AMENDMENT RECORD

Amendment Sheet No.	Initials	Date	Amendment Sheet No.	Initials	Date
1	VWW	2/4/73			
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					

TECHNICAL INSTRUCTION

GP.1

SWITCHING CIRCUITS AND LOGIC

CONTENTS

SECTION 1

FUNDAMENTALS OF SWITCHING CIRCUITS

Introduction

This Section provides an introduction to switching circuits and demonstrates how their operation can be defined in terms of a simple logical analysis. A glossary of standard symbols (derived from British Standard 3939) is given in Section 4.

Principles

Many switching circuits can be considered as combinations of series-connected and parallel-connected switches and it is useful to analyse the properties of these two simple forms of circuit.

Fig. 1.1. Series Circuit

Series Circuits

If we consider the two series-connected switches P and Q in Fig. 1.1 we can make the following statement:

> If switch P is closed and switch Q is closed, the circuit through the switching network is continuous.

This statement can be analysed into three separate parts:

(a) Switch P is closed.
(b) Switch Q is closed.
(c) The circuit through the switching network is continuous.

The complete statement can therefore be summarised in the form:

If (a) AND (b) then (c).

Each of the three statements is referred to as being *two-valued,* in that it can be only true or false. The validity of statement (c) is dependent upon the validity of statements (a) and (b), and this dependence is shown in Table 1 which sets out the values of (c) for all possible combinations of values of (a) and (b).

It is normal practice to represent a true statement or a closed switch, by the symbol 1, and a false statement, or an open switch, by the symbol 0.

Table 1

(a)	(b)	(c)
true	true	true
true	false	false
false	true	false
false	false	false

These two symbols are referred to as the two truth-values of the statement. A table which sets down the truth-values of a statement under various conditions of its component elements is referred to as a *truth table.* Table 2 shows Table 1 set out in terms of its truth values.

Table 2

(a)	(b)	(c)
1	1	1
1	0	0
0	1	0
0	0	0

In Table 1 the various values of (c) were deduced by direct inspection of the particular circuit configuration. However, the values of (c) can be calculated directly if the truth values of (a) and (b) are known.

Consider the simple algebraic expression

$$m \times n = t$$

where the variables m and n are each restricted to two values either 1 or 0. Applying the rules of conventional algebra to the equation, the values of t can be calculated for all possible combinations of values of the variables m and n. These values are set out in Table 3.

Table 3

m	n	$m \times n$ (= t)
1	1	1
1	0	0
0	1	0
0	0	0

Table 3 has precisely the same form as Table 2, from which it can be inferred that the truth value of (c) can be found by multiplying together the truth-values of its component statements (a) and (b); in other words the mathematical symbol of multiplication can be substituted for AND. Thus the statement

If (a) AND (b) then (c)

gives

(c) = (a) x (b) or (c) = (a) . (b)

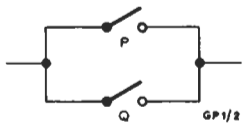


Fig. 1.2. Parallel Circuit

Parallel Circuits

Consider the two parallel-connected switches P and Q shown in Fig. 1.2. For this circuit we can make the statement:

> If switch P is closed OR switch Q is closed, the circuit through the switching network is continuous.

The elements of this statement are, as for the series circuit:

(a) Switch P is closed.
(b) Switch Q is closed.
(c) The circuit through the switching network is continuous.

This provides the general statement:

If (a) OR (b) then (c).

The associated series of truth values of (c) is given in Table 4.

Again the truth of (c) was deduced by direct reference to the circuit, but it can be calculated from a knowledge of the truth values of the component statements.

Table 4

(a)	(b)	(c)	or	(a)	(b)	(c)
true	true	true		1	1	1
true	false	true		1	0	1
false	true	true		0	1	1
false	false	false		0	0	0

Consider the simple algebraic expression:

$$m + n = t.$$

Applying the same conditions as in the previous example, the values for t for all possible combinations of m and n will be as in Table 5.

Table 5

m	n	$m + n\ (= t)$
1	1	1 + 1
1	0	1
0	1	1
0	0	0

The second, third, and fourth rows of Table 5 are identical to the corresponding rows of Table 4 which leads to the conclusion that the truth values of (c) in Table 4 can be found by adding together the truth values of the component statements (a) and (b). The apparent anomaly of the first row of Table 4 is explained by the fact that the circuit through the switching network in Fig. 1.2 is complete if switches P and Q are both closed; i.e., if (a) is true and (b) is true.

This can be summarised in terms of truth values as:

true + true = true,

or 1 + 1 = 1.

remembering that 1 is not a normal integer, but a truth value.

The general rule is, therefore, that in the statement:

If (a) OR (b) then (c),

the value of (c) can be found by adding together the truth values of the component statements (a) and (b); in other words, the mathematical symbol for addition can be substituted for OR giving:

$$(c) = (a) + (b)$$

The Concept of Negation

In circuit analysis it is often useful to be able to express the idea of negation. Consider the statement:

(a) Switch P is closed.

The negation of this statement is:

(b) Switch P is NOT closed.

These two statements are said to be complementary and the negated statement is denoted by a line over it thus:

$$\text{NOT (a)} = (\bar{a}) = \text{(b)}.$$

This leads immediately to:

$$\text{NOT true} = \text{false, i.e., } \bar{1} = 0,$$

and $$\text{NOT false} = \text{true, i.e., } \bar{0} = 1.$$

Fig. 1.3. Normally-open and Normally-closed Contacts

The normally-open and normally-closed contacts of a relay represent complementary elements, and for Fig. 1.3(a) we can say:

> The contact will be closed when the relay is energised.

This statement can be represented by the symbol A.

The corresponding statement for Fig. 1.3(b) is:

> The contact will NOT be closed when the relay is energised.

This statement can be represented by the symbol $\bar{A}$.

Note that $A + \bar{A}$ can assume only the values (1 + 0) or (0 + 1), i.e.,

$A + \bar{A}$ must always equal 1.

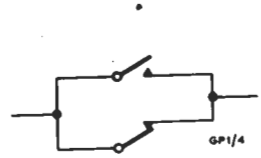


Fig. 1.4. $A + \bar{A} = 1$

This is exemplified by the switching network in Fig. 1.4. Similarly $A.\bar{A}$ can assume only the values 1.0 or 0.1, i.e.

$A.\bar{A}$ must always equal 0.

This is exemplified by the switching network in Fig. 1.5.

Fig. 1.5. $A.\bar{A} = 0$

Inclusive and Exclusive Configurations

The OR circuit already described under the heading *Parallel Circuits* provides an output at level 1 not only when P or Q is at level 1, but also when *both* P and Q are at level 1. This configuration is termed an *inclusive* OR gate. If an OR gate cannot provide an output at level 1 when more than one of its inputs are at level 1, it is termed *exclusive.*

The operation of an exclusive OR gate with two switching elements P and Q can be defined by the statement:

> The circuit through the switching network is continuous if P AND NOT-Q, or NOT-P AND Q, are closed.

