S T A N D A R D

HART® COMMUNICATION PROTOCOL

Network Management Specification

HCF_SPEC-085, Revision 2.0

Release Date: 18 June 2012

Release Date: 18 June 2012

Document Distribution / Maintenance Control / Document Approval

To obtain information concerning document distribution control, maintenance control and document approval, please contact the HART Communication Foundation at the address shown below.

Copyright © 2007, 2008, 2011, 2012 HART® Communication Foundation

This document contains copyrighted material and may not be reproduced in any fashion without the written permission of the HART Communication Foundation.

Trademark Information

HART® is a registered trademark of the HART Communication Foundation, Austin, Texas, USA. Any use of the term HART hereafter in this document, or in any document referenced by this document, implies the registered trademark. All other trademarks used in this or referenced documents are trademarks of their respective companies. For more information contact the HCF Staff at the address below.

Attention: Foundation Director
HART Communication Foundation
9390 Research Boulevard
Suite I-350
Austin, TX 78759, USA
Voice: (512) 794-0369
FAX: (512) 794-3904

http://www.hartcomm.org

Use of imperatives in HART Specifications

The key words (imperatives) "must", "required", "shall", "should", "recommended", "may", and "optional" when used in this document are to be interpreted as follows:

Must **Must**, **Shall**, or **Required** denotes an absolute mandatory requirement. For example, "All HART Field Devices must implement all Universal Commands"

Should **Should** or **Recommended** indicates a requirement that, given good cause/reason, can be ignored. However, the consequences of ignoring the requirement must be fully understood and well justified before doing so.

May **May** or **Optional** identifies a requirement that is completely optional and can be supported at the discretion of the implementation. May can be used to identify optional Host Application or Master functionality and, when this is the case, does not imply the function is optional in Field Devices.

Intellectual Property Rights

The HCF does not knowingly use or incorporate any information or data into the HART Specifications which the HCF does not own or have lawful rights to use. Should the HCF receive any notification regarding the existence of any conflicting Private IPR, the HCF will review the disclosure and either (a) determine there is no conflict; (b) resolve the conflict with the IPR owner; or (c) modify this specification to remove the conflicting requirement. In no case does the HCF encourage implementors to infringe on any individual's or organization's IPR.

Table of Contents

Table of Figures

List of Tables

Preface

This preface is included for informational purposes only.

This major revision introduces HART-IP, the specification for transport of HART communications over Internet Protocol-based (IP) networks. In addition several sections of this Specification include clarifications and corrections. In particular:

- Sections 1 through 4 are largely unchanged. References to FIPS-197, AES, GUID and Endianness have been added along with a few new definitions and acronyms.
- Section 5 is unchanged.
- Section 6 contains some corrections and clarifications. In addition, an overview of HART-IP was added.
- Sections 7 and 8 include a few clarifications and corrections.
- Section 9 contains many clarifications and correction based on feedback from developers and the HCF Working Groups
- Section 10 (NEW!) specifies HART-IP, a relatively simple specification for the transport of HART over IP-based networks. HART-IP works over UDP or TCP using IPv4 or IPv6. To enable rapid acceptance and product development, HART-IP payloads are based on HART Token-Passing Data-Link Layer PDUs.

  Security best-practices for IP-based products continue to evolve and HART-IP is designed to be security agnostic. In other words, while security is not specified all products should implement best-practice security measures.

  HART-IP was initially developed for I/O Systems (e.g., multiplexers and Gateways). However, HART-IP is also well suited for traditional process instruments and analyzers. Power-Over-Ethernet (POE) combined with HART-IP enables development of a wide range of fast, new HART compatible products.

Introduction

This introduction is included for informational purposes only.

The specification defines Network Layer requirements for HART-enabled products. In particular, the specification segregates the requirements into 4 major areas: Network Layer services, wired networks, wireless mesh networks, and general network management requirements. The majority of the requirements can be found in

- Section 6 provides an overview of the specification. This section provides background and requirements for both wired and wireless networks. An overview of the hierarchal topology characteristic of wired networks is provided. Requirements and considerations for I/O systems performance, burst mode operation, and multi-drop networks are discussed.

  For WirelessHART, general mesh network characteristics and requirements are discussed. An overview of the operation of the mesh is provided. Mesh components, routing and security requirements are introduced.
- Section 7 focuses on Network Layer services. Service Primitives provides a useful "black box" view of the Network Layer and provided important requirements. While a useful abstraction for wired networks, the services are most applicable to WirelessHART networks.
- Section 8 focuses on wired networks and is mostly material transferred from the *Command Summary Specification.* One focus is the format of the Application Layer payload. HART 6 expanded the command number from 8 bits to 16 bits. This section specifies the mapping of 16-bit commands into the existing Token-Passing Data-Link PDU. Requirements supporting multi-drop networks are also specified in this section.
- Section 9 specifies the WirelessHART Network Layer requirement. These requirements include detailed specifications of the NPDU, Network Layer security, requirements for the embedded lightweight Transport Layer, the operation of the Network Layer (with state machines), and specify Network Layer procedures.

  Security is always on and based on secure end-to-end session between devices with AES-128 symmetric keys. All payloads are enciphered such that only the final destination can decipher and utilize the payload.
- Section 10 defines HART-IP, the specifications for communicating HART over Internet Protocol-based networks. HART-IP is a relatively simple mechanism for carrying standard Token-Passing Data-Link Layer PDU over IP networks. In addition, HART-IP includes specifications for opening, keeping alive and closing sessions.
- Section 11 was largely transferred from the *Command Summary Specification* and procedures for establishing communications with HART field devices. While essential for wired networks, these techniques are also useful in wireless ones, too. For example, Find Device is useful when trouble-shooting networks and Command 75 is essential to WirelessHART Adapters.

1 SCOPE

This specification specifies and establishes the Network Layer and Network Management requirements for HART-compliant networks. The Network Layer is the point of convergence for traditional HART Token-Passing Networks, WirelessHART TDMA-based networks and HART over IP (see Figure 1). Above this layer resides the HART Application Layer that defines allowed HART data types, procedures and commands. Below this layer reside the two major HART disciplines: wired Token-Passing networks, and TDMA wireless communication technologies. In other words, this document specifies the rules used by HART products to communicate via the Network Layer over either wired or wireless HART networks.

Figure 1 shows the scope of this specification. This document includes:

- An overview of Network Layer requirements.
- Network Layer service requirements. These are segregated into common communication services and technology specific management services.

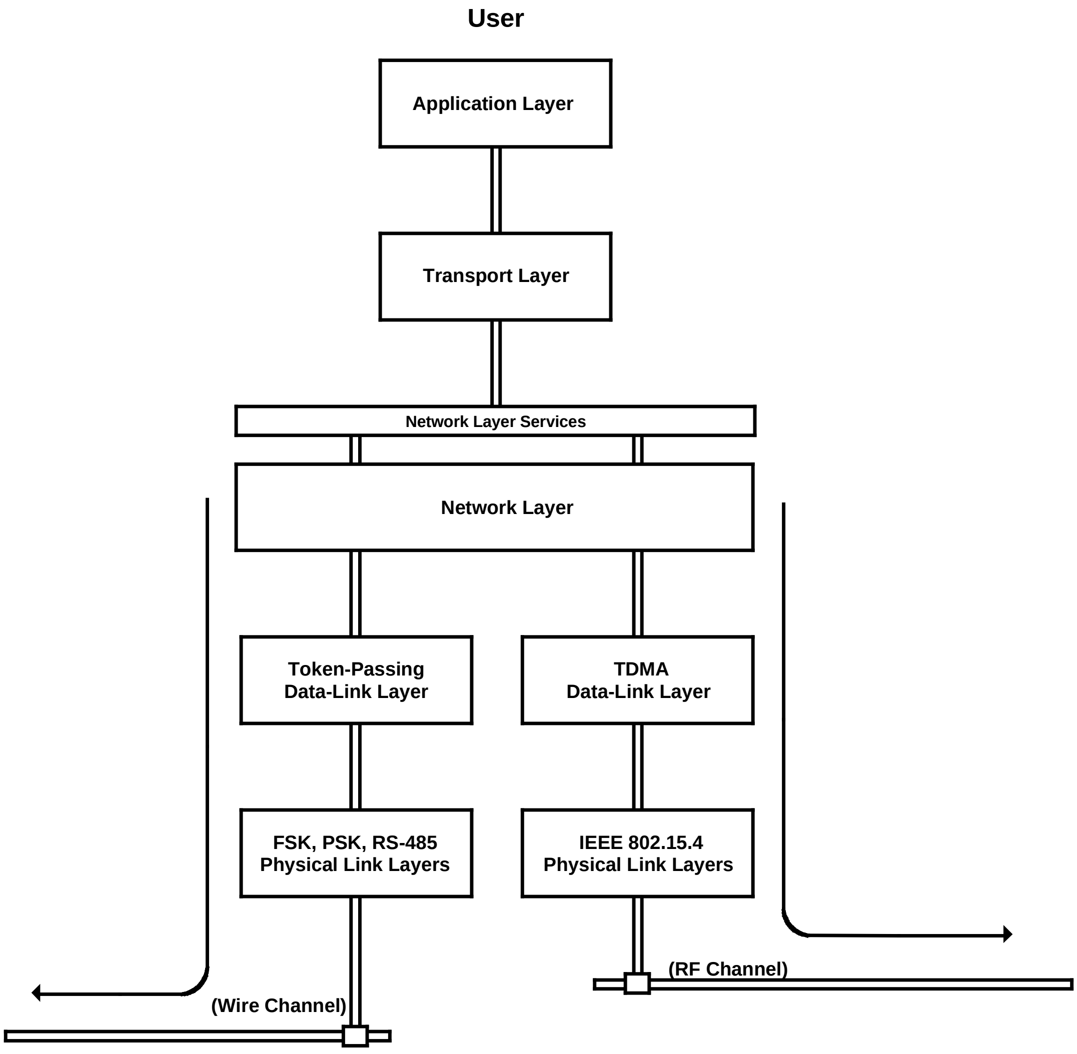


Figure 1. Network Layer Scope

- Wired and wireless Network Layer specifications.
- Common Network Management procedures.

The segregation of requirements into these categories is intended as a frame of reference rather than as a description of an actual implementation. While actual implementations may vary, all wired requirements in this specification are mandatory. While wireless support is optional, if a product supports WirelessHART communications all wireless requirements are mandatory.

Unless specifically noted, HART data is transmitted most significant byte first (i.e., big endian).

2 REFERENCES

These documents published by the HART Communication Foundation are referenced throughout this specification:

HART Field Communications Protocol Specification. HCF_SPEC-12.

TDMA Data-Link Layer Specification. HCF_SPEC-75

Token-Passing Data-Link Layer Specification. HCF_SPEC-81

Command Summary Specification. HCF_SPEC-99

Command Practice Command Specification. HCF_SPEC-151

Block Data Transfer Specification. HCF_SPEC-190

WirelessHART Device Specification. HCF_SPEC-290

2.1 Related HART Documents

References to other standards, clarifying documents and applicable patents are listed in this subsection.

WirelessHART User Guide. HCF_LIT-84

Coexistence Test Plan. HCF_LIT-85

2.2 Related Documents

The following are applicable IEEE documents:

IEEE STD 802.15.4(b). *Wireless Medium Access Control (MAC) and Physical Layer (PHY)*

Diffie, W. and Hellman, M. "Privacy and Authentication: An Introduction to Cryptography". *Proceedings of the IEEE*, Vol. 67 No. 3 (March 1979). pp 397-427

The following provides general guidelines for the specification of communication protocols.

ISO 7498-1 *Information Processing Systems — OSI Reference Model — The Basic Model*

The following reference describes communication specification techniques used in this document including Service Primitives (SPs) and time sequence diagrams:

Halsall, F. *Data Communications, Computer Networks and Open Systems*. Third Edition. Addison Wesley. 1992

The following reference describes the methods for specifying state transition diagrams used in this document.

Hatley, D., and Pirbhai, I. *Strategies for Real-Time System Specification*. Dorset House, 1987.

The following reference provides additional information about and algorithms for AES-128 cipher for security.

National Institute of Standards and Technology, U.S. Department of Commerce, "Specification for the Advanced Encryption Standard", Federal Information Processing Standards Publication 197 (FIPS-197), November 2001.

B. Gladman’s AES related home page http://fp.gladman.plus.com/cryptography_technology/.

The following references provide additional information about Globally Unique Identifier (GUID) and the Big-Endian byte ordering:

> Wikipedia contributors, "Globally unique identifier," Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/w/index.php?title=Globally_unique_identifier&oldid=455054396 (accessed October 24, 2011).
>
> Wikipedia contributors, "Endianness," Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/w/index.php?title=Endianness&oldid=455208015 (accessed October 24, 2011).

3 DEFINITIONS

Some of the following definitions are included in the *HART Field Communications Protocol Specification*. However, these definitions are critical to the understanding of this specification. As a result, they are included and their meaning amplified.

Channel RF frequency band used to transmit a modulated signal carrying packets.

Client The requesting program or user in a client/server relationship. Initiates requests on behalf of a user/program and waits for replies to transfer the requested information back to the user/program.

Coexistence Coexistence is the ability of one system to perform a task in a given shared environment in which other systems have an ability to perform their tasks and may or may not be using the same set of rules (IEEE).

Gateway A Network Device containing at least one host interface (such as serial or Ethernet), acting as ingress or an egress point.

Graph A routing structure that forms a directed end-to-end connection between Network Devices.

Handheld A host application residing on a portable device

Hop A term used to describe the data being passed from one device to another as a means to lengthen the transmit distance. Also used to denote the function of changing channels.

Interoperability Interoperability is the ability for like devices from different manufacturers to work together in a system and be substituted one for another without loss of functionality at the host system level.

IP Address The internet protocol address of the device.

Join Process by which a Network Device is authenticated and allowed to participate in the network. A device is considered Joined when it has the Network Key, a Network Manager Session and a normal (not join) superframe and links.

Latency The time it takes for a packet to cross a network connection, from sender to receiver. Latency specifications shall (unless otherwise noted) represent a 2-sigma value. i.e., the latency shall be achieved 95% of the time.

Link The full communication specification between adjacent nodes in the network. i.e., the communication parameters necessary to move a packet one hop.

Neighbor Adjacent nodes in the network.

Network Manager The application responsible for configuration of the network; scheduling communication between neighbors; management of the routing tables and monitoring and reporting the health of the network.

Network Device A device with a direct Physical Layer connection to the network. Each network device (e.g., field device or gateway) has a HART Unique Address that is used in communication with the device.

Node An addressable logical or physical device attached to the network.

Nonce A number constructed so as to be unique to the current packet to ensure that old communications cannot be reused in replay attacks

Packet A generic reference to the set of data communicated across a network

Peer The correspondent node at the other end of the communication link. The communication link terminates at the same protocol layer in the correspondent node.

Physical Layer Layer 1 in the OSI model. The Physical Layer is responsible for transmission of the raw bit stream and defines the mechanical and electrical connections and signaling parameters for devices.

Port The specific UDP or TCP port used for connection of this service.

Remote IO System A HART Remote IO System typically provides several wired HART input connections to several field devices with a single output to a host system.

Security Manager An application that manages the Network Device's security resources and monitors the status of the network security.

Server Provides services to other programs or users. Never initiates requests but may issue unsolicited responses (burst messages or notifications). The server waits for requests from one or more clients and replies to client with appropriate response.

Session A semi-permanent interactive information exchange.

Slot A fixed time interval that may be used for communication between neighbors.

Socket One endpoint of a two-way communication link between two programs running on the network. The communication link uses either UDP or TCP as the message protocol, a local and remote IP address and a local and remote port

Superframe A collection of slots repeating at a constant rate. Each slot may have a link associated with it.

Time Sequence Diagram A diagram used to illustrate the interrelationship between the Protocol services. The protocol layer of interest and the lower, intervening layers are treated as a "black box". The internal workings of these layers are not shown on this diagram. The time sequence diagram shows the interactions between the service primitives over time. Sometimes referred to as a Message Sequence Diagram.

Timetable The parameters specifying the application domain, routing and scheduling of communications allocated between two peer devices. Except for communications with the Network Manager, all communications are governed by a Timetable.

Time To Live A field in the network header of each packet that specifies how many more hops a packet can travel before being discarded.

Transport A group of methods and protocols responsible for encapsulation of application data suitable for transfer over a physical medium.

Unicast The sending of a packet to a single node in the network.

4 SYMBOLS/ABBREVIATIONS

All Symbols and Abbreviations used in this specification are listed in this section.

The acronym	Its definition
ACK	See Acknowledge
ASN	See **A**bsolute **S**lot **N**umber
CCA	**C**lear **C**hannel **A**ssessment
dBm	dBm is an abbreviation for the power ratio in decibels (dB) of the measured power, referenced to one milliwatt (1 mW). 0 dBm =1 mW; 10 dBm= 10 mW; 20 dBm= 100 mW; 30 dBm= 1 W
DLPDU	**D**ata-**L**ink **P**rotocol **D**ata **U**nit (i.e., a Data-Link Layer packet)
EUI-64	**E**xtended **U**nique **I**dentifier (64 bits long)
FSK	**F**requency **S**hift **K**eyed
IP	Internet Protocol – a data-oriented, unreliable network layer protocol used for communicating data across a packet-switched network. (Network Layer of the OSI model)
LSB	**L**east **S**ignificant **B**yte. The LSB is always the last byte transmitted over a HART data link
MIC	**M**essage **I**ntegrity **C**ode
MSB	**M**ost **S**ignificant **B**yte. The MSB is always the first byte transmitted over a HART data link.
NPDU	**N**etwork **PDU**
OUI	Organizationally Unique Identifier
PDU	Protocol Data Unit. The packet of information being communicated.
PSK	**P**hase **S**hift **K**eyed
RSL	**R**eceived **S**ignal **L**evel. The signal level (in dBm) at a receiver input terminal.
TCP	Transmission Control Protocol – reliable, connection-oriented, in-order delivery of a stream of bytes. (Transport Layer of the OSI model)
TDMA	**T**ime **D**ivision **M**ultiple **A**ccess
TPDU	**T**ransport **PDU**
TTL	**T**ime **T**o **L**ive
UDP	User Datagram Protocol – does not guarantee reliability or packet ordering and is connecionless. (Transport Layer of the OSI model)

5 DATA FORMAT

In HART Protocol command specifications, service descriptions and data table requirements, the following key words are used to refer to the data formats. For more information about these formats, refer to the *Command Summary Specification*.

Bits Each individual bit in the byte has a specific meaning. Only values specified by the command may be used. Bit 0 is the least significant bit.

Date The Date consists of three 8-bit binary unsigned integers representing, respectively, the day, month, and year minus 1900. Date is transmitted day first followed by the month and year bytes.

Enum An integer enumeration with each numeric value having a specific meaning. Only values specified in the Common Tables Specification may be used.

Float An IEEE 754 single precision floating point number. The exponent is transmitted first followed by the most significant mantissa byte.

Latin-1 A string using the 8-bit ISO Latin-1 character set. Latin-1 strings are padded out with zeroes (0x00).

Packed A string consisting of 6-bit alpha-numeric characters that are a subset of the ASCII character set. This allows four characters to be packed into three bytes. Packed ASCII strings are padded out with space (0x20) characters.

Signed-nn An signed integer where nn indicates the number of bits in this integer. Multi-byte integers are transmitted MSB – LSB.

Time The Time consists of a unsigned 32-bit binary integer with the least significant bit representing 1/32 of a millisecond (i.e., 0.03125 milliseconds).

Unsigned-nn An unsigned integer where nn indicates the number of bits in this integer. Multi-byte integers are transmitted MSB – LSB.

6 OVERVIEW

HART enables communication with smart process instrumentation and controls, and supports both wired and wireless networking technologies. Wired communication supports both point-to-point and multidrop topologies. Wireless communication utilizes mesh technology that supports a wide variety of topologies

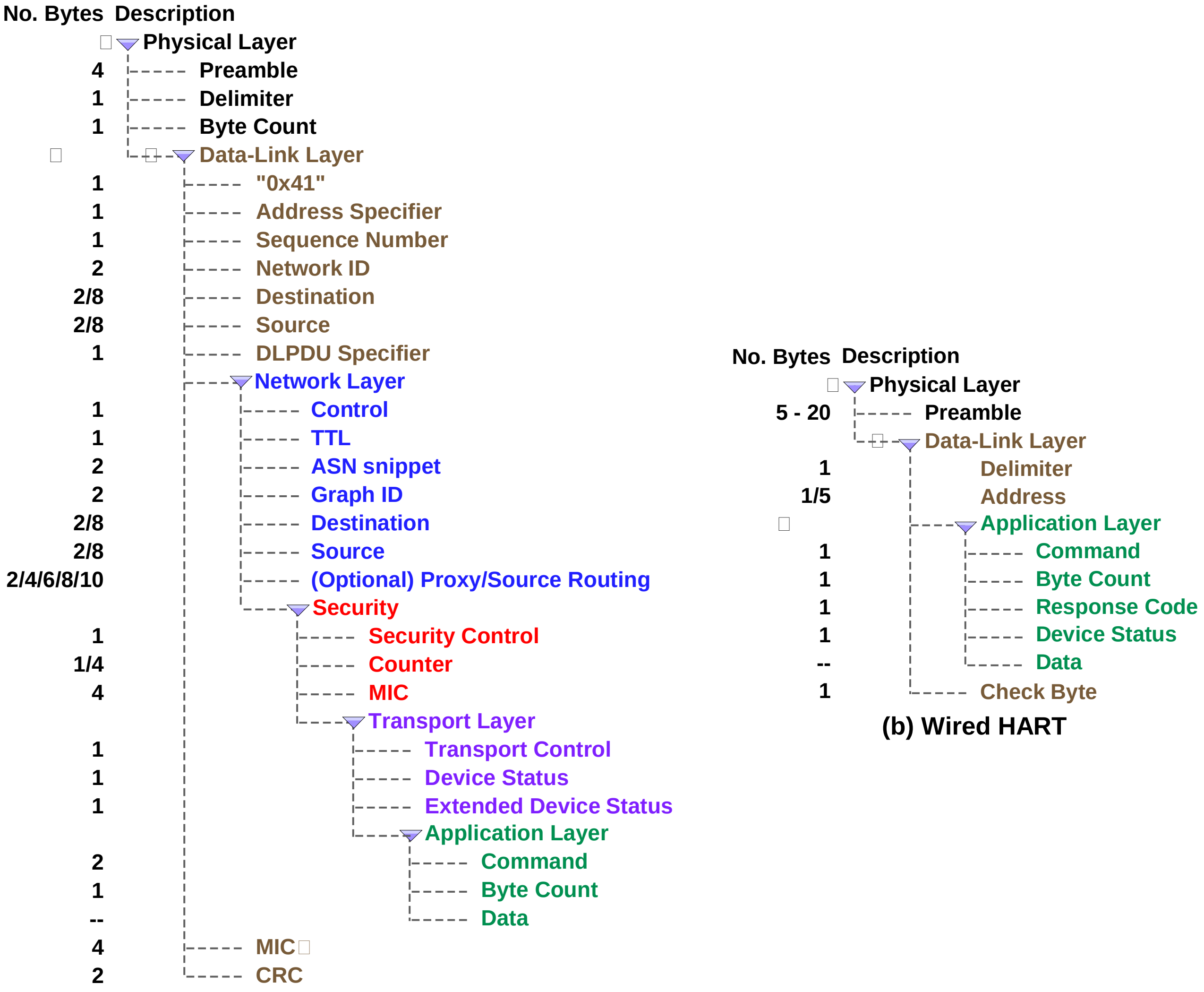
HART supports a variety of network communication traffic including

- **Request/Response.** This is directed communication between a host application and a single, specific device. The address of source and destination device is specific and unambiguous. All HART commands specify the request and response data to be communicated.
- **Publishing of Process Data.** In wired HART, this is accomplished by placing the token-passing Physical Layer into burst mode. The data is published using the response portion of the HART command. This is, in effect the communication of the response.
- **Broadcast messages** use standard HART request/response communications however; the destination address uses the broadcast address. For broadcast messages, the destination node is specified using information other than the address. For example, Command 21 includes the Long Tag of the destination device. In this example, the device with that Long Tag answers the request. In the case of some commands, (e.g., Write Network Key) multiple destinations are possible.
- **Block Data Transfers** use Commands 111 and 112 to establish a pipe between two nodes that allow data to be streamed between them. The data transfer can be quite large and is automatically segmented into multiple packets for transfer across the network. The data stream is re-assembled at the destination. The segmentation and reassembly is transparent to the Application Layer (see the *Block Data Transfer Specification*).

In both wired and wireless networks, HART focuses solely on field device communication. However, the requirements between the two Physical and Data-Link Layers are quite different:

- Wired networks need only a small amount of Network Layer support (see Subsection 6.1). This support includes managing the connected field device communication parameters and establishing communication field devices. Consequently, the PDU used for wired communications does not require Network Layer fields (see Figure 2).
- While WirelessHART is still focused on field device communications, reliable, secure mesh communications results in many more requirements than wired token-passing communications (see Subsection 6.2). Mesh communications is key to reliable communications and it requires a complete network layer and robust security as well.

  Since WirelessHART networks may be several hops deep, a lightweight Transport Layer is also included. Unlike wired HART networks, the link-level acknowledge is not sufficient to ensure a packet is successfully transported from the source to destination. This Transport Layer is in addition to the requirement found in the *Block Data Transfer Specification*.



No. Bytes	Description
	Physical Layer
4	Preamble
1	Delimiter
1	Byte Count
	Data-Link Layer
1	"0x41"
1	Address Specifier
1	Sequence Number
2	Network ID
2/8	Destination
2/8	Source
1	DLPDU Specifier
	Network Layer
1	Control
1	TTL
2	ASN snippet
2	Graph ID
2/8	Destination
2/8	Source
2/4/6/8/10	(Optional) Proxy/Source Routing
	Security
1	Security Control
1/4	Counter
4	MIC
	Transport Layer
1	Transport Control
1	Device Status
1	Extended Device Status
	Application Layer
2	Command
1	Byte Count
--	Data
4	MIC
2	CRC

(a) WirelessHART

No. Bytes	Description
	Physical Layer
5 - 20	Preamble
	Data-Link Layer
1	Delimiter
1/5	Address
	Application Layer
1	Command
1	Byte Count
1	Response Code
1	Device Status
--	Data
1	Check Byte

(b) Wired HART

Figure 2. Summary of PDU Formats

6.1 Wired HART

Wired HART primarily serves traditional process automation systems utilizing 4-20mA based networks of smart field devices. Since wired HART field devices are fundamentally 4-20mA devices, backward compatibility with existing plant systems and personnel is maintained. Consequently, plant personnel can utilize HART compatible field devices with little knowledge of the protocol and gradually adopt HART protocol features at their own pace.

In these systems, HART focuses on the last leg of communications to the smart field device (see Figure 3). HART networks normally utilize a point-to-point hub and spoke topology similar to that seen in 10base-T Ethernet networks. Remote I/O is often used to place the I/O system (the hub) near the process thus reducing costs and simplifying troubleshooting. In addition to point-to-point networks, HART can be used in multi-drop applications.

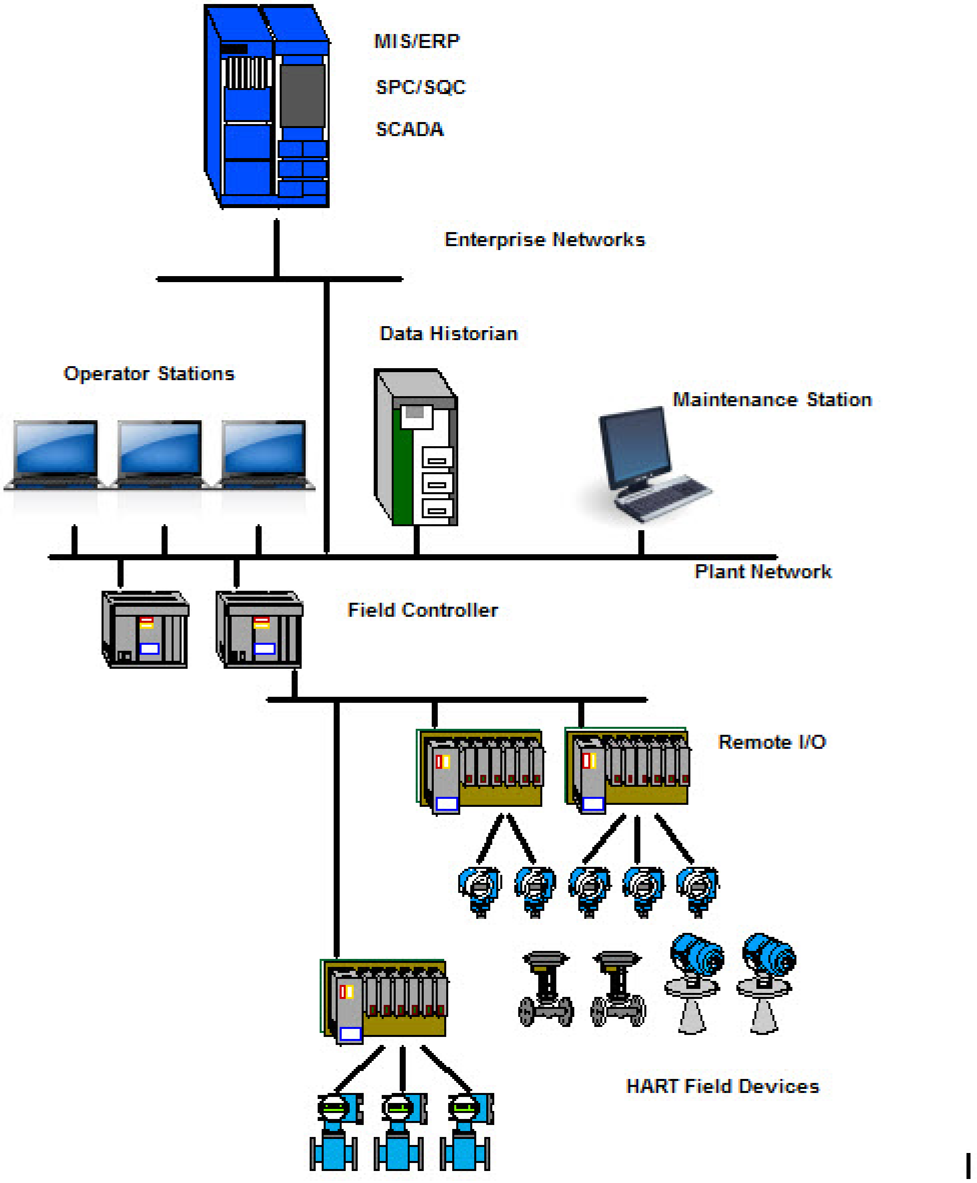
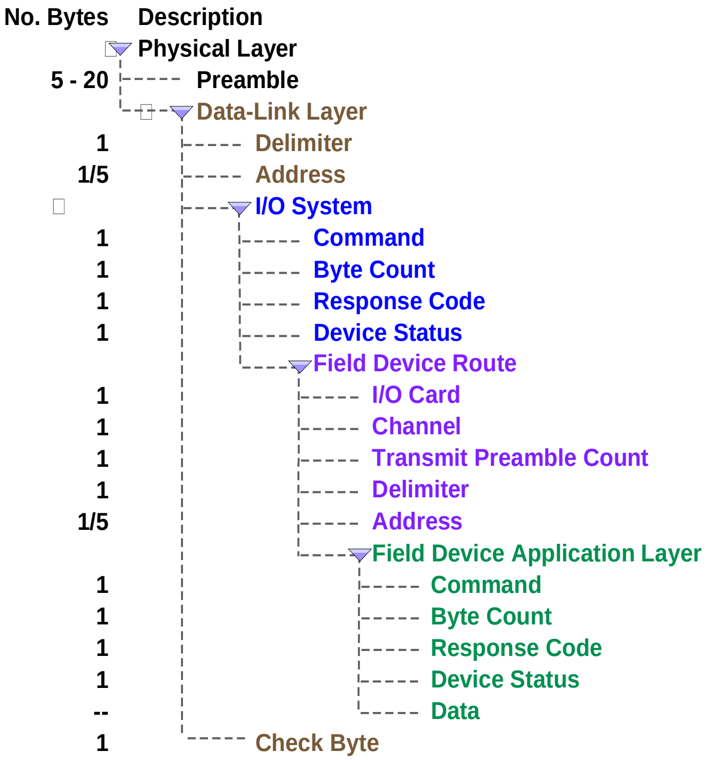


Figure 3. Process Automation System with Wired HART

While HART has traditionally not specified a Network Layer, HART has long been used to route packets to their final destination. Figure 4 shows a typical network routing technique used to add Network Layer routing information to HART messages. In this example, a HART message is wrapped inside another HART message. In this case, Command 77 contains the routing information (I/O card, channel) that tells the I/O system how to route the embedded HART message to the final field device. Command 77's address information routes the message from the controller to the Remote I/O over, for example, a network of Remote I/O connected via HART over RS-485.

