
Chapter 9

Real-Time Networks for Distributed Control

A Summary...

Basics of networking in relation to distributed control. Issues involved in using networks for real-time control purposes. Proprietary control networks - the Controller Area Network (CAN) and the Local Operating Network (LON).

Read This Chapter If...

- ◆ You want to learn how computer networks relate to real-time control and what their limitations are.
- ◆ You need to know why LANs could not be used for low level devices until VLSI chip solutions were developed.
- ◆ You want to know about commercially available LANs for control purposes.

9.1 Introduction

Most discussions about Local Area Networks and networking theory do not relate directly to engineering control systems. It would be quite reasonable to suggest that the majority of networks, currently in existence, have been designed primarily to transfer files from one node to another. Even in situations where we refer to networks as being "deterministic" and predictable, we still don't generally expect that they will transfer data in the sort of time-frames that we would normally associate with control functions. In this chapter, we will examine the special attributes required in a network in order for it to be deemed suitable for real-time control purposes. We will also examine a few proprietary systems that are available at the time this book has gone to press.

In order to understand the limitations of the traditional network, one really needs to understand the way in which control systems typically function. There are essentially three forms of control that we will examine. These are:

- (i) Hierarchical
- (ii) Distributed
- (iii) Heterarchical.

Since there is always some heated debate as to the meaning of these terms, we shall define them herein in our own way so that you can then equate them to whatever equivalent form you have been brought up to recognise. Our definition of these control systems is shown in Figures 9.1 (a), (b) and (c).

The hierarchical control structure is perhaps the oldest of the three forms of control. Prior to the advent of microprocessor technology, hierarchical structures were the only viable means of controlling systems such as machines, power generation systems, etc. The host system was typically a very expensive device, with relatively little power by modern terms. The host exercised closed loop control over systems via intermediate actuators, sensors and transducers.

The distributed control structure is a hybrid between the hierarchical system and the heterarchical system. It is called distributed because the total processing work associated with controlling a system is distributed over a number of intelligent devices which, in turn, interact with local sensors transducers and actuators.

The heterarchical control structure is a completely distributed system, where a number of intelligent devices all interact with one another in order to control a system. In a well designed system, the failure of one device should not cause an uncontrollable situation to arise.

