Deji Chen
Mark Nixon
Aloysius Mok

WirelessHART™

Real-Time Mesh Network for Industrial Automation

Springer

WirelessHART™

Deji Chen • Mark Nixon
Aloysius Mok

WirelessHART™

Real-Time Mesh Network for Industrial Automation

Springer

Deji Chen
Emerson Process Management
12301 Research Blvd.
Research Park Plaza, Bldg. III
Austin, TX 78759
USA
deji.chen@emerson.com

Mark Nixon
Emerson Process Management
12301 Research Blvd.
Research Park Plaza, Bldg. III
Austin, TX 78759
USA
mark.nixon@emerson.com

Aloysius Mok
Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712
USA
mok@cs.utexas.edu

ISBN 978-1-4419-6046-7 e-ISBN 978-1-4419-6047-4
DOI 10.1007/978-1-4419-6047-4
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2010925230

Printed on acid-free paper

To my father Zhengcai Chen, my mother Jiangen Wang, my wife Qing Li, and my newborn daughter Daphne Chen.

Deji Chen

To Joshua and Jordan.

Mark Nixon

To my family especially my wife Amy who has been a great companion in life's adventures.

Aloysius Mok

Acknowledgements

We would like to thank the following persons without whom this book will not be possible. First we thank Jose Gutierrez, Ron Helson, and Quan Wang for their encouragement for starting the book. We owe a special gratitude to Wally Pratt, the chief engineer at the HCF, and the members of the specification team consisting of Eric Rotvold, Robin Pramanik, and Tomas Lennvall. Without all of your hard work this book would not have been possible. We also thank Veena Gondhalekar, Jianping Song, Song Han, Xiuming Zhu, Terry Blevins, Willy Wojsznis, and Greg McMillan for cooperating with us on the WirelessHART related research and development effort which provided the material for this book. Song Han and Xiuming also provided valuable feedback to the initial draft of the book. We thank Kelly Orth for important input to the book. We thank Mike Sheldon for much needed advices. And finally, we thank Susan Lagerstrom-Fife and Jennifer Maurer for their understanding in working with us closely to bring this book to reality.

Table of Contents

Table of Figures

List of Tables

Introduction

The process control industry has seen generations of technology advancement, from pneumatic communication to electrical communication to electronic communication, from centralized control to distributed control. At the center of today's distributed control systems are operator workstations. These operator workstations provide the connection between those overseeing and running plant operations to the process itself. With each new generation of products the operator workstation has become increasingly more intelligent. Newer applications provide advanced alarming, control, and diagnostics. Behind all of these applications are smarter devices. These smart devices provide greater process insight, reduce engineering costs, and contribute to improving the overall operational performance of the plant. Smart devices include advanced diagnostics that can report the health of the device and in many cases, the health of the process that the device is connected to. It is not uncommon for smart devices to include diagnostics that can detect plugged lines, burner flame instability, agitator loss, wet gas, orifice wear, leaks, and cavitations. These devices tell the user how well they are operating and when they need maintenance. Improvements in sensor technology and diagnostics have lead to a large variety of smart devices. So how do users connect the capabilities of these smart devices to their existing control system infrastructures?

The answer is wireless. Wireless technology has matured to the point that it now can be safely applied in industrial control, monitor, and asset management applications. It provides a cost-effective alternative communication path for many legacy control systems, enabling access to the intelligent information in field devices. For new opportunities, measurements that were cost-prohibitive now can be measured and included through wireless into the control and monitoring strategies. It provides a cost-effective, simple and reliable way to deploy new points of measurement and control without the wiring costs and without having to completely change existing systems. It also provides an infrastructure for both central as well as mobile users to access their process and process equipment.

The process control industry has been in need of a wireless solution that establishes a global standard. The benefit of a standard is to insure that devices from multiple suppliers work together – thereby lowering risk and cost for both the supplier and the user. HART™ Communication Foundation (HCF) is in a unique position and has developed such a standard.