Construction of Truth Tables for n Variables

As stated previously, many switching circuits are composed of combinations of parallel-connected and series-connected elements, and in an analysis of the operation of such circuits, all possible combinations of the conditions of the elements must be considered.

To ensure that this condition is fulfilled for a large number of elements, a logical system of laying out the associated truth table must be adopted.

Each variable can take either of two values, and Fig. 1.6 shows how all possible combinations for four variables can be generated

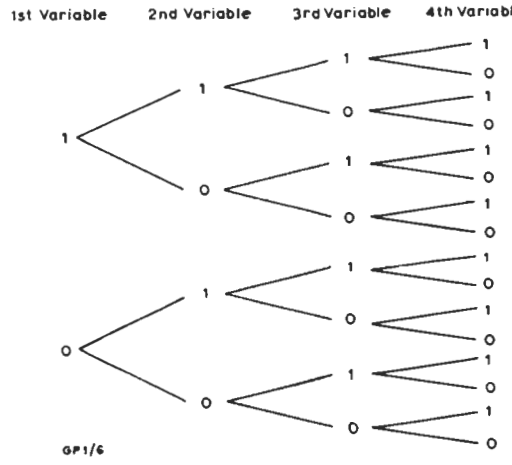


Fig. 1.6. Method of Generating All Possible Combinations of Four Variables

Table 6 demonstrates how the normal form of a truth table is obtained from Fig. 1.6.

Table 6

1st Variable	*2nd Variable*	*3rd Variable*	*4th Variable*
1	1	1	1
1	1	1	0
1	1	0	1
1	1	0	0
1	0	1	1
1	0	1	0
1	0	0	1
1	0	0	0
0	1	1	1
0	1	1	0
0	1	0	1
0	1	0	0
0	0	1	1
0	0	1	0
0	0	0	1
0	0	0	0

For n variables, the general rules for the formation of a table are as follows:

1. In the first column, write 2^{n-1} ones followed by 2^{n-1} zeros.
2. In the second column, write 2^{n-2} ones followed by 2^{n-2} zeros.
3. In the nth column, write 2^{n-n} ones followed by 2^{n-n} zeros. (Note that $2^{n-n} = 2^{o} = 1$.)

A proof of these rules can be found in a textbook on logic.

Example 1

Our analyses of the basic series-connected and parallel-connected circuits will now be used to find the performance of a typical switching circuit.

A switching network formed by the contacts of three relays A, B and C is shown in Fig. 1.7. All the relays have normally-open and normally-closed contacts, connected as shown, and it is required to determine the conditions under which the network is conducting

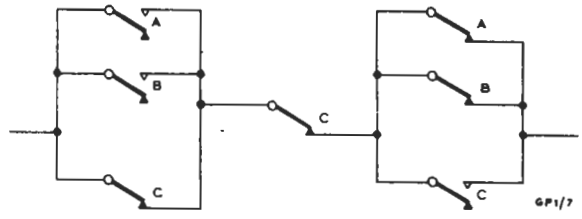


Fig. 1.7. Switching Network formed by the Contacts of Three Relays

A general statement of the required conditions is that

The network will be conducting if

$$\left\{\begin{array}{c}\text{A or}\\ \text{NOT-C or}\\ \text{B}\end{array}\right\} \text{ and } \{\text{NOT-C}\} \text{ and } \left\{\begin{array}{c}\text{NOT-A or}\\ \text{C or}\\ \text{NOT-B}\end{array}\right\}$$

has a truth value equal to 1, i.e., if

$$(A + \overline{C} + B) \times (\overline{C}) \times (\overline{A} + C + \overline{B}) = 1$$

There are three variables A, B and C and the formulae given previously enable the combinations to be set out as shown in Table 7. This final column of the table gives the value of the equation for each of the eight combinations

The final column shows that the equation has a truth value of 1 under the following three conditions:

(a) Relay A energised, relay B NOT-energised, and relay C NOT-energised.
(b) Relay A NOT-energised, relay B energised, and relay C NOT-energised.
(c) Relay A NOT-energised, relay B NOT-energised, and relay C NOT-energised.

This simplifies to:

Relay C NOT-energised and *either* relay A *or* relay B *or both* NOT-energised.

Table 7

A	B	C	$\bar{A}$	$\bar{B}$	$\bar{C}$	$(A+B+\bar{C})\times\bar{C}\times(\bar{A}+C+\bar{B})$
1	1	1	0	0	0	$1\times0\times1=0$
1	1	0	0	0	1	$1\times1\times0=0$
1	0	1	0	1	0	$1\times0\times1=0$
1	0	0	0	1	1	$1\times1\times1=1$
0	1	1	1	0	0	$1\times0\times1=0$
0	1	0	1	0	1	$1\times1\times1=1$
0	0	1	1	1	0	$0\times0\times1=0$
0	0	0	1	1	1	$1\times1\times1=1$

Logic Circuits

The methods demonstrated previously can be extended to the analysis of all types of switching networks (including those which use transistors as switching elements). Such circuits are commonly referred to as logic circuits.

Logic Levels

The logic values 0 and 1 are usually defined in terms of voltages on the signal paths of the logic circuit.

Thus:

1 is equivalent to say, $+E_1$ volts.
0 is equivalent to, say, $-E_2$ volts.

This is often shown in tabular form thus:

Logic	
1	$+E_1$ volts
0	$-E_2$ volts

The term *positive logic* is applied to systems in which the truth value 1 represents a level of potential which is more positive than that represented by 0; the converse is defined as *negative logic.*

Duality

Consider a logic circuit where the output F is a function of two variables A and B and where the input levels are capable of assuming only +2 volts or −3 volts. Assume that the circuit behaves according to the combination shown in Table 8.

In positive logic the −3 volt level is the 0-state and the +2 volt level is the 1-state. Substitution of the logic values for the voltage levels gives the result shown in Table 9.

Table 8

Inputs		Outputs
A	B	F
−3V	−3V	−3V
−3V	+2V	−3V
+2V	−3V	−3V
+2V	+2V	+2V

This is the truth table for the AND function, as defined in Table 2. Therefore the circuit performs the AND operation in positive logic.

In negative logic the −3 volt level is the 1-state and the +2 volt level is the 0-state. Substitution of the logic values for the voltage levels gives the result shown in Table 10.

Table 9

Inputs		Outputs
A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

Table 10

Inputs		Outputs
A	B	F
1	1	1
1	0	1
0	1	1
0	0	0

This the truth table for the OR function, as defined in Table 4. Therefore the circuit performs the OR operation in negative logic.

Tables 8 to 10 and the associated explanations show that the same circuit can perform either the AND operation or the OR operation, depending on whether positive or negative logic is used. In the same way an inverting logic circuit can perform either the NOT-AND (NAND) or NOT-OR (NOR) operation, depending on whether positive or negative logic is used. Therefore the logic function performed by a given device is determined by whether positive or negative logic is used.