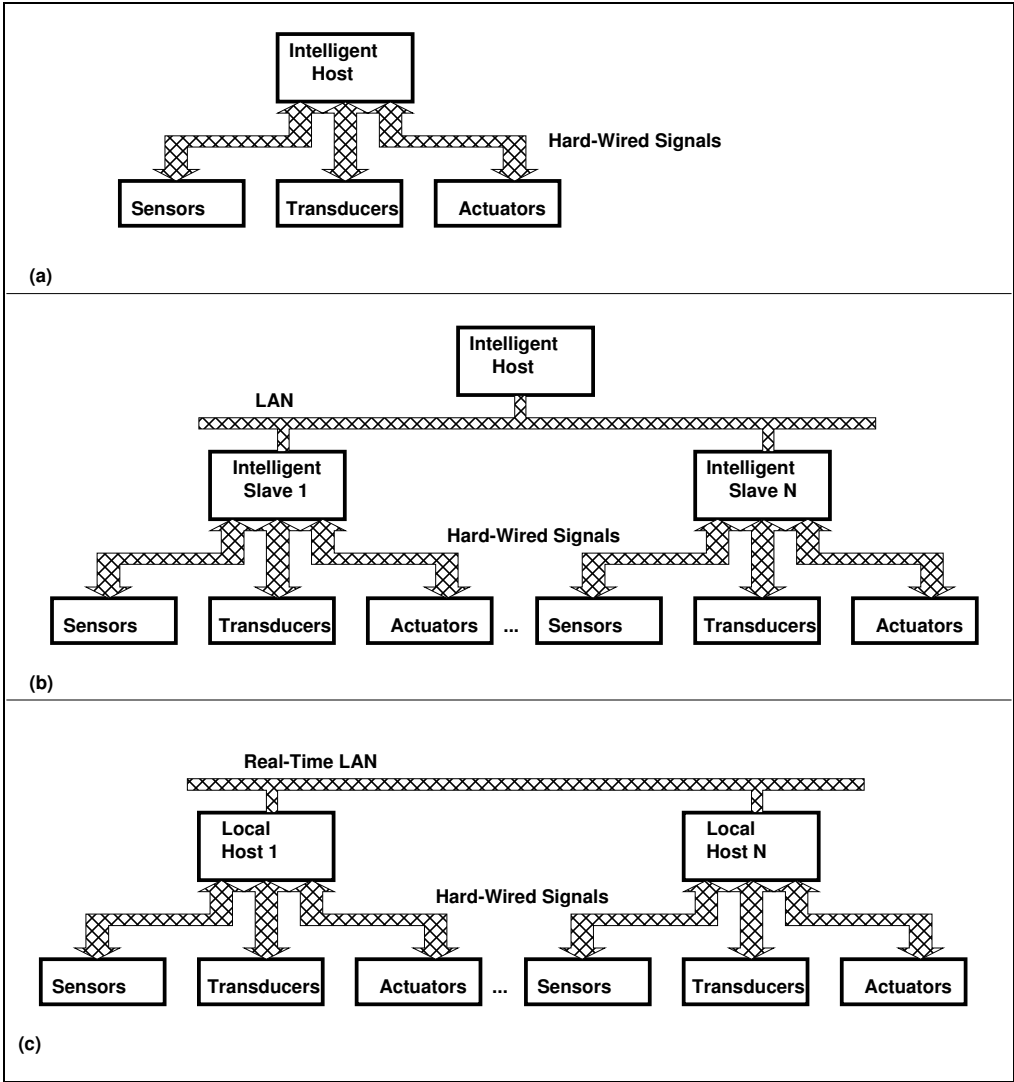


Figure 9.1 - Possible Control Architectures
(a) Hierarchical; (b) Distributed; (c) Heterarchical

Hierarchical control systems have always been far easier to comprehend than heterarchical systems. In particular, hierarchical control systems are closely aligned to classical analogue control theory. Distributed control systems are also relatively easy to come to terms with, in the sense that one can see great advantages in devolving control functions into local blocks and relieving the load upon the host system. The heterarchical control system, on the other hand, is one which intuitively appears to have a great deal of potential, but normally suffers from practical implementation problems.

The distributed and heterarchical control systems both emerged with the development of powerful microprocessors and Digital Signal Processors (DSPs) and both depend upon a reliable LAN to transfer data. In both these systems, we are concerned with the throughput of the network in terms of the number of messages that can be transmitted over a given time, rather than the number of bytes. This is because control information in distributed and heterarchical systems tends to be short in length, but needs to be transmitted very quickly.

In the distributed system, the throughput of the network is actually dependent upon the distribution of control functions. In other words, if the slave processors in the system are very intelligent, then the information flow may be relatively sparse under normal circumstances. A good example is a Flexible Manufacturing System, where a host computer controls robots, CNC machines and PLCs. The slaves in this case only require a preliminary down-loading of program files and then occasional start messages issued by the host. Unless a serious error occurs in the slave devices, there is little need for them to transmit information (other than simple acknowledgement messages) back to the host. On the other hand, the slave processors may be little more than microprocessor or DSP controlled transducers which require a constant flow of information from the host. In these instances, the throughput of the network needs to be very high in terms of the number of messages transmitted per second.

A heterarchical control system is a fully decomposed architecture that is normally critically dependent upon its network for success. Since the network messages are an integral part of the control structure, the throughput of the network must be high enough to enable the total control algorithm to execute. However, if we have a situation where each of the local host processors are very intelligent in their own right, then we can still minimise the demands upon the network as we do in the distributed system.

The hierarchical control system, as we have shown it in Figure 9.1 (a), has no network associated with it. However, this is a historical restriction that we have always assumed to be insurmountable as a result of technical limitations in networking. In fact, thus far, we have always assumed that networking was only for "intelligent devices", such as computers and other processor controlled devices. We have never considered the possibility that simple transducers, switches, actuators, etc. should be connected to a network.

In the light of our traditional understanding of networks, it would appear to be rather ludicrous to suggest that a switch or a transducer should be connected to a full seven layer OSI network. The cost of the interface would be prohibitive. In fact, the interface would need to be a complete computer system composed of a multi-tasking processor, RAM and EEPROM. The processor would need to power a transceiver device that provided the line-drivers for the communications medium on the network. So, in terms of our traditional understanding we would have never contemplated the idea of reorganising and redistributing our concepts of control through networks.

