
Chapter 2

Computers And Control Systems Within Manufacturing

A Summary...

Multiple-axis motion controllers. Computer Numerical Control Systems. Robotic Control Systems. Programmable Logic Controllers. Computer Control of Manufacturing Systems. Flexible Manufacturing Systems. Integration of Computer Aided Design, Manufacture and Management Information Systems and Inventory Control Systems (MRP).

Read This Chapter If...

- ◆ You need to understand about the manufacturing equipment, computers and systems that need to be integrated through data communications.
- ◆ You would like to come to terms with the "intelligence level" of the various devices that have to be integrated through data communications and networking.

2.1 The Range and Scope of Computers within Manufacturing

Within any modern manufacturing organisation, computers may be used at a number of different levels, including:

- Management / Financial Information Systems
- Production and Inventory Control / MRP / MRPII
- Software Simulation
- Computer Aided Process Planning
- Computer Aided Design and Drafting
- Control of Automated Mechanisms (Programmable Logic)
- Data-acquisition
- Machine Control Systems (Computer Numerical Control)
- Robot Control
- Continuous Chemical Process Control
- Production Line Control (In-line Transfer Machines)
- Flexible Manufacturing System Control.

Perhaps, in an ideal world, it may be reasonable to suggest that a single, powerful, computer could perform all these tasks simultaneously. In such a system, all the meaningful operating data about an entire factory could be centralised and quickly accessed by any one of the active tasks. Since the entire manufacturing system is encased in a single (closed) computer, software and data retrieval could readily be standardised and optimised. Errors from external sources could be reduced to a manageable level. Of course in an ideal world we could probably find a computer flexible enough to perform all these functions efficiently. It would naturally also come with an "iron-clad" lifetime guarantee that clearly stated that the computer would never break down.

As the real world would have it, computers are only designed with a limited range of applications in mind. The scope of activities that take place within a manufacturing organisation is so diverse that it cannot, realistically, be implemented through a single computer. Therefore, through the years of computerisation in manufacturing, we have been endowed with an enormous and diverse range of computers and computer controls. The architecture of these systems is often as diverse as the functions that they perform. However, we have now come to realise that the ability of all these computer applications to share data (or databases) is of crucial importance to manufacturing, in order to:

- minimise times between work orders and production
- minimise inventory and stock levels
- minimise bottle-necking of parts / products in plant
- minimise production / product costs
- minimise design errors and transmission of design errors
- minimise overall response times to changing market demands
- maximise equipment utilisation
- maximise product consistency and quality
- maximise flexibility of production equipment
- prevent unnecessary (repetitive) human entry of data.

The solution then would be to somehow interconnect all of the diverse computer systems within a manufacturing organisation through an "ether" that would allow a wide range of applications to share information. In computer terms the "ether" is referred to as a data communications network. When a network exists within a relatively small area (under 1 kilometre say), then it is more specifically referred to as a Local Area Network or LAN.

The ideal solution would be something like the one shown in Figure 2.1. It illustrates a concept, referred to in data communications, as "Plug-in Compatibility". In other words, a wide range of different (computer-based) devices that can all plug into a central network or "ether".

