Eastern Economy Edition

Second Edition

COMPUTER-BASED INDUSTRIAL CONTROL

KRISHNA KANT

PHI

Computer-Based Industrial Control

Second Edition

Krishna Kant
Senior Director
Department of Information Technology
Ministry of Communications and Information Technology
Government of India

PHI Learning Private Limited
New Delhi-110001
2010

COMPUTER-BASED INDUSTRIAL CONTROL, Second Edition
Krishna Kant

ISBN-978-81-203-3988-0

The export rights of this book are vested solely with the publisher.

Eleventh Printing (Second Edition) ... ... May 2010

Published by Asoke K. Ghosh, PHI Learning Private Limited, M-97, Connaught Circus, New Delhi-110001 and Printed by Rajkamal Electric Press, Plot No. 2, Phase IV, HSIDC, Kundli-131028, Sonepat, Haryana.

To

the memory of my parents

R.R. Verma and Gulab Devi

and

my nephew

Vivek

Contents

Foreword

The history of control can be traced back to the history of mankind. Since the beginning, man had been devising artifacts to sense various parameters and to control them. While the recorded history of technology development can be considered in terms of thousands of years, the serious study of technology and its development has occurred only during the last 100 years. The two World Wars have been responsible for the rapid development of automation technology. Undoubtedly, the advancements in micro-electronics technology and the emergence of microprocessors have triggered a boom in these developments.

The process control area can be divided into three streams, namely the Techniques, the Components and Sub-systems, and the System Architecture. These streams though developing independently are converging each other. The process control techniques involve the classical controls and its developments. The last few decades have seen the consolidation of classical controls as well as a shift towards Model Based Adaptive as well as Self-Tuning Control. These techniques are being used increasingly in industries. The latest developments are in the field of intelligent control using expert systems, fuzzy logic controller and neurocontrollers.

The field of *components* has seen a number of developments in the form of specialised microprocessors (microcontrollers) which can be used for process control directly. Sensors and actuators, the two major components in process control, have developed enormously in the form of Silicon Sensors, Fibre-Optic Sensors, Biosensors, Electronically Actuated Valves, Digital Valves, etc. The field of subsystems has seen various developments in the form of SCADA systems, Remote Terminal Units for Telemetry and Telecontrol, Programmable Controllers, Distributed Digital Controllers, Personal Computers, etc. The software systems supporting the process control have developed into Real-time Programming Languages.

The *architecture* developed for process control over the years includes Distributed Digital Control, Distributed SCADA Systems, Multi Microprocessor Architecture using Local Area Network Concepts, Telemetry and Telecontrol Systems using Remote Terminal Units, etc.

These streams have been successfully implemented in a number of applications in industries. The present book covers all the streams and the developments in detail, including the case studies of automation in some of the major industries. Exposure to these developments is more often than not provided to undergraduate as well as postgraduate students in colleges and universities. The book is, therefore, useful for students offering courses on instrumentation, process control, automation, etc. at both the levels.

I know Dr. Krishna Kant since 1979, when he joined the Department of Electronics as a Senior Systems Engineer in the Appropriate Automation Promotion Laboratory. During these years, he has been deeply involved in the development of a number of automation projects and subsystems. The book is therefore a result of his experience and learning in this field. The presentation of the subject in the book is simple and gradual. This book is a must for all practicing instrumentation engineers in the field.

Dr. N. Seshagiri
Special Secretary, Planning Commission &
Director General (NIC)

Preface

Technology is changing very fast and it is essential that academics should keep in touch with it. Over the last decade there have been considerable developments, and new areas like nanosensors, fault-tolerant systems and many more have emerged. The first edition of the book has been very popular and this has given me immense satisfaction. I have received continuous feedback from different institutions, research organizations as well as industries to revise the content. On the basis of the feedback received, the content has been thoroughly revised and made up to date. In addition the following new topics have been added to the book:

- Controller Operation
- Control System Response
- Controllability of Process
- Control Loop Tuning
- Control Loop Robustness
- Liquid Flow Transducers
- MEMS Sensors
- Biosensors
- Nanosensors
- Reliable System Development Strategy
- MEMS Valves
- Fieldbus System
- LonWorks: Control Network Technology on a Chip
- Industrial Personal Computer Development

The sequence of chapters has also been changed and case studies have been added in some chapters.

I am thankful to all those who have sent their feedback either verbally or through letters or e-mails to me. I am particularly thankful to Dr. V.I. George, HOD, and other faculty members of Instrumentation and Control Engineering Department, Manipal Institute of Technology, Manipal, and Dr. Smriti Srivastava, Assistant Professor, Instrumentation and Control Engineering Department, NSIT, New Delhi, for their valuable suggestions.

I hope that the book fulfils the expectations of academics as well as professionals engaged in research and industry.

Krishna Kant

Preface to First Edition

Today, the study of process control systems or automation in various engineering institutes and universities is confined to theoretical aspects, which can hardly be applied to real-world problems. This book, which is an outgrowth of my teaching experience in the University of Delhi as well as my practical experience, is intended to give a practical thrust to the subject. This is because, I found, during the course of delivering my lectures, that there was a dearth of books available which could be useful to the students or the practicing engineers when faced with practical problems. Hence the need for a book such as this.

The main objective of this profusely illustrated book is to impart solid application-oriented knowledge of the technology spanning the Industrial Control field. This technology includes control instrumentation (i.e. sensors, controllers and actuators), computer hardware and software, data communication links and the relevant advanced control techniques. It also incorporates some specialised subsystems like Programmable Controllers, SCADA systems, and Remote Terminal Units, Distributed Digital Control is an important field which has come up in a big way in all industrial control applications. Another area, though recently introduced and which has made great impact, is the application of Personal Computers as a low-cost tool. These have become increasingly powerful, thanks to the advent of more and more powerful microprocessors. In this book, special emphasis is laid on case studies pertaining to the application of various fields like Distributed Digital Control, Programmable Controllers, SCADA, and Remote Terminal Units. The knowledge gained through these case studies could be utilised in solving real-world problems.

The book *introduces* the subject of automation—its historical development, basic functions and current trends. Chapter 1 is devoted to the fundamentals of automatic process control and its various types. Chapter 2 discusses the transducers which sense the process and present the electrical signals to the computer. Covering both conventional and futuristic transducers, it also includes a special section on Biosensors. Chapter 3 gives the user a detailed account of the building blocks of the automation systems and various modules used in these automation systems. Direct digital control structure as well as control algorithms have been discussed in

Chapter 4. Programmable controllers and their programming aspects like ladder diagram, Boolean mnemonics and functional blocks have been covered in Chapter 5.

Distributed digital control is covered in Chapter 6. The four-level hierarchy of a typical distributed digital control has been discussed in detail, including the examples of some of the well-known distributed control systems. Chapter 7 presents a cursory glance of display systems, computer graphics and graphical user interfaces used in distributed control systems. Chapter 8 discusses various types of actuators and control valves as well as the interfacing of the control elements to computers and microprocessors. The intricacies of software which control various tasks and the real-time operating systems have been presented in Chapter 9. The personal computer and its applications in real-time environment along with its interfacing to the outside world are the topics of Chapter 10. This chapter also presents the present system design strategies followed by various manufacturers of the modules which can be readily used with personal computers. Advanced control techniques using modeling and simulation have been briefly presented in Chapter 11.

Chapter 12 discusses the application of industrial control systems in various industries in the form of case studies. It gives an in-depth analysis of the process intricacies and presents various control strategies for these industries, viz. steel, cement and power, water treatment plant and irrigation canal.

Chapter 13 briefly describes intelligent controllers covering AI systems, expert controllers, fuzzy controllers and neuro-controllers.

While writing this book, special care has been taken to make it *modular,* and thus each chapter can be viewed as a specific module. It has been assumed that the reader has the fundamental knowledge of process control and the basic background in electronics, control instrumentation and microprocessors.

This book would be found useful by undergraduate and postgraduate students undergoing courses on control system design, instrumentation engineering and advanced control systems. These courses are offered in almost all the universities in India. Teachers will find this book very useful while planning their courses. As this book is more oriented towards applications rather than theory, the practising engineers should find this book highly useful as well.

Krishna Kant

CHAPTER

0

Introduction

In our industrial society today, the information is the most strategic resource. It is, therefore, inevitable that the information technology sector will gain ground over manufacturing and goods producing sector. Creation, processing and distribution of information will therefore be the predominant economic activity of the future. The information technology sector will in general include tools for accomplishing the above task. The identifiable parts are computers, communication and industrial controls.

The nature of both data handling and processing task is changing. Computing is moving from a sequential, centralised world to parallel, dccentralised world in which a large number of systems must work together, thus calling for a new generation of general purpose computers. Technologically and socially, the needs for future generation of computers are becoming increasingly demanding for example computer architecture (including distributed architectures supporting computer networks, viz., Local Area Networks (LAN), Wide Area Networks (WAN), and Parallel Architectures) which provide high speed computers for numerical calculations and VLSI architectures make full use of the potential of VLSI technology.

The VLSI technology is making very rapid strides which has enabled the industrial applications of computers to become more and more viable. The processing architectures, the memory technology and the man-machine systems which are important for processing and distribution of information in industries are undergoing metamorphic changes. The computing power is facing an exponential growth, while the cost per bit of information processing is rapidly falling. It is becoming clearly evident that the digital computer must form the basis for the industrial control system of the future.

As a result, one of the most important recent trends in the development of automatic controls in all industries has been to cover the whole plant under a unified coordination and control, and to begin automating the entire operational supervision system using a hierarchy of computer. Presently, systems are being designed using expert systems, neural networks, transputers and high performance microprocessor for dedicated applications. The multimedia based man-machine interface is the topic of research being pursued.

The rapid progress of nanotechnology is bringing a new era of development. Research and development on nano processors and nano sensors is being pursued with vigour.

0.1 EXPECTATIONS FROM AUTOMATION

To control a complex process we must first understand the factors which influence the performance of the process. In industries, the quality of the input raw materials varies and so is the skill of the operator. Hence, the importance of process control becomes significant in order to produce materials of consistent quality at a competitive price.

The dependence on the skill of an individual operator was found to be inadequate, and hence, the use of process instruments came in. This was further extended when the concept of automation was introduced. An automatic control system could perform a repetitive job; but could not take any decision in the event of variable circumstances. Gradually, the important industries started using microprocessors and computers for real-time application in the process control. Here the control system monitors the status of the process continuously and takes corrective action dynamically to stabilise the process.

Complex process industries like steel, cement, power, etc. require both extensive closed loop controls, exhibiting a high degree of accuracy as well as sequence controls with complex logic functions.

An analysis of these requirements leads to the following demands on the control system concept:

— High reliability and availability
— Fast trouble shooting
— Simple operation
— Easily configurable
— High accuracy and reproducibility of process parameters and process variables
— Low cabling cost
— Availability of process computers for optimization functions and processing of operating data
— Flexibility as regards modifications and extensions required by the process.

0.2 BASIC FUNCTIONS

Automatic control of any modern industrial plant, whether achieved by a computer-based system or by conventional means, involves an extensive system for the automatic monitoring of a large number of the different variables, operating under a very wide range of process dynamics. It requires the development of a large number of functions, some of which might be quite complex, for the translation of the plant variable values into the required control correction commands. Finally, these control corrections must be transmitted to another very large set of widely scattered actuation mechanisms of various types. Because of the nature of the manufacturing processes involved, these may, and often do, require the expenditure of very large amounts of material and energy. Also, plant personnel, both operating and management, must be aware of the current status of the plant and each of its processes.

In addition, such an industrial plant always faces the problem of adjustment of production schedule according to the customer's needs, as is the new order stream being continually received. This should be achieved while maintaining a high plant productivity and the lowest practical production costs. The problem is handled in most cases through computer-aided, production-control system along with an in-process and finished goods inventory as judged adequate by plant personnel.

It has also been repeatedly shown that one of the major benefits of the use of the digital computer control systems in industrial plants is its role of a "control systems enforcer". In this mode, the lower level computers main task is to continually assure that the control system equipment is actually carrying out the job that it was designed for, i.e. to keep the units of the plant production system operating at some optimal level, and to ensure that the controllers have not been set on manual, but that on the optimal set-points.

Often the task carried out by these control systems have been those which a skilled and attentive operator could readily have done. The difference lies only in the degree of the attentiveness which can be achieved over the long run.

As stated earlier, all these functions must be factored into the design and operation of the control system which will operate the plant, including the requirements for *maximum productivity* and *minimum energy* and raw material usage. As the overall raw material, energy and productivity based requirements become more complex, the need of more and more sophisticated and capable control systems became inevitable. In order to attain needed complexity and sophistication the field must gravitate more and more towards digital computer-based systems.

0.3 HISTORICAL DEVELOPMENT OF CONTROL SYSTEMS

All the human beings are gifted with five senses to understand and explore nature. However, it was soon realised that these senses are not adequate to satisfy their curiosities and demands and thus they started augmenting them. In order to explore, measure and gain access to the unknown it became absolutely necessary to develop diagnostic instrumentation. Most of the early instruments provided measurements of simple parameters, viz., length, area, volume, weight, etc. It was in the fields of astronomy and navigation that the most significant and accurate instruments were invented. As time elapsed, precision instruments pertaining to various other fields also were developed.

Today, instrumentation is present in all spheres of life. It meets man's requirements in all fields ranging from house to agriculture, entertainment to space exploration, simple measurement to complex process industries etc.

With the rapid development in electronic industry, the philosophy of instrumentation and controls has totally changed. Now a days, the control philosophy is based on System Structure. The System Structure depends heavily on the technologies developed in recent years by the computer and telecommunication industries.

Historically the developments in the field of control systems can be grouped in (Fig. 0.1) the following eras:

— Early Development
— Pioneering Period
— Direct Digital Control Period
— Micro-computer Period

0.3.1 Early Development

In the early phase of development, machines were used to substitute for human physical power and the manual supervision and control was a necessity. Later the equipments for flow and quantitative measurement were developed and used.

The pace gained momentum when *pneumatic process controllers* first became readily available in the 1930's. These were self-contained devices and main disadvantage was that the operator had to be mobile in order to supervise these and had difficulty in gaining an overall impression of plant operations when large number of controllers were in use.

Subsequently development of transmitters (1950) gave a standard signal (3-15 psi, 4-20 mA) which could be sent over long distances, and thus, allow controllers to be grouped together for supervision. This concept of centralised supervision was introduced in mid 1960's by which time Electronics Instrumentation was gaining popularity because of the availability of semiconductor devices.

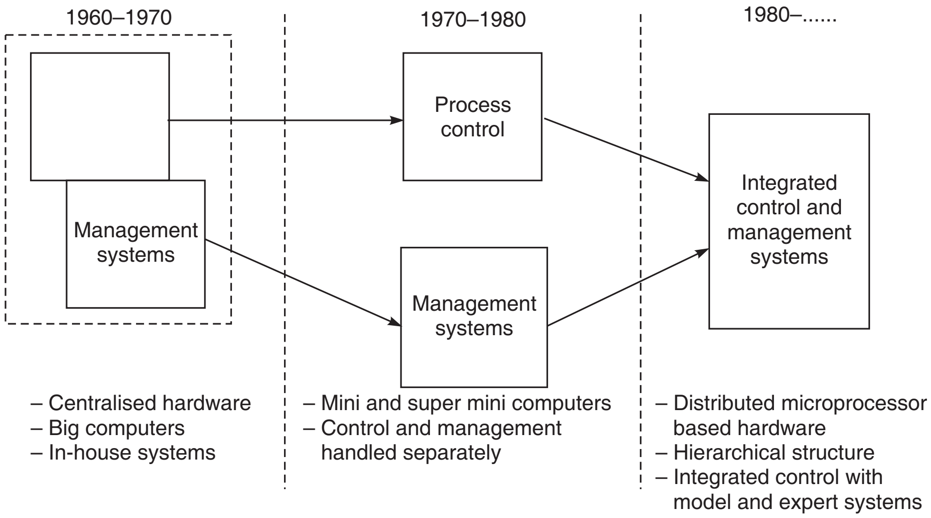


Figure 0.1 Historical development.

During this period, PID (Proportional-Integral-Derivative) regulators became general substitute for automation problems as stand-alone control system. However, start-up and shut-down was through relays. The relays were also used extensively in sequence control system which along with the process control system often worked at cross purpose. However, the advantages and need of automation were established. The digital computers were first used in Monosanto Chemical Company Port, Arthur refineries in 1950. IBM 1700 was then used in oil companies.

0.3.2 Pioneering Period

The rapid development of electronic integrated circuits in the late 1960's was significant in two ways: *firstly* it prompted the design of the small and reliable analog electronic panel mounted instruments which have predominated in the last decade and *secondly* it led to the miniaturisation of digital electronics and the development of mini- and micro-computers.

The computers of this period were slow, expensive and unreliable and were therefore used in supervisory mode only. Two different approaches emerged for process control:

— Operator guide
— Set-point control

Major tasks performed by computer included production planning/scheduling, report generation on production, raw material and energy consumption etc. Control programs were hampered by lack of process knowledge and lack of good modelling methodology,

In early 1960's the major applications involved—control of steel mills, chemical plants, and electric power generation.

0.3.3 Direct Digital Control Period

With the introduction of computers, engineers felt the need of utilising computers advantages in instrumentation field through centralised control system. The advantages were optimisation, alarm and logging function, historic analysis, trending, sequence control and self diagnostic techniques. This gave birth to *Direct Digital Control* philosophy in instrumentation with a back up of *Analog Control System*.

In 1962, ICI in England brought in a computer for Direct Digital Control which meant a computer controlling the process directly. The panel was simple; a digital display and a few buttons.

Special DDC languages were evolved for programming. User simply introduced inputs, outputs, regulator types, scale factors etc., while configuring system for their specific applications.

A number of specialised programs were developed in 1960's for DDC. Mini-computers made process control easier. Efficient process control systems based on DDC were designed. The number of process computers grew from about 5000 in 1970 to about 50,000 in 1975.

0.3.4 Microcomputer Period

Early 1970's gave birth to a new element in the family of Semiconductor Technology, namely MICROPROCESSOR. With the introduction of Microprocessor, Instrumentation Industries around the world, entered a new era due to the availability of the sophisticated control functions, high reliability, flexibility, speed, improved operational technique and the ease of the use in the applications covering various process requirements. The compatibility of Microprocessor to *Digital Data Communication* made it easy for system design engineers to connect widely separated system components over a *Data Communication Bus.* It is this ability to connect widely separated system components, which has led to the use of the term *Distributed System*.

0.4 CURRENT TRENDS IN COMPUTER CONTROL OF PROCESS PLANTS

The first total distributed control system was announced by Honeywell, USA(TDC2000). Since then, microcomputers have made their way into practically countless applications. Present day systems combine PLC functions, Data logging and Digital Control functions. New technologies, such as Artificial intelligence have emerged and new systems such as Expert Systems are increasingly becoming popular.

In recent years digital systems have become the mainstream of instrumentation and control technology. The digital systems have not only created a variety of advanced control systems, which considerably improved process controllability and product quality, but also is revolutionising the philosophy of control system as a whole. Developments in the field of microelectronics based on VLSI technology gave a new dimension to the electronics revolution, in the form of microcomputer based systems.

It is worthwhile, therefore, to briefly mention the current technology trend in the field of computer based instrumentation and control. The current trend in sensors and transducers is towards integrating *intelligence* into these along with the primary functions. The silicon integrated circuit technology has spread into this field as well. Semiconductor sensors are already popular for all applications, and the trend is to provide on-chip signal conversion and certain amount of intelligence for limit checking, etc. These are also reported to be directly compatible to be interfaced with control computers and other actuating elements. Fibre-optic technology has made possible the development of sensors which .provide immunity against electromagnetic noise in the measurements. The Japanese iron and steel industry has already put them into practice and realised such benefits.

Programmable logic controllers are now designed around microprocessors. They facilitate the user to write programs in much easier and understandable language. New languages for programmable logic control are developed. The computer then generates the logic diagram on a CRT. PLCs are becoming available with powerful graphic facility to allow the user to change the relay logic diagram interactively. However, the conventional relay logic programming facility is also being provided for the user.

The recent developments in nanotechnology have provided a new dimension to control technology and systems. Tiny and robust nanosensors are emerging as a result of continuous research and development efforts. The nano processors are soon going to replace microprocessors.

0.4.1 Centralised Computer Control System

The centralised computer concepts suffered many setbacks. In early years, of development, computers were slow, unreliable, memory sizes were limited and programming had to be done in machine language. Further, to help justify higher cost, vendors incorporated all types of computer control functions including supervisory and DDC in one main frame at a central room located in the plant. As time elapsed, though computers became faster with bigger memory size and added features, the centralisation led to further problems such as the need for vast plant communication system. This was required to bring *process* signals to centralised computer system and return the *control* signals to the field. The probability of the failure of that one computer resulted in demand for a complete analogue back up system, paralleling the DDC. The complexity in programming the computers tended to increase and thus worsen the difficulties. A typical centralised computer control system is shown in Fig. 0.2.

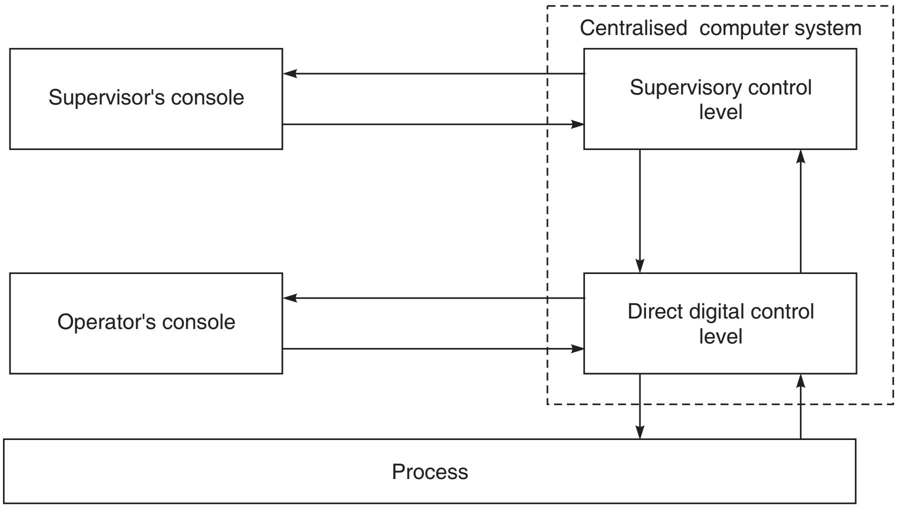


Figure 0.2 Centralised computer system.

0.4.2 Distributed Control Systems

With the rapid advances in VLSI technology and the advent of single chip microcomputers with powerful processing capabilities which can be distributed physically, the distributed computer controls emerged. A distributed system is one which has several microcomputers which can be physically spread all over and each assigned with a specific task and all mutually linked through a data highway which can be coaxial or fibre-optic. Each of these microcomputers perform its own task concurrently and independently of the microcomputers in the system. Thus, this type of parallel processing provides excellent system response time and eliminates the possibility of any single-point failure crashing the whole system. The Honeywell Company, USA in 1969, took a courageous step to design an alternative to the centralised computer control system.

The ‘TDC 2000’ system as it was called, solved the problems of reliability by distributing the control functions to cover only few loops as well as providing a digital back up capability.

This high reliability has greatly contributed to the success of distributed control. Several key factors such as increased reliability, elimination of disks to execute control algorithms, redundancy and extensive error-checking for communication highways, distributed display functions, redundant capability, back up control processors with automatic change-over are combined to achieve such a success. The MTBF for single-loop controllers seem to be atleast more than 20 years, though this figure may drop for multi-loop controllers. The advantage of distributed system however is the ability to upgrade (as the technology improves) without obsoleting the entire system.

Computer languages such as, Real-Time Fortran, Pascal, ADA and C have become most popular with distributed computer control systems. This is coupled with rapid strides in the developments in fibre-optics technology for wide-band commuuications between different computers in the system. Fibre-optic communication has enabled high data transmission rates of the order of gigabits/sec with a distance of about 2 km between two stations for reliable communication.

Man-machine interfaces such as powerful interactive colour graphics systems enable the user to generate/configure the various control loops, changing limits of any channel in any particular control loop 'on-line', and observe the performance of various control loops. The microprocessor based video terminals have made the task of human operators quite easy. Figure 0.3 shows a distributed digital control system.

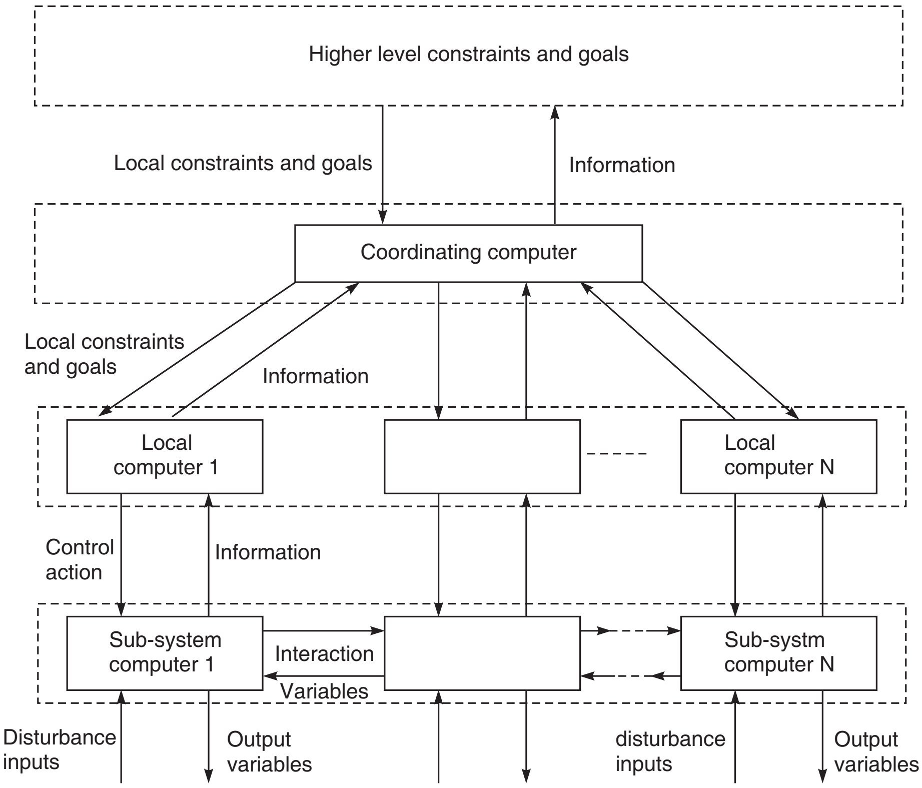


Figure 0.3 Distributed digital control system.

0.4.3 Hierarchical Control Systems

The development of the distributed digital control systems greatly simplified the computer's connection to the process. Combination of the three levels of control (each with distinct duties), namely, the *dedicated digital controllers* for process loops, *direct digital control* for certain process variables and *supervisory control* levels, constitutes a hierarchical system. The upper computers depend upon the lower level devices for process data and the lower level systems in turn depend upon the higher level systems for even more sophisticated control functions such as an overall plant optimisation. Therefore, by combining company's production scheduling and management information functions with the process control functions, one can develop a total plant hierarchical computer control system as shown in Fig. 0.4.

It must however be remembered that all the elements in a hierarchical system can exist as individual elements. It should be noted that the different levels in a hierarchical system do not necessarily represent separate and distinct computer or hardware levels. One or more of these operational levels can be combined into one computer depending on the size of the system.

0.4.4 Process Models

Modeling and simulation of systems has gained considerable importance during last three decades. This is due to rapid development of system and control theory during this time and especially due to rapid development of the computer technology. Generally speaking, no plant automation system can now be designed and put into work without using the methods of model building and simulation of the system to be automated.

0.4.5 Advanced Control

What exactly is advanced control? Depending on an individual's background, advanced control may mean different things. It could be the implementation of feed forward or cascade control schemes; of time-delay compensators; of self-tuning or adaptive algorithms; or of optimization strategies. To some it may mean Intelligence Based on Expert System, Fuzzy Logic, or Artificial Neural Network. Here, the views of academics and practicing engineers can differ significantly.

It has been recently reported that advanced control can improve product yield; reduce energy consumption; increase capacity; improve product quality and consistency; reduce wastage; increase responsiveness; improve process safety; and reduce environmental emissions. By implementing advanced control, savings ranging from 2% to 6% of operating costs have been quoted. These benefits are achieved by reducing process variability, thereby allowing plants to be operated to their designed capacity.

In the recent past, many manufacturers of international stature have successfully made use of robotics to improve their production capability. A close corollary to robotics is Artificial Intelligence (AI). Although the problems of emulating human intelligence are staggering, one branch of AI-expert systems has reached the level of practical economic application.

Expert systems are computer systems which use knowledge and inference procedures developed from human expertise. The *knowledge* usually takes the form of computer rules coupled with observed facts. *Inference* is the logical process which combines rules and facts to produce new facts.

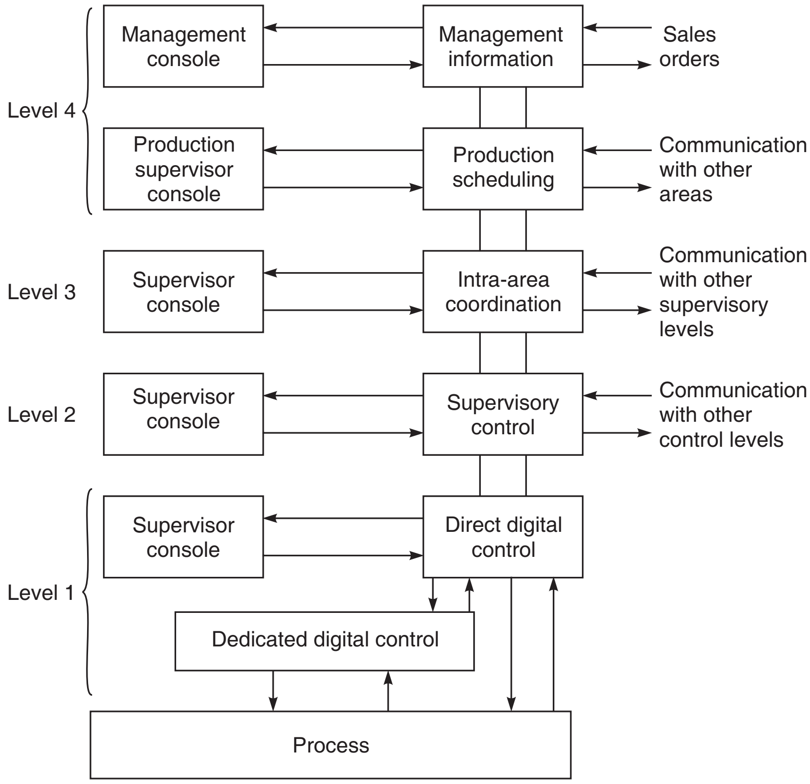


Figure 0.4 Total plant hierarchical control system.

The two most widely used logic patterns in expert systems are the "Forward Chaining" and "Backward Chaining" patterns. Forward chaining systems accept input data and move through a rule base, thus deducing new facts. For instance, an input fact "a" will be incorporated into the rule "If a Then b" and the new fact "b" will be deduced and added to a list of other known facts. This logic pattern is useful in those applications where implications of each piece of new data must be evaluated.

Forward chaining systems are commonly called "production systems" because in each cycle they produce an additional fact. Rules in production systems are called "productions". AI researchers seem to favour this kind of system because it is a reasonable model of human thinking.

The "Backward Chaining" logic pattern, is "goal-oriented". The logic of this kind of system is accumulation of facts which will satisfy certain specific references of "goals". This logic is easier to computerise because the goals "inferred facts or conclusions" are already known and the system is concerned only with facts which will support a logical inference. This technique starts with a goal to be proven and works backwards to resolve it. Backward chaining expert systems direct user sessions by asking very specific questions. Facts unrelated to a current specific goal are not normally accepted.

If a narrow focus and lack of generality is accepted, the goal-directed approach can provide systems which operate efficiently, and are easy to write, easy to understand and provide excellent results in their area of expertise. The key components and interfaces of a typical backward chaining expert system are shown in Fig. 0.5.

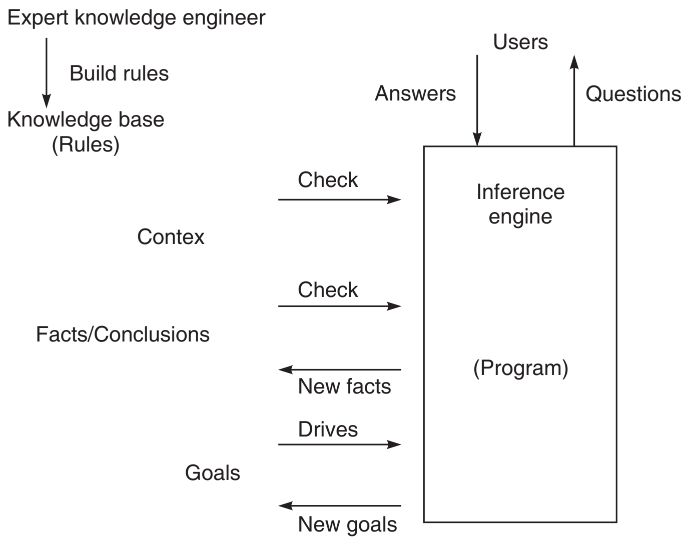


Figure 0.5 Typical backward chaining expert system.

Living systems exhibit control which many times surpass the best control system algorithms and strategies. It is therefore natural to look into computational and control paradigms used by nature. Artificial neural network is such a biologically inspired paradigm. The artificial neural network has the following unique characteristics which makes it very attractive for different applications:

(a) They can be trained;
(b) They can approximate non-linear functioning;
(c) They can be easily implemented on parallel hardware for fast computation.

The above characteristics exhibit the intelligent behaviour of the artificial neural networks and make them good candidates for the control of non-linear processes for which no perfect mathematical model is available.

0.5 CONCLUSIONS

The field of automation and control is pretty vast and deep. We have introduced the subject in this chapter. In subsequent chapters, we shall be discussing some of the important topics in detail.

Process control and other industrial applications of computers have always been a relatively small fraction (approximately about 10 per cent) of the computer field, and thus it is a beneficiary but not a driver of this technology. Process control indeed had an input into the requirement of computer systems (necessary speeds, word lengths, memory size, reliability, allowable cost, etc.) but as a second order effect; process control has influenced the characteristics of the systems (which would have been built anyway) but probably did not generate their production in the first place. However, since it is believed that the needs of the process control field are not that much different from other applications, the general acceptable standards in the design of the computer systems should not be detrimental to the needs of the industrial computer systems.

SUGGESTED READING

Arzen, K.E., Experiments with expert systems for process control, *Proc. 1st Internat Conf. on Appl. Art. Intel. in Engg. Practice,* Southampton, UK, 1986.

Astrom, K.J., Process Control—Past, present and future, *IEEE Contr. Syst. Mag*., **6**, No. 4, pp. 3–9, 1985.

Astrom, K.J. and Wittenwork, B., *Computer Controlled Systems,* Prentice Hall, Englewood Cliffs, New Jersey, 1984.

Ball, K.E., World Markets: How much does automation control, *InTech,* **34,** No. 1, pp. 19–23, 1987.

Bennet, S. and Linkens, D.A. (Eds.), *Computer Control of Industrial Processes,* Peter Peregrinus, Hertfordshire, UK, 1987.

Bennet, S. and Linkens, D.A., *Real-Time Computer Control,* Peter Peregrinus, Hertfordshire, UK, 1984.

Bhatkar, V., Artificial intelligence applications in process control, ISDC '86, *Int. Sem. on Distr. Contr.,* New Delhi, 1986.

Bhatkar, V. and Kant, K. (Eds.), *Microprocessor Applications for Productivity Improvement,* Tata McGraw-Hill, New Delhi, 1988.

D'Abred, F.A., Replacing installed process control computers, *InTech,* **29,** pp. 57–59, 1982.

Gupta, M.M., and D.H. Rao, *Neuro Control System*: *Theory and Applications*, IEEE Press, USA, 1994.

IFAC, Artificial intelligence in-real-time control—IFAC Workshop, Swansea, UK, 21–23 Sept., 1988, Pergamon Press, Oxford, UK, 1988.

Leitch, R. and Francis, J., Towards Intelligent Control Systems *In Expert Systems and Optimization on Process Control,* Proc. of Seminar, Dec. 1985, pp. 62–73, 1985.

McTamaney, L.S., Real-time intelligent control, *IEEE Expert,* 2, Winter 1987, pp. 55–68, 1987.

Myers, W., Expert systems, *IEEE Expert,* **1,** No. 1, pp. 100–109, 1986.

——, Introduction to expert systems, *IEEE Expert,* **1,** No. 1, pp. 100–109, 1986.

Popovic, D., State-of-the-art and future trends in industrial process automation, *Electronics Information and Planning,* **13,** pp. 686–691, 1986.

Rajaram, N.S., Artificial Intelligence: Its impact on the process industries, ISA-INTECH, **33,** No. 4, pp. 33–36, 1986.

Reynolds, D.E., Bolton, C.B. and Martin, S.C., AI Applied to Real Time Control: A case study, *Proc. of Conf. on Appl. of AI to Engineering Problems,* 1986.

Sargent, R.W.H., The Future of Digital Computer-based Industrial Control System-A projection, *Proc. of 10th Ann. Contr. Conf.,* pp. 63–241, 1984.

Sriram, D. and Rychener, M.D., Expert systems for engineering application, *IEEE Software,* pp. 3–5, 1986.

Stephanopoulos, G., The Scope of Artificial Intelligence in Plant-Wide Operations, *Proc. of 1st Internat. Conf. on Foundations of Comp. Aided Process Operations,* Elsevier, Park City, 1987.

Syrbe, M., Saenger, F. and Long, K.F., Centralised and distributed control system, *Process Automation,* No. 1, pp. 18–25, 1983.

Talukdar, S.N. and Cordozo, E., Artificial intelligence technologies for power system operations, Report EL-4323, Electric Power Research Inst., 1986.

Williams, T.J., Computer Control Technology—Past, present and probable future, *Trans. Inst. Meas. Control,* **5**, No. 1, pp. 7–19, 1983.

CHAPTER

1

Fundamentals of Automatic Process Control

1.1 INTRODUCTION

To understand process control techniques, one can use two major approaches. The first is the mathematical approach, where we make use of Laplace transforms, Z-transforms, control equations, stability criteria, system functions, Bode plots, etc. This approach definitely helps us to look into various intricacies of control system theory. A good number of books are devoted to this subject. According to the other approach, automatic process control is considered on a conceptual plane, i.e., as viewed by a system engineer. This implies that the system intricacies are understood and different control actions are seen from the application point of view. The emphasis here is not on advocating a particular type of approach or pointing out the pitfalls of one approach over the other, but to present the different facets of automatic process control on the conceptual plane.

1.2 PROCESS DEFINITION

The fundamental requirement for a process control engineer will be to define the process rightly. The process itself may be defined in a number of ways—either as a set of different sub-processes and activities between them, by just describing the input/output relation, or by going through each and every basic component and describing its composition. However, for the purpose of process control, a process is defined in terms of the various components and characteristics of the process required for control.

Figure 1.1 shows an example of water heating system. The water is input at the bottom and the steam enters at "steam in". The hot water comes out at “water out” and steam at "steam out". In this particular example, the parameter we wish to control is the temperature of water. This is known as *control variable*. This can be controlled by varying the flow of steam. Thus, the steam flow will be known as *manipulated variable* or *controlled variable*. The other variables which we are not considering in this experiment are temperature of water input, ambient temperature in the atmosphere, etc. The variation in these parameters (variables) affects the controller and these are known as *load variables*.

Thus an automatic controller can be defined as "A mechanism that measures the value of control variable and manipulates the manipulated variable or controlled variable to limit the deviation of the control variable from the limit set".

This limit is known as *set-point*. The control of water heating system is shown in Fig. 1.2.

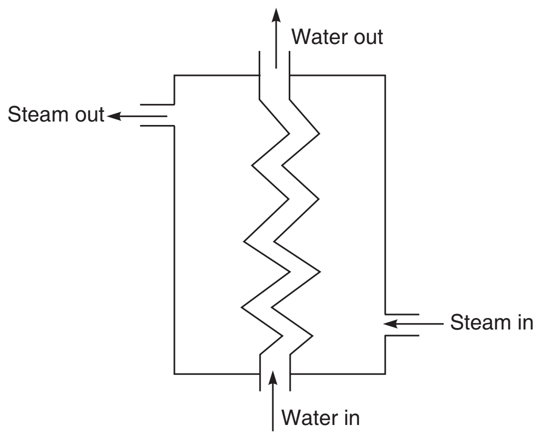


Figure 1.1 Water heating system.

We have not yet defined the basic characteristics of the process which affect control. Processes have characteristics of delaying or retarding changes in the values of process variables and these therefore increase the difficulty of control. This is known as *process time lag*. The three characteristics of process which are responsible for time lags are: Capacitance, Resistance and Dead Time.

The *capacitance* of the process is its ability to store energy or material. The *resistance* of the process is its basic capability to resist the transfer of energy. In the experiment shown in Fig. 1.1, the wall of the tank and also the coil having the water stores the energy, which can be termed the *capacity* of the process. Similarly, the steam layers around the coil will have an insulating effect and will resist the transfer of heat from steam to water. This insulating effect will act as the resistance of the process. Every process takes a finite amount of time to transfer the disturbance to the point of measurement like in our example shown in Fig.1.2. If, however, the temperature of input water drops suddenly, then it will take some time to detect its effect at the output, as water will take a finite amount of time to reach the point of measurement. During this time, no change takes place in the temperature of output water. This delay is known as *transportation lag* or *dead time* and is typical of the process. It is clear that dead time or transportation lag will depend on the placement of the measurement point, velocity of disturbance, total distance, etc. It is very difficult to measure the capacity and the resistance of any process. However, there can be a good estimate of the total process time lags.

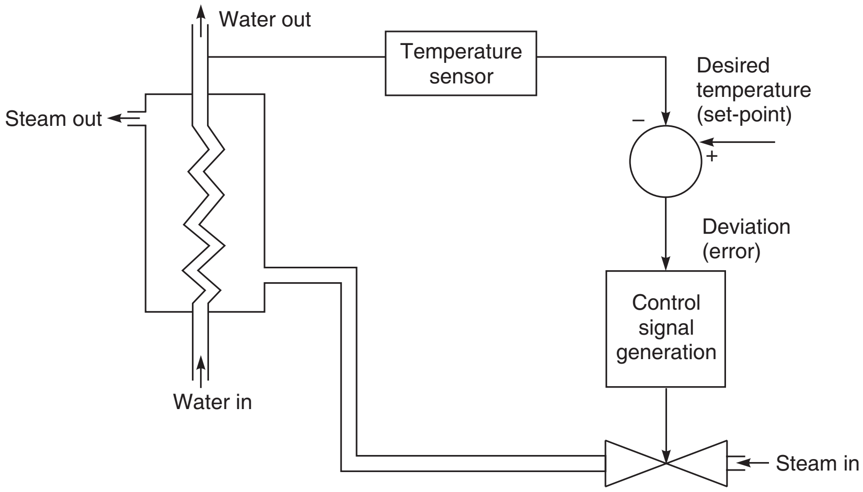


Figure 1.2 Control of water heating system.

Control systems may be categorised as either open loop control systems or closed loop control systems. Closed loop control systems are also popularly known as *feedback control systems*. The difference between the two control strategies is in terms of the role of the operator versus the role of the controller.

1.3 OPEN LOOP CONTROL

Open loop control strategy is basically operator-based and the controller works under his or her command. Figure 1.3 shows the schematic view of an open loop control system.

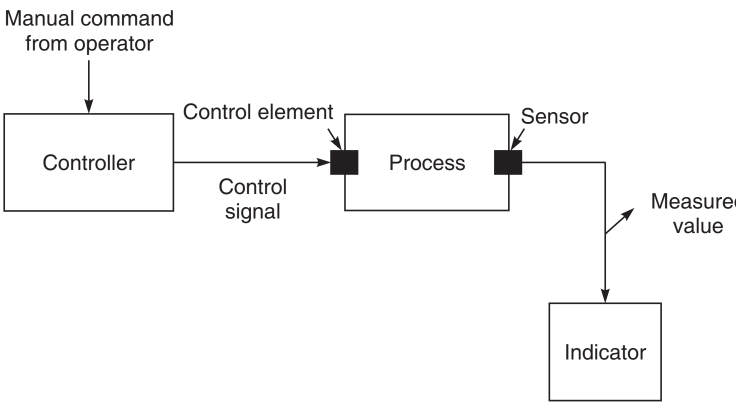


Figure 1.3 Open loop control—schematic view.

The following are the features of the Open Loop Control System:

- The sensor measures the process parameter, i.e., control variable.
- The indicator displays the measured value of the parameter.

- The operator knows the limit (set-point) beyond which the parameter should not move.
- Based on the deviation of the measured value from the set-point (error), the operator will control the process by suitably changing the manipulated variable through the controller.
- The controller under the command of the operator will send a control signal to the process. The control signal will change the manipulated variable using the control element, which may be either a control valve or a simple relay on an electric motor.

Figure 1.4 shows the open loop control of a water heating system. The temperature of water out (control variable) is measured and displayed. In order to maintain the temperature at a fixed value (set-point) the operator controls the flow of steam (controlled or manipulated variable) through the controller. In this case the controller will have knobs to increase or decrease the steam flow. The controller will send the appropriate control signal to the valve to vary the flow.

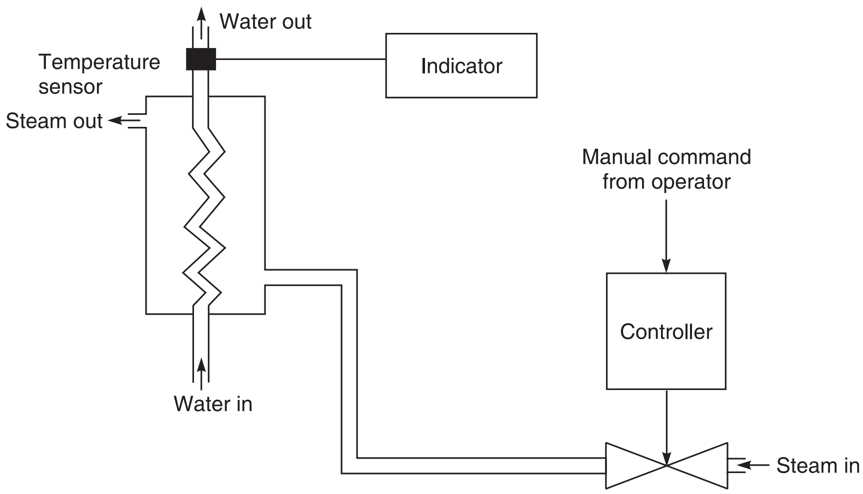


Figure 1.4 Open loop control of water heating system.

It is evident that the operator is an integral part of open loop control and his efficiency is directly reflected in the efficiency of the control system. An inefficient operator may not only control the process in an inefficient way, leading to wastage of energy and raw material, but also may adversely effect the process and even damage the process. In the water heating example, very high temperature may break the pipeline.

1.4 CLOSED LOOP CONTROL

Closed loop control system is that in which the control action is decided by the controller based on the system output.

A closed loop system measures the actual system output, compares it with the expected value and determines the error. This error value is then used to control the system output in order to obtain the desired output (Fig. 1.2).

1.5 BASIC PRINCIPLES OF SINGLE CONTROLLER LOOP

Figure 1.5 shows the basic principle of a simple control loop. The level to be controlled is monitored by a level-sensing system with incorporated amplifier, with the total being called a *transmitter*. The level transmitter (LT) sends a signal (4 to 20 mA or 0.2 to 1 bar) proportional with the level in the tank to the level controller (LC).

The controller is the intelligent part of the loop and has three basic elements:

1. A *memory* for the desired level in the tank, which is called the *set-point* (*S*);
2. A *comparator* for the set-point versus measured value (MV), as reported by the transmitter, that produces the difference between these two, which is called the *control deviation* or *error*;
3. An *amplifier* that amplifies the control deviation and produces the output of the controller.

The control valve receives the output from the controller and subsequently controls the input of the process.

It should be stated clearly here that for a controller the input of the process is always the control valve or its equivalent. Through this element the controller can influence the process and consequently sense the results on the transmitter, the later one thus becoming the output of the process.

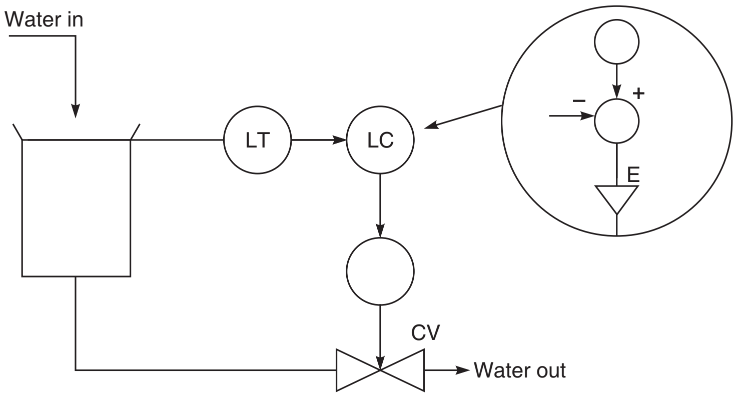


Figure 1.5 Level control.

In the example in Fig. 1.6, it can be seen that for the control circuit the so-called *information flow* goes upward in the process from the control valve to the transmitter, while the *material flow* clearly goes downward in this application.

All the processes should be considered as having three serial subsystems in the following sequence:

1. A *device* that is able to influence the input of the process. This is very often a control valve; in other cases it can be a speed-controllable motor. These devices are mostly commanded by receiving a standard size signal of 4 to 20 mA.

2. The *actual process*, for instance in Fig. 1.5, with the outgoing flow as input and the level as output. The behaviour of this process can be seen as the effect of a change in the control valve position upon the level in the tank.
3. A *device* that measures the output of the actual process and translates this value back into the 4–20 mA range, thus making the information understandable for the controller. The device is also called *sensor*, *transducer* or *transmitter*.

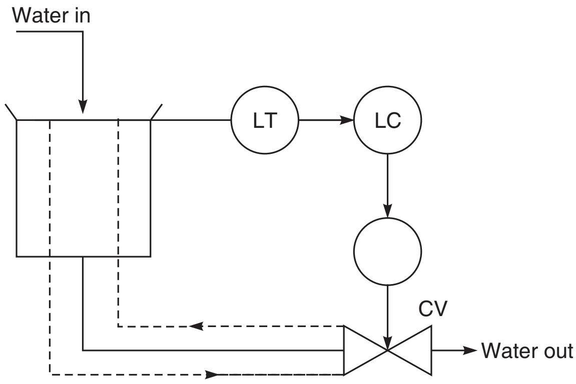


Figure 1.6 Information flow in level control.

It is very important to follow the above method of interpreting the processes to be controlled, for the understanding of the dynamic behaviour of these processes. The controller 'sees' the whole process in totality and sends a signal in the range of 4–20 mA (control signal) and who reports back with some kind of signal (through the transducer).

Next, the task of the controller in a control loop is basically very simple. Since the difference between the measured value (MV) of the level and the desired value thereof (SP) is calculated, the controller should reduce this control deviation (error) to nil. Therefore, if the control deviation is zero, apparently the control valve is in the right position.

If the level is lower than the desired value, the control deviation will be positive and the controller output will be higher, thus forcing the control valve to open up further.

The basic question for the controller, however, is: What should be the relation between the control deviation and the variation in the position of the control valve? Various types of controls are enumerated here.

1. Two-position control
2. Multi-position control
3. PID control
4. Ratio control
5. Cascade control.

1.6 TWO-POSITION CONTROL

Two-position control is the simplest and cheapest form of automatic control used in process control. This is used when the control variables need not be maintained at precise values. Typical examples are controls of alarm or shutdown functions.

In this mode of control the final control element is moved quickly from one of the two fixed positions to the other, depending upon whether the control variable is greater than the set value or not. If these two are fully open and fully close positions, then this mode is termed *on-off control* (Fig. 1.7).

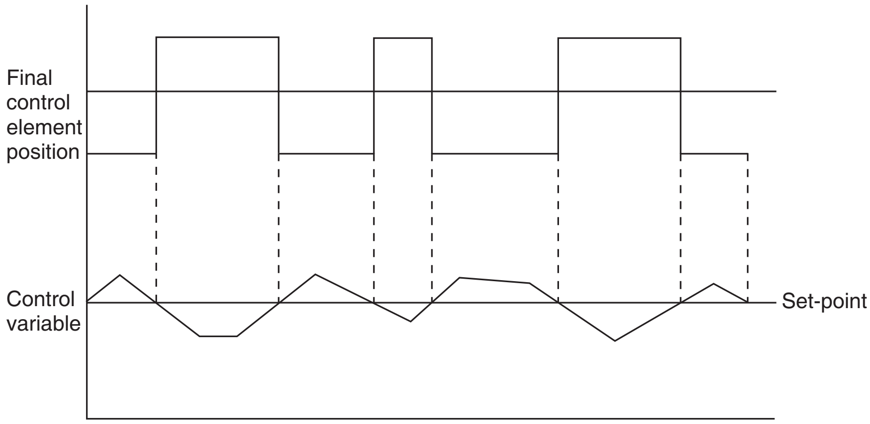


Figure 1.7 Two-position control.

A variation of the simple two-position control action is the two-position control with a differential gap in which a low set-point and a high set-point are defined. Figure 1.8 shows its characteristics. The valve is closed as the control variable crosses the high set value and remains closed till it crosses the low set value; then the valve is opened again.

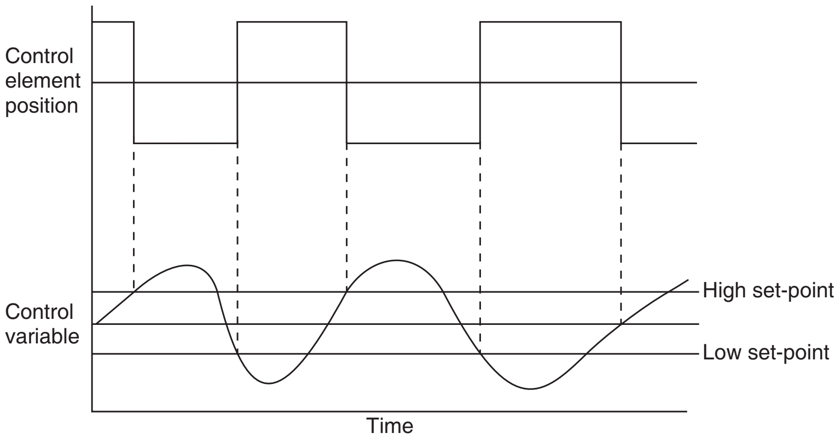


Figure 1.8 Two-position control with differential gap.

This type of control is often used in industries where large capacities are involved and energy inflows or outflows are small compared to the system capacity.

1.7 MULTI-POSITION CONTROL

In this control, the final control element is moved to one of three or more fixed positions, each corresponding to a definite range of values of the controlled variable, as shown in Fig. 1.9. In this control, the valve is closed when the control variable is above the high set value. When it is in the mid-band (i.e., between the high and low set values), the valve is half open and when it is lower than the low set value the valve is fully open.

The drawback of both two-position and multi-position controls is that they rarely produce exact correction. The control variable continues to cycle around the set-point with considerable deviation on either side of the set-point, due to the large capacitance and dead time involved, even when all the associated variables are constant.

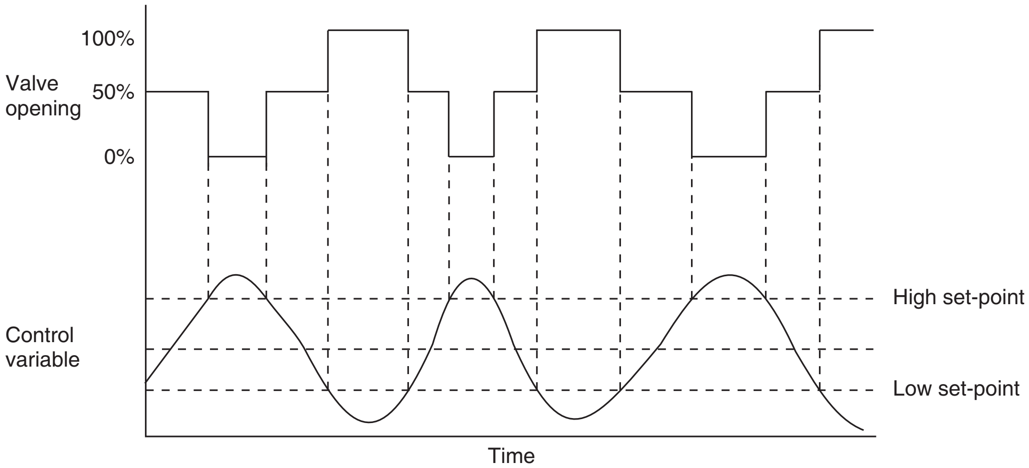


Figure 1.9 Multi-position control.

1.8 PID CONTROL

PID control means *proportional, integral or derivative control*, meaning that the control signal may be either proportional to, derivative of, or an integral of the error value. These three types may be used separately or in combination.

1.8.1 Proportional Control

In the systems with proportional control there is continuous linear relation between the value of the control variable and the correction applied. The applied correction is changed by the same amount for each unit of deviation (Fig. 1.10).

In proportional controllers, when the control variable deviates from the set value due to momentary disturbance, the controller gives a correction which is proportional to the deviation. The correction forces the controlled variable towards the set value, reducing the error, which in turn causes a reduction in the corrective action.

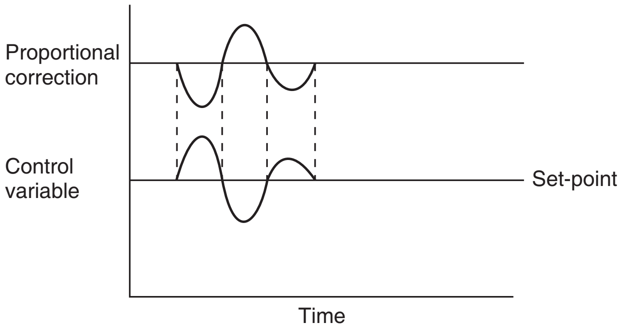


Figure 1.10 Proportional control.

The following equation describes the input-output relationships:

$$Q_i = \frac{100}{PB} \times e + Q_o$$

where

Q_i = Output at any given time
Q_o = Output when $e = 0$, i.e., zero error
e = Error signal, i.e., set-point value – measured value
PB = Proportional band in percentage.

Proportional band (PB) is defined as the percentage of full-scale change in input required to change the output from 0 to 100%.

The proportional gain is defined as

$$\text{Gain} = \frac{100}{PB}$$

Gain amplifies the error, i.e., the deviation between the set point and the measured value, to establish a power level. The proportional band (PB) expresses the gain of the controller as a percentage of the span of the instrument. A 20% PB in gain of 5 equates to 10% PB in gain of 10. If a controller has a span of 2000°C (1000°C around the set-point), then 20% PB (10% around set-point) will equate to control range of 100°C around the set-point. Suppose the measured value is 20°C below the set-point, then

$$\text{Gain} = 10 \text{ (in one direction)}$$

$$\text{Output} = 10 \times 20 = 200$$

Considering the span of the controller (1000°C around the set-point), this equates to 20% heat.

The proportional band determines the magnitude of response to error. If the proportional band is small, implying high gain, the controller will be over-responsive and will oscillate. On the other hand, if the proportional band is large, implying low gain, then it could lead to *control* *wander* due to lack of responsiveness. The ideal situation will be achieved when the proportional band is as narrow as possible without causing oscillation. The effect of change of proportional band on the measured value (temperature) is shown in Fig. 1.11. As proportional band is reduced, the measured value comes closer to the set-point and finally starts oscillating, making the system unstable.

If the deviation from the set-point, i.e., the error has been caused by variation in load variables, then the disturbance may persist. In this case the corrective action taken by the controller may be insufficient to bring the measured value closer to the set-point. This results in a residual error called *proportional offset*.

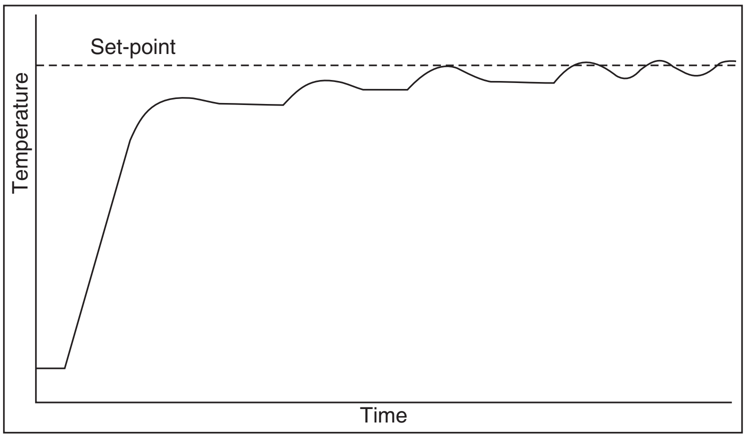


Figure 1.11 Proportional-only control.

For most processes the proportional controller has a disadvantage. To achieve a variation of the position of the control valve, it needs a control deviation. Hence, in the example of Fig. 1.5, when the incoming flow is higher than normal, logic dictates that to maintain the right level the outgoing flow must be higher and consequently the control valve must be partially opened. The proportional controller will only do so if at its input a control deviation exists and if, because of a continuous higher inflow, a continuous variation of the control valve position is needed; consequently a continuous control deviation will exist. As this continuous control deviation is due to the fact that the controller has a proportional algorithm, this effect is known as *proportional control deviation* or *proportional offset*.

This finally leads to the fact that the actual level in the tank is not the same as desired. To achieve the desired level, the set-point on the controller would have to be reset to a slightly lower value. This manual action to compensate the proportional offset is called *manual reset* (Fig. 1.12). Thus the limitation of proportional control alone is that it is not suitable for taking care of load variations.

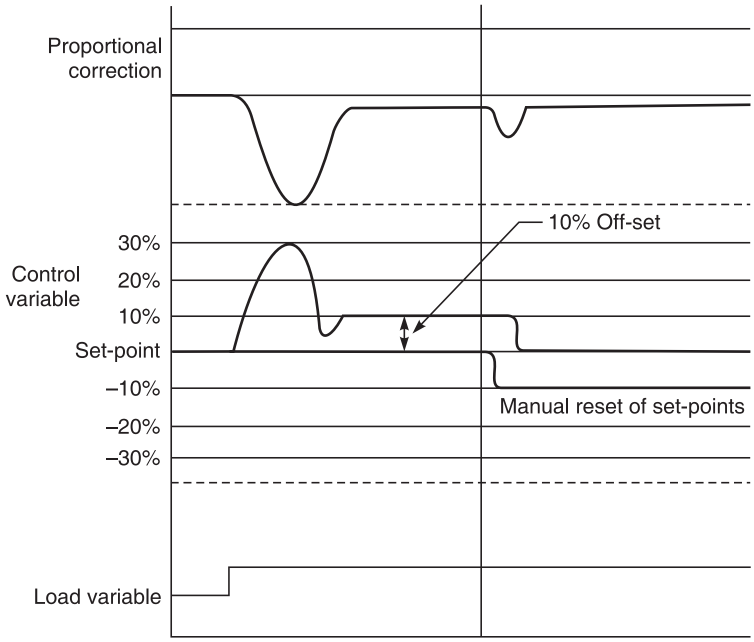


Figure 1.12 Offset in proportional control.

1.8.2 Integral Control

The integral control signal is proportional to the error signal integrated over a period of time. The integral action continues to build up correction till the error is forced down to zero. The integral control overcomes the drawback of the offset error present in proportional control, as the control responds to both magnitude of error and duration of error (Fig. 1.13). The correction given by the integral action is given by

$$I = \frac{1}{TI} \int e \cdot dt$$

where

I = Integral control signal
TI = Integral time constant
e = Error at any instant t.

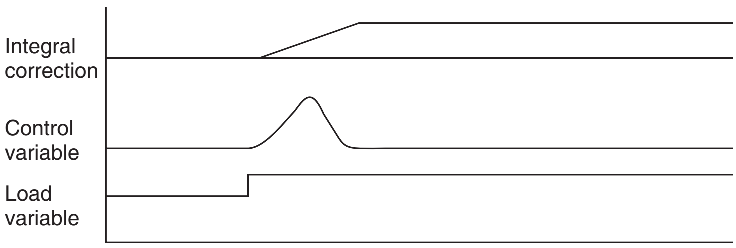


Figure 1.13 Integral control.

The limitation of integral control is that it is slow action, as correction builds up gradually.

The integral gain or integral constant is usually expressed either as a time constant in either minutes or seconds, or as the inverse of a time constant expressed in repeats per minute. If expressed as time constant *TI*, the longer the integral time constant the more slowly the controller will act, since longer integral time means fewer repeats per minute, which leads to slower response.

If the integral time constant is very small, then the response will be very fast. Thus power level will be shifted very quickly. This may cause oscillation since controller may act faster than the system it is controlling. On the other hand, a large integral time constant may result in very sluggish control.

Integral action (also called *automatic reset*) is probably the most important factor governing control at set-point. The integral term slowly shifts the output level as a result of the deviation of the measured value from the set-point. If the measured value is below the set-point in the heating control of Fig. 1.2, then the integral action will gradually increase the output power level till the error becomes zero.

1.8.3 Derivative Control

In this control, the correction is proportional to the rate of change of error. As soon as there is deviation, the derivative control generates a momentary excess correction which speeds up the corrective action (Fig. 1.14).

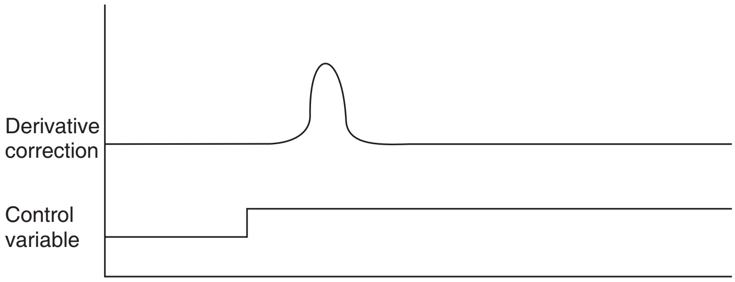


Figure 1.14 Derivative control.

The correction given by derivative control is

$$D = TD\,\frac{de}{dt}$$

where

D = Derivative control signal
TD = Derivative constant
e = Error signal at any instant t.

Derivative action (also called *rate*) provides a sudden shift in the output power level as a result of quick change in measured value. It is possible that hot water in Fig. 1.2 may be utilized in another process and thus a large deviation may have adverse effect. Therefore, when the measured value, e.g., temperature in Fig. 1.2 drops quickly, then the derivative term will provide a large change in the output power level in an attempt to correct the perturbation before it goes too far.

1.8.4 Proportional Plus Integral Control

This controller not only varies the position of the control valve proportional to the error, but also keeps enlarging this variation of the control valve position with time. This integral action of the controller continues with time, until the offset has completely vanished. Therefore, the integral action of the PI controller is also called *automatic reset*.

The correction given by the PI-controller is

$$PI = \frac{100}{PB} \times e + Q_0 + \frac{1}{TI}\int e \cdot dt$$

The controller with the PI algorithm enables the control loop to bring back the measured value (MV) to the set-point value. Figure 1.15 shows the action of the PI controller. The time needed to achieve the required results depends on the tuning of the controller.

A PI controller has the following two parameters to be tuned:

(a) The Proportional Band (PB)
(b) The Integral Time Constant (TI).

If a controller is tuned at 100 per cent, an error of 1 per cent will result in a controller output variation of 1 per cent. However, if the controller is tuned at 25 per cent PB, the same error will equate to an output variation of 4 per cent. Logic results in an output variation of 0.25 per cent for 1 per cent error at 400 per cent PB setting.

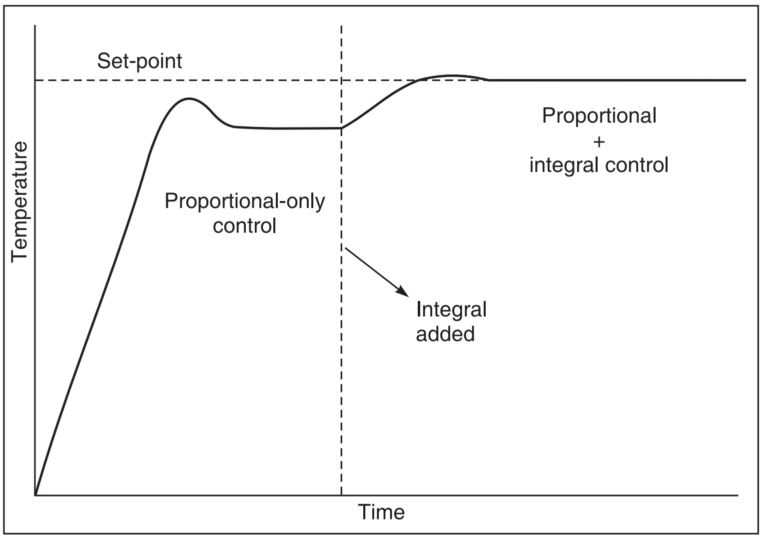


Figure 1.15 Proportional plus integral control.

The integration time is the time the controller needs to drag the output for an equal variation, as initially done by the proportional action of the controller. If a controller is tuned at 50 per cent PB and integration time is 20 seconds, then an error of 2 per cent, appearing suddenly, would result in a sudden variation of the control valve position of 4 per cent, a continuous rising of this 4 per cent to 8 per cent in the next 20 seconds, a further rise to 12 per cent in the next 20 seconds, and so on. Evidently then, in a normal operating control loop the process output would also change, and the error would diminish and finally disappear.

The tuning of a controller is a delicate matter, since excessively strong actions of a controller create instability. In such a case the output of the process will continuously vary between upper and lower limits, the span between these limits being larger as the controller actions are stronger.

As to the tuning of proportional and integral actions, however, the following rules can be stated:

(i) *Lower proportional band*. Stronger proportional action will result in faster reduction of offset and therefore create instability.

(ii) *Shorter integration time*. Stronger integral action will result in faster reduction of offset and create instability.

1.8.5 Proportional Plus Integral Plus Derivative Control (PID)

The algorithm used for more demanding tasks is the PID controller with proportional, integral and derivative action.

The correction given is represented by

$$PID = \frac{100}{PB} \times e + Q_o + \frac{1}{TI} \int e \cdot dt + TD \frac{de}{dt}$$

where *PB*, *TI* and *TD* have the usual meanings, as explained earlier.

PID controllers are used where the derivative action can help to compensate for lags in the process. For example, in temperature control loops, the adverse effect of lag associated in the temperature-measuring element can be partially reduced by derivative control action. The controller senses the rate of movement away from the set-point and starts moving the control valve earlier than in the cases of only proportional and proportional plus integral actions.

The derivative action can be best understood by referring back to the level control loop of Fig. 1.5 and presuming that this loop is initially in a stable condition, which means that the controller maintains a certain level on the set-point value. Next, the set-point is brought to a higher value.

While an intelligent operator handles the control valve, he would open this quite forcefully and consequently a quick rise in the level is seen. On the arrival or near-arrival of the level at the set-point value, the position of the control valve is reduced back to the original position, thus ensuring a quick fill-up of the volume necessary to get the level to a higher value.

After that, the operator would continue to control the level in the same way, as a PI controller would do, reaching an offset with adequate proportional actions and subsequently with I-type dragging as a function of time. Apparently the operator reacts in the beginning on any change in the error, the speed and the size thereof.

The derivative action acts exactly in the same manner on change in the error, and it gives an additional controller output change which is proportional to the size and speed of change of the error. The main problem with the derivative action is that it amplifies any noise in the process signal, producing fluctuations in the control valve position. Therefore it is used only on signals that are free of noise (e.g., temperature). It is not recommended in flow control or level control loops. It should be remarked, however, that a well-tuned derivative action gives faster control and tends to stabilize the control loop.

This mode of control employs the advantages of all the three following modes:

1. *Derivative action* reduces the overshoot which often occurs when integral action is added to proportional action.
2. *Derivative action* counteracts the lag characteristics introduced by integral action.
3. *Integral action* reduces the offset when added to proportional control.

Figure 1.16 shows the behaviour of a typical feedback control system, using different kinds of control, when it is subjected to permanent disturbance, like step change in the load variable.

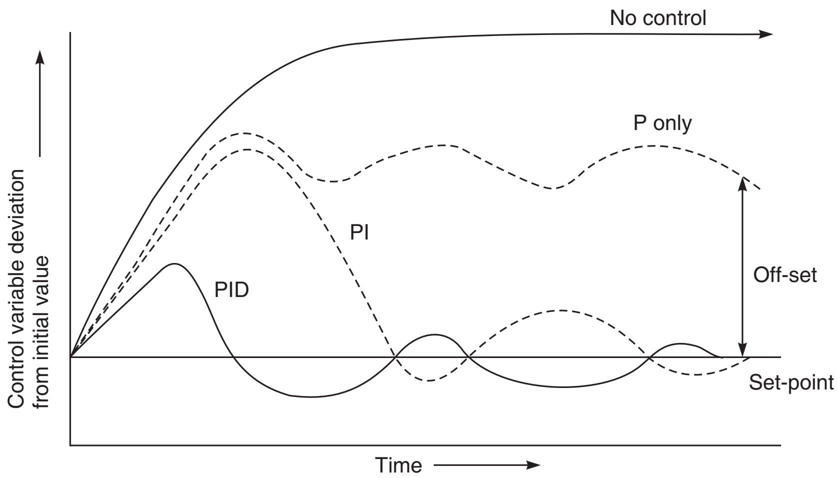


Figure 1.16 Behaviour of different feedback control actions (P, PI, PID).

1.9 CONTROLLER OPERATION

Figure 1.17 shows the feedback control loop with a switch known as *auto/manual switch*. When this switch is closed (auto mode), then the measured value from sensor is compared with the set-point and the error signal is generated. On receipt of the error signal the controller will generate the control signal which is fed to the control element (valve, motor, etc.) for controlling the process. All the actions take place automatically as in the normal feedback control loop operation.

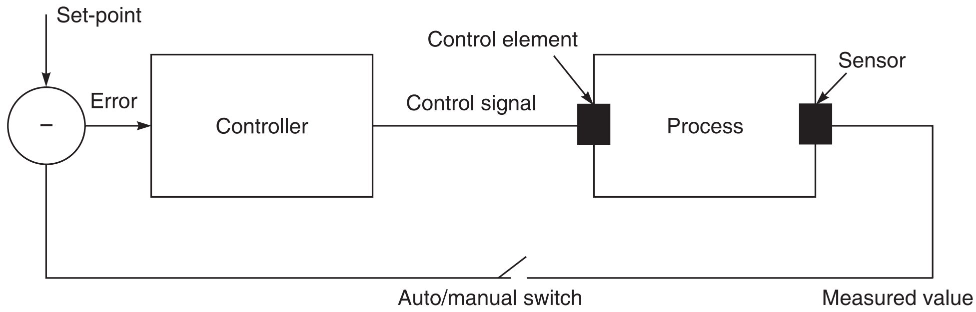


Figure 1.17 Feedback control loop with auto/manual switch.

On the other hand, if the switch is open (manual mode), then the measured value is not compared with the set-point and the error signal is not generated. The manual operator will generate the control signal by operating switches or knobs on the front panel of the controller. It is the same as the open loop control operation. Thus the manual operation mode is also called the *open loop control mode*. The automatic operation mode is the same as the feedback control mode.

A controller can be put in either auto or manual mode using auto/manual switch provided on the panel. Normally the controller is started in the manual mode and then switched to auto mode immediately. In auto mode the measured value will be compared with the set-point, and the error signal and the control signal will be generated. Now, if the measured value is not near the set-point, then a large error and consequently a large control signal will be generated, which will cause oscillation. Here the process will become unstable and remain so till the measured value comes near the set-point. This may even damage the process. In the water heating example (Fig. 1.2), if the set-point is 100 °C whereas the water temperature is 10 °C initially, the operator will control the water temperature till it becomes 70-80 °C in manual mode before switching to auto mode. This is known as *bumpless transfer*. If the number of loops is large, then it will become cumbersome for the operator to oversee the transfer from manual to auto mode for every loop. To avoid this, the controllers have the bumpless transfer facility. A controller with bumpless transfer, when switched to auto mode, will make the set-point value close to the measured value and then, in a gradual manner, bring it to the desired set-point.

1.10 CONTROL SYSTEM RESPONSE

It is important to understand the response of the process control system. This will help in controller adjustment and tuning. The system response may be categorized as either normalized response, underdamped response, overdamped response or critically damped response.

1.10.1 Normalized Response

When the process control system response is in medium speed and the measured value does not overshoot the set-point, the system response is categorized as *normalized*. The control system brings the measured value to the set-point at medium speed, but never allows it to overshoot. In an application like cola making, if the syrup temperature crosses the set-point, then the shelf life of cola will be adversely affected. Normalized response is also called *first order response*.

1.10.2 Underdamped Response

When the response of the process control system is faster and the measured value is allowed to oscillate (overshoot and undershoot) around the set-point slightly, then it is called an *underdamped response*. Figure 1.18 shows the underdamped response of a process control system.

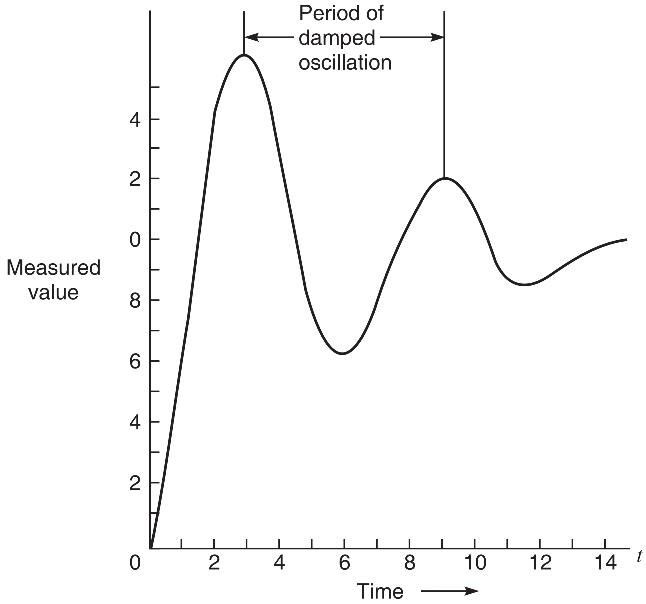


Figure 1.18 Underdamped system response.

The Quarter Amplitude Decay (QAD) is an underdamped response in which overshoot and undershoot are reduced in four periods and the measured value becomes equal to the set-point. This response is also called *4:1 decay ratio* or *¼ decay ratio response*.

1.10.3 Overdamped Response

When the response of a process control system is very slow, it is categorized as *overdamped*. The measured value does not overshoot the set-point and is brought to the set-point level slowly (Fig. 1.19).

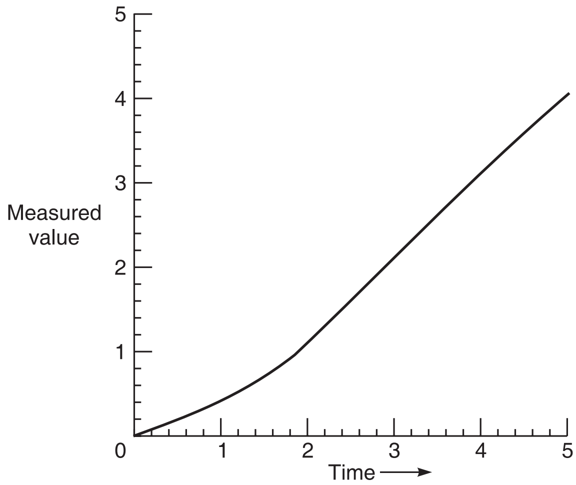


Figure 1.19 Overdamped system response.

Critically damped response

When the response of a process control is very fast, then it is categorized as *critically damped.* As shown in Fig. 1.20, the measured value is brought to the set-point level in minimum possible time.

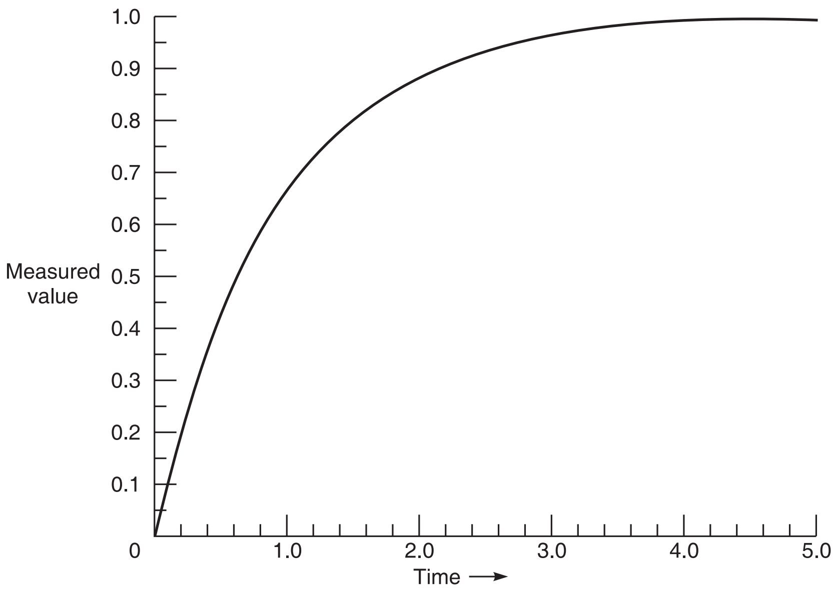


Figure 1.20 Critically damped system response.

1.10.4 Stability versus Response

The degree of stability of the control loop is also judged by the type of response. For oscillatory response, i.e., underdamped response, the degree of stability is indicated by the decay ratio, i.e., the ratio of successive peaks of the response. For non-oscillatory response (i.e., overdamped and critically damped response), the degree of stability can be expressed as the *damping factor*. The critical damping response has the damping factor of 1 and represents the fastest possible response without overshoot (Fig. 1.20).

Quarter Amplitude Decay response (Fig.1.21) is often used for the tuning process control loop. It is due to the reason that it represents a compromise between fast response and stability.

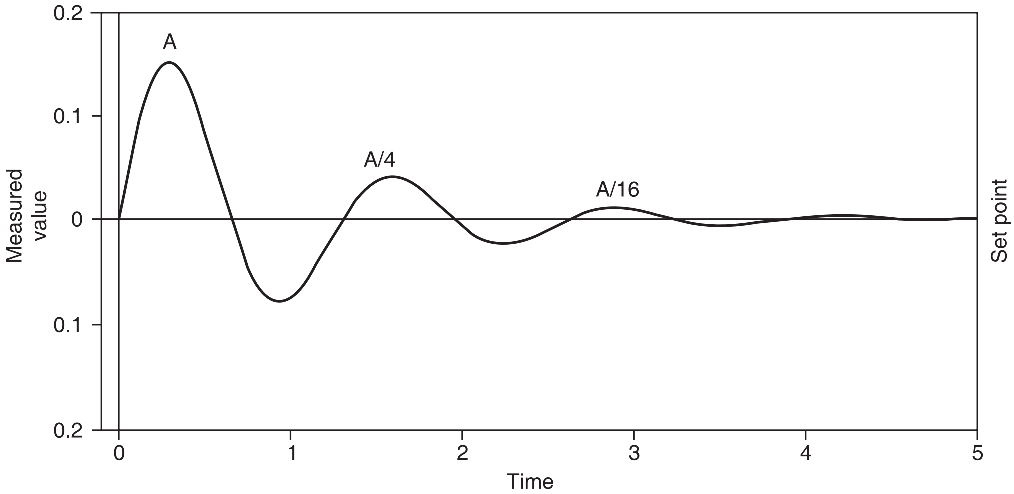


Figure 1.21 Quarter amplitude decay response.

A decay ratio higher than ¼ would produce a more sustained oscillation. (A ratio of 1 clearly indicates continuous oscillation without decay.) It would increase the danger of continuous oscillation under changed process conditions, thus making the control loop unstable.

1.11 CONTROLLABILITY OF PROCESS

Before attempting to control a process through P, PI or PID control strategy, it is advisable to find out whether the process is controllable or not. In fact, depending on the characteristics, a process may be categorized as *easy*, *medium* or *poor* for controllability assessment.

The controllability of a process may be derived from the process reaction curve shown in Fig. 1.22 as the result of a step action. The time *TU* refers to the dead time of the process as it is the time between the step action and reaction of the process. The time *TG* is the process time constant. The ratio *TG*/*TU* is used to determine the single control loop controllability. For a process to be easily controllable, the dead time must be low and thus the ratio *TG*/*TU* must be large. On the other hand, if the process reacts after a long time, one can conclude that the tuning of parameters of the PID controller will be difficult or may even be an almost impossible task. Controllability may be categorized as in the following empirical equations:

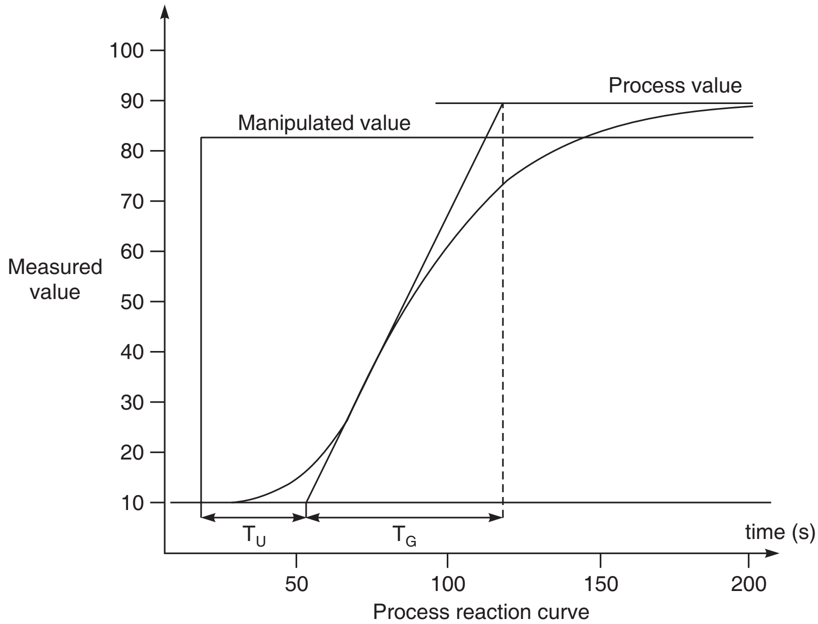


Figure 1.22 Determining controllability of process.

Easy controllability — $TG/TU \geq 10$
Medium controllability — $10 < TG/TU > 3$
Poor controllability — $TG/TU < 3$

Thus, if the *TG*/*TU* ratio is less than 3, or in other words, if the dead time is more than one-third of the process time constant, then the tuning of the conventional PID controller is bound to be a challenging task.

1.12 CONTROL LOOP TUNING

Tuning of control loop may be necessary while installing a new control loop for the process or reworking out the tuning in case of old loops. One of the parameters to be decided before loop tuning relates to the set-points of the control variables. It is possible that change in a set-point and tuning the process around the new set-point may directly affect the profit.

Figure 1.23 shows the relationship between profit-loss and the temperature set-point for a temperature control loop in a chemical plant. The set-point for the loop may be fixed between points A and B, so that the ultimate product is within the quality constraints. It must be understood that every product has a tolerance range for quality. If the product is outside the quality range, then it will incur losses due to high rejection rate. At the same time, to bring out a quality product one must not incur losses due to high energy consumption. The quality range must be translated to the set-point range. As shown in the Fig. 1.23, the maximum profit may be reaped when the process is controlled very close to point B. However, in order that the set-point constraint is not exceeded, one must keep a margin of standard deviation away from it. Thus the realistic set-point selected will be point C, as shown in Fig. 1.23. Two other factors will also come into play while selecting the set point. These are:

1. If the loop is controlled better, then point C can be closer to point B. In this type of loop, minimizing the effect of upsets through proper tuning will bring point C very close to point B and reap maximum profit. However, in case the control is poor, point C must be faraway from point B.
2. Loop stability, if poor, will further move point C away from point B.

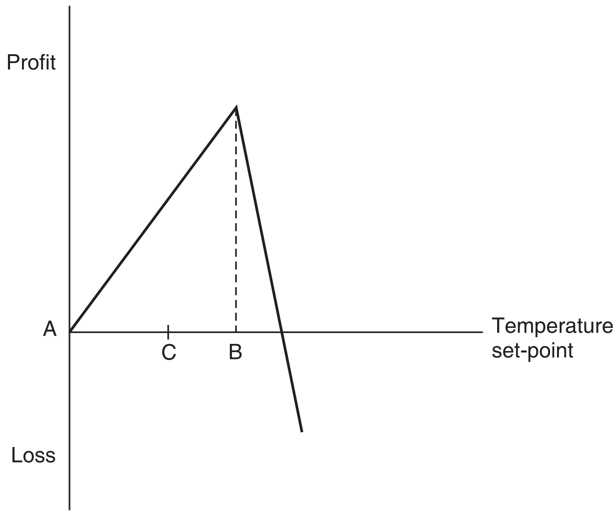


Figure 1.23 Profit–loss versus temperature set-point.

1.12.1 PID Controller Tuning Techniques

After the set-point has been decided by going through the logic enunciated above, it is important to ensure that the controller is properly tuned, so that it regulates the process to adhere to the set-point constraint. Proper tuning of the controller is very important as it will improve product quality, reduce wastage, shorten downtime and consequently save money and increase profit.

Tuning of a PID control loop primarily involves the adjustment of its control parameters like proportional gain, integral time, and derivative time to the optimum values for obtaining the desired control loop response. The desired response in a majority of cases is either the fastest response to a set-point change or the fastest return to the set-point after a load change. The loop stability is to be preserved while tuning. The objective of the controller tuning method is to obtain the fastest response consistent with the stability requirement.

The response time of the process is another important factor. It may take between a few minutes to a few hours for a set-point change to produce a stable effect. In addition, some processes have a degree of nonlinearity, so parameters that work well at full load condition don't work when the process is starting up from no load.

A number of techniques for the tuning of the PID controller have been proposed.

When a set-point or load change occurs, the control loop response is as shown in Fig. 1.24. The controller response covers both the magnitude and the time duration of the process deviation around the set-point. The deviation is large in the beginning but gradually comes down and becomes zero. One of the criteria to determine the effectiveness of the controller is the area of the response curve between the measurement and the set-point line. The minimum the area, the better is the controller in terms of tuning.

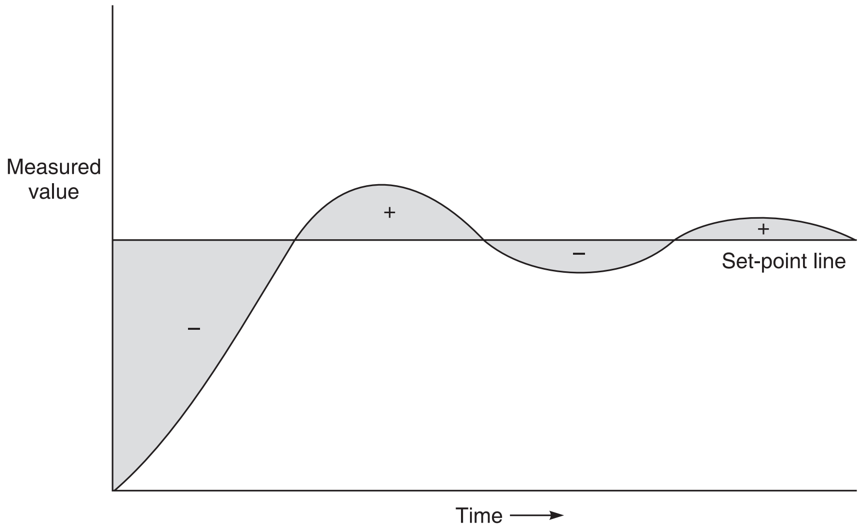


Figure 1.24 Idealised control loop response.

Based on the above general criteria, the following four minimum error integral tuning criteria have been developed:

(a) Integral of the Absolute Value of the Error (IAE)

$$IAE = \int |e(t)| \cdot dt$$

(b) Integral of the Square of the Error (ISE)

$$ISE = \int e^2(t) \cdot dt$$

(c) Integral of the Time-Weighted Absolute Value of the Error (ITAE)

$$ITAE = \int t \cdot |e(t)| \cdot dt$$

(d) Integral of the Time-Weighted Square of the Error (ITSE)

$$ITSE = \int t \cdot e^2(t) \cdot dt$$

These mathematical criteria are used primarily for academic purposes together with process simulations for the study of control algorithms.

For practical PID control loop tuning, two methods are commonly used. These are:

1. Process Reaction Curve Technique
2. Closed Loop Cycling Method.

These techniques were proposed by J.G. Ziegler and N.B. Nicholis in 1942. Both the techniques use the achieving of the 4:1 decay ratio as the optimal tuning method. As proposed by Ziegler and Nicholis, Fig. 1.25 shows the Quarter Amplitude Decay, i.e., the 4:1 decay ratio.

As an example, if the initial perturbation is of + 40 degrees, then the subsequent response of the controller should result in an undershoot of – 40 degrees, followed by an overshoot of + 2.5 degrees.

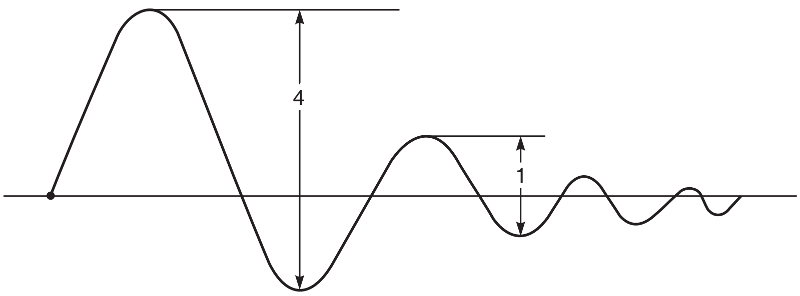


Figure 1.25 Quarter amplitude decay.

1.12.2 Process Reaction Curve Technique

To obtain the process reaction curve, the controller is disconnected from the process and a step input of power is made. The power level input, *P*, may be any convenient safe amount but should be introduced when the process is stable. A chart recorder may be used to obtain what is known as the process reaction curve, i.e., the graph of the measured value with time. A simple process reaction curve is shown in Fig. 1.26.

In Fig. 1.26, the time *L* is the lag time (also called *dead time*). It is due to the transportation lag: A straight line drawn tangent to the process reaction curve at the point of inflection will have a slope *R*.

The PID values may be calculated using these two terms by using the equations given below:

$$PB = \frac{R \times L}{P} \times \frac{100\%}{\text{Span}}$$

$$TI = 2 \times L$$

Proportional band *PB* is expressed as a percentage of the instrument span. *TI* (integral) and *TD* (derivative) are time constants and are expressed in minutes. *P* is the percentage power level, used as the step input, divided by 100% and is expressed as a fraction.

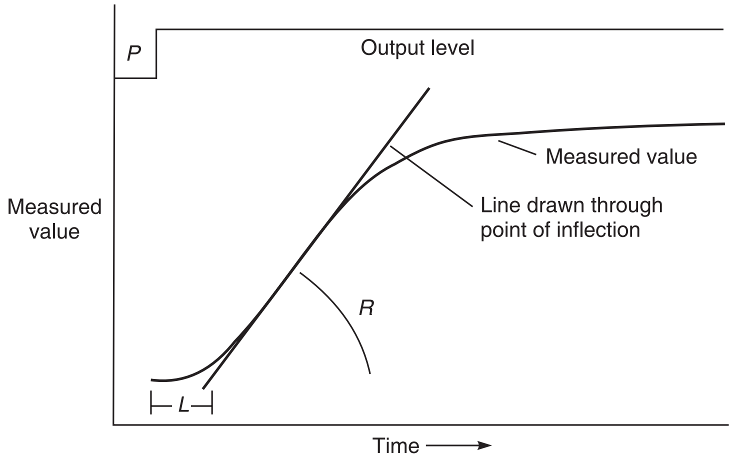


Figure 1.26 Simple process reaction curve.

G.H. Cohen and G.A. Coon expanded the process reaction curve technique after thorough evaluation. Figure 1.27 shows the expanded process reaction curve for a heating process.

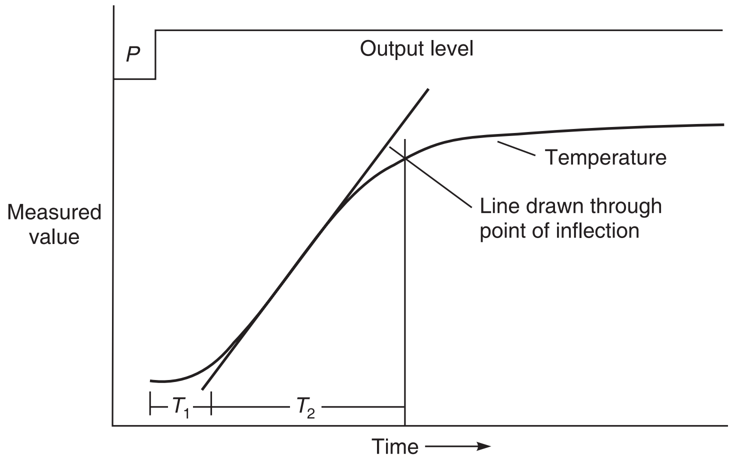


Figure 1.27 Expanded process reaction curve.

Now, if P = Step input of power used for evaluation, expressed as a percentage of the maximum allowable power, and

T = Final attained temperature expressed as a percentage of the span of the instrument as a result of the step input P, then

$$K = \frac{T}{P}$$

Using Table 1.1, the values of PB (Proportional Band), TI (Integral Time Constant) and TD (Derivative Time Constant) may be calculated.

Table 1.1 Tuning Constants—Process Reaction Curve

Controller	*Proportional band (PB)*	*Integral time constant (TI)*	*Derivative time constant (TD)*
Proportional only (*P*)	$\frac{KT_1}{T_2\,(1 + T_1/3\,T_2)}$	Not applicable	Not applicable
Proportional and integral (*PI*)	$\frac{KT_1}{T_2\,(0.9 + T_1/2\,T_2)}$	$\frac{T_1\,(30 + 3T_1/T_2)}{9 + 20\,T_1/T_2}$	Not applicable
Proportional and derivative (*PD*)	$\frac{KT_1}{1.25T_2 + T_1/6}$	Not applicable	$\frac{T_1\,(6 - 2\,T_1/T_2)}{22 + 3\,T_1/T_2}$
Proportional integral and derivative (*PID*)	$\frac{KT_1}{T_2\,(1.3 + T_1/6\,T_2)}$	$\frac{T_1\,(32 + 6\,T_1/T_2)}{13 + 8\,T_1/T_2}$	$\frac{4T_1}{11 + 2\,T_1/T_2}$

It must be noted that both the techniques are aimed to achieve quarter amplitude decay, which may not be suitable for all applications, and a suitable trade-off may be needed to be worked out. To reduce overshoot and increase the settling time, proportional band and integral time constant may be increased.

1.12.3 Closed Loop Cycling Technique

In this technique, the characteristic frequency of the process is determined. The characteristic frequency is considered to be an accurate representation of the system's responsiveness, and it can therefore be used to derive the controller time constants.

To derive the characteristic frequency, the integral and derivative time constants are made zero, and therefore the PID controller is reduced to a proportional-only controller. The proportional band is reduced till the control loop oscillates. The control loop cycles with a characteristic frequency (Fig. 1.28). The time period T may be used to determine controller time constants.

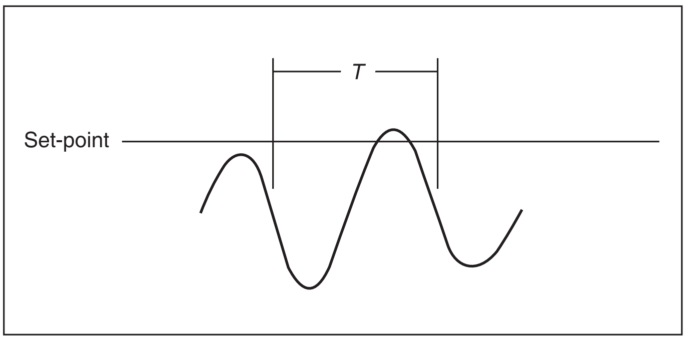


Figure 1.28 Closed loop cycling.

The following is a step-by-step procedure to tune the PID loop using the closed loop cycling method:

1. Make
   Integral Time Constant = 0
   Derivative Time Constant = 0.
   Thus eliminate integral and derivative action from the controller.
2. Decrease the proportional band to the point where a constant rate of oscillation is obtained. The system is now oscillating at its characteristic frequency. Measure the period of oscillation *T*.
3. Widen the proportional band until the process is just slightly unstable. This is the proportional band's *ultimate sensitivity*.
4. The proportional band's ultimate sensitivity width is defined as *P* and is used for calculating the actual proportional band.

The proportional band *PB*, integral time constant *TI* and derivative time constant *TD* can be determined as follows:

Controller	*Proportional Band (PB)*	*Integral Time Constant (TI)*	*Derivative Time Constant (TD)*
Proportional Only (P)	2 *P*	—	—
Proportional and Integral (PI)	2.2 *P*	0.8 *T*	—
Proportional Integral and Derivative (PID)	1.67 *P*	0.5 *T*	0.12 *T*

The above settings are for establishing control with Quarter Amplitude Decay. Some control processes may be either critically damped, overdamped or underdamped. Figure 1.29 shows the response curves for these control processes. The altered values of *PB*, *TI* and *TD* for these controllers are as follows:

Controller	*Proportional Band (PB)*	*Integral Time Constant (TI)*	*Derivative Time Constant (TD)*
Underdamped	*P*	0.8 *T*	0.125 *T*
Critically damped	1.5 *P*	*T*	0.167 *T*
Overdamped	2 *P*	1.5 *T*	0.167 *T*

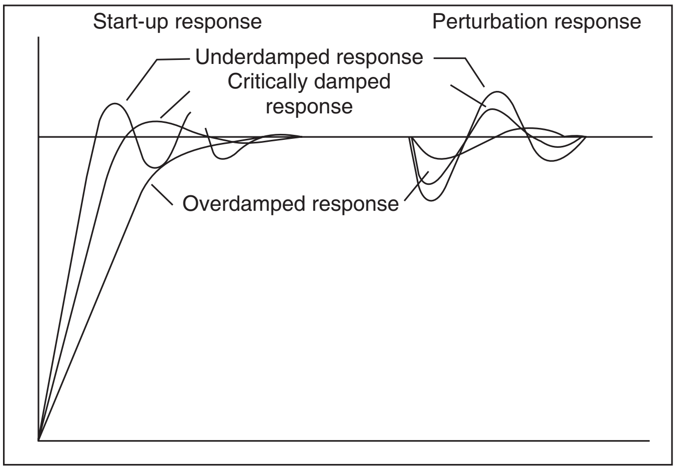


Figure 1.29 Typical response curves.

1.13 MULTI-VARIABLE CONTROL

So far, we have discussed about feedback control loops with one manipulated or controlled variable and one measured or control variable. In real life, however, the control depends on many control variables, though there is only one controlled variable for each control loop. Such control loops are known as *multi-variable control loops*. Multiple input signals jointly affect the action of the control system. Examples of multi-variable control are:

- Cascade control
- Ratio control.

1.13.1 Cascade Control

In many control schemes, the variation of some quality in the manipulated variable often degrades the control performance. In normal feedback, these changes have to enter the process, the effect is felt at the control variable, and then only is correction possible. Where such changes are frequent, the system will be upset very often. The cascade control scheme gives us a method to reduce this discrepancy. Figure 1.30 shows the cascade control block diagram.

In cascade control, one controller manipulates the set-point of another. Each controller has a sensor, but only the master controller has a set-point while only the slave controller has an output to process.

The secondary controller, manipulated variable and sensor constitute the inner loop. The outer loop consists of all elements including the inner loop. The dynamics of the inner loop should be faster than that of the outer loop, e.g., the flow through the inner loop and the temperature through the outer loop.

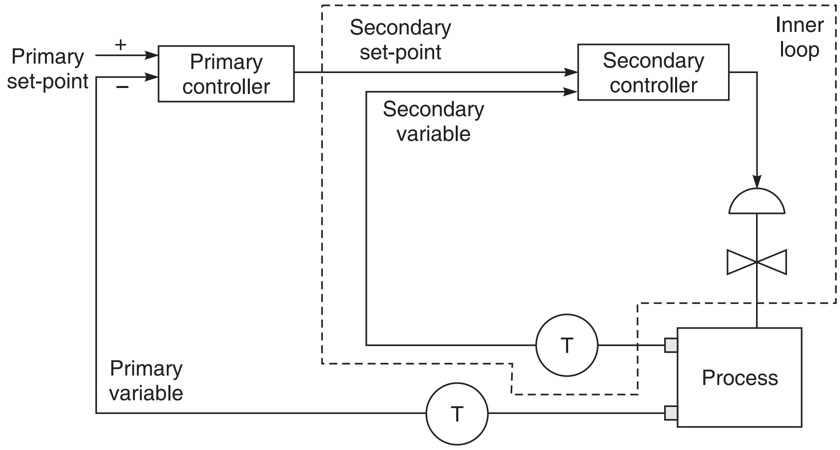


Figure 1.30 Cascade control.

The cascade control of water heating system is shown in Fig. 1.31. The temperature transmitter (TT) measures the temperature of the water outlet and sends the signal to the temperature control (TC), which compares the temperature the of water outlet the with the desired temperature (primary set-point) and adjusts the set-point for steam flow (secondary set-point) for the flow controller (FC). The flow controller receives the value of steam flow from the flow transmitter (FT), compares it with the set-point fixed by the Temperature Controller and sends the control signal to the control valve to effect flow control.

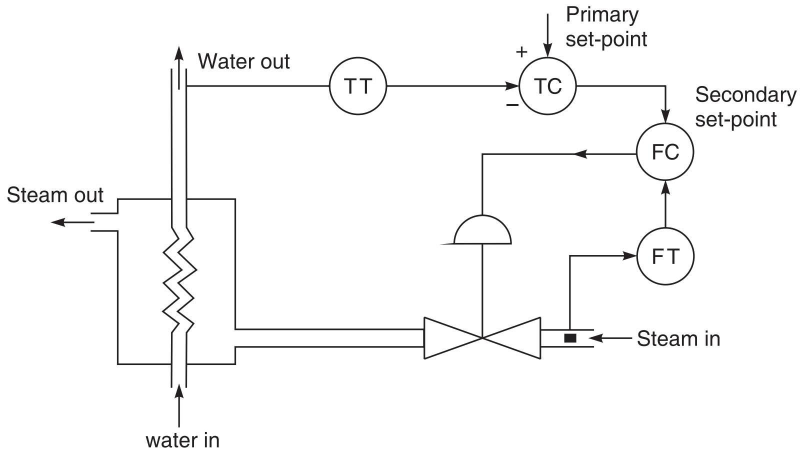


Figure 1.31 Cascade control of water heating system.

1.13.2 Ratio Control

In a ratio control system, a dependent variable (called *secondary variable*) is controlled as a function of an independent variable (*primary variable*). The latter may be a free variable, i.e., measured but not controlled, or it may be automatically controlled. Although ratio control is encountered most frequently in connection with the control of flows, it is also used for other variables like temperature, pressure, and composition. A typical system involving ratio control of two types of flow is shown in Fig. 1.32.

An example of ratio control is the feed control of ingredients in various processes. In an automobile, a particular ratio of tetraethyl lead to gasoline is to be maintained to produce the desired octane number. Feed control to a chemical reactor is another example where a particular ratio of the different components is to be maintained. Other examples include cement kiln speed versus slurry flow control, steam flow versus airflow in boiler control, etc.

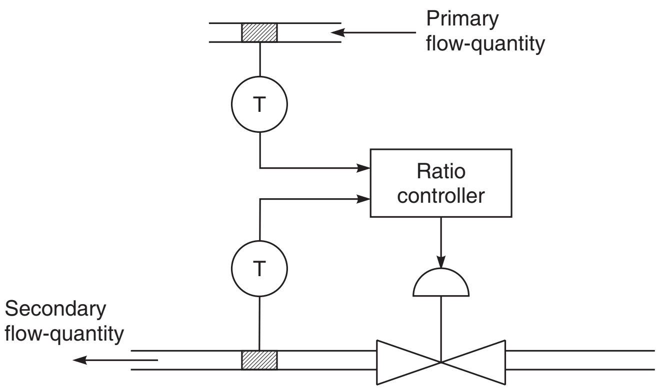


Figure 1.32 Ratio control.

1.14 CONTROL LOOP ROBUSTNESS

The important questions here are: Once a loop has been tuned for best performance, does its performance change? Will the performance change affect the stability of the control loop?

The answer to the first question is—yes. The performance of the loop declines with time even if no parameter has been changed. It has been noticed that the performance of the loop declines by 50% over six months. This may be due to simple wear and tear of the process plant, control valve and other parts. Other additional factors that may play a part in this change are:

- Change in plant feed
- Change in valve
- Physical modification like change in tank height, diameter, pipe build-up, etc.

To answer the second question, let us define three parameters, namely process gain, dead time and lag time.

The process gain is the ratio of the change in process value, i.e., the measured value over the change in controller output. Thus, if the process gain is higher, then the change in the controller output will result in a higher change in the measured value. The dead time is the amount of time that it takes for the measured value to change after the controller output changes. The lag time is the amount of time after the dead time period that the measured value takes to reach 0.632 of the final value after a step change in the controller output. The lag time gives us the capacity element of the process.

The robustness of the control loop can be determined using process gain and dead time by drawing a curve called the *robustness plot*. The robustness plot shows how sensitive the loop is to process changes. Figure 1.33 shows the robustness plot for a control loop. On the X-axis is the process gain, whereas on Y-axis is the dead time of the process. The plot has two regions: a stable region and an unstable region. These two regions are separated by a line called the *verge of instability* or the *curve of marginal stability*.

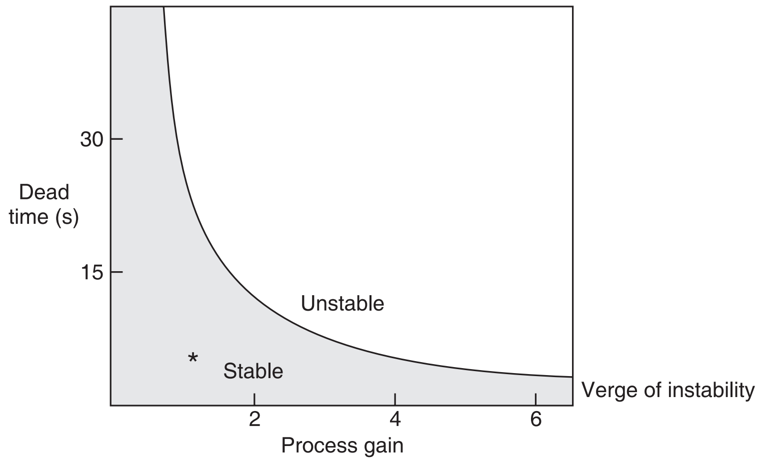


Figure 1.33 Robustness plot of control loop.

The* in the lower portion (stable portion) of the plot shows the current dead time and the process gain of the loop. If the process gain is increased (by changing the proportional, integral and derivative constants), this* will move towards the right and will reach the verge of instability. In such a case, any load change will put the loop in oscillation. The measured value will oscillate around the set-point and the controller output will also oscillate continuously (Fig. 1.34).

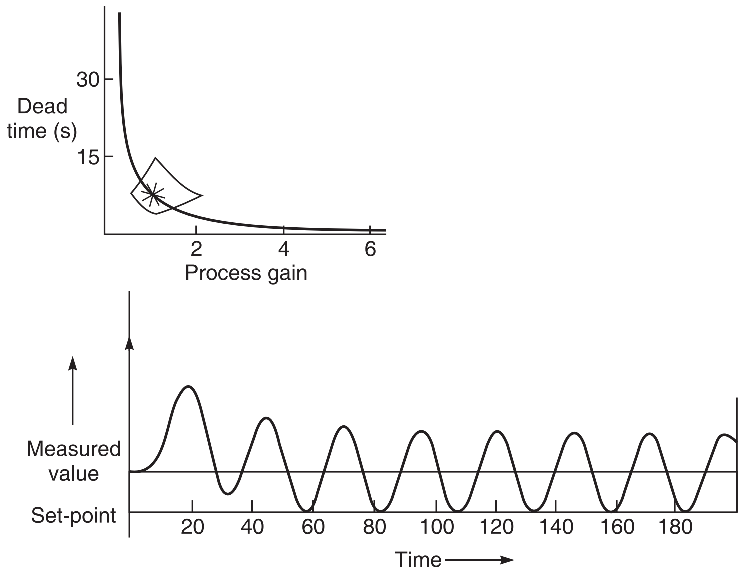


Figure 1.34 Control loop response on verge of instability.

Thus the current control loop parameters (marked by*) must be away from the verge of instability for good robustness. There is always a trade-off between tuning and stability. The robustness plot shows the effects of process change or difference in tuning on the robustness or sensitivity of the control loop.

1.15 FEEDFORWARD CONTROL

The purpose of feedforward control is to protect the control system against changes in load. We measure the signals which have the potential to upset the process and transmit these to the controller. The controller makes appropriate computations on the basis of the signals, calculates the new required value of the manipulated signal and sends it to the final control element. Then the control variable remains unaffected in spite of load changes. In this case the control variable is not 'fedback' but load variables are 'fed forward'. Figure 1.35 shows the block diagram of feedforward control.

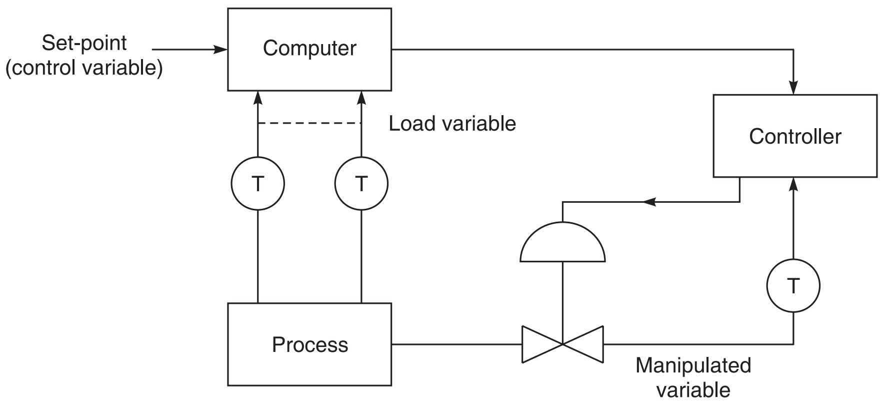


Figure 1.35 Feedforward control.

Thus in feedforward control strategy, information concerning one or more conditions that may disturb the control variable is converted into corrective action to minimize its deviation. In our example of cascade control of the water heating system (Fig. 1.32), if load variables—like the temperature of input water and the flow of input water—change, then control will becomes sluggish since the temperature of the water outlet will first drop and then only the control action be initiated. Using feedforward control, the process disturbances may be anticipated and estimated, and corrective action can be initiated even before the control variable value has changed, so as to minimize the deviation.

Figure 1.36 shows the flow and temperature transmitters, which measure the flow and temperature values and later feed these to the computer. On the basis of these values (water flow, actual and desired temperature of the water, and set-point), the computer calculates the amount of heat required by the water to acquire the desired temperature. This is used to control the steam flow. The disadvantage of feedforward control is that although there are a number of variables which affect the control variables, all of these cannot be measured. In the present example, heat dissipation through a walls of the container, inaccuracies in flow and temperature measurement are some of the variables which affect the process. In order to compensate for these inaccuracies, a feedback loop is included in the feedforward loop (Fig. 1.37).

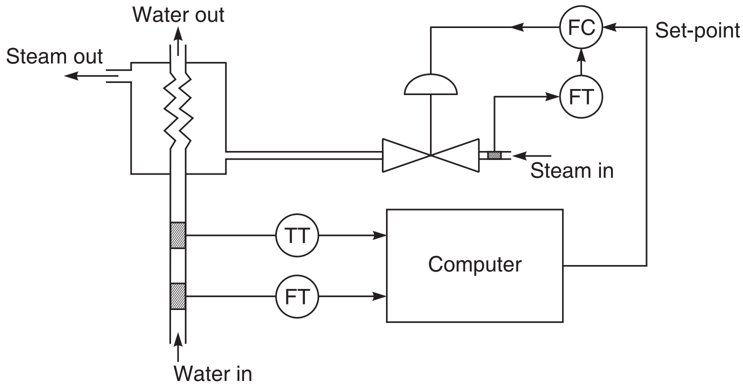


Figure 1.36 Feedforward control of water heating system.

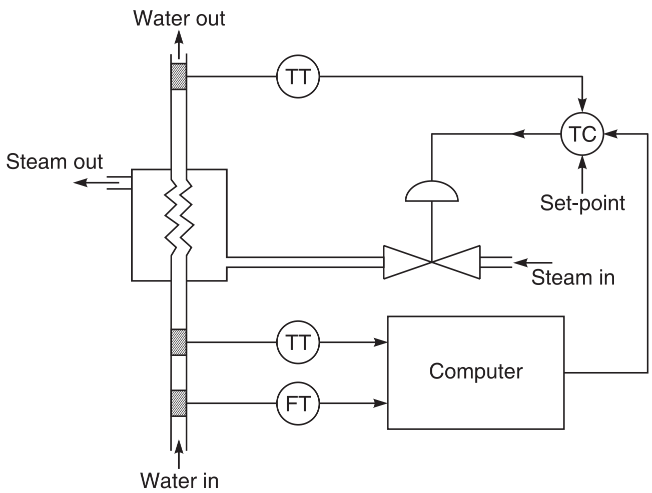


Figure 1.37 Feedforward–feedback control combination for water heating system.

1.16 CONCLUSION

In this chapter, we have introduced the basic concepts associated with process control systems. These concepts will help us in attaining a better understanding of the contents of the subsequent chapters.

SUGGESTED READING

Anstrom, K., and Haggland, T., *PID Controller: Theory, Design and Tuning*, ISA Press, Research Triangle Park, North Carolina, 1995.

Cadzow, J.A., and Powell, J.D., *Digital Control of Dynamic Systems*, Addison Wesley, Reading, Massachussetts, 1980.

Gagnepain, J.P., and Seborg, D.E., Analysis of process interactions with applications to multiloop system design, *Ind. Engg. Chem. Pro. Des. Dev*., **21**, No. 1, pp. 5–11, 1982.

Isermann, R., *Digital Control Systems*, Springer-Verlag, New York, 1981.

Lefkowitz, I., and Schoeffler, J.D., Multilevel control structures for three discrete manufacturing processes, *Proc. IFAC World Congress*, Pergamon Press, Oxford, 1972.

Leish, J.R., *Applied Control Theory*, Peter Peregrinus, Hertfordshire, UK, 1982.

Martin-Souchez, Z.M., A new solution to adaptive control, *Proc. IEEE*, **64**, No.8, pp. 1209–1218, 1976.

Morris, H.M., Control processors move next to the process, *Contr. Engg*., **34**, No.1, pp.87–90, 1987.

Owens, D.H., Some unifying concepts in multivariable feeback design, *Int. J. Control*, **33**, No. 4, pp. 701–711, 1981.

Ray, H.W., *Advanced Process Control*, McGraw-Hill, New York, 1981.

Robinson, C., Instrumentation 1986: Emerging technologies secure a foothold on the process plant floor, *InTech*, **33**, pp. 203–207, 1986.

Schoeffler, J.D., The incorporation of advanced control techniques in process control, *Proc. 7th Adv. Control Conf*., pp. 17–46, 1981.

Swanda, A.P. and Seborg, D.E., Controller performance assessment based on set point response data, *Proceedings of the American controller conference*, 1999.

Visioli, A., Optimal tuning of PID controllers for integral and unstable processes, *IEEE Proceedings on Line No. 20010197*, 2000.

Weide, B.W., et al., Process control: Integration and design methodology support, *IEEE Computer*, **17**, pp. 27–32, 1984.

CHAPTER

2

Transducers: Present and Future

2.1 INTRODUCTION

> I often say that when you can measure what you are speaking about and express it in numbers, you know something about it, but when you cannot measure it, you cannot express it in numbers your knowledge is meagre and or unsatisfactory kind. —Lord Kelvin

The above exhortion may appear debatable or even naive to a philosopher, to a pure mathematician or to an artist but it is of profound significance to the students of engineering and life science and to scientists and engineers.

Lord Kelvin's statement seems to spring from David Hume's or Charval's philosophy that our knowledge and understanding of the universe is nothing more than the knowledge conveyed to us through our sensors. Thus, it is to be recognized that our ability to understand and control the nature rests primarily on our ability to devise artifacts in the form of sensors which interface us to the nature enveloping us.

There are basically three kinds of transformations which are continuously occurring in the nature (Fig. 2.1). These transformations pertain to matter, energy and information. If we consider

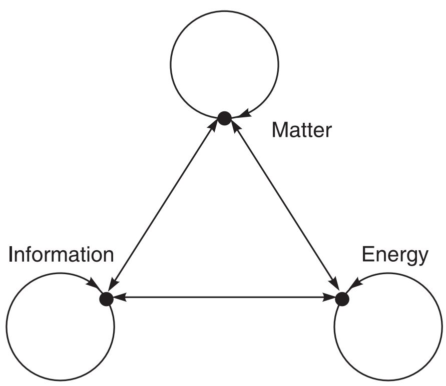


Figure 2.1 Transformations occurring in nature.

the phenomena of conservation of energy and conservation of rate of entropy production, we find that there are only two fundamental transformations, namely, energy and information. A matter transformation can therefore he visualised as energy transformation. If we isolate the man from environment, we find that man is continuously exchanging matter, energy and information with the environment and in this manner man is an open system par excellence. In fact, man's ability to transform matter and energy is extremely limited. However, he has almost unlimited capacity of information transformation. It is this capacity which we use to devise ways and means to control other two transformations. It is also observed that all energy and information transformation occur simultaneously, i e., they are equipresent (Fig. 2.2).

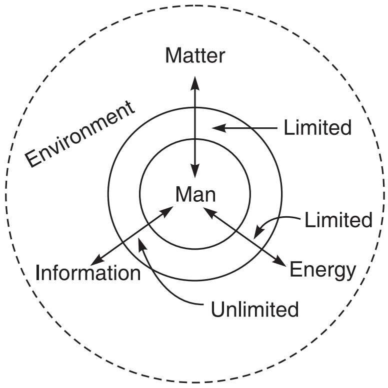


Figure 2.2 Simultaneous occurrence of energy and information transformation.

In this chapter we shall study various types of transducers devised by man to collect information regarding nature.

2.2 TRANSDUCER—DEFINITION AND NATURE

There seems to be tremendous confusion regarding the definition of transducers. Different texts have advanced different definitions and even the various professional societies like ISA, IEEE do not give a consistent definition. What is more striking is that sometimes not only do they connote the different meanings, but also they are fundamentally wrong. For example, the widely accepted definition that transducer is an energy conversion device includes not only sensor but also actuator and even processor. By transducer we definitely do not mean a motor or a generator. For that matter this definition includes every being in a dynamic system.

Many terms like instrument, transmitter, detector are being generally used in modem industries in place of transducers. However, the only term which merit consideration as an alternate is the *sensors*.

From our discussions so far, we may deduce that the transducers should be defined as "a device which affects transformation of information from one form of energy to another".

Thus in a transducer, there is a maximum information transformation and minimum energy transformation. This definition clearly excludes transmitters, processors and actuators from its scope. However, a widely used and accepted definition of transducer is that, “it is a device which provides a useful output in response to specific measurand”. The measurand being a physical quantity, property or condition which is measured.

2.3 TRANSDUCER FUNCTIONS

If we analyse the above definition it is evident that the transducers perform two major functions. These are:

— to measure/to sense, and
— to convert the measurand value to a useful output.

Depending on the type of output from the first function, the conversion function may or may not be present. We will analyse these two functions separately.

2.3.1 Measurement

While selecting a transducer for any specific measurand, one should consider the following criteria:

— Type of measurand
— Number of measurements
— Sensing element
— Transduction element
— Range of measurement

2.3.2 Conversion

The conversion involves presentation of measured value in a specific format to the user. In some cases the measured value is amplified also. For example, some transducers involve the conversion of AC output from the transduction element to DC output. Some transducers even convert analog signal to digital so that they can be interfaced to computer directly.

2.4 CHARACTERISTIC OF TRANSDUCERS

2.4.1 Measurand Characteristics

A transducer is normally designed to sense the specific measurand and to respond only to this measurand. However, in some cases, measurand may even be calculated by their relationship to the measurand sensed by the transducer. For example, pressure transducer measures pressure; displacement transducer measures displacement; acceleration transducer (accelerometer) measures acceleration. However,

— displacement transducers can be used to measure position;
— displacement transducers can be used to measure velocity; and
— acceleration transducers can be used to measure velocity.

The higher and lower limits of measurand value form the range of transducer. The range may be unidirectional (0–10 psid—pounds per square inch differential), bi-directional (±3 g), asymmetrically bi-directional (–2 + 10 g) or expanded (3200 to 3800 rpm—rotations per minute). The algebraic difference between the limits of the range is called the span of the transducer. For example, the span of –2 + 10 g accelerometer is 12 g.

2.4.2 Electrical Characteristics

Transducers normally produce analog output in the form of current, voltage ratio, voltage amplitude or variation of other parameters such as capacitance, inductance etc. The output can also be produced in the form of frequency (for example: 0 to 1224 Hz) or in the form of binary numbers which represent the output of transducers in digital era. Normally all transducers (except self-generating types, for example electromagnetic, piezoelectric etc.) require external electric excitation in the form of AC/DC voltage or current. The impedance (Z_{in}) measured across the excitation terminal is called the input impedance and across output terminals, the output impedance (Z_s) of the transducers (Fig. 2.3). The load impedance (Z_L) is the impedance presented at the transducers output by external circuitry and transmission line. Mismatch at Z_L and Z_{in}, causes load errors.

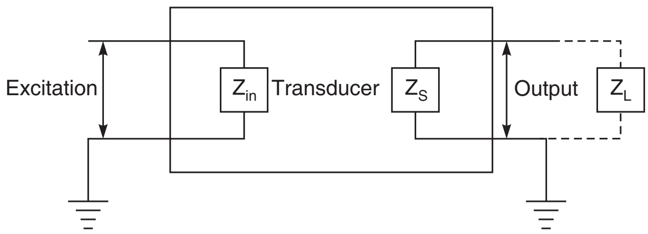


Figure 2.3 The input, output and load impedance across the excitation terminal.

The transducer with built in rectifier to convert AC to DC voltage output will present ripple in the DC output. In case a transducer is having integral amplifier, then amplifier characteristics are reflected at the output. These include output noise in the form of harmonics, gain instability and recovery time.

2.4.3 Static Characteristics

Transducer characteristics which are determined during a calibration cycle at room conditions are called static characteristics. To determine static characteristics, transducer calibration is performed in indoor conditions in the absence of acceleration, shock or vibration.

Accuracy and precision

Every transducer is expected to follow an ideal or theoretical output/measurand relationship. However, due to practical limitations the output of transducer is affected by the non-ideal behaviour. It causes the indicated measurand value to deviate from the true value. The difference between these two values is called transducer error which is normally expressed in percentage of Full-Scale Output (% FSO). The accuracy of a transducer is defined as the ratio of the error to the full-scale output.

Precision is the closeness with which measurements are distributed about their mean value. It refers to the degree of agreement on a set or group of measurement

Repeatability

The ability of transducer to reproduce output relation and same measurand value, when it is applied consecutively under the same conditions in the same directions is called the repeatability of a transducer.

Linearity

The closeness of a transducer calibration curve to specify a straight line is called the linearity of transducer. The transducers are designed with linear output/measurand relationship as this tends to facilitate data reduction.

Resolution

Output of transducers normally change in small discrete steps when the measurand is varied over the range. The magnitude of these output steps (called resolution when expressed in % FSO) is different at various steps in different ranges. Due to this reason, resolution is normally expressed as maximum resolution as the greatest of all the steps. Figure 2.4 shows the difference between accuracy and resolution in case of shooting. A transducer should possess both high resolution and high accuracy at the same time.

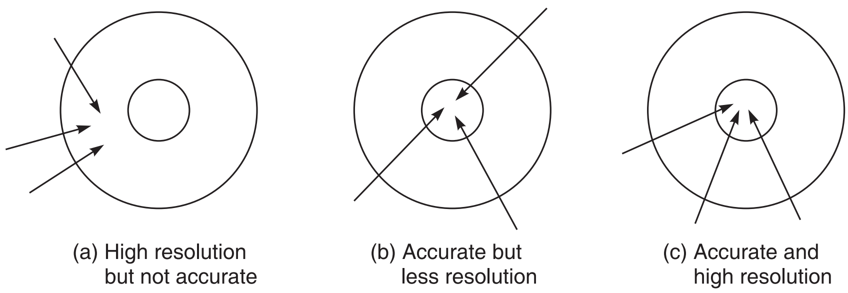


Figure 2.4 Difference between accuracy and resolution in case of shooting.

2.4.4 Environmental Characteristics

When the transducer operates under the conditions which are different from the room conditions, a number of errors can appear in the transducer output.

The environmental conditions of the transducer for storage, shipping and operation are well specified. Errors caused due to change in environmental conditions are called environmental errors. Environmental effects on transducers are normally temporary, i.e. transducers will start functioning normally as soon as operational environmental conditions are restored. However, in some cases the transducer can get damaged or its characteristics may get permanently changed.

2.5 TRANSDUCER CLASSIFICATION

There are several criteria on the basis of which transducers have been classified. We shall discuss here about these major criteria.

2.5.1 Classification Based on Energy

All electrical transducers have been classified under two categories: (a) Active transducer, and (b) Passive transducer.

Active transducers are self-generating type. They do not require electric energy. They work on the principle of conservation of energy. For example, thermoelectric and piezoelectric devices. *Passive transducers* are based on principle of energy controlling and they require a secondary electrical source for operation, for example strain gauge devices.

2.5.2 Classification Based on Technology

Transducers are also been classified on the basis of technology used for their design. For example, they are categorised as mechanical transducers, electrical transducers, electronic transducers etc.

2.5.3 Classification Based on Measurand

The transducers are classified by the 'measurand type' they are supposed to measure. For example, pressure transducers, displacement transducers etc.

Unfortunately, there is no standard way for the transducer classification but the classification based on the measurand type is most common in practice. We shall use this type of classification while describing the various types of transducers available today. The terms sensor and transducer have been used interchangeably in the book.

2.6 TECHNOLOGY TREND

2.6.1 Conventional Transducers

The transducer technology is quite varied in nature. Conventional technology uses the electric and electromechanical principles to sense various measurands. Linear Voltage Differential Transformer (LVDT) uses the inductance principle to measure the displacement/position. To measure temperature, the principle of change in resistance is used for RTD (Resistance Temperature Detector) and thermistors, whereas thermocouples use Seebeck's principle of thermoelectricity. Table 2.1 illustrates the conventional transducer technology in case of pressure and temperature.

2.6.2 Silicon Transducers

Silicon is known to be highly effective material for transducing many physical parameters including light levels, force and temperature. Silicon process technology is highly developed

Table 2.1 General Pressure and Temperature Sensing Techniques

Transduction method	*Principle of operation*	*Range*	*Approximate accuracy error*	*Advantages*	*Disadvantages*
Pressure					
Capacitive	Deflections of pressure diaphragm acting as one plate of a parallel plate capacitor cause capacitance changes	0.01 to 200 psi	0.05%	High accuracy and sensitivity; ruggedness; temperature insensitivity	High cost; unsuitability for high pressure
Inductive	Deflections of pressure diaphragm or Bourdon tube cause inductance changes in inductance bridge or differential transformer	0.04 to 100000 psi	0.5%	High outputs; wide pressure range	Instability with temperature; susceptibility to shock and vibration
Piezoelectric	Pressure on a quartz or Rochelle-salt crystal produces an electrostatic voltage across it	0.1 to 1000 psi	1%	No need for excitation; wide frequency-response, pressure, and temperature ranges.	Low output and accuracy; instability
Piezoresistive (strain gauges)	Pressure induced strain in sensing element causes resistance change in gauges	0.5 to 10000 psi	0.25 to 0.5%	High sensitivity low hysteresis and cost (semiconductor types); ruggedness; wide temperature range	Low output; temperature sensitivity
Temperature					
Thermoelectric (thermocouple)	Electromotive force is generated at the junction of two dissimilar metals, each at a different temperature	–200 to +2000°C	1 to 5%	Wide temperature range; high temperatures	Low output accuracy, and sensitivity; instability; high cost
Resistance (thermistors resistance temperature detctors)	Resistance changes because of temperature in metal oxides of metallic conductors	–100 to +400°C	1 to 10%	(Thermistors) high output and sensitivity; low cost (RTDs) high accuracy, and linearity; wide temperature range	(Thermistor) nonlinearity; small temperature range (RTDs) high cost; long thermal time constant
Semiconductor diode	Base-emitter voltage of a forward-biased diode changes with temperature	–55 to +200°C	0.1 to 1%	High accuracy, stability and linearity; low cost range	Low output; limited upper-temperature

and welt suited to high volume production. Work on high performance linear circuits, such as data converters bas revealed that technology is capable of high precision, required for many sensing applications. Since transducers are devices producing low level analog output, some signal conditioning (e.g. amplification or encoding) is often required before transmission to digital world of computer. The designer will have an additional degree of freedom with silicon sensors which enable to integrate signal conditioning circuitry into the transducer chip.

The silicon sensors are rugged as well. The Young's Modulus of elasticity for silicon is greater than that for steel. This makes these sensors suitable for industrial applications. The analog signal outputs of transducers are converted to digital by Analog to Digital Converter. The digital information is then further processed by computer. Silicon sensors with on-chip signal conditioning and Analog to Digital Converter will soon start appearing in the market.

The development of silicon sensors fabricated using IC batch processing technology has spurred by space research and bio-medical applications in the late 1980s. Taken with the advances in LSI fabrication techniques, this development held the promise that a family of sensors for a variety of parameters could be fabricated in silicon at extremely low cost. As many as dozen firms have pursued this path.

2.6.3 Fibre-Optic Transducers

It has been well established that the communication via optical fibres confers many advantages over the more conventional metallic cable systems and these include,

— high information density over long distances;
— low losses;
— light weight;
— economy of the material with respect to metallic conductor like copper; and
— immunity from electromagnetic and atmospheric interferences such as fog, smog, rain and snow.

In the field of fibre-optic communication, problems due to high sensitivity of fibres to external influences like phase sensitivity, microbending losses and model noise are encountered. In the field of sensor technology, the same problems are exploited to develop sensors. Fibre-optic sensors have assumed considerable importance in instrumentation in recent years due to the following advantages:

1. High sensitivity as compared to other sensors.
2. Geometric versatility of sensors, which enables the realization of any arbitrary shape for them
3. Common technology base from which sensors for various parameters can be realised, e.g. sensors for acoustic, magnetic, thermal, and mechanical measurement are possible using fibre-optics.
4. Simplicity of technology
5. Ease of signal transmission over long distance.

Basically a fibre-optic sensor comprises of a Light Source (e.g. LASER, LED etc.), injecting a signal into sensor fibre, a Light Detector (e.g. Photodiode) for receiving the signal after the light has been modulated by the sensor fibre and an electronic system for processing the detected signal into useful electrical quantity (Fig. 2.5).

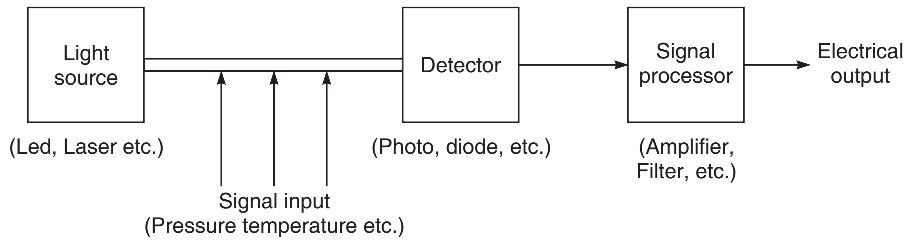


Figure 2.5 Schematic diagram of a fibre-optic sensor.

Figure 2.6 illustrates types of fibre-optic sensors now being developed. Of these, first system which uses optical sensors element combined with fibre-optic signal transmission line, is the least developed and it is on this type of system that research is being concentrated. The second type of system is reflective optical sensor which senses changes in a beam of light

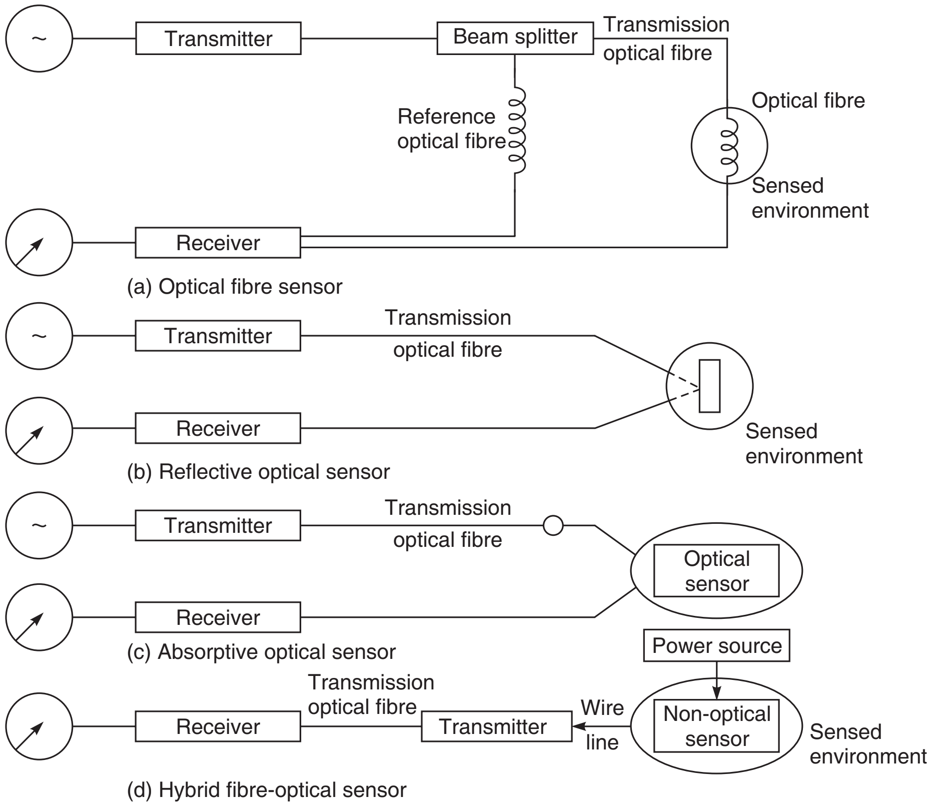


Figure 2.6 Typical fibre-optic sensors.

between fibre ends at a gap in the transmission line. The third system is a discrete optical sensor with a fibre-optic transmission line. This is also known as absorptive optical sensor. The final type is a discrete non-optical sensor element, with the fibre-optic transmission line. It is also known as hybrid fibre-optic sensor.

2.7 DISPLACEMENT/MOTION TRANSDUCERS

The displacement and motion of a body can be measured in many ways. The transducer may work on the principle of potentiometer, inductor, the reluctance, capacitor and even piezoresistance.

2.7.1 Linear Variable Differential Transformer (LVDT)

LVDT transducer is primarily used to measure the linear displacement. It comprises a transformer with one primary and two secondary coils with a movable core between them. The schematic diagram of LVDT is shown in Fig. 2.7.

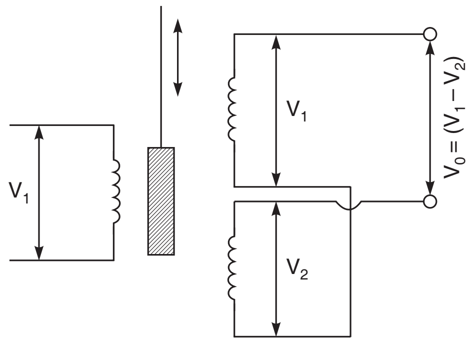


Figure 2.7 A schematic diagram of LVDT.

The secondary coils are identical and positioned symmetrically with respect to the primary coil. These coils are connected in opposite phase. The movable core is connected to the member whose displacement is to be measured. When the core is at centre, the voltages V_1 and V_2 in both the secondary coils are equal. Thus output voltage V_0 is zero, when displacement is zero. When the core is displaced, then V_1 and V_2 are different due to asymmetry of the core location with respect to secondary coils. This causes a finite non-zero output V_0.

The .output voltage V_0 is linear with displacement over a wide range. It undergoes a phase shift of 180 degree when core passes through zero displacement position, as shown in Fig. 2.8. Residual voltage V_r at zero displacement is normally 1% or Less of maximum linear voltage. This is caused by stray magnetic and capacitive effects. Residual voltage can be reduced by providing adequate grounding or by balancing the output circuit with resistive and capacitive shunt. Low pass filter at input will also help in the reduction of residual voltage. Figure 2.9 shows a cross-section of an LVDT. Commercial LVDT transducers have a range from 0.001 to several inches. However detection of microinches displacement can also be made by certain special types of LVDTs.

2.7.2 Capacitance Gauges

A simple capacitance gauge for displacement measurement consists of a pair of equal area parallel plates. One of the plates is movable while the other is fixed. The movable plate is connected to the member whose displacement is to be measured. The capacitance C between the plates is given by:

$$C = K \cdot E \cdot A/X$$

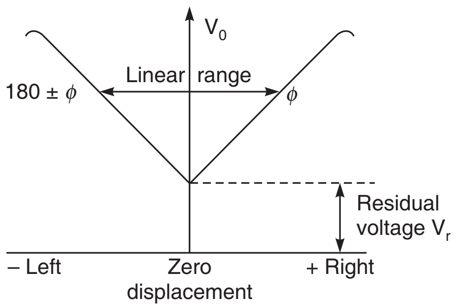


Figure 2.8 Illustration of LVDT undergoing a phase shift of 180 degree when core passes through zero displacement position.

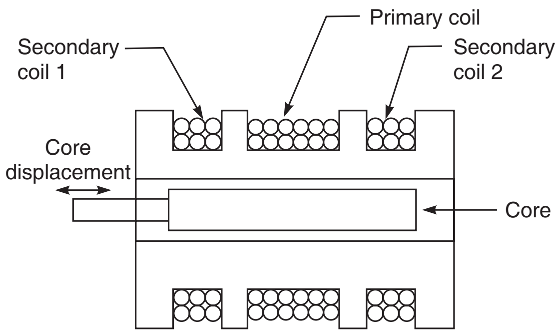


Figure 2.9 A sectional view of a linear variable-differential transformer (LVDT) (Gulton).

where

A = area of plate;
K = constant (the value being 0.0885 when dimensions are expressed in centimetres and 0.225 when dimensions are expressed in inches);
E = dielectric constant;
X = distance between plates.

The displacement in the movable plate changes the distance between the plates thus changing the capacitance. The change in capacitance (and thus displacement) can be measured by connecting an AC voltage source and measuring the change in voltage.

However, normally differential capacitor is used for displacement measurement. Figure 2.10 shows a schematic of differential capacitance gauge.

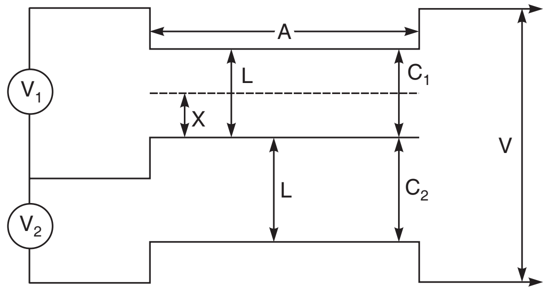


Figure 2.10 A schematic of differential capacitance gauge.

In normal position when displacement is zero, the capacitances C_1 and C_2 are equal. The middle plate is movable and the distance of middle plate from the two other plates is equal, when displacement is zero.

$$C_1 = C_2 = K \cdot \mathrm{E} \cdot A/L$$

The displacement input changes the position of centre plate with respect to other plates. For a displacement "X" of centre plate towards plate 1.

$$C_1 = K \cdot E \cdot A/(L - X),\ C_2 = K \cdot E \cdot A/(L + X),$$

$$V_1 = VC_2/(C_1 + C_2) = V(L - X)/2L,\ V_2 = VC_1/(C_1 + C_2) = V(L + X)/2L$$

$$\text{Difference voltage } V_0 = V_1 - V_2 = VX/L$$

Thus, V_0, the difference voltage is linear function of displacement X. The outputs V_1 and V_2 of the two capacitors are fed into a differential measuring circuit to obtain difference voltage.

2.7.3 Silicon Displacement Transducers

Many silicon pressure transducers discussed later can be used to measure the displacement also. KEZ10 silicon pressure transducer from Philips has been used to measure extremely small displacements (20 to 50 µm). Philips also offers KMZ10 silicon position sensors working on magnetoresistive principle. It can be used to measure both linear as well as angular displacements, from a few millimetres up to tens of centimetres with resolution up to 1 micrometre (µm). Hall effect silicon transducers have also been used for this purpose.

2.7.4 Fibre-Optic Displacement Transducers

Mechanical Technology Inc., USA introduced an optical displacement transducer in 1967. It uses light reflection technique which is still in use for vibration analysis and some other applications. A scheme for the measurement of displacement is shown in Fig. 2.11. This cantilever type fibre optic displacement sensor has been developed at Sanders Association Inc. for acoustic hydrophone applications. A multimode fibre is mounted as a mass loaded cantilever bar. There is an output fibre to form an optical transmission path by aligning itself to cantilever fibre. The cantilever fibre is connected to the subject. Any lateral displacement in cantilever fibre will result in misalignment between two fibre end surfaces. This will change the intensity of light in the output fibre.

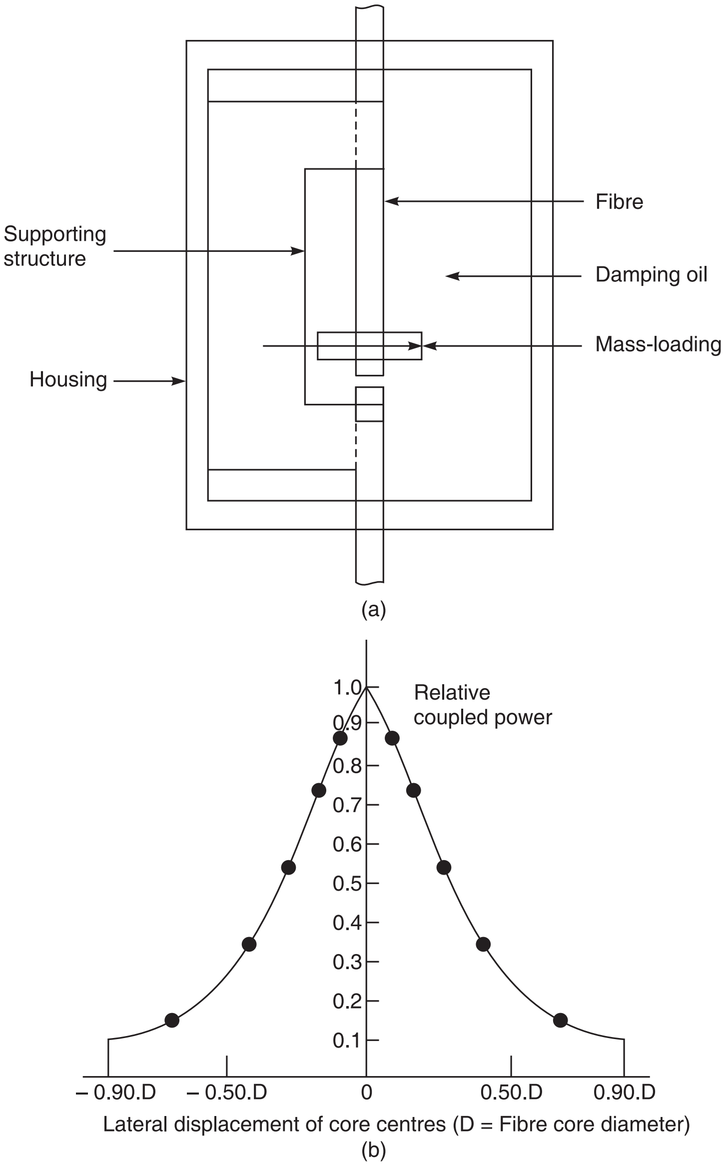


Figure 2.11 A fibre-optic Linear accelerometer: (a) Device configuration, and (b) Response curve a function of lateral displacement.

General Electric Research laboratory has developed optical transmission grating for measurement of displacement. Two optical fibres (Fig. 2.12) are coupled through two optical transmission gratings, one of which is stationary and other is movable. The movable grating is connected to the subject. The movable gratings act like a shutter between the two fibres. The amount of light that can pass from input to output fibre is linearly proportional to lateral displacement between two gratings.

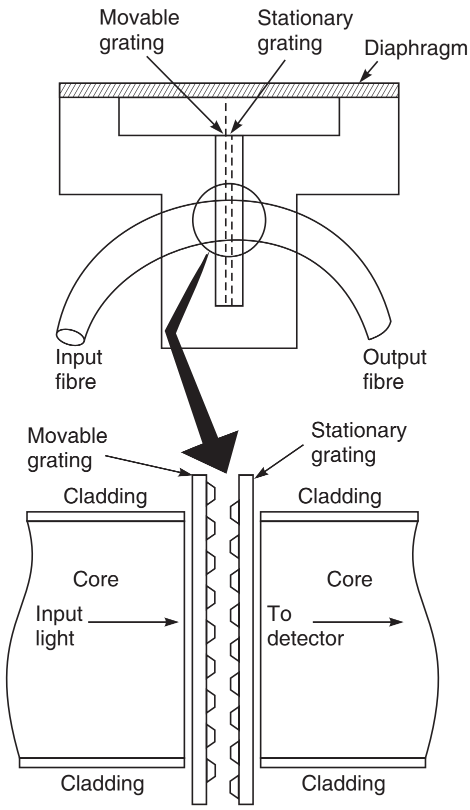
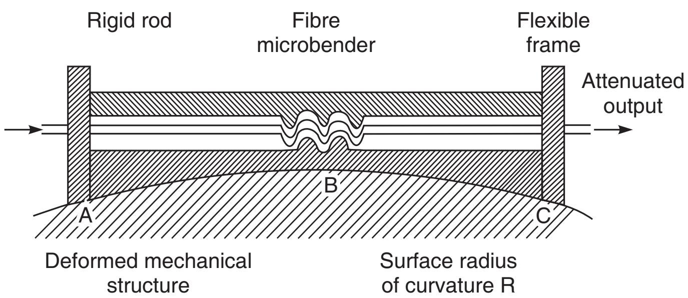


Figure 2.12 Moving grating hydrophone.

Displacement sensors based on twisting or bending of optical fibre bas been developed at TRW (USA). When an optical fibre is bent, the transmission loss increases since the optical modes are coupled out from the core region into the cladding region of the fibre. An optical fibre is placed between two corrugated plates as shown in Fig. 2.13. The loss during light transmission is measured and the displacement is derived from that.



(a)

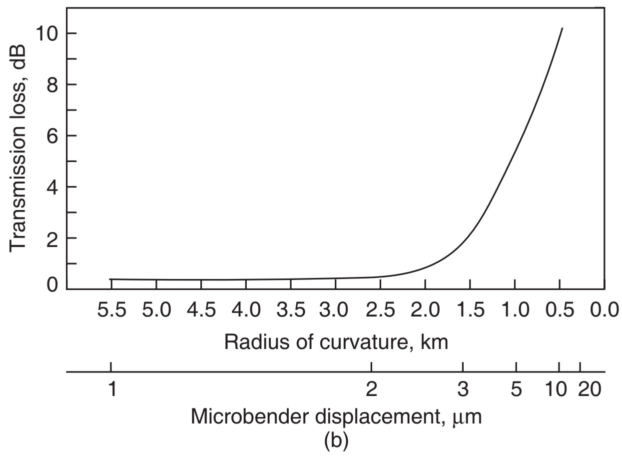


Figure 2.13 (a) A microbending fibre-optical sensor, and (b) Response curve of sensor.

A commercially available device for displacement measurement using optical fibres is based on light collection in a broken optical fibre path (Fig. 2.14). The light launched from input fibre is reflected by the surface and collected by the output fibre. The amount of light collected depends on the gap between the fibre end surface and the reflecting surface. The change in amount of light collected in the output fibre is detected, in order to measure the displacement of reflecting surface which is connected to the subject.

2.8 TEMPERATURE TRANSDUCERS

The temperature measurement involves the expansion properties of solid, liquid or gases or changes in electrical properties of certain materials. For very high temperature, the colour change is taken as a interim for measurement. The temperature is thus dependant parameter since material property is involved in its measurement.

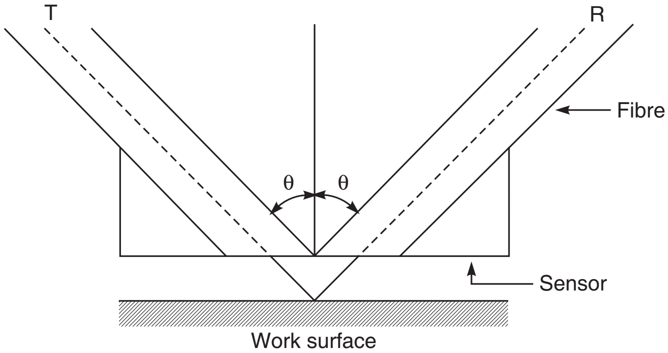


Figure 2.14 Fibre-optical displacement sensor.

The bimetal thermometer, filled in thermometric system and vapour pressure thermometer are examples of temperature transducers which involve the expansion properties of solid, liquid and gas respectively. The examples of electrical temperature transducers are resistance thermometers, thermocouples, thermopiles and so on etc. The optical methods are used for very high temperature transducers.

2.8.1 Resistance Temperature Detector

The Resistance Temperature Detector (RTD) is based on the principle that the resistance of a metal changes with temperature. Generally platinum and nickel are used as metal in RTD's. The relation between temperature change and metal resistance is given by

$$RT = R_0\ (1 + \alpha \Delta T)$$

where

R_0 = original resistance,
RT = resistance when a temperature difference ΔT is applied,
α = temperature coefficient of the metal.

Generally platinum (–190 to 660 °C) and nickel (0 to 325 °C) are used as metal in RTDs.

The resistance wire diameter may vary from 0.002 to 0.06 cm depending on the range. The wire is tested for purity and wound on a framework to form a coil. Different frameworks are chosen for different kinds of applications (Fig. 2.15). For instance, a mica-cross is used for general purpose application, whereas for surface temperature measurement a strain gauge type mesh may be used. In case of measurement of temperature of flowing liquid/air as self-supporting helical form is more useful.

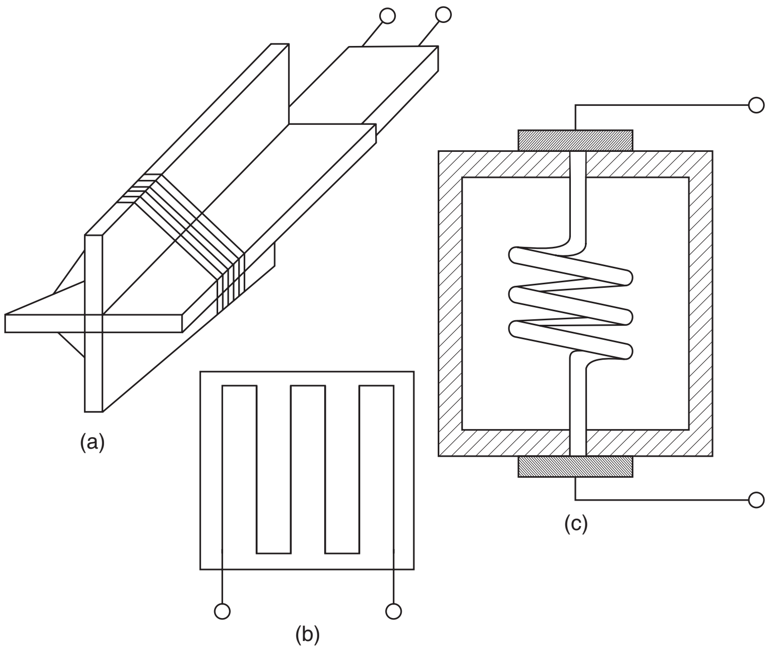


Figure 2.15 Types of resistance elements for measuring temperature: (a) Mica-cross winding type, (b) Strain gauge (mesh) type, and (c) Self-supporting helical type.

A variety of materials are used m the production of resistance temperature detector. Common among these are nickel, nickel-iron alloy, copper and platinum. Due to extra stability, platinum is popular, and 100 ohms platinum sensor is perhaps the most widely used RTD. Thin film RTDs are recognised for their long term stability.

The resistance temperature detectors are most accurate of all temperature transducers. An accuracy of 0.0001 degree centigrade can be achieved by these transducers and these are very convenient for small temperature differences. However, they have inherent drawbacks of heat loss, i.e., self-heating error (= I^2RT) thermo e.m.f and requirement of separate power pack.

2.8.2 Thermocouples

Thermocouples are the most important temperature transducers in the industry applications. They work on the Seebeck's principle of thermo e.m.f that when two dissimilar metals are joined as shown in Fig. 2.16 with the two junctions J_1 and J_2 at temperatures t_1 and t_2 respectively, then an e.m.f. is generated, causing a current to flow in the circuit.

The relation between the output voltage (in millivolt) and junction temperature has been found empirically as,

$$E = \sum_{n=1}^{k} \frac{1}{n} \cdot \alpha \cdot t^n$$

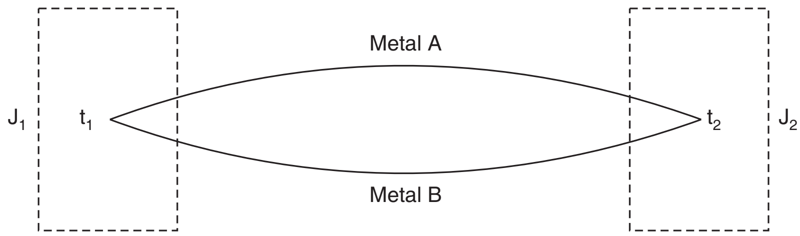


Figure 2.16 Seebeck's principle of thermo e.m.f.

The term t is the hot junction temperature when the cold junction is maintained at 0°C. In practice $K = 3$ has been found sufficient. The constant α depends on the materials of thermoelements (metals). For appropriate measurement, the linear relation between E (output voltage) and t (hot junction temperature) will be required with high sensitivity.

Following are the most commonly used industrial thermocouples:

S. No.	*Type*	*Combination*	*Temperature Range*
1.	T	Copper-Constantan	– 180 to 370°C
2.	J	Iron-Constantan	0 to 760°C
3.	K	Chromel-Alumel	0 to 1260°C
4.	R	Platinum- Platinum	0 to 1480°C
	S10	Rhodium	

Type J thermocouples are useful in the environment where there is a lack of free oxygen. Unprotected type J thermocouples may be used up to 290°C in reducing atmosphere. Heavier wire and protection wells should be used for 540 to 870°C temperature measurement. Type K on the other hand is suitable for oxidizing atmosphere where excess of free oxygen is present. Types R and S10 thermocouples are called noble metal thermocouples and are used for higher temperature ranges. These should be protected by impervious tubes when used at temperature above 540°C.

The main problem encountered with thermocouples is the low signal level. The electrical noise on the signal lines is quite higher than the temperature signal. Thus measuring device should have a high input impedance and very high common mode noise rejection.

Industrial thermocouples have only one junction, the hot junction which is connected to the process. The cold junction or reference junction is at ambient temperature. This requires compensation. The assumption that the cold junction is at constant temperature does not hold in the industrial environment. This also requires compensation and is called *cold junction compensation*.

A practical approach for cold junction compensation is to measure the temperature of cold junction by an RTD or thermistor and correct the thermocouple indicated temperature. This can be done by determining and adding a correction temperature to the instrument or by modifying the thermocouple signal directly.

Thermocouple tables are provided by the manufacturers. The tables show the voltages obtained at various temperatures. Table 2.2 shows a part of table for type R thermocouple referenced to cold junction at 0°C. We shall illustrate the cold junction compensation using this table.

Table 2.2 Extract from Thermocouple Tables for Type R (platinum-platinum/l3% rhodium) (Referenced to cold junction at 0°C. Voltage in μV.)

	Temperature (°C)									
	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
0	0	5	11	16	21	27	32	38	43	49
10	54	60	65	71	77	82	88	94	100	105
20	111	117	123	129	135	141	147	152	158	165
800	7949	7961	7973	7986	7998	8010	8023	8035	8047	8060
810	8072	8085	8097	8109	8122	8134	8146	8159	8171	8184
820	8196	8208	8221	8233	82.46	8258	8271	8283	8295	8308
830	8320	8333	8345	8358	8370	8383	8395	8408	8420	8433
840	8445	8458	8470	8483	8495	8508	8520	8533	8545	8558
850	8570	8583	8595	8608	8621	8633	8646	8658	8671	8683
930	9589	9602	9614	9627	9640	9653	9666	9679	9692	9705
940	9718	9731	9744	9757	9770	9783	9796	9809	9822	9835
950	9848	9861	9874	9887	9900	9913	9926	9939	9952	9965
960	9978	9991	10004	10017	10030	10043	10056	10069	10082	10095

1. Let us assume that ambient temperature T_2 measured by thermistor is 25°C and thermocouple voltage is 8.46 millivolt (mV).
2. The table shows thermocouple voltage of 0.141 millivolt (mV) for 25°C temperature.
3. Corrected voltage = Thermocouple voltage + voltage corresponding to ambient temperature.
$$= 8.46 + 0.141 = 8.601 \text{ mV}.$$
4. From table voltage 8.595 mV corresponds to 852°C and voltage 8.608 mV corresponds to 853°C. The actual temperature can be found by interpolation.
   1°C difference = 8.608 – 8.595 = 0.013 mV.

Thus (8.601 – 8.595) = 0.006 mV and temperature correction $= \dfrac{0.006}{0.013}$

$$= 0.46°\text{C}$$

Thus corrected temperature = 852°C + 0.46°C

$$= 852.46°\text{C}$$

Figure 2.17 sliows the basic thermocouple circuit in temperature measurement. The load generally consists of signal conditioning circuit. The temperature difference between sensing junction and reference junction causes a current to flow through the load.

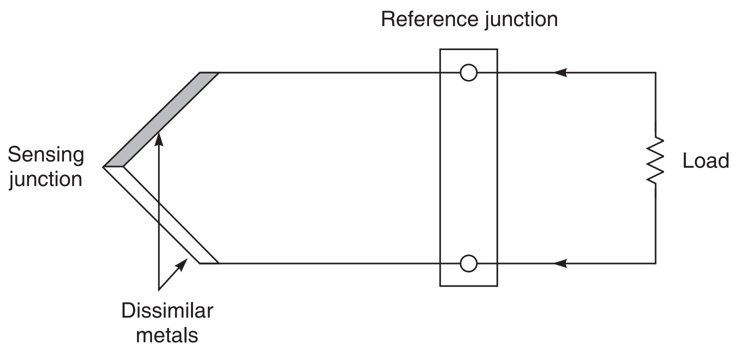


Figure 2.17 Basic thermocouples circuit in temperature measurement.

2.8.3 Thermistors

Thermistors are semiconducting resistance temperature transducers, with large coefficient of resistance. The negative thermistors are more common in the industry than positive thermistors (limited to 50 to 225°C range). The relation of a negative temperature resistance coefficient is shown in Fig. 2.18. The non-linear scale over the entire range of operation may be made linear by applying various compensation schemes. A single low resistance in parallel to the thermistor will reduce the sensitivity but increase the linearity, as shown in Fig. 2.19.

Oxides and sulphides of copper, cobalt, manganese etc. are used in the manufacturing of thermistors. The bead, rod and disc type thermistors are more common (Fig. 2.20). The thermistor range for common purpose is specified as from 100 to 300°C. However, special thermistors made of aluminum oxide cover a high temperature range from 800 to 1000°C.

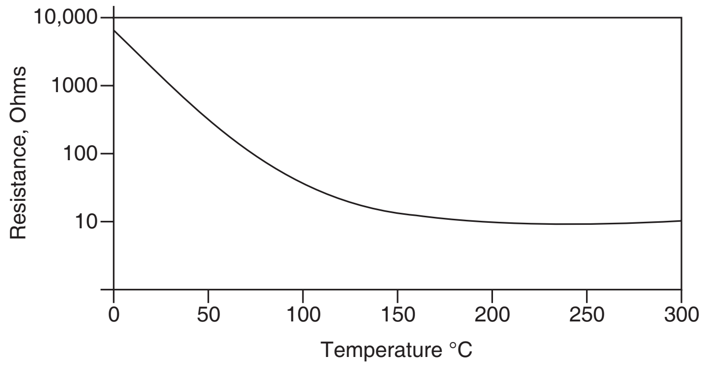


Figure 2.18 The non-linear relation of a negative temperature resistance coefficient.

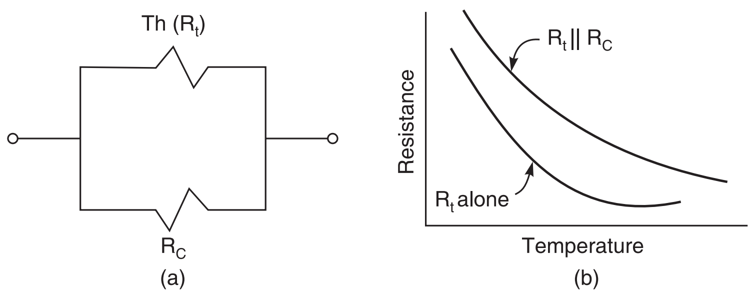


Figure 2.19 Compensation for non-linearity of thermistor response: (a) Scheme of paralleling with a low resistance, and (b) Response curves of uncompensated and compensated thermistor resistances.

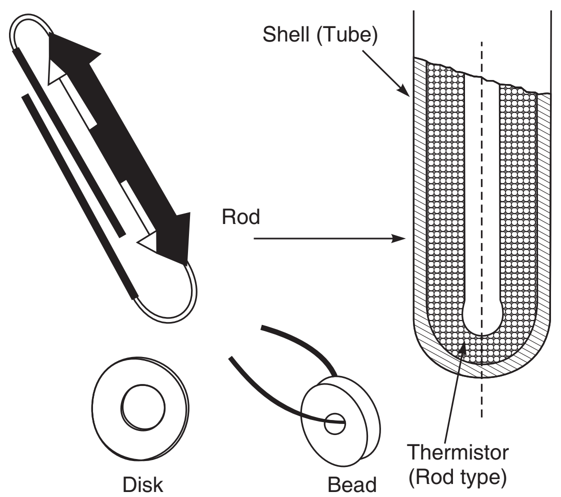


Figure 2.20 The bead, rod, and disk type thermistors.

2.8.4 Pyrometers

Pyrometers are used when the measurement transducers cannot be put into contact with the process. It may be due to very high temperature (as in blast furnace) or hot mobile bodies (as in case of rolling mills). The basic principle of measurement through pyrometers is to measure directly or by colour comparison the energy radiated by the hot body. In case the radiated energy is measured directly, it is called radiation pyrometery and when the energy is measured by colour comparison, it is called optical pyrometery.

Radiation Pyrometers

Figure 2.21 shows the principle of radiation pyrometery. The radiations emitted by the hot body are focused on a thermal detector element using a focusing lens. The voltage produced by the detector follows the empirical relation given below

$$V = K_v T^n$$

where

$3.5 < n < 4.5$
V = output voltage,
T = temperature of hot body,
K_v = constant.

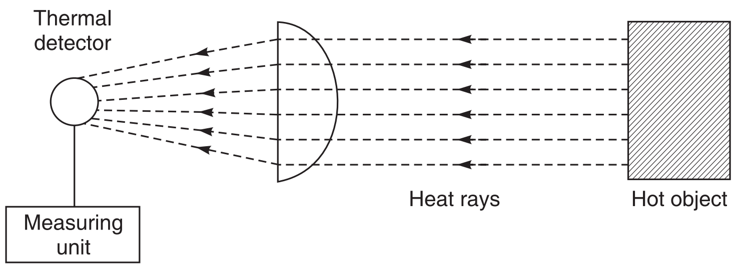


Figure 2.21 Illustration of principle of radiation pyrometery.

The value of K_v is obtained from experimental calibration. Commonly used thermal detectors are thermopiles, photocells, thermistors etc.

Optical Pyrometers

The basic principle of optical pyrometers is shown in Fig. 2.22. The radiation received from the hot object are focused on a filament by means of a lens. The filament is viewed by a microscopic system. The image of the target is formed at the filament position and the filament is viewed in a direction against the background of the target surface. The brightness of filament is adjusted till it disappears from the field of view. The current flowing through the filament may be measured. In fact the ammeter connected in series with the filament may be directly calibrated to give temperature of target.

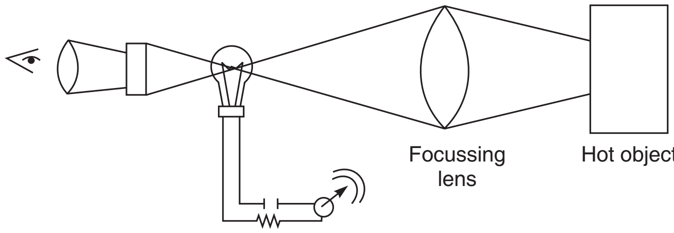


Figure 2.22 Basic principle of optical pyrometers.

Optical pyrometers are usable in the range between 700 to 3000°C and are more accurate than radiation pyrometers. The lower temperature range is limited due to limitation of human eye to compare the radiation sources. However, automatic optical pyrometers are also available for measurement and control. Figure 2.23 shows an operational diagram of automatic optical pyrometer (*Courtesy*: Leeds and Northrup Co).

The multiplier phototube collects the radiations from target and standard lamp alternately, through rotating light modulator disc. If the brightness of the two sources is not same then a signal is fed to the lamp through preamplifier, demodulator and integrator. The integrator drives the standard lamp circuit, so that the brightness of lamp equals that of hot object. An automatic gain control is provided to prevent the measuring system becoming over sensitive at higher temperatures. Automatic gain control varies the demodulator gauges width i.e. amount of signal passed through the system. Millivolt recorder can be calibrated to display the temperature. The lamp current could be used in other subsequent circuits as measure of temperature.

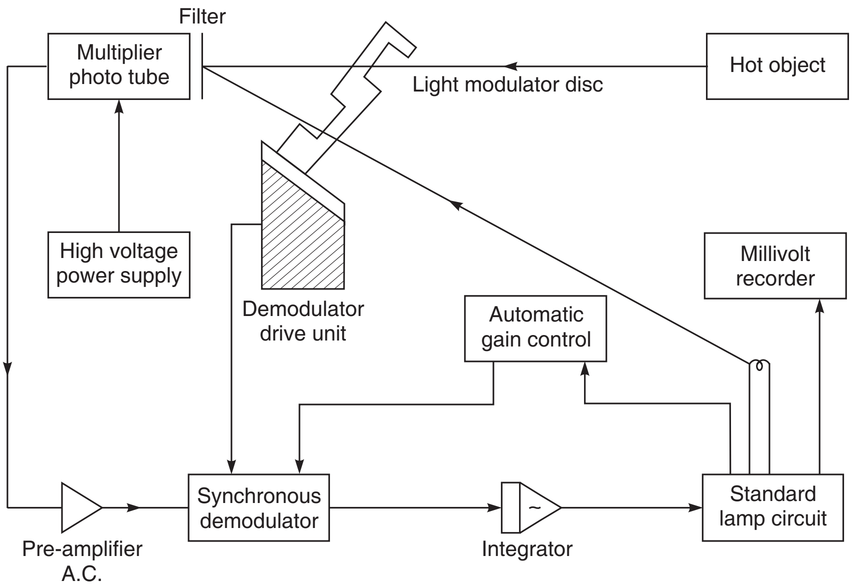


Figure 2.23 Operational diagram of automatic optical pyrometer.

2.8.5 Silicon Temperature Transducers

Over the past few years, the increasing use of integrated circuits in control systems has stimulated a demand for electronics temperature sensor that is both accurate and reliable. Silicon temperature sensors are well placed to meet this demand. These sensors make use of temperature dependence of resistivity exhibited by silicon. Figure 2.24 shows this dependence for *n* and *p* type silicon at several doping levels *N*. The initial rise in resistivity is caused by the fall in free charge carrier mobility with rising temperature, and over this region, silicon exhibits a positive temperature coefficient of resistance. At higher temperature, when intrinsic semiconductor properties of silicon predominate, the mobility increases and resistivity falls.

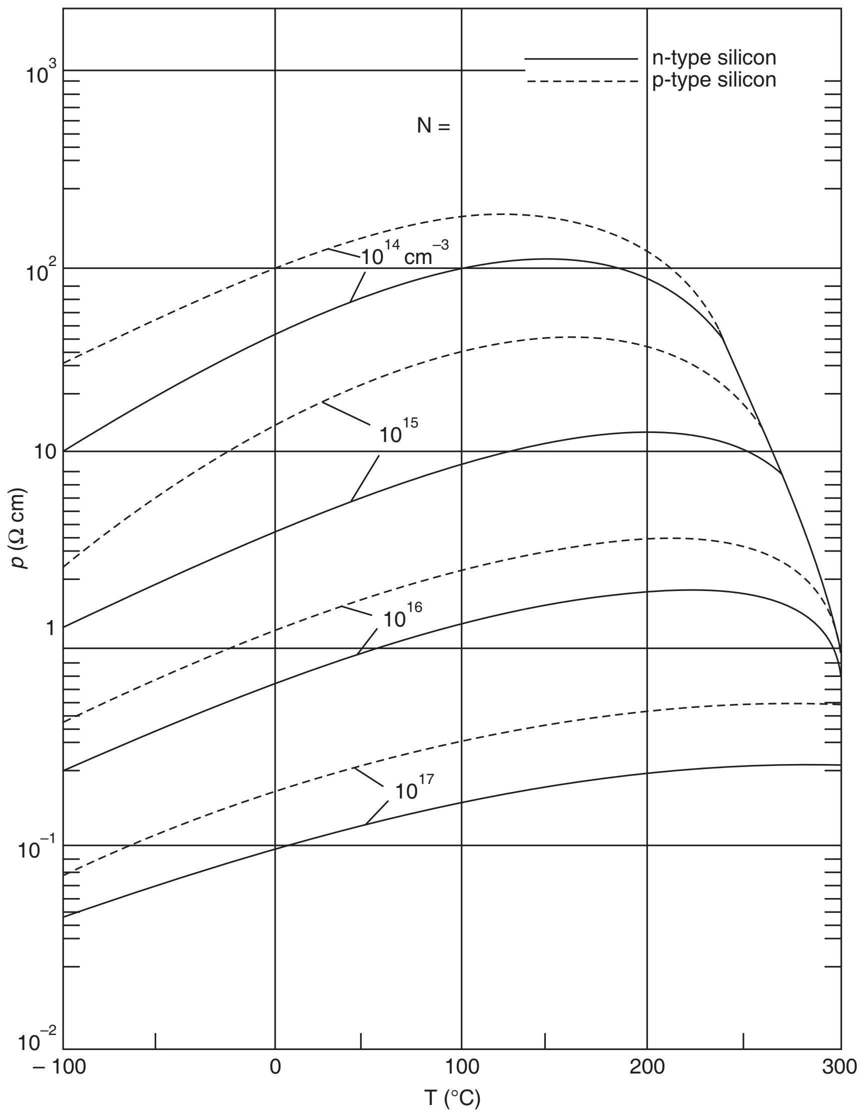


Figure 2.24 Temperature vs. resistivity response curve for *p*-and *n*-type silicon.

Silicon diode temperature sensors have been developed by many semiconductor companies. In these devices, the base emitter voltage of a forward biased diode changes with temperature. These are low cost devices offering high accuracy, stability and linearity, but suffer from the disadvantages of low output and limited upper temperature range. These devices do not make use of properties of silicon as such.

Analog devices offers two devices AD590 and AD592 series as silicon temperature transducers. AD590 offers linear current and output of 1 microampere (μa) per degree kelvin. It could be used in any temperature-sensing application below +150°C, in which conventional electrical temperature sensors are currently employed. In addition to temperature measurement, these sensors can be used in variety of other applications, like flow rate measurement, level detection of fluids, biasing proportional to absolute temperature etc. It is available in five models I, J, K, L and M and each model is available in TO 52 can or flat package. The AD590 is particularly useful in remote sensing applications. It is insensitive to voltage drops over long lines due to high impedance current output. Any well insulated twisted pair cable is sufficient for operation hundreds of feet from the receiving circuitry.

The AD592 is available in three performance grades AD592AN. AD592BN and AD592 CN. Al1 the devices are packaged in a plastic TO 92 case. The device operating temperature range is from 25 to +105°C. It offers excellent linearity and output as micro amp. per degree K. Typical application areas include; appliance temperature sensing, industrial temperature control, automatic temperature measurement and control etc. Like AD 590, it is also well suited for remote sensing applications.

National Semiconductor has introduced LM 34, LM 35, LM 134, LM 135 and LM 3911 series of silicon temperature sensors, for different range and accuracy. LM 34 series is calibrated directly in degrees F, with temperature range of –45 to +150°C. Other series are calibrated in degree centigrade with temperature range of –55 to +150°C. These are available in TO 92 or TO 46 package and do not require any external calibration. LM 3911 is basically a temperature controller (range, –25 to +85°C). It includes a temperature sensor, a stable voltage reference and an operational amplifier (op. amp.) on the same chip. The op. amp. can be used as comparator to switch on the output as temperature sensed crosses the set-point. This makes the device suitable for on-off temperature control.

Philips has developed KTY81/83/84 series of planar silicon temperature sensors. The devices are manufactured, using the reliable planar technique. A layer of silicon nitride is used to protect the crystal surface and to provide additional protection, the entire crystal is coated with phosphor glass. Where KTY 81 and 83 can be used to measure temperature from 55 to 150°C, KTY 84 offers temperature range from 65 to 300°C. KTY 81 is available in SOD 70 type encapsulation while KTY 83 and 84 are available in SOD 58 type encapsulation. These devices can be operated with a constant voltage source or constant current source and linearization can be easily achieved in both cases.

Kulite has developed a unique sensor ITQ 1000, which offers integrated temperature and pressure measurement. The temperature within range 25 to +125°C and pressure within range

20 to 500 psi can be measured using this sensor. Its STQ and STH series of diffused silicon temperature sensors offer a wide operating range of –46 to 177°C.

Motorola offers MTS 102, 103 and 105 series of silicon sensors in TO 92 cases for –40 to +150°C. Intersil had similarly announced ICL 8073 and ICL 8074 temperature transducers in TO 52 cans over operating range –55 to +125°C.

2.8.6 Fibre-Optic Temperature Transducers

Displacement sensors can be made to measure the temperature by using thermoexpansion or temperature-sensitive bimetal. The bimetal displacement caused by the temperature, generates misalignment between a pair of fibre lens connector assembly. This varies the amount of light arriving at the output end which is measured.

Temperature-sensitive transparent substances like liquid crystals and semiconductors are proposed to be used by Rockwell International Corporation as attenuator for temperature measurement, using optical fibres. Variation in temperature results, in colour changes in liquid crystals and band gap shifts in semiconductors. Thus, the amount of attenuation will depend on the temperature.

Rockwell International Corporation, USA, bas also proposed a bi-fringent digital sensor which uses crystal polarization solution with temperature. Another approach being investigated is the use of a miniature optical Fabryperot resonator cavity at the tip of an optical fibre. The change in temperature causes variation in cavity length, resulting in changes in the fibre-end surface reflectivity which can he used to measure the temperature. Holographic processing of light information is also being developed at Rockwell International corporation.

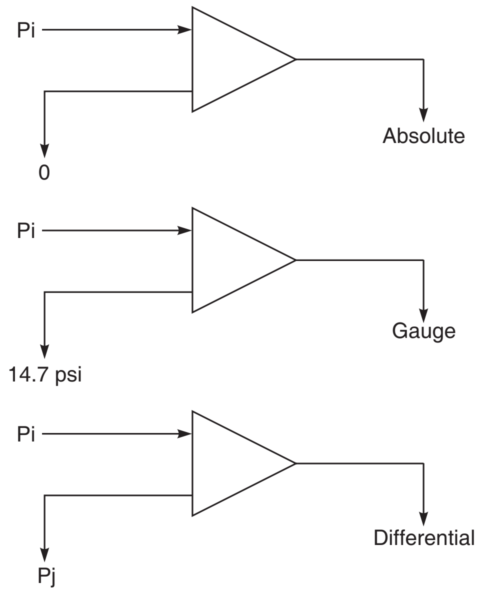
TRW (Technology Research Centre), USA, has developed a spectral temperature sensor. It is based on black body radiation of the heated fibre tip. A photodetector is used remotely to measure the optical power and thus temperature.

2.9 PRESSURE TRANSDUCERS

Pressure is very important parameter for any process. In various units of a plant, very high or very low pressure both may cause problems and may lead to partial or complete failure in operation. Let us discuss some nomenclatures before going into actual measuring systems.

Absolute pressure is the force exerted by the fluid per unit area of the wall of the container. It is represented in psia (pound per square inch absolute) in FPS unit. *Gauge pressure* is the difference between the absolute and the local atmospheric pressure (14.7 psi). It is normally expressed in psig (pound per square inch gauge) in FPS unit. When gauge pressure is negative, it is called vacuum. *Differential pressure* is a measure of pressure difference between both sides of transducer thin structure with both sides exposed to a different pressure level. It is expressed in psid (pound per square inch differential in FPS unit).

Symbolically these may be represented as follows:



The mechanical type pressure transducers include Bourdon tubes, Bellows, and Diaphragm gauges. Where bellows and diaphragm gauges are suitable up to about 4000 to 8000 psi, the Bourdon tubes are useful for high ranges.

The electrical methods of pressure measurement can be divided into two categories: *primary* and *secondary*. The secondary methods use a capacitive or inductive sensor (capacitance gauge, LVDT etc.), to convert displacement caused by bourdon tube, diaphragm etc. The primary pressure transducers use electrical methods and include piezoresistive (Strain gauges), piezolelectric (quartz crystal) etc. The capacitive and inductive type secondary sensors have already been described under displacement transducers. We shall discuss here piezoelectric, piezoresistive, silicon and fibre-optic pressure transducers.

2.9.1 Piezoelectric Transducers

A class of solid polycrystalline dielectric materials when deformed by the application of force generate electric charges and vice versa. This is known as *piezoelectric effect.*

The charge produced due to the deformation by the application of pressure can be measured by a pair of electrodes mounted suitably. Natural crystals like quartz, Rochelle salt and synthetic materials like lithium sulphate, ammonia dihydrogen phosphate etc. exhibit the piezoelectric phenomenon.

The piezoelectric transducer is cut from a larger crystal in the direction of any of the electrical or mechanical axis, perpendicular to optical or crystal axis. The electrical axis (3 sets) is known as *X* axis and the mechanical axis as *Y* axis (Fig. 2.25).