What we have failed to consider in terms of our traditional approaches to control is that the cost of the network interfaces will diminish substantially with:

- Increased processor power
- Increased production volumes
- Standardisation of networks.

We already know that standardisation in networking is a difficult concept and we also know that it generally comes through defacto usage of a given system more than it does by Government specification and regulation - the Microsoft DOS system is the classic example here. The logic we therefore need to consider is that if we can generate low cost network interfaces for simple devices (such as switches, etc.), then production volumes will dramatically increase, costs will reduce even further and the normal technology spiral will ensue.

The interesting point to note is that since the early 1990s, a number of companies have been pursuing just such a course of action. This is a radically different approach to the end-user-specification method tried by General Motors in the 1980s, with only limited success. In this chapter we shall briefly examine some of these proprietary networks and their implications for control. We shall do this with the assumption that eventually, one of these systems will ultimately achieve a defacto standard status and perhaps, change the way in which we structure our control systems.

9.2 Typical Control Applications for Real-Time Networks

At the time that this book had gone to press, there were no networks which could justifiably be called "real-time" in the classical analogue control sense of the word. When we talk of analogue control, we are generally dealing with what could be classified as instantaneous response - or more appropriately, devices which can provide output signals within nanoseconds or microseconds of having received some input energy. The very nature of the networking process, encompassing media access (contention resolution), modulation, demodulation, packetisation (framing) and depacketisation (de-framing) of data, error detection and correction, does not yet lend itself to nanosecond or even microsecond response times. However, since most modern control systems are computer or processor based, we have come to accept that some small delays can be tolerated by an appropriate structuring of the control architecture.

If a network is sufficiently well designed, then there are many instances where it would be possible to use such a network for system control. In this section, we will only examine a few basic areas to demonstrate the current scope for real-time networks.

(a) *Vehicle Control Systems*

Most modern cars feature one or more microprocessors and/or DSP devices that are used to coordinate a range of functions, including:

- Engine management and fuel injection
- Electric, programmable seats
- Climate control
- Safety systems (air-bags, seat-belt utilisation, etc.)
- Body management (suspension stiffness, body distortion, etc.) systems.

Modern vehicle management systems are in fact based upon distributed control systems, receiving feedback from hundreds of sensors, transducers and switches and actuating solenoids, relays, stepper and servo motors.

It would be reasonable to suggest that if the control functions are carefully distributed then small delays, caused by networking, could be tolerated in order to provide drivers with a sophisticated control system, capable of monitoring all important vehicle activities.

The alternative to using a network in a vehicle is to simply hard-wire all sensors back to their relevant controllers and eventually feed this information back to the driver via some intermediary processor. The problem with this approach however, is that it tends to generate enormous amounts of wiring, leading to reliability problems with harness connectors, spurious signals due to electromagnetic interference and space and interference problems due to moving harnesses. Another point of concern is the enormous cost of building the wiring looms required for vehicles whose complexity is constantly increasing.

These problems are obviously not restricted to cars and are in fact magnified when one considers the complexity of modern buses, tram cars and aircraft, where dozens or hundreds of seats need to be supplied with a range of different functions including seat control, air-conditioning control, etc. Clearly networking needs to be considered for all these control applications.

(b) *Machine Control Systems*

Modern production machines and robots are composed from a collection of servo motors, each feeding back information to a central computer, which coordinates axis movement. However, recent developments in servo technology have changed the nature of control systems from the traditional hierarchical approach shown in Figure 2.3, to one where the servo controllers are provided with intelligence and can be scheduled by the host.

A number of servo motor control systems are now available with limited communications facilities such as RS-232 and RS-422 based point-to-point links or RS-485 based multi-drop networks. Even modern stepper motors (indexers) are becoming available with communications links. The problem with all these devices is that there are no standard or even common communications protocols to which they adhere. Moreover, the communications protocols often chosen for these networks are generally not optimised (that is, fast enough) to enable these devices to form a networked structure with the CNC or robot control.

A low level "real-time" network, capable of transmitting small packets of data at very high speed is required in order to facilitate a re-think on the basic design of CNC and robotic control systems.

