SYSTEM INTERFACING

FOR ON-LINE

COMPUTER CONTROL

A Thesis submitted for the degree of

DOCTOR OF PHILOSOPHY

at the University of Aston in Birmingham

by

M. J. Rakic. Dipl. Eng.

October 1976

CONTENTS

Page

ACKNOWLEDGEMENTS

The author wishes to thank Professor R. H. Thornley MSc. Tech, PhD., A.M.C.T., E. Eng., F.I. Mech. E., F.I.Prod.E., Mem.J.S.M.E. for laboratory facilities and permitting this work to be carried out in the Department of Production Engineering, University of Aston in Birmingham and Dr. D. A. Milner, B.Sc. (Eng.) PhD., C.Eng., M.I. Mech.E., M.I.Prod.E. for suggesting the topic, supervising the work and constant encouragement and help throughout the whole project.

Thanks are also due to the University of Aston in Birmingham without whose final support of research could not have been done.

The author also wishes to thank Dr. Z. Ranic, for helpful and constructive suggestions, most valuable discussions and beneficial criticism, Mr. P. Jovanovic and Mr. J. Brindley for useful discussions, and also to Mr. M. Stanojevic and Mr. M. Gligic.

Finally, and by no means least, he acknowledges the patience and encouragement of his Mother and his Wife.

DECLARATION

No part of this work described in this thesis has been submitted in support of an application for another degree or qualification of this or any other University or other institution of learning.

M Ravic

Summary

On-line computer control of Numerically Controlled (NC) machine tools and associated peripheral equipment such as the large electrograph plotter is considered with a view to the problems of interfacing. The system interface connects the Benson-Lehner plotter to the minicomputer in a behind-the-tape-reader (BTR) mode.

A simulation of the actual signals from the bus-bar of the minicomputer is built so that it will be a relatively simple job to make the final connection. Also the transmission link between the computer and the plotting system requires high reliability of data transmission. The signals at the BTR unit and computer interface are of serial form but in order to link the units over a distance it is necessary to interpose some form of transmission system. The transmission lines have been built using twisted pairs with an overall screen to minimise inductive and capacitive coupling. The transmitter unit accepts characters in a parallel form and transmits them in serial form with the appended control and error detecting bits.

The system is capable of transmitting digital data at various rates dependent upon the demands of the peripheral unit under control.

LIST OF FIGURES

Fig. No.

33. The last function of the modulator unit.
34. The example of P.W.M.
35. The example of Sync. pulse.
36. The example of error checking.
37. The block diagram of the Receiver.
38. Electrical diagram of the Receiver.
39. Electrical diagram of the Encoder.
40. Interface
41. Time delay.
42. General interface board.
43. Address part.
44. Latch circuit.
45. Timing diagram of 'in-out' instructions.
46. Interrupt timing diagram.
47. External Bus cable Terminator.
48. Block diagram of Mini-computer.
49. Mini-computer system.
50. Block diagram of Central Processor.
51. Mini-computer Nova 1220.
52. Data words.
53. Nova system configuration.
54. Priority system.
55. The Benson-Lehner Electroplotter.
56. Block Diagram of Plotter.
57. Output from Tape Reader.
58. Wave forms from paper tape.
59. Timing diagram of holes.
60. Timing diagram of a programme.
61. Representation of paper tape.
62. Clutch circuit.
63. Connection of clutch circuit and circuit for correct timing of sprocket pulses.
64. Oscillogram of data and the sprocket.
65. Time delay circuit between data and the sprocket.
66. Sprocket generator.

67. Timing circuit between data and 'Enable' pulses.
68. 'Lock' (Enable) pulse.
69. System Grounding.
70. Vertical connection into the hierarchy of systems.
71. Four level diagram of a hierarchy.
72. Classification of decisions.
73. Parallelogram.
74. Dodecagon
75. Sector
76. Rectangle.
77. Four pointed star
78. Cartesian co-ordinates.
79. Interface components.

CHAPTER I

1. Introduction

In the Production Engineering Department of the University of Aston are several NC tools and a plotting table which are to be incorporated in a Direct Numerical Control (DNC) manufacturing system. In 1973 a decision was made to purchase a Data General Limited Nova Minicomputer in order to update an existing vertical milling machine and also to facilitate DNC and adaptive control of machine tools. Work previously has been carried out on an Adaptive Control Constraint (ACC) system. The machine chosen for retrofitting was a 2½D Starrag vertical milling machine fitted with an Olivetti CNZ system which incorporated a Tally tape reader, using E.I.A. tape format. This machine was used for the manufacture of prototype components, many of which were of complicated design. Also existing in the Department was a Benson-Lehner electroplotter; this plotter was originally used to check the tapes inputted to the NC machine tools.

It was therefore decided to include the electroplotter in the DNC manufacturing system using a behind-the-tape configuration. The plotting system is depicted in Figure 1.

1.1 The Simulation

Figure 1 shows the complete configuration of the designed interface. Examination of the construction of electronic circuits and the simulation of the entire system is performed by this interface.

1.1.1 Board of simulated switches from computer

This board consists of 21 two-position biased changeover switches. The plus position corresponds to the unit logical 'one' and the minus

position to the logical 'zero'. Seven of these are data switches in accordance with ASCCl code used by minicomputer Data General Nova 1220 and six switches corresponding to decimal 36 are used to address the electro-plotter. Eight other switches are available for functions such as Datoa, Datio, Clear, etc., so that when final connection is made to the minicomputer, these lines are available on the bus bar.

1.1.2 External Bus cable Terminator Board

Following the instructions of General Data Ltd, all data signals from the bus bar are connected to this terminator. Suitable resistors are used at the board to ensure matching impedance, hence minimising noise level during transmission of the signals. The outputs of this board are connected to the general interface board in parallel.

1.1.3 Memory Board

Figure 2 shows the memory board which is connected in parallel between the external bus cable terminator and the general interface boards. The data frame bus comes into the read/write access memory consisting of 256 address locations with word lengths of eight bits. The switch S_1 serves to reset all the circuits. The switch S_2 sets the contents of program into the addresses of the memory, starting always from the first address. A counter with maximum counts of 256 indicates the number of addresses used at anytime.

1.1.4 General Interface Board

This board of interface is built to

(a) Select device, in this case the electroplotter.

(b) Generate a specific time to control some particular function.

(c) Give decision for 'DONE' and 'BUSY'.

(d) Receive data which set in motion the electroplotter.

A display unit consisting of eight emitting diodes is connected to

the output of this board, the board indicating the entering of data into the memory and into the transmitter.

1.1.5 The transmitter

The transmitter, through the shift register, accepts characters from the memory board in parallel form and transmits them in serial form with appended control and error detecting bits.

The speed of transmission is determined by frequency of the signal generated within the master oscillator. A clutch circuit is employed at the transmitter, whose voltage effectuates the start and stop of transmitted message. Depending upon the speed required, twisted-pair or coaxial cable may be used in the transmission lines.

1.1.6 The Receiver

The serial to parallel receiver takes the characters required for programming the electroplotter in binary form, from the transmitter, verifies the information and outputs, in parallel form to the Encoder. This hardware encoder converts the ADCII code into the Elliott803 format that is required for movement of the electroplotter.

In the receiver are :delay circuit, clutch circuit and output amplifier. The delay circuit ensures exact delay timing between the data and sprocket pulses. The clutch circuit changes the output voltage between zero and plus five volts, this effectuating the start and stop of the receiver signals. The clutch signal also synchronises the reading-in of the block data (X and Y words) as required by the plotter. The output amplifier emits six volts, this voltage being required at input to the plotter. Five data lines and one sprocket signal go from this amplifier in parallel, to the behind-the-tape-reader and finally to the existing buffer of the electroplotter.

CHAPTER II

2. Review of relevant literature

Research into the Computer control and interfacing has been approached by many authors starting explicitly in 1967 and making a sharp slope in these last years.

American researchers have been very active in this field and many papers with different ideas and suggestions have been put forward.

We made comparisons with other leading countries in this field, for example, U.S.A., Japan and West Germany, but we could not find that they more advanced than Great Britain in the theory of computer control and interfacing; We were not able however, to compare the numbers of people practically employed in this field.

A wide range of actual references have been collected but only a limited number have been used.

Digital computer interface design is of vital and increasing importance in simulation, control, instrumentation, and data processing [1]. The article report reviews the important techniques for transferring data and control signals between digital computers and the outside world and discusses device selection and control for a party line I/O bus, device control, sensing, interrupts and automatic data channels with direct memory access.

It describes special interfaces for random - process measurements, and interface employing an intermediate digital processor are discussed.

This paper gives some practical solutions about the interface, and some parts information about interface for random process is advanced.

Kinter [2] discusses the two most common interfacing methods and

how they work. Viewed functionally, control - computer interfaces fall into two classifications : single-word transfers under computer program control and the second one, direct memory block transfers primarily under external control.

The first type requires the least amount of hardware, and gives the lowest transfer rate; the second requires more circuits but gives a high transfer rate.

Spur and Wentz[3] report how and where a small high-speed real-time electronic computer is used. The systems for direct numerical control (DNC) include usually a mini computer communicating directly with the NC machine to be controlled by means of a dispatcher.

Software and hardware are in modular structure to control units up to six machine tools of varied manufacture and type.

In 1972 such systems have been presented in the USA, Japan and Western Europe and should be installed in such factories where many NC machine tools are to be controlled.

The connection to the control unit can be behind the tape-reading-device or at the corresponding registers and memories. In this paper there are very advantageous ideas but they fail to give evident examples of the manufacturing pieces which can or which have been done. The paper in that sense is not practical.

The last level of computer involvement, CAM (Computer Aided Manufacturing), is still to be demonstrated, and potential applications are still subject to speculation. The bottleneck in manufacturing is the lack of suitable software, particularly with regard to the human interfacer problem. The next three to five years will require a major rational--level goal-orientated effort in software development and sensing tech-

niques. As the software picture becomes organised, the manufacturing plants will rapidly come around to true CAM systems. The exact path is not clear at present.

In the last years in Great Britain considerable effort has been made in this field, mainly in the Universities.

Wightman [5] gives a lot of interesting facts from the experience. He discusses the problem of the real time which arises because the computer is a serial device and suggests, if a designer is faced with this time constrain, he can remove some of the system logic from software, apply it to hardware, and interface this hardware to the computer through the computer's input/output section.

This procedure permits the hardware to aid the computer by performing some of the calculations or logic for it, while the computer is doing other calculations.

The real design dilemma is in making this hardware/software trade-off, so that the computer receives maximum utilisation and a minimum of hardware is added to the system. This design provides the maximum possible flexibility at minimum cost.

In the following, the advantages will be reviewed which result from the design of a control system of the type shown in figure 3.

The computer control system can reduce the complexity of the machine tool system. For example, the computer can calculate to compensate for minor machine misalignments.

Machine tool users can make changes from one size of machine to another, and from the control of two, three or four axes relative to a particular machine (assuming the appropriate servo interface is incorporated) by reading in the program tape designed for the particular machine.

The computer may, with a suitable data input and display, be capable

of indicating far more detailed error information in the event of program errors or machine failure than conventional control systems.

This means the computer can be programmed to provide a "watchdog" function, which ensures that programs are carried out with calculated frequency and that each of the individual programs is running correctly.

The machine - control system utilising a mini computer should be much less prone to obsolescence than a hard-wired control system, because the control - system character can be changed and features added as they become available. With a hard-wired control system this procedure is not economically practicable because in most cases it would require the control system to be returned to the manufacturer for rebuilding.

The same author [6] concludes, computer numerical control is more versatile and cheaper than wired control systems. Andreiv [7] considered that the problems encountered in applying a mini-computer to control of a limited production process is the high cost associated with the system integration effort and system software. The author states the fact that the necessary hardware is available from many computer manufactures, however, the users must spend many engineering hours customising interfaces, intergrating, and checking out the system. Secondly, they must know adequate languages or hire a programmer. In this case the cost of the software is normally equal to or sometimes, higher than that of the hardware itself, thus doubling the cost of the system.

In my opinion that problem still remains in 1975. The author writes about industrial interface which meets two criteria, interfacing flexibility operating reliability in "noisy environments".

The last criteria has been very useful during the planning of our particular interface.

Crossley and Lewis [8] give short reviews of the techniques used by

the firm Plessey without any details about the transmission of data for long distances the adjustment and timing behind the tape reader unit and data distribution.

The main ideas under subtitle are, transmission link, EIA standard behind-the-tape-reader unit, Plessey behind-the-tape-reader unit, the method of referencing part programme, DNC Computer configuration, size of data buffers, and Data Distribution, which are quite new and useful.

A behind-the-tape-reader (BTR) system is a DNC system in which the conventional hard-wired controllers and the tape readers at each machine tool are retained. A computer is executing data transfer to several machines in parallel with their tape reader input. A BTR system is shown in Fig. 4.

Deam has called this system a 'maximum flexibility'. With this ability to connect many different types of machine by various manufactureres through standardised interfaces, and, at the same time, allow independent operation of any machine tools, a maximum degree of flexibility is obtained.

Under the subtitle, "a critical look at computer control and the practical compromise"[8], the necessary programming, the size of the memory available and the speed of operation of the computer have all been looked at in detail by considering a milling machine cutting at a speed of 100 in/min with interpolation for every 0.0001 in. of movement. Only 60 micro-seconds will be available for each calculation and since a mini-computer may have between two and five micro-seconds for an addition and between eight and 20 micro-seconds for a multiplication, little time will be left for the performance of other procedures.

Even in systems where a larger computer is used on a time-sharing basis the data handling rates are likely to be critical.

The most powerful arguments in favour of computer control rely upon the flexibility that it can offer. But, as outlined previously, full advantage can be gained only from this flexibility if the programming difficulties can be overcome.

The paper (10) gives how computer application in the field of production technology can be broken down; control functions, planning, optimisation routines, analysis routines, simulation, design automation and operations research.

Under the subtitle, 'Control functions' we look at the use of the computer in the work of the production control department which can ensure flexibility and economy. In addition, the computer can analyse the information and modify and amend the details of manufacturing orders.

To take a simplified example in Planning Applications any project which cannot be arranged as a straight flow line, (i.e., one process proceeding directly to another), management has the problem of working out the correct sequence of activities. The modern way of doing this is by critical path analysis.

The computer cannot help here. Optimisation routing, linear programming simplex, linear programming transportation and assignment programming are designed to assist optimisation of resources utilisation, on profits.

The computer utilising an appropriate program will then be able to specify the mix that will maximise profit and indicate the resources and other factors involved.

The same paper has briefly given a revised display of facts 'why' and 'where' it is useful to use computers in Production Technology and Management.

The Machine Tool Industry Research Association (MTIRA) in this

country has developed its own CNC system incorporating a number of novel features. This system uses mini computer to perform the decoding, arithmetic, logic and control functions. It forms a basis of a general purpose CNC system for up to 6 axes on any common type of machine tool.

The fig.5 shows a block diagram of the system indicating the flow information between components. The mini computer communicates with most other parts of the system via specially developed general purpose interface.

The interface is able to pass information to a number of 'write only' registers, each holding one word, and to receive information from a number of 'read only' registers.

The interface can put out up to 256 bits of information to control the machine tool and receive the same amount of information back. The interface information is still in binary at the computer voltage level, (typically 0 to 5V), and it has to be converted to operate particular devices. Conversation is carried out in the block labelled 'peripheral electronics'. The MTIRA system is independent of the machine tool and of the types of control equipment. Another feasibility study carried out at Cranfield uses three CNC systems which have been successfully developed for line notion, positioning and continuous path. The live motion control project was started in 1968. The system had the minimum hardware interface required by the time four phase stepping motors used. This consisted of a two bit reversible shift register and four amplifiers. Every time an increment movement is required, an output from the computer to the interface is necessary. To simplify matters a separate bit is used for each direction on each motor.

A co-ordinate positioning control project began in 1969, based on a Digico Micro 16 computer and electrohydraulic pulse motor. This system

was designed for a relatively low resolution 0.001 in, with high positioning speed up to 480 ins/min coupled with high torque and acceleration. The values used were based on the findings of a technical survey into the requirements of British Industry, carried out by a post-graduate project group.

The system utilised a computer/machine interface to translate signals into a form capable of driving the actuatores.

A continuous path control was initiated in 1970 which involved the use of the same milling machine fitted with pulse motors on two axes, with provision for fitting a third axis at a later date. The block diagrams is shown in fig. 6.

The milling machine is in fact fitted with a dual interface so that it may receive single incremental commands for continuous path milling, or alternatively, commands for larger movements. Manual commands operate directly on the interface because no extra hardware is needed, but in an engineering system it may be better to go via the computer.

Conclusions

In the conclusion of this chapter it must be said that the literature on this topic suffers from one major short coming. A great deal of work has been done on the theoretical aspects of this work but rarely could the practical solution of the problem be found in the literature.

In most of the papers on this topic, the authors have given the general idea with the description of the problems in connection with using the minicomputer with the aim of process control.

As far as the theoretical aspect of the subject is concerned the possibilities are very interesting and lucrative. Nevertheless as far as the practical aspect of the problem is concerned, the authors have not

taken the solution to cover a complete design of an interface between a mini-computer and the machine. The reason for this may be due to the restrictions imposed by the manufacturers.

CHAPTER III

3.1. Introduction

The information and data processing abilities of present-day general purpose computers are very significant and impressive, and it is customary to assign to them a variety of duties which may have nothing to do with the plant being controlled. The most important types of computers are process control computers,[11] which simultaneously do all the computational tasks needed in industry, ranging from the pay-roll to optimum control of a particular process. Actually, optimum use of a computer only occurs when its computational ability is being taxed to the limit.

In theory, control computers can handle optimisation problems but, of the static and dynamic variety in the latter we must require, however, that the dynamics of the process does not outrun the computer.

In practice, computer control of static processes is dominating the field. A computer can be used directly in the control loop, i.e., on-line computer control, or it may be used off-line, in which case it serves as an adviser to the advisor.

In figures 7 & 8 are shown two extreme cases; in between one has a whole spectrum of variations.

The off-line computer is a convenient arrangement when the process changes take place slowly, as in chemical processes.

The on-line variety of computer control is employed when the process of surveying must be or should be performed on a more or less continuous basis. The on-line computer is, as a rule, less available for time-sharing for other tasks. Because of this 'special purpose

computers' are often designed which will do one job only - say for instance - controlling the process.

3.2. Computer involvement in Numerical Control of Machine Tools

The rapid developments in electronics and widespread use of small computers has led to considerable reduction in their price, Figure 9, coupled with increased capability and operational reliability. For this reason and increased versatility, the decision has been made by some manufacturers to replace the NC controllers by mini computers whose software can then be adopted to match the particular control functions of the machine tool.

Numerically controlled machines constitute a class of automated machines which do not rely on orthodox mechanical methods for sequencing and positioning operations, but depend on more versatile and sophisticated means for programming machine functions and positioning tool slides. The worlds first NC machine tool was made at the Massachusetts Institute of Technology in 1952, and during the early 1950's work also started on the automation of machine tools in Gt.Britain, notably on a routing machine for the aircraft industry [12].

3.3. Machine Tool Control

The problem of controlling a machine tool can be roughly divided into two areas:-

1) Sequencing machine tool functions such as spindle drive, speed selection, coolant and tool considerations.
2) Control of cutting rate and tool positioning.

The accuracy of the machined parts produced will depend upon the accuracy of the template, which controls the tool position via a servo-

system, and upon the accuracy of the initial tool setting relative to the workpiece. Thus such machines although basically simple to control and therefore, relatively cheap to purchase, require a longer time to set-up than the more sophisticated numerically controlled machines in which punched cards and magnetic tape provide the sequencing and positioning demands, and servo-systems position the tools to the required accuracy.

As a result, numerically controlled machines are generally considered more suitable for single and small batch size production.

3.4. Brief review of N.C. machine tools.

Figure 10 shows a comparison of NC machine tool production in four countries during the last decade. It shows that the dominance of the American sector has been lessened during the last five years. British industry, due no doubt to economic recessions, has been virtually static. The European NC industry, notably West Germany (especially in the field of turning machines) has been expanding rapidly. The Japanese have been by far the most active in recent years and the development of NC or group control systems for machine tools have been rigorously pursued by, among others, Fujitsu Fanuc Ltd. They appear to have overtaken the Americans and West Germany in numerical terms and also forward planning in automisation of the machining process.

3.5. The Computer in Numerically Controlled Machine Tool Systems

Technology is now advancing so rapidly that technical products are becoming obsolete at an ever increasing rate. In order to meet this fast changing demand, in addition to providing the expansion capability necessary to face the challenge of competition, the small series or batch

production firms, in particular, will need to give serious consideration to future production methods. Conventional NC goes some way to solving these problems, but, greater flexibility and increased sophistication of control can only be achieved by the incorporation of the computer into the machine tool control system.

The purpose of a computer in NC system is to carry out some or all of the control functions performed by hardware in a conventional NC system and to offer additional benefits. The basic hardware of the machine tool has a life of some 10 or 20 years, whereas the typical control system life is 3 to 5 years, both in terms of obsolescence, and component life.

It is therefore advantageous to consider the implications of replacing the standard control by some form of computer control.

3.6. The Execution of control functions

Control functions can be executed by programmes residing in the computer memory. The programmes are referred to as "software", and the system called "soft-wired" NC, can be implemented at various levels, depending upon the degree of system control required.

There are two broad functions that make up the NC process:

a) The control function: - Conversion of the coded instructions into the required machine movements and actions.

b) The supporting: - conversation of the part drawing and the processing and tooling technologies into a set of coded instructions.

The control function can do special purpose logic circuits, by a programme in a general purpose computer which is suitably interfaced with the machine tool, are by Hibrid. Combination of special purpose logic circuits and a general purpose computer is shown in Figure 11.

3.7. Computer Numerical Control. (C.N.C.)

This is an NC system in which a dedicated stored programmed mini-computer is used to perform some or all of the basic NC functions. This system is used for a simple machine tool (or a number of machine tools if the total number of controlled axes is small) in accordance with control programmes stored in the read-write memory of the computer. It is shown in Figure 12.

The control and operating programmes are read into the computer via the tape readers and held in the memory store.

The operating programme is the logic of the control system which defines the futures of the control, whereas the NC programme is the set of cycle instructions which define the movements of the machines. When a NC programme is read in, the data is processed according to the logic of the control programme and digital signals are provided at the output. Since many of the machines serve drives will only respond to analogue signals, it is necessary to interpose a digital to analogue converter.

3.8. The Role of 'Hardware'

The number of control functions that can be handled within the mini-computer is dependent upon storage and speed limitations and it may be necessary to leave some of the functions in hardware. In the hardware/software trade off, a rule is to use 'wiring' for repetitive operations which consume considerable computer time, e.g., slide interpolation. For smooth slide movements data would have to be supplied every 2 to 10 m sec. If the interpolation is done by wired logic, the computer would not need to supply data to this logic more than every 50 m sec.

A solution to this problem of storage and processing speed is the use of micro-programmes. This takes the form of a read-only integrated circuit board on which is stored instructional routines. The advantage of micro-programming is the read line which can be as low as 0.2 μsecs (micro secs). This is more than five times faster than the read time for a magnetic core storage memory.

Since these micro-programmes can only be changed by altering the circuit board, which is less convenient than software changes, it has been referred to as "hirdwared".

C.N.C. offers considerable flexibility since it is much easier, quicker and cheaper to modify software than the hardware of a conventional N.C. system. For example, changes from one type or size of machine tool to another can be affected by reading in a new programme tape. Also by means of a new control programme tape, the features of the control system can be expanded or modified to take advantage of advances in technology. This provides a buffer against obsolescence in a way that conventional hard wired systems cannot.

3.9. On-line Process control by digital computers.

Process control through digital computer offers many advantages over conventional analogue methods. Realization in the computer equipment, coupled with a system design based on a clear recognition and understanding of the requirements of computer control.

Figure 13 shows the items of equipment which make a typical control system are:

(a) Control computer (mini-computer) with core store, and, for larger systems, discs store.

(b) Cabling system, to bring the necessary plant measurement signals

to the computer, and to route (to start) to the process actuators the control signals generated by the computer system.

(c) Input devices to process the incoming plant signals and route them into the computer at the right time and in the right sequence.

(d) Output equipments, to convert the digital output of the computer into a set of signals to control particular plant.

(e) Programme's facilities,computer monitor panel, paper tape reader, printer, process control panels.

The operation of the control system is defined by the computer programme, which consists basically of a series of instructions punched onto paper tape reader and loaded into the computer through its paper tape reader on punch on the printer and loaded directly into the computer.

The computer obeys programs held in its core store; in larger systems a number of programs can be held on disc loading store, and transferred to the core store when required for use.

3.10. Advantages of computer control.

Computer control offers several important advantages over control by conventional analogue methods. The first is the economic benefit obtained by running the process more profitably. The second is the flexibility of the computer and in particular its adaptability to change. The third lies in the facilities provided by a computer system for general use on live calculations.

3.10.1 Improved Control

The plant performance and therefore profitability is improved by minimizing the effect of unwanted disturbances on the process, that is by improved control. Computer control is a major advance in this respect.

The very future of a computer control system is that the control calculations are carried out by program, so that is not constrained by the hardware limitations present in analogue systems. Advanced control functions can therefore be implemented without difficulty, whenever they are needed to improve the quality of control. Operations such as pure time delays, non-linear gains and feed-forward functions are readily available.

3.10.2 Flexibilities

The control procedure of a computer control system is defined by data held in the computer core store. Modifications to these procedures only involve changes in the stored information.

The system can therefore, be changed without difficulty, for example, to take advantage of improved knowledge of process behaviour.

Standard control software enables the process engineer to make such changes by keying the simple messages through a standard keyboard; the computer does not have to be re-programmed, and the changes can be implemented on-line.

3.10.3 Easy way to change one size of machine to another

Machine tool users benefit from a general-purpose control system which may be interconnected to any one of a number of different machines, change-over is made of operations being obtained by priming item with a special executive tape. Thus, changes from one size of machine to another, and from the control of two, three or four axes relative to a particular machine ,assuming that an appropriate serve interface is incorporated - may be made by reading in the program tape designed for the particular machine.

3.10.4 On-line Computing Facility

The computing facility of computer control enables it to carry out operations in addition to those directly concerned with control or optimization. For instance alarm monitoring and data logging, are of general application and are therefore included as part of software. Others, such as performance calculations, are programmed specifically for individual processes, and programming aids, such as higher level languages are provided to facilitate this work.

3.10.5 Optimization

A further contribution to process profitability may be achieved by the use of optimizing techniques. The plant measurements used for control can also be used for an optimization program to determine the optimum process operating conditions, taking into account economic considerations such as market demands. An optimization program maximizes a process 'performance index', which may be a technical index as yield or throughput, but is more usually an economic factor such as profit or return on investment.

The optimization program calculates the values to which the controlled variables of the process should be held in order to give optimum performance in the presence of disturbances, such as feed stock variations or market demands and the cost of labour and utilities.

The optimization procedure usually uses a model of the process, say a set of equations representing its behaviour, with hill-climbing on linear programming techniques to achieve the optimum solutions.

In a hill-climbing system a series of values of the controlled variables is tried out on the model, Figure 14, successive values being chosen by a pre-defined strategy to increase the performance index until a maximum is reached. The values which lead to this maximum on the model

are then used on the process itself.

With linear programming techniques the process behaviour is expressed as a series of linear equations which allow a direct calculation of the optimum values of the controlled variables.

On the other side, the efficiency of optimizing techniques depends on the number of variables involved. If the number of variables is too great then the hill-climbing process may not converge, while if linear programs are used solution lines may become excessive. The number of variables can however be minimized by improved knowledge of plant behaviour and improved control system design.

3.11. Short Conclusion

To summarise, a computer control technique will provide all functions offered by a conventional hard-wired system plus a number of important operating advantages. It enables control systems to be made more versatile and used on more than one type of machine with a consequent reduction in cost, because standardization of equipment reduces the engineering work required in the case of the custom-built equipment. Such system has a potentially greater project life than an equivalent hard-wired system, as it can not only be updated with new technology, but can also be stretched and adapted for more complex and ambitious automated systems, as required.

CHAPTER IV

4.1. Introduction

The connection of a peripheral device such as a N.C. machine tool to a mini-computer directly via bus transmission is possible if the distance is short, according to suggestions of the manufacturers.

Data-General Ltd, gave only 50 feet distance between the computer and the controlled machine, but in this case the N.C. plotter is more than 150 feet away from the computer. In the first case of short distance, one can connect directly to the machine at the bus in parallel using twisted pair wires, regardless of noise, interference and reflection.

In the second case, a long transmission line, one needs to convert the parallel data to serial form.

4.2. The modes of operation of digital systems

There are two common modes of operation of digital systems, namely parallel and serial. In the parallel mode, each bit that is needed to represent the information occurs on a separate wire simultaneously together with other bits on the other wires. In the serial mode, the pulses are serially, one after another, and the information transferred by this pulse sequence can be transmitted from one place to another over a single communication link. In the simplest case, one data wire and one ground wire. Figure 15 shows the serial presentation of a Binary number of 101010.

4.3. The basic circuit of serial to parallel transmission.

The circuit shown in Figure 16a transforms a parallel bit code into a serial code. In the circuit there are four inputs, $2^0, 2^1, 2^2, 2^3$ parallel data and one output. This circuit is composed of one two bit counter, and one two bit selector. Parallel information is continually available at the input. The clock will set information the 'two bit counter' through the states 0, 1, 2, 3. As a result, the counter will select the gates 0, 1, 2, 3. As a result, first bit 0 will be sampled and will appear as a pulse or no pulse at the output, then bits 1, 2 and 3. Figure 16b shows the serial output 1010.

4.4. Short Distance Data Transmission

The short distance data transmission is called, that, between local mini-computer and devices, where the binary form of the data is retained either as pulses or a 'base-band' signal. A base-band transmission is defined as a signal where the original binary signal is not used to modulate a carrier signal. The last explanation make the distinction between long and short distance communication.

In Figure 17 is shown a system data links for connecting the main computer to the small on-line computers that collect experimental data. These links use a short parallel transmission technique in order to obtain fast transfers of large quantities of data. Parallel transmission is suitable over the distances in the experimental area.

The system uses balanced twisted transmission lines with difference amplifier receivers. This solution was adopted in order to obtain connections of adequate length with low noise, low susceptibility to circulating ground current, and low crosstalk. The connection between central computer and the experimental area use a long twisted pair lives.

4.5. Synchronous System

In digital computer [13], there is an electronic oscillator called the clock, which generates a sequence of electrical pulses. As shown in Figure 18a, these pulses occur at a fixed interval time. The clock rate in the majority of todays digital computers are from 1 to 100 megacycles per second. The time between two adjacent pulses is called a clock period.

The clock pulse initiates information transfers among the registers, and each micro-operation is completed in one or several clock periods. When the computer operates in this way, it is called 'Sychronous', because the information transfers in and out from computer are in synchronism with the clock pulses.

The transfer out from computer means that data is transferred in a parallel way on a short distance, in that case the conception, sychronous, is valid, which in the case of long distance transmission would be impractical. Each operation instead takes a time interval and completion of one operation initiates the next operation.

The clock may also be built to give sequences of pulses. In this case, it is called a 'multiple - phase clock'. A multiple - phase clock has as many output lines as the number of phases, and a sequence of pulses is produced at each of the output lines. The time at which the pulse of one sequence occurs is equally off-set from the pulses of the other sequences. In Figure 18b, are shown the three sequences of pulses of a three phase clock. These three sequences may be obtained by distributing the clock pulses from an oscillation alternatively to the three output lines. The pulse rate on each output line of a three phase clock is 1/3 of the frequency rate of oscillation. By using the three pulses from the clock cycle, sequencing three micro-operations can be obtained.

4.6. The possibility of error

When information is transferred from the computer to appropriate sections into the system, it is possible for an error to occur. These errors occur for a great many reasons and cannot be completely eliminated regardless of how carefully the circuit is designed. The reasons for this lies in component failure and interference.

The full explanation of the system which performs error detection and correction in the particular work will be given in Chapter 5.

4.7. Long Distance Data Transmission

This title has been chosen to differentiate from the transmission of data between local data processing devices where the binary form of data is retained. Using standard interfaces the distance which may be placed between the computer and its peripheral is limited to a few feet. To go beyond this is dangerous if data is to be collected without corruption. Transmitter and receiver technology can, however, be used for distances up to a hundred feet. This system operates with serial data which is directly compatible with any of the peripherals available with serial interface options. See Figure 19.

4.8. Synchronisation in Serial Data Transmission

One of the major controls required in serial transmission systems is the character or word synchronisation of the receiver with the sending side i.e., the transmitter. With transmission links using particular receiver and transmission, the start of the first word cannot be directly indicated, because a message is complemented by starting the data train with a particular character, and sweeping through the beginning of the

received train for that character, as shown in Figure 20.

BITO is without the signal and for this reason the counter on the receiver side is started to count to the 4, and in the register to enter 1; in the meantime because of the lack the signal, when the counter is counted up the eight, it is reseted all the registers and made ready the system for transmitting of the information, namely BITO is performed the synchronisation of the logic. That process must be repeated after each turnaround of the transmitting information.

4.9. Transmission Line Information.

Transmission lines may be approximated by the lumped representation shown in Figure 21. The effect of the resistance, Ro on the characteristic impedance, Zo, is negligible, but it will cause some loss in voltage at the receiving end of long lines. The inductance and capacitance of the line in the presence of a ground plane are a function of the dielectric medium, the thickness and width of the line, and the spacing from the ground plane. Also, it can be shown that a signal sent down a line of constant characteristic impedance will travel along the line without distortion.

4.10. The Characteristic Impedance Zo

The characteristic impedance of a transmission line is an important quantity that directly governs the phase relationship between harmonic voltages and currents on a line.

Commercially available transmission lines have definite values of characteristic impedance such as 50 and 300 ohms, etc. This implies that the value is not only independant of frequency, but is purely resistive. Figure 22 shows a transmission line circuit where the term-

inal load is a non-reflecting termination and if V_1 and I_1 are the voltage and current at any co-ordinate Z on the line the characteristic impedance is

$$Zo = \frac{V_1}{I_1} = \frac{V}{I} = Z$$

4.11. Coaxial cable and twisted pair.

Coaxial cables and twisted pair lines have a defined characteristic impedance and are commonly referred to as transmission lines. Coaxial cable offers many advantages for distributing high frequency signals. The well defined and uniform characteristic impedance of the line permits easy matching. The ground shield minimises crosstalk. The coaxial line must be properly terminated with a resistance load equal to the characteristic impedance of the line.

The reactive component of the termination is of increasing importance at high frequencies. At such frequencies, the reactive component can change the terminating impedance, thus causing reflections on the line. In addition the effective inductance or capacitance would distort the output waveform, causing additional reflection down the line.

Low attenuation at high frequencies makes coaxial cable extremely vital for this type of transmission. For the microsecond frequency domain twisted pairs can be made from standard wire, twisted about 30 twists per foot. The cable then having a characteristic impedance of about 110 ohms.

4.12. General Pattern of Interfaces.

There are two broad classifications namely, single word transfers under computer program control and direct memory block transfers primarily

under external control. The first type requires the least amount of hadrware but gives the lowest transfer rate, the second one requires more circuitry but gives a high transfer rate.

All devices are connected to a common data transfer signal line usually called the in and out bus. Usually one bus is used for both directions of transfer.

To connect to the in-out bus, every device must have certain fundamental networks. Each device must have a selection net which guarantee that the device will respond when its device code is given by the programme.

The programme can handle in-out by sensing the busy and done flags or by allowing the peripheral device to interrupt when it requires service.

Signals on the control lines from the processor synchronise all the transfers on the data lines, start and stop devices, also control the programme interrupt and data channel. On the control lines to the processor the device can indicate the state of its busy and done flags and request a programme interrupt on the data channels.

The signal on the control line from the processor not only specifies a particular function but also supplies all the timing information needed for the execution of that function.

A device control unit usually requires timining circuits for its own internal operations, but no timing functions need be performed by the circuits that connect to the bus. All such timing is supplied by the computer's processor in the signals sent over the bus control lines. Moreover the control lines are set up so that a given device need only connect to those that correspond to the functions which the device requires as indicated in Figure 23.

CHAPTER V

ELECTRONIC CIRCUITS

5.1. Introduction

At the beginning of this chapter it is suitable to define some concepts of subsequent subjects. It is right to consider this system as a 'communication system' because it contains a source, a transmitter, a transmission path and a receiver, see Figure 24.

(i) The source selects one message out of a set of possible messages to be transmitted to the receiver terminals.

(ii) The transmitter operates on the message and produces a signal suitable for transmission path.

(iii) The channel is merely the medium used to transmit the signal from the transmitting to the receiving point. In the simplest case it is a pair of wires with time invariant transmission characteristics. The signal is almost always perturbed by noise. During transmission the signal or some part of the information sent out is lost.

(iv) The receiver operates on the received signal and attempts to reproduce from it the original message. Ordinarily it will perform approximately the mathematical inverse of the operations at the transmitter.

5.2. Basic Diagram of the Transmitter

The diagram in Figure 25 shows the units comprising the transmitter. The master oscillator gives twice the carrier frequency of the

transmitter, the speed of the data transmitted being dependent upon this frequency. The modular unit performs pulse width modulation of the message to be sent and also sychronises the signals going to the receiver.

The error generating part, performs the generation of parity bits in order to correct noises and other disturbances which can easily occur in complex systems. The sequential logic is implemented by different gate circuits, controls and ensures that the signals are performed in the correct sequence.

The data lines from the computer are loaded in parallel, into a shift register and converted into serial form. The contents of this shift register are sent out via the coaxial cable to the receiver and the necessary shift pulses are then generated by a local clock.

5.3. Electrical details of the Transmitter.

For the master oscillator the Texas Instrument SN74123 monolithic timing circuit is used. The free running frequency of the unit is controlled by external resistors and capacitors and is capable of oscillating at 10MHz. The wide range of frequency change (stabilised at $25^{o}C$) is advantageous, and for this application the resistors and capacitors are calculated for a frequency of 500KHz. See figure 26.

5.4. Counter Units.

The out frequency from the oscillator fosc (Figure 27) is applied to the input 'A' of the 7490/1 decade counter circuit. The flip-flop 'A' is used as a binary divide by two element so that at the output Q_A which is connected to the input 'B', half the oscillators frequency fosc/2 is obtained. This fosc/2 serves as carrier.

The input 'B' is used to obtain binary division by five at the

Q_b, Q_c and Q_d outputs (Figure 27), where only the last one is used. The last output Q_d gives 50KHz frequency, namely fosc/10.

This output Q_d is now applied to the second 7490/12 decade counter and ultimately a frequency of 25KHz is generated.

Then both the 7490 circuits are now connected to the same reset pulse, which is the logical output of a NOR circuit where phase (PH 1) and (PH 2) are input.

All four outputs from the last decade counter are connected to the four inputs of the 7442/15 BCD to decimal circuit. This circuit belongs to the modulator unit and its purpose is to decode BCD to decimal output.

The same number of pulses appear at all decimal outputs from pin 1 to pin 7. At the output 9 on 7442/15 appears 22 pulses in the period t_1 (Phase PH 1) and similarly 22 pulses in the period t_2 (PH 2) giving 44 pulses during the total phase as shown in Figure 28a.

From the last output Q_d of 7490/12 decade counter, which gives the lowest frequency in the transmitter's system, the output pulses are applied to the input A of the 7493/6 - 4 bit binary counters.

The counter 7493/6 counts 16 pulses, which correspond to the number of data bits.

In phase 3 (PH 3) the data is strobed into 4 times 7495/21, 22, 23, 24 shift registers during the read period when data is sent from the transmission line to the receiver.

The outputs Q_b and Q_c of 7493/6 and the next counter 7473/7 are used in further sequential logic circuits to perform exactly the twenty-two group bits which contain the full information of data given from the simulated switches or eventually, the minicomputer.

5.5. The functioning of the Transmitter.

Action starts at phase zero (PH 0) by the registration of the data into 16 bit right shift registers composed of 7495 circuits. Each circuit is a four bit parallel access shift register and are connected serially as shown in Figure 27.

Each of these have four parallel data inputs A, B, C, D respectively, and these come from the 16 data outputs given on the computer memory busbar.

Parallel loading is accomplished by applying four bits of data and taking mode control input and the clock pulse 1 high. The mode control is termed phase three (PH 3), see Figure 28b. Shift right is accomplished on the high-to-low transition of clock 1, when the mode control is low. At the same time all 7490 decade counters and 7493 four bit binary counter sets on the zero position. The decade counter set applying PH 0 at the input Ro(1) and binary 7493 applying BIT-0 at the input Ro(1). BIT-0 (B_o) is used for synchronising receiver operation with the transmitter. The complete check error parity generator consists From 8, 13, 14, 17, 18 circuits, see Figure 27. This register set to OOOOO, at PH 0. Immediately after PH 0 start the transition into phase 1 (PH 1), when parity check bit are generated. During this phase there exist 22 clock pulses (CL1). Only the first sixteen are active, i.e. they are used for rotation of the counter 4 times 7495 circuits. In this period parity bits in the second register are generated, that is, a pattern of five bits. It is ratio of two polynominals; see appendix 2. This is in fact rotation of 16 clock's pulses. It is the end of phase one (PH 1) and in the error's circuits 3 times 7473 and 7486 under the numbers 8, 13, 14, 17, see Figure 27, remains the remainder of the rotation of 16 bits eventually.

At the output from the last 7495 register, pin 10, No. 24 circuit in Figure 27, performs the timing diagram "DATA", see Figure 29, which at the same time comes in logical circuits No. 18, No. 11 and No. 9, to generate five bits necessary for error detection.

Phase two (PH 2) performs function of transmission of the information. It consists from 16 bits of data (information) plus 5 bits of parity bits, entirely 21 clock pulses plus one for bit zero (0). Phase two start by bit 0, when it has finished counting off 45 cycles, not 50 cycles because the receiver needs for its proper operation shorter time for BIT-0 (B_o). In the next phase (PH 2) the modulator starts to work. It generates 16 clock bits, which are applied to four 7495 shift register circuits at input clock 1 (R-SHIFT) and performs shifting of the contents of the register. During this time all registers in the modulator unit are in zero state, and every bit start to count at zero state. When 21 bits are finished automatically start phase zero (PH 0) and automatically injects new data or some, depending of particular current program.