Similar techniques have been used by HART-based multiplexers since about 1991.



No. Bytes	Description
	Physical Layer
5 - 20	Preamble
	Data-Link Layer
1	Delimiter
1/5	Address
	I/O System
1	Command
1	Byte Count
1	Response Code
1	Device Status
	Field Device Route
1	I/O Card
1	Channel
1	Transmit Preamble Count
1	Delimiter
1/5	Address
	Field Device Application Layer
1	Command
1	Byte Count
1	Response Code
1	Device Status
--	Data
1	Check Byte

Figure 4. Routing HART Field Device PDU through I/O Systems using Command 77

6.1.1 Multi-Drop Connections

All HART-compatible host and slave devices must support multi-drop. In multi-drop, several HART field devices are connected via the same pair of wires. Latency is increased because, several commands may be enqueued and waiting for transmission.

In Multi-Drop networks, throughput remains the same as with networks containing a single field device (approximately 1.4 FSK, or 9.9 PSK Command 9 transactions per second). However, latency increases and is proportional to the number of device on the loop. Even with the degraded latency, multi-drop is suitable for many applications (e.g., temperature monitoring or tank inventory management).

However, whether multi-dropped or point-to-point, I/O system design can impact wired HART performance.

6.1.2 Multi-Channel I/O Systems

For optimum performance, I/O systems should include one (hardware or software) modem for each channel. With such an I/O system, utilization of all HART capabilities is practical including continuous status and diagnostics, remote configuration of instruments, acquisition of secondary process variables from multi-variable field device and the deployment of multi-drop networks.

Lower performance I/O systems often multiplex HART communications by using one HART modem chip to support several I/O channels. With multiplexed I/O, all transactions must be serialized and the host must resynchronize its communications each time the HART channel is changed. Table 1 summarizes system performance based versus I/O architecture. Multiplexed I/O is clearly the worst performer and is even worse then multi-dropped HART. In fact, real-world performance may be even worse (e.g., due to delays introduced by other system overhead).

Table 1. Estimates of I/O System Performance for FSK (PSK) Networks

	Latency in Seconds			**Throughput** in Transactions per Second		
Number Channels	**Point-Point**	**Multi-Drop**	**Multiplexed**	**Point-Point**	**Multi-Drop**	**Multiplexed**
1	0.73 (0.101) ↓	0.73 (0.10)	0.73 (0.10)	1.38 (9.98)	1.38 (9.87) ↓	0.97 (6.98) ↓
4		2.90 (0.41)	4.12 (0.57)	5.51 (39.5)		
8		5.81 (0.81)	8.25 (1.15)	11.0 (79.0)		
16		11.6 (1.62)	16.5 (2.29)	22.0 (158.0)		
32		23.2 (3.24)	33.0 (4.59)	44.1 (316.0)		

Note: Based on Command 9 communicating four Device Variables

6.1.3 Burst-Mode

Burst-mode allows a field device to be configured to continuously transmit digital process values. All HART hosts must support burst-mode, however, utilization of burst-mode will depend on the application. If common multi-variable field devices are employed, burst-mode maximizes the update rate of secondary process variables. With this revision of the Protocol, Burst Mode receives a major enhancement (see the *Common Practice Command Specification*). Multiple Burst message can share the Burst Mode channel and Command 48 can be transmitted based on change in device status. With these enhancements, all hosts should utilize burst messages for cyclical acquisition of process values.

6.2 WirelessHART

WirelessHART is a sensor mesh communication system (see Figure 5) that simplifies network and device installation and allows the end user to tailor the installation and its topology to satisfy specific application requirements. The basic elements of a WirelessHART network include:

- Field Devices that are connected to the Process or Plant Equipment. All network devices, including field devices, must be able to source and sink packets and be capable of routing packets on behalf of other devices in the network.
- A Gateway that enables communication between Host Applications and WirelessHART field devices in the WirelessHART network. Every WirelessHART Network includes a WirelessHART Gateway. Gateways, in turn, may include one or more Access Points.
- A Network Manager that is responsible for configuration of the network, scheduling communication between WirelessHART devices (i.e., configuring superframes), management of the routing tables and monitoring and reporting the health of the WirelessHART network. While redundant network managers are supported, there must be only one active network manager per WirelessHART network.

6.2.1 Mesh Networks

Figure 5 shows a basic WirelessHART network with the field devices deployed in a mesh topology. In this example network, there is 1 gateway, 1 network manager and 13 field devices (labeled a-n). All communication occurs, for example, by moving packets from the gateway, through the intermediate devices, to the packet's destination. Each movement of a packet from one device to another along the route to the packet's final destination is called a hop. Requirements specifying the communication of packets between adjacent network devices can be found in the *TDMA Data-Link Layer Specification*.

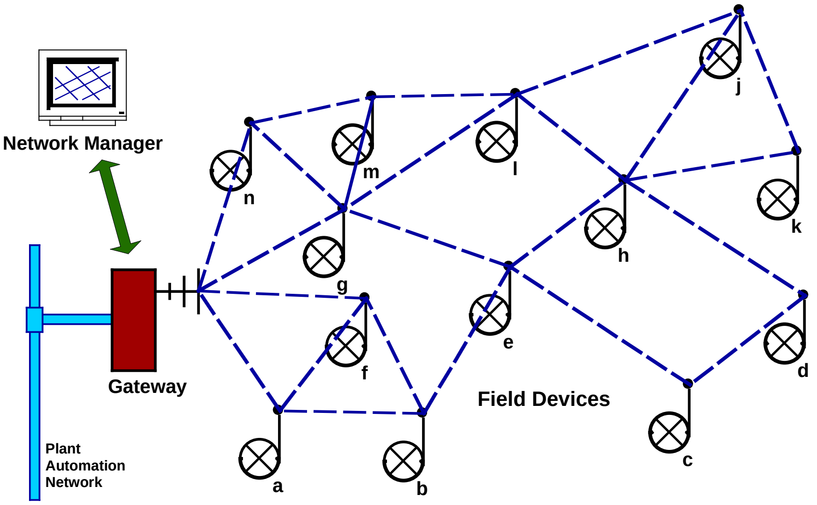


Figure 5. WirelessHART Network

All devices must be able to source and sink packets and be capable of routing packets on behalf of other devices in the network. The routing of packets from their initial source to their final destination may take several hops. The actual routing of packets is the responsibility of the Network Layer.

Within this network, nodes a, f, g, and n are one hop from the gateway. Since these devices can pass packets directly to the gateway, communication with these devices has the lowest latency. However, since WirelessHART uses mesh technology, redundant links are included to improve system reliability by allowing packets to be routed around (for example) interference. In Figure 5, node a can communicate directly to the gateway but can also communicate via node f to the gateway if the direct route becomes blocked.

Nodes c, d, j and k are several hops away. All intermediate devices (e.g., field devices g and e) must be capable of receiving and forwarding packets to and from these devices. Since the packets must make several hops to and from nodes c, d, j and k the communication latency to these devices is longer than the devices that are one hop away. However, WirelessHART mesh technology allows the physical size of the network to become larger and allows communication around obstructions. Furthermore the number of the devices that can be supported in a single network can be larger than that supported by star networks.

By supporting mesh communication technology, WirelessHART networks can be installed in a wide range of topologies. WirelessHART compatible devices can be deployed in a star topology (i.e., all devices are one hop to the gateway) to support a high performance application, a multi-hop overly-connected mesh topology for a less demanding (e.g., monitoring) application, or any topology in between. In fact, WirelessHART technology is flexible enough that a variety of applications (both high and low performance) can operate in the same network.

The WirelessHART Field Network maintains very high reliability using several mechanisms including multiple paths to network devices, multiple frequencies, and retries. If improved reliability is required, more paths can be inserted by adding additional access points and field devices. Additional network access points can be used to increase throughput and reduce latency.

Precise time synchronization is critical to the operation of networks based on time division multiplexing. Since all communication happens in slots, the Network Devices must have the same notion of when each slot begins and ends, with minimal variation. In a typical WirelessHART Network, time propagates outwards from the Gateway.

6.2.2 WirelessHART Components

This section discusses the types of devices and other elements that are associated with a WirelessHART installation. The WirelessHART Network supports a wide variety of devices from many manufactures. Figure 6 illustrates the basic elements of a WirelessHART installation along with types of equipment and functional components that may be present. In some cases, like a Field Device, the product may be a physical device. In other cases (e.g., Network Manager), a logical or abstract element is described. The items shown in Figure 6 include:

- Field devices are connected to and characterize or control the Process. They are both a producer and consumer of packets and must be capable of routing packets on behalf of other Network Devices.
- Adapters connect to existing HART compatible field devices and enables communication to all of them via a WirelessHART Network (see the *WirelessHART Devices Specification*).
- A Gateway enables communications between Host Applications and devices that are members of the WirelessHART Network. The Gateway has one or more Access Points interconnecting the Plant Automation Network and the WirelessHART Network.
- Handhelds and other Maintenance Tools are portable applications used to configure, maintain or control plant assets. Only portable equipment directly connecting to the WirelessHART Network fall into this category.
- The Network Manager is responsible for configuration of the network; scheduling communication between Network Devices; management of the routing tables and monitoring and reporting the health of the WirelessHART Network.
- The Plant Automation Network connects client applications to the gateway and, consequently, the WirelessHART Network's members.
- Host Applications are the tools used by plant staff to monitor, manage and control plant operations and plant equipment. Host Applications include all tools communicating to Network Devices via the Gateway (e.g., plant automation controllers and maintenance tools).

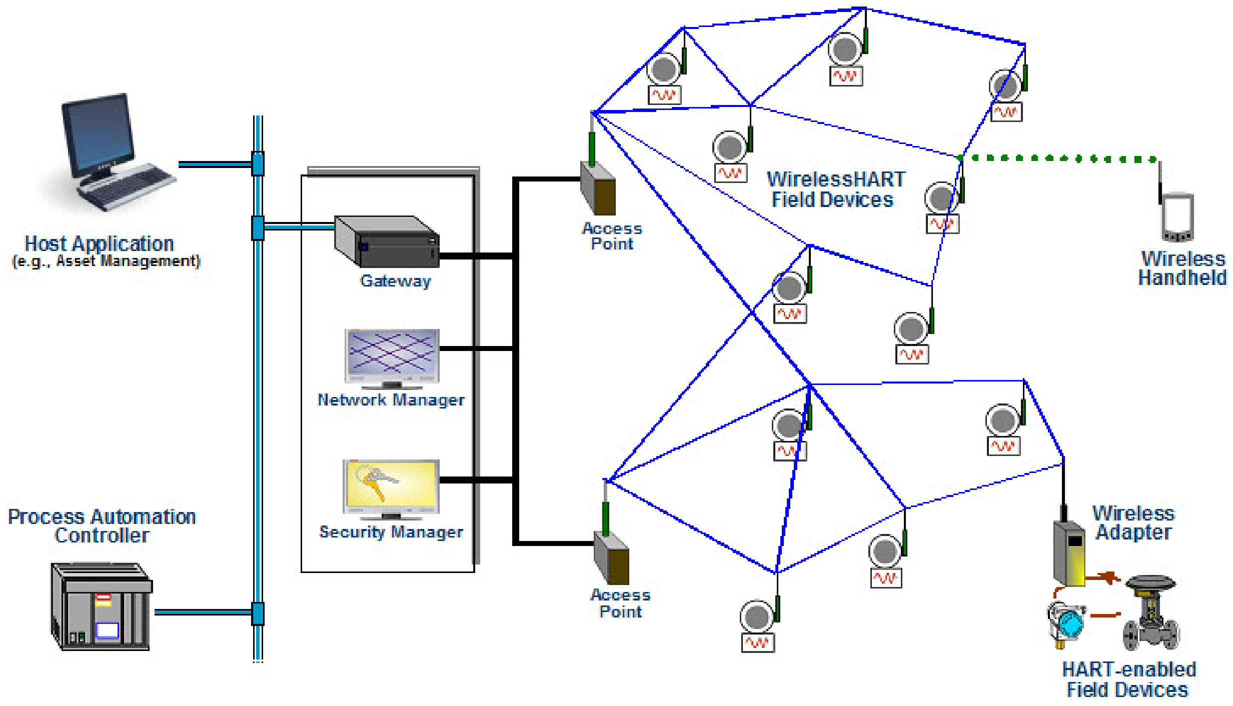


Figure 6. WirelessHART Elements of a WirelessHART Installation

6.2.2.1 Network Devices

Network Devices have a direct Physical Layer connection to the network. Each network device has a HART Unique Address that is used in communications with the device. Each Network Device also has properties holding information on update rates, sessions, and device resources covering items such as the size of the Superframe, etc. Each Network Device contains a list of Neighbor Devices that it has identified during its listening operations (neighbors can be identified devices during any receive time slot).

Typical Network Devices include Field Devices, Adapters and Gateways. At least one of the Physical Layers supported by the Token-Passing Data-Link Layer must be included in the Network Device.

> Note: Dedicated routers are also Network Devices. However, since all Network Devices must be capable of routing they offer only a subset of the capabilities of, for example, a WirelessHART field device.

All Network Devices must be capable of routing packets on behalf of other Network Devices. The Network Device uses internal routing tables to decide which Network Device to forward the packet to. If Graph Routing is used then the Graph ID is used to select the neighbor to forward to. If Source Routing is used then the next entry in the Source Route, or the final destination address itself, is used to determine the next Network Device to forward the packet to. Network Devices are described in detail in WirelessHART Device Specification (HCF_SPEC-290).

6.2.2.2 WirelessHART Field Device

The most common type of Network Device is a Field Device. The WirelessHART field device is a Network Device that integrates wireless communications into the traditional HART field device. The field device may be line, loop, or battery powered or they may be powered in some other fashion. They may support traditional current loop signaling or not.

6.2.2.3 WirelessHART Adapter

The Adapter is a Network Device that connects to an existing HART-enabled field device or a wired HART loop that may be multi-dropped, enabling the field devices to communicate via a WirelessHART Network. Thus, the adaptor has to support traditional HART communication and WirelessHART and translate between the two. The adapter supports the publishing of process data and status and allows complete access to the configuration of the connected HART-enabled devices.

6.2.2.4 WirelessHART Gateway

A Gateway is a Network Device with one or more Access Points. The Gateway connects the WirelessHART Network to a plant automation network allowing data to flow between the two networks. Network Device data collected by the Gateway is communicated to the plant automation network using its protocols and interfaces. This communication includes, for example:

- Routine communication of process-related data and events. This communication is cyclical and occurs on a predictable periodic interval.
- Status and other event generated data communication occur as the result of a field device maintenance or failure or as the result of abnormal process conditions. This communication is sporadic but must occur in a timely fashion.
- Configuration and maintenance related communication generally occurs in bursts. Maintenance, configuration and diagnostic activities result in many packets over a short time interval being communicated to a specific field device. However, configuration and maintenance on a given device is infrequent (e.g., once in every three months or longer).

In other words, the Gateway provides host applications access to the Network Devices. A Gateway can be used to convert from one protocol to another, as go-between two or more networks that use the same protocol, or to convert commands and data from one format to another.

While a network has only one Gateway, in many situations the Gateway will be virtualized and support more than one Access Point. These multiple Access Points each have their own physical address and are used to improve network throughput and reliability. In other words, more packets per second through the network are possible and the network is resistant to the failure of a single access point.

To simplify support for redundant Access Points, every Gateway has a fixed; well know address (Unique ID = 0xF981 0x000002; Nickname = 0xF981). This also allows all devices joining the network to easily detect when they are promoted from Quarantined to Operational. For all application communications (i.e., non-Network Manager communications), the Gateway is normally common to all graphs allowing packets to be automatically routed via the most convenient Access Point. Since all routes must include alternate Access Points, if an Access Point fails then network traffic will be impaired but communication will still be possible. When a Gateway uses multiple Access Points the Network Manager must provide redundant routing for each Network Device using at least of two of the Gateway's Access Points.

See the *WirelessHART Devices Specification* for more information.

The Gateway is the clock source for the network and one or more of its Access Points may propagate the clock to the network. If there are several Access Points providing the clock, it is the Gateway's responsibility to ensure they stay synchronized with each other. This leads to one of three possible configurations: a single Access Point (see Figure 7); multiple Access Points providing the network clock (see Figure 8); or multiple Access Points with at least one not providing the network clock (see Figure 9). In any case, there must be at least one Access Point providing the clock to the network. When an Access Point does not source the network clock it must synchronize to the Network Devices acting as its clock parents.

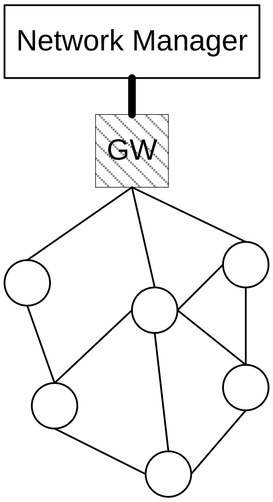


Figure 7. Single Access Point with Clock

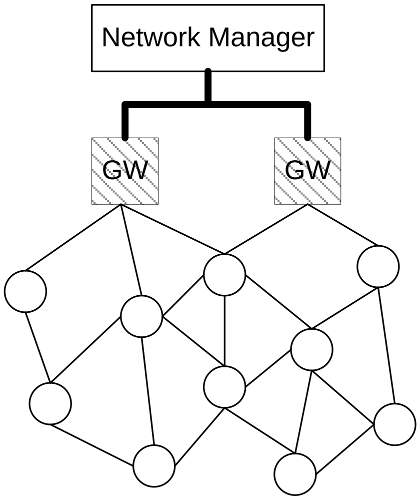


Figure 8. Multiple Access Points with Clocks

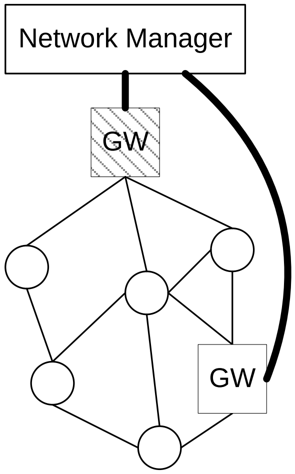


Figure 9. Access Point Not Providing Clock

6.2.2.5 Network Manager

The Network Manager is an application that manages the WirelessHART Network and its Network Devices. The Network Manager forms the WirelessHART Network, joins and configures new Network Devices, and monitors the network.

The Network Manager contains a complete list of Network Devices and ensures each has a network-unique short 16-bit Nickname. The Network Device list maintained by the Network Manager is used for network functions such as routing and scheduling. The Network Manager is responsible for configuration of the network; scheduling communication between WirelessHART Devices; management of the routing tables and monitoring and reporting the health of the WirelessHART Network.

As part of its system functions, the Network Manager collects performance and diagnostic information. This information is accessible during run-time making it possible to view and analyze the behavior of the overall network. If problems are detected the reconfiguration of the network is performed while the network is operating. This network grooming is performed continuously as the overall network operation and performance varies due to changes in network load and environmental conditions.

While redundant Network Managers is possible, there must be one and only one active Network Manager per WirelessHART Network. The Network Manager has a fixed well known address (Unique ID = 0xF980 0x000001; Nickname = 0xF980). See the *WirelessHART Devices Specification* for more information.

Since the Network Manager is an application rather than a Network Device, the location of the Network Manager application is not restricted by this specification. However, the Network Manager must have a secure communication channel to the Gateway and the Security Manager.

6.2.2.6 Security Manager

Join, Network and Session Keys must be provided to the Network Manager and Join keys must be provided to Network Devices. These keys are used for device authentication and encryption of data in the network. The Security Manager is responsible for the generation, storage, and management of keys. There is one Security Manager associated with each WirelessHART Network. The Security Manager may administer multiple WirelessHART Networks.

While operation of and requirements for the Security Manager are outside the scope of this specification, these applications are often used to manage these keys. The Security Manager may be a centralized function in some plant automation networks, servicing more than just one WirelessHART Network and in some cases other networks and applications.

6.2.2.7 WirelessHART Handheld

Portable computing is enabling the creation of the Wireless Worker. This worker represents the next generation instrument technician or plant operator. This nomadic worker is enabled using wireless technology (e.g., WiFi, WirelessHART, etc) to electronically access process screens, plans, schematics, and other plant documentation while managing, maintaining, commissioning WirelessHART Devices.

The Handheld is a portable WirelessHART-enabled computer containing a Host Application. Handhelds are used to configure devices, run diagnostics, perform calibrations, and manage network information inside each device. Handhelds are not required to support routing. When used in a maintenance lab Handheld Devices can connect directly to WirelessHART Field devices through their FSK modem.

> Note: It is possible for Handhelds to access the Gateway directly via a Wi-Fi infrastructure. However, in this scenario the handheld is just another Host Application. In other words, in this specification, Handhelds are defined as connecting directly to the WirelessHART Network.

When operating with a formed WirelessHART Network, this Handheld joins to the target Network Device (i.e. the target device must be within one-hop). When operating with a target Network Device that is not connected to a WirelessHART Network, the Handheld must operate as the combination of a Gateway and Network Manager by forming its own WirelessHART Network with the target Network Device.

6.2.3 Routing

WirelessHART supports Graph and Source routing.

- **Graph**. The network topology can be represented as all of the devices and the directed links between them. A Graph Route is a subset of the directed links and devices that provides redundant communication routes between a source and a destination device. The route actually taken is based on current network conditions when the packet is conveyed from the source to the destination.
- **Source**. A Source Route is a single directed route (devices and links) between a source and a destination device. The source route is statically specified in the packet itself. Current network conditions could break the packet's specified route causing packet loss.

Only the ID of the Graph Route to be used is in the packet and, consequently devices in a Graph Routes must be configured prior to its use. Each intermediate Network Device is configured with a fragment of the overall Graph route. The device must contain a list of all the links that can be used to forward a packet along the Graph. Graph Routes are redundant, highly reliable, and, should be used for normal, routine communications (alarms, request/response, publishing, etc.) both upstream and downstream.

Source Routing contains the entire route specification in the packet and, consequently, intermediate devices require no knowledge of the Source Route in advance. As the packet is routed, each intermediate device propagates the packet to the next device in the packet's Source Route List. Source Routes are not redundant and may fail at anytime. Consequently, Source Routes should only be used for testing routes, troubleshooting network paths or for ad-hoc communications (e.g., routing Join Responses).

All devices must support both Source and Graph routing.

6.2.3.1 Graph Routing

A Graph Route is a directed list of paths that connect network endpoints. The specific paths associated with each graph must be explicitly configured by the Network Manager in the individual Network Devices. A single network instance may have multiple graphs, some of which may overlap. Each Network Device may have multiple graphs going through it, even to the same neighbors. Graphs are unidirectional.

Every graph in a network is associated with a unique Graph Id. When using Graph Routing, the device places the Graph ID value in network header (see Subsection 9.1.1). Devices receiving the packet then forward it along the set of paths belonging to the Graph to the destination.

Each Graph ID in the device should have multiple associated neighbors (see the *TDMA Data-Link Layer Specification*). In a properly configured network, all devices will have at least two devices in the Graph through which they may send packets (ensuring redundancy and enhancing reliability). A device routing a packet must lookup the Graph Id and then send the packet to any of the neighbors listed. Graph Routing is illustrated below in Figure 10.

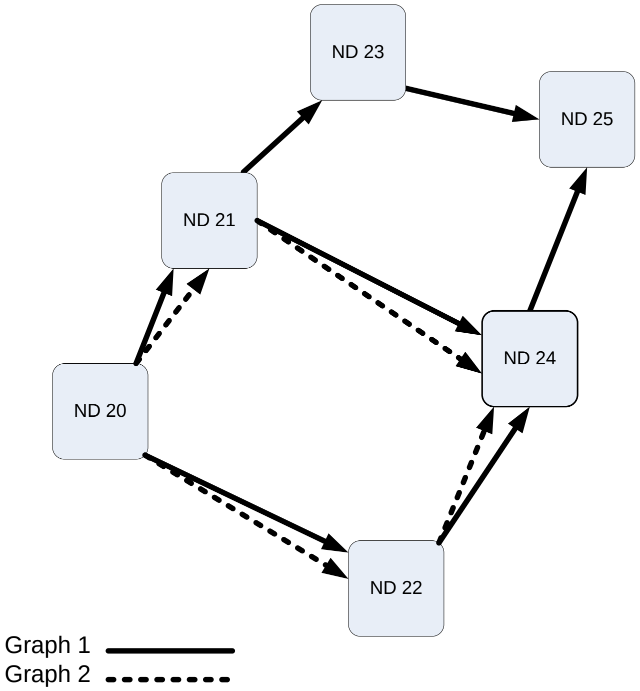


Figure 10. Graph Routing

In Figure 10, ND 20 communicates with ND 25 using Graph 1. To send a packet on that graph, ND 20 may forward it to ND 21 or ND 22. From those devices, the packet may take several alternate routes, but either way, following Graph 1 it will end up at ND 25. Similarly, to communicate with ND 24, ND 20 sends packets on Graph 2 (i.e., through ND 21 or ND 22).

All devices must support Superframe Routing. Superframe routes must use Graph ID 0x0000-0x00FF[1]. When Superframe Routing is performed, the Superframe ID is placed in the NPDU. Devices receiving a Graph ID less than 0x0100 looks for the Superframe[2] with the same value. If successful, the device may forward the packet to any Neighbor with a normal, unicast transmit link in that Superframe[3].

6.2.3.2 Source Routing

Source Routing specifies, in optional NPDU fields (see Subsection 9.1.2), the specific device-by-device route (i.e. a list of addresses) a packet must take when traveling from the source device to the destination device. As the packet is propagated, each intermediate device looks at the source route address list and forwards the packet to the next device in the list. Since only the Network Manager knows the complete topology of the network, only Network Devices configured by the Network Manager with a Source-Route can initiate a source-routed transaction. Source Routing is shown below in Figure 11.

Source Routing specifies a single path and, if one of the intermediate links fails, the packet is lost. Consequently, Source Routing is much less reliable than Graph Routing. When a source route fails the device at the point of the failure must notify the Network Manager. It is the responsibility of the Network Manager to take corrective action.

[1] Graph Routing is "Best Practice". Graph Routes use Graph ID 0x0100-0xFFFF.

[2] The Superframe used for Superframe Routing may not be active.

[3] The transmit links in the Superframe are a substitute for the Graph Edges written using Command 969. Any link in any active Superframe may be used to forward the packet.

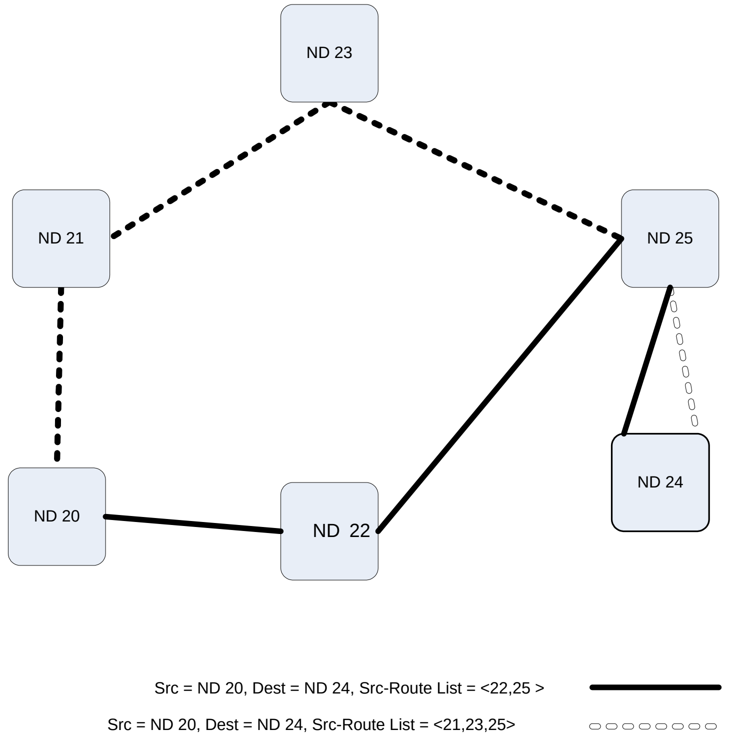

ND 23
ND 21
ND 25
ND 24
ND 20
ND 22
Src = ND 20, Dest = ND 24, Src-Route List = <22,25 >
Src = ND 20, Dest = ND 24, Src-Route List = <21,23,25>

Figure 11. Source Routing

In Figure 11, suppose that ND 20 that wishes to send a packet to ND 24. The routing table on ND 20 may contain <21,23,25> as the source route for ND 24. In that case, ND 20 will originate a packet containing <21,23,25> in the header, and send the packet to ND 21. ND 21, upon receiving the packet will send it to ND 23 after finding it in the header. ND 23 will send the packet to ND 25. Finally, ND 25 will send the packet to ND 24 (the final destination). Alternatively, ND 20's route to ND 24 may be <22,25>. In that case, ND 20 will originate a packet containing <22,25> in the header, and send it to ND 22. In either case, the packet will end up at ND 24.