The HART communication protocol has served as the world's leading process communication technology for smart instruments since 1989. Industrial suppliers are manufacturing and shipping HART products in record numbers. More than 30 million HART devices are installed and in service worldwide. 75% of the smart devices installed are HART-enabled.

However, about 85% of those installed smart HART devices have their rich data stranded in the devices without being accessed by diagnostic applications. This is often due to the cost and the difficulty of accessing this information. Only the process variable data is communicated via the 4-20mA analog signal.

Continuing decades of open standard development, the HCF and its 230 plus member companies sought a wireless standard designed for the unique demands of the process industry. Drawing on the technical resources and expertise of the member companies, HCF created a new wireless technology that expands the capabilities of the HART protocol while protecting the global installed base. With the latest evolutionary enhancement to the global HART standard, the Foundation takes the proven field communications, networking and security protocols and integrates them into a simple, reliable and secure wireless standard.

WirelessHART™ Technology is Simple

The WirelessHART standard is a robust technology yet simple to implement. It provides the same safe, easy, dependable experience that users know and expect from HART-related products by maintaining compatibility with existing HART devices, tools, and systems. This enables users with HART experience to quickly and easily gain the benefit of wireless technology.

The easiness comes partly from the automatic network features. A WirelessHART mesh is a self-organizing self-healing adaptive network. It automatically adjusts to changes in plant infrastructure, e.g., when a new instrument is added to the network.

The easiness also comes from the wireless nature. Reduced wiring and materials costs leads to easy installation and commissioning, hence reduced labor cost. Without wires, the network can be easily extended to remote areas. And the network can be created one device at a time rather than one big configuration start up.

Compatibility with the wired HART standard makes the deployment of wired and wireless devices together possible, and allows WirelessHART systems to be seamlessly integrated to existing host, DCS, or asset management applications.

Reduced engineering and project cost and time lead to other operation benefits as well. Accessing additional measurements through WirelessHART mesh is fast and easy, which eliminates manual data collection, extends visibility to remote plant areas such as tank farms, utilities, etc. Having access to additional measurements and diagnostics enables loop troubleshooting, streamlines maintenance procedures, and makes it possible to a much wider range of sites to realize predictive maintenance. Wireless technology can also be used to aid in regulatory compliance such as health, safety and environmental compliance monitoring.

Wireless has other benefits as well. The elimination of wires makes it possible to mount instruments on movable assets such as railcars and stationary rotating equipment such as kilns and have those devices communicate with the main control system in the same manner as traditional devices. Wireless also makes it much easier to quickly set up temporary configurations for process studies.

WirelessHART Technology is Reliable

The WirelessHART standard includes several features to provide reliable communications in all industrial environments, which are often unfriendly to wireless communications. Industrial facilities usually have dense infrastructure with metals that interferes with wireless transmission. There are frequent movement of large equipment and changing conditions. Numerous sources of radio-frequency and electromagnetic interference may cause communication challenges. The WirelessHART standard uses both direct sequence and frequency hopping spread spectrum techniques to spread communications among different physical channels.

A WirelessHART mesh is a redundant, self-healing network. Instead of star or tree topology, it is a truly a mesh network with multiple access points. It monitors paths for degradation and repairs itself, finds alternative paths around obstructions, and randomly communicates on different channels. During the lifetime of the mesh network, it constantly adapts itself to changes in the environment. Adaption makes use of health reports and diagnostic information which are continually communicated by all devices in the network.

A WirelessHART mesh also includes many techniques to coexistent nicely with other wireless networks; they being WirlessHART networks or different networks. It employs clear channel assessments tests on the targeted channel before actual transmission. Those constantly noisy or occupied channels could be blacklisted not to use. The transmission is highly synchronized, which both provides on-time messaging and optimizes bandwidth and radio time.

