1995

Software design of a cell controller for a flexible manufacturing system

Diane F. Livingston
Lehigh University

Livingston,
Diane F.

Software Design of a Cell Controller for a Flexible Manufacturing System

October 8, 1995

SOFTWARE DESIGN OF A CELL CONTROLLER FOR A FLEXIBLE MANUFACTURING SYSTEM

by

Diane F. Livingston

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Industrial Engineering

Lehigh University

1995

Certificate of Approval

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

9/26/95

Date

Dr. Gregory L. Tonkay,

Professor in Charge

Dr. Louis A. Martin-Vega

Chairman of the Department

Acknowledgements

This paper would not have been possible without the support and encouragement of my partner in life, Jeffrey. I would also like to thank my advisor, Dr. Gregory Tonkay, for all his help and advise.

TABLE OF CONTENTS

List of Tables

List of Figures

Abstract

Flexible manufacturing systems can dramatically improve productivity and reduce costs in a very diverse and changeable manufacturing environment. The cell controller is the heart of the FMS, integrating the various pieces of equipment into an automated system which works together to manufacture a product. As a result, the design of the cell controller is critical to the success of the FMS.

The research presented in this thesis describes the software design of a cell controller for a flexible manufacturing system. The cell controller integrates an AS/RS and cartrac system into a complete material handling system for the FMS with future intentions to integrate the machine tools. The control system follows the hierarchical control concept. In order to manufacture a product, the cell controller communicates directly with the supervisory controllers of the AS/RS and cartrac system, which in turn control their respective sub-systems.

This thesis also describes modifications to the supervisory controllers of the sub-systems and the technique used to control communications between the cell controller and supervisory controllers of the sub-systems. Along with software development and computer communications, information is necessary to make decisions and therefore, coordinate the manufacture of a product. The cell controller and supervisory controllers use various databases, which are described.

Chapter 1 Introduction

Because of fierce competition, US manufacturers are looking for innovative ways to increase sales and reduce costs, especially on the production floor. Customer demands for more product variety and quality continues to reduce the life cycle of products creating a business environment characterized by change. In fact, 50% of the sales of most products will occur within eighteen months after introduction into the marketplace (Levin et al., 1993). In order to remain competitive, manufacturers must implement production systems that meet two, often conflicting, objectives: (1) flexibility with respect to product variety and (2) productivity.

Flexible manufacturing systems (FMSs) meet this competitive challenge by providing the flexibility required for small batch manufacturing at the productivity levels achieved with large volume manufacturing. An FMS can be defined as "a system consisting of a group of processing stations (i.e., computer-controlled machines, vision systems, inspection stations) interconnected by means of an automated materials handling and storage system and controlled by an integrated computer system" (Levin et al., 1993). A flexible manufacturing cell (FMC) is a subset of an FMS and has a cellular layout. The computer controlling the FMS integrates the functions of the different system elements and therefore, determines its flexibility. As a result, the design of the controller, especially the software design, will determine the successfulness of the FMS.

This thesis describes the design of the software controlling a flexible manufacturing cell. The Manufacturing Technology (ManTech) Laboratory at Lehigh University was used to implement and test this software design. The ManTech Laboratory housed automated manufacturing equipment, which included a cartrac material handling system, automated storage and retrieval system (AS/RS), and several automated machines. The goal was to create a flexible manufacturing cell, first integrating the cartrac and the AS/RS together, and finally integrating the machine tools. The current layout of the cell is shown in Figure 1.1.

The storage and handling system of an FMS will dramatically effect its productivity, efficiency, and profitability. The material handling system must be flexible and versatile. The cell's material handling system in the ManTech Laboratory consisted of two components: the AS/RS and the cartrac system. The AS/RS provided permanent storage of materials and automatically retrieved items when instructed to do so. The cartrac system was used for work-in process (WIP), either moving it from one station to another in the cell or providing temporary storage. As part of the cartrac system, a transfer device was used to transfer items between the AS/RS and the cartrac.

Previously the AS/RS and cartrac system were controlled separately, each with its own set of controllers. Each had a supervisory computer providing an easy-to-use human interface to the system. The supervisory computer allowed the human user to control the operation of the system being used. The supervisory computer translated the wishes of

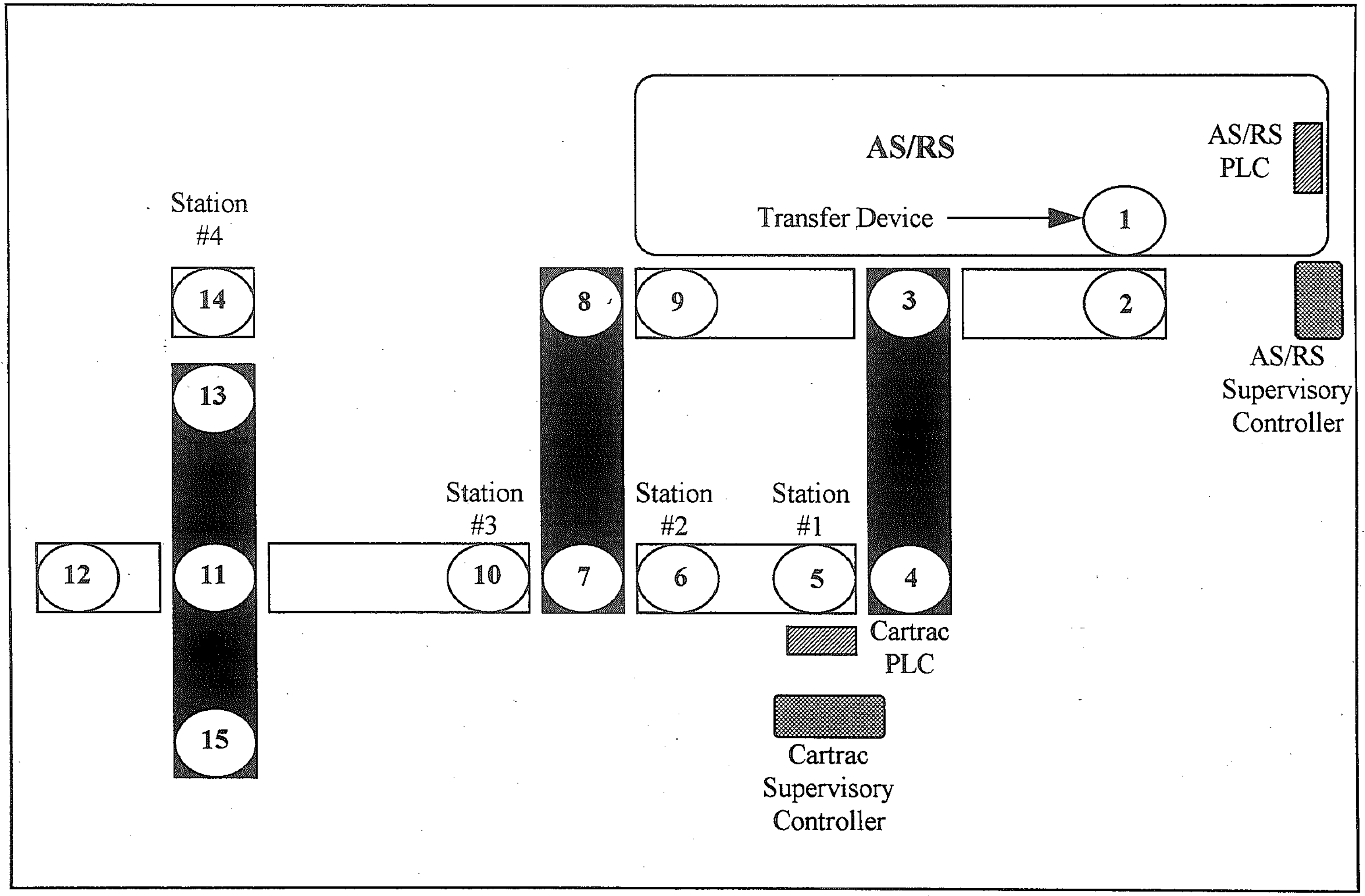


Figure 1.1 System Layout

the operator into lower level commands and sent those low level commands to a Programmable Logic Controller (PLC). The PLC used these commands to directly control and sequence the physical hardware. The cartrac system was controlled with it's own supervisory computer and PLC, while the AS/RS was controlled with it's own supervisory computer and PLC.

The previous material handling system controllers made its operation highly dependent on the actions of the human operator. For example, if an operator wished to send an item stored in the AS/RS to a station on the cartrac, he first used the AS/RS's controller to remove it from storage and position it on the transfer device. Once the item was on the transfer device, the operator walked over to the cartrac's controller and used it to transfer the item from the AS/RS to the cartrac and finally send it to the station. As evident from the above scenario, the previous material handling system was prone to human error because it was not fully automated. If, for example, an item was not correctly placed on the transfer device by the AS/RS, and the operator, not noticing the problem, proceeded to instruct the cartrac to transfer the item, the item might have fallen off the transfer device possibly damaging itself or the material handling equipment. Fully automating the system actually reduces its complexity to the human operator and dramatically reduces errors by minimizing the human interface.

The work presented in this thesis describes the cell controller that fully automates the material handling system by integrating the AS/RS and the cartrac systems. The cell controller will communicate via a local area network (LAN), using high level commands, to the supervisory controllers of the AS/RS and cartrac. Until actual machines are integrated into the cell, the cell controller simulates machine processing. Although most flexible manufacturing cells are used to machine parts, the proposed cell controller provides the flexibility to assemble or machine parts.

Several databases have been developed to support the flexible manufacturing cell. A part database contained information on parts, including the part number, bill of material, and routing. Information on processed parts was stored in a process database. In this database, processed parts were assigned a bar code, providing for the following information: part number, quantity, status (WIP, finished, or shipped), number of operations (from the routing) completed, and a list of the actual machines used in processing the part and the date processed on that machine. Finally, the AS/RS controller used an inventory database, which contained information on the parts stored in the AS/RS. This information included tote number, part number, quantity, coordinates of the tote location, bar code, part status (raw material, WIP, or finished part), and last date into the AS/RS.

The cell controller was designed with some constraints, mostly due to limitations of the equipment the controller will be integrating. Since the cartrac controller only provided the capability of operating a single cart on the cartrac at a time, the cell was limited to a single cart. Because parts were stored in totes in the AS/RS and to minimize the need for human intervention, each tote contained only one part, allowing the cell to manufacture only one part at a time. Since the ManTech Laboratory is a teaching laboratory as well as a research laboratory, the cell controller must allow the equipment in the cell to work in a stand-alone manner. Therefore, the AS/RS, cartrac, and machine tools can be used separately, independent of the flexible manufacturing cell. Finally, the cell controller was constrained to existing structures and commands.

Chapter 2 Literature Survey

2.1 Overview of Flexible Manufacturing Systems

In 1968, the Molins System 24 was implemented as the first flexible manufacturing system. Since then, the flexibility of FMSs has enabled manufacturers to produce products in small batches in a cost-effective way. FMSs are often designed to manufacture a family of related parts. They usually support machining operations, although they can be designed to produce assemblies. While all flexible manufacturing systems have many things in common, they might look differently due to the manufacturing environment the system resides in, magnitude of the mean time of events, and complexity of the events (Greenwood, 1986).

The major components of all FMSs include (Greenwood, 1986 and Groover, 1987):

1. ***Processing stations***. While these workstations usually consist of CNC machine tools performing machining operations, other types of workstations being introduced include assembly operations and inspection stations.
2. ***Material handling and storage***. Automated material handling equipment transports the work-in-process between processing stations.
3. ***Computer control system***. The coordination of the activities of the processing stations and material handling system is accomplished using computer control.

4. ***Communications system***. A network, usually a local area network (LAN), enables the passing of information between the various computers controlling the FMS.

5. ***Human labor***. Although most of the operations of an FMS are automated, human beings are needed to manage these operations.

The distinction between flexible manufacturing systems and flexible manufacturing cells is somewhat blurred. A cell usually refers to a group of machines either manually operated and/or automated, possibly (but not always) containing automated material handling, and possibly (but not always) being computer controlled. On the other hand, a flexible manufacturing system usually refers to a fully automated system with automated workstations, automated material handling, and computer control. Another distinction sometimes made is that a cell consists of a grouping of three or fewer machines, while a system consists of four or more machines (Groover, 1987).

The benefits of using FMSs can be enormous, especially if one considers that an average part spends 95% of its time in a factory waiting, 2% of its time being unloaded and loaded from processing stations, and 3% used to actually perform production operations. Assuming that a machine tool can be operated 24 hours a day 365 days a year, only 6% of its time is productive. Some of the benefits of FMSs include (Rembold, 1985):

1. ***Operate equipment 24 hours a day***. If operators are not present during night shift, the computer supervises the operation independently and if necessary, turns the system off when problems arise.

2. ***Minimize direct labor***. The computer automates most of the process. Labor is needed mainly for observance functions and maintenance.

3. ***Minimize lead time***. This is performed by the computer, which knows the machine status and production schedule.

4. ***Reduction of in-process inventory***. A FMS operates on the flow line production principle. Therefore, in-process inventory buffers are reduced to a minimum.

5. ***Reduce tools and fixture requirements***. Universal tools and fixtures are used because FMSs produce a larger part spectrum. As a result, retooling and setup times are shorter.

6. ***Obtain a high flexibility***. An FMS can process a variety of different part types simultaneously leading to more diversified production.

All of the above benefits result in greater productivity, lower costs, and the ability to respond more quickly to customer needs.

2.2 Material Handling and Storage

The material handling and storage system dramatically effects the efficiency, productivity, and profitability of any flexible manufacturing system. V. Daniel Hunt states that "materials handling for CIM, especially for complex application such as delivery of multiple parts to an assembly station, may be one of the biggest problems facing integrated automation" (Hunt, 1989). These systems tie the individual workstations together. Using computer control, automated materials handling (AMH) systems store and move products and materials. Some examples of handling and storage equipment include automatic guided vehicle systems (AGVS), car-on-track conveyors, horizontal transporters, in-floor towline conveyors, and automated storage and retrieval systems (AS/RS).

Three general applications exist for AMH systems. First, they are used to shuttle parts between stations in a CIM system, operating on commands from the CIM controller. For example, after it receives a message indicating that a workstation has finished work on a part, the controller will order the AMH system to retrieve the part and bring it to the next workstation in the routing. Secondly, AMH systems are used for moving work in process (WIP) from one manufacturing stage to the next within a factory. This is similar to the first application except the system is serving the entire factory. This can become extremely complex because the system has more area to cover, can encounter more potential obstacles and logistical difficulties in establishing paths for the AMH carriers, and handles a wider range of materials. The final application is in automated storage and retrieval systems (AS/RS). AS/RS is an automated warehouse, storing parts in racks and retrieving those parts on computerized carts and lift trucks. In order to conserve space and limit the number of carrier devices needed, the storage rack systems tend to be very tall. Some advantages of AS/RSs include fewer (but more highly trained) staff, reduced land needs for the plant, more accurate inventory records, and reduced energy use (Hunt, 1989).

2.3 Control of FMS

The component that has the greatest impact to an FMSs success is its computer control system and its costs reflect that. The computer control system typically costs between 15 and 40% of the total system cost. While the costs of control hardware are decreasing, the costs of software and integration continue to rise as cells require greater operating flexibility. Virtually all of these systems require a significant amount of specially written, application specific software (Greenwood, 1986).

2.3.1 Factory Control Hierarchy

The concept of the factory control hierarchy illustrates where the cell controller fits into an automated factory. The five levels of the factory control hierarchy are the following:

1. ***Factory level control***. This is the highest level of control, performing information management, manufacturing engineering, and production management functions.
2. ***Shop-floor level control***. This level is responsible for the real time management of jobs and resources within the shop-floor, performing the functions of task management and resource allocation. Task management schedules job orders,

shop-floor support services, and equipment maintenance. Resource allocation allocates workstations, tools, materials, and storage areas.

3. *Cell level control*. The sequencing of a batch of jobs through workstations and various support services, such as materials handling, is managed at this level Modules exist to do scheduling, batch management, dispatching, monitoring, and analysis.

4. *Workstation level control*. This level directs and coordinates the activities of small integrated groupings of shop-floor equipment.

4. *Device level control*. Particular pieces of equipment are controlled at this level.

The above hierarchy is only a guideline. Actual systems usually do not have exactly five layers (Haritos, 1991).

2.3.2 Cell Control - Functions

At the cell level, the computer control system performs several functions grouped in the following categories (Groover, 1987):

1. *Control of each workstation*. The individual machining or assembly stations usually operate under some form of computer control.

2. *Distribution of control instructions to workstations*. The computer control system coordinates the processing at the individual stations. For example, machining a part requires that part programs be downloaded to the machines of that workstation.

3. ***Production control***. The control system makes decisions such as part mix and rate of input of parts onto the system. In order to perform this function, the computer routes an applicable pallet to the load/unload area and tells the operator to load the desired raw material.

4. ***Traffic control***. The primary material handling system, which moves parts between workstations, is regulated by the computer control system.

5. ***Shuttle control***. The control system also regulates the secondary material handling systems at each workstation. The shuttle systems need to be coordinated with the primary material handling system and synchronized with the operations of their respective workstations.

6. ***Work handling system monitoring***. The status of each cart in the material handling systems and the status of each of the workpart types in the system must be monitored by the computer.

7. ***Tool control***. The computer control system must monitor and control the cutting tool status, including monitoring the tool-life and accounting for the location of each tool in the system.

8. ***System performance monitoring and reporting***. Various management reports can be generated by the computer.

