1 – Fieldbus Layers

FOUNDATION™ Fieldbus has several different "layers." This chapter discusses three of these layers:

1. Physical Layer: The various topologies and types of data blocks used by FOUNDATION Fieldbus.
2. Communication Layer: How Fieldbus uses and assigns device registers.
3. Parameter Classes: The function or role of the information generated on the network.

This chapter provides the background on the how and what of Fieldbus. So let's start. What is Fieldbus?

Fieldbus is a bi-directional digital communication network that enables the connection of multiple field instruments and processes and operator stations (HMI: Human-Machine Interfaces). They carry out control functions and enable monitoring by means of supervision software. Figure 1-1 shows how these three layers (Field, Fieldbus, and Supervisory System) interrelate.

The FOUNDATION Fieldbus protocol was based on the ISO/OSI seven-layer communications model, although it does not include all layers. It can be divided into the Physical Layer (dealing with instrument connection techniques) and the Communication Stack (dealing with the digital communication among the devices). These are the OSI layers used by FOUNDATION Fieldbus. Figure 1-2a represents how the different components of the FOUNDATION Fieldbus protocol maps to the OSI seven-layer model.

The Physical Layer is OSI layer 1, the Data Link Layer is OSI layer 2, and because FOUNDATION Fieldbus is a relatively simple network protocol with little cross-network communication, OSI layers 3 through 6 are not used. The Fieldbus Message Specification and Fieldbus Access Sublayer are part of OSI layer 7, and the Application Layer and the User Layer in which Function Blocks are defined reside above this. The Fieldbus Communication Stack is comprised of layers 2 through 7 of the OSI model.

Figure 1-1 – Digital control system architecture

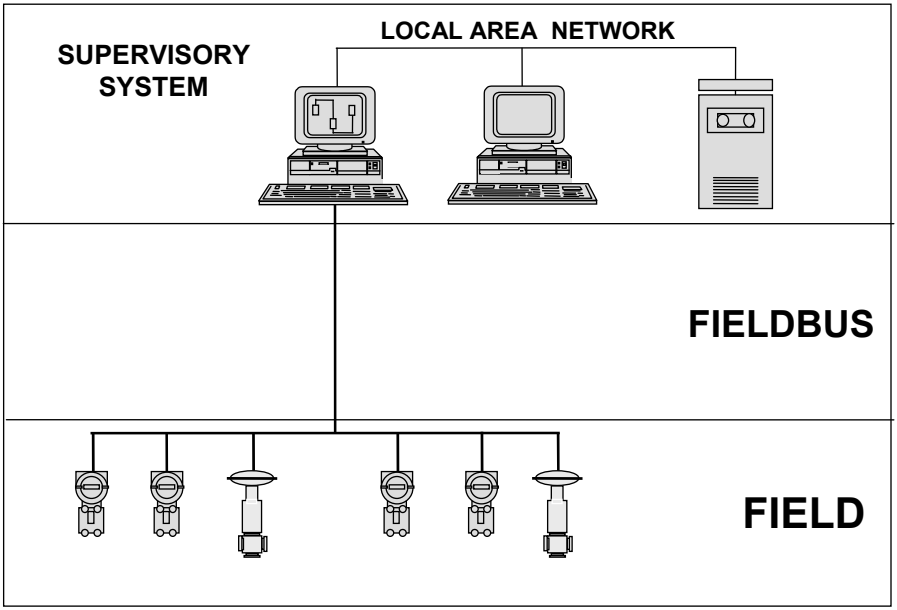
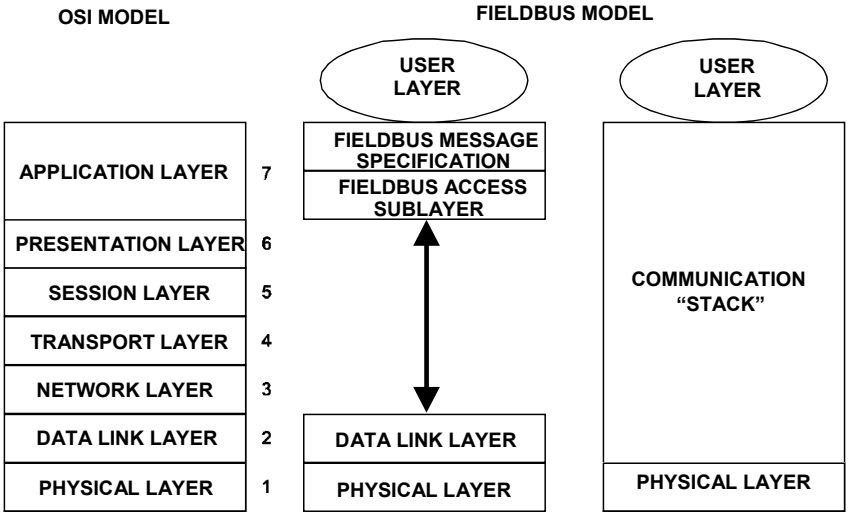


Figure 1-2a – OSI model compared with Fieldbus model



As a message is transmitted from one device to another on the network, it must pass through all of the OSI layers, and in the process, the data packet

is developed, as shown in Figure 1-2b, where the numbers in the figure represent the approximate number of 8-bit octets used to transfer the user data up and down the stack.

Figure 1-2b – Fieldbus data transfer packets

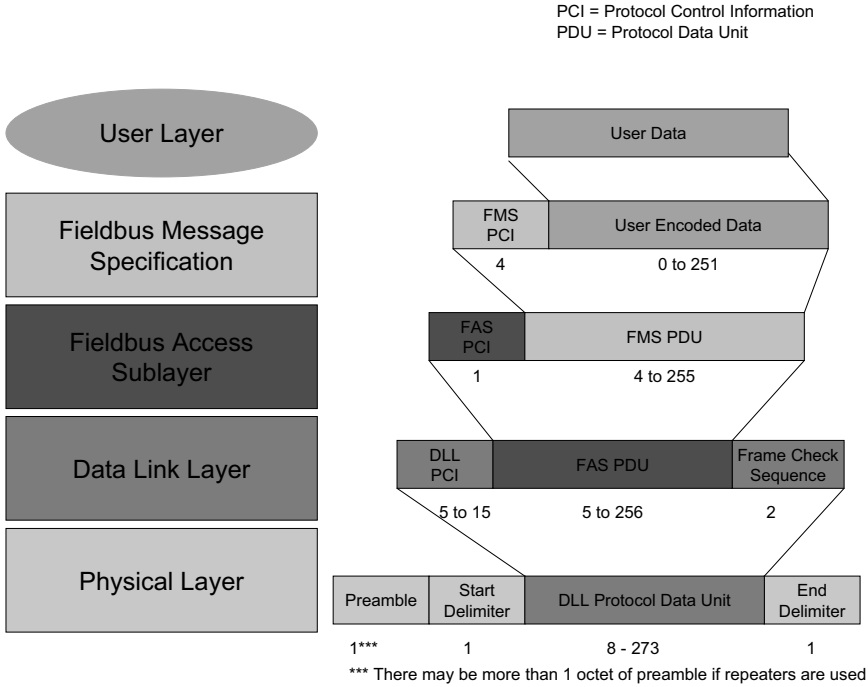
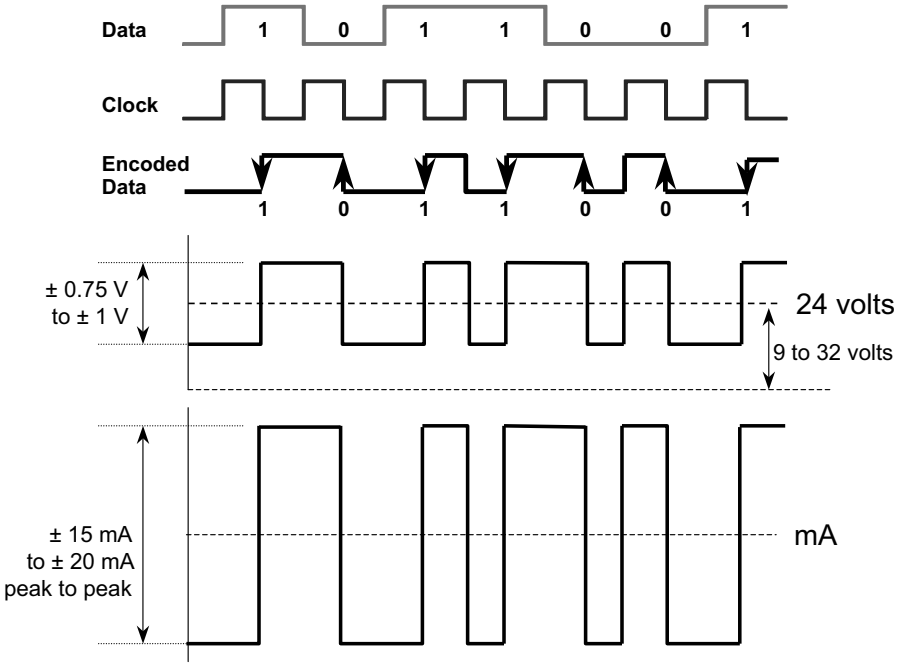


Figure 1-3 represents Manchester encoding, which is how the actual data is encoded in the H1 FOUNDATION Fieldbus network. Manchester encoding adds a time reference signal to the data signal to determine the signal boundaries. One way the protocol increases the level of noise immunity versus other communication techniques is that it looks for a transition every 32 ±10% microseconds to see if there is a change in state, up or down. If there is no change within this "gate," then there is no communication on the network. Because FF only looks for a transition during this short time period, the amplitude of the signal itself is not the critical element in determining if there is a message to send.

