Preferred citation: S. Eilon. The future of control procedures for management. In **Papermaking Systems and their Control**, *Trans. of the IVth Fund. Res. Symp. Oxford, 1969*, (F. Bolam, ed.), pp 690–711, FRC, Manchester, 2018. DOI: 10.15376/frc.1969.2.690.

THE FUTURE OF CONTROL PROCEDURES FOR MANAGEMENT

Prof. S. EILON, D.Sc., Ph.D., D.I.C., F.I.Mech.E., F.I.Prod.E., Management Engineering Section, Mechanical Engineering Department, Imperial College of Science and Technology, London

---

Introduction

THERE is no difference, at least in principle, between control of administrative systems and control of other systems. The purpose of a control procenure is to specify the way in which the behaviour of a system can be affected. The controller of the system scrutinises information about the performance of the system and every so often he has to make a decision, namely, to choose between several courses of action open to him. This decision is transmitted to the system with the expectation that it will react in a certain way. This cycle of events—monitoring and evaluation of the system's behaviour, followed by a decision for corrective action—is the essence of the control process, irrespective of whether it is control of inanimate systems or managerial control of industrial enterprises.

There are, of course, many facets of the industrial environment that can make managerial control very complex. For one thing, information about the performance of the system is rarely complete, but, as managers often have to act within very severe time constraints, they may be unable to afford the time required for further data collection. Then, the system's response may be uncertain, as is very often the case in economic and social systems both on the macro and the micro scale; so the decision maker has to act in the knowledge that there may be serious discrepancies between his expectation and the system's actual subsequent behaviour. A third important feature that management of an industrial system has to contend with is that the time lag between decision and response can be quite substantial. This is certainly the case at the higher echelons of the management hierarchy, when questions about the allocation and conversion of resources are contemplated. Add to that the fact that there are many levels at which control can be considered, that a system to describe an industrial enterprise can be regarded as a plethora of

systems within systems and that any one manager may have several roles to play in more than one single system and we begin to realise the enormity of the task of analysing management control in a general way.

One approach is to start with very simple systems and to inquire into control relationships within them; in particular, to examine the interactions among controllers when several are assigned to look after the same system. Such an examination may well result in certain general concepts and definitions about linkages between controllers and the way that they can affect the state of the system. These concepts may then be helpful in the study of complex procedures in large systems.

The fewest states that a controller or a system can assume is two, since by definition a single-state system cannot be controlled and, similarly, a single-state controller can have no effect on the system. There are numerous examples of systems operating in a two-state mode—lighting circuits (the light is either ON or OFF), go or not-go gauges in inspection of components, appointments by selection boards (a candidate is either accepted or rejected), jury decisions in court (guilty or not guilty) and so on. The digital computer, which conforms to the rules of binary logic, is another such example. For the sake of convenience, let us label the two states as 1 and 0 or ON and OFF, respectively. There is no special merit in these names, except that 1 and 0 are used in binary arithmetic and ON and OFF describe the two states of a current flowing through electrical circuits. As the two-state operation is fundamental to the understanding of the control function, we shall now examine the two-state control process in some detail.

The two-state single controller

IF THE controller may be either in the state ON or in the state OFF and, similarly, if the system may assume either of these states, then all the possible control procedures may be listed in what is generally called a *truth table* (Table 1). Each procedure expresses the correspondence between the states of the controller and the states of the system. In procedure 2, for example, when the controller switches to ON, the system moves to state ON; when the controller switches to OFF, the system reverts to OFF.

TABLE 1—FOUR PROCEDURES FOR A TWO-STATE SINGLE CONTROLLER

		System			
Controller	*Procedure*	*1*	*2*	*3*	*4*
ON		ON	ON	OFF	OFF
OFF		ON	OFF	ON	OFF

Of the four possible procedures, clearly the first and the last have no need for a controller, since the state of the controller in neither case affects the

state of the system. Procedure 2 calls for *affirmation*, namely, it requires the system to assume the state indicated by the controller. Procedure 3 is that of *negation* (or *inversion*) and here the state of the system is the opposite of that of the controller. If in procedure 3 the labels of the states of the system are reversed and the labels of the state of the controller are left as they are, then procedure 2 is obtained. This is reminiscent of the NOT operator in Boolean algebra and the NOT gate in circuit theory: when an output is connected to an input through a NOT gate, the output is ON when the input is OFF and vice versa.

We see, therefore, that for a two-state system with a single controller A, only two control procedures are feasible—

Procedure	*Controller*	*System*
Affirmation	A	As A (or A for short)
Negation	A	Not A (or $\overline{A}$ for short

In administrative systems, the NOT gate has the connotation of management by exception. The controller may be described as a passive component in the system; as long as he remains in a state of passivity (denoted as OFF), the activity under his charge continues unabated. When he perceives an exception to normal working, he takes action (he switches to the ON position) and causes the system to stop; this is precisely how a NOT gate is defined.

Linkages among several controllers

SUPPOSE that several controllers are assigned to the same system, so that there is complete correspondence between the states of each controller and the states of the system. It is important to note that it is not the state of the individual controller under such conditions that necessarily determines the state of the system. If that were the case, a one-to-one correspondence would have to exist, then either the roles of the controllers are incompatible (when one controller requires it to be in a different state) or the controllers must all act in unison to ensure that no conflict occurs. When the controllers are set to work in unison, then a one-to-one correspondence exists not just between the system and each controller, but among all the controllers as well.

In the absence of a one-to-one correspondence, the state of the system is determined by the combination of the individual states of the controllers. The way in which the controllers are connected defines the control procedure for the system, namely, the correspondence between the composite states of the group of controllers as a whole and the states of the system.