5.6. Modulator - Pulse Width Modulation (PWM).

The technique of varying the width of a pulse by a modulating signal (DATA) is called pulse width modulation (PWM). Either the leading edge or lagging edge or both may be varied by the modulating signal. PWM gives a good performance signal - to - noise ratio.

To obtain PWM the action starts at circuit No. 10 in the Figure 27 which is a logical NAND circuit. The pulse $\overline{\text{START PH2}}$ is applied at input pin 2 and pulse $\overline{9}$ the decimal output of 7442 at input pin 5. The output pulses $\overline{\text{BIT-0}}$ ($\overline{B_o}$) and BIT-0 (B_o) are obtained from the pins 6 and 3 respectively. Using $\overline{B_o}$ as the input to another gate on the same circuit the outputs $\overline{\text{PH2}}$ = PH1 and PH2 = $\overline{\text{PH1}}$ are derived from the pins 8 and 11 which constitute the inputs to the next circuit, see Figure 30.

The above mentioned pulses are now applied to the next NAND circuit at the input 1 and 12, see Figure 31. The DATA train pulses are applied at pin 2 which gives the logical output on pin 3. The pin 3 is connected in turn to the exclusive NOR circuit at input 1, No. 13, which belongs to the unit of check error parity generator.

The input pins 9 and 10 are connected together resulting in a NOT circuit. The output at pin 8 is connected to K input of the circuit 7473/No. 8. The input J of the same 7473 circuit is connected to the input pins 9 and 10. This arrangement is designed to check error parity generation.

BIT-0 and $\overline{6}$ are applied to NOR circuit No. 11 at pins 9 and 8 respectively and at the output pin 10, see Figure 32, get train pulses consist of positive„6" with distance of B_o between them. It is„clock 2" so called in the Figure 27. If the DATA and pulse $\overline{2}$ are applied at pins 11 and 12. If the DATA is high it will 'inhibit' 2, but if the DATA is low it will give 'enable' „2", hence the train of pulses consisting of the DATA and the „2" pulses appear at the output. This train of pulses and the clock 2 are then applied to the pins 5 and 6. The output at pin 4 is now comprised of pulses 2 and 6. The train of pulses „6" is present at all times but the presence of 2 and 6 is dependent upon the DATA level. The resultant pulses at pin 4 and BIT-0(B_o) are then applied at pin 3 and 2 respectively.

The Width Pulse Modulation (WPM) starts at the output at this gate. The output of this gate is controlled by BIT-0.

5.7. The Last Stage of Operation - The Complete Message to the Receiver.

The circuit No. 16 and No. 20 are employed to provide the complete message, as shown in Figure 33.

The inputs to circuit No. 16 are provided for from circuit No. 11 (start PWM) and circuit No. 20. If it is necessary, however, the signal from circuit No. 20 (used for the synchronisation purposes) to be made logically positive before being fed to the circuit. This is achieved by taking the original signal from the circuit No. 20 together with $\overline{PH2}$ through a NOR gate, as shown in Figure 33.

The output from circuit No. 16 together with the signal containing the carrier frequency (fosc/2) provides the complete message, available at pin 13, see Figure 27.

The message is complete because it carries all the information, i.e. the data, the 'sync' pulse and PWM, see Figure 29.

The message and PH 0 are then taken through the last NAND gate to provide signal for the transmission line.

In this circuit forms the SYNC pulse. At the pin 1 pulse BIT-0 is applied and at pin 2 the pulse $\overline{0}$ which comes from 7442 BCD to decimal decoder No. 15, see Figure 27 circuit No. 20.

The output from the pin 3 is applied to the input at the pin 9 of the same circuit, and on the input at pin 10 came the signal from the output of the pin 11. This output is the logical signal of NAND function composed of BIT-0 (B_o) and $\overline{1}$ from 7442 decoder, see Figure 34.

The output at pin 8, then gives the SYNC pulse. The SYNC pulse is used at every data bit to synchronise logic and make ready to receive new data, see Figure 35.

5.8. The Error Checking and Correction.

One of the biggest problems in transmitter receiver system is to find an error in the message. The kind of error is unpredictable and it may cause, for example, the shutdown of an operation or even destroy

the plant. One way to minimize the errors is simple parity. Parity adds a single bit to a data word for checking purposes. A newer technique is error checking and correction. Parity adds a single bit to a data word for checking purposes. In odd parity, for example, the extra bit is set to make the total number of 1's odd. To a word with three 1's, then, an Oparity bit to be added. If a single bit error occurs in the word the total number of 1's changes to even, the parity would detect this difference. But parity only detects the presence of a single - bit error somewhere in the data word.

5.9. The Operation of Error Checking and Correction.

Error checking and correction is incorporated in both the transmitter and receiver, see Figures 26 and 37. As the 16 - bit data words are read into the shift registers five binary digits are added for checking. The check field allows for five different parity calculations based on the various bit combinations, examples of which are given below. Hamming's code is used to construct the check field from the data word stored in the shift register. Figure 36a shows the data word 1011100100100010. Bits 2, 5 and 10 through 15 together with the check bits Co must satisfy odd parity. There are three 1's, so the first check bit Co is set to 0 Data bits 4 through 10 and 15 must satisfy even parity so C1 is set to 1. Data bits 1 through 3, 7 through 9, 14 and 15 together with check bit C2, must satisfy even parity. C2 is set to 1.

Data bits 0, 2, 3, 5, 6, 9, 12 and 13 together with check bit C3 must satisfy even parity. C3 is set to 1.

The fifth check bit C4 is set with the data bits 0, 1, 3, 4, 6, 8, 11 and 13 to satisfy odd parity, i.e. in this case 0. The five computed

bits comprise a fault code , with code 00000 meaning no error. See appendix 2.

5.10. The Basic Diagram of the Receiver.

The block diagram, in Figure 37, shows the essential parts of the receiver. The diagram shows the pulse width detector, control unit and counters, check error unit, parallel access shift register, latch register and sprocket generator. The message consists of 16 data bits, 5 error checking bits and carrier frequency is sent from the transmitter to the pulse width detector. The detector is also connected with the control unit (sequential logic) and I, J, K, L counters. At the serial input of four bit parallel register are applied data. The outputs of this register are connected to the latch register whose outputs are in the ASCII code and are linked to encoder. All 'enable inputs' are connected on lock pulses which in logical high state enables transfer of data from the input to the output. The same lock pulses are applied to the sprocket generator circuit and enable precise start timing of sprocket pulses.

5.10.1 Serial to parallel register

The register consists of four 7495 circuits as illustrated in Figure 38 and to transform from serial to parallel the message is applied at serial input to the first circuit 7495/12. The shift right is accomplished on the high-to-low transition of clock 16, this being applied in parallel to all four circuits. The mode control and shift left are connected together and all four circuits in parallel to the train pulses named 'clear'. The pulses clear is formed in the NOR gate by circuit No. 6, from DATA and pulse '8'. Changes at the mode control input should normally be made while both clock inputs are low.

Such configurations of circuits 7495 has made it possible to con-

vert the message transmitted in serial form into the message obtained in the parallel form.

5.10.2 The output stage - The latch storage.

For this stage 7475 circuits are used, which are 4 bit-bistable latches, see Figure 38, circuit No. 23 and 24. They are suitable for use as temporary storage for binary information between processing units and output.

The data from serial to parallel register are presented at a data (D) input of latch circuits. The inputs are transferred to the Q outputs when the 'Lock' pulse is applied at enable pin is high. The Q outputs will follow the data input as long as the Lock remains high.

5.11. Encoder (ASCII to ELLIOTT 803 code

The minicomputer output is in ASCII code but the Benson-Lehner plotter requires Elliott 803 5 hole code. The encoding is done in hardware (so that simulation without the actual use of the computer may be done), the symbols appropriate to both codes are depicted in Table 1. Only the symbols which are required for programming the plotter have been considered.

From Table 1 may be written the five logical equations which are necessary for the design of the encoder circuit.

In the Elliott code table, under, for example column (4) (C) binary 1's or 0's are given reading from top to bottom of the column. The 1's in the Elliott 803 may be compared, reading horizontally, with the ASCII 1's as illustrated below.

ELLIOTT 803	ASCII						
4							
(C)	F	E	D	C	B	A	
1	1	1	1	1	1	1	..... (1)
1	1	0	1	1	1	1	..... (2)
1	1	0	1	1	0	1	..... (3)
1	1	0	1	1	1	0	..... (4)
1	1	0	0	0	0	0	..... (5)
1	0	0	1	0	1	0	..... (6)

Reading this Table in conjunction with Figure 39 the following equations may be obtained.

$(C) = (E + F) \cdot (A + B + C + D)$ from eqt. (1)

$= \overline{c} \cdot \overline{q}$ where $\overline{c} = E + F$ and $\overline{q} = (A + B + C + D)$

(C) $= (\overline{E} + F) \cdot (A + B + C + D)$ from eqt. (2)

$= \overline{b} \cdot \overline{q}$ where $b = E + F$

(C) $= (\overline{E} + F) \cdot (A + \overline{B} + C + D)$ from eqt. (3)

$= \overline{b} \cdot \overline{m}$ where $\overline{m} = A + \overline{B} + C + D$

(C) $= (\overline{E} + F) \cdot (\overline{A} + B + C + D)$ from eqt. (4)

$= \overline{b} \cdot \overline{n}$ where $\overline{n} = \overline{A} + B + C + D$

(C) $= (\overline{E} + F) \cdot (\overline{A} + \overline{B} + \overline{C} + \overline{D})$ from eqt. (5)

$= \overline{b} \cdot \overline{o}$ where $\overline{o} = \overline{A} + \overline{B} + \overline{C} + \overline{D}$

(C) $= (\overline{E} + \overline{F}) \cdot (\overline{A} + B + \overline{C} + D)$ from eqt. (6)

$= \overline{o} \cdot \overline{j}$ where $\overline{o} = \overline{E} + \overline{F}$ and $\overline{j} = \overline{A} + B + \overline{C} + D$

Similarly the five equations may be written:

(A) $= \overline{b}.\overline{q} + \overline{b}.\overline{k} + \overline{b}.\overline{m} + \overline{b}.\overline{d} + \overline{b}.\overline{j} + \overline{b}.\overline{h}$

(B) $= \overline{b}.\overline{q} + \overline{b}.\overline{k} + \overline{b}.\overline{n} + \overline{b}.\overline{d} + \overline{o}.\overline{j} + \overline{b}.\overline{j} + \overline{b}.\overline{i}$

(C) $= \overline{c}.\overline{q} + \overline{b}.\overline{q} + \overline{b}.\overline{m} + \overline{b}.\overline{n} + \overline{b}.\overline{o} + \overline{o}.\overline{j}$

(D) $= \overline{b}.\overline{k} + \overline{b}.\overline{m} + \overline{b}.\overline{n} + \overline{b}.\overline{o} + \overline{b}.\overline{d} + \overline{o}.\overline{j}$

(E) $= \overline{c}.\overline{q} + \overline{b}.\overline{q} + \overline{b}.\overline{o} + \overline{b}.\overline{d} + \overline{o}.\overline{j} + \overline{b}.\overline{h} + \overline{b}.\overline{i}$

The encoder is composed from NOT, two inputs NAND, eight inputs NAND, two input NOR and one BCD to decimal decoder circuits. It can be seen from Table 1 that if the symbol (pen up) is required then the ASCII code will be (A = 1, B = 1, C = 1, D = 1, E = 1, F = 1) and in Elliott code it will need to be((A) = 0, (B) = 0, (C) = 1, (D) = 0, (E) = 1).

CHAPTER VI

6. Interface - Mini Computer

The use of the word 'interface' is quite commonly used and people quite often speak about building an interface, when in face, they intend to construct a box to join two things together, see Figure 40.

The mini computer is a sequential machine, i.e. can do only one thing at a time, so, it is not particularly suited to monitor and control events that dynamically happen in real time. On the other hand, the equipment that we are trying to connect to a mini computer is most probably of such an imposition, and therefore is unwilling to disclose the details of its inner workings, and to yield to its controls.

This term should be treated with care since it does not refer to standard equipment with clearly identifiable elements that can be observed in every interface. For example, an interface used to control a production line may consist of a cabinet full of relays and registers, whereas an interface used in a DDC (direct digital control) loop may contain analog switches and analog-to-digital and D/A converters.

An interface may be located next to the mini computer and at some distance incorporating in its construction certain items of data communication hardware. The difficulties lie in the process where variables that are to be monitored and controlled are not suitable form to be connected to a mini computer. Much development work still needs to be done in this area.

6.1. Where are the difficulties of interfacing?

Interfacing is one of the main difficulties encountered in tying a mini computer to a given application.

Recently managers were delighted to learn that they could have a computer suitable for their specific application for only, say, £10,000. However, they were not so pleased to find out later that the particular interface would cost them ten - twelve times more than the original amount.

6.1.1 The classification and reasons of interfacing problems

No standard hardware

Unlike mini computers manufacturing where mass production techniques are used, interfacing usually involves special design and construction. The mini computer manufacturers help to ease the situation by providing modules for custom built units. These may be simple to assemble for the experts, but still a complex proposition for the average user. Furthermore, these modules provide no help at the point where the interface is tied to a process.

6.1.2 Cost

Interface equipment can easily exceed the cost of the computer mainframe. One should also not overlook the fact that special hardware such as transducers are needed to link to an application.

6.1.3 Reliability

Mini computers are inherently highly reliable machines. However, the interface, which may contain electromechanical units, operating in a noisy environment can cause serious deterioration of the overall performance, for instance:

6.1.4 Time delay and sampling errors

These difficulties are of a fundamental nature due to a sequential operation principle of the mini computer. The signal shown in Figure 41 cannot be transferred in a continuous manner by the minicomputer. First there is the restriction of the m.c. itself, i.e. that it can sample only one point at a time. Then comes the restrictions introduced due to time sharing of some parts of the interface, such as the A/D converter. Therefore, a 'sampling' process is needed whereby the mini computer periodically interrogates, under programme control, the necessary variables. Obviously this is a time consuming process for the minicomputer.

Another method that can be used is to interrupt the mini computer when the data is ready to be processed such as when an analog voltage crosses a threshold. This can be implemented through an interrupt technique.

Between the sampling of a variable and introducing a corresponding change, time delay is required, called the response time. This becomes significant when fast response times are expected from the system.

6.2. Design of plotter's general interface relate to NOVA 1220

In the Figure 42 is shown complete diagram of plotter's general interface.

6.2.1 The address part

The plotter must decide the plotters selection lines to generate a select level that ensures that only the single addressed plotter responds to the program. In the Figure 43 is shown the address part. The signals DS0 to DS5 (Device selection) are placed from the processor as the device code. In the instruction word these are bits 10 - 15. The

lines select one of 62 devices (code 01 - 76) that may be connected to the bus. Only the selected device responds to control signals generated during the instruction.

Decoding is performed by a NAND circuit, but for the reason that the device selection lines provide only one polarity (in this case it is low level), the inputs to the gate must be inverted for all device code bits that are high level, say 1's.

In this particular case the address for plotter is performed in binary as $100100_{(2)}$ corresponding 36 in decimal code.

6.2.2 The Data

All data which the plotter needed are transferred between the bus and plotter's interface via these seven lines, see Figure 44.

For programmed output the processor places the accumulator specified by the instruction on the data lines and generates DATOA, see diagram in Figure 42, to load the data from the lines into the corresponding latch buffer in the device selected by DS0 - DS5 then the strobe pulse is high.

6.2.3 The controlling network

This network specifies the state of the device and requests interrupts in containing four flip-flop circuits 7474, that is INT,REQ,INT, DISABLE, BUSY and DONE, see Figure 42 and Table 2.

The Input-Output reset, generated by the processor, clears all of these flip-flops directly. Signals generated by the control function part of an Input-Output instruction place the flip-flops only if the device (plotter) has recognised its code or the address lines.

The clear pulse clears all except INT, DISABLE, the start pulse clear DONE and INT.REQ fliplflops, but sets BUSY to place the device in operation. When the device is selected, the states of BUSY and DONE

are placed on the SELB (selected busy) and SELD (selected done) lines, see Figure 42.

The flip-flops INT,DIS is controlled exclusively by a particular bit MASK in MSKO (MASK out, generated by the procedure during the MSKO instruction).

Once DONE has been set, and provided INT,DISABLE is clear, the leading edge if the next RQENB signal from the processors sets INT REQ, whose high (1) state puts the INTR request signal on the bus.

RQENB is generated in every processor cycle, and as soon as either INT,DISABLE is set or DONE is cleared, the next RQENB clears INT,REQ, running the request.

Because of the serial nature of the priority determining signal on the bus, it is important that the request signals be synchronised by the processor. Hence DONE must not generate INTR directly. Moreover INT REQ must change state only on the leading edge of RQENB. In order to ensure sufficient time for request acknowledgement to work properly.

6.3. Timing diagram In-Out instructions

Throughout the duration of any input-output instruction, the processor holds the device code on the device address lines (DSO-DS5) for decoding by the device, see Figure 45 timing diagram in-out instructions.

6.3.1 Data In

The processor generates DATIA (data in accummulation) to place the corresponding buffers on the data lines in the device selected by DSO-5. At the end of DATI level the processor starts the data into accummulation selected by the instruction. Following the transfer the processor generates the pulse for start, clear.

6.3.2 Data Out

While the processor places the accummulator selected by the instruction on the data lines, it generates DATOA (data out) to load the data from the lines into the corresponding buffer in the device selected by DSO-DS5.

When using a mask to set the device priorities for the programme interrupt, the processor executes the same sequence as for data output but generates MSKO (in place of a DATO pulse) to set up the Interrupt Disable flags in all devices according to the information on the data lines.

6.4. Interrupt timing diagram

There are three cases that must be mentioned here, see Figure 46, the interrupt request, interrupt acknowledgement (device indentification and flag clearing). When a device completes an operation it sets DONE. In every cycle the processor generates RQENB which places the interrupt request signal INTR on the bus from a given device if its DONE flag is set and its Interrupt Disable flag is clear. The leading edge of RQENB must be used to set INT REQ to ensure sufficient time for the serial INTP function to settle down before the processor attempts to discover which device has priority.

6.5. External Bus Cable Terminator

Cabling from the back panel to external equipment is made via connectors at the back of the unit. In the NOVA 1220 the rear end of the back panel is the IO bus connector.

When an external bus is connected, signal that originates in the processor and drive devices must be terminated in the manner shown in

the figure 47 $\overline{\text{DCHR}}$ (data channel request) to OVFLO (overflow) according to table 3.

The bus system is designed for a maximum length of 50 feet including signal path length within devices and inside the processor. A bus line within a device may be a single wire if it runs less than 9 inches from the IO Connector. For greater distances is good practice to run twisted pairs from the input connector to the receiver circuits. The threshold noise level can be improved substantially by using the filter circuit at the input to the board on the signals indicated by a dagger (†) in the table 3.

CHAPTER VII

Experimental Results

7. Introduction

This chapter is somewhat the continuation of chapter 5 in which is already given the description of the transmitter and the receiver. At this point we shall consider the importance and construction of the other electronic circuits which are used in the purpose of adjustment and perfection of whole system. Here also, are given timing diagrams of the electroplotter and interface which are important for the connection of the interface and the particular machine.

The structure and performances of the two main parts of this system, the mini-computer Nova 1220 and Benson- Lehner Electroplotter are given also.

7.1. Mini-Computer

Mini-computers first appeared about the beginning of the sixties. Their progress followed closely the development of digital circuit technology. To illustrate the rate of expansion in 1969 about 8000 machines were installed. In 1975 this figure is expected to reach 40000 installation.

The mini-computer applications cover a wide range such as instrumentation of intensive care units, nuclear reactor control, spectroscopy, machine tool control, data handling, business applications, telecommunications and so on.

7.1.1 The Hardware of Mini-Computers

7.1.2 How a Mini-Computer System is Organized

There are three types of device in a computer system. These are shown in Figure 48.

The central processor unit (CPU) controls the operation of the system and performs arithmetic and logical computations. Memory is used to store data and the programs which instruct the CPU. Data is transferred between the CPU and peripheral devices. Each peripheral device is connected to the CPU by an interface which converts the electrical connections of the peripheral to a form compatible with the CPU.

The computer system is shown in more details in Figure 49. The CPU communicates with peripherals in two ways. Firstly data can be transferred along a set of wires referred to as the 'input/output bus'. This bus connects to interfaces and is used for the majority of peripheral devices, e.g. paper tape readers/punches, card reader/punches, line printers, digital plotters, machines tools and visual displays. The second method of transferring data is directly between a peripheral and memory, independent of the CPU. This method of 'direct memory access (DMA) is most often used by high speed peripherals such as magnetic tape, drum disc backing stores.

7.1.3 Central Processor

The central processor is the control unit for the entire system, it contains three basic elements, these are the control, arithmetic and memory transfer units. Each of these contain a number of registers. The control unit directs data transfers within the CPU and controls the operations performed within the registers.

The arithmetic unit is used for arithmetic and logical operations

of data.

The memory transfer unit controls all operations between memory and the CPU. A schematic layout of the organisation of the main CPU registers is shown in the Figure 50.

Timing of the operations within the processor is governed by an electronic clock which produces a pulse at regular intervals. The time between the clock pulses is the processor cycle time. At the beginning of the processor cycle the contents of the location specified by the program counter are loaded into the instruction register via the memory buffer register. The instruction is decoded and the operation performed within the arithmetic unit. For instance, the time to execute an add instruction may be 2.5 n sec where the memory cycle time is 100 nano second.

7.1.4 The Word

The CPU and memory operate on 'words' of data, i.e., a number of bits. In the particular 1220 Nova mini-computer, see Figure 51, the processor handles words of sixteen bits, which are stored in a memory with a maximum capacity of 32768 words.

The bits of word are numbered 0 to 15, left to right, see Figure 52 as are the bits in the registers that handle the words. Registers that hold addresses are fifteen bits.

Words are used either as computer instructions in a program, as addresses, as operands, i.e. data for the program.

Almost every computer has its instructions represented in a different format. However, there are three basic classes of instruction format

Memory reference

Register operate

Input/Output

A memory reference instruction, as its name implies, references memory during the instruction execution.

The mini-computer Nova 1220 use four basic formats from instruction words, which are shown in Figure 52. The type of instruction format depends on the values of bits in the instruction word; for instance if the first three bits are '0' it is 'jump and modify', but if bit one is 0, and second and third is 1, they specify 'in - out instruction'.

The transfer takes place between the accumulator addressed by bits 3 and 4 and the device selected by bits and the device selected by bits 10 - 15. Bits 8 and 9 of the function part specify an action to be performed, such as starting the device.

7.1.5 The Memory

The medium in which data or program is stored is called a 'memory'. The memory can be divided into two basic categories,'read-write'memory of which the contents can be changed by program and'read only'memory (ROM) in which the contents cannot be changed by program.

The conventional type of read-write is Ferite core stored although semiconductors stores are now being used. 'Read only'memory is often a diode matrix but can also be Ferite core store.

7.1.6 Input-Output

The usual method of transferring data between a peripheral device and the CPU is programmed input-output. It takes place along the input-output bus which in the Nova 1220 is a set of 50 parallel wires, see Table 3. The bus consists of sixteen bi-directional data lines, six device selection lines; nineteen control lines from processor to devices; six control lines from devices to processor. A signal on a control line specifies a particular function and supplies all timing information needed for the execution of that function, see Figure 53 which shows a

Nova System configuration.

7.1.7 Bus Signals

The binary signals on the bus have two states, low and high, which correspond respectively to nominal voltage levels or 0 and +2.7 volts. Any level between ground and 0.4 volts is interpreted as low, any level more positive than 2.2 volts is interpreted as high.

The level listed for a signal in Table 3 is the voltage level on the line when the signal represents a 1, or produces the indicated function.

7.2. The Action of Interrupt

The majority of input-output devices transfer data at a rate significantly slower than the operating speed of the CPU. The running time of an input-output routine is mainly spent waiting for the device to accept or transmit a data word. The utilization of the CPU can be increased so that it can continue with another task until the device is ready. This is possible using the interrupt facility available with all mini-computers.

A device interrupt, transmitted when the device is ready, causes the CPU to stop its present task and begin a special interrupt program. This program discovers which device caused the interrupt and transfers control to the appropriate device service routine.

A typical automatic hardware interrupt priority system is shown in Figure 54.

7.2.1 The Level of the Interrupts System

In this system four levels of interrupt are defined. Normally any number of devices can be connected to any one of these levels.

7.2.2 Mask Register

This is used to select, by program, those interrupts that should be enabled at any one time. The mask register can be set up by an instruction, and can be used to reallocate priority by program.

7.2.3 Priority Logic

Priority logic is used to establish which should be the first interrupt to be accepted by the CPU.

7.2.4 Address Logic

Each interrupt level is assigned as a special location in memory, to which program control is passed when the CPU accepts an interrupt.

7.2.5 Interrupt Enable-Disable

Used to enable or disable the whole interrupt system. It is under the control of the program. The interrupt timing diagram is shown in Figure 46, Chapter 6.

7.3. The Electroplotter

The Benson-Lehner electroplotter is an automatic graph plotter, using 5 holes Elliott 803 Code punched tape, see Figure 55. The plotting area is 30" x 30". If the plotting head is outside this area, the servodrive to the head is switched off automatically when the plotting table limit is reached. Should this occur it is necessary to clear the store before plotting can recommence.

In the input format are groups of information separated by the carriage return symbol. The group (line) can contain up to 99 words, each separated by at least one space. Each word may consist of between 4 and 13 digits, preceded by a mathematical sign. In operation any two words, and the most significant digit required in each, are selected normally on the plotter. A signed four digit block is used, the decimal

point not being permissible.

7.3.1 The Block Diagram

Figure 56 illustrates in block diagram from the sequence of operations concerned with the plotting mechanism. The operations are as follows:

1. A block in formation from an X-word and a Y-word is read via the tape reader and word selector into the store.
2. The block is processed according to certain instructions which are set externally by hand on the plotter console, are contained on the tape.
3. The processed information is converted to an analogue voltage.
4. Once the command position has been reached the next block of information is released from the store processor unit on to the servomotor and the next block of information on the tape is read into the store from the tape reader.

This sequence continues until STOP instructions are reached. In 803 code this instruction is '/' (slash).

7.3.2 Processing of numeric information for acceptance by the Plotter

For its operation it is necessary that co-ordinates X and Y be presented in a manner which will enable it to function correctly.

To be acceptable to the plotter the information must be arranged on the tape in the manner thus -

BLOCK 1	Sign (+ or -)
	X1 WORD
	SPACE (AT LEAST ONE)
	Y1 WORD
	CRLF

	SIGN (+ or -)
	X2 WORD
BLOCK 2	SPACE
	Y2 WORD
	CRLF
	SIGN (+ or -)

Shown here are two separate blocks of information which would cause the plotter to draw a line from X1, Y1 to X2, Y2.

7.4. The Output Signal from the Tape Reader

Electrical signal outputs from the reader which correspond to perforations in the tape, consists of negative - going pulses of uniform amplitude nominally equal to the negative supply voltage. Figure 57 shows five outputs point A, B, C, E, F which correspond to five holes on the paper tape. Point D corresponds to sprocket holes and other connections are "clutch", "tape run" like different input/output voltages, there is also negative supply voltage 6V to the data circuits, less - to be precise - the drop of about 0.2V in the fully conducting transistors. This 6V is the value of amplitude of the output pulses. This amplitude corresponds to the value of the voltage which performs when holes of data are read from very sensitive photo-sensors circuits. The output impedance is essentially a resistance of 820ohm, which forms the collector load of the output transistor.

7.4.1 Types of Signals

There are two types of output signals, corresponding to the two types of holes in the tape:

a) data holes

b) sprocket holes

The form of the signals from the photo-sensors is determined by the physical form of the punched holes and light mask.

The sprocket holes are used to generate "strobe pulses", which are of short duration, Figure 57 shows that duration is 5 msec.

The sprocket pulses aretimed by suitable circuits to give the most reliable samples of signals from the data holes. This accuracy is required because the paper tape reader is replaced by the interface electronic circuits. The strobe pulses derived from the sprocket holes may be used for mechanical positioning data holes. We can therefore translate on electrical language as "synchronisation" of data pulses by sprocket. It is necessary that the pulses ratio is so important.

The normal mark to space ratio for the sprocket holes is 10 to 30 and for the data signals 47 to 53, see Figures 58 and 59. So, if full cycle of the sprocket pulse is T = 5 msec the parts are, T1 = 3.5 msec and T2 = 1.5 msec, these times being important for the circuits which are already built in the electroplotter hardware.

7.5. The Timing Diagram

The plotter will recognise data and control information punched on 5 holes tape in standard Elliott803 code. The X, Y co-ordinate data can each be a signed number of up to 13 digits of which 4 are used. The X, Y data being separated by at least two spaces. Carriage return being end of X, Y information. 200 characters a second the period of a data cycle would be :

$T = \frac{1}{200} = 5$ msec. This time corresponds to the sprocket repetitive frequency. On the Figure 60 is the timing diagram which shows relation between tape by programme, the sprocket strobe and the digit pulses,

is to establish synchronisation between data pulses and sprocket strobe pulses. The paper tape reader, reads character by character in a serial manner. The character consists of from one to five traces A - E. The character is read in parallel, see Figure 61.

7.6. The Clutch's problem

The clutch assembly consists of an electromagnetic which is connected to a pinch roller, so when the magnet is energised the pinch roller rises and presses the tape against the rotating capstan. This mechanical action created electrical, that is, without moving the paper tape, output voltage at 'Clutch' point is - 13V, by moving is 0V. This change of voltage causes some action in hardware of plotter and makes sure correct work of inner sequential logic.

The action of 'Clutch' and action of 'Start' button activates the plotter. To ensure the same conditions at the interface part, that is, the clutch signal ought to start transmitter and receiver synchronously by plotter. For this purpose clutch point from the plotter is connected by one wire via electronic circuit, say, "Clutch's circuit" which can change the level of D.C voltage from - 13V (plotter side) to 0 volt (receiver-transmitter side), and from 0 volt (plotter side) to +5 volt (receiver-transmitter side), see Figure 62a.

7.6.1 The Clutch Circuit and the connection with the Transmitter and Receiver

This circuit consists of two transistors, see Figure 62b, of the commonemitter configuration. The first transistor is a P-n-P transistor which serves for negative logic. If the input is -13V (Vi = -13V), then the parameters are chosen so that the first P-N-P transistor is in saturation and then, through the resistor between collector of PNP and base of NPN transistors, the voltage on the base of NPN is sufficiently

positive (+o.6V) to saturate it, that is, output is zero voltage.

Meanwhile, if input is zero volt the PNP transistor is off, the base of second NPN transistor is on negative voltage. So that the second transistor is off, that is, output is +5V. This output is connected to the input 1B of 74 123 circuit which belongs in the Figure 63a memory and counter group. See Figure 2. The circuit 74 123 is dual monostable multivibrators and here is used as generator, which performs the pulses of the same duration as sprocket pulses. The function table of particular circuit shows that, if state of input 1A is in transition from high to low and at the same time input 1B is on high level, the output 1Q produces the pulse which frequency depends the time constant of the circuit.

This connection start and stop the 74 123 circuit, ensures synchronisation between sprocket pulses and contents in the memory, that is, the data on a particular program which is stored in the memory.

7.7. The Circuit for correct timing of the Sprocket pulses

The Figure 63b shows interconnection of the circuits which perform correct timing of the sprocket pulses at the receiver side, by using the clutch's pulse voltage.

The clutch's circuit from the Figure 62b is used in this configuration to give at the output of transistor BFY52 two states, that is, zero output and +5V output, the +5 volt corresponds with the state logical one. Both of these states are dependent of the clutch voltage. The mentioned output is connected at the input, 2, of NAND circuit 7400. The pulse named 'lock' is applied at input 1 of the same NAND circuit. The 'lock' pulse is always present during the action of the transmitter and then the level is logical one. The output of this gate changes between logical zero and one and it is connected to the other NAND gate at

pin 4. The permanent positive +3 voltage, which represent logical level one, is applied at the input of the pin 5, so, the output level, which is generated at pin 6, depends on the state on the input 4. This output is used as the input to the circuit 74 123 at pin 9, which creates the sprocket pulses.

These two last configurations enabled the simultaneous start of the action of transmitter and receiver with the action of the plotter. At the same time plotter requires synchronisation, during the reading data, between sprockets and data holes and the circuit from Figure 62b performs that which is already described.

7.8. The Delay circuit which performs correct timing between Data Pulses and Sprockets

In the same chapter under the title, 'The Timing Diagram' is shown already the relation between data holes and sprocket hole and it is noticed that the sprocket's pulses synchronise the reading of the data. In Figure 64 is shown the oscillogram taken at the output behind the tape reader of the plotter, with the pulses of data and sprocket. In the diagram the time delay from the loading edge of sprocket pulses can be seen, which in this partiticular case is between 0, 3 and 0, 4 milisecond, see Figure 65a. The diagram in Figure 65a shows how the time delay circuit is designed. The loading edge of the data pulse triggers the first multivibrator of the 74123 circuit. The sprocket pulses are applied at the input B1 to the AND gate, see Figure 65b, the input A is connected to the ground and through the NOT circuit gives always logical level one. According to the truth table for 74 123 circuit the output Q_1 is enabled during the transition of the data pulses from low to high level. The time constant t_1 is calculated to produce

necessary time delay of 0,3 milisecond. The output Q_1 is connected through the NOT circuit to the other multivibrator in the same 74 123 circuit.

The time constant T responds to the width data pulses. The lugging edge of the delay pulse triggers the second multivibrator when transition from high to low is made. The output Q_2 produces the data pulses.

7.9. The Calculation of the Time Constant of the Sprocket Generator

For sprocket pulses areused second monostable circuit and output 2Q performs the sprocket pulses. The Figure 66 shows electrical interconnection of this circuit. All the connections were made according to the function table.

The output pulse is a function of the external capacitor and resistor. For $C_{ext_2} >$ 1000 PF, the output pulse width (tw) is defined as:

$$tw = K\, R_{ext}\, C_{ext}\quad (1+\ \frac{0.7}{Rext})$$

where:

Rext is in KOHMA

Cext is in PF

tw is in nanosecond

K is 0.28 for 74 123

Above formula is given in the TTL Data Book - Texas Instruments. The diagram in the same book shows that is not allowed to use the resistor under five kiloohm. By using the shown formula it wasn't possible to get expected width of the sprocket pulses. To get desired time constant T, one trimmer potentiometer of 50K is connected in the serial way with five kiloohms resistor. The value of capacitor

is 1 microfared.

7.10. The Circuit which performs correct timing of Data Pulses with Lock (Enable) pulse.

The design in Figure 67 is used to perform correct timing of the data, two 7475 latch circuits with pulse called 'lock' enable parallel output of data. The timing is achieved by 'lock' pulse. The eight data from these latches circuits are applied to the eight input gates of two '7408 AND' circuits as shown in Figure 67. The other input gates of the same two AND circuits are connected to each other in a common link. This common link is connected to the output 1Q of the monostable circuit 74 123. The monostable produces always output when the lock pulses appear in the input 1B of the circuit 74 123. The duration of the output pulses is equal to the width of the data pulses. In such a way the 'lock' pulse makes precise timing of data output through the 7408 circuit from D_1 to D_8. The 'lock' pulse is at the end of every group of 21 clock pulses, see Figure 68, where particular data are located. These eight data outputs are connected to the 'Ascii' input of the decoder, which is already described in the Chapter 5.

7.11. The System Grounding

To obtain the best performance a good system ground is required. All grounds should be connected to a common ground point, normally near to the power supply. All logic circuits are connected to a ground circuit. In the case there are relays, motors and other noise generating devices all are wired to a separate ground network connected to the common ground point. Standard noise suppression techniques should be supplied, i.e. diodes across relays, and capacitors across dc motor

brushes. All mechanical parts such as panels, cabinet doors and chassis should be grounded with a third ground. A mounting frame is often used for this if good conduction can be made at points of contact.

If some pieces of equipment in the system are left ungrounded they may carry transient voltages that will interfere with the rest of system. In the Figure 69 the three separate ground systems connected to a common point are shown which will eliminate noise on the ground signal. Heavy ground loads should be used on large systems to minimize any voltage drop along the ground line.

7.12. Simulation and Programming

The program used in the simulation technique is of the same pattern as the one used from the tape reader.

Before starting programming by switches, the electroplotter must be set to origin. All 'Origin offset' and 'Data' switches should be set to zero. 'Clear, 'Origin' and 'Servo on' switches should be pressed. Note that there is a one minute delay for the servo-power after the electroplotter is first switched on.

If, after pressing the 'start' switch the plotter fails to operate, the 'Clear' switch may have not been previously operated.

For every new programme it must be started by 'Figure shift', 'New line' and then X-blcok by signs + or - as shown in Figure .

The plotter identifies 'words' i.e. numbers by counting from each new line. Therefore new line must be used at the end of each point or a set of values, but nowhere else.

There must be no decimal point. The full number of digits must appear every time.

Each word in a group represents a different variable and may be plotted to a different scale and different offset to any other word.

On the figures 73, 74, 75, 76, 77 and 78, are given six different geometrical pictures together with their associated programmes. Table 4 shows the representation of Elliot 803 code, that is, how characters are represented by position of the switches.

7.12.1. Manipulation

Correct operation of the system interface is obtained by the following steps:

1. Switch on plus five and minus six volts supplies.
2. With the switch S_1, 'Reset', set electronics to zero position.
3. Set the 'Write-Read' switch to 'Write'position.
4. On the 'Board of simulated switches from computer', see Figure 1, start by programming with 'Data switches', from '10' to '15'. The other switches ('Addresses' and 'Instructions') stay in the same positions all the time.
5. With the switch 'Memory in', see Figure 2, load a particular data into the memory.
6. When the programme is completed, return to the first address at the memory using the switch 'Reset' and with the switch 'Write-Read' set in the 'Read' position and using the 'Start' button on the plotter command table start with the execution of programme.

Figure 79 shows the photograph of interface components.

CHAPTER VIII

8.1. The Structure of the Central System

If one mini-computer can control more than sixty different machines which are interconnected by various types of interfaces to it, then, this system can be called 'multi-system'. One of the most important and complex tasks of governing by the 'multi-system' is to determine the radical structure of the observing system. When such a system is considered, then at first sight it seems justifiable to create a system which makes the control from one 'central point'. In such a system the complete information about the state of all the machines under the control and which feed back information, comes from the central point, for instance a mini-computer. The central point according to the information received about the state of the system produces the control action for each of the machines in the system. For inexperienced people such structure of the system would appear to be ideal because in this case the total information about the system is stored at one central point, but still exists the possibility of the most corrective calculations for the effective criterion and finding the solution which ensures the optimal control in action. Meanwhile in reality it is not easy to materialize such a system.

To perform an effective control, only for one object in the system it is necessary to obtain and feed in sufficient information, but for systems which comprise large number of objects the quantity of information proportionally increases. For this reason it would be necessary in the central control point to assemble a large amount of varied information and ensure its modification. In this case it is very difficult

to realise the target of the observing system, that is, to perform the working optimum regime of the system.

In the system with 'central point' the task of finding out the working optimum regime ought to be solved in the space of rather larger dimensions which, in practice, results in even greater computation difficulties. Even if it is possible to find the best working regime of the system, the time spent will have been excessive and the control commands gained with much delay.

8.2. Some more Imperfections of the Central Control Point

The system consisting of the central control point and an observing structure, has a very large inflexibility, because due to its adaption to the modifications which happen not in the particular parts of the observing structure but in the central control point, say, mini-computer. Also the reliability of the function decreases in this system. The errors occuring during the working of the system becomes difficult to correct. The amount of correction depends on the state of the system.

Thus the conclusion is, that the system with the central control point is unfavourable in reference to other available systems.

8.3. The Hierarchical Structure

The imperfections indicated in the system with the central control point can be considerably overcome by the hierarchical structure of the central system. A characteristic pecularity of the hierarchical structure is the successive division of the system up to (upon) the subsystems, between which are established the co-ordinations.

The central device on the higher level governs the large units of the system from which every one has their own control part. Each of

these units is divided into smaller sections, which are also equipped with corresponding control parts and so on up to the elementary part of the observing system, whose continued division into sections is without purpose. The essential characteristics which every hierarchy has is vertical arrangement of subsystems which comprise the overall system, priority of action or right of intervention of the higher level subsystems, and dependence of the higher level subsystems upon actual performance of the lower levels.

8.4. Vertical Arrangement

The hierarchy contains a vertical arrangement of subsystems, and then the system it seems is consisting of a family of interacting subsystems as shown in Figure 70. The terms 'system' and 'subsystem' means a transformation of the input data into outputs. If the transformation is performed in actual time, the system is 'on-line system', if all operations are not performed simultaneously, with the operations of other blocks, then it is 'off-line system'. Figure 71 shows a four level diagram of a hierarchy where the lowest level unit IV need to solve relatively simple tasks, that is, they respond to the devices with limited capacity of the modification of informations.

The task of the next higher level, in figure 2 are units III, is to solve the problems which are only indispensible (necessary) for reciprocal adjustment of elementary units, that is, they do not require in detail the information about controlling objects.

The same consideration is related to all higher levels in more general form, so that the unit on the first level (unit I) receive the most concrete (particular) information about the controlling system.

8.5. Right of Intervention

The operation of a subsystem on any level is influenced directly and explicitly from the higher levels, most often from the immediately superceding (higher) level. This influence is binding on the lower levels and reflects a priority of importance in the action and targets of the higher levels, so this influence will be referred to as intervention.

In on-line systems, intervention usually appears in the form of changing parameters in the lower level subsystems. In off-line applications, the problem or solution on any level depends upon the solution of the problem on the higher level. The problem on the lower level is well defined only after the higher level problem is solved.