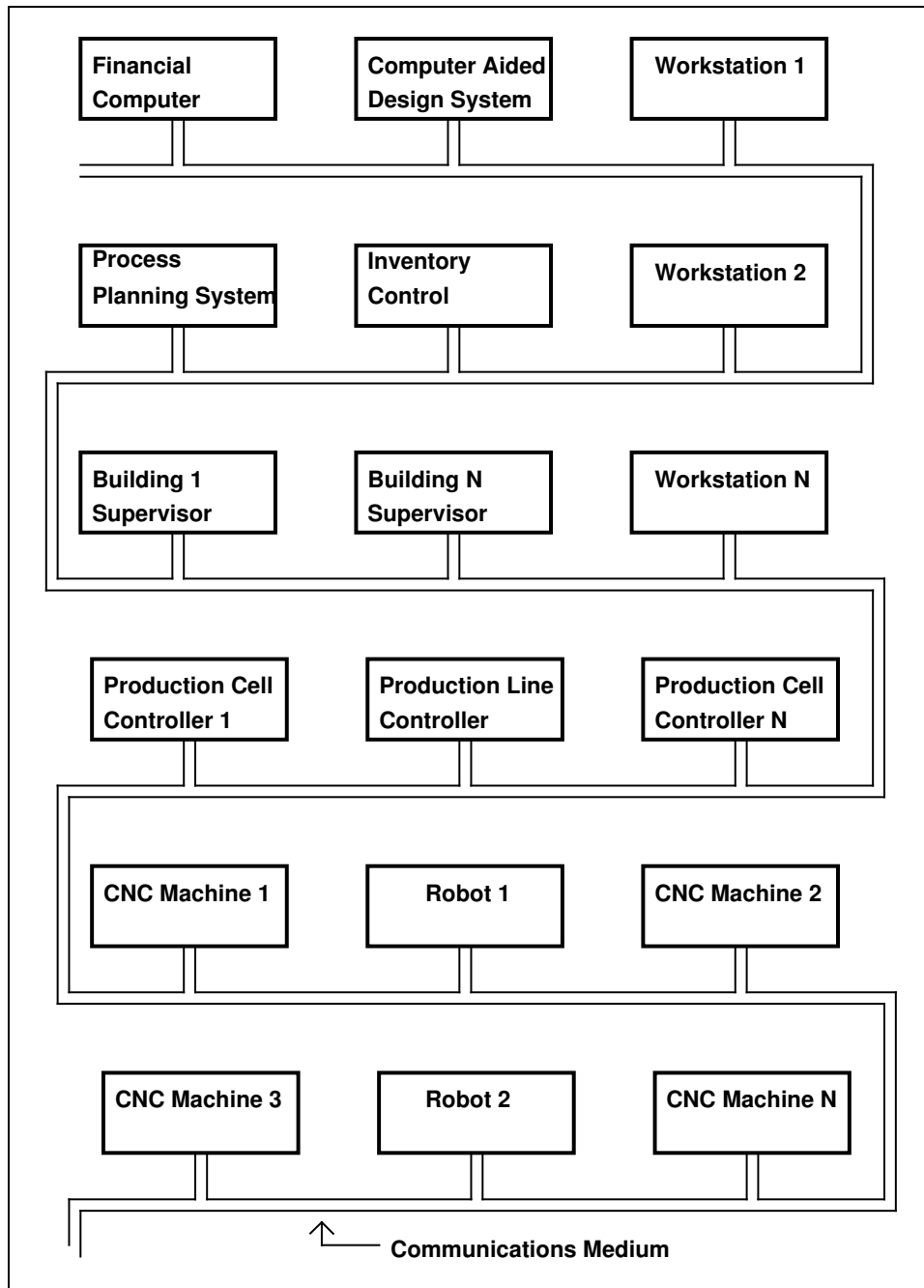


Figure 2.1 - An All-embracing LAN in Manufacturing

Plug-in Compatibility can best be compared to the power supply used in the home. Power supply lines of fixed voltage and frequency (the bus) run throughout a house, with a number of outlets (tapping points) located at convenient locations. Ideally we choose to have more outlets than equipment in order to provide for future expansion and flexibility.

Electrical appliances are designed with an "interface" to the domestic power system. That is, their power supplies and plugs conform to a fixed standard of voltage, frequency and plug design. We can then plug-in appliances to any outlet, provided that they are compatible with the standard.

The obvious problem with the practice of designing systems that are "Plug-in Compatible" is that in order for appliance manufacturers to produce products with a given interface, the bus system must be standard and widely accepted. If every home ran a different voltage and frequency power supply, then appliance manufacturers would simply not find it profitable to produce interfaces for every possible combination.

The power supply system requires only a very simple standardisation (voltage, frequency and plug design) in order to make equipment manufacturers conform. And yet, as we are all aware there is no worldwide standard for general purpose power outlets. Every country has its own standard. However, the problems of non-standard, power supply systems, within a given country, were addressed at a very early stage by government bodies, so that "Plug-in Compatibility" could be achieved.

The numbers of parameters that have to be defined and standardised in terms of data communication and computer networking, between intelligent devices, are enormous in comparison to those associated with power supply. Additionally, we have a situation where data communications and networking have arisen in an ad-hoc manner, largely driven by computer equipment manufacturers with little or no Government intervention in the standardisation process.

We will make a detailed examination of the problems associated with achieving "Plug-in Compatibility" for manufacturing in subsequent chapters. For the moment however, it is necessary to accept that it is desirable, in principle, for all intelligent devices in a factory to be able to exchange information with all other intelligent devices. The practicality of this goal depends largely upon the architecture and flexibility of the intelligent devices within the system. We shall now examine some of the most commonly used devices and systems within manufacturing and assess their communications capabilities and requirements.

2.2 Programmable Logic Controllers

Programmable Logic Controllers (PLCs) are perhaps the most prolific of all modern industrial control systems. They are used for a wide range of applications and are very diverse in their capabilities. A PLC is shown schematically at its simplest level in Figure 2.2.

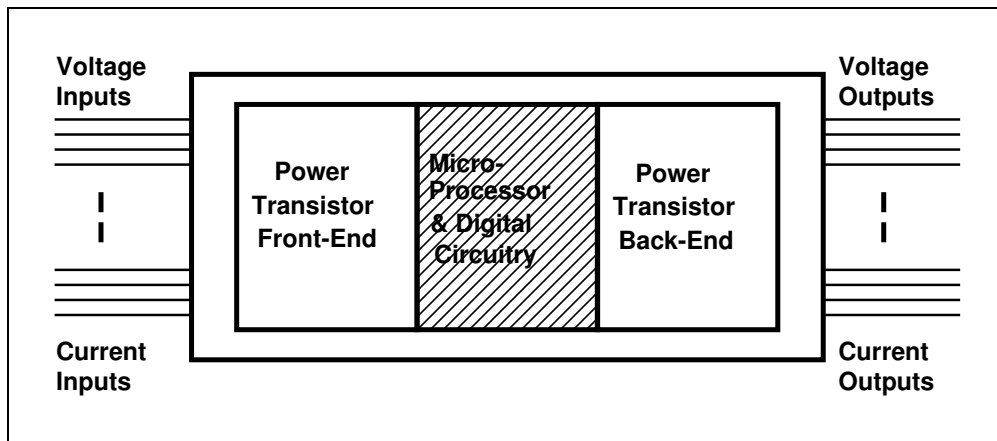


Figure 2.2 - Schematics of a Programmable Logic Controller

PLCs use power-transistor technology, in combination with microprocessors and digital circuitry, in order to produce a specialised computer system for high power switching and control. The power transistor front and back ends, are used to buffer the low-voltage microprocessor computer circuitry from high power industrial inputs and outputs. PLCs therefore provide the ideal combination of a small computer system together with interfacing to the industrial environment.

Initially, PLCs were introduced to simply replace the dated relay-ladder logic systems used to implement Boolean functions on industrial control signals. The following is a typical PLC function:

"If Input A is high and Input B is greater than 50 volts, then delay 10 seconds and then set Output C to High."

The inherent ability of PLCs to perform such functions makes them ideal for sequential control functions where (say) a number of hydraulic and pneumatic actuators and sensors have to be governed. For example, the opening and closing of safety doors or the switching of fluid pumps in a production system.

Modern PLCs are relatively inexpensive items, which are industrially rugged in design and extremely modular in structure. It is commonplace to buy a Central Processing Unit PLC, together with any number of bus-connected Input/Output (I/O) modules. This allows both simple and complex machines to be based upon the same, basic PLC unit. An Original Equipment Manufacturer (OEM) may choose to design a machine using a basic PLC system (with say 10 to 20 inputs and outputs) and then purchase expansion I/O modules as customer, design requirements change.

Programming languages for Programmable Logic Controllers are as diverse as the controllers themselves. Originally, PLCs were only programmable in "**ladder-logic diagrams**" that are a pictorial representation of Boolean circuits coupled with delay and timer elements. This seemed to be a logical step, since it meant that designers, who were used to designing relay circuits, could simply transfer to design using PLCs. However, as people grew to realise the enormous range of design applications for these programmable devices, it became less and less attractive to use the now dated, ladder-logic, diagrams.