6.2.4 Security

End-to-end communications are managed on the Network Layer by sessions. A device may have more than one sessions defined for a given peer device. In fact, almost all network devices will have at least two sessions with the Network Manager: one for pair-wise communication and one for network broadcast communication from the Network Manager. All devices will also have a Gateway session. The sessions are distinguished by the Network Device addresses assigned to them. For the pair-wise session with the Network Manager, a device's standard Network Device address will be used; for the broadcast session, the special Nickname address 0xFFFF will be used[4].

A network device must keep track of security information (encryption keys, nonce counters) and transport information (reliable transport sequence numbers, retry counters, etc.) for each session in which it participates.

6.3 HART-IP

HART over IP primarily serves as a high bandwidth connection between Host Applications (e.g., process automation systems, plant asset management systems, etc.), HART-enabled I/O (e.g., multiplexers, Gateways) and HART-IP enabled instrumentation. It combines the ubiquitous Internet Protocol (IP) network infrastructure and the HART network and application layers. As a result, plant personnel can utilize infrastructure already deployed and understood to provide HART compatible system connectivity and integration.

[4] The Nickname 0x0000 is reserved and must not be assigned to any device.

7 NETWORK LAYER SERVICES

This section specifies the operation of the Network Layer from a "black box" point of view. This section specifies the Service Primitives (SPs) supplied by the Network Layer to the Application Layer. In addition to specifying the individual SPs, time sequence diagrams are included to indicate the order in which the SPs should be used and the order of event occurrence at the protocol layer boundaries. See *Token-Passing Data-Link Layer Specification* for more information on the service specification methodology.

The Services described in this section are used to obtain:

- Message services supporting bi-directional request/response communication traffic and unidirectional notifications (e.g., for publishing process data). These are common to all underlying Data-Link and Physical Layers.
- Management services for WirelessHART Network Layer configuration.

All SPs described here must be supported by the device unless otherwise stated. The mapping of these SPs into an implementation is entirely a local matter and is in no way restricted by this specification.

In the definition of the SPs, parameters are defined. Some parameters are optional and may not be present in all invocations of the SP. Optional parameters are distinguished by enclosing them within square brackets ("[","]") in the SP definitions.

7.1 Network Layer Message SPs

Message SPs provide services supporting the basic transfer of data between devices. The Network Layer supports request/response communications and one-way notification traffic. In addition to normal request/response traffic, the Application Layer may request guaranteed service. When requested, the guaranteed service will perform retries as needed to ensure a response is obtained from the destination device. The time sequence diagram for the message SPs is shown in Figure 12.

In Figure 12, Sequence 1 illustrates a block transfer sequence. Block transfer is an un-acknowledged propagation of a data segment across the network (see the *Block Data Transfer Specification*).

The transmit sequences illustrates request/response traffic between devices. Sequence 2 shows a basic, acknowledged request/response transaction. In this sequence, the TRANSMIT.request inserts the message into the Network Layer's transmit queue. When a message is received and validated, the correspondent Network Layer generates a TRANSMIT.indicate. In return, the Application Layer generates a TRANSMIT.response, which is communicated to the source Network Layer. Upon reception of the response, the source Network Layer generates a TRANSMIT.confirm to the Application Layer.

Finally, in Sequence 3, the notification or publishing transaction is illustrated. In this transaction, the TRANSMIT.confirm is generated immediately after the message is conveyed to the underlying Data-Link Layer. Publishing is an un-acknowledged and relies on over-sampled, repetitive operation to simplify the network transaction.

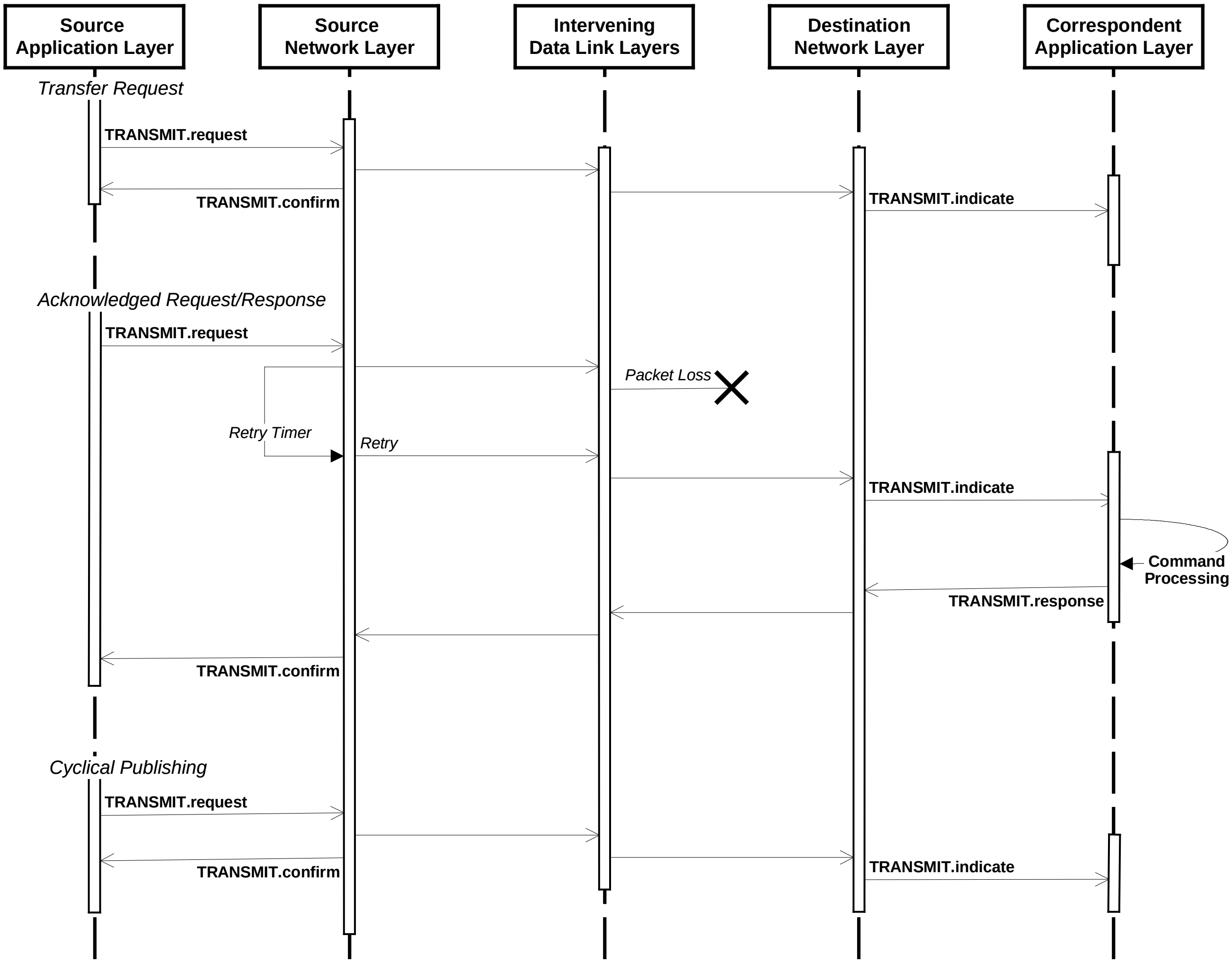


Figure 12. Network Layer Message Sequence

`TRANSMIT.request(packetHandle , dest, priority, timetableID, transportType, payload)`

This SP is used by the Application Layer to send the packet to one or more devices in the network. The parameters included with the service request include:

- ***`packetHandle`*** - The packetHandle is supported for the convenience of the Application Layer. The Network Layer returns this value in the corresponding TRANSMIT.confirm allowing the Application Layer to match requests with responses.
- ***`dest`*** - indicates the packet's destination and is one of the following:
  - ***`uniqueID`*** - the Unique ID (long address) of the destination device for payload. The NPDU shall include and the control byte shall indicate a long destination address.
  - ***`nickname`*** - the Nickname (short address) of the destination device for payload. The NPDU shall include and the control byte shall indicate a short destination address.
  - ***broadcast*** - the broadcast address is indicated in the NPDU and the control byte shall indicate a short destination address.

- ***priority*** - The packet priority is determined by the contents of the payload and is one of {management, process data, normal, or alarm} see *TDMA Data-Link Layer Specification* for more information on packet priorities.
- ***timetableID*** - It is the responsibility of the initiator of a transaction to ensure it has obtained sufficient communication bandwidth from the Network Manager to allow the transaction to commence. The timetableID identifies the bandwidth quota to be used for this transaction. If there is insufficient remaining bandwidth for the TRANSMIT.request then an error must be returned and the payload is not transmitted. If the destination is the Network Manager then timetableID is ignored (Network Manager is not restrained by a Timetable).
- ***transportType*** - The transport type (see Table 2) indicates the Transport Layer operation requested and is used to set the Transport Byte (see Figure 29). The TRANSMIT.confirm is generated as follows:
  - For Transfer Request, Transfer Response, Publish-Broadcast or Search-Broadcast service, the Network Layer promulgates the packet and immediately generates a TRANSMIT.confirm.
  - For a Request-Unicast or Request-Broadcast, the Network Layer shall perform retries to ensure a response from the destination device is received. If the retries are exhausted the TRANSMIT.confirm will indicate an error (see subsection 9.2).
  - For Publish / Notify the Network Layer promulgates the packet and immediately generates a TRANSMIT.confirm. Publish transactions are typically used to cyclically publish process data.
- ***payload*** - The contents of payload parameter is transmitted to the destination device.

Table 2. Transport Type Codes

		Transport Byte Setting		
Code	**Description**	**B'cast**	**ACK'ed**	
0	**Transfer Request.** This is used by the Block Data Transfer Mechanism (master side)			→ Req
1	**Transfer Response**. This is used by the Block Data Transfer Mechanism (slave side)			← Rsp
2	**Request-Unicast** (TRANSMIT.request only). This used by (master side) Devices executing Request/Response transactions (e.g., when configuring a device)		X	→ Req
3	**Response-Unicast** (TRANSMIT.response only). This used by (slave side) Devices executing Request/Response transactions (e.g., when configuring a device)		X	← Rsp
4	**Search-Broadcast** (TRANSMIT.request only). This is used to send a broadcast message when attempting to identify a specific device (e.g., Command 21)	X		→ Req
5	**Publish-Broadcast** (TRANSMIT.request only). This is a broadcast announcement to all network devices.	X		← Rsp
6	**Request-Broadcast** (TRANSMIT.request only). (e.g., changing Network ID)	X	X	→ Req
7	**Response-Broadcast** (TRANSMIT.response only). This is the unicast response to the corresponding Request-Broadcast.	X	X	← Rsp
8	**Publish / Notify** (TRANSMIT.request only). (e.g., process data)			← Rsp
9.	**Search-Response** (Transmit.request only) response to a Search-Broadcast	X		← Rsp

The Network Layer must be capable of buffering at least one message in addition to the message associated with the current transaction.

When this SP is invoked, it must validate the parameters (e.g., invalid destination address, no route to destination) and reject it if any errors are detected. In this case, the TRANSMIT.confirm SP shall be invoked indicating the error.

Otherwise the payload will the forwarded to the correspondent device by constructing the NPDU, invoking the Transport Layer (see Subsection 9.2), authenticating and enciphering NPDU (see Subsection 9.1.3) and forwarding the NPDU to the Data-Link Layer.

TRANSMIT.indicate (sequenceNumber, srcAddr, priority, transportType, payload)

This SP is invoked by the Network Layer when a packet is received, and provides the payload to the client layer. Parameters included in the SP include:

- ***srcAddr*** - indicates the address of the source device generating the payload. This address depends on the network topology and may be, for example, the Primary Master, the Gateway, or the Network Manager.
- ***sequenceNumber***- The sequenceNumber provided from the Network Layer to the client layer. The sequenceNumber must be returned in the corresponding TRANSMIT.response SP.
- ***priority*** - The packet priority as provided in the TRANSMIT.request
- ***transportType*** - The transport type (see Table 2). The client layer shall respond based on the transportType as follows
  - For a Transfer Request or Publish/Notify TRANSMIT.indicate no TRANSMIT.response shall be invoked. The transaction is complete.
  - For a Request-Unicast or Request-Broadcast the device must generate a corresponding TRANSMIT.response SP.
  - For a Search-Broadcast the device may generate a corresponding TRANSMIT.response SP.
- ***payload*** - The data to being transported.

TRANSMIT.response (sequenceNumber, payload, timetableID)

This SP is executed by the Field Device to respond to all incoming TRANSMIT.indicate SP that require a response (e.g., transportType "Request-Unicast"). The sequenceNumber must be identical to that provided in the corresponding TRANSMIT.indicate SP. The TimetableID identifies the bandwidth quota and route to be used for this transaction. If there is insufficient remaining communication bandwidth for the TRANSMIT.response then an error must be returned and the payload is not transmitted. If the destination is the Network Manager then the TimetableID is ignored (Network Manager is not restrained by any Timetables).

The payload either contains the response data or a delayed response. All responses must contain at least command completion status (i.e., the Response Code) for the command(s) in the TRANSMIT.indicate.

The transportType is inferred from the NPDU associated with the TRANSMIT.indicate and identifies the type of Network Layer transaction. Table 3 shows the transportType that must be inferred based on the code provided in the TRANSMIT.indicate.

Table 3. Transport Type Codes Pairs

Request-Unicast	Response-Unicast
Request-Broadcast	Response-Broadcast
Search-Broadcast	Search-Response

The device uses the addresses and priority associated with the TRANSMIT.indicate (identified by the sequenceNumber parameter) to generate and promulgate the NPDU back to the transaction originator.

TRANSMIT.confirm (packetHandle , localStatus, [payload])

This SP is returned to the Application Layer to communicate the results of a previously executed TRANSMIT.request. The slave response (if any) is returned.

For request packets, TRANSMIT.confirm returns the response payload. For notification, it indicates the packet has been sent to the Data-Link Layer (and the payload is empty).

In all cases localStatus indicates the resulting status of the communication transaction. The localStatus indicates success, warning or error. For warnings and errors the localStatus must have codes that indicate the cause of the warning or error (e.g., payload too large).

FLUSH.request (packetHandle)
Deletes the indicated packet.

FLUSH.confirm (packetHandle , localStatus)
Indicates whether the packet was deleted.

7.2 Wireless Network Layer Management Services

7.2.1 Local Management Services

Management SPs support both configuration of the Network Layer and access to statistics that it gathers. The fundamental SP is a LOCAL_MANAGEMENT sequence.

> Note: None of the SPs in this section require any data to be transmitted over the communication link. Remote management of the device's Data-Link Layer configuration is possible using Application Layer messaging of standard HART commands.

These SPs allow configuration on power up by the device's upper layers. This also allows management of the Field Device's non-volatile and programmable non-volatile memory to be isolated from the Network Layer implementation.

Management SPs may be accessed long after the Field Device has been on-line. For example, the Application Layer may receive a command from a network manager that changes the slots to be used when communicating.

LOCAL_MANAGEMENT.request(service, [data])
This SP is used to configure Network Layer properties. The parameters Services and Data are defined in the table below.

LOCAL_MANAGEMENT.confirm(service, status, [data])
This SP is used to return the results of a corresponding LOCAL_MANAGEMENT.request. The status shall return the results of the executed the request.

LOCAL_MANAGEMENT.indication(service, status, [data])
This SP is used to notify LOCAL_MANAGEMENT of an un-requested Network Layer event report

Table 4. Local Device Management Commands

Service	Data	Description
RESET		Reset and initialize the Network Layer. All network tables are cleared when this primitive is invoked. This primitive is normally invoked on device power-up or when the device is being installed in a new network.
WRITE_SESSION_KEY		Sets the session and nonce.
	Unsigned-8 sessionId	
	Bits-8 sessionType	Bitmap {broadcast, unicast, join}
	Unsigned-40 destNodeAddress	Long Destination address (Unique ID)
	Unsigned-16 destNickname	Short Destination address
	Unsigned-16 myNickname	The device's Nickname or the Broadcast Address
	Unsigned-128 sessionKey	

Service	Data	Description
	Unsigned-32 correspondentNonceCounter	
	Unsigned-8 *numSessions	
DEL_SESSION		delete a session
	Unsigned-8 sessionId	
	Unsigned-8 *numSessions	
ADD_ROUTE		add route to a given destination address..
	routeId	
	Unsigned-16 graphID	
	Bits-8 routeType	Bitmap: {Maintenance (default), Publish, Block Transfer, Event}
	Boolean isDefault	
	Boolean isGraphRoute	
	Unsigned-40 destUniqueID	Long Destination address
	Unsigned-16 destNickname	Short Destination address
	Unsigned-16 srcRouteHops [8]	Up to 8 Nicknames that lead to the destination. Unused addresses shall be set to the destination's Nickname
	Unsigned-8 *numRoutes	
DEL_ROUTE		delete route information
	routeId	
	Unsigned-8 *numRoutes	
DEFAULT_ROUTE		set given route as default
	routeId	
	Unsigned-8 *numRoutes	
READ_PDU_TIMEOUT		Reads the number of slots since packet's birth until it is discarded. Device checks ASN Snippet against current ASN
	Unsigned-16 maxPacketAge	Maximum number of slots a packet can live while hopping the mesh.
WRITE_PDU_TIMEOUT		Writes the number of slots since packet's birth until it is discarded. Device checks ASN Snippet against current ASN
	Unsigned-16 maxPacketAge	
READ_TTL		The value TTL is initialized to when a new packet is generated.
	Unsigned-8 TTL	
WRITE_TTL		
	Unsigned-8 TTL	

Note: * indicates that the value is returned.

7.2.2 Network Layer Constants and Attributes

Table 5. General Network Layer Attributes

Attribute	Description
Unsigned-40 Unique ID	Used to construct the EUI-64 (long) address. (EUI-64 can be constructed by pre-pending HCF OUI. See *TDMA Data-Link Layer Specification*)
Unsigned-16 Nickname	Short Address
Unsigned-8 DefaultTTL	Packet life limit in hops. Specifies the number of hops a packet can travel before being discarded (Defaults to 249)
Unsigned-16 maxPacketAge	Packet life limit in time using the ASN. Indicates the number slots after which a packet must be discarded. maxPacketAge may be set to any number of slots between 100 and 60,000 slots (Defaults to 30,000 slots i.e., 300 seconds)
Time HealthReportTime	Period at which to publish health reports (Defaults to 15minutes)
Time BcastReplyTime	Maximum amount of time to reply to a broadcast message. (Defaults to 60 seconds)
Time maxReplyTime	Used to trigger retires by Transport Layer (Defaults to 30 seconds)
Unsigned-8 maxRetries	Number of retries used by the Transport Layer when performing an acknowledged packet transmission (Defaults to 5).
Unsigned-8 minAdsNeeded	Preferred number of different Advertisements before issuing join request (Defaults to 3).
Time AdWaitTimeout	The amount of time to wait while attempting to receive additional Advertisements (Defaults to 30 seconds)
Unsigned-8 maxJoinRetries	Join retry limit (Defaults to 5)
Time JoinRspTimeout	Join response timeout (Defaults to 120 seconds)
Time ChannelSearchTime	The amount of time to stay on a given channel while listening for Advertise packets (Defaults to 400 milliseconds). Note: This actually used by the TDMA Data-Link Layer but discussed in this Specification.
Time ActiveSearchShedTime	Max amount of time to stay in active search mode while joining (Defaults to 4000 seconds). After this interval lapses the device transitions to passive search mode. Note: This actually used by the TDMA Data-Link Layer but discussed in this Specification.
Time PassiveCycleTime	When in passive search mode, the period over which the device cycles between sleeping and listening (Defaults to 600 seconds). The sleep interval equals the PassiveCycleTime minus the PassiveWakeTime. Note: This actually used by the TDMA Data-Link Layer but discussed in this Specification.
Time PassiveWakeTime	When in passive search mode, the amount of time to be awake listening for the network (Defaults to 6.5 seconds) Note: This actually used by the TDMA Data-Link Layer but discussed in this Specification.

Table 6. Session Table Attributes

Attribute	Description
Unsigned-8 SessionTableSize	Size of session table (i.e., maximum number of entries)
Unsigned-8 NumSessionTableEntries	Number of entries currently in the SessionTable.
SessionTable	Set of session table entries. See Subsection 9.1.3

Table 7. Route Table Attributes

Attribute	Description
Unsigned-8 RouteTableSize	Size of route table (i.e., maximum number of entries)
Unsigned-8 NumRouteTableEntries	Number of entries currently in the RouteTable
RouteTable	Set of route table entries

8 WIRED NETWORK LAYER SPECIFICATION

8.1 Data Field Format

This section defines the contents and format of the Command, Byte Count and Data fields when communicating via wired HART network. Figure 13 shows the Frame Format specified by the *Token-Passing Data-Link Layer Specification*. In other words, this section defines how host applications and slave devices interface to the HART Application Layer to accomplish two-way communication.

Delimiter	Address	Expansion	Command	Byte Count	[Data]	Check Byte

Figure 13. Token-Passing Data-Link Layer Frame Format

This section defines the format of the Data field based upon command number, master requests and slave responses, and the type of status information contained in the Data field. Beginning with HART 6 the Protocol supports 16-bit command numbers. These are supported using a backward compatible technique even though the Token-Passing Data-Link Layer native only supported an 8-bit command number. Central to the technique is the use of the "Command Number Expansion Flag". This flag consists of 31 (0x1F) in the Command field. When present it indicates that a 16-bit command number field is located inside the Data field. In other words, a 31 in the Command field indicates an extended command number and requires a minimum of two bytes[5] in the Data field.

Using this technique, the Data field supports two-way communication between devices in the following ways:

- When communication is successful, the Data field contains the information communicated between devices.
- For extended commands the Data field includes the two-byte extended command number allowing devices to determine unambiguously the information content of the message.
- For all slave responses the Data field contains at least the two Command Status Bytes providing continuous feedback to host applications.
- When an error is detected in the host communication, the slave response does not contain the information indicated by the command number. This gives host applications clear indication that the error prevented successful execution of the command.

To meet the above objectives the structure of the Data field varies between eight different Data field formats. Figure 14 can be used to determine the format of a Data field for any HART message. There are two master request formats (normal and extended command) and five possible slave response formats. Slaves generally echo the master request except when there is an error. When an error is detected, the slave response is truncated returning only the error information. For slave devices not supporting extended command numbers further simplification can reduce the data field formats to four: Master Request, Normal Slave Response, Communications Error Slave Response and Command Error Slave Response.

[5] If no data bytes are present, then a field device must answer "Too Few Bytes Received".

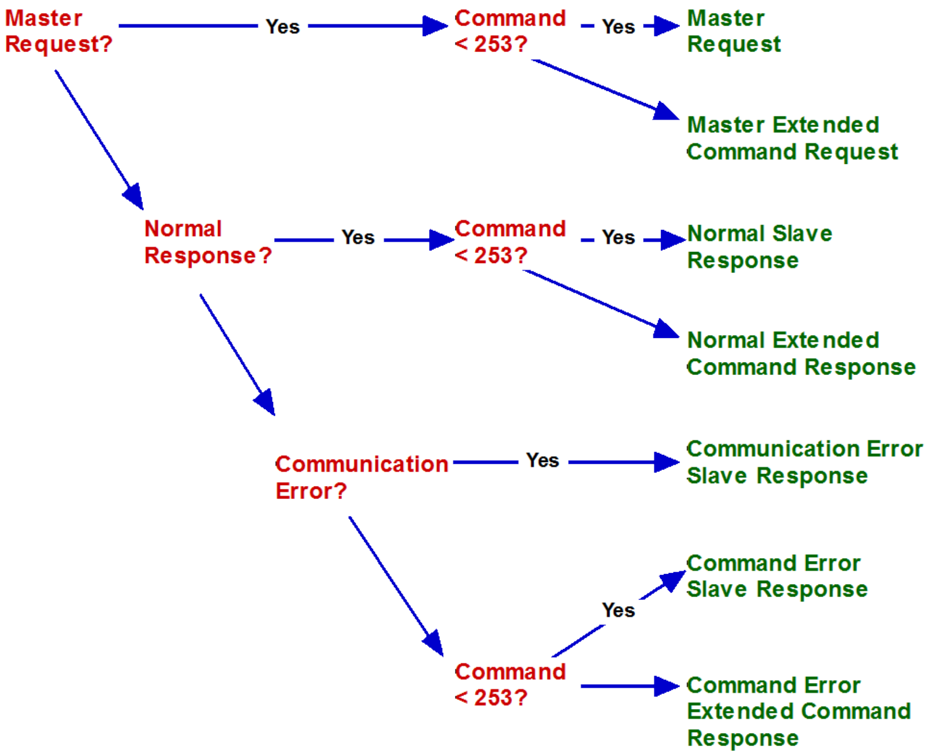

Master Request?
Yes
Command < 253?
Yes
Master Request
Master Extended Command Request
Normal Response?
Yes
Command < 253?
Yes
Normal Slave Response
Normal Extended Command Response
Communication Error?
Yes
Communication Error Slave Response
Command Error Slave Response
Yes
Command < 253?
Command Error Extended Command Response

Figure 14. Determining Data Field Format

These eight Data field formats consist of differing combinations of the following six sub-fields:

- **Request Data Bytes.** This sub-field contains the data items communicated from the host to the field device. The content and length of this field is defined in the specification for the issued command. For any command the bytes in this field are always numbered starting at zero, although the first byte in the Request Data may not be the first byte in the Data field of the message.

- **Response Data Bytes.** This sub-field contains the data items communicated from the field device to the host. The content and length of this field is defined by the Command Specification. The bytes in this field are always numbered starting at zero, although the first byte in the Response Data sub-field is never the first byte in the Data field of the message.

- **Extended Command Numbers.** This is a 16-bit command number used to extend the number of HART commands to 65,536 (see *Command Summary Specification*).

- **Communication Status.** This byte is multiplexed with the Response Code byte and indicates field device detection of a communication error. A communication error is always indicated by a one (1) in the most significant bit of this byte. When the field device does not detect a communication error, the Response Code is returned in the response message.

- **Response Code.** This byte is multiplexed with the communication status byte and indicates the status of the field device's execution of the host request. The most significant bit of the Response Code is always zero (0).

- **Device Status.** The Device Status sub-field provides a high level indication of the field device health and status. Field Device Status is returned in every slave response to provide continuous host feedback regarding field device health and operation.

These six sub-fields are not included in the construction of every message. The following sections describe message construction in application layer communication based upon the function being performed.

8.1.1 Requests With Single Byte Command Numbers

Figure 15 shows the format of the application layer fields that must be used for a master request containing a single byte command number. The complete command number is contained in the one byte Command field and the Request Data Bytes start at byte zero (0) of the Data Field.

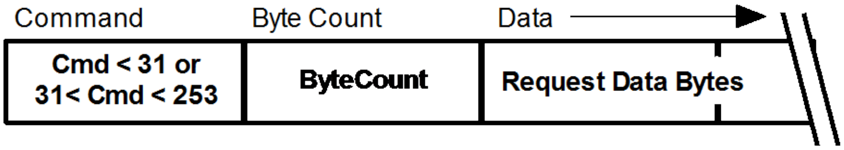


Figure 15. Master Request

All slave response messages must contain two status bytes (see Figure 16). Normally, the first status byte contains the Response Code providing information about the field device's execution of the command. See Section 8.1.3 for response formats when the field device detects an error.

The second status byte contains Field Device Status information. The Response Data Byte field follows these two bytes provided, an error was not encountered in the master request.

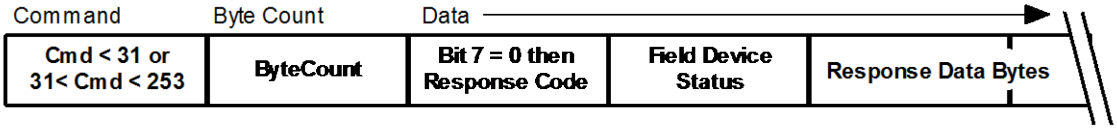


Figure 16. Normal Slave Response

8.1.2 Request With Extended Command Numbers

Figure 17 shows the format of the application layer fields that must be used for a master request containing an extended command number. When the Command field is 31 (0x1F), the first two bytes of the Data field must contain the Extended Command sub-field. These two bytes must be followed by the Request Data Byte field as defined in the command specification. A device receiving a message with a Byte Count less than 2 and 31 in the Command field must return the single-byte command error Response Code 5, "Too Few Data Bytes" (see Figure 20).

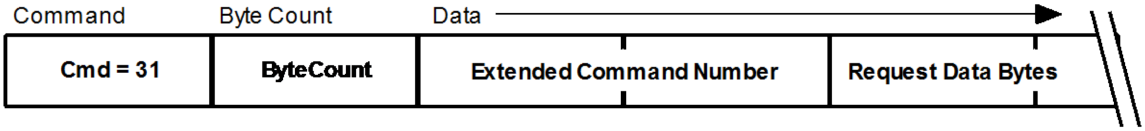


Figure 17. Master Extended Command Request

All slave response messages to extended commands must contain both the two status bytes plus the two byte extended command number (see Figure 18). The two status bytes are the same as for single-byte command numbers. Section 8.1.3 characterizes response formats when the field device detects an error.

The Response Data Byte field defined in the command specification must follow these four bytes provided an error was not encountered in the master request.

