NETWORKING OF PLC'S

A PROJECT REPORT

Submitted by

ARVIND.R 71203106009
KHOUSHIK.S 71203106023
VISHNU VENUNATH 71203106060
DEEPAK DAS.K 71203106300

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY: CHENNAI – 600 025

APRIL 2007

ANNA UNIVERSITY: CHENNAI – 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"NETWORKING OF PLC'S"** the bonafide work of **"ARVIND.R, KHOUSHIK.S, VISHNU VENUNATH and DEEPAK DAS.K"** who carried out the project work under my supervision.

SIGNATURE

Dr.Rajeshwari Mariappan

HEAD OF THE DEPARTMENT

Department of ECE,
Kumaraguru college of
Technology
Coimbatore 641006

SIGNATURE

Mrs. A.Amsaveni. M.E.,

SENIOR LECTURER

Department of ECE,
Kumaraguru college of
Technology
Coimbatore 641006

The candidates with university register numbers 71203106009,71203106023,71203106060,71203106300 were examined by us in the project viva voce examination held on 24.04.07

Internal Examiner

External Examiner

CERTIFICATE

WKS/PSL/P/06-07 09/03/2007

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

Deepak Das
Arvind R
Khoushik. S
Vishnu Venunath

Students of B.E in Electronics & Communication Engineering from Kumaraguru College of Technology, Coimbatore have successfully completed their project work titled " **Networking of PLC**" in our organization from 1st February to 9th March 2007

Their performance and conduct during the period were found good.

For **APOLLO TYRES LTD**

ASSO. MANAGER (HR)

P.O. Perambra, Thrissur Dt., Kerala - 680 689, Tel: (0480) 2725901-09, Fax: 91-480-2725922, Cable:TYREMAKERS
Registered Office: 6th Floor, Cherupushpam Building, Shanmugham Road, Kochi - 682 031, (Kerala).
Tel: (0484) - 2381902-3, Fax: (0484) - 2370351, Cable: APOLLOTYRES

TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	**(v)**
	LIST OF TABLES	**(iv)**
	LIST OF FIGURES	**(v)**
	LIST OF ABBREVATIONS	**(vii)**
1.	**INTRODUCTION**	
	OUTLINE OF THE PROJECT	1
	INTRODUCTION TO COMPANY	2
2.	**PROJECT DESCRIPTION**	
	2.1 EXISTING SYSTEM	3
	2.2 PROBLEM IDENTIFIED	4
	2.3 PROPOSED SOLUTION	5
	2.3.1 NETWORKING USING PROTOCOLS	5
3.	**HARDWARE DESIGN**	
	3.1 LITRATURE REVIEW	6
	3.1.1 NETWORKING	6
	3.1.2 COMPONENTS FOR COMMUNICATION	9
	3.1.3 TOPOLOGY	10
	3.1.3.1 BUS TOPOLOGY	10
	3.1.3.2 RING TOPOLOGY	11
	3.1.4 THE OSI MODEL	12
	3.1.4.1 FUNCTIONS OF OSI LAYERS	13
	3.1.5 PROGRAMMABLE LOGIC CONTROLLER	17
	3.1.5.1 INTRODUCTION	17
	3.1.5.2 DEFINATION TO PLC	18
	3.1.5.3 FUNCTION OF PLC	18

3.1.5.4 PARTS OF A PLC 19
3.1.5.5CONFIGURATION 21
3.1.5.6 CQM1 FEATURES 22
3.1.6 RS232 CONFIGURATION AND FEATURES 27
3.1.7 STUDY OF RS485 30
3.1.7.1 DIFFERENTIAL SIGNALS WITH RS485 32
3.1.7.3. NETWORK TOPOLOGY WITH RS485 36
3.1.7.4. RS 485 FUNCTIONALITY 37
3.1.7.5. ADVANTAGES OF RS485 38
3.1.8 RS232 TO RS485 CONVERTER 38
3.1.8.1 FEATURES 39
3.1.8.2 RS232 TO RS485 CONVERTER SPECIFICATION 39
3.1.9 TOKEN PASSING 40
3.1.9.1 TOKEN FRAME 42
3.1.10 FUNCTIONAL BLOCK DIAGRAM 42

4. **ALGORITHM**
4.1. INITIALIZATION 43
4.2. PRIORITY 43
4.3. TRANSMISSION 44
4.4. RECEPTION 45
4.5. SECURITY 46

5. **FLOWCHART** 47
6. **CODING**

7. **CONCLUSION**

7.1 CONCLUSION 56

7.2 FUTURE SCOPE 57

REFERENCES

APPENDICES

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
3.1	BAUD RATE	29
3.2	FEATURES OF RS 485	32
3.3	COMPARISON OF RS 484 WITH RS 323,RS422	35
4.3.1	FRAME FORMAT	44

LIST OF FIGURES

FIGURE NO.	FIGURE NAMES	PAGE NO.
3.1	NETWORK CRITERIA	7
3.2	DATA COMMUNICATION SYSTEM COMPONENTS	9
3.6	BUS TOPOLOGY	11
3.7	RING TOPOLOGY	11
3.8	LAYER ARCHITECTURE	12
3.9	PARTS OF PLC	19
3.10	PLC ARCHITECTURE	21
3.11	SYSTEM CONFIGURATION	22
3.12	POWER SUPPLY UNIT	23
3.13	INDICATOR MODULE	26
3.14	RS 232 DB 25 PIN OUT	27
3.15	RS 232 DB 9 PIN OUT	28
3.16	NOISE IN STRAIGHT AND TWISTED PAIR CABLES	33
3.17	NEIWORK TOPOLOGY	36
3.18	RS 485 CONVERTER	38
3.19	BLOCK DIAGRAM	42

LIST OF ABBREVATIONS

TXD	TRANSMISSION OF DATA
RXD	RECEPTION OF DATA
AC	ALTERNATING CURRENT
DC	DIRECT CURRENT
RAM	RANDOM ACCESS MEMORY
ROM	READ ONLY MEMORY
EPROM	ERASABLE PROGRAMMABLE READ ONLY MEMORY
I/O	INPUT /OUTPUT
PC	PROGRAM COUNTER
PSW	PROGRAM STATUS WORD
PSEN	PROGRAM STORE ENABLE
TMOD	TIMER MODE CONTROL
DIP	DUAL IN PACKAGE
TTL	TRANSISTOR-TRANSISTOR LOGIC
SFR	SPECIAL FUNCTION REGISTER
ASCII	AMERICAN STANDARS CODE FOR INFORMATION INTERCHANGE
FTAM	FILE TRANSFER, ACCESS, AND MANAGEMENT
bps	BITS PER SECOND
kbps	KILOBITS PER SECOND
EIA	ELECTRONICS INDUSTRIES ASSOCIATION
USB	UNIVERSAL SYNCRONOUS BOARD
COM	COMMON
LED	LIGHT EMITTING DIODE
PLC	PROGRAMMABLE LOGIC CONTROLLER
UART	UNIVERSAL ASYNC RECEIVER/TRANSMITTER

ABSTRACT

ABSTRACT

The project titled "Networking of PLC's" (programmable logic controller) focuses on data transmission and retrieval between them, which helps them to share their functionalities to produce the best results which is of wide use in industries to control machineries.

For e.g. If one plc controls temperature and the other controls pressure. When these two plc's are networked, data sharing takes place. When a change in temperature takes place corresponding change in pressure occurs thus keeping the whole system in control based on the requisites.

The major hardware components:

- OMRON cqm1h PLC
- RS-232 to 485 converter

The Plc's are networked using RS-232c interface cable and multi-port serial data communication is achieved using RS-485 converter.

In the case of networking, a protocol is to be defined. Protocol is a convention or standard that controls or enables the connection, communication, and data transfer between endpoints.

One such protocol is created in our case using the Token bus (IEEE 802.4 standard) methodology with the help of specified software which is dedicated to the particular PLC given by its manufacturer. The software is such developed that the performance, reliability and security are brought to its best level enabling to accomplish the objectives.

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We deem it a great pleasure to place on record our deep sense of gratitude and indebt ness to **Dr Joseph V. Thanikal**, Principal, Kumaraguru College of Technology, for providing us this excellent opportunity to work on this project.

We express our sincere gratitude to **Dr. Rajeshwari Mariappan**, Head of the department of Electronics and Communication Engineering, for the immense concern and encouragement shown during the course of the project.

We profusely thank our project coordinator, **Mr.K.Ramprakash**, Professor, for his moral support and motivation given to us regarding our project work.

We express our deep sense of gratitude to our project guide, **Mrs A.Amaveni M.E**, Senior Lecturer, for guiding us throughout by his constant support, valuable comments, suggestions, and for the inspiration and guidance given to us in molding our project.

It is a matter of privilege and honour for us to place on record **Apollo Tyres Limited**, Perambra for providing us with a great opportunity to work on this project. Our sincere thanks to **Mr.P.Asokan**, Group Manager (Electrical And Electronics) and **Mr.Sateeshan**(Senior HR), for their unstinted co-operation and encouragement at all levels to undergo this project work.

The sense of fulfillment in completing this project would remain but incomplete without us thanking our respected teaching and non teaching staffs, our beloved parents, our intimate friends and family members for their unlimited support in crossing an important milestone in our life.

INTRODUCTION

CHAPTER 1
INTRODUCTION

1.1 OUTLINE OF THE PROJECT:

Automation, the boon for all the industries, as is known by all, means the replacement of human, physical and mental activities by machines compared to mechanization which is the replacement of only human physical labour by machines. So automation eliminates the undesirable of mechanization in which the operator functions as an integral mechanical part of production.

Automation has assumed an increasingly important role in the development and advancement of modern civilization and technology. The design of new equipment with greater usefulness and capabilities is bringing about an ever-increasing growth in the development of automation.

A company can survive in the market only if new product, with accuracy and models are introduced in low cost. The time taken between the customer need and the final finished product to market should be less then only the company can have a good hold on the market and earns profit.

In our project the most effective tool of automation is PLCs and they are try to be networked with the help of a well known protocol. The protocol is used for communicating various peripheral devices.

We went for serial mode as it satisfies the requirement completely. Serial connections use a dedicated serial cable connection between a master and one or more slave devices. A specific standard is used such as RS232 and all connected devices should support and be configured for the same standard.

1.2 INTRODUCTION TO COMPANY

The history of **Apollo Tyres Ltd.** is about a company's passion, determination and will to surpass all obstacles and emerge as a leader in the Indian tyre industry named after the radiant Greek Sun God Apollo; the company has created a niche for itself in the Indian tyre market. After three decades of consistent growth, today, Apollo Tyres Ltd. is India's premier tyre manufacturing company.

Apollo Tyres Ltd's history dates back to the early Seventies. Mr Mathew T Marattukalam, Jacob Thomas and his associates obtained the company's license in 1972. In 1974 the company was taken over by Dr. Raunaq Singh and his associates. The tyre project was implemented in 1976 in Perambra, Kerala. The commercial production began in 1977 with an installed capacity of 420,000 each of tyres and tubes.

History of Apollo tyres

- 1973 - License approved by government
- 1974 - Taken over by Raunaq Singh
- 1975 - Perambra branch started on 13th April
- 1976 - Got recognized
- 1982 - Manufacturing of radial tyres
- 1991 - Second plant at Baroda started
- 1995 - Acquired premier tyres in Kerala
- 2003 - Production of radial capacity of tyres was 6600
- 2004 - Launch of Apollo accelere a tubeless tyres
- 2006 - Acquired Dunlop in South Africa

PROJECT DESCRIPTION

CHAPTER 2

PROJECT DESCRIPTION

2.1 EXISTING SYSTEM

The company was found to be using Omron CQM1 plc for curing process. There 32 curing machines were controlled by 32 individual plc's. These plc's control the sequencing of various operations. These old versions of Omron plc's are not competent to make effective communication between the neighboring controllers. The feeding of programs and other updating where done individually to all plc's. Hence the probability of error occurrence was found to be high. The maintenance was also difficult because the operator don't know how, where and what the problem occurs. The malfunctioning of the plc's was found very different with each other. Finally the operator ends with entangled wires a loss of few precious working hours.

In this competitive world accuracy plays a very important role in customer satisfaction. The accuracy in microns is not possible by higher percentage of human intelligence. Hence the use of automation is essential to match the market standard. The survival of company in the market also depends upon the timings of the company to place the requirements of the customers. This can be only done if the changes in the production sequence and production methodology can be updated quickly. Hence our project deals with networking of the plc's which will be effective and communicating the market requirement and the present working production scenario.

Few new Omron plc's CQM1H were also found which are having communication board and can communicate with other plc's but in the company they used RS 232 to make communication. Hence point-to-point communication is possible networking more than one plc's using RS 232 increasing the density of wires with out achieving a reliable communication system.

The main objective of our project was to bring both the versions CQM1 and CQM1H under reliable networking. So that the company requirements of using both the type of the plc's can be satisfied without any huge loss. This was possible by seeking the communication facility in old Omron CQM1 plc and matching it with the new CQM1H plc. So they can be feed into same network topology under same protocols specification.

2.2 PROBLEM IDENTIFIED

- The old Omron CQM1 plc's were unable to communicate.
- Due to use of RS232 the density of wires was very high.
- Error in applied protocol to higher version of plc's to communicate.
- Updating in controllers individually was difficult.
- Maintenance was found to be complicated

2.3 PROPOSED SOLUTION

2.3.1. NETWORKING USING PROTOCOLS

This was found to be most efficient way of communicating between plc's. In this type of communication the data and the control signals are supposed to pass through seven different layers to make the communication possible. Here in this design OSI model is used as the basic model. The selection of the talker or updating all the system is done by the mechanism called tokenpasing. In this mechanism a frame of control information's are send by the main master plc to the neighboring plc and hence the plc having the token can only be communicated to the master it means the master sends all the data to the token listener. Once the data are updated in the system the token is passed to the next system and again the master sends the data to the new listener. This is how the whole network is updated with out any error. In our project the designed protocol is used to cross all the layers of communication.

HARDWARE DESIGN

CHAPTER 3

HARDWARE DESIGN

3.1 LITRATURE REVIEW

The literature pertaining to the following system, which are employed in this process is presented in the review.

The main concepts studied in this project.

- Networking
- OSI model
- Topologies
- Programmable logic controller
- RS232 and RS485
- RS232 TO RS485 converter
- Protocol

3.1.1 NETWORKING

Networking is the set of devices connected by media link. A node can be a computer, printer, or other devices capable of sending and or receive data generated by other nodes on the network. The links connecting the devices are often called communication channels.

Distributed processing

Networks are the distributed processing, in which a task is divided among multiple computers. Instead of a single large machine being responsible for all aspects of a process, each separate computer handles the subset.

Network criteria

To be considered effective and efficient, a network must meet a number of criteria. The most important of these are performance, reliability, and security.

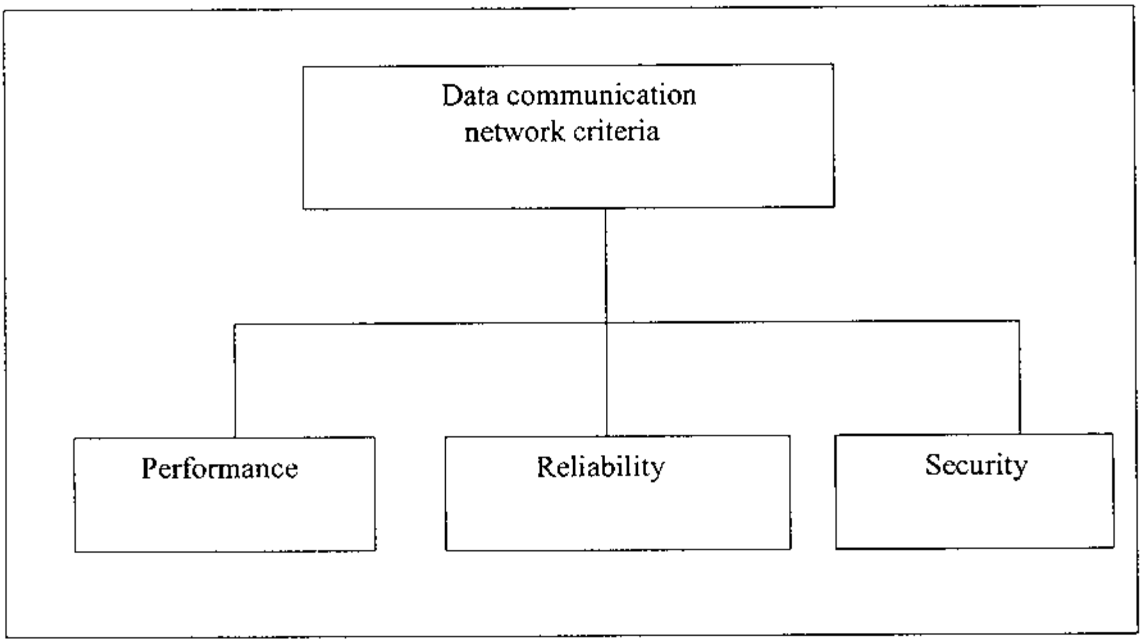


Figure 3.1 Network criteria

Performance

Performance can be measured in many ways, including transmit time and response time. Transmit time is the amount of time required for the message to travel from one drive to another. Response time is the elapsed time between an inquiry and a response.

The performance of the network depends on a number of factors, including the number of users, type of transmission medium, the capability of the connected hardware, and the efficiency of the software.

- Number of users
- Type of transmission medium
- Hardware
- Software

Reliability

In addition to accuracy of delivery, network reliability is measured by frequency of failure the time it takes a link to recover from a failure, and the network's robustness in a catastrophe.