WirelessHART Technology is Secure

The WirelessHART standard employs robust security measures to protect the network and secure the data at all times. These measures include the latest security technologies to provide the highest levels of protection available. It uses industry standard 128-bit AES encryption algorithm at multiple tiers. The secret network key is used at the data link layer to authenticate each data transmission. At the network layer, each session has a different key to encrypt and authenticate per-to-peer communication. A different join key is used for each device to encrypt and authenticate during device join process. In addition, the network manager periodically changes all the keys during the life time of the network.

The WirelessHART standard also employs several technologies to protect the mesh network itself. It uses channel hopping at the timeslot level; the actual physical transmission channel is selected at the point of transmission. Device transmission power is controllable by the network manager. High power can be used in noisy environments while low power can be used when the physical area the network is installed in is very small – for example when a bioreactor is being used. Low power also makes it more difficult for intruders to sniff communication traffic.

The WirelessHART[1] standard focuses on the needs of the industry. It supports all phases of the plant life cycle, from fast engineering, deployment and commissioning, to easy diagnosing and trouble shooting, to cost-effective move from scheduled to predictive maintenance. Officially released in September 2007, it is the first open and interoperable wireless communication standard for process automation. It is cost-effective and backward compatible with currently installed HART devices, and is fully supported by the HART-enabled device, tools and systems used today. In addition to past and present, the HART standard makes sure that the current and future devices will work together.

[1] HART and WirelessHART are trademarks of HCF. Other trademarks:
Bluetooth is a trademark of Bluetooth SIG, Inc.
CC2430 is a trademark of Texas Instrument.
ControlNet and DeviceNet are trademarks of ODVA.
DeltaV and Rosemount are trademarks of Emerson Process Management.
Foundation Fieldbus is a trademark of Fieldbus Foundation.
FreeScale, BeeKit, MC1321, MC1322, JM128, and Codefire are trademarks of FreeScale Semiconductor, Inc.
IEEE, EUI-64, OUI, and 802 are trademarks of the Institute of Electrical and Electronics Engineers, Inc.
ISA is a trademark of International Society of Automation.
ISO and OSI are trademarks of the International Organization for Standardization.
Maxim is a trademark of Maxim Integrated Products.
Modbus is a trademark of Modbus Organization.
OPC is a trademark of OPC Foundation.
Profibus is a trademark of Siemens, Inc.
Wi-Analys, Wi-HTest, HCF_KIT-190, and HCF_KIT-193 are trademarks of HCF.
Wi-Fi is a trademark of Wi-Fi Alliance.
Wibree is a trademark of Nokia Corporation.
Wikipedia is a trademark of Wikimedia Foundation.
WiMAX is a trademark of WiMAX Forum.
WINA is a trademark of Wireless Industrial Networking Alliance.
Zigbee Alliance is a trademark of Philips Corporation.

PART I WirelessHART in a Nutshell

In this part we condense the whole WirelessHART specification into a short booklet. After reading this part, the reader is encouraged to further read the specification itself to gain detailed knowledge. Only a HART member is granted full access to the HART standard. However, the WirelessHART Communication Specification (HART 7.1) was approved by the International Electrotechnical Commission (IEC) as a Publicly Available Specification (IEC/PAS 62591Ed. 1) on 19 September 2008. Please refer to the HART Communication Foundation website http://www.hartcomm.org for details.

Chapter 1 tells the history of HART.
Chapter 2 is the physical layer.
Chapter 3 is the data link layer.
Chapter 4 is the network layer.
Chapter 5 is the application layer.
Chapter 6 describes the whole WirelessHART mesh network.

Terms Used Describing Message Flow within the Stack

Table I.1 Terms used describing message flow within the stack.

Message Term	Meaning of the message
Request	Upper layer sends a request to the lower layer
Confirm	Lower layer replies to the request
Indicate	Lower layer sends a request to the upper layer, mostly containing a new message from another network node.
Respond	Upper layer replies to the indication.

For example, Node A sends a command to Node B, who sends a command response back to A. The command from A will be a *request* message travelling downward through A's network stack layers and will be an *indicate* message travelling upward through B's stack layers. The command response from B will be a *respond* message travelling downward through B's stack layers and will be *indicate* message travelling upward through A's stack layers. Within a node, after a message is transmitted out, a *confirm* message could be sent upward within the stack layers.