Figure 1-3 – Manchester encoding



The Data Link Layer (DLL) is a mechanism to transfer data from a node to the other nodes that need the data. The Data Link Layer also manages the priority and order of such transfer requests, as well as data, address, priority, medium control, and other parameters, all related to message transfer.

Only one device on a link is allowed to use the medium (Physical Layer) at a time. The Link Active Scheduler (LAS) controls medium access.

1.1 Topology

1.1.1 Application Layer

The Application Layer provides an interface for the device's application software. This layer defines how to read, write, or start a task in a remote node. The main task of this layer is to define syntax for the messages.

The main components of the Application Layer are the Fieldbus Access Sublayer (FAS) and the Fieldbus Message Specification (FMS).

The FAS uses the scheduled and unscheduled features of the Data Link Layer to provide a service for the Fieldbus Message Specification (FMS). The types of FAS services are described by Virtual Communication Relationships (VCR).

The VCR is like the speed dial feature on your memory telephone. There are many digits to dial for an international call–an international access code, country code, city code, exchange code, and the specific telephone number. This information only needs to be entered once and then a "speed dial number" is assigned. After setup, only the speed dial number needs to be entered for dialing to occur.

In a similar fashion, after configuration, only the VCR number is needed to communicate with another Fieldbus device.

Just as there are different types of telephone calls, such as person-to-person, collect, or conference calls, there are different types of VCRs. VCRs and their management are covered in more detail in Chapter 5.

Fieldbus Message Specification (FMS) services allow user applications to send messages to each other across the Fieldbus using a standard set of message formats.

FMS describes the communication services, message formats, and protocol behavior needed to build messages for the User Application.

Data that is communicated over the Fieldbus is described by an "object description." Object descriptions are collected together in a structure called an object dictionary (OD).

The object description is identified by its index in the OD. Index 0, called the object dictionary header, provides a description of the dictionary itself and defines the first index for the object descriptions of the User Application. The User Application object descriptions can start at any index above 255.

Index 255 and below define standard data types such as Boolean, integer, float, bitstring, and data structures that are used to build all other object descriptions.

A Virtual Field Device (VFD) is used to remotely view local device data described in the object dictionary. A typical device will have at least two VFDs: a Network and System Management VFD and a User Application VFD.

Network Management is part of the Network and System Management Application. It provides for the configuration of the communication stack. The Virtual Field Device (VFD) used for Network Management is also used for System Management, and provides access to the Network Management Information Base (NMIB) and to the System Management Information Base (SMIB). NMIB data includes Virtual Communication Relationships (VCR), dynamic variables, statistics, and Link Active Scheduler (LAS) schedules (if the device is a Link Master). SMIB data includes device tag and address information and schedules for Function Block execution.

1.1.2 User Layer

The User Layer defines the way of accessing information within Fieldbus devices so that such information may be distributed to other devices or nodes in the Fieldbus network. This is a fundamental attribute for process control applications.

The architecture of a Fieldbus device is based on blocks, with the Function Block, which as the name implies is an object-based function designed to execute a range of control functions that are responsible for performing the tasks required for the current applications, such as data acquisition, feedback and cascade loop control, calculations, and actuation. Every Function Block contains an algorithm, a database (inputs and outputs), and a user-defined name, typically the loop or tag name since the Function Block tag number must be unique in the user's plant). Function Block parameters are addressed on the Fieldbus by means of their TAG.PARAMETER-NAME.

A Fieldbus device includes a defined quantity of Function Blocks of which at least one block must be instantiated or defined.

Function Block. The FOUNDATION Fieldbus Function Block, especially its models and parameters–through which you can configure, maintain, and customize your applications–is a key concept of Fieldbus technology.

What is a Function Block? A Function Block is a generalized concept of the functionality in field instruments and control systems, such as analog input and output as well as PID (Proportional-Integral-Derivative) control. The FOUNDATION Fieldbus specification, FF-890, "Function Block Application Process–Part 1," gives fundamental concepts, while Part 2 and later parts give various Function Block details.

The Function Block Virtual Field Device (VFD) contains three classes of blocks: Resource Block, Function Block, and Transducer Block.

Resource Block. A Resource Block shows what is in the VFD by providing the manufacturer's name, device name, Device Description (DD), and so on.

The Resource Block controls the overall device hardware and Function Blocks within the VFD, including hardware status.

Tip 1 – The mode of the Resource Block controls the mode of all other blocks in the device.

Transducer Block. A Function Block is a general idea while the Transducer Block is dependent on its hardware and principles of measurement. For example, a pressure transmitter and magnetic flow meter use different measurement principles but provide an analog measured value. The common part is modeled as an AI (Analog Input) Block. The difference is modeled as Transducer Blocks, which provide the information on the measurement principle. A Transducer Block is linked to a Function Block through the CHANNEL parameter of the Function Block.

In addition to converting the signal between a digital number and a physical signal (milliVolts, capacitance, frequency etc.) or output (pressure, current, etc.), Transducer Blocks are becoming ever more important because they are also the blocks used to capture and store all the diagnostic and maintenance-related data for a device. A number of Standard Transducer

Blocks have been defined, including the Common Block (to define the minimum requirements for all Transducer Blocks) and Temperature, Pressure, and Advanced Positioner Blocks. The Advanced Positioner Block is a requirement for partial stroke testing, which is needed for Safety Instrumented Fieldbus applications. The Flow Transducer Block is likely to be released in 2012.

It is end-user demand and economics that are driving the need for Standard Transducer Blocks since, without a standard interface to the maintenance data contained within each device, it is a cumbersome task to take full advantage of the diagnostic capabilities of a digital transmitter, using modern software and asset management systems.

Transducer specifications are generally defined by the device developers. The transducer specifications establish the base scope of transducer functions. A device may have additional functions, but it **must** contain the functions specified in the specification to be interoperable within the given specification.

Function Block. A Function Block is a generalized model of measurement and control. The three Function Block classes are:

1. Standard Block, as specified by the Fieldbus Foundation.
2. Enhanced Block, a Standard Block with additional parameters and algorithms but still fully defined by the appropriate FF specifications
3. Extended, Open Block or a Vendor-Specific Block, designed by individual vendors with parameters not defined by the FF specifications but rather by the device DD file. Extended blocks must contain the Standard Block parameters so basic connectivity and communications will always be possible.

The Function Blocks MAI (Multiple Analog Input), MAO (Multiple Analog Output), MDI (Multiple Discrete Input), MDO (Multiple Discrete Output), and FFB (Fully Flexible Function Block), defined in Parts 4 and 5 of the Function Block Application Process specifications, were developed as part of the High-Speed Ethernet (HSE) process. The "M-series" of blocks are able to transfer a group of eight PV (process variable) signals as a single message on the Fieldbus Network and because HSE is fully backwards com-

patible with H1, a number of H1 devices, such as temperature multiplexers, are taking advantage of the MAI block.

The most novel of the new blocks, however, is the Fully Flexible Function Block (FFB), as it is able to be fully programmed by the end user, using any of the IEC 61131-1 programming languages.

Like all object-based Fieldbus Function Blocks, the FFB is a "wrapper" for the actual functions that reside and execute inside of it. The Fieldbus specifications define a set of parameters that must be common to all Function Blocks to ensure interoperability and communications between the various blocks, devices, and host system. Since each component of the Fieldbus specification is treated as an object and is, to some extent, similar to a subroutine or function call in a computer program, it is possible for each manufacturer to write its own code for the object to execute, as long as the results are presented in the predefined format. It is this lack of definition for the function itself that makes the FFB possible.

The FFB can be configured by the end user with any of the IEC 61131-1 languages to whichever function is required. Thus, a device supporting the FFB can be configured or programmed for a variety of purposes, from protocol converter to a nano-PLC that performs batch/recipe operations or complex multivariate control calculations, such as artificial neural networks or fuzzy logic.

The FFB specification contains many useful Function Blocks; however, the one developed to help Fieldbus in the manufacturing industry, where discrete control is more prevalent, is the device controller (DC) block, which is intended to control any two- or three-state physical device. The device controller accepts a set point and causes the device to drive to that set point. Time is allowed for the transition, but alarms are generated if the physical device fails to reach the desired state or loses that state after the transition is complete. The DC block has inputs for control of the set point by external logic or commands from a host, as well as permissive, interlock, and shutdown logic functions. An operator may temporarily bypass a faulty limit switch after visual confirmation of the state of the physical device. The parameter DC_STATE displays one of 14 states that describe the current control condition, while the parameter FAIL gives specific reasons for failures.