- Frequency of failure
- Recovery time of a network after a failure
- Catastrophe

Security

Network security issues include protected data from unauthorized access and viruses.

- Unauthorized access
- Viruses

3.1.2 COMPONENTS FOR COMMUNICATION

A data communication system is made up of five components

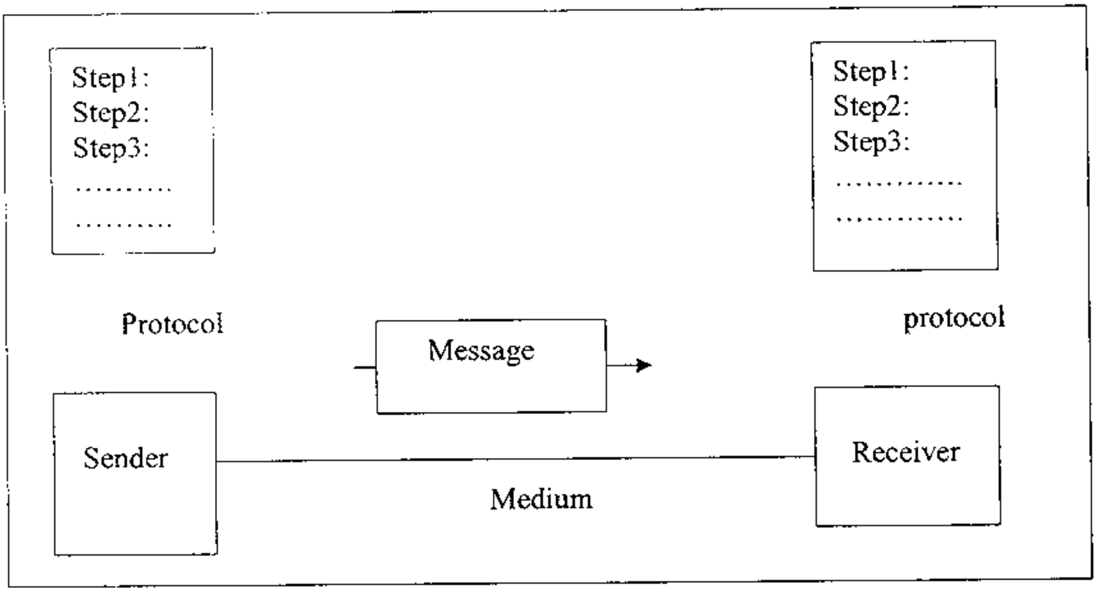


Figure 3.2 Data communication system components

Message: The message is the information to be communicated. It can consist of text, picture, sound or video.

Sender: Is the device that sends the data message. It can be computer, telephone exchange or plc's.

Receiver: The receiver is the device that receives the message. It can be computer or networked plc's.

Medium: The transmission medium is the physical path by which a message travels from senders to receiver. It consists of twisted pair wire, coaxial cable, fiber optics cable, laser, or radio waves.

Protocol: A protocol is a set of rules that governs data communication. It represents an agreement between the communicating devices. Without a protocol, two devices may be connected but not communicating.

3.1.3 TOPOLOGY

The term topology refers to the way a network is laid out, either physically or logically. Two or more devices connect to a link. Two or more link forms a topology. The topology of a network is the geometric representation of the relationship of all the links and the linking devices called nodes. There are five basic topologies possible.

- Mesh
- Star
- Bus
- Tree
- Ring

The topology which we implemented is the token bus topology, which combines the physical configuration of bus topology and collision-free feature of token ring.

3.1.3.1 BUS TOPOLOGY

The preceding example all described point-to-point configurations. A bus topology, on the other hand, is multipoint. One long cable acts as a backbone to link all the devices in the network. Nodes are connected to the bus cable by drop lines and taps. A tap is a connector that either splices into the main cable or punctures the sheathing of a cable to create a contact with the metallic core.

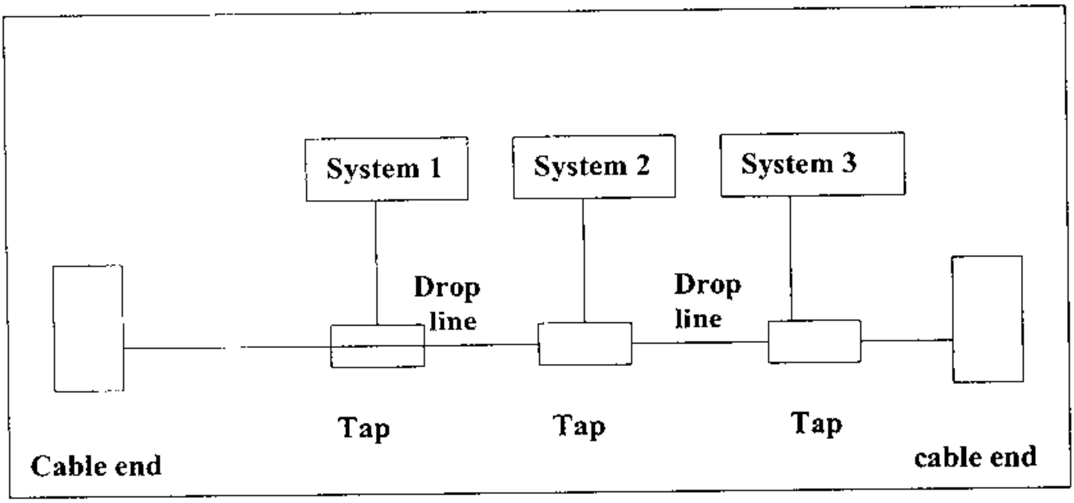


Figure 3.6 Bus topology

3.1.3.2 RING TOPOLOGY

In a ring topology, each device has a dedicated point-to-point line configuration only with the two devices on either side of it. a signal is passed along the ring in one direction, from device to device, until it reaches its destination. Each device in the ring incorporates a repeater. When a device receives a signal intend for another device, its repeater regenerates the bits and passes them along.

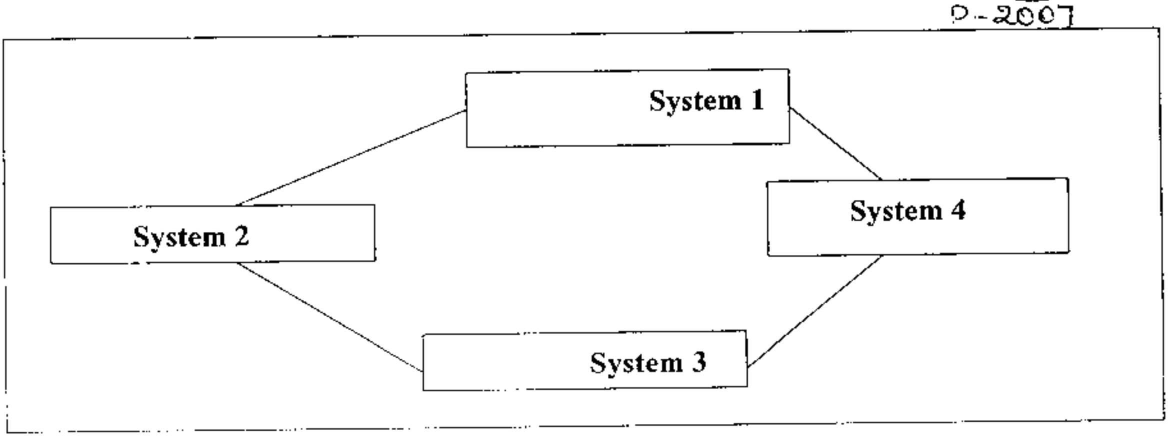


Figure 3.7 Ring topology

3.1.4 THE OSI MODEL

The open system interconnection model is a layered network for the design of the network system that all communication between all the devices present in this world. It consists of seven different but related layers, each of which defines a segment of the process of moving information across the network. The OSI model is built of seven-ordered layer. The layer involved when a message is sent from device A to device B. As the message travels from A to B it may pass through many intermediate nodes. These intermediate nodes usually involve only the first three layers of the OSI model. In developing the model the distilled the processes of transmitting data down to its most fundamental elements.

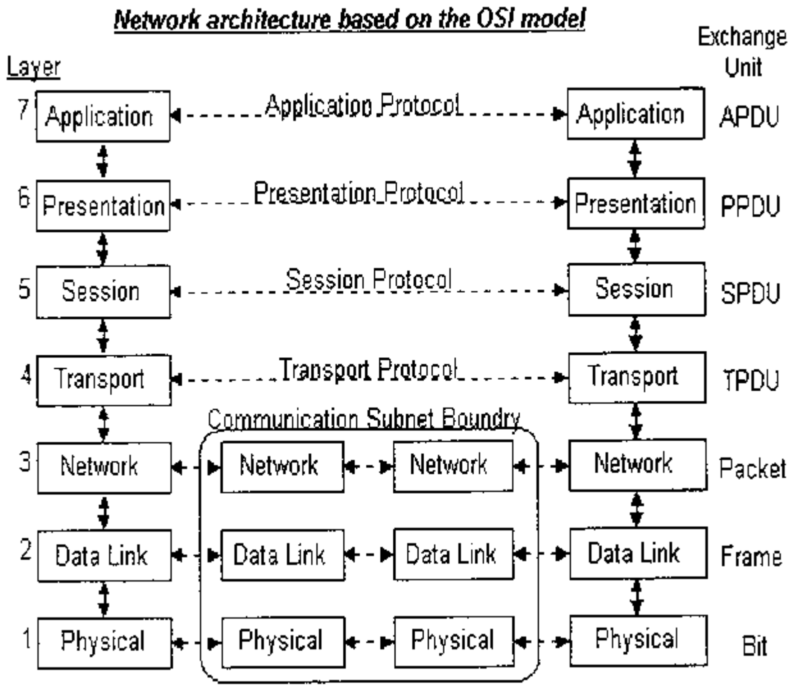


Figure 3.8 Layer Architecture

3.1.4.1 FUNCTIONS OF OSI LAYERS

In this session we briefly describe the functions of each layer in the OSI model.

- Physical Layer
- Data Link Layer
- Network Layer
- Transport Layer
- Session Layer
- Presentation Layer
- Application Layer

Physical layer

The physical layer coordinates the functions required to transmit a bit stream over a physical medium. It deals with mechanical and electrical specifications of the interface and transmission medium. It also defines the functions and procedures that physical devices and interface s have to perform for transmission it occurs.

The physical layer is concerned with the following:

- Physical characteristics of interface and media
- Representation of bits
- Data rate
- Synchronization of bits
- Line configuration
- Physical topology
- Transmission mode

Data link layer

The data link layer transforms the physical layer transforms the physical layer, a raw transmission facility, to a reliable link and is responsible for node-to-node delivery. It makes the physical layer appear error free to the upper layer.

Specific responsibilities of the data link are:

- Framing
- Physical addressing
- Flow control
- Error control
- Access control

Network layer

The network layer is responsible for the source-to-destination delivery of a packet possibly across multiple networks [links]. Whereas the data link layer oversees the delivery of the packet between two systems on the same network (links), the network layer ensures that each packet gets from its point of origin to its final destination

If two systems are connected to the same link, there is usually no need for a network layer. However, if the two systems are attached to different networks with connecting devices between the networks, there is often a need for the network layer to accomplish source-to-destination delivery.

Specific responsibilities of the network layer

- Logical addressing
- Routing

Transport Layer

The transport layer is responsible for source-to- destination delivery of the entire message. Where as the network layer oversees end-to-end delivery of individual packets, it does not recognize any relationship between those packets. It treats each one independently, as though each piece belong to a separate message, Whether or not it does. The transport layer, on the other hand ensures that whole message arrives in tact and in order overseeing both error control and flow control at the source to destination level.

For added security, the transport layer may create a connection between the two end ports. A connection is a single logical path between the source and destination that is associated with all packets in a message. Creating a connection involves three steps connection establishment, data transfer, and connection release. By confining transmission of all packets to a single pathway, the transport layer has more control over sequencing, flow, and error detection and correction.

Specific responsibilities of the transport layer are:

- Service-point addressing
- Segmentation
- Connection control
- Flow control
- Error control

Session layer

The services provided by the first three layers (physical, data link, and net work) are not sufficient for processes. The session layer is the network dialog controller. It establishes, maintains, and synchronizes the between communicating systems.

- Dialog control
- Synchronization

Presentation layer

The presentation layer is concern with the syntax and semantics of the information exchanged between two systems.

Specific responsibilities of the presentation layer include the following:

- Translation
- Encryption
- Compression

Application layer

The application layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services such as electronic mail, remote file access and transfers, shared database management, and other types of distributed information services

Specific services provided by the application layer include the following:

- Network virtual terminal
- File transfer, access, and management (FTAM)
- Mail services
- Directory services

3.1.5 PROGRAMMABLE LOGIC CONTROLLER

3.1.5.1 INTRODUCTION

The most effective tool of automation in industries is plc. It is a automatic multiple relay which is preferable to be operated in robust environment. Before the advent of solid-state logic circuits, logical control systems were designed and built exclusively around electromechanical relays. Relays are far from obsolete in modern design, but have been replaced in many of their former roles as logic-level control devices, relegated most often to those applications demanding high current and/or high voltage switching.

In the late 1960's an American company named Bedford Associates released a computing device they called the MODICON. As an acronym, it meant Modular Digital Controller, and later became the name of a company division devoted to the design, manufacture, and sale of these special-purpose control computers. Other engineering firms developed their own versions of this device, and it eventually came to be known in non-proprietary terms as a PLC, or Programmable Logic Controller. The purpose of a PLC was to directly replace electromechanical relays as logic elements, substituting instead a solid-state digital computer with a stored program, able to emulate the interconnection of many relays to perform certain logical tasks.

A PLC has many "input" terminals, through which it interprets "high" and "low" logical states from sensors and switches. It also has many output terminals, through which it outputs "high" and "low" signals to power lights, solenoids, contactors, small motors, and other devices lending themselves to on/off control. In an effort to make PLCs easy to program, their programming language was designed to resemble ladder logic diagrams. Thus, an industrial electrician or electrical engineer accustomed to reading ladder logic schematics would feel comfortable programming a PLC to perform the same control functions.

PLCs are industrial computers, and as such their input and output signals are typically 120 volts AC, just like the electromechanical control relays they were designed to replace. Although some PLCs have the ability to input and output low-level DC voltage signals of the magnitude used in logic gate circuits, this is the exception and not the rule.

3.1.5.2 DEFINITION TO PLC

As the name implies, PLC performs "Control actions" based on "Desired logic”. As Programmable controller is a solid-state control system with a user programmable memory that reads the input condition and sets the output condition to control a machine or process.

3.1.5.3 FUNCTION OF PLC

The basic function of a PLC is to provide output commands to a machine or process on some combination of a set of input condition to that machine or process.

The PLC may perform timing, counting and other functions dependant on the design of the PLC. The internal wiring of a PLC is fixed and the logical function that it must perform are programmed into a "memory" hence the name "Programmable controller".

3.1.5.4 PARTS OF A PLC

A typical PLC can be divided into parts as illustrated in figure. These components are the

- Central processing unit
- Memory unit
- Input / Output section
- Power supply

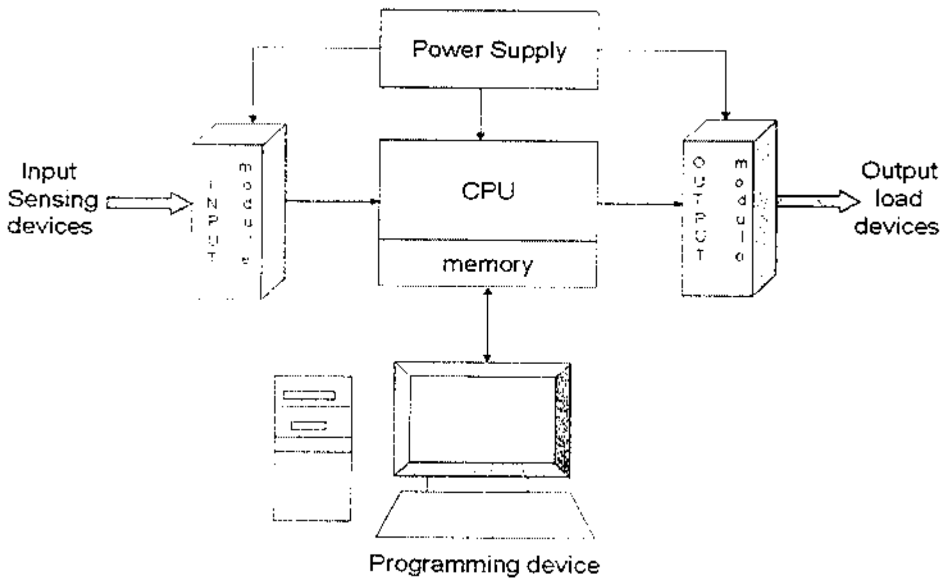


Figure 3.9 Parts of PLC

There are two ways in which I/O is incorporated into the PLC: Fixed and Modular. Fixed I/O is typical of small PLCs that come in one package with no separate, removable units. The processor and I/O are packaged together and the I/O terminals are available but cannot be changed. The main advantage of this type of packaging is lower cost. The number of available I/O points varies and usually can be expanded by buying additional units of fixed I/O. One disadvantage of fixed I/O is its lack of flexibility.