The interaction between network layers

A lower layer usually provides a set of Application Program Interfaces (APIs) to the upper layer. They are called Service Primitives (SPs). The messages sent from the upper layer to the lower layer are request or respond SPs. The messages send upward will be confirm or indication SPs.

SPs are categorized into two sets, the Message SPs and Management SPs. The Message SPs are for the message transmissions. The Management SPs are used to configure each stack layers.

Chapter 1 Overview

Abstract This chapter provides a general description of the WirelessHART standard. The HART standard has been in existence for more than 20 years. It has the most field devices deployed in the world among all fieldbus networks. In its latest release, version 7, the WirelessHART component is introduced to bring wireless technology to the process field. The WirelessHART standard is a secure networking technology operating in the 2.4GHz ISM radio band. It utilizes IEEE 802.15.4™ compatible DSSS radios with channel hopping on a packet by packet basis. The WirelessHART standard shares the same application layer with the wired counterpart. It has its own network layer, data link layer, and physical layer, which are briefly described in Section 1.3. Section 1.4 gives a simple example of a WirelessHART application.

1.1 About the HART Standard

The HART standard (www.hartcomm.org) has been in existence since the late 80's. In its initial release the HART Field Communications Protocol was superimposed on a 4-20mA signal providing two-way communications with smart field instruments without compromising the integrity of the measured data. During its 20 plus years of existence the HART protocol has evolved from a simple 4-20mA based protocol to the current wired and wireless-based technology with extensive features supporting security, unsolicited data transfers, event notifications, block mode transfers, and advanced diagnostics. Diagnostics now include information about the device, the equipment the device is attached to, and in some cases the actual process being monitored. The evolution of the HART standard is summarized in Fig. 1.1.

The most recent version of the HART standard, version 7, introduces many new features providing improved performance, enhanced diagnostics, and better maintenance capabilities. The HART Version 7:

- includes wireless mesh networking,
- adds time synchronization and time stamping of data,
- enhances publish/subscribe (burst mode) messaging,
- adds transport layer,
- adds network layer,
- adds pipes for high speed file transfer, and
- adds security/encryption/decryption.

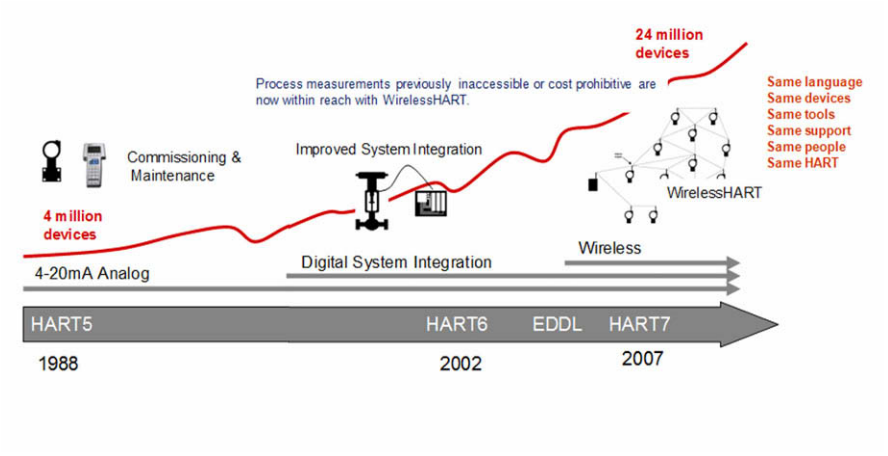

24 million devices
Process measurements previously inaccessible or cost prohibitive are now within reach with WirelessHART.
Same language
Same devices
Same tools
Same support
Same people
Same HART
Commissioning & Maintenance
Improved System Integration
WirelessHART
4 million devices
Wireless
4-20mA Analog
Digital System Integration
HART5
HART6
EDDL
HART7
1988
2002
2007

Fig. 1.1 The Evolution of HART.

