

Industrial Automation Tutorial

Introduction

Industrial communications refers to the wide range of hardware and software products and protocols used to communicate between standard computer platforms (PC, Macintosh, or workstation) and devices used in industrial automation applications. This tutorial describes the fundamentals of serial interfaces and the simpler software protocols commonly used with industrial devices. There are communications many protocols that require sophisticated hardware and software to ensure robust, reliable, and sometimes real-time operation. These industrial networks are sometimes generically referred to as fieldbus. This tutorial will introduce you to some of the many industrial networking protocols in use and under development.

Serial Interface Standards

Many devices used in industrial applications use EIA standards RS-232, RS-422, or RS-485 to connect to computers and to one another. (In this tutorial, we use the term RS-xxx to refer generically to RS-232, RS-422, and RS-485). A common misconception about these specifications is that they define specific software protocols. The ANSI/EIA RS-xxx standard specifies only the electrical characteristics – not the

	RS-232	RS-422	RS-485
Type of transmission lines	Unbalanced	Differential	Differential
Maximum number of drivers	1	1	32
Maximum number of receivers	1	10	32
Maximum cable length meters (feet)	15.2 (50)	1.2 (4000)	1.2 (4000)
Maximum data rate	20 kb/s	10 Mb/s	10 Mb/s

Table 1. Features of RS-232, RS-422, and RS-485

soft-ware protocol. Table 1 summarizes the main features of these three serial interfaces as defined in their respective standards documents.

RS-232 (ANSI/EIA-232 Standard) is the serial connection found on IBM-compatible PCs. It is used for many purposes, such as connecting a mouse, printer, or modem, as well as industrial instrumentation. Due to improvements in line drivers and cables, applications often increase the performance of RS-232 beyond the distance and speed listed in the standard. RS-232 is limited to point-to-point connections between PC serial ports and devices.

RS-422 (EIA RS-422-A Standard) is the serial connection used on Apple Macintosh computers. RS-422 uses a differential electrical signal, as opposed to the unbalanced signals referenced to ground with RS-232. Differential transmission, which uses two lines each for transmit and receive signals, results in greater noise immunity and

longer distances as compared to RS-232. The greater noise immunity and distance are big advantages in industrial environments.

RS-485 (EIA-485 Standard) is an improvement over RS-422 because it increases the number of devices from 10 to 32, and defines the electrical characteristics necessary to ensure adequate signal voltages under maximum load. With this enhanced multidrop capability, you can create networks of devices connected to a single RS-485 serial port. The noise immunity and multidrop capability make RS-485 the serial connection of choice in industrial applications requiring many distributed devices networked to a PC or other controller for data collection, MMI, and other operations.