Compartments into which separate modules can be plugged divide modular I/O. This feature greatly increases your options and the unit's flexibility. You can choose from the modules available from the manufacture and mix them anyway you desire. The basic modular controller consists of a rack, power supply, and processor module (CPU), I/O modules, and an operator interface for programming and monitoring. The modules plug into a rack. When a module is slid into a rack, it makes an electrical connection with series of contacts called the Backplane, located at the rear of the rack. The PLC processor is also connected to the backplane and can communicate with all the modules in the rack.

The power supply supplies D.C power to other modules that plug into the rack. For large PLC systems, this power supply does not normally supply power to the field devices. With large systems, power to field devices is provided by external alternating current (A.C) or direct current (D.C) supplies. For small and micro PLC systems, the power supply is used to power field devices

.

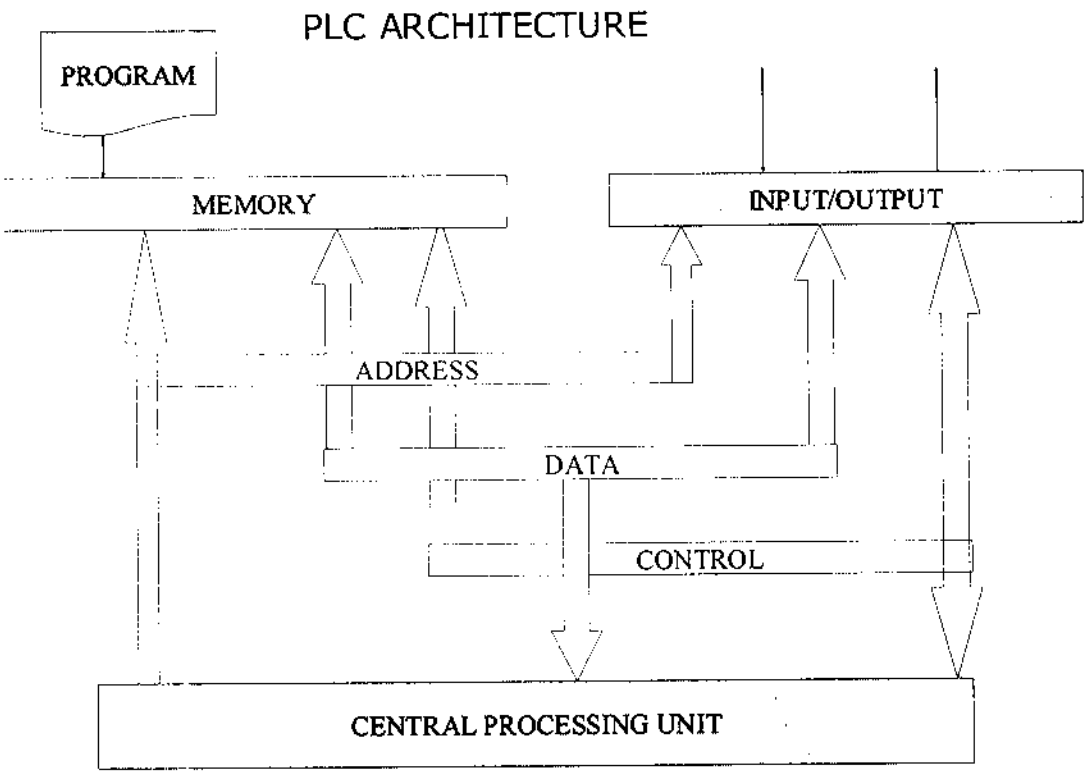


Figure 3.10 PLC Architecture

The processor (CPU) is the brain of the PLC. The typical processor usually consists of a microprocessor for implementing the logic and controlling the communication among the modules. The processor requires memory for storing the results of logical operations performed by the microprocessor. Memory is also required for the program EPROM or EEPROM plus RAM.

3.1.5.5 CONFIGURATION

The CQM1 is a compact, high-speed PC composed of a Power Supply Unit, a CPU Unit, and I/O Units. All of these Units connect at the sides to form a single PC, which is normally mounted to a DIN track. All CQM1 CPU Units, except for the CQM1-CPU11-E, are equipped with an RS-232C port that can be connected directly to a host computer, another CQM1, or other serial devices. The following diagram shows the system configurations possible with the CQM1. Refer to Section 2 Hardware Considerations for more details on system Components and specifications.

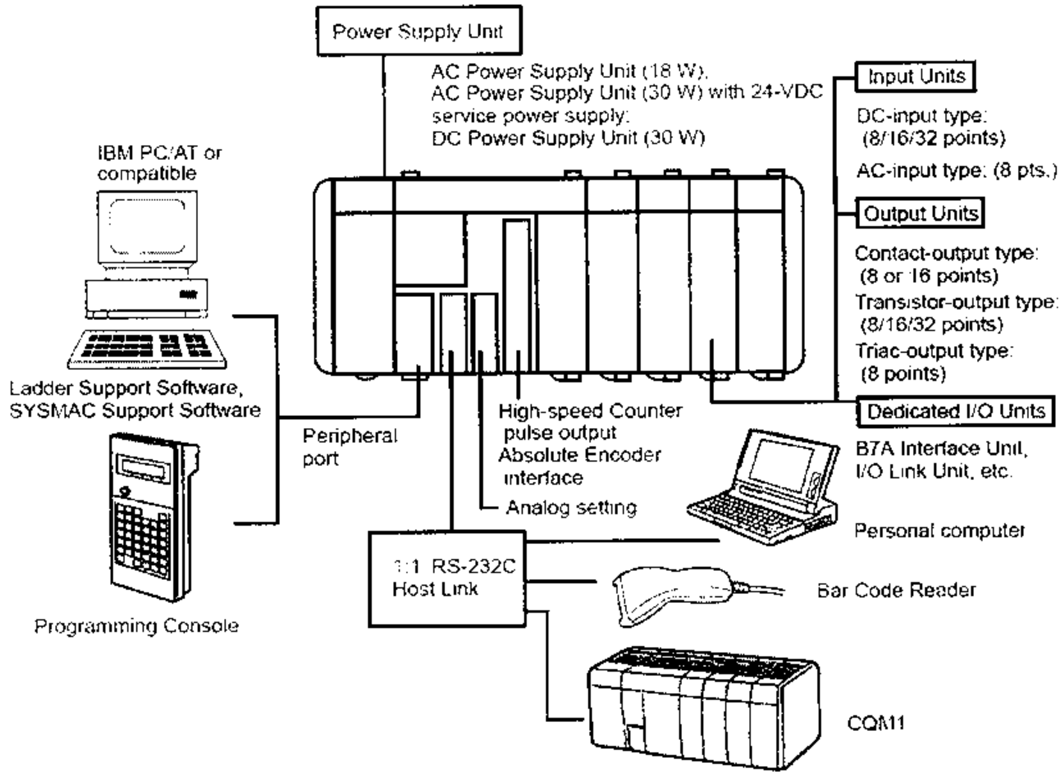


Figure 3.11 system configuration

3.1.5.6 CQM1 FEATURES

The CQM1 provides many advanced features, including the following:

- The CPU Unit provides 16 built-in input terminals.
- I/O Units can be added to increase I/O capacity.
- The CQM1 is much faster: about 20 times faster than P-type PCs.
- High-speed timers and counters are built in.

Power Supply Unit

There are three AC Power Supply Units available, the CQM1-PA203, the CQM1-PA206, and the CQM1-PA216, and one DC, the CQM1-PD026. Select a Power Supply Unit that matches the current consumption of the system.

The following diagram shows the basic components of a Power Supply Unit.

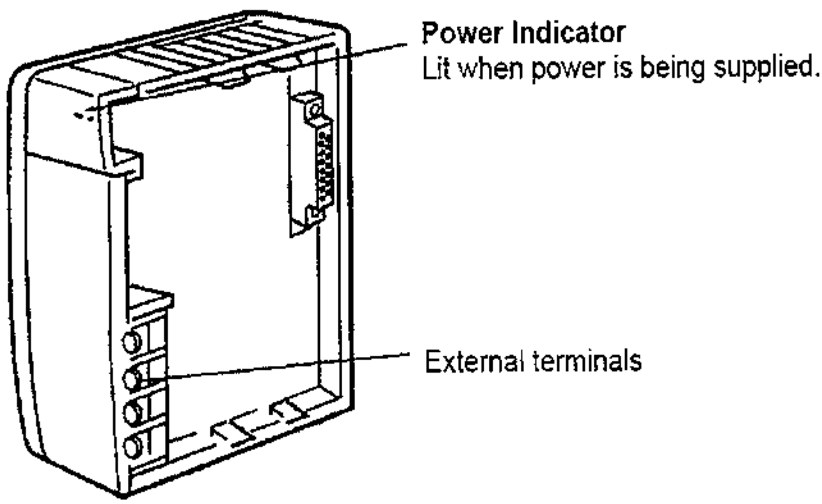


Figure3.12 Power supply unit

Interrupts

- Input interrupts are used to process input signals from an external device this is shorter than the program execution time. Input signals with a pulseWidth as short as 0.1 ms can be used.

- Scheduled interrupts can be performed using a high-speed interval timer.
- Single-phase pulses up to 5 kHz and two-phase pulses up to 2.5 kHz can beInput. High-speed counter interrupts can be combined with pulse outputs for
  Applications such as motor control. The CQM1-CPU43-EV1 and CQM1-CPU44-EV1 can accept single-phase pulses up to 50 kHz and two phases
  pulses up to 25 kHz. The high-speed counter has two points added.

Communications

A peripheral port and RS-232C port are available and are used to communicate with external devices using the following methods.

- Host Link

  The CQM1 using the host link can communicate with a personal computer and Programmable Terminal using host link commands.
- RS-232C

  The CQM1 using the RS-232C can read data from a bar code reader or measurement device and output data to a printer.

- 1-to-1 Link

  A data link can be created with a data area in another CQM1 to monitor the other PC's status and synchronize processes controlled by the PCs.

Convenient I/O Instructions

A single instruction can be used to input or output data, simplifying the program.

- The TEN KEY INPUT instruction can be used to read 8-digit BCD data input
  from a ten-key.
- The HEXADECIMAL KEY INPUT instruction can be used to read 8-digit hexadecimal
  key input data from I/O Units.
- The DIGITAL SWITCH instruction can be used to read 4 or 8-digit BCD data
  from digital switches.
- The 7-SEGMENT DISPLAY OUTPUT instruction can be used to output 4 or
  8-digit data to 7-segment displays

Macros

The Macro instruction can be used to call and execute subroutines, designating the I/O word for the subroutine as an argument. Using an argument to specify a subroutine I/O words allows subroutines to be used more easily in different locations, simplifying the program.

CPU Unit

The CQM1 is a compact, high-speed PC made up of a CPU Unit, Power SupplyUnit, and I/O Units that together provide up to 256 total I/O points.

The CQM1-CPU42-EV1 CPU Unit provides a built-in analog setting function. It has four dedicated volume controls, and their respective values

(0 to 200 BCD) appear in words 220 to 223. This function can be used for operations such as changing timer and counter set values during operation. The CQM1-CPU43-EV1 CPU Unit provides a built-in pulse input and output function. It has two dedicated ports for high-speed counting of up to 25-kHz two-phase pulse inputs from a device such as a rotary encoder and outputting up to 50-kHz pulses to a device such as a stepping motor. The CQM1-CPU44-EV1 has two ABS interfaces that can directly receive inputs from absolute-type rotary encoders.

Indicators

CPU Unit indicators provide visual information on the general operation of the PC. Although not substitutes for proper error programming using the flags and other error indicators provided in the data areas of memory, these indicators provide ready confirmation of proper operation. CPU Unit indicators are shown below and are described in the following table.

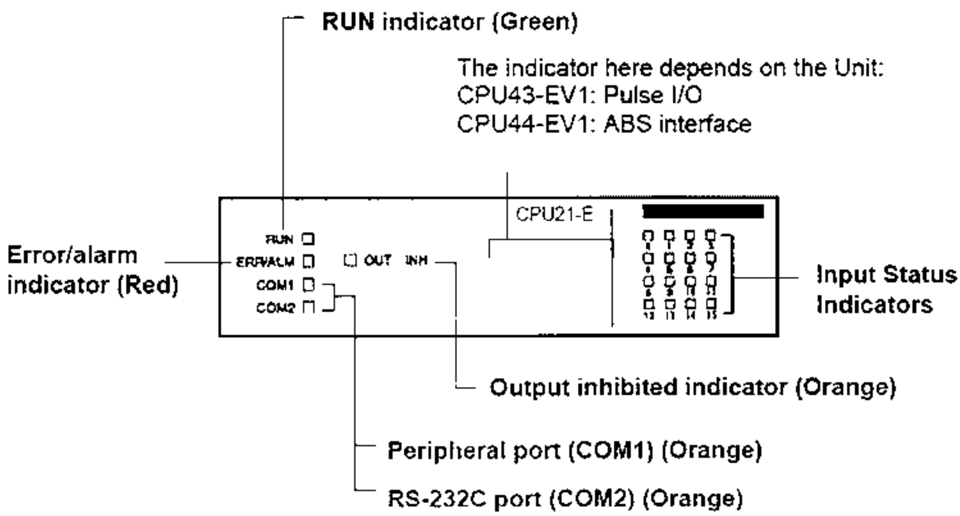


Figure 3.13 Indicator module

3.1.6 RS232 PIN CONFIGURATION AND FEATURES

The RS232 connector was originally developed to use 25 pins. In this DB25 connector pinout provisions were made for a secondary serial RS232 communication channel. In practice, only one serial communication channel with accompanying handshaking is present. Only very few computers have been manufactured where both serial RS232 channels are implemented. Also on a number of Telebit modem models the secondary channel is present. It can be used to query the modem status while the modem is on-line and busy communicating. On personal computers, the smaller DB9 version is more commonly used today. The diagrams show the signals common to both connector types in black. The defined pins only present on the larger connector are shown in red. Note, that the protective ground is assigned to a pin at the large connector where the connector outside is used for that purpose with the DB9 connector version.

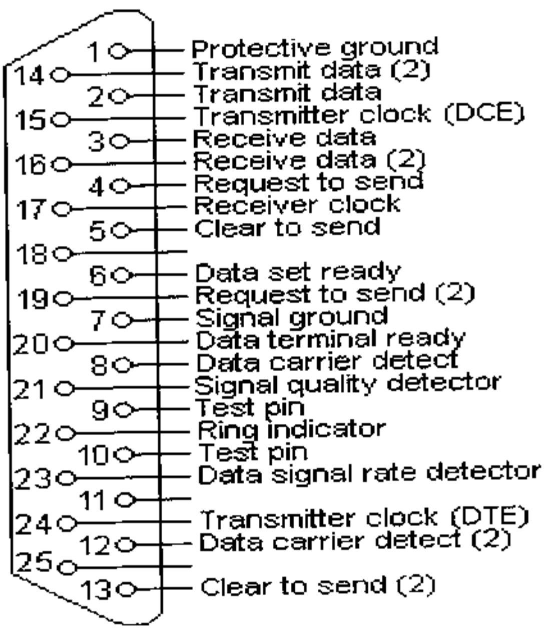


Figure 3.14 RS232 DB 25 pin out

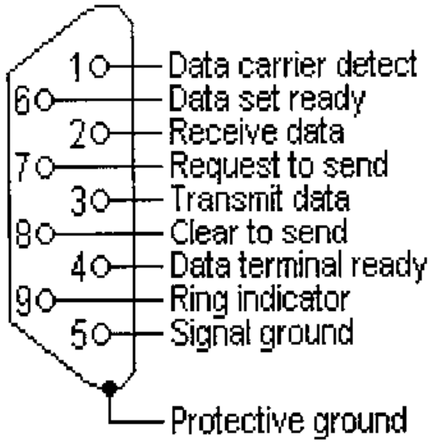


Figure 3.15 RS232 DB 9 pin out

Serial data is any data that is sent one bit at a time using a single electrical signal. In contrast, parallel data is sent 8, 16, 32, or even 64 bits at a time using a signal line for each bit. Data that is sent without the use of a master clock is said to be asynchronous serial data.

Several communications standards exist for the transfer of asynchronous serial data. Common PC's transfer data using the EIA RS-232C (also known as V.28 or V.24). Updated versions of this standard include RS-232D and EIA/TIA-232E, but most literature still refers to the RS-232C or RS-232 standard.

The baud rate for a serial connection is the number of bits that are transmitted per second. It is specified in bits/second or baud. For example, a 110 baud serial link transfers 110 bits of data per second.

The EIA RS-232C standard permits data rates up to 19200 bps and cable lengths up to 400 meters

data rate (bps)	maximum distance (meters)	(feet)
19200	15	45
9600	25	76
4800	50	152
2400	100	304
1200	200	608
600	400	1216

Table 3.1 Baud rate

Data transfer rate

The rate of data transfer in serial data communication is stated in bps(bits per second). Another widely used terminology for bps is baud rate. Baud rate is defined as the number of signal changes per second. In modems, there are occasions when a single change of signal transfers several bits of data. As far as the conductor wire is concerned, the baud rate and bps are the same. It must be noted that in asynchronous serial data communication, the baud rate is generally limited to 100,000 bps.

RS232 standards

To allow compatibility among data communication equipment made by various manufactures, an interfacing standard called RS232 was set by the Electronics Industries Association (EIA) in 1960. RS232 is the most widely used serial I/O interfacing standard.

Since the RS232 standard was set long before the advent of the TTL logic family, its input and output voltage levels are not TTL compatible. In RS232, a 1 is represented by

-3 to -25 V, while a 0 bit is +3 to +25 V, making -3 to +3 undefined. For this reason, to connect any Rs232 to a microcontroller system we must use voltage converters such as MAX232 to convert any RS232 the TTL logic levels to the RS232 voltage level, and vice versa. MAX232 IC chips are commonly referred to as line drivers.