1.2 About the WirelessHART Standard

HART 7 includes a major new communication protocol, the WirelessHART protocol, supporting wireless applications. Like wired HART protocol, WirelessHART protocol targets fixed sensors and actuators. Rotating equipment, such a kiln dryer, and flexible manufacturing are also target markets. WirelessHART is needed both for protecting user's investment in their installed base and for new opportunities. We need to preserve legacy products and applications, to continue existing work practices and trainings; we also need to use wireless to lower the cost of making a measurement, access advanced diagnostics information, and have better equipment monitoring.

The WirelessHART standard leverages existing standards such as the HART standard, the IEEE-802.15.4 standard (www.ieee802.org/15/pub/TG4.html, Gutierrez et al. 2007), AES-128 encryption (US FIPS Publication 197), and DDL/EDDL (www.eddl.org). It can do whatever the wired HART part can do and more. The WirelessHART technology is a secure networking technology operating in the 2.4GHz ISM radio band. It utilizes IEEE 802.15.4 compatible DSSS radios with channel hopping on a packet by packet basis. The WirelessHART network supports a wide variety of devices from many manufactures. Fig. 1.2 illustrates the basic network device types, which include:

- field devices as basic devices performing field sensing or actuating functions,
- router field devices mainly serve as routers,
- adapter field device that binds wired HART devices into the mesh,
- handheld device carried by mobile users,

- access points that connect field devices to the gateway,
- a single gateway (may be redundant) that functions as a bridge to the host applications, and
- a single network manager (may be redundant) that may reside in the gateway device.

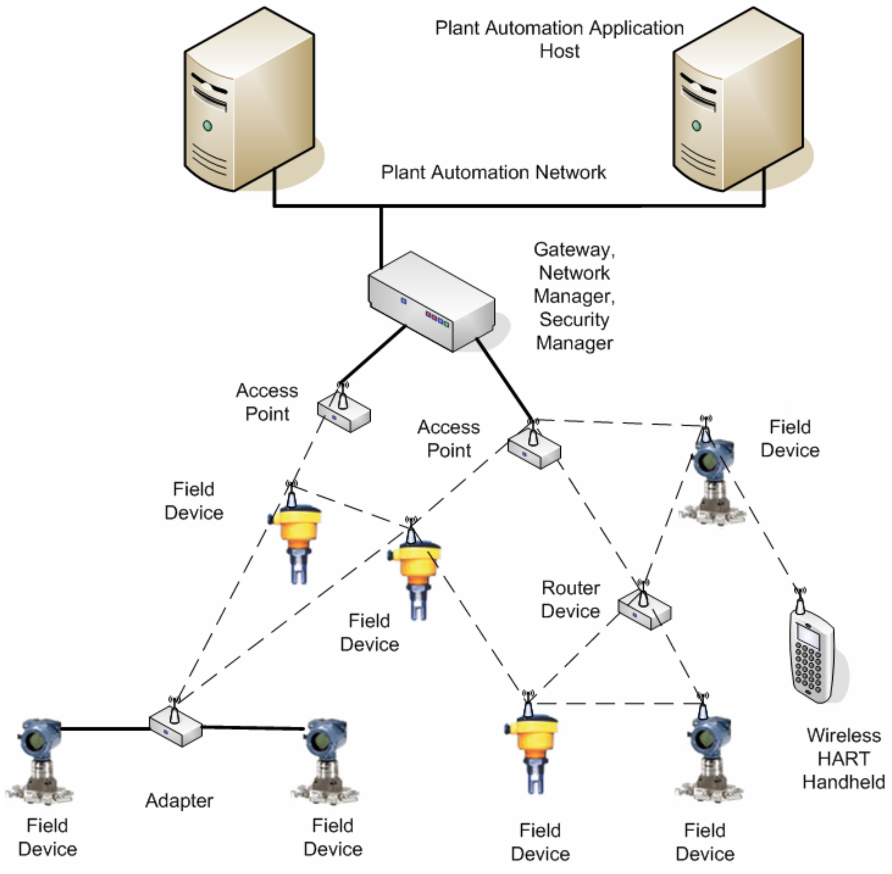


Fig. 1.2 Example of a WirelessHART Network.