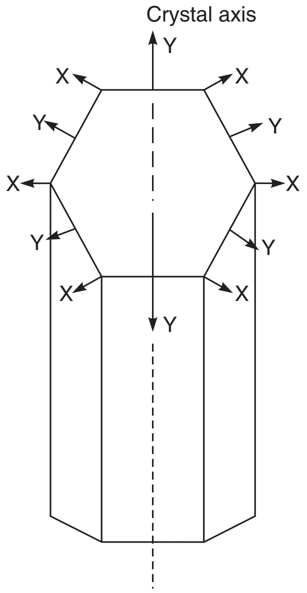


Figure 2.25 A piezoelectric transducer cut from a larger crystal with electrical axis (X) and mechanical axis (Y).

The pressure measurement using a piezoelectric crystal is shown in Fig. 2.26. Let, F be the force applied in the direction of Z, resulting in total charges Q and output voltage e then,

$$\text{Pressure } (p) = \text{force/area} = F/\text{area of crystal}$$

$$\text{Charge sensitivity } (d) = \text{charges per unit force generated} = Q/F$$

$$\text{Voltage sensitivity } (g) = \text{field produced per unit stress} = (e/t)/p$$

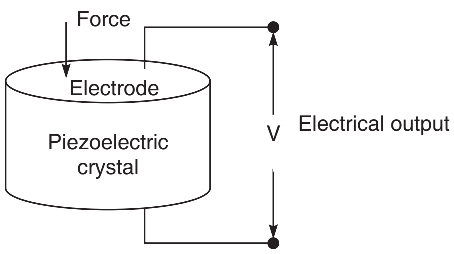


Figure 2.26 The pressure measurement using a piezoelectric crystal.

We, therefore deduce that output voltage $e = pgt$, where, t = thickness of transducers.

Thus, output voltage is linear function of pressure applied to the crystal.

The natural crystals are preferred as compared to synthetic crystals. This is because they show higher degree of thermal and mechanical stability, can sustain higher stress, have low leakage and exhibit good frequency response. Only barium titanate, out of synthetic crystals shows these characteristics.

Some piezoelectric transducers are designed as resonant types. In this case, the crystal is driven at resonance from a suitable oscillator and the voltage required to achieve this is measured. The application of pressure changes the resonance frequency and the voltage required to achieve the resonance. The change in voltage is measured to determine the stress.

2.9.2 Piezoresistive Transducers

When a wire is stretched within the elastic limit, it will have an increase in length and corresponding decrease in diameter. Thus resistance of wire changes due to the strain. This is called *piezoresistive effect*.

If ΔL be the increase in length and ΔR be the increase in resistance then,

$$\Delta R/R = K \cdot \Delta L/L = K \cdot \sigma/E$$

where

R = original resistance,
L = original length,
E = Young's modules of elasticity,
σ = stress = force/area

The value of K varies between 2 and 6 for metals and for semiconductor materials values up to and above 180 are obtained.

The strain measuring circuit consists of a bridge as shown in Fig. 2.27. One arm of the bridge contains the strain gauge, while other arms have standard resistor of equal resistances, as that of gauge resistance in the unstrained condition. The current through the bridge arm measures the strain.

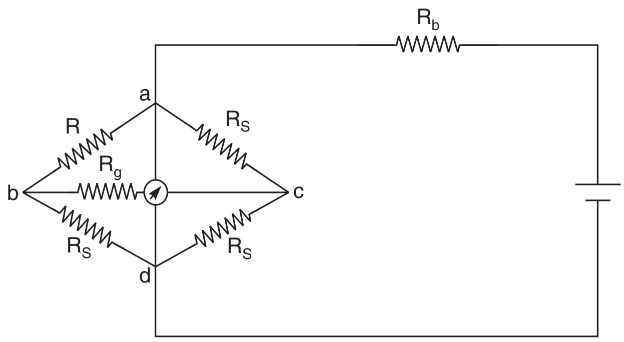


Figure 2.27 An illustration of a strain measuring circuit.

The construction of strain gauge is shown in Fig. 2.28. It consists of a measuring element embedded in a carrier material for better handling. Electrical connection is realized via external electrodes. According to the relevant manufacturing processes, four major strain gauge types are distinguished: Flat strain gauges, Wrapped around strain gauges, Foil strain gauges, and Semiconductor strain gauges (Fig. 2.29).

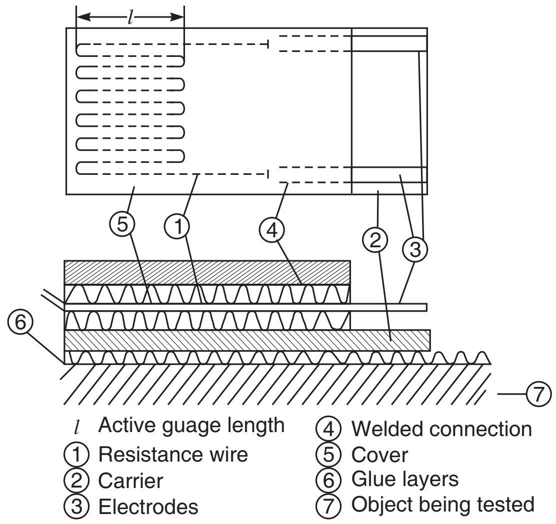


Figure 2.28 Construction of a strain gauge.

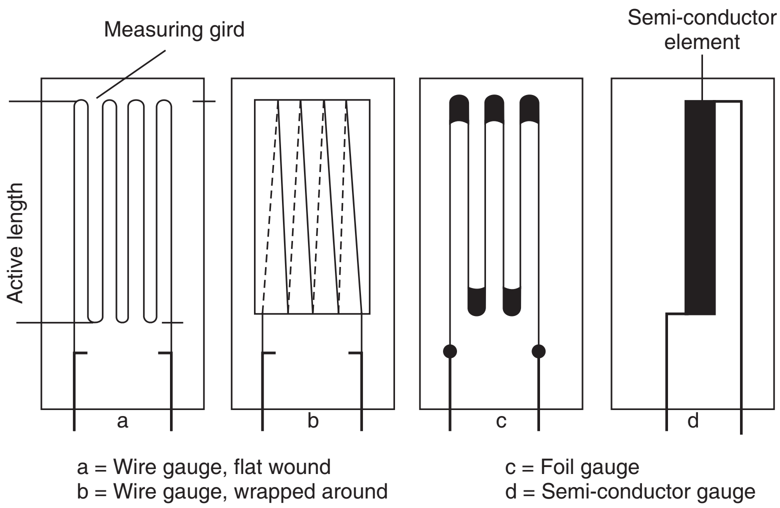


Figure 2.29 Types of strain gauges.

Wire strain gauges are made of constantan (60% Cu, 40% Ni alloy) wire of about 20 to 30 μm diameter, wound flat or around a piece of carrier material. The foil strain gauge consists of a rolled constantan foil of 2 to 10 μm thickness. The measuring grid is produced by etching, after application of carrier.

The types of strain gauges mentioned may be put in several arrangements. The most commonly used are linear gauges used for measuring material strain in a known direction. In case the main direction of strain is unknown, rosette gauges are used. These consist of several measuring grids arranged on a common carrier (Fig. 2.30).

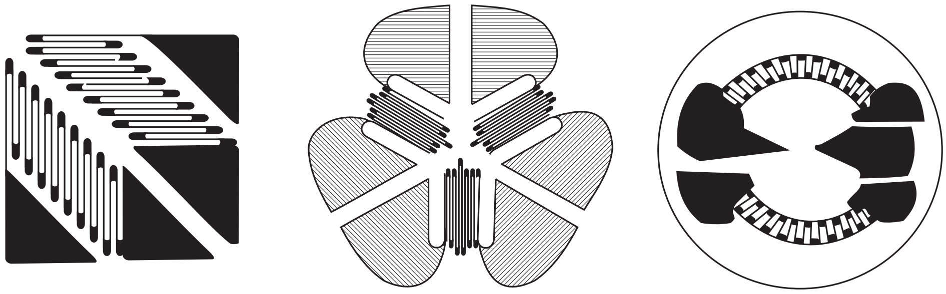


Figure 2.30 Various measuring grids arranged in a common carrier.

2.9.3 Bridgeman Resistive Transducer

When a wire is subjected to pressure from all sides, its electrical resistance changes due to distortion produced in the crystal lattice. In most common metal wires the pressure causes the decrease in resistance, where antimony, bismuth, lithium, maganin show the increase in resistance with pressure as shown in Fig. 2.31. In cesium it initially decreases for small values of pressure changes and reaches a minimum, beyond which it increases with increase of pressure. The relation can be approximated by equation,

$$R_p = R_0 (1 + \beta \Delta P)$$

where

β = pressure coefficient of resistance,
R_0 = resistance at standard atmospheric condition,
ΔP = pressure difference applied,
R_p = resistance when pressure is applied

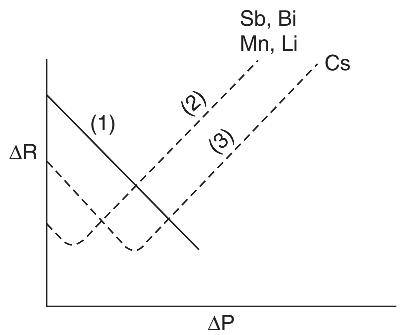


Figure 2.31 Pressure-resistance characteristics of Bridgeman transducers.

The value of β is quite large for alkyl metals, antimony and bismuth but these materials are not convenient for practical realization of Bridgeman gauge. The construction of Bridgeman gauge is shown in Fig. 2.32. It consists of a bone ring of about 1 cm diameter and 0.5 cm thickness, wound with a 38 gauge insulated maganin wire (β = +2.3 × 10^{-6} cm^2/kg) to have a total resistance of 100 ohms.

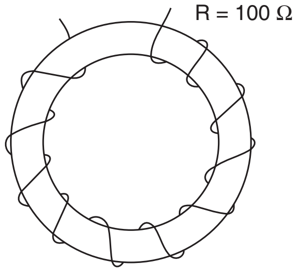


Figure 2.32 Construction of a Bridgeman gauge.

2.9.4 Silicon Pressure Transducers

Piezoresistive integrated sensor technology is today's emergent transducer technology. Silicon is very responsive to strain with K factors of –100 to +180. The silicon pressure transducers incorporate a diffused four-arm wheatstone bridge on the surface of silicon diaphragm. Of particular note is the high natural frequency, low hysteresis and super thermal and environmental performance. Though originally silicon was adopted because of its large piezoresistive coefficient and compatibility with transistor fabrication techniques, it has now been found to possess excellent, transducer characteristic as well.

Figure 2.33 shows a thin diaphragm piezoresistive pressure sensor using integrated circuit technology. The top view, cross-section and enlarged details are shown in Fig. 2.33(a), (b) and (c) respectively.

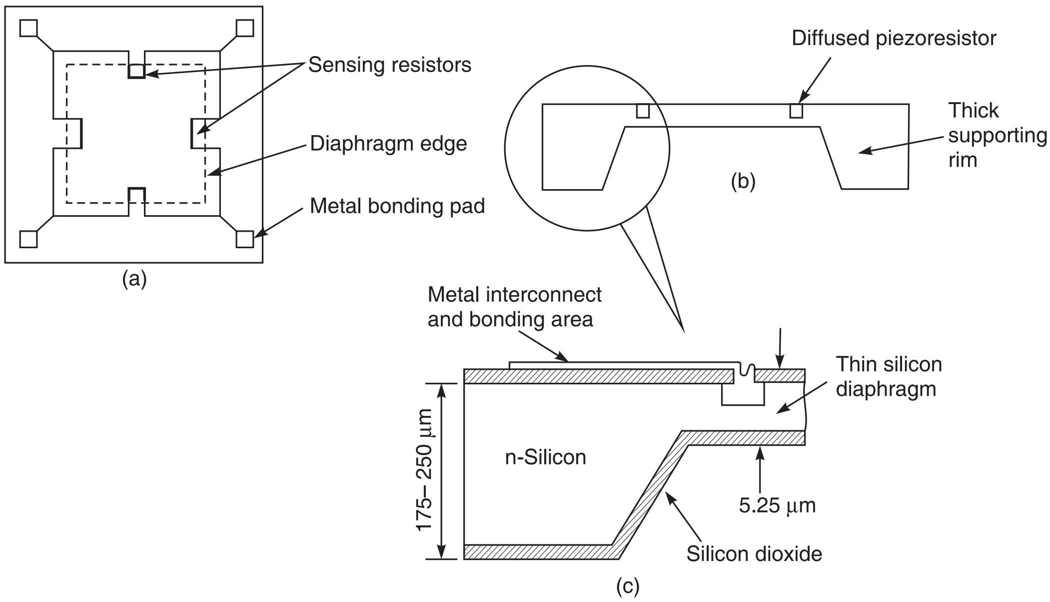


Figure 2.33 Silicon piezoresistive pressure sensor: (a) Top view, (b) Cross section, and (c) Enlarged detail.

Figure 2.34 shows the action of sensor chip when pressure is applied. Due to pressure, the resistance of diffused bridge resistors changes and output is generated.

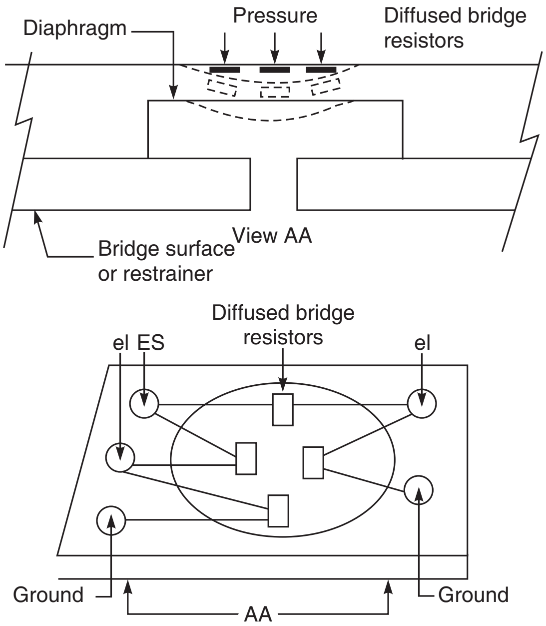


Figure 2.34 Illustration of sensor chip action.

IC Sensors USA has developed C series of Piezoresistive silicon sensors. The models 10, 20, 30, 40, 12, 22, 32, 42, 13, 23, 33, 43, 80, 90, 103, 104, 105, 113, 114, 115, 210, 300, 310, 410, 419, 1210/1220, 1219/1229 and 1419/1429 offer different ranges of pressure measurement from 0–2 psi to 0–5000 psi differential, gauge or absolute. Out of these, there are some models specially suited for use with corrosive or conductive fluid or gaseous media. In some models (12, 22, 32, 42) integral temperature compensation is provided, whereas in others the temperature compensation is accomplished with the addition of three resistors whose values are specified.

In 1959, Kulite Semiconductor Products Inc. began producing first bulk silicon and later diffused silicon pressure sensors for military and aerospace need. Kulite now produces a wide range of silicon pressure transducers. CQ–030, CQ–080 series transducer are perhaps the smallest transducers in existence today. The diaphragm diameter ranges from 0.030 inch to 0.080 inch. The measuring pressure ranges from 0–5 psi to 0–500 psi.

Kulite produces vanous series of pressure transducers like XT, XTE, XST, XCQ, XCS, HKS, ETM, LQ, IPT, VM, BM, PTQS, PTQH and ETQ. Each series has different applications and pressure range. As an example IPT750 and BM 5750 are flush metal diaphragm pressure transducers suitable for measurement of liquid level in tanks (range 0–5 psi to 0–1000 psi) used in food and brewing industries. IPT–1100, BM-1100 and BMEl100 series are standard metal diaphragm pressure transducers with integral inlet and port adapter. It can be used in a number of applications including hydraulic and pneumatic pressure measurement. Versions with integral amplifier with an output of 5 volt are also there. The measuring range varies from 0–5 psi to 0–10000 psi.

The Philips also manufactures pressure transducers, of various ranges. KPZ 10G and KPZ 11G use a thin film polycrystalline resistor layer mounted on one side of the metal membrane to translate direct pressures into changes of resistance. Reference pressures of –1 to +2 bar (KPZ 1OG) or –1 to +10 bar (KPZ 11G) can be measured. A 6-pin package encapsulates the composite metal and thin film membrane. There is provision for an IC to give temperature compensation, data processing and offset voltage trim. Since they provide direct contact on the metal side of the membrane, these sensors are suited for harsh environments which require direct contact with the fluid being measured, such as oil or hydraulic pressure monitoring in automobiles.

The KEZ 10 (0 to 10 Newton) and KEZ 11 (0 to 30 Newton) use a similar technology but measure strain. Main application will be in weighing equipment as well as other equipment requiring force measurements. These sensors can also he used to measure extremely small displacements (20 to 50 μm max).

KP 100A is another pressure sensor, manufactured by Philips. It measures absolute pressure up to 2 bar in clean-air or inert media environments. Using a monolithic silicon membrane with diffused resistors, the sensor incorporates its own temperature compensation circuit. The complete sensor is encapsulated in a 6-pin DIL package for direct PC board mounting. One side of the membrane is left to the atmosphere or other inert media which in turn imposes pressure on the membrane thus, causing a change in resistance of the diffused resistors.

Conrac Corporation System West division has designed a silicon on sapphire (SOS) transducer for pressure measurement. It is a high temperature, high accuracy sensor in which a silicon piezoresistive strain gauge is grown epitaxially on a sapphire diaphragm, as shown in Fig. 2.35. The series 4720 transducer can handle up to 1000 psi absolute pressure. The monolithic construction imparts a high degree of insensitivity to shock and vibration. Also strength to weight ratio of sapphire is twice that of steel. This also reduces the error due to vibration and shock. The design is aimed to achieve longer transducer life. However, the superior performance of the transducer adds to their cost.

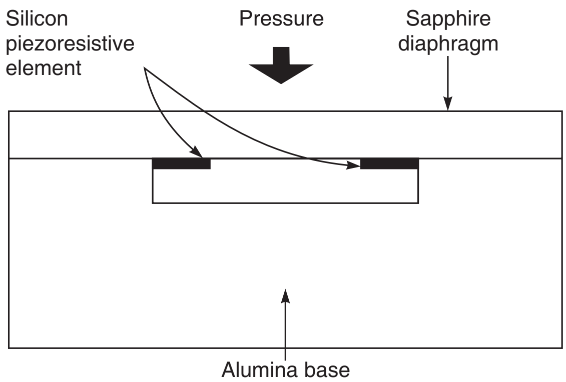


Figure 2.35 High performance pressure transducer using silicon on sapphire technology.

A pressure sensor from Nova Sensor, USA incorporates Silicon on Insulation (SOl) technology to operate in high temperatures. The initially offered pressure range for Novapix sensor is 0 to 100 psi. The sensor uses a proprietary wafer lamination technique to form piezoresistive elements integral to silicon diaphragm. The operating temperature range is 0 to 250°C.

Foxboro/ICT produces a silicon pressure transducer which ranges from TO.8 PCB mountable sensors to hazardous area rated pressure transmitters. The silicon sensor is protected by a steel diaphragm which is welded to the pressure connection for NEMA-4 sealing.

To reduce the cost of pressure transducers, Motorola has adopted a totally new approach for pressure measurement through piezoresistor method. Instead of opting for conventional Wheatstone bridge circuit, Motorola used a single P-type cross shaped silicon element, as shown in Fig. 2.36. Because of the shape the transducer is called X-ducer. A constant current passes through the longitudinal axis of elements thus generating a transverse voltage across the latitudinal axis. When pressure is applied this output voltage changes. The sensor is available in simple plastic package which further reduces its cost. A pressure range of 1 to 40 psi (differential and absolute) can he handled.

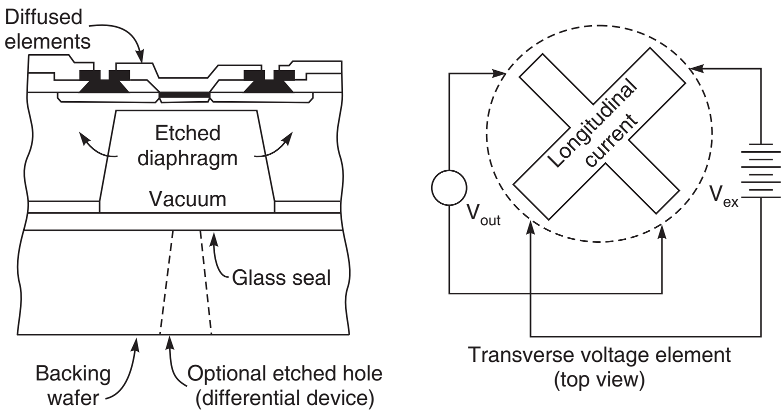


Figure 2.36 Motorola's X-ducer for pressure measurement.

Monolithic Sensors Inc. has developed a capacitance based silicon pressure sensor which can he used for low pressure measurement, has high thermal stability and high strength. Since their introduction in 1980s, the silicon pressure transducers have undergone considerable improvement.

2.9.5 Fibre-Optic Pressure Transducers

Dressner Industries, USA have brought out a pressure sensing device which uses movement of a diaphragm to amplitude modulate a light source.

Sperry Research Centre and TRW, USA have developed a photoelastic fibre-optic sensor for pressure measurement. The propagation modes in fibres possess polarization properties. These properties depend on the relationship between some preferred direction in the fibre and electric and magnetic field directions of the light waves. In fact, the light emerging from a multimode optical fibre of sufficient distance contains a large number of spatial and temporal modes whose intensities are equally divided into the two orthogonal polarization states.

The scheme developed uses a pair of optical polarizers as light valve to rotate the plane polarized light or to change the plane of polarization. As shown in Fig. 2.37, the first polarizer allows half the optical power to enter the glass block with well defined polarization status. The second polarizer is located behind the glass box. It is aligned in the cross state relative to the first polarizer. Thus in normal situation, resultant light at the output optical fibre will be nil. When pressure is applied to glass block, the polarization is changed by photoelastic effect; resulting in light in output fibre.

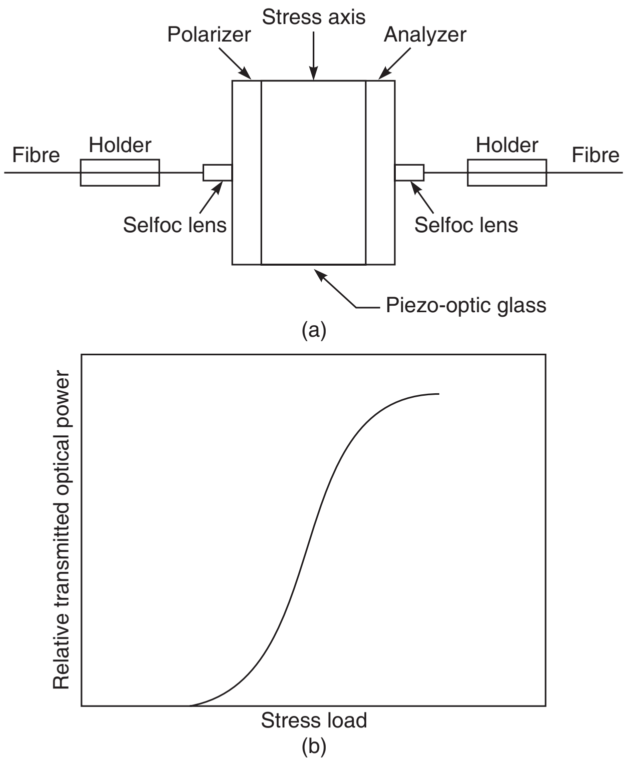


Figure 2.37 The photoelastic fibre-optic pressure sensor: (a) Device configuration, and (b) Pressure sensor response obtained at TRW.

The Fabryperot resonator cavity (discussed in Fibre-optic Temperature Transducers) made at tip of optical fibre suffers changes in length due to pressure also. A pressure sensor can also be designed using this approach.

2.10 LIQUID LEVEL TRANSDUCERS

The level sensing methods include the use of fluid pressure sensor, float, capacitive or conductive methods. Using these methods, the level of any liquid in a tank or any other container can be measured.

2.10.1 Fluid Pressure Transducers

The schematic of fluid pressure transducers is shown in Fig. 2.38. Any of the fluid pressure sensors like bellows, bourdon tube, diaphragm sensors, can he used at the bottom of tank. The fluid pressure is directly proportional to the level of liquid in the tank.

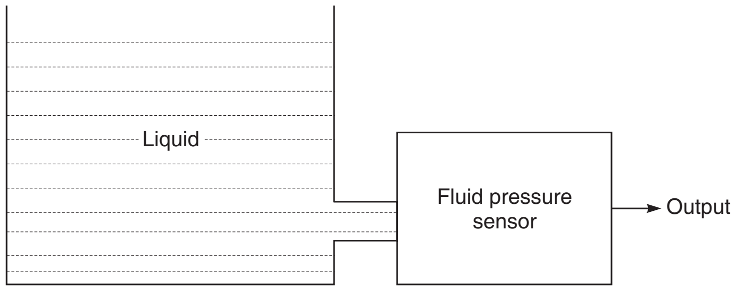


Figure 2.38 The schematic of fluid pressure transducers.

2.10.2 Float

It is the most common liquid level sensor. The float is made of hollow metal or plastic ball, which will always remain at the level of liquid. The float is operated with potentiometer. The other end of float is connected to movable arm of potentiometer as shown in Fig. 2.39. As the liquid level varies, the float goes up or comes down and this operates the movable arm of potentiometer. The change in resistance is directly proportional to the change in liquid level.

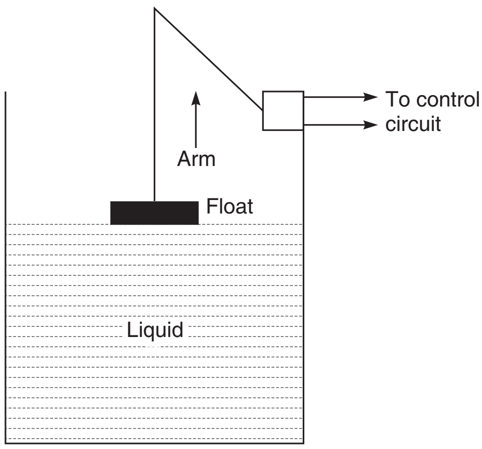


Figure 2.39 Float as liquid level sensor.

2.10.3 Capacitive Transducers

The capacitive liquid level sensor is used in case of non-conductive liquids If the tank containing the liquid is made of metal (Fig. 2.40), one insulated metal electrode is inserted in the tank, and variation of capacitance between electrode and tank wall with liquid level is measured The liquid forms the dielectric medium between the two parallel plates. As the liquid level varies, the capacitance also varies. If the tank is not made of metal then two electrodes are used and variation of capacitance between them is measured. The capacitance output is proportional to the liquid level.

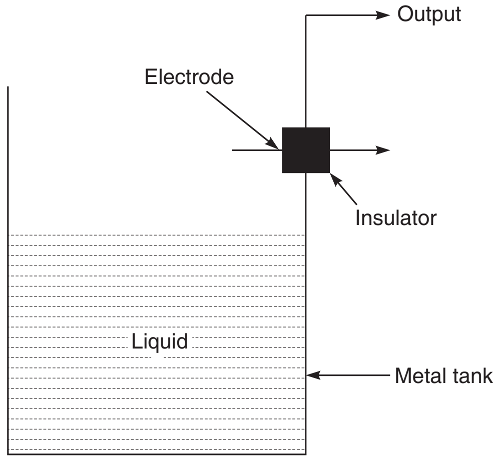


Figure 2.40 Capacitive transducer (in case of non-conductive liquids).

2.10.4 Conductive Transducers

If the liquid is conductive (Fig. 2.41) then we measure the change in resistance between two electrodes or one electrode and container wall. The current flows through the liquid and as the liquid level varies the resistance also varies proportionally.

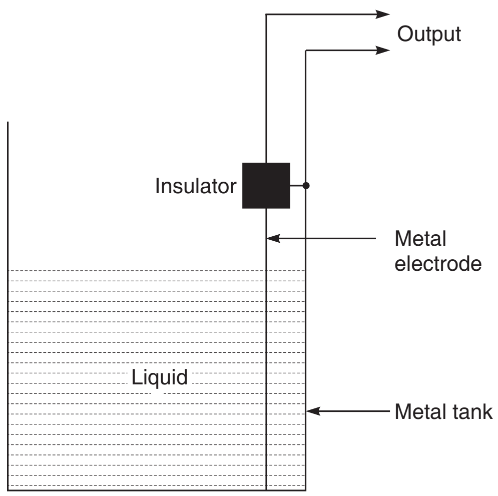


Figure 2.41 Conductive transducer (in case of conductive liquids).

2.10.5 Limit Transducers

In many applications, it is only required to know when the liquid has reached a pre-defined level, so that any desired control action (closing or opening of valve) may be taken. Most common and simplest device is float switch. It is widely used in water tanks. The arm of the float is connected to a switch through a rod (Fig. 2.42). When float reaches a particular level, the switch is operated through the rod. When the level decreases, the float comes down and again the rod operates the switch in the opposite direction. An electrode can be set at the particular level in a metal container in case of a conductive liquid. When liquid touches the electrode, the circuit is completed (Fig. 2.43). In case of non-metal container, two electrodes are used as shown in Fig. 2.44.

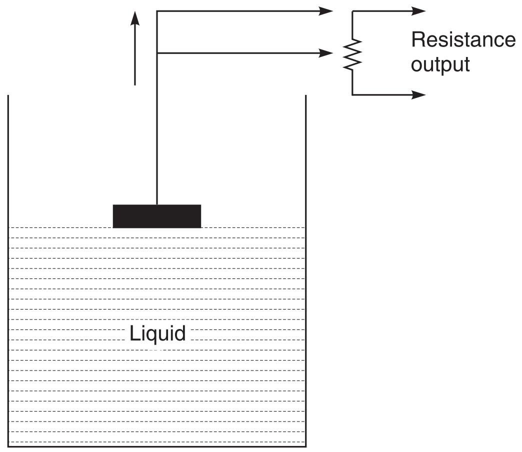


Figure 2.42 Limit transducers.

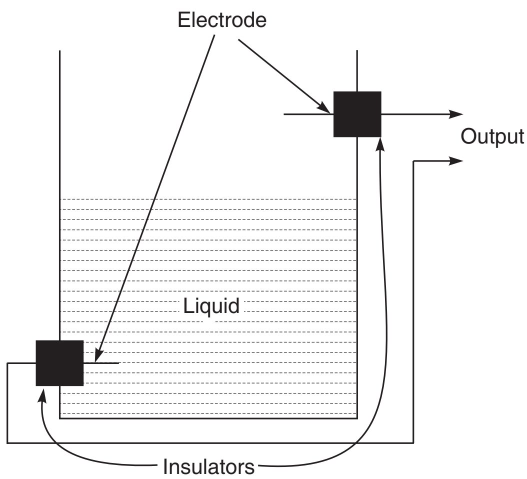


Figure 2.43 Diagram illustrating completion of a circuit when conductive liquid touches electrode.

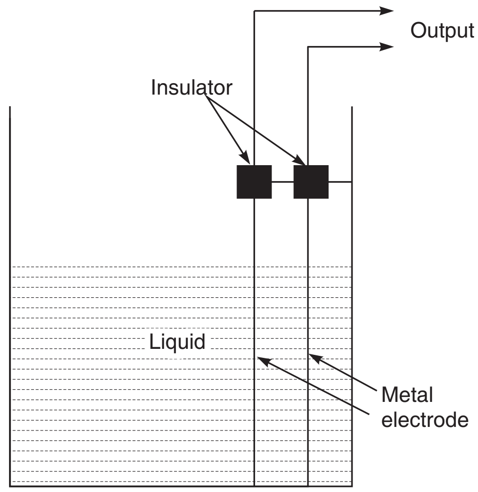


Figure 2.44 Use of two metal electrodes in nonmetal tank.

2.10.6 Silicon Liquid Level Transducers

Texas Instruments Inc. USA has developed a sensing element ST 004, to sense the liquid level. A tiny silicon chip resides in the sensor with a level wire connected to either face. The ST 004 sensor fits into a probe ST 004-B which has holes through which liquid can enter. When liquid reaches the level of probe, it enters through the holes and comes in contact with sensor chip. This changes the temperature of the silicon chip and there by changes the current passing through the chips.

2.10.7 Fibre-Optic Level Transducers

TRW and Lewis Engineering have developed an optical fibre sensor for liquid level, based on the principle of total internal reflection. The scheme is shown in Fig. 2.45. It contains an input

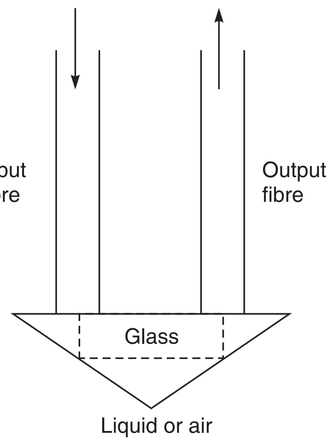


Figure 2.45 Liquid level sensor based on total internal reflection.

fibre, an output fibre and glass reflective surface. The input and output fibres and reflective surface are arranged in such a manner as to provide total internal reflection when surrounded by air. When liquid touches the glass surface, the refractivity of surface changes, thus eliminating the total internal reflection. A signal can therefore, be generated when liquid reaches a particular level.

2.11 LIQUID FLOW TRANSDUCERS

Liquid flow measurement is very important. In almost all industries, liquid flow is through pipelines. In irrigation canals as well as water and waste water treatment plants, the flow of water is measured in open channels. The measurement techniques for open channel flow are different from that of pipelines. There are numerous types of flowmeters for both pipeline and open channels. We shall discuss only those which are most commonly used.

2.11.1 Pipe Line Flow Transducers

The different pipeline flow transducers may be categorized as:

- Variable area flow transducers, e.g., rotameter;
- Differential pressure flow transducers, e.g., orifice plate, venturi tube;
- Velocity type flow transducers, e.g., magnetic flowmeter, vortex shedder;
- Displacement type flow transducers, e.g., turbine meter.

Rotameter

Rotameter belongs to the variable area flowmeter family. In its basic form the rotameter (Fig. 2.46) consists of a vertically oriented tapered glass tube, with the large end at the top, and a metering float which is free to move within the tube. Fluid flow causes the float to begin to rise in the tube as the upward pressure differential and buoyancy of the fluid overcome the effect of gravity.

The float will rise till the annular area between the float and tube increases sufficiently to allow a state of dynamic equilibrium between the pressure differential and buoyancy factors (upward) and gravity factors (downward). The height of the float is an indication of flow rate, and the tube can be graduated in appropriate flow units.

These meters typically can have up to a 12-to-1 turndown (ratio of maximum to minimum measure of flow), and industrial accuracies of +/– 2% or even 1% of the full scale rating.

The variable area flowmeter has the basic advantages of relatively low cost, accurate and reliable performance, simplicity,

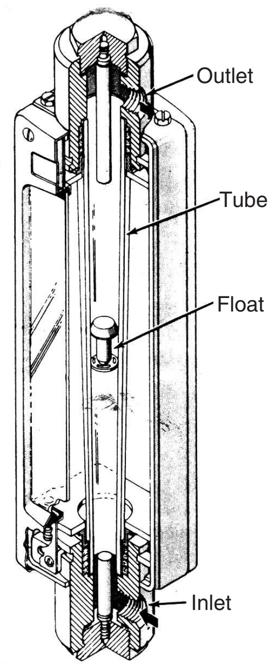


Figure 2.46 Rotameter.

and inherent versatility. It is available in a wide variety of metal and plastic bodies and with alarm and control options. It can be used with a wide variety of liquid, gas and steam applications.

Orifice Plate

An orifice plate is a very simple device installed in a straight run of pipe. The orifice plate contains a hole smaller than the pipe diameter. The flow gets constricted and experiences a pressure drop. The differential pressure can be related to flow.

The typical orifice plate has a concentric sharp-edged opening, as shown in Fig. 2.47. Because of the smaller area the fluid velocity increases, causing a corresponding decrease in pressure. The flow rate can be calculated from the measured pressure drop across the orifice plate.

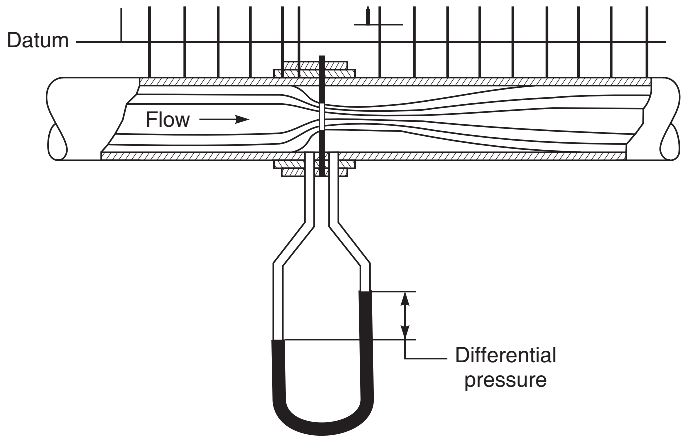


Figure 2.47 Orifice plate sensor.

In practice, the orifice plate is installed in a pipe between two flanges. Pressure taps on either side of the plate are used to detect the difference. The major advantages of orifices are that they have no moving parts and their cost does not increase significantly with pipe size.

Venturi Tube

The venturi tube shown in Fig. 2.48 is similar to an orifice meter, but it is designed to nearly eliminate boundary layer separation and thus form drag. The change in cross-sectional area in the venturi tube causes a pressure change between the convergent section and the throat, and the flow rate can be determined from this pressure drop.

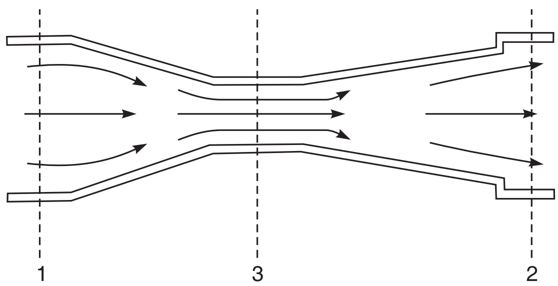


Figure 2.48 Venturi tube sensor.

The basic principle of operation is the same as that of orifice plate, as a venturi tube also measures flow rates by constricting fluids and measuring a differential pressure drop.

Venturi tubes allow for flow measurement with lower head losses than orifice plates. Venturi tubes of cast iron cones are most commonly used in pipes with diameters of 4 to 32 inches (10 to 80 cm). Pipes of up to 10 inches (25 cm) in diameter usually utilize machined venturi constrictions. Larger diameter pipes (upto 48 inches or 1.2 m) usually employ a welded sheet metal convergence.

Magnetic Flow Meter

The operating principle of magnetic flow meters is based upon Faraday's Law of Electromagnetic Induction, which states that "A voltage will be induced in a conductor moving through a magnetic field". See Fig. 2.49.

$$E = K \times B \times D \times V$$

where

E = Induced voltage
B = Strength of the magnetic field
D = Conductor width
V = Velocity of the conductor
K = Constant.

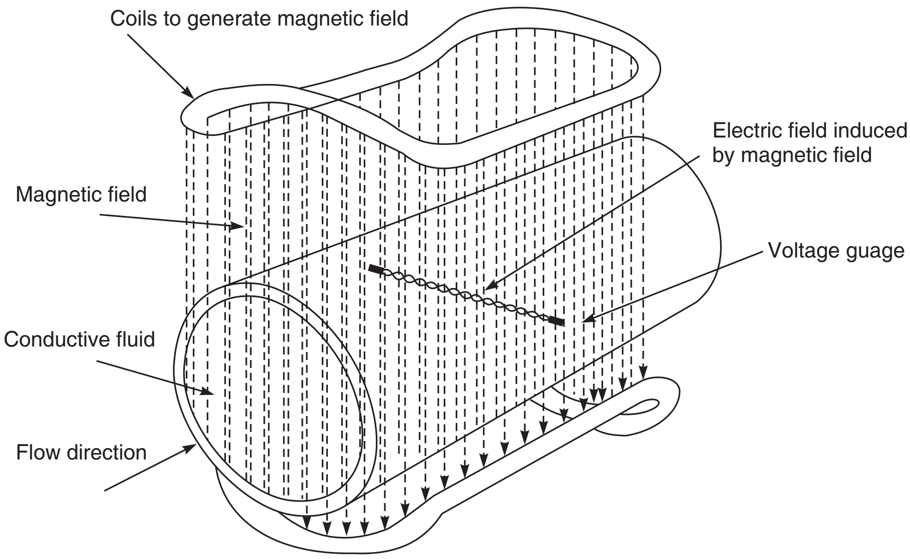


Figure 2.49 Magnetic flow meter—operation principle.

The magnitude of the induced voltage E is directly proportional to the velocity of the conductor V, conductor width D and the strength of the magnetic field B. Magnetic field coils placed on opposite sides of the pipe generate a magnetic field. As the conductive process liquid moves through the field with average velocity V, electrodes sense the induced voltage. The width of the conductor is represented by the distance between electrodes. An industry liner prevents the signal from shorting to the pipe wall. The only variable in this application of Faraday's law is the velocity of the conductive liquid *V*, because the field strength is a controlled constant and the electrode spacing is fixed. Therefore, the output voltage *E* is directly proportional to liquid velocity, resulting in the linear output of a magnetic flowmeter.

This voltage is picked up by sensing electrodes mounted in the meter tube and sent to the transmitter, which takes the voltage and calculates the flow rate based on the cross-sectional areas of the meter tube. Magnetic flowmeters are also known as *electromagnetic flowmeters* or *induction flowmeters*.

A prerequisite of using magnetic flowmeters is that the fluid must be conductive. A lining of nonconductive material is often used to prevent the voltage from dissipating into the pipe section when it is constructed from conductive material.

The following are the advantages of magnetic flowmeters:

- Minimum obstruction in the flow path, which yields minimum pressure drop
- Low maintenance cost because of no moving parts
- High linearity
- Can be used in hazardous environments or to measure corrosive or slurry fluid flow.

Vortex Shedder

When a fluid stream encounters a rock or another obstruction, it separates, moves around the object and flows downstream. At the point of contact, eddy currents or vortex swirls are formed alternately on either side of the object. This creates a local increase in pressure and a local decrease in velocity on one side of the obstruction. Meanwhile, it creates a local decrease in pressure and a local increase in velocity on the other side of the object. After the shedding of a swirl from one side, the process is reversed and a vortex or swirl is shed from the other side. Figure 2.50 shows the vortex shedding phenomenon. The frequency of this alternating shedding process is proportional to the velocity of the flowing stream as it passes the point of contact.

The vortex shedding flowmeter is a volumetric flowmeter. In the vortex shedding flow meter, the flow path is obstructed by a bluff body (or strut) that creates the vortex swirl.

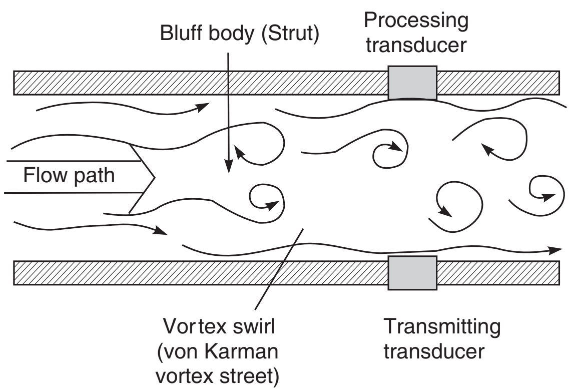


Figure 2.50 Vortex shedder.

The vortices trail behind the cylinder, alternately from each side of the bluff body. This vortex trail is called the *von Karman vortex street* after von Karman's 1912 mathematical description of the phenomenon. The frequency at which these vortices alternate sides is essentially proportional to the flow rate of the fluid.

Inside, atop, or downstream of the shedder bar is a sensor for measuring the frequency of the vortex shedding. This sensor is often a piezoelectric crystal which produces a small, but measurable, voltage pulse every time a vortex is created. Since the frequency of such a voltage pulse is also proportional to the fluid velocity, a volumetric flow rate is calculated using the cross-sectional areas of the flowmeter with the help of the following relationship:

$$F = S \times \frac{V}{L}$$

where

F = Frequency of vortices
L = Characterstic length of bluff body
V = Velocity of flow
S = *Strouhal number*, which is essentially a constant for a given body shape within its operating time.

The frequency is measured and the flow rate is calculated using the flowmeter electronics.

Turbine Meter

The most common displacement flow measuring device is the turbine meter. In a turbine meter, a rotor is placed in the flow path. Usually, the rotor is magnetically coupled so that each rotation produces a pulse. The spin of the rotor is proportional to the velocity of the fluid.

The design of a turbine-type flowmeter is shown in Fig. 2.51. It has an axially mounted freely rotating turbine rotor, having its axis coinciding with the centre-line of the pipe and the direction of flow of the fluid. The fluid in motion impinges on the rotor blades, resulting in the

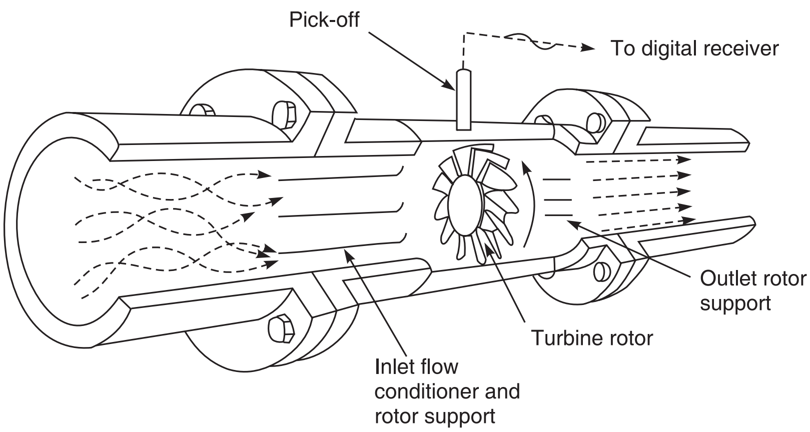


Figure 2.51 Turbine flowmeter.

development of a torque on each blade or wing of the rotor due to the shape and curvature of the blades. The rotor goes into rotation with an angular velocity proportional to the fluid velocity. Under steady-state conditions, the volume flow rate is proportional to the angular velocity. The rotor is supported by ball or sleeve bearings on a shaft, which in turn is held rigidly inside the meter. The rotor is supported by ball or sleeve bearings on a shaft, which in turn is held rigidly inside the meter. The rotor speed is measured by means of an electromagnetic transducer and associated digital readout.

The turbine meter is highly accurate and durable. Turbine meters are restricted only by the fact that they must be used in clean noncorrosive environments.

2.11.2 Open Channel Flow Measurement

Open channel flow is defined as flow in any channel in which the liquid flows with a free surface. Examples include rivers and irrigation channels. Certain closed channels such as sewers, when flowing partially full and not under pressure, are also classified as open channels.

Effective use of water for crop irrigation requires that flow rates and volumes be measured and expressed quantitatively. Measurement of flow rates in open channels is difficult because of nonuniform channel dimensions and variations in velocities across the channel.

Open channels are used to conduct liquids in most sewer systems, water treatment plants, sewage treatment plants, industrial waste applications, and irrigation systems.

Three methods are prevalent for automatically measuring open channel flow:

- Hydraulic structures
- Area velocity
- Slope-hydraulic radius.

Hydraulic Structures

The most common method of measuring open channel flow is the hydraulic structures method. A calibrated restriction inserted into the channel controls the shape and velocity of the flow. The flow rate is then determined by measuring the liquid level in or near the restriction.

The restricting structures are called *primary measuring devices*. They may be divided into two broad categories—weirs and flumes.

Weirs: Weirs allow water to be routed through a structure of known dimensions, permitting flow rates to be measured as a function of depth of flow through the structure. Thus, one of the simplest and most accurate methods of measuring water flow in open channels is by the use of weirs.

In its simplest form, a weir consists of a bulkhead of timber, metal, or concrete with an opening of fixed dimensions cut in its top edge. This opening is called the *weir notch*; its bottom edge is the *weir crest*; and the depth of flow over the crest (measured at a specified distance upstream from the bulkhead) is called the *head* (H). The overflowing sheet of water is known as the nappe.

Two types of weir exist: sharp-crested weirs and broad-crested weirs. Only sharp-crested weirs are described here because they are normally the only type used in flow measurement.

The sharp edge in the crest causes the water to spring clear of the crest, and thus accurate measurements can be made. Broad-crested weirs are commonly incorporated in hydraulic structures of various types. Although these are sometimes used to measure water flow, this is usually a secondary function for them. The components of a sharp-crested weir are shown in Fig. 2.52.

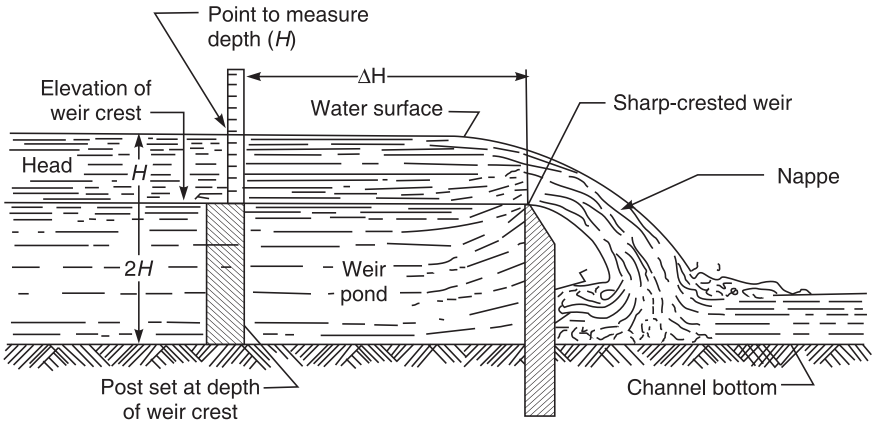


Figure 2.52 Sharp-crested weir.

Weirs are classified according to the shape of the opening. The most common types of weirs are the triangular (or V-notch) weir, the rectangular weir, and the trapezoidal (or Cipolletti) weir.

The flow rate over a weir is determined by measuring the liquid depth in the pool upstream from the weir.

V-Notch Weir: This device is especially recommended for metering flows less than 1 cubic ft per second (cfs), equivalent to 0.65 million gallons per day (mgd), and is suitable for measuring slowly changing flows up to 10 cfs. Extensive experiments have been made to determine the calibration data for V-notch weirs with included angles of 60°, 90°; two acceptable formulae for calculations are given below. See Fig. 2.53.

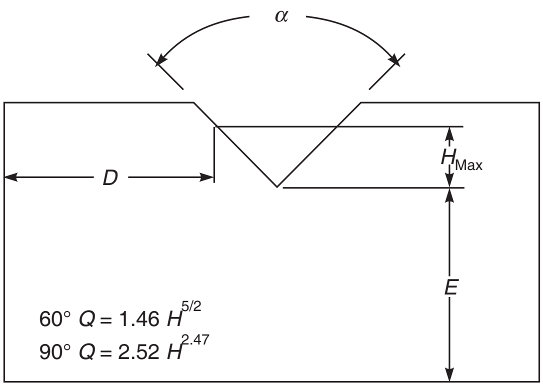


Figure 2.53 V-notch weir.

For 60°: $Q = 1.46\ H^{2.50}$

For 90°: $Q = 2.52\ H^{2.47}$

Rectangular Weir: This weir is capable of high-capacity metering, and is simple and inexpensive to construct. To assure complete contraction of the nappe, the side and bottom clearance dimensions of the notch must equal or exceed those shown in Fig. 2.54.

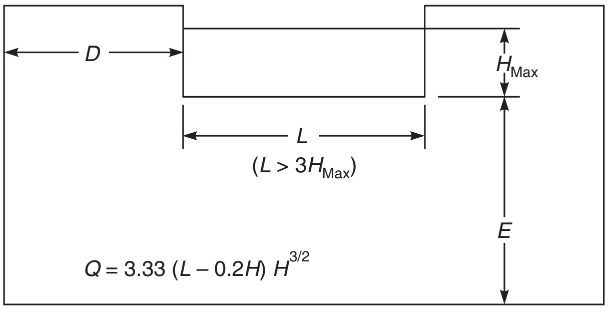


Figure 2.54 Rectangular weir.

Cipolletti Weir: This weir, shown in Fig. 2.55, is similar to the rectangular weir except for sloping sides (1 horizontal to 4 vertical) of the notch. The design has the advantage of a simplified discharge formula which is more convenient to work with.

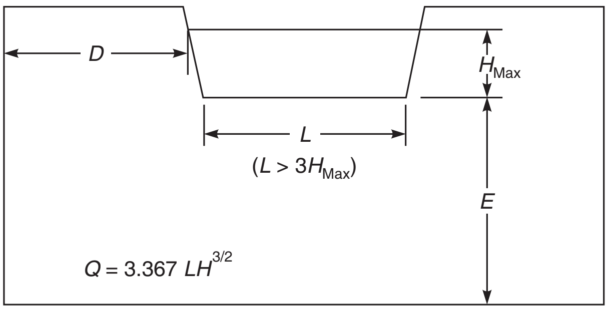


Figure 2.55 Cipolletti weir.

The weir selected should be that most adapted to the circumstances and conditions at the sites of measurement. Usually, the rate of flow expected can be roughly estimated in advance and used to select both the type of weir to be used and the dimensions of the weir.

Measurements made by means of a weir are accurate only when the weir is properly set, and when the head is read at a point some distance upstream from the crest, so that the reading will not be affected by the downward curve of the water.

Some general rules should be observed in the construction and installation of weirs. A weir should be set at right angles to the direction of flow, in a channel that is straight for a distance upstream from the weir at least ten times the length of the weir crest. The crest and sides of the weir should be straight and sharp-edged. The crest of the rectangular and Cipolletti weirs should be level and the sides should be constructed at exactly the proper angle with the crest. Each side of the V-notch weir should make a 45° or 30° angle with a vertical line through the vertex of the notch.

The channel upstream should be large enough to allow the water to approach the weir in a smooth stream, free from eddies, and with a mean velocity not exceeding 0.3 foot per second. While installing, one must avoid restrictions in the channel below the weir that would cause submergence. The crest must be placed higher than the maximum downstream water surface to allow air to enter below the nappe.

Flume: A flume is a specially shaped open channel flow section providing a restriction in channel area and/or a change in channel slope. The flow rate in the channel is determined by measuring the liquid depth at a specified point in the flume.

The most common flume is the *Parshall flume* (Fig. 2.56). The flow rate through a Parshall flume is determined by measuring the liquid level one-third of the way into the converging section.

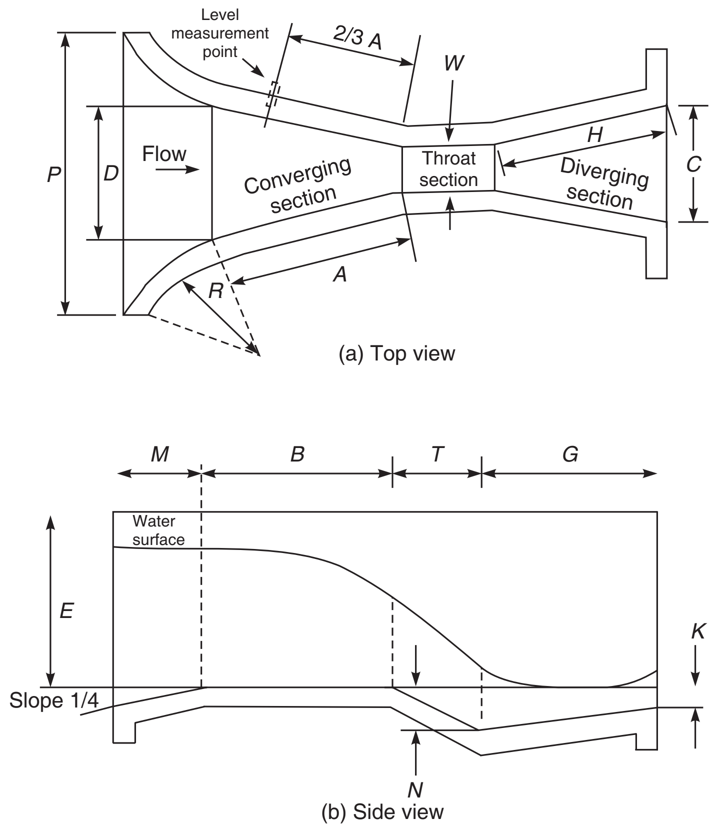


Figure 2.56 Parshall flume design.

Parshall flumes are designated by the width of the throat, which ranges from one inch to 50 feet. The throat width and all other dimensions must be strictly followed, so that standard discharge tables can be used. The head is measured in the flume upstream of the throat—in the so-called "approach channel". For Parshall flumes, the head is measured upstream from the throat at a distance of 2/3rds of the length of the approach channel (A = length of approach channel in Fig. 2.56).

All flumes must be built with their dimensions in strict accordance with specifications in published documents such as the ISO and ASTM standards. Otherwise, discharge analysis must be conducted for the specific flume, beginning with theory and proceeding to experimentation, to modify the theory by physical observations.

The methodology for the flume calculations follows that of ISO 9826, (1992) for the Parshall flume. LMNO Engineering Research and Software, USA, have performed calculations for discharge, using parshall flume, adhering to the ISO 9826 (1992) standard.

The flow discharge calculation is governed by the equation

$$Q = Ch^n$$

where

Q = Discharge in cubic metres per second

h = Head in metres measured at a specified point

C, n-coefficients. The values of C and n vary with throat width b and can be determined using the graph shown in Fig. 2.57.

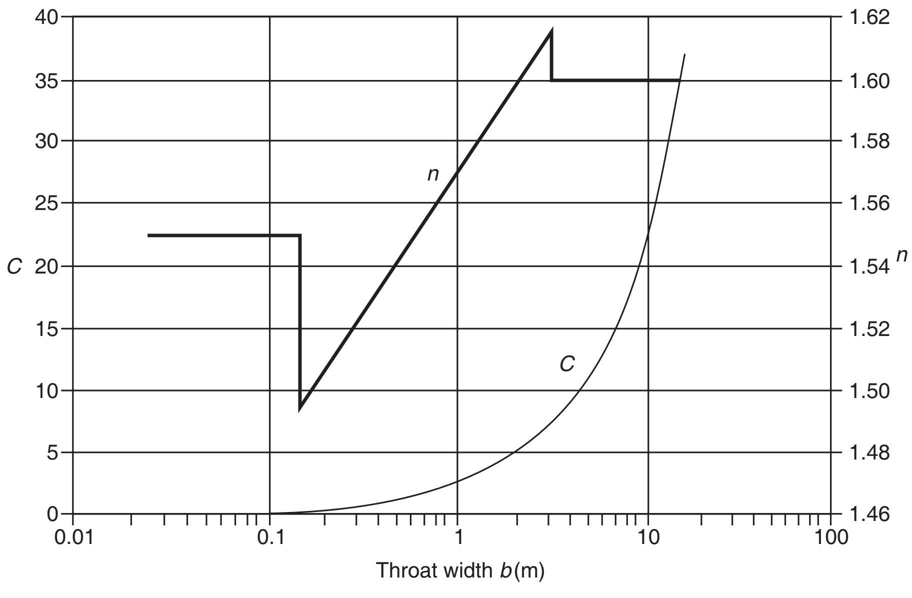


Figure 2.57 Parshall flume coefficients.

The LMNO Engineering calculation allows $0 \leq h \leq 3$ and $0.01 < b < 16$ m, but the calculation is most accurate when used within the ISO 9826 recommendations of $h \leq 2$ m and

$0.152 \leq b \leq 15.24$ m. For $b \leq 0.152$ m, C and n values are from Herschy (1995). For $b \geq 0.152$ m, C and n values are from ISO 9826 (1992).

Parshall flumes are usually constructed of reinforced concrete, but may also be of wood. Stainless steel and fibreglass-reinforced plastic liners have been used for metering corrosive solutions. Surfaces are true planes, finished smooth with close adherence to specified dimensions.

Area Velocity

The area velocity method calculates flow rate by multiplying the area of the flow by its average velocity. This is often referred to as the *continuity equation*, $Q = A \times V$.

For convenience, most area velocity flowmeters use a single sensor to measure flow rate (see Fig. 2.58). Doppler ultrasonic is used to measure average flow velocity, while an integral pressure transducer measures the level in the channel. The flowmeter converts this level into the area of the flow based on the size and shape of the channel.

The main advantage of the area velocity method is that it can be used to measure flow under a wide range of conditions. These are:

- Open channel
- Surcharged
- Full pipe
- Submerged
- Reverse flow.

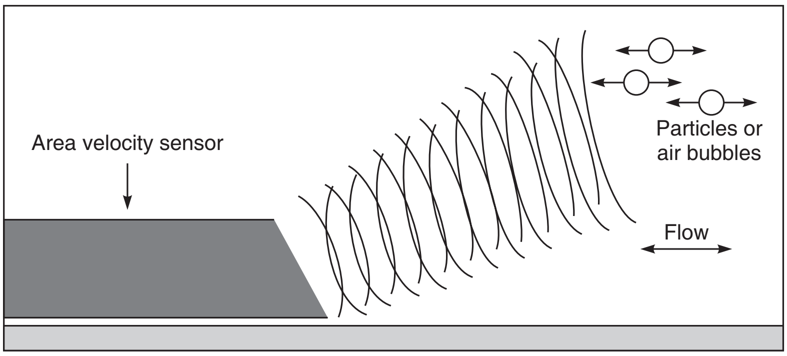


Figure 2.58 Area velocity flowmeter.

2.11.3 Semiconductor Flow Sensor

Silicon temperature sensors are also used to measure liquid flow. These sensors use the phenomenon of heat transfer.

In this approach, the measurement of the flow of a fluid (gas or liquid) is based on the determination of the heat transfer coefficient from a heated sensor surface to the fluid, the so-called *boundary layer method*. The heat transfer coefficient is the ratio of the heat flow from surface to fluid and the temperature difference between surface and fluid. It depends, amongst other things, on the flow velocity. In this application, the sensor is heated continuously.

By putting the heated sensor along the liquid flow and by determining the power loss through the surface, the heat transfer coefficient can be determined and the flow velocity may be calculated. Because the boundary layer upstream is less thick than the boundary layer downstream, as is illustrated in Fig. 2.59, and because the fluid is also heated when flowing past the hot surface, a temperature difference ΔT between the upstream and downstream ends of the flow sensor is induced.

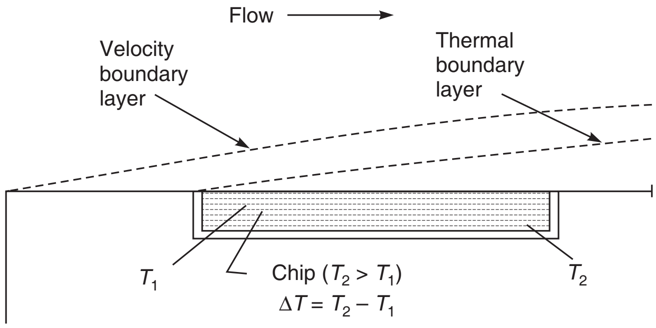


Figure 2.59 Semiconductor flow sensor—operation principle.

Since flow causes a temperature difference in a silicon chip, temperature difference is a function of the flow velocity. Temperature difference ΔT can be expressed as follows:

$$\Delta T = C(T_s - T_f)V$$

in which C is a constant, T_s the temperature of the sensor, T_f the temperature of the fluid and V the flow velocity.

For a typical silicon flow sensor, the relation is shown in Fig. 2.60. Thus, by measuring the temperature difference between the upstream and downstream edges of the sensor surface, the liquid flow can be determined. A number of configurations of silicon temperature sensor are possible.

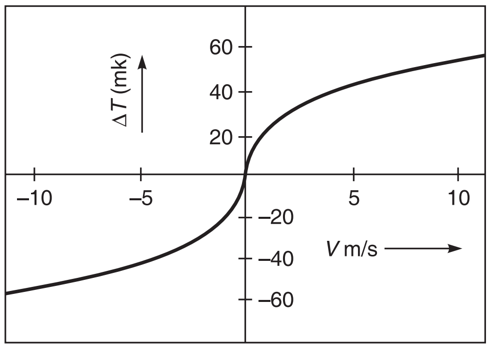


Figure 2.60 Semiconductor flow sensor: temperature–flow relationship.

The diffused Wheatstone bridge in silicon chip sensor may be used [Fig. 2.61 (a)]. The two bridge resistors lying across the flow detect the temperature difference, whereas the two resistors parallel to the flow also change their resistance but in an identical way.

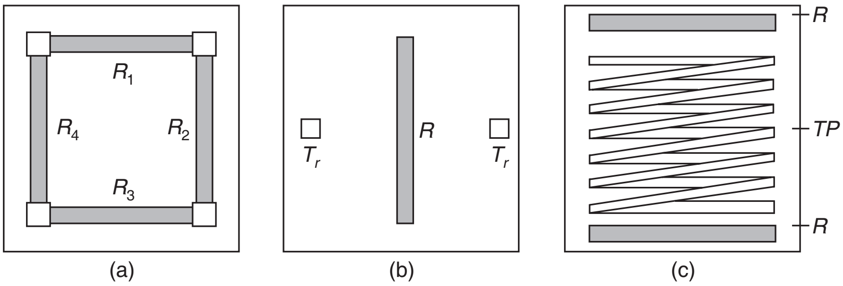


Figure 2.61 Different silicon-based flow sensors: (a) Wheatstone type; (b) Two-transistor type; (c) Thermopile-based type.

Another silicon flow sensor is based on two transistors for the measurement of the temperature difference. As is shown in Fig. 2.61 (b), the chip is heated by a resistor or a power transistor located in the centre of the chip. Upstream and downstream, two transistors are used for temperature measurement. When the transistors do not match properly, which can be expected in view of the rather large intermediate distance, appreciable offset may be observed. Therefore, in a third approach as is shown in Fig. 2.61 (c), a thermopile without offset is used for the measurement of the temperature difference. The heating resistors are placed at the edges parallel to the flow.

The sensor may also be placed on the outside of a tube and to measure the flow through the tube wall instead of placing the sensors directly in the fluid. Thus the sensor is protected from attack by the fluid. Another interesting feature is that the sensor can be constructed in such a way that they are direction-sensitive. The largest temperature gradient is always parallel to the flow vector.

2.11.4 Fibre-Optic Sensor

The vortex shedder sensor for flow measurement has been already described. The fibre-optic vortex shedding flow sensor has been more recently developed. This flow sensor has the advantage that the measuring accuracy is essentially independent of any changes in fluid temperature, viscosity, density, and light source intensity. The flow sensor (Fig. 2.62) has a sensing element, consisting of a thin metallic obstruction, and a downstream metallic bar attached to a multimode-fibre microbend sensor. As shown in Fig. 2.62, the vortex pressure produced at the metallic bar is transferred, through a diaphragm at the pipe wall, to the microbend sensor located outside the process line pipe. The microbend sensor converts the time-varying mechanical force caused by vortex shedding into a corresponding intensity modulation of the light. Therefore, the frequency of the signal converted into the electric voltage at the detector provides the flow velocity information.

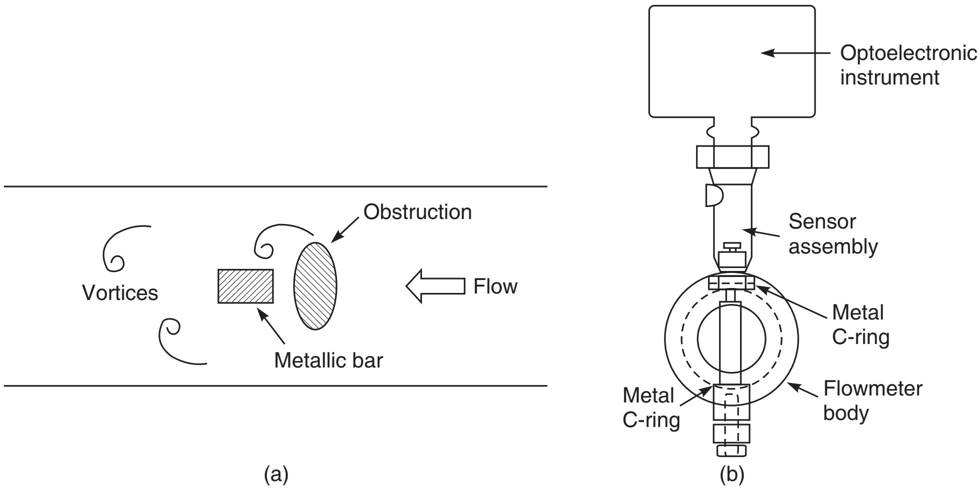


Figure 2.62 Fibre-optic vortex shedder: (a) operation principle; (b) schematic diagram.

A new design of vortex-shedding flowmeter with a Fibre-optic Fabry–Perot Interferometer (FFPI) as the sensing element has also been developed to measure the liquid flow velocity in a pipe. The flow velocity is determined by measuring the frequency at which the reflected optical power from the FFPI is modulated. Experimentally, a linear dependence of the optical modulation frequency on flow velocity is observed over the range of 0.14 to 3.0 m/s, in good agreement with theoretical prediction.

Optical fibre drag-force flow sensors have also been developed to measure the speed and direction of fluid flow. Multidirectional fluid flow measurement is made possible by vectorial addition of the orthogonal flow components. The flow sensor comprises a fibre-optic strain gauge, a cantilever-deflecting element made of rubber, and a drag element. The fibre-optic strain gauge was designed by inserting grooves into a multimode plastic optical fibre. As the fibre bends, the variation in the angle of the grooves causes an intensity modulation of the light transmitted through the fibre.

2.12 INTELLIGENT SENSORS

The advancements in microminiaturisation has resulted in combination of sensor and electronics in the same chip, thus leading to the integrated sensors or intelligent sensors. At the first step, hybrid units were developed which mounted one or more sensors, their interfaces and even microprocessor into a single package, resulting in the functional equivalent of an integrated circuit.

It is even possible to place sensors within sensors. Thus monitoring of internal temperature of sensor could be done by another internal sensor, and output of main sensor could be modified in order to compensate for known parameter changes with temperature. The output signal from sensor need not be continuous analog but may be converted to frequency, phase, pulse width as well as serial or parallel fom by integrated processing power within sensor.

2.12.1 Desirable On-chip Signal Processing

Following categories of on-chip signal processing have been identified for integration with silicon sensors:

(i) *Amplification:* The electrical output signal of sensor is weak, and therefore, not suitable for transmission. This can be amplified.

(ii) *Signal conditioning and formatting:* The output signal of sensor may be converted to digital for better noise immunity. The digital signal can then be encoded in several forms like serial, parallel, frequency, phase and pulse rate. The output format from sensors could also be standardised. Most have high internal impedance, which makes these susceptible to noise. On-chip impedance transformation could be achieved to solve this problem.

(iii) *Improvement in characteristics*: On-chip electronics could bring the following improvements in the characteristics of sensors:

  (a) *Non-linearity*: Most sensors show some non-linearity. This could be improved using look up table or feedback system.

  (b) *Cross sensitivity:* Most sensors show undesirable sensitivity to strain and temperature: These parameters can be measured by incorporating such sensor elements in the sensors. The effect could be compensated through on-chip electronics.

  (c) *Frequency response:* The frequency response of sensor could be improved through proper feedback mechanism, using on-chip electronics.

  (d) *Parameter drift*: The offset, sensitivity and linearity undergo change with time as component values change. Using on-chip accurate current and voltage sources, this problem could be overcome.

2.12.2 Present Status

The silicon pressure sensors have been in use for quite some time. A number of piezoresistive silicon transducers are available now with on-chip amplification and temperature compensation. A pressure sensor with frequency output has also been designed. Honeywell's Process Control Division has developed an intelligent pressure sensor that compensates for both temperature changes and offsets due to static pressure loading effects.

The Sharp Corporation has developed a combined humidity/temperature sensor on a single chip. It was difficult to combine humidity sensor with FET, since silicon transistor elements are sensitive to moisture. The problem has been solved by using humidity-resistant silicon nitride layer on the surface of the element, and a double gate electrode structure. The temperature sensing range is between –20 to +100°C with response time of 30 seconds. The humidity change from 0 to 100 per cent may be sensed with response time of 60 seconds or less.

A number of developments in intelligent sensors have been reported in the area of *tactile sensors* used in Robots. Tactile sensing is a continuous variable sensing of forces in an array and is meant to relate to skin-like properties where areas of force sensitive surfaces are capable of reporting graded signals and parallel pattern of touching. This is in contrast to simple touch, which is force sensing at a single-point or binary sensing at multiple sites. Since high level of integration and processing is basic requirement in robotics, the intelligent sensing efforts are directed towards robotic applications.

2.13 MEMS SENSORS

The quest to develop smaller, smarter and low-power devices saw the emergence of MEMS (Micro Electro Mechanical Systems) technology. MEMS technology has graduated from academic development to become an integral part of many common products. But the design challenges and problems associated with practical implementation of MEMS technology are getting solved slowly.

As with all new technologies, both designers and users of MEMS devices have a learning curve to overcome. The effort is worthwhile, as the latest generation MEMS devices have enabled innovative new products with high performance and low cost.

2.13.1 MEMS Sensor—An Introduction

In early MEMS systems a multi-chip approach, with the sensing element (MEMS structure) on one chip and the signal conditioning electronics on another chip, was used. While this approach is simple from a process standpoint, it has many disadvantages:

- The overall silicon area is generally large.
- Multi-chip modules require additional assembly steps.
- The yield is generally lower for multi-chip modules.
- Larger signals from the sensor are required to overcome the stray capacitance of the chip-to-chip interconnections and stray fields necessitating a larger sensor structure.

Considering that integration is the most cost-effective and high-performance solution, an integrated approach to MEMS (where the sensor and signal conditioning electronics are on one chip) was pursued.

2.13.2 MEMS Sensors—Design Challenges

The electronic design of MEMS sensors is very challenging. In most MEMS sensors, mechanical systems are designed to realize a variable capacitor. Electronics is used to convert the variable capacitance to a variable voltage or current, and to amplify, linearize, and in some cases, temperature-compensate the signal. This is a challenging task as the signals involved are very minute.

The integrated approach present further challenges. Many standard production steps that improve the mechanical structure degrade the electronics and vice versa.

The fabrication process design challenge is perhaps the greatest one. Techniques for building three-dimensional MEMS structures have to be devised. Chemical and trench etching can be used to “cut out” structures from solid polysilicon, but additional process steps must be used to remove the material underneath the patterned polysilicon to allow it to move freely.

Standard plastic injection-moulded IC packaging cannot be used because of the moving parts of the MEMS structure. A cavity of some type must be maintained around the mobile MEMS structure. So alternative low-cost cavity packaging was developed. In addition, this package must also be mechanically stable as external mechanical stress could result in output changes.

Even mundane tasks, such as cutting the wafer up into a single die, becomes complicated. In a standard IC the particle residue created by the sawing process does not effect the IC. In a moving MEMS structure, these particles can ruin a device.

MEMS sensors, like almost all electronic devices, do not exhibit ideal behaviour. While most designers have learned how to handle the non-ideal behaviour of op-amps and transistors, few have learned the design techniques used to compensate for non-ideal MEMS behaviour. In most cases, this type of information is not available in textbooks or courses, as the technology is quite new. So, generally designers must get this type of information from the MEMS manufacturers.

2.13.3 MEMS Sensor Generations

MEMS sensor generations represent the progress made in micro-sensor technology and can be categorized as follows:

1st Generation: MEMS sensor element is mostly based on a silicon structure, sometimes combined with analog amplification on a microchip.

2nd Generation: MEMS sensor element is combined with analog amplification and analog-to-digital converter on one microchip.

3rd Generation: Fusion of the sensor element with analog amplification, analog-to-digital converter, and digital intelligence is carried out for linearization and temperature compensation on the same microchip.

4th Generation: Memory cells for calibration and temperature compensation data are added to the elements of the 3rd MEMS sensor generation.

2.13.4 MEMS Sensor Developments

MEMS technology has been widely used for inertial sensing. This includes accelerometers and rate gyroscopes. Others are automotive crash sensors, high-resolution seismic sensors and high-g sensors.

Accelerometers

The physical mechanisms underlying MEMS accelerometers include capacitive, piezoresistive, electromagnetic, piezoelectric, ferroelectric, optical, and tunnelling. The most successful types are based on capacitive transduction. The reasons are: the simplicity of the sensor element itself, no requirement for exotic materials, low power consumption, and good stability over temperature. Although many capacitive transducers have a nonlinear capacitance versus displacement characteristic, feedback is commonly used to convert the signal to a linear output.

The output can be either analog, digital or ratiometric to the supply voltage, or any of the various types of pulse modulation. Sensors with digital output are convenient when the data must be transmitted without further noise degradation.

There are many contributors to noise in an accelerometer—the sensor itself, the readout electronics, mechanical damping, and all electrical resistances. MEMS sensors are so small that the Johnson noise of the devices' mechanical resistance must be considered, while it is usually ignored in larger sensors. Just as Brownian motion agitates bacteria and dust motes, it can be a large force on a tiny MEMS component.

Analog Devices has developed the successfully integrated MEMS accelerometer ADXL202E. It is claimed to be world's smallest mass produced, low-g, low-cost, integrated MEMS dual-axis accelerometer. The sensor has the ability of sensing both dynamic acceleration (i.e., shock or vibration) and static acceleration (i.e., inclination or gravity).

Differential capacitance is measured using synchronous modulation or demodulation techniques. After amplification, the x-and y-axis acceleration signals each go through a 32 kΩ resistor to an output pin (C_x and C_y) and a duty cycle modulator (the overall architecture can be seen in the block diagram in Fig. 2.63). The user may limit the bandwidth, and thereby lower the noise floor, by adding a capacitor at the C_x and C_y pins.

The output signals are voltage proportional and pulse-width-modulation (PWM) proportional to acceleration. Using the PWM outputs, the user can interface the ADXL202 directly to the digital inputs of a microcontroller using a counter to decode the PWM.

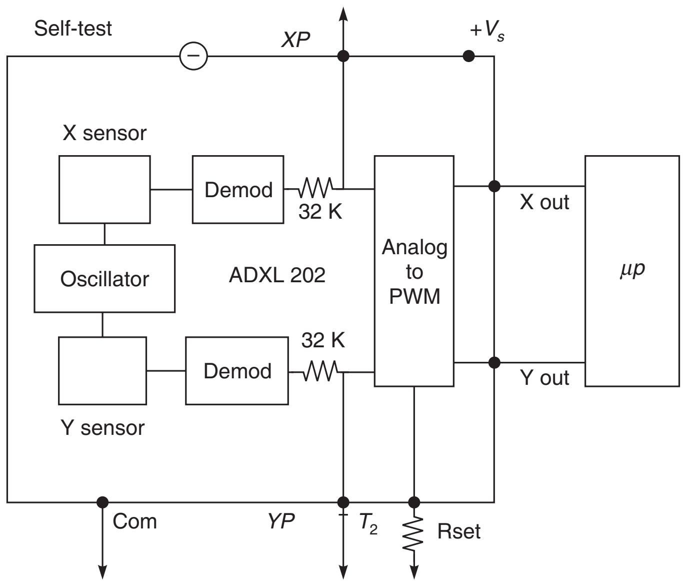


Figure 2.63 MEMS accelerometer—ADXL 202E.

Angular Accelerometers: MEMS angular accelerometers are used primarily to compensate for angular shock and vibration in disc read/write head assemblies. These devices, while similar to linear accelerometers in terms of design, fabrication, and readout, are designed with *zero pendulosity* (i.e., the centre of gravity is located at the centroid of the support springs). They are compliant to rotational motion, yet stiff with respect to linear motion. Delphi and ST Microelectronics, manufacturers of angular accelerometers, use capacitive MEMS sensors and custom CMOS ASICs.

Geophones: Geophones can be thought of as accelerometers with very high sensitivity and no DC output requirement. With no drift of bias stability specifications, geophone design can be optimized to give the lowest noise floor. Applications include seismic sensing, machinery vibration and failure prediction, tracking and identification of vehicles or personnel, and underwater pressure gradient sensing.

Conventional geophones incorporate permanent magnets and fine wire coils to measure velocity above their fundamental resonances. This is in contrast to capacitive accelerometers, which measure acceleration below their fundamental resonances. There are also piezoelectric and ferroelectric accelerometers. Micro-machined sensors can offer size and weight advantages over conventional sensors, as well as digital output which is easily transmitted noise-free.

One MEMS geophone that has been commercialized by Applied MEMS Inc. incorporates a three-wafer stack in which the centre wafer forms the proof mass. The device is read out using a custom CMOS mixed-signal ASIC with a ΣΔ. loop operating in force-feedback mode.

Gyroscopes: Many types of MEMS gyroscopes have appeared in the relevant literature, with most falling in the categories of tuning-fork gyroscopes, oscillating wheels, Foucault pendulums, and wine glass resonators. Conventional (non-MEMS) spinning wheel gyroscopes are common, but an MEMS device with no springs that undergoes levitation and rotation has not yet been commercialized.

Tuning fork gyroscopes contain a pair of masses that are driven to oscillate with equal amplitude but in opposite directions. When rotated, the Coriolis force creates an orthogonal vibration that can be sensed by a variety of mechanisms. Draper Lab has developed a gyroscope by using comb-type structures to drive the tuning fork to resonance.

Rotation causes the proof masses to vibrate out of plane, and this motion is sensed capacitively with a custom CMOS ASIC. The technology has been licensed to Rockwell, Boeing, Honeywell, and others.

The resonant modes of a MEMS inertial sensor are extremely important. In a gyroscope, there is a typically a vibration mode that is driven and a second mode for output sensing. In some cases, the input and output modes are degenerate or nearly so. If the I/O modes are chosen such that they are separated by 10%, the open-loop sensitivity will be increased due to the resonance effect. It is also critical that no other resonant modes be close to the I/O resonant frequencies.

Samsung Crop. has put a large effort into inertial sensors for automotive and consumer electronics applications. Gyroscopic stabilization of camcorders has been shown to improve picture quality, and gyroscopes enhance vehicular safety in multiple ways.

In *vibrating-wheel gyroscopes* the wheel is driven to vibrate about its axis of symmetry, and rotation about either in-plane axis results in the wheel's tilting—a change that can be detected with capacitive electrodes under the wheel, as shown in Fig. 2.64.

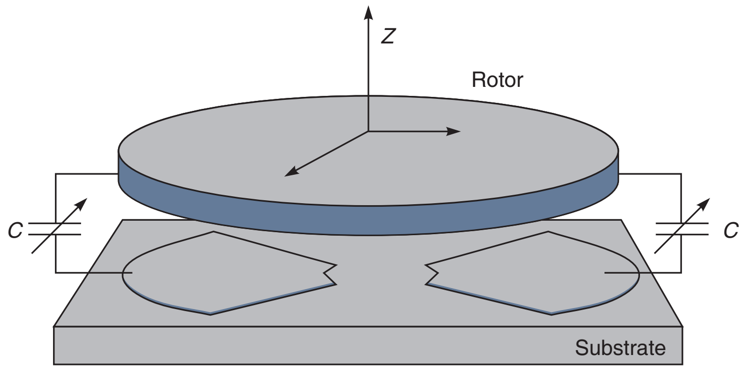


Figure 2.64 Vibrating-wheel gyroscope developed by Bosch.

It is possible to sense two axes of rotation with a single vibrating wheel. A surface micromachined polysilicon vibrating wheel gyroscope has been designed at the Sensors and Actuators Center of the University of California at Berkeley.

A third type of gyroscope is the *wine glass resonator*. Fabricated from fused silica, this device is also known as *hemispherical resonant gyroscope*. Researchers at the University of Michigan have fabricated resonant-ring gyroscopes in planar form. In a wine glass gyroscope, the resonant ring is driven to resonance and the positions of the nodal points indicate the rotation angle. The input and output modes are nominally degenerate, but some tuning is required due to imperfect machining. The output signal is monitored and nulled, yielding a rate gyroscope.

Silicon Sensing Systems, a joint venture between Sumitomo and British Aerospace, has brought to market an electromagnetically driven and sensed MEMS gyroscope. A permanent magnet sits above the MEMS device. Current passing through the conducting legs creates a force that resonates the ring. This Coriolis-induced ring motion is detected by induced voltages as the legs cut the magnetic field.

Analog Devices has been working on MEMS gyroscopes for many years, and has patented several concepts based on modified tuning forks.

The company has recently introduced the ADXRS family of integrated angular rate-sensing gyroscopes, in which the mass is tethered to a polysilicon frame that allows it to resonate in only one direction. Capacitive silicon sensing elements interdigitated with stationary silicon beams attached to the substrate measure the Coriolis-induced displacement of the resonating mass and its frame (Fig. 2.65).

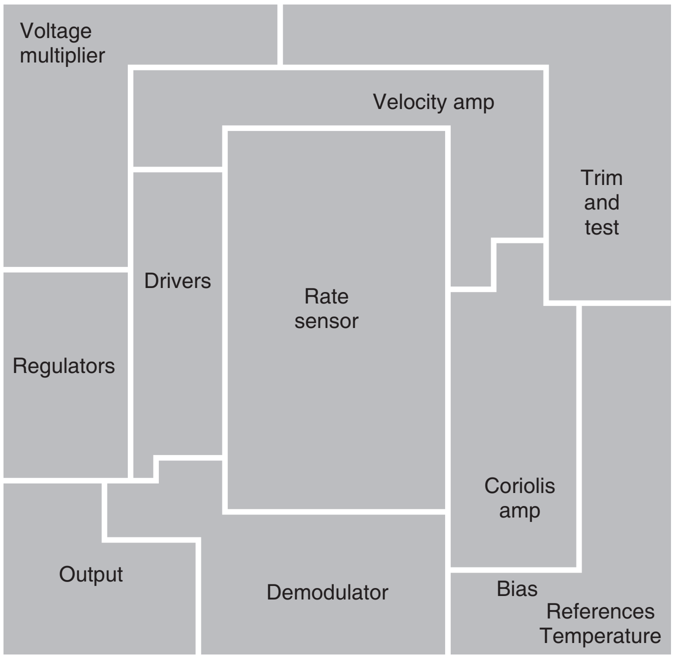


Figure 2.65 ADXRS angular rate sensing gyroscope design.

Foucault pendulum gyroscopes are based on a vibrating rod that is typically oriented out of the plane of the chip. They are therefore challenging to build with planar fabrication tools, but recent advances in MEMS technology allow very high aspect ratio MEMS that make it possible to fabricate the pendulum without hand assembly of the rod.

2.14 BIOSENSORS

The advent of biosensors have led to positive developments in the field of process control. The combination of electro-chemistry, biochemistry, physics and integrated circuit silicon technology has resulted in highly specific, sensitive, selective, accurate and reliable micro-biosensors. The application of biosensors are primarily in the following areas:

1. Robotics (sensing devices for automated machinery in hostile environment, vehicles, health care),
2. Environment (detection of toxic chemical in air, water and soil),
3. Industrial Control (water and waste monitoring, cosmetic testing, fermentation, food and drug processing),
4. Medical (drug testing, blood banks screening laboratories etc.),
5. Agricultural (diagnosis of plant and animal diseases, solid and water testing, BOD measurement, quality control of meat and plant products).

Currently, four general sensors technologies, viz., electronic, surface acoustics waves, optoelectronic and microchromotography are being investigated in different laboratories.

Microelectronics biosensors, based on conductimeteric principle have been introduced and are giving good results.

2.14.1 Biosensor Technology

A *biosensor* is an analytical device which uses biologically sensitive material to detect biological or chemical species directly, without complex sample processing. Usually a biologically sensitive material is attached to a suitable transducing system which converts the biochemical response to a quantifiable electrical signal. The biologically sensitive material can be an enzyme, organelle, membrane component, bacterial cell, an antibody or an antigen. Figure 2.66 shows the configuration of a generalised biosensor. When biological molecules interact, this results in the change in one or more physico-chemical parameters associated with the interaction. This change may produce ions, electrons, gases, heat, mass or light. These quantities are converted into the electrical signal by transducers. The transducer signal may be

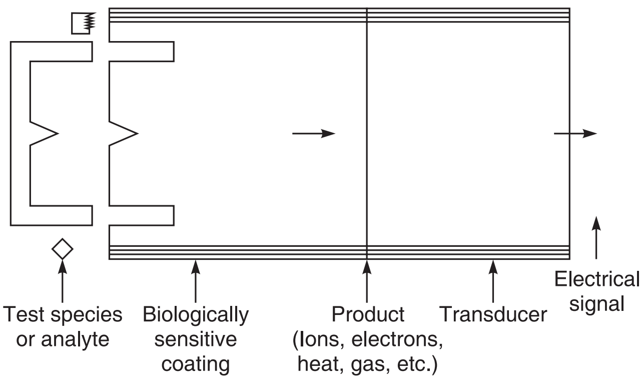


Figure 2.66 Configuration of a generalised biosensor.

amplified, *processed* and *displayed* in suitable form. Table 2.3 shows the types of the transducer measurement, and typical applications which have been exploited so far. The biosensors may be of the following types:

Table 2.3 Types of Transducers Exploited in Biosensors

Transducer system	*Measurement mode*	*Typical applications*
1. Electrochemical		
(a) Conductimetric	Conductance	Enzyme-catalysed reactions
(b) Enzyme electrode	Amperometric (current)	Enzyme substrates and immunological systems (antibody-antigen)
(c) Field effect transistors (FET)	Potentiometric (voltage)	Ions, gases, enzyme substrate and immunological analytes
(d) Ion-selective electrodes (ISE)	Potentiometric (voltage)	Ions in biological media, enzyme electrodes, immunoelectrodes

(e) Gas-sensing electrodes	Potentiometric (voltage)	Gases, enzymes, organelle, cell or tissue electrodes, enzyme immunoelectrodes
(f) Impedimetric	Impedance	Enzyme immunosensors
2. Piezoelechic crystals, acoustic devices	Mass change	Volatile gases, vapours and immunological analytes
3. Optoelectronic, fibre-optics and waveguide devices	Optical	pH, enzyme substrates, immunological analytes
4. Thermistors, diodes	Thermometry/calorimetric (heat)	Enzyme, organelle, whole cell or tissue sensors for substrates, gases, pollutants, antibiotics, vitamins, immunological analytes.

1. Conductimetric biosensors
2. Amperometric biosensors
3. Potentiometric biosensors
4. Piezoelectric biosensors
5. Optical biosensors

Conductimetric biosensors

The conductimetric principle of measurement is applicable to chemical systems because many chemical reactions produce ions and thus the conductivity of the solution is changed. Microconductimetric urea biosensors have been developed in which a small quantity of urea enzymes is immobilised over the sample pair of interdigitated serpentined network on the sensor (Fig. 2.67) chip by forming a cross-linked enzyme, i.e. albumin membrane with glutaraldhyde. When the test solution containing urea is placed over this membrane, decomposition of urea takes place which increases the overall conductivity of test solution.

Amperometric biosensors

Amperomeaic biosensors are also called enzyme electrodes. These sensors combine the specificity and selectivity of enzymes with the analytical power of electrochemisny. The method involves application of a constant potential between the sensing electrode and an auxiliary electrode in the test environment and measuring the resultant steady-state current. An amperometric chip is similar to a conductimetric chip, except that it contains two additional patterns; a counter electrode and a pattern in silver/silver chloride as reference electrode. The processing of these devices is therefore very similar to that mentioned earlier for conductimetric devices.

Potentiometric biosensors

Potentiometric measurements operate on the principle of accumulation of charge density at an electrode surface, resulting in development of a significant potential at that electrode. These sensors are mainly based on field effect transistors.

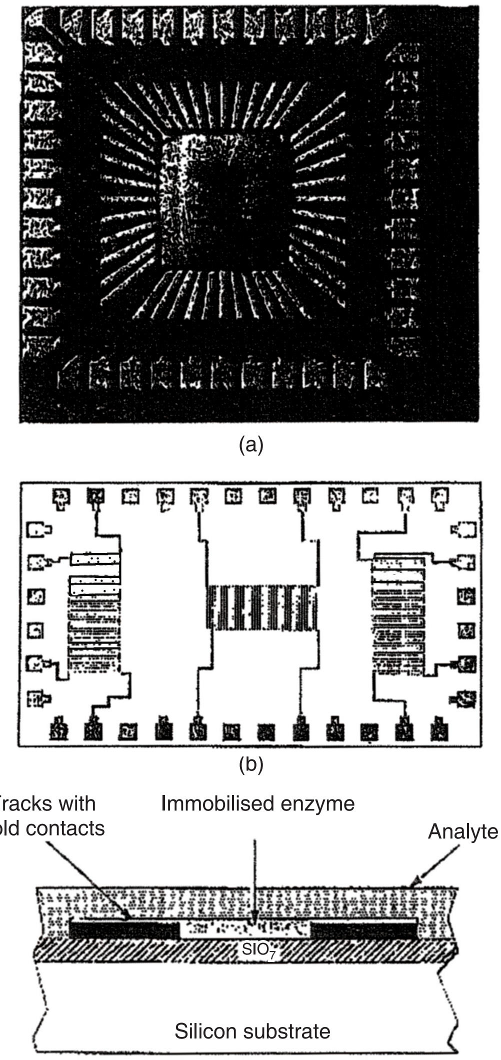


Figure 2.67 (a) A microelectronic enzyme conductimetric device on a chip carrier, (b) Microcircuit surface, and (c) Cross-section of a pair of tracks with immobilised enzymes.

Piezoelectric biosensors

The functioning of piezoelectric biosensors is based on the principle of measurement of change in resonant frequency of piezoelectric crystal as a result of mass changes on its surface. These changes are caused by the interaction of test species with a biospecific agent, immobilised on the crystal surface. The frequency of vibration of crystal normally decreases as the analyte binds to the receptor, coating the surface. Such sensors normally operate by propagation of acoustoelectric waves along the surface of the crystal and are commonly referred as Surface

Acoustic Wave (SAW) devices. A second generation SAW devices for detection of formation of antibody-antigen complexes in fluids are under development and testing stage.

Optical biosensors

The optical biosensors are becoming very popular, since the roles of superior fibre optics and optoelectronic transducers for measuring biological reactions have been realised. Enzyme reactions change the optical properties of certain substances. Light emission from biological element (bioluminiscence) or its response to illumination may he conveniently monitored via fibre or other optical wave-guide devices.

2.15 NANOSENSORS

The trend towards small sensors began with the miniaturization of macro techniques, which led to the now well-established field of microtechnology. Electronic, optical, and mechanical microtechnologies have all profited from the smaller, smarter, and less costly sensors that resulted from work with ICs, fibre optics, other micro-optics, and MEMS (Micro Electro Mechanical Systems). As we continue to work with these minuscule building blocks, there will be a convergence of nanotechnology, biotechnology, and information technology, among others, with benefits for each discipline. Substantially smaller size, lower weight, more modest power requirements, greater sensitivity, and better specificity are just a few of the improvements that we will see in sensor design in the near future.

Nanosensors and nano-enabled sensors have applications in many industries, among them transportation, communication, building and facilities, medicine, safety, and national security, including both internal security and military operations. Nanowire sensors may be used to detect chemicals and biologics. Nanosensors placed in blood cells have been used successfully to detect early radiation damage in astronauts. Nanoshells that detect and destroy tumours have also been developed.

2.15.1 Nanotechnology—An Introduction

A nanometre (nm) is one thousand millionth of a metre. For comparison, a red blood cell is approximately 7,000 nm wide and a water molecule is almost 0.3 nm across. At the nanoscale the properties of materials can be very different from those at a larger scale. Nanoscience is defined as the study of phenomena and manipulation of materials at atomic, molecular and macromolecular scales, where properties differ significantly from those at a larger scale; and nanotechnology as the design, characterization, production and application of structures, devices and systems by controlling shape and size at the nanometer scale.

The properties of materials can be different at the nanoscale for two main reasons:

Nanomaterials have a relatively larger surface area when compared to the same mass of material produced in a larger form. This can make materials more chemically reactive (in some cases materials that are inert in their larger forms are reactive when produced at nano-scale in one dimension (for example, very thin surface coatings), in two dimensions (for example, nanowires and nanotubes) or in all three dimensions (for example, nanoparticles).

The bulk properties of materials often change dramatically with nano ingredients. Composites made from particles of nano-size ceramics or metals smaller than 100 nanometres can be much stronger than what is predicted by existing materials science models. For example, metals with a so-called *grain size* of around 10 nanometres are as much as seven times harder and tougher than their ordinary counterparts with grain sizes in hundreds of nanometres. The causes of these drastic changes stem from the weird world of quantum physics. The bulk properties of any material are merely the average of all the quantum forces affecting all the atoms. As you make things smaller and smaller, you eventually reach a point where the averaging no longer works.

Nanotechnology enables us to create functional materials, devices, and systems by controlling matter at the atomic and molecular scales, and to exploit novel properties and phenomena.

Much of nanoscience and many nanotechnologies are concerned with producing new or enhanced materials. Nanomaterials can be constructed by 'top-down' techniques, producing very small structures from larger pieces of material, for example, by etching to create circuits on the surface of a silicon microchip. They may also be constructed by 'bottom-up' techniques, atom by atom or molecule by molecule. One way of doing this is *self-assembly*, in which the atoms or molecules arrange themselves into a structure due to their natural properties. Crystals grown for the semiconductor industry provide an example of self-assembly, as does chemical synthesis of large molecules. A second way is to use tools to move each atom or molecule individually. Although this positional assembly offers greater control over construction, it is currently very laborious and not suitable for industrial applications.

2.15.2 Nanotechnology Developments

The present status of nanotechnology is the result of several 20th century advances. Of particular importance was the ability to manipulate individual atoms in a controlled fashion—a sort of *atomic bricklaying*—by techniques such as *scanning probe microscopy*. Initial successes in producing significant amounts of silver and gold nanoparticles helped to draw even more attention, as did the discovery that materials and devices on the atomic and molecular scales have new and useful properties due to surface and quantum effects.

Another major event was the development of Carbon Nano Tubes (CNTs)—extremely narrow, hollow cylinders made of carbon atoms. Both single- and multi-walled CNTs could, for example, be functionalized at their ends to act as biosensors for DNA or proteins. The single-walled versions can have different geometries (Fig. 2.68). Depending on the exact orientation of the carbon atoms, a CNT can exhibit either conducting (metallic) or semiconducting properties. This characteristic, and the ability to grow CNTs at specific locations and manipulate them afterwards, make it likely that the tubes will be important for electronics and sensors. For instance, they can be used in the fabrication of nano field-effect transistors for electronics or as biological probes for sensors, either singly or as an array.

It is now possible to contemplate sensing the interaction of a small number of molecules, processing and transmitting the data with a small number of electrons, and storing the information in nanometer-scale structures. Fluorescence and other means of single-molecule detection are being developed. IBM and others are working on data storage systems that use proximal probes to make and read nanometer-scale indentations in polymers.

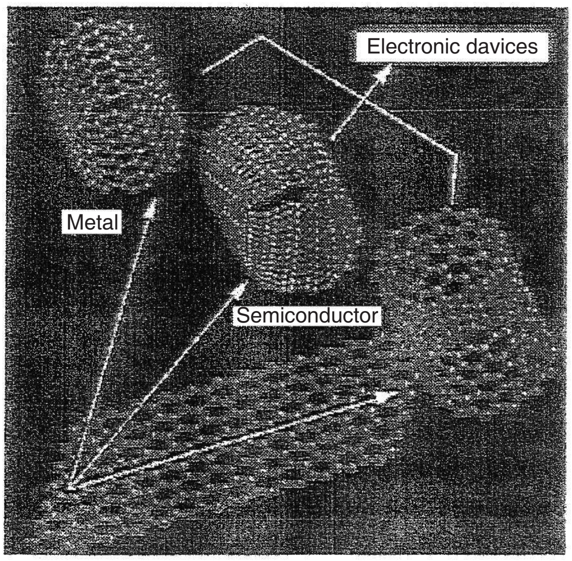


Figure 2.68 Carbon nano tube.

2.15.3 Nanosensor Developments

Few sensors today are based on pure nanoscience, and the development of nano-enabled sensors is in the early stages; yet we can already foresee some of the possible devices and applications. Sensors for physical properties were the focus of some early development efforts, but nanotechnology will contribute most heavily to realizing the potential of chemical and biosensors for safety, medical, and other purposes. A number of nanosensors are under development in different laboratories.

Pressure Sensor

Researchers at Georgia Institute of Technology, USA, have developed a nano wire pressure sensor that can be used to measure extremely low pressures.

Zinc oxide nanowires are extremely sensitive to tiny forces, in the nano- to pico-newton range. When a small force (arrow) bends a nanowire, electrical charges accumulate on the wire's surface and decrease the current flowing through the wire. The principle could be used to make small pressure sensors that can be implanted in the body and on aircraft and space shuttles (Fig. 2.69).

By connecting the two ends of a zinc oxide nanowire to electrodes, researchers have made devices similar to the transistors in electronic devices. In an electronic transistor, applying a voltage to the gate electrode controls the flow of current between the source and drain electronics. In the new pressure-sensing transistor, the two electrodes that the nanowire is connected to act as the source and the drain, but there is no gate. Instead of applying a voltage at the gate, one simply bends the wire.

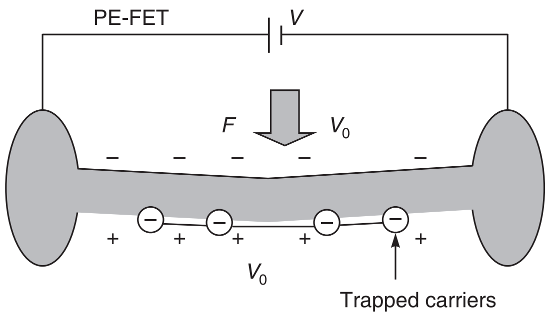


Figure 2.69 Nano pressure sensor.

When the nanowire bends, the stretched outer side of the bent wire becomes positively charged, while the compressed inner surface becomes negatively charged. The difference in charges creates a voltage that substitutes for the gate voltage. Zinc oxide is biocompatible and so one could implant the nanowire pressure sensor in the arm to monitor blood pressure continuously. The sensor could transmit the pressure reading to a receiver in one's watch, that will display the data. The project is at the laboratory stage right now, but offers exciting possibilities in future.

The concept could also be applied to other types of sensing. One use for the device could be as a biosensor. The principle is that molecules striking or sticking to the nanowires would deform the wire and change the current through it. It could also result in development of a chemical sensor, in which the chemical reaction disturbs the nanowire.

Chemical Sensors

Various nano tube-based gas sensors have been developed. A miniaturized gas ionization detector has been developed using CNTs (Carbon Nano Tubes). The sensor could be used for gas chromatograpy. Titania nanotube hydrogen sensors have been incorporated in a wireless sensor network to detect hydrogen concentrations in the atmosphere. Also, a chemical sensor based on nano tube molecular wire has been developed for gaseous molecules such as NO_2 and NH_3.

Touch Sensors

Scientists have developed a self-assembling nanoparticle device that has touch sensitivity comparable to that of the human finger, a capability far beyond that of any mechanical device now available.

The device consists of alternating monolayers—layers of gold nanoparticles 10 nanometres (10 billionths of a metre) in diameter and cadmium sulphide nanoparticles 3 nanometres thick, separated by alternating layers of polymers that act as dielectric barriers.

These are conductive and semiconductive materials (gold and cadmium sulphide). When you press on the device with an applied voltage across the thickness, it results in a larger current and electroluminescent light from the semiconducting particle. By focusing the emitted light intensity from the cadmium sulphide particles or the change in local current throughout the device, the amount of pressure applied may be known.

The touch resolution of the human finger is 40 microns (40 millionths of a metre). Using nanoparticles, we can attain resolutions close to human touch, which is about 50 times better than what is out there today.

2.15.4 Design Problems and Realities

Many of the design constraints for nanosensors are similar to those for microsensors (silicon sensors), notably interface requirements, heat dissipation, and the need to deal with interference and noise, both electrical and mechanical. Dealing with unwanted molecules and signals in very small systems often requires ancillary equipment and low-temperature operation to reduce noise. Furthermore, the very sensitive, tailored surfaces of these sensors are prone to degradation from the effects of foreign substances, heat, and cold. But the ability to install hundreds of sensors in a small space allows malfunctioning devices to be ignored in favour of good ones, thus prolonging a system's useful lifetime.

Although the excitement over nanotechnology and its prospective uses is generally well-founded, the development and integration of nanosensors must take into account the constraints imposed by physics, chemistry, biology, engineering and commerce. For example, as nanotechnologies are integrated into macro-sized systems, we'll have to provide for and control the flow of matter, energy, and information between the nano and macro scales.

2.15.5 Future of Nanosensors

Nanotechnology is certain to improve existing sensors and be a strong force in developing new ones. The field is progressing, but considerable work must be done before we see its full impact. Among the obvious challenges are reducing the cost of materials and devices, improving reliability, and packaging the devices into useful products. Nevertheless, we are beginning to see nano-scale materials and devices being integrated into real-world systems, and the future looks very bright indeed for technology on a tiny scale.

In either case, the combination of nano-scale top-down and bottom-up processes gives materials and device designers a wide variety of old and new tools. Designers can also combine microtechnology and nanotechnology to develop new sensor systems.

Although presenting a significant challenge, integration of nano scale technologies could lead to tiny, low-power, smart sensors that could be manufactured cheaply in large numbers. Their service areas could include *in situ* sensing of structural materials, sensor redundancy in systems, and size- and weight-constrained structures such as satellites and space platforms.

2.16 FUTURE TRENDS

The advancements in microelectronics technology are bringing forth new improvements in the silicon sensors. At present, the emphasis is placed on developing sensors which can be directly interfaced to microprocessors. These sensors (called digital sensors) incorporate signal conditioning unit as well as analog to digital converter unit on the same chip

Researchers are also working on the development of intelligent sensors in various organisations. Intelligent sensor comprises a microprocessor (also on the same chip) and would be capable of carrying on limited processing. Integrated sensor with the microprocessor on the same chip has certain disadvantages too. Such a sensor may not be suitable for hazardous applications. In a hot, humid or caustic environment if microprocessor is put along with the sensor then the whole system may start behaving erroneously. However, in such applications, transmission of data to microprocessors at the remote locations may also have transmission problems. Large-scale improvements are expected in this technology in coming years and ultimately it may become possible to develop an integrated sensor plus microcontroller on the same chip, specialised for a particular application.

Another area of development being pursued is “Expert System Integrated Intelligent Sensors”, i.e. Expert Sensor. Though it may take a number of years to achieve this goal, but with a provision of existing microprocessor, and AI technology, the goal looks feasible to achieve. Such an expert sensor would have on-chip signal conditioning circuit, powerful microprocessor and an expert system shell.

In the past several years, the development of optical sensors compatible with optical fibre transmission lines has become the goal of major research efforts. Much of these efforts have been directed towards fibre interferometric sensors, because they possess the following significant advantages:

1. Fibre interferometric sensors have potential for excellent sensitivity.
2. These sensors have unique automatic versatility.
3. These sensors have potential for detecting a variety of fields.
4. This capability makes multisensors fibre applications extremely attractive.

A typical configuration of a fibre interferometric sensor is illustrated in Fig.2.70. Laser beam is split into reference and sensing fibres. The sensing fibre is suspended in water and exposed to the field to be detected, while the reference fibre is in an enclosed environment isolated from the field. The field to be detected induces an optical phase shift in the sensing fibre, relative to the reference fibre. A modulator is incorporated in the reference arm to provide

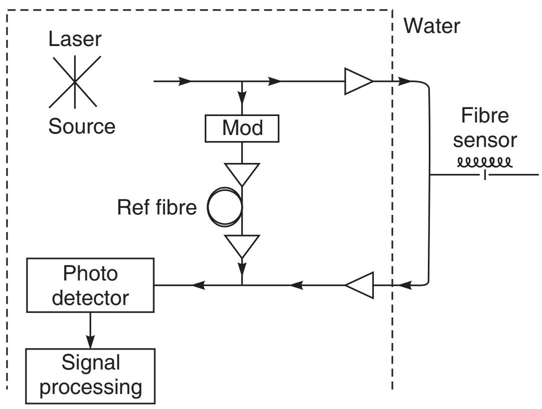


Figure 2.70 Fibre interferometric sensor.

frequency shift or modulation. Two beams are recombined on the beam splitter and detected by a photodetector. The signal is finally demodulated and with appropriate demodulation, fibre interferometric sensors can provide excellent sensitivity. The greatest advantage is the versatility as illustrated in Fig. 2.71. However, for the development of fibre interferometric sensors for various applications, more research on the demodulation techniques will have to be pursued. For an omni-directional element, the sensor can be designed into any shape so long as it is small compared to wavelength.

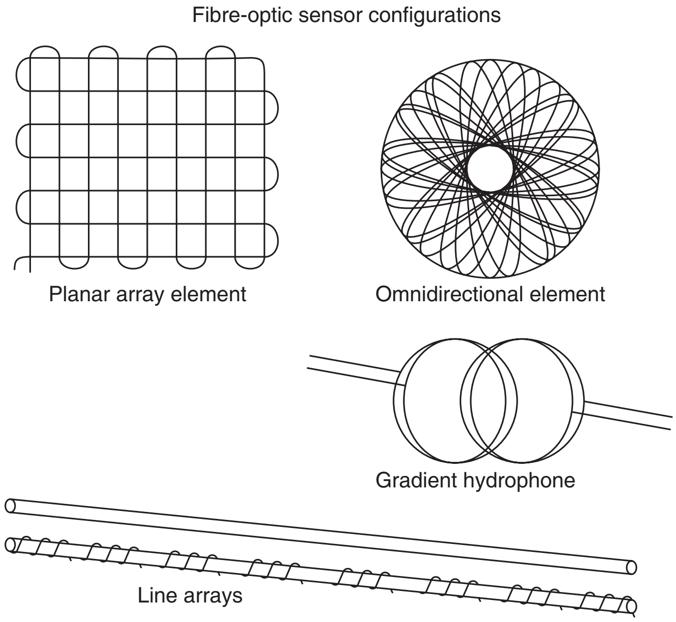


Figure 2.71 Potential geometries for fibre sensors.

It is quite clear that application of electro-optics and optical fibres for instrumentation is an important growth area in the immediate future. It is unlikely that sophisticated and expensive fibre optic systems operating well in scientific laboratory will be compatible to harsh environment of the process industries. What is required in fact is the availability of simple optical transducers with fibre-optic links. It is expected that research efforts in various parts of the globe will be directed towards this goal.

The biosensors are the product of fast emerging technology and it will take a few years before these are used widely in the industries. At present the limiting factors are robustness and shelf stability of biological sensing layers. The shelf instability problem, is overcome by storing the device under refrigeration which inhibits its use in industrial environment. A number of methods for immobilisation are under development to incorporate biological sensing element in membranes or gels, basically to increase the operational life of the devices. The main advantage of modified FET biosensors is its potential for producing a multisensor chip, incorporating an array of biologically sensitive devices with fully integrated signal processing and analysis circuitry. The FET device technology however suffers from problems of reliability, operating limitation and the fabrication.

It is expected that miniaturised conductimetric urea biosensors devices and associated instrumentation will find applications in blood analysis, urea determination and so on. The institutes which are actively involved in the development of biosensors worldwide are:

1. Institute of Chemical and Biochemical Sensor Research, Germany.
2. Technical University, Munich, Germany.
3. Cranefield Biotechnology Centre, Cranefield, UK.
4. Norwegian Water Technology Centre, Norway.
5. University of Lund, Sweden.
6. Institute for Technical Biochemistry, University of Stuttgart, Germany.
7. IGEC Marconi Material Technology, Casewell, UK.
8. University of Liverpool, UK.
9. *Centre de Recherche Public Hennri Tudor*, Luxembourg.

SUGGESTED READING

Allocca, John A., and Allen Stuart, *Transducers*: *Theory and Applications,* Reston Publishing Company, Virginia, 1994.

Bank, Walter, Expert systems for sensor use, *Sensors*, 2, No. 6, May 1984.

Cole, J.H., et al., Research update on fibre-optic sensors, *Int. Fiber-optic Communication*, March 1991.

Considine, Douglas M., *Encyclopedia of Instrumentation and Control*, McGraw-Hill, New York, 1971.

Cui, Y., et al., Nanowire nanosensors for highly sensitive and selective detection of biological and chemical species, *Science*, Vol. 293, pp. 1289–1292, Aug 17, 2001.

Dakin, John, and Brian Culshaw, *Optical Fiber Sensors*: *Principles and Components*, Vols. I and II, Artech House, Massachussetts, 1988.

Herschy, Reginald W., *Stream Flow Measurement*, E & FN spon (an imprint of Chapman and Hall). 1995.

Hutten, H., New developments in biosensors, *Elektrotechnik and Informationstechnik*, 109, No.5, 1992.

Information catalogues from M/s Philips, Kulite, Motorola, Fox Boro, Analog Devices and Gould, USA, 1985.

Karube, I., and Tamiya, E., Biosensors for environmental control, *J. Pure & Appl. Chem*, 59, 1987.

Kong, J., et.al., Nanotube molecular wires as chemical sensors, *Science,* pp. 622–625, Vol. 287, Jan 28, 2000.

Krishna Kant, *Microprocessor-based Data Acquisition System Design*, Tata McGraw-Hill, New Delhi, 1987.

Loeppert, Peter V., New capacitance pressure sensor minimizes thermal effects, *Instrumentation and Control Systems*, Feb. 1992.

Lowe, C.R., An introduction to the concepts and technology of biosensors, *Biosensors*, I, 1985.

Mansfield, P.H., *Electrical Transducer for Industrial Measurement*, Butterworth & Co., London, 1973.

Marshman, C.E., Electronic biosensors, *Phys. Bull.*, 37, 1986.

Meter, Lori Van, A bright future for fibre-optic sensor systems, *Int. Fiber optic Communication*, March 1981.

Middel, Hock, and A.C. Hooger Werf, Smart sensors: Where and when, *Proc. Transducers'85*, IEEE, 2, No.7, 1985.

Miller, Richard K., and Terri C. Walker, *Artificial Intelligence Applications in Sensors and Instrumentation*, SEAI Technical Publications, 1988.

National Semiconductor, Data Acquisition Linear Devices Data Book, 1989.

Nelson, Arthur R., Passive multiplexing techniques for fibre-optic sensor system, *Int. Fiber-optic Communication*, March 1981.

Ngeh-Ngwainbi, J., Suleiman, A.A., and Guilbault, G.G., Piezoelectric crystal biosensors, biosensors & bielectric crystal biosensors, *Biosensors & Bioelectronics*, 5, p. 136, 1990.

Norton , Harry N., *Handbook of Transducers*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

Parr, E.A., *Industrial Control Handbook, Vol 1: Transducers*, Industrial Press, New York, 1987.

Proceedings of 2nd National Seminar on Physics and Technology of Sensors, University of Poona, India, Feb. 1995.

Rechnitz, G.A.,Biosensors: An overview, *J. Clin. Lab. Anal.*, 21, 1987.

Regers, K.P., and Lin, J.N., Biosensors for environmental monitoring, *Biosensors & Bioelectronics*, 7, 1992.

Roger, Allen, New applications open up for silicon sensors: A special report, *Electronics*, 6, Nov. 1980.

——, X-shaped sensor simplifies transducer, *Electronics*, 22, Nov. 1979.

Schmid, R.D., *Nachr. Chem. Tech. Lab.*, Biosensors, 35, 1987.

Second European Workshop on Biosensors for Environmental Monitoring, Kings College, London, Feb. 1994.

Sethi, Rajender S., Transducer aspects of biosensors, *Biosensors & Bioelectronics*, 9, 1994.

Small Wonders, Endless Frontiers: *A Review of the National Nanotechnology Initiative*, National Academy Press, 2002.

Turner, A.P.F., Karube, I and Wilson, G.S. (Ed.), *Biosensors: Fundamentals and Applications*, Oxford University Press, UK, 1987.

Van der Berg, A., Koudelka-Hep, M., Van der Schoot, B.H., and De Rooij, N.F., From biochip to integrated biochemical analysis system, *IEE Colloqium On Biosensors*: *A Viable Monitoring Technology* (Digest No. 145), 1991.

Vo-Dinh, T., Cullum, B.M., and Stokes, D.L., Nanosensors and biochips: Frontiers in biomolecular diagnosis, *Sensors and Actuators B*, Vol. 74, pp. 2–11, 2001.

Yazdi, N., Ayazi, F., and K., Najafi, Micromachined inertial sensors, *Proc. IEEE*, Vol. 86, No. 8, Aug. 1998.

CHAPTER

3

Building Blocks of Automation System

3.1 INTRODUCTION

The development in the field of automation and in the field of intelligent machines (i.e. starting from computers, microprocessors to present day expert systems and neural networks) were almost simultaneous. It is a known fact that growth in computer and microprocessor technology was one single big cause for the growth in automation techniques like DDC, Distributed control, Adaptive control etc. In this chapter we shall be introducing the building blocks and systems that are used in automation system. We begin by introducing the computers and microprocessor advancements. The concepts of microprocessors, the facilities and advancements have been discussed briefly. The actual pins and signals, bus cycles, interfacing etc. have been avoided. The multiprocessor structures and local area network (LAN) structure have been described next. The multiprocessor structures interfaced in tightly coupled common memory structure or in loosely coupled local area networks play very important role in distributed control or control involving large computation requiring high performance microprocessors. The other important building blocks of any control system, i.e., analog and digital I/O modules will be discussed in this chapter followed by Supervisory Control and Data Acquisition (SCADA) system and Remote Terminal Units.

3.2 PROCESSLNG SYSTEM

3.2.1 Computers and Microprocessors

The computers are predecessors to microprocessors. The basic concepts of computers were evolved before the dawn of microprocessor. The same concepts were extended in microprocessors to provide single chip CPU function in a microcomputer.

By the early 1970's, small integrated circuits (TTL Logic) were well established while MOS integrated circuits, such as calculator components had started to appear. The use of a minicomputer or control processor (as against the use of general purpose computers) was also well known. It was clear to some semiconductor engineers that if the calculator chip could become more general, it would have wider application. Also, the mini-computer users were confident that if it could be made more compact and cheaper its application areas will become further wide. These were the two mainstreams that led to the microprocessor development. Undoubtedly, what made the microprocessor possible was MOS technology and the remarkable properties of silicon, providing that, the microprocessor was inevitable.

Figure 3.1 shows the block diagram of computer showing different units and is known as Von Neumann Organisation of Computer. The computer organisation proposed by Von Neumann envisages that binary number systems are used for both data as well as instructions. There is a direct correlation between the microprocessor organisation and the organisation proposed by Von Neumann for computers. Due to advancement in micro-miniaturisation, the arithmetic logic unit and control unit have been put on single chip and this chip is known as microprocessors. Figure 3.2 shows the computer organisation with microprocessor chip, memory, input and output devices. The data bus will carry the data information and the control bus carries the control information like I/O Read, I/O Write, Memory read, Memory write, etc. A generalised microprocessor structure has been shown in Fig. 3.3. A microprocessor will contain arithmetic logic unit (ALU), timing control unit, a number of scratch pad registers, stack pointer, program counter etc. Stack can be maintained in the memory along with programme and data area. The input/output units can be interfaced through I/O ports.

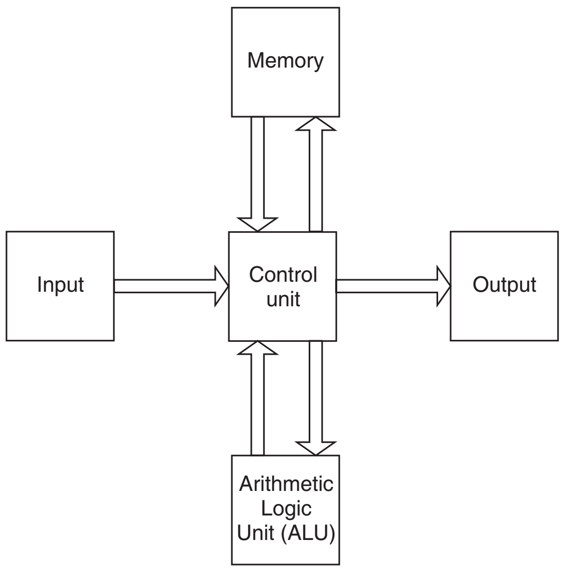


Figure 3.1 Von Neumann organisation of computer.

The idea here is not to describe the internal details of microprocessors, but to make readers aware of microprocessor structure and its advancements.

Every microprocessor will offer the following facilities:

(i) *Interrupts.* Interrupt is a facility provided by the microprocessor to the outside environment by which attention of microprocessor can be diverted to do some higher priority job.

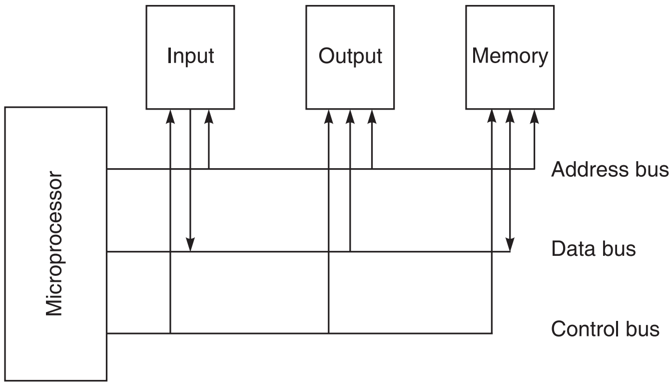


Figure 3.2 Computer organisation with microprocessor chip.

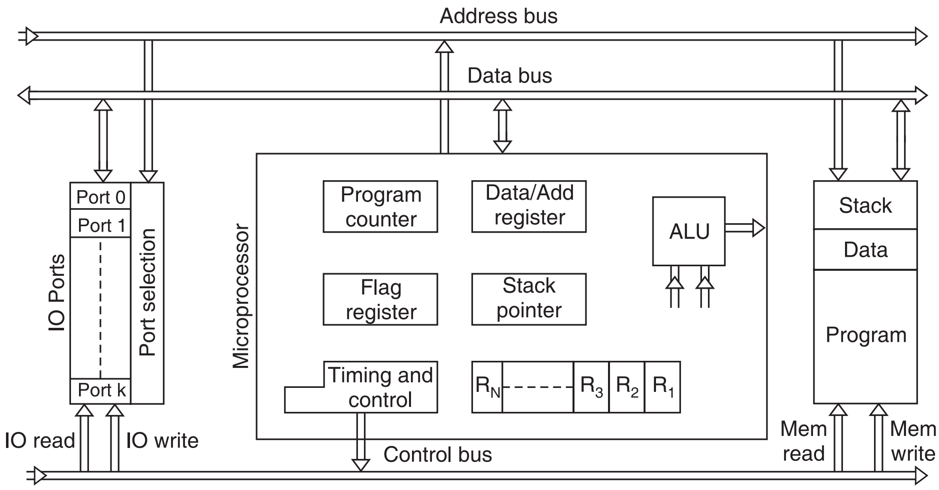


Figure 3.3 A generalised microprocessor structure.

The interrupts are used for various purposes in different environments. Microprocessor can be interrupted to initiate data transfer, to execute control sequence, to control large power plants, or to check the status of a process at any particular instant. A number of interrupt request pins are provided by each microprocessor for this purpose.

(ii) *Direct memory access.* In order to facilitate the fast input/output devices for data transfer, microprocessor provides the facility by allowing them to transfer the data directly to or from memory. Microprocessors offer the facility through two pins—DMA Request and DMA Acknowledgement.

(iii) *Serial data transfer*. Some devices like teletype serial printers etc. are serial in nature. In addition to this, for long communication, serial communication is used to avoid skewing of bits. To minimise the number of interconnected wires, the data is transferred bit by bit on single line in serial communication. The microprocessors providing serial data transfer facility will have two pins for input and output of serial data and special software instructions to effect the data transfer.

3.2.2 Architectural Advancements of Microprocessors

A number of advancements that had taken place in the computer have migrated to microprocessor field with the continual advancement of microelectronics technology. The concept of multitasking, pipelining, and multiprocessing are already there in latest microprocessors. The pipelined microprocessors have a number of smaller independent units connected in series like a pipeline. Each unit is able to perform its tasks independently and the system as a whole is able to give much more throughput than the single microprocessor. The multitasking provides the environment in which the microprocessor can execute multiple tasks simultaneously by cycle stealing. The concept of virtual memory increases the memory capacity beyond the physical memory space possible through width of address bus. The Memory Management Unit (MMU) is now integrated on microprocessor chips. Thus microprocessors of 90s are equivalent to supercomputers of 80s and the process of growth is going to continue.

In the following sections, we shall discuss the evolution of microprocessors very briefly.

3.2.3 Evolution of Microprocessors

Typical of early microprocessors were Intel 4004, Rockwell PPSK, Burroughs Mini-d and Fairchild PPS-25, which were all calculator-based chips with emphasis on arithmetic operations. These had surprisingly good internal facilities (such as on-chip registers) but little emphasis was given to speed. The *P*-Channel MOS logic at that time was considerably slow in itself and the microprocessors were given a very small package (e.g. 16 pins) which required multiplexing of data streams.

8-bit microprocessors

The development in 8-bit microprocessor is due to the transition of microelectronics technology from LSI to VLSI technology and is based on *N*-channel metal oxide semiconductor work. With the high mobility of negative charged carriers (electrons) in *N*-channel MOS, higher logic speeds and greater packaging density than that in *P*-channel MOS was achieved.

Performance was improved by using a separate data and address bus with enough address bits for large amount of memory. On-chip registers addressed by the register addressing mode helped to speed up the programs. Instruction sets became more sophisticated with good arithmetic, data transfer and control facilities. The interrupt facility became standard.

With N-MOS, the microprocessors were able to drive the rest of the system without extra TTL chip, thus reducing the components count. Led by the INTEL 8080, every 8-bit microprocessor has its own peculiar characteristics.

Intel 8080, 8085

The 8080 microprocessor (Fig. 3.4) represented a dramatic improvement over 8-bit P-MOS 8008 microprocessor. The 8080 has a separate data and address bus but control signals are multiplexed on the data bus, requiring external latches to extract them. There are several on-chip registers but very few instructions can address the main memory directly; addressing being mostly register indirect. There are separate set of instructions for register, memory and input/output. Execution speed is quite high. There are useful interrupt and subroutine facilities. It requires external clock generator circuit and three supply voltages. The architecture reflected a switch to a parallel bus organisation though the system interface was still not clear. Since 8080 emerged earlier than other microprocessors, it became very popular and was taken as industry standard.

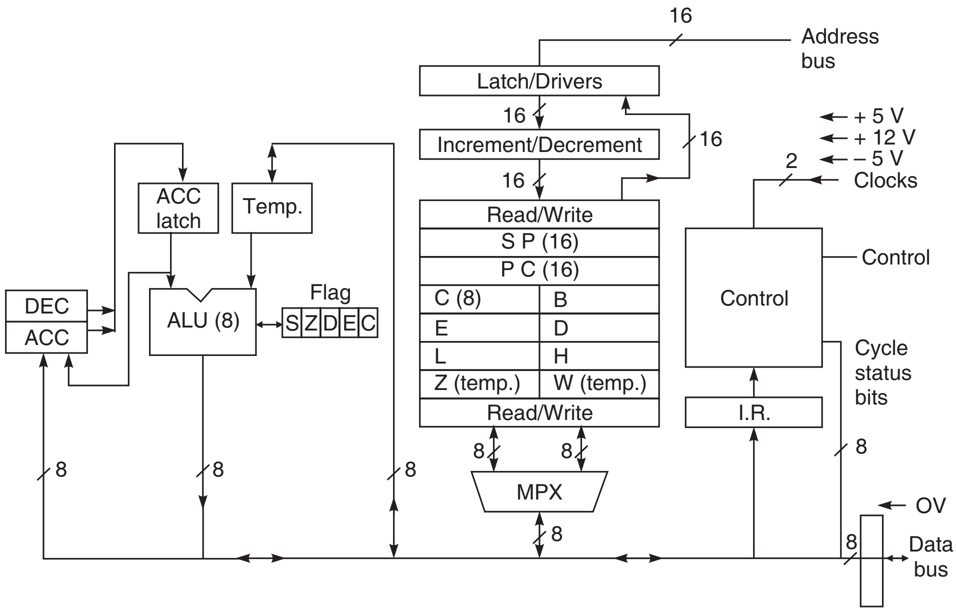


Figure 3.4 Intel 8080 architecture.

The 8085 improved upon the 8080 for it needed only one +5V supply instead of three different voltages. The clock generator is integrated on-chip and the control lines need not be demultiplexed from the data bus. There are multiple interrupts provided on-chip. Yet the instruction set is almost unchanged and little extra speed is offered. On-chip serial input/output facility is present in addition. It has therefore been aimed to make a system easier and cheaper to build rather than to give higher processing performance.

Zilog 80

The *Z* 80 microprocessor was designed by the same team who created the 8080. It has twice as many CPU registers as the 8080 and many more instructions. In other ways its advantages over the 8080 are similar to the 8085's. The instruction set has more addressing modes which facilitates the development of shorter programs of greater speed. The designers have tried to make the instructions which appear most often in the program, to take up the least space and those which are executed most often the fastest

Motorola 6800, 6802 and 6809

The approach adopted by Motorola for the development of their 8-bit microprocessor was quite different. The microprocessor has two accumulators but no general purpose data register on-chip. A parallel data bus concept was also adopted. All I/O devices are interfaced in memory mapped I/O. Good addressing modes (direct, page, indexed) and single level addressing (memory and I/O read/write use same instruction due to memory mapped I/O) made the program easy to learn. The fastest execution time (2 micro second) was similar to the 8080, but only a single low level clock and one +5 V supply voltage was required.

Later versions of 6800 (68A00, 68B00), are much faster than earlier versions because of changes in technology. The fastest instruction executes in 1 micro second (μs).

The 6802 has an on-chip RAM of 128 bytes but is slower. Otherwise, it is similar to 6800 with a clock oscillator on-chip.

Although an 8-bit machine, 6809 (Fig. 3.5) has many 16-bit characteristics. It has been made upward compatible with 6800. The CPU architecture, registers, and instruction set are extensions of 6800 format. The two accumulators of 6809 can be used together for 16-bit operations. There are four index registers (unlike only one in 6800) of which two are also used as stack pointers (U is the upper stack, S is for subroutine/interrupt stack). There is a page

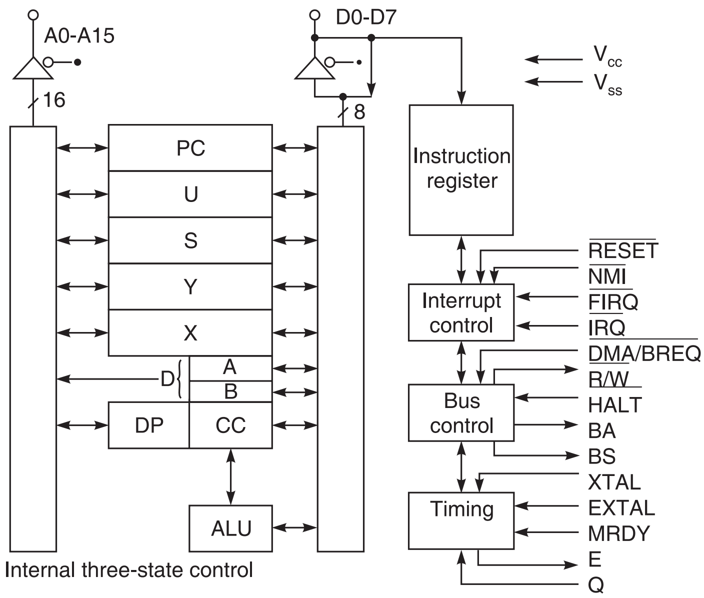


Figure 3.5 MC6809 block diagram.

register (DP register) for short branch instructions. The addressing modes are numerous and useful and include variations on direct, indexed, indirect indexed etc. There is an 8-bit multiply instruction and 16-bit add, subtract, store and compare instructions. The 6809, therefore represents an attempt to offer an enhanced 8-bit microprocessor which is economical to use.

16-bit microprocessors

The advent of the 16-bit microprocessor was marked with certain new concepts like pipelining, virtual memory management, etc. Powerful addressing modes were evolved. Multiply and divide instructions were also introduced. The era of 16-bit microprocessor began in 1974 with the introduction of PACE chip by National Semiconductor, and CP 1600 by General Instruments. Several powerful microprocessors have been developed since then.

Intel 8086, 80186 and 80286

The 8086 (Fig. 3.6) was the first of the new breed of high performance 16-bit microprocessors. The HMOS technology allowed over 28,000 transistors to be used in the design and gave it a high speed. Memory components in HMOS are also very fast, down to less than 100 ns access time. It achieves its speed without needing fast memory components or even a separate data and address bus. The CPU consists of two parts, namely Execution Unit and Bus Interface Unit. These parts act independently to achieve high speed. The bus interface unit maintains a queue of six instructions for execution unit. The memory is divided into four segments of 64 kilobytes each. The 20-bit address bus allows 1 million bytes of memory. The use of four segment registers (one for each segment) allows the program modules to be placed anywhere in the memory. 8086 also has two 16-bit pointers, two index registers and four 16-bit general purpose registers.

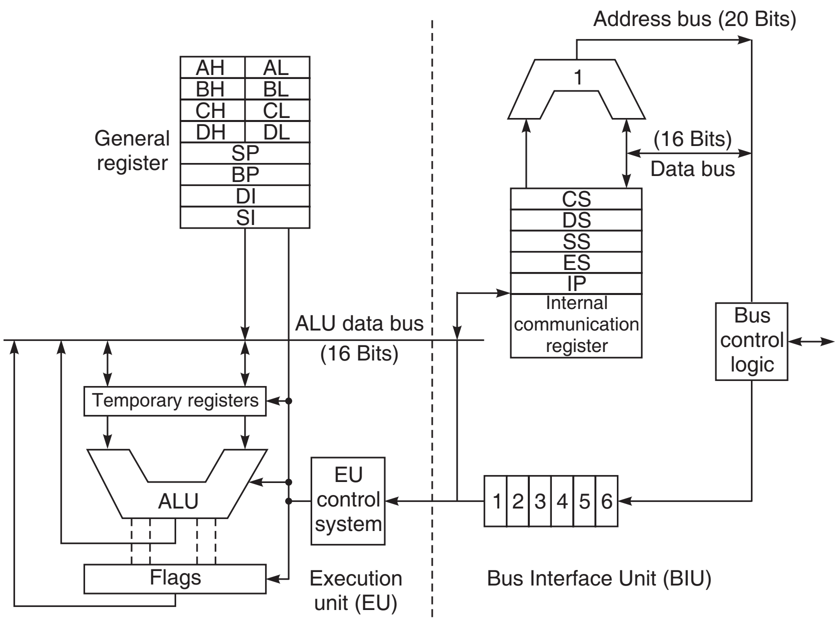


Figure 3.6 8086 block diagram.

Powerful addressing modes and instruction set have been evolved. The microprocessor uses two address instruction and operation may take place between two registers, register and memory, register and immediate or memory and immediate. Memory addresses can be direct, indirect (via base or index register) or indexed (via base or index register). The processor can be operated in minimum or maximum mode by wiring a pin to +5V or ground.

The maximum mode allows multiprocessor environment. The 8086 instruction set is upward compatible to 8080 and 8085 with exception to RIM and SIM instructions. This compatibility is at the source level only.

The 8088 processor was introduced later than 8086. The internal structure of 8088 microprocessor is the same as its predecessor. The difference however lies in data bus which is 8-bit wide in case of 8088. The 8088 was introduced for systems which were originally designed around 8080 or 8085 and needed upward compatibility. The 8088 architecture is shown in Fig. 3.7.

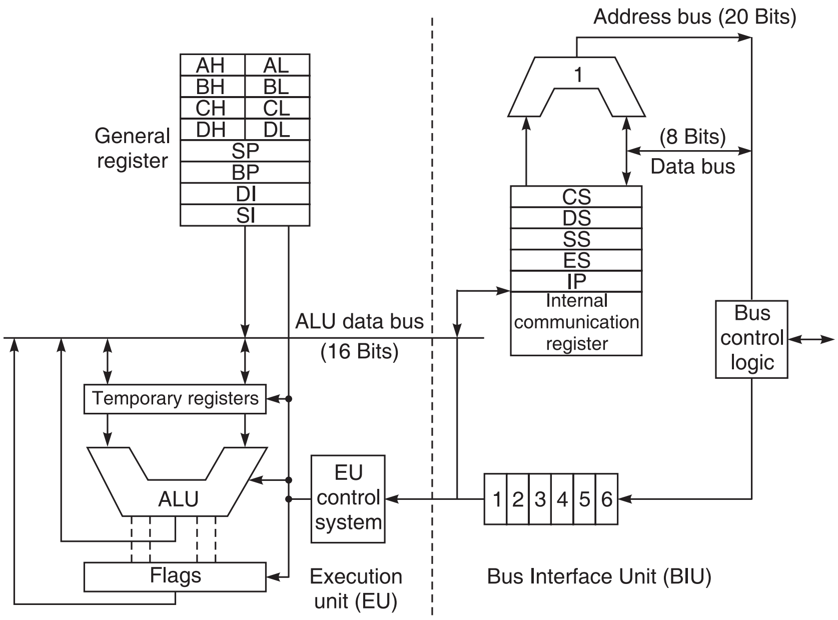


Figure 3.7 8088 elementary block diagram.

The Intel 80186 (Fig. 3.8) introduced in 1982, offers twice the performance of standard 8086 and offers 12 additional instructions. Integrated on the chip are clock generator, DMA controller with two independent channels, three programmable 16-bit timers, 8086 CPU (8 MHz version), etc. The multi-CPU configuration can he achieved through HOLD and HLDA.

The Intel 80286 also introduced in 1982, offers still higher performance, up to 6 times that of 8086. It has on-chip 10 MHz processor (8086), memory management unit with four level memory protection and support for virtual memory and operating system. It supports 16 megabytes physical and 1 gigabytes virtual memory. The 80286 uses a superset of 80186 instruction set with sixteen additional instructions. This processor is specially designed for multiuser and multitasking systems.

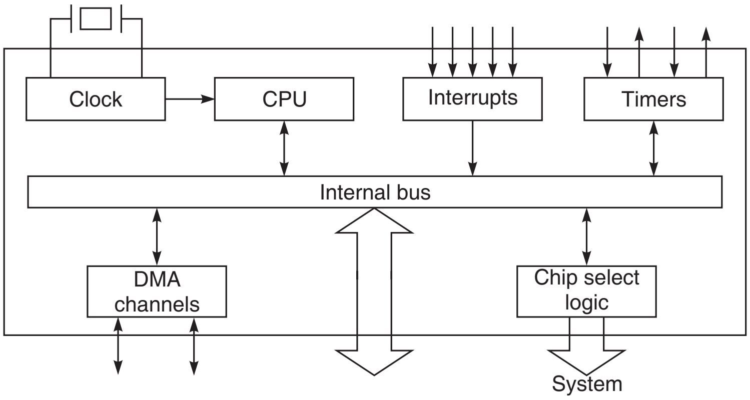


Figure 3.8 Intel 80186 architecture.

Zilog 8000

By using VLSI techniques, Zilog have managed to pack an extremely powerful 16-bit microprocessor on a single NMOS silicon chip. Two basic versions of Z8000 are being produced. Z8001 a segmented addressing system allows access to 23-bit address bus and permits up to 8 million 16-bit words or 16 megabytes of memory to be used. Z8002 is non-segmented version with 64 kilobytes memory. It is possible to use Z8000 in multiprocessor environment. Z8000 does not have a dedicated register for use as the accumulator. Instead it uses a bank of sixteen general purpose 16-bit registers any of which may be used as accumulator. Memory in the Z8000 system may be divided into areas for system and user and also into separate data and program areas which are all defined by status control lines from the processor. The main addressing modes provided by Z8000 are register, indirect register, direct, immediate, indexed, relative, base address and base indexed. There are 110 basic instruction types which may be executed on various modes to give over 400 different types of operations.

Motorola 68000

The MC68000 (Fig. 3.9) is quite different from other 16-bit processors in many respects. Though it is a 16-bit microprocessor, it has largely 32-bit wide data organisation and a flexible array which gives it a very high processing throughput. Hardware multiplication and division logic are included on the chip. This further increases the processing speed in complex calculations. Twenty three address bits provide 16 megabytes of direct addressable memory space. No dedicated accumulator has been provided, instead it uses a bank of eight 32-bit general purpose registers D0–D7, of which any register can be used as accumulator. The system is able to handle 32-bit operations with ease.

Fourteen different basic addressing modes are provided on the MC68000. There are 56 basic instructions type in MC68000 instruction set. Combined with the addressing modes, they form a powerful instruction repertoire.

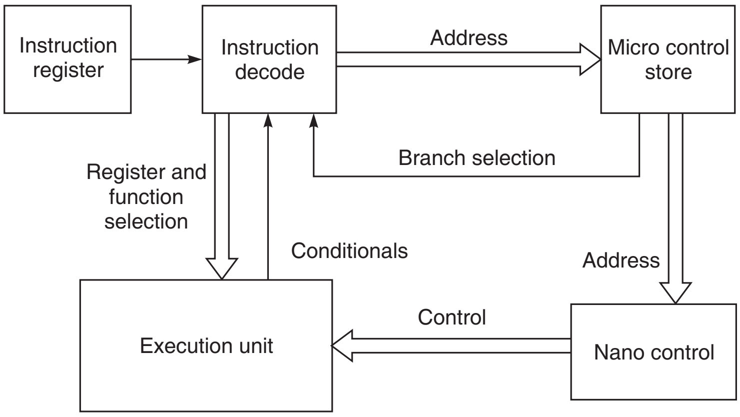


Figure 3.9 Motorola 68000 architecture.

32-bit microprocessors

The era of 32-bit microprocessor began in 1981 with the introduction of iAPX 432. In 1980 IBM implemented IBM 370 CPU on a single chip but since it is not offered commercially, this is generally overlooked. Other 32-bit processor are Belmac-32A microprocessor from Bell Labs, the 32-bit CPU chip from Hewlett Packard, INTEL iAPX series, Motorola 68020 and 68030 etc.

Intel iAPX 386

80386 is 32-bit member (Fig. 3.10) of iAPX 86 family. It is software compatible to 8086, 8088, 80186 and 80286. New concepts like caching, pipelining are provided along with high

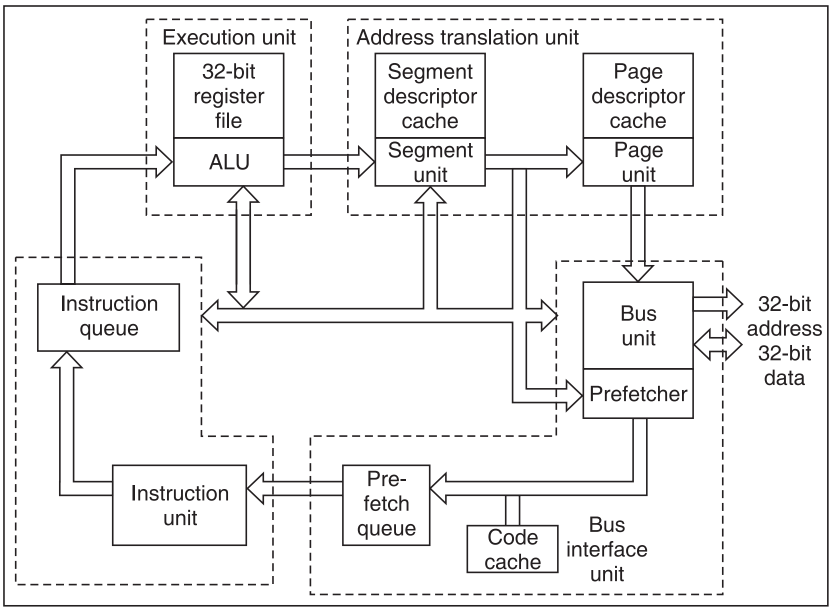


Figure 3.10 iAPX 386 pipeline.

performance bus, and high speed execution unit. The 80386 provides two to three times the performance of 80286. It has pipelined architecture with parallel fetching, decoding, execution and address translation inside the CPU. It provides full 32-bit architecture and internal implementation including 32-bit register file, instruction queue, address translation unit, address bus and data bus. It has hardware supported multitasking and virtual memory support. The physical memory up to 4 gigabytes can be addressed whereas the virtual memory up to 64 terabytes can be addressed per task. It has hardware enforced protection up to four levels to provide protection of sensitive code data within a task. The general purpose registers of 80386 support 32-bit data and addressing. They also provide for 8 and 16-bit compact addressing.

Motorola 68020, 68030

Motorola 68020 (Fig. 3.11) is a 32-bit virtual memory processor. It has fast on-chip instruction cache to improve execution speed and bus bandwidth. It is object code compatible to 68000. The pipelined architecture has high degree of internal parallelism, which allows multiple instructions to be executed concurrently. The processor has sixteen 32-bit data and address registers, and supports 18 addressing modes and 7 data types. Four gigabytes memory can be directly interfaced. The clock frequency can be 12.5, 16.67, 20 or 25 MHz.

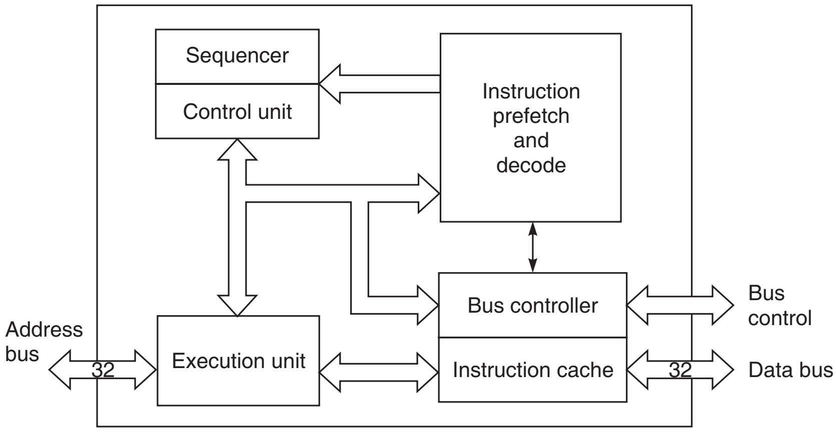


Figure 3.11 68020 block diagram.

Motorola 68030 (Fig. 3.12) is second generation 32-bit enhanced microprocessor based on 68020 core. It is object code compatible with 68020 and 68000. The paged memory management unit translates addresses in parallel with instruction execution. The processor contains 256 bytes instruction cache and 256 bytes data cache. The clock frequency can be 16.67 or 20 MHz.

Intel iAPX486

iAPX486 is an advancement over iAPX386 microprocessor series of Intel. It contains pipelined structure having arithmetic logic unit, cache unit, bus interface, instruction decode, prefetch check and floating point unit. It is upward compatible with 8086 and is widely used (Fig. 3.13).

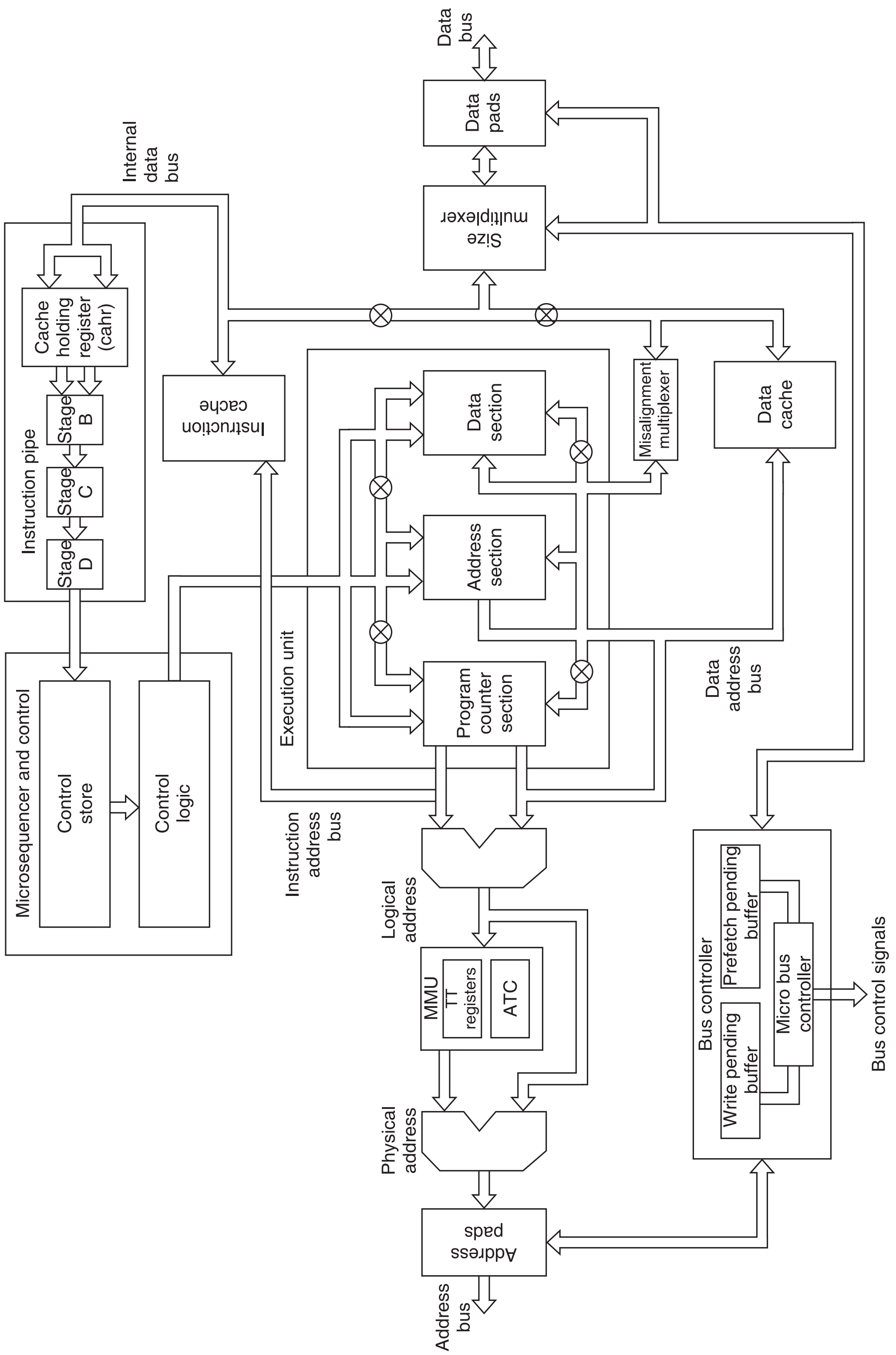


Figure 3.12 68030 block diagram.

Intel's pentium processor

The pentium processor, the newest and the most powerful member of Intel's X86 microprocessor family, incorporates features and improvements made possible by advances in semiconductor technology. A super-scaler architecture (Fig. 3.14), improved floating point unit, separate on-chip code and write back data cache, a 64-bit external data bus and other features like branch prediction, multiprocessing support etc. provide platform for high performance computing.

Intel i860

It is 64-bit microprocessor that delivers the kind of power and capability associated with super computers. It integrates super computer features like 64-bit architecture, parallelism and vector processing and takes full advantage of advanced design techniques like reduced instructions set computing, pipelining, score boarding, by-passing, delayed branching, caching and hard-wired 3-D graphic instructions. It incorporates a risk integer unit, a floating point unit, a 3-D graphic processor, data and instruction cache, memory management and a bus control unit (Fig. 3.15).

Bit-slice processor

AM2901 bit slice processor is available in a slice of 4 bits. Depending on the word size required, more than one slices can be connected to form a higher bit processor, i.e. 8-bit processor can be obtained by connecting two slices, 12-bit processor by connecting three slices, and so on. The IC chip is based on bipolar technology. Thus these processors are faster than processors based on N-MOS technology. User level microprogramming is offered in bit slice processor. These are most suited for special applications.

3.2.4 Microcomputers and Microcontrollers

Microcomputers are microprocessors with on-chip memory. Some microcomputer chips contain timer/counter, interrupts handling, also along with processor and memory Timer/counter and interrupts are useful for control application and these microcomputers are called *microcontrollers*. In some cases analog to digital converter and digital to analog converter have also been integrated on the chip. There is a variety of microcomputers/microcontrollers from different manufacturers like 8048, 8051 and 8096 series of Intel, Z8 from Zilog, M6801 and 68HC11 from Motorola, 1650 series from General Instruments, IM6100 from Internal Inc. etc. In general, these chips come in three versions namely "on-chip ROM version", "on-chip EPROM version" and "ROM-less version". The later two versions are used for development purposes. After the development is complete, the large number of 'on-chip ROM' version chips can be obtained by getting the program fused at source, at low cost.

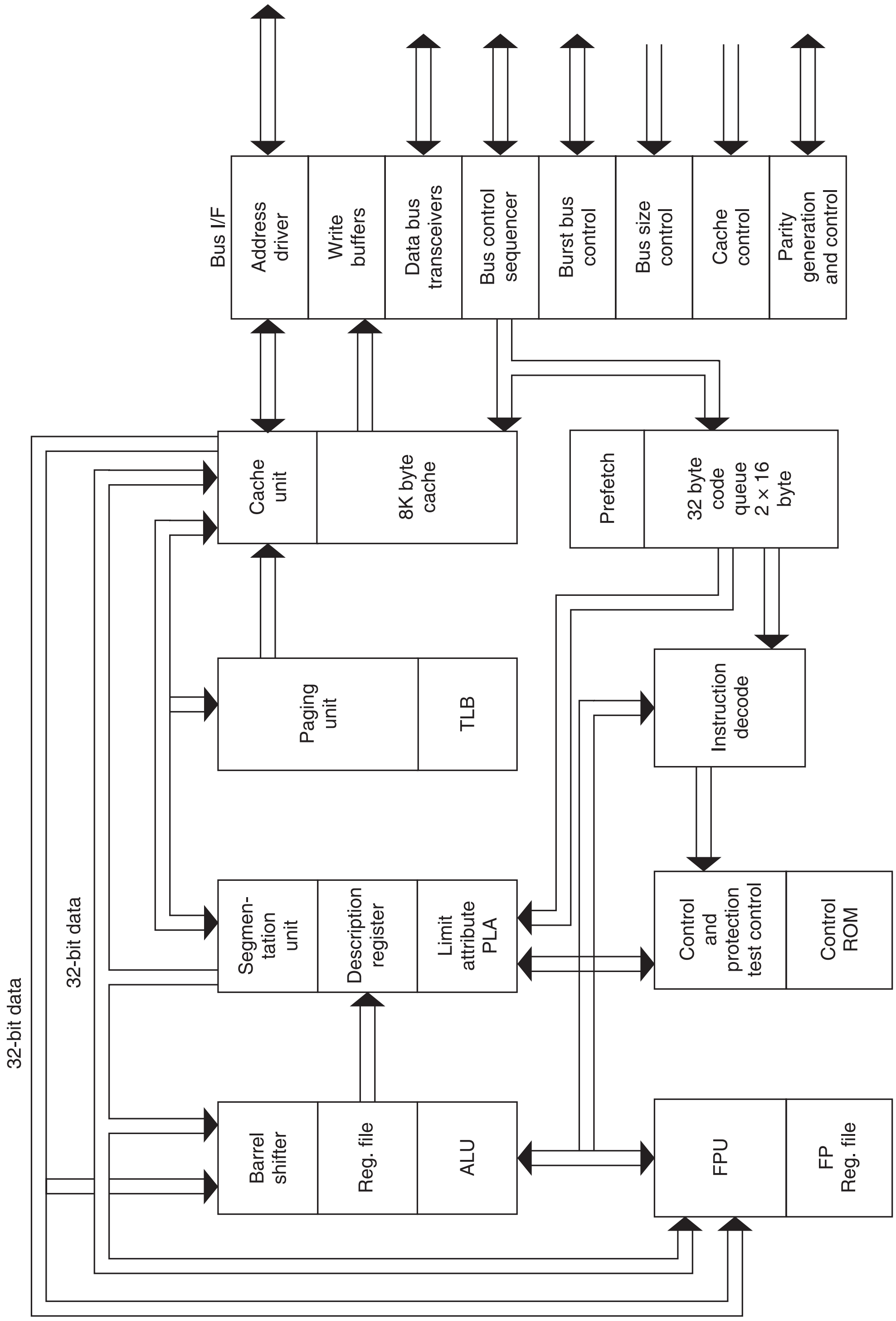


Figure 3.13 Intel 80486 block diagram.

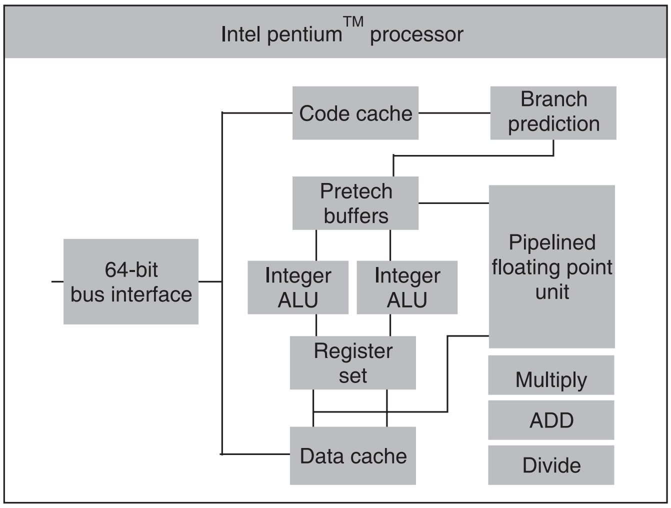


Figure 3.14 Intel's pentium processor architecture.

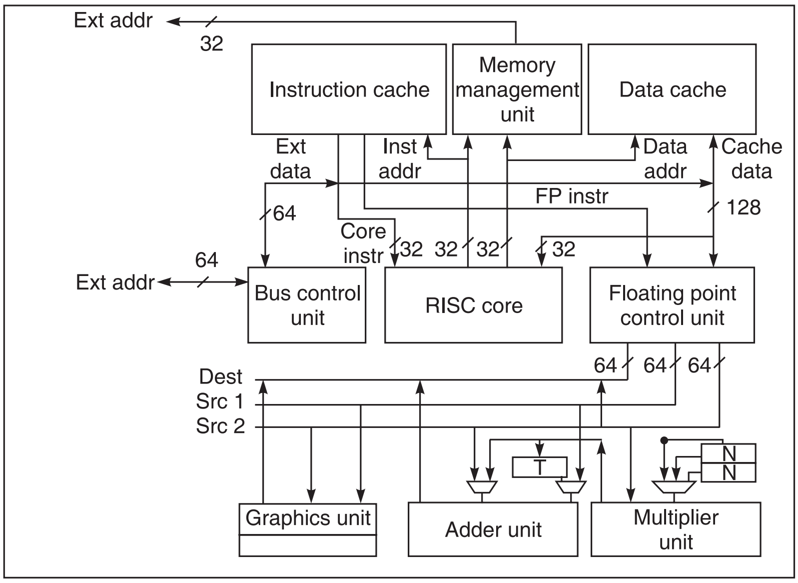


Figure 3.15 Intel i860 block diagram.

Intel 8051 series

Intel 8051 microcontroller (Fig. 3.16) is an 8-bit microcontroller with on-chip 8-bit CPU, 4 kilobytes of RAM 21 special function registers, 32 I/O lines and two 16-bit timer/counter. It offers 64 kilobytes address space for external data and 64 kilobytes of address space for external program memory, a five source interrupt structure with two priority levels, a full duplex serial port and bit address capability for boolean processing. Intel 8031 is a ROM-less 8051 and 8751

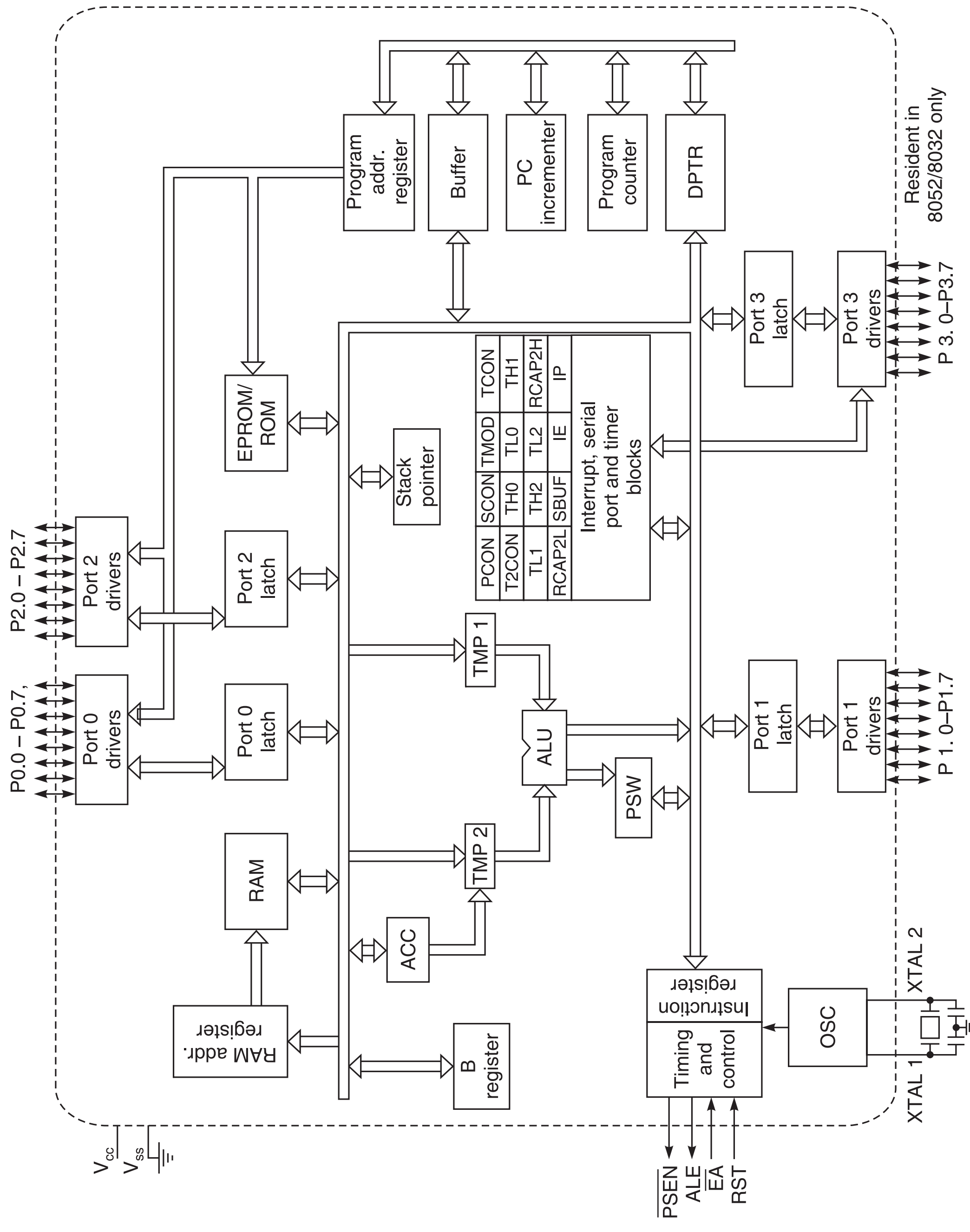


Figure 3.16 Intel MCS-51 architecture.

is an 8051 with EPROM instead of ROM. Software instruction include powerful multiplication, division, bit set and bit test operations.

Intel 8096 series

16-bit microcontrollers of (MCS-96 series) (Fig. 3.17) of Intel are extensions of 8051. CPU supports bit, byte and word operations including 32-bit double word operation. Four high speed trigger inputs are provided to record the time at which external events occur, six high speed pulse generator outputs are provided to trigger external outputs at preset times. The high speed output unit can simultaneously perform timer functions. Up to four such software timers can be in operation at one time.

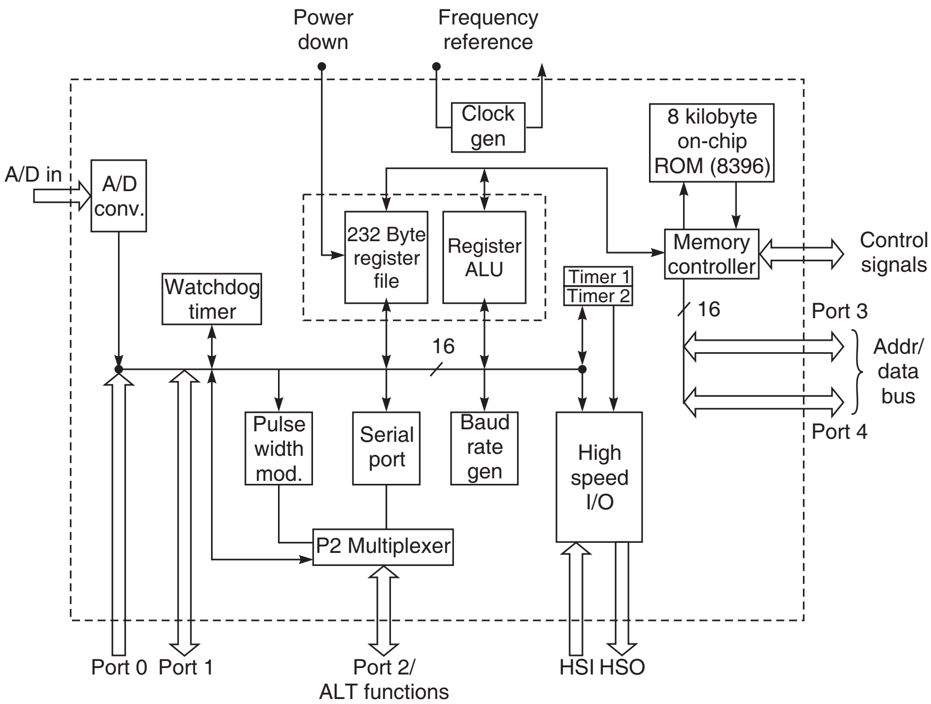


Figure 3.17 MCS-96 architecture.

An onchip A/D converter converts up to 4 (8095, 8395) or 8 (8097, 8397) analog input channels to 10-bit digital value. Also provided on chip are a serial port, a watch dog timer and a pulse width modulated output signal. 8 kilobytes of on-chip RAM is available in case of Intel 8396, 8394, 8397 and 8395 whereas 8096, 8094 and 8095 are ROM-less version. As stated earlier A/D conversion is available only with 8095, 8395, 8097 and 8397.

Motorola 68HC11

The Motorola 68HC11 is an 8-bit microcontroller (Fig. 3.18). It has internal 16-bit address bus. It has at least 512 bytes of EEPROM and is available in more expensive versions with either 2K

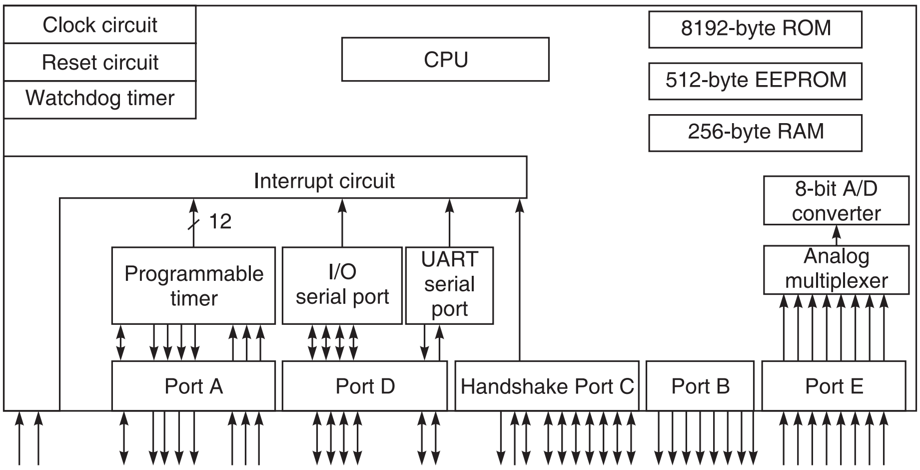


Figure 3.18 Motorola 68HCll microcontroller architecture.

or 8K of EEPROM. The motorola 68HCll has five parallel ports. Any line not serving the specialised alternate as shown in figure, can serve a more general functions. The other facilities include 8 channel, 8-bit ADC, serial port, programmable timer, UART port etc.

3.2.5 The Transputer

The principle behind the design of the transputer is to provide the system designer with a building block component which can be used in large numbers to construct very high performance systems. The transputers have been specifically developed for concurrent processing. The on-chip local memory assists in eliminating processor to memory bottlenecks and each transputer supports a number of asynchronous high speed serial links to other transputer units. The efficient utilisation of processor's time slices is carried out by a micro coded scheduler.

The transputer to transputer links provide a combined data communications capacity of 5 megabytes/sec. and operate concurrently with internal process. This is a radical difference from the shared bus concept employed in the majority of multiprocessor architectures. It allows parallel connection without overhead because of the complex communication between conventional parallel processors. The advantages over multiprocessor buses are as follows:

1. No contention for communication.
2. No capacitive load penalty as transputers are added.
3. The bandwidth does not become saturated as system increases in size.

The system supports high level concurrent programming language Occam specifically designed to run efficiently on transputer systems. The Occam allows access to machine features and removes the need for a low level assembly language.

Figure 3.19 shows the architecture of IMS "T-800 T-30" transputer. The main features of this chip are (a) Integral hardware 64-bit floating point unit, (b) 2.25 sustained megaflops/sec.,

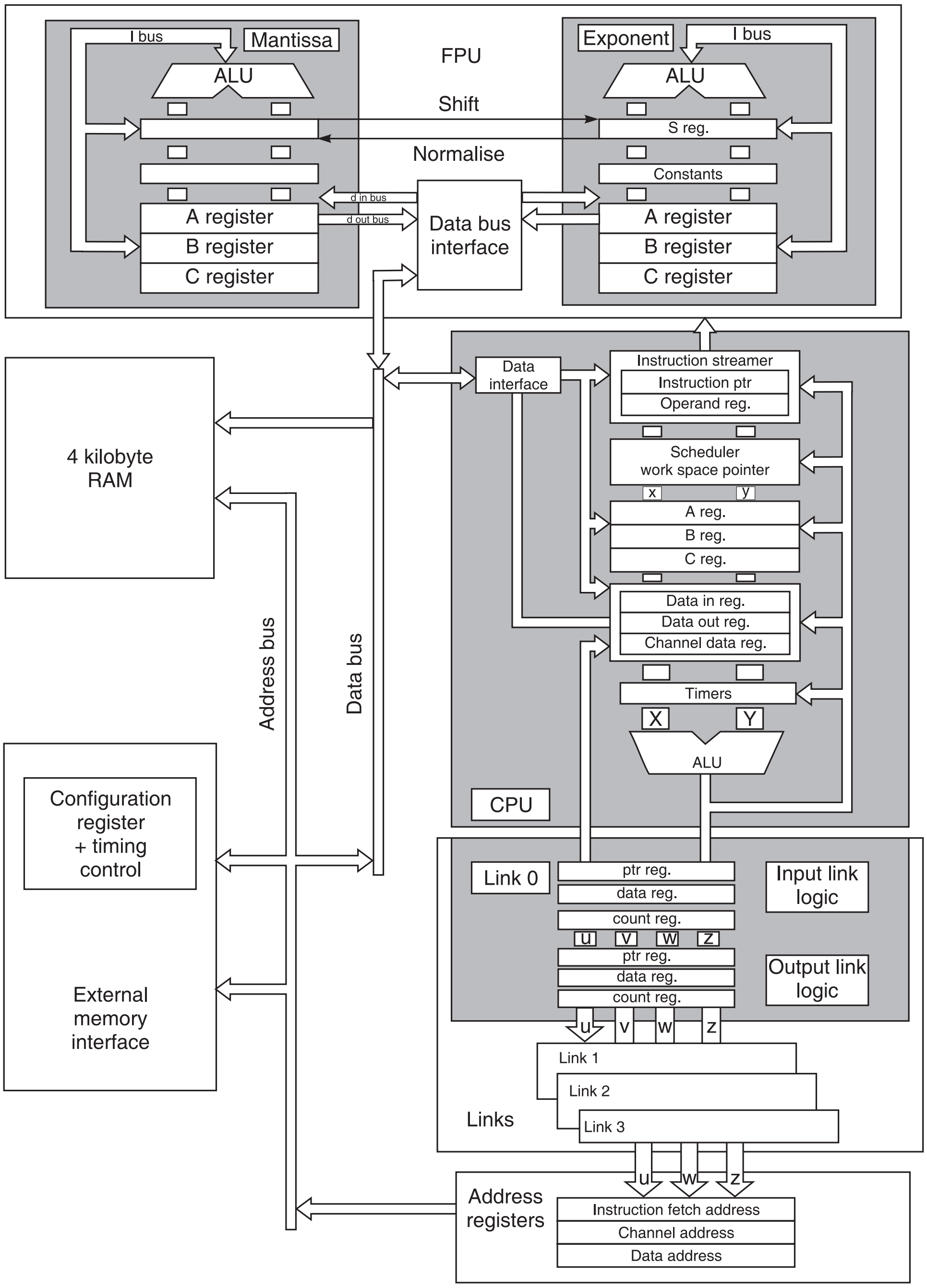


Figure 3.19 IMS T-800-30 transputer architecture.

(c) Full 32-bit transputer architecture, (d) 4 kilobytes on-chip RAM for 120 megabytes/sec. data rate, (e) 32-bit configurable memory interface, (f) External memory band width of

40 megabytes/sec., (g) High Performance Graphics Support, (h) Single 5 MHz clock input-DRAM refresh control, (i) Four 10/20 megabits/sec. INMOS serial links, (j) External event interrupt—Internal Timers, (k) Support for run-time error diagnostics, Boot from ROM or link etc.

3.2.6 Cell Microprocessor

The Cell Broadband Engine—or Cell as it is more commonly known—is a microprocessor designed to bridge the gap between conventional desktop processors (such as the Athlon 64 and the Core 2 family) and more specialized high-performance processors.

In terms of simple analysis, the cell processor can be split into four components: external input and output structures, the main processor called the Power Processing Element or PPE (a two-way simultaneous multithreaded Power ISA v.2.03 complaint core), eight fully-functional co-processors called the Synergistic Processing Elements or SPEs, and a specialized high-bandwidth circular data bus—connecting the PPE, the input/output elements and the SPEs—called the Element Interconnect Bus or EIB.

Both the PPE and SPE have RISC architectures with a fixed-width 32-bit instruction format. The PPE contains a 64-bit General Purpose Register set (GPR), a 64-bit Floating Point Register set (FPR), and a 128-bit Altivec register set. The SPE contains 128-bit registers only. These can be used for scalar data types ranging from 8 bits to 128 bits in size or for SIMD computations on a variety of integer and floating point formats. System memory addresses for both the PPE and SPE are expressed as 64-bit values for a theoretic address range of 2^{64} bytes (16,777,216 terabytes). In practice, not all of these bits are implemented in hardware. Local store addresses internal to the SPU processor are expressed as a 32-bit word.

Architecture

Cell chip can have a number of different configurations. The basic configuration (Fig. 3.20) is a multicore chip composed of one Power Processor Element (PPE), which is sometimes called

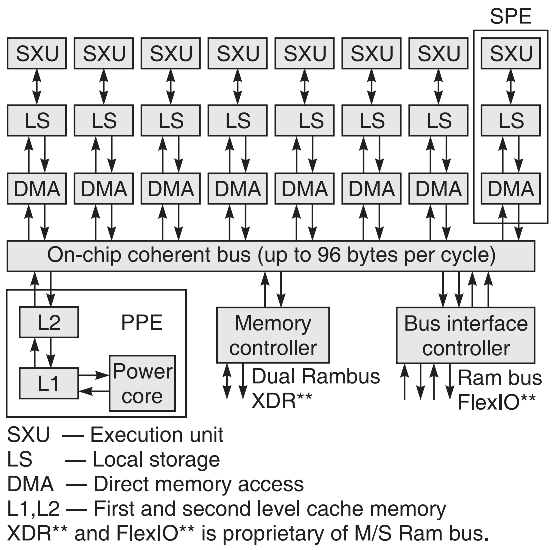


Figure 3.20 Architecture of Cell microprocessor.

the Processing Element or PE, and multiple Synergistic Processing Elements (SPE). The PPE and SPEs are linked together by an internal high-speed bus called the Element Interconnect Bus (EIB). Due to the nature of its applications, Cell is optimized towards single precision floating point computation. The SPEs are capable of performing double precision calculations, albeit with an order of magnitude performance penalty.

Power Processor Element (PPE)

The PPE is the Power Architecture based, two-way multithreaded core which acts as the controller for the eight SPEs which handle most of the computational workload. The PPE will work with conventional operating systems due to its similarity to other 64-bit Power PC processors.

Synergistic Processing Element (SPE)

SPEs are designed for vectorized floating point code execution. Each SPE is composed of Synergistic Processing Unit (SPU) and a Memory Flow Controller (MFC) having DMA, MU, and bus interface. An SPE is a RISC processor with 128-bit SIMD organization for single and double precision instructions. With the current generation of the Cell, each SPE contains a 256 Kilo byte embedded SRAM for instruction and data, called Local Storage, which is visible to the PPE and can be addressed directly by software. Each SPE can support up to 4 GB of local store memory.

Possible applications of the cell processor include video processing, server, video games, home cinema, supercomputing, cluster computing, distributed computing and mainframes.

IBM's latest supercomputer, IBM Roadrunner, is a hybrid of the General Purpose CISC Opteron and the Cell processors. This system assumed the Number one spot on the June 2008 Top 500 list as the first supercomputer to run at petaflops speeds, having gained a sustained 1.026 petaflops speed using the standard linpack benchmark. IBM Roadrunner uses the Power Xcell 8i version of the Cell processor, manufactured using 65 nm technology and enhanced SPUs that can handle double precision calculations in the 128-bit registers, reaching double precision 102 Gflops per chip.

An open source software-based strategy was adopted to accelerate the development of a Cell BE ecosystem and to provide an environment to develop Cell applications.

3.2.7 Configurable Processors

The most common embedded microprocessor architectures—such as the ARM, MIPS, and Power PC processors—were developed in the 1980s for stand-alone microprocessor chips. These general-purpose processor architectures are good at executing a wide range of algorithms, but designers often need more performance in critical portions of their designs than these microprocessor architectures deliver. The two approaches most often used to solve this performance gap are: (i) to find a processor that will run at a higher clock rate (thus extracting more performance from the same processor architecture) and (ii) to hand-design acceleration hardware. Even fast DSP (Digital Signal Processing Chip) architectures from the standard DSP vendors cannot match the speed of a custom-tailored hardware solution.

A configurable processor is a microprocessor that can be tailored to an application or a set of applications. There are three general ways to configure a processor:

- By selecting from standard configuration options, such as bus widths, interfaces, memories, and preconfigured execution units (floating-point units, DSPs, etc.);
- By adding new registers, register files, and custom task-specific instructions that support custom data types and operations such as 56-bit data and operations for security processing or 256-bit data types and operations for packet processing;
- By using programs that automatically analyze the C code and determine the best processor configuration and ISA (instruction-set architecture) extensions.

Configurable processors are delivered as synthesizable RTL (Register Transfer Language). In the electronics design field, it refers to the coding style in hardware description languages that ensures that the code model can be converted to real logic functions in a given hardware plateform like FPGA code, ready for placement into an FPGA (Field Programmable Gate Array) or SOC (System on Chip) design. The best configurable processors also come with automatically tailored software-development tools that reflect the task-specific ISA extensions.

A configurable processor can implement wide, parallel, and complex datapath operations that closely match those used in custom RTL hardware. The equivalent datapaths are implemented by augmenting the base processor's integer pipeline with additional execution units, registers, and other functions developed by the chip architect for a target application.

There is a recent trend to design a customized processor for an application. This is an extension of the bit slice processor approach, but at a lower level.

3.3 MULTIMICROPROCESSOR SYSTEMS

The architecture proposed by Von Neumann was Single Instruction Single Data (SISD) stream. A number of computers have been designed around this structure. The architecture of various microprocessors, as discussed in previous section is also based around Von Neumann architecture. The single instruction and single data stream computers are easy to conceptualise and design since the computer is executing only one instruction at a time. The data flow is from/ to only from one input/output unit at an instant. Multitasking concept was used for increasing the speed of program execution. This allows a number of programs resident in the computer's memory at one time. The computer switches from current task to other task as and when an I/O instruction is encountered. Since I/O units are comparatively slower than CPU, the computer on encountering the I/O instruction, initiates its execution and then starts executing another program. On completion of I/O, the computer gets signal to switch back to the original program. This optimises the CPU time. However, the branch and return addresses as well as the status of various programs are to be maintained by CPU. A number of new concepts have been introduced both in computers and microprocessors with the aim of increasing the speed, by incorporating parallelism in memory and processing. These concepts are:

1. Parallelism in memory
   — Interleaving

— Cache memory
— Multiple memory access

2. Parallelism in processing
   — Pipelining
   — Pipeline vector processing
   — Parallel processing

Most of the above concepts have found their way to microprocessors. The pipelining, cache memory, vector processing etc. are widely used in todays high-performance microprocessors.

It was however clear that parallelism is necessary for increased speed which is measured by Millions Instructions Per Seconds (MIPS) executed by the CPU. It was thought that instead of SISD architecture, data and instruction stream can be increased. The classification of computer architecture with respect to data stream and instruction stream is shown in Fig. 3.21. Other equally important reasons for introducing parallelism was reliability through redundancy in control systems and geographically or functionally distributed control systems.

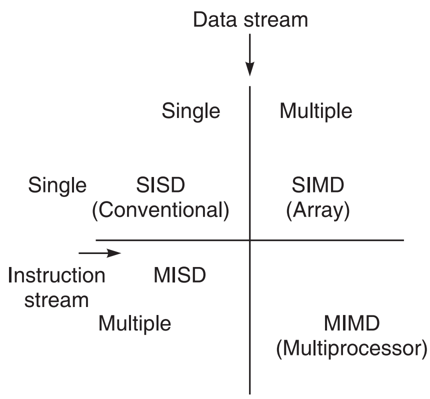


Figure 3.21 Classification of computer architecture.

The parallelism may be required in the systems due to various reasons. Some of the reasons are discussed below:

Performance

The performance of a system increases due to dedicated hardware for a particular task. In multimicroprocessor system, the independent parallel tasks for any application where a system is to be used, should be identified. These tasks should be allocated to individual processors, in order to increase the performance i.e. the speed of the systems manifold. The overall performance upgradation will be decided by a number of factors like execution time of individual tasks, total number of parallel tasks, and also the type of processors used etc.

Reliability

In many of the processes, it is imperative that the systems should function satisfactorily even in case of hardware or software failures. This results in duplication of the system elements and external resources. As an example, in the avionics, the expected mean frequency of failure per 10 hours flight is 10^{-9}. The conventional microprocessor frequency of failure is 10^{-2} to 10^{-5}. If a microprocessor based system is used for avionics application then it has to be suitably replicated to increase the reliability. In case of process industries also, similar but less stringent reliability criteria will exist and will amount to multiprocessor system for fail safe operations.

Distributed applications

There are number of applications in which the plant is distributed. For example, in case of offshore oil distribution, where the various distribution nodes are independent and are situated quite far from each other. In such applications, each node is managed by a separate microprocessor. These nodes work parallely and report to a single supervisory computer. Similar applications are found in number of other industries, like steel, refineries, etc.

General purpose computers

There is often a conflict regarding high MIP ratings and low cost among the designers of general purpose computers. The test is to provide cost effective general purpose computer with increased performance. Since the entire computer field is flooded with microprocessors, the general approach is to put more number of microprocessors together to achieve higher MIP ratings. However, it cannot be said that if the MIP ratings of single microprocessor was 1 MIP then by putting 4 of these together one would have the ratings of 4 MIP computer. This proportional relation does not exist due to overheads by operating systems and memory management.