Time-independent gates

THE concepts of AND gates and OR gates,[1, 2] which are used in designing

electrical circuits, are helpful in considering the fundamental ways in which controllers can be connected to form control procedures. To these concepts, we add the ANDOR gate, which can be constructed as a combination of the basic AND, OR and NOT gates, as we shall see later. The ANDOR gate is not commonly identified as a separate gate in circuit theory, but it is a convenient concept to employ, particularly in the study of administrative control procedures. The three gates (henceforth, the terms *gate* and *link* are used interchangeably) are defined as follows—

AND gate	If there are several inputs connected to an output through an AND gate, the output is ON when all the inputs are ON and the output is OFF when one or more inputs are OFF.
OR gate	If there are several inputs connected to an output through an OR gate, the output is ON when one or more inputs are ON and the output is OFF when all the inputs are OFF.
ANDOR gate	If there are several inputs connected to an output through an ANDOR gate, each input can cause the output to be switched ON or OFF.

To these (see also Fig. 1), we should add the NOT gate, which was discussed earlier. It does not define any linkage between controllers, but acts as a negation operator—

NOT gate	If an output is connected to an input through a NOT gate, the output is ON when the input is OFF and vice versa.

The operation of the AND, OR and ANDOR gates is summarised in Table 2, which shows the state of a system with two controllers A and B, where each controller has two states designated as 1 and 0 (or ON and OFF or TRUE and FALSE), respectively, in the form of a truth table.

Changing the state of the system

THE results in Table 2 reveal some interesting facets of the way in which each controller can effect a change in the state of the system. Suppose that the system is in state 1 and that the two controllers are linked by AND. If an event occurs that causes either controller to want to change the state of the system from 1 to 0, he is able to effect such a change.

TABLE 2—THE TWO-STATE TWO CONTROLLER SYSTEM

Controllers		*System*		
A	B	AND	OR	ANDOR
1	1	1	1	1
1	0	0	1	0
0	1	0	1	0
0	0	0	0	1

If both controllers want the system to change, then again the control procedure AND allows both to act and to achieve that aim.

If, on the other hand, the system is in state 0, then the action of one controller (even of both) may not be effective in changing the state of the system from 0 to 1 when the controllers are linked by AND. It all depends on the initial

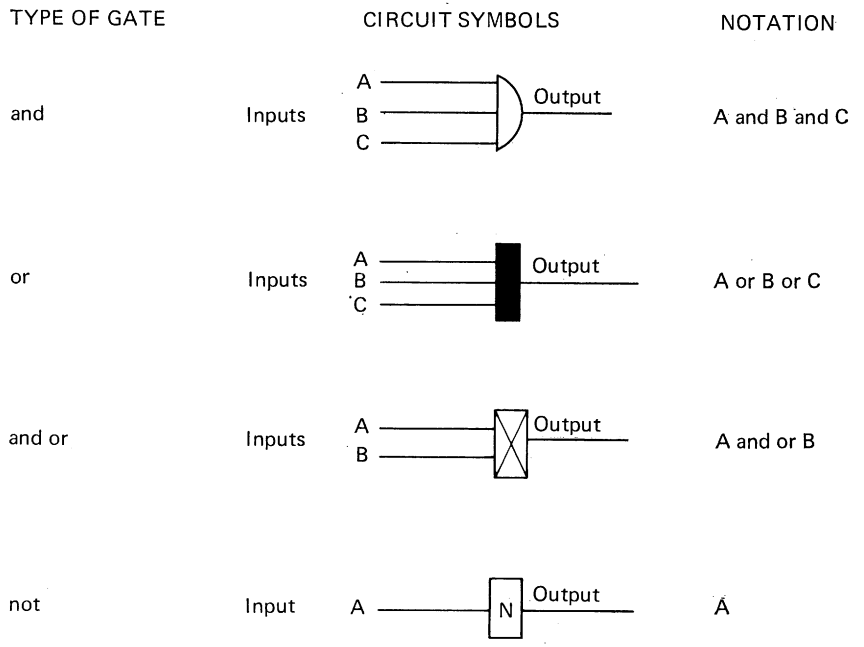

Fig. 1—Four types of gate

state of the controllers. For example, if one is at 1 and the other at 0, then only the latter can effect a change in the state of the system; if the former acts (by changing his state) or if both act simultaneously (each reversing his own state), no change in the state of the system will result.

In the case of the OR link, the situation with regard to change is a mirror image of AND. If the system is at 0, either controller or both can effect a change; if the system is at 1, then the result depends on the initial state of the controllers (see Table 3).

The ANDOR link allows either controller to trigger off a change; if both controllers act simultaneously (each reversing his own state), then no change in the state of the system can take place and this result holds irrespective of the initial state of the system or the initial respective states of the controllers.

TABLE 3—ONE OR TWO CONTROLLERS ACTING TO CHANGE THE STATE OF THE SYSTEM

System's initial state	*Controllers' action*	*Effect on the system* AND	OR	ANDOR
1	One acts	Change	?	Change
	Both act	Change	?	No change
0	One acts	?	Change	Change
	Both act	?	Change	No change

Notes

1. A controller acts by switching his initial state (from 1 to 0 or from 0 to 1).
2. The symbol ? in the table means that the effect on the state of system depends on the initial states of the controllers.

The NOT gate

THE combination of these three gates with a NOT gate causes the system to behave in the opposite way described in Table 3 and the control procedures to result are—

NAND = NOT AND
NOR = NOT OR
NANDOR = NOT ANDOR

The contrast between the AND and the NAND procedures is shown in Table 4 and similar relationships between between the controllers and the system for NOR and for NANDOR are also given in the table.