The important characteristic of multi-level systems is that the higher level unit do not completely control the activities of the lower level units. The lower level decision units have to be given some freedom of action to select their own decision variables. These decisions might be, but are not necessarily, the ones which the higher level unit would select.

It is essential for the effective usage of the multilevel structure that the decision units be given a freedom of action. A suitable division of decision-making efford amond the units on different levels should be established; only then can the existence of the hierarchy be justified, see Figure 72.

CHAPTER IX

Conclusion and Future Work

9.1. Conclusions

9.1.1 The design of a system interface which connects an on-line computer to an NC machine tool, for example, an electro-graph plotter, is feasible and such a system is practicable.

9.1.2 In principle, the design of a system interface depends upon the control system of the NC machine tool.

9.1.3 The advantages of the design system are:

a) Two wires only, or
telephone line or
coaxial cable

b) The distance between computer and electroplotter is not very important, because the carrier frequency is in simsoidal form.

c) The input signal at the receiver side of 200mv is sufficient for the receiver to function.

d) If a word site exceeding 16 bit (say 1024) needs to be used, only the frequency of the master oscillator has to be changed.

e) Using a 'TUNNEL' diode as oscillator can extend the range of transmission speed to the megacycle region.

9.1.4 A conventional tape reader NC machine tool can be converted to operate from an on-line computer by a connection behind the tape reader.

9.1.5 The development of system interface will largely depend on the availability of new electronic components, particularly the use of microprocessors, in interface hardware.

9.1.6 The economics of a system interface must be carefully considered. This problem is unresolved at present; no satisfactory answers have emerged

despite the large number of papers dealing with economic considerations.

9.1.7 The design of a system interface, both the hardware and the programming will require teamwork and will not be achieved by one man's effort.

9.1.8 In this particular system interface TTL logic form was used for the electronic circuitry, which is more flexible, compared to DTL technique.

9.2. Future Work

9.2.1 Efforts should now be made to link system interfacing further inside the hardware of the machine. An adequate investigation at this problem could cut some expensive hardware in the machine, and purchasers' expenses could be considerably reduced.

9.2.2 An inherent characteristic of the design approach was the lack of flexibility of the system architecture due to the use of hardwired techniques. To improve flexibility a microprocessor could be considered.

9.2.3 In future work, the use of a microprocessor as a basic element of the system interface can give the following advantages:

a) The designer may use the same component at each site and simply change the programme or Read-Only memory.

b) The standardisation provides faster design time, is more economical, enables the system to adapt easily to changing requirements and requires less maintenance and documentation.

c) Parallel operations such as data acquisition are done at the terminals while the master computer manages and analyses the current data.

d) The new terminals may be added without disturbing the rest of the system.

9.2.4 By employing microprocessors in the machine interface programme control is simplified. Displays are tailored to the process which can then be easier to follow.

The operator may select the data and machines to be displayed in the format he desires. The data may be displayed cyclically, selectively or in entirety.

9.2.5 By using microprocessors the system hierarchy becomes a great deal more flexible. The system consists of computer, interface and machines of various degrees of significance.

REFERENCES

1. Korn, A. G. — Digital computer interface systems. Simulation Vol 11, pp 285 - 287, December 1968.
2. Kinter, S. M. — Interfacing a control computer with control device. Control Eng. Vol 16, pp 97 - 99, November 1969.
3. Spur, M and Wentz, W. — A system for direct numerical control of machine tools. M.T.D.E. Conference, Manchester, pp 421-425, 1972.
4. Evans, L. — Computerised Numerical Control Revised. Control Eng. pp 33 - 36, August 1972.
5. Wightman, E. J. — What are the benefits of computer numerical control. Engineering Digest, pp 45 -50, June 1972.
6. Wightman, E. J. — Computer control systems for machine tools. Mech. Eng. CME, pp 49 - 50, March 1974.
7. Andreiev, N. — Easy to use computer control system for small processes. Control Engineering pp 27 - 29, April 1973.
8. Crossley, T.R. and Lewis, W. D. — A DNC Prototype, Plessey, pp N/1-N/6, 1974.
9. Milner, D. and Oliver, A.F. — Computer involvement in numerical control of machine tools. The Production Engineer, July/August 1974, pp 240 - 247.
10. Milner, D. A. — Computers in production technology and management. Engineer's Digest, September 1974.
11. Elgerd, O.L. — Control Systems Theory. McGraw-Hill Kogakusha Ltd. pp 495 - 530, 1967.
12. Wightman, E.J. — Developments in computer control systems for machine tools. Proc. I. Mech. Eng., Vol 188 No. 6, pp 49 - 75, 1974.

13. Lewin, D. — Theory and design of digital computers. London Nelson 1972.

14. Greensite, A.L. — Elements of Modern Control. Theory. Vol. 1 New York : Spartan, 1970.

15. Langill, A.W. — Automatic Control systems engineering. Vol 1. Prentice-Hall, 1965.

16. Noton, A.R.M. — Modern control engineering. New York, Oxford. Pergamon Press, 1972.

17. Flores, I. — Computer Design. Prentice-Hall, pp 121 - 135. 1967.

18. Peklenic, J. — Advances in manufacturing systems. Research and development, Ed. J. Peklenic, Oxford. Pergamon 1971.

19. Ledgerwood, B.K. — Control engineering manual. New York, McGraw-Hill, 1957.

20. Dedoo, G.J. — Integrated circuits and semi-conductor devices. McGraw-Hill, 1971.

21. Johnson, E.F. — Automatic process control. McGraw-Hill, 1967.

22. Savas, E.S. — Computer control of industrial processes. McGraw-Hill, 1965.

23. Shinskey, F.G. — Process control systems. McGraw-Hill, 1967.

24. Heath, F.G. — Digital computer design. Edinburgh : Olivery & Boyd, 1969.

25. Harrison, H.L. and Bollinger, J.G. — Introduction to automatic controls. 2nd ed. Scranton (Pennsylvania) Intertex books 1969.

26. Hurst, S.L. — Threshold logic - An engineering survey. London : Mills & Boon, 1971.

27. Williams, G. — Boolean algebra with computer application. New York, London : McGraw-Hill, 1970.

28. Maddoxk, R.J. — Intermediate electronics - Book 2. London : Butterworths, 1970.

29. Hovnessian, S.A. and Pipes L.A. — Digital computer methods in engineering. McGraw-Hill, 1960.

30. Seely, S. and Tarnoff, N.H. and Holstein, D. — Digital computers, engineering. New York, London : Holt, Rinhart and Winston, 1970.

31. Murphy, G.J. — Basic automatic control theory. Princeton (N.J.), Van Nostrand, 1957.

32. Milman, J. and Halkias, C.C. — Integrated electronics : Analog and digital circuits and systems. New York, (Maidenhead) : McGraw-Hill, 1972.

33. Morris, N.M. — Logic circuits. London : McGraw-Hill 1969.

34. Arrowsmith, J.R. — An introduction to numerical control of machine tools. London :Chapman & Hall 1964.

35. I.B.M. Shapiro, H.S. and Slotnick, O.L. — Journal of Research and Development. Vol 3 - 1959.

36. A.I.E.E. Transactions - Part II — Applications and Industry : Vol. 81.

37. Eveleight, V.W. — Adaptive control and optimization techniques. London :McGraw-Hill 1967.

38. Davenport, W.P. — Modern data communication. Hayden Book Co. Inc. 1972.

39. Shannon, C.E. and Weaver, W. — The mathematical theory of communication. Urbana University of Illiniois P. 1949.

40. Goodyear, C.C. — Signals and information. London : Butterworth, 1971.

41. Milner, A.D. — Adaptive control of feedrate in the milling process. Int. J. Mach. Tool. Pergamon Press, Vol. 14, pp 187 - 197, 1974.

42. Petterson, W.W. — Error - correctioning codes. The M.I.T. Press, Cambridge (Mass) 1961.

43. Mitrovich, D. — Servo sistemi i njhova primena u automatizaciji. Naucna Knjiga 1958 : Beograd, Jugoslavija.

44. Soucek, B. — Mini-computer in data processing and simulation. Wiley-Interscience 1972.

45. Milner, A.D. Some aspects of adaptive and computer control applied to a milline machine. A thesis for the degree of Doctor of Philosophy. Faculty of Engineering, pp 128 - 147, June 1974.

46. Marcus, A. Automatic industrial control, Prentice-Hall, 1966.

47. Loberg, A.G.H. Numerical control of machine tools, the present situation. Mach. & Prod. Eng. 30th May, 1973.

48. Stute, G. Computer applications in manufacturing systems. Prolamat Conference, Budapest, April 1973.

APPENDIX I

BOOLEAN ALGEBRA AND COMPUTER LOGIC

Boolean algebra has certain basic operations, just like any branch of mathematics. Boolean algebra uses binary digits 0 and 1 to define logical decisions.

OR Operation

OR operation may be defined by two integers I and J by the statement:

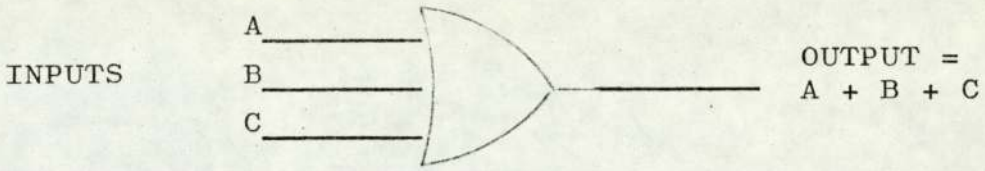
If I or J equals 1, then the result is 1. Otherwise the result is zero.

The plus sign is used to represent OR.

Truth Table for an OR gate

INPUTS		OUTPUTS = I + J
I	J	
0	0	0
0	1	1
1	0	1
1	1	1

The symbol for an OR gate is:



The output will be 1 (true) if any one of the inputs has the value of 1.

AND Operation

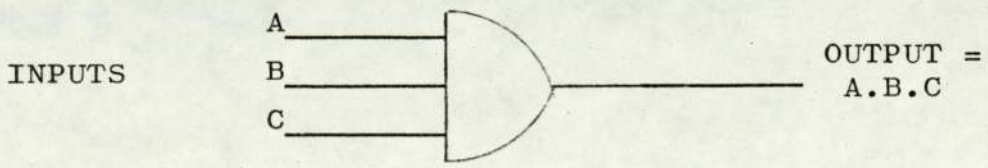
The AND operation, for the two integers I and J, may be defined by the statement:

If I and J are both 1, then the result is 1. Otherwise the result is zero. The dot is used to indicate the AND operation.

Truth Table for an AND gate

INPUTS		OUTPUT = I . J
I	J	
0	0	0
0	1	0
1	0	0
1	1	1

The symbol for an AND gate is:



The output will be 1 only if all the inputs have the value of 1.

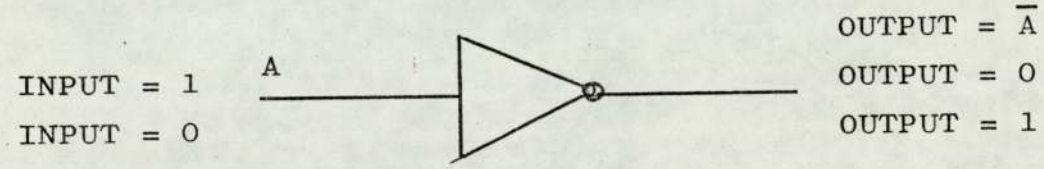
The NOT function

The NOT function specifies that the output will always be the inverse of the input

NOT 1 = 0
NOT 0 = 1

Inverter

The gate that produces the NOT function is called an inverter. The symbol for a NOT gate is:



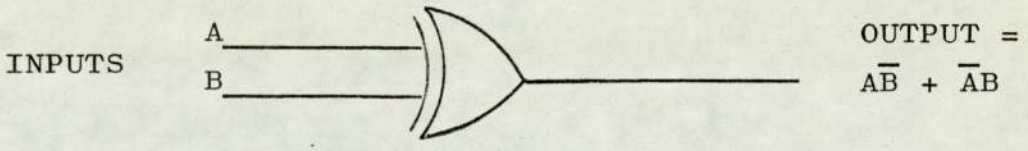
The EXCLUSIVE OR function

The EXCLUSIVE OR function differentiates between inputs which are identical and inputs which are different. The output value is 1 when the input terms are different and 0 when the inputs are the same.

Truth tables for two-input EXCLUSIVE OR gate

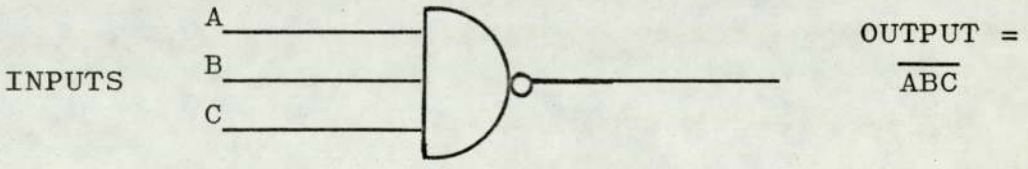
INPUTS		OUTPUTS =
A	B	$A\overline{B} + \overline{A}B$
0	0	0
0	1	1
1	0	1
1	1	0

The symbol for an EXCLUSIVE OR gate is



NAND and NOR Operations

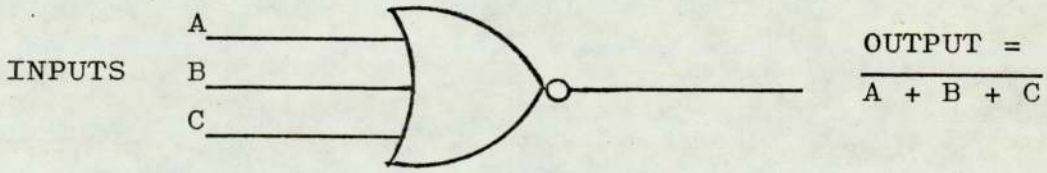
By combining an inverter with an AND gate or an OR gate, two additional logical operations are possible. The grouping of an AND gate followed by an inverter is called a NOT AND or NAND gate. The symbol for a NAND gate is pictured below



If all the inputs have a value of 1, the output is 0, and if any of the inputs have a value of 0, the output will be 1.

INPUTS			OUTPUT = $\overline{ABC}$
A	B	C	
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

The grouping of an OR gate followed by an inverter is called a NOT OR or NOR gate. The symbol for a NOR gate is



If any of the inputs have a value of 1, the output will be 0.

INPUTS			OUTPUT= $\overline{A+B+C}$
A	B	C	
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

DE MORGAN'S THEOREM

In computer design, logic gates are combined to produce any desired output from a set of known inputs. De Morgan's theorem is a valuable aid in designing such combinations. An elementary form of this theorem is:

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$

which states that the inverse (or NOT) of the combination A AND B is equivalent to the separated ORed inverses.

A second form of De Morgan's theorem is

$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

by judicious application of this theorem, logic equations can be put into an easily implemented form. Suppose, for example, it is desired to implement the equation:

$$F = (A.\overline{B}) + (\overline{A}.B)$$

entirely with NAND gates. Note that this is the EXCLUSIVE OR operation described earlier. First we negate both sides of the equation:

$$\overline{F} = \overline{(A.\overline{B}) + (\overline{A}.B)}$$

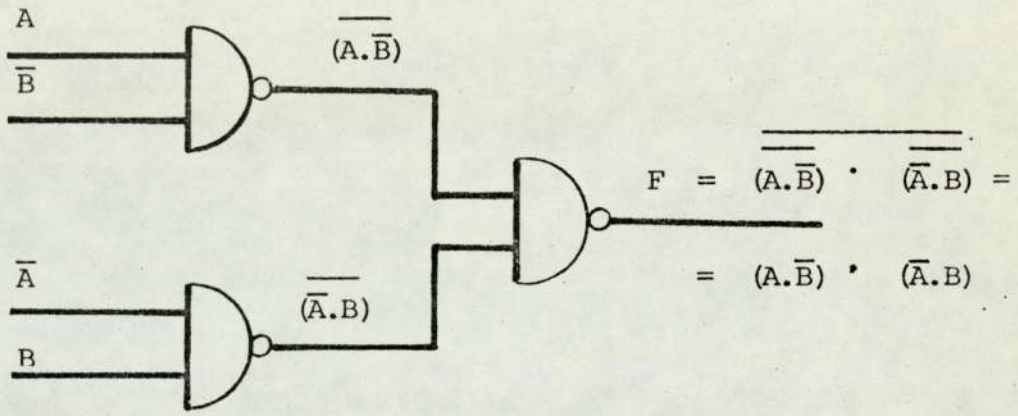
Then apply De Morgan's theorem:

$$\overline{F} = A.B \cdot \overline{(\overline{A}.B)}$$

The only logical operators left are ANDs (i.e. no ORs). In addition, all but one AND is covered by negation (or NOT). Negating both sides of the equations gives:

$$F = \overline{\overline{(A.\overline{B})} \; . \; \overline{(\overline{A}.B)}}$$

This equation is quite easy to implement with NAND gates. The corresponding circuit is:



Comparison of the circuit diagram with the last equation shows how a logic function can be built up a block at a time. There are no rules concerning when to use De Morgan's theorm. Each designer is left to his own ingenuity and experience.

APPENDIX II

Cyclic Redundancy Codes

Introduction

Cyclic redundancy codes are used in data transmission between the transmitter and the receiver. The purpose of this code is to realise error detection and correction of transmitting data. An understanding of the codes are based on a minimal mathematicl background.

Basic theory

Coding theory treats data mathematically in polynomial form. A polynomial is a summation of various powers of one or more variables, with a coefficient attached to each term in the summation, for instance

$$ax^3 + bx^2 + cx + d$$

and in this cyclic code, all coefficients are binary and there is only one variable x, which is present only to permit mathematical treatment. The coefficients a, b, c, d are the real data.

The highest power of x is attached to the most significant bit, while decreasing powers go in order with bits lesser significance. For instance, the binary number 1100110011 can be represented as a polynomial, with the left most element of the number the most significant:

$$x^9 + x^8 + x^5 + x^4 + x + 1$$

where the least significant bit is 0; thus, since $x^0 = 1$, and it is without an attached variable.

In particular, the binary case, all numbers are mapped into either 0 or 1. Since -1 is mapped into +1, addition and subtraction operations

are identical. In this fact lies all basic manipulations of binary cyclic codes.

Division of one polynomial, D(x), by another G(x), produces a quotient polynomial, Q(x), and usually a remainder polynomial, R(x), if the division is not exact.

$$\frac{D(x)}{G(x)} = Q(x) + \frac{R(x)}{G(x)}$$

or

$$D(x) = Q(x) \cdot G(x) + R(x)$$

See examples:

(a) $x^8 + x^5 + x^2 + 1 \;:\; x^5 + x^3 + x + 1 = x^3 + x + 1$

$x^3 + x^6 + x^4 + x^3$

$x^6 + x^5 + x^4 + x^3 + x^2 + x$

$x^6 + x^4 + x^2 + x$

$x^5 + x^3 + x + 1$

$x^5 + x^3 + x + 1$

$0 = R(x)$

$D(x) = (x^3 + x + 1)(x^5 + x^3 + x + 1) = Q(x) \cdot G(x)$

(b) $x^7 + x^3 + 1 \;:\; x^5 + x^3 + x + 1 = x^2 + 1$

$x^7 + x^5 + x^3 + x^2$

$x^5 + x^2 + 1$

$x^5 + x^3 + x + 1$

$x^3 + x^2 + x = R(x)$

$R(x) = (x^2 + 1)(x^5 + x^3 + x + 1) + (x^3 + x^2 + 1)$

$= Q(x)G(x) + R(x)$

The redundancy bits

A basic coding concept is to modify a polynomial that represents a train (number) of data so that is is exactly devisible by another polynomial, the divisor is called the generator polynomial G(x). This modification is done appending certain calculable bits, called redundancy bits, to the data train. These extra bits are essential in any error detecting and correcting code, because they distinnguish code words containing errors from other code words. Redundancy bits are determined by dividing the data polynomial by the generator G(x), polynomial and adding the resulting remainder to the original data polynomial. Since, in the binary arithmetic system, the sum and difference are the same, adding the remainder to the data is equivalent to subtracting it. In the equation

$$D(x) = Q(x) \cdot G(x) + r(X),$$

where the remainder, R(x) is substracted from both sides, the right side becomes a product of polynomials:

$$D(x) - R(x) = Q(x) \cdot G(x)$$

This product is divisible by G(x), since that is one of its factors. Therefore, the division and addition forms a new polynomial that is always exactly divisible by G(x). Such polynomials correspond to code words; the remainder comprises the redundancy bits.

If the complete polynomial code word to be transmitted or stored is designated C(x), then

$$C(x) = D(x) + R(x) = Q(x) \cdot G(x)$$

When data with redundancy bits attached, T(x), are received or fetched, they may be divided again by G(x). If the remainder is 0, all bits are assumed to have arrived unaltered. The error could be such that the disturbed polynomial is still divisible by G(x). This disturbance would constitute an undetectable error.

Any disturbances in the received polynomial can be represented by an error polynomial, E(x). When E(x) is subtracted from the received polynomial, Z(x), or equivalently added to it, the result is the sum of the correct data and redundancy polynomials:

$$T(x) = Z(x) - E(x) = Z(x) + E(x)$$

Obviously, any attempt at error correction requires that this error polynomial be found.

The Accomplishment of cyclic redundancy code

The circuits to perform the division may be implemented from standard logic components or are available in chip form. All such circuits follow the same basic principle, to produce remainder after division by G(x). Anything exactly divisible by G(x) is to 0, G(x), of course, is exactly divisible by itself, and so is component to 0. For instance, polynomial

$$G(x) = x^9 + x^6 + x^5 + x^4 + x^3 + 1.$$

Set g(x) = 0 and solve for x^9;

$$x^9 = x^6 + x^5 + x^4 + x^3 + 1.$$

Then, whenever x^9 occurs, it may be replaced by all lower order terms of the generator polynomial.

In a register of (m + 9) elements, each element represents a memory for the coefficient for one power of x, see figure a . If all register's elements are set to 0, and a 9 bit data polynomial is loaded into the first nine positions and shifted right, each shift multiplies the polynomial by x.

The parallel loading of successive data polynomials followed by a right shift forms the entire polynomial. Successive characters are added, modulo 2 (the binary case), which in logic is simply an exclusive or. Higher powers are created by early data.

For each bit that overflows from the x^8 position, a feedback substitution of congruent terms for x^9 can be made, see figure b. Overflow from x^8 may occur with every shift after the first. This enables the register to have only nine elements; the remainder of the division process is found in the register after all the characters have been loaded. It is transmitted or stored as the last character of the record. If feeding back output of the x^8 register position is puzzeling, remember that shifting is equivalent to multiplication by x; for a bit to enter the feedback loop, a shift is required, transforming x^8 into x^9.

Lowest order term of the polynomial is x^1, whereas when all data have been entered, the arrangement of figure a will have entry in x^0. This can be corrected in either of two ways:

(a) by simply shifting the data one more, with no input after the last entry is made, or

(b) by premultiplying all bits by x.

Premultiplication is done by entering all bits one position to the right.

One recovery of data from a transmission channel or from storage, an identical circuit is used on the receiver side, because division by G(x) is still central. If, after division, the remainder is not 0, an error has been detected.

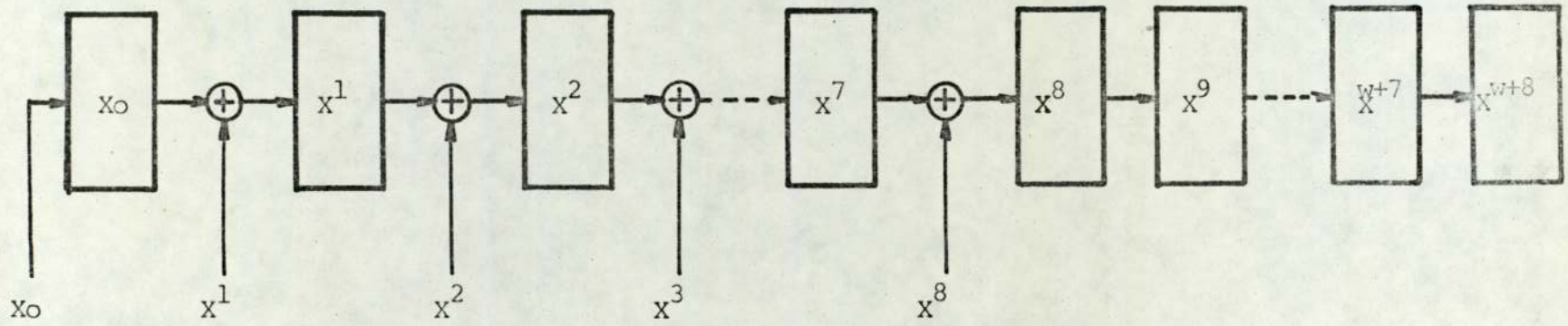

Xo
x1
x2
x3
x7
x8
x9
x w+7
x w+8
Xo
x1
x2
x3
x8

Fig. a

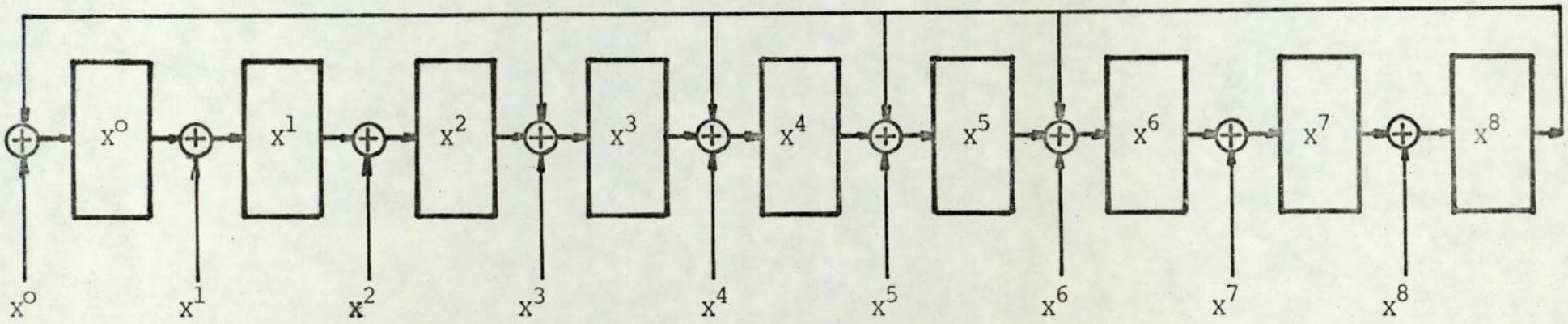

x o
x1
x2
x3
x4
x5
x6
x7
x8
x o
x1
x2
x3
x4
x5
x6
x7
x8

Fig. b

SYMBOLS	SIGNIFICANCE	(16) (E)	(8) (D)	(4) (C)	(2) (B)	(1) (A)	(F) (2)	(E) (1)	(D) (8)	(C) (4)	(B) (2)	(A) (1)	ASCII E+F	ASCII A+B+C+D
?	PEN UP	1	0	1	0	0	1	1	1	1	1	1	3=c	15=q
/	STOP	1	0	1	1	1	1	0	1	1	1	1	2=b	15=q
+	PLUS	0	1	0	1	1	1	0	1	0	1	1	2=b	11=k
-	MINUS	0	1	1	0	1	1	0	1	1	0	1	2=b	13=m
.	DECIM. POINT	0	1	1	1	0	1	0	1	1	1	0	2=b	14=m
∇	SPACE	1	1	1	0	0	1	0	0	0	0	0	2=b	o=0
FIG. SHIFT	FIG. SHIFT	1	1	0	1	1	1	0	0	1	0	0	2=b	4=d
LINE FEED	NEW LINE	1	1	1	1	0	0	0	1	0	1	0	σ=σ	10=j
*	PEN SELECTION	0	0	0	1	1	1	0	1	0	1	0	2=b	10=j
(	HERALD	1	0	0	0	1	1	0	1	0	0	0	2=b	8=h
)	ANTIHERALD	1	0	0	1	0	1	0	1	0	0	1	2=b	g=i

TABLE 1

Bus Signals

The binary signals on the bus have two states, low and high, which correspond respectively to nominal voltage levels of 0 and +2.7 volts. Any level between ground and .4 volt is interpreted as low any level more positive than 2.2 volts is interpreted as high. The level listed for a signal in the following table is the voltage level on the line when the signal represents a 1 or produces the indicated function. A low signal is indicated in the prints by a bar over its name.

Signal	Direction	Level	
DS0 to DS5	To device	Low	Device Selection. The processor places the device code (bits 10–15 of the instruction word) on these lines during the execution of an in-out instruction. The lines select one of 62 devices (codes 01–76) that may be connected to the bus. Only the selected device responds to control signals generated during the instruction.
DATA0 to DATA15	Bidirectional	Low	Data. All data and addresses are transferred between the processor and the devices attached to the bus via these sixteen lines. For programmed output the processor places the AC specified by the instruction on the data lines and then generates DATOA, DATOB or DATOC to load the data from the lines into the corresponding buffer in the device selected by DS0–5, or generates MSKO to set up the Interrupt Disable flags in all of the devices according to the mask on the data lines. For data channel output the processor places the memory buffer on the data lines and generates DCHO to load the contents of the lines into the data buffer in the device that is being serviced. For programmed input the processor generates DATIA, DATIB or DATIC to place information from the corresponding buffer in the device selected by DS0–5 on the data lines, or generates INTA to place the code of the nearest device that is requesting an interrupt on lines 10–15. The processor then loads the data from the lines into the AC selected by the instruction. To get an address for data channel access the processor generates DCHA to place a memory address from the nearest device that is requesting access on lines 1–15 and then loads the address into the memory address register. For data channel input the processor generates DCHI to place the data buffer of the device being serviced on the data lines and then loads the contents of the lines into the memory buffer.
DATOA	To device	High	Data Out A. Generated by the processor after AC has been placed on the data lines in a DOA to load the data into the A buffer in the device selected by DS0–5.
DATIA	To device	High	Data In A. Generated by the processor during a DIA to place the A buffer in the device selected by DS0–5 on the data lines.
DATOB	To device	High	Data Out B. Equivalent to DATOA but loads the B buffer.
DATIB	To device	High	Data In B. Equivalent to DATIA but places the B buffer on the data lines.
DATOC	To device	High	Data Out C. Equivalent to DATOA but loads the C buffer.
DATIC	To device	High	Data In C. Equivalent to DATIA but places the C buffer on the data lines.
STRT	To device	High	Start. Generated by the processor in any nonskip IO instruction with an S control function (bits 8–9 = 01) to clear Done, set Busy, and clear the INT REQ flipflop in the device selected by DS0–5.
CLR	To device	High	Clear. Generated by the processor in any nonskip IO instruction with a C control function (bits 8–9 = 10) to clear Busy, Done and the INT REQ flipflop in the device selected by DS0–5.
IOPLS	To device	High	IO Pulse. Generated by the processor in any nonskip IO instruction with a P control function (bits 8–9 = 11) to perform some special function in the device selected by DS0–5 (this signal is for custom applications).
SELB	To processor	Low	Selected Busy. Generated by the device selected by DS0–5 if its Busy flag is set.
SELD	To processor	Low	Selected Done. Generated by the device selected by DS0–5 if its Done flag is set.

Table 2

RQENB	To device	Low	Request Enable. Generated at the beginning of every memory cycle to allow all devices on the bus to request program interrupts or data channel access. In any device RQENB sets the INT REQ flipflop if Done is set and Interrupt Disable is clear. Otherwise it clears INT REQ. In any device connected to the data channel RQENB sets the DCH REQ flipflop if the DCH SYNC flipflop is set. Otherwise it clears DCH REQ.
INTR	To processor	Low	Interrupt Request. Generated by any device when its INT REQ flipflop is set. This informs the processor that the device is waiting for an interrupt to start.
INTP	To device	Low	Interrupt Priority. Generated by the processor for transmission serially to the devices on the bus. If the INT REQ flipflop in a device is clear when the device receives INTP, the signal is transmitted to the next device.
INTA	To device	High	Interrupt Acknowledge. Generated by the processor during the INTA instruction. If a device receives INTA while it is also receiving INTP and its INT REQ flipflop is set, it places its device code on data lines 10–15.
MSKO	To device	Low	Mask Out. Generated by the processor during the MSKO instruction after AC has been placed on the data lines to set up the Interrupt Disable flags in all devices according to the mask on the lines.
DCHR	To processor	Low	Data Channel Request. Generated by any device when its DCH REQ flipflop is set. This informs the processor that the device is waiting for data channel access.
DCHP	To device	Low	Data Channel Priority. Generated by the processor and transmitted serially to the devices on the bus. If the DCH REQ flipflop in a device is clear when the device receives DCHP, the signal is transmitted to the next device.
DCHA	To device	Low	Data Channel Acknowledge. Generated by the processor at the beginning of a data channel cycle. If a device receives DCHA while it is also receiving DCHP and its DCH REQ flipflop is set, it places the memory address to be used for data channel access on data lines 1–15 and sets its DCH SEL flipflop.
DCHM0 DCHM1	To processor	Low	Data Channel Mode. Generated by a device when its DCH SEL flipflop is set to inform the processor of the type of data channel cycle desired as follows: DCHM0 DCHM1 0 (H) 0 (H) Data out 0 (H) 1 (L) Increment memory 1 (L) 0 (H) Data in 1 (L) 1 (L) Add to memory In addition to performing the necessary functions internally, the processor generates DCHI and/or DCHO for the required in-out transfers.
DCHI	To device	High	Data Channel In. Generated by the processor for data channel input (DCHM0 = 1) to place the data register of the device selected by DCHA on the data lines.
DCHO	To device	High	Data Channel Out. Generated by the processor for data channel output (DCHM0–1 ≠ 10) after the word from memory or the arithmetic result has been placed on the data lines to load the contents of the lines into the data register of the device selected by DCHA.
OVFLO	To device	High	Overflow. Generated by the processor during a data channel cycle that increments memory or adds to memory (DCHM1 = 1) when the result exceeds $2^{16} - 1$.
IORST	To device	High	IO Reset. Generated by the processor in the IORST instruction or when the console reset switch is pressed to clear the control flipflops in all interfaces connected to the bus. This signal is also generated during power turnon.

IN-OUT BUS SIGNAL CONNECTIONS

H High
L Low

B Bidirectional
D From processor to device

P From device to processor

Signal	Level	Direction	Panel Pin	External Bus Pin		Signal	Level	Direction	Panel Pin	External Bus Pin	
CLR†	H	D	A50	K	2	DCHM0	L	P	B17	d	27
DATA0	L	B	B62	w	3	DCHM1	L	P	B21	e	28
DATA1	L	B	B65	z	4	DCHO†	H	D	B33	h	29
DATA2	L	B	B82	AD	5	DCHP IN	L	D*	A94	b	30
DATA3	L	B	B73	AB	6	DCHP OUT	L	D*	A93		
DATA4	L	B	B61	v	7	DCHR	L	P	B35	j	31
DATA5	L	B	B57	r	8	DS0	L	D	A72	X	32
DATA6	L	B	B95	AE	9	DS1	L	D	A68	V	33
DATA7	L	B	B55	n	10	DS2	L	D	A66	U	34
DATA8	L	B	B60	u	11	DS3	L	D	A46	H	35
DATA9	L	B	B63	x	12	DS4	L	D	A62	S	36
DATA10	L	B	B75	AC	13	DS5	L	D	A64	T	37
DATA11	L	B	B58	s	14	INTA†	H	D	A40	D	38
DATA12	L	B	B59	t	15	INTP IN	L	D*	A96	c	39
DATA13	L	B	B64	y	16	INTP OUT	L	D*	A95		
DATA14	L	B	B56	p	17	INTR	L	P	B29	f	40
DATA15	L	B	B66	AA	18	IOPLS†	H	D	A74	Y	41
DATIA†	H	D	A44	F	19	IORST†	H	D	A70	W	42
DATIB†	H	D	A42	E	20	MSKO	L	D	A38	C	43
DATIC†	H	D	A54	M	21	OVFLO†	H	D	B39	l	44
DATOA†	H	D	A58	P	22	RQENB†	L	D	B41	m	45
DATOB†	H	D	A56	N	23	SELB	L	P	A82	a	46
DATOC†	H	D	A48	J	24	SELD	L	P	A80	Z	47
DCHA†	L	D	A60	R	25	STRT†	H	D	A52	L	48
DCHI†	H	D	B37	k	26	Power on	+5	D		B	49

TABLE 3

Ascii code - output from computer

TAPE CODE 87654.321	CHARACTER
	RUN OUT
	0 ZERO
	1
	2
	3
	4
	5
	6
	7
	8
	9
	+
	-
	SPACE
	/
	CAR RET
	NEW LINE
	ERASE
	TAB
	DEC PT
	END/TAPE
	£
	%
	(
	)
	*
	?
	:
	&
	=
	'
	,
	"

Elliott 803 code- needed for Electroplotter

TAPE CODE 54.321	CHARACTER
	RUN OUT
	0 ZERO F.S.
	1 F.S.
	2 F.S.
	3 F.S.
	4 F.S.
	5 F.S.
	6 F.S.
	7 F.S.
	8 F.S.
	9 F.S.
	+ F.S.
	- F.S.
	SPACE
	/ F.S.
	CAR RET
	LN FEED
	L SHIFT
	DEC P.F.S.
	£ F.S.
	% F.S.
	(F.S.
	) F.S.
	* F.S.
	? F.S.
	: F.S.
	& F.S.
	= F.S.
	' F.S.
	, F.S.
	@ F.S.
	FIG SHIFT.