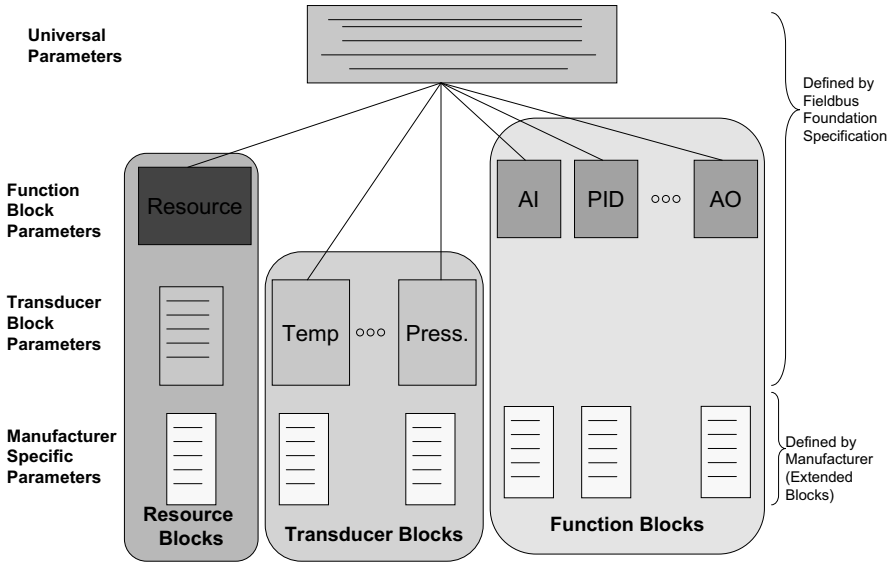
Unfortunately, the interfaces to program FFB are not yet fully interoperable. This means that an FFB from Manufacturer A must be programmed and configured by the host and software tools of Manufacturer B, and vice versa. However, once the FFB has been prepared and compiled through DD Services (the binary file that is used by field devices and hosts to execute the information from the DD file), it can be executed by any system supporting the FFB block type.

FFB technology was successfully demonstrated at the International Specialty Products facility in Lima, Ohio, in May 2005. The demonstration consisted of converting one of the three filter beds in the process from control in the host to field-based control, using linking devices containing FFBs from two manufacturers. The first FFB controlled the 10 quick opening/closing valves (250 milliseconds) on one side of the filter, and then control was transferred to the second linking device and its FFB to control the second bank of 10 valves. After that, control was passed back to the host to control the third filter bed's operation.

Figure 1-4 shows not only how the various function blocks work together but also the different parameters that are used in each of the Standard, Enhanced, and Extended Blocks available in a device. Simplistically, the Universal parameters define the basis for the Standard Blocks, Enhanced Blocks build on this concept, and then manufacturers can further expand on the Enhanced Blocks with their own enhancements.

Tip 2 – The Function Block extensions provided by manufacturers are not defined by the Foundation, so they may not be the same between two different manufacturers.

Figure 1-4 – Device description hierarchy



Despite the fact that the enhancements are not defined by the Fieldbus Foundation, they will be supported by all host systems capable of reading the associated DD and Capabilities Files.

A block has a series of parameters, which have continuous indexes. It is because these indexes are continuous that the DD revisions for devices must match between a field device and its associated Host system. If a newer DD revision device is associated with an older DD revision in the Host, the links between where the Host thinks a parameter resides in memory and the actual device memory address no longer match. For example if the AO Block in DD revision 01 starts at memory address 600 when revision 02 is published, the AO Block may now start at memory address 645 meaning that if the Host system is looking for the parameter at address 600, it is likely to get a data type mismatch as a minimum and certainly the incorrect data.

Table 1-1 shows these various blocks as defined by the Fieldbus Foundation in the indicated part of the specification.

Table 1-1 – Fieldbus Function Blocks

Part-2 Blocks	**Standard Blocks**
AI	Analog Input Block
DI	Discrete Input Block
ML	Manual Loader Block
BG	Bias/Gain Station Block
CS	Control Selector Block
PD	P, PD Controller Block
PID	PID, PI, I Controller Block
RA	RATIO Station Block
AO	Analog Output Block
DO	Discrete Output Block
Part 3	**Enhanced Blocks**
DC	Device Control Block
OS	Output Splitter Block
SC	Signal Characterizer Block
LL	Lead Lag Block
DT	Dead Time Block
IT	Integrator (Totalizer) Block
(More blocks are under development)	
Part 4	**Multiple I/O Blocks**
MDI	Multiple Discrete Input Block
MDO	Multiple Discrete Output Block
MAI	Multiple Analog Input Block
MAO	Multiple Analog Output Block
Part 5	
FFB – Flexible Function Blocks	IEC 61131 Blocks

As Figures 1-4a through 1-4c show, a Function Block has input, output, and contained parameters. Figure 1-4a is a typical Analog Input (AI) Block, 1-4b is an Analog Output (AO) Block, and 1-4c is a PID Block. Data generated in a block is made available from an output parameter, which can be linked to the input parameter of other Function Blocks. The Fieldbus Foundation does not define how each manufacturer is to implement the algorithms to complete the functions shown in these figures, but rather defines the func-

tionality, parameters, and “connections,” such as channel out, etc., between each of the Function Blocks. This allows each manufacturer to differentiate their product from their competitors’ through increased efficiency and features, such as improved signal conditioning and diagnostics.

Floating-point parameters have a valid range of $\pm 1.2 \times 10^{38}$ to $\pm 3.4 \times 10^{38}$, as well as (in some cases) the values -Inf or +Inf.

Discrete Blocks have 256 valid enumerated states, which means that in addition to simple logic 0 or 1, they can also be used to represent specific states, such as open, closed, true, false, start, stop, running, etc.

In the case of an AO Block, if the actual device reaches either of its open or closed limits, the block will set the corresponding limit in the status element of the associated back-calculation output parameter.

This tells the PID Block to not push the output further in that direction, thus preventing reset windup in the loop.

The operator normally sets the PID loop mode from the Block mode parameter in the PID Block. Control is stopped by setting this parameter to MAN. However, if the operator wishes to “hand operate” the AO Block, it is better to remain in AUTO mode and enter the set point instead.

Figure 1-4a – Analog Input Block (AI)

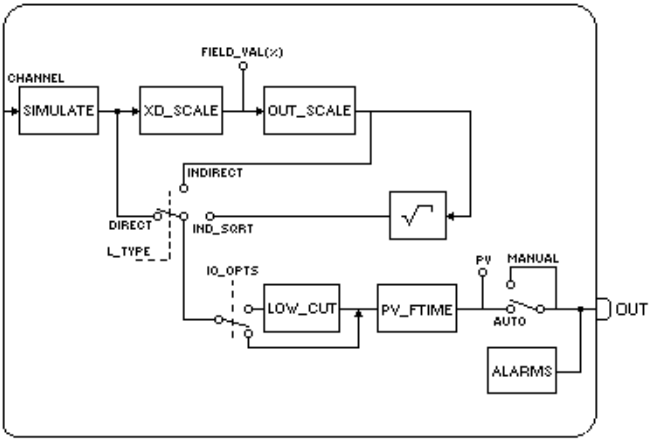


Figure 1-4b – Analog Output Block (AO)

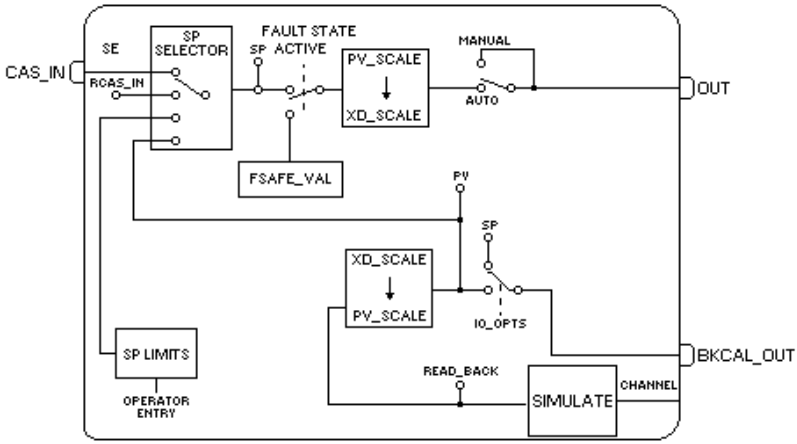
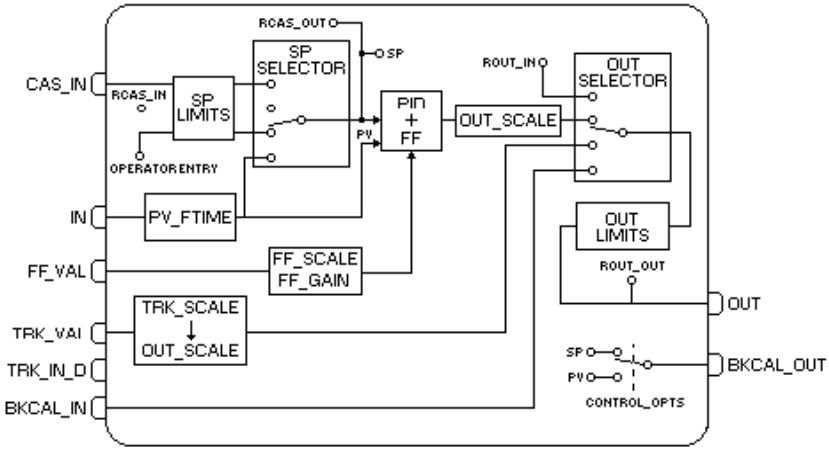


Figure 1-4c – PID Block (PID)



1.1.3 Testing and Registration

All FOUNDATION Fieldbus devices that have a DD file need to pass two separate suites of tests before they can achieve the Fieldbus "check mark" of the Fieldbus logo with a check mark in the lower right-hand corner, which confirms compliance with the relevant suite of test specifications. One of the FF "check mark" tests is for the Communications Stack, while the other tests for device interoperability. Although it is not an exact indication of

what each of these test suites checks, a simplified way of thinking of the two test suites is that the Conformance test checks for media access/control OSI Layers 2-7, while the Device Interoperability Test or ITK checks device conformance to the User Layer.