TABLE 4—NAND, NOR, NANDOR

Controllers A	B	*System* AND	NAND	OR	NOR	ANDOR	NANDOR
1	1	1	0	1	0	1	0
1	0	0	1	1	0	0	1
0	1	0	1	1	0	0	1
0	0	0	1	0	1	1	0

It is not difficult to see now how ANDOR can be described in terms of AND, OR and NOT. If an AND procedure and a NOR procedure are linked by an OR operation, the ANDOR procedure is obtained, hence—

ANDOR = (AND) OR (NOR)
similarly, NANDOR = (NAND) AND (OR)

Examples

SEVERAL examples of electrical circuits are shown in Fig. 2. The three switches in (*a*) control the current flowing through the line; they are connected in series and form an AND gate. The switches in (*b*) are in parallel and form an OR gate. In (*c*), the output is ON if either A and B are ON or if C is ON, whereas (*d*) shows an example in which A and either B or C should be ON for the output

to be ON. In (*e*), when the solenoid C is energised, it moves the switch to an OFF position; when the solenoid is not energised, the switch moves to an ON position: the output for the whole circuit is ON when A and B are ON or when D is ON or when C is not ON. In (*f*), an ANDOR gate is shown; if the switches A and B are as shown (call this position ON) or if both are switched to the other position, the output is ON; if one is switched OFF, the output is OFF.

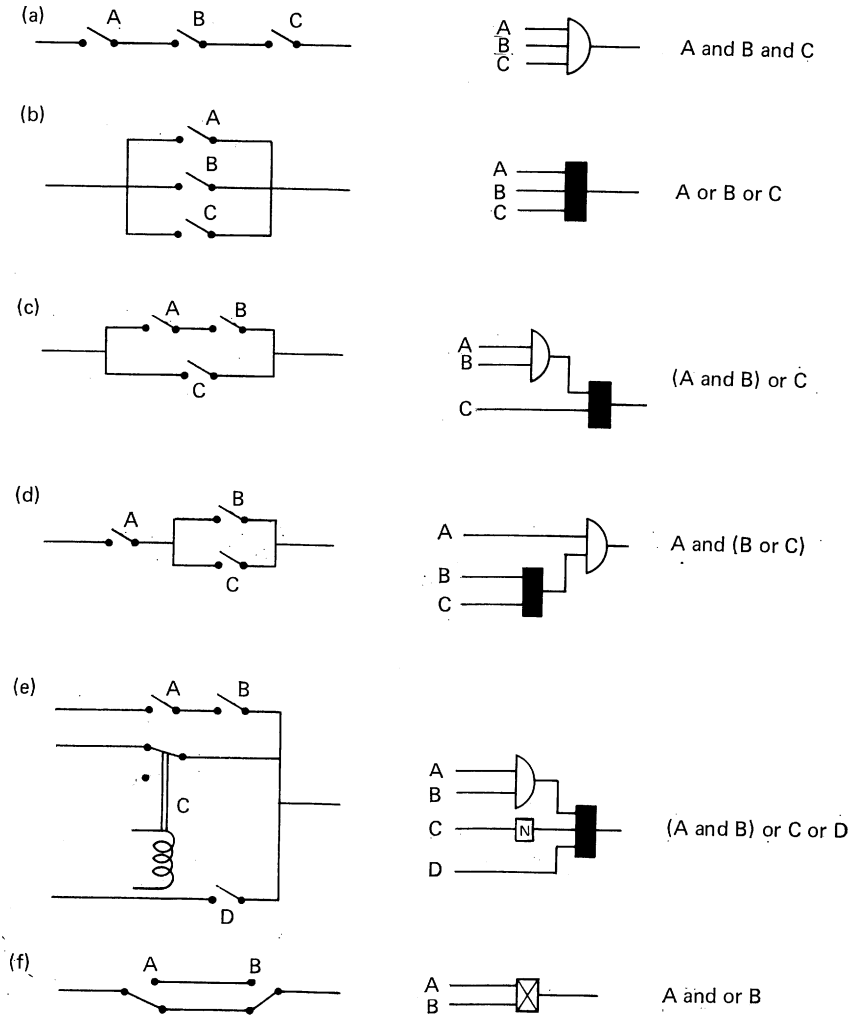

Fig. 2—Examples of electrical circuits

Control by members of a set

THE notation of Fig. 1 does not provide a convenient way of describing activities authorised by a majority vote or by agreement between a given number of members of a group of controllers. As these forms of control are common in administrative systems, a suitable notation for this purpose would be an advantage. For example, if two signatures of any three executives A, B, C are needed to validate a cheque, the procedure may be described as—

$$(A \text{ and } B) \text{ or } (B \text{ and } C) \text{ or } (C \text{ and } A)$$

but more conveniently as Z(2), which means that authorisation may be given by two members of set Z, where $Z = \{A,B,C\}$. Thus, the notation $Z(x)$ means that only x members of a set need to agree on a given action and $Z(m)$ requires a majority of the members of the set to agree.

Control for administrative systems

THE linkages forming AND, OR and ANDOR gates conform precisely to the three control procedures discussed in another paper[3]—

Control in series All the controllers connected in series need to agree on a course of action for it to take place. If one controller objects, he overrules the others. This procedure is designated as the AND gate.

Control in parallel When controllers are connected in parallel, they all have to agreed before an activity can be stopped. This procedure is a mirror image of control in series, but here each controller can overrule the others if he objects to the activity being terminated—this is, in fact, the OR gate.