Table 4

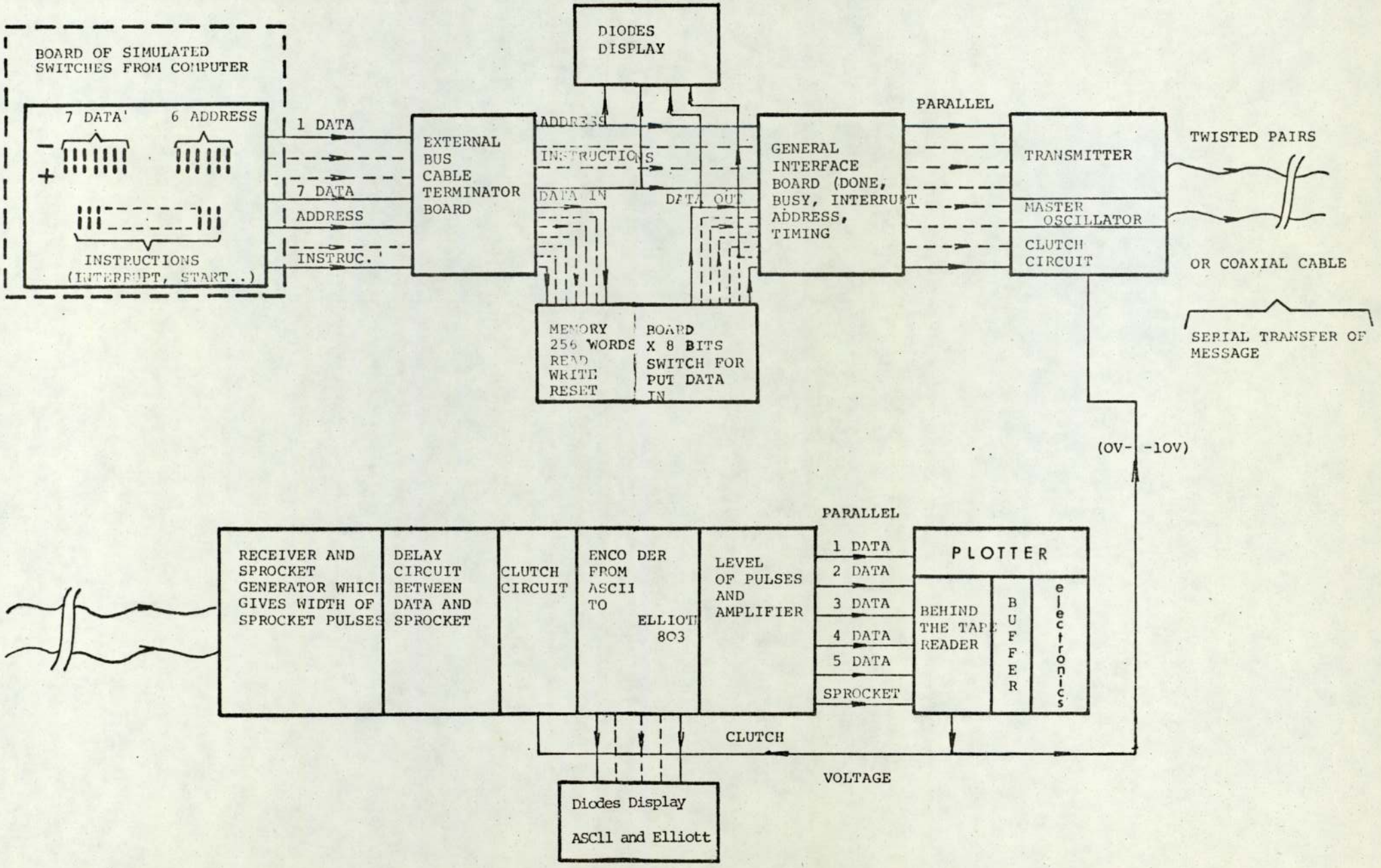


Fig 1. The plotting system

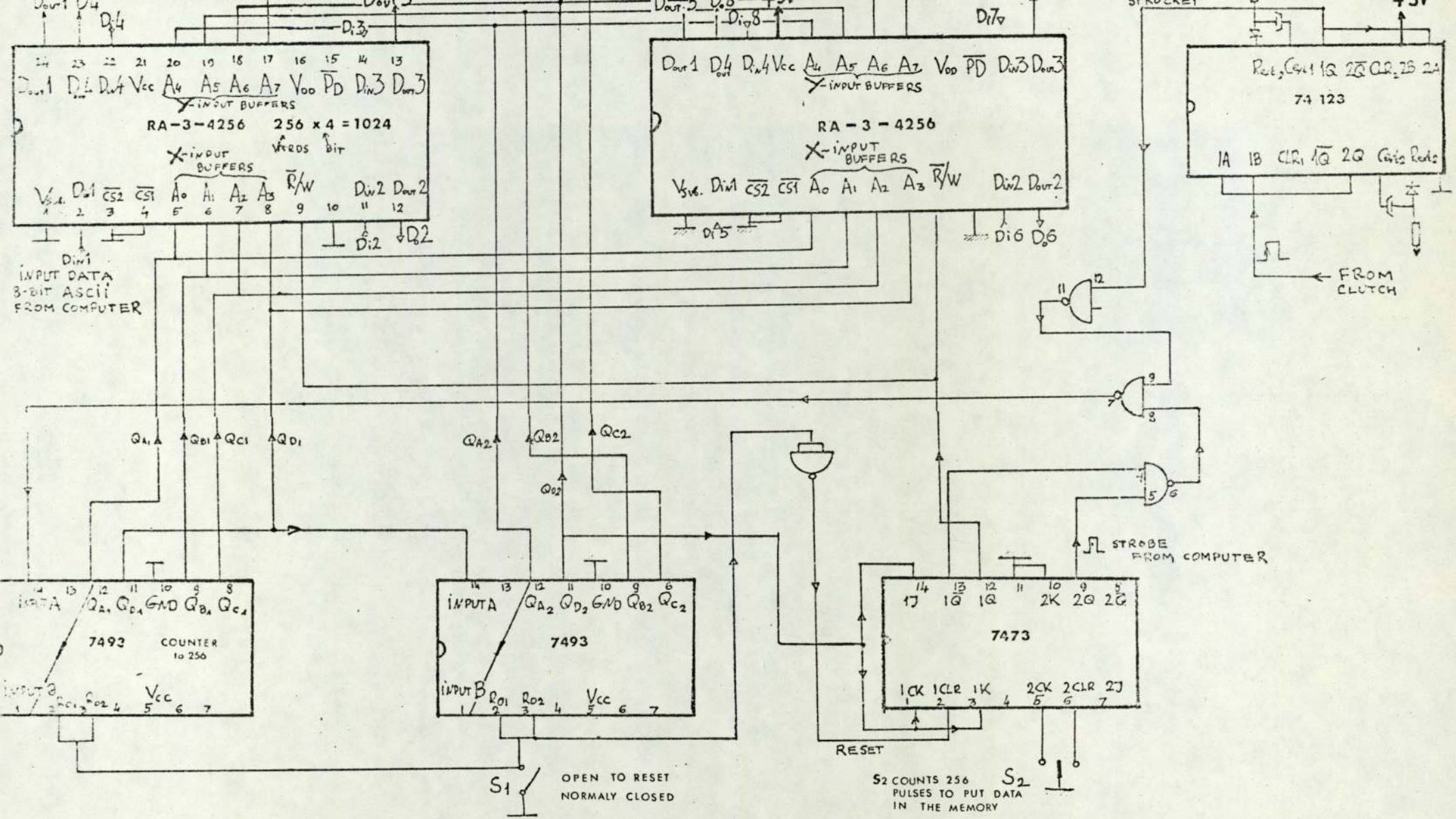


FIG. 2. Memory Board

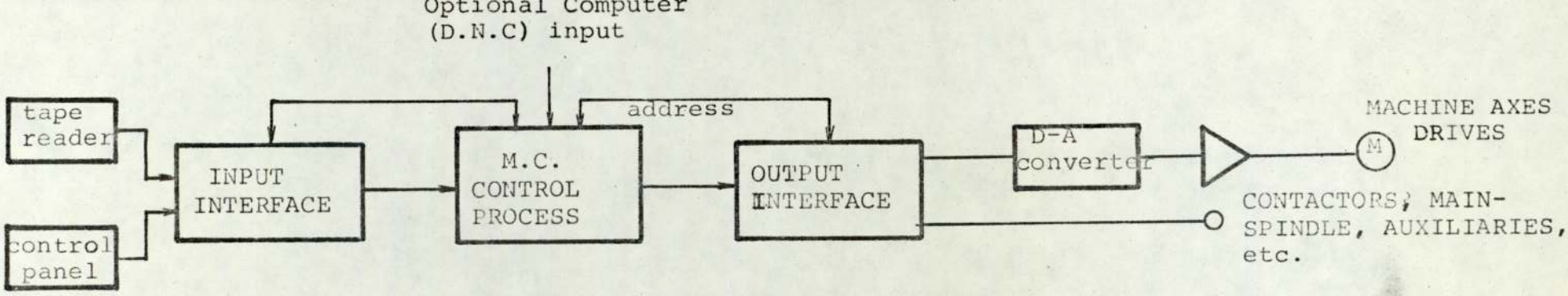


Fig. 3
Simplified block diagram of a mini-computer C.N.C. system

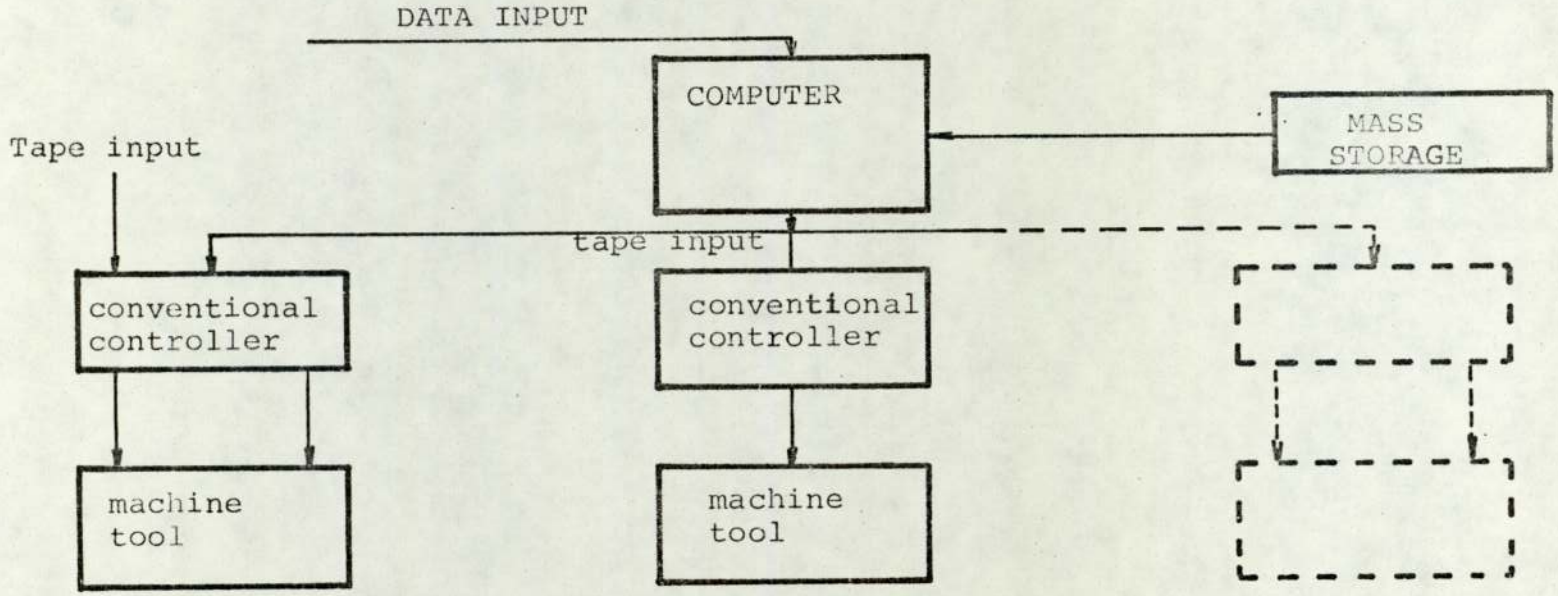


Fig. 4
Direct numerical control (BTR) system

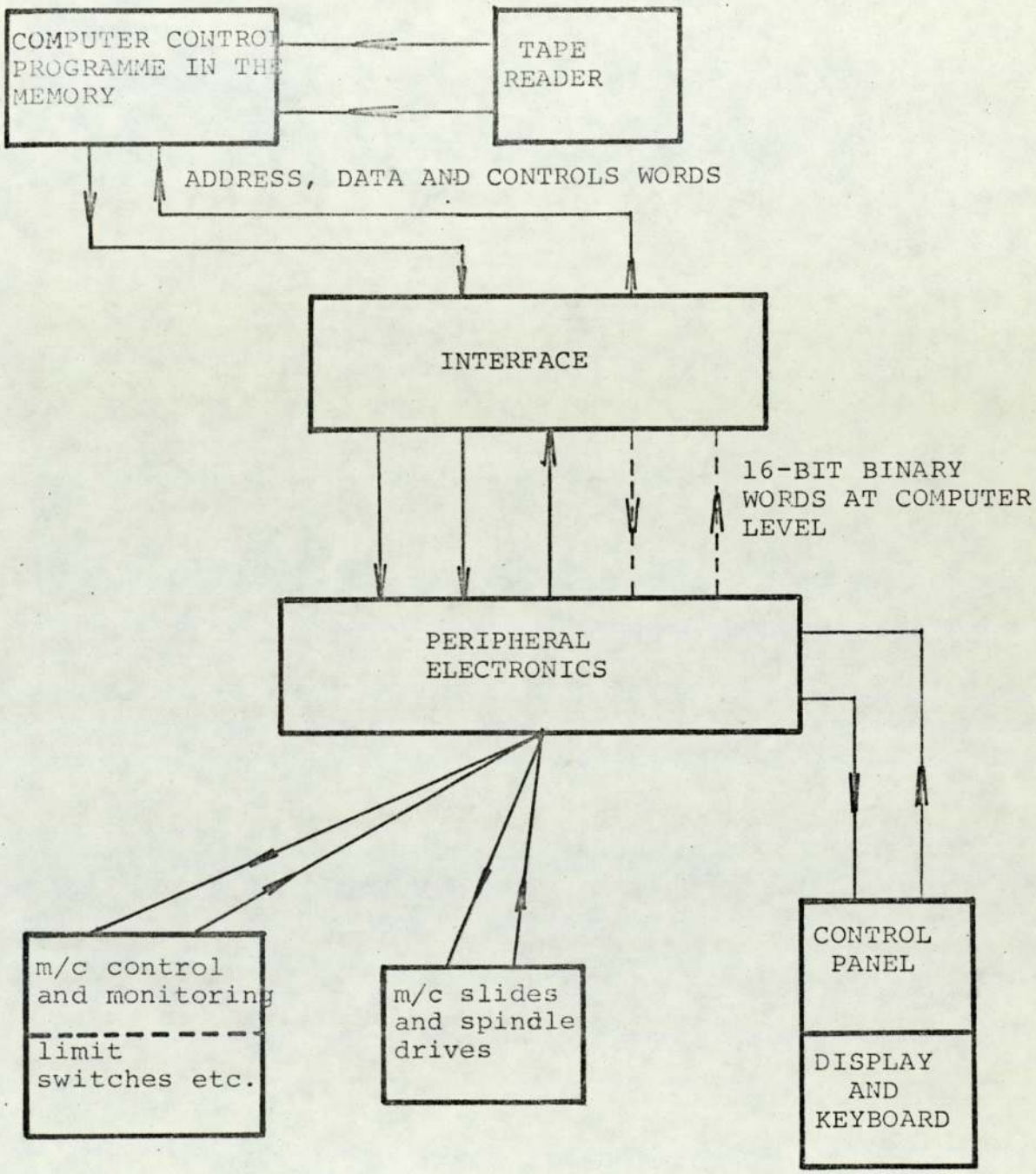


Fig. 5. The flow diagram of information between components.

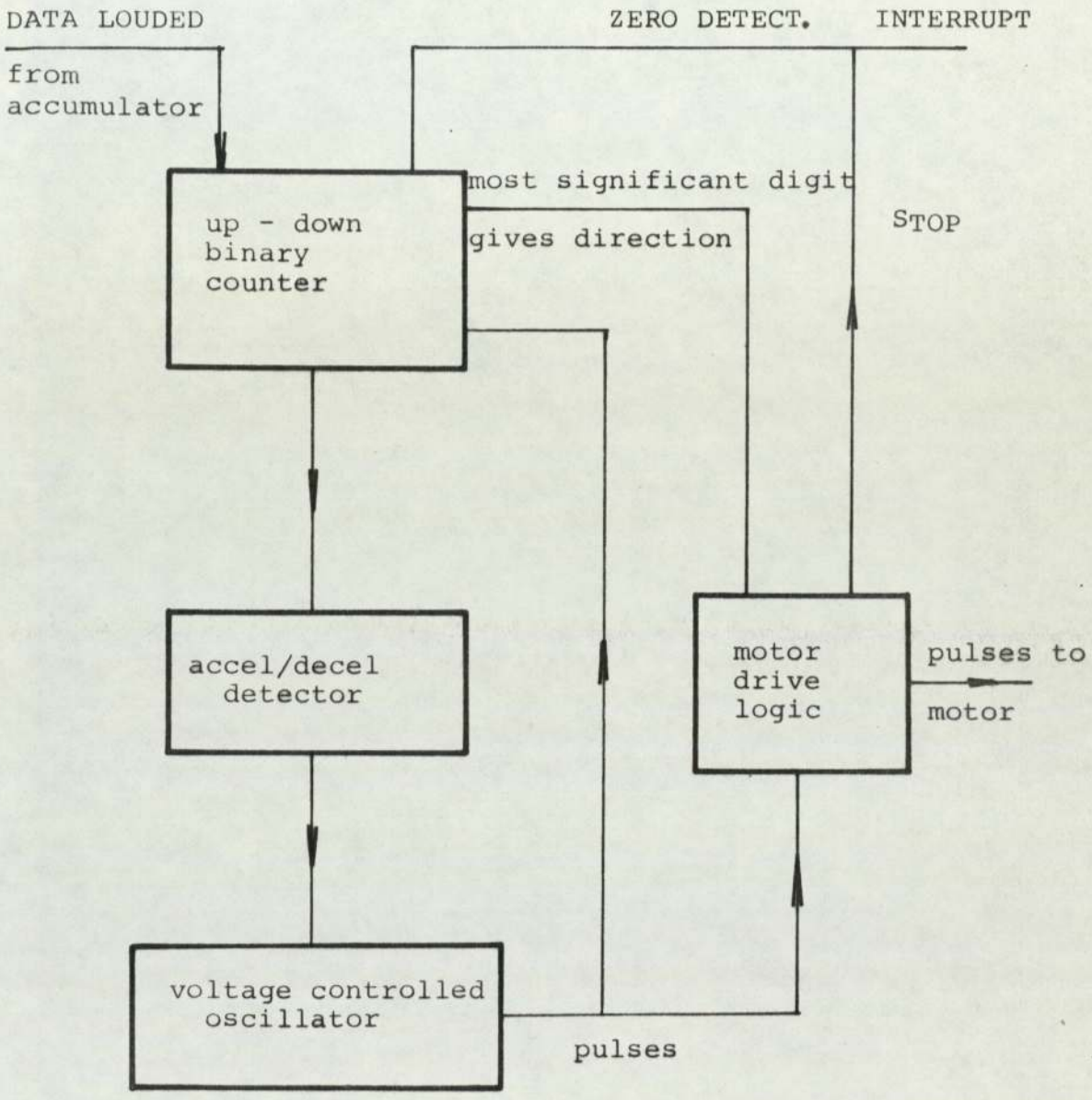


Fig. 6. The continious path control.

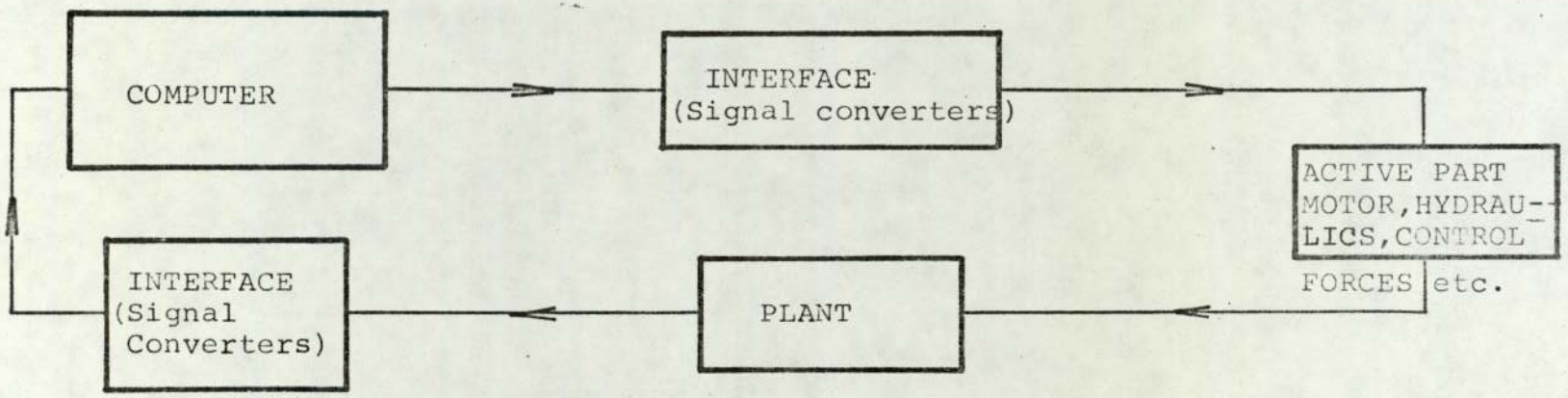


Fig. 8 On-line computer control

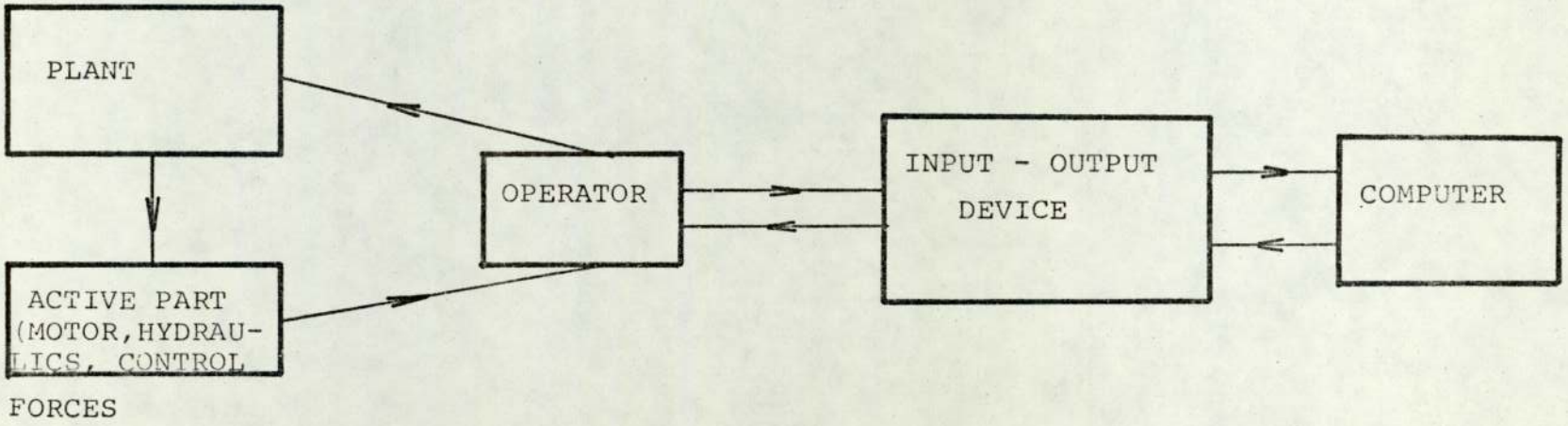


Fig. 7. Off-line computer control.

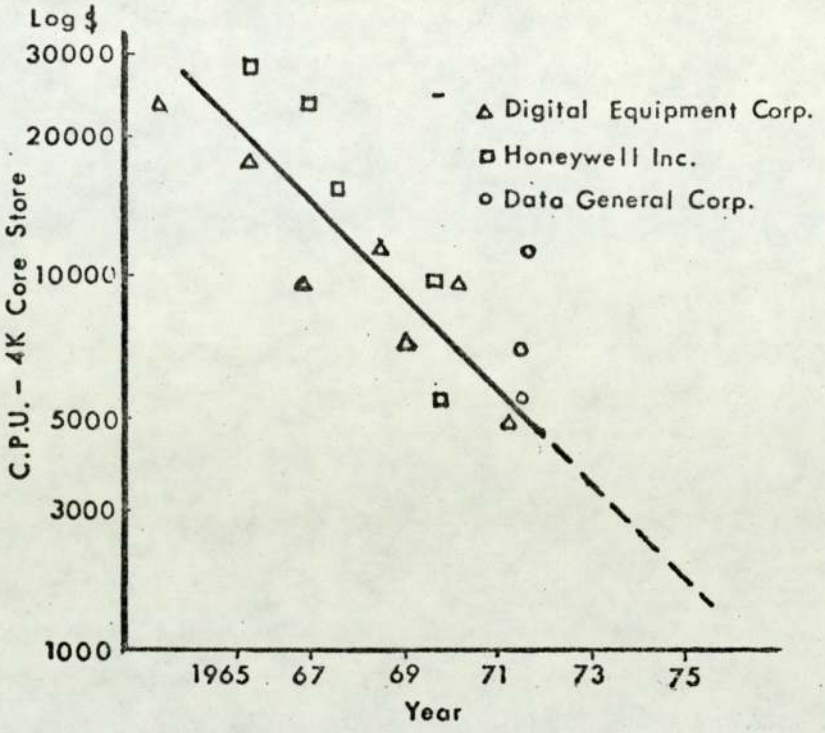


Fig. 9. Computer price data.

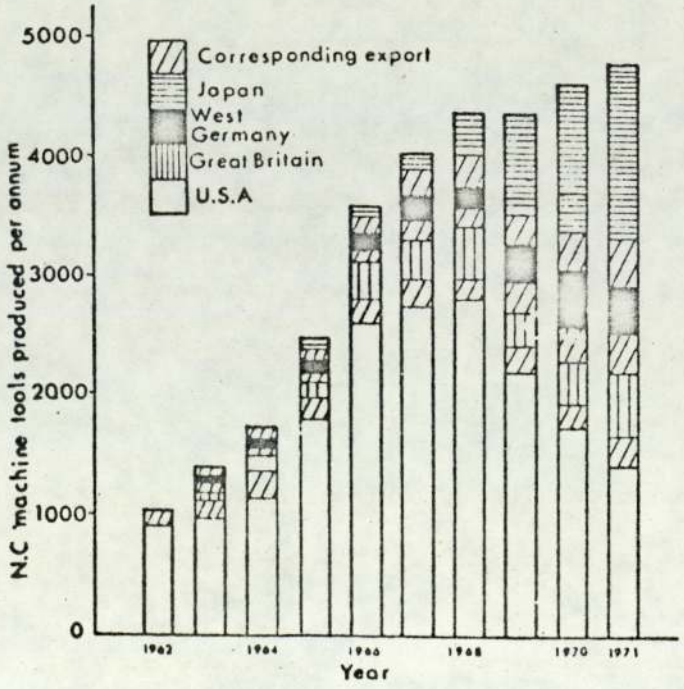


Diagram 10. Comparison of NC machine tool production.

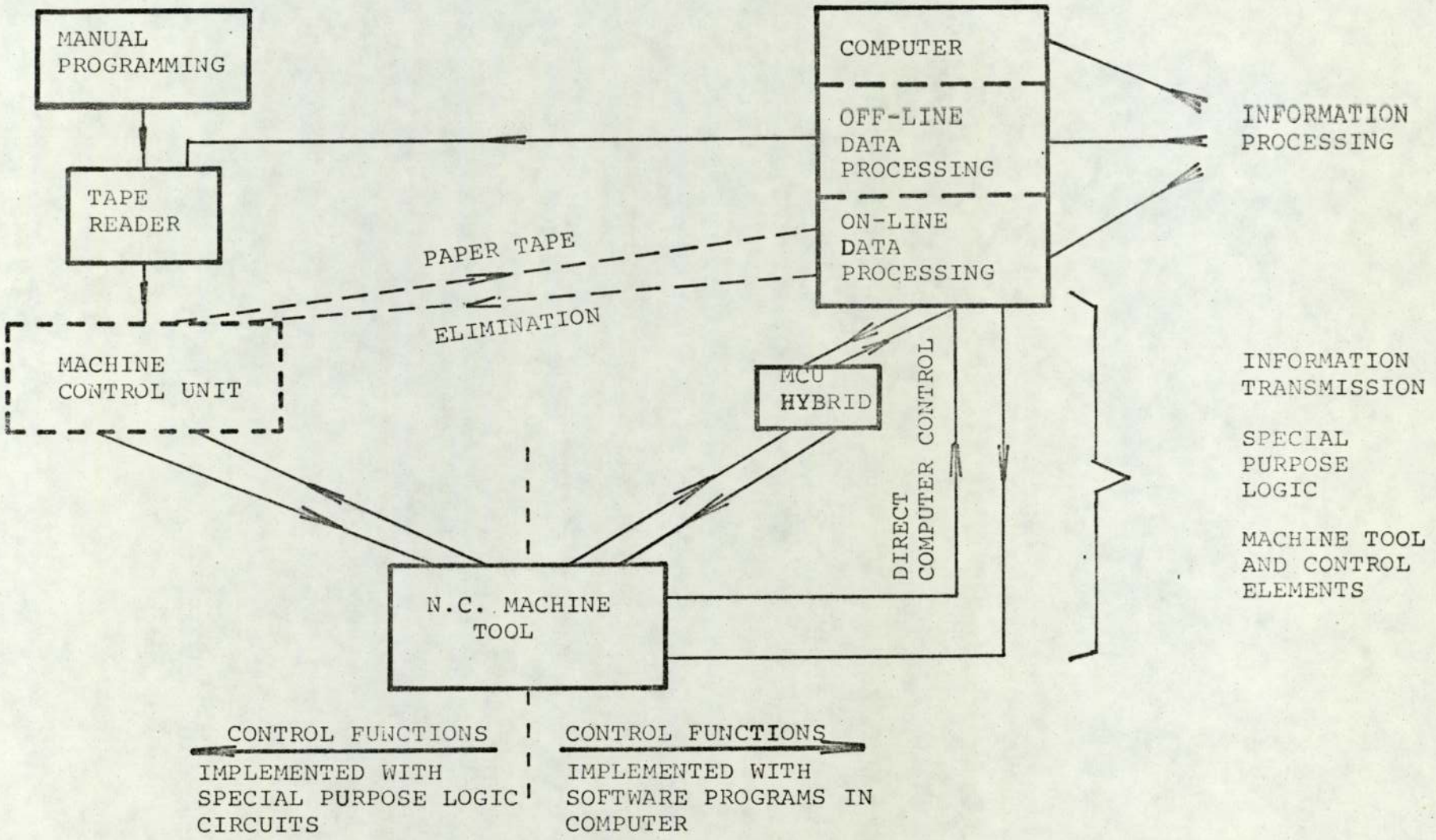


Fig. 11 The execution of control functions

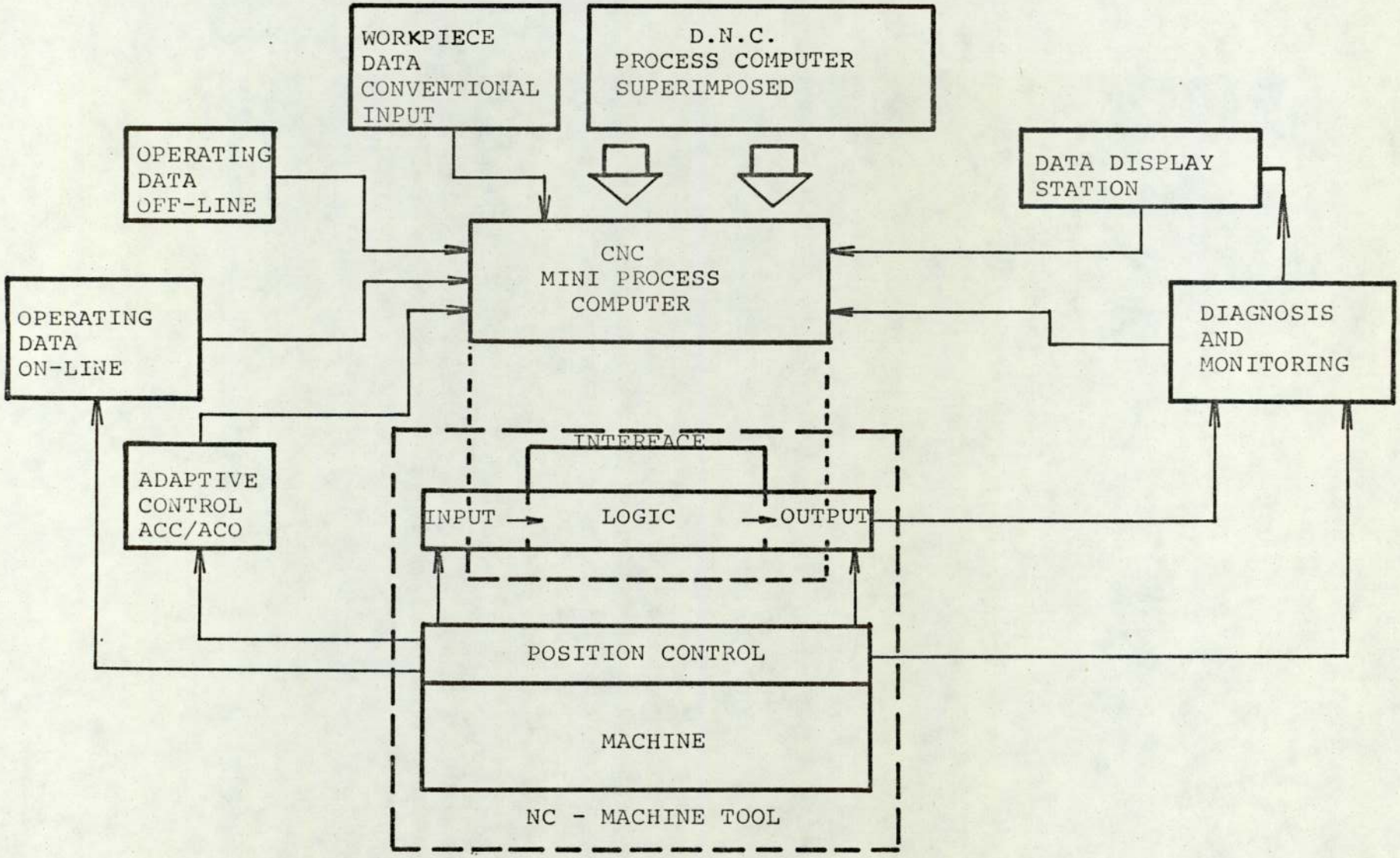


Fig.12. Extended C.N.C. concept.

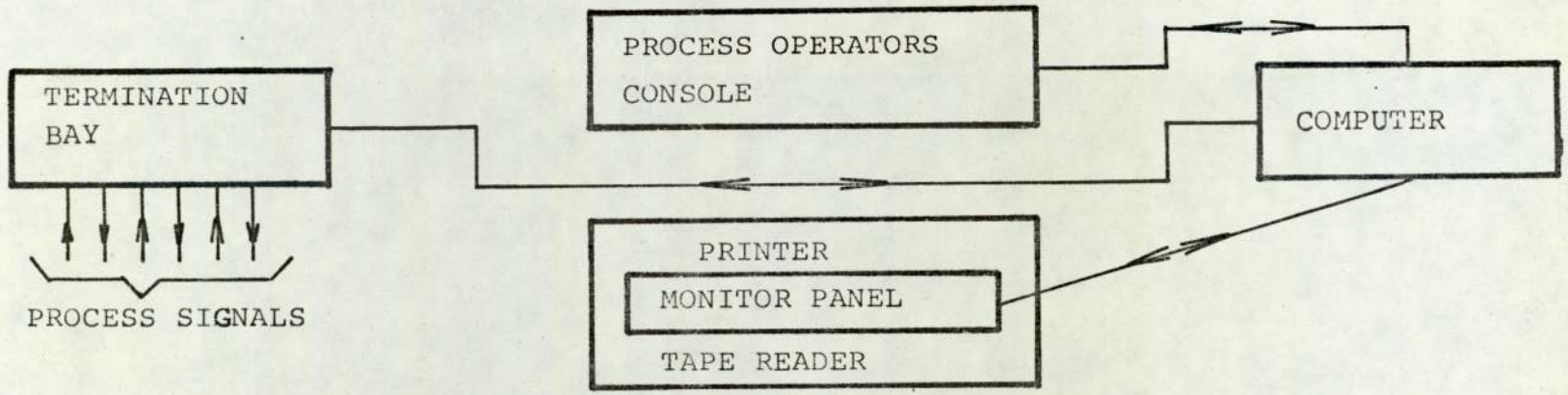


Fig.13. Typical control system.

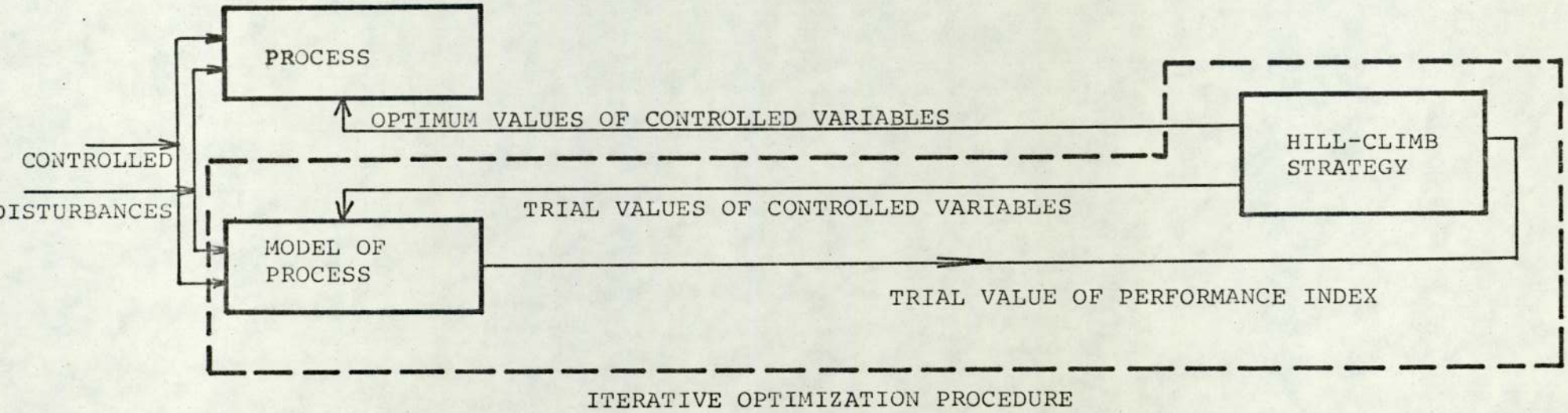


Fig. 14. Hill-Climbing system

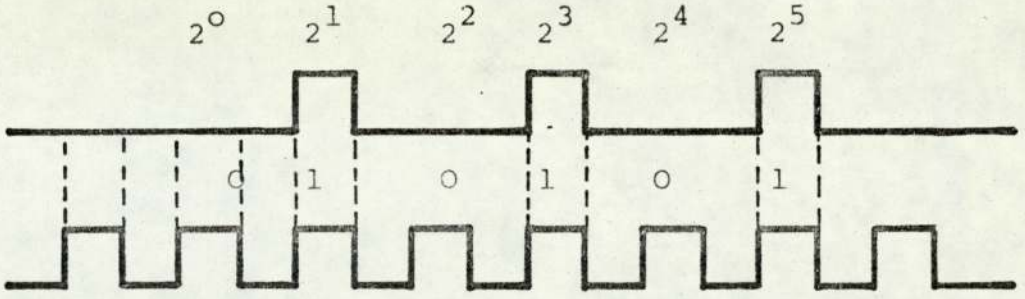


Fig. 15. Serial presentation of 101010.