The Fieldbus Foundation has partnered with the Fraunhofer Institute, based in Karlsruhe, Germany, for completion of the Conformance Test System that checks the Communications Stack, which does not change much over time. Every manufacturer then uses an approved stack to build their Fieldbus device, which is then sent to the Fieldbus Foundation for interoperability testing.

1.1.4 Interoperability Test System

The ITK is conducted at the Foundation offices in Austin, Texas, where the device is connected to the test suite so that approximately 500 different tests can be run. The pass rate for these tests is 100%, so if just one test fails, the manufacturer, after making any modifications to correct the problem(s), needs to repeat all the tests. Devices that successfully complete the tests are given the FF "Check Mark" and are then listed on the Fieldbus web site as such, along with the DD file that was used for the test. A device that passes ITK 5.1 and supports the Field Diagnostics Profile to support enhanced Device Description and graphics will have this feature separately listed on the registration certificate.

The ITK is normally revised approximately every 18 months and as of late 2011 is on revision 6. Prior to any FF product receiving a "check mark" it must go through a rigorous quality assurance process to verify that any products getting this approval are truly interoperable.

The Fieldbus Foundation test procedure must pass through all the steps below before the specification is approved:

1. A need for the specification is identified from a request by industry, and once the Board of Directors agrees that this is a worthwhile activity, a call for participants is distributed to all Foundation members. This call for volunteers also includes a request for volunteers to lead the activity as Editor, Project Leader, etc.

2. Once the new standard development committee has been formed, the committee meets and agrees on the project scope, the project leader prepares a project plan, outlining the deliverables and estimated timeline for the project to the Technical Steering Committee for review and approval.

3. The first step in the actual development process is for the development team to create a set of Use Cases to clearly define the problem or problems that are going to be solved by the new standard.

4. From these Use Cases the team develops a Draft specification that fully describes how the products can be built to solve the Use Cases.

5. The Foundation issues a "Call for Prototypes" to request that at least two independent suppliers build products in compliance with the new Draft Preliminary Specification.

6. The prototypes and the Foundation's test kit are then brought together to test against this specification to make sure that everyone interprets the specification in the same way, and once they have done so, with any resulting questions being resolved by the development team, it is accepted that the theory described in the specification will actually work. It is this step in the specification approval process that is key in verifying device interoperability because it confirms by three separate sources that they have all interpreted the specifications in the same way and met the requirements as defined in the original Use Cases.

7. A Preliminary Specification is then made ready for distribution to members and the Technical Steering Committee for review and final approval.

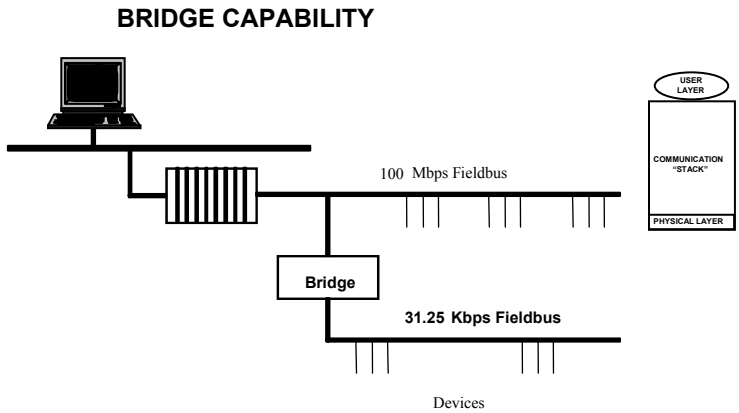
8. The Technical Steering Committee (TSC) (Fieldbus Foundation's Standards Board) reviews any comments received, and after all the Fieldbus Foundation members have a final chance to comment, the TSC then approves the document as a Final Specification.

This entire process typically takes more than two years.

1.1.5 Physical Layer

This is the Fieldbus layer connected with instrument devices in the field. The standardized data transmission speed of the H1 network is 31.25 Kbps; as stated by the standard, all other speeds shall be used for high-speed interconnection of bridges and gateways (see Figure 1-5).

Figure 1-5 – Fieldbus bridge capability



The standard determines the following rules (among others) for the speed of 31.25 Kbps:

1. A Fieldbus instrument shall be able to communicate with the following number of instruments:
   - From 2 to 32 instruments in a non-intrinsically safe connection and power supply separated from the communication wiring.
   - From 2 to 16 instruments receiving power by the same communication wiring in an intrinsically safe connection.
   - From 1 to 24 instruments receiving power by the same communication wiring in a non-intrinsically safe connection.

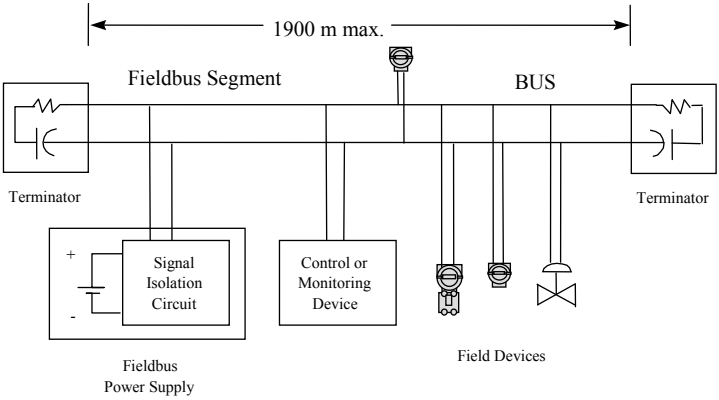
Note: *Most host systems are restricted to 16 devices per network (64 devices per 4 port card) or are otherwise restricted by the number of parameters they can manage per port. The result is that physical power is not always the limiting factor in the*

number of devices that can be added to a network. More often the limiting factor for fast macrocycle installations is bandwidth.

Tip 3 – This rule does not forbid the connection of more instruments than the specified number. Such limits have been established, considering a consumption of 9 mA ± 1 mA, with a power supply of 24 VDC, intrinsic safety barriers with an output of 11–21 VDC, and a maximum current of 80 mA for the instruments located within the hazardous area.

2. The length of the entire bus segment with the maximum number of instruments operating at a speed of 31.25 Kbps shall not exceed 1900 m in the section of the trunk plus all spurs (Figure 1-6).

Figure 1-6 – Maximum length of Fieldbus network

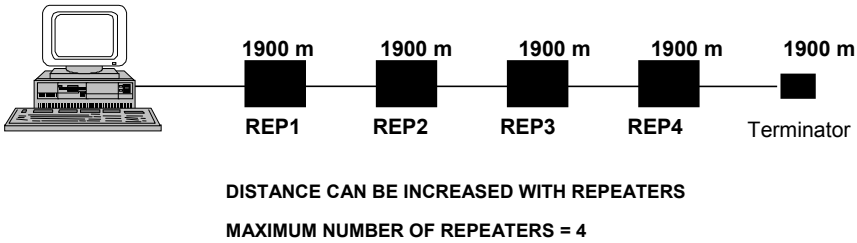


Tip 4 – This rule does not forbid the use of longer lengths, provided that the electrical characteristics of the instruments are observed.

3. The maximum number of repeaters for regenerating the waveform between two instruments cannot exceed four (Figure 1-7).

Repeaters are used to expand a Fieldbus network. Repeaters can be either energized or de-energized. When four repeaters are used, the maximum distance between any two devices in a segment is 9500 m.

Figure 1-7 – Fieldbus network with repeaters



4. A Fieldbus system must be able to remain in operation while a device is being connected to it or disconnected from it.
5. Failure in any communication element (except for short-circuit or low impedance) shall not affect communication for more than 1 ms.
6. The polarity of systems using twisted pairs shall be respected, their wires shall be identified, and the polarity shall be observed in all connection points. According to the Fieldbus standard, devices themselves are not to be polarity-sensitive, but this is not always the case.
7. For systems with a redundant physical medium:
   - Each channel must comply with the network configuration rules.
   - A non-redundant segment cannot be between two redundant segments.
   - Repeaters must also be redundant.
   - The identification number of the channels must be maintained in the Fieldbus, that is, the Fieldbus channels must have the same numbers as the physical channels.
8. Cable shields shall not be used to conduct power.

1.1.6 Topologies

Several topologies may be used in Fieldbus design. Figure 1-8 shows the topologies that will be detailed below. Power supplies and terminators are not shown so that the figures can be more easily understood.

