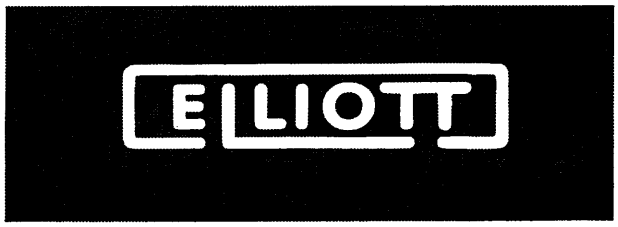
ARCH MANUAL

MANUAL
OF
THE ARCH COMPUTING SYSTEM
OF ON-LINE PROCESS CONTROL



Process Computing Division
Elliott Brothers (London) Limited
Elstree Way, Borehamwood,
Hertfordshire, England.
Telephone: Elstree 2040

A member of the ELLIOTT-AUTOMATION Group

ARCH MANUAL

1

part 1

INTRODUCTION TO THE ARCH COMPUTING

SYSTEM OF ON-LINE PROCESS CONTROL

ARCH

reg'd trade mark

ARCH MANUAL

List of Parts

Printed in England by Edward Mortimer Ltd, London and Halifax

ARCH MANUAL PART 1

contents

S1101/3 11.63

INTRODUCTION TO THE ARCH COMPUTING SYSTEM OF ON-LINE PROCESS CONTROL

1.0 THE ARCH MANUAL

The object of this manual is to familiarize the reader with the ARCH computing system of control for industrial processes. The manual is divided into a number of self-contained parts, each dealing with a particular aspect of the ARCH system.

In this part a general introduction is given to the nature of control problems and some of the approaches which may be adopted in attempting to solve them.

2.0 CONTROL PROBLEMS

Process plants, such as steel works, gas works, electricity-generating stations, paper works, chemical works, etc., are usually divided into a number of separate units, each performing a particular task, but closely connected to other units in the process.

A typical control system for such a unit consists of a number of automatic controllers, each controlling one process-variable such as temperature, pressure, flow, speed, position, etc. Each controller tries to maintain a single variable at the value set by a human operator. Assisted by gauges, strip-chart recorders and alarm annunciators, the operators attempt to co-ordinate all the controller set-values so as to maintain a uniform and acceptable product quality. Every process-unit foreman makes some attempt to ensure not only that his product is on-specification but also that it is produced in an economical way. Plant management has the unenviable task of co-ordinating each unit in the process in an attempt to obtain an efficient, integrated process-plant.

Recent advances in instrument technology and electronic engineering have materially assisted in both local and over-all control of processes. Primary measurement has been considerably improved during the last decade, on-stream analysis is at last a practical proposition and telemetering, alarm scanning and data-logging are accepted practices. There is still room for improvements, particularly in the field of primary measurement, but the major problems now are how to improve the methods of co-ordinating the settings of individual controllers in a particular unit and how to obtain better integrated control of the whole plant.

Already there are a few instances of improved co-ordination in the setting of controllers. For example, in boiler control, the relationship between combustion air flow and fuel input is maintained by a ratio controller and a demand for more steam output may be linked automatically to the set-value of the combustion air-flow controller. In this case the 'rules' defining the inter-connection of the variables are simple and can be effected automatically by the use of existing designs of automatic controller. In most processes however, the rules are much more complex and difficult to define. At present the rules used often differ from operator to operator and depend on ill-defined techniques such as 'intuition' and 'experience' as well as on purely 'logical' decisions. It is often necessary to undertake a patient investigation based on operational-research techniques even to discover what information is required about the process in order to control it.

Figure 1 shows a block diagram of an automatic process-unit controller. Inputs of information are obtained in the form of (1) the specification of the desired product and (2) from measured variables in the process-unit itself. The outputs (3) specify the set-values of the automatic controllers controlling the individual process-variables. Other inputs (4) are obtained from related process units and there may be an output of information (5) to other

unit-controllers. The controller operates on the information inputs (1), (2) and (4) according to a definite set of rules and produces the set-values (3).

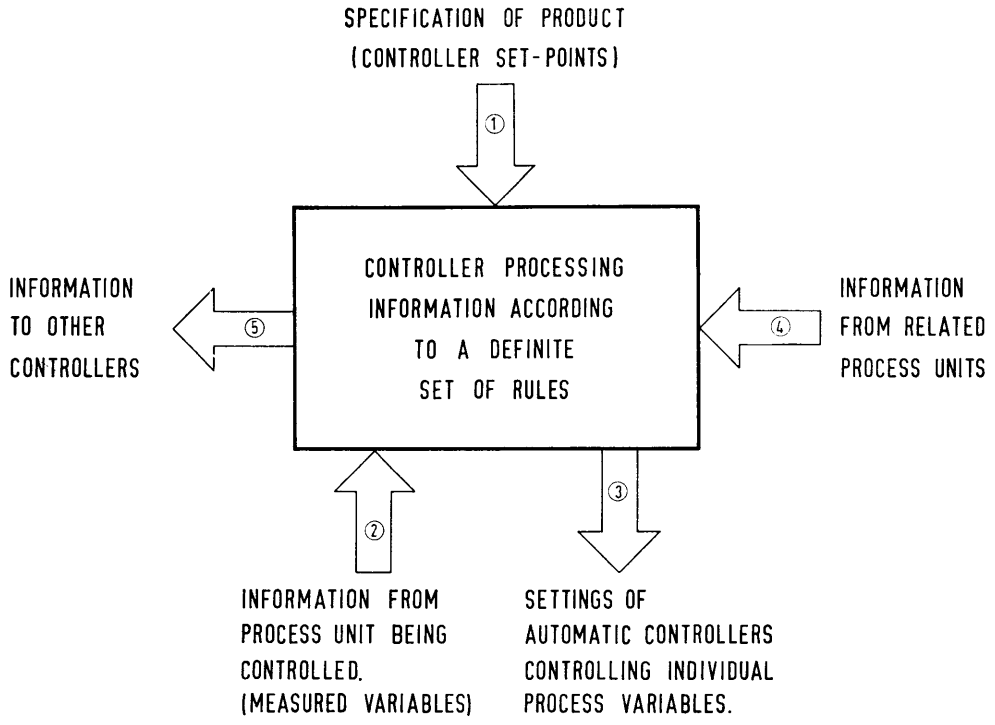


Figure 1 AUTOMATIC PROCESS-UNIT CONTROLLER

The action of this multi-variable controller is analogous to that of a single-variable controller where the measured variable is compared with the desired value (specification) and the output is derived according to simple mathematical rules which may be expressed in the form of a single three-term differential equation. The rules for processing the information in the multi-variable controller however, are almost always vastly more complicated than for a single-variable controller.

3.0 THE RULES OF CONTROL

The classical method of attempting to obtain the rules of control is to endeavour to build a 'mathematical model' of the process, relating all the measured and controlled variables to the product specification, in the form of a set of mathematical equations. A data logger may be connected to the process to obtain records of measured variables taken while the plant is made to operate in a variety of 'steady-state' and 'transient' modes. Subsequent analyses may then be carried out on electronic computers using various mathematical techniques such as cross-correlation, auto-correlation etc. It is very important to realize, however, that data loggers, computers and even the mathematical techniques themselves are only 'tools' and their blind application without process understanding is very unlikely to lead to worthwhile results.

In some processes it is possible to build a mathematical model based on the fundamental chemical or physical equations of the process. Again, there can be danger in this approach because the equations are often over-simplified to keep them to manageable proportions and this leads to wide divergences between so-called 'theory' and 'practice'. Even with simplification, the fundamental approach can lead to unmanageably complex relationships; for example, a simplified mathematical model of one plate of a distillation column may involve three

variables and that of a complete column 120 variables!

Both approaches often lead to a very complex set of mathematical equations defining the rules of control and very large, fast and expensive computers are needed to put them into effect.

A third approach is based on the premise that many existing processes are in fact controlled quite well by operators working according to rules which do not appear to involve the solution of very complex equations. An automatic controller using these rules should be able to produce about the same results as human operators. The advantages of installing an automatic controller, however, are that variations in operator performance are eliminated and a start has been made in defining the rules of control. These rules can then be improved and refined by results obtained from mathematical analysis, a knowledge of the fundamentals of the process and, most important of all, by the practical experience obtained by trying to control automatically. It may not be economical, possible or safe to perform all the tasks of measurement, control and actuation entirely automatically, but these are not reasons for doing nothing at all.

For instance, a computer-controlled installation in which the operations are partly manual and partly automatic has been in operation in a steel works in England since 1960. This installation is effecting considerable savings by determining the best way of cutting up stainless steel billets so as to meet customers' orders with minimum wastage. The customers' orders are fed into the computer-controller manually and the length of a rolled billet is measured and fed in automatically. The setting of the actuators (saws and shears) to cut up the billet is done manually according to output information from the controller displayed visually to the operators.

The controller operates according to a set of empirical rules which have been improved and refined as a result of user experience.

4.0 THE EVOLUTIONARY APPROACH

It is possible to start to improve the control of a process by the introduction of a multivariable controller to one small section of the process. This controller would be of the type shown in figure 1, and would initially receive its specification of product (desired value) from a human operator. Its output would normally control the set values of a small number of automatic controllers, although in the first instance it might be safer for it only to advise human operators on the correct setting. The information about the controlled unit process would usually be obtained automatically by direct measurements, although again some of this information may be input manually when no suitable measuring instrument is available. The rules of control may be obtained and improved as described briefly above and in more detail in part 7.

If the introduction of this first controller leads to improved control of that process unit, controllers may be introduced to other process units. Later it may be possible to introduce further high-level controllers which in turn take over the manual control of the set-values of the unit controllers. Such a system is shown in figure 2 where C_{11}, C_{12} and C_{13} are the unit-process controllers and C_{21} and C_{22} are the higher-level controllers. An even higher-level controller C_{31} might top this 'pyramid'. In this way, a 'hierarchy' of automatic control is evolved, similar in structure to that of factory management itself, the controller at the top of the pyramid C_{31} corresponding to the senior management, the controllers C_{21}, C_{22}, etc., to the junior management, controllers C_{11}, C_{12} etc., to foremen and the controllers R_{11}, R_{12}, etc., to the factory workers. This is, of course, only an analogy and the controllers do not replace the personnel referred to! Nevertheless, there is a striking similarity in the flow of information, commands passing downwards and selected information passing upwards.

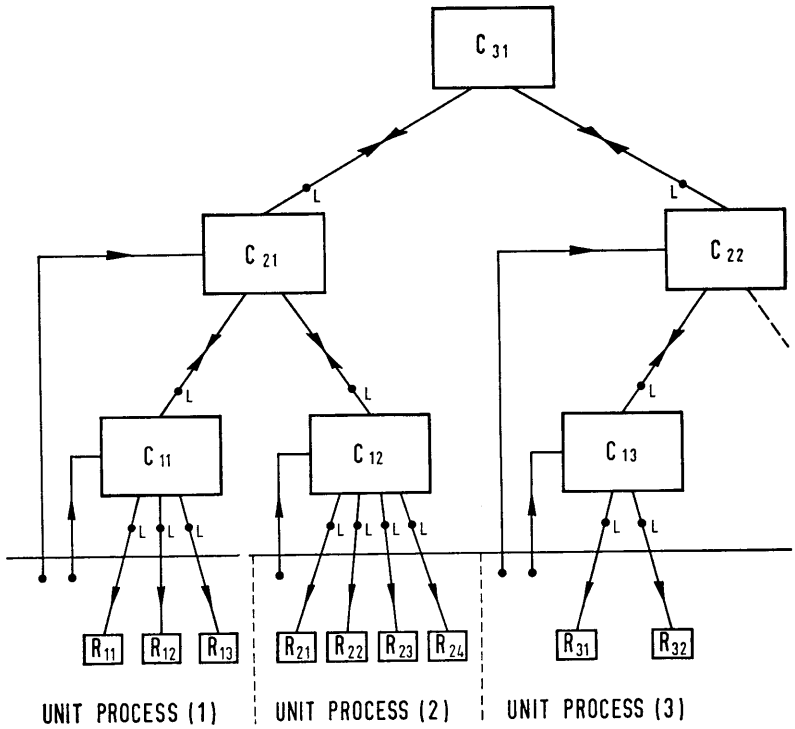


Figure 2 HIERARCHY OF CONTROLLERS

5.0 THE ARCH SYSTEM OF CONTROL

ARCH consists of a wide range of 'modules' or building bricks, which enable multi-variable controllers of the types described above to be constructed. The range is such that each controller may be 'tailored' exactly to the job in hand by the simple assembly of a suitable selection of modules. If the rules of control require to be changed in the light of experience, then the controllers may be modified by the addition or replacement of modules. 'Link' modules are available to enable controllers to be linked together in the form of a hierarchy. These Link modules, marked 'L' in figure 2, enable process operators to take over manual control at any level, as well as providing 'fail-safe' automatic links between all types of controller.

ARCH modules are essentially electronic 'computing' modules and a simple introduction to electronic computing principles is given in part 2 of this manual. This also explains the differences between analogue and digital computers. Part 3 explains analogue computing techniques in more detail and gives a basic description of ARCH analogue modules. Part 4 explains digital computing techniques in more detail and gives a basic description of ARCH digital modules. Part 5 gives a basic description of ARCH Link modules and ARCH measurement techniques. Part 6 contains a description of the programming of ARCH digital machines. Part 7 is an application section showing how various combinations of ARCH modules carry out specific rules of control. Full specifications of all ARCH modules are contained in part 8.

ARCH MANUAL

2

part 2

COMPUTING METHODS

ARCH

reg'd trade mark

ARCH MANUAL PART 2

contents

S1102/3 11.63

1.0 FORMS OF REPRESENTATION

Industrial processes are concerned with physical parameters such as temperature, pressure, weight, speed, etc. In order to be able to apply a computer to process control, these parameters must be measured and the measured values represented in a form that the computer can accept.

There are two forms of representation of measured values: '*analogue*' and '*digital*'.

1.1 Analogue Representation

A car speedometer is an example of analogue representation (in this case, of speed). The pointer of a speedometer may take up any position on the scale and sweeps through every value from zero to near full scale as the car accelerates from rest to its maximum speed. At any time, the amount by which the pointer has moved from zero, is said to be an 'analogue' of the car's speed. The feature of analogue representation is that the measured quantity (in this example, speed) is converted into another physical quantity (pointer position) in a *continuous* way, i.e. the value of the measured quantity is *continuously* represented by the value of the analogue and there is no minimum change or step required in the measured quantity to cause a change in its analogue.

In electrical process-control instruments and analogue computers the measured process-quantities (variables) are usually represented by electrical voltage or current analogues. For example, temperature is often measured by using a thermocouple which gives out a small direct voltage which is an analogue of its temperature. Pressure may be converted into a proportional electrical signal by balancing the force produced by a bellows against another produced by a direct current flowing through a coil of wire in a fixed magnetic field. The current flowing in such a coil is a direct-current analogue of the pressure applied to the bellows. In some instruments, the process variable is converted into an alternating voltage using a differential transformer. There are in fact a wide variety of 'standards' in use at the present time. In ARCH all the incoming analogue signals are converted so as to be in the range 0V to $\pm$5V d.c. before any computations are performed.

1.2 Digital Representation

An example of digital representation is the indication of distance by a car mileometer. The distance covered by the car is displayed as a number and this number increases *in steps* as the car moves, i.e. there is a minimum *step* of 1 mile needed in the measured quantity before the mileometer value changes. The mileometer is in fact a mechanical counter which counts the number of revolutions of the car's wheels (through a suitable gear-ratio of course). Similar counting techniques are used in process instrumentation; for example liquid-volume may be measured as integrated flow by counting the number of revolutions of a paddle wheel rotated by the liquid as it flows through a pipe. The length of a steel billet may be measured by counting the number of rotations of the mill rollers (with corrections to allow for slip between the rollers and the billet).

The digital representation of a quantity then is a number, often (although by no means necessarily) obtained by counting individual increments.

Numbers used in 'everyday life' are usually represented in the 'decimal' scale (scale of 10), e.g. the number 367 has three decimal 'digits' and means 3 hundreds + 6 tens + 7 units, i.e. $3\times10^2+6\times10^1+7\times10^0$. In the decimal representation each digit can have any one of ten discrete values from 0 to 9 inclusive.

It *is* possible to represent digits in an electronic computer as say, a voltage of between 0 and 9 V d.c. giving 10 values 1 V apart. Unfortunately practical limitations of circuit design tend to make this representation unreliable, and lead to complex circuits. It is much better to employ circuits having just two states of 'on' and 'off' representing the values '1' and '0' respectively. Circuits of this type (i.e. two-state) are very simple and can accept wider variations in circuit component values without error.

For this reason, digital computers use numbers represented in the binary scale (scale of 2) in which each digit has one of two possible values, i.e. 0 or 1.

For example, the number 10111 has 5 binary digits (bits) and means:

$$1\times2^4+0\times2^3+1\times2^2+1\times2^1+1\times2^0$$
$$= 16 + 0 + 4 + 2 + 1$$
$$= 23$$

The two discrete states of a digit may be represented in the computer by say, 0V and −10V d.c. respectively. Multi-digit numbers thus appear as a pattern of 10V-high pulses. It is possible to perform all the usual arithmetic operations of addition, subtraction, multiplication and division on binary numbers (see part 6).

Binary numbers are not easily recognised by human operators nor is an inexpensive converter from binary to decimal available. This has led to the use of the binary-coded decimal (B.C.D.) representation in many simple digital machines (see part 6). It is sufficient to note here that B.C.D. representation is 'redundant', i.e. requires more than the absolute minimum number of bits to represent a number, but it is more easily converted to pure decimal.

e.g. 0101 1001 (eight digits B.C.D.)=59=111011 (six digits binary).

Note that in the B.C.D. scale each decimal digit is directly represented by four binary digits.

2.0 ANALOGUE AND DIGITAL COMPUTERS

Having represented the measured variables in one form or the other, they must be read into the appropriate type of computer in order to carry out the operations necessary to achieve control.

2.1 The Analogue Computer

In the analogue computer a unit is required for each step in a calculation. A block diagram of an analogue computer is shown in figure 1.

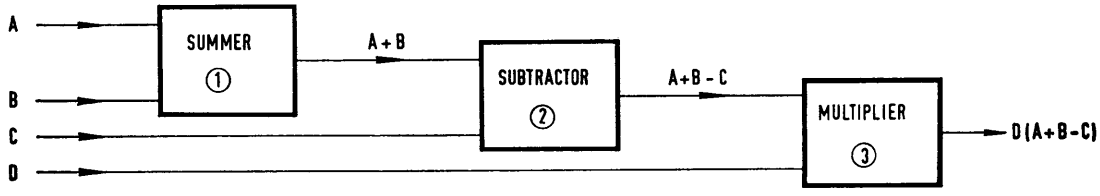


Figure 1 ANALOGUE COMPUTER

Unit (1) adds the inputs A and B together, unit (2) subtracts input C from this sum and unit (3) multiplies the result by D giving an output of D (A+B−C). The four inputs, A, B, C and D

represent process variables and the output continuously represents the answer of the calculation.

Each stage in the calculation can only be performed to a finite accuracy owing to limitations of the electronic circuits used. Errors of less than $\pm 0.1\%$ of full scale can be obtained economically for simple operations like addition and subtraction and rather greater errors occur with the more complex operations such as multiplication, division, square-root extraction, etc. As the computation becomes more complex, more and more analogue units are required and the error of the answer increases. Put another way, to obtain an answer of a given accuracy, the accuracy required of each stage in the computation increases as the computation becomes more complex. It is possible to obtain high over-all accuracies with errors of less than $\pm 0{\cdot}1\%$ of full scale but the cost rises very rapidly indeed with the degree of accuracy required.

2.2 The Digital Computer

In the digital computer, a completely different approach is taken to that of the analogue computer. All calculations are carried out in one relatively expensive arithmetic unit which is used over and over again (in time) to carry out each stage in a calculation. An 18-bit binary arithmetic unit has a maximum resolution of ± 1 part in 2^{18} i.e. a maximum error of approximately $\pm{\cdot}0004\%$ of full scale. It could therefore be used to perform one thousand additions say, before the maximum error in the result rises to $\pm 0{\cdot}4\%$ of full scale. An arithmetic unit with twice the number of bits costs only about twice as much, but has a maximum resolution of ± 1 part in 2^{36}, i.e. about $\pm{\cdot}0000002\%$ of full scale, and could be used for a million additions before the maximum over-all error rises to $\pm 0{\cdot}2\%$ of full scale.

Thus in digital machines very complex calculations can be performed to a high degree of accuracy. The number of times the arithmetic unit is used is proportional to the complexity of the calculation and thus the time taken to perform a given calculation is proportional to its complexity.

2.3 Relative Merits

Figure 2 shows, in a qualitative way, cost against complexity of calculation for the two types of computer.

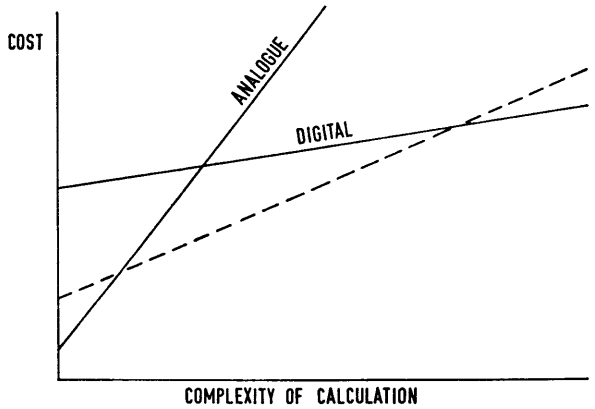


Figure 2 COST CHARACTERISTICS

It can be seen that analogue computers have a low 'fixed' cost and that their cost increases in proportion to the calculation complexity because the amount of equipment required increases as the calculation becomes more complex. Digital computers on the other hand, have a large fixed cost and thereafter the cost increases much more slowly with complexity of calculation. This fixed cost corresponds to the cost of the arithmetic unit and associated sequencing equipment which is always required even for simple calculations. The proportional part of the cost is mainly the cost of storing the numbers used in the calculations, a factor which tends to increase with complexity (see part 4). It can be seen that relatively simple problems are often best performed in an analogue way whereas digital techniques have advantages at greater degrees of complexity.

The dotted line on figure 2 represents cost against complexity of ARCH digital computers. From this it can be seen that the 'fixed cost' associated with digital computers is considerably reduced. This is due to the 'modular' form of these machines, a feature described in detail in part 4 of this manual.

ARCH MANUAL

3

part 3

ANALOGUE TECHNIQUES

ARCH

reg'd trade mark

ARCH MANUAL PART 3

contents

appendices

S1103/3 11.63

ANALOGUE TECHNIQUES

1.0 INTRODUCTION

The basic principle of analogue representation and computation has been described in part 2.

In practice, the d.c. electrical analogue is undoubtedly the most convenient and allows the full exploitation of the advances which have been made in the field of electronic engineering. Problem variables are represented by the instantaneous magnitude of a d.c. reference voltage and in dynamic problems the independent variable is represented by time.

The analogue computer is ideally suited to the solution of differential equations both linear and non-linear and, in its general-purpose role finds its greatest application to problems which involve complicated sets of such equations. Its ability to solve these at high speed and the ease with which changes of coefficients can be made are responsible for its widespread use as a systems-design tool.

In on-line process control applications, analogue techniques are most useful where continuous calculations are to be performed on a modest number of process variables, particularly where non-linear relationships must be established.

This part describes in detail the methods used in ARCH to perform the various operations which are essential for an on-line analogue computer.

2.0 MATHEMATICAL OPERATIONS

The realisation of a computer requires devices which are capable of performing mathematical operations upon the voltages which represent the variables.

The operations which may be required for process-control applications are classified below.

(a) **Scaling.** i.e. multiplication or division by constant coefficients.

(b) **Summation.** i.e. addition or subtraction of variables and/or constants.

(c) **Integration** with respect to time.

(d) **Special transfer functions** such as phase lead or lag.

(e) **Multiplication** and **division** of variables.

(f) **Linearisation.** i.e. square-root extraction, correction of transducer characteristics, etc.

(g) **Function generation.** i.e. squares, logarithmic and empirical laws, etc.

(h) **Analogue storage.**

These operations will each be considered with particular reference to the methods used in ARCH. For those readers who wish to obtain more detailed information regarding the general principles a bibliography is given at the end of this part of the manual.

2.1 The Operational Amplifier

Linear operations are performed by means of an operational amplifier, an electronic device consisting of a very-high-gain d.c. amplifier and a network of passive input and feedback impedances as shown in simple schematic form in figure 1.

The triangle represents the d.c. amplifier, the base corresponding to the input terminal and the apex to the output terminal. Z_1, Z_2 etc. are input impedances and Z_f is the feedback impedance. The arrangement is shown as a four-terminal network with all the voltages referred to a common earth line. In subsequent diagrams this earth line is omitted, it being understood

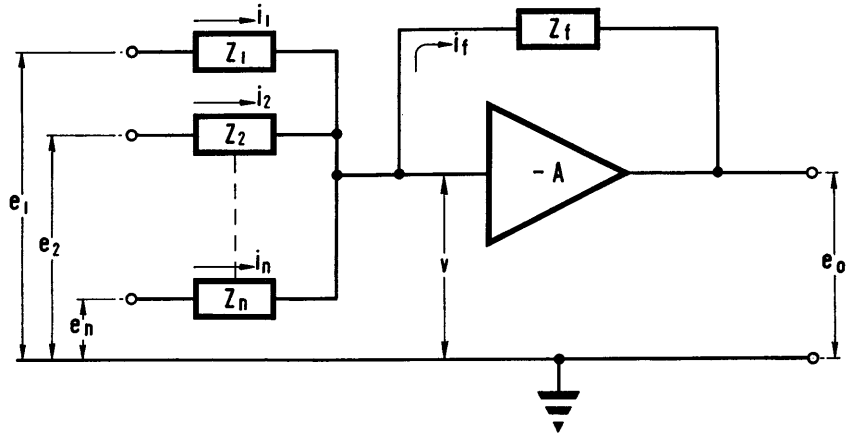


Figure 1 OPERATIONAL AMPLIFIER WITH GENERALISED IMPEDANCES

that all voltages are with reference to earth. The combination is a negative-feedback amplifier, so that there is necessarily a change of sign between input and output. If the gain of the amplifier is very high, the voltage v required at the input terminal to drive the output to its limit is very small and the input terminal is at 'virtual earth' potential. If also the amplifier input current is very small, a simple analysis may be made to give the relationship between the input and output voltages of the operational amplifier. This relationship is known as the 'transfer function' of the operational amplifier.

The sum of the currents flowing into and away from the amplifier input terminal is equal to zero, thus:

$$i_1 + i_2 + \ldots\ldots i_n = i_f$$

$$\therefore \quad \frac{e_1}{Z_1} + \frac{e_2}{Z_2} + \ldots\ldots \frac{e_n}{Z_n} = -\frac{e_o}{Z_f}$$

Re-arranged, this yields:

$$e_o = -Z_f\left[\frac{e_1}{Z_1} + \frac{e_2}{Z_2} + \ldots\ldots \frac{e_n}{Z_n}\right]$$

This idealised transfer function may be used to establish a whole range of linear relationships by choosing suitable input and feedback impedances. The choice is usually restricted to resistors and capacitors since these components can be readily obtained to the desired order of accuracy.

It must be remembered that this simple analysis is based upon the following assumptions:

(a) infinite gain in the d.c. amplifier

and (b) zero amplifier input current.

In practice these are not completely true and there are other sources of error such as amplifier voltage drift and finite bandwidth. The ARCH d.c. operational amplifier has been designed to minimise all such errors when used in accordance with a set of simple rules which are given in a later section.

2.2 Scaling

The arrangement used for multiplication by a constant coefficient is shown in figure 2.

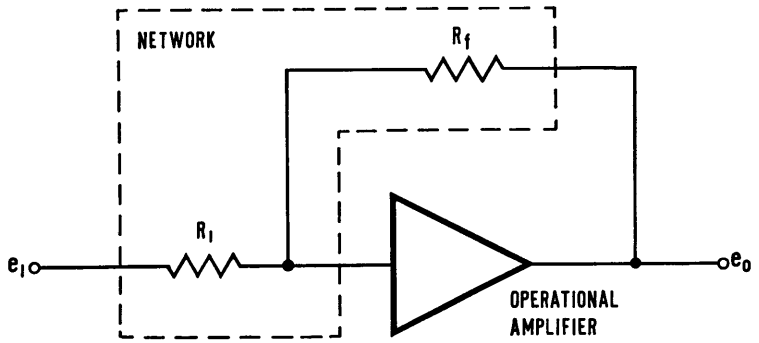


Figure 2 SCALING

The input and feedback impedances are pure resistors, R_1, R_f and from the general transfer function the output of the amplifier is given by:

$$e_o = \frac{-e_1 . R_f}{R_1}$$

The ratio of feedback to input impedance is chosen to give the desired relationship between the input and output voltages. If resistors of equal value are chosen the unit is a unity-gain sign-changer or phase-inverter, a unit which is frequently required in a computing system.

In practice it is often necessary to provide easily-variable coefficients or scale-factors and this is achieved by the use of a 'coefficient potentiometer' as shown in figures 3a and 3b.

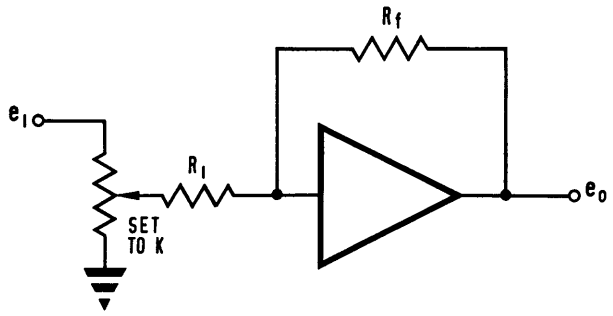


Figure 3a MULTIPLICATION BY A CONSTANT

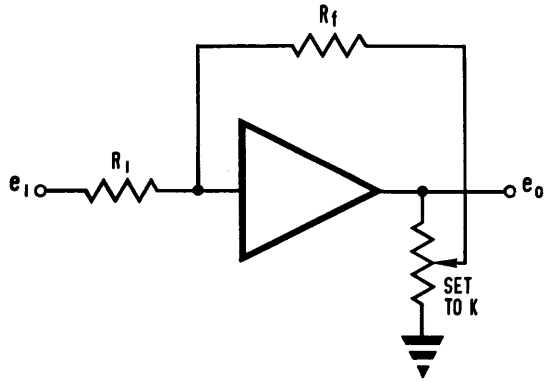


Figure 3b DIVISION BY A CONSTANT

Considering figure 3a, if the potentiometer slider is unloaded and is set to $K\%$ of full scale, then the transfer function is given by:

$$\frac{e_o}{e_1} = \frac{-K.R_f}{R_1} \qquad \text{where } K \text{ is in the range 0 to 100\%.}$$

The ratio $\frac{R_f}{R_1}$ is chosen to give the desired maximum value of the coefficient and the actual value is set on the potentiometer. The total resistance of the potentiometer should be much smaller than that of the input resistor in order to avoid loading errors, particularly when it is desired to set the coefficient directly from the potentiometer dial.

Division by a constant is performed by the arrangement shown in figure 3b.

The transfer function is given by:

$$\frac{e_o}{e_1} = \frac{-R_f}{KR_1}$$

The ratio $\frac{R_f}{R_1}$ is chosen to give the maximum value of $\frac{1}{K}$ and the actual value of K is set on the potentiometer. Low potentiometer-settings should be avoided and K must never equal zero since this removes the feedback and gives rise to instability.

2.3 Summation

The arrangement used for addition or subtraction is shown in figure 4.

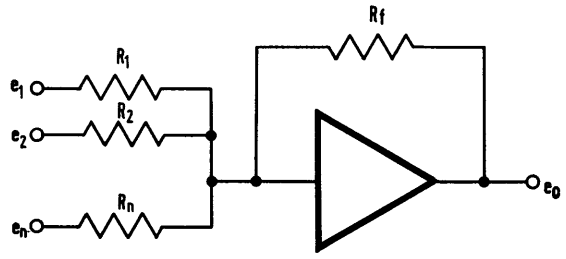


Figure 4 ADDITION OR SUBTRACTION

From the general transfer function the relationship is given by:

$$e_o = -\left[\frac{e_1 R_f}{R_1} + \frac{e_2 R_f}{R_2} + \cdots\cdots \frac{e_n R_f}{R_n}\right]$$

The output voltage is thus the algebraic sum of the input voltages, each scaled by a constant coefficient. Any or all of the input channels can have variable coefficients using potentiometers as previously described. Subtraction is obtained merely by changing the sign of the appropriate input voltage or voltages.

2.4 Integration

Integration with respect to time may be performed by replacing the feedback resistor by a capacitor as shown in figure 5.

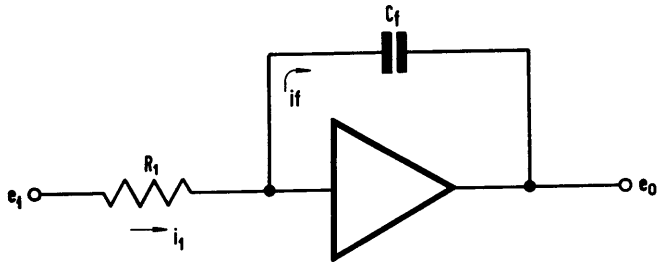


Figure 5 BASIC INTEGRATION

The impedance of the capacitor is given by $\frac{1}{pC_f}$ where p is the operator $\frac{d.}{dt}$

From the general transfer function the relationship

$$e_o = \frac{-e_1 . \frac{1}{pC_fR_1}}{}$$ is obtained, which corresponds to integration.

An alternative approach for those readers unfamiliar with operational methods is as follows:

$$i_1 = i_f$$

$$\therefore \quad \frac{e_1}{R_1} = -C_f . \frac{de_o}{dt}$$

and $$\therefore \quad \frac{de_o}{dt} = \frac{-e_1}{R_1C_f}$$

Integrating throughout gives:

$$e_o = -\frac{1}{R_1C_f} \int e_1 . dt$$

The quantity R_1C_f is called the integrator 'time-constant' and the reciprocal is called the integrator 'gain'. If the dimensions of the resistance and capacitance are megohms and microfarads respectively, then the unit of time is the second.

It is often necessary to allow for the constant of integration; this is achieved by the application of an initial charge to the feedback capacitor so that the output voltage assumes the correct value at time $t=0$ i.e. at the start of the computation.

The integral of the algebraic sum of a number of input signals can be obtained by the addition of extra input resistors. The standard ARCH integrators allow for three inputs. The output voltage is then given by:

$$e_o = -\frac{1}{C_f} \int \left(\frac{e_1}{R_1} + \frac{e_2}{R_2} + \frac{e_3}{R_3} \right) dt$$

A complete integrator with its control relays and initial-condition setting facilities is shown in schematic form in figure 6.

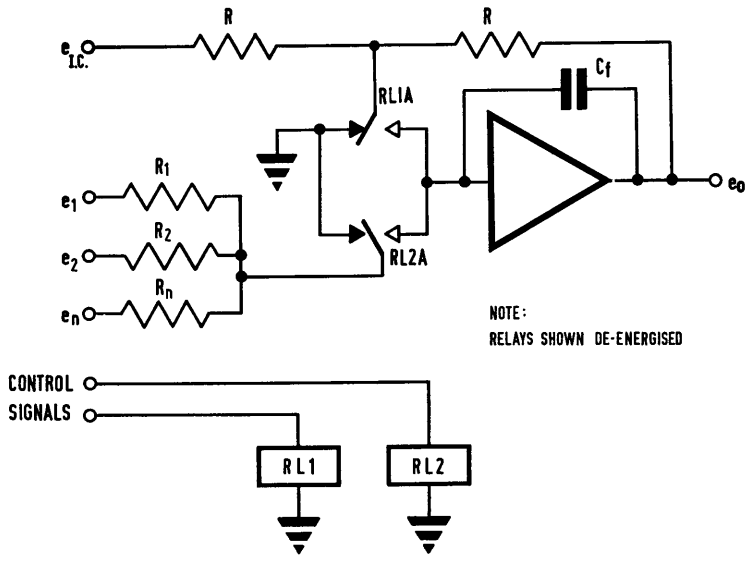


Figure 6 INTEGRATOR

Three different operating modes can be obtained by suitable programming of the control relays as indicated in Table 1.

Mode	Relay RL1	Relay RL2
(a) **Initial condition**	*energised*	*de-energised*
(b) **Integrate**	*de-energised*	*energised*
(c) **Hold**	*de-energised*	*de-energised*

Table 1

The different modes function as follows:

(a) Initial condition

The normal input resistors are disconnected from the amplifier input terminal and connected to earth. Two equal-value resistors are connected as special input and feedback impedances and a reference voltage is applied to charge the feedback capacitor to give the correct initial output voltage.

(b) Integrate

The junction of the initial-condition resistors is disconnected from the amplifier input terminal and connected to earth. The normal input resistors are re-connected and the amplifier output will change at a rate determined by the sum of the input voltages and the time-constants of each channel.

(c) Hold

The input resistors are disconnected from the amplifier input terminal and connected to earth. The amplifier output is thus clamped at the value attained immediately prior to the de-energisation of relay RL2.

The principal source of error in an integrator is the amplifier input current which causes the output voltage to change even when the inputs are zero, the rate of change of output being given by $\frac{i}{C_f}$. To minimise this error the largest possible value of capacitance should be used. Errors are also caused by capacitor leakage so that only high-quality components should be used. The ARCH integrator modules use 5μF polystyrene capacitors and with a one second

time-constant have a drift-rate of not more than 200μV per second. For extremely critical applications involving long-term integration a drift-corrector sub-module may be added which reduces the drift-rate under the same conditions to not more than 10μV per second.

2.5 Phase Lag

This example is given to illustrate the use of the general transfer function to obtain more complex transfer functions.

The arrangement used to provide a phase lag is shown in schematic form in figure 7.

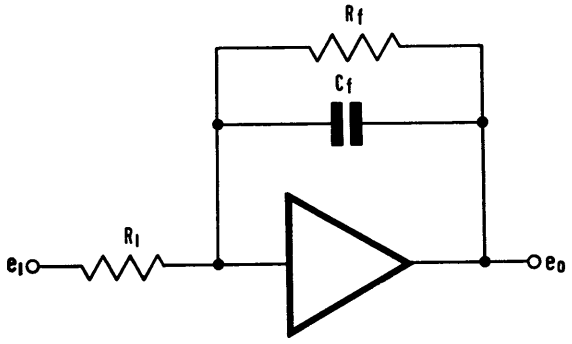


Figure 7 PHASE LAG

The effective feedback impedance is:

$$Z_f = \frac{\frac{R_f}{pC_f}}{R_f + \frac{1}{pC_f}} = \frac{R_f}{1+pR_fC_f}$$

Substitution of the effective feedback impedance into the general transfer function yields:

$$\frac{e_o}{e_1} = -\frac{R_f}{R_1}\left[\frac{1}{1+pR_fC_f}\right]$$

This transfer function can be realised by using an integrator with one input resistor connected in parallel with the feedback capacitor and permanently energising the control relay RL2. Alternatively, a capacitor can be connected in parallel with the feedback resistor of a summer.

2.6 Multiplication of Variables

The most commonly required non-linear operation is that of multiplication of variables and there are many different methods of obtaining the desired products.

Multipliers can be divided into two major classes, namely single-quadrant and four-quadrant. A single-quadrant multiplier can only accept inputs which have the same sign and the output cannot change sign. On the other hand, in a four-quadrant device both input signals can be of either sign and the output changes sign in accordance with the inputs.

In the majority of process-control applications the nature of the physical properties involved is such that only single-quadrant multipliers are required. Occasionally, four-quadrant multipliers are required, for example, for obtaining the products of error signals. For this reason the principal method of multiplication adopted for ARCH is the 'logarithmic'

type. This is a versatile method which offers extreme flexibility although it is restricted to single-quadrant operation.

The logarithmic multiplier is based upon the identity:

$$\log_a xy = \log_a x + \log_a y$$

This can be realised quite simply by means of a summing amplifier with input networks which produce currents proportional to the logarithms of the input signals. An arrangement for producing the sum of two logarithms is shown in block diagram form in figure 8a.

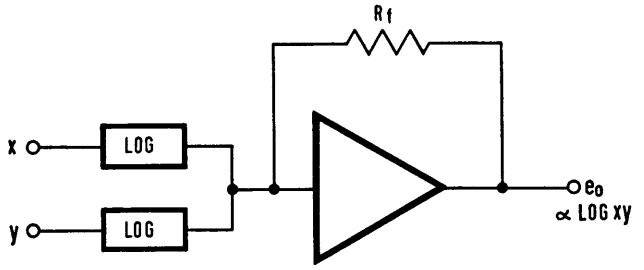


Figure 8a SUM OF TWO LOGARITHMS

If the linear feedback resistor is replaced by a network which provides a feedback current proportional to the logarithm of the amplifier output voltage, then the output of the combination is proportional to the product xy. This is shown in figure 8b – for convenience, the feedback-network is called an 'antilog' sub-module.

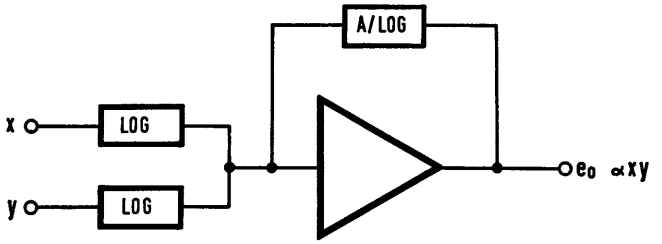


Figure 8b LOGARITHMIC MULTIPLIER

The technique is very flexible since multiple products such as xyz can be produced merely by the addition of extra input networks. Powers and roots are obtained by simple scaling, and division requires only a change of sign of the appropriate input voltage. The ARCH range includes sub-modules for +Log, −Log, +Antilog and −Antilog for the construction of a wide range of multipliers, dividers, etc. Complete multipliers and dividers are available as modules. The networks employ straight-line approximation to the functions, a technique which is discussed in a later section.

Four-Quadrant Multiplier

This multiplier operates on the 'quarter-squares' principle which is based upon the identity:

$$xy = \tfrac{1}{4}[(x+y)^2-(x-y)^2]$$

This is realised in a similar manner to the logarithmic multiplier by means of a summing amplifier and input networks. The input networks consist of function generators and passive summing networks which produce the terms $\frac{1}{4}(x+y)^2$ and $\frac{1}{4}(x-y)^2$ as currents which are summed and scaled to provide an output voltage proportional to the product xy. The arrangement used is shown in block diagram form in figure 9.

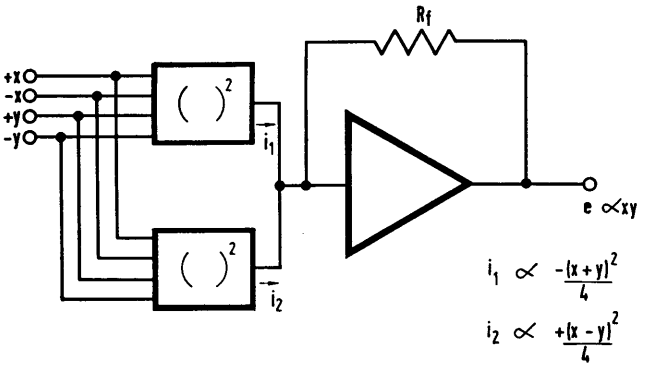


Figure 9 'QUARTER-SQUARES' MULTIPLIER

Both methods of multiplication have some common features. Standard operational amplifiers and networks are used and both methods lend themselves to modular construction. They are both fast multipliers, the response being determined solely by the operational amplifier.