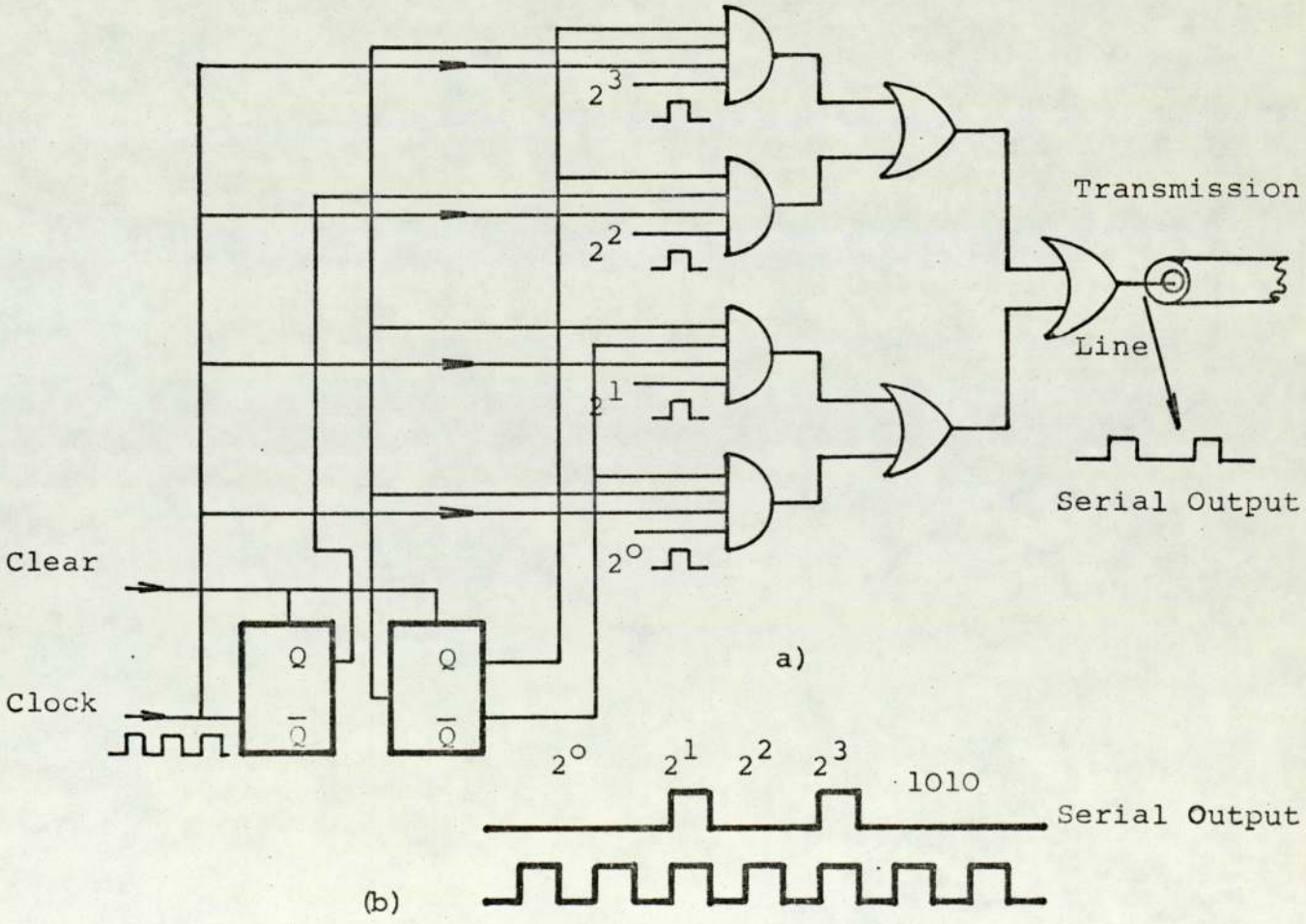


Fig. 16. Serial to parallel transmission

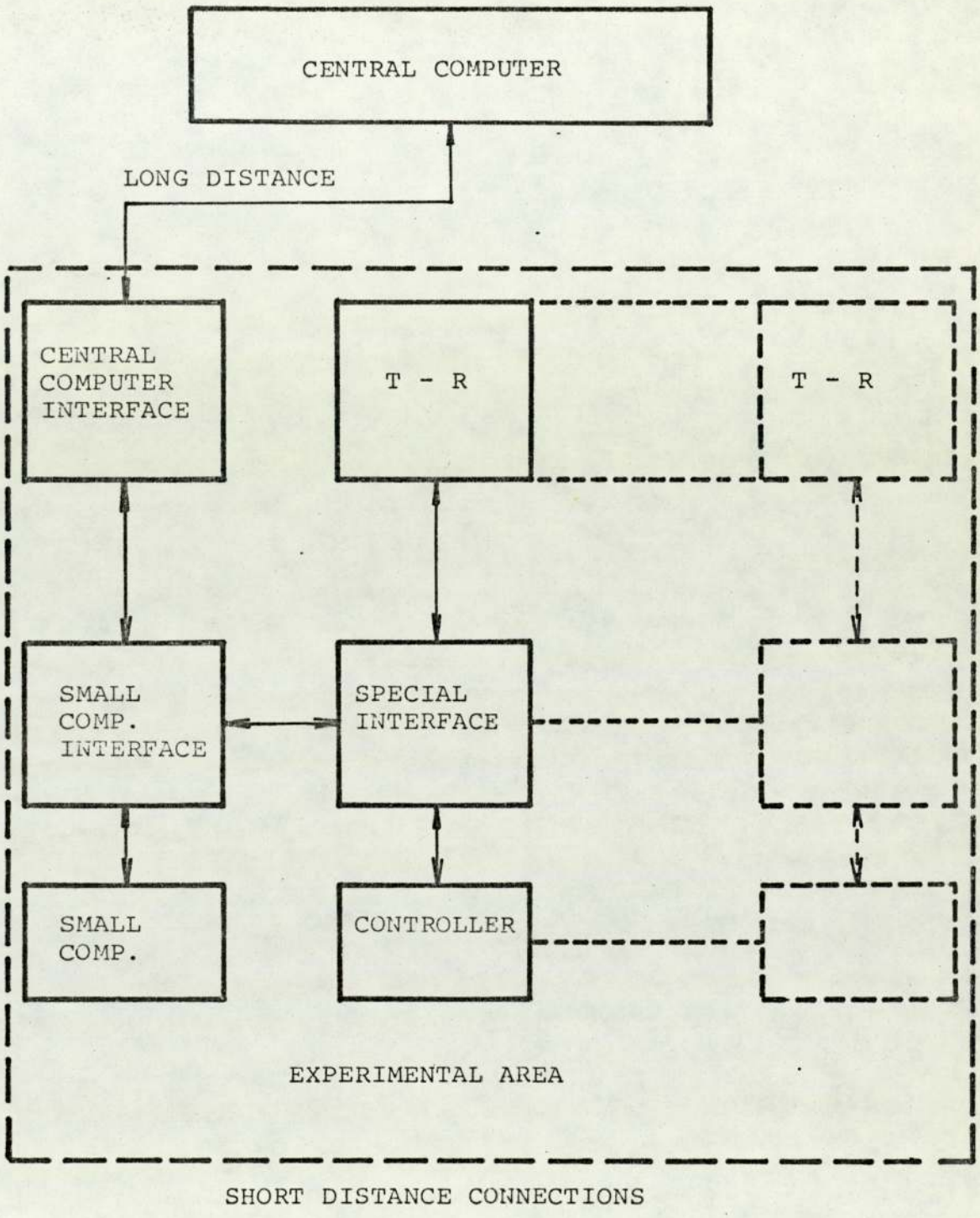
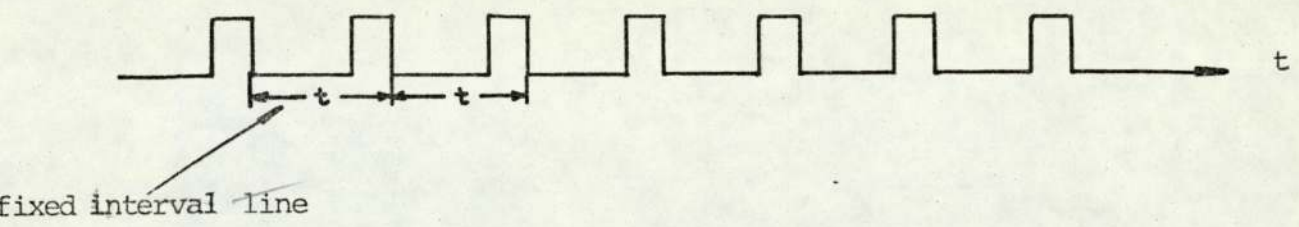
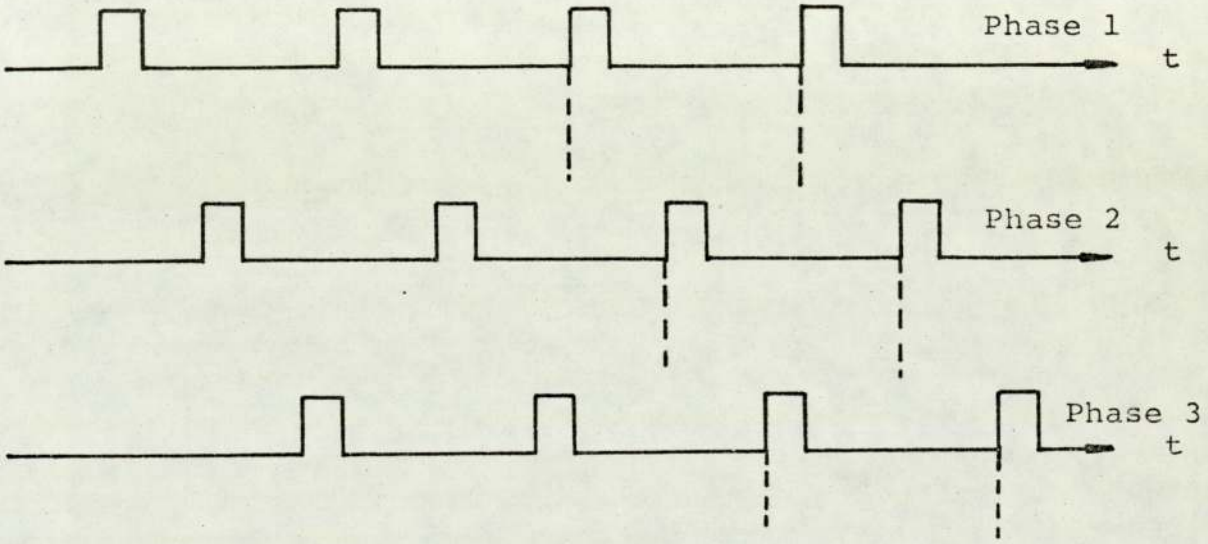


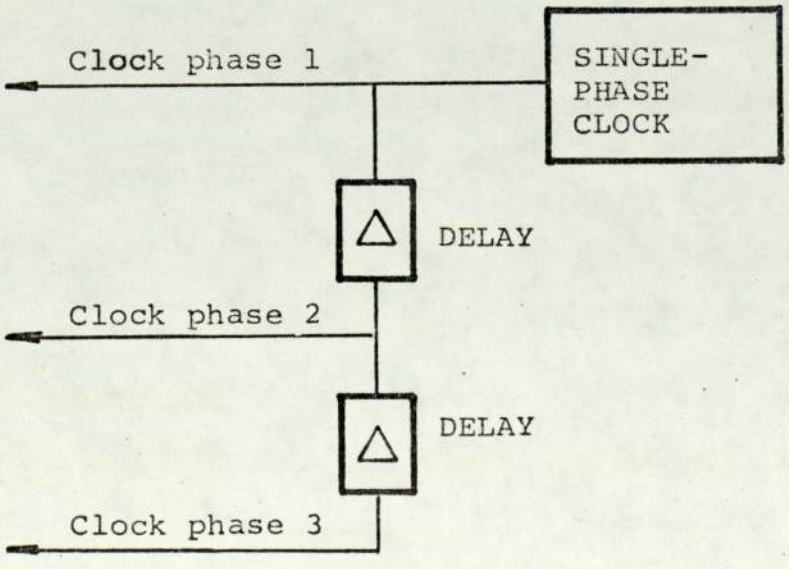
Fig. 17. System data links.



a)



Pulses from a three-phase clock



b) clock pulses

Fig. 18. Clock pulses

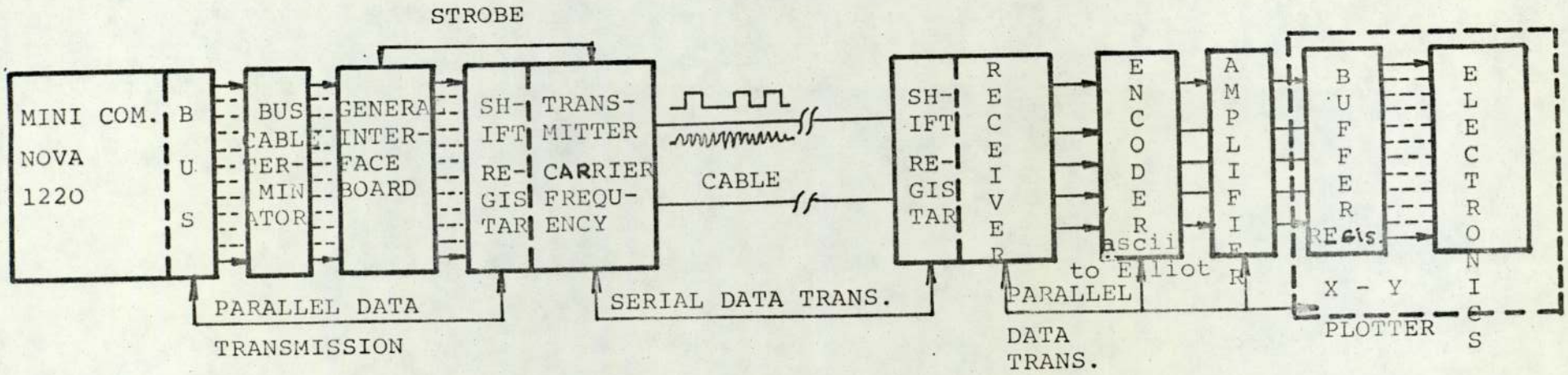


Fig.19. Long distance data transmission

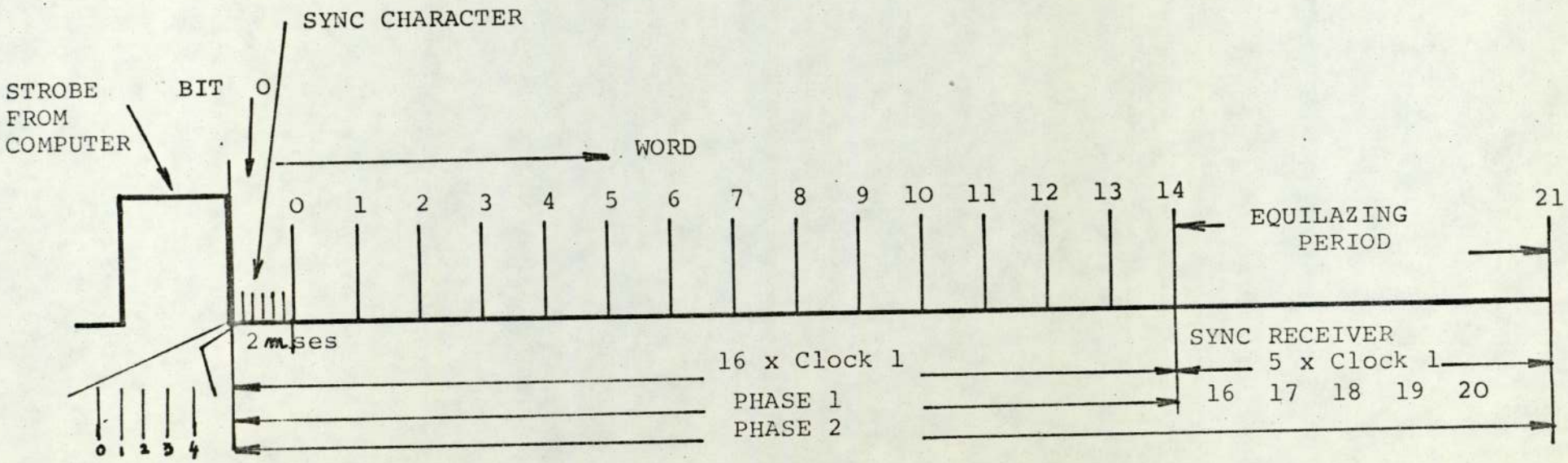


Fig.20. Timing diagram of synchronisation.

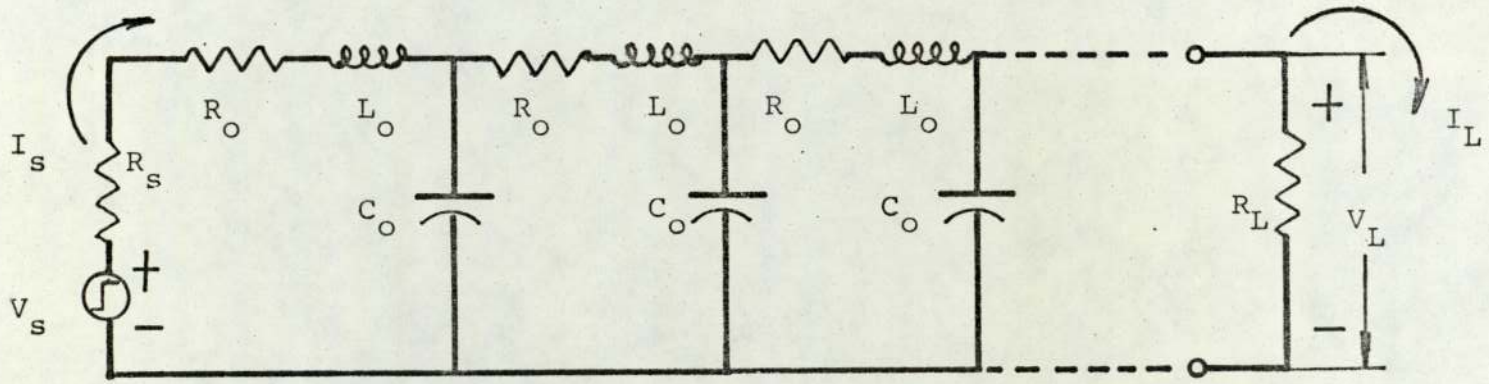


Fig. 21. Transmission line by the lumped representation.

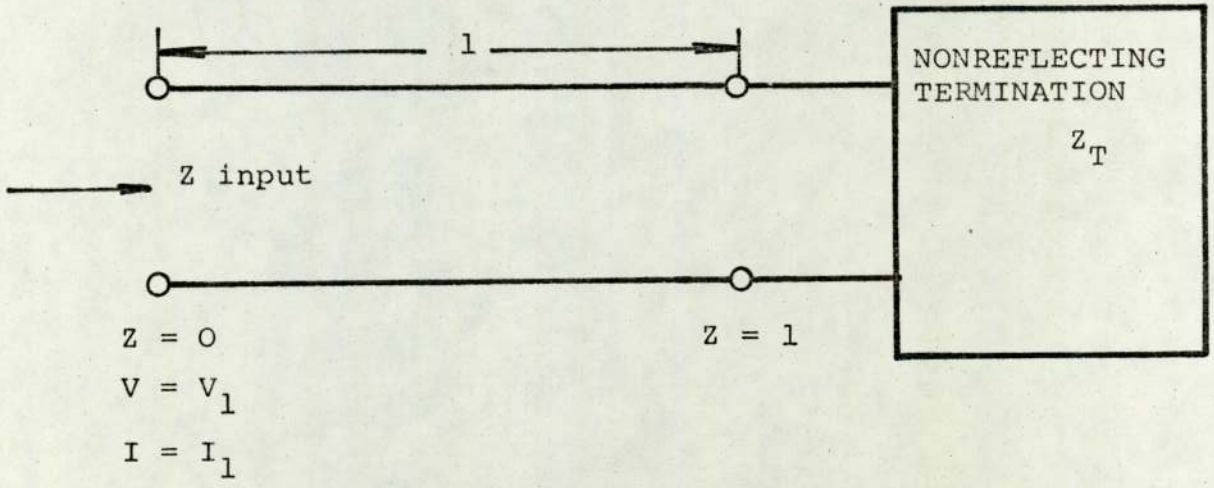


Fig.22. Transmission line circuit.

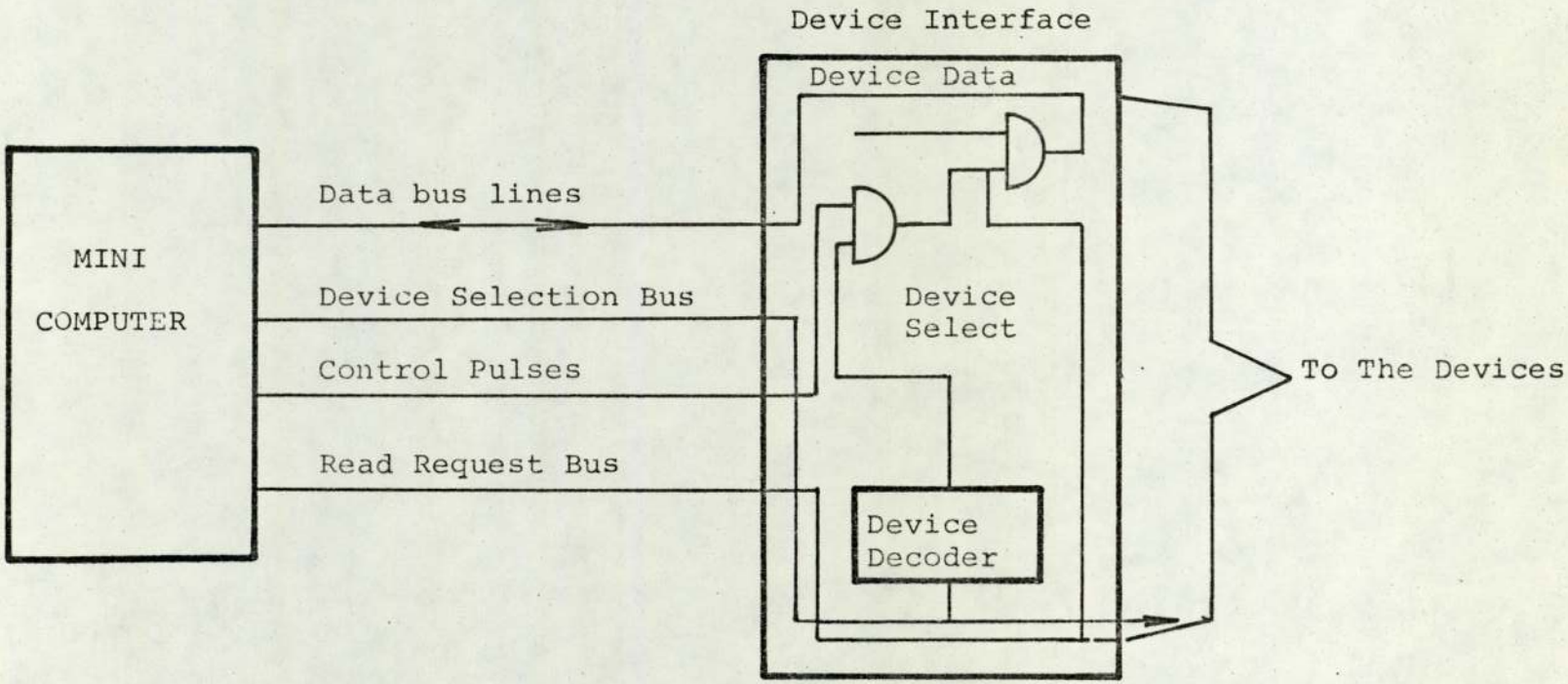


Fig. 23. General pattern of interface.

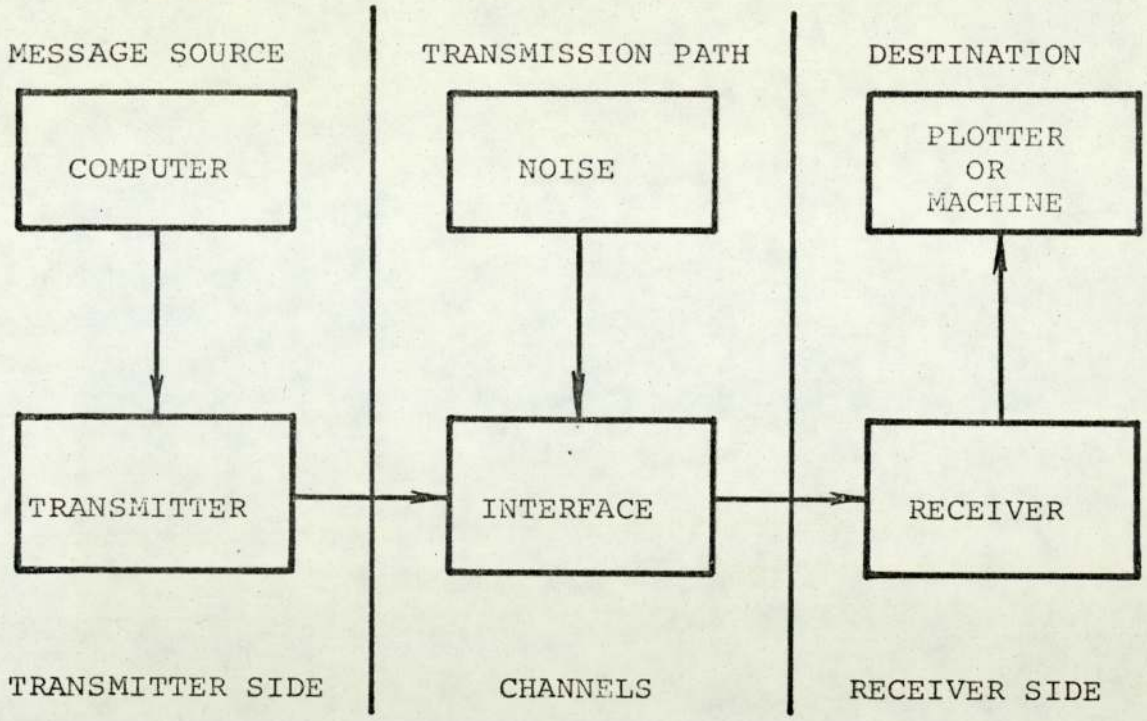


Fig. 24. Communication system.

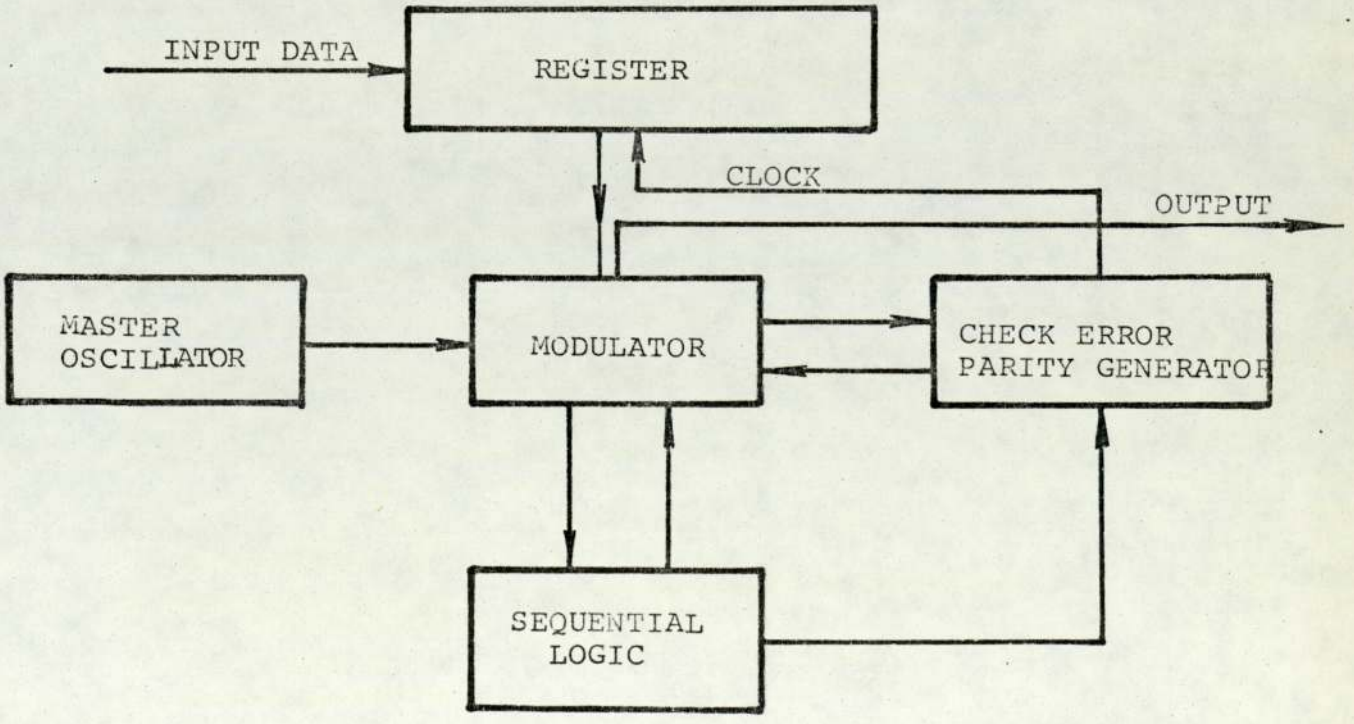


Fig. 25. The block diagram of the transmitter.

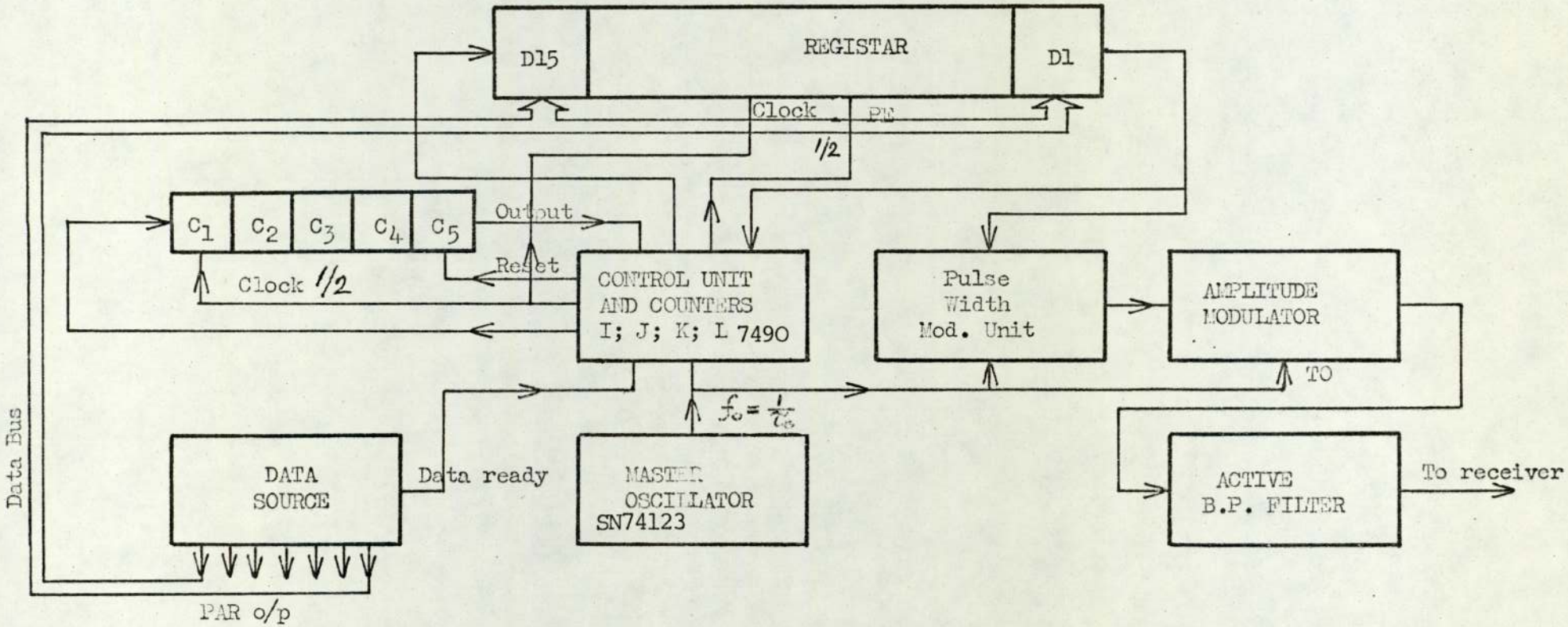


Fig. 26. Transmitter.

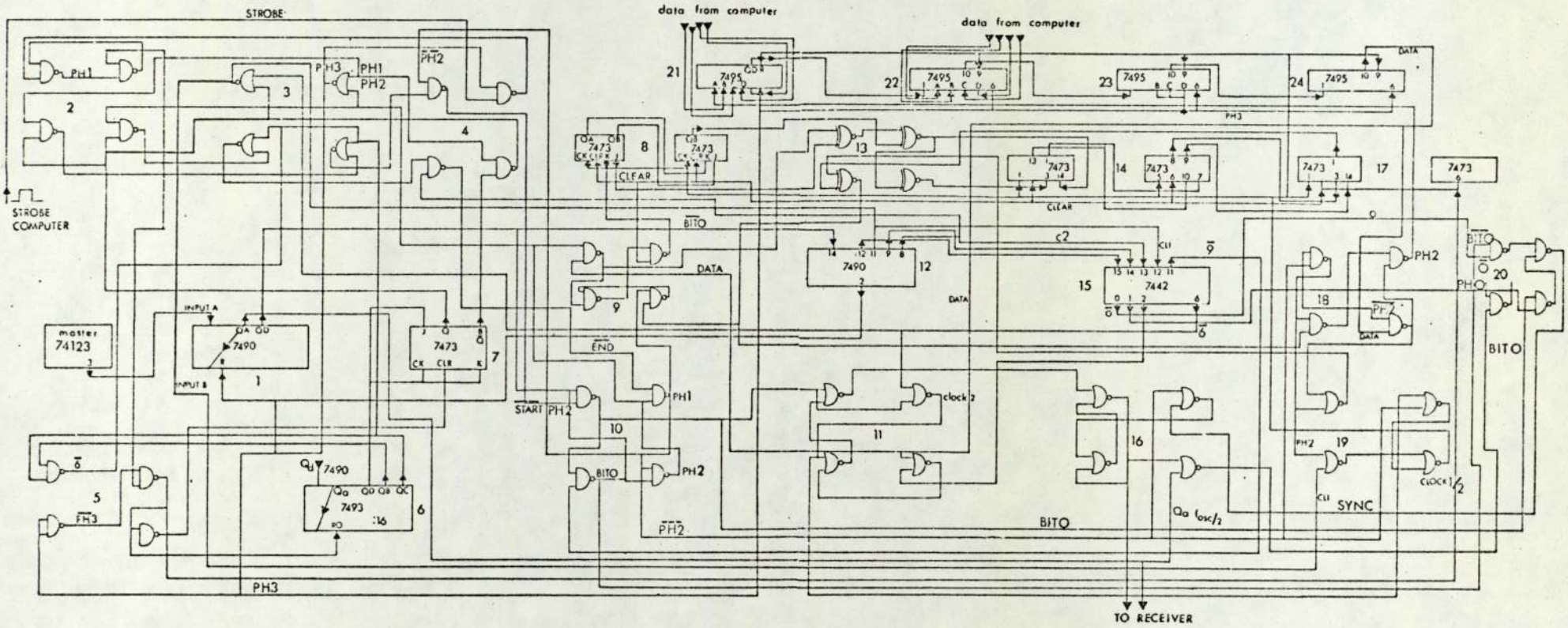


Fig. 27. Electrical diagram of the Transmitter.

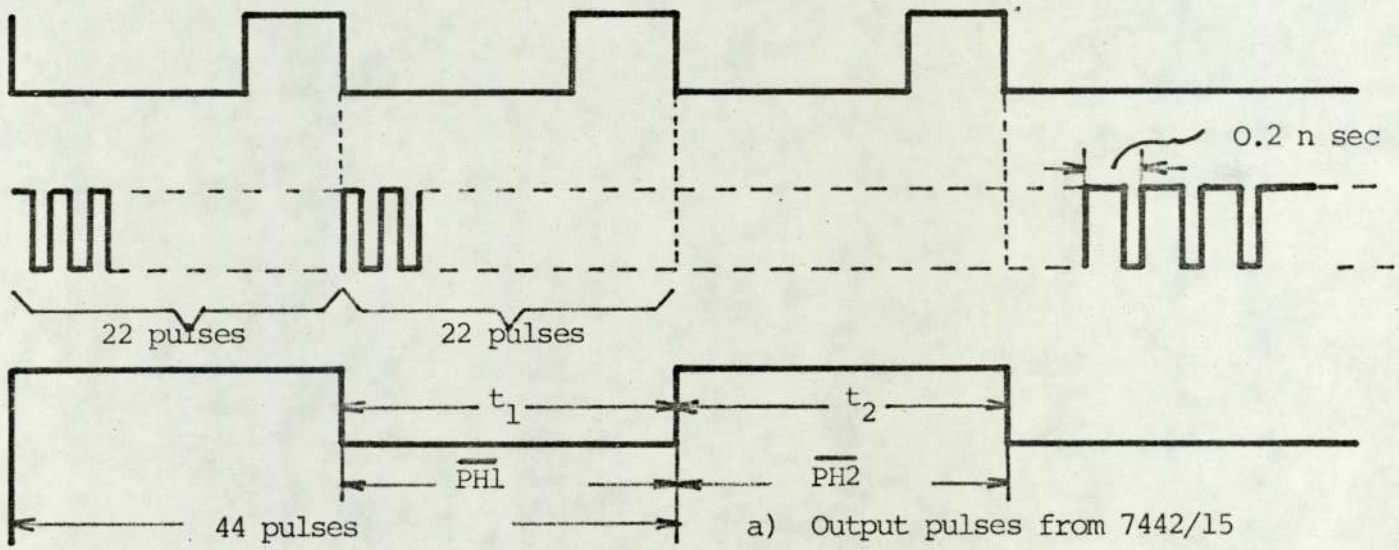
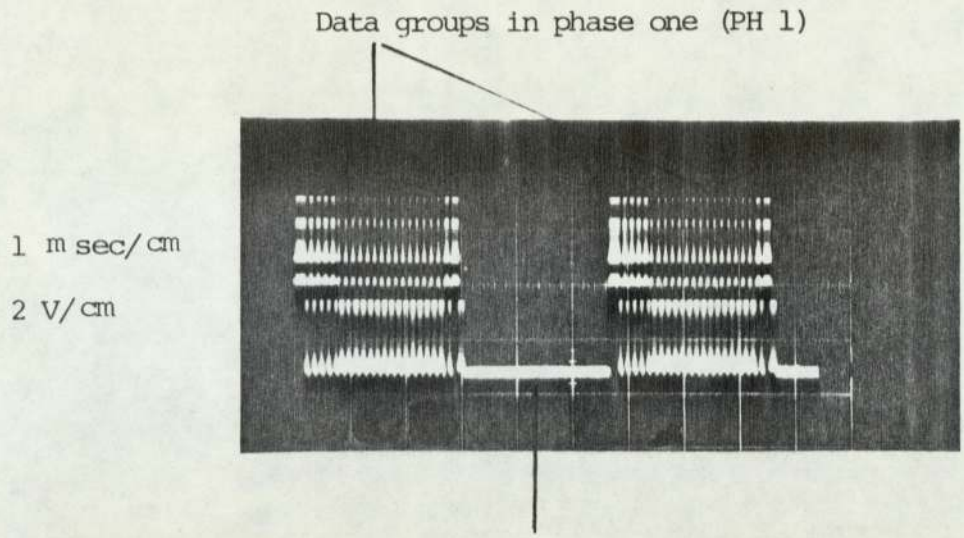
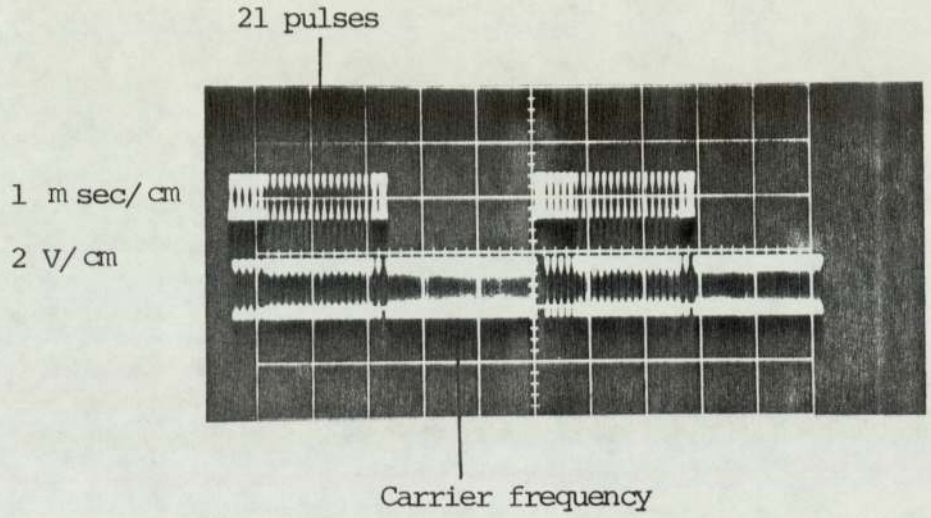


Fig. 28. Timing diagram of PHi, PH2, and PH3





Number 6 in Ascii code

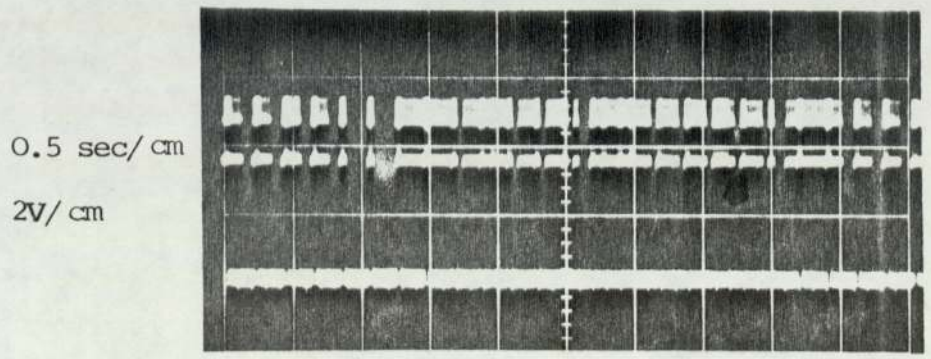


Fig. 29. Timing diagram of Data

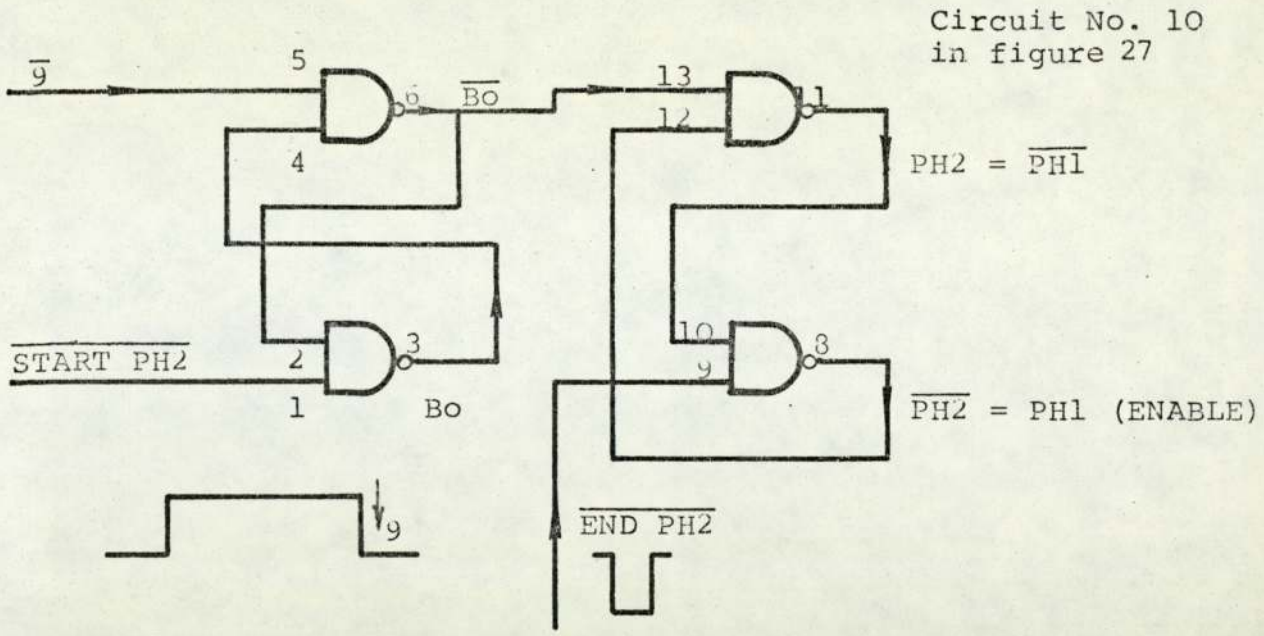


Fig.30. Pulse width modulation.

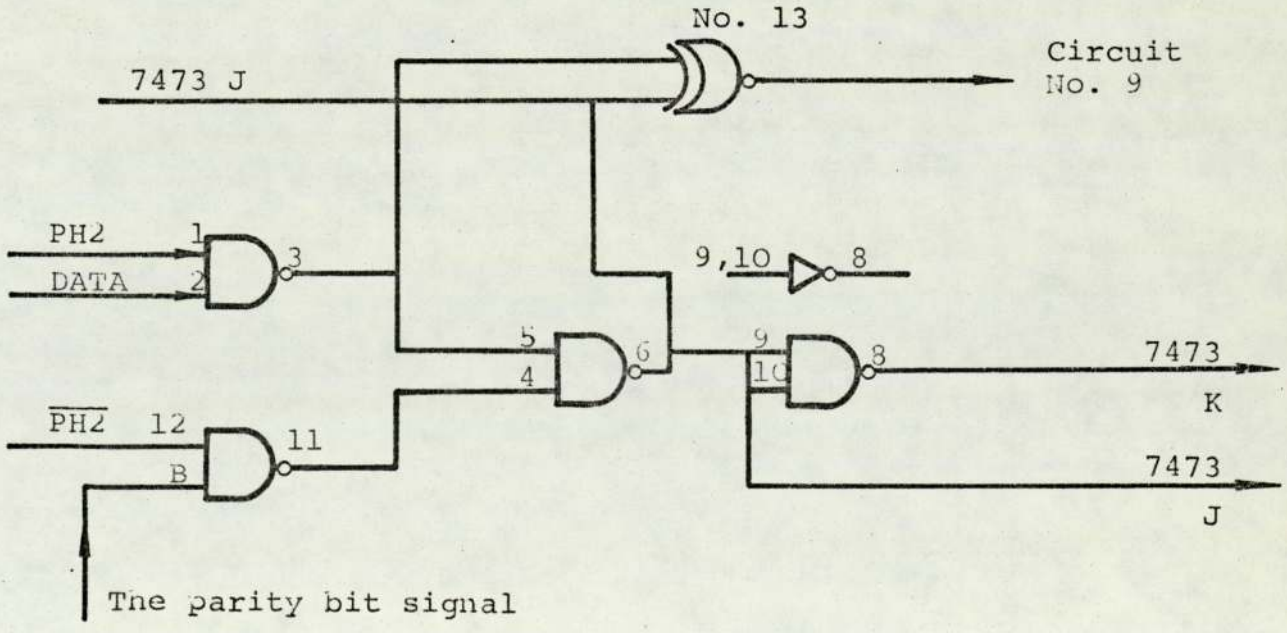


Fig.31. Pulse width modulation.