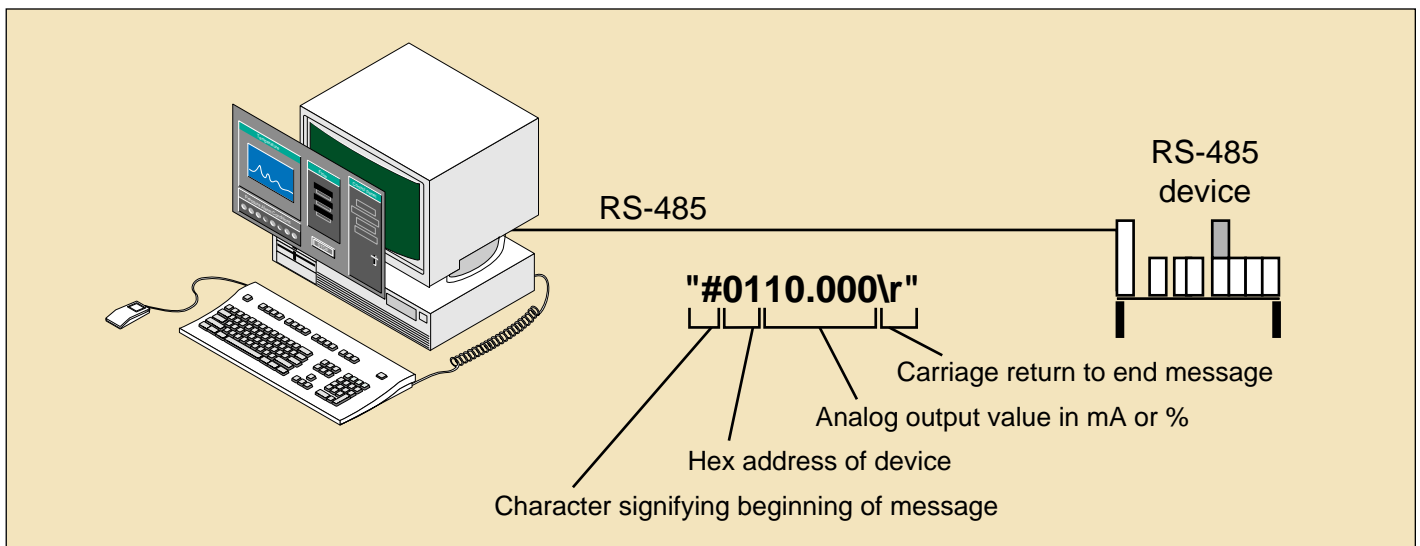


Figure 1. An example of a command string sent to a 6B analog output module.

Software Protocols

Note that RS-xxx protocols do not specify anything about the software used to communicate with devices. A common question asked about RS-485 devices in particular is, “How do I address my RS-485 device?” The answer is, “It depends on the device.” RS-485 specifies only that it is capable of connecting multiple electrical loads (devices) on a single set of wires. It does not say anything about how to address (or communicate at all with) these loads. It is the job of a particular device’s software protocol to describe how to address a specific device.

There are very few protocols that have widespread use by serial instrument manufacturers. The most common solution is a set of ASCII strings that constitute commands to the device. These protocols are most commonly implemented as asynchronous protocols, so-called because the transmitter and receiver do not have any tightly-coupled synchronization mechanisms. These networks are commonly master/slave, where one device (usually the computer) is the master and the devices are slaves. Normally, all devices power up in receive mode, waiting to receive messages. When the master transmits a message, all devices receive the message (on a multidrop network) and determine if the message is addressed to them. If it is, they act upon the message, writing data on the serial interface if required to do so. Figure 1 shows an example of a command string used to set the output level of an Analog Devices 6B analog output module. Similar commands are used to read data from a module.

Most manufacturers of serial devices invent their own ASCII protocol for their particular device. It is a relatively straightforward task to read and write these ASCII strings using the standard serial functions built into a language such as Visual Basic. You must understand how to use the string formatting commands and serial functions in order to build and parse command strings. Instrument

libraries in LabVIEW, BridgeVIEW, or LabWindows/CVI encapsulate these types of commands into high-level VIs, servers, or functions for configuring and querying serial devices. OptoMux and ModBus ASCII are examples of serial protocols that use standard serial ports and have achieved broad acceptance across many devices and suppliers. (See Table 2 for a list of devices that use the modbus protocol.)

Another protocol commonly used with standard PC serial ports (using an electrical adapter for physical signal conversion) and that has found wide usage across process control instrumentation is the HART (Highway Addressable Remote Transducer) protocol. HART is a hybrid network that uses the 4-20 mA analog signal commonly found on process control instrumentation and adds to it a digital signal. The digital signal is added in such a way that it does not interfere with the standard 4-20 mA functionality. In a control system where the HART protocol is understood, the digital signals can be read from the device to read and write data.

There are many protocols that are used on standard serial interfaces but use much more sophisticated data packaging schemes. These protocols use different mechanisms



Figure 2. PLCs typically use vendor-specific communications protocols that require specialized hardware interfaces and software drivers.

to ensure reliable, robust communication. Most such protocols are proprietary to a specific supplier, requiring special software drivers and often special interface hardware. Examples of these protocols are those used for communicating with PLCs, for example DataHighway+ from Allen-Bradley or ModBus+ from Modicon. (Figure 2)

Device Type	Manufacturer
Gas chromatograph	Daniels, Applied Automation, ABB
Power measurement	Multilin, Power Measurements, Bitronics, Sprecher+Schuh
DCS systems	Foxboro, Honeywell, Rosemount, Westinghouse, Fisher&Porter, Fisher Controls
Tank level instruments	Rosemount HTG, Gauging Systems, Enraf, Sarasota, Saab, Varec, Endress+Hauser
Value systems	Keystone Controls, Limitorque, Pakscan
Single loop controllers	Eurotherm, Micon, Toshiba
Flow computers/transmitters	OMNI, Daniels, MicroMotion, Eliot
PLCs	GE, Square-D, TI, Siemens, Westinghouse, Modicon
RTUs	Arcom, Automation Electronics, Westronics, (almost every RTU manufactured)
Variable speed drives	Allen-Bradley, Telemecanique/Square-D, Siemens, Toshiba, Magnetek, Safronics