2.7 Function Generation

The basic method of generating non-linear functions such as the logarithmic and square laws required for the multipliers, transducer linearisation and empirical laws is synthesis of the function by means of linear segmental approximations as shown in figure 10.

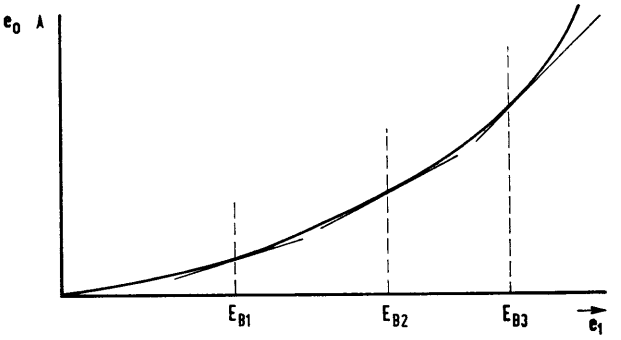


Figure 10 APPROXIMATION TO A CURVE

The points E_{B1} etc., at which the slope changes are known as 'break-points' and are realised by means of biassed-diode networks arranged as input or feedback impedances for an operational amplifier as shown in figure 11.

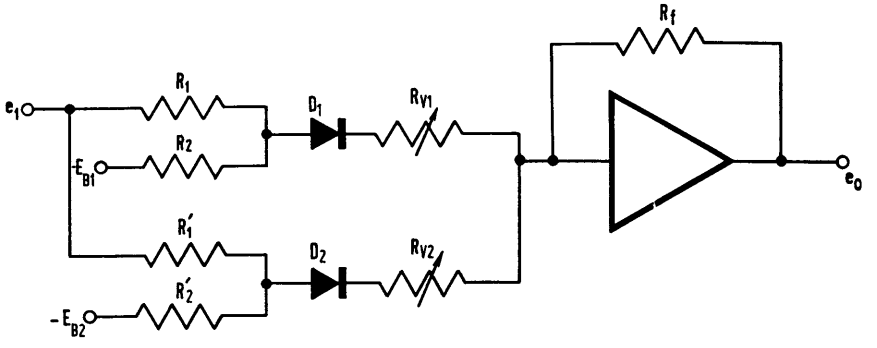


Figure 11 BASIC FUNCTION GENERATOR

As shown, the diodes will only conduct when the input signal is positive and greater than a value determined by the negative bias voltage and the resistors R_1 and R_2. The variable input resistors allow the slope of each segment to be adjusted to fit the required curve. If 'N' diodes are used, all at different bias levels, the output voltage is determined by the value of the feedback resistor and the current contribution of each conducting diode.

In general, the accuracy obtained for any given function will depend upon the number and positioning of the break-points. Functions which may be approximated by this method are restricted to continuous single-valued functions with maximum slopes of about 85 degrees.

ARCH function-generator networks use silicon diodes and precision wire-wound resistors and, to ensure the maintenance of accuracy throughout the working temperature range, the diodes are temperature-controlled.

2.8 Analogue Storage

Cases sometimes arise where it is necessary to store (memorise) the value of an analogue quantity for a short period of time. This can be achieved by the use of an ARCH Sample/Hold module. The arrangement used is shown in schematic form in figure 12.

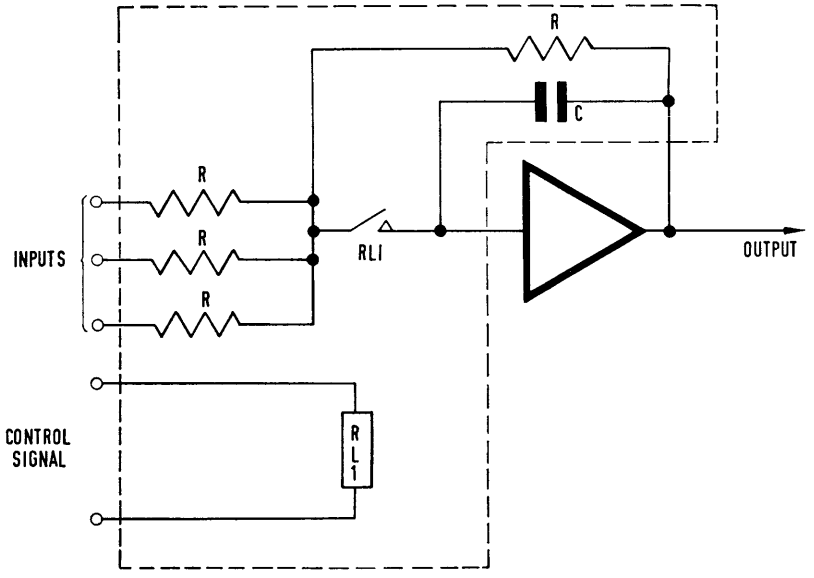


Figure 12 ANALOGUE STORAGE

The arrangement is essentially an integrator which is operated only in the 'initial conditions' or 'hold' modes. When the control relay is energised, the feedback capacitor is charged so that the amplifier output is equal to the sum of the input signals (with sign change). If the relay is de-energised, the output will be clamped at this value until such time as the relay is re-energised. In practice, the amplifier drift and capacitor leakage will cause the output voltage to decay so there is a definite limit on the permissible storage time for a given accuracy. The standard module can store a signal for 25 seconds with a maximum error of 1% full scale. If higher accuracy or much longer storage times are required then a drift-corrector sub-module must be added.

Storage techniques can be useful in a number of applications. For example, it is often necessary to perform a simple calculation upon a number of sets of process variables, the calculation being the same in each case. Input selection and storage allows a common analogue arithmetic unit to be used on a 'time-sharing' basis, in a similar manner to a digital computer.

2.9 General Rules

The following general rules have been embodied in the standard range of ARCH modules; however, the user should understand the rules and apply them when undertaking the construction of 'tailor-made' assemblies of sub-modules for special purposes. The derivation of the rules can be found in the books in the bibliography at the end of this part.

Summers and Scalers

1 The ratio of feedback resistance to the sum of the input resistances should be kept as low as possible to minimise drift errors.

2 The value of the feedback resistor should be kept as low as possible to minimise errors due to amplifier input current.

3 The number of input channels should be kept as low as possible to minimise errors due to the finite gain of the amplifier.

4 The input signals should be as large as possible within the limits of the amplifier output voltage range, i.e. ± 5 volts.

5 For critical applications a drift-corrector sub-module should be used.

6 When coefficient potentiometers are used, the resistance values should be chosen to minimise loading errors.

Integrators

1 The feedback capacitor value should be as large as possible to minimise the effect of amplifier input current.

2 The number and gains of input channels should be kept as low as possible to minimise amplifier finite-gain errors.

3 For long-term integration a drift-corrector sub-module must be used.

3.0 PROBLEM SOLUTION BY ANALOGUE COMPUTERS

3.1 Introduction

The application of analogue computing methods to the solution of specific problems involves a number of well-defined tasks. Firstly, the problem must be stated in the form of mathematical relationships, usually in the form of algebraic or differential equations obtained from a study of the physical system. It is vital that at this stage of the proceedings the objectives are clearly stated including such things as the required accuracy of the solutions, the accuracy of the available data and the expected ranges of the variables. This is the time to decide whether the problem is suitable for solution by analogue techniques. No hard-and-fast rules can be given, but as a general guide the following hints should be of value.

(a) Dynamic problems involving fast rates-of-change of variables, which must be solved in real time are best solved by analogue methods.

(b) Problems involving non-linearities, empirical relationships, etc., and requiring continuous solution are ideal for the analogue computer.

(c) Large sets of simultaneous algebraic equations are best solved by digital methods.

(d) Problems which involve large quantities of numerical data, such as statistical analysis are not suitable for solution by analogue methods.

(e) Problems which demand many logical decisions are best solved by digital methods.

(f) Problems requiring very high orders of solution accuracy must be solved by digital or combined analogue/digital methods.

These hints are not intended to be all-embracing but will at least enable the would-be user to avoid most of the major pitfalls into which the unwary can easily tumble.

Having decided that analogue methods are suitable, the next step is to re-arrange the equations into a form suitable for the preparation of a data flow diagram. This is a block diagram which shows the necessary mathematical operations and the order in which they must be performed. From this simple block diagram, a computer schematic diagram can be prepared showing the computing elements and the signal inter-connections.

The computer schematic can now be 'scaled' in accordance with the equations so that the variables are correctly represented and all the analogue voltages are within the working range of the computing elements. Scaling is a subject which often confounds the newcomer to analogue computing and will be dealt with in some detail. Finally, the necessary modules must be specified and a schedule prepared from which the computer can be assembled.

3.2 Problem Scaling

Scaling a problem is simply a matter of deciding the relationship between the computer voltages and the variables which they represent and choosing values of amplifier-gains and time-constants to give these relationships. In general, two types of scale-factors are involved, namely 'amplitude' and 'time'; in other words the dependent and independent-variable representation must be decided.

On-line process-control computers usually operate in 'real time', that is, the computer voltages vary at the same rate as the process variables. This means that a unity time scale-factor is required and problem scaling is reduced to assigning amplitude scale-factors to the voltages which represent the process variables.

Scaling is best understood by studying examples and a simple scaling example is given. The scaling of general-purpose computers is not considered in any detail but readers who wish to pursue the topic are referred to the bibliography.

3.2.1 Amplitude Scaling

The first requirement for scaling is that the maximum value that a variable can assume at any instant of time, is known or can be estimated. The maximum value of the variable can then be represented by the full-scale voltage of the computing elements, which in the case of ARCH is plus or minus 5 volts. For example, suppose that a gas pressure is to be represented and its maximum expected value is 450 p.s.i. – a convenient scale would be:

1 Volt=100 p.s.i., giving a full scale of 500 p.s.i.

In general terms, if x is the real variable and X is the analogue variable, then these are related by a scale-factor 'A' such that: $x=AX$.

Thus, for the example quoted, $A=100$ p.s.i. per Volt and the maximum value of voltage which X can attain is given by:

$$\frac{x_{max}}{A}=4.5\text{V}$$

To find the value of the real variable, the voltage representing it must be multiplied by the scale-factor. For the example given, if $X=2{\cdot}75$ volts then $x=275$ p.s.i. It should be noted that it is customary to denote real variable by lower-case letters and *computer* variables by capital letters.

Worked Example

To illustrate the principles, consider the simple equation:

$$y=kx+z \quad \text{where} \quad \begin{array}{l} x=\text{cu.ft. per hour} \\ z=\text{B.T.U. per hour} \\ k=\text{B.T.U. per cu.ft.} \\ x_{max}=10^4 \\ z_{max}=10^6 \\ k_{max}=10^2 \end{array}$$

The data flow diagram is shown in figure 13.

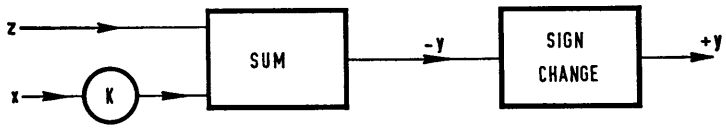


Figure 13 FLOW DIAGRAM FOR $y = kx + z$

Convenient scale-factors are chosen by dividing the maximum values of the variables by the full-scale computer voltage.

Thus, $$A_x=\frac{x_{max}}{5}=\frac{10^4}{5}=2\times10^3 \text{ cu.ft./hour per volt}$$

and $$A_z=\frac{z_{max}}{5}=\frac{10^6}{5}=2\times10^5 \text{ B.T.U./hour per volt}$$

The scale-factor for k is:

$$100\% \text{ potentiometer setting}=10^2 \text{ B.T.U./cu.ft.}$$

The maximum value of y is therefore given by:

$$y_{max}=(10^2\times10^4)+10^6=2\times10^6 \text{ B.T.U./hour}$$

A convenient scale-factor for y is therefore given by:

$$A_y=\frac{2\times10^6}{5}=4\times10^5 \text{ B.T.U./hour per volt.}$$

The computer equation is therefore:

$$Y=kX+Z \quad \text{where} \quad \begin{array}{l} Y=y/A_y \\ X=x/A_x \\ \text{and} \quad Z=z/A_z \end{array}$$

The computer schematic diagram can now be drawn as shown in figure 14. (It should be noted that it is customary to indicate the scale-factors by square brackets).

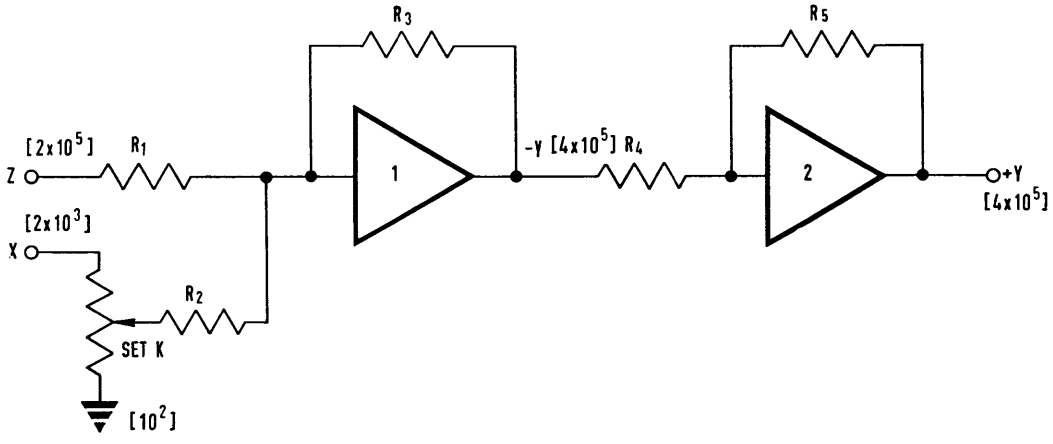


Figure 14 COMPUTER SCHEMATIC FOR $y = kx + z$

Values must be chosen for the input and feedback resistors which define the gains. This is achieved by taking the channels one at a time and dividing the input scale-factor by the output scale-factor.

$$\text{Thus for } Z, \frac{R_3}{R_1}=\frac{2.10^5}{4.10^5}=\frac{1}{2}$$

$$\text{and for } X, \frac{R_3}{R_2}=\frac{2.10^3.10^2}{4.10^5}=\frac{1}{2}$$

Note that for X, the input scale-factor is the product of the potentiometer scale-factor and that of the input signal. Application of the general rules given in section 2.9 yields suitable values for the resistors.

If a value of 1kΩ is chosen for the potentiometer and it is desired to set the coefficient k to 1% by means of a direct dial setting, then R_2 must not be less than 25kΩ. If a value of 50kΩ is chosen, this fixes the value of R_3 at 25kΩ and therefore R_1 must be given a value of 50kΩ. Amplifier No. 2 is a sign-changer and a suitable value for R_4 and R_5 is 10kΩ.

This scaling procedure ensures that the full range of the computing elements is used but is never exceeded at any point in the system. The results are interpreted by multiplying the computer voltages by the appropriate scale-factors. Thus if the output voltage of amplifier 2 is 3·65 volts then $y=3{\cdot}65 \,.\, 4 \,.\, 10^5=1{\cdot}46 \,.\, 10^6$ B.T.U. per hour. This particular scaling has one possible disadvantage, namely the fact that the two variables, y and z are both in units of B.T.U./hour but have different scale-factors. This can be avoided if it is accepted that the maximum value of Z is restricted to 2·5 volts, by assigning the same scale-factor to z as that of y and increasing the gain of the Z channel from $\frac{1}{2}$ to 1. However, in cases where this leads to very small voltage excursions the inconvenience of different scale-factors is usually the lesser evil.

In the case of integrators, the scaling is determined by the time-constants. It is convenient in the case of real-time applications to consider only the integrator-gain. The change of scale across an integrator can therefore be considered in the same way as a summer. Suppose that the scale at the input is N units, then the scale at the output is given by:

$$\frac{N}{G} \quad \text{where } G=\frac{1}{RC}$$

NOTE: If the dimensions of R and C are megohms and microfarads respectively, the dimensions of $\frac{N}{G}$ are units per second.

It is important that if the integrator has an initial condition this is scaled to give the correct output-scale-factor.

3.2.2 Time Scaling

Independent variables are represented in an analogue computer by time. In dynamic problems the independent variable is time, so time-scaling can be used to slow down rapidly-changing phenomena to bring them within the speed range of the computer. Conversely, very slow variables can be scaled so as to reduce the problem solution time.

The relationship is given by:

$T=Bt$ where T is the computer 'time',
t is the problem variable
and B is the scale-factor.

Time scaling is effected by writing into the original equations the term $\frac{T}{B}$ for the independent variable t. Any derivatives in the equations are also scaled and become:

$$\frac{d}{dt}=\frac{B.d}{dT} \quad , \quad \frac{d^2}{dt^2}=\frac{B^2.d^2}{dT^2} \quad , \quad etc.$$

Normally the only time-dependent computing elements are integrators so that time-scaling is achieved by choosing the time-constants and a problem solution can be easily speeded-up or slowed-down by changing every time-constant by the same proportion.

3.3 Simplification of Computer Schematics

In order to simplify the preparation of computer schematic diagrams, it is customary to use standard symbols to represent the various computing elements. The principal symbols are shown in appendix I. Effort is reduced since the actual input and feedback impedances are not drawn, the gains being indicated against each input.

3.4 Simultaneous Algebraic Equations

In many process-control applications simultaneous algebraic equations arise. These are usually restricted to perhaps two or three variables but are sometimes of a higher order. Although algebraic equations can be solved by analogue methods, certain difficulties arise with electronic computers. When a computer schematic is prepared it is found that only summing and scaling amplifiers are required and care must be exercised since such arrangements can be unstable under certain conditions.

A good rule-of-thumb which should be remembered is that an analogue computer loop with an even number of amplifiers is suspect since this represents a positive feedback condition. Such an arrangement should be carefully checked against the equations before it is accepted. The presence of one such loop does not always cause instability since often other stable loops may be present to prevent instability occurring. This is not the whole story, however; provided that the loop-gain is less than 1, a loop with an even number of amplifiers can still be stable. The loop-gain need only be slightly greater than 1 for the circuit to be unstable and for the voltages to quickly diverge until saturation occurs.

With an odd number of amplifiers in a loop it would appear that a stable system is obtained. This is not necessarily true, for the limited bandwidth of the amplifiers can cause phase-shifts which may be sufficient to give a positive feedback condition. The only 100% safe 'set-up' is therefore one in which the loop-gain is less than unity.

By way of illustration, consider the simple pair of equations:

$$x+3y=30 \ldots\ldots\ldots\ldots (1)$$
$$2x+\ y=20 \ldots\ldots\ldots\ldots (2)$$

Two approaches are possible, namely to solve (1) for x or to solve (2) for x.

Taking the first alternative, $x=30\ -3y$ and $y=20\ -2x$ and the computer schematic is as shown in figure 15a.

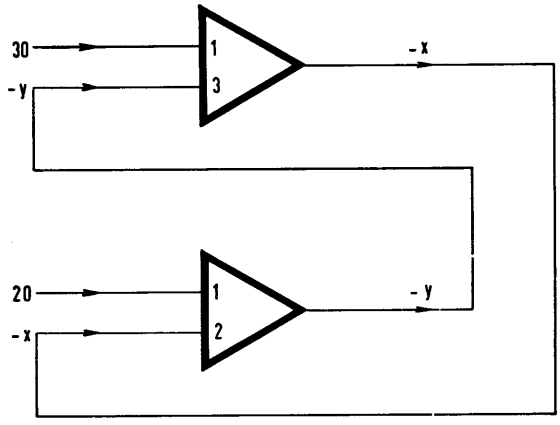


Figure 15a FIRST ALTERNATIVE

Checking the schematic shows that there is an even number of amplifiers in the loop and the loop-gain is greater than unity. This arrangement is therefore unsatisfactory.

Taking the second alternative, $x=\frac{1}{2}(20-y)$ and $y=\frac{1}{3}(30-x)$ and the computer schematic is shown in figure 15b.

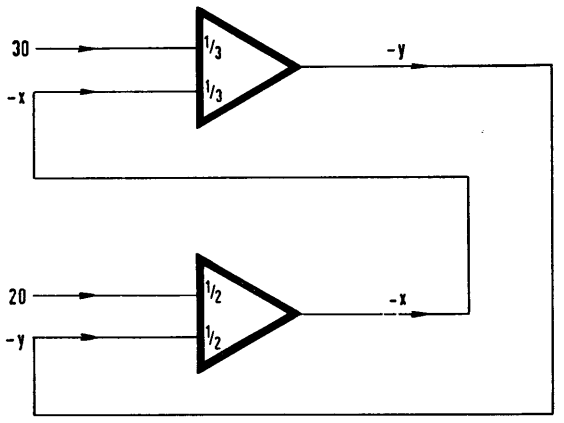


Figure 15b SECOND ALTERNATIVE

This arrangement, although still having an even number of amplifiers in the loop, should be stable since the loop-gain is less than unity.

Table 2 gives the maximum loop-gains for different numbers of amplifiers and any computer set-up which does not comply with these figures must be rejected as unsuitable.

Number of amplifiers in the loop	*Maximum loop-gain*
2	1
3	8
4	1
5	2·9
6	1
7	2·1

Table 2

Very often these conditions cannot be satisfied and some means of stabilising the loops must be found. The method usually adopted is replacement of the algebraic equations by a set of differential equations whose steady-state solutions satisfies the original equations.

Thus consider the algebraic equations:

$$a_{11}x_1+a_{12}x_2=b_1$$
$$a_{21}x_1+a_{22}x_2=b_2$$

These are replaced by the differential equations:

$$\frac{dx_1}{dt}+a_{11}x_1+a_{12}x_2=b_1$$
$$\frac{dx_2}{dt}+a_{21}x_1+a_{22}x_2=b_2$$

In the steady-state, the derivatives vanish and the original equations are satisfied. The differential equations can be solved by an analogue computer and, provided that a steady-state condition is reached, the solutions of the algebraic equations are obtained.

It would appear that the method is restricted to those sets of equations which yield steady-state solutions; techniques have been established which allow any set of equations to be transposed in such a way that a convergent solution can be obtained. Consideration of these techniques involves advanced matrix theory and is beyond the scope of this manual, for detailed information, the reader is referred to the bibliography.

3.5 Differential Equations

The basic approach to the solution of differential equations is best illustrated by some simple examples.

Example 1

In figure 16 a simple physical system is shown.

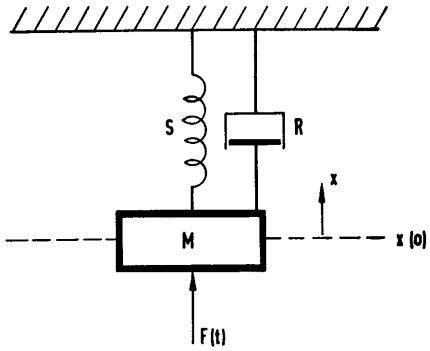


Figure 16 SIMPLE PHYSICAL SYSTEM

A mass M is suspended by a spring of stiffness S from a rigid platform and is also connected to a dashpot damper of resistance R. A force, $F(t)$ is allowed to act on the mass in the direction indicated.

The equation of motion of the mass due to the applied force is given by the differential equation:

$$M.\frac{d^2x}{dt^2}+R.\frac{dx}{dt}+Sx=F(t)$$

where M=mass in slugs (1 slug=32·2 lbs. mass),
R =pounds force per unit velocity of the dashpot,
S =pounds force per foot displacement of the spring,
and t =time in seconds.

The equation must be re-written in operational form and re-arranged to give the highest derivative only on the left-hand side, thus:

$$p^2x=-\frac{1}{M}\left(Rpx+Sx-F(t)\right)$$

The computer schematic diagram can now be drawn as shown in figure 17.

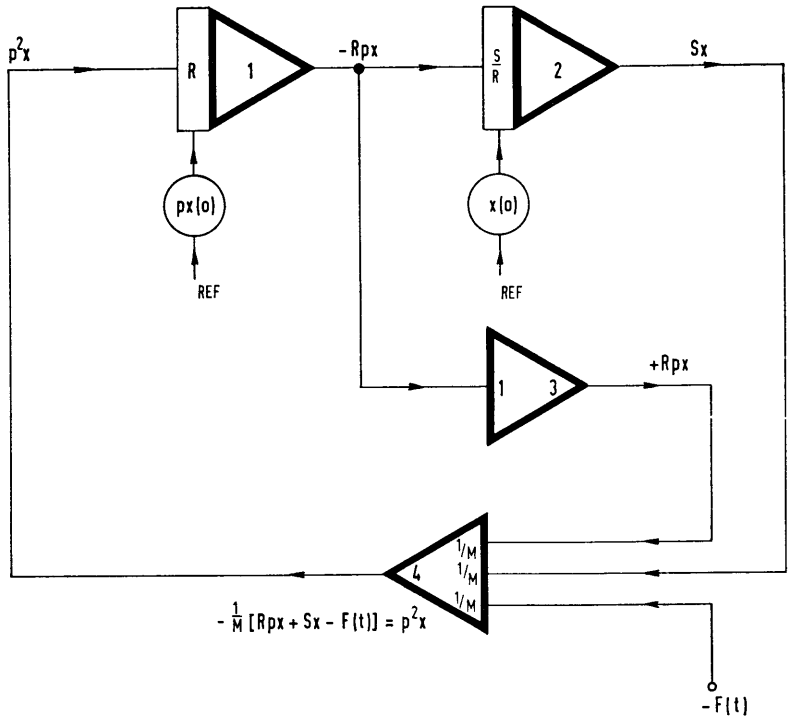


Figure 17 COMPUTER SCHEMATIC FOR THE SOLUTION OF A SIMPLE PHYSICAL SYSTEM

A voltage representing p^2x is assumed to exist and is integrated in amplifier 1 which has a gain of R; the output voltage thus represents $-Rpx$. This is integrated in amplifier 2 which has a gain of S/R and the output thus represents Sx. The output of amplifier 1 is inverted by amplifier 3 to provide the term, Rpx. The terms are summed in amplifier 4 each channel having a gain of $1/M$. The output voltage of amplifier 4 is thus representative of p^2x and can be connected to the input of amplifier 1 to provide the voltage which was assumed to exist. If a voltage representing the force $F(t)$ is applied to the appropriate input channel of amplifier 4, then the outputs of all the amplifiers will vary in accordance with the prescribed differential equation. The solution is obtained in graphical form by recording the output voltage variation of amplifier 2 against a time-base. Initial conditions for x and $\frac{dx}{dt}$ are applied to the integrators as described in section 2.4.

The response of the physical system to different applied forces may be readily obtained by choosing the form of the voltage which represents the force $F(t)$. The most commonly used 'forcing-functions' are the 'step' and sinusoidal functions. If potentiometers are used to adjust the gains, then whole 'families' of particular solutions may be quickly obtained for different coefficients of damping, spring stiffnesses, etc.

Example 2

Simultaneous differential equations may be easily solved by analogue methods. The approach adopted is illustrated by considering the following simple example:

$$\begin{aligned} & px-y=f(t) \\ \text{and} \quad & py+x=0 \end{aligned}$$

These equations are re-arranged to give:

$$\begin{aligned} & px=f(t)+y \\ \text{and} \quad & py=-x \end{aligned}$$

From these re-arranged equations, the computer schematic of figure 18 can be drawn.

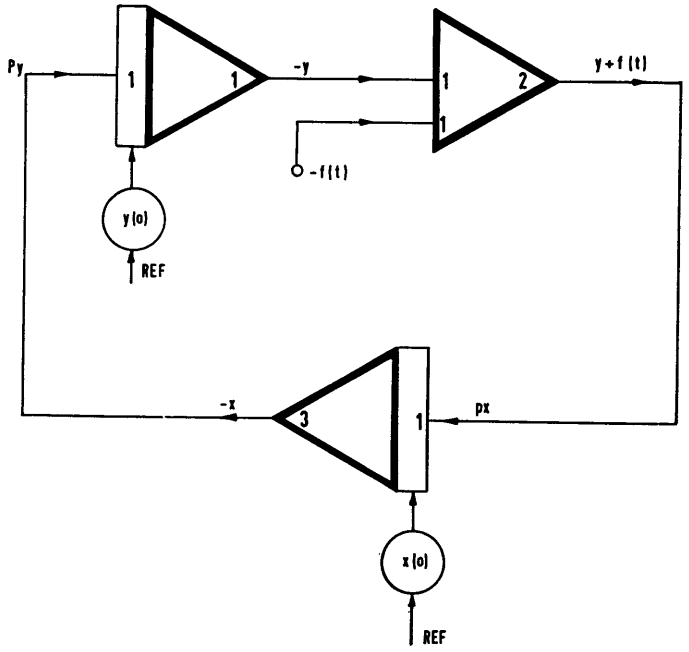


Figure 18 COMPUTER SCHEMATIC FOR SIMULTANEOUS DIFFERENTIAL EQUATIONS

This approach is used for any set of equations, no matter how complex; each equation is re-written to give expressions for the highest derivative of each of the dependent variables. Each highest derivative is then successively integrated until the variables and all the derivatives are obtained. The appropriate variables and derivatives are then scaled and summed to obtain voltages which represent the highest derivatives which can then be cross-connected to satisfy the equations. As with algebraic equations, any set-up involving loops with an even number of amplifiers is suspect.

4.0 PRACTICAL ASPECTS OF COMPUTER CONSTRUCTION

The general principles which have been given, together with the data sheets, should provide the basic facts necessary to undertake the construction of an on-line ARCH analogue system. There are, however, a number of practical points which are worthy of special mention.

4.1 Reference Voltages

In cases where really precise and stable reference voltages are required for the introduction of constants, integrator initial conditions, etc., an ARCH 5 volt reference module should be used. Often it is satisfactory to use the standard plus or minus 10 volt power rails for this purpose, the higher voltage being accommodated by suitable scaling in the functional units concerned.

4.2 Signal Earthing

For accurate computing it is essential that voltage differences due to earth currents are avoided. For this reason, a separate signal-earth terminal is provided on each module. In any assembly of modules, the signal-earth terminals should be linked from module to module by a conductor of substantial cross-section using the shortest possible route. The approximate electrical mid-point of this signal-earth busbar should be joined to the power-earth by a short link.

NOTE: the connection between signal and power earths must only be made at ONE point in the system.

In the case of large systems – say assemblies of thirty or more modules – it is recommended that a copper busbar with a cross-sectional area of about $\frac{1}{8}$ sq.inch is used for the signal-earth, with the feeds to the individual modules of 16 S.W.G. copper conductors.

4.3 Inter-connection of Modules

Short, direct, point-to-point wiring is recommended for the inter-connection of modules and 22 S.W.G. is usually satisfactory.

In applications where long leads are unavoidable, such as connections to plant transducers, screened twisted-pairs are recommended, the screens being connected to signal-earth. In cases where many inputs are involved, screened multi-core cable is usually satisfactory. The normal precautions taken with instrument-signal connections must be observed; thermocouples should be wired using the correct type of compensating cable and transducers such as resistance thermometers are best connected using the 'three-wire' system. The methods of converting low-level signals, current transmissions, etc., are dealt with in part 5 of this manual.

Bibliography

1. Electronic Analog Computers by G. A. Korn and T. M. Korn. *McGraw-Hill Book Co. Inc.*

2. Analog Computer Techniques by C. L. Johnson. *McGraw-Hill Book Co. Inc.*

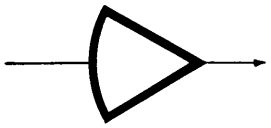
3. Analog Methods – Computation and Simulation by W. J. Karplus and W. J. Soraka. *McGraw-Hill Book Co. Inc.*

4. Analog Computation in Engineering Design by A. E. Rogers and T. W. Connolly. *McGraw-Hill Book Co. Inc.*

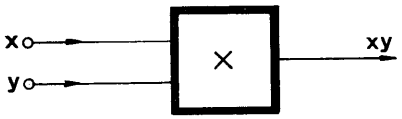
Part 3 Appendix 1

STANDARD SYMBOLS

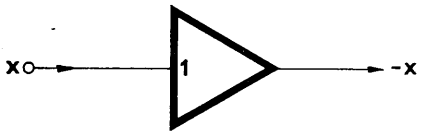
HIGH-GAIN AMPLIFIER



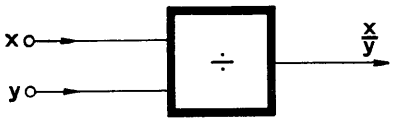
MULTIPLIER



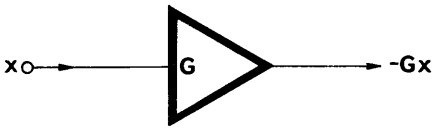
SUMMER USED AS SIGN CHANGER



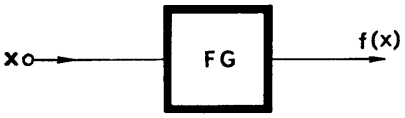
DIVIDER



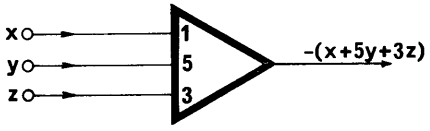
SUMMER USED FOR SCALING



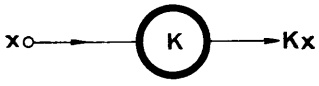
FUNCTION GENERATOR



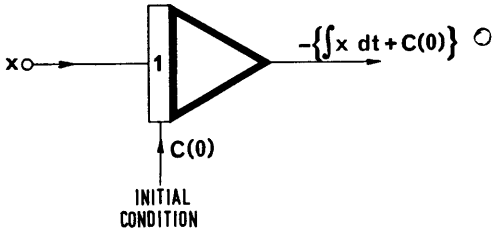
SUMMER (WITH SCALING)



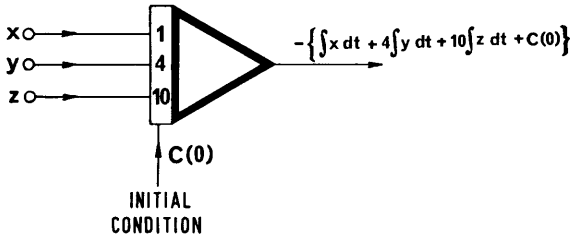
COEFFICIENT POTENTIOMETER



INTEGRATOR



INTEGRATOR WITH SUMMED SCALED INPUTS



ARCH MANUAL

4

part 4

DIGITAL TECHNIQUES

ARCH

reg'd trade mark

ARCH MANUAL PART 4

contents

S1104/3 11.63

1.0 INTRODUCTION

Electronic digital computers were introduced in the 1940s for solving mathematical problems at high speed. During the 1950s their use was extended to carrying out work in the business and office worlds. In the 1960s they are becoming increasingly used for the control of industrial processes.

It may not be immediately obvious why a machine developed for solving mathematical equations should have anything in common with one used for controlling an industrial plant. The answer to this is that both the solution of mathematical equations and the control of an industrial plant are carried out according to a set of rules.

In order to make these processes automatic therefore, a machine is required which is able to perform the necessary mathematical operations and also to take decisions and modify its own procedure in the light of the results obtained. The digital computer is such a machine.

2.0 TYPICAL DIGITAL COMPUTER

In order to understand what is involved in using a digital computer for industrial control, it is necessary to consider its internal working. Figure 1 shows the basic block diagram of a typical digital computer.

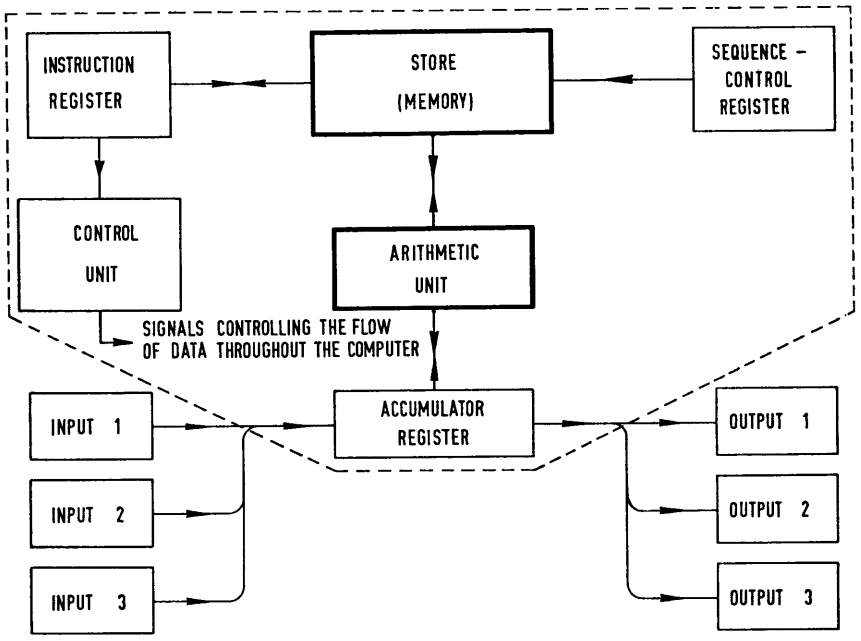


Figure 1 TYPICAL DIGITAL COMPUTER

As can be seen from this diagram, the digital computer consists of three main sections:

(1) Input This includes all the devices which enable measured values, calculation constants, etc. (known as input data) to be fed into the computer from external sources.

(2) Output This includes all devices which enable the results of the computer operations to be taken out and used for control.

(3) Central Processer This is the section shown inside the dotted line and is the section which processes the input data, i.e. carries out a number of specified operations on it in order to achieve the required output data.

These three sections vary considerably according to the application of the particular computer, hence it is necessary to go into each more fully.

2.1 Input

As explained in part 2 of this manual, numbers are represented in a digital computer by patterns of electrical pulses. Computer input devices therefore must be capable of converting the data required for input into these electrical pulses in the form that the computer can accept.

The type of input device used for any given application is dependent on the nature and amount of data required for input.

For instance, if a relatively small amount of data is required at any given time this can be put into a computer manually from control-panel switches or from the keys of a manual keyboard.

Where larger amounts of input data are required these methods may not be convenient or fast enough and punched paper-tape or punched-card inputs are used. These methods require pre-preparation of the input data into the form of a coded pattern of holes punched into paper-tape or cards. The input devices in these cases consist of paper-tape readers or card readers which convert the hole patterns into electrical pulses suitable for the computer to accept.

A form of input extensively required for process-control computers is that of measuring devices connected to the process. The majority of these measuring devices have electrical outputs with analogue representation. Inputs to the digital computer from these devices must therefore be via an analogue-to-digital converter (see part 5). In a few cases devices such as shaft encoders and digital transducers are used. These give electrical outputs in digital representation and can therefore be connected directly to the computer.

2.2 Output

Output devices may be regarded as the opposite to input devices; i.e. they convert the pattern of electrical pulses from the computer into various forms of presentation.

Again, the particular output device for a given application depends on the form of presentation and the amount of data required.

For instance, if a relatively small amount of output data is required and is used for operator guidance only, an illuminated display may be the best form.

If a record of the output data is required or if there is a relatively large volume of output data, a teleprinter or electric typewriter output is used. Such devices convert the coded electrical pulses into printed characters on a sheet of paper. Very large volumes of data may be printed out on high-speed line-at-a-time printers but the use of these is usually restricted to large business applications.

Punched-paper tape is a useful form of output since it is relatively fast and provides a digital record of the output data which can be converted into a printed record by feeding it into a suitable teleprinter. The device which converts the electrical pulses into holes in the paper-tape is known as a paper-tape punch.

In process-control applications, the computer output may be used directly to adjust the desired values of proportional controllers and to switch on/off controllers via suitable output devices.

2.3 Central Processer

As already mentioned, the central processer is the part of the computer in which the input data is operated on to achieve the required output. For most applications it is necessary to perform several operations on a number of inputs and to refer to the results of the previous calculations. There must therefore be something to hold numbers so that they can be used as required. In a digital computer, numbers are held in a *store* (*memory*).

2.3.1 Store

The store may be regarded as being an electronic filing system consisting of a large number of pigeon holes, each pigeon hole (*location*) having an identification number (*address*) and capable of storing one number.

2.3.2 Accumulator

This is a special store capable of holding one number only (i.e. a single 'pigeon-hole') and is the store to and from which all inputs and outputs are taken and to the content of which all arithmetic operations are performed.

2.3.3 Arithmetic Unit

The simple arithmetic operations are performed by taking one number out of the store and the other number out of the accumulator, feeding them both into an arithmetic unit which does the required calculation, i.e. add, subtract, multiply, etc., and then returning the answer to the accumulator.

2.3.4 Computer Instructions

In order to solve a complex problem it is necessary to perform a large number of simple operations in succession. That is to say the computer must be given a series of detailed *instructions* of what operations to perform and the addresses of the locations in the store in which the various numbers are to be found. Where there are a large number of instructions, it is necessary to store them so that they can be used as required. It is obviously convenient to store the instructions in the same way as the numbers are stored and to this end, each instruction is 'coded' into number form, and may be stored in any location in the store.

Most instructions concern operations involving the content of a particular location of the store, the arithmetic unit, and the accumulator. An instruction therefore consists of two parts: the operation (*function*) required, and the address of the location of the number concerned. The form of instruction of a typical machine is thus:

Instruction = Function (F) Address (A)

e.g.: 17 163 means function number 17 and is concerned with the number in location address 163. Each function is given a code number in the same way as each location has an address number. Instructions dealing with input and output are also coded in this way, the address in this case specifying the input/output device and not a storage location. The total range of instructions available in a particular computer is known as its *instruction code*. The particular sequence of instructions necessary to solve a problem is known as *the program* and the writing of these instructions as *programming*. This process is dealt with in detail in part 6.

With the program in the store, it is necessary to extract (*read*) each instruction in turn from the store and carry out (*obey*) the operation specified. The reading and obeying of the instructions is controlled by a control unit.

2.3.5 Control Unit

The first step in the execution of a program is to extract the first instruction from the store. Having extracted this instruction, it must be stored somewhere, and a special single-location store called the *instruction register* is provided for this purpose. The control unit then decodes the instruction into function and address and carries out the operation specified by controlling the flow of numbers to and from the accumulator, arithmetic unit and store. At the end of this operation the second instruction is read from the store into the instruction register (replacing the previous instruction), decoded and obeyed. The third instruction is then read and obeyed, and so on, until the calculation is finished. Thus it will be seen that the machine alternately performs two tasks, i.e. *reading* an instruction and *obeying* an instruction. Each of these tasks is called a *beat* and a machine operating in this way is called a *two-beat* machine.

During the read beat, the store address is specified by the number held in a *sequence-control-register* (S.C.R.) which is a single-location store which controls the sequence of instructions. At the beginning of the read beat, the next instruction is read from the store into the instruction register and one is then added to the address held in the S.C.R. Thus when the next read beat is reached, the new instruction is obtained from the location with an address one greater than the previous one. In this way, a simple sequence of instructions may be executed.

2.4 Computer Operation

To illustrate the operation of a simple series of instructions, suppose it is required to perform the calculation

$$x=a+b-c$$

where the number a is stored in location 100, the number b is in location 101, c is in location 102 and the answer x is required in location 103.

Given the function codes for getting numbers into the accumulator from the store and vice versa and for addition and subtraction, this problem needs four instructions stored in sequence in four locations (1, 2, 3 and 4 say).

The following table shows how the calculation proceeds.

