1990

Control of a mini-Cartrac material handling system in an FMS environment

Hung-Ren Lyou
Lehigh University

CONTROL OF A MINI-CARTRAC MATERIAL HANDLING SYSTEM IN AN FMS ENVIRONMENT

by

Hung-Ren Lyou

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

the Department of Industrial Engineering

Lehigh University

1990

Certificate of Approval

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

May 10, 1990
Date

Gregory L. Tonkay
Professor in Charge

M U Thomas
Chairman of the Department

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

ABSTRACT

The development of an automated material handling (AMH) system plays an important role in the operation of flexible manufacturing systems. The growing use of advanced computers and control equipment, such as personal computers (PCs); computer numerical control (CNC); and programmable logic controllers (PLCs), is enabling many companies to optimize the benefits of AMH systems. The Cartrac system, part of AMH equipment, is a conveyor system consisting of several carts that travel over tracks. It has been installed in the Department of Industrial Engineering at Lehigh University and is currently controlled by several PLCs.

This paper will describe the following topics:

1. The required devices in an AMH system.
2. The conversion of the manually operated Cartrac system into a fully automated system. In performing this task, the structure and control logic of the original system were completely changed. By rewriting the original, very large ladder logic programs in the PLCs, the newly developed control system provides many smaller, independent ladder logic programs acting as low level commands.
3. The programming technique of the high level commands. By programming the high level commands in a high level language, a personal computer acting as a cell controller will be able to control the logic sequence of the low level commands in the PLCs.

Chapter 1 Introduction

In 1986, SI Handling Incorporated donated a mini-Cartrac conveyor system to the Department of Industrial Engineering at Lehigh University. This system has been installed in the Manufacturing Technology (ManTech) Laboratory and is the material handling system for a flexible manufacturing cell. Figure 1.1 shows the current layout of the cell.

1.1 System Description

1.1.1 Cartrac System

The Cartrac system is a material handling system consisting of several carts that travel over tracks. The operation of the system in the Manufacturing Technology Laboratory is controlled by four Programmable Logic Controllers (PLCs) installed in this system. Various input devices, such as mechanical switches and photo sensors, and output devices, such as motors and solenoids that activate pneumatic cylinders, are used to control the movement of the carts on the programmed route. The system has 14 distinct stations where the carts can be located. The Cartrac system has been interfaced with an Automated Storage and Retrieval System (AS/RS) and a Kearney & Trecker (K&T) milling center using mechanical transfer devices.

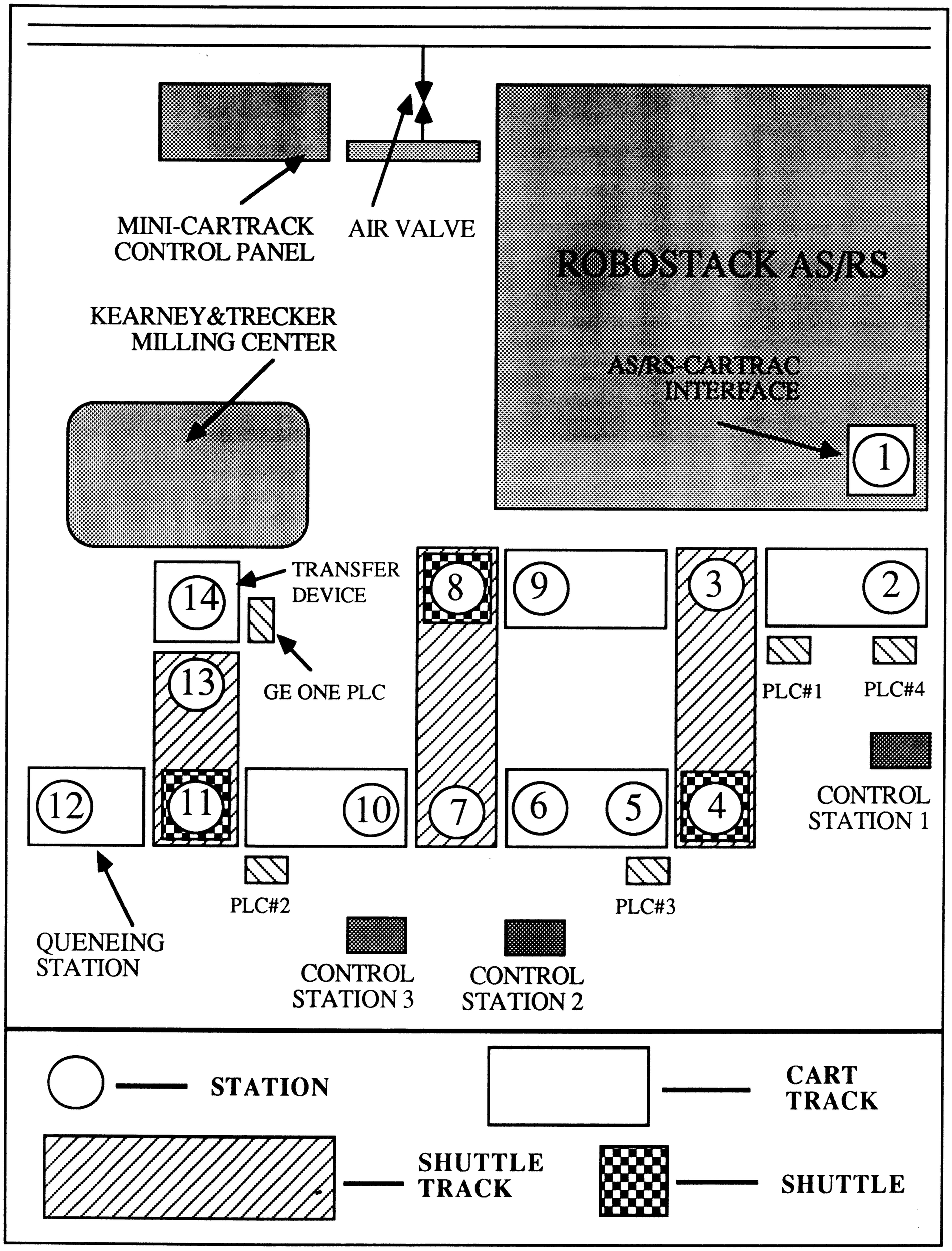


Figure 1.1 Layout of Cartrac System

1.1.2 Programmable Logic Controller

Programmable logic controllers (PLCs) are used in virtually every segment of industry where automation is required. They represent one of the fastest-growing segments of the industrial electronics industry. Since their inception, PLCs have proven to be the salvation of many manufacturing systems which previously relied on electromechanical control. As supplied by SI Handling, four Allen Bradley PLC-4s connected on a loop are used to control the system. An additional GE Series One Plus PLC is used to control the Cartrac-to-K&T transfer device.

1.1.3 Automated Storage and Retrieval System

An automated storage and retrieval system (AS/RS) is defined as a combination of equipment and controls which handles, stores, and retrieves materials with precision, accuracy, and speed under a defined degree of automation (Groover,1987).

All AS/RS systems consist of certain basic building blocks. These components are as follows :

1. Storage structure.
2. Storage/retrieval (S/R) machine.
3. Storage modules.
4. Pickup-and-deposit stations.

Storage structure is the fabricated steel framework that supports the loads contained in the AS/RS. It consists of one or more storage aisles that have storage racks for holding the materials to be stored.

S/R machine (also called a crane) is used to accomplish a storage transaction, delivering loads from storage and delivering them to the output station.

Storage modules are the containers of the stored material, such as pallets, steel wire baskets and containers, tote pans, storage bins, and special drawers.

Pick-and-deposit stations are used to transfer loads to and from the AS/RS. The P&D station can be manually operated or interfaced to some form of automated handling system, such as conveyor systems (e.g., roller, cartrac, chain) and AGVS.

The Robostack Automated Storage and Retrieval System is a single-aisle, twin row AS/RS installed in the ManTech Laboratory at Lehigh University. Figure 1.2 shows a schematic drawing of the system.

The Robostack AS/RS consists of two rows 12 levels high and 12 columns (bays) long. The positions (bay, level, row) are labeled as illustrated in Figure 1.2. This AS/RS can be controlled manually or semiautomatically from the crane control panel or automatically from the IBM 7552 industrial microcomputer. The pick-and-deposit station (also called AS/RS-Cartrac interface) is composed of a turntable, a load arm, and an unload arm. The sequence of loading and unloading totes at this station is controlled by a PLC.

1.1.4 Kearney & Trecker Milling Center

This milling center is a very large, complex NC machining center which has been modified with a GE Mark Century 2000 computer numerical control unit. A transferring mechanism (Figure 1.1) has been installed between the Cartrac system and the K&T milling center to interface these two systems. The logic sequence of the transferring

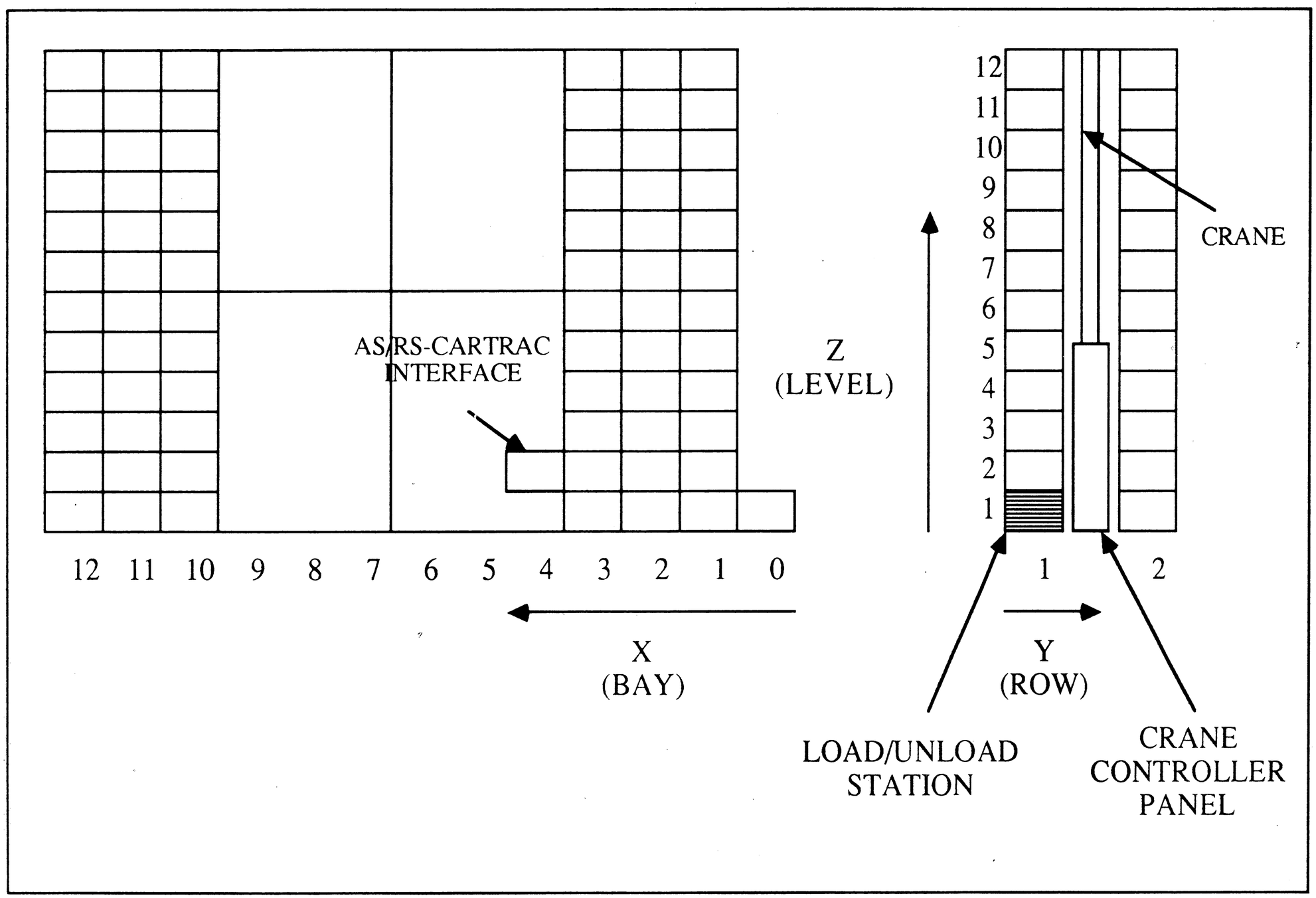


Figure 1.2 Layout of Robostack AS/RS

machine is controlled by a GE Series One Plus PLC that is not on the Allen Bradley loop of PLC's.

1.1.5 Cart, Shuttle and Queueing station

The Cartrac system uses individual carts riding on a two-railed track contained in a frame that places the track a few feet above floor level (Figure 1.3). The carts are not individually powered. Instead, they are driven by a rotating tube that runs between the two rails. A drive wheel, attached to the bottom of the cart and set at an angle to the center line of the rotating tube, rests against it and drives the cart forward. The cart speed is controlled by regulating the angle of contact between the drive wheel and the rotating tube. When the drive wheel is perpendicular to the tube, the cart does not move. As the angle is increased toward 45 degrees, the speed increases to a maximum value.

Because of the rail and rotating tube design, the carts are unable to turn around corners on the track. Hence, a shuttle is used as an auxiliary transportation device, to transport a cart between two sections of drive tubes such that it can carry a cart from the head of one station to the end of another station (Figure 1.1). Three shuttles are located in the Cartrac system (Figure 1.1), each of which is controlled by a different PLC.

The Queueing station, which is close to the K&T milling center, offers a space for a cart to wait while the K&T milling center is occupied. By using the Queueing station, the goal is to maximize the utilization of the machine tool and reduce the cycle time of the product.

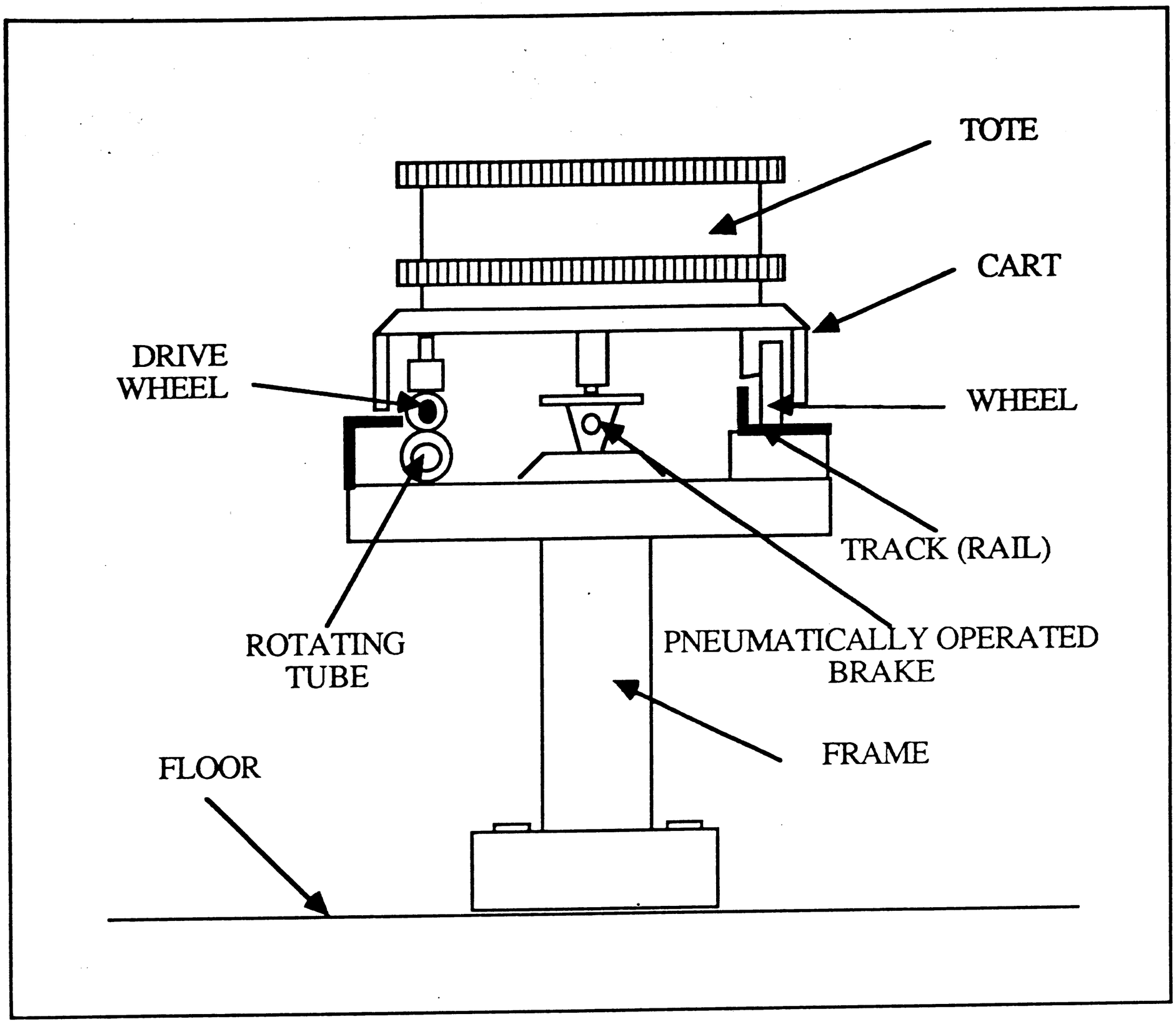


Figure 1.3 Diagram of cart on the track

1.1.6 Typical Flow

As mentioned above, the Cartrac system is one kind of automated material handling system. The functions of an automated material handling system are to provide work-in-process (WIP) interfacing, movement between work stations, fixture and tool changing support, a provision for automated buffer storage, the advancement of components or products through the manufacturing processes and eventually, to storage and dispatch.

As referred to in Figure 1.1, currently, the Cartrac is used to move the carts

containing the work pieces from the AS/RS to the K&T milling center or from the K&T milling center to the AS/RS. The flows of the carts are controlled by the PLCs and the manual control stations. The PLCs determine the logic sequence and the control stations determine the direction of the cart. Each control station contains a manual toggle switch to change the route of the cart and two push button switches, one to release the cart from a station and the other to provide for emergency stop. Because manual switches are required, the Cartrac system is not a completely automated material handling currently.

1.2 Present Control Logic

1.2.1 Description

As stated previously, the Cartrac system is controlled by several PLCs. This system uses three Allen Bradley PLC/4's to control the logic sequence of the carts plus an additional PLC/4 to control the AS/RS-Cartrac interface. Currently, as referred to in Figure 1.1, PLC/4 #1 controls the region of stations 2,3,4,5 and 6; PLC/4 #2 controls stations 11,12,13 and 14; PLC/4 #3 controls stations 7,8,9 and 10; and PLC/4 #4 is responsible for the interface between station 1 (AS/RS-Cartrac interface) and the AS/RS.

In the present implementation, four individual ladder logic programs have been entered in the four PLCs. Because there must be communication between these programs, the four PLCs are wired together to create a communication system, also called a loop. This loop passes information about each PLC's inputs, outputs, and flag bits. In this application, using a loop of PLCs has the following significant benefits:

1. Increases the memory of the system up to 2560 words.
2. Increases the I/O capabilities of the system up to 80 inputs and 48 outputs.
3. Connects the loop via the communication interface module to the Allen Bradley Data Highway.
4. Connects the loop directly to a computer via the communication interface module.

Because more than one PLC is in the loop, flag bits are used in the programs. Flag bits function like store bits, that is, they serve as a "scratch pad" to store information about the status of various inputs and outputs. Through the use of flag bits, this information is stored for use by other PLCs. For example, the program in PLC/4 #1 can use various inputs directly from other PLCs or instruct other PLCs to activate outputs by setting flag bits, provided the other PLC is programmed to activate the output when the flag bit is set.

1.2.2 Limitations

As mentioned, the Cartrac system is not a completely automated material handling system because manual control stations are required. The other limitations of the current implementation of the Cartrac system are:

1. The direction of a cart must be determined manually.
2. It is difficult to change parts of the ladder diagram program without understanding the entire system.
3. It does not perform many of the functions the operators would like.
4. It is difficult to understand and debug the large ladder logic programs.

5. It cannot be controlled with a high level computer. To control it with a computer, the program would require significant modification.
6. If the carts are not configured in a specific way upon startup, system faults will occur.
7. The logic provision of the Queueing station has not been implemented and therefore the buffer is not available.

1.3 Proposed Changes

Because there are many limitations in the current Cartrac system, a new logic system will be developed that uses hierarchical control concepts by dividing the tasks into high level and low level commands. Low level commands deal directly with sequencing the physical hardware. High level commands sequence the low level commands and thus require little knowledge of the physical hardware. By implementing low level commands properly, the programmer can modify the small, independent parts of the system more easily without affecting other parts of the system. Once implemented, the low level commands will only be changed when hardware is modified.

The logic and sequence of the high level commands can be easily modified by linking the low level commands in different sequences. The low level commands are concerned with physical hardware, such as limit switches, lights and motors, etc. The sequence of each low level command is determined by the fulfillment of a set of logic conditions, such as inputs, outputs, or timers, etc. Further, the functions of the high level commands will determine the route of the carts by sequencing the low level commands. Thus, the low level commands transfer the cart between two stations without considering

the sequence of the cart on a system level. The high level commands consider theinteraction of carts and processes in the system.