Command	Byte Count	Data →			
Cmd = 31	ByteCount	Bit 7 = 0 then Response Code	Field Device Status	Extended Command Number	Response Data Bytes

Figure 18. Normal Extended Command Response

8.1.3 Error Responses

Figure 19, Figure 20, and Figure 21 show the format of the response message when a field device detects an error in a the master request. The Response Data Bytes must not be returned when the field device detects an error in the master request.

A communication error has priority and is indicated when the most significant bit of the first byte of the Data field is set. See the *Data Link Layer Specification* for more information about slave device operation when a communication error is detected. Any slave response indicating a communication error must use the format shown in Figure 19.

Command	Byte Count	Data →	
Cmd	ByteCount = 2	Bit 7 = 1 then Comm. Error	Field Device Status

Figure 19. Slave Response with Communication Error

If there are no communication errors, the slave device must return the Command Response Code in the first byte of the Data field. Figure 20 shows the format of the slave response to a master request containing a single byte command number when the Response Code sub-field indicates an error. Response Codes must be less than 127 (0x7F) and have the most significant bit of the first byte of the Data field must be cleared (see the *Command Response Code Specification*).

Command	Byte Count	Data →	
Cmd < 31 or 31< Cmd < 253	ByteCount = 2	If Response Code is "Error"	Field Device Status

Figure 20. Slave Response for Single Byte Command with Command Error

Field Device Status must be returned in every slave response to provide continuous host feedback regarding field device health and operation.

For extended commands, the two-byte command number must be included in error response from the field device. The Response Code and the Field Device Status sub-fields are always returned first to maintain backward compatibility. Figure 21 shows the format of a slave error response[6] to an extended command.

Command	Byte Count	Data →		
Cmd = 31	ByteCount = 4	If Response Code is "Error"	Field Device Status	Extended Command Number

Figure 21. Extended Command Response with Command Error

[6] For example, this format would be used to return "Invalid Extended Command Number" if the Extended Command Number is 31. While not recommended, a device may also return "Invalid Extended Command Number" for any Extended Command Number less than 512.

8.2 Multi-drop Networks

A Multi-drop network connects multiple field devices and up to two masters across a common pair of wires or other medium. All devices shall support multi-drop operation. Masters must establish communications using one of the procedures in Section 11.2 . Independent of the Protocol, the interconnecting medium may place additional constraints on the number of field devices allowed on one network. For example, the *FSK Physical Layer Specification* assumes 17 (two Masters and 15 Slaves) devices when defining the maximum noise allowed to be generated by a device.

9 WIRELESS NETWORK LAYER SPECIFICATION

The Network Layer provides routing, end-to-end security, and transport facilities. It manages "sessions" for end-to-end communication with correspondent devices. Packets received via the Data-Link Layer's TRANSMIT.indicate SP, it transfers packets destined for the device itself from the Data-Link Layer to the client layer, and routes packets destined for other devices by sending them back to the Data-Link Layer. It also processes packets received from the Application Layer with the TRANSMIT.request primitive.

The following diagram shows how data packets flow through the Network Layer.

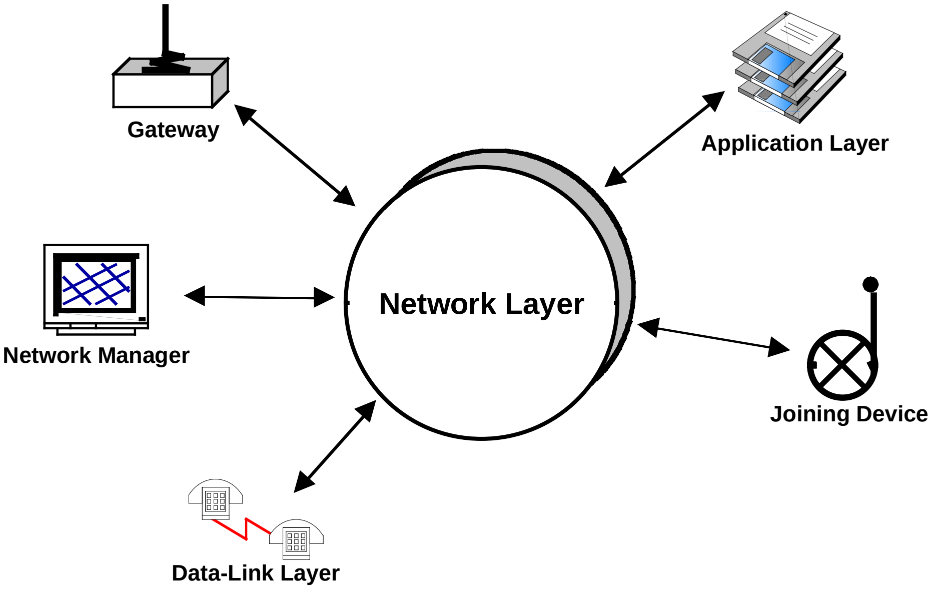


Figure 22. WirelessHART Network Layer Context Diagram

9.1 Wireless Network Layer PDUs

As shown in Figure 23, the WirelessHART Network Layer PDU consists of three distinct functions. First the Network Layer fields consist of those fields required to route the NPDU to its final destination. On top of that is a layer of security fields used to ensure private, unmolested communication between the NPDU's end points. Finally, the NPDU payload is enciphered and contains the information being exchanged across the network.

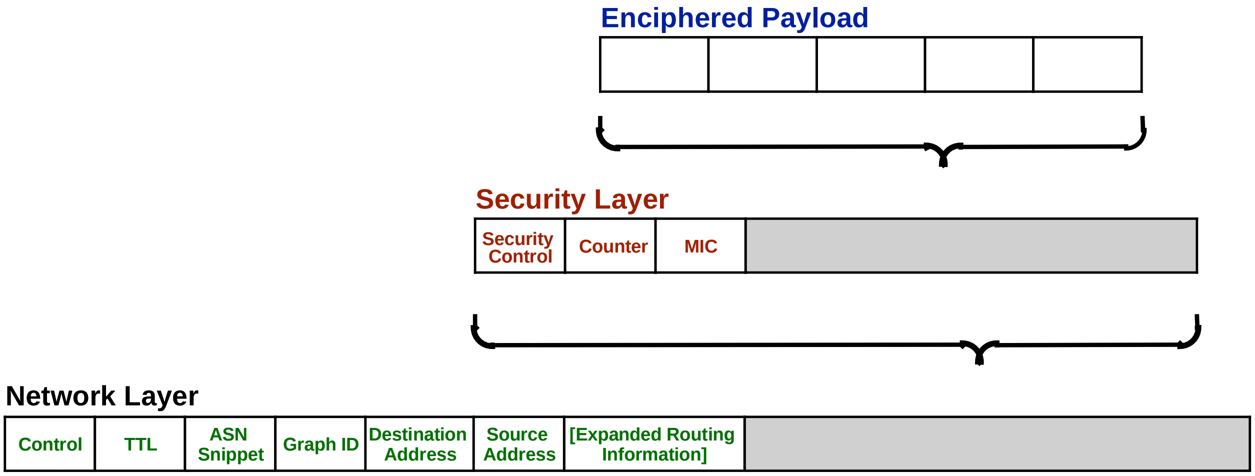


Figure 23. WirelessHART NPDU Structure

Collectively these three elements comprise the NPDU

9.1.1 Network Layer

The Network Layer PDU segment consists of the following fields:

- A 1-byte Control field;
- The 1-byte Time To Live (TTL) hop counter;
- The least-significant two-bytes of the Absolute Slot Number (Latency Count);
- A 2-byte Graph ID;
- The (final) Destination and (original) Source Addresses; and
- Optional routing fields.

The complete Network Layer PDU consists of these fields plus the security fields followed by the enciphered NPDU payload.

9.1.1.1 Control Byte

The first byte in the Network PDU is the **Control** byte (see Figure 24). The first two bits (bit 7 and 6) indicate whether the source and destination addresses are long (8-byte) EUI-64 addresses or short (2-byte) Nicknames. The next three bits (bits 5-3) are reserved and no device shall make any assumption regarding its possible future use. Devices built before any such future use is assigned shall set these bits to zero on transmission and masked off on reception.

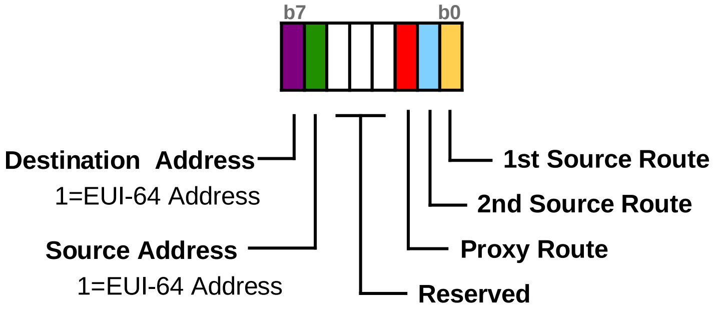


Figure 24. Network Control Byte

Bits 2-0 indicate, when set, the presence of the optional routing fields (see Subsection 9.1.2). When present, the proxy route is a 2-byte field and each Source Route is 8-bytes long. Consequently, based on bits 2-0, the length of the header can be extended by 0, 2, 10, or 18 bytes.

When the Network Layer receives a NPDU the **Destination Address** is inspected and, if the address matches the device's then the NPDU is authenticated and the payload deciphered. Once successful, the TRANSMIT.indicate primitive is invoked with the payload.

9.1.1.2 Time-To-Live

If the destination address does not match the device's then the **TTL** counter must be evaluated to determine whether the Network Layer discards or forwards the packet. The TTL counter controls the Time-To-Live for the packet and must be decremented on each hop the packet takes toward its final destination. When TTL reaches 0 the packet must not be forwarded to another device. If, when the packet is received by the Network Layer, the TTL is 0xFF then the TTL in not decremented and the packet is always forwarded onward toward its final destination (i.e., TTL is infinite).

9.1.1.3 ASN Snippet

The ASN Snippet field is set to the least significant 16 bits of the Absolute Slot Number when the Network Layer's TRANSMIT.request SP is invoked. This field provides coarse but critical real-time performance metrics and diagnostic information on the operation of the network. It also provides, when the full ASN is recreated, the age of the packet[7].

If the TTL is valid then the packet age must be compared to maxPacketAge. If the packet age is greater than the maxPacketAge the packet must be discarded. The difference between packet age and maxPacketAge shall be provided as the timeout to the Data-Link.

9.1.1.4 Graph ID

The Graph ID is used to route the packet to its final destination. The Graph ID identifies a list of nodes, any of which can be used for forwarding the packet toward the final destination.

Otherwise, the packet is forwarded based on the Graph ID and the other routing information (if present) included in the Network Layer Header.

[7] Packet age can be calculated as

```
IF   ((CurrentASN && 0xFFFF) < ASN snippet)
        ((CurrentASN && 0xFFFF) – ASN snippet) + 63356);
ELSE
        (CurrentASN && 0xFFFF) – ASN snippet
ENDIF
```

9.1.1.5 Source/Destination Addresses

The Source and Destination Addresses are each either 2 or 8 byte addresses. For more information, see the *TDMA Data-Link Layer Specification*. These addresses are not modified during propagation of the NPDU.

9.1.2 Special Routes

If any of bits 2-0 are set then one or more of the optional routing fields shown in Figure 25 are present in the NPDU header. When multiple fields are present they are included in the NPDU header in the order depicted. These fields include

- An optional 2-byte Nickname address for the proxy device; or
- Up to two optional Source Route segments each containing four Nicknames (8-bytes each).

[Proxy Address]	**[1st Source Route Segment]**	**[2nd Source Route Segment]**

Figure 25. Expanded Routing Information

9.1.2.1 Proxy Route

Proxy routing is used to communicate with devices that have not yet been integrated into the network by the Network Manager (i.e., devices that are in the process of joining). When bit 2 of the NPDU Control byte is set, the 2 byte Nickname[8] of a specific proxy parent for the final destination is indicated in the Proxy Address.

If the Proxy Address matches, the proxy parent is responsible for forwarding the NPDU to the device indicated by the Destination Address (see Figure 23).

9.1.2.2 Source Route

If the source route field is present, the addresses it contains are used to route the packet. Source routes should only be used to test or troubleshoot network paths.

Each Source Route field contains four addresses that designate the route the packet must follow. Each address is a 2-byte Nickname. Using the two segments, two to eight intermediate device addresses may be specified. The device addresses must be specified sequentially. Should all the addresses in a segment not be used the specified addresses are followed by the unused addresses set to 0xFFFF. When scanning a source route sequence the scan is terminated upon encountering the first 0xFFFF. When the end of the source route is reached (there are no more nicknames or a 0xFFFF terminator is encountered) the packet is transmitted to its final destination or along the NPDU Graph Route.

9.1.3 Security Sub-Layer

The security layer header of the NPDU (see Figure 26) is designed to: ensure private, unmolested communication; and allow for future possible enhancements to WirelessHART security. To this end, the security layer starts with a Security Control Byte that specifies the security employed. This field is followed by the fields needed by the security algorithms employed.

Security Control	**Counter**	**MIC**	

Figure 26. Security Sub-Layer

As indicated in Figure 27, the Security Control byte consists of a 4 bit enumeration (bits 0-3) that indicates the security strategy employed for this NPDU. The most significant bits (bite 4-7) of the Security Control byte are reserved and no device shall make any assumption regarding their possible future use. Implementations must mask off the most significant 4 bits. Devices built before any such future use is assigned shall set these bits to zero on transmission.

[8] The proxy Nickname must not be a broadcast address (i.e. 0xFFFF). The Nickname 0x0000 is reserved and must not be assigned to any device.

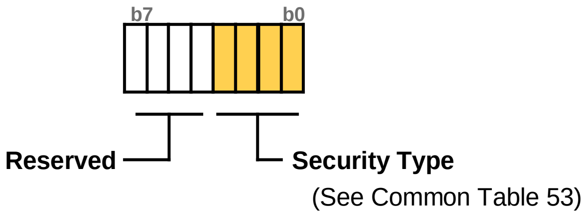


Figure 27. Security Control Byte

The overall length of the security layer header depends on the type of security employed (see Table 8). Furthermore, the NPDU payload is always encrypted using the technique indicated by the Security Type Sub-field.

Table 8. Security Layer Sizes

Security Type	Counter Length	MIC	Total Length
Session Keyed	8bit (LSB of 32bit Nonce Counter)	32Bits	6Bytes
Join Keyed	32bit Nonce Counter	32Bits	9Bytes

9.1.3.1 Sessions

For security, WirelessHART is session oriented and all devices must support multiple sessions. A session enables private and secure communication between a pair of network addresses. A field device always contains a Join session. In addition, four sessions are generally set up as soon as the device joins the network:

- Network Manager and the device (unicast). This session is used by the Network Manager to manage the device.
- Network Manager broadcast (to all devices in the network). All devices in the network have the same key for Network Manager broadcasts. This session is used to globally manage devices. For example, this can be used to roll a new Network Key out to the network.
- Gateway and device (unicast). This carries normal communications (e.g., process data) between the gateway and the device.
- Gateway broadcast (to all devices in the network).

Additional sessions may be added (e.g., to a handheld or another field device). Other than the Join session, the Network Manager creates all sessions and their corresponding key (e.g., when the device joins the network). Only the Network Manager may create or modify sessions. The Join key is unique, it is the only key that can be written by the Network Manager or using the device's maintenance port. Join sessions are always present in the field device and cannot be deleted.

> It is possible to deploy a session that connects two arbitrary devices in the network. However, this can result in undetected and unmonitored communication between the devices and, consequently, represents a security and safety threat. If the Network Manager supports peer-to-peer sessions between field devices all of the resulting communications should be routed via the Gateway thus allowing the detection and disruption of malicious behavior.

When and only when a new session is created, myNonceCounter is reset to zero and the peerNonceCounter is set to the Nonce Counter Value in the Write Session command (see *WirelessHART Command Specification*). All bits in the nonceCounterHistory shall be set (to 1) when the session is created. The Session Key itself is a write-only value. No device shall provide a means to read back any key (Session, Join, Network).

The attributes of a session are shown in Table 9. The combination of the correspondent address and session type must be unique (no other session may have both the same correspondent address and session type entries).

Table 9. Session Records

Name	Description
Enum-2 sessionType	One of {Unicast, Broadcast, Join}
Unsigned-40 peerUniqueID	Long address of correspondent (Expanded Device Type Code + Device ID). Note: The EUI-64 is constructed by pre-pending the HCF OUI. See *TDMA Data-Link Layer Specification* more information.
Unsigned-16 peerNickname	Short address of correspondent device.
Unsigned-128 sessionKey	(Write Only) Active Session key.
Unsigned-128 altKey	The pending or old Session Key.
Unsigned 40 TimeToSwitchKeys	ASN when the pending key becomes the active key. After this ASN the ALT key is the old Session Key. The old key must be discarded after 2 * "Maximum PDU Age"
Unsigned-32 peerNonceCounter	Largest nonce counter value received from the correspondent device.
Bits- *n* nonceCounterHistory Note: The value of *n* depends on the type of device (See the *WirelessHART Device Specification*)	An array of bits[9] recording the nonce counters received. Most significant bit is always set and corresponds to the current peerNonceCounter value. The least significant bit corresponds to 1+peerNonceCounter - sizeof (nonceCounterHistory) This creates a sliding window with each set bit recording a NPDU received with the corresponding nonce counter. The sizeof (nonceCounterHistory) equals the number of received NPDUs recorded. As the nonceCounterHistory is right shifted the history of the older NPDU are dropped.
Unsigned-32 myNonceCounter	Nonce counter for packets sourced by the device.
Ref transportable	Reference to Transport tables (normally 2) associated with correspondent
Ref routeTable	Reference to Route tables associated with correspondent destination

9.1.3.2 Join Sessions

Field Device

Join sessions are unique, are created when the field device is first powered up in the factory and persists (i.e. it is non-volatile) thereafter. The addresses of the Join session are set to the standard Network Manager addresses (see Subsection 6.2.2.5). The myNonceCounter entry is set to zero when the field device is first powered up, is incremented as it is used, and is never reset thereafter (even if the field device is power-cycled or reset). The field device's myNonceCounter is used for both join requests from the field device and join responses from the Network Manager. To protect against replay attacks on the Join Key, myNonceCounter is non-volatile.

Upon entering the "Requesting Admission" State (see Subsection 9.4.2) the field device must set (to 1) all values in the nonceCounterHistory.

Network Manager

Several strategies might be used to create the Join session on the Network Manager. For example, a Join session for a specific field device could be created in advance using the field device's Unique ID and the Join Key. No matter the strategy used to create the join session, when the Network Manager receives and validates a join request the NPDU counter field is copied to the session's nonce counter entries and is used to construct the join response.

[9] An array of bits is one implementation. For example, alternatively an array of nonce counter values (integers) could be used.

Table 10. Join Session

Name	Description
Enum-2 sessionType	Unicast
Unsigned-40 peerUniqueID	0xF980 0x000001
Unsigned-16 peerNickname	0xF980
Unsigned-128 sessionKey	(Write Only, Non-volatile) Join Session key.
Unsigned-32 peerNonceCounter	(Not Used)
Bits-32 nonceCounterHistory	32-bit nonce history. All bits set to 1 when device begins to join a network.
Unsigned-32 myNonceCounter	Non-volatile. Set to 0x00000000 at factory and never reset (only incremented) from then on.
Ref transportable	Reference to Transport tables (normally 2) associated with correspondent. These are null for join sessions (join requests are not "acknowledged" Transport Layer operations).
Ref routeTable	Reference to Route Table (Generated from the received Advertisement PDU)

9.1.3.3 NPDU Encipherment

A four-byte, keyed Message Integrity Code (MIC) is used for authentication of NPDUs and the deciphering of the Network Layer payload. All NPDUs that fail to authenticate shall be discarded. The NPDU Header is not enciphered to allow intermediate devices to successfully route the packet. The NPDU payload is enciphered to ensure communications remain private and secure.

A keyed MIC is used to ensure that the NPDU arrives successfully and unmolested from the indicated source device. The MIC is generated and confirmed using CCM* mode (Counter with CBC-MAC (corrected)) in conjunction with the AES-128 block cipher to provide authentication. This cipher requires four byte-strings as parameters:

- 'a', the additional data to be authenticated but not enciphered;
- 'm', is the message to be enciphered;
- 'N', the 13-byte nonce; and
- 'K', the 128-bit AES Key.

The NPDU payload is enciphered and is the byte-string 'm'. The NPDU header, from the NPDU Control byte through the NPDU MIC, is the byte-string 'a'. The TTL, Counter and MIC fields in byte-string 'a' are set to zero while enciphering the NPDU. These are replaced with their actual values before transmitting the packet.

The Network Layer Nonce (the 'N' byte-string) is 13-bytes long and shown in Table 11. Except for join responses, to create the Nonce:

- N[0] is set to zero.
- myNonceCounter is pre-incremented by one and written to the Nonce.
- The NPDU source address field is loaded into the Nonce (either the EUI-64 address or, if the device is currently joined in the network, the zero-padded Nickname).

For join responses, create the Nonce by:

- Setting N[0] to one;
- Using the peerNonceCounter value (from the join request) as the Nonce Counter; and
- Loading the joining device's EUI-64 address into the Nonce.

Table 11. NPDU Nonce (Byte-String 'N')

Byte	Format	Description
0	Unsigned-8	Set to 1 if Join Response otherwise set to 0
1-4	Unsigned-32	Nonce Counter (starting with MSB in N[1])

For EUI-64 source address

5	Unsigned-8	0x00 (HCF OUI)
6	Unsigned-8	0x1B
7	Unsigned-8	0x1E
8-9	Unsigned-16	Expanded Device Type Code (starting with MSB in N[8])
10-12	Unsigned-24	Device ID (starting with MSB in N[10])

For Nickname source address

5-10	Unsigned-48	Each byte set to 0x00
11-12	Unsigned-16	Nickname (starting with MSB in N[11])

Once the Nonce is constructed, the NPDU is enciphered. Once encipherment is complete, the TTL field is initialized and the NPDU security fields are populated. The entire NPDU is assembled by concatenating the NPDU Header, the security header, and the enciphered NPDU payload. Next, the NPDU is passed to the Data-Link Layer for propagation.

9.1.3.4 NPDU Authentication

At the destination device, the NPDU is once again processed using the AES-128 engine to authenticate the NPDU and decipher the payload. This cipher requires four byte-strings as parameters:

- 'a', is the NPDU header (NPDU Control through MIC) with the NPDU TTL, Counter and MIC fields set to zero;
- 'm', is the NPDU payload;
- 'N', the 13-byte nonce (see Table 11); and
- 'K', the 128-bit AES Key.

Authentication and decipherment starts by locating the correct session to determine the key and nonce counters. Next, the Nonce is constructed.

Nonce Re-Construction

The NPDU Nonce is re-constructed from the source address and by re-constructing (if necessary) the nonce counter. First, if the NPDU is a join response N[0] is set to one otherwise it is set to zero. The NPDU source address field is loaded into the Nonce (either the EUI-64 address or the zero-padded Nickname). Next, the nonce counter is constructed:

- If the message is a join request or response then the NPDU Counter is four-bytes long and must be copied to the nonce counter (N[1] - N[4]).
- Else, the NPDU Counter is one-bye long and the nonce counter must be reconstructed.[10]

The reconstructed nonce count is compared to the nonceCounterHistory. If it corresponds to any of the bits set in the nonceCounterHistory, the packet must be discarded. If the reconstructed nonce count is less than peerNonceCounter - sizeof(nonceCounterHistory) then the packet must be discarded.

[10] One possible algorithm is, for example, subtract the NPDU Count from the (peerNonceCounter +128) and put the three most-significant bytes of result in N[1]-N[3]. Copy the NPDU Count in N[4]. This algorithm seems to work well for received nonce values +128/-127 of the peerNonceCounter.

For Gateways or Network Managers supporting longer nonce histories, authentication must be performed with the reconstructed nonce counter and the reconstructed nonce counter plus and minus 256 to confirm the nonce used by the NPDU.

Authentication
If the nonce is successfully constructed and the packet verified to be unique (i.e. it is not a replay of a previous packet) authentication and decipherment can be performed. If the authentication fails, the packet is discarded.

Replay Protection
If the packet is authentic then the nonceCounterHistory is updated. Since NPDUs can arrive out of order or be lost completely, this sliding window algorithm is used to facilitate the communications and eliminate duplicate packets. The peerNonceCounter plus the nonceCounterHistory chronicles the NPDUs sinked by the device[11].

When a new Nonce Counter (N[1] - N[4]) value is encountered the peerNonceCounter plus the nonceCounterHistory must be updated to record its reception. If the received Nonce Counter is less than the peerNonceCounter the appropriate bit in the nonceCounterHistory must be set.

Otherwise, the nonceCounterHistory must be right shifted and the peerNonceCounter value updated. This is accomplished by subtracting the peerNonceCounter from the Nonce Counter. The resulting difference indicates the number of times the nonceCounterHistory must be right-shifted. After shifting, the MS bit of the nonceCounterHistory must be set. Finally, the Nonce Counter is copied to the peerNonceCounter in the session table entry.

9.1.4 Payload

For security, the payload field is always enciphered to prevent observation by intermediate devices as the NPDU traverses the network. The payload consists of Transport Layer information, Transaction ID, Device Status, Extended Field Device Status, and one or more commands. For more information see Subsection 9.2.

[11] If the PDU has a nonce counter less than can be recorded in the nonceCounterHistory, the PDU must be discarded.

9.2 Wireless Transport Layer

The Data-Link Layer ensures packets are successfully propagated from one device to another. The Transport Layer can be used to ensure end-end communication is successful. In other words, the Transport Layer can ensure packets are communicated successfully across multiple hops to their final destination. The Transport Layer supports both acknowledged and un-acknowledged transactions.

Un-acknowledged operation allows devices to send packets without requiring end-end acknowledgement and with no guarantee of packet ordering at the destination device. This method is useful, for example, for publishing process data. Since process data is propagated periodically end-end acknowledgement and retries have limited utility considering a new data point will be generated on a regular basis

In contrast, the acknowledged operation is used to construct a synchronous transport pipe across the network connecting to devices. The "transport pipe" allows devices to send packets and confirm their delivery. The Transport Layer orders the packets sent between devices' and tracks their delivery. This method is best suited for request/response traffic. When the acknowledged operation is employed the communication is synchronous. Only one transaction across the bus for that transport pipe is performed at a time.

Using acknowledged operation allows the client layer to ensure retries are automatically performed if communication latency becomes excessive. Since the Data-Link, in general, does not discard any packets it has assumed responsibility for the communication reliability for a well-formed WirelessHART mesh network normally greater than 4σ (99.9937%). Consequently, the primary benefit provided by the acknowledged operation is that the client layer will receive positive acknowledgement after NPDU arrival at the destination. This allows synchronous operation across the network as well as assuring the client layer that the NPDU was delivered and acknowledged.

9.2.1 Transport Layer PDU

This subsection specifies the format of the Transport Layer packet (TPDU). Each TPDU consists of the following fields:

- A transport byte used to ensure end-end packet delivery;
- The Device Status and Extended Device Status bytes; and
- One or more HART commands

Figure 28 illustrates the basic TPDU structure.

When the Network Layer TRANSMIT.request SP is invoked, (see Subsection 7.1) the transportType is indicated. This allows the Transport Layer to correctly configure the Transport byte (see Figure 29). A Transport Layer transaction is modeled as

- A "master" issuing a request packet and one or more "slaves" replying with a response packet; or
- A slave publishing a response packet.

For request/response transactions, the master may generate a broadcast request to the entire network. For example, the Gateway uses broadcast to "Poll by Long Tag" to find the device with the desired name.

Alternatively, a master may generate a packet and propagate the request to a specific device (i.e., Unicast request). For example, a timetable request generated by a field device is a request directed to the Network Manager. In turn, the Network Manager must generate a response packet and send it to the device.

For transactions consisting of only a response the Transport Layer simply pushes the packet to the destination. Burst Messages (see the *Common Practice Command Specification*) use this technique.

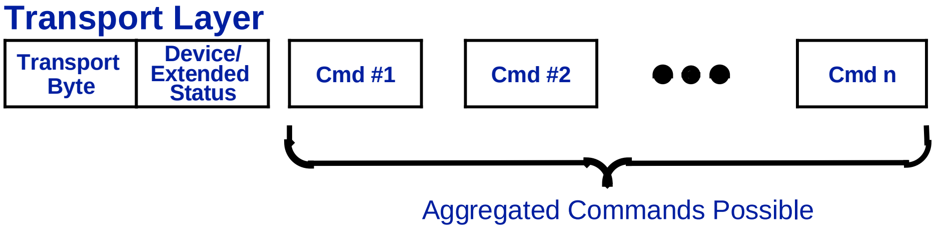


Figure 28. Transport Layer

9.2.1.1 Transport Byte

The first byte in the TPDU is the transport byte (see Figure 29) contains 4 subfields:

- The most significant bit (bit 7) is set when a transport pipe (i.e., acknowledged operation) is to be used. When set, the Transport Layer ensures that all requests are acknowledged and generate the TRANSMIT.confirm SP upon reception of the acknowledgement.
- The next bit (bit 6) is set when the NPDU is a response packet. If the response bit is set the payload shall only contain command responses.

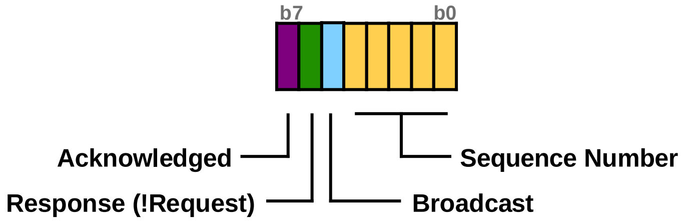


Figure 29. Transport Byte

- Bit 5, when set, indicates that a broadcast transport pipe is being used. This bit is set identically in the device response NPDU as in the request NPDU. This bit is used to identify the session (unicast or broadcast) that is the parent of the transport pipe and thus locate the correct transport table entry. The broadcast request may target all devices or (as indicated in a payload containing a Command 21 request) a single device. All devices receiving a broadcast NPDU that applies to themselves must generate a response packet directed to the source device.

  Bit 5 is ignored when the Acknowledged bit is reset.
- The sequence number field used to order and track packet traffic.
  - For acknowledged traffic, the sequence number is incremented by the transport master when it generates a request.
  - For un-acknowledged communications the sequence number shall be set to the least significant 5 bits of the packetCounter. The packetCounter is initialized to zero when the transport table is created and incremented for each unacknowledged packet conveyed using this transport pipe
  - In response packets, the slave returns the sequence number found in the request.