Table 2. Devices and Manufacturers that use the Modbus Protocol

Industrial Automation Tutorial

Industrial Networks

There is growing momentum toward more multivendor connectivity through standardized versions of the sophisticated industrial network protocols mentioned at the end of the previous section. In this tutorial, we will generally refer to these networks as industrial networks. There are some key reasons that users are looking to move to standardized industrial networks.

Open Systems – It is difficult and costly to integrate systems with instrumentation from several vendors because of the multitude of communication protocols. With standard protocols, devices from many suppliers can coexist on the same network and communicate with one another.

Cost Reduction in Wiring – Many systems are still using 4-20 mA analog instrumentation, requiring extensive point-to-point wiring. Multidrop wiring means lower installation costs.

Increased Information Need – In today's regulatory environment, companies are required to gather more information about their processes and the instrumentation connected to the processes. Traditional 4-20 mA instrumentation provides only one value, the process value. On a digital network, instruments can provide maintenance and diagnostics information for better tracking of instrument performance.

Sensorbus	Devicebus	Fieldbus
CAN	CAN	IEC/SP50
Serialplex	DeviceNet	Fieldbus Foundation
ASI	Profibus DP	Profibus PA
LONWorks	LONWorks	LONWorks
	FIPIO	WorldFIP
	SDS	
	Interbus-S	

Table 4. Classification of Industrial Device Networks

Intelligent Devices – Device vendors are putting more intelligence into their devices to satisfy customers' demands for more functionality at lower costs. The increased information available with a digital network is necessary for capitalizing on the extra capabilities made possible by intelligence in the devices.

Figure 6 on page 6-12 illustrates how standard industrial networks will transform today's control system.

Industrial networks are often characterized according to the type of device best suited for residence on the network. Three general classes of such networks are sensorbus, devicebus, and fieldbus. Table 3 summarizes the major characteristics of each type of bus.

There are many networks in use and under development today. Table 4 lists some of these networks and their bus classification(s). The reason for the many buses is that there is a very wide range industrial process and manufacturing

applications that can use digital communications. For example, simple proximity sensors on a conveyor belt can be networked together to a system that controls the movement of boxes on the belt. Another example is a control valve used to regulate the flow of crude oil within a petroleum refinery. These two examples show the extremes of digital communication. The proximity sensor has a simple function – transmit an on/off signal indicating if a box is near the sensor. This signal can be accommodated in a few bits of data. Diagnostic information from the sensor, if it exists at all, is probably limited to a single "health" indicator which again requires very little data to communicate. The control valve, on the other hand, could be expected to provide very sophisticated diagnostics, such as number of turns since last servicing, bearing friction, and ambient operating temperature. These parameters may be extremely critical in an environment such as a refinery where failures can result in dangerous situations and costly downtime. The network in

	Sensorbus	Devicebus	Fieldbus
Primary applications	Discrete, machine	Discrete, machine	Process
Typical control system	PLC	PLC	DCS
Data size	Less than 1 byte	Up to 32 bytes	Up to 1000 bytes
Microprocessor-based	No	Yes	Yes
Embedded intelligence	No	Varies	Yes
Diagnostics	No	Simple	Sophisticated
Response time	5 ms or less	5 ms or less	100 ms
Distance	Short	Short	Long
Example device	Discrete proximity sensor	Photoelectric sensor with diagnostics	Smart valve with PID capability and advanced diagnostics

Table 3. General Characteristics of Industrial Device Networks

Industrial Automation Tutorial

a refinery may therefore be expected to be extremely robust and able to handle large amounts of data. It stands to reason, then, that the proximity sensor and the control valve have different network requirements. Thus, the reason for such a wide variety of industrial communication networks.