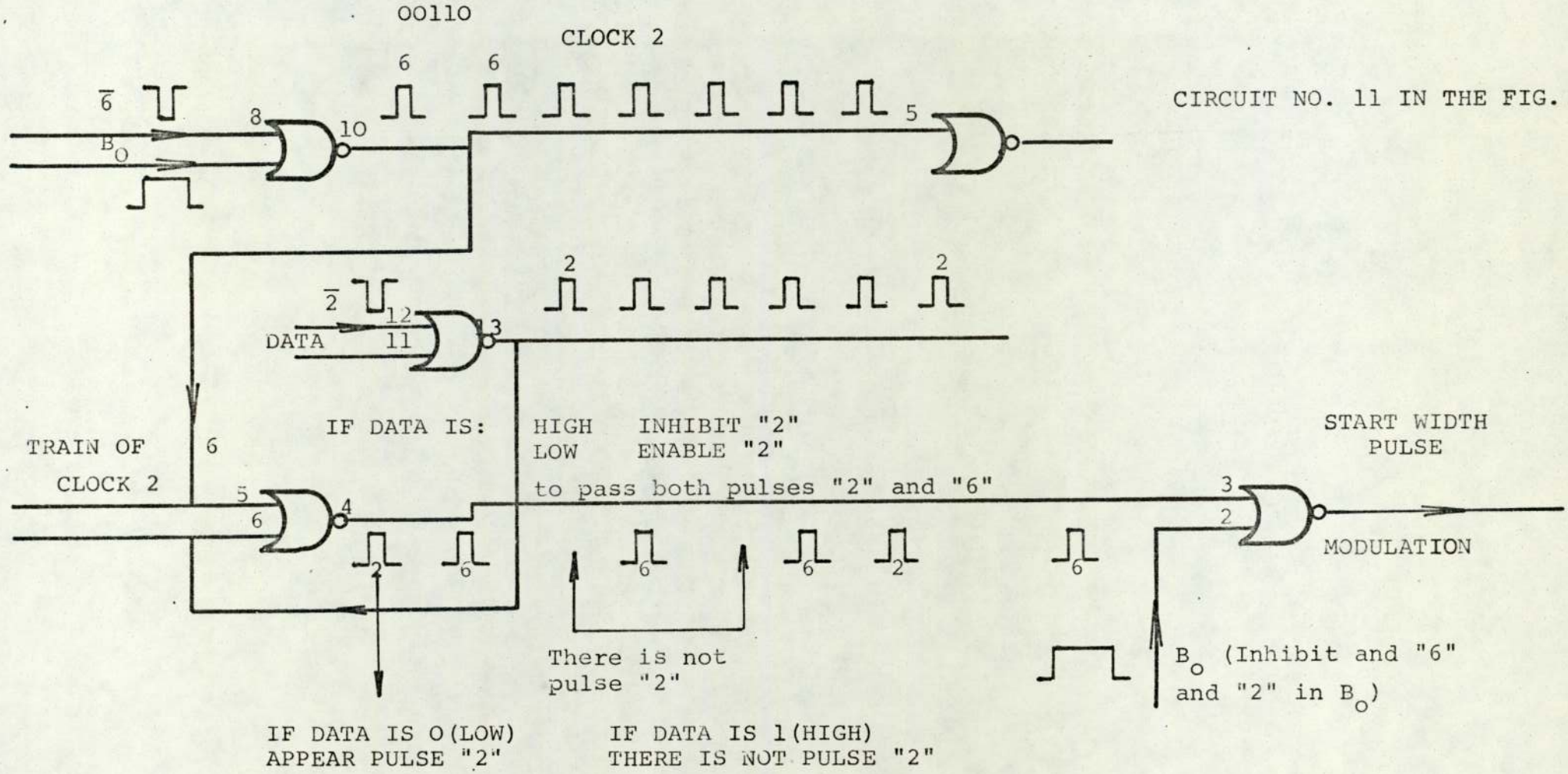


Fig. 32. Output of pulse width modulation.

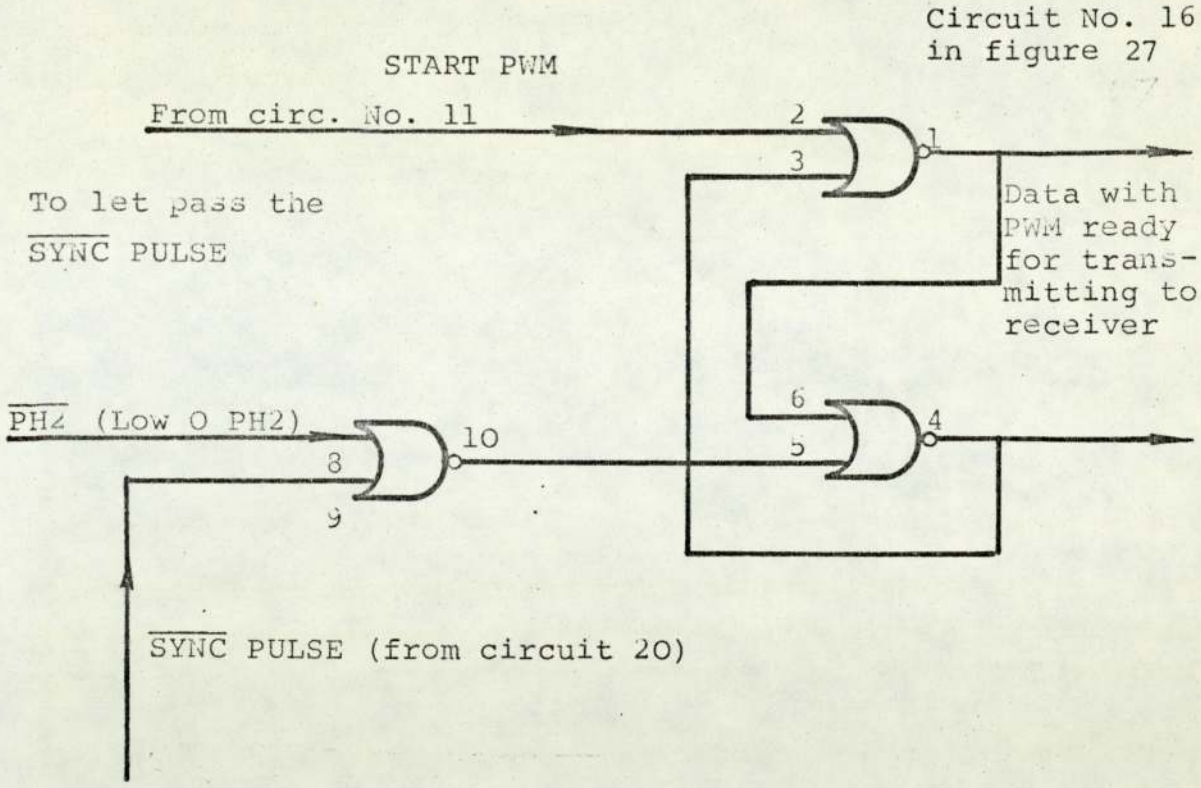


Fig. 33. The last function of the modulator unit.

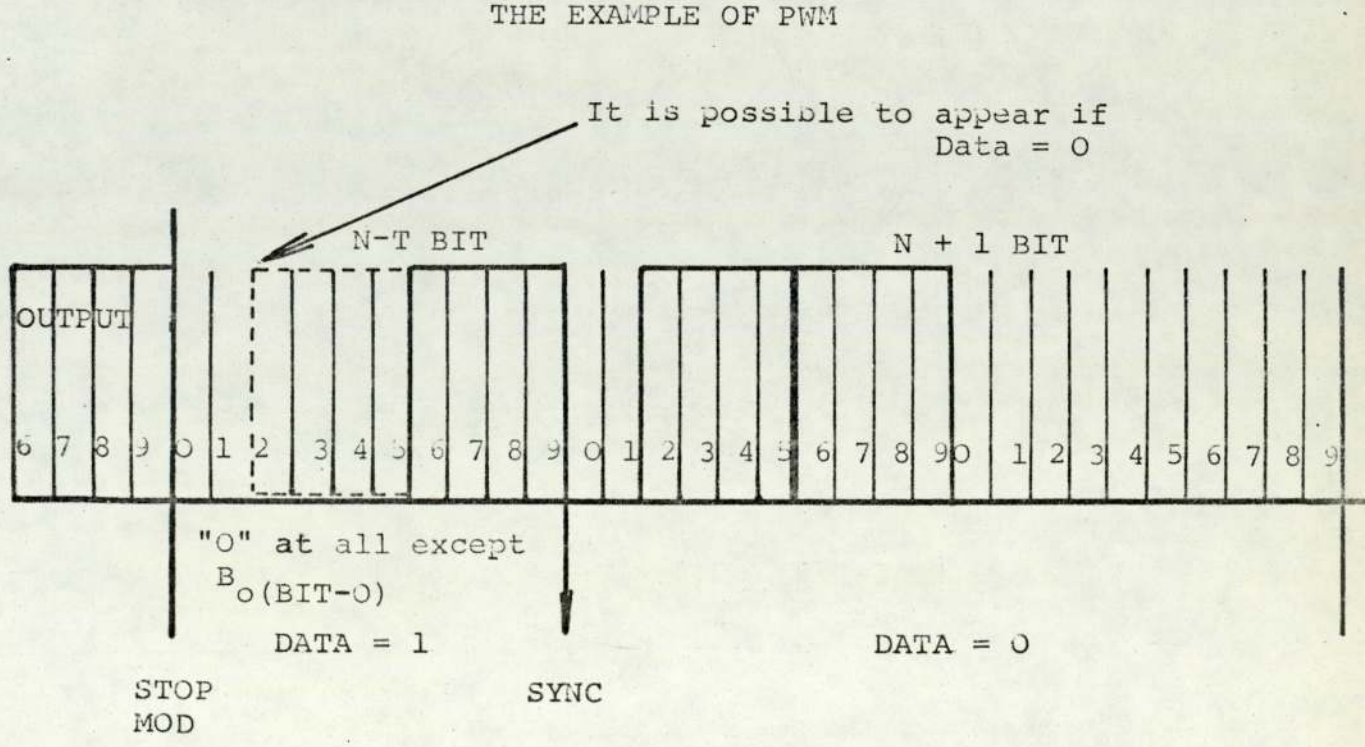


Fig. 34. The example of P.W.M.

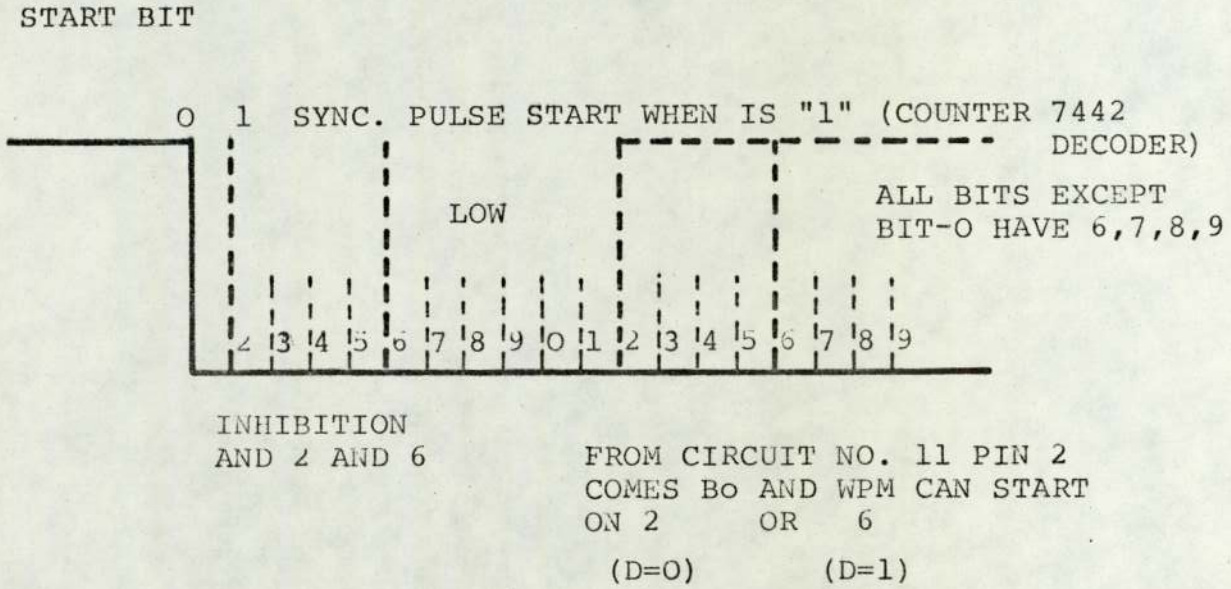


Fig. 35. The example of sync. pulse

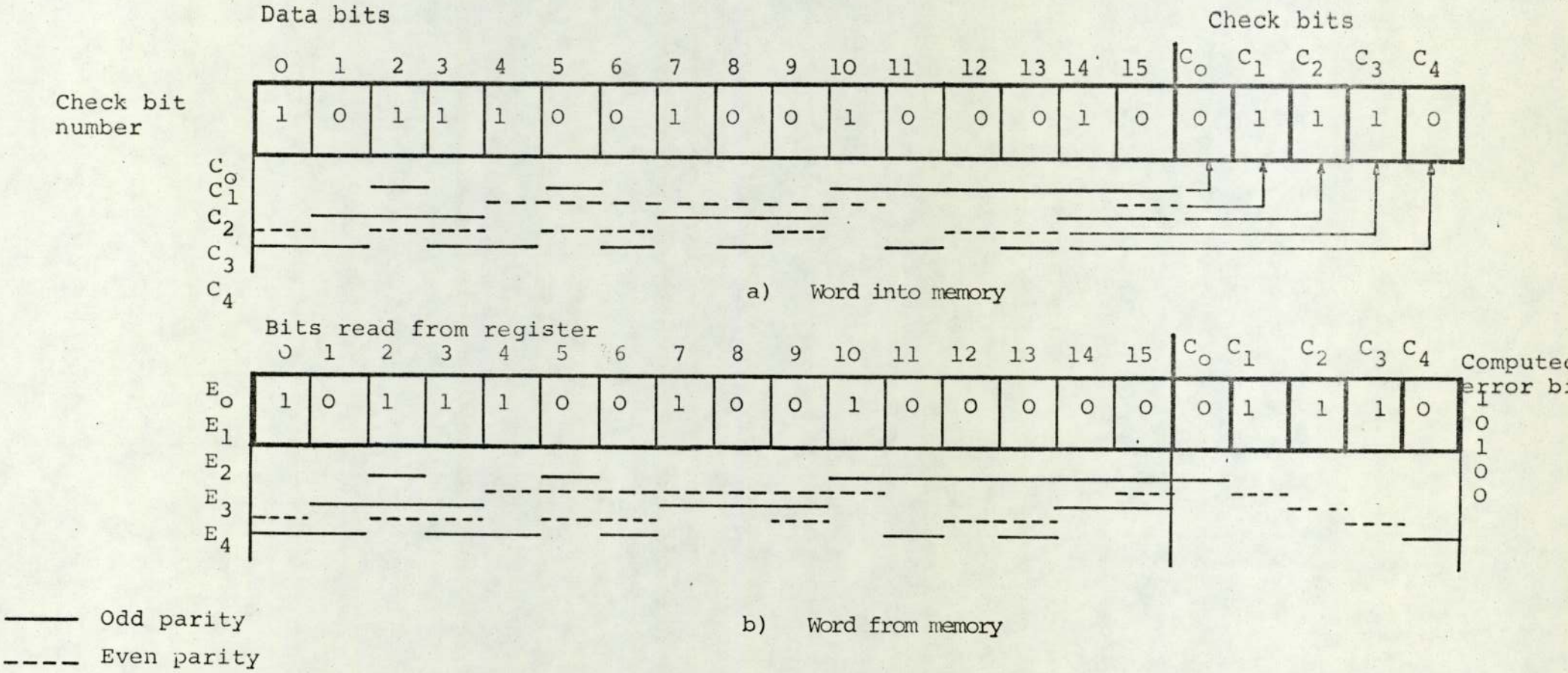


Fig. 36. The example of the error checking.

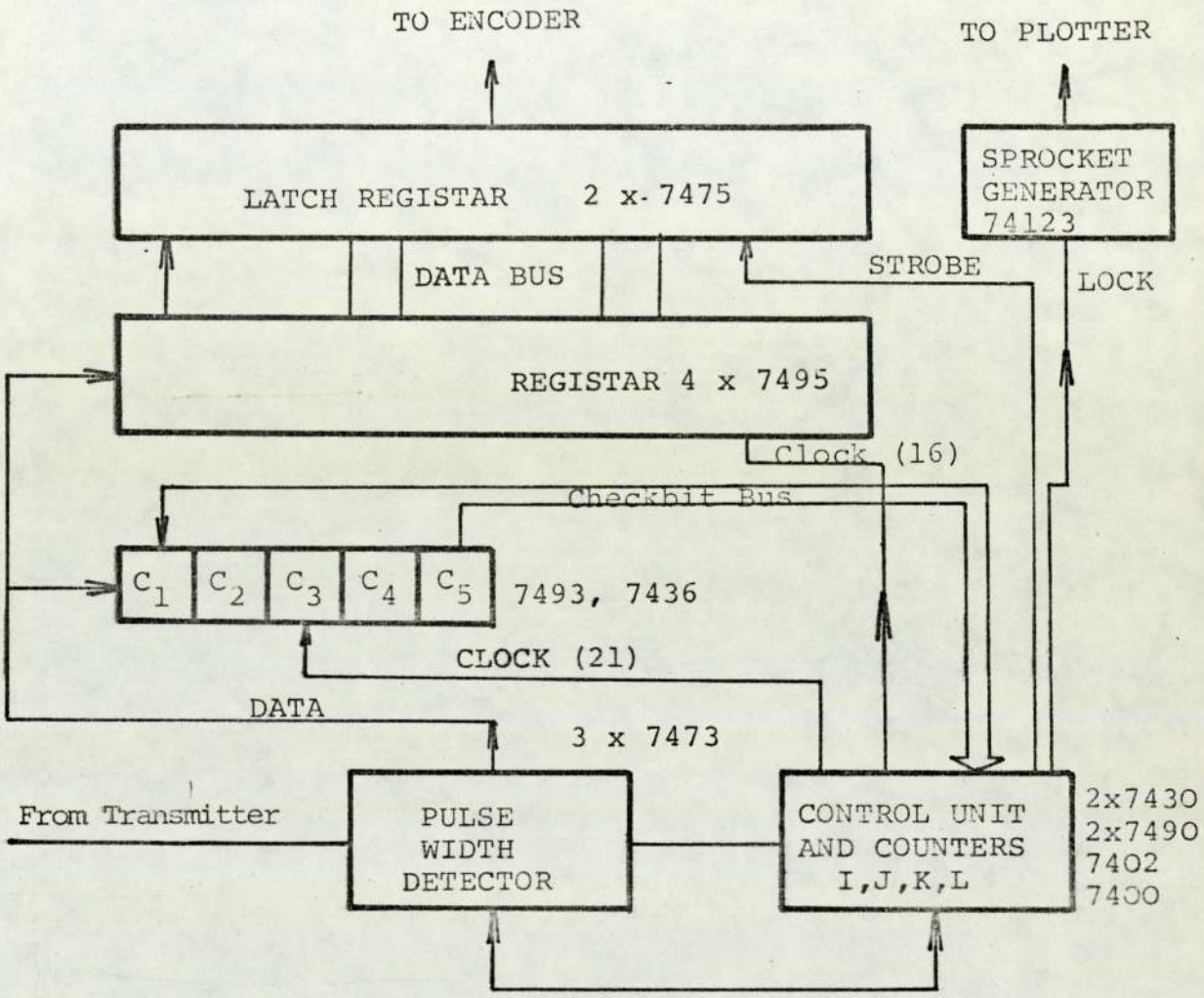


Fig. 37. The block diagram of the receiver.

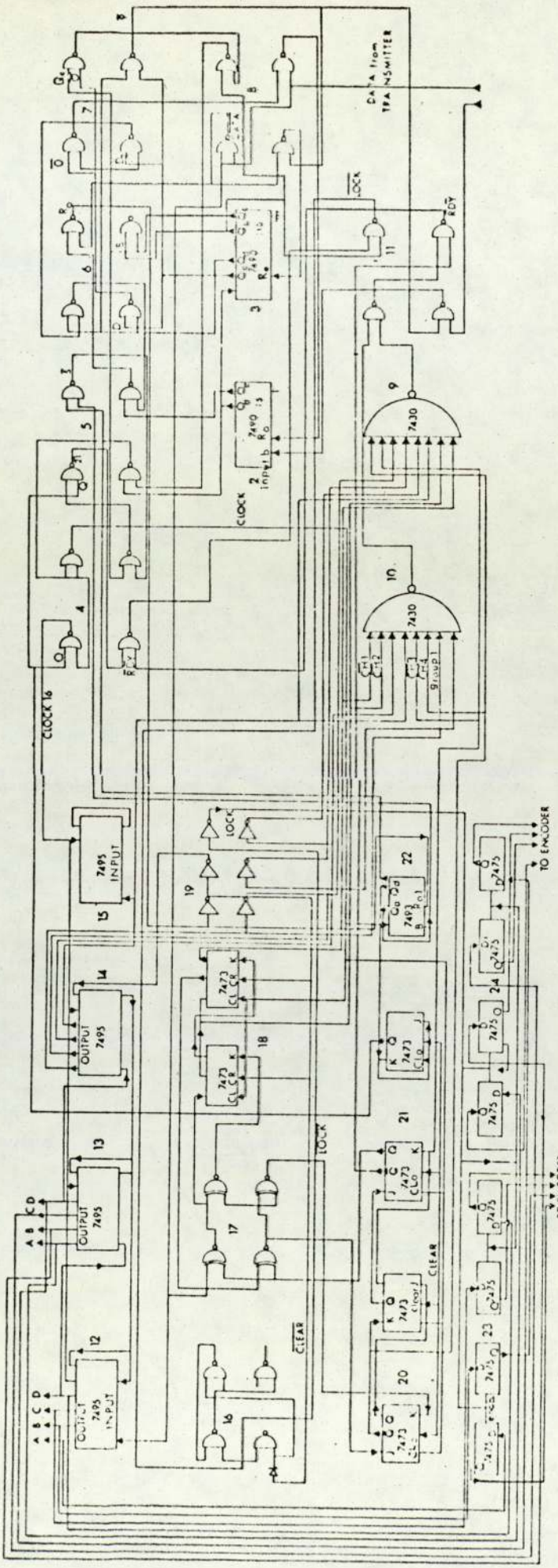


Fig. 38. Electrical diagram of the receiver.

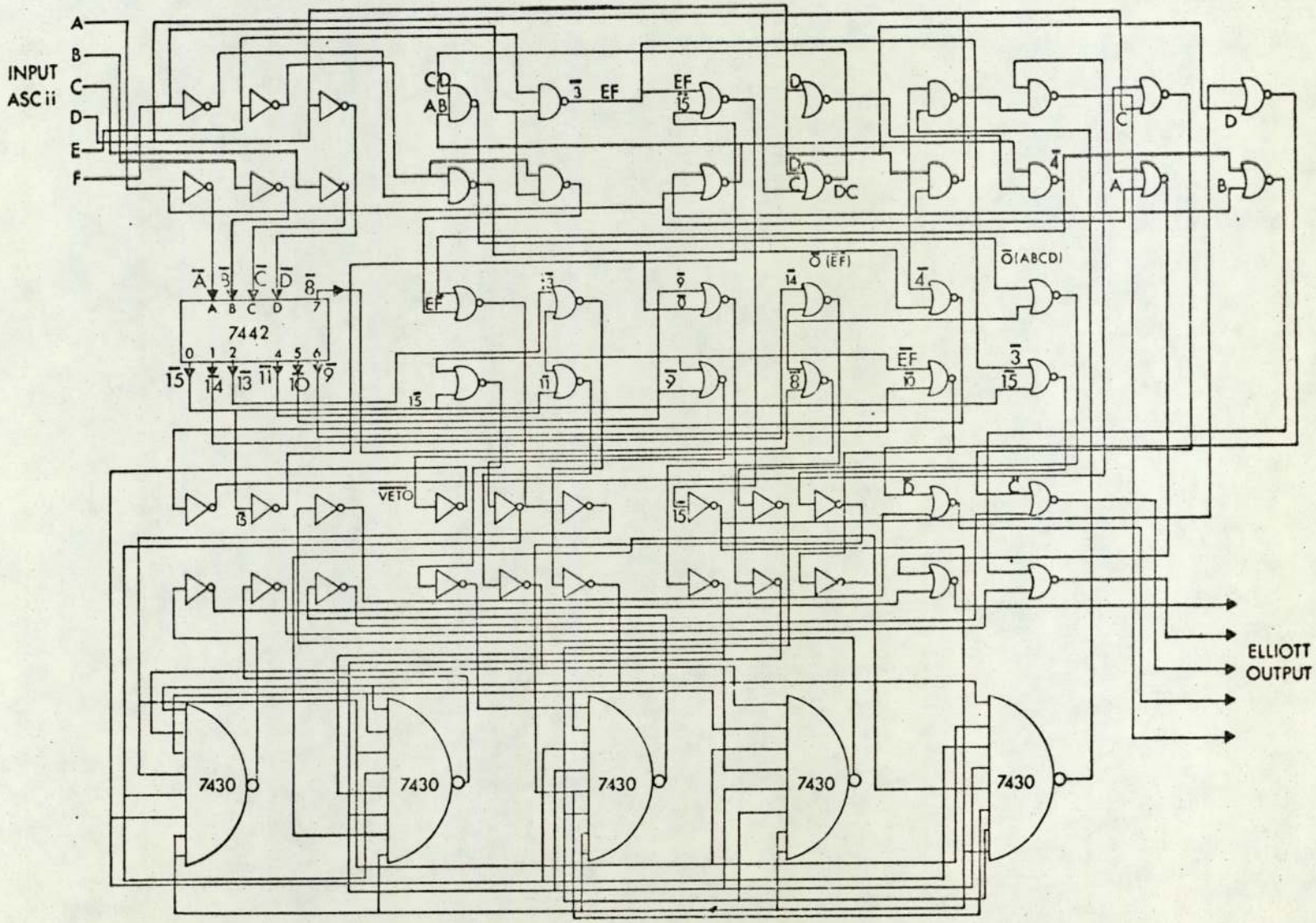


Fig. 39. Electrical diagram of the Encoder.

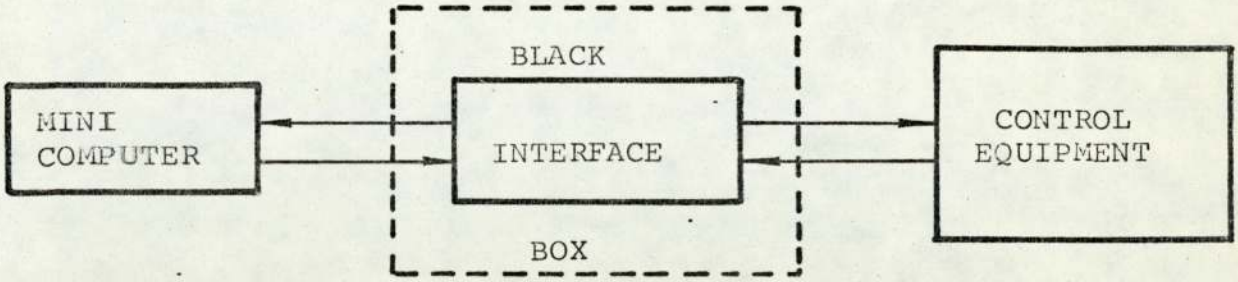


Fig. 40. Interface

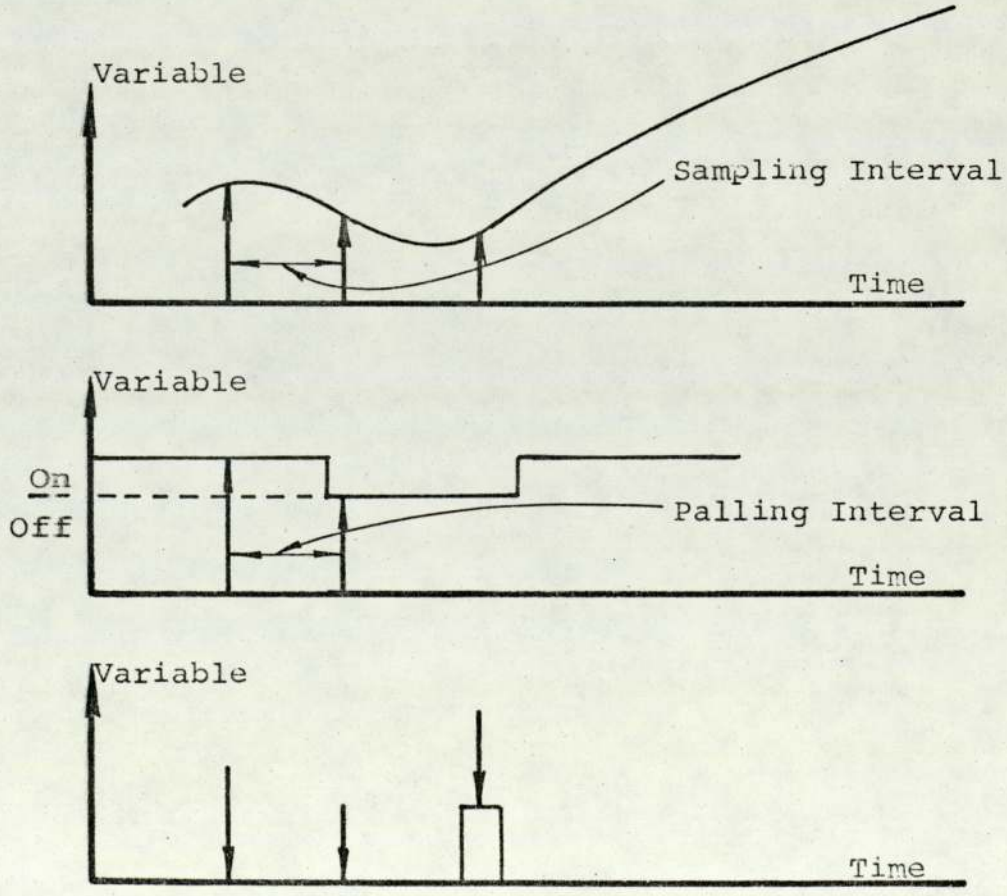


Fig. 41. Time delay.

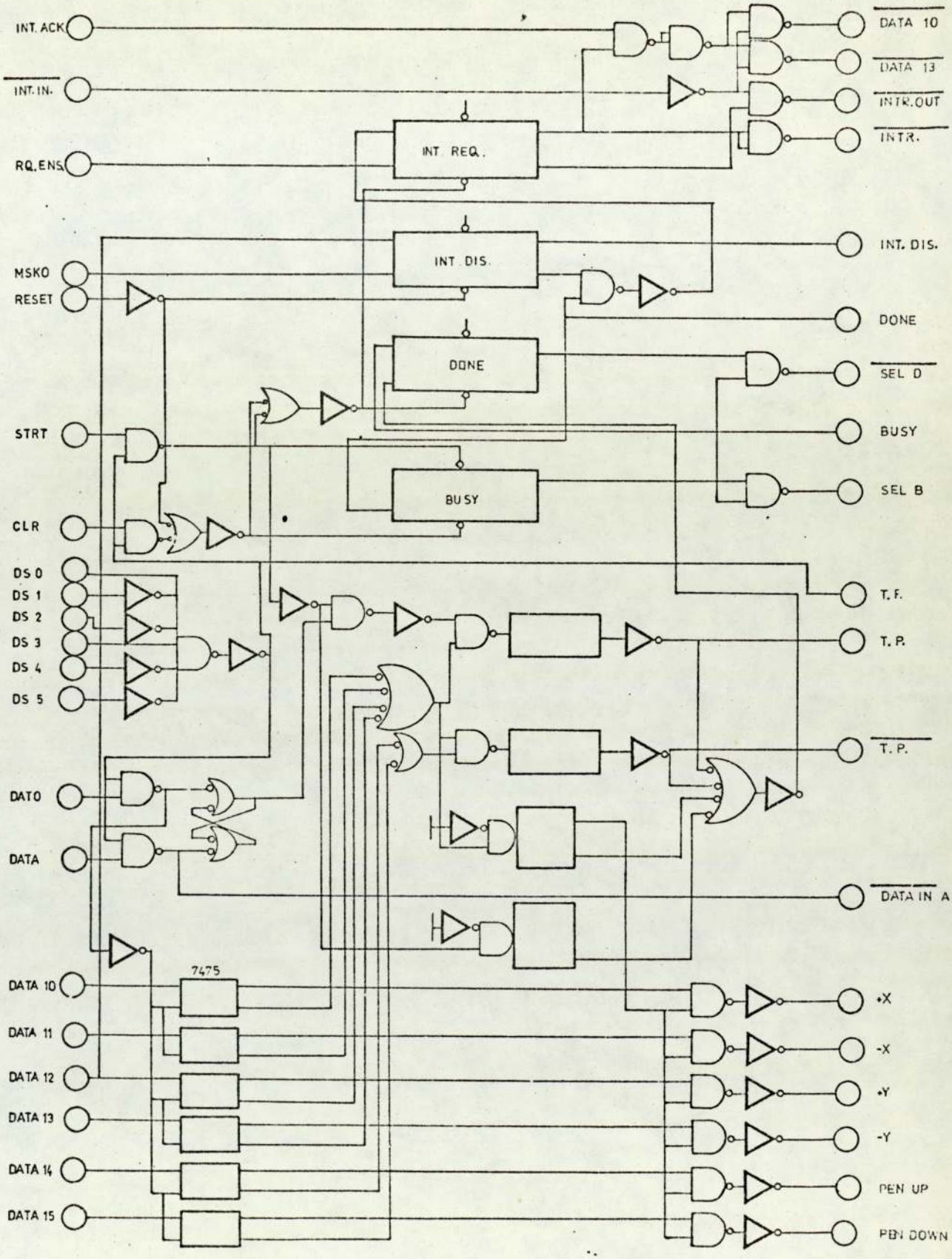


Fig. 42. General interface board

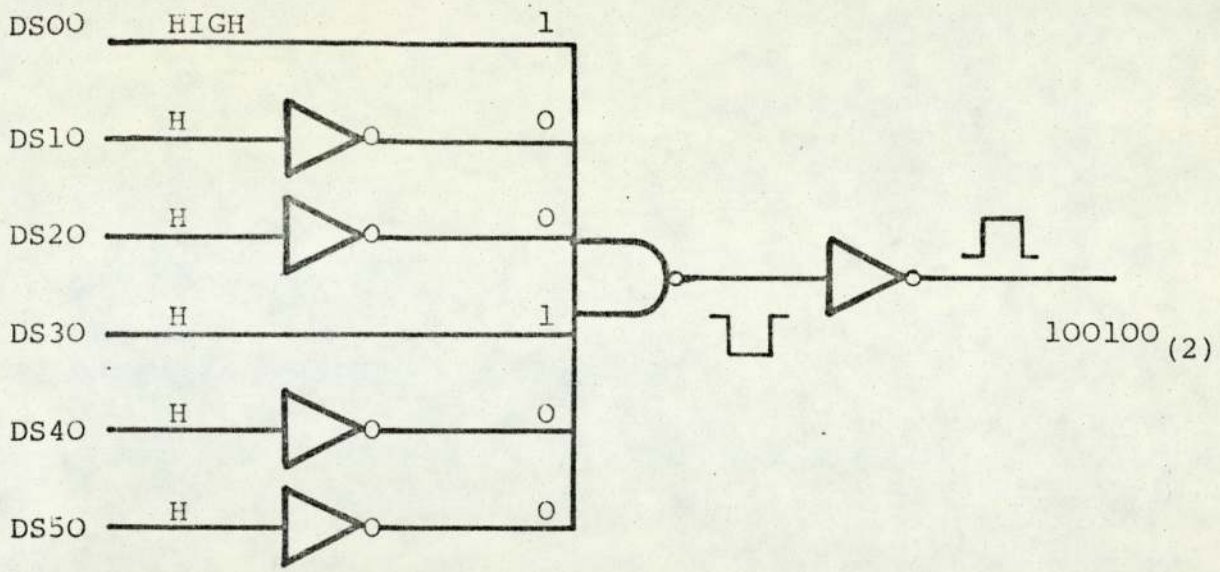


Fig. 43. Address part.

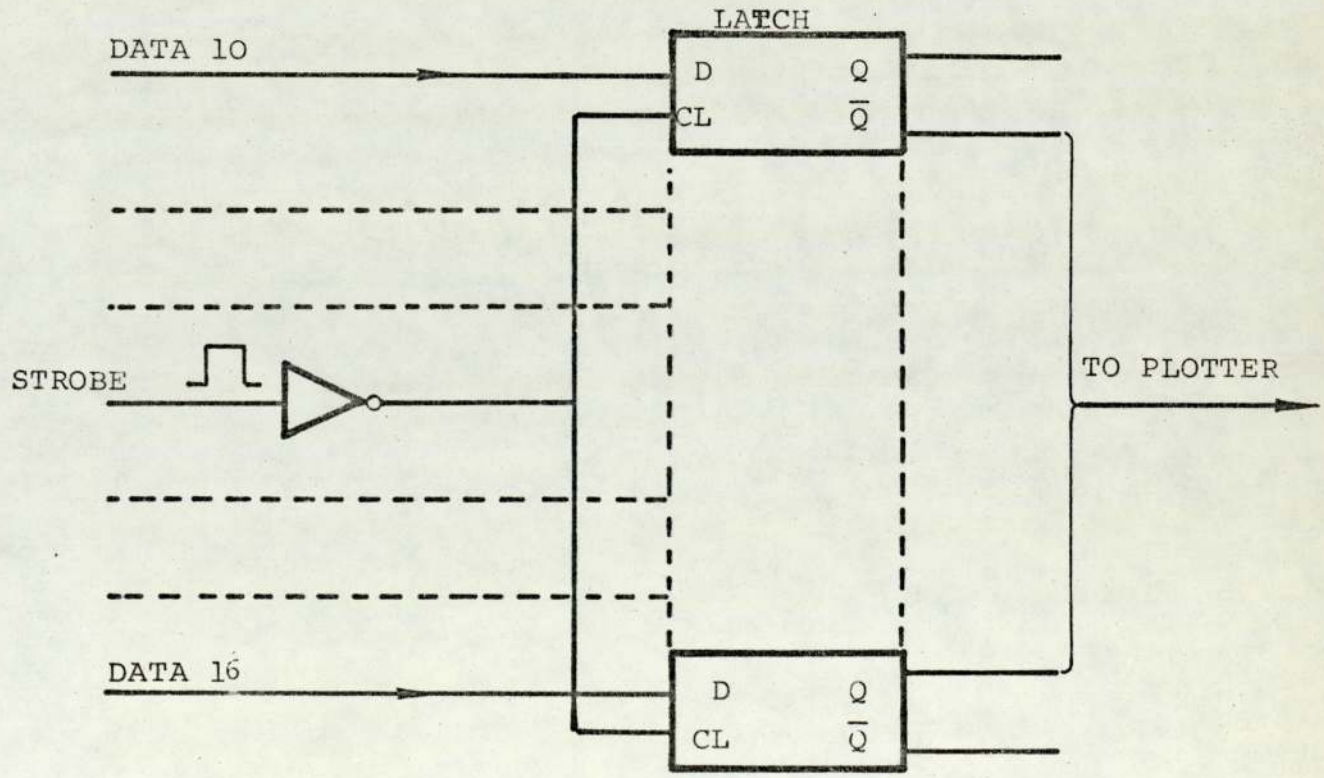


Fig. 44. Latch circuit.

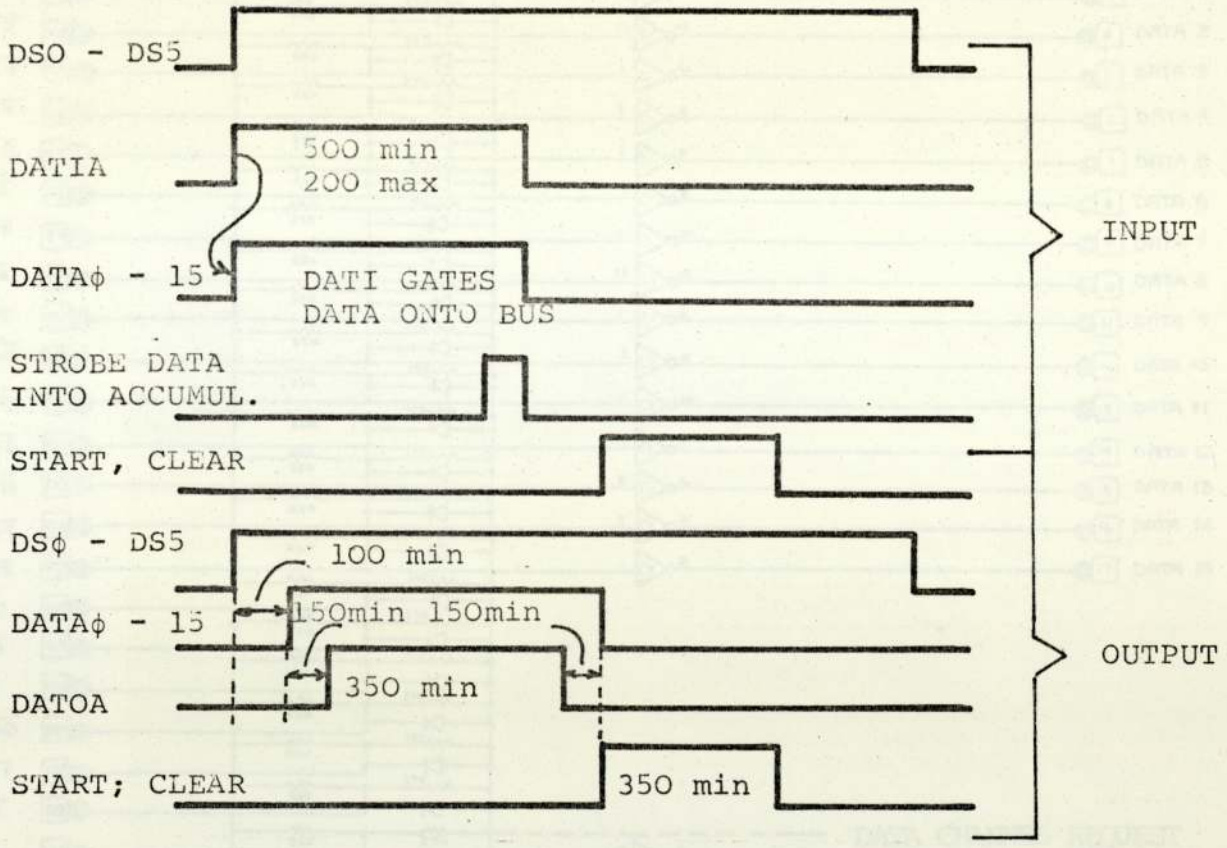


Fig. 45. Timing Diagram in-out instructions.