Conjoint control Each controller can decree that action be taken or that an activity be stopped. In other words, when a controller acts, he activates all the other controllers to whom he is connected in parallel. If he switches OFF, it is as if all the others switch OFF at the same time; if he switches ON, they all switch ON. Here, the last controller has the last word and annuls all previous actions of the controllers. This procedure is the ANDOR gate.

Other possible linkages

TABLE 2 lists only three control procedures resulting from various linkages between two controllers, but there arre many other possible procedures as can be seen from the exhaustive list in Table 5. Each procedure is concerned with

four outcomes for the system, the outcome in each case being either state 1 or state 0.

There is one procedure for which all the outcomes are state 1

$\binom{3}{4}$, namely, four procedures, with three outcomes as state 1 and one outcome as state 0.

$\binom{2}{4}$, namely, six procedures, having two outcomes as state 1

$\binom{1}{4}$, namely, four, having one outcome as state 1 and

one procedure for which state 1 does not emerge.

The total number of possible alternative procedures is $2^4 = 16$.

A close examination of these procedures reveals that some include various operations in Boolean algebra, such as *negation* (the NOT gate, as in alternatives 10, 11 and several others), *conjunction* (the AND gate in procedure 12), *disjunction* (the OR gate in procedure 2), equivalence (this is the ANDOR gate in procedure 8) and *exclusive disjunction* (which is the negation of ANDOR,

TABLE 5—SIXTEEN PROCEDURES FOR A TWO-CONTROLLER SYSTEM

Controllers		*System*															
A	B	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	1	1	1	1	0	1	1	1	0	0	0	1	0	0	0	0
1	0	1	1	1	0	1	1	0	0	1	1	0	0	1	0	0	0
0	1	1	1	0	1	1	0	1	0	1	0	1	0	0	1	0	0
0	0	1	0	1	1	1	0	0	1	0	1	1	0	0	0	1	0
			↑ OR			↑ NAND	↑ A	↑ B	↑ ANDOR	↑ NANDOR	↑ $\overline{B}$	↑ $\overline{A}$	↑ AND			↑ NOR	

namely, NANDOR, as shown in procedure 9). Furthermore, all the other procedures may be described in terms of the basic AND, OR and NOT gates. Further observations are summarised below.

Procedure

1. The system has only one state and has therefore no need for controllers.
2. This is the OR gate (namely, disjunction) so that the procedure in this column may be described as A or B.
3. If the labels for B were to be reversed, the result in this column would be identical with that in the previous column, namely, the procedure may be described as $\underline{\text{A or } \overline{\text{B}}}$.
4. $\overline{\text{A}}$ or B.
5. $\underline{\overline{\text{A}} \text{ or } \overline{\text{B}}}$. This is also the NAND procedure, namely—

$$\overline{\text{A}} \text{ or } \overline{\text{B}} = \text{A nand B}$$

6. Here, the system adheres to the state dictated by A; controller B is redundant.

7. The system is controlled by B; here, A is redundant.
8. This the ANDOR gate; also, note that—

$$\text{A andor B} = (\text{A and B}) \text{ or } (\overline{\text{A}} \text{ and } \overline{\text{B}})$$
$$= (\text{A and B}) \text{ or } (\text{A nor B}).$$

9. This is exclusively disjunction and may be described as—

$$\text{NOT(ANDOR)} = \text{NANDOR},$$

also A nandor B = (A nand B) and (A or B).

10. This is the reverse of column 7—that is, $\overline{\text{B}}$.
11. This is the reverse of column 6—that is, $\overline{\text{A}}$.
12. Here, we have conjunction, namely, the AND gate.
13. A and $\overline{\text{B}}$; this column is also the reverse of column 4—that is, the procedure may also be described as NOT ($\overline{\text{A}}$ or B).
14. $\overline{\text{A}}$ and B; alternatively, this procedure is NOT (A or $\overline{\text{B}}$).
15. $\overline{\overline{\text{A}}}$ and $\overline{\overline{\text{B}}}$ = A nor B.
16. As in procedure 1, the controllers are redundant here.

Boolean operations

TABLES 2, 4 and 5 suggest that Boolean algebra may be helpful for formal descriptions of control procedures and relationships between controllers. Consider the following Boolean operators[1, 2] involving A and B—

Operator	*Symbol*	*Meaning*
Conjunction	A:B or AB	If A has the value 1 and B has the value 1, then AB has the value 1, otherwise AB is 0
Disjunction (also called 'alternation')	AVB or A+B	If either A or B has the value 1, then A+B has the value 1, otherwise A+B is 0
Implication	A⊃B	If either A is 0 or B is 1, then A⊃B is 1, otherwise A⊃B is 0
Equivalence	A≡B or A/B	If A and B have the same value, then A/B is 1, otherwise A/B is 0

The operators *conjunction* and *disjunction* are also referred to as *Boolean* (or *binary*) *multiplication* and *addition*, respectively. The rules of these operators are—

Boolean multiplication (conjunction)	*Boolean addition (disjunction)*
1.1 = 1	1+1 = 1
1.0 = 0	1+0 = 1
0.1 = 0	0+1 = 1
0.0 = 0	0+0 = 0

We can now construct a truth table to find the value of several expressions consisting of A and B (Table 6). The results are obtained by simply performing the operations of Boolean addition or multiplication of the respective

states of A and B (or their complements $\overline{A}$ and $\overline{B}$) in each row, as required by the operator in each expression.