(c) ***Intelligent Building Control***

Commercial and domestic buildings, like cars, are becoming more and more complex. Zone-controlled air-conditioning systems, lighting control systems, fire protection systems, security systems, access control systems and the like are all features of modern office buildings. Around the home, most appliances are developed with some form of processor control and timing circuitry and office security and protection systems are gradually making their way into domestic situations.

Until recently, one of the only alternatives in combining these systems into cohesive controls was to use hard-wiring. Needless to say, this is quite expensive, particularly when one considers that each (fire protection, access control, etc.) system has traditionally been treated as an "island" of control and that little interaction between systems normally occurs. There is clearly a need for networking in this environment, but we are again confronted with the problem of interfacing low level devices (switches, air-conditioning baffle controls, etc.) to sophisticated networks. In other words, the cost of interfacing these devices may be orders of magnitude higher than the cost of the devices themselves. The provision of a network interface worth several thousand dollars for each light switch worth several dollars is clearly not a feasible solution to the problem.

(d) ***Manufacturing Management Systems***

Production and inventory control systems and materials and requirements planning systems are notorious for the problems caused by "old" factory floor data. It isn't simply a question of extracting information from intelligent devices such as CNC machines, robots and PLCs, but also a question of acquiring data from simple sensors, bar-code readers, part identification systems and so on. Many factories still acquire this data manually because it has been too difficult to provide networked feedback systems that can satisfactorily provide an automated solution.

In this environment, a low-level network that could provide feedback from simple devices, at low cost, could improve the problem of planning with "old" data. The time response in such a network is not the crucial factor, since a delay of milliseconds is clearly better than a manual delay of one hour, day or week. The real issue here is low-cost interfacing of low-level devices.

In each of the above situations where networks can be applied, the same issues arise. First and foremost is cost. Second is the ability to integrate low level devices into the network. A network which only includes some devices and sensors is not really a solution, but an exacerbation of the existing problem.

9.3 Proprietary Real-Time LANs - CAN and LON

The need for so-called real-time networks that can provide low cost interfaces to low level devices has not gone unnoticed. The problem, as usual, is non-standardisation of networks. Just a few of the proposed solutions (standards) to the problem include:

- Enhanced Performance MAP (factory automation)
- CAN (factory automation and automotive)
- Fieldbus (SP50) (factory automation)
- SAE (J1850) (automotive)
- BacNet (ASHRAE) (building control)
- IBI (building control)
- TRON (building control)
- Home Bus (building control)
- CEBus (EIA) (building control)
- LONWorks (building control, automotive and automation).

The problem with most of these "solutions" comes in terms of implementation. There is nothing new about this problem and it is common to nearly all networks. The difficulty that system designers and developers have is that manufacturers tend to equate the provision of semiconductor devices, which implement the physical and data link layers of the OSI model (to some specification), to the provision of a network. This simply isn't the case, because the developer is left to implement the upper layers of the OSI model.

From a system developer's point of view, The real issue in networking can be summarised in the following terms:

"Given a transducer, sensor, switch, microprocessor or DSP controller, how much hardware and software development work needs to be carried out in order to interface the device to a network?"

Very few commercially available networks are able to provide a satisfactory answer to this question and, from the above list, we will briefly examine two relatively promising proprietary systems. The two systems we shall briefly examine are the:

- (i) Controller Area Network (CAN) Bus
- (ii) Local Operating Network (LON) Works.

(i) *CAN Bus*

The Controller Area Network was developed by the Bosch corporation, specifically for the automotive industry and for factory automation purposes. The objective of the CAN system is to provide a network which can transfer small packets of data at relatively high speed over a range of media. The system can be classified with the term "real-time" as far as that terminology can be stretched for control purposes.

The Philips corporation has created the semiconductor devices that implement the lower layers of the network protocol in hardware. At the time of writing this text, the upper layers of the CAN system had to be implemented in software. The major application of CAN is in automotive control systems and the network has reportedly already been used by European car manufacturers, such as Daimler-Benz, in complex car control systems.

There is nothing remarkable about the CAN protocol itself as it is composed of network layers similar to those found in other networks. The lower network layers are designed to provide reliable high-speed data transfer for small packets and to handle network contentions via a special CSMA/CD scheme that provides prioritised packet handling for urgent messages. The special CSMA/CD scheme overcomes most of the traditional problems associated with the non-deterministic nature of this media-access scheme. Its development clearly follows on from the acceptance of the Ethernet system as a defacto standard in office-networks. Error checking in CAN is implemented through Cyclic Redundancy Check algorithms.