2.3.3 Cell Control - Configurations

In order to perform the many different functions discussed above, the cell controller can be designed with one of three different configurations: centralized, heterarchical, and hierarchical (Lin, 1994). In a centralized configuration, one processor controls the entire cell operations. It uses a lock-step sequential control which cannot handle failures or be modified for cell configuration changes unless the control software is substantially re-programmed.

In a heterarchical configuration, each entity in the cell has its own software module, communicating with each other through a network link. This localized control strategy is flexible and has a good fault tolerance because far less global information is used. A disadvantage of the heterarchical configuration is that the excessive communication between the software modules considerably slows down the system, resulting in delays when the number of processes is large.

The most widely used configuration in manufacturing systems is hierarchical control. A hierarchical system uses a cell control master computer to coordinate the operations of all equipment, constantly monitoring all process functions. Each process function is controlled by a slave computer. The cell controller and equipment communicate by passing control commands and execution status feedback as messages in a computer network using well-defined and well-organized communication protocols.

One advantage of the hierarchical configuration is that the control problem is partitioned, limiting the complexity of any one module no matter how complex the entire system is. Also, the re-programming effort is minimized when a particular controller needs to be modified because the scope of the controller is limited to its (single) parent and (many) child nodes.

Mollo describes a hierarchical control system with three levels: *high* or supervisory level, *middle* or on-line control level, and *low* or machining control level. The lowest level controls each individual piece of equipment by managing its machining, assembly, or handling operations; managing the exchange of correct information with hardware and the upper levels; and managing the diagnostic local signals or messages. The middle level synchronizes the operations of several machines and controls the flow of parts throughout the system. While the low level is concerned only with the operation of its own machine, the middle level must also consider the surrounding environment. The highest level is the intelligence center of the system, performing the following functions: management of part programs database, management of statistical database, creation of files of statistical information, and management of tools database (Mollo, 1985).

2.3.4 Cell Control - PLC vs PC

Traditionally, programmable logic controllers (PLCs) were used as cell controllers, but PCs are being used more and more on the factory floor. PCs and PLCs seem to complement one another - each one's strengths complement the other's weaknesses. The PLC can be thought of as an optimized computer, tailored specifically for efficiently handling the movement of data values thoughout the control system and for quickly solving the logic equations of complex systems. On the other hand, the PC is less optimized for control tasks. The PC's strength of being able to adapt to a wide variety of applications actually limits its effectiveness in an application when compared to a computer optimized to that task. As a result, the PC is generally not as effective as the PLC in controlling industrial machines and processes (Howard, 1989).

PLCs also outshine the PC in input/output capabilities. PLCs can handle high-speed I/O very efficiently as required in real-time control systems. A programmable controller can decipher large numbers of input signals and use that information to control large numbers of outputs, repeating this process many times per second. Also, PLC manufacturers provide a wide variety of I/O modules, rugged enough to withstand the harsh conditions of plant-floor operation and able to meet the many needs of industrial applications with off-the-shelf solutions.

The operating system of the PC limits its ability to handle real-time I/O. Most PCs use the standard PC/DOS or MS/DOS, which are single-task operating systems. These operating systems do not provide the computer with the ability to periodically monitor the I/O ever few milliseconds or respond to interrupts from the hardware as required of real-time control systems. Also, it is difficult to get I/O to and from a PC. The special I/O modules often required by industrial and process control applications for handling the devices in the factory are not always compatible with the personal computer (Murphy, 1988).

Personal computers do provide several advantages over programmable computers. PLCs do not offer a good operator interface. They use simple devices, such as pushbuttons and indicator lights, to convey very limited information to the operator. On the other hand, the PC can be programmed to provide an excellent operator interface. Detailed operator instructions, context-sensitive menus, historical trend charts, and alarm and diagnostic messages are only a sample of the many ways a PC can convey easily-understood information to the operator. The analytical abilities of the PC also outperform the PLC. The PLCs limited math functions, sparse memory, and lack of mass storage only allow it to perform simple analysis of its data. In contrast, the PCs math smarts, plenty of memory, and mass storage allow it to perform extended mathematical functions and manipulate databases (Howard, 1989).

Because the PC and PLC complement each other so well, many flexible manufacturing systems are controlled using a combination of both computers. This provides the system with the ideal controller having the best of both worlds. The PC and PLC can communicate with one another using a serial data communication link using an electrically simple link, such as the PLC's programming port, or a network interface adapter (Howard, 1989).

Connecting the PC to the PLC augments the data acquisition and control functions of the PLC in several ways. The PC can provide the operator interface for a single PLC or network of PLCs. The PLC can provide necessary raw data to the PC, which can then manipulate that data and provide information to the operator. The mass storage capability of the PC allows it to store raw or processed data that has been collected by the PLC (Howard, 1989).

2.4 Computer Communications

An integrated manufacturing environment requires that information transfer easily and reliably across a network of computers, which probably are of different brands. Referring to the factory control hierarchy discussed in section 2.3.1, information is passed from lower to higher levels and instructions are issued from higher to lower levels. The communication requirements are different depending on the level. The critical response time decreases as the level increases. Whereas the device level measures time in milliseconds, the cell level measures time in seconds and the factory level in hours. Communication requirements can also be examined in relation to information and control requirements. The importance of control dominates the lower levels, while communication and information become more and more important as the level increases in the hierarchy. Johansson states that communications is the single most important factor at the cell level, followed by information and then control (Johansson, 1989).

All FMS communication systems are remarkably similar, regardless of the actual FMS in which they are used. The requirement of all FMS communications systems are the following (Greenwood, 1986):

1. To allow files to be sent and received from the devices in the FMS.
2. To send control instructions to the devices.
3. To receive status information from the devices.

One of the main communication problems is that the computers and computer-based devices in the system are not always compatible in their ability to communicate with one another. While all of these devices use digital pulses to represent data, the format and interpretation of these pulses differ. Communications protocols combat this problem by providing a set of formalized procedures to be followed by each device in the network whenever the exchange of data takes place (Groover, 1987).

2.4.1 MAP/OSI

Designed by the General Motors Corporation in conjunction with other companies, Manufacturing Automation Protocol (MAP) is a communications protocol providing a set of standards for use in factory communications. In the 1980s, GM realized that in their factories the computers and programmable machines from different vendors need to be able to exchange data. As a result, they developed and adopted MAP, requiring that all of their vendors use the MAP standard (Groover, 1987).

MAP is based on the full Open Systems Interconnection (OSI) seven layer model, a specification defined by the International Standards Organization (ISO). The layers in the model are based on functionality in that each layer has a distinct function in terms of communicating between levels among the devices in the network. The functions of the first four levels can be classified as interconnection functions, whereas the functions of the top three layers are interworking functions. Interconnection functions deal with problems of creating and maintaining a data pipeline permitting two devices to communicate regardless of any technological differences between them. This concerns the physical aspects of data transfer. On the other hand, interworking functions deal with higher-level issues relating to the data requirements for meaningful communication between devices to accomplish useful applications (Groover, 1987).

The seven layers of the OSI reference model are shown in Figure 2.1. The layers are described as follows (Popovic, 1990):

1. ***Physical layer*** is the lowest layer in the system. It controls the physical medium, used for the transmission of information within the network, by providing physical, functional, and procedural standards for accessing the medium.
2. ***Data link layer*** provides the ability to transmit information along the addressed communication link in an error-protected manner, and corrects any errors which occur in the physical layer.

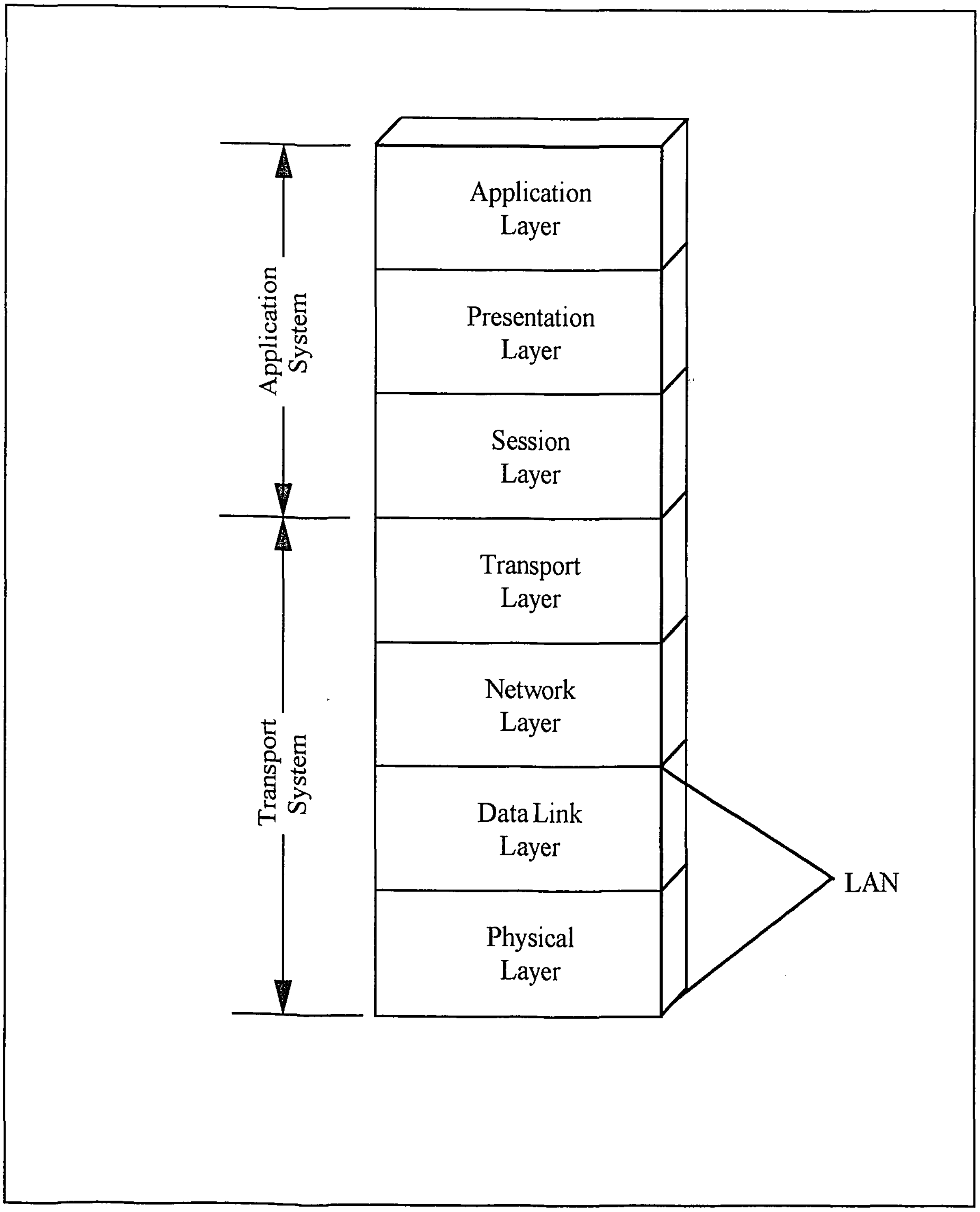


Figure 2.1 Seven Layers of the OSI Model

3. *Network layer* establishes transmission paths and network connections within the network.

4. *Transport layer* provides reliable transparent data transfer between the communicating devices.

5. *Session layer* opens, controls, structures, and terminates the communication between two parties, and also defines the transport connection to be used.

6. *Presentation layer* provides for communication not dependent of the device. The devices provide information in some specific language, which is then translated into a neutral language, understandable to the communication system.

7. *Application layer* is responsible for information transfer between application processes, primarily being concerned with the semantic of the application protocol.

2.4.2 Local Area Networks

Communication between computers in a factory occurs over a local area network (LAN). Groover defines a LAN as "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". Shown in Figure 2.2, the three most common topologies used in local area networks are: star network, ring network, and bus network (Groover, 1987).

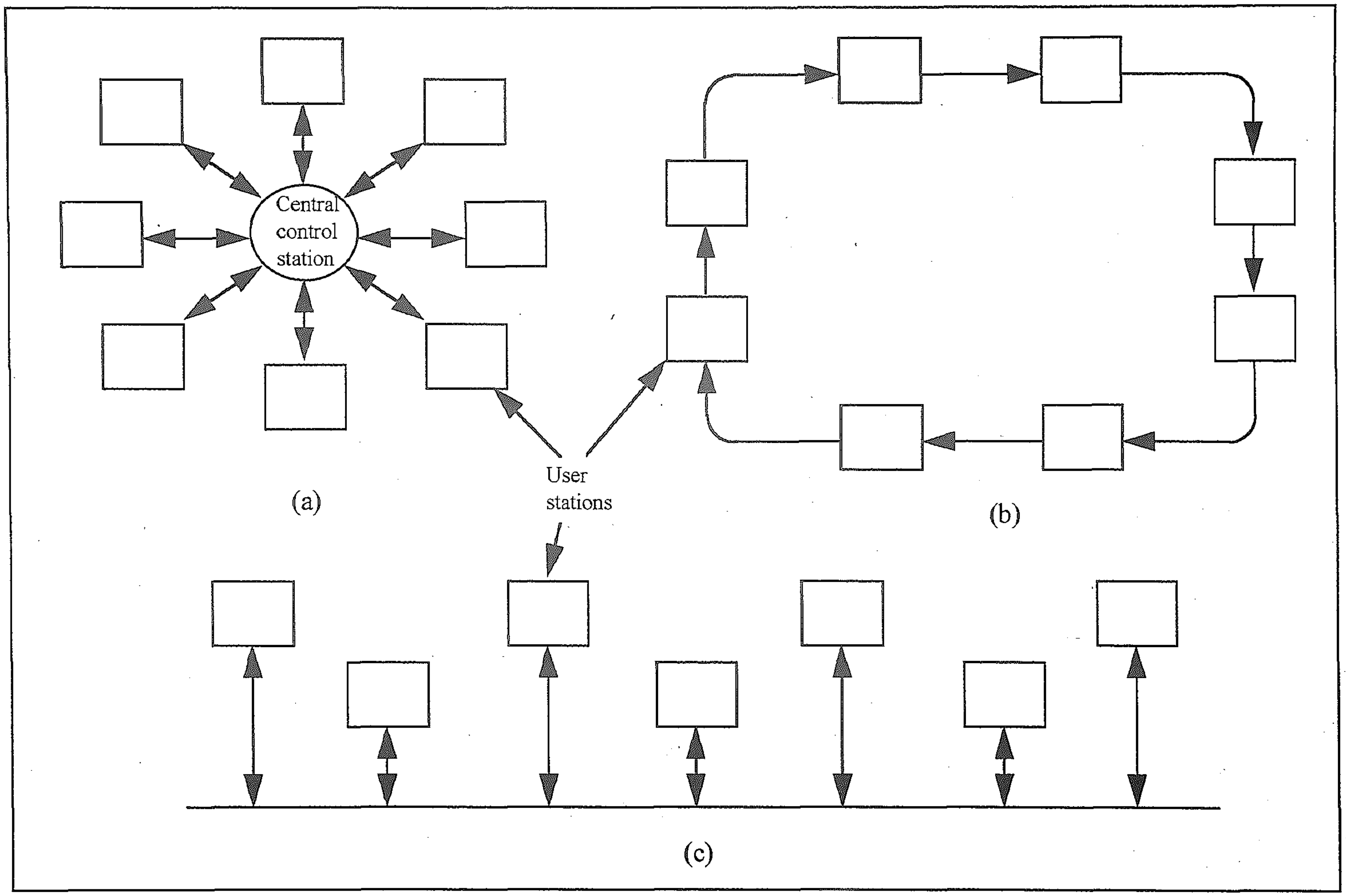


Figure 2.2 Three Common Network Configurations used in Local Area Networks: (a) Star; (b) Ring; (c) Bus.

In a star network, each device is connected to a central control station. In order to send a message from one device to another in the network, the message must go through the central station first and then to its destination. As a result, the central station controls the communications flow between the devices (Groover, 1987).

The individual stations in a ring network are connected in a continuous ring with each station having a neighboring station on both sides. Each station is assigned an address. When a station wishes to communicate with another, it sends the message, which contains the destination address, over the network. The message is relayed from station to station, with each station checking the destination address of the message. If it is the recipient, the station loads the message into memory; otherwise, the station sends the message to the next station in the network (Groover, 1987).

The bus network connects the individual devices with a single main transmission line. Similar to the ring topology, messages containing the destination address are sent through the bus and checked by each station until the recipient is found. Compared with the other two topologies, the bus network is the best bet for a factory LAN for several reasons. First, plant layouts often change to match changes in production requirements. The bus network permits these changes without major disruptions to the rest of the network. Second, the arrangement of the main transmission line of the network can closely match the machine layout in the factory. Third, generally the bus network is easier to repair and maintain than the ring or star networks (Groover, 1987).

While the central control station coordinates the communication flow in the star network, the ring and bus networks use one of two access methods to control the message transmissions between stations. The first method called *token-passing* uses a token, which is passed at high speeds from station to station in a predetermined sequence. Only the station in possession of the token can transmit a message. When a station receives the token, it can either transmit a message or pass the token to the next station if it has no message.

The second access method used by the ring or bus networks is the *carrier-sensed multiple access with collision detection* (CSMA/CD) method. Before transmitting a message on the network, a station listens to see if the another transmission is occurring. If the network is not being used, the station immediately sends its message. Otherwise, the station waits and tries again. If two or more stations try to send messages at the same time, a collision will occur, corrupting the messages. The stations will detect that a collision has occurred and will stop their transmission. Each station will wait a different amount of time and attempt to transmit again until a successful transmission has occurred.