The choice of logic can be indicated by either:

(a) stating on the logic diagram whether a positive or negative logic convention is used exclusively on the diagram, or

(b) indicating the logic convention in force at the inputs and outputs of each logic element on the diagram.

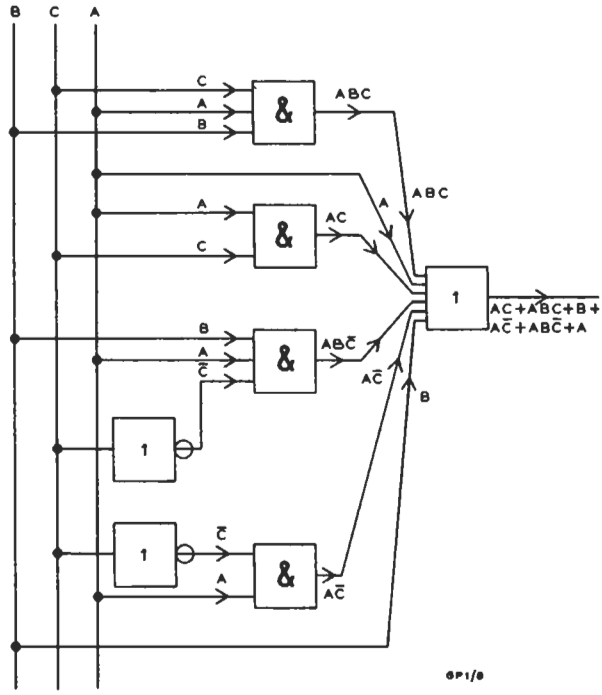


Fig. 1.8. Logic Network considered in Example 2

Example 2

Suppose that it is required to establish the conditions necessary to obtain an output from the logic network shown in Fig. 1.8.

The output from the circuit is controlled by an OR gate and if any of the inputs to this gate are present, then the network will provide an output. This gives rise to the following statement:

There is an output from the circuit if

A and B and C or
A or
A and C or
A and B and NOT-C or
A and NOT-C or
$\overline{B}$

have a truth value of 1

There is thus an output from the circuit if:

$$(A \times B \times C) + A + (A \times C) + (A \times B \times \overline{C}) + (A \times \overline{C}) + B = 1.$$

There are three variables, A, B, and C, and Table 11 shows how the value of the above equation is derived for all possible combinations of A, B, and C.

Table 11

A	B	C	$\overline{A}$	$\overline{B}$	$\overline{C}$	ABC	AC	$AB\overline{C}$	$A\overline{C}$	A+AC+ABC+$AB\overline{C}$+$A\overline{C}$+B
1	1	1	0	0	0	1	1	0	0	1
1	1	0	0	0	1	0	0	1	1	1
1	0	1	0	1	0	0	1	0	0	1
1	0	0	0	1	1	0	0	0	1	1
0	1	1	1	0	0	0	0	0	0	1
0	1	0	1	0	1	0	0	0	0	1
0	0	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0

The conditions under which an output is present are extracted from Table 11 and grouped as shown in Table 12.

The table shows that if either of the inputs A or B is present, there will be an output from the network irrespective of whether C is present or not present.

This leads to the conclusion that a sufficient condition for the network to provide an output would be if A or B were present. The circuit of Fig. 1.8 can therefore be replaced by one OR gate as shown in Fig. 1.9.

Table 12

A	B	C
1	1	1
1	1	0
1	0	1
1	0	0
0	1	1
0	1	0

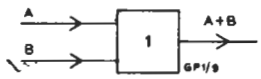


Fig. 1.9. OR Gate Equivalent to Circuit of Fig. 1.8

Combinational and Sequential Logic

All circuits so far considered have outputs which are functions of a combination of static input conditions, and their operation is therefore defined in terms of what is referred to as *combinational* or *static* logic. However, in many logic circuits, some inputs are derived from the output of the circuit, and are therefore functions of the previous state of the circuit. Because the output of such circuits is dependent upon a sequence of input conditions, their operation is defined in terms of a *sequential* logic.

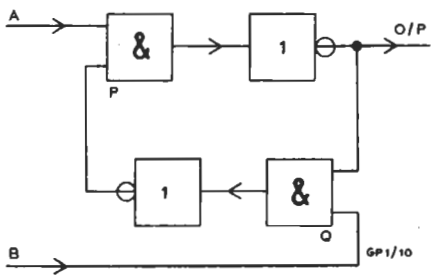


Fig. 1.10. Sequential Logic Circuit

An example of such a circuit is shown in Fig. 1.10. Input B of this circuit is held at level 1 and, if the output is at level 0, then by inspection input A must be at level 1. If input A changes to 0, the output changes to 1, and Q now has two inputs at level 1. As a consequence of this the input to gate P from gate Q is held permanently at level 0 and further changes in level of input A therefore have no effect on the output. In other words, the effect of a change in level of input A is determined by the state of the output in a period immediately prior to the instant at which A is changed. Thus it is necessary to specify this previous state of the output before the result of a change of input conditions can be deduced.

In certain sequentially-operated circuits, instability can arise on application of particular static input conditions. An example of such a circuit is given in Fig. 1.11 in which, if the input A is held at level 1, the output alternates between its two states. To avoid this situation it is desirable that such circuits should have a pulsed input, the duration of which is determined by the sum of the propagation delays of the circuit.

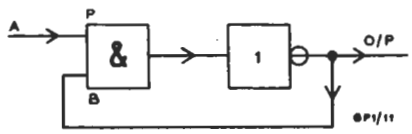


Fig. 1.11. Sequential Logic Circuit with Possible Instability

The operation of sequential logic circuits can be shown in a truth table in a similar manner to that described previously for combinational arrangements. The sequential truth table differs slightly from the combinational truth table in that it specifies the period in which the binary levels shown are sampled. A sampling period commences at one change of state of the input conditions and ends at the next change of state; these periods need not be of equal duration. Thus in any sampling period, say the *n*th, the output Q of a given configuration of gates is defined as Q^n. If the circuit contains feedback loops, then output Q^n may be a function of the previous output Q^{n-1}, as well as of other inputs which may have been set a discrete number of periods previously. An example of this would be an input A which was set to a given binary level three periods previously and which at the *n*th sampling period, would be A^{n-3}.

Thus Q^n could be defined by

$$Q^n = f(Q^{n-1}, A^{n-3}).$$

Although Q^n is determined by a binary level which existed three periods previously, information as to what this level was could be provided during any subsequent sampling period by the use of a memory type of circuit. As this memory could

provide the information in the $(n-1)$th period, an expression showing the relation between inputs and output could be written in terms of only two consecutive sampling periods. This is in fact the normal method of defining the output of sequential logic circuits. A suitable memory circuit would be given by the configuration in Fig. 1.10. This circuit provides a permanent record of a change of state of input A from binary level 1 to level 0; the memory can be reset by changing the binary level of input B to zero.