MAX232

Since the RS232 is not compatible with today's microprocessors and microcontroller, we need a line driver (voltage converter) to convert the RS232's signals to the TTL voltage levels that will be acceptable to the 8051's TXD and RXD pins. The MAX232 converts from RS232 voltage levels to TTL voltage levels, and vice versa. One advantage of the MAX232 chip is that it uses a +5 V power source that is the same as the source voltage for the 8051.

3.1.7 STUDY OF RS485

When a network needs to transfer small blocks of information over long distances, RS485 is often the interface of choice. The network nodes can be PCs, microcontroller, or any devices capable of asynchronous serial communications. Compared to Ethernet and other network interfaces, RS-485's hardware and protocol requirements are simpler and cheaper.

The RS-485 standard is flexible enough to provide a choice of drivers, receivers, and other components depending on the cable length, data rate, number of nodes, and the need to conserve power.

The interface popularly known as RS 485 is an electrical specification for multipoint systems that use balanced lines. RS-485 is similar to RS-422, but RS-422 allows just one driver with multiple receivers whereas RS-485 supports multiple drivers and receivers. The specification document (TIA/EIA-485-A) defines the electrical characteristics of the line and its drivers and receivers.

An RS-485 network can have up to32 unit loads, with one unit load equivalent to an input impedance of 12k. By using high-impedance receivers, you can have as many as 256 nodes. An RS-485 link can extend as far as 4000¢ and can transfer data at up to 10 Mbps, but not both at the same time. At 90 kbps, the maximum cable length is 4000¢, at 1 Mbps it drops to 400¢, and at 10 Mbps it drops to 50¢. For more nodes or long distances, you can use repeaters that regenerate the signals and begin a new RS-485 line. Although the RS-485 standard says nothing about protocols, most RS-485 links use the familiar asynchronous protocols supported by the UARTs in PCs and other computers.

A transmitted word consists of a start bit followed by data bits, an optional parity bit, and a stop bit. Two ways to add RS-485 to a PC are on an expansion card and by attaching an RS-485 converter to an existing port. Converters for RS-232 are widely available and Inside Out Networks has developed a USB–to–RS-485 converter, also available from B&B Electronics. On microcontrollers, you can connect an RS-485 transceiver.

Typical Data rates:	Up to 115,200 baud
Max Distance:	4000 feet
Cabling Requirements:	Twisted Pair plus signal ground
Duplex:	Half or Full depending on configuration
Multidrop:	Up to 32 nodes
RS-485 Tech Support Tip:	Always run a signal ground with RS-485
Common Applications:	Industrial Equipment, Modbus, Machine to Machine (M2M) communications

Table 3.2 Features of RS485

3.1.7.1 DIFFERENTIAL SIGNALS WITH RS485

Longer distances and higher bit rates

One of the main problems with **RS232** is the lack of immunity for noise on the signal lines. The transmitter and receiver compare the voltages of the data- and handshake lines with one common zero line. Shifts in the ground level can have disastrous effects. Therefore the trigger level of the **RS232** interface is set relatively high at ±3 Volt. Noise is easily picked up and limits both the maximum distance and communication speed. With **RS485** on the contrary there is no such thing as a common zero as a signal reference. Several volts difference in the ground level of the **RS485** transmitter and receiver does not cause any problems. The **RS485** signals are floating and each signal is transmitted over a **Sig+** line and a **Sig-** line. The **RS485** receiver compares the *voltage difference* between both lines, instead of the *absolute voltage level* on a signal line. This works well and prevents the existence of ground loops, a common source of communication problems. The best results are achieved if the **Sig+** and **Sig-** lines are twisted. The image below explains why.

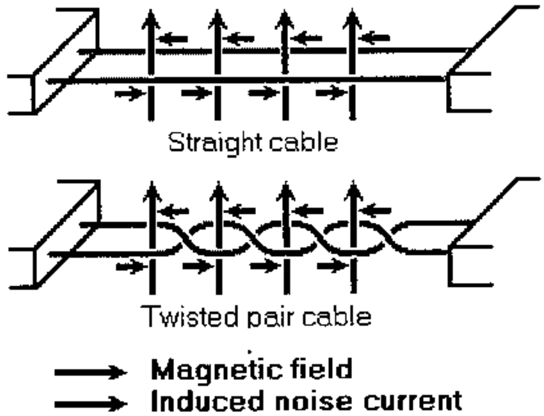


Figure 3.16 Noise in straight and twisted pair cables

In the picture above, noise is generated by magnetic fields from the environment. The picture shows the magnetic field lines and the noise current in the **RS485** data lines that is the result of that magnetic field. In the straight cable, all noise current is flowing in the same direction, practically generating a looping current just like in an ordinary transformer. When the cable is twisted, we see that in some parts of the signal lines the direction of the noise current is the opposite from the current in other parts of the cable. Because of this, the resulting noise current is many factors lower than with an ordinary straight cable. Shielding—which is a common method to prevent noise in **RS232** lines—tries to keep hostile magnetic fields away from the signal lines. Twisted pairs in **RS485** communication however add immunity which is a much better way to fight noise. The magnetic fields are allowed to pass, but do no harm. If high noise immunity is needed, often a combination of twisting and shielding is used as for example in **STP**, *shielded twisted* *pair* and **FTP**, *foiled twisted pair* networking cables. Differential signals and twisting allows **RS485** to communicate over much longer communication distances than achievable with **RS232**. With **RS485** communication distances of 1200 m are possible.

Differential signal lines also allow higher bit rates than possible with non-differential connections. Therefore **RS485** can overcome the practical communication speed limit of **RS232**. Currently **RS485** drivers are produced that can achieve a bit rate of 35 mbps.

What does all the information in this table tell us? First of all we see that the speed of the differential interfaces **RS422** and **RS485** is far superior to the single ended versions **RS232** and **RS423**. We also see that there is a maximum slew rate defined for both **RS232** and **RS423**. This has been done to avoid reflections of signals. The maximum slew rate also limits the maximum communication speed on the line. For both other interfaces—**RS422** and **RS485**—the slew rate is indefinite. To avoid reflections on longer cables it is necessary to use appropriate termination resistors.

3.1.7.2 CHARACTERISTICS OF RS485 COMPARED TO RS232, RS422

	RS232	RS422	RS485
Differential	no	yes	yes
Max number of drivers Max number of receivers	1 1	1 10	32 32
Modes of operation	half duplex full duplex	half duplex	half duplex
Network topology	point-to- point	multidrop	multipoint
Max distance (acc. standard)	15 m	1200 m	1200 m
Max speed at 12 m Max speed at 1200 m	20 kbs (1 kbs)	10 Mbs 100 kbs	35 Mbs 100 kbs
Max slew rate	30 V/µs	n/a	n/a
Receiver input resistance	3..7 kΩ	□ 4 kΩ	□ 12 kΩ
Driver load impedance	3..7 kΩ	100 Ω	54 Ω
Receiver input sensitivity	±3 V	±200 mV	±200 mV
Receiver input range	±15 V	±10 V	-7..12 V
Max driver output voltage	±25 V	±6 V	-7..12 V
Min driver output voltage (with load)	±5 V	±2.0 V	±1.5 V

Table 3.3 comparison of RS485 with RS232, RS422

We also see that the maximum allowed voltage levels for all interfaces are in the same range, but that the signal level is lower for the faster interfaces. Because of this **RS485** and the others can be used in situations with a severe ground level shift of several volts, where at the same time high bit rates are possible because the transition between logical **0** and logical **1** is only a few hundred-milli volts.

Interesting is, that **RS232** is the only interface capable of full duplex communication. This is, because on the other interfaces multiple senders share by multiple receivers and—in the case of RS485—the communication channel. **RS232** has a separate communication line for transmitting and receiving which—with a well-written protocol—allows higher effective data rates at the same bit rate than the other interfaces. The request and acknowledge data needed in most protocols does not consume bandwidth on the primary data channel of **RS232**.

3.1.7.3 NETWORK TOPOLOGY WITH RS485

Network topology is probably the reason why **RS485** is now the favorite of the four mentioned interfaces in data acquisition and control applications. **RS485** is the only of the interfaces capable of internetworking multiple transmitters and receivers in the same network. When using the default **RS485** receivers with an input resistance of 12 kΩ it is possible to connect 32 devices to the network. **RS485** repeaters are also available which make it possible to increase the number of nodes to several thousands, spanning multiple kilometers. And that with an interface which does not require intelligent network hardware: the implementation on the software side is not much more difficult than with **RS232**. It is the reason why **RS485** is so popular with computers, PLCs, micro controllers and intelligent sensors in scientific and technical applications.

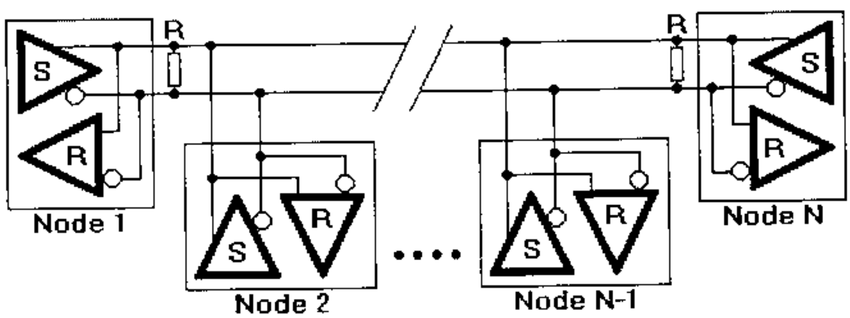


Figure 3.17 Network topology

In the picture above, the general network topology of **RS485** is shown. N nodes are connected in a multipoint **RS485** network. For higher speeds and longer lines, the termination resistances are necessary on both ends of the line to eliminate reflections. Use 100 Ω resistors on both ends. The **RS485** network must be designed as one line with multiple drops, not as a star. Although total cable length maybe shorter in a star configuration, adequate termination is not possible anymore and signal quality may degrade significantly.

3.1.7.4 RS485 FUNCTIONALITY

And now the most important question, how does **RS485** function in practice? Default, all the senders on the **RS485** bus are in tri-state with high impedance. In most high-level protocols, one of the nodes is defined as a master, which sends queries or commands over the **RS485** bus. All other nodes receive these data. Depending of the information in the sent data, zero or more nodes on the line respond to the master. In this situation, bandwidth can be used for almost 100%. There are other implementations of **RS485** networks where every node can start a data session on its own. This is comparable with the way ethernet networks function. Because there is a chance of data collision with this implementation, theory tells us that in this case only 37% of the bandwidth will be effectively used. With such an implementation of a **RS485** network it is necessary that there is error detection implemented in the higher-level protocol to detect the data corruption and resend the information at a later time.

There is no need for the senders to explicitly turn the **RS485** driver on or off. **RS485** drivers automatically return to their high impedance tri-state within a few microseconds after the data has been sent. Therefore it is not needed to have delays between the data packets on the **RS485** bus.

3.1.7.5 ADVANTAGES OF RS485

- Connect DTE's directly without the need of modems
- Connect several DTE's in a network structure
- Ability to communicate over longer distances
- Ability to communicate at faster communication rates

Figure 3.18 RS485 converter

3.1.8 RS 232 TO RS485 CONVERTER

The RS232 to RS485 Converter allows access to a multi-drop RS485 network or connection to a point-to-point RS485 device. Our converter allows up to 32 RS485 (2-wire) devices to be connected to a standard PC

COM port. It offers an "auto turnaround" feature allowing software developers to utilize standard COM drivers. No fancy DLLs required.

3.1.8.1 FEATURES

- 8-24VDC operation
- Simple screw-terminal connections
- Connects to standard PC COM port
- 32 RS485 Devices (or loads) allowed
- Baud Rates up to 115,200.

The connection is a “straight through” RS232 connection ready to plug directly into a PC COM port. There are no handshaking requirements. Since the RS485 side of the converter is a half-duplex differential pair, any characters that are transmitted by the PC or host device via RS232 will get an echo of each character transmitted. This is useful in detecting collisions on the RS485 network there is no address generation or protocol within the device, so the host has the ability to utilize whatever protocol is required for the system configuration you are connecting to. It can be as simple or as complex as you need. There is also an auto-turn-around feature in that the RS485 driver is only enabled during active data bits. This eliminates buffer enable/disable glitches as well as blind-intervals while the buffer data direction is changing.

3.1.8.2 RS232 TO RS485 CONVERTER SPECIFICATION

- DB9 RS232 connector compatible with PC
- TXD and RXD translated from RS232 to RS485
- Direction of transmission controlled by PC RTS line

- Handshake loop the PC connection so it works with all software
- RS485 signals output on 6 way modular jack
- Indicator LED(s) to show communications traffic
- Power supplied by external unregulated 9v plug pack
- RS485 Termination resistor, jumper selectable
- Pullup/Pulldown resistors on RS485 to establish line-idle condition

As the design proceeded we added a couple of extra features over and above the specification, for the sake of completeness they were: RS485 Transmission not just controlled by RTS, but also switched to 'Transmit' whenever the RS232 TXD line goes to space (active). While special software running on the PC should use the RTS line to control RS485 TXD, passive mark pulling and RS485 TXD activated by RS232 space allows a quick and easy test of RS485 serial COMs with existing software.

3.1.9 TOKEN PASSING

Token passing is a channel access method where a "token" is passed around between nodes that authorizes the node to communicate. Token passing schemes are a technique in which only the system which has some "token" can communicate. The token is a control mechanism which gives authority to the system to communicate or use the resources of that network. Once the communication is over, the token is passed to the next candidate in a sequential manner. The most well-known examples are token ring and ARCNET.

Token bus is token ring over a virtual ring on a coaxial cable. A token is passed around the network nodes and only the node possessing the token may transmit. If a node doesn't have anything to send, the token is passed on to the next node on the virtual ring. Each node must know the address of its neighbour in the ring, so a special protocol is needed to notify the other nodes of connections to, and disconnections from, the ring. Token bus was standardized by the IEEE 802.4 Working Group. It is mainly used for industrial applications.

Stations on a token ring LAN are logically organized in a ring topology with data being transmitted sequentially from one ring station to the next with a control token circulating around the ring controlling access. Physically, a token ring network is wired as a star, with 'hubs' and arms out to each station and the loop going out-and-back through each.

More technically, token ring is a local area network protocol which resides at the data link layer of the OSI model. It uses a special three-byte frame called a token that travels around the ring. Token ring frames travel completely around the loop.

Each station passes or repeats the special token frame around the ring to its nearest downstream neighbour. This token-passing process is used to arbitrate access to the shared ring media. Stations that have data frames to transmit must first acquire the token before they can transmit them.

Token ring LAN speeds of 4 Mbit/s, 16 Mbit/s, 100 Mbit/s and 1 Gbit/s have been standardized by the IEEE 802.5 working group.

Token ring networks had significantly superior performance and reliability compared to early shared-media implementations of Ethernet (IEEE 802.3), and were widely adopted as a higher-performance alternative to shared-media Ethernet.

3.1.9.1 TOKEN FRAME

When no station is transmitting a data frame, a special token frame circles the loop. This special token frame is repeated from station to station until arriving at a station that needs to transmit data. When a station needs to transmit a data frame, it converts the token frame into a data frame for transmission.

3.1.10 FUNCTIONAL BLOCK DIAGRAM

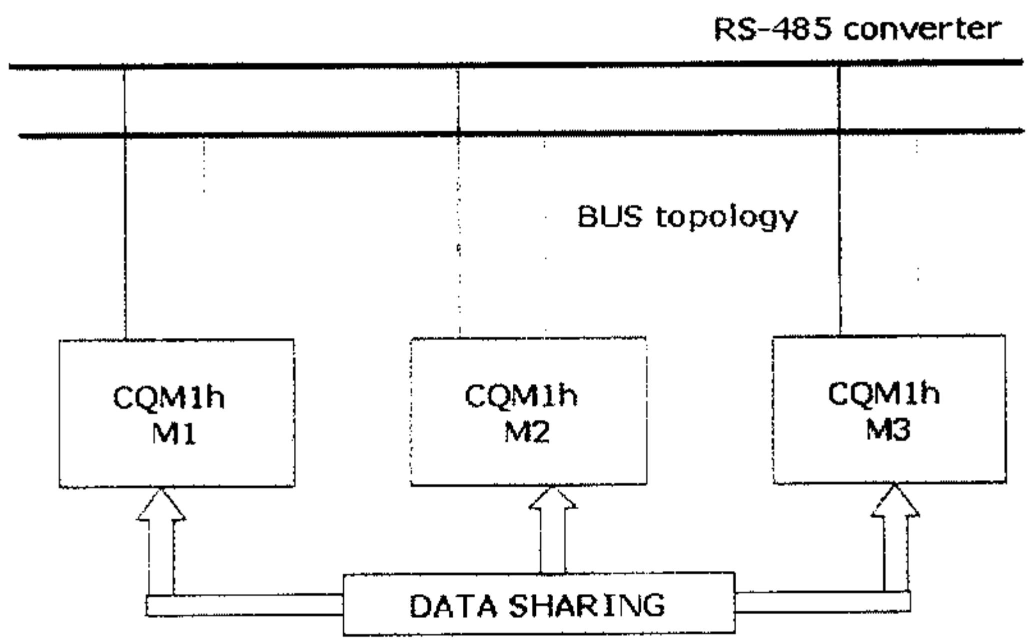


Figure 3.19 Functional Block Diagram

ALGORITHM

CHAPTER 4

ALGORITHM

- Initialization
- Priority
- Transmission
- Reception and Error checking
- Security

4.1 INITIALIZATION

- Hardware is checked to be perfect.
- PLC's are initialized to perform communication by defining communication registers with the required bit selection.
- The baud rate of all PLC's is set to 9600bps; this is achieved by using the software (WLP soft).
- The node ID's of the PLC's are fixed by us. By storing the value in register.