Bus with Spurs Topology. The Bus with Spurs topology uses a single bus to which devices and spurs are directly connected. Several devices may be connected to each spur. The total spur length is limited according to the number of spurs and the number of devices per spur. This is summarized in Table 1-2. This spur length table is not absolute. It merely serves as a guideline for designing networks.

Figure 1-8a – Physical layouts – Single combined segment

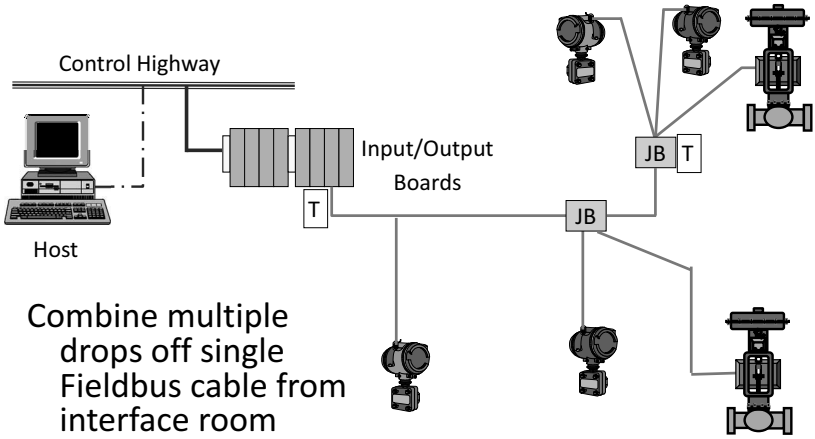
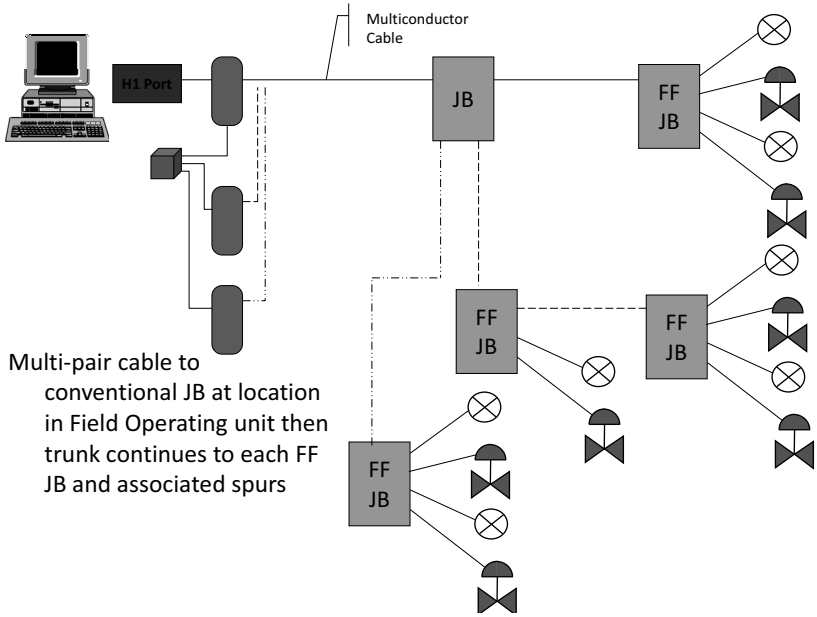


Figure 1-8b – Wiring practices – Cable efficiency



Point-to-Point Topology. In the Point-to-Point or Daisy Chain topology, all devices used in the application are connected in series. The Fieldbus trunk is routed from one device to the next, being interconnected to the terminals of each Fieldbus device.

Table 1-2 — Maximum spur lengths

Total devices per network	One device per spur, m (ft)	Two devices per spur, m (ft)	Three devices per spur, m (ft)	Four devices per spur, m (ft)	Maximum total length, m (ft)
1–12	120 (394)	90 (295)	60 (197)	30 (98)	439 (1440)
13–14	90 (295)	60 (197)	30 (98)	1 (3)	384 (1260)
15–18	60 (197)	30 (98)	1 (3)	1 (3)	329 (1080)
19–24	30 (98)	1 (3)	1 (3)	1 (3)	220 (720)
25–32	1 (3)	1 (3)	1 (3)	1 (3)	10 (32)

Caution: *Point-to-Point topology is rarely used since the failure of one device in the network will result in total network failure.*

Tree Topology. Tree topology concentrates the connection of several field devices to couplers/junction boxes. Because of its distribution, the Tree topology is also known as a "Chickenfoot" or Star configuration.

End-to-End Topology. End-to-End topology is used to directly connect two devices. The connection may be entirely located at the field (a transmitter and a valve with no other devices connected) or to connect a field device (a transmitter) to the Host device.

Mixed Topology. Mixed topology, as the name implies, mixes the three most commonly used topologies connected to one another. However, the maximum length of a segment, including the spurs to the total length, shall be observed. Figure 1-8a shows how this topology might be configured by combining individual spurs with several multiple drop field device couplers.

Many installations are also taking this one step further by running multi-conductor H1 cables to a conventional junction box at a convenient location, either on the edge or centrally located within a unit operation area, and then extending individual trunk cables to one or more Fieldbus device coupler assemblies/enclosures strategically located in closer proximity to the devices themselves, thus minimizing total installed cable cost. This is shown in figure 1-8b.

Bridges are used to connect Fieldbus segments operating at different speeds (and/or physical layers such as wires, fiber optics, radio, etc.) in order to obtain a large network. A bridge is shown in Figure 1-5, connecting the H1 and High-Speed Ethernet (HSE) networks.

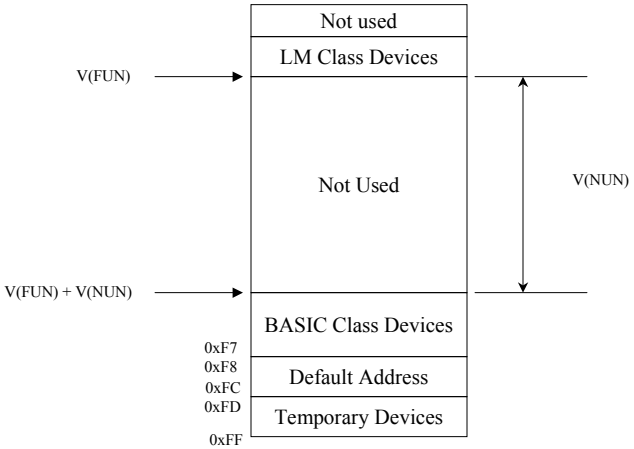
A gateway depending on the manufacturer can be used to connect one or more H1 segments to other types of communication protocols, such as Ethernet, RS-232, MODBUS, Modbus/TCP, Profibus, etc.

1.2 Communications

Figure 1-9 shows how the registers in a Fieldbus device are assigned.

Figure 1-9 – FOUNDATION Fieldbus node addresses

0x10 — V(FUN)	Address for Link Master Class devices
0xF7 — V(FUN) + V(NUN)	Address for Basic Class devices
0xF8 — 0xFC	Default address for devices with cleared address
0xFD — 0xFF	Address for temporary devices such as a handheld communicator



Every Fieldbus device has a unique 32-bit hardware address identifier made up of a 6-byte manufacturer code, a 4-byte device code, and a serial number. This makes it possible to uniquely distinguish each device from the others. The Fieldbus Foundation assigns the manufacturer codes, while the manufacturer assigns the device type code and sequential serial number.

A temporary device, such as a handheld communicator, has a node address in the temporary range. The Link Active Scheduler (LAS) has a node address of 0x04 or the series of lowest addresses. If a device has an address in the gap V (NUM), it will never be able to join the network. The V (FUN) and V (NUN) parameters are accessible through Network Management.

Several Data Link (DL) addresses are reserved for specific purposes. For example, devices can share the same system-wide Data Link Service Access Point (DLSAP) for alarm reception.

Foundation devices are classified into device classes: BASIC, Link Master (LM), and Bridge. An LM class device has the ability to be the LAS, while BASIC class devices do not have this functionality. In addition to LM capability, a Bridge class device has the functionality to connect networks.

One and only one device in a network can be the LAS at any one time; therefore, at least one LM (or Bridge) class device is needed in a link. LM devices will try to acquire the LAS role when no LAS exists on start-up or when the current LAS fails.

Tip 5 – The LM device with the lowest node address becomes the new LAS for the network.

Other secondary or Backup LM devices observe the LAS activity and can assume the primary or Master LAS role if the operating LAS fails.

The LAS manages the scheduled communication part of the synchronized data transfer between Function Blocks.

A Function Block output parameter is a *Publisher* of data, and other Function Blocks that receive this data are called *Subscribers*. The LAS controls periodic data transfer from a Publisher to Subscribers using the Network Schedule.

Other network communications take place in an asynchronous way. The LAS is responsible for giving all nodes on a link a chance to send messages.

The third role of the LAS is to maintain network communications. The LAS does this by giving the token to all devices detected by the LAS. When a new device is added to the network, it must be recognized by the LAS and added to the token rotation list, which is called the *Live List*.

A Fieldbus device may have user applications that are independent from each other and do not interact. A Fieldbus device consists of Virtual Field Devices (VFD) for such individual applications.

A Fieldbus device has at least two VFDs. One is the Management VFD, where network and system management applications reside. It is used to configure network parameters, including VCRs, as well as to manage devices in a Fieldbus system. The other is a Function Block VFD, where Function Blocks exist. Most field devices have more than two Function Block VFDs.