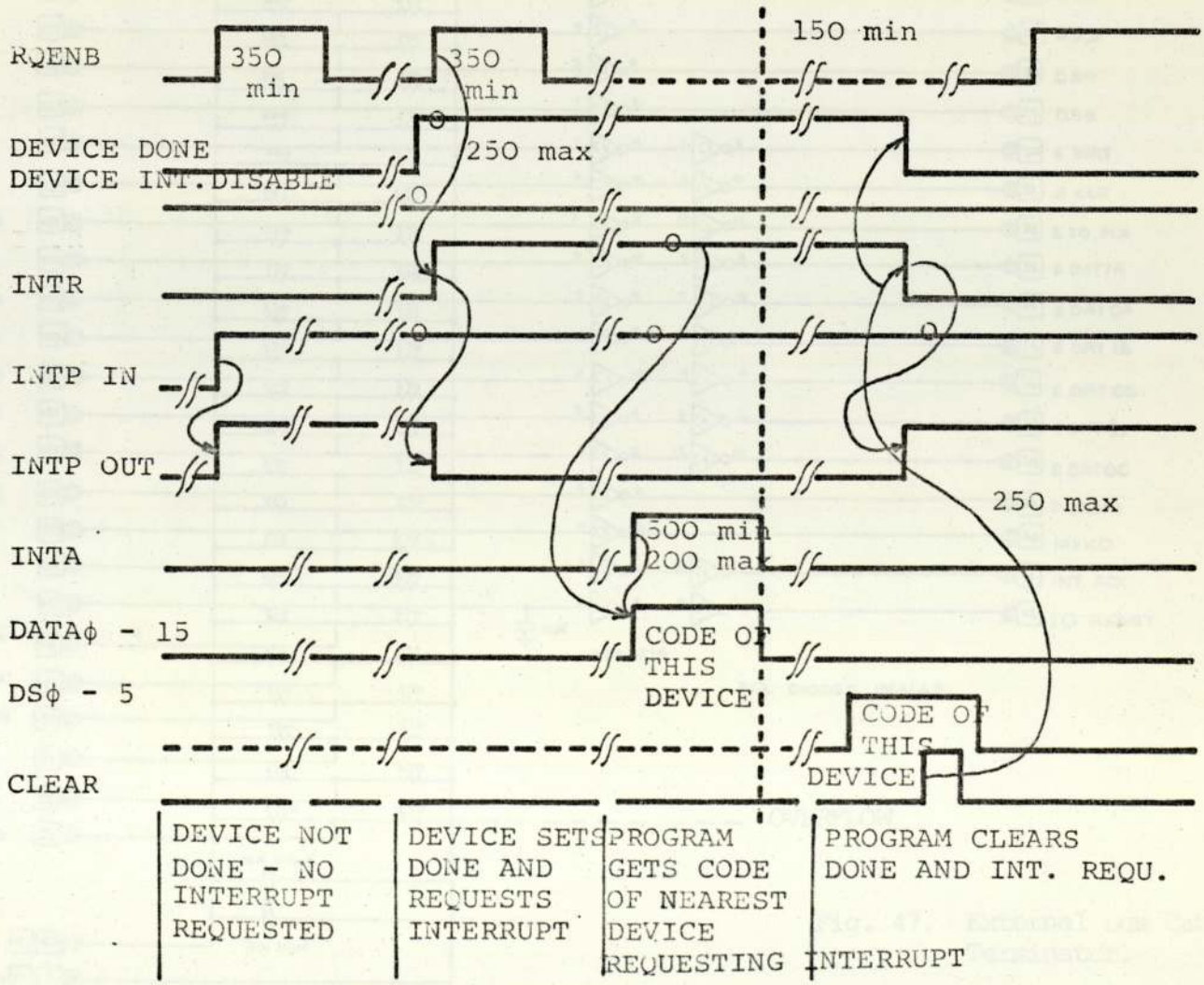


Fig. 46. Interrupt timing diagram.

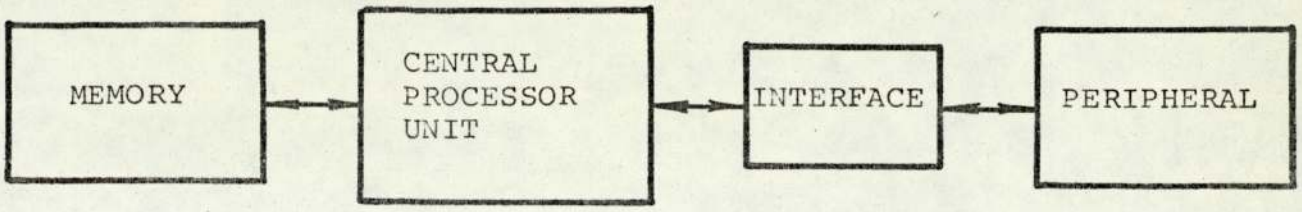


Fig. 48. Block diagram of Mini-computer

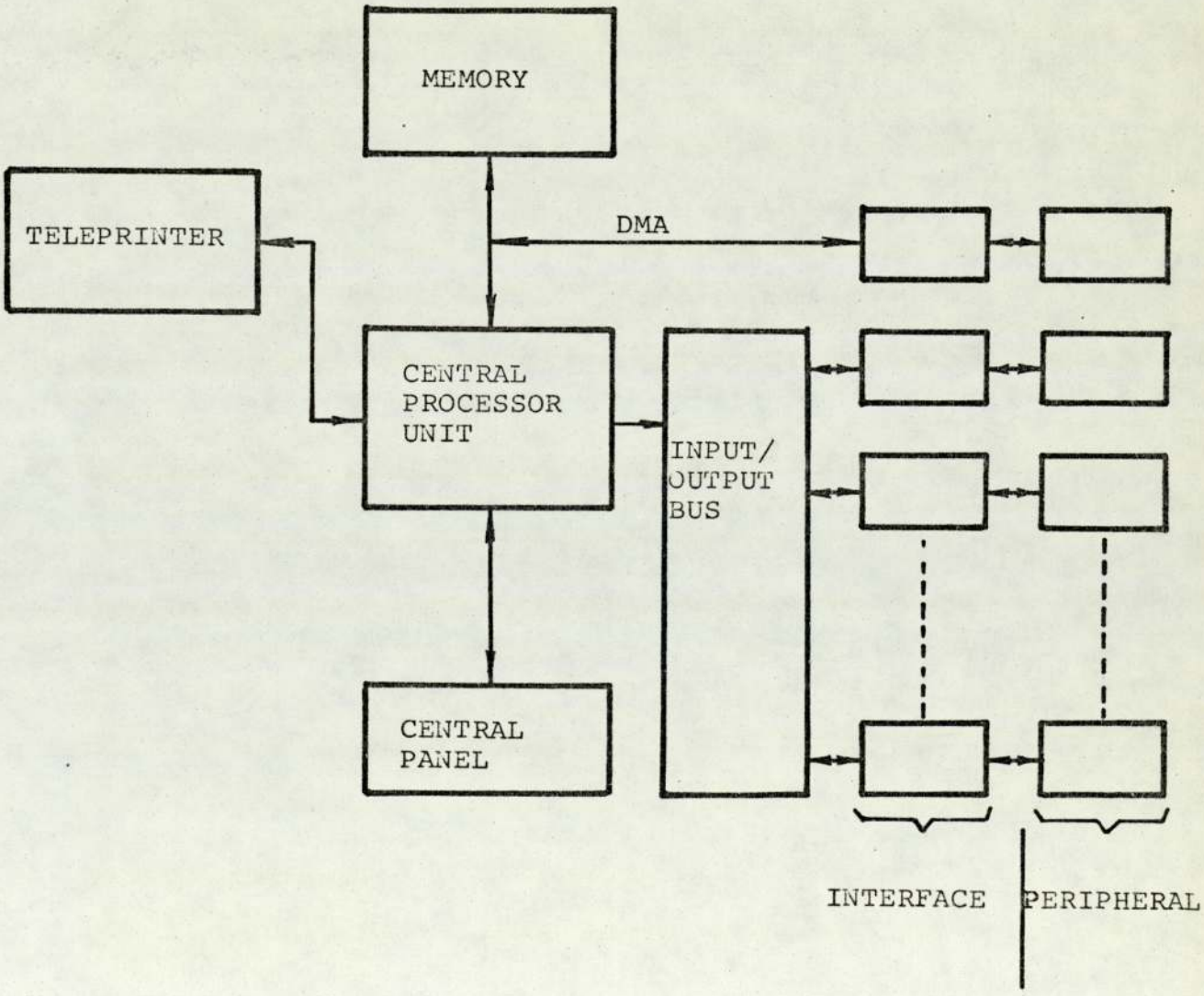


Fig. 49. Mini-computer System.

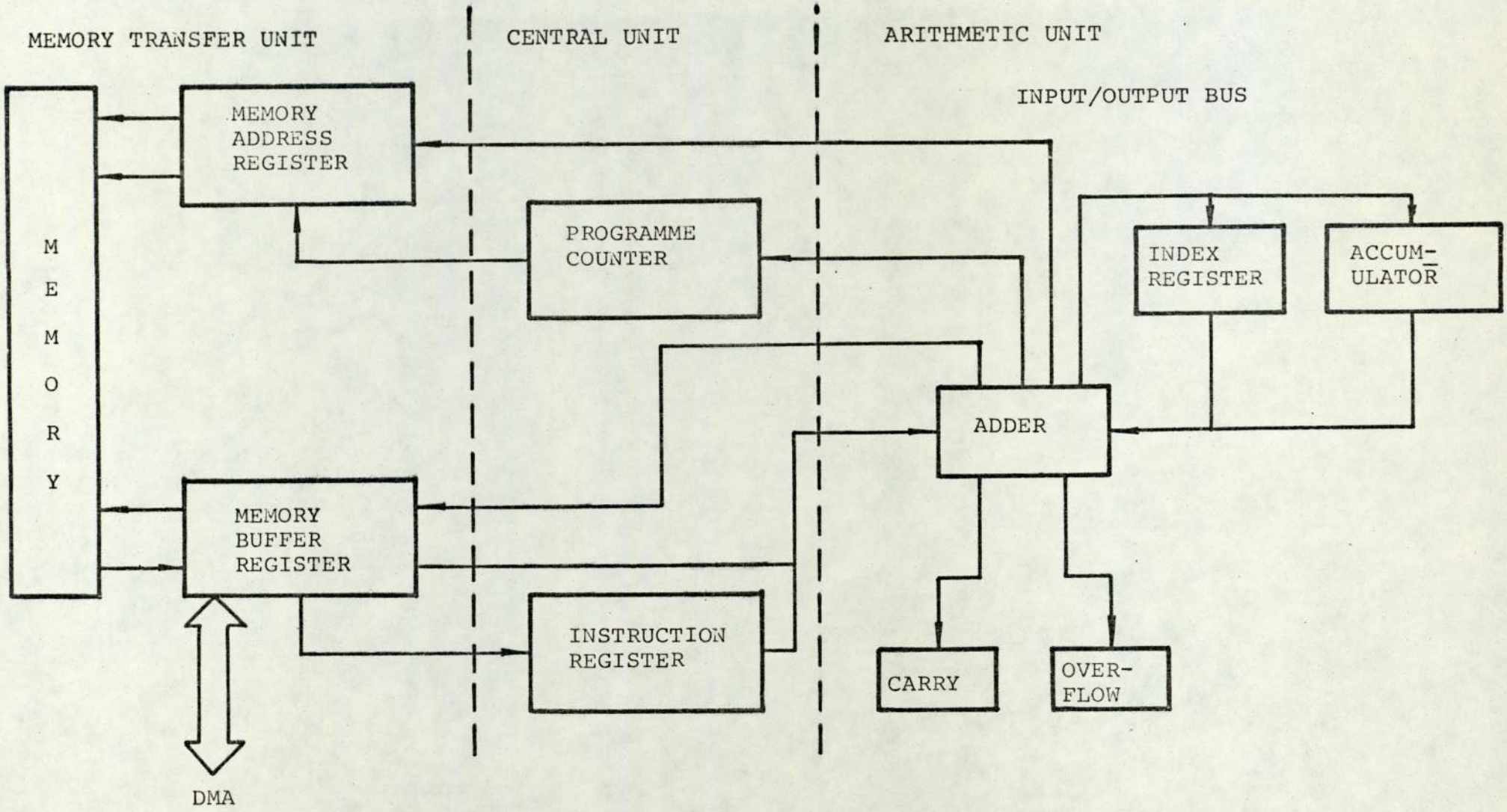


Fig. 50. Block diagram of Central Processor

MINI COMPUTER NOVA 1220

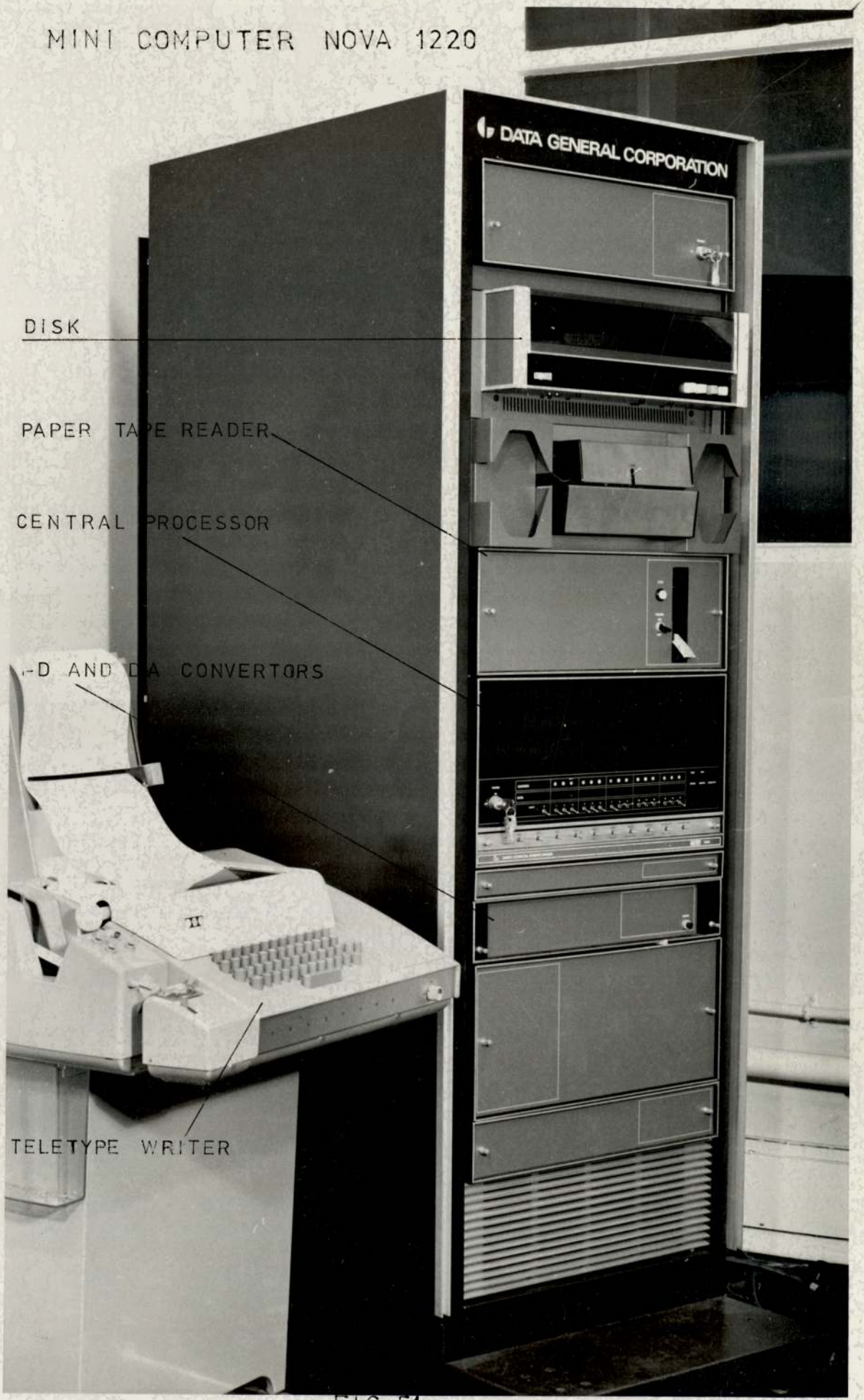
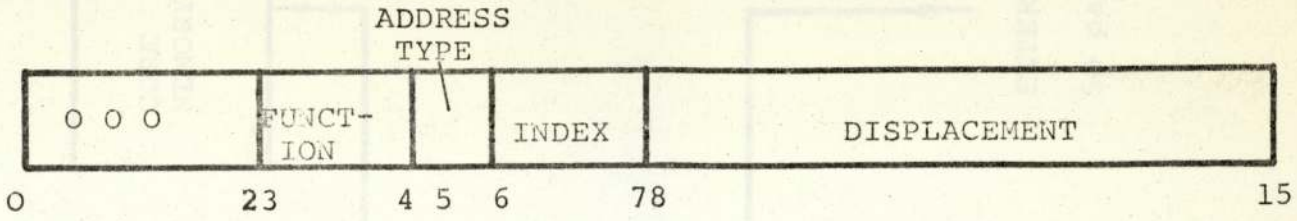
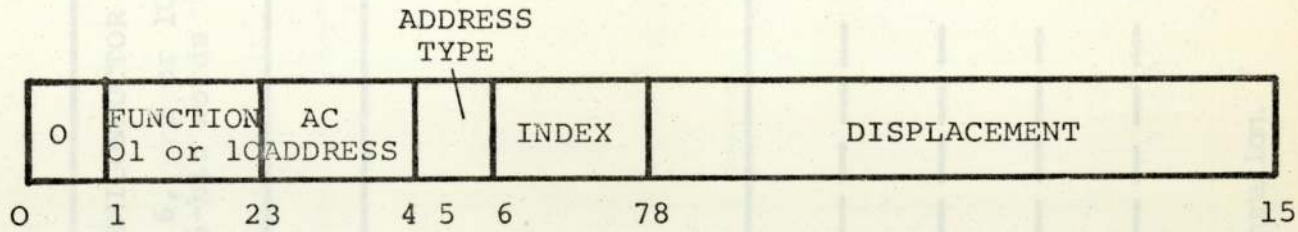


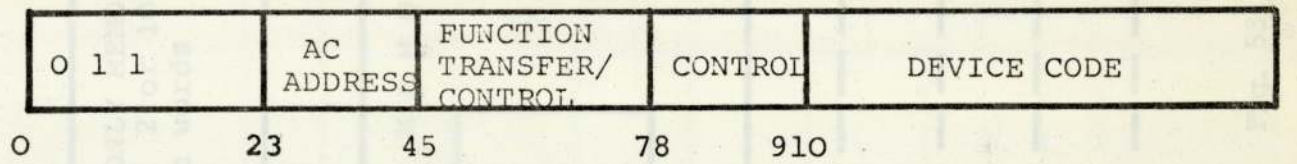
FIG 51



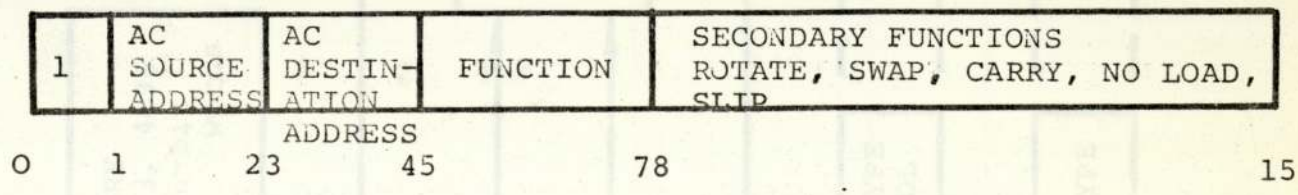
JUMP AND MODIFY MEMORY FORMAT



MOVE DATA FORMAT



IN-OUT FORMAT



ARITHMETIC AND LOGIC FORMAT

Fig. 52. Data words.

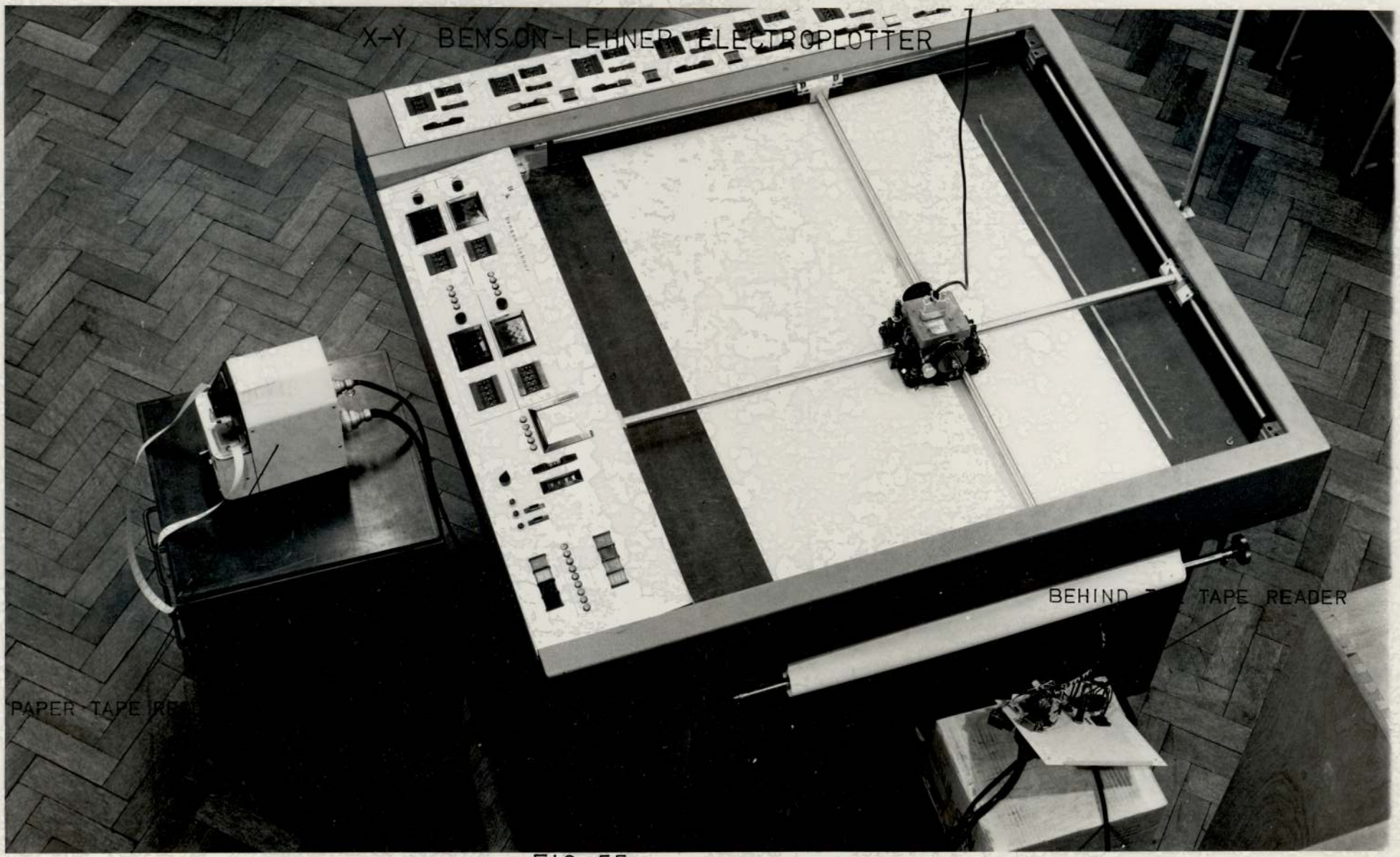

X-Y BENSON-LEHNER ELECTROPLOTTER
BEHIND
TAPE READER
PAPER TAPE

FIG 55

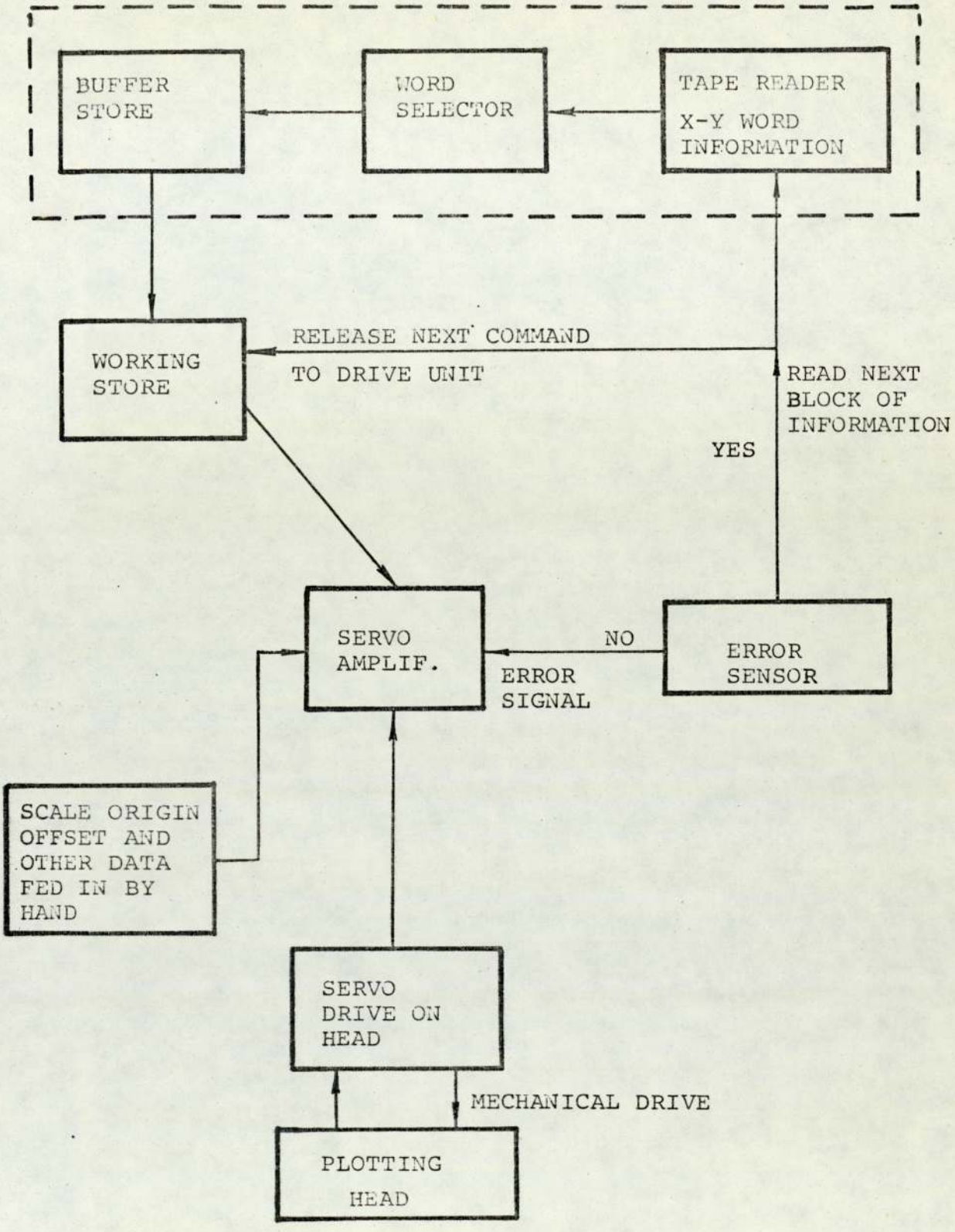
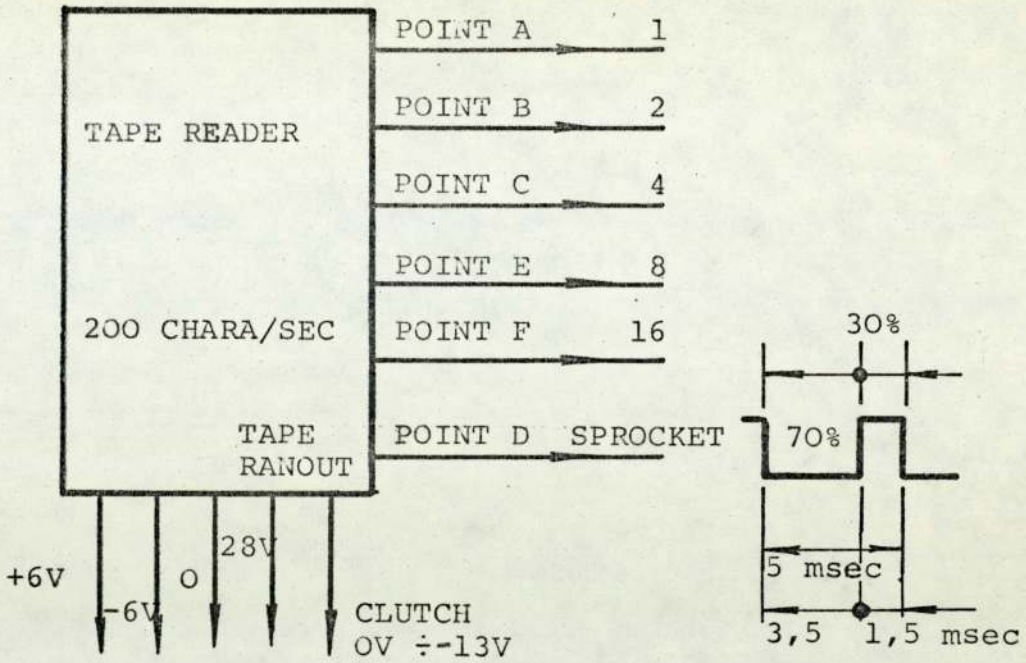


Fig. 56. Block Diagram of Plotter.



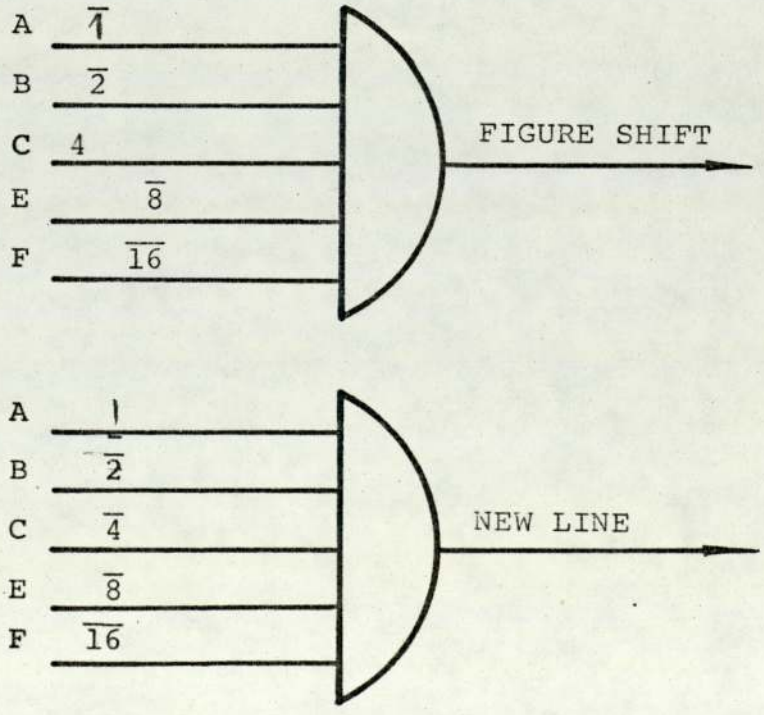


Fig. 57. Output from Tape Reader.

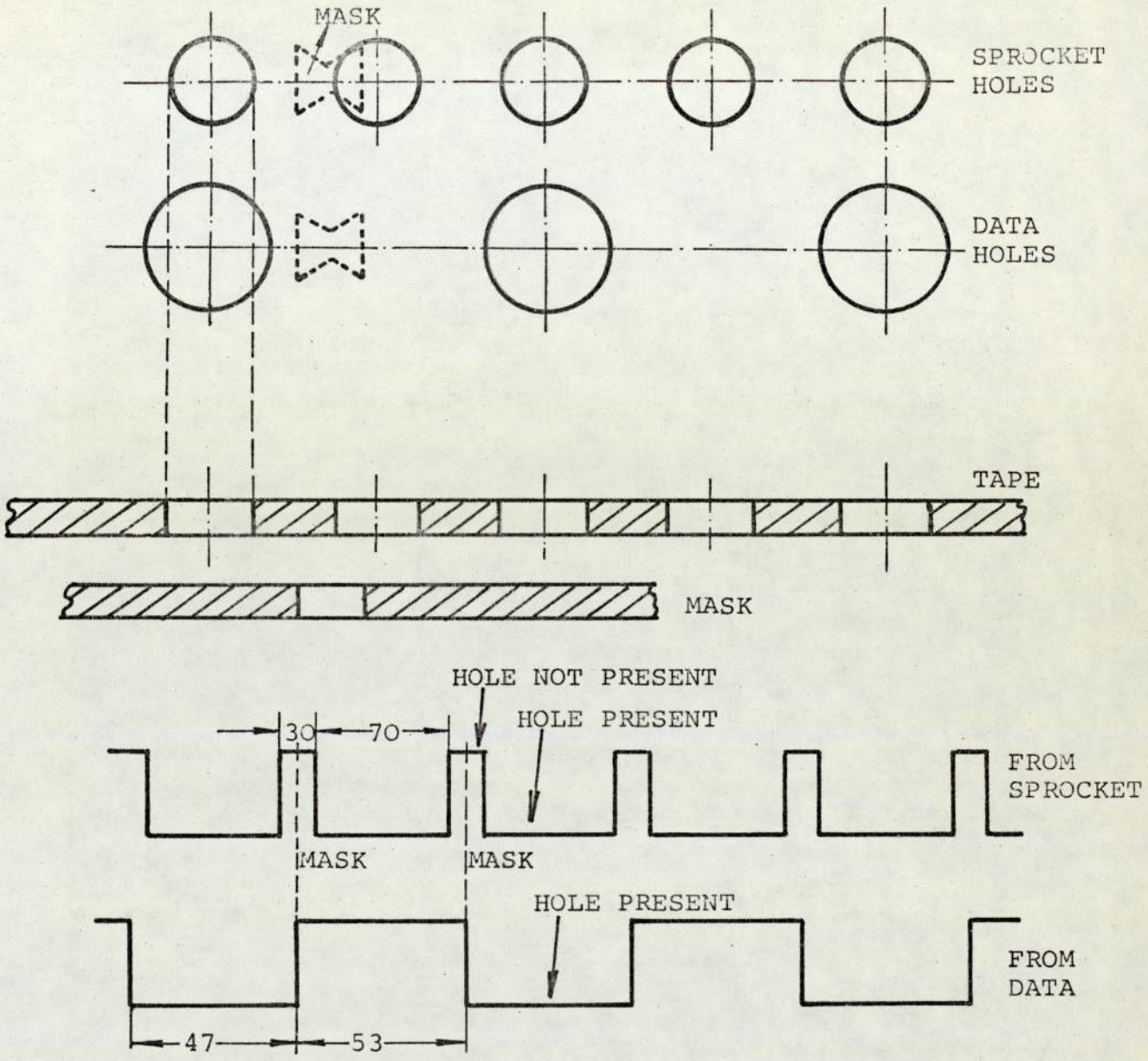


Fig. 58. Wave forms from paper tape.

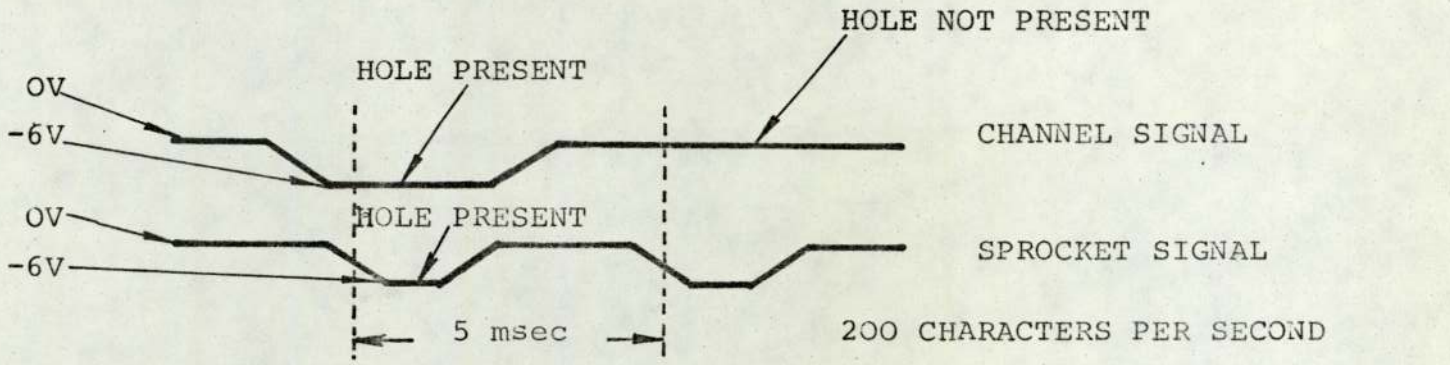


Fig. 59. Timing diagram of holes

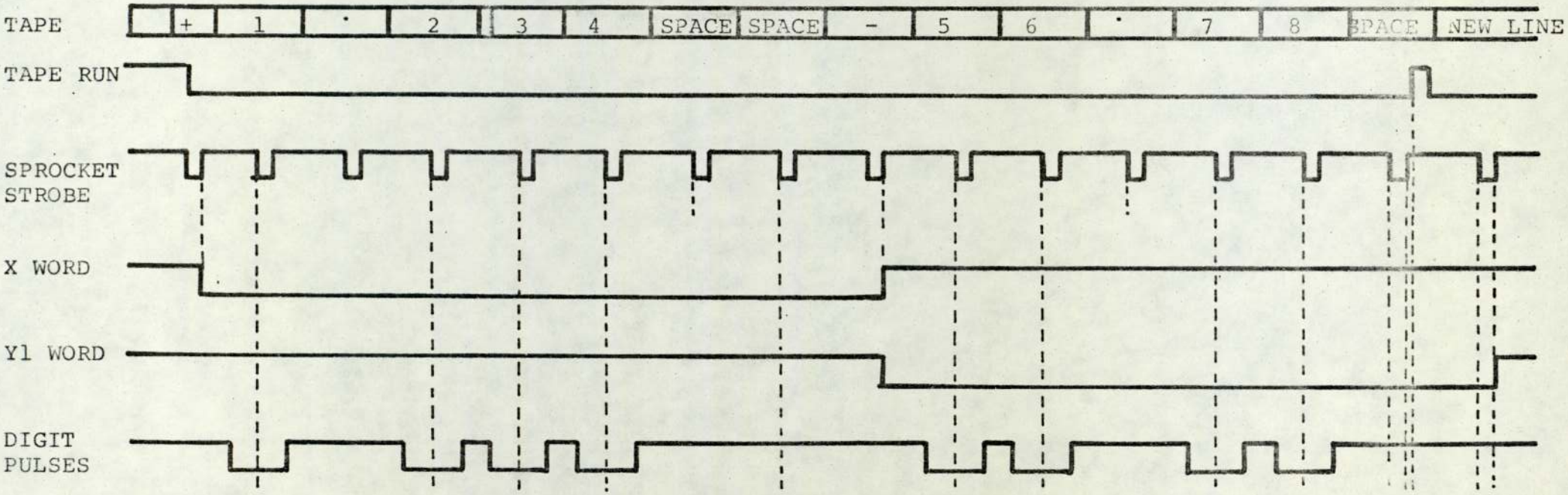


Fig. 60. Timing diagram of a programme.