Address of Instruction	Instruction (coded) Function	Address	Instruction (decoded)	Operation
1	1	100	*Replace contents of acc. by contents of store location 100*	$a \rightarrow acc$
2	3	101	*Add contents of acc. to contents of store location address 101 and place answer in acc.*	$a+b \rightarrow acc$
3	4	102	*Subtract the contents of store location address 102 from the contents of the acc. and place answer in acc.*	$(a+b)-c \rightarrow acc$
4	2	103	*Replace the contents of store location address 103 by the contents of the acc.*	$x=a+b-c \rightarrow store$ (N.B. x will also remain in the acc. at this point)

2.4.1 'Transfer' Instructions

One of the important features of a digital computer is the ability to break the normal sequence of instructions and 'jump' to a new sequence if certain conditions apply.

For instance, in the previous example, the number x may be either positive or negative depending on whether c is less than $(a+b)$ or greater than $(a+b)$ and subsequent operations involving x may require to be different depending on this sign. This can be achieved by a 'transfer' instruction which causes the normal sequence of instructions to continue if the accumulator content is positive but to be broken if the accumulator content is negative. Such an instruction may be in the form 9 50, meaning if the accumulator content is negative, take the next instruction from location 50 instead of the next location in sequence. This form of instruction simply causes the content of the S.C.R. to be replaced by the address part of the instruction if the accumulator sign is negative.

Suppose for example that a is the weight of a charge of substance A and b the weight of a charge of substance B. Suppose now that both of these charges are placed in a container whose weight cannot be measured directly but whose maximum safe capacity by weight is c. The sign of x in the calculation $x=a+b-c$ thus determines whether or not conditions are safe. If conditions are safe, $a+b$ is less than c and x is negative but if they are unsafe x is positive.

Thus if a 9 50 instruction is placed in location 5 of the above program the machine will continue in the normal sequence if the conditions are unsafe but 'jump' to location 50 if conditions are safe. In this way, the computer takes an elementary 'decision' to do one thing or another.

'Priority interrupt' is another form of transfer instruction, this time initiated from a source such as a pushbutton switch or contact external to the machine. This enables the computer operator or the operation of a contact to interrupt the machine's normal sequence of operations and cause it to jump to a predetermined set of instructions which are thus given priority over the normal sequence.

2.4.2 'Modify' Instructions

Another property of a digital computer which makes it so versatile is its ability to 'modify' its own instructions. Such modification can be done quite easily because instructions are coded into number form *and therefore may be operated upon by the arithmetic unit.* For example, the instruction 3 501, which means add the contents of location 501 to the accumulator, may be changed to 3 502 by adding 1, and into 3 503 by adding a further 1 and so on. In this way it is possible to total the contents of say 100 different locations, using the same basic instruction 100 times over, and modifying its address by 1 after each addition to the accumulator. In this way, only a few instructions are needed to obtain the complete answer, whereas if no modifications were possible over 100 instructions would be needed.

3.0 ARCH DIGITAL MACHINES

From the previous section, it can be seen that a variety of digital computers to suit different applications may be built by connecting the appropriate input and output 'modules' to the central processer. In ARCH this concept of building different machines by putting together suitable combinations of modules is carried one stage further *by constructing the central processer itself* from modules.

3.1 Arch Digital Modules

There are two basic types of ARCH digital module: *operational* and *control*. The operational modules are the ones which perform specific input, output, storage or arithmetic operations and the control modules control the flow of data (numbers and instructions) into and out of these operational modules.

3.1.1 Operational Modules

Transfer of data between modules is by means of a common system of *data busbars* to which all operational modules are connected (see figure 2). The passage of data from one module to another is in 'parallel' digital form. That is to say, there are as many wires in the busbars as there are binary digits (*bits*) in the number or instruction (*word*). All data pass from the operational modules to the *output busbars* and, via the busbar-drive module to the *input busbars* which feed the data inputs of all the operational modules. The busbar-drive module is merely a set of power amplifiers to enable large numbers of modules to be connected to the busbars without seriously loading the data outputs.

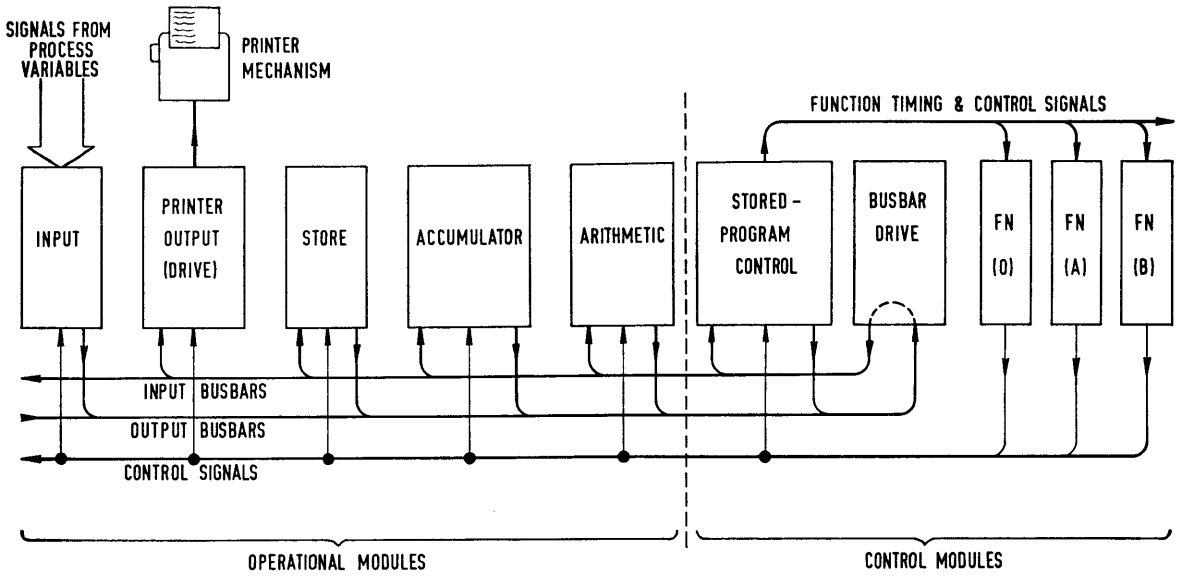


Figure 2 ARCH DIGITAL COMPUTER

The only other connections to the operational modules are the control signals from the control modules and, in the case of input/output modules, the connections to the input/output device.

A typical ARCH operational module is shown in figure 3.

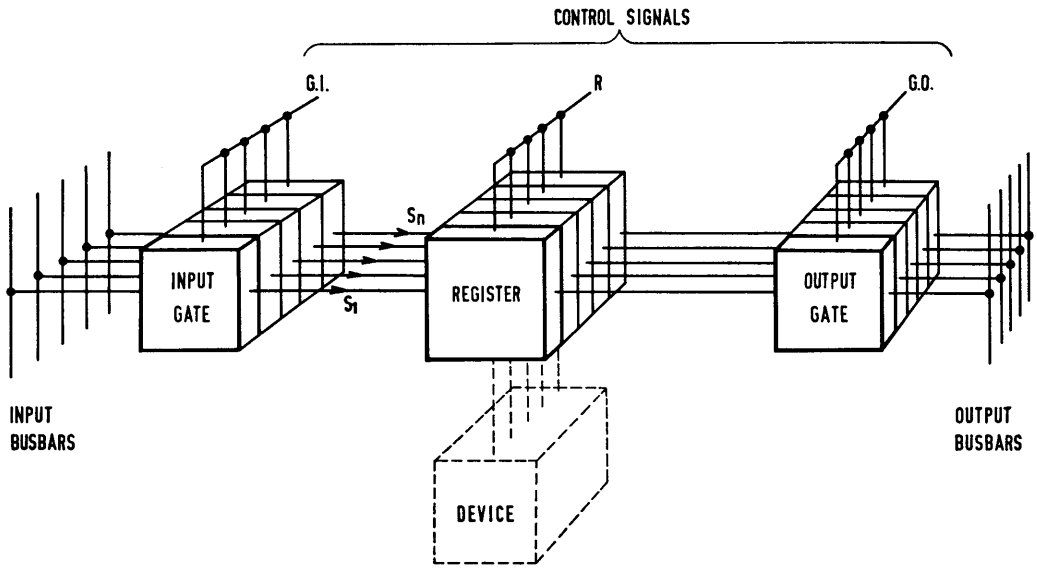


Figure 3 TYPICAL OPERATIONAL MODULE

The parallel nature of the busbars and of the module itself, is clearly shown. The register consists of *N* single-bit memories. Each memory is either *on* (one) or *off* (zero), depending on whether or not it has been set by the individual pulses S_1, S_2, $S_3 \ldots S_n$.

In order to get the number on the input busbars into the module register, the following sequence of events takes place.

(1) All the one-bit memories are *reset* to the zero (off) state by the common waveform **R.** This is a short pulse which lasts just long enough to change the state of the memories.

(2) The pattern of 1's and 0's on the input busbars is then allowed through the input 'gates' for a short period to produce the individual 'setting' pulses S_1, S_2, $S_3 \ldots S_n$. This is achieved by 'opening' the input gates by the common control waveform **G.I.** The memories forming the register are then set in the same pattern of ones and zeros as is on the busbars during the time the input gates are open.

When the gate is closed, the pattern of 1's and 0's on the busbars can be changed without affecting the setting of the register.

To obtain the output data from the module, the output gates are opened for a short time by the common control waveform **G.O.** The pattern stored in the register appears on the output busbars and on the input busbars via the amplifiers in the busbar-drive module. It should be noted that the busbar-drive amplifiers reverse the sign of (*invert*) the pulses, so that a 'one', which is a negative-going pulse on the output busbars (B_{OUT}), becomes a positive-going pulse on the input busbars (B_{IN}).

The 'device' shown in figure 3 is representative of the 'operational' part of the module which may be an input/output device, store, or arithmetic unit.

Suppose that a number is to be transferred from the accumulator to an output module, e.g. a printer. The register in the output printer drive module is first reset to zero by reset signal R.-OP.R. The output gates of the accumulator and the input gates of the printer drive are then opened simultaneously by G.O.-Acc. and G.I.-OP.R respectively. The pattern of ones and zeros corresponding to the content of the accumulator appears on B_{OUT} and, inverted, on B_{IN}, passes through the printer-drive input gates and sets the printer-drive register. There is a slight delay between the opening of the output gates of the accumulator and the appearance of the number on the busbars. This is due to the small delays which are inherent

in the transistor circuits used. There is a further delay as the pulses pass through the printer-drive input gates to produce the setting waveforms 'S'. The delays differ slightly from channel to channel (digit to digit) but the relative lengths and overlaps of the control waveforms are chosen to ensure an adequate 'safety-margin' of operation.

Once the required number has been transferred to the printer-drive register, the busbars may be used for other data transfers between modules while the printer is going through the mechanical operations necessary to print out the number (character) held in the printer-drive register. During the time taken to print one character (usually about 100ms), several hundred transfers of data between other modules can take place. The printer-drive module, however, must not be disturbed during this period and it is therefore said to be *busy*. The module generates a *busy* signal, for the duration of the printing cycle, which is used to prevent the computer program from giving another print instruction to the same printer too soon (see part 6).

All ARCH operational modules, whether they be stores, arithmetic, input or output, communicate with one another via the busbars in a manner similar to that described above, the flow of data being controlled by the control signals generated by the control modules. The data sheets in part 8 of this manual give the exact details of each of these modules.

3.1.2 Control Modules

As in the typical digital computer described in section 2, the operation of an ARCH digital machine is controlled by a program of instructions, coded as numbers, and stored in a store. These instructions are alternately read out from the store into an instruction register and decoded and obeyed. A sequence-control register (S.C.R.) determines the address of the next instruction to be read out of the store.

In ARCH the instruction register, the S.C.R., and the basic circuits for controlling the two beats (read and obey) and the timing of the functions, are contained in a stored-program control module.

It can be seen from figure 2, that a number of function modules are connected to the stored-program control via function-timing and control lines. The function control lines consist of a set of 'function busbars' which, during the obey beat, feed the pattern of digits corresponding to the function part of the current instruction in the instruction register to the function modules. All the function modules receive the signals on these busbars and decode them, but only the one having the code corresponding to that of the current function is selected and energised. The function-timing lines consist of a set of timing busbars which are used to transmit to the function modules the output of a timing unit in the stored-program control module. Under the control of these timing signals, the selected function module feeds the appropriate operational modules with the various control signals, e.g. R, G.I., G.O., etc.

Suppose, for example, that the instruction register contains the instruction 6 3 (output the contents of the accumulator to an output device number three). At the beginning of the obey beat, the function module dealing with function 6 is selected and, under the control of the timing unit, generates the sequence of control signals necessary to achieve this function.

The transfer of data between the accumulator and the output device (printer) is carried out by the control signals R.—OP.R., G.O.—Acc. and G.I.—OP.R as previously described. G.I.—OP.R, which is generated by the function module concerned with function 6 is connected to all the output devices, hence it is necessary to select the particular device specified by the address part of the instruction before the transfer can take place. This selection is carried out prior to the transfer by gating the address part of the instruction register content onto the output busbars and using this to set a one-bit memory in the specified output device and reset its register. This one-bit memory 'primes' the specified input gate so that when G.I.—OP.R and G.O.—Acc. are generated, the content of the accumulator is transferred only to the register in the device specified by the address.

The control signals generated by the function 6 module are shown in figure 4.

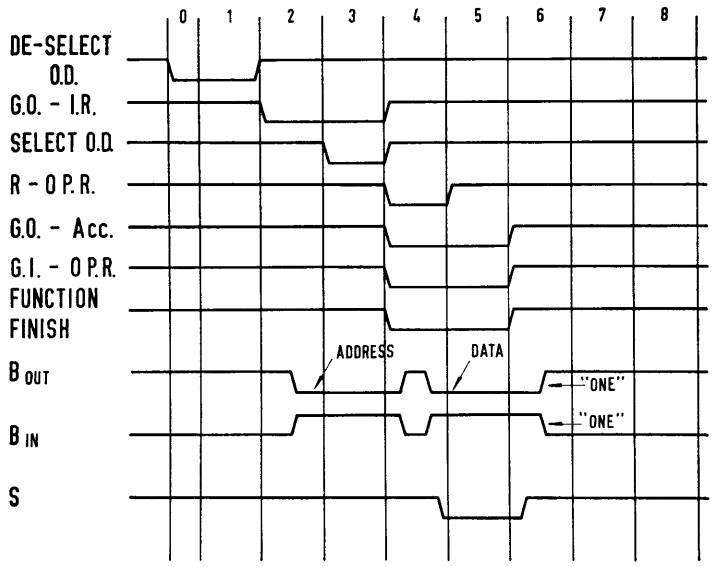


Figure 4 TIMING DIAGRAM OF CONTROL SIGNALS

In timing periods 0 and 1, DE-SELECT O.D. resets the one-bit memory in the previously-selected output device. In periods 2 and 3 G.O.—I.R. gates the address part of the instruction register content onto the output busbars and in period 3, SELECT O.D. causes the one-bit memory in the specified output device to be set. In periods 4 and 5, R.—OP.R., G.O.—Acc. and G.I.—OP.R. effect the transfer from the accumulator to the output register. FUNCTION FINISH, which is also generated during periods 4 and 5 is fed back to the stored-program control to change the machine to the 'read' beat at the end of period 5.

Instructions relating to input devices are carried out in a similar manner to that described above with control signals SELECT I.D., G.O.—IP.R., etc.

Transfers from the store to the accumulator and vice versa also involve gating the instruction address (in this case, specifying the store location) onto the output busbars prior to the actual transfer. In these cases the address is gated into an address register in the store module in order to select the specified location.

Arithmetic instructions usually involve the store, arithmetic and accumulator modules and the arithmetic function modules generate the necessary control signals to gate numbers out of the store and the accumulator, gate them into the arithmetic module and gate the result back into the accumulator.

In each case, the function timing is controlled from the timing busbars and a FUNCTION FINISH signal is sent back to the program control to cause a change of beat when the function is complete.

During the read beat, the function busbars are held at zero, and function module number zero is therefore selected. This module causes the content of the S.C.R. to be gated into the store address register (via the busbars), the content of the location specified to be read out from the store and this new instruction to be gated into the instruction register (via the busbars).

The examples given above are typical of most functions, there are however, a number of important variations which will now be described.

Transfer Control (Function 8)

When function 8 is obeyed the content of the S.C.R. itself is replaced by the address part of the instruction, and the old content is sent to a specified location in the store. Thus a new 'branch' in the program is started, and the address for continuing the old branch is remembered.

Conditional Transfer (Function 9)

In function 9 the content of the S.C.R. is replaced by the address part of the instruction if the sign of the accumulator is negative, otherwise the next instruction is taken from the next location in sequence.

Modify (Function 5)

In function 5 the content of the specified storage location is sent to a special register called the 'modify register' in the arithmetic module, and a modify 'memory' is set. The setting of this modify memory causes the content of the modify register to be added to the next instruction as it is read out of the store and passed to the instruction register.

3.2 Practical Aspects of Construction

The selection of digital modules chosen for a particular application depends on the nature of the control problem. It is only necessary to assemble the particular operational modules which provide the required functions and the particular control modules to control the way in which these functions are performed.

Each module is a complete unit which requires no interconnection apart from that necessary to connect it into a system. The connections to each module consist of the data inputs and outputs, the control signals and the power supplies. All these connections are taken to a set of pins at the rear of the module. Each of these pins is given a number taken from a 'master list' in which a number is allocated to every possible ARCH signal. Interconnection of modules is therefore very simply achieved by connecting together all pins of the same number on all the modules. For instance, the data inputs are numbered 1 to 18, hence the input 'busbars' are formed by connecting together pins with these numbers on all the modules.

Interconnection is best carried out with 26 S.W.G. wire for the signal connections and 16 S.W.G. or 23/·0076 for the power-supply connections. Wiring should be as short as is practically possible; this can be assisted by grouping the modules so that for instance the function modules are positioned next to the stored-program control module, etc.

ARCH MANUAL

5

part 5

CONVERSION TECHNIQUES

ARCH

reg'd trade mark

ARCH MANAUL PART 5

contents

S1105/3 11.63

1.0 INTRODUCTION

ARCH computing modules are divided into two groups, namely, analogue and digital. There are many applications of ARCH which warrant the use of both analogue and digital techniques and it is apparent that some means is needed to convey information from one type of module to the other. The inputs to ARCH are usually obtained from transducers and the outputs from ARCH are taken to process-controllers; here again a 'language' difficulty exists. A means must be provided to read the input signals accurately and to convert the output signals into a suitable form for the controllers. The converter modules described in this part provide this means and also permit the interchange of data between the analogue and digital modules.

2.0 CONVERSION BETWEEN ANALOGUE AND DIGITAL REPRESENTATION

In this section the methods of performing conversions between analogue and digital forms of representation are considered.

2.1 Analogue-to-Digital Converter (A.D.C.)

The ARCH analogue-to-digital converter modules are used for the input of analogue signals to digital modules.

The conversion is performed by electronic circuits whose operation is based upon a potentiometric-comparison principle. The analogue input signal is measured by successively comparing its magnitude with accurate reference potentials (or currents) in a number of discrete steps, each step corresponding to a digit of a fixed significance. For example, in the converter with pure binary outputs, the reference potentials (or currents) are arranged in a binary progression, each successive reference being one half of its predecessor, i.e. one half full-scale, one-quarter full-scale, etc.

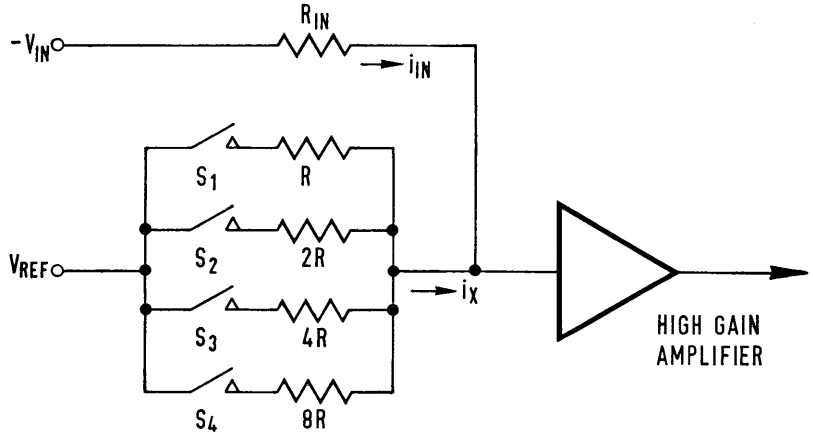


Figure 1 FOUR-BIT BINARY COMPARATOR

The basic principle is illustrated in figure 1, which shows a simple four-bit binary comparator. The resistor values are chosen in binary progression so that each resistor, when switched into the circuit, causes a current to flow which is exactly one half of the current due to the preceding resistor. The input signal $-V_{IN}$ is applied to a standard resistor R_{IN} and produces a negative current i_{IN} which is compared with the sum of the positive reference currents by means of a very-high-gain amplifier. The sign of the amplifier output voltage is determined by the direction of flow of the net input current, and thus indicates whether the sum of the reference currents i_x is greater or less than i_{IN}. The switches are closed one at a time, in sequence, in an attempt to satisfy the null-condition, $i_x+i_{IN}=0$.

The sequence is as follows:

(1) S_1 is closed and left closed if i_x is less than i_{IN}; otherwise it is opened again.
(2) S_2 is closed and left closed if i_x is less than i_{IN}; otherwise it is opened again.
(3) This procedure is repeated for the remaining switches until the null-condition is as closely satisfied as possible. In general, a perfect null can only be obtained with a large number of reference currents.
(4) The final state of the switches gives a binary number which is proportional to the analogue input signal, each closed switch representing a '1' and each open switch a '0'. Switch S_1 is the most-significant digit (M.S.D.) and switch S_4 is the least-significant digit (L.S.D.).

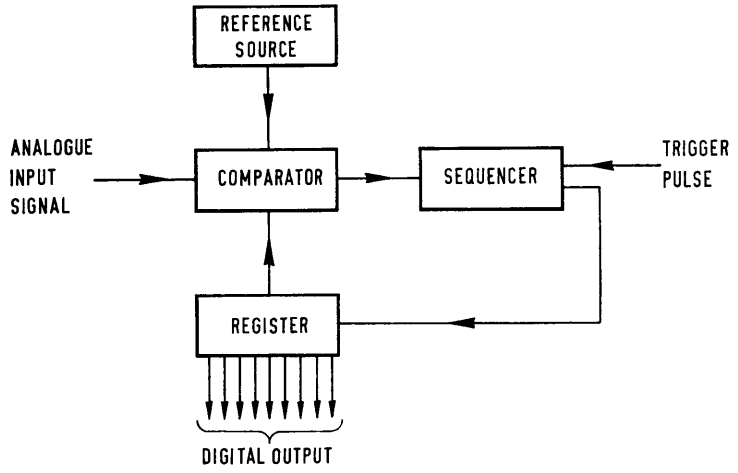


Figure 2 BLOCK DIAGRAM OF BASIC ARCH A.D.C.

A block diagram of a basic ARCH A.D.C. is given in figure 2. The 'switches' in the comparator are transistorised switching circuits controlled by a register which consists of a number of bistables (single-bit memories), one per digit. The bistables are controlled by a sequencer. A 'trigger' pulse initiates the operation which starts by resetting the register to remove the previous contents. The input signal is then examined to determine its polarity and a 'sign' digit is entered into the register. The M.S.D. switch is then closed and the comparator output instructs the sequencer whether or not this digit is required. If it is not required the M.S.D. bistable is reset; otherwise it is left set. This procedure is repeated in sequence until the L.S.D. has been determined. The register then contains a digital number proportional to the magnitude of the analogue input signal.

Two types of A.D.C. are available, namely, a binary converter and a binary-coded decimal (B.C.D.) converter. Both types accept input signals in the range 0 to 5 volts d.c. and this input can be positive or negative in each case. The binary converter has a 12 bit (plus sign-digit) digital output which gives a full-scale of 4095 steps with a resolution of one step. The B.C.D. converter has three full decades of four binary bits each and a least-significant decade

of two binary bits. This gives a full-scale of 9997 (the L.S.D. can only be 7, 5, 3 or 0).

The conversion accuracy of potentiometric A.D.C.s depends upon the ratios of the resistors which define the reference currents. The precision, or repeatability, depends upon the stability of the resistors and the reference voltage. Accurate performance is maintained in the ARCH A.D.C. over a wide temperature range (0 to 45 degrees Centigrade) by the use of precision wire-wound resistors with very low temperature coefficients, the selection tolerances being chosen to give the necessary ratio accuracies. The resistors are switched by means of silicon diodes which are temperature-controlled thus avoiding errors due to variations of the forward resistances with temperature change. The reference voltage is derived from a temperature-controlled Zener diode in a precision voltage-stabiliser, which employs ARCH d.c. operational amplifiers in a special feedback circuit and provides a very high degree of voltage stability.

2.2 Digital-to-Analogue Converter (D.A.C.)

The D.A.C. is constructed in a similar manner to the A.D.C. as shown in figure 1.

Similar resistor values are used and these are switched to a common line by means of the switches S_1 to S_{12} which in turn are controlled by one-bit memories.

The required digital pattern is set into the one-bit memories and the appropriate switches S_1 to S_{12} are closed *simultaneously*. The currents so produced are summed using an operational d.c. amplifier to give an output voltage proportional to the digital input.

The ARCH range of D.A.C.s provides for inputs in binary or binary-coded decimal form and positive or negative numbers.

3.0 INPUT SIGNALS TO ARCH

The input signals to ARCH are derived in a variety of ways and take many different forms. Before they can be accepted by the computing modules, two requirements must be fulfilled:

(a) The input signals must be of the correct form i.e., d.c. voltage for the analogue modules and binary or binary-coded decimal for the digital modules.

(b) The value of the input signals must be correct i.e. $-10V=0/0V=1$ for the digital modules and up to 5V full scale for the analogue modules.

The majority of input signals are obtained from primary measurement devices and transducers which are located in different parts of the process plant. Transducers sometimes give high-level output signals, but more generally they are low-level devices and the signals require amplification and often conversion from one form of representation to another. In practice, the input signal which can be used directly is the exception rather than the rule.

3.1 Analogue Inputs

Unfortunately, there is no universally accepted standard for analogue signals; those in common use are listed below.

(a) Pneumatic signals, generally 3 to 15 p.s.i.

(b) Direct voltage or current, e.g. 0 to 10 millivolts, 0 to 10 milliamps, 4 to 20 milliamps, etc.

(c) Alternating voltage, e.g. 0 to 0·5 volts r.m.s. 50c/s.

To satisfy the input requirements of ARCH modules, pneumatic signals are converted into electrical form by means of suitable pressure transducers. Pneumatic signals are therefore not discussed separately.

Electrical signals, in particular low-level d.c. signals, are subject to interference, such as 'noise' pick-up which can introduce serious errors in the measurement of the signal. Such interference influences the methods of amplification of high-level and low-level signals and is therefore discussed in more detail below.

3.1.1 Interference to Analogue Input Signals

In the majority of processes, the primary measurements are made at points which are remote from the computing equipment. The signals must therefore be transmitted over considerable distances and the connecting cables almost always pick up 'noise' from adjacent power cables or similar interference generators. Such noise can be very troublesome, it giving rise to spurious input signals and, in extreme cases, causing saturation of amplifiers or other input devices. Special techniques must therefore be adopted to minimise pick-up, and input modules must be provided to detect accurately very small signal voltages which are 'masked' by noise.

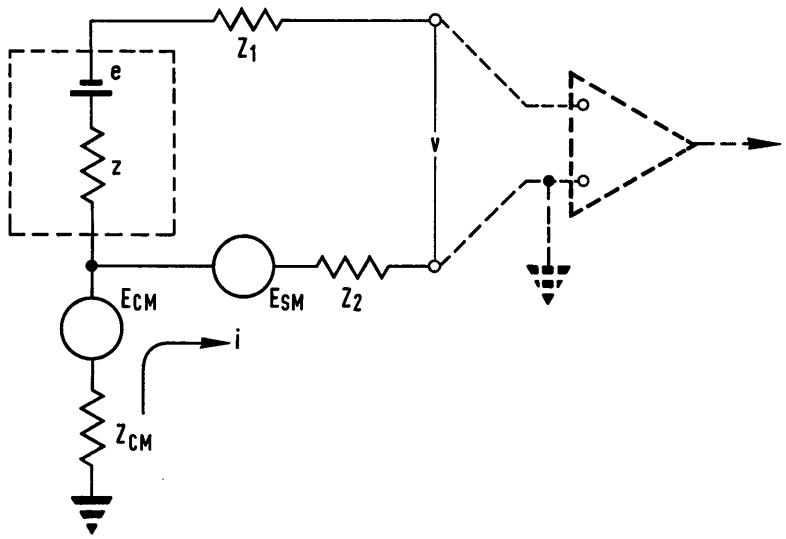


Figure 3 INTERFERENCE SIGNALS

The problem is illustrated in figure 3, which shows a d.c. transducer, represented by a battery having an e.m.f. of *e* volts and an internal impedance of z ohms. In practice, transducers are seldom completely isolated from 'earthed' masses which are in fact, not at true earth potential. Potentials from these sources, together with noise pick-up, cause unwanted voltages to appear at both transducer terminals. These voltages are represented in the figure by a voltage generator, E_{CM} and their impedances by an impedance, Z_{CM} to true earth. E_{CM} is called the 'common-mode' voltage and may be d.c., a.c. or in some cases, d.c. with a superimposed a.c. component. The a.c. component is usually at power frequency or its harmonics and can have an amplitude of many volts, with a source impedance ranging from a few ohms to a few hundred megohms.

Pick-up also produces a noise signal, E_{SM}, which is added algebraically to the true signal such that the voltage which appears between the lines at the far end of the cable is given by $v=e+E_{SM}$. The E_{SM} voltage is known as the 'series-mode' noise voltage.

If an amplifier having one earthed input terminal is connected as indicated by the dotted lines in figure 3, the common-mode voltage E_{CM} produces a current through the line impedance Z_2 in series with Z_{CM}. This current in the so-called 'earth-loop' produces a voltage drop across Z_2 which is in series with the transducer output signal. This amplifier connection thus causes

the common-mode voltage to be converted into a series-mode signal at the amplifier input. For example, if E_{CM} is 100 volts, Z_{CM} is 10 megohms and Z_2 is 100 ohms, then the voltage drop in the line is approximately one millivolt. Typically, a transducer such as a thermocouple has a full-scale signal of 10 millivolts, so that in this case the noise is 10% of the maximum signal – a very serious error. If, the transducer has a high-level output of say, 5 volts, the error for the same example is reduced to a mere 0·02%; it is thus obvious that this effect is more serious on low-level signals.

In order to overcome the earth-loop problem it is necessary to use a special type of amplifier which has a completely 'floating' input stage. Even in these circumstances the common-mode voltage, E_{CM}, can cause a series-mode voltage to be generated if the distributed line-impedances are unbalanced. The lines have resistance, inductance and capacitance to earth distributed throughout the length of the line; these can be approximated by 'lumped'

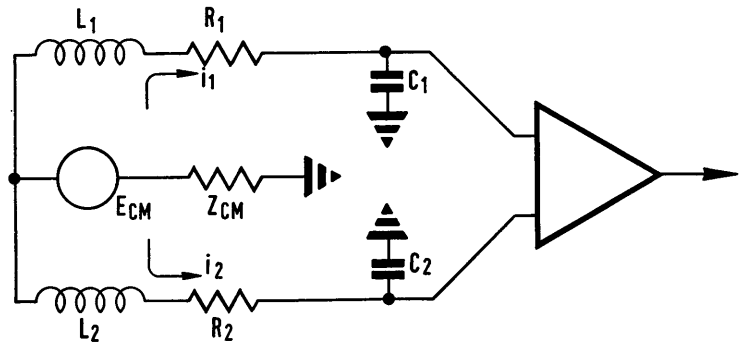


Figure 4 COMMON-MODE INTERFERENCE

constants as shown in figure 4. The common-mode voltage, E_{CM}, causes current to flow and unless $L_1=L_2$, $R_1=R_2$ and $C_1=C_2$, a voltage difference exists between the amplifier input terminals and this voltage is amplified as if it were a genuine input signal. This condition of unbalance is aggravated in the case of thermocouples because a special compensating cable is often used to connect the couples to a common cold-junction at the amplifier input terminals. Such cables may be resistively unbalanced by as much as 3:1. This means that ideally the lines should be balanced by means of padding resistors and capacitors, inserted in the lines immediately before the amplifier. Some series-mode noise will still be present, however, due to pick-up and imperfect balancing, and this noise *must be attenuated by means of a filter placed before the amplifier input.*

General precautions should be observed with input signals in order to minimise noise pick-up - prevention being better than cure. These are summarised below.

(a) If possible, signals should be produced at, or converted to, high-level *at the point of origin.*

(b) Balanced, 'twisted-pair' connecting cables should be used to minimise pick-up due to electro-magnetic fields.

(c) The conductors should be enclosed in a braided copper sheath, or screen, to minimise pick-up due to electrostatic fields.

(d) With low-level signals, care must be taken to avoid spurious emf.s. due to thermo-electric effects at terminals, i.e. junctions of dissimilar metals must be avoided or arranged to cancel each other.

(e) Connecting cables should be routed to avoid 'parallel runs' close to power cables.

(f) Earth-loops must be avoided.

Greater care is required in preventing noise pick-up when the genuine input signal is varying rapidly (or when a fast scanning rate is used, in the case of a common amplifier being shared among a number of inputs) than is the case for slowly varying inputs. This is due to the fact that the noise filter must have a wider pass-band (or faster transient response) and hence has less attenuation at noise frequencies. Particular care is required when scanning rates of five points per second or above are reached.

3.1.2 ARCH Low-Level-Signal Amplifier

Low-level signals are amplified to the standard ARCH input level (5 volts full-scale) by means of a special signal amplifier which satisfies the following requirements:

(a) A high rejection to common-mode noise.

(b) A low output-drift voltage.

(c) A range of accurate forward-gains.

Requirement (a) is met by employing a shielded isolating transformer in the input stage of the amplifier and requirement (b) by the use of 'chopper' techniques. The forward-gain accuracy is obtained by means of extremely high gain inside the amplifier and the application of feedback.

An isolating transformer is shown in figure 5.

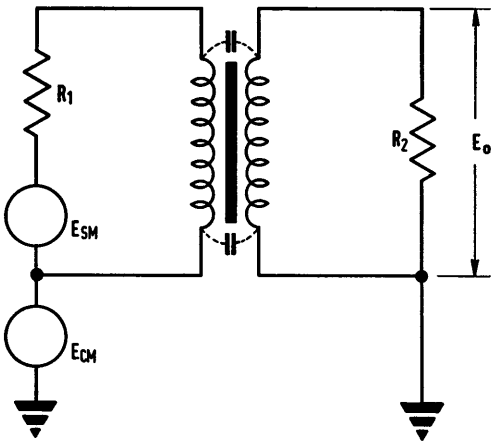


Figure 5 ISOLATING TRANSFORMER

E_{SM} is the signal (transducer output plus series-mode noise) and E_{CM} is the common-mode voltage. If the transformer is perfect and has a 1:1 transformation ratio the voltage developed across R_2 is given by:

$$E_0 = \frac{R_2}{R_1 + R_2} \cdot E_{SM}$$

However, owing to leakage capacitance and other effects, there will be an additional secondary voltage due to the common-mode voltage E_{CM}. This can be largely overcome by providing a shield as shown in figure 6. The transformer must be carefully constructed so as

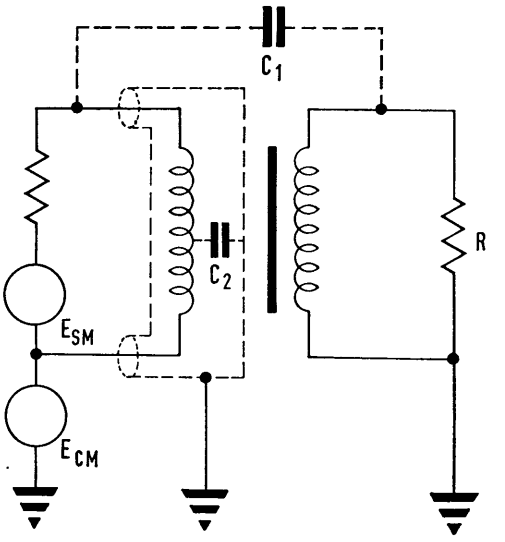


Figure 6 SHIELDED WINDINGS

to minimise the leakage capacity, C_1 which is the result of imperfect shielding.

The layout of the ARCH signal amplifier is shown in block schematic form in figure 7.

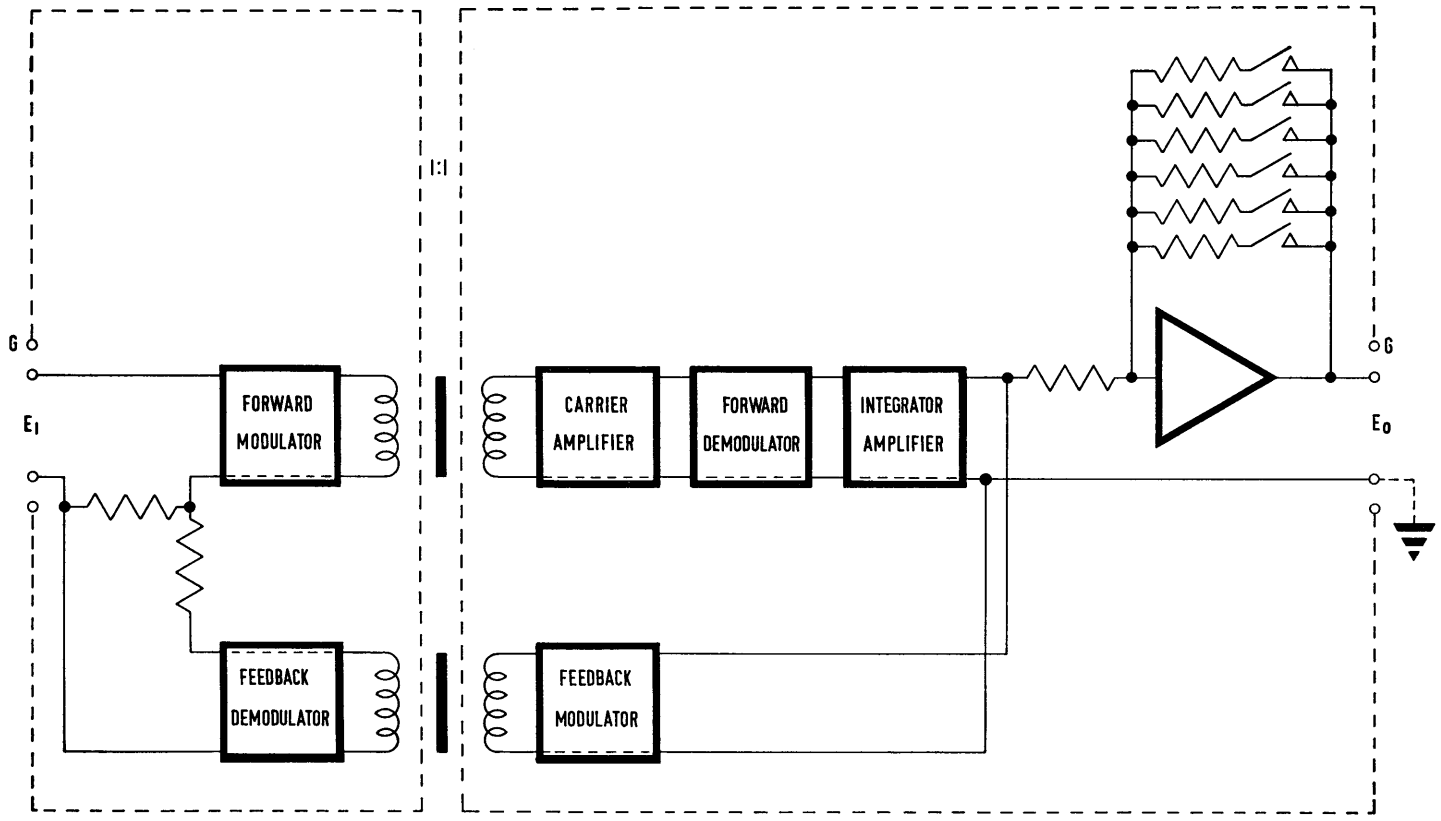


Figure 7 BLOCK SCHEMATIC OF ARCH SIGNAL AMPLIFIER

It is, in fact, a feedback control system. The input signal is summed with a feedback signal and the resultant 'error' signal is modulated by a 500c/s chopper. The modulated error signal is applied to the primary winding of the 1:1 input transformer, which removes the d.c. level and rejects the common-mode voltage. This square-wave signal, which is balanced with respect to the signal-earth, is amplified by an a.c. coupled drift-free amplifier (carrier amplifier). The output of the carrier amplifier is demodulated and then smoothed in an integrating amplifier to provide a high-level voltage output whose magnitude and phase are precisely related to the error signal. This integrator output voltage is fed-back via a 1:1 isolating circuit which consists of a chopper-modulator, an isolating transformer and a demodulator. The system acts to reduce the error signal to zero and when this condition is satisfied the output voltage of the integrator is exactly proportional to the input signal.

The output of the integrator is taken to an operational amplifier, which is outside the feedback-loop, for further amplification. The gain of this final output amplifier may be selected remotely. This allows 'scaling' to suit different transducers, in applications where the amplifier and A.D.C. are 'time-shared' amongst a number of input signals.

3.1.3 High-Level D.C. Voltage Input Signals

Transducer output signals which are at high-level, with full-scale values in the range 1 to 5 volts say, do not require a high degree of amplification and are easier to handle than low-level signals.

Signals which are given with respect to earth can be accepted directly by the ARCH analogue modules, but a 1:1 'buffer' amplifier may be required to avoid source-loading. Such a buffer is usually required with analogue multipliers and dividers and with the A.D.C.

Where the transducer output is 'floating' and cannot be directly connected to earth, an arrangement using operational amplifiers should be used (see figure 8).

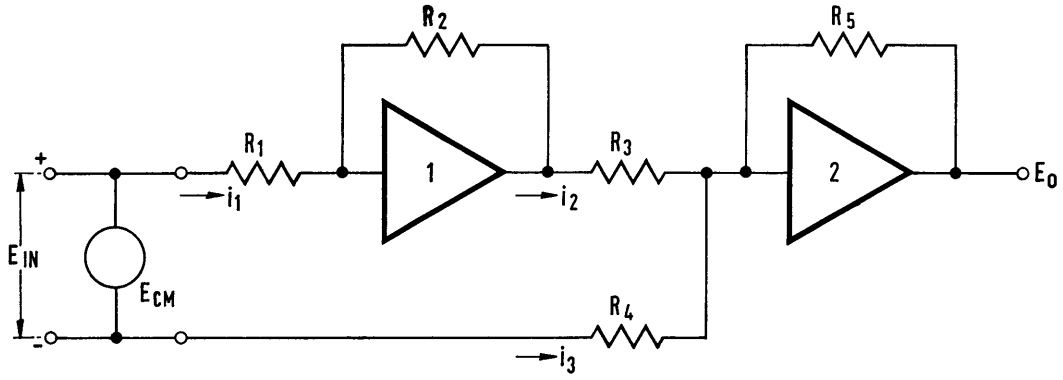


Figure 8 AMPLIFICATION OF FLOATING HIGH-LEVEL SIGNAL

The common-mode voltage, E_{CM} appears at both input terminals, but is rejected if the following conditions are satisfied:

(a) Amplifier no. 1 has precisely unity gain and 180 degrees phase-shift.