Further Reading

Only the most elementary forms of switching algebra have been introduced into the analysis of the circuits in this Section. Much more sophisticated methods of approach based on Boolean algebra have been devised and the following books are recommended as providing explanations of the use and applications of these methods. A work on elementary logic is included, as it provides a useful introduction to ideas which are often taken for granted in more advanced books.

1. **BAILEY, C. A. R.** Sets and logic (1 and 2). Arnold.
2. **MARCUS, M. P.** Switching circuits for engineers. Prentice Hall.
3. **HURLEY, R. B.** Transistor logic circuits. John Wiley and Sons.
4. **DEAN, K. J.** An introduction to counting techniques and transistor circuit logic. Chapman and Hall.
5. **FLEGG, G.** Boolean algebra and its applications. Blackie.

P. D. M. 6/67
revised **T. E. S. 2/71**

SECTION 2

BASIC BISTABLE DEVICES

General

Bistable devices, usually in integrated-circuit form, are among the most commonly used 'building blocks' for logic circuits. The term bistable embraces a family of electronic devices all of which have two stable states and hence possess the ability to store information. Bistables are often referred to as 'flip-flops' but in Britain a flip-flop is defined (see British Standard 204: 1960 Term 32042) as a device with only one stable state, i.e. as a monostable device; therefore the term flip-flop should be avoided to prevent confusion.

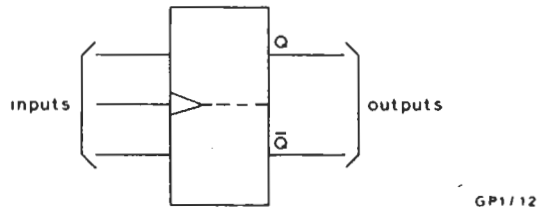


Fig. 2.1. Logic Symbol for a Bistable Device

The logic or block diagram symbol for a bistable device with normal inputs and outputs is shown in Fig. 2.1 and consists of two adjacent squares; one square shows the Q output and the other shows the inverse ($\bar{Q}$) output. All inputs which when fed with a logic 1 set the Q output to 1 are taken to the same square as the Q output, and all inputs which when fed with a logic 1 set the $\bar{Q}$ output to 1 are taken to the same square as the $\bar{Q}$ output. An input which changes the state of the bistable regardless of its previous state is taken to the junction of the two squares. A glossary of bistable symbols (derived from British Standard 3939) is given in Section 4.

Input Circuits

Bistable input circuits can take one of the three forms listed below.

Capacitor-coupled

Each input is coupled to the bistable via a capacitor so that the circuit reacts only to transitions in the input signal.

Direct-coupled

Each input is direct-coupled to the bistable and a change of state is induced by a change of input level.

Hybrid or Edge Triggered

Each input is direct coupled, but the circuit arrangements are such that the bistable reacts only to transitions in the input signal.

Synchronous Operation

Basic bistable devices change state whenever the the correct input (or input combination) occurs. However, when large numbers of bistables are combined in logic circuits, it is sometimes necessary to ensure that the changes of state occur synchronously. This is done by providing each bistable with an additional input to which clock pulses are applied, these pulses providing a time reference for the logic circuit.

Bistable Classifications

Bistables can be either clocked or unclocked and are further classified by the way in which they respond to input signals. The basic classifications are detailed below; variations are obtained by combining the characteristics of various types of bistable.

For simplicity the truth tables given for the various types of bistable refer only to the Q output and ignore the inverse, or $\bar{Q}$, output. The suffixes n and (n + 1) are used in the truth tables to define the Q output in terms of time. Q_n gives the state of the Q output just before an input (or inputs) changes state; Q_{n+1} gives the state of the output one clock pulse later.

RS Bistable

The RS bistable (alternative names are Set/Reset and Set/Clear) registers information in accordance with an instruction (signal) applied to one of its inputs. RS bistables are available in clocked and unclocked versions. The clocked version has an additional input to which clock pulses are applied. Truth tables for (a) unclocked and (b) clocked RS bistables are given in Table 1; the two truth tables differ only in that (a) gives instantaneous values of Q and (b) gives the output of Q during the following clock pulse.

Table 1

(a) Unclocked

R	S	Q
0	0	no effect
0	1	1
1	0	0
1	1	*

(b) Clocked

R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	*

*The output obtained with the input combination R = 1, S = 1, is uncertain; therefore this input combination is not used.

T Bistable

T (or Trigger) bistables have only one input; in block or logic diagrams this single input is applied to the junction of the two squares that constitute the symbol. A T bistable changes state for every input pulse, regardless of its existing state. Thus the device functions as a counter (i.e. as a binary divider). The truth table for a T bistable is given in Table 2.

Table 2

T	Q_n	Q_{n+1}
0	0	0
1	0	1
0	1	1
1	1	0

JK Bistable

The JK bistable (the origin of this term is unknown) is a synchronous device which combines the functions of RS and T bistables. The truth table for a JK bistable is given in Table 3.

Table 3

J	K	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	$\overline{Q}_n$

Note: When both the J and K inputs are at logic 1, each clock pulse input changes the state of the bistable; in other words the Q output becomes the inversion ($\overline{Q}$) of what it was before the arrival of the clock pulse. Under these conditions the device functions as a binary divider in the same way as a T bistable.

In addition to the J, K and clock inputs, most JK devices provide direct access to both sides of the bistable by means of Set and Reset (or Preset and Clear) terminals. Because these inputs operate directly on the device, information fed to them takes priority over any other input information.

Master/Slave Bistable

Master/Slave bistables are clocked devices which consist of two interconnected bistable sections, a master section and a slave section. Information can be fed from the data inputs into the master section only when the leading edge of a positive-going clock pulse is present, the information being transferred to the slave section by the trailing edge of the same clock pulse, as shown in Fig. 2.2. Thus the transfer of information from input to output is delayed by a time which depends on the clock-pulse duration.

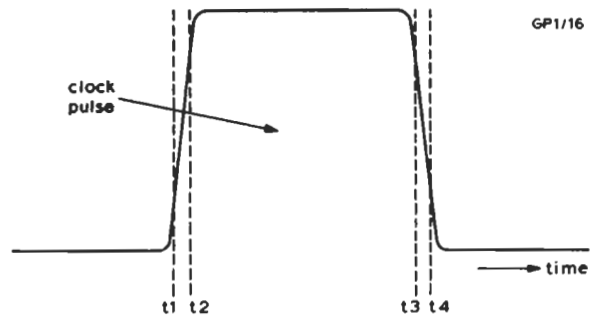


Fig. 2.2. Clock-pulse Timing of a Master/slave Bistable Device

Master/Slave bistables can be either RS or JK types; therefore the appropriate RS or JK truth table can be applied, although the action takes place in two steps.