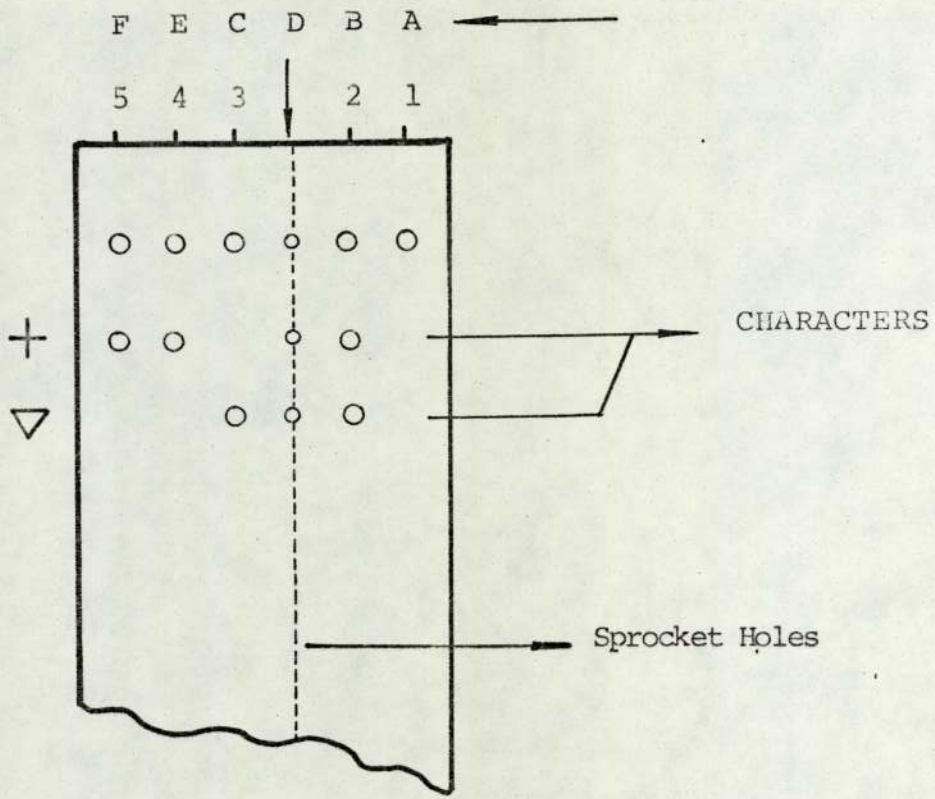
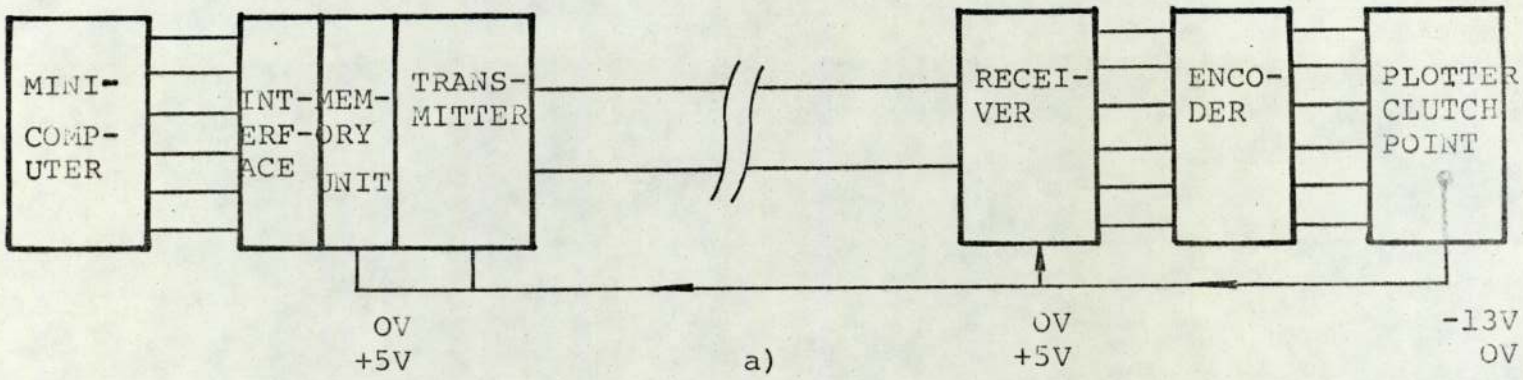
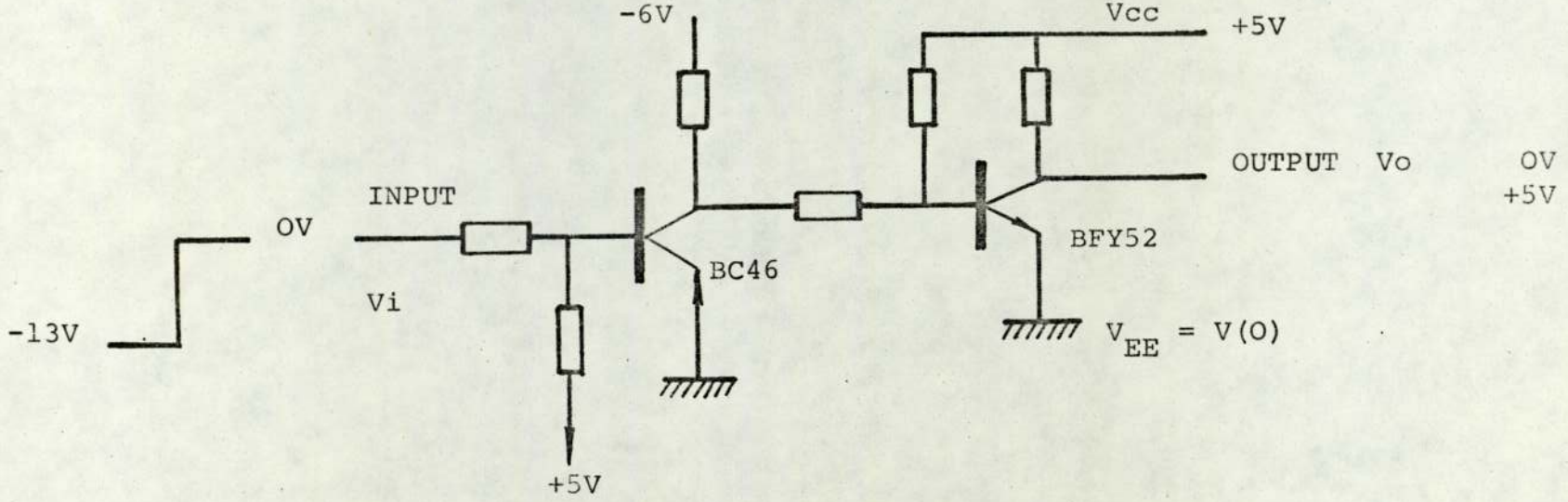


Fig.61. Representation of the paper tape

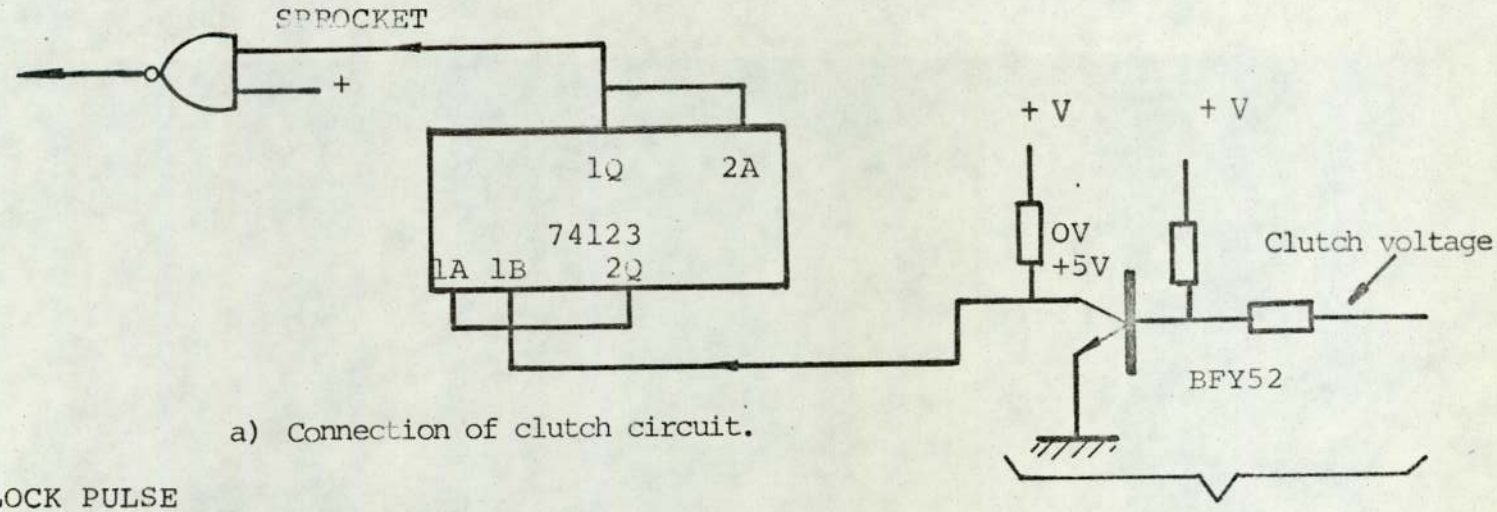


a)

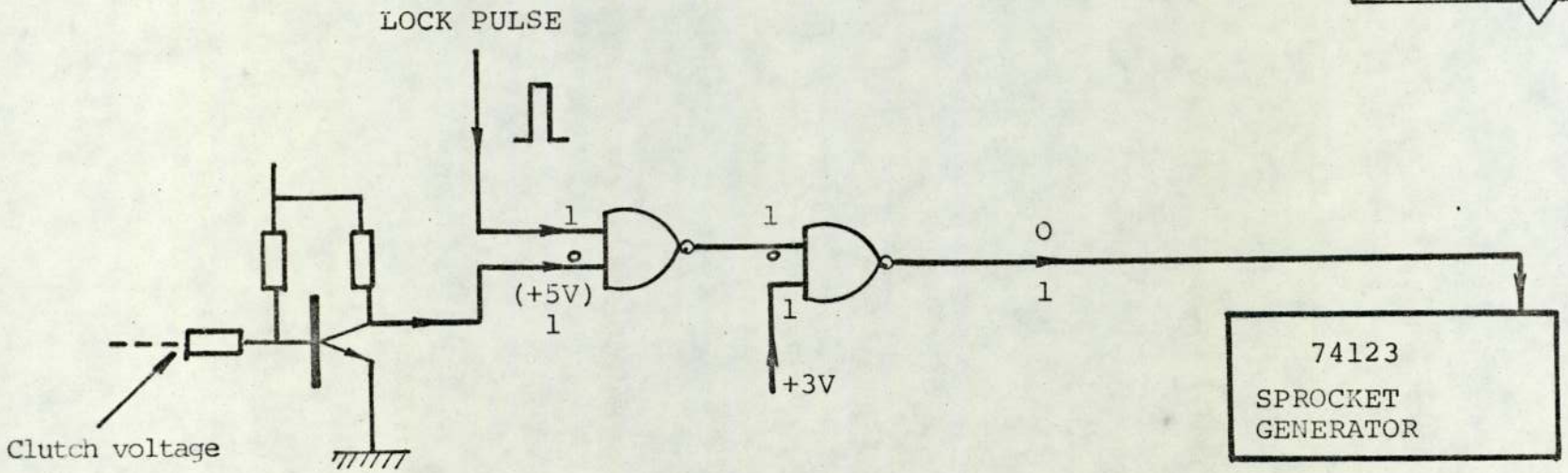


b)

Fig. 62. Clutch circuit.



a) Connection of clutch circuit.



b) The circuit for correct timing of sprocket pulses.

Fig. 63. Connection of clutch circuit and circuit for correct timing of sprocket pulses.

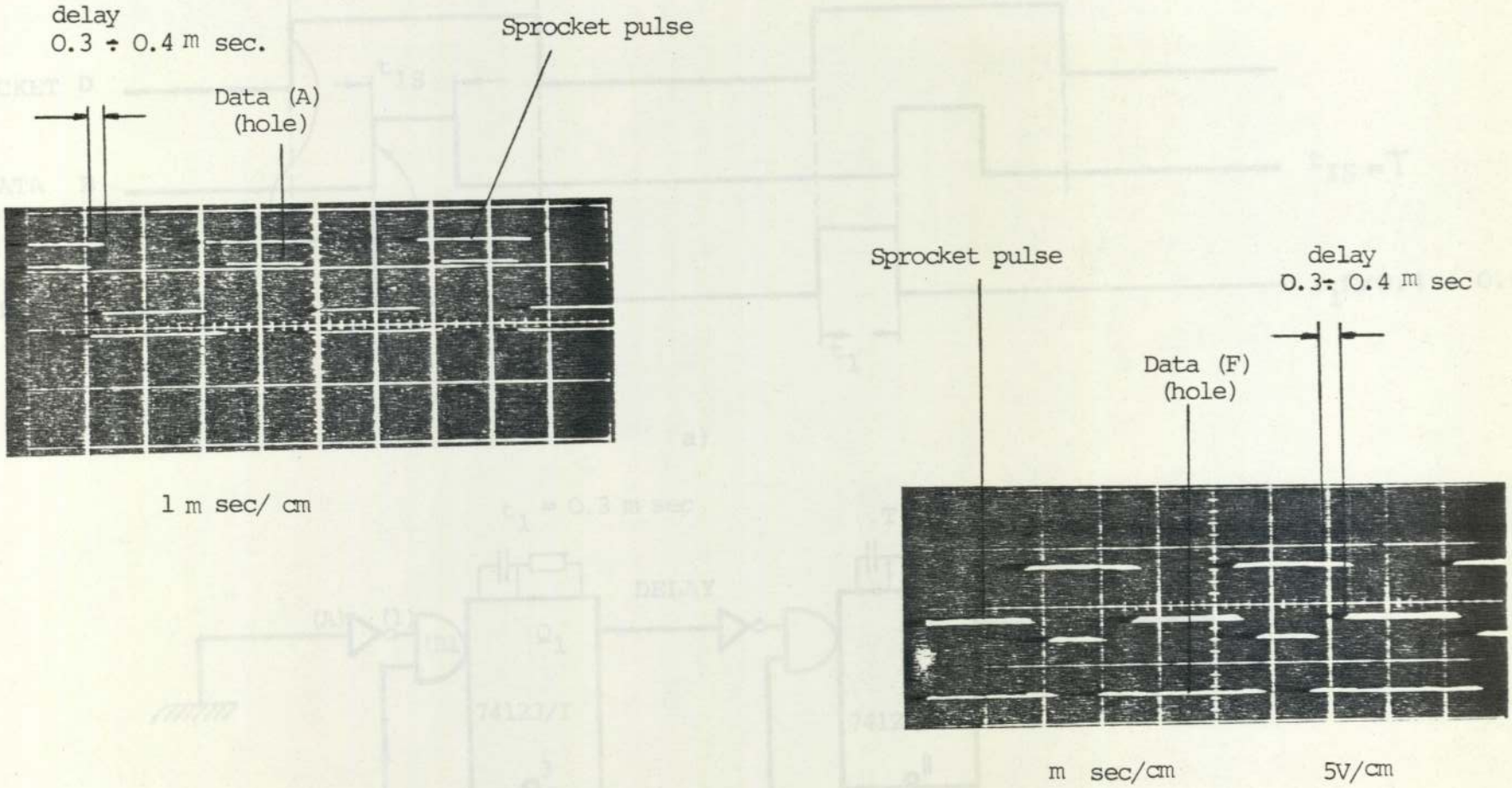


Fig. 64. Oscillogram of data and sprocket.

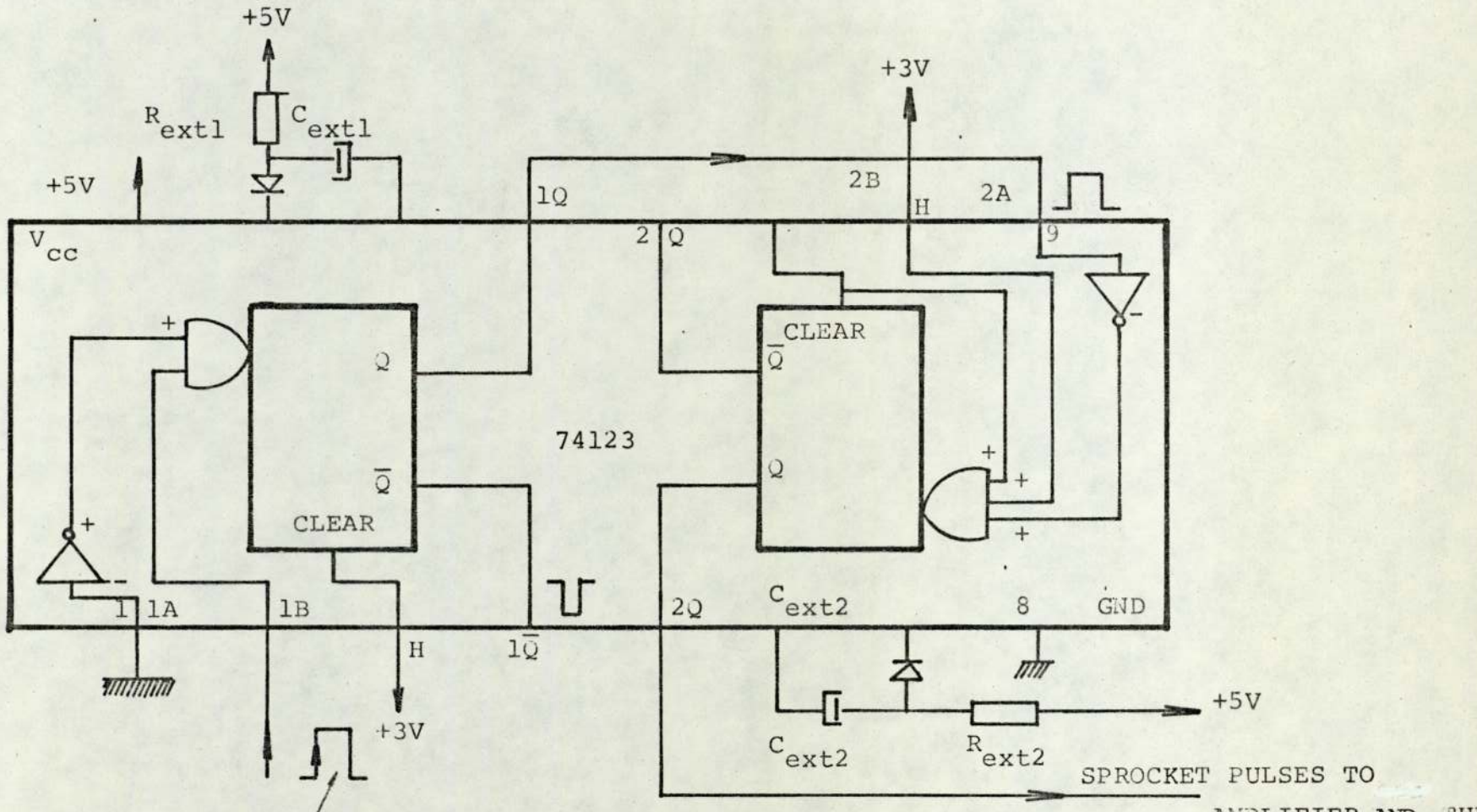


Fig.66. Sprocket Generator.

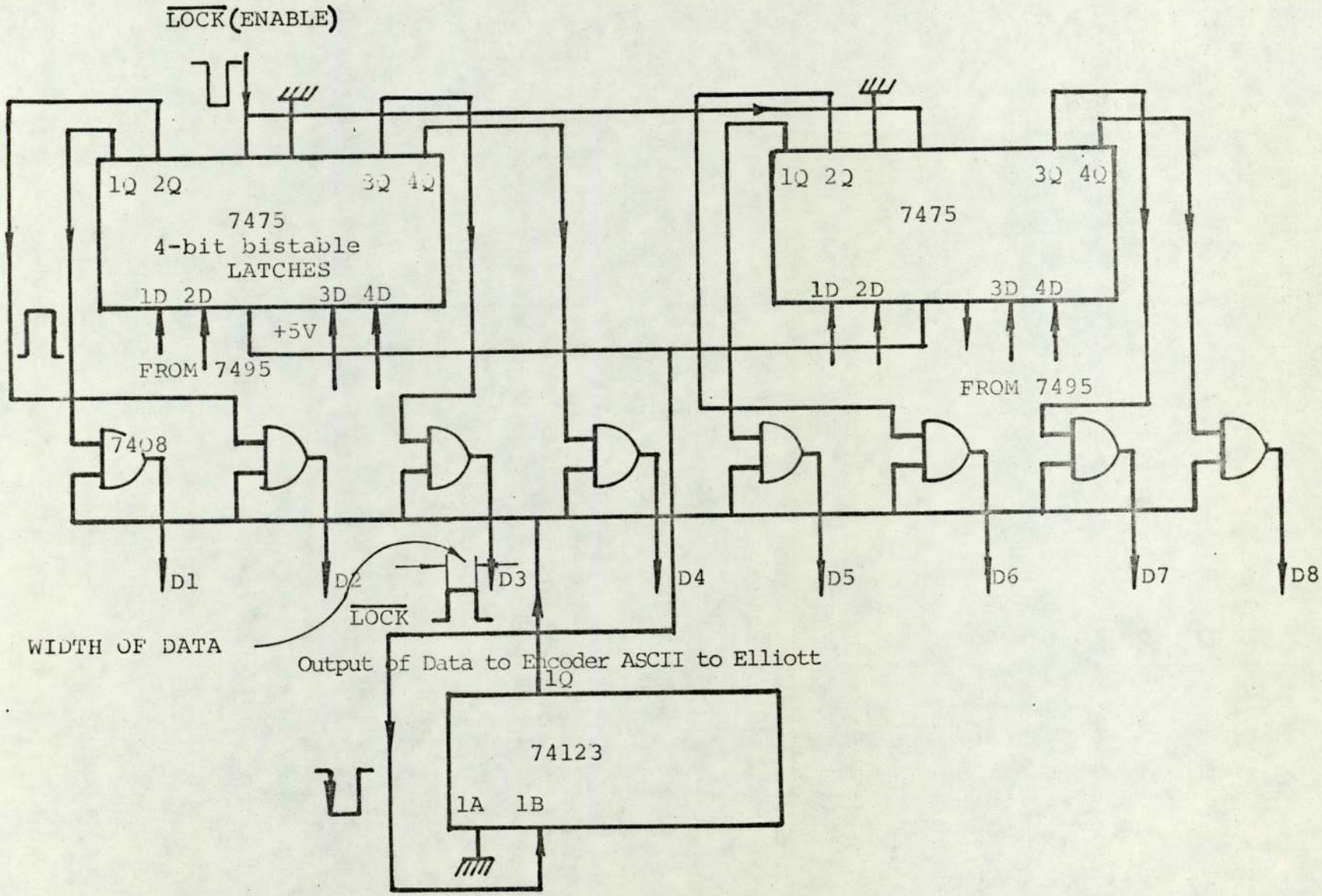
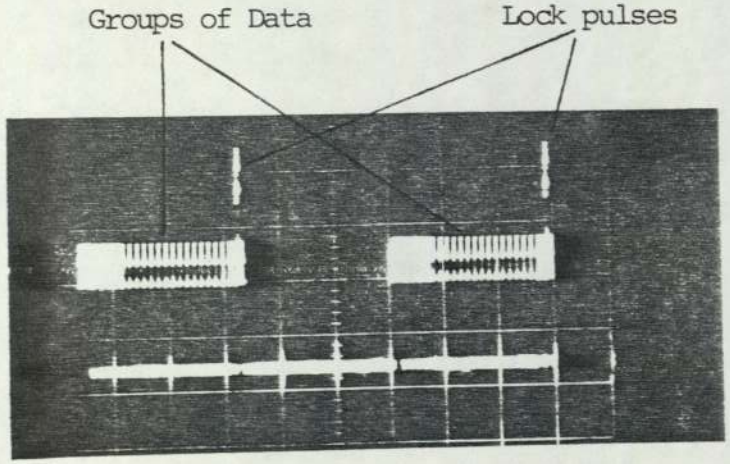


Fig. 67. Timing circuit between data and enable pulses.



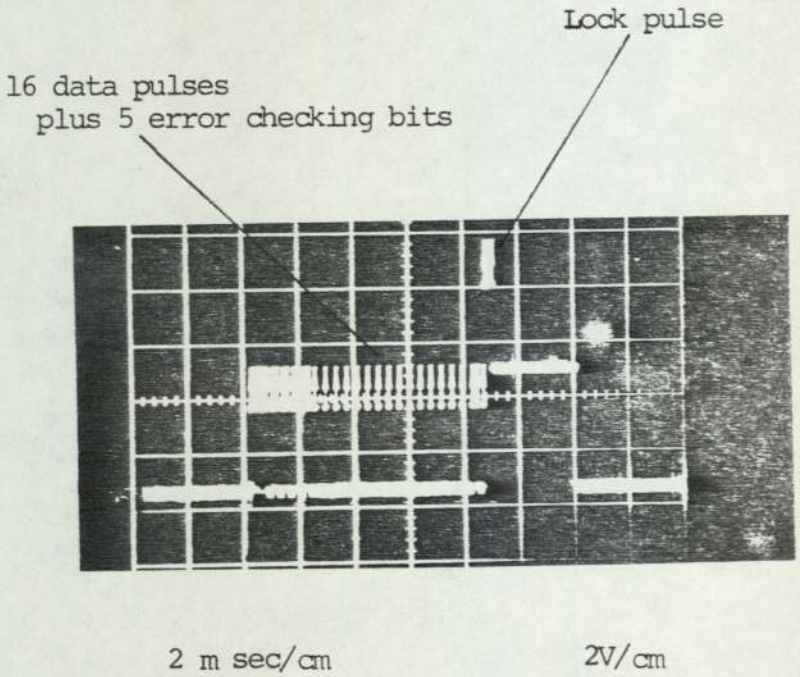


Fig. 68. Photographs of Data and Lock pulse.

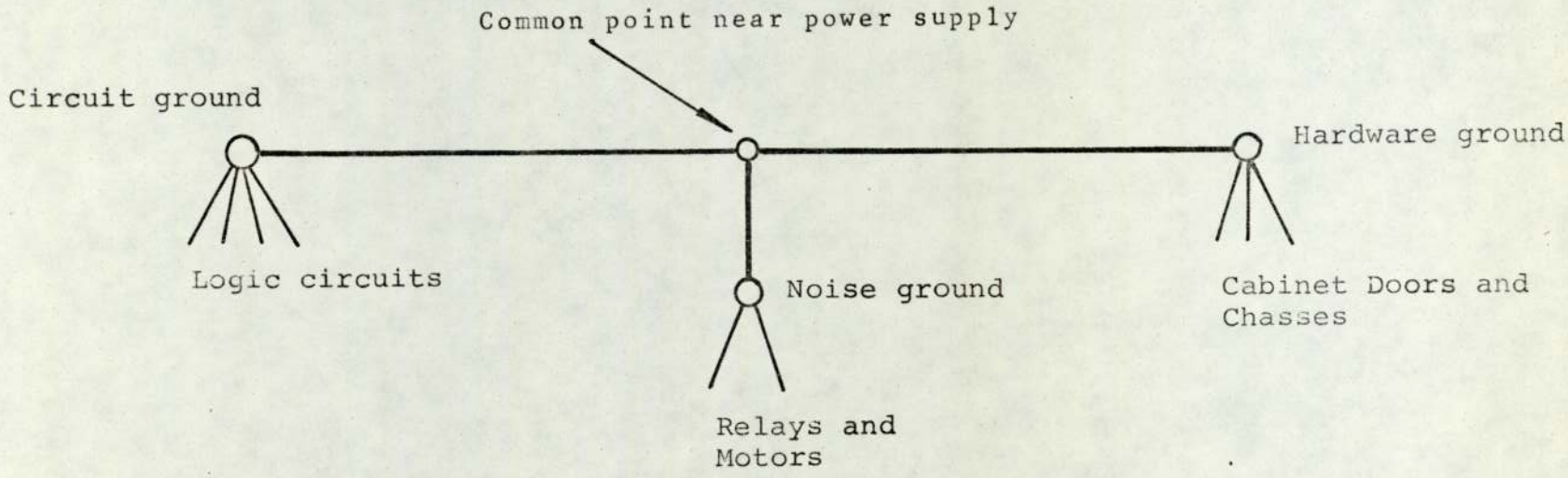


Fig. 69. System Grounding

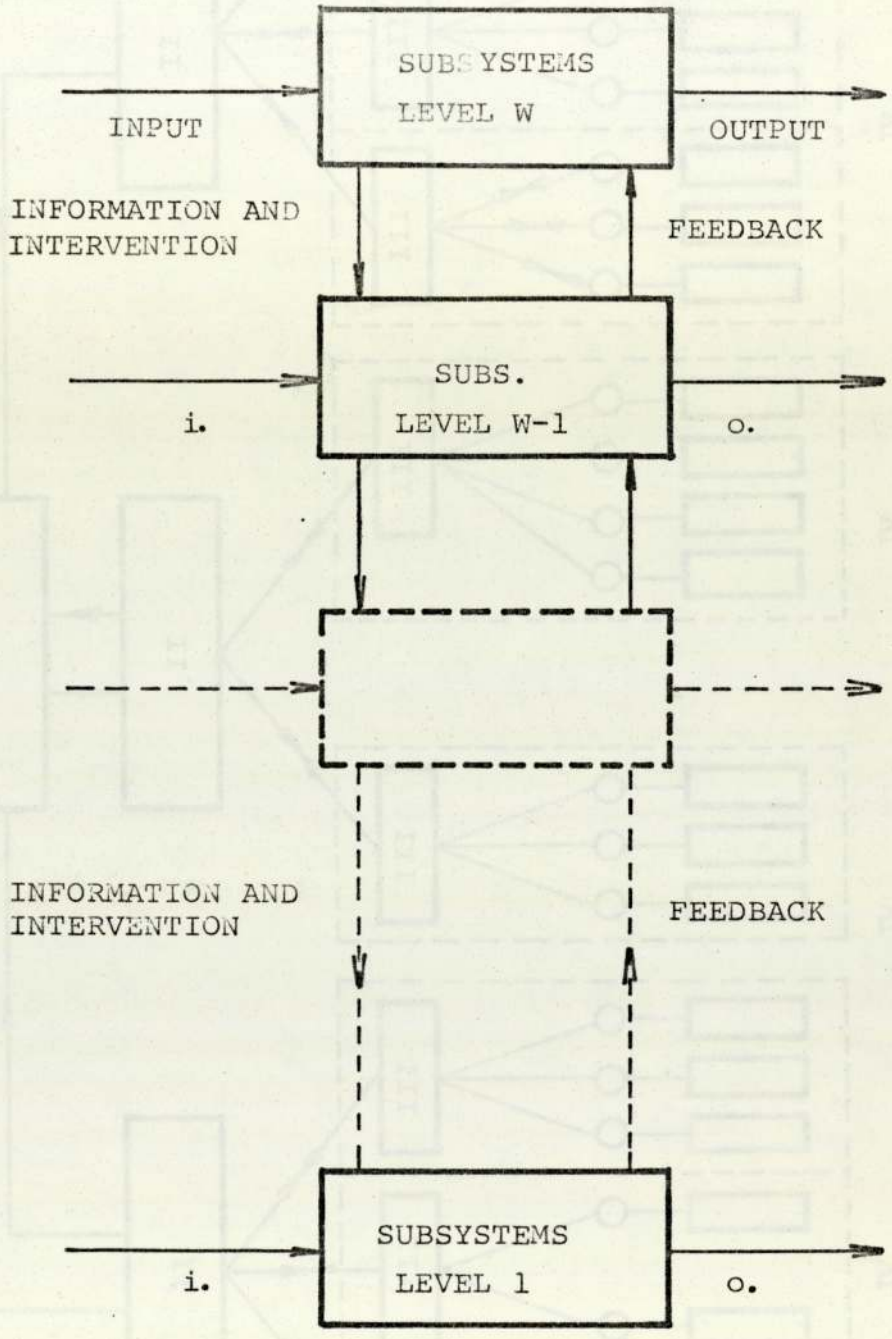
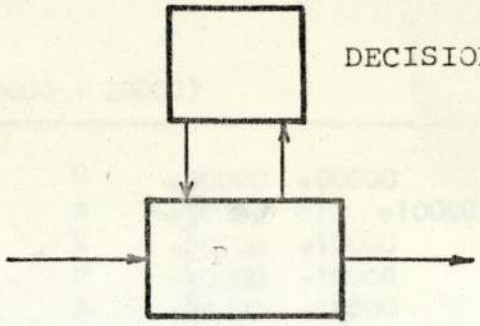
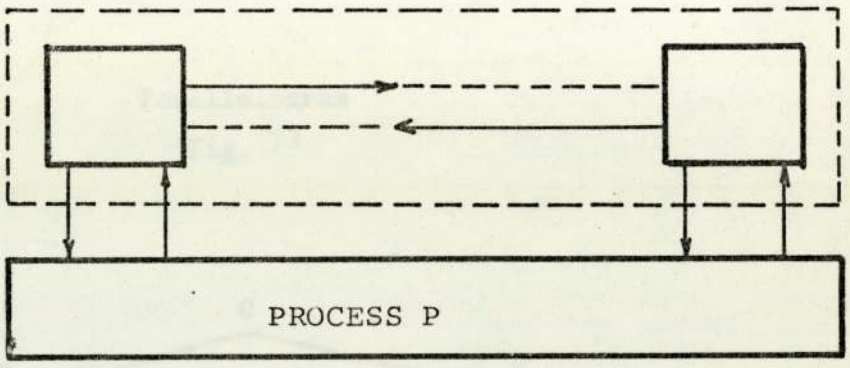


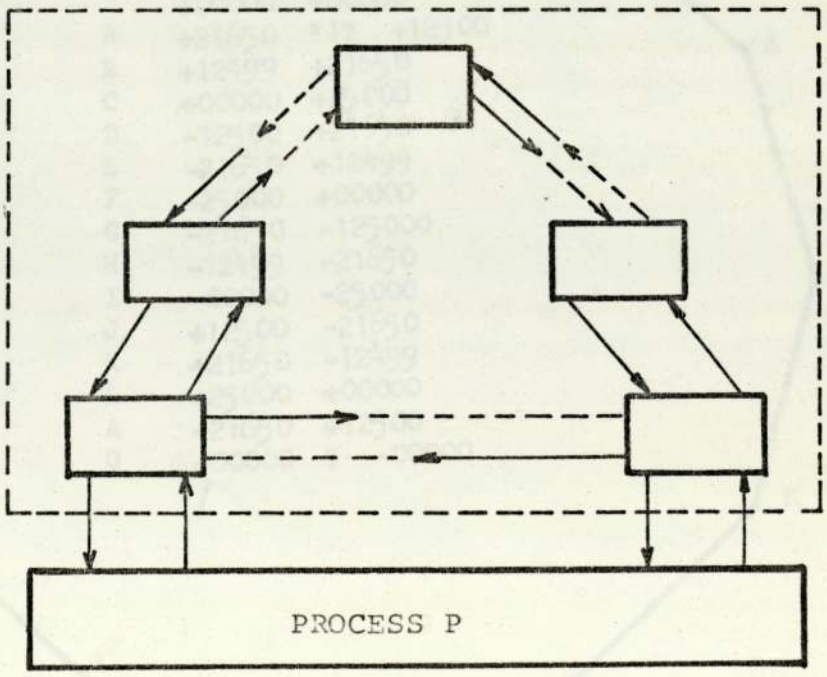
Fig. 70. Vertical connection into the hierarchy system.



SINGLE LEVEL SINGLE-GOAL SYSTEM

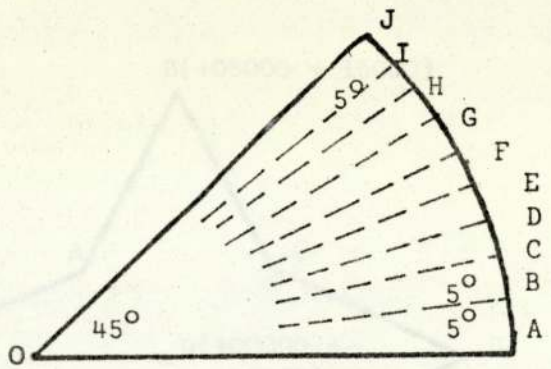


SINGLE-LEVEL MULTIGOAL SYSTEM



MULTI-LEVEL MULTIGOAL SYSTEM

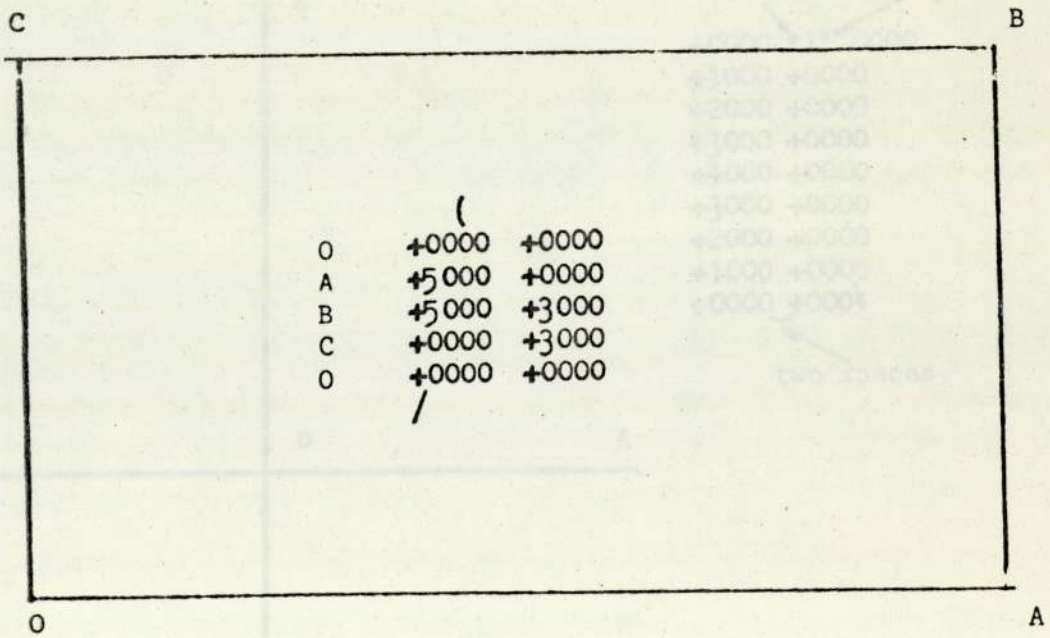
Fig. 72. Classification of decisions.



O	+00000	*1?	+00000
A	+25000		+00000
B	+24904		+02178
C	+24620		+04341
D	+24148		+06470
E	+23492		+08550
F	+22657		+10565
G	+21650		+12500
H	+20478		+14339
I	+19151		+16069
J	+17677		+17677
O	+00000		+00000

Sector

Fig. 75



O	+0000	+0000
A	+5000	+0000
B	+5000	+3000
C	+0000	+3000
O	+0000	+0000

Rectangle

Fig.76

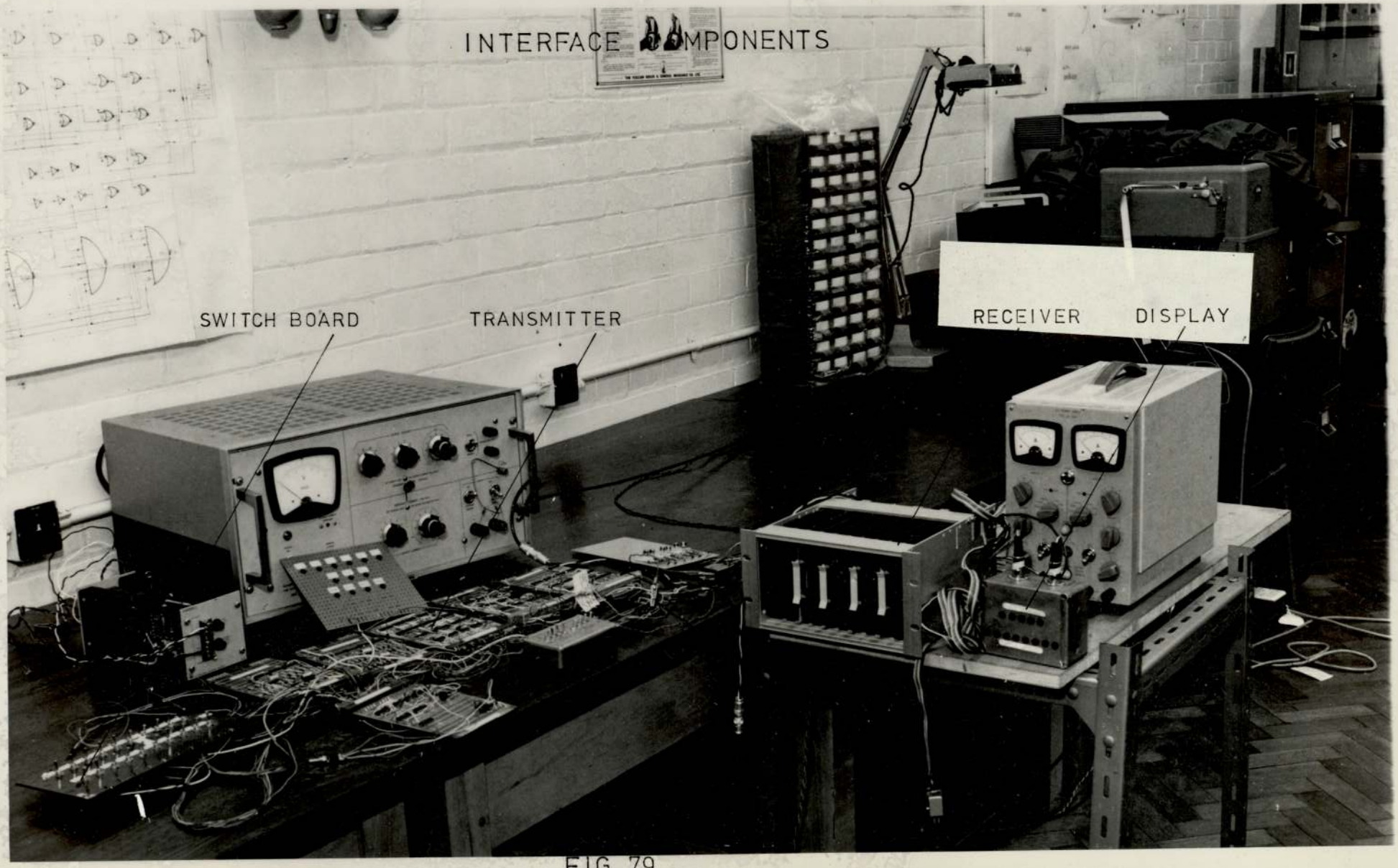

INTERFACE COMPONENTS
SWITCH BOARD
TRANSMITTER
RECEIVER
DISPLAY

FIG 79