9.2.1.2 Device/Extended Status

The Device Status and Extended Device Status bytes are included in all TPDUs. The format of the Device Status byte can be found in the *Command Summary Specification*. Information about the Extended Device Status byte can be found in Common Table 17.

9.2.1.3 Aggregated Commands

With some limitations, WirelessHART allows multiple HART commands to be transported in a single transaction[12]. This is especially useful when reading device configurations. WirelessHART natively supports HART 16-bit command numbers. The format of commands transported over WirelessHART is shown in Figure 30. They consist of a 16-bit command number, the (1-byte) length of the data field (including the Response Code, if appropriate) and the data field. Commands 0-255 are zero filled to form the 16-bit command number (e.g., Command 9 is transmitted as 0x0009). Command 31 is reserved and any response to a Command 31 request must respond with "Command Not Implemented".

[12] Devices must parse each embedded command in order generating the appropriate response data and Response Code. See Command 78 (in the *Common Practice Command Specification*) for more information.

16-Bit Command Number	Length	Data

Figure 30. WirelessHART Command Format

The transport byte indicates whether the transaction is a request or a response. For responses, the first data byte is always the Response Code[13]. The response must contain the same commands in the same order as in the request. If the data field is too long to fit in the NPDU then "Payload Too Long" (Response Code 60) must be returned.

General requirements for aggregating commands include:

- In general, WirelessHART commands (0x0300 - 0x03FF) are not aggregated with other HART commands. This simplifies command parsing in multi-processor designs.
- Only network manager write commands shall be included in "Network Management" transactions. Network Management transactions are acknowledged NPDUs containing write commands that modify network behavior.
- Many HART commands are fully autonomous and some host applications can "fire and forget" these commands. For example, in many cases, a combination of read commands can be aggregated in "Routine" transactions.

Additional restrictions may be imposed in the requirements for specific commands. Furthermore, certain processes (e.g., device calibration) may prevent command aggregation. This behavior is already supported in host applications.

9.2.2 Transport Table

Like security sessions, devices must track multiple transport pipes. The properties that must be managed for each active transport session are shown in Table 12. For each acknowledged communication link a new entry in the Transport Table entry is created. Field devices will act as a slave in at least three cases (Unicast with Network Manager, Broadcast with Network Manager, and Unicast with Gateway). These support request/response traffic used to configure and manage the field device.

Note: Data Publishing, Notifications and Block Data Transfer are unacknowledged and, consequently, do not require a transport table entry.

On the other hand the Gateway and the Network Manager will need to track many transport sessions using several for each device. For example, when an acknowledged broadcast is generated by the Network Manager or Gateway the Transport Layer shall ensure acknowledges are received from all affected devices (e.g., every device in the network). In this case the transport table tracks considerably more information than shown in Table 12.

Each entry in the Transport Table includes bits indicating whether it is active (a transaction is in process), the device is performing as the master, and whether it is Broadcast.

When the transport table entry is for the master end of the pipe, the retry count and retry timer are tracked. When the retry timer lapses the request TPDU is resent and the retry counter incremented. When the maximum retries is exceeded the Transport Layer notifies its client that a fault has occurred. The counter must be reset whenever a new transaction commences.

Most importantly, the (if acting as a master) last request payload or (if acting as a slave) the last response payload is cached along with the corresponding sequence number. This allows the transport master to resend the request if needed. Furthermore, if the device is the transport slave, the response can be resent if a repeated request is received. Repeated requests are identified by the repetition of the sequence number. The sequence number is initialized to a random number by the master device when the transport table is created and then incremented for each new transaction.

[13] Unlike Token-Passing Data-Link Layer Commands, Device Status is not included in the data field of aggregated WirelessHART commands (See Figure 28).

Table 12. Transport Records

Content	Description
Bits-1 Active	Set if the transport is ACTIVE (reset if acknowledge received)
Bits-1 Master	Set if the device is the MASTER (i.e., the side sending requests)
Bits-1 Broadcast	Set if the request packet is Broadcast
Unsigned-5 sequenceNumber	On Master, the sequence number for the outstanding request On Slave, the sequence number last acknowledged.
Unsigned-8 TPDUHandle	On Master only, the handle from the TRANSMIT.request corresponding to lastTPDU.
Byte[] lastTPDU	On Master, the last unacknowledged packet (or NULL if none) On Slave, the last acknowledged payload
Unsigned-8 retryCount	On Master only, Number of communication attempts
Time responseTimer	Timer to trigger a retry

9.2.3 Transport Layer Operating Sequence

The basic operation is shown in Figure 31 using the sequence performed when the Network Key is changed. The Write Network Key (see the *WirelessHART Command Specification*) includes the key and the Absolute Slot Number indicating when use of the key shall commence.

The sequence begins with the Network Manager (the master in this example) generating an Acknowledged Broadcast Request NPDU and promulgating it to the network (see ①). Since it is a broadcast request, the transport table entry associated with the Broadcast Session entry in the Network Manager must be used.

The Network Layer propagates the NPDU to all devices in the network. Each device, in turn, acknowledges the Write Network Key command (see ②). If necessary the device creates the Transport Table Entry associated with the Network Manager-device Broadcast session.

The device replies and the acknowledge is enciphered using the unicast device-Network Manager session. This ensures intervening devices cannot, for example, spoof the acknowledge. However, to allow the Network Manager to correlate the acknowledge to the correct transport table entry, the broadcast bit remains set in the Transport Byte.

The Network Manager's Transport Layer receives acknowledges from all devices (retrying as needed). At the stipulated Absolute Slot Number the Network Key in all devices roll over to the new value (see ③).

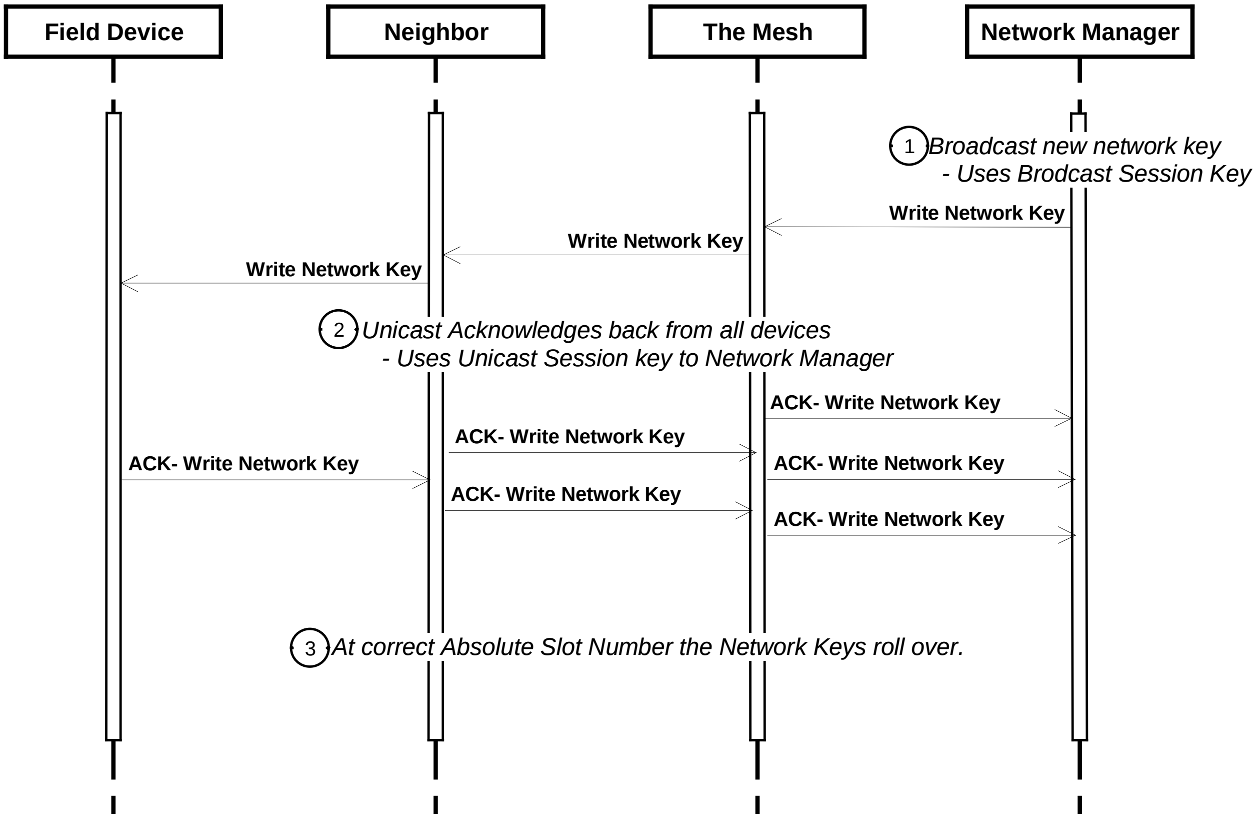


Figure 31. Using Transport Layer to Change Network Key.

9.2.4 Transport Layer Operation

The Transport Layer builds upon the end-to-end secure sessions provided by the Security sublayer (see Subsection 9.1.3). The security sessions establish a connection between two devices. For each security session there can be two one-way transport pipes (i.e., the field device is the slave for one of the pipes and the master in the other). For each security session the corresponding transport table entries shall be created automatically or when the transport pipe is first used.

To summarize, the basic reliable operation consists of:

- The master generating a request
- The master's Transport Layer tagging it with a sequence number
- The request's propagation through the network
- Reception by the destination (slave) device
- Processing of the request and the generation of the response
- The slave propagating the response back to the master using the same sequence number.

During this process the master will use the response timer to trigger retries as needed to ensure delivery of the request and reception of the slave's response. An error is signaled if the retry counter exceeds its maximum value.

9.2.4.1 Master Request Generation

When the client layer invokes TRANSMIT.request SP the transportType parameter is inspected and the most significant three bits of the Transport Byte are set accordingly.

Un-Acknowledged Operation

If the Acknowledged bit is reset then the least significant five bits of the packetCounter are copied into the Sequence Number field. Construction of the NPDU is completed and it is passed to the security sublayer for transmission. The TRANSMIT.confirm SP is invoked and the Network Layer transaction is complete.

Acknowledged Operation

If the Acknowledged bit is set, then the corresponding Transport Table entry must be located or, if it does not exist, the entry must be created.

If there is already an unacknowledged packet pending, one already buffered up for this transport pipe, and no more buffers are available then the TRANSMIT.request fails and TRANSMIT.confirm SP is invoked to signal the error and the request is discarded.

When the transport pipe is available, the sequenceNumber from the table entry is incremented and the least significant five bits are copied to the corresponding field in the transport Byte. The NPDU must be then passed to the security sublayer for transmission. The packet must also be buffered for possible future retries.

The Active flag is set in the Transport Table entry and the retryCount is initialized to 0. The replyTimer is initialized to maxReplyTime and started to await the response.

9.2.4.2 Master Retries

When a replyTimer expires a retry must be generated. The master shall increment the retryCount. If the count is exhausted (i.e., it exceeds maxRetries) then the TRANSMIT.confirm SP is invoked to notify the client layer of the failure. The Active flag is reset and the packet buffer released.

When the retries are exhausted a Transport Layer Failure occurs and the device (Gateway, field device, etc.) must notify the Network Manager (e.g., using Command 791). Upon receiving notice of a Transport Layer Failure the Network Manager must delete and reestablish the corresponding Session. If this is not possible, then the Network Manager must force the offending device to rejoin the network.

If the retries are not exhausted, the saved copy of packet (i.e., the lastTPDU) is resent with the same Transport Byte as used in the original packet.

9.2.4.3 Propagation to the Destination Device

When received from the Transport Layer the packet is processed by the Security Sublayer and propagated to the Data-Link Layer. From there, it is sent across the network and after possible several hops the packet arrives at its destination(s). During transit the NPDU (including the Transport Byte) are enciphered to prevent their molestation.

9.2.4.4 Receiving the Request NPDU.

When the device receives a request NPDU from a correspondent master, it must inspect the Transport Byte.

Un-Acknowledged Operation

If the Acknowledged bit is reset then the TRANSMIT.indicate SP is invoked with the data field and the sequenceNumber set to the value in the Transport Byte. The Transport Layer transaction is complete.

Acknowledged Operation

However, if the Acknowledged bit is set then the sequence number must be validated. First, the correct Transport table entry must be found by locating the session associated with the correspondent address. There are generally two sessions for each correspondent address (one Broadcast and the one Unicast). Using the Broadcast bit the correct session is selected and the Transport table entry is accessed.

If the transport table entry does not exist then it must be created. The Active bit is set, the Master bit reset and the sequenceNumber is set to one less than that in the current packet.

If the transport table entry already exists, then the packet's sequenceNumber is compared to that found in the Transport Table. If they are the same then the packet is a retry and the buffered response is re-sent. If the sequenceNumber is not one greater than that in the Transport table entry then the packet is discarded.

If the transport table is new or the sequenceNumber is one greater than that in the Transport table entry, then the TRANSMIT.indicate SP is invoked with the data field and the sequenceNumber set to the value in the Transport Byte and the transportType set accordingly.

9.2.4.5 Generation of the Response NPDU

Upon reception of a TRANSMIT.indicate containing a request packet the client layer must process the request and provide a response.

Un-Acknowledged Operation

Once the response is prepared, If the Acknowledged bit is reset then the client layer must invoke the TRANSMIT.request SP with the same sequenceNumber as in the request. Furthermore, the response must set the Broadcast flag identically to that in the TRANSMIT.indicate. Construction of the NPDU is completed and it is passed to the security sublayer for transmission and the TRANSMIT.confirm SP is invoked.

Acknowledged Operation

Once the response is prepared, the client layer must invoke the TRANSMIT.response SP with the same sequenceNumber as in the request. Furthermore, the response must set the Broadcast flag identically to that in the TRANSMIT.indicate.

Once the TRANSMIT.response SP is invoked, the correct Transport table entry must be found by locating the security session associated with the correspondent address. There are generally two sessions for each correspondent address (one Broadcast and the one Unicast). Using the Broadcast bit the correct security session is selected and the Transport table entry is accessed.

The sequenceNumber provided in the TRANSMIT.response is compared to that found in the Transport Table. If the sequenceNumber is not one greater than that in the Transport table entry then the SP fails and the packet is discarded.

Otherwise, the packet is buffered and the sequenceNumber in the Transport Table updated. The device will use the buffered payload to repeat replies in the event of duplicate requests. Next, the NPDU must be passed to the security sublayer for propagation to the transport pipe's master.

9.2.4.6 Master Collecting the Acknowledgement.

When the device receives a response NPDU it must inspect the Transport Byte.

Un-Acknowledged Operation

If the Acknowledged bit is reset then the TRANSMIT.indicate SP is invoked with the data field and the sequenceNumber set to the value in the Transport Byte. The Transport Layer transaction is complete.

Acknowledged Operation

However, if the Acknowledged bit is set then the sequence number must be validated. First, the correct Transport table entry must be found by locating the session associated with the correspondent address. There are generally two sessions for each correspondent address (one Broadcast and the one Unicast). Using the Broadcast bit the correct session is selected and the Transport table entry is accessed.

If the transport table entry does not exist then the packet is discarded.

If the transport table entry already exists, then the packet's sequenceNumber is compared to that found in the Transport Table. If they are the same then the packet completes the Transport Layer transaction. The TRANSMIT.confirm SP is invoked with the data field and the packetHandle set to the value from the TRANSMIT.request. TRANSMIT.confirm status is set appropriately.

Finally, the request packet buffer is released, the Active bit reset and the replyTimer disabled.

9.3 Wireless Network Layer Operation

9.3.1 Overview of Network Layer Behavior

A WirelessHART enabled device progresses through a series of states starting in the Idle state and continuing until the device is Operational and a full participant in the mesh network. There are 5 principal states and they are briefly described in the Table 13. These states are discussed in more detail in the following Subsections.

Table 13. Definitions of Network Layer States

State	Description
Idle	The device is quiescent and its wireless transceiver is not active. It has no knowledge of the WirelessHART network.
Joining	The device is listening for the network, attempting to acquire an advertisement and requesting admission to the network.
Quarantined	The device has successfully joined the network but only has a security clearance to talk with the Network Manager. It is not available or allowed to perform data acquisition or control functions or otherwise communicate with the Gateway.
Operational	The device can be accessed by Host Applications via the Gateway. It is integrated in the system's operation.
Suspended	The device is quiescent. All of its network tables are intact.

The state transition diagram is shown in Figure 32. This diagram depicts the events that allow the device to start-up, locate and join the network, and finally progress from being a new network member to becoming a fully operational network device. In addition, the diagram shows the device (and perhaps the network) being suspended.

Furthermore, the device can be forced to disconnect (e.g., to be removed from the process for depot-level maintenance) or to re-join at any time.

9.3.1.1 Idle

After a device rest or on power up the device should enter the Idle state. While in this state

- The device shall not attempt to communicate wirelessly; and
- Initial provisioning of the device should be perform using the maintenance port;
- The Join Key and Network ID are normally written to the device

The device may stay in this state until instructed to initiate the Join process.

9.3.1.2 Joining

Once initiated the Join process can have three outcomes: success, failure, or be aborted (e.g., by the reception of a Disconnect Command). To be successful the device must

- Locate the network;
- Capture an Advertise packet;
- Synchronize to the network;
- Request Admission to the network from the Network Manager; and
- Receive Network Keys and a Network Manager session; and
- Be provisioned with Superframe, links, graph and route.

Once one or more Advertise packets have been received the device must request admission using the Join Key. The Join request may be answered by the Network Manager immediately or the device may need to retry the request.

A failure shall occur if the device sends more join requests than allowed by maxJoinRetries without receiving the security keys and session parameters from the Network Manager. In this case, the device must return to the Idle state.

Once the keys and Network Manager session are established, the device must pend on reception of a Superframe, Route, Graph and Links from the Network Manager. Once this occurs, the device has joined and is quarantined.

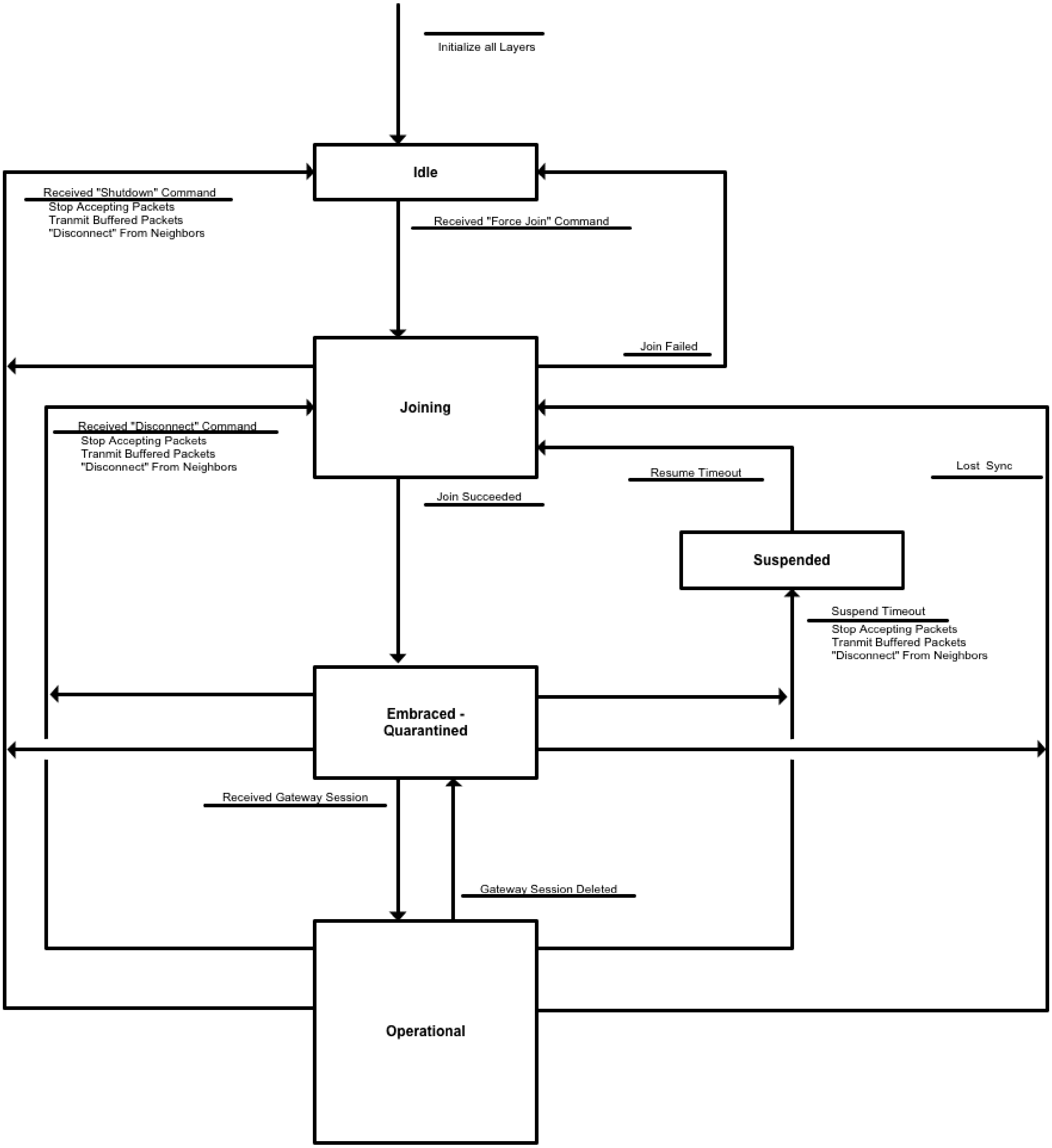


Figure 32. WirelessHART Network Layer Operation

9.3.1.3 Embraced-Quarantined

Quarantined devices are limited network partners. Devices should remain quarantined until approved for deployment into the network application. Approval may take the form of the operator authorizing the use of the device in his system and placing the device into service. This quarantine step parallels normal practices in the process industry and adds an additional layer of security (if desired) to the join process

While quarantined the device should only be enabled by the Network Manager to source and sink packets (i.e., the device must forward packets as configured by Network Manager irrespective of whether it is Quarantined or Operational). Normal network reports (e.g., neighbor reports) must be generated.

9.3.1.4 Operational

The device becomes operational upon the reception of the Gateway session parameters. This allows host application access to the device and begin its interaction with it.

Once the device enters the operational state it is a full network partner and must begin performing the mission designated by its configuration. Based on its configuration parameters the device must request bandwidth, for example, to periodically publish data. Request/response traffic with host applications is enabled.

Since the device is trusted full network partner, it may actively support operation and grooming of the network. This includes routing packets onward through the network to their final destination. The device also participates in advertising and supports new devices joining the network.

9.3.1.5 Suspended

When quarantined or operational the device may be suspended (along with the network). This places the device into a quiescent state for the time specified. This is normally a brief amount of time and done, for example, for safety purposes during mining blast operations.

When in this state the device continues to track elapsed time while leaving the radio disabled. Once the specified time has lapsed, the device progresses to the Re-syncing state.

9.3.2 Network Layer Data Model

All devices must maintain a series of tables that control the communications performed by the device, supply routing information, support end-end acknowledgements and ensure the privacy of the communications. The communication tables and the relationships between them are shown in Figure 33. Within the device the Session table is central and contains references to all correspondents the device can successfully communicate with. The tables controlling communication activities include:

- Session table. All communications center on the security session. A session record establishes a secure pipe between the device and a specific correspondent device. No communication to a correspondent device is possible without a security session and all devices must support multiple security sessions (see Subsection 9.1.3).
- Transport table. The Transport table is used to support end-end acknowledged transactions with automatic retries. There are generally two Transport records per session (see Subsection 9.2).
- Route table. The Route table serves as the locus for adding routing information to a new NPDU being generated as the result of a TRANSMIT.request.
- Source-Route table (optional). A Source-Route list is attached to some Routes. The Source-Route (when present) contains up to 8 device addresses tracing the Route (or a portion of it) from the device to the correspondent.
- Timetable. The Timetable indicates the Destination, Route and scheduling quota allocated by the Network Manager. The same Route may be used by more than one Timetable.

Note: Graphs and Neighbors are discussed in the *TDMA Data-Link Layer Specification*.

- The Graph table. Graphs are used to route messages from their source to their destination.
- The Neighbor table. The neighbor table is a list of all devices that the device may be able to communicate with. The device does not know the entire route rather there are references from the graph

to the neighbor that indicate the legal next hop Data-Link destinations. Neighbors with links to the device are listed first followed by detected (discovered) neighbors.

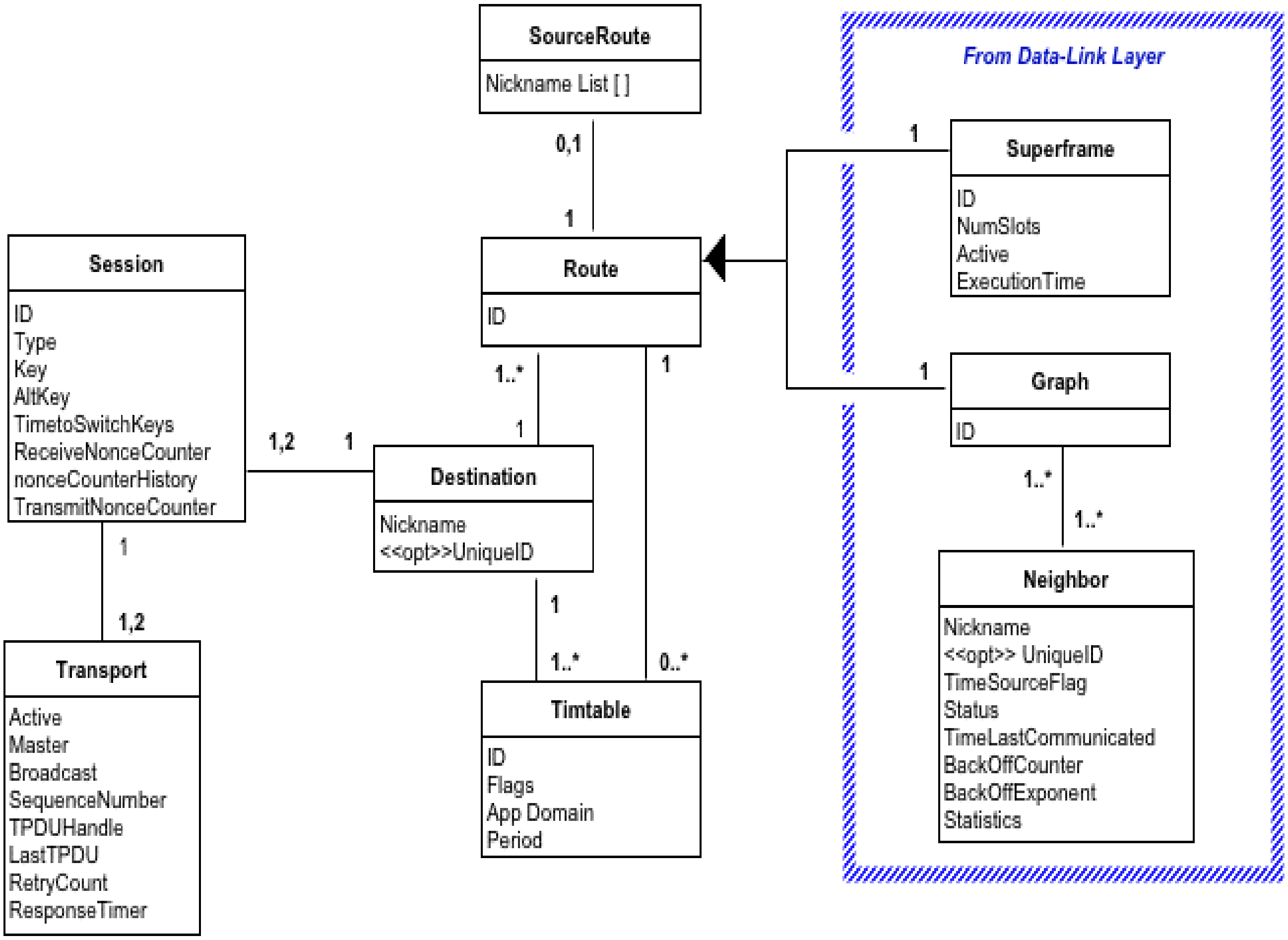


Figure 33. Wireless Network Table Relationships[14]

Field Devices must support the minimum number of record entries shown in Table 14.

[14] Although specific implementation of data and configuration storage is left up to the designer, the descriptions of the fields are critical understanding device requirements. Some fields described in the tables in this section may be calculated or derived from other information, and do not necessarily occupy space on the device.

Table 14. Minimum Table Space Requirement

Description	Minimum Required
Sessions	8
Correspondent Device	1 per Session
Transport	2 Per Session
Routes	8
Source-Routes	2
Timetables	16

9.3.2.1 Routes

Routes define one or more unidirectional path from one device to another. All routes (except those to the Network Manager[15]) have one or more Timetables associated with it. The Route/Timetable combination enable manages network congestion. Furthermore, multiple routes to a given destination allow the Network Manager to balance energy consumption across multiple battery-powered devices.

While a Route defines an entire path to the destination, only the Network Manager maintains a complete record of the Route. Field Devices only know the next hop along the Route (i.e., the Neighbors that are valid recipients for moving the PDU onward toward the destination).

Routes are only used when the device generates a new NPDU.

Table 15. Route Records

Content	Description
Unsigned-8 routeID	Route ID
Ref DestinationID	A reference to the destination of this Route.
Ref GraphID	A reference to the graph used to get packets to the destination. If null[16] then only a Source-Route exists to this destination.
Ref SourceRoute	A reference to the Source-Route used to get packets to the destination. If null then only a Graph route exists to this destination.

9.3.2.2 Timetables

When a device needs to establish a communication connection another device it requests Network Manager to allocate a Timetable for that communication. Each destination and Application domain (e.g., burst messaging, maintenance) has its own Timetable. Timetable records identify the Route to the destination device and the scheduling quota allocated by the Network Manager. The communication connection is control Timetable. Devices must not transmit more packets then allowed by the corresponding Timetable. When the device deletes a Timetable the Network Manager de-allocates the network resources (e.g., superframe and links) supporting the Timetable.