2.5 Databases

Information is key to the success of integrated manufacturing systems, such as a flexible manufacturing system. Computer integrated manufacturing systems are heavily dependent on data communication. A cell controller relies on its information when making decisions. Therefore, the information must be accurate and timely. MacDow and Dirk claim that "the concept of CIM is based on the goal of having all the information that relates to a manufacturing decision available at the time of the decision" (MacDow, 1986). The manufacturing data base stores and processes a large number of data, making the design of the data base very important. Not only is the amount of data increasing but these data may be interrelated through very complex ties as the computer applications grow in the factory (Rembold, 1985).

Basically, two groups of data exist in manufacturing. The first group includes the following:

1. Information about the product, such as the bill of materials.
2. Information contained in the process planning file concerning manufacturing operations and process capability.
3. Information about the available processing machines.

The second group of data, which is used for order processing, typically includes planning and processing an order, machine selection and process sequencing, scheduling of shop orders, and material scheduling (Rembold, 1985).

A data base management system (DBMS) provides a consistent view of companywide data to all users, allowing them to update the data in a shared-access environment, as well as providing security, transaction logging, and query languages that sort data and create management reports. DBMSs are successful because they support data share, independence, consistency, and integrity (Hurson, 1993). Data redundancy occurs when the same data is duplicated in different files and should be avoided. Data integrity ensures that when the same information is stored in multiple files, any change in one file also changes the other files. The problem of data dependence occurs when information must be combined from two or more files which are incompatible with one another (McKeown, 1993).

The four basic types of DBMSs are: hierarchical, network, relational, and object-oriented. Of the four types the best candidates for use in advanced manufacturing systems are relational and object-oriented DBMSs because they offer the most flexibility, achieve maximum data independence, reduce redundancy, and achieve the most data integration. The hierarchical model resembles the structure of a tree, where the "trunk (parent) has several "branches" (descendants), and each branch, in turn, has several smaller "branches" (next generation of descendants). Therefore, the data elements have hierarchical relationships or one-to-many relationships. Each succeeding data element is linked to only one leading data element (McKeown, 1993).

In the network model, the data elements have many-to-many relationships with each other. Therefore, each data element can have multiple parents and multiple children. Logical relationships are represented in the form of networks, composed of nodes and branches. The nodes are the data elements and the branches are the relationships or links between them, which are essential to the functioning of the network (McKeown, 1993).

In the relational model, the data base is composed of tables of attributes or relations. The tables are flat files with the rows corresponding to records and the columns corresponding to fields or attributes. For example, an inventory table would have a record on each item in the inventory with the attributes containing specific information on that item. The attributes might include part number, description, vendor number, and quantity on hand. The primary key, which uniquely identifies each record, is used to access data in the table. The primary key in the inventory table would be the part number. Foreign keys create links between the tables, allowing one to combine information from multiple tables into management reports (McKeown, 1993). The foreign key in one table is the primary key in other. Therefore, that foreign key links the two tables together. In the above example, the foreign key is vendor number, which provides a link to the vendor table. Therefore, the user could create a management report with items in inventory and information on their respective vendors.

Object-oriented data bases attempt to implement a collection of programming concepts, known as object-oriented programming, in a multi-user, shared-access environment. Some of these concepts include objects, classes, instances, messages, methods, encapsulation, and inheritance. Objects are packages of information (data) and descriptions of its manipulation (procedures). Classes describe similar objects, whereas instances are objects of a particular class. Messages are specifications for manipulating an object. Methods describe the sequence of actions to be performed when a message is received. Encapsulating an object protects its data from the outside world. In order to access an object's data, one must send a message to that object. When the object receives the message, it activates the corresponding method and returns the result to the sender. The concept of inheritance allows one to create a subclass, which inherits all or part of the properties of the superclass (Hurson, 1993).

In an object-oriented data base, all conceptual data entities are modeled as objects. An object consists of private memory holding its state and has a object identifier uniquely identifying it from other objects. It can be primitive only containing a value, or complex containing a collection of attributes. Objects send messages in order to communication with one another. After receiving a message, an object will execute the corresponding method. Collectively, attributes and methods are called properties. Instances are objects of the same class, and therefore, share the same definition for attributes and methods, responding to the same set of messages (Hurson, 1993).

Chapter 3 Previous System Description

3.1 The Cartrac System

Purchased from SI Handling Systems Incorporated in 1986, the cartrac was the primary material handling system for the flexible manufacturing cell. The purpose of the cartrac was to move WIP from one station to another and if necessary, provide temporary storage for WIP. The system consisted of several carts, used to move pallets, traveling over tracks and three shuttles. As depicted in Figure 3.1, a frame supported the two-railed track, placing it a few feet above floor level. The carts were driven by a rotating tube in combination with a drive wheel. The position of the drive wheel determined the speed of the cart. The cart achieved a maximum speed when the drive wheel was at a 45 degree angle to the tube and stopped moving when the wheel was perpendicular to the tube. Because the carts could not turn at the corners of the track, three shuttles transported the carts between the tracks.

Carts were moved between the cartrac and AS/RS using a transfer device, composed of a turn table, load arm, and unload arm. The turntable provided the proper orientation to the pallet being transferred between the two systems, and also provided a connection over a slight elevation difference between the AS/RS and cartrac. A load arm pushed the pallet from the cartrac onto the turntable, while an unload arm pushed the pallet from the turntable onto the cartrac.

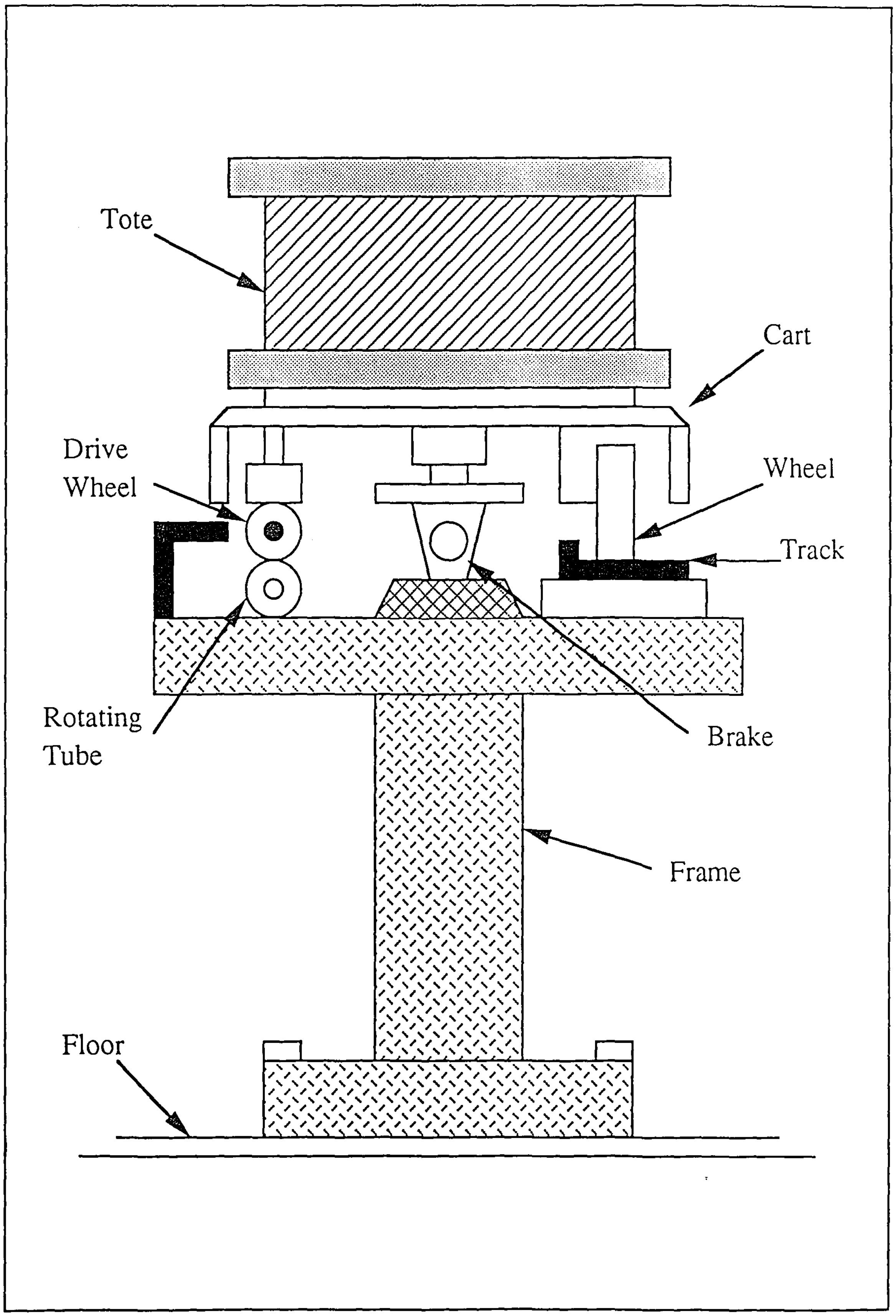


Figure 3.1 Cart and Track

Previously, the operation of the cartrac system was controlled by a microcomputer (or PC) connected to a Programmable Logic Controller (PLC). The microcomputer acted as a supervisory computer, providing a way for the user to enter a cart's route in a user-friendly atmosphere. The software, written in QBASIC, asked the user to enter the desired route of the cart, using the following notation:

A	Transfer from cartrac to AS/RS
C	Transfer from AS/RS to cartrac
T	Move to transfer device
1, 2, 3, or 4	Move to the station indicated (1, 2, 3, or 4)

For example, the operator would enter C12TA to transfer the pallet from the AS/RS to the cartrac, move it to station 1 and then 2, return it to the transfer device, and finally, transfer it from the cartrac to AS/RS.

The computer translated the route entered into low level signals and sent those signals in the proper sequence to the PLC. The PLC used the low level signals to sequence the physical hardware. PLCs are solid-state devices used to perform logic functions previously accomplished by electromechanical relays (Petruzella, 1989). The PLC made logical decisions based on inputs received from input devices, such as mechanical switches and photo sensors, and provided output to output devices, such as motors and solenoids, controlling the movement of the carts.

A block diagram of the cartrac's basic control system is shown in Figure 3.2. The handshaking signals required to control a cart are described in Table 3.1. Signals sent from the supervisory computer informed the PLC which station to send the cart to or whether to transfer the cart to the AS/RS or cartrac. Signals sent from the PLC to the supervisory computer indicated whether movement is taking place or a machining center is processing. Transfers between the AS/RS and the cartrac were controlled using the signals communicated between the PLC and transfer device.

The supervisory computer and PLC used coils or memory locations (C0, C1, etc.) in the PLC to communicate with one another by setting/resetting them and periodically checking their values. The supervisory computer would set only one destination signal (C0, C1, C2, C3, C4) at a time and would reset that signal when the PLC had reset C20 and set C23. Once a cart arrived at the transfer device, the supervisory computer could set either C5 or C6, but not both, indicating the type of transfer needed. Finally, the supervisory computer would set one of four other signals (C11, C12, C13, C14) while the machining center processed and reset that signal upon completion of the processing.

As shown in Table 3.1, the PLC sent three signals to the supervisory computer. The PLC set C20 when the cart had completed the move and reset it when the supervisory computer provided the next move. C21 was set when a cart reached a station. The PLC set C23 when a cart was moving and reset the signal upon completion of the move. The

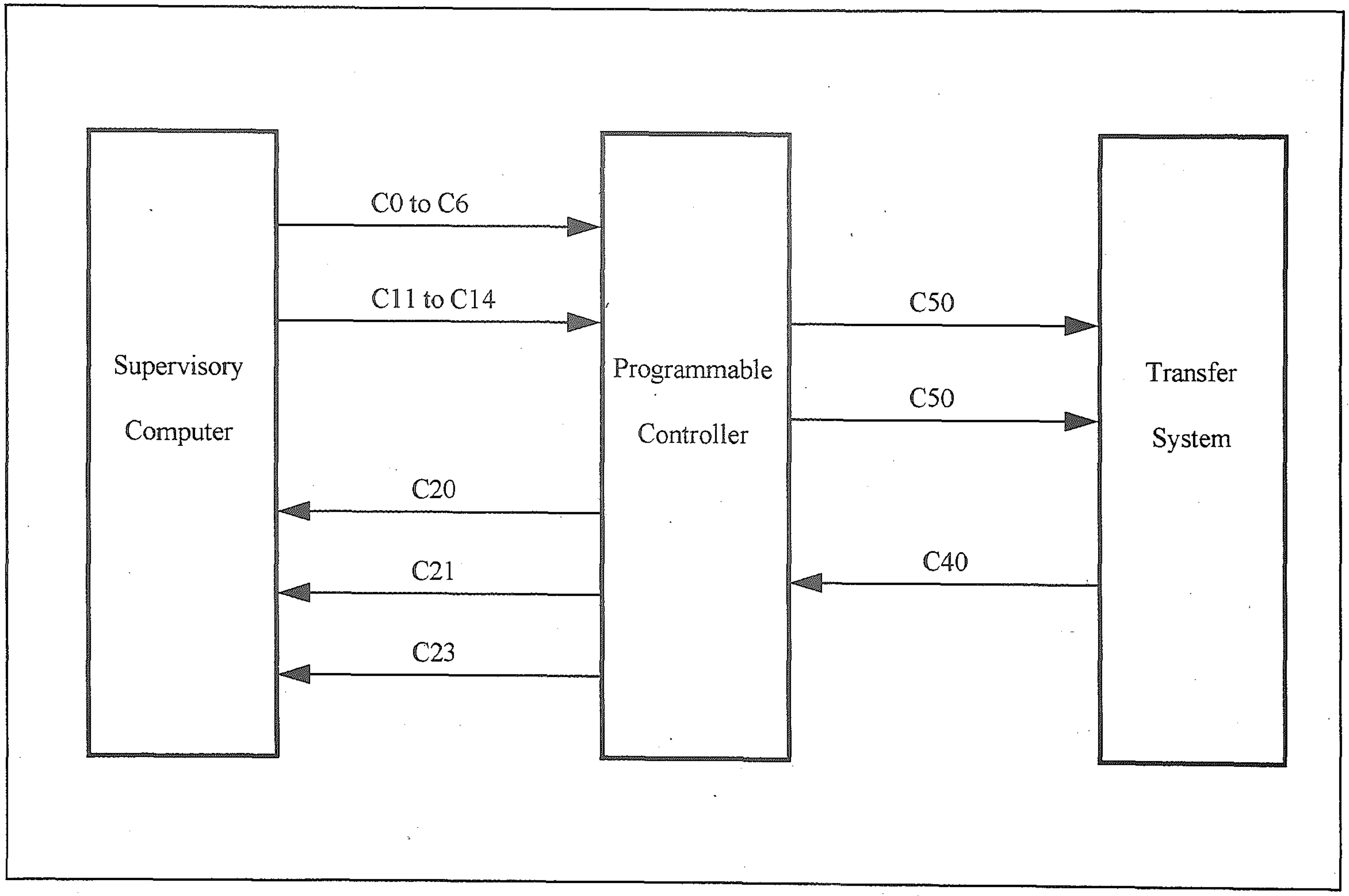


Figure 3.2 Cartrac Control System Block Diagram

Table 3.1 Handshaking Signals between Cartrac Supervisory Controller and PLC

From/To	Signal	Description
Supervisory PC to PLC	C0	ON = next destination is ASRS transfer device
	C1	ON = next destination is Machine #1
	C2	ON = next destination is Machine #2
	C3	ON = next destination is Machine #3
	C4	ON = next destination is Machine #4
	C5	ON = transfer pallet from ASRS to cartrac
	C6	ON = transfer pallet from cartrac to ASRS
	C11	ON = Machine #1 is processing
	C12	ON = Machine #2 is processing
	C13	ON = Machine #3 is processing
	C14	ON = Machine #4 is processing
PLC to Supervisory PC	C20	ON = present move is complete
	C21	ON = run machining process
	C22	ON = move command is being processed
Transfer Device to PLC	C40	ON = transfer in process
PLC to Transfer Device	C50	ON = transfer from ASRS to cartrac OFF = transfer from cartrac to ASRS
	C51	ON = begin transfer

supervisory computer would not send a signal to the PLC indicating a new move unless C20 was set and C23 was reset.

The PLC communicated with the transfer device when the supervisory computer requested a transfer of a cart on the transfer device. Immediately after receiving a transfer signal (C5 or C6) from the supervisory computer, the PLC reset C20 and set C23, indicating the processing of the transfer. Using signal C50, the PLC indicated the direction of the transfer to the transfer device and set C51 instructing the transfer device to begin transferring the cart. The transfer device set C40 during the transfer and reset it upon completion. The PLC reset C51 once the transfer had actually begun, and set C20 and reset C23 once the transfer was completed. The communication protocol described above for the cartrac system is shown in Figure 3.3.

3.2 The Automatic Storage and Retrieval System

In 1987, Lehigh University purchased the AS/RS housed in the ManTech Laboratory. The AS/RS was a single-aisle, twin row system. As shown in Figure 3.4, the system was twelve levels high with two rows and twelve columns. Similar to the cartrac system, the AS/RS was controlled using a supervisory computer and PLC. The operator could control the AS/RS either manually or semi-automatically directly from the PLC using the control panel, or he could automatically control the system from the supervisory computer.