Many PLCs were sold with specialised implementations of the BASIC programming language as a built-in feature. This allowed a much more sensible and structured approach to system design to be used. It was also a logical step, since the proliferation of Personal Computers meant that more and more people were comfortable with the concept of programming in a language, rather than using diagrams. As with most technology items, the power and flexibility of the PLCs has grown to meet the ever-increasing range of activities for which they are used. PLCs are now available with specialised "C" or "PASCAL" compilers, that allow complex program development for control applications.

High-resolution, graphic display screens (akin to those available with PCs and Workstations) are now commonplace on Programmable Logic Controllers. Some PLCs have also bridged the gap between computer systems and industrial controllers, through direct connections into the internal bus of some mini-computer systems. This is referred to as "back-plane connection" and is important because older PLCs had to be connected to other computer systems through relatively slow serial communications links.

Traditional workstations, PCs and mini-computers are still a more suitable platform on which to perform large calculations than the PLCs themselves. Therefore it is often necessary for a process control computer to do the "number-crunching" while a PLC does the data-acquisition. In continuous process control applications, serial communications links between a PLC and host computer are too slow to enable closed-loop response times to be achieved. Until the advent of back-plane (bus) connected PLCs, this meant that either a powerful PLC had to do both the number-crunching and data-acquisition or, that a mini-computer had to perform both tasks. Neither was an optimal solution.

Many PLCs however, do not require a high-speed, back-plane connection to the outside world, and can be remotely linked to other computers through a serial communication link. A number of PLCs have a communications task (program), running concurrently with other control programs, so that a remote device can supervise the PLC through the serial link.

The programming languages on some of the more sophisticated PLCs will allow a user to develop serial communications programs, which can link the PLC to the outside world. A wide range of PLCs can also be purchased with special network interface cards and modules that will allow them to be linked into a variety of industrial Local Area Networks. Simple PLCs on the other hand, may be equipped only with specialised development languages and have minimal access to the outside world, through data communications links.

The criteria typically used to select a PLC for a particular application include the following:

- PLC Programming Language
- Number of Inputs and Outputs (I/O capability)
- Expansion Capability
- Processor Execution Speed
- Modularity of Design
- Ruggedness of Design
- Capacity for Integration with other systems through:
 - Serial Communication
 - Back-plane (Bus) Communication
 - Local Area Network Communication.

Overriding these technical factors are always the considerable political (social) factors, which cause a company to choose a PLC vendor based upon conformance with other systems already installed in a plant. This reduces the need for maintenance personnel to become familiar with a wide range of programming and implementation techniques.

2.3 Multiple Axis Motion Controllers (CNC and Robotics)

Many machines and devices within manufacturing consist of little more than a number of servo-motor driven axes, which are used to either position an end-effector (tool) or a work-piece so that the work-piece can be either moved, machined or processed. The most common examples of this are Computer Numerical Control (CNC) machine tools and robots.

Despite the obvious physical differences between, say, an articulated welding robot and a CNC milling machine, the principles behind the control systems are essentially similar. The schematics of CNC and robot control systems are shown in Figure 2.3.

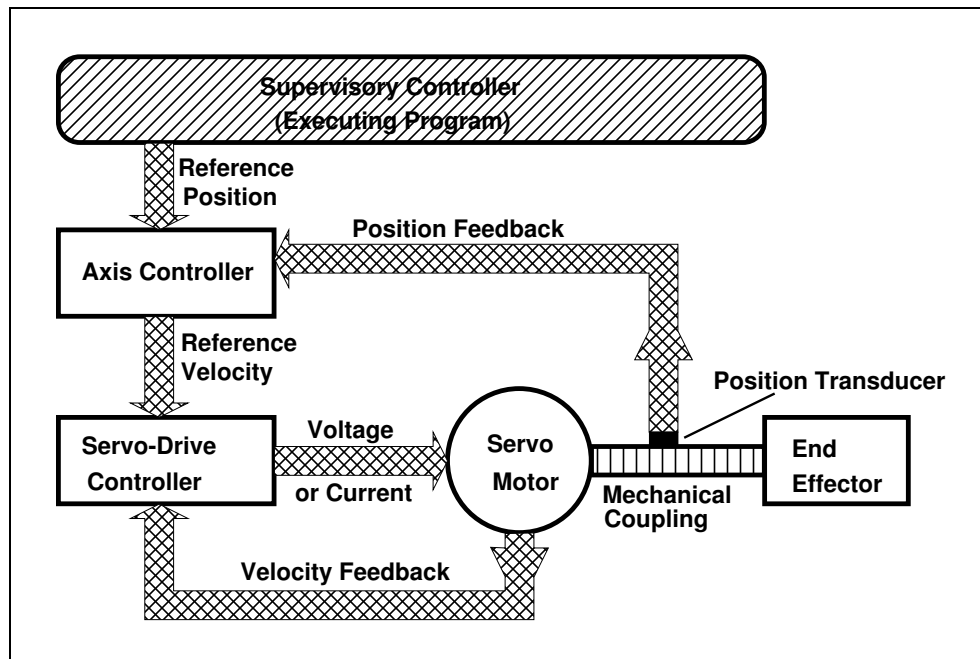


Figure 2.3 - Schematics of an Axis Control System

The supervisory (co-ordinating) controller (processor) in robotic and CNC systems is responsible for a number of simultaneous activities including the user (system) interface and part program parsing and execution. As each step (block) of a part program executes, the supervisory controller passes down positioning information to the axis control computer.

The axis control computer is responsible for achieving the desired position, using appropriate acceleration and deceleration curves. It does this by sending reference velocities to the servo-drive controller, on the basis of the actual position that has been achieved (a negative feedback loop).

The relationship between the servo-drive controller and the servo-motor is dependent upon the type of motor in use. In d.c. systems, the servo-controller varies the current to the field windings of the motor. In both synchronous and asynchronous (induction) motor, variable speed a.c. systems, the servo controller supplies the armature of the motor with a variable frequency supply voltage.

The transducers used to provide velocity and position feedback vary according to the application. Commonly, shaft-mounted tacho-generators are used to provide velocity feedback and linear resolvers or pulse-code transducers provide position feedback.