1.3.1 Low Level Commands

The reason for using low level commands is to separate the original, very large ladder logic programs into many smaller, independent ladder logic programs. Each independent program controls the motion of a cart from one station to an adjacent station. All of the original ladder logic programs in the PLCs except PLC/4 #4 have been replaced with programs implementing the independent, low level commands. Since the program in PLC/4 #4 is a single command which controls the movement of the totes between the AS/RS and the Cartrac, it has not been modified in the new control system.

In the Cartrac system, 21 low level commands are currently defined to move carts between all the stations. So far, using this technique, the system has achieved the following benefits:

1. All of the motors except those at the stations 5 and 9 can stop automatically when they are not in use.
2. A cart can be placed on any station before the system starts up without a system fault.
3. The Queueing station (station 12) logic has been implemented.
4. Through the using of flag bits, the programs in one PLC can indirectly activate outputs from other PLCs.

5. Compared to the original programs, the revised programs are much easier to understand and debug.

1.3.2 High Level Commands

The purpose of the high level commands is to allow the use of a high level language, such as C, to control the sequence of the low level commands in the PLCs. The high level commands will eliminate the need for the manual control stations to route the pallets. A PC (personal computer) will play the central role, instead of the PLCs, to determine the direction of motion of the carts. In the proposed control strategy, the following high level commands have been defined:

1. Move from AS/RS to K&T Milling Center.
2. Move from AS/RS to Queueing Station.
3. Move from Queueing station to K&T Milling Center.
4. Move from Queueing station to AS/RS.
5. Move from K&T Milling Center to AS/RS.
6. Move from K&T Milling Center to Queueing Station.
7. Loop carts in the system and wait for additional commands.

Each high level command consists of a set of low level commands. The sequence of the low level commands is determined by the logic and sequence of physical moves required to perform each of the high level commands.

1.3.3 Interface to Cell Controller

Upon integration of the high level and the low level commands, the Cartrac system can be automatically controlled by the computer without human intervention. Nevertheless, the reality of communications between the Cartrac system and other automated devices, such as the AS/RS and K&T milling center, has not been fully achieved. In order to integrate these three systems (Cartrac, AS/RS and K&T milling center), a higher level computer will be added as a *cell controller*. Figure 1.4 shows the proposed architecture of the flexible cell. This flexible cell is structured as a two-layered hierarchy of devices. The cell controller, represented as the higher level computer, has the capacity of real-time multi-tasking to sequence and control the Cartrac, AS/RS and K&T milling center. The AS/RS is currently under the control of the cell controller. This research will provide the framework to allow the control of Cartrac to be added to the cell controller.

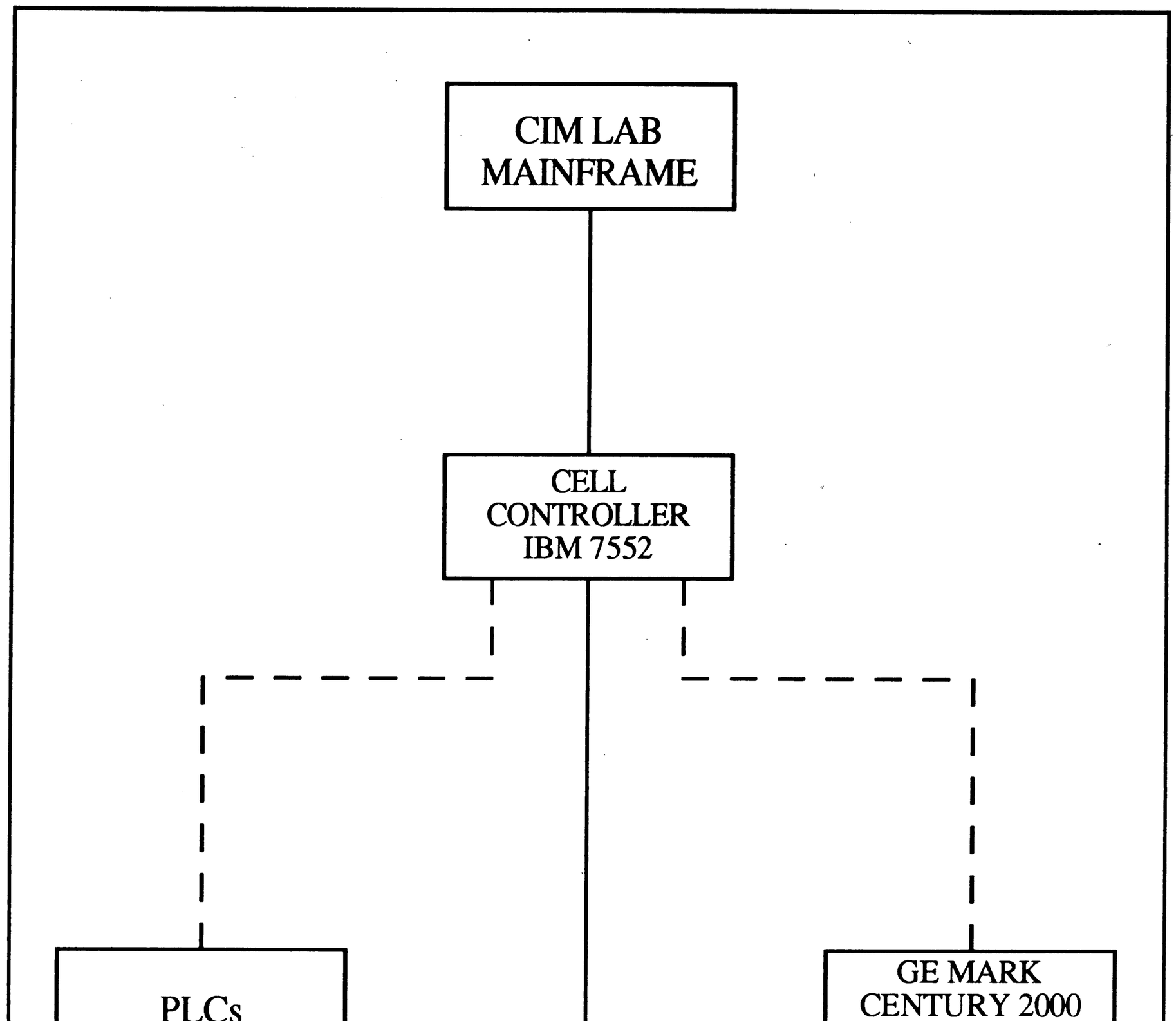


Figure 1.4 The flexible cell architecture

Chapter 2 Survey of the Literature

2.1 Flexible Manufacturing Systems

2.1.1 Survey of FMS

A *flexible manufacturing system* (FMS) consists of a group of processing stations, interconnected by means of an automated material handling and storage system, and controlled by an integrated computer system (Groover, 1987). The machining process is presently the largest application area for FMS technology. There are three basic components of a flexible manufacturing system:

1. ***Processing station***. In present-day applications, these workstations are typically computer numerical control (CNC) machine tools that perform machining operations on families of parts. However, flexible manufacturing systems are being designed with other types of processing equipment, including inspection stations, assembly workheads, and sheet metal presses.

2. ***Material handling and storage***. Various types of automated material handling equipment are used to transport the workparts and subassemblies between the processing stations, sometimes incorporating storage into the function.

3. ***Computer control system***. Computer control is used to coordinate the activities of the processing stations and the material handling system in the FMS.

One additional component in the FMS is human labor. Human beings are needed to manage the operations of the flexible manufacturing system. Functions performed by people include loading raw workparts into the system, unloading finished parts from the

system, changing and setting tools, equipment maintenance and repair, NC part programming, and programming and operating the computer system.

2.1.2 Material Handling and Storage System

Material handling systems (MHS) have an important role in the operation of flexible manufacturing systems. Traditionally, the material handling systems have focused on inventory and transport control, but by connecting with other computers and control equipment, such as PCs, CNCs and PLCs, the systems have become automated material handling (AMH) systems.

The material handling and storage system in a flexible manufacturing system should perform the following functions (Groover, 1987):

1. ***Random, independent movement of workparts between workstations***. This means that parts must be capable of moving from any one machine in the system to any other machine.

2. ***Handle a variety of workpart configurations***. For prismatic parts, this is usually accomplished by using pallet fixtures in the handling system. The fixture is located on the top face of the pallet and is designed to accommodate different part configurations by means of common components, quick-change features, and other devices that permit a rapid buildup of the fixture for a given part.

3. ***Temporary storage***. The number of parts in the FMS typically exceeds the number of parts actually being processed. In this way, each machine can have a queue of parts waiting to be processed. This helps to increase machine utilization.

4. ***Convenient access for loading and unloading workparts***. The handling system must provide a means to load and unload parts from the FMS. This is often

accomplished by having one or more load/unload stations in the system.

5. ***Compatible with computer control.*** The handling system must be capable of being controlled directly by the computer to direct it to the various workstations, load and unload stations, etc.

The types of material handling equipment that have been used to transfer parts between stations in a FMS include: roller conveyors; cart-on-track conveyors; other types of conveyor systems, such as in-floor towline carts; automated guided vehicle systems; and industrial robots. They constitute what is sometimes called the *primary material handling system* in the FMS. The primary material handling system is sometimes supported by an automated storage system such as an automated storage/retrieval system (AS/RS). The types of material handling equipment typically utilized for the basic five FMS layouts are summarized in Table 2.1.

2.1.3 Case Study I

As mentioned above, a system which lends itself to PLC control and is readily linked with mainframe computers to meet accurate logging and monitoring requirements is currently installed at BAe Warton Ltd,. A tool carousel, servicing ten machining centers, is part of BAe Warton's large FMS complex for machining prismatic components. This tool carousel represents a typical precision interfacing requirement met by a powered platen conveyor system (Serzeniewski, 1987).

The machining centers are placed on a central platen Cartrac conveyor track which

Table 2.1 Material handling systems used for the five FMS layouts (Groover, 1987).

Layout configuration	Typical material handling system
1. In-line	Conveyor system, shuttle system
2. Loop	Conveyor system
3. Ladder	Conveyor system, AGVS
4. Open field	AGVS, in-floor tow-line carts
5. Robot-centered cell	Industrial robot(s)

is built on a mezzanine deck, where a cart conveys the cutting tool pallets between the machine carrousels and a pallet buffer area. A robot is used to insert and retrieve cutting tools from the machine carrousels and transfer them at the interface buffer area to AGVs which handle the supply of the tools. The tool changing system is controlled by a local microcomputer with software and hardware supplied in a turn-key package which is interfaced with the overall FMS management system.

The cart or platen is fundamental to such a system since it forms the mobile jig used to convey components or assemblies down the production line. The Cartrac conveyor provides an ideal basis for precision interfacing since it offers the following benefits:

1. Few limitations on the size and weight of the component to be conveyed as platens can be purpose-built.
2. Controlled acceleration and deceleration during indexing.
3. High speed indexing (5.5 meters in 7 seconds).
4. Inter-station alignment not critical.

5. Production line length unlimited and layout varied.

6. High accuracy and repeatability.

7. Few moving parts reduces maintenance costs.

2.1.4 Case Study II

The Allis-Chalmer's FMS is a random process machining system made up of the following equipment (Castle, 1982):

1. Machine tools.
2. Control computers
3. Palletized workpiece fixtures.
4. Material handling system.
5. Machine shuttles.
6. Casting load/unload stations.

The system utilizes ten machine tools. Five of these are standard NC machining centers with automatic tool changes. One is a NC rough milling machining center. The remaining four are NC duplex head indexing machines.

The control of the system is handled by two minicomputers. One computer is called a DNC computer. Another computer, called the FMS computer, controls the entire system including the DNC computer, the material handling system, and all movements and processing of the palletized fixtures and parts. The FMS computer also handles communication with the casting load/unload stations.

The movement of the palletized fixtures within the system is conducted by the

material handling system under the control of the FMS computer. The material handling system consists of 28 carts, 12 under-floor tow-lines to power the carts, 64 stop blades to halt the carts and hold them in position, and 9 diverter switches to change the direction of cart travel. The linking of the machine tool worktables with the material handling system is achieved by machine shuttle units. The buffering station helps to isolate the machine tool worktable from the dynamics of the material handling system and assure a smooth flow of workpieces through the worktable.

2.1.5 Important Points of the Case Studies

As discussed in the two cases above, the Cartrac system at Lehigh will provide the following benefits:

The main advantage of the Cartrac system is that it allows asynchronous or independent motion of the carts. This is particularly useful when the cycle time of the operations on the line vary. The asynchronous motion also enables the work piece to be stationary while work is being performed on it.

The second advantage of the Cartrac system compared to many other conveyor systems is that the carts can achieve relatively high accuracies of position. This permits their use for positioning work during production.

The third advantage of the Cartrac system is its provision of built-in buffer stations. This helps to maximize the utilization of the system and reduce the cycle time of the product.

Another advantage of a Cartrac system is its clean operation. For this reason, Cartrac systems are being used in any environment, from automative spot welding to electronics assembly.

As mentioned above, the purpose of the mini-cartrac system is to provide an *automated material handling* (AMH) system for the flexible manufacturing cell in the ManTech Lab.

2.2 Control of Manufacturing System

2.2.1 Programmable Logic Controller

One of the most common low level controllers on the shop floor is the PLC. A typical PLC can be divided into three parts as illustrated in Figure 2.1. These three components are the central processing unit (CPU), the input/output (I/O) section, and the programming device (Johnson, 1987).

1. The ***CPU*** is the "brain" of the system. Internally, it contains various logic gate circuits. The CPU is a microprocessor-based system that replaces control relays, counters, timers, and sequencers. It accepts input data from various sensing devices, executes the stored user program from memory, and sends appropriate output commands to the control devices.

2. The ***I/O section*** consists of input modules and output modules. The purpose of I/O is to condition the various signals received from or sent to the external devices. Input devices such as push buttons, limit switches, photo sensors, selector switches, and thumbwheel switches are hard-wired to terminals on the input modules. Output devices such as small motors, motor starters, solenoid valves, and indicator lights are hard-wired to the terminals on the output modules.

3. The ***Programming device*** or terminal, is used to enter the desired program into the memory of the processor. The program can be entered using *ladder logic language*

which is one of the programming languages for PLCs (Figure 2.2). More details of this language will be explicated in section 2.3. The program determines the sequence of operation and ultimate control of the equipment or machinery. The programming device can be a hand-held unit with a light emitting diode (LED) display or a desktop unit with a cathode-ray tube (CRT) dispaly.

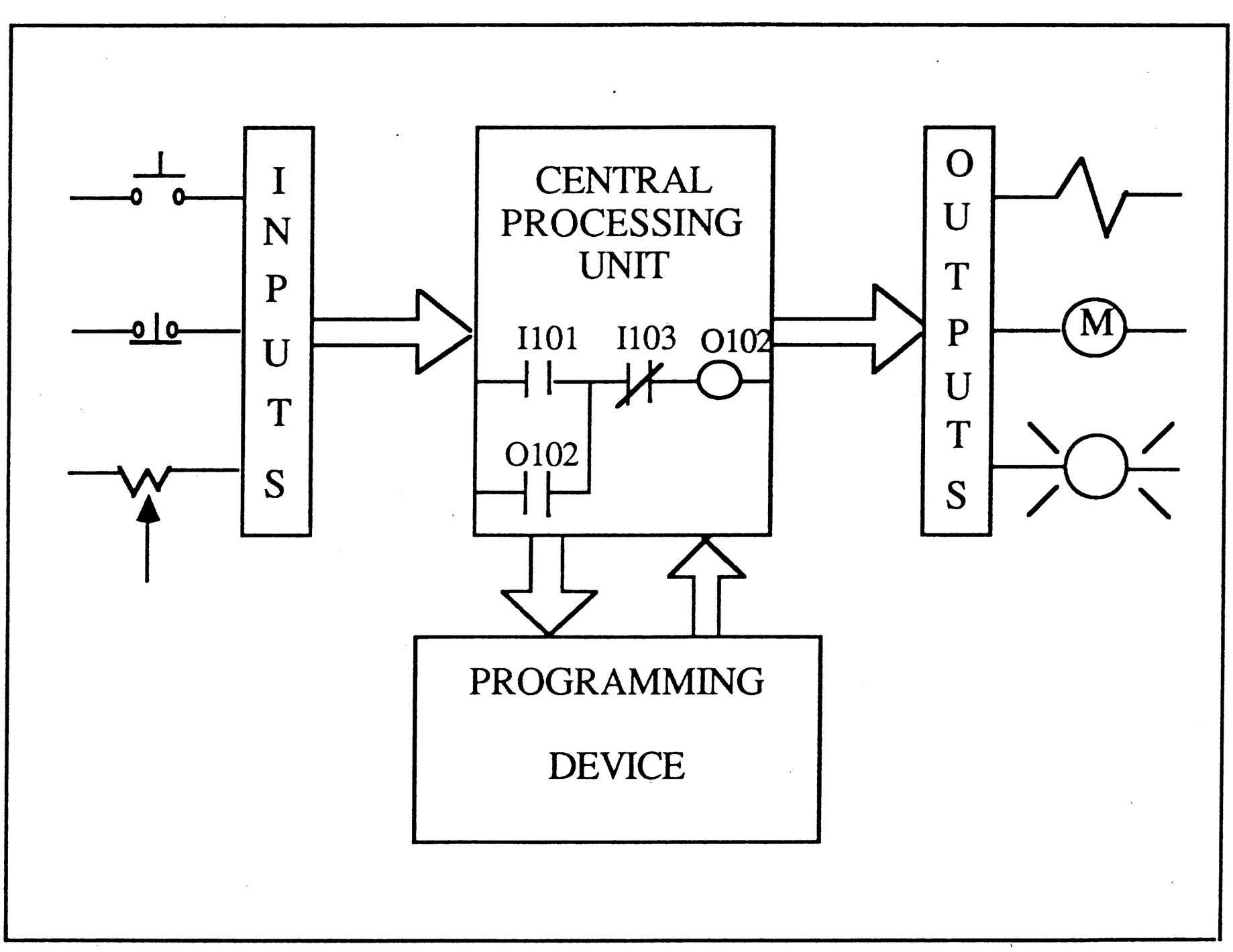
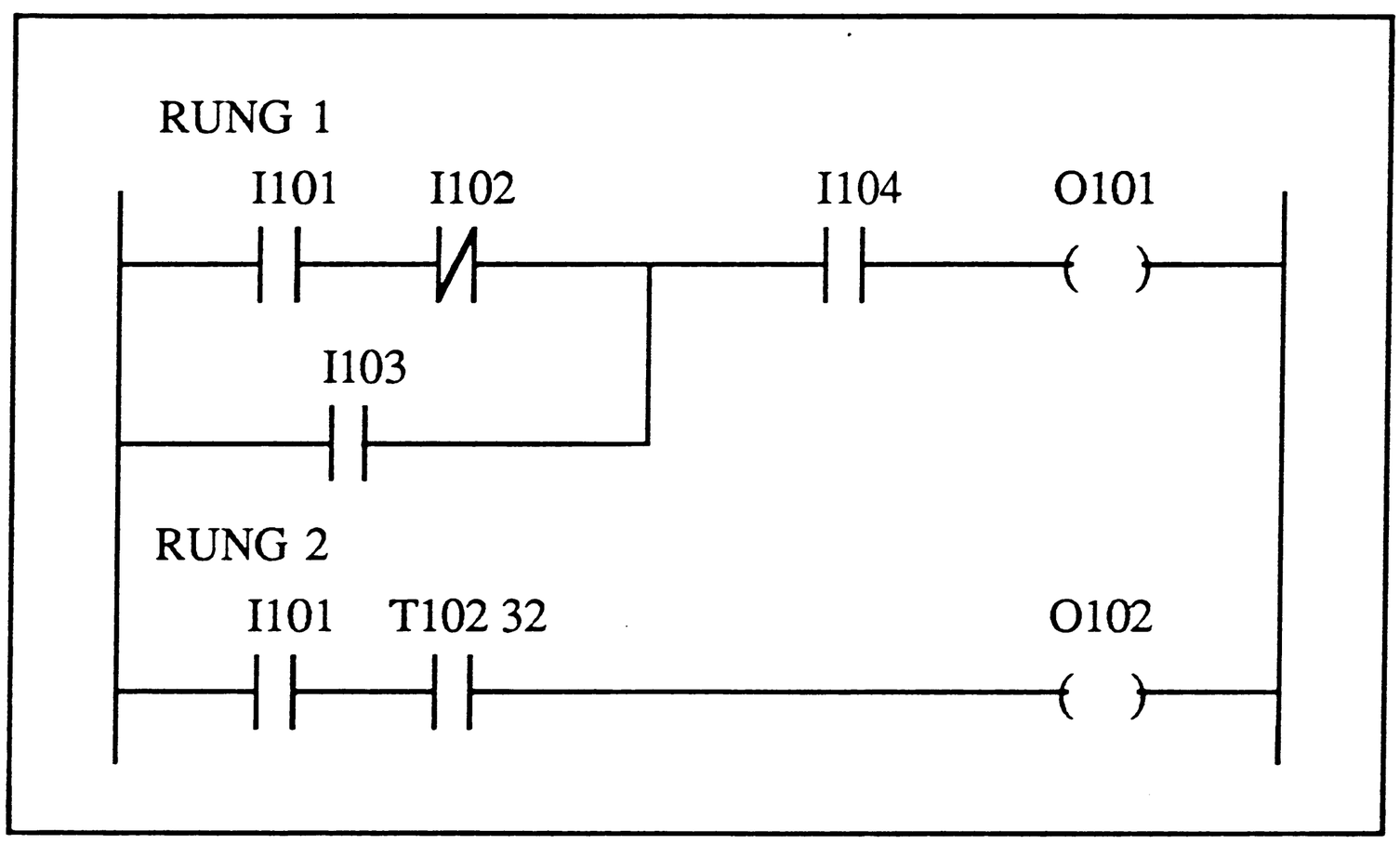


Figure 2.1 Diagram of PLC function blocks (Johnson, 1987)

PLCs offer several advantages over a conventional relay type of control. The programmable logic controller has eliminated much of the hard-wired requirements

associated with conventional relay control circuits. It is small and inexpensive compared to equivalent relay-based process control system. It also offers solid-state reliability, lower power consumption, and ease of expendability. Unlike the computer, the PLC is designed to operate in the industrial environment with wide ranges of ambient temperature and humidity. A well-designed PLC is not usually affected by the electrical noise that is inherent in most industrial locations.



