


Fig. 15. Function required by Example 2 is seen in (a) and (b). State diagram in (c) provides turn-on and turn-off sets for use in NAND realization. Circuit is shown in (d).

The design procedure will now be applied to the solution of two problems.

Example 1

Design a fault detector with the following terminal characteristics. The appearance of a fault signal f activates an alarm bell, turns a green light off and a red light on. The operator turns off the bell by pressing an acknowledge button a. When the fault is cleared, the red light turns off, the green light turns on and the bell is reactivated to attract the operator's attention. The bell is turned off when the operator presses the acknowledge button. Should the fault clear before the operator has responded, the circuit is to reset. Also if a fault reappears before the operator has responded the green light turns off, the red light turns on and the bell turns off.

Step 1. See Figs. 15(a) and (b)

Step 2. A suitable state diagram is shown in Fig. 15(c).

Step 3. The state table corresponding to Fig. 14(b) is shown in Fig. 14(c).

Applying Caldwell's merging rules to the state table in Fig. 14(c), states S_0 and S_1 can be merged to form state S_{01} and states S_2 and S_3 can be merged to form state S_{23}. The reduced state table is shown in Fig. 14(d).

The internal state diagram based on the reduced state table is shown in Fig. 14(e).

Step 4. By direct reference to Fig. 14(e) the turn-on and turn-off sets are: Turn-on set of A = af. Turn-off set of A = a$\bar{f}$. Therefore the NAND circuit equation for A is

$$A = af + A\overline{a\bar{f}}$$
$$A = af + A(\bar{a} + f)$$
$$g = \bar{f}$$
$$r = f$$
$$b = f\bar{A}\bar{a} + \bar{f}A\bar{a}$$

The corresponding circuit is shown in Fig. 14(f).

Example 2

Water is pumped into a water tower by two pumps p_1 and p_2, where p_1 is an auxiliary pump used for boosting purposes. Both pumps are to turn on when the water goes below level 1 and are to remain on until the water reaches level 2, when pump p_1 turns off and remains off until the water is below level 1 again. Pump p_2 remains on until level 3 is reached when it also turns off and remains off until the water falls below level 1 again.

Level sensors are used to provide level detection signals as follows:

Signal $a = 1$ when the water is at or above level 1, otherwise $a = 0$.

Signal $b = 1$ when the water is at or above level 2, otherwise $b = 0$.

Signal $c = 1$ when the water is at or above level 3, otherwise $c = 0$.

Develop a sequential logic circuit to control the pumps p_1 and p_2 according to the specification given above.

Step 1. See Figs. 15(a) and (b)

Step 2. A suitable state diagram is shown in Fig. 15(c).

Step 3. It is left as an exercise for the reader to draw the state table and examine the possibility of state reduction.

Step 4. By direct reference to Fig. 15(c) the turn-on and turn-off sets are:

Turn-on set of A = bB

Turn-off set of A = $\bar{B} + \bar{a}B$

$= \bar{B} + \bar{a}$

Turn-on set of B = $\bar{a}\bar{A}$

Turn-off set of B = cA

Therefore the NAND circuit equations are:

$$A = bB + A(\overline{\bar{B} + \bar{a}})$$
$$= bB + AaB$$
$$B = \bar{a}\bar{A} + (\bar{c} + \bar{A})B$$
$$p_1 = \bar{A}B$$
$$p_2 = \bar{A}B + AB$$
$$= B$$

The corresponding circuit is shown in Fig. 15(d).

Article 5 of the series will be a discussion of clock-driven circuits.