TABLE 6—VALUES OF SOME BOOLEAN EXPRESSIONS

A	B	$\overline{A}$	$\overline{B}$	A+B	$\overline{A+B}$	$\overline{A}+B$	AB	$\overline{AB}$	$\overline{A}\,\overline{B}$
1	1	0	0	1	0	0	1	0	0
1	0	0	1	1	0	1	0	1	0
0	1	1	0	1	0	1	0	1	0
1	0	1	1	0	1	1	0	1	1
Procedure in Table 4		11	10	2 ↑ OR	15	5	12 ↑ AND	5	15

AND, OR, ANDOR

COMPARING the results in Tables 5 and 6, we find that—

$$A\ B = A \text{ and } B$$
$$A+B = A \text{ or } B$$

The AND gate is equivalent to conjunction (that is, the Boolean multiplication of the individual states of the linked controllers) while the OR gate is equivalent to disjunction (namely, it involves the Boolean addition of the individual states). Thus, the Boolean expression AB should be read as 'A and B' and A+B should be read as 'A or B' (rather than 'A plus B').

If we examine the definition of *equivalence*, we find that it coincides with the ANDOR gate and, if this operator is denoted by a stroke /, then A/B is read as 'A andor B' or as 'A stroke B' (but not 'A divided by B'). From procedure 8 in Table 5, it is evident that—

$$\begin{aligned} A/B &= A \text{ andor } B \\ &= (A \text{ and } B) \text{ or } (\overline{A} \text{ and } \overline{B}) \\ &= AB+\overline{A}\,\overline{B} \end{aligned}$$

and this is easily verified by performing the Boolean addition of the columns for AB and $\overline{A}\,\overline{B}$ in Table 6.

Operations involving several controllers adhere to the usual algebraic conventions, so that—

$$A+B = B+A$$
$$A\ B = BA$$
$$A+(B+C) = A+B+C$$
$$A(BC) = A\ B\ C$$
$$\text{also } A\ (B+C) = AB+AC$$
$$\text{similarly } (A+B)(C+D) = AC+AD+BC+BD$$

Alternative procedure for the two-controller two-state system

ALL THE 16 alternative procedures for two controllers handling a two-state system are listed in Table 5. This list is derived by enumerating all the possible outcomes for the states of the system and relating them to the corresponding states of the two controllers. It is not difficult to see that this is in fact an exhaustive list and covers all the possible combinations of linkages between the controllers. One way to enumerate all these various combinations is as follows—

	Alternatives	*Procedure in Table 5*
(*a*) Outcome for the system is independent of the states of the controllers	1	1
	0	16
(*b*) Outcome for the system depends on the state of one of the controllers	A	5
	$\bar{A}$	11
	B	7
	$\bar{B}$	10
(*c*) Outcome for the system depends on disjunction of the controllers	$A+B$	2
	$A+\bar{B}$	3
	$\bar{A}+B$	4
	$\bar{A}+\bar{B}$	5
(*d*) Outcome for the system depends on conjunction of the controllers	AB	12
	$A\bar{B}$	13
	$\bar{A}B$	14
	$\bar{A}\bar{B}$	15
(*e*) Outcome for the system depends on disjunction of the linkages in (*d*)	$\bar{A}\bar{B}+AB$	8
	$\bar{A}B+A\bar{B}$	9

All the other combinations in (*e*) are already covered in (*b*) and (*d*) as can be seen from the results in Table 7.

TABLE 7—RESULTS FOR DISJUNCTION OF THE LINKAGES IN (*d*)

	AB	$A\bar{B}$	$\bar{A}B$	$\bar{A}\bar{B}$
AB	AB			
$A\bar{B}$	A	$A\bar{B}$		
$\bar{A}B$	B	$\bar{A}B+A\bar{B}$	$\bar{A}B$	
$\bar{A}\bar{B}$	$AB+\bar{A}\bar{B}$	$\bar{B}$	$\bar{A}$	$\bar{A}\bar{B}$

Each cell in this table is derived by performing the Boolean addition for the corresponding row and column linkages. For example, the result for the second row and the first column is $A\bar{B}+AB = A(\bar{B}+B) = A$. The ten results (the cells above the diagonal are a mirror image of the cells below) include the four listed in (*b*) and the four listed in (*d*), leaving only two outcomes for (*e*). It can be shown that the outcomes for the system that depend on disjunction of the linkages in (*c*) have already been listed; similarly, that the outcomes that depend on conjunction of the alternatives in (*c*) or in (*d*) have all been accounted for.

Sequential links

A NOTABLE feature of Boolean algebra is that all the elements in an expression are assumed to act simultaneously, so that there is no merit in the order in which they appear in a given gate, thus—

$$\begin{aligned} A \text{ and } B &= B \text{ and } A \quad \text{that is, } AB = BA \\ A \text{ or } B &= B \text{ or } A \quad \text{that is, } A+B = B+A \\ A \text{ andor } B &= B \text{ andor } A \text{ that is, } A/B = B/A \end{aligned}$$

In circuit theory, the sequence of operation of controllers is generally of no importance, since it is the final state of the controllers that determines the state of the system and this is illustrated in the examples in Fig. 2. Besides, there is no question of hierarchy in a procedure such as A and B; both controllers are equal in status and in their ability to affect the state of the system. This is not to suggest that they necessarily use this ability equally well or that they act with equal frequency. If A is quicker than B to discern changes in the environment and reacts accordingly, then the procedure A and B results in B behaving comparatively sluggishly and A doing most of the work of switching ON and OFF. Nevertheless, both have *equal opportunity* in this control procedure and this is what is referred to in an earlier paper[4] as first-order control.