Note: I = Input, O = Output, T = Timer

Figure 2.2 PLC relay ladder logic diagram

2.2.2 Application Areas

The programmable logic controller is used in a wide variety of control applications today. Applications for PLCs can be categorized in a number of different ways,

including industrial and general application categories. Industrial applications include many in both discrete manufacturing and process industries. Automotive industry applications, the genesis of the programmable controller, continue to provide the largest base of opportunity. Other industries, such as food processing and utilities, provide current development opportunities.

There are five general application areas in which programmable logic controllers are used. These five general areas are explained briefly below (Johnson, 1987).

1. ***Sequence Control.*** This is the largest and most common application for PLCs today, and is the closest to traditional relay control in its 'sequential' nature. Sequential control is found on individual machines or machine lines, on conveyor and packaging machinery, and even on modern elevator control systems.

2. ***Motion control.*** This is the integration of linear or rotary motion control in the PLC. This could be a single or multiple axis drive system control, and can be used with servo, stepper, or hydraulic drives. Examples of this control include metal cutting (grinders, lathes, and milling machines), metal forming (press brake), assembly machines, and other applications requiring multiple axes of motion which can be coordinated for both discrete part and process industry applications.

3. ***Process Control.*** This is the ability of the PLC to control a number of physical parameters such as temperature, pressure, velocity, and flow. It involves the use of analog (continuously variable) I/O to achieve a closed-loop control system. The use of *Proportional-Integral-Derivative* (PID) software allows the PLC to replace the function of stand-alone loop controllers. Typical examples of applications include plastic injection molding machines, extrusion process machines, heat treat furnaces, and many other batch-type control applications.

4. ***Data Management***. With the advanced instruction sets and expanded variable memory capacities of the newer PLCs, it is now possible for the system to act as a data concentrator, collecting data about the machine or process it is controlling. This data can then be compared to reference data in the memory of controllers, or can be sent via a communication link to another intelligent device for analysis or report generation. Data management is frequently found on large material handling systems, in unmanned flexible manufacturing cells, and in many process industry application.

5. ***Communications***. This is the ability for a PLC to have a 'window' to other PLCs and intelligent devices. One of the most active development areas in today's industrial control arena and much *Local Area Network* (LAN) activity is currently driven by the *Manufacturing Automation Protocol* (MAP) communications standard. MAP, an activity initiated by General Motors, is intended to connect multivendor intelligent devices. Communications are most often used with a factory host computer, for the purpose of collecting process data and configuring the controllers for a certain production sequence.

As mentioned above, these applications may be found alone, or in combination on a variety of apparatus, Table 2.2 lists a sample of the many applications and industries using PLCs.

2.2.3 Case Study

The engineering team of Cincinnati Milacron's Electronic System Division based at Biggleswade in Bedfordshire have combined their vast experience of NC, PLC, microcomputer and programming language to produce a controller for a manufacturing cell (Taylor, 1987). It has a microcomputer for data management and multi-line communication as well as the NC systems and a PLC for parts movement and

scheduling which allows the versatility of the available I/O to provide a standard interface to the cell. This cell is called the Acramatic Flexible Manufacturing Cell Controller (FMCC) which consists of a three bay floor standing console designed for shop floor use in an industrial environment.

Table 2.2 Programmable logic controller applications (Johnson, 1987)

Annunciators	Injection Molding
Auto Insertion	Assembly
Bagging	Motor Winding
Baking	Oil Fields
Blending	Painting
Boring	Palletizers
Brewing	Pipelines
Calendaring	Polishing
Casting	Reactors
Chemical Drilling	Robots
Color Mixing	Rolling
Compressors	Security Systems
Conveyors	Stretch Wrap
Cranes	Slitting
Crushing	Sorting
Cutting	Stackers
Digestors	Stitching
Drilling	Stack Precipitators
Electronic Testing	Treading
Elevators	Tire Building
Engine Test Stands	Traffic Control
Extrusion	Textile Machine
Forging	Turbines
Generators	Turning
Gluing	Weaving
Grinding	Web Handling
Heat Treating	Welding

Data management occupies one bay of the cabinet and employs an IBM PC XT microcomputer. Storage and maintenance of all part program information is transmitted

to the requesting machine control system. Data may be input to the console from a paper tape reader, from the machine control systems, or from an optional interface to the next higher level of communication within the total manufacturing scheme. Any stored data may be output on command to a paper tape punch or to the higher level communication interface. This data may include statistical information transmitted from the machine control system such as probe data, tool data, cycle times, part counts and spindle utilization.

The central bay of the console provides a control monitoring and supervisory facility for the movement and scheduling of parts through the manufacturing cell. The cabinet contains an Acramatic APC-2 process control and scheduling computer, a visual display unit (VDU) and a sealed touch keyboard with VDU addressed functions. Standard interface signals are provided for the workstations, loaders or automatic work changes where jobs are introduced to or removed from the cell and for the pallet transport system. This allows the console to control the movements of any type of the parts handling system, the typically linked conveyors, robots or automatically guided vehicles (AGV).

The third bay of the FMCC provides a control monitoring facility for the CNC systems in the cell. The console contains a visual display unit and sealed touch keyboard which duplicates the data display and entry facilities of any Acramatic CNC control in the cell. Upon selecting the required control system, the operator has access to all the control system displays and can manually input data as if he were at the workstation.

As illustrated in the above case study, it would be beneficial to integrate three similar systems in the ManTech Lab: the Cartrac currently controlled by PLCs; the AS/RS controlled by a microcomputer and a PLC; and the K&T milling center controlled by

a CNC controller. An IBM 7552 industrial computer will play a major role, acting as a cell controller and coordinating these three systems with the automated material handling system.

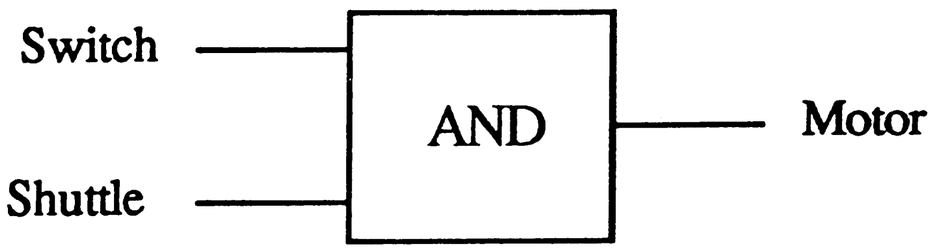
2.3 Fundamentals of Logic

2.3.1 The Binary Concept

The PLC, like all digital equipment, operates on the binary principle. The term *binary principle* refers to the idea that many things can be thought of as existing in one of two states. The states can be defined as "high" or "low," "on" or "off," "yes" or "no," and "1" or "0." For example, a light can be on or off, a switch open or closed, or a motor running or stopped. This two-state binary concept, applied to gates, can be the basis for making decisions. The gate is a device that has one or more inputs with which it will perform a logical decision and produce a result at one output. Figure 2.3 gives two examples that show how logic gate decisions are made.

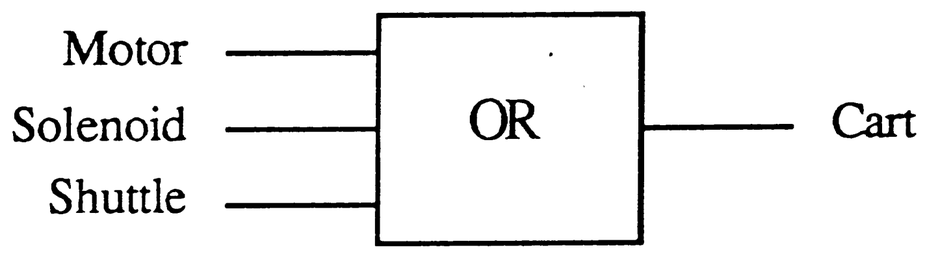
2.3.2 Combinational Logic - AND, OR, NOT

The operations performed by digital equipment are based on three fundamental logic functions: AND, OR, and NOT. Each function has a rule that will determine the outcome and a symbol that represents the operation. For the purpose of this discussion, the outcome or output is called M and the signal inputs are called A,B,C, etc. Also, binary 1 represents the presence of a signal or the occurrence of some event, while binary 0 represents the absence of the signal or nonoccurrence of the event.



The motor can be turned on only when the switch is on and the shuttle arrives at the specified station.

(a) The Logical AND



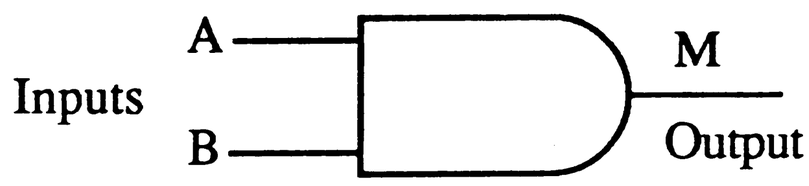
The cart can move when the motor is on or the solenoid is activated or the shuttle is at home position.

(b) The Logical OR

Figure 2.3 The logic gate decision

The AND Function

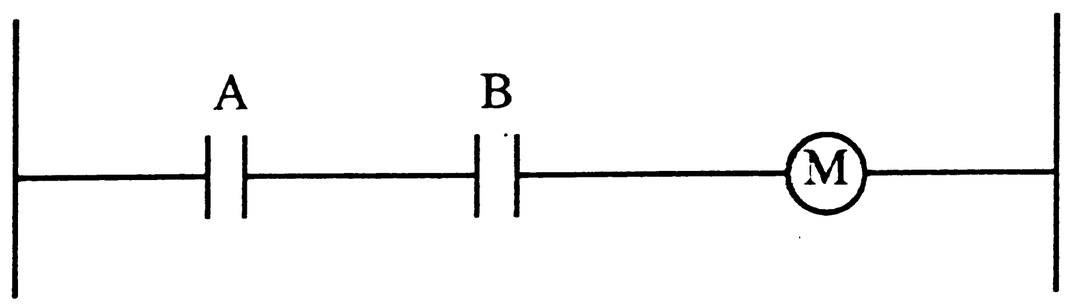
The symbol drawn in Figure 2.4(a) is called an AND gate. An AND gate is a device with two or more inputs and one output. The AND gate output is 1 only if all inputs are 1. The truth table in Figure 2.4(b) shows the resulting output from each of the possible input combinations. Figure 2.4(c) shows a practical application of the electrical circuit for an AND gate function.



(a) Two-input AND gate symbol

Inputs		Outputs
A	B	M
0	0	0
0	1	0
1	0	0
1	1	1

(b) AND tunth table

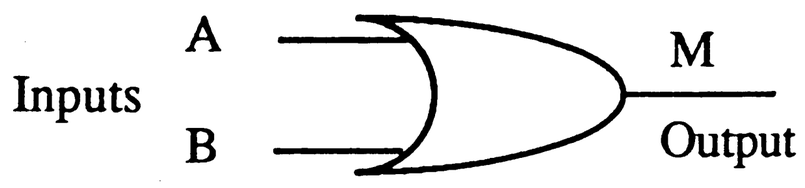


(c)

Figure 2.4 Diagram of logic AND function

The OR Function

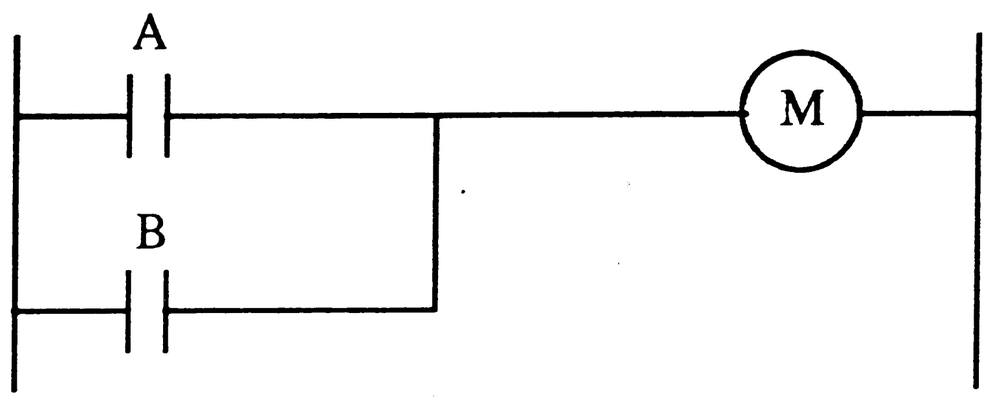
The symbol drawn in Figure 2.5(a) is called an OR gate. An OR gate can have any number of inputs but only one output. The OR gate output is 1 if one or more inputs are 1. The truth table in Figure 2.5(b) shows the resulting output M from each possible input combination. Figure 2.5(c) shows a practical application of an OR gate function.



(a) Two-input OR gate symbol

Inputs		Outputs
A	B	M
0	0	0
0	1	1
1	0	1
1	1	1

(b) OR tunth table

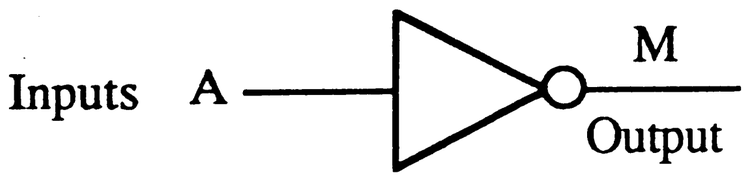


(c)

Figure 2.5 Diagram of logic OR function

The NOT Function

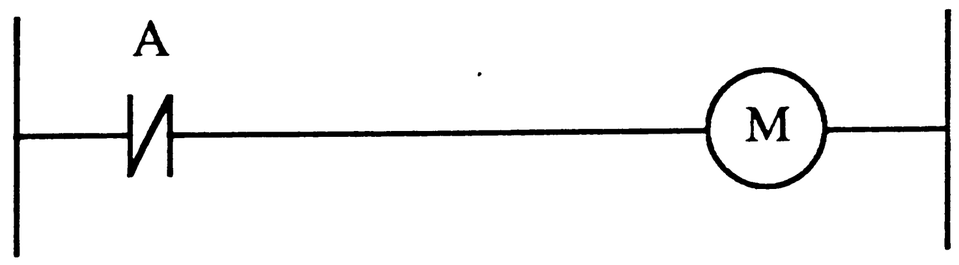
The symbol drawn in Figure 2.6(a) is that of a NOT function. Unlike the AND and OR function, the NOT function can have only one input. The NOT output is 1 if the input is 0. The output is 0 if the input is 1 (Figure 2.6(b)). The result of the NOT operation is always the inverse of the input and the NOT function is, therefore, called an *inverter*. Figure 2.6(c) shows an example of a practical application of the NOT function.



(a) NOT function

Inputs	Outputs
A	M
0	1
1	0

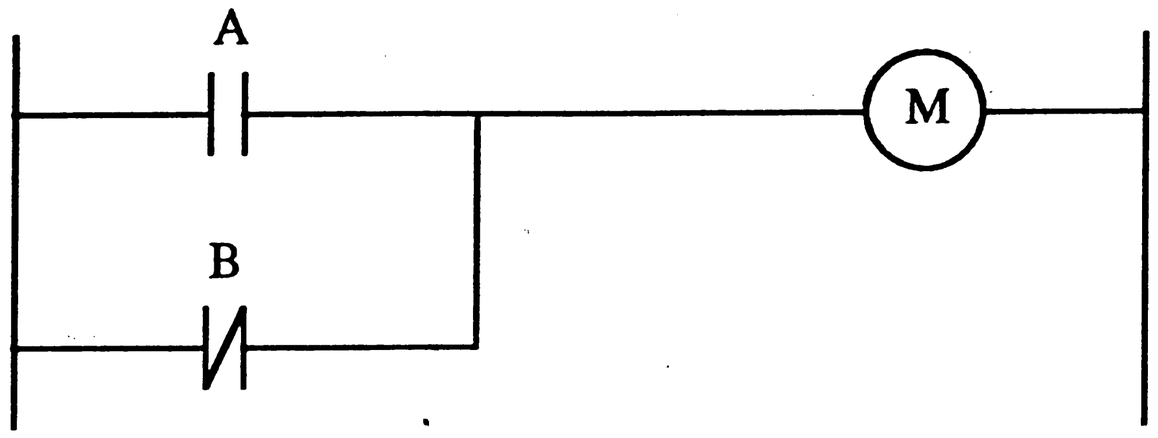
(b) NOT tunth table



(c)

Figure 2.6 Diagram of logic NOT function

The NOT function is most often used in conjunction with the AND or the OR gate. Figure 2.7(a) shows the NOT function connected to one input of an OR gate and the truth table is shown in Figure 2.7(b).



(a)

A	B	NOT B	M
0	0	1	1
0	1	0	0
1	0	1	1
1	1	0	1

(b) Truth table

Figure 2.7 NOT gate function application

2.3.3 Hard-Wired Logic Versus Programmed Logic

The term *hard-wired logic* refers to logic control functions that are determined by the way devices are interconnected. Hard-wired logic can be implemented using relays and relay ladder diagrams. Relay ladder diagrams are universally used and understood in industry. Figure 2.8 shows a typical relay ladder diagram of a motor STOP/START control station with pilot lights.

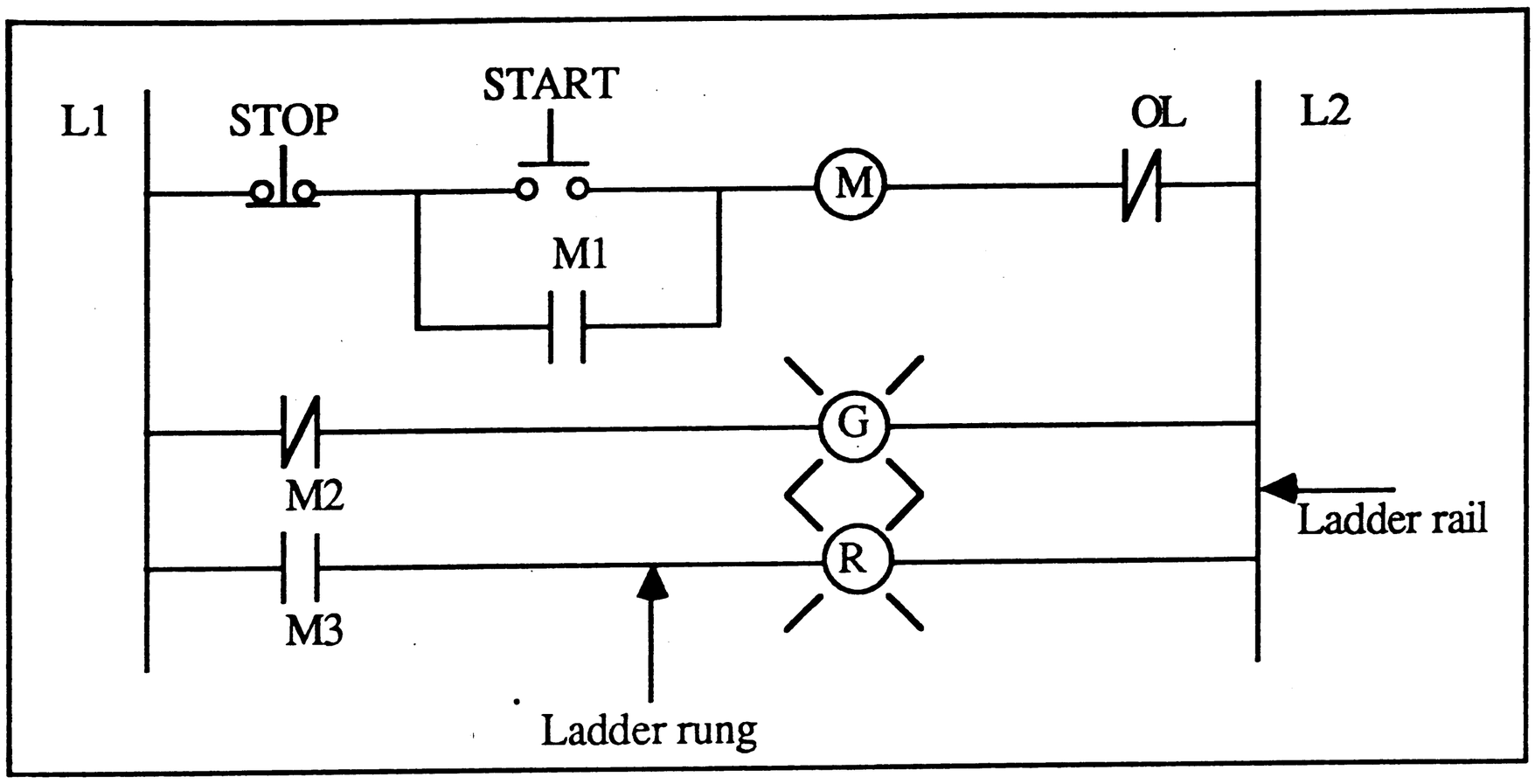


Figure 2.8 Relay ladder diagram (Petruzella, 1989).