Table 16. Timetable Records

Content	Description
Unsigned-8 timetableID	Timetable ID
Ref DestinationID	A reference to the destination of this Timetable.
Bits-8 timetableFlags	Flags (e.g., source, sink, intermittent)
Enum-8 appDomain	The Application Domain associated with this Timetable
Time avePeriod	For cyclical communications (e.g., burst mode), packets are generated with this period. Network Managers must ensure latency for cyclical communication shall not exceed 1/3 the period. For intermittent communications specification this is the required latency.
Ref RouteID	The Network Manager returns the routeID when it allocates or modifies the Timetable. More than one Timetable can use the same routeID.

[15] The initial Network Manager Route is (initially) created from the Advertisement when the Field Device begins joining the network. Field Devices must assume the Network Manager has allocated sufficient bandwidth for all correspondence with the Network Manager.

[16] A null Graph ID is indicated by 0xFFFF.

9.3.3 NPDU Management

One of the core responsibilities of the Network Layer is the processing of NPDUs as they are received. There are three Network Layer clients that can source or sink an NPDU (see Figure 34):

- The Application Layer is a packet sink (signaled using the TRANSMIT.indicate SP) and a source (invoking the TRANSMIT.request SP)
- The Data-Link Layer that sources packets for consumption or routing by the Network Layer.
- A Joining Device serviced via the Data-Link. When Proxy routing the device acts as the NPDU destination on behalf of the Joining Device.

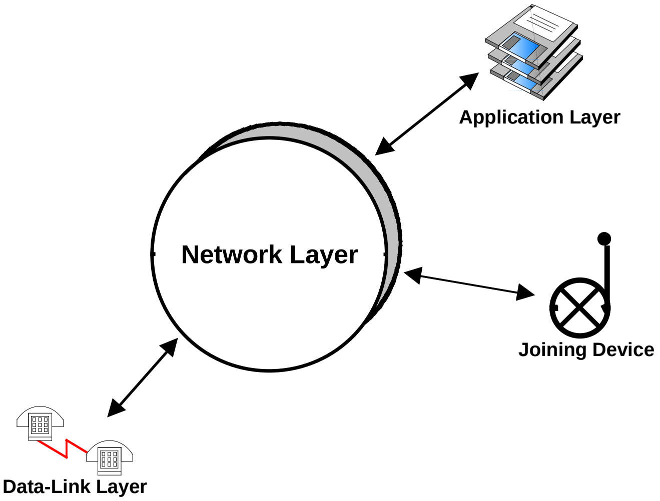


Figure 34. NPDU Clients

9.3.3.1 Processing Packets from the Application Layer

When a TRANSMIT.request or TRANSMIT.response is received from the Application Layer NPDU must be constructed. The field shall be set according the procedures indicated in Table 17.

Table 17. NPDU Construction

Field	Procedure
Destination	Set to value in Application Layer request (long, short, broadcast).
Source	Set to long address if Nickname uninitialized.
ASN Snippet	Set to current ASN value
TTL	Normally set to default value.
Graph ID	Set based on results of route selection
Control	Set the address bits based on source and destination address sizes. Destination addresses are indicated in the TRANSMIT.request. Source address shall be the device's EUI-64 address until the device's Nickname is initialized. Set the routing bits based on results of route selection
Proxy Route	Set based on results of route selection
Source Route(s)	Set based on results of route selection

The Route (and Graph ID) must be extracted from the Timetable[17]. In addition, all packets from the Application Layer have a default route based on the TRANSMIT.request fields which must be chosen as indicated in Table 18. Once the route has been identified the Graph ID is copied into the NPDU.

Except when testing routes, packets are Graph routed. However, if a Source-Route is attached then the source route addresses must be included in the NPDU. Unused Source-Route address entries in the NPDU must be set to 0xFFFF. If source routing is used the Graph ID should also be valid[18].

Table 18. Default Route Based on Priority and Transport Type

TRANSMIT.request Field		Resulting Application Domain[19]	Default Route Type
Priority	Transport		
Command	Don't Care	Maintenance	First "Maintenance" route to the Network Manager
Data	Don't Care	Publish	First "Publish" Route to the Gateway
Normal	Transfer	Block Transfer	First "Block Transfer " Route as indicated by destAddress
	! Transfer	Maintenance	First "Maintenance " Route as indicated by destAddress
Alarm	Don't Care	Event	First "Event " Route to the Gateway or Network Manager as indicated by destAddress

In the case of TRANSMIT.response the Application Domain from the Timetable shall be used to set the Priority (e.g., Publish Application Domain sets priority to "Data"). If the Timetable does not exist then the priority must be set based on the destination as follows: Network Manager sets priority to "Command"; Gateway sets priority to "Normal"; Peer-Peer communication defaults to priority "Normal".

9.3.3.2 Processing Packets Received from the Data-Link

The Network Layer will be passed NPDUs as the Data-Link Layer receives them. The Network Layer must route these packets to the correct destination: back to the Data-Link Layer, onward to the Joining device, or up to the Application Layer.

Packets Addressed to the Device

If the device is the NPDU's final destination then the NPDU is authenticated and deciphered, discarding if necessary (see Subsection 9.1.3). The Transport Layer must be invoked (see Subsection 9.2) which, in turn, forwards the payload to the Application Layer.

Packets with a Broadcast address

When the NPDU is received and the destination address is the broadcast address then the NPDU is authenticated and deciphered, discarding if necessary (see Subsection 9.1.3). The Transport Layer must be invoked (see Subsection 9.2) which, in turn, forwards the payload to the Application Layer. In addition, the NPDU must be forwarded as described the next subsection.

An NPDU (e.g., Command 961) received as part of a Request-Broadcast (see Subsection 7.1) may result in responses from a large number of devices. For these transactions, a random back-off time is chosen to delay the device response and minimize the flood of simultaneous responses. The back-off time is a random value chosen between zero and BcastReplyTime.

An NPDU (e.g., Command 21) received as part of a Search-Broadcast will generate a single, unique response and the response must be generated immediately.

[17] If the Route ID referenced by the Timetable does not exist then the default Route from Table 18 must be used.

[18] Graph ID must be set to 0xFFFF in the NPDU if only a Source-Route is used.

[19] If a route matching the Application Domain is not available then the first route to the destAddress must be used.

Forwarding Received Packets

Prior forwarding an NPDU the device must check and update the TTL. If the TTL count is exhausted the NPDU must be discarded. Next the ASN Snippet is inspected and, if the maxPacketAge is exceeded the packet must be discarded. While inspecting the ASN Snippet, the timeout value to be passed to the Data-Link must be calculated.

Upon confirming the packet does not need to be retired, its destination address must be inspected and routed as specified in Table 20. When analyzing a received NPDU the fields in the Table 20 must be evaluated from left to right (i.e., starting with whether the NPDU contains Source-Route information). A "No" in the Graph column means that the Graph ID is invalid (i.e., the Graph ID is 0xFFFF). The combinations NOT depicted are illegal and result in the NPDU being discarded[20]. The "Action" column indicates the routing action to be performed based on the state indicated in the first four columns. This results in either a specific address being provided the Data-Link for the next hop or for the Data-Link to broadcast the NPDU.

Furthermore, Table 19 specifies (in order) fundamental routing requirements and supplements the specifications in Table 20. When routing an NPDU the bullets must be evaluated in the order listed.

Table 19. Ordered, Fundamental Routing Requirements

1	If the NPDU contains a proxy address that matches the device's then final destination is a joining device and it is the device's responsibility to deliver the NPDU. The device must route the packet to the joining device by passing the NPDU back to the Data-Link indicating the Data-Link destination address is the joining device's.
2	If NPDU is Source-Routed and the device's Nickname is in the Source Route list then the NPDU must be transmitted to the neighbor with the nickname that follows the device's nickname. If the device's nickname is the last on the source route then the source routing is not used to forward the NPDU.
3	If Source-Routing is not possible and the Unicast NPDU destination address matches a neighbor[21] and a transmit link to that neighbor exists then the NPDU must be routed directly to the neighbor. In this case, the packet is passed to the Data-Link for propagation directly to the final destination. If this fails (i.e., an error is returned from the Data-Link[22]) then the packet must be Graph-Routed (if possible).
4	Otherwise, the packet must route onward based on the Graph ID contained in the NPDU.

Two errors are possible: the NPDU reaches the end of a Source-Route without reaching its final destination; and the NPDU reaches the end of a graph route without reaching its final destination. These errors can only happen when the final destination is Unicast and the packet was not being broadcast across the Data-Link.

[20] For example, a Broadcast DLPDU without a Superframe ID in the NPDU Graph ID field is illegal and the NPDU must be discarded.

[21] A Proxy device substitutes for the final destination. Consequently, when Proxy Address is included in NPDU then Proxy address is the Data-Link neighbor destination.

[22] Data-Link should retry it transmission in this case at least twice.

Table 20. Routing of Forwarded Packets

Routing		Destination Address		
Source	**Graph**	**DLPDU[23]**	**NPDU**	**Action**
Yes	Yes	Unicast	Unicast	Forward the NPDU to (1) the next address in the Source-Route[24]; or, (2) if source route exhausted, forward along the Graph route (also see 3 and 4 in Table 19). If both the Graph and Source routes exhausted, then signal a Source-Route error.
			Broadcast	Forward the NPDU to (1) the next address in the Source-Route[24]; or (2) if source route exhausted, forward to all neighbors on the Graph (NPDU is discarded at the end of the Graph)
		Broadcast	Don't care	Graph ID must be Superframe ID. Forward the NPDU to (1) the next address in the Source-Route[24]; or (2) if source route exhausted, forward the NPDU using a broadcast link in the Superframe (NPDU is discarded at the end of the Graph).
	No	Don't care	Unicast	Forward the NPDU to (1) the next address in the Source-Route[24]; or (2) if source route exhausted signal a source route error.
			Broadcast	Forward the NPDU to the next address in the Source-Route; (NPDU is discarded at the end of the Source-Route).
No	Yes	Broadcast	Don't care	Graph ID must be Superframe ID. Continue broadcasting the NPDU using broadcast link in the Superframe. NPDU discarded at end of Graph.
		Unicast	Unicast	Forward NPDU along the Graph using any normal link to neighbor on the Graph (also see 3 and 4 in Table 19). If end of Graph signal graph route error.
			Broadcast	Forward NPDU to all neighbors on the graph. NPDU discarded at end of Graph.

[23] When DLPDU is broadcast then the Graph ID holds the Superframe ID used to identify the corresponding broadcast links.

[24] The next address in the Source-Route must be the first one of either (a) the address listed immediately after the device's address is in the Source-Route list (if there is one); or (b) the first address in the Source-Route list (if it is a neighbor). Failing both of these two conditions the source route is exhausted.

9.4 WirelessHART Procedures

9.4.1 Initializing a WirelessHART Network

Prior to forming the network, the Network Manager must be provisioned with the Network ID. Using this ID and a supply of security keys network formation can be initiated. This begins with the Network Manager creating a secure a private connection with the Gateway. As part of its initialization sequence, the Network Manager will download to each of the Gateway's Access Points:

- The network management superframe supporting base bandwidth require to monitor and service the network;
- The network graph supporting upstream traffic to the Network Manager;
- The Join superframe and Join Links allowing new devices to join the network; and
- Dedicated and shared Links (both transmit and receive) supporting management of devices, the transport of health reports, and communication of alarms (e.g., path down).

In general, the Gateway's Access Points are configured to be active in every slot. This maximizes the advertising, network management, and join packets available to the network. Once the Network Manager enables the first superframe, ASN 0 is established (i.e. the network is born). Once the Gateway's Access Points begin transmitting Advertise packets, devices can join the network and the network begins forming.

There are three components of network formation: *advertising*, *joining*, and *parameter negotiation*. As part of advertising, Network Devices that are already part of the network may send packets announcing the presence of the network. Advertise packets include the Network ID, ASN, join frames and join links. Devices that are trying to join the network listen for these packets. Once an Advertise packet for their network is heard, the new device can attempt to join the network. The join sequence is described in Section 9.4.2.

As the device joins the network, both the device and the Gateway requests bandwidth from the Network Manager. For example, the device asks for bandwidth to publish process data and the Gateway requests bandwidth to support request/response traffic. The Network Manager uses these requests to gauge and manage the available bandwidth. Assuming there is sufficient bandwidth, the newly joined device is allocated superframes and links according to the requests. If bandwidth becomes constrained then the Network Manager may reduce the bandwidth requested or refuse to allocate any at all.

9.4.2 Joining

"Joining" refers to the process used by device to obtain access to the network and to become integrated into it. The key steps in the joining process include:

- Periodic Advertise packets by existing network members to allow the network to be identified. The Advertise packet also includes sufficient information for the new device to synchronize to the network and communicate on the correct link (i.e., on the right slot number and channel).
- Monitoring by the new device to locate and synchronize to the network.
- Establishing a secure channel between the new device and the Network Manager. This is done using the Join Key to encipher the initial communications between the two devices. The Join Key should be different for every device.
- Verifying the trustworthiness of the new device. This is done several ways: The device's Identity (Command 0) and Long Tag (Command 20) are presented to Network Manager. Furthermore, the Join Key is, in effect, a password. All of three of these items should be used when considering allowing the device into the network. Furthermore, Network Managers should implement a whitelist (see Commands 815 and 816) to ensure only known, trusted field devices join the network.
- Once the device is deemed trustworthy, it can be provisioned and allowed into the network. Initially the device can be quarantined until the plant operations are ready to begin utilizing the device.

In this Subsection an overview of the Join process is provided followed by detailed Network and Data-Link Layer join requirements.

9.4.2.1 Overview of the Join Sequence

An overview of the Join Sequence is shown in Figure 35. For a new device to become operational it must be provisioned with Network ID; locate and synchronize with the network; petition the Network Manager for access; obtain session keys; and gain the bandwidth necessary to meet the obligations the device's configuration has imposed. The general progression that must be followed for the joining device to become operational includes:

- Initial device provisioning consists of obtaining the Network ID and Join Key.
- The device must begin listening for network traffic to allow it to synchronize to the network clock and identify potential parents.
- Next, the device presents its credentials to the Network Manager to demonstrate the device is trustworthy. The credentials include the device's identity and Join Key and, if these credentials are valid, the device is admitted to the network.
- Once the Network Manager has scrutinized the device's credentials and deems the device trustworthy, the Network Manager provides the first keys (Network Manager Session Key and the Network Key) to the joining device.
- Once security requirements for new devices have been met the Network Manager proceeds to integrate the device into the network. This is accomplished by provisioning the device with normal superframes and links.
- The Network Manager may choose to leave the device Quarantined. In this case the device can participate in the network but does not have a Gateway session.
- Once the quarantined device obtains a session with the Gateway it becomes Operational. It then begins acquiring the bandwidth and communication resources required to publish process data and events as dictated by its configuration.

Each of these steps are discussed in detail in the following paragraphs.

Initial Device Provisioning

Prior to attempting to join the network, devices require two pieces of data: the Join Key[25] and the Network ID. The Network ID identifies the network the device is to Join and the Join Key is the network password that will allow the device to join the network. These two items should be written to the device using a standard HART-enabled maintenance tool (e.g., a Generic Host or a Universal Host, see *Command Summary Specification*) via the device's maintenance port. The maintenance port must comply with the *Token-Passing Data-Link Layer Specification*.

Once initial provisioning is complete the device can be immediately configured to attempt joining the network. Alternatively, the device can be mounted in the process and then the end user can use a HART-enabled maintenance tool to place the device into join mode.

The maintenance tool may also be used to monitor the join process allowing the operator to intervene (if necessary).

Listening for Network Traffic

Once the join has been initiated, the device's Data-Link is placed into search mode to synchronize to the network and receive Advertise packets. While in this mode, the device will identify advertising neighbors and gather neighbor statistics (e.g., average signal level).

Once one or more advertising neighbors have been identified the joining device selects one of them to join through. The first join attempt shall be made via the neighbor demonstrating adequate Receive Signal Level and indicating the lowest Join Priority. The NPDU Graph ID must be set to the same value contained in the Advertisement DLPDU received from the neighbor being joined through.

[25] When displayed to or entered by the end user, the Join Key shall be displayed/edited in Hexadecimal Format

Presenting Credentials

Once synchronized to the network the device generates a Join Request (see ② in Figure 35) and sends it to the Network Manager. The Join Request and its encipherment contain the device's credentials for the Network Manager to inspect prior to allowing the device to join the network. The device's objective is to obtain a Session Key (for further communication with the Network Manager) and a Network Key (for Data-Link Layer device-to-device, one hop security and authentication). There are three key credentials:

- The Device's Identity (Command 0 response)
- Its Long Tag (Command 20 response); and
- The list of Neighbors detected by the device (Command 787 response).
- The device's Join Key (i.e., the device's password to the network).

The first three, in the order listed, must be included in the join request's payload. The Command 787[26] response should contain 2 neighbors: the neighbor being joined through and one other neighbor. The second neighbor shall be the advertising neighbor not being joined through with the highest RSL. The device must listen on all receive links advertised by the two neighbors sent in Command 787.

The Join Request must be transmitted at "Normal" priority.

The Join Key is used as the session key for all communications until the device receives Session and Network Keys from the Network Manager. Consequently, successful authentication by the Network Manager confirms the device has the correct Join Key.

Since the joining device does not know the Network Key, (i.e., the Data-Link key) the well-known key (see the *TDMA Data-Link Layer Specification*) is used for communicating the DLPDUs with the advertising neighbor (i.e., the device's prospective parent). Once the prospective parent receives the packet it is forwarded through the mesh using the Network Key the same as any other DLPDU.

[26] Normally, unsolicited Command 787 response packets from devices to the Network Manager contain unlinked neighbors. However, in the join request Command 787 contains neighbors with join-links specified in the advertisement received by the joining device.

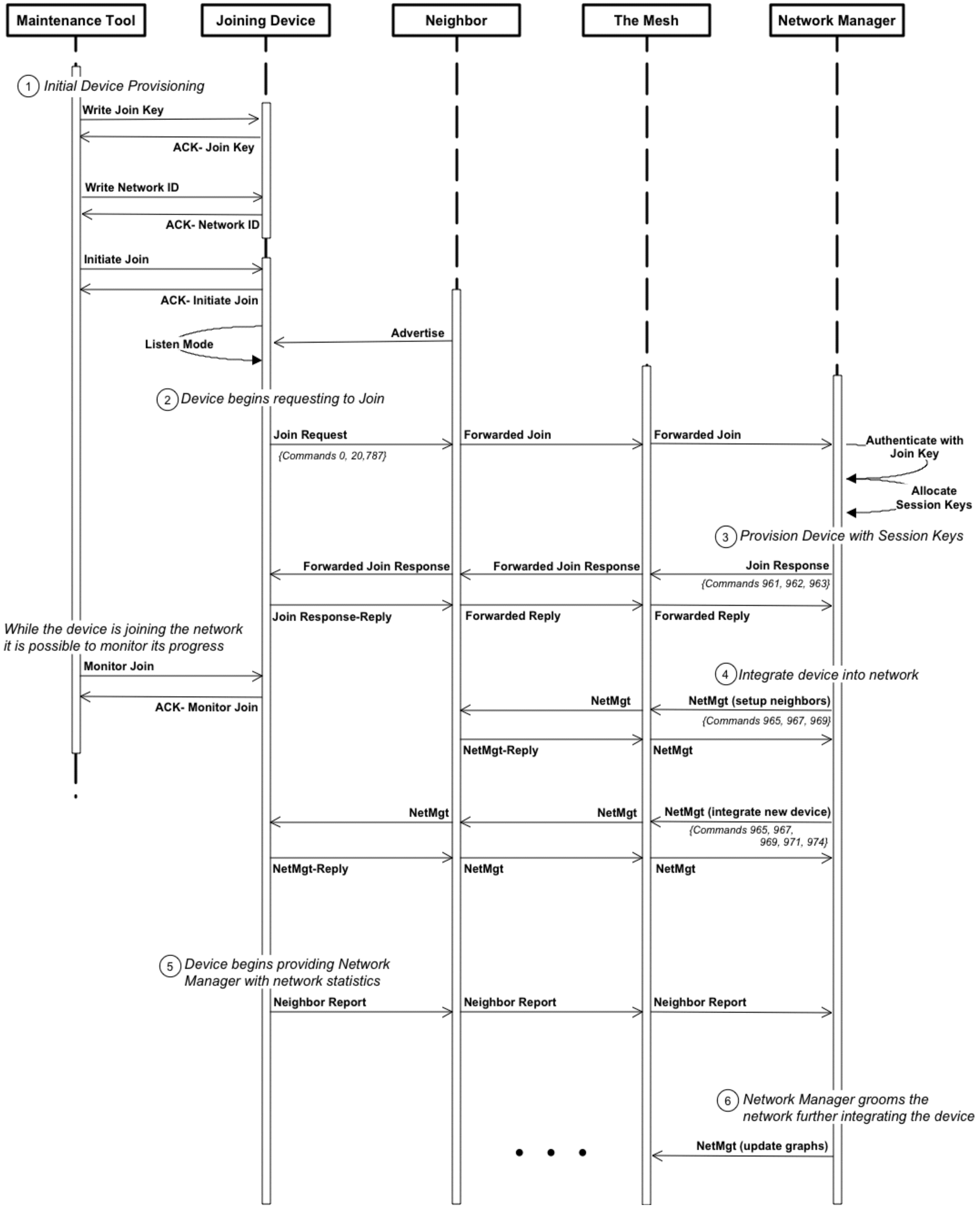


Figure 35. Join Sequence

Once the join request is sent, the device starts a response timer in the same fashion as used in the Transport Layer. When this timer lapses another join request is generated to the next available advertising neighbor. Join requests are sent until maxJoinRetries is exceeded.

Getting the First Keys

When the Network Manager receives a Join Request it must confirm the request was from a trusted device. A trusted device:

- Has a trusted identity (i.e., the right device name);
- Is using an appropriate Join Key (the right password); and
- Properly combines the device name and password with each other.

The Join Request has sufficient information to satisfy all of these criteria. Identity is established using the binary information in the Command 0 or using the Long Tag in Command 20.

There are a wide range of methodologies and security techniques to establish whether a joining device is to be trusted. Selection of the strategy employed by Network Managers to confirm the join device is a trusted device is beyond the scope of this specification. However, WirelessHART does support several strategies that directly may be used by a Network Manager:

- If a device Whitelist is supported then only devices on that Whitelist shall be allowed to join the network.
- If a device Blacklist is supported then devices on that Blacklist shall NOT be allowed to join the network.

In addition, the Network Manager should always check the active device list to ensure a duplicate device (e.g., duplicate Unique ID) is not trying to spoof its way into the network.

If the device's credentials are in order[27] then the join request authenticates, the Network Manager allocates Session Keys, the device's Nickname (i.e., the 2-byte short address) and writes these along with the Network Key back to the joining device using:

- Command 961 Write Network Key (truncated);
- Command 962 Write Device Nickname Address; and
- Command 963 Write Session (truncated).

Like the Join Request the Join Response NPDU is enciphered using the Join Key. The Join Response NPDU containing Commands 961, 962, and 963[28] should be routed to the joining device via its prospective parent[29].

The device replies to the write commands received from the Network Manager. The Join Response-Reply packet uses the new Session and Network Keys and the newly-created Network Manager Unicast Transport Layer pipe.

When the Join Response-Reply is transmitted the device starts a response timer in the same fashion as used in the Transport Layer. When this timer lapses another join request is generated to the next available advertising neighbor.

[27] If the credentials are NOT in order then no join response shall be generated and the device information must be added to the Rejected Device List (if supported). The Rejected Device List must be 5 entries long or 15%the length of the Network List, whichever is greater.

[28] While all three of these commands must be in the join response they may be embedded in the join response NPDU in any order.

[29] While the prospective parent will likely be one of the two neighbors reported in the join request it may not be the same device first forwarding the join request.

Device Integration into the Network

Now, the device has keys and a network ID and is, albeit awkwardly, able to communicate with the Network Manager. At this point in the Join sequence, its communication requires the Network Manager to use proxy routing to reach the joining device.[30] The next step is for the Network Manager to integrate the joining device more tightly into the network. This includes:

- Providing the device with at least two time source neighbors[31] (Command 971);
- Specifying the Graph (Command 969) used in the Network Manager Route (Command 974);
- Updating the communication tables in the device's neighbors; and
- Transferring the device's communication from join links to normal links (Command 965 and Command 967).

Examples of the associated communication activity are shown starting at ④ in Figure 35. This communication activity will continue for some time and result in the Network Manager writing a series of graphs, routes, superframes and links to the joining device. After this, the device is integrated into the network and no longer may use join links for communication. After receiving Command 971 and Command 974, the device must reject any packets using the Join Session[32].

The device has now "Joined" the network.

Quarantine

When a device is a member of the network and has no session with the Gateway, the device is "Quarantined". While quarantined the device can only communicate with the Network Manager and shall not publish process data. The device shall operate normally, conveying DLPDUs received from neighbors, generating neighbor reports and network statistics. The Network Manager can modify the network communication configuration as needed to groom the network.

As soon as the device enters the Quarantine state, it begins generating health reports (see ⑤ in Figure 35). Upon entering this state, the HealthReport timer is initialized to HealthReportTime (see Table 5) and the timer is enabled. When a device joins the network the Network Manager is responsible for allocating sufficient bandwidth for network management communication traffic. Management communication traffic does not have a Timetable and, consequently, not constrained. The Network Manager must write the Route to be used for network management communication traffic (using Command 974).

The device generates health reports whenever the HealthReport timer lapses[33]. Health reports are transmitted at the "Command" priority level. Following every health report the HealthReport timer is reset to HealthReportTime.

Health reports consist of the response PDUs for Command 779, 780, and 787. Using Command 780, statistics on all linked neighbors are returned. Command 787 reports on all detected neighbors that do not have links to the device. Depending on the network and number of devices, multiple packets are normally generated. Health reports are aggregated into as few NPDUs as possible.

Depending on the Network Manager's security strategy it may immediately write the Gateway session to the quarantined device or leave the device quarantined for a time. In any case, the device will remain quarantined until it receives a session allowing communication to the Gateway.

[30] The Network Manager shall not delete or suspend any join links while a device is in the process of joining.

[31] In some cases a joining device may only have a single neighbor.

[32] The Network Manager must configure the Graph, Superframe and Links before creating the Network Manager Route and identifying the time-source neighbor.

[33] After joining the network, the first health report must be generated after identifying (at least) three neighbors or when the HealthReport timer expires, whichever comes first. If multiple Advertise packets were captured the first report could be generated immediately after entering the Quarantine state.

Becoming Operational

When a new Gateway session is written to the device it becomes operational. Once this session is operational, a Gateway will normally begin filling the Gateway's data cache for device. This will result in a surge in request/response traffic to the device and may be several hundred transactions. This communication uses the Maintenance Timetable the Gateway received from the Network Manager. The Maintenance Timetable is owned and controlled by the Gateway. In turn, the Network Manager must write this Timetable to the device (using Command 973) to establish the route and communication bandwidth used for device responses.

Upon receiving a session to a new device, the Gateway should request significant bandwidth (e.g., one request/response per second) to the device while it fills its cache then lower it to a normal rate (e.g., one request/response per 10 seconds).

In addition, external host applications (e.g., instrument management packages, process automation controllers, etc.) may begin communicating with the device. The Gateway should increase and decrease the Maintenance bandwidth guaranteeing responsive communications with the field device. This communications rate should be similar to that found in FSK-based communications.

Once the device receives its Gateway session, it must obtain enough bandwidth and communication resources to meet its responsibilities to the process automation system as a whole. Once operational the device begins requesting bandwidth for the process data (the Publish Timetable) and alarms (the Event Timetable) it must publish.

Once operational the Network Manager will continue adding or refining superframes and links used by the (now operational) device. Although the Network Manager will continuously adjust communication resources on its own in response to changing communication requirements, normally the Gateway or operational device will trigger Network Manager activity itself by requesting resources from the Network Manager. This is covered in a separate procedure.

9.4.2.2 The Network Layer Join Process

The join process is managed from the Network Layer using two cascaded state machines. The Network Layer state machine enforces the high-level join procedure (including retries for security purposes) while the Data-Link state machine is focused on synchronizing the device to the communication slot times (see the "Data-Link Join Process" Subsection). The primary objective of the Join Process (at the Network Layer) is to receive admission to the network and obtain a Superframe (with links) allowing the device to communicate reliably with the Network Manager.

Upon beginning the process, the Data-Link is signaled to enter its join process and begin actively searching for the network.

Searching

While in the searching mode the Network Layer waits for reception of an Advertise packet. While in this mode the Data-Link (see the "Data-Link Join Process" Subsection) is searching for the network, synchronizing to it and receiving DLPDUs.

Once the ADVERTISE.indicate signal is received from the Data-Link the Network Layer sequences to the "Got an Advertising Neighbor" state.