(b) $R_1=R_4$

(c) $R_3=R_4$

Under these conditions:

$$i_2=-\left[\frac{E_{IN}}{2R_1}+\frac{E_{CM}}{R_1}\right]$$

$$i_3=\frac{E_{CM}}{R_4}-\frac{E_{IN}}{2R_4}$$

$$\text{and,}\quad E_0=-R_5[i_2+i_3]$$

Therefore, since $R_1=R_4$:

$$E_0=-R_5\left[-\frac{E_{IN}}{2R_4}-\frac{E_{CM}}{R_4}+\frac{E_{CM}}{R_4}-\frac{E_{IN}}{2R_4}\right]$$

$$\therefore\quad E_0=\frac{R_5.E_{IN}}{R_4}$$

The common-mode voltage has thus been rejected and the input voltage is amplified by the factor $\frac{R_5}{R_4}$. The value of R_4 is fixed (it is determined by the minimum permissible resistance to earth for the transducer) so that R_5 is chosen to give the necessary gain. The resistor values should be chosen in accordance with the general rules for analogue modules – see part 3. Generally, suitable values for R_1, R_2, R_3 and R_4 are 100 kilohms and in practice series, variable trimmers should be provided to allow a precise balance to be obtained. Signal-to-noise ratios of 5000:1 can usually be obtained.

3.1.4 D.C. Current Input Signals

Transducers which have current output signals are frequently encountered. These signals are accommodated by converting the current to a voltage by completing the circuit via a stable, precision resistor. The resistor value should not normally exceed say, 200 ohms, which for 5mA full scale, produces a full-scale voltage of 1 volt. If one side of the transducer can be earthed, the voltage across the resistor can be applied directly to the ARCH analogue modules

(except multipliers and dividers, which require a buffer-amplifier). For transducers which cannot be connected directly to earth, the technique described in section 3.1.3 must be adopted. Alternatively, particularly where other low-level signal inputs are required, a low-level signal amplifier may be used on a 'shared' basis by means of an analogue selection module.

Most so-called d.c. current-output transducers, in fact, provide a rectified output current which contains harmonics of the power frequency. Consequently, a filter is usually required to attenuate this 'ripple'; generally a simple two-stage R-C filter inserted between the amplifier and the conversion resistor is adequate. For applications, where the input signal is with respect to earth, a standard ARCH low-pass Amplifier may be used. (See part 8).

3.1.5 A.C. Input Signals

Signals from a.c. plant transducers are converted to high-level d.c. (5 volts) by means of an ARCH a.c./d.c. converter. This unit accepts standard 0-0·5 volt r.m.s. 50c/s input signals which are one-side-earthed and are usually the outputs of differential transformers. The conversion is performed by means of an ARCH operational amplifier, with feedback via semi-conductor diodes, the arrangement performing a linear demodulation. The d.c. output is smoothed in a further low-pass amplifier.

3.2 Digital Inputs

It is frequently necessary to connect inputs from decade switches, on/off switches, shaft digitisers, transducers with digital outputs, etc. into ARCH digital machines. Modules are provided to cater for all these forms of input (see part 8) and it is only necessary to ensure that the inputs have the correct levels of 'on' and 'off'.

ARCH digital modules accept digital inputs with levels of 0V and −10V, these levels being represented in the machine by 1 and 0 respectively. Digital input signals may also be subject to common-mode and series-mode noise which must be taken into account before the inputs are connected to the digital modules.

4.0 OUTPUT SIGNALS FROM ARCH

The output signals from ARCH may take several forms depending upon the system requirements. Analogue signals may be presented to the operator by means of indicating and/or recording instruments and digital signals by means of visual displays, electric type-writers, etc. (see part 4). Output signals may also be required to adjust automatically the 'set-values' of process controllers. In ARCH such signals are derived from Link mechanisms.

4.1 The ARCH Link Mechanism

The Link mechanism provides the means of communication between the computing system and the plant. It converts the outputs of the computing modules into a form suitable for the remote set-value adjustment of proportional, two- or three-term controllers. It allows the set-value to be raised or lowered in small, discrete steps under the control of the computer output signal. The set-value is indicated by a pointer and calibrated dial and adjustable stops are provided to allow the permissible range of set-value variation to be restricted within given

low and high limits. Link mechanisms are available to suit the usual types of controller, namely:

(a) pneumatic	3 to 15 p.s.i.,
(b) a.c.	0 to 0·5 V r.m.s., 50 c/s.,
and (c) d.c.	0 to 10 mA or 0 to 5V, etc.

Alternative types are available having 100 steps or 400 steps.

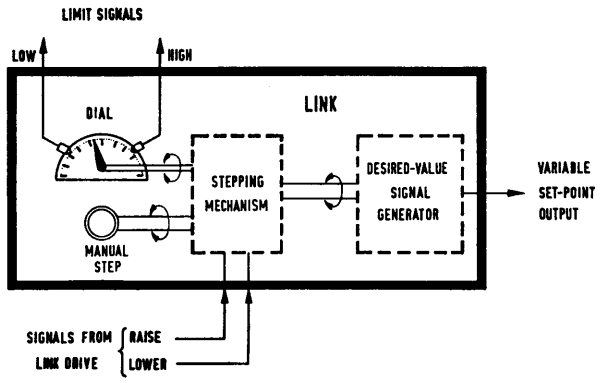


Figure 9 BLOCK DIAGRAM OF LINK MECHANISM

Figure 9 shows a Link mechanism in block schematic form. A stepping-mechanism is pulsed by a 'drive' module, which receives its input from the computer. Two lines are provided, one for 'raise' and one for 'lower'. The low and high limit stops are both mechanical and electrical; when the indicated limit is reached, a micro-switch operates and provides a warning signal to the computer (an alarm signal can also be given to warn the operator). Should the warning signal be ignored, the mechanical stops prevent the variation of the set-value by more than one step beyond the limit. In use, the Link-drive is made inoperative in the event of a computer failure, and the link mechanism remains at its last setting. This arrangement provides a 'fail-safe' characteristic since the process-controller continues to operate safely, but not necessarily at optimum performance. The operator can, if necessary, intervene and adjust the set-value by manual operation of the Link mechanism, using the control-knob which is provided.

Means must be provided to inform the computer of the set-value. This is realised by converting the Link mechanism output to the standard 5 volt d.c. ARCH signal and feeding this back to the computer as an input signal. This feedback signal is compared with the desired set-value and the deviation is used to control the drive-module and hence the stepping mechanism. In order to avoid 'hunting' i.e. alternatively raising and lowering the set-value by one step, a 'dead-band' of plus and minus one half a step is introduced. In ARCH digital machines the comparison and control is carried out by the program, whereas in an analogue machine a special comparator and drive module is used.

ARCH MANUAL

6

part 6

DIGITAL PROGRAMMING TECHNIQUES

ARCH

reg'd trade mark

ARCH MANUAL PART 6

contents

S1106/3 11.63

ARCH MANUAL PART 6

contents (cont.)

ARCH MANUAL PART 6

contents (cont.)

appendices

DIGITAL PROGRAMMING TECHNIQUES

1.0 INTRODUCTION

ARCH digital machines function by obeying previously-prepared programs of simple instructions automatically and at high speed.

In order to prepare such a series of instructions, a complete understanding of the task under consideration is essential. The program writer (programmer) must be able to express the steps necessary to solve the problem as a sequence of operations to be performed by the machine.

To be able to accomplish this, it is necessary to have a knowledge of the operations that a machine can perform and the way in which these are specified in a program.

This part of the ARCH manual constitutes an introduction to programming ARCH digital machines. It is specifically intended for the reader with little or no knowledge of computer programming.

1.1 Computer Structure

A programmer is primarily concerned with the functional structure of a machine and the way in which the various operational modules can be used to deal with a particular problem.

For programming purposes the operational modules are considered as being in one of the following functional categories: input/output, arithmetic and storage.

1.1.1 Input/Output

This category embraces all the devices and channels which either introduce data into the machine or enable data to be sent out; typical of these functions are:

Punched paper-tape input.
Punched paper-tape output.
Input of on/off signals from plant.
Input of data from analogue-to-digital converters.
Output of data to visual displays.
Drive to Link mechanisms.
Output of on/off controls to the plant.

1.1.2 Arithmetic

The basic functions of addition, subtraction, multiplication and division are performed by the various arithmetic modules. These modules also perform functions such as shifting, collation and modification; these functions are described in detail later.

1.1.3 Storage

The data stored in a machine consist of the instructions to be obeyed (i.e. the program) and numbers used in the execution of that program. The program is placed in the machine

before the calculation commences, together with constants etc. relevant to the problem. When the program is started, additional data collected direct from the process or plant are also stored together with calculation results, sub-totals and so on.

All the storage in a machine is divided into many different compartments (*locations*). Each location can hold one unit of data, i.e. one number or one machine instruction known as a 'word'. The locations are numbered from 0 upwards, the number designated to each location being referred to as its address. The word stored in a location, whether it be a number or an instruction, is referred to as the location content. For example, the word stored in the location whose address is 1005 may be referred to as 'the content of location 1005'.

This is more usually written *c* (1005) or in more general terms *c*(N).

The process of putting a word into one of these storage locations is known as writing into that location, and the process of getting a word out, as reading from it.

There is one exception to these rules concerning storage. There is a special one-word store known as 'the accumulator'. It does not have an address as such, and its content is usually denoted by *c*(acc.).

The accumulator is the store to and from which all inputs and outputs are taken and to which all arithmetic operations are performed.

For example, when the machine performs an addition function, the content of a specified location in the main store *c*(N) is added to the content of the accumulator *c*(acc.) and the result is placed back in the accumulator.

In shortened form this is written as:

$$c(\mathrm{N}) + c(\mathrm{acc.}) \rightarrow c(\mathrm{acc.})$$

2.0 PROGRAM INSTRUCTIONS

A computer program consists of a series of instructions. This section gives a general description of the various types of these instructions and their combination into a simple program. A more detailed description of the complete range of ARCH instructions is given later.

2.1 Types of Instruction

The computer program must contain explicit instructions for every action required. These instructions fall into two categories, arithmetic and control.

2.1.1 Arithmetic Instructions

These instructions cause arithmetic functions such as addition, subtraction, etc., to be performed on the number in the accumulator. As mentioned in section 1.1.3 these usually also involve the content of a specified store location. The result of an arithmetic function is always placed in the accumulator ready for the next machine operation.

2.1.2 Control Instructions

All instructions apart from those relating to arithmetic functions are control instructions. These deal with the routing of data into and out of the accumulator and the determination of the order in which the instructions are to be performed.

2.2 Form of Instructions

Each machine instruction is composed of two parts denoted by F & A.

F specifies the function, i.e. the operation to be performed. This is a code number between 0 and 31, each number signifying a particular machine function.

A is a number which is either the address of a store location or a further specification of the function. When used for specifying the address of a store location it has a range of 0 to 8191 inclusive.

2.2.1 Examples of Instructions

The following table contains a few examples of ARCH instructions giving particulars of the name, coding and operation of each one.

Name of Instruction	Function F	Address A	Operation Performed
Replace	**01**	**N**	*The content of the accumulator is replaced by c(**N**). c(**N**) remains unchanged.*
Write (into store)	**02**	**N**	*The content of the accumulator is written into location* **N.** *c(acc) remains unchanged.*
Subtract	**04**	**N**	*c(**N**) is subtracted from c(acc) and the result is placed in the accumulator. c(**N**) remains unchanged.*

As can be seen from the operations performed by these instructions, the first two deal with routing data into and out of the accumulator and are therefore control instructions, whereas the last is an arithmetic instruction.

2.2.2 Examples of Simple Programs

Example 1. Suppose that part of a program requires that the content of location 9 is subtracted from the content of location 8 and that the result is to be placed in location 10. The numbers in locations 8 and 9 are x and y respectively.

The instructions which the computer must obey are given below, together with an indication of the relative states of the accumulator and the store locations involved.

		Contents after obeying each instruction			
F	**A**	*c(acc)*	*c*(8)	*c*(9)	*c*(10)
	Initially	~	x	y	~
01	8	x	x	y	~
04	9	x-y	x	y	~
02	10	x-y	x	y	x-y

It should be remembered that the program itself is stored in storage locations and therefore these locations cannot be used as 'working space', i.e. locations needed for storing calculation data and results. When writing down a series of instructions it is usual to write

down the location in which each instruction is stored. Thus, in this example, if the program were stored in locations 100 upwards, it would be written:

Instruction Address	F	A
100	01	8
101	04	9
102	02	10

Stationery for use in program preparation is pre-printed with these column headings as will be seen later.

The three instructions in locations 100 to 102 are taken in turn automatically by the machine. When c(100) has been obeyed c(101) is selected and obeyed and so on.

This procedure continues in sequence unless the program instructs otherwise. For a number of reasons, however, it is frequently necessary to break this sequence and to follow an alternative branch in the program should certain conditions exist.

A typical case occurs in the following example.

Example 2. c(532) and c(383) are both unknown positive numbers and it is required to store the larger of the two in location 40.

In order to establish the method of approach it is of value to draw a program flow diagram (see figure 1).

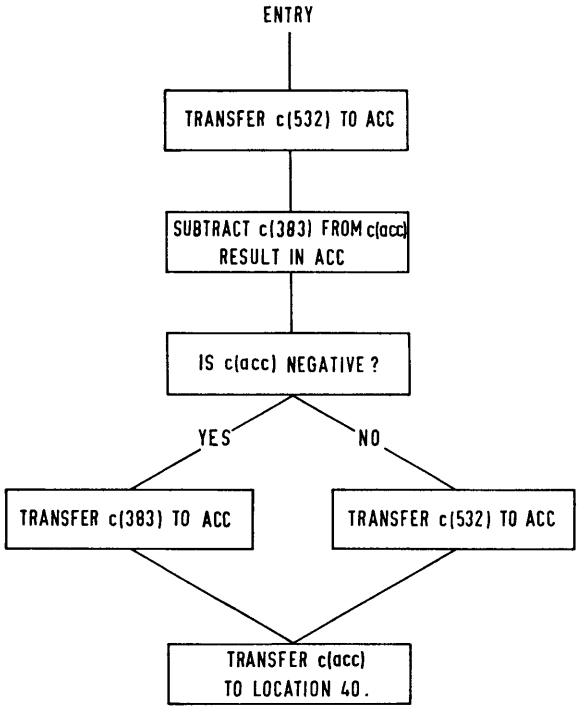


Figure 1 PROGRAM FLOW DIAGRAM

Each of the steps in this flow diagram represents a program instruction and it can be seen that after the third instruction, the program transfers to one of two alternative instructions dependent on whether c(acc.) is positive or negative. This instruction is known as a 'conditional transfer'.

N.B.: c(acc.)=0 is treated as positive (see section 3.2).

The function for conditional transfer is function 09 and its form is:

Name	F	A	Notes
Conditional Transfer	**09**	**N**	*If the number in the accumulator is positive or zero, the computer takes no action other than to proceed to the next instruction in the normal sequence. If the accumulator is negative, (i.e. <0) the computer breaks normal sequence and obeys the instruction in location* **N** *and then continues to obey instructions in sequence from that new location address.*

Thus, the required program could be as follows:

Instruction Address	F	A	Notes
100	01	532	*c(532)→c(acc)*
101	04	383	*c(acc)−c(383)→c(acc)*
102	09	230	*Transfer control if c(acc) negative*
103	01	532	*c(532)→c(acc)*
104	02	40	*c(acc)→c(40)*
230	01	383	*c(383)→c(acc)*
231	02	40	*c(acc)→c(40)*

Example 3. Another case for the use of a transfer-control function occurs in a program 'loop'. This is usually a set of instructions which is repeated a number of times before proceeding to the next section of the program.

Suppose for example that it was required to keep repeating the program in example 2. Thus, whenever the larger of the two numbers has been stored in location 40 the machine must be transferred back to the first instruction in order to start all over again.

The function for unconditional transfer is function 8:

Name	F	A	Notes
Unconditional Transfer	**08**	**N**	*The address of the next instruction in the current series is transferred to a predetermined location and a new series of instructions is entered, starting with the instruction contained in location* **N.**

The flow diagram thus becomes as shown in figure 2.

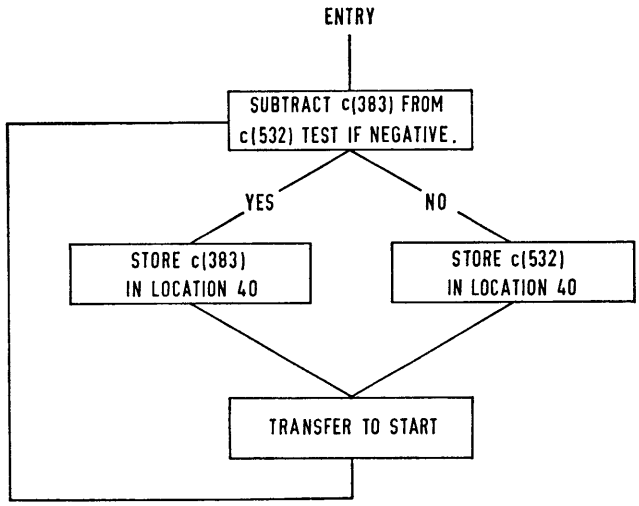


Figure 2 FLOW DIAGRAM SHOWING TRANSFER OF CONTROL

and the program becomes:

	Instruction Address	F	A	Notes
ENTRY	100	01	532	
	101	04	383	*Subtract*
	102	09	230	
	103	01	532	
	104	02	40	*Store c*(532) *if greater*
	105	08	100	*Transfer to start*
	230	01	383	
	231	02	40	*Store c*(383) *if greater*
	232	08	100	*Transfer to start*

The reason for storing the next instruction in the current series before transferring control, is to enable this series of instructions to be re-entered after completing the program loop. This is particularly useful if the same loop is entered from several different locations.

3.0 REPRESENTATION OF DATA

The data stored in a digital computer consist of the program instructions and the numbers used in the execution of that program. The previous section deals with the representation of the program instructions and in this section the representation of the numbers used in the execution of that program is considered.

3.1 Binary Notation

All information in the computer is stored in binary code as mentioned in the section describing the ARCH digital modules. The basic principles of binary notation are given in

appendix I. Although a complete understanding of binary mathematics is not essential in order to write a program, an appreciation of its principles is of value. In binary notation, each digit position represents a specific power of 2, and the presence or absence of that particular power of two is indicated by a 1 or a 0 respectively. The lowest power of 2 is always written in the least-significant position. Thus the binary number 10111 (reading from left to right) represents:

$$\begin{aligned} & 1.2^4+0.2^3+1.2^2+1.2^1+1.2^0 \\ =&16 + 0 + 4 + 2 +1 \\ =&23 \end{aligned}$$

Similarly, binary digits to the right of a binary point represent negative powers of 2, i.e. $\frac{1}{2}$ $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$, etc. and are referred to as *binary fractions.*

Binary Fraction	Fraction	Decimal
·1	$(1\times\frac{1}{2})$	·5
·11	$(1\times\frac{1}{2})+(1\times\frac{1}{4})$	·75
·1101	$(1\times\frac{1}{2})+(1\times\frac{1}{4})+(0\times\frac{1}{8})+(1\times\frac{1}{16})$	·8125

3.2 Machine Range and Number Representation

In keeping with all calculating devices, digital computers impose some limit upon the range of numbers which can be represented directly. This does not present any restriction on the use of numbers outside this range however.

Use of a slide rule illustrates this point when multiplying, for example, −·0005 by 20·5. This can be accomplished without any difficulty despite the fact that the slide rule may only be graduated in the range of +1 to +10. The reason why this is possible is, of course, the fact than when using the slide rule the operands are scaled and the result has to have a scale-factor applied also.

Similar techniques are used in computers when numbers are outside the range which can be represented directly.

In ARCH digital computers this range is from −1 to just less than +1. This range is arrived at by assuming that the binary point lies between the two most-significant binary digits of a word. All binary digits to the right of this binary point therefore represent negative powers of 2 as described above. The most-significant digit of a word is known as the *sign* digit, a zero in this position indicating a positive number and a one, a negative number.

The following examples illustrate this representation in a word having 18 binary digits.

Positive numbers

(1) **0 11010 00000 00000 00** (The grouping is purely for convenience of interpretation)
$=+2^{-1}+2^{-2}+2^{-4}$
$=+\frac{13}{16}$ or +·8125

(2) **0 00000 00000 00000 01**
$=+2^{-17}$
This is the smallest positive number possible by direct representation.

(3) **0 11111 11111 11111 11**
$=+2^{-1}+2^{-2}+\ldots\ldots\ldots\ldots+2^{-17}$
$=+1-2^{-17}$
This is the largest positive number possible by direct representation.

Negative numbers

(1) **1 01100 00000 00000 00**
$=-1+2^{-2}+2^{-3}$
$=-\frac{5}{8}$ or $-\cdot 625$

Note that the most-significant digit is considered as representing −1 and all subsequent binary digits as representing positive fractions.

(2) **1 00000 00000 00000 00**
$=-1$
This is the largest number with a negative sign possible by direct representation.

(3) **1 11111 11111 11111 11**
$=-1+2^{-1}+2^{-2}+2^{-3}\ldots\ldots\ldots+2^{-17}$
$=-2^{-17}$
This is the smallest number with a negative sign possible by direct representation.

NOTE: Zero is represented as 0 00000 00000 00000 00. Since this has a zero in the sign-digit position, it is considered as being a positive number.

3.3 Scaling

The natural range of the machine is such that if in an 18-digit word, a number is not between -1 and $1-2^{-17}$, it must be scaled.

This is usually the case and often all numbers have to be scaled. The way in which a number is scaled is subject to a number of considerations and requires care to prevent out-of-scaling (or *overflow* as it is more commonly known) from occurring when numbers are added, multiplied, etc.

To illustrate this point, consider the case where it is required to store the number 29. In binary representation this is **11101.** These five digits can of course occupy any five consecutive positions in an 18-binary digit word. However, if the most-significant of the five digits were placed in the most-significant digit position of the word, it would be interpreted as the negative number given by $-1+\frac{1}{2}+\frac{1}{4}+\frac{1}{16}$; e.g. **1 11010 00000 00000 00.** This positioning is therefore not permissible.

The binary representation of 29 could be positioned, however, as:

(a) **0 11101 00000 00000 00**
or (b) **0 00000 00000 00111 01**
or (c) **0 00000 00111 01000 00**

(a) and (b) are extremes and (c) is an intermediate case. All versions (and others in between the two extremes) are permissible *provided that the programmer specifies the scale factor.*

Hence: (a) is 29×2^{-5}
(b) is 29×2^{-17}
(c) is 29×2^{-12}

If this scaling is not specified, a variety of decimal values could be wrongly interpreted. For example, it is stipulated that (a) is 29×2^{-5}, but if the scale factor of 2^{-5} is omitted, (a) may be interpreted as 58×2^{-6} or 116×2^{-7} or 232×2^{-8} and so on.

3.3.1 Choice of Scale Factor

The choice of a particular scale depends upon a compromise between the maximum number that can be handled and the size of the smallest step required.

If large numbers are anticipated it may be necessary to apply a scale factor of 2^{-17} as in (b) above. This means that all the digits in the word, except the sign digit, can be used to represent integers, and in this way the maximum positive capacity of an 18 digit-word is $2^{17}-1$

$$=131,071.$$

(This may of course be a 'scaled' number itself, depending upon engineering units involved; e.g. megawatts, cu.ft./min., lbs/hr, etc.)

When a 2^{-17} scale is used, a change in the least-significant digit of the binary number constitutes a change of unity in the represented integer.

Occasions arise where the smallest step required is fractional and other scales have to be used. Considering (c) above for instance, 29×2^{-12}, which is **0 00000 00111 01000 00**, has five binary places to the right of the '29'. Thus a change in the least-significant digit of the word is equivalent to a change of 2^{-5} which is ·03125. At the same time, the maximum positive integer possible with a scale of 2^{-12} is **0 11111 11111 11000 00** which is 4095.

3.3.2 Packing

By scaling techniques it is possible to divide a word up into a small number of discrete sections. This is of advantage when storing a lot of fairly short numbers because one store location can contain more than one number.

Consider an 18-digit word; if it is mentally 'split in half' the effect is:

$$2^{-0}.....2^{-8} \quad \& \quad 2^{-9}.....2^{-17}.$$

It is possible to store one integer with a scale of, say 2^{-8}, in the first half (most-significant half), and another integer with a scale of 2^{-17}, in the second half.

This technique is termed *packing* and in the example considered, the two numbers must be separated from each other when any mathematical function is to be performed upon either of them. A machine function called *collate* is used for this separating operation and is described later in section 4.1.

3.3.3 Arithmetic Operations

As with a slide rule, it is necessary to remember the scale applicable to the result of any arithmetic operation in a digital computer.

Addition and subtraction functions offer no scale problems provided that the two operands have the same scale (and provided that in the case of addition, the sum is within the maximum capacity). If these conditions are satisfied, the result will be correct and have the same scale as the operands.

For example, if 9×2^{-12} is subtracted from 29×2^{-12}, the answer 20×2^{-12} will be produced. Nevertheless, it is most important that the scale of the result is specified, otherwise 20×2^{-12} may be interpreted as 40×2^{-13} or 80×2^{-14} and so on.

Multiply and divide functions demand greater care because even when the two operands have the same scale, the result has a different scale.

For example, 53×2^{-15} multiplied by $5\times2^{-15}=265\times2^{-30}$.

In order to get the same scale factor in the product as in the operands, it is necessary to double the product $\frac{1}{2}n$ times, where n is the power of 2 in the scale factor of the product. This can be achieved by a 'left-shift' (see section 4.2) of $\frac{1}{2}n$ places. In this case, therefore, the product must be doubled fifteen times.

3.4 Data Input and Output

ARCH has provision for the connection of a wide range of input and output channels any combination of which may be used in any particular machine.

The input channels include:

(a) paper-tape
(b) keyboard
(c) digital transducer
(d) 24-hour clock
(e) digitised outputs of analogue signals

The output channels include:

(f) page printer (teleprinter)
(g) typewriter
(h) paper-tape punch
(i) visual display
(j) digital control signals
(k) Link mechanism.

The form of input/output data depends on the type of channel. For instance, channels (c), (d), (e) and (i) usually deal with purely numerical data in which case, the data are represented in pure binary. The number of binary digits varies according to the capacity of the actual channel.

The Link mechanism requires only a raise/lower signal which is specified by one digit coded as 0=raise, 1=lower. Similar two-state signals also apply to channel (j).

Input/output data need not necessarily be restricted to purely numerical data and channels (a), (b), (f), (g) and (h) may involve alpha/numeric data, i.e. letters and numbers. For this reason, the binary values of the data in these cases are allocated specific meanings according to a laid-down code known as *telecode*.

3.4.1 Telecode

Paper-tape input/output data are recorded by punching holes on a length of paper tape. Up to five information-holes can be punched across the width of a tape. This group of between 0 and 5 information-holes across the tape is called a *telecode character*.

A character always contains a small hole between the second and third most-significant information-hole positions. This is known as the sprocket hole and is required for synchronization purposes only.

The presence of an information hole in the tape represents a 1, and the absence of a hole a 0. In this way a character constitutes a five-digit binary number between 00000 and 11111 and these combinations are used to code all the figures, letters and control signals required. Telecode is also used for the teleprinter and typewriter output channels, although in these cases the telecode character is sent out in the form of electrical pulses and not as punched paper-tape, and the equivalent alpha/numeric character is printed on a sheet of paper. The complete Elliott telecode is given in appendix III. It will be seen that any group of five digits has two possible meanings, i.e. letters or figures. The required interpretation has to be indicated by the prior use of 'figure-shift' or 'letter-shift' characters in the same way that a typist uses her shift keys. The typewriter output channel has no figure-shift/letter-shift facility and can therefore only handle up to 32 different characters from a five-binary-digit telecode output.

It will be noticed that in the case of numbers the four least-significant digits in a character have the binary form of the number represented. Conversion between telecode numbers and binary numbers is thus very simple.

3.4.2 Parity

Inspection of the telecode characters shows that characters representing numbers all contain an odd number of ones and this odd number is made up by a one in the most-significant position whenever the binary equivalent of the number consists of an even number of ones.

This additional digit in the left-hand position is termed the *parity* digit.

This technique is used to detect errors in reading or punching. For instance, if owing to a failure an extra hole were punched, a required hole were omitted or all holes were omitted, the resultant character would not contain an odd number of holes. In this way, failures of this nature can be detected in numerical data to the computer by a parity-check facility.

4.0 INSTRUCTION CODE

Each program instruction is composed of two parts, denoted by F and A. F specifies the function, i.e. the operation to be performed, and A specifies either the address of a storage location or further specifies the function.

This section details the total range of instructions available in ARCH. This is known as the ARCH instruction code and a summary of this code is given in appendix II.

N.B. In practice in any particular ARCH machine, the number of instructions available is dependent on the modules used.

4.1 Basic Functions

The A digits in these functions always specify a store location.

4.1.1 Function 01 Replace

The content of the accumulator is replaced by the content of the location specified by A.

Example	F	A	
	01	2731	*Replace the content of the accumulator by the content of location 2731.*

The content of the specified location remains unchanged by the function.

4.1.2 Function 02 Write (into the store)

The content of the accumulator is written into the location specified by A. The accumulator content remains unchanged.

Example	F	A	
	02	1020	*Write the content of the accumulator into store location 1020.*

4.1.3 Function 03 Add

The content of the store location specified by A is added to the content of the accumulator. The sum is placed in the accumulator.

Example	F	A	
	03	500	*Add the content of location 500 to the content of the accumulator.*

When the function has been carried out the sum is in the accumulator. The content of the specified location remains unchanged.

4.1.4 Function 04 **Subtract**

The content of the store location specified by the address is subtracted from the content of the accumulator. The difference is placed in the accumulator.

Example:	F	A	*Subtract the content of location 230 from the content of the accumulator.*
	04	230	

The difference is placed in the accumulator, and $c(230)$ is unchanged.

4.1.5 Function 10 **Collate**

The contents of the accumulator and of the location specified by the address are combined to form a new word which has a 1 in each digit position in which there are 1's in both the original words. All other digit positions in this new word contain 0's. The content of the accumulator is then replaced by this new word.

A typical use of this function is the extraction of a number or telecode character from a location which contains more than one item, e.g. when numbers have been packed (see section 3.3.1).

Example: $c(200)$ is **0 00000 11100 00111 11** which represents the two numbers 7×2^{-8} and 31×2^{-17} packed in the same location. It is required to extract the former number. This is achieved by collating $c(200)$ with the content of another storage location (57, say), which has all ones in the most-significant nine digit positions and all zeros in the least-significant eight positions, thus:

$c(200)$	**0**	**00000**	**11100**	**00111**	**11**
$c(57)$	**1**	**11111**	**11100**	**00000**	**00**
collate $c(200)$ & $c(57)$	**0**	**00000**	**11100**	**00000**	**00**

The word has 1's only in the digit positions which contained a 1 in both $c(200)$ and $c(57)$. In other words, the number 7×2^{-8} which was required has effectively been 'extracted' from $c(200)$.

The 'program' to achieve this is as follows:

Instruction Address	F	A	Notes
5	01	200	$c(200)\rightarrow c(acc)$
6	10	57	Collate $c(acc)$ & $c(57)\rightarrow c(acc)$
			$c(200)$ *and* $c(57)$ *both remain unchanged*

4.1.6 Function 05 **Set Modifier**

An instruction containing this function causes the next instruction in sequence to be modified by the arithmetic addition to it of the content of the location (modifier) specified by the address digits.

Example: It is required to obtain the total output of a process by summing the outputs of 130 batches previously stored in locations 501 to 630. This must be achieved by 130 addition instructions, e.g.

```
03  501
03  502
03  503  etc.
```

Using the 05 function the necessity of storing 130 different instructions is eliminated by storing one addition instruction and *modifying* its address by one each time it is obeyed. A program to show the use of this instruction to solve the above example is given on the next page.

ARCH programme sheet

SHEET No. 1

Date 20th July 1962
No. of Sheets 1
Ref. No. A.M.6 Prog.1

Programme Obtaining the total process-output from individual batch outputs
Section 4.1.6

Notes	Address	Instruction F	Instruction A	Notes
	700	00	0	Zero
	1	00	1	Subtracting constant
	2	00	129	Count constant - modifier
ENTRY →	3	01	700	Clear accumulator
712 →	4	05	702	Set modifier with c(702)
	5	03	501	Modified add instruction
	6	02	40	Sub-total or total to loc. 40
	7	01	702	} Decrease modifier
	8	04	701	}
	9	09	→	EXIT ADDRESS Test if end of additions
	710	02	702	Store new modifier
	1	01	40	Prepare for next addition
	2	08	704	Re-enter loop
	3			
	4			
	5			
	6			
	7			
	8			
	9			
	0			
	1			
	2			
	3			
	4			
	5			
	6			
	7			
	8			
	9			

In this program, a function 8 instruction (see 4.3.1) is used in location 712 to make a program *loop*. The numbers are added, starting from location 630 and working downwards to 501, by decreasing the modifier by 1 each time round. This is done in order to be able to use a function 09 instruction (see 4.3.2) to detect the end of the loop.

It should be noted that by using function 05, the problem may be programmed using 13 locations instead of the 130 needed in the first approach.

4.1.7 Function 16 **Multiply**

The contents of the accumulator and the location specified by the address are multiplied together. The content of the specified location remains unchanged.

When two 18-digit binary numbers are multiplied together, a 35-digit product results, so that at the end of a multiplication function, the 18 more-significant digits of the product are placed in the accumulator and the 17 less-significant digits in an auxiliary register.

4.1.8 Function 17 **Divide**

The number formed by the combined contents of the accumulator and the auxiliary register is divided by the content of the location specified by the address.

The result of the function (the quotient) is placed in the accumulator. The least-significant digit of the quotient is 'rounded off' automatically, there is no 'remainder' after a division function.

4.1.9 Function 18 **Accumulator to Auxiliary Register**

This function is used in association with the multiplication and division functions. The 17 least-significant digits of the accumulator replace the content of the auxiliary register, the content of the accumulator remaining unaltered.

4.1.10 Function 19 **Auxiliary Register to Accumulator**

This is the converse of function 18 and is used after multiplication. For example, suppose 7×2^{-17} is multiplied by 30×2^{-17}. The result is 210×2^{-34} (see sections 3.2 and 3.3), which represents the binary number equivalent to 210 stored in the least-significant eight digit positions of the auxiliary register. In this case, therefore, the content of the auxiliary register must be transferred to the accumulator before the number is processed further.

4.1.11 Function 20 **Square Root**

This function makes no use of the address digits and is written:

F	A
20	0

This instruction computes the square root of *c*(acc.) and places the result back in the accumulator. This function can be performed only on positive fractions.

The number whose square root is to be found is regarded as a 'double-length' number with the least-significant half all **0's**. This produces a 'full-length' root with the same scale factor as the original number.

4.2 Shift Function

These functions cause the data held in the accumulator to be shifted by a certain number of digit positions (places) left or right.

For example, if a word **0 00011 01111 00101 10** is shifted left three places, it becomes **0 11011 11001 01100 00** . If this is shifted right four places it becomes **0 00001 10111 10010 11**

Thus, shift functions can be used to manoeuvre a binary pattern or number into specific digit positions.

Shifting can also be used either to accomplish multiplication and division by powers of two or alteration of scale factor.

For example, if the number **0 00000 00000 00001 11** representing 7×2^{-17} is shifted two places left it becomes **0 00000 00000 00111 00** which represents 28×2^{-17} or 7×2^{-15}.

In this straightforward type of shift, the digits shifted 'off' one end are lost and zeros are inserted at the other end. A special form of shift known as 'end-around' prevents the loss of these digits. In this type of shift, the digits shifted off one end appear in the same order at the other end.

For example, if a word **1 01110 00110 01000 10** is shifted left (end around) three places, it becomes **1 10001 10010 00101 01**

The straightforward shift alters the position of the sign digit and cannot therefore be used for negative numbers. In order to deal with these numbers therefore, a special form of shift known as arithmetic shift must be used.

For example, the word **1 01100 00000 00000 00**

represents $-1+\frac{1}{4}+\frac{1}{8}$

$=-\frac{5}{8}$.

Shifted right one place (arithmetically) it becomes

1 10110 00000 00000 00

i.e. $-1+\frac{1}{2}+\frac{1}{8}+\frac{1}{16}$

$=-\frac{5}{16}$.

The whole word has been shifted right one place and the sign digit regenerated.

4.2.1 Function 12 **Shift Right One Place (Arithmetic)**

This function shifts the content of the accumulator one place to the right and preserves the sign digit. The address digits have no significance in this type of instruction and are regarded as zero.

4.2.2 Function 13 **Shift Left One Place (End Around)**

The content of the accumulator is moved to the left and the previously most-significant digits are shifted to the least-significant positions.

4.2.3 Function 14 **Multiple Shift Right (Arithmetic)**

This is similar to function 12 except that the number of shifts right is specified by the A digits of the instruction. The maximum value of A is equal to the word-length in the accumulator.

4.2.4 Function 15 **Multiple Shift Left (End-Around)**

In a similar way to function 14, the number of shifts left is specified by the A digits of the instruction up to the same maximum value of A.

4.3 Transfer-Control Functions

It is frequently necessary to break the normal sequence of a program and transfer to instructions stored elsewhere.

Two functions provide this facility.

4.3.1 Function 08 Unconditional Transfer

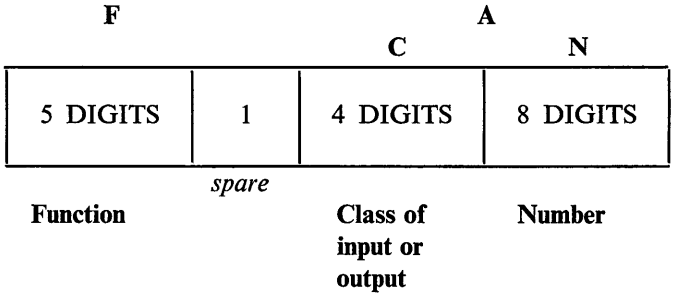
This causes the machine to obey the instruction stored in the location specified by A instead of the instruction following in the normal sequence. At the same time, this function also transfers the address of the instruction following in the normal sequence to a predetermined location. The use of this is illustrated in section 6.3.

4.3.2 Function 09 Conditional Transfer

If the number in the accumulator is positive or zero, the machine takes no action other than to proceed to the next instruction in normal sequence. If *c*(acc.) is negative, however, the machine breaks the normal sequence and obeys the instruction in location A. It then continues to obey instructions in sequence from that new location address.

4.4 Input/Output Functions

The A (or address) digits of all input/output instructions are used to specify further the function. Generally speaking, input and output are to and from the accumulator, although in some cases the content of the accumulator is used to specify further the function. The allocation of digits in these instructions is as follows:



The **class** digits specify the type of input or output, e.g. tape reader, tape punch, typewriter, keyboard, etc. The **number** digits specify the channel of a specific class, i.e. which typewriter, which keyboard, etc.

The number and type of input/output functions available and used in a specific ARCH system depends upon the application. The most common of these functions are given below.

4.4.1 Function 06 Output

Output functions have the following classes:

Class 1 **Trigger A.D.C.**
An output instruction in this class causes the analogue-to-digital converter (A.D.C.) specified by the N digits, to be triggered i.e. to start the conversion process.
When the analogue input has been switched to the A.D.C. input by means of an analogue selection instruction (Class 12), the analogue selection busy must have cleared before this instruction can be given.

Class 2 **Control**

One of the control facilities in ARCH machines is that of priority interrupt. This is provided so that should a particular condition occur in say, a process, the machine may be interrupted from its normal routine to give priority to the investigation of that condition. The N digits associated with this facility are restricted to having three different values, each value affecting the interrupt facilities. These are:

- N=0 Reset interrupt demand.
- N=1 Prohibit interrupt. This is used to protect a vital section of the main program from interruption.
- N=2 Release interrupt. This releases the prohibition on interrupt.

This facility is described in greater detail in section 4.4.3.

A fail-safe feature of ARCH machines is the watchdog facility. The watchdog is a section of the machine which is continually attempting to give a system self-check failure alarm and is prevented from doing so by regular reset signals. These reset signals are produced by 06 class 2 N=3 instructions.

In an on-line program it is usual to include a self-contained test program using the basic machine functions to 'compute' a number that is the binary equivalent of a 06 class 2 N=3 instruction. Such an instruction is known as a pseudo-instruction, the principles of which are described in section 4.4.3. Thus, if the test program does not function correctly the watchdog is not held back and a system-failure alarm is given. The interval between reset signals is normally 1 minute.

Class 3 **Paper-Tape Punch Output**

The five least-significant digits of the accumulator are transferred to the paper-tape punch output module specified by the N digits.

Class 4 **Printer Output (Teleprinter)**

The printer is specified by the N-digit positions of the instruction and the telecode character to be output must be positioned in the five least-significant digit positions of the accumulator.

The effect of a class 4 output instruction is that the telecode character held in the least-significant five digits of the accumulator is transferred to the appropriate printer-drive module which then causes the required character to be printed.

Class 5 **Digital Output (1 bit on/off)**

The appropriate on/off output signal channel is given by the N digits. If the content of the accumulator is negative, the channel gives an 'off' or 'reset' output. If the accumulator is positive, an 'on' or 'set' is output.

Class 6 **Visual Display**

The N digits of this type of instruction specify the number of a particular decade of visual display, and the four least-significant digits in the accumulator are transferred to the visual display module for decoding and display on the specified decade indicator.

Class 7 **Alarm Indicator**
The alarm indicator concerned is identified by the number given by the N digits.
If the content of the accumulator is negative, an 'alarm' condition is indicated and if it is positive, a 'normal' condition is signalled.

Class 8 **Typewriter Output**
The telecode character held in the five least-significant digits of the accumulator is transferred to the typewriter output module specified by the N digits of the instruction.

Class 9 **Strip-Printer Output**
The N digits specify the number of the strip-printer (line printer) required and the telecode character held in the five least-significant digits of the accumulator is transferred to the specified strip-printer output module.

Class 10 **Link Mechanism Output**
The N digits specify the number of the Link mechanism concerned. If the content of the accumulator is negative, the Link set-point output is lowered by one step. If the accumulator content is positive, the Link output is raised by one step.

Class 11 **Digital-to-Analogue Converter**
The content of the accumulator is transferred to the digital-to-analogue converter specified by the N digits and conversion to analogue representation is initiated. The number of digits depends on the type of converter used.

Class 12 **Analogue Selection**
Selection of an analogue input signal is achieved by transferring the content of the accumulator to the analogue selection module into which the group of inputs specified by the N digits is connected. The contents of the accumulator (3 B.C.D. digits in the range 0-999) specifies the point number of the analogue signal in that group.

Class 13 **Decade Switch Reading (I.M.I.C.)**
This class is concerned with preparing to read data set on the decade switches of ARCH I.M.I.C.s (Industrial Manual Input Consoles). Apart from these decade switches, each I.M.I.C. has a pushbutton and an OK, WAIT and QUERY lamp display. This display automatically goes to WAIT after its associated pushbutton has been pressed and has to be set to OK or QUERY by program. The N digits associated with this class are given below.

N = 0 Illuminate QUERY lamp
N = 1 Illuminate OK lamp

Where more than one I.M.I.C. exists in a system, the lamp illuminated by the above instruction is that associated with whichever of the pushbuttons has been pressed.

N = 2 – 255 Select and prepare to read specified decade switch (each switch in a system is allocated one number in this range).