Super computer design

There are certain applications which are beyond the scope of general purpose computers and are governed by very high MIP ratings. Some of these applications are weather forecasting, aerodynamic modelling, astrophysics etc. The purpose of using multimicroprocessor systems is same as that in case of general purpose computer, i.e. higher MIP ratings. However, the cost ceases to be a criteria in such systems.

3.3.1 Microprocessor Interconnections

There are a number of ways microprocessors may be interconnected to form a single multimicroprocessor system. These are: (a) Shared Bus; (b) Multiport memory; (c) Bus window; and (d) Cross bar switches.

All these techniques use common memory space between different microprocessor systems. The memory space also serves as interface between different microprocessors. The interconnecting bus is parallel. It may follow S-100, IEEE 488 or IEEE 796 bus standard. A number of systems have been designed using Intel Multibus also. The temporary interconnection follows Master-Slave configuration. The parallel bus has lines for address, data, control, interrupt and bus exchange. The bus exchange lines allow several masters to share the bus so that they can communicate with the slaves in an exclusive manner. Such a bus system can operate with a 10 MHz clock transmitting 16-bit addresses and 8/16-bit data in parallel.

Shared bus

The Shared Bus multiprocessor architecture is basically a Master-Slave Configuration in which Master Device requests for bus and communicates with slave device. In this configuration *Processor* is always designated as *Master* and *Memory* always as slave. This provides temporary link between two devices that need to communicate. At any given instant only one master-slave relationship can exist over a bus. The system may have many processors which will compete for becoming master by getting control on the bus. The concept is same as in Intel Multibus System. The concept of shared bus is shown in Fig. 3.22. The Shared Bus must resolve

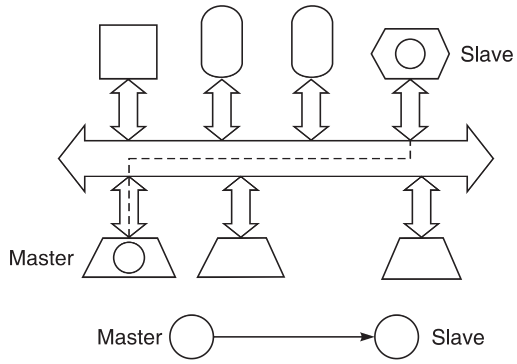


Figure 3.22 Shared bus.

the request received from many competing masters. Bus arbitration therefore, plays a very important role in the shared bus. Following three signals are involved in bus arbitration:

— Bus **Busy Line**
— Bus **Request**
— Bus **Acknowledge**

The Bus arbitration is essential to grant control of bus to one of the competing masters who have placed a request for the bus. The bus *acknowledgment* is issued to the *master* and the bus is shown *busy* for the other processors, till such time the present master releases the bus (Fig. 3.23). The bus arbitration is shown in Fig. 3.24. The arbitration scheme may be based on round robin arbitration, serial arbitration or parallel arbitration.

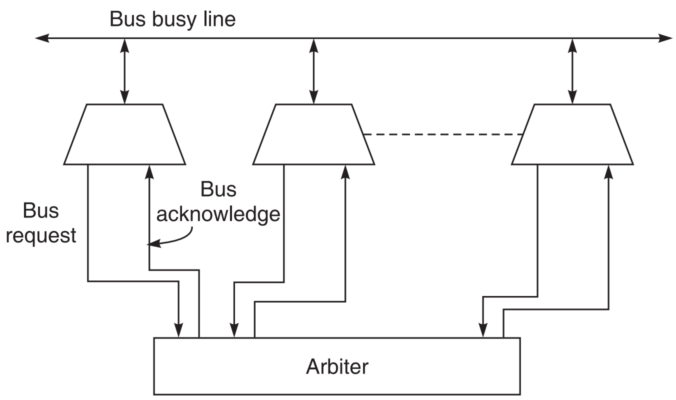


Figure 3.23 Bus arbitration.

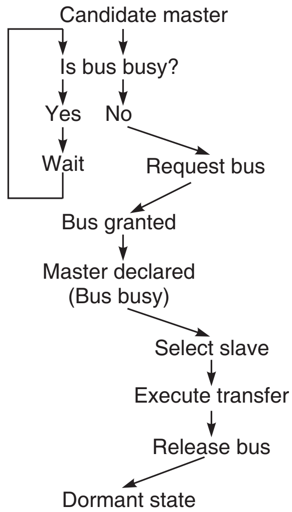


Figure 3.24 Bus allocation procedure.

A round robin arbiter (Fig. 3.25) polls each master in a fixed circular sequence. All microprocessors are assigned equal priority and are placed in a circular fashion. There is a pointer which is stepped up from one microprocessor to the next. Whenever there is a request for transfer pointer is moved. Out of all the requesting microprocessors, the one which is polled first by the pointer is granted the bus. When bus is released the pointer starts from that position on receipt of a transfer request.

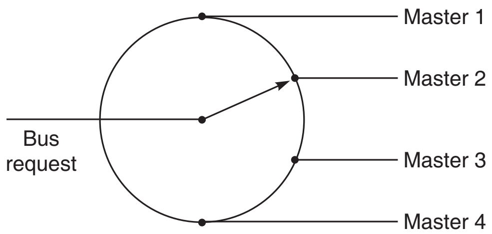


Figure 3.25 Round robin arbiter.

A serial arbiter (Fig. 3.26) allocates fixed priority to each microprocessor. These microprocessors are connected in the form of daisy chain. Each master has an input called priority input (PI) and an output called Priority Output (PO). The PI of left most master is connected to logic high (H). The left most master has the highest priority and the extreme right master is having the lowest priority. The priorities of intermediate masters depend on their position with respect to extreme left master in the chain. Normally, the left most high logic level H is propagated through each processor to the right. A master requesting the bus will force PO output to logic low state, which will be propagated (to indicate the request of higher priority master) to the extreme right master. When a higher priority master interrupts, the bus is released by lower priority master after preserving the pointers, etc. and entering the **wait** state. The bus

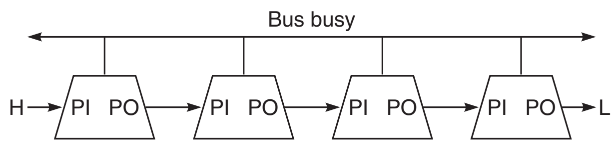


Figure 3.26 Serial arbiter.

busy line is used as acknowledgment of **Bus request**. The arbitration time depends on the length of chain which is a disadvantage.

The parallel arbitration scheme (Fig. 3.27) on other hand has the advantage of fast resolution of bus arbitration. It uses a parallel priority resolver circuit which continuously checks the bus requests. When a number of requests are received simultaneously, it awards the bus to the highest priority master. The speed of priority resolver does not depend on the number masters. If bus is captured by a master and then a request from higher priority master is received, then current master releases the bus after saving the context.

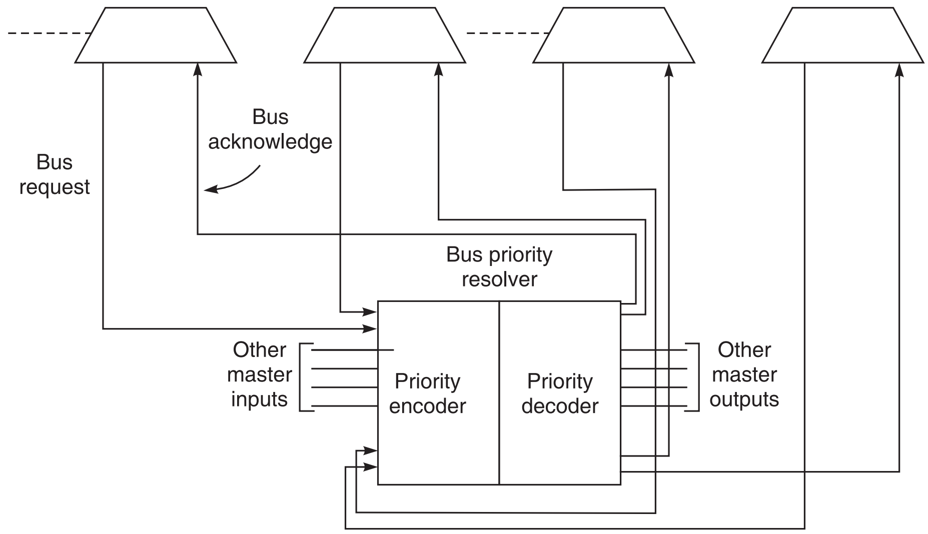


Figure 3.27 Parallel arbiter.

The shared bus system offers a very fast transfer medium, but has an inherent disadvantage. Because of the common memory, the bus is used as shared medium for even OP Code Fetch, Operand read/write etc. which is main cause for bottleneck for all microprocessors. This bottleneck may be removed by assigning local memory to each microprocessors for instruction and private data. By manipulating address lines, local memory and shared memory space can be separated. The Multibus structure based on same concept is shown in Fig. 3.28.

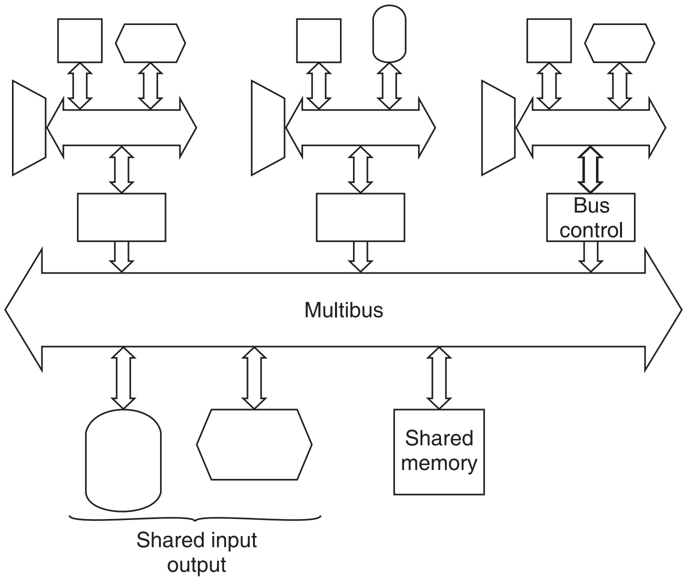


Figure 3.28 Intel multibus system.

Multiport memory

Multiport memory is basically memory unit which can be accessed by several processors at the same time. This allows instructions to be shared by number of processors thus avoiding duplication. It is also possible to transfer data prepared by one processor to the other processor. Though conceptually attractive, only dual port memories could be implemented that too for special applications (Fig. 3.29). These are not very much suited for multimicroprocessor applications.

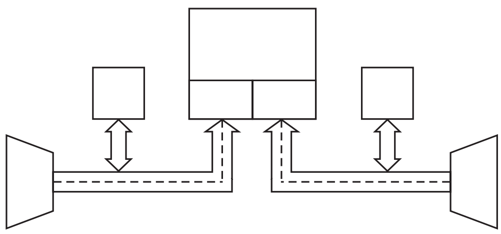


Figure 3.29 Multiport memory.

Bus window

A Bus Window connects two microprocessor buses through a common memory. In this case bus request of microprocessor A (Fig. 3.30) are taken as the request of microprocessor B for specified address zones. This implies that microprocessor B and its I/O devices can be accessed by microprocessor A as its own memory and I/O device. This technique provides very fast interconnection with very little programming overhead. The bus access follows Direct Memory Access protocol. However, it results in loss of effective memory space and also loss of overall speed due to DMA transfer. The CM* multiprocessor system has used bus window technique in which fourteen DEC LSI 11 processors can be connected with 4 K local memory each. There is a master unit called KMAP which decides access to address beyond 4 K. This unit decides a microprocessor in whose memory the address beyond 4 K should be mapped.

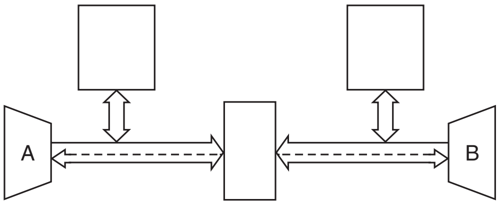


Figure 3.30 Bus window.

Cross bar switches

The Cross Bar Switch provides interconnection path between memory module and processor. The structure (Fig. 3.31) is similar to Post and Telegraph cross bar connection. The parallel data path is switched in this case. This allows a number of processors to have simultaneous access as long as there is no conflict. This interconnection scheme is very efficient but is very complex. A multiprocessor system, has been implemented using DEC PDP-11 processor in cross bar connections.

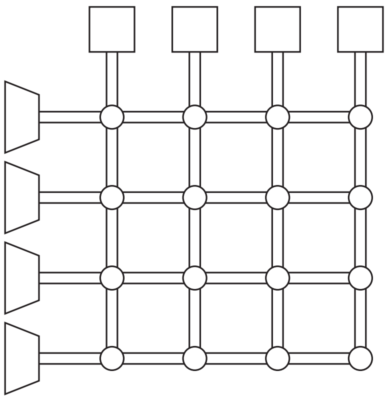


Figure 3.31 Cross bar switch.

3.4 LOCAL AREA NETWORKS

Local Area Networks are basically Loosely Coupled System having autonomous microprocessors with local memories interconnected via I/O circuits. The transfer of information requires

Input-Output operations (Fig. 3.32). Both serial or parallel interconnections are possible. Following are the techniques used.

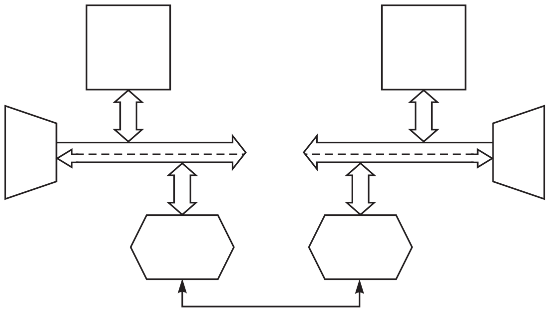


Figure 3.32 Local area network interconnection.

3.4.1 Contention Bus (Ethernet)

Contention Bus technique (Fig. 3.33) is very popular in data networks using computers. It was first used in Aloha network via radio communication. In this technique number of microprocessors can be connected to common system medium by means of controller. A number

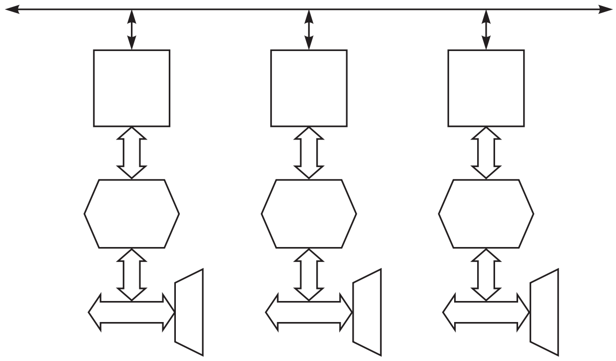


Figure 3.33 Ethernet structure.

of media access protocols are possible. The best known media access protocol is CSMA/CD with packet switching. This technique is called Carrier Sensing Multiple Access/Collision Detection. A brief description of this protocol is being given in the following:

1. Each microprocessor assembles the messages to be sent in packets which contain the source and destination address, message and error detection component.
2. When a microprocessor is ready to transmit, it senses transmission media to find out if it is busy. This is called ***carrier sensing.*** If the transmission media is not busy, then microprocessor sends the packet of information. This packet of information will be decoded by each microprocessor connected to the bus and microprocessor whose address is mentioned in the packet will receive the message.

3. It is very much possible that more than one microprocessor may sense the transmission media at the same time and start transmitting. In such cases the packets will collide resulting in no transmission. In case the collision occurs, a tone is generated. All the transmitting microprocessors on listening that tone stop transmitting further and will restart transmitting from the beginning after a delay of random time.

CSMA/CD is also known as nonpersistent media access protocol. The other protocols in this category are p persistent and predictive p persistent. With p persistent type, a channel is repeatedly checked. When it is found to be idle, the packet is transmitted with probability p (a fixed value between 0 and 1). This method improves throughput as two nodes are now less likely to transmit at the same time. The predictive p persistent type is like p persistent protocol except that p is dynamically adjusted to match the expected traffic on the network. For an idle or lightly loaded network p is large (0.8 to 1). With increasing network traffic p is reduced to compensate for the increased number of nodes that require access to the network. The method is also referred as CSMA/CA (collision avoidance). It is considered very useful for control applications using LAN.

In CSMA/CR (collision resolution) all nodes that need to transmit begin transmitting their unique collision resolution pattern which is assigned by system integrator at installation time. One bit at a time, the node accesses are revoked until the node with the highest priority is left to transmit its packet. This protocol sometimes may result into condition in which nodes may be locked out in presence of higher priority node activity.

3.4.2 Loop (Ring) System

The Loop or Ring System is deterministic in nature as against Ethernet system described above. In the Ethernet system, it is not possible to fix a time by which the message originated from source would have reached the destination. This makes Ethernet less suitable for real-time application which requires surety for message transfer. In the Ring System, communication may be serial or parallel. The processors are arranged in the form of ring. If data is to be passed from processor A to processor B, then all the intermediate processors collect the message and pass it to the next processor. Each processor decodes the message packet to see if destination matches with its address. If no, then the message packet is transferred to the next processor. It is easy to implement, since it relies on message circulation by serial or parallel data transfers. If serial lines are used then serial communication protocols that are supported by serial I/O circuits can be used to build the ring protocol. For parallel data transfer, a character by character (8-bit) or word by word (16-bit) transmission may be the basic unit of transmission and more complex protocol may be built.

The loop connection is shown in Fig. 3.34. This structure is vulnerable to single link or node failure which eventually results in breaking of loop. It is very necessary to provide a bypass circuit when processor is taken out from loop. This requires special circuits and methodologies to perform loop access. These methods are briefly described here.

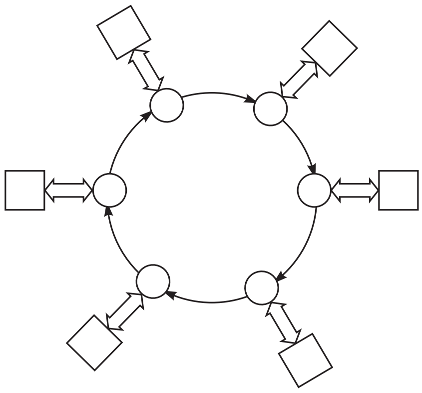


Figure 3.34 Loop (ring) structure.

Register insertion method

The register insertion loop (Fig. 3.35) requires a buffer register which is basically a shift register. Whenever a device has information to send, it stores in the buffer register. The register is connected into circuit in series whenever there is convenient gap between other packets travelling round. The register stays in series with ring and all packets of data are diverted through the register. When the packet originally transmitted by the processor at that location returns to it and is completely stored in the register, the register is switched out of the circuit. The operation in register insertion loop is explained in Fig. 3.36. Insertion of a new node into loop causes a delay of one packet transmission time.

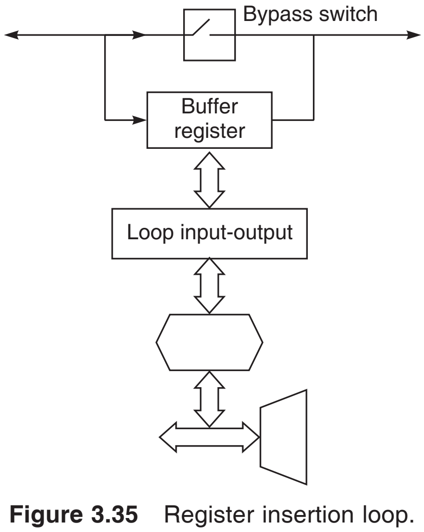


Figure 3.35 Register insertion loop.

Control token method

In this method, a unique character sequence called control token is passed around the ring. Any processor which intends to transmit the data, must wait for the control token to come to it. It

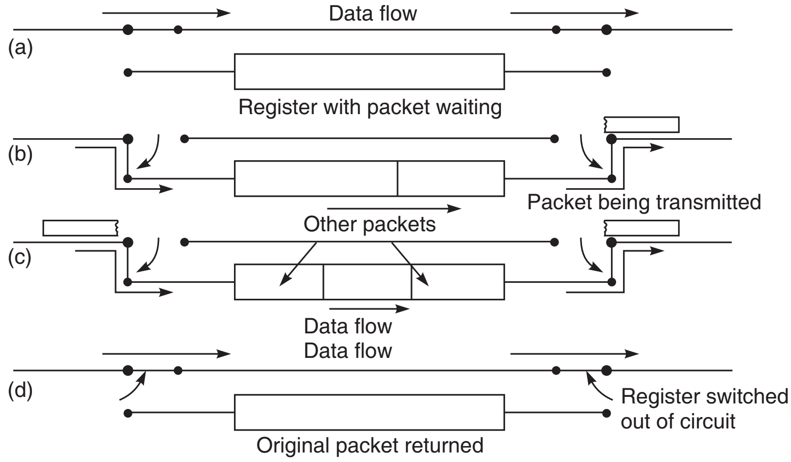


Figure 3.36 Register insertion operation.

removes the control token and places it behind the packet of data which is ready for transmission in shift register. The hardware circuit is same as that in Register Insertion method. The contents of shift register along with token at the end are sent into the ring. The register is then switched out of the circuit and the device waits for the return of its packets. Under the normal circumstances, the first incoming packet received by the device must be the one that it sent out. The packet is then removed and the register is switched out of the circuit. The operation is explained in Fig. 3.37.

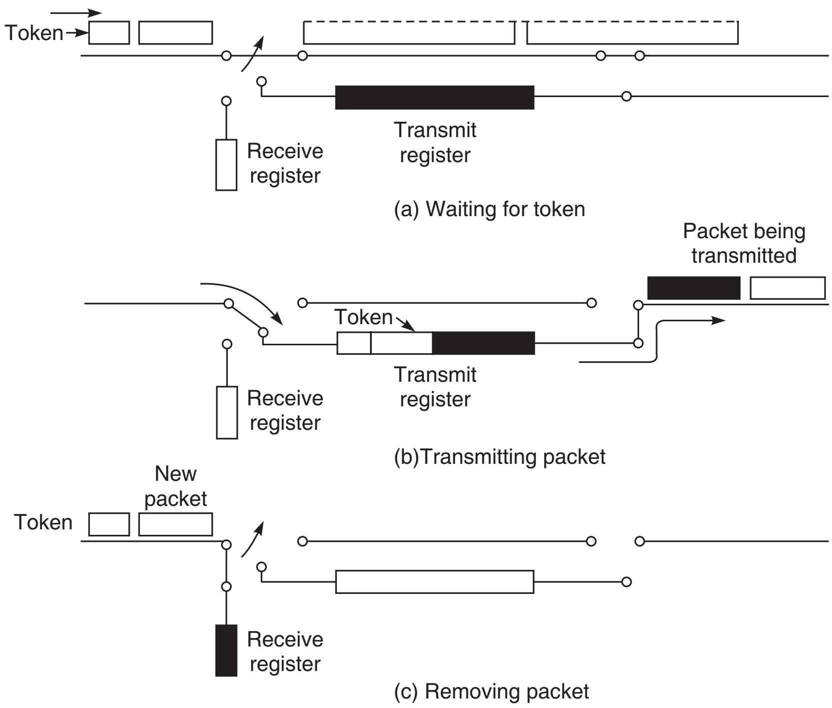


Figure 3.37 Token passing ring operation.

Empty slot method

The hardware circuit is shown in Fig. 3.38. The operation of empty slot technique is shown in Fig. 3.39. One or more skeleton packets or slots circulate continuously around the ring. The number of slots depends on the length of ring or loop. At ring start up, one processor generates a slot and sends it around the ring. If it returns to the sender then the ring must be complete and it can begin operation. When a processor has information to transmit, it waits for the empty slot to arrive. The control field in the header determines whether a slot is empty or full. The processor sets the empty/full flag to ‘full’, places the destination address in the header and shifts the data packets from its buffer to the data field of slot, as it is passed through the repeater. The slot continues around the ring until it reaches the destination. The destination processor reads and stores the information in buffer and sets a flag at the end of packet indicating that it has been received. The slot ultimately reaches the sender, passing through intermediate repeaters. The sending processor knows by counting the packets in the loop, that this packet was sent by it. The sender checks the acknowledgment of receipt field to find out if it was received at the destination. It then sets the empty/full flag to ‘empty’, thereby allowing other processor to use the slot. One of processors is used as loop controller to monitor the slots being full moving due to failure of destination. The packet format for a specific loop structure namely the Cambridge Ring is shown in Fig. 3.40.

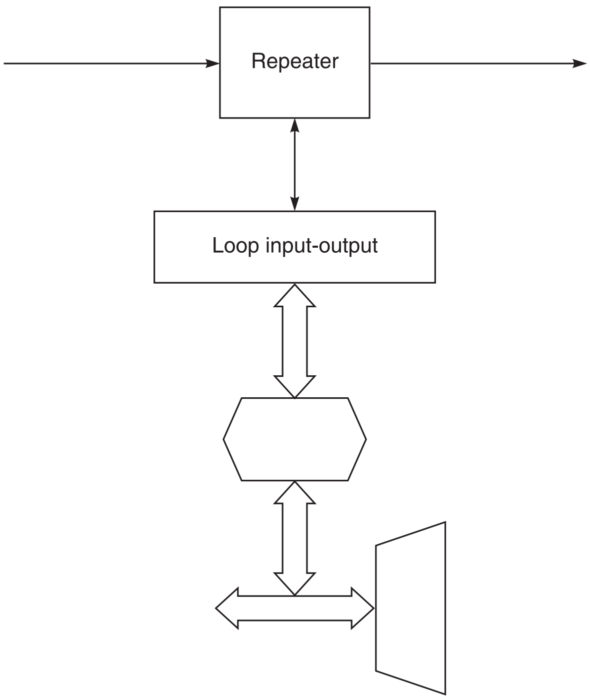


Figure 3.38 Empty slot loop.

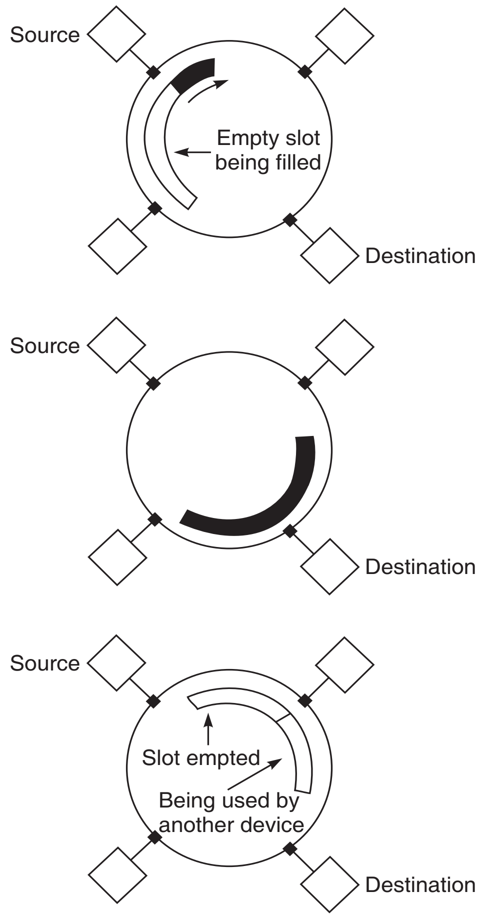


Figure 3.39 Empty slot operation.

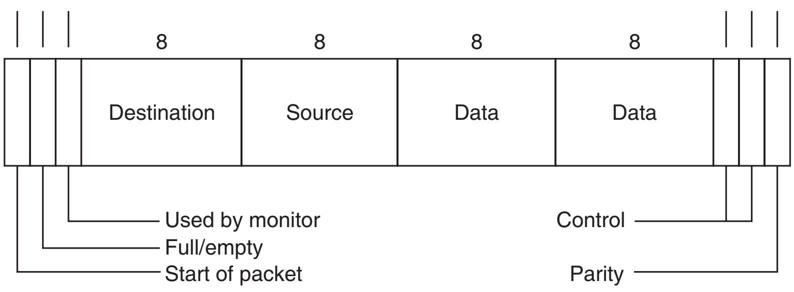


Figure 3.40 Cambridge ring packet format.

3.5 ANALOG AND DIGITAL I/O MODULES

After having discussed the computers and microprocessors, we shall be now dealing with the modules which connect the process to the data processing unit. Fig. 3.41 show the basic signals in real-time systems.

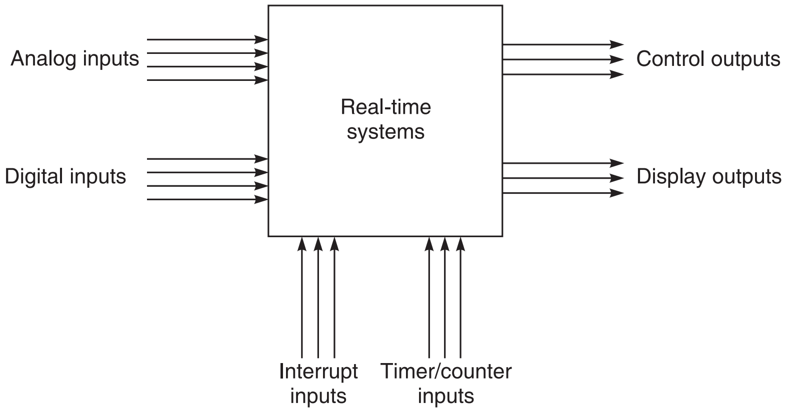


Figure 3.41 Basic signals in real-time systems.

Analog input signals are received from sensors and signal conditioners and represent the value of measurand like flow, position, displacement, temperature, etc. The role of a sensor is to measure the parameter for which it is constructed and present an equivalent electrical signal as output. The signal conditioner takes as input the output of sensor and suitably conditions it to be acceptable to real-time systems. The signal may be amplified, filtered or/and isolated in signal conditioner depending on the sensor type and its electrical characteristics.

Digital input signals refer to the ON-OFF states of various valves, limit switches, etc. One digital input signal represents status of the limit switch or valve and is represented by one bit of information for real-time systems. Normally digital input signals are compatible to real-time systems and can be inputted directly. In case some signal amplification/reduction is required, signal conditioning unit is added, before real-time system.

Interrupt input signals draw the attention of real-time system towards certain abnormal situations in the environment or the process monitored/controlled by real-time system. The real-time system on receipt of interrupt signal attends to the abnormality pointed out and resumes its normal work from the point where it was suspended. The abnormalities may be in terms of (a) exceeding of certain limits of some parameters like temperature, flow, etc. which may require to be controlled immediately, (b) power failure in which case all the parameters must he stored and work may be suspended, or (c) some process faults which must he notified immediately. One interrupt signal will correspond to only one particular abnormality which needs to be attended. Real-time systems attend to abnormalities by executing special programs called *Interrupt Servicing Routines.* Thus there is one to one correspondence between an abnormality—interrupt signal and interrupt servicing routine.

In order to enable the real-time system to suspend its current programme, execution of the interrupt servicing routine and restarting of the suspended program is needed. The facility to store the status of programme and fetch the same afterwards is essential.

Timer/counter input signals are important pan of any real-time system. Through these signals the concept and measure of real-time is derived. These signals are used as clock input to timer/counter in real-time system or gate input to enable/disable different timers/counters. The timer circuits may be used to initiate events at defined intervals. The counter circuits on the other hand, may be used to count the occurrences of any defined event. The output of timer/counter may be used as interrupt signal to real-time system.

Display output signals are used to drive the display devices like, LED, LCD, VDU, Audio Alarms etc. The display of status of process, various control valves etc. is very important to the operator. Apart from this, the limits set for various parameters at different places in the process are also displayed for the benefit of operators. The display output signals carry the information which is displayed on one or more display devices. Some of the real-time systems do not control the process but display the various parameter values, their variations, limits set etc. for the benefit of operators, who eventually control the process by manually operating various control valves. Such real-time systems are called *Data Acquisition Systems.*

Control output signals are required to drive the control valves, motors etc. to perform the control action decided by the real-time systems. The control action desired may be simple ON-OFF control of valves/motor or fine control of motor speed, position, displacement, flow and level through control valves. The control output signals are analog signals which can drive various actuators. However with the emergence of digital actuators, these analog signals will soon be replaced by digital signals.

The major hardware sub-systems of real-time control systems are shown in Fig. 3.42. We shall now describe the function of each module.

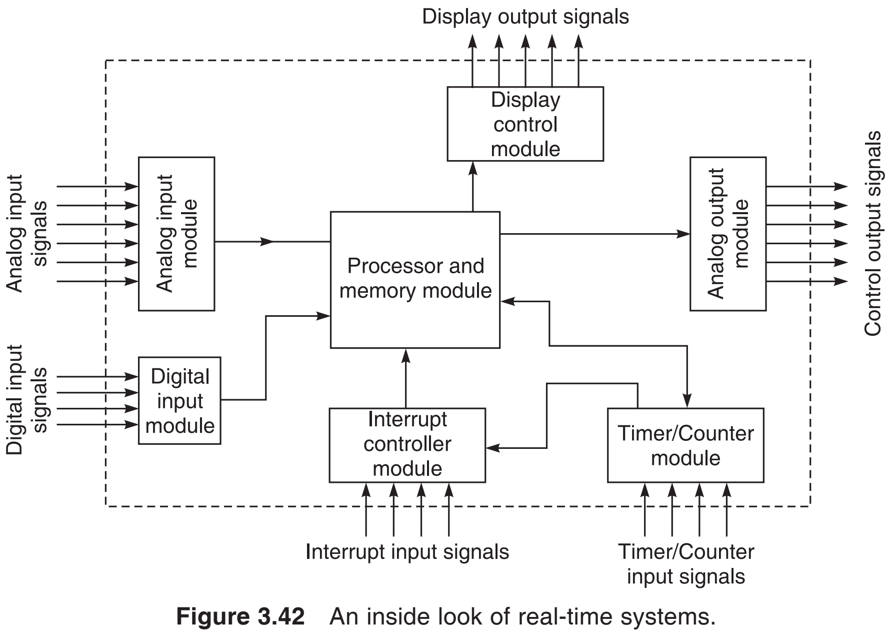


Figure 3.42 An inside look of real-time systems.

3.5.1 Analog Input Module

The module continuously scans the analog input signals in the pre-defined order and frequency, converts them in to the digital and then sends these values to processor and memory module for processing. The order of scanning of analog input signals is defined and set in analog input module. This may either he done by software programming and fusing the programme on EPROM of analog input module, or setting the order of scanning through thumb wheel switches which may be provided to the user. The analog input module may also provide signal conditioning for some standard transducers like thermocouples, LVDTs, strain gauges, etc. In such cases, analog input modules may be different for different types of transducers. Since this may prove to be an expensive solution for the user, normally analog input modules do not contain signal conditioning.

Depending on the application and sophistications, the analog input module may contain local intelligence or may be commanded by the master processor. The internal block diagram of an analog input module without local intelligence is shown in Fig. 3.43. The analog input module of Fig. 3.43 operates under the command from processor in the following manner:

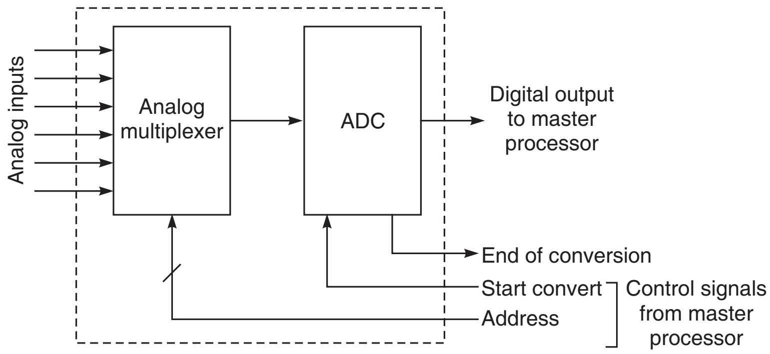


Figure 3.43 Analog input module with no local intelligence.

1. The processor initiates multiplexer by sending the address of input channel.
2. The multiplexer connects the particular channel to the ADC.
3. The processor sends the Start Convert signal to ADC. The ADC converts the analog signal to digital, puts it at the output and issues End of Conversion signal.
4. The processor on receipt of the End of Conversion signal, reads the ADC output and stores in memory.
5. The operation is repeated by processor by sending the address of next channel to multiplexer.

This structure of analog input module is suitable for lower end of applications with fewer analog points and lower speed. This is an appropriate and low cost solution for such applications.

The block diagram of analog input module with local intelligence is shown in Fig. 3.44. The intelligence is provided by the processor and memory. The processor does the scanning of various channels and stores the digital values in local memory. The interaction with main processor is achieved through Direct Memory Access or Interrupts, initiated by main processor.

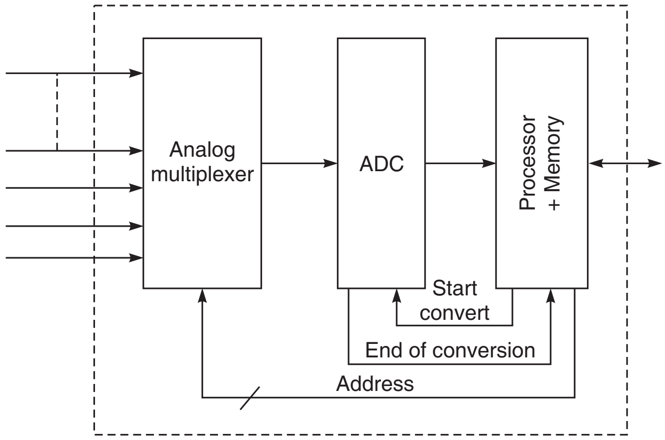


Figure 3.44 Analog input module with local intelligence.

3.5.2 Digital Input Module

The digital inputs cm be accepted directly by the processor. Thus no analog to digital converter is required in digital input module. The block diagrams of digital input module without and with local intelligence are shown in Figs. 3.45 and 3.46 respectively. Each digital input channel consists of *n* bits which are transferred in parallel The explanation of the two block diagrams are similar to the respective analog input modules.

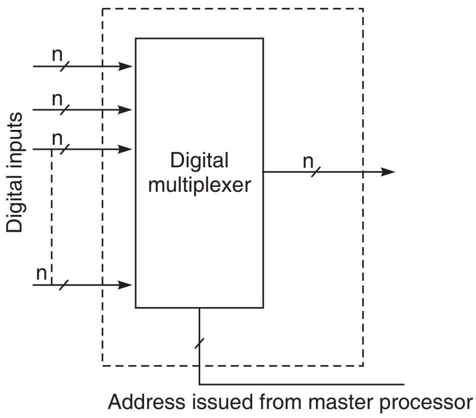


Figure 3.45 Digital input module without local intelligence.

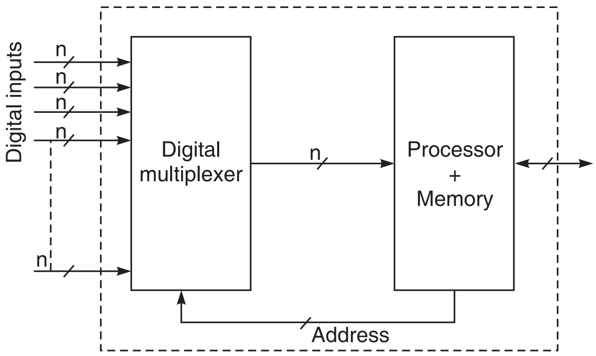


Figure 3.46 Digital input module with local intelligence.

3.5.3 Analog Output Module

The objective of analog output module is to provide appropriate control signals to different control valves. Figure 3.47 shows the structure of analog output module which is derived by

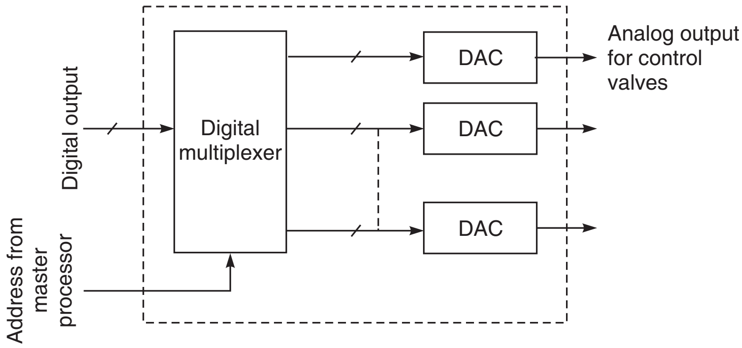


Figure 3.47 Analog output module—Structure I.

reversing the analog input module shown in Fig. 3.43. No local intelligence is necessary in analog output module. The demultiplexer switches the digital output received from the master processor to the output channel whose address is specified. The digital to analog converter of particular channel will convert the input digital value to equivalent analog signal which is connected to control valve, motor etc.

Where the structure shown in Fig. 3.47 is simple to understand and design, it is not modular. That is, if number of control valves increase then addition of similar circuit will he tedious. Another major disadvantage is that the structure is not compatible to output commands in assembly language of master processor. Let us explain this point in detail.

In the master processor the various I/O signals are accessed via ports. In Fig. 3.47, the address bits of demultiplexer are coming from an output port in master processor. When master processor is setting address bits it is actually writing a byte/word to output port.

Similar is the case for digital output which is input to digital demultiplexer. Thus master processor will be executing following two I/O instructions to generate analog output for control valves:

— Write data byte/word in digital output port.

— Write address byte/word in address port.

According to this structure, for a programmer the knowledge of lower circuit details is necessary which is however not desirable. What will a programmer desire then? The programmer would like to address each control valve through a port number and address this port number in assembly or high level language. Typically, if port 'n' refers to control valve number 'n' then the I/O instruction should be—Write data byte/word in port 'n'.

The structure of analog output module, shown in Fig. 3.48 fulfils this requirement and is universally used in almost all control systems. If number of control valves increase then a similar circuit can be put in parallel for different port numbers.

3.5.4 Interrupt Control Module

Interrupt control module, basically provides interface between a number of interrupting devices and master processor. The conflict between different devices needing the immediate attention of master processor at the same time is resolved by allocation of priorities to devices. The priorities

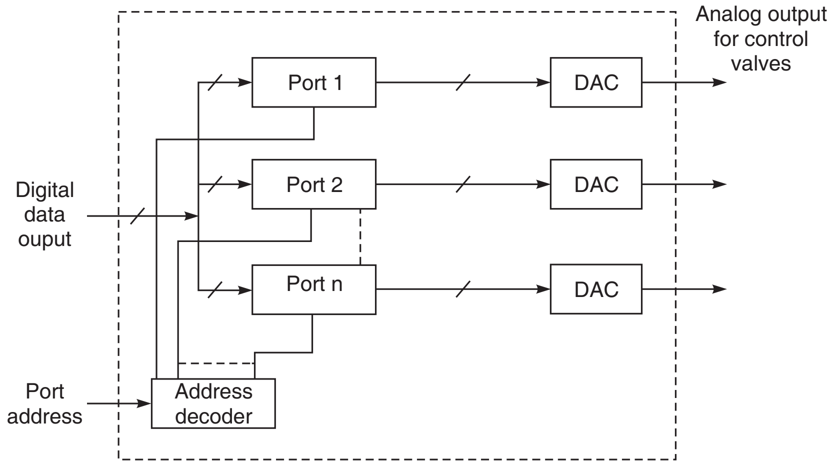


Figure 3.48 Analog output module—Structure II.

may be static, i.e. fixed or dynamic (priorities may change depending on the situation). The interrupt controller finds out the highest priority device request to interrupt master processor, then finds out the processor status. It is possible that processor is executing a higher priority interrupt or interrupt bas been disabled in which case the interrupt request is kept pending. Only in case the processor is executing lower priority interrupt and interrupt is not disabled then interrupt signal is sent to processor. The module also keeps the status of interrupt requests received. The block diagram of 8259A interrupt controller is shown in Fig. 3.49.

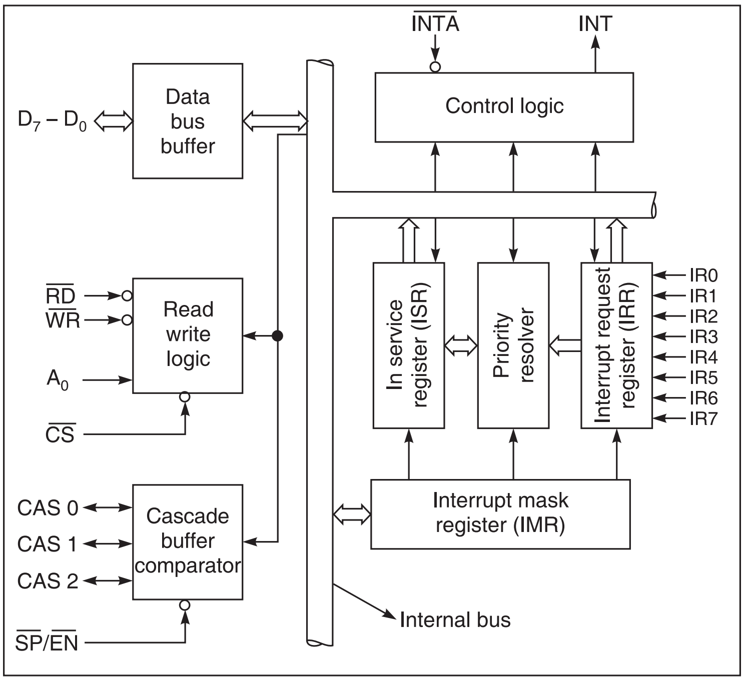


Figure 3.49 Blcok diagram of an 8259A interrupt controller.

3.5.5 Timer/Counter Module

Timer/counter module basically consists of a number of times/counters which may be cascaded or used independently. Each timer/counter may be programmed in different modes in which case the timer/counter output will be different. The modes may be interrupt on zero count, rate generator, monoshot etc. We shall not discuss the modes of timer/counter here. The structure of a timer/counter is shown in Fig. 3.50.

The processor loads the count in the form of data byte/words. The clock may be derived from processor clock or may be provided externally. The gate signal is used to enable/disable the counter operation. The processor may read the current counter value at any instant by stopping the counter using gate signal or read it on the Fly, i.e., without stopping the counter. The output may be used to interrupt the processor or in any other way as programmed.

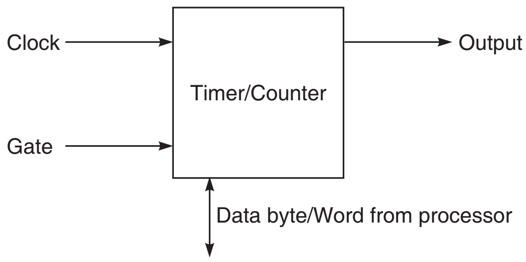


Figure 3.50 Timer/Counter module.

3.5.6 Display Control Module

Display control module consists of following independent sub-modules.

— Manual entry sub-module
— CRT controller sub-module
— LED/LCD control sub-module
— Alarm annunciator sub-module
— Printer controller sub-module

The manual entry to the system may be via thumb wheel switches, various ON-OFF command switches and/or keyboard. The keyboard may be a full ASCII keyboard or a specialized numeric and task-oriented keyboard. There are both advantages and disadvantages in using any manual entry submodules and we will not discuss these here. Basically, manual entry sub-modules have built in buffer to store current setting of thumb wheel switches and last command/data entered through keyboard. The processor may read these values on its own or may be interrupted whenever a new data/command is entered.

CRT controller sub-module interfaces main processor to Visual Display Unit, which is used to show the status of process by displaying transducers values, present set points entered through manual entry sub-module, historical trend of various parameters, mimic diagram of process, alarm status etc. All the above display tasks are provided in the main processor through software.

LED/LCD control sub-module interfaces array of LED/LCD to main processor. This sub-module accepts data bytes/word from main processor and displays it on LED/LCD. Alarm annuciator controller sub-module generates ON-OFF signal for each type of alarm. The processor may send an alarm byte/word to the alarm annunciator controller sub-module which decodes byte/word and sends ON-OFF signals for various types of alarms. These alarm modules are separate and require only digital signal to light the incandescent bulbs and/or audio alarms.

Printer controller sub-module is printer interface to main processor. Generally it has local intelligence for printer control, a data buffer to store the data for printing etc. It accepts the data bytes/words from main processor and prints it for the benefit of operator. The interaction with main processor may be through periodic direct memory access or under the direct control of main processor (programmed I/O transfer).

3.6 SUPERVISORY CONTROL AND DATA ACQUISITION SYSTEMS

After having dealt with the basic hardware modules of a real-time system, let us first concentrate on Supervisory Control and Data Acquisition (SCADA) system, since it is the first step towards automation. The basic functions carried out by an SCADA system are:

— Channel scanning
— Conversion into engineering units
— Data processing.

Figures 3.51 shows the block schematic of SCADA. Before considering other features of SCADA, let us discuss the basic functions.

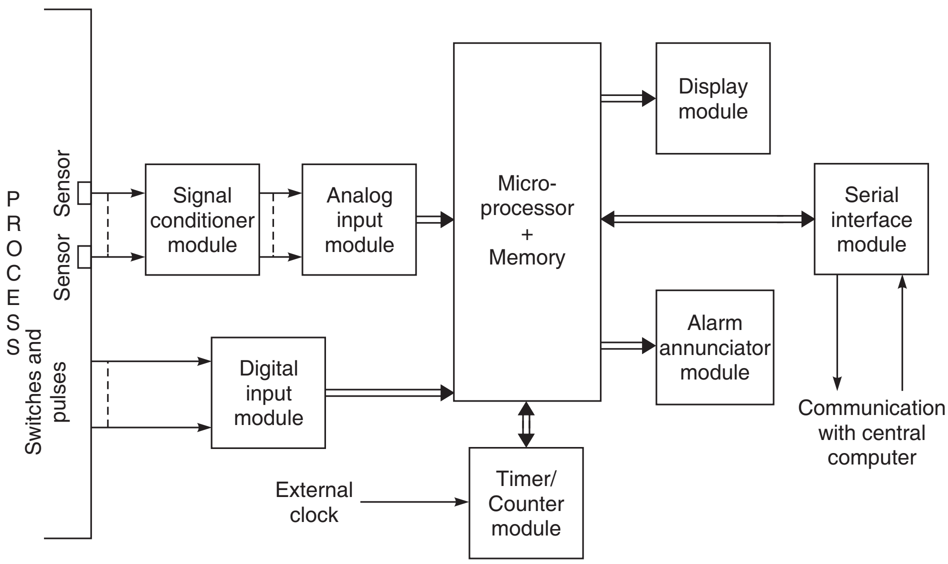


Figure 3.51 Supervisory control and data acquisition system.

3.6.1 Channel Scanning

There are many ways in which microprocessor can address the various channels and read the data.

Polling

The microprocessor scans the channels to read the data, and this process is called *polling***.** In polling, the action of selecting a channel and addressing it, is the responsibility of processor. The channel selection may be sequential or in any particular order decided by the designer. It is also possible to assign priority to some channels over others, i.e. some channels can be scanned more frequently than others. It is also possible to offer this facility of selecting the order of channel addressing and channel priorities to the operator level, i.e. make these facilities as dynamic.

The channel scanning and reading of data requires, the following actions to be taken:

— Sending channel address to multiplexer

— Sending start convert pulse to ADC

— Reading the digital data.

For reading the digital data at ADC output, the end of conversion signal of ADC chip can be read by processor and when it is 'ON', the digital data can be read. Alternatively, the microprocessor can execute a group of instructions (which do not require this data) for the time which is equal to or greater than conversion time of ADC and then read ADC output. Another modification of this approach involves connecting the end of conversion line to one of the interrupt request pins of the processor. In this case the interrupt service routine reads the ADC output and stores at predefined memory location.

The channels can be polled sequentially, in which case the channel address in first step above increases by one every time or they may be scanned in some other order. In the later case, a channel Scan Array can he maintained in memory (Fig. 3.52). The Scan Array contains the addresses of the channels in the order in which they should he addressed. The ASCN array in Fig. 3.52, has 9, 10, 1, 2 ... as entries in sequence. Thus, the first channel to he scanned will be channel 9, followed by 10, 1, 2 .... As the pointer reaches the last entry in the array, the first entry is again taken up (i.e. channel 9 is scanned).

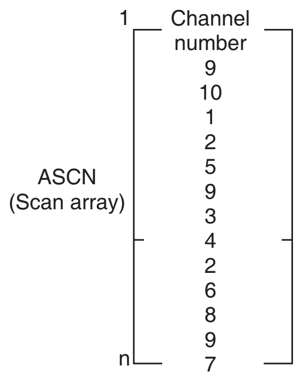


Figure 3.52 Channel scan array.

If a channel number is repeated in the array, then that particular channel will be scanned repeatedly. Thus it is possible to scan some channels more frequently than others. This gives them higher priority over others. In Fig. 3.52 the channel 9 is scanned 3 times, channel 2 is scanned 2 times while other channels are scanned once during a cycle. Figure 3.53 shows the flowchart for one scan cycle.

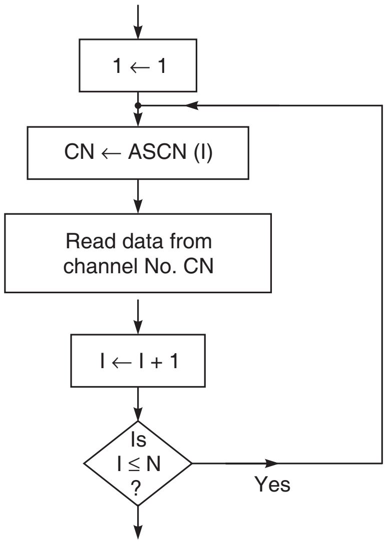


Figure 3.53 Scan cycle flow chart.

The processor may scan the channels continuously in the particular order as illustrated by the flowchart or the channels may be scanned after every fixed time period. The second approach requires a timer/counter circuit whose output is connected to interrupt request input. The scan routine for one channel is incorporated in 'Interrupt Service Routine'. It is also possible to make at time gap between two channels as variable. This would require a $n \times 2$ dimension scan array as shown in Fig. 3.54. The Interrupt Service Routine fetches the time gap value for next channel, loads the timer/counter with the value and initiates the timer/counter before returning to main program.

The scan array may he decided at the design stage of SCADA and fused permanently in (ROM) Read Only Memory. Thus the channels are always scanned in that particular order. However, it may be desirable to offer the facility of changing the sequence at the operator level. The operator may like to take this action depending on the condition of the plant being monitored. As an example, if at any instant operator finds out that the transducer connected to channel 9 has generated some fault, then he might take decision to bypass the channel 9, as otherwise its data will be taken into analysis to produce incorrect result. In another situation, it may become necessary to scan some channel more frequently for some time to observe its response to some modifications incorporated in the system. The operator should thus be able to insert new scan array at any time. This facility may be provided through a key switch which may be connected to interrupt request input of processor. The Interrupt Servicing Routine will accept new scan array and store in place of the old one.

	Channel number	Time gap
1	9	FF (H)
	10	0F (H)
	1	F0 (H)
	2	
	5	
	9	
	3	
	4	
	2	
	6	
	8	
	9	
n	7	

Figure 3.54 Scan array with time.

Interrupt scanning

Another way of scanning the channels may be to provide some primitive facility after transducer to check for violation of limits. It sends interrupt request signal to processor when the analog signal from transducer is not within High and Low limits boundary set by Analog High and Analog Low signals. This is also called *Scanning* by *Exception.* When any parameter exceeds the limits then the limit checking circuit would send interrupt request to microprocessor which inturn would monitor all parameters till the parameter values come back within pre-specified limits. This allows a detailed analysis of the system and the problems by the SCADA system

The limit checking circuit for one channel is shown in Fig. 3.55. Two analog comparators check whether the input signal is within *high* and *low* limits. The output is ORed and the final output is used as interrupt request to microprocessor. This limit checking should not be construed as alarm condition, but the condition for the start of any abnormality which may generate alarm condition or may be controlled by the system before any alarm condition is reached. Therefore, the system should be watched closely during this time.

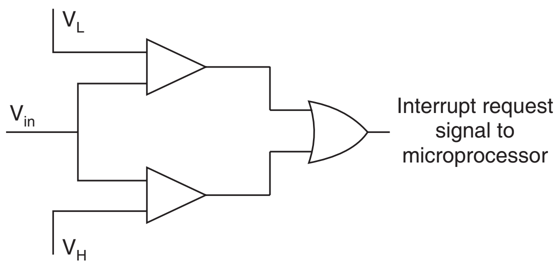


Figure 3.55 Interrupt request generation on limit violation.

3.6.2 Conversion to Engineering Units

The data read from the output of ADC should be converted to the equivalent engineering units before any analysis is done or the data is sent for display or printing. For an 8-bit ADC working in unipolar mode the output ranges between 0 and 255. An ADC output value will correspond to a particular engineering value based on the following parameters.

1. Calibration of transmitters
2. ADC mode and digital output lines.

The transmitter output should be in the range of 0-5 V or 4-20 mA range. Depending on the input range of measurand value for transmitter, a calibration factor is determined. If a transmitter is capable of measuring parameter within the input range X_1 and X_2 and provides 0–5 V signal at output then calibration factor is

$$1\ \text{Volt} = \frac{X_2 - X_1}{5}\ \text{units}$$

If we are converting this signal to digital through an 8-bit ADC (Input range 0–5 V) in unipolar mode then

$$5\ \text{V} = 255 \text{ and } 0\ \text{V} = 0$$

i.e.

$$1\ \text{Volt} = \frac{255}{5}$$

Thus the conversion factor is

$$\text{ADC output } \frac{255}{5} = \frac{X_2 - X_1}{5} \text{ engineering units}$$

$$\text{ADC output } 1 = \frac{X_2 - X_1}{255} \text{ engineering units}$$

If the ADC output is Y then the corresponding value in engineering units will be

$$\frac{Y(X_2 - X_1)}{255}$$

$$\text{Conversion factor is therefore } \frac{(X_2 - X_1)}{255}$$

The conversion of ADC output to engineering units, therefore, involves multiplication by conversion factor. The conversion factor is based on the ADC type, mode and the transmitter range. This multiplication can he achieved by shift and add method in case of 8-bit microprocessor. For 16-bit microprocessor, a single multiplication instruction will do the job.

3.6.3 Data Processing

The data read from the ADC output for various channels is processed by the microprocessor to carry out limit checking and performance analysis. For limit checking the *Highest* and *Lowest* Limits for each channel are stored in an array (Fig. 3.56). When any of the two limits is violated for any channel, appropriate action like alarm generation, printing, etc. is initiated. The limit array shown in Fig. 3.55, simplifies the limit checking routine. Through this, the facility to dynamically change the limits for any channel may also be provided, on the lines similar to scan array described in Section 3.6.1.

$$\begin{array}{cc} \text{Higher limit} & \text{Lower limit} \\ \multicolumn{2}{c}{\begin{bmatrix} 10 & 02 \\ 20 & 09 \\ \vdots & \vdots \end{bmatrix}} \end{array}$$

Figure 3.56 Limit array.

In addition to limit checking, the system performance may also be analysed and report could be generated for the manager level. This report will enable the managers to visualise the problems in the system and to take decisions regarding system modification or alternate operational strategy to increase the system performance. The analysis may include histogram generation, standard deviation calculation, plotting one parameter with respect to another and so on. The software can be written depending on the type of analysis required.

The facilities like scanning, limit checking, etc. could be incorporated in a simple way using arrays in software. Let us first examine channel scanning, using scan array in memory which contains the channel numbers in one particular order of scan. For two applications using the same hardware, only the scan array will have to be changed for the scanning. The same argument is valid for limit checking. The conversion to engineering units can also be achieved by storing the conversion factors for different channels in any array. The conversion factors stored in the array will change with the transducers i.e. it will be different for different applications. The simple print programme which takes the data in particular format from memory and sends for printing may also be same for different applications. However, the designer will have to write specialized routine for arranging the data in memory in any particular format required. The analysis and report generation programs will be application dependent and will have to he written separately for different applications. Here also the routines for histogram etc. may be used by different applications if they are written in generalised manner.

3.6.4 Distributed SCADA System

In any application, if the number of channels are quite large then in order to interface these to processor, one has to use multiplexers at different levels. Figure 3.57 shows the interfacing of

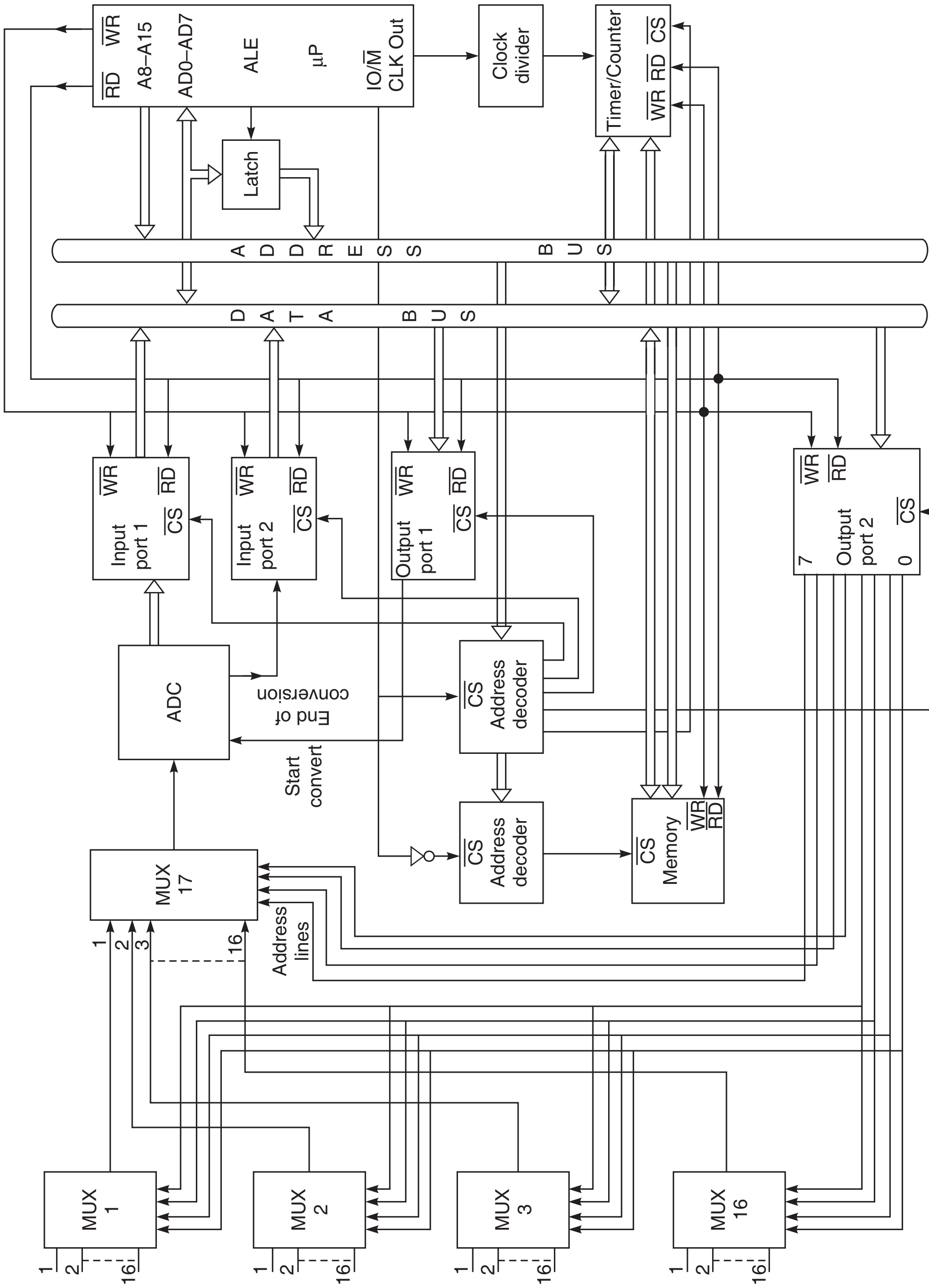


Figure 3.57 256 channel SCADA with single microprocessor.

256 channels, using 17 multiplexer of 16 channels each. The 8 address lines are used to address 256 channels. Out of the 8-address lines, upper four are used to select a particular multiplexer and lower four lines are used to select a particular channel in the multiplexer.

The 8-bit channel address thus directly maps into channel number and can be manipulated in any way. The other parts are same as described earlier. This approach will be suitable for the processes which are basically slow. Even if a channel is scanned only once in every scan, it is only after 255 channels have been scanned, limit checking and analysis have been performed, a particular channel will be addressed again. This is not acceptable in many processes.

For the process plants where the structure of Fig. 3.57 does not suit, the only alternative is to use more than one SCADA system and distribute the channels among them. But, for performance analysis on the process plant, it is mandatory that the data from various channels should reach **a** central location where it can be consolidated and analysed to generate the reports on plant performance. Figures 3.58(a) and (b) show the interfacing of number of SCADA systems with central computer in *star* configuration and *Daisy chain* configuration respectively.

The SCADA system directly connected to transducers are called nodes and are the same as the systems described earlier. They scan the channels using one of the techniques discussed, earlier; convert the data into engineering units, perform the limit checking, generate alarm, if data item crosses the limit and generate print out.

In addition to these functions, the data regarding the channels in the node are transferred to central computer which analyses the system performance and generates print out. The print outs are generated by exception, i.e. unnecessary data is not printed at any point. At the node level, the print out is required for the operators to run the system. Depending on the node performance, operator may decide to monitor any channel more frequently, change the limits etc. The print out at the central node is required for the managers to take long term decision to optimise the performance. The details on the channel performance, limit violation are not required at this level. On the other hand, histogram on the input and output material flow and the fuel consumption etc. may he more helpful.

The concept of local area networks or microprocessor interconnections can be used in case of distributed SCADA system very effectively. Thus we conclude that the distributed SCADA is the ultimate solution for complex process plant monitoring.

3.7 REMOTE TERMINAL UNIT

The Remote Terminal Units (RTUs) are basically distributed SCADA based systems used in remote locations in applications like oil pipelining, irrigation canals, oil drilling platforms etc. They are rugged and should be able to work unattended for a long duration. There are two modes in which Remote Terminal Units work.

1. Under command from central computer
2. Stand alone mode

Since these RTU's have to operate for a long duration unattended, the basic requirements would be that they consume minimum power and have considerable self-diagnostic facility. Following are the main parts of remote terminal units.

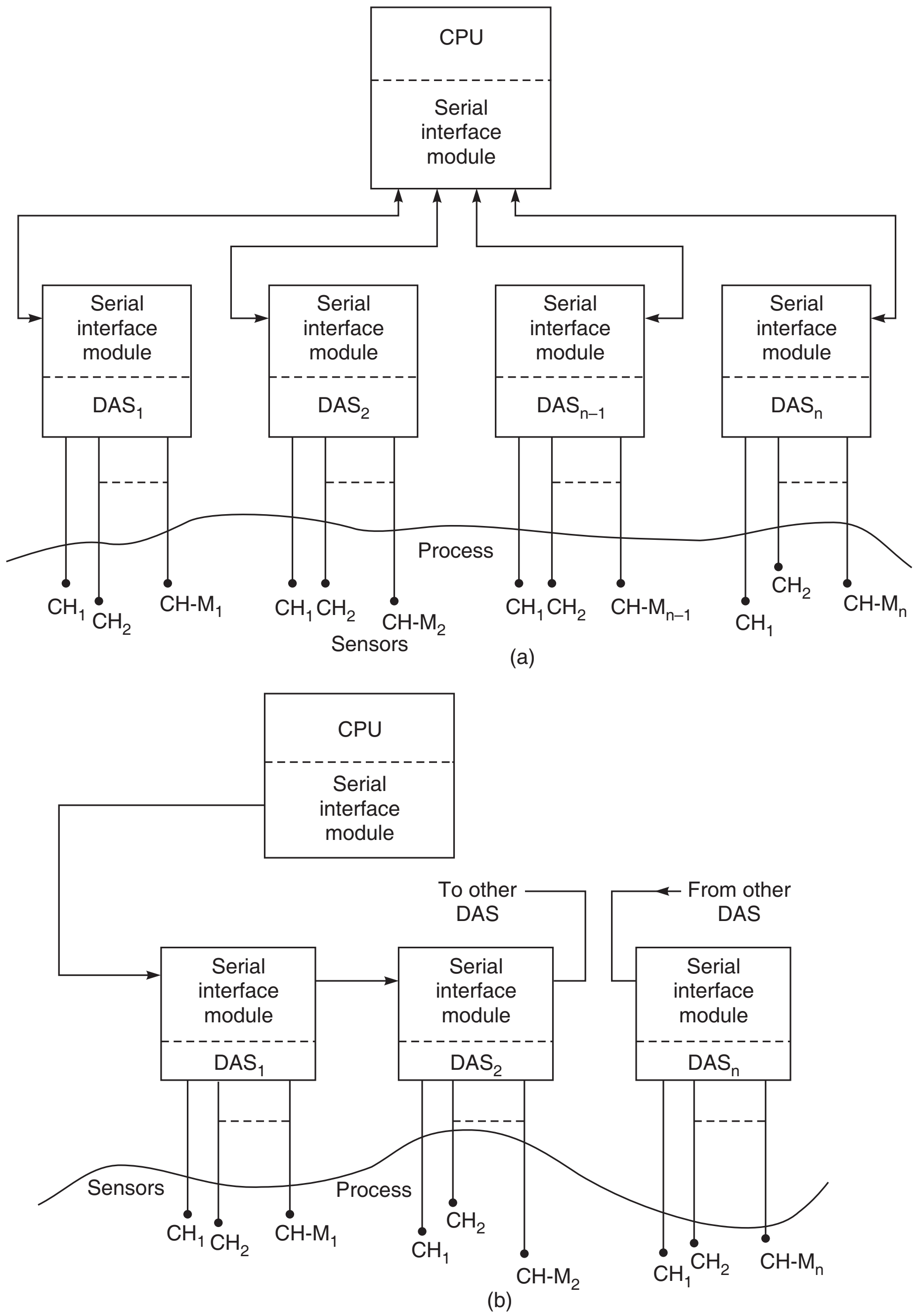


Figure 3.58 (a) Distributed SCADA structure (Star configuration), and (b) Distributed SCADA structure (Daisy chain configuration).

3.7.1 Input/Output Modules

Input/Output modules contain analog input modules, analog output modules, digital input modules and digital output modules. These modules have already been described.

3.7.2 Communication Module

The communication module is the most important portion of remote terminal unit and has the interface available with 2-wire/4-wire communication line. Some of the RTUs may also have built-in transceivers and modems. Following are the basic communication strategies that a RTU may use depending on the application need:

Wireline communications

The wireline communication may have number of options and these options can be selected depending upon the distance between central computer and RTU or between two RTUs. These options are enlisted here:

(i) *Option 1—RS-232C/442.* RTU can support communication via standard RS-232C/442. The I/O ports can select the average levels as well as the baud rates.

(ii) *Option 2—Switch line modem.* When the user wants to use the existing telephone lines for communication, the switchline modem can be effective. Such RTUs contain the facilities like auto answer, auto dial and auto select baud rates. The modem is ideal for data networks configured in time or event reporting RTUs and for master station polling networks.

(iii) *Option 3—2-wire or 4-wire communication*. The modem residents in the RTU can be configured to 2 or 4-wire communication on dedicated lines. The same communication protocol is used for all devices making the actual network configuration transparent to the user.

Terrestrial UHF/VHF radio with store and forward capability

The RTU may support a complete line of UHF/VHF terrestrial radios. The communication protocol in these RTUs is transparent to the user and supports CRC intelligence, error checking, packet protocol for error free data transmission. The store and Forward capability of the RTU minimises the required input in large numbers.

Satellite communications

In the applications where wireline and terrestrial radio communications are impossible or cost prohibitive, the satellite communications may be desirable. Some of the latest RTUs provide the facility to be interfaced to one-way or two-way satellite communication using Very Small Aperture Terminals (VSATs). These terminals use one metre antennas and have data rates from 50 to 60 Kbps (Kilo bits per second).

Fibre-optic communications

For applications where electromagnetic interferences or hazardous electrical potentials exist, the RTU can be networked using fibre-optic cables. The same communication protocols and networking concepts available for wireline and radio are used for fibre-optic communication.

3.7.3 Special Software Facilities

The special software facilities which are required in RTUs and are not available in SCADA units mentioned above are given here.

(i) *Quiescent Mode Operation.* Since the transmitter consumes maximum power in RTU in the communication especially in terrestrial and satellite communications, it is switched on only when the RTU has some information packets ready for sending. Thus the RTU receives all the information from central computer. It may be a polling message or a request to perform certain control functions. Since the receiver is kept on all the time, these information are received and proper action is initiated. Only when the RTU has to send some information to central computer which may be an urgent message or an acknowledgment of the action taken or acknowledgment of message received, the transmitter is switched **on.** The quiescent mode saves considerable amount of power for RTUs.

(ii) *Down loading of limits from Central Computer.* Generally the RTUs behave much like stand alone SCADA or direct digital control systems. Thus they collect the data from various sensors, perform signal conditioning, filtering, conversion to engineering units and store them in the memory. They also perform the limit checking on these values and inform central computer on violation of limits, if any, immediately. Since these RTUs are at remote locations it should be possible to change these limits remotely from central computer. This is called *Down Loading of Limits.* The central computer in such cases makes a special request to RTU to change the limits. The RTUs will then enter a special mode for the change of basic parameters and perform the functions in an interactive way.

(iii) *Exceptional Reporting.* The microprocessor based RTUs are intelligent enough to perform number of functions independently and the need to conserve power in these remote locations is paramount. This means that the RTUs must communicate to central computer mostly in receive mode and only exceptionally in transmit mode. The RTUs normally have intelligence to perform all the functions including limit checking and only when the limits are violated the central computer is informed. The other message that goes regularly is regarding "all well" condition of RTUs. Thus RTUs perform self-diagnosis which must be done by the resident microprocessors by executing different diagnostic software.

Figure 3.59 shows the block diagram of RTU. The software environment includes real-time operating system and real-time language to facilitate the system development. The concepts of real-time operating system and real-time language have been discussed in separate chapter in this book.

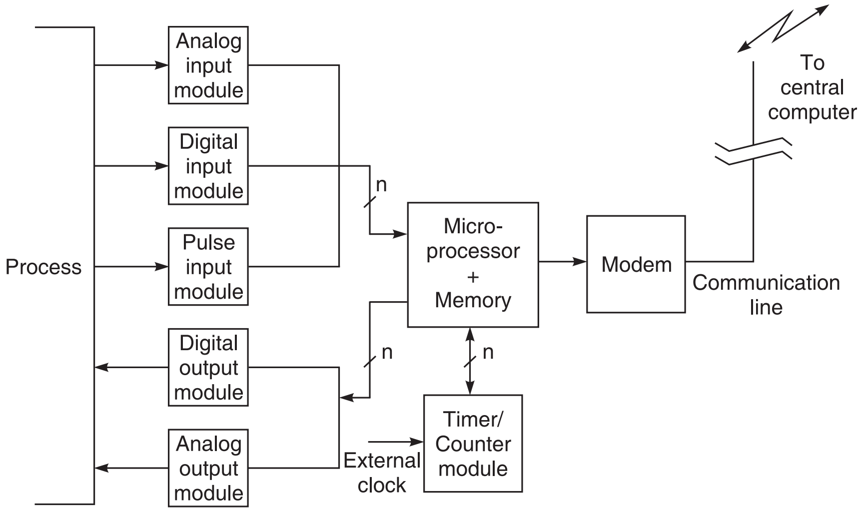


Figure 3.59 Remote terminal unit (Block diagram).

3.8 RELIABLE SYSTEM DEVELOPMENT STRATEGY

Real-time system operation demands high level of availability as well as error-free operation. The real-time process control systems directly control the process, and any system failure and/or erroneous operation may lead to process damage and may also affect the safety of operators. Thus, these applications demand high level of system reliability. Numerically, system reliability is defined as the probability that the system will not fail under specified conditions.

Standards and guidelines are available for the development of safety critical systems, the most relevant being IEC 61508. These have been defined by military authorities and also the railways and aerospace industries. These development standards may be used while designing a process control system. In keeping with these guidelines, an analysis must be performed at the early stage of system design in order to assign a system integrity level which allows us to define the accepted failure rate of system under consideration.

3.8.1 Causes of System Failure

A system will consist of a number of subsystems. As an example, a digital stand-alone system will have I/O modules, memory modules, CPU card, I/O connections, power supply, etc. A process control system consists of sensors, controllers, control valves, motors, limit switches, power supply as well as their interfaces to the actual process being controlled. The causes of failure may be categorized as either common cause or common mode. Common cause failure covers those causes which will affect a specific system at a time. These include component failures, faults in supply lines of water, gas, etc., power supply failure, cable fault, and faults in other shared resources within a system. By having more than one identical unit in parallel, i.e., by providing redundancy, common cause failures can be reduced. Common mode failures, on the other hand, affect more than one system at any instant. As an example, an ESD (Electro

Static Discharge) will affect two identically redundant outputs. If there are two redundant digital units and one of the units starts behaving erratically due to some trisient signal, then in all probability the other unit will also behave in that manner.

Such type of failure may be reduced by providing diversity, i.e., by having units of different design but performing the same task in parallel.

Thus, providing some amount of redundancy seems to be a viable solution. However one should not forget the following two cardinal principles of reliable design:

(i) Use the most reliable components;

(ii) Use the least possible complexity consistent with required system performance.

It has been well established that redundancy as a tool to increase reliability is more effective in a system using reliable components than in a system with unreliable components. Thus redundancy will be of no use in increasing system reliability if the non-redundant system is unreliable in the first place. There are a number of relevant terms such as Fail-Safe System, Fault Tolerant System, Graceful Degradation, Double Modular Redundancy, and Triple Modular Redundancy. We shall deal with these in terms of increasing levels of complexity.

3.8.2 Fail-Safe System

In many real-life situations, the failure of a system may not only cause interruption in the operation but may also result in catastrophy, i.e., damage to other devices and human beings. As an example, if the traffic light controller fails and it results in green light in all directions, considerable damage through accidents may be caused. Similarly, in the case of gas supply control in a petro-chemical plant, system failure may result in damage to the plant and also the operators.

In the first case, the failure of the traffic control system should result in blinking red light, whereas in the second case the failure in gas supply control should result in gas flow reduced to zero and system shutdown. In addition, high-priority alarms will also be required. It must be understood that failures are inevitable in spite of good design. These may be due to wear and tear of components, environmental factors as well as human error.

A fail-safe system describes a feature or a device which, in the event of failure, responds in a way that will cause no harm or minimum harm to other devices or danger to personnel. Fail-safe systems are used wherever the highest degree of safety needs to be guaranteed for humans, machines and the environment. This means that accidents and damage as a result of fault must be avoided at all costs. Thus, when controlling dangerous or critical machinery, it is necessary to devise and implement fail safe strategies to ensure that the machine operates safely even when elements of the control hardware or software fail.

A viable strategy for a stand-alone system without redundancy is to define a safe state for all control outputs. Thus, a designer can create a control system with high degree of immunity from hardware and software problems. It is necessary that Fail-Safety analysis be conducted for the whole process before undertaking design of such a system. It must be validated for the complete process, of which fail safe control system hardware and software are components. The system integrators are responsible for defining the safety requirements and desired system behaviours under fault conditions and validating these requirements for their specific applications.

National Instruments offers the CompactRio fail-safe control system based on the above strategy. CompactRio uses FPGA (Field Programmable Gate Array) to define the safe state of all control outputs within itself. The I/O channel passes through FPGA, which is the most reliable part of the system. Thus in case of faults, the FPGA will steer the controller output to safe state. However, the process/system and the FPGA should not fail at the same time. Another strategy adopted is to provide redundancy, as we shall see in the forthcoming sections.

3.8.3 Fault-Tolerant System

Fault tolerance is the property that enables a system to continue operating in the event of failure of some of the components. In a networked environment like the distributed control system, a fault in one system (e.g., a packet fault) may get transmitted to another system and cause failure in that system. Thus, fault tolerance must be a property of individual systems as well as the environment through which they interact.

A viable strategy for fault tolerant system design comprises anticipating exceptional conditions and building the systems to cope with them. The basic aim is that the system should be able to self-stabilize after the fault and converge towards an error-free state.

The above strategy may not be successful in some cases. In such applications, fault tolerance is implemented by providing redundancy, as we shall see in the forthcoming sections.

3.8.4 Graceful Degradation Systems

Graceful degradation is the ability of the system to maintain limited functionality even when a portion has been rendered inoperative. Whenever one sub-system of the overall system fails, the tasks are allocated to other sub-systems through a crossover switch (Fig. 3.60). The purpose of graceful degradation is to prevent catastrophic failure.

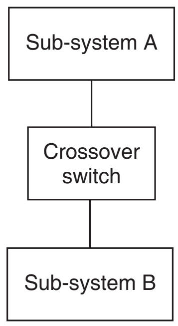


Figure 3.60 Graceful degradation strategy.

In the first strategy, when a sub-system fails, the tasks performed by the sub-systems are allocated to other sub-systems. This causes performance degradation but all the features of the system are maintained. The performance degradation is proportional to the severity of failure. In the second strategy, when a sub-system fails the tasks performed are reallocated in such a way that critical features are maintained with no degradation but non-critical features are temporarily shed.

Graceful degradation strategy has been extensively used in the design of the network of the US Advanced Research Project Agency (ARPANET) which is the predecessor to the Internet.

It is imperative to create a scaleable model of the system to implement the graceful degradation strategy.

The structural details of the graceful degradation system is shown in Fig. 3.61. The system is capable of performing tasks A1, A2, A3, A4, and B1, B2 , B3, B4. These tasks are performed by two sub-systems, A and B. In normal course, sub-system A performs tasks A1 to A4 and sub-system B performs tasks B1 to B4. However, both the sub-systems have the necessary software to perform all the tasks. Thus, if required, all the tasks can be performed by either sub-system A or sub-system B.

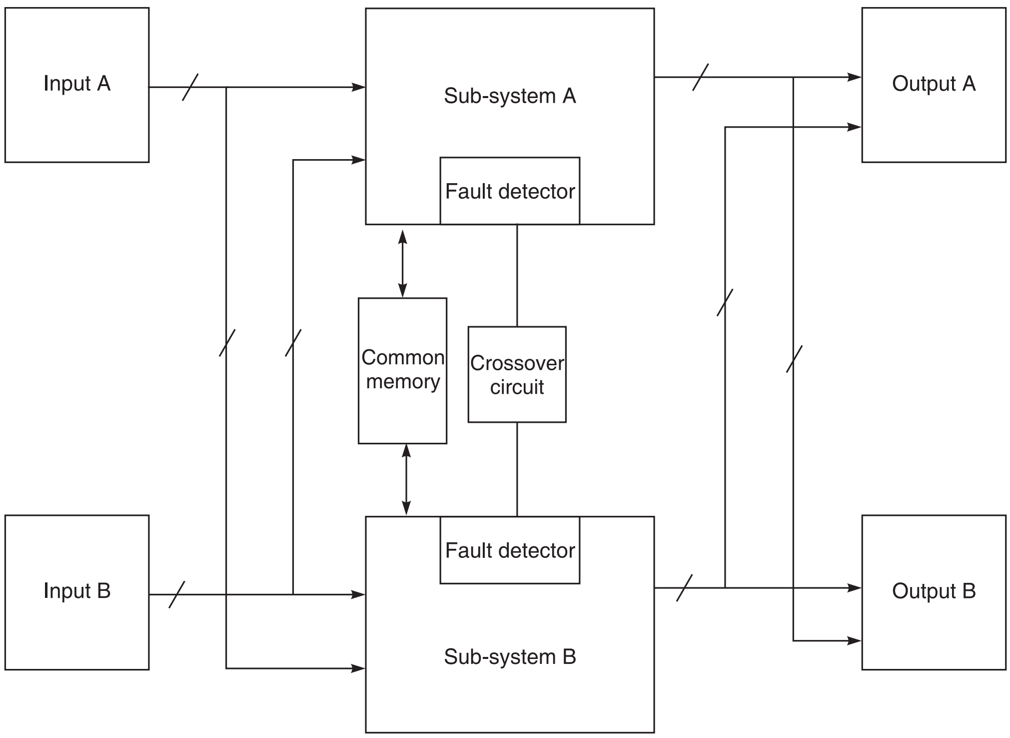


Figure 3.61 Graceful degradation system structure.

The inputs corresponding to tasks A1, A2, A3 and A4 are fed to sub-system A, whereas inputs corresponding to tasks B1, B2, B3 and B4 go to sub-system B. Similarly, sub-system A controls some output points and sub-system B controls some other output points. However, these input–output interfaces are maintained in both the sub-systems. Thus, either of the sub-systems can accept all the inputs and control all the output points. Both the sub-systems maintain common memory, which have all the necessary data. We have already dealt with dual port memory and multibus structure in previous sections.

The fault detection circuit at the module level is contained in both the sub-systems. Whenever a sub-system fails, the fault detection circuit sends alarm and activates the crossover circuit. When the crossover circuit is activated, all the input and output units get connected to the healthy sub-system. The healthy sub-system also gets a message that it has to perform all the tasks, i.e., all the software modules are to be executed if strategy 1 is followed or some specific software module is to be executed in case of strategy 2 described above.

Since only one sub-system performs all the tasks, performance degradation will take place. We have described a system consisting of only two sub-systems with crossover circuit. The system may contain a number of sub-systems A1, A2, … and B1, B2, ... with a crossover circuit between them. The graceful degradation system strategy is used when complete replication of the system is cost-prohibitive.

3.8.5 Lockstep System Concept

The term *lockstep* has originated from prisons, where marchers walk in synchronism and as closely together as physically practical. The concept has been used to achieve reliability through redundancy. The term *lockstep systems* refers to redundant systems that run the same set of operations at the same time in parallel.

Each system progresses from one well-defined state to the next well-defined state. When a new set of inputs reaches the system, it will process them, generate new outputs and update its state. The outputs from lockstep systems can be compared to determine if there has been a fault. Further decisions on action will be based on the type of fault and the lockstep system configuration. Dual Modular Redundancy (DMR) and Triple Modular Redundancy (TMR) are two popular configurations of lockstep systems.

3.8.6 Dual Modular Redundancy (DMR)

A system with two replications of each sub-system or element is termed Dual Modular Redundant (DMR). It is also known as 1oo2 or *1 out of 2 system*. The same inputs are provided to each replication and the same outputs are expected. The outputs are compared using a voting circuit. In case of mismatch of outputs, fault is declared and the faulty system is switched off and taken for repair, whereas the normal control operation is continued with the other system.

You must recognize that at many instances it will become difficult to arbitrate two systems (to find the faulty system), when their outputs differ, at the end of a step.

It is therefore common practice to configure DMR systems as master/slave rather than in lockstep. The slave unit acts as a *hot standby* to the master (Fig. 3.62). In normal course the master processes all the outputs, even though the connections to the input–output units are duplicated to each system. At regular intervals the master copies its state to the slave unit. In case the master fails, it switches itself off and the slave unit is ready to continue from the previous known good step.

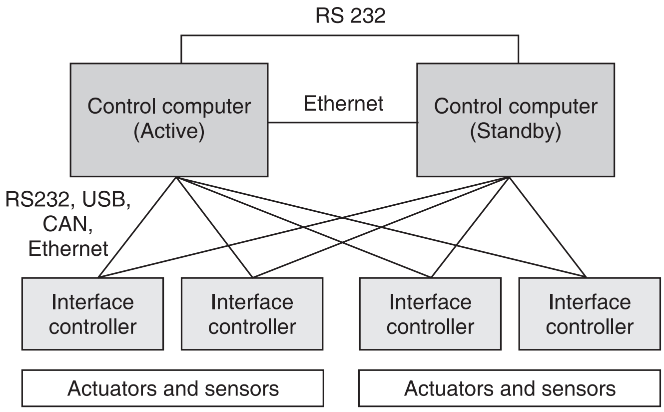


Figure 3.62 Dual modular redundancy.

3.8.7 Triple Modular Redundancy (TMR)

In case of Triple Modular Redundancy, the systems are triplicates, i.e., 3 systems or sub-systems operate in parallel. They receive the same set of inputs, perform the same operation and therefore the same outputs are expected from them. The outputs are fed to a voting circuit. If the output from one unit disagrees with the outputs from the other two, then the unit is assumed to have failed and the matched output from the other two is treated as correct.

The concept of Triple Redundancy mode was laid out by von Neumann, the renowned mathematician and computer pioneer. The TMR concept of von Neumann is shown in Fig. 3.63. What we call *voting circuit* was called *majority organ* by von Neumann. It accepts input from three sources and delivers the majority opinion as an output. The TMR system is also known as *2oo3* or *2 out of 3 system*.

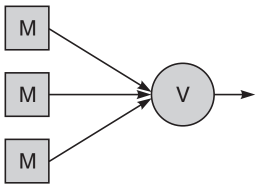


Figure 3.63 Triple modular redundancy—basic configuration.

In the above configuration, the voting circuit is not expected to fail and thus becomes the weakest link. An alternate TMR configuration proposed is shown in Fig. 3.64. It employs three identical voting circuits instead of one voting circuit, providing redundancy at the level of voting circuit as well. Hence a single voting circuit failure will not cause system failure. It must be understood that if we assume the voting circuit (V) to be fail-safe, then the reliability of the two configurations will be the same.

The advantages of the TMR architecture as compared to the DMR architecture are as follows:

- 2 out of 3 voting provides superior fault detection of all I/O, hardware and control algorithms.
- It eliminates single-point failures.
- Reliability is vastly improved by extending fault coverage to nearly 100%.
- There is greater flexibility for implementing a variety of fault-tolerant configurations.
- There is superior latent fault detection.
- It provides on-line serviceability.

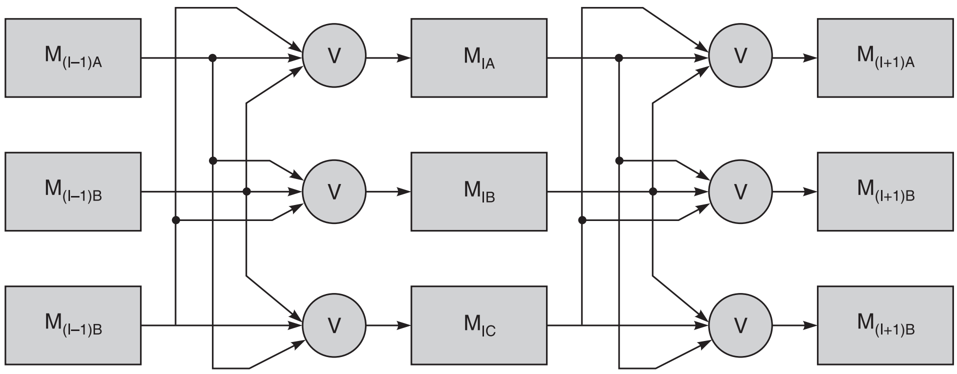


Figure 3.64 Triple modular redundancy with voting circuit redundancy.

3.8.8 Examples of TMR Systems

A number of control system manufacturers have developed control systems based on the Triple Modular Redundancy concept.

IC Systems have developed the Trusted Industrial Control System. At the heart of each trusted system is the controller chassis which houses the TMR processor, its companion slot for bumpless change to a spare, and 8 interface slots. These slots accept I/O modules, communication modules, or interface modules to expanders. A wide range of dedicated I/O modules are available. These include both analog and digital modules as well as a unique universal I/O module that allows mixing of the signal type on the same module. The TMR concept is implemented in each of the modules.

Figure 3.65 shows the design at the T8451 trusted TMR 24 VDC digital output module. The module contains interfaces to 40 field output devices. Fault tolerance is achieved through Triple Modular Redundancy (TMR) within the same module for each channel. The module provides on-board Sequence of Events (SOE) reporting with a resolution of 1ms. The “event” is the actual sensing of the field device loop current.

Each triplicate section (slice) of the module receives output data from the Trusted TMR Processor and sends its data to an internal safety layer hardware voter. Each slice uses the voted data to control the outputs. Every output channel consists of a six-element voted array. The output switch array provides fault-tolerant uninterrupted control under an output switch failure condition.

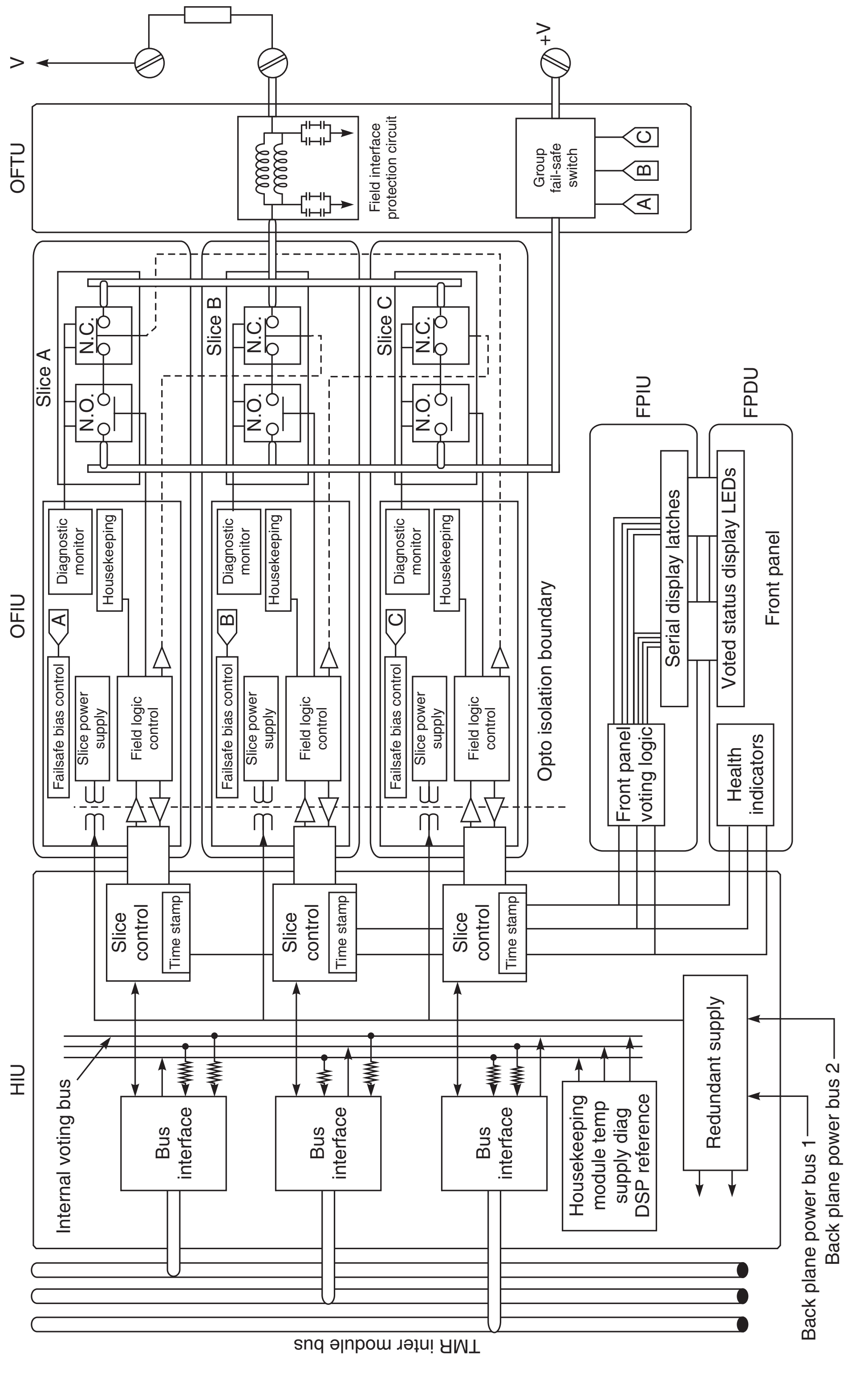


Figure 3.65 T 8451 trusted TMR 24 VDC digital output module.

When a module fault is detected, the Trusted TMR Processor will flag the unhealthy condition, indicating the need to replace the module. Control continues until a replacement (healthy) module is available. Replacement modules can be located in a standby slot next to the module or in a designated SmartSlot with a temporary connection to the same Trusted Termination Assembly.

A range of software modules are available with the hardware. These include software packages for process control algorithms, peer to peer communication, process historian package, toolset drivers, etc.

The MicroNet TMR control system incorporates the features of the MicroNet described above in a Triple Modular Redundant (TMR) control architecture. The MicroNet TMR uses the Motorola CPU architecture with double exchange voting. It consists of three isolated kernel sections. Each section includes its own CPU, CPU power supply, and up to four I/O modules. The I/O modules can be used for simplex I/O, redundant I/O, triple redundant I/O, or any redundancy combination. Each kernel I/O section is expandable into one or more of the MicroNet chassis. Interface modules provide inter-rack communications.

The kernel sections individually monitor all input data, perform all application calculations and generate all output values and responses. Outputs are assessed with the *2 out of 3* voting logic. With this configuration, any fault or number of faults associated with a kernel can be tolerated without affecting the system operation.

The high-density MicroNet cards provide first-out indication for monitored system events to reduce troubleshooting time. These cards will timestamp the event within 1 ms for discrete inputs and 5 ms for analog inputs.

The MicroNet TMR uses two power supplies, each of which powers the control from a separate power source. Inside each power supply are three independent power converters, one for each CPU and I/O section. The triplicated power architecture provides maximum protection against hardware failures. It also provides full TMR relay modules for critical discrete outputs. A six-relay configuration accomplishes this. If any one relay fails in a normally open configuration, or if any two relays fail in a normally closed configuration, the contact path is not interrupted and the fault does not interrupt normal operation. Latent fault detection is used to monitor and detect any relay faults.

The MicroNet TMR control will drive multiple actuator coils and current drivers to support double-redundant and triple-redundant field devices.

The High-Integrity Control System architecture of RTP Corporation is extremely scaleable and flexible, and incorporates multiple levels of redundancy. Its unique design is built upon a set of standard hardware and software components that supports multiple factory-configured applications. These standard components (processors, chassis, I/O and configuration environment) are designed to be configured in different systems ranging from single-controller or dual-redundant up to a triple-modular redundant solution.

RTP has created a new TMR architecture designed specifically for process control. The 2300-T HICS (High Integrity Control System) ensures the highest integrity and the highest availability of user data. It utilizes voting routines to ensure that the process is protected and downtime is eliminated.

A 2300-T has more flexibility than traditional TMR systems. Where past TMR systems were limited because of the hardware used to synchronize CPUs and because their I/O was not distributable, RTP uses high-speed networking technology including high-speed logic switching and a fast Ethernet-based I/O bus. As a result, the three controllers that comprise the 2300-T can be housed separately to ensure survivability in a catastrophe such as fire. Additional I/O chassis can be located remotely from the three controllers. To ensure added availability, a dual-powered chassis can be added for both processor and I/O.

Where requirements may dictate several redundant processors, the 2300-T ensures process integrity and availability for less cost and less complexity. High Integrity Validation Redundancy protects against hardware failure for high system availability, but does not provide for error checking. For critical operations such as program changes, special routines check and recheck the program validity. Outputs from all three processors are sent to the I/O chassis where the voting occurs as close to the process as possible.

RTP offers a complete family of high-integrity voting systems for process control.

2300-T: For the highest level of integrity and availability, the 2300-T Triple Modular Redundant system features voting with triple, dual or common I/O as required.

2300-D: The 2300-D HICS is built on the same advanced technology as 2300-T for a dual-redundant solution. The voting solution uses advanced diagnostics to assist in results adjudication. The 2300-D features dual-redundant processors, triple, dual or common I/O as required, and up to 32 remote I/O chassis all built upon the same components and configuration environment (RTP NetSuite) as the 2300-T.

Both the 2300-T and 2300-D support a medium to large distributed control system. It has an architecture of up to 10,000 I/O points per node. For smaller systems, the Micro2000D series HICS also provides a voting solution using dual processors and I/O. The Micro2000D series uses a subset of the same components as the 2300-T and RTP NetSuite.

The 3420 is a configurable steam turbine controller that utilizes the proven Tricon V9 Triple Modular Redundant (TMR) fault-tolerant hardware for the control system platform. The system uses 2 out of 3 voting to provide high-integrity, error-free, uninterrupted process operation with no single point of failure within the controller chassis.

The core turbine control program can be configured to start, warm up, accelerate and run most one and two-valve steam turbines. It allows testing of overspeeding trip devices and provides several options for tripping the turbine. It is also programmable. The core control program can access half of the installed I/O. Custom programs can be added to the core control program to access the remaining I/O. These custom programs can be used to start and stop lube oil pumps and sequences of turning gear hydraulic system or to control a pressure-reducing valve for the steam system.

The 3420 provides automatic control of start-up, loading and shutdown of both one and two-valve steam turbines. The two-valve configuration allows control of extraction and/or induction applications. The 3420 has a number of control features that allow it to protect and control both the turbine and the driven equipment.

Egypt Basic Industry Corporation has implemented the Triconex TS 3000-Triple Modular Redundant System for Turbine Compressor Control at the largest ammonia plant of Egypt.

Three Triconex TS3000 TMR systems are deployed for air compressor, feed gas compressor, SynGas compressor and ammonia refrigeration compressor controls. System functions include governor, surge control, equipment protection, vibration monitoring and control, and monitoring of auxiliary equipment such as lubrication systems.

For emergency shutdown, two Triconex TMR systems are deployed to protect the entire main plant area as well as the dock area at Sokhna Port. The five TMR systems connect with more than 1,100 I/O points around the complex. It has resulted in significant benefits such as simplified engineering, reduced spare parts requirement and efficiencies in operator and maintenance staff training.

3.9 CONCLUSIONS

In this chapter, we have dealt with basic building blocks of automation systems excluding sensors and actuators which are described in separate chapters. The selection of microprocessor, SCADA, RTU must be directed by application demand. In future, more and more facilities would be getting integrated in the intelligent systems. In very near future microcontrollers integrated with silicon sensors and digital actuators may he designed for specific applications.

SUGGESTED READING

Ahson, S.I., *Microprocessors with Applications in Process Control*, Tata McGraw-Hill, New Delhi, 1984.

Andrews, D.W. and Schultz, G.D., A token-ring architecture for local area networks, *Proc. COMPCON*, pp. 615–629, 1982.

Andrews, M., *Programming Microprocessor Interfaces for Control and Instrumentation,* Prentice Hall, Englewood Cliffs, New Jersey, 1982.

Artiwick, B., *Microcomputer Inteijacing*, Prentice Hall, Englewood Cliffs, New Jersey, 1980.

Can; J.J., *Design of Microcomputer-based Instrumentation*, Prentice Hall, Englewood Cliffs, New Jersey, 1982.

Fletcher, W.I., *An Engineering Approach* to *Digital Design*, Prentice Hall, Englewood Cliffs, New Jersey, 1984.

Franta, W.R. and Chalamtac, I., *Local Networking: Motivation*, *Technology and Performance*, Lexington Books, Lexington, Massachussetts, 1981.

Garett, P.H., *Analog Systems for Microprocessors and Minicomputers*, Prentice Hall, Englewood Cliffs, New Jersey, 1978.

Hordeski, M , Interfacing minicomputers in control systems, *Instrumerzts and Control Systems* (*USA*), **51**, No 11 , pp. 59–62, 1987.

Kant, K., *Microprocessor-based Data Acquisition System Design,* Tata McGraw-Hill, New Delhi, 1987.

Katz, P., *Digital Control Using Microprocessors,* Prentice Hall, Englewood Cliffs, New Jersey, 1981.

Keiser, E.G., *Local Area Networks,* McGraw-Hill, New York, 1989.

Kompass, E.I., A microprocessor in every control loop, *Contr. Engg.,* **23,** No. 9, pp. 123–126, 1976.

Laduzinsky, A.J., Local area networks expand the horizon of control and information flow, *Contr. Engg.,* **30,** No. 7, pp. 53-56, 1983.

Lorifeme, B., *Analog-Digital and Digital-Analog Conversion,* Heyolen, London, 1982.

Matley, J. et al. (Eds.), *Practical Process Instrumentation and Control,* Vol. 11, McGraw-Hill, New York, 1986.

Mitzutoni, M. and Smith L.E., Using microelectronics in single-loop controller, *Proc. 3rd Ann. Contr. Engg. Conf,* Barnington, pp. 238–243, 1984.

Needham, R.K. and Herbert, A.J., *The Cambridge Distributed Computing System,* Addison-Wesley, Reading, Massachussetts, 1982.

Steckohn, A.D. and Den Otter, J., *Industrial Applications for Microprocessor,* Prentice Hall, Englewood Cliffs, New Jersey, 1982.

CHAPTER

4

Final Control Element

4.1 INTRODUCTION

The final control elements play a very important role in controlling a process. The controller after deciding about the control action, executes the action on the process using the control element. Figure 4.1 shows the control element, sensor, comparator and controller with a process. The sensor senses particular parameter and sends it to the comparator in acceptable voltage/current form. The comparator compares the parameter value with set-point and determines

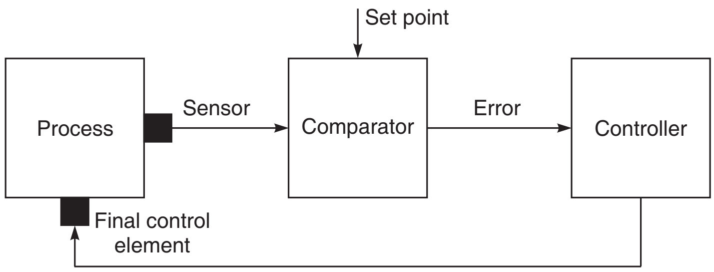


Figure 4.1 Final control element in process control loop.

the error. In order to make the error zero, the controller determines the control action using one of the control strategies, namely proportional, proportional and integral or proportional integral and derivative. The signal corresponding to control action is sent to control element for controlling the process. The control element consists of two parts, namely, control valve and actuator. The control actions may be divided into two categories.

— Start/stop of a sub-system

— Variation of process parameters

Figure 4.2 explains the actuator and control valve placed in the control loop. The actuator accepts the signal from controller and then performs control action using the valve. The actuator may be pneumatic, hydraulic, electric, electronic or digital.

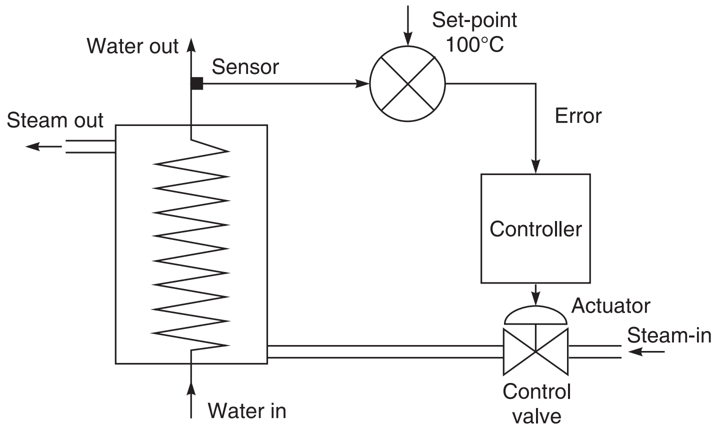


Figure 4.2 Actuator and control valve in control loop of water heating system.

Generally, pneumatic and hydraulic operations are often referred to as same. The components used in both the cases and principle of operation are very similar.

4.2 PNEUMATIC ACTUATION

Pneumatic operation comprises several forms of technologies. This branch of technology is associated with industrial processes in the similar manner as electronic components with production of different equipments like TV receivers, radar, computers etc. Compressed air technology (Pneumatic) possesses a number of special characteristics that distinguish it from Hydraulic. Pneumatic Control Systems operate on a supply of compressed air which must be made available in sufficient quantity and at a suitable pressure according to the capacity of the system. A compressor is a machine which takes in air, gas or vapour at certain pressure and discharges the fluid at higher pressure. Thus the key part of pneumatic facility is the supply of compressed air from the compressor. Just as the electric power supply is the source of energy for all electronics and electrical equipments, compressed air is for Pneumatic equipments.

Depending on the compressor type, capacity may range from a few litre per minute to as much as about 50,000 cubic metres per minute. Discharge pressures may be between a few mm. of water to more than thousand bars. Pneumatic control normally works with air at pressure of around 6 bar. The limits extend from about 3 bars minimum to about 15 bars maximum.

4.2.1 Pneumatic Cylinders

A cylinder comprises a piston provided with a circular piston rod moving in a closed barrel. By supplying compressed air to either side of the piston, one can make it move i.e. the cylinder performs its stroke. Figure 4.3 shows the working principle of a cylinder. If compressed air is supplied to the rear cylinder chamber, the piston and the piston rod will move through the barrel towards the front end cover. If the air is supplied to the front cylinder chamber, the movement will be in opposite direction, i.e. the piston and rod will move towards the rear end cover.

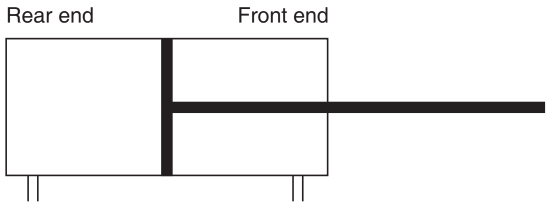


Figure 4.3 A pneumatic cylinder.

4.2.2 Single Acting Cylinders

A single acting cylinder is capable of performing an operating motion in one direction only and equipped with only one inlet for the operating air pressure. The return stroke is normally achieved by providing a spring (Fig. 4.4). They are produced in several designs, like Diaphragm cylinders, Rolling diaphragm cylinders and Piston type cylinders. The piston type cylinder is encountered most frequently in pneumatic control applications. Due to only one inlet for operating air pressure, these cylinders are limited to working stroke in one direction. Since they develop power in one direction only, heavy control equipment could not be attached to them which requires to be moved on the piston return stroke. The field of applications includes embossing, riveting, stretching, etc.

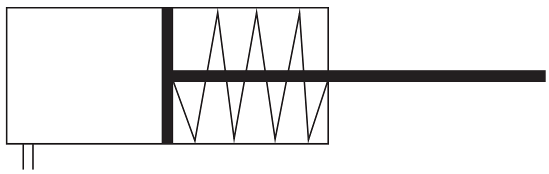


Figure 4.4 Single acting cylinder.

4.2.3 Double Acting Cylinder

Double acting cylinders are always of the piston type and are equipped with two inlets for the operating air pressure, one on either side of the piston. These cylinder are capable of performing operating motion in both possible directions of piston movement (Fig. 4.5). If compressed air hits the surface of piston and pressure is released from the opposite volume, the piston rod moves out. Whereas, if compressed air hits the reverse side of the piston and pressure is released from the opposite volume, the piston returns to its initial position.

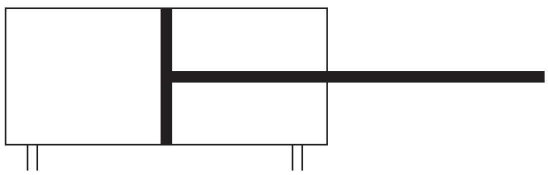
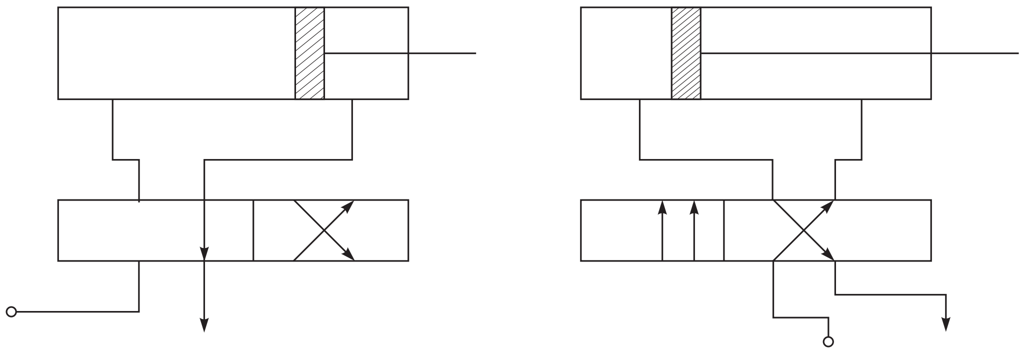


Figure 4.5 Double acting cylinder.

Double-acting cylinders are operated through a pneumatic valve which connects compressed air supply and exhaust to the two parts of the cylinder. While moving in the forward direction (forward stroke), the pneumatic valve connects the rear-end part to the air supply and the front-end to the exhaust; on the other hand, while moving in backward direction (reverse stroke), the supply is connected to the front-end part and exhaust to the rear-end part. The cylinder operation through the 4/2 pneumatic valve is shown in Fig. 4.6.



(a) Forward stroke (b) Reverse stroke

Figure 4.6 Operation of double-acting cylinder through 4/2 pneumatic valve.

The activation for pneumatic valve may be through a manual switch/lever (manually actuated pneumatic valve) or through an electrical signal (electrically activated pneumatic valve or electropneumatic valve). Electropneumatic valves are normally fitted with a solenoid which accepts an electrical signal and moves the valve to make the connections of air supply and exhaust to the desired parts of the cylinder.

4.2.4 Special Type Cylinders

Special type cylinders are designed for specific purposes and are different from regular cylinders. There are many variations in special designs of cylinders.

Cushioned pneumatic cylinders

If double acting cylinders are used for moving large masses then cylinders with final position damping are preferred. Air cushion is provided in some cylinders (called cushioned cylinders) to retard the piston speed as it approaches the terminal end. Figure 4.7 show the operating principle of cushioned cylinder. The main flowpath of the outgoing air is cut off at a distance of one inch from the end of stroke. This is achieved by a cushion nose attached to the piston and a cushion chamber at the end of the cylinder. When piston comes near the end of stroke, the cushion chamber shuts off the direct air outlet. Excess pressure is set up on air cushion in the remaining cylinder volume. At this stage, air may only leave the cylinder through a needle valve restriction. When the movement of the piston is reversed, air may freely enter the cylinder via non-return valve and piston is capable of making stroke at full power and speed. The cylinder can have cushions on both ends of the stroke.

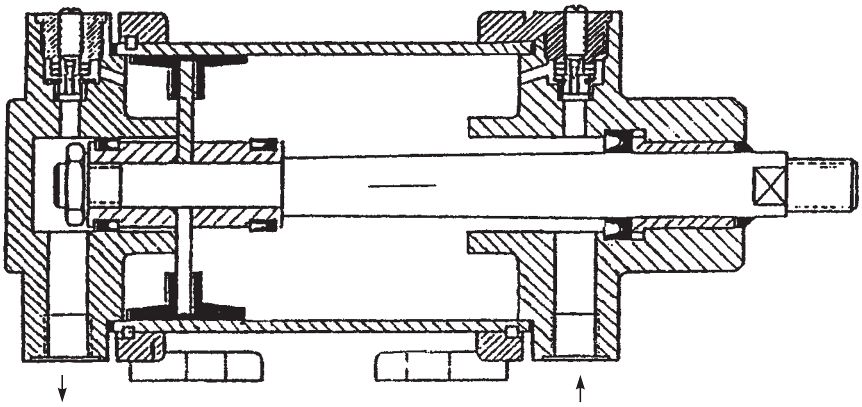


Figure 4.7 Cushioned pneumatic cylinder.

Double rod cylinders

These cylinders have rod extending on both the sides of piston as shown in Fig. 4.8. Since the two piston surfaces are equal in size, equal power is exerted in both directions of movement.

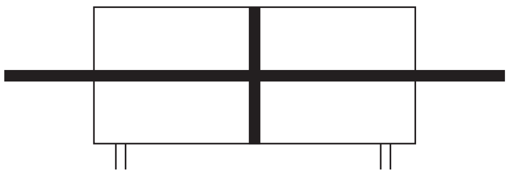


Figure 4.8 Double rod cylinder.

Tandem cylinders

Tandem cylinders (Fig. 4.9) comprise two separate double acting air cylinders arranged in line in one cylinder body so that the power generated by two is added together thereby approximately doubling the piston output. The product of air pressure multiplied by piston area for both pistons is transmitted to the extending piston rod. Tandem cylinders are used in applications which require large amount of power and the space for installation of cylinder of large diameter is not available.

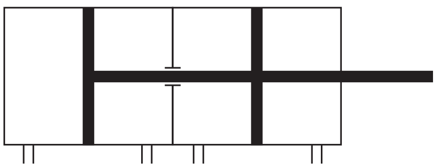


Figure 4.9 Tandem cylinder.

Multiple position cylinders

Multiple position cylinders are a combination assembly of at least two double acting cylinders in one, with pistons and rods in opposed arrangement. This results in a four-position cylinder as shown in Fig. 4.10. It is possible to combine a multiple number of pistons and rods into one cylinder assembly in order to obtain an actuation with six or eight positions. The field of application includes level actuation, sorting devices, etc.

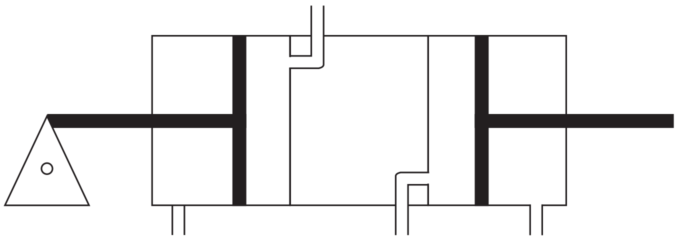


Figure 4.10 Multiple position cylinders.

Impact cylinders

Impact cylinders are designed to achieve high piston stroke velocity resulting in high impact energy of the output power. The cylinder has a pre-pressurizing chamber in which compressed air builds up. The effective surface of the front piston side is smaller than that of the reverse side. Only when the inlet air has built up a certain pressure in this chamber will a seat open, which results in a higher pressure acting suddenly on the piston. This produces a powerful impact stroke, which is effective in one direction only. The return stroke is accomplished as with any normal cylinder. Prime applications of impact cylinders include punching, riveting, beading, stamping, etc.

Telescope cylinders

Telescope cylinders have two or more sleeves which together reduce the collapsed length of a long stroke cylinder. These are usually made with a maximum of about six sleeves and are available as single acting push type as shown in Fig. 4.11. When pressure is applied, the piston inside starts moving out and then follow the subsequent pistons respectively, taken in outward sequence. The telescopic tubes are returned to their initial position by outside forces.

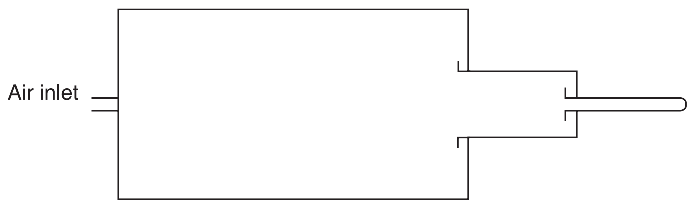


Figure 4.11 Telescope cylinder.

Rotary cylinder

The reciprocating linear movement of the piston is transmitted by a rack portion machined on the piston rod to a gear wheel in order to produce rotary or angular output motion. Rotary cylinders come in different designs as shown in Fig. 4.12. The output motion covered may be a maximum of 360 degrees but is usually smaller in angle, for instance 180 or 290 degrees.

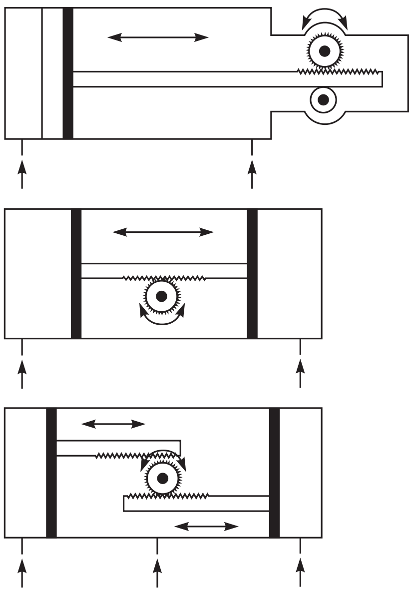


Figure 4.12 Rotary cylinders.

4.2.5 Deciding Factors for Pneumatic Cylinder

1. *Diameter* which is application dependent; force is directly proportional to diameter.
2. *Stroke length* is also application dependent. Usually 0.5 metre maximum stroke length is used. Telescope cylinder is used for more length.
3. *Type of actuation* is one of the deciding factors. An actuator can be single acting/spring return or double acting type.
4. *Port size* may be usually 1/8”, 112”.
5. *Cushioning* or *non-cushioning*—Generally cushioning type.

4.3 HYDRAULIC ACTUATION

Conceptually, hydraulic technology and components of hydraulic actuation are same as in pneumatic actuation. Thus the techniques explained in the previous section on pneumatics are applicable to hydraulic also. The only difference is that while pneumatic works on supply of compressed air, hydraulic works on the supply of compressed oil. We shall, therefore, not be dealing with hydraulic actuation technology separately.

4.4 ELECTRIC ACTUATION

The electric actuation is accomplished by relay, motor, solenoid or thyristor. Since electric and electronic actuators are becoming more prevalent now, we shall describe these in detail.

4.4.1 Relay

A relay may be defined as electrically actuated contact maker or breaker. The design of relay has been shown in Fig. 4.13. It consists of an iron bar with a coil and an armature mounted on base. This arrangement provides a fixed contact through base and movable contact through iron. When current flows through coil, the iron bar becomes magnetic and armature gets attracted towards that. As a result of this, contact closure takes place and the controlled circuit between movable contact and fixed contact is switched on. It is possible to have ‘make arrangement’ to affect more than one contact closures (changeovers) in the same relay.

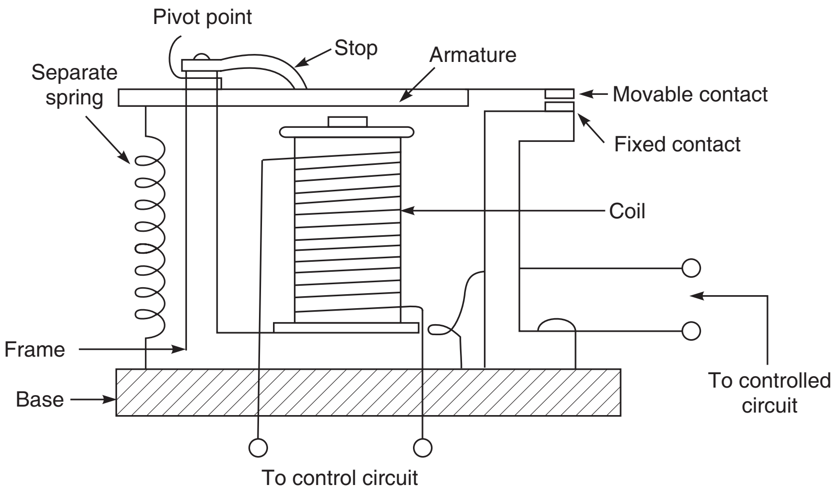


Figure 4.13 Design of relay.

Figure 4.14 shows a relay with three changeovers. A number of operations like make, break, make or break, make before break and break before makes break are possible. These have been shown in Fig. 4.15. The use of relay in actuation has been explained in Fig. 4.16. Basically relay is used as a device to switch on power to circuit remotely. When the control switch closes, the relay contact is made, causing current to flow in control circuit.

In a number of applications, specially when relay is interfaced to computer, the basic requirement will be that of a control circuit which is not disconnected, even when the computer has disconnected itself from the relay circuit. This requirement is fulfilled by Latching Relay (Fig. 4.17). When switch A is closed, the relay is operated and contacts are made. At this instant, current finds an alternate path from the coil and closed contact *X*. As a result, coil remains magnetized even when A is opened and thus the contact *Y* remains closed. The coil can be demagnetized only by switch B.

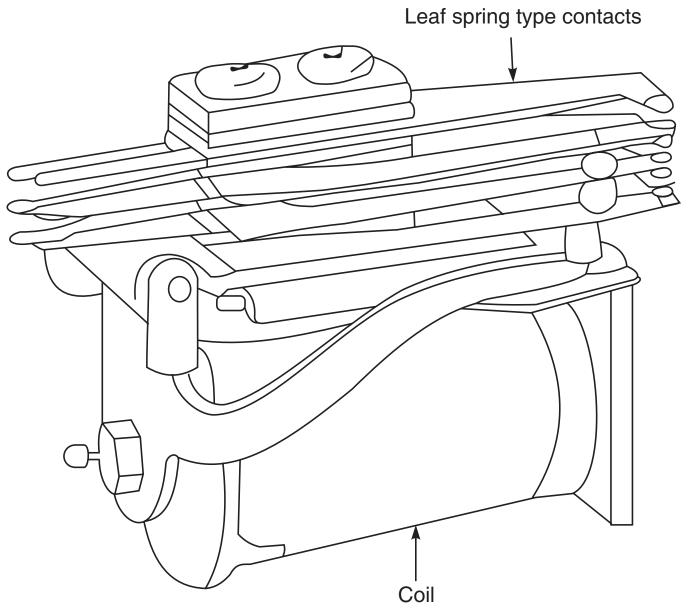


Figure 4.14 Relay with three changeovers.

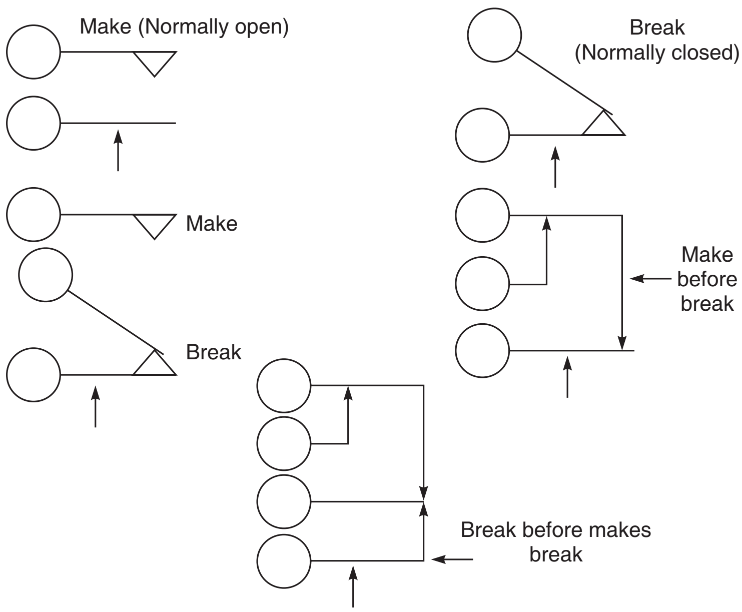


Figure 4.15 Typical relay contact arrangements.

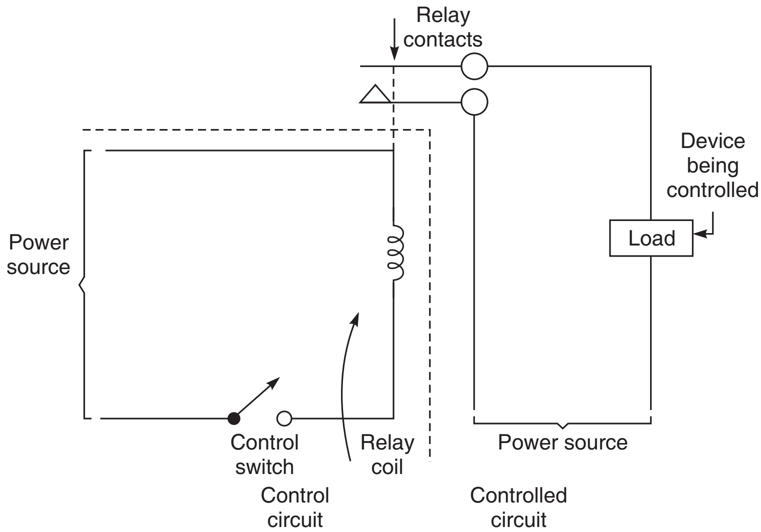


Figure 4.16 Relay as actuator.

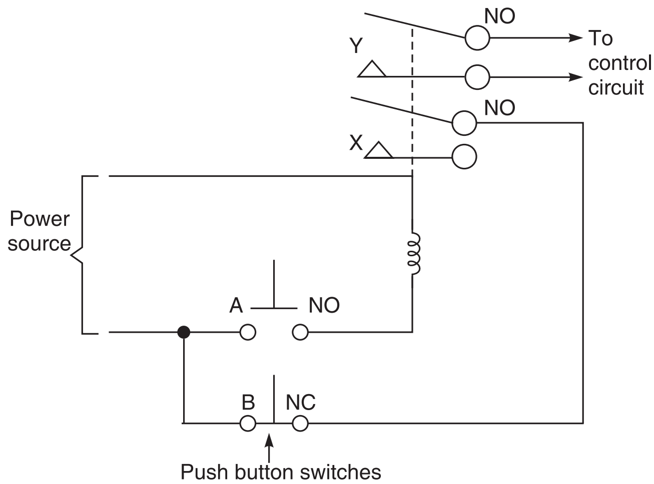


Figure 4.17 Latching relay.

Relay is normally used for affecting the circuit connection and sometimes a large current flows through the circuit initially, specially in case of generators. It is therefore, desirable that their output may he allowed to stabilise when they are switched on and only after a lapse of some time the power be connected to other circuits. In Time Delay Relay this is achieved by the arrangement of heater element and bimetallic strip (Fig. 4.18). When the power is initially switched on it flows to heater element only. Due to heat the bimetallic strip becomes hot after some time and the contact *X* is made. This power now reaches the coil and both the contacts, *Y* and Z are made. Now, even when the bimetallic strip becomes cold and contact *X* becomes open, the coil remains magnetized and thus contact *Z* remain closed, and *Y* remain open.

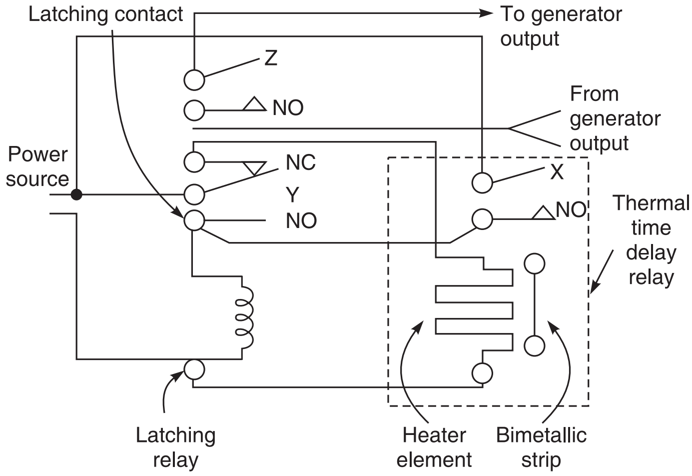


Figure 4.18 Time delay relay.

4.4.2 Reed Relay

The Reed relay consists of two mercury coated electrodes sealed in a glass enclosure filled with an inert gas. The glass enclosure is enclosed in a magnetic coil. The contacts can be activated by a 5 volt signal. The Reed relay is normally used for low current rating and is very common in computer and data handling and switching applications (Fig. 4.19)

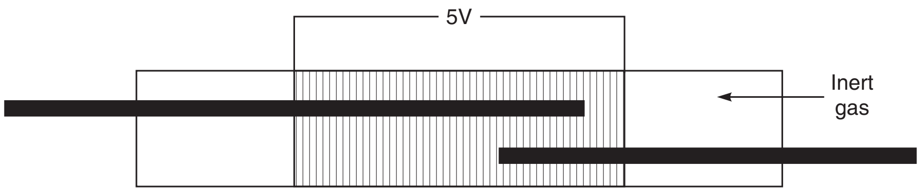


Figure 4.19 Design of Reed relay.

4.4.3 Solenoid

A solenoid converts electrical energy into mechanical energy. The solenoid operation is explained in Fig. 4.20. It consists of magnetic coil and a plunger made of iron which is connected

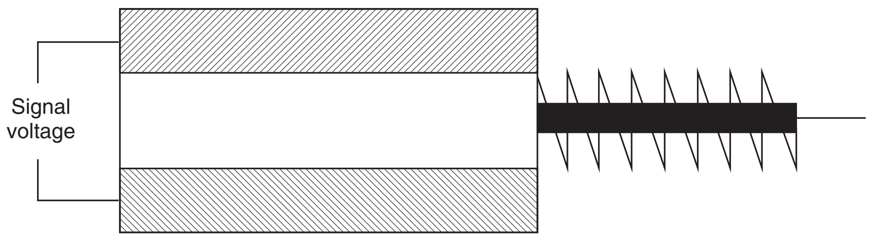


Figure 4.20 Solenoid.

to the coil using spring. When voltage is applied to the coil, magnetic field is generated which brings the plunger inside the coil. The amount of movement of plunger for a particular application will be proportional to the amount of current. However, it will also depend on spring force, type of plunger etc. Solenoids are available for both AC as well as DC voltages.

Following are the different ratings of solenoid coil:

AC	DC
240 V	24 V
	48 V
	110 V
	230 V

As explained in the Fig. 4.21, solenoid can be used to 'pull' a device or 'push' a device.

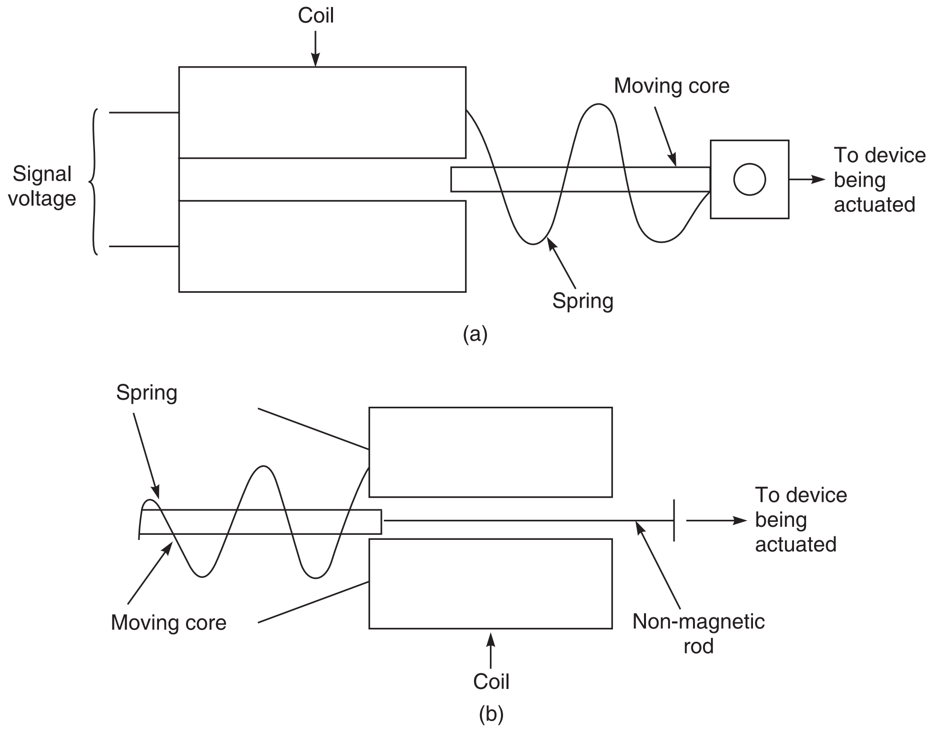


Figure 4.21 (a) Solenoid as puller, and (b) Solenoid as pusher.

As mentioned in Section 4.2.3, the solenoids are used as activators in electropneumatic valves. The signal to the solenoid may come directly from a switch or through a relay. Figure 4.22 shows the operation of a pneumatic cylinder using a solenoid-operated electropneumatic valve.

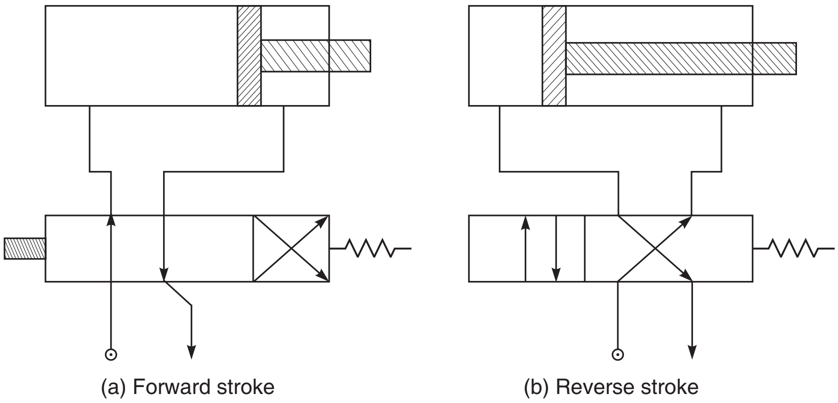


Figure 4.22 Operation of pneumatic cylinder through solenoid-operated electropneumatic valves.

Interfacing with computer

The interfacing of solenoid with microprocessor is shown in Fig. 4.23. When the connected port bit becomes one, the coil is magnetized and the solenoid is energised by contact closure. This circuit may be used for switching on the devices like fan, motor etc.

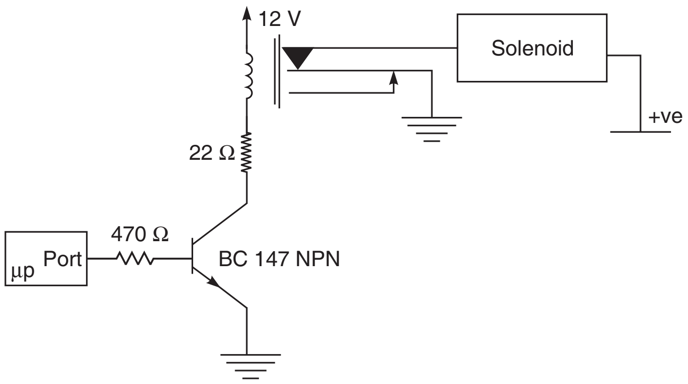


Figure 4.23 Solenoid interfacing with microprocessor.

4.4.4 Thyristor

Also known as Silicon Controlled Rectifiers (SCR), thyristor is basically a three-terminal device like transistor, but it behaves more like a semiconductor diode. The circuit symbol of thyristor, is shown in Fig.4.24. Either **AC** or DC voltage signal may be used as gate signal provided gate voltage is large enough to trigger the thyristor into the 'ON' condition. When triggered the thyristor allows the current to flow between anode and cathode Thus it can be used to control both **AC** or **DC** current in the following manner.

1. *When Input* = +ve and thyristor is triggered, i.e. gate voltage = +ve; Output = Input. If gate voltage becomes less than trigger voltage, then thyristor remains triggered.

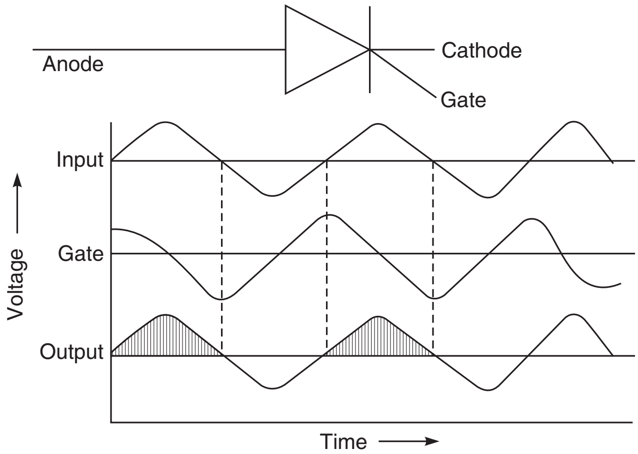


Figure 4.24 Symbol of thyristor and characteristic wave form.

2. *When Input* = 0, thyristor trigger is reset. If gate voltage becomes greater than trigger voltage, thyristor is triggered.
3. *Input* = –ve, Output = 0, irrespective of whether thyristor is triggered or not.

Figure 4.22 shows the relationship between input voltage, gate voltage and output voltage.

4.4.5 Triac

Triac is basically two thyristors joined back-to-back. Unlike thyristor, it conducts in both the directions and is very well suited for controlling **AC** voltage and **AC** powered devices, like **AC** motor. The circuit symbol of triac and wave form is shown in Fig. 4.25.

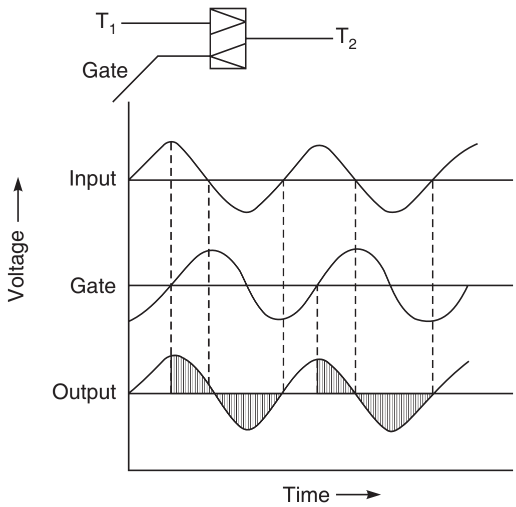


Figure 4.25 Symbol of triac and characteristic wave form.

4.4.6 A Case Study

Let us now try to solve a simple real life problem of an industry. It is a typical material handling problem (Fig. 4.26), in which an object arrives in a conveyer is to he transferred to another conveyer. The solution of this problem through pneumatic cylinders S_1, S_2, S_3, and pneumatic valves have been described in the figure. We shall try to solve this problem using microprocessor or computer. Since S_1, S_2 and S_3 are pneumatic cylinders. We will require three 4/2 pneumatic valves which can be actuated by 24 V. these valves are controlled by computer signals C_1, C_2 and C_3 using reed relays R_1, R_2 and R_3 (Fig. 4.27). The µSW, micro-switch, when 'ON' indicates the arrival of an object in source conveyer.

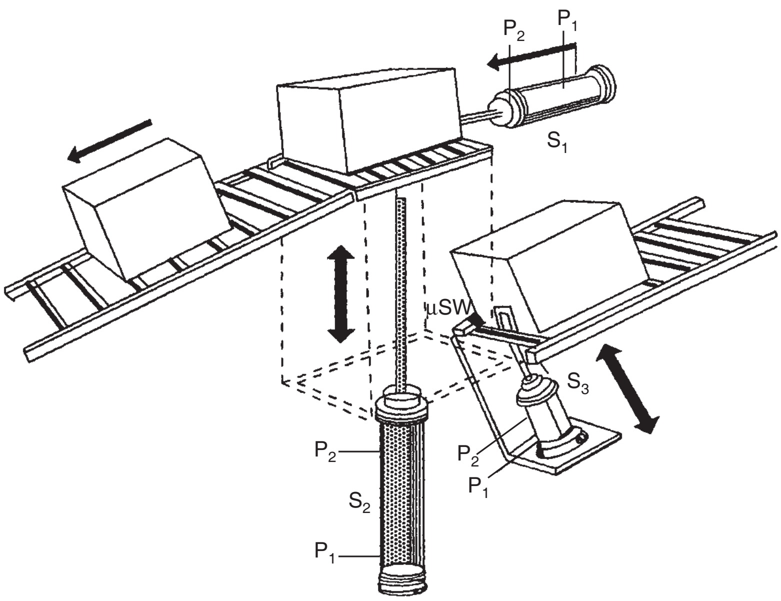


Figure 4.26 Material handling problem.

When C_1 = 1 (5 V) then,

R_1 is actuated, i.e. 24 V is applied to V_1

V_1 is actuated, i.e. pneumatic supply to pneumatic cylinder S_1 port P_1 is switched 'ON'

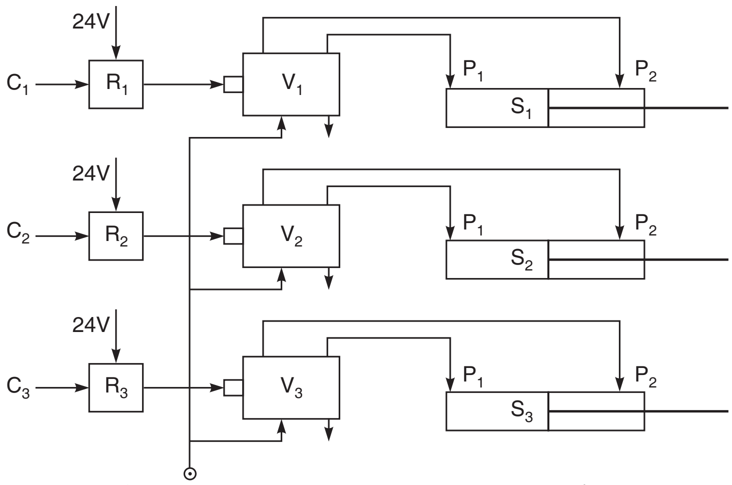
Port P_2 is connected to exhaust.

Let us symbolise this as: $S_1 P_1 = 1$, $S_1 P_2 = 0$.

When $C_1 = 0$ then, R_1 retums to normal position i.e. 24 V to V_1 is disconnected. V_1 returns to normal position, i.e. pneumatic supply to pneumatic cylinder S_1 port P_2 is switched 'ON' and P_1 is connected to exhaust.

This is symbolised as $S_1 P_1 = 0$, $S_1 P_2 = 1$.

The pneumatic sequence algorithm is shown in Fig. 4.28. The algorithm has been translated in terms of signals C_1, C_2, C_3 in Fig. 4.29. Thus with the help of three digital signals, the computer will be able to perform this material handling.



C_1, C_2, C_3 – TTL signals from microprocessor
R_1, R_2, R_3 – Reed relays
V_1, V_2, V_3 – Pneumatic 4/2 valves actuated by 24 V
S_1, S_2, S_3 – Pneumatic cylinders

Figure 4.27 Material handling through microprocessor—Schematic diagram.

INITIAL	–	$S_1\ P_1$	= 0,	$S_1\ P_2$	= 1	
		S_2P_1	= 1,	$S_2\ P_2$	= 0	
		$S_3\ P_1$	= 1,	$S_3\ P_2$	= 0	
IF μSW =		1 THEN				
		$S_2\ P_2$	= 1,	$S_2\ P_1$	= 0	
	–	DELAY	–			
		$S_3\ P_2$	= 1,	$S_3\ P_1$	= 0	
	–	DELAY	–			
		$S_3\ P_2$	= 0,	$S_3\ P_1$	= 1	
		$S_2\ P_2$	= 0,	$S_2\ P_1$	= 1	
	–	DELAY	–			
		$S_1\ P_2$	= 0,	$S_1\ P_1$	= 1	
		DELAY				
		S_1P_2	= 1,	$S_1\ P_1$	= 0	
		DELAY				
		REPEAT				

Fig. 4.28 Pneumatic sequence algorithm.

```
INITIAL   –  C1      = 0
             C2      = 1
             C3      = 1
IF µSW =     1 THEN
             C2      = 0
          –  DELAY   –
             C3      = 0
          –  DELAY   –
             C3      = 1
             C2      = 1
          –  DELAY   –
             C1      = 1
          –  DELAY   –
             C1      = 0
             DELAY
             REPEAT
```

Figure 4.29 Digital sequence algorithm.

However, if S_1, S_2, S_3 are solenoids, then the operation will become more simplified since V_1, V_2,V_3 will not be required as computer signals C_1, C_2 C_3 will be able to operate solenoids through reed relay R_1, R_2 and R_3.

4.5 MOTOR ACTUATORS

The subject of electric motor is quite extensive. However, in this chapter it has been taken up only from the point of view of concept and application, all the unnecessary details have been left out. Motorized valves are very common in industries and are replacing the traditional pneumatically actuated valves. Motors are used to give the rotary motion to the valve wherever required.

4.5.1 DC Motor

The DC motor operates on simple principle of attraction between unlike magnetic poles and repulsion between like magnetic poles. The principle of DC motor is explained in Fig. 4.30. An armature containing one wire loop is put between two electric magnets. When DC current is applied to the armature coil, it becomes magnetic and its poles tend to occupy the position nearer to the opposite poles of electric magnet. Thus the armature will rotate by 180 degrees. Using the commutator and brushes, the direction of current in armature is reversed when it rotates by 180 degrees, and thus the continuous rotation of armature takes place. In practice, motors have many sets-of loops wound around armature pole. The DC motor is symbolized through field winding and armature symbol. There are three types of DC motors depending on the position of field winding with respect to armature in the circuit (Fig. 4.31).

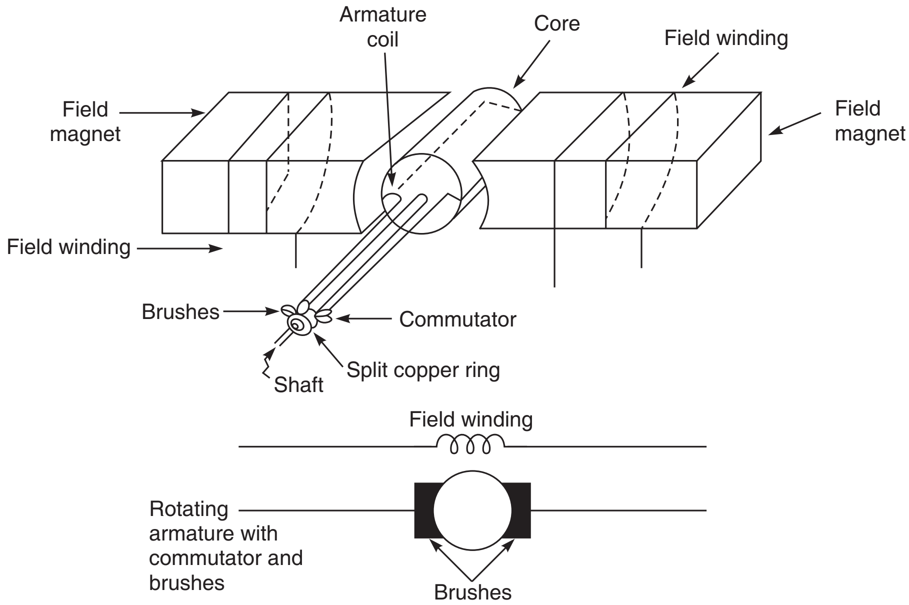


Figure 4.30 DC motor operation.

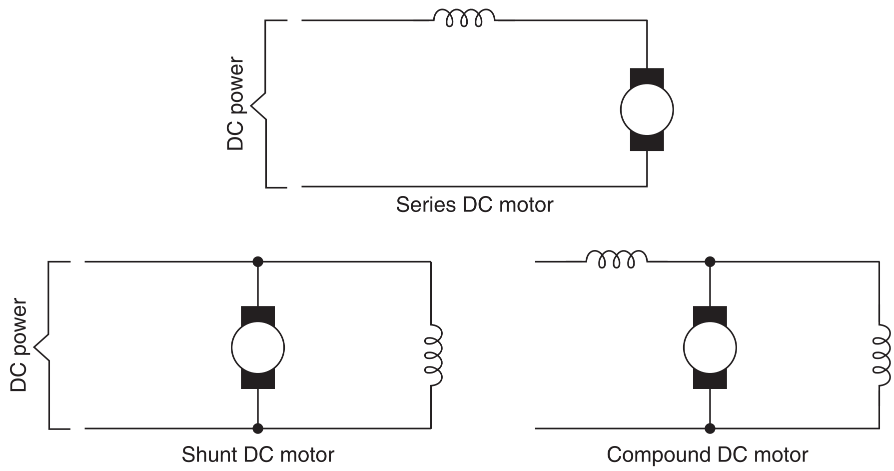


Figure 4.31 DC motor types.

Series DC motor

When the field winding is in series with armature winding and power-line, the motor is termed as series DC motor. The characteristic of these motors is large initial torque and are therefore used in control applications where inertia of heavy load must be overcome. Such applications include electric loop, crane, elevator, etc.

The speed of these motors may be controlled using variable resistance in series. These variable resistors can be rheostat or variable resistance network (Fig. 4.32(a)).

Shunt motor

The shunt motor has field winding in parallel or shunt with armature winding. The characteristic of these motors is the constant speed and these are used in the areas where constant speed is important. The speed control is through variable resistance in series with armature or field winding (Fig. 4.32(b)).

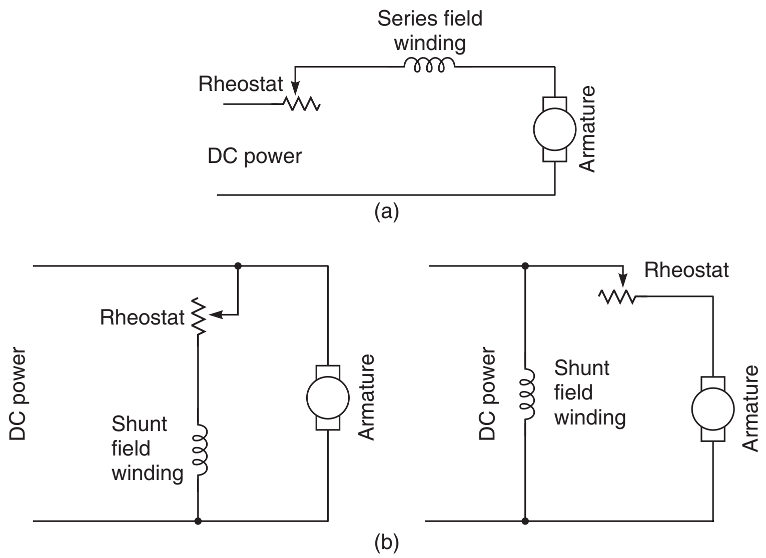


Figure 4.32 Speed control circuit for DC motors: (a) Series, and (b) shunt.

Compound motor

Basically compound motors are combination of series as well as shunt motors with two windings one in series and other as shunt. The series winding has few turns of heavy wire while shunt winding has many turns of fine wire. These motors possess characteristics of both series and shunt motors and have large torque as well as constant speed in load variation. The speed control is achieved through variable resistance or rheostat in armature circuit, field winding or both.

The direction of movement of DC motor can be reversed by reversing either armature or field winding polarity. However, if polarity of both armature and field winding are reversed then the motor will be rotating in the same direction.

4.5.2 AC Motor

The AC motors operate by the same principle as DC motor. In AC motors, field windings are called stator and armature is called rotor. There are three basic types of AC motors,

1. Induction motor
2. Synchronous motor
3. Universal motor.

AC motors can he operated by single-phase AC power or poly-phase (two-phase or three-phase) power.

Induction motor

The rotor of induction motor basically comprises of copper bars embedded in insulated drum and connected at their ends to form windings. The stator has number of sets of poles to which AC power is fed. The working of induction motor with two sets of poles *A* and *B* has been explained in Fig. 4.33. Current wave form *A* and current wave form *B* has phase difference of 90 degree between each other. At (1) current *A* is maximum and current *B* is zero and thus magnetic field is created horizontally between stator poles *A*. The rotor will also move horizontally. Whereas at (2), *A* drops to zero and *B* becomes maximum, the magnetic field is created vertically and rotor also moves by 90 degree. Thus with the movement of current wave

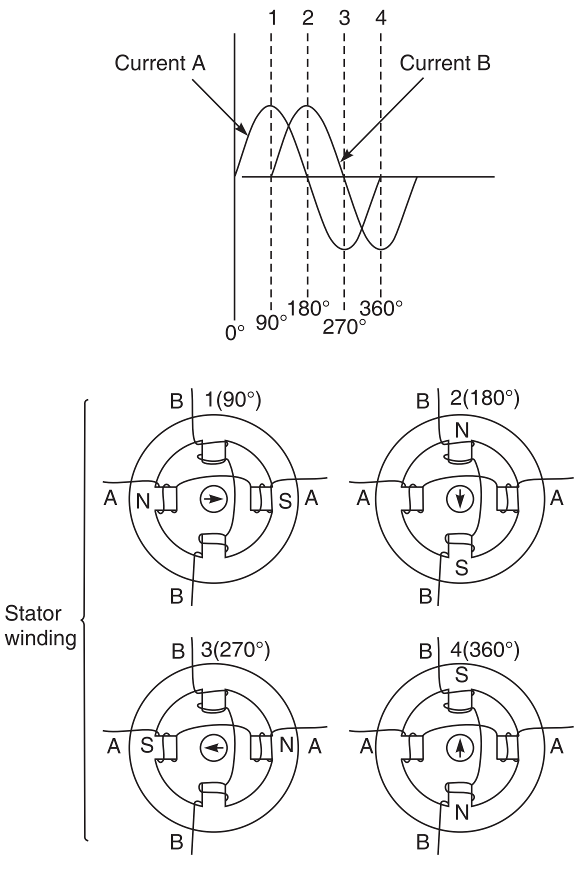


Figure 4.33 Induction motor operation.

form the rotor is also moving at the same speed. However, when load is attached, the rotor speed is decreased from the synchronous speed. The difference between the synchronous speed and full load speed is called slip. If the induction motor is to be operated from single-phase power then the power wave form has to be splitted into two with phase difference at 90 degree. This can be achieved by using capacitor or inductor in the circuit.

Synchronous motor

In synchronous motor the stator is same as in an induction motor, but the rotor does not depend on induced current from stator for magnetic field. The rotor contains winding which is excited by a DC source by means of slip rings and brushes. In fact, synchronous motor behaves like induction motor in the beginning but when rotor speed becomes maximum then DC current is applied to rotor winding. The magnetic poles created by DC current get locked with revolving field of stator and rotor rotates with this field. The characteristic of synchronous motor is zero slip, i.e., constant speed like clocks, counters, etc.

Universal motor

Universal motors are basically series DC motors which can work with AC current as well. They are used in light duty equipments like fans, vacuum cleaners, etc.

4.5.3 Interfacing of AC/DC Motor and Speed Control

Figure 4.34 shows the speed control of AC/DC motor in closed loop fashion. The AC power is connected to thyristor network which rectifies AC power to DC. The filter network produces a filtered DC voltage which is fed to DC motor. The speed is sensed by a tachogenerator and is fed to microprocessor which calculates the firing angle of the thyristor based on the speed required. The microprocessor then sends the gate control signal to the thyristor to effect the control. This circuit can he used for AC motor as well as DC motor control. However, in case of AC motor filter network is not required.

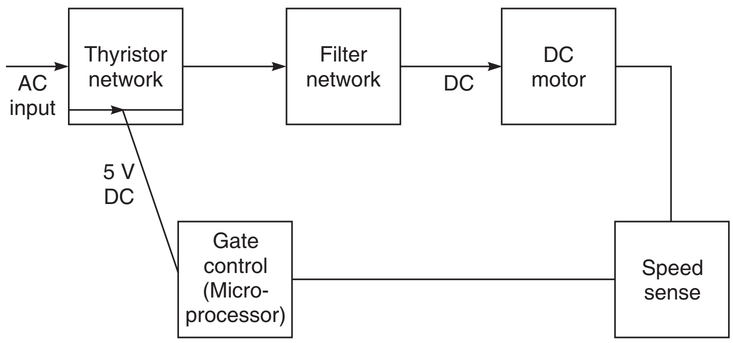


Figure 4.34 Speed control—ACDC motor.

Figure 4.35 shows the speed control of universal motor through voltage across resistance R_1 and R_2. This can be achieved through digital to analog converter to which R_2 can be connected as load. This way microprocessor can be interfaced to control the universal motor.

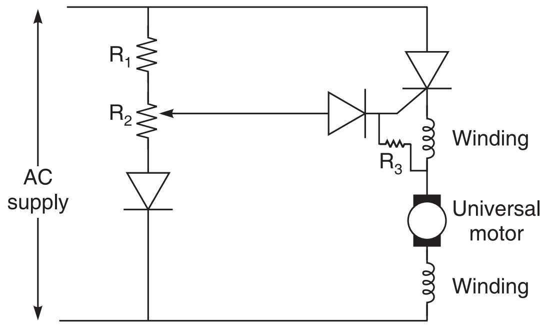


Figure 4.35 Speed control of universal motor.

AC motor control through gear system

Generally in order to reduce the speed of an AC motor output gear system is used. When the speed reduces, the torque increases and this helps in the movement of heavy loads. Normally, two types of gears, viz. worm gear and spur gear are common in the industry. These are shown in Figs. 4.36 and 4.37 respectively. Worm gear is self-locking thus load cannot move downward by back driving the motor. In case of spur gear, load can back drive the system but frictions are normally provided for protection. Worm gear uses more power than spur gear and is approximately one-half as efficient as the spur gear. However, speed reduction from 1800 rpm to 1 rpm is possible through worm gear only.

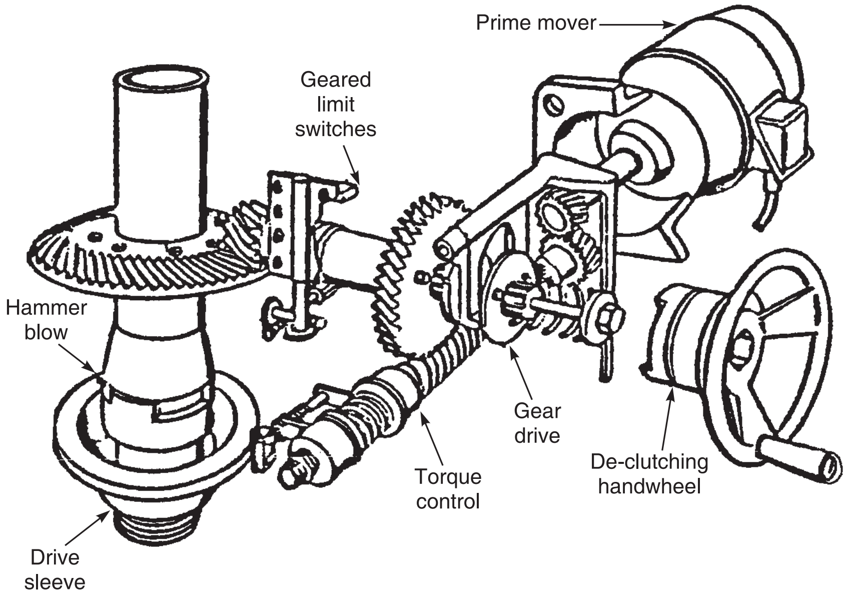


Figure 4.36 Worm gear reduction.

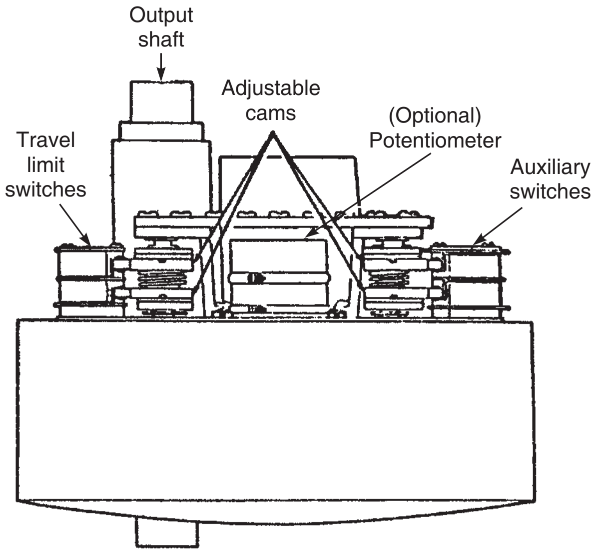


Figure 4.37 Spur gear reduction.

4.5.4 Interfacing Motor to Actuated Device

The interfacing of motor to actuated device is done by using clutch and brake system. Clutch is used to quickly engage or disengage the motor from device being actuated and consists of a friction disk. When actuated, it comes into contact with the disk in motor and rotates due to friction. Actuation is through solenoid (Fig. 4.38). In brake system, housing is connected/disconnected to motor on receipt of brake or brake release signal. The actuation of brake is through solenoid (Fig. 4.39)

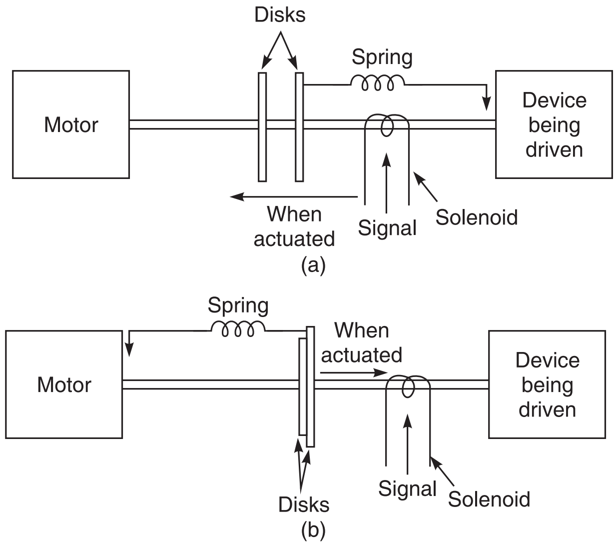


Figure 4.38 Operation of friction disk clutches.

It is possible to have two configurations for interfacing motor to actuated device. In the first configuration, the actuator signal is used to engage the device with motor through clutch/break. This configuration is shown in Figs. 4.38(a) and 4.39(a). In the other configuration, the actuator signal is used to disengage the device with motor through clutch/break. This configuration is shown in Figs. 4.38(b) and 4.39(b).

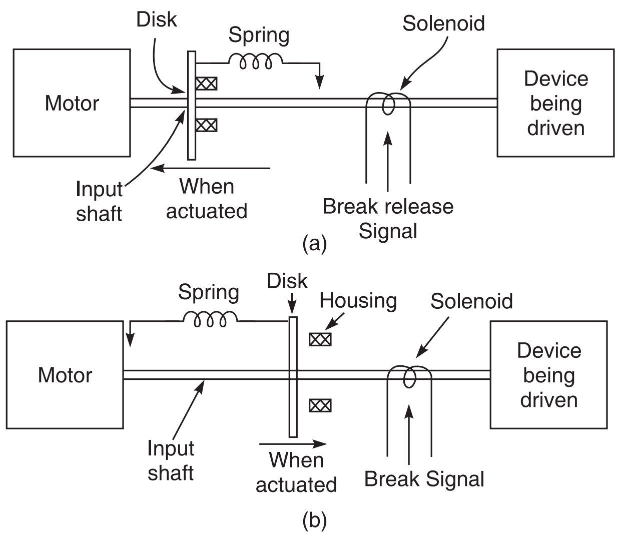


Figure 4.39 Operation of solenoid actuated brakes.

4.5.5 Stepper Motor

Stepper motor, as the name indicates, moves in steps. When the stepper motor drive circuitry receives a step pulse, it drives the motor through a precise angle (step) and then stops until the next pulse is received. Consequently, provided that the maximum permissible load is not exceeded, the total angular displacement of the shaft is equal to the step angle multiplied by the number of step pulses received. This relation is further simplified as the shaft position is directly proportional to the number of step pulses supplied, since the step angle for any particular motor is fixed.

The step angle for the stepper motor ranges from 0.45 degree to 90 degrees, the most common angle being 1.8 degree, i.e. 200 steps per revolutions. The positional error at each step is typically ±5%. The important point to note is that this error is not cumulative, i.e. irrespective of the number of pulses supplied, the final position accuracy is always within ±5% of one step.

Another major advantage of stepper motors over alternative drive system is that stepper motor type positioning control loop can be open. That means that there is no need for displacement transducer or complicated feedback (closed loop) control systems. A simple counter can count the number of pulses which are being sent to the motor drive electronics. This may be done by an independent counter or by the microprocessor. Furthermore, the rate of pulses to the motor can also be controlled, and therefore, velocity or the acceleration of the linear/rotation displacement can also be controlled without any feedback at all.

There are three types of stepper motors:

1. Variable reluctance
2. Permanent magnet
3. Hybrid motor

The basic four-phase variable reluctance motor is shown in Fig. 4.40

Variable reluctance type of stepper motor works on the reaction between an electromagnetic field and a soft-iron rotor. The schematic diagram in Fig. 4.40 shows an 8-pole stator and 6-pole rotor stepper motor. Each stator pole has a winding and these windings are grouped into the 4 sets of 2 poles each. Out of 4 sets of poles, one set of poles is energised at a time. As shown in Fig. 4.40, two rotor poles are aligned to the energised set of poles of the stator while the other four rotor poles are midway between the remaining stator poles (15 degrees misalignment with the stator poles B, C and D). Now if pole B or C or D is energised, a 15-degree clockwise or counterclockwise rotation takes place because the rotor seeks to close the shortest flux path. The variable reluctance type of stepper motor can achieve high stepping rates and higher torque.

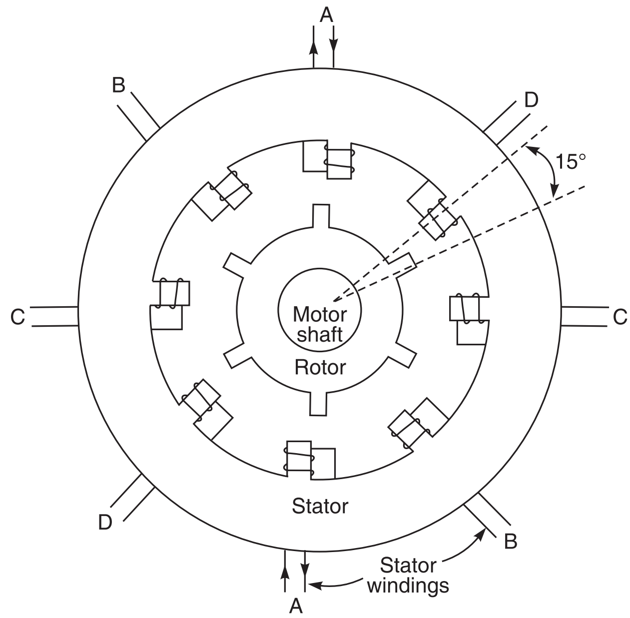


Figure 4.40 Four-phase variable reluctance stepper motor.

The multiple stack variable reluctance stepper motor is improvement over single stack design. In multiple stack format a separate stator/rotor assembly is provided for each phase. However rotors are still mounted on common shaft. Such motors may be controlled more precisely than normal variable reluctance stepper motors.

Permanent magnet type of stepper motor has stator or rotor as permanent magnet which causes poles and teeth to align and effect rotation, in presence of electrically produced flux.

Hybrid motors have structural characteristics of both permanent magnet and multiple stack variable reluctance stepper motors. These motors have multiple stator/rotor assemblies but contain permanent magnet mounted axially between the rotor assemblies.

The stepper motors differ from each other in a number of aspects. These are, (a) number of phases, (b) sequence of energisation of phases, and (c) the DC current required per phase. The stepper motor manufacturer will therefore, specify in the data sheets of the motor about number of phases, phase sequence to be applied for a particular direction (clockwise or anti-clockwise) and maximum voltage current required per phase.

For accurate linear positioning, a lead screw and nut may be connected to the stepper motor. The displacement will be proportional to the number of pulses supplied:

$$\text{Linear displacement} = \frac{\text{No. of pulses (step)} \times \text{Pitch of lead screw}}{\text{Steps per revolution}}$$

Now, if lead screw has a pitch of 5 mm/revolution and step angle of 1.8 degree (200 steps/revolution), then each step produces linear travel of 0.025 mm. Now, such an arrangement will find immediate application in automatic latches, milling and drilling machines and positioning mechanism of all sorts.

The interfacing of stepper motor requires a circuit which can generate the step pulses at the desired rate and the direction signal; power amplifier to amplify these low voltage/power signals to the power required by the motor phases. This can be done very easily by microprocessor by using just an output port and simple software, and power amplifier circuit.

Let us assume that we have a stepper motor with following characteristics:

1. Four-phase motor; one phase to be activated at a time.
2. Clockwise direction if, phase activation is in the sequence — A, B, C, D, A and anticlockwise if the phase activation sequence is — D. C, B, A, D.
3. Voltage at common point is 24 volts, current per phase is Im and Step angle is 1.8 degrees.

The block diagram of the interface with stepper motor is shown in the Fig. 4.41.

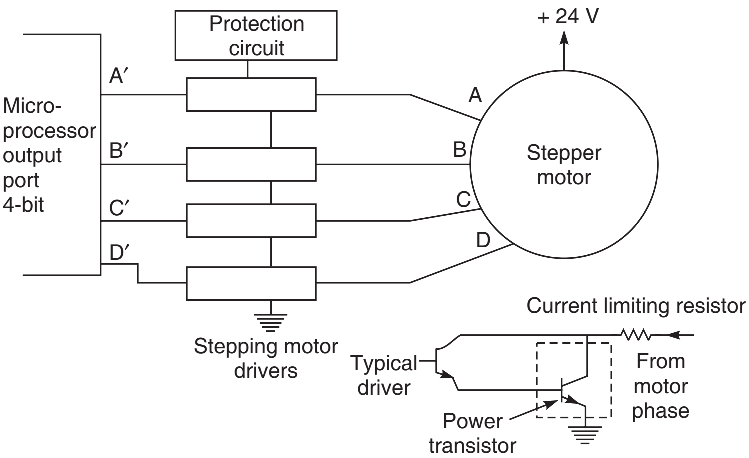


Figure 4.41 Microprocessor interface with stepper motor.

Stepper motor driver circuit will have to be designed specially for a particular motor. A driving transistor (or transistor pair/pairs) capable of providing sufficient gain and desired Source/sink current at desired voltage should be selected. It becomes necessary, particularly when the motor of high current type needs to have some protection circuit, which is meant to safeguard the motor and to check that only desired numbers of phases are 'ON' and the current passing through the stepper motor coils is not dangerously high (otherwise the motor may burn). This circuit may give a feed-back to microprocessor.

Now, the remaining tasks will be done by the microprocessor. It will generate the phase activation signals (on A; B; C; and D; lines) on the output port and will wait for the required phase activation time and then depending upon direction of rotation, it will activate corresponding phases. The microprocessor can easily wait for any amount of time starting from few microseconds to even hours. So, it is very easy to control the speed of the motor. Signals required to move the motor by 6 steps in clockwise direction are shown in Fig. 4.42.

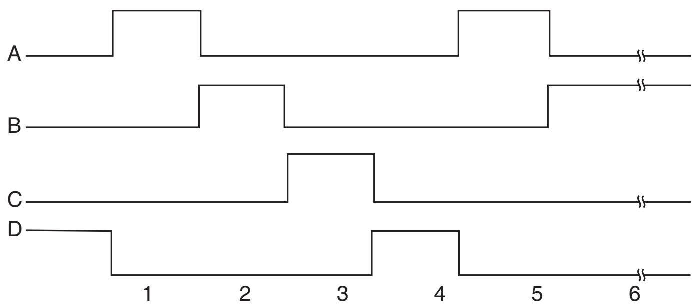


Figure 4.42 Signals for stepper motor movement.

A stepper motor with 1.8 degrees step angle, 4-phase, permanent magnet type has been interfaced to the microprocessor as shown in Fig. 4.43. A 4-bit latch (IC 7475) is connected to the databus of the microprocessor and its output is amplified through transistor 2N 3055. This motor can move from low speed to up to 30,000 steps per second, i.e. 150 rpm. It is also possible to move the motor by one-half step. Figure 4.44 shows the phase activation sequence for full step (1.8 degrees).

4.6 CONTROL VALVES

There are two types of valves prevalent in the industry:

1. Shut-off valve
2. Throttling valve

4.6.1 Control Valve Characteristics

The flow characteristics of control valve define the related flow variation with respect to percentage valve opening, for a constant pressure drop across the valve body. Following types of characteristics are common in control valve.

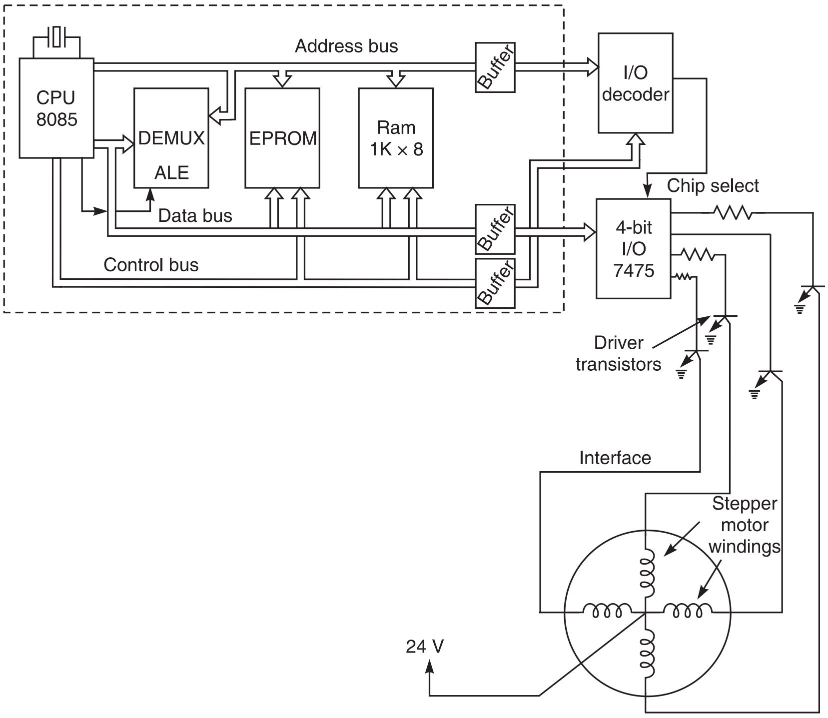
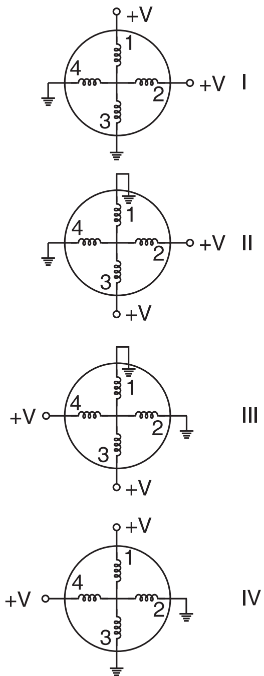


Figure 4.43 Stepper motor control by intel 8085 microcomputer—Block diagram.



Colour	B	B	B	B	Value
Output	$\overline{Q_1}$	Q_1	Q_0	$\overline{Q_0}$	
Winding No.	2	4	3	1	
I	1	0	0	1	09
II	1	0	1	0	0A
III	0	1	1	0	06
IV	0	1	0	1	05
I	1	0	0	1	09

Figure 4.44 Phase activation sequence of stepper motor.

Linear characteristic

The flow is directly proportional to the valve opening for a constant pressure drop (Fig. 4.45). The relationship thus can be represented as a straight line.

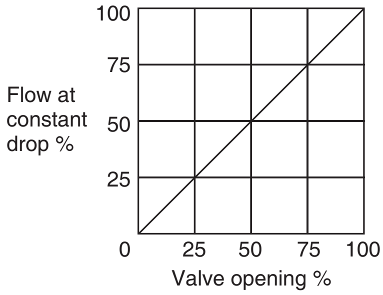


Figure 4.45 Linear characteristic.

Equal percentage characteristic

According to the property of equal percentage characteristic (Fig. 4.46) equal opening of valve must produce equal percentage change in the flow at constant pressure drop, based on the flow just before the change is made.

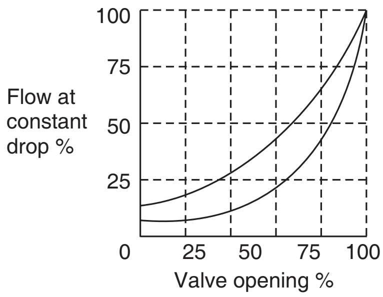


Figure 4.46 Equal percentage characteristic.

Quick opening characteristic

In this type of characteristic (Fig. 4.47) the relationship between flow and valve opening is

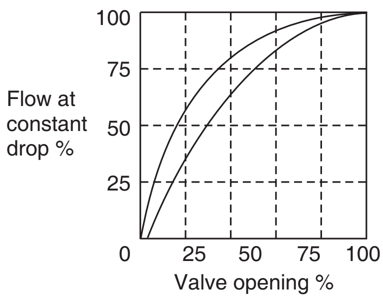


Figure 4.47 Quick opening characteristic.

approximately linear up to 60% to 70% of valve opening. After this limit the flow does not change rapidly with the change in valve opening.

4.6.2 Control Valve Categories

The control valves may be grouped into following three major categories (Fig 4.48) according to their characteristics.

1. Sliding stem type
2. Rotary stem type
3. Miscellaneous type.

Each of these categories is further divided into number of sub-categories depending on the body material, pressure ratings, flow capacity, relative shut-off capability, end connection and size.

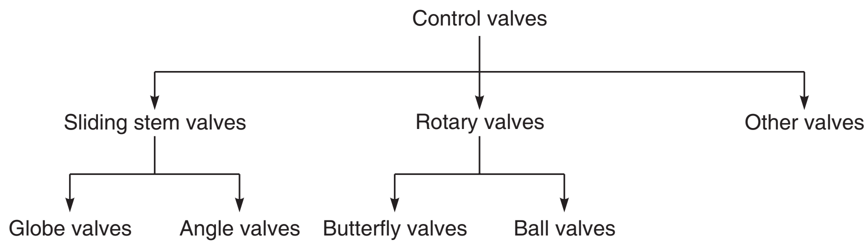


Figure 4.48 Major categories of control valves.

Sliding stem valves

Sliding stem valves are most commonly used in industries and are popularly known as globe type of valves, considering the shape of the valve body. These valves are available in a number of types and variations, depending on the shut off capability requirement, pressure drop requirement, temperature requirement, and type of fluid. The ANSI B.16-104-1976 (FCI 70.2) classification of control valves with respect to maximum leakage and maximum seat leakage with respect to port diameter is shown in Fig. 4.49. Since these type of valves are in use for a long time, a number of modifications have been incorporated in the design, body material etc. The double-seated valves (Fig. 4.50) which were popular once, have more or less been replaced by compact single-seated type.

The single-seated valves are widely used valves in sliding stem category due to following advantages:

— Availability of wide variety of configurations
— Good flow shut off capability
— Reduced plug mass which leads to less vibrations
— Easy maintenance.

Class	Maximum leakage
I	No test required
II	0.5% of rated valve capacity
III	0.1% of rated valve capacity
IV	0.01% of rated valve capacity
V	5 × 10 ‘ml/minute of water per inch of orifice diameter per psi differential
VI	ml per minute of air or nitrogen versus port diameter

Nominal port diameter		Maximum seat
In.	mm	leakage ml/minute
1	25	0.15
1.5	38	0.30
2	50	0.45
3	75	0.90
4	100	1.70
6	150	4.00
8	200	6.75

Figure 4.49 Valve seat leakage classification (ANSI B16–104–1976).

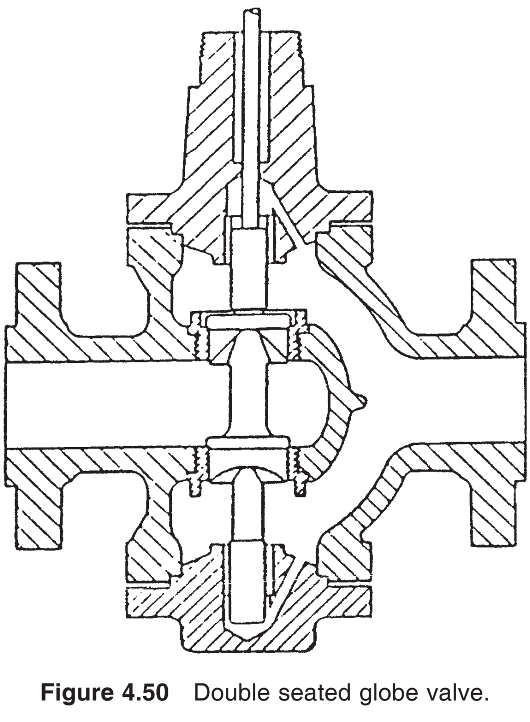


Figure 4.50 Double seated globe valve.