The processors used in such control systems are relatively powerful devices, capable of running multiple programs simultaneously (multi-tasking). For example, the processor acting as the axis controller, may be capable of co-ordinating 5 or 6 axes simultaneously. Depending upon the complexity of the system, the axis controller and supervisory functions can even be individual tasks running on a single processor.

Both CNC and robot control systems tend to be very specialised in their design. They are optimised to achieve multi-axis control, with minimum user programming and hence the languages that they use tend to be somewhat restrictive.

For historical reasons, related to early hardware limitations, CNC machines are traditionally programmed in a "G-Code" language. This dated system provides a user with a number of sub-programs, commonly prefaced with either a "G", "F", "S", or "T" and suffixed with a subroutine number or parameter. These facilitate the movement of a cutting tool through a predefined path; the selection of a cutting tool and so on. However, few (if any) of these languages will allow a programmer to do more than this.

G-Code languages were never intended to provide the user with routines for accessing various aspects of the machine controller itself and for interfacing it to the outside world. These features, which are now both desirable and important, are difficult or impractical to implement on CNCs. For example, with traditional CNC, it is often difficult to display user programmed screens as a part program executes. Further, it is generally not possible to access the serial communications facilities of a CNC through the G-Code language itself.

CNC designers often attempt to augment the limited features of the G-Code languages by running additional (concurrent) tasks on the CNC. For example, many CNCs have programs, designed to handle serial communications and remote commands, running as tasks, while a part program executes. This type of task is referred to as a Direct Numerical Control or DNC task/facility. It generally enables a host computer to remotely control a CNC machine through a serial link. There are other tasks, such as concurrent, graphic information displays, which are also added by a number of manufacturers.

The deficiency with this form of controller architecture is that it provides a closed (black) box to the end-user. There is normally no mechanism by which an end-user company can re-program it in order to change the graphics displays, or the way in which serial communication occurs. So, while CNCs provide great flexibility in terms of cutting and shaping materials, they generally provide very little flexibility in terms of tailoring the user environment. This is an important point to note in regard to the ability of CNC systems to be linked up to other devices.

Robot controllers are often better than traditional CNC systems in terms of their programming flexibility. Robot controller technology evolved at a later (more opportune) stage of computer development than CNC. Unlike CNC machines, robots seldom, if ever, work as devices in total isolation from other systems. They are generally linked to other computer controlled or logic controlled mechanical systems. For example, a spray-painting robot must be linked to the production line that feeds it with work-pieces for painting - otherwise there can be no inter-locking between line movement and robot cycles.

It is far more difficult to categorise the capabilities of robot controllers, because they are far more diverse in architecture than CNC systems. Some robot-controllers can be programmed in PASCAL or "C" in the same manner as any normal computer system. These systems provide users with a high level of access to the internal hardware of the controller itself. This makes such controllers more amenable to interfacing with the outside world.

Some, "less sophisticated" robot controllers are analogous to CNCs and can only be programmed in restrictive, specialised, movement languages (similar to proprietary G-Codes). These systems suffer from the same interfacing disadvantages as CNC systems. That is, unless a remote-control communications task is available to run on the controller, then it is not possible to provide an intelligent interface to the outside world.

CNC systems and robot controllers generally come with built-in Programmable Logic Controllers, usually of specialised and complementary design. The PLCs are integrated into the CNC or robot control system, under the control of the main processor. They are used to control a range of sundry functions requiring high voltage or current switching. For example, on a CNC machine, the internal PLC may control the switching of coolant pumps, the opening and closing of doors, etc. On a robot, the PLC may control the opening and closing of grippers and the switching of inter-locked equipment. The inputs and outputs of PLCs on both robot and CNC systems are accessible from within the robot programming language or G-Code programming language. This scheme is shown in Figure 2.4.

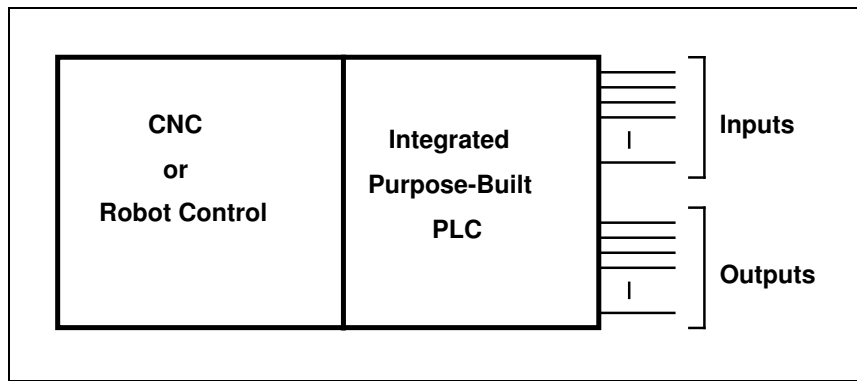


Figure 2.4 - Integrated PLC Control of High Power Peripherals

Both CNC and robot controls are generally provided with a "hard-wire" interface to the outside world. This provides a simple means of integrating the devices into automated systems. In a hard-wire interface, spare inputs and outputs from the integrated PLC are selectively connected to external devices so that they can be inter-locked. Program execution is then made dependent upon the condition of inputs.

For example, if we wished to use a robot to feed a CNC machine with work-pieces, a hard-wire inter-locking arrangement, such as the one shown in Figure 2.5 may be used.

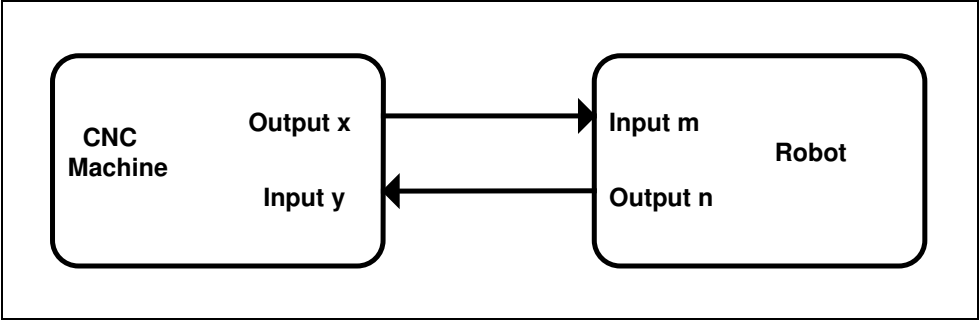


Figure 2.5 - Inter-locking a CNC Machine to a Robot

In such a system, the CNC machine program should start execution as soon as the robot has loaded a part (ie: when the robot program has been completed). The robot should unload a part when the CNC machine program has ended. This can be achieved by the robot setting output number "n" high when it completes a program (last executable line of code) and the CNC machine setting output "x" high when it completes a program (last executable line of code). The first lines of code on both the CNC and robot are to wait for inputs "y" and "m", respectively to go high before continuing. Figure 2.6 shows the form of the two inter-locked programs:

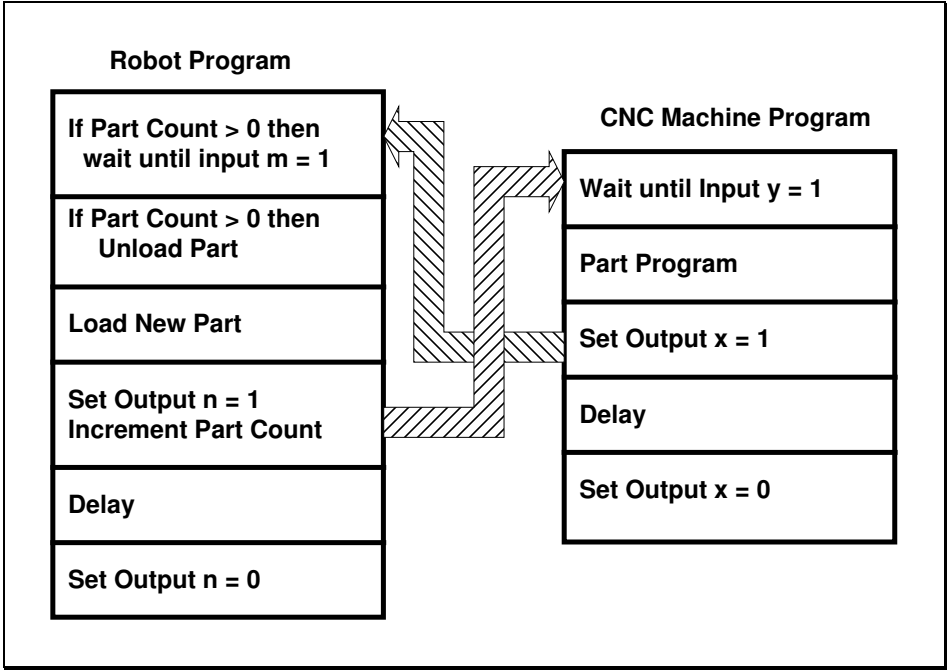


Figure 2.6 - Program Structure for Interlocking Robot and CNC

Note that the robot program needs a "flag" variable to distinguish the first part from the subsequent parts. This is so that it does not wait for a "finished" signal from the CNC machine before the machine has been fed with a part.

This simple, hard-wired form of communication and control has some severe limitations. Firstly, neither of the devices know what the other is doing, while the other device is executing a program. For example, if an unexpected condition arises in the system (say the robot arm positions a part incorrectly or else fails to pick it up), then the CNC machine will execute its machining program without knowledge of the error status. This type of situation could damage the robot arm, the work-piece or the CNC machine itself.

The inability of hard-wired communications to handle abnormal (error) situations, by intelligent interrogation of inter-locked devices, can only be overcome through over-engineering of the mechanical systems to increase safety factors. The more appropriate solution however, is for devices to be able to exchange important status information during program execution (not just before and after) through intelligent communications links.

2.4 Linking Computer Aided Design to Manufacture

Complex CNC machine programs are seldom developed on the machines themselves. They are either developed off-line on a PC based, ASCII word-processor or on some form of Computer Aided Design system. In either event it is necessary to down-load the CNC programs, developed on a remote (host) computer system to the CNC machine. It should also be noted that the CNC machines themselves are seldom equipped with low cost, bulk program storage facilities (such as the hard-disks found on computers). Therefore, for long term storage of programs, developed or modified on the machines, it is also necessary to be able to up-load the files to a host computer.

In the past, program transfer and storage was achieved by the mechanical punching a paper (mylar) tape and physical transfer of the tape from machine to machine. However, file transfers between host computers and CNCs are now most commonly performed through a (serial) data communications link, such as that shown in Figure 2.7.

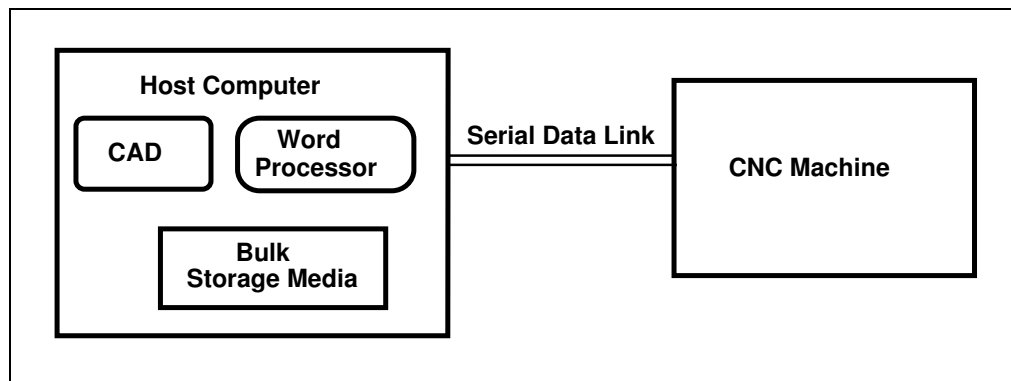


Figure 2.7 - Serial CAD to CAM Links

Maintaining the integrity of program files is of crucial importance in both the up-loading and the down-loading processes. Errors in CNC program files could result in damage to:

- cutting tools
- the machines themselves (servo-drives, etc.)
- valuable work-pieces.

This damage can be caused by corruption of program variables, such as feed-rates, positioning commands, tool-selection codes, etc., during file transfer.