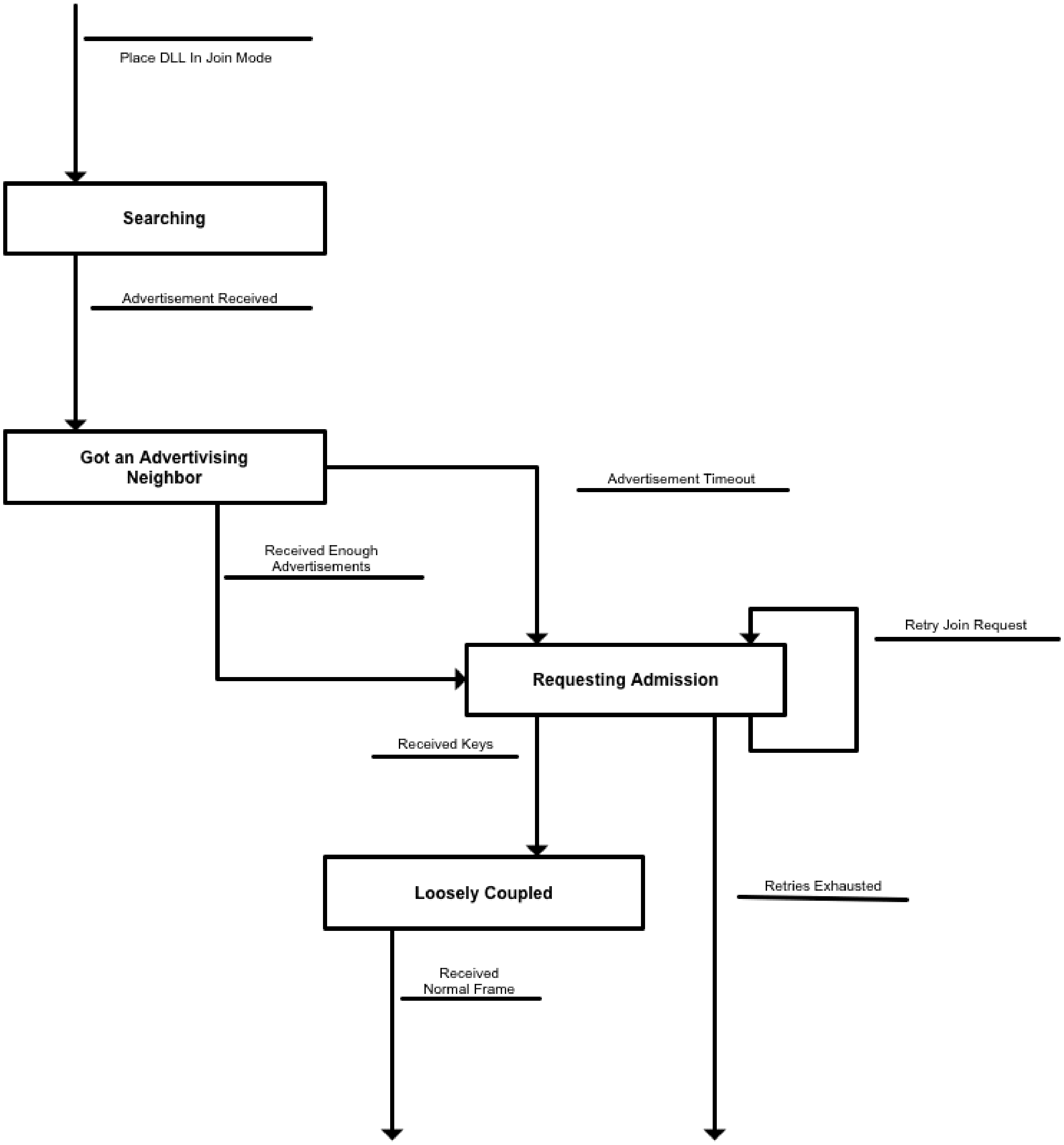


Figure 36. Network Layer Join Procedure

Got an Advertising Neighbor

Upon entering the "Got an Advertising Neighbor" the AdWaitTimer is initialized to AdWaitTimeout (see Table 5) and started. The device continues to wait for additional Advertise packets to be received. When the desired number of different Advertise packets has been received or the AdWaitTimer times out the device moves to the "Requesting Admission" state.

Requesting Admission

Upon entry into the "Requesting Admission" state the device must send a Join Request to the Network Manager, initializes the JoinRspTimer to JoinRspTimeout (see Table 5), and starts the JoinRspTimer. If the Network Manager responds by writing the Network key, the Network Manager session, and the device's Nickname, the device progresses to the "Loosely Coupled" state.

Otherwise, upon JoinRsp timeout the join request is issued again and the JoinRetry counter is decremented (see Table 5). Join Requests continue to be transmitted until the retries are exhausted or a response from the Network Manager admits the device into the network.

If the retries become exhausted, the state machine is terminated and exits with an error.

Loosely Coupled

Once the device has a Nickname and keys, it is able to communicate with the Network Manager. However, the device's connection to the network is tenuous at best (it only can talk via the shared join links). The device must stay in the "Loosely Coupled" state until it receives a Superframe and Links from the Network Manager. If the device gets a Superframe and links then it signals the Data-Link that joining is successful and exits the Network Layer join process successfully. Reliable communications between the Network Manager and the device is now ensured.

9.4.2.3 Data-Link Join Process

Note: While this could be included in the *TDMA Data-Link Layer Specification*, the join process is a device-level process and requires close coordination between the Network and Data-Link Layers. Consequently, it is located in this specification for clarity.

The Data-Link is triggered to enter the network search mode by the Network Layer. The objective of this mode is to synchronize the device's slot timing to that of the network. The first step in synchronizing is to begin actively searching for the network.

Active Search

While in the "Active Search" state the device must leave its transceiver on in receive mode while continuously listening for packets. When the Data-Link enters this state it sets a timer to ActiveSearchShedTime to bound the network active search time. If this time lapses without identifying the network, the device transitions to the "Passive Search" state.

The device's transceiver remains on a channel for the interval indicated by the ChannelSearchTime value. When that time period lapses the device must switch to the next channel and resume listening. Channels are scanned sequentially until ActiveSearchShedTime lapses or a packet is received.

When a packet is received, the device sequences to the "Packet Received" state.

Passive Search

The Passive Search state is a reduced cycle, power saving mode of operation entered when the network cannot be located. In this mode the device continues to prospect for network packet, just at a slower rate. The Device wakes up and listens for the interval indicated by the PassiveWakeTime. If no packet is received, the device turns off its transceiver and returns to low-power mode. This cycle repeats as specified by the PassiveCycleTime.

If a packet is received, the device transitions to the "Packet Received". The device can also be forced back into the "Active Search" state by the reception of the "Force Join Mode" command.

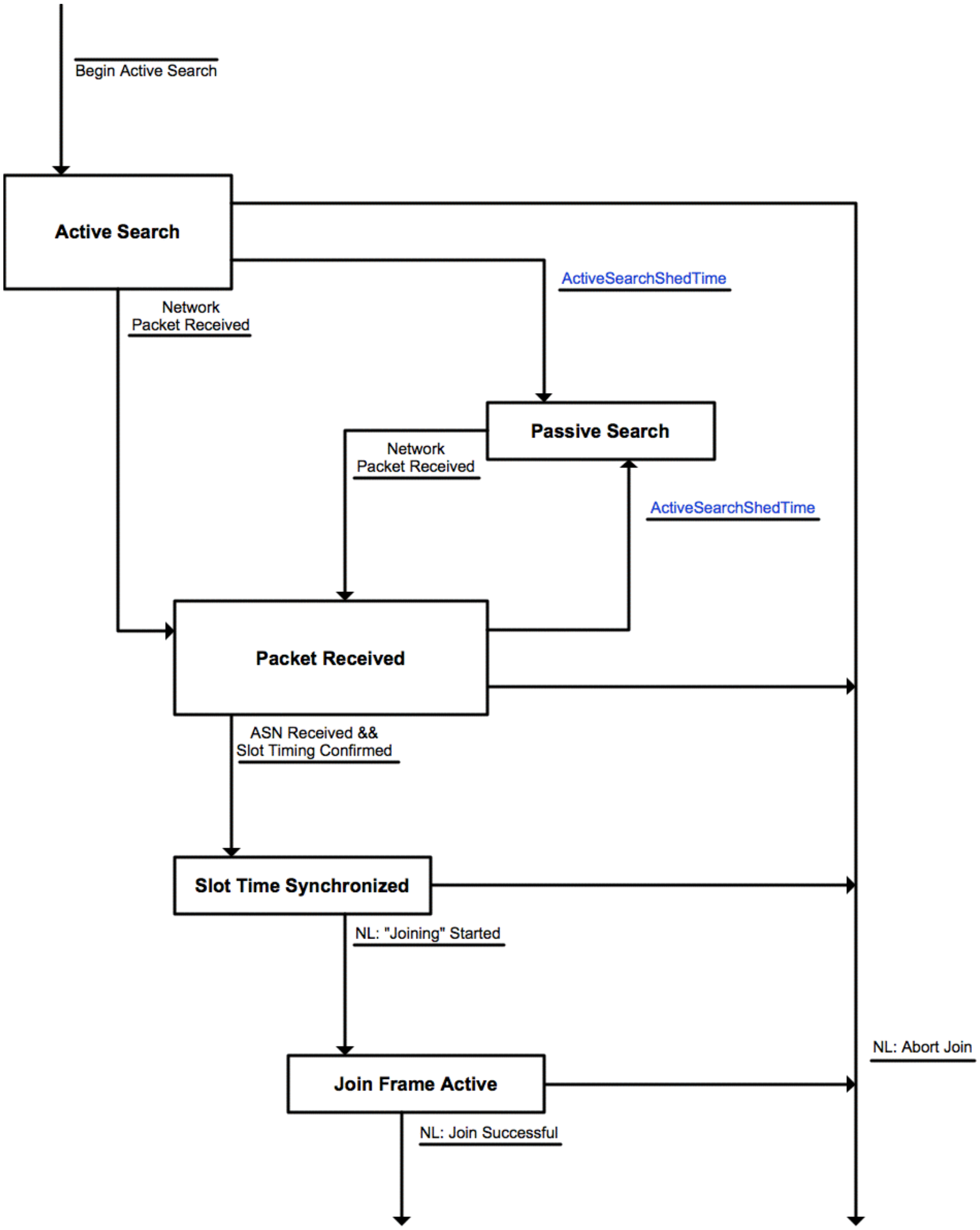


Figure 37. Data-Link Layer Network Search Procedure

Packet Received

The objective of this state is to synchronize the device's slot timer to the network. This state[34] is entered when a network packet is received (i.e., a packet with the correct Network ID).

If the DLPDU is not an ACK, then the start time of that packet is recorded and the device's slot time is established. Subsequently, the start time of all additional non-ACK packets are compared to the devices slot timing and statistics are compiled measuring the device's synchronization to the network slot time. The device is considered synchronized when the slot timing statistics converge.

While in this state the device continues to capture network packets. As it does so, it must update its neighbor table as normal. In addition, channels continue to be changed as indicated by the ChannelSearchTime value.

[34] Upon entering this state the ActiveSearchShedTime is restarted. If this time lapses without identifying the network, the device transitions to the "Passive Search" state.

When an Advertise packet is received the tables in the device are updated including;

- The superframe and links are created or updated as needed;
- Graphs are created or updated using the Graph ID in the Advertisement DLPDU;
- Graph edge is created to the neighbor generating the advertisement; and

Each unique Graph ID received via an Advertisement PDU should internally create a Route to the Network Manager with IDs in the range 0xF0-0xF9.

In addition, the channels searched shall be limited to those indicated in the Advertise packet's Channel Map.

This device transitions to the "Slot Time Synchronized" state when (1) the slot time is synchronized, (2) the Absolute Slot Time has been aligned to the network When these two criteria are met the Data-Link transitions to the "Slot Time Synchronized" state.

Slot Time Synchronized
Once this state is entered the Data-Link can reduce its listening to slot times. As packets are received, the device's timers are updated keeping it in sync with the network.

At this point the Data-Link is pending on the Network Layer to generate and propagate a join request. However, before the Network Layer will generate a Join request an Advertise packet must be received.

When an Advertise packet is received, the Network Layer is signaled using the ADVERTISE.indicate SP. The device must configure the Join superframe(s) and links as indicated in the Advertise packets it receives. These frames remain disabled until join requests are generated. In addition, the Graph indicated in the Advertise packet is initialized and the connection to the advertising device created.

When a join request is received from the Network Layer, the device enables the join superframe(s) and transitions to the "Join Superframe Active" state

Join Superframes Active
The first join attempt shall be made via the neighbor demonstrating adequate Receive Signal Level and indicating the lowest Join Priority. Once the join request is propagated the device shall begin issuing Keep-Alive packets, as needed, to maintain synchronization with Neighbors connected via Join Links. The device stays in this state until the Network Layer signals that the Join was successful or the join is aborted.

Before the Join Request is generated, the Join Superframe received via the Advertise packets are enabled. Joined frames contain shared slots. Consequently, collisions with other joining devices are probable. To minimize collisions resulting from many devices trying to join simultaneously the Back-Off Exponent (BOExp) must be initialized to four (4) prior to transmitting the first join request (see *TDMA Data-Link Layer Specification*) and the Back-Off Counter (BOCntr) calculated. This results in the slot being used for transmitting the Join Request being randomized.

Once the Join is successful the Join Superframes, Graphs, Routes and Links must be deleted.

9.4.3 Device leaving the network

In the network operation there will be times when a device is either suspended or disconnected from the network. If this occurs, the device must go through a complete re-join sequence. Prior to departing the network the device should not accept any additional packets, empty its packet buffers (if possible) and it must send a disconnect PDU to all of its linked neighbors.

9.4.4 Neighbor Discovery

The neighbor discovery process is used to learn of potential connectivity to devices in the network. The device maintains a list of discovered devices at the end of its Neighbor table (i.e., following the neighbors with links to the device). Network Devices periodically report neighbor information in their Health Reports using Commands 780 and 787. The Network Manager uses the information in the Health report to adjust the overall network graph and in some cases, adjust the schedule.

Neighbor Discovery is accomplished by three means: new neighbors joining via the device; communications between other devices in the network that are overheard; and the special discovery process. In other words, devices must continuously[35] listen for communications from their neighbors and for communications from new neighbors. Continuous monitoring of neighbors and the discovery of new neighbors is critical to the maintenance of the mesh and the enhancement of communications reliability.

The Discovery process consists of discovery links shared by all devices in the network. Devices must listen on each Discovery link and, occasionally, must transmit on one of the Discovery links. This Discovery process directly promotes the discovery of new neighbors. See the *TDMA Data-Link Layer Specification* for more information.

[35] When not in conflict with a pending transmit, devices must service all receive links. When servicing a receive link the device may receive a packet addressed to the device or a packet addressed to another device. Packets addressed to the device are processed as usual. Authenticated packets not addressed to the device should be used to update the discovered neighbors. Discovery Keep-Alive packets must be used to update the discovered neighbors list.

9.4.5 Path Failure

Path failures are reported to the Network Manager when devices lose connectivity to neighbors. Figure 38 depicts three devices and shows that Device A and Device B have been successfully communicating (e.g., A published via B). However, interference or some blockage disrupts communication between A and B. Of course, the mesh allows A to continue responding via an alternate route (e.g., via Device C). Since communication was lost, A and B begin transmitting Keep-Alive packets to probe the connection. Furthermore, attempts to forward normal network packets across the suspect path will continue as well until the Network Manager deletes the links using the path. Communications continues to be problematic and, after the pathFailInterval lapses, a Path-Down Alarm (Command 788) is generated by both of them.

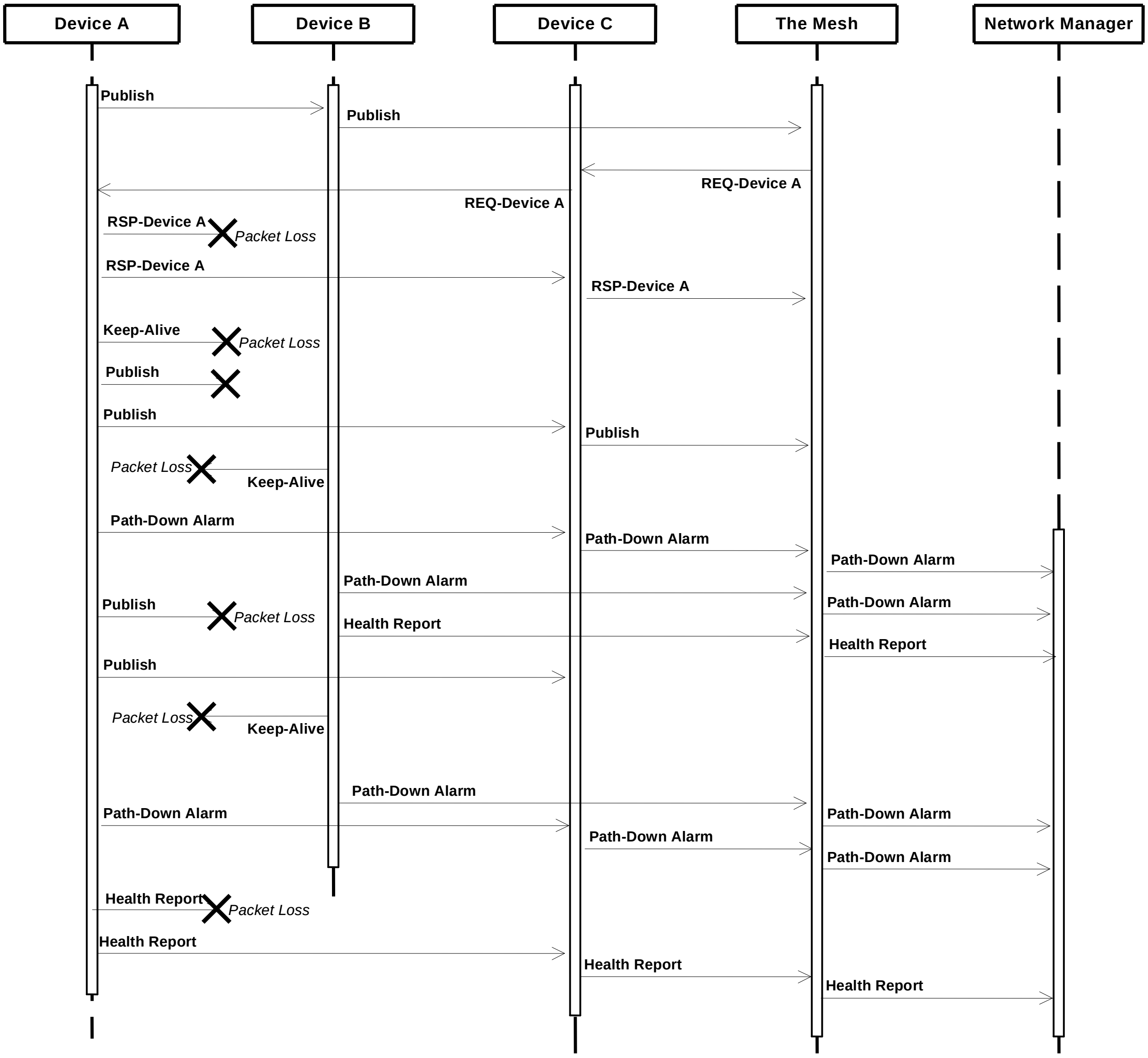


Figure 38. Path Failure

As each device's HealthReport timer lapses, the devices generate health reports, which include indications of any problems the device is having with a neighbor. Notice that, since the devices joined the network at different times, the health reports do not occur simultaneously.

The devices continue trying to reestablish communications until the links between them are deleted by the Network Manager. It is common for broken paths to be restored as a temporary environmental effect passes. If the disruption persists, additional Path-Down Alarms will be generated when the pathFailInterval lapses again.

9.4.6 Changing the Security Keys

Changing the security keys used by the network is an important operation that must be done periodically to ensure the overall integrity of the system. There are four classes of keys to consider: the Join Key; the Network Key; Broadcast Session Keys; and Unicast Session Keys. Two techniques can be used to change these keys. First, since the Network Key and the Broadcast Session Keys are normally common to all devices, these keys can be all changed in a single broadcast commands (e.g., Command 961 or Command 963). Figure 39 illustrates a broadcast message changing the Network Key.

The Join and Unicast Session Keys, on the other hand should be different for every device (and, of course, different from each other). Therefore a Unicast Command 963 or Command 768 changes these keys. The Join Key should be changed immediately after the device exits the quarantine state and enters the operational state.

Key changes can also take affect immediately or at a specified ASN (i.e., a specified future time). Network Key changes should always be scheduled. An immediate change of the network key will disrupt network (in most cases).

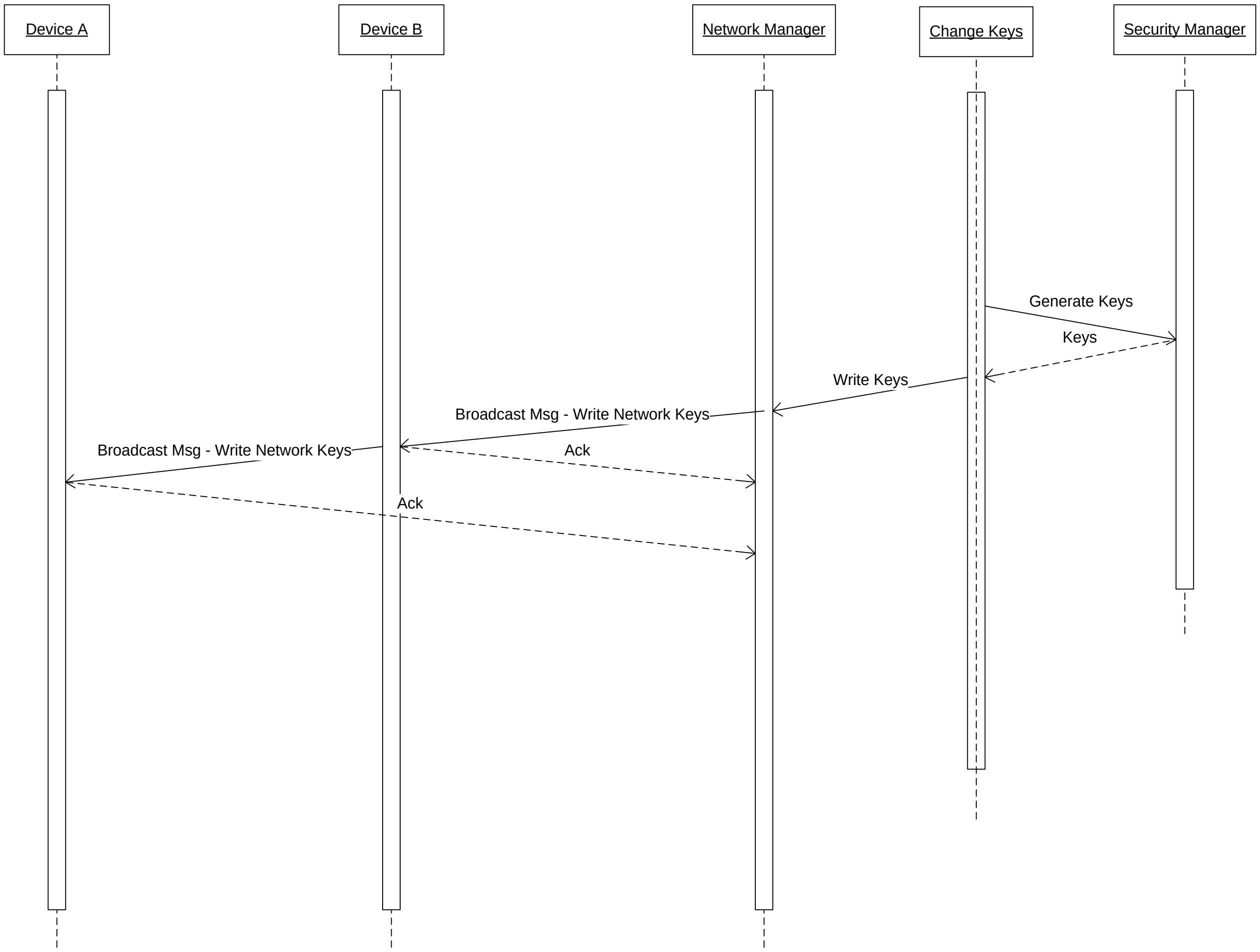


Figure 39. Changing Network Keys

10 HART-IP

HART enables communication with smart process instrumentation and controls, and supports both wired and wireless network topologies. The TCP/IP communication transport extends the applicable physical layers to those that support TCP/IP communication. Possible Physical Layers include Ethernet (802.3), Wi-Fi (802.11b/g) or even RS232 using PPP.

The remainder of this section will describe the basic architecture and components of HART over IP as well as basic implementation guidance.

10.1 General Architecture

The general architecture may include different device types. Standard network infrastructure equipment (switches, routers, bridges, etc.) is outside the scope of this document but may exist and therefore must be accommodated. A HART Device (server) may be a stand-alone Field Device, a Gateway connected to WirelessHART Field Devices or an I/O System connected to traditional 4-20mA HART-enabled Field Devices. Devices such as a serial to HART-IP converter are also possible.

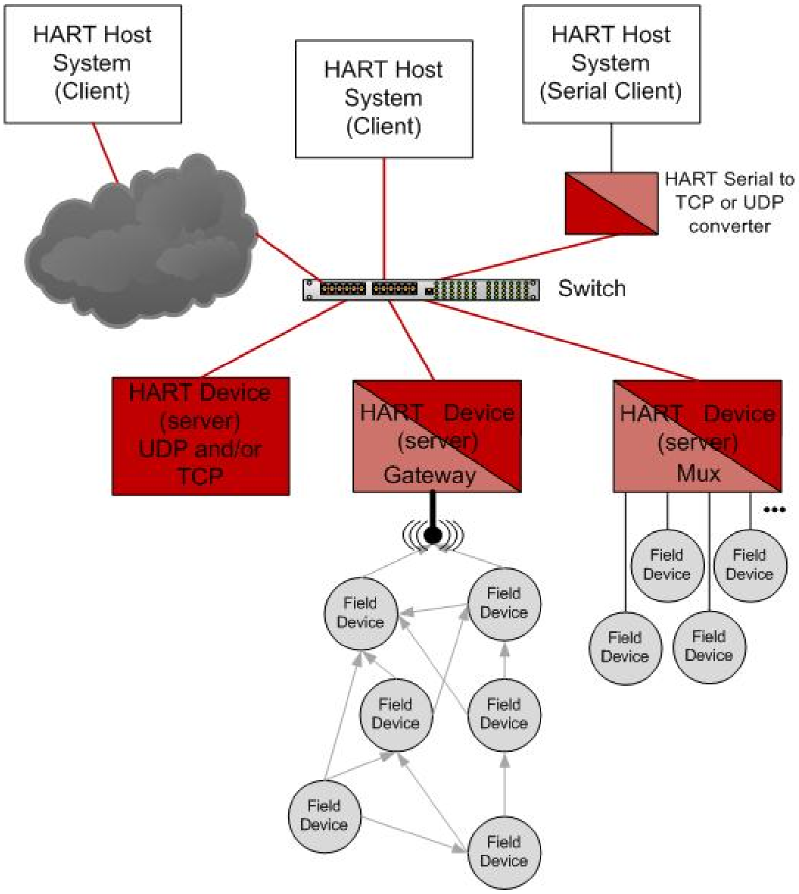

HART Host System (Client)
HART Host System (Client)
HART Host System (Serial Client)
HART Serial to TCP or UDP converter
Switch
HART Device (server) UDP and/or TCP
HART Device (server) Gateway
HART Device (server) Mux
...
Field Device
Field Device
Field Device
Field Device
Field Device
Field Device
Field Device
Field Device
Field Device
Field Device
Field Device

Figure 40. General HART-IP network architecture

10.1.1 HART Host System – Client

The HART client allows a Host Application to implement information exchange with a remote device. The client builds a HART request with information sent by the user application to the client interface. The request is then sent to UDP or TCP port of the server. The server response will be returned via the same transport (UDP or TCP) as the request.

10.1.2 HART IP Device – Server

A HART Device (server) listens for a HART request. The request is processed and a response is returned to the client that made the request. A server must listen to the well-known TCP/IP port **5094.** However, some Host Systems/Clients may require another port be used for HART because of firewall configurations or other network infrastructure needs. Consequently, clients and servers must allow configuration of additional TCP/IP port numbers. The server must listen to both the additional TCP/IP port(s) and the well-known TCP/IP port.

In addition, servers must:

- Support at least 2 clients (sessions). Each session shall have an independent Configuration Changed and More Status Available flags.
- Must implement security appropriate to the end-user application. IP security best practices are beyond the scope of this Specification.
- Support both the TCP and UDP.

10.2 Basic HART-IP Requirements

- It is recommended that a HART Client keep the UDP or TCP initiated HART application session opened with the server and not open and close the application session for each transaction.
    - However a server must be capable of accepting a session close request from the client.
- Several HART transactions may be initiated simultaneously on the same application session.
    - The Sequence Number must be used to uniquely identify and match requests with responses.
    - A server may respond with a NAK if it is unable to manage a response to all the requests initiated. This response is similar to a busy response on the FSK interface in that the Host application should pause for a moment and try the request again.
- If a device supports both client and server functionality, it is necessary to open separate sessions for the client and the server.
- A UDP or TCP HART frame must transport only one transaction. This is to ensure requests can be properly matched with responses using the Sequence Number.

10.2.1 Time-out Management

There are no specific response timeouts requirements. However, to avoid network congestion and failures, Clients must take into account the application and possible transmission delays to determine an appropriate timeout.

10.3 Message Format

The basic message format includes a header to describe the content of the HART-IP payload and the Token-Passing Data-Link Layer PDU format message (see Figure 41 and the *Token-Passing Data Link Layer Specification*). The HART-IP payload must not include the FSK preamble.

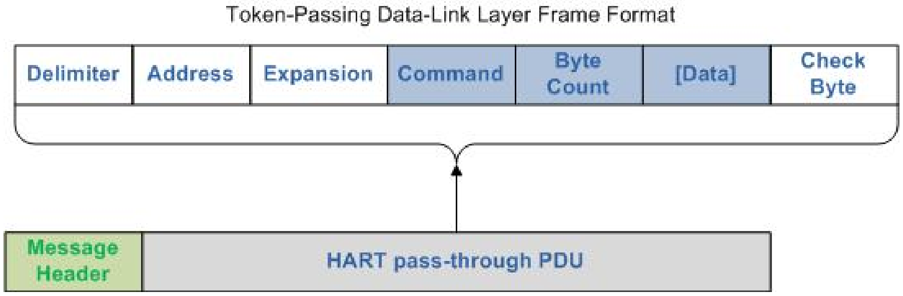
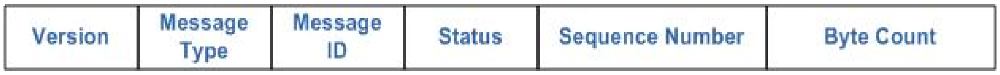


Figure 41. Message Format with embedded Token-Passing PDU (Message ID = 3)

10.3.1 Header

The HART-IP header is fixed length and all fields are mandatory (see Figure 42). The HART-IP header is specified in Table 21 and in the following Subsections.