Single-seated valves may be stem guided (Fig. 4.51), top guided (Fig. 4.52) or top and bottom guided (Fig. 4.53). T he top and bottom guided valves are no more popular considering the disadvantages of having more plug mass and more probability of leakage than other two types of valves.

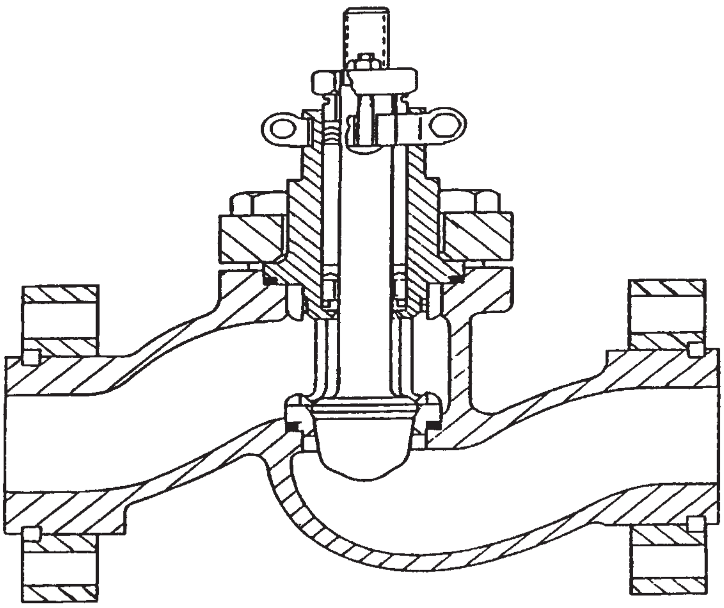


Figure 4.51 Stem guided single-seated valve.

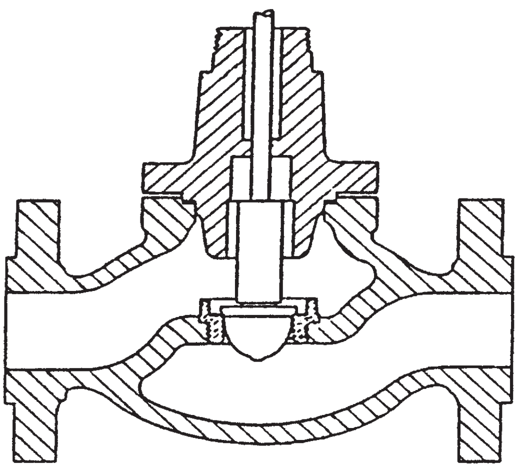


Figure 4.52 Top guided single-seated globe valve.

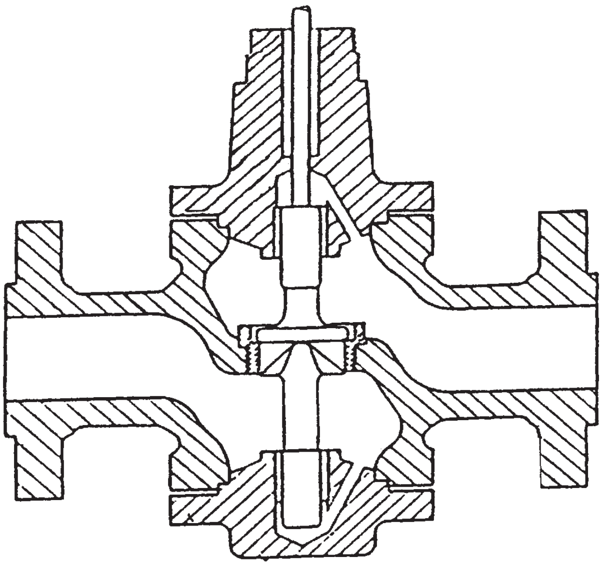


Figure 4.53 Top and bottom guided single-seated globe valve.

However out of stem guided and top guided valves, the stem guided valves are considered better when fluid contains solid mass or fluid is highly corrosive type.

The cage valve is basically a single-seated valve and is designed to suit the requirement of process industries. Cage valves can be of two types:

(i) *Stem or top guided type.* Valves which use the cage for clamping the seat ring into valve body (Fig. 4.54).

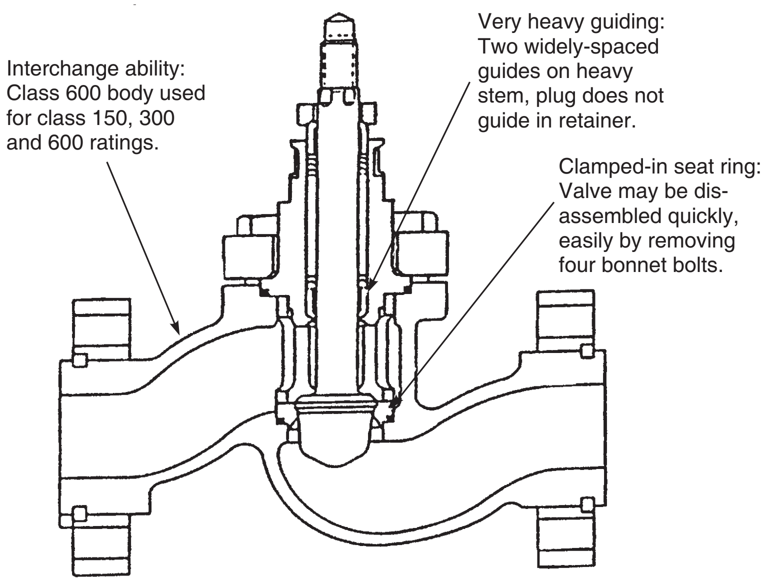


Figure 4.54 Top guided cage valve [*Courtesy*—Valtek Inc.].

(ii) *Cage guided type.* Valves which use the cage for guiding as well as clamping the seat ring into valve body (Fig. 4.55).

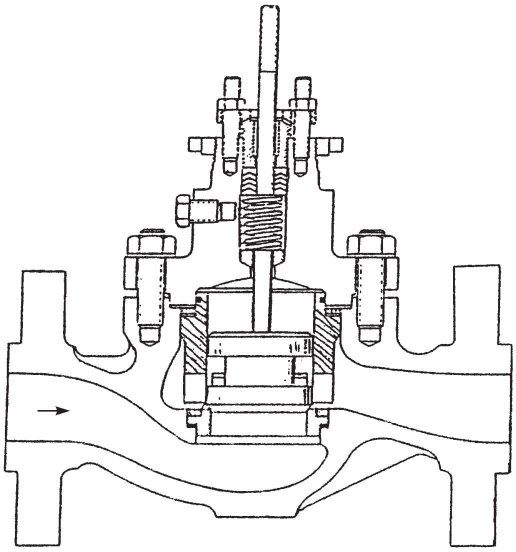


Figure 4.55 Cage guided cage valve [*Courtesy*—Fisher Control].

The top entry bonnet and trim design makes it very easy to carry maintenance work and change trim in cage valves. There are a number of variants of cage valves.

Like cage valves, split body valves are another variant of single-seated valves (Fig. 4.56). The design was made to handle slurries, gummy fluids, and corrosive fluids. The seat ring is clamped between the body halves and body can be easily disassembled for maintenance. This

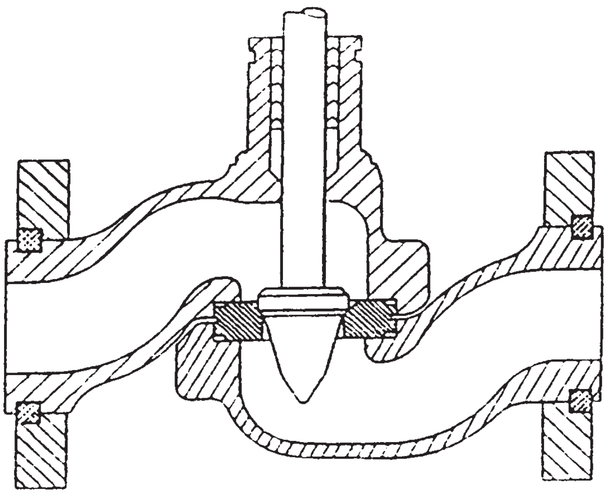


Figure 4.56 Spilt body valve [*Courtesy*—Masoncilan International].

design is loosing the ground because of limitation on installing special trim modification as compared to other stem guided valves discussed earlier.

There are many other variants of single-seated valves. These are angle valves, Y-style valves, three-way valves, jacketed valves, etc. We shall not discuss all these here, as they have very specialised applications. The reader may, however, refer to available literature on these.

Rotary stem valves

The rotary stem valves include buttefly valves, ball valves and a number of special designs. These can handle a variety of applications except those involving very small flow or very high pressure drop. It is estimated that 20% of control valves used are rotary stem valve type.

(i) *Butterfly valve.* Butterfly valves are one of the oldest types of valves used and its variations are common in consumer products like furnace dampers, automobile carburetors, shower heads etc. Modern butterfly valves are suitable for variety of applications including high pressure drops, tight shut-off and for fluids with corrosive characteristics.

The mechanical design of butterfly valve is shown in Fig. 4.57. The major elements of a butterfly valve are valve body, disc and shaft. The disc is fastened with the shaft and can be rotated around it by rotating the shaft. A lever handle or a hand wheel with rotary gear box can be used to rotate the shaft manually. However automatic operation can be achieved by attaching electrical, pneumatic or hydraulic motor drive to the shaft. The 90° rotation of shaft will convert the valve from fully open position to fully closed position (Fig. 4.58). When fully open, the disc is parallel to the flow and when fully closed, the disc is perpendicular to the flow.

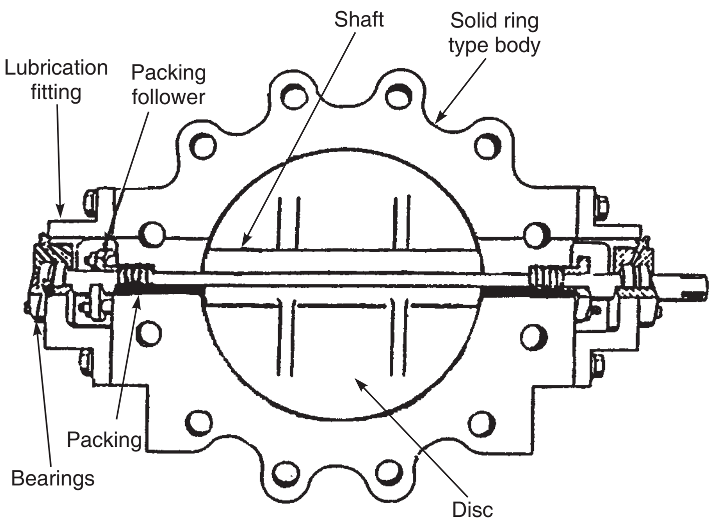


Figure 4.57 Butterfly valve [*Courtesy*—Fisher Control].

Following are the categories of butterfly valves based on construction and capability:

— Swing-through butterfly valves
— Tight shut-off butterfly valves.

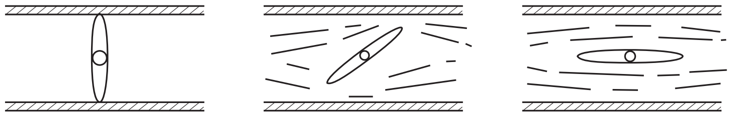


Figure 4.58 Butterfly valve vane positions.

Swing-through valves have symmetrical disc and shaft design with some clearance between disc and body. The solid ring type body is mounted between pipe flanges. The thickness of shaft is a function of maximum pressure drop and torque required. Special disc shapes like cambered, fish tail etc. are available, to reduce the torque and increase throttling capacity. These types of butterfly valves are normally limited to rotation of 70 degree open position, and are suitable for heavy patterns with larger diameter shaft. This is due to the reason that disc profile projection tends to disappear into the shaft area as the valve opens.

Tight shut-off butterfly valves may be one of the following two types.

— Elastomer/plastic liner valve

— High performance butterfly valve (HPBV)

Elastomer or plastic liner butterfly valve may also have encapsulated disc. Various elastomer materials are possible. Normally, materials with more rigid backup ring which completely line the bore of the valve and valve gasket face area are used. For corrosive fluids, liners of plastic such as teflon are available. In some cases, disc is also encapsulated in the elastomers or Teflon (Fig. 4.59). Since elastomers are attacked by the process fluid. these may be softened resulting into swelling or cracking. Plastic liners also have similar disadvantages. Thus, elastomer or plastic liner must be properly selected for trouble-free valve operation

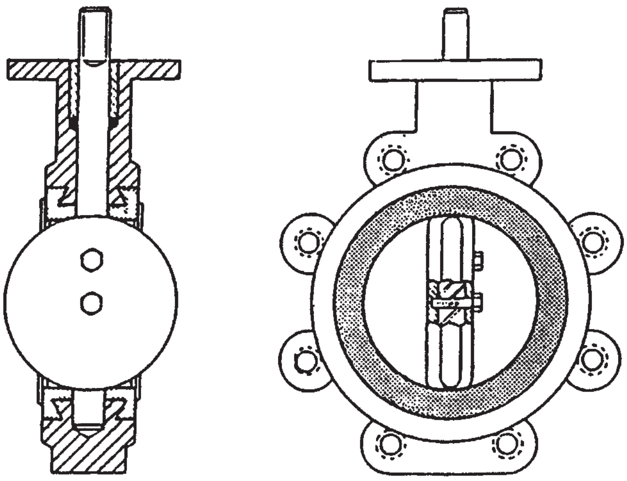


Figure 4.59 Butterfly valve with encapsulated disc [*Courtesy*—Keystone Valve].

High Performance Butterfly Valves (HPBVs) are the most significant design advancement. The notable features of these valves are (a) tight shut-off, (b) reduced operating torque, (c) excellent throttling capability, (d) ability to operate with relatively high pressure drops and (e) available in compact size, reduced weight and lower cost. These features have made HPBV superior to previously described, butterfly valves. The main characteristics of HPBV design are separable seat ring contained in the body and the eccentric cammed disc (Fig. 4.60). The camming action enables the disc to back-out of and into the seat before and after the disc rotation. This is accomplished by having shaft offset from both the outlines of the disc and valve body. Normally, plastic material like teflon or various elastomers are used to design the seats. The seat design varies with manufacturer and may vary from very simple to complex.

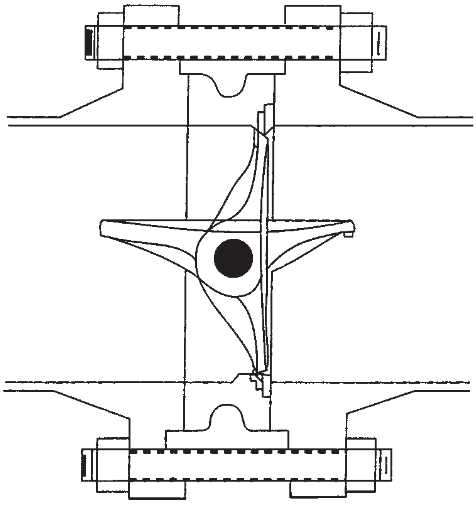


Figure 4.60 High performance butterfly valve with cammed disc [*Courtesy*—Valtek Inc].

A recent design of butterfly valves is *floated disc* design. This is appropriate for compressible fluid application, as it is capable of significant noise reduction over other types of valves.

(ii) *Ball valves*. This sub-category of rotating stem valves consists of a spherical plug and a valve body. The rotation of spherical plug in the valve body controls the flow. Three types of ball valves are available:

— Conventional ball valve
— Characterised ball valve
— Cage ball valve.

Conventional ball valves are quarter turn type, i.e. 90 degree rotation of ball may change the valve position from fully open to fully close or vice versa. This type of ball valve can be used for precise control of flow through valve body or for tight shut-off. This conventional ball valve can thus play dual role of precise throttling as well as tight shut-off valve. Figure 4.61 shows the design of 'top entry pierced ball valve' which is a type of conventional ball valve.

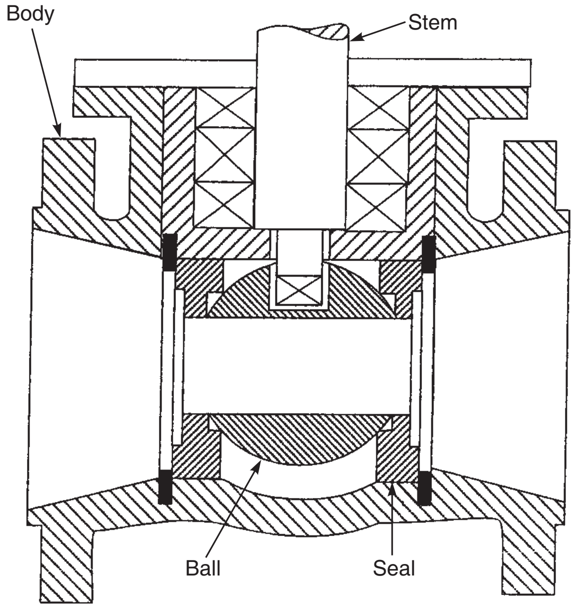


Figure 4.61 Top entry pierced ball valve.

The rotation of stem by 90 degrees will fully close or fully open the valve. The ball is pierced as the name indicates and thus in one position it fully connects the input and output ports of the valve, whereas when turned by 90 degrees it blocks the flow completely.

The conventional ball valves may he actuated manually (through a handle), pneumatically, electrically, hydraulically or by using combination of these. The motor actuators have already been discussed in the previous section. The valves are designed such that they have minimum friction between body and ball and require minimum torque to turn the ball. The flow characteristics of conventional ball valve are approximately *equal percentage valve characteristics.*

The construction of *characterised ball valves* is shown in Fig. 4.62. The ball of conventional ball valve is replaced by a V-notched ball or a parabolic ball. In characterised ball

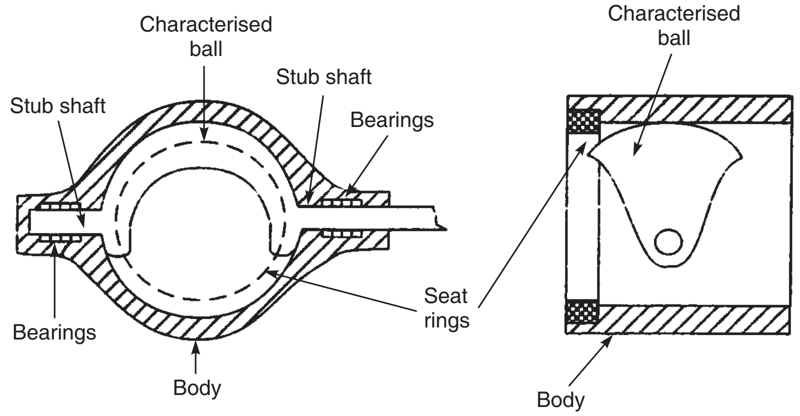


Figure 4.62 Design of characterised ball valve.

valve, the ball has been modified so that only a portion of it is used. The edge of partial ball can be modified to obtain desired valve characteristics. These type of valves have a number of design problems unlike conventional ball valves. These are not popular for high pressure applications. The quarter turn movement of ball will fully close or fully open the flow as shown in Fig. 4.63.

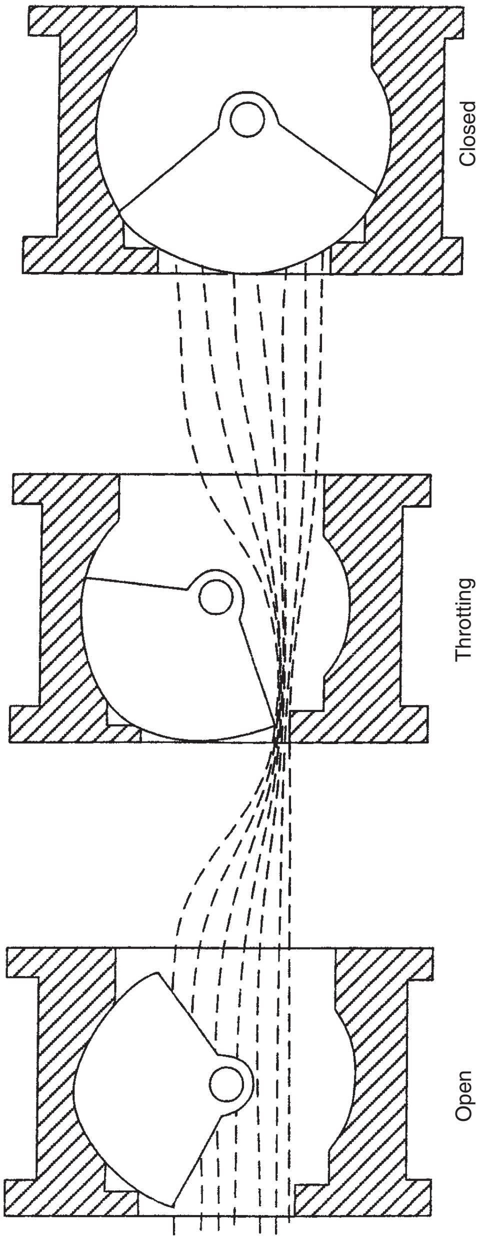


Figure 4.63 Position of characterised ball valve.

The conceptual design of *cage ball valve* is shown in Fig. 4.64. In this type, the ball is positioned by a cage in order to open, throttle or shut off the flow. The valve consists of a venturi-ported body, seat rings and a cage and ball combination to effect the control. The ball is lifted by the stem and the cage rolls the ball out of the seat for fully open or throttling position. In case of throttling, the movement of cage will depend on the desired flow. The cage valves are available from 1/4" to 14" sizes with pressure range of 150 psig to 2,500 psig. The range span of cage ball valve is very high, depending only upon the ability of the actuator to position the cage.

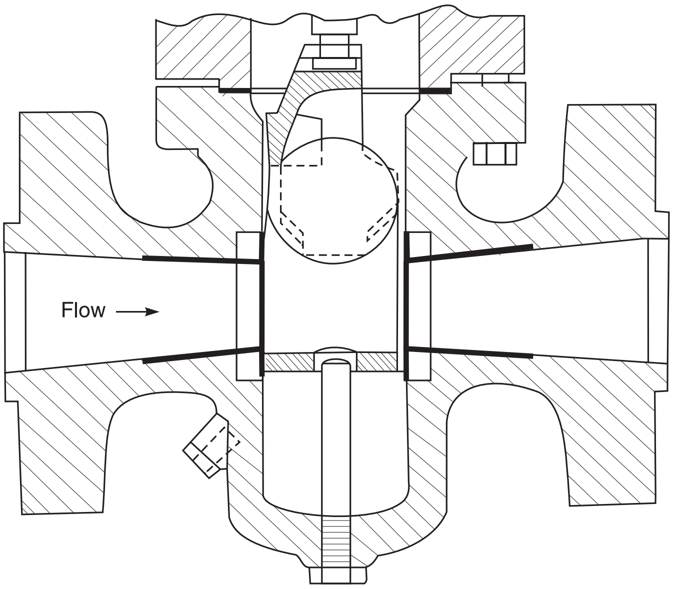


Figure 4.64 Cage ball valve.

For emergency closure operations, a variation of the cage ball valve is often applied. The valve contains springs and piston to set the high and low pressure limits of the valve. When high limit is exceeded then piston is pushed down to eject the ball from holder to the seat. When pressure becomes lesser than the low pressure limit then low pressure spring pushes the piston down. The ball is held firmly on seat by the differential pressure. The system pressure is equalised by opening a internal bypass (Fig. 4.65)

4.6.3 Miscellaneous Valves

Miscellaneous valves include valves with special application and design. Considering their application spectrum only some of these special valves have been discussed briefly.

(i) *Gate valves.* Gate valves are popular in paper and pulp industry and those applications where the fluid is semi-solid. The gate valve design is shown in Fig. 4.66. It can be throttling or shut-off valve. The valve may be actuated by a linear actuator. When the fluid is semi-solid, the bottom of the gate is sharpened so that fluid does not stick and damage the valve.

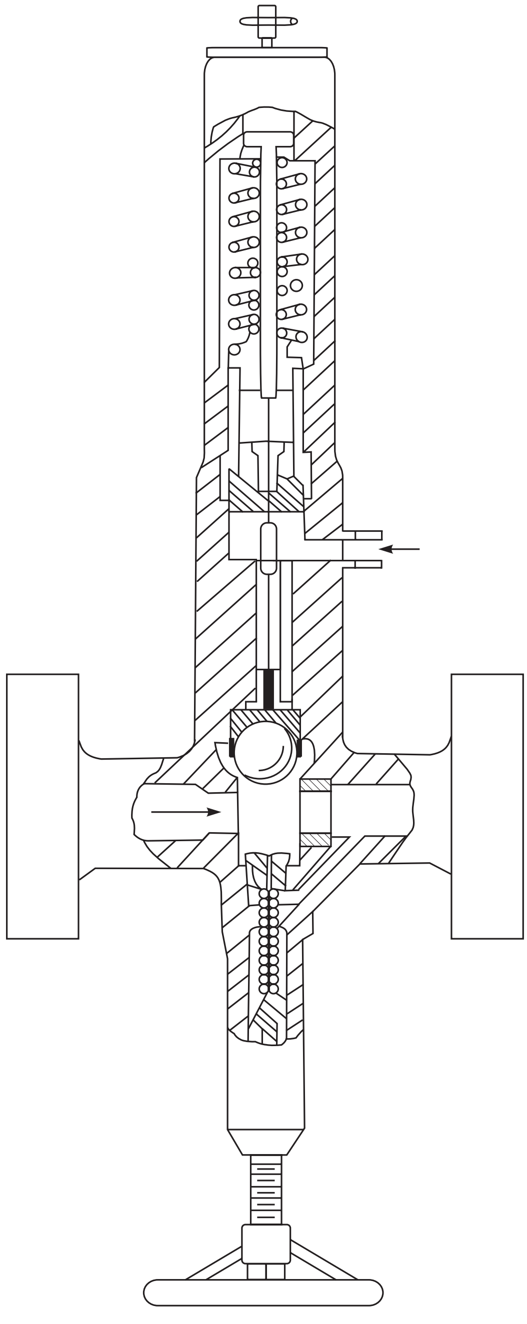


Figure 4.65 Ball and cage valve for emergency closure.

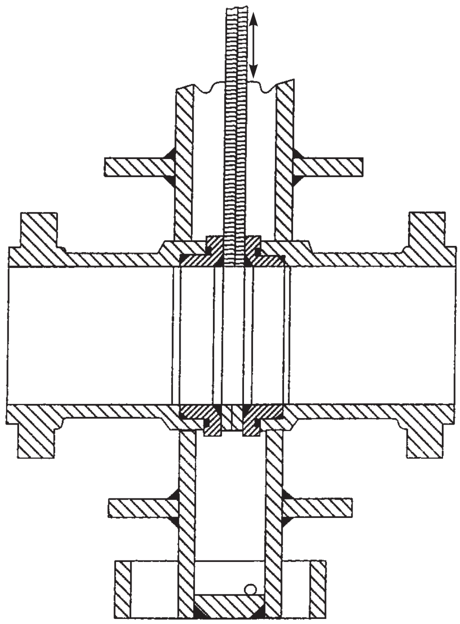


Figure 4.66 Gate valve.

(ii) *Pinch valves.* Pinch valves are popular in mining, water treatment plants, sewage disposal plants, ore processing plants, chemical industry etc. They offer high abrasion resistance, high corrosion resistance and many other attractive features. They can be mechanically clamped (Fig. 4.67) or pneumatically/hydraulically clamped (Fig. 4.68). Mechanically

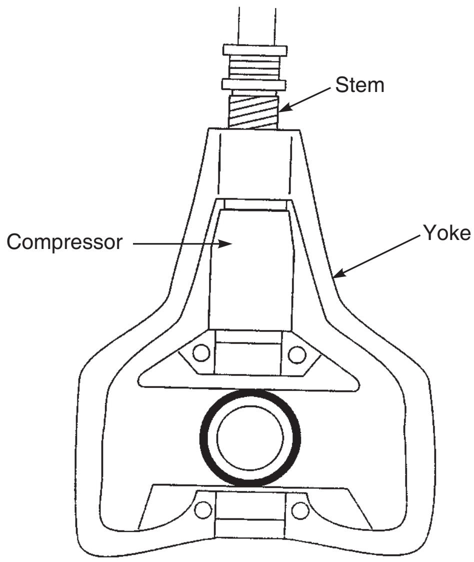


Figure 4.67 Mechanically clamped pinch valve [Courtesy—Resistoflex Corp.].

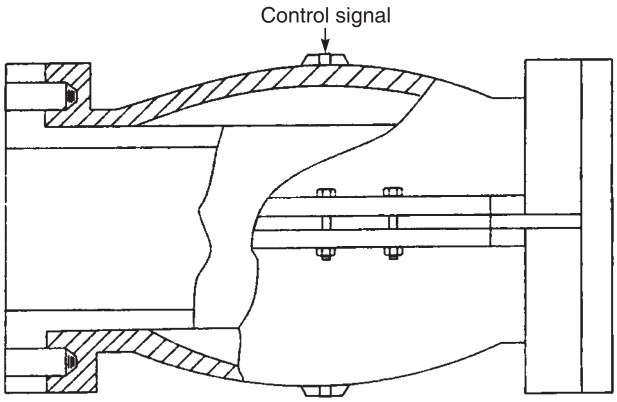


Figure 4.68 Pneumatically clamped pinch valve [Courtesy—Red Valve Co.].

clamped pinch valve may use screw clamps, rollers, wedges to compress the tube. Pneumatically/hydraulically clamped pinch valves have the tube contained within a metal housing which is operated by either pneumatic or hydraulic power. It can be made to throttle or close depending on the pneumatic/hydraulic power applied.

(iii) *Diaphragm valve.* Diaphragm valves are also known as weir valve. The design of diaphragm valve is shown in Fig. 4.69. By moving the diaphragm close to or away from weir, the flow is controlled. The compressor plays lead role in valve actuation. When pressure is applied on the diaphragm through compressor, the diaphragm moves towards weir and the flow decreases. When the pressure is withdrawn, the diaphragm moves away form weir. The diaphragm valves are similar to pinch valves and can be use in similar applications.

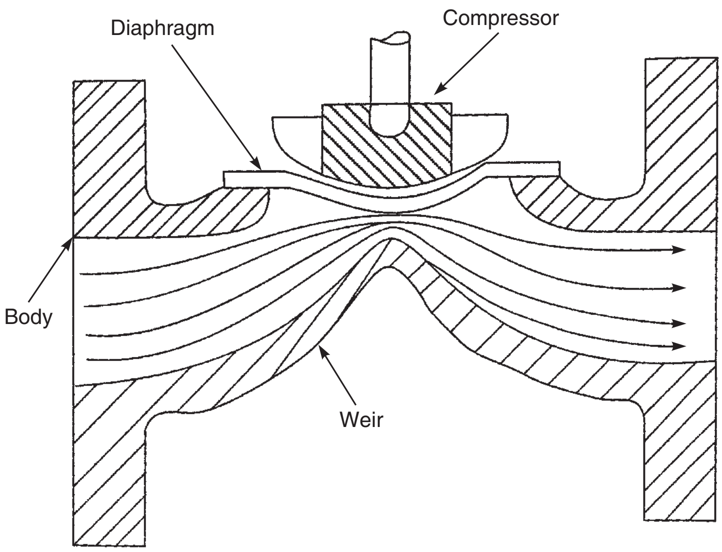


Figure 4.69 Diaphragm valve.

(iv) *Electronic solenoid operated valves.* Figure 4.70 explains the operation of a throttling solenoid valve with LVDT type position feedback. This is very useful in control applications. It has a positioner which accepts 4-20 mA signal and sends required DC signal to solenoid. The solenoid then moves the disk through plunger to affect the required throttling. The LVDT gives the current position of the valve through which the actual flow can be derived.

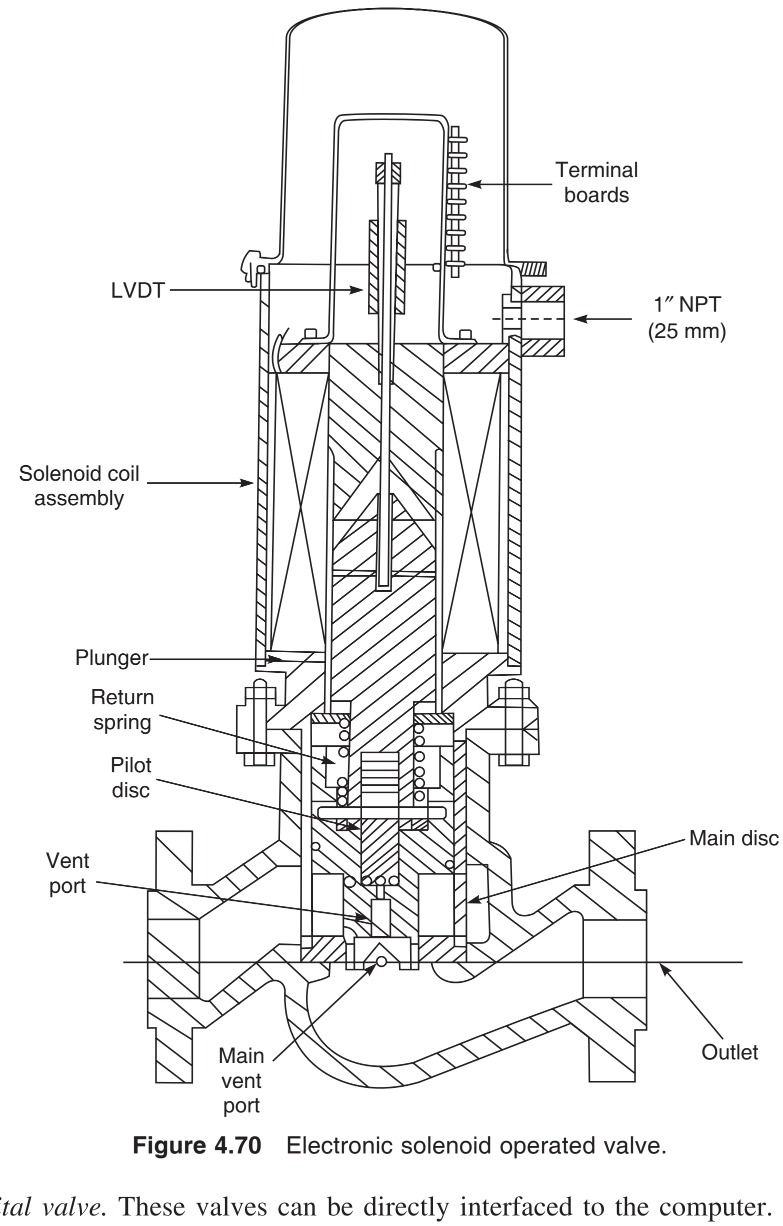


Figure 4.70 Electronic solenoid operated valve.

(v) *Digital valve.* These valves can be directly interfaced to the computer. The design of these valves is shown in Fig. 4.71. The working is shown in Fig. 4.72. It basically consists of a

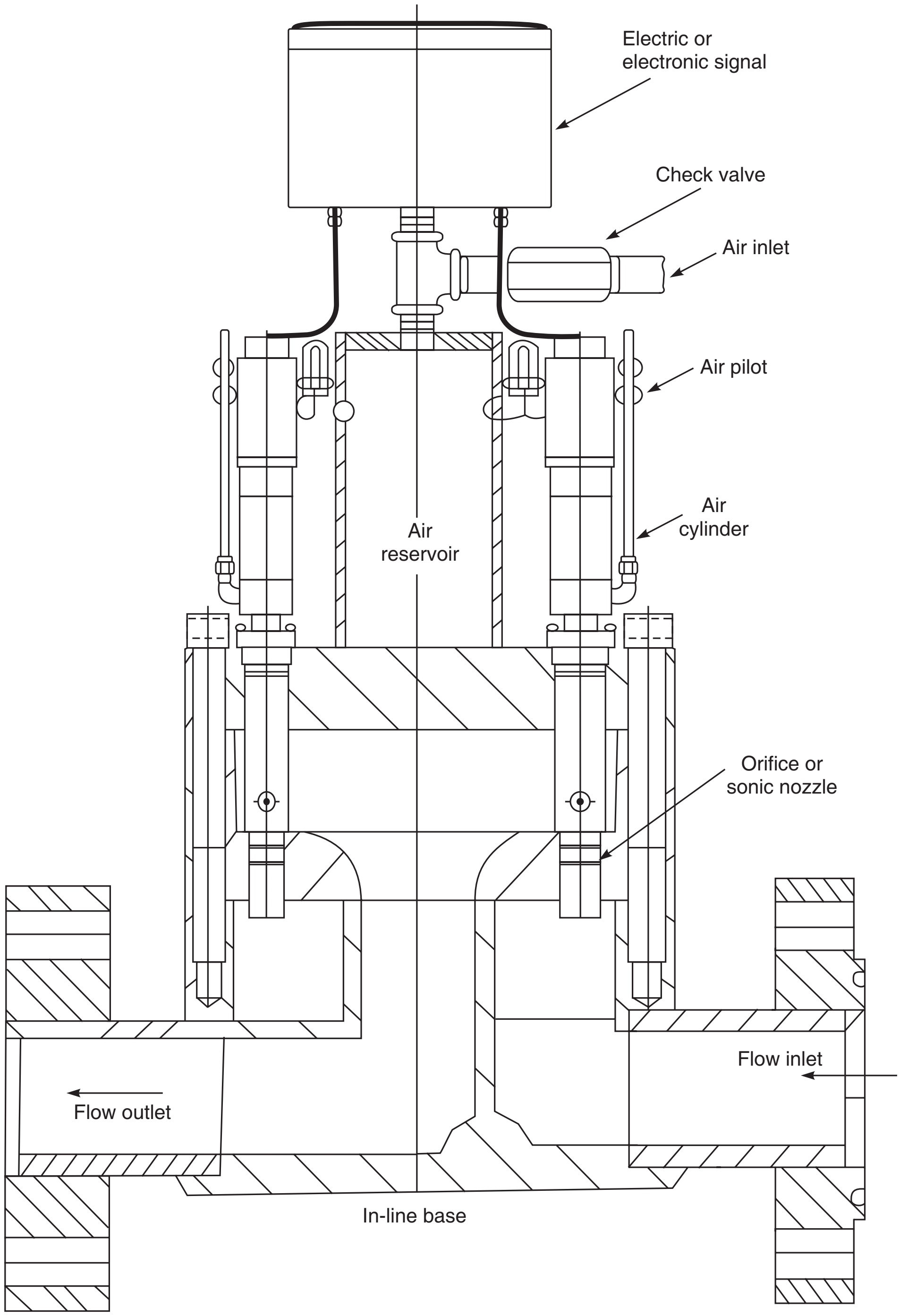


Figure 4.71 Digital valve.

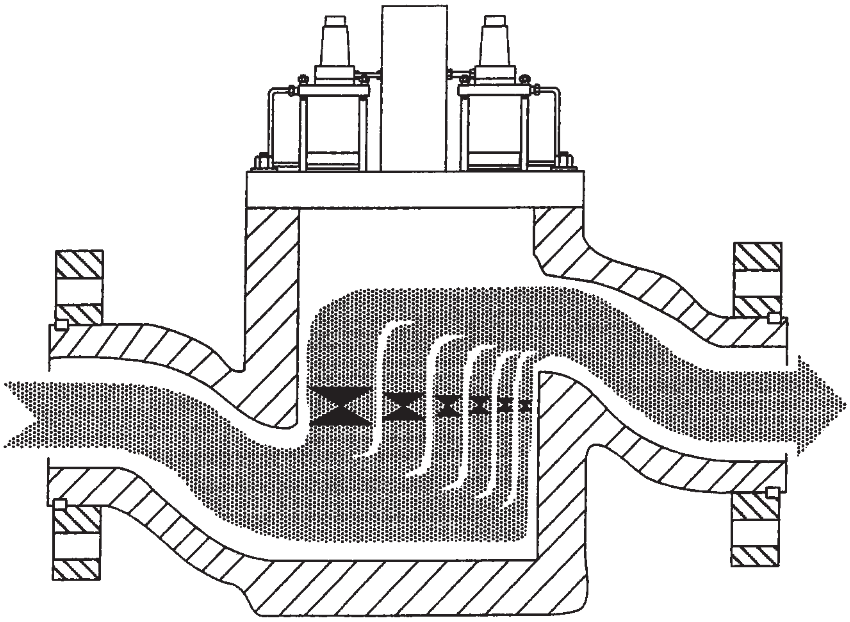


Figure 4.72 Digital valve design.

group of valve elements assembled into common manifold. These elements have binary relationship between themselves. For example, if the size of smallest element is *X* then the sizes of subsequent elements will he 2*X*, 4*X*, 8*X* and so on. Each element is controlled by individual electric or electronic signal. Thus, an 8-bit digital valve will have 8 parallel on-off electrical or electronic signals. The range span of a 8-bit valve will be 255:l. There has to be 1:l relationship between binary weighted signal and binary weighted flow area as explained above. In 8-bit valve, it is represented as,

	Binary weight signal		Flow area (per cent)
	255 (All bits	= 1)	100
(MSB)	Bit-7	= 1	50
	Bit 6	= 1	25
	Bit-5	= 1	12.5
	Bit-4	= 1	6.25
	Bit-3	= 1	3.125
	Bit-2	= 1	1.56
	Bit-l	= 1	0.78
(LSB)	Bit-0	= 1	0.39

For proper working, the leak tight construction is essential because the LSB contains 0.39 per cent flow area.

Let us consider the following example:

When MSB is ‘1’ and others are ‘0’, i.e, 1 0 0 0 0 0 0 0, opening = 50 per cent of flow area.

When number is 0 1 1 1 1 1 I 1, opening = 49 per cent flow area.

In this case the transfer from lower to higher number will not be smooth. To facilitate smooth transfer, an 8-bit valve contains two 25 per cent elements instead of one 50 per cent element.

Thus, 8-bit valve may be constructed using 9 elements with flow rating (per cent) as: 25, 25, 25, 12.5, 6.25, 3.125, 1.56, 0.78, 0.39.

Similarly, sixteen elements are present in 12-bit valve and 14-bit valve contains 18 elements. The largest element is having flow rating of 12.5 per cent.

Commercial 8-bit valve has 12 elements with following flow ratings:

Bit-0	– 0.39%
Bit-1	– 0.78%
Bit-2	– 1.56%
Bit-3	– 3.125%
Bit-4	– 6.25%
Bit-5	– 12.5%
Bit-6	– 2 Elements of 12.5%
Bit-7	– 4 Elements of 12.5%

Thus, 7 elements of 12.5 per cent are present.

Each element has a plunger and a seat. Plunger is operated by a solenoid or solenoid piloted cylinder. Figure 4.73 explains the internal working of a 8-bit digital valve. Following are the advantages of digital valves,

— Fast response
— Exactly repeatable performance
— High resolution
— Any desired valve characteristic can be provided by programming the digital command.

Digital valves of sizes 3/4" to 10" are commonly available. With pressure rating upto 10,000 psig.

4.7 MEMS VALVES

MEMS valve is a Micro-Electro-Mechanical Systems (MEMS) product designed specifically for fluid control across a number of flow control applications. This tiny silicon wafer, about the size of a button, can control the flow of liquids, mists and gases at high pressures and high flows, replacing a traditional valve as large as a standard flashlight. Its lighter weight and smaller size coupled with its true linear flow control characteristics have the potential to lead to increases in vehicle fuel economy while reducing power consumption.

A typical automobile has 50 valves that need to be opened and closed automatically and regulated closely. In addition to controlling pressure, fluid-control valves regulate the rate the flow of fluid, i.e., how brake fluid, transmission oil, refrigerants and other fluids are used in the car.

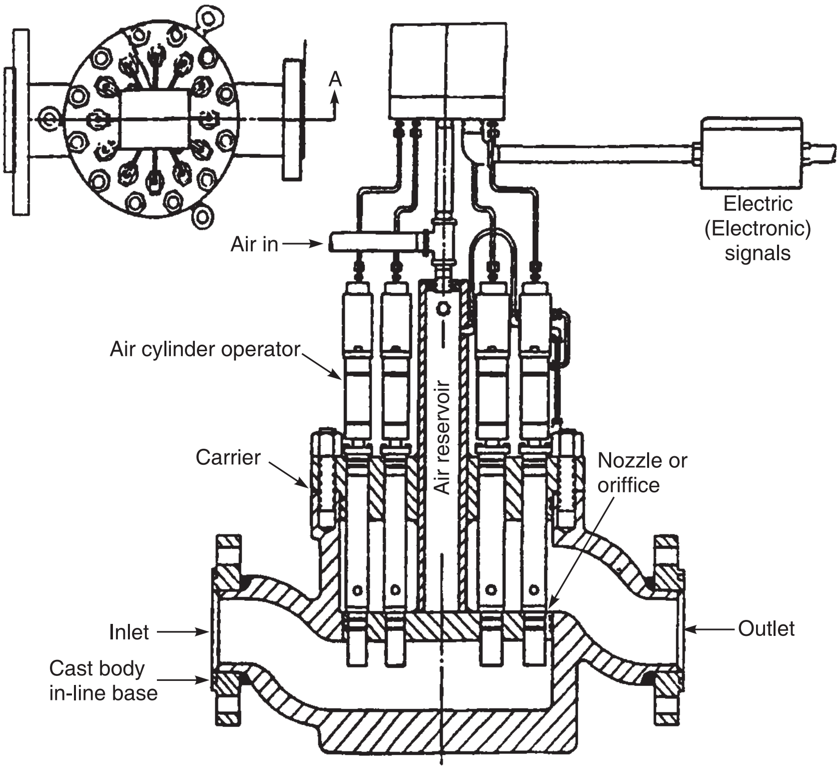


Figure 4.73 Digital valve operation.

Replacing the bulky mechanical valve devices with MEMS valves has the potential to revolutionize automotive fluidic systems designs, leading to improved vehicle mileage and reduced automotive greenhouse gas emissions. Its cost, size, weight, durability, performance and component integration capability offer strong benefits for auto manufacturers and automotive systems suppliers.

4.7.1 MEMS Valve Technology—An Introduction

MEMS is an approach to fabrication that uses the materials and processes of micro-electronic fabrication. It facilitates and conveys the advantages of miniaturization, multiple component integration, and microelectronics to the design and construction of electromechanical systems.

The MEMS valves are assembled by combining multiple layers of silicon wafers, each with a unique geometric structure etched into it or through the silicon surface. The silicon wafers are bonded together with a technique known as *fusion bonding*. Wafer cleanliness, surface roughness and alignment are the critical factors in fusion bonding. The bonding is carried out with a high-temperature annealing process.

MEMS devices have already penetrated the market in the form of microfluidic devices (e.g. inkjet print heads), telecommunication components and sensors. Microstaq is the first company to develop the technology and commercialize it for flow control applications.

Standard silicon wafers are used in the manufacture of the MEMS micro valves. The voltage characteristics of the valves are determined by selecting an appropriate resistivity of the silicon wafer. The resistivity of the wafer is controlled through a boron doping process when the wafers are made. The resistivity, specified in ohm-cm, is only critical in wafers intended to carry current.

Valve size or die size will depend on the valve design, but will typically be in the range of 1 cm^2. The thickness of the valve will vary on the basis of the thickness of the wafers used and the number of wafers used in the valve stack. A typical three-layer stack valve, like the valve depicted in Fig. 4.74, will result in a valve thickness of approximately 2 mm. Since the valve size will vary on the basis of the design, the number of valves yielded per wafer stack will also vary. Additionally, silicon wafer diameters of 4, 6 and 8 inches (or higher) can be used, which will also determine the part count per stack.

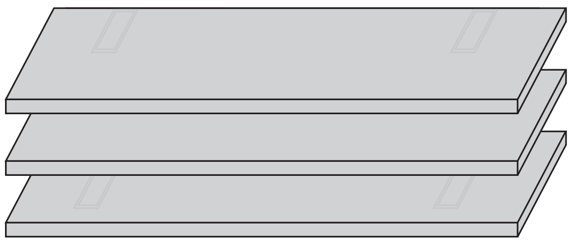


Figure 4.74 Microvalve—schematic design.

4.7.2 MEMS Control Valve

Microstaq has developed two basic types of Silicon Controlled Valves (SCV):

1. Spool valve
2. Direct acting valve.

Spool valve is operated by controlling the pressure on either side of the spool. This can be achieved by using a microvalve pilot or by using pressures in the system. The operation is similar to that of the conventional sliding stem valve, in which the sliding element is used to block or open a port. The MEMS microvalve uses a sliding plate valve, where a silicon plate moves in a cavity over openings in a port plate. The centre wafer constitutes the sliding element of the spool valve. The positioning is achieved by controlling the pressure balance across the spool. Feedback mechanism may be integrated into the device by strategically placing slots or holes in the spool. This will create variable feedback orifices that are functions of spool position.

The direct acting microvalve provides a proportional flow control capability. The valve provides a variable orifice that is proportional to the power applied. The device is driven by providing a fixed voltage and then Pulse Width Modulate (PWM), signal to provide the range of average voltages that will stroke the valve. Thus the valve will have an opening proportional to the PWM signal.

One would notice the similarity in operation between the digital valves described earlier and the direct acting microvalves.

The differences between a standard solenoid valve and the MEMS microvalve are vast and are measured in orders of magnitude. The comparison between solenoid valve and microvalve is shown in Table 4.1. The most significant difference is the size and weight of the microvalve, which is obvious just by visually comparing the valves as shown in Fig. 4.75. Since the valve is made entirely of silicon, there is no longer any concern over the susceptibility of seals and metals to fluids. The valves can operate with any fluid that is non-corrosive to silicon and at temperatures that challenge the material used to interface the microvalve to the system. The fundamental operating mechanism of the direct acting microvalve is a flow of electrical current through silicon 'ribs', which in essence are multiple resistive elements. These silicon ribs expand due to the resultant thermal expansion of the silicon and translate into a linear displacement of the valve mechanism. Unlike the inductive solenoid device, the microvalve has no electro-magnetic interference (EMI) emissions and does not require any elaborate valve drive electronics. There is the added benefit of non-interference with sensors that may be located near the flow control device. In addition, the MEMS-based valve technology lends itself to integration with MEMS-based pressure transducers as well as other sensing technologies. Valve and sensing elements can be affixed to the same manifold with a common electrical interface.

Table 4.1 Comparison between solenoid valve and microvalve

Key Metric	*SCV*	*Solenoid*	*Impact*
Size	0.8 cm^3	41.2 cm^3	98% Reduction
Weight	0.35 g	170 g	99% Reduction
Power	1A @12v	3A @12v	66% Reduction
Reliability	High	Limited	Longer life
System	Silicon	Metal	Significant cost reduction
Electronics	Resistive	Inductive	Low driver complexity
EMI	None	High	Low design complexity
Interface	Simple	Multiple bore	Low manufacturing complexity

Several versions of direct acting valve are available with flow areas ranging from 0.05 mm^2 to 2 mm^2 and operating voltages of 6 and 12 V. The valves can operate at pressures in excess of 2,000 psi and over a temperature range of – 40 to 150°C. A flow capacity test using nitrogen

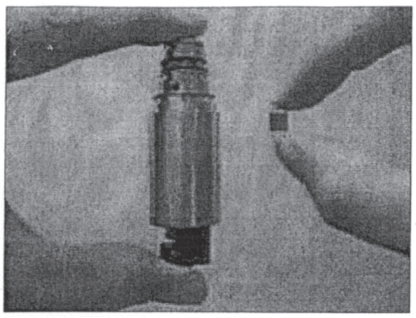


Figure 4.75 A size comparison of solenoid valve and MEMS microvalve.

(N_2) at 100 psi yields flows from 6 L/min to 40 L/min. Valve leakage will vary with operating conditions and will be greater than zero.

The spool valve has operating capabilities similar to those of the direct acting valve. Spool valves with orifice areas in the 2–4 mm^2 range have been designed. Flow capacity test in nitrogen at 100 psi yields flows in the range of 150 to 175 L/min. To operate at the high pressure limit of the direct acting valve, the spool valve must either be encapsulated with a metal cap or designed into the system in such a way that it is surrounded by the high-pressure fluid.

4.7.3 Applications

Since the MEMS microvalve is a flow control device, it has potential applications in many industries. Today's limitations are measured by the flow and pressure capacity of the valve, but these capabilities will grow as the technology is developed. Current applications span across the industrial, process control, automotive and commercial/residential markets. These applications include refrigeration, where microvalves are being developed for variable displacement compressor control and electronic control of the refrigerant's thermal expansion. In hydraulics, the microvalve has found a home as a pressure control device for transmission clutches and is also being developed for hydraulic braking applications. In addition, it is used to control hydraulic and pneumatic actuators (cylinders).

In many high-flow applications, solenoid valves are used to pilot much larger valves. Similarly, the microvalve can be used to pilot a larger flow device, eliminating the need for a solenoid valve. This 'hybrid' valve concept can be applied to many flow control applications, where the solenoid device is replaced by the microvalve which is used to drive a conventional mechanical valve mechanism.

4.8 CONCLUSIONS

This chapter presented an overview of control valves and actuators. This is definitely not complete as all aspects of control elements have not been covered. However, an attempt has been made to give a brief outline of the trend and the technology available.

SUGGESTED READING

Baumann, H.D., Trends in control valves and actuator, *Instruments and Control Systems,* Oct. 1982.

——, The use for butterfly valves in throttling applications, *Instruments and Control Systems* May 1979.

Beard, Chester S., *Final Control Elements,* P.A. Chilton Co., Philadelphia, 1969.

Colaven, M.R., Solenoid valve barrier, *Instruments and Control Systems,* August 1979.

Fernbaugh, A., Control valves; A decade of change, *Instruments and Control Systems,* Jan. 1980

Hammitt, D., Rotary valves for throttling, *Instruments and Control Systems,* Jan. 1977.

Howlesky, M.F., Adapting electric actuator for digital control, *Instrumentation Technology,* March 1977.

Laugill, A.W., New control valve accept Digital Signals, *Control Engineering,* August 1969.

Liptak, B.G., *Instrumentation Engineering Handbook,* Chilton Book Company, Pennsylvania, 1985.

Losey, J.A., Control valve update, *Instrumentations and Control Systems,* Jan. 1981.

Luckewich Mark, MEMS microvalves: the new valve world, valve world (www.valve_world.net), May 2007.

Monis Warren, Digital valve as a transfer standard, *Gas Magazine,* Sept. 1980.

Ussy, J.D., Electric valve actuators can feel safe, *Instruments and Control Systems,* May 1980.

CHAPTER

5

Display Systems

5.1 INTRODUCTION

The primary function of an information display system, as an effective communication vehicle, is to furnish spectators accurate information, presented legibly and in a format easily understandable.

The display, in addition to being dimensioned to provide adequate and sufficient information which is easily legible, must also be integrated into the architectural design of the facility. One of the major elements in the design of an effective information display system is therefore, the physical design of the display. This is in order to ensure architectural conformity with the facility i.e. control room, shop floor etc.

We shall first outline some of the parameters which can be used to assess these displays, and then discuss some of the display characteristics in terms of these parameters. As a basis for comparison, displays of existing proven design as well as those which are foreseen as future possibilities will be discussed. At the end, we shall discuss the graphic displays, their design and other considerations.

5.2 DISPLAY PARAMETERS

Since the display forms the interface between man and machine, many of the display parameters are subjective and qualitative; others can be quantified.

Quantitative parameters

Quantitative parameters are listed below together with their units of measurement:

Parameter	Units of measurement
(a) Size range	Millimetres, height of characters
(b) Character font	Matrix or segment definition
(c) Contrast ratio	The ratio of the luminous intensity of an ON element to an OFF element
(d) Viewing angle	That angle degree off-normal to a display element at which the luminous intensity of the element is reduced to 58% of the reading measured normal to the panel.
(e) Switching time	Seconds
(f) Power consumption	Watts per character
(g) Thermal operating	Degrees centigrade range
(h) Humidity operating	Percentage relative humidity, Non-range condensing
(i) Colour	—
(j) Operating life	Hours or cycles

Qualitative Parameters

The qualitative parameters which can be assessed on the basis of "Good-Poor", Simple-Complex", etc. are listed here:

1. Legibility—Appearance as finished display and ease of viewing
2. Brightness—Measure of the light flux incident on a surface
3. Display resolution
4. Resistance to short term power failure
5. Mechanical strength
6. Complexity of drive circuitry
7. Maintainability
8. Status
9. Cost per character

We shall now discuss some of the important parameters and their effects.

Viewing distance and size

Considerable engineering efforts have been expanded over the years in determining the best display media for maximum readability and legibility.

Of course, if the letters and other graphic elements are large enough, they can be read at any reasonable distance. In instances where the display is used as an output device, certain guidelines should be followed, particularly if the display is to be viewed for long periods of time.

The viewing distance is determined by character size, line and character spacing, viewing angle and ambient light There is a direct relation between the character size and viewing distance. In general, rule-of-thumb ratio of 500 : 1 between maximum viewing distance and character height is applicable, i.e. a viewer with normal eyesight will be capable of recognizing and reading characters which are 1/500 of viewing distance. Then at a viewing distance of 10 m the characters should have minimum height of 20 mm. However, to assure rapid assimilation of the messages as well as to ensure that even the spectators with poorer eyesight will be able to read the displayed information easily, the calculated letter height should be increased by a factor of 20% to 50%. Therefore, if a given viewing distance is 25 m the standard letter height should be at least:

$$25{,}000 : 500 = 50 \text{ mm} + 20\% = 60 \text{ mm}.$$

Another factor influencing legibility is the horizontal spacing between characters and the vertical spacing between lines. In practice, a minimum line-to-line spacing of 10/7 of the character height, and a character-to-character spacing of 7/5 is adequate. The character width is required to ensure a satisfactory and easy-to-read display, especially when all character positions of a display board screen are fully utilised for the display of information. Full-matrix displays allow selection of character spacing, for optimum effect between maximum display capacity and maximum legibility. Wider spacing also permits both horizontal and vertical lines to be drawn. Underlining words or phrases for emphasis, and boxing sections of the board for clear presentation of information are then both possible without visual crowding.

On the other hand, in applications where we would like to view the entire screen in a single glance, such as in movement control monitoring, the screen should be far enough to allow good visual acuity over the entire surface.

For instrumentation and control applications, displays are normally installed indoors only, often in air conditioned environment. Thus, viewing distance is not always a relevant factor except in case of mimic displays which is normally put at quite a distance from the control panel. In such cases, control panel may consist of 61 cm colour VDU with keyboard for interaction.

Contrast

The direct relationship between luminance levels and the ability to perceive luminance differences explains as to why the contrast of a display is normally expressed as a ratio. The absolute luminance values of the image and background surfaces of a display are, within reasonable limits, not nearly as important as their relative luminance values. In fact, the contrast ratio of a display device is probably the most important single characteristic contributing to the legibility of a display and its “looking-good” appearance.

Unfortunately, the contrast ratio is also the most difficult display parameter to predict with precision, and this difficulty is compounded when an attempt is made to compare different display technologies, or even different devices within the same technology. In theory, the contrast ratio of a display device can be simply calculated as follows:

$$\text{Contrast ratio} = \frac{\text{Maximum luminance}}{\text{Minimum luminance}} = \frac{\text{BD (Display)}}{\text{BB (Background)}} = \frac{B_{on}}{B_{off}}$$

It is then a matter of personal preference whether the value is expressed as a ratio (e.g. 7:1), a numerical value (7.0), or as a percentage (700%).

The following factors influence contrast of the display devices:

— Construction of display device
— Brightness
— Light emission spectrum
— Reflective properties of display device
— Viewing position
— Illumination

The effective maximum and minimum luminance values may not, however, be easily obtained. Many manufacturers of display devices present only the "intrinsic" contrast ratios of their displays—without taking into account the effect of reflected ambient light. The environmental conditions in which the displays may be used are too variable to anticipate. This leaves it up to the display-system designer, or user to predict (or suffer with) the "extrinsic" contrast ratio, which can be significantly lower. Reflected ambient light increases both the numerator and denominator of the maximum/minimum contrast-ratio calculation. This can sharply reduce the effective contrast, with corresponding reductions in legibility and gray-scale range. For example if reflected ambient light has a luminance equal to half the maximum image luminance, an intrinsic contrast ratio of 10 : 1 is reduced to an extrinsic contrast ratio of only 2.5 : 1 (10 + 5 divided by 1 + 5).

To complicate matters further, a number of other contrast-calculation conventions have been established, some of which result in values identified as "contrast" rather than "contrast ratio". But the two terms are generally used interchangeably in the same written material. It is advisable, therefore, to determine the ways in which contrast ratios have been calculated before using them for comparison. The following relations, for example, may be encountered:

$$\text{Contrast} = \frac{\text{Background} - \text{Image}}{\text{Background}}$$

$$\text{Contrast ratio} = \frac{\text{Background} + \text{Image}}{\text{Background}}$$

The first formula may be positive or negative, depending on whether the display image is black-on-white or white-on-black. It can also be smaller or greater than unity. The second formula assumes that the background luminance contributes to the image-element luminance, even in the absence of reflected ambient light.

Another way to specify contrast is to state the "shades of gray" which the device can display. As noted earlier, the minimum discernable luminance difference is a function of the luminance level—which is approximately 3% for most observers. The shades-of-gray convention uses an arbitrary multiplier (value) of 1.4 (the square root of 2) for each gray-scale step.

In small indoor displays, excellent contrast can be achieved by using LED displays with proper filters. Alternatively, LCDs inherently provide good contrast for easy viewability. Some of the LED displays are also viewable in sun light. A more contemporary display widely used for distant viewing uses EMD technology. EMD (Electro-Magnetic Display) uses a dot matrix pattern of character in which each individual dot can be electro magnetically changed to black or bright against a black background, thus providing unmatched contrast characteristics.

Viewing angle

As the viewing position progresses further away from the display, the angle at which the surface is viewed becomes more critical. When the spectator is viewing the display from a nearby position, a simple movement of the head will narrow the angle. When viewing is from a greater distance, the viewer may have to move several feet in the direction of the display to obtain a more direct view.

Objectionable distortions begin to take place at horizontal angles beyond 30 degrees. At 45 degrees, distortion is objectionable in any application. Consideration must also be given to the vertical angle at which the spectators view the display. Generally any angle beyond 30 degrees above the horizontal plane of the spectator's eyes will cause viewing discomfort.

Operating life

With most electronic displays, operating life means the average time to halve the original brightness.

Legibility

The first requirement of a visual display device is that the displayed information must be visible clearly, accurately, and without ambiguity under the conditions of use. These characteristics can be grouped under the term "legibility".

It is evident, that three major factors, viz. contrast, luminance, and size of the elements affect legibility. In order to provide the maximum amount of information within a display of a given size, it is an advantage to keep the size of the graphic elements (e.g., line width, character height) to a minimum which can be established primarily by the viewing distance. But the legible minimum is directly affected by the other two factors, i.e. contrast and luminance. With a given graphic element size, legibility can he significantly enhanced by increasing either one or both of these variables.

The human eye (or more precisely, the eye-brain system) is a differential-input device. It is designed to gain information by interpreting the differences detected in colour values and luminance levels. This outlines the importance of contrasting colours and luminance contrast, which can be generally defined as the difference between the colour or luminance. This represents information and the corresponding values for the "background" or non-information areas of the display surface.

The ability to discern luminance differences is the more primitive attribute, and is far more important than colour detection in terms of information acquisition and interpretation. More than three-fourths of the light-sensing cells in the human eye are colour blind at low luminance levels and the eye can see only shades of gray. The bandwidth of a "full colour" television colour signal averages less than one third that of the luminance signal. Colour is, in effect, a bonus display quality, a valuable enhancement, but only after the more fundamental legibility contrast has been fully met.

Display resolution

The "resolution" of a display provides a first-approximation measure of the amount and variety of information which can be presented on the display surface. Resolution also helps to determine the aesthetic appearance of a display, independent of the displayed information.

The term Display resolution may he defined in various different ways. It may refer, for instance, to either the "density", or the total number of addressable display elements. The elements themselves may be defined as dots, lines, image-background transitions (with two transitions for each dot or line), or hypothetical grid intersections.

Similar considerations are applicable to computer-based display systems. Source, data and display device resolutions are independent variables. If the emphasis is on the physical display, we can use the total number of separately addressable display elements under the control of the system software, or operator as our resolution measurement.

Expressed in this manner, resolution becomes a power factor, with exponential increase in the potential amount of displayed information, as the number of display elements increases linearly.

The exponential nature of the resolution relationship implies that as the number of elements reaches into the hundreds, or thousands, further increase in resolution becomes meaningless in terms of the gross number of patterns, or images which can be formed. The benefits of increased resolution include enhanced legibility and access to finer detail (number of readable characters, density of lines, subtle graduations in gray-scale intensity).

5.3 DISPLAYS IN PROCESS CONTROL ENVIRONMENT

The 'central core' of the latest control and data processing systems is essentially latest computers with one or more microprocessors. In contrast to the classical analog techniques the new systems utilise digital signal processing and all input and output are represented in terms of *numbers.*

It is interesting to note that human brain is not quite receptive to the numbers and work much more efficiently with pattern recognition. This is perhaps because numbers are of recent origin in the history of human evolution. For this reason, it is good idea to mimic classical display technique like displaying hands in digital wrist watches. It is also effective to display curves than actual numeric results. This is achieved using Computer Graphics.

Through computer graphics, it is possible to illustrate the complex processes in real-time. By using colour graphics, advanced animation techniques and sophisticated software it is possible to generate three-dimensional display of a plant in working condition.

5.4 COMPUTER GRAPHICS

Computer graphics play a significant role in establishing men and machine interface. Computers could collect large amount of data, reorganise, carry out complex computation and print thousands of lines of results in few minutes. However, all these informations when presented to the manager of the plant for decision making, more time is consumed to read and interpret the information to assess the actual state of the process. The numbers are artificially generated codes for accurate representation and computation, but our brain further processes these numbers and converts them to pictorial form for storage, recall and decision making applications. Thus computer graphics is one of the most powerful media to strengthen the link between the computer and the human operator and increase the throughput. Using computer graphics, it is possible to represent the results in pictorial form thus reducing the processing requirement of brain. Thousand of data points could be viewed at a glance if these are plotted systematically. More complex data having more than two variables may be represented using three-dimensional plot of advanced graphics techniques.

Incoming data from different parts of a large plant could be quickly interpreted if these informations are displayed on the symbolic or diagrammatic representation of the plant, by mapping the monitored data on its geographic origin (Fig. 5.1) Complex displays could be further simplified using colour graphics which allow overlapping of various diagrams, curves and write-ups without loosing clarity. Colour is also very useful to emphasize critical parameters, limit values and other important parameters. By schematic diagrammatic representation of a plant or process, it becomes easy to monitor and control the process.

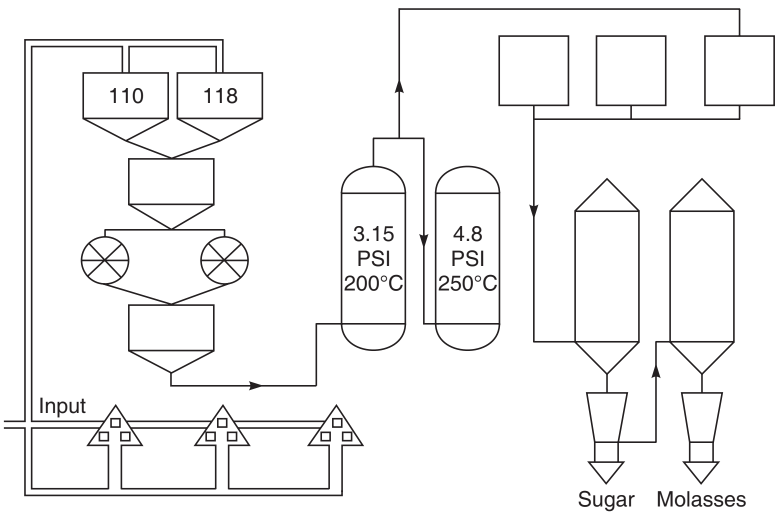


Figure 5.1 Symbolic representation of plant.

Another important use of computer graphics is simulation. With the help of computer, simulation software and colour graphics it is possible to simulate actual working of a plant. Simulation is extremely useful for design, research and specially manpower training. It is appropriate to recall that pilots of modem aircrafts get their training through similar computer graphics flight simulators. It is possible to simulate the faulty conditions or emergency situations in the plant which are usually difficult to experience in a real plant.

5.4.1 How Graphics Work

Computers generate graphics through colour ink-pen plotters, photo plotters, plasma display, cathode ray tube display (CRT) and many other forms. Amongst all, CRT graphics is the most popular and is used for industrial controllers and monitors. Hence we will restrict our discussion to CRT graphics. Two different methods of display are used in CRT graphics, viz.

— Vector Graphics Display

— Raster Graphics Display

In vector graphics, lines are drawn on CRT screen by actually moving the electron beam between two end points of the straight line, like we draw lines on paper. Although this seems to be a straightforward method but due to hardware implementation complexity vector graphic display is not popular for CRT graphics except in high resolution application like CRT photo-typesetting.

Raster graphics is most commonly used computer graphics. Raster graphics use CRT which is used in television sets and is known as TV monitor. Most home computers use this type of display. In TV image, the beam moves along the horizontal scan lines from left to right (the movement of the electron beam from right to left being blanked). These are called “Scan lines”. The intensity and colour of the beam is modulated to reproduce a picture. When the electron beam completes scanning one cycle from the beginning at upper left corner to the end at lower right corner, one raster is formed.

There are two types of scanning used in various CRT applications. One is “non-interlaced scanning” in which only one beam of electrons scans across the CRT screen to form a raster. Typical examples are computer monitors at low resolutions. The other is “interlaced scanning” in which two electron beams scan the alternate scan lines producing one raster. Domestic TV monitors use interlaced scanning.

Consider a diagonal straight line on TV screen (Fig. 5.2) which in fact consists of series of illuminated dots on each horizontal line. A straight line which appears continuous on the screen, actually consists of the dots appearing at different times. It is possible to compute the dot timing using a microprocessor which depends on the end coordinates of the line. However it is difficult for the microprocessor alone to beam these dots to the screen in time. It will be a very slow process, as for a resolution of 512 × 256, it will be necessary to send these dots as fast as 100 nano-second (ns) or 10 mega-cycles per second. This is achieved by keeping a separate memory (sometimes dual ported memory) outside the computer which with the help of high speed hardware sends the dots faster to the screen.

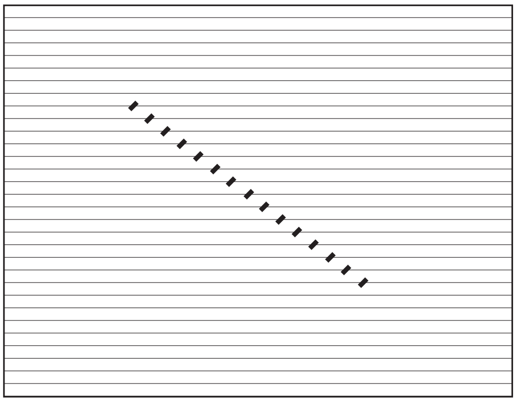


Figure 5.2 Diagonal straight line on TV.

5.4.2 Computer Graphic Generation

Figure 5.3 shows a simplified block diagram of computer graphics scheme. In brief, there are four major blocks:

— Display/Video RAM
— Character generator
— Video serialiser (Parallel to serial shift register),
— Control and timing.

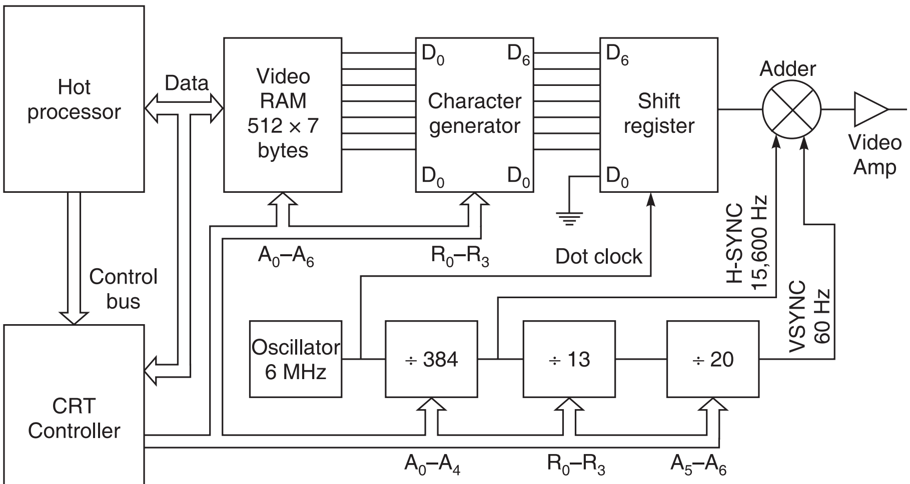


Figure 5.3 Computer graphics circuit block diagram.

The RAM stores the ASCII or EBCDIC codes for the characters to be displayed on VDU and can be altered by the microprocessor when some new characters are to be displayed. For 80 column of 25 characters each, the memory requirement will be about 2 kilobytes, considering atleast one byte per character. In VDUs for process control applications, this memory can be of the order of 2 megabytes because, each pixel on the screen requires 1 byte of data for various attributes.

The function of the block diagram is as follows. To start with first page of display, the character counter and the row counter outputs are all cleared (0) by the CRT controller to address the first character in the display or video RAM. ASCII or EBCDIC code corresponding to the addressed character will be available from the RAM. This output is applied to the data input lines of the character generator.

Character generator is a RAM which stores the dot matrix pattern of all characters to be displayed. MC6571 is a typical such character generator.

The output of row counter is applied to the RAM. The character generator then outputs a 7-bit dot pattern from its contents to form the first dot row of the first character. The parallel output from the character generator must be converted into a serial stream of bits in order to turn on and off the electron beam as it sweeps across the screen. The eight data input is tied to logic low (0) so as to provide a dark dot gap between the characters. The high frequency clock at the input of the 'parallel to serial converter', for serializing the character generator output is called "dot clock". This is because it controls the rate at which dot information is sent out of the video amplifier.

After the dot pattern corresponding to first character row has been clocked out, the character counter is incremented by one, so that it points to the next character row in the video RAM. This process continues until all the dot patterns corresponding to the first rows of all the 80 characters are clocked out. After that the horizontal synchronization pulse is generated to cause the sweeping beam come back to the left of the screen once again while turning off the beam.

Once the beam has retraced back, the character counter is reset to zero so that it points to the first character row in the video RAM. The row counter is however incremented by 1 so that character generator can now output dot pattern corresponding to second row of first character. After clocking out the dot pattern, the character counter is again incremented by one to repeat the process.

This process is continued till dot patterns for second rows of all characters are clocked out. After that again a horizontal synchronization pulse is generated to reset the sweeping beam. In this way, the electron beam sweeps across the screen and the dot patterns modulate the beam in accordance with serial bit patterns output by the shift register. Once the beam has reached to bottom right comer of screen, one vertical synchronization pulse is generated to bring back the beam to the upper left comer, and the entire process is therefore repeated to keep the screen refreshed. To avoid blinking in display, the screen must be refreshed 30 to 60 times a second.

Video-RAM is multiplexed between CPU and graphics display screen. It is the CPU which interprets the input commands, computes the display coordinates and places them to Video-RAM. This process takes considerable amount of CPU time and makes the display slow. In order to make display faster like a real-time animated film, dedicated special processors are being used. These LSI devices are called Graphics Display Processor (GDP) or Video Display

Processor (VDP). Intel's 82720 and Texas Instruments TMS 9918, TMS 9929 are some of the examples. GDPs reduce CPU overhead considerably by taking direct commands like:

— Plotting a point
— Drawing lines
— Vectoring symbols and character sets
— Zooming and panning
— 3D simulation
— Colour display
— Supporting light-pen.

Texas Instruments TMS 9929 is specially suitable for Industrial Process Control Application. Table 5.1 lists the specifications of TMS 9929. A special feature of TMS 9929 VDP is its sprite graphics. It is possible to generate three-dimensinal (3D) effect by sprite graphics and object could be moved on the screen very fast. Consider the example of Fig. 5.4, a picture containing landscape and a car is represented in several overlapping planes, as if picture is drawn on transparent sheets and stacked together. Any of these planes could he moved with respect to other planes. In this case, if the plane with car's image is moved, it will appear as if running on a road through the bushes and the house giving a live 3D effect.

Table 5.1 Specifications of TMS 9929 Video Display Processor

1.	Horizontal resolution 256 pixels
2.	Vertical resolution 192 pixels
3.	16 colours
4.	32 sprites for simulation of 3D
5.	Four display modes
	(a) Graphics I (256 by 192 dots-limited colour)
	(b) Graphics II (256 by 192 dots-extended colour)
	(c) Text mode (24 lines of 40 user-defined characters)
	(d) Multicolour mode (64 by 48 low resolution positions)
6.	Composite video output
7.	External video and sync inputs
8.	Real-time interrupt output
9.	16 kilobytes video RAM
10.	Automatic, transparent dynamic RAM refresh

5.4.3 Interfacing Graphics

As we have seen that the graphics is an inseparable part of the computer. It is the computer which interprets or routes the graphics commands. Interface between computer and graphics is of two types:

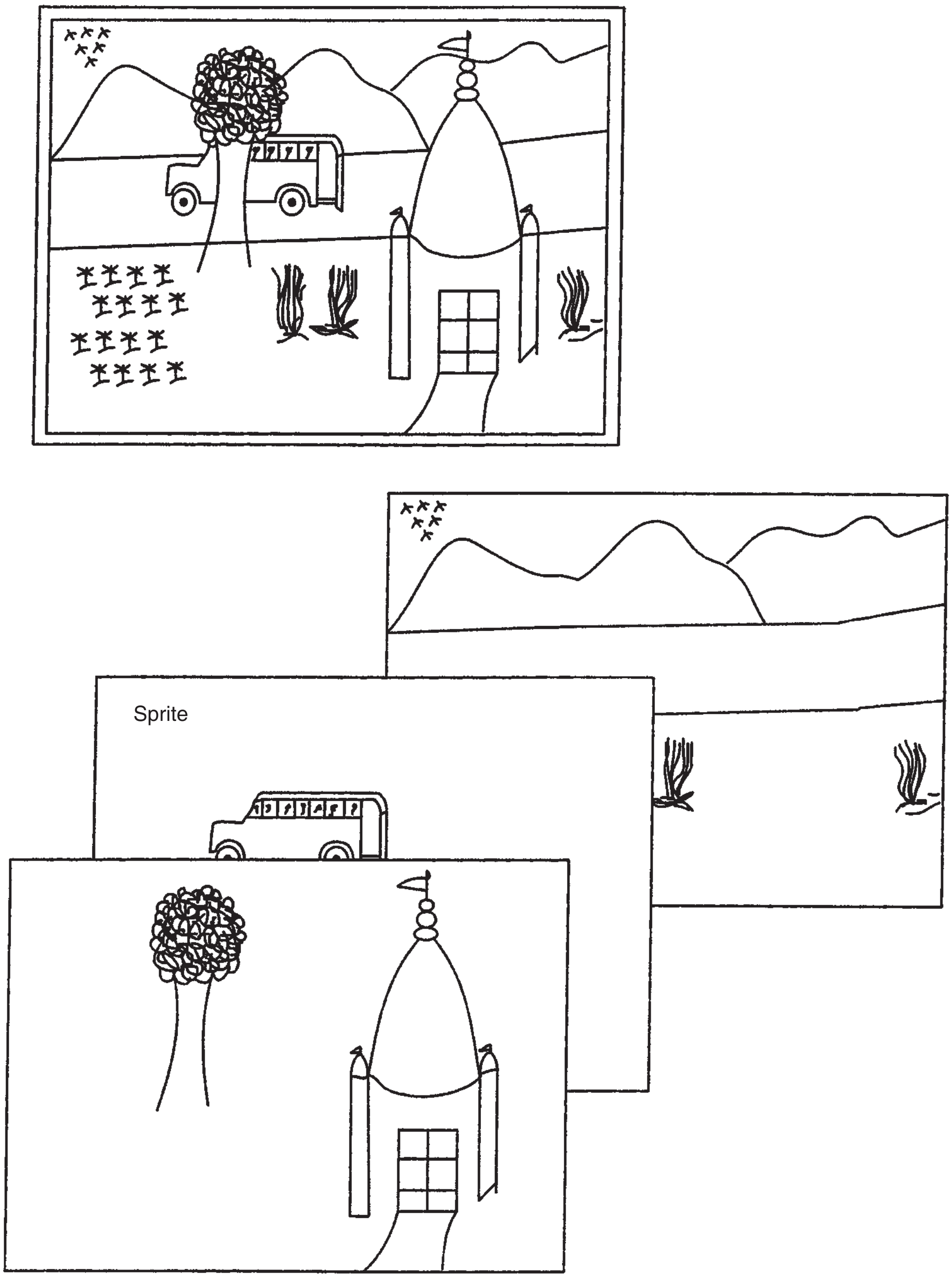


Figure 5.4 Sprite graphics—An example.

1. Memory mapped
2. I/O mapped

When Graphics RAM or Video refresh RAM forms part of computer memory, it is known as *Memory Mapped Graphics*. In this method communication between computer and graphics is fast but computer has to sacrifice a large portion of its memory for the same. In I/O Mapping, input output ports are used to communicate with graphics circuits. This method has become popular after introduction of GDP's and VDP's. Although communication between computer and GDP is slow, GDP itself reduces large computational burden from host computer.

Another invisible interface is software interface. As computer has to interpret input requests, process and transfer them to Graphics hardware, it is necessary to have some kind of operating system or software package to accomplish all these functions. It is achieved either by using primitive machine code software or high level language. To utilise graphics resources without knowledge of hardware details and complex timing requirement, high level languages are interfaced to graphics. Basic and Logo are some of the popular languages used for computer graphics.

5.5 CONCLUSIONS

The function of a visual display is to communicate information to the human brain by creating patterns of light, which can be sensed and interpreted by the human eye-brain visual system. Display requirements, and display comparisons are dictated, therefore, by both machine and human considerations as given below:

- What type and volume of information must be communicated?
- What is the required rate of data transfer?
- How fast can the viewer assimilate the information—without confusion or error?
- To what extent does the display technique take advantage of the observer's visual capabilities?
- Even more important, how effectively does the display exploit the limitations of the eye-brain linkage?

There are a number of other considerations, on both sides of the interface, which may be secondary to the communication task yet, often or not. may dictate the choice of a particular display technology, or device. The system designer must take into account, the following:

- The physical form of the display, including factors such as the display area, size of available devices and the depth of the device package as a function of the display area.
- Safety or power-supply constraints.
- Economic evaluations of display device and electronic circuitry required to drive the display.

The technical and economic characteristics of a display are subject to engineering analysis. The human factors influencing the design of a display system are not so easily defined. The display should not only communicate information, but should also "appear good", by whatever standard is likely to be applied by the intended viewer. Following considerations are taken into account:

— Aesthetic taste
— Cultural conditioning
— Intelligence
— Attentiveness
— Patience.

SUGGESTED READING

Aronson, R.L., CRT terminals make versatile computer interface, *Control Engineering,* April 1970.

Bylander, F.C., *Electronic Displays,* McGraw-Hill, New York, 1979.

Dallimont, R., New design for process control console, *Instrumentation Technology,* Nov. 1973.

Ellis, R.K., Color graphics CRTS provide window into factory operations, *Instrument and Control Systems,* February 1983.

Farmer, E., Design tips for modem control room, *Instrument and Control Systems,* March 1980.

Harrington, S., *Computer Graphics—A programming approach,* 2nd ed., McGraw-Hill, New York, 1987.

Liptak, B.G., *Instrument Engineers Handbook,* Chilton Book Company, Pennsylvania, 1985.

Mass, James W., *Industrial Electronics,* Prentice Hall International, London, 1995.

McCready, A.K., Man-machine interfacing for the process industries, *In Tech,* March 1982.

Plastock, B.A. and Kalley, G., *Theory and Problems of Computer Graphics,* McGraw-Hill, New York, 1986.

Pratt, W.K., *Digital Image Processing,* John Wiley and Sons, New York, 1978.

CHAPTER

6

Direct Digital Control—Structure and Software

6.1 INTRODUCTION

The advent of microprocessor has changed the field of process control completely. The tasks which were performed by complex and costly minicomputers are now easily programmed using microcomputers. In the past computer was not directly connected to the process but was used for supervision of analog controllers. The analog controllers were interfaced to the process directly as well as through specialised control for dedicated functions (Fig. 6.1). The analog controllers and specialised controllers were called level 2 and level 1 control respectively.

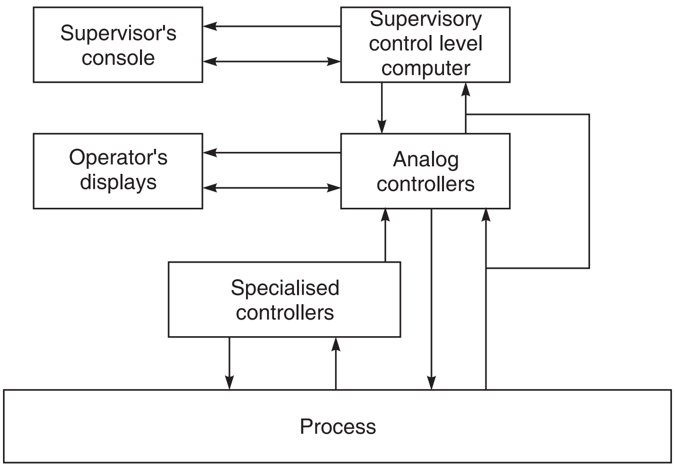


Figure 6.1 Supervisory computer control.

The emergence of economical and fast microprocessor has made analog controllers completely out-dated, as the same functions can be performed by digital computers in more efficient and cost effective way.

6.2 DDC STRUCTURE

The DDC (Direct Digital Control) directly interfaces to the process for data acquisition and control purpose. That is, it has necessary hardware for directly interfacing (opto-isolator, signal conditioner, ADC) and reading the data from process. It should also have memory and arithmetic capability to execute required P, P + I or P + I + D control strategy. At the same time, the interlace to control valve should also be part of DDC. Figure 6.2 shows the various functional blocks of a direct digital control system. These functional blocks have been described in number of books on microprocessor. The multiplexer acts like a switch under microprocessor control. It switches and presents at its output the analog signal from a sensor/transmitter. The analog to digital converter converts the analog signal to digital value.

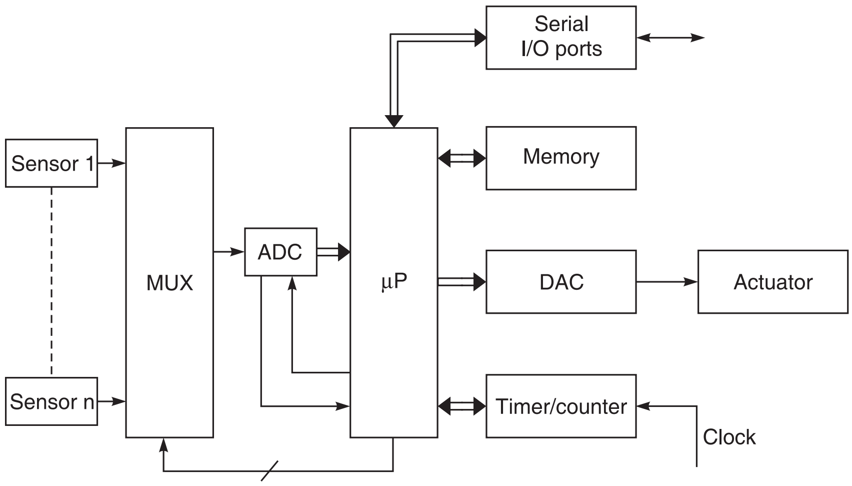


Fig. 6.2 Direct digital control.

The microprocessor performs the following tasks.

1. It *reads* the various process variables from different transmitters through multiplexer and ADC.
2. It *determines* the error for each control loop and executes control strategy for each loop.
3. It *outputs* correction value to control valve through DAC.

6.3 DDC SOFTWARE

The main part, DDC software is program for control loops. There are two algorithms for programming a three-mode PID control loop:

— Position algorithm
— Velocity algorithm.

6.3.1 The Position Algorithm

The three-mode controller has been explained in Chapter 1. The PID correction has been derived in Section 1.8.5. It can be represented by,

$$Y_n = KP \cdot e_n + KD \cdot \frac{\Delta e}{\Delta t} + \frac{1}{KI} \int_0^n e \cdot \Delta t + Y_0 \tag{1}$$

where

Y_n = valve position at time n

Y_0 = median valve position

KP = proportional constant = 100/PB (where, PB = proportional band in per cent),

KI = integral constant = $1/TI$ (where TI = integral time constant)

KD = derivative constant = TD (where TD = derivative constant)

e_n = error at instant t_n = $(S - V_n)$

V_n = value of controlled variable at instant t_n

S = set-point

The PID control can be realised with a microprocessor based system, if only the above equation is implemented in the software. Apparently, it is very difficult to write the software for implementing the above equation for a microprocessor based system. However, the above equation can be modified such that its software implementation becomes easy. The modifications are discussed in the following section.

The integral term at any given instant t_n is equal to the algebraic sum of all the control forces generated by the integral control action from the beginning to that instant.

Thus integral term can be represented as

$$\frac{1}{KI} \sum_{t=0}^{n} e_t \cdot \Delta t$$

and the differential term, $KD \cdot \Delta e/\Delta t$ at any instant t_n is proportional to the rate of change of the error.

Thus, differential term, can be represented as

$$KD \cdot \frac{e_n - e_{n-1}}{\Delta t}$$

where e_n is the current error and e_{n-1} is the previous error calculated at instant t_{n-1}.

Thus, with these modifications the three-mode controller equation will become:

$$Y_n = KP \cdot e_n + KD \cdot \frac{e_n - e_{n-1}}{\Delta t} + \frac{1}{KI} \sum_{i=0}^{n} e_i \Delta t + Y_0 \tag{2}$$

The integral and the differential control forces are dependent upon the interval between the two consecutive errors. This interval is the inverse of the rate at which the value of the controlled variable is measured i.e. the sampling rate. Hence the provision for defining the sampling rate should be made available in the software.

The flow-chart for calculating PID control output based on above equation (Eq. 2) is shown in Fig. 6.3

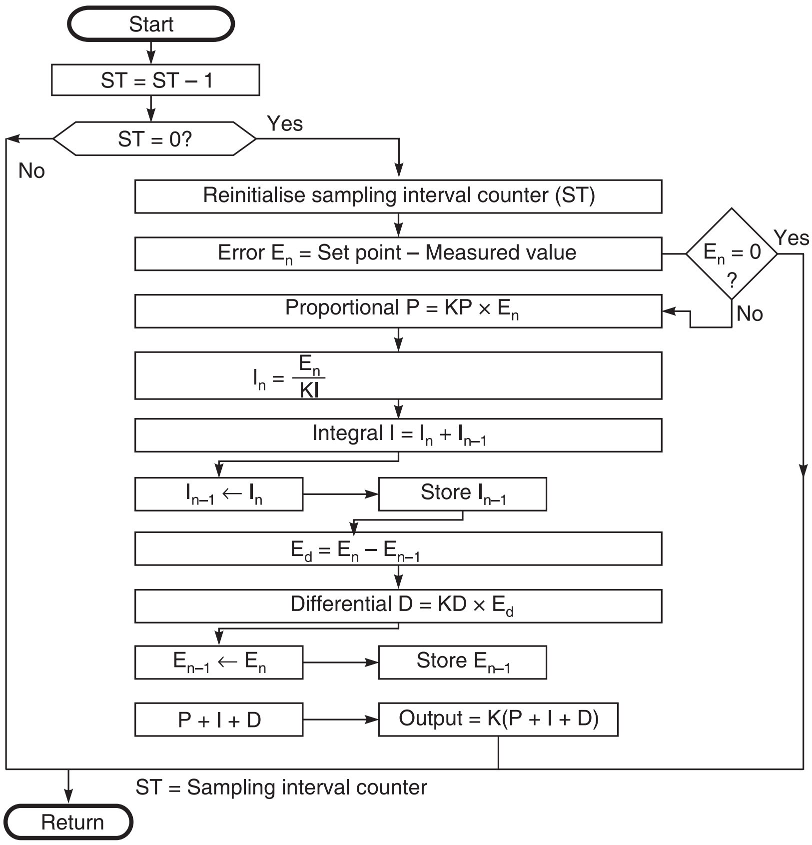


Figure 6.3 Flow chart of PID control.

The 'sampling interval counter', the set-point, the proportional constant *KP*, the integral constant *KI* and the derivative constant *KD* are defined by the user.

The two modifications that can be performed on above equations are Trapezoidal rule for integral term and interpolation technique for derivative term.

Trapezoidal rule for integral term

The integral term can be represented by using trapezoidal rule.

$$\sum_{i=0}^{n} \frac{e_i + e_{i-1}}{2} \Delta t \qquad (3)$$

This will give better accuracy than previous term,

$$\sum_{i=0}^{n} e_i \, \Delta t$$

based on rectangular rule.

Interpolation technique for derivative term

The first difference in the derivative term, $(e_n - e_{n-1})$ is affected by noise, and thus differentiation is sensitive to data error and noise. The noise can be reduced by using analog or digital filters.

However the technique commonly used is interpolation method with four-point control difference technique (Fig. 6.4).

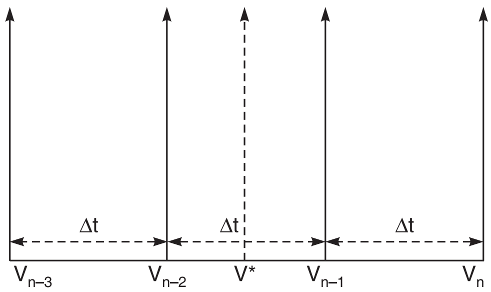


Figure 6.4 Interpolation technique for derivative term.

Let V_n, V_{n-1}, V_{n-2} and V_{n-3} be the values of controlled variable at current and their previous consecutive sampling intervals.

$$V^* = \frac{V_n + V_{n-1} + V_{n-2} + V_{n-3}}{4} \tag{4}$$

$$\frac{\Delta V}{\Delta t} = \frac{1}{4}\left[\frac{V_n - V^*}{1.5\,\Delta t} + \frac{V_{n-1} - V^*}{0.5\,\Delta t} + \frac{V^* - V_{n-2}}{0.5\,\Delta t} + \frac{V^* - V_{n-3}}{1.5\,\Delta t}\right]$$

$$= \frac{1}{6\Delta t}[V_n - V^* + 3V_{n-1} - 3V^* + 3V^* - 3V_{n-2} + V^* - V_{n-3}]$$

$$= \frac{1}{6\Delta t}[V_n + 3V_{n-1} - 3V_{n-2} - V_{n-3}] \tag{5}$$

Since set-point is constant,

$$\frac{\Delta e}{\Delta t} = \frac{\Delta V}{\Delta t} = \frac{1}{6\Delta t}[e_n + 3e_{n-1} - 3e_{n-2} - e_{n-3}] \tag{6}$$

Thus, with these two modifications the controller equation becomes

$$Y_n = KP \cdot e_n + \frac{KD}{6\Delta t}[e_n + 3e_{n-1} - 3e_{n-2} - e_{n-3}] + \frac{1}{KI}\sum_{i=0}^{n} \frac{e_i + e_{i-1}}{2} + Y_0 \tag{7}$$

In position algorithm computer recalculates the full value of the valve setting at each sampling interval. As shown in Fig 6.2 the analog signal is sent to valve actuator through DAC.

The position algorithm has distinct property that it maintains its own reference in Y_0. However it has two drawbacks namely lack of bumpless transfer from manual to auto switching, and reset wind-up due to integral saturation in test mode. These drawbacks are not present in velocity algorithm.

6.3.2 The Velocity Algorithm

In number of control loops, the final control element is stepper motor or stepper motor driven valve. In such cases, the requirement at the computer output will be a pulse train specifying the change in valve position. Thus output of position algorithm cannot be used, since it gives the new position of the valve, in absolute term.

In velocity algorithm, the computer calculates the required change in valve position. The output is digital pulse train which can be directly used in case valve is stepper motor driven. In case of other valves, stepper motor combined with slide wire arrangement as shown in Fig. 6.5 can be used. The same function can be performed by an integrating amplifier.

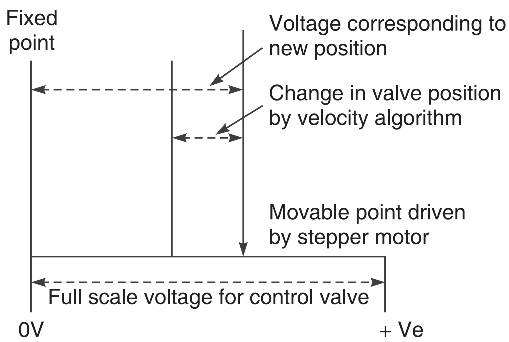


Figure 6.5 Slide wire arrangement.

The Eq. (2) of position algorithm derived earlier is,

$$Y_n = KP \cdot e_n + KD\left(\frac{e_n - e_{n-1}}{\Delta t}\right) + \frac{1}{KI}\sum_{i=0}^{n} e_i \,\Delta t + Y_0 \tag{8}$$

where, Y_n is valve position at t_n.

At t_{n-1} i.e. at previous instant, the valve position was,

$$Y_{n-1} = KP \cdot e_{n-1} + KD\left(\frac{e_{n-1} - e_{n-2}}{\Delta t}\right) + \frac{1}{KI}\sum_{i=0}^{n-1} e_i \, \Delta t + Y_0 \tag{9}$$

The change in valve position ΔY_n at t_n will thus be,

$$\Delta Y_n = Y_n - Y_{n-1}$$

$$= KP \cdot (e_n - e_{n-1}) + \frac{KD}{\Delta t}(e_n - 2e_{n-1} + e_{n-2}) + \frac{1}{KI} e_n \cdot \Delta t \tag{10}$$

The integral term and derivative term can be modified by using Trapezoidal rule and Interpolation technique similar to position algorithm

$$\text{Integral term} = \frac{1}{KI}\left[\sum_{i=0}^{n} \frac{e_i + e_{i-1}}{2} - \sum_{i=0}^{n-1} \frac{e_i + e_{i-1}}{2}\right]\Delta t$$

$$= \frac{1}{KI}\left[\frac{e_n + e_{n-1}}{2}\right]\Delta t \tag{11}$$

$$\text{Derivative term} = \frac{KD}{6\Delta t}\,[(e_n + 3e_{n-1} - 3e_{n-2} - e_{n-3}) - (e_{n-1} + 3e_{n-2} - 3e_{n-3} - e_{n-4})]$$

$$= \frac{KD}{6\Delta t}\,[e_n + 2e_{n-1} - 6e_{n-2} + 2e_{n-3} + e_{n-4}] \tag{12}$$

By substituting modified integral and differential terms in (10), we get

$$\Delta Y_n = KP \cdot (e_n - e_{n-1}) + \frac{KD}{6\Delta t}[e_n + 2e_{n-1} - 6e_{n-2} + 2e_{n-3} + e_{n-4}] + \frac{1}{KI}\left[\frac{e_n + e_{n-1}}{2}\right] \cdot \Delta t \tag{13}$$

The relationship between position and velocity algorithm is,

$$\Delta Y_n = Y_n - Y_{n-1}$$

i.e.

$$Y_n = \Delta Y_n + Y_{n-1}$$

$$= \Delta Y_n + [\Delta Y_{n-1} + Y_{n-2}]$$

$$= \Delta Y_n + \Delta Y_{n-1} + [\Delta Y_{n-2} + Y_{n-3}] = \ldots = \sum_{i=0}^{n} \Delta Y_i + Y_0$$

Logically also, the present valve position is equal to original position plus sum of all the changes occurred so far.

The velocity algoritum at equation (13) exhibits two measure problems:

— Controller drift
— Integral overshoot.

We shall discuss these and modify the algorithm accurately,

Controller drift

The velocity algorithm should always include integral term, otherwise it will give rise to controller drift. To explain this let us substitute the following equation in velocity algorithm

$$e_n = S - V_n,$$

where

S = set-point and
V_n = value of controlled variable at t_n.

$$\Delta Y_n = KP\,(V_n - V_{n-1}) + \frac{KD}{6\Delta t}\,(V_n + 2V_{n-1} - 6V_{n-2} + 2V_{n-3} + V_{n-4}) - \frac{\Delta t}{KI}\left[S - \left(\frac{V_n + V_{n-1}}{2}\right)\right]$$

From the above, it is clear that only integral term has set-point and thus this term will force controlled variable to come to set-point. If integral term is not present in velocity algorithm, then controller drift may be caused. The proportional term of velocity algorithm may give rise to oscillations. Let us consider the velocity algorithm with only proportional and integral terms:

$$\Delta Y_n = KP\,[e_n - e_{n-1}] + \frac{\Delta t}{KI}\left[\frac{e_n + e_{n-1}}{2}\right]$$

Case I

When error is increasing, i.e. when value of controlled variable is moving away from set-point.

$$\text{Proportional term} = KP\,(e_n - e_{n-1}) = +\text{Ve}$$

$$\text{Integral term} = \frac{\Delta t}{KI}\left[\frac{e_n + e_{n-1}}{2}\right] = +\text{Ve}$$

Thus,

$$\text{Output } \Delta Y_n = +\text{Ve}$$

Case II

When error is decreasing because of control action, i.e. when value of controlled variable is moving towards set-point.

$$\text{Proportional term} = KP\,[e_n - e_{n-1}] = -\text{ Ve}$$

since $e_n < e_{n-1}$

$$\text{Integral term} = \frac{\Delta t}{KI}\left[\frac{e_n + e_{n-1}}{2}\right] = +\text{Ve}$$

If integral term is not over balancing the proportion term then, Output ΔY_n = negative.

The negative correction may increase the error thus, giving positive correction as per Case I. The positive correction may decrease the error and give negative correction.

The controller response may thus oscillate. The oscillation problem explained above can be solved by disregarding the sign of proportional term and assigning it the same sign as integral term. Thus,

$$\text{Proportional tern} = (\text{sign of integral term}),\ KP\ |e_n - e_{n-1}|$$

Integral overshoot

This modification while solving one problem, creates another problem of integral overshoot and consequently integral oscillations. When proportional term is forced to have the same sign as integral term, the value of controlled variable will reach the set-point at a faster rate and overshoot it.

The integral term,

$$\frac{\Delta t}{KI}\left[\frac{e_n + e_{n-1}}{2}\right]$$

will register the change in the direction of error and oppose this along with proportional term and give negative correction. The value of controlled variable may overshoot the set-point in the opposite direction, giving rise to oscillation (Fig 6.6).

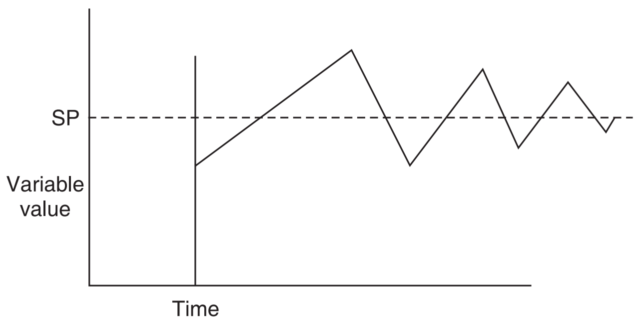


Figure 6.6 Case of oscillation in velocity algorithm.

The ideal solution problem will be to fix a band across the set-point. When value of controlled variable is outside the band the proportional term takes the sign of integral term. Thus controlled variable will reach the band steeply. Whereas when the value of controlled variable is inside the band, the proportional term takes the sign as calculated, i.e. $(e_n - e_{n-1})$. This gives damping action inside the band while controlled variable reaches set-point, thus preventing integral overshoot and oscillation (Fig. 6.7).

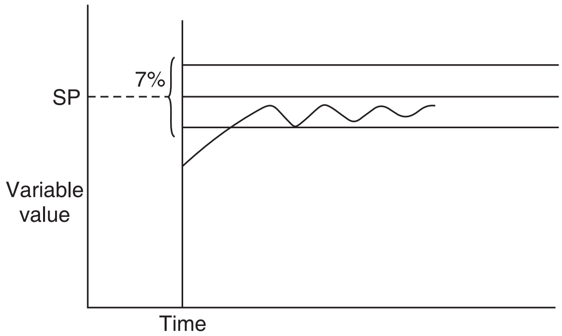


Figure 6.7 Effect of set-point band in velocity algorithm.

It has been found that 7% full scale set-point band gives good results on simulation tests of systems with first and second order time constants. The above modification to the velocity algorithm will yield the following equation.

$$\Delta Y_n = \frac{\Delta t}{KI}\left[\frac{e_n + e_{n-1}}{2}\right] + \frac{KD}{6\Delta t}[e_n + 2e_{n-1} - 6e_{n-2} + 2e_{n-3} + e_{n-4}] + \text{Proportional term}$$

(set-point Band) SB = 0.07 of full scale

When $|e_{n-1}| > \text{SB}$

Proportional term = (sign of integral term) $KP\,|e_n - e_{n-1}|$

When $|e_{n-1}| < \text{SB}$

Proportional term = $KP\,|e_n - e_{n-1}|$

The flow chart for velocity algorithm is shown in Fig. 6.8. The ADC output can be directly used for calculation if set-point and SB are expressed in terms of fraction of ADC output values. The range of ADC output is taken as full scale value. This avoids time consuming portion of Conversion to Engineering Unit in software. If proportionality constant KP, integral constant (1/*KI*) and derivative constant *KD* can be expressed as fraction of ADC value, then in addition it will save computer time in calculation.

6.3.3 Position vs. Velocity Algorithm

Reference position

The major advantage of position algorithm is the reference position. In its equation itself, the reference position of control valve is maintained as Y_0. This is however not there in velocity algorithm. Thus whenever there is disruption due to shut down, communication error/failure or any other reason, the median position of valve will be known in position algorithm and eventually valve will catch up without any synchronization problem. In velocity algorithm, external device like stepper motor or integrating amplifier should store the last position of valve.

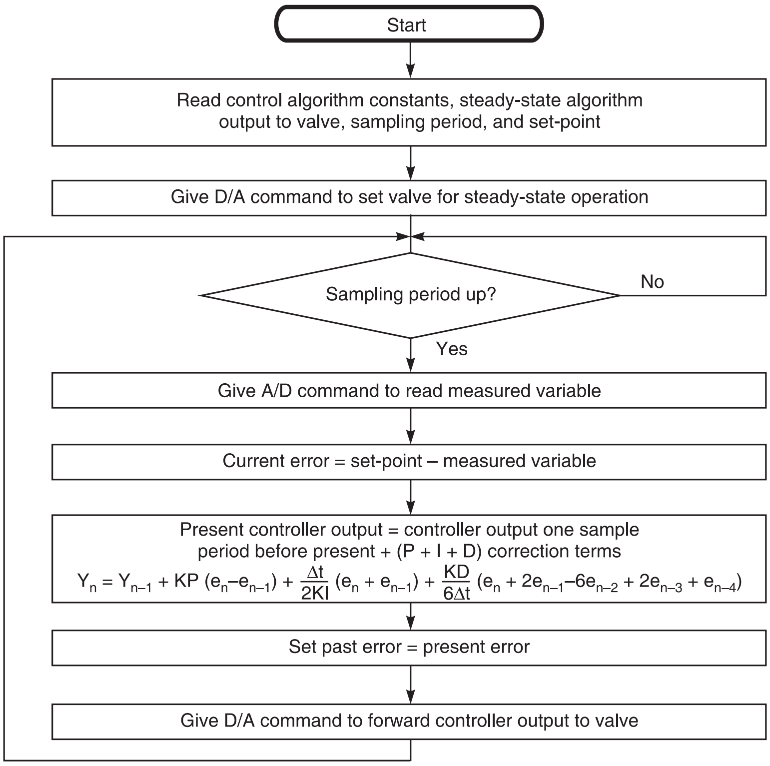


Figure 6.8 Program flow chart for velocity algorithm.

Reset wind up

When the DDC control loop is being tested, then sensor inputs are connected to see whether the DDC accepts the signals. But control valves in process are not connected, so that process is not disrupted. In position algorithm, the integral term increases to its limit (saturate or windup) because when the process is not connected, process reactions are same, i.e. sensors read more or less the same value and for computer error is persisting. In this case when the DDC system is finally connected, the correction drives control valve to an extreme, causing process disruption. This problem does not arise in velocity algorithm where the integral term is given by

$$\frac{\Delta t}{KI}\left[\frac{e_n + e_{n-1}}{2}\right]$$

A possible solution to above problem may be to disregard integral term in test mode. However, then the integral mode, cannot be tested along with other modes by connecting the system inputs.

Auto manual switching

The auto manual switching has been explained in Section 1.9. The DDC control loop with position algorithm causes bumps at the time of manual to auto switching. At the time of switching, the algorithm must be initiated to zero error situation. Thus output must be equal to

Y_0, i.e. median position of valve. Depending on the process dynamics, operator selects the median position of the valve and corresponding Y_0 value. The DDC algorithm has no knowledge of this precise Y_0 value, and thus switching causes bumps. Velocity algorithm is not susceptible to this problem since Y_0 value does not exist in the algorithm.

Thus velocity algorithm with corrections incorporated is best suited for DDC Control loop.

6.3.4 Cascade Control

The cascade control, has been discussed in Section 1.13.1. The primary controller senses the primary variable; and based on primary set-point, calculates the secondary set-point for the secondary controller. The secondary controller senses the secondary variables and performs control action through control valve on the process. The cascade control loop may be configured using velocity algorithm as shown in Fig. 6.9.

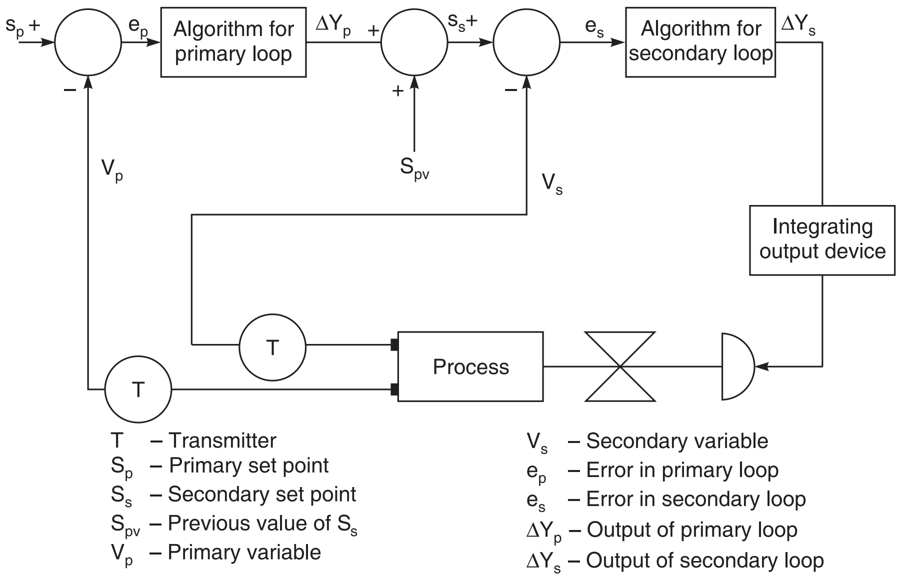


Figure 6.9 Cascade control algorithm structure.

Now in terms of DDC algorithm, the primary and secondary control loop algorithms can be combined as follows:

$$\Delta Y_p = F(V_p, S_p)$$

$$S_s = \Delta Y_p + S_{pv}$$

$$\Delta Y_s = F(V_s, S_s)$$

The equations for ΔY_p and ΔY_s are same as that for ΔY_n explained earlier. Thus in computer memory following needs to be stored.

Constants: S_p, S_{pv}

Variables: S_s, V_p, V_s, e_p, e_s, ΔY_p, ΔY_s

At the next instant S_{pv} takes the value of S_s. The DDC algorithm may first execute the primary loop. Then main program may calculate S_s and then DDC algorithm may again be executed for secondary loop.

6.3.5 Ratio Control

Basic elements of ratio control have been described in Section 1.13.2 and close examination of ratio control reveals its similarity with cascade control. The ratio control measures the value of primary and secondary variables, and in order to maintain a ratio between primary and secondary variables, it controls secondary variable.

Considering primary variables as V_p, it controls secondary variable as V_s and the ratio as R then ratio controller should maintain the value of secondary variable $V_s = R \cdot V_p$. Thus ratio control is very similar to cascade control.

Now if S_s = Secondary set-point, then,

$$S_s = R \cdot V_p \text{ or } S_s = S_{pv} + KP\,(\Delta V_p).$$

Thus, ratio control is cascade control with following changes:

— Primary loop has only proportional control

— Primary set-point = 0.

The value of S_s become S_{pv}, at next instant. The structure of ratio control algorithm is shown in Fig. 6.10.

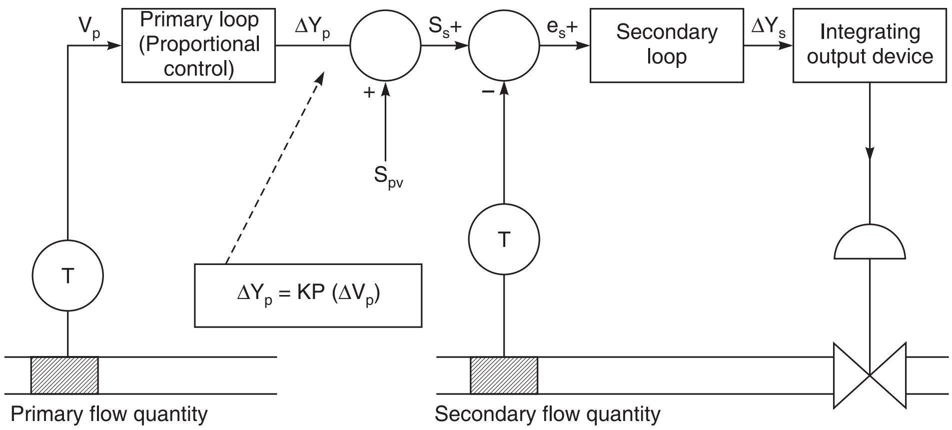


Figure 6.10 Algorithm structure of ratio control.

6.3.6 Multivariable Control

The same concept can be extended to multivariable control, where cascade can be in parallel or in series. The structure of multivariable control is shown in Figs. 6.11 and 6.12. The derivation of algorithm is straightforward for these structures. However in order to achieve meaningful control the DDC hardware should be very fast with multitasking environment. Otherwise, all the control loops may not be executed in time and process time requirements may not be fulfilled.

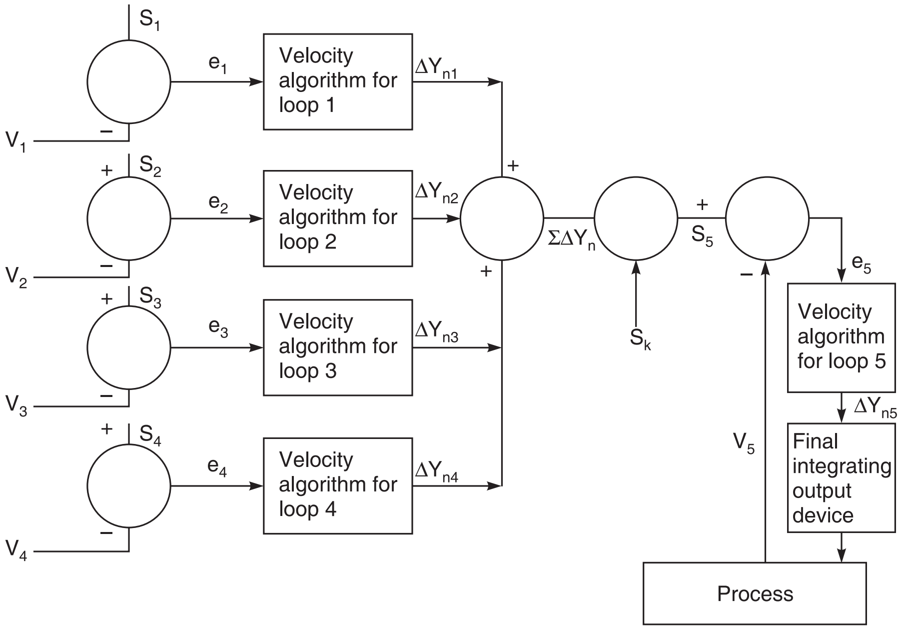


Figure 6.11 Multivariable control algorithm—Parallel structure.

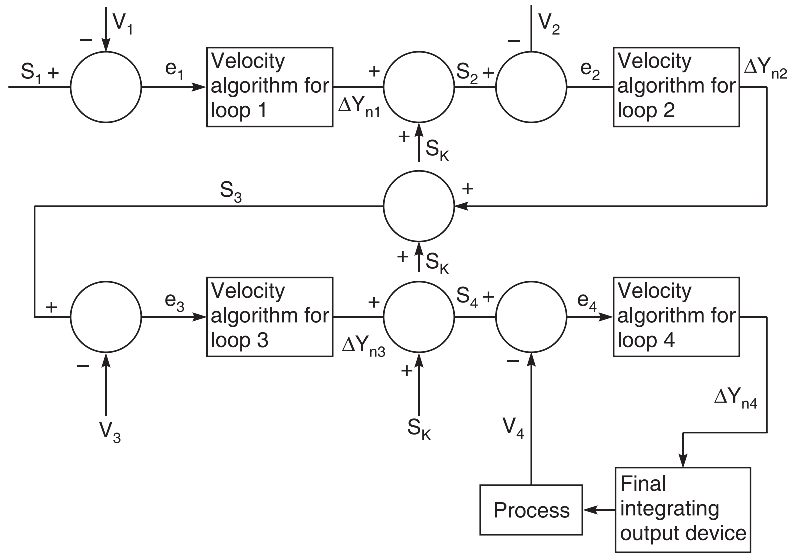


Figure 6.12 Multivariable control algorithm—Series structure.

In such cases DDC may he designed around parallel hardware using fast microcomputer chips like Intel 860, 80360, 80486 etc. or transputer chips. A parallel microcomputer based DDC architecture, suited for parallel multi-variable cascade loops is shown in Fig. 6.13. The common memory stores all the variables and is used for transferring parameters between various systems.

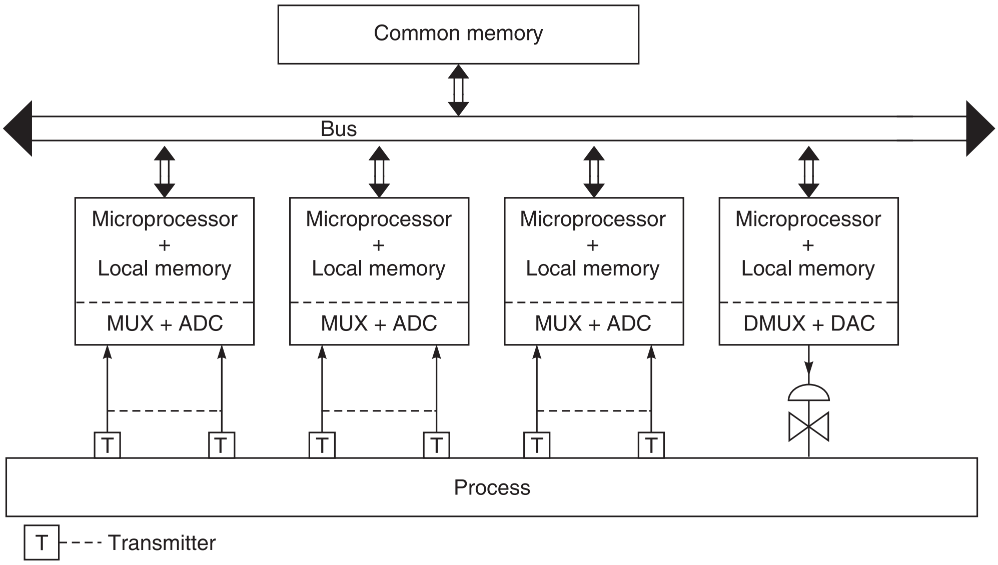


Figure 6.13 Multimicrocomputer based DDC structure.

6.3.7 Computer Instrumentation

Each simple loop and each loop of the cascade, ratio or multiple cascade systems can be treated exactly alike, if the digital description of a control loop is by the memory location of the following:

1. The input (ADC value)
2. The set-point (ADC value)
3. The proportional, integral, derivative gains (ADC values)
4. The output (address).

This loop description can be a great help to the instrumentation-design engineers, if each kind of loop component is tabulated separately according to the plant instrumentation numbering system. The instrument engineer would have only to specify the pertinent input variables which establish the setting of each final output element. A programmer could easily put these variables together (as described above) to get the desired control action.

6.3.8 Feed Forward Control

Feed forward control has been investigated widely in recent years. This method uses a priori intelligence about perturbation to anticipate and, if possible, to compensate for the effects of these upsets on process output. Better control results, by inserting a function into the control loop—that closely duplicates the process response between the upset and final control element. Corrections applied much earlier will result in too much anticipation of the system response resulting in over control and oscillation.

6.4 CONCLUSIONS

The analog controllers or direct digital controllers discussed here basically control single variable. That is, they measure the error occurring in one plant variable and attempt to correct it by applying a change to the operating level of *one and only one* plant control variable. One can think of the mathematical model for the control action taking place, as the configuration of the controller, i.e. (a) the choices of modes to use and (b) the values of the controller gains established for each mode by the process of tuning the controller (as listed earlier). Cascade and ratio control become more complex, for they require two input variables and the tuning of as many as six control modes.

Multivariable systems may be as simple as developing the tuning of a lead/lag network (two time constants). However, these may often require the actual establishment of a true mathematical model through plant tests and/or engineering analysis for their implementation. The more complex the system, the more necessary it is to develop the actual mathematical model and not its representation by tuning parameters.

Imposition of more advanced control systems, such as dynamic optimization requires a knowledge of the dynamic behaviour of the process in mathematical equation form and thus a mathematical model of the process. We shall be briefly dealing with mathematical models in Chapter 11.

SUGGESTED READING

Bristol, E.H., Design and programming control algorithms for DDC systems, *Contr. Engg.,* **24,** pp. 24–26, 1977.

Cox, J.B. et al., A practical spectrum of DDC chemical-process algorithms, *ISA J.,* **13,** pp. 65–72, 1966.

Dawies, W.D.T., Control algorithms for DDC, *Instr. Pract.,* **21,** No. 1, pp. 70–77, 1966.

De Bolt, R.R. and Powell, B.E., A natural 3-mode controller algorithm for DDC, *ISA J.,* pp. 43–47, 1966.

Gerry, J.P., A comparison of PID control algorithms, *Contr. Ens.,* **34,** pp. 102–105, 1987

Goff, K.W., A systematic approach to DDC design, *ISA J.* pp. 44–54, 1966.

Halme, A. and Ahave, O., Automatic tuning of PID and other simple regulators in digital process automation systems, *Proc. Amer. Cont. Conf.,* San Diego, California., pp. 74–78, 1984.

Ortega, R., Experimental evolution of four microprocessor-based advanced control algorithms, microprocessing/microprogramming. **10,** No. 4, pp. 229–245, 1982.

Popovic, D. et al., Conceptual design and C-implementation of a microcomputer-based programmable multiloop controller, *J. of Microcomp. Appl.,* **12,** No. 2, pp. 159–165, 1989.

CHAPTER

7

Distributed Digital Control

7.1 INTRODUCTION

Modem automation problems in the industry which need to be solved by using a process control computer, range from monitoring, supervision and control of a small part of a production plant, to the integral control and management of a large plant. This, in consequence implies need of different approaches for design of hardware and software of the automation system.

The availability of powerful microcomputers has made it possible to implement computer control systems which do not require the facilities of a large, powerful mainframe computer system, with expensive peripherals. Such distributed processing systems are suitable for use in the process control as well as the general business applications environment. It is also possible to combine process control systems with manufacturing management and business computer systems. In such systems, the computing power is available at the point of use. Instead of using a large monolithic mainframe computer for performing a large number of unrelated tasks, a number of microcomputers are used to carry out the required tasks.

In the case of a process control system, the limit is reached if a separate microcomputer is used for controlling each individual process loop, as was the case with conventional analog controllers, or for controlling a small number of loops close to the process. However, this does not imply that each microcomputer is used in a stand-alone fashion. Efficient use of microcomputers and their associated data demands that such microcomputers should be linked together or to a host system at a higher level. In a large and complex system, there might be a number of levels at which different types of computer systems are used depending upon the types of functions to be performed. The type of configuration used will depend upon the characteristics of the production environment in which one is operating as well as the requirements of the overall control system. It is most likely that future computer systems, in process control as well as general business applications, will make an ever-increasing use of distributed processing concepts, simply because small computer systems are produced in very large quantities and are relatively inexpensive. The computing power provided by linking together such inexpensive computers is much greater than that available from a mainframe computer which costs almost equally.

7.2 HISTORY

Automation equipment has increasingly been used to monitor and control the industrial plants, in particular Chemical and Petrochemical, Fertilizer, Steel, Power, Refineries and so on.

In the earlier years, the individual computers had been attached to the different parts of the plants to be automated, which has led to an assembly of distributed, mutually independent and dedicated small computers (Fig. 7.1) These dedicated computers have wider applications in the fields of

— experiment automation in the laboratory, test field etc.;
— small plant automation;
— partial plant automation;
— signal and image processing.

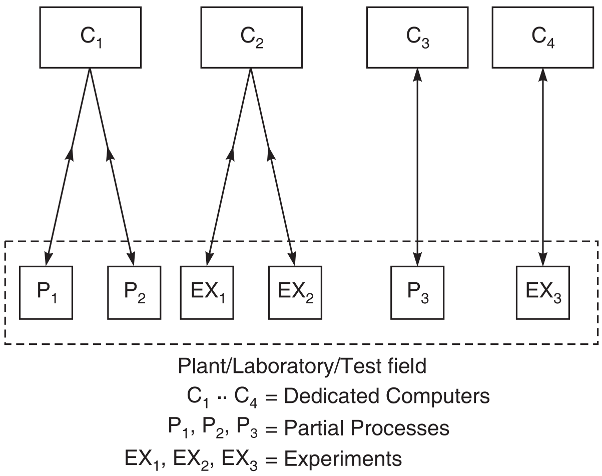


Figure 7.1 Dedicated computers concept.

This was the state of decentralised computer-based automation of the 60's However, due to the fact that even the small dedicated computers were relatively costly, a centralised, single-computer automation structure was introduced, containing a middle-scale or large-scale process control computer as its central part (Fig. 7.2). Following essential functions were concentrated in the computer itself:

— Process monitoring
— Data acquisition
— Alarming and logging
— Data processing
— Data archiving
— Process control.

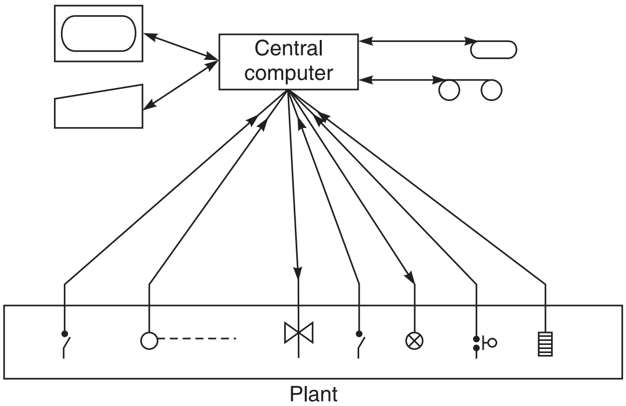


Figure 7.2 Centralised computer control concept.

Sometimes, some of the production planning and plant management functions were also added, so that the central computer truly became a concentrator of most important process and plant data. These were the typical function of a computer-oriented automation system up to the middle 70's.

7.2.1 Distributed vs. Centralised Control

Most of the computer control systems implemented during the sixties and seventies were centralised, whereby a single large computer was used to acquire data from one or a number of processes and control a large number of process loops. Computers and their associated peripherals were relatively expensive, and it was therefore, necessary to utilise the available computing power efficiently.

The cost of interfacing the computer to one or more processes was relatively high as long and costly screened cables were needed, which were necessary in a typical production environment. In such centralised computer process control systems, it was necessary to bring all the cables back to the central control room.

This approach clearly has its own advantages and disadvantages. On the positive side, it is possible to achieve overall co-ordination and optimisation of the process. Large amounts of process data can be stored on associated peripherals, such as hard discs and subsequently analysed to improve the process performance. It is necessary to use only one set of expensive peripherals such as discs, printers and plotters.

Process optimisation can only be carried out by analysing earlier data, developing/validating process models and continuously updating them. Data from different sections of a process are required to solve relatively complex equations for inferring the values of variables or for determining the changes to input process variables which can be manipulated to counteract the effect of process disturbances.

Large computers, with multiprogramming facilities, or even minicomputers operating in foreground-background mode can carry out these necessary calculations while concurrently performing the input/output functions necessary in computer process control systems. Another significant advantage has been the availability of some packages, with extensive facilities, for computer control of different processes. Many mainframe computer system manufacturers provided such computer software packages in order to promote the sale of computer hardware suitable for process control. Considerable progress was made in the field of computer process control systems using such centralised computer systems.

While offering the benefits outlined above, the approach of using centralised computers for process control purposes suffers from a number of disadvantages also. First and foremost is the reliability. Any computer used for controlling processes must be highly reliable, since any system breakdown will result in disruption of production and complete shutdown of the process, which could prove to be extremely expensive. A highly reliable system can only be developed by building a considerable redundancy element into it. One possible approach is to have a standby computer system, ordinarily used for development work, and use it for process control purposes in the event of the breakdown of the main unit.

The cost of a centralised computer process control system is obviously an important consideration. Mainframe and large minicomputers used for centralised process control are still relatively expensive. Such computers are produced in small quantities and hence have to carry large overheads. The provision of standby computer to be used in the event of the breakdown of main unit can be a costly exercise; cabling and interfacing costs are also high. A number of terminals might be required to keep shop floor process operators and supervisors informed about the state of the process. Computing power is not available at the point of actual use. Furthermore, it is difficult to ensure integrity of the process database.

A serious objection for introduction of centralised computer systems was from the very beginning, the computer itself. Not only its computational speed could get critical, but more than that, its reliability was not as high as required for plant control. This was primarily due to that fact that the computer failure could have catastrophic consequences for the plant itself.

Furthermore, typical for this system configuration is the fact, that the existing plant instrumentation (e.g. the sensors, transmitters, actuators, keyboards, indicators, recorders, etc.) has to be connected to the computer so that hundreds and thousands of connections should be implemented between the plant instrumentation and the computer. That this was not a serious obstacle to introduce the centralised computer systems for automation of running ("old") plants, is due to the fact that the connections of such kind have already been implemented for the purpose of manual, centralised plant automation by using the conventional controllers. However, this was not easily accepted when planning the automation of the new plant, since the wiring and other installation costs can override the cost of the computer system itself. Thus, a new automation structure was searched for, which would be (a) more reliable, (b) easier and less costly to install, and (c) more transparent with respect to the data structure and automation functions.

This has created the logic philosophy of future systems development. The data should be stored where they have been created and where they will be needed. Similar principles hold for the functions also. Only the data required for "other purposes" within the automation system will be distributed. For instance, data and functions required for

— local control and supervision of plant should be located next to the plant instrumentation;

— "higher" purposes (optimisation, set-point value calculation etc.) should be allocated near the plant operator;

— production planning and plant management should be situated closer to the relevant plant staff.

As a consequence, hierarchical automation system structure was introduced, containing also some elements of a distributed computer system, popularly known as Decentralised Computer System (Fig. 7.3). The system was widely used in the 70's and is still in use now.

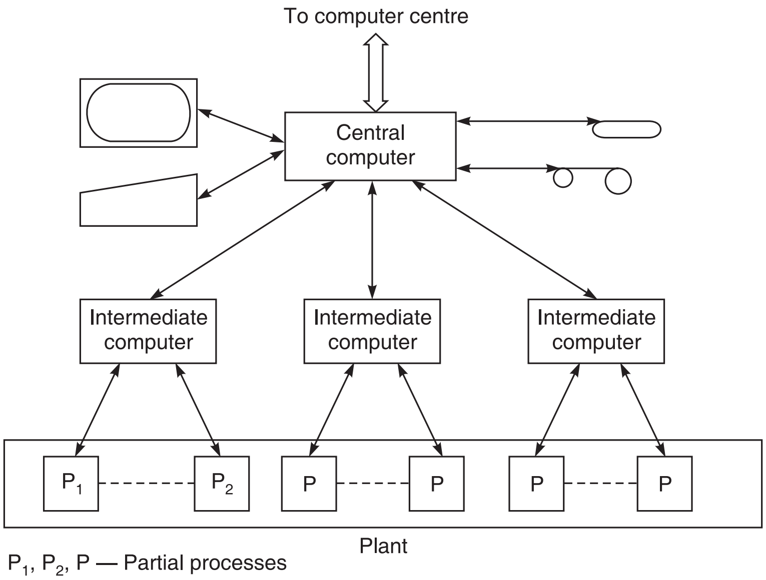


Figure 7.3 Decentralised computer control concept.

The decentralised computer system, although very sophisticated and transparent, can only be considered as a transitional concept for the real distributed computer systems for plant control. The main reason for transition from decentralized to the totally distributed automation concept lies in the rather inherent implementation difficulties of a decentralised system. These may be due to (a) hardware problems (mainly interfacing problems), or (b) software problems (compatibility and transportability problems).

7.2.2 Advantages of Distributed Control Systems

Following are the main advantages of distributed control system over centralised control system.

1. Provided that the overall system structure has been carefully designed, the system implementation can be carried out in a modular fashion thereby providing incremental system growth.

2. The cost of distributed system is lower than the cost of a centralised system which performs the same functions. Small computers are relatively inexpensive and, when linked together, they provide more computing power than a large mainframe system of the same cost. The cost of microcomputers is still lower and the actual cost of implementing a module at a later stage may further come down.
3. The cost of upgrading the system will also be lower. Replacement of a large centralized mainframe computer system is a very expensive exercise. Small, relatively inexpensive systems can be easily upgraded, if necessary, without incurring heavy expense. If the centralised computer system has reached the limit of its processing capability, it may not be possible to accommodate extra functions without incurring heavy expenditure or deterioration of system response time. This can prove critical in process control systems.
4. Each processor carries out a clearly defined set of functions and system growth could be achieved by incorporating additional processors. The tasks to be carried out by individual processors are relatively simple. Complex operating systems, with sophisticated multi-programming, time-sharing and real-time interrupt facilities, may not be necessary at the lower levels. Application programs are easy to develop, since they have to only perform a limited number of functions. The application software required for handling a large number of real-time functions is very complex.
5. A distributed processing system is inherently more reliable than a centralised processing system. Each small system has a small number of components and hence is more reliable. It is easy to provide back-up for small expensive micro and minicomputers used for real-time functions. The probability of the simultaneous failure of a large number of individual processors is extremely low.
6. There is substantial reduction in the cost of interfacing the process to the computer system. It is no longer necessary to use large lengths of screened cable to feed the sensor data to the centralised computer system. The first level microcomputers or micro-controllers interfaced to the process will be in the immediate vicinity of the process, requiring short cable lengths. A number of sensors might be interfaced to one first level microcomputer.
7. Distributed processing systems are more flexible than a centralised processing system. The current configuration can be easily altered, within limits, to conform to the changed requirements. Microcomputers with firmware programmed to carry out specified discrete functions are available in the market.
8. Distributed processing systems allow the duplicate storage, if necessary, of critical data. For example, critical data relating to process conditions might be stored on the floppy discs attached to a supervisory micro/rninicomputer, as well as on hard discs attached to a production control computer system.
9. The management of computer operations can be simplified. Level 3 and 2 systems can have control over the computers used in their areas, while at the same time can provide a link to the data used and created in other areas. This is perhaps the main advantage of distributed data processing, compared to decentralised processing in which stand-alone and independent processors do not communicate with each other.
10. In a process control environment, the use of microcomputers/controllers, placed in the immediate vicinity of the process, results in a minimisation of data losses and errors. In

a centralized system, process data can be lost over long distances or errors can be introduced. Considerable system overhead is required to recover the lost data.

7.3 FUNCTIONAL REQUIREMENTS OF (DISTRIBUTED) PROCESS CONTROL SYSTEM

The basic functional requirements of a distributed process control system are that it should—

1. have a consistent and uniform system approach;
2. fulfill and perform all operational, process and plant control functions;
3. automatically control the process and plant in normal operation within the specified limits and tolerances but also permit manual operation;
4. provide at any time, the operating personnel with comprehensive information on the status of plant and process for control and maintenance purposes, fault detection and localisation;
5. permit the manipulation without specialised programming knowledge of parameters and process control functions, as may be required or desirable from time to time;
6. be so designed that future expansions can be easily and economically implemented;
7. enable highly economical plant operation.

The architecture or configuration of a system is one of the prime factors for its reliable and effective operation. In any plant, following persons interact directly with the systems:

1. Plant operator
2. Maintenance Engineer
3. Design and Development Engineer
4. Manager and Supervisor

Each of these persons visualises the system from his working point of view and thus has some requirements towards the system configuration. While selecting the configuration, these requirements must be taken into consideration.

7.3.1 Plant Operator's Requirements

Majority of the plant operators are using conventional type of controllers, alarm annunciators and data acquisition procedures. Thus they are attuned to a particular way of operation. In adopting new system also, they would like to be provided with the facilities which may not be necessary. Similarly, there are various other facilities they would like to have to reduce their mental and physical strain. Following are some of the important points based on the operators' requirements, which should be considered while selecting the system architecture.

— Easy handling of the system.
— Auto/Manual station provision.
— Easy process accessibility.
— Instantaneous access to any area of the process.
— Provision for simultaneous view of two or more subprocesses.

— Continuous monitoring of the process dynamics,
— Provision for accurate and reliable data.
— High speed graphics facility with updated dynamic data.
— Provision for tuning a control algorithm parameter which is physically not accessible to the operator.
— Demand printing of a log which may physically reside somewhere else.
— Provision for modification of a system log.
— Extensive alarm monitoring and status reporting on colour CRT.
— Quick response of abnormal conditions and convenient suppression of unwanted alarms.
— Provision for knowing the quality of a point.
— Quick isolation of an area in trouble.
— A separate display for alarms (if desired).
— Sequence of event recording.
— Hard copy facility of alarms and status of inputs.
— Historical records accessibility.
— As far as possible, single push button/key actions.
— Less crowding of equipment in operator's room.

7.3.2 Maintenance Engineer's Requirements

Normally, in a plant, a maintenance engineer has to undertake maintenance of a large variety of instruments. For this purpose different types of spares are to be stored and handled. Trouble shooting and fault repairing is often time-consuming. Repair generally requires shutdown. Therefore, while selecting distributed process control system, care should be taken to minimize the burden of maintenance engineer. Following points may be considered for this purpose:

— Modular design.
— Use of identical hardware components in various sub-systems, as far as possible.
— Self diagnostics facility.
— Provision for repairs without shutdown of complete process.
— Higher order of redundancy.
— Easy availability of spare parts and components.
— High MTBF—Mean Time Between Failures.
— Low MTTR—Mean Time To Repair.

7.3.3 Design/Development Engineer's Requirements

This engineer is the key figure in running the system and modifying the system behaviour as the need arises. In addition to the hardware design aspect, he undertakes software alterations also. These may be to change the control strategy and thereby build custom control algorithms, add new symbols and shapes in the library and so on. He should be able to access the database and be able to take over as an operator also. Some of the very urgent requirements of a design and development engineer (in addition no operator's requirements) are given below:

— Provision of direct interaction with the process.
— Provision for using modulating control, sequential control or only data acquisition.
— Facility for building custom control algorithms and adding them to the algorithm library.
— Provision for building display diagrams from either a hand-drawn sketch directly on a CRT screen or in a simple high level problem-oriented language.
— Option to building often used display diagram shapes and adding these to the system shape library for their convenient use in building of diagrams.
— Availability of custom defined keys on the keyboard.
— Facility of PROM burner.
— Provision for developing and integrating custom application programs separately.
— Possibility of testing application programs with real-time data.
— Facility for building the system logs using simple problem-oriented language.
— Provision for initiating, implementing and activating the functioning of various sub-systems.
— Facility for retrieving and analysing the historical data.
— On-line expansion of the system.

7.3.4 Manager/Supervisor's Requirements

Management, in general, as more concerned about the system behaviour for the proper operation of the plant on the whole. A large amount of data and reports at present are required to evaluate the plant operation. An effective information system is therefore required. Similarly, there are various other requirements which the management would like to consider. Following are some of the important points to be taken into consideration:

— Provision of overall plant operation display with facility of selecting a particular area of the plant.
— Continuous awareness of process dynamics.
— Provision for accurate and reliable data,
— Facility for sending and receiving error-free messages.
— Availability of different types of logs.
— Trip analysis provision.
— Availability of historical data and other reports.
— Quick reporting of abnormal conditions.
— Easy system operation.
— Easy and fast maintenance.

— High reliability of the system.
— Easy expandability.
— Possibility of backup of vital functions.
— Training of plant personnel in minimum period.
— Fast and error-free communication with other units of the plant.
— Fast response time of the system even with worst loading conditions.
— Facility for using the system for development purpose also.
— Isolation of faulty sub-system without degrading the performance of the remaining system.
— Cost effective system.

7.3.5 Distributed Control Systems Evolution

The development in the area of computer-based plant automation in the 70's was strongly influenced by standardisation efforts in control and computer engineering. During this period, hardware as well as software standards were worked out which have extended the compatibility of equipment and the portability of programs.

Interfacing of computers has been an essential outcome of this development: back-plane, short- and long-distance buses including the Local Area Networks (LAN) have been standardised and accepted as a versatile solution for design of complex hierarchical and distributed, computer-based automation systems.

In addition to this, the modularity concept in hardware and software design has been accepted as a most suitable one for saving time and development costs.

Furthermore, the standardisation of application software for plane control and signal processing have created some software packages which have been designed to be universal, flexible and easy to apply. These software packages have the following common software functions:

— Input signal conditioning
— Validity check
— Engineering unit conversion
— Linearisation
— Digital filtering and smoothing
— Averaging and extrapolation
— Peak-value search and tracing
— Pulse and digital signal evaluation and processing
— Trend check and monitoring
— Open and closed-loop control
— Logging

These are realised as functional blocks and inserted into the software system which is called "Library of Functions".

On the other side, important technological revolution took place in the field of semiconductor electronics. As a result, the capability of integrated circuit chips augmented and at the same time their price also decreased. The designer was given the possibility to design its system as sophisticated as possible and to implement its software as transparent as possible by using higher-level real-time languages that eventually saved his time and reduced the developing costs of software.

If considered together, the individual trends mentioned above essentially contributed to the development of new automation concept, and to the creation of new design philosophy of computer-based automation systems. The outstanding characteristics of these systems are not only the hierarchy and distributivity but also the outstanding flexibility of their configuration and their online re-configuration.

Examples of such new automation systems are numerous. In the meantime, there are over 50 modem Distributed Computer Control Systems in the market today, such as,

— CP-80 (AEG-Telefunken)
— DCI 4000 (Fisher and Porter)
— MAX 1 (Leads and Northup)
— MICON (VDO)
— PROCONTROL (Brown Boveri)
— MOD III (Taylor and ABB)
— MV 8000 (Beckmann)
— PCS 8000 (Philips)
— PLS 80 (J.C. Echardt)
— P 4000 (Kent)
— PMS (Ferranti)
— PRO VOX (Fisher Controls)
— TDC 3000 (Honeywell)
— TELEPERM-M (Siemens)
— TOSDIC (Toshiba)
— TPS system (Honeywell)
— LonWorks system (Echelon), and so on.

These systems have been introduced only a few years after the first distributed computer control system (Honeywell's TDC 2000-system) has been installed.

7.4 SYSTEM ARCHITECTURE

The distributed process control system should have the following characteristics. The system should

— be of functional and topographical distributed design and have a hierarchical structure;
— distribute the functions among devices and stations in accordance with the hierarchy and modularity of the automation tasks and suit the individual requirements involved;

— have devices which perform all the automation functions such as signal conditioning, logic and regulatory control, monitoring, protection and optimisation as required in each plant area and which can be combined to form complete stations;
— be equipped with data hiways (Bus System) ensuring a comprehensive data exchange within and between the individual stations of the plant;
— be so designed that the data traffic takes place directly between the involved controllers and input/output devices, and must not be directed through intermediate stations;
— permit the use of remote input/output stations for front-end signal input and output in order to minimise cabling expenditure;
— make the signals transmitted to controlled drives and actuators directly available at the respective drive controller for the purpose of autonomous manual control;
— have central operator stations for comprehensive process monitoring and interaction;
— permit the incorporation of remote operator stations for monitoring and interacting with the plant areas;
— log operator interventions, important messages and alarms;
— ride through power outages without loss of stored process data;
— allow the setup and alteration at any time of operating hierarchies for different operating locations.

The control functions should principally be implemented in multi-purpose processing stations. The modular design of these stations should be such that their processing capability and storage capacity can be expanded in steps by plug-in modules to suit the following requirements:

— In the case of a data hiway failure, the stations should continue to operate autonomously.
— In the case of a station failure, the functions of the other stations and the data hiways should not be impaired.
— Disturbances in a station sub-assembly should not affect the correct performance of other devices.
— The failure of individual components in a station should be detected and diagnosed. Signals affected by a component failure should be automatically provided with error flags.
— In the case of a component failure in a loop, only the affected loop is disabled (single-loop principle).
— An interface to the drive and the manual control board, integrated on the individual processor card, enables autonomous manual emergency operation.
— In the case of failure of individual components, the basic functions of the station, e.g. Internal Signal Exchange and Power Supply are not impaired.

The data exchange within and between the individual multi-purpose processing stations should not exceed an average transfer time. Furthermore, the data should be available, within this transfer time, to the entire plant for monitoring, interaction and further processing in process computers or other stations.

The multi-purpose processing stations shall have provisions for plug-in peripheral devices for all required input/output signals. The peripherals should meet the following requirements:

— Input signal conditioning for direct reprocessing (range adjustment, linerisation of temperature characteristics, flow rate correction).
— Electrical isolation or independence from process disturbances.
— Reasonableness check by monitoring of the measurement range.
— Monitoring of limit values of analog input signals; the limit values should be adjustable via the data hiways.
— Signal filtering, including high grade transient power frequency voltage suppression.

The important characteristic of a Distributed Control System is its layered structure. Each layer corresponds to the group of functions to be performed on the "lower" layer on getting some instructions for that from the "higher" layer. Although it is possible to distinguish a large number of functional layers, we shall explain here the organisational principles of a distributed control on the examples of a 4-layer system. (Fig. 7.4)

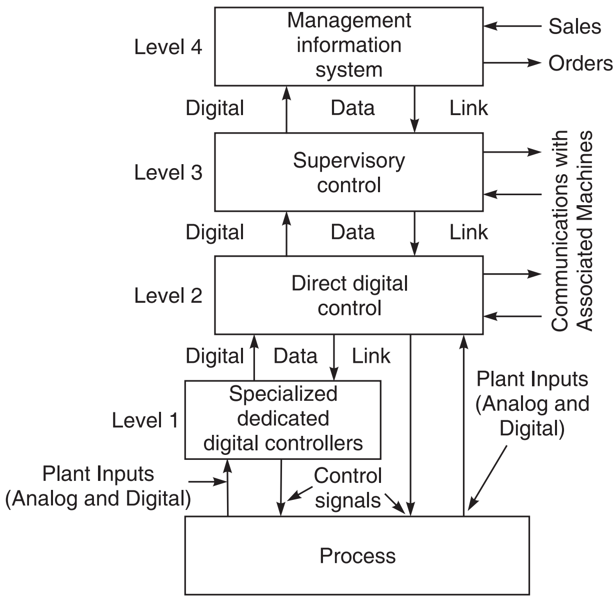


Figure 7.4 Hierarchy for distributed control system.

Level 1—Field level

Here, the First Level (LEVEL 1) is the Dedicated Digital Control layer, containing the lowest-level functions, like: detection of any emergency conditions and shut down; and to provide manual back up if all other layers fail.

Level 2—Area control level

The Second Level (LEVEL *2)* is Direct Digital Control Layer. The functions of this layer are: (a) to process data (temperature, flow, pressure, etc.) collection, message and alarm signals processing; (b) open (sequential) and closed-loop control; and (c) hardware (process-interface) test.

Level 3—Plant control level

The Third Level (LEVEL 3) comes above the DDC layer, and contains functions like optimal control, adaptive control, and self-tuning control.

Level 4—Plant management level

Finally, the Fourth Level (LEVEL 4) contains the production planning, and resource scheduling and other management functions which are related to the process or plant management.

Corresponding to this, different computers will be attached to different layers, generally dedicated microprocessor-based system to Level 1, microcomputers to the Level 2, medium-scale computers to the Level 3 and a large-scale computer to the Level 4.

A word of caution should be mentioned at this juncture. Each manufacturer of distributed control system has given name of his choice to systems at different levels in the hierarchy. Thus, the reader is advised to look carefully into the architecture. This will be clear when we describe distributed control systems of AEG Telefunken in next section and other distributed control systems in subsequent sections.

Due to advancement in hardware technology (micro-miniaturisation) as well as large numbers of specific requirements, there is tendency towards merging of level 1 and level 2 as well as level 2 and level 3 systems. Thus, current DDC systems at level 2 take care of functions at level 1. Similarly, supervisory control systems which cater to DDC functions are also available as current market trend. Some of the control system theoreticians observe this trend as going towards centralized control.

Now, we shall describe the structure of Distributed Control Systems through an example.

7.5 DISTRIBUTED CONTROL SYSTEMS

In Fig. 7.5, a typical Distributed Control System is represented, based on AEG-Telefunken Logistat CP80 Automation structure for industrial control.

It is obvious, that the system structure is the hierarchical one, divided into the fallowing hierarchical levels.

Level 1—Process Level

Level 2—Unit Control Level

Level 3—Group Control Level

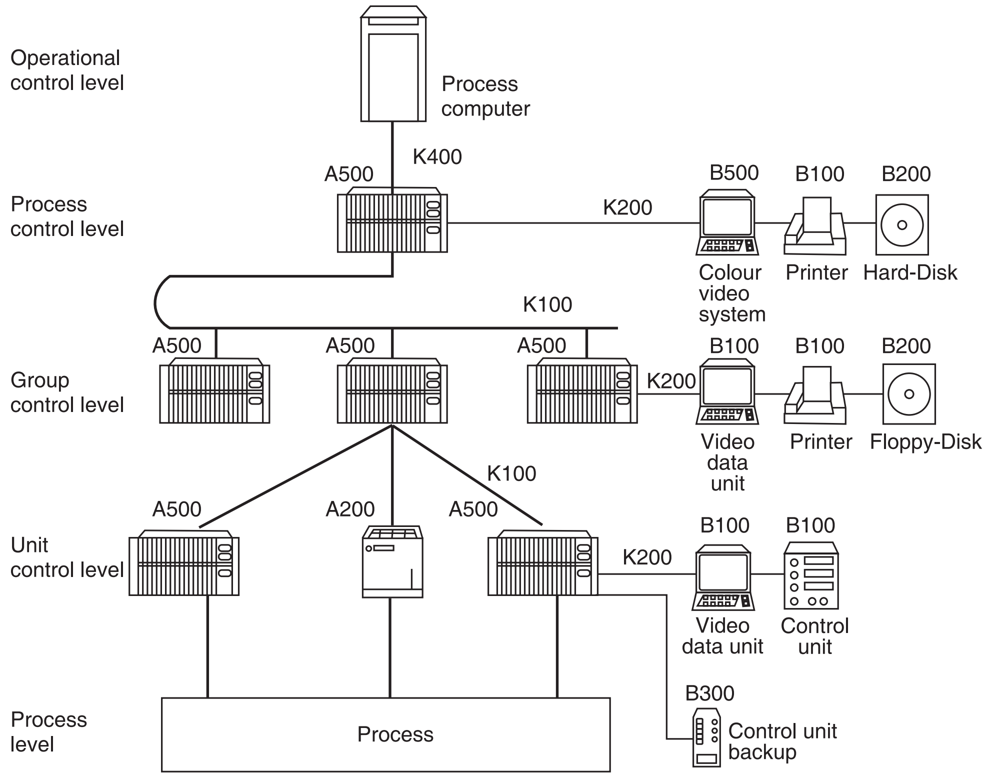
Level 4—Process Control Level

Level 5—Operation Control Level

This example will clearly show that distribution of functions at various levels may be different for different manufacturers.

Process level

It contains the functions, typical for the conventional plant instrumentation for interlocking, alarming, manual back-up control etc. This is in fact not a pertinent part of a Distributed Control System but an essential interface to it.



Level	Functions during operation/display	Automation
Process control level	Central off-normal recording and analysis, optical and acoustical process display with graphs, curves, tables, management information	Balancing Optimizing Planning Monitoring
Group control level	Remote control Remote parameterizing Status logs Fault sequence logs Charge logs Process data filling Program data filling	Open-loop control Arithmetic, Closed-loop control Text processing Administration
Unit control level	Control intervention Parameterizing Visual signal displays	Open-loop control Arithmetic Closed-loop control Signalling Monitoring Measured value processing Positioning

A. –Automation devices
B. – Operating and presentation devices
K. – Coupling systems

Figure 7.5 Distributed computer control system: An example.

Unit control level

It contains the main functions for local

— data collection and supervision;
— data check and alarming;
— data logging;
— open and closed loop control etc.

Here, individual sensors, actuators, indicators and control loops will be handled, which belong to a clearly defined, relatively independent plant part. In addition to this, some monitoring and supervision facilities will be provided for the local or field plant operator. Due to this reason the computer unit belonging to a limited number of measuring points (up to 32) and of control loops (8 to 16) is called the Field Control Station.

Group control level

Where the individual Field Control Stations are grouped and attached to the relevant Group Control Stations, belonging to a greater well-defined part of a large-scale plant or to a completed small-scale plant. This system level is really a non-obligatory one, but still it could be very useful when automating large industrial plants. It is this aspect of systems' reliability and availability that makes the introduction of this automation level advisable. The level contains software necessary for, (a) partial process models, (b) local process optimisation, and (c) regional plant monitoring etc.

Process control level

This is the highest control level as far as process itself is concerned, not including the production planning and plant management.

At this level, the central software will be stored, which is necessary for complete, (a) plant supervision, (b) archiving and logging of plant data, (c) optimal plant control based on mathematical process model and/or objectives of the plant control, and (d) central plant monitoring, alarming, etc.

Operational control level

The computer at the highest level performs production scheduling, order booking, etc. The software at this level includes,

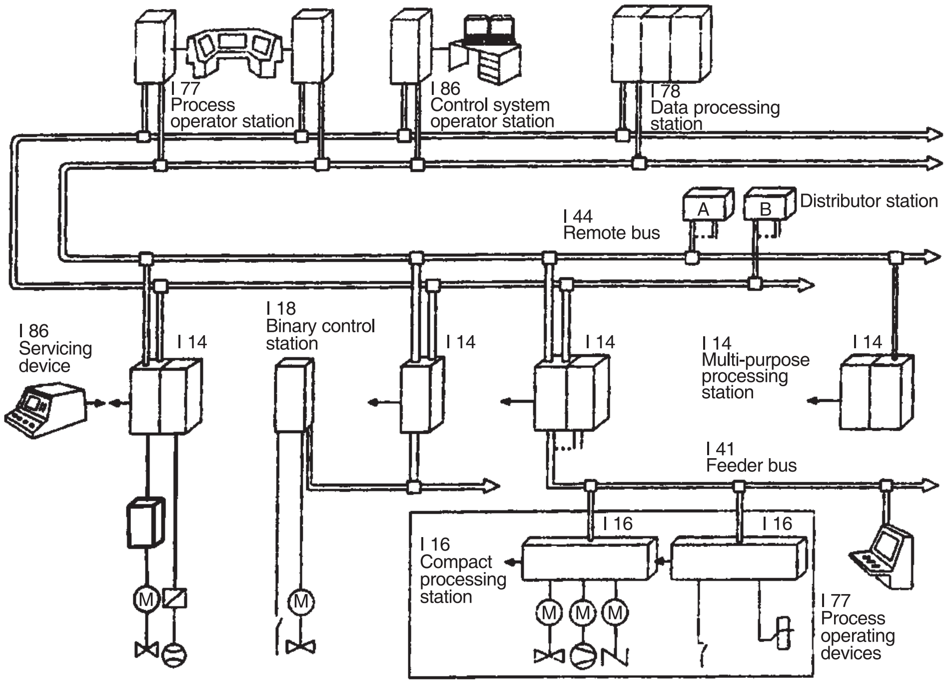
— Production Planning
— Plant Management
— Order Booking
— Personnel Management
— Machine Management

The unit control levels of the automation system are interconnected with the appropriate data communication paths mutually.

To each level, a number of different monitoring and hard-copy documentation equipments will be attached.

7.5.1 Distributed Control Sub-Systems

The distributed control systems follow bus-based architecture. The bus-based architecture for distributed control systems is shown in Fig. 7.6. The structure shown belongs to BBC PROCONTROL system. Other systems follow the similar structure.



Procontrol I consists mainly of the following system components:
Procontrol I 14 – Multi-purpose processing station
Procontrol I 16 – Compact processing station
Procontrol I 18 – Binary control station
Procontrol I 41 – Feeder bus
Procontrol I 44 – Remote bus
Procontrol I 77 – Process operator station, process operating devices
Procontrol I 78 – Data processing station
Procontrol I 86 – Control system operator station, servicing devices

Figure 7.6 PROCONTROL system.

The main sub-systems of Distributed Control System are:

— Field stations
— Presentation and monitoring devices
— Communication units.

There are a variety of options on technology, and capability of these units. In the following section we shall briefly present these sub-systems which form the heart of distributed control systems.

7.5.2 Local Field Station

The smallest intelligent unit of a distributed control system is the multi-loop control system situated in the field and attached to a smaller group of local controllers. Different manufacturers have designed systems considering their visualisation and assigned different names to their subsystem. The Honeywell TDC 3000's field stations is known as "Controller File." The Siemens TELEPERM field stations are called 'Automation Sub-System' and so on. The station (as a module of the system) usually contains:

1. A micro computer system with
   — 32 to 64 KB RAM;
   — 16 to 32 analog inputs;
   — 8 to 16 analog output;
   — up to 48 digital inputs and outputs;
   — a serial and a parallel bus interface.
2. Hard copy output terminal.
3. Local display terminal.

Figure 7.7 shows the block diagram of local field station of BBC PROCONTROL system. Here, the field station is called 'Compact Processing Station'.

The station is completely independent, as regards the control functions attached to it. Only the set-point values can externally be input via a keyboard or transferred via a bus. Usually, 8 to 16 closed-loop controllers can be implemented in a system along with some open-loop controllers and some monitoring paths for process variables.

The hardware of the field station is designed to be highly reliable. For this purpose, twin or multicomputer configurations as well as a doubled bus and other (analog and digital) inputs and outputs will be applied. Besides this, a double power supply system is provided and some backup facilities for most important controller are added.

As regards the software, a simplified real-time operating system will be implemented to administer the system hardware and the application programs needed for process supervision and control. The application programs, however, are implemented in a modular (block-oriented) way and will usually be generated before being loaded into the final working memory. The modules (functional blocks) used for implementation of individual control-loops of the station will be assembled together out of a program library, which belongs to the distributed computer control system itself, well known as the Library *of Functions.*

Library of functions

As already mentioned, standardisation of application software functions has enabled the system designer to essentially reduce the software generation costs by using the functions needed out of a common recourse called Library of Functions.

In the Library of Functions, routines are situated essentially for the following functional groups:

— Inputting of signals
— Input signal conditioning

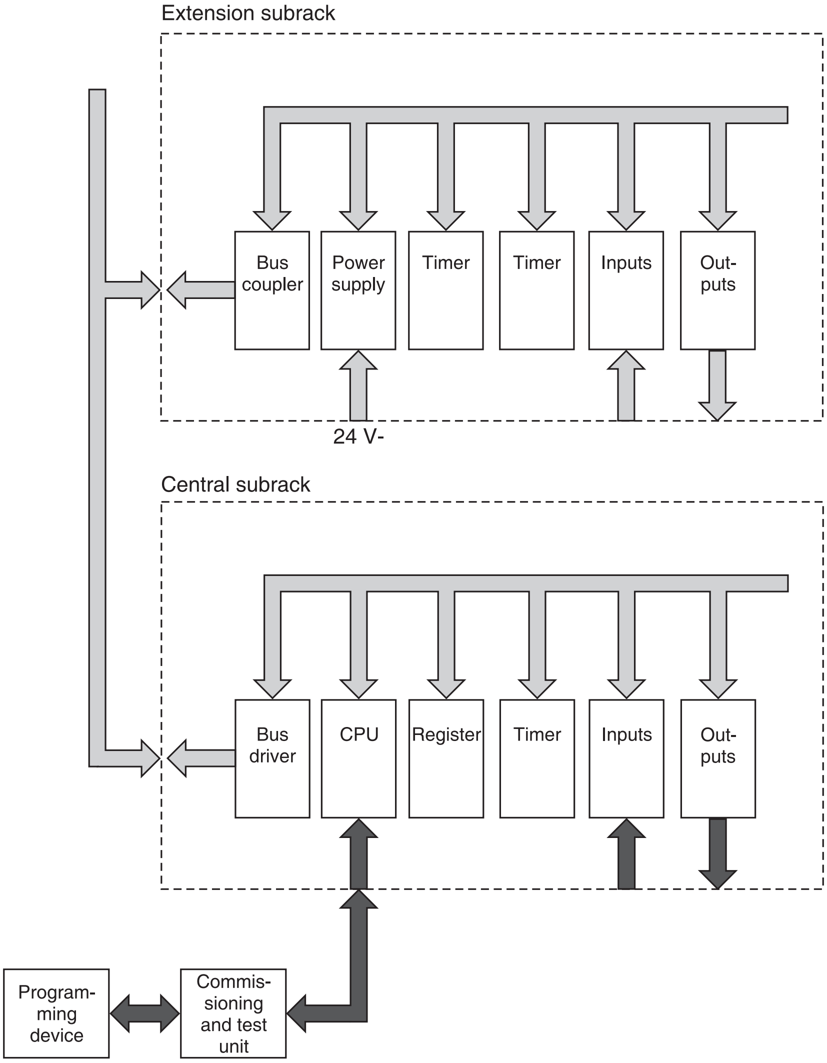


Figure 7.7 Local field station—PROCONTROL.

— Signal and hardware test
— Limit values and trends check
— Alarming
— Reporting and logging
— Mathematical and logic signal processing
— On-line and off-line signal processing
— Visual display and printing
— Output signal conditioning
— Signal outputting.

Each of the functional groups stated can be presented in the library in many different variants, for instance, different algorithms for signals averaging, filtering, closed-loop control, etc. The individual functions will be represented as functional blocks with a standardised number of inputs and outputs and a prescribed maximum number of parameters. To each function, as well as to each parameter a name will be attached. The functions will usually be written in the machine (assembly) language of the computer to conserve memory requirement as well as to minimise their total run time. However, from the transparency and portability reasons they could also be written in a higher real-time language like in Real-Time Fortran or in Language C.

Disregard the programming language used, the program core of each function will preferably be written in the re-entrant form, whereby the necessary data (input, output and parameter values) will be exchanged or kept stored during the execution of the function on the CPU-stack.

7.5.3 Presentation and Monitoring Device

For efficient process monitoring and control, efficient communication is necessary between the process operator and the process to be automated. For this purpose, different data terminals will be incorporated into the system, the most important being the coloured displays with the corresponding operator key boards. On the display screen the following will be visually represented:

— Current values of measuring data
— Positions of actuators
— Logs of plant
— History of individual signals
— State of different plant parts, control loops, etc.

The man-machine dialogue between the process operator and the automation system is carried out. The items to be represented can usually be chosen by the operator; the alarm and the urgent messages being automatically represented by their generation within the plant. Furthermore, the indication (or display) of most important process values will be given at the places within the plant mimic diagram (also represented on the screen) where these are actually situated within the plant.

It is, in general possible to classify the information, represented simultaneously on the screen, as

— plant overview;
— group overview;
— local multi-loop representation of individual process variables reporting and logging;
— representation of secondary plant data (statistical parameters, etc.)

The TDC 3000 provides different types of displays through universal station. A brief summary is given in the following as an example.

TDC 3000 displays are used in three operating modes:

— Normal conditions
— At time of process upset
— In the event of control equipment malfunction

Normal condition displays

TDC 3000 system provides the facility to monitor the continuous as well as batch or sequential processes. We shall describe the display facilities in both these cases briefly.

Continuous process displays

The operator's view of process and plant information is divided into areas (Fig. 7.8) which may correspond to single process sub-systems, units, or entire sections of the plant. Parts of existing areas may be combined to produce other areas. This effectively allows an operator to assist another operator or view different groupings. Within a given area, it is sometimes desirable to present variations of process displays during plant start-up or shutdown conditions. These displays may be stored and called-up by a single operator command. The operator is offered a family of displays that provide as broad or as detailed a view of the process as is desired. Both current and historical process information is available, and the operator may select a display hierarchy that progressively exposes him to more detailed information with each successive step. The following displays are available:

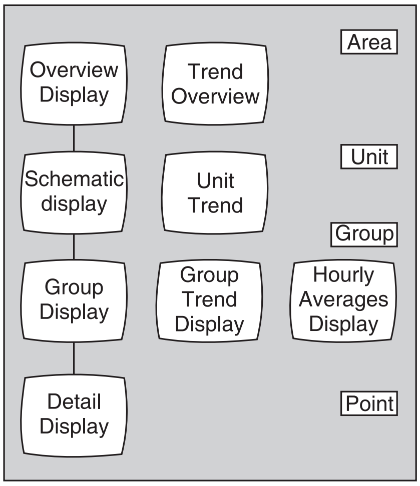


Figure 7.8 Continuous process display.

(i) *Overview display.* It provides deviation and status information (Fig. 7.9) for up to 36 groups of eight points of 36 sequences each. From this display, the operator may quickly access any of the 36 groups displayed in the overview or user defined displays. Overlays can be configured to cluster groups or sequences, as they relate to plant areas. Analog points are displayed as vertical deviation bars while digital points show their alarm status.

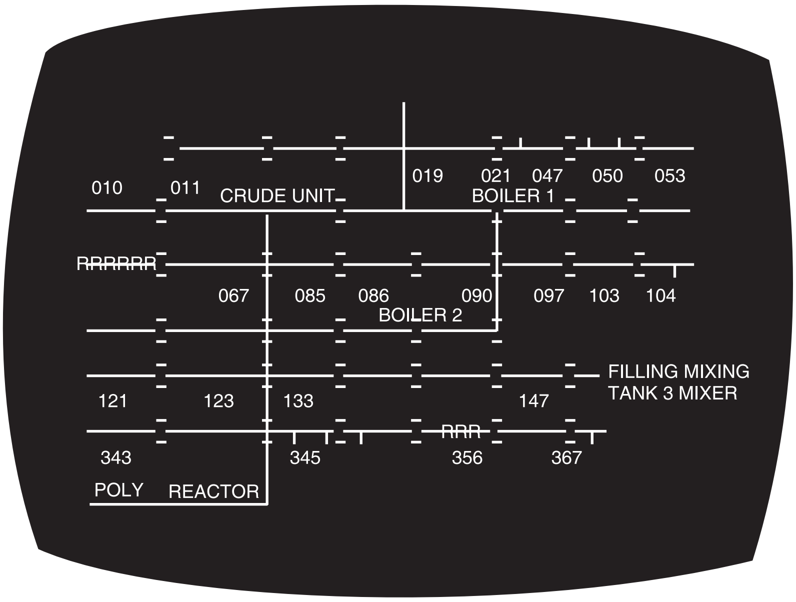


Figure 7.9 Overview display.

(ii) *Schematic display.* Hundreds of user-defined schematic displays (Fig. 7.10) may be created and displayed on the universal station. These process displays may be defined by the engineer to augment the standard format displays. Containing both alphanumeric and graphic

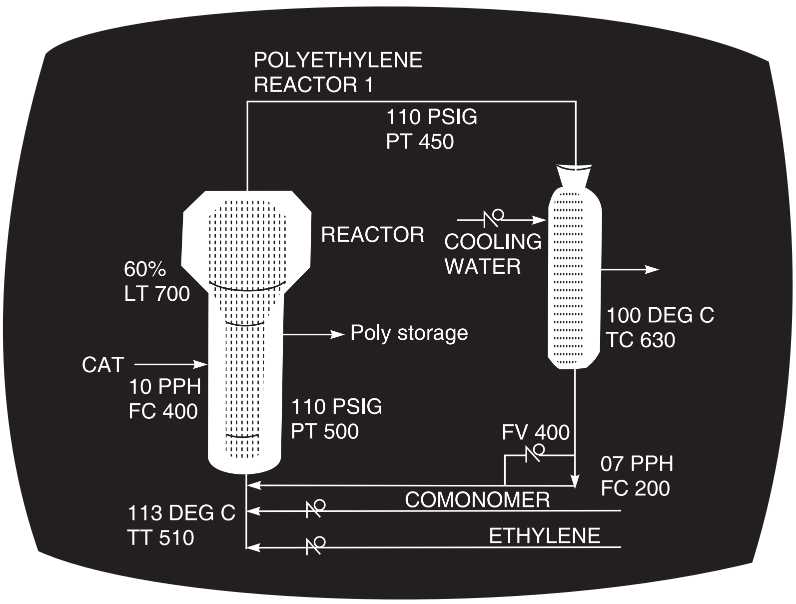


Figure 7.10 Schematic display.

information, they can represent a whole area, a unit, a single tag, or may be based on concepts unique to the user's plant. These schematics may be accessed using either the keyboard or the touch screen. All display targets are assigned to permit touch screen access of any other display. A powerful picture editor is part of system to facilitate the user to create schematic displays.

(iii) *Group display.* Group display (Fig. 7.11) shows parameters for up to eight points and permit operator manipulation. These may be indicating or controlling analog points, single or dual input/output digital clusters, or counter inputs. Additional detail data is progressively exposed to the operator as individual points are selected, either by keyboard or touch screen. Up to 400 group displays may be configured in any universal station.

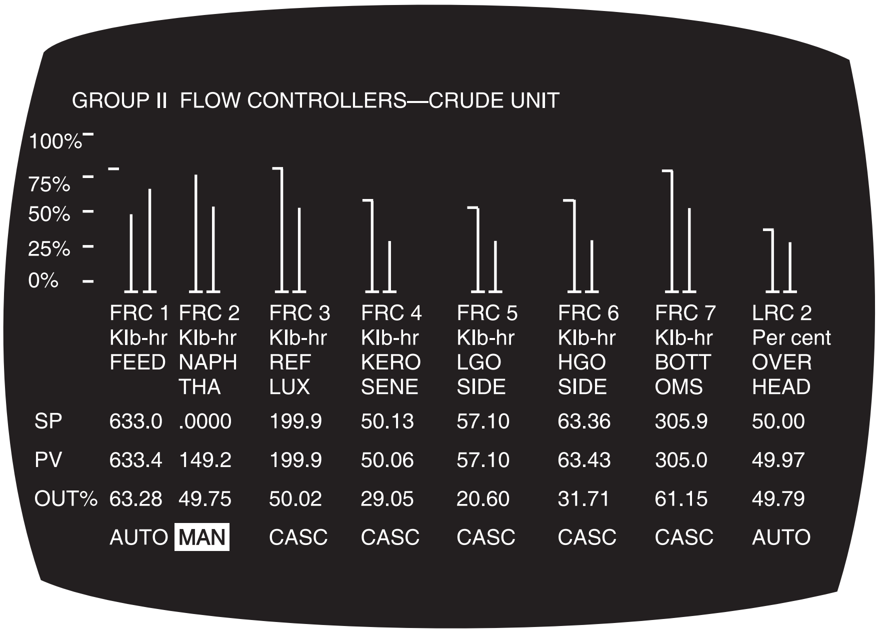


Figure 7.11 Group display.

(iv) *Group display with trend.* Four trend curves can replace the normal bar graph and digital point indications of any group display. These can be preselected or selected from up to 8 variables by the operator (Fig. 7.12).

Trend curves for the other 4 points in the group can replace the bottom part of the group display (Fig. 7.13). History prior to call-up will be retrieved from Basic, Extended, or multifunction controllers or from History Modules. Depending on the source, the samples will range from 20 seconds to one minute, for a full display range of 20 minutes to 8 hours. Trend curves are updated for the period the display is kept on the screen.

(v) *Group display with hourly averages.* Hourly averages for the last 10 hours for each of the 8 points in a group can also be selected to replace the bar graph segment of the group display (Fig. 7.14).

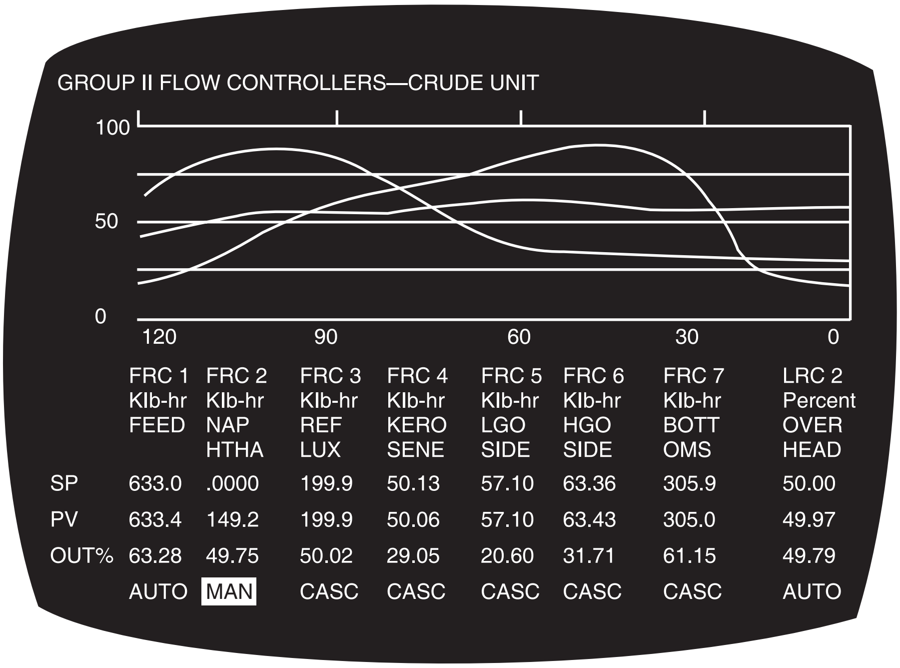


Figure 7.12 Group display with trend.

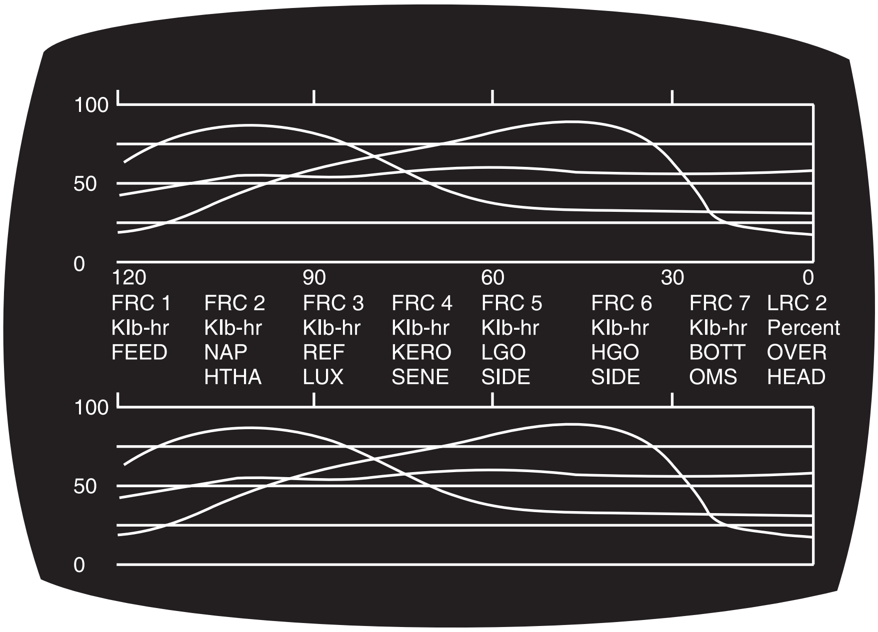


Figure 7.13 Trend curve updating.

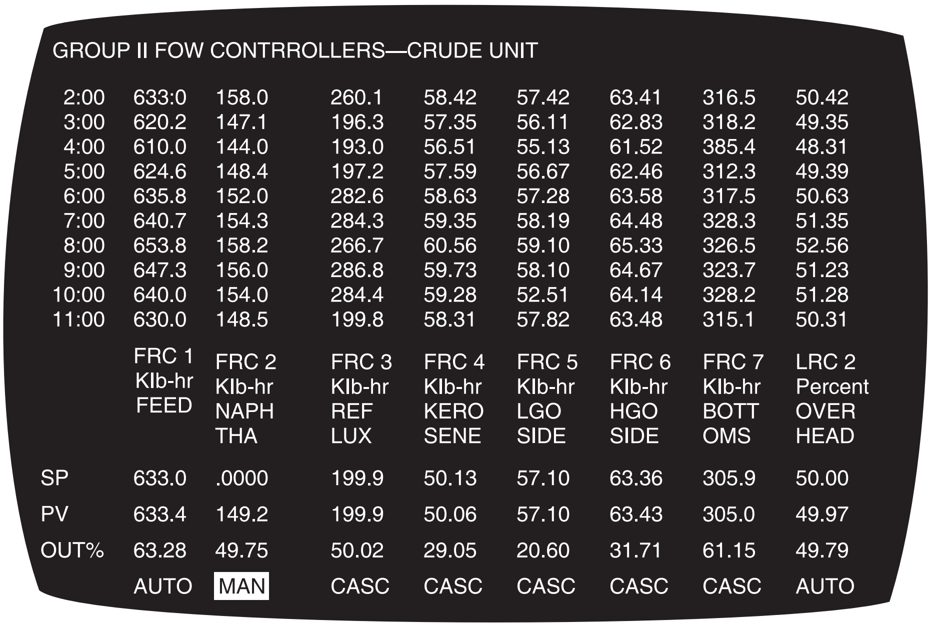


Figure 7.14 Group display with hourly average.

(vi) *Detail display.* The detail display provides visibility of all parameters relating to a particular point. Detail displays may be selected for any point in the system. The 3200 points available in the 400 group displays are accessed directly from the group. These displays (Fig. 7.15) can be called up by touch screen or by tag name from the keyboard.

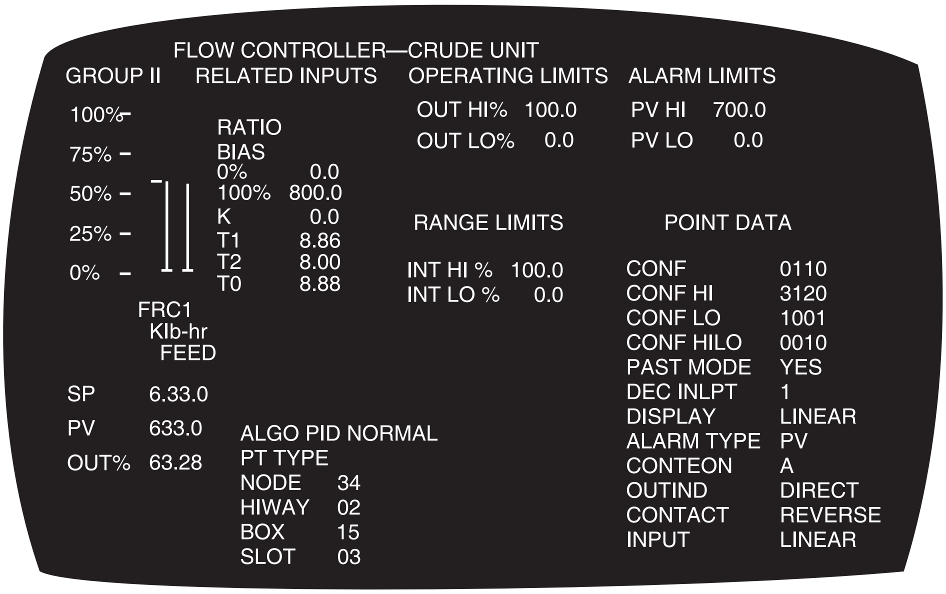


Figure 7.15 Detail display.

(vii) *Trend overview and unit trend.* The trend overview (Fig. 7.16) includes 12 groups of 2 trended variables each, for a total of 24 trends per display. There are similar displays for each of the 36 units within the area. Trace period is selectively 2 or 8 hours.

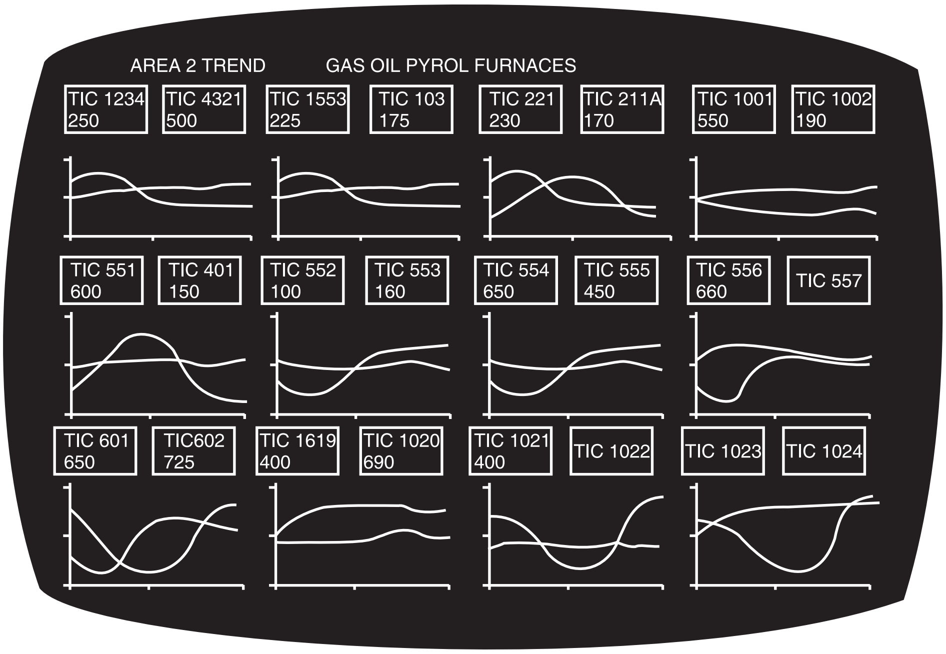


Figure 7.16 Trend overview display.

Batch/Sequence operation displays

In addition to continuous process operations, TDC 3000 provides the facilities to control and monitor discontinuous/batch and sequential processes. The universal station provides displays, specifically designed to simplify management of these processes (Fig. 6.17).

(i) *Batch summary display.* This provides (Fig. 7.18) a one-line summary of information on each active batch.

(ii) *Batch display.* Batch display (Fig. 7.19) shows the status of process modules related to a particular batch.

(iii) *Process module displays.* These displays provide more detailed information about the current batch than the batch display. A process module is the smallest piece of a plant from which a sequence may operate. The process operator can control the execution of the sequence in each process module. The operator can access any process module display in the system.

(iv) *Sequence unit summary display.* A summary is provided by this display of the current status of process modules and sequences for the area assigned to the station. The organisation of the display is defined by the user to allow logical groupings of status information.