Transmission of large quantities of data through cables (whether conducting or optic fibre) is by no means an error free proposition. Communications are subject to noise, **E**lectro-**M**agnetic **I**nterference (EMI), synchronisation and timing errors on transmitting and receiving equipment. We accept that statistically, out of a large number of transmitted bits, some will be incorrect.

If data corruption causes visual damage to the machine or work-piece then we may immediately suspect the cause. However, since CNC machining is concerned with achieving complex shapes and accuracy in the order of microns, problems due to corrupted program files may not always be conspicuous or easy to trace until it is too late. For these reasons, we need to:

- (i) Minimise errors occurring during transmission
- (ii) Provide a mechanism for detecting errors
- (iii) Check all incoming data
- (iv) Detect and/or correct any corrupt information through re-transmission procedures.

The first objective can most readily be achieved by selecting an appropriate communications "medium" in which noise and EMI are minimised. Although shielded, "twisted-pair" cables and co-axial cables are commonly used as communications media, they are both susceptible to EMI. Optic fibre systems, which use light pulses rather than electrical signals for transmission, are now a preferred alternative, since they are immune from EMI.

Objectives (ii) to (iv) can all be realised by establishing a set of rules by which both the receiving and transmitting devices can check and correct for errors in data transmission. This set of rules is referred to as a "communications protocol". Most modern computer systems can be programmed to respond to any protocol. However, we again note that many CNC machine controllers do not have this programming flexibility. For this reason, CNC manufacturers often equip their controllers with built-in communications protocol software (DNC) to perform this task. In these situations, the software on the host computer is tailored to match the protocol of the CNC. Since each CNC manufacturer generally has a unique communications protocol, the establishment of secure, error-free links becomes a time consuming and expensive proposition.

Some CNCs have neither a built-in protocol nor the architecture upon which this can be embedded. Such controllers are often equipped with only a "file dump" facility that operates through the serial communications port. The "file dump" allows CNC controllers to either read or write an entire file, from/to a host computer. In order to interact with the CNC file dump facility, a host computer runs a piece of software referred to as a "terminal emulator", which performs the complementary (read or write) procedure to the CNC. However, the file dump technique makes no provision for error detection or correction. It is therefore not possible to guarantee the accuracy of files that have been dumped from a host system to a CNC (and vice-versa). "Dumped" files must therefore always be checked manually.

2.5 Manufacturing Systems

A number of different, metal-cutting, manufacturing systems are used in order to satisfy the performance criteria demanded by a wide range of industries, including workshops, automotive and aerospace manufacturers. The common system configurations are shown in Figure 2.8, which gives an indication of how each system fits into annual volume / variety regions in the production environment.

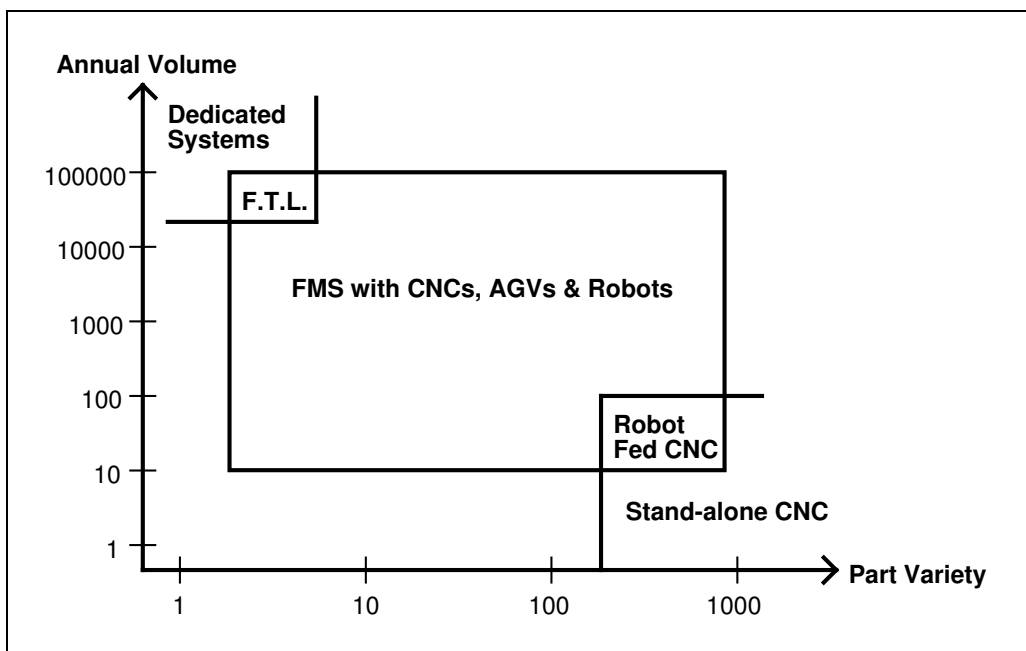


Figure 2.8 - Realms of Metal-Cutting Manufacturing Systems

The systems mapped onto the graph in Figure 2.8 have differing performance characteristics and also place different demands upon the data communications used by their control systems. We shall now examine the structure of each of the integrated systems and the way in which communication between devices occurs.

The dedicated, in-line transfer machine is shown in Figure 2.9 and is a high-volume, low part variety system. It is composed of a number of machining stations and a transfer conveyor. Each of the machining stations is designed and tooled for a specific application. For each station, tools are loaded into an induction motor driven, multi-spindle cutting head, which has an advance and retract motion. When a work-piece comes into position within a machine, the head advances for a fixed period, then retracts, to allow the work-piece to move down-line to the next destination. Each machining module is generally controlled by a PLC, which is hard-wired to its sensors, coolant pumps, etc. These machining modules are generally not user-programmable devices. They are pre-programmed to perform only a fixed task.