As mentioned previously, industrial network protocols are much more sophisticated than the simple command sets commonly used with typical serial instruments. Dedicated hardware and software drivers are required to provide robust connections between computer platforms and each of these networks. Options for using these networks today include DDE servers and DLL function libraries for the Windows environment. Though it is not clear that any one network will satisfy all industrial networking requirements, these buses will bring more standardized interconnection between computers and the devices used in industrial automation applications.

FOUNDATION Fieldbus

FOUNDATION Fieldbus is a very sophisticated industrial network specifically targeted at the need for robust, distributed control in process control environments. The fieldbus specifications were developed by the Fieldbus Foundation, a group representing over 80% of the world's suppliers of industrial automation systems, devices, and services. FOUNDATION Fieldbus is based upon existing standards and other proven technologies, including work from the ISA (International Society for Measurement and Control), IEC (International Electrotechnical Committee), Profibus (Process Fieldbus, a German national standard), FIP (Factory Instrumentation Protocol, a French national standard), and HART (Highway Addressable Remote Transducer, a widely-used process instrumentation protocol).

The FOUNDATION Fieldbus communication protocols are based on the OSI (Open Systems Interconnect) seven layer communi-

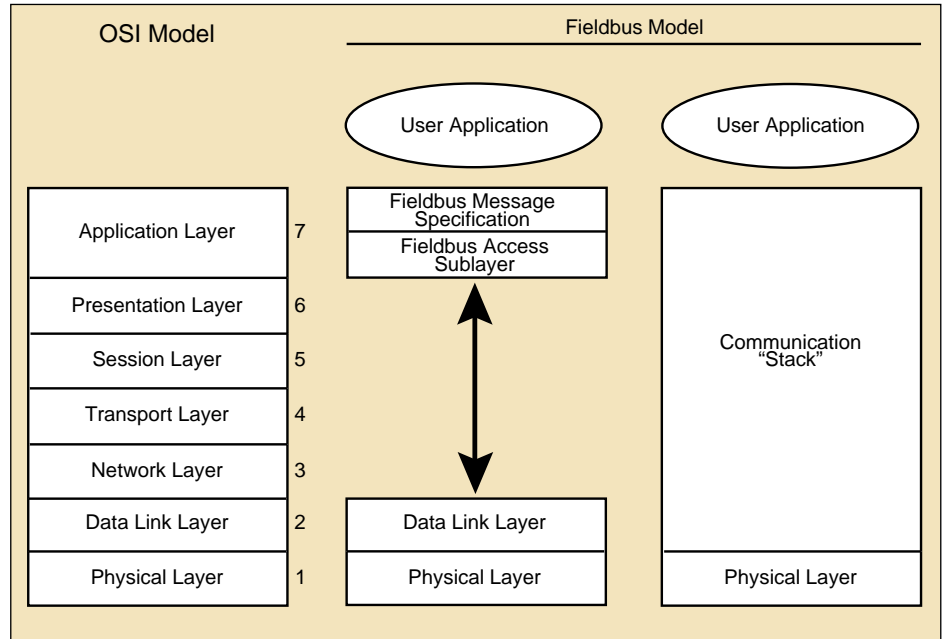


Figure 3 – The Fieldbus model compared to the OSI 7-layer communications model.

Note that OSI does not define a user application.

cations model. FOUNDATION Fieldbus has optimized the OSI architecture for process control by removing the middle layers that are generally associated with non-time critical applications such as file transfer. FOUNDATION Fieldbus technology consists of a) the Physical Layer, b) the Communication "Stack", and c) the User Application. Figure 3 shows these components modeled using the OSI layered communication model. The physical layer (electrical, wiring, and so on) is based upon standards created by ISA/IEC (ISA S50.02-1992, IEC 1158-2). These physical layer standards specify data communication rates of 31.25 kb/s, 1 Mb/s, and 2.5 Mb/s. Devices operating at 31.25 kb/s can draw their power directly from the network. This Physical Layer also specifies the capability for intrinsically safe operation.

The layers above the Physical Layer together are often referred to as the "stack" for fieldbus. Detailed discussion of the stack is beyond the scope of this tutorial. Several characteristics and functions in the Data Link Layer, however, are key to the distributed, real-time control capabilities of FOUNDATION Fieldbus:

1. The Data Link Layer is based on a token passing protocol.
2. The Link Active Scheduler (LAS) is a centralized device that acts as the arbitrator of the bus.
3. The LAS executes a schedule that makes possible deterministic communication.
4. The LAS distributes time to the network to permit all devices to share the same sense of time.