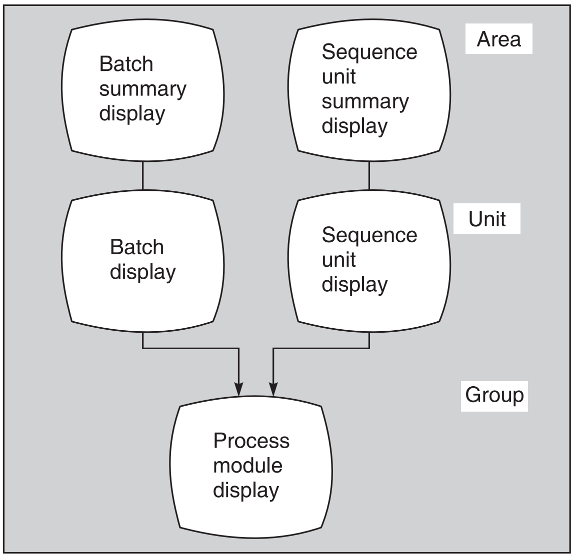
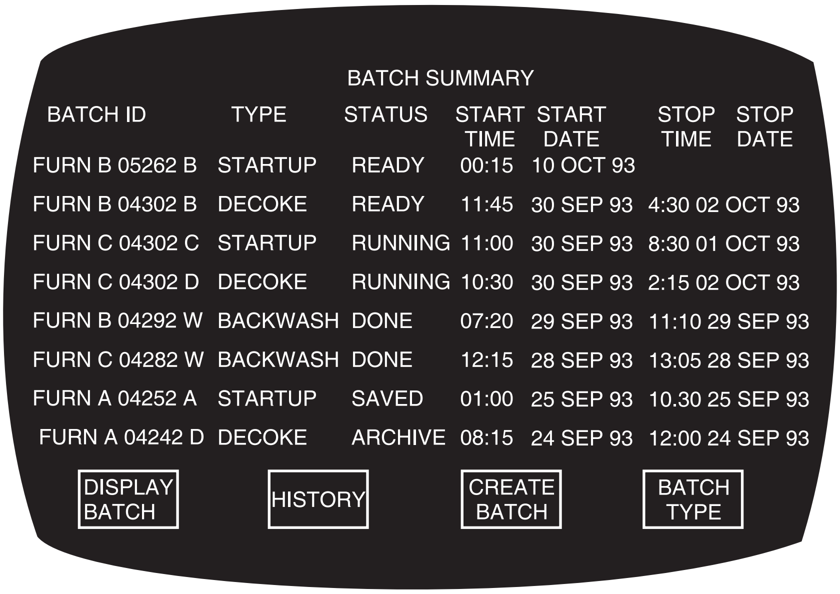


Figure 7.17 Batch/Sequence operation display.



BATCH SUMMARY

BATCH ID	TYPE	STATUS	START TIME	START DATE	STOP TIME	STOP DATE
FURN B 05262 B	STARTUP	READY	00:15	10 OCT 93		
FURN B 04302 B	DECOKE	READY	11:45	30 SEP 93	4:30	02 OCT 93
FURN C 04302 C	STARTUP	RUNNING	11:00	30 SEP 93	8:30	01 OCT 93
FURN C 04302 D	DECOKE	RUNNING	10:30	30 SEP 93	2:15	02 OCT 93
FURN B 04292 W	BACKWASH	DONE	07:20	29 SEP 93	11:10	29 SEP 93
FURN C 04282 W	BACKWASH	DONE	12:15	28 SEP 93	13:05	28 SEP 93
FURN A 04252 A	STARTUP	SAVED	01:00	25 SEP 93	10.30	25 SEP 93
FURN A 04242 D	DECOKE	ARCHIVE	08:15	24 SEP 93	12:00	24 SEP 93

Figure 7.18 Batch summary display.

(v) *Process module group display.* This display allows the operator to change the status and execution mode of sequences and process modules.

(vi) *Message summary display.* Message summary display lists up to 100 of the most recent messages produced by control language programs in application modules or produced by a

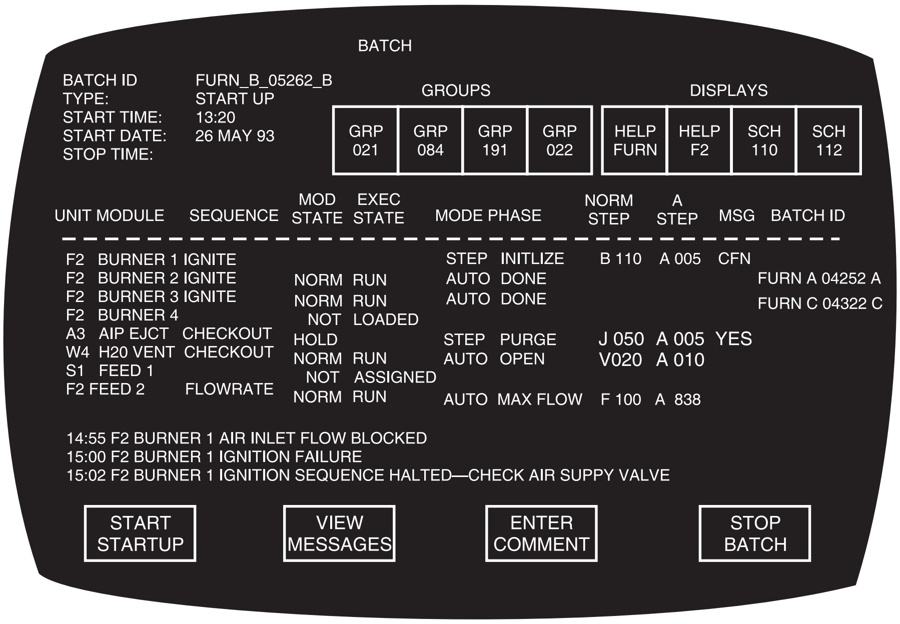


Figure 7.19 Batch display.

multi-function controller on the Data Hiway. These messages are for the operator's information or may contain requests for specific actions.

Process upset displays

TDC 3000 has been designed to provide the operator with the most effective and relevant data not only during normal operating conditions, but also during process upsets. Just as the normal condition displays provide progressive exposure to plant data, the universal station helps to focus the operator's attention and actions during upset conditions.

(i) *Alarm annunciator display.* This display (Fig. 7.20) operates much like a 60 window conventional annunciator panel. Up to 5 alarm points supported by Boolean logic may be assigned to each window, for a maximum of 300 alarms available from this display. The five most recent emergency priority alarms are listed at the top of the display.

Also, displayed on the alarm annunciator display are 36 windows indicating the highest priority alarm in the various units of an area assigned to the station. Simply touching the window on the screen will acknowledge indicated alarms and call up associated displays.

(ii) *Area and unit alarm display.* The area alarm summary display (Fig. 7.21) and unit alarm summary display, each lists up to 100 of the most recent alarms in the area or units assigned to the station.

Control system malfunction displays

Traditionally, operators have had to monitor and differentiate between process upsets and process control system malfunctions. TDC 3000 provides a clear demarcation between the process and the process management system, thus giving the operator a clear view of both functions. A family of displays is devoted to these functions (Fig. 7.22).

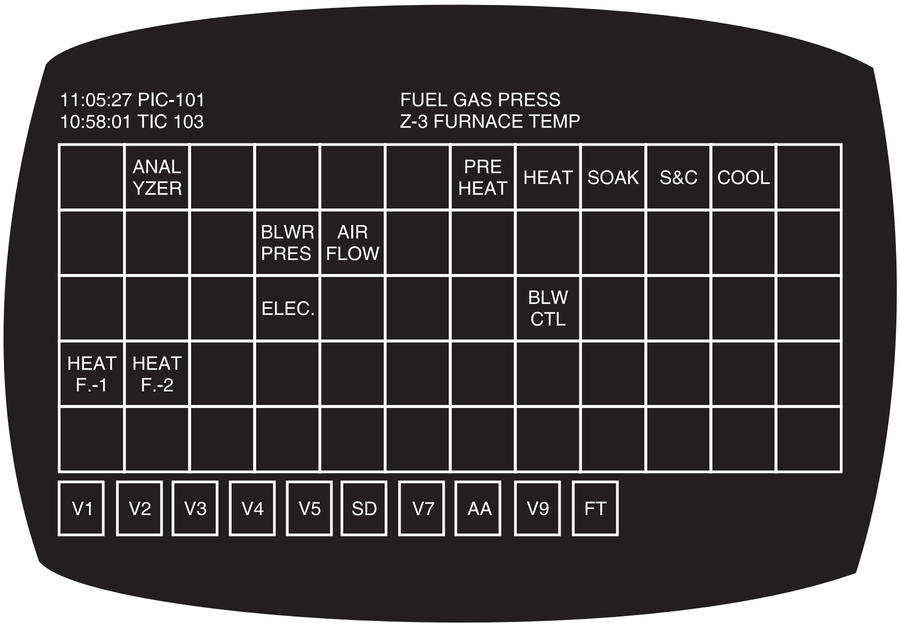


Figure 7.20 Alarm annuciator display.

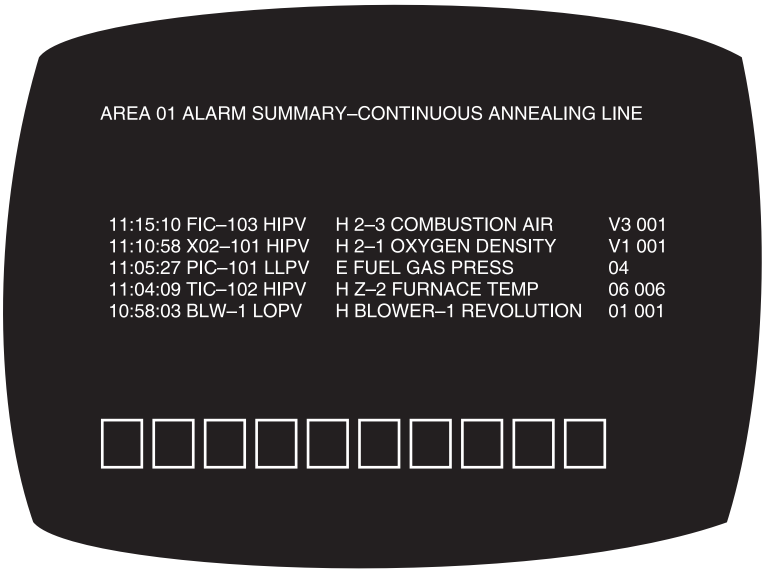


Figure 7.21 Area and unit alarm display.

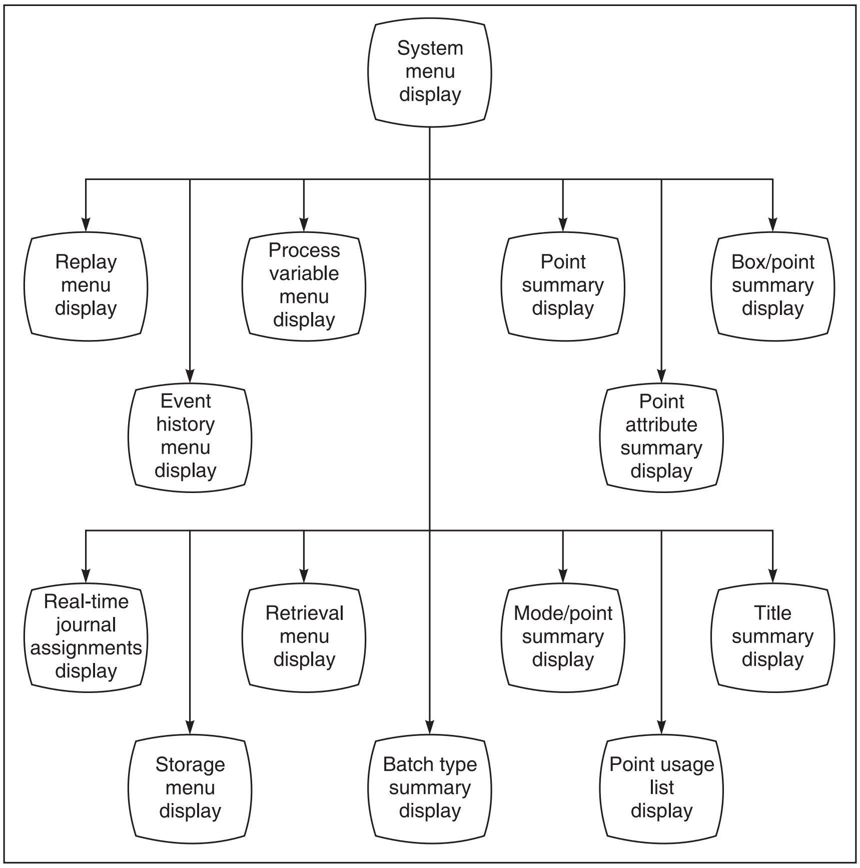


Figure 7.22 Process management system display.

These displays show an overview of the status of all the universal stations in the console, universal station, peripheral devices, each module and gateway on the Local Control Network, and each box on the Data Hiway. These displays access all other system displays including the following:

— Console overview
— Console status and assignment, unit assignment
— Gateway status
— Application module status
— History module status
— Computing module status
— Hiway status
— Box diagnostic.

Through these displays the operator besides being able to verify current operational status can also perform the following functions:

1. Change certain status conditions and assignments of process areas and units.
2. Start up or shut down modules or gateways, switch from primary to back-up modules or gateways, change universal station personalities, and save or restore module, gateway, and box databases.
3. Switch from the primary to the back-up Data Hiway, save and restore box databases, and download the trend memory programmed in certain boxes.

7.5.4 Communication Options in Distributed Control Systems

Communication paths are an essential requirement of a decentralised automation system. These provide the link for data flow between automation equipment attached to the same or to different automation levels.

In fact, the path must be implemented from the top to the bottom of the system, to transfer any necessary control action down to the individual final elements, or to collect the actual informations from them. In special situations, even the application programs will be loaded down to the peripheral stations on demand and initiated there.

The topology of automation system is to a great degree determined by the structure of the process to be automated. The same holds for the communication system used.

For instance, short and long distance buses will be used in a combination to cover a big area of distributed plants, like in the chemical industry, steel and iron plants, mining industry etc. At the same time, a cost-effective interconnection of a large number of equipment distributed over a large area will be implemented. For short distances, the star connection is applicable which usually (combined with the bus and point-to-point connection) will result into a mesh or network structure, as shown in Fig. 7.5.

At Level-1, usually the point-to-point connection will be applied to connect individual sensors and actuators to analog and digital inputs and outputs of the local field station. Recently, field bus was introduced for this purpose (Fig. 7.23) interconnecting the distributed data multiplexers. For shorter distance, an instrumentation bus can be applied, like the. IEEE 488-bus, or special buses like K-110 bus of AEG-Telefunken.

For Level-2 and Level-3 systems long-distance buses extendable up to 5 or 10 km, are generally applied. Here, there are also many buses and data hiways developed by different computer vendors, some of these being of even national or international standards e.g. PDV-bus, PROWAY, IEEE 488-bus, etc. Long distance buses are usually serial buses.

It should also be pointed out that for interconnections of functional. parts of the local station, some local highly parallelised buses will be applied, like for instance back-plane buses. The integration of the process and the office automation has finally made necessary the application of local area networks (LANs) at the highest automation level for instance Local Control Network within the TDC 3000 system of Honeywell.

7.6 CONFIGURATION

The configuration of distributed control system is performed in two steps.

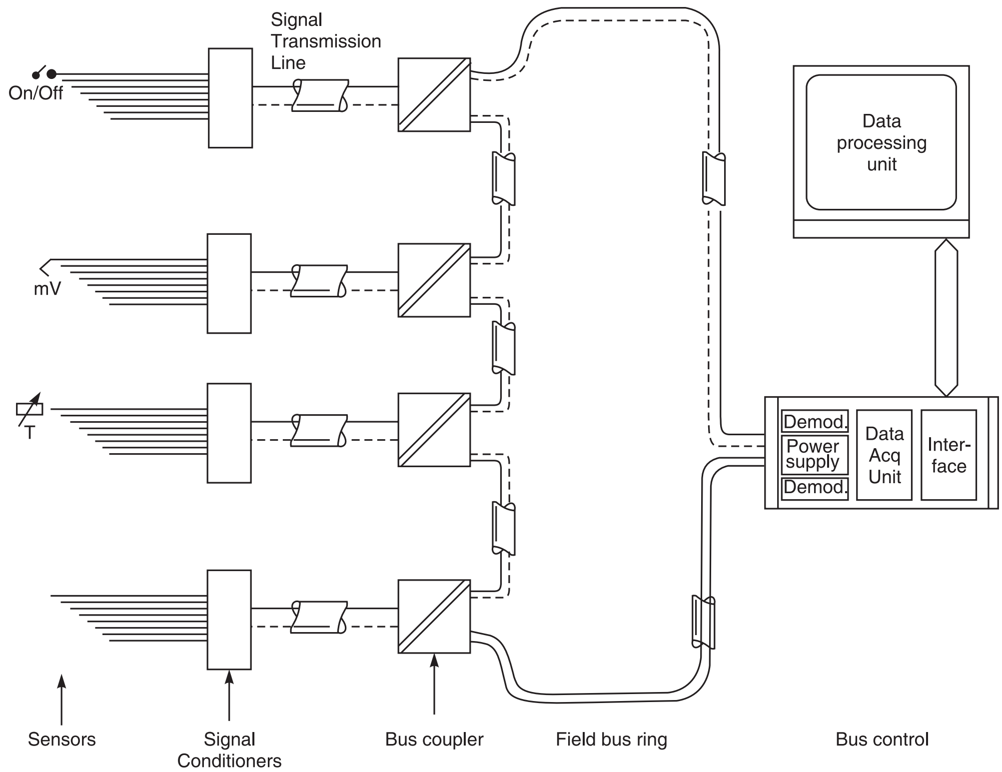


Figure 7.23 Field bus.

Step 1—Configuration of operating system

The operating system is configured to define the following:

- Composition of groups and overviews displays
- Period for Trend Displays
- Hiway Station Priorities
- Message Tables

For DCS systems which are not bus-based, the configuration is done by the manufacturer. The bus-based systems can be configured using keyboard from operating station. Figure 7.24 displays configuration dialogue of the operating system.

Step 2—Configuration of controller functions

The individual controllers are configured for the control strategy.

The configuration of controller functions is done by using various software control modules which are resident in memory. These software modules are known as algorithms. Table 7.1 presents a list of these algorithms in a typical distributed control system. These algorithms are joined together by software addresses (memory and registers store information and addresses)

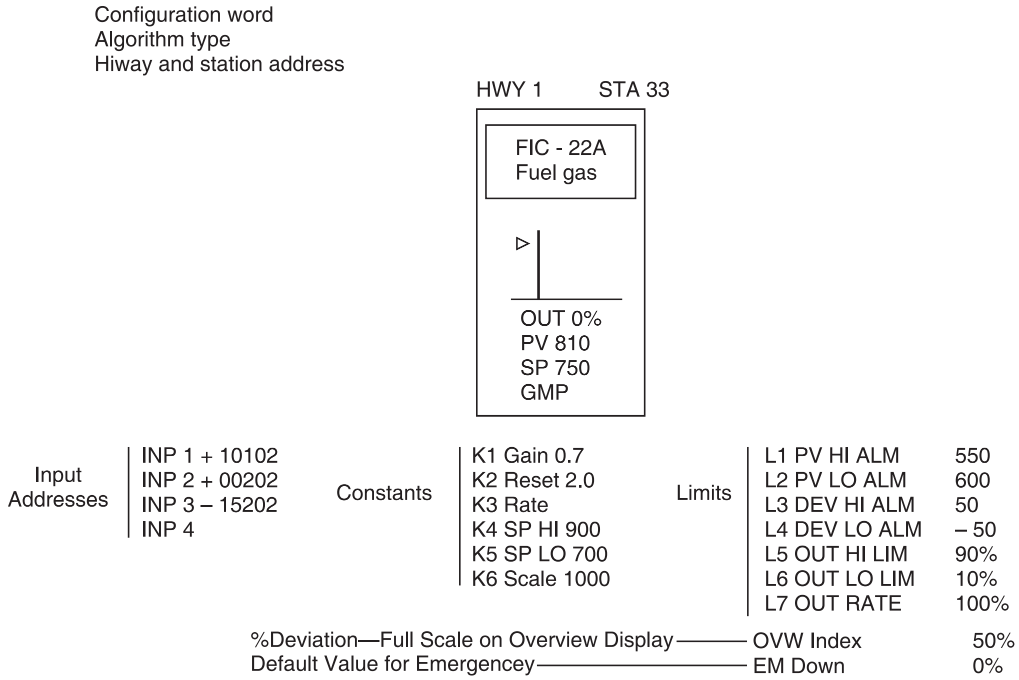


Figure 7.24 Configuration dialog of system.

Table 7.1 Distributed Control Functions Entered by Configuration

1. Hiway definition—Assign station numbers to the various hiway stations and define the type of station (controller file, operator station, etc.).
2. Define the priority of the station.
3. Define overview index scale—This is the percent of deviation that will show on the overview display as full scale.
4. Define points to he recorded.
5. Make a table of units of measure.
6. Make a table of messages.
7. Define tag names and service information.
8. Assign points to group displays and groups to overview displays.
9. Define titles for group displays and for overview displays.
10. Define alarm points to be grouped and summarized.
11. Define alarm priorities.
12. Define trend time spans for points to be trended.
13. Download configured information from controller files into memory. Reload configuration information from memory into controller files.
14. Enter addresses (source of inputs), parameters (gain, reset, sensitivity, ratio, etc.), and limits (alarm limits, output rate, etc.) into individual time slot functions.
15. Utility routines: format disks, copy, combine recipes.

to form a combination of functions. This is very similar to the manner in which two chips are joined together. Considering the similarity, the joining of software modules is known as *soft wiring*.

The controller configuration can be performed by defining the function at high level. This function in turn takes care of configuration at lower level algorithms. On the other hand, controller function can also be configured by defining lower level algorithms and sequencing them to form the control strategy.

7.7 SOME POPULAR DISTRIBUTED CONTROL SYSTEMS

Having described the concepts and sub-systems of distributed control systems, we shall discuss in detail some of the very popular distributed control systems.

7.7.1 Leeds and Northup Max-1 System

The basic components of the system at the local level are listed here.

1. *Controller Loop* (mostly analog functions), or batch (both analog and digital functions).
2. *Operator Mini-Station* comprising a Data Input Keyboard and a multi-colour video screen for display.
3. *Analog Input/Output Terminal Board* for transmitter inputs and valve outputs.
4. *Digital input/output Terminal Board(s)* for contact closures.
5. *Manual-Station Junction Panel* and Manual Stations, for independent hard manual backup.
6. *Records Station* for external storage of database (configuration of loops and parameter values), printout of alarm lists and configuration data, connection of trend recorders. It can also be a communication link to an external intelligent programmable device.
7. *Graphics Terminal* for display of custom colour graphics with real-time data.

The basic controller is a microprocessor-based control module. It accepts up to 30 analog inputs. Up to 256 digital inputs and outputs can also be processed. There are 8 analog outputs. The controller file has 16 times slots, 8 primary and 8 auxiliary. The 8 primary slots have an analog output. The auxiliary slots can be used for computation, cascading etc. The microprocessor scans these slots in turn twice a second. It performs the calculations specified by slot configuration.

There are 40 different algorithms like PID, multiplier, divider, two-mode adaptive tuning, non-linear function generation, logic control functions including chainable 8 step 8 output sequencer and ramp generator for set-point programming.

Up to 8 controllers can be connected to a mini station (Fig. 7.25). The mini station can display data from up to 64 analog control loops. The mini station consists of the VDU and keyboard. The high scan-rate CRT offers a combination of seven colours, for either data or background, to contrast the differences in data displayed and improve the 'man/machine' interface.

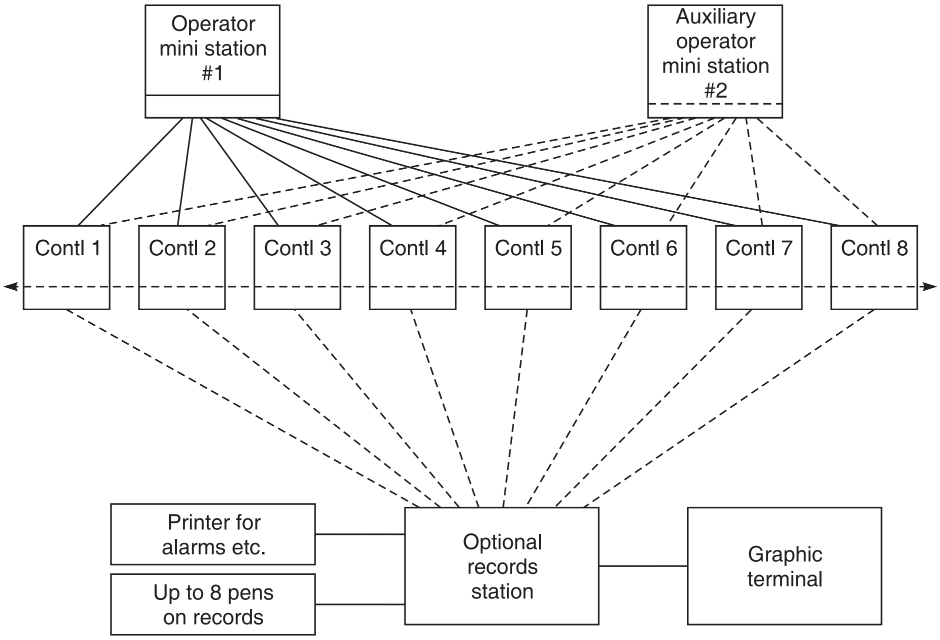


Figure 7.25 Leeds and Northup Max-I system.

Through keyboard selection, the video screen can provide three different types of displays:

1. Group Display—current operating data for all eight primary or eight auxiliary slots.
2. Digital Display—current operating data, historic or real-time trend curve, and configuration data and parameters for a single slot.
3. Process Input Display—current values and engineering data for all 30 analog inputs.

7.7.2 Control Bailey Micro-Z System

The controller has two microprocessors—the control microprocessor and the communication microprocessor (Fig. 7.26). The algorithm library resides in a ROM. The specific settings of the controller to match a problem are featured in a removable personalisation card. The personalisation card consists of the process control scheme and the controller parameters. The communication processor manages two serially connected asynchronous 2,400 baud lines, allowing to communicate with the operating centre through the concentrator or the selector. Both the lines are independent and galvanically isolated.

To achieve a video operating centre, a high speed serially connected bus is used to transmit data between the various items. Distance may be as much as 2 km between a cabinet room and the control room.

The data is transmitted along the following two types of supports:

1. Shielded twisted pairs in the areas where due to the presence of several subscribers, by-passes are necessary on the bus.

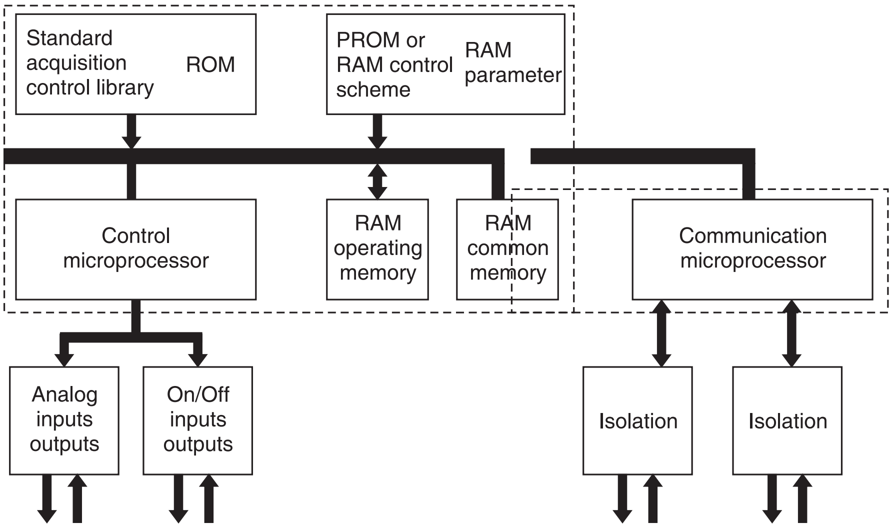


Figure 7.26 Control Bailey micro-Z system.

2. Optic fibre for long distances without by-pass between two groups of subscribers.

On the general purpose control station, the various functions of separate set-point station and control station are collected. This station allows sequential control of the various process loops. A loop is called on the selection keyboard which allocates the selector which is indeed a solid-state multiplexer.

Transmission between Micro-Z cards and the general purpose control station is asynchronous up to 2,400 baud. The selector-control station assembly is completely independent of the concentrator-bus system. Since each Micro-Z card is provided with two 2,400 baud serially connected communication lines, two communication systems can coexist in the same unit and on the same Micro-Z card. The operation can be from one of the three types — (a) Centralised Video Station, (b) General Purpose Control (GPC) Centre, and (c) Conventional Control Stations. The general purpose stations can be connected to any control loop and adjustments made. The GPC can act as plant level control. Either GPC or conventional control station can act as redundancy for the Video Control Centre.

7.7.3 Honeywell TDC-2000 System, TDC-3000 System and TPS System

The TDC 2000 and TDC 3000 systems follow a modular architecture, so that complete implementation of a large process control system may consist of a large number of units/ modules which may include 60 or more microprocessors, as shown by the overall schematic in Fig. 7.27, or may have only the few basic units needed to control small systems. Here we shall describe briefly the technical features of the basic controller. Other hardware units necessary for the entire system are more briefly described; emphasizing those utilizing separate microprocessors.

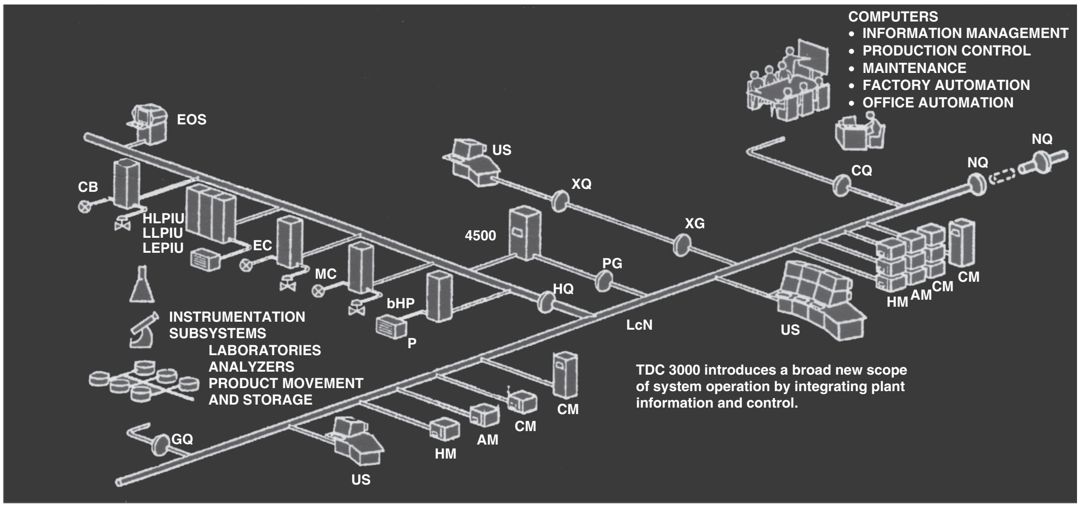


Figure 7.27 TDC 3000 system architecture.

The basic modules are:

1. Controller file
2. Operator station
3. Hiway Traffic Director (HTD)
4. Analog unit
5. Process Interface Unit

Before examining the details, let us first obtain an overview of the system to understand how these modules function together.

As discussed above, the system architecture is that of multiprocessor units linked by a data hiway (or “hiway”, as it is frequently spelled in control systems practice). Each processor is a bus-oriented microprocessor linked to the data hiway (hiway) through interface units that permit DMA data transfer between units.

Referring to Fig. 7.27, the flow of data on the hiway is directed by the HTD, which repeats messages on the various branches, assigns priority access to units competing for the serial channel and conducts the polling of modules acquiring change-of-status data. The HTD does not check messages for accuracy; this is accomplished by individual interface units.

The hiway connects the individual controller files (microprocessors). Each one of these controller files is wired to process transducers (I/O Devices) and can acquire PV (Process Variable or Control Variable) and SP (Set Point) data or control, up to eight individual loops. Conventional appearing panel meters or recorders, displaying PV and SP data, can be wired to the controllers using plug-in cards. (Analog signals can also be modified with electronic analog computing units or signal conditioners (auxiliaries) but these are also direct-wired and do not affect the digital operation of the system). PV-SP data can be displayed digitally by means of a Data Entry Panel (DEP) which also permits configurations of up to 16 controllers, or manual control of any loop, via a push-button operator interface. All this data exchange is carried on external to data hiway and is independent of it. Therefore, the controller file and its peripherals can be used in a standalone mode to provide conventional regulatory control of eight loops.

Each controller, however, interfaces with the hiway, permitting any part of its database to be communicated to another module. These units have a function similar to the "intelligent" terminals used in data processing computer systems, in that the display and the keyboard operation are controlled by the microprocessor, using its own ROM program and database (up to 40K words of memory) without reference to a central computer. The keyboard permits to select various fixed program displays showing the status of the overall system or process, or that of individual controllers and loops. It also allows the operator to intervene directly in the configuration of controllers and in the automatic or manual control of loops. The operator stations can be used individually or combined in an operation centre console so that several displays can be presented simultaneously to the operator.

The data hiway also provides essentially the same service for host computers interfaced with it for supervisory or data processing purposes. That is, the host computer may have access to each controller's database and can manipulate it or the output to the process, or operate on and display the data, in the same manner as the operator. The host computer of course can be of any size and power so long as it obeys the hiway protocol, or it can be omitted entirely, leaving the full functioning to multiprocessor system.

Figure 7.27 also shows modules called analog units, connected to the hiway and process similarly to the controllers. These units perform the same function as far as the hiway system is concerned, but do not have the instructions for process control algorithms that allow independent operation of the controller. They are thus primarily I/O devices, permitting data acquisition and manual control of loops via the operator station (in manual mode) or a host computer in DDC mode. Since they lack independent regulatory loop control, their program memory structure is much simpler and they can service 16 loops (16 analog inputs and 16 analog outputs) rather than the controller's 8 loops. Also, because the analog units are not controllers, they need not be configured and have no need of the DEP.

Another significant module appearing in Fig. 7.27 is the Process Interface Unit (PIU). This device is unique in that it can be polled by the HTD. Its major function is to interface the process and acquire large quantities of data, either digital (status of contact closure, for example) or analog, such as thermocouple millivoltages. A change in digital status is recognised as a response to the regular polling inquiries of the HTD, after which the unit is requested to transmit its new data. When this capability is not used (and it is useful mainly with a sequence or digital logic system controlled by a specifically programmed host computer), the PIU is employed as an analog data acquisition unit (A/D and D/A) and responds on the hiway only to requests from the operator station or the host computer, in the same way the analog unit and controller file respond.

Honeywell has announced its next generation system—the Total Plant Solution (TPS) system, which provides a unified environment for business and control information. Figure 7.28 shows the architecture of TPS system. The backbone of the architecture is the merging of the plant intranet (i.e. Ethernet) with the process networks including Honeywell's Local Control Network (LCN), Universal Control Network (UCN) and Foundation Fieldbus.

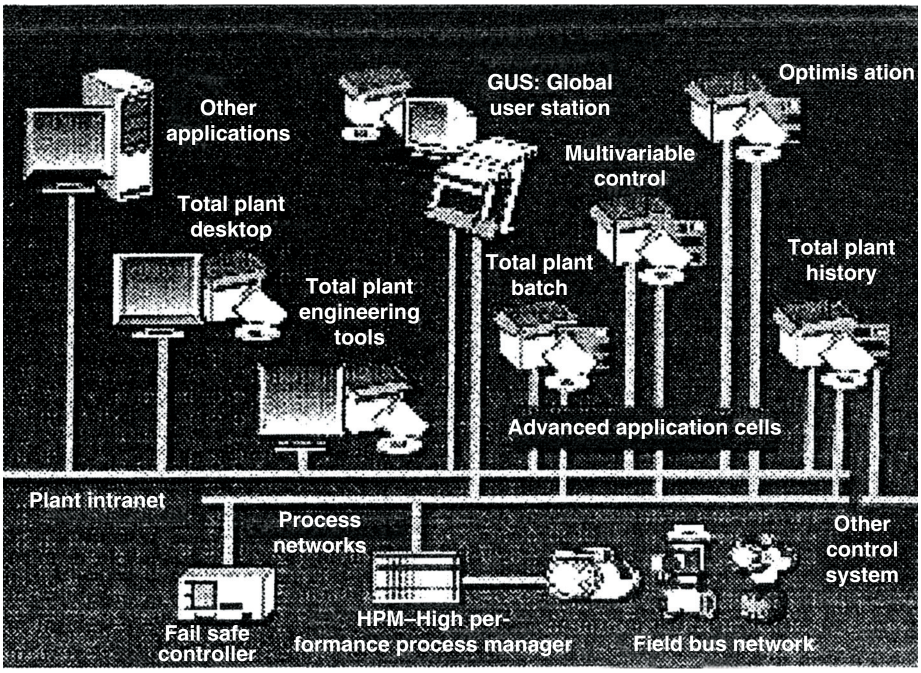


Figure 7.28 Total plant solution system [*Courtesy*—Honeywell].

The basic components of TPS system are:

— Global User Station (GUS)
— Total plant desktop module
— Total plant engineering tools
— Total plant history module
— Advanced application software packages like Total plant batch control, Multivariable control, Optimization etc.

GUS (Global User Station) and TPS applications software are based on Windows NT and support all microsoft tools such as OLE, ODBC, SQL (structured query language), net DDE (net dynamic data exchange) and video. It also supports OPC (OLE for process control) which extends OLE (Object Linking and Embedding) technology for common object exchange among different process control applications.

Like TDC 2000 and TDC 3000 architecture management of common database is strong feature of TPS. The heart of TPS architecture is GUS designed to be the Windows NT graphical user interface to all Honeywell control system—TPS as well as TDC 3000 and future fieid bus systems.

7.8 FIELDBUS SYSTEM

Fieldbuses are special local area networks that are dedicated to data acquisition and the control of sensors and actuators. They typically run over low-cost twisted pair cables. They differ from many traditional LANs (such as Ethernet) in that they are optimised for the exchange of short point-to-point status and command messages.

The original application for factory automation fieldbuses was to connect remote I/O termination units to the PLC. The original remote I/O unit was a small equipment rack to contain the PLC's I/O cards, but modern remote I/O (or block I/O) often appears to be an electrical termination block for connecting sensors and actuators. The signal-processing intelligence of the field bus is contained in the block.

7.8.1 Fieldbus Types

The main standards for Fieldbus products are:

- **FOUNDATION Fieldbus.** This was formed in 1994. The H1 standard has defined low-speed 31.25 kbps transmission. The H2 standard (which is equivalent to WorldFIP) operates at 1 Mbps. The Fieldbus foundation is an initiative of mainly USA-based vendors. Its main aim is to standardise Fieldbus for the petrochemical/chemical industries. It aims not to replace traditional DCSs (Distributed Control Systems), but to integrate with them.
- **WorldFIP (or WorldFIP Europe).** This standard has been incorporated into many products and supports a 1 Mbps transmission rate. WorldFIP contributes to the Fieldbus Foundation in their standardisation process.
- **Profibus.** This has three main types: FMS (Flexible Manufacturing Systems), DP (Distributed Peripherals) and PA (Process Automation). FMS and DP use RS-485 signalling, whereas PA uses the IEC physical layer at low speeds. FMS and DP are part of EN50170.

Other buses, such as the CAN bus, use only the lower layers of functionality, especially for remote I/O. The CAN bus is a truly distributed control system as it does not need a controller to control the flow of data between nodes.

The CAN protocol is an ISO-defined standard (ISO 11898) for serial data communication at bit rates up to 1 Mbps. It was initially developed for the automotive industry, and has the great advantage that it uses a common bus which reduces the need for wiring harnesses. It has since outgrown this application. The standard includes a physical layer and a data-link layer, which defines different message types, arbitration rules for bus access, and methods for fault detection and fault confinement.

7.8.2 FOUNDATION Fieldbus

FOUNDATION Fieldbus is an open specification for sensors, actuators, analysers, and so on. It allows:

- The control functionality to actually reside in field devices; and
- The support of other diagnostic, process operation and maintenance functions within field devices.

FOUNDATION Fieldbus enables time-critical closed loop control data to move back and forth on its fieldbus network. The architecture of FOUNDATION Fieldbus is “field control”, or the seamless construction of a control loop consisting of a highly interconnected set of function blocks to perform the closed loop control function with, or without, the participation of a host controller. Those who favour it consider field control to be more reliable, more accurate, less expensive, and more responsive than distributed control.

In the past, 4–20 mA standards have been used to transmit plant information to controllers. This has in some places been replaced by transmitter vendors providing their own digital protocol to allow bi-directional communication between the control system and smart transmitters. Fieldbus has finally allowed a standardized method for process control to move from being centralised to become distributed, and the control to actual reside in field devices such as transmitters, valves and analysers. It provides a digital communications channel and a user layer to provide intercommunications. Its benefits are:

- **Interoperability.** This allows different suppliers to be used for devices.
- **Wiring cost savings.** One communication channel can transmit many digital signals.
- **Flexible control implementations.**
- **Increased field information.** This includes processed data, averages, minima, maxima, diagnostic information and operational information.

FOUNDATION Fieldbus was a response to the needs for a two-way, all-digital, data-transmission network technology for use in process control. From the beginning, it was to replace the 4–20 mA DC transmissions previously used for analog control instrumentation. It was also to use the same type of wire typically used for analog transmission, supply power to field instruments, and to fully conform to intrinsic safety requirements. Fieldbus was initially defined by the ISA’s SP50 fieldbus standards committee, which outlined a two-way, multidrop, digital communications standard for the interconnection of sensors, actuators, instruments and control systems.

7.8.3 Fieldbus Topology

Most analog transmission methods and many digital field communications methods require a single twisted-pair wire for the transmission of a single process variable. Fieldbus differs from this in that it can connect buses point-to-point, either with spurs, as a daisy chain, as a tree, or as a combination of any of these, as shown in Fig. 7.29.

Bus with spurs. All the devices connect to a common bus and they connect through junction boxes.

Daisy chain. All the devices are chained one by one to each other. It is similar to the bus with spur, but does not use junction boxes. It is a useful method of connecting devices, as new devices can be added by simply daisy-chaining from a close device. The disadvantage is that the devices must be disconnected in order to connect a nearby device, unless a special connector (that allows a connected device to remain connected) can be used.

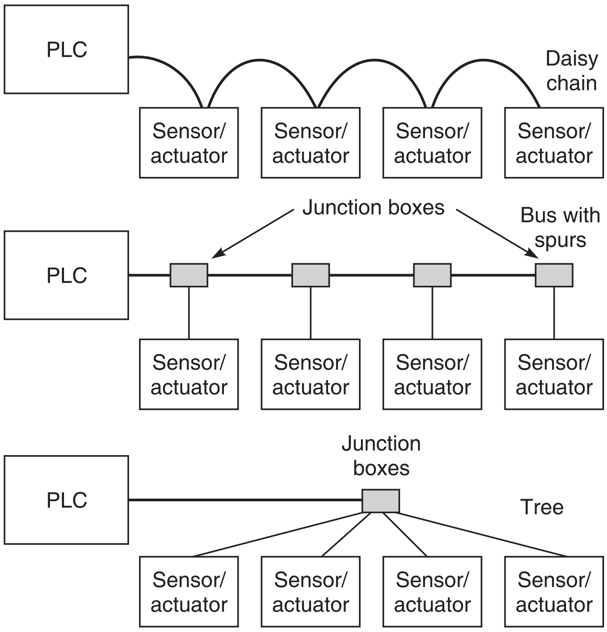


Figure 7.29 Fieldbus connection topologies.

Tree. This type of topology uses a single junction box, with the devices connecting directly to the junction box. Typically, it is used when devices are added and deleted from the network on a regular basis.

7.8.4 Fieldbus Layer

The FOUNDATION Fieldbus consists of two main layers: the communications layer and the user layer; the components in these layers are illustrated in Fig. 7.30. The user layer operates above the communications layer and includes function blocks, resource blocks, transducer blocks and alarm notifications.

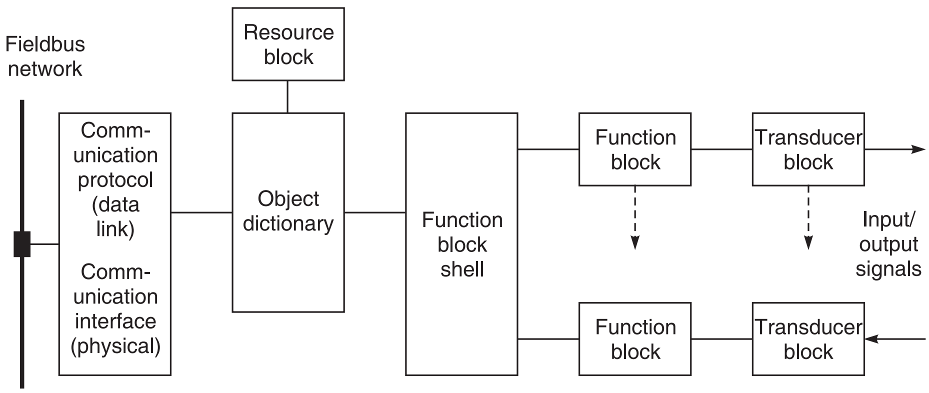


Figure 7.30 FOUNDATION Fieldbus architecture.

Function Blocks (FB). The user layer supports device configuration and uses function blocks. A device can have any number of function blocks. These are used for control, diagnostic, safety and production accounting purposes, and define such things as standardised parameter names, data types, a cascade initialisation mechanism, status propagation, trend collection mechanism, execution scheduling mechanism, block modes and behaviours in response to mode changes, status of process variables, and rules for propagation of status and behaviours in response to status changes. They include standardised function blocks, vendor-enhanced function blocks and vendor-customised function blocks. Two of the blocks are:

Resource blocks (RB). Each device also has an RB, which contains parameters related to the physical device such as manufacturer ID, type of device, revision, memory usage and free space, computational time and device state (on-line/off-line/standby/fault condition, etc.).

Transducer blocks (TB). Each device has a TB, which stores the parameters' association with the sensor or actuator.

Alarm Notifications. Each block (functional, resource or transducer) can produce an alarm notification for any problem which is associated with that particular block, such as process problems with function blocks, sensor/actuator problems with transducer blocks, and overall device problems with resource blocks.

The devices have parameters, which are structured using an object dictionary, in a standardised method of interrogating and referencing parameters over the communications link.

In FOUNDATION Fieldbus there are 16 priority levels divided into the following four classifications:

1. Priority 0 disables alarms including the setting of the alarm condition status flag;
2. Priority 1 disables report, but causes the status flag to be set;
3. Priority 2–7 sets advisory alarms and priority;
4. Priority 8–15 sets critical alarms.

Device description language

The FBs, TBs and RBs are not just limited to a standardised set of parameters. A new Device Description Language (DDL) allows manufacturers to specify additional parameters in a standardised manner. This includes names, data types, enumerations, units, valid ranges, user entry limits, entry conditions (such as out of service or manual mode), connection properties, presentation information and help text. Updates can be easily installed with DDs (Device Descriptions), which are compiled forms of DDL. This allows easy updates and bug fixes on equipment, as updates can be downloaded into the equipment.

Control

The two main control advantages of the Fieldbus are that it truly distributes control and that control processing can be done concurrently rather than in a centralised manner in a controller. The devices can implement many of the control functions that a traditional DCS would do. Most of the control functions are implemented by:

- Two input blocks (analog and digital).
- Six control blocks, such as PID (proportional–integral–derivative).

Other (less used) control functions include pulse input, arithmetic, dead time, splitter and signal characterisation.

Most current control systems use a DCS (Distributed Control System) to control and the transmitter simply determines the process variable. Smart transmitters will change this as basic regulatory control can be moved to the transmitter. This reduces the loading on the controller, and as the functions become more complex, may eliminate the controller altogether.

Control with Fieldbus is relatively easy if the devices are located on the same bus, are located in close proximity, and if the elements of control are located relatively close to each other. This allows function blocks to be linked without having to span different bus segments, and thus reduces delays.

Messages on the bus are divided into two classes:

Cyclic. These messages involve process data which is transferred between linked function blocks and can be made part of a network.

Acyclic. These are single transfers of data. The scheduling of these is determined by the control equipment and is flexible in its approach, thus allowing the bandwidth to be used effectively.

Diagnostic information

Maintenance methods differ from plant to plant. These are:

Preventative Maintenance (PM). This is where the plant is inspected and, if necessary, replaced before faults occur. In some cases, PM can cause more problems than it is worth, because when a piece of plant is disturbed it can often lead to faults that would not have happened. Unfortunately, operating PM properly requires a great deal of information about the previous history of operation of the plant.

Deferred Maintenance (DM). This is where maintenance is deferred to save costs. Unfortunately, deferred maintenance can often lead to long-term costs, typically causing plant shutdowns, complete rebuilds for expensive equipment, or in general, an unsafe plant.

In the past, manufacturers have built in diagnostic information to microprocessor-based devices. Unfortunately, the method of implementation has been non-standard. Typically, each diagnostic signal required an additional 4–20 mA signal to be sent to a host or DCS. In some cases, a proprietary digital protocol allowed the transmission of multiple diagnostic signals over the same pair of wires. All this required extra control system programming and alarm handling. The Fieldbus overcomes this with a standardised comprehensive alarm reporting mechanism and the DDL. A host- or DCS-supporting Fieldbus does not require any special configuration or programming to accept the manufacturer-specific predictive diagnostic information.

The diagnostic information can be used to determine when a device needs to be maintained or replaced. For example, an instrument may have a battery backup. The microprocessor can then monitor the voltage level of the battery. If the level falls to a given value, the microprocessor can sound an alarm that the battery requires maintenance. The intelligent plant that warns its operators when it is about to fail is one step closer to the requirement.

The Fieldbus allows for peer-to-peer communications. Thus, intelligent sensors can talk to each other, and allow the interaction of devices, typically to make calculations from process measurements that allow instruments to determine if a fault is localised or due to a process upset.

7.8.5 WorldFIP

WorldFIP operates at 1 Mbps over twisted-pair cables, and is a reliable method of transmitting variables (from sensors and to actuators) and messages (such as events and configuration commands). It uses a bus arbitrator that broadcasts variable identifiers to all the nodes on the network. This triggers the required node which produces the node to respond with the required value. All modules that need this value must then read it. Its main characteristic is that it supports a distributed, decentralised database of variables. It does not require node addresses, as messages are broadcasted by a bus arbitrator, and then the response is from the node which contains the processor parameter.

The WorldFIP protocol is an open-system international fieldbus standard (EN50170) and is used to interface to Level Zero (sensors and actuators) and Level One (PLCs, controllers, and so on) devices. It can be used in many different architectures such as centralised, decentralised and master-slave. The control algorithm either can be located within a single processor or can be distributed. Figure 7.31 shows the layers of the WorldFIP standard.

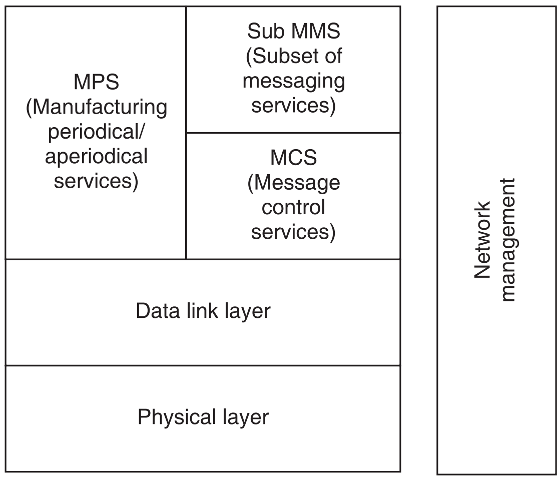


Figure 7.31 Layers of WorldFIP standard.

Physical layer

The physical layer ensures the transfer of bits from one device to another. In the main specification the transmission rate is 1 Mbps over Shielded Twisted-Pair (STP) or Optical Fibre Cable (OFC). The three defined rates are S_1(31.2 kbps), S_2 (1 Mbps) and S_3 (2.5 Mbps), and there is an additional speed of 5 Mbps (fibre-optic).

Data link layer

The data link layer supports two types of service:

1. Exchanges of variables;
2. Message transfers.

These can be either cyclic or an explicit user request. A cyclic message is when the system configures object names. These exchanges are automatically sent without the user requesting them. An explicit user request involves requesting variables and the related response.

Application layer–Physical layer interface

The data link layer provides an interface between the application layer and the physical layer. It consists of a number of produced and consumed buffers, which contain the latest values updated by the user or by the network. These buffers are overwritten when the value is updated, and are automatically created on the initial configuration of a station.

7.8.6 Profibus

Although Profibus started out as a standard communications link between PLCs and host systems such as HMI, the earlier profibus–FMS was too slow to support HMI update.

When a standard connection with PLC remote termination units or remote multiplexers became a requirement, Profibus–DP came out to solve both problems. The high speed of Profibus–DP, up to 12 Mbps, became its most attractive asset. This makes Profibus–DP both a control level bus and a fieldbus. Profibus International prefers the term *profibus* rather than any of its modifiers such as FMS, DP or PA, but industry continues to use these designations.

Profibus–PA is a hybrid protocol using Profibus–DP command structures but the same physical layer as FOUNDATION Fieldbus. Profibus–PA is for use in traditional process control applications where delivery of DC power to the field instrument and support of intrinsic safety is necessary**.** Unlike FOUNDATION Fieldbus, Profibus–PA is a master/slave network that is an extension of Profibus–DP.

Normally, the field instruments wire to a field junction box where they terminate in a Profibus DP/PA coupler. Profibus–DP serves as the higher-level control level fieldbus to connect PA segments to the control system master.

7.8.7 Control Area Network (CAN)

Controller Area Network (CAN) was developed mainly for the automobile industry, and is now popular in factory automation. It transmits at 1 Mbps and uses twisted-pair cable for up to 40 devices. Its main features are:

- Nodes can communicate when there are no nodes communicating on the bus.
- It uses a non-destructive bit-wise arbitration which allows fast detection of multiple accesses. This in turn allows full use of the bandwidth.
- It uses a message priority system, which is based on an 11-bit packet identifier.
- The architecture can have many masters, and involves peer-to-peer communications of multicast transmissions.

- It provides automatic error detection, signalling and retries.
- It uses short data packets of eight bytes.

Bus Values

The CAN bus defines two levels for bits on the bus. These are:

Dominant level. All nodes on the bus connect to the common bus. A dominant bit will always overrule a recessive bit. In a wired-AND implementation of the bus, a dominant level is represented by a logical zero and the recessive level by a logical one.

Recessive level. A recessive level on the bus is always overruled by a dominant level. If all levels are at a recessive level, the resulting level will also be recessive. In a wired-OR connection, a recessive level will be a logical zero (as a single logical one level will also make the output a logical one).

Error detection

A key component in creating a reliable bus is to provide strong error detection. The main method that the CAN bus uses for detection of errors is by monitoring bit levels. For this, transmitters check the transmitted level with the level on the bus and can quickly detect collisions on the bus. At a higher level, cyclic redundancy check, bit stuffing and message frame check are used.

The error detection scheme detects all global errors, all local errors at transmitters, up to five randomly distributed errors in a message, and burst errors of length less that 15 in a message. The resultant probability of undected errors is less than 4.7×10^{-11}.

The instrumentation industry has moved over the years from instrumentation networks made from dumb instruments, which reported back to a central controller, to smart instruments with distributed control. Fieldbus is really the first true implementer of totally distributed systems; but as the scope of the fieldbus is limited to specific areas, there is still need for a global control system (such as DCS). The fieldbus is excellent at allowing local control and parameter conversion, but is not so good at providing a global control strategy. This disadvantage is outweighed by reduced cabling costs, as the fieldbus and the devices easily connect to the bus.

7.9 LONWORKS: CONTROL NETWORK TECHNOLOGY ON A CHIP

LonWorks (local operation networks) is a control networking technology developed for a variety of control applications. LonWorks enables manufacturers to add local intelligence and communication capability to their products—from devices as simple as light switches and occupancy sensors to more complex devices such as PCs, PLCs and controllers.

LonWorks is a networking platform specifically created to address the unique performance, reliability, installation, and maintenance needs of control applications. The platform is built on a low-bandwidth protocol created by Echelon Corporation for networking devices over media such as twisted pair, power lines, fibre optics and RF. It is popular for the automation of various functions within buildings such as lighting and HVAC (Heating/Ventilation/Air Conditioning system).

LonWorks devices are intelligent. Because the devices are intelligent, LonWorks control networks do not need computers or master control panels to operate, and the devices communicate with each other peer-to-peer without any master communications panels or gateways. Thus LonWorks control networks have a flat, non-hierarchical architecture and no single point of failure.

In 2005 the European Community released the LonWorks-based EN 14908 building automation. China has also ratified the technology as a national control standard, GB/Z 20177.1-2006. Manufacturers in a variety of industries including building, home, transportation, utility, and industrial automation have adopted the platform as the basis for their product and service offerings.

7.9.1 LonWorks Technology

In LonWorks network, intelligent control devices called *nodes* communicate using a common protocol. Each node in the network contains embedded intelligence that implements the protocol and performs control functions. In addition, each node includes a physical interface that couples the node's microcontroller with the communication medium. The basic building blocks of LonWorks are:

- Neuron chip
- LonTalk communication protocol
- LonWorks transceivers
- Network management services.

Neuron Chips

The heart of each node is a neuron chip containing the LonTalk protocol. The neuron chip serves as the sole processor in most LonWorks nodes. If the application demands more processing or I/O power for the node, then the neuron chip can also be used as a communication coprocessor working with another host. The neuron chip thus provides a scalable solution.

In order to achieve economical and standardised deployment, Echelon designed the neuron chip. The name *neuron* was chosen to point out the similarities between proper network control implementation and the human brain. There is no central point of control in the brain. Millions of neurons are networked together, each providing information to others through numerous paths. Each neuron is typically dedicated to a particular function, but loss of any one does not necessarily affect the overall performance of the network.

The neuron chip is a system-on-a-chip with multiple processors, read-write and read-only memory (RAM and ROM), and communication and I/O subsystems. The read-only memory contains an operating system, the LonWorks protocol, and an I/O function library. The chip has non-volatile memory for configuration data and for the application program, both of which are downloaded over the LonWorks network. At the time of manufacture, each neuron chip is given a permanent unique-in-all-the-world 48-bit code, called the *neuron ID*. A large family of neuron chips is available with differing speeds, memory type and capacity, and interfaces. The neuron chip is actually three 8-bit inline processors in one. Two execute the LonWorks protocol; the third is for the device's application. The chip is, therefore, both a network communications processor and an application processor, significantly reducing the implementation cost for most LonWorks devices. The device manufacturer provides an application code to run on the neuron chip and I/O devices to be connected to the neuron chip.

A complete operating system including an implementation of the LonWorks protocol, called *neuron chip firmware*, is either contained in the ROM, or attached to, every neuron chip. Most LonWorks devices include a neuron chip, which has an identical embedded implementation of the LonWorks protocol. This approach eliminates the 99% compatibility problem and ensures that connecting LonWorks devices together on the same network requires little or no additional hardware.

LonTalk Communication Protocol

The LonTalk protocol is a complete seven-layer protocol that ensures that nodes can interoperate using an efficient and reliable communications standard. A variety of communication media such as twisted pair, power line, fibre-optic, coaxial cable, RF or infrared are supported under LonWorks.

The built-in communication protocol and processors removes the need for any development or programming in these areas. Referring back to the ISI/OSI reference model of a communication protocol, we see that the neuron chip provides the first six layers. Only the application layer programming and configuration needs to be provided. This standardises the implementation, which makes development and configuration relatively easy.

Two physical layer signalling technologies, twisted pair and power line carrier, are typically included in each of the standards created around the LonWorks technology. Additionally, the LonWorks platform also has an affiliated IP tunnelling standard, EIA-852, in use by a number of manufacturers to connect the devices on previously deployed and new LonWorks-based networks to IP-aware applications or remote network management tools.

LonWorks Transceivers

LonWorks transceivers provide a physical communication interface between the neuron chip and the LonWorks network. They are available for a variety of communication media and topologies. LonWorks control modules integrate neuron chip, communication transceiver, memory and clock in one compact module. Echelon, the designers of LonWorks, offers control modules both with and without transceivers. Echelon control modules include a transceiver and are Lonmark-compliant.

The GCM-10 (Generic Control Module) and LTM (LonTalk Module) do not have an onboard transceiver and can be used with any third-party or custom transceivers. Any microprocessor, microcontroller, PC, workstation, PLC, DDC or computer can become a node on the LonWorks network and thereby communicate with other LonWorks nodes. The optimum interfaces to these processors may be through either a parallel bus (ISA, VME or PC/104), a serial interface (RS-232-C), or even a phone line.

Network Management

Network interfaces can also serve as gateways to other control networks. LonWorks offers support for multiple media through routers. Routers can also be used to control network traffic and partition sections of the network from traffic in another area, increasing the total throughput and speed of the network.

The addressing, configuration and establishment of connections among LonWorks nodes are facilitated by a group of network management services built into every neuron chip. Some of these tasks may be performed at the time of manufacture, or they may be postponed until the network is commissioned in the field. A number of development tools are also offered as *LonBuilder workbench*. These tools include an environment for developing and debugging applications at multiple nodes, a network manager for installation and configuration, and a protocol analyser to examine network traffic and to debug errors.

Echelon, the manufacturer, has expanded the LonWorks technology with the development of LonWorks Network Services (LNS) and LonWorks Component Architecture (LCA). LNS and LCA facilitate full-range communication from the sensor to the data network personal computer levels as shown in Fig. 7.32. LNS takes advantage of client-server architecture, whereas LCA performs Object Linking and Embedding (OLE) for common database access. This ensures interoperability of data exchange among previously separated LonWorks networks. It also facilitates the linking of 1,024 million nodes in LonWorks.

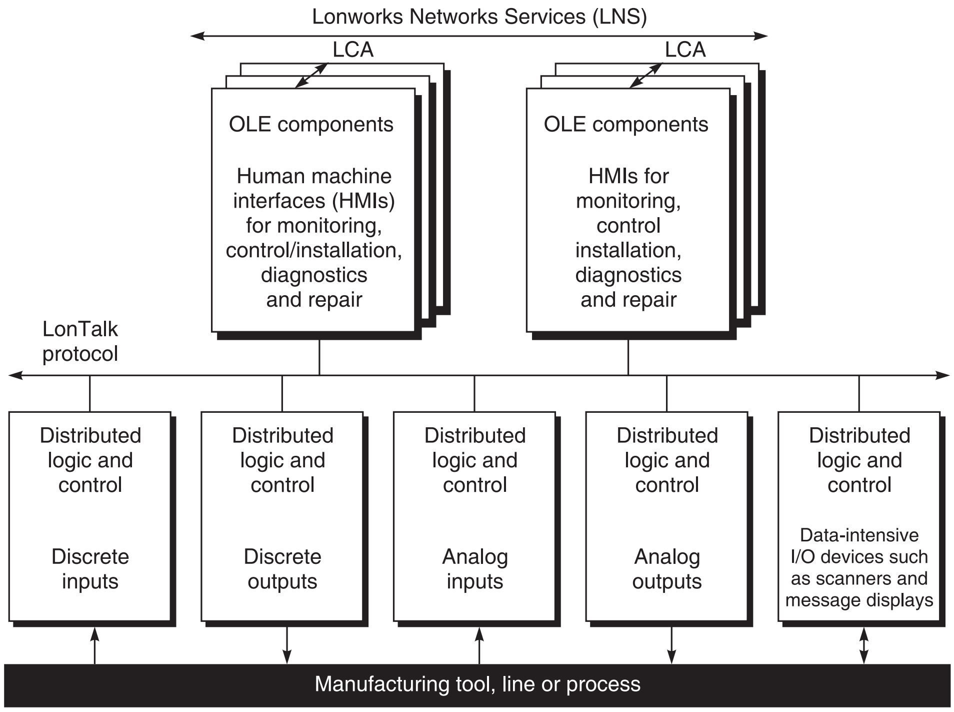


Figure 7.32 Extended LonWorks architecture.

7.9.2 Applications

LonWorks systems have been used in following application areas:

- Supervisory Control and Data Acquisition (SCADA)

- Material supply lines
- Semiconductor manufacturing
- Lighting control systems
- Energy management systems
- Heating/ventilation/air-conditioning (HVAC) systems
- Security systems
- Home automation
- Consumer appliance controls
- Public street lighting, monitoring, and control.

LonWorks technology has been incorporated into devices by many of the major construction industry vendors, including Belimo, Fugi Electric, General Electric, Honeywell, Hubbell, Johnson Controls, Philips Lighting, Siemens Building Technologies, Square D., Trane, the Wattstopper, and many others. Cisco Systems has collaborated with Echelon to develop a standard architecture to connect LonWorks networks to the Internet.

LonWorks technology has also been utilised in major construction projects in the United States, Europe, and Asia. LonWorks is used in the 54,000-seat Pusan Asiad Main Stadium, located in Pusan, South Korea; Enel's 47 million-home remote electric meter reading project; and the eastern region headquarters for the US Federal Aviation Administration, located adjacent to Kennedy Airport in Queens, New York.

The Tai Lake Basin supervisory system monitors and controls water floodgates in and around Shanghai, China, using LonWorks automation hardware, Echelon i.LON 100 Internet Servers, and Broadwin Web Access software. The Supervisory Control and Data Acquisition (SCADA) system monitors 36 water floodgates using 13 WebAccess SCADA nodes communicating to Echelon i.LON 100 Internet servers for remote monitoring and control from a central control centre via a secure WAN. The floodgate sites are normally unmanned, but the system also allows local monitoring and control.

The Quinghai–Tibet railway, the world's longest high-altitude railway, is using Echelon's technology to monitor and control various systems, including a state-of-the-art oxygen supply system for passengers. Reliability and proven performance are necessities in any public transit system. In order to keep passengers from suffering from altitude sickness, pressurised cars and special train engines that can function with little oxygen are used to reduce the effects of the high altitudes. Echelon's technology is used in the control network to monitor the oxygen supply system, which provides individual oxygen supply for passengers at high altitudes and is integrated with the train's traditional heating, ventilation and air-conditioning (HVAC) system. In addition, Echelon's technology is used to monitor braking, door, lighting, and power supply systems for safety and environmental monitoring purposes.

Sifang Rolling Stock Research Institute, China, and the China Academy of Rail Sciences have created a standardised system on Echelon's LonWorks networking platform for safety monitoring on next-generation passenger trains in China, effectively making it a de facto standard for high-speed rail travel in the world's largest passenger rail transport market. Echelon's LonWorks networking platform monitors safety features on over 4,500 high-speed passenger train cars, including brakes, doors and power supply.

The American Association of Railroads is utilising the LonWorks networking platform in stations for remote monitoring and control of track switching mechanisms. Electropneumatic braking systems also utilise Echelon's power line signalling technology and protocol.

7.10 CONCLUSIONS

Distributed Control Systems provide a very convenient method of distributing computing power to its point of use. Hierarchical distributed processing systems are particularly suitable for implementation in manufacturing companies. The advantages of this particular approach include progressive build-up of such systems over a period of time and provision of computer power to all the users.

SUGGESTED READING

Bryan, G.C. and Umbers, I.G., *Distributed Systems for Process Monitoring and Control,* Woren Spring Laboratory, Stevenage, Hertfordshire, UK, 1982.

Erickson, I.L., Centralized vs. distributed process control, *ISA Int. Conf.,* Houston, Texas, 1980.

Erickson, I.L. and Purvis, J.R. (III), Centralized vs. distributed process control: A cost comparison, *Intech,* **28,** No. 6, pp. 61–63, 1981.

Krigman, A., Distributed Control: Pipe dreams to reality, *Intech,* **31,** pp. 7–81, 1984.

Krishnamurthy, J., Trends in distributed control, *Proc. Of 4th Ann. Contr. Engg. Conf.,* pp. 279–500, 1985.

Lukas, M.P., *Distributed Control Systems: The evaluation and design,* Van Nostrand Reinhold, New York, 1986.

McWilliams, P., Low level interface aids distributed intelligence, *C & I,* pp. 58–59, 1986.

Polock, W.K., Evaluating distributed control system—A generic approach, *Proc. 3rd Ann. Contr. Engg. Conf. Control Engineering,* pp. 121–124, 1984.

Shoffler, J.D., Distributed computer systems for industrial process control, *IEEE Computer,* **17,** pp. 11–18, 1984.

Tinham, B., Getting a perspective on distributed control, *C&I*, pp. 62–122, 1987.

Vandiver, R.L., What is distributed control, *Hydrocarbon Processing,* **60,** No. 6, pp. 91–94, 1981.

William, T.J., Distributed digital computer-based industrial control systems in the western world and Japan: A status report, *Int. Sem on Distr. Contr.,* 86, New Delhi.

Young, R.C., *A Distributed Process Computer Control Networks,* JACC, San Francisco, 1980.

Zimmerman, C.K., Evaluating distributing control systems: Where do we go from here, *Contr. Engg*., **31,** No. 10, pp. 109–112, 1984.

CHAPTER

8

Real-time Programming

8.1 INTRODUCTION

Those of us who are proficient in programming using FORTRAN, COBOL, ALGOL, PASCAL etc. for applications in problem-solving may find real-time programming less familiar. In this chapter we shall deal with various aspects of real-time programming.

The most important aspect of a real-time programming system is the management of information flow. Figure 8.1 show the various basic modules of a real-time system along with information flow between them.

8.1.1 Input Sub-system

Input sub-systems are used to measure different parameters like temperature, flow, pressure etc. or status of limit switches, ON/OFF switches, and other binary status information.

Analog sensor

Also called transducers or transmitters. They present at their output an electrical signal whose value is proportional to the value of parameter being measured.

Digital sensor

These sensors have Analog to Digital Converter (ADC) embedded in the same package in addition to analog sensor. At their output they present a digital value proportional to the parameter value.

Binary switches

They present one-bit value corresponding to ON/OFF status, limit switches at various parts of process.

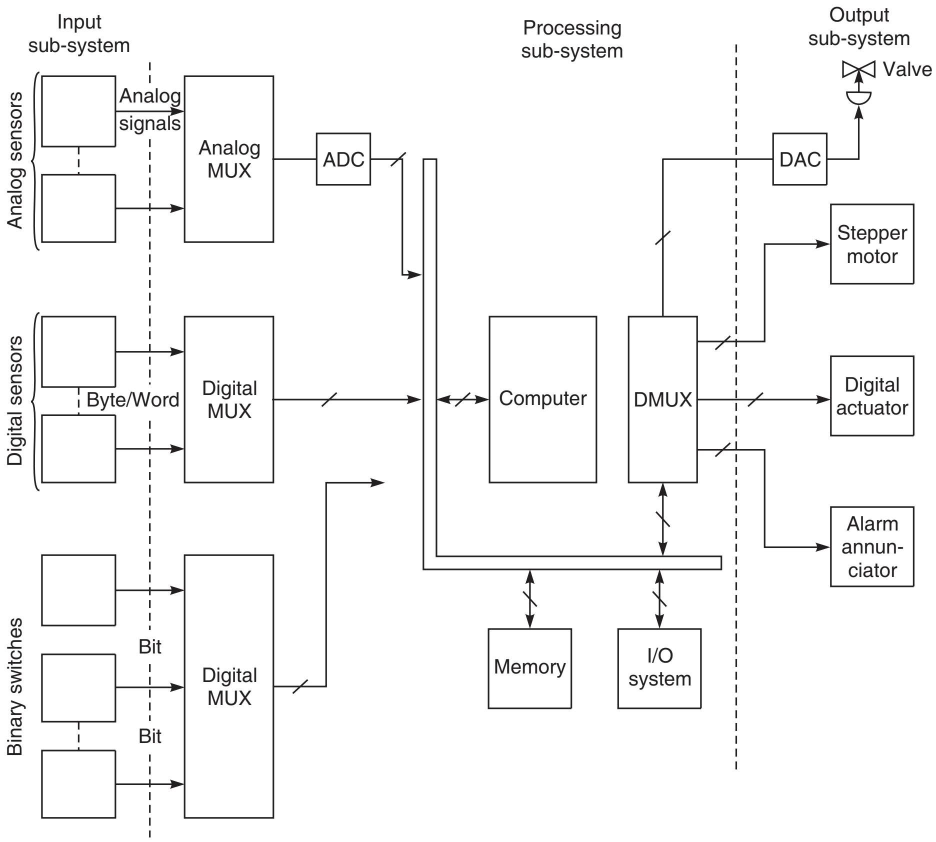


Figure 8.1 Information flow in real-time system.

8.1.2 Processing Sub-system

The computer and associated modules connected to computer through bus constitute the processing sub-system. The associated modules include analog multiplexer, digital multiplexer, ADC, memory, DAC, demultiplexer and I/O modules like console, mimic panel etc. The computer receives information in byte/word form from analog sensor through ADC or directly from digital sensor. The binary switches send the information in bit form which is stored in the computer memory.

Depending on the parameter value, set-point, control loop configuration and control algorithm, the correction value is determined and is sent to output valve through DAC. The stepper motor and digital actuators accept control correction value in digital form. The alarm annuciators have separate modules for individual alarms. Thus each requires only one bit information to activate.

8.1.3 Output Sub-system

The control valve and actuator take analog signal which may come from computer through DEMUX and DAC. The stepper motor needs digital information for each step movement. Similarly digital actuator takes byte/word for control correction. The alarm annunciator may be connected directly and requires only activation signal which may come in bit form.

8.1.4 Information Processing

The computer executes the control algorithm on the information received from analog and digital sensors. Thus the processing performed is

INPUT (VALUE)
ERROR = (SET-POINT – VALUE)
RUN CONTROL LOOP 1 (ERROR, CORRECTION)
OUTPUT (CORRECTION) TO VALVE (N)

Thus the information is received, stored, manipulated and the output is given in byte/word form. On the other hand, the status information is in bit form. The type of processing required is

If (SW1 is ‘ON’ AND (SW2 is ‘OFF’ OR SW5 is ‘ON’))
then SEND ALARM 1
else
If (SW1 is ‘ON’ AND SW2 is ‘ON’)
then RUN CONTROL LOOP-5 (ERROR, CORRECTION)
else Repeat

Thus computer must be able to read the status of bits directly (Bit Addressing), check their status and manipulate (Bit manipulation) and take decisions.

8.1.5 Interrupts

Interrupt processing is the most important feature of real-time systems which distinguishes these from on-line systems. Interrupt processing is directly related to ‘response time’ of the real-time systems. Too many interrupts and slow processing will ultimately affect the main job and result in sluggish performance of system. At the same time efficient interrupt processing, together with well-defined priority structure will lead to better performance. The priority structure defines the following features:

1. In case of two or more interrupt requests at the same instant, which one will have priority and will be processed.
2. Whether a higher priority interrupt request can interrupt a lower priority interrupt?
3. The mechanism by which different interrupts can be disabled temporarily or masked.
4. Interrupts which cannot be masked.

The factors which mainly affect the interrupt processing (Fig. 8.2) are context switching and interrupt latency. *Context switching* is defined as time and overhead to switch between two tasks or routines when an interrupt has been accepted for processing. This includes not only the time to load program counter with new address but also time to save the status variables. This depends on the computer architecture as well as interrupt management structure.

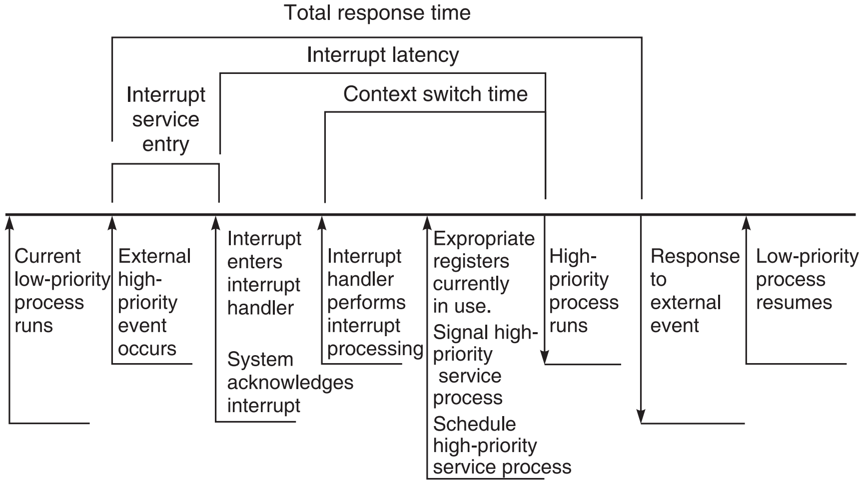


Figure 8.2 Interrupt processing steps.

Interrupt latency is time lag between the acknowledgment of interrupt to the start of execution of interrupt servicing routine.

Considering the importance enjoyed by interrupts in real-time systems, its careful treatment forms an important and integral part of red-time systems.

8.1.6 Real-time Programming

A real-time system can be broken into a number of independent sub-systems called processes. Figure 8.3 shows the following five independent processes which act independently but share resources to monitor and control the plant parameters:

— Data collection (Process A)
— Limit checking and engineering unit conversion (Process B)
— Alarm generation (Process C)
— Control algorithm execution (Process D)
— Output signal generation (Process E)

These are functionally independent but communicate with each other, either through data bus or memory data buffer.

A process may therefore be defined as a hardware or software or combined sub-system designed to perform defined tasks. A process is therefore having local processing capability or local intelligence.

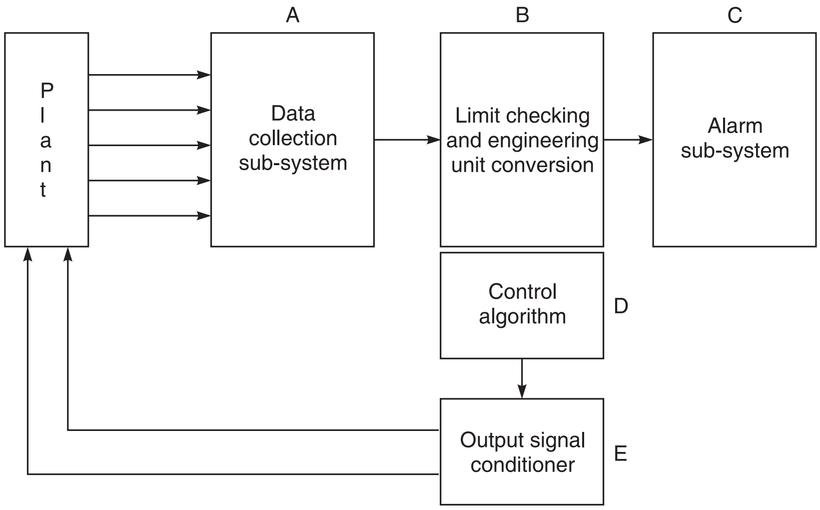


Figure 8.3 Process and tasks.

As in Fig. 8.3 Data Collection (A) may be implemented using fixed or programmable hardware (microprocessors). It collects regularly the analog and digital data from the plant, converts these into appropriate form, filters the noise and transfers to (B) Limit Checking Process, using data bus or common data path.

The Limit Checking Process performs limit checking and engineering unit conversion functions and transfers the data to Alarm Generation Process (C) and Control Algorithm Execution Process (D). Alarm Generation has been conceived as a separate hardware to which Limit Checking Process transfers the data on data path. On the other hand, Control Algorithm Execution Process is implemented on the same hardware as Limit Checking Process. The reason is evident; both limit checking and control algorithm execution processes are software sub-systems and the data is transferred between these using memory data buffer.

Finally control algorithm execution process transfers the data on correction value to Output Signal Generation Process (E) which converts these into analog, conditions them and then sends to the control valves in the plant.

8.2 MULTI-TASKING

A process may be defined as a number of tasks which may be executed on the same hardware simultaneously. A task is defined as a logical entity which contains a part of the process functions and the conditions under which these will be invoked and conditions under which these process functions will end or carry on repetitively.

As an example, the process (B) Limit Checking may contain the following tasks:

Task B1 : Storing the values sent by process (A). This task must be initiated after every 2 seconds.

Task B2 : Calculation of average of last 4 values for each parameter. This task may be executed simultaneous to B1.

Task B3 : Comparison of average parameter values with set-points and calculation of error limits. Comparison of average parameter values with high and low limits for alarm generation.

Task B4 : Communication with process (C) on limit violation.

Task B5 : Communication with process (D) on deviation from set-point (error).

An ideal situation is the one in which none of the tasks are competing with each other for hardware resources. In this situation, the tasks run sequentially and also meet the response time criteria. Figure 8.4 shows the ideal situation in case of process (B).

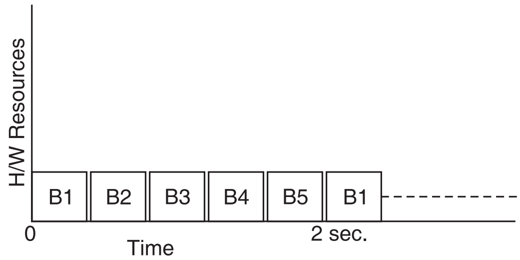


Figure 8.4 Sequential task execution.

In this case, each task may be thought of as a subroutine in FORTRAN or procedure in ALGOL, PASCAL etc. and the process invokes these sequentially to complete one cycle. The repetition of cycles is achieved through IF-THEN-ELSE loop.

```
          Process B
          Variable Definition
10        PCLOCK = CLOCK;          Clock is current time implemented as output of
          CALL     Task B1         timer which is continuously changing. PCLOCK
          CALL     Task B2         is value of clock at previous sample.
          CALL     Task B3
          CALL     Task B4
          CALL     Task B5
20        If CLOCK < (PCLOCK + 2) THEN GO TO 20
          else Go to 10
```

This looks very simple and elegant. Then why do not we implement this approach?

To meet the timing requirement under peak demand we may require extensive hardware resources. This may turn the whole application economically infeasible. If a hardware is not fast enough to satisfy the timing requirement when tasks are executed sequentially, then one is left with following strategies:

1. The tasks compete with each other to share the hardware resources. They are executed simultaneously to optimally use the resources and also to meet the timing requirement.
2. The tasks are assigned priorities on the basis of which selection of task to use the hardware resources is done. This is also to satisfy the timing requirement of important tasks. This is known as multi-tasking.

In the non-ideal situation the sequential task execution may look like as given in Fig. 8.5.

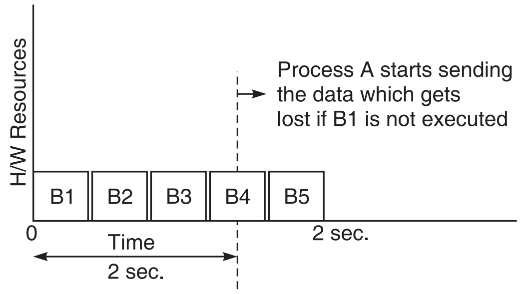


Figure 8.5 A non-ideal situation.

The logic for simultaneous execution of tasks on the same hardware (multi-tasking) is same as multi-programming. In multi-programming, a program may be suspended if, (a) I/O or memory is not available or (b) the program has some idle time (e.g. a delay loop) on computer which can be used by other program.

Similarly in multi-tasking a task may be suspended if,

1. I/O resources are not available;
2. Task itself cannot proceed for want of some data which are to be transferred by other task;
3. There is delay in the execution of task;
4. A higher priority task is ready for execution.

Let us examine if we can improve the situation in Fig. 8.5 by using multi-tasking. We may safely assume that in each task there is a finite idle time during which CPU 'waits' for an event to occur. The multi-tasking uses this CPU time to execute other tasks. As an example, in Task B1, the event starts on receipt of Timer pulse of 0,2, ..., seconds. Then CPU is made to wait for a pulse from process A, indicating that the data is ready. The task B1 reads the data and stores in memory. It then again waits for pulse from process A. On receipt of fifth data, the task should terminate but action must be taken to re-initiate it on next timer pulse.

We assume that CPU has to wait under task B1 for 100 milliseconds for pulse to come from process A, then one of the two cases shown in Figs. 8.6 and 8.7 may be true. In case I, it is

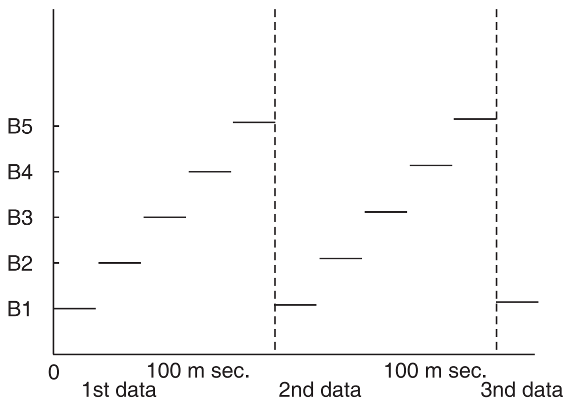


Figure 8.6 Multi-tasking—Case I.

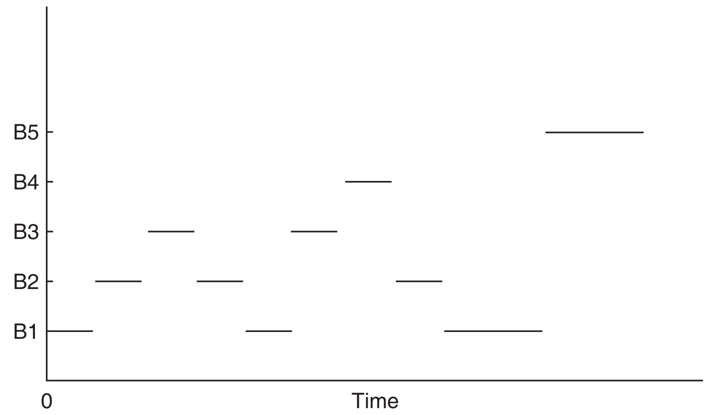


Figure 8.7 Mult-tasking—Case II.

assumed that during idle time of task B1 the CPU may complete all other tasks. Thus in 500 milliseconds one cycle of process B is complete leaving remaining part of 2 seconds.

In case II, no such assumption has been made. Thus the execution of tasks is at random, based on the priority and their readiness.

8.2.1 State Transition Diagram

In multi-tasking, a task goes through a number of states during its execution. This is because the CPU at any instant is executing only one task, whereas other tasks may be suspended, inactive, scheduled or ready to be executed. A task may go through the following different states:

— Inactive
— Scheduled
— Ready
— Running
— Suspended.

Inactive

A task is in *inactive* state when it is not in the reckoning for execution. A running task when terminated at the end of execution goes to inactive state.

Scheduled

A task which is *scheduled* to run at a specified time (which may be ‘now’ also) is in scheduled state. The task is under the control of timer/counter (clock program).

Ready

A task is in *ready* state if all the basic requirements for its execution have been fulfilled while the CPU is currently executing a task which has higher priority.

Running

A task which is being currently executed is in *running* state. It means that this is the highest priority task, out of all the ready tasks.

Suspended

A task is in *suspended* state if its execution has been suspended due to inherent delay in the task or when further processing is not possible as it requires data from other task or I/O unit.

The state transition diagram of tasks is shown in Fig. 8.8. The transition between different states of tasks is decided and done by task management sub-system of operating system. Some real-time programming languages like PEARL also give these facilities at programmer's level.

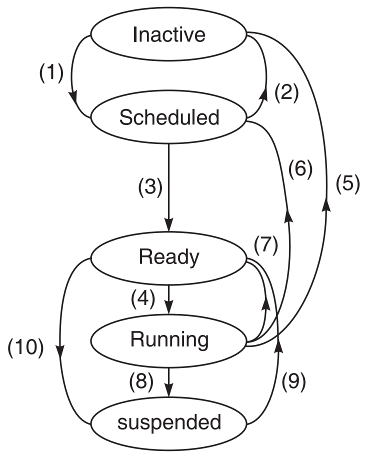


Figure 8.8 State transition diagram of tasks.

A brief explanation of state transition diagram is given here:

Inactive to scheduled

A task in the inactive state can shift to scheduled state when it is activated by declaring its schedule by another task.

Scheduled to inactive

A scheduled task becomes inactive when it is prevented by other task. In such a case all the pending requests for its activation are cancelled.

Scheduled to ready

A scheduled task becomes ready when all the preconditions for its execution have been satisfied. This transition is automatically done by operating system (O/S) when it finds that the execution of a task is due.

Ready to running

This transition is done by O/S depending upon the priorities of ready tasks and priority of running task. When the *execution* of a task is completed, the O/S selects the task having highest priority from ready tasks and makes it *running*. When a task having higher priority than that of running task is made ready then the running task is made ready and highest priority task is made *running* by initiating its execution.

Running to inactive

A running task when terminated goes to inactive state. The termination of running task may occur either when the execution is completed or when the program goes to some fatal error condition.

Running to scheduled

A task while running may reschedule itself on completion for the next cycle. The task will again be executed at the scheduled time.

Running to ready

When a task having priority higher than currently running task is made ready, it necessitates that the running task is made ready and higher priority task is made to execute.

Running to suspended

When a task executes a delay statement of the type "DELAY BY 20 SECOND, it is suspended and another task having highest priority from ready tasks is made to execute. Similar action is taken when O/S finds that the execution of task cannot proceed for want of data which is supposed to come from another task or I/O unit.

Suspended to *ready*

A task which was suspended previously, transits to ready state when it becomes ready to be executed again, i.e.

- — when delay time has elapsed, or
- — when data required by the task has been received, or
- — when command is received from running task to resume a suspended task

Ready to suspended

A running task may suspend a ready task when it is found that the execution of the latter is not required presently. Thus the ready task which may have been executed partially goes to suspended state. This task can come out of suspended state by command to resume from any running task.

8.3 TASK MANAGEMENT

Tasks need to be defined by the following attributes for effective management:

- — Task name
- — Task status
- — Specification of code to be executed
- — Address in memory of code
- — Specification of data to be processed
- — Initial status of processor
- — Intermediate status of processor when task was suspended
- — Pointer to another task descriptor.

The information can be put in two categories, viz. *static information* which does not change as the execution of task progresses and *dynamic information* which undergoes change with the execution of task. The various attributes of static and dynamic informations regarding the task are tabulated here.

Static information	*Attributes*
Task name	May be represented in the same way as variable name.
Specification of code	May contain, memory requirement, segments, I/O requirements. It may also contain the names of tasks from/to which it will receive/send information.
Address in memory of code	Starting address in memory of the task.
Specification of data	Data may be byte/word/double word, integer or floating point, characters as message for I/O or for processing.
Initial status of processor	Values in various registers in the processors including flag register and program counter.

Dynamic information	*Attributes*
Task Status	Inactive, Scheduled, Ready, Running, Suspended.
Intermediate status of processor when the task was suspended	Values in various registers in processor including flag register, program counter and stack pointer.
Pointer to another task	This is address of another task in task descriptor table having same status.

8.3.1 Task Descriptor Table

Figure 8.9 shows the structure of the task descriptor table. The structure is a linked list with forward pointers. The address of first five tasks of different status are stored separately, in the list header. Thus we have the following tasks in list of header addresses:

— 1st *Inactive* Task in Task Descriptor Table
— 1st *Scheduled* Task in Task Descriptor Table
— 1st *Ready* Task in Task Descriptor Table
— Only *Running* Task in Task Descriptor Table
— First *Suspended* Task in Task Descriptor Table.

In each task descriptor we have a pointer to another task descriptor stored. Using these pointers we can create four chains one for each type of tasks. The task having 0 (zero) as pointer to another task is the last element in the chain. Figure 8.10 explains the data structure through an example.

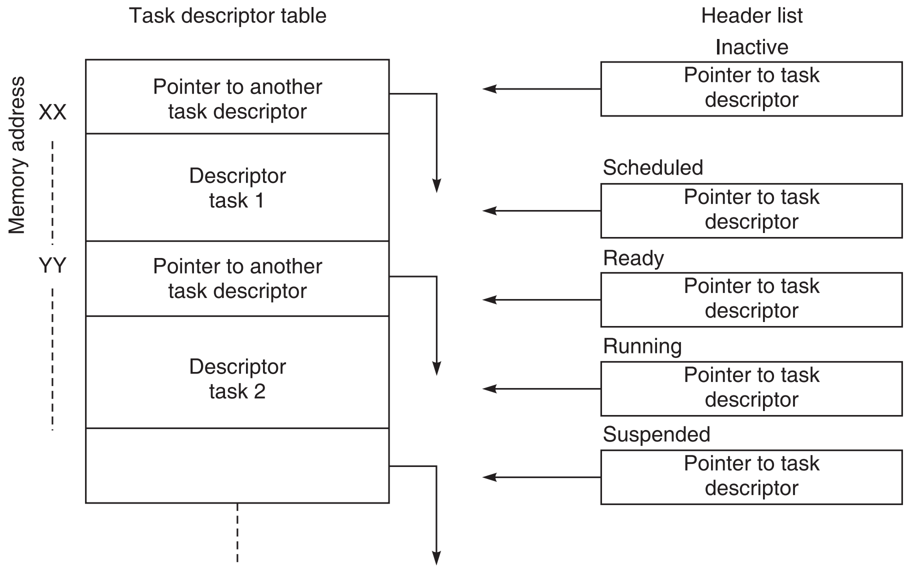


Figure 8.9 Structure of task descriptor table.

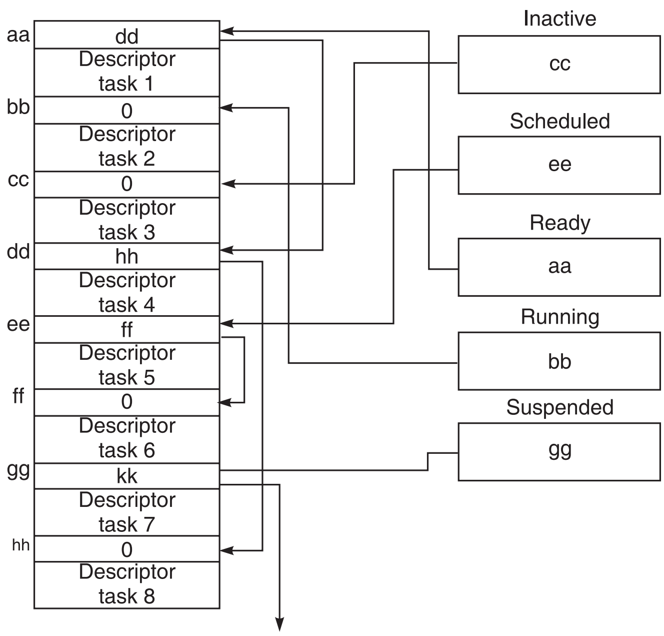


Figure 8.10 Task descriptor table—An example.

It must be noted that at any instant there will be only one running task in the system. The task descriptors can be added or deleted by changing the contents of pointers.

8.3.2 Deadline vs. Priority

This must make an interesting debate. So far, the importance of task is reflected by a number called *priority number*. The convention normally followed (though not sacroscant) is that the higher the priority number, the lower is the priority of the task. Thus the highest priority task will have priority number as zero whereas lower priority tasks will have higher priority numbers.

The priority structure is that if a task 'T_1' of priority number P_1 is presently running and task 'T_2' of priority P_2 ($P_2 < P_1$) has become ready then:

1. Context of T_1 is stored
2. T_1 is put in *suspended* tasks. The pointers in task description table are changed to delete task T_1 from *running* and add it in *suspended task* chain
3. Task T_2 is executed. Task T_2 status is changed and it is put as *running task* in task description table.

But what are the benefits of this approach?

It is very straight-forward, uncomplicated procedure where higher priority jobs are allowed to be completed than lower priority ones. But it falls short of basic requirement. Earlier, we have stated that logical validity of any process or task is related to the stipulated time by which it has to be completed. If not completed by that time schedule the task or process is not considered valid. Thus in real-time system, we are interested to see that all the tasks are completed on schedule or at least average delay caused is minimised. This obviously is not guaranteed by priority scheduling mechanism.

Let us therefore try to incorporate the time element called *deadline*, for each task (Fig. 8.11). We shall try to do deadline scheduling for the task and examine the advantages and drawbacks of this approach against the priority scheduling approach.

Task	Activation time	Processing time	Deadline	Virtual deadline
T_j	r_j	t_j	d_j	d_j^*

Figure 8.11 Deadline scheduling.

It is necessary to define the following terms to understand deadline scheduling:

Activation Time *(r_j)* is time schedule for activation of task-j, i.e. the time when the execution of task j should begin.

Processing Time (t_j) is time that would be required to execute the task-j.

Deadline *(d_j)* is time by which the execution of task-j must be completed

Virtual Deadline ***(d_j*)*** is minimum deadline of task-j. It is determined by taking minimum of deadline, and deadline calculated considering that all higher priority tasks have been processed before.

$d_j^* = \min (d_j, \{d_k : T_j < T_k\})$

$T_j < T_k$ means T_j has higher priority than T_k and must be completed fore T_k.

Algorithm

1. For every task T_j determine virtual deadline d_j^*.

   $d_j^* = \min [d_j, \{d_k : T_j < T_k\}]$
2. Assign CPU to ready task T_j which has minimum value of d_j^*.
3. Process T_j, till completion or till task T_l with $d_l^* < d_j^*$ is activated and is ready to preempt task T_j
4. If $d_j^* = dj^*$ with $i < j$ then choose T_i
5. Repeat till all the tasks are scheduled.

Example: As an example, let us try to apply the deadline scheduling algorithm on the task set shown in Fig. 8.12. The task set contains thee tasks with different processing time, deadline and activation time.

Task	Processing time	Deadline	Activation time
T_j	t_j	d_j	r_j
1	4	8	0
2	3	6	3
3	3	10	7

Figure 8.12 Task set.

Let us analyse two cases: (a) when the tasks are non-preemptable and, (b) when the tasks are preemptable.

The virtual deadline for the three tasks will be same as for deadline mentioned.

Case I: Tasks are non-preemptable

(i) Time = 0

Task 1 is executed

(ii) Time = 3

Task 2 is made ready. Though deadline for Task 2 is less than that for Task 1 but since the tasks are non-preemptable, no action is taken.

(iii) Time = 4

Task 1 is completed. Task 2 is made running.

(iv) Time = 7

Task 3 is made ready. Task 2 is completed. Task 3 is made running.

(v) Time = 10

Task 3 is completed.

In this case Task 2 is delayed as shown in Fig. 8.13.

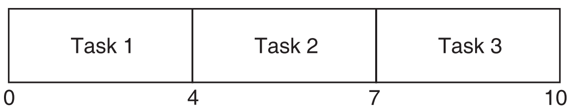


Figure 8.13 Non-preemptable solution.

Case II: Tasks are preemptable.

(i) Time = 0

Task 1 is made running.

(ii) Time = 3

Task 2 is made ready. The deadline for Task 2 is less than deadline for Task 1. That is why, Task 1 is made ready; Task 2 is made running.

(iii) Time = 6

Task 2 is completed. There is only one ready task (i.e. Task 1) and it is made running.

(iv) Time = 7

Task 3 is made ready.

Task 1 is completed.

Task 3 is made running.

(v) Time = 10

Task 3 is completed

There is no delay in this case (Fig. 8.14).

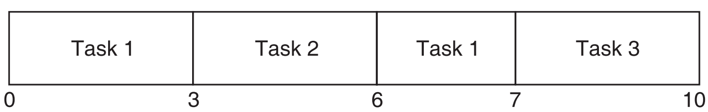


Figure 8.14 Preemptable tasks.

The deadline scheduling algorithm though looks straight-forward and simple at first instant, gains complexity as number of tasks increases. The idea however is to minimise the delay suffered by tasks. In many cases the feasible solution does not exist which renders the algorithm useless. In such cases, priority scheduling or any other scheduling mechanism must be specified.

8.4 INTER-TASK COMMUNICATION

Inter-task communication is achieved through following means in a real-time operating system:

— Mailboxes
— Semaphores
— Regions.

Since communication between two or more tasks is very important in multi-tasking system, we shall deal with these concepts in detail.

8.4.1 Mailboxes

In inter-task communication through mailbox, the *first task* will send a token for an object to the mailbox and the *second task* will go to the mailbox to receive the objects token. The object may be a segment that contains data needed by the waiting task. Figure 8.15 depicts the mailbox

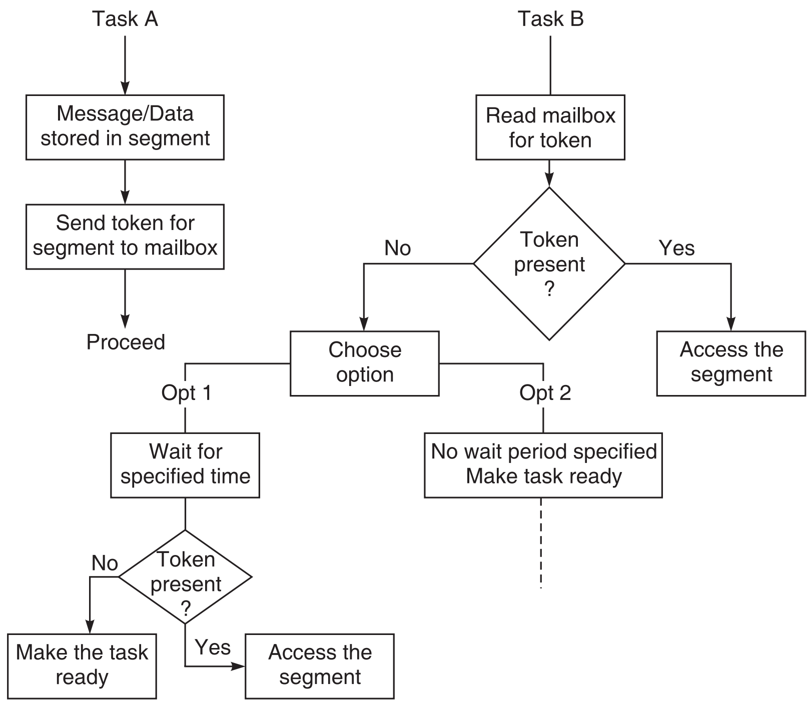


Figure 8.15 Functioning of mailbox.

function for inter-task communication, using flow chart. After receiving the token, task B can access the object. Segment can store the data that is to be passed from one task to another. The segment may be blank and its transfer to waiting task may act as signal of completion of some portion of the task. This may be used for mutual exclusion. If a task visits the mailbox and does not find the token then it has two options:

1. The task may opt to wait for specified time, after which it visits mailbox again and if the token is still not present then the task executes further.
2. The task is not willing to wait then it executes further.

What we have described above is mailbox scheme in iRMX-86. Each mailbox has two queues, one for the tasks waiting to receive the objects and another for objects sent by the tasks but not yet received.

8.4.2 Semaphores

A semaphore is a structure with a semaphore counter (S) which is a nonnegative integer and a FIFO queue for waiting processes. The semaphores are used for process synchronization, mutual exclusion and protection of critical sections. The two operations that are defined on semaphore are:

— Wait (S)
— Signal (S)

Where S is semaphore counter. The explanation of these operations is as follows:

Wait(S)—Wait for semaphore.
Semaphore may be considered as a signal from another task. In this operation, the task executing *wait*, is basically waiting for the signal from other task.

If S = + ve, then signal has already come. The task resets the signal ($S \leftarrow (S - 1)$) and proceeds execution,

If S = 0, then signal is not received. The task is suspended and placed in the semaphore queue, until woken up by 'signal' operation.

Signal(S)—Send semaphore signal.
The semaphore queue is checked for any task which may be waiting for semaphore signal.

1. If semaphore queue is empty then the task to which the semaphore signal had to go, has not executed 'wait' operation as yet. The semaphore signal is placed ($S \leftarrow S + 1$) and the task proceeds.
2. If semaphore queue is not empty then one or more tasks are waiting for semaphore signal. One of the waiting tasks (depending on priority or any other criteria) is made runnable and then the task proceeds.

8.4.3 Region

The *region* is a mechanism, by which a task can guard the simultaneous use of shared data. Each task requiring access to shared data has to wait for its turn at the region associated with that data. When a task is allowed to access the shared data, no other task can access that data unless the previous task releases control, even though the tasks may be running concurrently. This is done basically to protect the data from *corruption*. Region ensures that

— only one task has the access to shared data at any instant; and

— task having access cannot be preempted by other tasks desiring access.

8.4.4 Example

To illustrate the utility of these means for inter-task communication, let us consider process (B) in Fig. 8.3. The five major tasks in the process have already been defined. In this illustration we consider two tasks only.

Task B1— Storing the values sent by process (A), at an interval of 2 seconds.

Task B2— Calculation of average of last 4 values for each parameter; comparison of average parameter values with set-points and calculation of error; comparison of average parameter values with high and low limits for alarm generation.

The task B1 here is same as task B1 in Section 8.2, whereas task B2 here is combination of task B2 and task B3. The algorithm for task B1 and task B2 will be:

Task B1

```
begin;
I = 0
Read J
Repeat until I > 5
begin;
I = I + 1
Input VALUE
BUFFER (I, J) = VALUE
end;
J = J + 1
If J ≤ 4 then
write J
else
begin;
J = J – 4
write J
end;
end;
```

Task B2

```
begin;
K = 0
Repeat until K ≥ 5
begin;
If K < I then
K = K + 1
Process BUFFER (K, 1)
end;
end;
```

The ideal situation will be the sequential execution of task B1 followed by task B2 and other tasks. But as discussed in the previous section, this is not possible for the time limit of 2 seconds is very short. Thus only solution possible is to execute these tasks concurrently.

When these tasks are interleaved arbitrarily, a number of peculiar situations may arise. Considering the relative speed of these two tasks, both the tasks may take same time or one may take more time than the other. The variable I is being used in both the tasks. Similarly array BUFFER is accessed in both the tasks. However these are manipulated only in task B1. In order to achieve the correct result, it is imperative that task B1 is invoked after every 2 seconds gap and also, the task B2 is invoked only after most recent samples are available. This is to ensure that average for all the parameters is time-uniform. Consider a case where some problem has erupted in the plant and the values of some parameters have suddenly increased. If time uniformity is not ensured, then there is likelihood of initiating the corrective action for some parameters without taking into consideration the other parameters. It may lead to disaster in cascade control system. This problem is known as *Task Synchronization.*

Considering that both the tasks can preempt each other, let us see the following execution of these tasks and the situation emerging out of this.

Task B1	**Task B2**
:	:
:	$K = 2$
$I = 1$	
	Suspended
$J = 4$	
$I = I + 1 = 3$	
Input VALUE	
Suspended	Invoked $K < I$ Yes
	$K = K + 1 = 3$
	Process BUFFER (3,1)

BUFFER (3, p; p = 1, 4), contains values from previous sample.

The genesis is that task B2 is executed faster than task B1. If task B2 takes more time than task B1, then this problem will not arise.

Solution:

(i) *Mailbox scheme.* Using two operations Receive Mail and Send Mail, the mailbox scheme has been illustrated here:

Task B1	**Task B2**
begin;	begin;
$I = 0$	$K = 0$
Read J	Repeat until $K \geq 5$
Repeat until $I \geq 5$	begin;
begin;	Receive mail (Token)
$I = I + 1$	If $K < I$ then
Input VALUE	$K = K + 1$

```
BUFFER (I, J) = VALUE
Send mail (Token)
end;
J = J + 1
If J ≤ 4 then
Write J
else
begin;
J = J – 4
Write J
end;
end;
```

```
Process BUFFER (K, 1)
end;
end;
```

When send mail (Token) is encountered in task B1 then token is sent to the mailbox and task B1 proceeds. In task B2, the receive mail (Token) looks for a token in mailbox and task is forced to wait for the token in queue till the same is sent. Thus task synchronization is accomplished.

(ii) *Semaphore scheme*. The two operations, signal and wait described earlier are used to illustrate the solution of task synchronization problem using semaphore. Initially SEMA = 0.

Task B1

```
begin;
I = 0
Read J
Repeat until I ≥ 5
begin;
I = I + 1
Input VALUE
BUFFER (I, J) = VALUE
Signal (SEMA)
end;
J = J + 1
If J ≤ 4 then
Write J
else
begin;
J = J – 4
Write J
end;
end;
```

Task B2

```
begin;
K = 0
Repeat until K ≥ 5
begin;
wait (SEMA)
If K < I then
K = K + I
Process BUFFER (K, 1)
end;
end;
```

Similar to mailbox scheme, task B2 is forced to wait for semaphore signal from task Bl before it can proceed further. The part execution of tasks B1 and B2 along with semaphore is shown in Fig. 8.16. Using semaphore, task B2 is prevented from running faster than task B1.

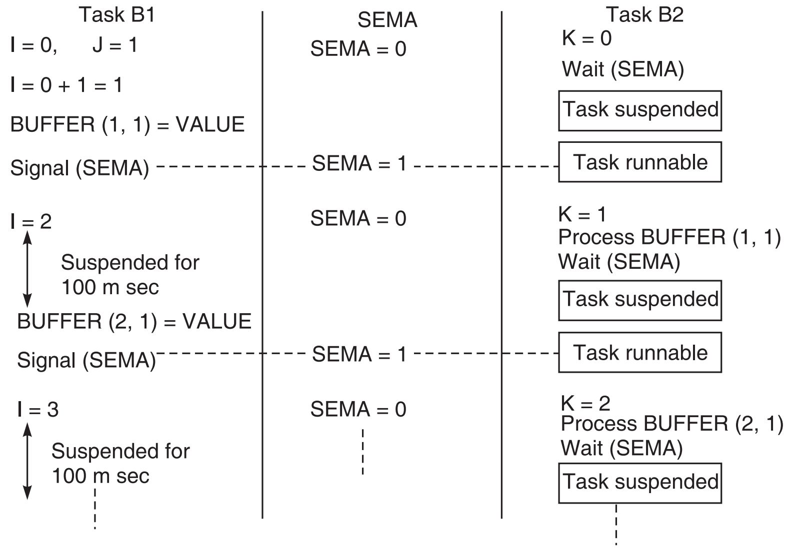


Figure 8.16 Semaphore for task synchronization.

(iii) *Regions scheme.* By declaring variables I and array BUFFER as shared data between tasks B1 and B2 and controlling its access by regions, the task synchronization can be achieved. However, it must be noted that these data are manipulated only in task B1 and therefore, are not shared in true sense. The two operations Receive Control and Send Control will illustrate the regions concept and its use to solve this problem.

Task B1

begin;
$I = 0$
Read J
Repeat until $I \geq 5$
begin;
Receive control
$I = I + 1$
Input VALUE
BUFFER (I, J) = VALUE
Send control
end;
$J = J + 1$
If $J \leq 4$ then
Write J
else
begin;
$J = J - 4$
Write J
end;
end;

Table B2

begin;
$K = 0$
Repeat unit $K \geq 5$
begin;
Receive control
If $K < 1$ then
$K = K + 1$
Process BUFFER (K, 1)
Send control
end;
end;

Let us now discuss the problem of shared data between two or more tasks. To illustrate the problem, we will re-define the tasks B1 and B2.

Task B—Store the values of five parameters sent by process (A) at every 2 second interval. Add the present value to the total.

Task B2—Take the average of four values. Initialise total and counter.

Algorithm

Variable — array TOTAL (5) — Total number of parameter values
— array COUNTER (5) — Total number of values

Task B1	**Task B2**
begin;	begin;
$I = 0$	$K = 0$
Repeat until $K > 5$	Repeat until $I > 5$
begin;	begin;
	$K = K + 1$
$I = I + 1$	If COUNTER $(K) > 4$
	then
Input VALUE	begin;
TOTAL (I) = TOTAL (I) + VALUE	AVG (K) = TOTAL (K)/COUNTER (K)
COUNTER (I) = COUNTER (I) + 1	TOTAL (K) = 0
end;	COUNTER (K) = 0
end;	end;
	end;
	end;

Shared variables TOTAL and COUNTER are manipulated in both the tasks. If one task is modifying the variables and another task is invoked then there is probability of incorrect operations and result. This manipulation of shared variables must be enclosed in a block and a task while executing the block should not be preempted.

In the above algorithm, consider the following cases:

1. Task B2 is invoked immediately after "TOTAL (I) = TOTAL (I) + VALUE". The calculation of "AVG (K) = TOTAL ((K)/COUNTER (K) for $K = I$" will be wrong since COUNTER (I) has not been incremented.
2. Task B2 is preempted just after "AVG (K) = TOTAL (K)/COUNTER (K), and task B1 is invoked. The next VALUE for each parameter will be added to previous TOTAL. When task B2 is invoked again, the TOTAL and COUNTER are made zero.
3. Task B1 is invoked just after TOTAL (K) = 0 in task B2. VALUE is added to TOTAL (which is initialised to zero) and COUNTER is incremented (which is not initialised). The counter will become zero again when task B2 is invoked.

Thus in multi-tasking environment the sharing of data by different tasks may create number of problems due to manipulation of variables in number of tasks. Let us now discuss the various ways to solve this problem.

(i) *Mailbox.* Mailbox mechanism, as described earlier is designed for communication between two tasks. Although this can be used for task synchronization, for data sharing between two tasks, same structure cannot he used.

The task synchronization is achieved by passing data from one task to another. However, the protection of some part of the task from preemption may require creation of a block in the task, which is not possible using mailbox.

(ii) *Semaphore.* Using the operations *wait* and *signal,* critical section of a task can be protected by semaphore from preemption. The modified algorithm is

Task B1	**Task B2**
begin;	begin;
Initially SEMA = 1	
$I = 0$	$K = 0$
Repeat until $K > 5$	Repeat until $I > 5$
begin;	begin;
	If COUNTER (K) > 4 then
$I = I + 1$	
Input VALUE	begin;
Wait (SEMA)	Wait (SEMA)
TOTAL (I) = TOTAL (I) + VALUE	AVG (K) = TOTAL (K)/COUNTER (K)
COUNTER (I) = COUNTER (I) + 1	TOTAL (K) = 0
	COUNTER (K) = 0
Signal (SEMA)	Signal (SEMA)
end;	end;
end;	end;
	end;

In the beginning of task B1, when wait (SEMA) is executed then variable SEMA is checked. Since SEMA = I initially, it is made zero and task B1 proceeds. If task B2 wants to preempt task B1 now, its execution will stop at wait (SEMA). Since SEMA = 0, task B2 will be suspended and placed in semaphore queue. When the execution of task B1 reaches signal (SEMA) then SEMA is incremented by one, and task B2 in queue is made runnable. Now, task B2 may preempt task Bl. When wait (SEMA) is executed in task B2 (task B1 is suspended) then SEMA is made zero again and execution proceeds without any interruption. If task B1 tries to preempt, then at wait (SEMA) it will be placed in semaphore queue and only when SEMA is made one by signal (SEMA) of task B2, task B1 will be made runnable. Thus semaphore is effective in protection of shared data.

(iii) *Regions.* Mutual exclusion, i.e. access to shared data by only one task at particular instant can be very effectively achieved using regions. The two operations *Send control* and *Receive control* are used to form a block.

Task B1

```
begin;
I = 0
Repeat until K > 5
begin;
I = I + 1
Input VALUE
TOTAL (I) = TOTAL (I) + VALUE
COUNTER (I) = COUNTER (I) + 1
send control
end;
end;
```

Task B2

```
begin;
K = 0
Repeat until I > 5
begin;
If COUNTER (K) > 4 then
begin;
AVG (K) = TOTAL (K)/COUNTER (K)
TOTAL (K) = 0
COUNTER (K) = 0
send control
end;
end;
end;
```

8.4.5 Semaphore vs. Regions

Both Semaphore and Regions have the capability to solve the problem of task synchronization as well as mutual exclusion. Before deciding to opt for one of these, we must be aware of merits and limitations of two methods, over each other.

(i) *Priority bottlenecks.* The semaphore queue may be FIFO or priority queue. In case of priority queue, when a semaphore is available then highest priority task in the queue is made runnable. Suppose three tasks X, Y and Z have high, medium and low priority respectively. If these tasks have to solve the mutual exclusion problem for shared data using priority queued semaphore then following events may occur:

(a) Low priority task Z obtains access to shared data and is executing.

(b) High priority task X wishes to access the shared data but has to wait in semaphore queue (waiting for semaphore signal).

(c) Medium priority task Y, does not require to access the shared data. It is made runnable by an external signal. It is able to preempt low priority task Z and continues to run till completion.

The step (b) above is fair since task Z is using the data and till it surrenders control, task X, though with high priority should wait. However step (c) is unfair as a high priority task is made to wait for low priority task.

The *regions* solve this problem in a simple but effective way: In case the priority queue has opted for task, and if the task at the head of queue has higher priority than current task having access, then priority of the latter is raised temporarily to match the priority of the task at the head of queue. When the task having access surrendered access, its priority reverts back to original value. This priority adjustment prevents the priority bottleneck.

(ii) *Tying up shared data.* In multi-tasking environment, when a block of data is shared by a number of tasks using semaphore, and if, task currently having access is suspended then semaphore will prevent any other task from having access of shared data. Only after the suspended task is resumed and has surrendered the access, the other task can use the shared data.

If the task having access is deleted, then all other tasks are prevented from using the shared data and only external reset may retrieve the situation.

In case of regions which had access to shared data, task cannot be suspended or deleted by other task till access is surrendered. This prevents tasks from tying up shared data.

(iii) *Deadlock.* To avoid deadlock is a major concern of any multi-tasking system. Also known as 'deadly embrace', it occurs when two or more tasks permanently lock each other out of required resources. Basically faulty system design using regions or semaphore is the main cause of deadlock. Now, we present two examples of deadlock situation using semaphore and regions:

Example I: We are modifying task B1 and task B2 described earlier to include both task synchronization as well as mutual exclusion. Initially SEMA 1 = 1 and SEMA 2 = 0

Task B1	**Task B2**
begin;	begin;
$I = 0$	$K = 0$
Repeat until $K > 5$	Repeat until $I > 5$
begin;	begin;
$I = I + 1$	
Input VALUE	If COUNTER (K) > 4 then
Wait (SEMA 1)	begin;
TOTAL (I) = TOTAL (I) + VALUE	Wait (SEMA 1)
COUNTER (I) = COUNTER (I) + 1	AVG(K) = TOTAL (K)/COUNTER (K)
Wait (SEMA 2)	TOTAL (K) = 0
Signal (SEMA 1)	COUNTER (K) = 0
end;	Signal (SEMA 2)
	Signal (SEMA 1)
	end;
	end;

The idea is to ensure that task B2 gets a chance for execution after every sample in task B1. However following situation may occur.

1. Task B1 encounters wait (SEMA 1). SEMA 1 becomes zero and task proceeds, till it encounters wait (SEMA 2). Since signal (SEMA 2) is in task B2, task B1 is suspended and task B2 is executed.
2. Task B2 encounters wait (SEMA 1). Since SEMA 1 is zero, task B2 is suspended. In this peculiar situation, both the tasks are waiting for semaphore signal from each other and are stuck indefinitely. The same is explained in Fig. 8.17.

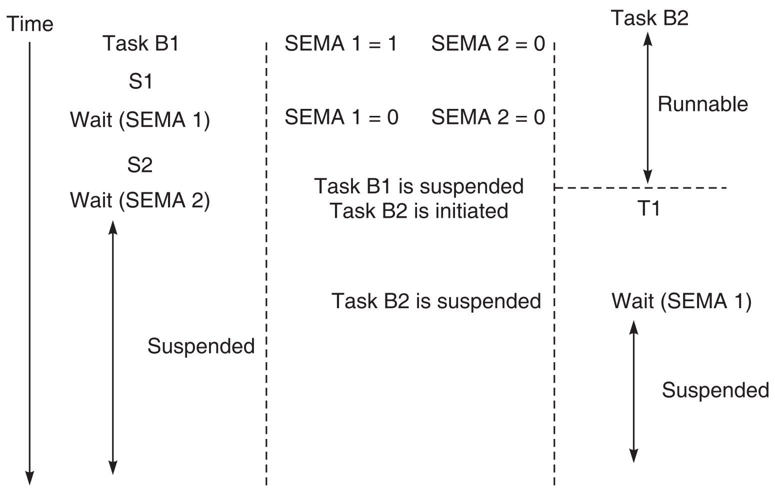


Figure 8.17 Example of deadlock using semaphore.

Example II: Let us consider two tasks—X (high priority) and Y (low priority) along with two blocks of shared data—data set 1 and data set 2. Both the tasks need to access both the data blocks. Access to data blocks is governed by region 1 (data set 1) and region 2 (data set 2). Following situations may develop if both the tasks are using nested regions (Fig. 8.18):

1. Task Y requests access to set 1 via region 1. Access is granted.
2. Task Y is interrupted by an external interrupt. Task Y completes the access to data set 1 but does not surrender the access. Task X is initiated.
3. Task X requests for access to data set 2 via region 2. Access is granted.
4. After completing the access to data set 2 and without surrendering, now task X requests for access to data set 1 via region 1. Access is not granted since Y is having access and task X is suspended.
5. Task Y is executed. It requests access to data set 2 via region 2. Access is not granted as X is having access. Y is suspended. Thus both the tasks remain in the state of suspension for all the time.

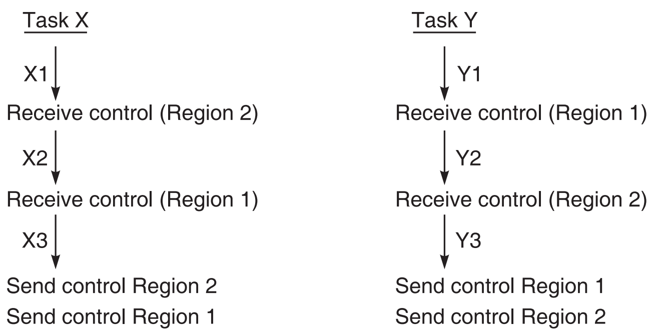


Figure 8.18 Example of deadlock using regions.

Since the deadlocks occur due to faulty system design, the only way to avoid these will be to exercise lot of caution before planning to use semaphores or regions. The nested regions must be used in one uniform sequence for all the tasks.

8.5 REAL-TIME OPERATING SYSTEMS VERSUS REAL-TIME PROGRAMMING LANGUAGES

What we have described in earlier sections are basically the ingredients of real-time programming systems. These basic features are incorporated using operating system and real-time programming languages. A real-time language must support facilities to protect shared data; task synchronization and real-time O/S must provide proper platform for programming language to operate effectively as well as support the remaining structure. Since program execution occurs through interaction between programming language and operating system, the question of facilities to be supported by operating system and real-time programming language is debatable.

In number of cases, real-time operating systems have been designed by augmenting conventional operating system with real-time executives. This approach provides the programmer a variety of tools to develop programs he is familiar with. In addition specific function for real-time applications are supported by real-time executive, which is basically an application software for operating system.

The second approach adopted is to develop a dedicated stand alone operating system like iRMX. To get UNIX advantage, real-time operating system with same system call as that of UNIX have been developed. This enables the extension of any software developed on UNIX.

In the similar way real-time programming languages have been designed by upgrading existing general purpose languages having some desired characteristics. Real-time PASCAL was developed by Texas Instruments by providing a new structure concept called ‘system’. This augmentation also supports some minimal real-time functions like procedure communication, synchronization among process, power up and power failure processing, memory management etc. This replaces the conventional operating system by providing run-time support for programs.

The concept of run-time support instead of operating system is useful specially in cases where the system hardware is limited. Consider the case where the hardware does not provide disk, console or any other peripheral. Such applications are numerous in offshore oil drilling, defence, mines, robots etc. In addition, many real-time process control programs cannot meet the memory limitation in microprocessor systems (with no back-up memory) if fitted with general purpose real-time operating system due to later’s overheads. With run-time support system, a language with “real-time” facilities may accomplish the goals in many cases. However, run-time support systems are not for general purpose. They are tailored to a machine architecture for which they are designed. This however gives them speed advantage over operating systems since they do not incur overhead of services like I/O, libraries which the application does not require.

8.6 REAL-TIME PROGRAMMING LANGUAGES—A SURVEY

The desirable traits of a real-time programming language are,

— multi-tasking capability
— constructs to directly implement real-time functions
— modern programming features to ensure programme correctness.

We have already discussed, the concept of multi-tasking. Thus any discussion on that will be mere duplication. Multi-tasking is normally provided in real-time operating systems but many real-time applications do not use operating system. They are programmed using language which provides sufficient run-time support. This saves memory and speeds up the program execution. In such cases, multi-tasking is supported by programming language and run-time support executive.

The constructs that are used to implement real-time functions in programming language are like schedule, wait, signal etc.

We have already illustrated the use of wait and signal for implementation of semaphore mechanism, process synchronization as well as protection of critical section of program. Following is the format of schedule in PEARL:

$$\text{Schedule} \begin{bmatrix} \text{Activate} \\ \text{Suspend} \\ \text{Continue} \\ \text{Delay} \\ \text{Terminate} \\ \text{Prevent} \end{bmatrix} \text{Task name}$$

A task may be scheduled on the basis of time or on the basis of event. The task is accordingly placed in time queue or event queue, by the compiler.

Thus, the implementation of these constructs helps in program development. Lastly, the facilities for reliable programming are needed more than anything else. Since real-time programs are used for controlling the process, any error may create a havoc. Thus the features like modular programming, strongly enforced data typing and other rules are essential.

Process control languages may be grouped in two major categories: Problem-Oriented Languages and General Purpose Languages.

8.6.1 Problem-Oriented Languages

Problem-Oriented Languages are specially developed for various types of applications. A number of languages in this category have been developed for applications like control of batch reactors, sequential control of processes, etc. The objectives of problem oriented programming are—to fill up the data files and to inform software on the actual numerical parameters, to specify the execution of package and to describe non-standard operations. The problem-oriented languages may be either directly formatted "fill in the blanks systems" or high level English-like languages.

Fill in the blank systems

The fill in the blanks technique has been basically useful for unskilled programmers. Although this is a strictly formatted system but the programmer need not bother about the formatting since the entire software is menu-driven. The programmer has to enter different numerical parameters for data acquisition and conditioning, filter and conversion coefficient, dead band or time delay, type of filter conversion equation etc. by means of a series of questions or menu displayed on the computer CRT screen. The programmer is able to make the proper choice of algorithms for each of the control loops from the suitable options. The fill in the blanks system is basically a pre-programmed interpreter type program having a large number of selection choices with a well developed skeleton database. The programmer thus is able to develop an operating control program for his process sitting in front of the CRT screen. The two fill in the blanks systems which have been popular are *Prospro* and *Biceps*. These are basically supervisory control packages.

In Prospro, non-standard arithmetic operations are supported through a general equation and an adjustment equation. The user may opt for his particular equation by specifying his own coefficient which may also he zero. This is performed by special menu. An assembly like procedural language is provided for programming non-standard operations. The statements for arithmetic operations, comparison, conditional and unconditional branch, time operation, adjustment and program control operations have been provided. Prospro has been used for control on catalytic reactor.

The fill in the blanks programming is suitable for wide range of applications which require fast response. Since the user is able to program it on the CRT screen, the program is bulky and inefficient.

High level problem-oriented languages

It is not uncommon to develop problem-oriented languages by using popular computer languages such as FORTRAN and BASIC as the base. A number of such programs have been developed for a wide range of different processes and manufacturing operations. An example of the use of high level languages in programming process control is AUTRAN (Automatic Utility Translator) developed for data acquisition and supervisory control systems.

The AUTRAN language has following types of statements.

1. Group specifications
2. Input and output specifications
3. Control operation specifications
4. Control processing
5. Input and output processing
6. Alarm response specifications
7. Timing specifications.

The language has a version of FORTRAN which has a subset with additional statements for I/O variable list, tasking statements and logging statements. The purpose of AUTRAN is to eliminate steps that were repetitive and limit programming to process description, control action, alarming, displaying and logging.

Typical actions that may be controlled using AUTRAN include:

1. Opening and closing of valves
2. Starting and stopping of motors
3. Closing and opening of switches
4. Activating and deactivating DDC loops
5. Tuning DDC loops
6. Setting or adjusting alarm loops
7. Initiating other AUTRAN programs and logging requests.

This language is suitable for control systems engineer. The system engineer plans the process control by specifying process variables with alphanumeric equation, computer channel address and stores these definitions in computer memory. The AUTRAN compiler checks for syntactic and semantic violation. The compiler also checks the various commands for any semantic errors. AUTRAN was designed for CDC 1700 computer system with a real-time operating system, a data acquisition system, a direct digital control system and a process executive software which are needed to complete the process control software operating with AUTRAN.

8.6.2 High Level General Purpose Process Control Languages

General purpose languages are basically procedural languages. Thus most of their statements can be executed. For process control applications, the language must have the characteristic of multi-tasking and higher run-time efficiency. The high level languages are beneficial for process control applications considering their short program development cycle, transportability etc. However, high level languages are a degree less efficient than problem oriented languages. This efficiency factor may play a crucial role in many applications. It is therefore advisable that the critical sections of the program used very frequently and also the functions which have to he performed within a fixed time duration must be programmed in the lower level languages like the assembly and machine. The other disadvantage of general purpose higher level language for process control are ‘On-Line Program Testing’ and ‘Program Modifications’. The base language selected for development of general purpose programming languages are algorithmic languages like FORTRAN, PL/l etc. Since FORTRAN has been constantly popular since its inception, a number of general purpose process control languages have been developed as real-time extensions of FORTRAN. Purdue FORTRAN was developed by the FORTRAN Committee of the workshop on Standardisation of industry programming languages held in Purdue. This workshop was held to standardise different process control systems. The language extensions were released for the different actions which are required in real-time systems. These include FORTRAN for tasking, process I/O and bit string manipulation.

The other languages popular for real-time application include CORAL RTL/2, PEARL, PROCOL and ADA. While designing languages like CORAL and RTL, run-time efficiency has been the prime objective. These languages exhibit a straight-forward structure with no explicit real-time features. Machine dependent procedures and micros are used to implement time operations like tasking and input/output operations. It is possible to insert assembly or machine code sequence in the high level program. The micro-instructions can he defined and used throughout the program. *Pearl* and *Procol* have special language level facilities for real-time process control operations. These facilities include system description, tasking, synchronization and process I/O.

System description

The system description comprises specifications of physical points on which the interrupts may occur; the types and number of peripheral equipments and communication points and the complete data path including the computer type, features of CPU, memory size, communication channels etc.

Tasking

The tasks can be generated statically or dynamically. In static task generation, a given task code is 'fixed' with a task name whereas in dynamic generation the name of the task is associated with the task on activation. The format of the task instruction under Pearl is given here.

AT AFTER ALL	TIME	ACTIVATE CONTINUE SUSPEND RESUME PREVENT TERMINATE	TASK (NAME)

The task schedule is given in the same way as discussed earlier. The task operation can be attached to an event, time or both. The languages Pearl and Procol use semaphore for task synchronization. The concept of semaphore has already been explained. The semaphore operations *wait* and signal are implemented by using instructions *request* and *release.* The request operation decreases the value of semaphore by one. If the value is negative then task containing the request operation is suspended. The release operation increases the value of the semaphore by one. This enables the highest priority task waiting in the semaphore queue to be executed.

Input/Output

Pearl and Procol handle input/output operations in different ways. The Pearl has following statement for process I/O.

"MOVE SOURCE TO SINK".

The source may be defined as a symbolic name of communication register of a device and sink may be defined as the name of the memory location for process input operations. For process output operations, source may he memory location and the sink may he communication register of device which is connected to the bus. The Procol implements process I/O through separate input/output statements. These statements in Procol require identification of data source and sink and also formatting scheme. The formatting scheme includes feasibility checking, fulfilling format conversion and logical checking. This facility is not provided in the

Move statement of Pearl. If formatting is however needed then a gauge is attached to the Move statement which attaches a procedure required for transformation.

RTL/2

While designing RTL, the basic structure of ALGOL is extended in order to provide parallel programming and other on-line facilities which are necessary in real-time applications. Certain restrictions on ALGOL structures have also been incorporated for consistency. The conceptual difference between quoted statements and assignment statements is generalised. Those statements which donot define their successor explicitly are permitted to have parallel lines, for instance assignments statements. This classification has evolved the notions of program statements and program procedures. A *program procedure* is defined as the procedure which does not contain *go to* statements leading to outside procedure either directly or through a parameter. This facility has been helpful in ensuring the program security.

A number of multi-programming statements for real-time applications have been included. The main statements are used to specify response of the machine to the interrupt. The activation statements can be used in procedure based parallel programming systems. The execution of an activate statement leads to a new activation of the procedure called by procedure statement. A procedure activation may he in one of the following four states:

— Running—being executed by processor;
— Ready to run;
— Held up—partially executed but now temporarily suspended;
— Dead—procedure activation has been completed.

Protection statement basically comprises two statements, namely, *secure* and *release*. The function of *secure* is to decrease the value of its apparent facility by one, whereas the function of release is to increase its apparent facility by one. These statements can be used for process synchronization.

In order to implement RTL in a variety of systems, considerable flexibility in compiler design is absolutely necessary. The techniques used in the language may not be appropriate for compilers in scientific steps. The object program should include test and measurement simulation facilities.

ADA

For different real-time designers, the experience of designing and programming using ADA have been varied. Some have declared it as ultimate language for embedded system whereas others have rejected it outrightly as a language which produces unreliable systems. The ADA does not provide real-time constructs directly but facilitates software designer to create his own real-time programming features. Thus, the user has freedom to design suitable applications and specific systems. This facility has, however, an apparent risk that the designer can create programs which cannot he maintained. Under ADA, the conventional notations can also be redefined under certain conditions. Thus its constructs are subject to modification depending on the personal preference of designer. ADA has a capability to overload operators as well as functions. This means that in addition to facilitating the new definition of data type (as in PASCAL) ADA allows the designer to define the new operators as well. Thus in ADA, the programmer may overload the operator "plus(+)" to mean something else than addition, for example, merger of two files etc. The overloading of functions allows the programmer to define routines like sine, cosine etc. with different number of parameters than the standard routines.

Though these capabilities provide the programmer tremendous power and scope for system design but may cause nervous breakdown while debugging a system. The multi-tasking feature in ADA permits the programmer to handle interrupts without going to lower level mechanism such as machine architecture. The interrupt handlers are normal ADA tasks which are attached to interrupt by compiler.

C *and* C++

The C language is widely used high level programming language for system programming as well as application development for embedded system. The main features that distinguish C from other languages are—efficiency of generated code, the simplicity and wide availability of compilers and development tools.

The C++ language as a superset of C with object oriented facilities is a natural candidate for these developers moving from applications developed in C to a world of more complex software in which object oriented abstractions help in handling the complexity and increasing reuse.

Many programming languages are unsuitable for real time programming due to their non-deterministic nature. As an example, languages that have built in garbage collector (garbage collector is a system program which automatically untags unused memory space and allocates it to free memory space pool) are unsuitable because garbage collector might wake up at the wrong moment and halt all other operations till it finishes. A real time software application program that controls fuel supply to aircraft engine cannot be interrupted even for a moment.

The designers of C++ deliberately refrained from adding features that would disqualify it from being used in real time applications. It is fact that every feature of C++ is not deterministic. For example dynamic memory allocation and operator dynamic-cast are not deterministic because of extraneous factors like heap fragmentation. However, unlike other languages C++ doesn't force these features on users. Thus user can avoid these features ensuring a deterministic operation.

Java

The size and complexity of embedded application have grown over the time. This resulted in C and C++ application design becoming fragile and cost of maintaining them have grown very high. The evolving complexity of embedded systems has led to the need for a new language that helps developers to manage complexity, increase productivity and reduce time to market.

Java has proven to be language and plateform of choice resulting in substantial increase in quality and productivity. In 1997, an expert group from 50 major technological companies including Sun, Microsoft, IBM, Motorola considered Java for real time applications and held it superior to other languages due to following main reasons:

— Higher level of abstraction.
— Supports dynamic loading of new classes.
— Highly dynamic, supporting object and thread creation at run time.
— Supports components creation and reuse.

— Supports application portability.

— Supports distributed applications.

Under Java community process (JCP) generalized specifications for real time extensions to Java called RTSJ have been developed to make Java technology more deterministic and enable it to meet rigorous timing requirement for mission critical real time applications. The Sun Java Real Time System (Java RTS) is Sun's commercial implementation of real time specification of Java (JSR-001).

Java RTS enables developers to take full advantage of Java language ecosystem while maintaining the predictability of current real time development plateforms. This means that real time components and non-real time components can coexist and share data on a single system. The main features of Java RTS are;

— New real time threads, scheduling and synchronization.

— New memory management schemes.

— Asynchronous event handling and asynchronous transfer of control

— Time and timers.

— Direct access to physical memory.

Java RTS brings the world of real time programming to create applications that involve physical world (Sensors, Process, Control Valves etc.). It offers a very predictable, low latency software architecture for embedded system applications.

8.7 iRMX REAL-TIME OPERATING SYSTEM

This operating system has gained considerable popularity and has a fairly representative structure. It has the following components:

— Nucleus

— Terminal handler

— Debubgger

— Basic I/O system

— Extended I/O system

— Application loader

— Human interface

We shall confine our attention to the nucleus, using which all the other components are built.

The nucleus of iRMX operating system is essentially a resource manager. The resources which the operating system iRMX can handle are divided into the following areas:

— Processor time

— Objects

— Memory

Processor Time is the key resource of the system which is managed by iRMX. Many tasks require the processor which is placed in the ready state. The processor always executes the highest priority tasks. If more than one task of that priority is ready then the processor is allocated the task that has been ready for the longest period. Once a task gets the control of processor, it is executed till preempted by a higher priority task or an external event.

The iRMX nucleus has the following functions:

— create, schedule and delete tasks;

— control access to system resources;

— provide inter-process communication;

— enable the system to respond to external events

Tasks are processes or sequential programs whose executions in the CPU are interleaved in time by the nucleus which allocates 'time slices' to tasks in the 'ready' state. When it gets a time slice the task shifts from 'ready' to 'running' state. After its time slice is over it goes back to 'ready' state.

A task has three other states:

— Asleep

— Suspended

— Asleep-suspended

It can enter the *asleep* state voluntarily for a finite time or be forced into it by the non-availability of some wanted resource. In the latter case, it may go back to ready state when the resource becomes available. The *suspended* state is similar to the asleep state but several levels of suspension are possible. A task can be suspended by itself or by another task. A suspended task can get back to ready state only when another task 'resumes' it. The level of suspension of a task is decided by the number of 'suspend's issued to it. The same number of 'resume's are required to take it back to *ready state*. If a task is in the asleep state when it is suspended by another, it enters the 'asleep-suspended' state. From here it can move either to the asleep state or to the suspended state depending on whether a resume is issued first or the asleep state terminates first.

Tasks are one type of 'object' supported by iRMX nucleus. The other types of objects are:

Jobs	: environments in which tasks execute
Segments	: pieces of memory
Mailboxes	: objects which enable task to send or receive other objects
Semaphores	: objects that enable tasks to send signals to other tasks
Regions	: objects that ensure mutually exclusive access to shared data.

In addition to these, the nucleus also allows the user to define extension objects which are new object types. Instances of extension objects are called composite objects.

An iRMX based application will be a set of jobs organised in tree structure. The root job is supplied by the nucleus. Each job is created with a memory pool and an initial task. This task can do the necessary initializations and create the other tasks of the job. Tasks can create jobs also. They will be considered *offspring* jobs of the job containing the task(s) which created them and their memory pools will he borrowed from the pool of the parent job. The nucleus supports a set of commands or primitives which can be used to invoke its facilities. Its internal data structures are invisible to the user and it ensures the integrity of the objects created by means of its primitives.

The nucleus supports the definition of handlers for interrupts and for exceptions. Interrupt handlers enable interrupts to be handled without invoking a task. Exceptions, i.e. error conditions can be handled by defining handlers that report the error and take remedial action or abort the task. Some default handlers are included in the nucleus. The user can define his own set of exceptions and handlers if he wants to.

iRMX provides an elegant way to design software with well-defined and highly modular constructs. Such software will be easy to understand, test and maintain.

The features and facilities described so far belong to iRMX I operating system, which was originally named as iRMX 86. It is most successful of all iRMX versions so far. iRMX II was developed to support real-time applications around 80286, 80386 and 80486 family of microprocessors. Continuing along the up grade path is iRMX III operating system. It supports 80386, 80486 and Pentium family of microprocessors. Later iRMK was developed and released as part of effort to produce an operating system that offered full distribution of real-time objects throughout all CPUs in a Multibus II system. iRMK was first real-time kernel for the Intel 386 family of microprocessors and was introduced on Multibus I. Multibus II and embedded designs. In 1994 iRMK was integrated into the iRMX III operating system increasing its speed and flexibility.

The latest versions of iRMX are iRMX for Windows operating system and iRMX for PCs operating system.

The layered architecture generally followed by all iRMX operating systems is shown in Figure 8.19. Each layer represents software interface to the next layer and offers different set of capabilities to user and applications. Each version of iRMX operating system described above implements these layers in different ways. The comparison between different versions of iRMX operating system is shown in Table 8.1. We, now present salient features of these iRMX operating systems versions.

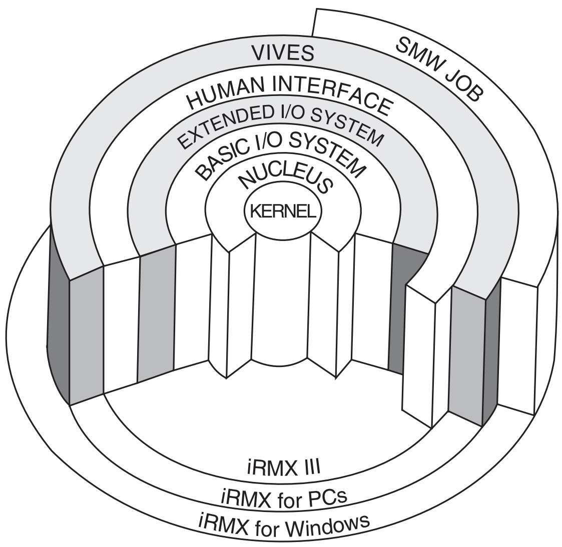


Figure 8.19 Layered architecture of IRMX operating systems.

iRMX I operating system

It offers real-time processor management for the 8086, 80186, 80188, 80286, 80386 and 80486 family of microprocessors. The features of iRMX I have already been described. Only 1 megabyte of memory can be addressed irrespective of processor type. Data accesses for read or write operations are also limited to 64 kilobyte memory at a time due to hardware limitation on segment length. iRMX I supports Multibus system boards or is stored in PROM in to a Single Board Computer or a 'boot and forget' applications.

Table 8.1 iRMX Versions*

	iRMX I	*iRMX II*	*iRMX III*	*iRMX for Windows*
Current release	I.8.1	II.4.1	III.2	2.0B
First released	1980	1985	1989	July 1992
Nucleus size	14–26**	39	56	N/A
Full OS size (KB) w/o Network†	308	378	550	4MB w/Windows & Dev. User
Network size (KB)	158	220	230	230
Remote boot	Multibus I	Multibus I	Multibus I	PC/AT, EISA, MCA
Prommable	Yes	Yes	Yes	No
Max. segment size	64KB	64KB	4GB	4GB
Max. number of objects	64K	8192	8192	8192
SBM/SDB support	PROM	PROM	Softloaded	Softloaded
Source-level debugger	Soft Scope 86	Soft Scope I	Soft Scope III	Soft Scope III
Multibus II support	Class B ZAP	Built in	Built in	Loadable job
CPU mode	REAL	Protected	Protected	Protected
CPU type	8086, 80 × 86	80286, 386, 486	386, 486, Pentium	386, 486, Pentium
Max. application size	1 MB	16MB	4GB	4GB
Number of priority levels	255	255	255	255
DOS as VM86 task	No	No	No	Yes
Windows 3.1	No	No	No	Yes

*iRMX product features

**Size depends on number of system calls configured

†Approximate value only—actual size depends on device-driven configuration.

iRMX II operating system

It is offered on 80286, 80386 and 80486 family of microprocessors. It executes in a special mode of the microprocessor called Protected Virtual Address Mode or PVAM. Unlike iRMX I operating system, the tokens for iRMX II objects are selectors that refers to an entry in descriptor table. iRMX II supports Multibus I, Multibus II and can also be stored in PROM into embedded designs. It was also offered for an IBM PC–AT platform called the Intel system 120 as a special way to have either iRMX operating system or DOS hooted from the same media.

iRMX III operating system

It is offered on the Intel 386, Intel 486, Pentium and Pentium Pro families or microprocessors as the first 32-bit entry into real-time operating system family. It also executes Protected Virtual Address Mode accessing 4 gigabytes of memory. It supports Multibus I, Multibus II and some embedded designs. iRMX III operating system uses a different debugging scheme than used by iRMX I and iRMX II, where System Debug Monitor (SDM) reside in PROM on the SBC hosting operating system. In iRMX III, SDM is loaded into RAM and user Intel 32-bit monitor as protected mode interface to the hardware.

iRMX for windows operating system

iRMX for windows is designed with the Intel 386, Intel 486, and Pentium based PC in mind. It has ability to load or unload jobs, operating system extensions and device drivers dynamically for human interface. It also supports PC compatibles in the Multibus I and Multihus II environments as well as IBM or compaq compatible PCs.

Communication between two operating systems is very important for acquisition and control applications. iRMX for windows operating system offers four main methods for operating system interaction on either a file or single byte level.

- Real-time Extension Library—allows popular DOS tools to make calls to iRMX nucleus, create objects, pass or receive data or access protected mode memory.
- File to file access between DOS and iRMX: Windows operating system is made possible since both operating systems reside on the same media and share file structures.
- OSI Network support is embedded within iRMX for windows operating system and allows local and remote access via open NET, Novel1 Netware or TCP/IP.
- Dynamic Data Exchange (DDE) interface built into iRMX for window operating system can be used by applications in Microsoft Windows 3.1 to communicate to real-time applications. DDE router for message communications is also provided for efficient message communication.

The iRMX for Window operating system loads and runs on a standard Window system. Upon initialization, it creates a separate execution environment, takes over the CPU and encapsulates Windows as lowest priority iRMX task. The iRMX operating system scheduler determines which task will run, insuring that “ready to run” highest priority task is always made running task.

Whenever a real time task is ready to run, it preempts Windows and its execution environment. The scheduler handles all the associated real time activities and resumes Window task after all the real time activities have been completed.

Real-time applications based on the Intel X86 architectures are predominantly developed using one of the real-time operating systems: iRMX I, iRMX II, iRMX III or iRMX for windows operating system. Each of these operating systems behaves differently, satisfying unique needs of each design and possessing characteristics distinct from others.

8.8 REAL TIME LINUX

A number of architectures of Linux support real time applications. These architectures differentiate between hard real time and soft real time applications. In an application the operating system is required to support the deadlines of real time tasks (even under worst case processing loads) then application is called hard real time application. On the other hand if operating system can support application dead lines on average and it is acceptable to user then such applications are called soft real time applications.

Hard real time applications are those where missing a deadline may cause catastrophe. Consider a vehicle Antilock Braking System (ABS). This control system samples the speed of each wheel on a vehicle and controls each brakes pressure to stop it from locking up. For the control system . to work, sensor sampling and control must be performed at high speed at periodic intervals. This means that other processing must be preempted to allow ABS task to execute at desired period. This is example of hard real time system and application.

In many applications hard real time support may not required. In soft real time applications, system can miss the dead line without causing the failure of overall system. As an example short disruption in voice communication or losing a video frame will not fail the respective systems.

Following are some of the architectures of Linux developed to support real time applications.

8.8.1 Thin Kernel Architecture

The thin kernel (or micro kernel) architecture uses a second kernel as an abstraction interface between the hardware and the Linux kernel (Fig. 8.20). The non-real time Linux kernel runs in the background as a lower priority tasks of the thin kernel and hosts all non-real time tasks.

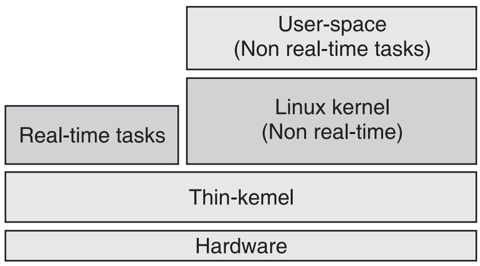


Figure 8.20 Thin kernel architecture.

Real time tasks run directly on the thin kernel. The thin kernel intercepts interrupts to ensure that non-real time kernel do not preempt the operation of thin kernel. This enables the hard real time support to be provided. However, the disadvantage of this architecture is that real time and non-real time tasks are independent which makes debugging very difficult.

8.8.2 Nano Kernel Architecture

The thin kernel architecture described above relies on a minimized kernel that includes task management. The nano kernel architecture goes a step further by minimizing the kernel even more. It is thus more of a Hardware Abstraction Layer (HAL) than a kernel in this architecture. The nano kernel provides for hardware resource sharing for multiple operating systems operating at higher layer (Fig. 8.21). It also supports prioritization of higher layer operating system and thus supports hard real time applications.

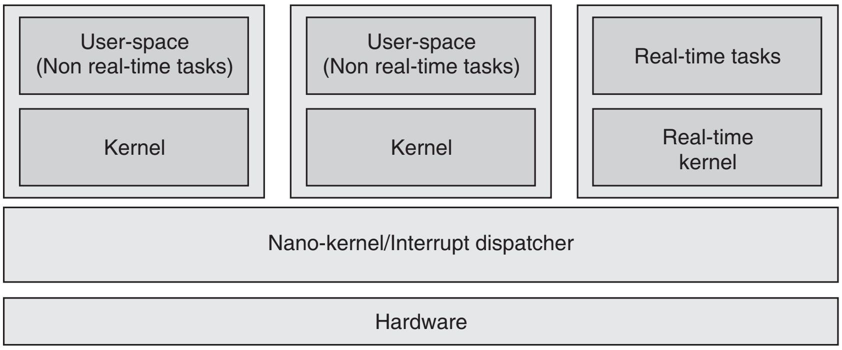


Figure 8.21 Nano kernel architecture.

8.8.3 Resource Kernel Architecture

This architecture adds a module to kernel to provide reservations for various types of resources. The reservations guarantee access to time multiplexed system resources like CPU, network, bandwidth etc. These resources have several reserve parameters such as period of recurrence, required processing time and deadline. The resource kernel architecture provides a set of Application Programs Interfaces (APIs) to allow tasks to requests these reservations (Fig. 8.22).

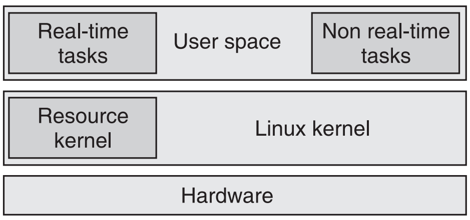


Figure 8.22 Resource kernel architecture.

The resource kernel merges the requests to define a schedule to provide guaranteed access. Dynamic scheduling of work load is performed by the kernel by using Earliest Deadline First (EDF) scheduling algorithm. In this way support for hard real time applications are provided.

8.8.4 Real Time Application Support Using Standard 2.6 Kernel

In the present Linux architecture having 2.6 kernel, it is possible to provide support to soft real time applications by making kernel fully preemptable (Fig. 8.23). When a user space process makes a call in to the kernel through a system call then it cannot be preempted. Thus if a low priority process makes a system call then high priority process will be made to wait until the system call is complete before it can gain access to the CPU.

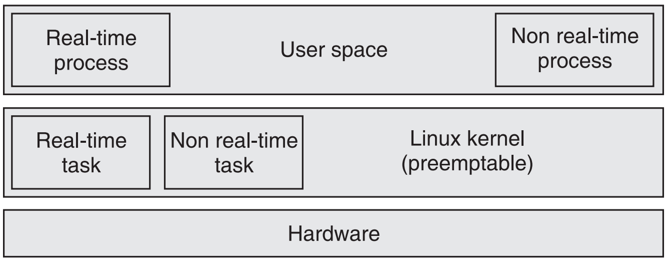


Figure 8.23 Standard 2.6 kernel architecture.

The above anomaly has been corrected through a configuration option—CONFIG_PREEMPT, which changes the behaviour of kernel by allowing processes to be preempted if higher priority task is available. This enables support for soft real time applications but with a overall lower throughput.

8.9 CONCLUSIONS

We have introduced the concepts of real-time programming in this chapter. The control system programmers have to exert extra caution while programming to avoid deadlocks, and to have smooth task synchronization. A number of advanced operating systems, programming languages and software tools have been designed to aid the control system programmer. However the experience of control system programmer in foreseeing such problems and avoiding them would be an essential requirement, for successful implementation of any such project.

SUGGESTED READING

Bromine, L., Choosing a language for control, *Instruments and Control Systems*, **57,** No. 4, pp. 57–59, 1984.

Copeland, J.R. and Roland, E.T., Programming languages provide real-time control, *Instrument and Control Systems*, **57,** No. 9, pp. 65–76, 1984.

Davis, W.S., *Operation Systems: A systematic view*, Addison-Wesley, Reading, Massachussetts, 1987.

EWICS TC2 8118, IRTB Industrial Real-Time BASIC, Draft Standard European Workshop on Industrial Computer Systems, 1981.

Gait, J., A Class of High Level Languages for Process Control, *Computers in Industry*, **4,** No. 1, pp. 67-70, 1983.

Gertler, J. and Sedlok, J., Software for process control-A survey, 4th IFAC (IFIP Conf: on *Dig. Comp. Appl. in Proc. Control*, Zurich), 1974.

Glass, R.L., *Real-Time Software,* Prentice-Hall International, London, 1983.

Gordon, M.E. and Rodinson, W.B., Using preliminary Ada in process control application, *AFIPS Conf. Proceedings*, **49,** AFIPS Press, Arlington, Va., pp. 597–606, 1980.

IEEE, *Workshop on Languages for Automation*, Computer Society Press, New York, 1988.

Kompass, E.J., The importance of control operating systems, *Contr. Engg.,* **35,** No. 3, pp. 39–41, 1988.

Martin, T., Industrial experience with process control programming language PEARL, *IFAC World Congress*, Munich, Germany, 1987.

Menn, A.A., Distributed operating systems for computer control systems, *Autom. Remote Control* (USA), **49,** No. 1, pp. 1–27, 1988.

Packer, J., Simpler real-time programming with the transputer, *Technical Note*, **51,** INMOS, Bristol, UK, 1988.

Schmidt, G. and Swik, R., Microcomputers in automatic control applications—A software point of view, *IFAC World Congress*, Munich, 1987.

Stankovic, J.A., Misconceptions about real-time computing: A serious problem for next-generation systems, *IEEE Computer*, **21,** No. 10, pp. 10–19, 1988.

Tucker, B.A., *Programming Languages*, McGraw-Hill, New York, 1986.

Young, S.J., *Real-Time Languages: Design and development*, Ellis Horwood, Chichester, UK, 1982.

——, *Introduction to Ada*, Ellis Horwood, Chichester, UK, 1983.

CHAPTER

9

Personal Computer in Real-time Environment

9.1 INTRODUCTION

The personal computer has brought major revolution in all application fields, and real-time applications are not exceptions. Having discussed different modules of real-time systems, we shall now deal with general aspects of personal computers and how they can be used in real-time environment. The interfacing of PC to different I/O modules will also be described.

However we shall not discuss here the details of 8088 microprocessor, keyboard, printer, VDU etc., as the descriptions of these standard items are available in many books. The description of some pins and signals of 8088 will be dealt with, as and when required.

9.2 PERSONAL COMPUTER: SYSTEM AND FACILITIES

The basic components of personal computer are System Unit, Keyboard, VDU, and Printer. The system unit is the heart of personal computer and consists of processor board, power supply, floppy disc drivers, and speaker. In the initial design i.e. PC–XT, the processor board contained 8088 microprocessor, up to 640KB RAM, 40KB ROM, I/O adapters for the keyboard, tape cassette and audio speaker.

The processor board also contains five system bus slots which facilitates the usage of additional devices to be interfaced to the personal computer. This feature is widely used in real-time environment as all real-time modules are interfaced through these slots. Following are features of some of the add-on cards available in the market:

1. Diskette drive controller card allows attachment of two 5¼, inch floppy drives internal to the system unit and two external drives.
2. Parallel printer port card, offers cemtrpmocs printer interface.
3. Serial port card has a single asynchronous serial port with a RS232-C electrical interface that allows attachment to a modem for data communication over telephone lines.

4. Game control adapter card allows the attachment of joy-sticks to the IBM PC to support game programs.
5. Colour graphic adapter card provides the ability to connect a variety of monitors and displays. It supports both monochrome and colour in both text and graphics modes.
6. SDLC adapter card is used to attach terminals and computers together over telephone lines by using Synchronous Data Line Control (SDLC) Protocol for data communication. This card requires a special synchronous modem in order to be used over telephone lines.
7. Memory Expansion Cards allow expansion of RAM available on the processor card. Expanded memory can have capacity from 1 to 16 megabytes memory space.

Obviously, none of these cards fit in to real-time systems described previously. The designs of cards offering real-time facilities will be discussed later. Figure 9.1 shows the block diagram of PC along with different cards described above.

Many advancements in the microprocessor technology have resulted in various advanced configurations of PC's. They are:

PC-XT — 8088 based PC with Hard disk drive storage—10 MB to 80 MB, 640 KB of RAM, Monochrome or Colour graphics adapter

PC-AT — 80286 based PC with Hard disk drive storage—40 MB to 120 MB, 16-bit ISA bus. Monochrome/Colour EGA/VGA adapter

PC-SuperAT — 80386/80486/Pentium based PC with 16-bit ISA bus or 16/32-bit EISA (Extended Industry Standard Architecture) bus, PCI bus, 4MB to 64MB RAM, Monochrome/Colour EGA/VGA adapter

9.3 PC BUS AND SIGNALS

The PC Bus signals are available on five 62 pin cards slots on the processor board. All the signals are at TTL logic levels except power and ground. The 8088 microprocessor bus was demultiplexed and augmented by providing signals for DMA, Interrupt, Memory and I/O Read/Write Control, Wait State generation, Memory Refresh, and Error Detections, to constitute PC Bus. We shall describe these signals functionally in the following paragraph.

9.3.1 PC and XT Bus Signal Lines

Signals Line	**Description**
A0–A19	Twenty System address lines. During I/O cycles, only the lowest 10 lines are actually used.
D0–D7	Eight bi-directional data lines.
ALE	Address Latch Enable. This signal goes high to indicate that a valid address is present on A0–A19 during a memory access.
IRQ2–IRQ7	Six maskable interrupt request lines.
DRQ1– DRQ3 and $\overline{\text{DACK1}}$ – $\overline{\text{DACK3}}$	DMA request and acknowledge lines. There's no DRQ0 on the bus; DMA channel 0 is used for DRAM refresh on the PC and the XT.

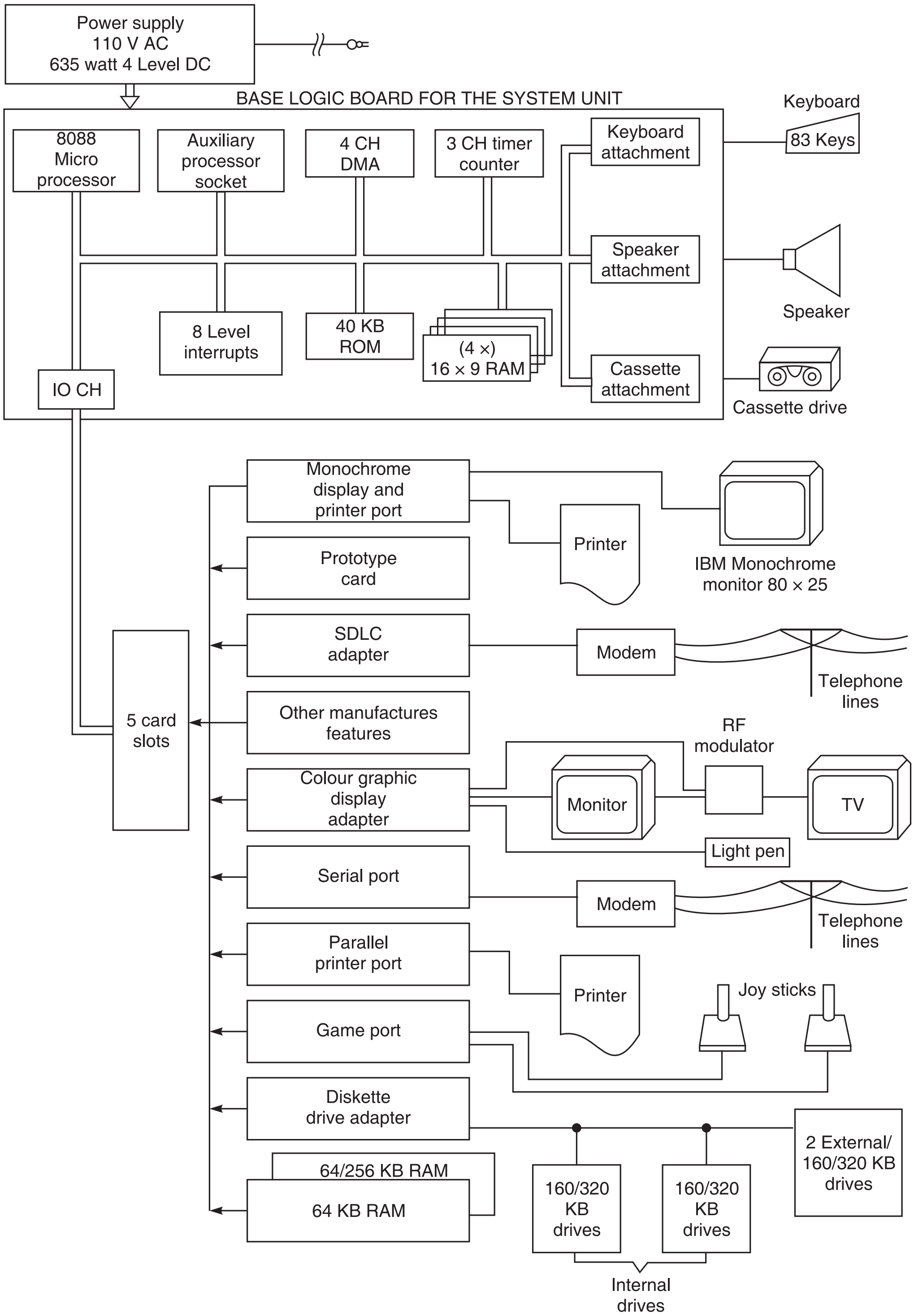


Figure 9.1 Block diagram of PC and its features.

IO CH RDY	A signal used by a memory or peripheral board to generate wait states.
$\overline{\text{IOR}}$, $\overline{\text{IOW}}$ and $\overline{\text{SMEMR}}$, $\overline{\text{SMEMW}}$	I/O and memory read and write strobes.
OSC	A 14. 31818-MHz clock used by some video board. It's not synchronized with respect to rest of the bus.
CLK	The bus clock signal (4.77 MHz in the original PC and proportionately faster on later machines). This clock is synchronized with respect to the read and write strobes.
AEN, TC	Address enable and terminal count. These control signals are used during DMA cycles.
$\overline{\text{IO CH CHK}}$	Alerts the processor to parity and other errors via a non-maskable interrupt.
RESET DRV	Indicates that the system is being reset.
+5VDC, –5VDC +12VDC, –12VDC, GND	Power-supply and Ground lines

9.3.2 PC-AT ISA Bus Signals

Following are additional signal lines for the 16-bit ISA bus

Signal Lines	**Description**
D8-Dl5	The eight new data lines
$\overline{\text{SBHE}}$	System Bus High Enable, which indicates when these data lines are being used.
IRQ10–12. IRQ14–15	More interrupt lines. IRQ13 is absent because that interrupt is reserved for the math coprocessor.
DRQ0–$\overline{\text{DACK0}}$ DRQ5–DRQ7 $\overline{\text{DACK5}}$–$\overline{\text{DACK7}}$	More DMA control lines for new DMA channels. On the AT, DMA channel 0 is no longer used for refresh and is therefore available for other purposes.
$\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$	Memory read and write strobes. These signals are active on all memory cycles, while SMEMR and SMEMW are active only on cycles that fall within the address space of the PC for compatibility reasons.
$\overline{\text{MASTER}}$	A new signal that lets a board become a bus master on the AT bus. A bus handover using this signal requires several cycles, and the master must relinquish the bus periodically to allow memory refresh (or do the refresh itself).
$\overline{\text{MEM}}$, CS16, I/O CS16	Signals used by a peripheral board to tell the motherboard that it is capable of handling a 16-bit data transfer.

9.3.3 EISA Bus Signals

EISA introduces the following major advances:

- — Memory capacity greater than 16 megabytes
- — Support of multiple bus master devices with high-speed burst transfer rates
- — 32-bit data transfers for CPU, DMA and bus master devices
- — Enhanced Direct Memory Access (DMA) arbitration and transfer rates
- — Programmable edge-or level-triggered (shareable) interrupts
- — Automatic configuration of system and expansion boards.

Compatibility with ISA

The EISA maintains full compatibility with the existing industry standard. The signals required for the EISA bus enhancements are provided through a superset expansion board connector on the system board. In this way, existing 8- and 16-bit boards can be installed in EISA slots. The bus functional and performance enhancements, such as DMA improvements, are provided similarly via superset features with the default bus operations remaining fully ISA compatible. Because EISA machines maintain full compatibility with existing ISA boards, EISA products do not require newly designed boards. New expansion boards need to be developed only for those future applications that require access to the high-performance ElSA functions.

Following are additional signal lines added to ISA to create EISA

Lines	**Description**
$\overline{\text{BE0}}$ – $\overline{\text{BE3}}$	Byte enables. These signals indicate which byte lines of the 32-bit data bus are involved in the current bus cycle. They're analogous to the BE0 through BE3 signals on the 80386 and 80486 microprocessors.
$\overline{\text{M}}$ /IO	Distinguishes between an EISA bus cycle and EISA I/O cycle.
$\overline{\text{START}}$	Indicates the start of an EISA bus cycle.
$\overline{\text{CMD}}$	Provides timing control within an EISA bus cycle.
$\overline{\text{MSBURST}}$	Indicates that a master is capable of performing burst cycles.
$\overline{\text{SLBURST}}$	Indicates that slave is capable of accepting burst cycles.
EX32, EX16	Indicate that a slave is an EISA board and can support a 32- or 16-bit cycle, respectively. If neither of these signals is asserted at the beginning of a cycle, the bus falls back to an ISA-compatible mode for that cycle.
EXRDY	Indicates that an EISA slave is ready to terminate a cycle.
$\overline{\text{MREQn}}$	Asserted by potential master *n* for requesting the bus.
$\overline{\text{MAKn}}$	Indicates to master *n* that it has been granted the bus.

D16–D31 — The new data lines that, combined with the data lines on the ISA bus, make the EISA 32-bit databus.

LA2–LA16, LA17–LA31 — New address bus lines. Like LA17-LA23, these lines aren't latched on the motherboard and so provide a fast path to the peripheral boards. Note that there's no need for an LA1 or LA0; the byte enable lines are used. Also note that there are now 32 address bits, supporting the full address range of the 80386 (rather than the 24-bit address range of the 80286). This lets system RAM grow above 16 megabytes.

9.4 INTERRUPTS

The interrupt structure is supported by two ways in the PC environment. The main microprocessor (8088) has two pins for interrupt requests: NMI, i.e. Non-Maskable Interrupt and INTR Maskable Interrupt.

There are three sources of NMI interrupt in PC:

— a base board RAM parity error;
— an auxiliary processor (floating point processor 8087) interrupt request;
— an I/O channel check request.

Since NMI cannot be masked internal to 8088, the PC does the masking/unmasking externally. An I/O register port bit on base board is used for the purpose.

Logically this can be represented as shown in Fig. 9.2.

— If I/O register port bit is set and if interrupt request is available on one of the three lines, then NMI is set.
— If I/O register port bit is reset, then NMI is reset (Here, Set = 1 and Reset = 0).

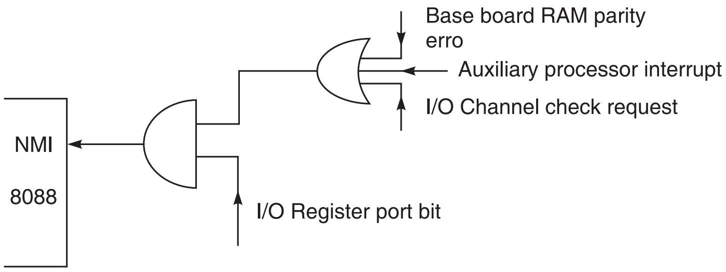


Figure 9.2 Non-maskable interrupt logic.

The maskable INTR interrupt line is used to support the multiple source interrupts. The complete interrupt management is carried out by a dedicated programmable interrupt controller (PIC) device (Intel 8259). This device expands the interrupt source to 8 different independent levels, designated as IRQ0 to IRQ7. Out of these 8 interrupt lines, the following are used by the system internally for the resource management purpose:

IRQ0—Output of internal timer 8253-5 channel-0

IRQ1—Keyboard scan code interrupt

IRQ3—RS-232C serial port

IRQ4—RS-232C serial port

IRQ5—Parallel printer port

IRQ6—Diskette DRV status

IRQ7—Parallel printer port

The interrupt lines IRQ2 to IRQ7 are available on the system bus. The PC may not use IRQ4, IRQ6 or IRQ7, if the particular function is not used. External devices or Interfaces can make use of these interrupts.

9.4.1 PC-AT Interrupts

In the PC-AT design the microprocessor NMI and two 8259 interrupt controller chips provide 16 levels of maskable system interrupts. The following shows the interrupt level assignments in decreasing order of priority.

Level	**Function**
Microprocessor NMI	Parity or I/O channel check
Interrupt controller-I	
IRQ0	Timer output-0
IRQ1	Keyboard
IRQ2	Interrupt from controller-2
IRQ3	Serial port-2
IRQ4	Serial port-l
IRQ5	Parallel port-2
IRQ6	Diskette controller
IRQ7	Parallel port-1
Interrupt controller-2	
IRQ8	Real-time clock interrupt
IRQ9	Software redirection to INT 0AH
IRQ10	Reserved
IRQ11	Reserved
IRQ12	Reserved
IRQ13	Coprocessor
IRQ14	Fixed Disk Controller
IRQ15	Reserved

9.4.2 EISA Interrupts

EISA allows each interrupt to be individually configured for level- or edge-triggered operation. Edge-triggered operation provides full compatibility with existing interrupt driven ISA devices.

Level-triggered operation facilitates the sharing of a single system interrupt by a number of devices. This mode of operation might be used, for example, on a fully loaded system to share a single interrupt between a number of serial ports in a multiserial-port board.

9.4.3 Interrupt Controller

Figure 9.3 shows the block diagram of 8259 interrupt controller. The four functional blocks namely Interrupt Request Register, Priority Resolver, In Service Register and Interrupt Mask Register can be programmed by the designer. The initialization and programming of 8259 interrupt controller can be done by microprocessor by sending Initialisation Command Words (ICWs) and Operation Command Words (OCWs).

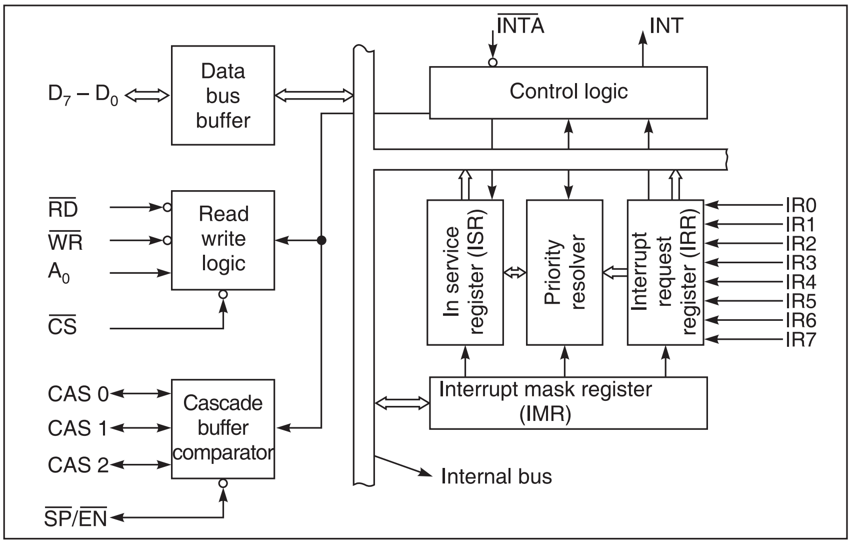


Figure 9.3 Block diagramof an 8259 interrupt controller.

The Interrupt Request Register accepts the 8 interrupts and latches them. An interrupt can be level triggered or edge triggered. All the 8 interrupts can be programmed to be edge triggered or level triggered by setting bit-3 of ICW1 (Initialisation Command Word 1). The Interrupt Mask Register is an 8-bit register which can be programmed to mask or unmask any interrupt request to the controller by setting the bits of OCWl (Operation Command Word 1).

The Priority Resolver takes as input the pending interrupt request from Interrupt Request Register and selects an interrupt request based on the priority mode, set by programming the OCW2 (Operation Command Word 2). The In Service Register stores the priority level of interrupt which is being serviced presently. It stores the result of priority resolver.

The 8259 controllers can be cascaded to expand the number of interrupts. Though the PC bus doesn’t contain the cascade interface lines, it is possible to achieve interrupt expansion by using 8259 in cascade mode externally.

Modes

Since 8259 is a general purpose interrupt controller, a number of facilities have been provided by the designers to facilitate its use in different application environments. This chip can be interfaced to MCS 80, MCS 85, iAPX 86 or iAPX 88 processors. The initialisation and programming is done by host processor using Initialisation Command Words (ICWs) and Operation Command Words (OCWs). The 8259 can be operated in one of the following modes:

— Fully nested mode
— Rotating priority mode
— Special mask mode
— Polled mode.

(i) *Fully nested mode.* This is the default mode for 8259, i.e. if no other mode is programmed, the 8259 works in fully nested mode. The interrupt lines are on descending order of priorities starting from IRQ0 (highest) to IRQ7 (lowest). The priority resolver determines the highest priority interrupt request based on the above criteria. The highest priority interrupt vector is placed on the bus during acknowledgment sequence. A bit corresponding to interrupt level selected is set in In-Service Register. This bit when set, inhibits all further interrupts of same or lower priorities.

(ii) *Rotating priority mode.* Often a designer is faced with a situation in which all the interrupt request sources are having the same priority. In such cases if a device has interrupted the processor, then it must wait for other devices to lodge their interrupt requests and get serviced before it can interrupt again. In such cases, the interrupt source after service must be placed lowest in the priority list. Thus the priorities of different interrupt sources, must be rotated after each service of an interrupt. There are two types of rotating priority mode: (a) automatic rotation and (b) specific rotation.

In Automatic rotation mode the interrupting device receives lowest priority automatically as soon as it is serviced. Thus in a worst case it may have to wail till all other seven devices have been serviced.

In Specific rotation mode, the bottom priority can be programmed using “Set Priority” command. By fixing the bottom priority to a particular interrupt level, the priorities of other levels can be set. In this case, if the IRQ6 is fixed to bottom priority then IRQ7 will be having highest priority, IRQ0 next lower and so on.

(iii) *Special mask mode.* Some applications require that user programme must be able to mask or unmask different interrupts. The special mask mode allows user to change the interrupt priorities by setting bits in Interrupt Mask Register (IMR) thus enabling some interrupts individually under software control.

When special mask mode is entered, bits in OCWl which act as IMR are examined by 8259 and only those interrupts which are not masked are enabled.

(iv) *Polled mode.* In the polled mode INT output and $\overline{\text{INTA}}$ input of 8259 are not used. The microprocessor polls the 8259 for any possible interrupt request. The 8259 examines the IS register and selects an appropriate interrupt. The level of this interrupts is sent to the microprocessor on the data bus.

The poll command is issued by microprocessor by setting P = 1 in OCW3. The 8259 waits for $\overline{RD}$ (Read) pulse to come from microprocessor. The $\overline{RD}$ pulse when received is treated as $\overline{INTA}$ by 8259. The 8259 sets the appropriate IS bit if there is interrupt request.

The microprocessor receives the IS word, and determines the level of interrupt request. The location of Interrupt Service Routine is derived by microprocessor using the value of W0–W2 and the corresponding ISR is invoked.

This mode is useful when it is required to expand the number of interrupt priority levels to more than 64. Another application may be, when similar tasks are to be performed in response to different interrupts. Let us assume as an example that in response to each level of interrupt a display of interrupt and its level is done at first instant, then parameter values are read from different memory locations depending on the level of interrupt and finally, specific actions relating to each interrupt level are performed. Since the ISR locations are determined by software in microprocessor, it is possible to allocate same memory address for ISR of all similar interrupts. The program flow would look like as shown in Fig. 9.4. This approach results in saving of memory space without any overhead.

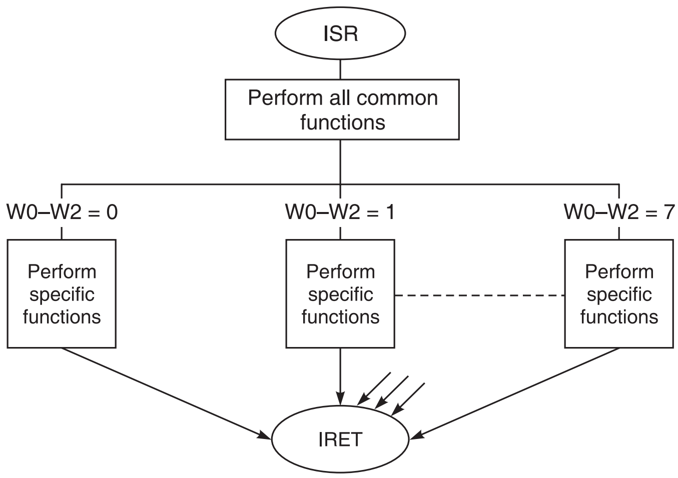


Figure 9.4 Polled mode program flow.

9.4.4 Cascading of 8259

In a large system which has more than eight interrupt sources, more than one 8259 is required to be used in cascade fashion. The 8259 provides facilities to allow it to be used in cascade fashion. The cascading is achieved through master-slave configuration, i.e. the first level 8259, which is connected to microprocessor through Interrupt Request signal is designated as Master. All the second level (maximum 8) 8259's which submit their interrupt requests signal to master 8259 are designated as slaves. The cascade configuration of 8259 is shown in Figs. 9.5 and 9.6. Figure 9.5 shows the interfacing between more than one 8259 with microprocessor, while Fig. 9.6 shows the interconnection between different 8259's.

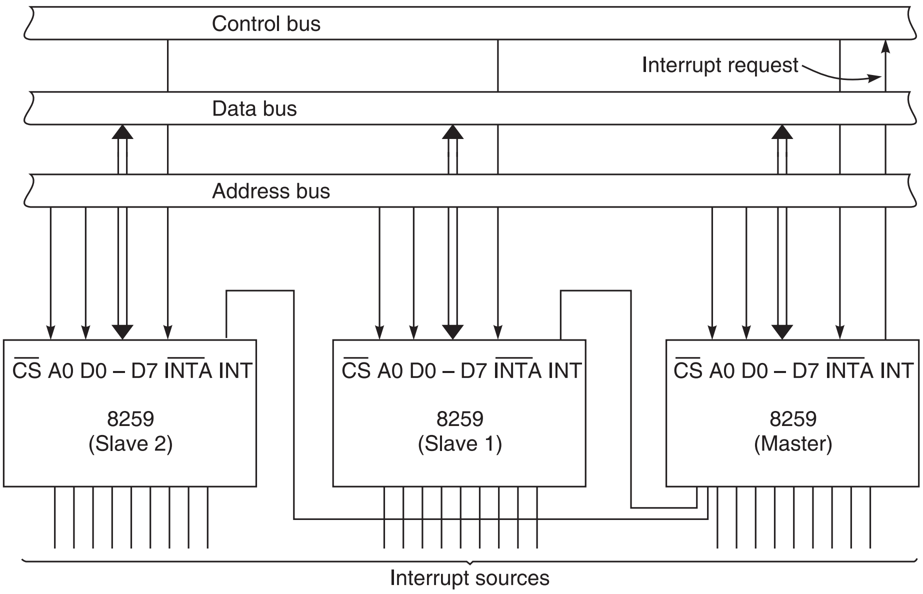


Figure 9.5 Interfacing of 8259 with microprocessor.

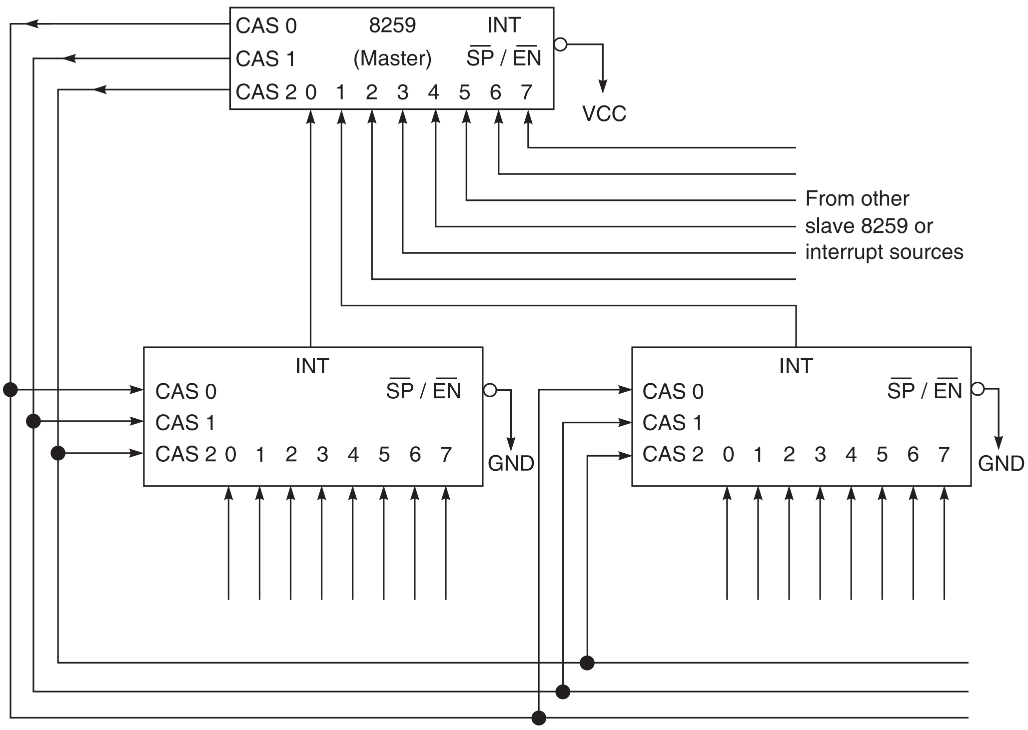


Figure 9.6 Cascading of 8259 with microprocessor.

Cascade modes

There is no restriction on master or slave 8259 regarding modes in which they can work when cascaded. In a system containing one master and eight slave 8259, it is possible to use different modes for different slaves and master by programming them appropriately. Depending on the mode set, each slave examines the pending interrupt requests by looking at Interrupt Request Register, and selects an interrupt request for sending it to master. The bit pertaining to the selected interrupt is set in In-Service Register. The master 8259 examines all the interrupt requests received from different slaves, and then based on mode set, selects an interrupt for sending to microprocessor.

The above treatment of different modes of 8259 makes this very clear that in cascade mode only fully nested mode has some limited utility. Other modes do not work or put so much of software overhead that programming becomes difficult and system becomes totally inflexible. We have mentioned that fully nested mode has some limited utility. The limitation is imposed due to the fact that once microprocessor is processing the interrupt originating from a slave 8259, another interrupt from same slave will not be recognised by the master even when the later interrupt is of higher priority than the earlier interrupt. Thus an ISR cannot be interrupted by an interrupt from same slave. This poses severe limitation on the system designer and programmer.