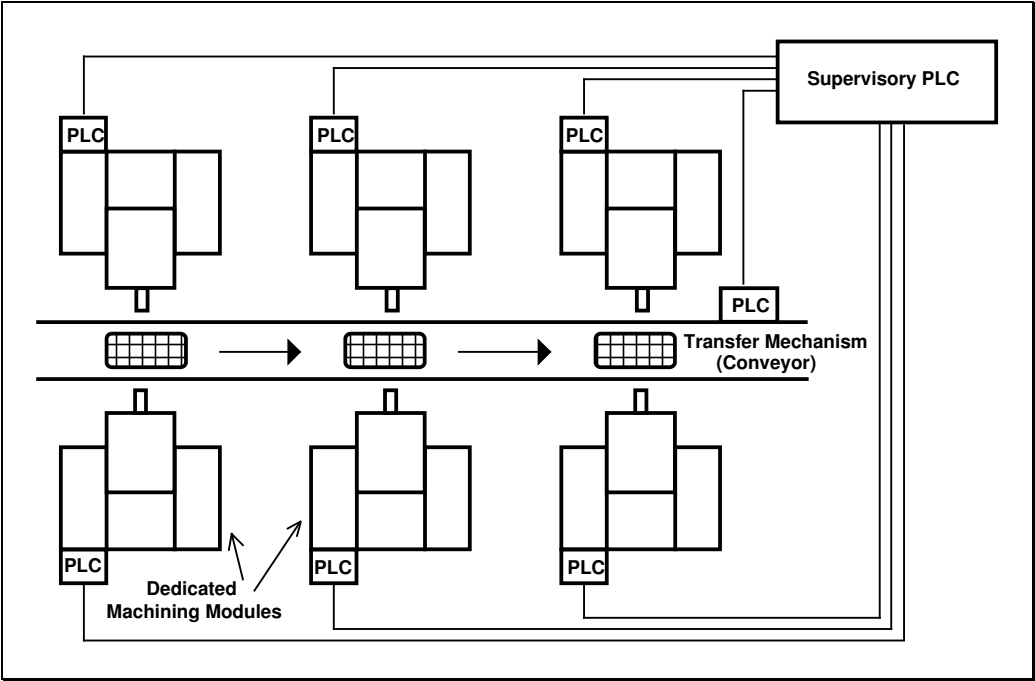


Figure 2.9 - Schematic of Dedicated, In-Line Transfer Machine

The transport mechanism for in-line transfer machines is also a PLC controlled device. In simple systems, the transport mechanism controller is also the system controller, and is hard-wire inter-locked to the dedicated machining modules. In more complex systems, a separate, high powered PLC is used to coordinate the running of the system and drive mimic-panels and graphic information displays.

In transfer machines, where individual modules are controlled by a range of PLCs, produced by different vendors, it is common practice to simply hard-wire from the supervisory PLC to other PLCs in the system. However, in a single PLC vendor environment a number of proprietary solutions are generally feasible. Some of these solutions allow PLC data buses to be inter-connected through a back-plane system for information exchange. Other solutions allow for interconnection of PLCs through high speed Local Area Networks. Regardless of which system is chosen, the objective is for the supervisory PLC to implement sequential control over the system through input/output inter-locking with individual module controllers.

Rotary transfer machines are analogous to in-line transfer machines except that parts transfer, from machine to machine, occurs through via an indexing mechanism in a circular path. The control principles however are almost identical.

The hard-wire, inter-locking, communications techniques, shown in Figure 2.9, for dedicated systems are generally adequate because:

- individual machining modules are relatively simple devices, executing simple, fixed, programs
- the amount of information which any, one machine can feed back to a supervisor is comprised of little more than off/on limit-switch status
- the supervisory controller does not need to change programs on individual modules in the system.

Dedicated manufacturing systems, of the type shown in Figure 2.9, fulfil a vital role in the high-volume production of a small variety of parts. However, in order to vary the type of part that passes through such a system, it is necessary to manually re-tool each of the machining stations. If the type of part to be produced is radically altered, then such systems require major re-engineering, or as is often the case, complete replacement. Since these systems are designed for the production of a specific item, their cost and production life are calculated on the basis of anticipated product life.

Increased competition in manufacturing, coupled with increasing consumer demands for new products, mean that product life-spans are decreasing. The cost effectiveness of dedicated production systems is therefore diminished accordingly. In addition, companies driving towards export competitiveness with products now find that they need to produce a "family" of products, tailored to specific global markets. These requirements engender a need for flexibility in production systems.

Flexibility in production systems is achieved through the ability of individual modules in those systems to respond to changes in part variety. This is in turn, achieved through the use of fully programmable machining modules and flexible parts transport techniques.

It would be sensible to suggest that flexibility in manufacturing could be achieved by taking a dedicated system, such as that shown in Figure 2.9, and replacing some or all the fixed machining modules with CNC machines. This is common practice, and the result is referred to as a "Flexible Transfer Line" or FTL. However, it should be noted that the cost ratio of a CNC machining module to a dedicated module is in the order of 10 to 1. It is therefore not economically feasible to replace, say 50 dedicated machining modules in a fixed system with 50 CNC machines. Generally, each CNC machine uses automatic, programmable tool changing in order to perform the functions of a number of dedicated modules. Thus, production flexibility is increased but throughput is decreased in what becomes a normal trade-off situation.

While it may be common in a dedicated production line to have 50 to 100 machining stations, a flexible production system may have only 5 to 10 CNC machining stations, performing the same net function at a lower production rate. However, the benefits in flexible production become self-evident when production needs change, because flexible systems can respond very quickly to new demands, with minimal human intervention.

The transfer line arrangement of Figure 2.9, whilst very fast, does not provide the optimal transport mechanism for maximum production flexibility. Robots, Gantry Robots and Automated Guided Vehicles (AGVs), on the other hand, provide a high degree of transportation flexibility at the cost of production throughput. All three devices use relatively sophisticated control systems. AGVs in particular, commonly use a powerful PLC as a Constant System Monitor (CSM), which governs the positions to which vehicles move.

The Flexible Manufacturing System (FMS), designed for a very wide variety of parts, is more likely to resemble the schematic shown in Figure 2.10, rather than that of 2.9. The intelligence level of each module (machine) within the system is much greater than that within the dedicated production line. CNC machines in sophisticated FMS environments may even be augmented with specialised robots to transfer tools from AGVs to machine tool carousels and vice-versa.

In a complex FMS environment, where a number of different part-types may be within the system simultaneously, the controller is required to:

- co-ordinate the flow of work-pieces of differing types, from one machine to another, based upon a rolling schedule
- activate different part programs on CNC machines, as required by the part-types present in the system
- down-load part programs to CNC machines as required by the machines
- co-ordinate (inter-lock) the role of the work-piece transport system with the operation of CNC machines.