When we examine administrative systems, we find that time-independent gates are not adequate to cover the variety of procedures that can be designed and it is therefore necessary to add the time dimension or the sequence in which controllers are required to perform their duties. It is suggested that the requirement for such a sequence can be denoted by an asterisk in the control expressions (namely, and*, or*, andor*). This notation has the following meaning—

A and* B	A acts before B; if A is ON, the matter is referred to B and, if he is ON, the output is ON, otherwise it is OFF; is A is OFF, however, there is no need to refer the matter to B, because his action will not affect the outcome.
A or* B	A acts before B; if A is ON, the output is ON and there is no need to refer to B; if A is OFF, however, then the matter is referred to B, who by switching to ON will overrule A.
A andor* B	A acts first, but, whatever he decides, the matter is referred to B, who may acquiesce or reverse A's decision; the state of A signifies the state that he wishes the output to be at (that is, he is ON when he wishes the system to be ON and OFF for the system to be OFF); the state of B signifies his reaction to A's decision: B is ON when he approves and OFF when he disapproves. The resulting state of the system is then derived from Table 2, in which the state ON is denoted by 1 and OFF by 0.

Sequential links can be described by Boolean expressions similar to those used for time-independent links, except that an asterisk is added to denote the sequence in which the controllers have to perform—

$$A^*B = A \text{ and}^* B$$

$$A \overset{*}{+} B = A \text{ or}^* B$$

$$A/^*B = A \text{ andor}^* B$$

The examples in Table 8 perhaps illustrate the way in which such a language can be used to describe administrative procedures. In these examples, $Z_1 = \{n_1$ known members$\}$ and $Z_2 = \{n_2$ known members$\}$; the two sets need not be mutually exclusive.

It should be noted that, for a given control procedure, the correspondence between the composite state of the controllers and the state of the system is not dependent on whether the gates in the procedure are time-independent or sequential. For any given set of states of the controller, the result for the system would be the same, as can be verified by comparing the summary in Table 9 for sequential gates with that in Table 3 for independent gates. In other words, the algebraic rules for computing the outcome of any procedure that involves sequential gates are precisely the same as when the notations for sequential performance of controllers are deleted. The sequential procedure allows a controller to be ignored, however, when it is evident that he can have no effect on the outcome.

TABLE 8—EXAMPLES OF CONTROL PROCEDURES

Example	*Activity*	*Authorisation*	*Notation*	*Boolean notation*
1	Payment of accounts	Director A or B or anyone of the group Z_1	A or B or $Z_1(1)$	$A+B+Z_1(1)$
2	Appointment of executives	3 members of committee Z_2, then approval by director B	$Z_2(3)$ and* B	$Z_2(3)^*$ B
3	Requisition of materials	Any member of Z_1 or two of Z_2 or B	$Z_1(1)$ or $Z_2(2)$ or B	$Z_1(1)+Z_2(2)+B$
4	Dealing with a customer's complaint	Any two members of Z_1; if the complaint is dismissed, it is then referred to B, who may decide otherwise	$Z_1(2)$ or* B	$Z_1(2)^*+B$
5	Major capital expenditure	Majority agreement by Z_1, but either A or B may overrule this decision	$Z_1(m)$ andor* (A or B)	$Z_1(m)/^*(A+B)$

Consider, for example, the following three procedures—

A and B . . . . . (*1*)
A and* B . . . . . (*2*)
B and* A . . . . . (*3*)

where the probability of A being ON is $p_A = 0{\cdot}8$ and the probability of B being ON is $p_B = 0{\cdot}6$. Let us assume that the states of the two controllers are independent of each other. The probability of the system being ON is 0·48 for each of the three procedures. In (*1*) and (*3*), B is required to be ready to act all the time, whereas he is required in (*2*) to act only 80 per cent of the time (since for the rest A is OFF and then the system is OFF, whether B wishes it to be at OFF or not). As for A, in the first two procedures, he acts all the time; in the third, only 60 per cent of the time. In reality, of course, the states of the controllers may be highly correlated, so that the probability of the system being ON may be much higher than 0·48, but the essential argument remains valid—namely, that the sequential linkages AND* and OR* dispense with the services of a controller when these services are not expected to be effective.

The definitions of the sequential links AND*, OR*, ANDOR* and the summary in Table 9 contain an interesting allusion to the way in which the state of controller B is to be interpreted. *In* AND* *and in* OR*, *the state of B signifies the state that he wishes the system to be in.* From Table 9, it is clear that whenever B points to 1 the system is at 1 and whenever B points to 0 the system follows suit, although—as we have seen—there are circumstances in which B is not called upon to act at all. *In* ANDOR*, *however, the state of B signified his agreement* (denoted by 1) *or disagreement* (denoted by 0) *with A's decision.* These observations are summarised in Fig. 3.

TABLE 9—RESULTS FOR SEQUENTIAL LINKS

Link	*Controller* A	B	*System*
A and* B	1	1	1
	1	0	0
	0	–	0
A or* B	1	–	1
	0	1	1
	0	0	0
A andor* B	1	1	1
	1	0	0
	0	1	0
	0	0	1

It should be recalled that the ANDOR (and likewise the ANDOR*) link allows *either* controller to change the state of the system. If, irrespective of the initial

state of the system, B's will must prevail in a sequential link, then ANDOR* proceeds as follows—

1. Suppose that initially the states are—

A	B	System
1	1	1

An event occurs that requires A's decision, which can be signified by 1 or 0, depending on which state he wishes the system to be in after the event. When A has made his decision, B can either agree (denoted by

Controller	AND*	OR*	ANDOR*
A	State of controller corresponds		
B	to state he wishes the system to be in		1—if he agrees with A 0—if he disagrees with A