Version	Message Type	Message ID	Status	Sequence Number	Byte Count

Figure 42. HART-IP Header

Table 21. HART-IP Header Specification

Parameter	Byte	Format	Description	Client	Server
Version	0	Unsigned-8	Protocol version number (currently 1)	Client initialized to 1	Maximum version supported by the server.
Message Type	1.7-1.4	Bits-4	Bits 1.4 – 1,7 = Reserved	Client initialized to zero.	Recopied by the server from request
	1.3-1.0	Enum-4	Message type: 0 = Request 1 = Response 2 = Publish/notification 15 = NAK All other values = Reserved	Client initialized to Request	Server initialized to Response, Publish/notification or NAK
Message ID	2	Enum-8	Message Identifier: 0 = Session Initiate 1 = Session Close 2 = Keep Alive 3 = Token-Passing PDU 128 = Discovery All other values = Reserved	Client initialized	Recopied by the server from request
Status Code	3	Enum-8	Overall communication status for the message. Status values for Response messages are listed in the following section based on message identifier.	Client initialized to 0 (success)	Initialized by the server
Sequence Number	4-5	Unsigned-16	Unique sequence number for mapping a response to a request. This must be unique for all messages, including those used for DRM.	Client initialized	Recopied by the server from request
Byte Count	6-7	Unsigned-16	Number of bytes in the message including message header and body.	Client initialized	Server initialized

10.3.1.1 Version

This field must be set to one (1). In future updates to this specification, the version may be incremented. A server that does not support a version of a header should not respond. A client should always start with version 1. The response from the server will indicate to the client what header version the server supports. A server must be backward compatible with any previous header version.

10.3.1.2 Message Type

This field indicates the type of message (e.g., request, response, publish, NAK) contained in the PDU. The Message Type tells the server how to manage the sequence number. For a request, the server copies the sequence number into the response. If the server is sending a publish or notification it should use its current publish/notification sequence number.

It also provides the capability to define message flags that can alter the way the receiving device responds to a request. If a device does not understand the flag or is not capable of using the flags they should be ignored.

The server may respond to a Request with a NAK instead of the normal Response message. The NAK allows the server to provide feedback to the host system that its resources are currently all allocated. The host system must then pause additional requests for a period of time make the request again. The NAK provides the server the ability to control message flow control based on resources. If the message type is NAK, there is no message payload.

If the server is configured for burst, events or notifications, the publish/notification messages will begin to be sent from the server after the session is initiated by the client. The publish/notification messages are inserted into the session message stream based on the burst, event or notification configuration. Therefore a client must be able to differentiate between a standard response and publish/notification messages for proper message handling at the host application.

If the server receives a response message type, it should be ignored. If any reserved bits are set in the message type field, they must be ignored, cleared for the response and the proper response generated.

10.3.1.3 Message ID

The Message ID describes the message content of the PDU.

10.3.1.4 Status

The status is returned in each response. It is an indication of the status of the particular message identifier sent and indicates if any warnings or errors occurred in the UDP or TCP messaging. Note there is a separate response code for each embedded command as part HART application layer message

10.3.1.5 Sequence Number

A client may initialize the Sequence Number to any value with the session initiation. From that point forward it must increment the Sequence Number. When the Sequence Number reaches the maximum value (65535) it must wrap to zero.

When a new session is initialized, the server must initialize the publish/notification Sequence Number to 0 for any publish/notification messages it will send. The Sequence Number is then simply incremented by the server as each publish/notification message is sent.

10.3.1.6 Byte Count

This is simply the total number of bytes in the Header plus the Body.

10.4 Message Body

The message body starts at the first byte after the HART message header in the UDP or TCP payload. This section describes the format and content of the message body for each message ID and type combination.

As with other HART commands, forward compatibility allows parameters to be added to these message ID's in the future. Therefore servers must allow for these additions and not fail when additional parameters are present.

10.4.1 Session Initiate (Message ID = 0)

This message is used to initiate a session between a Host application/client and a field device or gateway server. This message must be exchanged with the server before other messages will be accepted from the client. If requests are sent before a session has been initiated, the server will not respond even if the message is valid in every other way.

The client sends the request to the well-known port on the server. If a successful exchange occurs, the server returns the response either from the well-known port or from a previously unused UDP (ephemeral) port. This port is then used by the client to send and receive all subsequent messages for the session.

For UDP or TCP, the client sends a Session Initiate request to the well known port on the server. If the initial message is anything other than Session Initiate the server will not respond. Once open, this session is then used by the client to send and receive all subsequent messages for the session. The session and socket will be closed if the server receives a Session Close message id or the session inactivity timeout occurs.

The server limits active sessions based on available resources. If the server is unable to initiate the session because all available sessions are used, it must return the "All Available Sessions in Use" status to the client.

When initiating a UDP or TCP session, the client must specify an "Inactivity Close Time". If the server looses communication with the client (based on the Inactivity Close Time lapsing) the server must close the session. For TCP sessions only, the Inactivity Close Time may be set to zero (0). When this occurs the port will stay open based on the TCP KEEPALIVE settings. The server response must contain the actual values used by the server ("Set to Nearest Possible Value" must be returned if the response value differs from the request value).

Request Data Bytes

Parameter	Byte	Format	Description
Master Type	0	Unsigned-8	1 = Primary Master All other values = Undefined
Inactivity Close Time	1 – 4	Unsigned-32	Milliseconds of inactivity before the session is closed

Response Data Bytes

Parameter	Byte	Format	Description
Master Type	0	Unsigned-8	1 = Primary Master All other values = Undefined
Inactivity Close Time	1 – 4	Unsigned-32	Actual inactivity time value set for the server

Status Codes

Code	Class	Description
0	Success	No error occurred
1		Undefined
2	Error	Invalid Selection (Invalid Master Type)
3 – 4		Undefined
5	Error	Too Few Data Bytes Received
6	Error	Device Specific Command Error
7		Undefined
8	Warning	Set to Nearest Possible Value (Inactivity timer value)
9 – 13		Undefined
14	Warning	Version not supported
15	Error	All Available Sessions In Use
16	Error	Access Restricted, Session already established
17 – 255		Undefined

10.4.2 Session Close (Message ID = 1)

This message is used by the HART Host application/client to request the server close a session.

Note: For TCP, the session is closed whenever the socket for this session is closed for any reason.

Request Data Bytes

None

Response Data Bytes

None

Status Codes

Code	Class	Description
0	Success	No error occurred
1 – 5		Undefined
6	Error	Device Specific Command Error
7 – 255		Undefined

10.4.3 Keep Alive (Message ID = 2)

This client must transmit this message periodically when no other communications with the server are occurring to keep the session active/open. This message should be sent when communication has been inactive for 95% the "Inactivity Close Time".

Request Data Bytes

None

Response Data Bytes

None

Status Codes

Code	Class	Description
0	Success	No error occurred
1 – 5		Undefined
6	Error	Device Specific Command Error
7 – 255		Undefined

10.4.4 HART PDU (Message ID = 3)

This message embeds a full Token-Passing Data Link Layer formatted message as the payload. This provides the server addressing information for routing the message to sub-devices of a Multiplexer or Gateway device so Host applications/clients don't need to be re-written. The format of the response must take the same format as the request. Figure 41 provides a diagram of the wired PDU format.

A client must accept DR responses from any IO system. When sending a message to an IO system sub-device that is not accessible, the server must respond with DR_DEAD.

Request Data Bytes

Parameter	Byte	Format	Description
HART PDU	0 – N	Unsigned-8 [ ]	HART PDU

Response Data Bytes

Parameter	Byte	Format	Description
HART PDU	0 – N	Unsigned-8 [ ]	HART PDU

Status Codes

Code	Class	Description
0	Success	No error occurred
1 – 4		Undefined
5	Error	Too Few Data Bytes Received
6	Error	Device Specific Command Error
7 – 14		Undefined
15	Error	Unsupported Message ID
16	Error	Access Restricted (Server resources exhausted)
17 – 34		Undefined
35 – 255		Undefined

10.5 Session Management

This section describes the basic session setup, maintenance and clean-up.

10.5.1 General Overview

A HART-IP server may allow any number of sessions. The number is based on available server resources. Once the maximum number is reached, further session initiate requests will be refused with an error response (All Sessions In Use).

10.5.2 Normal Operation

Normal operation consists of:

- A client sending a Session Initiate (Message Id = 0) request
- The server sending the session initiate response with the actual keep alive time
- The client sending any combination of:
  - HART pass-through PDU (Message Id = 3) request followed by a server response
  - Keep Alive (Message Id = 2) request followed by a server response
  - HART pass-through PDU publish (Message Id = 3)
- The client sending a Session Close (Message Id = 1) request
- The server sending the session close response and then closing the connection for the client.

Figure 43 shows an example sequence diagram of a normal set of Message Id's and Sequence Numbers sent during a normal session. The values shown in parentheses are sample sequence numbers. Figure 44 shows some possible error sequences and the expected response from the HART server.

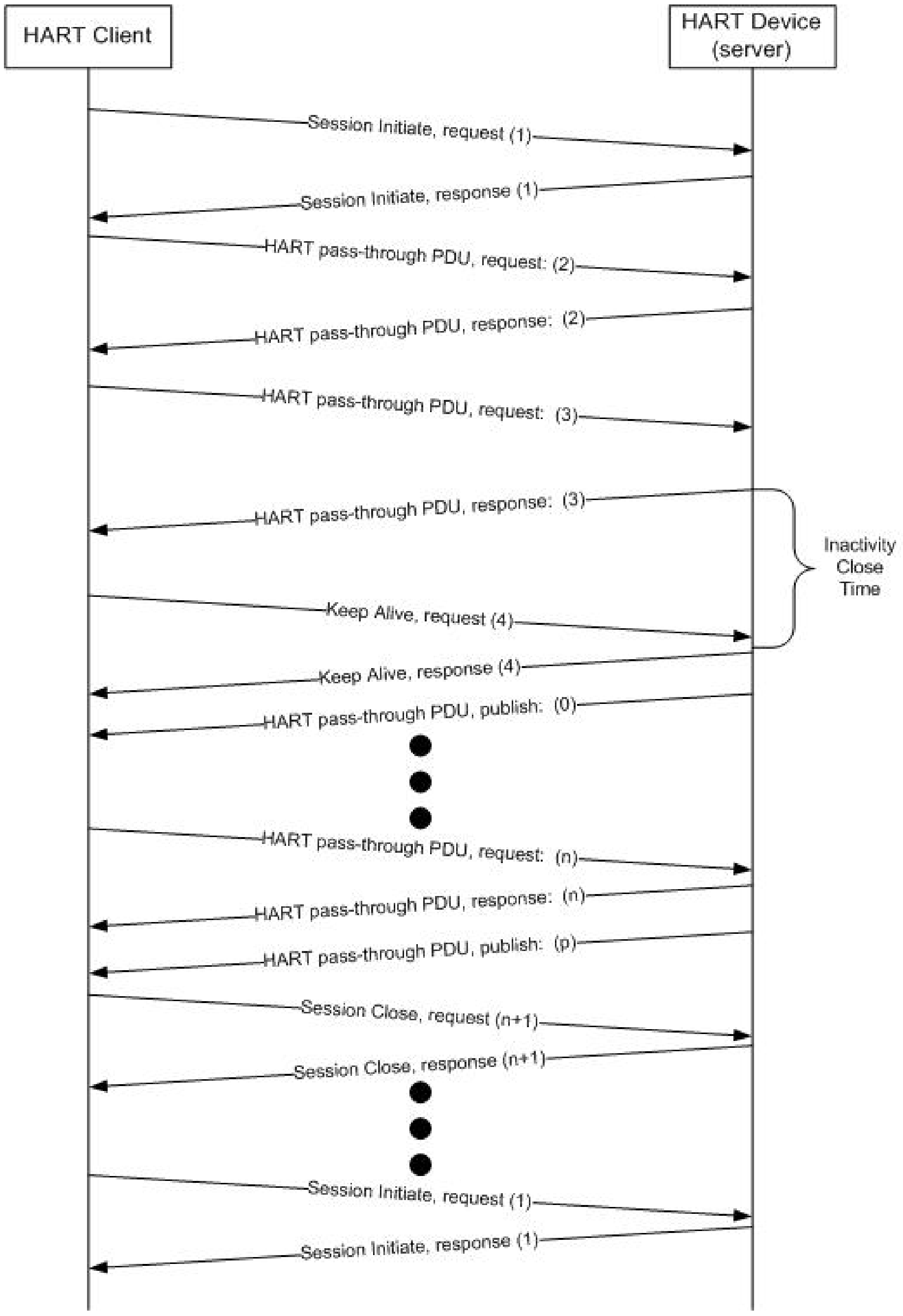

HART Client
HART Device (server)
Session Initiate, request (1)
Session Initiate, response (1)
HART pass-through PDU, request: (2)
HART pass-through PDU, response: (2)
HART pass-through PDU, request: (3)
HART pass-through PDU, response: (3)
Inactivity Close Time
Keep Alive, request (4)
Keep Alive, response (4)
HART pass-through PDU, publish: (0)
HART pass-through PDU, request: (n)
HART pass-through PDU, response: (n)
HART pass-through PDU, publish: (p)
Session Close, request (n+1)
Session Close, response (n+1)
Session Initiate, request (1)
Session Initiate, response (1)

Figure 43. HART TCP normal transaction sequence diagram

10.5.3 Error Conditions

Figure 44 shows example sequence diagrams of error conditions that a server may encounter

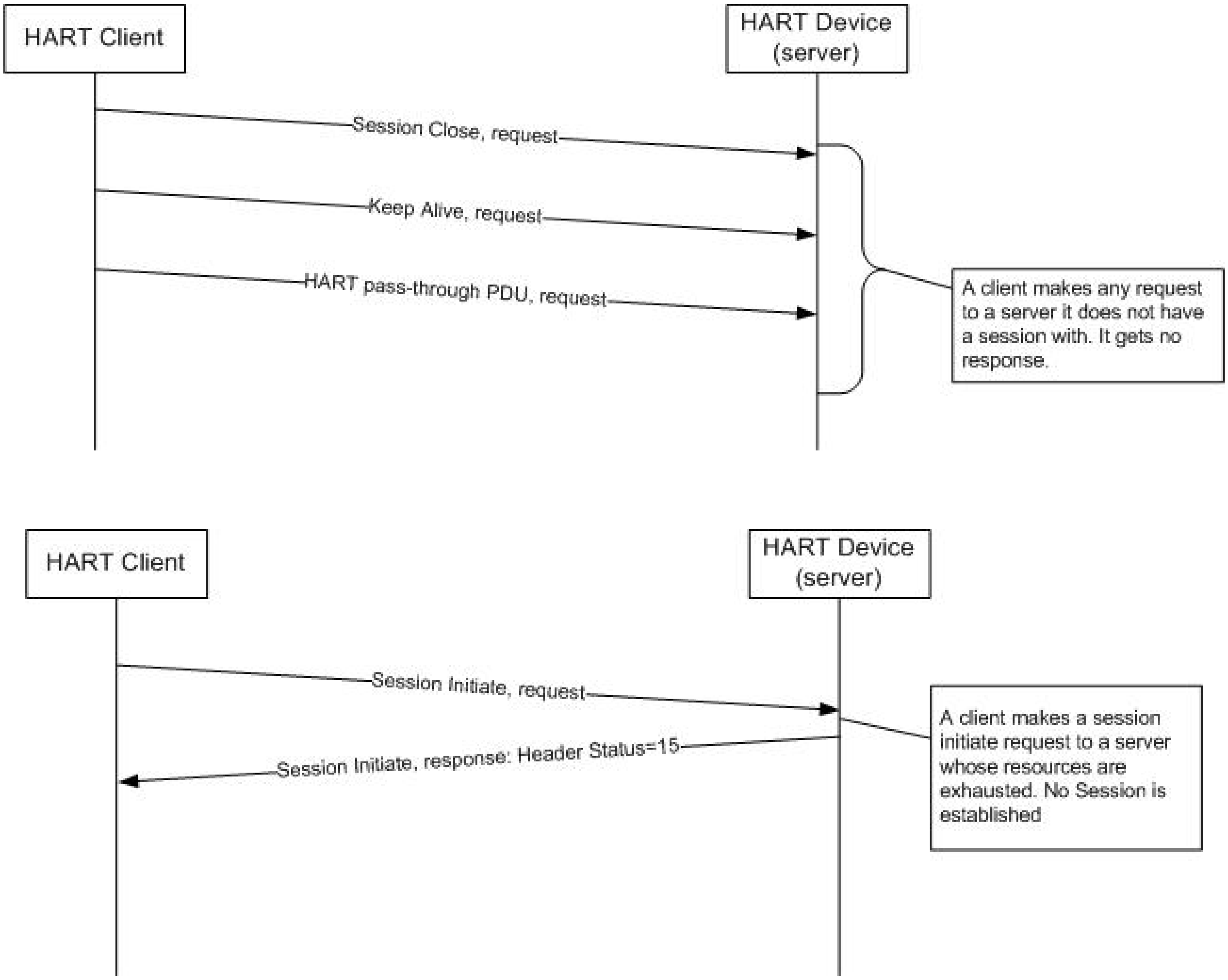

HART Client
HART Device (server)
Session Close, request
Keep Alive, request
HART pass-through PDU, request
A client makes any request to a server it does not have a session with. It gets no response.
HART Client
HART Device (server)
Session Initiate, request
Session Initiate, response: Header Status=15
A client makes a session initiate request to a server whose resources are exhausted. No Session is established

Figure 44. HART TCP example error transaction sequence diagrams

10.6 Discovery Protocol

Support the HART-IP Discovery Protocol is optional for clients and servers[36].

The Discovery Protocol consists of a HART-IP Client UDP broadcast followed by UDP unicast response from HART-IP servers (see Figure 45). The Discovery UDP packets consist of the standard HART-IP header (see Figure 42) with the Message ID set to 128 (Discovery). The Discovery The following diagram illustrates the exchange of Discovery UDP messages between a Client and a device during the discovery process.

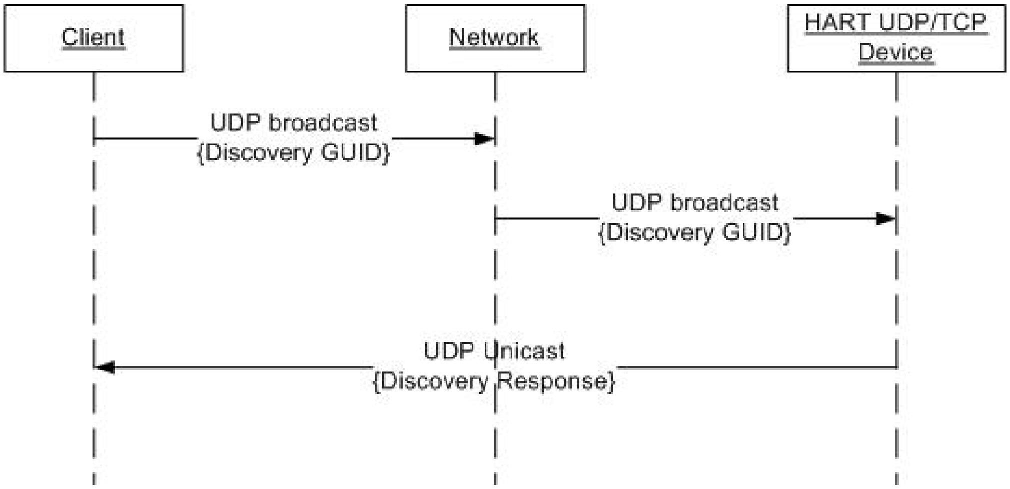

Client
Network
HART UDP/TCP Device
UDP broadcast {Discovery GUID}
UDP broadcast {Discovery GUID}
UDP Unicast {Discovery Response}

Figure 45. HART-IP Discovery Protocol sequence diagram

Servers supporting Discovery typically listen to the well-known TCP/IP port 5094 for discovery broadcasts from potential clients. However, servers, clients or plant network requirements may require a different port be used for Discovery. Consequently, clients and servers should allow configuration of different optional TCP/IP port number for use in Discovery.

When a broadcast message is received, the content of the message is inspected for the correct "Discovery Globally Unique Identifier (GUID)". The default (standard) Discovery GUID is shown in Figure 46. The Discovery GUID must be transmitted in network byte order (i.e., big endian)

4C0F38AC - 48AE - 4935 - B689 - 8F21F85FC03
0

Data1 Data Data Data4
2 3

Figure 46. Default Discovery GUID

If the discovery broadcast request contains the Discovery GUID the server must respond with unicast discovery response. The unicast UDP response contains the Discovery GUID, the version of the response packet, the payload byte count, IP Address, Port number and the device’s Command 0 and 20 responses. The IP Address and Port Number specify where the client shall initiate the session with the device. The discovery request and response packets are defined in Subsection 10.6.3.

The Port and GUID used for discovery should be configurable parameters in the device

10.6.1 Discovery Request (Broadcast)

The broadcast UDP packet consists of the standard HART-IP header (see Figure 42) with the Message ID set to 128 (Discovery), the Message type set to 0 (Request) and the payload set to the Discovery GUID. Typically a host system will send discovery requests every 15 seconds. This keeps the broadcast traffic to a minimum and does not put an excessive message management burden on the servers. Also it is expected that a host

[36] Even if the Discovery Protocol is supported clients and servers must support manual entry of IP addresses and port numbers.

system discovery client will only make requests when new devices are added to the network. Once the new server(s) are discovered, the discovery client will stop broadcasting requests.

10.6.2 Discovery Response

The discovery response is only sent if the GUID in the discovery request matches the value in the device. The discovery response is a unicast UDP packet sent from the server. The response payload is placed inside a standard HART-IP header (see Figure 42) with message id set to discovery and message type set to response. The device returns the GUID and identification information to the client. Version 1 supports IPv4 and IPv6 addressing.

10.6.3 Discovery Data Field Formats

This subsection summarizes the Discovery Client Request and Server Response packet formats. These fields are inserted into the UDP packet following the standard HART-IP Header (see Figure 42).

Discovery Client Request Data Bytes

Byte	Format	Description
0-3	Unsigned 32	GUID "Data1" (e.g., 0x4C0F38AC)
4-5	Unsigned 16	GUID "Data2" (e.g., 0x48AE)
6-7	Unsigned 16	GUID "Data3" (e.g., 0x4935)
8-15	Unsigned 64	GUID "Data4" (e.g., 0xB689 8F21F85FC030)

Discovery Server Response Data Bytes

Byte	Format	Description
0-3	Unsigned 32	GUID "Data1"
4-5	Unsigned 16	GUID "Data2"
6-7	Unsigned 16	GUID "Data3"
8-15	Unsigned 64	GUID "Data4"
16	Unsigned-8	Discovery response payload version (e.g., 1)
17.7-17.1	Bits-7	Reserved (must be set to 0)
17.0	Bits-1	IPv6 when set (reset=IPv4)
18-19	Unsigned-16	Byte Count is entire payload
20-35	Unsigned-128	IPv6 Address (or IPv4 Address with Bytes 24-35 set to 0x00)
36-37	Unsigned 16	Port. Port number for establishing a session with the server (device)
38-39	Unsigned 16	Command 0
	Unsigned 8	Byte Count
	Unsigned 8	Response Code (Success)
	Bits 8	Device Status
	Unsigned *nn*	Command 0 Response Data (See Universal Command Specification)
	Unsigned 16	Command 20
	Unsigned 8	Byte Count
	Unsigned 8	Response Code (Success)
	Bits 8	Device Status
	Unsigned *mm*	Command 20 Response Data (See Universal Command Specification)

11 NETWORK MANAGEMENT

The Protocol has many additional features typically found in the ISO Network Layer. This section describes the mechanisms and procedures for masters to locate and address slave devices and to accomplish proper message routing. In particular, this section includes:

- A description of the Identity Commands that allow communications with field device to be established.
- The definition of Sub-Devices that allow communications to HART-compatible devices via intermediate bridging or I/O devices. Commands are provided to allow the identification of connected devices and the routing of messages to them.
- Procedures that are specified that allow masters to identify field devices and initiate communications with them.
- Specifications for device support of multi-drop networks and the routing of messages across multi-drop networks.
- Requirements for support of Burst Mode Operation and references to the commands used by a master to manage Burst Mode Operation.

11.1 Identity Commands

Identity Commands are used by a master to identify a field device and begin a communication session with the field device. There are five Identity Commands:

- Command 0 Read Unique Identifier
- Command 11 Read Unique Identifier Associated With Tag
- Command 21 Read Unique Identifier Associated With Long Tag
- Command 73 Find Device
- Command 75 Poll Sub-Device

All of these commands return the same Response Data Bytes. These commands return the data items necessary to

- Generate the long frame address;
- Route commands to the appropriate field device; and
- Determine the command set and data items supported by the field device.

Command 0 is unique in that it is the only command that still supports short frame messages. Command 0 can use the polling address to automatically identify devices on the loop. Commands 11, 21, and 73 all use long frame messages containing the Broadcast Address. The Broadcast Address has all zeros for the field device address and uses Request Data Bytes (Command 11, 21) or a mechanical means (Command 73) to identify the field device. Command 75 allows sub-devices connected to an I/O system to be identified (see the *Common Practice Command Specification*).

Once basic addressing and command set identification is complete, the host should use Commands 13 (Tag), 15 (Private Label Distributor), and 20 (Long Tag) to complete the identification of the field device.

11.2 Establishing Communication with a Field Device

To establish connection with the field device, the master must determine the long frame address of the field device. This long frame address is used by all HART commands (see the *Data Link Layer Specification*). The long frame address can be determined using one of the following procedures:

- Poll using Command 0;
- Poll for Sub-Devices Using Command 75;
- Poll by Tag using Command 11;
- Poll by Long Tag using Command 21;
- Mechanically Identify the field device using Command 73; and
- Manual Entry of the Extended Device Type, and Device ID.

For each of these techniques, if an asynchronous Physical Layer (e.g., FSK or RS-485) is used then until the connection is established with the slave, the master should use a large number of preamble characters (e.g., 20). Once connected to the field device, the Identity Command indicates the minimum number of preambles the master must use.

Masters must support all procedures in this section.

11.2.1 Using Polling Addresses

A master can poll the loop using all legal short frame addresses from Command 0 to identify the devices connected to the loop. Command 0 is unique because master request messages are the same for all versions of the Protocol. Using the identity information the master can determine the Protocol version supported by the field device and assemble the long frame address for the slave device. The basic procedure is:

1 The Master sends Command 0 using short frame format.

2 If a slave with the requested short frame (polling) address is connected to the loop, the slave must answer. Since there are several polling addresses possible, polling all addresses may take a few minutes (perhaps longer if retries are attempted). As a result the master should allow the user to limit the range of Polling Addresses when using this field device identification algorithm.

3 Steps 1 and 2 are repeated until all the devices on the loop are identified.

This procedure requires all field devices on a loop to be set to a unique Polling Address prior to commissioning the loop.

11.2.2 Polling for Sub-Devices

When a Bridge Device or I/O System is detected on the network, a master may use Command 75 to identify the connected sub-devices. Using the identity information, the master can determine the Protocol version supported by the field device and assemble the long frame address for the slave device. The basic procedure is:

1. The Master identifies a Bridge Device or I/O System (i.e., bit 2, Protocol_Bridge_Device, in the Flags byte of Identity Commands is set).
2. The Master issues Command 74 to determine how many sub-devices may be connected.
3. The Master issues Command 75 to a polling address on an I/O Card Channel.
4. If a sub-device with the requested polling address is connected, it must answer.
5. Steps 3 and 4 are repeated until all the sub-devices are identified.

This procedure requires all sub-devices connected to an I/O Card Channel be set to a unique Polling Address prior to commissioning. Since the number of possible sub-devices is potentially very large, the I/O System must provide information to minimize the polling time required by the host. This is achieved primarily by setting the data item in Command 74 appropriately. In addition, I/O systems are encouraged to implement Command 75 Response Code 9, "No Sub-Device Found" to expedite sub-device identification and minimize master polling time.

11.2.3 Polling by Tag or Long Tag

All HART field devices support both an 8 character Tag, and a 32 Character Long Tag. Tags have been used in process plants to identify field devices long before smart field devices were ever possible. HART allows a field device to be identified using the Tag (via Command 11) or Long Tag (via Command 21). In both cases the request message is transmitted using the Broadcast Address with the Data field containing the string indicating which field device is to answer. The basic procedure is:

1. Allow the user to enter the Tag or Long Tag to be located.
2. The Master sends the appropriate command using long frame format containing the Broadcast Address.
3. If a slave with the Tag or Long Tag is connected to the loop, it must answer.
4. Steps 1 through 3 are repeated until all the devices on the loop are identified.

This procedure requires all field devices on a loop to be set to the correct Tag and Long Tag prior to commissioning the loop.

Note: Broadcast addresses are addressed to all field devices (see *Data Link Layer Specification*). As a result, Broadcast addresses must be routed through I/O Systems and Bridge Devices to all sub-devices.

11.2.4 Mechanical Identification of the Field Device

If the field device supports Command 73, the field device can be mechanically identified. Field devices implementing this command only respond when physically/mechanically signaled to do so. For example, the technician presses a special button or combination of buttons that indicate the slave is to answer this command. The basic procedure is:

1. The user arms the field device (e.g., by pressing a button).
2. The master then sends the Command 73 to identify the field device
3. The field device answers the command once and disarms itself.
4. Steps 1 through 3 are repeated until all the devices on the loop are identified.

In addition to establishing the connection to the field device, this procedure allows the user to verify the device is installed in the correct plant location and on the correct wire pair.

11.2.5 Manual Entry of the Extended Device Type, and Device ID

Field device identification using the above procedures requires little or no information to be supplied by the user. In addition to the above procedures, the communications connection can also be established by the manual entry of the data necessary to construct the long frame address. With the long frame address known, Command 0 can be sent (with the long frame address) to complete the identification of the field device.

Since this requires manual entry of data not generally available to the user, masters must not rely on this procedure as its primary means of establishing a connection to the field device.

ANNEX A. REVISION HISTORY

A.1. Revision 2.0 (18 June 2012)

Added HART-IP and addressed and integrated over 50 additional issues recorded in HCFTracker

- Section 2. References to FIPS-197, AES, GUID and Endianness have been
- Section 3-4. Added a few new definitions and acronyms.
- Section 6-9 includes corrections and clarifications based on feedback from developers and the HCF Working Groups. An overview of HART-IP was added to Section 6.
- Section 10 (NEW!) specifies HART-IP, a relatively simple specification for the transport of HART over IP-based networks. HART-IP works over UDP or TCP using IPv4 or IPv6. To enable rapid acceptance and product development, HART-IP payloads are based on HART Token-Passing Data-Link Layer PDUs.

  Security best-practices for IP-based products continue to evolve and HART-IP is designed to be security agnostic. In o her words, while security is not specified all products should implement best-practice security measures.

  HART-IP was initially developed for I/O Systems (e.g., multiplexers and Gateways). However, HART-IP is also well suited for traditional process instruments and analyzers. Power-Over-Ethernet (POE) combined with HART-IP enables development of a wide range of fast, new HART compatible products.

A.2. Revision 1.0 (27 August, 2007)

Initial Revision.