Reading the switches is carried out by an Fn.07 Class 5 instruction.

4.4.2 Function 07 **Input**

The format of input functions is very similar to the output functions and the following classes apply.

Class 1 **Analogue-to-Digital Converter**
The A.D.C. concerned is addressed by the N digits and the number from the converter is transferred to the most-significant digit positions of the accumulator.

Class 2 **Control-Digits**
Control digits are special one-bit digital inputs which enable the program to detect the state of peripheral devices. Each digit represents the state of a peripheral device; thus, the two states of a control-digit may indicate whether or not a printer is busy (i.e. printing).
Control-digits may be derived from any sort of on/off signals such as those from pushbutton switches, mechanism contacts, limit switches, contactors, etc. In this way, the state of any external device can be examined before a program branch involving it is pursued.
The control-digit specified by the N digits is transferred into the most-significant (sign) position of the accumulator.

Class 3 **Paper-Tape Reader**
The tape-reader channel is specified by the N digits and the least-significant five digits on the tape are read and transferred to the five least-significant positions of the accumulator. The tape-reader then steps on to the next character.
Note. If the machine is provided with the optional parity-check facility (see class 6) the parity-digit is not read into the accumulator.

Class 4 **Keyboard Input**
The N digits specify the number of the keyboard to be read. The five-digit character from the keyboard is transferred into the five least-significant digit positions of the accumulator.
The pressing of a key on a keyboard is indicated by a 'ready' signal which may be input as a control digit. The keyboard input instructions must be given within 40 ms of the start of this ready signal.

Class 5 **Digital Input (Miscellaneous)**
The digital input source is given by the N digits and the digits so obtained are transferred into the least-significant positions of the accumulator. This class of input enables the outputs of digital-switches, shaft encoders, digital transducers, etc. to be fed into an ARCH machine.

Class 6 **Parity Check**
This class is associated with the optional facility of automatic parity-check on the paper-tape input channels. When a function 07 class 6 instruction is given, the content of a one-digit parity-failure memory is transferred into the most-significant digit position of the accumulator. If this transferred digit is a one, a parity-failure is indicated. This instruction also resets the parity-failure memory. The N digits specify the number of the associated paper-tape reader.

Class 7 **Clock**
This class is concerned with the input of the time indicated by the clock module. The binary-coded decimal digit of time

specified by the N digits is transferred to the least-significant four digit positions of the accumulator. The allocation of N digits is as follows:

N	Input
0	tens of hours
1	units of hours
2	tens of minutes
3	units of minutes

4.4.3 Priority Interrupt

The priority-interrupt facility mentioned in function 06 class 2 is an external control-transfer instruction. Priority-interruption may be demanded by manual operation of the appropriate pushbutton-switch on the machine control-panel. The effect of priority-interruption is similar to that of the unconditional transfer instruction. The normal program sequence is immediately broken and the next instruction is taken from some other location (in this case, from location 0 of a specified store). The action taken by the machine on priority-interrupt thus depends on the series of instructions written in that location onwards. Once a priority-interrupt demand has been made, no further interrupt demand can be made until that demand is reset by a function 06 class 2 N=0 instruction. (This is indicated to the operator by a lamp associated with the priority-interrupt pushbutton). The priority-interrupt facility can be inhibited by means of a function 06 class 2 N=1 instruction. This inhibition is released by giving a function 06 class 2 N=2 instruction. In certain suitably safeguarded circumstances, priority-interrupt demand may come from contacts incorporated in the plant.

4.5 Pseudo-Addresses and Instructions

As described at the beginning of section 4.4, the A digits of an instruction are sub-divided for input and output functions. These two sub-divisions have their own range of numbers.

For example, the C digits, of which there are four, can represent classes from 0 to 15 and the N digits can specify a number from 0 to 255.

Thus, an output instruction to say, Link mechanism no. 5, could be written:

Function	Class	Number
06	10	5

Writing an instruction in this way becomes cumbersome, however, when it is mixed with non input/output instructions. Furthermore, problems arise in reading such an instruction into the machine in the first place.

To overcome this the C and N digits of these instructions are considered to be a pseudo-address.

Consider the C and N digits above.

Class	Number
10	5

these are represented in the form

Class (4 digits)	Number (8 digits)
1010	**00000101**

If this digit pattern is interpreted as the 12 least-significant digits of a basic instruction, i.e. its address, it becomes:

$$\begin{aligned} & 1\times2^{11}+1\times2^{9}+1\times2^{2}+1\times2^{0} \\ = \; & 2048 + 512 + 4 + 1 \\ = \; & 2565 \end{aligned}$$

Thus an alternative way of writing:

Function	Class	Group
06	10	5

is:

Function	Address	
06	2565	where 2565 is a pseudo-address.

On this basis the reader may wish to confirm the following pseudo-addresses:

F	C	N		F	A(Pseudo)
06	1	1	Trigger A.D.C. no. 1	06	257
06	3	1	Output to Tape-punch no. 1	06	769
06	4	2	Output to printer no. 2	06	1026
06	7	67	Output to alarm indicator no. 67	06	1859
07	1	1	Input reading from A.D.C. no. 1	07	257
07	2	5	Input control-digit 5	07	517
07	4	1	Input from keyboard no. 1	07	1025

5.0 PROGRAM SUB-ROUTINES

Routine, as used in programming terminology, is a contraction of the words *Routine program*. As the name implies, this is a complete program which is used as a matter of routine under certain conditions.

In most programs, a series of instructions dealing with a 'sub-problem' is frequently required. These 'sub-problem' instructions could of course be written down every time they are needed, but this is wasteful of storage space. It is normally more advantageous to isolate one set of these instructions in the store and refer to them every time they are required.
A series of instructions such as this is called a sub-routine.

The value of a sub-routine is far-reaching for, having once been written, it may be incorporated in future programs.

For example, in a typical on-line computing system it may be required to perform the calculation $\sqrt{\frac{\triangle p \times P}{T}}$ on perhaps twenty different sets of measurements every few minutes. This may be accomplished by writing a sub-routine to solve the equation $F = \sqrt{\frac{\triangle p \times P}{T}}$ and then when each set of measurements is made, using this sub-routine to calculate F by substituting the measured values on the equation.

When the sub-routine is finished, its last instruction transfers back to the main program at the point of previous exit.

Other useful sub-routines may be: linearize thermocouple output; print out alarm information when a preset limit is exceeded; output telecode characters, etc.

In many applications the amount of multiplication and division is small compared with the problem as a whole, and may not justify the inclusion of the optional arithmetic modules which provide functions 16 and 17 (multiply and divide). In these circumstances multiplication and division can be accomplished by use of sub-routines which achieve these functions by a series of additions and shifts. This 'add-and-shift' principle is illustrated (in decimal) in the following example.

Suppose it is required to multiply 51 and 3 together. The add-and-shift technique is:
(a) add five 3's together; this equals 15
(b) shift this sub-total left one decimal place; this gives 150
(c) add one 3, giving 153.

Multiplication and division using this technique take a few milliseconds longer than if

they were accomplished by functions 16 to 17. This increase in time may be small compared with the total time to complete the main program and the times involved in the process.

If, subsequently, owing to increasing computer load, this method of sub-routined multiplication and division becomes an embarrassment, then the appropriate multiply and divide modules may be added.

Another very important ARCH program routine is that for reading the program and relevant data into a variable-store machine. This of course poses a variation on the old conundrum of 'which comes first' – the input routine or the program.

In ARCH, this input routine may be permanently stored in a fixed-store module, i.e. built into the machine during manufacture. The instructions in this routine are usually referred to as the 'initial instructions' and some of their features are described later.

6.0 EXAMPLES OF PROGRAMS AND SUB-ROUTINES

The programmer must be able to express the steps necessary to solve a problem as a sequence of operations to be performed by the machine. This section gives a few examples of the use of the instruction code in complete programs and sub-routines.

6.1 Example 1

PROBLEM: It is required to generate all the even numbers between 0 and 20 inclusive, add them together to give x, calculate $\frac{1}{2}(x-100)$ and store the solution in storage location 8.

METHOD: This problem can be solved using a number of programs, for example:

(a) a series of consecutive instructions to form the even numbers which are stored separately in locations N to $N+9$. These are followed by successive addition instructions to give x, a subtract 100 instruction and a multiply by $\frac{1}{2}$ (or divide by 2) instruction;

(b) a series of consecutive instructions to form the even numbers and add them together as they are formed, e.g.

add 2 and 0,	store	in	location	100 and 120.
add 2 to c(100),	,,	,,	,,	100 (next even number)
add c(120) to c(100),	,,	,,	,,	120 (sub-total)
add 2 to c(100),	,,	,,	,,	100 (next even number)
add c(120) to c(100),	,,	,,	,,	120 (new sub-total)
				. . . and so on

followed by an instruction to subtract 100 from the so-formed x and another to multiply the result by $\frac{1}{2}$.

Both of these approaches take up many locations containing similar groups of instructions repeated over and over again.

This is a justifiable case for using a program-loop, so an alternative method is:

(c) A small program-loop forming the even numbers and cumulative sub-total followed by a subtract 100 instruction. Since this result is to be multiplied by $\frac{1}{2}$ (i.e. 2^{-1}, a binary number), the multiplication can be achieved by a shift-right instruction as described in section 4.2.

The flow diagram of this approach is shown in figure 3 and the program on the next page.

ARCH

program sheet

SHEET No. 1

Date 23rd July 1962
No. of Sheets 1
Ref. No. A.M. 6 Prog. 2

Program To calculate ½(x - 100) where x = sum of even nos. from 0 to 20 inc.
Section 6.1

Notes	Address	Instruction F	Instruction A	Notes
	0	00	0	Zero } Calculation constants
	1	00	2	}
	2	00	9	}
	3	00	1	}
	4	00	100	Subtractor }
	5	00	0	Next even number
	6	00	0	Sub-total
	7	00	0	Count
	8	00	0	RESULT
ENTRY →	9	01	2	} Set count
	10	02	7	}
20 →	1	01	5	}
	2	03	1	} Form even number
	3	02	5	}
	4	03	6	} Form sub-total
	5	02	6	}
	6	01	7	}
	7	04	3	} Reduce count by one
	8	09	21	} and transfer if negative
	9	02	7	}
	20	08	11	Re-enter loop
	1	01	6	} Subtract 100 from x
	2	04	4	}
	3	12	0	Shift right one place (= x/2)
	4	02	8	Place result in location 8
26 →	5	01	0	} Dynamic stop.
	6	08	25	}
	7			
	8			
	9			

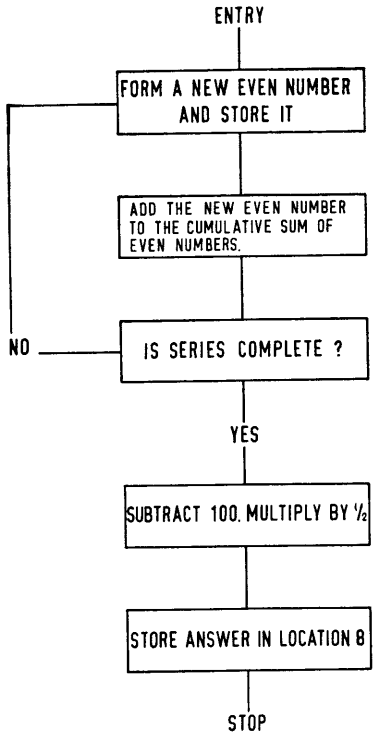


Figure 3 PROGRAM LOOP

The application of functions 09 and 08 should be noted, particularly 08 in address 26. Once the machine reaches this instruction it cannot get out of the small loop in locations 25 and 26 without manual intervention. The instruction in location 25 is a 'dummy' instruction and such a loop is known as a 'dynamic stop'. This is an accepted method of 'stopping' a program. This loop can be broken only by a priority-interrupt.

6.2 Example 2

PROBLEM: Clear the contents of locations 0 to 4075 and give the teleprinter a line-feed signal when complete.

METHOD: The method is shown on the program sheet on the next page.
Particular attention is drawn to the use of

functions 05 (Modify)
and functions 07 (class 2 – input control-digit)
& 06 (class 4 – output to teleprinter)
which use pseudo-addresses.

6.3 Initial Instructions

In section 5, mention was made of the input routine that is frequently 'built' into a machine during manufacture.

Figure 4 shows the flow diagram of a typical set of initial instructions.

The routines incorporated in these particular initial instructions are:

ARCH

program sheet

SHEET No. 1

Date 25th July 1962

No. of Sheets 1

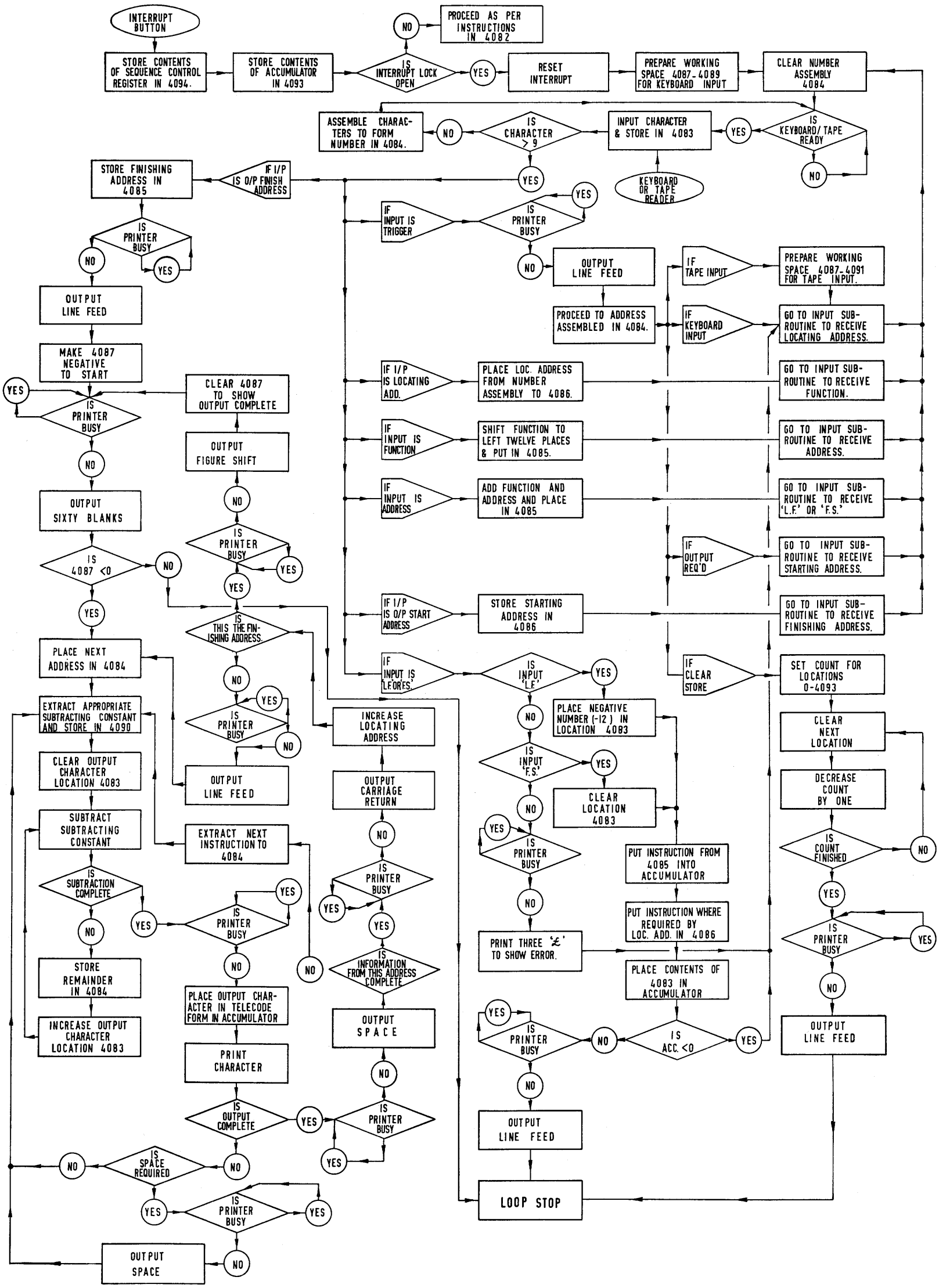
Ref. No. A.M. 6 Prog. 3

Program To clear locations 0 to 4075 and output l.f. to printer when complete

Section 6.2

Notes	Address	Instruction F	Instruction A	Notes
	4070			
	1			
	2			
	3			
	4			
	5			
	4076	00	4075	Count record and modifier
	7	00	0	Zero
	8	00	1	Subtracting constant
	9	00	30	l.f. (line-feed) symbol
ENTRY →	4080	01	4077	Clear the acc.
	1	05	4076	} Modified instruction to clear
	2	02	0	} next storage location
	3	01	4076	} Reduce modifier
	4	04	4078	
	5	09	4088	Test if end and count down
	6	02	4076	} Re-enter loop
	7	08	4080	
↳	8	07	513	} Test if printer is busy
↳	9	09	4087	}
	4090	01	4079	} Output l.f.
	1	06	1025	}
↳	2	01	4077	} Dynamic stop
↳	3	08	4092	}
	4			
	5			
	6			
	7			
	8			
	9			

Figure 4 FLOW DIAGRAM OF INITIAL INSTRUCTIONS



(a) input from paper-tape reader (decimal character)
(b) input from keyboard of teleprinter (decimal character)
(c) output contents of store to teleprinter (alpha/numeric character)
(d) clear store.

The method of entry into one of these routines is as follows.
(1) Operate the priority-interrupt pushbutton (see section 4.4.3) which causes the initial instructions to enter a routine which 'looks' at the teleprinter keyboard.
(2) Type the code for the relative routine on the teleprinter keyboard. Receipt of this code is acknowledged by the output of a line-feed signal on the teleprinter.

The specified routine is then entered.

6.3.1 Input from Paper-Tape Reader Code: 41 *cr* (carriage return)

This routine causes data on punched-paper tape to be read into the machine via the paper-tape reader and stored in specified locations.

Each instruction is preceded by the address in which it is to be stored (its directory). The tape format is: directory (4 characters)
space
instruction function (2 characters)
space
instruction address (4 characters)
carriage return
line feed
directory, etc.

For instance, the tape for example 2 (clear locations 0 to 4075) is as follows:

blanks (about 30 to permit easy loading into the tape reader).

4076	*sp*	00	*sp*	4075	*cr*	*lf*
4077	*sp*	00	*sp*	0000	*cr*	*lf*
4078	*sp*	00	*sp*	0001	*cr*	*lf*
4079	*sp*	00	*sp*	0030	*cr*	*lf*
4080	*sp*	01	*sp*	4077	*cr*	*lf*
4081	*sp*	05	*sp*	4076	*cr*	*lf*
		etc.		etc.		
4092	*sp*	01	*sp*	4077	*cr*	*lf*
4093	*sp*	08	*sp*	4092	*cr*	*fs*

sp stands for space, *cr* for carriage return, *lf* for line feed and *fs* for figure shift

Note that after the last instruction, *fs* is used instead of *lf* so that the input routine can detect the end of the data to be read in. When the *fs* symbol is detected, the machine acknowledges the signal by outputting an *lf* to the printer.

Input can be stopped at any moment by pressing the interrupt button. The input routine includes a check on the characters being read in and causes £ £ £ to be printed out in the event of an error in the format. The machine then continues to read the tape.

6.3.2 Input from Keyboard Code: 51 *cr*

This routine enables instructions to be input via the keyboard provided that the same input format is used as for paper tape.

When an apparent typing error is detected, a series of £ signs is printed in the same way as in the indication of tape-reader input errors.

6.3.3 Output Contents of Store to Teleprinter Code: 175 *cr*

The starting and finishing addresses of the locations to be printed out must be typed on the keyboard with the following format:

Starting address (4 decimal digits),
space
finishing address (4 decimal digits),
space
carriage return

When this is acknowledged with *lf*, 60 blanks are signalled, (in case the teleprinter has a punched-paper-tape attachment) followed by the contents of the locations specified in the same format as for input, including *fs* in place of *lf* after the last instruction. A further 60 blanks complete the output.

6.3.4 Clear Store Code: 232 *cr*

This routine causes the contents of the variable-core-store module to be cleared.

Part 6 Appendix I

BINARY NUMBERS

1.0 BINARY NOTATION

Binary notation is a means of representing all numbers using the two characters **0** and **1** only. The reason for its wide adoption for automatic data-processing is that it is generally easier to construct mechanisms and circuits having only two stable states rather than any larger number.

In order to understand binary notation fully it is of value to consider the fundamental principles of the 'everyday' form of numerical notation, namely, decimal.

In decimal notation, there are ten different characters which, in increasing order of value, are 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9. In other words, 9 has a larger value than 6, 4 has a larger value than 3, etc.

Consider the decimal number 666. Although the same value character is used in all three positions, the significance of each is different since the left-hand six signifies 6 hundreds, the middle six, 6 tens and the right-hand six, 6 units.

Thus 666 is an abbreviated way of indicating

$$6\times100+6\times10+6\times1.$$

The contribution of a particular digit to the total value of a decimal number is determined by the value represented by the character and by its position relative to a decimal point, each of these positions signifying a particular power of ten.

$$\text{i.e.} \quad 666=6\times10^2+6\times10^1+6\times10^0$$

$$\text{and} \quad 6{\cdot}66=6\times10^0+6\times10^{-1}+6\times10^{-2}$$

The binary form of numerical notation uses only the two characters **0** and **1** which have the same value as in decimal notation.

As in decimal notation, the contribution of a particular binary digit to the total value of a binary number is determined by the value represented by the character (**0** or **1**) and its position relative to a point (in this case called the binary point). However, in binary notation, each character position relative to the binary point signifies a particular power of two.

For example, the binary number 10111. represents $1\times2^4+0\times2^3+1\times2^2+1\times2^1+1\times2^0$

$$= \quad 16 + 0 + 4 + 2 + 1$$
$$= \quad 23.$$

Similarly, 1·0111 represents $1\times2^0+0\times2^{-1}+1\times2^{-2}+1\times2^{-3}+1\times2^{-4}$

$$= \quad 1 + 0 + \tfrac{1}{4} + \tfrac{1}{8} + \tfrac{1}{16}$$
$$= \quad 1\tfrac{7}{16}$$

The following serve to illustrate further this principle.

Decimal	Binary
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100

In order to convert from decimal into binary, a set of simple rules must be followed They are:

(a) subtract from the decimal number to be converted the largest power of two which gives a positive answer and record a **1**,
(b) decrease the subtrahend by one power of 2. Determine if this new value is greater than, equal to or less than the remainder of the previous subtraction,
(c) if it is greater, record a **0** and repeat (b); if it is equal or less, subtract it from the remainder, record a **1** and repeat (b).

This process is continued until the subtrahend is 2^0, when the pattern of **1**'s and **0**'s recorded gives the required binary number.

For example: convert 374 to binary

Decimal Number	Subtrahend	Remainder	Binary Number
374	256 (2^8)	118	**1**
	128 (2^7)	(*−ve*)	**0**
	64 (2^6)	54	**1**
	32 (2^5)	22	**1**
	16 (2^4)	6	**1**
	8 (2^3)	(*−ve*)	**0**
	4 (2^2)	2	**1**
	2 (2^1)	0	**1**
	1 (2^0)	(*−ve*)	**0**

Thus, the binary equivalent of 374 is **101110110.**

2.0 BINARY ARITHMETIC

The basic rules of binary arithmetic are in many ways much simpler than decimal as shown in the following tables.

Decimal Addition Table

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18

Decimal Multiplication Table

×	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

Binary Addition Table

+	**0**	**1**
0	**0**	**1**
1	**1**	**0** carry **1**

Binary Multiplication Table

×	**0**	**1**
0	**0**	**0**
1	**0**	**1**

The use of these rules may now be illustrated by the following examples.

Example 1 Binary Addition

101111	.....	32+ 8+4+2+1= 47
1011011	.....	64+16+8+2+1= 91
10001010	.....	128+ 8+2 =138

N.B. (a) **0+1=1**
(b) **1+0=1**
(c) **1+1=0** and **1** to carry
(d) **1+1+1=1** and **1** to carry

Example 2 Binary Subtraction

10011	.....	16+2+1=19
1001	.....	8+1 = 9
1010	.....	8+2 =10

N.B. (a) **0−1=1**+borrow **1**
(b) **1−0=1**
(c) **1−1=0**

These basic principles of binary addition and subtraction are illustrated further by examples of multiplication and division.

Example 3 Binary Multiplication

```
  1001  ..........................   8+1= 9
     x                                  x
  1101  ..........................   8+4+1=13
  1001
 0000
1001
1001
1110101  ........  64+32+16+0+4+0+1=117
```

Example 4 Binary Division

```
      1011  ..............        8+2+1        =11
101 | 110111                4+1=5 | 32+16+4+2+1=55
     101
      111
      101
       101
```

Occasionally, binary notation is referred to as 'pure' binary in order to distinguish it from coded forms of binary. One of these coded forms widely used is *binary-coded decimal.*

3.0 BINARY-CODED DECIMAL

This form of representation is of some advantage for certain applications owing to the fact that it is technically simpler to translate from B.C.D. to decimal (for printing) than from 'pure binary' to decimal.

The principle of B.C.D. is simple; quantities are represented as if they were in pure decimal, i.e. units, tens, hundreds, etc., but each decimal digit is represented by its binary equivalent.

For example, the B.C.D. representation of 374 is:

0011	**0111**	**0100**
Hundreds	Tens	Units
3	7	4

It is apparent that more binary digits are required than if pure binary were used, but translation from B.C.D. to decimal is obviously easier.

Part 6 Appendix II

THE ARCH INSTRUCTION CODE

The contents of the accumulator and location N **after** the instruction has been obeyed are indicated by **a′** and **n′** and the initial contents by a and n. A dash in the **n′** column indicates that a storage location is not involved.

Central Processer Instructions

Function	Operation	Result **a′**	Result **n′**	Time *Read & Obey*
01	Replace c(acc.) by c(store)	n	n	150μs
02	Write c(acc.) into c(store)	a	a	150μs
03	Add c(acc.) and c(store)	$a+n$	n	210μs
04	Subtract c(store) from c(acc.)	$a-n$	n	210μs
05	Set modifier	a	n	150μs
06	Output (see later)			140μs
07	Input (see later)			140μs
08	Unconditional transfer control c(S.C.R.) stored	a	n	140μs
09	Conditional transfer control	a	n	120μs
10	Collate c(acc.) and c(store)	$a\&n$ *collated*	n	180μs
11	Write c(S.C.R.) into c(store)	–	S.C.R.	150μs
12	Shift c(acc.) right one place The sign digit is preserved	$a\rightarrow$ one place	–	140μs
13	Shift c(acc.) left (end around) one place only	a↩ one place	–	140μs
14	Shift c(acc.) right N places. The sign digit is preserved	$a\rightarrow$ N places	–	(N+1)64+100μs
15	Shift c(acc.) left (end around) N places	a↩ N places	–	(N+1)64+100μs
16	Multiply c(acc.) by c(store) Double length product in acc. and AR	$a\times n$ (*in acc.+AR*)	n	1·37 to 2·27 ms
17	Divide c(acc.) and c(AR) by c(store). Quotient placed in acc.	$\frac{a\&c(AR)}{n}$	n	2·32 ms
18	Replace c(AR) by c(acc.)	a	a in AR	160μs
19	Replace c(acc.) by c(AR)	c(AR)	AR unchanged	160μs
20	Place $\sqrt{c}$(acc.) in acc.	$\sqrt{a}$	–	2·5 to 2·83 ms

30μs must be added to the time if the instruction is held in the variable core store and a further 30μs if n is also held in the variable core store.

If any instruction is modified its read-and-obey time is increased by 50μs.

Part 6 Appendix II (cont.)

Function 06 Output

The A digits of the instruction and sometimes the contents of the accumulator further specify the function.

Class	Number	Accumulator	Operation
1	A.D.C. number	Not used	Trigger A.D.C.
2	Control code number	Not used	Priority interrupt and watchdog control
3	Paper-tape punch number	Five least-significant digits specify character	Output on punched paper-tape
4	Teleprinter number	Five least-significant digits specify character (telecode)	Print the telecode character in acc.
5	Digital output number	*c*(acc.) negative=off *c*(acc.) positive=on	External on/off signal
6	Decade number	Five least-significant digits specify decimal digit to be displayed	Output character to visual display
7	Indicator number	*c*(acc.) negative=alarm *c*(acc.) positive=normal	Signal alarm indicator
8	Typewriter number	Five least-significant digits specify the character to be typed (telecode)	Output to electric typewriter
9	Strip-printer number	Five least-significant digits specify the character or control action	Output to strip-printer
10	Link number	*c*(acc.) negative=lower *c*(acc.) positive=raise	Link drive
11	Converter number	*c*(acc.) specifies value to be converted	Output to digital-to-analogue converter
12	Input-group number	12 least-significant digits specify input point number (in B.C.D.)	Initiate selection of analogue input
13	Decade Switch number	Not used	Select decade switch and prepare to input data

Part 6 Appendix II (cont.)

Function 07 Input

Class	Number	Accumulator	Operation
1	A.D.C. number	Most-significant digit positions contain A.D.C. value	Input value of A.D.C.
2	Control digit number	Most-significant digit position contains the control digit	Input a control digit
3	Paper-tape reader number	Five least-significant digits contain input	Input from punched-paper tape reader
4	Keyboard number	Five least-significant digits contain input	Input from keyboard
5	Digital-input number	Least-significant digit positions contain input	Input digital signal
6	Paper-tape reader number	if *c*(acc.) negative=failure *c*(acc.) positive=no failure	Tape input parity check and reset
7	Time digit	Four least-significant digits contain B.C.D. number from clock	Sample digital clock

The time taken to read and obey all input and output instructions is 130 μs. The asynchronous operating time of the various input and output devices does not hold the machine up unless, for example, a quick succession of 'output-to-paper-tape punch' instructions occurs. The machine may then be held up pending clearance of a 'busy' signal – see example in section 6.2.

Part 6 Appendix III

ELLIOTT TELECODE

Binary	*Decimal*	*Tape Punching*	*Character* *Figure Shift*	*Letter Shift*
00000	0	·	bl	bl
00001	1	· o	1	A
00010	2	· o	2	B
00011	3	· oo	*	C
00100	4	·o	4	D
00101	5	·o o	$ or &	E
00110	6	·oo	=	F
00111	7	·ooo	7	G
01000	8	o ·	8	H
01001	9	o · o	'	I
01010	10	o · o	,	J
01011	11	o · oo	+	K
01100	12	o ·o	:	L
01101	13	o ·o o	−	M
01110	14	o ·oo	.	N
01111	15	o ·ooo	%	O
10000	16	o ·	0	P
10001	17	o · o	(	Q
10010	18	o · o	)	R
10011	19	o · oo	3	S
10100	20	o ·o	?	T
10101	21	o ·o o	5	U
10110	22	o ·oo	6	V
10111	23	o ·ooo	/	W
11000	24	oo ·	@	X
11001	25	oo · o	9	Y
11010	26	oo · o	£	Z
11011	27	oo · oo	fs	fs
11100	28	oo ·o	sp	sp
11101	29	oo ·o o	cr	cr
11110	30	oo ·oo	lf	lf
11111	31	oo ·ooo	ls	ls

Part 6 Appendix IV

POWERS OF 2 IN DECIMAL

2^n	n	2^{-n}
2	1	·5
4	2	·25
8	3	·125
16	4	·062 5
32	5	·031 25
64	6	·015 625
128	7	·007 812 5
256	8	·003 906 25
512	9	·001 953 125
1 024	10	·000 976 562 5
2 048	11	·000 488 281 25
4 096	12	·000 244 140 625
8 192	13	·000 122 070 312 5
16 384	14	·000 061 035 156 25
32 768	15	·000 030 517 578 125
65 536	16	·000 015 258 789 062 5
131 072	17	·000 007 629 394 531 25
262 144	18	·000 003 814 697 265 625
524 288	19	·000 001 907 348 632 812 5
1 048 576	20	·000 000 953 674 316 406 25
2 097 152	21	·000 000 476 837 158 203 125
4 194 304	22	·000 000 238 418 579 101 562 5
8 388 608	23	·000 000 119 209 289 550 781 25
16 777 216	24	·000 000 059 604 644 775 390 625
33 554 432	25	·000 000 029 802 322 387 695 313
67 108 864	26	·000 000 014 901 161 193 847 656
134 217 728	27	·000 000 007 450 580 596 923 828
268 435 456	28	·000 000 003 725 290 298 461 914
536 870 912	29	·000 000 001 862 645 149 230 957
1 073 741 824	30	·000 000 000 931 322 574 615 479
2 147 483 648	31	·000 000 000 465 661 287 307 739
4 294 967 296	32	·000 000 000 232 830 643 653 870
8 589 934 592	33	·000 000 000 116 415 321 826 935
17 179 869 184	34	·000 000 000 058 207 660 913 467
34 359 738 368	35	·000 000 000 029 103 830 456 734
68 719 476 736	36	·000 000 000 014 551 915 228 367
137 438 953 472	37	·000 000 000 007 275 957 614 183
274 877 906 944	38	·000 000 000 003 637 978 807 092
549 755 813 888	39	·000 000 000 001 818 989 403 546
1 099 511 627 776	40	·000 000 000 000 909 494 701 773

SOME USEFUL CONSTANTS

π =	3·141 592 653 590	$1/\pi$ =	0·318 309 886 184
$\log_{10}e$ =	0·434 294 481 903	$\log_e 10$ =	2·302 585 092 994
$\log_{10}2$ =	0·301 029 995 664	e =	2·718 281 828 459
$\sqrt{2}$ =	1·414 213 562 373	$\sqrt{3}$ =	1·732 050 807 569
1 radian =	57·295 779 513 082°	1° =	0·017 453 292 520 radian

ARCH MANUAL

7

part 7

APPLICATIONS

reg'd trade mark

EXAMPLE 1

THE EVOLUTION OF A CONTROL HIERARCHY FOR A CONTINUOUS PROCESS

INTRODUCTION

This example illustrates how integrated control over a complete process can be evolved in stages, each stage initially being treated as a separate problem and then subsequently linked into an over-all control system.

THE PROCESS

The imaginary process in this example is a chemical one and is divided into three main process units: blending plant, chemical reactor and purification plant.

A block diagram of this plant is given in figure 1.

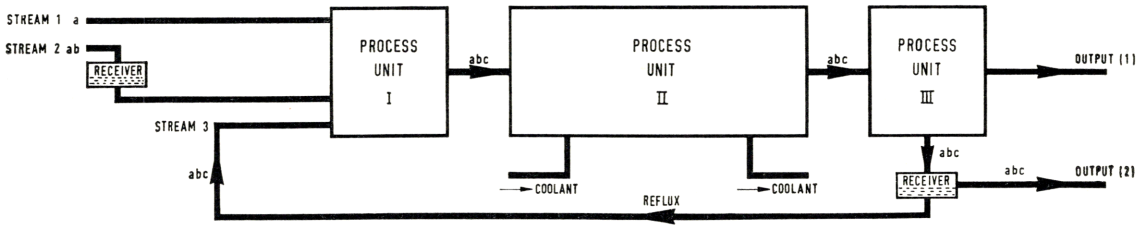


FIGURE 1 BLOCK DIAGRAM OF PLANT

Unit I is a three-stream blender.

Stream no. 1 contains substance a.
Stream no. 2 contains substances a & b.
Stream no. 3 contains substances a, b & c.
(Stream no. 3 is a reflux from unit III.)

The blended product from unit I is the feedstock to the chemical reactor unit II. This unit is designed to enrich the feedstock in substance c at the expense of a and b.

The product of unit II is the feedstock to unit III which refines it so as to produce a high-purity substance c with a controlled amount of impurity due to substance a.

OPERATION

The principles by which the process operates are, of course, known before the process plant is constructed and certain elementary rules of control must be laid down in order to get any output at all. These rules are concerned with the correct and safe operation of the process and involve knowledge of the values of various parameters such as temperature, flow, etc. and the ability to set these to specified values. The first stage of control is thus to install indicating and control instruments and any other equipment necessary to provide the operators with information and a certain amount of control.

FIRST STAGE OF CONTROL

In the first instance, the process may be controlled using entirely conventional instrumentation.

This is shown in figure 2. (Page 7:5)

This first stage of control represents the state of affairs in many process plants today. The product is produced more or less on specification and with varying efficiency (i.e. at varying cost of production). There are often two or three grades of product (output 1) all of which can be sold, although they may fetch different prices. If the quality of the output (1) should drop below the minimum it can always be re-processed. The exact rules of control used by the operators of each unit are ill defined, there is a lack of information from the unit being controlled and there is probably no information at all from neighbouring process units. This may lead to wide variations of efficiency on different shifts (i.e. with different operators). There may even be 'unexplained' transients in some units just after a change of shifts as the new operator adjusts the controls to his favourite settings. Even good operators can produce poor results on occasions, due to lack of concentration, caused perhaps by a row with the wife or indigestion.

There is no unique path to improved control. It depends on a large number of circumstances, e.g. availability of instruments, layout of the process plant, the personality of particular unit managers, etc. The important thing is to make a start somewhere, because only by trying can results be obtained and the way found to the next stage.

SECOND STAGE OF CONTROL

Suppose that it is decided to attempt to improve the control of process-unit I (the blending area).

Since stream 2 has a fairly large reservoir its composition is unlikely to change rapidly and it is decided to continue the analysis for %b by laboratory sample. %b and %c in the reflux are liable to change more rapidly however and an on-stream analyser is installed to measure these percentages.

From the measurements *already being made in this area* it is now possible to install an automatic multi-variable controller C1 to control the ratio a/b to the desired value.

This controller is shown in figure 3. (Page 7:6)

The rules of control for this process unit are fairly simple. a/b is the required ratio of mass flows of substances a and b, and this must be equal to the total flow of substance a in streams 1, 2 and 3, divided by the total flow of substance b in streams 2 and 3.

The problem can be stated as follows:

$$\frac{a}{b}=\frac{K_1F_1+K_2F_2\left[1-\frac{b}{100}\%\right]+K_3F_3\left[1-\frac{b_r}{100}\%-\frac{c_r}{100}\%\right]}{K_2F_2\frac{b}{100}\%+K_3F_3\frac{b_r}{100}\%}$$

Where a/b is the desired ratio of substances a and b

F_1 is the flow in stream 1
F_2 is the flow in stream 2
F_3 is the flow in stream 3
b is the percentage of substance b in stream 2
b_r is the percentage of substance b in stream 3
c_r is the percentage of substance c in stream 3
K_1, K_2 and K_3 are constants relating to the densities of the substances.

The controller must therefore solve this equation to yield the flow F_1 which is necessary to maintain the ratio a/b at the desired value. Since this is a comparatively simple calculation it is decided to solve it by analogue means.

As the first step towards the production of a computer schematic, the equation is re-arranged to yield:

$$\therefore \quad F_1=\frac{K_2}{K_1}F_2\left[1-b\left(1+\frac{a}{b}\right)\right]+\frac{K_3}{K_1}F_3\left[1-b_r\left(1+\frac{a}{b}\right)-c_r\right]$$

all variables being in percentages of full scale.

The computer schematic for controller C1 is shown in figure 4. (Page 7:6)

The first result to be expected from this second stage of improved control is a better and more consistent conversion rate (a+b→c) in the unit II reactor, because it is now running more steadily at its designed value for a/b.

Secondly, it leads naturally to the evolution of the next (third) stage of control since the information about the mass flow rates of substances b and c can be fed forward and displayed to the operators of unit II.

From figure 2 it can be seen that unit II is run by maintaining the temperature gradient T_1, T_2, T_3 at a specific 'best' value. The value is a best average figure. In actual fact the best temperature gradient depends on the particular value of b and c flowing at the time. Thus it is now possible to try to obtain empirically the relationship between T_1, T_2, T_3 and b and c. This may be done by data-logging the readings of the various variables and correlating them with special laboratory analyses of samples of the output from the reactor. The data-logging may be done either over any extended period of normal running of the reactor, or during a special series of tests. Eventually it may be possible to obtain a set of best temperature gradients (T_1, T_2, T_3) for each value of b and c as they vary independently in steps.

THIRD STAGE OF CONTROL

Figure 5 (Page 7:7) shows the third stage in the programme for improved control. An extra controller has been added in the area of process-unit II.

Inputs to this controller are the 'feed-forward' information of the mass-flow rates of substances b and c and the measurements of the coolant temperature from TI1 and TI2. Using this information the controller adjusts the set-values of the temperature controllers TIC1, TIC2 and TIC3 via Link mechanisms to maintain the temperature gradient at the best value within the limits of safe coolant temperature rise.

Unit II is the key unit in the process, because it is here that product c is actually manufactured. Its controller C2 may therefore be required eventually to provide supervision of the controllers in the other units, probably according to complex rules. The relationship between the mass flows of substances b and c and the best values of the temperature gradient in the reactor may also be found to be complex. For these reasons, a digital controller is chosen for C2.

The rules of control are not known in complete detail but sufficient is known to make a worthwhile start to control by using the empirical relationships:

$$T_1=T_1'\,(1+e)$$
$$T_2=T_2'\,(1+e)$$
$$T_3=T_3'\,(1+e)$$

T_1', T_2', T_3'=f(b, c) where function f is determined empirically.

$e=A.T.+B.\dfrac{dT}{dt}$ where A & B are empirical constants.

$T=T_{max}-T_R$ where T_{max} is the maximum safe coolant temperature rise (as set) and T_R is the actual measured rise (T_5-T_4).

The flow diagram of the initial program used to carry out the necessary calculations is shown in figure 7 (Page 7:8) and the block diagram of the ARCH modules necessary to execute this program is shown in figure 6. (Page 7:7)

The third stage of improved control should lead to further increase in conversion rate ($a+b \rightarrow c$) because the temperature gradient is better than the average figure used previously. It is also possible to run the reactor closer to the maximum safe temperature because more stable control conditions enable the safety-margin to be reduced. This also improves efficiency.

THE FOURTH STAGE OF CONTROL

In the fourth stage an on-stream analyser is installed on the output stream and the '%a' signal so obtained is used to control the set-value T_6 of the temperature controller TIC4. A simple cascade controller C3, is used in which the measured value of %a is compared with the set-value of the temperature.

This controller is shown in figure 8. (Page 7:9)

This stage should lead to improved performance due to tighter control on the exact specification of the product.

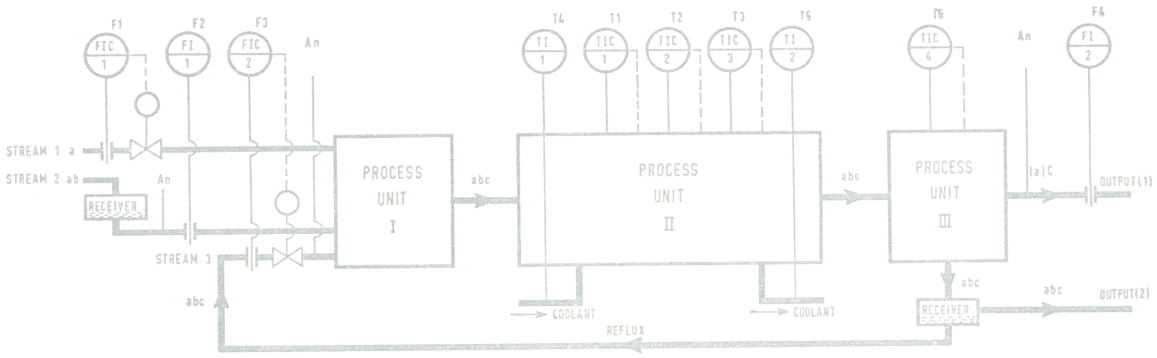
THE FIFTH STAGE OF CONTROL

In stage five, it may be possible to expand the program in the digital controller C2 to give some 'feedback' to the set-value of $C1\left(\frac{a}{b}\right)$ This program can also be extended to produce analysis figures such as plant conversion efficiency and to provide a smooth economical change-over from one product specification to the next. Extra storage, an output printer and, if necessary, extra functions, and more complex arithmetic facilities may be added as required.