Transfer - AS/RS to Cartrac:

Supervisory PC	C0(1)		C0(0)			
PLC		C20(0), C23(1), C50(1), C51(1)		C51(0)		C20(1), C23(0)
Transfer Device			C40(1)		C40(0)	

Transfer - Cartrac to AS/RS:

Supervisory PC	C0(1)		C0(0)			
PLC		C20(0), C23(1), C50(0), C51(1)		C51(0)		C20(1), C23(0)
Transfer Device			C40(1)		C40(0)	

Move to Machine 1:

Supervisory PC	C1(1)		C1(0)		C11(1)	C11(0)	
PLC		C20(0), C23(1)		C21(1)			C21(0), C20(1), C23(0)

Move to Machine 2:

Supervisory PC	C2(1)		C2(0)		C12(1)	C12(0)	
PLC		C20(0), C23(1)		C21(1)			C21(0), C20(1), C23(0)

Move to Machine 3:

Supervisory PC	C3(1)		C3(0)		C13(1)	C13(0)	
PLC		C20(0), C23(1)		C21(1)			C21(0), C20(1), C23(0)

Move to Machine 4:

Supervisory PC	C4(1)		C4(0)		C14(1)	C14(0)	
PLC		C20(0), C23(1)		C21(1)			C21(0), C20(1), C23(0)

Notation:

C0(0) = reset coil C0
C0(1) = set coil C0

Figure 3.3 Communication Protocol between Cartrac Supervisory Controller and PLC

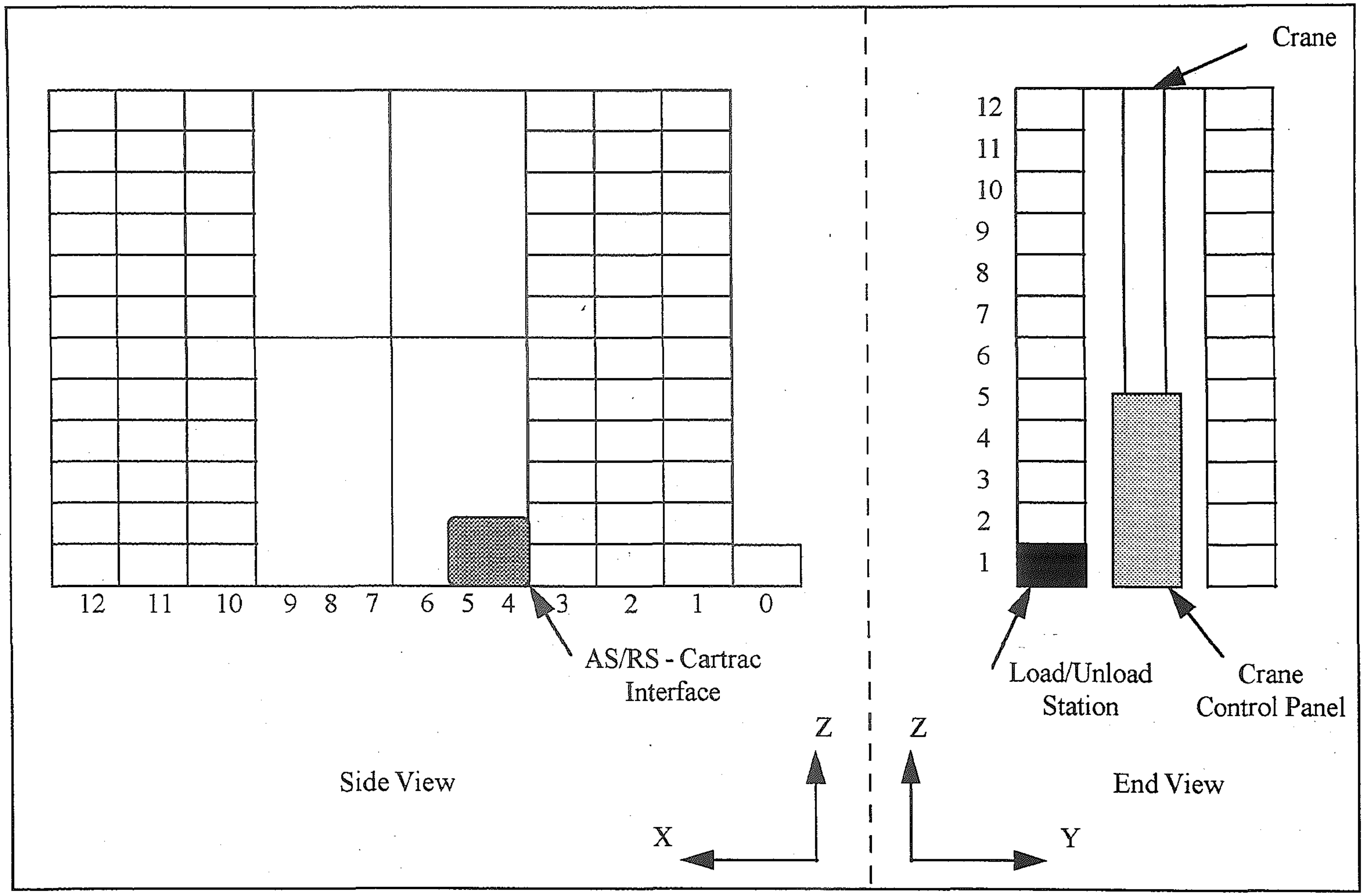


Figure 3.4 AS/RS Layout

Using the supervisory computer, the operator could control the AS/RS in a user-friendly way. Written in C++, the computer software provided a main menu of options available for the operator, including retrieving a tote, storing a tote, changing the contents of a tote, and viewing the tote list. Each tote was assigned a tote number, which was provided by the operator in order to retrieve or store that tote or change the contents. If the operator did not know the tote number, he was given the opportunity to see a listing of the totes.

The inventory information was stored in an array of objects of class ToteInventory, called ASRSTotes. The information of every storage location in the AS/RS was kept in the ASRSTotes array, even if the location had no physical tote (or pallet). The tote database contained the following information: tote number, location coordinates (x, z, and y), contents, owner, and last date used or changed. If an actual tote was not presently in that location, the tote's contents indicated that no tote existed.

The AS/RS controllers located tote positions using a coordinate system shown in Figure 3.4. The x variable corresponded to the column number of the AS/RS, looking at its side. Previously, only columns 1, 2, 3, 10, 11, and 12 had bins associated with them. Totes were retrieved from the load/unload station at column 0, located at the north end of the ManTech Laboratory. The height of the tote location was provided using the z variable. The y coordinate represented the system's two rows, numbered 1 and 2, corresponding to the side of the machine from which a bin should be loaded or unloaded.

When standing at the north end of the system and looking south, row 1 was the left side and row 2 was the right side.

In order to retrieve and store a tote, the supervisory computer and PLC communicated using instructions and messages. Based on the operation required, the supervisory computer passed the instructions shown in Table 3.2. Instruction 1 set the home position coordinates (x and z variables), which was convenient for positions used frequently. In order to initialize the system after a breakdown, instruction 2 provided the PLC with the coordinates of its current position. Using instruction 3, the supervisory computer told the PLC to move to the given coordinates. Similarly, instruction 4 told the PLC to move to the home position, previously defined using instruction 1. The supervisory computer told the PLC to store a tote using instruction 5 and to load a tote using instruction 6. A flaw existed in the system preventing instruction 5 from storing a tote to the right of the machine (row = 2). Therefore, instruction 7 was used when a tote was stored in row 2. Instruction 7 combined instructions 3 and 5 by moving to the tote location and then storing the tote. Similarly, instruction 8 combined instructions 3, 6 and 4 by moving to the tote location, loading the tote, and finally moving to the home position.

As shown in Figure 3.5, the supervisory computer and PLC communicated using a protocol consisting of three sets of messages. The supervisory computer used message one to send the specific command. Message two instructed the PLC to execute the

Table 3.2 Commands from AS/RS Supervisory Controller to PLC

Instruction No.	Command	Message (AAAAAAAA)
1	Set Home	00010000
2	Set Current	00010100
3	Move to x, z	0x0z0200
4	Move Home	00010300
5	Store a Tote (only stores left)	00014400
6	Retrieve from Left	00015400
	Retrieve from Right	00011400
7	Move and Store, Left	0x0z4600
	Move and Store, Right	0x0z0600
8	Move and Retrieve, Left	0x0z5500
	Move and Retrieve, Right	0x0z1500

Message One:

```
Supervisory PC   4E2105          0130313832303036313030303430311709   02AAAAAAAA03LL      04
PLC                     4E2106                                      06                 06
```

Message Two:

```
Supervisory PC   4E2105          013031383230303634303030323031170A   0280000380    04
PLC                     4E2106                                      06            06
```

Message Three:

```
Supervisory PC   4E2105          0130313032303036313030303430311701                 06    04
PLC                     4E2105                                      0602xxzz??RR03LL    04
```

Where:

xx	=	the column location
zz	=	the shelf location
LL	=	LRC byte (exclusive or of the bytes between 02 and 03)
??	=	varies with every command; not important
RR	=	80 before task is complete; 81 upon task completion
AAAAAAAA	=	command specific; see Table 2.2 for values

Figure 3.5 Communication Protocol between AS/RS Supervisory Controller and PLC

command given in message one. Finally, the computer waited for the PLC to complete processing of the command using message three.

Chapter 4 Development of Controllers for the FMS

Creating a flexible manufacturing cell integrating the AS/RS and cartrac system required the development of a cell controller. In order to control the cell, the cell controller sent high level commands to the AS/RS and cartrac supervisory controllers. Communications between controllers occurred over a local area network. The existing AS/RS and cartrac controllers had to be modified, providing an interface with the cell controller.

Figure 4.1 shows the configuration of the cell's controllers. The cell controller communicated directly with the AS/RS and cartrac supervisory controllers, which in turn, communicated directly with their corresponding PLC. The AS/RS and cartrac supervisory controllers did not communicate with one another. As a result, only the cell controller could integrate the actions of the AS/RS and cartrac together.

This design provided a hierarchy of control to the flexible manufacturing cell. The cell controller coordinated the actions necessary to manufacture a product by sending and receiving high level messages to/from the AS/RS and cartrac supervisory computers. The supervisory computers each controlled a sub-system of the cell by communicating with their respective PLC using lower level messages. Finally, each PLC directly controlled the hardware in its system by providing output to output devices.

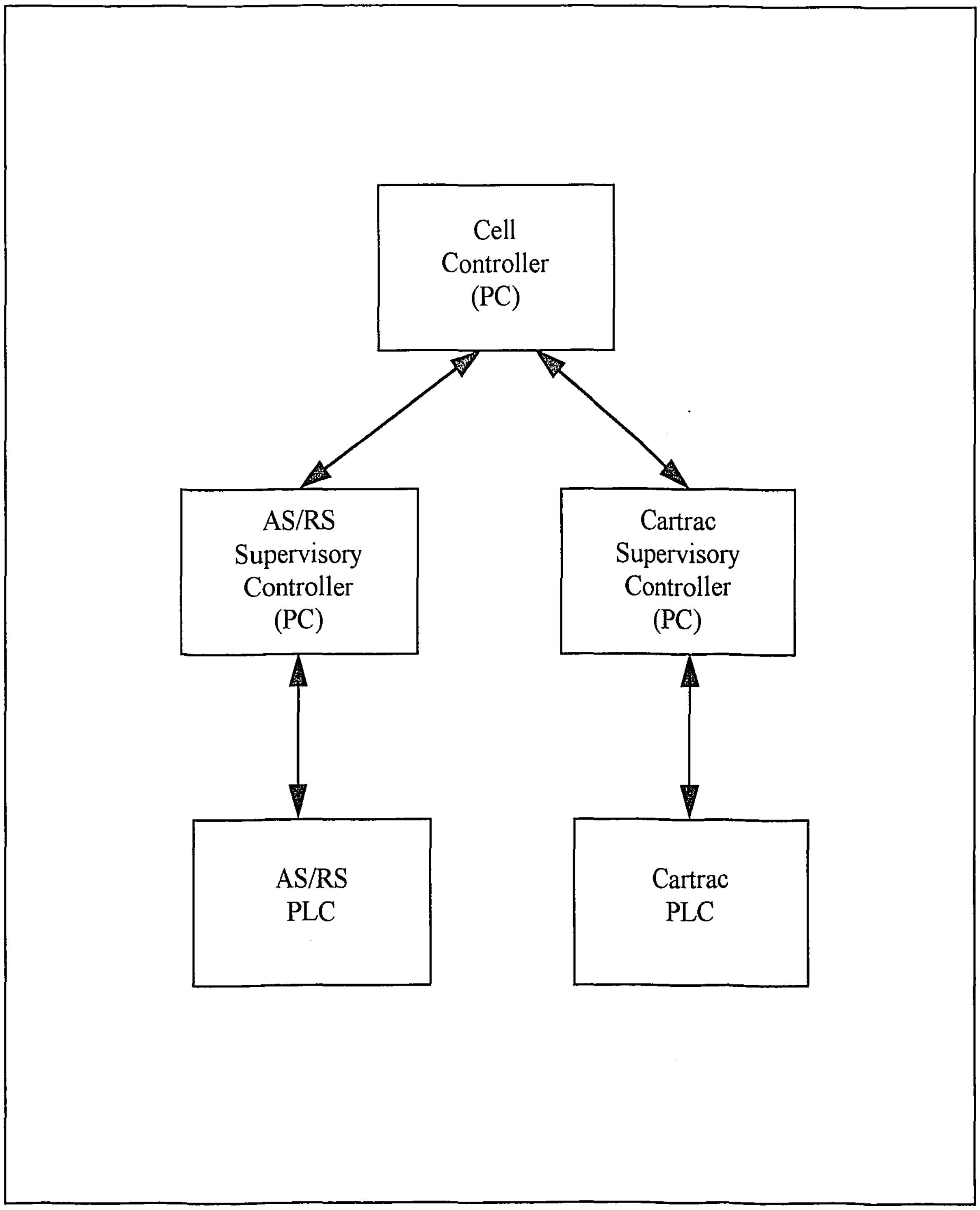


Figure 4.1 Configuration of Flexible Manufacturing Cell Controllers

4.1 Development of the Cell Controller

The design of each cell controller varies somewhat depending on the actual system it controls. Even so, much of the fundamental logic of the software design of different cell controllers is similar. In producing a part, all cell controllers need to coordinate the retrieval of the raw materials, machine or assembly processing according to the part's routing, storage of the completed or partially completed part, and movement of the part between these locations. To afford greater flexibility, the design of the cell controller in the ManTech Laboratory allowed for the processing of both machined parts and assemblies. Figure 4.2 shows the flowchart of the cell controller software.

4.1.1 Databases

First, the cell controller read in two databases supporting its operations, the part database and process database. The part database stored generic information on all parts used by the system, including raw materials and end products. This information included part number, description, number of components, bill of material, number of routing operations, and routing. Parts with no bill of material and/or routing, such as raw materials or purchased parts, were assigned a zero for number of components and/or number of routing operations. A part could only be contained in another part's bill of

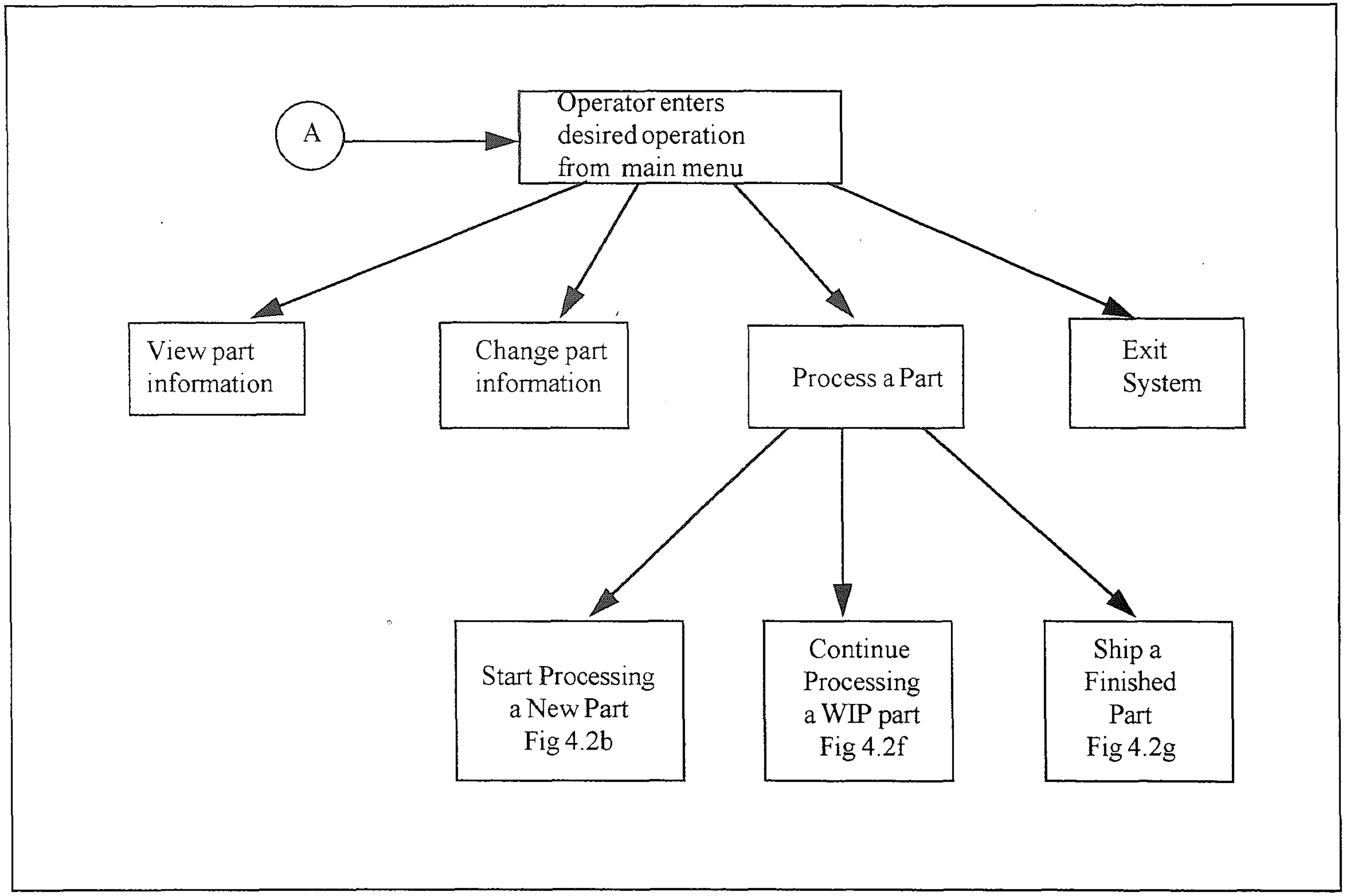

A
Operator enters desired operation from main menu
View part information
Change part information
Process a Part
Exit System
Start Processing a New Part Fig 4.2b
Continue Processing a WIP part Fig 4.2f
Ship a Finished Part Fig 4.2g