A key "8th layer" of FOUNDATION Fieldbus is the User Application, or User Layer. The User Layer defines "blocks" that represent the functions and data available in a device. Rather than interface to a device through a set of commands as commonly used with communication protocols, a FOUNDATION Fieldbus user interacts with devices through a set of blocks that define device capabilities in a standardized way. In this tutorial, we'll describe the most important block type – Function Blocks.

Function Blocks are the core components with which a user specifies the behavior of a control system. FOUNDATION Fieldbus defines standard sets of Function Blocks.

Industrial Automation Tutorial

Function Block Name	Symbol
Analog Input	AI
Analog Output	AO
Bias	B
Control Selector	CS
Discrete Input	DI
Discrete Output	DO
Manual Loader	ML
Proportional/Derivative	PD
Proportional/Integral/Derivative	PID
Ratio	RA

Table 5 – The 10 basic standard function blocks defined by FOUNDATION Fieldbus.

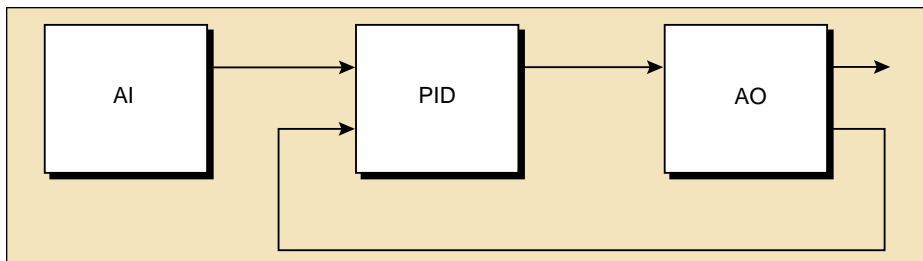


Figure 4 – A Basic Control Loop Using AI, PID, and AO Function Blocks.

There is a set of 10 basic control and I/O functions (see Table 5).

The inputs and outputs of individual functions blocks can be connected to specify communication of data on the bus. Even more importantly, the execution of a function block can be precisely scheduled. This is a very key capability of FOUNDATION Fieldbus because it allows execution of control loops directly over the network. The Function Blocks themselves reside in individual devices but the overall scheduling of execution is specified and executed across the network. Figure 4 shows a simple control loop with three Function Blocks – AI, PID, AO.

The function blocks shown in Figure 4 could be implemented on the fieldbus in several different ways. The AI, PID, and AO could each reside in separate devices, such as a transmitter, loop controller, and valve. Alternatively, the PID itself could reside in the control valve. In the second case, there is no explicit controller device. In either system, the user's view is the same – a series of connected function blocks and an execution schedule. The

second system, however, shows the true potential of FOUNDATION Fieldbus – distributed control, where the control functionality exists out in the field rather than concentrated in larger controllers.

A second important feature of the FOUNDATION Fieldbus User Layer is Device Descriptions. A key objective for FOUNDATION Fieldbus is interoperability – the ability to build systems comprised of devices from a variety of manufacturers and take full advantage of both the standard and unique capabilities of every device. Function Blocks go a long way to ensuring a consistent model of a control system. From a system point-of-view, however, a mechanism is needed to document in a standard way the types of functions available in any given device. To achieve this end, FOUNDATION Fieldbus defines the Device Description (DD), is a standardized description of the functions available in a device. Using the DD, the host in a control system, for example a Windows NT-based MMI, can obtain the information necessary to create the human interface for interacting with the device to configure parameters, perform calibration and diagnostics, and

other functions. Note that the DD is a mechanism for describing the functions in a device. This is the key to fieldbus interoperability. This can be contrasted to a more simplistic and common approach to the problem of compatibility and interchangeability, namely by specifying that only a given set of functions can be used in a device to ensure that a given system can always talk to a new device. This would severely restrict the ability of a device manufacturer to innovate by adding new device features, and there would be never-ending contention about the “right” set of functions upon which to standardize. With the DD, developers can add new features and be confident that host systems can learn about and take advantage of these features in standard way.