CONCLUSION

It has been shown how an integrated system for a process plant may be evolved using ARCH. Each stage in the evolution requires a relatively small capital outlay, which may be justified by successful results, before proceeding to the next stage. The control hierarchy advocated has the additional advantage that a failure of one section does not lead to a failure of all.

EXAMPLE OF THE EVOLUTION OF A CONTROL HIERARCHY



In unit I, the main requirement is to keep the ratio $\frac{\%a}{\%b}$ of the blended product at the value required for unit II. At the same time, it is desired to use as much of stream 2 as is available, it being a relatively low-cost substance, and it is also necessary to use all the reflux (stream 3) from unit III.

Control over the ratio is therefore achieved by adjusting the flow of stream 1 by means of a flow-controller FIC.1. The set-value of this controller is adjusted manually by the operator as a result of calculations involving the %b and flow of stream 2 and the %b and %c and flow of stream 3. The percentage figures are obtained from regular laboratory tests on samples of the substances and the flow rates from the flow-rate indicating instrument FI.1 and controller FIC.2.

The chemical reaction in unit II produces maximum useful output when the reactor is run at a specific temperature gradient. Three temperature indicator/controllers TIC.1, TIC.2 and TIC.3 are used to indicate the temperature-gradient to the operator. He attempts to keep it at the correct value, at the same time preventing the temperature-rise in the coolant (indicated by the relative readings on TI.1 and TI.2) from exceeding the limits of safety for the reactor.

Unit III has only one controlled parameter, namely temperature. This is indicated and controlled by TIC.4. The operator receives an occasional report of the final product analysis and if a change in the % is required he adjusts the set value of this controller. This may lead to temporary upsets in the reflux.

The product flow rate is indicated by FI.2.

FIGURE 2 FIRST STAGE OF CONTROL—CONVENTIONAL INSTRUMENTATION

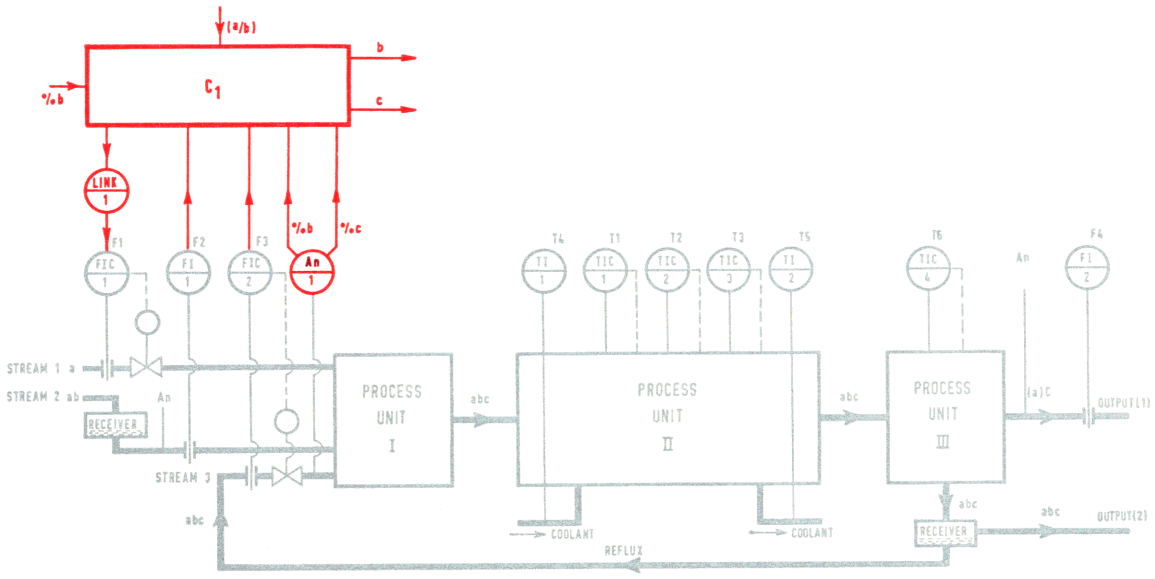
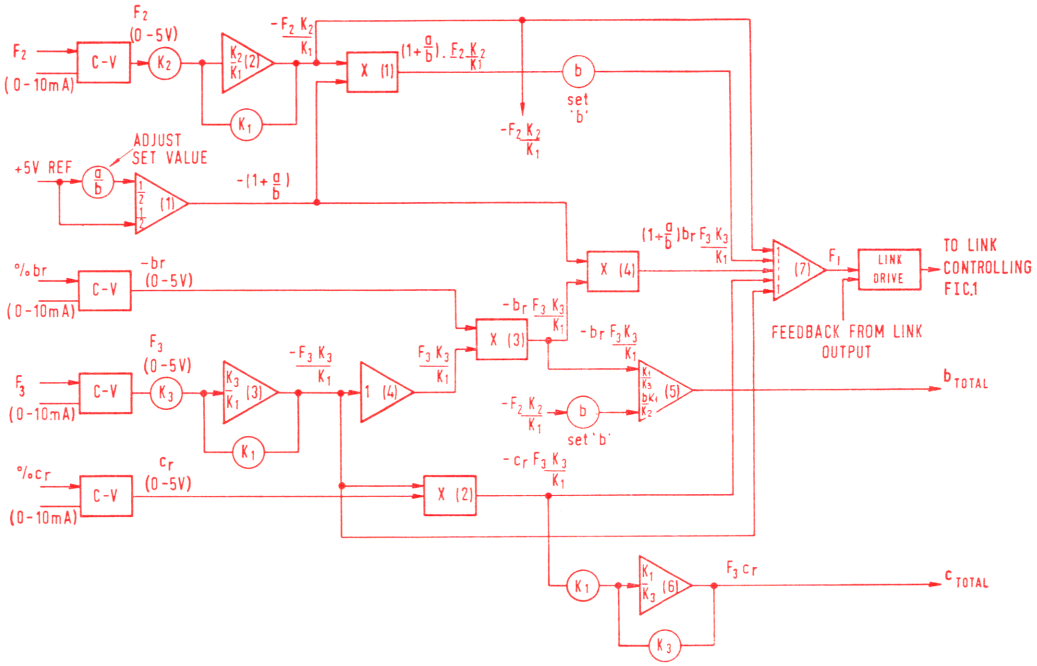


FIGURE 3 SECOND STAGE OF CONTROL—ANALOGUE CONTROLLER C1 FOR UNIT I



Except for %b, the inputs to C1 are taken from the electrical signals already available in the installed instrumentation. %b is inserted by hand as a result of the laboratory analysis. The output of the controller feeds a Link mechanism, the output of which specifies the set-value of the controller FIC.1.

F_2, F_3, %cr and %br are 0-10 mA signals from the measuring instruments and are converted to 0-5V signals (see part 5). The set-value a/b is derived from a coefficient potentiometer connected to a 5V reference module.

FIGURE 4 COMPUTER SCHEMATIC FOR C1

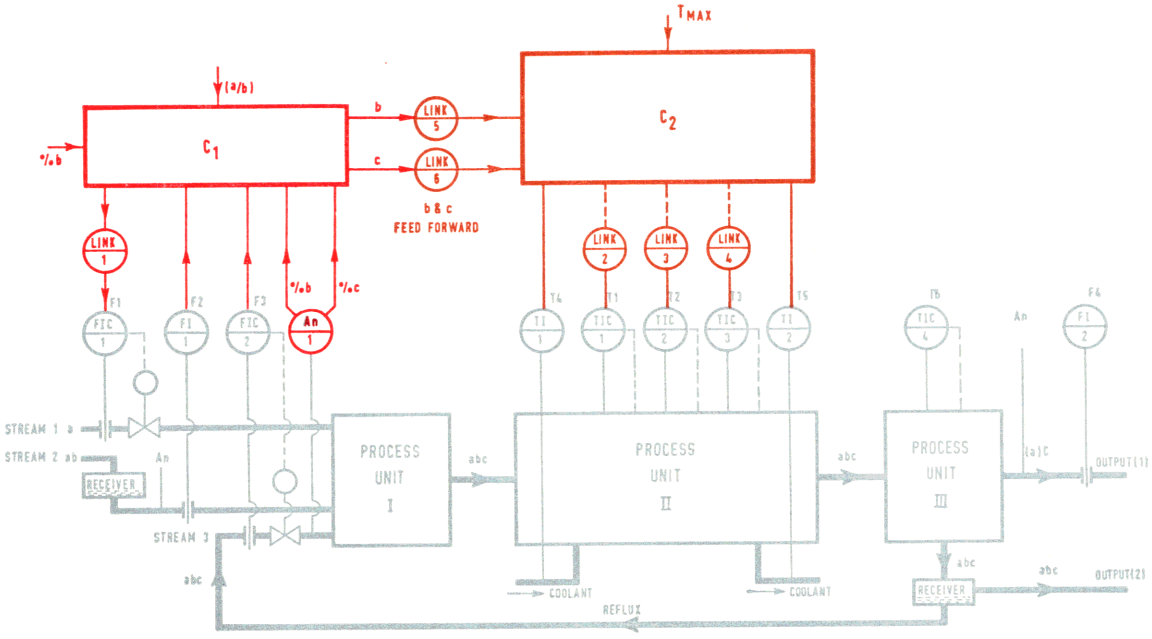


FIGURE 5 THIRD STAGE OF CONTROL—DIGITAL CONTROLLER C2 FOR UNIT II

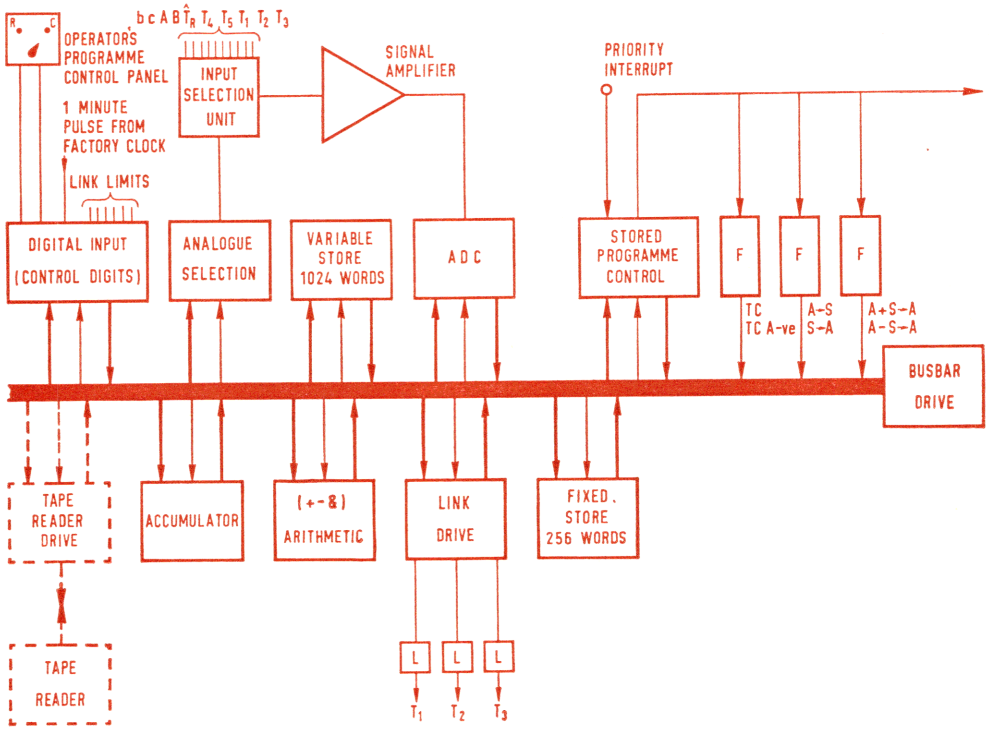
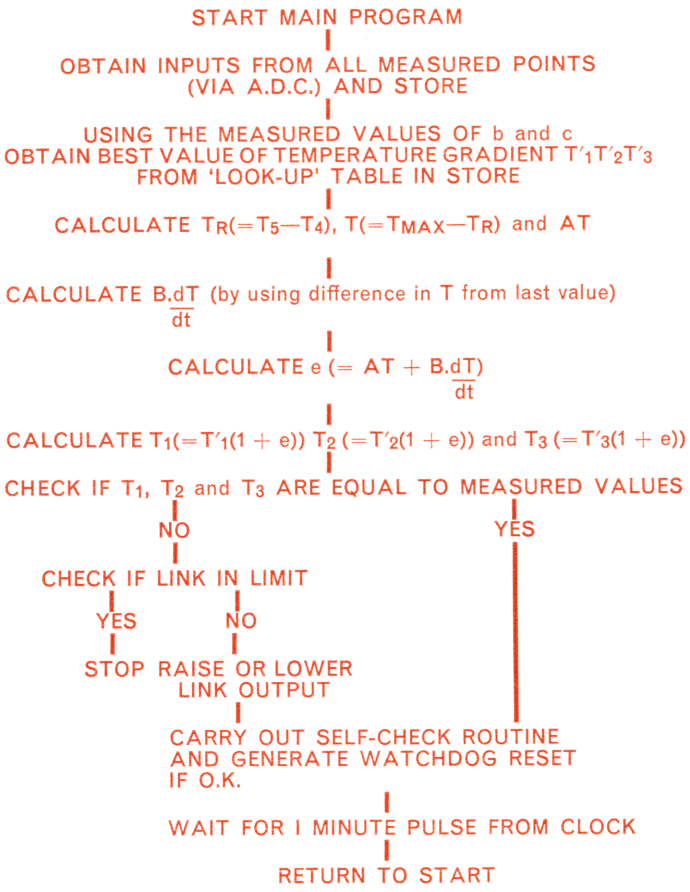
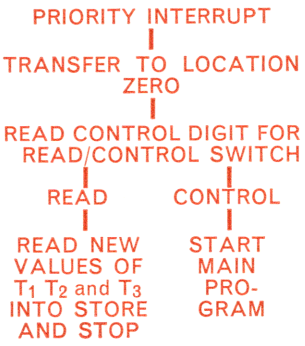


FIGURE 6 BLOCK DIAGRAM OF C2

EXAMPLE OF THE EVOLUTION OF A CONTROL HIERARCHY





The whole routine takes less than 2½ sec. but there is no point in allowing it to be repeated more often than once a minute since the controlled variables must be given time to settle down.

The program operates on the cycle described above unless any of the Link mechanisms steps on to a limit or unless manual intervention is required.

If one of the Link mechanisms reaches a limit stop (indicated by a control digit) the program stops and the watchdog gives an alarm to the operator. The operator must then rectify the condition and then cause the program to start again by pressing the priority-interrupt button.

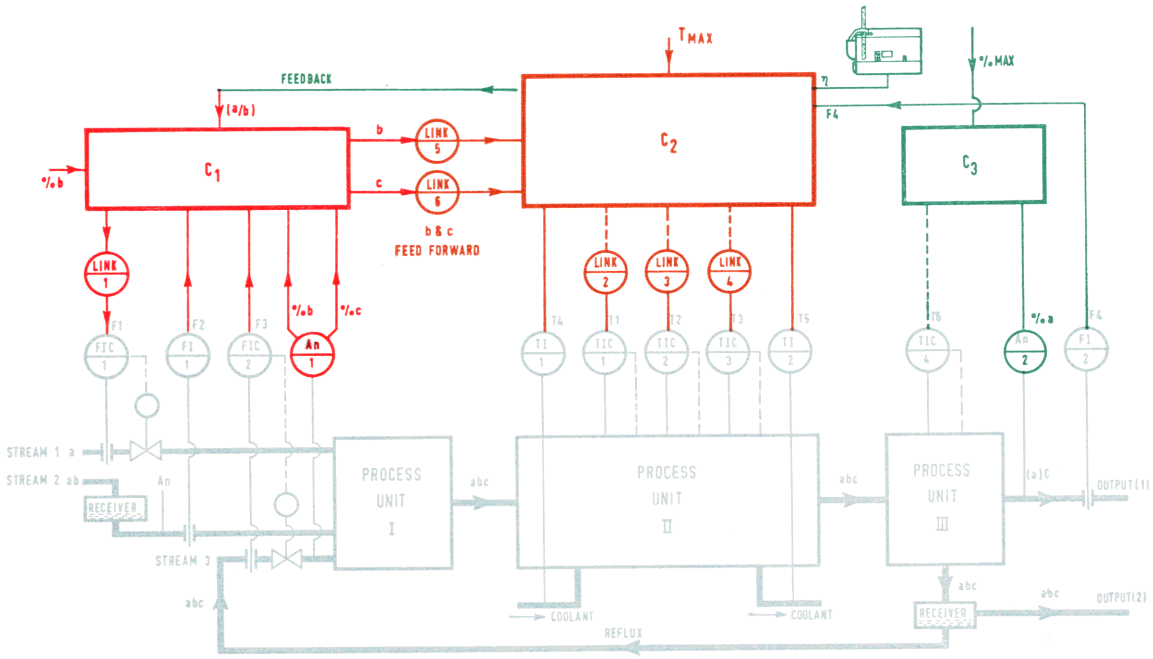
The priority-interrupt button is also used for manual intervention of the program. For instance it may be required to read a new look-up table into the store. This is achieved in this program by means of a 'Read'/'Control' switch feeding one of the control digits. With this switch in the 'Control' position, priority-interrupt causes the main program cycle to be entered but in the 'Read' position it causes a program sub routine to be entered which reads the new table into the store via a tape reader.

The complete program instructions can be held in a 256-word fixed store with a few locations of variable store for working space, i.e. to hold sub-totals of calculations etc. The 'look-up' table is held in several hundred locations in the variable store.

Note, it is possible to put the 'look-up' table in an enlarged fixed store thereby considerably reducing the size of the variable store and eliminating the tape reader. This would have the disadvantage that the look-up table could not be changed without switching the machine off for a period and would also make the development of future programs more difficult.

FIGURE 7 PROGRAM FLOW DIAGRAM FOR C2

EXAMPLE OF THE EVOLUTION OF A CONTROL HIERARCHY



A third controller C3 is added to provide automatic control over the temperature in Unit III. The % a in the output is fed into this controller from an on-stream analyser AN.2 and the specification for the maximum percentage of substance a in the output is fed in manually.

Further modules are added to the digital controller C2 so that the analysis and efficiency figures etc. produced by extending the computer program may be made available to the plant operator. In this case, a paper-tape output channel is shown. The extended program also makes possible automatic feedback of the set-value $\frac{a}{b}$ for controller C1.

More information channels may be provided by extending the number of modules in C1 or C2 or extending the program of C2, as the control system develops. More feedback or feed-forward information from other related processes may also be used.

FIGURE 8 FOURTH AND FIFTH STAGES OF CONTROL

ARCH MANUAL

8

part 8

MODULE DATA SHEETS

ARCH

reg'd trade mark

ARCH digital modules are divided into two groups, namely, operational modules and control modules.

The operational modules are those which, on receipt of the appropriate control signals, perform a specific operation of input, storage, arithmetic or output. The control modules are those which control the flow of data between the operational modules in order to carry out the instructions laid down in the program.

All the data inputs and outputs of the operational modules are designed to be connected to a set of busbars, via which all data transfers take place. The other connections to the operational modules are the signals from the control modules and, in the case of input/output modules, the connections to the external devices. In general all the data inputs/outputs and control signals are listed in full on these sheets with a brief explanation of the operation of each.

The control modules consist of the busbar-drive module which provides the 'drive' to the data busbars, the program-control modules which control the carrying out of the program and the function modules, each of which, on demand from the program control module, generates a particular series of control signals to carry out a specific program function.

The choice and number of modules used for a particular application depends entirely on the nature of the application and modules may be added or removed at will as requirements change.

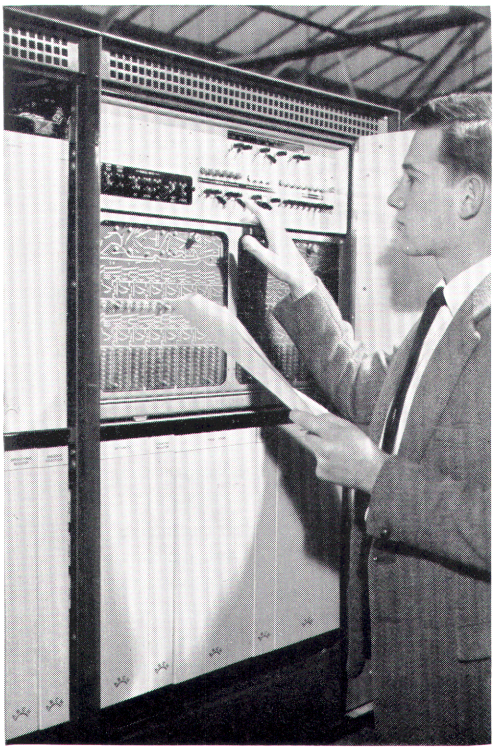
INDEX OF SHEET NUMBERS

For further information please refer to other sheets or contact

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group



Operation of an ARCH digital computer from the supervisory panel. Computer assembled from digital modules mounted in an ARCH double-bay cabinet. Cover removed from core store module below panel.

OPERATIONAL MODULE

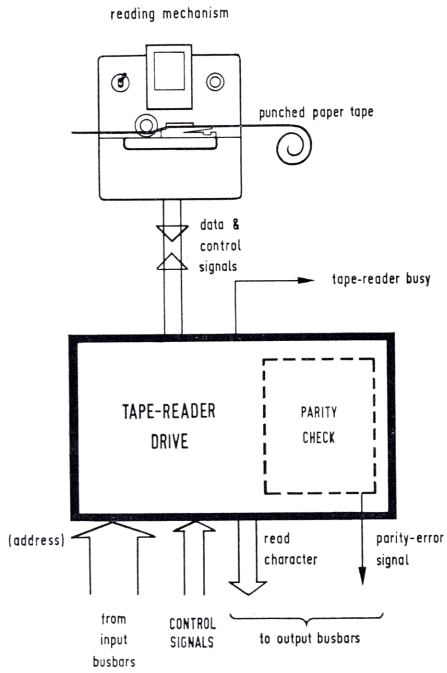
PAPER - TAPE INPUT

SIZE : see table

CATALOGUE No. see table

An input module consisting of an Elliott paper-tape reading mechanism with suitable electronic drive.

This module is used mainly during the comission-ing stages of an ARCH digital computing controller to read instructions and data into a variable-store module. (After commissioning, the tested pro-gram may be transferred to a fixed-store module.)



On receipt of the relative control signals, the 5-bit character at the reading station is gated in parallel out to the data output and the tape is automatically advanced to the next position.

AUTOMATIC PARITY CHECK : A version of this module containing an automatic odd-parity and out-of-scale check on the 5-bit char-acter is also available for use with numerical data. When even-parity is detected i.e. when there is an even number of 1's (holes) in a character or when the number is out-of-scale i.e. 10–15, a parity-error signal is stored. This parity-error signal is treated as a separate input channel having its own address (see control signals).
In this version only the four 'data' bits of the read character are gated out and a quick check may be made on a whole message by reading in the stored parity-error input when reading the tape is com-plete.

CONNECTIONS

DATA INPUT : Address from input busbars
(1) Paper-tape = address class 3
(2) Parity = address class 6

DATA OUTPUTS : (1) 5-bit read character (connected to least-significant five output busbars).
(2) 1-bit parity-error digit (1 = error) connected to most-significant output busbar.

CONTROL SIGNALS : DE-SELECT I.D. : Resets the input-device address memory prior to the selection of a new input.

SELECT I.D. : Gates in the address from the data input to set the memory associated with the appropriate input device.

G.O. – IP.R. : (1) Gates 5-bit character out to data output, and initiates advance of tape to next position.
or (2) Gates one-bit parity-error out to data output.

OTHER CONNECTIONS

Clutch/brake control and data signals to and from paper-tape reading mechanism.
TAPE-READER BUSY: A signal which indicates the period during which the tape is not available for reading, i.e. is moving between characters.

	CATALOGUE No.	SIZE
BASIC MODULE :	D24196	1W
MODULE WITH PARITY CHECK :	D24197	2W

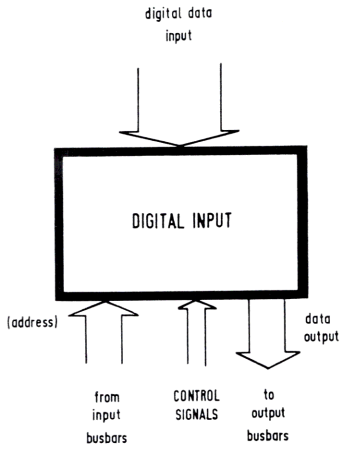
OPERATIONAL MODULE

DIGITAL INPUT (DATA)

SIZE : 2W
CATALOGUE No. D24195

An input module enabling up to 24 bits of digital data to be gated into an ARCH digital controller.

This module is used to gate in digital inputs from keyboards, control desk switches, digital transducers, shaft encoders, etc.



The module consists of 24 gates arranged in four groups of six bits, each group having a separate address.

CONNECTIONS

DATA INPUT : Address from input busbars (class 5).

DATA OUTPUT : Selected six bits (connected to least-significant six output busbars).

CONTROL SIGNALS :
- DE-SELECT I.D. : Resets the input-device address memory prior to the selection of a new input.
- SELECT I.D. : Gates in the new address from the data input to set the memory associated with the appropriate input gate.
- G.O. – IP.R. : Gates the specified six bits out to the data output.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

OPERATIONAL MODULE

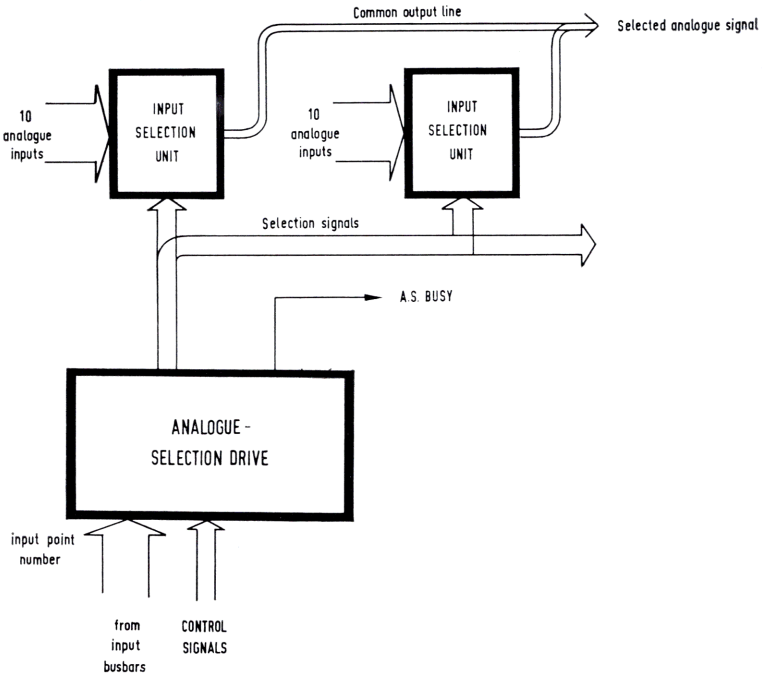
ANALOGUE SELECTION

An input module enabling one specified analogue signal to be selected out of a number of analogue inputs.

SIZE : see table

CATALOGUE No. see table

This module is used to enable inputs from various points in a plant to be 'scanned' in any required sequence so that the analogue signals can be sequentially amplified (if necessary) and converted to digital form in an A.D.C. for subsequent input into an ARCH digital computing controller.



Each analogue input signal is connected to the contact of a two-pole sealed mercury-wetted relay in an input-selection unit. The signal from the required input point (specified by the number gated into a point-number register from the data input) is selected by energising its associated relay via a selection matrix. This connects the signal onto a common output line. Provision is made in the input-selection units for the inclusion of networks such as filters, level-correction networks, etc.

Two drive modules are provided, one of which gives selection for up to 10 inputs, the other for up to 100 inputs (see table).

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England

Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

CONNECTIONS

DATA INPUT : (1) Address from input busbars (class 1).
(2) 12-bit (=3 binary-coded decimal digits) point-number from 12 least-significant input busbars.

CONTROL SIGNALS :

DE-SELECT I.D. :	Resets the input-device address memory prior to the selection of a new input.
SELECT I.D. :	Gates in the address from the data input to set the memory associated with the appropriate input device.
R. – OP.R. :	Resets the point-number register to zero.
G.I. – OP.R. :	Gates the 3 binary-coded decimal digit number from data input into the point-number register and initiates the selection of this point.

OTHER CONNECTIONS

SELECTION SIGNALS : 10 lines+1 line per input-selection unit.

A.S. BUSY : Indicates when selection is taking place.

MODULE	CATALOGUE No.	SIZE
ANALOGUE SELECTION 0 – 10 Way*	D24190	3W
ANALOGUE SELECTION 0 – 100 Way*	D24191	4W
INPUT-SELECTION UNITS 10 WAY†	A24004/3	1W

*Not including INPUT-SELECTION UNITS
†This is a sub-module

OPERATIONAL MODULE

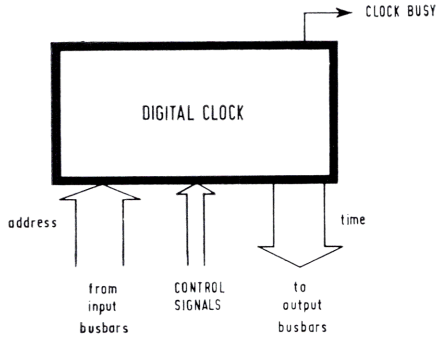
CLOCK

SIZE : see table

CATALOGUE No. see table

A module giving real-time output on a 24-hour clock basis.

The clock output may be used to provide program synchronisation and time indication on visual display, output printer or typewriter.



The module consists of a counting register having four binary-coded decimal digit outputs corresponding to tens of hours, hours, tens of minutes and minutes respectively. Various types are available operating from timed pulses, power supply frequencies or an internal crystal-controlled oscillator (see table below).

CONNECTIONS

DATA INPUT : Address from input busbars (class 7).

DATA OUTPUT : 4 bit (=1 binary-coded decimal digit of time).

CONTROL SIGNALS :

DE-SELECT I.D. : Resets the input-device address memory prior to the selection of a new input.

SELECT I.D. : Gates in the new address from the data input to set the memory associated with the appropriate input device.

G.O. – IP.R. : Gates the specified binary-coded decimal digit of the time out to the output busbars.

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England

Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

OTHER CONNECTIONS

CLOCK BUSY : Indicates the period during which the clock is changing from one value to the next.

TYPE	TIMING SIGNAL	CAT. No.
TIMED PULSE	Switching to OV. every 15 secs., 30 secs. or 1 min. (Minimum pulse width: 40 ms.)	D24203
SUPPLY FREQUENCY	6.3V. 50 c/s	D24202
CRYSTAL CONTROLLED	Internal Crystal Oscillator	D24250

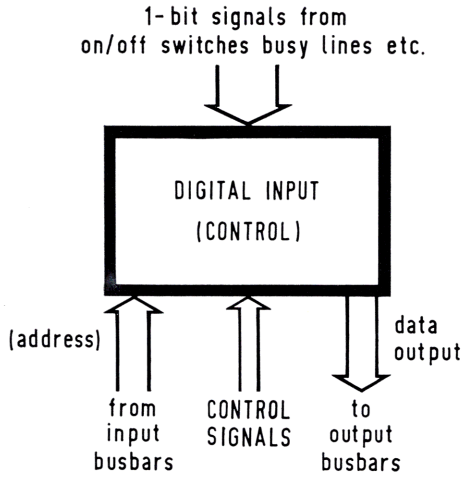
OPERATIONAL MODULE

DIGITAL INPUT (CONTROL)

SIZE : 2W
CATALOGUE No. D25383

An input module enabling up to 16 individual control digits to be gated into an ARCH digital controller.

Control digits are used to indicate the on/off state of 'busy' signals from input/output modules, control panel on/off switches, alarm or limit contacts etc.



The module consists of 16 gates, each gate having a separate address. The selected digit position is fed to the sign (most-significant) digit position on the output busbars so that a conditional transfer instruction can be used immediately after the input of the control digit.

CONNECTIONS

DATA INPUT : (1) Instruction address from input busbars (class 2).

DATA OUTPUT : Selected control digit (connected to most significant output busbars).

CONTROL SIGNALS :

DE-SELECT I.D. : Resets the input-device address memory prior to the selection of a new input.

SELECT I.D. : Gates in the new address from the data input to set the channel associated with the appropriate input gate.

G.O. – IP.R : Gates the specified control digit out to the data output.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

OPERATIONAL MODULE

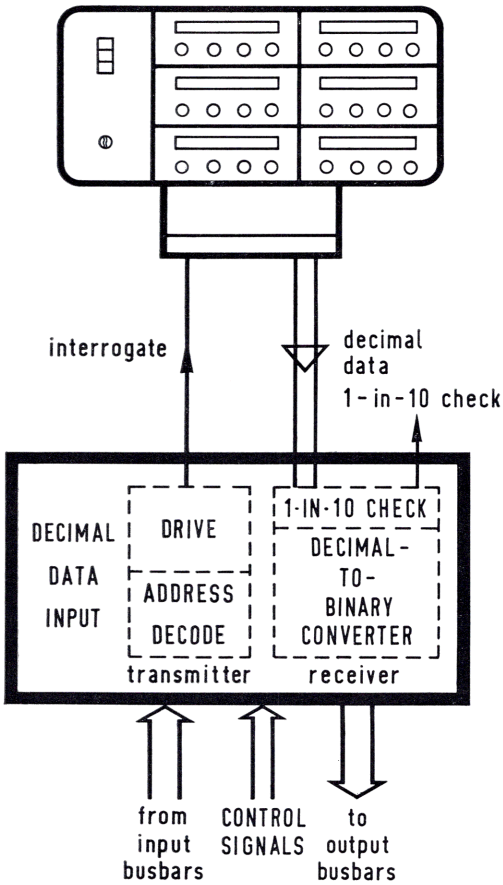
DECIMAL DATA INPUT

SIZE : see table

CATALOGUE No. see table

Input modules which select and read the setting of remote decade switches, decade counter outputs etc.

The modules are primarily intended for use with Arch Industrial Manual Input Consoles (see sheet 5003).



The modules are divided into two sections: transmitter and receiver. The transmitter selects and interrogates the specified decade switch, the receiver checks and converts into binary the received data.

Selection and interrogation of the specified decade switch, results from an output instruction. Input of the binary data is by input instruction. A 1-in-10 check on the received data ensures that only one decimal line is energised at any particular time. This 1-in-10 signal can be used as a control digit.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

CONNECTIONS

DATA INPUT : (1) Instruction address from input busbars (class 15).
(2) Decade switch number.

DATA OUTPUT : Binary equivalent of decade switch setting (connected to the least significant four output busbars).

CONTROL SIGNALS (TRANSMITTER) :
- DE-SELECT O.D. : } as for all output modules.
- SELECT O.D. : }
- R – OP.R : resets the decade switch number register to zero.
- G.I. – OP.R : gates the decade switch number from the data input into a register to generate the appropriate INTERROGATE signal.

(RECEIVER) :
- DE-SELECT I.D. : } as for other input modules.
- SELECT I.D. : }
- G.O. – IP.R : Gates binary equivalent of selected decade switch setting out to data output.

The switch specified in the output instruction remains selected until another output instruction is given. The distance between the drive module and the switches can be up to 1 mile.

The basic module caters for up to 24 switches but is extendable by the addition of extra transmitter stages.

OPERATIONAL MODULE

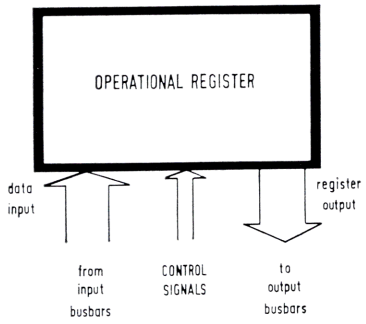
OPERATIONAL REGISTER

SIZE : 2W

CATALOGUE No. see table

A register store with parallel gate-in and gate-out facilities.

Chief applications of this module are as the 'accumulator' of an ARCH computer and as the means of obtaining data shift (see table).



The data input is gated into the register where it remains stored until a reset is applied. Gating out does not destroy the stored data.

In the data-shift operational registers, the gated-out data is shifted with respect to the data input.

CONNECTIONS

DATA INPUT : 18 bits from input busbars.

DATA OUTPUT : 18 bits to output busbars.

CONTROL SIGNALS :

G.I. :	Gates the parallel data input into the register.
G.O. :	Gates the register content out to the data output.
R. :	Resets the register content to zero.

OTHER CONNECTIONS

Direct outputs from sign digit of accumulator for transfer-control function.

MODULE	CATALOGUE No.
Accumulator	D24177
Left shift (one place)	D25098
Right shift (one place)	D25100

OPERATIONAL MODULE

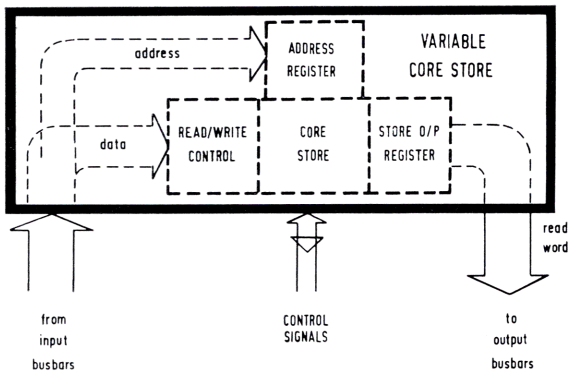
VARIABLE STORE

SIZE : 15W

CATALOGUE No. see table

A core store, the content of which can be varied under program control.

This module is used to store changeable data, i.e. experimental programs, variable plant data, etc., which may subsequently be transferred to a fixed store.



The word-length of the store is 18 bits and words are written into and read out of the store in parallel.

Once the read or write operation has been initiated by the appropriate control signal, the store enters a self-sequencing read/write cycle during which time a busy signal is provided.

CONNECTIONS

DATA INPUT : (1) Address from instruction register (via input busbars).
(2) 18-bits from input busbars.

DATA OUTPUT : 18 bits to output busbars.

CONTROL SIGNALS :

R. – V.S.A.R. : Resets the address register to zero.

G.I. – V.S.A.R. : Gates the new address from the data input into the address register.

STORE WRITE : Causes the data input to be written into the storage location specified by the address.

STORE READ : Causes the content of the storage location specified by the address to be read into the output register and thence to the data output.

G.O. – V.S.R. : Gates the content of the store output register out to the data output.
N.B. It is not necessary to use this

signal during the 'read' operation but it may be used to extend the time for which the store register content appears at the data output for monitoring purposes.

OTHER CONNECTIONS

STORE BUSY : Fed to the programme-control module to indicate the period during which the store is not available, i.e. reading or writing in progress.

CAPACITY	CATALOGUE NUMBER
4096 words (18-bit)	D24161
1024 words (18-bit)	D24172
256 words (18-bit)	D24163
64 words (18-bit)	D24164

OPERATIONAL MODULE

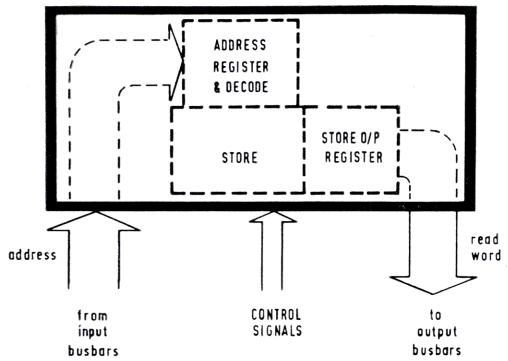
FIXED STORE

SIZE : see table

CATALOGUE No. see table

A permanent (wired) store in which the stored data or instructions are fixed and are not destroyed by read-out or by switching off.

This store is used to contain invariable data and fixed programs.



The store word-length is 18 bits and words are read out in parallel to the data output. Several different versions of this store, containing 'initial instruction' programs, etc are available.

CONNECTIONS

DATA INPUT : Address from instruction register (via input busbars).

DATA OUTPUT : 18 bits to output busbars.

CONTROL SIGNALS :

R. – F.S.A.R. : Resets the address register to zero.

G.I. – F.S.A.R. : Gates the new address from the data input into the address register.

STORE READ : Causes the content of the storage location specified by the address to be read into the store-output register and thence to the data output.

G.O. – F.S.R. : Gates the content of the store output register out to the data output. It is not necessary to use this signal during the 'read' operation but it may be used to extend the time for which the store register content appears at the data output (for monitoring purposes).

CAPACITY	CATALOGUE NUMBER	SIZE
64 words (18-bit)	D26631	7W

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England

Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

OPERATIONAL MODULE

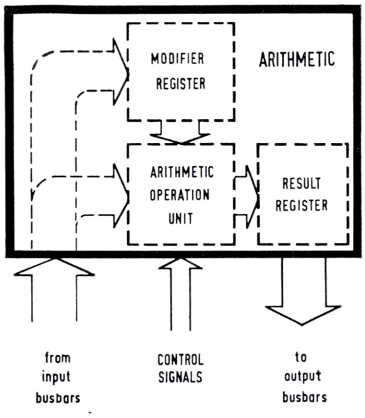
ARITHMETIC

SIZE : 8W

CATALOGUE No. D24172

The basic ARCH arithmetic module providing addition, subtraction and collation facilities.

This module enables simple arithmetic operations to be performed by an ARCH computing controller. More comprehensive arithmetic functions are obtained by using this module in conjunction with the auxiliary arithmetic modules (see other data sheets).



One of the numbers in the calculation (Y) is gated into the modifier register and the calculation is then performed on this number and the number subsequently appearing on the input busbars (X). The result of the calculation is stored in a result register prior to being gated out.

CONNECTIONS

DATA INPUT : 18-bit numbers from input busbars.

DATA OUTPUT : 18-bit result to output busbars.

CONTROL SIGNALS :

A/S and A.C.1. : Two control signals which, in combination, specify the arithmetic operation to be performed.

A/S	A.C.1.	OPERATION
0	1	X+Y (Add)
1	1	X−Y (Subtract)
1	0	X&Y (Collate)

R. – M.R. : Resets the modifier register.

R. – RR. : Resets the result register.

G.I. – M.R. : Gates the first number (Y) into the modifier register from the input busbars.

G.I. – RR. : Gates the result of the calculation into the result register.

G.O. – RR. : Gates the result-register content out to the data output.

OPERATIONAL MODULE

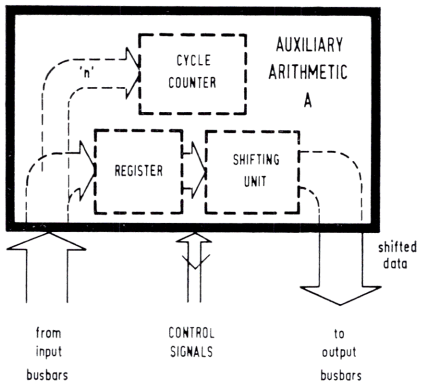
AUXILIARY ARITHMETIC A

SIZE : 6W

CATALOGUE No. D24174

An auxiliary to the arithmetic module providing the additional facility of multiple shift of data in either direction.