A measurement or control application consists of Function Blocks connected to each other. Function Blocks are connected through Link Objects in the Function Block VFD. A Link Object connects two Function Blocks within a device, or a Function Block to a VCR for Publisher or Subscriber.

A Function Block must get input parameters before its algorithm can be executed. Its output parameters must be published after algorithm execution. Therefore, algorithm execution and Publisher-Subscriber communication must be orchestrated when blocks are distributed among devices. The System Management and Data Link Layer cooperate to achieve this by using the Link Scheduling (LS) time that is distributed and synchronized by the Link Active Scheduler (LAS).

Other network communications take place in an asynchronous way. The LAS is responsible for giving all nodes on a link a chance to send messages. This asynchronous or acyclic communication should constitute the majority of the macrocycle time on a network. Figure 1-10a shows a typical network with two independent loops.

Note that any device that is either not performing an internal calculation or participating in a publish/subscribe communication, which means it is part of the control loop and must either publish (share) its process variable or subscribe (read) the process variable from another device in its loop, is able to receive the Pass Token and participate in a client server communication.

Figure 1-10a – Function block scheduling and macrocycle

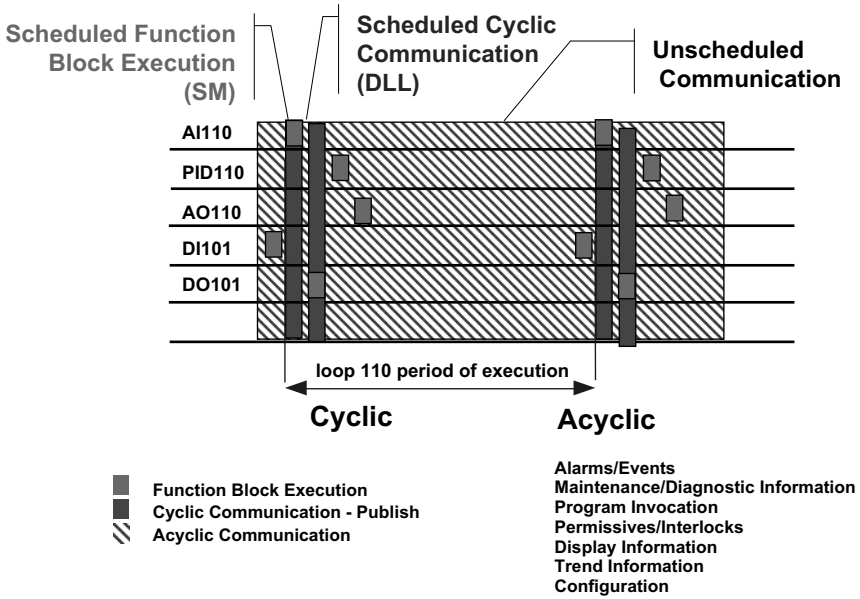
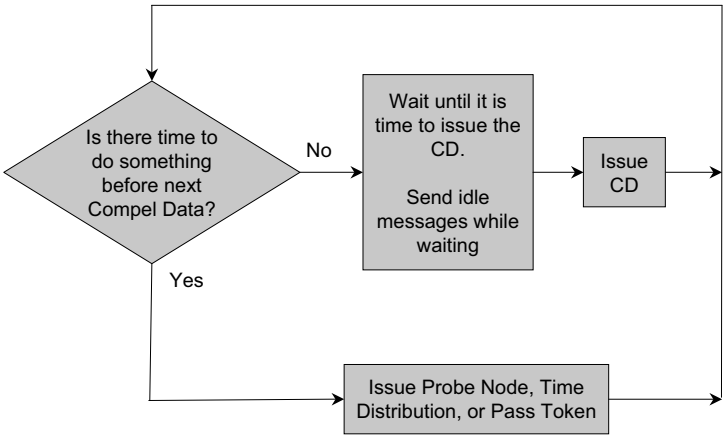


Figure 1-10b represents the algorithm used by the LAS to determine the next action it needs to take while ensuring that all deterministic communications happen at their assigned time.

Figure 1-10b – LAS algorithm



1.3 Parameter Classes

Block parameters are classified into three classes: input, output, and contained parameters. Function Blocks can have all of these classes, while the Resource Block and Transducer Blocks have only contained parameters.

An *input parameter* is an input of a Function Block and can accept one output parameter of another Function Block. Its data type must be the same as the output parameter.

An *output parameter* is a record consisting of a value (analog or discrete) and its status.

A *contained parameter* is neither input nor output. It is accessible only through an on-demand Read or Write request. Its data type can be any of those defined by the Fieldbus Foundation.

Input parameters, output parameters, and some contained parameters are records with an associated status. Status shows whether the value of this parameter is useful or not. If the value is useful, the status is GOOD. If the value is not useful, the status is BAD. The status can be UNCERTAIN when the block is not 100% confident that the value is useful. Blocks have an option to interpret UNCERTAIN as GOOD or BAD.

The acyclical traffic time may be user-defined and configured by means of the configuration software (typical macrocycle times are 250 ms).

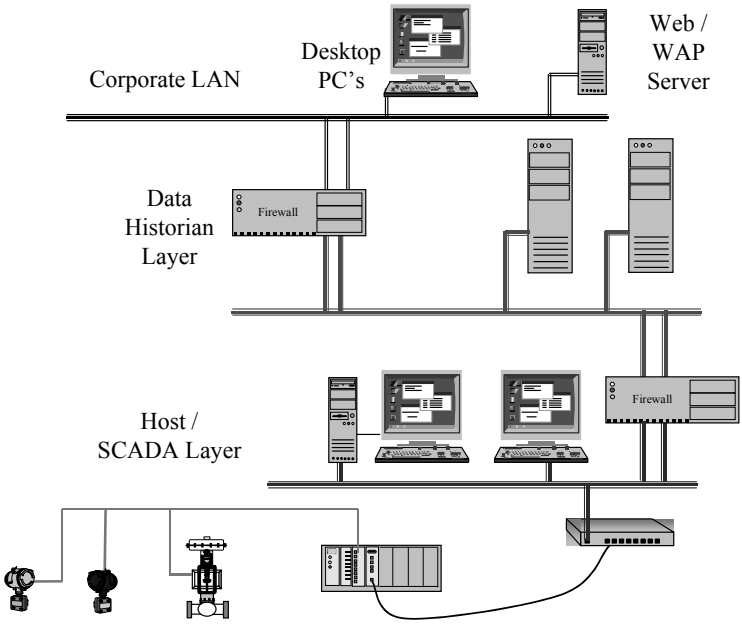
The above discussion concentrates on communications at the device or H1 level. However, all this data must be shared with other parts of the corporation, including the Distributed Control System[1] (DCS), panel operators via the Human-Machine Interface (HMI), and other parts of the corporation as appropriate. However, the security and integrity of the Supervisory Control and Data Acquisition (SCADA) Layer must be ensured and a variety of

1. Historically, DCS has meant Distributed Control System. However, in the new era of digital communications made possible with Fieldbus technology, the more appropriate term is Digital Control System, and this term will be used hereafter in this text.

methods are available to do so. The most common method, however, is to use firewalls and switches.

This is not different from what is done to protect a corporate local area network (LAN) that is connected to the Internet through the use of firewalls, switches, and related technologies. The net effect is just like with the ISA-95 model, there are different "levels" within the enterprise that are connected to each other. Figure 1-11 represents this concept of layers with the lowest layer being the process itself, then layer or level 1 the sensors (this is where the H1 network resides), Layer 2 the controllers, Layer 3 the Historians, and Layer 4 the corporate or business LAN.

Figure 1-11 – SCADA layer Fieldbus traffic management



As shown earlier, "layers" of networks, such as H1 and perhaps an Ethernet-based system for remote I/O and data paths, are within each of these ISA-95 levels as well. Data paths are important to prevent information overload to various components or interfaces/devices on the network. The best example of this is device diagnostic information. The panel operator needs to know only the process measurement and its status; hence all the other

information should be routed through a separate system that can be better integrated with a facility's computerized maintenance system. Figure 1-11 also shows how these two data paths may exist on the same Control/ SCADA Layer. This data "splitting" is possible because Fieldbus uses Publisher-Subscriber relationships to transfer data between memory registers/ devices. In this case, the process data is subscribed to by the operator interface while the diagnostic data is subscribed to by the Condition Monitoring System.

How should all this new information be used? There is a great deal of activity under way on a number of fronts related to, but not part of, the Fieldbus effort that will enable the integration of this information into the larger corporate network environment. Some of these initiatives include:

- OPC – Open Connectivity technology, as developed by the OPC Foundation (www.opcfoundation.org) and formerly known as Object Linking and Embedding (OLE) for Process Control, is a technology to enable real-time transfer of data from one system to another through the use of standardized call functions. The OPC Foundation is developing this open standard.
- XML – Extensible Mark-up Language is the most complex form of language used to display information via the Web. XML is object-based, and therefore each piece of information can potentially be self-defined, thus making it viewable by anyone anywhere.