D Bistable

The D (delay) bistable is a device which transfers information from the D input to the Q output on receipt of a positive-going clock pulse. In its simplest form the device has only D and clock-pulse inputs. Between clock pulses, a change in the state of the D input has no effect on the Q output. The D bistable therefore acts as a store. The truth table is given in Table 4: the states shown are those after the leading edge of the clock pulse.

D_n	Q_{n+1}
0	0
1	1

If the $\overline{Q}$ output is connected to the D input and the input information is applied to the clock-pulse input, the device acts as a divide-by-two stage.

SECTION 3

MULTIPLE BISTABLE DEVICES

Shift Register

Introduction

A shift-register consists of a series-connected chain of bistable elements in which each element stores one binary digit. The stored information can be shifted either to the right or to the left (depending on the construction of the register) by the application of clock pulses to a shift line. If the contents are shifted to the right the register is said to be forward-shifting; if the contents are shifted to the left the register is said to be reverse-shifting. A reversible shift register can be made to shift in either direction as required, the reversing action being carried out by gates.

1-state. The second clock pulse allows the second information bit to enter the register, whereupon the 1 previously stored in the first stage shifts to the second stage and the first stage changes to the O-state. On receipt of the third clock pulse, the first stage changes to the 1-state, the second stage changes to the O-state and the third stage changes to the 1-state. Subsequent clock pulses feed the remainder of the signal into the register, the information being shifted one stage to the right by each clock-pulse, until the process is completed. Normally these changes occur on the positive-going edges of the clock pulses.

The signal stored in the register is read out by

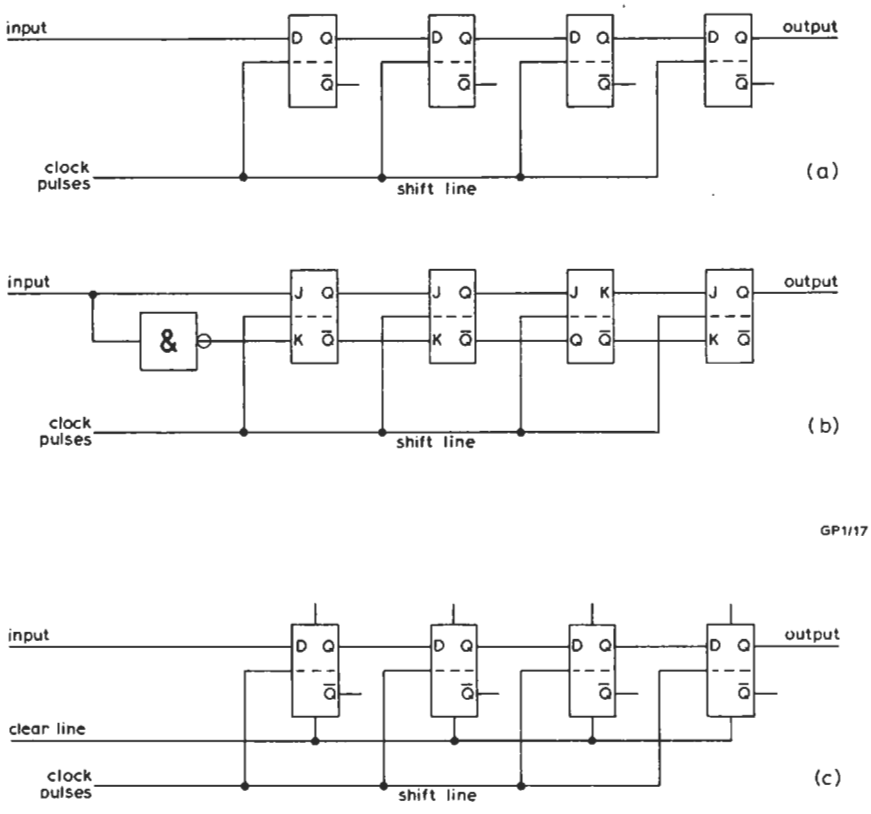


Fig. 3.1 Serial Input/Serial Output Shift Registers

Shift registers are usually constructed from JK Master/Slave or D-type bistables and can be built to accept information either in serial (sequential) form or in parallel form.

Serial Input and Output Shift Registers

The circuit diagram of a serial-input/serial-output shift register which uses D-type bistables is shown in Fig. 3.1(a) and one which uses JK Master/Slave bistables is shown in Fig. 3.1(b). The operation of Fig 3.1(a) is described below.

Assume that the register is initially empty (i.e. with all stages in the Q=O state) and that a five-bit word, say 10101, is applied to the data input. The first clock pulse allows the first information bit to enter the register and the first stage changes to the

applying a succession of clock pulses to the shift input while the data input is held in the O-state.

The shift register shown in Fig 3.1(b) is similar in operation to that already described. Here, however, input information is fed directly to the J input and via an inverter to the K input; moreover with JK Master/Slave bistables the transfer of information takes place on the negative-going edge of the clock pulses (see Section 2).

Some registers are fitted with an additional input which provides a clearing facility, as shown in Fig 3.1(c). The application of a logic 0 to the Clear line sets all stages of the register to 0 and thus obviates the need to hold the data input at 0 for a succession of clock pulse cycles.

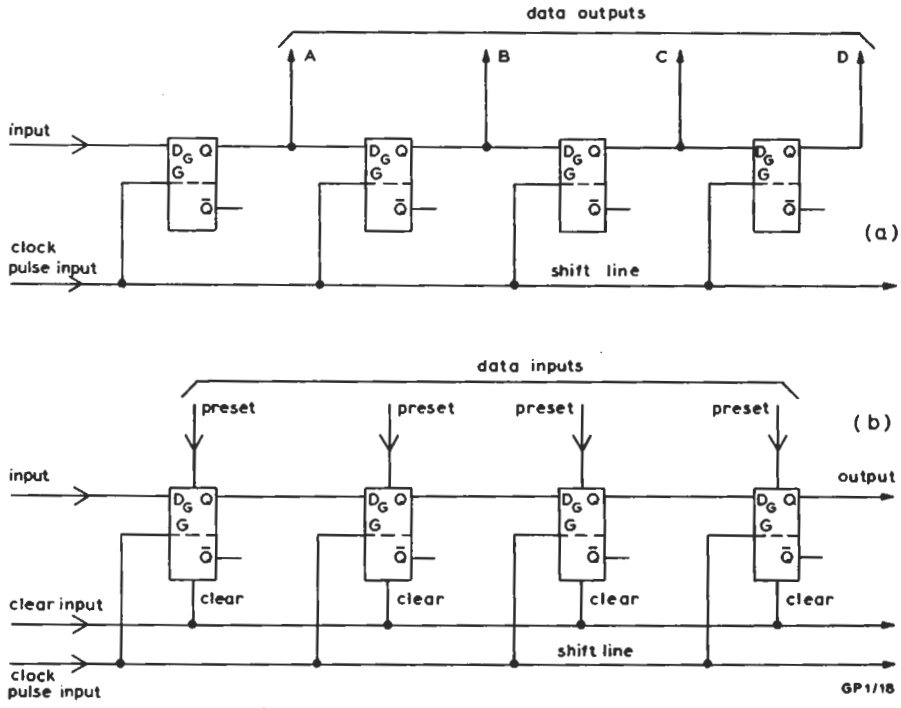


Fig. 3.2 Serial/Parallel Shift Registers
(a) Serial Input/Parallel Output
(b) Parallel Input/Serial Output

Serial/Parallel Shift Registers

A serial-input/parallel-output shift register is shown in Fig. 3.2(a). This type of register operates in the way already described except that all the stored information is read out simultaneously.