The control scheme is drawn between two vertical power supply lines. All the components are placed between these two lines, called *rails* or *legs*, connecting the two power lines with what look like *rungs* of a ladder - thus the name, *ladder diagram*. Table 2.3 shows some hard-wired devices of the common sensors and actuators that represent the physical inputs and outputs of programmable control systems (Johnson, 1987).

Table 2.3 Sensors and actuators (Johnson, 1987)

Sensors	Actuators
Pushbuttons	Motor Starters
Limit Switches	Indicator Lights
Photo Cells	Variable Drives
Proximity Switches	Solenoids
Thermocouples	Control Relays
Potentiometers	Analog Valves
Load Cells	Chart Recorders

Included are the pushbutton and limit switches, solenoids and pilot lights. These are the arms, legs, eyes and ears of the control system, and their status is registered and controlled by the software of the control program. Unlike the hard-wired logic which is fixed and can be changed only when the devices are altered or added, programmable control is based on the basic logic functions which are programmable and easily changed. These functions (AND,OR,NOT) are used either singly or in combination to form instructions that will determine if a device is to be switched on or off. The form in which these instructions are implemented to convey commands to the PLC is called the *language*. The most common PLC language is *ladder logic language*. Figure 2.9 shows the transition from the relay ladder diagrams to the ladder logic (or programmable control) language.

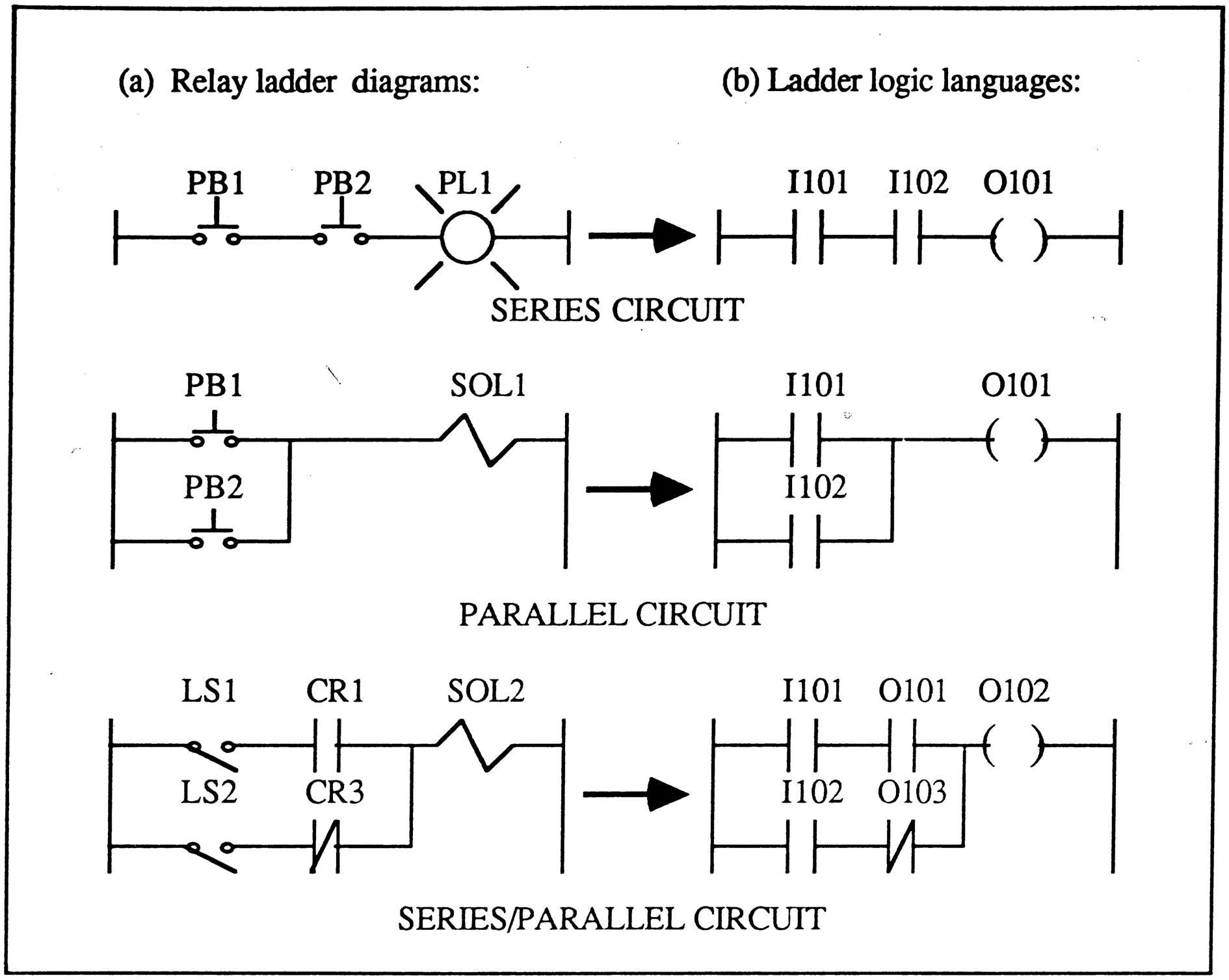


Figure 2.9 Diagram of conversion examples.

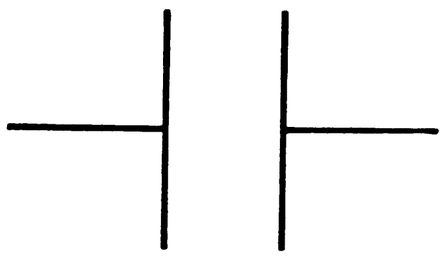
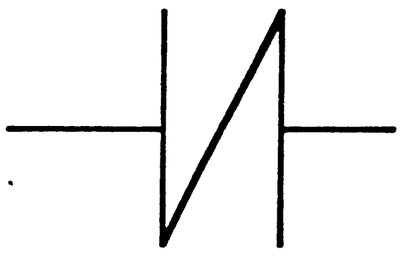
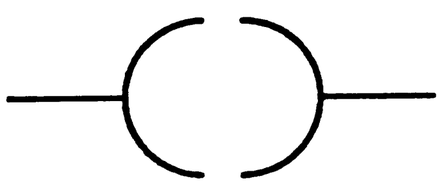
2.4 Programming Techniques

2.4.1 Relay-Type Instructions

The **ladder logic language** is basically a *symbolic* set of instructions used to create the controller program. These ladder instruction symbols are arranged to obtain the desired control logic that is to be entered into the memory of the PLC. Because the instruction set is composed of contact symbols, ladder logic language is also referred to as *contact symbology*.

Representations of contacts and coils are the basic symbols of the ladder logic instructions set. Table 2.4 shows the basic symbols used to translate relay control logic to contact symbolic logic (Petruzella, 1989).

Table 2.4 Contact,Coil,and Mnemonic Convention and Notations

 Normally Open Contact	Represents an input to the control logic. The active state places a 1 in the logic status stable. Inactive, the status table receives a 0. "Current" flows in the active state.
 Normally Closed Contact	Similar to the normally open contact, except that the logic states are reversed. "Current" flows in the inactive state.
 Output(coil)	Represents an output from control logic. It is "energized" from any combination of true input contacts and can be a real-world output or an internal coil output.

The main function of the ladder logic program is to control outputs based on a set of input conditions. A ladder logic program can be broken down into rungs. In general, a rung consists of a set of input conditions, represented as contacts, and an output instruction at the end of the rung (Figure 2.10). The output instruction could consist of coils, counters, timers, etc. Each control or coil symbol is referenced with an address

number that identifies what is being evaluated and controlled.

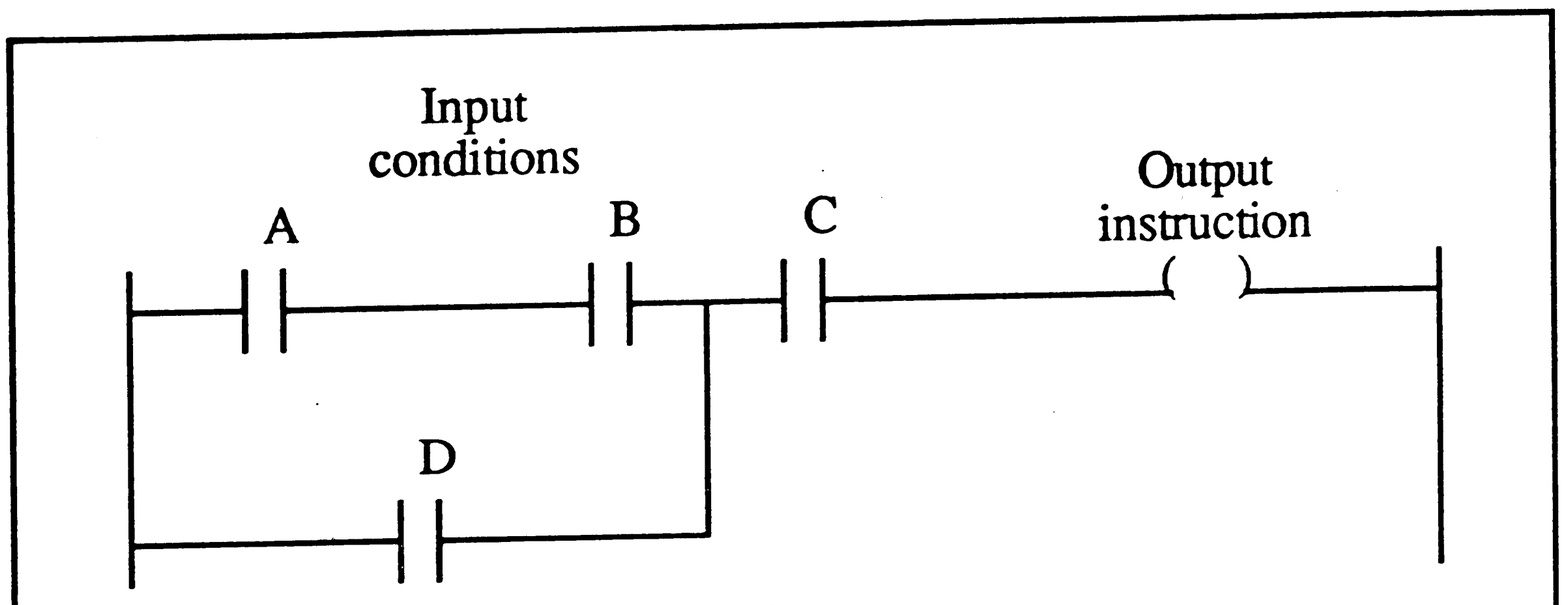


Figure 2.10 Ladder rung (Petruzella, 1989).

For an output to be activated or energized at least one left-to-right path of contacts must be closed. A complete closed path is referred to as having *logic continuity*. When logic continuity exists in at least one path, the rung condition is said to be TRUE. The rung condition is FALSE if no path has continuity (Figure 2.11).

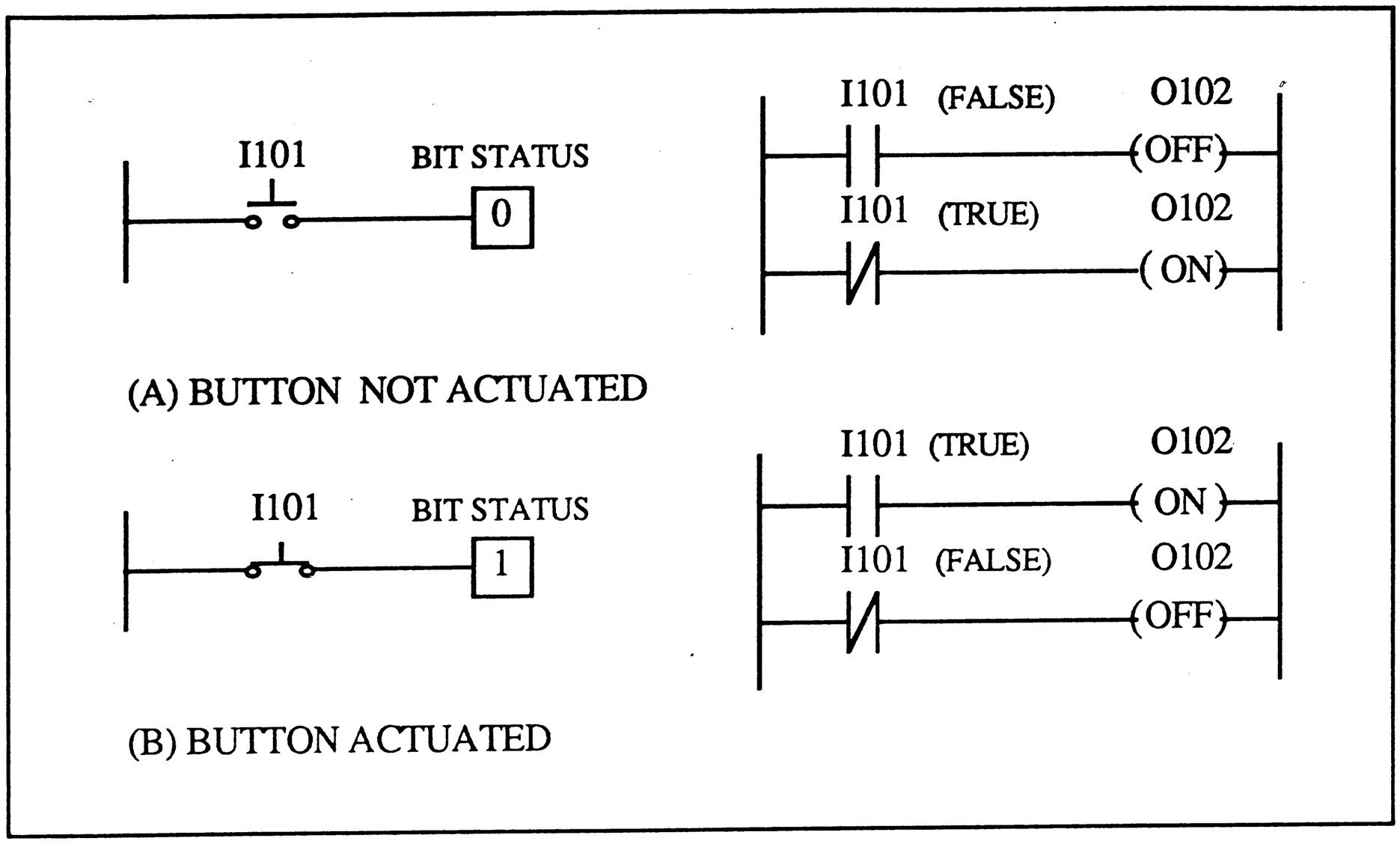


Figure 2.11 Status bit examples.

2.4.2 Instruction Addressing

To complete the entry of a relay-type instruction, an address number must be assigned to it. This number will indicate what PLC input is connected to what input device and what PLC output will drive what output device (Figure 2.12)

The addressing of real inputs and outputs, as well as internals, depends upon the PLC model used. When using Allen Bradley PLC-4s and referring to Figure 2.12, each input and output bit has a four-character address explicated as follows:

1. The first character is an I if the address is an input address or an O if the address is an output address.
2. The second character identifies the communication loop number of the PLC being

programmed.

3. The third and fourth characters match the numbers of the bit addresses of the inputs or outputs in the PLCs.

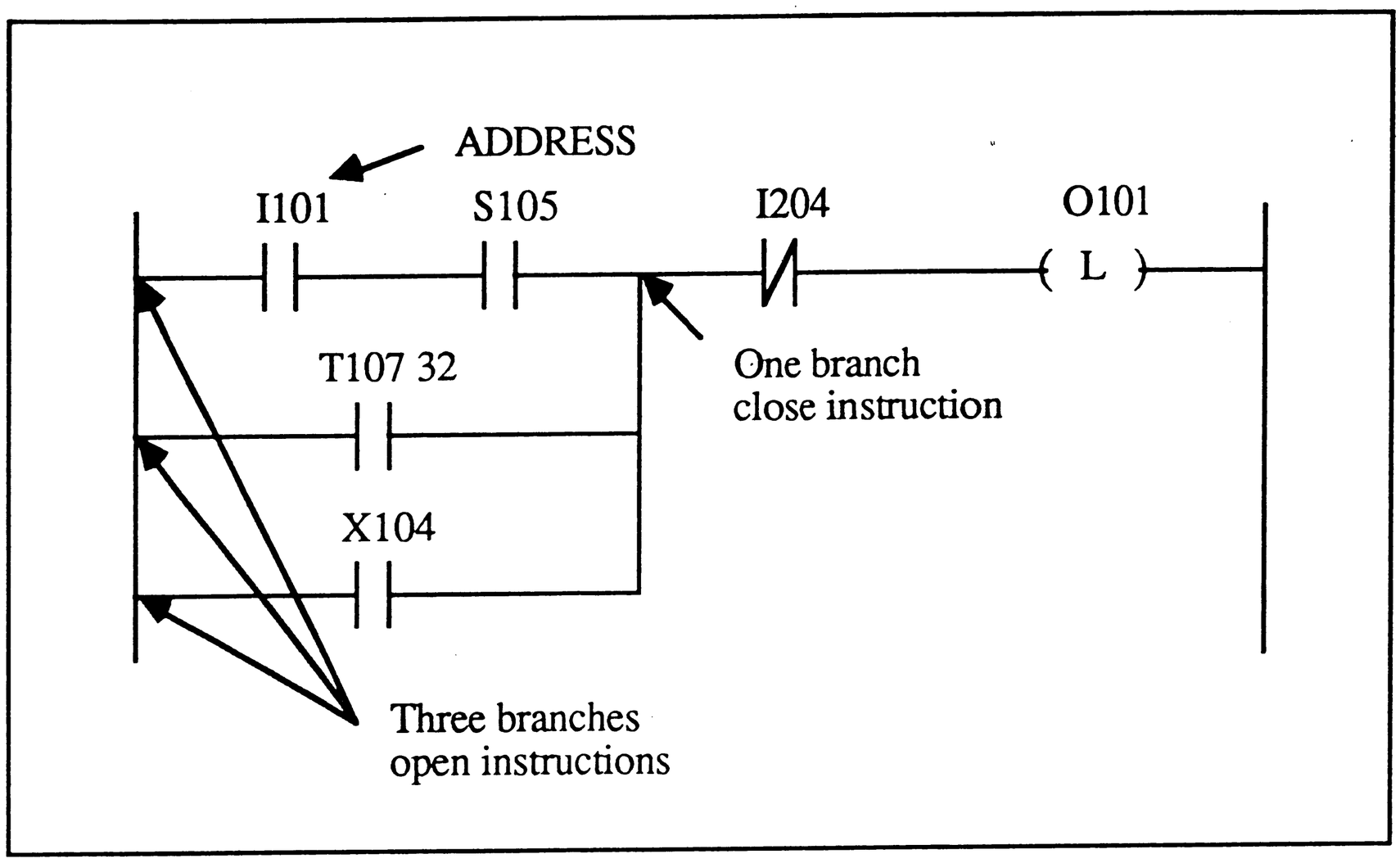


Figure 2.12 Typical addressing format.

Table 2.5 illustrates the various types of addresses that programmers can use to create the address numbers in the program (Bulletin 1773, 1983).

2.4.3 Branch Instructions

Branch instructions are used to create parallel paths of input condition instructions. This allows more than one combination of input condition (OR logic) to establish logic continuity in a rung. Figure 2.12 illustrates a simple branching condition. The rung will be TRUE if at least one path is TRUE. A branch OPEN instruction is used to begin

each parallel logic branch. A single branch CLOSE instruction is used to close the parallel branch.

Table 2.5 The types of address.

Type of address	Controller I.D.	Bit address range
I = Input	1 - 8	01 - 20
O = Output	1 - 8	01 - 12
X = Flags	1 - 8	01 - 32
S = Stores	1 - 8	01 - 99

Type of address	Controller I.D.	Timer/Counter number	Bit description
T = Timer	1 - 8	01 - 32	15 - Timer clock 16 - Enable 31 - Timer timing 32 - Done
C = Counter	1 - 8	01 - 32	15 - Count-down enable 16 - Count-up enable 31 - Overflow underflow 32 - Done

For each PLC model there is a limit to the number of series contact instructions that can be included in one rung of a ladder diagram, as well as a limit to the number of parallel branches. The only limitation on the number of rungs in PLC is memory size. Figure 2.13 shows the matrix limitation diagram for the Allen Bradley PLC-4. A maximum of four parallel line (paths) and seven series contacts per rung is possible.