1.3.1 EDDL

The majority[2] of today's "smart" analog instruments use EDDL (Electronic Device Description Language), often shortened to DD (Device Description) in conversation. EDDL is the base technology underlying the HART (Highway Addressable Remote Transducer), Profibus PA, and FOUNDATION Fieldbus protocols by defining a significant part of their User Layer.

The original EDDL specification as defined by IEC 61804 Part 2 contained little support for graphics; thus, the interface and degree of integration

2. Many discrete devices also use EDDL; however, they are not yet in the majority.

between field devices and hosts was not only limited, but the level of integration and information that could be presented to a user varied with each installation, depending on the host being used.

One technology that was developed to attempt to overcome these limitations was FDT/DTM (Field Device Tool/Device Type Manager). The FDT/DTM technology uses EDDL as the basis for its definitions so that device manufacturers can define the "look and feel" of how the user accesses the device information for other than Process Variable (PV) and related signals, such as those used to configure and calibrate the device.

The HART Communications Foundation, Profibus International, and the Fieldbus Foundation also realized that users required improved interfacing with their devices and as a result have developed, through a joint activity, a number of graphical support functions to EDDL while maintaining 100% backwards compatibility with the original specifications prepared in the early 1990s. The key new capabilities of EDDL include the following:

- Lets images be presented with installation/maintenance information.
- Uses charts and menus to present and input data more consistently.
- Uses graphs as X-Y plots to plot one variable against another, such as a radar waveform (see Figure 1-12) or a valve signature (see Figure 1-13).
- Provides basic mathematics for a first-order equation of the form y=mx+b as also seen in Figure 1-13.
- Offers a history/archive to provide information over time.

Other, less obvious enhancements that will be invisible to end users have also been made to improve the ability of devices and host systems to communicate.

Encouraged by the success of this first project, the three organizations that rely on EDDL technology are now working with the OPC Foundation to add more features and capabilities to EDDL for extension beyond the traditional control system environment.

Figure 1-12 – Radar level gauge

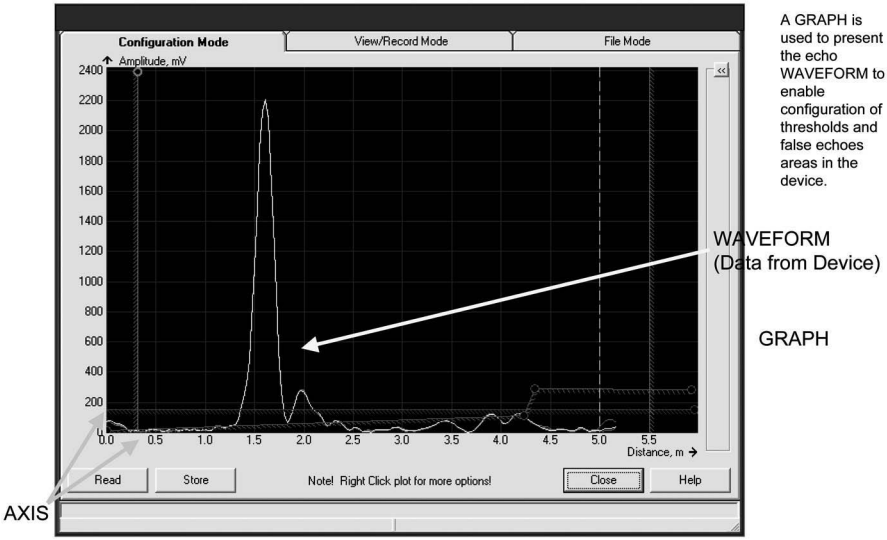
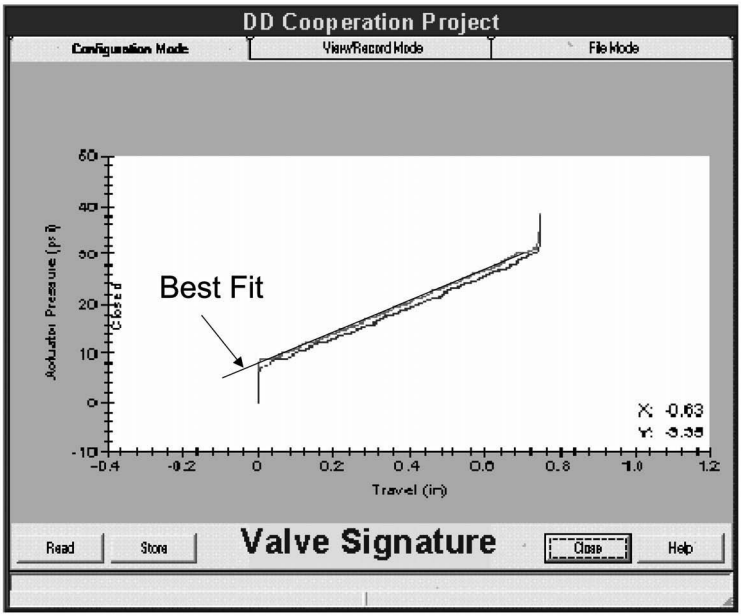


Figure 1-13 – Valve signature with best fit line



1.3.2 FDT/DTM

Enterprise automation requires two main data flows: a “vertical” data flow from enterprise level down to the field devices, including signals and configuration data, and a “horizontal” data flow between field devices operating with the same or different communication technologies.

With the integration of fieldbuses into control and engineering systems, the vertical information flow, such as that associated with the maintenance and set-up devices in automation systems, is increasing.

Although Fieldbus and device-specific software tools exist, there is no unified way to integrate those tools into higher-level, system-wide tools for planning, control, engineering or asset management. FDT (Field Device Tool) has the objective of doing so, and toward that end a series of standards has been prepared under the project IEC 62453/ISA-62453-n-2011 (103.00.nn), where n refers to the various individual parts (mostly different protocols) within the series of standards with this number.

To ensure the consistent management of a plant-wide control and automation technology, it is necessary to fully integrate fieldbuses, devices and subsystems as seamless parts of a wide range of automation tasks covering the whole automation life cycle.

FDT provides an interface specification for developers of components to support function control and data access within a client/server architecture.

The availability of this standard interface facilitates development of servers and clients by multiple manufacturers and supports open interoperation.

A device or module-specific software component called a DTM (Device Type Manager) is supplied by a manufacturer with the related device type or software entity type. Each DTM can be integrated into engineering tools via defined FDT interfaces. This approach to integration is, in general, open for all fieldbuses and thus supports the integration of different devices and software modules into heterogeneous control systems.

As a result of this common interface defined by the standard and Layout Guides, significant savings are available in operating, engineering and maintaining the control systems.

The objectives of the FDT standards are to support:

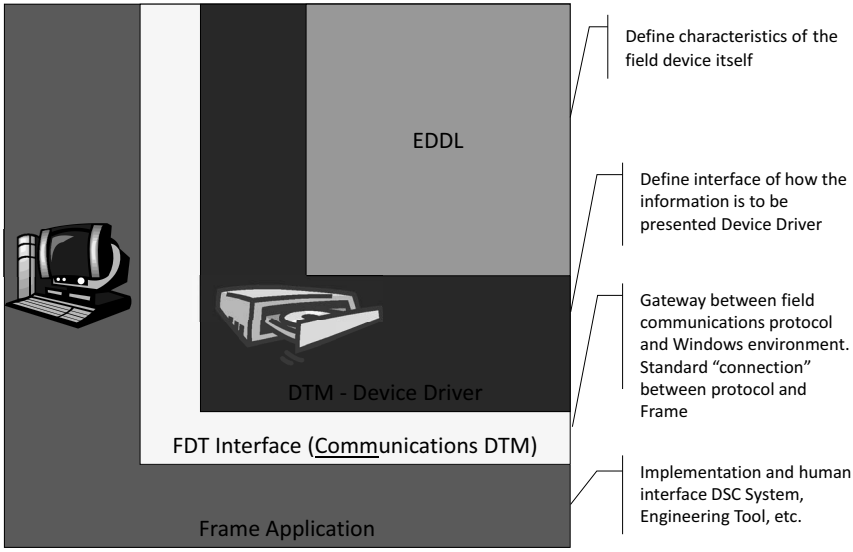
- Universal plant-wide tools for life-cycle management of heterogeneous Fieldbus environments, multi-manufacturer devices, Function Blocks and modular sub-systems for all automation domains (e.g., process automation, factory automation, and similar monitoring and control applications).
- Integrated and consistent life-cycle data exchange within a control system, including its fieldbuses, devices, function blocks, and modular sub-systems.
- Simple and powerful manufacturer-independent integration of different automation devices, Function Blocks, and modular sub-systems into the life-cycle management tools of a control system.

The FDT concept is made of three components; two FDTs and a DTM, as shown in Figure 1-14.

The purpose of the first FDT, the Frame Application, is to provide the following: Common Environment, Network Configuration, Navigation, User Management, Device Management, and Database Storage. The second FDT, the interface, is like a printer interface and provides the gateway or "interpreter" between the Frame Application and its associated Windows™ drivers and the tunnel through the appropriate digital communications protocol, such as Fieldbus, HART, or DeviceNet.

The DTM is like a printer driver, and just like a printer driver, it tells a computer how to print characters on a page. The DTM works by defining all the parameters related to a specific field device and how they should appear on the appropriate HMI and associated software. In the cases of HART, Profibus PA, and FOUNDATION Fieldbus, the DTM, at its core, is based on interpreting and expanding on the device's DD file.