In WirelessHART protocol communications are precisely scheduled using an approach referred to as Time Division Multiple Access (TDMA). The vast majority of communications are directed along graph routes. Scheduling is performed by a centralized network manager which uses overall network routing information in combination with communication requirements that devices and applications have provided. The schedule is subdivided into slots and transferred from the network manager to individual devices; devices are only provided with the slots for which they have communication requirements. The network manager continuously adapts the overall network graph and network schedule to changes in network topology and communication demand. The relationship between the components and within the WirelessHART communication stack is shown in Fig. 1.3.

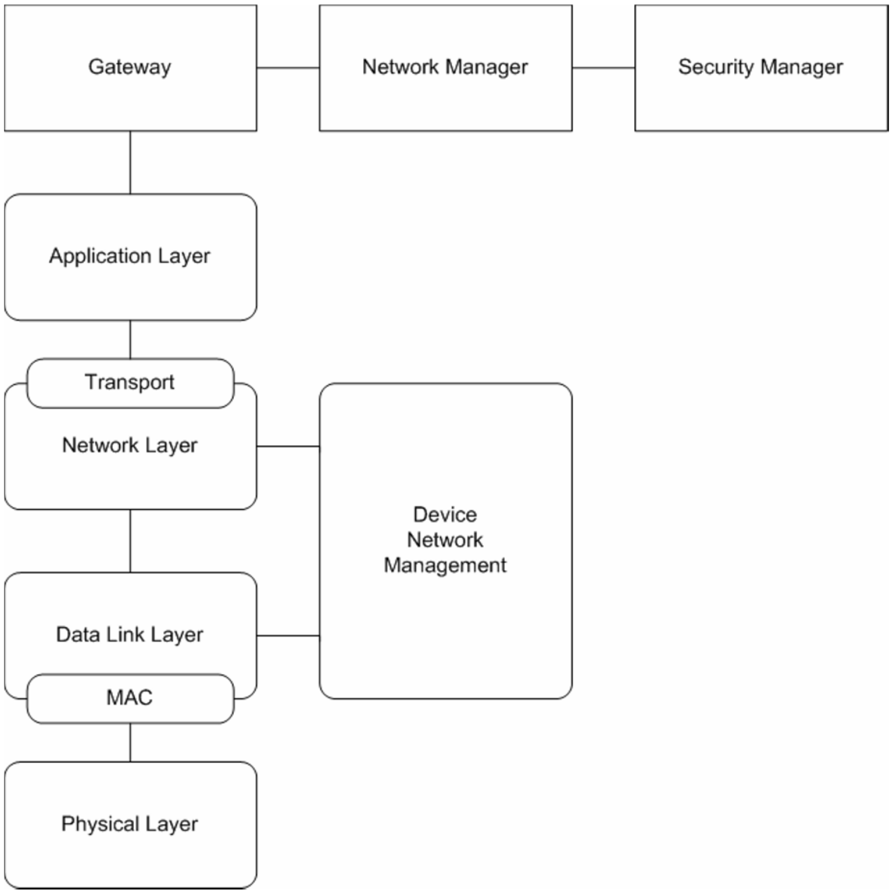
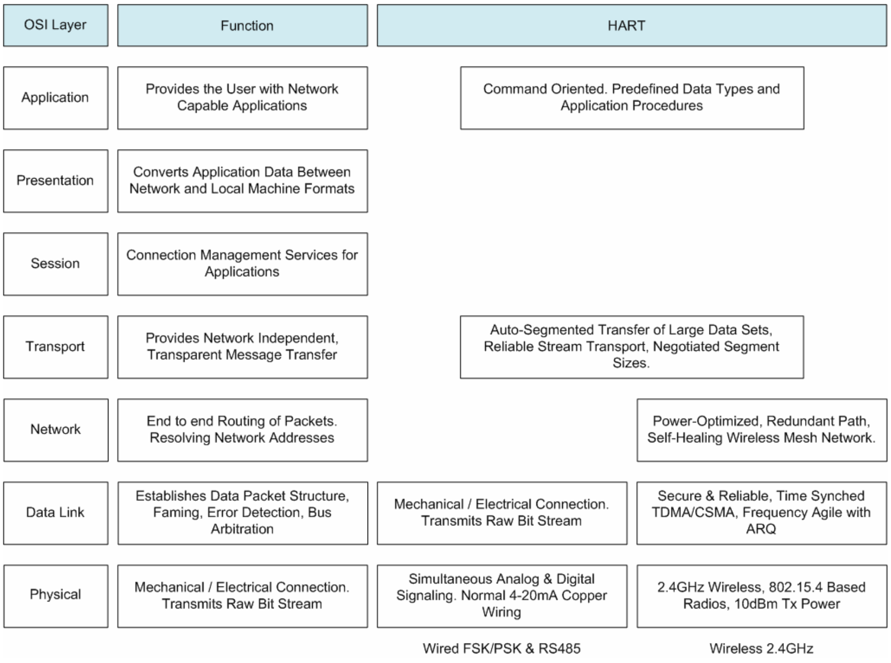