4.2 PRIORITY

- PLC's are checked whether transmission or reception is "ON"
- If there is no such functionality then the internal clock is activated.
- The clock is stopped whenever transmission is required.
- The clock pulse values are checked in order to fix priority.

- The PLC with the less number clock pulse has the most priority and it indicates the PLC to start transmission after the current transmission is over.

4.3 TRANSMISSION

- The standard data (ex: temperature) is allowed to be stored in a particular register.
- That particular data is always checked for its constancy.
- A loop is run such that the standard data when changed follows two steps
    1. The changed values transmitted and
    2. The changed value is copied to the data position which is used for comparison.
- Now the transmission is done in sixteen bits per transaction.
- A request is send to the receiver to ask if it is free when acknowledgement is received back then follow

Start bit	Node id	Destination address	Source id	Data	LRC	Stop bit

Figure 4.3.1Frame Format

- If there is any change in values the start bit is transmitted. Following which node ID which is default set by us in specified address location is transmitted.
- If negative acknowledgement is received node ID is changed and sends.
- Destination address is also default value already stored by the user in the PLC which is address location in the receiving PLC where the change in data required.
- Next source ID is send and then sends the content of the address location of changed data.
- The LRC value of the sent data is calculated and it is stored in a particular location,
- If data to be send is over then end of transmission or abort message is send.
- If data to be send is present above steps are repeated again.

4.4 RECEPTION

- It gives acknowledgement signal to the transmitter when requested for.
- If start bit is received then it receives the following node ID and stores it in a particular location.
- The node ID compares with the PLC's node id.
- Note: The destination address is stored in a variable x. the source address is stored in another variable Y.

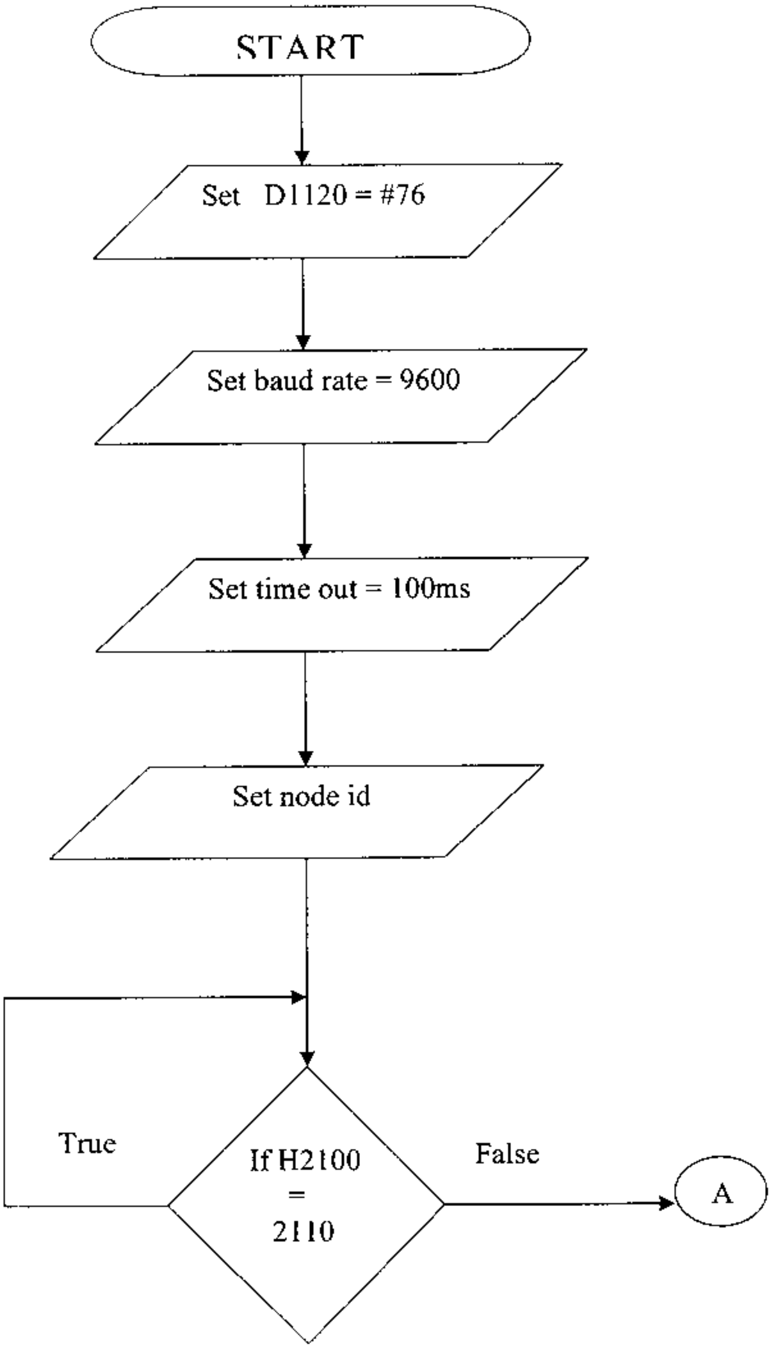
- If it is true the next set of data is received. Else a negative acknowledgment is sent using a variable Y,
- The next set of data is received and stored in the destination address which is stored in the variable X.
- The LRC value of data is calculated,
- The LRC value is sent to the transmitter PLC
- The LRC value in the transmitter PLC is compared with the LRC value which is received by the transmitter PLC.
- If the values are same send acknowledgement to the receiver else send negative acknowledgement to the receiver if acknowledgement is received.
- Then the data is correct else the whole function is repeated.
- Check whether the end of transmission is received.
- If yes terminate the operation else receive the next set of data.

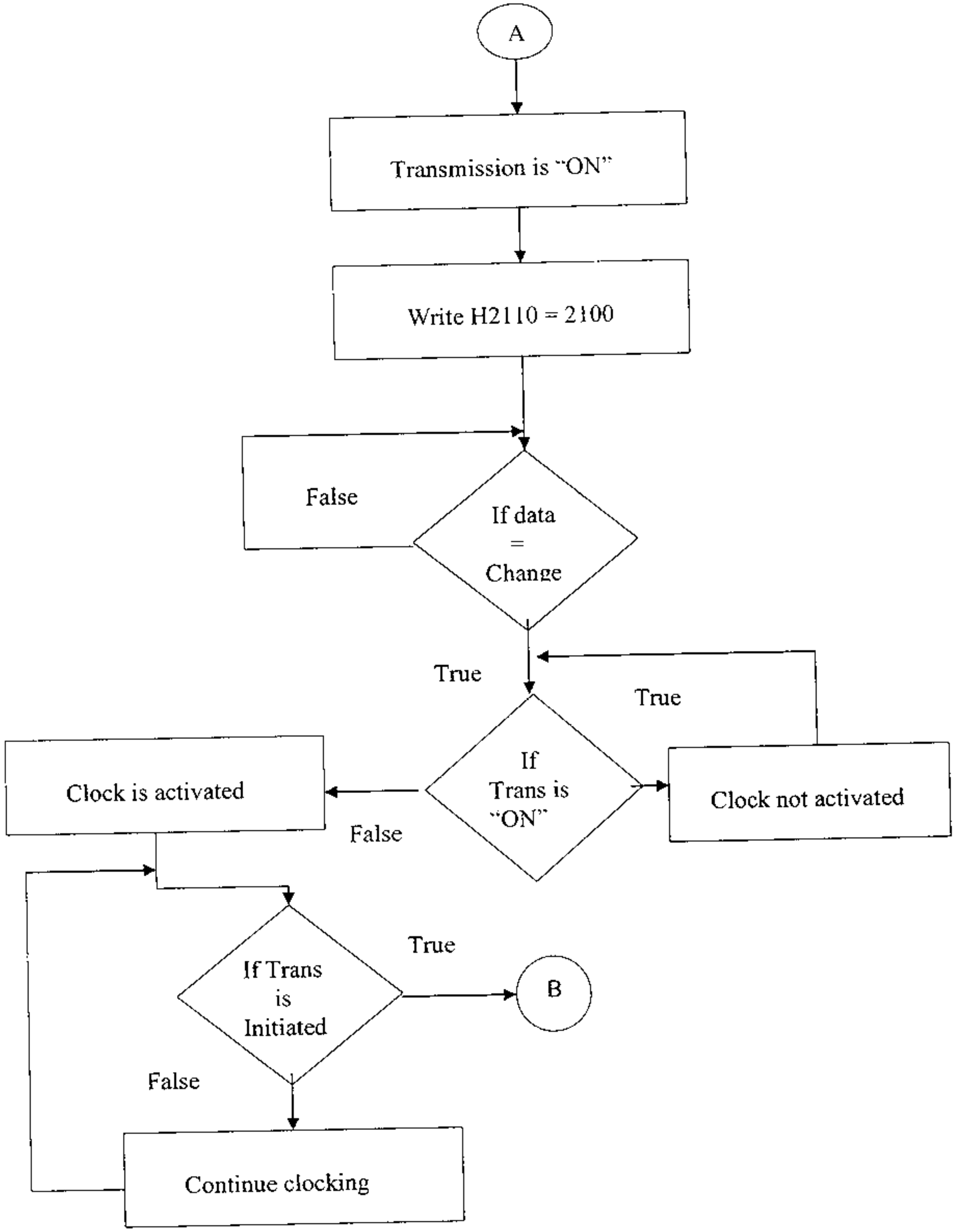
4.5 SECURITY

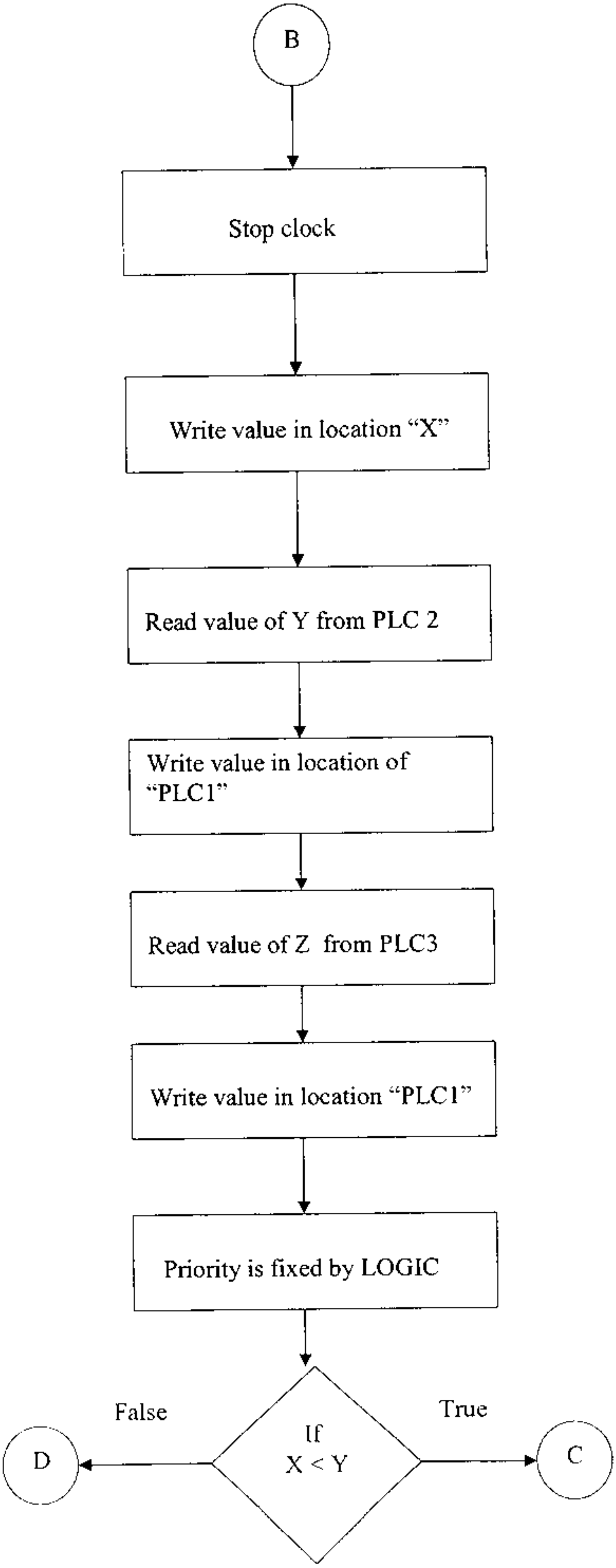
- A key is used to encrypt the data before transmission.
- The same key is used to decrypt the data in the receiver.