A parallel-input/serial-output shift register is shown in Fig. 3.2(b). Here all the information is read into the register simultaneously, but the output is in serial form and is read out under the control of clock pulses.

Ring Counters

A ring counter (alternative names are circulating shift register or commutator) is a shift register with the output connected to the input.

The circuit diagram of a three-stage ring counter is shown in Fig. 3.3; the initial state of the stages is set by means of the preset and clear inputs. For the purposes of description assume that the stages are set so that A is at 1 and B and C are at 0. When the first clock pulse is applied to the counter B changes to 1 and A, because it receives its input information from C, changes to 0. The second clock pulse changes C to 1 while A remains at 0 and B becomes 0. Each succeeding clock pulse shifts the 1 by one step, thus the 1 circulates through the register for as long as the clock pulses are applied.

A variation of the basic ring counter is the twisted-ring or Johnson counter, for which a circuit diagram is given in Fig. 3.4. The twisted-ring counter differs from the ring counter in that the $\bar{Q}$ output (not the Q output) of the final stage is fed back to the J input.

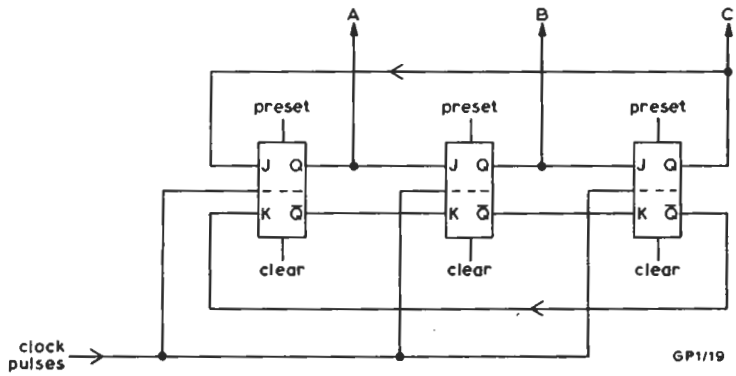


Fig. 3.3 Ring Counter

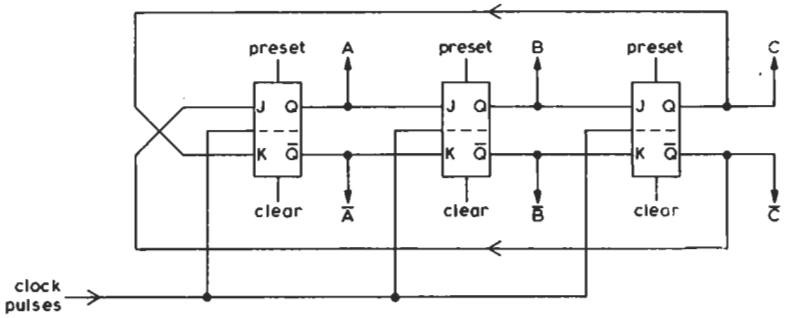


Fig. 3.4 Twisted Ring (Johnson) Counter

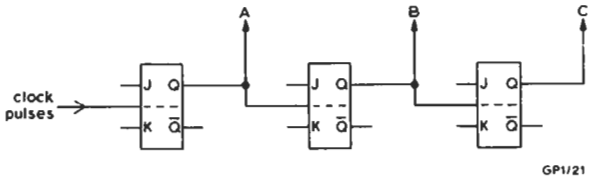


Fig. 3.5 Asynchronous Binary Up-counter

TABLE 1

Input Pulses	*C*	*B*	*A*	*Counter State*
0	0	0	0	0
1	0	0	1	1
2	0	1	0	2
3	0	1	1	3
4	1	0	0	4
5	1	0	1	5
6	1	1	0	6
7	1	1	1	7

Counters

A binary counter consists of a number of binary dividers connected in cascade. Counters can be divided into two classes: asynchronous and synchronous; both clases can be sub-divided into up-counters (counters which count upwards from zero) and down-counters (counters which count down towards zero). In asynchronous counters the trigger input of each stage is derived from the output of the preceding stage. This gives rise to the so-called ripple-through effect; i.e. the last stage cannot change state until all the preceding stages have changed state and, as a result, there is an appreciable propagation delay in the counter equal to the sum of the stage delays. In synchronous counters clock pulses are fed to all stages and those stages which are required to change state for a given count pulse are operated on simultaneously. Thus synchronous counters have shorter propagation delays than asynchronous counters.

Asynchronous counters can be constructed from RS, JK or (by connecting the $\bar{Q}$ output back to the D input) D type bistable devices. Synchronous counters are constructed from JK bistables, usually of the Master/Slave type.

Asynchronous Counters

The circuit diagram of a three-stage asynchronous binary up-counter is shown in Fig. 3.5 and the associated truth table is given in Table 1. The wave-

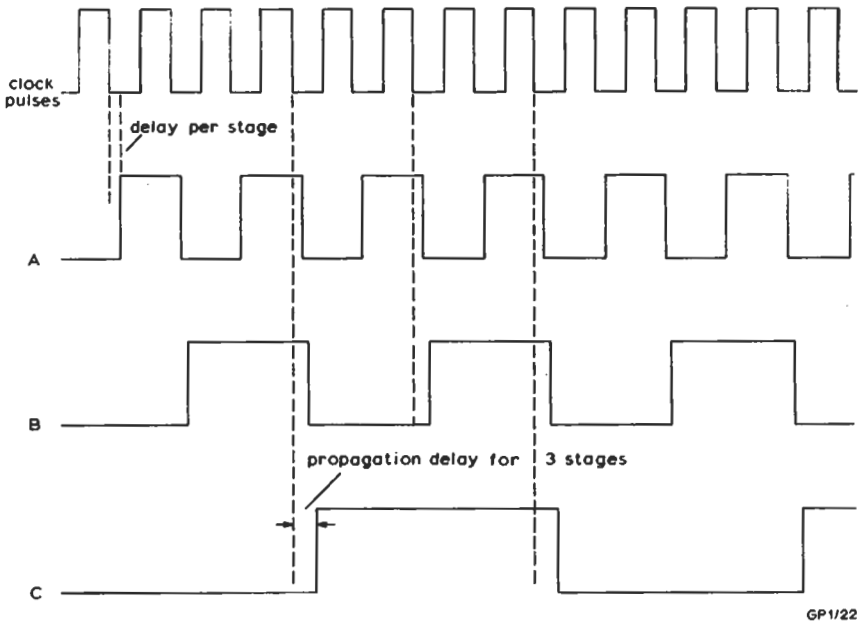


Fig. 3.6 Waveforms for Asynchronous Up-counter Showing Propagation Delays

forms obtained at the counter outputs are shown in Fig. 3.6 to illustrate the cumulative propagation delay.