Given that the CAN system has already received some acceptance in the European community, as a result of its implementation in the car industry, it is anticipated that a number of third party products will emerge in order to simplify the networking process. At this point in time, the upper layers are a sticking point in implementation. Although software is available for a range of different microprocessors and DSPs, the real issue that needs to be resolved is how the network can be interfaced to simple, low-level devices.

(ii) *The LON Works Architecture*

This is perhaps one of the most interesting and promising networking developments to have emerged in recent times. The LON system was developed by the Echelon corporation in the United States, with semiconductor support devices implemented by the Motorola corporation. It is amongst the very first, complete OSI networking systems to be implemented almost totally in hardware.

The basis of the development is the so-called Neuron chip, produced by Motorola. The device is shown in Figure 9.2, which has been reproduced (with permission) from Motorola literature.

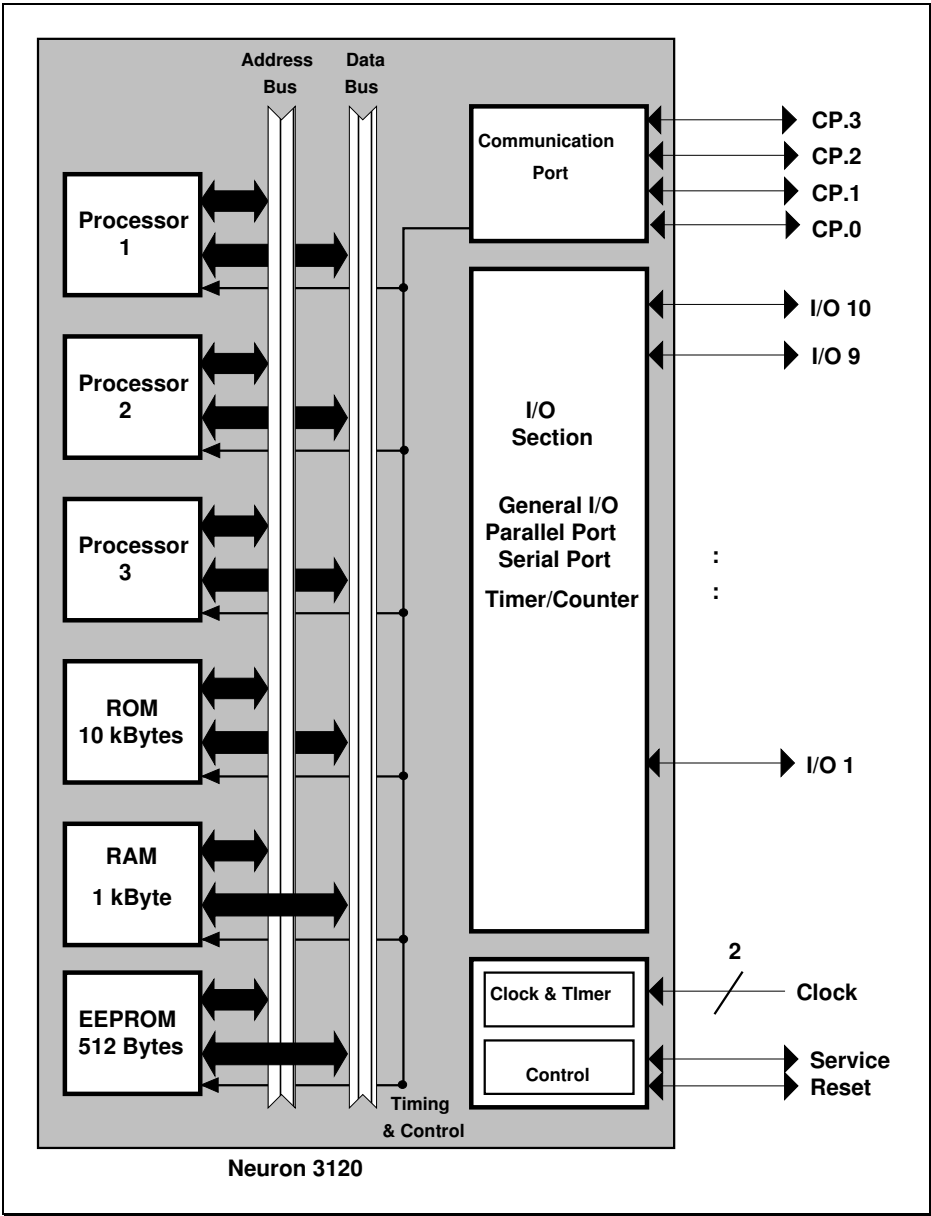


Figure 9.2 - Motorola 3120 Neuron Chip
(Reproduced from Motorola Original Diagram)