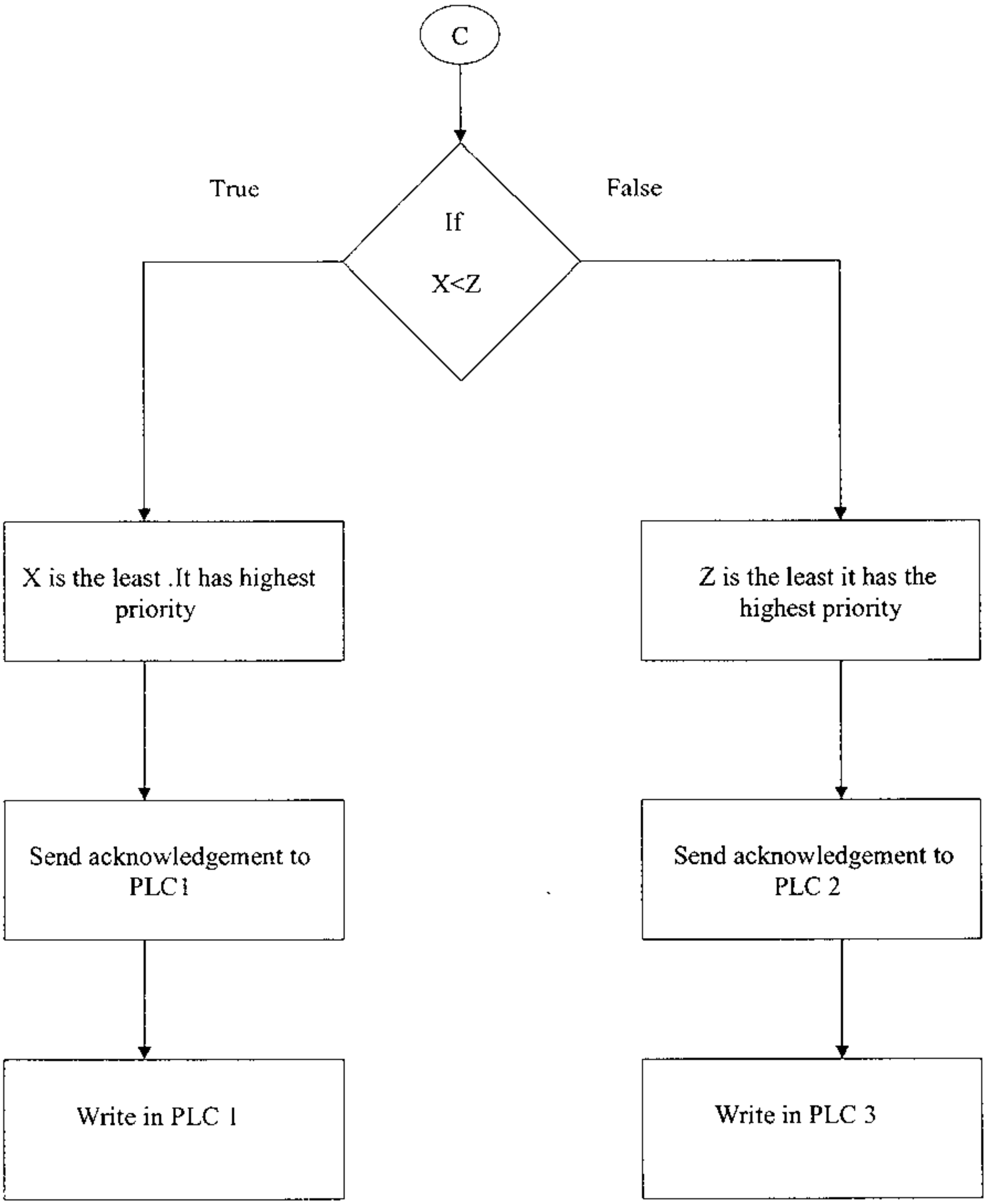
FLOW-CHART

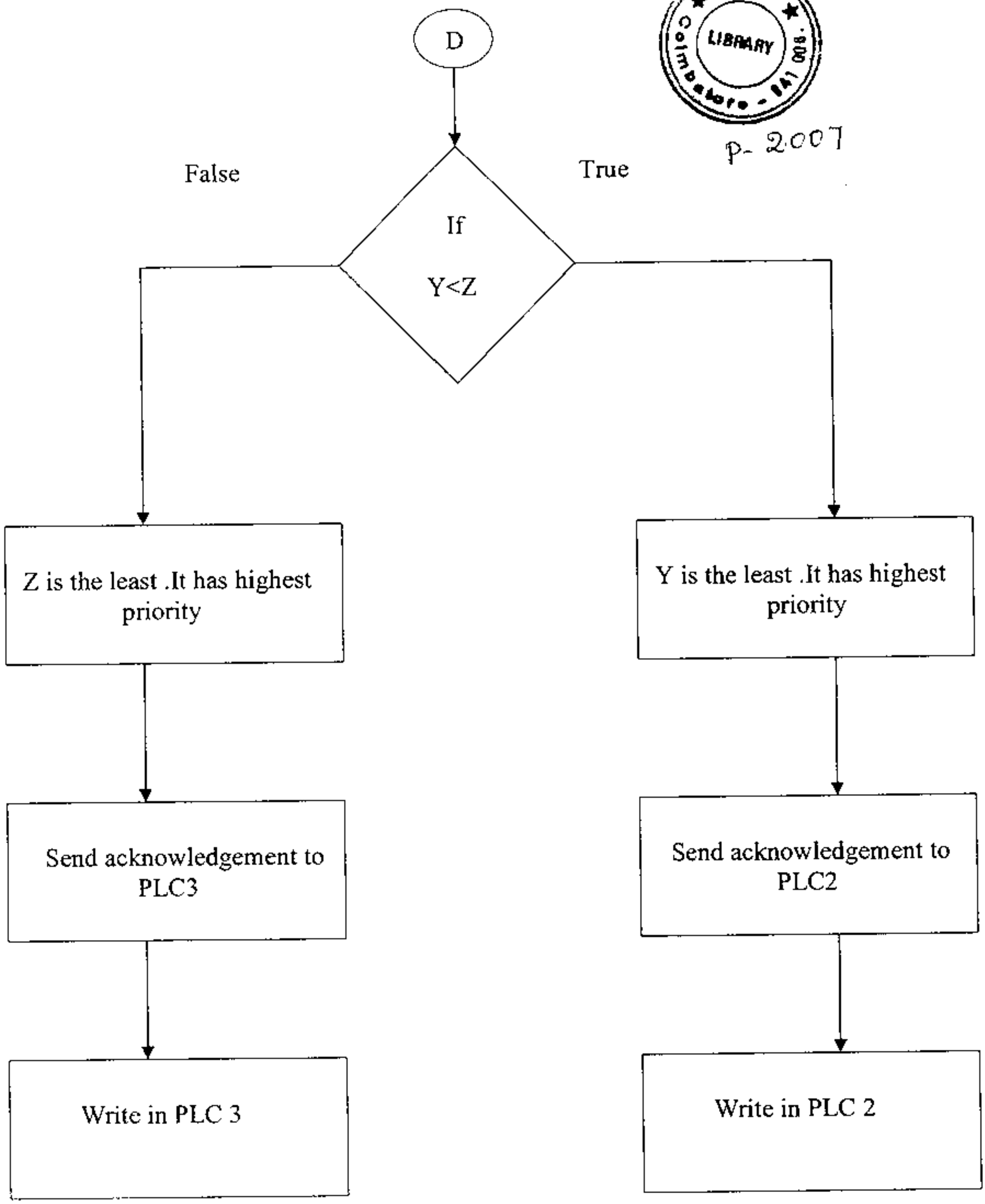
CHAPTER 5
FLOW CHART

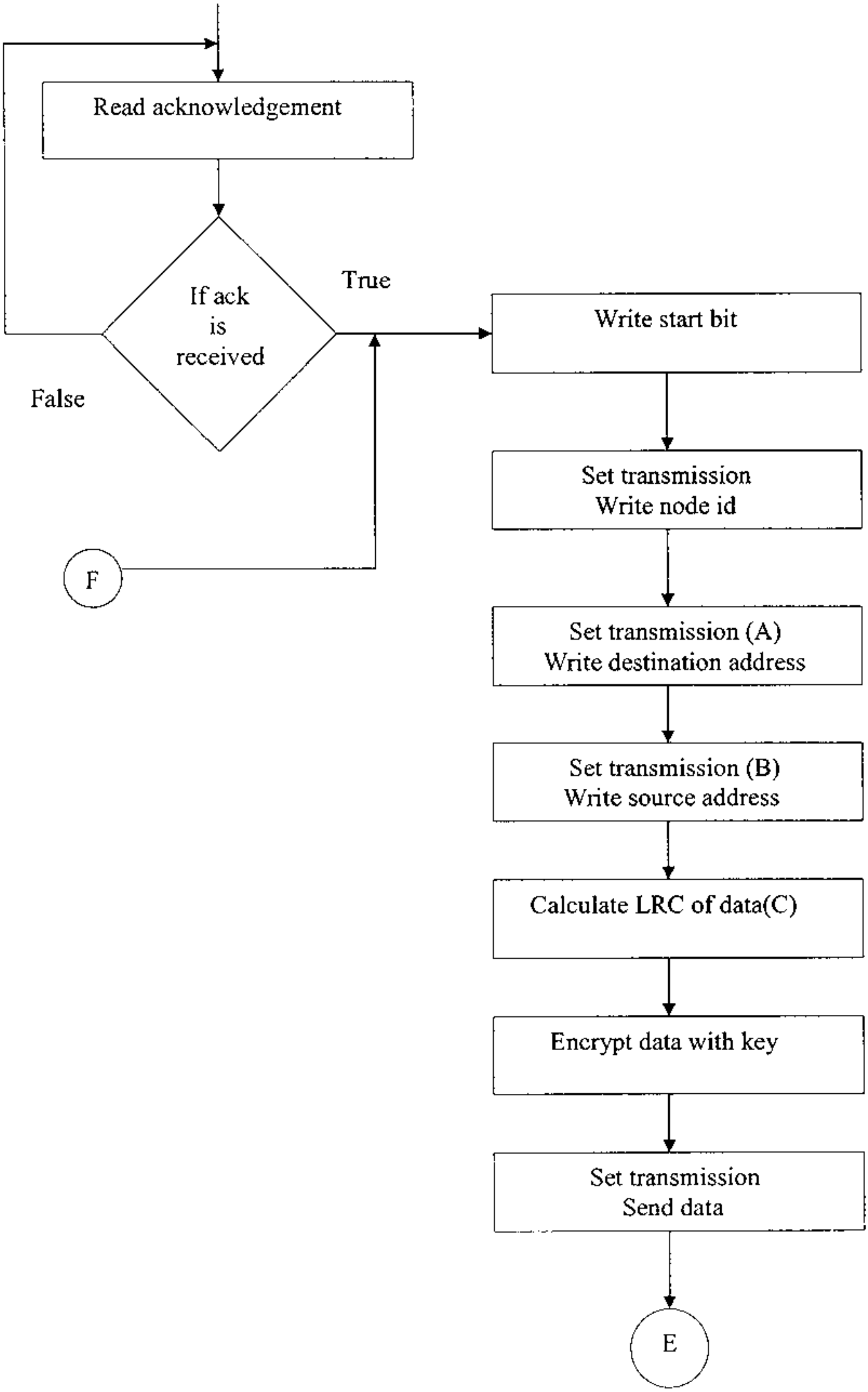


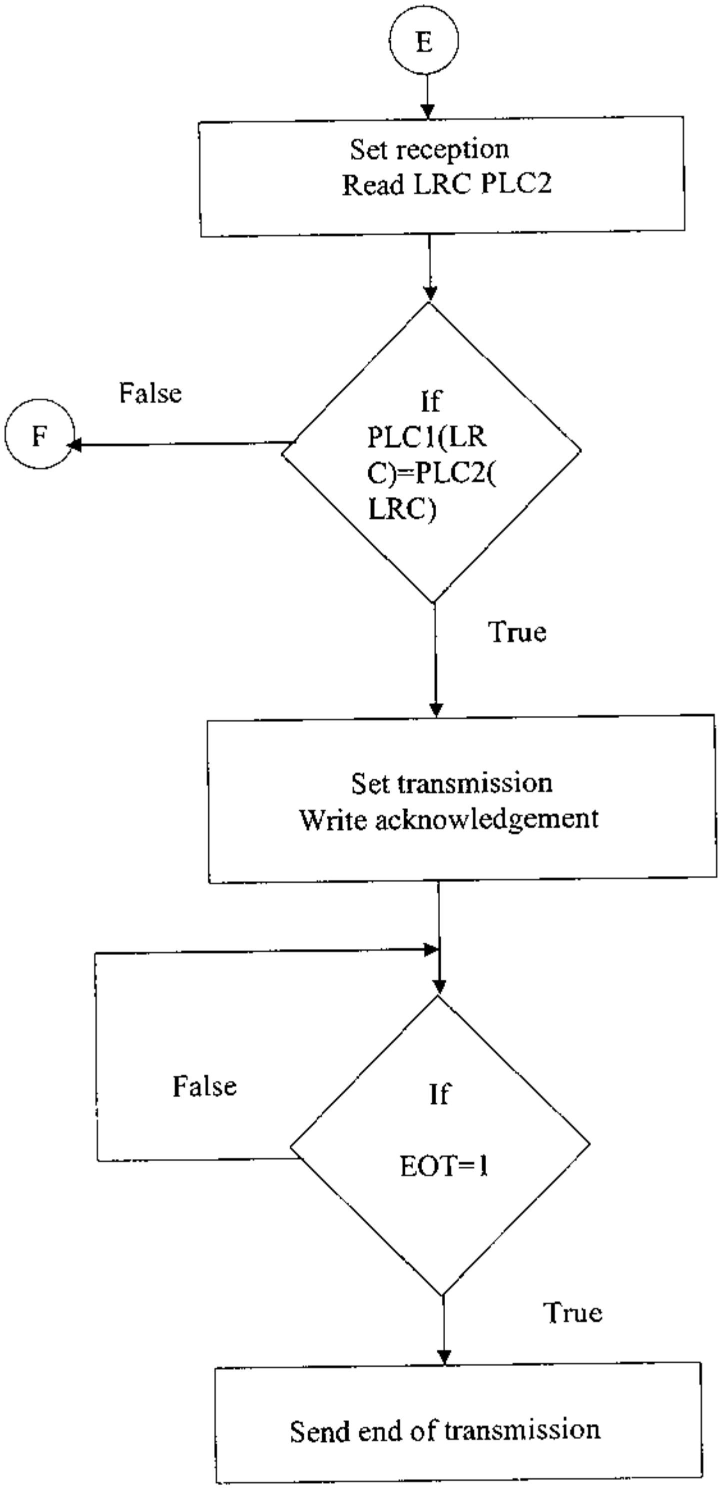

A
Transmission is "ON"
Write H2110 = 2100
False
If data = Change
True
True
Clock is activated
If Trans is "ON"
Clock not activated
False
True
If Trans is Initiated
B
False
Continue clocking

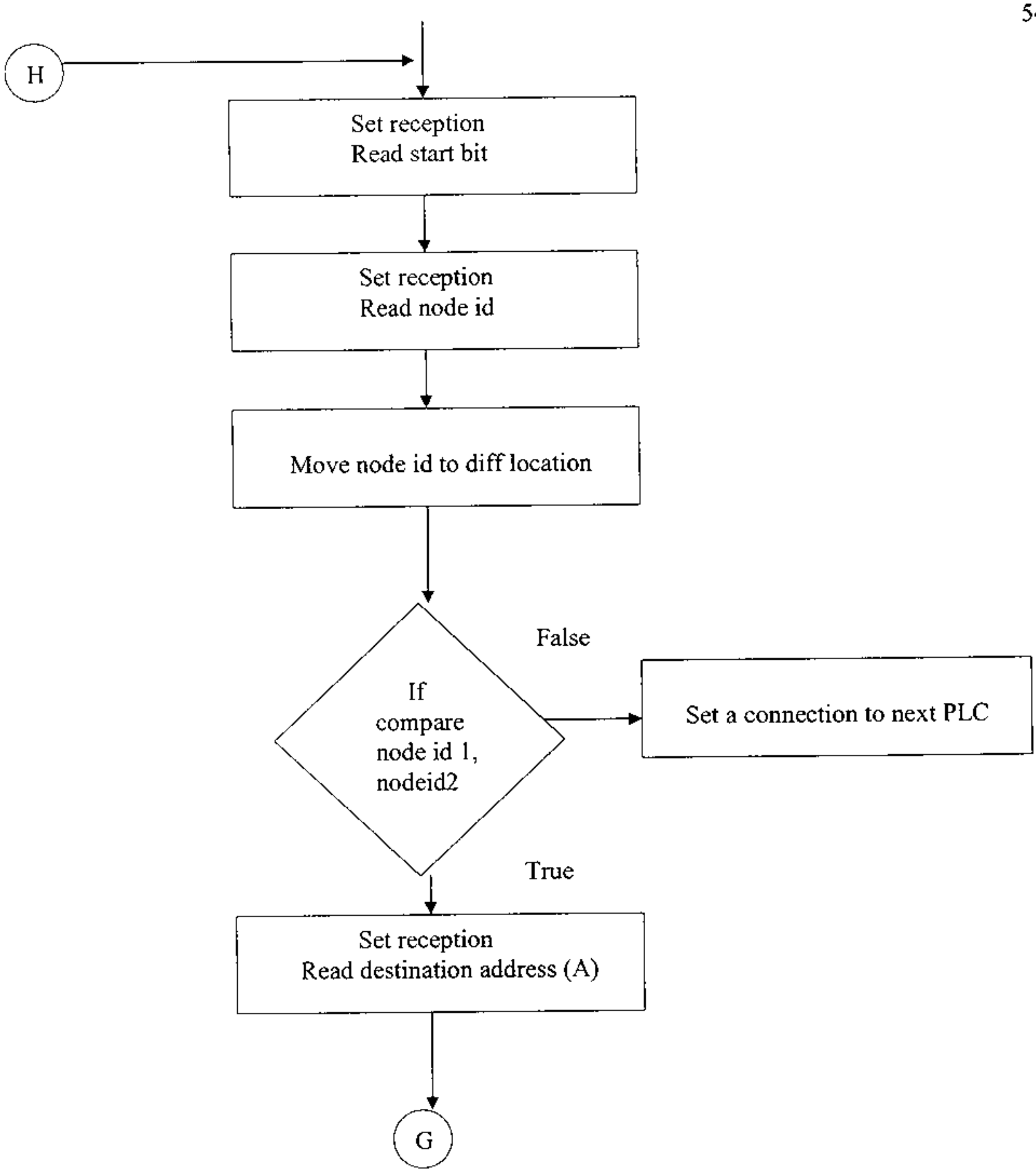

B
Stop clock
Write value in location "X"
Read value of Y from PLC 2
Write value in location of "PLC1"
Read value of Z from PLC3
Write value in location "PLC1"
Priority is fixed by LOGIC
False
If X < Y
True
D
C

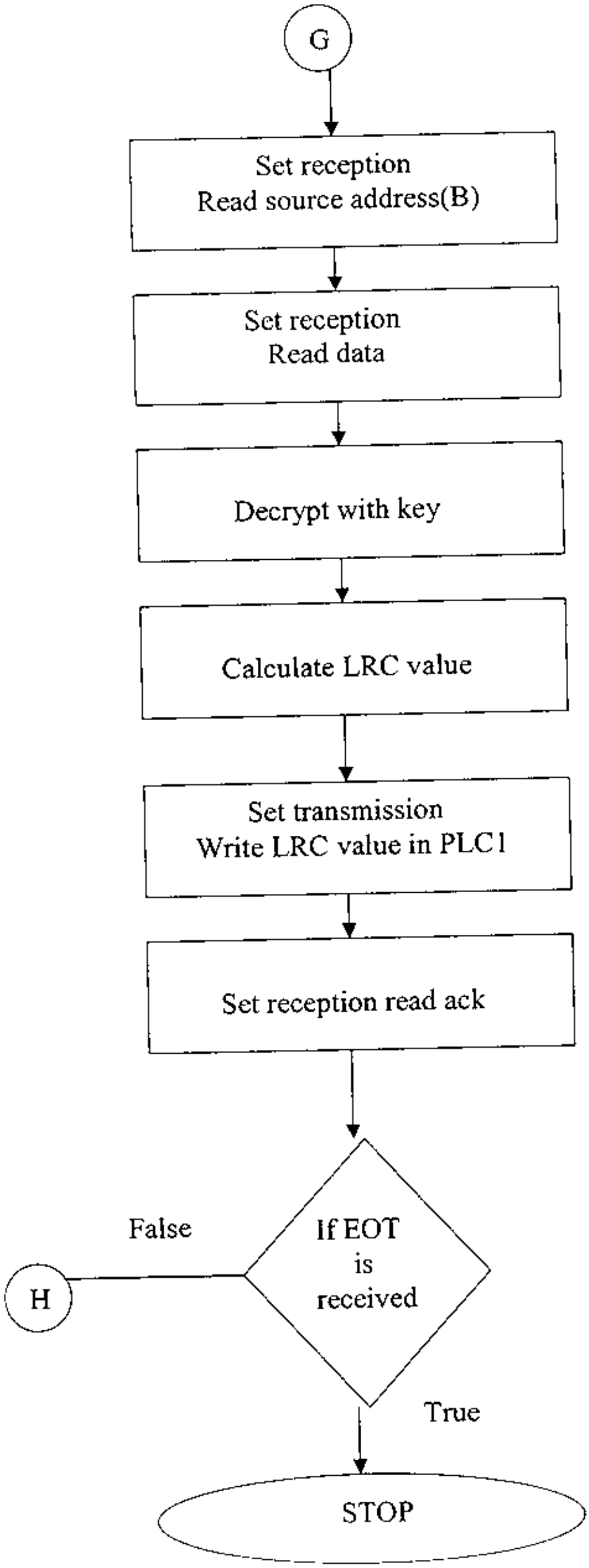

C
True
False
If
X<Z
X is the least .It has highest priority
Z is the least it has the highest priority
Send acknowledgement to PLC1
Send acknowledgement to PLC 2
Write in PLC 1
Write in PLC 3