Figure 1-14 – Field device interface communications



All three of the standards used to support FDT developments are now part of the IEC TC65E standards. OPC UA (Universal Architecture) is going forward as IEC 62541, EDDL is part of IEC 61804, and as indicated above, FDT is IEC 62423.

The Frame Application provides the following capabilities to FDT technology:

- A common environment for presentation of the device data.
- Network Configuration to establish the communication links between the HMI and the field device through one or more protocols between the two end points.
- Navigation between the various windows, devices, and protocol communication messages.
- User Management to control who has access to what portions of the network and data.
- Device Management by linking the appropriate DTMs together to enable communication and viewing of the data.

- Database Storage to archive and retrieve information.

The DTM, which is provided by the device manufacturer, is the driver representing the actual device. The DTM, which has a standardized Graphical User Interface to the Frame Application, can be loaded on any Frame Application and includes the complete parameters of the device.

There are two types of DTMs: the COMM DTM represents communication to the devices like PC communication cards, couplers, gateways, and linking devices while the Device DTM represents field devices like valves, sensors, actuators, transmitters, motors, and pumps.

The FDT organization, a not-for-profit association of international companies dedicated to establishing the FDT Technology as an international standard, has adopted a number of key life-cycle policy elements, including a requirement that all DTMs and FDT Frame Applications must be maintained for at least 10 years after initial release. In addition, a maintenance release (update) must not require an update of the operating system or any of the hardware components. FDT components must also be interoperable within a major FDT version while remaining compatible to application (instance) data of a former version of the component.

Individual vendors certify their FDT components according to the certification procedures of the FDT Group before they are released to the market, and all FDT Group members must apply the rules and guidelines specified in the Life Cycle Policy for their FDT-based products.

1.3.3 Field Device Interface (FDI)

The FDI cooperation project was announced at Interkama 2007 as a joint effort between the Fieldbus Foundation and the FDT organization. The intent of this agreement is to develop a single common interface for the complete range of data communications in a modern control system, from configuration through commissioning and maintenance.

FDI, the Field Device Integration project represents the next evolutionary step in Device Description language on which the three predominant field device protocols (HART, FOUNDATION Fieldbus, and Profibus PA) are based.

Consequently, FDI will have a significant impact on the future look and feel of digital field sensors, especially after the formation in 2011 of a separately incorporated entity, the FDI Corporation.

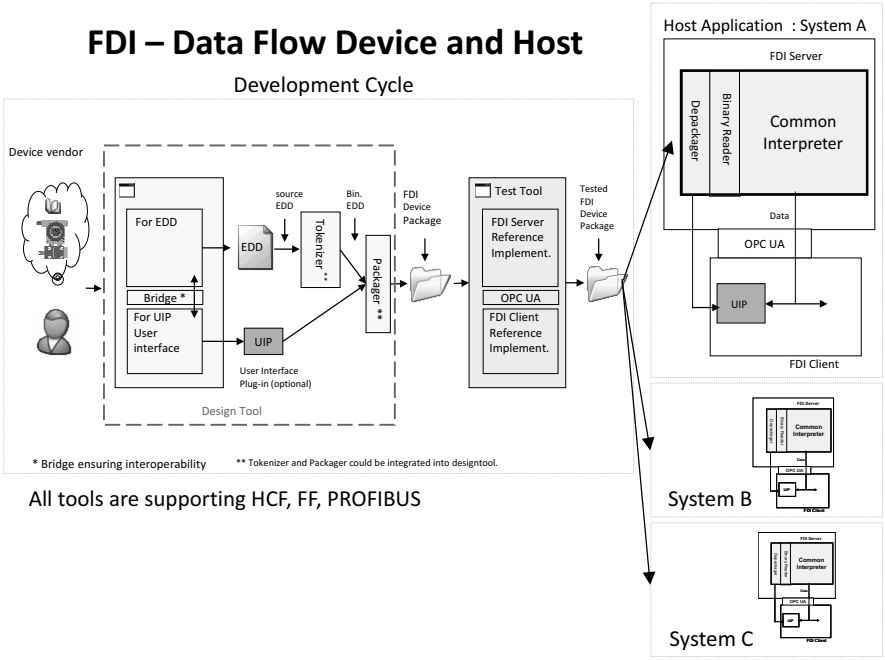
At the NAMUR (a not-for profit German end-user trade association) Annual General Meeting in late 2011, the FDI Corporation demonstrated a multi-vendor Host system prototype, using FDI device packages for FOUNDATION Fieldbus, HART, and PROFIBUS device integration. The purpose of the prototype was to verify the FDI concepts and apply the standard Host components in a system context to demonstrate FDI functionality. Publication of the first draft of the FDI specifications was expected by the end of 2011; completion of conformance test concepts, along with validation and release of the specifications for member review within the foundations, will follow in mid-2012; and the complete FDI standard Host components, such as EDD Engine and User Interface (UI) Engine, by the FDI Corporation should be done by the end of 2012.

So what is FDI? In simple terms, when completed, FDI will be the replacement for all EDDL (IEC 61804-3) based languages—HART, FOUNDATION Fieldbus, and Profibus PA.

While EDDL is a common, text-based description of a device, the text description is normally converted to a "binary DD" through a tokenizer before being shipped with the device. Unfortunately, the format of the binary DD is different for each process Fieldbus, even though they originate from the common EDDL language. The manufacturing company members of the FDI Corporation have made it a high priority to harmonize the binary DD through secondary standards and tools so that the result will be a single binary format file, regardless of the protocol used by the device.

Figure 1-15 illustrates how all these systems will work together. The FDI developer for each of the protocols to be supported will develop a tokenizer to "convert" its original version of the EDD language to the new common interface so that the end result to the FDI server will be a single file format that can be read by the common interpreter.

Figure 1-15 – FDI Flow Device and Host



The EDDL file for each protocol will be processed through a tokenizer much like is done today; this also ensures backwards compatibility. Because each protocol is not exactly the same, but rather closer to 90% the same, it will be necessary to develop an FDI Developer environment for each of the three EDDL-based protocols to assist the protocols in defining how to map the various parameters of each protocol to the appropriate FDI parameters.

The resulting binary file from the tokenizer will then be passed to a "packager," where it will be converted to an FDI file. Note that at their discretion, a device manufacturer will be able to define a User Interface Plug-in that is integrated into the FDI file by the "packager" to create the single common file. FOUNDATION Fieldbus device manufacturers may do this today by creating "Extended" Function Blocks that contain information beyond what has been fully defined by the Fieldbus Foundation.

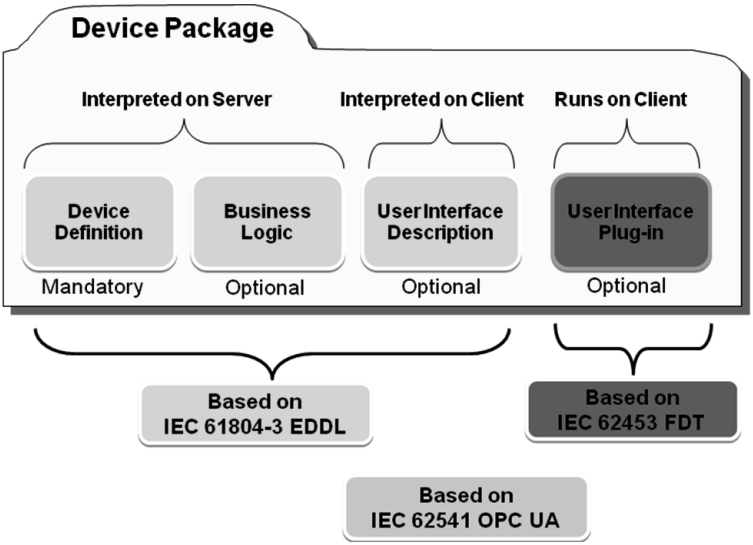
What will be important to end users, of course, is the interoperability of these devices, which will be ensured through the appropriately colored green "test tool" box to verify the "raw" FDI Device Package is compliant

with the standard so that after testing, the "Tested FDI Device Package" will have the necessary "check mark" from the appropriate organization that the devices are not only compliant with FDI but also backwards compatible with existing equipment. This is obviously important for the instances when a new device needs to be added to an existing network, either for expansion or because of failure of an installed device, and everything will have to continue to work together seamlessly as they did prior to the change.

When the new device is connected and communicating on the network, the process needs to be reversed, with the DCS/host converting the FDI information into a format usable by the internal system databases. This is not any different than is done today, where each system needs to interpret the information from the field to the appropriate database register within the Host. (If you have used Modbus, you have had to do this manually when cross-referencing a specific terminal input from an I/O card to the appropriate memory register.)

Since all the components, such as the Device Definition and User Interface, are based on either existing or proposed IEC standards, the IEC documents shown in Figure 1-16 will also need to be updated.

Figure 1-16 – Device Package showing documents to be updated



Note that the User Interface Plug-In, which will be used to provide improved access to the maintenance, diagnostic, and related parameters in the field devices, will use the knowledge gained from the use of FDT technology, and combine that with the open interoperable communications capabilities of OPC UA to provide a platform independent access to the rich data set contained in a modern digital field device.