The Neuron chip contains all seven layers of an OSI communications protocol within a single chip, which is produced by Motorola at a very low cost. The 3120 chip, shown in Figure 9.2, contains three processors, which together implement the LON-Talk protocol and interact with user hardware via the input/output lines in the I/O section of the chip. The LON Works architecture is based upon a prioritised CSMA/CD media access system, which facilitates real-time networking. The system developer links their device to the Neuron chip via a short program, written in a special implementation of the C language, and stored in EEPROM on the Neuron chip.

Each Neuron device has its own unique 47-bit address, which is preconfigured during production. The system developer does not really need to become involved in the addressing, contention resolution or other technical networking issues, which are all handled by the Neuron device. The main tasks of the system designer are then to:

- Connect the local device (switch, transducer, DSP, microprocessor, etc.) to the neuron chip via the I/O lines
- Connect the Neuron chip to the network medium via a matching transceiver chip (which can drive twisted-pair, powerline or radio-frequency media)
- Program the Neuron chip to interact with the network via the special Neuron programming language.

The overall network structure then becomes as shown in Figure 9.3.

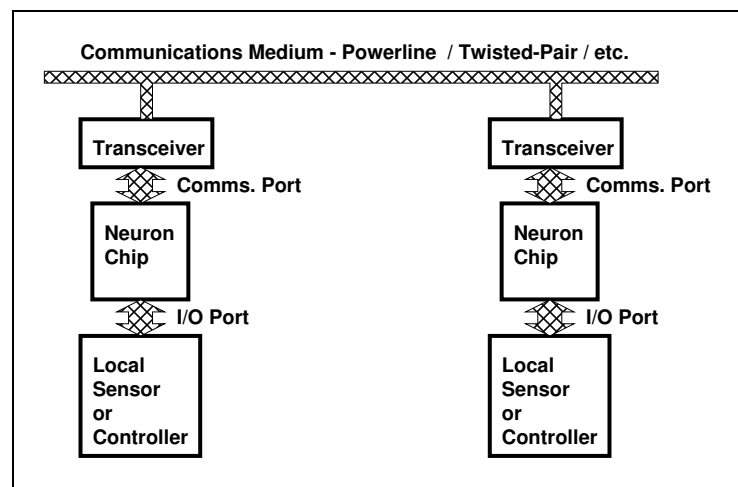


Figure 9.3 - System Design Using LON-Architecture

The benefits of the LON system are predominantly in the development stage, where the Neuron programming system isolates the developer from the majority of networking functions. The sophisticated network programmers interface even alleviates the need for system developers to concentrate on addressing problems, since these can readily be accounted for in the high level Neuron "C" language syntax.

Of particular interest with the LON Works architecture are the range of transceivers that have been made available. These include powerline transmission and radio frequency transmission. The powerline system is of particular importance to the building industry where cabling costs can be substantially reduced by having intelligent Neuron-based nodes (light fittings, light switches, alarms, etc.) networked to one another through the normal building power supply bus. The radio frequency transmission system is also of interest to manufacturing installations where it may be necessary to network devices which are located on rotating or moving machinery and are otherwise difficult to connect with cables, etc. Unfortunately, both of these alternatives offer relatively low data transmission rates and are not suitable for control environments that are network intensive.

9.4 Conclusions

The development of the proprietary systems such as the CAN system and the LON system have been included herein because they are amongst the first systems to turn control networking into a commodity that can readily be exploited by system developers.

These are the first signs that networking for control applications can be achieved for a low cost, in terms of individual interface hardware and development time. They are certain to form new trends in distributed control structures for both manufacturing systems and building control systems. What remains now is for us to see which of these emerging systems will ultimately gain the status of defacto standards and how these systems will tie into the more traditional office-type networks to which we have already become accustomed.