D
False
True
If
Y<Z
Z is the least .It has highest priority
Y is the least .It has highest priority
Send acknowledgement to PLC3
Send acknowledgement to PLC2
Write in PLC 3
Write in PLC 2


Read acknowledgement
If ack is received
True
False
Write start bit
Set transmission
Write node id
Set transmission (A)
Write destination address
Set transmission (B)
Write source address
Calculate LRC of data(C)
Encrypt data with key
Set transmission
Send data
F
E


E
Set reception
Read LRC PLC2
False
F
If
PLC1(LRC)=PLC2(LRC)
True
Set transmission
Write acknowledgement
False
If
EOT=1
True
Send end of transmission


H
Set reception
Read start bit
Set reception
Read node id
Move node id to diff location
False
If
compare
node id 1,
nodeid2
Set a connection to next PLC
True
Set reception
Read destination address (A)
G


G
Set reception
Read source address(B)
Set reception
Read data
Decrypt with key
Calculate LRC value
Set transmission
Write LRC value in PLC1
Set reception read ack
If EOT
is
received
False
H
True
STOP

CODING

```
    ttt: BOOL;
    dd: BOOL;
    ee: WORD;
    sss: WORD;
    AA: WORD;
    XXX: WORD;
    AS: BOOL;
    SA: BOOL;
END_VAR
```

```
                    SS
                    IJK
   ttt  TRANSMIT              IJK_ERROR1        AS
    dd  START                 IJK_ERROR2    SA
  9600  ijk_baud
     1  ijk_PARITY
     1  ijk_STOP
     7  ijk_DATA
    ee  ijk_node_id
   sss  ijk_data_id
    AA  ijk_DATA_LEN
   XXX  ijk_COMM_RETRY
```

```
    NNN: NODE_RECEIVE;
    D2: BOOL;
    d3: BOOL;
    ddd: DATA_RECEIVE;
    xx: BOOL;
    t4: TON;
    rega: WORD;
    t5: TON;
    d4: BOOL;
    tran: BOOL;
    TRANSMITTERS: TRANSMITTER;
    TRANS: TRANS_SETTING;
END_VAR
VAR_INPUT
    TRANSMIT: BOOL;
    START: BOOL;
    ijk_baud: WORD;
    ijk_PARITY: WORD;
    ijk_STOP: WORD;
    ijk_DATA: WORD;
    ijk_node_id: WORD;
    ijk_data_id: WORD;
    ijk_DATA_LEN: WORD;
    ijk_COMM_RETRY: WORD;
END_VAR
VAR_OUTPUT
    IJK_ERROR1: BOOL;
    IJK_ERROR2: BOOL;
END_VAR
```

0001 HOME

```
                              TRANS
                          TRANS_SETTING                                        AND
     ijk_PARITY  PARITY            TRANSMISSION_ENABLE                                          TRA
       IJK_BAUD  BAUDRATE                                     ENABLE   START
       ijk_STOP  STOPBIT
   ijk_DATA_LEN  DATA_LEN
```

0002

```
   TRANSMIT1       START_RECEIVE
```

0003

```
   TRANSMIT1      TRANSMISSION
```

0004

```
            t4
            TON
   d1    IN        Q
   t#10s PT        ET
```

06

TIME_TO_WORD MOVE
t4.ET rega

07 nn

D1 NEXT1

08

D2 NEXT2

09

d3 next3

10

MOVE EQ
SBUF S D1
126

11

D1 HOME

12 NEXT1
NNN
NODE_RECEIVE
D2 RESET NODE_VERIFIED S D2

13

D2 HOME

14 NEXT2
ddd
DATA_RECEIVE
d3 reset data_rec S d3

15

d3 home

R d2

17

t5

MUL WORD_TO_TIME TON

rega d4 IN Q

100 PT ET

18

t5.Q home

R d3

19

xx home

020 TRANSMISSION:

TRANSMITTERS

TRANSMITTER

TRANSMIT TRANSMIT_ENABLE communication_error1 IJK_ERROR1

ijk_node_id NODEID communication_error2 IJK_ERROR2

ijk_data_id DATAID

ijk_data DATA

ijk_COMM_RETRY RETRY_COUNT

F:\ijk.pro

```
    D3: BOOL;
    D4: BOOL;
    SCON_REG: WORD;
END_VAR
VAR_INPUT
    PARITY: WORD;
    BAUDRATE: WORD;
    STOPBIT: WORD;
    DATA_LEN: WORD;
END_VAR
VAR_OUTPUT
    TRANSMISSION_ENABLE: BOOL;
END_VAR
```

```
001 HOME:

                EQ
  BAUDRATE                D1
        9600

002

             EQ
  PARITY               D2
       1

003

              EQ
  STOPBIT               D3
         1

004

                EQ
  DATA_LEN               D4
           7

005

        AND
  D1                                       >HOME
  D2              TRANSMISSION_ENABLE
  D3
  D4

006

      MOVE
  70               SCON_REG
```

```
    DUMMY1: BOOL;
    DUMMY2: BOOL;
    TP2: TP;
    DUMMY3: BOOL;
    NODE_RECEIVE: BOOL;
    LRC_RECEIVE: BOOL;
    DATA_RECEIVE: BOOL;
    DUMMY4: BOOL;
    TP3: TP;
    DUMMY5: BOOL;
    TP4: TP;
    SBUF: WORD;
    DUMMY6: BOOL;
    C1: CTU;
    node: NODE_TRANS;
    DATAS: DATA_TRANS;
    TP5: TP;
    DUMMY7: BOOL;
    DUMMY8: BOOL;
    D1: BOOL;
    D2: BOOL;
END_VAR
VAR_INPUT
    TRANSMIT_ENABLE: BOOL;
    NODEID: WORD;
    DATAID: WORD;
    DATA: WORD;
    RETRY_COUNT: WORD;
END_VAR
VAR_OUTPUT
    communication_error1: BOOL;
    communication_error2: BOOL;
END_VAR
```

001 HOME:

```
                         AND
TRANSMIT_ENABLE                     START_TRANSMIT
    NODE_RECEIVE
      LRC_RECEIVE
```

002

```
                         OR
communication_error1                >HOME
communication_error2
```

003

```
                         TP1
                         TP
START_TRANSMIT   IN              Q
         t#1MS   PT              ET
```

DUMMY2 NEXT1

006

MOVE
126 SBUF

007

TP1.Q S DUMMY1

008

AND
TP1.Q S DUMMY2
DUMMY1

009

DUMMY2 HOME

010 NEXT1

node
NODE_TRANS
NODEID NODE_ID RECEIVE_ENABLE NODE_RECEIVE
RETRY_COUNT RETRY_COUNT ERROR communication_error1
D1 RESET TRANSMITTED S DUMMY3
DUMMY2 TRANSMIT_ENABLE

011

DUMMY3 D1

012

DUMMY3 HOME

D2 RESET TRANSMITTED S DUMMY4

DUMMY3 TRANSMIT_ENABLE

014

DUMMY4 D2

015

DUMMY4 HOME

016

TP5
TP

START_TRANSMIT IN Q
t#1MS PT ET

017

MOVE
126 SBUF

018

TP5.Q S DUMMY7

019

AND
TP5.Q S DUMMY8
DUMMY7

020

DUMMY8 R DUMMY1
R DUMMY2
R DUMMY3
R DUMMY4
R DUMMY5
R DUMMY8
R DUMMY7
R DUMMY6

```
    D1: BOOL;
    D3: BOOL;
    D4: BOOL;
    LRC: WORD;
    D5: BOOL;
    DD: BOOL;
    D6: BOOL;
    C1: CTU;
END_VAR
VAR_INPUT
    DATA_ID: WORD;
    DATA: WORD;
    RESET: BOOL;
    TRANSMIT_ENABLE: BOOL;
END_VAR
VAR_OUTPUT
    RECEIVE_ENABLE: BOOL;
    ERROR: BOOL;
    TRANSMITTED: BOOL;
END_VAR
```

001 HOME

```
RESET    R D1
         R D2
         R D3
         R D4
         R D5
         S D6
```

002

```
D2     >NEXT
```

003

```
D4     >NEXT1
```

004

```
                      TP1
                      TP
TRANSMIT_ENABLE  IN          Q      >HOME
          T#1ms  PT          ET
```

005

```
          MOVE
DATA_ID              SBUF
```

007

AND
D1 S D2
TP1.Q

008 NEXT

TP2
TP
D2 IN Q
T#1MS PT ET

009

MOVE
DATA SBUF

010

TP2.Q S D3

011

AND
D3 S D4
TP2.Q

012 NEXT1

D4 HOME

013

XOR
DATA_ID LRC
DATA

014

TP2
TP
D4 IN Q RECEIVE_ENABLE
T#1MS PT ET

015

RECEIVE_ENABLE START_RECEIVE

17

RECEIVE_ENABLE S D5

18

AND

DD R D1

D5 R D2

RECEIVE_ENABLE R D3

R D4

R D5

S D6

019

DD TRANSMITTED

020

C1

OR CTU

DD RECEIVE_ENABLE CU Q ERROR

RESET RESET CV

3 PV

F:\ijk.pro

```
    d2: BOOL;
    T2: TP;
    REG2: WORD;
    d3: BOOL;
    d4: BOOL;
    t3: TP;
    d5: BOOL;
    d6: BOOL;
END_VAR
VAR_INPUT
    reset: BOOL;
END_VAR
VAR_OUTPUT
    data_rec: BOOL;
END_VAR
```

0001 home

```
reset   R d2
        R d3
        R d4
        R d5
        R d6
        R d1
```

0002

```
d2      next1
```

0003

```
d4      next2
```

0004

```
                       T1
                       TP
START_RECEIVE   IN          Q
       T#1ms    PT          ET
```

0005

```
        MOVE
SBUF               REG1
```

0006

```
t1.Q    S d1
```

008

d2 home

009 next1

T2
TP

START_RECEIVE IN Q
T#1ms PT ET

010

MOVE
SBUF REG2

011

t2.Q S d3

012

AND
d3 S d4
t2.Q

013

d4 home

014 next2

XOR
reg1 sbuf
reg2

015

t3
TP

d4 IN Q
t#1ms PT ET

016

t3.Q S d5

d6 data_rec

F:\ijk.pro

```
    TP2: TP;
    C1: CTU;
    DD: BOOL;
    RETRANSMIT: BOOL;
    D3: BOOL;
END_VAR
VAR_INPUT
    NODE_ID: WORD;
    RETRY_COUNT: WORD;
    RESET: BOOL;
    TRANSMIT_ENABLE: BOOL;
END_VAR
VAR_OUTPUT
    RECEIVE_ENABLE: BOOL;
    ERROR: BOOL;
    TRANSMITTED: BOOL;
END_VAR
```

001 HOME

RESET R D1
R D2
R DD

002

D2 NEXT

003

OR
TRANSMIT_ENABLE
RETRANSMIT

TP1
TP
IN Q HOME
T#1ms PT ET

004

MOVE
NODE_ID SBUF

005

TP1.Q S D1

006

AND
D1 S D2
TP1.Q

08

TP2
TP
D2 IN Q RECEIVE_ENABLE
T#1MS PT ET

09

RECEIVE_ENABLE START_RECEIVE

010

EQ
SBUF S DD
NODE_ID

011

RECEIVE_ENABLE S D3

012

AND
DD R D1
D3 R D2
TP2.Q R D3

013

DD TRANSMITTED

014

C1
CTU
OR
DD RECEIVE_ENABLE CU Q ERROR
RESET RESET CV
RETRY_COUNT PV

```
VAR_INPUT
    RESET: BOOL;
END_VAR
VAR_OUTPUT
    NODE_VERIFIED: BOOL;
END_VAR
```

001 HOME

```
RESET    R D1
```

002

```
D1      >NEXT
```

003

```
          EQ
SBUF               S D1
    1
```

004

```
D1      START_TRANSMIT
```

005

```
D1    R START_RECEIVE
```

006

```
D1      >HOME
```

007 NEXT

```
               TT
               TON
      D1  IN          Q
  T#1MS   PT          ET
```

008

```
      MOVE
1               SBUF
```

REFERENCES

REFERENCES:

1. Behrouz A. Forouzan, 'Data Communications and Networking', Tata McGraw Hill edition 2003
2. Frank D. Petruzella, 'Programmable Logic Controllers', Glencoe McGraw-Hill second edition
3. http://www.omron.com /
4. http://www.profibus.com /

CONCLUSION

CHAPTER 6
CONCLUSION

5.1 CONCLUSION

Networks are changing the way we do business and the way we live. Business decisions have to be made ever more quickly. And the decision maker requires immediate access to accurate information.

At the end of the project we conclude that the working is one of the most efficient tools to minimize the real time working errors. After understanding the problem faced by the company we proposed two solutions, one with use of microcontroller and the other using communication protocols. Out of the above two solutions the method using protocols was found more efficient. It does mot include any additional component so the maintenance of the system is very less. The chances of error occurrence is also very less.

We have learnt valuable technical knowledge like networking, microcontroller programming and various communication protocols.

Finally we were successful to network the old Omron PLC's with the combined concepts of controllers and protocols. This also helped the company to reduce huge loss. This project will definitely bring end to robust human working.

Our sincere efforts toward the completion of this project has given us the valuable experience to use our all fundamental knowledge and has become the stepping stone for our professional life

.

5.2 FUTURE SCOPE

- New protocols can be developed which can make the communication easier.
- New buses can be used to increase the number of node of connection
- New inbuilt communication facility can be developed in the PLC.
- Wireless communication can decrease the wirings.

APPENDIX

1-1 Features

The CQM1H is a compact Programmable Controller (PC) that supports communications and other advanced functions. It is a package-type PC that is mounted to DIN Track to control small to medium-size machines.

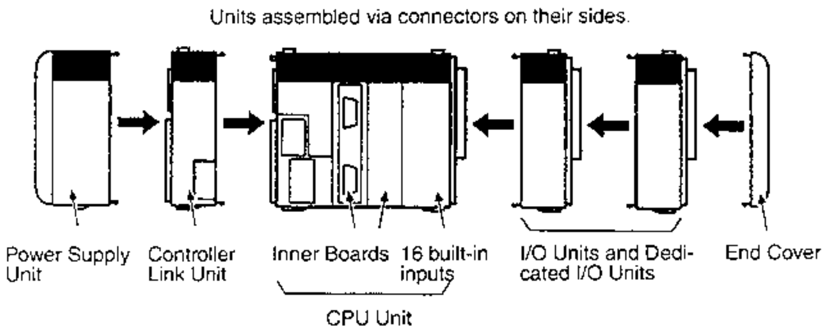
A flexible system configuration is enhanced by serial communications with a protocol macro function, user-installed boards called Inner Boards, network communications, a wide range of monitoring and setting methods, higher speed, and larger capacity. These features enable added-value machine control.

- Mount up to two Inner Boards to add communications or control functions. Communications Functions: Serial Communications Board Control Functions: High-speed Counter Board, Pulse I/O Board, Absolute Encoder Interface Board, Analog Setting Board, and Analog I/O Board
- Mount a Controller Link Unit to connect to a Controller Link Network.
- Connect simultaneously to both a Programming Device and a Programmable Terminal (PT).
- Obtain higher speed and capacity in comparison to the CQM1: 1.25 times faster, twice the program capacity (15.2 Kwords), twice the I/O capacity (512 points), and twice the data memory capacity (12 Kwords).
- Use new instructions.
- Maintain compatibility with previous models of PC.

Flexible System Configuration

The CQM1H does not require a Backplane and is constructed by connecting Units via connectors on the sides of the Units, allowing flexible system configuration. The CPU Unit contains 16 built-in DC input points. Two Inner Boards can be mounted in the CPU Unit. One Controller Link Unit (a Communications Unit) and a combined maximum of eleven I/O Units and Dedicated I/O Units can also be connected. If an Expansion I/O Block is used, a maximum of 16 Units can be connected. (See *1-2-1 Basic Configuration.*)

Note
1. The CQM1H is mounted to DIN Track.
2. Only the CQM1H-CPU51/61 CPU Units support Inner Boards and the Controller Link Unit.



Higher Speeds and Greater Capacity

Execution times have been reduced to 0.375 µs for the LOAD instruction (from 0.50 µs for the CQM1), to 17.7 µs for the MOVE instruction (from 23.5 µs) and to 0.70 ms for overseeing (from 0.80 ms), reducing the total cycle time by approximately 25%.

- The program capacity, the I/O capacity, and the data memory capacity have all been approximately doubled. The program capacity has been increased to 15.2 Kwords (from 7.2 Kwords for the CQM1); the I/O capacity, to 512 points (from 256 points); and the data memory capacity, to 6 Kwords of DM and 6 Kwords of EM (from 6 Kwords of DM only).