A three-stage asynchronous down-counter is shown in Fig.3.7 and the associated truth table is given in Table 2. In Fig. 3.7 the stage interconnections are taken from the $\bar{Q}$ output instead of from the Q output as in Fig. 3.5 although the count is still observed on the Q output.

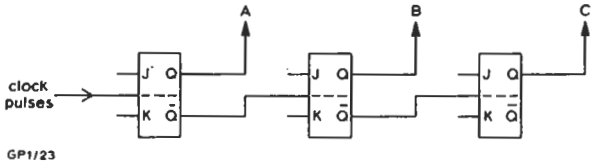


Fig. 3.7 Asynchronous Binary Down-counter

Synchronous Counters

Synchronous counters are more complex than asynchronous counters because additional gating circuits are required for their operation. Where integrated circuits are used the gates are often contained in the same module as the bistable device. A three-stage synchronous up-counter is shown in Fig. 3.8. The truth table is the same as for the asynchronous up-counter mentioned earlier and is given in Table 1.

TABLE 2

Input Pulses	*C*	*B*	*A*	*Counter State*
0	1	1	1	7
1	1	1	0	6
2	1	0	1	5
3	1	0	0	4
4	0	1	1	3
5	0	1	0	2
6	0	0	1	1
7	0	0	0	0

By definition (see Section 2) a JK bistable acts as a counter only when both the J and K inputs are set to 1. In Fig. 3.8 all stages of the counter are clocked simultaneously but the J and K inputs are fed via AND gates which are controlled by the outputs of the preceding stages. Therefore the J and K inputs of any stage are at 1 only when all the preceding stages are at

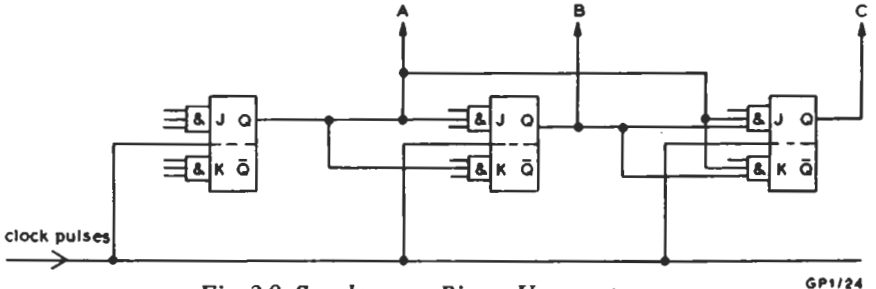


Fig. 3.8 Synchronous Binary Up-counter

1 also. For certain types of logic (e.g. TTL) unconnected inputs to the AND gates are effectively at 1.

Counters with Feedback

Binary counters divide by 2^n where n is the number of stages in the counter. To obtain a count which is not a power of 2 from a binary counter, the counter must be modified by feedback (sometimes called knockback) techniques.

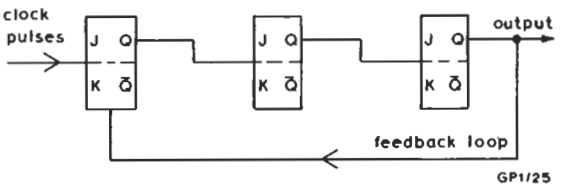


Fig. 3.9 Divide-by-seven Counter

Fig. 3.9 shows a three-stage counter which, without feedback, would divide by 8. The output of the last stage, which is delayed with respect to the input pulse because of the propagation delay through the counter, is differentiated and fed back to the first stage as a feedback pulse. Because the input and feedback pulses are not coincident in time, the feedback pulse causes the first stage of the counter to change state and the overall count is reduced by one. The division ratio of the circuit is $2^n - 1 = 7$. If the feedback is applied to the second stage of the counter instead of the first the division ratio becomes $2^n - 2 = 6$.

Certain division ratios cannot be obtained using a single feedback loop. For example, a divide-by-ten or decade counter requires two feedback loops, as shown in Fig. 3.10.

The division ratio of a counter with feedback may be deduced in the following way. Consider first the innermost feedback loop (in Fig. 3.10) which embraces bistables B and C. Without feedback the division ratio of this pair would be 2^2, i.e. 4, but feedback reduces this to 3. Now consider the second feedback loop embracing bistables A, B and C. Without the second feedback loop this would have a division ratio of 2 x 3, i.e. 6, but feedback reduces this to 5. Bistable D has a division ratio of 2, making the overall division ratio 2 x 5, i.e. 10.

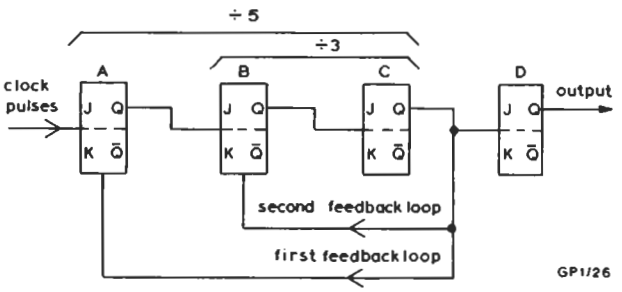


Fig. 3.10 Divide-by-ten Counter

The stage which is outside both feedback loops is shown at the end of the counter in Fig. 3.10 but it could equally well have been placed at the beginning. Both arrangements give a count of 10 but with the final 2:1 stage the overall mark/space ratio is 2:1 whereas with the 2:1 stage at the beginning the overall mark/space ratio is 4:1.

Dividers

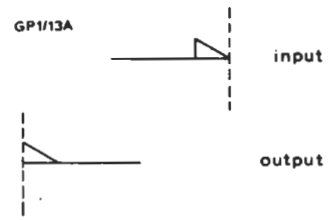
The counters described above operate on regularly-occuring input signals or input signals occuring at random intervals. Where a counter is used for regularly-occuring inputs it is often called a *frequency-divider* or more simply *a divider.*

SECTION 4

LOGIC SYMBOLS

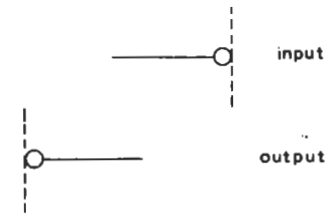
The logic symbols given in this Section are taken from British Standard 3939, (Graphical Symbols for Electrical and Electronic Diagrams) Section 21. Note that semi-circular gate symbols have now been superseded by rectangular ones, but semi-circles may still be found on many existing logic diagrams.