Fig. 3—The meaning of the controllers' states

state 1) or disagree (state 0) and the results for the system are shown below—

A's decision	*B's reaction*	*Comment*	*Result for the system*
1	1	No change	1
1	0	A considers a change unnecessary, but B overrules him	0
0	1	A wants a change, B agrees	0
0	0	A wants a change, B disagrees; result is no change	1

2. Suppose now that the initial conditions are—

A	B	System
0	0	1

After an event occurs, there are again the same four possibilities as enumerated in *1.*

3. Suppose the initial conditions are—

A	B	System
1	0	0

After an event, the four possibilities are—

A's decision	*B's reaction*	*Comment*	*Result for the system*
1	1	A believes the system should change to 1 (he held this view before the event) and B agrees	1
1	0	A wants a change, B disagrees; result is no change	0
0	1	A wants the system to remain in its present state and B agrees; result is no change	0
0	0	A wants no change, but B overrules him	1

4. Finally, if the initial conditions are—

A	B	System
0	1	0

the four possibilities would be similar to *3*, with the same results for the system.

Thus, irrespective of the initial state of the system, the correspondence between the controllers and the system complies with that stated in Tables 2 and 9. This conclusion can be easily verified for the AND* and OR* links.

Hierarchical aspects of control

THE introduction of sequential links provides an interesting new angle of looking at some facets of hierarchical control. The link AND* implies hierarchy in the sense that in A and* B the controller A is required to act in the first instance, but for his action to take effect he needs B's approval; if A does not wish to act, however, the matter is not even referred to B. In this case, B can overrule A only *when the latter decides to act*, but not otherwise, as shown in Table 9.

The link OR* has a similar hierarchical connotation, except that in A or* B the controller B does not need to confirm A's decision to act; but, if A decides not to act, then (and only then) B is approached and he must decide whether to uphold or to reverse A's decision. The link ANDOR*, however, requires that every decision of A is referred to B.

In addition to the way in which sequential linkages affect the participation of the individual controller in the decision-making process, they may well affect the type of decisions made by frontline controllers. For example, in the procedure A or* B, controller A knows that his decisions are subject to a close scrutiny by B only when A is OFF and, if he wishes to avoid such scrutiny, his judgment may well be impaired, resulting in his decisions being more biased towards the ON state than he would have been had he acted as a single controller. Such situations are not uncommon in administrative systems when the frequency of

disagreement between A and B (when A is OFF and B is ON) is interpreted as a measure of A's incompetence or when A begins to anticipate B's decisions and ceases to switch to OFF if he expects B to overrule him.

It appears then that there are two distinct aspects of hierarchy. The first relates to the relative frequency with which a controller is called upon to act: if two controllers A and B are assigned to a system, B's position may be considered more privileged if he need not handle every problem that the system is presented with. Controller A is the one to act in the first instance and there are matters that he can handle satisfactorily without reference to B, whose involvement is therefore less frequent than A's.

The other aspect of hierarchy is that of authority or the ability of one controller to overrule decisions of others. Thus, B may be regarded as superior to A if B has the authority to reverse A's decisions, but not vice versa.

Consider the various linkages that were discussed earlier.

1. Time-independent linkages

AND Each controller has the power to stop an activity.

OR Each controller has the power to start an activity.

ANDOR Each controller can either start or stop an activity.

In addition, each controller can act as frequently as any other. There is therefore no special privilege (either in terms of frequency of action or in terms of authority) that any one controller enjoys and that is denied to others. Similarly, the procedure $Z(x)$ for control gives equal opportunity for all members.

2. Sequential linkages (for two controllers)

AND* In A and* B, the latter acts less frequently than the former, also B can overrule A, but only for certain decisions of A.

OR* Here, the position is similar to the AND* linkage, but the type of decisions that are referred to B are different.

ANDOR* Both controllers are called upon to make decisions in every case, but, while B can overrule A (in A andor* B), A has no authority to overrule B. Thus, B enjoys a privileged position of authority, though not that of lower frequency of action.

Clearly, then, time-dependent linkages introduce hierarchical control. Linkages AND* and OR* remove B from the front line and give him some authority over A; ANDOR* gives B full authority, but to exercise it he must scan *all* the decisions of A. It would seem that remoteness from the scene of action and complete authority are not compatible; to achieve the former, some of the latter must be sacrificed.

Sequential gates are, incidentally, relevant in designing training programmes. When a trainee A is connected to a trainer through an AND* or OR* linkage, only some of A's actions are examined by B, either those that are more

important to the working of the system or those for which A's training is considered to be incomplete, whereas the ANDOR* linkage calls for all actions of the trainee to be checked by the trainer.

The multi-role controller

A CONTROLLER need not be confined to looking after the performance of a single system or a single task. He may be assigned to control several systems and his role may change from system to system or from task to task and so may his relationships to other controllers change, depending on the control procedures in force.

Take as an example two controllers A and B who jointly supervise three systems and suppose the controllers are linked as follows—

For system 1	A and B
For system 2	A and* B
For system 3	B or* A

The role of A in relation to B is different for each of the systems. In this example, even hierarchical relationships are not maintained. In system 1, no hierarchy is specified; in system 2, B has a hierarchical advantage; but, in system 3, A enjoys an advantage.

In establishing what hierarchical relationships exist between two managers in an organisation, it is therefore necessary to list their control linkages with respect to the many tasks that they are associated with; the wider the variety of such linkages, the more difficult it is to describe the relationship between the managers in a simple and concise statement. Such a state of affairs also raises the problem of the 'carryover effect'. Human controllers do not behave like automatons that are capable of switching from one mode of control to another without any after-effects. An ambivalent hierarchical relationship between two controllers that depends on how they are connected at any moment in time may well lead to an erosion of the intended procedures, whereas a more consistent hierarchical relationship reduces the frequency with which each controller needs to switch and to adapt to a new role and this is therefore more conducive to the maintenance of the designed procedures.