Controller Area Network (CAN)

The Controller Area Network is a serial bus growing in popularity as a device-level network. CAN was developed by Bosch to address the needs of in-vehicle automotive communications. Automobiles have a variety of control devices in them, for such functions as engine timing, carburetor throttle control, and antilock brake systems. With increasing demands placed upon these systems for safety, performance, and customer needs, CAN was developed to provide a digital serial bus system to connect controllers. CAN has been standardized internationally (ISO DIS 11898 and ISO DIS 11519-2) and is already available in a number of silicon implementations. The CAN protocol meets real-time requirements encountered in many automotive applications. The network protocol can detect and correct transmission errors caused by electromagnetic interference. Also, the network itself is relatively easy to configure and offers the ability to perform centralized diagnostics.

Comparison of automotive and industrial network requirements show that a number of characteristics of CAN also make it suitable for industrial applica-

tions. These characteristics include low cost, suitability for harsh electrical environments, good real-time capabilities, and ease of configuration. There are now many examples of CAN being the basis for networks used in industrial manufacturing applications. CAN is particularly well-suited to networking smart I/O devices, as well as sensors and actuators, either in a single machine or in a plant. Several industrial devicebus systems have been built upon CAN. Allen-Bradley developed DeviceNet, a CAN-based protocol now maintained by the Open DeviceNet Vendor's Association. Other such industrial networks include CANopen, developed by CAN in Automation (CiA) and the Smart Distributed System (SDS), developed by Honeywell Microswitch.

CAN is a communications protocol specification that defines parts of the OSI Physical and Data Link Layers. CAN does not specify the entire Physical Layer or the Medium upon which it resides, or the Application Layer protocol used to move data. (See Figure 5).

As with the other network protocols previously mentioned, detailed technical description of CAN and CAN-based protocols is beyond the scope of this tutorial. Listed below are some key technical aspects of CAN.

1. Typical data rates are 125 k/s to 1 M/s, dependent upon the distance over which the network is operating. The allowable distance ranges from 40 m at 1 Mb/s to 500 m at 125 kb/s.

2. CAN communications are performed in a unit called a frame. CAN frames can be up to 8 bytes in length.

3. CAN provides extensive error correction, including bit monitoring (comparing transmitted bits to received), bit stuffing, CRC checksum, acknowledgment by all receivers, frame check (verify length), automatic retry, and fault confinement (defective devices automatically shut off).

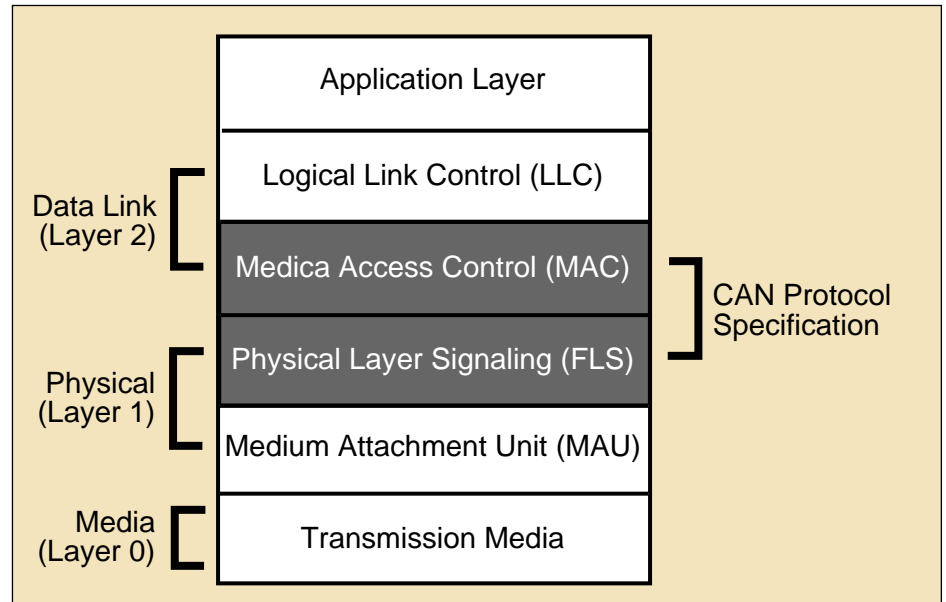


Figure 5 – CAN defines parts of the Physical and Data Link Layers of the OSI 7 layer com model .