To circumvent these limitations, 8259 allows Special Fully Nested Mode when cascaded. In this mode, when an interrupt request from a certain slave is in service, this slave is not locked out from master's priority logic, and further interrupt requests from higher priority interrupt levels within the slave will be recognised by the master and will initiate interrupts to the processor.

9.4.5 Programming the 8259

Before normal operation can begin, each 8259 in the system must be brought to a starting point. This is performed by microprocessor by issuing Initialisation Command Words (ICWs) to 8259.

Once the 8259 has been initialised, it can be made to operate in various interrupt modes by microprocessor by using Operation Command Words (OCWs). The OCWs can be written into the 8259 any time after initialisation.

Initialisation command words (ICWs)

The ICWs format is shown in Fig. 9.7(a-e). The initialization sequence starts with ICWl followed by ICW2, and depending on requirement may be followed by ICW3 and ICW4.

The ICW3 is required only in case of cascaded 8259 which is determined from SNGL (bit-l of ICW1). The initialisation sequence is shown in Fig. 9.8. The initialisation command words are self explanatory.

Operation command words (OCWs)

The OCWs format is shown in Fig. 9.9(a-c). OCWl sets and clears the mask bits in Interrupt Mask Register (IMR). In OCW2, the three lower bits L2, L1, L0 indicate the interrupt level to be acted for specific EOI or specific rotation. The R,SL, and EOI bits control the rotate and EOI modes and their combinations.

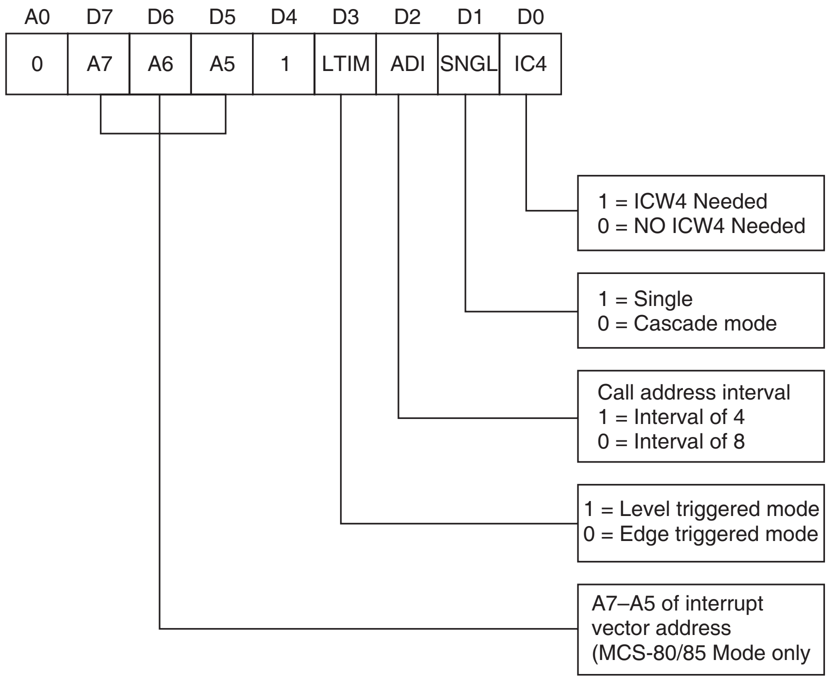


Figure 9.7(a) Initialisation command word—1.

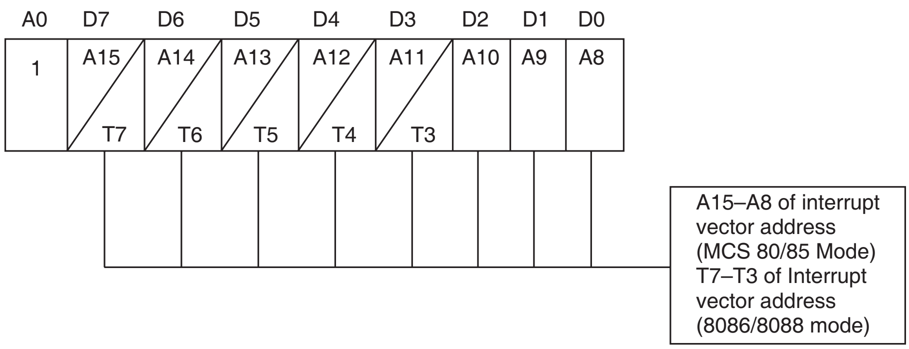


Figure 9.7(b) Initialisation command word—2.

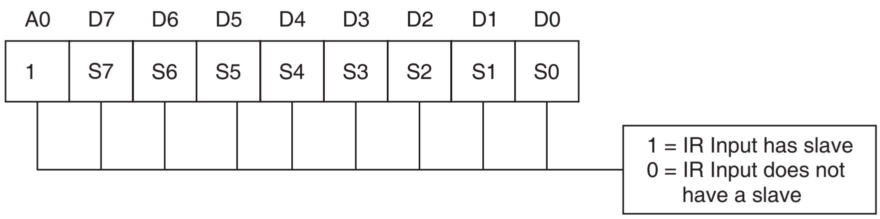


Figure 9.7(c) Initialisation command word—3 for master device.

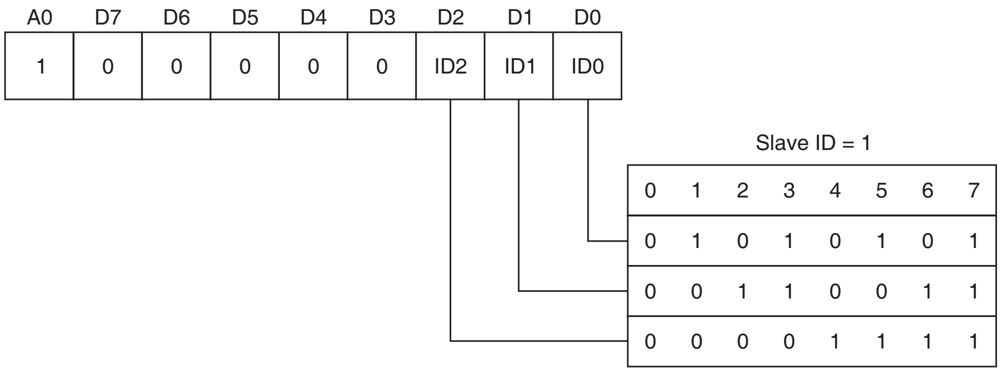


Figure 9.7(d) Initialisation command word—3 for slave device.

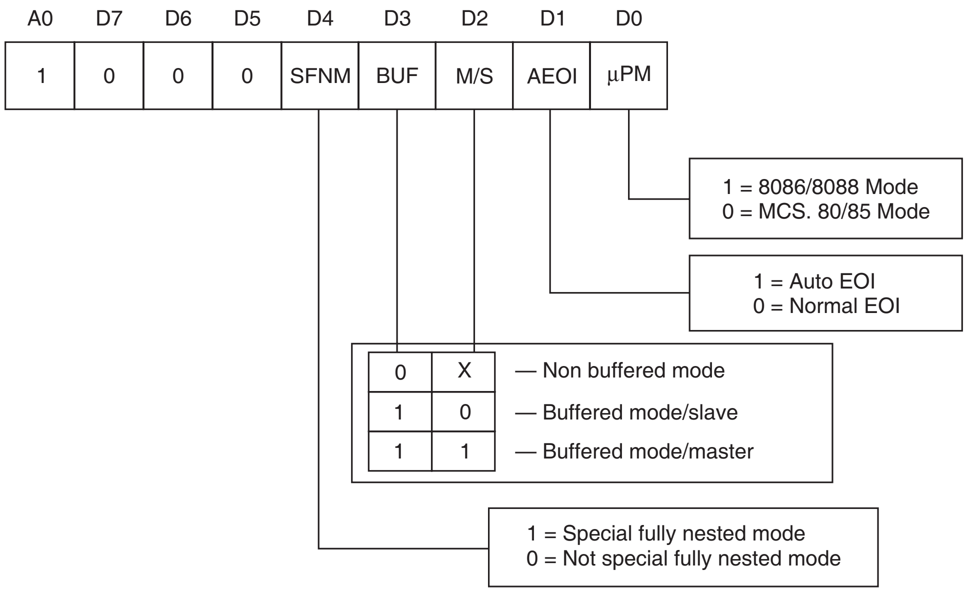


Figure 9.7(e) Initialisation command word—4.

Using OGW3 the microprocessor can,

— read IR register;

— read IS register;

— send poll command to 8259;

— set special mask;

— reset special mask.

The ESMM bit enables the functioning of SMM bit.

9.4.6 Interfacing of 8259

The interfacing of 8259 at block level is described in Fig. 9.10. The instructions used for reading from and writing to 8259 are IN ⟨port⟩ and OUT ⟨port⟩. The equivalent instructions in C language are *in p* (*portid*) and *out p* (*portid*). The address lines are decoded using decoding techniques to select the 8259 through chip select $\overline{CS}$ signal. The decoding techniques and interfacing is discussed in detail later.

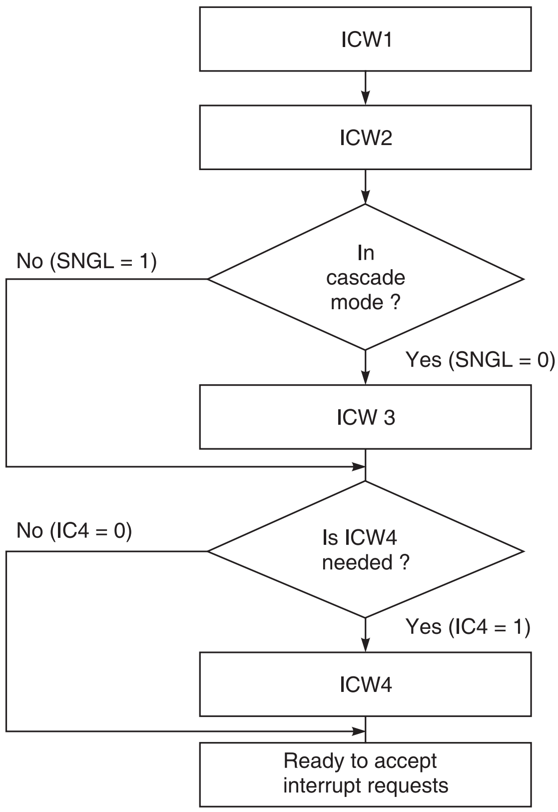


Figure 9.8 Initialisation sequence.

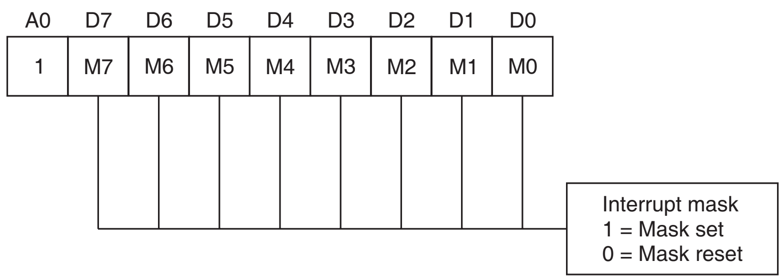


Figure 9.9(a) Operation control word—1.

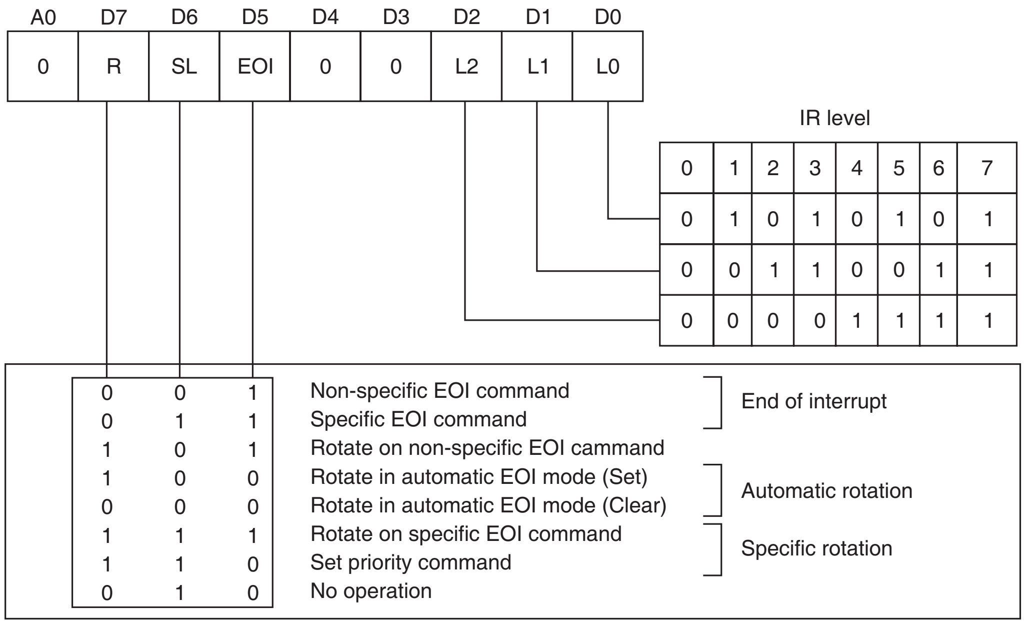


Figure 9.9(b) Operation control word—2.

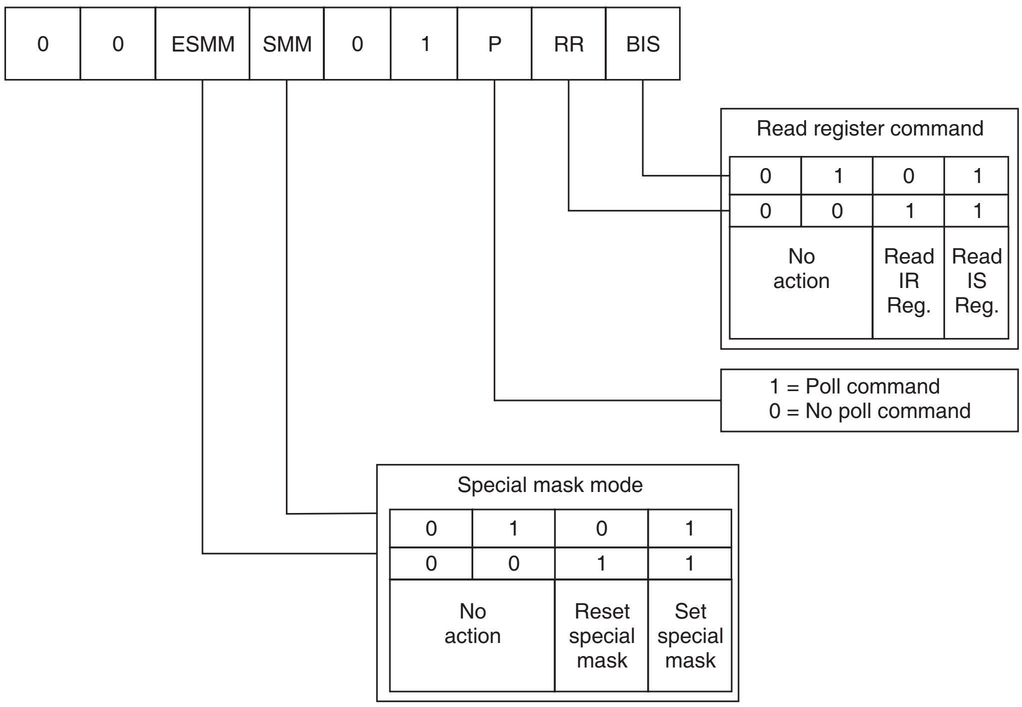


Figure 9.9(c) Operation control word—3.

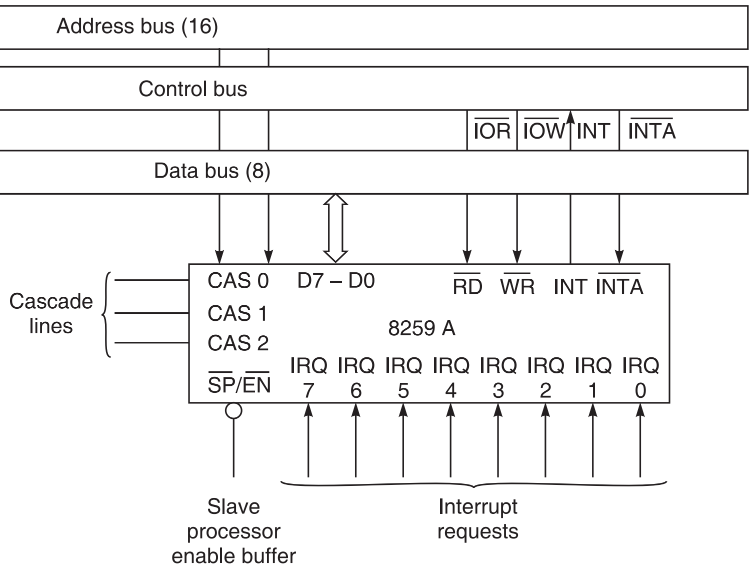


Figure 9.10 8259A interface to standard bus system.

9.4.7 Role of 8259 in Personal Computer

Figure 9.11 shows the vector pointers of the 8259 interrupt controller in PC low memory. The vector table of 8259 interrupt controller has been shown In Fig. 9.12. The IRQ0 is located at 0020(H) and IRQ7 is located at 003C(H). The 8259 Interrupt controller can be addressed through I/O port addresses 0020 (H) and 0021(H) as follows.

Port Address	Register	
0020 (H)	ICW1	Initialisation Mode
0021 (H)	ICW2	
0021 (H)	ICW3	
0021 (H)	ICW4	
0021 (H)	OCW1	Operation Command Mode
0020 (H)	OCW2	
0020 (H)	OCW3	

The Initialisation mode can be entered by writing ICWl to port address 0020. Three more bytes corresponding to ICW2, ICW3 and ICW4 must follow before Operation Command Mode is entered. These three bytes are written in sequence to I/O port address 0021 (H).

As described earlier and as shown in Fig. 9.8, the ICW3 or ICW4 may or may not be required in the system. We shall see later that in PC, ICW3 is not used.

The operation command mode is entered at the end of initialisation mode. The OCW1 is written at I/O port address 0021 (H). This is followed by OCW2 and OCW3, both at I/O port address 0020 (H). Where in OCW2, D4 = 0 and D3 = 0; in OCW3 D4 = 0 and D3 = 1 to

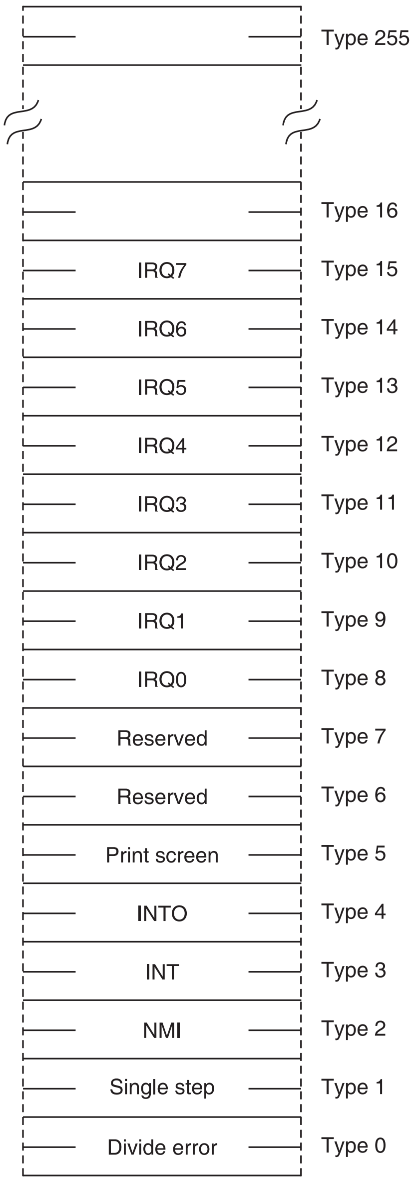


Figure 9.11 Vector pointers of 8259 interrupt controller in PC low memory.

differentiate between the two. The bit definitions of ICWs and OCWs have already been covered earlier. The PC BIOS sets following on initialisation.

(a) ICW 1 = 13 (Hex)
   (i) ICW4 required
   (ii) Single mode

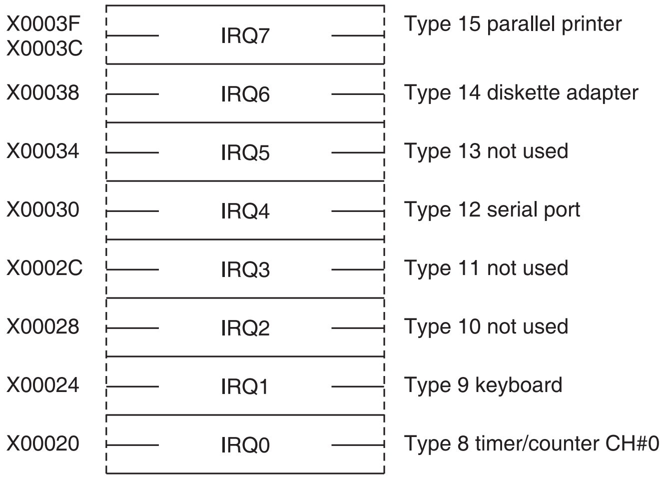


Figure 9.12 Vector table of 8259 interrupt controller.

(iii) No call address interval in 8088
(iv) D4 = 1 signifies that byte is ICW1.

(b) ICW2 = 08 (Hex.)
(i) 8088 mode
(ii) (D3 to D7 correspond to A5 to A9)
A9 A8 A7 A6 A5 = 00001 (B)
i.e. vector table of 8259 controller is located at 0020 (H).

(c) ICW3—Not used. Since Dl in ICWl is one signifying no cascade mode, 8259 skips to ICW4 after ICW2

(d) ICW4 = 09 (Hex)
(i) 8088 mode
(ii) No Automatic EOI
(iii) Single 8259 in master mode
(iv) Buffered Data Bus System
(v) Single controller so no Special Fully Nested Mode

The OCWl, OCW2 and OCW3 can be used to perform different functions like,

OCW1—Interrupt masking and unmasking.

OCW2—Rotate and change the priority of Interrupt requests and to select the EOI mode to be used.

OCW3—Read IRR and ISR, issue poll command, set and reset special mask.

The poll command and special mask is not used in PC-BIOS.

9.4.8 Interrupt Expansion on PC

On examining the interconnection of 8259 in cascade mode in Figs. 9.5 and 9.6, we find the use of following signals:

A0	—	Bit 0 of address bus
$\overline{CS}$	—	Chip select signal derived by decoder using the address bus.
D0–D7	—	Data bus
INT	—	Interrupt output of 8259
CAS0–CAS2	—	Cascade lines on which the master releases the slave address.
$\overline{SP}/\overline{EN}$	—	Input signal to designate a 8259 as master or slave by connecting VCC or GND signal.
IRQ0–IRQ7	—	Input interrupt request from source
$\overline{RD}$	—	Read control signal from microprocessor
$\overline{WR}$	—	Write control signal from microprocessor
$\overline{INTA}$	—	Interrupt acknowledgement signal to 8259

To cascade 8259, the presence of above signals is necessary. Close examination of PC Bus will reveal that $\overline{INTA}$ and CAS0-CAS2 signals are not present.

The $\overline{CS}$ is derived from address bus for each 8259, using decoding circuits which will be discussed later. It is thus evident that the 8259 on PC base board cannot be connected in the cascade mode as depicted in Fig. 9.6. However, it does not mean that the users of PC in real-time mode will have to be contented with maximum six or minimum two interrupts discussed earlier.

While discussing the poll mode we had indicated that $\overline{INTA}$ signal is not used. Following are the activities in the poll mode:

1. The 8259 selects an interrupt depending on the pending interrupt requests. The interrupt request is made to microprocessor on INT line.
2. The microprocessor accepts the interrupt request and executes Interrupt Service Routine. The structure of ISR is shown in Fig. 9.4.
3. As a first step in ISR, a poll command is issued lo 8259. The 8259 accepts the poll command as $\overline{INTA}$.
4. The microprocessor performs read operation on the 8259. The 8259 sends the level of selected interrupt to microprocessor in response to $\overline{RD}$.
5. The software jumps to the sub-level to process the interrupt.
6. Before exit, the ISR EOI command is issued to reset the bit in IS register.

The interrupt expansion in personal computer follows the same concept but the hardware/software structure is different. The hardware structure of interrupt expansion in PC is shown in Fig. 9.13, whereas the structure of a ISR is shown in Fig. 9.14 (a–b). Since the base board 8259 is interfaced to the microprocessor using $\overline{INTA}$, the poll mode is necessary only for second Level 8259. Following are the activities:

1. All the second level 8259s select interrupt requests depending on the mode set and pending requests. The 8259s send the interrupt requests to base board 8259 on INT lines.

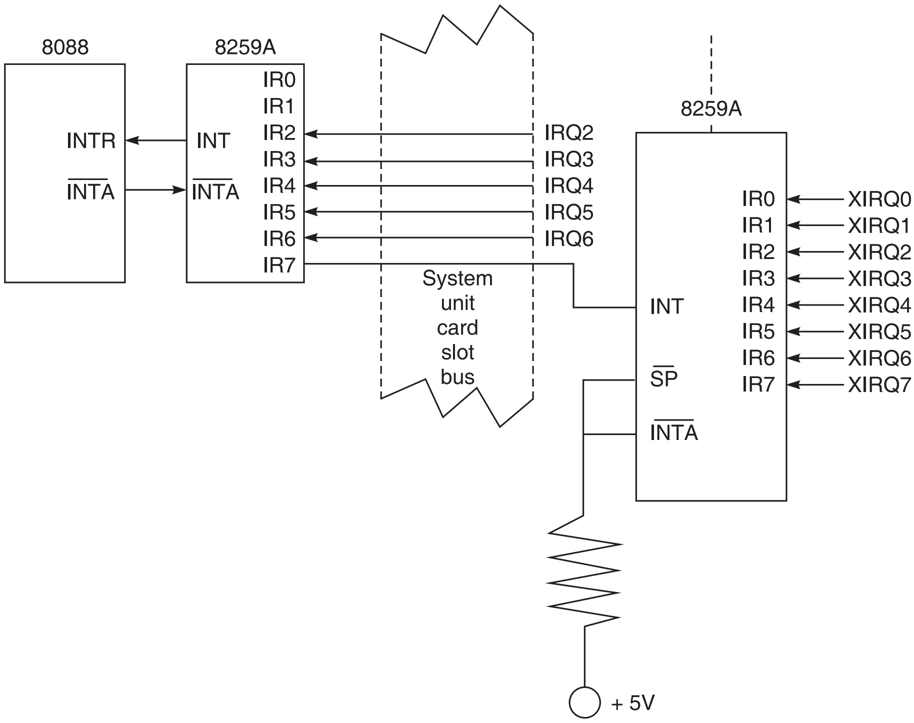


Figure 9.13 Block diagram of interrupt expansion circuit.

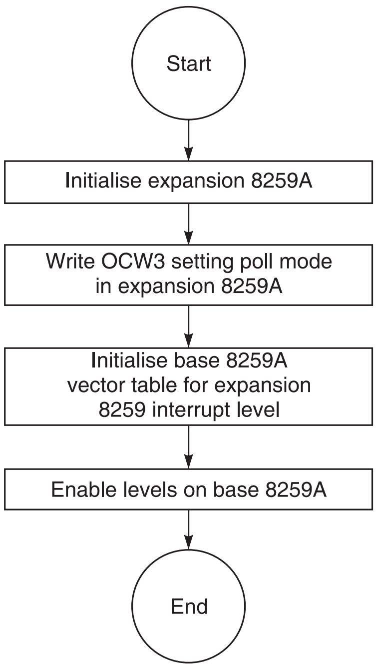


Figure 9.14(a) Initialisation for 8259 expansion.

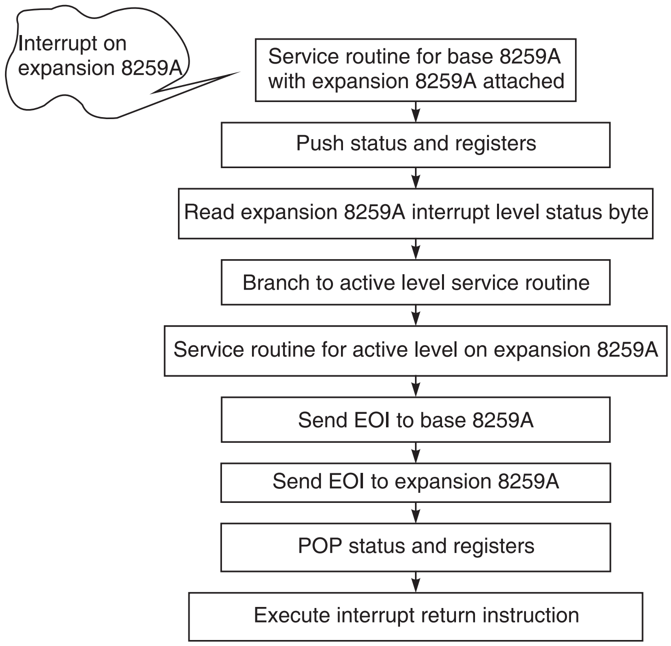


Figure 9.14(b) Service routine for the expansion 8259A device.

2. The base board 8259 examines the input interrupt requests from second level 8259s and depending on the mode, selects an interrupt. The interrupt requests is made to 8088 on INT line.
3. The 8088 sends $\overline{\text{INTA}}$ pulse and base board 8259 sends pointer to the vector table. The 8088 executes the ISR.
4. The poll command is issued to the particular second level 8259 connected to the interrupt level selected in base board 8259.
5. The ISR performs read operation on the second level 8259 which sends the level of interrupt request selected.
6. ISR jumps to the appropriate sub-level to process the interrupt.
7. Before exiting, two EOIs are issued for both the levels of 8259s.

Using this technique, the total number of interrupt can be expanded to $6 \times 8 = 48$ (only IRQ2–IRQ7 are available on Bus). To expand the number of interrupts beyond 48, one more level of 8259s can be connected. That means, each of the 48 interrupt lines to second level 8259s can be connected to one 8259 at third level. Using these 48 additional 8259s, the total number of interrupts will be expanded to $48 \times 8 = 384$. There will be two-level polling as against one level polling described above. Figure 9.15 shows the hardware inter-connection. The software needs little change and can be designed easily.

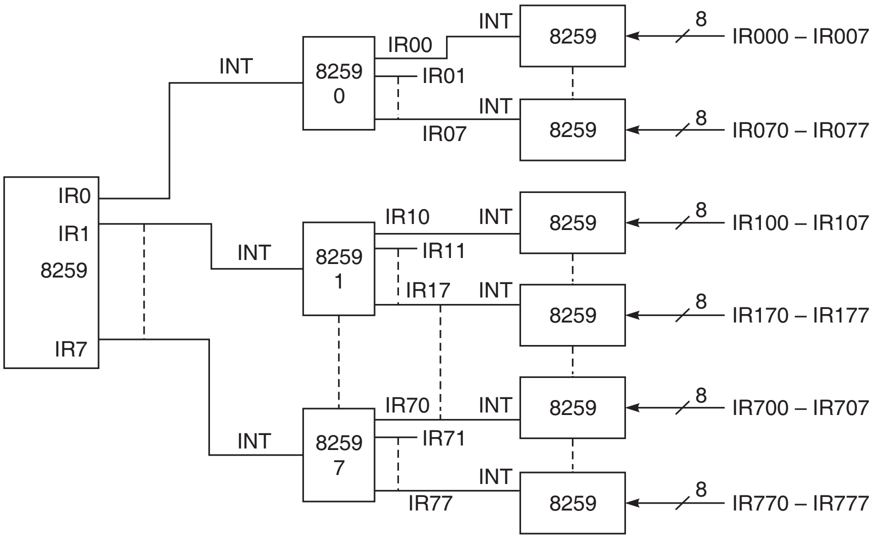


Figure 9.15 Two-level polling for interrupt expansion.

9.5 INTERFACING PC TO OUTSIDE WORLD

Assuming that the readers are familiar with interfacing techniques using microprocessor, a very brief and general overview is presented herewith.

9.5.1 Memory Mapped vs. I/O Mapped Interfacing

The I/O mapped interfacing technique treats the I/O addresses and memory addresses separately. The I/O addresses are known as port numbers. A port is defined as an interface between actual I/O device and microprocessor. From hardware point of view, a port is a simple register or a shift register. It may reside in control unit of I/O device or may be outside, implemented separately. The generation of port is shown in Fig. 9.16.

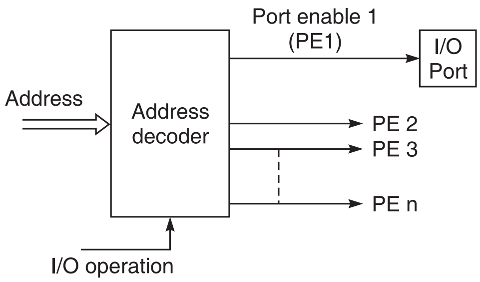


Figure 9.16 I/O mapped I/O.

The microprocessors which offer I/O mapped interfacing have instructions to enable reading from and writing to the ports. These instructions are different from memory read/write.

Whenever I/O read/write instructions are executed, the microprocessor generates 'I/O operation signal' which enables the address decoding for generation of particular port enable signal. The 8088 has IN and OUT instruction to read and write to any port respectively. The 'I/O operation signal' is called $IO/\overline{M}$ signal. When $IO/\overline{M} = 1$, the read/write is from device and when $IO/\overline{M} = 0$ the read/write is from memory.

The memory mapped interfacing on the other hand, treats I/O addresses same as memory addresses. The confusion between memory address space and I/O address space is avoided by defining these distinctly so that there is no overlap. This amounts to sacrifice of some memory space for I/O devices. The advantage of this scheme of interfacing is that the powerful memory read/write instructions can be used for I/O read/write also. The interfacing of microprocessor to the I/O device will still be through an ordinary register or a shift register as in case of I/O mapped technique, but these registers will be addressed as memory locations rather than port numbers. The scheme is described in Fig. 9.17.

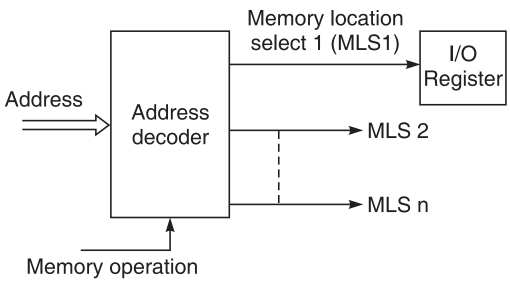


Figure 9.17 Memory mapped I/O.

9.5.2 I/O Address Decoding Techniques

Fixed address decode

The simplest form of decoding is the linear select method. This requires the smallest amount of logic but can be used for fewer input and/or output ports. One address bit is associated with each I/O port and is logically ANDed with $IO/\overline{M}$ and $\overline{RD}$ or $\overline{WR}$ to generate an input or output device select pulse, as shown in Fig. 9.18. However if number of ports is large than 8205 (1 of 8) decoder chip of Intel (Fig. 9.19) or a number of 74139/74138 decoders can be used (Fig. 9.20), to generate large number of device select pulses. Figure 9.21 shows a number of 74154 (1 out of 16) decoder chips used to generate total 256 device select pulses.

A port can be either input port, i.e. microprocessor can read the data from the port or output port, i.e. microprocessor can write the data to the port. Though it is technically possible to use any port for both input and output, but it is not recommended or done as good design practice. It makes the circuit complicated and creates confusion at times. Some I/O devices require handshake protocol for data transfer while some others may require interrupt driven data transfer. Thus every time one needs to interface a device to microprocessor, special circuits are to be developed. So far we have considered the port address generation in a sequence. If the port

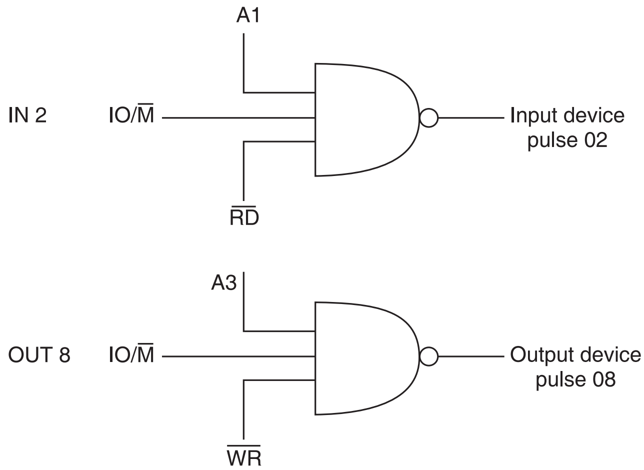


Figure 9.18 Generation of single device select pulse.

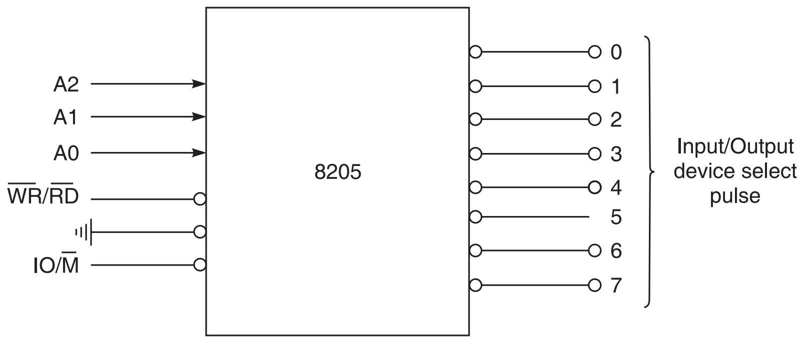


Figure 9.19 Generation of multiple device pulses using 8205.

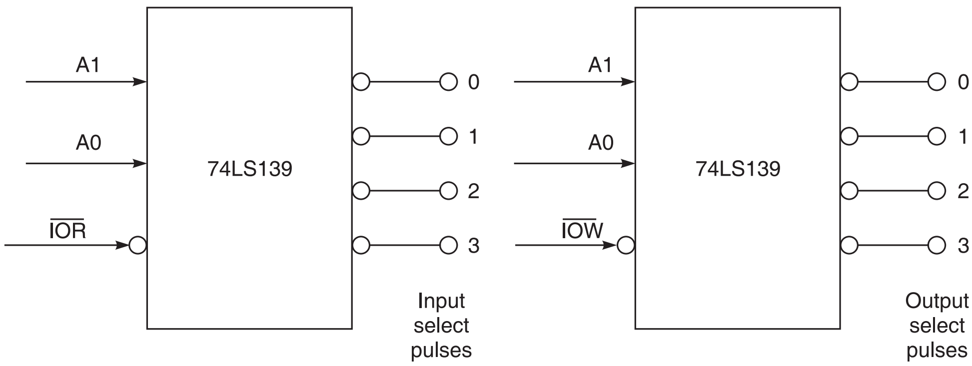


Figure 9.20 Generation of multiple device select pulses using 74LS139.

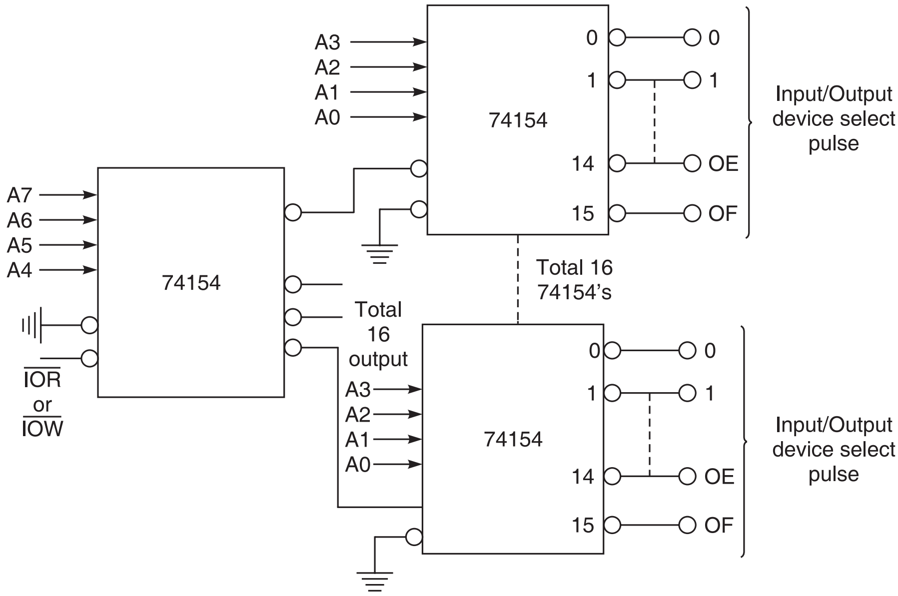


Figure 9.21 Generation of 256 input-output device select pulses using 7A154.

address is selected randomly, then more extensive decoding is required. A group select signal is generated by decoding higher order address bits and then lower order address bits along with $\overline{\text{IOR}}$ and $\overline{\text{IOW}}$ are decoded to generate Write and Read signals for specific ports.

Switch selectable address decode

In case of fixed address decode logic, if the port address is random for which group select signal is to be generated then the circuit becomes complicated. In case, we wish to change the port address due to addition of some other cards or features, then in case of fixed decode logic, the whole circuit has to be re-designed and fabricated. The switch selectable decode technique overcomes these limitations. The port address for group select is set using the switches. This is compared with the contents of higher order lines of address bus and when there is a match, group select signal is generated. This signal is combined with $\overline{\text{IOR}}$, $\overline{\text{IOW}}$ and lower order address bus lines to generate Read/Write signals for individual ports (Fig. 9.22)

PROM *select decode*

This is another alternative to switch select decode technique. An EPROM is programmed to output the select signals for various port addresses which may be combined with $\overline{\text{IOR}}$ and $\overline{\text{IOW}}$ to generate Read/Write signals for various ports.

The address information is presented to the PROM and data byte read represents the particular port which is selected (Fig. 9.23). It may appear to be a wastage of EPROM resources but this

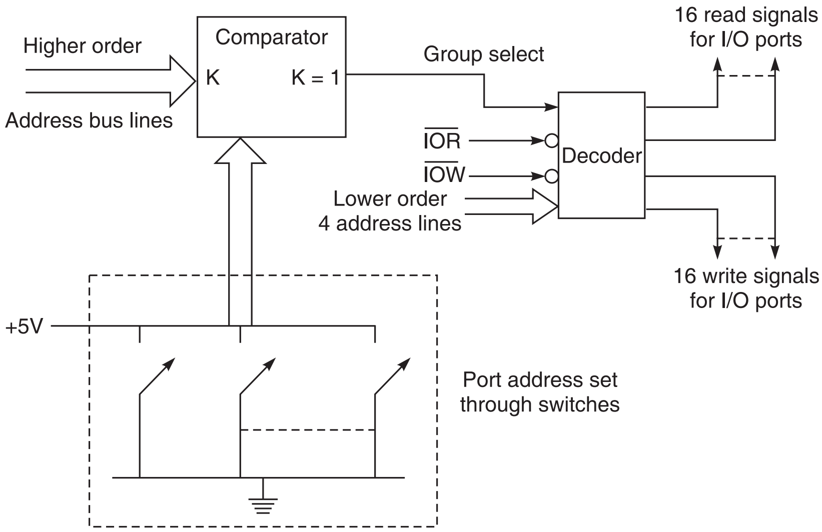


Figure 9.22 Switch selectable decoder.

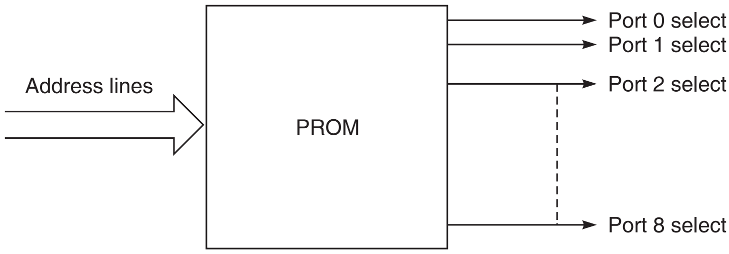


Figure 9.23 PROM select decode.

technique offers maximum flexibility as another EPROM with different port addresses can be put to work in any new environment.

9.5.3 I/O Port Addressing in PC Environment

The 8088 processor sends the port address on lower 16 address lines (A0–A15) out of 20 address lines. Thus total 65,536 unique input and same number of output ports are supported by the processor. The PC however restricts the use of ports addresses to only lower 10 bits (A0–A9). Thus in PC, total 1024 input and 1024 output ports only are supported.

The input ports supported are placed in two categories:

— Those resident on system base board

— Those resident on system bus card slots,

The first 512 input ports can be defined only on system base board. This corresponds to the port address 0000H to 01FFH. The next 512 input ports (address 0200H to 03FFH) can be defined only on the system bus card slots.

Let us decode these 4 address boundaries:

A9	*A8*	*A7*	*A6*	*A5*	*A4*	*A3*	*A2*	*A1*	*A0*
0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1

Thus, when,
A9 = 1—Input port address is on system bus card slot
A9 = 0—Input port address is on system base board

However this restriction does not hold for output ports. Thus output port address in card slots may be from 000H to 03FFH, i.e. any of the 1024 output ports may be defined on card slots. However the output port addresses used by PC on base board must not be used on card slots so as to avoid the selection of both the ports

The port address map of IBM PC is shown in Figs. 9.24 and 9.25. On mother board, the port address 00C0H to 0lFFH are not used. These can be decoded for use as output ports on card slots.

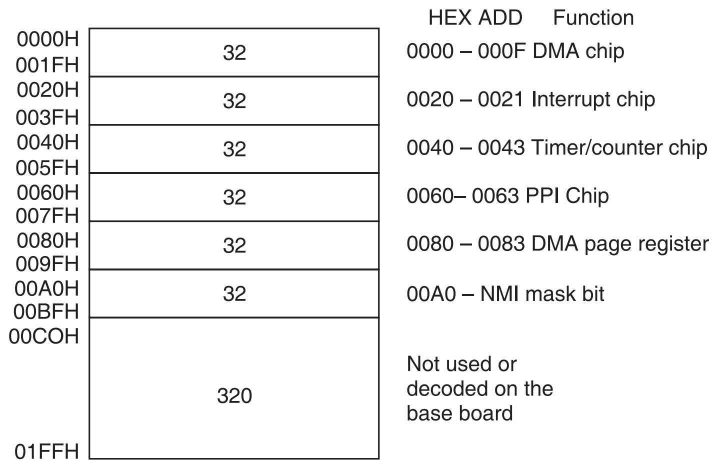


Figure 9.24 Base board I/O port address usage.

The I/O port on PC may be expanded by using the higher order address bits A0 to A15. In order that these port address bits may be used effectively, it must be ensured that two ports are not selected at the same time. If a port has been defined and decoded externally by address F200H, the port with address 0200H if defined will also get selected on the card slot. To circumvent this, we must find one unused port address in the normal port address map. This is decoded using switch selectable technique to generate group select signal. This group select signal is then combined with higher order '6' bits A10—A15 to generate 64 new port addresses. Taking same example, if 0200H is not used in the system card slots, then 64 port addresses

	HEX Adress	Uses
1	0200H	Not used
1	0201H –	Game control adapter
118	0202H – 0277H	Not used
8	0278H – 027FH	Second printer port adapter
120	0280H – 02F7H	Not used
8	02F8H – 02FFH	Second serial adapter card
120	0300H – 0377 H	Not used
8	0378H – 037FH	Printer port adapter
48	0380H – 03AFH	Not used
16	03B0H – 03BFH	Monochrome display adapter
16	03C0H – 03CFH	Not used
16	03D0H – 03DFH	Colour graphics adapter
16	03E0H – 03EFH	Not used
8	03F0H – 03F7H	5 1/4 Inch diskette drive adapter
8	03F8H – 03FFH	Serial prot adapter

Figure 9.25 Card slot I/O port address usage.

starting from 0200H to 0FE00H in which higher order bits are varying from 02 to 0FEH can be generated. Figure 9.26 shows the basic diagram.

The 8088 microprocessor supports a memory address space size of 1 MB. The memory map of PC has been shown in Fig. 9.27. It can be divided into following portions:

1. BIOS ROM upper—8 KB
2. Basic interpreter—32 KB
3. Base board RAM 64 KB containing DOS, DEBUG and BASIC
4. Display Buffer
   (a) colour graphic card adapter
   (b) monochrome display adapter
5. RAM expansion in I/O channel—576 KB card slot.

The function of port is not to process the data but only to receive the data from microprocessor in response to OUT command and only to send the data to microprocessor in response to IN command. The IN and OUT commands are translated into $\overline{IOR}$ and $\overline{IOW}$ signals respectively which are combined with port address-by decoding circuit to generate **port read** and **post write** signals respectively. The decoding techniques have already been discussed earlier.

Figure 9.28 shows the simplistic view of input port. The input port is like a transparent buffer which on receipt of read signal, transfers the data input from outside device to the

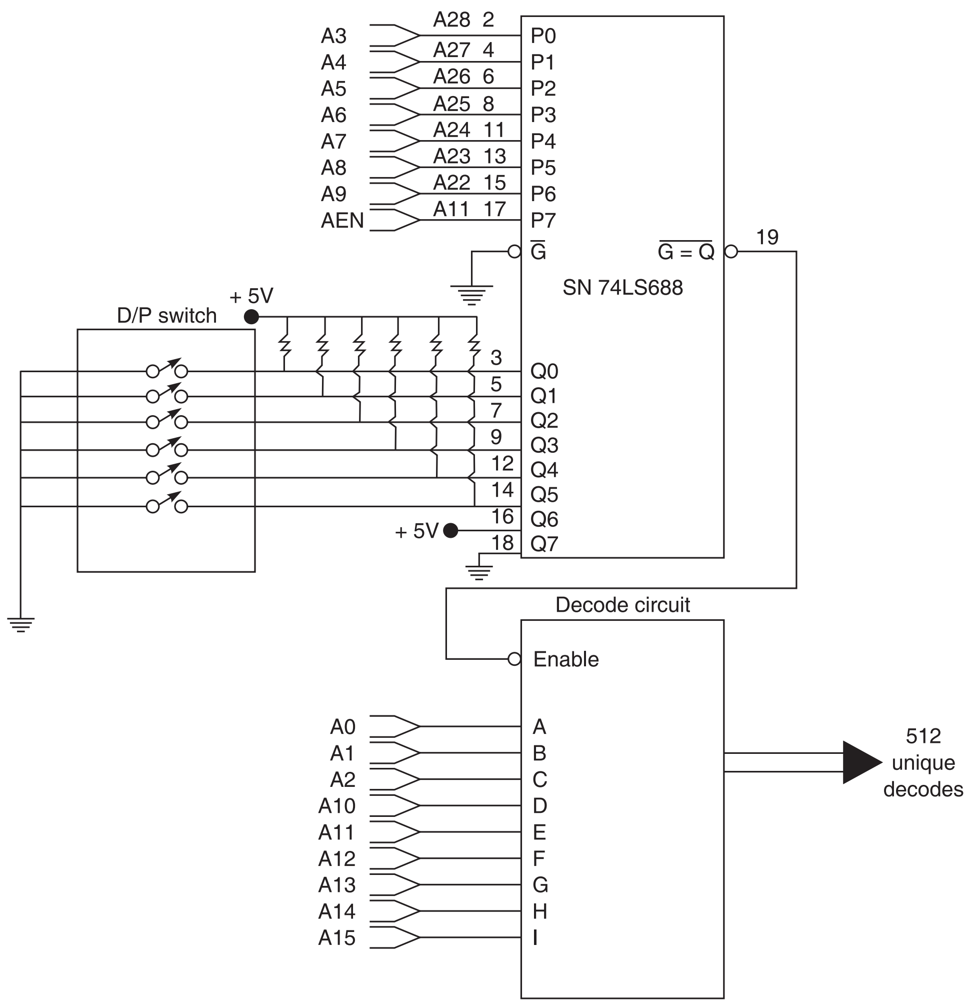


Figure 9.26 Extending port address.

microprocessor databus. The data is accepted from the databus and stored in the AL register of 8088 microprocessor. The simplistic view of output port is shown in Fig. 9.29. The output port is a latch which captures the data sent by microprocessor OUT instruction.

Figure 9.30 shows an output port implemented by 74273 an octal D-type latch. The data ready signal indicates to the output devices that new data has been loaded to the port. Since the output port may be connected directly to alarm annunciators, motors, valves etc. It is necessary to reset the bits on "power on". The bus signal RESET is connected to CLR for this purpose.

Example: Assume that an input port address is 0F00H. The instruction

MOV DX, OFOOH

IN AL, DX,

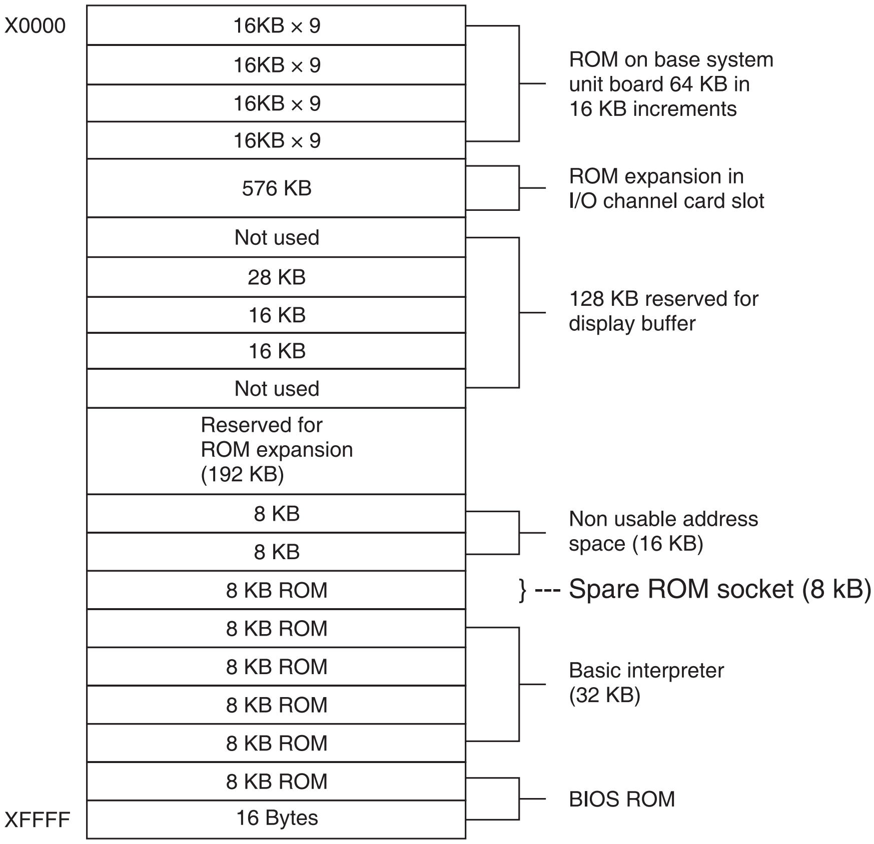


Figure 9.27 PC memory map.

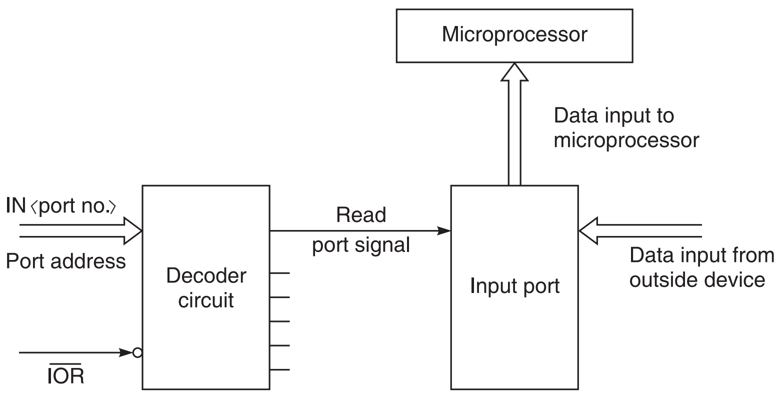


Figure 9.28 Input port.

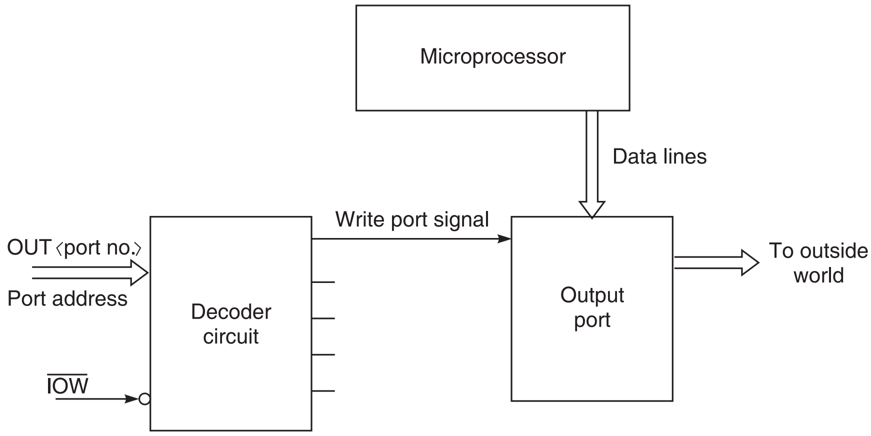


Figure 9.29 Output port.

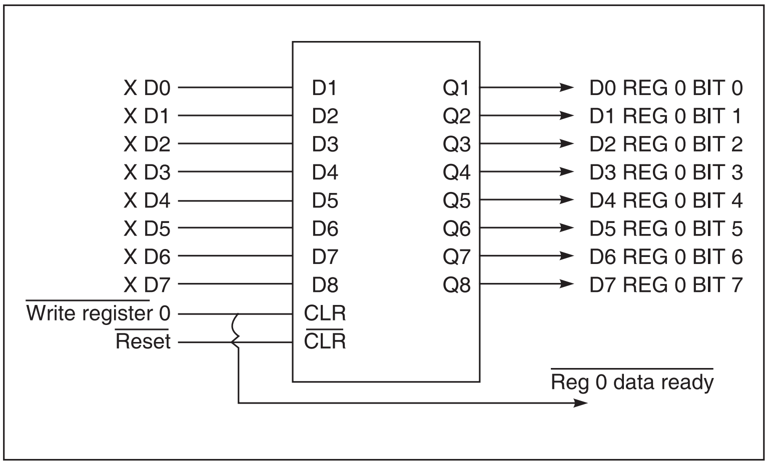


Figure 9.30 Latched D0 register for output port.

will read the data from port no. 0F00H and store in AL register. Similarly, if Output port address is 00F0H then,

MOV AL, 0FFH
MOV DX, 00F0H
OUT DX, AL,

will make all the bits of output port no. 00F0H as one (1).

The input port can be made to sense level, pulse or transition by designing special circuits. Similarly, output port may have built-in shift register, counter, etc. and special circuits have to be designed to accomplish these. We shall however not go in to the design aspects of more complicated input and output ports.

The address decode of PC (0300H-031FH) for prototype cards is shown in Figs 9.31 and 9.32. The decode lines can be used for ADC, DAC, OE. etc.

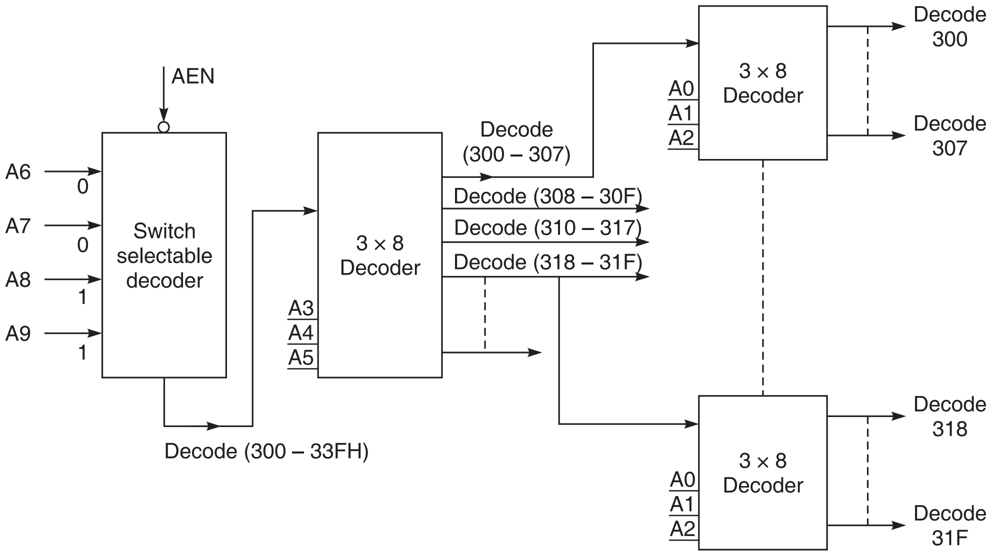


Figure 9.31 Switch selectable decode for interfacing to PC—Block diagram.

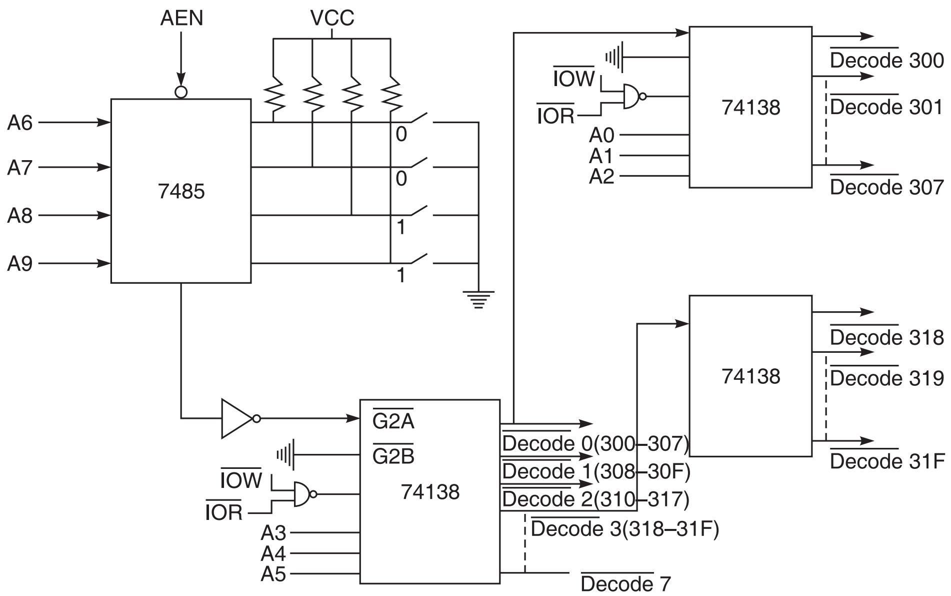


Figure 9.32 Switch selectable decode for interfacing to PC—Circuit diagram.

9.5.4 Interfacing an ADC to PC

Figure 9.33 shows the pin diagram of ADC0809 which contains 8 channel multiplexer on the chip. Following I/O signals in addition to analog input signals are required to interface the ADC to PC:

INPUT	OUTPUT
Channel address (3 bits)	Data output (8 bits)
Start signal	End of conversion
Output enable	

Number of Input and Output ports required for PC will be

Input Ports-2

Output Ports-1

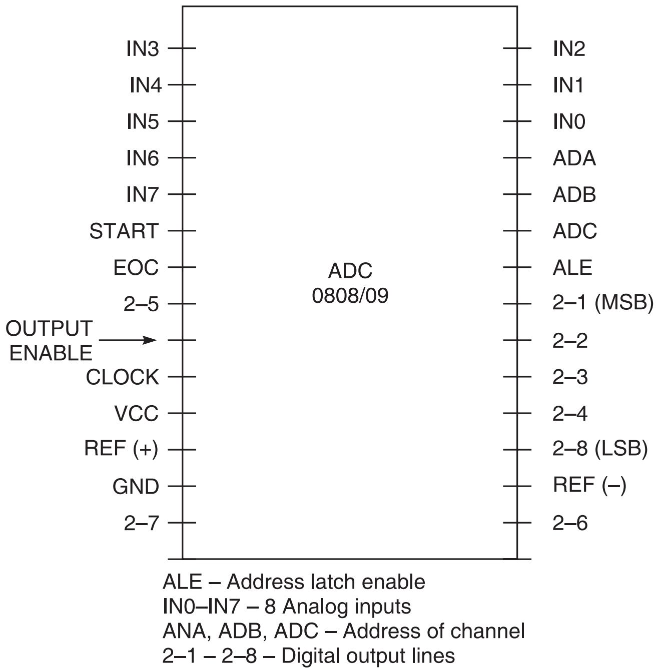


Figure 9.33 ADC 0808/09 pin diagram.

Figure 9.34 shows the schematic diagram of ADC interface to PC. The circuit diagram in Fig. 9.35 shows the interface of ADC to PC.

9.5.5 DAC interface to PC

In DAC interface, only one output port will be required. Writing data to port automatically starts the conversion in to analog and resultant analog signal is presented at the output of DAC. Figure 9.36 shows the schematic and Fig. 9.37 the circuit diagram.

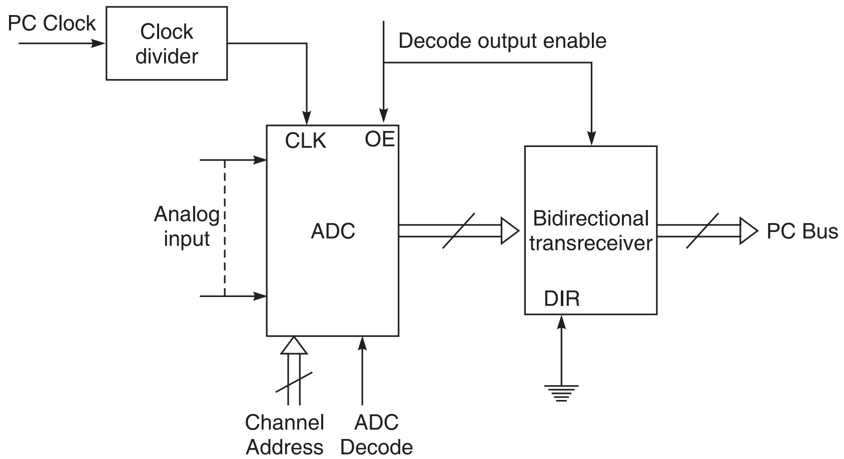


Figure 9.34 ADC interface to PC—Block diagram.

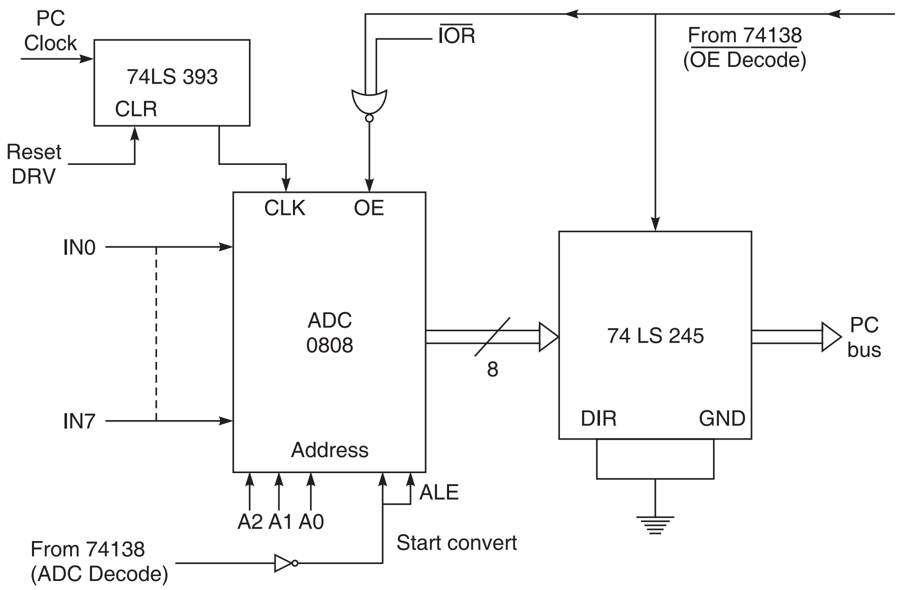


Figure 9.35 ADC interface to PC—Circuit diagram.

9.6 PERSONAL COMPUTER IN REAL-TIME ENVIRONMENT

Looking into the potential of PC to solve the real-time process control problems, a number of companies have designed hardware and software which make PC suitable in real-time applications. They have developed real-time versions of IBM PC, interface cards and real-time operating system. We shall discuss this briefly to give an idea about the capability and availability. Specialised PC buses for embedded system applications were also developed.

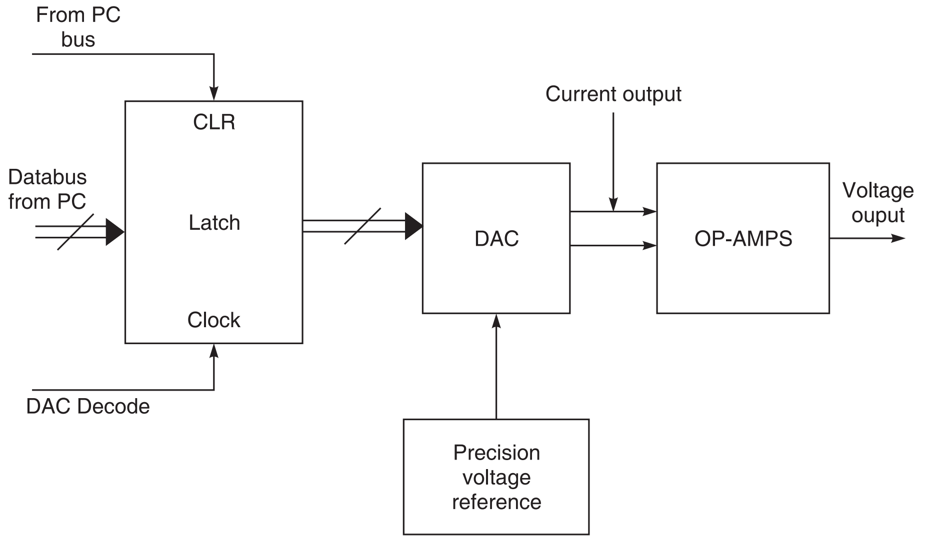


Figure 9.36 DAC interface to PC—Block diagram.

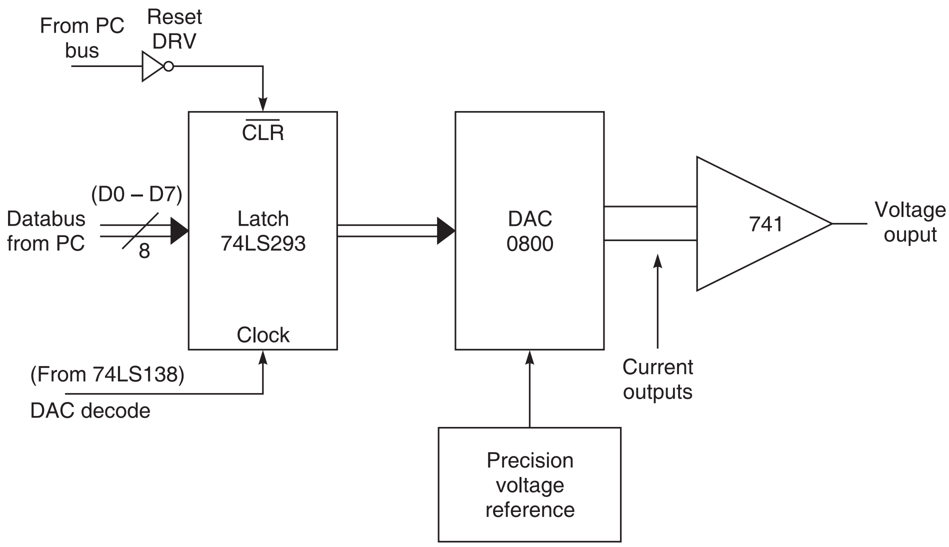


Figure 9.37 DAC interface to PC—Circuit diagram.

Depending on the type of applications, i.e. requirement of channels, data type, scan time, data acquisition and control, the personal computer can be used along with plug-in boards or external system.

9.6.1 PC Bus for Embedded Applications

Over the past decade, PC architecture has become an accepted platform for far more than desktop applications. PCs are widely used as embedded controllers within laboratory instruments, communication devices and medical equipment, to name just a few examples. An important advantage of using PC architecture is that its standardised hardware and software components are widely available. These components are also significantly more economical than traditional non-PC bus architectures such as STD, VME, and Multibus.

Thus, by standardising hardware and software around the broadly supported PC architecture, embedded system designers can substantially reduce development costs, risks, and time. This will translate into faster time-to-market, lower product cost and the ability to bit critical market windows with timely product introductions.

Need has therefore been felt for a more compact implementation of the PC bus to accommodate the reduced space and power constraints of embedded control applications. These goals have to be realised without sacrificing full hardware and software compatibility with the popular PC bus standard. This would allow the PC's hardware, software, development tools, and system design knowledge to be fully leveraged. In other words, the PC bus has to be standardised while capturing all of the benefits that the design engineer wants from the PC bus.

PC/104 was developed in response to this need. It offers full architecture hardware and software compatibility with the PC bus, but in ultra-compact (3.6" × 3.8") stackable modules. PC/104 is therefore ideally suited to the unique requirements of embedded control applications, including full standardisation by an international group of companies who manufacture products based on the PC/104 standard.

Like the original PC bus itself, PC/104 is the expression of an existing de facto standard. In 1992, the IEEE began a project to standardise a reduced form-factor implementation of the IEEE P996 (draft) specification for the PC and PC/AT buses for embedded applications. The PC/104 specification has been adopted as the "base document" for this new IEEE draft standard, called the **P996.1 Standard for Compact Embedded-PC Modules.**

The key difference between PC/104 and the regular PC bus (IEEE P996) are:

- **Compact form-factor**. The size gets reduced to 3.6 by 3.8 inches.
- **Unique self-stacking bus**. This eliminates the cost and bulk of backplanes and card cages.
- **Pin-and-socket connectors**. Rugged and reliable 64- and 40- contact male/female headers replace the standard PC's edgecard connectors.
- **Relaxed bus drive (6 mA).** This lowers power consumption to 1–2 watts per module and minimises component count.

There are two basic ways in which the PC/104 module may be used in embedded system designs:

Standalone module stacks: As mentioned before, PC/104 modules are self-stacking. In this approach, the modules are used like ultra-compact bus boards, but without backplanes or card cages. Stacked modules are spaced 0.6 inches apart. Companies using PC/104 module stacks within their products frequently create one or more of their own application-specific PC/104 modules.

Component-like applications: In another configuration, the modules function as highly integrated components plugged into custom carrier boards, which contain application-specific interfaces and logic. The modules' self-stacking bus can be useful for installing multiple modules in one location. This facilitates future product upgrades or options, and allows temporary addition of modules during system debug or test.

PC/104 is intended for specialised embedded computing environments where applications depend on reliable data acquisition despite an often-extreme environment.

Unlike the popular ATX form factor, which utilizes the PCI bus and is used for most PCs, the PC/104 form factor has no backplane, and instead allows modules to stack together like building blocks. The stacking of buses is more rugged than typical bus connections in PCs. This is a result of mounting-holes in the corner of each module, which allow the boards to be fastened to each other with standoffs.

While a typical system (also referred to as a *stack*) includes a motherboard, analog-to-digital converter, and digital I/O (data acquisition) modules, other peripherals are finding their way into the market, including GPS receivers, IEEE 802.11 controllers and USB controllers. The PC/104 computer bus (first released in 1992) utilises 104 pins. These pins include all the normal lines used in the ISA bus, with additional ground pins added to ensure bus integrity. Signal timing and voltage levels are identical to the ISA bus, with lower current requirements.

The PC/104-Plus form factor adds support to the PCI bus in addition to the ISA bus of the PC/104 standard. A PC/104-Plus module has a PC/104 connector (ISA) plus PCI-104 connector (PCI). The PCI-104 form factor includes the PCI connector, but not the ISA connector, in order to increase the available board space. The PCI-104 standard is incompatible with PC/104 boards.

System structure

A system composed of PC/104, PC/104-Plus, or PCI-104 modules is often referred to as a *stack*. Although many stacks include modules which are all of the same form factor, it is not uncommon to find PC /104 modules in a stack with PC/104-Plus modules. Each stack must contain at least one motherboard or CPU, which acts as a controller for the peripheral components. The motherboard is often referred to as a Single Board Computer (SBC), as it often has interfaces for all standard PC components (i.e., keyboard, mouse, serial ports, etc.). This controller must support the signalling buses used on all add-on modules.

9.6.2 Plug-in Boards

Applications which include the Data Acquisition and Monitoring System and may involve the control of a few control valves are simple and do not have severe time constraints. These can be managed by personal computers interfaced to ADC, DAC, and timer cards on a PC slot. These cards are controlled by a microprocessor on a personal computer, and peripherals of the PC are used for input and output. The user program in the PC will do the channel selection, initiate analog to digital conversion, perform computation, display calculated values on the VDU and initiate digital-to-analog conversion. Thus, time-critical applications cannot use plug-in boards. The interface of the personal computer to the ADC and DAC have been discussed in detail in the previous section. A number of manufacturers have started supplying a combination of these for simple applications, along with tailor-made software for DAS and control applications. Data acquisition cards that plug into the chassis of a desktop PC or a PC-compatible industrial computer have made measurement and control extremely economical for the typical lab or industrial user. Today, a PC has the same amount of computing power as yesterday's RISC workstations. Desktop and panel-mounted industrial PCs are available with the latest processors, enormous amounts of memory, fast disc drives, high-speed networking capability, and powerful operating systems.

In the past, setting up a PC-based data acquisition system was very difficult. Hardware and software were primitive, and so installation and configuration required setting jumpers and DIP switches, allocating operating system resources, changing config.sys settings, changing IRQ levels, writing software drivers, and performing hardware calibration. The current trend is toward plug-and-play systems that configure, install, and calibrate themselves. Plug-and-play software selects and assigns system resources automatically. Plug-in boards offer a great deal of software compatibility, virtually eliminating the need to write drivers or special software to interface them to a system.

A typical plug-in data acquisition board offers the following features:

Analog inputs: 8 DE or 16 SE

Resolution: 12- or 16 bit

Input range: mV and V

Sampling rate: 30,000–330,000 samples per second

Analog outputs: 2

Digital I/O: 8–16.

There are hundreds of data acquisition board configurations available. Many other boards are available with more inputs, higher resolution, and faster scanning speeds. A typical plug-in data acquisition card has a sampling rate of 30,000–250,000 samples per second. A high-speed data acquisition card, on the other hand, operates at 330,000 to 20 million samples per second. Applications for such high sampling speeds include wind tunnel testing, auto crash testing, video processing, ultrasonic imaging, and waveform analysis. At such speeds, the data acquisition card must be able to communicate with its computer via direct memory access (DMA). This allows the card to transfer large quantities of data directly to the computer's memory. In some cases, the card may have dual-channel DMA.

A high-speed data acquisition board often has a considerable amount of First-In-First-Out (FIFO) memory. Typically, a high-speed data acquisition board can store 64,000 samples. Some can store more than a million samples, either on the board or by using additional memory on daughter boards. A typical high-speed data acquisition board above 330,000 samples per second has the following features:

Analog inputs: 8 DE or 16 SE

Resolution: 12-bit

Input range: mV and V

Sampling rate: 330,000 to 1,000,000 samples per second

Analog outputs: 2

Digital I/O: 8.

Often, as speed increases, resolution drops. Of course, 14-bit and higher resolutions are available in high-speed data acquisition boards at a cost.

Bus Architecture Options

Plug-in data acquisition cards are designed primarily for different types of bus architectures. Selecting the proper bus architecture is important as a wrong choice can limit or defeat expansion plans. The following is a description of the various buses available for IBM PC compatibles.

ISA: The first Industry Standard Architecture (ISA) bus was introduced with the original 8088-based PC/XT in 1980. It can address 8- and 16-bit connectors for 16 data lines and 24 address lines. This allows the bus to address more memory. Neither of the ISA buses can, however, act as a bus master. Many plug-in boards list both PC/XT and PC/AT compatibility, and many modern computers still have one or more ISA expansion slots.

EISA (Extended ISA): Another progression of ISA, EISA added support for intelligent bus-master expansion cards. This is important because if the data acquisition board cannot take control as a bus master, it may not be able to transfer data any faster than an ordinary ISA board. With bus mastering, the data acquisition board makes the DMA transfer at the highest possible rate. EISA supports 8-, 16-, and 32-bit data transfers at speeds up to 33 Mbps.

PCI: The Peripheral Component Interconnect (PCI) bus is a processor-independent bus that supports up to 64-bit addressing, bus mastering, and burst transfer rates up to 1,056 Mbps (132 Mbytes per second). The standard is still evolving, with developments such as the Compact PCI standard for industrial applications, and hybrid versions that work with VME, PMC, Multibus, and STD. Virtually every Pentium PC being shipped today has a combination of ISA/PCI or EISA/PCI buses.

PC/104: Intended for embedded computer applications, PC/104 cards are very compact at 3.6 in. × 3.8 in. PC/104 also allows card modules to self-stack without backplanes or card cages, and has pin-and-socket bus connectors designed for high reliability in harsh environments. It is essentially a miniaturised version of the standard PC bus architecture. Also in continual development, the latest generation (PC/104-Plus) operates at the same speed as PCI.

USB: The Universal Serial Bus (USB) is currently being offered in most commercial PCs, it is essentially an upgrade of the venerable RS-232 port. It allows up to 127 USB devices to connect to a PC. USB peripherals are automatically detected and configured when connected and could include virtually every peripheral device now being used, including keyboards, monitors, touchscreens, modems, and data acquisition devices. It operates at 12 Mbps.

A number of companies all over the world are manufacturing and offering plug-in boards for PCs for industrial control applications. These include industrial control boards for ISA, EISA, PCI, and PC/104 buses. These boards have analog inputs (differential and/or single-ended), timers/counters, digital I/O, and I/O connectors. In addition to these, hardware plug-in boards and powerful Window/Linux-based software modules for end-user application development are also offered by manufacturers of these boards. A number of plug-in boards for different specifications and applications are being offered.

Two technology advancements are making the use of low-cost plug-in boards for time-critical multifunction I/O possible. First, the PCI bus, now ubiquitous on today's PCs, has the bandwidth necessary to acquire data from multiple sources on a single low-cost plug-in board as well as to generate multiple channels of output to the same plug-in boards. The ISA bus, which has been replaced by PCI, simply couldn't deliver this bandwidth.

The second advancement is the way in which multifunction A/D boards are designed. In the past, standard digital I/O devices, such as the 8255 chip, were used for digital I/O on nearly all boards. Also, the 8253/8254 chip was used to provide the counter/timer capability on most boards. Although these devices made it easy for board designers and manufacturers, their architecture made them incapable of being time-synchronised with one another or the analog input.

Today's new board designs are incorporating all of the digital I/O and counter/timer capability on more powerful and customisable ASICs (Application-Specific Integrated Circuits) or high-density FPGAs (Field Programmable Gate Arrays), making it possible to time-correlate their I/O capability with the board's analog input and output capability.

9.6.3 External System

The external systems are basically external enclosures having microprocessor based systems with memory and connection to the outside world through MUX, ADC, DAC, TIMER etc. These are connected to PC with serial link, if these boxes are away from PC otherwise parallel link when PC is in close vicinity of these boxes. The parallel link may be IEEE 488 GPIB system bus or direct connection through DMA lines. The serial bus may be RS 232C, RS 422 etc. In case of external systems, the tasks are shared between PC and the boxes. As an example, the boxes may collect the data from different parts of plant, collate and then send the data on serial or parallel link to PC which will generate alarm, display the plant status and execute control algorithm to determine the control valve setting. The value of control valve position may be transferred to appropriate box for sending the signal to control valve (Fig 9.38). One of such systems is Schlümberger 65xxx series DAS modules, which uses a PC resident card and external DAS modules for various combination of input transducers and control functions. A dedicated software running in PC facilitates buffering, storing, formatting and displaying received informations.

9.7 INDUSTRIAL PERSONAL COMPUTER DEVELOPMENT

The personal computer (PC) has experienced an unprecedented success story and has become a firmly established part of everyday life, including industrial environments. Together with

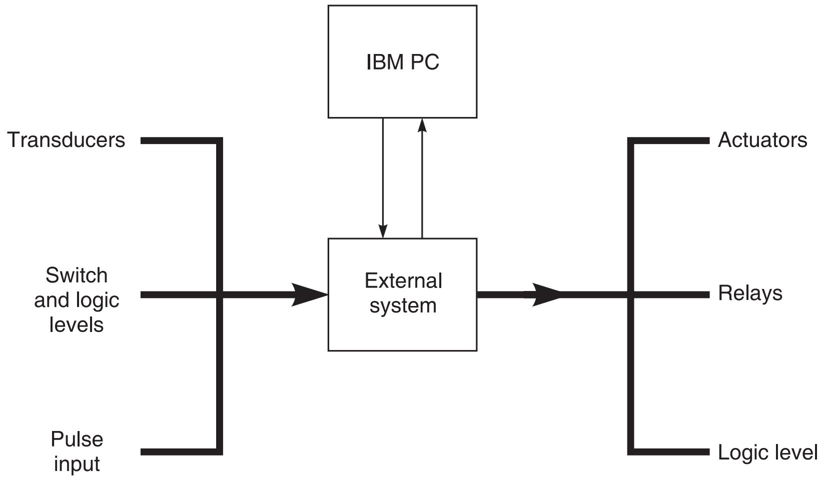


Figure 9.38 External system interface.

associated software, PCs in different shapes and forms are available for a wide range of diverse automation tasks such as control of machines, processes or logistics systems, networking of system components, data acquisition, and image processing. For classic control tasks, PC-based control technology offers excellent scalability and flexibility and is therefore increasingly being used in place of hardware PLCs.

Industrial PCs offer features different from consumer PCs in terms of reliability, compatibility and expansion options. Industrial PCs are typically characterised by being manufactured in lower volumes than home or office PCs. Construction is typically of thicker guage material, either cold-rolled steel or aluminium, with added industrial features like high-performance fans, shock mounting on the drives, air filters, lockable doors, etc. A common category of industrial PC is the 19" rackmount form factor. Industrial PCs typically cost considerably more than comparable consumer PCs with similar performance. Single-board computers and backplanes are used primarily in industrial PC systems.

9.7.1 Benchtop, Rackmount, Wallmount and Panelmount Industrial PCs

An industrial PC is a rugged PC that can be either a benchtop, rackmount, wallmount or panelmount system, depending on the application. Industrial PCs offer a higher temperature operating range than normal PCs along with shock-mounted drives, forced air cooling (if required), card clamps to protect against vibration and lockable controls.

Having decided on which system would best suit the application, the user can decide what size of chassis to select.

The benchtop and rackmount industrial computers provide highly reliable solutions adapted for very demanding needs and applications. These rugged PCs are designed to perform in hostile environments with vibration, shocks and high temperature while still providing the latest and fastest system speed required. The enclosures of all rackmount and benchtop products include a filtered positive air-cooling system as well as protection against vibration and shock on the internal hard and floppy drives.

Rackmount chassis are normally available in 2U, 4U or 6U format, and the choice is generally dictated by the number of slots (ISA/PCI) required and the media storage needed.

Wallmount chassis are offered in various sizes depending on the number of slots (ISA/PCI) needed for the application and whether a full-size or half-size CPU board is to be used. A range of different backplanes offer the choice of bus to be used, and various combinations of slots for plugging in add-on I/O boards allows users to build systems to their requirement.

Backplanes should be selected in conjunction with the slot CPU that best supports the application requirements. Particular consideration should be paid to processor speed and I/O functionality.

The panelmount industrial computers are designed to provide a rugged compact enclosure with great flexibility regarding speed processing and bus technology needed for any industrial application. The enclosure of all panelmount products is protected against vibration and shock on its internal drive bay and on its front-accessible 3.5" floppy drive. Each computer is provided with brackets for convenient wall or panel mounting.

9.7.2 Environmental Specifications

All industrial-grade PCs normally must be able to work in the shop floor environment in a factory. They must conform to the following general environmental specifications:

Operating

- **Temperature:** 0 to 45° C (32 to 113°F)/Up to 50° C (122°F) optional
- **Relative humidity:** 8 to 85% non-condensing
- **Shock:** 24 G, 2 ms
- **Vibration:** 100 to 500 Hz: 0.3 G.

Non-operating

- **Temperature:** – 25 to 65°C (–13 to 149°F)
- **Relative humidity:** 5 to 95% non-condensing
- **Shock:** 200 G, 2ms
- **Vibration:** 100 to 500 Hz: 4 G.

9.7.3 Entry Level Industrial PC

These are low-cost industrial PCs required for medium performance range in terms of computing as well as SCADA requirements. The general features include:

Processor. Intel Celeron or Intel Pentium III; up to 1.26 GHz

Memory. 128 MB standard upgradeable to 256 MB

Hard Disk Drive. Ultra ATA-100, 7200 rpm, 8.5 ms: 2 GB (standard) to 6 GB

Video Card. ATI range 128/8 MB/16 MB/32 MB

Ports. –2 serial –1 parallel with ECP–PS/2 and AK keyboard with PS/2 mouse on System Board model or AT keyboard–PS/2 mouse on Passive Backplane model

Operating System. –Microsoft Windows 98 NE –Microsoft Windows NT Workstation 4.0 –Microsoft Windows 2000 –Real-Time Linux.

9.7.4 Mid-Range Industrial PCs

Reliable operation with increased performance and security the are main characteristics of mid-range industrial PCs. In this range of industrial PCs, powerful processors (Intel Pentium IV or equivalent) are often used. They also contain hard disc drives through the RAID controller, and network interface through the integrated fast Ethernet. The general features include:

Processor. Intel Pentium IV processor supporting Hyper-Threading Technology up to 3.2 GHz processor speed

Memory. 256 MB standard upgradeable to 4 GB DDR SDRAM memory

Hard Disk Drive. ULTRA 160 SCSI, 10000 rpm, 5.2 ms: 18.4 GB to 73.9 GB

Network. Integrated Fast Ethernet 10 Base T/100 Base TX

Video Card. ATI/32 MB/64 MB/128 MB

Ports. –2 serial – 1 parallel with ECP–PS/2 keyboard with PS/2 mouse on System Board model or AT keyboard and –PS/2 mouse on Passive Backplane model –2 Universal Serial Bus.

Operating System. –Microsoft Windows 2000 –Microsoft Windows XP –Real Time Linux

9.7.5 High-Performance Industrial PCs

These industrial PCs are meant for complex software applications. They normally have dual processor technology, graphics technology, high-performance RAID controller, network interface, etc. The general features include:

Processor. Dual Intel Pentium III up to 1.26 GHz or Dual Intel Pentium IV supporting Hyper-threading Technology up to 3.2 GHz processor speed.

Memory. – 256 MB standard upgradable to 768 MB –Configurations including ECC memory

Hard Disk Drive. ULTRA 160 SCSI, 10000 rpm, 5.2 ms: 18.4 GB to 73.9 GB

Network. Intel 82559 Fast Ethernet

Video Card. ATI Radeon/32 MB

Ports. –2 serial –1 parallel with ECP –PS/2 and AT keyboard with PS/2 mouse on System Board model or AT keyboard and PS/2 mouse on Passive Backplane model –2 Universal Serial Bus

Special features. –Cards hold-down bracket –Automatic restart

Operating System. – Microsoft Windows NT Workstation 4.0 –Microsoft Windows 2000 – Microsoft Windows XP – Real Time Linux.

9.7.6 New Technological Features

The new technological features being developed include touch screen display, built-in security support and optional wireless interface.

For security, biometrics was developed and offered as an optional feature. This will be offered as an integrated feature in the near future. For security-heavy applications, press type fingerprint, finger vein, palm vein and iris recognition technology are under development. They will also be used for entry security and windows log in/out.

Another step towards more advanced products is the integration of wireless LAN and PAN, and even mobile wireless technology as an option.

Many industrial PCs are offered with Bluetooth to connect barcode scanner, printer, keyboard, mouse and speakers. Industrial PCs for automotive applications use GPRS, 3G and GPS as technology options.

9.7.7 Real-time Interface

Like any other PC, industrial personal computers can be interfaced to outside world through I/O ports This concept has already been discussed in the previous sections. Figure 9.39 shows the connection of industrial PC with device through I/O port. The device in the figure may be any controller, input module of data logger or any test and measuring device. This type of interface assumes the intelligence to be associated with PC only. However, if we want to off-load PC from routine data logging job then we have lo build up intelligence at the level of interface between PC and I/O device.

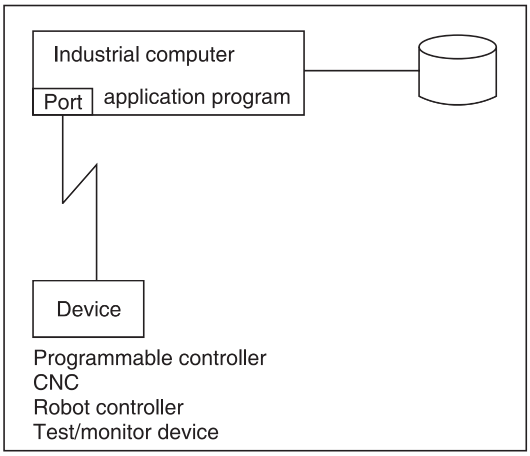


Figure 9.39 Industrial personal computer interface to outside world.

9.7.8 Application Specific Software

Considering the potential of IBM PC in solving real-time problems, number of companies have developed software packages oriented towards different application scenario. We shall describe in brief, these software packages and applications in which these can be used.

Integrated control

The FIX (Fully Integrated Control System Software) developed by M/S Intellution Inc., and Loopworks developed by M/s Equinox Data Corporation are widely used in close loop control applications.

The FIX is completely interactively configurable software and requires no programming. The main functions provided are Data Logging, Control, Scan and Alarm, Database and Colour Graphic Displays. Database and graphic displays can be configured on-line without disturbing the process. Other capabilities of FIX are historical trending, automatic data sampling, compression and storage, graphics data playback etc. The graphic editor in FIX enables to create special picture elements and animate interactive operator displays. The control strategy may be configured as PID, Feed Forward, On/Off, Supervisory Control or Batch Sequencing. (A detail description of FIX is given under Section 9.9—PC based Distributed Control System.)

The Loopworks is an advanced process monitoring and control software for the IBM PC. The software supports the OPTOMUX (M/s Opto 22) line of I/O hardware. It allows the engineer to completely configure a multiloop control interactively without writing any program. The main functions provided by software are Process Monitoring, Data Logging, PID Control, Discrete Control, Compusensors, Logic Operations, Operator Displays, etc. Compusensors include calculations for sensor data like linearisation, flow/temperature correction, feed forward calculations etc. Logic files are generated in Lotus 1-2-3 campatible formats. Under PID Control, Hierarchical/Cascade control are also possible. Logic operations include evaluation of digital inputs, alarm conditions, analog inputs etc.

Programmable controller support

Following programmable controller support softwares have been designed for IBM PC:

— CAMM (Computer Aided Manufacturing Management) for Programmable Controllers
— Taylor Industrial Software
— PC TABLES and TAXI

CAMM (developed by CENTEC Corporation) provides monitoring of real-time activities of process. It collects and stores status of control points (sensor input values, control output levels, set-point values etc.) in real-time. It produces engineering analysis and production reports. IBM PC can be interfaced with any of the following programmable controllers:

— Allen Bradley PLC-2 family
— Analogic AWDS 4400 series
— General Electric Series 6

— Gould Modicon
— Texas instruments TTWAY.

Taylor Industrial software is a high quality ladder logic documentation system for the users of Allen Bradley, Modicon or Texas instrument Programmable Controllers. An I/O wiring generator is also available for Allen Bradley and Modicon Programmable controllers. It interrogates the documentation configuration file and produces I/O wiring diagram of swing arms and termination points.

Developed by Soft Systems Engineering Inc., PC TABLES and TAXI are basically programmable controller communications and data collection software for IBM PC. PC TABLES and TAXI provide software drivers to link IBM PC to the following programmable controllers:

— Allen Bradley PLC-2 family
— Gould Modicon
— Texas Instruments 520, 530
— Texas Instruments PM 550.

These drivers are easy to use and callable from BASIC. The drivers perform communication formatting and control. PC TABLES enable the user to display data points, textual descriptions, and current status in a multiple column table format. TAXI allows the movement of BCD data words from Allen Bradleys' PLC-2 to Lotus 1-2-3 or VISICALC spread sheet. Thus, apart from providing software drivers for PLC's, PC TABLES and TAXI allow you to collect historical data for trending.

Numerical control

After blueprint design of any part is completed the part programming does the translation from blueprint to Numerically Controlled (NC) machine code. Developed by CAM shack, Inc., Minuteman is a profile description Program. It understands the blueprint oriented geometry like points, circles, bends, lines, angles, distances etc.

The Minuteman is compatible with any CNC controller. It calculates the tool path based upon the internal diameter and has virtually unlimited two-half axis math capabilities. The programme provides a rectangular—polar conversion, triangle solver, unknown point on a line, unknown circles solver etc. The features of part programming like rotate, reverse, repeat, scale, mirror, copy offset are available. Other features include tool hold off at punch down, Z axis routines, graphics, teleprinter emulation etc.

In order to ensure that tool crib is not out of stock, Applied Control Technolites Inc. has developed software package CRIBWARE The software enables the user to have tool usage and cost information, allocation of tooling to specific operations, tool check out from CRIB and prevention of stock out by generating recorder report. The user may interactively create and modify—tool, job and operation; classify tools as returnable or non returnable, maintain records of tool usages; monitor tool check out; control tool purchases and generate management reports.

Process graphics

The 'screenware' software package developed by Computer Technology Corporation has comprehensive capability of graphic design and graphic animation. Colour graphic displays can be designed and animated to depict a process or machine. Any graphic shape like line, box, circle, ellipse, arc can be created. The package has features like fill, paints, hatch, copy, rotate, zoom, pan etc. with 16 colour pixel based graphics. Animation is accomplished through a library of 27 animation function commands.

Management information system

'Job Boss' is a software package which can be used on IBM PC to monitor the progress of different orders, their delivery schedules, job cost, job status, shop load, etc. It also collates the company's financial data in five modules—job control with inventory, accounts receivable, accounts payable, payroll and general ledger. The software enables you to enter routings, set up times, run times and use this information for all repeat orders. The system can calculate and recalculate shop schedule for changes, can assign multiple choices of delivery schedules to a customer in order to meet the requirement. It also prepares status report, backlog report and queue report to enable the user to know the exact status of job movement.

Maintenance management system

Following software packages in the application range are available:

— Maintenance Management Software
— MICROMAINT
— FLEETMAINT

Developed by ACME Visible Records, Maintenance Management Software is basically preventive maintenance and inventory system. The basic components of preventive maintenance are master equipment records, scheduling system for periodic maintenance due on specific equipment and log of work performed with costs and type of work for periodic maintenance when due. It can also print work orders for breakdown emergency. The logging of equipment down time, material and labour cost for repairs and routine maintenance is done automatically.

The inventory system automatically logs parts usage, budgeting information and number of breakdown of a part. The system also prints parts recorder report, which helps in maintaining minimum stock of every component.

The system, in addition, furnishes equipment status report, depicting average days each type of procedure like preventing maintenance, inspection etc. is due. This is past due report.

The MICROMAINT software package has been developed by M/s Diagonal Data Corp. It is a preventive maintenance and inventory control system. Following are the main features of the system.

— Printing of detailed maintenance management report
— Inventory system for spare parts

— Forecast of planned maintenance in future
— Scheduling and printing of preventive maintenance work order.

This software package can work in conjunction with CAMM software described earlier. This allows data from variety of external devices to be collected in real-time. The data on parameters like temperature of critical part (e.g. motors), total time equipment has been running, can be used by the system to initiate maintenance procedures.

FLEETMAINT is basically maintenance and inventory software system for a fleet of vehicles. The capabilities of system are same as MICROMATNT except that this system has been specially designed for maintenance of fleet of vehicles.

Inventory control

For comprehensive inventory control in a manufacturing plant, Computerware has developed a software package called 'Manufacturer's Inventory System', for IBM PC.

The basic facilities offered by the system include interactive invoicing, automatic inventory adjustment as and when the items are ordered, received and transferred to work in process or assigned to finished goods. The system can prepare bill of material for any assembly by taking into account current cost of component and no. of times used in assembly from database and by interactively taking labour and overhead cost, the cost of any finished good can be prepared and printed. The system helps the user to take management decisions.

Quality control

For attaining required quality level, a number of statistical calculations and operations are performed. Following software packages have been developed for this purpose:

— Quality Alert
— SQCpack

Developed by Anton Software Inc., Quality Alert is a statistical quality control software system which enables the user to perform Statistical Quality Control (SQC) calculations. The functions of 'SQC' available in system are

— Histogram and descriptive statistics for variable and attribute data
— Control charts for variable and attribute data
— Process capability for variable data
— Pareto analysis.

The SQCpack software system designed by Productivity Quality Associates Inc, is basically Advanced Statistical Process Control Software. The system offers the following facilities:

— Variable charts: X-Bar, Median, Individual or Moving average, Range Sigma, Moving Range.
— Frequency Distribution/Histogram
— Statistics including mean, standard deviation, test for normalcy, kurtosis, and skewness

— Capability analysis: including process capability index (Cp, Cr, Cpk), theoretical per cent out of spec, actual per cent out of spec., Z statistics (spec lines displayed on chart).
— Attribute Charts: viz. P, NP, C, U
— Problem-solving tools: like Pareto (including cost factors) and cost effect diagrams.
— Graphic representation of data.

The software supports automatic data collection through a device. The SQCpack can also be interfaced to Lotus 1-2-3 and dBASE III files.

Technical computing environment

MATLAB is a technical computing environment for high performance computation and visualization. Known as *matrix laboratory*, it integrates numerical analysis, matrix computation, signal processing and graphics in an interacting environment. It features a family of application specific solutions called toolboxes. These toolboxes help user to extend the functionality of MATLAB. Following are the main toolboxes:

— Signal processing toolbox
— Control system toolbox
— System identification toolbox
— Optimisation toolbox
— Neural network toolbox
— Spline toolbox
— Robust control toolbox
— μ-analysis and synthesis toolbox.

Mathworks who are developers of latest MATLAB have also developed a system for simulating dynamic systems. Known as SIMULINK, it is graphical environment to facilitate user to model system by drawing block diagram on VDU screen and manipulating them interactively. SIMULINK can handle linear, nonlinear, continuous time, discrete time, multivariable and multirate system.

Panel replacement and control

PowerVIEW software package designed by Nematron corporation is window-based development tool, which can be used to create panel replacement and control applications for any PC. It offers 'Fill in the blanks' set-up tools to create custom panel replacement. PowerVIEW's open PC architecture fully implements TCP/IP and DDE for wide open communications, through all levels of automation strategy.

Data acquisition and control

LabVIEW 4.0 software package developed by National Instruments is a powerful and flexible tool for developing industrial data acquisition and control systems. It can be used for a wide range of applications from supervisory and direct PID process control to complex data analysis and process simulation. LabVIEW is available on Windows plateform and offers wide range of instrumentation I/O options with drives included for serial instruments, GPIB, VXI hardware, PLCs, Motion controllers etc. It also includes connectivity tools like TCP/IP, DDE, net DDE etc. for company wide access to process data.

BridgeVIEW developed by National Instruments as advancement over LabVIEW, also runs on Windows plateform. It is designed to monitor and control one or more distributed units in client server architecture. The user can monitor the process, change set-point and send control instructions to individual data acquisition devices which may be PLC, plug in I/O boards or other distributed I/O systems. BridgeVIEW system contains three processes: BridgeVIEW engine, the user MMI applications and industrial automation device servers.

The LabVIEW and BridgeVIEW softwares are built around the graphical programming language "G". Developed by National Instruments, G facilitates programming in block diagram form, using icons and graphical symbols rather than text to describe programming actions. Programs are called Virtual Instruments (VIs) since they initiate in software the appearances and functions of actual instruments. VIs contain an interactive user interface with push buttons controls etc. BridgeVIEW includes libraries of functions and subroutines as well as ISA symbol and OLE (object linking and embedding) automation libraries. A new feature of BridgeVIEW is the G—Wizard which allows developers to reuse common pieces of VI diagrams.

In addition to the software mentioned here, a large number of packages have been designed on the similar lines for the IBM personal computer user. This has made PCs a very powerful tool in manufacturing control as well as process control applications. We shall discuss a few applications in Section 9.8.

9.8 REAL-TIME APPLICATIONS OF IBM PC

9.8.1 Servo-Control

We have discussed the control of motors in Chapter 4. The basic control of a motor can be done through PC by using the same circuit. However, servo-control of motor requires a PID type of algorithm to be executed. The motor speed and position may be precisely controlled using servo control.

National Semiconductor Corp., California USA, has introduced two digital motion controller ICs namely LM628 and LM629. Add-on servo motor controller boards for personal computer are available from Technology 80, Inc. These two ICs are identical except the output stage. The LM629 has an 8-bit pulse-width modulated output whereas LM628 controller has parallel 8-bit or multiplexed 12-bit output for driving a DAC to produce an analog motor-command signal.

The ICs communicate with the host computer using an 8-bit parallel interface. Users are provided with 22 high level programming commands. The position command data is loaded as 32-bit value, and accommodates more than $+/-10^9$ encoder counts. Velocity command data and acceleration are also 32-bit values and have a high resolution of $1/2^{16}$. Velocity ranges from 0-16,383 counts/sample interval and acceleration ranges from 0-16,383 counts/sample. The PID coefficients are loaded as 16-bit values, with a range of 0-32,767.

The coefficients and varied sampling frequency for the derivative term enable the user to control a wide range of motor sizes and mechanical system time constants, by using personal computer with servo motor controller board.

The PID coefficients, position, and velocity may be updated on the fly (during motion). This ability to make dynamic changes is useful in performing a variety of complex motions and in accommodating load variations.

The models 5628 and 5629 servomotor controller boards from Technology 80 are PC bus-compatible implementations of the LM628 and LM629 ICs. Each model controls one DC servomotor and offers users a means to develop and apply microcomputer servo control.

9.8.2 Manufacturing Control

Figure 9.40 shows the plant floor connectivity strategy developed by IBM. The manufacturing control computer (called cell control computer by IBM) provides the following functions on plant floor:

— Production monitoring
— Production control
— Host communication
— Communication with other controllers
— Operator Interface.

9.9 PC BASED DISTRIBUTED CONTROL SYSTEMS

By using PCs for data acquisition and control, several functions which were not possible using conventional system can now be realised. The personal computers by means of its speed of computation and ability to store and retrieve data can easily achieve these functions. The following capabilities are incorporated in the PC based distributed control systems:

— Fast real-time data logging capabilities
— Effective communications between the computer and the I/Os for scanning and outputting data
— Software alarm annunciation
— Events recording
— Dynamic mimic displays
— Procedural start-up and shut-down algorithms
— Library of algorithms to perform virtually any control and monitoring scheme
— Advanced diagnostic features to detect faults in the instrumentation system
— Performance and efficiency calculations
— Real-rime and historic trending facility

By implementing these features, the process efficiency is improved. The PCs also offer the ability to apply advanced control strategies discussed and also allow easy expansion of the control capabilities. Improved reliability and availability of the control system through cost-effective redundancy and back-up is achieved.

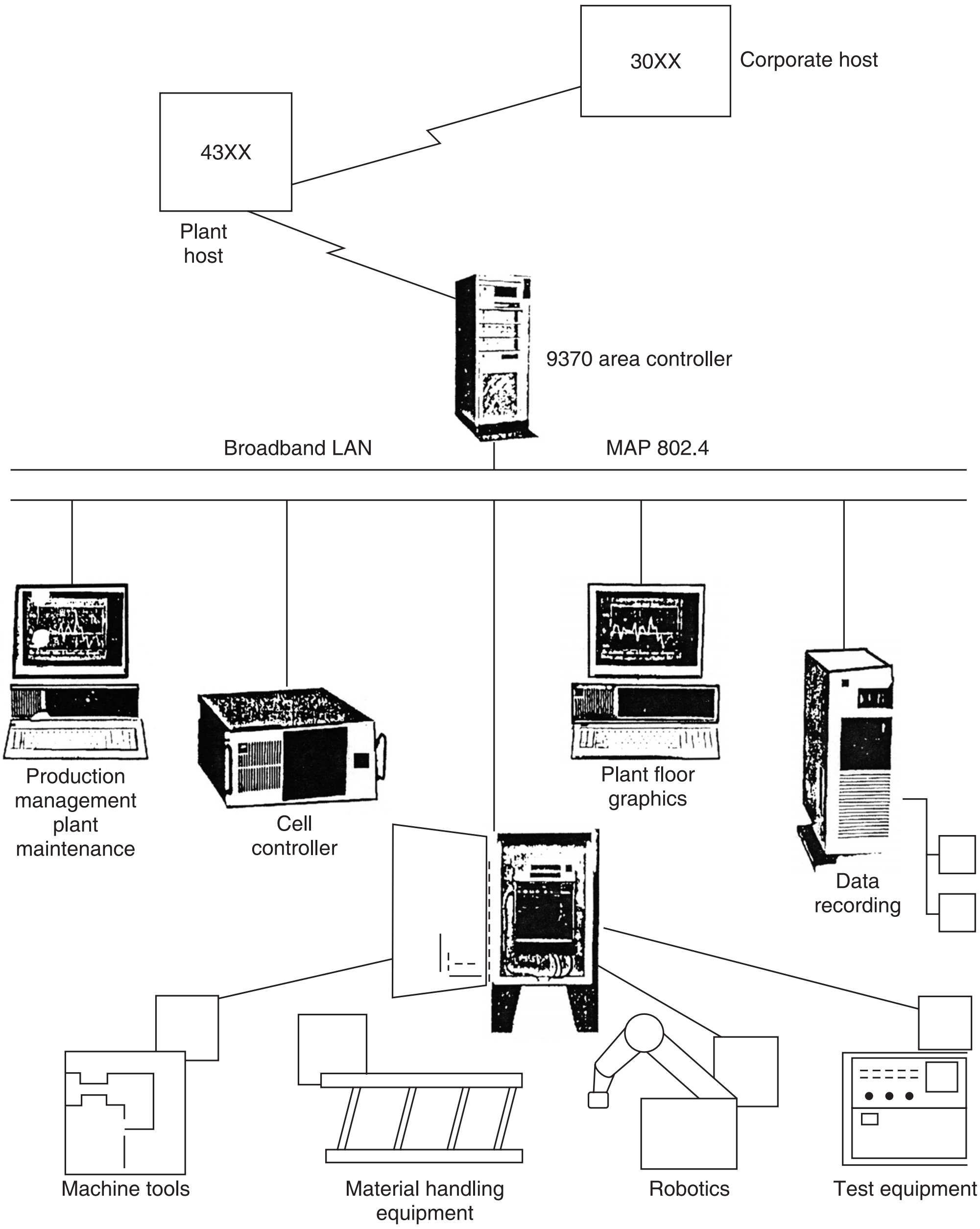


Figure 9.40 Plant floor connectivity strategy [*Source*—IBM manual].

The Distributed Control System (DCS) may be configured around an Pentium PC compatible computers functioning as the Operator's Console (OC) and Remote Terminal Units (RTUs). The system configuration, loop configurations and plant monitoring is done at the OC while the plant control is performed by the RTUs.

9.9.1 Operator's Console

The Operator's Console (OC) can be configured around the Pentium PC compatible computer. PC with a 16/32-bit microprocessor, 4/8 MB RAM, EGA/VGA graphics display and mass storage facilities are ideally suited to provide advanced graphics, mimics and associated displays. The operator must be able to configure the DCS through Operator's Console (OC).

Once the DCS is controlling the process, the OC acts as a window to the process, through which the operator can monitor current status, modify parameters, for manual operation. The operator can also log the status of any variable at programmed intervals for further analysis. The various screens like trending, mimic displays and bar graphs should be provided.

For display of process status, the operator should have access to a presentation of displays in a hierarchy of following four levels: Plant (overview), Area, Group and Loop. These display screens have been discussed in detail in Chapter 7.

Historical information for Process variable data should be available to the operator for trending, trouble shooting and optimising.

9.9.2 Remote Control Unit

As stated before, a remote control unit can be configured around a personal computer with plug-in board or personal computer with external systems. The configuration is dictated by the response time of the process.

Control algorithms are executed in the RTU, utilising a set of connectable, reconfigurable groups of control information referred to as control blocks and stored in appropriate look up tables. Control blocks are actually groups of user specified parameters. Designers of RTU, thus have to develop these generalised control algorithms as building blocks.

Control parameters are used to detail the operating characteristics of a particular control algorithm selected by the user. These indicate which inputs and outputs are to be used in executing a selected control algorithm.

A number of control block algorithms are resident in the RTU. These algorithms represent a single control strategy or may be distributed among a number of control strategies. Each control strategy controls a single process loop. Utilising different control strategies, RTUs can control a number of loops.

9.9.3 Communications

It is critical portion of DCS operation, and should be highly reliable. The data should be put in a queue, and should automatically be taken up for transmission as soon as the channel becomes available. The messages are repeatedly sent till all the acknowledgments are received. The data integrity should be maintained by a systematic acknowledgment protocol which identifies byte parity errors, transmission errors and frame errors. Also, the watch dog timers identify cable faults. In case of an error the entire frame is automatically retransmitted and the receiver discards the erroneous message.

The diagnostic and status information features should be provided to collect the information like number of retransmissions/frame, channel statistics such as total reception errors and transmission errors etc.

The PC based DCS provides a low cost solution to the process control needs wherein the major portion of the investments will be for the transducers and the final control elements, which anyway are required for any type of DCS. By providing distributed control at the RTU level, and monitoring and manual control at the OC, a true hierarchical distributed control system is realised.

Let us take an example: Intellution's FIX DMACS system provides a distributed system environment that exhibits key DCS features, while taking advantage of major network capabilities and standards. Although FIX DMACS is part of Intellution's FIX family of single station automation software for the IBM PC, the DMACS system represents a complete distributed system architecture that has been designed specifically for distributed processing while accommodating multiple classes of users within the system.

Colour graphic CRTs are used by one or more operators for observing process variables, process status, and alarm messages, as well as for entering operational commands. Still other computers within the system are frequently used to provide displays and reports to supervisors and managers.

A common practice is lo use third party personal computer software (like Lotus 1-2-3 and Oracle) to organise and analyse the process data. On-line communications with a plant host computer is also found to be a frequent requirement of end users.

The DMACS system architecture employed in the application consists of a number of computers (nodes), each connected in a peer fashion via an office-style communication network (in this case, IBM's broadband PC Network). The software that is loaded into each computer determines the function performed by that specific machine. These specific functions are listed here:

— "SCADA" nodes are responsible for data acquisition and control.
— "Display" nodes support colour graphic displays and other functions (such as alarm notification).
— "Engineering" nodes perform configuration of the databases for other machines.
— "Analysis" nodes perform all of the data concentration and host interfacing.

Exclusive of hardware restrictions, each computer can perform one or more of the above functions.

A disk-based file server, which is common in office networks, is not required in the industrial configuration, due to the delays and potential risks of failure that it introduces. However, a file server can be added to the network as a common data repository (in fact, other machines, not running the DMACS software, can coexist on the network).

The DMACS system runs in IBM's standard Windows OS environment, providing the user the straight-forward commands and wide variety of software. However, multi-tasking is a requirement, and a multi-tasking "shell" is installed on top of operating system once the system is started.

This shell provides prioritised multi-tasking, so that the I/O communications can be performed before data processing. In addition, the shell is preemptive: when a periodic clock interrupt occurs, the shell uses its dispatching algorithm to run the highest priority task that is waiting, and if need be, suspend execution of the current task. Overlapped disk I/O support, a function typically found only in larger operating systems like UNIX, allows DMACS to continue computer-bound tasks (like CRT display updates) while disk I/O (data archiving) is taking place.

With the user level database access subroutines of DMACS, the system engineer or programmer can add many special programs that also operate on line. These special programs can include such calculations as pH profiling, energy usage optimization, and host computer communications.

Most I/O devices, such as programmable controllers, interface to the computer in one or two ways: either through a serial port (such as the RS-232C) or through an interface card that installs directly into the IBM PC bus. Most systems require simultaneous support for more than one type of I/O. Therefore, multiple I/O drivers can be executed either by the control node processor, or by a separate parallel processor.

9.10 CONCLUSIONS

There are numerous applications of personal computers in different environments. We have discussed in this chapter, the potential of personal computers in addressing the real-time problems. With new developments, it is expected that personal computers will gain more power and soon many applications out of its domain presently will fall within its purview.

SUGGESTED READING

Batchelor, M., Personal computer distribution control, *C&I*, No. 5, 113, 1987.

Eggebrecht, L.C., *Interfacing to the IBM PC,* Haward W., Sama and Company, Indianapolis, 1983.

Gloss Brenner, A., *The Complete Handbook of Personal Computer Communications,* St. Martin's Press, New York, 1983.

Govindrajalu, B., *IBM PC and Clones Hardware, Troubleshooting and Maintenance,* Tata McGraw-Hill, New Delhi, 1994.

IBM PC Options and Adapters Technical References, IBM Corporation, Baca Raton, Florida, 1984.

IBM Web Page on internet (www.ibm.com).

Kompass, E.J., Reviewing PC based software, *Control Engineering,* **35,** No. 11, 1988.

Rubin S., Distributed processing using IBM PCs, *Control Engineering,* **35,** No. 10, 1988.

Sargent, Murray, Shoemaker, Richard L., *The IBM PC from the Inside Out,* Addison-Wesley, Reading, Massachussetts, 1986.

CHAPTER

10

Programmable Controllers

10.1 INTRODUCTION

A Programmable Controller (PC) is a device which performs discrete or continuous logic in process plant or factory environment. It was developed originally to replace the relays and thus early devices were capable of only sequential ON/OFF control. These were called Programmable Logic Controller (PLC) a name which is still popular.

Programmable Controller (PC) or Programmable Logic Controller (PLC) was first developed for General Motors Corporation in 1968 to eliminate costly scrapping of assembly-line relays during model changeovers. By 1971, programmable controllers were being used in applications outside automotive industry.

Early programmable controllers were conceived as just replacement for the relays. As they were capable of ON/OFF control only, their application was limited to machine and processes that required interlocking and sequencing application, such as transfer line and grinding and boring machines. Major benefits provided by these early machines were: (a) their role as diagnostic indicators to aid trouble shooting; (b) saving of installation space; (c) easily installable and reusable if projects were scrapped.

PLC's offered the following advantages in comparison to electromechanical relays.

— Economy
— Small physical size
— Suitable modular design
— High reliability
— Ease of programming
— Rugged construction
— Ability to communicate with computer

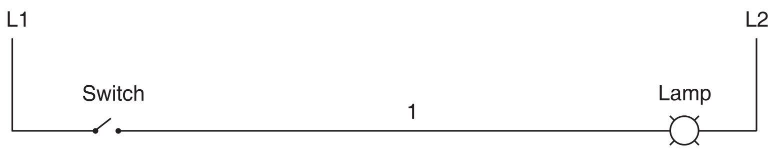
Innovations of microprocessor technology in early 1970's helped in adding greater flexibility and intelligence to the programmable controllers. Capabilities such as intelligent operator interface, arithmetic data manipulations and computer communications added new dimensions to programmable controller applications. Late 1970's saw addition of greater flexibility to programmable controller through hardware and software enhancements. Hardware enhancements included larger memory capacity, larger number of inputs/outputs handling, analog input/output and high speed data communication between programmable controllers. Through early 1980's and till today, there has been continuous upgradation of programmable controller technology. This technological upgradation reflects remarkable achievements in applications of microprocessor technology. Some of the major improvements are faster scan time, intelligent I/O (e.g., high speed counting and positioning), supervisory control capability, system documentation, local communication network, functional block instruction, ASCII message handling, providing production report and diagnosing its own failures and those of the machine or process.

10.2 PRINCIPLES OF OPERATION

The National Electrical Manufacturers Association (NEMA), USA, defines programmable controller as a digital electronic apparatus with a programmable memory for storing instructions to implement specific functions, such as logic, sequencing, timing, counting and arithmetic, to control machines and processes.

We shall now describe the basic building blocks of programmable controllers. We shall deal with the subject in the manner it developed so that we may understand the significance of building blocks of system.

Ladder diagrams are specialised schematics commonly used to document industrial control logic systems. They are called *ladder diagrams* because they resemble a ladder, with two vertical rails (supply power) and as many "rungs" (horizontal lines) as there are control circuits to represent. If we wanted to draw a simple ladder diagram showing a lamp that is controlled by a hand switch, it would look like this:



The L_1 and L_2 designations refer to the two poles of a VAC supply, unless otherwise noted. L_1 is the "hot" conductor, and L_2 is the grounded ("neutral") conductor.

In the early days logic control functions, i.e. timing, sequencing and control functions were provided through hardwired logic. Later programmable controllers have replaced these functions using relay logic based on three logic functions AND, OR and NOT. The combination of these functions determined whether a device must be switched *on* or *off*. The programmable controllers of early days used ladder diagram (also known as contact symbology) to programme these functions. A rung is the relay logic or contact symbology required to control an output. Figure 10.1 shows a typical ladder diagram rung in which X_1, X_2, X_3 and X_4 are input devices with normally open contacts and Y_1 is output device like control relay, pilot light, and so on.

The ladder rung means that in order to switch on Y_1, either X_1 and X_2 or X_3 and X_4 must be switched on.

$$Y_1 = (X_1 \cdot \text{AND} \cdot X_2) \cdot\text{OR}\cdot (X_3 \cdot\text{AND}\cdot X_4)$$

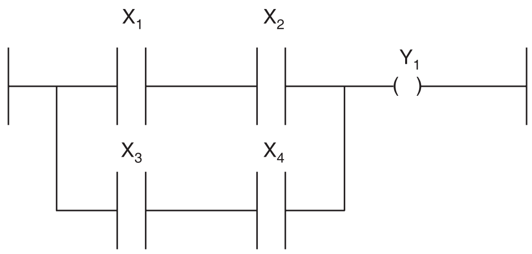


Figure 10.1 A ladder rung.

Following are common input and output devices used in a ladder rung.

Input devices	**Output devices**
Push button	Control relay
Selector switch	Solenoid valve
Proximity switch	Pilot light
Limit switch	Horn
Timer contacts	Timer

Symbols used in ladder diagram rung have specific functions. By and large they donot represent the input-output device specification but do convey the specific switching function. Examples of ladder diagram symbols are.

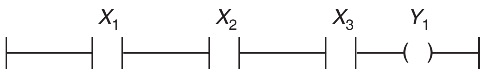
—| |— Normally opened contact
—| / |— Normally closed contact
—( )— Output
—(/)— Not output

The input may be a relay or switch contact. The normal position may be open or close. In the normally opened contacts the current flow will be affected when external command (manual push button, or electric relay) closes the contact. Similarly in normally closed contacts, the current flow will cease only when external command is given. The output may be one of the output devices mentioned above. In normal output devices, the current flows when all the input contacts are closed. Other types of output devices are known as “Not Output” devices. In these devices, the current flow is normally there but when all the input contacts are made, the current flow ceases:

In general, a ladder rung consists of a set of input conditions represented by contact instructions and an output instruction at the end of rung represented by the coil symbol. The coils and contacts are basic symbols of ladder diagram instruction set. All outputs are represented by coil symbol. Thus, instruction “Energize Coil” means “Turn on the Output”. While programming the ladder diagram, each contact and coil is referenced with an address number which identifies the port, variable or unit. This address number references the memory address of variable, data table or port.

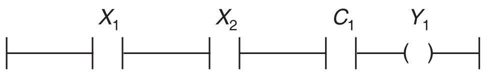
We shall now construct some ladder diagrams based on the operators of AND, OR and NOT.

10.2.1 AND Operation (Series Circuit)

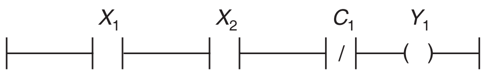


$Y_1 = X_1$ .AND. X_2 .AND. X_3

All the input contacts and output device are connected in series and when all the input contacts are true, output device is actuated. In the above example, X_1, X_2, X_3 may be limit switches with normally open contact. But these may also be control relays with normally open or normally closed contacts.



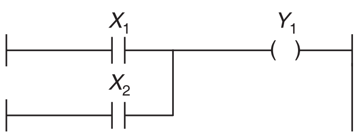
$Y_1 = X_1$ .AND. X_2 .AND. C_1



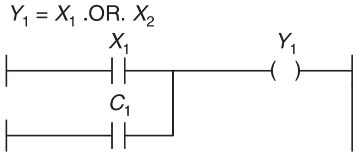
$Y_1 = X_1$ .AND. X_2 .AND. (NOT. C_1)

10.2.2 OR Operation (Parallel Circuit)

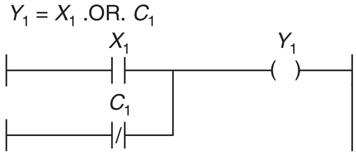
The input contacts are connected in parallel to each other and in series to output device. In order to activate output, one of the contacts may close, so that the circuit is completed. Examples of parallel circuits are



$Y_1 = X_1$ .OR. X_2



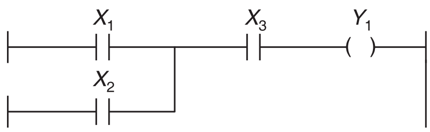
$Y_1 = X_1$ .OR. C_1



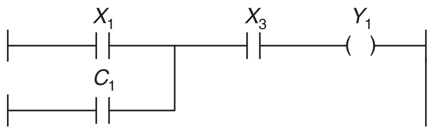
$Y_1 = X_1$ .OR. (NOT. C_1)

10.2.3 AND-OR Operations (Series-Parallel Circuits)

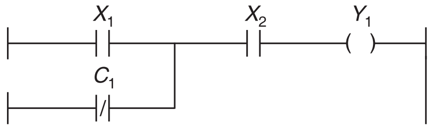
Some of the input contacts may be in series and others may be in parallel to each other.



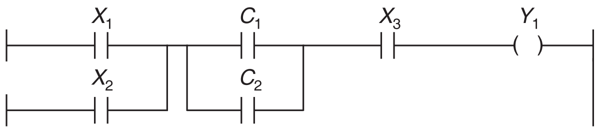
$Y_1 = (X_1 \text{ .OR. } X_2) \text{ .AND. } X_3$



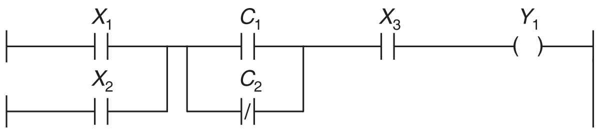
$Y_1 = (X_1 \text{ .OR. } C_1) \text{ .AND. } X_3$



$Y_1 = (X_1 \text{ .OR. (NOT. } C_1) \text{ .AND. } X_2$

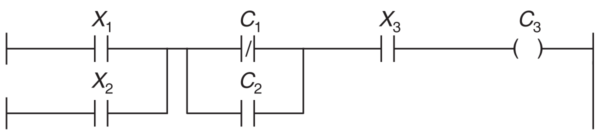


$Y_1 = (X_1 \text{ .OR. } X_2) \text{ .AND. } (C_1 \text{ .OR. } C_2) \text{ .AND. } X_3$



$Y_1 = (X_1 \text{ .OR. } X_2) \text{ .AND. } (C_1 \text{ .OR. (NOT. } C_2)) \text{ .AND. } X_3$

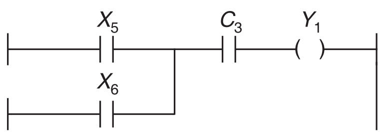
The output device may be a control relay which may be an input for other operations. Such output devices are known as internal outputs.



$C_3 = (X_1 \text{ .OR. } X_2) \text{ .AND. ((NOT. } C_1) \text{ .OR. } C_2) \text{ .AND. } X_3$

The contro1 relay C_3 may occur in another ladder diagram rung as input contact. This is very useful concept to provide interfacing of operations. In the above example, either X_1 or X_2

and C_2 or not C_1 and X_3 are to close in order to activate C_3. Now, C_3 may appear in another rung as,



Y_2 = (X_5 .OR. X_6) . AND. C_3

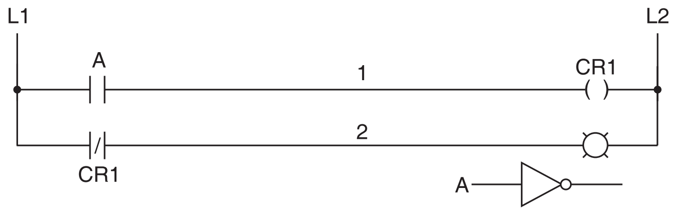
Thus until C_3 is activated Y_2 cannot be activated. In other words until the operations of previous rung explained are completed, this rung cannot proceed.

It is common to use boolean operators like ., + and – in place of AND, OR and NOT, the last two equations will now take the following form:

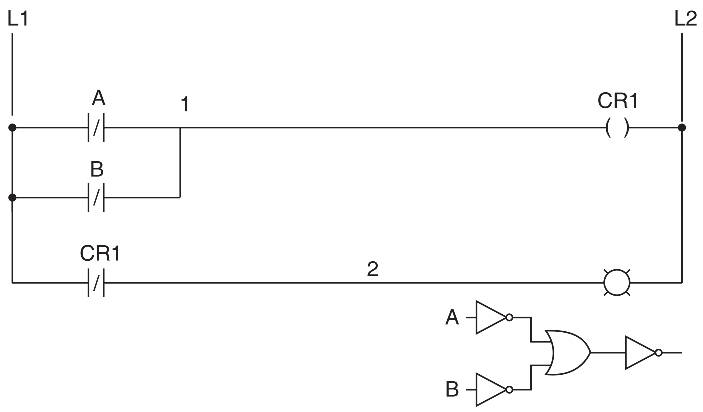
$$C_3 = (X_1 + X_2) \cdot ((-C_1) + (C_2)) \cdot X_3$$

$$Y_2 = (X_5 + X_6) \cdot C_3$$

We shall now take some examples. In the examples, we will call relay as control relay or CR. In the ladder diagram shown below, when the coil of CR1 (symbolised with the pair of parentheses on the first rung) is energised, the contact on the second rung opens, thus deenergising the lamp. From switch A to the coil of CR1, the logic function is non-inverted. The normally closed contact actuated by the relay coil CR1 provides a logical inverter function to drive the lamp in a direction opposite that of the switch's actuation status.



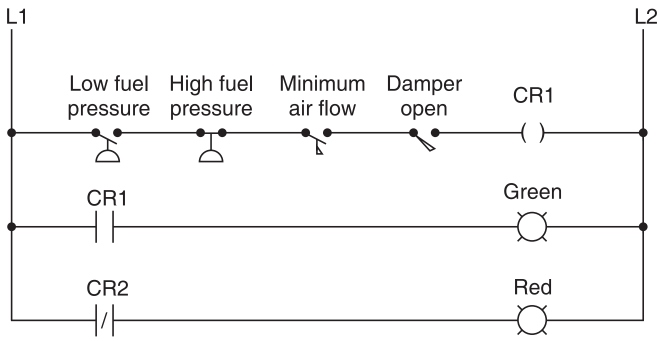
Applying this inversion strategy to one of our inverted-input functions created earlier, such as the OR-to-NAND, we can invert the output with a relay to create the following non-inverted function:



From the switches to the coil of CR1, the logical function is that of a NAND gate. CR1's normally-closed contact provides one final inversion to turn the NAND function into an AND function.

Permissive Circuit Design

A practical application of switch and relay logic is in control systems where several process conditions have to be met before a piece of equipment is allowed to start, e.g., burner control for large combustion furnaces. In order for the burners in a large furnace to be started safely, the control system requests "permission" from several process switches, including high and low fuel pressure, air fan flow check, exhaust stack damper position, access door position, etc. Each process condition is called a *permissive*, and each permissive switch contact is wired in series, so that if any one of them detects an unsafe condition, the circuit will be opened. The high fuel pressure contact is normally closed. It will open if the fuel pressure gets too high, i.e., in unsafe condition.



Green light = Conditions met: safe to start

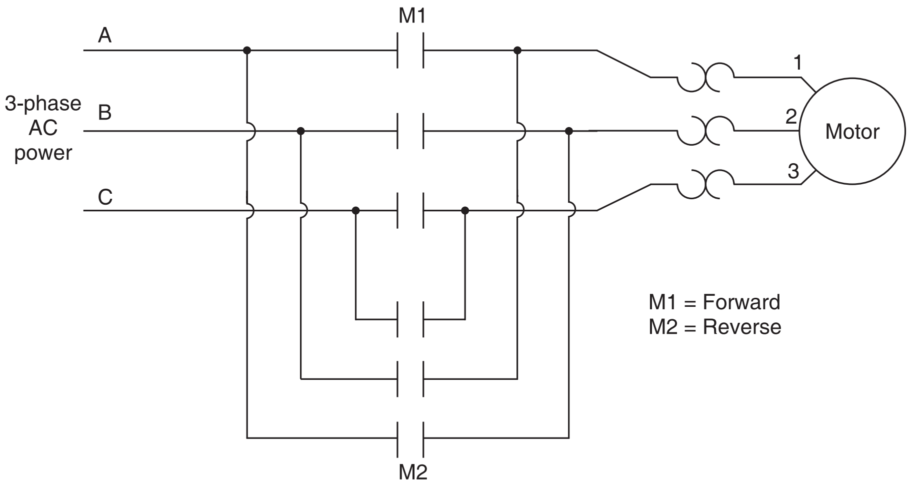
Red light = Conditions not met: unsafe to start

If all permissive conditions are met, CR1 will energise and the green lamp will be lit. The starting of burner control will be manual in this case. In real life, a control relay or fuel valve solenoid would be placed in the rung of the circuit to be energised. It will be activated when all the permissive contracts were "good", that is, all closed. If any one of the permissive conditions are not met, the series string of switch contacts will be broken, CR2 will de-energise, and the red lamp will light up.

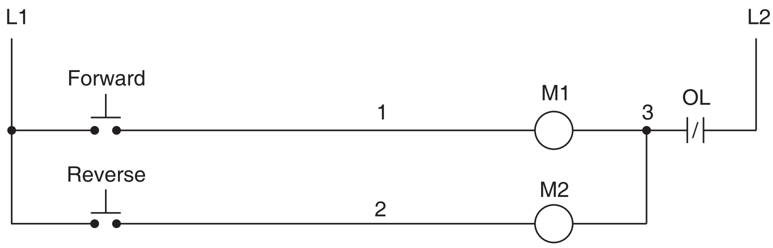
Interlock Circuit Design

Another practical application of relay logic is in control systems where it is required that two incompatible events should not occur at the same time, e.g., in reversible motor control, where two motor contactors are wired to switch polarity (or phase sequence) to an electric motor, the forward and reverse contactors should not be energised simultaneously.

When contactor M1 is energised, the 3 phases (A, B, and C) are connected directly to terminals 1, 2 and 3 of the motor, respectively. However, when contactor M2 is energised,



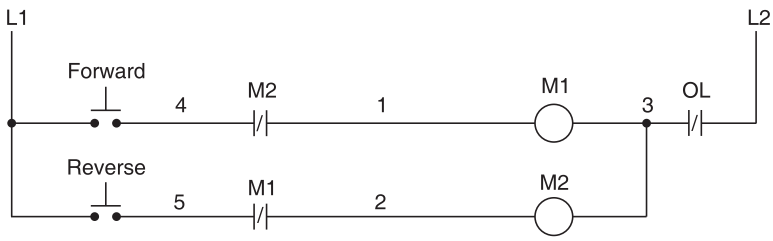
phases A and B are reversed, with A going to motor terminal 2 and B going to motor terminal 1. This reversal of phase wires results in the motor spinning in the opposite direction.



The normally-closed "OL" contact, which is the thermal overload contact activated by the "heater" elements, is wired in series with each phase of the AC motor. If the heaters get too hot, the contact will change from its normal (closed) state to the open state, which will prevent either contactor from energising.

This control system will work fine, so long as no one pushes both buttons at the same time. If someone were to do that, phases A and B would be short-circuited together by virtue of the fact that contactor M1 sends phases A and B straight to the motor and contactor M2 reverses them; phase A would be shorted to phase B and vice versa.

To prevent this from happening, the circuit must be so designed that the energisation of one contactor prevents the energisation of the other. This is called *interlocking*, and it is accomplished through the use of auxiliary contacts on each contactor, as shown below.



When M1 is energised, the normally closed auxiliary contact on the second rung will be open, thus preventing M2 from being energised, even if the "reverse" pushbutton is actuated. Likewise, M1's energisation is prevented when M2 is energised. Note as well how additional wire numbers (4 and 5) are added to reflect the wiring changes.

10.3 ARCHITECTURE OF PROGRAMMABLE CONTROLLERS

The programmable controllers are basically computer-based and therefore, their architecture is very similar to computer architecture. The memory contains operating system stored in fixed memory like ROM, rather than disk in case of computers. The application programs are stored in Read-Write portion of memory.

All programmable controllers contain a Central Processing Unit (CPU), Memory, Power Supply, Input/Output (I/O) modules and programming device. Figure 10.2 shows architecture of programmable controllers.

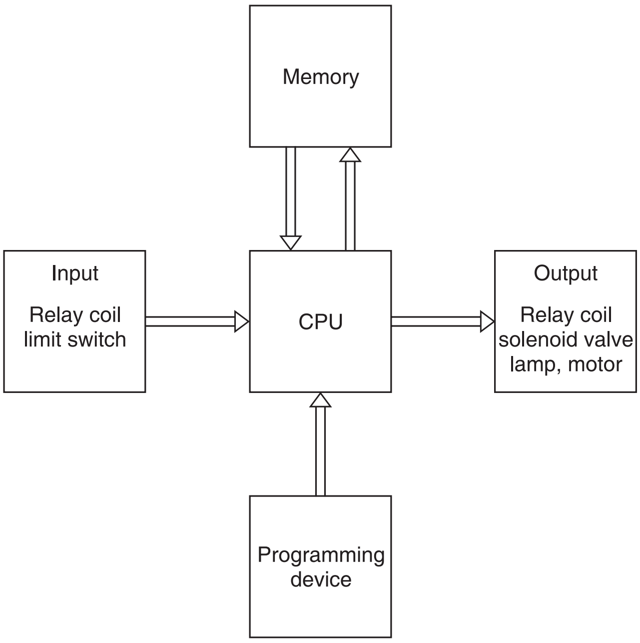


Figure 10.2 Architecture of programmable controllers.

The CPU, upon receiving instructions from the memory together with feedback on the status of the input-output devices, generates commands to the outputs. These commands control the output elements on a machine or process. Devices such as relay coils, solenoid valves, indicator lamps and motor starters are typical loads to be controlled.

The machine or process input elements transmit status signals to the input modules which, in turn, generate logic signals for use by the CPU. In this way, the CPU monitors elements such as push buttons, selector switches and relay controls on a machine or process.

The intelligence of programmable controller is derived from microprocessors which have tremendous computing and control capability. They perform all mathematical operations, data handling and diagnostic routines that were not possible with relays or their predecessor, the hardwired processor.

The power of programmable controller depends upon the type of microprocessor used. Small size programmable controllers use 8-bit microprocessor. Higher end controllers use bit-slice microprocessor to achieve faster instruction execution.

The operating system is the main work horse of system. It is necessary to distinguish between the instructions used by operating system to command the microprocessor and the instructions used by the programmable controller to handle the specific control problem. The operating system performs the following tasks:

— Execution of application program
— Memory management
— Communication between programmable controller and other units
— I/O interface handling
— Diagnostics
— Resource sharing

During program execution the processor reads all the inputs, takes these values and according to control application program, energizes or de-energizes the outputs, thus solving the ladder network. Once all the logic has been solved, the processor will update all the outputs. The process of reading the inputs, executing the control application program, and updating the output is known as SCAN. During a scan, processor also performs housekeeping tasks.

There are four basic steps (Fig.10.3) in the operation of all PLCs; Input Scan, Program Scan, Output Scan, and Housekeeping. These steps continually take place in a repeating loop.

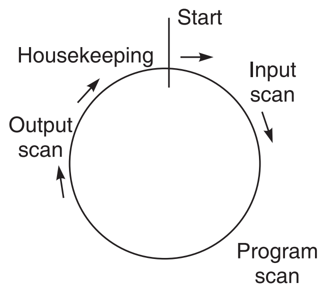


Figure 10.3 Programmable controller scan.

1. **Input Scan.** This detects the state of all input devices that are connected to the PLC.
2. **Program Scan.** This executes the user created program logic.
3. **Output Scan.** This energises or de-energises all output devices that are connected to the PLC.
4. **Housekeeping.** This step includes communications with programming terminals, internal diagnostics, etc.

10.3.1 Diagnostics

One of the processor tasks during the housekeeping operation is to check the soundness of the system. To achieve this, the processor performs error checks and sends status information to indicators that are generally located on the front of the CPU. Typical diagnostic informations include CPU running OK, battery OK and power supply OK. The CPU also has "watch-dog" timer output. The processor sends a pulse at the end of each scan indicating a correct system operation, If there is a malfunction, the timer would time-out and fault output would be activated.

10.3.2 Input/Output System

The input/output (I/O) system provides physical connection between the external devices (field equipment) and the CPU. Through various interface circuits, the controller can sense and measure physical parameters of a machine or process, such as proximity, position, motion, level, temperature, pressure, current and voltage. We have already discussed the sensors for various parameters in Chapter 2. Based on status sensed or physically measured values, the CPU issues commands that control various devices such as valves, motors, pumps and alarms. The details about valves and actuators have been described in Chapter 4.

Earlier versions of programmable controllers had interface circuits that could translate voltage level signals from limit switches (or push buttons etc.) in the field to the logic voltage signals required by the electronics in the CPU. Similarly, output circuits translated the logic voltage signal to levels appropriate to drive solenoid valves, motor starters, lamp, etc. These types of inputs and outputs are called discrete I/O. Modern programmable controllers possess a complete range and variety of discrete and data I/O (which includes analog I/O, register I/O and digital pulse tacho inputs).

Typical AC/DC input, AC/DC output and contact output circuits are shown in Fig. 10.4 (a-c)

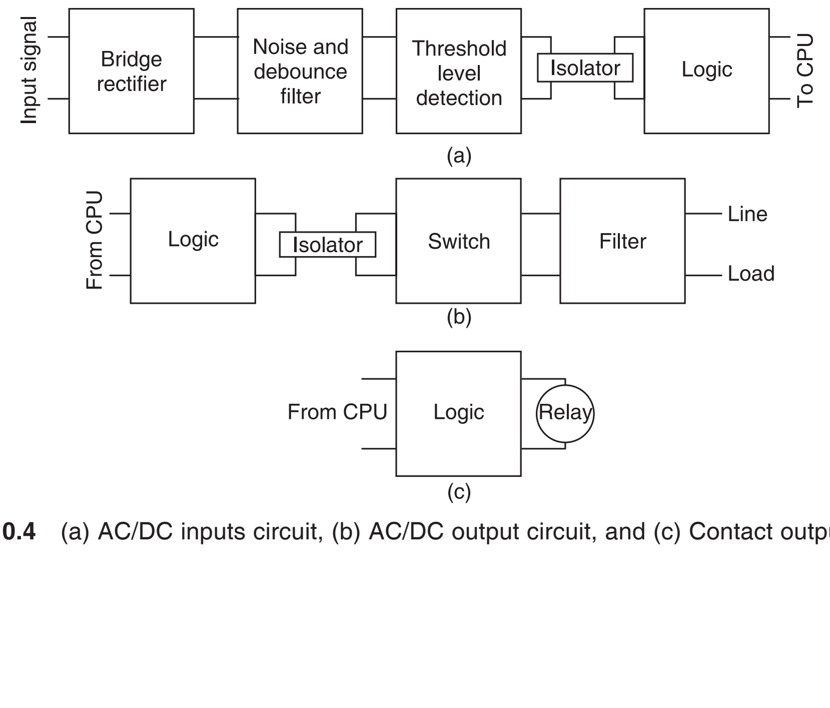


Figure 10.4 (a) AC/DC inputs circuit, (b) AC/DC output circuit, and (c) Contact output circuit.

Some of the programmable controllers offer remote I/O capability. This helps in locating groups of I/O away from CPU. Connection between CPU and remote I/O group is via twisted-pair wires, or a single coaxial cable or fibre-optic data link. Figure 10.5 (a) and (b) shows typical remote I/O configurations. Both star as well as Daisy chain type interconnection are possible. Distance of remote I/O varies from one manufacturer to another but typical distance could be in the range of 2 to 4 km.

10.3.3 Programming Devices

There are different types of programming devices available. Most commonly used programming devices are CRT and LED/LCD types. CRT type or large LCD type programmers use microprocessor and are generally intelligent and can also function independent (off line) of programmable controller and can be used to create, edit and monitor programs.

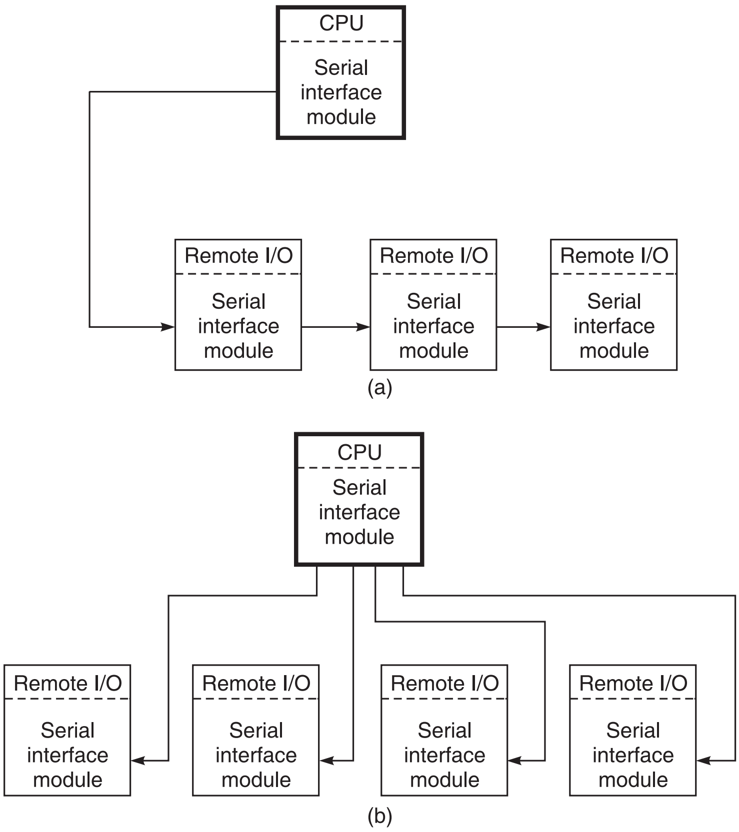


Figure 10.5 (a) Remote I/O daisy chain configuration, and (b) Remote I/O star configuration.

Their diagnostic capabilities are extensive, including full screen presentation of the program, with real time display of the status of each contact and output.

Many of the programmable controllers allow a single programming device or programming panel to be connected to group of programmable controllers in local area network fashion. This permits parameter or program changes from one centralised programming device.

10.4 PROGRAMMING THE PROGRAMMABLE CONTROLLERS

10.4.1 Programming Languages

The basic ingredients of languages for programmable controller are same as any other computer assembly language. The basic groups of operations in any assembly language are:

- Data transfer
- Data manipulation
- Arithmetic and logic
- Flow control
- Special functions

The computer engineers know that these operations are common in any computer-based systems and only their representations differ. Not to say that there are always special requirements of the systems and environment. These are covered under special functions. In case of programmable controllers, following types of programming languages are prevalent:

Lower level

- Ladder diagrams
- Boolean mnemonics

Higher level

- Functional blocks
- English statements

We have already discussed in brief about basics of *ladder diagrams*. The *Boolean mnemonics* are similar to assembly language of any computer/microprocessor where operation mnemonics are used to describe the various operations to be performed. The Boolean mnemonics replaces the ladder diagram symbols in one-to-one fashion. The *functional block* is basically a block-oriented language and has one function block for each operation. Such function blocks are easy to understand, as the data transfer from one block to another can be seen pictorially. The English statements are like higher level language of computer with English-like sentences. Those who are aware of BASIC, FORTRAN, PL/1, ALGOL, C, etc. may well visualise the characteristics of English-like languages. These characteristics are easy to understand and program, but the programs written in such languages are slower than those written in boolean mnemonics or ladder diagrams. We shall now deal with these languages with respect to various groups of operations mentioned above.

10.4.2 Ladder Diagram Instructions

Special functions

The special functions in ladder diagrams will relate to relay logic operations, timer and counter operations.

(i) ***Relay logic operations:*** Some of the features of relay logic operations were discussed while explaining the concept of ladder diagrams.

Let us now consider all the relay ladder diagram instructions commonly available—

- —| |— — Normally open contact
- —| / |— — Normally closed contact
- —( )— — Energize coil (Turn on output)
- —(/)— — De-energize coil (Turn off output)
- —(L)— — Latch coil (output is turned on and remains on even if logic continuity is not there due to change in input conditions. The output can be turned off only by unlatch coil instruction).
- —(U)— — Unlatch coil
- —| ↑ |— — OFF-ON transitional contact provides a one shot pulse whose width is same as single scan time, when trigger makes a OFF to ON transition. The trigger may come from external source or may be on internal output.
- —| ↓ |— — ON-OFF transitional contact provides a one shot pulse of one scan time width, when trigger makes a ON to OFF transition. The trigger may come from external source or may be an internal output.

(ii) ***Timer and counter operation:*** Basically, the operation of both timer and counter is same as timer operates like counter. Both timer and counter maintain two values in their internal registers, viz. preset value and the count value.

The preset value (PR) is decided by the programmer and stored during execution of ladder rung. The count value refers to the present count of the signal. This signal may refer to any event, which may occur randomly. This count indicates as to how many times the event has occurred after initialisation. When the count becomes equal to preset value then output is generated (energize/de-energize the coil) in the ladder rung. This is basic counter operation.

When signal occurs at fixed frequency, i.e., after every fixed interval of time, the counter performs as timer. Now 10 pulses, i.e. 10 counts will mean an elapsed time of *5* seconds, if signal is occurring after a regular interval of 0.5 seconds. When the count becomes equal to preset value (i.e. elapsed time = preset time value), output is generated in the ladder diagram rung. Thus for timer operation, the time base (i.e. regular fixed interval of signal) must also be defined. Following are the main ladder diagram instructions:

- —(TON)—Time Delay Energize (ON)
  PR =
  TB =

This instruction is used to perform delay function for any event when logic continuity is present.

Total delay = (Preset) PR × TB (Time Base)

If logic continuity is lost in between the counting, the count value is reset to zero. When preset value equals the count value, output coil is energized (*turned on*).

- —(TOF)—Time Delay De-energize (OFF)
  PR =
  TB =

The instruction is used to perform delay function for any event when logic continuity is absent.

Total delay = PR (Preset) × TB (Time Base).

If logic continuity is gained in between the counting, the count value is reset to zero. When preset value equals count value the output coil is de-energized *(turned off)*.

- —(RTO)—Retentive Timer Output

  PR =

  TB =

When it is required that the timer should not loose the count value when power fails or when logic continuity is changed, then RTO instruction is used. The operation is same as TON instruction. When logic continuity is changed or power fails, then count value is retained and in the event of re-establishing of lost logic continuity or power the counting begins again. The only way to reset the count value is by executing Retentive Timer Reset (RTR) instruction.

- —(RTR)—Retentive Timer Reset

If logic continuity exists, then count is reset to zero in retentive timer.

- —(CTV)—Up Counter

  PR =

Each time the programmed event occurs, the count increments by one. When preset equals the count, the output is energized. The count can be reset by counter reset instruction.

- —(CTD)—Down Counter

  PR =

Each time the programmed event occurs, the count is decremented by one. When preset equals count the output is energized. The count can be reset by counter reset instruction.

- —(CTR)—Counter Reset

When logic continuity is established, the addressed counter is reset.

Data transfer and data manipulation operations

This group of operations involve reading and writing of data from/to memory or register and comparing two data items to ‘equal to’, ‘less than’ or ‘greater than’ conditions. The memory locations are specified by the addresses. Following are the main ladder diagram instructions:

ADDR

- —|GET|— Get word

Get the data stored in the memory at specified address ‘ADDR’ and store it in the internal register for immediate use of other operations.

ADDR

- —|PUT|— Put word

Store the data of internal register in the memory at location specified by address ‘ADDR’.

ADDR

- —|CMP = |— Compare equal

Compare the content of memory location specified by ADDR with internal register or contents of two internal registers for equal to condition. The contact is closed if the operation is true.

ADDR

- —|CMP < |— Compare less than

Compare the content of memory location specified by ADDR with internal register content or contents of two internal registers for ‘less than’ condition. The contact is closed if operation is true.

ADDR

- —|CMP > |— Compare greater than

Compare the content of memory location specified by ADDR with internal register content or contents of two internal registers for ‘greater than’ condition. The contact is closed if operation is true.

Arithmetic operations

The arithmetic operations like addition, subtraction, multiplication and division are performed in this group. The operands may be two memory location contents with result stored in one or two memory locations (or registers). Following are the main ladder diagram instructions:

- —(+)— Addition performs the addition of two operand values specified. The result may be in one of the operand locations or may be specified separately.
- —(-)— Subtract performs the subtraction of two operand values specified. The result may be in one of the operand locations or may be specified separately.
- —(×)—(×)— Multiplication performs the multiplication of two operand values specified. The result is normally held in two operand registers.

  —(÷)—(÷)—Division performs the division of two operand values specified. The result is normally held in two operand registers. The first result register holds the integer value, while the second holds the decimal fraction.

Flow control operations

This group of instructions alters the sequential execution of program based on the outcome of conditions specified. The main types of operation under this group are —(MCR)—, —(SKD)—, —(ZCL)— and —(SKR)—, —(LBL)—, —(JMP)—, —(JSB)— and —(RET)—

The first four instructions namely —(MCR)—, —(SKD)—, —(ZCL)— and —(SKR)— perform the same operation with little variation. The MCR (Master Control Relay) and SKD (Skip and De-energize) instructions are exactly same whereas ZCL (Zone Control Last) state and SKR (Skip and Retain) instructions are exactly same.

The operation performed by these instructions is shown by the following If Then Else statement block

If (COND) = TRUE
Then
    Begin
    :
    End
Else
    Begin
    :
    End

If the specified conditions are true, then a block of instructions specified between the two boundaries of 'Begin' and 'End' are executed. If the specified conditions are not true, then execution of program jumps to first instruction outside the boundary of block.

The MCR, ZCL, SKD and SKR perform the begin operation to define the beginning of instruction block while, respective END MCR, END ZCL, END SKD and END SKR instructions signify the end of instruction block.

Following are the difference between these instructions.

1. When specified condition is False then all the non-latched outputs are de-energized in case of MCR and SKD.
2. When specified condition is False then all the output within the block will be held in their last state, in case of ZCL and SKR.

I.D
- —|LBL|— Label

It is used to associate an identification number to a particular ladder rung. It has no role in execution or program logic otherwise. The label is used in jump and jump to subroutine instructions.

I.D
- —(JMP)— Jump

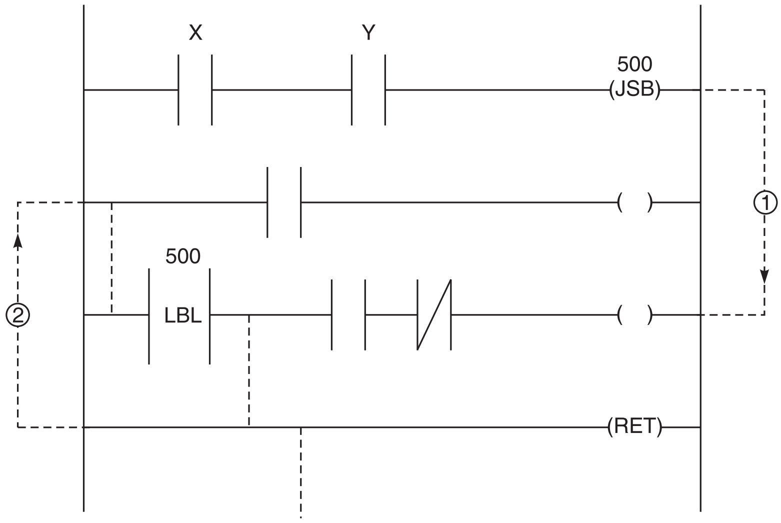
The program execution jumps to the rung with label I.D. The conditions may be specified in the rung before JMP to make it conditional jump instruction.

I.D
- —(JSB)— Jump to Subroutine

The program execution jumps to the ladder rung specified by the I.D. However before executing the jump, the address of next ladder rung (Return address) is stored in internal register for the purpose of return at the end of *subroutine*.

- —(RET)— Return from Subroutine

When encountered, the program execution returns to the main program from where the subroutine jump was taken. The return address of ladder rung is retrieved back from internal register and program execution jumps to return address. The subroutine operation is explained in Fig. 10.6.



① If X and Y close. The address of next rung is stored and program jumps to rung with label as 500.

② When RET is encountered then return address is retrieved and program execution returns.

Figure 10.6 Subroutine branch and return sequence.

Operation of pneumatic cylinder using ladder diagram

Figure 10.7 shows a basic circuit for the controlled operation of a double-acting cylinder using a solenoid operating 4/2 electropneumatic valve, a control relay and a switch. The normal position is shown in Fig. 10.7. Port P_1 is connected to the exhaust, P_2 is connected to the air supply, the rod is inside the cylinder, and the valve is not actuated.

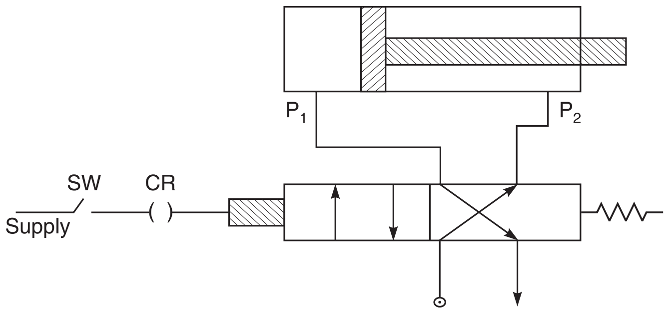
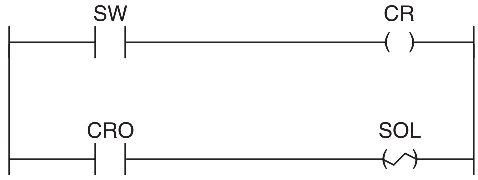


Figure 10.7 Controlled operation of double-acting cylinder.

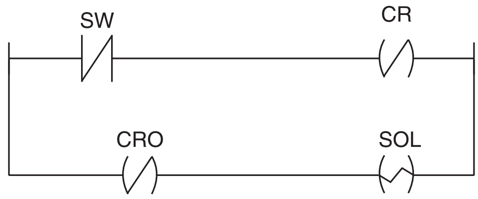
I. When switch SW is closed, the relay is energised.

II. This results in the normally open contact of the relay CRO being closed, the solenoid being powered and the valve being actuated.



III. The valve movement will take some time, during which no operation must be performed. Thus, time delay must be incorporated. During this time, the solenoid will move the valve, and thus P_1 will be connected to the air supply and P_2 will get connected to the exhaust. This will result in the movement of the rod outside the pneumatic cylinder. When the time is elapsed, the forward stroke is completed and the rod is fully outside. Coil CL is in the energised state and the normally open contact CRO is closed.

IV. Now at any time, the operator may opt for reverse stroke by opening the switch SW. This will result in the control relay being de-energised. The relay contact CRO will open consequently.



This will result in air supply to move the solenoid being cut, and so it will retract with the force of the spring. The valve will move to connect P_1 to the exhaust and P_2 to the air supply, and the reverse stroke of the cylinder will take place.

The ladder diagram for the operation of the double-acting cylinder through the 4/2 pneumatic valve is shown in Fig. 10.8.

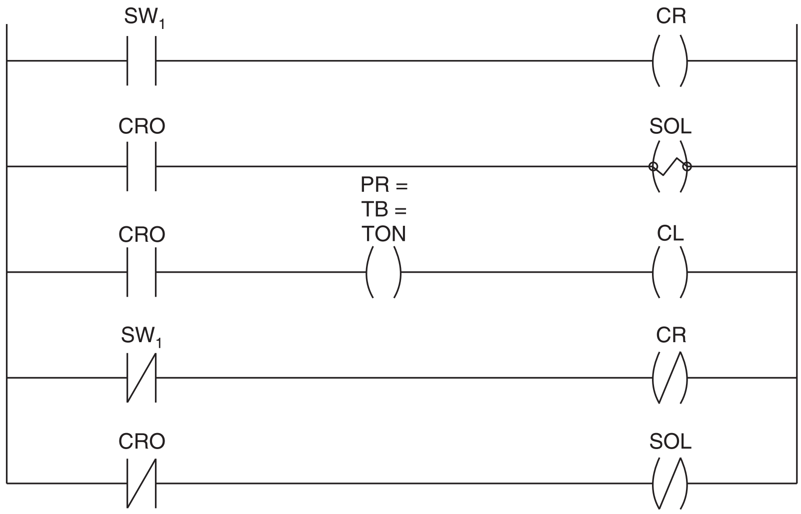
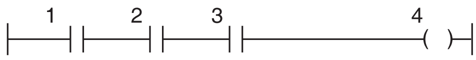


Figure 10.8 Ladder diagram for operation of double-acting cylinder.

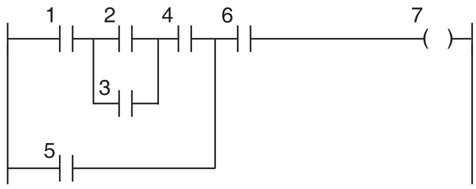
Program execution sequence

The execution sequence of the program instructions is shown below.

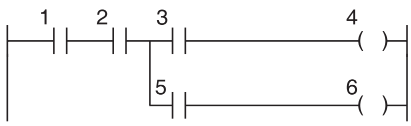
1. They are executed in the sequence from block 1 to the final block, which contains the ENDE instruction (or IRET in an interrupt program).
2. They are executed in the sequence from rung 1 to the final rung in a block (or the END instruction).
3. They are executed according to the following rules in any one rung:
   (a) When there is no vertical connection, they are executed from left to right.



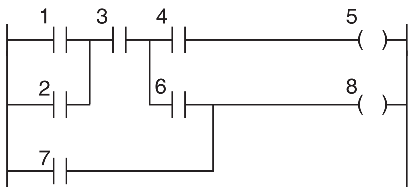
   (b) When there is an OR connection, the OR logic portion is executed first.



   (c) When there is a branch, they are executed in order from the upper line to the lower line.



   (d) A combination of (b) and (c) above.



10.4.3 Boolean Mnemonics

These are assembly level languages specially designed to express the logic of programmable controllers. There is nearly one to one correspondence between Boolean mnemonics and various ladder diagram symbols. Those, who are familiar with the assembly language of any computer will recognize the mnemonics and their meaning as different assembly languages have similar operations. Let us now illustrate the similarity between ladder diagram operations vis-a-vis Boolean mnemonics. Following are the ladder diagram instructions and corresponding Boolean mnemonic.

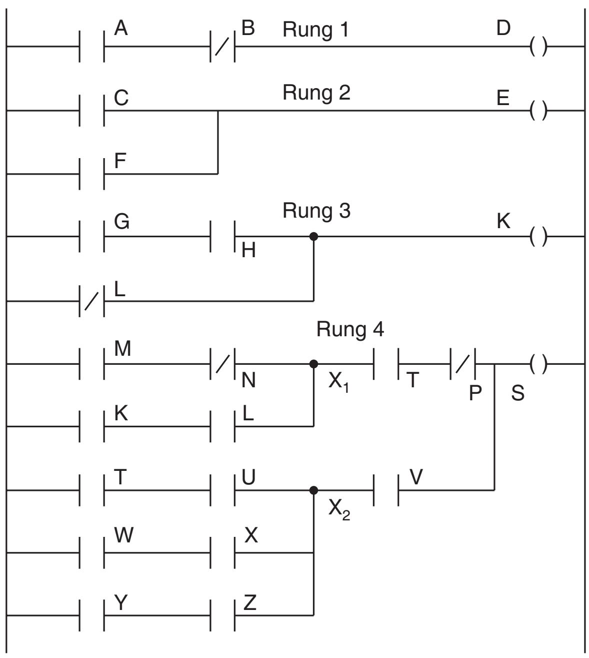
Mnemonic	*Function*	*Ladder equivalent*
LD	Load	—| |—
AD	And	—| |—| |—
OR	Or	—| |— (parallel)
OUT	Energize coil	—( )—
OUT NOT	De-energize coil	—(/)—
TIM	Timer	—(Ton)—
ADD	Addition	—(+)—
MUL	Multiplication	—(×)—
SUB	Subtraction	—(–)—
DIV	Division	—(÷)—
CMP	Compare	—(Cmp)—
JMP	Branch	—(Jmp)—

While some of the mnemonics are from assembly language of processors, others particularly special functions can be programmed using macros.

The Fig. 10.9 shows a relay ladder diagram and corresponding Boolean program.

Rung 1
LD A
$\overline{\text{AD}}$ B
STO D
Rung 2
LD C
OR F
STO E
Rung 3
LD G
AD H
$\overline{\text{OR}}$ L
STO K
Rung 4
LD M
$\overline{\text{AD}}$ N
STO X_1
LD K
AD L
OR X_1
AD T
$\overline{\text{AD}}$ P
STO S
LD T
AD U
STO X_2

LD W
AD X
OR X_2
STO X_2
LD Y
AD Z
OR X2
AD V
OR S
STO S



The Instruction Set

Sl. No.	Description	Mnemonics	Sl. No.	Description	Mnemonics
1	Load	LD	6	Or Complement	$\overline{\text{OR}}$
2	Load Complement	$\overline{\text{LD}}$	7	No. Operation	NOP
3	And	AD	8	Store	STO
4	And Complement	$\overline{\text{AD}}$	9	Store Complement	$\overline{\text{STO}}$
5	Or	OR	10	Jump	JP

Figure 10.9 Boolean mnemonics and ladder equivalents.

Different programmable controllers use different Boolean language, even though the functions may be similar. Figure 10.10 shows another Boolean language mnemonics.

10.4.4 Functional Blocks

For process engineers, functional blocks are perhaps easiest to program for the blocks do not require any prior knowledge of computer or relay ladder diagrams. The programmable controller provides functional blocks in the same way as ready-made bricks or blocks for house building are provided. It is therefore a language which makes programming easy, because it is based on logical structure and contains only functions which are used exclusively in *process control*.

Sl. No.	*Mnemonic*	*Function*
1.	RDNO	Reads and stores status of input (NO contact)
2.	RDNC	Reads and stores status of input (NC contact)
3.	ANDC	Logical AND of inputs at all successive addresses
4.	ORC	Logical OR of inputs at all successive addresses
5.	AND	Logical AND
6.	OR	Logical OR
7.	CAND	Logical AND with complement of input status
8.	COR	Logical OR with complement of input status
9.	OUT	Send result of operation to output device
10.	OUTS	Energize output device
11.	OUTR	De-energize output device
12.	STMR	Store result in flag
13.	SCMR	Recall result from flag
14.	CACC	Complement
15.	ANDMR	Logical AND with flag
16.	ORMR	Logical OR with flag
17.	CANDMR	Logical AND with complement of flag
18.	CORMR	Logical OR with complement of flag
19.	STRTC	Start timer/counter
20.	RCTC	Recall timer/counter
21.	RSTC	Reset counter
22.	ANDTC	Logical AND with time/counter
23.	ORTC	Logical OR with time/counter
24.	CANTC	Logical AND with complement of timer/counter
25.	CORTC	Logical OR with complement of timer/counter
26.	END	End of program

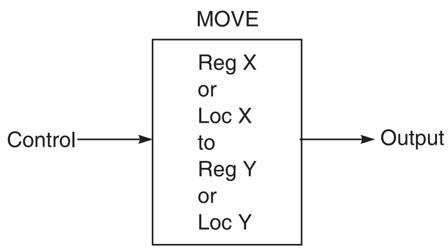
Figure 10.10 Typical Boolean language for programmable controller.

The common functional blocks include simple arithmetic and logic functions, viz., timer/ counters, comparators, shift registers, sequencers, PID block, regulators, fault signalling and so on. These functional blocks can be displayed on the computer screen using special function keyboard. These are represented as simple graphic symbols. The user has to enter the parameter and signal names and link the elements by means of cursor on the display. This is done in a language which makes it an extremely simple process. Following is brief description of functional blocks.

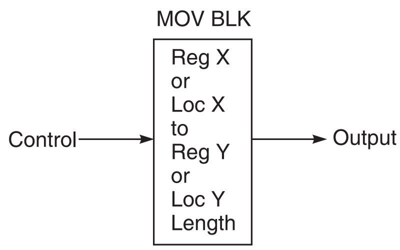
Data transfer operation

Following are the basic operations performed in this group.

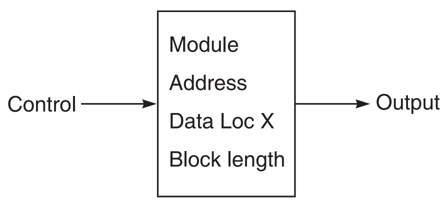
1. To move data from one location/register to another location/register. The operation is performed when control is energized and when operation is completed, output signal is energized.



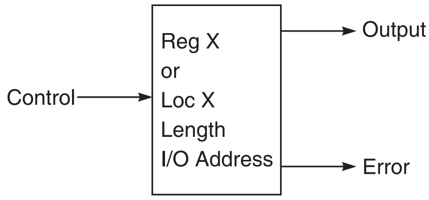
2. To move block of data from one group of memory locations/registers to other group of memory locations/registers.



3. Block transfer from/to Input/Output module. The input module may be ADC, BCD input, whereas output module may be stepper motor, DAC, display etc. A block of data stored in memory is transferred to module in case the module is output. The reverse operation takes place in case of input module.



4. ASCII transfer performs ASCII character transmission from programmable controller to peripheral devices.

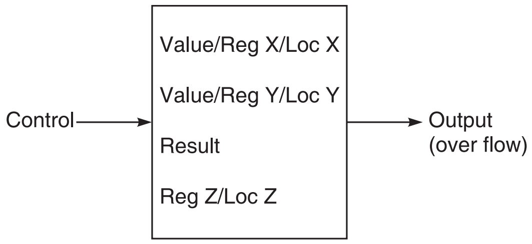


The error signal is energized when transmission cannot take place due to faulty mode or transmission fault.

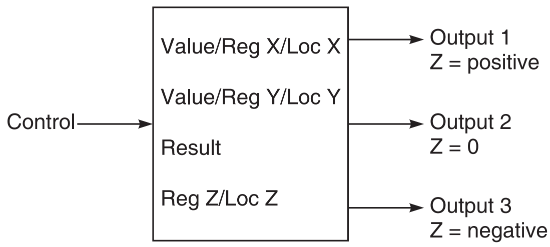
Arithmetic and logic operation

Most arithmetic operations use three registers/memory locations to operate. Two locations/ registers are used to specify two operands and third register/location is used to store result. The operands can also be specified as immediate value, in which case, the first two locations/ registers will not be required.

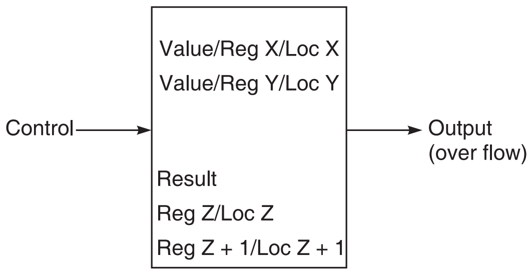
(i) *Addition*



(ii) *Subtraction*

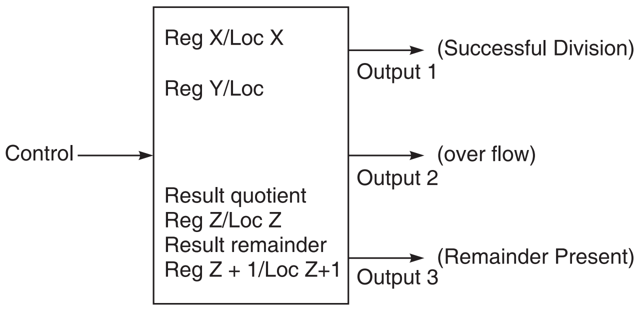


(iii) *Multiplication*



Some programmable controllers provide the facility of rounding off in which case 'Scale' parameter is also defined.

(iv) *Division*



(v) *Logic matrix:* The logic matrix is used to perform, AND,OR, EX-OR, NOT,NAND,NOR. functions on multiple group of operands.

As an example, one wants to AND memory locations m to $m + n$ with memory locations k to $k + n$ and store the result in l to $l + n$. This operation is depicted in Fig. 10.11.

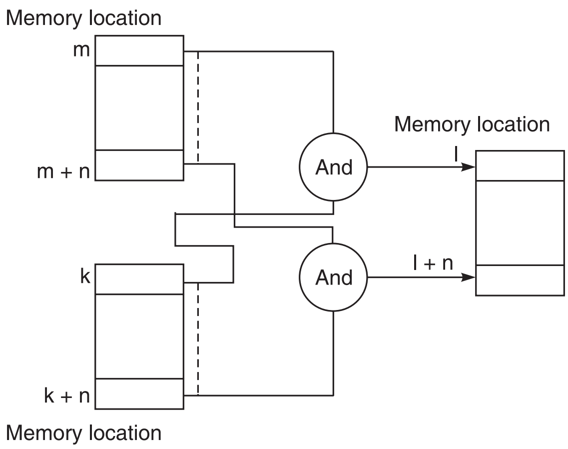
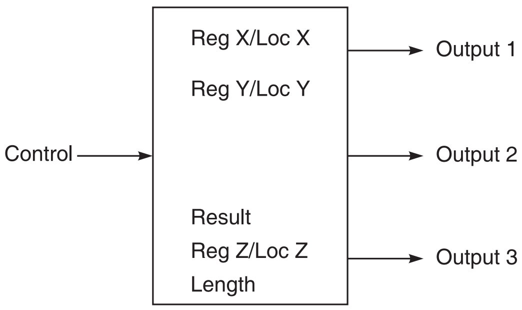


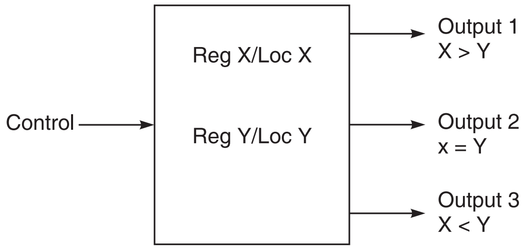
Figure 10.11 Logic matrix operation.

The generalised logic function symbol is given by



Output 1 is energized when logic operation starts;
Output 2 is energized when there is error; and
Output 3 is energized when operation is completed.

(vi) *Comparators:* This functional block is used to compare two values stored in registers or memory.



(vii) *shift:* The operation of shift instruction is shown in Fig. 10.12.

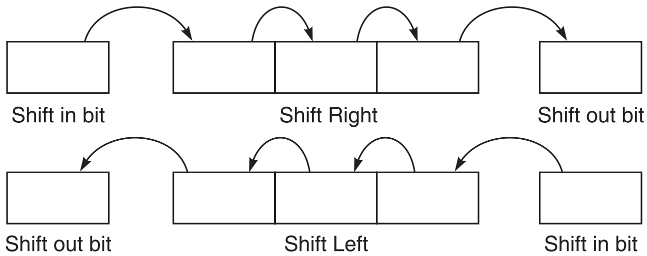
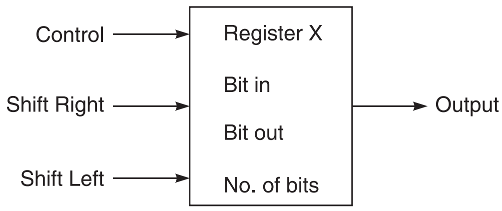


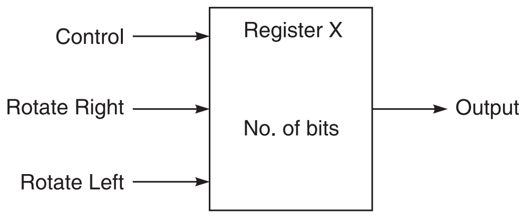
Figure 10.12 Shift operation.

The register holds the operand on which shift operations are to be performed. The two bits 'shift in' and 'shift out' have to be defined by the user. These bits can be used to check the status of particular bit of register. The number of shifts in right or left directions also needs to be specified by user. The symbol for shift is presented as



Output is energized when the operation is completed.

(viii) *Rotate:* Rotate operation is same as shift operation (Fig. 10.12). The difference is that while bits shifted out are lost in shift, they are restored as LSB (Rotate Left) or MSB (Rotate Right) in case of rotate operations. The symbol for rotate is,



(ix) *Timer/Counter:* The timer and counter functional blocks are same as their hardware I/O connections. Timer requires following information to function

Time base, i.e. frequency of time pulses

Preset value, i.e. time value till which timer should function

Current value, i.e. current value of accumulated time in timer

Following are the signals in timer/counter block:

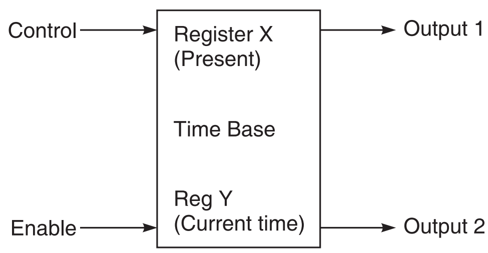
Control—acts as chip select

Enable—When activated, acts as run command. The timer runs till it is enabled.

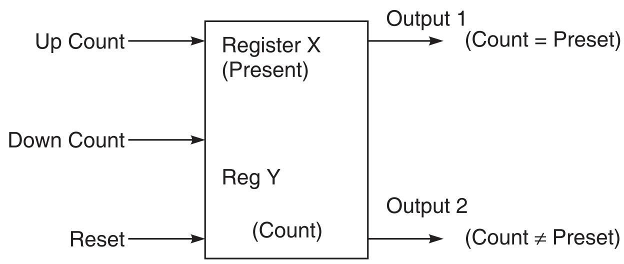
Output 1— (Time = Preset) i.e. Time up signal

Output 2 —(Time ≠ Preset) i.e. Time not up signal

The functional block is shown below.

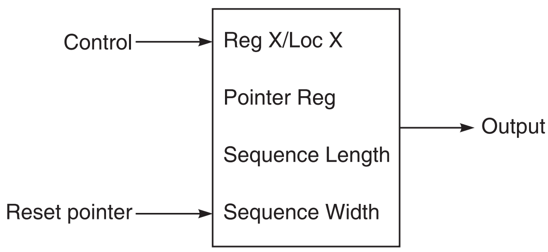


The counters are similar to timer in operation. They do not require time base as they count events. Thus they have two input signals for up and down counting. A preset value is stored and accumulated count is maintained as shown in the following figure.



The count resets to zero when Reset is energized.

(x) *Sequencer:* Sequencer block basically outputs contents of a sequence table in incremental manner in response to a control pulse. It is very useful in sequential control where sequence for any operation is pre-defined.



The pointer register contains the pointer to the table. Sequence length refers to the total no. of steps in table and sequence width means the no. of bits in each step. On each OFF-ON transition of control signal, the pointer register increments and content of table addressed by pointer register are placed at output.

(xi) *PID control:* The PID function block performs Proportional, Integral and Derivative Control. Following are the basic parameters needed to implement PID algorithm:
- — Proportional gain
- — Integral gain
- — Derivative gain
- — Set-point

In addition to these, current value of 'input variable' as well as control variable must also be know (Fig. 10.13).

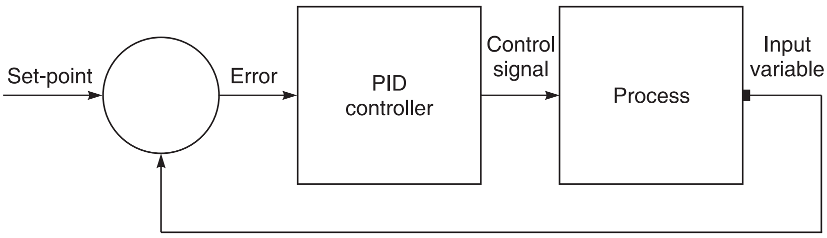
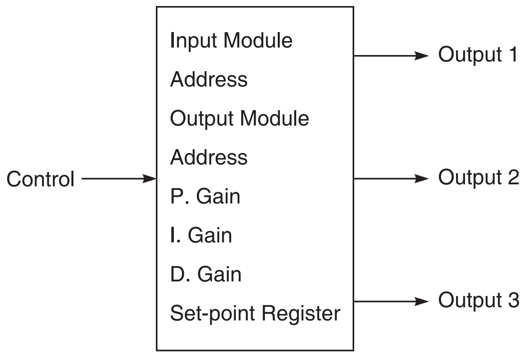


Figure 10.13 PID (Feedback) control loop.

Thus functional block will look like.



The input and output variables are specified through address of the module from where input is sensed and where the output signal terminates. The output signals may get energized in the following pattern:

Output 1 is energized when PID operation is on

Output 2 is energized when low level alarm is on

Output 3 is energized when high level alarm is on.

10.4.5 English Like Statement

Those who are aware of higher level computer languages (BASIC, FORTRAN, PASCAL, C) will find programming in functional blocks, relay ladder diagram or Boolean mnemonics, very cumbersome and time consuming. As an example—

The functional block for subtraction can be simply represented in FORTRAN as:

If ($X - Y$) 30, 40, 50

The program jumps to statement No. 30 (when $X < Y$), 40 (when $X = Y$) or 50 (when $X > Y$) depending on values of X and Y. It is possible to address input and output modules using higher level languages. Thus any kind of function can be easily provided in higher level languages. As an example, the relay ladder diagram of Fig. 10.9 can be represented in BASIC (Fig. 10.14).

10	D	=	A .AND. (.NOT. B)
20	E	=	C .OR. F
30	K	=	(G .AND. H) .OR. (NOT .L)
40	X_1	=	[M .AND. (NOT. N)] .OR. (K .AND. L)
50	X_2	=	(T .AND. U) .OR. (W .AND. X) .OR. (Y .AND. Z)
60	S	=	[X_1 .AND. T .AND. (NOT P)] .OR. (X_2 .AND. V)

Figure 10.14 Basic program for ladder diagram of Fig. 10.7.

A number of higher level English-like languages are available on the pattern of BASIC (e.g. SYBIL), or other English-like languages.

10.5 SOFTWARE

The controller software in general follows the structure shown in Fig. 10.15.

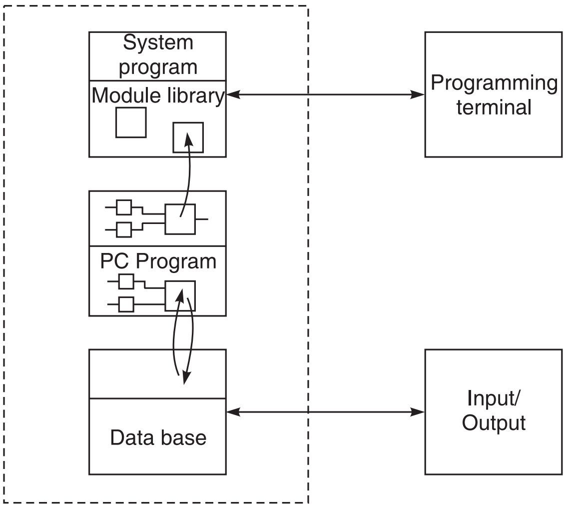


Figure 10.15 Structure of controller software.