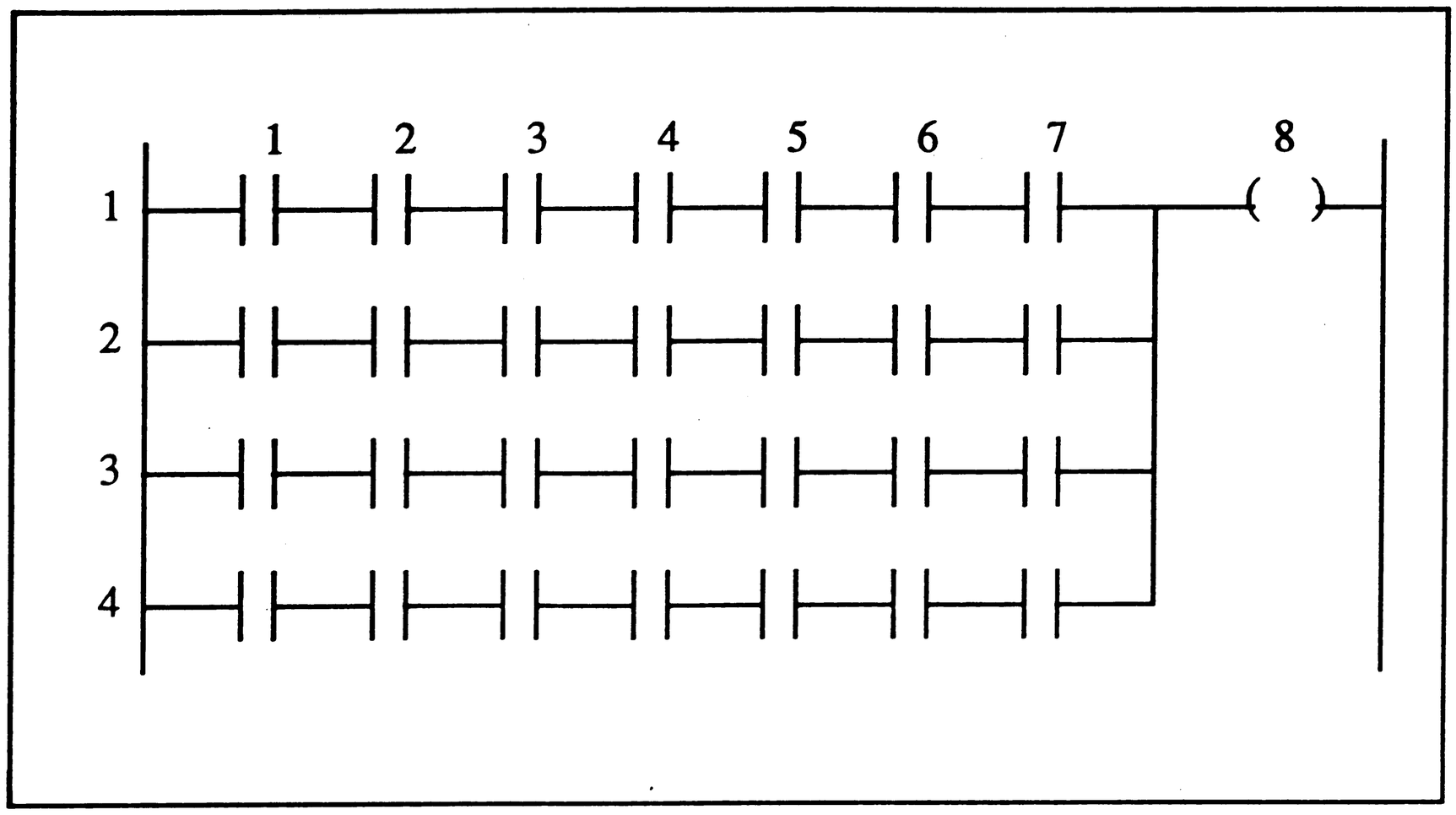


Figure 2.13 Typical PLC matrix limitation diagram

2.4.4 Internal Relay Instructions

Most PLCs have an area of the memory allocated for what are known as *internal storage bits*. These storage bits are also called *internal stores, internal coils, internal control relays*, or just *internals*. The internal store operates just as any output that is controlled by programmed logic; however, the store is used strictly for internal purposes. In other words, the internal store does not directly control an output device.

An internal control relay can be used when a circuit requires more series contacts or parallel lines than allowed in a rung. Figure 2.14 shows a circuit that allows for only seven series contacts when ten are actually required for the programmed logic. To solve this problem, the contacts are split into two rungs as illustrated in Figure 2.15. As referred to in Table 2.5 and Figure 2.15, the internal storage bits are addressed with four characters. These characters include:

1. S for a store.

2. The second character identifies the number of the PLC.
3. The third and forth characters are the address range which is from 01 to 99 to identify a particular bit in memory that function as a store. No more than 99 stores can be used.

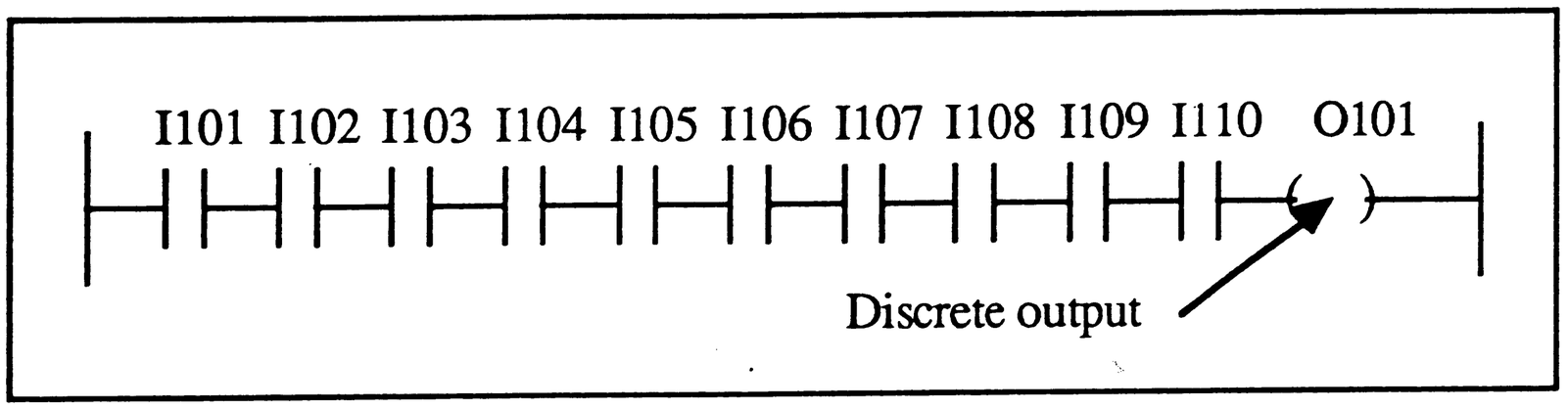


Figure 2.14 Contacts exceed limit of seven.

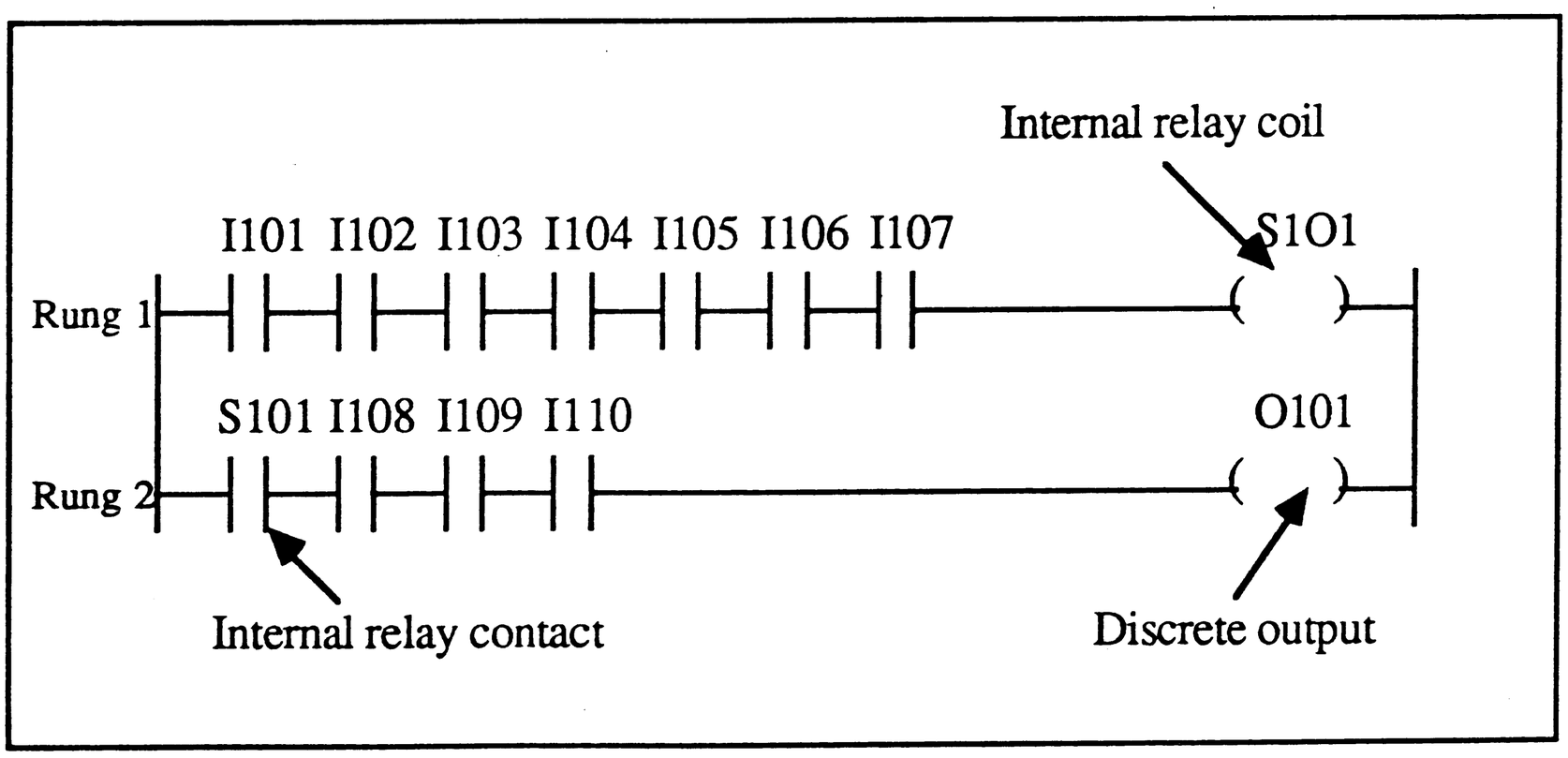


Figure 2.15 Contacts split into two rungs.

2.4.5 Programming Software for PCs

In industrial factory automation applications today, programmable logic controllers represent the typical remote front-end computer, perform real-time control, and

communicate with mini or microcomputers used as supervisory stations.

The relatively wide use of PLC relay ladder logic language stemmed from the fact that most discrete automation control systems were based on relay interconnections, and the programming symbols as well as documentation were based on pictorial representation of relay contacts. In order to interface with PLCs, a computer needs communications software designed for each specific PLC protocol. Application programs can call software subroutines to read and write blocks of data to or from the PLC. This can be done with most programming languages, such as BASIC, C, PASCAL and FORTRAN, etc. Sometimes, special hardware is also required to allow communication between the PLC and the computer.

2.4.6 Case study

A mineral processing plant located in South Africa utilizes a distributed sequence control with a high level sequencing language in conjunction with PLCs (Charalambous, 1988). The advantage of using a high level language plus a structured design technique allows the use of computers and PLCs to cost effectively address sequence and batch control applications formerly tackled with control equipment. Figure 2.16 shows the configuration of the PLC and supervisory computer system (Charalambous, 1988). The requirements of large sequence control systems are as follows:

1. The sequences have to be configurable using a high level sequencing language to increase productivity during system development.
2. The system contains a number of trips and interlocks which have to operate the sequences independently. These shutdown sequences have to be implemented in

the final system.

3. With the large volume of information, a strategy has to be devised on how to display and manipulate variables.
4. The control strategy to control the plant has to be simple, so as to aid the operators.
5. There is also a requirement to interface a laboratory computer system to the control system which will pass on the results of analyses which are used by the sequences.

The design requirements dictated the following functional approaches:

1. High level language ("CASL") for the supervisory computer.
2. Low level language ("Ladder Logic Diagram") for PLCs.
3. Software layering.
4. Hot standby.
5. Wide bandwidth Local Area Network (LAN).
6. Distributed processing.

The high level language - CASL is a "control and sequencing language" which has been used in several hundred applications in England and Europe. It is not sufficient to use a general purpose high level language for batch control. Essential additional features to be incorporated include:

1. High level commands (e.g. OPEN, CLOSE for valves).
2. Online sequence changes.

3. In-line documentation.

4. Sequence backup.

5. Error detection and recovery.

The system must present information to the operator in a clean and concise fashion, such as:

1. The control system should be able to provide a graphic display of the state of any part process.
2. On the same display, the system has to provide information of any sequences that pertain to that section of the plant.
3. An additional piece of information which will give the operator a clear picture of what is going on in the control system, is the state of the interlocks.
4. Allow the operator to manually override or force an output in order to safely control the plant manually.

This case provides a model for the design and implementation of the low level and the high level commands. Because there are several limitations in the Cartrac system, by integrating the low level and the high level commands, the Cartrac system can be operated automatically by the computer without human intervention.

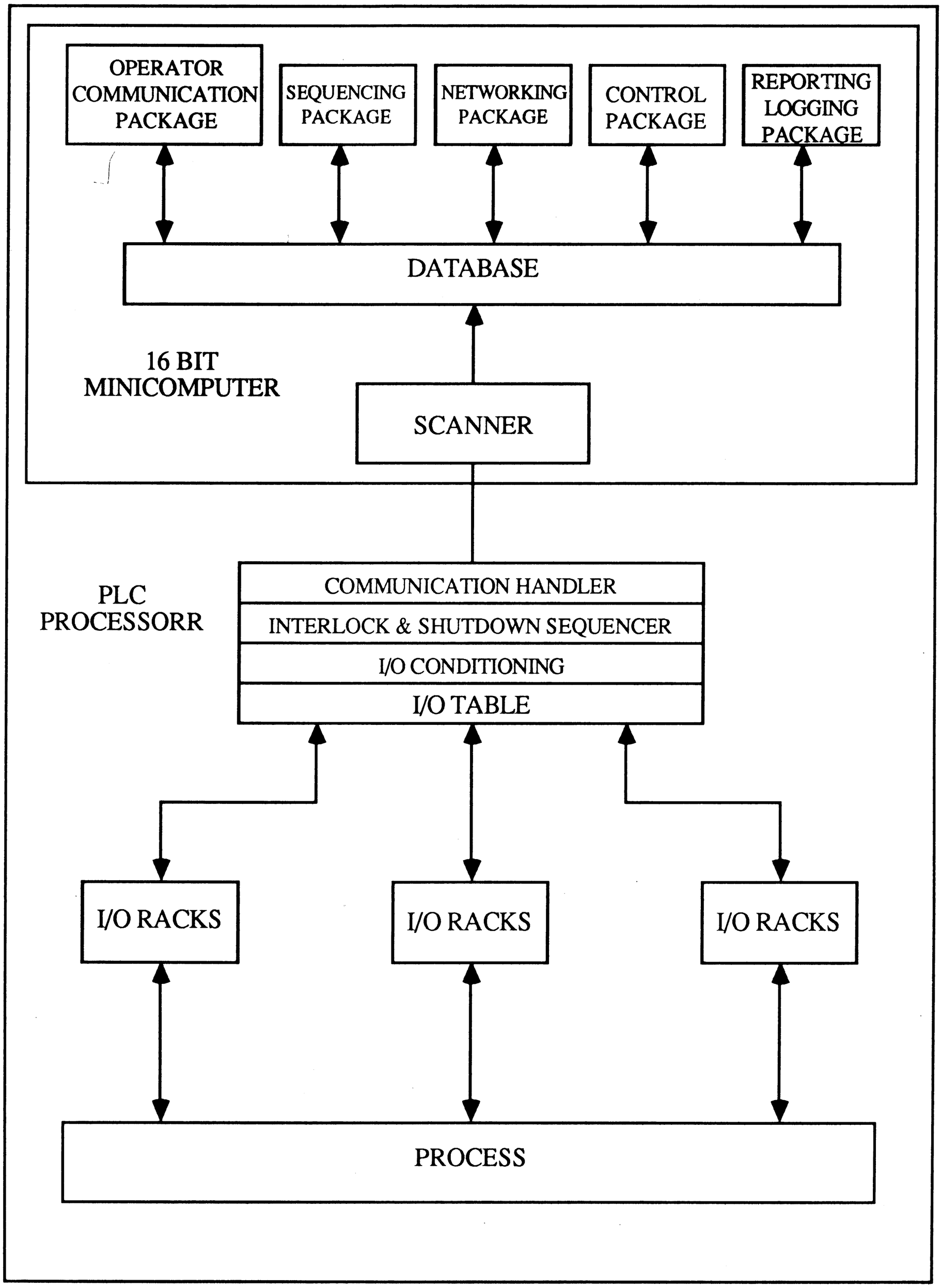


Figure 2.16 Configuration of PLC and Supervisiory Computer

Chapter 3 Low Level Commands

3.1 Physical Hardware Control

3.1.1 Input Control Devices

The sensors used with programmable logic controller systems are categorized as *manually operated switches* and *mechanically operated switches*. Some of these devices have been installed in the Cartrac system. The logic control of these devices will be interpreted as follows:

A ***manually operated switch*** is one that is controlled by hand. These include toggle switches, push-buttons switches, knife switches, and selector switches.

The *push-button switches* are commonly used by PLC systems. As referred to Figure 1.1, the Cartrac control panel (power supply station) uses three *illuminated push-buttons* for the system control. One of these push-buttons is to start and run the Cartrac system. The second push-button is used as a normal stop which stops the system under a normal status. Another push-button is provided for emergency stop in case of system faults or other hazardous conditions.

Before the improvement of the Cartrac system, selector switches and push buttons were used at each control station. Under the old control system, the direction of a cart was determined manually by using selector switches. The *momentary control push-buttons* on the control stations were used to release a cart at station 2 or return the shuttle to station 11 from station 13. Other illuminated push-buttons are employed for the emergency stops on the control stations.

A ***mechanically operated switch*** is one that is controlled automatically by such factors as pressure, position, or temperature. These include limit switches, proximity switches and temperature switches.

The *limit switch* is a very common industrial control device. Limit switches are designed to operate only when a predetermined limit is reached, and they are usually actuated by contact with an object such as a cart or a shuttle. These devices take the place of a human operator. Limit switches are used in the Cartrac system to govern the starting, stopping, or reversal of motors and release of the carts blocked by the pneumatically operated brake.

Proximity switches are part of a series of solid-state sensors. They sense the presence or absence of a target such as a cart without physical contact. The six basic types are magnetic, capacitive, ultrasonic, inductive, air jet stream, and photo electric. Photo sensors have been installed in the Cartrac system to detect whether the carts are situated on the shuttles or not. Other photo sensors installed at station 1 and station 2 also provide a function to examine whether a tote (or pallet) is present on a cart. Hence, by means of an input signal from the photo sensor, the control system of the AS/RS-Cartrac interface can determine whether to act to load or unload a tote or do nothing because there is no tote on the cart or on the turn-table.

3.1.2 Output Control Devices

A variety of output control devices can be operated by the controller output modules to control traditional industrial processes. These devices include pilot lights, control relays, motor starters, alarms, heaters, solenoids, solenoid valves, small motors, and horns.

A *solenoid* is an electromagnetic actuator that can be used to open and close a valve, electrical contact, or other mechanical device. The solenoids in the Cartrac system control pneumatic valves. They have been installed at all stations except station 11. Each solenoid can activate a pneumatic cylinder to position a plate to stop a cart. As referred to in Figure 1.3, the pneumatically operated brake is normally in the locked status, in other words, the cart cannot proceed to the adjacent station even though the rotating tube is still running. In order to unlock a pneumatically operated brake, a set of logic conditions such as the status of limit switches, photo sensors, motors, solenoids, or timers must be fulfilled.

The *motor* is another of the output devices in the Cartrac system. Motors are used to drive the rotating tubes which run between the two rails and power the carts. As mentioned above, the rotating tube can advance a cart from one station to an adjacent station. The status of the motors (ON or OFF) is also decided by a set of logic conditions. Specifically, the motor can be stopped when a shuttle or a cart touches a limit switch at the arrival station.

3.2 Division of Ladder Logic Programs

As stated before, it is difficult for people to understand and debug the large ladder logic programs stored in the PLCs. More importantly, the current configuration does not allow the use of a high level computer such as an IBM 7552 cell controller. Therefore, these large programs have been modified and divided into several smaller ladder logic programs. Each small ladder logic program executes a low level command. A main characteristic of the low level commands is that each one is an independent ladder logic program. In other words, the logic sequence of each command is not under the influence

of other low level commands.

In the Cartrac system, 21 low level commands are currently defined to move carts or shuttles from station to station. So far, using this technique, each Allen Bradley PLC except PLC #4 governs 6 to 7 low level commands. As illustrated in Figure 1.1, the control area of each PLC will be explicated as follows:

1. The low level commands in ***PLC #1***:

   - Command 1: move cart from station 2 to station 3.
   - Command 2: move cart from station 3 to station 9.
   - Command 3: move shuttle without cart from station 3 to station 4.
   - Command 4: move cart from station 3 to station 2.
   - Command 5: move cart from station 5 to station 4.
   - Command 6: move cart from station 4 to station 3.

2. The low level commands in ***PLC #3***:

   - Command 7: move cart from station 9 to station 8.
   - Command 8: move cart from station 8 to station 7.
   - Command 9: move cart from station 7 to station 6.
   - Command 10: move cart from station 7 to station 10.
   - Command 11: move shuttle without cart from station 7 to station 8.
   - Command 12: move cart from station 6 to station 5.
   - Command 13: move cart from station 10 to station 7.