Figure 4.2a High Level Flowchart of Cell Controller

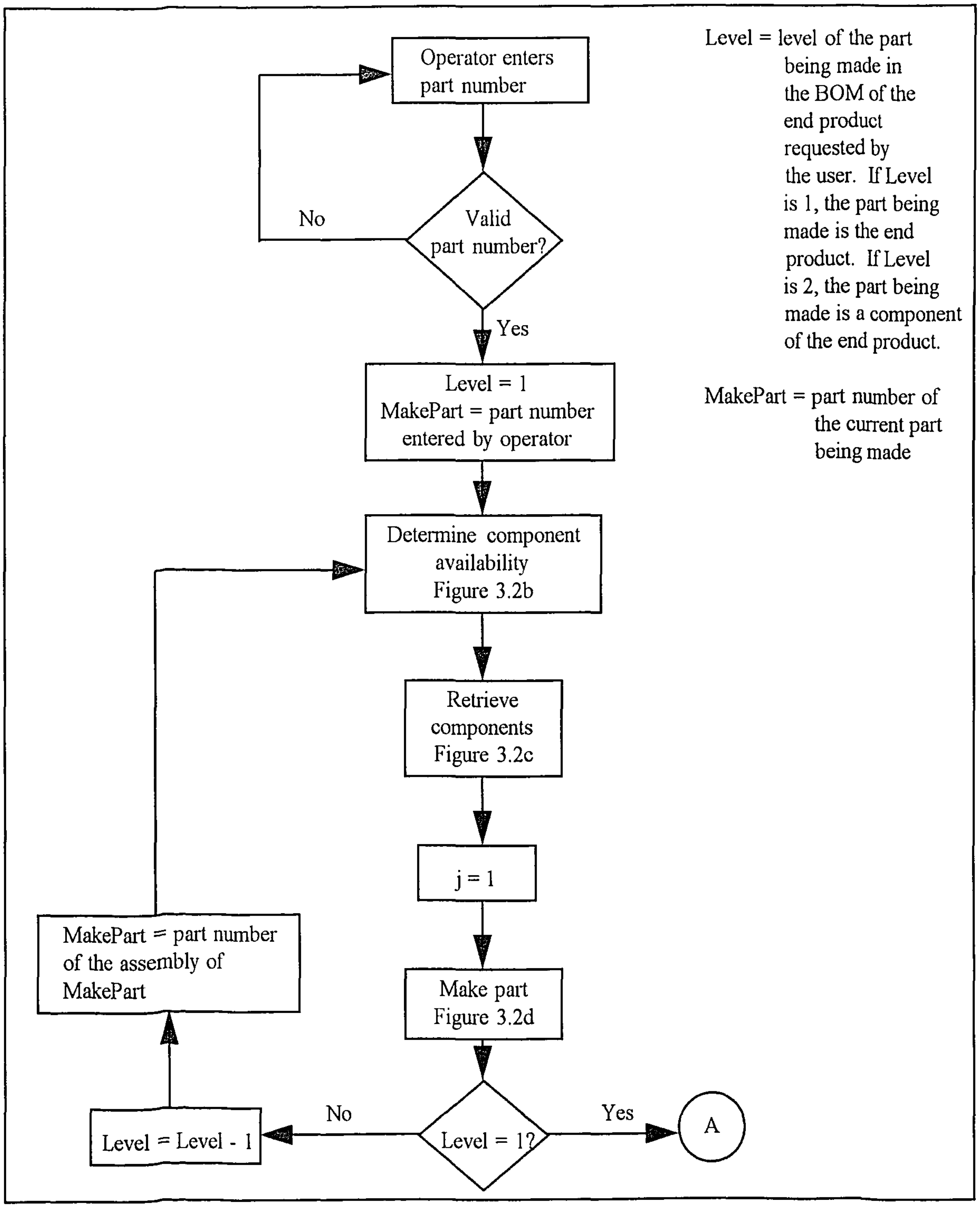

Operator enters part number
Valid part number?
No
Yes
Level = 1 MakePart = part number entered by operator
Determine component availability Figure 3.2b
Retrieve components Figure 3.2c
j = 1
Make part Figure 3.2d
Level = 1?
No
Yes
A
Level = Level - 1
MakePart = part number of the assembly of MakePart
Level = level of the part being made in the BOM of the end product requested by the user. If Level is 1, the part being made is the end product. If Level is 2, the part being made is a component of the end product.
MakePart = part number of the current part being made

Figure 4.2b Flowchart of Cell Controller - Start Processing

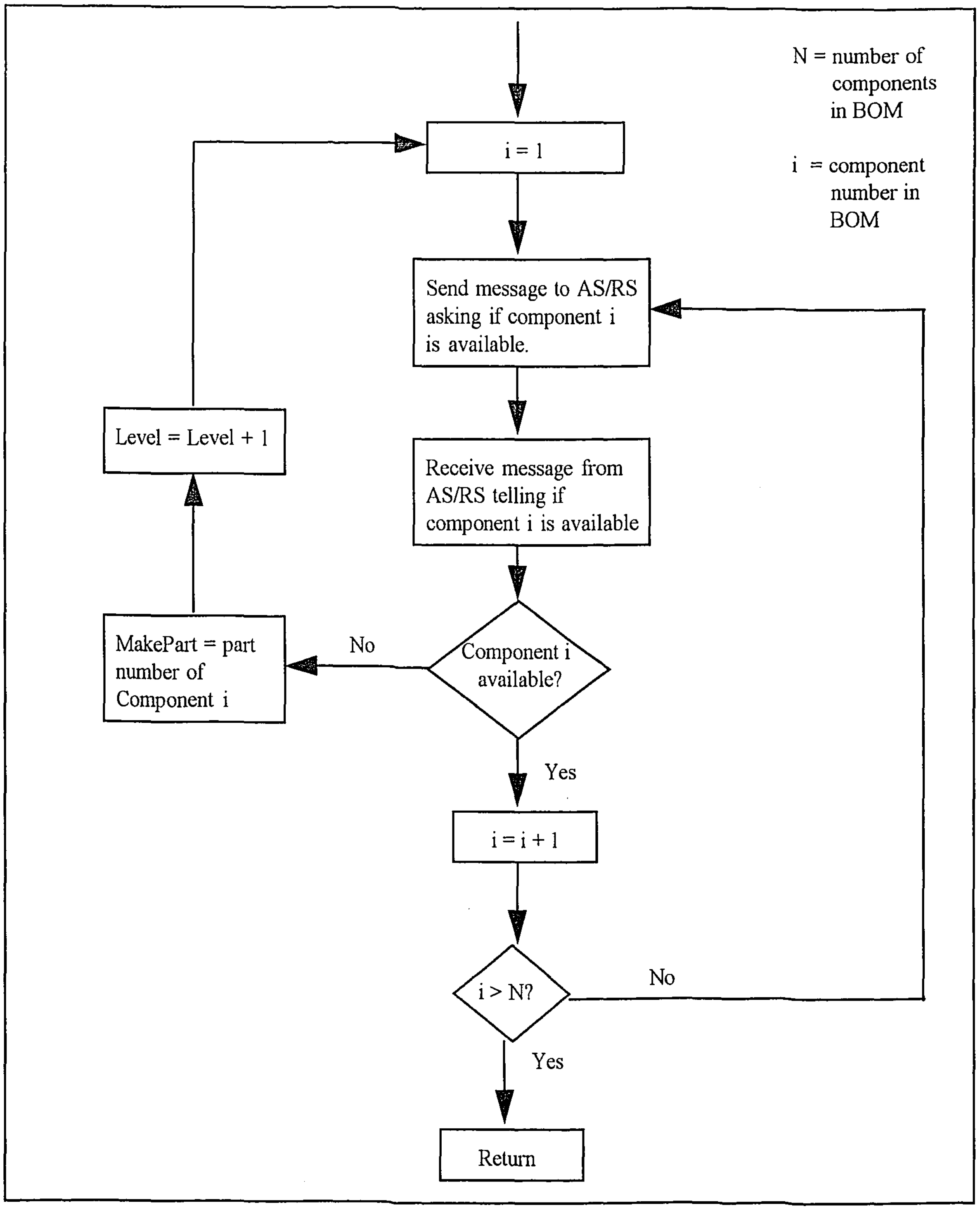
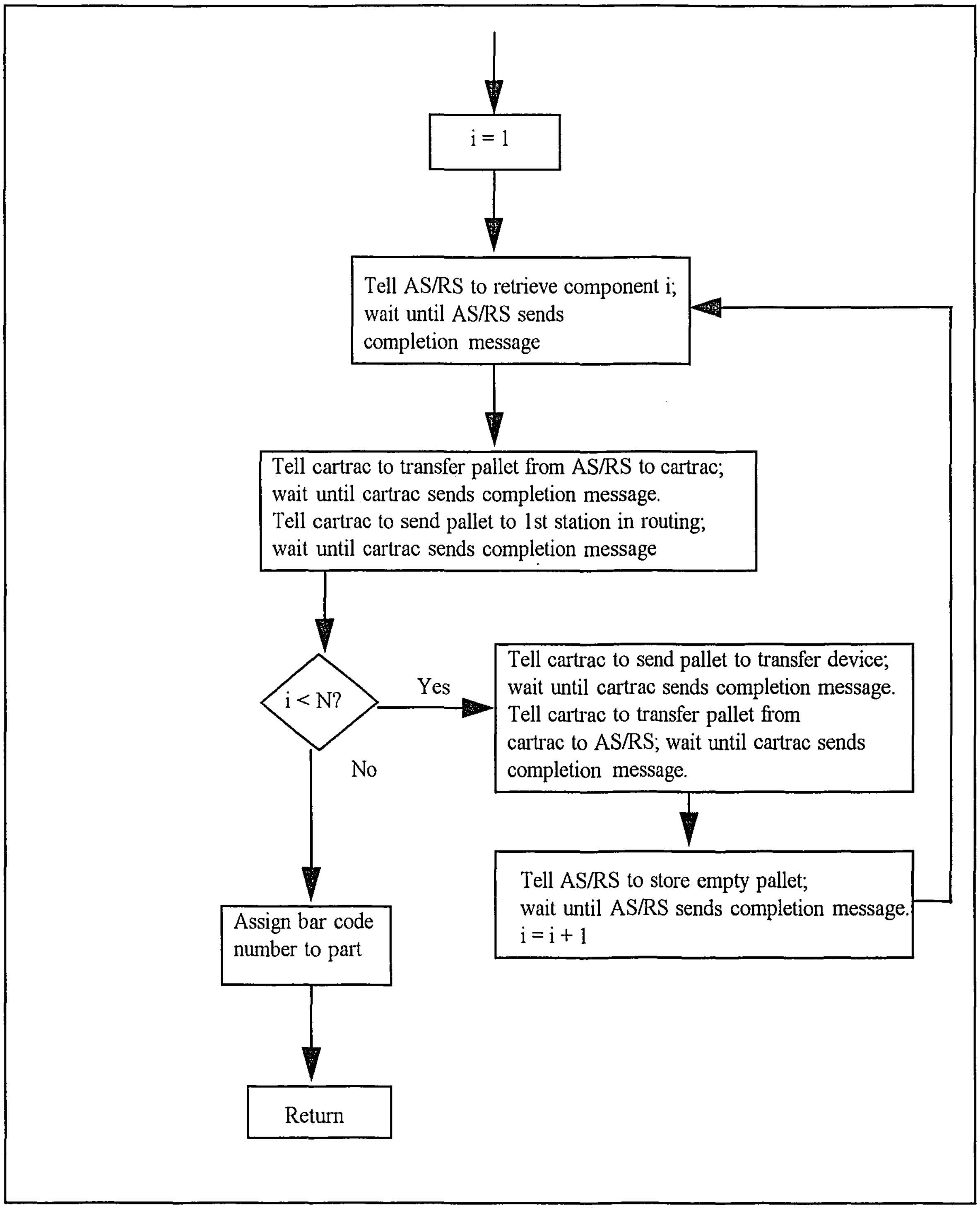

N = number of components in BOM
i = component number in BOM
i = 1
Send message to AS/RS asking if component i is available.
Receive message from AS/RS telling if component i is available
Component i available?
No
MakePart = part number of Component i
Level = Level + 1
Yes
i = i + 1
i > N?
No
Yes
Return

Figure 4.2c Flowchart of Cell Controller - Determine Component Availability



i = 1
Tell AS/RS to retrieve component i; wait until AS/RS sends completion message
Tell cartrac to transfer pallet from AS/RS to cartrac; wait until cartrac sends completion message. Tell cartrac to send pallet to 1st station in routing; wait until cartrac sends completion message
i < N?
Yes
No
Tell cartrac to send pallet to transfer device; wait until cartrac sends completion message. Tell cartrac to transfer pallet from cartrac to AS/RS; wait until cartrac sends completion message.
Tell AS/RS to store empty pallet; wait until AS/RS sends completion message. i = i + 1
Assign bar code number to part
Return

Figure 4.2d Flowchart of Cell Controller - Retrieve Components

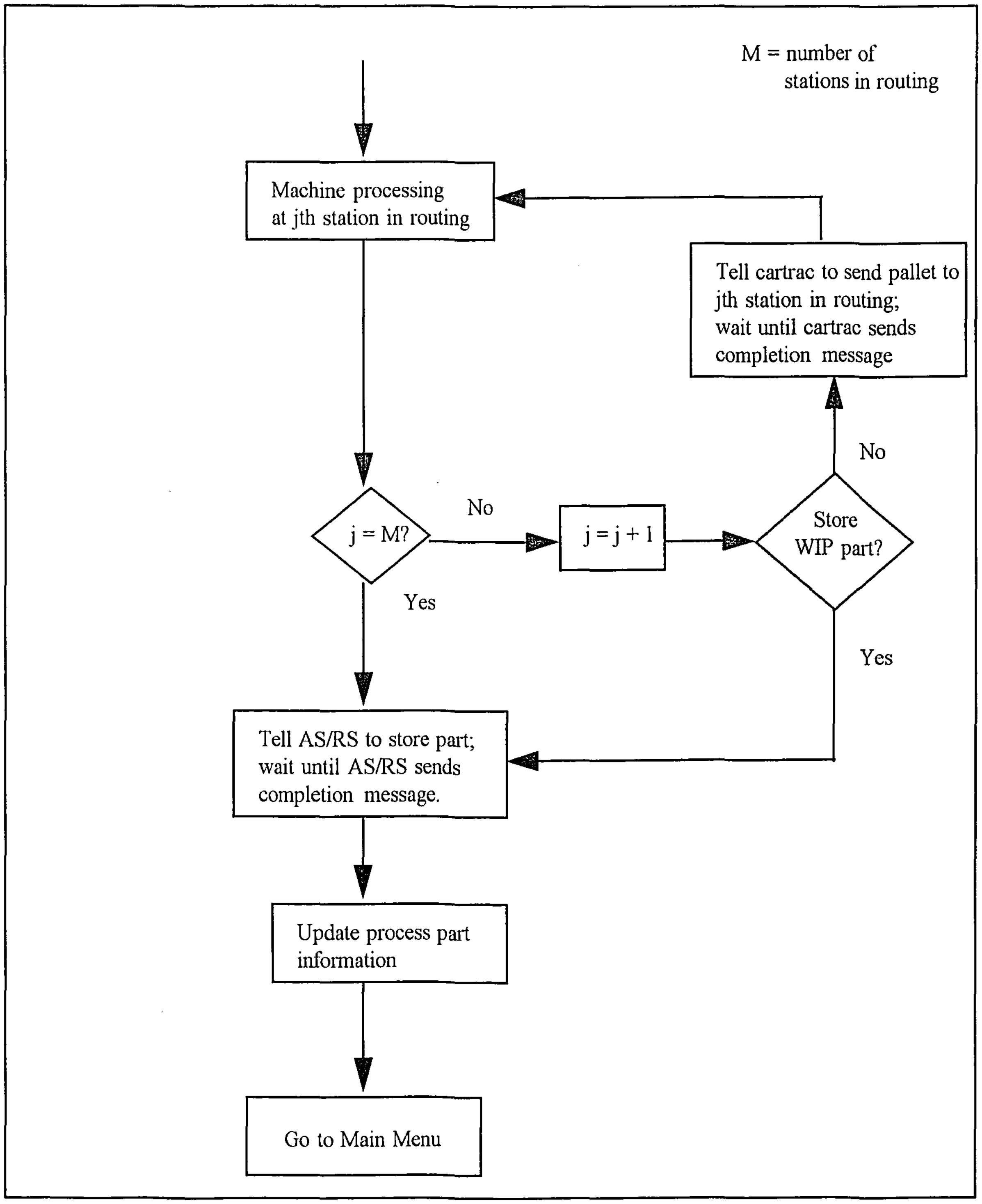

M = number of stations in routing
Machine processing at jth station in routing
Tell cartrac to send pallet to jth station in routing; wait until cartrac sends completion message
No
j = M?
No
j = j + 1
Store WIP part?
Yes
Yes
Tell AS/RS to store part; wait until AS/RS sends completion message.
Update process part information
Go to Main Menu