10.5.1 System Program

The complete system software in the controller is stored in an EPROM. It consists mainly of the following parts:

1. Operator dialogue for application configuration and commissioning and service functions
2. Software module library
3. Real-time executive which supervises the execution of the application program and performs tests on the system hardware.

10.5.2 Application Program

Controllers normally include EEPROM for storage of the application program, defined by the user. The use of the EEPROM (Electrically Erasable Programmable Read Only Memory) means that the application program is protected even with power supply failure.

The different versions of the controller being manufactured currently contain EEPROM of capacities which permits the use of various application programs of different sizes.

The Read Write Memory is used for the execution of the application program. The capacity of the memory is related to that of the EEPROM.

10.5.3 Communication Program

Programmable controllers can work under the command from the host computer. A number of Programmable Controllers can be connected in multi-drop fashion to a computer. A special part of the system software handles the communication interface of the host computer. Several controllers can be connected to the same communication link (multi-drop). Data can thus be read from and written to the individual controller.

10.6 CONFIGURATION

User can easily configure the programmable controller for performing the tasks required by the application. The configuration is based on a range of basic functions which are programmed permanently as well-defined programmable controller-modules. The functions performed by the programmable controller-modules can be of different types: (a) logical operations, (b) memory functions, (c) selectors, (d) counters, (e) arithmetical operations, (f) regulators, (g) input/output handling etc. The adaptive regulators in the controllers are also accessible in the form of programmable controller-modules.

The configuration of the system consists of linking together of suitable programmable controller-modules to obtain a required function. This is performed by means of a simple dialogue procedure using a programming terminal connected to the system.

The programmable controller-modules are interconnected in a structured manner to form a complete control system. The programmable controller-program is divided into functional units, called blocks, each of these contains a number of programmable controller-modules linked together. The blocks can, in turn, be linked together in a corresponding manner.

Following are the configuration steps to be followed:

(i) *Define the function structure:* Programmable controller modules are selected from the module library and are linked together.

(ii) *Setting* of *parameters:* Modules and signals are numbered. For some modules parameter values are chosen (max, min, channel number etc.). The result is a complete function diagram.

(iii) *Input:* The configuration is done via terminal in a dialogue with the controller. Necessary data is available in the function diagram.

The programmable controller-block is executed as a unit, i.e., all programmable controller-modules included are run through once. One of the following types of condition is applied to execution of the block:

— the block is executed at regular intervals of time;

— the block is executed with a period defined by a pulse train connected to the equipment;

— the block is executed in accordance with a condition written into the programmable controller-program.

The programmable controller-program (application program) which is the result of the configuration dialogue is stored in an EEPROM and parameters which define the functions of the programmable controller-modules and which are specified at the time of configurations are stored in addition to the configuration structure.

The programmable controller-program defined originally can be amended by means of simple commands from the programming terminal. Modules or blocks can be added or deleted and parameters can be changed.

A printer can be connected to the controller for hard copy listing of the programmable controller-program.

10.7 APPLICATIONS

Programmable controllers are used in many industries for one or more of the following functional areas:

1. Sequence control, timing, counting, data calculation.
2. Quick change of machine or process logic, to manufacture different items using the same machine or process equipment.
3. Auto-compilation of production/consumption/downtime/maintenance data.
4. Batch or continuous process control.
5. Open loop or feedback control, process data acquisition and display.
6. Precise motion/position control.
7. Reference generation, drive system computation, control and coordination.
8. Adaptability of control system to computers, colour/monochrome VDUs, data logging printers.
9. Hierarchical control and data acquisition system.

Following is a typical list (not exhaustive) of applications of programmable controller:

Tyre manufacture

Progrmmable controllers are ideal for controlling sequencing of events. For example in the tyre manufacturing process—the events like serving plies in the right order, rotary movement of drums, multiplanary movement of the spindles, and so on finally transform the raw material into the product (tyre).

Tyre curing press

Programmable controllers provide correct sequencing of the curing cycle, time measurement and control in each cycle. They monitor parameters such as temperature, pressure during a curing cycle and annunciate faults to the operator. Programmable controllers can also generate report for each shift with details about good cures, press downtime, etc.

Programmable controllers can also be used for controlling the processes like mixing of the raw rubber with carbon black, oil and other chemical additives, in rubber mixers.

Plastic injection moulding

Variables such as temperature and pressure can be controlled by the programmable controller, thus optimising the injection moulding process. Programmable controllers also control the velocity levels of injection to maintain consistent filling, thus reducing surface defects and shortening cycle time.

Programmable controllers are widely used for controlling the processing of synthetic rubber, manufacturing of traveller cases, plastic toys and so on.

Chemical batching

The batching ratios of various ingredients used in a process can be controlled by programmable controllers. The controllers also determine the rate of discharge of each *ingredient*. Predetermined batch recipes stored in the programmable controllers can be easily selected by the operator depending upon the production schedule.

Ammonia and ethylene processing

Programmable controllers control and monitor compressors which are used in the manufacturing process of ammonia, ethylene and other chemicals. Qualitative parameters like, temperature, power consumption, vibration, pressure, suction flow, etc are measured by the programmable controller.

In several other applications such as emergency/safety shutdown system, Automatic Wall Test (AWT), Pump-on control, water treatment, crude separation, Lease Automatic Custody Transfer Systems (LACTS), oil field control systems etc, programmable controllers are widely used.

Pulp batch blending

A major application area for programmable controllers is sequence control and quantity measurement of ingredients and storage of recipes for the blending process. The programmable controller also allows the operator to change the composition of the ingredients, if required. It also provides hard copy print-outs of batch/shift production, raw material and energy consumption.

Paper mill digester

Programmable controllers provide control of pulp digesters for the process of making pulp from wood chips. They calculate and control the amount of chips, based on density and the digester volume; determine, the quantity of cooking liquors and add the required amounts in sequence. Programmable controllers also control temperature till cooking is completed.

Gypsum board plant

Programmable controllers are used for controlling the three main sections of a gypsum board plant—*furnace section* where the programmable controller precisely controls the furnace temperature for preparation of wet board, *'wet and transfer'* section control for partial drying of the board and cutting to preset length, and finally the '*take off*' section where the board is dried completely in a hot air furnace controlled by the programmable controller.

Besides, programmable controllers are also used for controlling wood yards, paper mill production, energy management, boiler, soot blowers and production of various kinds of laminates.

Material handling, weighing and conveying

Programmable controllers can be used to advantage in weighing predetermined quantities of raw materials, carrying it over a group of conveyors at a predetermined flow rate, and taken to the production centre. In addition, programmable controllers monitor conveyors for faults and annunciate alarms.

Stacker-reclaimer

Programmable controllers control movements of the booms for stacking and reclaiming operation in desired patterns. A single programmable controller can be used to control a group of stacker-reclaimers for co-ordinated movements.

Storage and retrieval

In warehouses, programmable controllers control movements of cranes, for easy storage and retrieval. Inventory control figures are also generated and provided on request.

Plating line

Programmable controllers control movements of the hoist (which can traverse right, left, up and down) through various plating solutions based on a set pattern. The programmable controller keeps track of the location of the hoist at any point of time in the line.

Steel plants extensively use programmable controllers in coal yards, sinter yards, blast furnace charging. Programmable controllers are also used for storage, retrieval and packing of urea in fertilizer plants, coal/ash handling in power plants.

Manufacturing machines/production machines

Programmable controllers perform sequencing and interlocking of high speed production machines. They also generate data regarding the production rate of controlled machines, downtime etc. Programmable controllers with remote I/Os can be used for monitoring and controlling production machines at various locations in a large works complex.

'Cut to length' line

This advanced position control capability makes programmable controllers ideal in precision 'cut to length' lines. In addition, the programmable controller also performs sequencing and interlocking functions.

Tool changing

Programmable controllers can be used for tool changing in large production machines like machining centres. The programmable controller keeps track of as to when the tool is to be changed (based on tine type of machining to be done).

Programmable controllers are also used for machine fault monitoring and diagnostics, transfer lines, robot controls, Flexible Manufacturing Systems (FMS), special purpose machines, etc.

10.8 CONCLUSIONS

The applications of today's controller have gone far beyond simple control functions of its predecessors. Technology has not only made the controllers more capable, but also more affordable. Programmable controllers have become intelligent decision-making machines with a wide scope of applications that range from variable control functions, data acquisition, to report generation and supervisory control.

SUGGESTED READING

Babb, M., Allen-Bradley introduces a new line of PLCs, *Contr. Engg.,* **35,** No. 10, pp. 57–59, 1988.

Babb, M., PLC manufacturers get ready to take on the world, *Contr. Engg.,* **36,** No. 2, pp. 61–65, 1989.

Baron, G.R., Application of programmable controllers in process interlocks and shutdowns, *ISA Joint Spring Symp.,* Houston, Texas, 1983.

Brickley, G.J., New development in programmable controllers and peripherals, *Cont. Engg.,* **34,** No. 1, pp. 76–79, 1987,

——, G.J., Variety of languages offered in PLCs, *Contr. Engg.,* **35,** No. 1, pp. 44–46, 1988.

Flynn, W.R., Fifth annual programmable controller update, *Contr. Engg.,* **35,** No. 1, pp. 51–55, 1988.

Griffiths, C., Programmable controllers push in process control, *C&I*, No. 1, pp. 35–36, 1987.

Johnson, D., *Programmable controllers for Factory Automation*, Marcel Dekker, New York, 1987.

Quatse, T.J., Programmable controllers of the future, *Contr. Engg.,* **33,** No, 1, pp. 59–62, 1986.

Tinham, B., The changing face of programmable controllers, *C&I*, No. 1, pp. 29–32, 1987.

CHAPTER

11

Modeling and Simulation for Plant Automation

11.1 INTRODUCTION

Simulation of technical systems has increased the importance of modeling during the last three decades. This is due to the rapid increase of the system and control theory during this time and especially due to the rapid development of the computer technology. Generally speaking, no plant automation system can now be designed and made operational without using the methods of model building and simulation of the system to be automated. This holds for the technical systems as well as for the non-technical ones.

Modeling or model building has been well known for years among the mechanical, aeronautical, shipping and civil engineers. For optimal designing, small scale models of the objects were built and tested. The models were obviously the physical models of the objects (e.g. of the systems) to be designed, since the development of efficient electronic computers as simulators was either in a rudimentary stage or did not exist at all. As soon as network analyses and later on analog computers were developed, building of physical models of the systems lost its attractiveness since being more expensive, rigid (inflexible) and difficult to use than the analog modeling.

Thus, analog modeling and simulation of systems had become of interest for specialists of numerous branches of science and technology. The analogy consisted of the same mathematical description of the system to be modeled and of the same electronic circuit used as the model. Due to this fact, the application of electronic computers to the simulation of real systems (whose mathematical model has been implemented as an analog computing circuits or as an algorithmic program) has been a common means for engineers and scientists to analyse and design new systems. This especially holds for the development of automation systems which have to be designed optimally and to be tested without disturbing the industrial plant to be automated. This could be done by applying its mathematical model implemented on an analog or a digital computer.

11.2 OVERVIEW OF PROCESS MODELS

Any description of a system could be considered to be a model of that system. Although the ability to encapsulate dynamic information is important, some analysis and design techniques require only steady-state information. Models allow the effects of time and space to be scaled, extraction of properties and hence simplification, to retain only those details that are relevant to the problem. The use of models therefore reduces the need for real experimentation and facilitates the achievement of many different purposes at reduced cost, risk and time.

In term of control requirements, the model must contain information that enables prediction of the consequences of the changing process operating conditions. Within this context, a model could be either a mathematical or a statistical description of specific aspects of the process. It can also be in the form of qualitative descriptions of process behaviour. A non-exhaustive categorisation of model forms is shown in Fig. 11.1. Depending on the task, different model types will be employed.

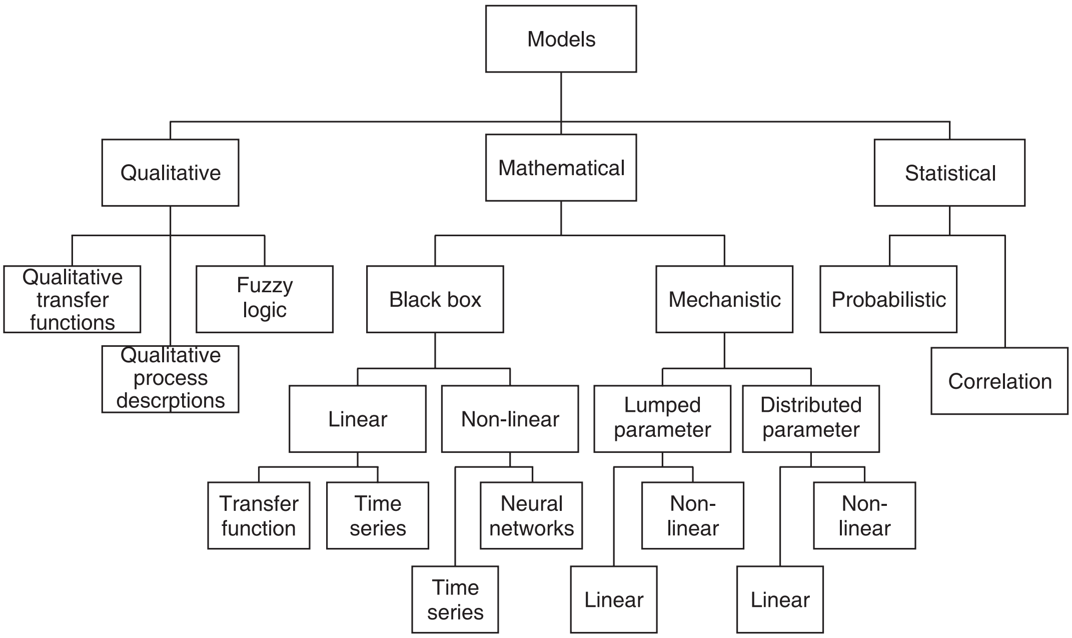


Figure 11.1 Classification of model types for process monitoring and control.

11.2.1 Mechanistic Models

If the process and its characteristics are well-defined, then a set of differential equations can be used to describe its dynamic behaviour. This is known as *mechanistic model development*. The mechanistic model is usually derived from the physics and chemistry governing the process. Depending on the system, the structure of the final model may be either a lumped parameter or a distributed parameter representation. Lumped parameter models are described by ordinary differential equations (ODEs), while distributed parameter systems representations require the use of partial differential equations (PDEs). ODEs are used to describe behaviour in one dimension, normally time, e.g., the level of liquid in a tank. PDE models arise due to dependence also on spatial locations, e.g., the temperature profile of liquid in a tank that is not well-mixed.

Obviously, a distributed parameter model is more complex and hence harder to develop. More importantly, the solution of PDEs is also less straightforward. Nevertheless, a distributed model can be approximated by a series of ODEs given simplifying assumptions. Both lumped and distributed parameter models can be further classified into linear or nonlinear descriptions. Usually nonlinear, the differential equations are often linearised to enable tractable analysis.

In many cases, typically due to financial and time constraints, mechanistic model development may not be practically feasible. This is particularly true when knowledge about the process is initially vague or if the process is so complex that the resulting equations cannot be solved. Under such circumstances, empirical or black-box models may be built using data collected from the plant.

11.2.2 Black Box Models

Black box models simply describe the functional relationships between system inputs and system outputs. They are, by implication, lumped parameter models. The parameters of these functions do not have any physical significance in terms of equivalence to process parameters such as heat or mass transfer coefficients, reaction kinetics, etc. This is the disadvantage of black box models compared to mechanistic models. However, if the aim is to merely represent faithfully some trends in process behaviour, then the black box modelling approach is just as effective. Moreover, the cost of modelling involved is orders of magnitude smaller than that associated with the development of mechanistic models.

As shown in Fig. 11.1, black box models can be further classified into linear and nonlinear forms. In the linear category, transfer function and time series models predominate. With sampled data systems this delineation is, in a sense, arbitrary. The only distinguishing factor is that in time-series models, variables are treated as random variables. In this absence of random effects, the transfer function and time-series models are equivalent. Given the relevant data, a variety of techniques may be used to identify the parameters of linear black box models. The most common techniques used, though, are least squares based algorithms.

Under the nonlinear category, time series features again together with neural network based models. In nonlinear time series, the nonlinear behaviour of the process is modelled by combinations of weighted cross products and powers of the variables used in the representation. The parameters of the functions are still linear and thus facilitate identification using least squares based techniques. Neural networks are not new paradigms to nonlinear systems modelling. However, the increase in cheap computing power and certain powerful theoretical results have led to a resurgence in the use of neural networks in model building (see Section 13.9.)

11.2.3 Qualitative Models

There are instances where the nature of the process may preclude mathematical description, e.g., when the process is operated at distinct operating regions or when physical limits exist.

This results in discontinuities that are not amenable to mathematical descriptions. In this case, qualitative models can be formulated. The simplest form of qualitative model is the ‘rule-based’ model that makes use of the ‘If-Then-Else’ constructs to describe process behaviours. These rules are elicited from human experts (see Section 13.4). Alternatively, genetic algorithms and rule induction techniques can be applied to process data to generate these describing rules. More sophisticated approaches make use of qualitative physics theory and its variants. These latter methods aim to rectify the disadvantages of purely rule-based models by invoking some form of algebra, so that the preciseness of mathematical modelling approaches could be achieved.

Of these, Qualitative Transfer Functions (QTFs) appear to be the most suitable for process monitoring and control applications. QTFs retain many of the qualities of quantitative transfer functions that describe the relationship between an input and an output variable, particularly the ability to embody temporal aspects of process behaviour. The technique was conceived for applications in the process control domain. Cast within an object framework, a model is built up of smaller sub-systems and connected together as in a directed graph. Each node in the graph represents a variable, while the arcs that connect the nodes describe the influence or relationship between the nodes. The overall system behaviour is derived by traversing the graph from input sources to output sinks.

Models derived from the use of Fuzzy Set Theory can also be classified as qualitative models (see Section 13.6). Proposed by Zadeh, fuzzy set theory contains an algebra and a set of linguistics that facilitates descriptions of complex and ill-defined systems. Magnitudes of changes are quantised as ‘negative medium’, ‘positive large’ and so on. The model combines elements of the rule-based and probabilistic approaches and sets of symbols with interpretations such as ‘If the increment of the input is positive large, the possibility of the increment on the output being negative small is 0.8’. Fuzzy models are being used in everyday life without our being aware of their presence, e.g., in washing machines and auto focus cameras.

11.2.4 Statistical Models

Describing processes in statistical terms is another modelling technique. Time-series analysis, which has a heavy statistical bias, may be considered to be in this model category. Nevertheless, due to its widespread and interchangeable use in the development of deterministic as well as stochastic digital control algorithms, the earlier classification is more appropriate. The statistical approach is made necessary by the uncertainties surrounding some process systems. This technique has roots in statistical data analysis, information theory, games theory and the theory of decision systems.

Probabilistic models are characterised by the probability density functions of the variables. The most common is the normal distribution, which provides information about the likelihood of a variable taking on certain values. Multivariate probability density functions can also be formulated, but interpretation becomes difficult when more than two variables are considered. Correlation models arise by quantifying the degree of similarity between two variables by monitoring their variations. This is again quite a commonly used technique, and is implicit when associations between variables are analysed using regression techniques.

System dynamics are not captured by statistical models. However, in modern control practice they play an important role, particularly in assisting in higher level decision making, process monitoring, data analysis, and obviously, in Statistical Process Control.

11.3 MODEL BASED AUTOMATIC CONTROL

The representative model of a process can be used via simulation to answer operational questions such as safety-related issues and to provide for operator training. However, this approach is not suitable for real-time automatic control Within the context of automatic control the inverse problem is considered, i.e., given the current states of the process, what actions should be taken to achieve desired specifications. Depending on the form of the plant model, different control strategies can be developed. The advantage of adopting a model based approach to controller development is illustrated in the block diagram shown in Fig. 11.2.

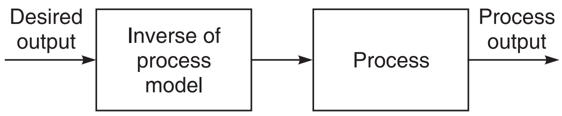


Figure 11.2 Model based controller.

By regarding the blocks to be mathematical operators, it can be seen that if an accurate model of the process is available, and if its inverse exists, then the process dynamics can be cancelled by the inverse model. As a result, the output of the process will always be equal to the desired output. In other words, the model based control design has the potential to provide perfect control. Hence, the first task in the implementation of modern control is to obtain a model of the process to be controlled. However, given that there are constraints on process operations—all models will contain some degree of error and that all models may not be invertible—perfect control is very difficult to realise. These are the issues that modern control techniques aim to address, either directly or indirectly.

In the process industries, black box models are normally used for controller synthesis, because the ill-defined nature of the processes makes mechanistic model development very costly. For process design purposes, precise characterisation is important. However, for the purposes of control strategy specification, controller design and control system analysis, models that can replicate the dynamic trends of the target processes are usually sufficient. Black box models have been found to be suitable in this respect and can be used to predict the results of certain actions.

Linear transfer functions and time series descriptions are popular model forms used in control systems design. This is because of the wealth of knowledge that has been built up in linear systems theory. Increasingly, however, controllers are being designed using nonlinear time series as well as neural network based models in recognition of the nonlinearities that pervade real-world applications.

11.4 DEFINITION OF TERMS

For appropriate usage of the terms *modeling* and *simulation,* we should first know their exact definition which to a greater extent is self-explanatory.

As already mentioned, the modeling of a system implies in fact building of its mathematical model. This consists of writing a set of mathematical equations along with a set of boundary conditions and limitations posed on a system itself. In case that the system to be modeled is a linear dynamic system shown in Fig. 11.3 (as many technical systems—at least approximately are) its mathematical model will be described by an ordinary differential equation of the type.

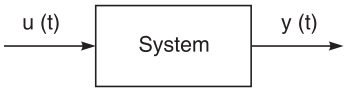


Figure 11.3 Block-diagram of a dynamic system.

$$y(t)^{(n)} + a_n{-}1\ y(t)^{(n-1)} + \ldots + a_1\, y(t) + a_0 = u(t)$$

where,

$y(t)$ = system output
$u(t)$ = system input
a_i = system parameters (i = 0, 1, 2, ..., (n – 1))

Dynamic system whose time-behaviour is described by an ordinary differential equation of the order n is a linear dynamic system of the same order n. It is called time-invariant system if its parameters (i.e., the coefficients a_0, a_1 …, a_{n-1}) are constant; otherwise it is a time variant system. In the modern system theory it is usual to describe the system in the state-space (Fig 11.4) in the form.

$$\mathbf{X}(t) = A(t)\ .\ \mathbf{X}(t) + \mathbf{b(t)}\ u(t)$$

$$\mathbf{Y}(t) = \mathbf{C(t)}\ .\ \mathbf{X}(t)$$

where, **X** is the state vector of the system and *A(t)* its system matrix containing essential system parameter. Further system parameters contain the vector **b(t)** (control vector) and the vector **C(t)** (observability or measurement vector).

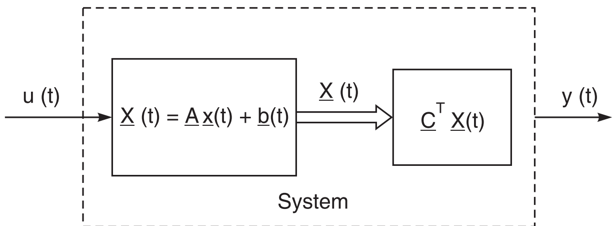


Figure 11.4 State-space systems representation.

Non-linear systems can be described by non-linear differential equations, sampled data systems by the equivalent difference-equations and the system with distributed parameters (like the flow and heat distribution systems, tubular reactors, industrial furnaces etc.) by partial differential equations.

Systems simulation is the implementation of the mathematical system model on an analog or digital computer and its execution. The mathematical model of the system implemented in a computer will also be called the simulation model of the system.

11.5 SYSTEM MODELING

In order to apply a computer system to control a technical process, the description of the system itself, e.g. its dynamic or time behaviour should be stored in the computer so that the control application software can choose the best strategy for the system for each situation or—more precisely—for each state of the system. The fact that the computers *are* able to understand the numerical and logical relations and perform the calculations, implies that the system description stored in the computer has to be in algorithmic and/or in logic terms. Thus, representing the differential equations and the corresponding boundary values and restrictions as the mathematical description is the best way to describe a system for the computer, since we cannot (or we still cannot) work with verbal (or textual) system descriptions.

Once we have stored the mathematical model of the system, any powerful method of modern control theory, namely (a) Calculus of variations, (b) Maximum principle, (c) Dynamic programming, and (d) Linear and non-linear programming can be effectively applied in order to implement the following control strategies:

— Time optimal
— Energy optimal
— Fuel optimal
— Quality optimal control.

Furthermore, based on mathematical model of the system, one of the following advanced control configuration and algorithms can be implemented:

— Dead-beat controller
— Minimum variance controller
— On-off and 3-state controller
— Dead-time controller
— Predictor and extrapolater
— Cascaded, ratio, feed-forward, and anyother complex controller
— Multi-variable controller
— Adaptive and self-tuning controller.

11.6 USES OF SYSTEMS SIMULATION

Based on the mathematical model of the system, stored in the computer, there are many engineering methods that could directly be applied while designing a system or an automation system for it. In the first case, we can use the system simulation technique for optimal system designs. For instance, by simulation of the system for given different values of a parameter the optimal value of this parameter can be found.

Example

As an example the optimal design of a tubular reactor for ammonium synthesis could be done using system simulation. The temperature distribution within the reactor, as shown in Fig. 11.5 (dotted line) has the form of a distributed pulse in which peak value (due to the exothermic reaction) can exceed the maximum allowable temperature for the catalyst or for the reactor walls. In this case the optimal reactor design will be such as to reduce the temperature peak down to the maximum allowable value so that the output of the reactor will still be as high as possible (see the continuous lines in Fig. 11.5).

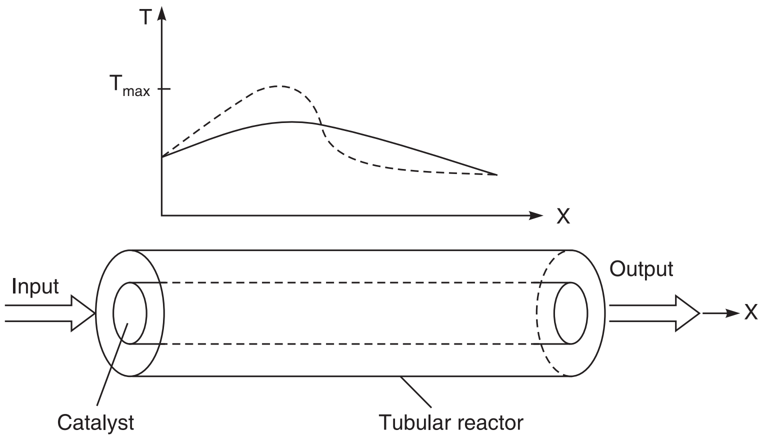


Figure 11.5 Temperature distribution in a tubular reactor.

There are many parameters of the reactor system that could be changed in order to solve the given problem. These are

— Distributed reactor cooling
— Partial feedback of the product to the reactor input
— Distributed density of the catalysts

The first application field of the simulation technique is to develop optimal system design. By storing the mathematical model of the reactor on digital computer, each of these strategies can be tried by developing software programs and executing them. Thus, the reactor need not be disturbed. As an experiment, the model of the reactor was stored in the computer using PEARL language (Process and Experiment Automation Real-Time Language). The last strategy was applied on this model: The concentration of catalysts within first 10% of its total length has been space distributed in different ways in order to find the optimal distribution pattern. Thus, without disturbing the reactor the optimality problem was solved. This optimal distributed pattern was then tried on actual reactor and results were matched with the simulation results. There are many other practical examples which can also demonstrate the powerfulness of the simulation technique for the optimal system design even when several parameters of the system have to be optimised simultaneously.

The second application field of the simulation technique is the design and realisation of the most advanced control strategies by using the mathematical model of the plant in an on-line optimisation procedure. Examples of such control strategies which have already found their multiple applications in nearly all industrial branches are,

— Predictive system control by using a predictive system model
— Multivariable system control by using the state-vector feedback
— Adaptive system control by using the reference system model.

Furthermore, the realisation of some advanced methods of the modern systems theory is exclusively due to the possibility of building of mathematical model of the plant. These methods are

— Systems identification
— Parameters estimation
— State observation
— State estimation.

Systems identification and process parameter estimation will be discussed in Chapter 13.

11.7 HOW TO BUILD THE MATHEMATICAL MODEL OF A PLANT?

Depending on the nature of the plant and the processes that take place in it certain basic knowledge of the subject areas (viz. physical sciences, engineering etc.) and prior information is needed for model building.

Based on this knowledge, the model equations on the various parameters can be written. These parameters include: energy, heat flow, force, momentum, and mass balance within the system as well as, dynamics, kinetics etc. of the processes (or of the reactions within the processes).

However, the model equations written in this way will contain the system parameters represented (to a greater degree) by a general (or symbolic) form only. No numerical values of the system parameters (or only a few of them) will be known to the systems engineer. This implies that most of the parameter values should be determined by calculations based on measuring data collected by experiment on the system itself. Thus, the procedure of model building of a plant contains two essential steps:

— Mathematical model definition by using the basic knowledge in engineering and of natural sciences; and
— Parameter estimation of the plant by experiment.

In the second step, the parameter estimation of the plant, will need the application of some data evaluation methods which in turn will give the exact or the estimated parameter values. The methods available for this purpose can be analytical or statistical. The analytical methods (when no noises are present in the system) are:

— Prony method
— Gradient method
— Method of quasi-linearisation

— Invariant imbedding
— Model reference method or statistical method.

Statistical methods for normal operating conditions include:

— Least squares method (LSM)
— Generalised least squares method (GLSM)
— Instrumental variables method (IVM)
— Maximum likelihood method (MLM)
— Stochastic approximations method (SAM).

11.8 MODEL EVALUATION AND IMPROVEMENT

After developing the mathematical model of the plant and estimating the plant parameters. The model should be evaluated. By this, the comparison of the time-behaviour of the system and the model can be understood. If the system and its model behave nearly in the same way under many different circumstances (for example for different input signals) the model is accepted as a good description of the system and can in each situation be applied as a full substitute for the system itself. In the opposite case, i.e. if the model does not behave like the system, the model is understood as not good enough to represent the system. This means, that some further experiments on the system are necessary or a new parameter estimation method need to be applied. Furthermore, if the model behaviour is essentially different from the system behaviour, in that care even the model structure (e.g., the model equations) will be tentatively rearranged so that eventually the system order will be changed and the estimation of the parameters repeated (Fig. 11.6). Once the proper system

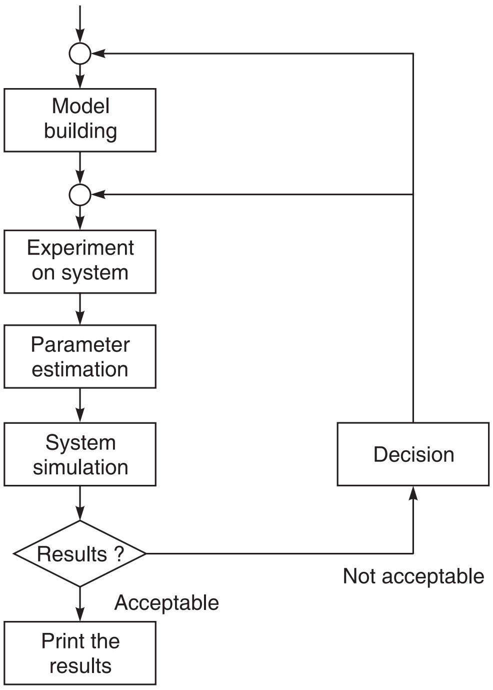


Figure 11.6 General approach to model building.

order is determined and the system parameters evaluated, the system can adequately be replaced by the model and the experiments performed on the model by simulation will give the same results as the equivalent experiments performed on the system.

The final step along this line should be an attempt to simplify the system model as far as possible, for a lower order model that can match the system behaviour within the desired or presented accuracy. This criterion is of primary interest when building adaptive or self-tuning systems.

11.9 MODERN TOOLS FOR MODELING AND SIMULATION OF SYSTEMS

As mentioned earlier, model building and simulation techniques have been applied for more than four decades to solve engineering problems. During this time, not only numerous complex technical problems have been solved but at the same time, a variety of tools have been developed, that make the application of both the techniques easier.

In the area of systems modeling the following changes have been identified:

1. The know-how in engineering has permanently been increased and the “original” theory is generally redefined in terms of the modem systems theory. The same holds for the natural sciences like—Physics, Chemistry, Biology etc. The engineers as well as the scientists have in the meantime mathematically defined their systems. In many cases some catalogues of mathematical models of plants, processes and phenomena have been compiled and edited for engineers and scientists. Only the model parameters should be adopted to the system parameters by experiment.
2. Modem systems theory has been developed as desired for modeling and simulation of the systems. Many useful “old” methods have once more been “re-discovered” and adopted for flexible algorithmic treatment and programming. Original method for “off-line” systems identification and for parameter tracking recursive methods which are easy to program and need less process data to be stored have been re-introduced instead of direct methods, working on complicated data set. Also, the influence of the noise and of other environmental disturbances have been taken into account by new parameter estimation methods developed in the mean time.
3. Besides this, many “universal” software packages have been developed for different computer systems for identification and parameter estimation of industrial plants. The practical use of such software packages is not difficult since these are interactive or CAE/CAD based. Parallel to this, many efforts have been done in this area to create new tools for systems simulation, some of which are mentioned here.

In the area of hardware not only the analog computers have been better adopted for simulation purpose—by interfacing them to the digital computers or by their extension to the hybrid computers but also many special computer systems have been developed exclusively for simulation purposes. Moreover, parallel digital computers have been developed to effectively replace the analog computers as far as the simulation speed is concerned. In addition to this, the interface to the system to be simulated has been improved enabling relatively simple system coupled on-line simulation that could be of interest for predictive systems control.

In the area of software, much has been done in development of new, special programming languages for systems simulation which could replace the general purpose simulation languages like, CSMP and GPSS.

In addition to this, in some application areas completed simulation programs have been developed which are suitable for educating and training the staff operating the plant.

11.10 APPLICATION EXAMPLES

For years, model building and simulation have been successfully applied to nearly all industrial branches and to other human activities. Examples given below represent only a small general outline of a long list which have been modeled and simulated.

— Paper, pulp and rubber;
— Sugar, food and brewing;
— Chemical, petrochemical and fertiliser;
— Steel and iron;
— Cement;
— Power generation and distribution, etc.

In this connection, problems like (a) Optimal system control, (b) Parameter tracking, (c) Adaptive and self-tuning control have been solved. This was done in order to save energy and/or raw material to increase the production volume, to improve the product quality, etc.

In the area of public services (but within big plants) the problems of optimal distribution and scheduling of energy, heat, gas, water resources, have successfully been solved by using the methods of,

— Predictive control
— Kalman filtering
— Linear and non-linear programming.

Besides these applications, model building and systems simulation have for years also been effective tools to solve the problems in other technical areas especially in, Bio-engineering and Bio-medicine. For instance the model building and system simulation in a bio-engineering unit have led to

— Optimal bacterial growth
— Optimal control of fermentation processes
— Optimal drug injection
— Automatic disease diagnostics and other problems.

Finally, the special simulation systems have helped in training of the staff for control of ships, aircrafts, nuclear power and other industrial plants.

11.11 FUTURE PERSPECTIVES

Today, a great number of specialists are involved in research and development in the field of modeling and simulation of dynamic systems. They permanently develop new methods and introduce new tools for this purpose, so that the future trends in this area is definitely promising. It is therefore not surprising to realise that there is no meeting or conference of experts in any field of applied human activities where no papers will be delivered concerning the simulation and model building of the systems specific for that very field.

Finally, there are also many national and international organisations and societies devoted to work in this area. Thus, it is to be expected that much work will be done and published in both model building and simulation of different systems. The future trend is towards the following areas:

1. Development and application of the methods for modeling and parameter estimation of nonlinear; distributed parameter; and complex systems.
2. Standardisation of identification methods and building of a comprehensive library of modular programs implementing the methods standardised. The library will be able to run on the most of the microcomputer systems since the programs are written in high level languages.
3. Building of non-algorithmic models for the processes like, diagnostic and decision making process.

Here flexible expert systems will be developed, for application in a specific field, and will be easily adopted by building the specific database of the system and the specific routes for its diagnosis. This part of the systems is already in use in the field of engineering and medicine.

11.12 CONCLUSIONS

Modelling and simulation tools have been in use from a very long time, but they are too inefficient and inadequate to deal with the advanced and complex nature of the modern control systems, since these tools are totally non-graphical in nature. Because of the limitation of graphical tools, design engineers used to heavily rely on traditional text-based programming and mathematical models. And this used to be a major cause of concern, since developing models in text-based programs wasn't just difficult and time-consuming but also highly prone to errors. Also, debugging the model and correcting the errors used to be a tedious process. It required a lot of trial and error cases before a final fault-free model could be created, as mathematical models used to undergo unseen changes during the transportation of the model through the various design stages.

These challenges are overcome by the use of graphical modelling tools. The modern day graphical tools cover all aspects of design. These tools are a very generic and unite graphical modelling environment; they reduce the complexity of model designs by breaking them into hierarchies of individual design blocks. Due to this, designers achieve multiple levels of model fidelity by simply substituting one block element with another. Graphical models are also the best way of documenting the engineer's ideas. It helps engineers to conceptualise the entire system in a much better way and simplifies the process of transporting the model from one stage to the other in the design process. Boeing's simulator EASY5 was among the first modelling tools to be provided with a graphical user interface.

The methods of modeling and simulation discussed here have not only been applied by some specialists in this field but these have also been used by a large number of engineers and scientists as an useful tool for their work. For meaningful application effectively learning of these medhods is necessary. Consequently, due to the fast changing needs, it is to be expected that in future the specialists dealing with the systems automation will look at the methods discussed here, in the same way as they look at the methods of signal and systems analysis or of systems control today.

SUGGESTED READING

Astrom, K.J., Theory and application of adaptive control—A survey, *Automatics,* **19,** No. 5, pp. 471–486, 1983.

Gopal, M., *Modern Control System Theory,* Wiley Eastern Ltd., New Delhi, 1984.

IEEE, Challenge to control: A collective view, *IEEE Trans. on Autom. Contr.,* AC–**32,** No. 4, pp. 275–285, 1987.

Kuo, B.C., *Automatic Control Systems,* Prentice Hall, Englewood Cliffs, New Jersey, 1982.

Landau, I.D., *Adaptive Control—The model reference approach,* Marcel Dekker, New York, 1979.

Ogata, K., *Discrete-Time Control Systems,* Prentice Hall, Englewood Cliffs, New Jersey, 1987.

Pradeep, B.D. and Ash, H.R., Elements of computer process control with advanced control applications, *Instrumentation Society of America,* Research Triangle Park, North Carolina, 1981.

CHAPTER

12

Industrial Control Applications

12.1 INTRODUCTION

A review of basics of automation to advanced control strategies has been completed. How do these apply to the industries and utilities around us, is the next question. Though volumes have been written and inexhaustible volumes can further be written for variety of industrial processes, we shall discuss here some representative major industries to demonstrate the applications of SCADA systems, programmable controllers and distributed control systems. Following important sectors have been considered in this chapter.

— Cement
— Power
— Water Treatment Plant
— Irrigation Canal
— Steel.

While describing the applications of automation in these industries, we shall describe the process, very briefly. It is presumed that knowledge of process is already available or has been acquired before going through any of these applications.

12.2 CEMENT PLANT

In Cement Plants, different units, viz. raw material handling, crushing, premixing, and blending preheater/precalciner, kiln, cooler, grinding, storage and packing section are situated throughout a span of approximately 2 km. As such, a geographically distributed control system is normally envisaged in a cement plant. There are nearly 200–250 parameters in a typical cement plant which must be continuously monitored at a centralised control room. The plant can be divided in following groups:

— Crusher Section
— Raw Mill Section
— Kiln and Coal Mill Section
— Cement Mill Section
— Packing Plant Section.

The main mechanical machines, which fall within the purview of the automation system, are:

— Ball Type Raw Mill
— Kiln with Double Preheater Streams and Precalciner
— Grate Type Cooler (having 3 grates)
— Coal Mill
— Open Circuit Cement Mill and Cement Storage.

12.2.1 Objectives of Automation System

The objectives of automation system for cement plant are:

1. Continuous operation of plant and machinery
2. Preparation of uniform raw mix
3. Stabilised kiln operation and minimum upsets of process dynamics
4. Maximised production
5. Minimised fuel consumption
6. Uniform quality of clinker/cement
7. Optimised and safe start-up of the plant
8. Adequate plant safety
9. Fault monitoring and analysis
10. Minimum down-time
11. Optimal control and performance monitoring
12. Automatic print-out of crucial parameters and consumption data for plant management and reliable records.

Above objectives are valid for any large size cement plant.

12.2.2 Automation Strategy

We shall now describe the automation strategy for important individual units in cement plant.

Raw mill automation

The first step in raw mill section is *blending* and *proportioning system.*

The DDC has been widely used to control the blending system in Cement Plants. There are two major advantages in raw mill blending by using direct digital control techniques. First, the blending program has the flexibility for the selection of a number of materials to be blended. The plants operate with a fixed number of feed materials most of the time. However, various types of cement are made which may require different types of raw materials, so it is advantageous for the blending program to be able to control the various situations. Second, the blending program can maintain control continuity during periodic *X-ray analyser standardisation* using an averaging technique.

Figure 12.1 shows the schematic diagram of Blending System Automation using DDC. The function of DDC and X-ray analyser in blending process is displayed in Fig. 12.2.

The prerequisite for uniform quality cement is the preparation of the raw material of uniform composition. The raw material during grinding is constantly subjected to X-ray analysis for determining chemical constituents such as calcium, silica, aluminium, iron and magnesia. The analysis is fed to DDC which controls set-points for electronic weigh feeders for uniform composition. The analyser system is having automatic sampling system. This system improves the quality of homogeneous raw material and lowers the production cost by reducing *burning zone temperature*; and at the same time limiting free lime content in the clinker, and as such, the life of kiln lining will increase considerably thus reducing kiln shut down time.

In automating a grinding mill, the following points are to be considered.

1. Grinding process should operate by itself over a long period of time.
2. Material level with grinding mill should match the hardness of the raw material.
3. Material of uniformly ground grade should be obtained.

The various process parameters used for measurement and control in case of ball mills are:

— Fresh feed rate
— Coarse return
— Bucket elevator loading
— Mill inlet and outlet pressures
— Mill level
— Speed of air separator drive
— Water spray to mill.

These parameters are used for measurement and control of mill, which depend on the control procedure, machinery and plant conditions, to achieve optimum operation of mill unit.

All the equipments in the raw mill section can be started sequentially by Programmable Controllers. The required operation can be programmed as ladder diagram for starting, stopping and sequential control of raw mill section.

Kiln automation

The heart of any cement plant is the kiln. It has to be operated properly for good quality of clinker as well as smooth operation of plant. Kiln is used for converting raw meal to clinker with desired properties. Hence kiln should be controlled for optimised kiln output, minimum fuel consumption and improved kiln lining life.

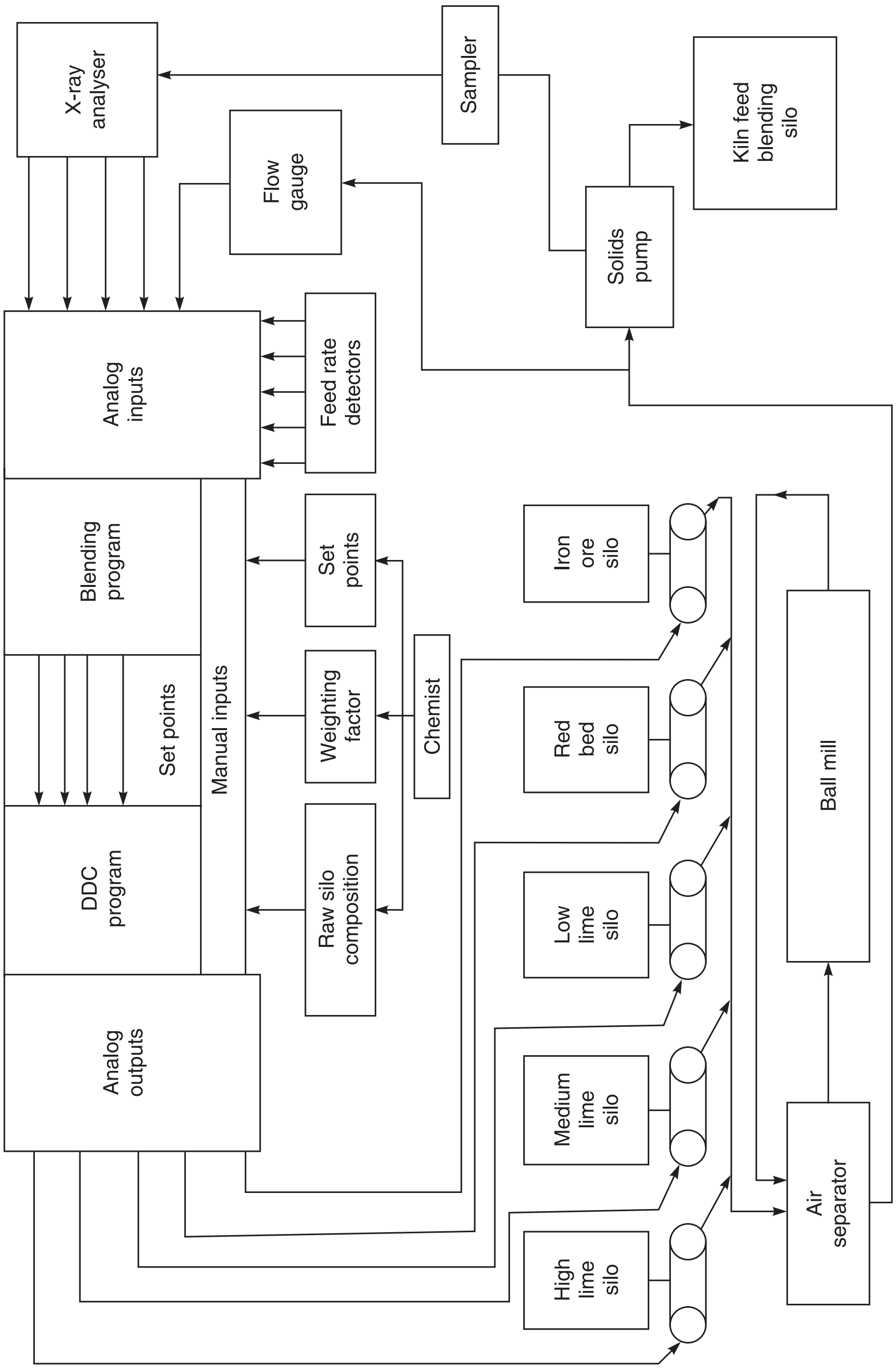


Figure 12.1 Blending system control.

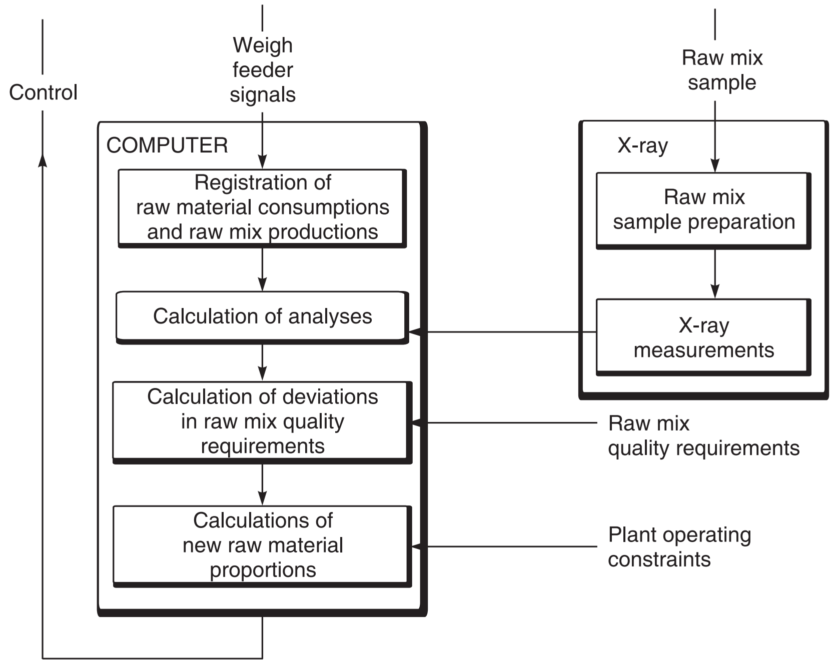


Figure 12.2 Function of X-ray analyser and DDC in blending system control.

The objective of optimal control of kiln operations are to achieve optimum throughput, optimal fuel consumption, optimal clinker quality and minimum production cost.

Following input parameters are used for optimised control of rotary kiln:

— Oxygen and CO contents
— Temperature of the burning zone
— ID fan speed
— Hood drought
— Exit gas temperature
— Primary air temperature
— Secondary air temperature.

On the basis of input data, computer will control the following parameters:

— Kiln speed
— Fuel feed rate
— Raw meal intake
— Grate stoke per minute time in case of grate cooler
— Gas flow rate.

In the distributed hierarchical control environment, the computer calculates the set-point and passes these to DDCs that control the kiln operation. Small variations in the set-points are controlled automatically by the computer. In the case of large disturbances causing kiln upset, the kiln operator has to intervene and control the kiln operation until it stabilises.

DDC can be used to control the temperature of cooler discharge and to improve the cooler efficiency by keeping the temperature of secondary air constant.

Burning zone temperature is usually measured by,

— Partial radiation pyrometer
— Total radiation pyrometer
— Optical pyrometer
— Two-colour radiation pyrometer.

Readings of first three types are affected by kiln environment while the readings of the last type of pyrometer are quite accurate, as the effect of intervening media is eliminated when the temperature is computed by comparing two radiation intensities at different wavelengths. With the help of two-colour radiation pyrometer, DDC can control fuel feed rate and keep burning zone temperature within desired limits. Kiln speed is also monitored and controlled by DDC and thereby clinker burning is controlled (Fig. 12.3).

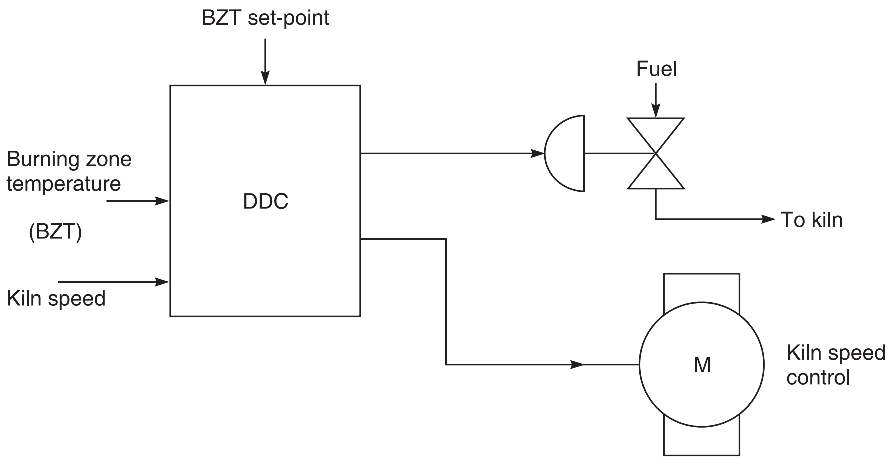


Figure 12.3 Clinker burning control—A schematic.

In addition to pyrometers to measure burning zone temperature, CCTV Cameras are also provided for kiln operator to observe the burning zone flame in the kiln. This enables the kiln operator to take decisions about kiln operation in case of manual override.

For stabilised control of kiln, back-end control is necessary. The control is done by monitoring the following parameters:

— O_2 and CO in exit gas
— Back-end drought
— Exit gas temperature.

Instruments measuring per cent content of Oxygen and carbon monoxide can be suitably interfaced with computer system which in turn controls the ID fan motor speed to keep oxygen and carbon monoxide within specified limits. The exit gas temperature is sensed by thermocouple. It is also interfaced with computer to vary ID fan speed (Fig. 12.4).

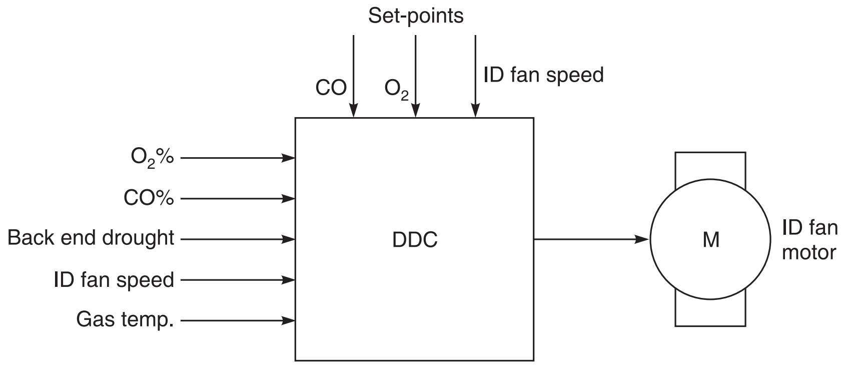


Figure 12.4 Kiln control schematic.

Raw meal feed-rate is controlled by weighfeeders and computer controls the weighfeeders on the basis of throughput of the kiln.

In the clinker cooling section, DDC controls the flow of secondary air and desired temperature of the secondary air entering the kiln (Fig. 12.5). This helps in cooling the clinker fast enough to prevent any damage to the material handling equipment.

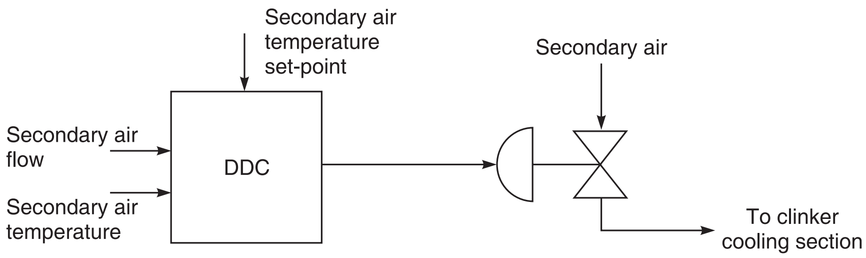


Figure 12.5 Clinker cooling control schematic.

Latest technique in kiln control in large size cement plant is done by fuzzy logic control systems. Kiln exhibits time varying non-linear behaviour and relatively few measurements are available. Consequently, automatic control is usually restricted to a few simple control loops on secondary variables, leaving the responsibility for the control of primary variables to the kiln operator. The usual approach, while seeking a higher degree of automatic control of the kiln operation has been to establish a mathematical model of the kiln process. However, experience indicates that mathematical models either become too simple to be of any practical value or too complex. The application of fuzzy logic techniques to control is tied together with concept of linguistic control rules.

We shall now describe one of the fuzzy control based system developed and used in cement kiln control. RAMPAX-KC used in Osaka Cement Co. Ltd. (Japan) is based on fuzzy control theory and optimal control theory and strong points of each theory are utilised. RAMPAX-KC manipulates the following variables:

— Kiln fuel
— Kiln speed
— Calciner fuel
— Cooler speed
— Oxygen content
— Cooler fan speed.

The RAMPAX-KC retrofits with existing process control systems in a cement plant. The RAMPAX-KC consists of following basic modules:

(i) *Input/output unit.* The RAMPAX-KC receives input signals from existing terminal block board. The selection of output signal is possible through mode switch.

(ii) *Software.* Following software modules function under operating system of RAMPAX-KC.
— Input/Output control module
— Control algorithm module
— Man-machine module.

(iii) *Control module.* The control module which is heart of RAMPAX-KC system contains number of sub modules shown in Fig. 12.6. The sequence of activities:
— Manager observes the state of kiln
— Supervisor commands the controller to · Start-up · Shut · Down · Optimise · Control the kiln through Fuzzy Controller and/or Optimal Controller.

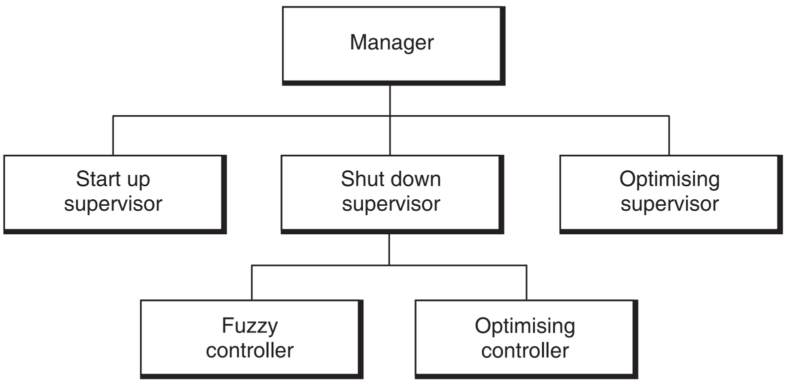


Figure 12.6 RAMPAX-KC control module structure.

Benefits

(i) *Short recovery time from disturbance.* Figure 12.7 shows the difference between human operations and operations using Fuzzy Controller (RAMPAX-KC). The Fuzzy Controller manipulates the kiln like an expert leading to shortened recovery time by one-half to two-third.

(ii) *Optimisation at steady state.* The relation between free lime and burning zone temperature is shown in Fig. 12.8. It is therefore possible to increase the free lime setting point without reducing clinker quality. Thus using optimal control it is possible to eliminate unnecessary fuel wastage and lower the free lime fluctuation.

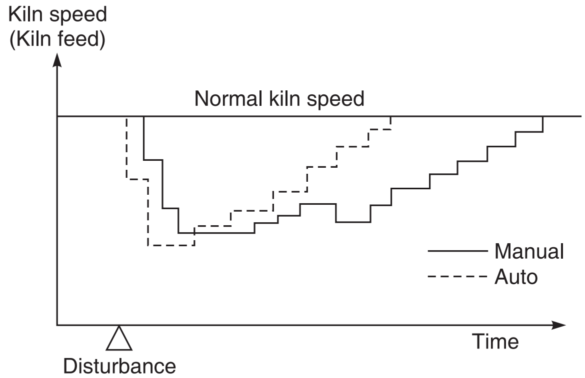


Figure 12.7 Disturbance recovery time using fuzzy controller.

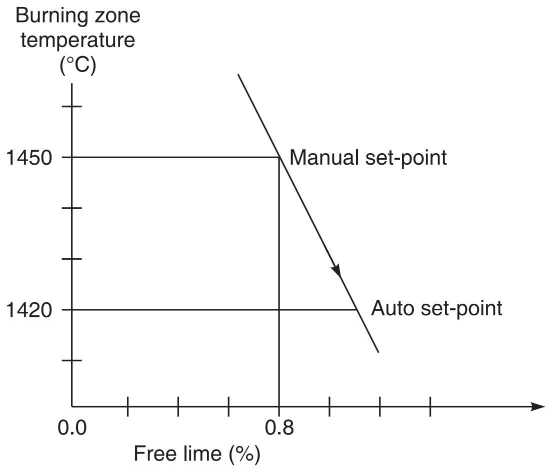


Figure 12.8 Optimum kiln control using fuzzy controller.

The RAMPAX-KC when used in 3000 ton/day kiln has resulted in annual saving of 2500 tons of coal.

Packing and dispatch automation

In packing and dispatch section, automatic bag filling to certain weight and automatic loading in trucks and wagons can be incorporated. The type and weight of cement to be filled up is selected on a keyboard. The PLC may be used to process the data and program the conveyer sequence from the respective silo and transmit the selected weight to control the truck or wagon filling. The filled weight will be logged in sequentially for reference.

12.2.3 Distributed Control System for Cement Plant—A Case Study

As a typical example of Distributed Digital Control in cement plants, let us cite an example of Jordan Plant at Rasadiya, Jordan.

The entire cement producing process from raw meal grinding department to finish grinding department has been controlled and monitored from the Central Control Room (CCR) which is situated in the Control Building near Burner Floor. The following departments have their own Local Control Room (LCR) respectively:

— Limestone crushing and additive crush-in (LCR-1)
— Gypsum crushing (LCR-2)
— Cement loading and packing (LCR-3).

Central control room (CCR)

The Central Control Room located in the Control Building contains the following:

(i) *Central control desk.* For the operation and monitoring of the various areas of the plant, a control console with a small inclined vertical panel for accommodating a few vital indicators and visual display units have been installed. Operation and monitoring of the production process have been carried out using colour graphic displays. Colour monitors and functional keyboards provide the operator with a good overview of the process and help him to take all the decisions and actions for starting, stopping of drives and putting set-point values, etc.

Operator Control Desk has been equipped with VDU Terminal with Track ball. One VDU Terminal is dedicated for Raw Grinding operation, another VDU Terminal is dedicated for Kiln operation and the third VDU Terminal is for Cement Mill operation. Another VDU Terminal is dedicated for Alarm Logging and Monitoring.

Operator Control Desk has been envisaged to provide centralised control and monitoring of the entire cement plant and has the following operation facilities:

1. All closed loop auto control system to be operated via VDU Terminal with track ball.
2. Programmable logic control and sequence interlock.
3. Alphanumeric display for process parameters.
4. Mimic diagram of the plant using colour graphics on VDU.
5. Trend display on colour VDU.
6. Control of electronic weigh feeders for raw mix proportioning with the help of on-line X-ray spectrometer.
7. Optimisation of the major plant equipment and auxiliaries.

(ii) *Central graphic panel.* A mosaic tile graphic panel with active mimic diagram has been provided with the flow diagrams for the two raw mills, the four cement mills and the two rotary kilns. This is equipped with luminescent diodes to signal the operating condition of the plant and additionally with push buttons for the "remote mode" condition. The "running" lamps and "Remote Start/Stop" switches are built into the mimic diagrams.

(iii) *C.C.T.V. monitors.* Six C.C.T.V. monitors are mounted at the upper part of the central graphic panel; two each for Raw Material Reclaim Area, Kiln Burning Zone, and Clinker Coolers respectively.

One Engineer's console has been envisaged for editing and updating programs for PLC and DDC system. The Engineer's console consists of one VDU Terminal.

Central units of Distributed Digital Control System (DAS, DDC and PLC) have been installed in Control Equipment Room. The central units are connected with the remote I/O—interface Unit installed in various electrical rooms through the serial signal transfer cables.

Local control room

In local control rooms, the control desks for operation and monitoring of that part of the plant have been installed. These control desks consist of two parts, (a) a vertical instrument part for mosaic flow diagram, and (b) a nearly horizontal operating part with necessary push buttons, meters and a writing area.

System architecture

The system is based on microprocessor based Distributed Digital Control, geographically distributed throughout the plant for economy and as per the latest trends in process plant instrumentation.

Figure 12.9 shows the system architecture used which consists of the following major parts:

— Process interface—Acquisition part
— Control system—Open loop-sequence control
— Man-machine interface.

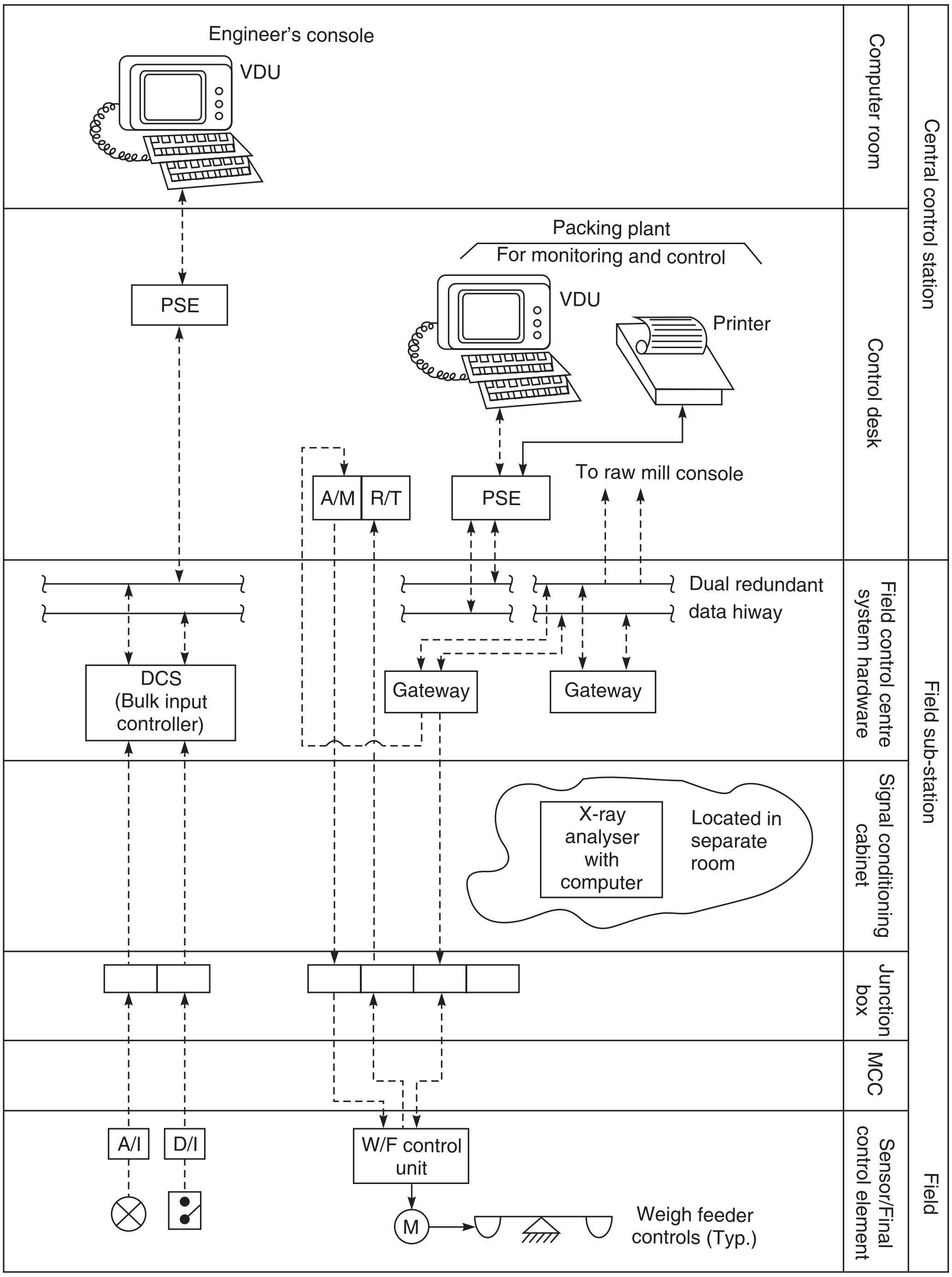
(i) *Process interface.* The process signal parameters are acquired with the help of multiplexers after the raw signals from the discrete field instruments are conditioned. The acquired data are then digitised. The digitised signals are further signal processed and validity of the data are checked. All these functions are carried out in geographically distributed cabinet RTU (Remote Terminal Unit) from which these data are communicated to the main system in binary form with the help of redundant bus.

(ii) *Control system.* The next part is the control and monitoring part. This section is responsible for carrying out the miscellaneous control algorithms. Both interlock sequence functions as well as modulating functional logics are carried out at these places. These control blocks exchange I/O over bus, to the RTUs. The control functions are achieved in two-tier mode:

— SGC-Sub—Group Control
— Drive Level Control Module.

Major logic of the group/sub-group etc. are performed at Sub-Group Control (SGC) whereas individual drive level logics are performed at Drive Control Module (DCM). At DCM back up relay logics are also implemented. The drives connected to MCC/DCM are interfaced at DCM level.

(iii) *Man-machine interface.* The last stage of the system is the Man-machine interface. Like control blocks, each of the MMI is connected to the system bus directly. Dedicated VDUs are assigned to different systems (Fig. 12.10). For MIS and alarm system, one dedicated VDU Terminal has been assigned. Normally, each of the VDU station and printer would be dedicated



PSE — Peripheral Support Electronics
A/I — Analog Inputs, D/I = Digital Input
A/M — Auto Manual Station, R/T — Remote Terminal

Figure 12.9 Control system for Jordan cement plant—System block architecture.

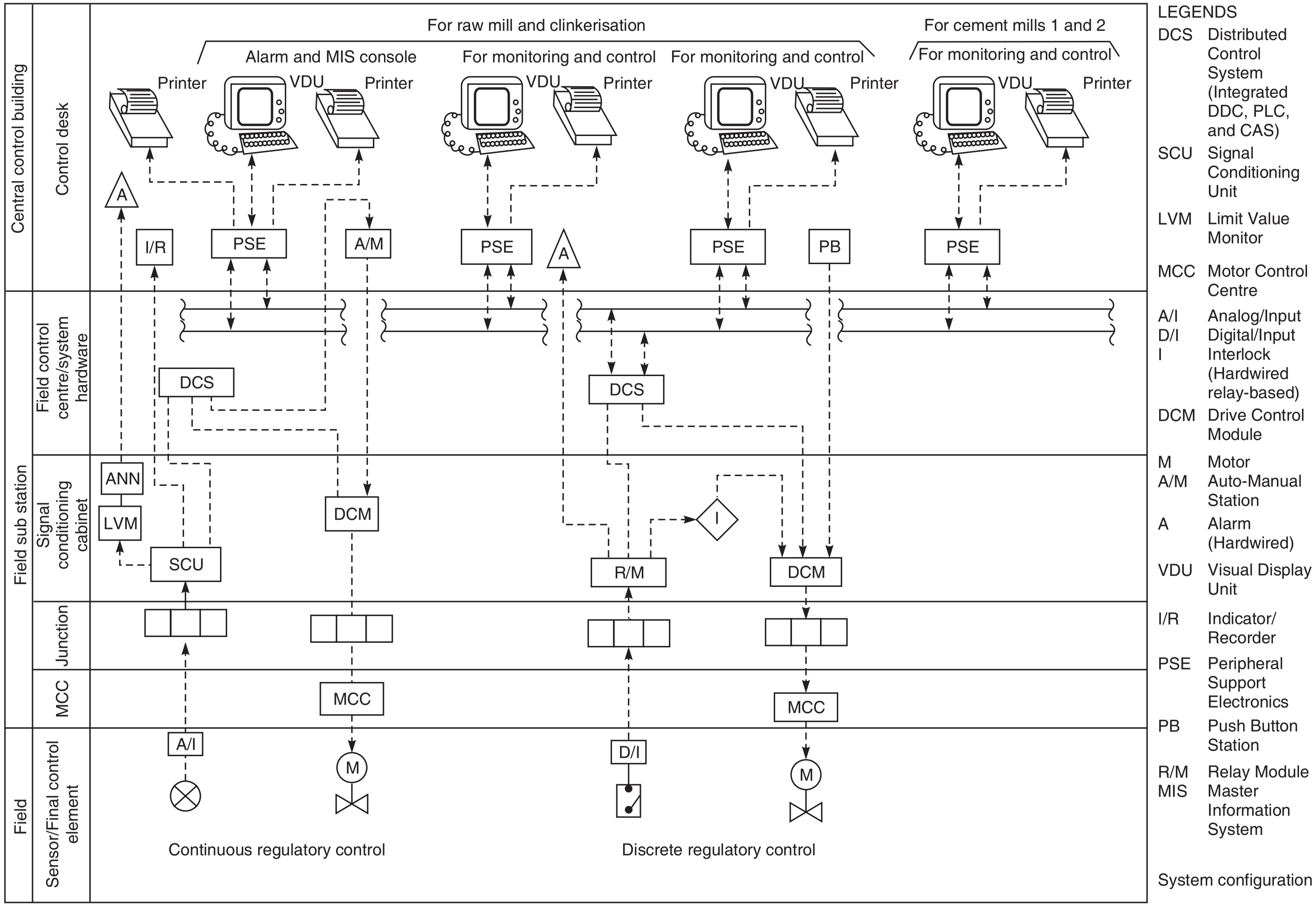


Figure 12.10 Control system for Jordan cement plant—System configuration.

for each section of the plant, but upon failure of the VDU, fall back feature had been provided with manual intervention. In addition to keyboard operation for control loop hardware, auto-manual stations are also provided.

The complete system configuration is shown in Fig. 12.10.

12.3 THERMAL POWER PLANT

Compared with other industrial applications, a thermal power plant due to high reliability requirements, has generally been conservative in adopting new control philosophy or hardware. With advancements made in the latest generation of Distributed Control System (DCS), the apprehensions towards the application of DCS have been proved rather baseless. It is also established that and complexities of control, analysis of data and optimisation can be better handled by such a tool only, provided it is applied with the right kind of application software.

We shall discuss a practical configuration of distributed control system for thermal power plants based on current experience in the industry.

In a Thermal Power Plant, certain number of inputs are manipulated to control certain other variables. The major input variables are,

— Fuel
— Combustion air
— Feed-water flow
— Spray-water flow
— Steam flow (Turbine).

The major control variables are,

— Electrical Megawatt (Output)
— Throttle steam pressure
— Super heated steam temperature
— O_2 in the fuel gas
— Furnace draft
— Drum level.

It may be possible to view the thermal power plant as a de-coupled process between the water side and the steam side in which the boiler drum acts as temporary energy fly-wheel to meet transient load demands.

However, the thermodynamics of the thermal power plant process and examination of the control loops reveal that in reality the process and the loops are highly interactive, both through signal interaction and through the process itself. Thus even if we tend to believe that major control loops are really independent; in essence they are not. A disturbance in one of them is transmitted via the process to affect the others.

Apart from the regulating controls of the plant, there are binary controls for running the auxiliaries and the overriding needs of plant protection based on analog variables crossing the operating limits. The thermal power plant being the source of basic energy input to the industry an outage or damage of critical equipment will have far reaching impact on the economy.

12.3.1 Automation Strategy

Figure 12.11 shows the block diagram of thermal power plant automation system. The basic features are:

1. Distributed Hierarchical System Structure which involves
   — complex functions to be executed in main computer along with MIS;
   — reliable and flexible DDC systems for unit control;
   — programmable controllers for sequence control;
   — independent diagnostic function and protection.

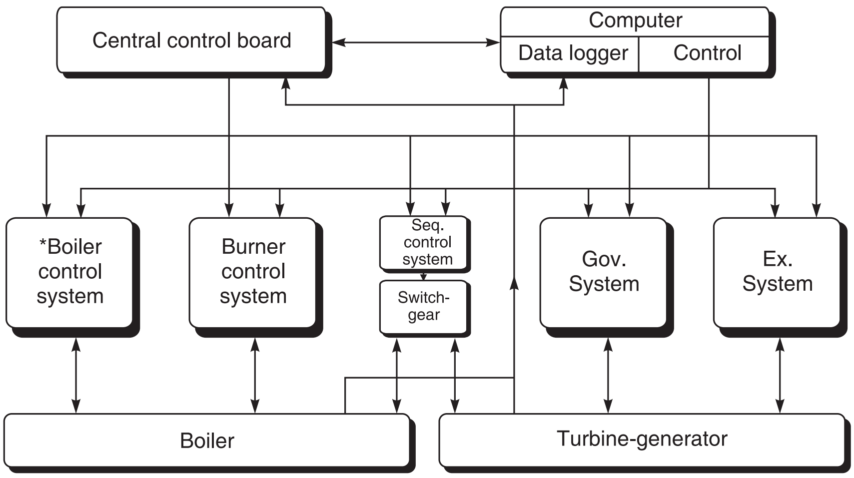


Figure 12.11 Power plant automation systems—Block schematic.

2. Software for power plant automation and other application functions like plant logging, trip review, performance calculation, plant diagnostics etc.
3. Man-Machine Interface.
4. Communication.

12.3.2 Distributed System Structure

Before deciding the structure of distributed control system for power plants, following must be considered.

1. Considering the complexity and inner activity of a power plant, it can be concluded that single loop integrity in the conventional sense in fact, does not improve the availability of the plant as may be thought of. Though it is extremely important to restrict the failure to a part of the plant, it does not necessarily prompt single loop approach.
2. To minimise the effect of failure of the control system on the plant, it is important to functionally distribute the control loops including binary controls. As mentioned above, that, in order to improve the system availability the critical processors should be made redundant/fault-tolerant, but in finalizing the same a judicious choice should be made.

3. It is important to note that in any thermal power plant redundancies are built in at the main auxiliary level. Further, the failure of an important auxiliary like a fan, a pump etc. may lead to the reduction of load but not to an unit trip. Thus it is very important to consider the structural redundancy of the plant. The design of a system architecture should strike an optimal balance between increased reliability and cost effectiveness.
4. The functional distribution of the binary controls should also be made in relation to regulatory control loops. The binary controls of the main auxiliaries, related to regulatory control loops should be performed in the same multi-loop controller. This would enable easy and fast interaction between the two. However, the extent to which this can be realised depends on the power of the CPU of multi-loop controllers and shall vary for different systems.

The functions of different control equipments have been proposed considering the above points. These are shown in Fig. 12.12 along with software modules to perform these functions.

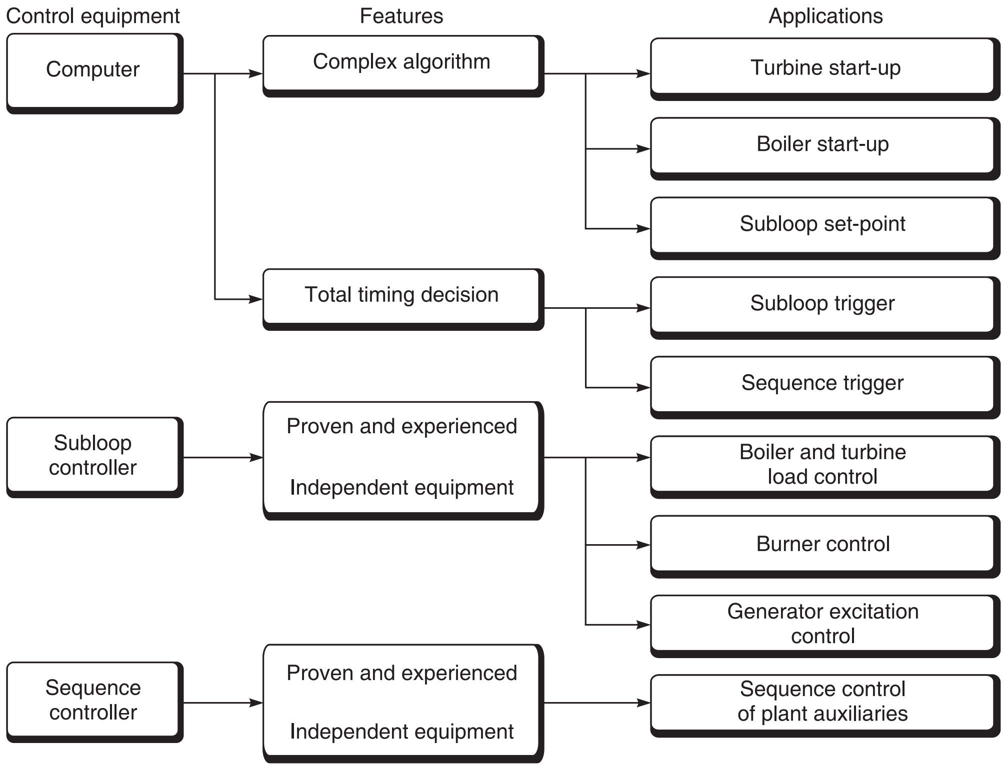


Figure 12.12 Control equipment and applications in power plant automation.

We shall now describe very briefly the following sub-units of Distributed Control System for power plants.

— Unit Control System for Automatic Boiler Control
— Diagnostic Function and Protection

— Unit Control System for Turbine Governor
— Unit Control System for Turbine Automatic Start up System.

Automatic boiler control

The schematic of DDC systems for automatic boiler control is shown in Fig. 12.13. The fuel control, air control and main control can be implemented on microcontrollers connected by I/O bus. Sub-controllers are provided for each major sub-control loop and a main controller supervises each sub-controller. The back-up control of main controller can also be provided for better reliability. Manual/Auto change over stations are shown with each actuator and manual over-ride control is possible in case of emergency.

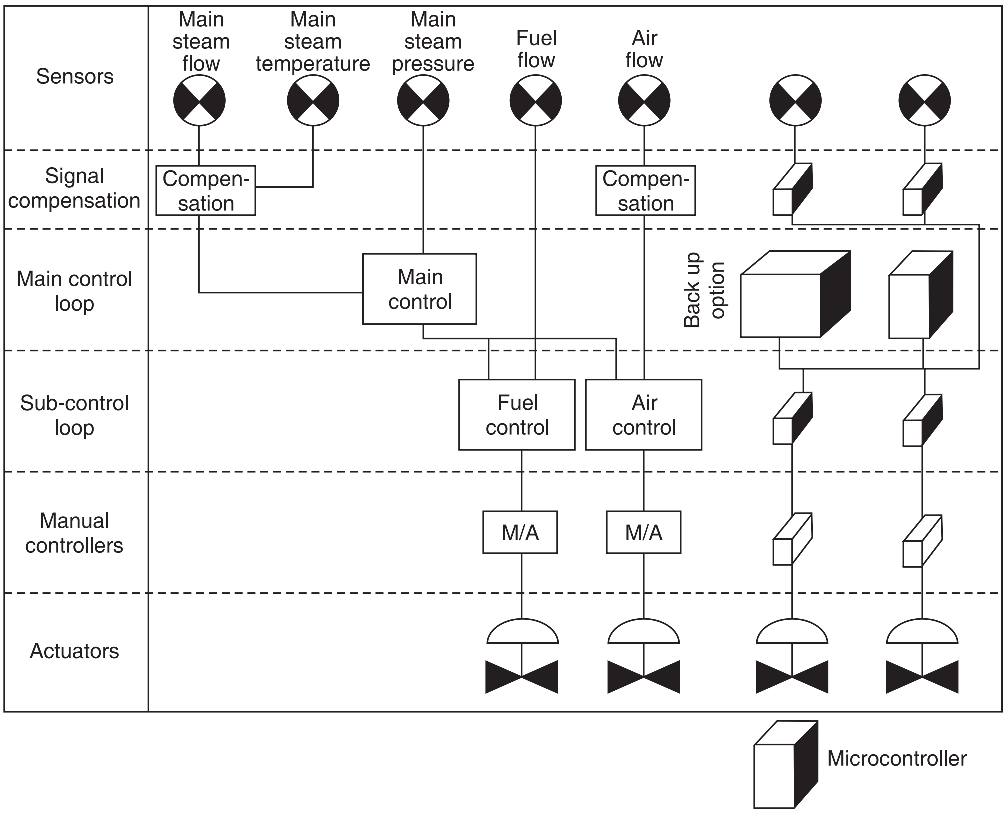


Figure 12.13 Automatic boiler control—Block diagram.

Automatic Burner Control is very important function of Boiler Control. It performs many functions such as furnace purge, sequential control, emergency trip and continuous monitoring of burner. Following are the main functions performed by automatic burner control system:

— Automatic shut down of igniters and burners on operator or computer command.
— Continuous monitoring of burners and igniters.

In addition, there are number of important functions of boiler protection like ‘all flame out’ and ‘critical flame out’ conditions of burner, open/close of air register. Since the limit switches and final actuators are located in severe environment, the electronic control system must be designed especially to eliminate the problem of noise and poor contacts. Generally, all input signals are taken to DDC through magnetic relays and also all output signals are transmitted to actuators in the form of relay contacts.

Diagnostic function and protection

Figure 12.14 shows the system configuration of diagnostic function and protection.

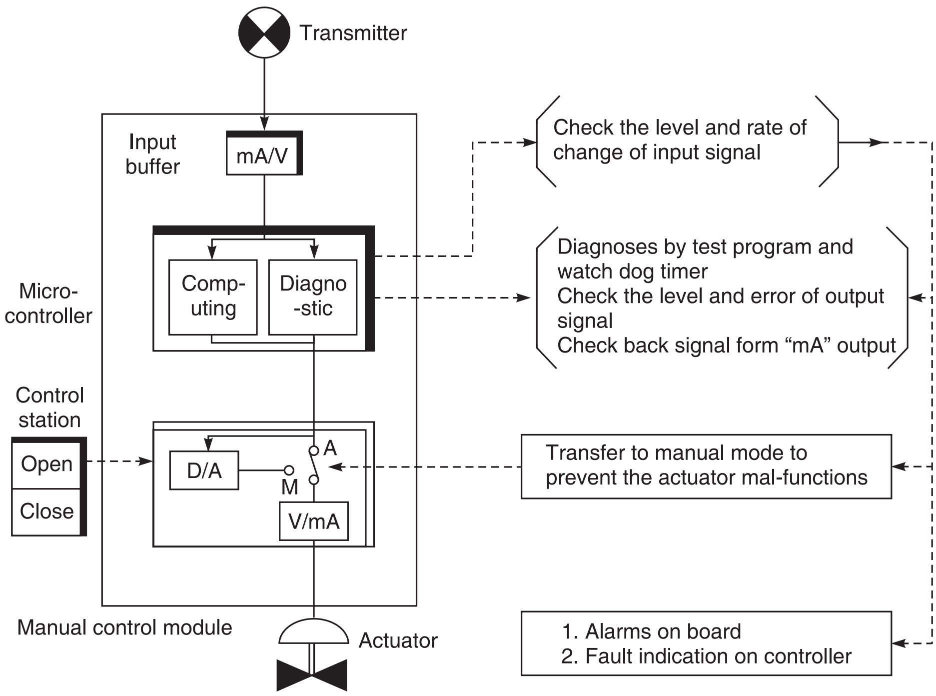


Figure 12.14 Diagnostic function and protection—Block diagram.

The circuit shown may be part of each logic card. The microcontroller checks the level and rate of change of input signal. It also checks the level and error of output signal. In case malfunctioning is diagnosed, the module is transferred to manual mode to prevent malfunctions of the actuator. At the same time, alarm is generated on main system and fault indication on module is put on.

Digital electro hydraulic governor

The Digital Governor performs speed governing run up and load control by controlling the position of the Main Stop Valve (MSV), Steam Control Valve (SCV), and Intercept Valve

(ICV). Main input signals are: speed, valve position, main steam pressure, first stage inner shell (steam) pressure, re-heat bowl steam pressure, condenser vacuum and generator output.

Figure 12.15 shows the function of Digital Electro Hydraulic Governor and its electrical and mechanical interfaces. Table 12.1 outlines the basic functions of governor. These functions are executed by software in Digital Electro Hydraulic Governor.

Turbine speed calculation is one of the most important functions of the Governor. A scheme for speed calculation is shown in Fig. 12.16. This scheme has been used by Hitachi in their Hitachi Turbine Automatic Start-up System. This method is a clever application of digital technology. The main features of this scheme are:

1. Turbine speed can be counted accurately in digital form over a wide range from 0 to rated speed.
2. Three detectors are provided. The signal where the mutual difference between these three signals is smallest is taken as the true value by CPU logic operation.
3. Reasonability check is performed. The turbine which is subjected to inertia, cannot show abrupt speed change. When there is an abrupt change the detector or process input/output devices concerned are judged as faulty.

In the circuit given in Fig. 12.16 two counters, namely, low speed counter and high speed counter are provided to determine the speed of turbine over a wide range from 0 to rated speed. When the turbine is at low speed, the delay in detection for a second or so is not significant. On the other hand as only a small amount of pulse informations is obtained, pulses input during long period of one second is counted on the low speed counter. At high speed, detection delay must be small. There are many pulse signals, so that through the period of 318 input pulses corresponding to the turbine speed, the high speed counter counts the number of pulses from a 250 kHz oscillator. If the count is denoted by NH then turbine speed is proportional to 1/NH

Valve actuation is another very important function of Digital Electro-Hydraulic Governor. Figure 12.17 shows valve actuating system of Hitachi Digital Electro Hydraulic Governor System. The valve actuator consists of Hydraulic Cylinder Servo Valve, Solenoid and Dump Valve for trip. High pressure operating oil from the valve actuator is sent to the hydraulic cylinder via shut off valve. In case of control valves, such as steam control valves, however, oil is sent via the electro-hydraulic servo valve which changes the valve position analog signal from the digital hydraulic governor controller to a hydraulic signal. In the case of stop valve, oil is sent via solenoid valve for testing. At load dump and trip, each main steam valve is made to discharge the oil in the hydraulic cylinder rapidly for preventing, the excess speed, of turbine. So a dump valve for trip is provided to shorten the valve closing time. By performing a valve closing test occasionally, dump valve action is confirmed.

Since Governor Systems are very important part of turbine, reliability through redundancy feature is provided in the controller by duplicating the controller and providing two identical controllers performing exactly the same functions. Only one controller is entitled to issue the control command. In case of failure of one of the controllers, the other controller takes over the operation without any loss of information, i.e. bumpless switching is performed. Figure 12.18 shows the hardware structure of Hitachi Digital Electro Hydraulic Governor having A and B duplex system containing the digital controllers and the common section.

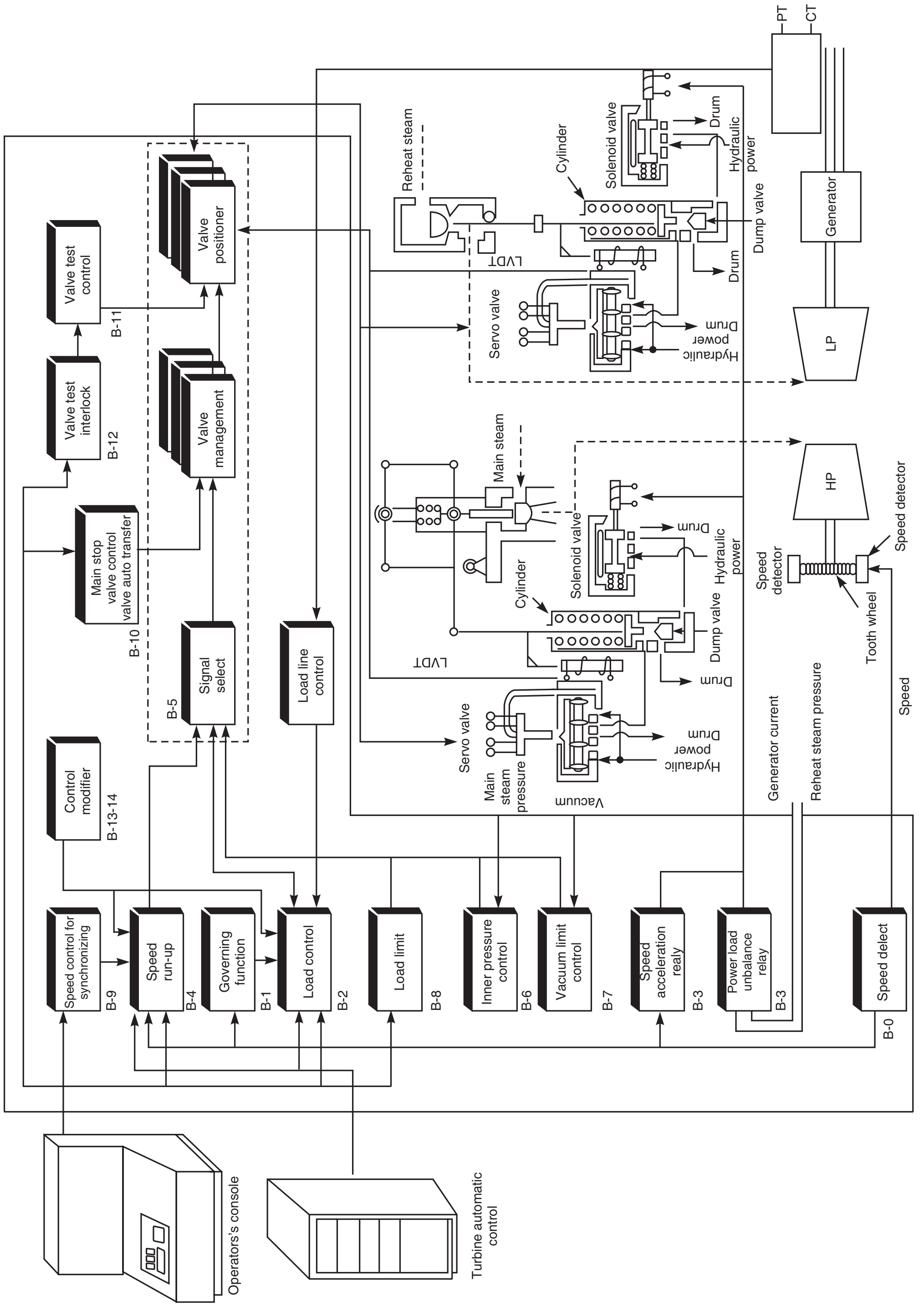


Figure 12.15 Digital electro hydraulic governor—functions [*Courtesy*—Hitachi, Japan].

Table 12.1 Basic Functions of Digital Electro Hydraulic Governor

Function No.	*Function*	*Description*
B-1	Governing function	To control the actual turbine output (steam flow) on the basis of the deviation between the rated speed and actual speed and of the specified speed regulation rate. This is the most fundamental function of a turbine governor.
B-2	Load control	To control turbine output, with the target load and load rate given from outside (e.g., through switch operation from operator's console).
B-3	Power load unbalance acceleration relay	To prevent turbine speed rise by rapidly closing the stop and control valves, when the load of the turbine generator is interrupted abruptly.
B-4	Run-up control	To run up the null speed to rated speed of the turbine.
B-5	Accurate load control	To control the turbine generator output to the specified value by negative feedback of the output so that main steam conditions (pressure and temperature) may not cause turbine output deviation.
B-6	Initial pressure limit	To limit the turbine generator output when the main steam pressure drops.
B-7	Vacuum limit	To limit the turbine generator output when condenser vacuum drops.
B-8	Load limiting function	To limit turbine generator load, independently of governing function (B-1) and load control (B-2).
B-9	Synchronizing function	Fine regulation of speed when turbine generator is synchronized with power system.
B-10	MSV-CV valve transfer function	To automatically transfer bypass and steam control valves when output of turbine of the main stop valve-by pass valve automatic start-up system is about 20%.
B-11	Valve test function	To close valves at a speed suitable for open/close test on main stop valve, steam control valve, and reheat stop valve, when interlock conditions for valve test are set outside DEHG board (e.g. auxiliary relay board) and when closed contacts are given.
B-12	Valve test interlock	Valve test functions including sequence function such as interlock to prevent simultaneous closing operation of two or more valves, when main stop, steam control, and reheat stop valves are tested.
B-13	Run-up control under abnormal turbine conditions	Automatic control function to run back or run up turbine speed to safe band under the condition of vibration generation in a turbine critical speed band, when turbine is run up (B-4).
B-14	Load run-back in the event of abnormality in equipment related to turbine	To lower turbine output in the event of abnormality in turbine-related equipment, e.g., when cooling water supply to generator stator is shut off.

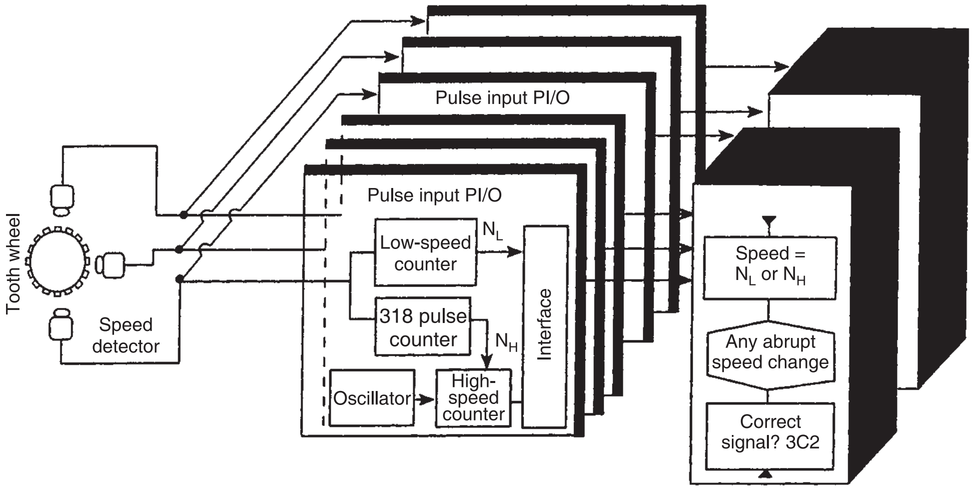


Figure 12.16 Turbine speed calculation [*Courtesy*—Hitachi, Japan].

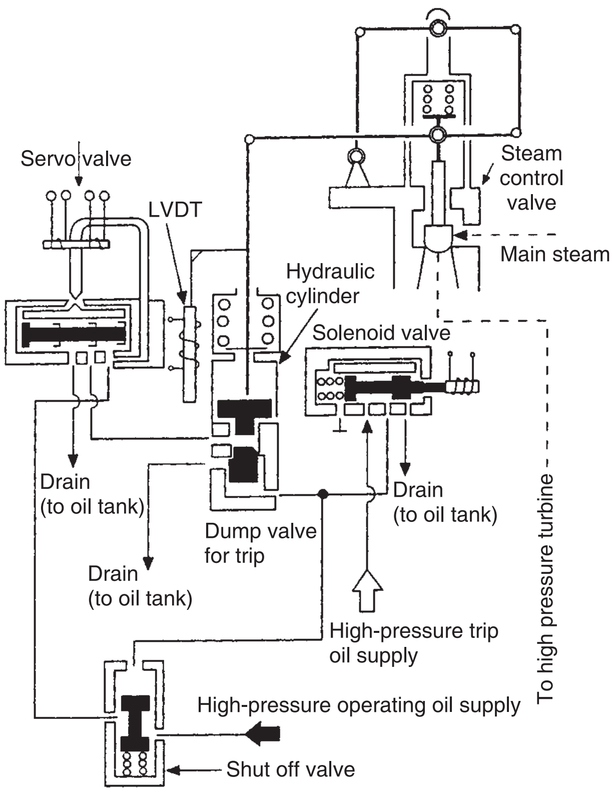


Figure 12.17 Valve actuation [*Courtesy*—Hitachi, Japan].

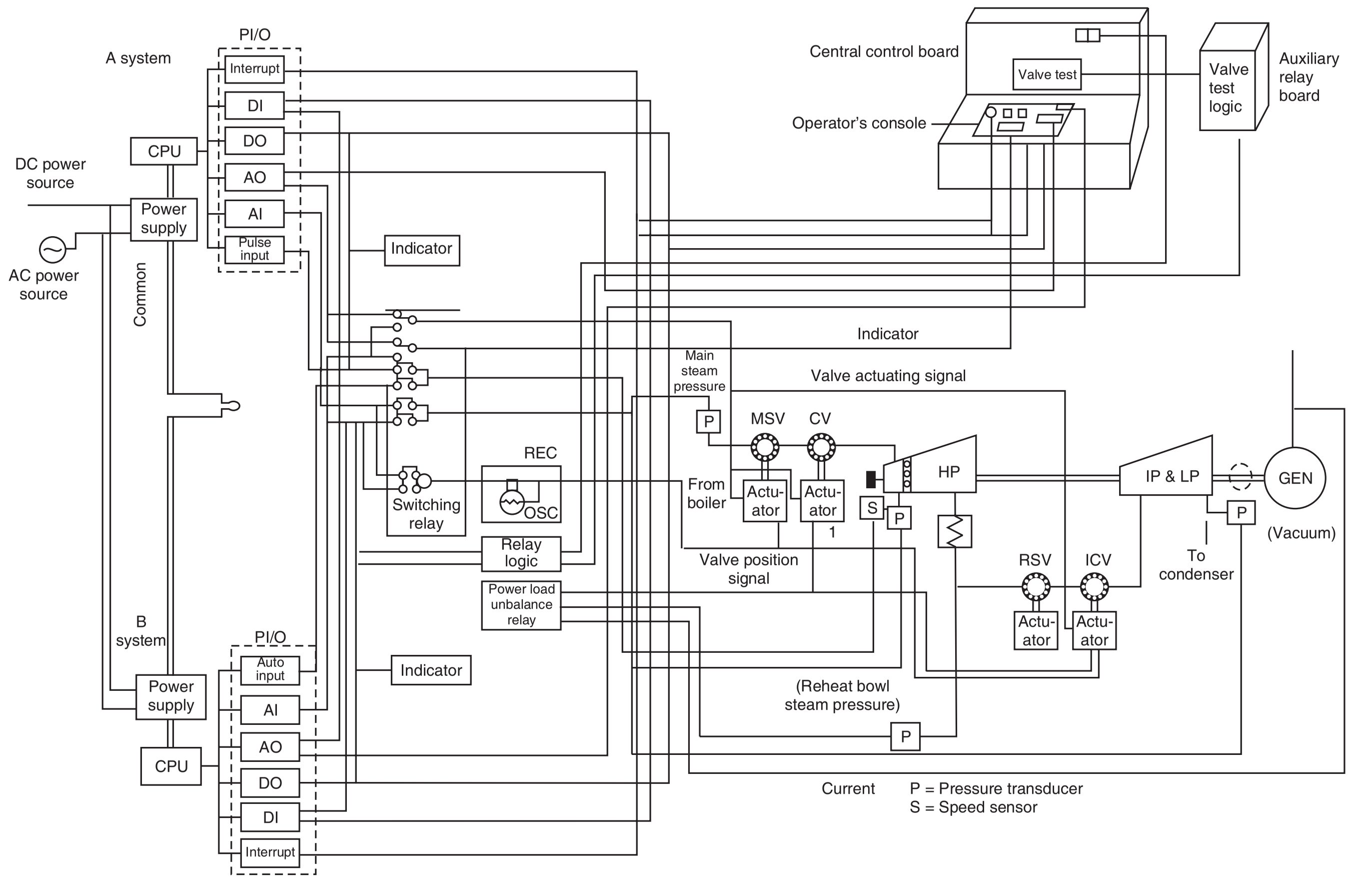


Figure 12.18 Duplicate controllers in digital electro hydraulic governor [*Courtesy*—Hitachi, Japan].

Automatic start-up system

With the increase in number of nuclear power stations, thermal power stations are being used increasingly for middle load operation, since they are frequently started up and shut-down. Also, as unit capacity enlarges, higher reliability comes into demand. To meet these requirements, the control system of thermal power stations is being digitalised. The control system consists of a unit computer for comprehensive control and digital controllers for sub-loop control.

Figure 12.19 shows the configuration of Turbine Automatic Start-up System. The microcontroller calculates the actual turbine rotor thermal stresses by heat transfer calculation around the rotor. The thermal stresses are most important factors to decide turbine start-up procedure. By using advanced control techniques, the turbine speed or load or both are controlled so that thermal stresses do not exceed allowable limits.

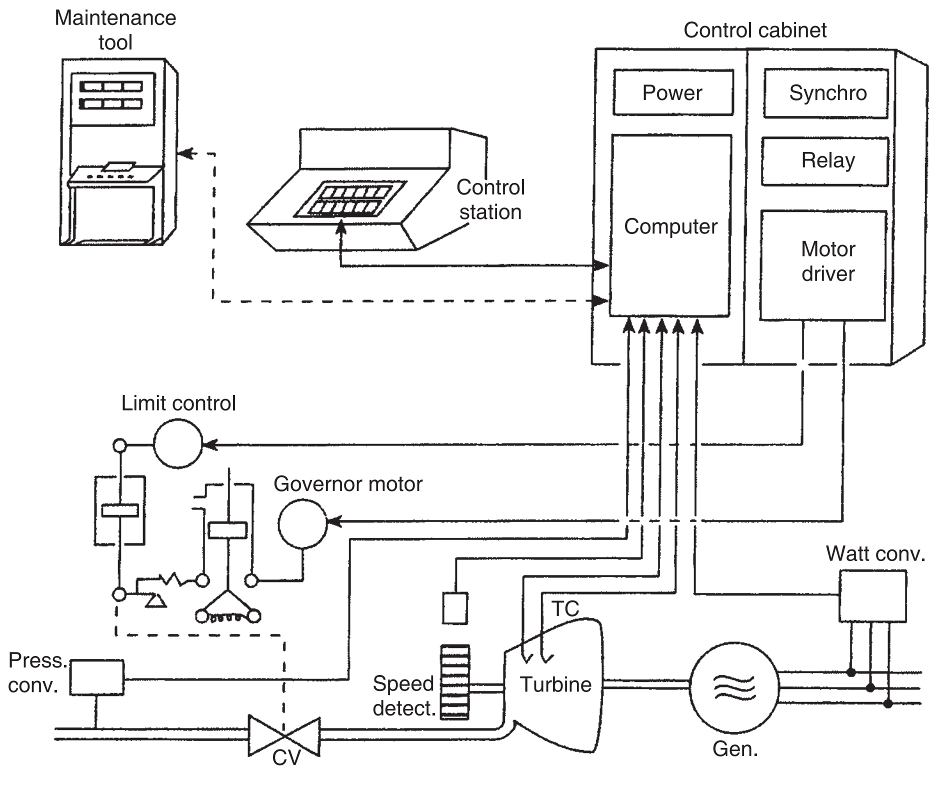


Figure 12.19 Turbine automatic start-up system.

The turbine start-up process is divided into following seven steps.

— Turbine preparation
— Rub check
— Acceleration
— Excitation

— Synchronization initial load
— Target load
— Normal operation.

In *turbine preparation,* the plant status is monitored, to check and ensure that all conditions necessary for turbine start-up are satisfied. In the *rub check* step, the turbine speed is regulated to 200–300 rpm by switching the load limit motor on and off. The operator makes sure that there is no rubbing, then proceeds to the next step.

In the *acceleration* step, speed is increased up to the rated speed at the speed rate selected in the stress control. In the *excitation* step, the excitation system is put to use. In the *synchronization initial load* step, the governor motor is driven to swing the turbine speed centering on the rated speed. In the meantime, the synchronizing relay outputs a voltage balance signal to the excitation system and after checking that the generator and power line have the same voltage and phase, sends a closing signal to the Main Circuit Breaker (MCB). When the MCB is closed, initial load is taken.

The *target load* step is a process in which load is raised to the target load set by the operator. The loading rate is controlled to the value selected in thermal stress control. The normal operation step is for load control, following completion of start-up. The generator output is controlled in order to follow the target load set by the operator. Figure 12.20 describes these steps as well as control functions.

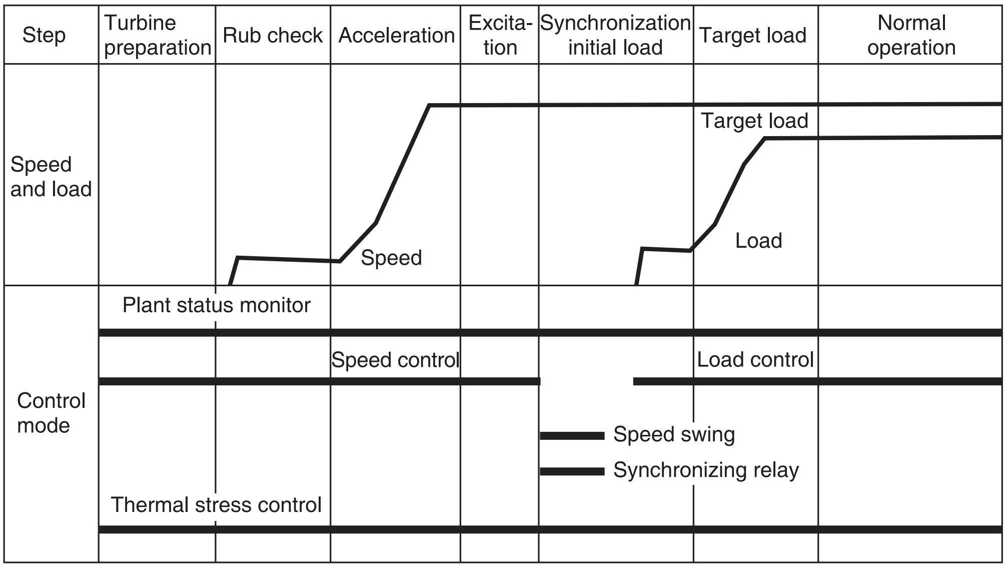


Figure 12.20 Turbine start-up steps.

Thermal stress control

The fatigue due to the thermal stress generated in the rotor during turbine start-up and load changing must be suppressed below the allowable level. Conventionally, schedules for acceleration rate, hold time of speed, etc. have been determined on the basis of temperature mismatch between turbine metal and steam supply before turbine roll. In this conventional method, start-up time tends to be long, because a large stress margin is taken in anticipation of a deviation of the startup pattern of steam condition. The recently developed thermal stress control is designed to calculate stress predictively and optimise speed rate and load rate.

Figure 12.21(a) shows the control flow of thermal stress. First, plant data are read. In case of first execution, the initial value of thermal distribution is set on the basis of measured turbine casing temperature. In “calculation of present thermal stress”, stress is sought by the present value of steam condition. The changing value of steam condition against turbine operation is studied and used for predictive control. For predictive control, calculation is performed by assuming a high value rate and lowering in succession. First, on the basis of the above mentioned study, it is predicted how the steam condition will change when operation is performed at the assumed rate. Then predictive stress is obtained from the predictive value of steam condition. If the predictive stress up to certain time ahead is within the allowable value, then assumed rate is indicated to the speed control and load control blocks. If the predictive stress is in excess of the allowable value, then predictive calculation is repeated by assuming a rate one rank lower.

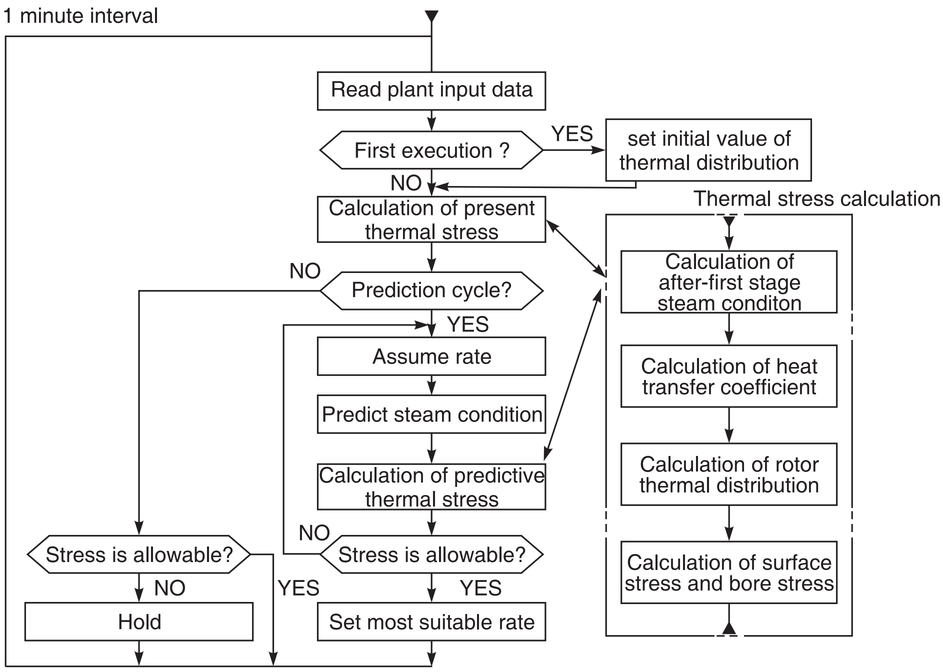


Figure 12.21(a) Thermal stress calculation flow diagram.

In thermal stress calculation, stress is obtained by calculating the “after-first-stage steam condition” and “heat transfer coefficient”, and seeking the rotor “thermal distribution”. Figure 12.21(b) shows the calculation method for the “after-first-stage steam condition”. Since measurements of the "after-first-stage steam condition" are inaccurate in the acceleration step and other steps where there is little steam flow, the after-first-stage steam condition in this method is obtained at high precision by calculating it from the main steam pressure, main steam temperature, turbine speed, and load.

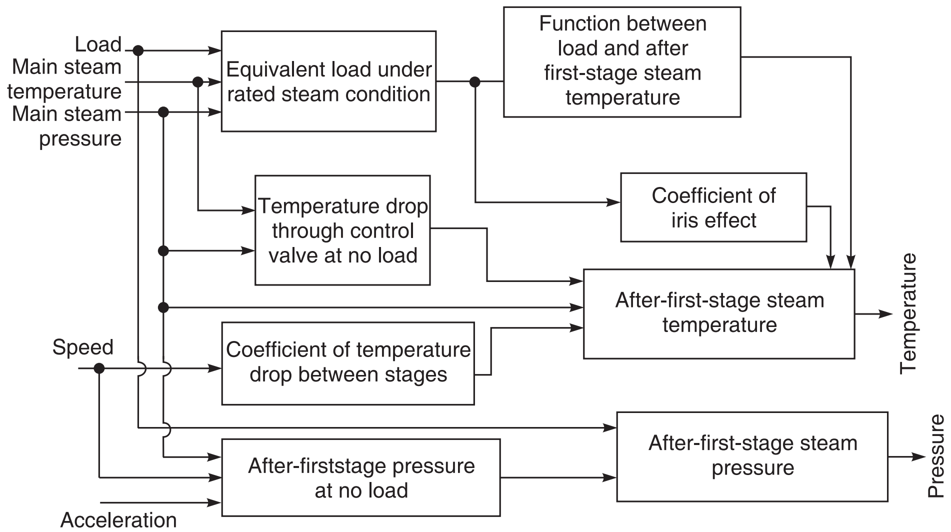


Figure 12.21(b) Calculation of "after-first-stage steam temperature and pressure".

In calculating rotor thermal distribution, a partial differential formula is solved by assuming that the rotor is cylindrical and that there is no heat flow other than that in the radial direction.

12.3.3 Man-Machine Interface

For DCS system, the most important criteria is the operator interface, as this is the window to the system utilisation.

To ensure reliability and availability, the CRT and the console electronics/processor should have one to one correspondence. The software structure of the system should be such that functions of any CRT along with the processor supporting it should be totally interchangeable with any other CRT and its processor. To offer this feature, the system should be based on distributed global database. The human interface software should be supported by a powerful graphic package. The display structure should be totally free format, to accept any orientation and display of data on the graphic screens according to the user's choice.

The system may allow operator interaction via pointing device (track ball). The system operations are prompted by menu bars, pull down menus, soft buttons and sensitised screen areas. The benefit of such technique lies in the fact that no keyboard is required to operate the system.

The screen movement for the different displays has to be designed with index and also via sensitised screen areas to move to the related displays. Moreover, overlay screens with face-plate, trend and important group parameters also are required. The above features ensure fast and easy access concentrating on a particular area.

The different types of displays that can be used are:

- Overview display
- Mimic graphic display
- Group parameter display
- Real-time and historical trend
- Face-plate display
- Alarm summary and history
- Logs and report display
- Deviation monitoring display
- Performance display
- X-Y plot.

Figure 12.22 shows a typical graphic display of automated power plant.

12.3.4 Software System

The software system is one of the key components for successful power plant automation. The proven and versatile software system is needed for establishing high performance and reliable automation system.

In an automatic thermal power plant, following on-line functions need to be performed by software

1. Complex algorithms
   - Turbine start-up
   - Boiler start-up
   - Sub-loop set-point calculation.
2. Total timing decision
   - Sub-loop trigger
   - Sequence trigger.
3. Direct digital control
   - Boiler and turbine load control
   - Burner control
   - Generator excitation control.
4. Sequence control of plant auxiliaries.

The features of the complete on-line functions and control equipment where these software modules may reside have already been described in Fig. 12.12.

Following are the lower level functions performed by each software module mentioned above:

- Graphic display of plant status on VDU
- Plant data acquisition and alarming
- Data logging.

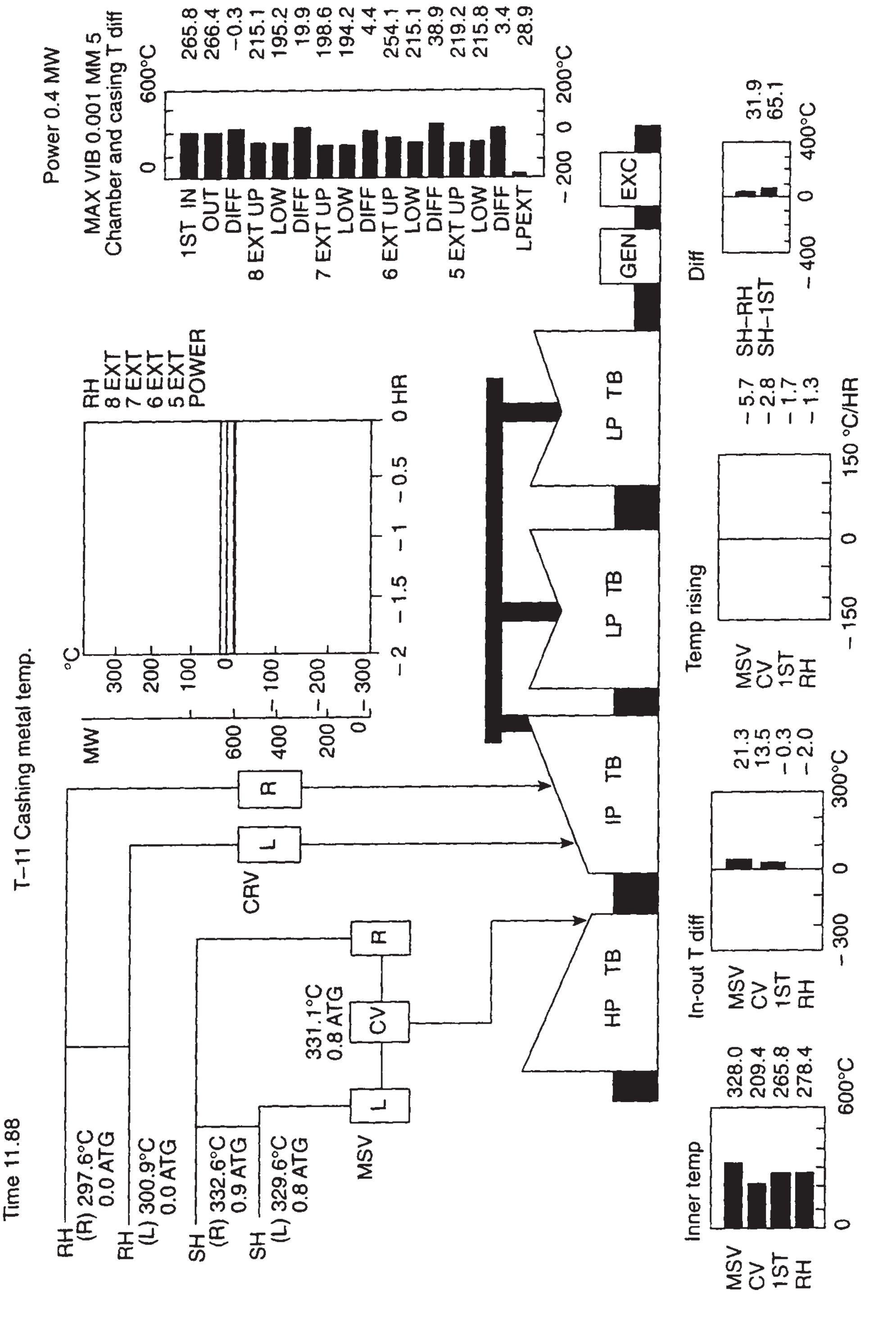


Figure 12.22 Graphic display of automated power plant.

Another important factor which should be given due importance is the selection of scan time for software block execution. The same should be chosen on the basis of the response time to the individual loops, and care must be taken not to overload the multi-loop controller. This aspect might totally jeopardise the cost-effectiveness of DCS system.

Application functions

Apart from these on-line functions, there are a number of application level functions like database management, logs and report generation, performance calculation, plant optimisation. production modeling etc. These may be carried out via high powered microprocessor based system provided with back-up storage devices. The latest generation of DCS system has come up with cost effective distributed processors to carry out the same functions.

The following application packages are required:

— Plant logging
— Sequence of event logging
— Post-trip review
— Boiler start up and shut-down analysis
— Turbine start-up and shut-down analysis
— Turbine diagnosis log
— Preventive maintenance package
— Performance calculation and heat rate deviation monitoring package.

Brief description of each of these application packages is given here.

(i) *Plant logging package.* A configurable software package can be provided that collects, processes, tabulates, displays, prints and stores historical data on a periodic and demand basis. Scheduled periodic logs are provided that automatically prints hourly, shift-wise, daily and monthly reports. These logs can include calculated variables for totals, maximums, minimums and averages as well as instantaneous measurements.

(ii) *Sequence* of *events reporting.* Sequence of events (state changes) throughout the system can be collected, correlated and printed with a resolution of one millisecond.

(iii) *Post-trip review.* Post-trip review provides a report which automatically produces a historical log of analog or digital points involved in a trip event. A trip event can be an alarm, or contact state change, or any other condition determined by the user. The information should be organised into user-defined collection groups. A report for individual collection group can be provided for each trip event. The last few reports should be saved on the system and be made available for printing. The data gathering function should be flexible to allow for a variety of time based sampling periods.

(iv) *Boiler start-up log.* The boiler start-up log is initiated at the start of boiler purge sequence based on the change in status of an input contact. The log may also be activated at the operator’s request from the operator log display. Once started, log gathers data at intervals selected by the operator and prints a report of all the data gathered. The log may be stopped by deactivating it from the operator log display.

(v) *Turbine start-up log.* The turbine start-up log may be initiated at the start of turbine start-up, based on the change in status of an input contact. The log may also be activated at the operator's request from the operator log display. Once started, the log gathers data at intervals selected by the operator and prints a report of all data gathered. The log may be stopped by deactivating it from the operator log display.

(vi) *Turbine shut-down log.* The turbine shut-down log may be initiated whenever the turbine generator is stopped, based on the change in status of an input contact. The log may also be activated at the operator's request from the operator log display. Once started, the log gathers data at regular intervals selected by operator. The log is stopped by deactivating it from operator log display.

(vii) *Turbine diagnosis log.* The turbine diagnosis log may be activated by the operator from the operator log display. Once started, the log gathers data at fixed intervals, and then prints a full report of that data.

(viii) *Performance calculation and heat-rate deviation monitoring package.* The real-time performance monitoring system must cater to provide the following:
— Various plant equipment performance calculations
— Controllable losses and heat rate deviation calculations
— Overall plant performance calculations
— Heat rate versus generation curves for economic dispatching.

All real-time data are smoothened by time averaging input points.

All final results should be suitably quality tagged and presented to the manager incharge of operation in the form of bar graphs, graphics, trends, tabular displays and hard copy reports.

Performance calculations modules. Various performance calculations are handled by configurable modules. These are

(a) *Boiler performance calculations* which include theoretical air, dry air, dry gas, air heater performance, boiler efficiency (input/output and/or heat loss method), boiler losses and boiler cleanliness factors.

(b) *Turbine performance calculations* which includes HP turbine efficiency, IP turbine efficiency, extraction flows, re-heater steam flow, IP and LP exhaust flows, expected first stage pressure, and net turbine heat rate.

(c) *Condenser calculations* which include actual heat transfer coefficient, expected heat transfer coefficient, actual condenser cleanliness factor, expected back pressure, and circulating water flow.

(d) *Feed water heater calculations* which include mass and energy balances, terminal temperature difference (TTD), drain cooler approach temperature (DCAT), and heat transfer coefficient.

(e) *Unit performance calculations* which include gross unit heat rate and net unit heat rate.

12.3.5 Communication

The communication system plays a major role in the distributed control system. The processors exchange signals amongst themselves and transmit the necessary signals to the operator. The power plant dynamics are generally faster than that of other plants. Further, there are areas where protection signals have to act almost instantaneously. The functional requirement calls for a high speed communication link preferably without any master or bus directors, and moreover, the present day industry requirement calls for adherence to standard communication protocol like IEEE 802 standards.

The system may use IEEE 802.3 CSMA/CD (Carrier Sense Multiple Access/Collision Detection) technique as the main communication protocol. Moreover, the communication system uses exception updating. In exception updating, all points are transmitted only when they change beyond an extra band introduced on the reporting margins.

12.3.6 Advanced Control Systems

Variable pressure control

In variable pressure thermal power plants, due to the pressure change in boiler, over or under pumping of large degree is required at load change as compared to constant pressure thermal power plants. These plants have a tendency of steam temperature fluctuation because of transient between feed water and fuel flows.

Adaptive Control System techniques can be used to control the variable pressure thermal power plants. The boiler model is used to calculate the most probable steam temperature as well as predict the steam temperature as shown in Fig. 12.23.

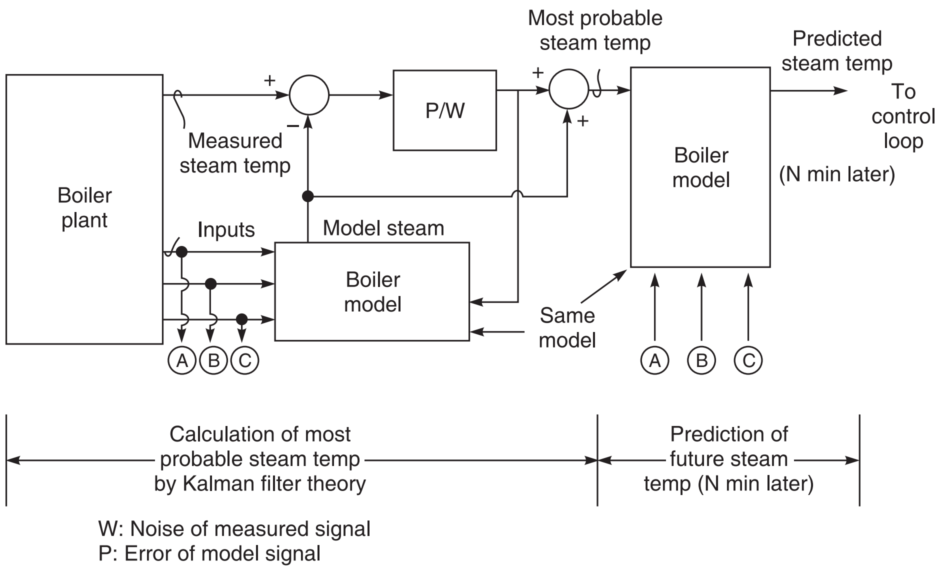


Figure 12.23 Adaptive control system for variable pressure control power plants.

Combined plant control

Gas turbine and steam turbine combined plant is another type of thermal power plant which requires complex control system. Kawasaki Power Station of Japan has a separate gas turbine and one steam turbine. The plant is completely automated from circulating pump 'start' to 'full' load or vice versa. Control components such as digital controllers and unit control systems are similar to ordinary thermal power plant automation. However, in case of combined plant control, intensive simulation analysis is performed to determine the performance of control system before deciding the control system configuration. Figure 12.24 shows the control system configuration at Kawasaki Power Station.

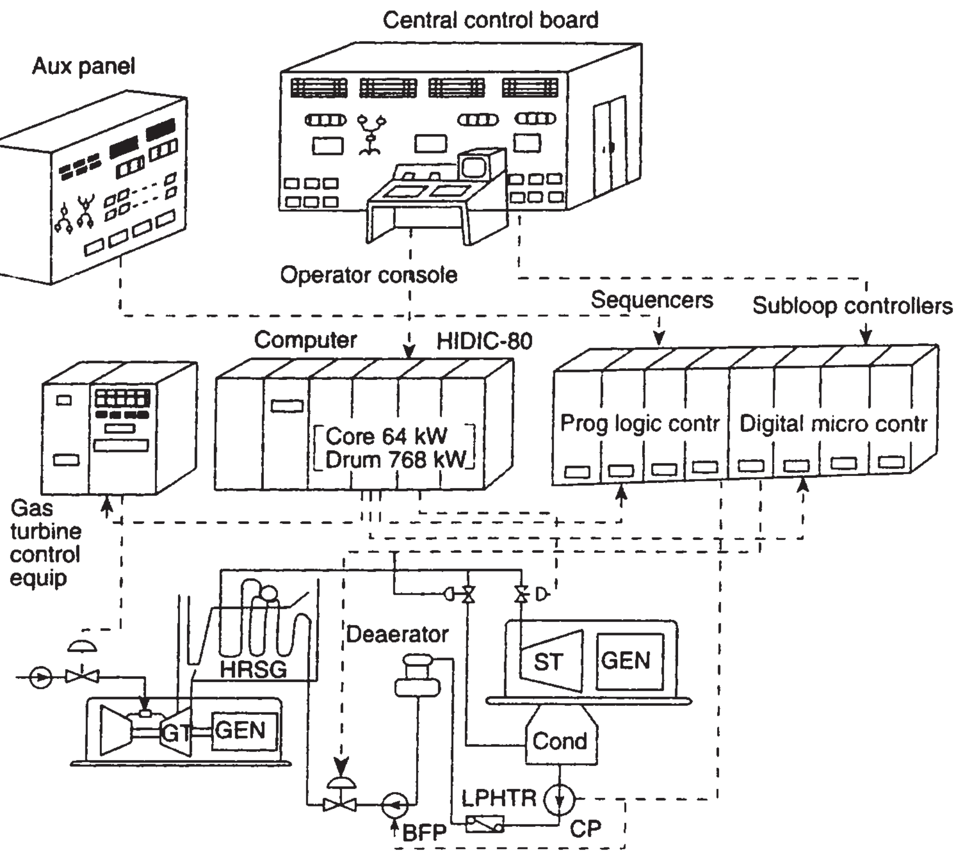


Figure 12.24 Combined power plant control Kawasaki power station, Japan,

It is apparent from the various features mentioned above that, even though introduced in a very judicious manner for power plant application, DCS systems with their flexibility and functionality are emerging out as inseparable part of the plant. As discussed earlier, the architecture of the system can vary from plant to plant and depends not only on the characteristics of the thermal power plant but also on the end user.

It is also identified that with such powerful tools there is ample scope of further improvement in the following areas some of which have already been implemented in modem power plants.

— Advanced control using predictive and model based algorithms
— Heat cycle optimisation
— Expert system based alarm analysis package.

12.4 WATER TREATMENT PLANT

The process of treatment for potable water is explained in Fig. 12.25. The various steps involved in water treatment are:

— Pre-chlorination
— Analysis of raw water quality
— Chemical dosing

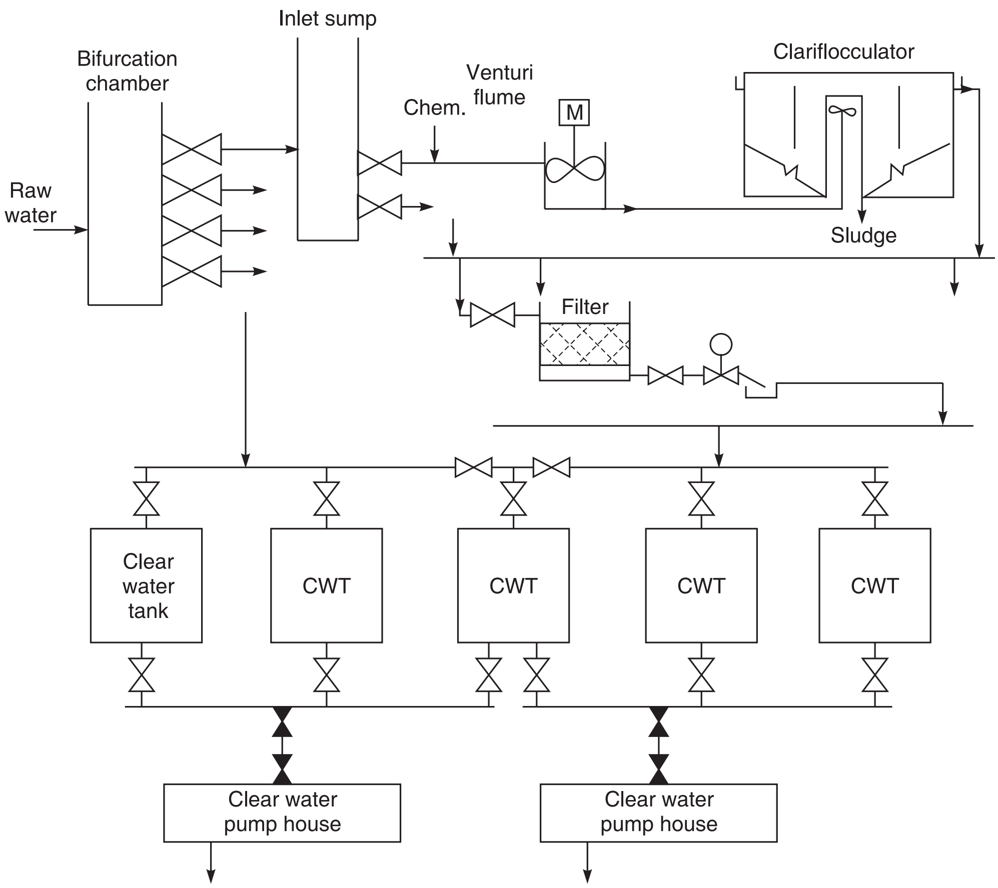


Figure 12.25 Water treatment plant.

— Clariflocculation
— Filteration
— Post-chlorination
— Pumping for water distribution.

The automation of water treatment plant is very important considering that majority of diseases are water borne. In addition to improvement in quality of water, the automation should result in saving of chemicals, uninterrupted supply and availability of more clear water.

12.4.1 Automation Strategy

The individual units in water treatment plant can be monitored and controlled by DDC. Following are the unit level controls required.

— Pre-chlorination control
— Chemical dosing control
— Sludge level control in sump of clarifloculator
— Sequential control of filter backwashing
— Post-chlorination control
— Water distribution control.

Pre-chlorination control

Following variables are monitored for liquid chlorine dosing at the raw water stage (Pre-chlorination):

— Raw water turbidity
— Raw water pH
— Raw water flow
— Liquid chlorine flow.

Depending on raw water turbidity, pH and flow of raw water the flow of liquid chlorine is decided on the basis of ratio control algorithm (Fig. 12.26). Both the controllers are implemented through software in DDC (Fig. 12.27).

Chemical dosage control

The alum dosing is done on raw water, after pre-chlorination based on the variables like, raw water flow, raw water turbidity, raw water pH.

The ratio control algorithm is implemented through software as given in Fig. 12.28.

Sludge level control

Figure 12.29 shows the schematic of sludge level control in the sump of clariflocculators. As soon as the sludge level reaches the high set-point, the sludge pump is started and is continued till sludge level reaches low set-point.

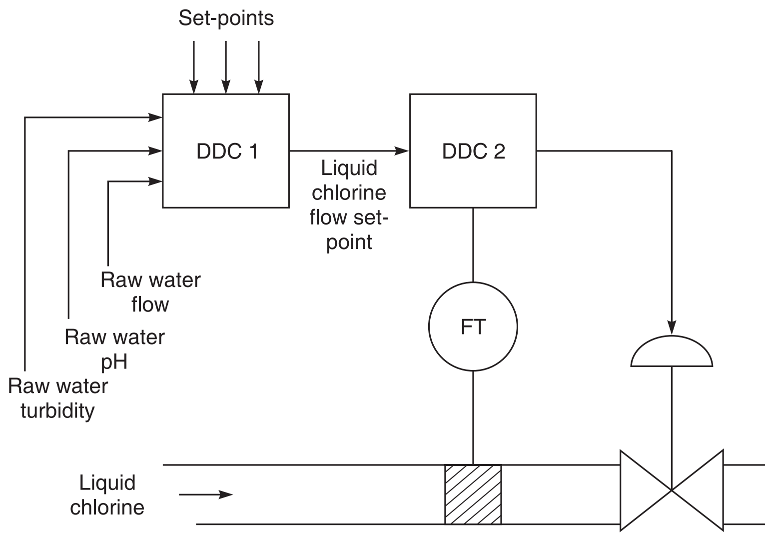


Figure 12.26 Pre-chlorination control—A schematic.

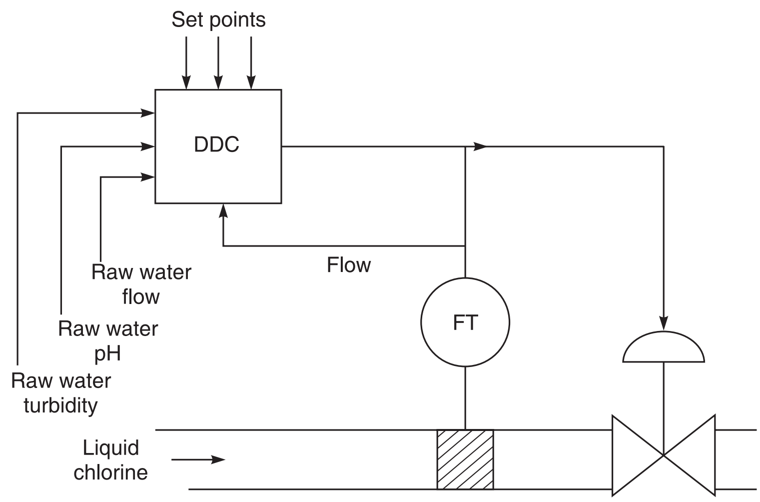


Figure 12.27 Ratio control of pre-chlorination—Single DDC schematic.

Filter backwashing control

The backwashing of filter is required to get rid of the impurities which if deposited in sand filters, result in increase of head loss of filter. The various control valves and actuators involved in filter backwashing are shown in Fig. 12.30 and the sequence of filter backwashing is shown in Fig. 12.31. We shall now develop the ladder diagram for backwashing application. Following notations have been used while developing the ladder diagram:

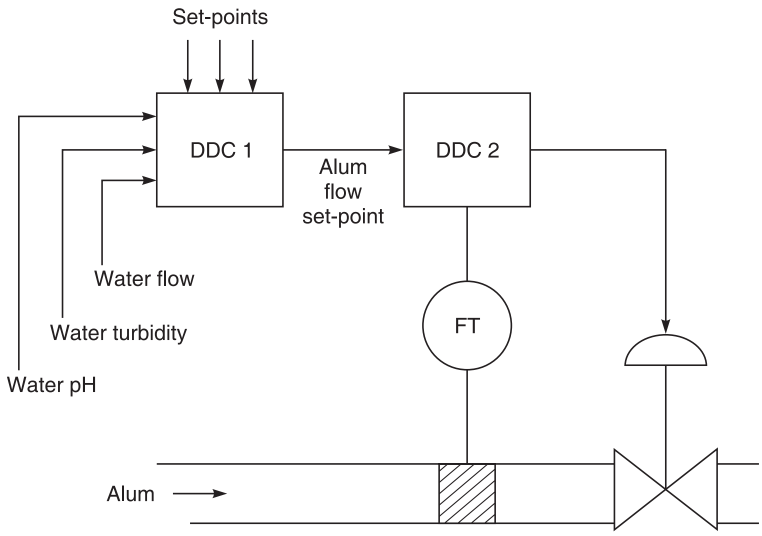


Figure 12.28 Alum dosing control—A schematic.

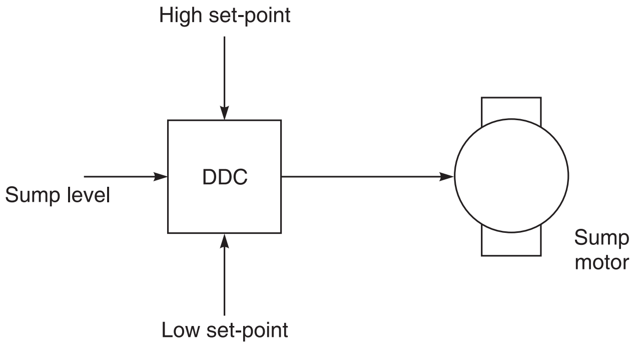


Figure 12.29 Sludge level control—A schematic.

(a) R_1 and R_2 are the relays for starting the backwashing operation.

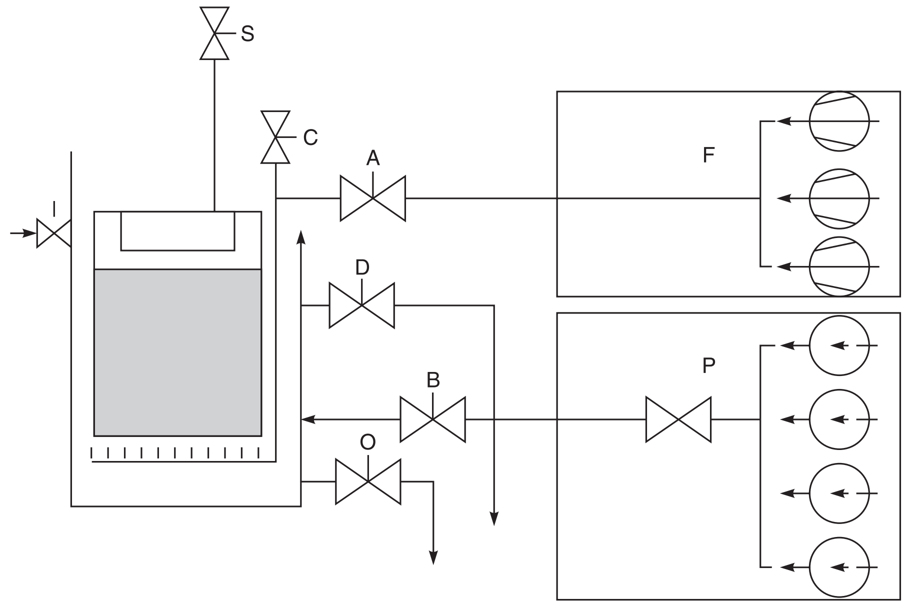
(b) *A, B, C, F, I, O, P,* and *S*, R_1 and R_2 are relays to start/stop an operation. *A, B, C, F, I, O, P* and *S* relays correspond to valve of same notation.

(c) Letters *a*, *b*, *c*, *f*, *i o*, *p*, *s*, r_1 and r_2 correspond to normally open contacts for corresponding relays *A, B, C, F, I, O, P, S*, R_1 and R_2.

(d) Small letter *ic, oc* correspond to normally closed contacts for corresponding relays *I* and *O*.

(i) When relay is energised
then valve is operated, i.e. open
NO contact = Closed
NC contact = Open

(ii) When relay is deenergised
then valve is closed



A = Air valve
B = Backwash valve
C = Drain valve
F = Blower section
I = Inlet valve
O = Outlet valve
P = Pump section
S = Syphon actuator

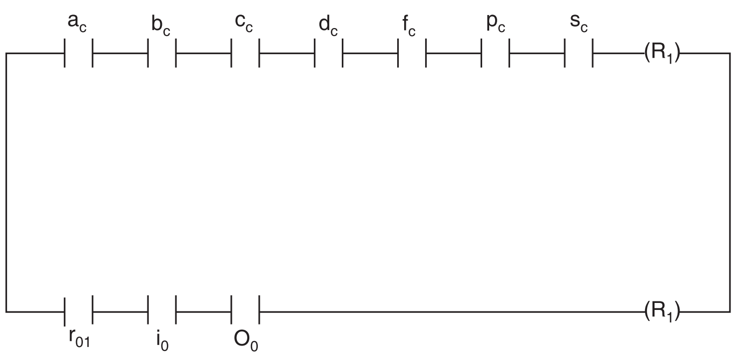
Figure 12.30 Filter configuration for backwashing.

NO contact = Open
NC contact = Closed

Now, we take up the operations in sequence as described in Fig. 12.31

Operation-I

Check initial conditions, i.e. Valve I = Open; Valve O = Open; all other Valves = Closed



FILTER WASHING SEQUENCE

O = Open or on, C = closed or off

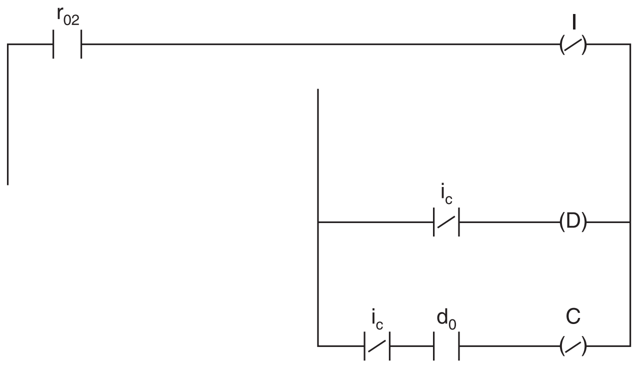
ALO = time in sec

VALVES	A	B	C	D	F	I	O	P	S	ALO
NORMAL OPERATION	C	C	C	C	C	O	O	C	C	
1 Close inlet valve						C				
2 Open drain valve				O						
3 Close outlet valve							C			
4 Allow low level										30
5 Act on syphons									O	
6 Allow										5
7 Stop act on syphons									C	
8 Allow level down										30
9 Open air valve	O									
10 Start blowers					O					
11 Allow air										300/800
12 Stop blowers					C					
13 Close air valve	C									
14 Open dearation air			O							
15 Allow										20
16 Close dearation air			C							
17 Open backwash valve		O								
18 Start backwash pumps								O		
19 Allow										300/600
20 Stop backwash pumps								C		
21 Close backwash valve		C								
22 Act on syphons									O	
23 Allow										5
24 Stop act on syphons									C	
25 Close drain valve				C						
26 Open inlet						O				
27 Open outlet							O			
NORMAL OPERATION	C	C	C	C	C	O	O	C	C	
VALVES	A	B	C	D	F	I	O	P	S	ALO

Figure 12.31 Filter washing sequence.

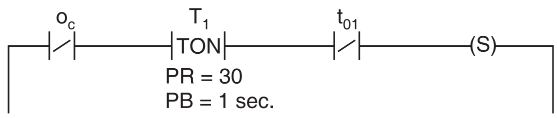
Operation-II

Close Inlet Valve I, Open Drain Valve D, and Close Outlet Valve C



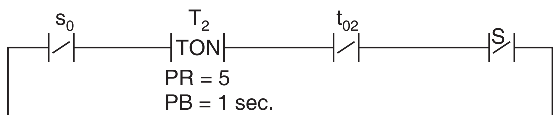
Operation-III

Allow low level (delay of 30 seconds); Act on syphon



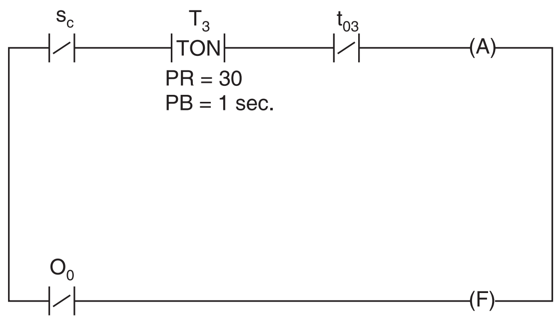
Operation-IV

Delay of 5 seconds; Close syphon



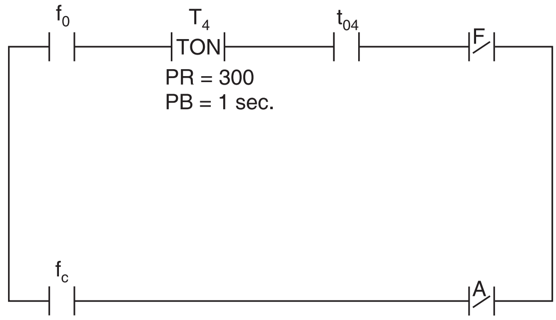
Operation-V

Delay of 30 seconds; Open air valve; Start blower



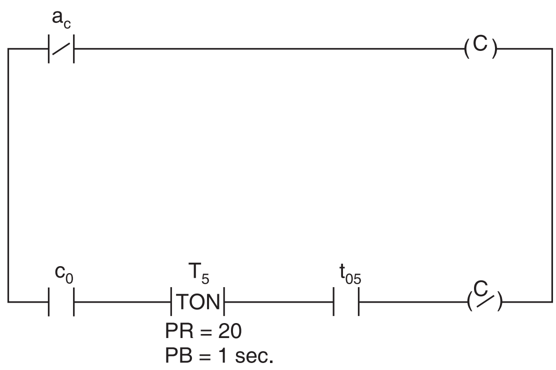
Operation-VI

Delay of 300 seconds; Stop blower; Close air valve



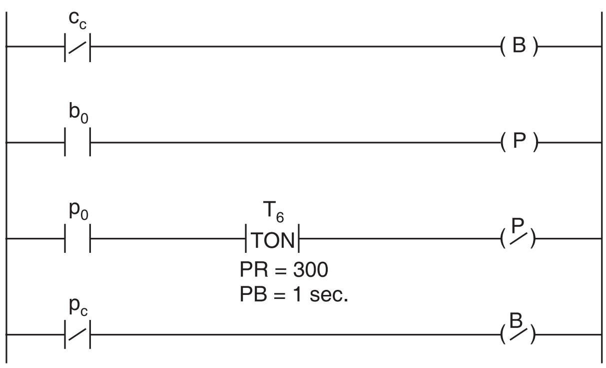
Operation-VII

Open dearation air valve; Delay of 20 seconds; Close dearation air valve



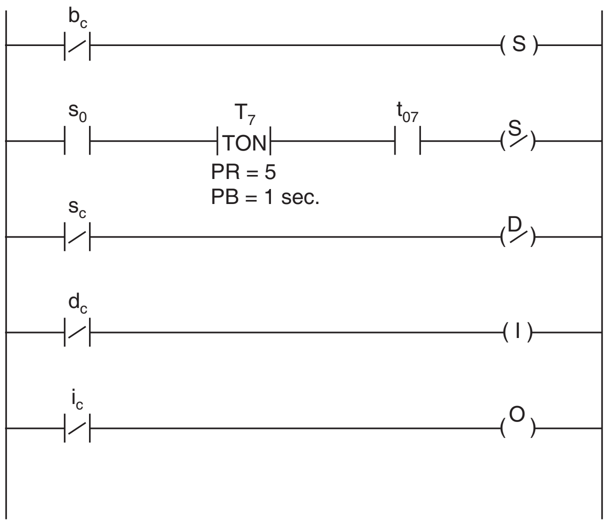
Operation-VIII

Open backwash valve; Start backwash pump; Delay of 300 seconds; Stop backwash pump; Close backwash valve



Operation-IX

Act on syphon; Delay of 5 seconds; Stop act on syphon; Close drain valve; Open inlet valve; Open outlet valve



The complete ladder diagram for sequential control of filter backwashing is shown in Fig. 12.32. This can be implemented easily, using a simple PLC or using sequence control algorithm on Digital Controller.

Post-chlorination control

The liquid chlorine is added to filtered water to compensate for impurities in the distribution lines which supply the water. The post chlorination performed is based on following variables:

— Water flow
— Residual chlorine present in water
— Liquid chlorine flow.

It is simple ratio control algorithm implemented through software on DDC (Fig. 12.33).

Water distribution control

Depending on water demands in various localities, the water distribution is controlled to satisfy the needs of all. Figure 12.34 shows the structure of water distribution control system. The water demands of various localities are input to computer which performs optimisation on the basis of present flow in each water main, level in clear water reservoir, etc. Based on the result of optimisation the computer directs the DDC to start/stop various pumps.

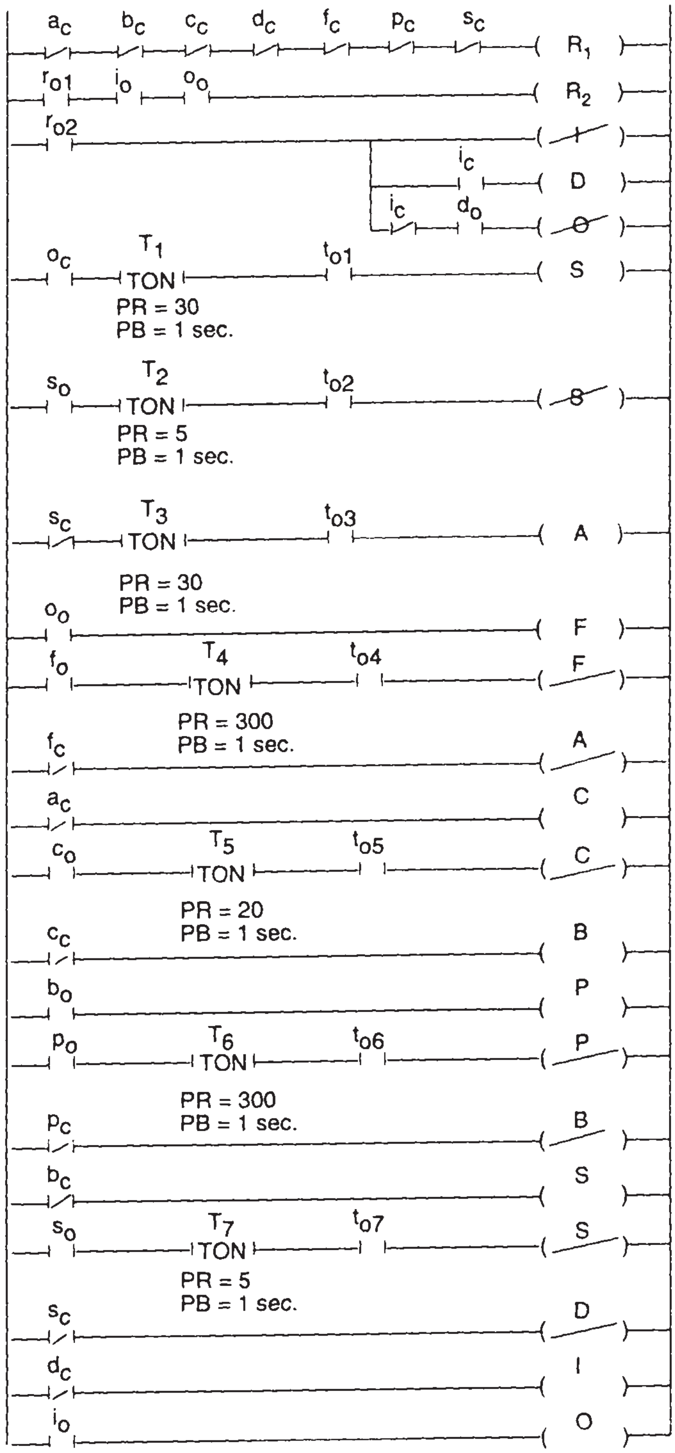


Figure 12.32 Ladder diagram for filter backwashing sequential control.

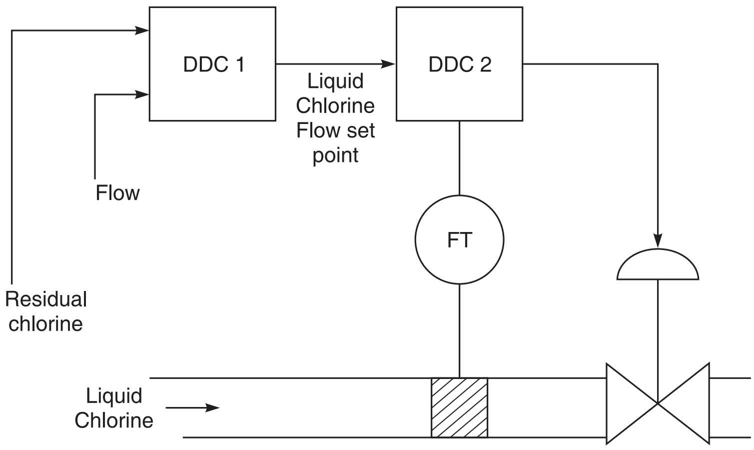


Figure 12.33 Post-chlorination control—Schematic.

12.4.2 Distributed Digital Control

Figure 12.35 shows the distributed control structure for an 100 MGD water treatment plant having 8 clariflocculators, 40 filters and 2 clear water pump houses. The plant is bifurcated structurally into two 50 MGD plants. The control hierarchy shown in Fig. 12.35 is clearly dictated by physical structure of plant.

The functions of various units are described below:

Central monitoring and MIS

The functions of Central Monitoring and MIS are

1. Monitor all distributed control operations
2. Plant wide monitoring status and alarm
3. Centralised control
4. MIS systems support
   — Long term historical storage of process data
   — Analysis and graphical plots of historical data
   — Report generation capability
   — Preventive maintenance management like inventory control and maintenance of plant operations summaries.

The Central Monitoring and MIS System can be implemented using Local Area Network with host computer performing majority of monitoring and MIS Functions. In addition personal computers with plant incharge, laboratory incharge and maintenance and inventory controller may perform important management functions. The two area controllers may also form nodes of this local area network. The LAN data hiway should be duplicated so that in case any of the area controllers fails, other area controller performs the function of both the area controllers.

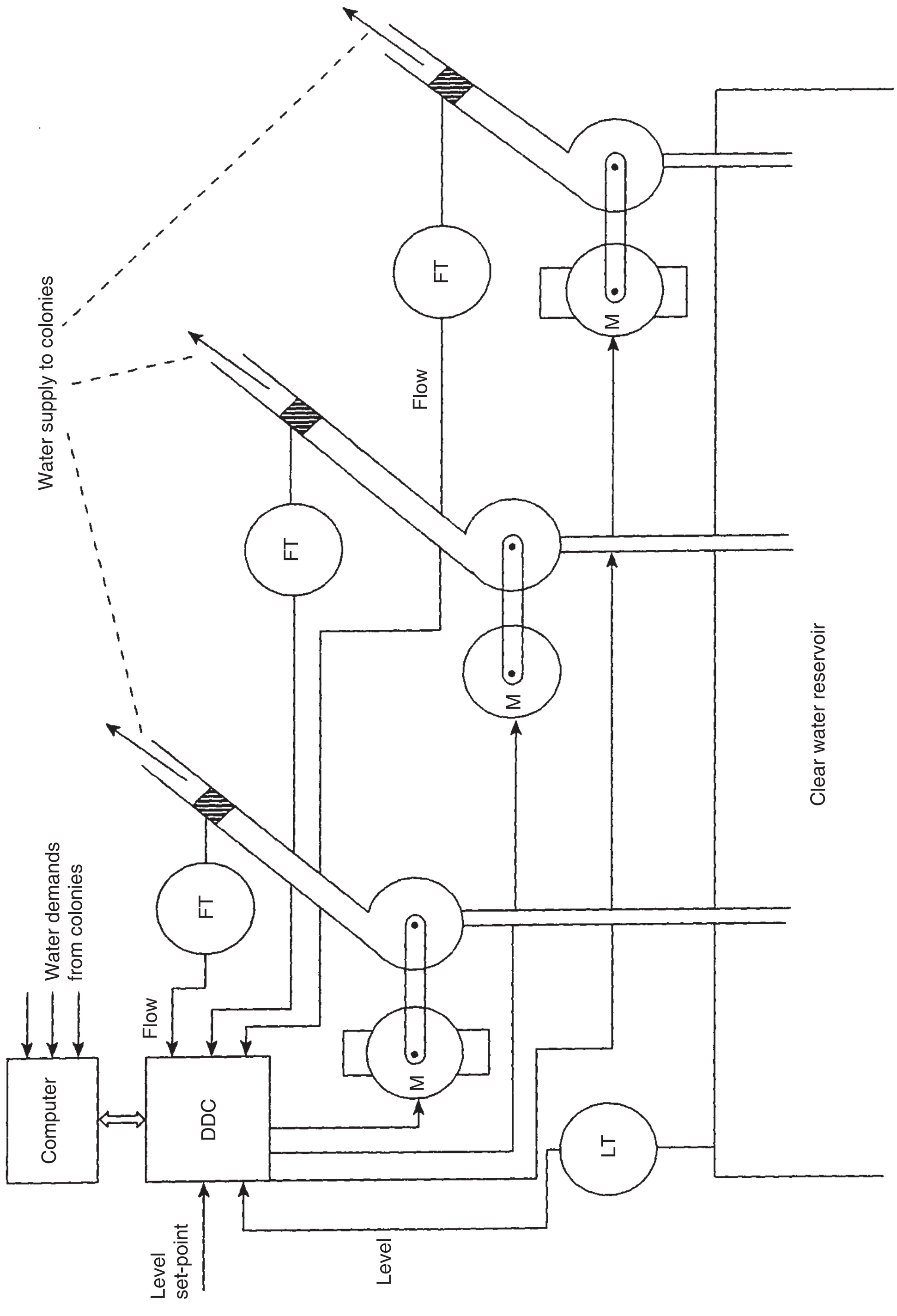


Figure 12.34 Water distribution control.

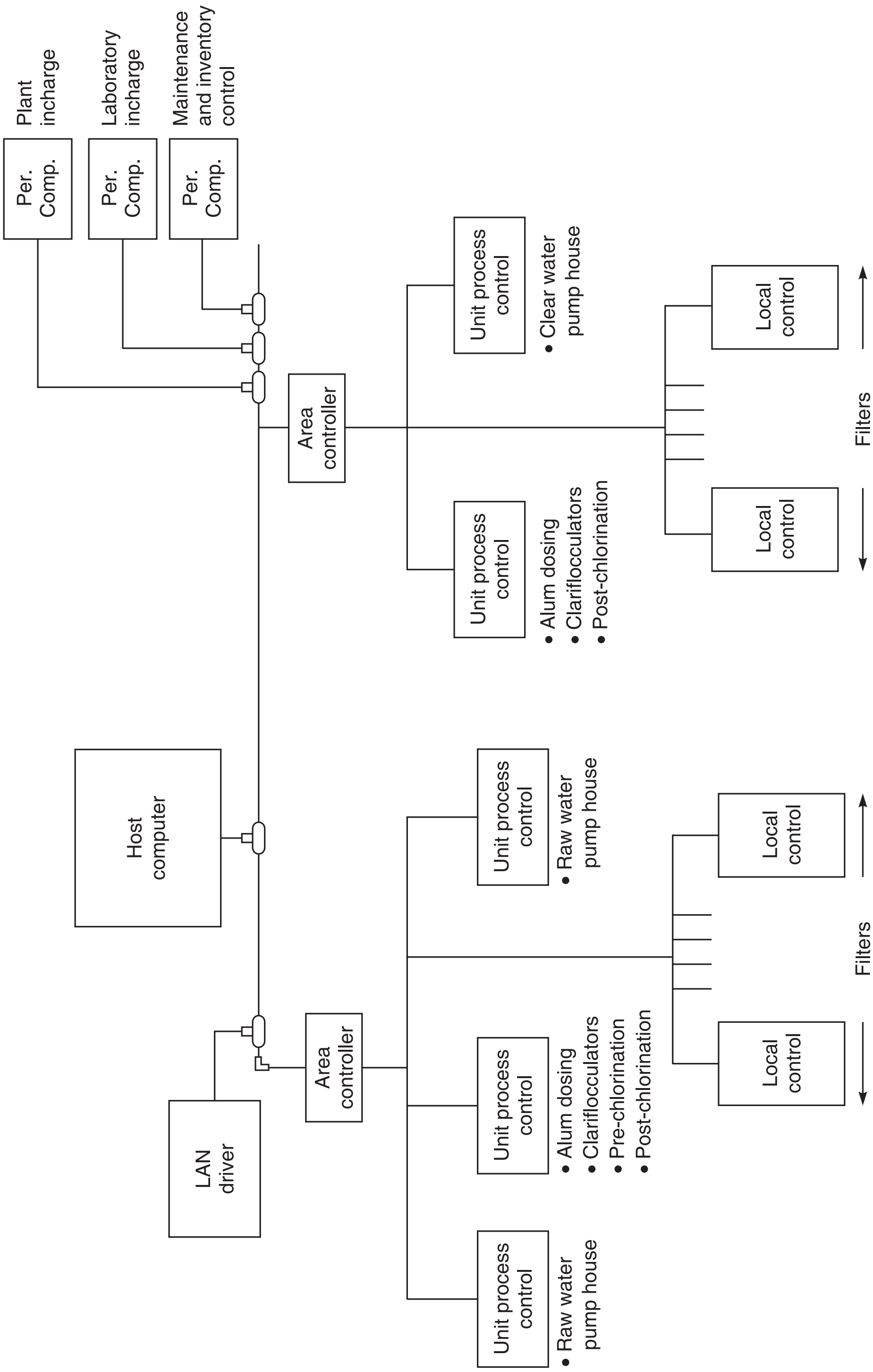


Figure 12.35 Distributed control structure for 100 MGD water treatment plant.

Area control

The functions of area control unit are:

- Status information on process equipment, automatic sequences, and primary elements
- Alarm reporting
- Supervisory control over plant processes
- Print operational reports and graphs
- Real-time trending of process variables
- Control of various auxiliaries in that area, i.e. backwash, alum dosing, chlorination etc. through unit process control and local control units.

There are two area controllers which can be implemented using Distributed Control Systems. In case of failure of one of the area controllers the other area controller should be able to take over the functions of both the area controllers in graceful degradation mode.

Unit process control

Following functions are covered in unit process control sub systems:

1. Monitoring and control of
   - raw water pump house
   - alum dosing
   - pre- and post-chlorination
   - clear water pump house.
2. Alarm generation.
3. Communication with area controllers.

Local control

Functions of local controllers are:

- Filter operation control
- Filter backwashing control
- Alarm generation
- Communication with area controller.

The Unit Process Controllers can be implemented using direct digital controllers. A number of local indicators and push buttons should be also provided at the control panels at every stage like raw water pump house, alum dosing etc. The local control of filters can be implemented using Programmable Logic Controllers. Each filter should also have a control panel containing push button control and indicators for manual back washing of filter in case local controller fails.

This hierarchy depicts the emphasis on redundancy at every stage, so that automation structure doesnot hamper the water supply.

12.5 IRRIGATION CANAL AUTOMATION

Many of the existing irrigation projects are presently suffering from problems of salinity, undependable water supplies, low cropping intensity and water scarcity at the tail end, resulting in low level of agricultural production.

The operation of irrigation canal basically involves management of following

— Information
— Organisational infrastructure
— Canal structure.

Proper irrigation canal management will involve the following:

Accurate information on irrigation water requirements

Since the objective of irrigation projects is to satisfy the crop demand as closely as possible, accurate information on crop water requirements of each outlet in the project area as a function of the growing season must be available.

Monitoring

The amount of water delivered at any given point in the system must be known, in order to make sure that the desired quantity is delivered at that point.

Accurate information on seepage losses

During the design stage, seepage losses are assumed which depend upon the type of soil through which the canal passes. If the assumed seepage loss is not correct or it varies due to variation in soil properties then it will result in delivery of either too much or too little water at a given point in an irrigation system.

Structural control

During the design stage, usually all the required structures are provided in the project. These structures must be easy to control. At the same time, they should be properly maintained all the time for effective control. The regulatory control of structure with respect to regulation in water flow should be well defined.

Communication

For high system performance, communication between the water users and the irrigation system operators is very important. Information regarding the availability of water supply, and the duration, frequency and turn time of each farmer under each outlet in the command area should be communicated to each and every farmer.

Travel time and canal dynamics

Depending upon the size of the project, water travel times can be significant. Because of the variation in the irrigation water requirements of crops during the crop growing season, the volume of water required at any given point in an irrigation system changes continuously throughout the growing season. To match the supply with the demand, several structures upstream of the point under consideration should be operated for several hours (in some cases several days) in advance. Because of the continuous fluctuations, it is very difficult for an operator to decide when and how much to open/close a given gate in the system to deliver irrigation water as per schedule.

Travel times, canal dynamics, and seepage losses are unavoidable in irrigation systems. Therefore, in order to improve system performance, the impact of these factors should be minimised by improved operational strategies. This calls for monitoring and feedback control including communication on a continuous basis. In addition, spatial and temporal distribution of irrigation water requirements in the project should be incorporated into the operation.

12.5.1 Automation Strategy

There are two basic control concepts that are used for irrigation canal control. These concepts are based on the location of information used for control, relative to control structure. The information may be regarding the flow, level at one or more points in the canal. If the information is from upstream of canal structure being controlled, the control concept is called

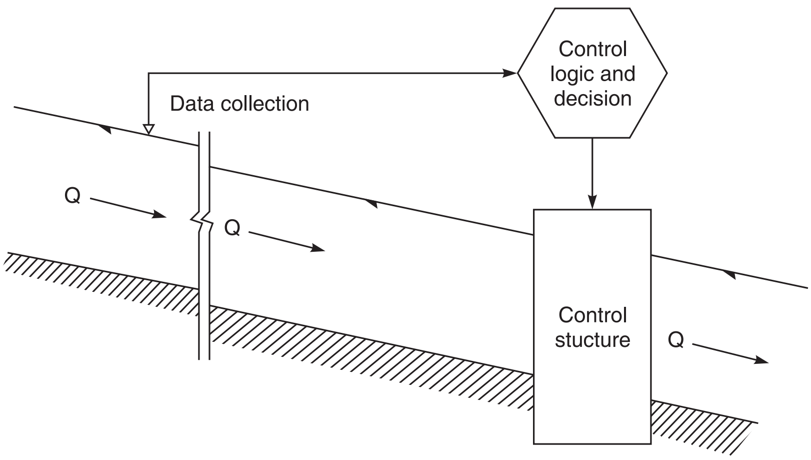


Figure 12.36 Upstream control concept.

Upstream Control (Fig. 12.36). On the other hand if the information is from Downstream of structure being controlled, the control concept is known as Downstream Control (Fig. 12.37). The upstream control is also known as Supply-based Control whereas down stream control is called Demand-based Control we shall not discuss these concepts in detail here as these are beyond the scope of this book.

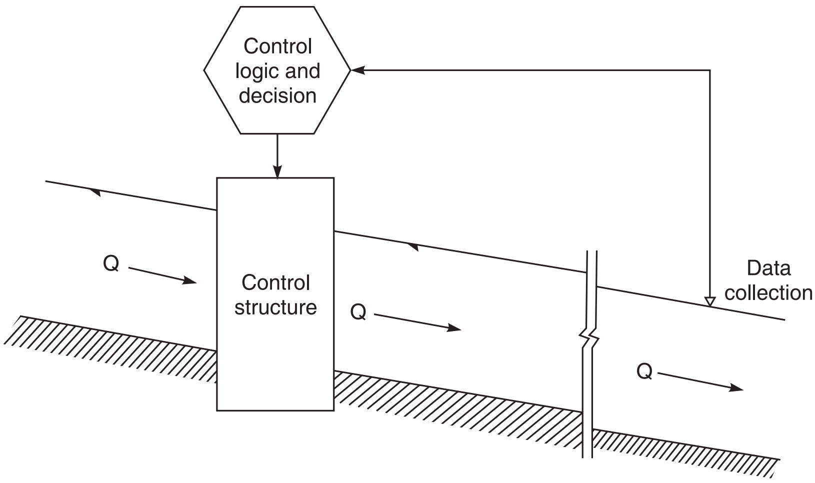


Figure 12.37 Downstream control concept.

The basic canal control methodologies in use are:

(i) *Local manual control.* Control is on-site by a human operator. This manual control is being exercised at present in majority of irrigation canals.

(ii) *Local automatic control.* Control is on-site by control equipment without human intervention. A local controller secures one or more pieces of information, such as gate position, level, flow etc. and calculates the desired control action like gate movement. It has communication with central computer for alarms. The local controller actually controls the gate movement also (Fig. 12.38).

(iii) *Supervisory control.* This method involves monitoring and control of the canal structure from central computer. Data such as water levels, gate positions, flow etc. are collected at each remote location including gate structures. The information collected at all remote locations is transmitted to central computer, where it is analysed and presented in a suitable form. Control decisions are taken by central computer and are transmitted to remote sites for execution such as gate movements.

We shall discuss this control methodology in detail now.

The telemetry and tele-control network is employed to enforce the automated management of irrigation canal. Figure 12.39 shows the structure of telemetry and tele-control network. Each RTU performs the following functions:

— Measurement
— Control
— Communication.

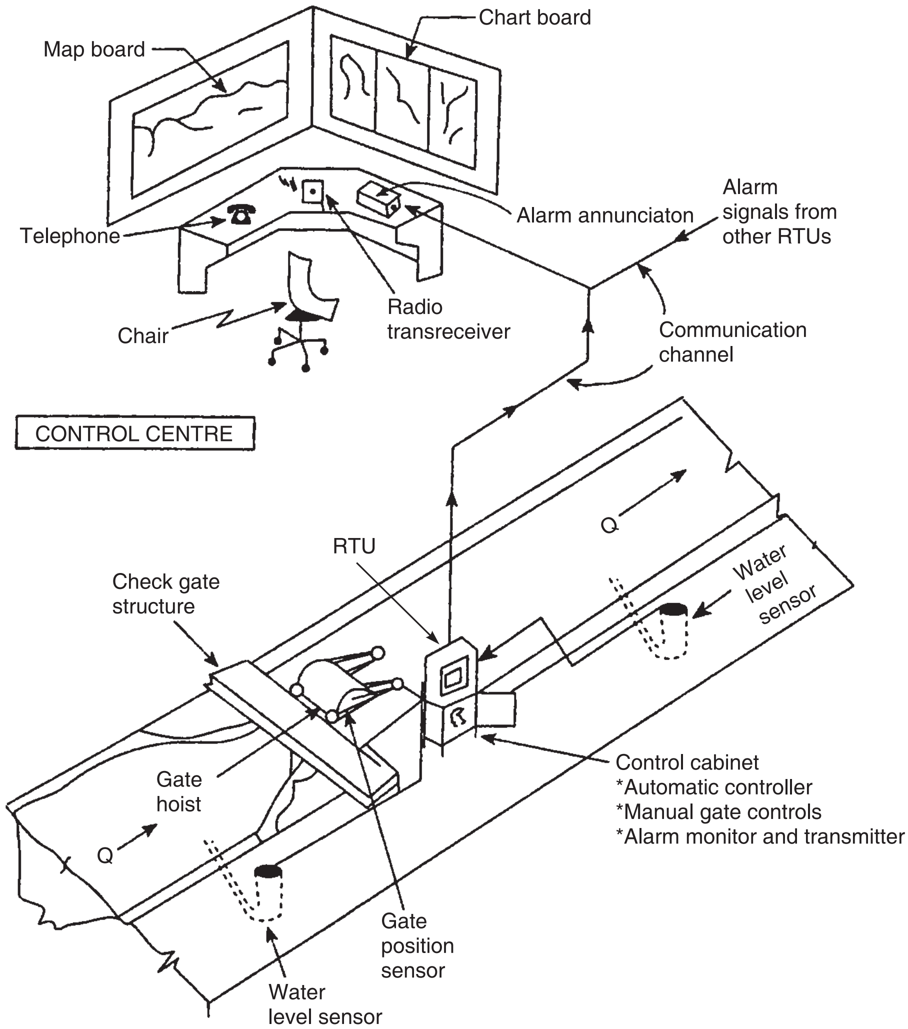


Figure 12.38 Local automatic control.

Measurement

The parameters measured by RTU may correspond to canal, gate structure or weather. The following canal data are measured by RTU in real-time:

— Water level
— Water flow.

The water level can be measured by *float, bubbler* or *ultrasonic sensors.* The float and bubbler system are well suited for canal water level measurement, as they are rugged in nature.

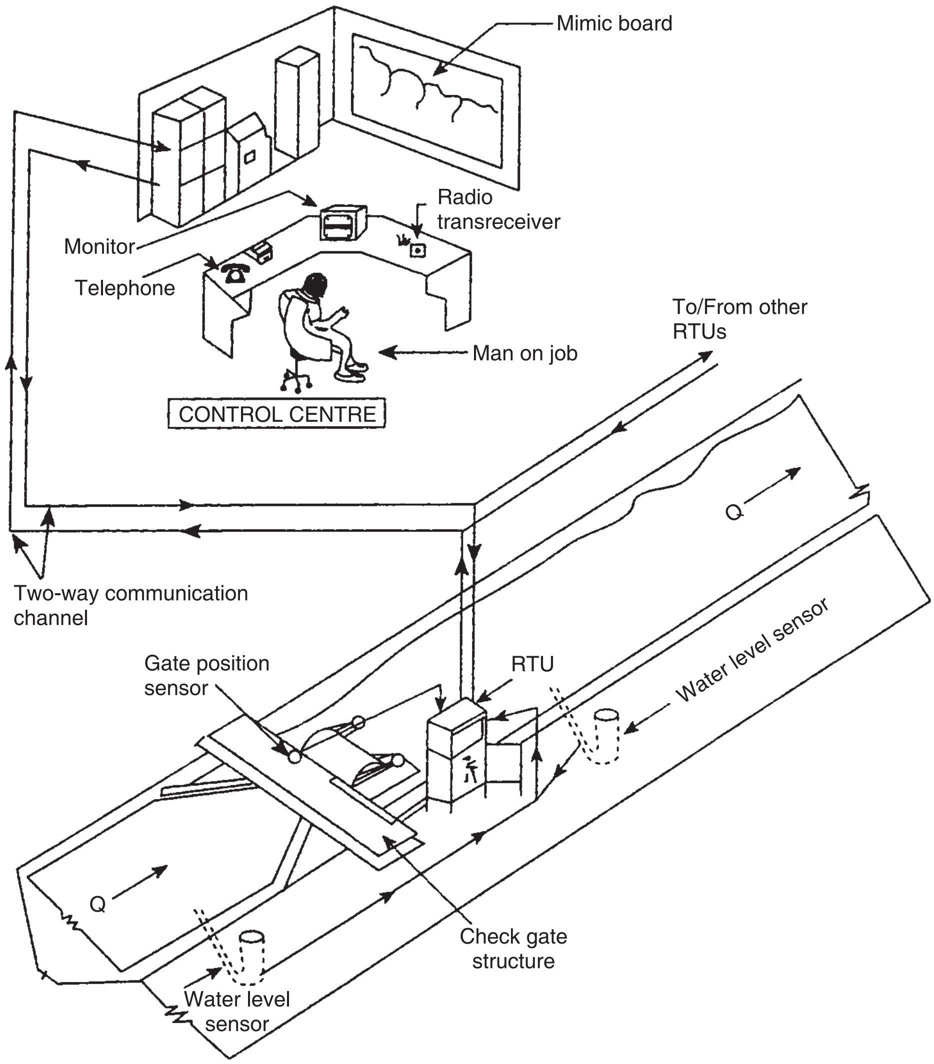


Figure 12.39 Supervisory control.

The water flow can be measured by *wier system, parshall flume* or *ultrasonic sensors.* The wier system is widely used in irrigation canals for flow measurements. The drawback of wier system is that it requires frequent maintenance.

The RTU installed at the canal gate structure measures gate position using potentiometric principle or optical shaft encoder. The potentiometric principle is more advantageous as voltage across potentiometric arm is proportional to gate opening. Thus, it gives the absolute gate opening at any instant. On the other hand, optical shaft encoder generates pulses when gate position changes. Thus, it is mandatory to store the previous position of gate in RTU in case optical shaft encoder is used. The optical shaft encoder based sensors are however more accurate, have longer life due to less wear and tear and thus are becoming more popular. The intelligent optical shaft encoder based sensors which do the limit processing and generate absolute position are also available.

The weather data is used in central computer to calculate crop water requirement. This in turn is used to prepare the optimum water schedule for various branches and distributaries.

The following weather data is measured for this purpose.

- Rainfall
- Evaporation
- Temperature
- Relative humidity
- Wind speed
- Wind direction
- Solar radiation etc.

The tipping bucket rain gauge is used for measurement of rainfall rate. The evaporation is measured by pan evaporators using float or load cell (to measure the change in water quantity due to evaporation). The air temperature can be measured by silicon diode, thermocouple or thermistors. The relative humidity is widely measured by dry and wet bulb temperature measurement. Thin film capacitors can also be used for this purpose. The anemometer and wind vanes are standard sensors for measurement of wind speed and wind direction. The last parameter, solar radiation can be measured by using thermopile detectors or silicon pyrometer.

Control

The RTU controls the gate position based on the direction from central computer. The central computer decides the new gate position based on water schedule to be executed as well as other water demands or emergency requirements (e.g. flood) that may emerge suddenly. Figure 12.40 shows the basic block diagram of gate position control by RTU which executes the DDC algorithm to control the gate position.

Communication

The communication options of RTU have been mentioned in Section 3.7.2 while describing RTU. In present sections, we shall reiterate these concepts with some more details. The communication options being used in irrigation canal control, using RTU and central computer include:

- Wireline communication
- Fibre-optic communication
- VHF/UHF radio communication
- Meteor burst communication
- Satellite communication.

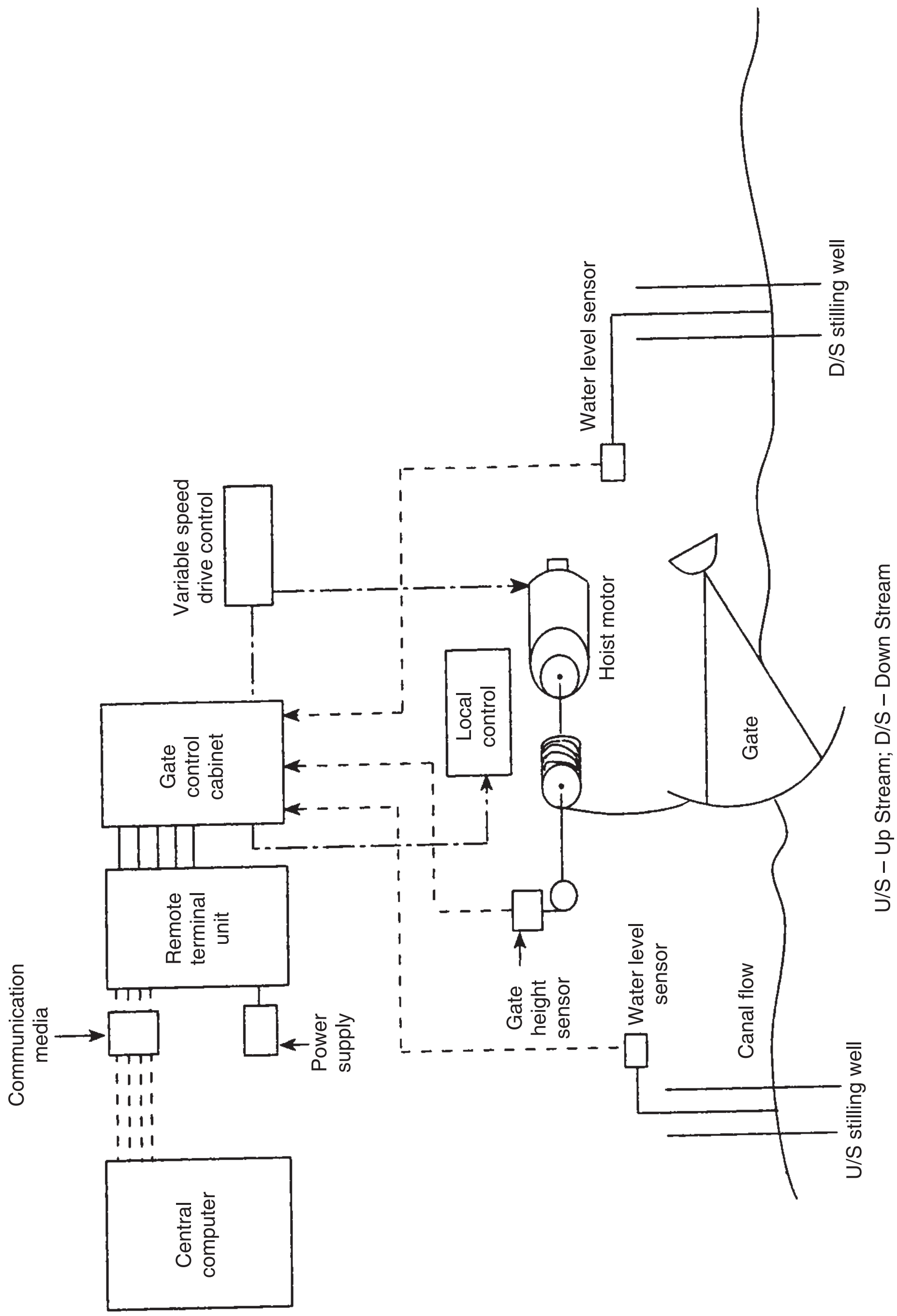
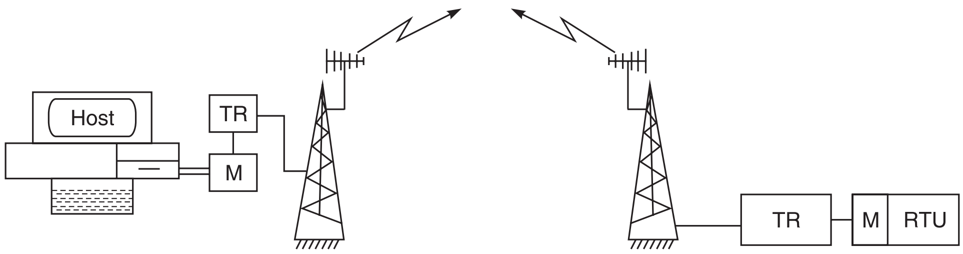


Figure 12.40 Computer-based remote control system operating CR radial gate.