4. Access to the CAN network is performed using a method called nondestructive bitwise arbitration. In this system, when a CAN node wants to send a frame, it waits for the bus to become idle, then starts its frame with an arbitration identifier (ID). Because of the underlying physical layer, a dominant bit (0) always overrides any recessive bit (1). As a node is writing its bits to the bus, it also reads the bus to determine if the bit on the bus is different than the bit written by the node. If the bits are different, the node stops its write because some other node has higher priority to the bus. Thus, the arbitration ID determines the priority of messages on the bus, with lower IDs having higher priority.

The industrial protocols built upon CAN add further specifications in such areas as wiring types, connectors, diagnostics indicators, configuration switches, and hot-swapping capability. Of great importance with such networks is the definition of objects for organizing data within devices as well as for defining classes of devices, such as switches, motor starters, and I/O systems. These higher-level software definitions add the ease of use and standardization to

enable CAN's widespread use in industrial automation applications.

Industrial Automation Tutorial

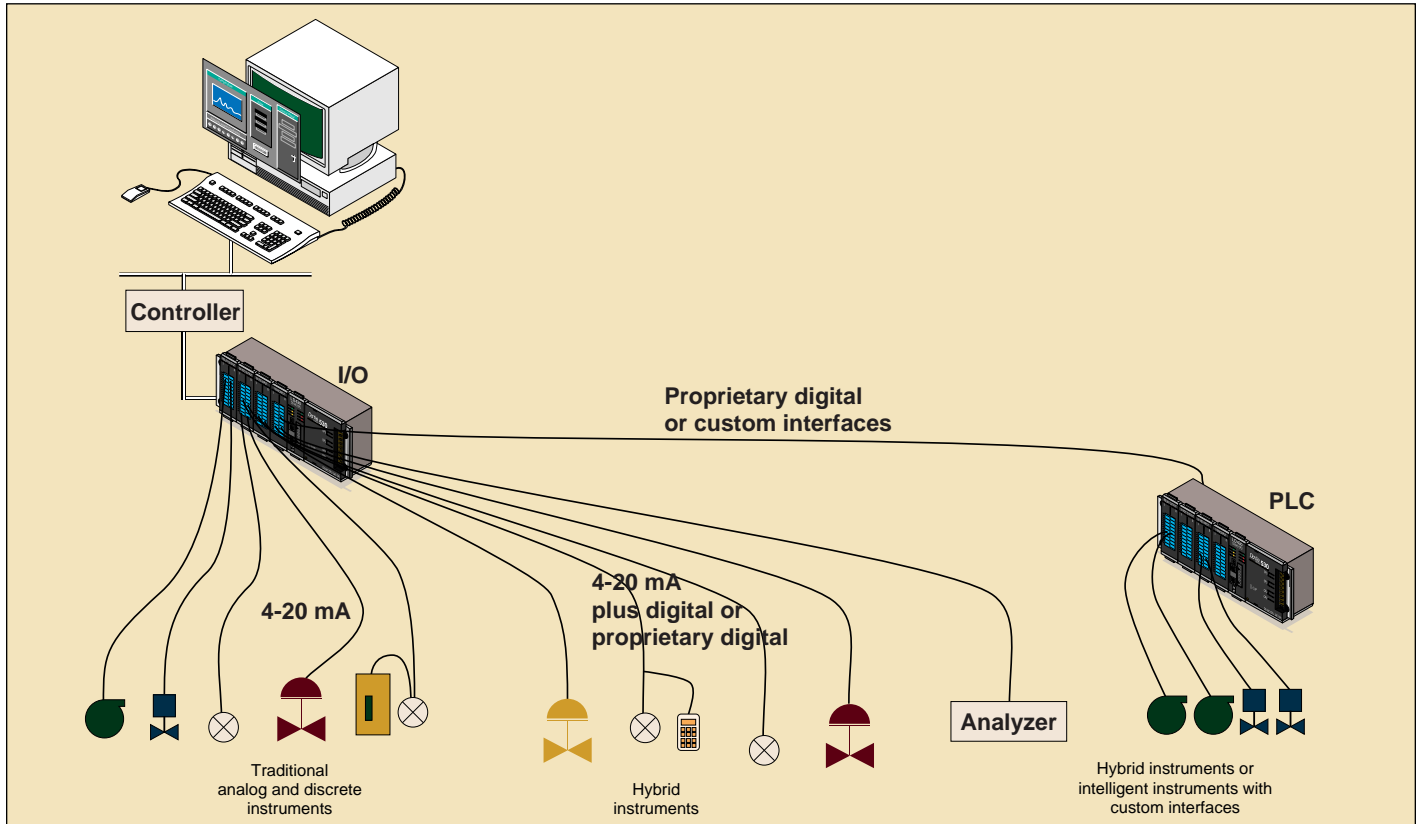


Figure 6a. Today's control systems use a combination of point-to-point analog connections and hybrid or proprietary digital communications networks.

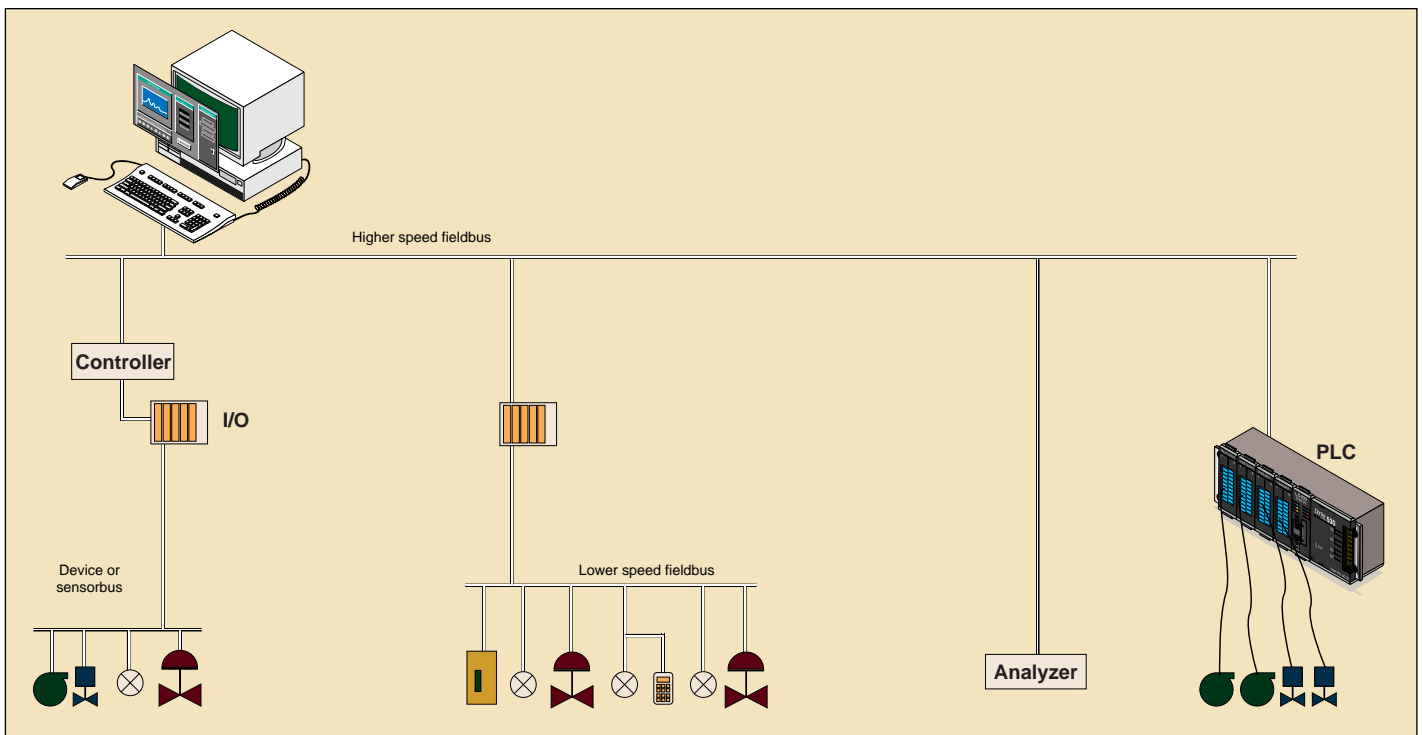


Figure 6b. Using industrial device networks, system architecture is simplified because devices share common protocols.