Figure 4.2e Flowchart of Cell Controller - Make Part

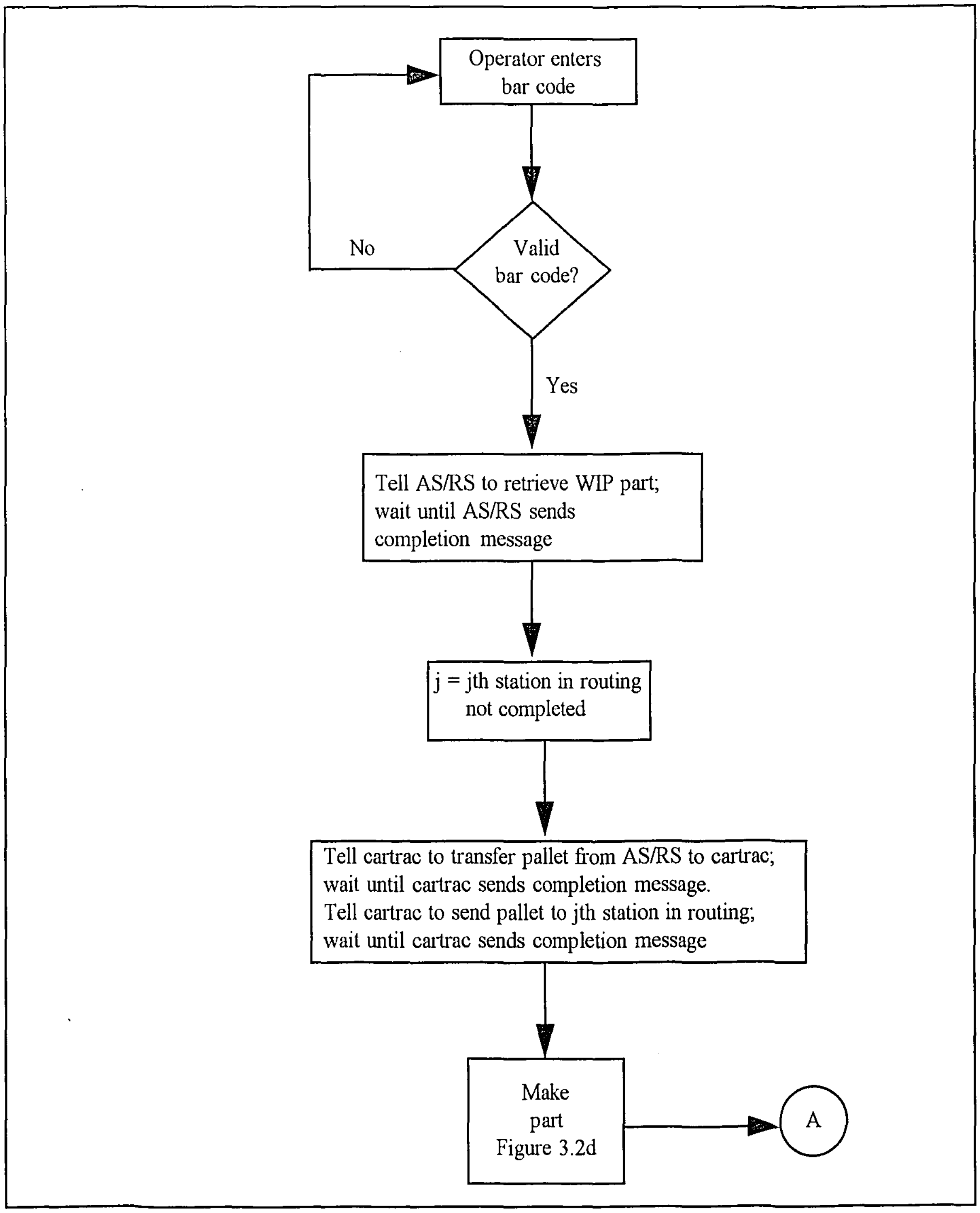

Operator enters bar code
Valid bar code?
No
Yes
Tell AS/RS to retrieve WIP part; wait until AS/RS sends completion message
j = jth station in routing not completed
Tell cartrac to transfer pallet from AS/RS to cartrac; wait until cartrac sends completion message. Tell cartrac to send pallet to jth station in routing; wait until cartrac sends completion message
Make part Figure 3.2d
A

Figure 4.2f Flowchart of Cell Controller - Continue Processing

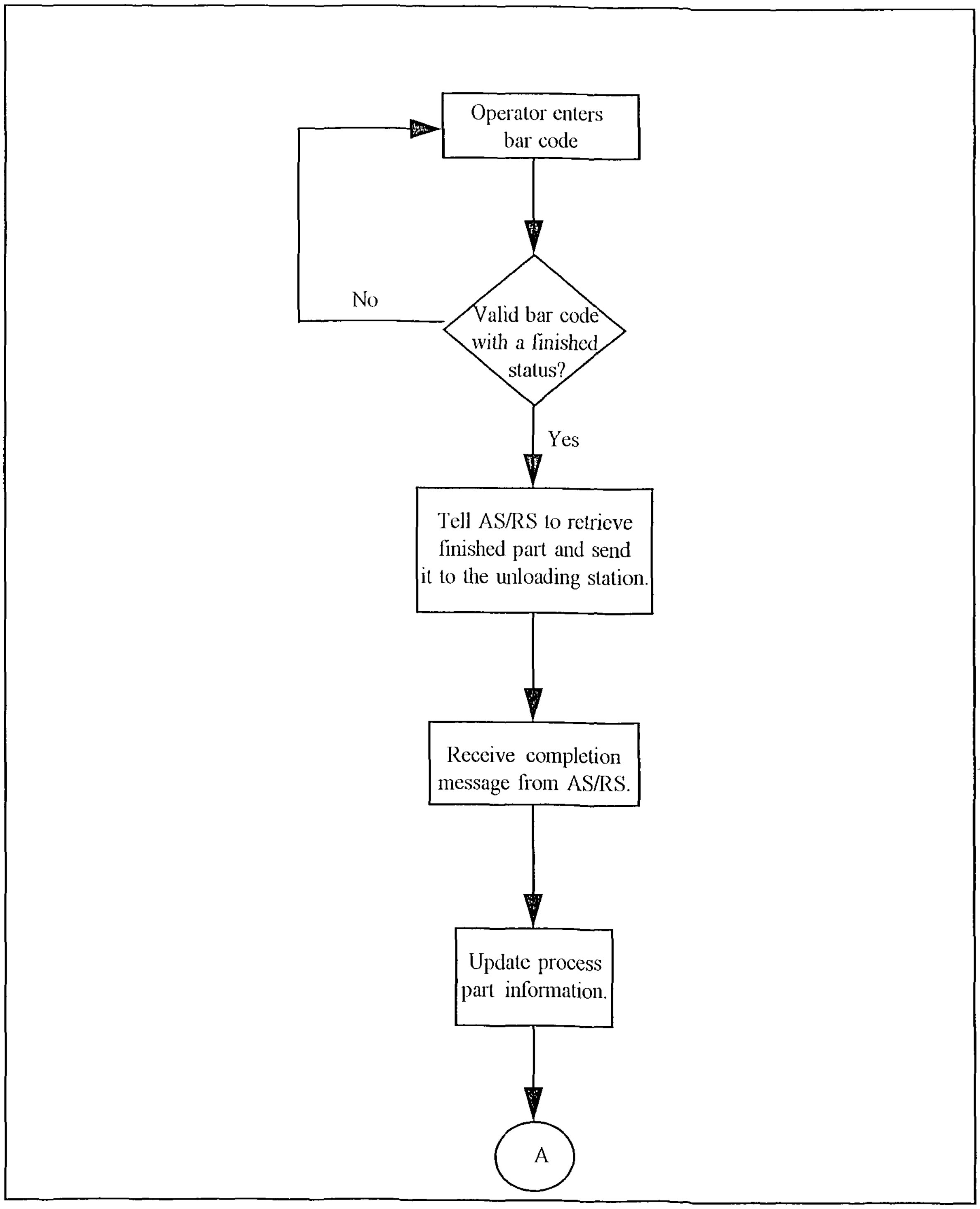

Operator enters bar code
Valid bar code with a finished status?
No
Yes
Tell AS/RS to retrieve finished part and send it to the unloading station.
Receive completion message from AS/RS.
Update process part information.
A

Figure 4.2g Flowchart of Cell Controller - Shipping a Part

material if it already existed in the database. This ensured that information on all parts used in the system was in the database.

The process database contained information on parts that had been processed, such as work-in-process (WIP), finished, and shipped parts. This information included bar code, part number, description, status, number of operations completed, last date processed or shipped, and the identification numbers of the machines used during processing and the processing date. Since only parts existing in the part database were used in the system, the part number in the process database must first exist in the part database.

The cell controller's main menu provided the operator with five options: (1) view part information, (2) view process part information, (3) change part information, (4) process a part, or (5) exit. Viewing part information provided the operator with information from the part database. Similarly, viewing process part information provided information from the process database. In changing part information, the operator could either change information on existing parts in the database or add a new part to the database. For existing parts, the operator could change the description, bill of material, or routing, but he could not change the part number, which is assigned by the controller when the part is added to the database. While the operator could change generic information on parts (part database), the cell controller did not provide an option to change information on processed parts (process database) because that information was obtained during the processing of a part and changes to it could cause inaccuracies in the database. The risks of changing the information on processed parts seemed to outweigh the benefits.

4.1.2 Processing a Part

The operator could choose to either *start processing*, *continue processing*, or *shipping* a part. *Start processing* removed raw material from the AS/RS, processed the part until completion (finished part) or until told to stop (WIP part), and returned the part to storage in the AS/RS. *Continue processing* was similar to start processing, except a WIP part was removed from the AS/RS rather than a raw material. Finally, *shipping* a part removed a part from the AS/RS and marked it shipped in the process database.

In order to start processing a part, the operator entered its part number. First, the cell controller asked the AS/RS controller if each component in the part's bill of material was available in the AS/RS. If all components were available, the cell controller would proceed to process that part. Otherwise, if a component was missing, the cell controller would attempt to make the component part, asking the AS/RS if all its BOM components were available. This continued until all components were available or the missing component was the purposed part. If a purposed part was missing, the cell would stop processing the part requested by the operator notifying him of the missing part. Otherwise, if a part was missing that was made in house, the cell controller would proceed to make the missing part. After the component part was processed, the controller would go back to the higher assembly and re-ask the AS/RS if all components were available.

If all components of a part were available in the AS/RS, the cell controller would proceed to retrieve the parts. A part was never processed unless all of its components were available and until they were all retrieved. All but the last component were retrieved from the AS/RS, sent to the first station in the routing, unloaded from the cart, and their empty carts sent back to the AS/RS for storage. The last component was retrieved in a similar manner, except its cart remained at the first station and was used to hold all of the components and finally the processed part.

First, the cell controller would tell the AS/RS controller to retrieve each component using the component's part number. The AS/RS would retrieve the tote from storage and placed it on the transfer device. After the AS/RS controller had notified the cell controller that a part had been retrieved, the cell controller would tell the cartrac controller to do the following operations for all but the last BOM component: (1) transfer the cart from the AS/RS to cartrac, (2) move the cart to the first station in the routing, (3) return the empty cart to the transfer device (occurs after operator has indicated that the part has been unloaded), and (4) transfer the cart from the cartrac to the AS/RS. For each of the above operations, the cell controller would send a message to the cartrac controller requesting the operation. After receiving a message from the cartrac controller indicating that the operation was completed, the cell controller would proceed to the next operation.

If the component was the last in the BOM, operations 3 and 4 would be omitted because the cart would remain at the first station.

After all components were pulled, the cell controller would assign a bar code number to the part, which remained at the first station until processing was completed. The cell controller would notify the operator that the operation was complete. Unless no more operations existed in the routing, the cell controller would ask the operator if processing should be stopped, returning the part to storage as WIP, or if processing should continue. If the operator decided to continue processing the part, the cell controller would send a message to the cartrac controller to move the cart to the next station. After the cartrac controller had sent a message to the cell controller that the move was complete, the cell controller would simulate machine processing at the second station. Again, after machine processing was complete, the cell controller would give the operator the option of storing the part as WIP or continue processing. This loop would continue until processing of the part was complete or the operator had chosen to send the part to storage as WIP.

Once the part was ready to be returned to storage, the cell controller would issue a command to the cartrac controller to send the cart to the transfer device. After receiving a message from the cartrac controller that the move was complete, the cell controller would tell the cartrac controller to transfer the cart from the cartrac to the AS/RS. Once transferring was complete as indicated from the cartrac controller, the cell controller would send a message to the AS/RS controller to store the cart. After identifying a tote location with no cart, the AS/RS controller would move the cart from the transfer device to that location. Finally, the cell controller would automatically compile the processing information and store that information in the process database.

Continue processing was very similar to *start processing*. Instead of entering a part number, the operator would enter a bar code number. The cell controller would ask the AS/RS controller if a part with the provided bar code existed in the AS/RS. If the part was in the AS/RS, the cell controller would tell the AS/RS controller to retrieve the part. After the part was retrieved, the cell controller would send a sequence of two messages to the cartrac controller to transfer the cart from the AS/RS to the cartrac and move the cart to the next station in the routing, waiting until after the first operation was complete to send the second message. Processing continued exactly the same as in *start processing*.

Shipping a part involved removing the part from storage in the AS/RS and updating the process database. First, the operator would enter the bar code of the part to be shipped. The cell controller would ask the AS/RS controller if the part existed in the AS/RS. If the part did exist, the cell controller would send a message to the AS/RS controller to retrieve the part and send it to its unloading station. The unloading station, usually referred to as home, was located on the north end of the AS/RS. Once the AS/RS controller notified the cell controller that it had completed the operation, the cell controller would notify the operator of completion and update the process database by changing the status to shipped and the date to the date of shipment.

4.2 Messages to/from AS/RS and Cartrac Controllers

In establishing communications between controllers, messages were developed to send the necessary information to the controllers. Table 4.1 shows the messages between the cell controller and AS/RS supervisory controller. Table 4.2 shows the messages between the cell controller and cartrac supervisory controller. As indicated in the tables, each message had a different meaning and many of the messages contained several pieces of information. For example, the cell controller would send message SW1122102111 to the AS/RS controller in order to store a part. This provided the AS/RS controller with the following information about the part to be stored: status is WIP, part number is 1122, bar code is 10211, and quantity is 1. The AS/RS controller used this information to perform the required operation and update its own databases.

The messages were grouped according to an instruction number. Each group was used to accomplish a particular kind of operation. For example, instruction number three in Table 4.1 was used to retrieve a part from the AS/RS. The cell controller sent the message P1122 to the AS/RS controller, telling the AS/RS to retrieve a tote with part number 1122. The AS/RS controller would respond with either R10211 (assuming the bar code is 10211) if the part had been retrieved successfully, EP if the pick could not be

Table 4.1 Messages between Cell Controller and AS/RS Controller

Instruct Number	From/To *	Description	Message
1	A to C	In automated mode	A
		In manual mode	M
2	C to A	Tell me if part number XXXX is in AS/RS	AXXXX
	A to C	Part number XXXX is in AS/RS as finished part	FXXXX
		Part number XXXX is in AS/RS as WIP part	WXXXX
		Part number XXXX is not in AS/RS	NXXXX
3	C to A	Pick a finished part with number XXXX and bring to transfer device	PXXXX
	A to C	Retrieved part; bar code of part is YYYYY	RYYYYY
		Error; cannot execute/complete pick	EP
		Error; part is not in AS/RS or is a WIP part	EN
4	C to A	Pick a WIP part with bar code YYYYY and bring to transfer device	BYYYYY
	A to C	Retrieved part; bar code of part is YYYYY	RYYYYY
		Error; cannot execute/complete pick	EP
		Error; part is not in AS/RS	EN
5	C to A	Store empty tote	SE
		Store raw material: P/N XXXX and Qty Z	SRXXXXZ
		Store WIP part: P/N XXXX, Bar YYYYY, Qty Z	SWXXXXYYYYYZ
		Store finished part: P/N XXXX, Bar YYYYY, Qty Z	SFXXXXYYYYYZ
	A to C	Storage complete: P/N XXXX	SXXXX
		Storage complete - empty tote	S1
		Error; cannot execute/complete storage	ES
6	C to A	Ship part: Bar code YYYYY	UYYYYY
	A to C	Shipment complete: Bar code YYYYY	CYYYYY
		Error; cannot execute/complete shipment	EU
		Error; part is not in AS/RS	EN
7	C to A	Go to main menu (manual mode)	H

* C = Cell Controller
A = AS/RS Controller

Table 4.2 Messages between Cell Controller and Cartrac Controller

Instruct Number	From/To *	Description	Message
1	R to C	In automated mode	A
		In manual mode	M
2	C to R	Send cart number C to transfer device	SCT
	R to C	Move complete, cart C	MC
		Error; cannot move cart	ES
3	C to R	Transfer from cartrac to AS/RS	TCA
		Transfer from AS/RS to cartrac	TAC
	R to C	Transfer complete	TC
		Error; cannot complete transfer	ET
4	C to R	Send cart C to station Y	SCY
	R to C	Move complete, cart C	MC
		Error; cannot move cart	ES
7	C to R	Go to main menu (manual mode)	GTM

* C = Cell Controller
R = Cartrac Controller

executed because an error had occurred, or EN if the pick could not be executed because the requested part was not in the AS/RS. The cell controller waited for a response from the AS/RS controller before continuing.