(i) *Wireline communication.* The coaxial wireline communication has been widely used in major irrigation canal projects. Some of the projects have used this as standby communication, while using Radio/Satellite as main communication system.

(ii) *Fibre-optic communication.* The fibre-optic communication has also been used in recent irrigation canal projects as standby option to main communication system employing radio/satellite communication.

(iii) *VHF/UHF radio communication.* The communication set-up using VHF/UHF radio communication is shown in Fig. 12.41. The central computer sends information using omni directional antenna, so that the information is received by all the RTUs in the canal. The RTUs on the other hand, use Yagi antenna to transmit information to central computer. Where the frequency range of VHF is 30 to 300 MHz, the UHF works in the frequency range of 300 to 3000 MHz. The VHF transmission is limited to a distance of 50 km at about 25 watts of transmitter power output with 12 to 18 metre mast. On the other hand, UHF transmission is limited to 60 km at about 30 watts transmitter output with 12 metre mast. Repeaters can be used to extend the coverage area in both cases.



M—Modem, TR—Transceiver

Figure 12.41 VHF/UHF radio communication set up.

The communication equipments required are radio modems, VHF/UHF transeceivers, guyed masts and antennas.

(iv) *Meteor burst communication.* Meteoric burst is a natural weather phenomenon which occurs at all the time. When a meteor is attracted by earth, it leaves a burning trail of charged particles which lasts for several seconds as a reflecting medium. If a pair of transmitter and receiver are properly located then message from transmitter can be received by the receiver after being reflected from the trail. The meteor trail acts like natural reflecting surface. The communication achieved is claimed to be less expensive and more reliable.

The operational principle of meteoric communication is as follows:

1. The master station transmits a continuous coded signal usually in the 40–50 MHz region.
2. When a meteor appears at the proper location, it reflects that signal, to receiving station.
3. The receiving station decodes the signal, turns on its transmitter and reflects the signal back along the same path to the master station.
4. Information can be sent in either direction until diffusion reduces the electron density in the trail.

The *ionized trail* is found long enough to enable message to be sent in burst mode from one station to another. Figure 12.42 shows the meteor burst communication.

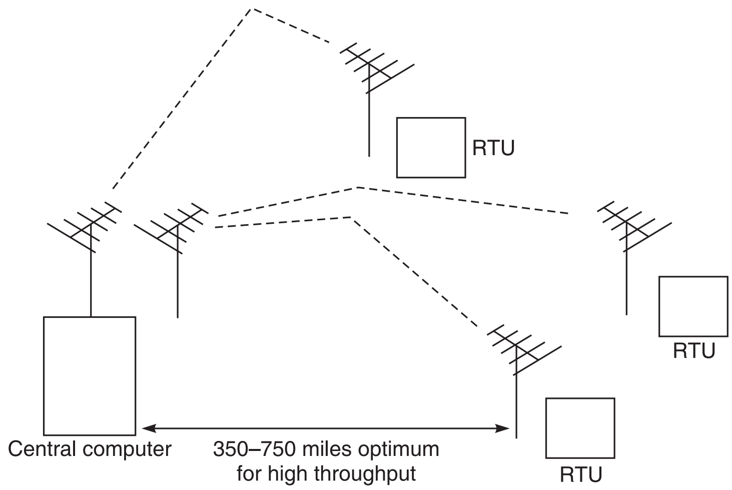


Figure 12.42 Meteor burst communication.

The meteor burst communication systems are in operation in Canada, USA, Egypt etc. Thus, it is proven technology.

(v) *Satellite communication.* Satellite communication is based on transponder action. A transponder is a repeater station in space which receives a signal, amplifies it and returns it to one or more earth stations. In order that the up and down frequencies do not interfere with each other, by international agreement, the 3.7 to 4.2 GHz band is used for the Satellite to Earth transmission and the 5.925 to 6.425 GHz band for the up-link. These are usually referred to as the 4/6 GHz bands.

The satellites are geosynchronous at an altitude of 36,000 km above the equator and hence have a period of 24 hours, the same as the earth thereby appearing to hover stationary above the earth at a fixed point. Even though signals to and fro travel at the speed of light (i.e. 3,00,000 km/sec) it takes 240 milliseconds delay. The very large bandwidth enables superfast data rates. The radiation from a satellite covers all stations within the foot print of the downward beam. In other words, it functions in the broadcast mode and hence encryption of messages is necessary to ensure privacy.

A new concept in communication ‘wizardry’ is the Very Small Aperture Terminal (VSAT). These VSAT terminals are relatively economical, have small antennas and can easily be accommodated on a roof top. As they have a very small aperture, alignment with the satellite is not a problem as the beamwidth is large. Figure 12.43 shows the satellite communication, using VSAT.

12.5.2 Decision Support System at Central Computer

The decision support system at central computer of irrigation canal automation system will have following modules:

— Irrigation scheduling module

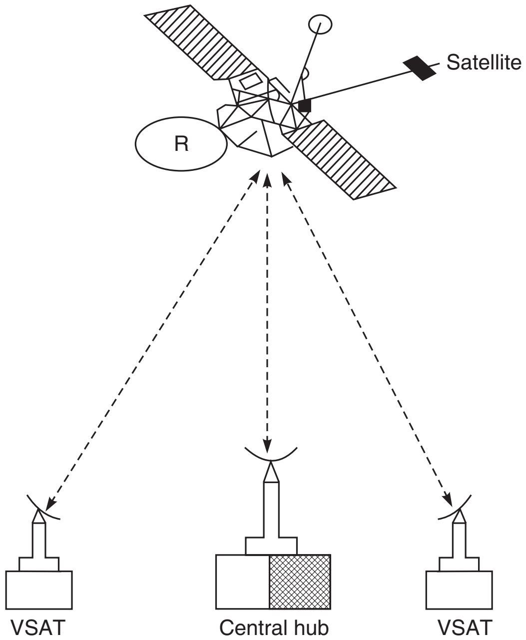


Figure 12.43 Typical VSAT network.

— Canal information system module
— Crop water requirement analysis module
— Canal flow model
— Flow accounting module.

Using the results of crop water requirement analysis module, and the real time data on level/ flow at various places in reservoir and canal, the irrigation scheduling module prepares the best water delivery schedule, so that even tail end farmer gets equal amount of water. The crop water requirement analysis module makes use of soil, crop and weather data to determine the ideal water delivery schedule. Different irrigation systems follow different rules for water delivery.

The irrigation scheduling module prepares an optimum schedule based on result of crop water analysis module and water availability based on flow/level data received from various sensors.

The central computer has to convert the delivery schedules in the form of gate movement at various canal structures and time schedule at which gate movement should be effected. Since water will take finite amount of time to travel from one structure to another the gate movement of two structures should have that much time difference.

Thus there is a necessity of canal flow model which works on basic canal dynamics data available in canal information system module. In order to convert delivery schedule (desired flow) into gate opening, by computer, the canal structures should also be part of canal flow model. This model may be steady state flow model or unsteady state flow model. Steady state flow simulation of canal operation will help in identification of critical locations and time periods through the entire period of canal operation. On the other hand, unsteady state flow simulation would help in formulation of control actions at identified time and location spots. Thus total canal automation will require unsteady state canal flow model.

There are a number of standard computer based models available for irrigation canal. Most popular among these are Utah State University (USU) canal model (Utah State University, USA) PROCAN (Energy Software, South Africa) and DWOPER (US National Weather Services). These models however should be adapted for any particular canal for simulating flow conditions in steady state or unsteady state.

The flow accounting module basically simulates seepage and other types of losses in canal and helps in water accounting etc. This module is used by all other modules to operate the canal effectively.

In addition, central computer will also be executing SCADA module, to acquire data from remote locations, and remote control of canal structures and diagnostic module to keep the complete system in working condition.

All the modules should be suitably interfaced with each other as per requirement with defined rules and procedures for canal system operation. The user must be provided required man-machine interface for easy operation.

12.6 STEEL PLANT

The Steel Industry has certain special characteristics when compared with other industries; the primary characteristic being its large scale of complex operations. The other characteristics are broad spectrum of technical processes like chemical technology in coke making, metallurgical technologies in iron and steel making, metallurgical and mechanical technologies in rolling etc. In addition, it is also associated with several other technologies, for the service units which keep the coke ovens, blast furnaces, steel melting shops and rolling mills running. This diversity and the scale of operations make it increasingly difficult to manage and control the processes effectively. In some areas, the environmental condition also contributes to the complexities of operations.

Following are the main zones in a steel plant:

1. *Iron zone* which includes Raw Material Handling, Coke Ovens, Blast Furnaces and Sintering Plant.
2. *Steel zone* which consists of LD Converters, Open Hearth Furnace, Electric Arc Furnace, Continuous Casting, etc.
3. *Mill zone* which contains Soaking Pits, Blooming Mill, Billet Mill, Wire Rod Mill, Plate Mill, Slab Mill, Storage Yard etc.
4. *Utility zone* which has the following main parts:
   — Gas Distribution
   — Liquid Fuel Distribution
   — Oxygen Generation and Distribution
   — Power Generation and Distribution
   — Steam Generation and Distribution

— Compressed Air Generation and Distribution
— Water Management
— Pollution Control Monitoring.

5. *Non-works zone* which includes miscellaneous functions like
   — Inventory Control
   — Personnel Management
   — Maintenance Management.

These zones are responsible for their individual functions and report to management which performs the tasks of both Plant Management and Production Planning.

The management and production planning functions in steel plant are the following.

— Maintenance Planning
— Inventory Control
— Project Management
— Financial Management
— Order Processing
— Scheduling
— Production Planning
— Material Tracking
— Customer Services
— Administrative and Personnel Management.

12.6.1 Automation Strategy

Steel Industry is one of the most complex industries in terms of process, size and *heterogeneity* of operations. The control of such an industry is therefore a gigantic task. At the lowest level, DDC and SCADA find wide applications. At higher level modeling and optimisation is necessary along with set-point calculation. At the highest level production control system drives the control of all the lower level units. We shall describe the distributed hierarchical control of steel plant with the control and management tasks performed at each level.

The automation strategy at different levels in distributed control system in steel plant has been described here in the following manner.

Level 4—*Production planning and MIS functions*

Level 3—*Area supervision functions*

Level 2—*Supervisory control functions*
— Monitoring and optimisation tasks
— Input data required from other supervisory controllers or from direct digital controller (Level 1)

Level I—*Direct digital control functions*
— Control tasks executed to perform above functions
— Input data required either through sensors or through operators
— Output parameters controlled.

This would enable engineers to develop the DDC algorithm for any particular zone or part of zone. The readers will also appreciate the task planning and distribution at different levels before actual development begins in distributed control environment.

12.6.2 Production Planning and Area Supervision

Figure 12.44 shows structure of levels 4 and 3 in the hierarchy of Distributed Control System. The tasks of level 4, i.e., management and planning should be implemented on mainframe computer. The computers at different areas (zones) will be of smaller sizes. These computers will be having large data storage capacity as well as good number crunching capability.

The functional distribution between various area computers have been shown in Fig. 12.44. The functions of area level computers are:

1. Establish the immediate production schedule (mill sequence, etc.) for its own area including transportation needs.
2. Apply constraints to the schedules of the production units to allow energy management, i.e., supply necessary information to the overall production scheduling system.
3. Locally optimise the costs for its individual production area.
4. Make area production reports.
5. Material tracking.
6. Use and maintain area practices data.
7. Maintain area inventory data, raw material usage data, energy usage data etc.
8. Carry out various personnel functions.
9. Diagnostics of self an lower level functions.
10. Real-time evaluation of plant performance in the operations of each area versus the goals supplied by the computer system.

The Area Computer should have large secondary storage capacity for capturing production statistics. It is desirable that area computers have capability of operating in background mode from a separate disc system. This would avoid the inadvertent modification of any foreground programme. This background capability will allow statistical studies to evaluate standard operating procedure and for various engineering studies.

12.6.3 Iron Zone

The iron zone includes the following

— Raw material handling
— Coke ovens
— Blast furnace
— Sintering plant.

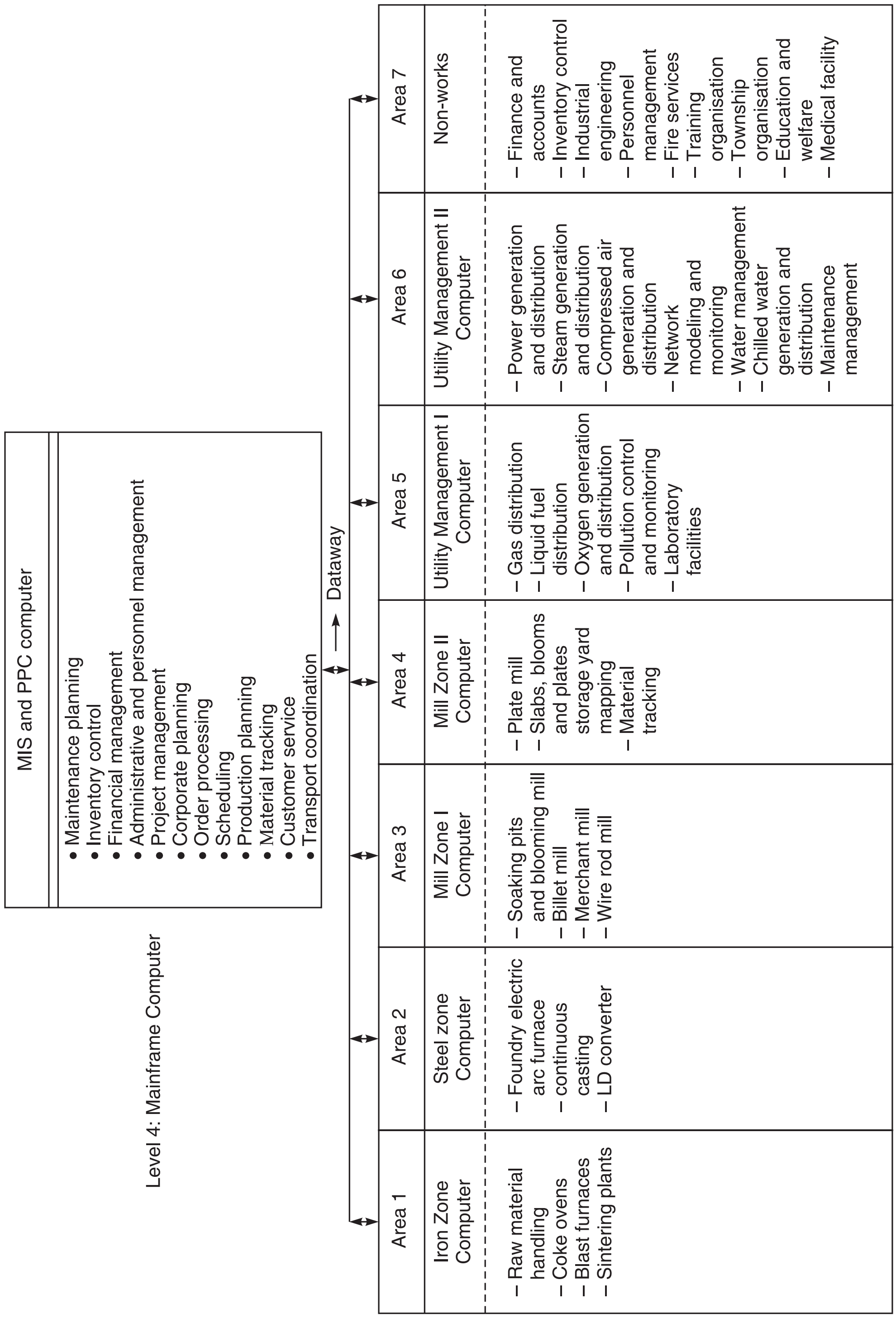


Figure 12.44 Steel plant hierarchical control system.

Raw material handling

Raw material facilities in a steel plant serve the enormous needs of the steel making process for basic raw materials like iron ore, coking coal and limestone.

The material is unloaded in raw material and coal storage yards from where it is transported to coke ovens, sintering plant, blast furnace and refractory material plant.

The computerisation of the raw material facilities covers blending, sizing and storage of the raw materials to be charged to blast furnace, sintering plant and coke oven. The operation of unloaders, conveyers, etc. can be remote-controlled from computer and the materials may be properly blended, sized and dried for feeding into the coke oven, sintering plant and blast furnaces. The measures for prevention of environmental pollution by deploying water sprayers and dust collectors also form a part of the total system.

A software package consisting of a mathematical model giving the details of coking properties and the bonding quality of various grades of coal can be used for blending these grades of coal. This will optimise the usage of expensive coke coal without affecting the coke quality.

Similarly, the sinter mix and blast furnace burden can be prepared with the help of computers.

I. Supervisory controller

(i) *Functions:*

— Grade monitoring of coal
— Grade monitoring of limestone
— Grade monitoring of ore
— Sequencing of charging order for blending to ensure correct quality of: coal, limestone and ore
— Development of optimum quality for sinter mix and blast furnace burden
— Development of optimum quality of coal
— Interaction with laboratory for database for generation of correct mix.

(ii) *Input data required:*

— Basic charging scheme to be executed
— Moisture content of material
— Other properties of materials like permeability of ore, bulk density of coal etc.
— Analysis of raw material charged to various raw material bins
— Material identification by source and bin location
— Characteristics of raw material needed for models.

II. Direct digital controllers

(i) *Functions:*

— Monitoring and controlling the weight, size, and quality of various materials
— Calculating the exact proportion needed for correct quality charge for coke oven, sintering plant and blast furnace
— Sequencing logic for equipments at yards, blast furnace, coke oven, and sintering plant
— Sequencing logic for positioning of skip or charging conveyors.

(ii) *Control tasks:*
- Monitoring of chemical composition of raw materials
- Monitoring of weight of each grade of material
- Control of moisture
- Monitoring of size of material
- Monitoring of blending operation to develop correct composition, size and moisture content
- Control of dust
- Monitoring and control of conveyer systems
- Selection of ratios of prime, medium and blendable coal against set points established by coke oven supervisory system
- Automation of opening and closing of gates of coal tower
- Sequencing logic control of the coke conveyer system
- Operation of equipment for screening and sizing of coke.

(iii) *Input data required:*
- Weight of material being unloaded
- Chemical composition of material
- Moisture content of material
- Weight of each charge for blending
- Dust content.

(iv) *Output parameters controlled:*
- Position of hopper openings
- Diversion of material to correct screen size
- De-humidification of charge (pre-heating, etc.)
- Water flow to spray washers
- Power of dust precipitators.

Figure 12.45 shows the schematic of raw material handling control.

Coke oven

Coke ovens consist of coal preparation plant, used for blending of coal which is to be fed into the coke ovens. The dry coke produced from the coke oven batteries is in different sizes.

The coke sorting plant sorts out coke into two or three fractions and sends the coke of acceptable size to the blast furnace through conveyors. The control strategy for coke oven is shown in Fig. 12.46.

I. Supervisory controller

(i) *Functions:*
- Optimisation of each coke oven's or batteries' operation for minimum energy use and/or maximum coke production.
- Development and adjustment of each oven's or batteries' internal operating conditions to achieve and maintain coke quality.

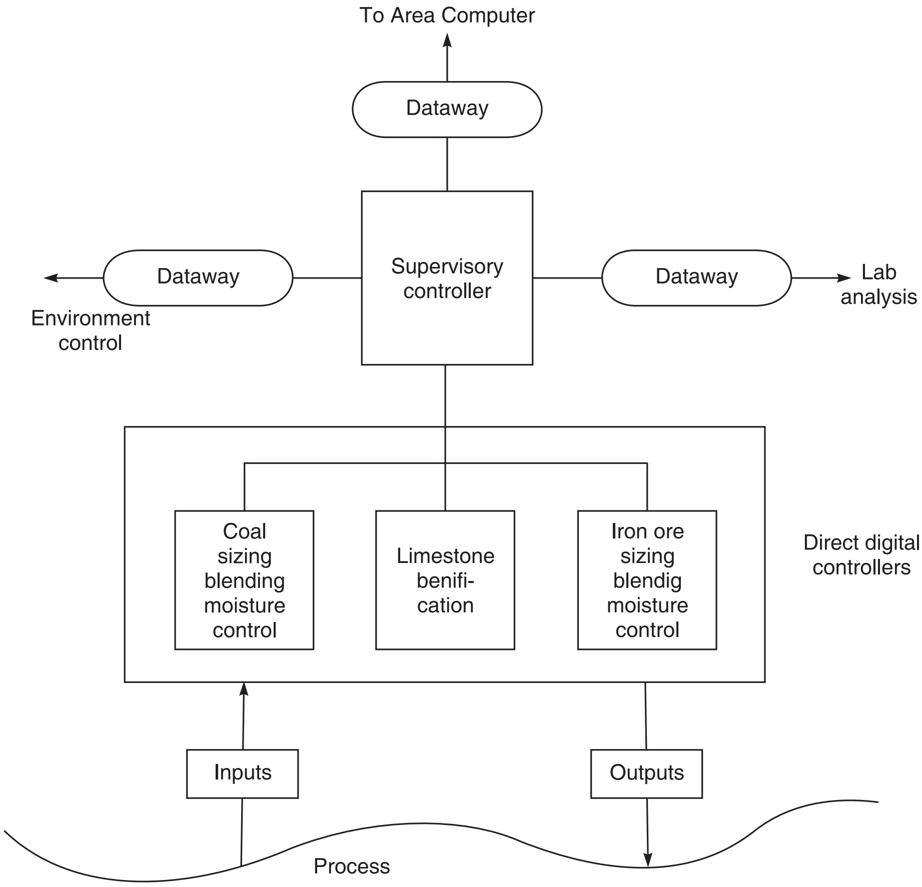


Figure 12.45 Raw materials handling and blending configuration.

(ii) *Input data required:*
- — Recommended coke oven practice
- — Basic charging scheme to be followed
- — Coal characteristics as needed for process control
- — Blast furnace gas heat content
- — Coke oven gas heat content
- — Coal moisture content
- — Bulk density of coal.

II. Direct digital controller

(i) *Functions:*
- — Sequencing logic and monitoring for charging cars and hoppers for oven charging and regulation of oven temperature to match desired coking practice
- — Control of combustion of each oven or battery to achieve minimum fuel use
- — Pressure control of the off-gas main
- — Implementation of oven pushing schedules.

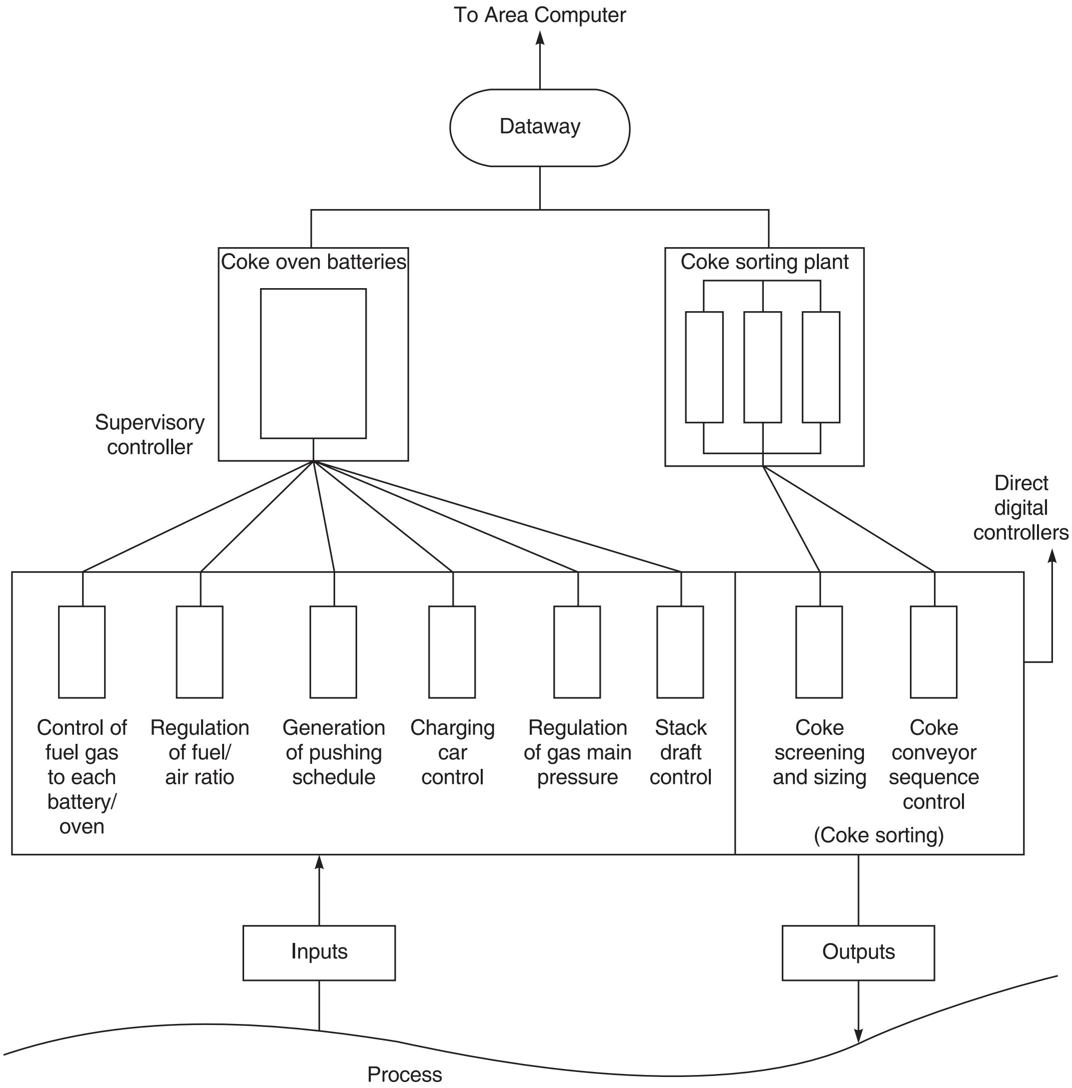


Figure 12.46 Coke oven region control.

(ii) *Control tasks:*

— Control of fuel gas to each oven combustion chamber
— Regulation of fuel air ratio for maximum fuel economy
— Control of each oven cycle including pushing
— Control of charging car filling, positioning and dumping
— Regulation of off-gas main pressure
— Determination of oven heating end-point and subsequent modification of oven cycle
— Stack draft control on combustion waste gas.

(iii) *Input data required:*
- — Fuel flow rate
- — Air flow rate
- — Coal storage bin level
- — Charging car position
- — Pusher car position
- — Oven temperatures
- — Exhaust manifold temperature
- — Charge weight
- — Charge height
- — Fuel temperature
- — Off-take gas temperature
- — Draft pressure
- — Waste gas composition (H_2, CH_4 and O_2)
- — Degree of coal grind
- — Pusher arm motor current.

(iv) *Output parameters controlled:*
- — Valve position, each fuel gas, each oven combustion chamber or battery
- — Air damper position, each oven combustion chamber or battery
- — Position of charging car output gate
- — Position of oven pusher mechanism
- — Position of output gate, each coal bin
- — Stack draft actuator position.

Sinter plant

Sinter plant consists of the following sections:

1. *Raw material receiving bins* to receive ore fines, limestone and dolomite in the sinter receiving bunkers and coke breeze into coke receiving bunkers. The discharge of different kinds of raw materials is done by disc feeders.
2. *Screening and flux crushing section* to prepare the ore mixture and for crushing the fluxes.
3. *Fuel and flux preparation section* to crush the limestone and dolomite and to grind the fuel.
4. *Raw material storage and charge proportioning section* to store the screened ore mixture, crushed flux and fuel and sinter returns. A proportioned sinter charge is made out of these ingredients.

Many steel plants use Dwight Lloyd sintering machines with the following working principle:

Each machine is provided with an ignition hood where a mixture of 12 per cent CO gas and 88 per cent of blast furnace gas is burnt. Sinter cakes are crushed in a sinter cutter to specified size. Sinter is cooled in pan coolers and stored in the loading bins.

The control of Dwight Lloyd Sintering Machine is shown in Fig. 12.47.

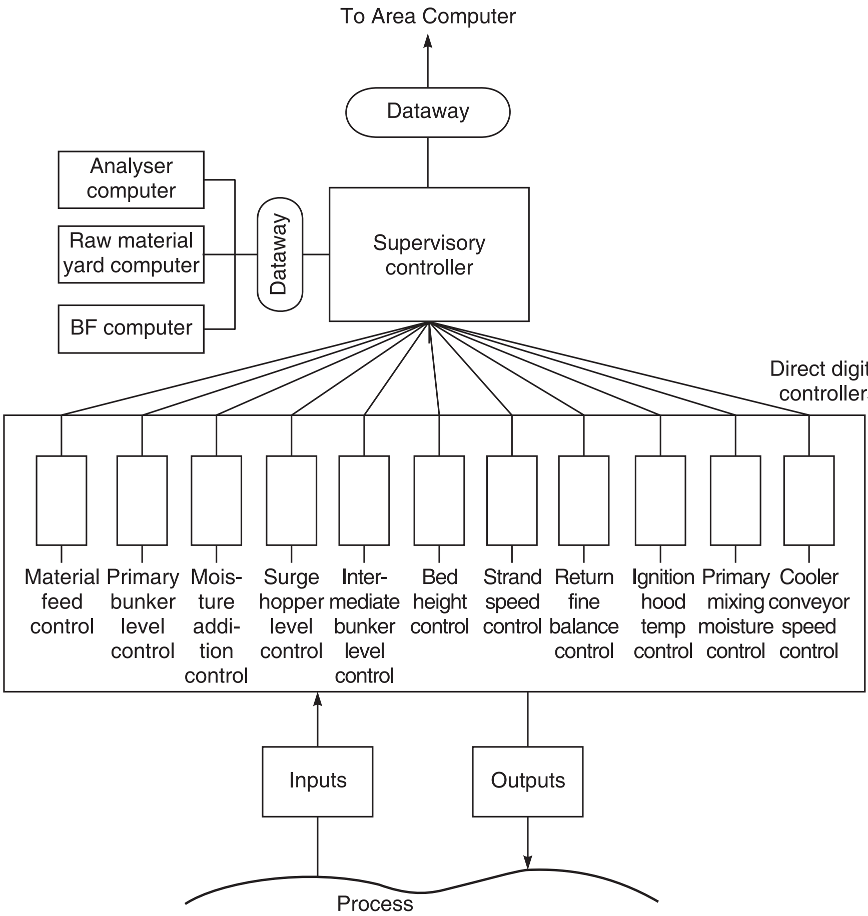


Figure 12.47 Sintering plant—Supervision and control.

I. Supervisory controller

(i) *Functions:*

— Optimisation of ratios of raw materials, coke, ore and return fines charged to Surge Hopper

— Minimisation of fuel usage on strand or pallet.

(ii) *Input data required:*

— Analysis of raw material charged to raw material bins

— Characteristics of coke charged to coke bins

— Moisture content of coke

— Analysis of ore feed charged to pallets

— Permeability properties of charged ore.

II. Direct digital controllers

(i) *Functions:*

— Material feed rate control
— Control of moisture content of feed
— Level control of charge on strand
— Composition control of strand material
— Control of charging density
— Control of bum-through with minimum additional fuel usage
— Hopper and bunker level controls
— Cooler conveyer speed control.

(ii) *Control tasks:*

— Cycling of raw material hoppers
— Cycling of coke bins
— Cycling of return fines bin
— Production of uniform mix on raw material charging conveyer
— Control of moisture of material in surge hopper
— Control of strand feed rate and strand speed
— Control of strand temperature and burn-through point
— Control of cooling of sinter product.

(iii) *Input data required:*

— Status of outlet gate, raw material bins
— Status of outlet gate, coke bins
— Status of outlet gate, return fines bin
— Water flow rate to secondary mixer
— Exhaust fan power
— Gas flow to igniting furnace
— Wind box temperatures
— Surge hopper level
— Permeability of raw sinter
— Hearth layering level (split gate position)
— Raw material conveyer power
— Raw material conveyer speed
— Raw material bin weights
— Coke bin weights
— Return fines bin weight
— Strand (pallet) speed
— Crusher power input
— Cooler conveyer speed
— Exhaust air flow temperature
— Exhaust air flow rate
— Moisture content, surge hopper material
— Surge hopper feeder

— Pallet level
— Pallet cut-off plate position
— Fuel gas analysis
— Bed temperature scan.

(iv) *Output parameters controlled:*
— Raw material bin outlet gate position
— Coke bin outlet gate position
— Return fines bin outlet gate position
— Control valve position, water line to secondary mixer
— Raw material conveyer, on-off
— Raw material conveyer speed
— Charging gate position (surge hopper-split gate)
— Charging gate position (ore bin)
— Control valve position, gas flow, igniting furnace
— Strand (pallet) power input for strand speed
— Cooler conveyer speed
— Surge hopper speeder feed control
— Pallet cut-off plate position
— Control valve position, water line to primary mixer (where present).

Blast furnace

The main constituents of blast furnace are:

(i) *Raw materials section.* Coke is supplied through series of conveyer systems from the coke screening section. Lump iron ore, limestone and additives are supplied from the ore yard by transfer of cars. Sinter is supplied through a conveyer system.

(ii) *Cast house.* Each furnace is equipped with one iron notch and two slag notches in the cast house. Each cast house has a mudgun for closing the tap hole.

(iii) *Stoves.* The stoves are designed to operate at a blast temperature of 900°C. Cyclic mode of operation of stoves is adopted. Automatic change over of stoves is provided.

(iv) *Furnace and stock house.* All the blast furnaces are designed for a high top pressure of one atmosphere. The stock house consists of rows of bunkers. Each furnace has two coke bunkers; one on either side over the skip pit. Only coke is screened at the stock house before charging into the skip. The charging of iron ore, sinter, limestone, manganese and quartzite is done by means of a scale car.

Hot metal is tapped into ladles and dispatched to the steel melting shop or pig casting machines. Slag is poured into ladles and sent to the slag granulation plant. The plant is provided with twin strand pig casting machines.

Figure 12.48 is an overall sketch of the computer control system used for each furnace. The blast furnace unit control and supervision has been divided into two separate functional units, stock house and charging (i.e., the top section of the furnace) and furnace operations and stoves (the lower section). Figs 12.49 and 12.50 outline the former while Figs. 12.51 and 12.52 show the control of the lower part of the furnace.

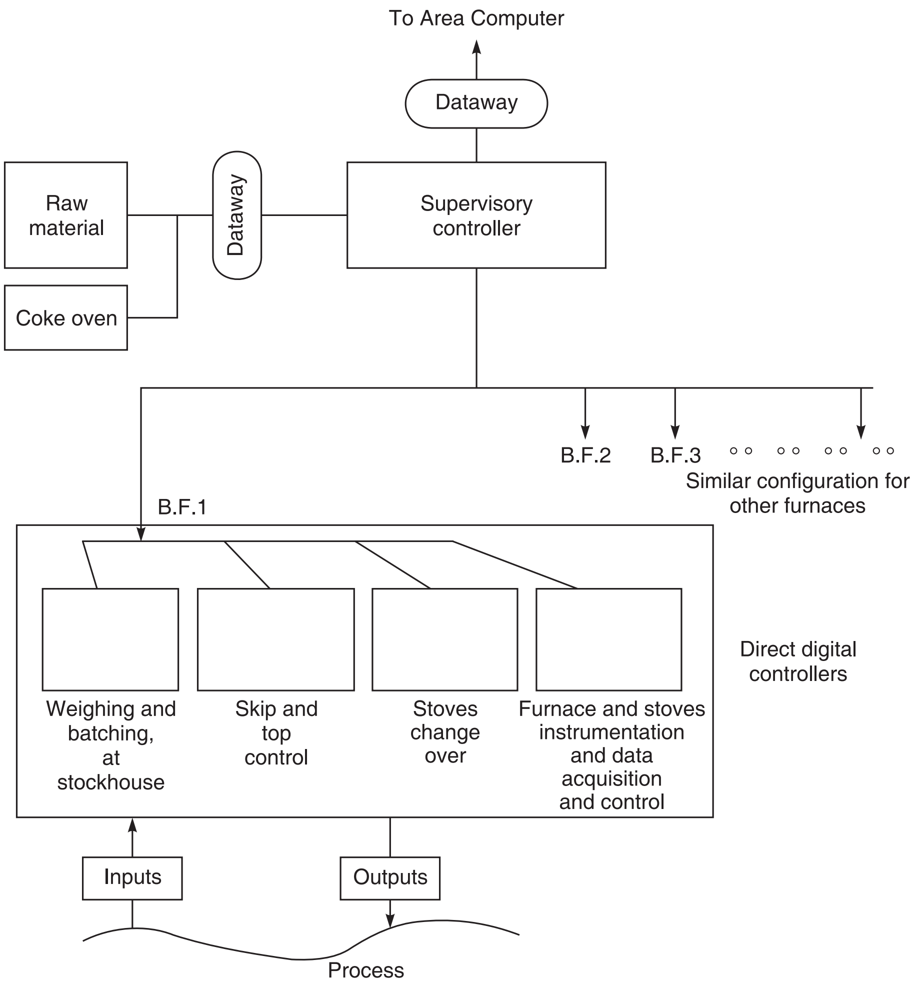


Figure 12.48 Blast furnace-Supervision and control.

I. Supervisory controller

(i) *Functions:*

— Optimisation of stove operation for each furnace

— Development and adjustment of furnace internal operating conditions for each furnace.

Blast furnace computer based supervisory controller system (Fig. 12.48) enables the blast furnace operator to assess the furnace's performance. This is in terms of both past trends and future projections as well as the furnace's current status. The system should be designed so as to generate real-time calculated parameters that represent the internal status of the blast furnace. Process improvements such as coke savings, improved production rate and decreased furnace

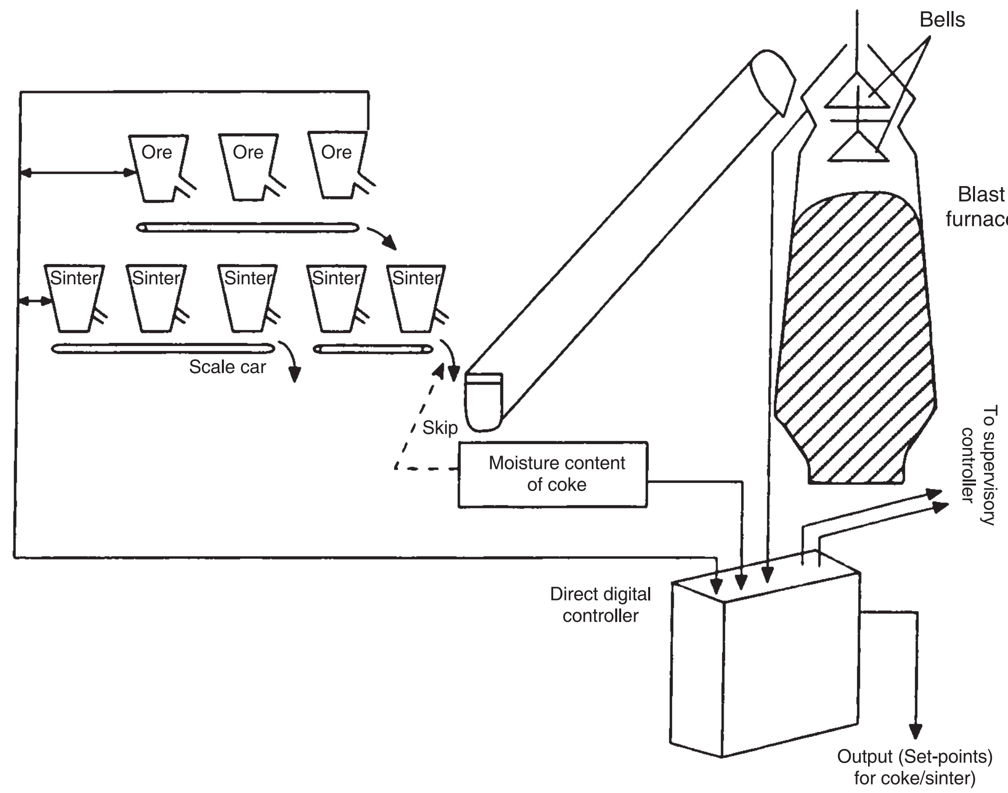


Figure 12.49 Blast furnace—Stock houses and charging control.

downtime can be achieved by using increased information availability. Several models may be used to calculate the relevant parameters. These are

1. Gas distribution from *above-burden* probes and from *below-burden* probes to periodically indicate the current gas flow pattern in the furnace. This could be presented on a graphical display;
2. Thermal and reducing conditions inside the blast furnace using closed heat and material balances for six major elements: iron, carbon, oxygen, hydrogen, nitrogen and coal dust;
3. Position of each charge within the furnace volume and displacing it by volume with subsequent charges. This tracking of charge position is used to determine burden coke rate and basicity at various levels within the shaft of the furnace;
4. Flame temperature and production rate within the tuyere zone:
5. Tracking of hot metal and slag quantities formed in the furnace hearth to determine the changes in production rates and the number of times of opening and closing of tap holes, etc.

(ii) *Input da ta required:*
— Metallic burden chemical analysis
— Metallic burden size analysis

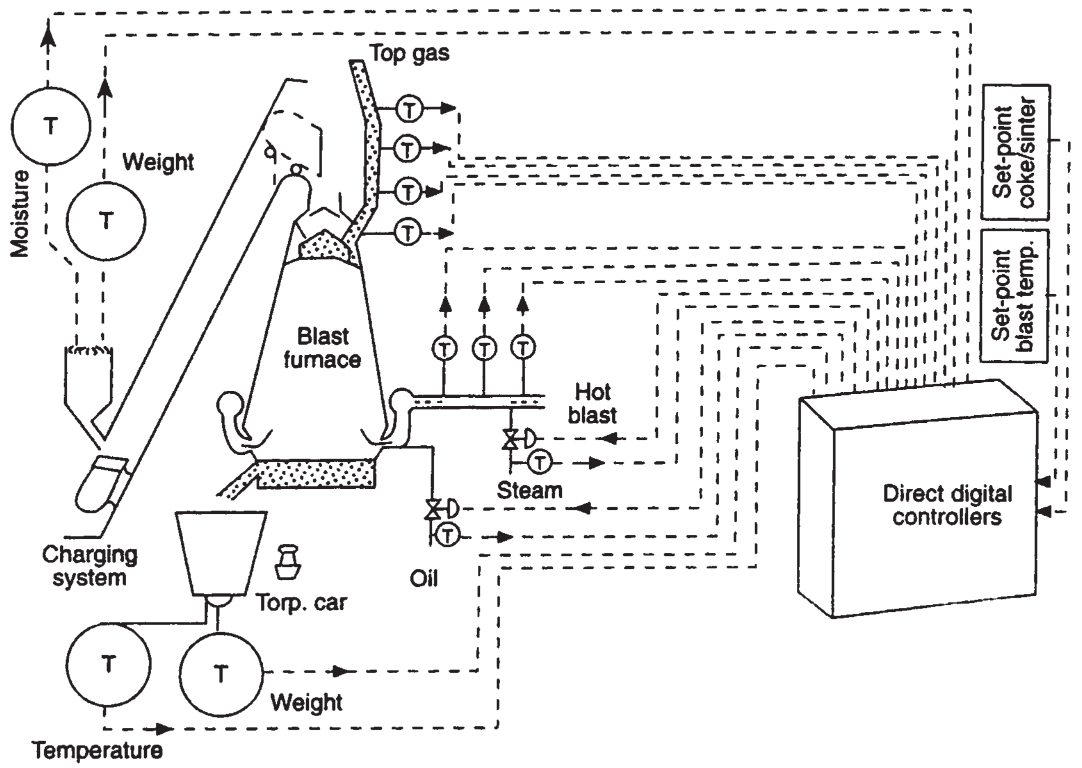


Figure 12.50 Blast furnace—Furnace operations control.

— Flux materials chemical analysis
— Flux materials particle size analysis
— Coke chemical analysis
— Coke particle size analysis
— Porosity of all solid materials
— Material identification by bin location
— Basic charging scheme to be followed.

II. (a) Direct digital controller—Stock house and charging (Figs. 12.49 and 12.50)

(i) *Functions:*

— Sequencing logic and monitoring for skip hoist, bells and/or chute for each furnace
— Sequencing logic and monitoring for scale car and hopper operation for each furnace.

(ii) *Control tasks:*

— Operation of skips hoist mechanism
— Sequencing of furnace bells and operation of stock distribution mechanism

— Sequencing of charging cars under furnace raw material hoppers and filling of skips to meet accepted furnace charging strategy
— Monitoring and filling of raw material hoppers
— Accumulation of amounts of raw materials charged and transmission to higher level machines for reordering of raw materials
— Control of the skip hoist, bell and chute system
— Scale car loading and operating logic control
— Control of stockline level
— Control of stock distribution at top of furnace.

(iii) *Input data required:*
— Locations of skip 1 and 2
— Positions of small and big bells
— Position of scale car in loading alley of stock house
— Weight of charging car and contents at each hopper
— Weight of material in each hopper
— Position of charging material distribution mechanism
— Stock level in furnace
— Infrared camera input
— Infrared camera position
— Coke and sinter moisture contents
— Other additives moisture content
— Top temperature
— Coke charging weight.

(iv) *Output parameters controlled:*
— Movement of skip hoist
— Movement of small bell
— Movement of big bell
— Movement of distribution mechanism
— Movement of scale car or belt
— Dumping of raw material hoppers
— Position of upper-burden probe
— Position of lower-burden probe
— Infrared camera position.

II. (b) Direct digital controllers—Furnace operation and stoves (Figs. 12.51 and 12.52)

(i) *Functions:*
— Blast furnace heat and mass balances for each furnace
— Sequencing logic for positioning of metal and slag ladles
— Sequencing logic for stoves for each furnace
— Control of gas cleaning equipment for each furnace.

(ii) *Control tasks:*
— Cyclic or staggered parallel operation of stoves for heating intake blasts
— Control of fuel addition to blast

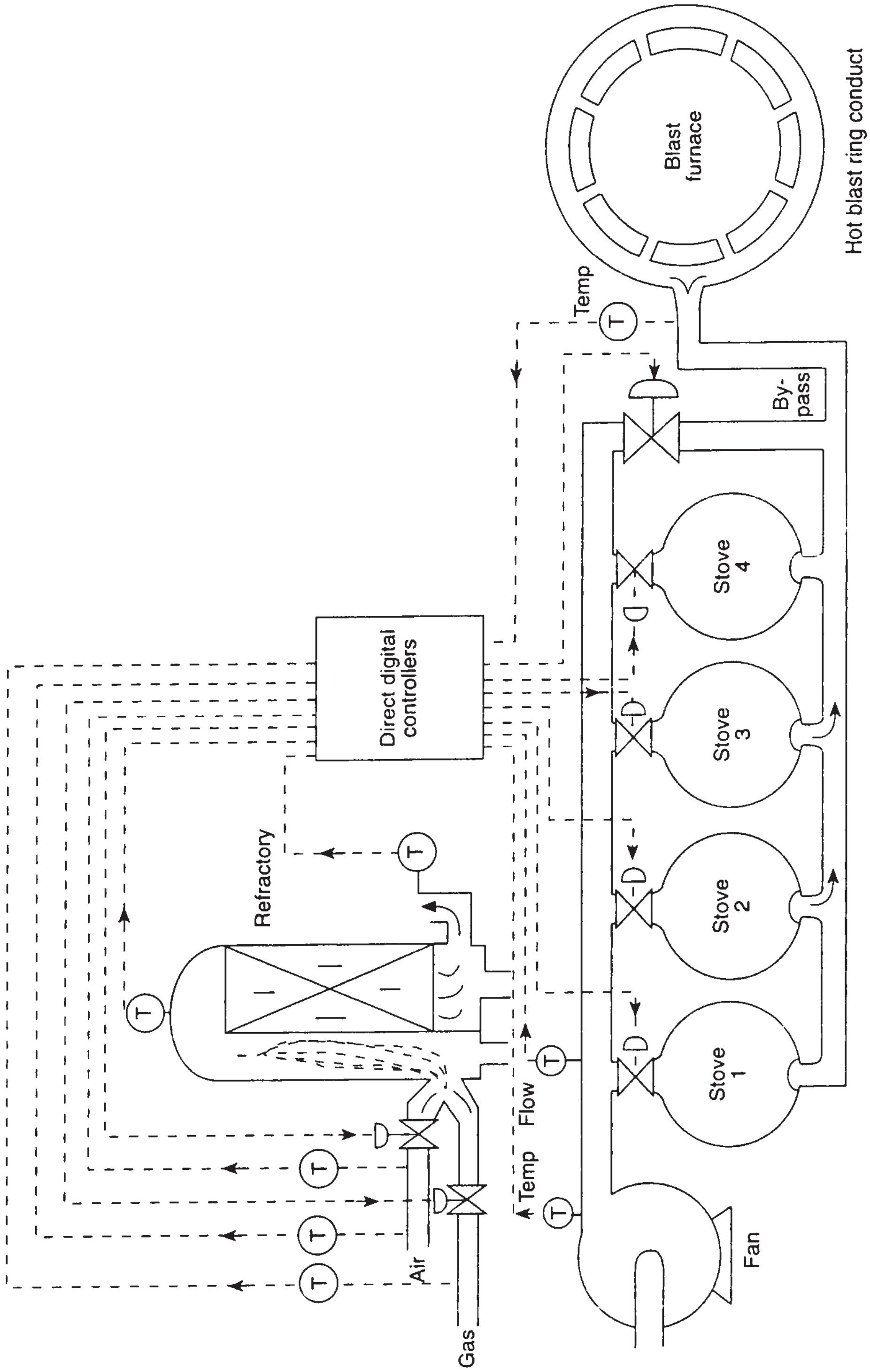


Figure 12.51 Stoves and blast heating control.

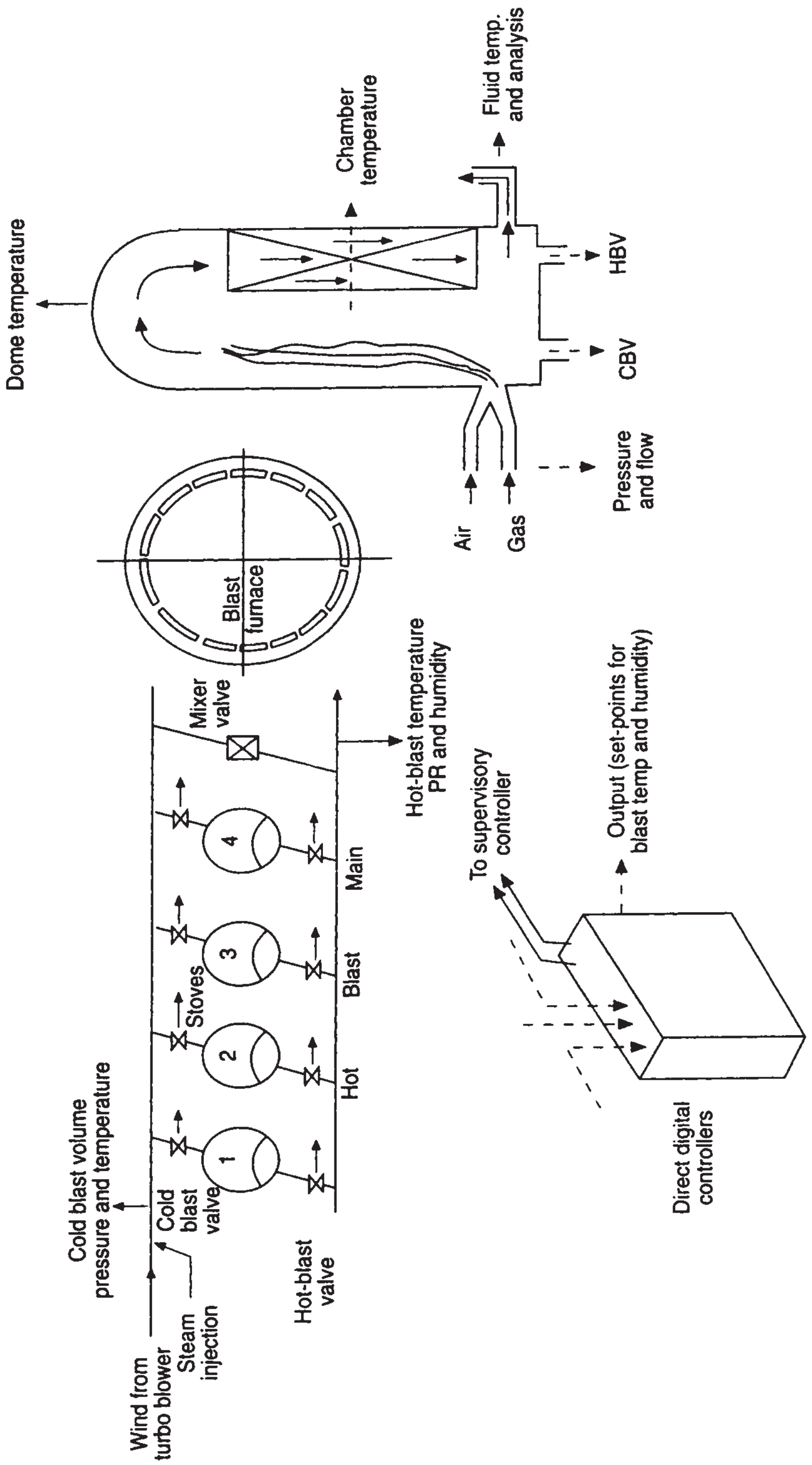


Figure 12.52 Additional details of stoves and blast heating control.

— Control of oxygen addition to blast
— Control of energy addition to stoves
— Monitoring and control of furnace cooling
— Control of blast pressure and flow rate
— Control of moisture addition to blast
— Control of blast temperature
— Monitoring and control of thermal condition of the hearth.

(iii) *Input data required:*

— Gas temperature at entrance and exit of each stove
— Dome temperature of each stove
— Checker chamber temperature of each stove
— Fuel flow rate at each stove
— Air flow rate
— Combustion gas analysis (CO, CO_2, H_2)
— Furnace top gas flow rate
— Dust rate out of furnace
— Furnace top gas pressure
— Furnace top gas analysis (CO, CO_2, H_2, N_2)
— Gas temperature at top of furnace (off take)
— Stock level in furnace
— Position of upper and lower burden probe
— Temperature at upper and lower burden probe
— Pressure at upper and lower burden probe
— Gas analysis at upper and lower burden probe
— Furnace blast flow rate
— Position of all diverting valves or vanes
— Furnace blast oxygen flow rate
— Furnace hot blast temperature
— Furnace blast moisture addition flow rate
— Blast moisture content (before and after the addition of moisture)
— Temperature of molten slag and molten iron at the exit of furnace
— Position of hot metal ladle under iron runner
— Weight of hot metal ladle car and contents
— Temperature of coolant water at entrance of furnace
— Temperature of furnace coolant at exit of each section
— Coolant flow in each section
— Power input to electrostatic precipitator
— Water flow to spray washer
— Ambient temperature at furnace site
— Furnace wall temperature at each section
— Fuel additive temperature at input
— Identification of ladle cars

— Furnace cold blast temperature
— Stove gas analysis
— Tuyere coolant outlet temperature
— Status and position of tapping drill
— Position of oxygen lance
— Oxygen flow rate
— Status and position of mudgun
— Furnace cold blast pressure
— Total steam flow
— Steam flow for blast humidification
— Steam flow for gas seal
— Gas seal pressure
— Differential water flow for each tuyere
— Foundation temperature
— Dust level in dust catcher
— Furnace output gas rate
— Pressure of raw gas, semi clean gas and clean gas
— Make-up water pressure and flow for slag granulation
— Total water pressure and flow for slag granulation
— Weight of granulated slag at loading point
— Moisture of granulated slag
— Chemical analysis of iron output
— Sulphur analysis and heating value of fuel additives
— Composition of dust from furnace
— Chemical analysis of slag
— Weight of dust and sludge recovered at gas cleaners.

(iv) *Output parameters controlled:*
— Positioning of gas diversion valves of stoves
— Stove fuel flow
— Combustion air flow
— Blast air rate, fuel rate, oxygen rate
— Operation of tapping tools and mudgun
— Positioning instructions for hot metal transfer cars
— Computed molten iron level
— Computed slag level
— Power to precipitators
— Water flow to spray washer
— Control of moisture addition to blast.

12.6.4 Steel Zone

As mentioned earlier, the steel zone in a steel plant consists of LD Converters; Open Hearth Furnace; Electric Arc Furnace; Continuous casting, etc.

We shall describe here LD Converters and Continuous Casting and their control strategy as all modern steel plants are equipped with these units.

LD converters

The LD converter shop consists of following:

- — Material Handling and Storage Section
- — Mixers
- — Converters.

In *material handling and storage section* bulk material is stored in overhead bins from where measured quantities can be charged into the converter through chutes. Metallic scrap is loaded in special containers and is charged with the help of charging cranes. Hot metal mixers are provided at one end of the shop.

The *converters* have detachable bottoms. An elaborate system is provided for cleaning and recovering the gases emanating from the converters. The recovered gas is expected to have a calorific value of 1900–2000 Kcal/m^3. The supervisory controller is envisaged to calculate the charge, flux and oxygen requirement of the process.

Figures 12.53 and 12.54 present sketches of elements of LD converter computer control schemes. In order to gain the maximum production advantages offered by the speed of the LD process, precision control of end-point carbon and temperature is required to prevent time-consuming reblows or cooling procedures.

Because of the speed of the process, it is imperative that the weights and chemical analysis of the materials charged to the furnace are known. In the area of scrap, hot metal and fluxes, the weights and chemical analysis should be accurately known, if control of end-point carbon and temperature is to be achieved.

Ladle metallurgy facility allows steel sulphurisation, ladle stirring, and alloy additions. Prime process inputs to this facility are steel specification and final chemical analysis desired. The system calculates synthetic slag and argon bubbling necessary to remove sulphur. Additional calculations are performed to determine reheat time resulting in chemical losses and necessary alloy additions to bring the heat within specified grade, temperature and casting constraints.

I. Supervisory controllers (Fig. 12.53)

Functions:

- — Grade monitoring for LDs
- — Sequencing of charging orders for LDs to ensure schedule and grade maintenance
- — Interaction with auxiliary equipment and other area computers for database generation for individual LD converter computers.

II. Direct digital controllers (Fig. 12.54)

(i) *Functions:*

- — Monitoring of the charging cycle conforming to desired grade
- — Adherence of course of blow and prediction of end point of oxygen addition

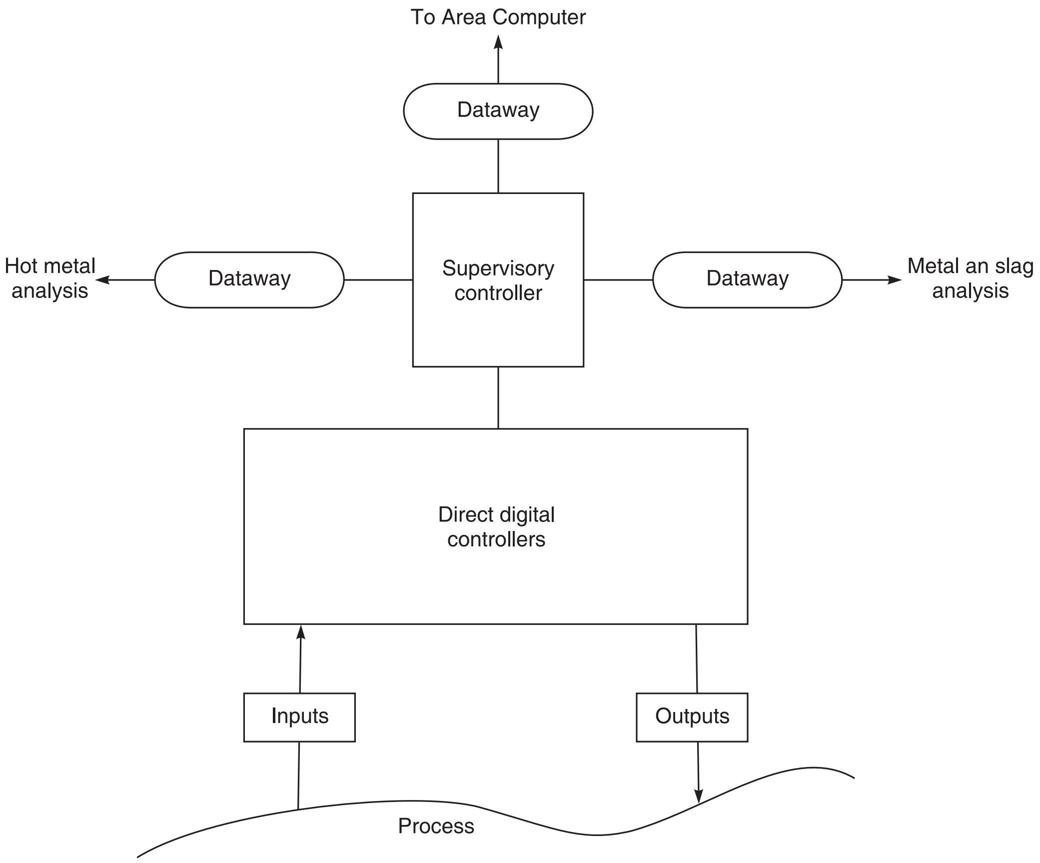


Figure 12.53 LD converter supervision and control.

— Monitoring of pouring cycle in response to plan of supervisor
— Sequencing logic for operation of LD's and related equipment
— Sequencing logic for positioning of moulds during pouring
— Prediction of oxygen flow requirements from steady-state model of LD process
— Computation of existing "Ladle" metal temperature
— Monitoring of hot and cold repairs.

(ii) *Control tasks:*

— Operation of all controls of the LD converters during a cycle
— Develop predictive model for analysis and temperature of hot metal
— Accurate monitoring of addition of all materials to the LD converters while charging
— Calculation and control of rate and amount of oxygen addition and scrap requirement using Steady State/Dynamic model
— Computation of rate of melting of scrap to ensure uniform product metal
— Computation of heat loss of iron in ladles from blast furnace

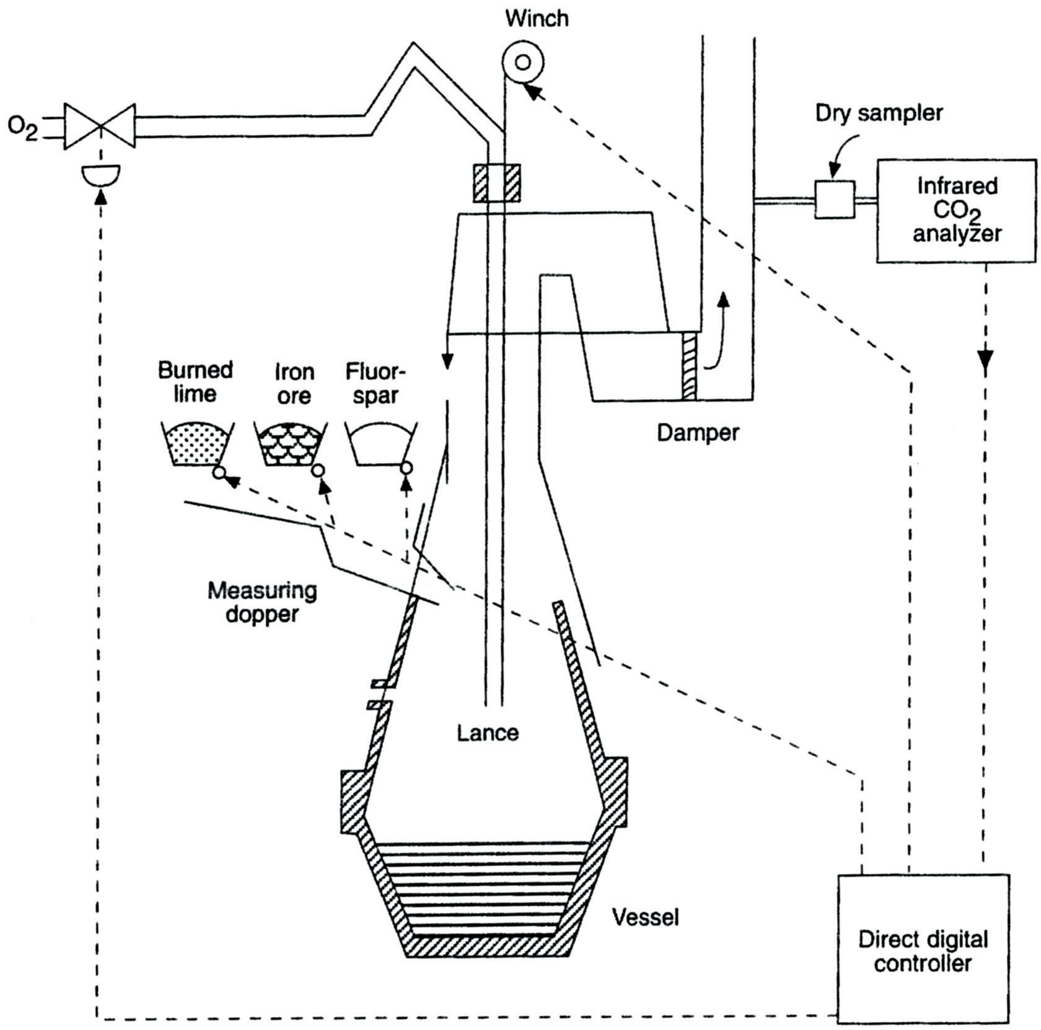


Figure 12.54 Some LD converter control elements.

— Monitoring of ladle repair and reconditioning
— Sequencing logic for operation of LD's and positioning of ladles during pouring
— Monitoring of alloy addition in ladles
— Monitoring of the argon rinsing operation
— Data acquisition and control of operation of the gas cleaning plant.

(iii) *Input data required:*

— Positions of oxygen lance and LD vessel
— Hood position
— Oxygen amount, temperature, pressure and analysis
— Ladle identification
— Weight of ladle, scrap box or hopper

- — Position of ladle or scrap box (addition and pouring)
- — Final hot metal analysis and input temperature at the ladle
- — Slag content of hot metal input (estimate)
- — Scrap temperature
- — Furnace off-gas pressure
- — Fuel gas analysis (O_2, CO, CO_2, and H_2)
- — Coolant temperature at in and out of lance
- — Coolant flow
- — Bath temperature
- — Carbon content
- — Location of ladles
- — Additives additions to the ladles
- — Fuel flow for heating ladles
- — Mixer contents temperature and level
- — ID fan bearing temperature
- — Position of the change-over valves
- — Monitoring of ignition system
- — Scrap analysis and type
- — Bath metal and slag analysis
- — Fuel gas analysis
- — Input hot metal analysis
- — Additives analysis.

(iv) *Output parameters controlled:*

- — Movement of LD converter vessel
- — Movement of the lance
- — Movement of the crane to position ladle
- — Valve position oxygen flow
- — Valve position lance water flow
- — Movement of the hood
- — Control of batch hopper
- — Movement of sublance
- — Scrap addition control
- — Ladle addition control
- — Flux, fluidizer, carburizer and coolant addition (vibro feeder control).

Continuous casting

A continuous casting shop casts the steel produced from the LD converters. This shop consists of number of slab casters and four-strand bloom casting machine.

The main parts of continuous casting shop are:

- — Argon rinsing bay
- — Tundish preparation bay

— Casting bay
— Maintenance bay
— Discharge bay
— Auxiliary bay
— Slab and bloom storage yard
— Mould repair shop.

Steel from converters is carried in steel teeming ladles with the help of transfer cars via on-line argon rinsing units. The tundish bay has the facilities for relining of tundishes, stopper assembly, and drying and pre-heating of tundishes. In the maintenance bay the withdrawal roll set, dummy bar separation mechanism and gas cutting units are located. In the discharge bay, cut slabs and blooms are discharged on to slab and bloom transfer cars. The storage yard is for receiving, cooling, inspecting, scarfing, programming and storage. In the mould repair shop, used moulds for casting are to be repaired and new moulds are to be assembled.

The operation and control of continuous caster discussed here is mainly based on principles developed by Adams patent and related documents listed there.

I. Supervisory controller (Fig. 12.55)

Functions:

— Schedule operation of the continuous caster to accommodate available metal and produce required slabs or blooms to sizes needed.
— Monitor movement of liquid steel ladles to ensure efficient operation of continuous caster.
— Keep track of identity of slabs or blooms leaving the caster and properly label them for future recovery.
— Compute correct surface temperature profile for control of cooling system.

II. Direct digital controller

(i) *Functions:*

— Regulation of mould temperature through coolant flow control
— Regulation of slab and bloom “Pull”
— Operation of cut-off mechanism
— Operation of marking device
— Tundish level control
— Mould level control
— Cooling spray control
— Monitor guide roll rack condition.

(ii) *Control tasks:*

— Monitor tundish metal level
— Monitor movement of ladles
— Monitor and control ladle pouring to tundish
— Monitor mould level control
— Regulate coolant flows to mould

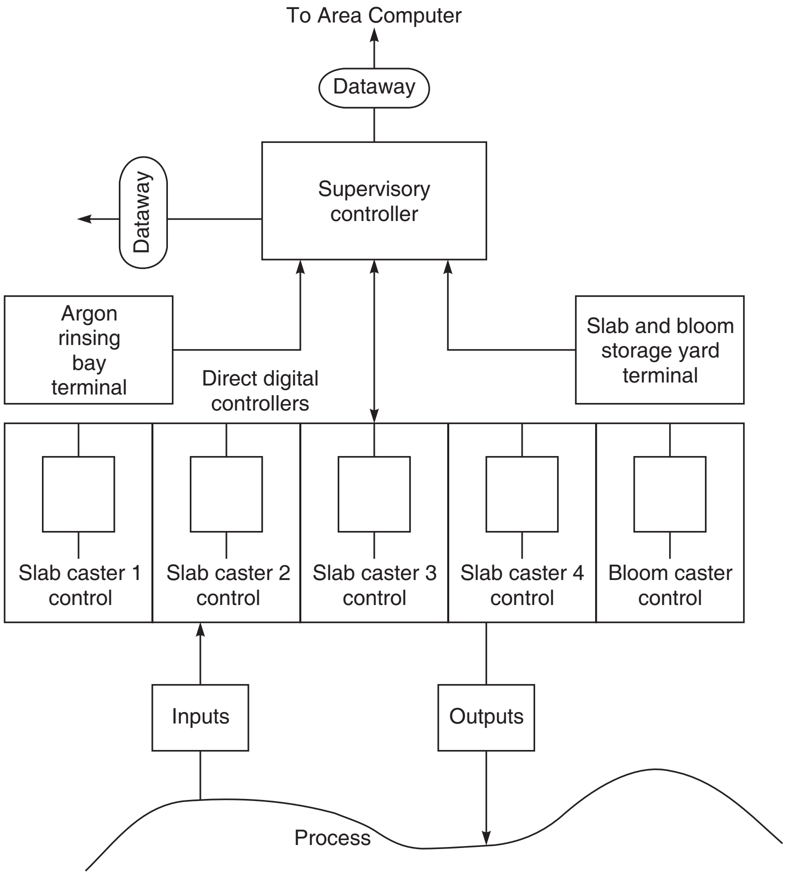


Figure 12.55 Continuous casting supervision and control.

— Regulate "Push-Pull" of slab or bloom string
— Measure slabs or blooms produced and control flame cutting machine
— "Marking" of finished slabs or blooms
— Start-up of caster
— Shut-down of caster
— Emergency response to breakout
— Determination of proper lubricant and/or powder flow
— Monitoring of ladle repair and reconditioning
— Monitoring of tundish repair and reconditioning.

(iii) *Input data required:*

— Temperature of metal in ladle
— Mould coolant temperature in and out and flow rate
— Strand surface temperature
— Spray coolant flow rate and inlet temperature
— Tracking input

— Position of flame cutting machine
— Tundish stopper position
— Ladle weights
— Liquid level
— Pinch roll speed and power input
— Ladle position and ladle stopper position
— Flow rate of inert gas to ladle
— Tundish metal temperature
— Flow rates of inert gas to tundish and mould
— Pinch roll gap settings
— Mould water pressure
— Spray coolant pressure
— Mould lubricant and/or additives flow rate
— Tundish position
— Main spray coolant pressure
— Guide roll rack forces
— Slab and bloom identification information
— Weight of metal in ladle
— Expected time of arrival of next heat
— Dimensions of slab and bloom being poured

(iv) *Output parameters controlled:*
— Coolant flow rates, at each location
— Positioning of tundish stopper
— Speed of pinch rolls
— Positioning of ladle stopper
— Mould oscillation input
— Pinch roll gap setting
— Lubricant and/or powder flow rate
— Cut-off system operation
— Identification of cut-off slabs or blooms
— Positioning of ladle
— Positioning of tundish.

12.6.5 Mill Zone

The mill zone consists of:

— Soaking pits
— Blooming mill
— Billet mill
— Wire rod mill
— Plate mill
— Slab mill.

Soaking pits and blooming mill

Heating of ingots and soaking is done in soaking pits. When ingots have cooled down sufficiently, they are stripped. However the interior of the ingot is still liquid and temperature distribution therefore is unequal. Before the ingots can be rolled into blooms, they should have a very well distributed temperature of about 1300°C. Therefore, they are heated in soaking pits for four to ten hours, depending on steel quality, ingot type and raked time. During the soaking process, the outerside of the ingot is reheated while the interior cools down and solidifies.

Each pit is heated by two burners situated in the upper part of the lateral walls of the pit. The fuel burnt is a mixture of coke oven gas and blast furnace gas with a calorific value of 1450 Kcal/cu m. Air is preheated in ceramic recuperators and the gas in metallic recuperators. The ingots are placed in the pits in such a way that proper circulation of flame around them is ensured.

Before putting ingots into the pit, coke breeze is placed at the bottom.

The blooming mill is designed to roll steel ingots into blooms for the continuous billet mill and the rail and structural mill.

Figures 12.56 and 12.57 show the control strategy for soaking pits and blooming mill.

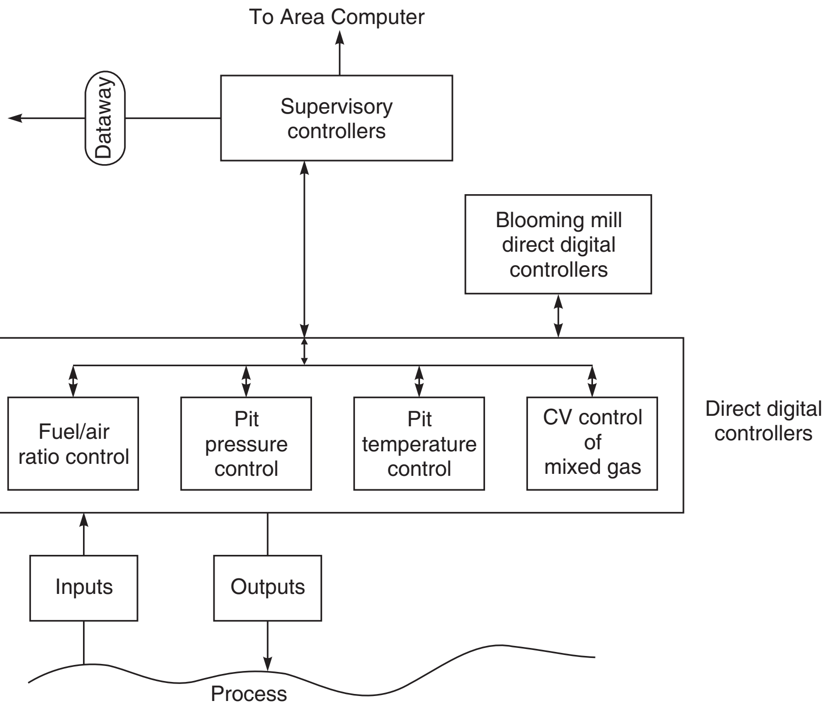


Figure 12.56 Soaking pit supervision and control computer system.

I. Supervisory controller

(i) *Functions:*

— Optimisation of soaking pit utilisation and fuel usage

— Scheduling of soaking pit and blooming mill as a unit

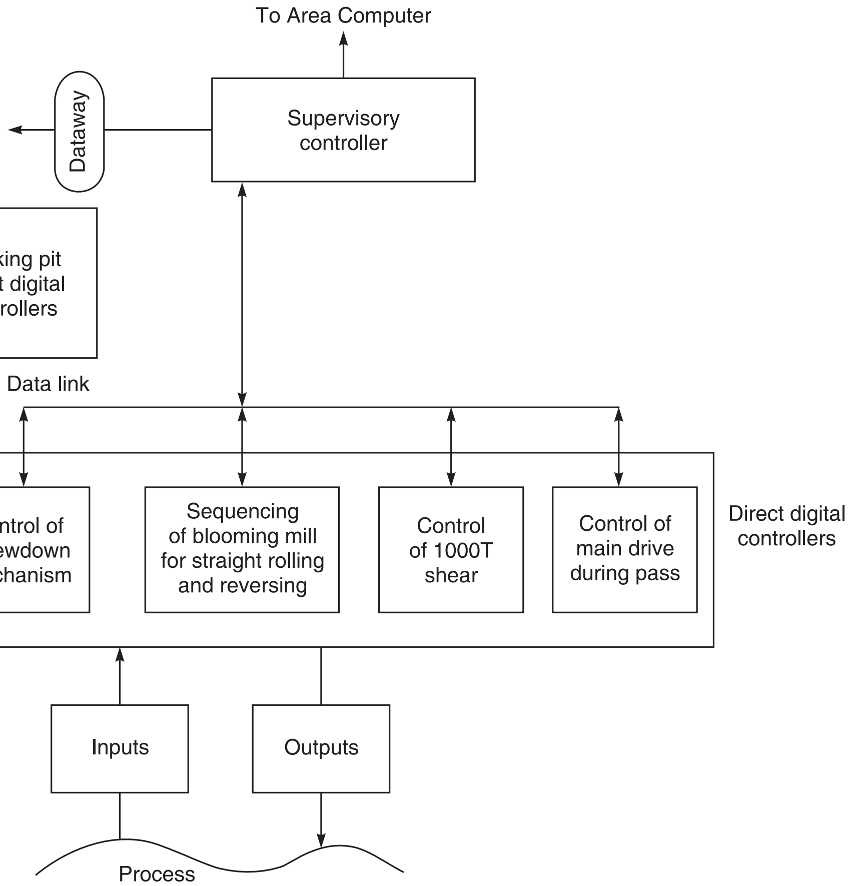


Figure 12.57 Blooming mill supervision and control computer system.

— Determine the proper time to strip ingots to retain maximum heat while preserving structural integrity
— Maintain record of cold ingot inventory including mapping
— Tracking of all ingots and resulting blooms
— Recuperator leakage model
— Short and medium term repairs of soaking pits.

(ii) *Input* data *required:*
— Identity of mould
— Metallurgical characteristics
— Teem time
— Pouring temperature
— Order of ingot teeming
— Estimated arrival times of trains at stripping yard
— Order of ingot stripping

— Stripping time
— Estimated arrival time of heat at the soaking pits.

II. (a) Direct digital controller—Soaking pits (Fig. 12.56)

(i) *Functions:*

— To apply a model to determine minimum cost (time, fuel and metallurgical quality) to bring ingots to proper temperature end-point
— To maintain proper fuel/air ratio at each burner to ensure maximum energy recovery, fuel efficiency and reduced scale formation
— Instructions to operate cranes to strip ingots, to charge pits and to remove ingots for rolling
— Maintain identity of ingots and transfer same to the blooming mill computer
— Monitoring of condition of pits and recuperators for maintenance scheduling, etc.

(ii) *Control tasks:*

— Regulation of fuel flow
— Control of air to maintain fuel/air ratio
— Crane control.

(iii) *Input data required:*

— Fuel flow in each pit
— Air flow in each pit
— Status of each pit
— Pit temperature and pressure
— Mould train position and identification and status
— CV value of the fuel gas
— Fuel gas analysis
— Input air temperature
— Ambient air temperature
— Ingot identification, stripping time, charging time, and buggy position
— Fuel temperature
— Recuperator temperature
— Identity of ingots and associated data to be carried forward
— Ambient temperature of "cold" ingots charged.

(iv) *Output parameters controlled:*

— Fuel flow rate of each pit
— Air intake setting of each pit
— Instructions for crane movement
— Pit pressure of each pit.

II. (b) Direct digital controller—Blooming mill (Fig. 12.57)

(i) *Functions:*

— Computation of roll gap setting for each pass of blooming mill
— Sequence of blooming mill rolling
— Operation of 1000 T shears

— Operation of conveyer
— Operation of bloom yard cranes
— Control of hot scarfing
— Detection of ‘pipe” and other defects in blooms.

(ii) *Control tasks:*
— Tracking of all ingots and resulting blooms
— Control of blooming mill during each pass
— Set-ups of blooming mill between the passes
— Sequencing of blooming mill through reversing cycles
— Control of roll tables before and after the mill stand
— Control of manipulator and tilter operation
— Control of hot scarfing
— Operation of 1000 T shears
— Stopper movement after 1000 T shears
— Stamping or marking of finished blooms for identification.

(iii) *Input data required:*
— Hot metal detector input
— Roll gap setting
— Ingot location orientation and temperature before and after rolling
— Roll force applied, roll speed and roll torque
— 1000 T shear position
— Manipulator current
— Oxygen flow to hot scarfer
— Fuel flow to hot scarfer
— Position of bloom for scarfing
— Bloom length and weight
— Water flow, roll cooling
— Drive motor power—main rolls
— Screw down motor currents
— Manipulator position
— Tilter motor current
— Roll table speeds
— Monitoring of temperatures of main drive motors and bearings
— Weight, analysis, and dimensions of ingots to be rolled
— Ingot temperature
— Final bloom dimensions desired
— Scarfing practice
— Metallurgical practices and restrictions
— Adaptive factor determined from first scale breaking pass.

(iv) *Output parameters controlled:*
— Roll gap setting, main rolls
— Ingot position

— Water flow rate
— Roll force to be applied
— Roll velocity and direction, main rolls
— Side guide manipulator settings
— Tilter setting
— Scarfer oxygen flow
— Scarfer fuel flow
— Table roller velocity and direction
— Movement of 1000 T shear
— Stamping or marking.

Other mills

The supervisory controller system functions and tasks performed by direct digital controllers are similar to Blooming Mill and are not presented here.

12.6.6 Utility Zone

Utility zone is very important part of steel plant and any automation should consider this zone along with other zones in order to achieve maximum benefits. We are not describing this automation strategy as their description would require a separate book. Through this book and other books and journals mentioned in suggested reading, the development of control strategy for any application will be an easy task.

12.7 CONCLUSIONS

Operation of industrial utility systems emphasise on reliability, safety and minimising costs. Due to inherent complexities of industrial units, achieving all the three objectives by applying human judgment becomes extremely difficult, if not impossible.

With the judicious application of advanced Distributed Control System, it is possible to ensure all the three parameters. From a functional viewpoint, modern distributed computer systems can perform all the conventional control, data indication/recording functions, and others like advanced interactive control, integrated regulatory, logic and sequence functions, alarm analysis, logs and report generation, performance evaluation, optimisation, preventive maintenance etc.

SUGGESTED READING

Asakawa, Y. et al., New development of rolling mill control systems, *Hitachi Rev.,* **34,** No. 4, pp. 181–186, 1985.

Chen, C.T., Serizawa, Y. and Hasegawa, K., Raw mix proportioning system for the Taiwan cement corporation, *Hitachi Rev.,* **32,** No. 6, pp. 297–302, 1983.

Dorei, V.S., Bhati, B.H. and Ullal, J.A., An overview of applications of DDC systems for thermal power plants, *ISDC* '86, *Int. Sem. on Distr. Control,* New Delhi, 1986.

Fahrny, J.C. and Zeta, A., Man-machine interface for control of cement production, *Brown Boveri Rev.,* **7,** pp. 335–361, 1985.

Flotho, G. and Scharpenberg, H., Automation in the water industry with the control system PROCONTROL I, *BBC-Brown Bowery Brochure on PROCONTROL I,* 1983.

Frech, R. et al., Modern methods of automating power system control, *Brown Boveri Rev.,* **70** (1/2), pp. 36–47, 1983.

Ganguly, G., Productive automation in cement plant, *ISDC '86 Int. Sem. on Distr. Control,* New Delhi, 1986.

Honda, T. and Shimizu, I., The most advanced control techniques for steel processing lines, *Hitachi Rev.,* **32,** No. 2, pp. 83–88, 1983.

Ishiyama, M. et al., Computer systems communications in a steel works with a standardized network technology, *IFAC Symp. on Digit. Comp. Appls. to Proc. Control,* Vienna, pp. 587–593, 1985.

Iso, H. et al., Computer control systems applied to iron and steel making, *IEEE Contr. Syst. Mag.,* **7,** No. 5, pp. 3–9, 1987.

Kaji, A., Computer control system for thermal power plants, *Hitachi Rev.,* **32,** No. 6, pp. 287–290, 1983.

Kashiwagi, M., Mori, S. and Harada, T., Computer control system for water and wastewater treatment plants, *Hitachi Rev.,* **27,** No. 3, pp. 146–152, 1978.

Kawano, M. et al., Computer control systems applied to iron and steel plants, *Hitachi Rev.,* **27,** No. 3, pp. 139–145, 1978.

Khan, Z.U. and Mohan, M., Distributed digital control system for power plants, *ISDC '86, Int. Symp. on Distr. Contr.,* New Delhi, 1986.

Lioka, M. and Setsu, N., Modern monitoring and control system HIACS-3000 for power plants, *Hitachi Rev.,* **35,** No. 5, pp. 271–274, 1986.

Mitsuoka, H. et al., Recent computer control systems for blast furnaces, *Hitachi Rev.,* **27,** No. 1, pp. 45–50, 1978.

Mori, S. et al., AQUAMAX-80, A new distributed control system for water and wastewater systems management, *Hitachi Rev.,* **29,** pp. 237–242, 1980.

Nakayama, M. et al., Advanced computer control system for steelmarking plants, *Hitachi Rev.,* **32,** No. 6, pp. 291–296, 1983.

Nigwara, W., Matsumara, J. and Nakata, A., Application of process control computers to thermal power plants, *Hitachi Rev.,* **27,** No. 3, pp. 133–138, 1978.

Nishitomo, S. et al., Recent computer control technology in the steel industry, *Hitachi Rev.,* **34,** NO. 4, pp. 194–200, 1985.

Nitta, Y. et al., Computer system for thermal power plant, *Hitachi Rev.,* 25, pp. 277–282, 1976.

Purkayastha, P. and Pal, J.K., Design considerations of distributed control architecture for a thermal power plant, *IFAC Syrnp. on Distr. Comp. Contr. Syst.,* pp. 69–77, 1985.

Sahani, S.K., Automation and distributed digital control in cement industry, *ISDC '86, Int. Sem. on Distr. Control,* New Delhi, 1986.

Skideth, P. and Smidth, F. L., Automatic kiln control at Oregon porland cement company as Durkee plant utilizing fuzzy logic, *Technical Note,* Paul Gramstrup ApS, Denmark, 1981.

Takagi, T., Shinomiya, F. and Miyaoka, S., Distribution network control for water supply system, *Hitachi Rev.,* **32,** No. 5, pp. 259–264, 1983.

Tanaka, S., Sakai T., Hirayama, H. and Takaoka, H., Distributed control system for combined cycle power plant, *IFAC Proc. on Distr. Comp. Contr. Syst.,* 1983.

Thorp, J.S., Control of electric power systems using real-time measurements, *IEEE Contr. Syst. Mag.,* **9,** No. 1, pp. 39–45, 1989.

Williams, R.V., *Control and Analysis in Iron and Steel Making,* Butterworths, London, 1983.

Yoshitani, Y., The background and present status of computer usage in the Japanese iron and steel industry: State-of-the-art, *Computers in Industry,* No. 1, pp. 263–275, 1980.

CHAPTER

13

Intelligent Controllers

13.1 INTRODUCTION

We have already discussed PID Control Strategy and the ways in which it can be represented in a computer. The relative advantages of velocity and position algorithms have also been discussed. In this chapter, we shall discuss the disadvantages of classical controllers and briefly describe adaptive and optimal controllers, artificial intelligent systems, expert controllers, fuzzy controllers and finally neural controllers. This chapter is introductory and is not intended to discuss the design of controllers in detail. The discussion in this chapter is confined to appreciation of these technologies, their advantages as well as their current status.

Figure 13.1 shows the conventional PID controllers, which has already been described in detail in Chapters 0, 1, and 6.

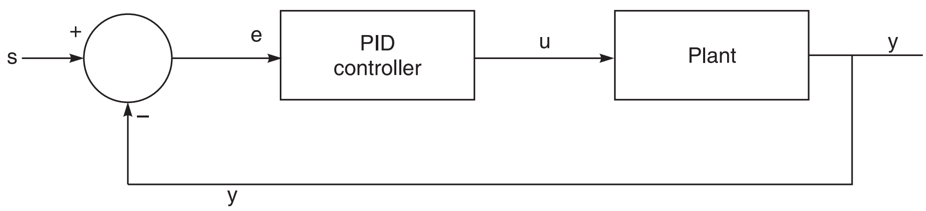


Figure 13.1 PID control strategy.

The aim of PID controller is to bring error *'e'* to zero by manipulating control signal *u*. However, for effective control which can take care of variations in load variables and other system parameters, the knowledge about the system and its behaviour in different conditions should be considered. For effective computer control, the system knowledge in the form of model or heuristics needs to be generated and stored.

13.2 MODEL BASED CONTROLLERS

We have already described the need and utility of computer models for plant automation in Chapter 11. In the computer model the system dynamics would be represented by model parameters and model equations. The model based control strategies may be classified into one of the two categories, viz. optimal or adaptive.

13.2.1 Adaptive Controller

Adaptive control involves modifying the control law used by a controller to cope with the fact that the parameters of the system being controlled are slowly time-varying or uncertain. For example, as an aircraft flies, its mass will slowly decrease as a result of fuel consumption; we need a control law that adapts itself to such changing conditions. The control of certain variables such as position and velocity of a dynamic system, and their derivatives, is required in various applications. Variations in the characteristics of the system being controlled and changes in the external conditions complicates the control requirements in many cases. High-speed military aircrafts, missiles, motor drives, chemical plants, rolling mills, and voltage regulators are a few examples of applications where a system output must be maintained in the presence of changing dynamic characteristics such as wind, terrain, frequency, voltage, inertia, component aging, loading, and temperature.

An adaptive control system measures certain Index Performance (IP) using the inputs, status and outputs of the system to be controlled. The desired IP value is stored in the controller. The controller also calculates the IP of system based on inputs, outputs and measured perturbations. The controller has built-in adaptation mechanism which modifies the controller parameters, so that the computed IP value may come closer to desired IP. Figure 13.2 shows the generalized mechanism of adaptive control system.

There are two main approaches to adaptive control.

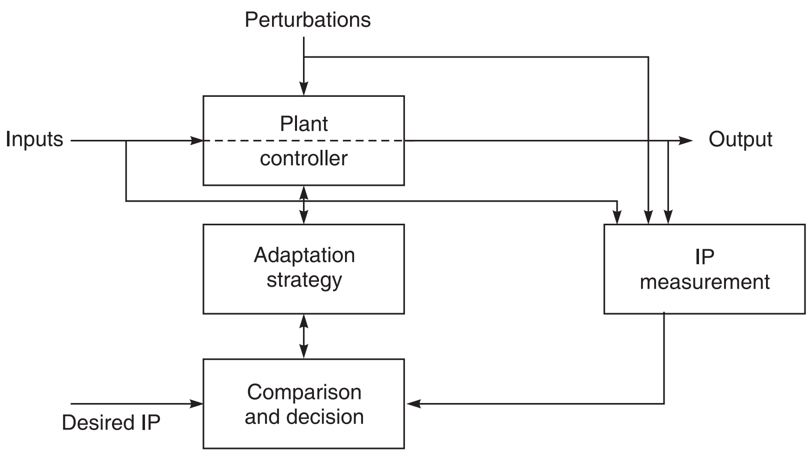


Figure 13.2 Adaptive control—Generalised mechanism.

Model reference adaptive control

The model reference adaptive controller has a system model to which all the inputs and known perturbations are applied in parallel to the actual system. The output of reference model (ym) and output of system (yp) are compared.

The general idea behind Model Reference Adaptive Control (MRAC), also known as Model Reference Adaptive System (MRAS) is to create a closed loop controller with parameters that can be updated to change the response of the system. The output of the system is compared to a desired response from a reference model (Fig. 13.3). The control parameters are updated on the basis of this error. The goal is to ensure that the parameters converge to ideal values that cause the plant response to match the response of the reference model.

MRAC consists of a reference model, chosen at the discretion of the developer, which provides an effective and flexible means of specifying desired closed-loop performance characteristics. The parameters of the controller, or the feedback gains, are adjusted in such a way that the errors between the reference model output and the actual system output are minimised. The objective of MRAC is to force the output of the system and its derivatives to be the same as that of the mathematical reference model. The reference model has dynamic characteristics which do not change as the system operates. The actual controlled system, with its changing characteristics, is forced to attain the unchanging characteristics of the model by means of the adaptive control technique.

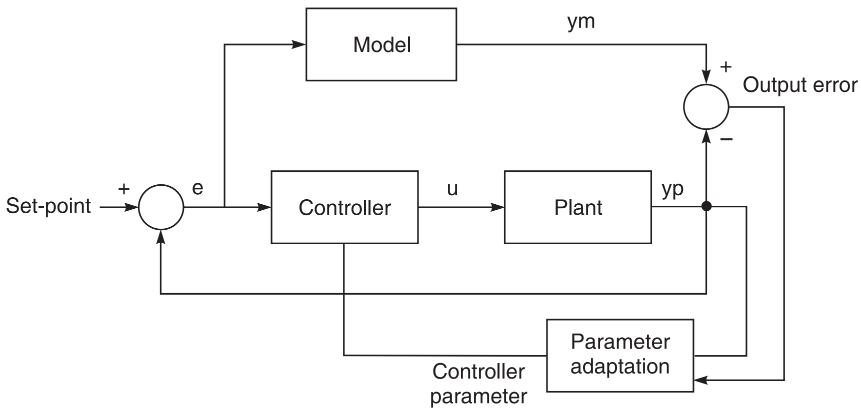


Figure 13.3 Model reference adaptive control.

Model identification adaptive controllers

As against model reference adaptive control, the model identification adaptive control uses a direct approach to implement the adaptive controller.

Model Identification Adaptive Controllers (MIACs) perform system identification while the system is running.

The system identification is performed through two functional blocks, namely, parameter estimator and parameter adapter (Fig. 13.4). The parameter estimation functional block

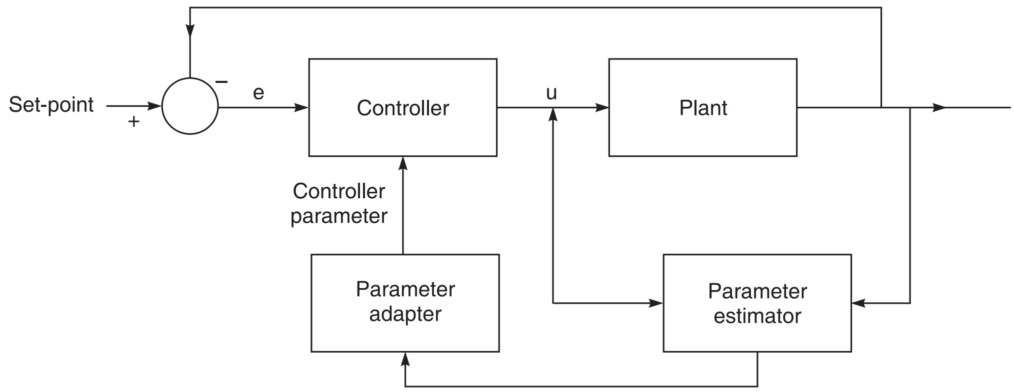


Figure 13.4 Model identification adaptive (Self tuning controller)

continuously estimates the system parameters based on system input and output. The parameter adapter functional block continuously adjusts the control parameters, so that Performance Index is maximised. Following are the methods that can be used for parameter estimation:

— Least square method
— Method of instrumental variables
— Maximum likelihood method
— Stochastic approximation method.

Self-Tuning Control

Self-tuning control represents one branch of model identification adaptive control that has been successfully applied in practice. Controller design requires knowledge of the plant to be controlled, which is not always readily accessible; self-tuning controllers gather such information during normal operation and adjust the controller designs online as required.

Self-tuning PID controllers simplify matters by executing the necessary tuning procedures automatically. These controllers basically observe the process reaction to a disturbance and set their tuning parameters accordingly.

The techniques for implementation of the self-tuning strategy vary. A common approach to automatic parameter selection involves mathematical modelling of the process using equations that relate the current value of the process output to a history of the previous outputs as well as the inputs applied by the controller. If the model is accurate, the controller can predict the future effect of its present efforts and tune itself accordingly.

For example, a process that reacts sluggishly to a step input can be modelled with an equation that predicts the next output as a weighted sum of just two measurements—the most recent input and the most recent output. A self-tuner can choose the weights in the sum to mathematically fit the model to match the input/output relationship that the process has demonstrated in the past.

Heuristic self-tuners, on the other hand, attempt to duplicate the decision-making process of an experienced operator. They adjust the tuning parameters according to a series of expert tuning rules such as "If the controller overreacts to an abrupt disturbance, Then lower the derivative parameter". This approach is similar to the expert controller approach described in Section 13.5.

Robust Control

Although great strides have been made in resolving the implementation issues of adaptive systems, many control system developers are still not convinced about the long-term integrity of the adaptive mechanism. This led the development of robust control.

Robust control involves, firstly, quantifying the uncertainties or errors in a 'nominal' process model. These may be due to nonlinear or time-varying process behaviours. This will lead to the description of the process under all possible operating conditions. The next stage involves the design of a controller that will maintain stability as well as achieve specified performance over this range of operating conditions. A controller with this property is said to be *robust*.

A sensitive controller is required to achieve performance objectives. Unfortunately, such a controller will also be sensitive to process uncertainties and hence suffer from stability problems. On the other hand, a controller that is insensitive to the process uncertainties will have poorer performance characteristics in that the controlled responses will be sluggish. The robust control problem is therefore formulated as a compromise between achieving performance and ensuring stability under assumed process uncertainties.

Thus, although robustness is a desirable property, and the theoretical developments and analysis tools are quite mature, the application is hindered by the use of daunting mathematics and the lack of a suitable solution procedure.

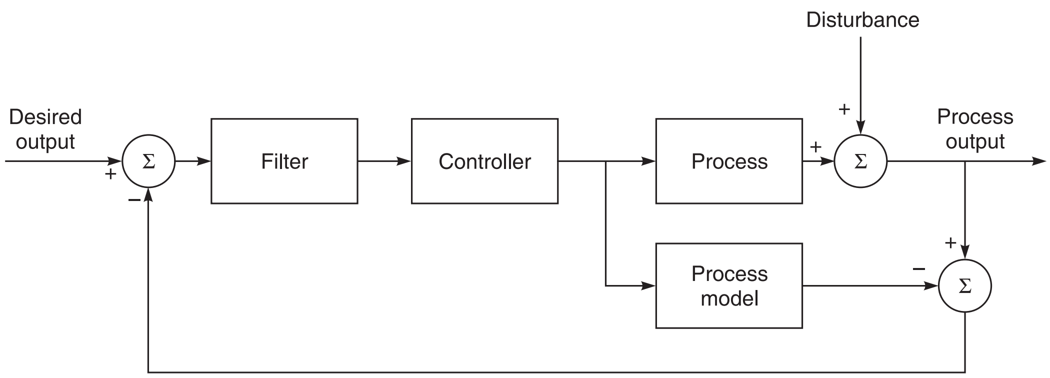


Figure 13.5 Schematic of internal model control strategy.

Internal Model Principle has been used in the design of robust controllers (Fig. 13.5). It implies that unless the control strategy contains, either explicitly or implicitly, a description of the controlled process, either the performance or the stability criterion, or both, will not be achieved. The corresponding 'internal model control' design procedure encapsulates this philosophy and provides for both perfect control and a mechanism to impart robust properties.

13.2.2 Optimal Control

It is evident that some criteria for optimal behaviour should be selected for any system operation. The criteria may be improved quality, increased production, saving in raw material, decreased waste etc. The optirnisation criteria selected for the plant is represented through a performance Index or objective function. The performance index is a measure of system performance that needs to be optimised. However, the constraints of the system in terms of storage capacity, minimum and maximum limit on variables should also be considered. If the process is in steady state then resulting optimum operating point can be achieved using conventional PID control. If the process changes with time, then new optimum set-points need to be computed at each new operating condition. If the process is changing dynamically all the time, then dynamic optimisation technique need to be used. The *steady state* optimisation of system involves the control of state variables for some fixed conditions. The *dynamic* system optimisation on the other hand computes a new optimum each time state variables change. The methods for dynamic system optimisation normally use finite time interval. The new optimum is calculated during this interval. One of the following methods for optimisation may be used:

— Linear Programming
— Non-Linear Programming
— Gradient Method
— Dynamic Programming
— Calculus of Variation
— Kalman Filter.

The Kalman filter is a method for providing an optimal estimate of variables in presence of noise. This approach for filter design has also been used for the design of optimal controller. Kalman filter technique provides recursion formula which can be readily used in computer

Figure 13.6 explains the optimal control of the process.

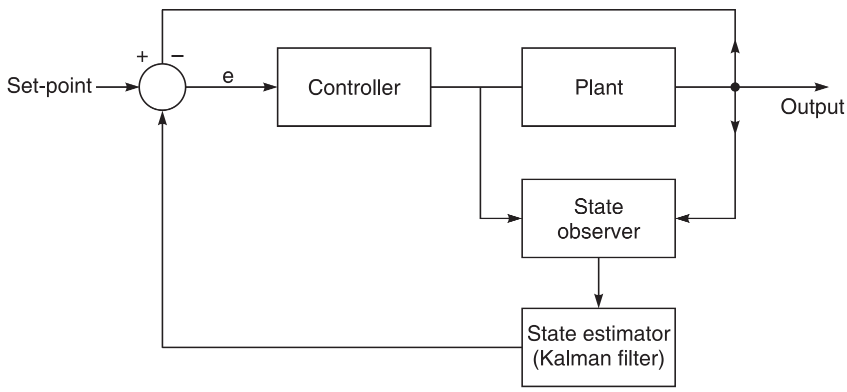


Figure 13.6 Optimal controller using Kalman filter.

Based on the inputs from state observer on the present state of the system, the state estimator determines the optimum control signal which will take the system to optimum performance index.

The parameter settings of a PID controller for optimal control of a plant depend on the plant's behaviour. Therefore, information about the plant is required to tune the PID controller. The tuning methods fall into two broad categories: online model-free methods and methods that build a model of the plant. The former tune the PID controller in a loop with the given plant using one of the optimisation algorithms mentioned before. The second approach builds a model of the plant and accordingly decides the parameters of the controller by using either a deterministic approach or an optimisation method.

Optimal control and its ramifications have found applications in many different fields, including aerospace, process control, robotics, bioengineering, economics, finance, and management science, and it continues to be an active research area within control theory.

13.3 PREDICTIVE CONTROL

The predictive control approach has been shown in Fig. 13.7 The system model is used to predict the future system outputs for different control actions which are based on current and past values of system inputs and outputs. Because of noise and uncertainties this prediction is normally limited to relatively short time. By predicting the future system outputs, the predictive control selects the best control action to avoid undesired deviations. The combination of adaptive and predictive control can be used effectively for unstable processes or reactors.

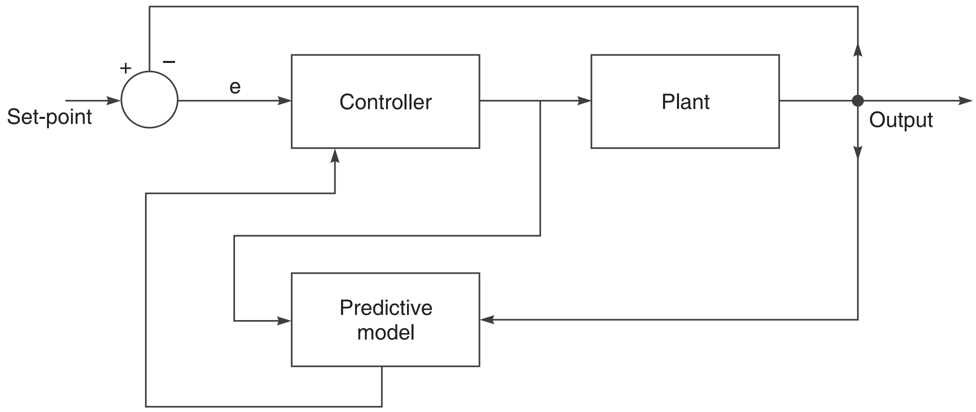


Figure 13.7 Predictive control approach.

Conventional predictive controllers use an "identifier" to determine the mathematical relationship of the system impulse response model, so that the predictive control technique works properly when used with systems having dynamic characteristics which change while they are operating. If the system model dynamics exactly match those of the controlled system, the input predictor will provide inputs to the system which cause the system output to follow exactly the desired trajectory without any lag error. The identifier requires a certain amount of computational time to update the model weights. In general, higher processing speeds of the controller result in better following accuracy of the desired trajectory, and in less likelihood of undesirable instability which can be induced by the computational time delays inherent in the use of an identifier.

13.4 ARTIFICIAL INTELLIGENT BASED SYSTEMS

Volumes have been written to describe what is basic intelligence. According to the definition proposed by psychologists—"ability to understand symbolic concepts is fundamental to human intelligence". The symbols may include name of objects, or attributes, e.g., pen, room, dark, thin etc. Accordingly Artificial Intelligence (AI) has been defined as the branch of computer science which deals with the software and hardware techniques to solve symbolic problems as against 'number crunching' problems solved by EDP machines.

The difference between conventional EDP systems and A1 systems is shown in Fig. 13.8. Where the conventional EDP systems have ability to perform numeric calculations efficiently, AI systems handle symbols as concepts and ideas. As an example AI systems can understand screw driver and relate it to screw or conveyer belt. Similarly these systems can understand the meaning of 'scheduling' and relate to 'operation sequence'. These relations cannot be expressed in normal EDP systems. The AI programs contain information like data, collection of separate words or characters and knowledge, like interconnection of words, structure and relation map. The three outputs of AI systems are opinion, analysis and recommendations. The human decision-making process is shown in Fig. 13.9. The learned facts, experience and intuition are three basic inputs which enable decision-making. Collectively these are known as *heuristics.* In AI system the decisions are taken on the basis of heuristics.

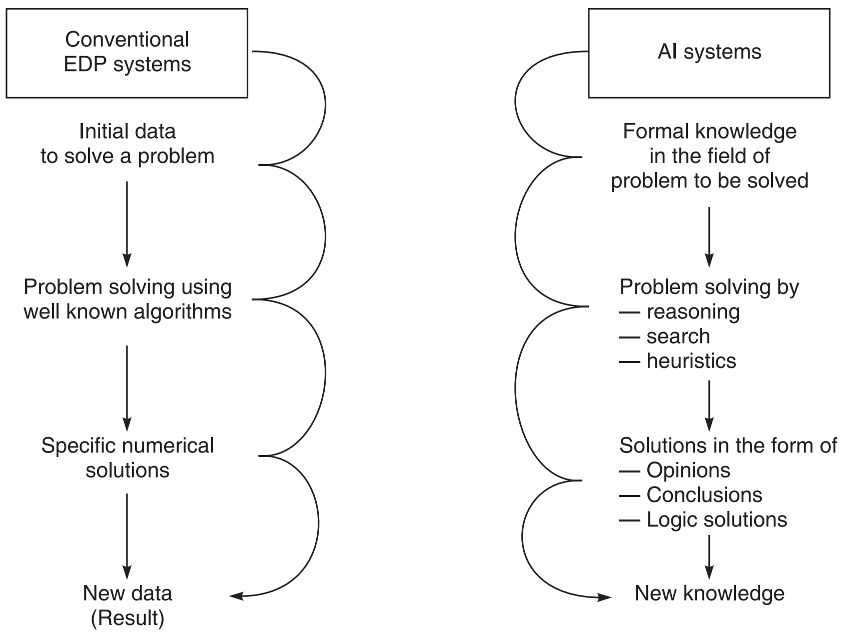


Figure 13.8 Difference in the working of conventional EDP system and AI systems.

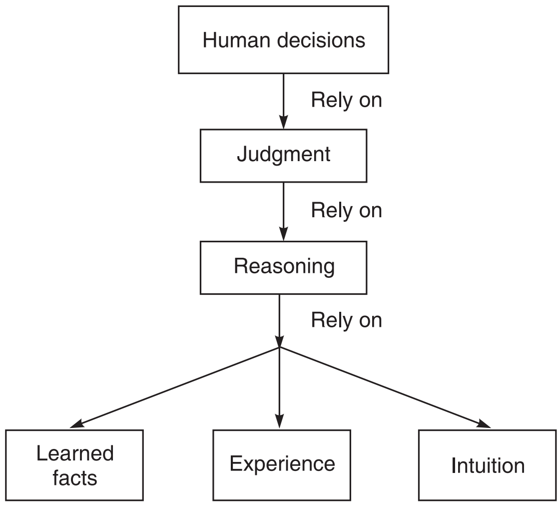


Figure 13.9 Human decision-making process.

AI systems can be divided into three major categories:

13.4.1 Natural Language Systems

AI systems that should be able to understand the natural language of the user like English, Hindi, French, German etc. as human beings can understand, fall in this category. Such systems will be easy means of communication with computer.

13.4.2 Perception System for Vision, Speech and Touch

Such AI systems can interpret visual scene or make inference about quality (crack, bend, broken) or physical orientation of the object. Vision system of robot falls in this category. The vision system of robot eliminates the need for reprogramming and readjustment of robot arm.

13.4.3 Expert or Knowledge Based Systems

These are the Artificial Intelligent systems (Fig. 13.10) that capture the expertise of human experts who are knowledgeable in a particular application domain. The information presented to Expert System may be

— Factual
— Experimental
— Incomplete
— Judgmental
— Speculative
— Uncertain
— Fuzzy
— Intuitive.

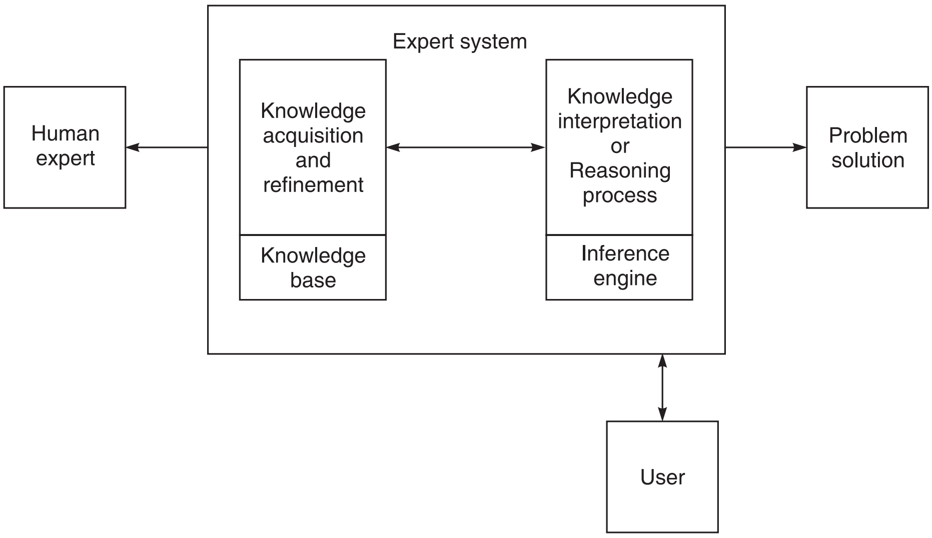


Figure 13.10 Expert system.

Expert System should derive conclusions based on such incomplete information, just like real-life human expert.

The general structure of expert systems is shown in Fig. 13.11. The domain knowledge is stored in the knowledge base of the system. The domain knowledge contains:

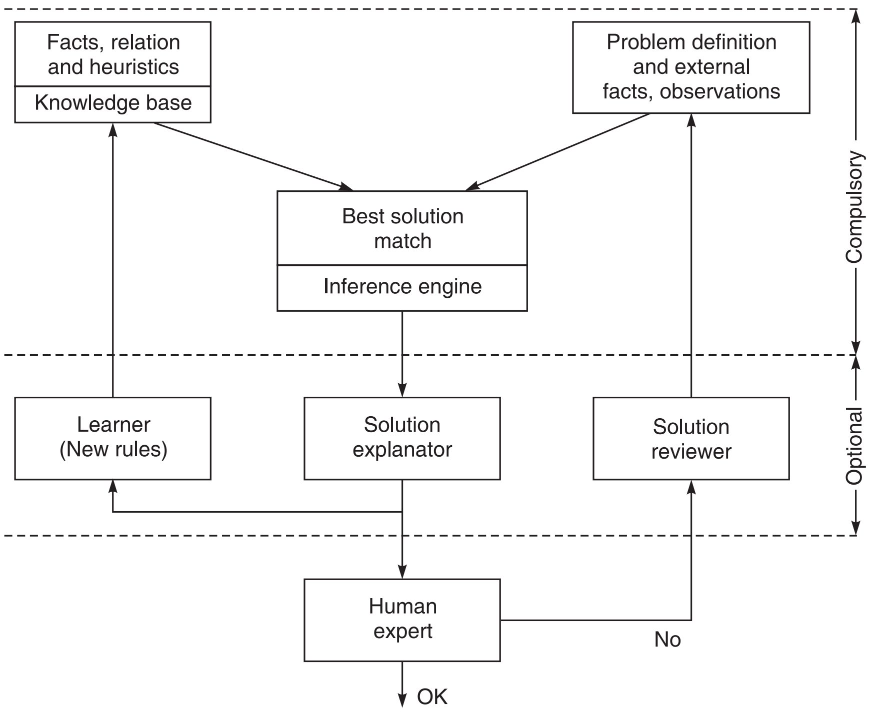


Figure 13.11 General structure of an expert system.

— Facts
— Relation between facts
— Heuristics.

The knowledge can be declarative and partially procedural. It is extracted from the domain expert by interviews conducted by Knowledge Engineer. The *inference engine* tries to determine the knowledge which solves the problem in hand. The *explanator* explains the reason for concluding the particular solution. The explanation given by explanator may be reviewed by human expert who may or may not accept the conclusion and its explanation. If human expert does not accept the conclusions, then he provides some more knowledge in terms of rules, facts or heuristics. The learner accepts the *new* knowledge and stores in knowledge base. The *solution reviewer* 'relooks' into the problem in hand and tries to find another solution. The process of reasoning, solution explanation, learning and solution review may be repeated till human expert accepts the solution determined by Inference Engine. The Learner, Solution Reviewer and Solution Explanator are important modules required for development and acceptance of expert system by human expert. These are therefore not compulsory modules after expert system is fully accepted. The minimum configuration of an expert system is shown in Fig. 13.12.

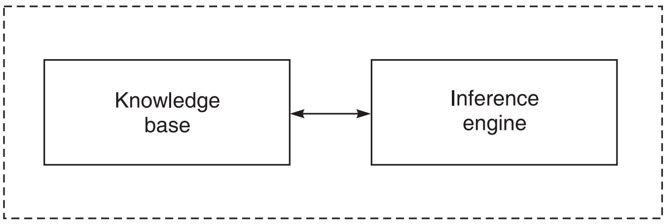


Figure 13.12 Minimum configuration of an expert system.

An expert system should have the following characteristics:

— Knowledge update capability
— Flexible problem solving strategy
— Reasoning and explanation capability.

Following steps are involved in building of an expert system:

— Knowledge acquisition
— Knowledge representation
— Inference strategy.

Knowledge acquisition

The role of knowledge engineer in building up expert system is very important. Knowledge engineer often faces what is widely known as 'knowledge bottleneck'. Following may be the reasons for this:

— Inexperienced people
— Scarce knowledge

— Poorly distributed knowledge
— Knowledge distributed among large number of people
— Knowledge not available with reliability.

It is the task of the knowledge engineer to overcome this bottleneck by rechecking on knowledge acquired by having discussions with different experts in the same domain.

Knowledge representation

The knowledge representation methodologies include.

— Semantic networks
— Frames
— Rules.

(i) *Semantic networks.* In semantic networks, nodes and arcs form graphical notation to represent objects (actions, events) and relationship among them. As an example, 'blast furnace produces' iron can be represented as shown in Fig. 13.13. Blast furnace, iron are the objects and produce is the relationship among them. Other relations may be 'is a', 'on', 'material', 'purpose', 'description' etc.

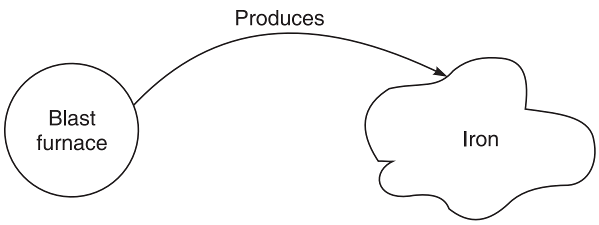


Figure 13.13 Simple relation in semantic network.

When a relation is true for a general class, then it should be true for all specific objects in that class. This is called inheritance relationship.

Figure 13.14 shows an example of semantic network. The 'purpose' and 'description' are inheritance relation in this example.

(ii) *Frames.* Knowledge representation by frames is similar to manner in which knowledge is stored in human memory. Frames are basically templates or moulds for holding clusters of related knowledge about a particular narrow subject. A frame consists of a number of slots. Each slot is a variable length memory area which holds different type of information associated with frame name. Figure 13.15 shows the structure of "common cold" frame.

Frames may be organised into an hierarchy in which the knowledge is presented framewise depending on the knowledge depth requirement. Figure 13.16 shows hierarchy of frames for a 'microprocessor based PID controller'. The topmost frame gives a general description of system. The subsequent layers progressively unravel more details about various sub-systems.

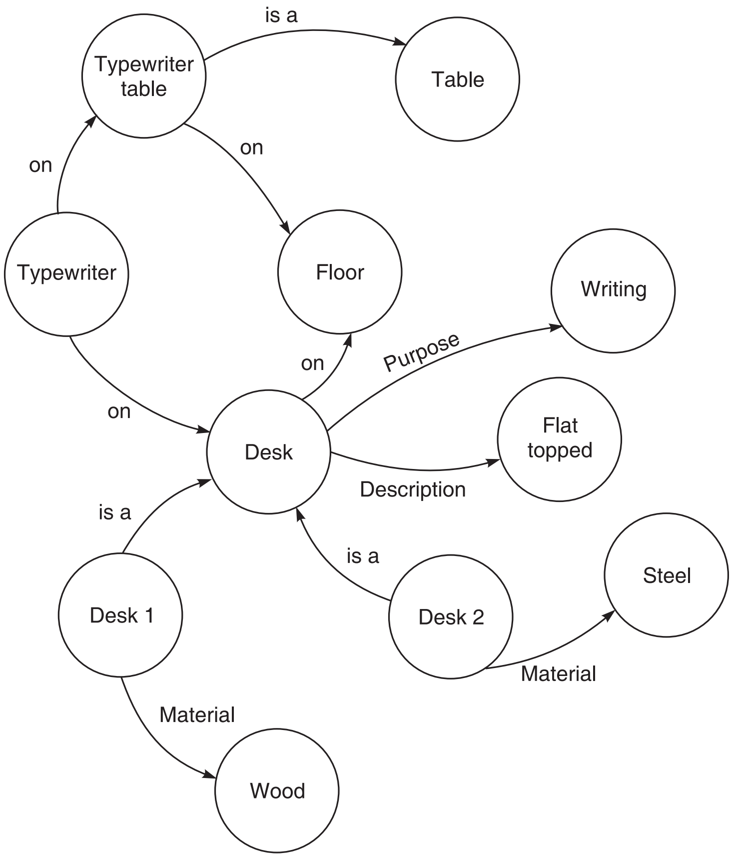


Figure 13.14 Semantic network—An example.

Slot 1:	Symptom value: (Stuffed nose) (Headache) (General malaise)
Slot 2:	Treatment value: (Take two aspirin) (Bed rest) (Call the doctor in the morning)

Figure 13.15 Common cold frame.

(iii) *Rules.* The method of knowledge representation by rules is easiest to understand and therefore most popular. Rules may be in one of the three forms:

(a) If {This happens} then do (That}
or

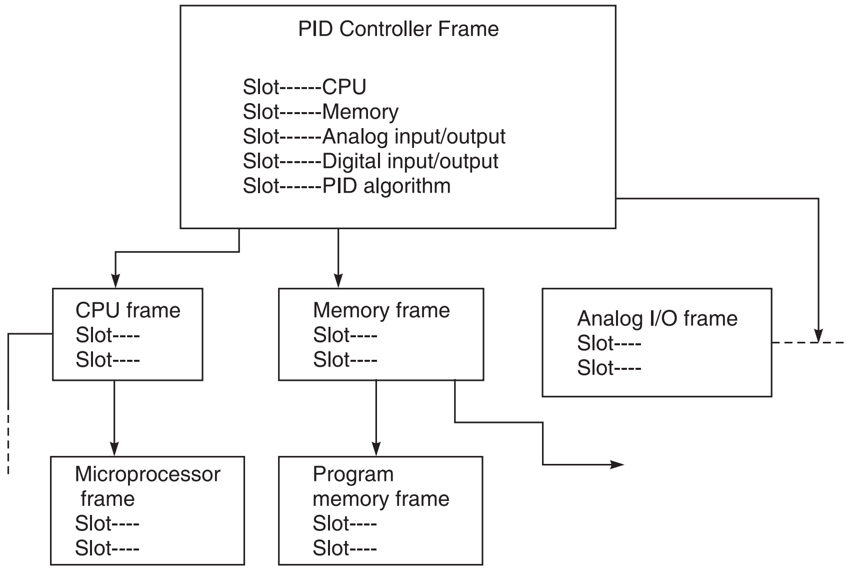


Figure 13.16 Hierarchies of frames for PID controller.

(b) If {This is the case) then {That is true}
or
(c) If {This is the case} then do {That)

These statements are opposed to conventional computer statement 'If (Cond) Then .....' in which condition is deterministic and number of paths in then—are few. However in rule based knowledge representation, the number of paths may be very large. This is a drawback of knowledge representation through rules.

Inference strategy

From its knowledge base, the A1 system should find out the right piece of knowledge which is applicable to the problem in hand. The reasoning mechanism is explained in Fig. 13.17.

Based on the observation, i.e. current situation for which the solution is to be found, the reasoning mechanism finds out the matching rule and there by conclusions are derived. There are two main reasoning mechanisms, viz. Backward Chaining and Forward Chaining.

Backward chaining starts with a goal, i.e. conclusion which is most probable and then tries to go backward and match the left hand side of rule, i.e. conditions. The conclusions may vary with different confidence level associated. It is also known as goal driven method. If a goal is not satisfied then another goal is pursued.

Forward chaining, on the other hand is top down approach. Some condition or data may cause a particular rule to be invoked. The system then proceeds in that direction until the goal, i.e. conclusion is reached or the conditions are not satisfied and it is not possible to proceed further. In latter situation, the reasoning system or inference mechanism goes back and starts

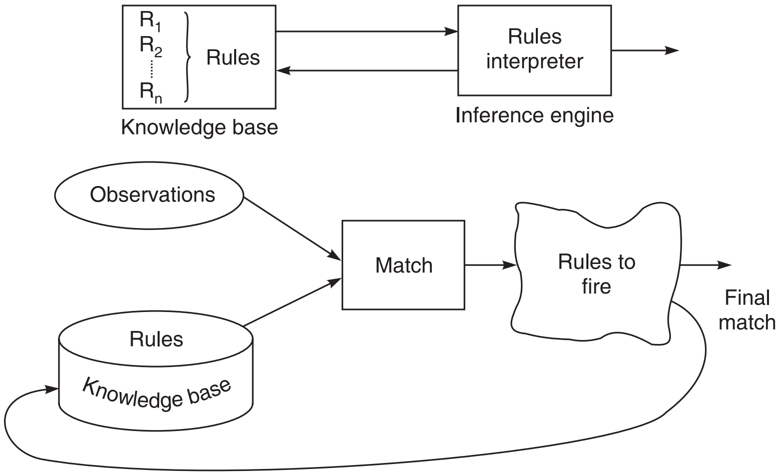


Figure 13.17 Inference strategy.

from the beginning. In forward chaining, there is however an obvious danger of entering a trap of long search.

Both the backward and forward chaining strategies may have further variation in Breadth First (BF) or Depth First (DF) approach, for searching. In breadth first approach, all premises of a rule are examined fully before attempting to gain further details. Thus BF places emphasis on getting overall picture, i.e. generalist approach. On the other hand, depth first approach digs deeper into details and follows a chain of rules. Breadth first approach is more successful when rule succeeds, but at times it may be laborious and annoying to the user. Depth first approach on the other hand directs the questions and pursues a theme like real experts.

13.5 EXPERT CONTROLLER

An ideal expert controller should be able to satisfy the following goals:

1. Ability to control a large class of processes which may be time varying, non-linear, with variety of disturbances.
2. Requirement of minimum prior knowledge about the process.
3. Ability to improve its performance with time, as it acquires more and more knowledge.
4. Ability to monitor the performance of the system and detect problems with sensors, actuators and other components.

Figure 13.18 shows the basic block diagram of an expert controller. In addition to main feedback loop, there is an outer loop involving parameter identification and expert system which performs the tasks of supervision and controller design. The controller may be simple PID controller or one of the advanced controllers described earlier while discussing adaptive or optimal control strategies.

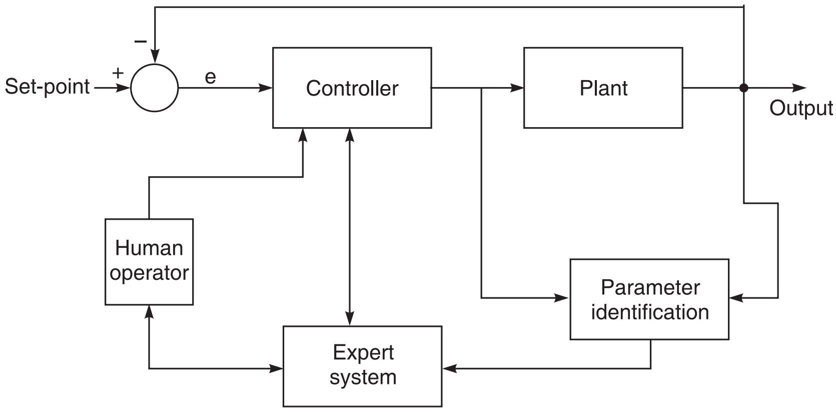


Figure 13.18 Expert controller—A block diagram.

13.6 FUZZY LOGIC SYSTEM

13.6.1 Introduction

Decision making in the real world is based on mostly vague and imprecise data. There is a famous saying that “Life is the art of drawing sufficient conclusions from insufficient data”. Although faster computers are now available for best results, the number crunching power alone does not suffice to achieve the effective process control, if the data itself is vague and imprecise. Some of the important questions that have haunted control engineers are:

1. Is it necessary to know a detailed mathematical model of the system in order to control it effectively?
2. Is it possible to make the controller learn as to how to control the plant effectively?
3. How does a human controller act in such circumstances where the data is vague and incomplete?

These considerations have led to great deal of investigation.

Fuzzy logic addresses these problems by presenting a meaning to these imprecise informations, so that these can be used effectively by computers. Fuzzy logic can be viewed as an extension of multi-valued logic system. In two-valued logic system, a proposition is either true or false. In multi-valued logic system a proposition may be true or false or have an intermediate truth value which may be an element of infinite truth valued set. In fuzzy logic, the truth values may range over the fuzzy subsets.

As an example in conventional computer systems, we express the temperature in degree centigrade, height in feet or centimetre and so on. However in real life we may say it is cold, not so cold or very cold or person may be tall, very tall etc. Fuzzy logic gives mathematical representation to this imprecise knowledge.

The linguistic variables provide link between natural language and representations which accommodate quantification. The linguistic variable has values consisting of words or sentences rather than numbers. As an example, height may assume the *values* of ‘very short’, ‘short’, 'medium', 'tall', 'very tall'. These values which are themselves linguistic variables may in turn be given meaning.

An important concept in fuzzy logic is *grade of membership*—A number which describes the extent to which an element is in a set. It also describes the truth value of a particular statement. The grade of membership of fuzzy set is defined by *membership function*. The membership function provides a direct linkage between fuzzy logic and fuzzy sets. It is defined by m_A (x). Where x is a member of A and A is some propositional or predicate class.

Figure 13.19 show the membership function for the set of young persons.

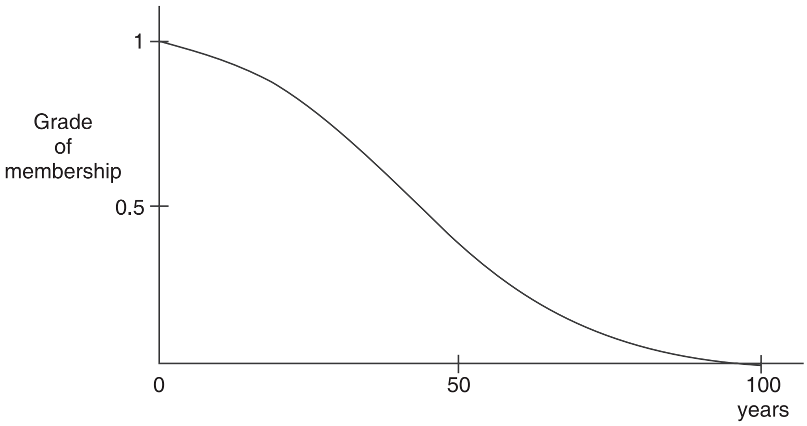


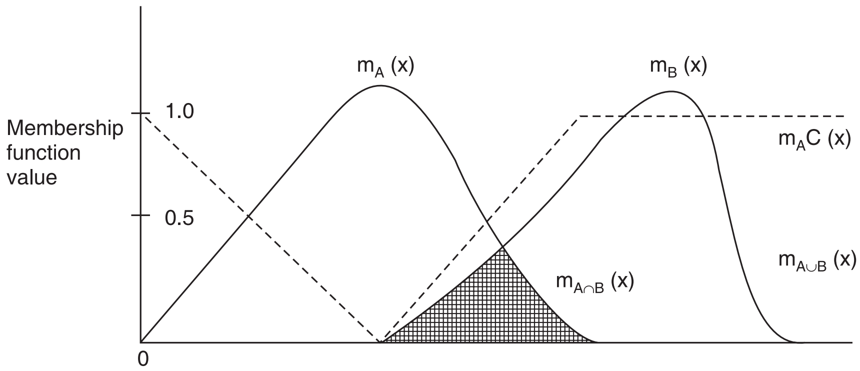
Figure 13.19 Membership function for set of young persons.

In fuzzy control applications, membership functions are constructed so that they determine the way observations of process variables are expressed in fuzzy sets. Appropriate membership functions are assigned to each range of process variable. This yields the fuzzy set values. As in crisp sets, the concepts of Complements, Intersection and Unions are defined for fuzzy sets and the membership functions of fuzzy sets satisfy these operations. Figure 13.20 shows these operations on membership functions of two sets.

We shall now discuss the concept of fuzzy controller design and its implementation. However the theory of fuzzy sets, fuzzy logic etc. has been purposely not included.

13.7 FUZZY CONTROLLER

A number of industrial processes such as blast furnace, cement kiln etc. are difficult to control due to their non linear behaviour and poor quality of variable measures. In some modem plants with process control computer, plant models have been used to calculate the required controller parameters thus automating the higher level control function. The plant models are approximation to real proces and may require large amounts of computer time. There are some successful implementations but difficulties have been experienced where processes operate over wide range of conditions. An alternative approach to the control of complex processes is to investigate the control strategies followed by a human operator which are based on intuition and experience. The fuzzy controllers are a step in this direction.



$m_A(x)$, $m_B(x)$ = membership function for two fuzzy sets A and B
$m_AC(x) = 1 - mA(x)$ = membership function of complement of fuzzy set A
$m_{A\cup B}(x) = m_A(x) \cup m_B(x)$ = membership function of union of two sets A and B
$m_{A\cap B}(x) = m_A(x) \cap m_B(x)$ = membership function of intersection of two sets A and B

Figure 13.20 Complement, intersection, and union operations on fuzzy sets.

The human operator can effectively control the process based on the control strategy even with imprecise data. As an example, typical rule for temperature control task executed by human operator would be: If temperature is **high** and **rising** then increase the cooling. The *high* and *rising* are imprecise magnitudes for temperature and its rate of change.

The fuzzy controller is designed to deal with such situations where available sources of information are inaccurate, subjectively interpreted or uncertain. The block diagram of fuzzy controller is shown in Fig. 13.21. The main constituents of fuzzy controller are:

— Fuzzyfier
— Knowledge base
— Decision strategy
— Defuzzyfier.

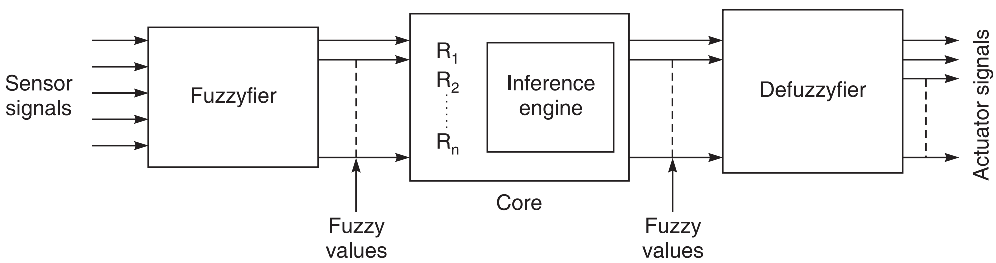


Figure 13.21 Fuzzy controller architecture.

13.7.1 Fuzzyfier

In the fuzzy controller, in order to include the control strategy of human operator the sensor readings which are non-fuzzy should be converted to fuzzy form by a process called fuzzification. In order to fuzzify the information, the knowledge from experts is obtained to classify the crisp information into different levels. For each measured variable, a matrix of values is prepared in advance. The values of variables are then classified into different levels from very high to very low. The membership function for each measured variable is also drawn. The sensor reading and resulting error values can be classified into 8 levels—"Positive Big, Positive Medium, Positive Small, Positive Nil, Negative Nil, Negative Big, Negative Medium and Negative Small". Figure 13.22 shows a look-up table relating fuzzy subsets to quantised error values for an application. These basic subsets may then be used with three basic operators of union, intersection and complement for further inference.

Quantised error	–6	–5	–4	–3	–2	–1	–0	+0	+1	+2	+3	+4	+5	+6
PB	0	0	0	0	0	0	0	0	0	0	0.1	0.4	0.8	1.0
PM	0	0	0	0	0	0	0	0	0	0.2	0.7	1.0	0.7	0.2
PS	0	0	0	0	0	0	0	0.3	0.8	1.0	0.5	0.1	0	0
P0	0	0	0	0	0	0	0	1.0	0.6	0.1	0	0	0	0
N0	0	0	0	0	0.1	0.6	1.0	0	0	0	0	0	0	0
NS	0	0	0.1	0.5	1.0	0.8	0.3	0	0	0	0	0	0	0
NM	0.2	0.7	1.0	0.7	0.2	0	0	0	0	0	0	0	0	0
NB	1.0	0.8	0.4	0.1	0	0	0	0	0	0	0	0	0	0

Figure 13.22 Look-up table to relate fuzzy subsets to quantised error values.

13.7.2 Knowledge Base

The knowledge base contains the set of decision rules through which fuzzy information obtained from fuzzyfier may be utilised to obtain decision. The decision rules are developed heuristically for a particular control task and are implemented as set of fuzzy conditional statements of the following form.

"If E is PB or PM and if CE is NS then CU is NM"

These expressions define fuzzy relation between a set of observed imprecise values of error (E) its rate of change (CE) and the appropriate change in control input (CU). The complete control strategy is defined in the above form of set of fuzzy rules. The above form is called 'conditional composition'. Other statements of the following form are also used.

Conjunctive Composition—A is Tall and B is Short

Disjunctive Composition—A is Tall or B is Short

Conditional and Conjunct—If A is Tall

Then Y is Short

Else Y might be Short.

13.7.3 Inference Strategy

Inference strategy involves executing the decision rules and then drawing conclusion. In real-life examples more than one rule may be applicable, resulting in more than one conclusion. The membership functions of antecedent (left hand portion) and conclusions (right hand portion) of rules are prepared in advance. The membership function of antecedent is evaluated with respect to membership function of observed value. This is further evaluated with respect to membership function of conclusions.

13.7.4 Defuzzifier

The defuzzification involves finding the exact value of conclusion (control signal) from its membership function. Following methods are usually used:

1. *Maximum height approach* in which the maximum value of resultant membership function is used.
2. *Mean of maximum approach* in which the arithmetical mean value of all the maximum available in the resultant membership function is used.
3. *Centre of gravity approach* which calculates the point of centre of gravity of resultant membership function curve. The centre of gravity of a membership function $m_A(x)$ is defined by:

$$\text{Centre of gravity} = \frac{\sum_{i=0}^{n} m_A(x_i) \cdot x_i}{\sum_{i=0}^{n} m_A(x_i)}$$

where n is the number of rules fired. The centre of gravity method is widely used and found very useful.

Figure 13.23 shows the fuzzy rule processing using membership value and input value. The fuzzy control loop is shown in Fig. 13.24.

13.8 FUZZY LOGIC TOOLS

Considerable advancement has taken place in development of fuzzy controllers. Several expert system shells based on fuzzy logic are commercially available. A fuzzy controller shell ERIC (Extended Rule Based System for Intelligent Control) for real time process control is quite powerful. Fuzzy logic chips were designed for using fuzzy based expert systems and more generally in rule based systems not requiring a high degree of precision. Fuzzy logic chips consist of four major components namely, rule set memory, an inference processor, a controller and I/O circuitry. The fuzzy logic chip designed by Watanabe and Togai in BEL Telephone Laboratories has execution speed of approximately 250,000 Fuzzy Logical Inference Per Second (FLIPS) at 16 MHz clock. A fuzzy inference accelerator which is a co-processor board has also been designed. The FP-3000 digital fuzzy processor designed in 1987 by M/s. Omron, Japan incorporates fuzzy inference and defuzzification functions on single chip. It enables high speed inference processing at the clock speed of 24 MHz. Three different groups of inference rules can be stored in memory.

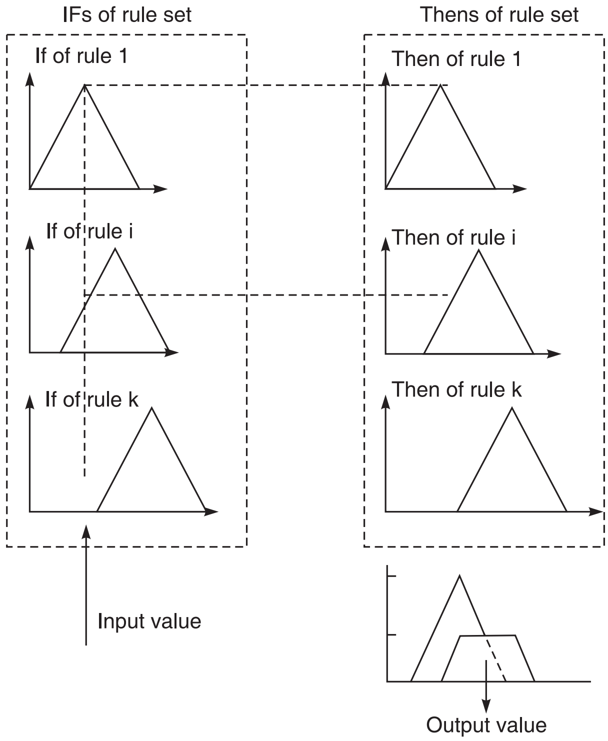


Figure 13.23 Fuzzy-rule processing.

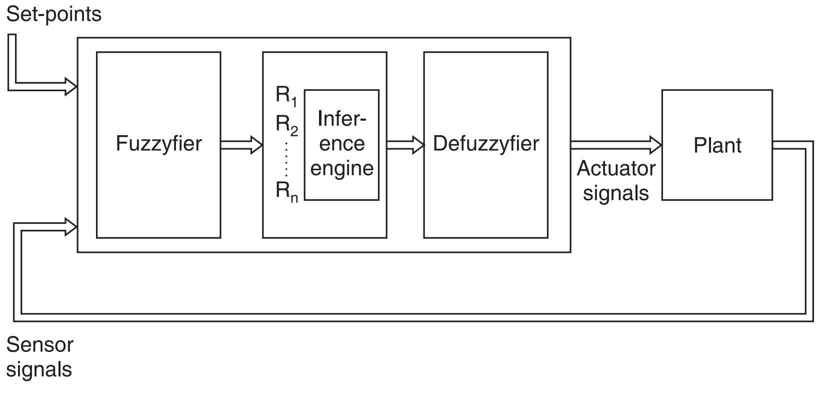


Figure 13.24 Plant control using fuzzy controller.

The general purpose fuzzy controller FRUITAX designed by Fuji Electric Co. has been successfully used in water treatment plant automation. Other fields where successful implementations have been reported include cement kiln control, steel mill and chemical plants.

Table 13.1 shows some of the fuzzy logic tools and products commercially available at present.

Table 13.1 Fuzzy-logic Tools and Products

Company	*Product*	*Description*
American Neuralogix Inc.	NLX 230 fuzzy micro-controller	Has 8 digital inputs, 8 digital outputs, 16 fuzzifiers; holds 64 rules. Evaluates 30 M rules/sec.
	ADS 230 fuzzy micro-controller development system	PC-compatible system uses NLX 230 with analog and digital I/O.
	NLX 110 fuzzy pattern correlator	Correlates eight 1-M bit patterns; expandable to 256 *n*-bit patterns.
	NLX 112 fuzzy data correlator	Performs pattern matching on serial data streams.
Aptronix Inc.	Fide (Fuzzy, Inference Development Environment)	Runs under MS-Windows on PCs. Supports development, fuzzy simulation, debug tracing, and 3-D display of control surfaces. Real-time code generation for microcontrollers. Software implementation of fuzzy logic in C.
Byte Craft Ltd.	Fuzzy-C	Preprocessor translates fuzzy source code into C source code.
Fuzzy Systems Engineering	Manifold editor	Runs under MS-Windows on PCs. Edits rules in a matrix display.
	Manifold graphics editor	Runs under MS-Windows on PCs. Colour graphics display of rules and fuzzy sets.
Hitachi America Ltd.	Microcontrollers	The company has performance benchmarks for its H8/300 and H8/500 microcontrollers in fuzzy-logic applications, performed by Togai Infralogic.
Hyperlogic Corp.	Cubicalc	Software for developing fuzzy-logic applications. Runs under MS-Windows with 286 or higher processor. Simulates fuzzy and non-fuzzy systems.
	Cubicalc-RTC	A superset of Cubicalc. Provides runtime compiler support and libraries for linking. Compatible with Microsoft C and Borland C.
	Cubicalc runtime source code	Generates C source code for use in compiling to a specific processor.
	Cubicard	Includes Cubicalc-RTC and PC-based hardware for analog and digital I/O.
Inform Software Cop.	Fuzzytech Explorer Edition	Introductory fuzzy-logic system. Software runs under MS-Windows. Accepts two inputs, one output, and five fuzzy membership sets per variable and 125 rules.
	Fuzzytech MCS-96 Edition	Full fuzzy development system for MCS-96 micro-controllers. Generates optimised assembly code.
	Fuzzytech Online Edition	Debug and modify facility for fuzzy-logic systems while they are running. Generates C source code.

(contd.)

Table 13.1 Fuzzy-logic Tools and Products *(contd.)*

Company	*Product*	*Description*
Integrated Systems Inc.	RT/Fuzzy Module	Simulation and code generation of fuzzy logic for real-time systems.
The Metus Systems Group	Metus	PC based fuzzy-logic development and simulation system. Provides high-level modeling and low-level development for embedded applications.
Modico Inc.	Fuzzle 1.8	PC-based fuzzy-logic shell. Generates source code for C and Fortran.
Motorola Inc.	Fuzzy-logic kernel for microcontrollers	Fuzzy processing kernels for 68HC05 and 68HC11 microcontrollers. Includes fuzzy knowledge-base generator to create code for kernel.
	Fuzzy-logic educational kit	Interactive training tool provides good introduction for understanding and using fuzzy logic. Runs under MS-Windows.
Togai Infralogic Inc.	TIL Shell + fuzzy C development system	Complete fuzzy development system generates C code and includes debug, fuzzy-simulation, and graphical-analysis tools.
	Microcontroller evaluation packages	Fuzzy development systems for Hitachi H8/300, H8/500, and HMCS400; intel 8051; and Mitsubishi 37450.
	FC 110	Digital fuzzy-logic processor (IC).
	FC 110 development system	Hardware and software development system for FC 110 Versions support PC bus, Sbus, and VME bus.

13.9 ARTIFICIAL NEURAL NETWORKS

13.9.1 Introduction

For long, the human brain and its working have been subject of research. In fact to quote from the Gray's anatomy—"The human nervous system is the subject which is most complex, widely investigated and yet poorly understood till today". Computer scientists have been trying to understand the brain in order to understand how such an enormous data in the form of numbers, symbols, pictures, events etc. can be stored in such a limited space in the human brain. Also, the capability of human brain to perform complex calculations by training themselves through trial, retrial, learning from experience and modifying the strategy based on the experience is still a subject of continuous research for computer scientists. For control engineers, the ability of human brain to adapt itself to a changing environment and also controlling the process even when the data is not fully available or it is available in very fuzzy form, is very important for the design of adaptive controllers which can behave like human controllers. At present, Artificial Neural Networks (ANNs) are being used in several applications related to process control as well as computer hardware/software development.

The basic unit of the brain is a *neuron.* A neuron can be imagined as a tiny processing element. Currently, it is believed that a normal human brain has approximately 10^{11} neurons. Figure 13.25 shows a biological neuron. It consists of the main body called *soma* where decisions are made, the *axons* which carry the output signal from the soma to other neurons, the *Dendrites* which bring input signals to the soma and the *synapses* which couples axons from other neurons to the dendrites of this neuron. The synapses are further classified as pre-synaptic terminals, which transmit the signals to other neurons and post-synaptic terminals which receive the signal from other neurons.

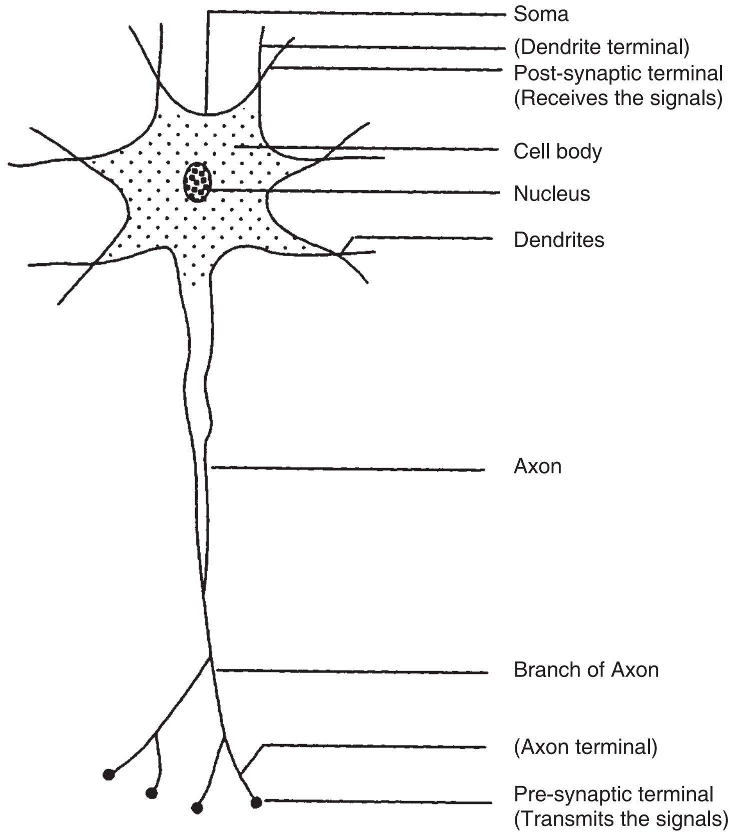


Figure 13.25 A biological neuron.

The computer scientists found a similarity between the above architecture and the Von Neumann's architecture proposed in 1940s for computers. A mathematical model of biological neuron is shown in Fig. 13.26. The synapses act on the signals $x_1, x_2, \ldots, x_n$ from different axons by synaptic weight parameter w. The resulting signals arrive at soma through dendrites. These signals are processed according to a non-linear *function of all the inputs*. The processed signal output is then carried by axons to other neurons. This model is a simplified version of biological neuron, in order to explain the concept.

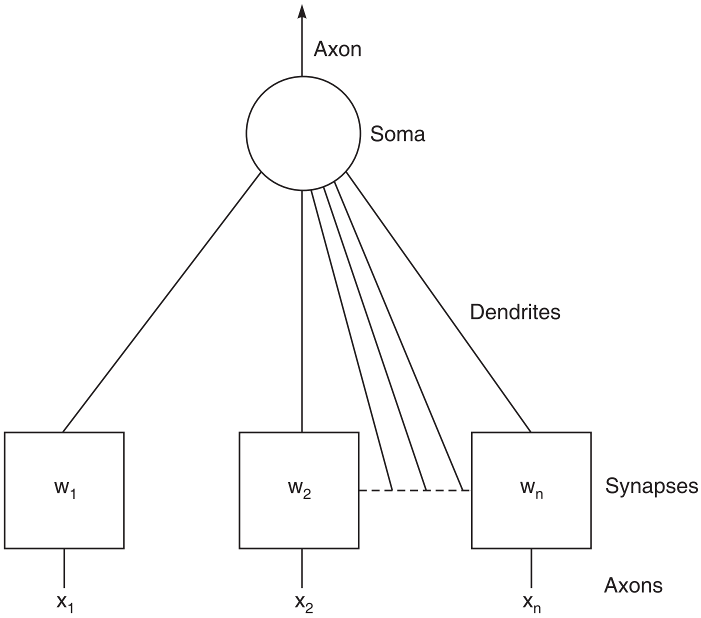


Figure 13.26 Mathematical model of a biological neuron.

McCulloch and Pits proposed a model of neuron (Fig. 13.27) in 1943. It consisted of a summing element and a threshold function. The neuron fires when the weighted sum of inputs $\Sigma w_i x_i$ reaches or exceeds the threshold u_i Though simple, the McCulloch and Pits neuron is

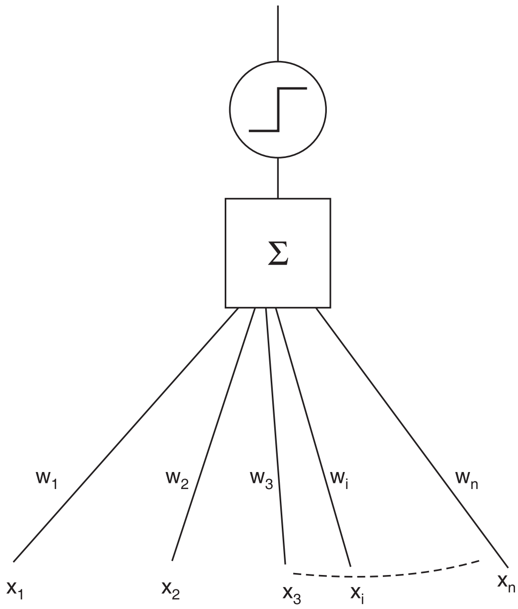


Figure 13.27 Schematic diagram of McCulloch and Pits neuron.

computationally a powerful device, as was demonstrated. It could perform AND, OR and NOT operations depending on the suitably chosen weights. However, the real neurons are far more powerful. The limitations of this simple neuron led to further developments.

A typical biological neuron is about 5 to 6 times slower than silicon logic gate. The silicon logic gate operates in nanosecond range whereas the biological neuron operates in the millisecond range. However, the brain is 'highly parallel' and contains 10^{11}neurons in total. The brain requires 10^{-16} Joules per operation per second energy as compared to 10^{-6} Joules per operation per second energy required by the fastest computer of today. In addition to this, the biological neurons have self-organising capabilities to perform complex operations like pattern recognition, much faster than the fastest digital computer available today.

The Artificial Neural Network is an attempt to mimic the action of the brain by using a similar structure. It consists of artificial neurons constituting a class of adaptive machines that perform computations through a process of learning.

13.9.2 Artificial Neural Network (ANN)—The Classification

The ANNs can be categorised on the basis of topology or learning methods.

Classification based on topology

(i) *Feedforward ANN.* The feedforward ANN are simple to understand. The neural structures in the brain are largely feedforward layered networks. Rosenblatt in 1957, gave the concept of *Perceptron* as layered feedforward network. The network consists of a set of input terminals which feed the input patterns to the rest of network. There may be one or more intermediate layers, followed by an output layer. The output of a layer is not fed to the same layer or preceding layers. Thus signals flow in forward direction only.

(ii) *Recurrent network.* Networks that are not strictly feedforward but include direct or indirect loops of connections are called Recurrent Networks. These include Hopfield Network, Stochastic Network, Boltzman machine etc.

Classification based on learning method

There are two categories of ANNs based on learning method.

(i) *Supervised learning.* In supervised learning (Fig. 13.28) the learning is done on the basis of direct comparison of output of network with known correct answer. The network receives the feedback about the errors and weights in different layers and adjusts to make the error zero. It is also known as learning through a teacher or *reinforced learning.*

The supervised learning networks are:

— Perceptron
— Adaline
— Hopfield network.

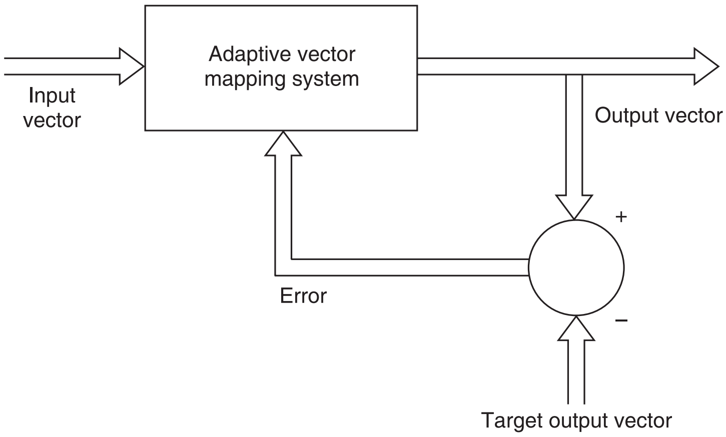


Figure 13.28 Supervised learning.

(ii) *Unsupervised learning.* In unsupervised learning, there is no teacher. Thus the network does not have any feedback of right or wrong answer. The network itself discovers the interesting categories or features in the input data. In many situations, the learning goal is not known in terms of correct answers. The only available information is in the correlation of input data or signals. The unsupervised networks are expected to recognise the input patterns, classify these on the basis of correlations and produce output signals corresponding to input categories. Unsupervised learning networks are:

— ART (Adaptive Resonance Theory)
— SOF (Self Organising Feature) maps or TFM (Topological Feature Maps) developed by Kohonen
— BAM (Bidirectional Associative Memory).

We shall describe here feedforward ANN with supervised learning methods, and their application in industrial control.

13.9.3 Learning Rules

The learning of ANN is by weight adjustment in different units at various layers so as to achieve the desired output. Following learning rules have been developed for ANNs:

— Simple Hebbian
— Delta Rule
— Generalised Delta Rule (Error Back Propagation)
— Competitive Grossberg (Winner take all)
— Kohonen
— Boltzmann (Simulated Annealing).

The generalised delta rule (also called Error Back Propagation) which has been widely used for both feedforward and recurrent ANN will be described in detail.

13.9.4 Perceptron

Rosenblantt in 1957 introduced the concept of Perceptron as a computing neuron. The model of perceptron is shown in Fig. 13.29. Perceptron as an artificial neuron has inputs x_1, x_2, …, x_n; a summing element; a non-linear element; connection weighing element, w_1, w_2, …, w_n that is adjustable connection weights and output y. The inputs x_1, x_2, …, x_n and the connection weights w_1, w_2, …w_n can be positive or negative. The artificial neuron shown here simulates a threshold function, which implies that the neuron will fire, if summation of x_i. w_i is positive and neuron will be inhibited if summation of x_i. w_i is negative.

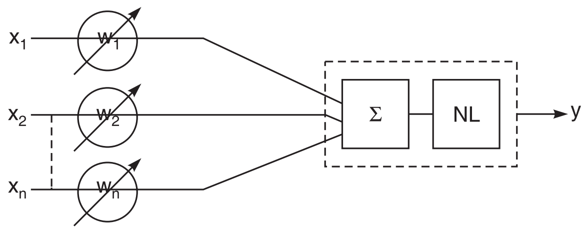


Figure 13.29 Perceptron model.

The non-linear function is basically uniform step function as shown in Fig. 13.30. The other non-linear functions which has been used is sigmoidal, or hyperbolic function. β is known as steepness parameter.

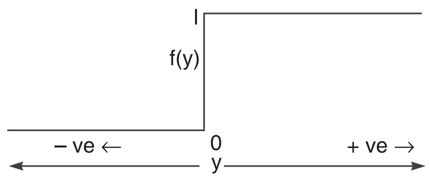


Figure 13.30 Unit step function.

Figure 13.31 shows the sigmoid function for different values of β. The hyperbolic function for $\beta = 1$ is shown in Fig. 13.32. Since these functions should saturate at both ends range, (0, 1) for sigmoid and (± 1) for hyperbolic functions is used. The steepness parameter is often set to 1 or 0.5.

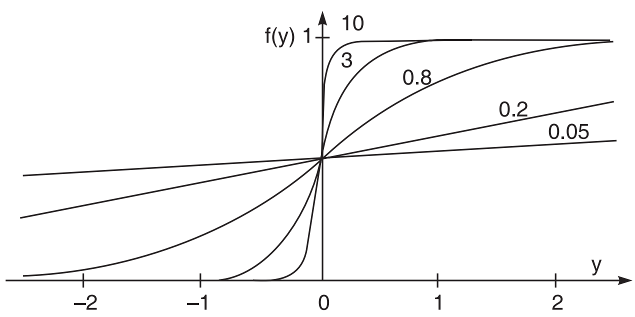


Figure 13.31 Sigmoid function for different values of β (Steepness parameter).

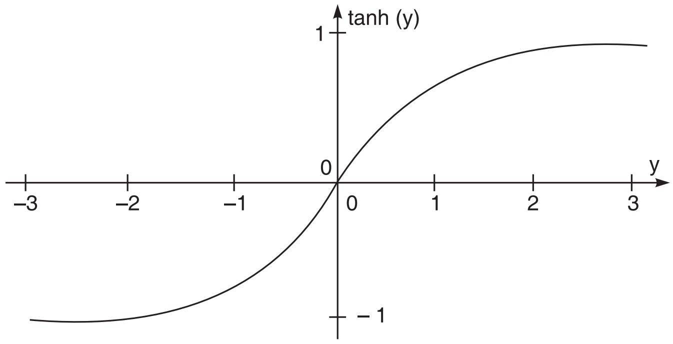


Figure 13.32 Hyperbolic function.

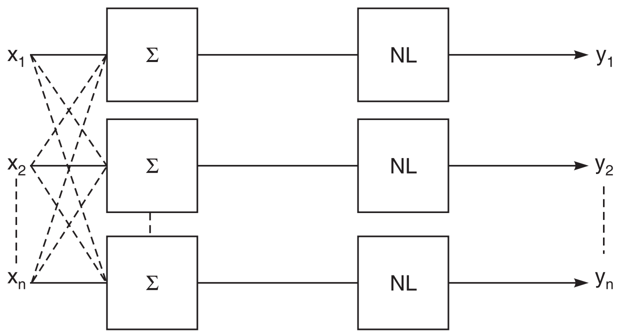


Figure 13.33 Multiinput multioutput perceptron (MIMOP) model.

13.9.5 Multi Input/Multi Output Perceptron (MIMOP)

The model of multi input/multi output perceptron, i.e., MIMOP is shown in Fig. 13.33. MIMOP can classify and recognise patterns by learning and association. Thus, it has two phases of operation. In phase I, the neural networks with MIMOPs are trained to recognise the patterns and in subsequent phase they are used for recognition. The MIMOP has inputs x_1–x_n and outputs y_1–y_n and $n \times m$ matrix of connection weights so that

$$y_j = f\left(\sum w_{ij} \cdot x_i + x_0\right)$$

x_0 is a bias signal. The learning of perceptron is equivalent to weight adjustment law. The gradient method is used for parameter adjustment, i.e. weight adjustment.

13.9.6 Multilayer ANN

So far we have dealt with the structure of one-layer Artificial Neural Network. Figure 13.34 shows simplified working of human photoperceptron for recognition of a pattern. It is clear that a human Photoperceptron is a multilayer neural network The analogy of human photoperceptron structure

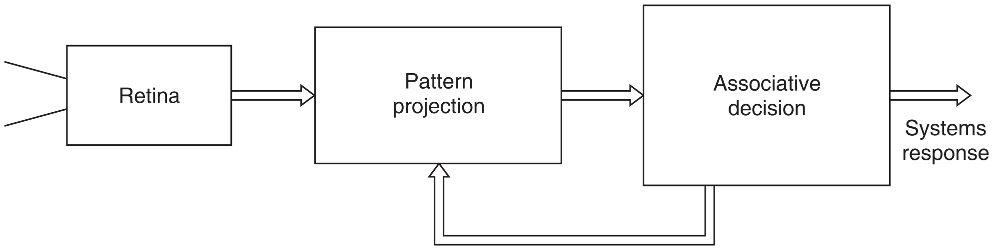


Figure 13.34 Pattern recognition by human brain.

to the Artificial Neural Network is shown in Fig. 13.35. It shows three layers, viz. input layer which performs feature detection, the output layer which performs pattern classification and a hidden layer which performs association of various signals. The law of learning is through weight adjustment. A number of methods like Hebbian law, Delta rule (Widrow-Hoff Method) etc. have been proposed and are being used.

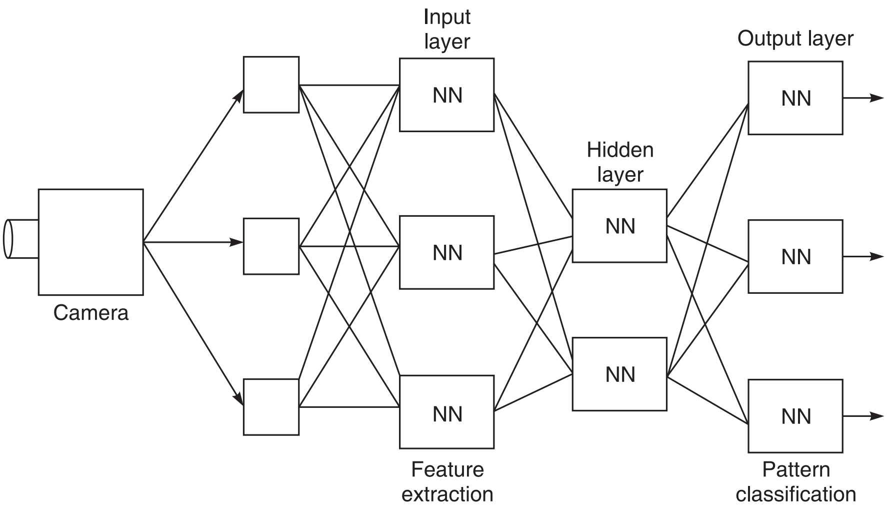


Figure 13.35 Human photo perceptron model.

13.9.7 Error Backpropagation Learning Algorithm

This technique was developed by Rumelhart, Hinton, and Williams in 1986. The backpropagation is important for multilayer networks with one or more hidden layers. The network is composed (Fig. 13.36) of three types of units, namely, input units, hidden units carrying internal representations and output units.

The operation of network is done in two phases. During forward phase the input is represented and propagated towards the output. During backward phase the errors are formed at the output and propagated towards input.

A sample vector O_j is generated at the input. Output vector O_k is compared with target output vector T_k provided by the teacher. The delta error signal $d_k = (t_k - o_k) \cdot f'(\text{net } k)$ i.e.

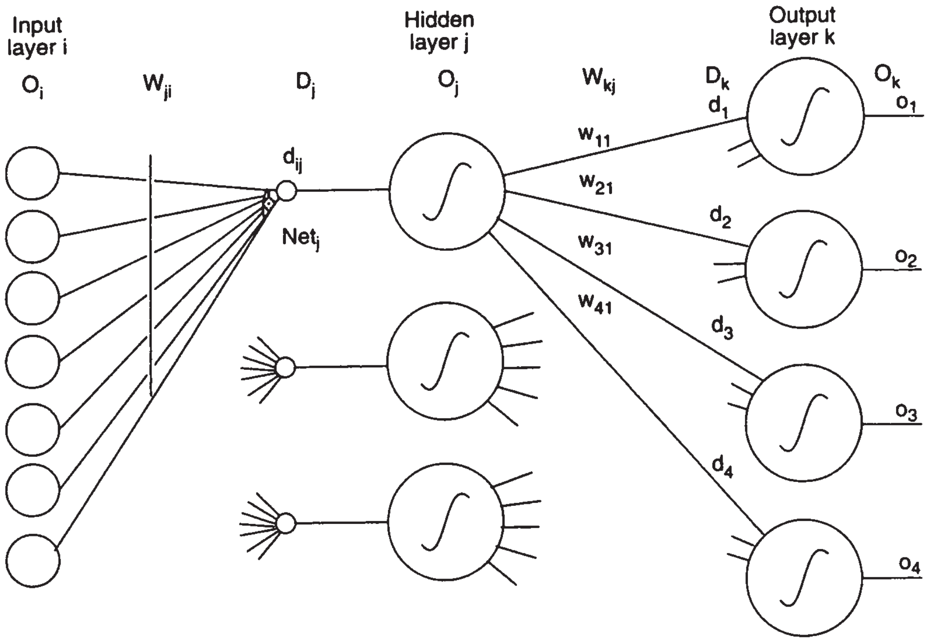


Figure 13.36 Backpropagation algorithm.

difference between output vector and target vector, obtained through sigmoid activation function (Fig. 13.37) for layer k is used as correction signal value. The weights w_{kj} at layer k are changed (Fig. 13.38) to make this correction. The units at preceding layers are taken next for weight correction and this is repeated till input units are reached.

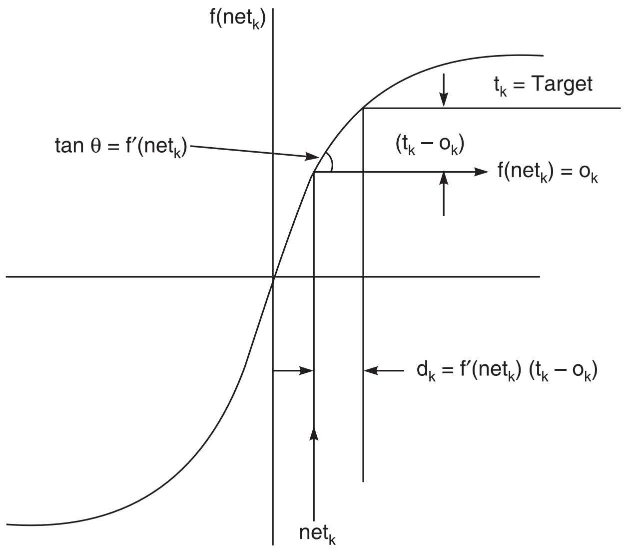


Figure 13.37 Correction signal value for layer *k*.

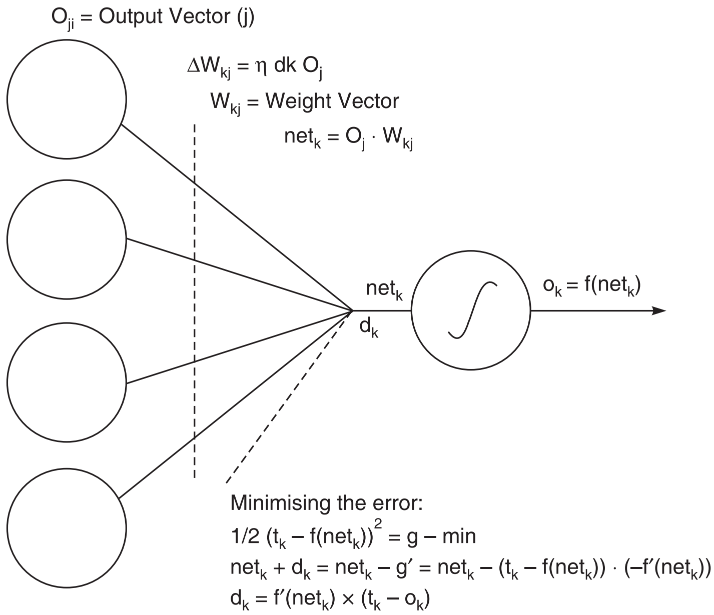


Figure 13.38 Weight correction in backpropagation algorithm.

Although ANNs are inspired by research in neurobiology, there is no evidence that backpropagation exists in actual neural mechanism and that synapses can be used in the reverse directions. Therefore, the backpropagation should be considered only as algorithm for self organisation of artificial learning systems.

13.10 NEURAL CONTROLLERS

Block diagram of an adaptive control system using neural network is shown in Fig. 13.39 wherein model reference adaptive control scheme is utilised. The neural network learns to control the process so that its output matches that of the reference model in some pre-defined

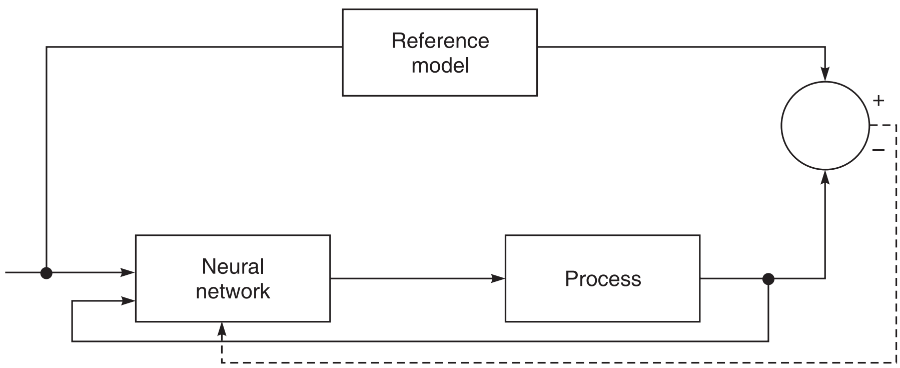


Figure 13.39 Adaptive control using neural network.

optimum manner. This may be done either in the form of finding the best possible state feedback for the system or by simply synthesising the control input. In fact, neural network can learn to control a non-linear system almost as easily as it can learn to control a linear system.

Another control strategy using neural networks is their use in identifying the system model. This enables the self-tuning strategy to be adopted using neural networks. The neural network identifies the parameters of a discrete time linear model of a given order by determining the weights that match the output of the model with that of the system for a given weight. This model can then be utilised for designing the control model.

Another possible approach may be inverse dynamic approach where the neural network can be trained to generate the input so that the given system has the desired output. The neural network used for identification can keep track of changes in parameters of the plant and this information can be used to update the neural network controller that generates the input to the plant (Fig. 13.40).

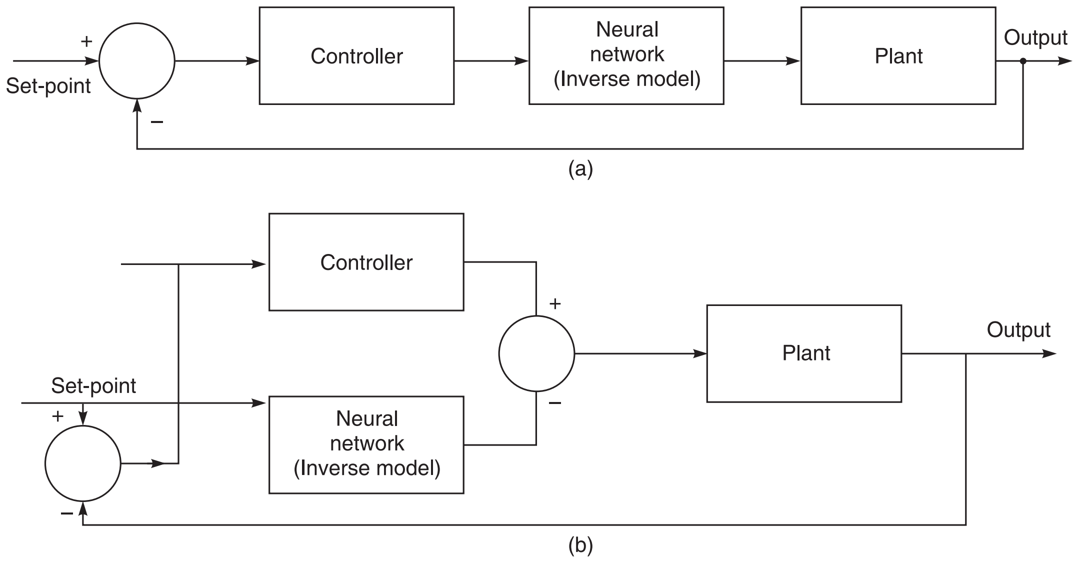


Figure 13.40 Inverse model based control: (a) Serial inverse model control, and (b) Parallel inverse model control.

13.11 VLSI IMPLEMENTATION OF NEURAL NETWORKS

The extremely parallel nature of neural network algorithms have been implemented in hardware using VLSI technology. Several approaches have been proposed and implemented for neural network hardware implementation.

Analog techniques

Although analog circuits suffer from lack of precision, analog implementation of artificial neural networks has been popular. The sum of products computation is well suited for analog paradigm. A synaptic weight is represented by conductance. Thus an input voltage applied across a register yields a current equal to voltage × conductance. The output computation may be easily implemented using trans-conductance amplifier. The input/output relation of this device is described by

$$I = \tanh(a\upsilon)$$

where υ is the voltage difference between the inputs υ_1 and υ_2 and a is a constant. The weight storage may be implemented as the voltage difference between two floating weights.

Digital techniques

The digital VLSI technology offers the advantages of higher precision, ease of weight storage and cost performance in the form of programmability over analog VLSI technology. The general purpose neurocomputer chip, known as CNAPS has 64 processors, each with 4 KB of local memory and with clock speed of 24 MHz. The attractive feature of the chip is that it is capable of implementing most current neural network algorithms with on-chip learning much faster than on conventional digital techniques.

Hybrid systems

Hybrid systems exploit the merits of both analog (compactness, potential speed and absence of quantisation effect) and digital (robustness, ease of weight storage and programmability) methods. Pulse mode signalling technique has been used in hybrid systems.

Special purpose VLSI chips

A number of neural VLSI chips are currently available. A configurable chip called ANNA (Analog Neural Network Arithmetic) Logic Unit is implemented using 0.9 pm CMOS technology. It has 4096 physical synapses. The resolution of synaptic weights is 6 bits and that of states (input/output of the neurons) is 3 bits. The chip uses analog computation internally but all input/output is digital. Other special purpose VLSI chips are Silicon Retina and Silicon Cochlea which use analog models based on neurobiology.

13.12 NEURO-FUZZY CONTROL SYSTEM

Recent research and design efforts are geared towards designing control systems with the combination of fuzzy and neuro control techniques. There are many similarities between neural and fuzzy controllers. Both the techniques can handle extreme non-linear behaviour of the system and allow interpolative reasoning. The main idea in integrating the neuro and fuzzy control techniques is to use the strength of each, resulting in Neuro-Fuzzy Control Systems.

The fuzzy knowledge representation can help the neural network with explanation facility. The neural network on the other hand can continue to learn and facilitate fuzzy controller in improving its performance. The integration of neuro and fuzzy controller will thus enable the automatic design and fine-tuning of membership function used in fuzzy control through learning of neural networks. Figure 13.41 shows the architecture of neuro-fuzzy control system proposed in 1992. The architecture has main elements, namely, Action-state Evaluation Network (AEN) and Action Selection Network (ASN). Both these elements are modelled by multi-layer neural networks. The Action Selection Network includes a fuzzy controller and acts as main controller. The AEN evaluates the performance of system and learns by updating the prediction of failure. It then advises the ASN on fine-tuning the fuzzy controller. Both AEN and ASN neural networks employ reinforcement learning with hidden layer, using modified error backpropagation scheme.

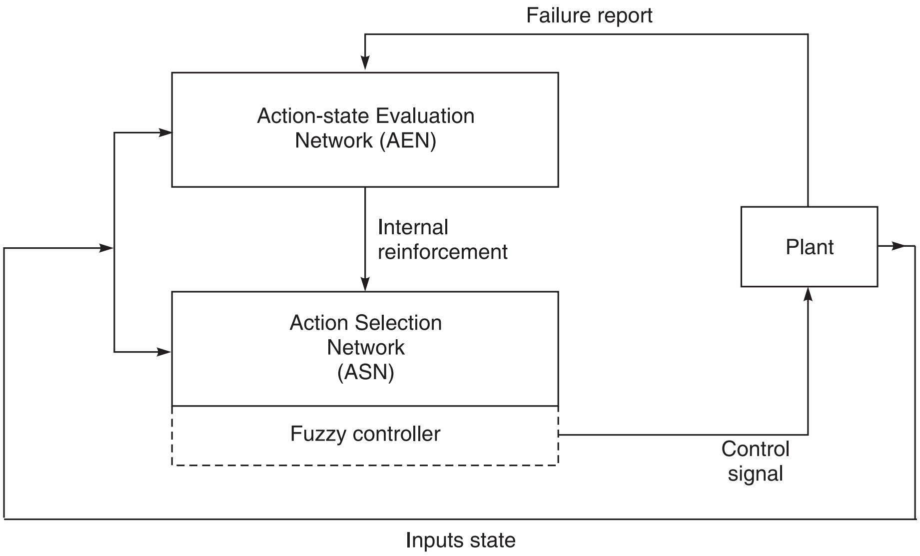


Figure 13.41 Architecture of neuro-fuzzy control system.

The combination of neuro controller and fuzzy controller has been successfully used to control the flatness of Cold Rolling Process in Sendzimir Rolling Mill, Japan.

13.13 CONCLUSIONS

In the process industries, the biggest challenge facing process engineers will be the reduction of various costs while maintaining product quality. Advanced process control is the most effective technology available to realise this objective, especially in established plants.

All information is of value, and no information should be discarded just because it does not conform to a particular model-building procedure. Thus, new modelling methods are also required. These should provide a framework where a priori knowledge of the process could be combined with the various existing modelling techniques, leading to so-called *grey-box models*. The resulting model should also be amenable for utilisation by the different modern controller designs, thus rendering controller synthesis independent of model types.

With regard to the primary modules making up an advanced control project, neural networks, nonlinear systems theory, robust control and knowledge-based systems are areas which appear to have captured the attention of both researchers and practitioners in the field of control engineering. This trend will continue well into the next decade. Areas that will receive particular attention are techniques that translate raw data into useful information; improved measurement methods including inferential estimation; multivariable non-linear predictive control; and formal techniques for analysing the integrity of neural network based methodologies.

Worldwide research efforts are continuing in the neural and neuro-fuzzy control systems. It is expected that this field will 'mature' in next ten years as we shall be having many more success stories in terms of real-life examples of actual implementation of neuro-fuzzy controllers.

There is worldwide awareness and research efforts in the area of neurobiology. The complete sequence of DNA has been decoded as a result of these efforts. These techniques find applications in intelligent control systems. Genetic algorithms have already been developed for electric power systems, and are one of the strongest search candidate for future intelligent systems. A simple genetic algorithm consists of three basic operations, viz. Reproduction/ Selection, Cross-over and Mutation. With the parallel computers becoming more and more popular and economic, genetic algorithms are going to become ideal choice for intelligent optimal control. In future, genetic algorithms would be used more and more in combination with fuzzy and neural systems.

SUGGESTED READING

Astrom, K.J., Anton, J.J. and Arzen, K.E., Expert control, *Automatica,* **22,** No. 3, 1986.

Berenji, H.R. and Khedkar, P., Learning and tuning fuzzy controllers through reinforcement, *IEEE Transactions on Neural Networks,* **3,** No. 5, 1992.

Bochen, K.A. and Andreou, A.G., A contrast sensitive silicon retina with reciprocal synapses, *Advances in Neural Information Processing Systems,* Elsevier, North Holland, 1992.

Brunko Soucek, *Neural and Concurrent Real Time Systems: The sixth generation,* John Wiley & Sons, New York, 1989.

Considine, Douglas M., *Process Instrumentation and Control Handbook,* McGraw-Hill, New York, 1974.

Goldberg, D.E., *Genetic Algorithms in Search, Optimization and Machine Learning,* Addison-Wesley, New York, 1989.

Gupta, M.M. and Nikiforuk, P.N., *Approximate Reasoning in Expert System,* Elsevier, North Holland, 1985.

Gupta, M.M. and Rao, D.H., *Neuro Control Systems: Theory and applications,* IEEE Press, USA, 1994.

Hattori, Satoshi, Nakajima, Masaaki, and Katayama, Yasunori, Fuzzy control algorithm and neural networks for flatness control of a cold rolling process, *Hitachi Review*, **42,** No. 1, 1992.

Hertz, John, Krogh, Anders, and Palmer, Richard, G., *Introduction to the Theory of Neural Computation,* Addison-Wesley, New York, 1991.

Hirota, K., *Industrial Applications of Fuzzy Technology,* Springer-Verlag, Berlin, 1993.

King, P.J. and Momdoni, E.H., The applications of fuzzy control systems to industrial processes, *6th IFAC Congress on Control Technology in the Service of Man,* 1975.

Kraus, T.W. and Myron, T.J., Self-tuning PID controller uses pattern recognition approach, *Control Engineering,* June 1984.

Negoita, C.V., *Expert Systems and Fuzzy Systems,* Benjamin/Cummings, California, 1984.

Nelson, Morgan, *Artifcia1 Neural Networks: Electronic implementation,* IEEE Computer Society Press, USA, 1990.

Nguyen, Derrick, H. and Widrow, Bernard, Neural networks for self learning control systems, *IEEE Control System Magazine,* April 1990.

Togai, M. and Watanbe, H.A., Fuzzy Inference Engine on a VLSI Chip—Design and implementation, *2nd Fuzzy System Symposium,* 1986.

Index