3. The low level commands in ***PLC #2***:

- Command 14: move cart from station 10 to station 11A.
- Command 15: move cart from station 11A to station 13.
- Command 16: move cart from station 10 to station 11B.
- Command 17: move cart from station 11B to station 12.
- Command 18: move cart from station 12 to station 11A.
- Command 19: move cart from station 13 to station 11A.
- Command 20: move cart from station 11A to station 10.
- Command 21: move cart from station 11A to station 12.

Note: 11A indicates that the shuttle is in the "home" position.

11B indicates that the shuttle is in the "away" position.

3.3 Implementation of Low Level Commands

3.3.1 Structure of Programs

Compared to the structure of the original large programs, the revised low level commands are more organized, easier to understand, and easier to modify. Programmers can modify one low level command without affecting the logic sequence of another low level command because each low level command has an independent logic sequence. The typical structure of 21 low level commands is shown in Figure 3.1.

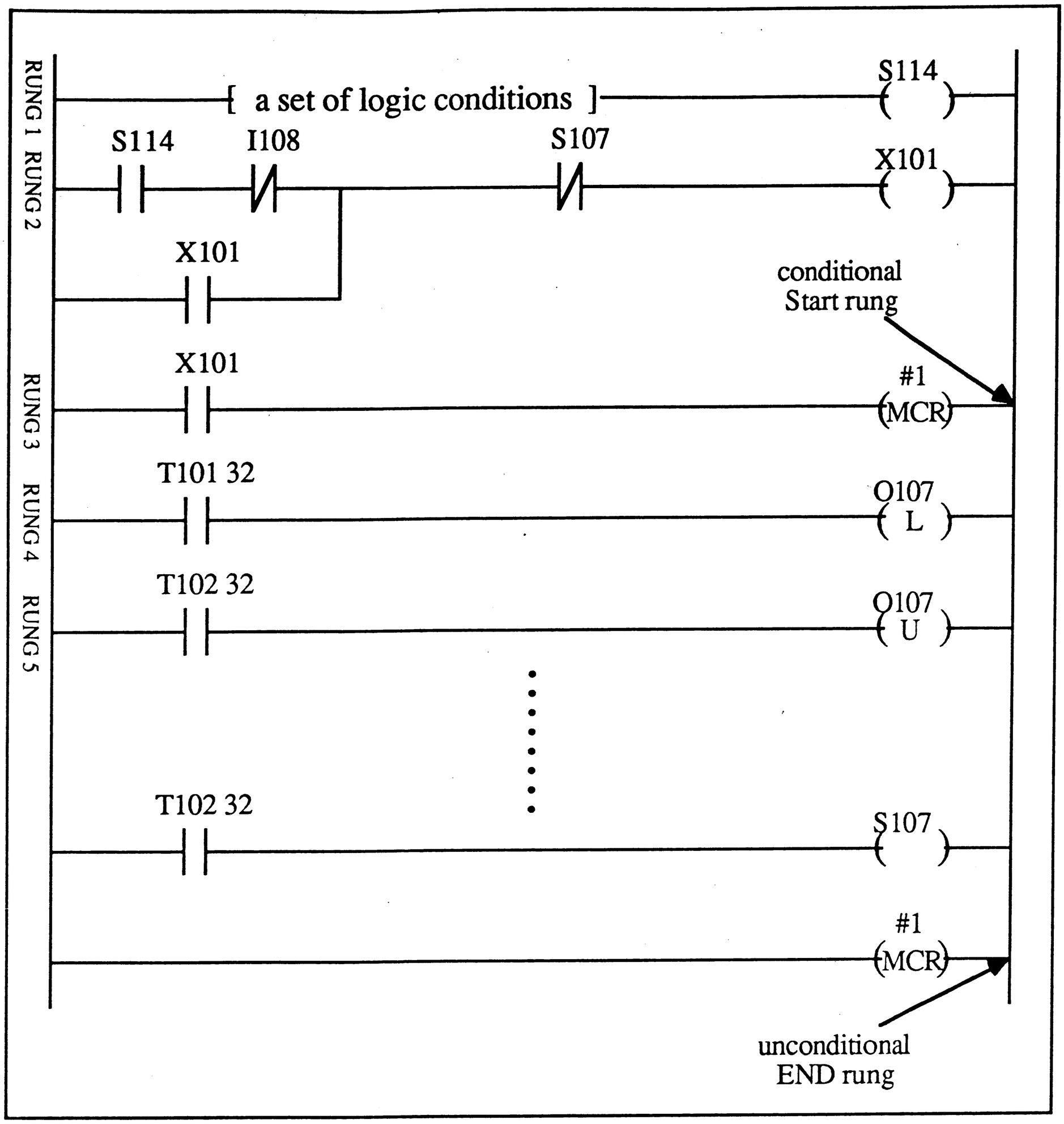


Figure 3.1 The structure of a low level command

The low level command programming technique will be explained in more detail by dividing it into the following sections:

Logic conditions to execute a low level command. For an output to be activated or energized, the path of logic conditions must be closed.

Master control reset. This instruction can be programmed to control an entire program or to control only selected rungs of a program.

Latching relays. They are used where it is necessary for contacts to stay opened and/or closed even though the coil is energized only momentarily.

Programming timers. They are used to activate or de-activate a device after a preset amount of time.

Addressing flag bits. Through the use of flag bits, the programs in one PLC can indirectly activate outputs on other PLCs.

3.3.2 Logic Conditions to Execute A Low Level Command

As mentioned in section 2.4.1, before the PLC activates the logic sequence of a low level command, the path of logic conditions (or contacts) must be completely closed. Otherwise, this low level command will be in the FALSE status and will not start execution. The PLC continues to scan the rest of the low level commands until one or more commands are in the TRUE status and then the PLC performs the logic sequence contained in these "TRUE" commands. Figure 3.2 shows a series of logic conditions used to activate command 1.

As stated earlier in section 2.4.4, an internal control relay such as S101 in the first rung can be used when a circuit requires more series contacts than the limitation of a rung. Allen Bradley PLC-4s allow for only seven series contacts in one rung. To solve this problem. the contacts are split into two rungs as Rung 1 and Rung 2 illustrated in Figure 3.2. The internal control relay is used strictly to combine the logic of the two rungs. It does not directly control an output device.

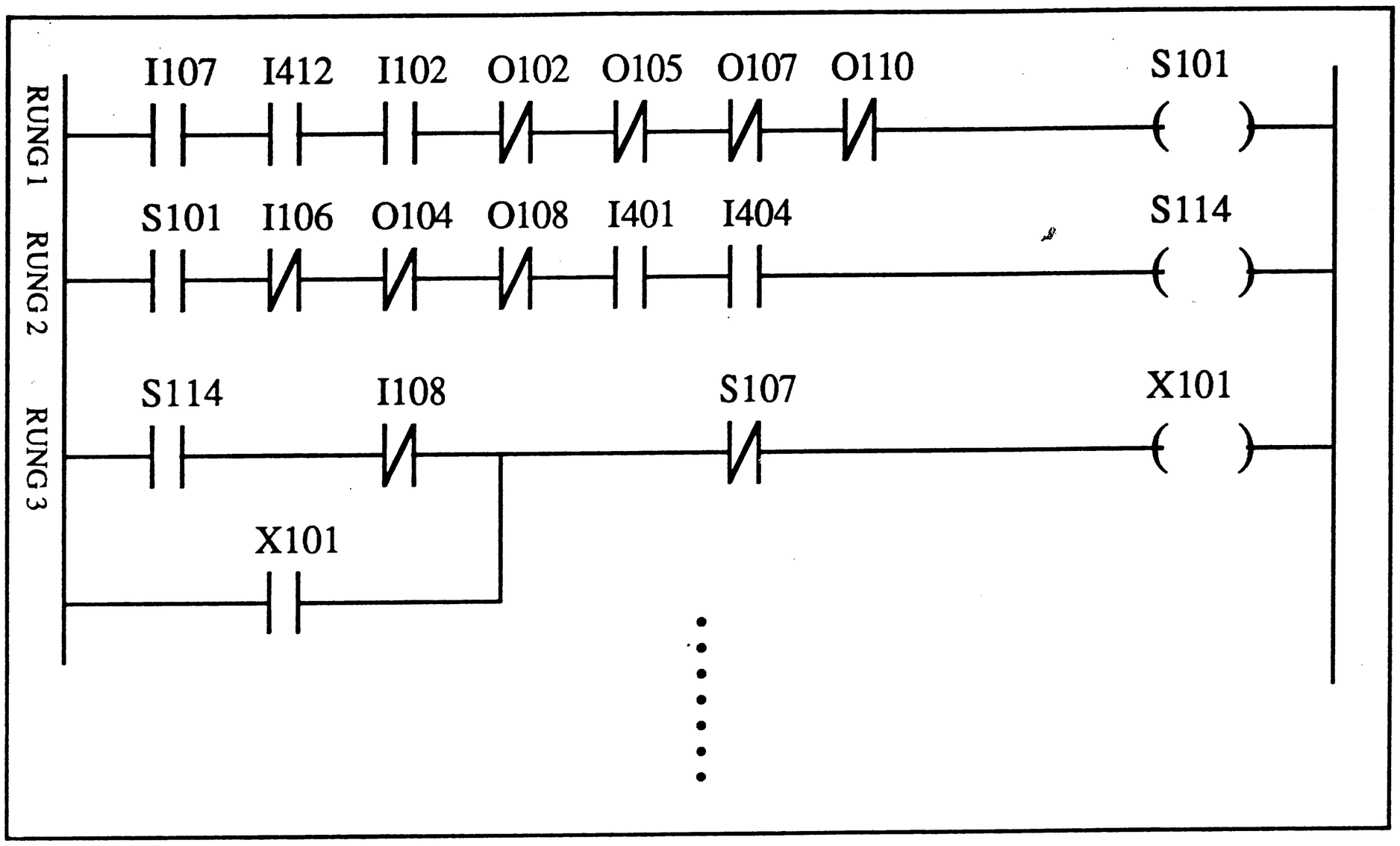


Figure 3.2 A set of logic conditions

As referred to in Figure 3.2, when a cart moves from station 2 to station 3, it must satisfy the following 13 conditions:

1. A cart is available at station 2, detected by limit switch (LS) I107.
2. A tote is on the cart at station 2, detected by photo sensor (PCR) I412.
3. A shuttle is available at station 3, detected by LS I102.
4. A pneumatical valve at station 3, which functions to pull a non-powered tube at station 3 into contact with a rotating tube at station 9, is not activated. It is detected by a solenoid O102.
5. The motor which rotates the tube to move from station 3 to station 4 is not energized. It is detected by O105.

6. A pneumatical valve at station 3, which functions to push a non-powered tube at station 3 into contact with a rotating tube at station 2, is not activated. It is detected by a solenoid O107.
7. The motor which rotates the tube to move from station 3 to station 2 is not energized. It is detected by O110.
8. No cart is present on the shuttle at station 3, detected by PCR I106.
9. The motor which rotates the tube to move from station 4 to station 3 is not activated. It is detected by O104.
10. A pneumatical valve at station 2 is not activated which means the cart will stop when it reaches this position, detected by the solenoid O108.
11. A load arm which moves a tote from station 1 to station 2 is completely retracted. It is detected by LS I401.
12. An unload arm which moves a tote from station 2 to station 1 is completely retracted. It is detected by LS I404.
13. The normally closed I108 is an output condition which receives a signal from one of the high level commands and determines the logic status of this low level command. In this case, I108 must be still closed.

The principle of programming the start of the other low level commands is similar to that of command 1 explicated above.

3.3.3 Master Control Reset

The *master control reset* instruction is represented with the symbol -(MCR)-. The MCR instruction can be programmed to control an entire circuit or to control only selected rungs of a circuit. Figure 3.1 represents a typical model of a programmed circuit with two MCR output instructions. By adding the second MCR instruction, the section of rungs between the two MCRs are called a master control reset zone. An MCR rung with conditional inputs is placed at the beginning of the fenced zone to be controlled. An MCR rung with no conditional inputs is placed at the end of the fenced zone to be controlled.

The MCR is an action instruction. When the MCR rung condition is TRUE such as the X101 instruction in Figure 3.1, the referenced output is activated and all rung outputs within the zone can be controlled by their respective input conditions.

If the MCR output goes FALSE, all non-retentive outputs within the fenced zone will be de-energized. As shown in Figure 3.1, the X101 instruction is a memory device that stays on when activated momentarily. The only way to turn off the MCR zone is to activate the internal control relay S107 in the zone. By linking contacts in series with S107, any one of several conditions could be used to turn off the command.

By using the MCR instructions, multiple low level commands can be operated in a single program. As long as the same inputs and outputs are not used in different low level commands, the low level commands remain independent.

3.3.4 Latching Relays

Electromagnetic latching relays are designed to hold the relay closed after power has been removed from the coil. Latching relays are used where it is necessary for contacts to stay opened and/or closed even though the coil is energized only momentarily (Petruzella, 1989). The latch instruction is represented by the symbol -(L)- and the latch coil is momentarily energized to set the latch and hold the relay in the latched position. The unlatch instruction is represented by the symbol -(U)- and the unlatch coil is momentarily energized to disengage the mechanical latch and return the relay to the unlatched position.

The use of the LATCH and UNLATCH coil instruction is illustrated in Figure 3.1. Both the latch -(L)- and the unlatch -(U)- coil have the same address (O107). When the timer T101 is momentarily actuated, the latch -(L)- in rung 4 becomes TRUE and the pneumatically operated brake is activated. This valve will remain activated when logical continuity of the latch rung is lost. When the unlatch -(U)- in the rung 5 becomes TRUE (timer T102 actuated), the valve is de-energized. The LATCH/UNLATCH instruction is *retentive* on loss of power. In other words, if the rung is latched, it will remain latched when power is lost and then restored.

3.3.5 Programming Timers

PLC timers are devices that provide the same functions as mechanical timing relays. They are used to activate or de-activate a device after a preset amount of time. The timer instruction as a relay coil is illustrated in Figure 3.3.

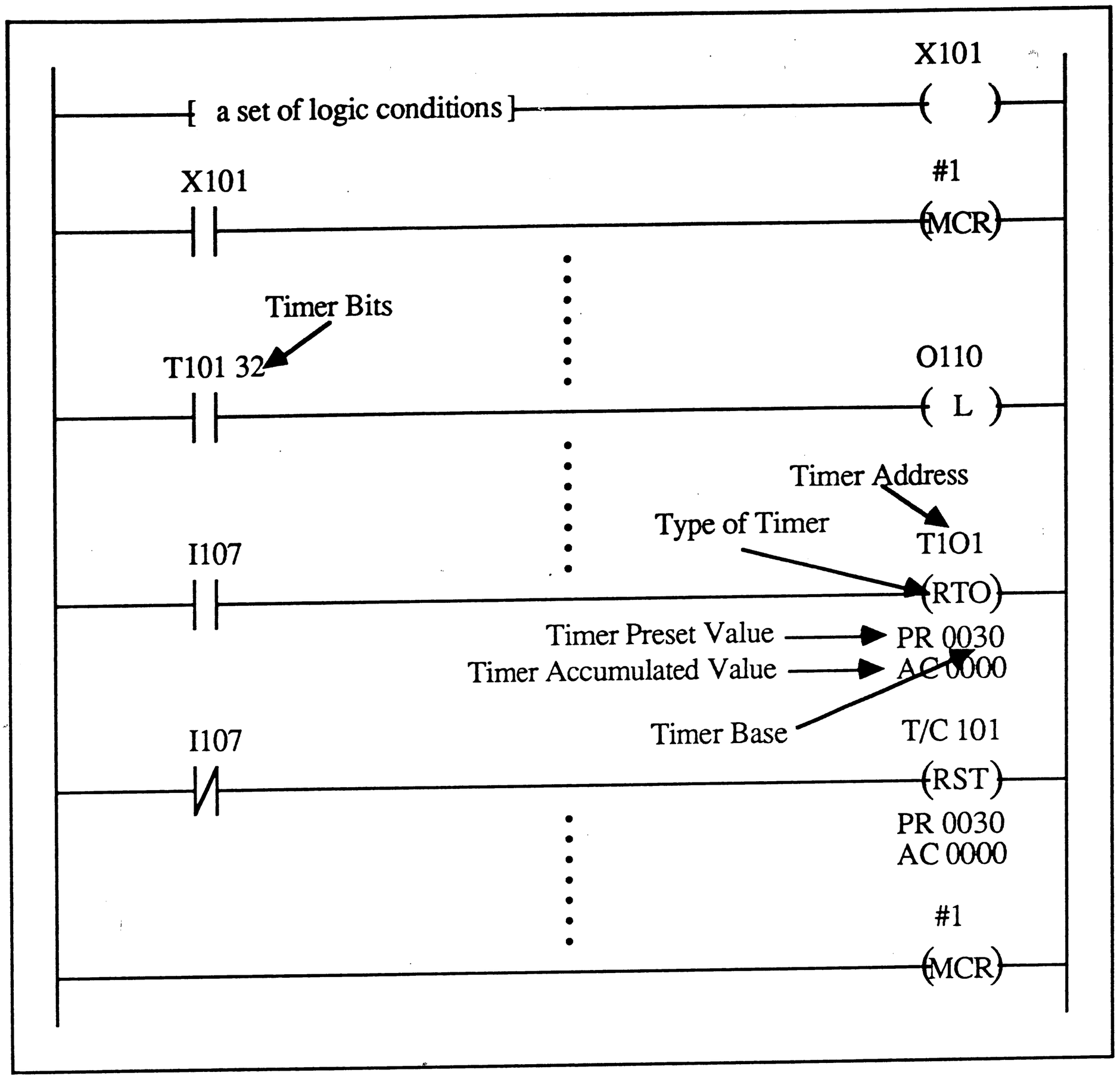


Figure 3.3 Retentive ON-DELAY programmed timer.

The timer is assigned an address such as T101 to be identified as a timer. Also included as part of the timer instruction is the time base, the timer preset value and the timer accumulated value. Programmers can specify the time interval by using a time base. The time base for Allen Bradley PLC-4's is always equal to one tenth of a second. Thus, if the time interval is 3 seconds, the value of time base would be entered 30. The timer preset (PR) contains the length of time that is counted before an output action is

taken. The accumulated value (AC) is the timer's own "stopwatch" that stops timing when it equals the preset value.

In addition to a timer as an output, the state of the timer bit can be addressed in the program. A programmer might want to examine a timer bit to see if that timer is timing or to see if it has finished timing. To address a timer bit, programmers must input a four digit address for a timer and add a two digit timer bit number. For example, T101 32 in Figure 3.3, addresses a particular status bit belonging to timer T101 and indicates that with the retentive timer-on instruction, this bit is set only when the accumulated value is equal to or greater than the preset value. There are four timer bits that programmer may want to examine:

1. ***Timer clock bit (bit#15)***. This bit represents the time base for timers. As the timer times, this bit is set and reset each tenth of a second.
2. ***Enable bit (bit#16)***. This bit is set when the rung controlling the timer is true and reset when it is false.
3. ***Timer timing bit (bit#31)***. This bit is set when the timer is timing and reset when it has finished timing. This bit is only used with the Retentive Timer-on instruction which will be described later in this section.
4. ***Done bit (bit#32)***. When this bit is set and reset depends upon which timer instructions are being used:

- With the Retentive Timer-on instruction, this bit is set only when the accumulated value is equal to or greater than the preset value.
- With the Timer-off Delay instruction, this bit is reset only when the accumulated value is equal to or greater than the preset value.

One type of timer used for the low level commands is called the *Retentive Timer-on* instruction which is represented with the symbol -(RTO)-. It begins timing when the rung path has logic continuity. Unlike the other types of timers, the RTO is retentive and will hold its accumulated value when the timer rung goes FALSE and will continue timing where it left off when the timer rung goes TRUE again. This timer must be accompanied by a timer RESET instruction to reset the accumulated value of the timer. The reset instruction represented with the symbol -(RST)-, is the only automatic means of resetting the accumulated value of a retentive timer. The RST instruction must be addressed to the same word as the RTO instruction. If any RST rung path has logic continuity, the accumulated value of the referenced timer is reset to zero.

As referred to in Figure 3.3, when the limit switch I107 at station 2 is set on, the retentive timer-on instruction T101 begins timing for the preset time of three seconds. While the timer is timing, the timer bit instruction T101 32 is reset, that is, the rung is false. When the timer finishes timing, this timer bit is set. In other words, the rung goes true and the motor O110 starts running. As soon as the cart leaves station 2, the limit switch I107 is turned off and the normally closed I107 in RST rung is set on. When the RST rung is true, the accumulated value of the timer T101 is reset to zero.