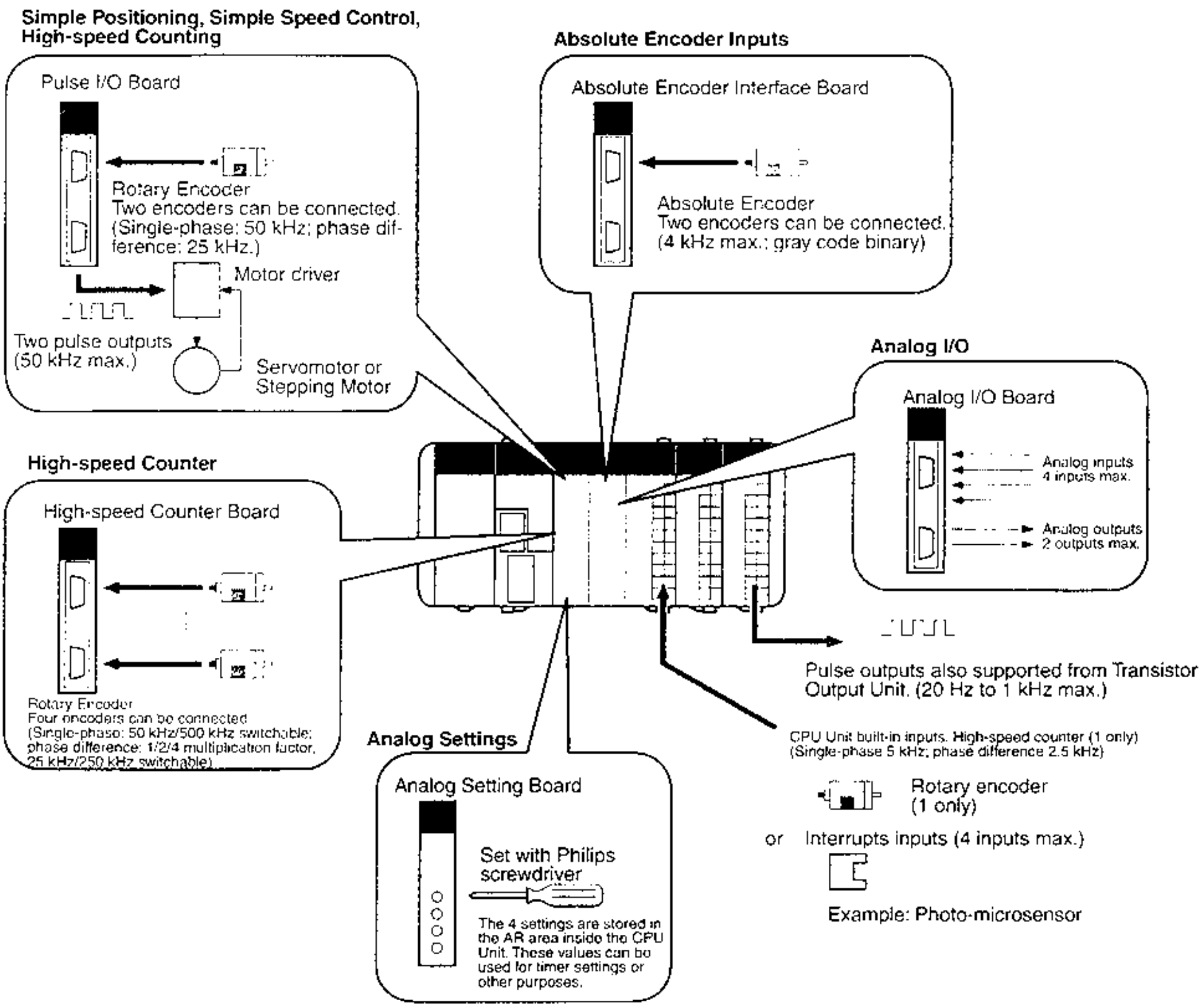
- A 16-Kword Memory Cassette can be mounted in the CQM1H to handle large user programs or more data. These features ensure a higher level of machine control and greater ease of use.

Increased Functionality with Inner Boards

The CQM1H features Inner Boards that allow serial communications, multipoint high-speed counter (rotary encoder) inputs, simple positioning (trapezoidal acceleration/deceleration pulse outputs), speed changes, PWM (variable duty-factor pulse) outputs, absolute rotary encoder inputs, analog I/O (4 inputs, 2 outputs), and analog settings.

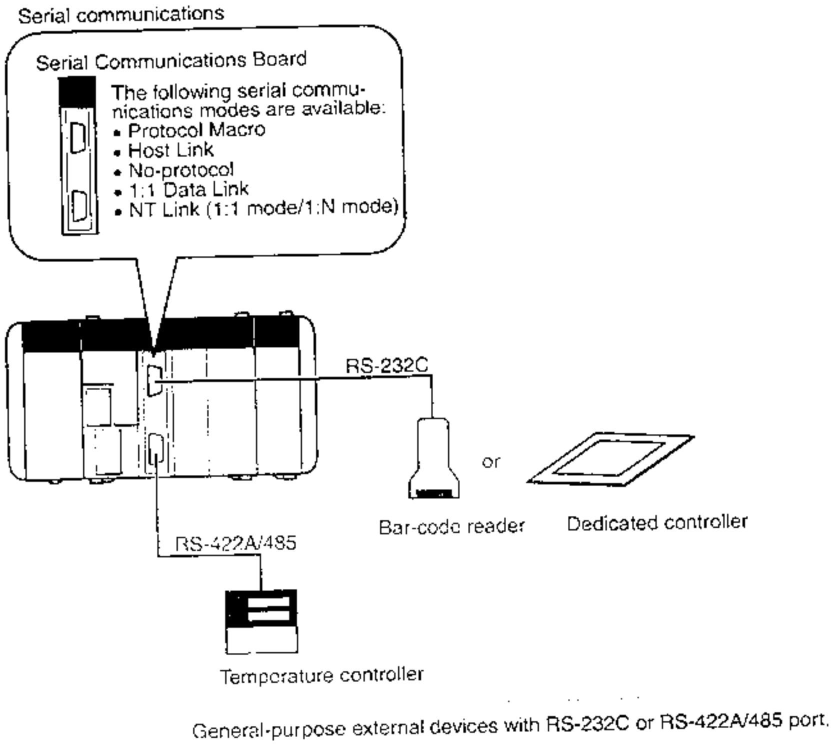
A Serial Communications Board, High-speed Counter Board, Pulse I/O Board, Absolute Encoder Interface Board, Analog I/O Board, and Analog Setting Board are available. These Inner Boards can be combined, mounted and used as required for the machine being controlled. (There are mounting restrictions for some of the Inner Boards.)

Note The CPU Unit also provides16 built-in inputs, as well as high-speed counter and input interrupt functions. Pulse outputs are also supported using a standard Transistor Output Unit.



etter Connections to achine Components with erial Communications

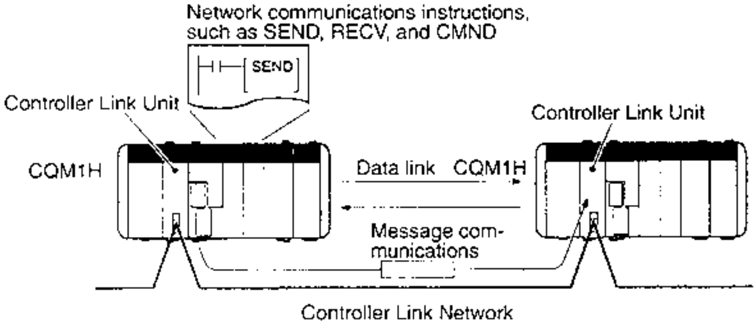
Connections can be easily made to general-purpose machine components and dedicated controllers. The Serial Communications Board (an Inner Board) supports a protocol macro function. You can create macros for protocols according to the communications specifications of the external device, allowing data transfers with general-purpose devices to be executed with a single PMCR instruction. Essentially any device with a serial port can be communicated with, such as temperature controllers, bar-code readers, and dedicated numeric controllers.



Distributed Control with Compact PCs with Network Communications

A Controller Link Unit can be included in the CQM1H. Data can be exchanged between several PCs using a Controller Link Network. Data links are supported to create shared data areas and message communications are supported to enable sending required data and commands using network communications instructions. The Controller Link Network can be easily constructed using twisted-pair cable. Data exchange is also supported with C200HX/HG/HE, CS1, CVM1, and CV-series PCs, as well as with personal computers.

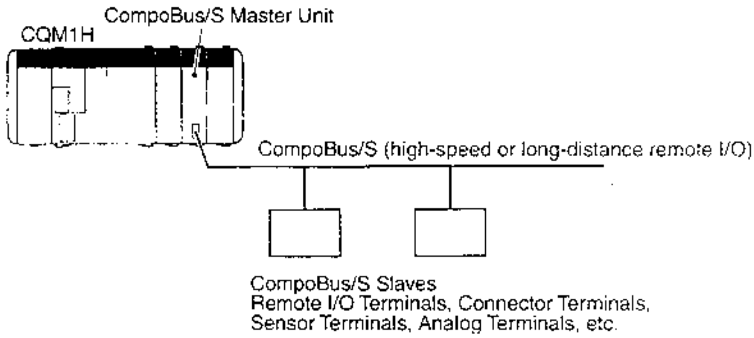
Note Data links can be created with another CQM1H or with a CQM1, CPM1, CPM1A, CPM2A, CPM2C, SRM1, C200HX/HG/HE, or C200HS simply by making a 1:1 connection between the built-in RS-232C ports in the CPU Units.



Baud rate: 2 Mbps; transmission distance: 1 km (when baud rate is 500 kbps); max. No. of nodes: 32.
A maximum of 8,000 words per node can be sent for the CQM1H.
Data exchange supported for CQM1, CQM1H, CS1, C200HX/HG/HE, and CVM1/CV-series PCs.

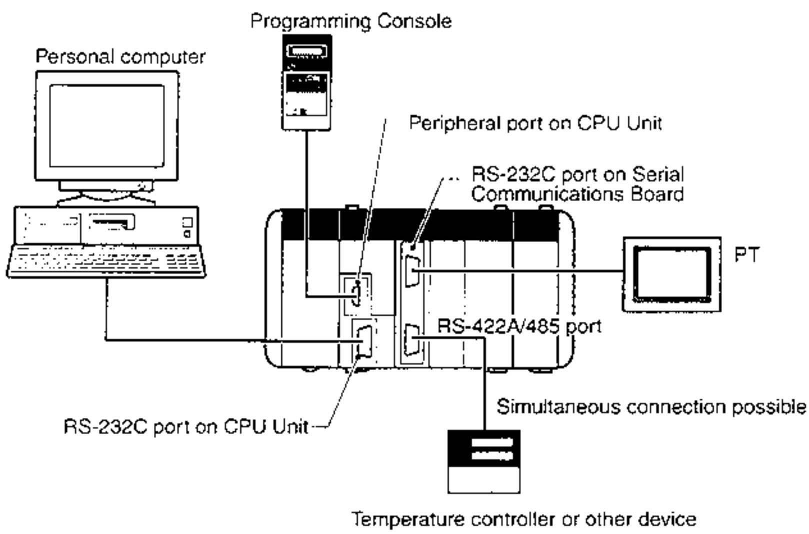
High-speed/ Long-distance Communications with CompoBus/S

A CompoBus/S Master Unit can be included in the CQM1H. High-speed or long-distance remote I/O communications can be performed with CompoBus/S Slaves. (The CompoBus/S Master Unit is a Dedicated I/O Unit for the CQM1H.)



A Wide Range of HMI Monitoring and Setting Methods

Programming Devices and Programmable Terminals (PTs) can be connected to up to four ports, two ports on the CPU Unit and two ports on a Serial Communications Board. It is thus possible to set up and monitor machine control from a PT while monitoring or programming from a Programming Console or a personal computer.



You can also program and monitor from a personal computer at a remote location via a modem. When used in combination with the protocol macro function, it is also possible to call the personal computer from the CQM1H using the PMCR(—) instruction, and when the connection is made, switch the serial communications mode to Host Link (for remote programming/monitoring) using the STUP(—) instruction.

If an Analog Setting Board is mounted, fine adjustments of settings, such as rotational speed or timer settings, are possible on-site using the adjustments on the front of the Board.

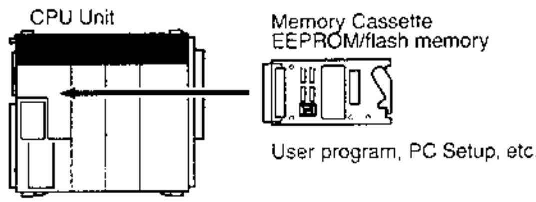
The ON/OFF status of a user-programmable DIP switch pin is stored in the AR area. The setting of this pin can be used on-site to switch between trial operation and actual operation, to switch set values, or to perform any other function that can be programmed in response to the changes in status of the AR bit corresponding to this DIP switch pin.

Easier Programming with a Complete Instruction Set and Interrupt Functions

Math instructions (such as floating-point math, exponential functions, logarithmic functions, and trigonometric functions), a TOTALIZING TIMER (TTIM(—)) instruction, a CHANGE RS-232C SETUP (STUP(—)) instruction, and network communications instructions have been added. In addition, complete interrupt functions for the CPU Unit are supported, including input interrupts, high-speed counter interrupts, and interval timer interrupts (with scheduled interrupts and one-shot interrupts). Interrupts from serial communications using a protocol macro (interrupt notification) are also supported. These interrupts enable easier and more flexible machine control.

Memory Cassettes for Program/Data Management; Clock Included

A Memory Cassette (EEPROM or flash memory) can be mounted in the front of the CPU Unit. User programs, data memory (read-only DM, PC Setup) and expansion instruction information can be saved and read in batch. It is also possible to make settings so that data contained in the Memory Cassette is loaded automatically at startup. This feature means that, in the event of battery expiration or careless programming/monitoring operations, data for user programs and data memory is not lost. It also means that changes in user programs required for different controlled machines can be made easily. Further, by using a Memory Cassette with a clock, times and dates can be used in the user program.



Compatibility with CQM1 Units

The Power Supply Units, Basic I/O Units, and Dedicated I/O Units for the CQM1 can be used in the CQM1H. Consequently, Dedicated I/O Units like Temperature Control Units, Sensor Units, B7A Interface Units, and CompoBus/D (DeviceNet) Link Units can all be used. In addition, user programs used on the CQM1, Programming Consoles for the CQM1 and conventional Memory Cassettes can also be used. (A conversion adapter is necessary to use the Programming Console.)

1-2 System Configuration

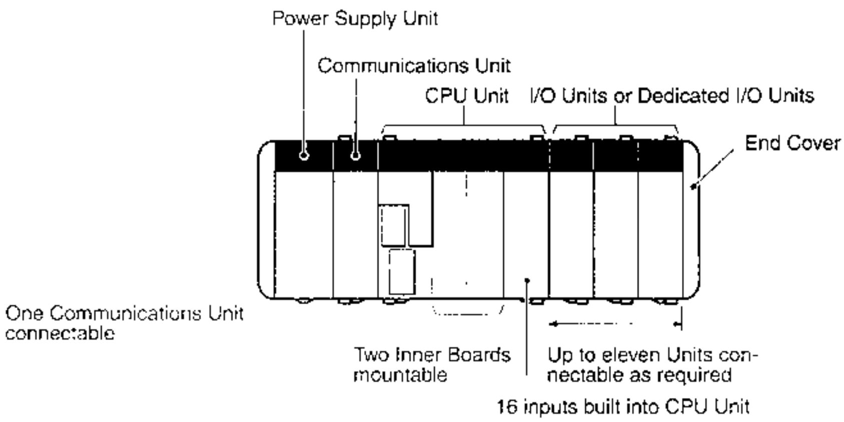
1-2-1 Basic Configuration

The PC configuration depends on the model of CPU Unit being used and on whether or not an Expansion I/O Block is connected. Examples are shown below.

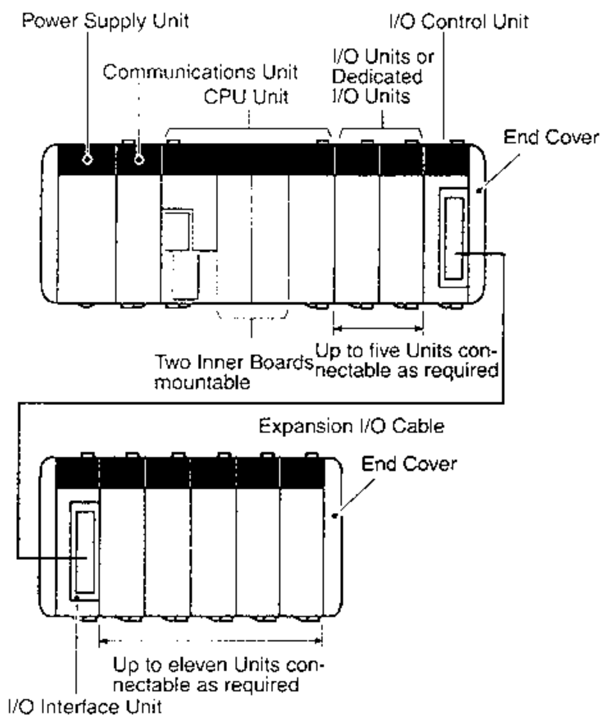
CQM1H-CPU51/61

Up to two Inner Boards can be mounted and one Communications Unit can be connected with the CQM1H-CPU51 or CQM1H-CPU61 CPU Unit. The configuration is shown below.

CPU Block Only



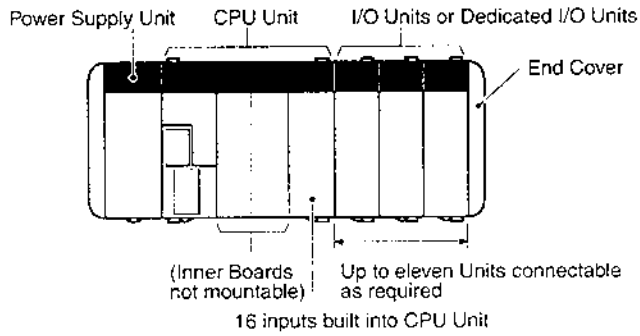
CPU Block and Expansion I/O Block



CQM1H-CPU11/21

The CQM1H-CPU11 and CQM1H-CPU21 CPU Units do not support Inner Boards or Communications Units. The configuration is shown below.

CPU Block Only



CPU Block and Expansion I/O Block