Indicators



(a) Polarity Indicator

Use to show that the 1-state is the less positive level; i.e. that negative logic is used. The absence of a polarity indicator shows that the 1-state is the more positive level i.e. that positive logic is used.

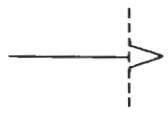


(b) Negation Indicator

Note: If desired, the connection line may be drawn through the circle.

(c) Inhibiting Input

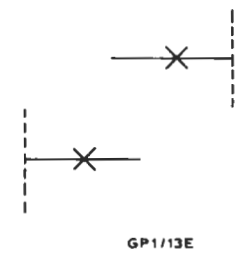
If the inhibit input of a logic element is at its 1-state, it prevents the output of that element from assuming its defined 1-state, whatever the value of the other input signals.



(d) Dynamic (edge-triggered) Input

The 1-state is defined as the transition from the 0-state to the 1-state, and not by the continued presence of the 1-state.

The absence of a dynamic symbol means that the input is a static (direct-coupled) one.

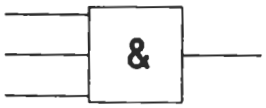


(e) Absence-of-logic Indicator

Input which does not carry logic information.

Output which does not carry logic information.

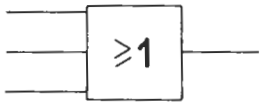
Gates



GP1/14A

(a) AND Gate

The output is at its defined 1-state if, and only if, all the inputs are at their defined 1-states.

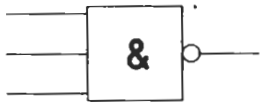


GP1/14B

(b) OR Gate

The output is at its defined 1-state if, and only if, one or more of its inputs are at their defined 1-states.

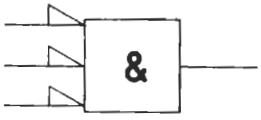
Note that '≥ 1' may be replaced by '1' if no ambiguity arises.



GP1/14C

(c) NOT-AND (NAND) Gate

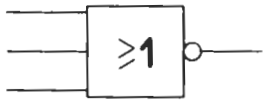
The output is at its defined 0-state if, and only if, all the inputs are at their defined 1-states.



GP1/14D

(d) NOT-AND (NAND) Gate with polarity indicator.

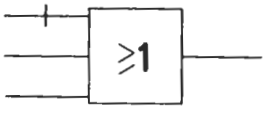
The output is at its defined 1-state if, and only if,all the inputs are at their defined 0-states.



GP1/14E

(e) NOT-OR (NOR) Gate

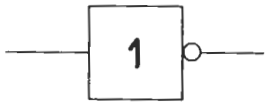
The output is at its defined 0-state if, and only if, one or more of its inputs are at their defined 1-states.



GP1/14F

(f) OR Gate with Inhibiting Input

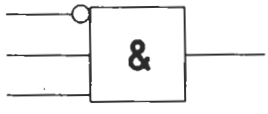
Applying a 1 to an inhibiting input prevents the output changing to its 1-state, whatever the state of the other inputs.



GP1/14G

(g) Negater

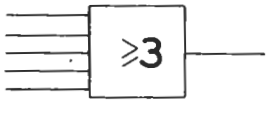
The output is at its defined 0-state only if the input is at its defined 1-state.



GP1/14H

(h) Negated Input to a Gate

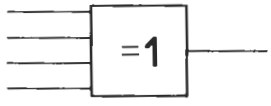
The application of a 0 to a negated input has the same effect as the application of a 1 to a normal input.



GP1/14I

(i) Logic Threshold

The output is at its defined 1-state only if the number of inputs at the defined 1-state reaches the number contained in the symbol.

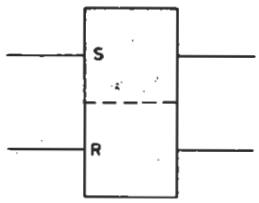


GP1/14J

(j) Exclusive OR

The output stands at its defined 1-state if one, and only one, of the inputs stands at its defined 1-state.

Bistable Elements

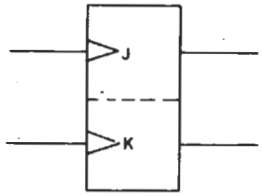


GP1/15A

(a) RS Bistable

If the two inputs are in different states, the output opposite the input which is in its defined 1-state goes to 1 and the other output goes to 0; i.e. the outputs are complemented. If thereafter, both inputs go to their defined 0-states the outputs do not change.

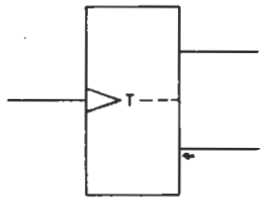
If both inputs, simultaneously, go to their defined 1-states, the behaviour of the element is uncertain.



GP1/15B

(b) JK Bistable

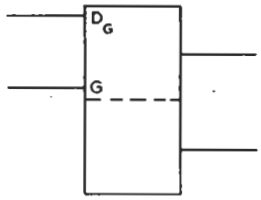
The outputs are always complementary. If one of the inputs goes to its defined 1-state, the adjacent output goes to 1. If both inputs, simultaneously, go to the 1-state the outputs change state. If both inputs go to their 0-states the outputs remain in their previous states.



GP1/15C

(c) T Bistable

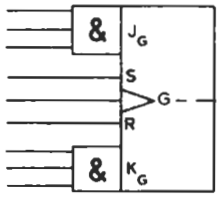
The outputs are always complementary. If the input goes to its defined 1-state, the outputs change state. If the input goes to its defined 0-state the outputs remain in their previous states.



GP1/15D

(d) DG (Delay) Bistable

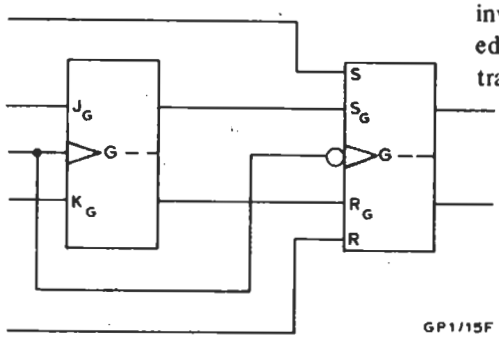
The suffixG means that the D input is subordinate to the G(clock) input. If the D input goes to 1 the adjacent output does not go to 1 until the G(clock) input also goes to 1.



GP1/15E

(e) Multi-input JK Bistable

With additional set, reset and clock inputs. The suffixGafter the J and K inputs means that these inputs are subordinate to the G input.



GP1/15F

(f) Master-slave Bistable

Note that the G(clock) input to the slave section of the device is inverted. Therefore the master section is controlled by the leading edge of a clock pulse, and the slave section is controlled by the trailing edge of the pulse.