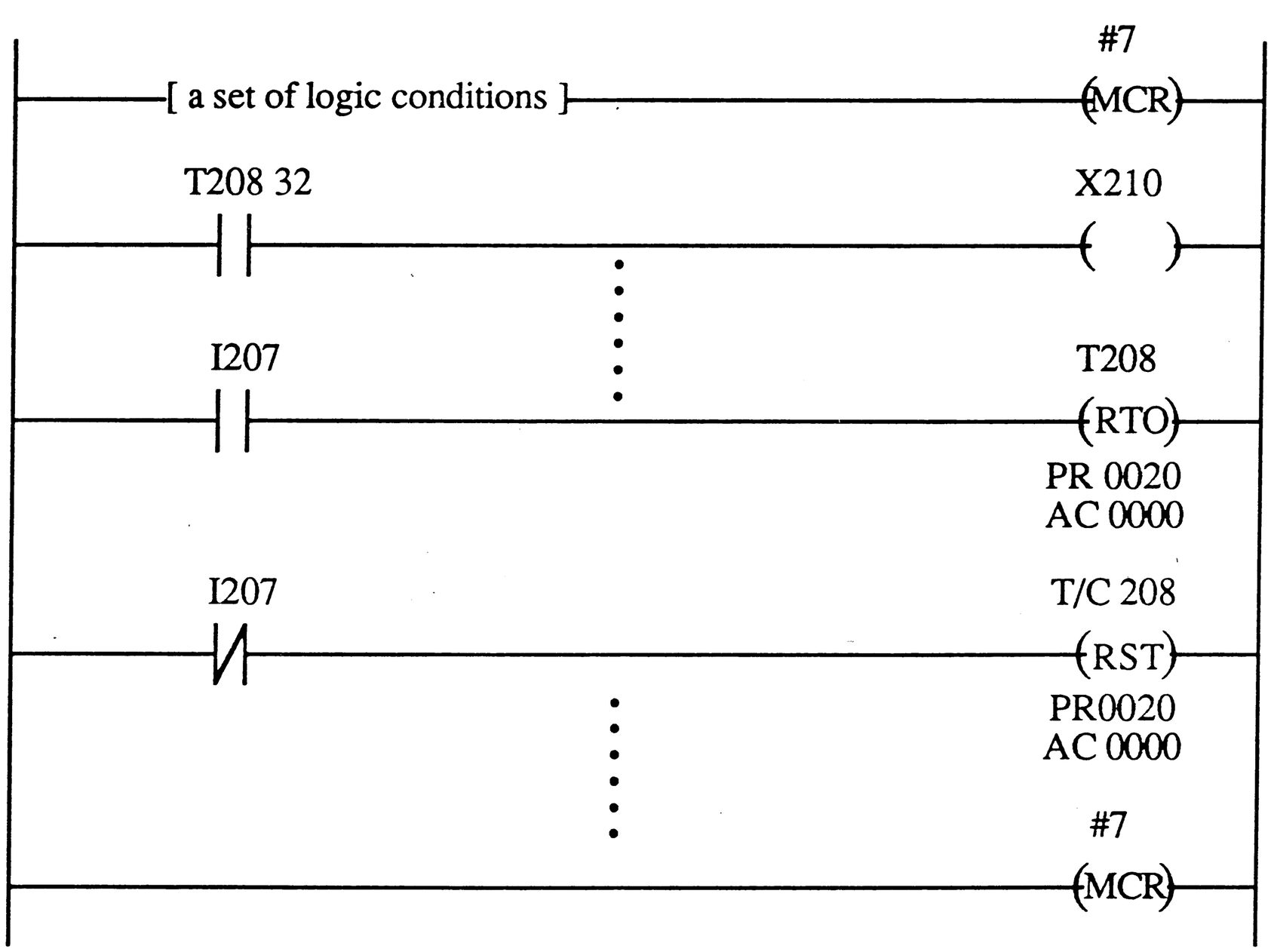
3.3.6 Addressing Flag Bits

When the low level commands are programmed in one PLC, they can use the status of contacts from other PLC's. Therefore, a rung in PLC "A" can check the status of a bit from PLC "B" in order to control one of PLC "A" outputs. Such a bit used in the low level commands is called a *flag bit*. As referred to in Table 2.5 in section 2.4.3, an example of a flag address is X210 where X stands for a flag, 2 stands for the PLC's

number, and 10 is the flag number.

Through the using of the flag bits, the programs in one PLC can indirectly activate outputs from other PLCs. Figure 3.4 shows how the status of a flag in PLC #2 is used to govern an output operation in PLC #3.

(a) A low level command in PLC #2.
Command 20: move cart from station 11A to station 10.



(b) Rung in PLC #3.

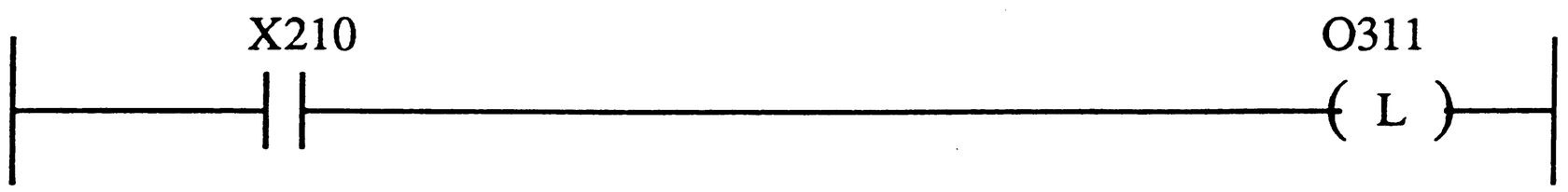


Figure 3.4 The status of flag bit.

As illustrated in Figure 3.4(a), when the limit switch I207 at station 11 is set on, the retentive timer-on instruction T208 begins timing for two seconds. After the timer finishes timing, the rung of timer bit T208 32 is true and then flag X210 is set. As soon as the flag in PLC #2 is in the ON condition, the flag rung in PLC #3 (Figure 3.4(b)) goes true and the motor O311 starts running to drive a cart from station 11 to station 10.

3.3.7 Determing When the Command is Finished

As illustrated in Figure 3.1, the X101 instruction on Rung 2 sends a signal back to the computer and indicates to the high level command when this low level command is completed. As stated in section 3.3.3, the only way to turn off this low level command is to activate the internal control relay S107 in the MCR zone. This action will cause the X101 instruction to be set off and stop the execution of the low level command. If the X101 instruction is in the "OFF" condition, the computer initiates the next low level command to move the cart, otherwise, it checks the status of the next cart and returns to check X101 later. This technique will be discussed in more detail in section 4.3.2.

Chapter 4 High Level Commands

4.1 High Level Languages

4.1.1 High Level Language Features Required

High level languages have been supplanting assembly language for many years in computer control systems. Many sources have pointed out the advantages of the high level language approach: less time required for programming, code that is more readable and understandable, and improved documentation of programs. The use of optimizing compilers and well-written interpreters have made high-level languages nearly as efficient as assembly language in the use of memory and processing time (Lukas, 1986).

However, the designer or evaluator of a high level language should be aware that a general-purpose language such as FORTRAN, PASCAL and C cannot be used in a real-time process control environment without a significant amount of modification. The user needs to evaluate the various language options carefully to make sure that the selected one meets the needs of the application. The major functional areas to be considered in such an evaluation include:

Interfacing with the Process. Most of the popular high level languages are designed for a data processing environment in which information is read from or written to data files in memory or to various computer peripherals (e.g., disks, CRTs, magnetic tape units, or printers). However, in a process control environment, information transfer involves continuous devices (such as thermocouples, differential pressure transmitters, or valve

positioners) and sequential devices (such as limit switches or motor starting relays). Ideally, the user of the high level language should be able to write input and output statements that manage this flow of information with minimal human intervention.

Coping with the Real-Time Environment. The second area of functional requirements for the high level language deals with the fact that the language has to operate in a real-time rather than an off-line or batch-processing environment. To be suitable for operation in the real-time environment, the high-level language therefore must have the following features:

1. *Interrupt-handling capability to allow programs to respond to external events.*
2. *Ability to schedule multiple application programs or tasks executing "simultaneously" (multi-tasking).* The scheduling of the tasks must be well-defined under all operating conditions and take into account different priorities of tasks and their triggering by external interrupts, elapsed time, or real time. The tasks must be able to share data with one another and synchronize their operation.
3. *Ability to manage resources that must be shared by different tasks.*

Interfacing with other Elements in a Distributed Control System. The third functional area to be considered when evaluating or selecting a high level language is its facility for communicating with devices within the distributed control system other than the sensors and actuators discussed earlier.

Providing Security Safeguards. The fourth requirement for a functional language is that it contains security safeguards to ensure that failures and errors in the system do not

cause an unsafe condition. Security is not only a language issue but it also concerns the design of the hardware and operating system as well.

4.1.2 Reasons for Using C Language

Until the early 1980s, high level programming languages were restricted primarily to larger mini-computer systems used in direct digital control, supervisory control, or data acquisition applications. In industrial control, they were not seriously considered for use in dedicated microprocessor-based systems.

However, this situation has changed significantly in the last several years. Through the increasingly widespread use of personal computers, high level commands can be implemented using a high level language, such as C, to control the sequence of the low level commands in the PLCs. Among the large variety of high level languages, C was chosen to control the Cartrac system for the following reasons (Lafore, 1988):

1. Most operations that can be performed on the personal computer in assembly language can be accomplished in C.
2. C is a well-structured language. Its syntax makes it easy to write programs that are modular and therefore easy to understand and maintain.
3. The better C compilers can now generate code that is efficient. Compared with C, it is often difficult to produce significant speed increases by rewriting in assembly language.
4. The C language includes many features specifically designed to help in the creation of large or complex programs.

5. C is more *portable* than most languages. That is, a C program can be easily converted to run on different computers.

4.2 Theory of Grouping Low Level Commands

4.2.1 Developing Software Control

In contrast with the fundamentals of the low level commands described previously, the high level commands use a high level language to monitor and control the operation of the process through the low level commands in the Cartrac system. While the hardware of the low level commands resembles conventional panelboard instrumentation, the hardware of the high level commands uses CRTs or similar advanced display technology in console configurations often called *video display units* (VDUs). The high level commands accept operator inputs through keyboards instead of the switches, push-buttons, and potentiometers characteristic of conventional operator interface panels (Lukas, 1986).

As stated before, one characteristic of a low level command is its independence from the operations of other low level commands. In order to control the desired route of a cart in the Cartrac system, the high level command must provide the proper input signals to the low level commands which are going to control the movement of this cart. These output signals from the high level command are converted into the input instructions of the low level commands through an interface device bridged between the microcomputer and the PLCs. An example of this conversion is illustrated in Figure 4.1.

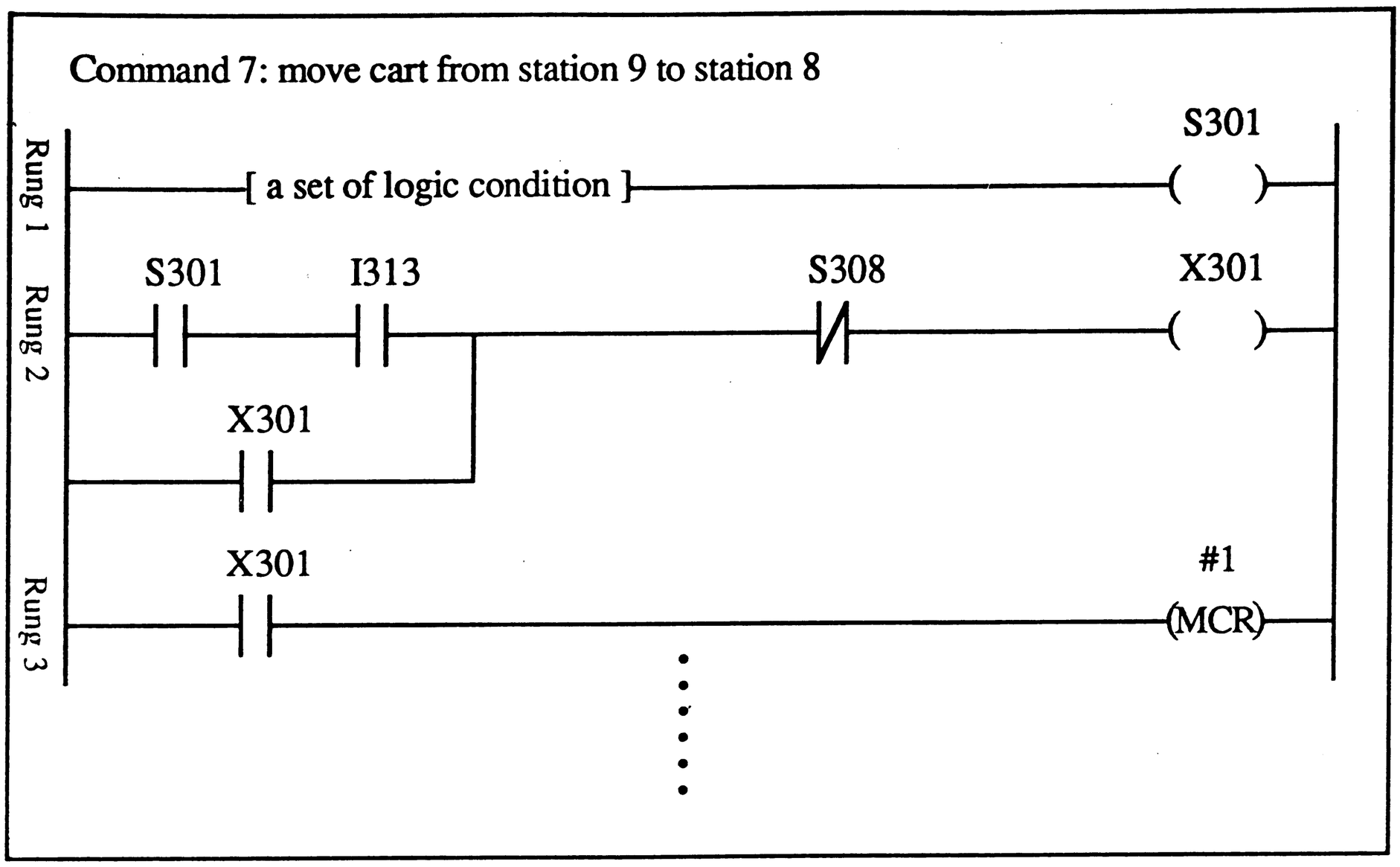


Figure 4.1 The input for software control

In order for the high level command to start a low level command, such as command 7, the following must occur:

The input instruction I313 in Rung 2, contact closed by a high level command, is one of a set of logic conditions required to start command 7. Its function is to receive an input signal from a high level command through an interface device. Besides a set of logic conditions in Rung 1 which must be TRUE, this conditional input I313 must be set on in terms of the right signal derived from the high level command.

4.2.2 Logic Sequence of the System

By using the above technique, the desired low level commands can be sequenced within each of the high level commands defined by the programmers. In the current

control strategy, the following seven high level commands have been defined:

1. Command 1: move cart from AS/RS to K&T Milling Center.
2. Command 2: move cart from AS/RS to Queueing Station.
3. Command 3: move cart from Queueing station to K&T Milling Center.
4. Command 4: move cart from Queueing station to AS/RS.
5. Command 5: move cart from K&T Milling Center to AS/RS.
6. Command 6: move cart from K&T Milling Center to Queueing Station.
7. Command 7: loop cart in the system and wait for commands.

Each high level command consists of a set of low level commands sequenced in a specific order. For example, when a cart moves from the AS/RS to the K&T milling center (command 1), the following steps of low level commands must be grouped sequentially according to the route of the cart:

1. Transfer a tote from station 1 to a cart at station 2. This low level command is currently not one of the 21 low level commands described earlier and is controlled by PLC #4.
2. Command 1: move cart from station 2 to station 3.
3. Command 2: move cart from station 3 to station 9.
4. Command 7: move cart from station 9 to station 8.
5. Command 8: move cart from station 8 to station 7.
6. If a part is being processed at the K&T milling center, the sequence continues with step 7, otherwise goes to step 12.

7. Command 9: move cart from station 7 to station 6.

8. Command 12: move cart from station 6 to station 5.

9. Command 5: move cart from station 5 to station 4.

10. Command 6: move cart from station 4 to station 3.

11. Go back to step 3: command 2.

12. Command 10: move cart from station 7 to station 10.

13. Command 14: move cart from station 10 to station 11A.

14. Command 15: move cart from station 11A to station 13.

15. Transfer this tote from station 13 to station 14. This low level command is not currently one of the 21 low level commands and the logic sequence of this command is controlled by a GE Series One Plus PLC.

16. Push the tote at station 14 forward to the K&T milling center. Its sequence is controlled by a GE Series One Plus PLC.

17. Notify the high level command that command 1 is finished and stop execution.

The programming technique for the other high level commands is similar to that of the high level command discussed above. The technique of grouping and sequencing the low level commands uses a *dynamic data structure* which will be explained in the following section. Figure 4.2 shows the flow chart of a cart which is moving from the AS/RS to the K&T milling center.

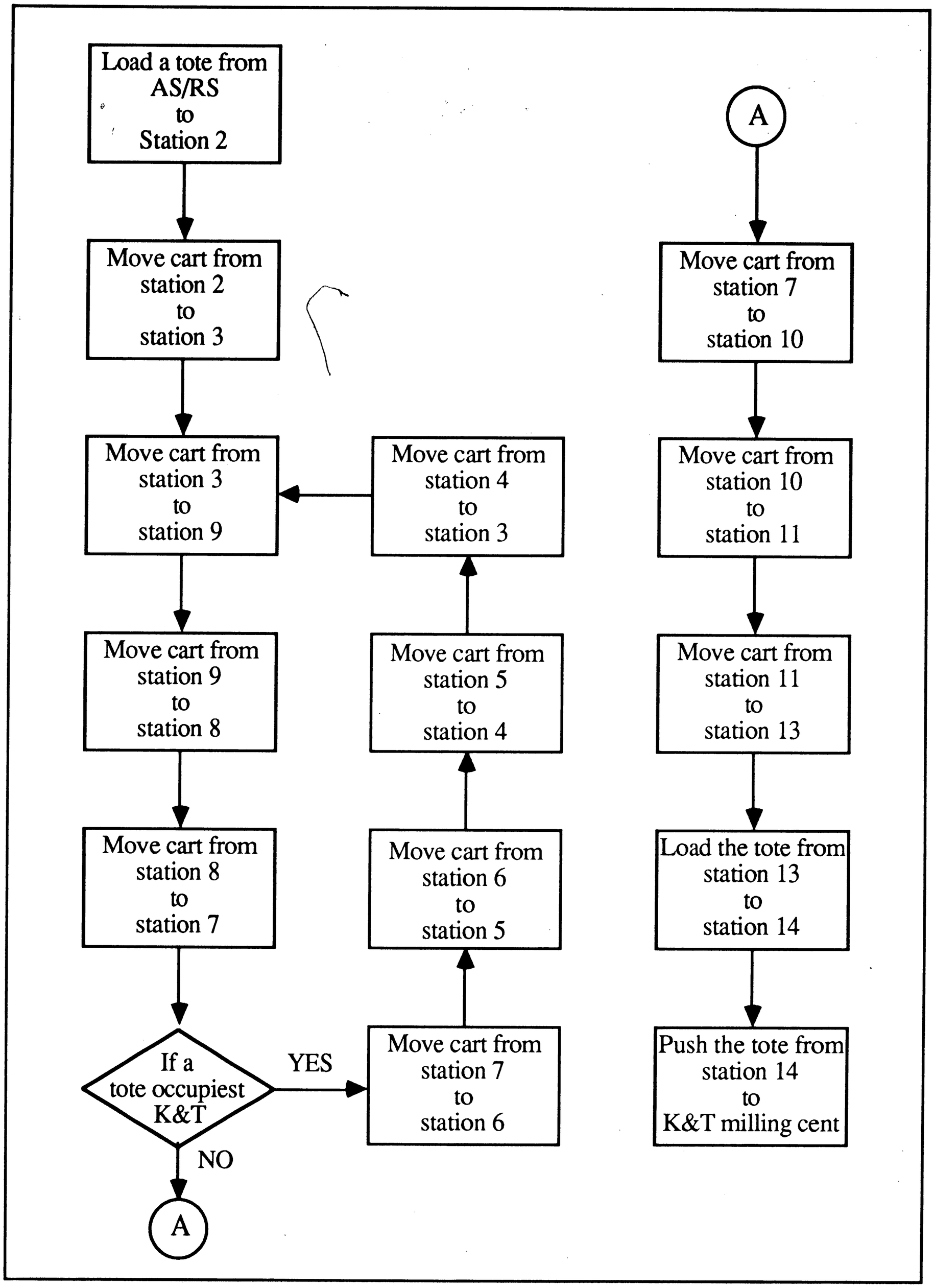


Figure 4.2 The flow chart of a high level command.

4.3 Implementation of High Level Commands

4.3.1 Structure of Program

The program of high level commands consists of *algorithms* and *dynamic data structures*. The linked list, one type of data structure, is used for constructing the high level commands. Unlike an array, a linked list may not access its storage in a random fashion because each piece of information carries with it a link to the next data item in the chain. A linked list requires a complex data structure. Its retrieval operation does not remove and destroy an item from the list. Linked lists are used for the following two main purposes (Schildt, 1987):

1. The first purpose is to ***create arrays of unknown size in memory***. If the actual size of a list is unknown, a linked list must be used. By using this technique for the Cartrac system, the operator can input data dynamically without considering the limitation of entering the number of carts in advance.

2. The second purpose is for ***disk-file storage of databases***. The linked list allows an operator to insert and delete items quickly and easily without rearranging the entire disk file.

For these reasons, linked lists are used extensively in database management. Linked lists can be either singly linked or doubly linked. Doubly linked lists consist of data as well as links to both the next item and the preceding item in the list. A linked list consists of structures related to one another by pointers, rather than by being members of an array. This basic idea is that each structure on the list contains a pointer that

points to the previous or next structure. The pointer in the first or last structure on the list does not point to anything, so it is given a value of 0 or null. Figure 4.3 shows how these links are arranged for the high level commands and the low level commands.