Some of these functions can be (and sometimes are) implemented through the hard-wire inter-locking of devices to the FMS controller, which can be a powerful PC, PLC, workstation or mini-computer. However, FMS control is more appropriately achieved through data communications between the controller and the other computerised (intelligent) modules within the system.

In section 2.3 it was noted that there are severe limitations with hard-wire inter-locking of computerised machinery to a system (cell) controller. The system controller is unaware of the status of computerised slave devices, while they are executing their programs. In a complex, FMS environment, the system controller must have the capacity to interrogate other equipment whilst programs are running. This gives the controller access to a wide variety of information regarding the status and error-conditions of machines, thereby allowing for intelligent decision making in the control algorithm.

Simple, hard-wiring techniques only allow devices to exchange one piece of data per wire (say an on/off state or transducer voltage). They do not allow one computer to transfer data files to another computer. This of course means that down-loading of CNC machine programs from a supervisory computer cannot be achieved with the hard-wired system alone. In simple hard-wired, FMS systems, machine programs are normally resident in the local memories of each machine during a production run. Programs are generally down-loaded (or file-dumped) to machines, via data communications links, prior to the start of automatic FMS control.

One of the major benchmarks of FMS is the ability to tolerate and reconcile fault conditions. Each module in the system performs a complex task and is therefore subject to a large number of possible faults or errors. It is costly for an FMS controller to shut down an entire system, simply because one machine has developed a fault. The objective is for the controller to attempt to maintain orderly and safe system operation even under certain fault conditions.

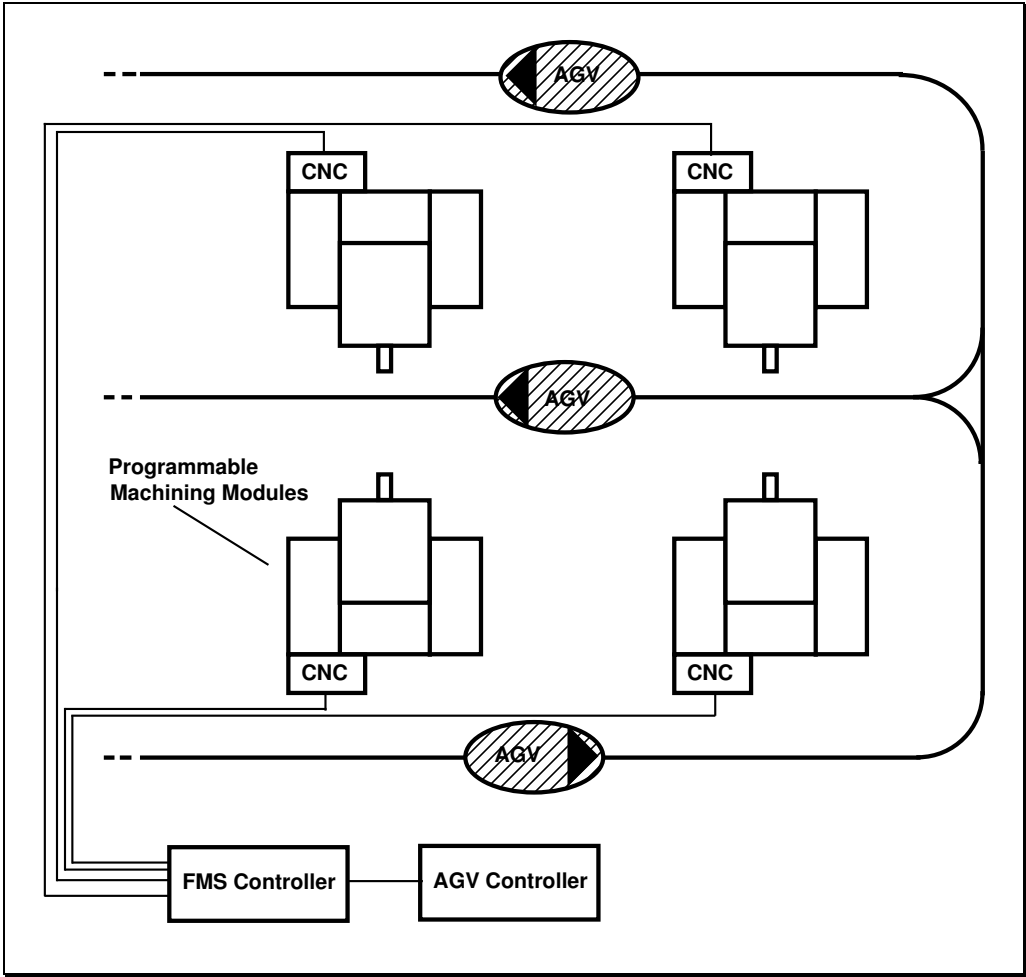


Figure 2.10 - Schematic of a Flexible Manufacturing System

If we consider a scenario where a CNC machine in an FMS system stops before its pre-defined cycle is completed, due to an internal error, then there may be a large number of causes. The action that the FMS controller can take to remedy the situation must be programmed on the basis of information available from the CNC. For example, the CNC may not have executed its program at all, because the program file may not exist in the machine's memory. The solution is for the FMS controller to down-load the file and again attempt to remotely start program execution. Alternatively, the CNC machine may have experienced a spindle-motor overload, in which case the FMS controller can only set up an external alarm and wait for human intervention.

Hard-wire communications do not provide an adequate means of conveying complex status information from a machine controller to a system control computer. This can only be achieved through more sophisticated data communications techniques.

A Local Area Network, similar to the idealised model of Figure 2.1, is the mechanism required in order for intelligent machines to communicate with supervisory systems, such as the FMS controller. In such a system, a supervisory computer can theoretically interrogate any other intelligent device at any time for status information. If an error exists, then the supervisor can determine the cause in the same way that a human would, and thereby attempt fault correction. Local Area Networks are now commonly used to link supervisory computers to slave devices in proprietary FMS systems (where equipment is essentially single-vendor).

In the past, the lack of network standardisation has made it difficult to realise a LAN in a multi-vendor FMS environment. However the push towards industrial networking standards by automotive manufacturers has made even this problem surmountable.