4.3 Modification of the AS/RS Controller

Incorporating the AS/RS into the flexible manufacturing cell meant controlling it from the cell controller. Since the AS/RS supervisory controller had been designed to completely control the AS/RS, its software design had to be modified to allow the cell controller to control the AS/RS through the supervisory controller. Adding a new mode called Automated to the supervisory controller gave the operator the option of either controlling the AS/RS from the AS/RS supervisory controller, as is currently done, or from the cell controller.

If the operator chose Automated Mode from the Main Menu, the AS/RS supervisory controller would send a message to the cell controller indicating that it was in automated mode. Then, the supervisory controller waited until the cell controller sent an instruction. As indicated in Table 4.1, the cell controller could send an instruction to do one of the following: identify the availability of a part, retrieve a finished part, retrieve a WIP part, store a tote, ship a part, or go to manual mode (main menu). After completing the required task, the AS/RS controller sent a reply to the cell controller and waited for the next instruction. This scenario repeated until the cell controller told the AS/RS controller to go to manual mode. Figure 4.3 shows the flowchart of the software additions to the AS/RS supervisory controller.

If the cell controller asked the AS/RS controller if a part was available in the AS/RS, the AS/RS controller would look for the part in the inventory database. The inventory database stored the following information for each tote location in the AS/RS: tote number, part number, quantity, tote coordinates (x, z, y), bar code number, status (raw material, WIP, or finished), tote owner, and last date into storage. If the part was located in the AS/RS, the AS/RS supervisory controller would send a message to the cell controller stating that the part was in the AS/RS and providing its status (finished or WIP). If the AS/RS controller did not find the part, it would tell the cell controller that the part was not in the AS/RS.

If it received a message to retrieve a finished part, the AS/RS supervisory controller would search the inventory database for a tote with the part number indicated in the message and a finished status. The AS/RS controller would retrieve the first tote located in the AS/RS by moving to the tote location, loading the tote onto the machine, moving to the transfer device location, and putting the tote on the transfer device. If successful, the AS/RS controller would send a message to the cell controller that part had been retrieved, identifying its bar code. If unsuccessful, the AS/RS controller would send an error message stating that it could not execute the pick or that the part was not in

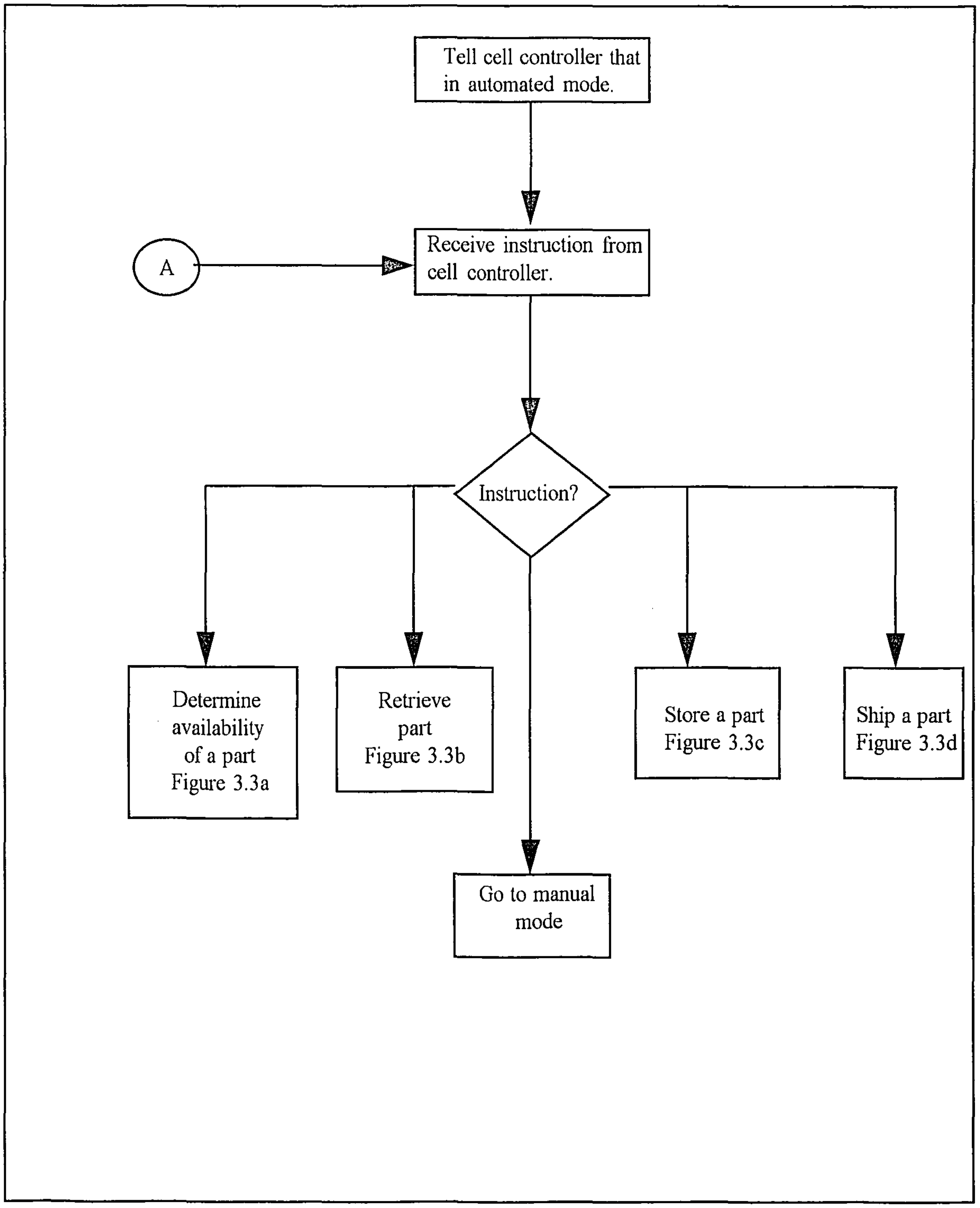

Tell cell controller that in automated mode.
A
Receive instruction from cell controller.
Instruction?
Determine availability of a part Figure 3.3a
Retrieve part Figure 3.3b
Store a part Figure 3.3c
Ship a part Figure 3.3d
Go to manual mode

Figure 4.3a Flowchart of AS/RS Controller - Automated Mode

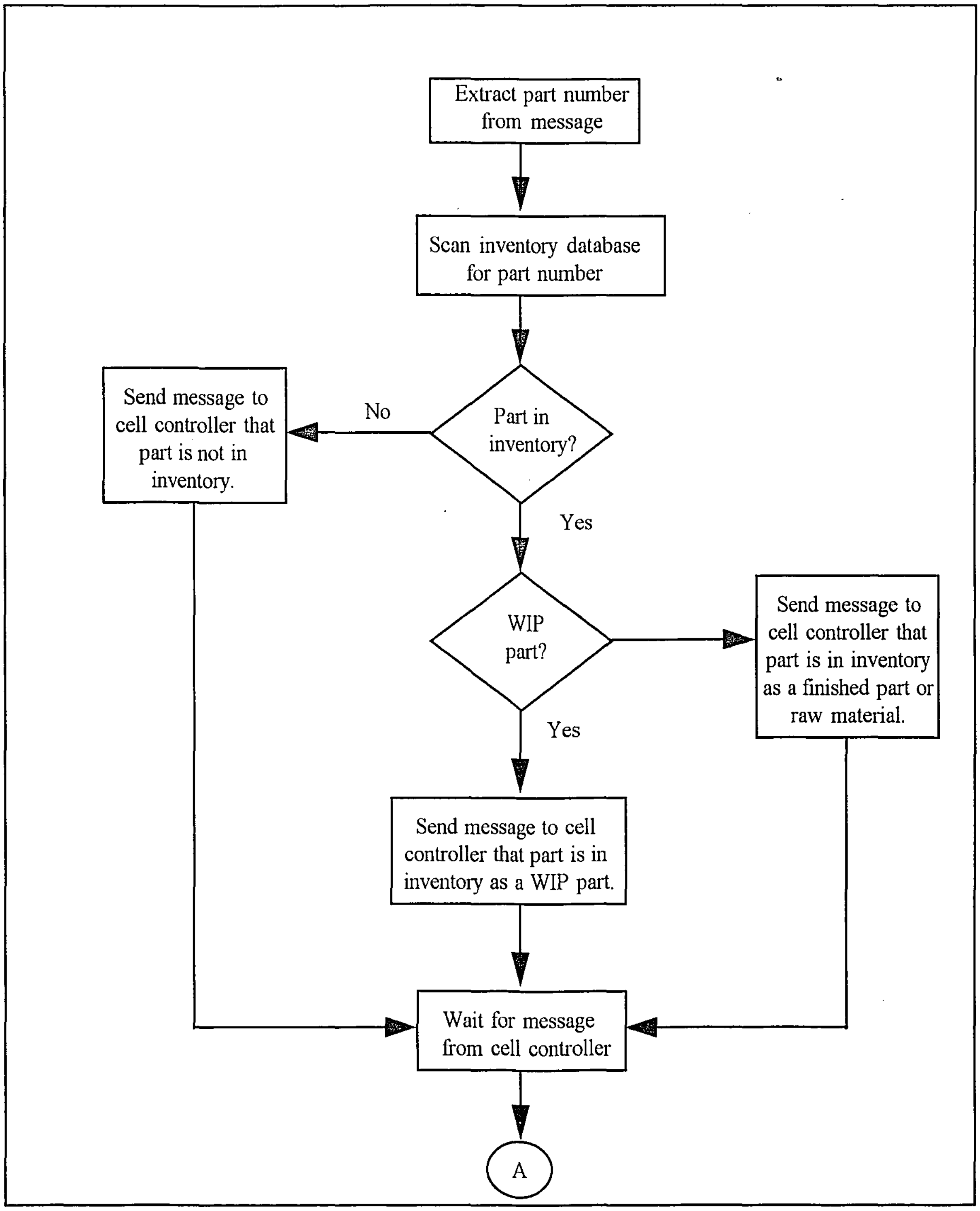

Extract part number from message
Scan inventory database for part number
Part in inventory?
No
Send message to cell controller that part is not in inventory.
Yes
WIP part?
Send message to cell controller that part is in inventory as a finished part or raw material.
Yes
Send message to cell controller that part is in inventory as a WIP part.
Wait for message from cell controller
A

Figure 4.3b Flowchart of AS/RS Controller - Determine Part Availability

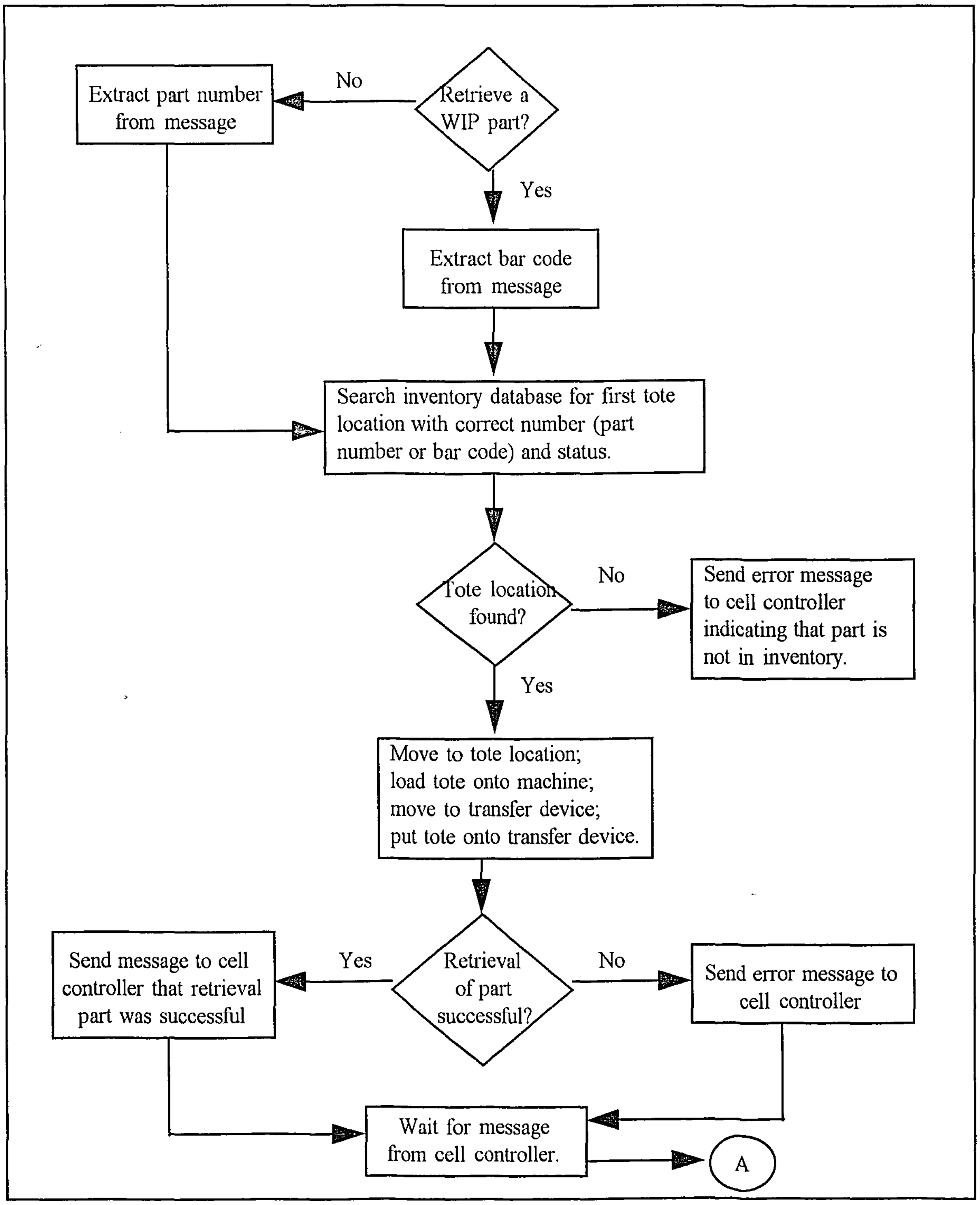

Extract part number from message
No
Retrieve a WIP part?
Yes
Extract bar code from message
Search inventory database for first tote location with correct number (part number or bar code) and status.
Tote location found?
No
Send error message to cell controller indicating that part is not in inventory.
Yes
Move to tote location; load tote onto machine; move to transfer device; put tote onto transfer device.
Send message to cell controller that retrieval part was successful
Yes
Retrieval of part successful?
No
Send error message to cell controller
Wait for message from cell controller.
A