This module is used in an ARCH computing controller having a shift-function module and enables data stored in a register (the accumulator) to be shifted by a specified number of places in a specified direction. Data shift may be used to provide the arithmetic operations of division and multiplication by powers of two or for 'packing' data.



The module caters for two forms of shift:

(1) *Arithmetic right shift* which causes the data to be shifted to the right, i.e. towards the less-significant end. The binary digits shifted 'off the end' are lost and the sign digit is perpetuated in the more-significant digit positions.

(2) *End-around left shift* which causes the data to be shifted to the left and the binary digits off the more-significant end to be reinserted at the less-significant end.

CONNECTIONS

DATA INPUT : (1) Address from instruction register (gated into cycle counter).
(2) 18-bit data from input busbars.

DATA OUTPUT : 18 bits to output busbars.

CONTROL SIGNALS : The control signals to perform the shift operations, to gate data in and out of the module and to reset the registers. These signals are obtained from the shift-function module.

ELLIOTT

PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

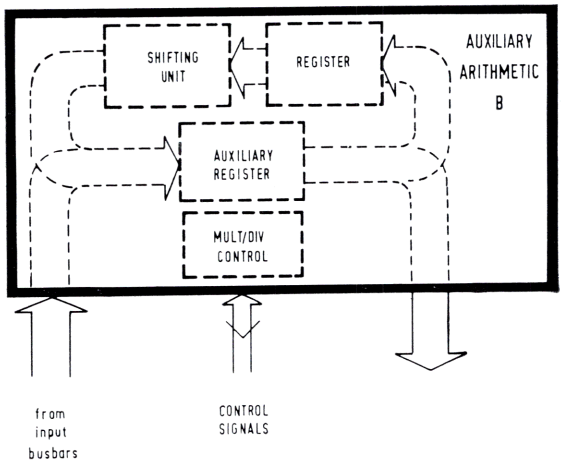
OPERATIONAL MODULE

AUXILIARY ARITHMETIC B

SIZE: 7W
CATALOGUE No. D24175

An auxiliary arithmetic module which in combination with the arithmetic and auxiliary arithmetic A modules provides multiplication and division facilities.

The multiplication of two single-word numbers produces a 'double-length' product. The extra register necessary to hold the second half of the product is contained in this module.



MULTIPLICATION

The general sequence of the multiplication process is as follows:

(1) the multiplicand is gated into the modifier register in the arithmetic module from the accumulator,
(2) the multiplier is gated into the auxiliary register,
(3) the multiplication process is carried out, leaving the product in the accumulator and auxiliary register.

DIVISION

The sequence for division is as follows:

(1) the divisor is gated into the modifier register,
(2) the dividend is treated as a double-length number and gated into the auxiliary register and accumulator,
(3) the division process is carried out, leaving the quotient in the auxiliary register.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

CONNECTIONS

DATA INPUT : 18-bits from input busbars.

DATA OUTPUT : 18-bits to output busbars.

CONTROL SIGNALS : The control signals to carry the multiplication and division operations and to transfer the content of the auxiliary register to the accumulator. These are obtained from the appropriate function modules.

OPERATIONAL MODULES

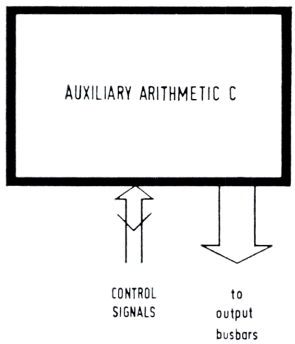
AUXILIARY ARITHMETIC C

SIZE : 3W

CATALOGUE No. D24176

This module, in conjunction with the arithmetic and auxiliary Arithmetic A and B modules enables square-root functions to be performed.

The square-root operation is performed using the multiplication and division facilities and adjusting intermediate results by means of signals generated in this module.



The number (positive fraction) is placed in the accumulator but is treated as a double-length fraction with the least-significant half composed entirely of zeros. This enables the result (which is placed in the accumulator) to be as accurate as the original number.

CONNECTIONS

DATA OUTPUT : 18 bits to output busbars.

CONTROL SIGNALS : The control signals to perform the square-root operation. These are obtained from the appropriate function module.

For further information please refer to other sheets or contact

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England

Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

OPERATIONAL MODULE

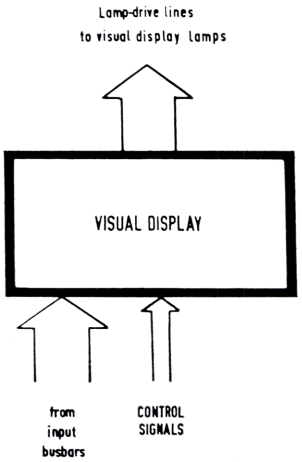
VISUAL DISPLAY DRIVE

SIZE : see table

CATALOGUE No. see table

A module which enables computer output data to be stored, decoded and displayed on in-line illuminated decade displays.

This module may be used to display operating instructions or as a monitor for stored data.



Data is gated into the module 4-bits (i.e. one binary-coded decimal character) at a time, decoded into decimal and displayed on the decade indicator specified by the address. Modules are available for 2- or 4-decade displays.

CONNECTIONS

DATA INPUT : (1) Instruction address from input busbars (Class 6).

(2) Data for display (4-bit B.C.D.).

CONTROL SIGNALS :

DE-SELECT O.D. : Resets the output device memory, prior to the selection of a new output.

SELECT O.D. : Gates in the address from the data input to set the memory associated with the appropriate output device.

R. – OP.R. : Resets the specified output register to zero prior to gating in new data.

G.I. – OP.R. : Gates the 4-bit B.C.D. character from the data input into a register from which it is automatically decoded and displayed on the specified decade of display.

MODULE	CATALOGUE No.	SIZE
2 DECADE	D24192	3W
4 DECADE	D24193	5W

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

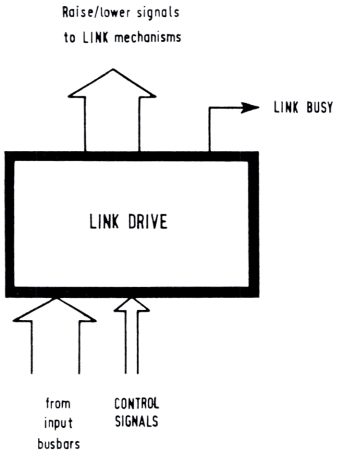
A member of the Elliott-Automation Group

LINK DRIVE

SIZE : 2W
CATALOGUE No. D24194

OPERATIONAL MODULE

Enables the set-point output of the LINK mechanism (see converter modules) to be raised or lowered one step at a time.
Up to 4 LINK mechanisms can be driven independently from this module.



On receipt of the appropriate control signals, the module generates a RAISE or LOWER pulse which causes the stepping mechanism in the specified LINK to raise or lower the set point output by one step.

CONNECTIONS

DATA INPUT : (1) Instruction address from input busbars (Class 10).
(2) Raise/lower digit (most significant digit 0=raise)
(most significant digit 1=lower)

CONTROL SIGNALS :

DE-SELECT O.D.	:	Resets the output device memory prior to the selection of a new input.
SELECT O.D.	:	Gates in the address from the data input to set the memory associated with the appropriate output device.
R. – OP.R.	:	Resets the specified output register to zero prior to gating in new data.
G.I. – OP.R.	:	Gates the raise/lower digit from the data input into a register and initiates the movement of the LINK mechanism in the direction specified.

OTHER CONNECTIONS

LINK BUSY : Indicates the period during which a LINK mechanism is moving from one step to the next. This is a common busy to all 4 mechanisms.

RAISE/LOWER signals to LINK mechanisms.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

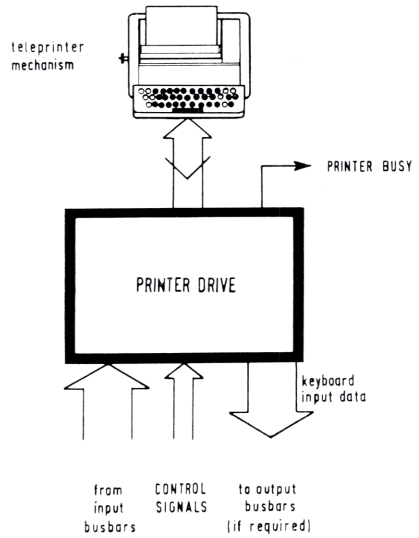
OPERATIONAL MODULE

PRINTER

SIZE : 4W
CATALOGUE No. D24198

A page printer (teleprinter) complete with suitable electronic control and drive.

This module enables data to be printed out with any format on continuous stationery. It also enables a printer with a keyboard to be used as an input device.



OUTPUT : When a five-bit telecode character is gated into the module, it is stored in a register and automatically printed by the page-printer mechanism. A 'busy' signal is provided during the time that the character is being printed. The maximum rate of printing is 10 characters/sec.

INPUT : The five-bit telecode character produced by pressing one of the keys on the keyboard is fed into the drive and may be gated out during the 40 ms that it is available (indicated by the presence of the 'ready' signal). The character is also printed.

CONNECTIONS

DATA INPUT : (1) Instruction address from input busbars (Class 4).
(2) Data for output (5-bit telecode).

DATA OUTPUT : Character from keyboard (5-bit telecode).

OUTPUT CONTROL SIGNALS :

DE-SELECT O.D. : Resets the output device memory, prior to the selection of a new output

SELECT O.D. : Gates in the address from the data input to set the memory associated with the appropriate output device.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

	R. – OP.R. :	Resets the specified output register to zero prior to gating in new data.
	G.I. – OP.R. :	Gates the 5-bit telecode characters from the data input into a register and initiates the printing of this character on the teleprinter mechanism.
INPUT CONTROL SIGNALS :	DE-SELECT I.D. :	Resets the input device memory prior to the selection of a new input.
	SELECT I.D. :	Gates in the address from the data input to set the memory associated with the appropriate input device.
	G.O. – IP.R. :	Gates the 5-bit telecode character from the keyboard out to the data output.

OTHER CONNECTIONS

PRINTER BUSY : Indicates the period during which the printer is printing a character.

PRINTER READY : Indicates the period during which a character generated by pressing one of the keyboard keys can be read into an ARCH computer by means of an input instruction. This busy is reset by the input instruction or automatically reset at the end of the printer cycle. This signal also sets up the PRINTER BUSY to prevent output to the printer when it is being used for input.

Data and control connections to the printer mechanism.

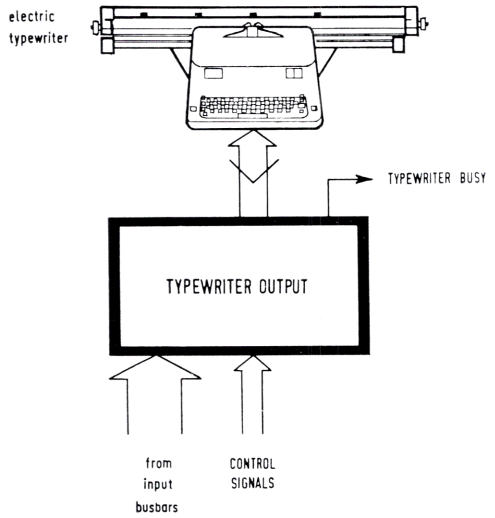
OPERATIONAL MODULE

TYPEWRITER OUTPUT

SIZE : see table
CATALOGUE No. see table

An electric typewriter complete with suitable electronic control and drive.

The typewriter has a 30-inch carriage and is particularly useful for applications involving a large amount of output, i.e. data logging, etc. Sub-modules enable 16, 24, or 32 distinct characters to be accommodated (including control characters such as ribbon-colour change, carriage return, etc.)



The 5-bit character gated into this module is automatically stored, decoded and typed out on the typewriter mechanism. Maximum rate of output: 10 characters per second.

CONNECTIONS

DATA INPUT : (1) Instruction address from input busbars (Class 8).
(2) Data for output (5-bit telecode).

CONTROL SIGNALS :

DE-SELECT O.D. :	Resets the output device memory prior to the selection of a new output.
SELECT O.D. :	Gates in the address from the data input to set the memory associated with the appropriate input device.
R. – OP.R. :	Resets the specified output register prior to gating in new data.
G.I. – OP.R. :	Gates the 5-bit telecode character from the data input into a register and initiates the typing of this character on the typewriter mechanism.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

OTHER CONNECTIONS

TYPEWRITER BUSY : Indicates the period during which the typewriter is typing a character.

Data and control connections to the typewriter mechanism.

MODULE	CATALOGUE No.	SIZE
16 characters	D24199	5W
24 characters	D24200	6W
32 characters	D24201	7W

CONTROL MODULE

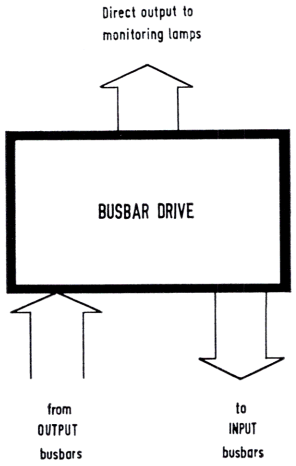
BUSBAR DRIVE

SIZE : 2W

CATALOGUE No. D24179

A module providing a power-drive connection between the OUTPUT BUSBARS and the INPUT BUS-BARS.

The parallel data outputs of all ARCH operational modules are connected to a set of OUTPUT BUS-BARS and the inputs to a set of INPUT BUSBARS. This module joins these two sets of busbars by means of a set of power amplifiers which provide the power to the data outputs necessary to drive the data inputs.



A large number of modules may be connected to the input and output busbars. (Up to 200 depending on loading.)

An additional drive output enables the number on the busbars to be displayed on monitoring lamps.

CONNECTIONS

DATA INPUT : connection from output busbars (18-bit)
DATA OUTPUT : connection to input busbars (18-bit)

18-bit direct outputs to monitoring lamps (24V 50 mA).

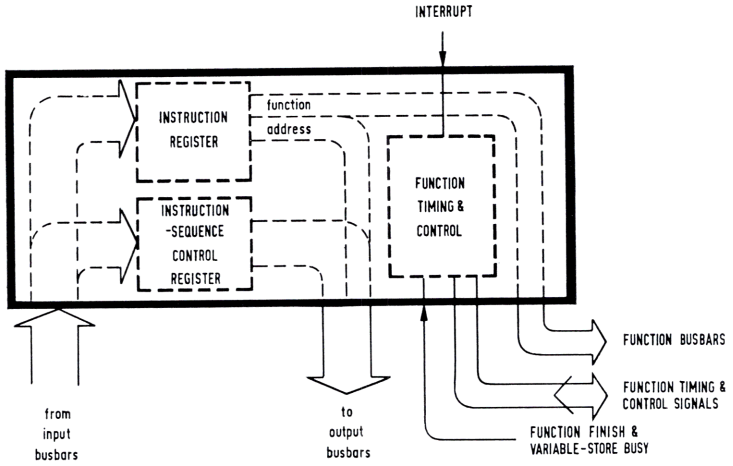
CONTROL MODULE

STORED-PROGRAM CONTROL

SIZE : 8W
CATALOGUE No. D24180

The central-control module for ARCH digital machines which have a program held in a store.

This module contains the instruction register, instruction-sequence control register and the function-timing and control units.



The beat-counter which controls the 'read' and 'obey' cycles of an ARCH digital machine is held in the function-timing and control unit. When this counter is set to 'read' the module dealing with function 0 is selected. Under the control of the function timing signals, this function causes the next instruction (as specified by the instruction-sequence control register) to be read from the store into the instruction register and then causes one to be added to the content of the instruction-sequence control register.
N.B. When the previous instruction is a 'modify' instruction a 'modify memory' in the timing and control unit is set and the instruction is modified before being read into the instruction register (see function module D).

The FUNCTION FINISH signal from the function 0 module changes the beat counter to the 'obey' beat and the function module specified by the instruction register content is then energised.

When this function is complete the FUNCTION FINISH signal is generated which returns the beat counter to the 'read' state so as to read the next instruction and so on.

The priority-interrupt line is fed into the timing and control unit. This line causes interruption of the instruction sequence after the completion of the instruction in progress at the time of interruption. The effect of interruption is that the sequence-control register content is retained at the value set after the previously obeyed instruction but the next instruction is read from location zero of a specified store (usually a fixed store).

Also fed into the timing and control units are lines from maintenance-panel switches which enable the computer to be operated manually. The functions available to these lines include single-step operation, control of the read/obey beat counter, manual input to the instruction register and accumulator, and monitoring of the store, arithmetic and programme-control registers.

The watchdog or self-check unit is contained in this module. This unit gives an alarm output unless reset by a 'reset watchdog alarm' output instruction. The watchdog alarm can be inhibited by an on-line/off-line input so that no alarm occurs if the machine is working off-line.

CONTROL MODULE

FUNCTION A

The function module generating the control signals required for:

function 00 (read next instruction)

function 08 (unconditional transfer of control)

and function 09 (conditional transfer of control)

SIZE: 2W

CATALOGUE No. D24181

FUNCTION 00 Read Next Instruction

This function is not specified from the program but is automatically selected when the program-control module is in the 'read' beat.

The control signals generated cause the content of the sequence-control register to be gated into the store address registers and the specified instruction to be read out of the appropriate store into the instruction register. A control signal is also generated which adds one to the S.C.R. content when the 'instruction' has been read out. If the 'modify' memory in the stored-program control module has been set by a previous function 05 instruction, the content of the modifier register of the arithmetic module is added to the selected instruction prior to it being gated into the instruction register.

FUNCTION 08 Unconditional Transfer of Control

This function causes the content of the sequence-control register to be stored in location 0 of a specified store and the address part of the instruction register content then to be gated into the sequence-control register replacing the previous content. This function thus causes the next instruction in the program to be taken from this new address instead of the address occurring in the normal sequence. Storing the S.C.R. content enables the program to return to the point at which the jump in sequence occurred. This instruction enables a program to carry out 'loops' of instructions.

FUNCTION 09 Conditional Transfer of Control

This function causes the address part of the instruction register content to be gated into the sequence-control register *only* if the content of the accumulator is negative. This function enables the program to make a simple 'decision' to follow one of two paths, i.e. to jump sequence or to remain in the normal sequence, dependent upon whether the sign digit of the accumulator is 1, (Acc. negative) or 0 (acc. positive).

CONTROL MODULE

FUNCTION B

SIZE : 2W

CATALOGUE No. D24182

The function module generating the control signals required for:

function 01 (replace c(acc) by c(store))
function 02 (write c(acc) into store)
function 06 (output)
function 07 (input)
and function 11 (write c(S.C.R.) into store)

FUNCTION 01 Replace c(acc) by c(store)
The control signals generated when this function is specified, cause the word held in the store location specified by the address part of the instruction register content, to be gated out of the store and into the accumulator. This word is also read back into the same location if the store has destructive read-out.

FUNCTION 02 Write c(acc) into Store
This function causes content of the accumulator to be written into the store location specified by the address part of the instruction register content. The accumulator retains its original content and is not reset to zero.

FUNCTION 06 Output
This is the only function dealing with output. The particular output device to which the function applies is specified by the address part of the instruction register content (see Instruction Code). The control signals generated by this function are those concerned with selecting the specified output channel (DE-SELECT O.D. & SELECT O.D.) and gating the output of the accumulator into this channel (R.-OP.R., G.I.–OP.R. G.O.–Acc.).

FUNCTION 07 Input
This function operates in a similar way to function 06 but deals with input devices. The control signals generated by this function are those concerned with selecting the specified input channel (DE-SELECT I.D. & SELECT I.D.) and gating the output of this channel into the accumulator (R.–IP.R., G.O.–IP.R., G.I.–Acc.).

FUNCTION 11 Write c(S.C.R.) into Store
This function operates in a similar way to function 02 except that the content of the sequence control register is written into the specified store location.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

CONTROL MODULE

FUNCTION C

SIZE : 3W
CATALOGUE No. D24183

The function module generating the control signals required for:
function 03 (add)
function 04 (subtract)
and function 10 (collate)
for machines having an arithmetic module.

FUNCTION 03 Add c(acc.) to c(store)

The sequence of operations carried out by this function is as follows:
(1) Reset the modifier and result registers in the arithmetic module and specify the arithmetic operation (Add).
(2) Read the content of the store location specified by the address part of the instruction register content, into the modifier register of the arithmetic module.
(3) Gate the content of the accumulator onto the output busbars to perform the addition function, and place the result in the result register.
(4) Gate the content of the result register out of the arithmetic module and into the accumulator.

FUNCTION 04 Subtract c(store) from c(acc.)

The sequence of control signals is the same as that for addition except that a 'subtract' function is specified and the result left in the accumulator is the difference between the two numbers.

FUNCTION 10 Collate c(acc.) and c(store)

The sequence of control signals is the same as that for addition except that a 'collate' function is specified and the result left in the accumulator is a number derived from collating the content of the accumulator with the content of the specified store location. i.e. this number has 1s in those digit positions which were both one in c(acc.) and c(store) and 0s in all other digit positions.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group
11.63

CONTROL MODULE

FUNCTION D

The function module generating the control signals for:

function 05 (set modifier)
function 12 (right shift one place)
and function 13 (left shift one place)
for machines having an arithmetic module and/or shift registers.

SIZE: 2W

CATALOGUE No. D24184

FUNCTION 05 Set Modifier
This function causes the content of the store location specified by the address in the instruction register to be gated into the modifier register in the arithmetic module and sets the 'modify' memory in the stored-program control module.

The fact that the 'modify' memory is set, causes the content of the modifier register to be added to the next instruction to be read from the store before it is gated into the instruction register.

This function can be used to *generate* a large number of similar instructions from one basic instruction rather than having to store them individually.

FUNCTION 12 Right shift one place (Arithmetic)
The control signals generated by this function cause the content of the accumulator to be shifted right (i.e. toward the least-significant end) one place by gating the accumulator output into a right-shift module and then out of this module back into the accumulator. The digit 'shifted off the end' is lost and the sign digit is repeated in the most-significant position.

FUNCTION 13 Left shift one place (End-around)
This function operates in a similary way to the right-shift function but requires a left-shift module. In this case the digit shifted off the left-hand (most-significant) end is re-inserted at the least-significant end.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

11.63

CONTROL MODULE

FUNCTION E

The function module generating the control signals required for:
function 14 (multiple shift right)
and function 15 (multiple shift left)
for machines having arithmetic and auxiliary arithmetic A modules.

SIZE : 2W

CATALOGUE No. D24185

FUNCTION 14 Shift right N places (Arithmetically)

The control signals generated by this function cause the accumulator content to be shifted right by a number of places equal to the value of the address part of the instruction register content.

This is achieved by repetatively gating the accumulator content into the register in the auxiliary arithmetic module and gating it back into the accumulator via a one-digit shift unit set to shift right.

The number of times that this is carried out, is controlled by a cycle counter in the auxiliary arithmetic module, into which the value N is gated from the instruction register.

Each time the accumulator content is shifted right one place, the sign digit is repeated in the most-significant position and the digit shifted off the least-significant end is lost. This function can be used for dividing the accumulator content by 2^N where N is the number of places shifted.

FUNCTION 15 Shift left N places (End-around)

This function is performed in the same way as function 14 except that the shift unit in the auxiliary arithmetic unit is set to shift left one place and the digits shifted off the left-hand (most-significant) end are re-inserted at the least-significant end.

End-around shift is used for 'packing' characters and shifting data in preparation for output instructions.

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

CONTROL MODULE

FUNCTION F

The function module generating the control signals for:

function 16 (multiply)

for machines having arithmetic, auxiliary arithmetic A, and auxiliary arithmetic B modules.

SIZE: 2W

CATALOGUE No. D24186

FUNCTION 16 Multiply c(acc.) by c(store)

The control signals generated by this module cause the content of the accumulator to be multiplied by the content of the store location specified in the address part of the instruction register content and the double-length product so formed to be placed half in the accumulator and half in the auxiliary register.

The general sequence of the multiplication process is as follows:

(1) The multiplicand is gated into the modifier register in the arithmetic module from the accumulator.

(2) The multiplier is read out of the specified store location and gated into the auxiliary register.

(3) The multiplication process is then carried out using the add and shift facilities of the arithmetic modules and the product is placed in the accumulator and auxiliary register.

The most-significant digits of the product are held in the accumulator and the least-significant digits in the auxiliary register.

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England

Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

11.63

CONTROL MODULE

FUNCTION G

The function module generating the control signals required for:
function 17 (divide)
for machines having arithmetic, auxiliary arithmetic A, and auxiliary arithmetic B modules.

SIZE: 2W
CATALOGUE No. D24187

FUNCTION 17 Divide c(acc.) and c(A.R.) by c(store)

The control signals generated by this module cause the combined double-length number formed by the content of the accumulator and the content of the auxiliary register in the auxiliary arithmetic B module to be divided by the content of the storage location specified by the address part of the instruction register content.

The general sequence of the division process is as follows:
(1) The content of the storage location specified by the address part of the instruction-register content (the divisor) is gated into the modifier register.
(2) The divisor is divided into the dividend formed by the combined content of the accumulator and auxiliary register and the quotient so formed is placed in the accumulator.

The functions necessary to effect transfers between the accumulator and the auxiliary register are contained in function module H.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group
11.63

CONTROL MODULE

FUNCTION H

SIZE : 2W
CATALOGUE No. D25235

The function module generating the control signals required for:
function 18 (replace c(A.R.) by c(acc.)
and function 19 (replace c(acc.) by c(A.R.)
for machines having function G and/or function F modules.

FUNCTION 18 Replace c(A.R.) by c(acc.)

This function is used for machines having a function G module and it enables the double-length dividend to be formed.

The sequence of control signals generated by this function causes the content of the accumulator to be gated into the auxiliary register (A.R.) contained in the auxiliary arithmetic B module.

FUNCTION 19 Replace c(acc.) by c(A.R.)

This function is used for machines having a function F module and it enables the quotient formed by a division function to be put into the accumulator.

The sequence of control signals generated by this function causes the content of the auxiliary register (A.R.) in the auxiliary arithmetic B module to be gated into the least-significant positions of the accumulator (the auxiliary register length is always one digit less than the length of the accumulator due to the fact that it has no sign digit).

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group
11.63

CONTROL MODULE

FUNCTION J

SIZE : 2W

CATALOGUE No. D25236

The function module generating the control signals required for:

function 20 (square root)

for machines having arithmetic and auxiliary arithmetic A, B & C modules.

FUNCTION 20 Place $\sqrt{}$c(acc.) in accumulator

This function causes the square-root of the positive fraction held in the accumulator to be calculated and the result placed back into the accumulator replacing the original fraction.

The number in the accumulator is treated as a double-length number with c(acc.) as the most-significant half and with the least-significant half composed of zeros.

The control signals generated by this function cause the square root of c(acc.) to be calculated using the multiplication and division facilities of the arithmetic module and auxiliary arithmetic modules A and B. Auxiliary arithmetic module C is used to control the calculation by adjusting the intermediate results as they are produced.

This function enables the square-root of a positive fraction to be determined to the same number of places as the fraction itself.

ARCH analogue modules are primarily intended for use in industrial control and computation systems but may be used in a wide variety of other applications including general-and special-purpose computers and simulators.

The signal levels, power-supply voltages and packaging are compatible with ARCH converters and digital modules thus allowing combined analogue and digital systems to be assembled.

The modules are fully transistorised and extensive use is made of printed-wiring techniques.

The complete range covers most of the usual analogue functions including integrators, multipliers and function-generators.

The range of modules has been sub divided into two sections:

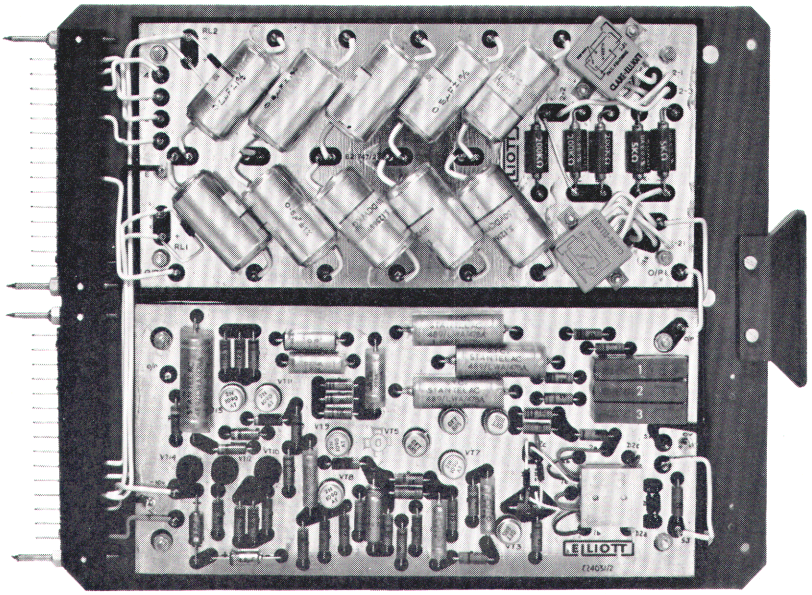
LINEAR FUNCTIONS	2100 – 2299
NON-LINEAR FUNCTIONS	2300 – 2399

For further information please refer to other sheets or contact

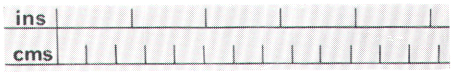
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

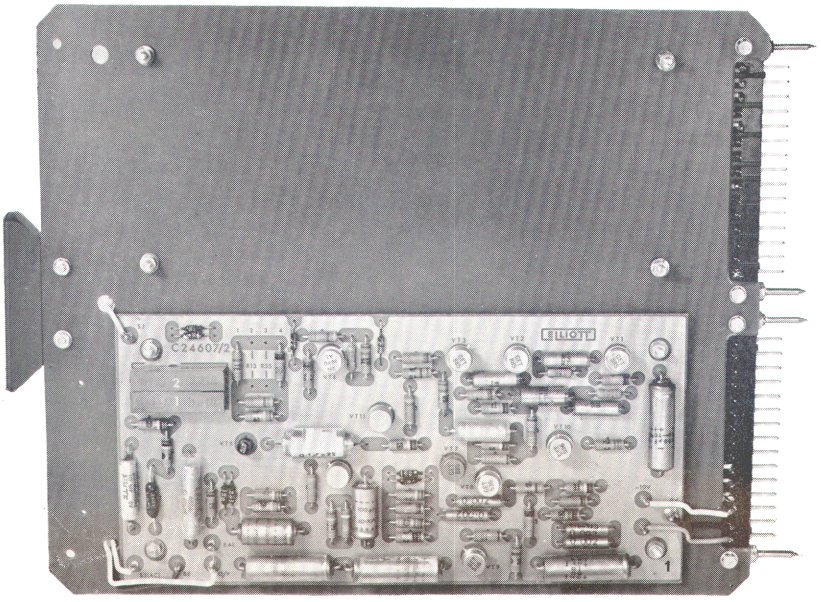
ELLIOTT

A member of the Elliott-Automation Group



Operational amplifier (lower) and network (upper) assembled to form an H size ARCH analogue module.

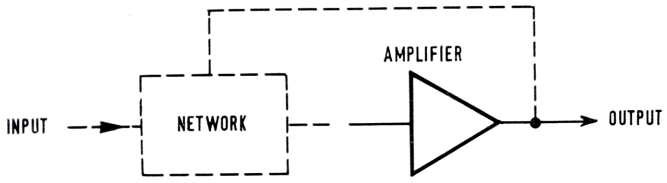




Drift-corrector mounted on the reverse side of the module to give it improved drift characteristics.

D.C. OPERATIONAL AMPLIFIER TYPE A

A fully transistorised d.c. amplifier with low drift and high open-loop gain designed for use with a wide range of input and feedback networks.



TYPICAL AMPLIFIER CONNECTION

SIZE : 1Q
CATALOGUE No. D 20928

SPECIFICATION

Open-loop gain at d.c.	Typically $>100{,}000$ (40,000 min.)
Open-loop gain at 25 c/s	3×10^4
Open-loop gain at 1 kc/s	6×10^2
Open-loop 3db point	25c/s
Drift referred to input	$<\pm25\mu V$ per °C
	$<\pm50\mu V$ long term at constant temperature
Output range	$\pm5V$ into 250Ω minimum load resistance
	Range reduced 3db per octave above 100c/s
Short-circuit protection	Output unconditionally proof against short circuit
Output impedance	75Ω max. (open-loop)
Maximum capacitive load	$1\mu F$
Input impedance	Not less than $8k\Omega$
Maximum overload input current	50mA peak for 1 sec
	10mA peak continuous
Recovery time (from I/P overload or 0/P short circuit)	Less than 10 ms
Input current	Not greater than $10^{-9}A$ (at constant temperature)
Working temperature range	0°C to 50°C
Recommended feedback impedances	500Ω to $200k\Omega$ N.B. Amplifier stable up to $10M\Omega$

POWER REQUIREMENTS : +10V : 50mA max. −10V : 50mA max.

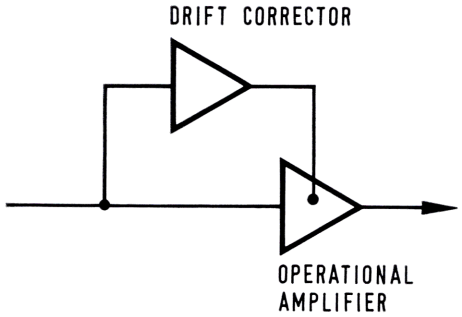
OPERATIONAL MODULE

DRIFT CORRECTOR

SIZE: IQ
CATALOGUE No. C24869

A fully transistorised 'chopper-type' amplifier for use in conjunction with the ARCH d.c. operational amplifier to provide a complete drift-corrected, very high gain, computing amplifier.

The drift-corrector is in the form of a sub-module which can be added to any of the functional modules, summers, integrators, etc. to give improved performance. The principal applications are to integrators which are required to operate over lengthy integration periods and sample/hold modules which are required to remain in the 'hold' condition for long periods of time.



COMBINED SPECIFICATION:

d.c. Correction factor:	Typically 1,000, never less than 500.
d.c. Input impedance:	100 Kilohms
d.c. Open-loop gain:	Typically 10^8, never less than 2×10^7.
Drift referred to input:	$\pm$ 5 microvolts per °C
Input current:	5×10^{-11}A at constant temperature. Drift: 5×10^{-11}A per °C

All other characteristics are as for the d.c. operational amplifier.
See Data Sheet No: 2100.

Power Requirements: + 10V: 60mA −10V: 65mA

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

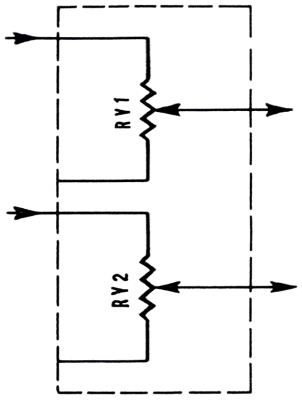
COEFFICIENT POTENTIOMETERS

Two 10-turn helical potentiometers fitted with counting-type dials and clamps.

This module is used for variable pre-set coefficients or parameters or variable manual input data such as set-values, integrator initial conditions etc.

All connections are brought out to separate pins so that each potentiometer may be used as a variable resistor or as a potential divider.

SIZE : 2H
CATALOGUE No. see table



SPECIFICATION

POTENTIOMETER RATING: 2 Watts
LINEARITY: 1% (0·1% to special order)
VALUES: See table

CATALOGUE No.	VALUES (Ω)
C21362/1	500
C21362/2	1,000
C21362/3	5,000
C21362/4	10,000
C21362/5	20,000

SUMMERS

SIZE : see table

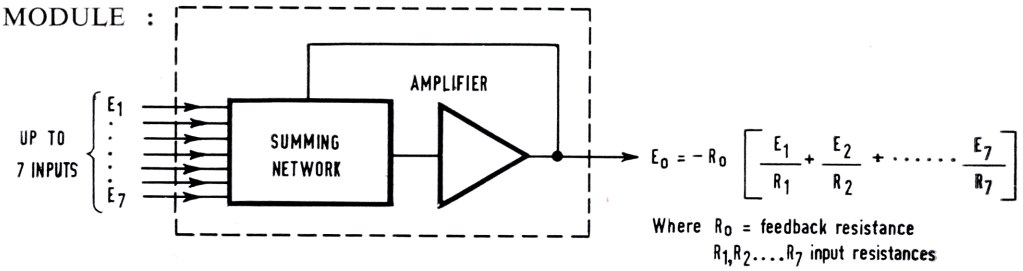
CATALOGUE No. see table

SUMMING NETWORKS : **For use with ARCH d.c. operational amplifiers to form SUMMING MODULES for summing, subtraction, sign changing and scaling.**

Each network component board provides for a maximum of one feedback and seven input channels.

Each channel provides for two series components and in two channels 'T' networks may also be accommodated. Complex networks to realise special transfer functions may thus be accommodated on the basic board (shown first in the table below) apart from the standard networks shown in the rest of the table.

SUMMING MODULE :



ACCURACY : NETWORK : All resistors precision wire-wound with 0.1% selection tolerance
MODULE : ±0.2% maximum gain error at d.c.
(bandwidth, drift, etc. determined by amplifier)

GAIN	INPUT RESISTANCE	FEEDBACK RESISTANCE	CATALOGUE NUMBERS			
			NETWORK	SIZE	MODULE	SIZE
—	NONE	NONE	B 23041	1Q	C23042	1H
7 × 1	10k	10k	B 23043/1	1Q	C23044/1	1H
7 × 1	100k	100k	B 23043/2	1Q	C23044/2	1H
4 × 1 3 × 10	100k 10k	100k	B 23043/3	1Q	C23044/3	1H
SIGN CHANGERS	10k	10k	B 23043/4	1Q	C23044/4	1H
	100k	100k	B 23043/5	1Q	C23044/5	1H
2 × 1 2 × 2 2 × 5 1 × 10	100k 50k 20k 10k	100k	B 23043/6	1Q	C23044/6	1H

POWER REQUIREMENTS : As for d.c. operational amplifier (see sheet 2100)

INTEGRATORS

SIZE : see table

CATALOGUE No. see table

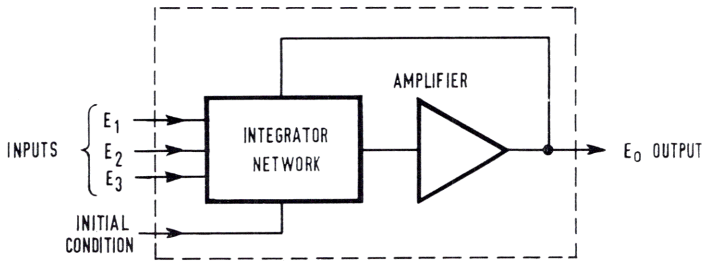
INTEGRATOR NETWORKS : **For use in conjunction with ARCH d.c. operational amplifiers to provide INTEGRATOR MODULES.**

'Initial Condition' and 'Hold' facilities are provided by means of high-quality sealed relays.

The network provides for three input channels and allow for a maximum feedback capacitance of 5μF.

The capacitors are high-quality polystyrene-dielectric components and precision wire-wound resistors are used.

INTEGRATOR MODULE : These modules may be used to integrate a variable with respect to time, generate time-functions, solve differential equations, etc.



TRANSFER FUNCTION :

$$Eo = -\frac{1}{C}\int\left(\frac{E_1}{R_1} + \frac{E_2}{R_2} + \frac{E_3}{R_3}\right)$$

where Eo is the output in the range −5V to +5V
C is the feedback capacitor value
E_1, E_2 and E_3 are input voltages
R_1, R_2 and R_3 are input resistor values

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

SPECIFICATION OF STANDARD INTEGRATOR MODULES

TIME CONSTANTS : Max. error ±1%
Stability ±0.1%

DRIFT (at constant temperature) : Not exceeding ± 200μV per sec. (with 5μF. feedback capacitor)

With drift corrector : Not exceeding ±20μV per sec.

INITIAL CONDITION : Max. error of setting ±0.2%

Networks are available with standard time-constants as indicated in the table.

The blank integrator network module (shown first in the table) enables other time-constants to be obtained by the insertion of resistors and capacitors to customer requirement.

To minimise errors due to input current, the largest value of feedback capacitor should be chosen.

TIME CONSTANT	C	R	CATALOGUE NUMBERS			
			NETWORK	SIZE	MODULE	SIZE
—	NONE	NONE	C23037	1Q	C23038	1H
3×1 SEC	5μF	200 k	C23039/1	1Q	C23040/1	1H
3×10 SEC	5μF	2M	C23039/2	1Q	C23040/2	1H
3×0·1 SEC	5μF	20k	C23039/3	1Q	C23040/3	1H
3×0·01SEC	5μF	2 k	C23039/4	1Q	C23040/4	1H
1×0·1 SEC 2×1 SEC	5μF	20 k 200 k	C23039/5	1Q	C23040/5	1H

POWER REQUIREMENTS :

AMPLIFIER: +10V : 50mA max. −10V : 50mA max.

NETWORK (2 relays used): −24V : 35mA per relay

LOW-PASS AMPLIFIER

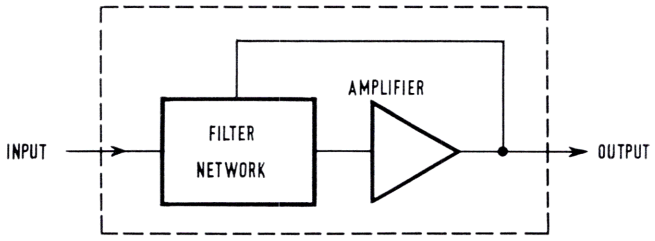
SIZE : 1H

CATALOGUE No. C21256

LOW-PASS FILTER NETWORK : **A network containing passive C-R networks which are used as input and feedback impedances for an ARCH d.c. operational amplifier to form an active low-pass filter.**

The gain is precisely defined in the pass-band and the attenuation has a constant slope.

LOW-PASS AMPLIFIER MODULE : This module is for use as a buffer amplifier, filter for 'noisy' input signals, etc.



SPECIFICATION

d.c. gain	1 ± 0·002
3db point	5c/s
Attenuation rate	12db/octave
Gain at 50c/s	35db down
Input impedance (d.c.)	100kΩ
Output range	± 5V
Minimum load resistance	250Ω
Rise time for step input	< 150ms

POWER REQUIREMENTS : + 10V : 50mA max. —10V : 50mA max.

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England

Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

LOGARITHMIC GENERATORS

SIZE : see table
CATALOGUE No. see table

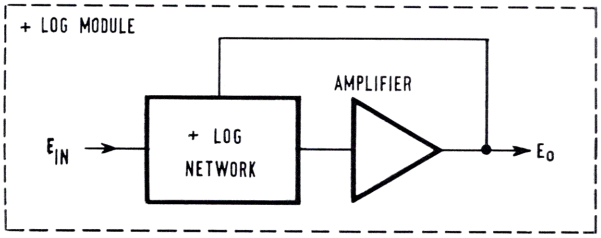
LOGARITHMIC NETWORKS : **For use in conjunction with ARCH d.c. operational amplifiers to provide LOGARITHMIC MODULES as function generators, multipliers, dividers, etc.**

The networks form variable input or feedback impedances which allow the input signals to determine the overall closed-loop gain of the associated amplifier. The functions are approximated by straight-line segments using biassed diode/resistor networks.

High function-stability is achieved by mounting all diodes in a temperature-controlled block.