Multi-control procedures for the the same system

FOR any given set of circumstances, only one control procedure should apply, otherwise ambiguities may arise over the relationships between controllers and their system. This does not mean that all such relationships need to be determined by a single procedure. There could be certain circumstances that require one procedure to be used and others that call for another. So long as the two sets of circumstances are mutually exclusive, only one control procedure will apply at any one time.

Consider the following example. Two controllers are assigned to a system, but the linkages between the controllers depend on the circumstances, three classes of which are identified—

1. Quality characteristics that define the performance of the system need not be too stringent and the control procedure is then A or B.
2. Quality characteristics are not too stringent, but, because B has many other duties to attend to, the procedure becomes A or* B.
3. Quality characteristics are stringent and it is therefore necessary to institute the procedure A andor* B.

In this example, the two controllers are linked in three different ways, depending on the prevailing circumstances. It is as if a control procedure of a higher level is superimposed on the system. The supercontrol procedure consists of a supercontroller X whose task is to monitor certain parameters of the environment and of the system and to identify with the aid of this information the class of circumstances that prevail (see Fig. 4). If the class of circumstances changes, the supercontroller switches to another state, which corresponds to a new control procedure that the two controllers should adopt. Thus, as long as class *1* prevails, the controllers behave according to A or B; when the supercontroller perceives that the circumstances have changed to *2*, he switches the controllers to A or* B and so on. There is a one-to-one correspondence between the classes of circumstances and the control procedures of the system.

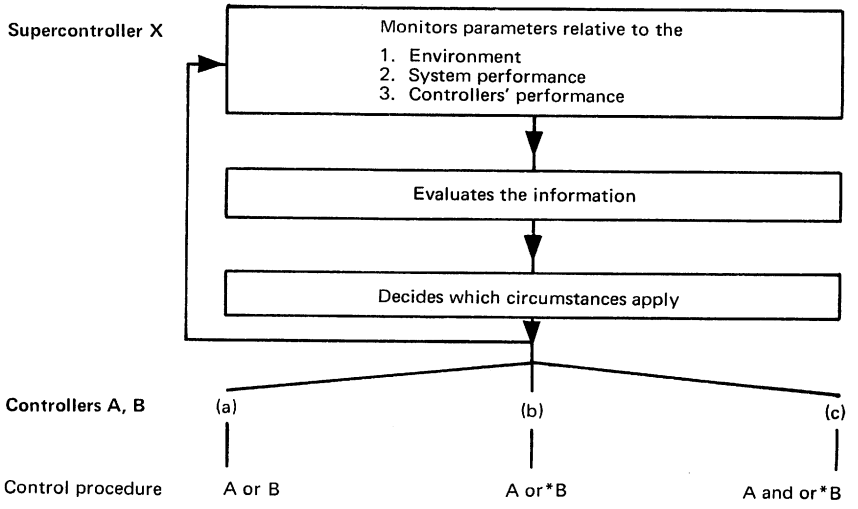

Fig. 4—An example of a supercontroller determining which control procedure to apply

Multi-controllers

AS THE number of controllers of a system increases, the number of alternative procedures that can be devised increases very rapidly. In the case of a single controller in charge of a two-state system, the number of states of the controller is two, whereas the number of possible procedures is four as shown in Table 1. For two controllers, the number of possible combinations of the controllers' states is four, whereas the number of possible procedures is 16, listed in Table 5.

Consider now the case of three controllers A, B and C. There are eight possible combinations of their states—

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

In general, the number of these combinations for n controllers is 2^n and the number of procedures that can be enumerated for the system is $2^{(2^n)}$ and this number becomes formidable even for modest values of n as shown in the following examples.

Number of controllers n	*Combinations of controllers' states,* 2^n	*Possible procedures,* $2^{(2^n)}$
1	2	4
2	4	16
3	8	256
4	16	65 536

The number of possible procedures is compounded even further when the concept of sequential links between the controllers is introduced and naturally even further when a system with more than two states is examined.

In determining control procedures for administrative systems, not all the possible alternatives need of course be considered. For example, in the two-controller, two-state system described in Table 5, two alternatives (1 and 16) need no controllers and four alternatives (6, 7, 10 and 11) need only one controller, so that the choice of a control procedure narrows down from 16 to 10 alternatives and for administrative systems possible to six (AND, OR, ANDOR, NAND, NOR, NANDOR) or even to three (AND, OR, ANDOR). Nevertheless, the point should be made that, for a system with a larger number of states or controllers, there is a plethora of alternatives to choose from in the design of control procedures.

Conclusions

I HOPE that this paper has demonstrated some of the intricacies that may be implied in assigning several controllers to look after a single task or a single system. The need for meticulous specifications for the linkages between controllers becomes self-evident and this is particularly relevant to administrative systems for which we often find that rigorous definitions are lacking or inconsistent, with the resultant uncertainties both from the point of view of the individual controllers and from the point of view of the system.

References

1. Culbertson, J. T., *Mathematics and Logic for Digital Devices* (D. Van Nostrand, New York, 1958)
2. Hoernes, G. E. and Heilweil, M. G., *Introduction to Boolean Algebra and Logic Design* (McGraw-Hill, New York, 1964)
3. Eilon, S., 'Linkages between controllers': *J. Mgt. Studies*, 1969, **6** (1), 45–52
4. Eilon, S., 'Control systems with several controllers': *J. Mgt. Studies*, 1965, **2** (3), 259–268