4.3.2 Elements of The Program

As shown in Figure 4.3, this high level program has declared two types of data structures. One data structure which assigns a high level command to each cart is named "cart structure." Another one which links the stations and starts the low level commands is named "station structure." The data structure for each element on the linked list "cart structure" (the top ones in Figure 4.3) is defined as follows:

1. High_level: It indicates which high level command has been assigned to he cart is defined by the operators.
2. Cart_priority: The priority of carts in the system. It will be discussed in the next section.
3. Cart_number: The high level commands assign each cart a number automatically in order to distinguish one cart from others.
4. First_station: The starting station of the cart. It is defined by the operator in terms of the location of the cart in the system.
5. Previous_cart: This is a pointer that points to the previous cart structure. The pointer in the first structure (cart 1) on the list does not point to anything, so it is given a value of 0 or null.
6. Next_cart: Similar to Previous_cart, this pointer points to the next cart structure. The pointer in the last structure on the list does not point to anything, so it is given a

value of 0 or null.

7. First_command: This is a pointer that points to a different data structure that determines what low level commands will be executed and in what order they will be executed. This structure will be explained in more detail later. This pointer, indicates the first station structure or first low level command, to be executed. This pointer is generally not moved.
8. Current_command: This pointer points to the station structure at which the cart is situated currently. Because this pointer is movable, it will point to the next station structure as soon as the cart proceeds to the next station in the system.
9. Last_command: This pointer indicates the last station structure or last low level command to be executed. This pointer is generally not moved.

The data structure for each element on the linked list of station structure (the smaller block in Figure 4.3) is defined as follows:

1. Low_level: As mentioned in the section 4.2.1, this is an input signal to start a low level command in the PLC.
2. Front_station: It indicates the current station of the cart.
3. Rear_station: It shows the next station that the cart will proceed to.
4. Previous: This is a pointer that points to the previous station structure.
5. Next: This pointer indicates the next station structure.

One characteristic of the high level commands is to allow multiple carts to run simultaneously in the system. To do this, the program examines the current status of cart

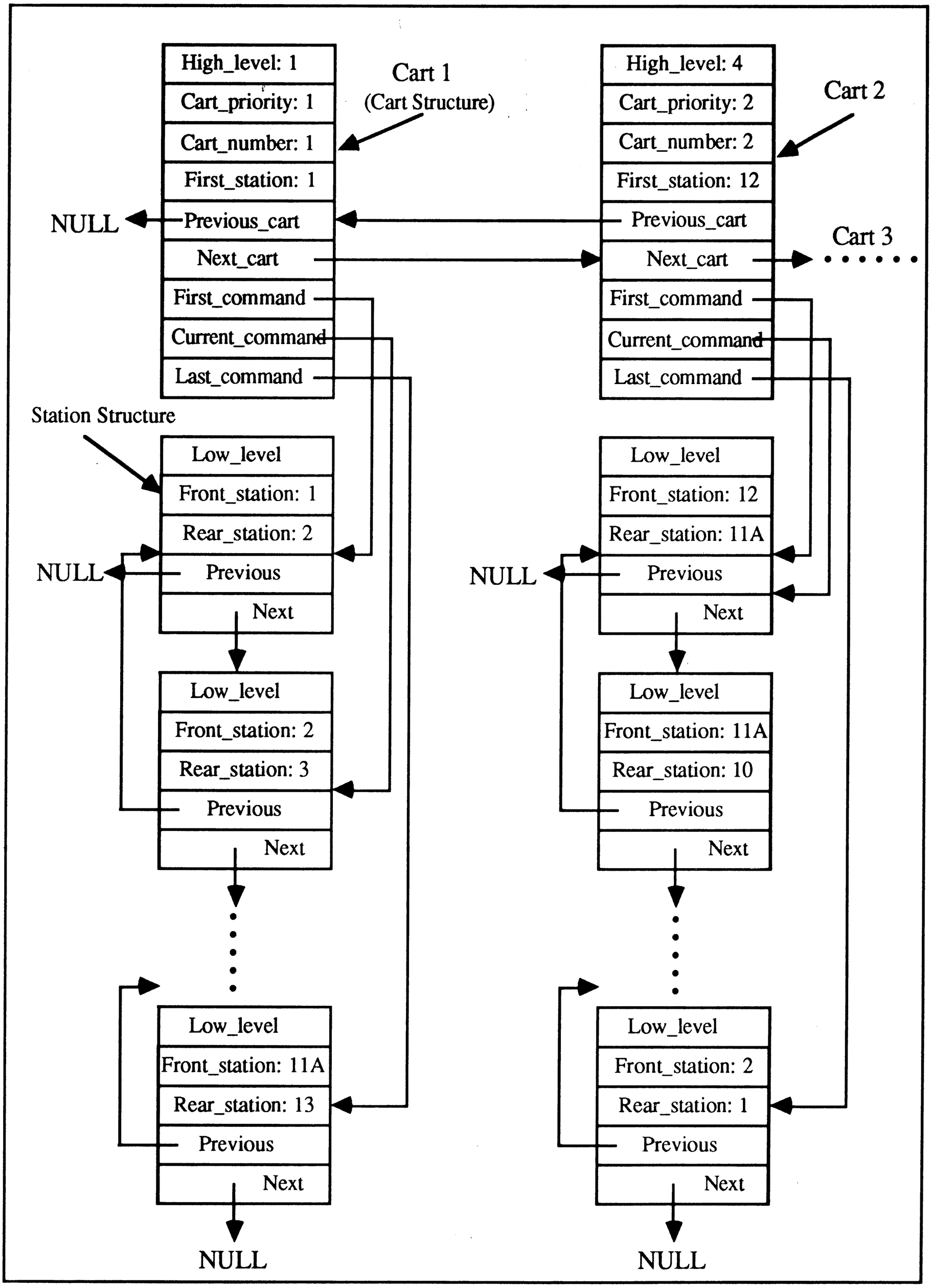


Figure 4.3 A doubly linked list of high level commands

1. If cart 1 is not executing a high level command or the operation of the first station structure or first low level command is not completed, the program will continue to scan the status of the next cart structure (cart 2) through the pointer of Next_cart on the cart 1 structure. If a high level command is specified, the program checks the status of the current low level command for cart 1 through the pointer, Next, in the station structure. If the low level command is finished, the next low level command is executed. Otherwise, the cart 2 structure is scanned. Because the operation of the computer is very fast, it can control the movement of the carts at the same time even though the program scans the linked lists of the cart and the station structures sequentially rather than simultaneously.

An example of a high level command in Figure 4.4 will show how to tie this example to the flow chart of a cart in Figure 4.2. This example is the linked list of a high level command with the same structure as shown in Figure 4.3. It controls the movement of a cart from the AS/RS to the K&T milling center.

First, the pointer of Current_command points to the station structure #1 to indicate the original status of the cart before the execution of this command. This pointer will point to station structure #2 while the cart is moving from station 2 to station 3. The program also checks the number of Front_station. When the current number of Front_station is equal to 8 in structure #5, the program will examine if a part is being processed at the K&T milling center. If a cart is on the K&T milling center, the pointer pointing to structure #5 will change to the next structure and move the cart from station 7 to station 6. The pointer will advance through the next structures until it reaches structure #9 which then points back to structure #3. If there is no part on the K&T milling center, the pointer pointing to structure #5 changes directly to structure #10 to

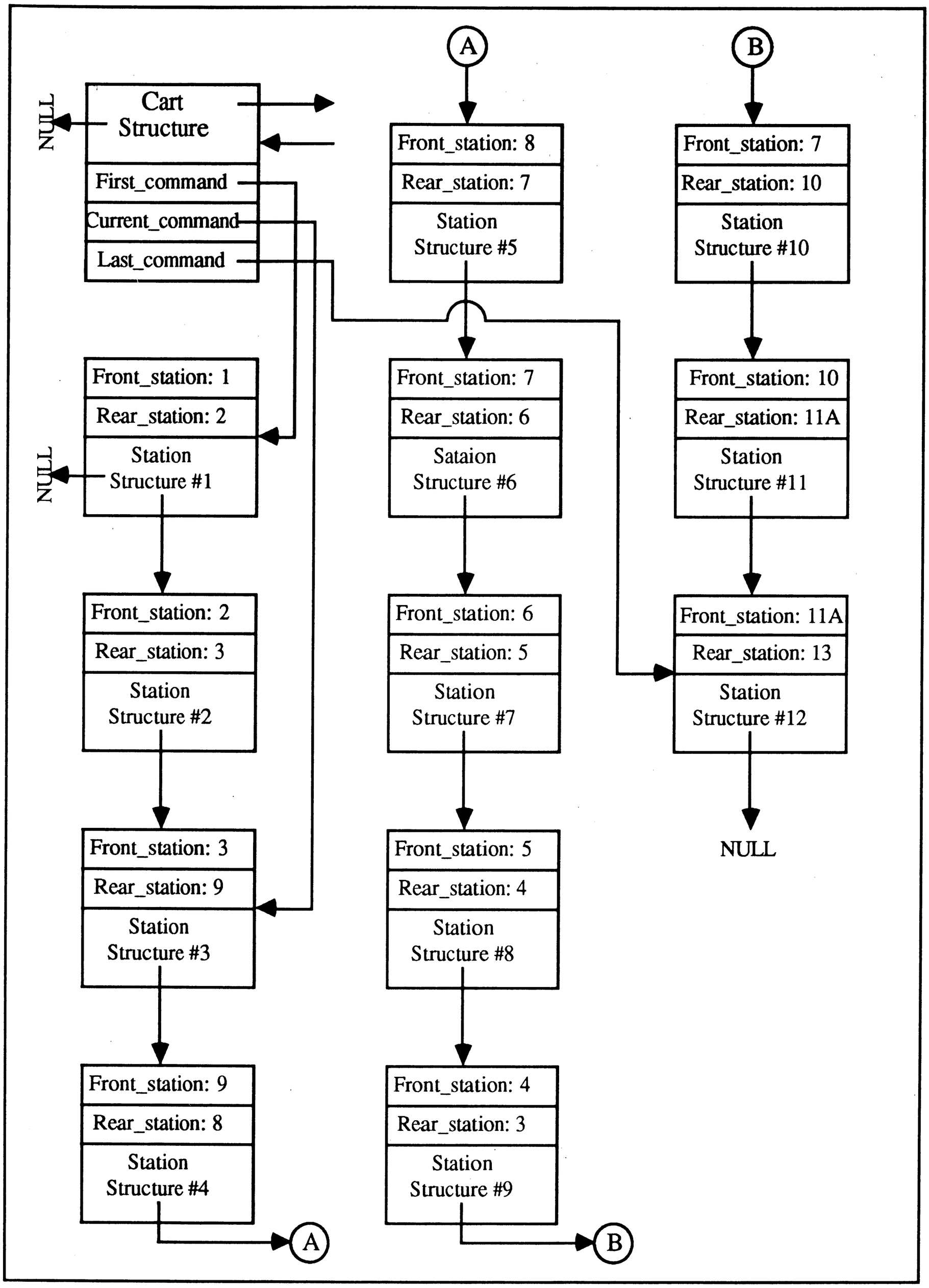


Figure 4.4 The linked list of a high level command

activate the movement of the cart from station 7 to station 10 and stops the execution of this high level command after the cart arrives at the K&T milling center.

4.3.3 Options of the Cartrac System

After the implementation of the high level commands, a menu driven program was developed to initialize and control the cartrac system. The main menu has the following options:

1. Add a cart.
2. Delete a cart or all carts.
3. Modify a cart.
4. Check the carts.
5. Execute the Cartrac system.
6. Stop and exit the system.

Before running the system, the operator must establish the linked lists in the computer by defining the high level commands. By choosing the first option - "Add a cart", a new element is added to the linked list. Furthermore, a high level command as well as the current location must be assigned to each cart. For example, if a cart is situated at station 2, the following three high level commands (functions) are provided as choices:

1. Move cart from AS/RS to K&T milling center.
2. Move cart from AS/RS to Queueing station.

3. Move cart from AS/RS to loop and wait for commands.

4. No command is defined now.

The monitor only shows the high level commands associated with this station instead of all of seven high level commands. If function 1 or function 2 is chosen, the next question asked is:

"Does this cart have the first priority (y/n) ?"

When a cart is defined with first priority, it can arrive at its destination sooner than other carts with a lower priority. The purpose of the first priority is to move a cart over the track with the least interruption. Otherwise, the cart is given the second priority and may loop in the system if other carts have the first priority and are attempting to go to the same destination. If function 3 (move cart from AS/RS to loop and wait for commands) is selected, the cart is assigned the third priority automatically, in other words, it has the least priority. This cart loops in the system and waits for a command to go back to the AS/RS.

When the operator defines the current status of carts, he can remove a cart or carts by choosing the second option of the main menu. By choosing the third option, he can modify the status of a cart. The fourth option can list the current status of all carts in the Cartrac system so that carts can be added, deleted or modified correctly. After defining each cart, the operator chooses the fifth option of the main menu to activate the system. While the system is running, the computer shows the current movement of carts and informs the operators when a command has completed for a cart. The system then continues to execute the other high level commands while waiting for a new command for this cart which does not have a high level command currently. By pushing an "M"

key on the keyboard, the operator is able to interrupt the system and then change the desired commands. The "M" key can also be used to normally stop the movement of carts and return to the main menu.

Chapter 5 Summary

5.1 Summary of Current Work

Upon integration of the high level and the low level commands, the Cartrac system can be automatically controlled by a computer without human intervention. So far, the low level commands have been implemented by using the ladder logic language and saved in the PLCs. The purpose of using low level commands is to divide the original, very large ladder logic programs into many smaller, independent ladder logic programs. In the Cartrac system, 21 low level commands have been defined to move carts between all the stations. By using master control reset (MCR) instructions in the programs, the low level commands are implemented as independent ladder logic programs. Furthermore, through the use of flag bits, the programs in one PLC can indirectly activate outputs from other PLCs.

In contrast to the low level commands, the high level commands are implemented using a high level language, such as C, to control the sequence of the low level commands in the PLCs. In the current control strategy, seven high level commands have been defined. By using the principle of dynamic data structures, such as the doubly linked lists, high level commands have the ability to sequence the low level commands and control multiple carts simultaneously in the system to perform a task.

5.2 Contributions of Project

Although there are still limitations in the current Cartrac system, at the completion of the low level and the high level commands, the Cartrac system will achieve the following benefits:

1. In contrast to the feature of the original large programs, the revised low level commands are easier to read and more flexible to modify without influencing the logic sequence of other low level commands.
2. All of the motors in the system except those at the station 5 and station 9 can stop automatically when they are not in use.
3. A cart can be placed on any station before the system starts up without a system fault.
4. The queueing station (station 12) logic has been implemented.
5. The logic and sequence of the high level commands can be easily modified by linking the low level commands in different sequences.
6. The sequence of the cart movement is determined according to the priority of carts in the system.
7. The high level commands will eliminate the need for the manual control stations to route the pallets. A computer (IBM 7552) will play the central role, instead of the PLCs, to control the movement of the carts.
8. By using a high level language, the Cartrac system will be easily interfaced to the flexible manufacturing cell in the future.

5.3 Directions for Future Works

At the completion of this project, the low level commands have solved the limitations existing in the Cartrac system. The high level commands, using the principle of the dynamic data structure, have a capacity to group the low level commands and manage the sequence of carts without human intervention. However, the integration of the low level and the high level commands is still infeasible because an interface device and software between the PLCs and the computer (IBM 7552) have not yet been installed. As referred to in Figure 1.4, only the AS/RS is currently under control of the IBM 7552 cell controller. Future research will provide the framework to allow the Cartrac system, currently controlled by the PLCs, and the K&T milling center, controlled by the GE Mark Century 2000 CNC controller, to be appended to this cell controller.

In the near future, a CNC turning center will be added to the cell. Therefore, the enhancement of the high level logic, some additional low level commands interfacing the Cartrac with the turning center, a transfer device between these two machines, and the integration between the cell controller and the turning center are required. Once these items are accomplished, the Cartrac system will become an automated material handling system for the flexible manufacturing cell and the cell will be an application for *computer integrated manufacturing* (CIM).

GLOSSARY

AGVS(automated guided vehicle system) - a material handling system that uses independently operated, self-propelled vehicles that are guided along defined pathways in the floor.

AMH(automated material handling) - comprising the systems for material storage, transport, and delivery within a manufacturing facility. It includes both the newer computer control systems as well as the more conventional equipment.

AS/RS(automated storage/retrieval system) - a combination of equipment and controls which handles, stores, and retrieves material with precision, accuracy, and speed under a defined degree of automation.

cart - a device which carries a tote is driven by a rotary tube that runs between the two rails.

cartrac - a conveyor system uses individual carts riding on a two-railed track contained in a frame that places the track a few feet above floor level.

cell - a manufacturing unit consisting of two or more workstations with the material handling systems, storage buffers, and other items necessary to connect them.

CIM(computer integrated manufacturing) - a business strategy for the efficient and integrative use of computers to expand and perform the intellectual work in the manufacturing system.

CNC(computer numerical control) - a technique in which a machine tool control uses a computer to store numerical control instructions generated earlier by CAD/CAM for controlling the machine.

controller - a computer, or group of computers, used to control a machine tool, robot, or similar device or group of devices.

DNC(direct numerical control) - the use of a shared computer to distribute part program data by way of data lines to remote machine tools.

FMS(flexible manufacturing system) - a group of NC machine tools that can randomly process a group of parts having different process sequences and process cycles using automated material handling and central computer control to dynamically balance resource utilization so that the system can adapt automatically to change in part production mixes and levels of outputs.

K&T(Kearney & Trecker) milling center - a very large, complex NC machining center which has been modified with a GE Mark Century 2000 computer numerical control unit.

ladder logic language - a symbolic set of instructions is used to create the controller program in the PLCs.

LAN(local area network) - a nonpublic communications system that permits the various devices connected to the network to communicate with each other over distances from several feet to several miles.

limit switch - a device designed to operate only when a predetermined limit is reached, and it is usually actuated by contact with an object such as a cart or a shuttle.

MAP(manufacturing automation protocol) - a number of organizations have contributed to the development of a communications protocol designed for use in a factory local area network.

MCR(master control reset) - an instruction can be programmed to control an entire circuit or to control only selected rungs of a circuit in the PLCs.

NC(numerical control) - a technique for the control of machine tools or factory processes in which information on the desired actions of the system is prerecorded in numerical form and used to control the operation automatically.

PC(personal computer) - an electronic machine which can perform mathematical and logical calculations and data processing functions in accordance with a predetermined program of instructions.

PID(Proportional-Integral-Derivative) - control methods for continuous systems.

PLC(programmable logic controller) - a digitally operated electronic apparatus which uses a programmable memory for the internal storage of instructions for implementing specific functions such as logic, sequencing, timing, counting, and arithmetic to control, through digital or analog input/output modules, various types of machines or processes.

shuttle - an auxiliary transportation device which transports a cart between two sections of drive tubes such that it carries a cart from the head of one station to the end of another station.

solenoid - an electromagnetic actuator that can be used to open and close a valve, electrical contact, or other mechanical device.

tote - a device which carries workparts as they are transferred between stations through the sequence of operations.

WIP(work-in-process) - the amount of product currently located in the factory that is either being processed or is between processing operation.

BIBLIOGRAPHY

Bulletin 1773, PLC-4 Microtrol Programmable Controller, Product Guide, Allen-Bradley Co., 1983.

Castle, Dennis H., "Allis-Chalmers Corporation's Experience with FMS," Presented at the SME Flexible Manufacturing Systems Seminar, Dearborn, Michigan, April, 1982.

Charalambous, C., and Conning, A.J., "Distributed Sequence Control Utilising a High Level Sequencing Language in Conjunction with PLCs," Software for Computer Control, 1988, pp. 27-35.

Groover, Mikell P., Automation Production Systems and Computer Integrated Manufacturing, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1987.

Johnson, David G., Programmable Controllers for Factory Automation, Marcel Dekker, Inc., New York, N.Y., 1987.

Lafore, Robert., Turbo C Programming for the PC, Revised Edition, Howard W. Sams & Company, Inc., Indianopolis, Ind., 1988.

Lukas, Michael P., Distributed Control Systems, Van Nostrand Reinhold Company, Inc., New York, N.Y., 1986.

Petruzella, Frank D., Programmable Logic Controllers, McGraw-Hill, Inc., Berkeley, CA., 1989.

Pierson, Robert A., "Trends in Material Handling," Automation, March, 1988, pp. 10-16.

Schildt, Herbert., Advanced TURBO C, McGraw-Hill, Inc., Berkeley, CA., 1987.

Serzeniewski, M., "Design and Application of Programmable Flexible Transportation Systems," Proc. 4th European Conf., Automated Manufacturing, May, 1987, pp. 151-162.

Taylor, I.K., and Ford, M.J., "The Relationship Between Numerical Control Systems, Programmable Controllers and Overall Computer Control in the Development of a Cell," Proc. 4th European Conf., Automated Manufacturing, May, 1987, pp. 79-88.

VITA

Mr. Hung-Ren Lyou was born in Taiwan, Republic of China in 1963. He received a Bachelor of Science degree in Industrial Engineering from Chung-Yuan Christian University in 1985. After his discharge from the military service in 1987, he served with the Quality Control Department in Hoover Electrons Co., Ltd. until 1988. At this time he entered the graduate program of Industrial Engineering at Lehigh University. Now he is applying for a job in his country and is planning to return home after July in 1990.