Figure 4.3c Flowchart of Cell Controller - Retrieve Part

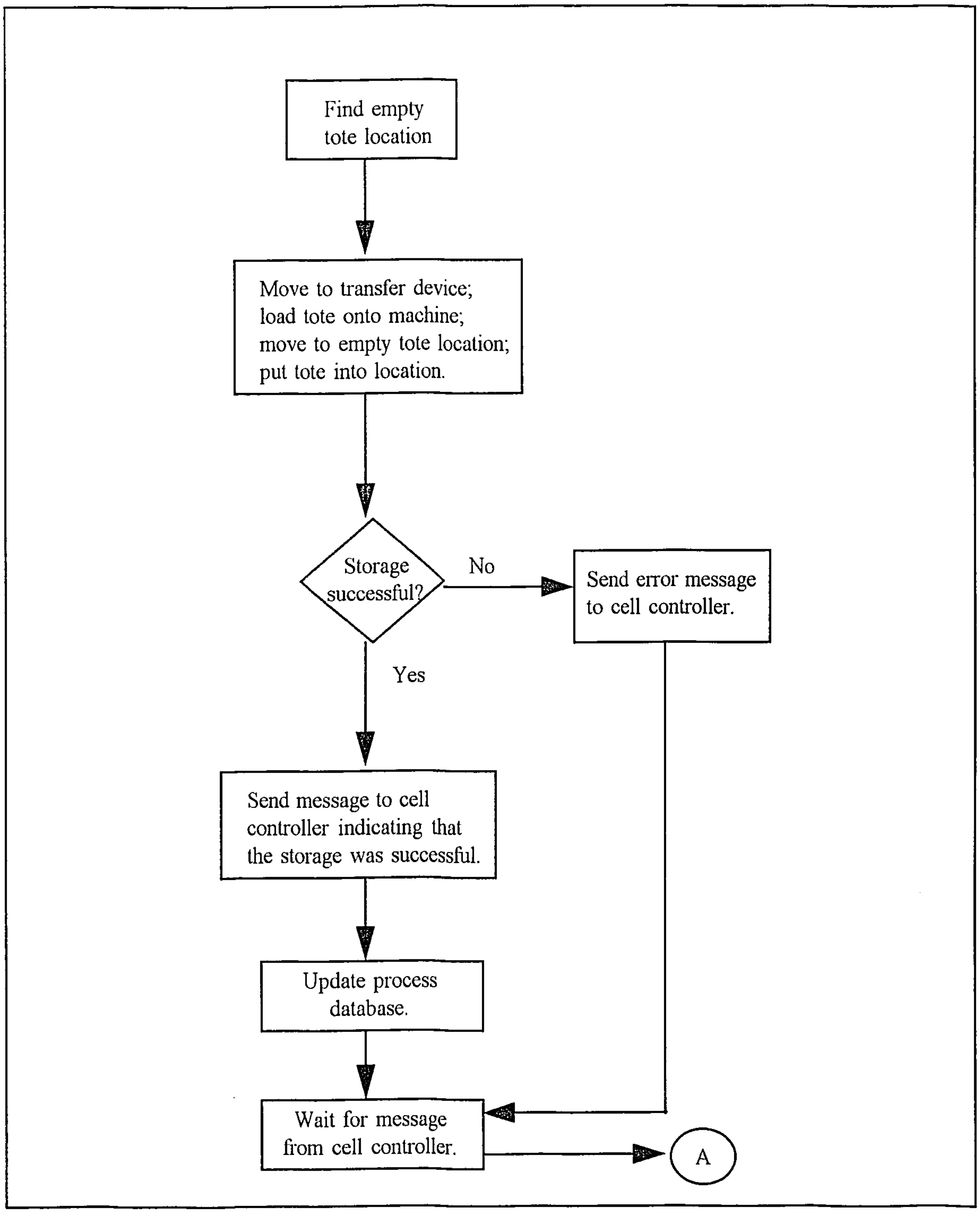
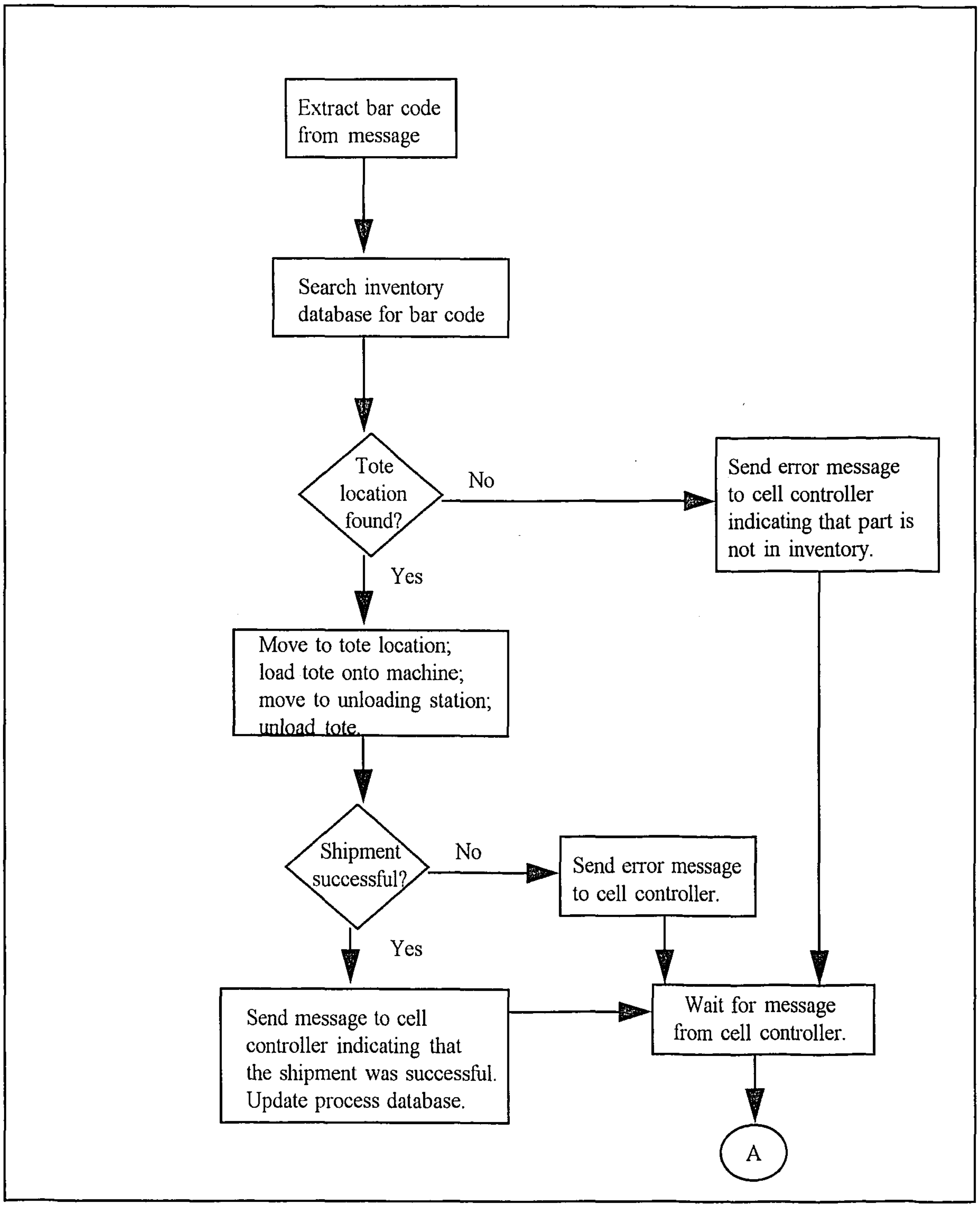

Find empty tote location
Move to transfer device; load tote onto machine; move to empty tote location; put tote into location.
Storage successful?
No
Send error message to cell controller.
Yes
Send message to cell controller indicating that the storage was successful.
Update process database.
Wait for message from cell controller.
A

Figure 4.3d Flowchart of AS/RS Controller - Store a Part



Extract bar code from message
Search inventory database for bar code
Tote location found?
No
Send error message to cell controller indicating that part is not in inventory.
Yes
Move to tote location; load tote onto machine; move to unloading station; unload tote.
Shipment successful?
No
Send error message to cell controller.
Yes
Send message to cell controller indicating that the shipment was successful. Update process database.
Wait for message from cell controller.
A

Figure 4.3e Flowchart of AS/RS Controller - Ship a Part

inventory as a finished part. A request to retrieve a WIP part was handled similarly, except the part's bar code was used to locate it rather than its part number.

The cell controller could tell the AS/RS controller to store an empty tote, raw material, WIP part, or finished part. The message from the cell controller would identify the type of storage and the information required for the inventory database. For example, the AS/RS controller needs access to the bar code of a finished or WIP part in order to ship a part or retrieve a WIP part. Therefore, the inventory database contains bar codes of those parts and the cell controller would send that information when requesting their storage. In order to store a tote, the AS/RS would search for an empty tote location, move to the transfer device, load the tote on the machine, move to the empty tote location, and put the tote in that location. The AS/RS supervisory controller would send a message to the cell controller when the storage was successfully completed or an error had occurred.

In order to ship a part, the AS/RS supervisory controller would search the inventory database for the bar code provided in the cell's message. Once located, the AS/RS controller would move to the tote's location, load the tote onto the machine, move to the unloading station, and unload the tote. Upon completion, the AS/RS supervisory controller would send a message to the cell controller and update the inventory database.

4.4 Modification of the Cartrac Controller

The software design of the cartrac controller also had to be modified in order to incorporate the cartrac into the flexible manufacturing cell. Similar to the AS/RS controller, a new mode called Automated was added to the cartrac controller. Under this mode, the cartrac controller would receive routing information from the cell controller.

After executing the program, the cell controller asks the operator which mode - automated or manual. If automated mode was chosen, the cartrac controller notifies the cell controller that it is in automated mode. The cartrac controller proceeds to wait for an instruction from the cell controller. Referring to Table 4.2, the cell controller could tell the cartrac controller to send a cart to the transfer device, send a cart to a station, transfer a cart from the AS/RS to the cartrac, transfer a cart from the cartrac to the AS/RS, or go to manual mode. After the cell controller sends an instruction, the cartrac controller sends the lower level commands to the PLC in order to complete the operation requested by the cell controller. After the operation is completed, the cartrac controller sends a message to the cell controller that the operation was completed. Then, the cartrac controller waits for the next instruction from the cell controller. This process continues until the cartrac controller switches to manual mode.

4.5 Error Handling

In attempting to process a part, the cell controller could experience two kinds of errors: communication errors and execution errors. Communication errors arise when one of the controllers (cell, cartrac, or AS/RS) does not receive a message expected from another controller or receives a message that was not correct or valid. On the other hand, execution errors result when an error occurred while the AS/RS or cartrac was attempting to complete an operation requested by the cell controller.

The controllers were designed to wait 60 seconds for a message. After 60 seconds, the controller would indicate to the operator that communications are taking longer than expected and allow the operator to interrupt communications, causing a communication error to occur. Also, a communication error would immediately occur if an invalid message was received. If an error did occur, the controller would stop its operation and switch to manual mode or its main menu.

An execution error would occur if the AS/RS or cartrac controller experienced an error while processing an operation causing the operation to be stopped. As a result, the AS/RS or cartrac controller would send a message to the cell controller indicating that an execution error had occurred. The cell controller would stop the part's processing by instructing each controller to switch to manual mode and notifying the operator of the error. The cell controller would also switch to its main menu.

4.6 Communications

A small local area network called Lantastic was used to provide communications between the cell controller and the other two controllers. The cell controller sent and received messages to/from the AS/RS and cartrac supervisory controllers using the LAN. Four files residing on the C drive of the cell controller were used as message files to write messages to and read messages from. The cell controller communicated with the AS/RS controller using files CTOA.CMM (cell to AS/RS) and ATOC.CMM (AS7RS to cell), by writing messages to CTOA.CMM and reading messages from ATOC.CMM, while the AS/RS did the reverse - wrote to ATOC.CMM and read from CTOA.CMM. Similarly, the cell controller used files CTOR.CMM (cell to cartrac) and RTOC.COMM (cartrac to cell) to communicate with the cartrac controller. The LAN provided the AS/RS and cartrac controllers with the ability to read from and write to the files located on the cell controller's C drive.

Since two controllers had access to each message file, coordinating the communications was necessary to ensure that a controller did not read or write to a file while another controller was reading or writing to that file. In other words, the system must ensure that a file was not read from or written to until that file was ready. A message file was always written to after it was read from. There were never two reads in a row or two writes in a row. One controller wrote to the file, then another controller read from the file, then the first controller wrote to the file, and so on.

The above sequence of reads and writes made control of the communications possible using a file attribute called the archive bit. Normally, DOS will set a file's archive bit when the file has been modified, although a software application can set and reset the archive bit. The following algorithms were developed and used in order to control the file's access:

Writing to a file:

1. Wait till archive bit = 1
2. Write to file
3. Reset archive bit to 0

Reading from a file:

1. Wait till archive bit = 0
2. Read from file
3. Set archive bit to 1

A file was ready to be written to when its archive bit was one. After it wrote to a file, the controller would reset the file's archive bit to zero, indicating that the file was ready to be read from. After it had read from a file, the controller would set the archive bit to one, indicating that the file could be written to.

The above algorithms ensured that a controller would never overwrite a message in a file unless that message had already been read by the other controller. Also, two controllers would never have access to the same file at the same time because the archive bit would either be set or reset, never both. Since the AS/RS controller and cartrac controller first sent messages to the cell controller indicating that they were in automated mode, the cell controller set the archive bits of files ATOC.CMM and RTOC.CMM at the beginning of its execution. Therefore, the cell controller must be started before the AS/RS and cartrac controllers are put in automated mode. If this was not done, it was possible that the cell controller would set the archive bit after the AS/RS and cartrac controllers had already sent the messages and reset the archive bits. This would prevent the cell controller from ever reading the messages from the other controllers, causing a communication error to occur.

Chapter 5 Conclusion

5.1 Summary of Current Work

Flexible manufacturing systems have become an important tool in manufacturing efforts to improve productivity, reduce costs, and satisfy diverse customer wishes. The cell controller is the heart of any FMS, integrating the functions of the different system elements and determining its flexibility. Therefore, the design of the cell controller will determine the FMS's success. This thesis describes the software design of the cell controller of an FMS in the ManTech Laboratory at Lehigh University.

The cell controller integrated the cartrac system and AS/RS, creating the cell's material handling system. This design provided a hierarchy of control to the flexible manufacturing cell. In the manufacture of a product, the cell controller coordinated the necessary actions by sending and receiving high level messages to/from the cartrac and AS/RS supervisory computer. The two supervisory computers, each controlling a sub-system of the cell, communicated with their respective PLCs using lower level messages. The lowest level of control consisted of the PLC directly controlling the hardware in its system by providing output to output devices based on inputs and low level messages it had received.

High level messages were sent between the cell controller and two supervisory controllers over a local area network. Message files residing on the C drive of the cell controller were used to write messages to and read messages from. Two controllers had access to each message file. Therefore, controlling the communications was necessary to ensure that a controller did not read or write to a file while another controller was reading and writing to the same file.

In order to ensure that messages were never overwritten unless they had already been read by the receiving party, a message file was always read from after it was written to. The above scheme was implemented using a file attribute called the archive bit. A controller could only read a file if its archive bit was zero. After reading the file, the controller immediately set its archive bit to one. The second controller would wait until the archive bit was one to write to that file. After writing to the file, the controller would set its archive bit to zero, indicating to the first controller that a new message was ready to be read.

Information is imperative in a controllers ability to make decisions. Several databases were developed to support the flexible manufacturing cell. A part database stored generic information on parts, such as part number and bill of material. A process database contained information on the processing of parts. The AS/RS controller also used an inventory database, containing information on parts stored in the AS/RS.

5.2 Contributions of this Research

In summary, the contributions of this research are as follows:

1. Software design of a cell controller for a flexible manufacturing cell.
2. The cell controller integrates the AS/RS and cartrac systems into a fully automated material handling system for the flexible manufacturing cell.
3. The design uses a hierarchy of control, partitioning the control problem and therefore, limiting the complexity of any one controller in the system. Also, this minimized modifications to the existing control systems.
4. The development of a series of messages communicating information and instructions between the cell controller and the AS/RS and cartrac supervisory controllers.
5. Two communication algorithms provide for control of the communications, ensuring that message files are only accessed by one controller at a time. Also, message files are only read from when new messages are present and only written to after its contents have been read.
6. The cell controller provides a good easy-to-use human interface. The system is menu-driven, user-friendly, and provides necessary system information to the user.
7. The equipment in the cell can be used in either an integrated manner in the cell or in a stand-alone manner, operating independently from the cell.
8. The ability to produce machined parts and assemblies in the cell.

5.3 Directions for Future Work

Currently, the cell controller integrates the cell's material handling system, consisting of the AS/RS and cartrac. Other automated pieces of equipment, such as machine tools, need to be linked to the cell controller, creating a complete flexible manufacturing system. The cell controller presently simulates the processing of parts on machines. Some of the machine tools in the ManTech Laboratory, which can be integrated into the FMS, include the K&T milling center and CNC turning center.

The cartrac control system only provides the capability of operating a single cart on the cartrac at a time. Therefore, the cell is limited to a single cart and can only manufacture one part at a time. Future work should include modification to the cartrac control system, allowing for multiple carts on the cartrac. The high level messages between the cell controller and cartrac supervisory controller already provide communication on the cart number.

Finally, the cell controller does not perform production planning and scheduling tasks. The current system asks the user to identify the parts to manufacture. Future work should provide the cell controller with the enhancements necessary to make planning and scheduling decisions rather than relying on humans to perform those tasks.

Bibliography

Greenwood, N.R., "FMS - Control and Communications. The Problems and the Potential," Proceedings of the 5th Conference on Flexible Manufacturing Systems, 3-5 November 1986, pp. 1-7.

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

Haritos, G., Butler, M., King, P., and Ghosh, S., "Software Generation of Manufacturing System Control," Factory Automation and Information Management, CRC Press, Inc., Boca Raton, Fl, 1991.

Howard, Ron, "PCs and PLCs Duel for Control", InTech, November 1989, pp. 38-41.

Hunt, V. Daniel. Computer-Integrated Manufacturing Handbook, Chapman & Hall, New York, NY, 1989.

Hurson, A.R. and Pakzad, Simin, "Object-Oriented Database Management Systems: Evolution and Performance Issues," Computer, Feb 1993, pp. 48-60.

Johansson, Mikael, "Communications Issues in Manufacturing," CIM Review, Fall 1989, pp. 37-44.

Levin, L., Fielding, E., and Ackhurst, K., "Developing a Computer Control System for a Flexible Manufacturing Cell," The International Journal of Flexible Manufacturing Systems, Vol. 5, 1993, pp. 143-159.

Lin, L., Wakabayashi, M., and Adiga, S., "Object-Oriented Modeling and Implementation of Control Software for a Robotic Flexible Manufacturing Cell," Robotics and Computer Integrated Manufacturing, Vol. 11, No. 1, 1994, pp. 1-12.

MacDow, R. and Dirk, A., "Managing Multiple Databases in CIM," Advanced Manufacturing Systems, IFS (Publications) Ltd and Springer-Verlag, New York, NY, 1986.

McKeown, Patrick G. and Leitch, Robert A. Management Information Systems: Managing with Computers, Harcourt Brace Jovanovich, Inc., 1993.

Mollo, M., "A Distributed Control Architecture for a Flexible Manufacturing System," Flexible Manufacturing Systems, IFS (Publications) Ltd and Springer-Verlag, New York, NY, 1985.

Murphy, Pat, "PCs and PLCs: Partners in Control," InTech, July 1988, pp. 33-36.

Petruzella, Frank D. Programmable Logic Controllers, McGraw-Hill Book Company, New York, NY, 1989.

Popovic, Dobrivoje and Bhatkar, Vijay. Distributed Computer Control for Industrial Automation, Marcel Dekker, Inc., New York, NY, 1990.

Rembold, U., Blume, C., and Dillmann, R. Computer-Integrated Manufacturing Technology and Systems, Marcel Dekker, Inc., New York, NY, 1985.

Vita

Diane Livingston was born to Vernon and Mildred Berndt in Brooklyn, New York on April 21, 1965. In 1987, she received her B.S. in Industrial Engineering from Virginia Polytechnic Institute and began her career with Westinghouse Electric Corporation as an industrial engineer. In 1990, she left Westinghouse to work for Rockwell International, leaving in 1993 to pursue a master's degree. After two years as a full-time graduate student, she has returned to Corporate America as a systems analyst with Merck & Company.