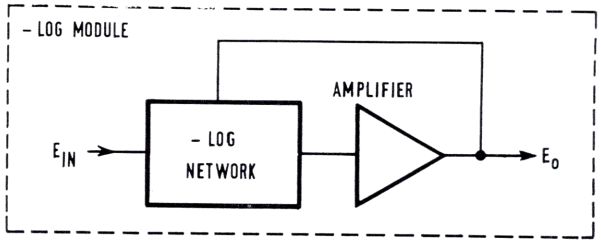
The four basic networks are: positive and negative log; positive and negative antilog.

LOGARITHMIC MODULES :



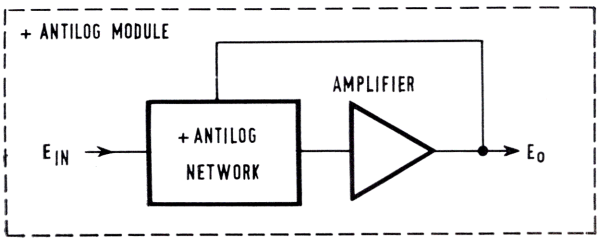
TRANSFER FUNCTION : $Eo = -2{\cdot}5 \log_{10}\left(\frac{E_{IN}}{5}\right)$

RANGE : E_{IN} : 0·05V to 5V
Eo : 5V to 0V



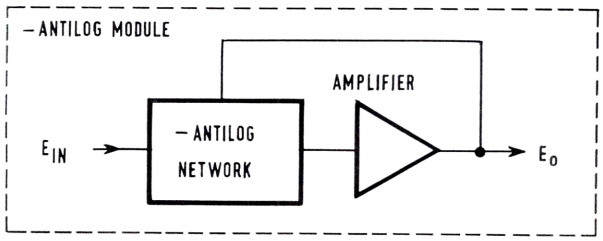
$Eo = 2{\cdot}5 \log_{10}\left(\frac{-E_{IN}}{5}\right)$

E_{IN} : —0·05V to —5V
Eo : —5V to 0V



TRANSFER FUNCTION : $Eo = 5 \text{ antilog}_{10}\left(\frac{-E_{IN}}{2{\cdot}5}\right)$

RANGE : E_{IN} : 0V to 5V
Eo : 5V to 0·05V



$Eo = -5 \text{ antilog}_{10}\left(\frac{E_{IN}}{2{\cdot}5}\right)$

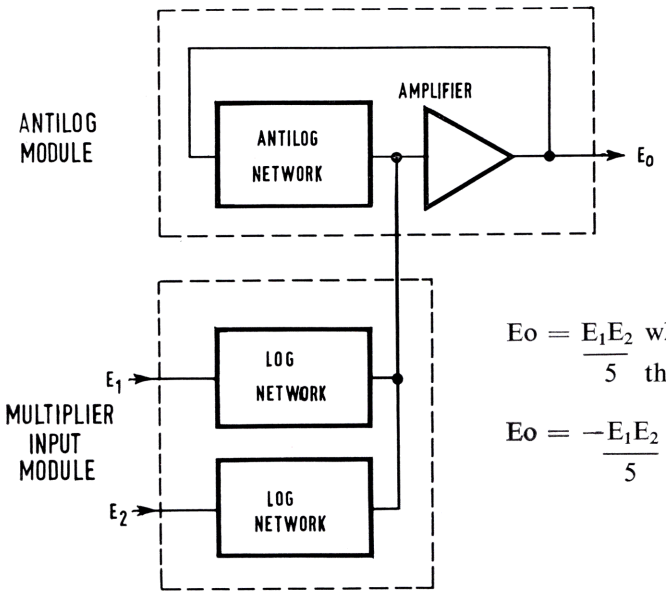
E_{IN} : —5V to 0V
Eo : —0·05V to —5V

MAX. ERROR : LOG MODULES ± 0·1% Full Scale.
ANTILOG MODULES ± 0·2% Full Scale.

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

ASSEMBLIES

SINGLE-QUADRANT MULTIPLIER : The connection of two or more similar logarithmic networks with an antilog module of the opposite sign provides a logarithmic single-quadrant multiplier.



$Eo = \frac{E_1E_2}{5}$ where E_1 and E_2 are in the range −50mV to −5V

$Eo = -\frac{E_1E_2}{5}$ where E_1 and E_2 are in the range 50mV to 5V

Two standard multiplier input modules are provided:

+MULTIPLIER INPUT MODULE having two + LOG NETWORKS for use with positive inputs

−MULTIPLIER INPUT MODULE having two −LOG NETWORKS for use with negative inputs

Max. error : ± 0·3% of full scale (Larger errors may occur at the extremes of the input ranges)

SINGLE-QUADRANT DIVIDER : Similarly, a single-quadrant divider may be assembled by connecting log networks of opposite polarity to an antilog module. (+antilog if divisor is positive, −antilog if divisor is negative)

In this case, the transfer function is given by:

$Eo = \frac{5\,E_1}{E_2}$ where E_1 and E_2 must be of opposite sign and in the range 50mV to 5V and modulus E_2 must be greater than modulus E_1

A standard divider input module is provided having two log networks of opposite polarity.

NETWORK	CATALOGUE No.	SIZE	MODULE	CATALOGUE No.	SIZE
+ LOG	C21568	1Q	+ LOG	C21651	1H
− LOG	C21902	1Q	− LOG	C21854	1H
+ ANTILOG	C21570	1Q	+ ANTILOG	C21652	1H
− ANTILOG	C23045	1Q	− ANTILOG	C23046	1H
—	—	—	+ MULTIPLIER INPUT	C21633	1H
—	—	—	− MULTIPLIER INPUT	C23047	1H
—	—	—	DIVIDER INPUT	C21778	1H

POWER REQUIREMENTS (1 NETWORK) : +10V: 10mA max. −10V: 35mA max. −24 d.c. smoothed: 400mA pk. 100mA av.

SQUARE-ROOT GENERATOR

SIZE : see table

CATALOGUE No. see table

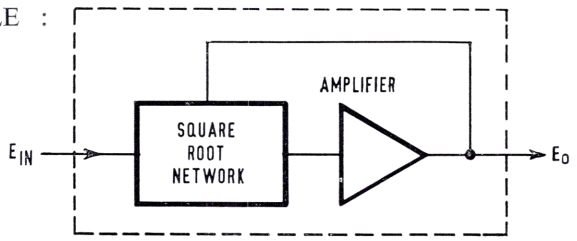
SQUARE-ROOT NETWORK : **For use with an ARCH d.c. operational amplifier as a square-root function generator.**

The network forms a variable feedback impedance for the amplifier so that the closed-loop gain is a function of the input signal.

The function is approximated by straight-line segments using biassed diode/resistor networks.

This module enables square-roots to be determined, flow signals to be corrected, etc.

SQUARE-ROOT MODULE :



	+ SQUARE-ROOT	− SQUARE-ROOT
TRANSFER FUNCTION	$E_O = -\sqrt{5E_{IN}}$	: $\sqrt{-5E_{IN}}$
RANGE	E_{IN} : 0.05V to 5V	: −0·05V to −5V
	E_O : −0.5V to −5V	: 0·5V to 5V
ACCURACY	Max. error : ±0.5% Full scale	

	CATALOGUE No.		SIZE
	+SQUARE-ROOT	−SQUARE-ROOT	
NETWORK	C21571	C24537	1Q
MODULE	C23031	C24534	1H

POWER REQUIREMENTS : +10V : 60mA max −10V : 60mA max

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

SQUARE-ROOT GENERATOR

SIZE : see table

CATALOGUE No. see table

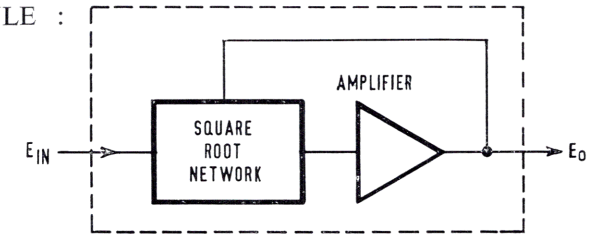
SQUARE-ROOT NETWORK : **For use with an ARCH d.c. operational amplifier as a square-root function generator.**

The network forms a variable feedback impedance for the amplifier so that the closed-loop gain is a function of the input signal.

The function is approximated by straight-line segments using biassed diode/resistor networks.

This module enables square-roots to be determined, flow signals to be corrected, etc.

SQUARE-ROOT MODULE :



	+ SQUARE-ROOT	− SQUARE-ROOT
TRANSFER FUNCTION	$E_o = -\sqrt{5E_{IN}}$	: $\sqrt{-5E_{IN}}$
RANGE	E_{IN} : 0.05V to 5V	: −0·05V to −5V
	E_o : −0.5V to −5V	: 0·5V to 5V
ACCURACY	Max. error : ±0.5% Full scale	

	CATALOGUE No.		SIZE
	+SQUARE-ROOT	−SQUARE-ROOT	
NETWORK	C21571	C24537	1Q
MODULE	C23031	C24534	1H

POWER REQUIREMENTS : +10V : 60mA max −10V : 60mA max

SAMPLE / HOLD

SIZE : see table
CATALOGUE No. see table

SAMPLE/HOLD NETWORK : **For use with an ARCH d.c. operational amplifier to provide analogue short-term storage or memory.**

The module is essentially an integrator without the normal input resistors and is switched from 'Initial Condition' to 'Hold' modes.

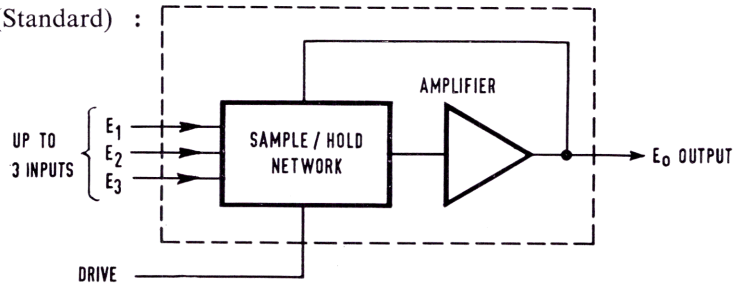
The input signal is applied as if it were an 'Initial Condition' and is sampled at intervals determined by an external DRIVE waveform which controls a sealed relay.

Three input channels are provided to allow a summed signal to be sampled.

A polystyrene-dielectric capacitor is used as a feedback capacitor and has a maximum value of 5μF. Precision wire-wound resistors are used and in the standard network have a value of 5kΩ.

A blank network board is also available to allow special network components to be mounted.

SAMPLE/HOLD MODULE (Standard) :



During sampling $E_o = -\left(\frac{E_1+E_2+E_3}{1+0.025p}\right)$ where E_o is in the range − 5V to + 5V $\left(p=\frac{d}{dt}\right)$

During hold, the output drift should not exceed ±200μV per second (at constant temperature)
The signal can thus be stored to within ±0·1% full scale for 25 seconds.
or to within ± 0·1% full scale for 250 seconds with drift-corrector.

NETWORK	CATALOGUE No.	SIZE	MODULE	CATALOGUE No.	SIZE
BLANK	C23033	1Q	BLANK NETWORK	C23034	1H
STANDARD	C23035/1	1Q	STANDARD	C23036/1	1H

POWER REQUIREMENTS : +10V : 50mA max. −10V : 50mA max. −24V d.c. smoothed : 35 mA max. for relay drive

ARCH converter modules enable combined analogue and digital systems to be assembled by providing conversion between the two types of representation.

The analogue-to-digital converter (A.D.C.) and digital-to-analogue converter (D.A.C.) modules provide direct conversion between ARCH analogue and digital signals. These modules have no moving parts and are composed of solid-state components throughout.

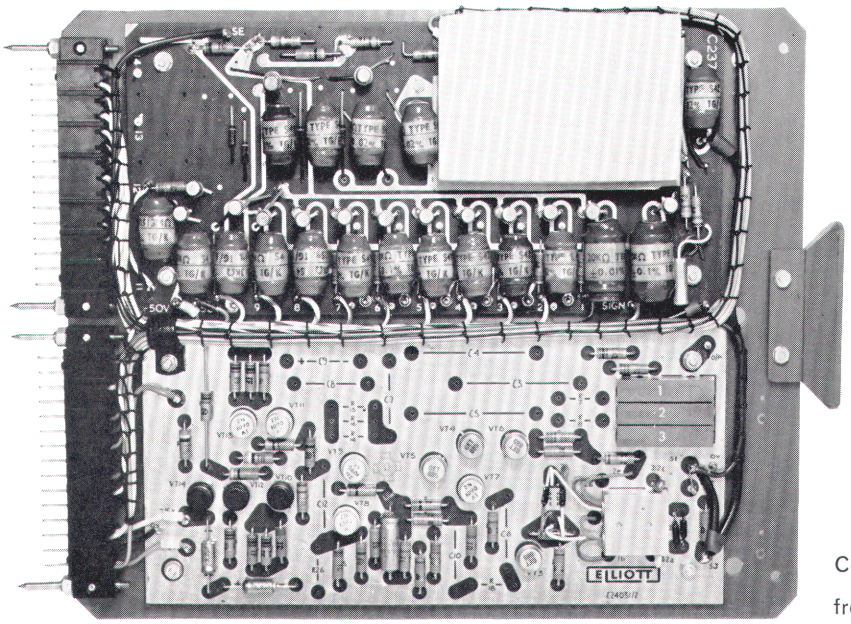
The Link controller modules provide conversion from ARCH analogue or digital signals to analogue signals suitable for operating the set point of conventional plant control loops.

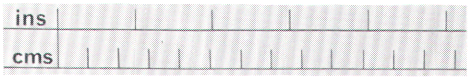
For further information please refer to other sheets or contact

PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

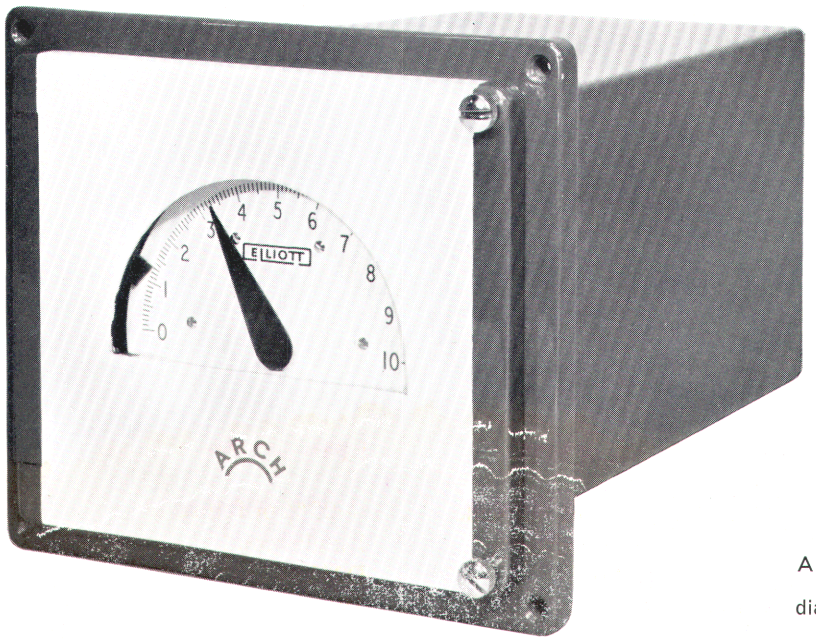
ELLIOTT

A member of the Elliott-Automation Group





Comparator amplifier assembly from ARCH A.D.C./D.A.C.



ARCH Link mechanism showing dial, pointer and limit stop.

ANALOGUE-TO-DIGITAL CONVERTER (1)

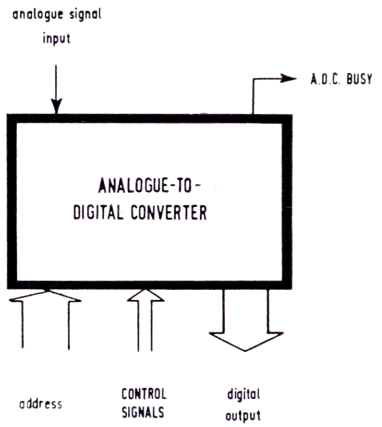
SIZE : see table

CATALOGUE No. see table

An A.D.C. module used for the input of analogue signals to an ARCH digital machine.

This module is complete with the gating facilities necessary for direct connection to the busbars in an ARCH digital machine.

Versions having binary and binary-coded decimal outputs are available (see table).



The converter has no moving parts, it being constructed from solid-state components throughout. The potentiometric method of conversion is used. Two instructions refer to the A.D.C. The first is an output instruction which initiates conversion, the other is an input instruction which gates the digital output of the A.D.C. onto the output busbars (see below).

SPECIFICATION :

	BINARY A.D.C.	BINARY-CODED DECIMAL A.D.C.
INPUT :	0V to ±5V d.c.	0V to ±5V d.c.
INPUT IMPEDANCE :	1kΩ	1kΩ
PARALLEL OUTPUT :	1 sign bit+12-bit binary	1 sign bit + 16-bit B.C.D. four decades: first three decades 0-9 last decade 0, 3, 5 or 7
RESOLUTION :	1 part in 4096	1 part in 4,000
MAX. ERROR OF CONVERSION :	±·05% of full scale	±·05% of full scale
CONVERSION TIME :	1 millisecond	1 millisecond
CATALOGUE Nos.	D25258	D25259
SIZE	7W	7W

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

11.63

CONNECTIONS

DATA INPUT : Instruction address from input busbars (class 1).

DATA OUTPUT : (1) Binary: 13 bits to most-significant output busbars.
or (2) B.C.D.: 17 bits to most-significant output busbars.

OUTPUT CONTROL SIGNALS :

DE-SELECT O.D.	:	Resets the output device memory prior to the selection of a new input.
SELECT O.D.	:	Gates in the appropriate address from the data input to trigger the A.D.C.

INPUT CONTROL SIGNALS :

DE-SELECT I.D.	:	Resets the input device memory prior to the selection of a new input.
SELECT I.D.	:	Gates in the new address from the data input to set the memory associated with the appropriate input device.
G.O. – IP.R.	:	Gates the A.D.C. register content out to the data output.

OTHER CONNECTIONS

A.D.C. BUSY	:	Indicates the period during (1ms) which conversion is taking place.

A version of the A.D.C. is available which is triggered automatically by the analogue selection module. In this way selection and conversion of an analogue input are carried out by one instruction, a separate 'trigger A.D.C.' instruction not being required.

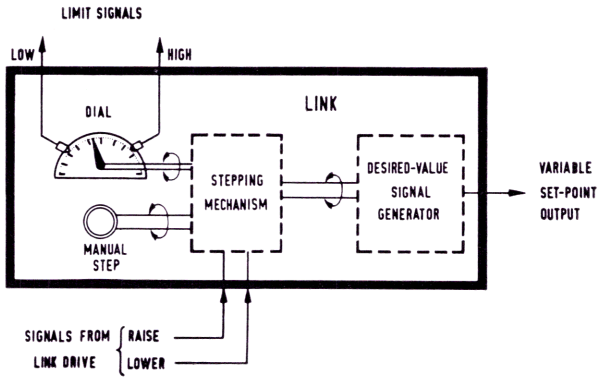
LINK

SIZE : see table

CATALOGUE No. see table

A mechanism giving an analogue output which can be varied over a range of 100 steps.

The analogue output is raised or lowered one step by means of a pulse-operated stepping mechanism which positively locks into position in between stepping pulses.



The stepping mechanism and dial arrangements are common to all types but a range of LINK modules is available with different desired-value signal generators (see overleaf).

Manually-adjustable limit stops are provided on the front dial.

These limit stops are mechanically locked to prevent overstepping the limit. Each stop contains a pair of switch contacts which give electrical indication that the stop has been reached.

A knob on the front of the mechanism permits manual adjustment of the set-point output to any value within the limits set.

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

11.63

SPECIFICATIONS

STANDARD STEPPING MECHANISM

RESOLUTION	100 steps
MAX. SPEED	5 steps/sec.
INPUT (RAISE OR LOWER)	24V pulse (1 per step) min. width: *make*: 40ms *break*: 40 ms
COIL RESISTANCES	40/40Ω
INDICATION	2″ pointer on 180° scale 100 steps indicated
WORKING TEMPERATURE RANGE	−10°C. to +40°C.

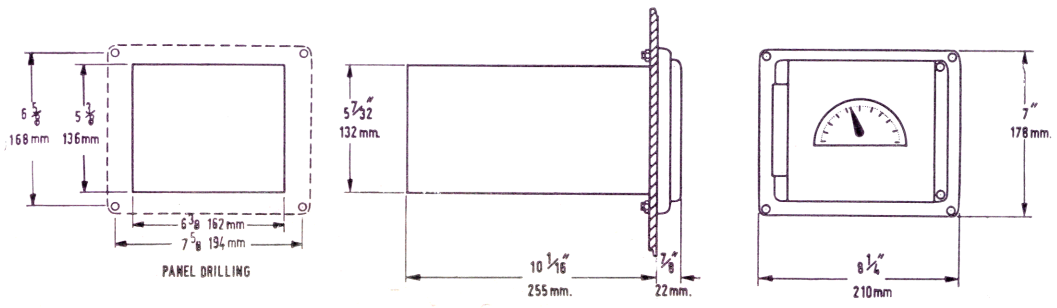
LINK MODULES

DESIRED-VALUE SIGNAL GENERATOR	PNEUMATIC	ELECTRICAL A.C.	ELECTRICAL D.C.
STEPPING MECHANISM	STANDARD	STANDARD	STANDARD
SUPPLY	20 p.s.i.	115V a.c. (reg.)	up to 25V d.c.
OUTPUT RANGE	3 p.s.i. – 15 p.s.i. (0·2 at. – 1 at.)	0 – 0·5V a.c.	0 – full scale or 250Ω potentiometer
OVERALL LINEARITY	±½%	±½%	±½%
CATALOGUE NUMBER	D19785	D19802	D20476

CONSTRUCTION : Each mechanism is mounted on a steel chassis which is fully enclosed in a sealed steel case with a windowed front panel showing the scale. The steel chassis can be drawn forward for maintenance purposes without withdrawing the mechanism completely.

MOUNTING : Panel mounting. Position not critical but calibration must be carried out in the mounted position.

FIXING DIMENSIONS :



Apart from providing set-p e d ital form of input enables LINK modules to b converters.

ARCH analogue and digital modules operate on the same power levels, namely +10V and −10V.

−24V is also necessary in some applications to operate contactors, solenoids and similar devices.

Any ARCH assembly therefore requires power supplies for +10V, −10V and possibly −24V.

The range of power modules provided consists of a number of proprietary supply units of selected specification and are suitable for all ARCH applications. Each of these modules occupies seven H-size board positions.

The standard ARCH single-bay power cabinet contains up to 12 power modules together with on/off switches, monitor lamps and power-distribution terminals.

For further information please refer to other sheets or contact

PROCESS COMPUTING DIVISI

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts
Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

POWER-PROTECTION

SIZE : 2W

CATALOGUE No. D26419

Provides protection to the power supplies and also to the modules in an ARCH system by ensuring correct operating levels.

This module continuously monitors the outputs of the power modules and shuts off all the supplies if any one supply voltage is not within its prescribed limits. It also provides a monitor lamp indication to enable a faulty supply to be traced.

Both long-term and short-term deviations from the acceptable limits cause the supplies to be shut off to prevent damage to the equipment.

Long-term deviations in power-supply voltage are detected in power-supply limit detectors in this module which cause the outputs of the power modules to be shut off by means of silicon-controlled rectifiers (S.C.R.s).

Very fast deviations in power-supply voltage automatically cause the S.C.R.s to fire and shut off the supplies.

The module provides protection for up to 12 separate supplies.

The module is in the form plug-in boards of the standard ARCH pattern and fits into the same cabinet or rack as the computing modules. It is connected to the power supplies by a cable which carries the reference and supply-monitoring connections as well as those to the monitor lamps.

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

11.63

Cabinets, racks, panels and associated 'hardware' of the ARCH system described in this section are all built on a 'modular' system so that additional or changing requirements can easily be accommodated.

The standard H-and W-size plug-in boards on which the electronic module elements are mounted, fit into socket channels which can be mounted in the standard ARCH cabinet and racks or in any other convenient mounting assembly.

The standard ARCH cabinets have a number of different shelf arrangements to accomodate any combination of contents required. The separate 'bays' of these cabinets can be joined together to form a multiple-bay assembly.

Other hardware details associated with input/output mechanisms are also given in this section.

For further information please refer to other sheets or contact

PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

ELLIOTT

A member of the Elliott-Automation Group

11.63

ARCH single-bay non-industrial cabinet

MODULE SIZES

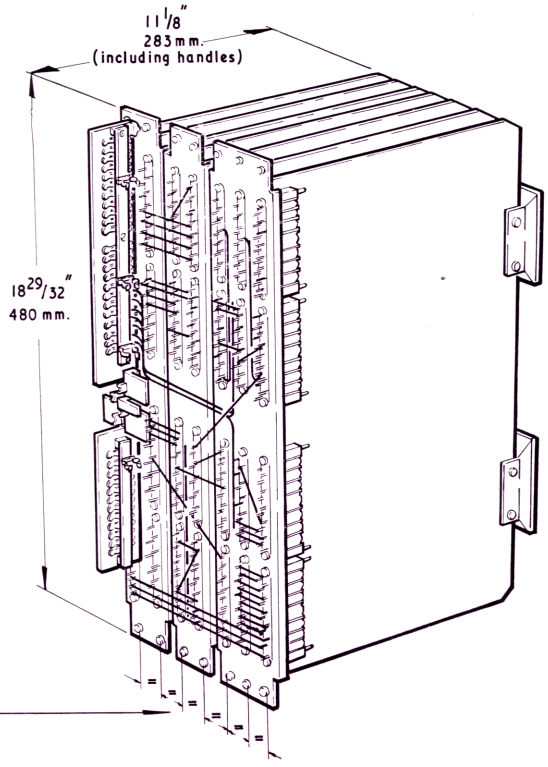
ANALOGUE MODULES

The Arch d.c. operational amplifier and the various input and feedback networks are each mounted on flat boards 3.8 in. (9.6cm) × 8 in. (20.4cm) designated Q size.

Two of these Q size boards mount on an H size plug-in board (see over). An ARCH analogue module consists of one or more of these plug-in boards with the Q size boards fitted and the plug connections made. The number preceding the Q in the analogue module data sheets indicates the number of socket positions taken up (not necessarily the number of plug-in boards).

The socket channels for mounting these boards into a cabinet or rack are available separately (see over).

DIGITAL MODULES



A 7W Arch digital module showing plug-in boards, socket channels, back-wiring and connecting fin

Arch digital modules are composed of a number of plug-in W size boards (see over) which contain the logic elements, **together with the socket channels and associated interconnecting wiring.** The input and output connections (data busbars and control signals) are made on a numbered 'fin' on the rear left of the module. Each connecting point on this fin is given a unique number and connections between modules are made only between points bearing identical numbers.

Here again, the number preceding the W in the digital data sheets indicate the number of W size socket positions taken up.

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

A member of the Elliott-Automation Group

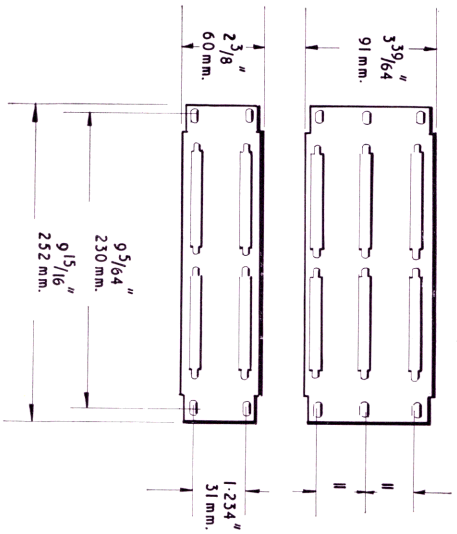
11.63

SOCKET CHANNELS AND PLUG-IN BOARDS

The two standard sizes of Arch plug-in boards are H and W. The sizes of these boards and the socket channels into which they plug are given below.

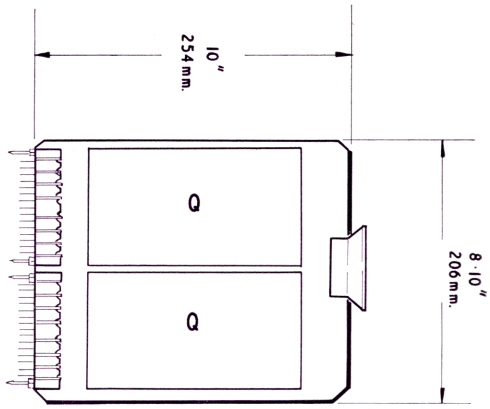
H SIZE

SOCKET CHANNELS
(COMPLETE WITH SOCKETS)



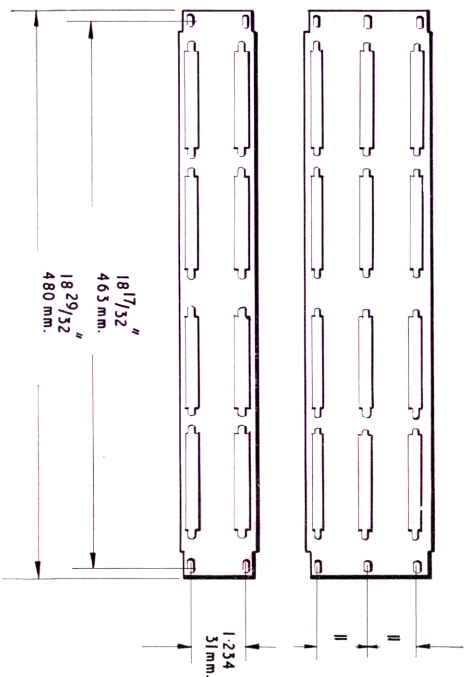
2 WIDE (2H)	B25382
3 WIDE (3H)	B24071

PLUG-IN BOARDS
(COMPETE WITH PLUGS & HANDLES)

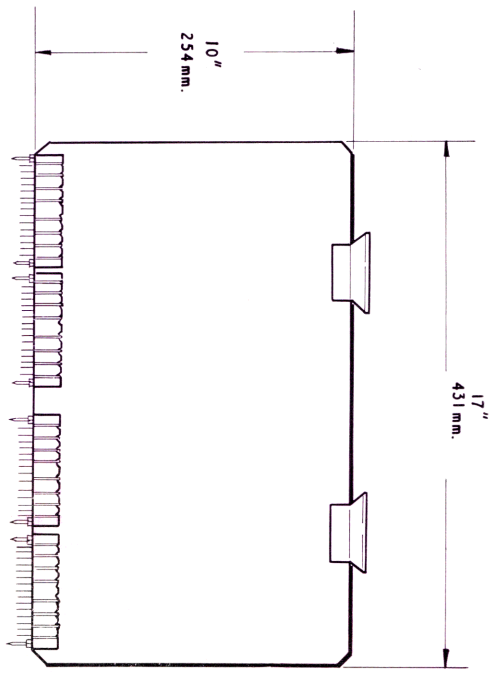


DRILLED FOR 2Q BOARDS	C21261
DRILLED FOR 24 MINILOGS	C20285

W SIZE



2 WIDE (2W)	C20287
3 WIDE (3W)	C21888



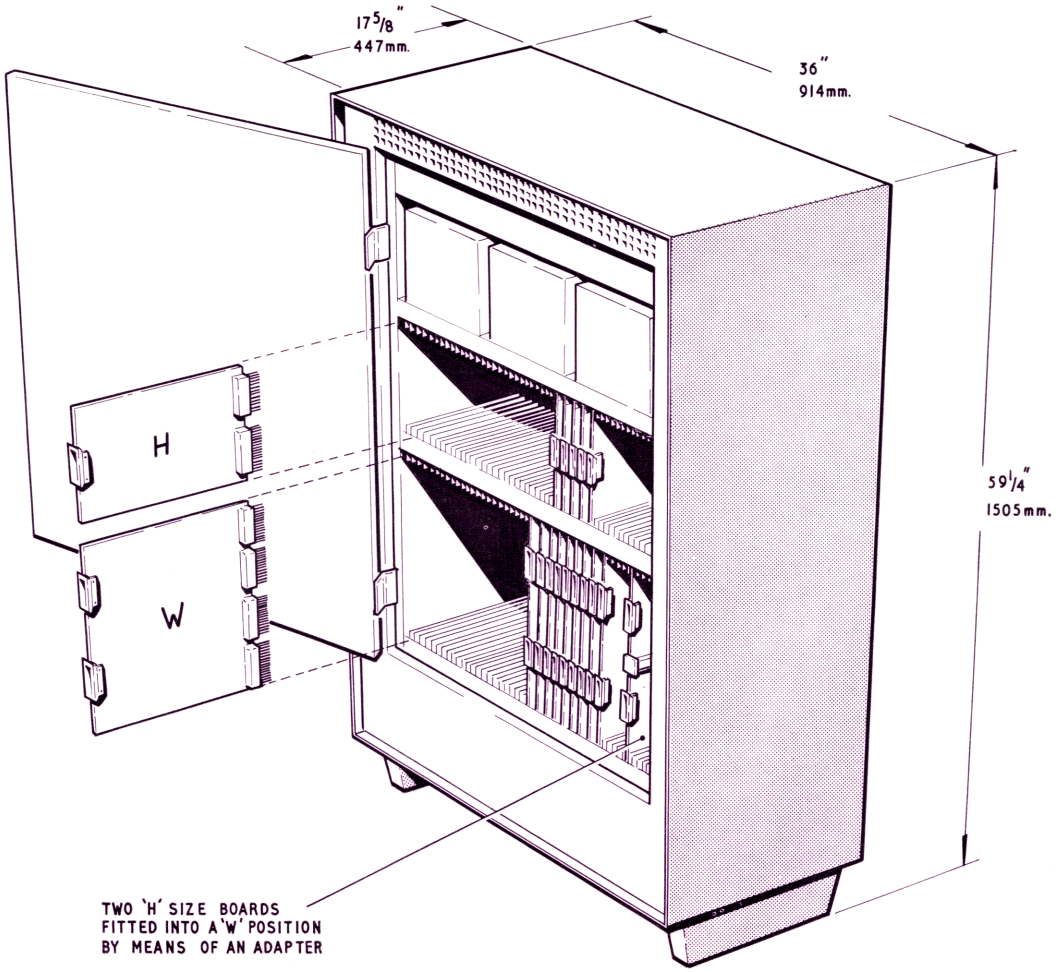
DRILLED FOR 48 MINILOGS	C21839

The plugs and sockets are 15-way with gold-plated contacts.

NON-INDUSTRIAL CABINET

A cabinet of unit construction which can be tailored to suit the requirements of any ARCH system.

Various shelf types and positions give mounting for H- and W-size plug-in boards and power modules and the large hinged doors provide easy access to the cabinet interior.



A single-bay Arch non-industrial cabinet with power modules mounted on the top shelf, H size modules on the second shelf and W size modules on the on the lower shelf.

The shelf arrangements and number of bays are arranged to suit particular applications (see over).

ELLIOTT

PROCESS COMPUTING DIVISION

Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267

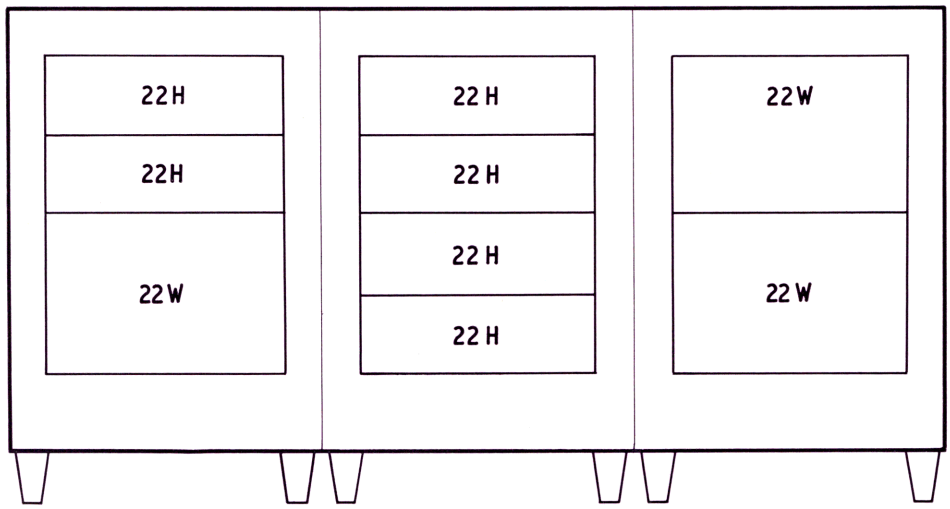
A member of the Elliott-Automation Group

11.63

NON-INDUSTRIAL CABINET

Two or three single bays can be fixed together to provide multi-bay cabinets under one outside 'skin'.

The shelves for the H- and W- size plug-in boards are complete with the U-section board guides. Up to 22 boards can be plugged into any shelf position.



A three-bay cabinet showing various combinations of shelf arrangement.

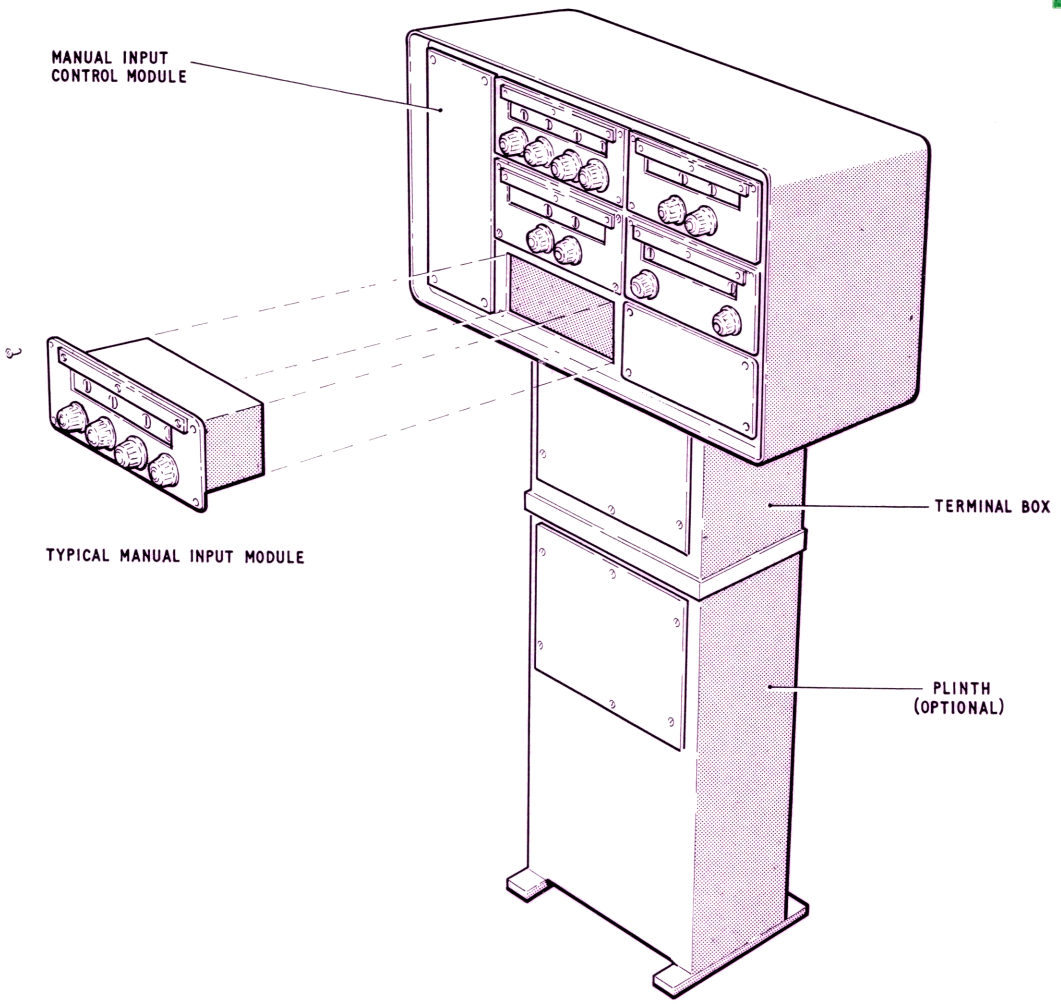
The cabinets may be ventilated by means of two fans (with air filters) mounted in the bottom compartment. Ventilation air is expelled through slots above the sealed doors.

A removable panel below each door provides access to the bottom compartment which has provision for the mounting of terminal blocks, fuses etc.

INDUSTRIAL MANUAL-INPUT CONSOLES

Designed for the input of decimal switch data into an ARCH digital computer (via the Decimal Data Input Modules).

The consoles consist of a number of multi-switch manual input modules and a control module mounted in heavy-industrial casing.



Clearly displayed above each switch is a number or letter indicating the switch position. Up to four switches may be accommodated in each manual input module.

The manual input control module is used for mounting the push-button which initiates the input of the data and may also be used to contain appropriate visual displays, i.e. O.K., WAIT and QUERY.

The control module may also contain a 'group selection' decade switch which is read in the same way as the other switches. The positions of this switch are used to indicate to the program which group or groups of decade switches contain the message to be read when the pushbutton is pressed.

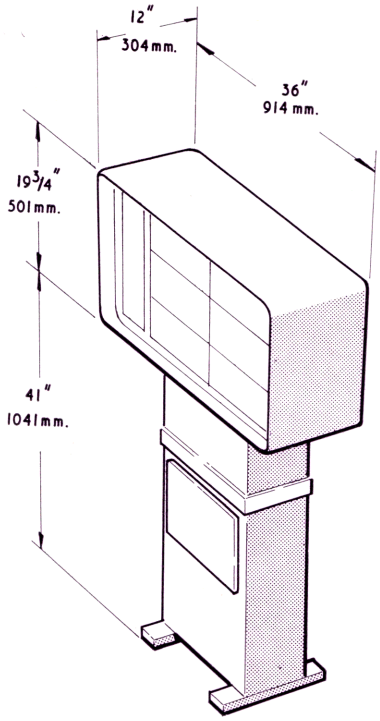
(see over)

ELLIOTT
PROCESS COMPUTING DIVISION
Elliott Brothers (London) Ltd Elstree Way Borehamwood Herts England
Telephone Elstree 2040 Ext 267
A member of the Elliott-Automation Group

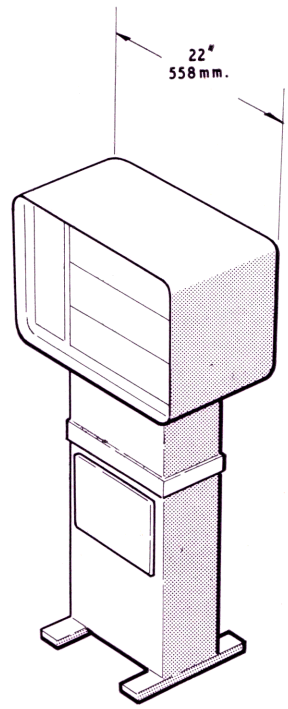
11.63

INDUSTRIAL MANUAL-INPUT CONSOLES

Two consoles cater for a maximum of three and six manual input modules.



6-WAY CONSOLE



3-WAY CONSOLE

Any arrangement of up to four switches can be obtained on any module. Since the switch data is read into the computer one switch at a time, its significance is determined by the computer program and the switches may be 'grouped' as required. Blank panels are fitted where no switches are required.

Connections are made to the console through a terminal box below the switch facia. Each manual input console is sealed against the ingress of dust and dirt and high-quality switches are used.

The consoles may be wall or plinth mounted.