Fig. 1.3 WirelessHART Communication Stack.

The HART Foundation also extends the HART Device Registration Program to include WirelessHART devices. The WirelessHART Device Registration Procedure document outlines the testing and registration requirements for WirelessHART devices. Testing and registration of a device ensures the quality of the Hart device and the interoperability of all devices that claim HART compliance. Similarly, the requirements the WirelessHART Device Registration Procedure sets forth assure the interoperability of wireless devices in a multi-vendor environment and the conformance of those devices to the requirements of the HART Communication Protocol Specification. Devices that successfully pass the registration requirements are allowed to carry the *HART Registered* mark.

1.3 The Layers

Fig. 1.4 illustrates the architecture of the WirelessHART protocol stack according to the OSI 7-layer communication model. As shown in this figure, WirelessHART protocol stack includes five layers: physical layer, data link layer, network layer, transport layer and application layer. In addition, a central network manager is responsible for overall network routing communication scheduling (Song 2008).



OSI Layer	Function	HART	
Application	Provides the User with Network Capable Applications	Command Oriented. Predefined Data Types and Application Procedures	
Presentation	Converts Application Data Between Network and Local Machine Formats		
Session	Connection Management Services for Applications		
Transport	Provides Network Independent, Transparent Message Transfer	Auto-Segmented Transfer of Large Data Sets, Reliable Stream Transport, Negotiated Segment Sizes.	
Network	End to end Routing of Packets. Resolving Network Addresses		Power-Optimized, Redundant Path, Self-Healing Wireless Mesh Network.
Data Link	Establishes Data Packet Structure, Faming, Error Detection, Bus Arbitration	Mechanical / Electrical Connection. Transmits Raw Bit Stream	Secure & Reliable, Time Synched TDMA/CSMA, Frequency Agile with ARQ
Physical	Mechanical / Electrical Connection. Transmits Raw Bit Stream	Simultaneous Analog & Digital Signaling. Normal 4-20mA Copper Wiring	2.4GHz Wireless, 802.15.4 Based Radios, 10dBm Tx Power
		Wired FSK/PSK & RS485	Wireless 2.4GHz

Fig. 1.4 Architecture of HART Communication Protocol.

Physical layer

The WirelessHART physical layer is based mostly on the IEEE 802.15.4-2006 2.4GHz DSSS physical layer. This layer defines radio characteristics, such as the signaling method, signal strength, and device sensitivity. Just as the IEEE 802.15.4 protocol, the WirelessHART protocol operates in the 2400-2483.5MHz license-free ISM band with a data rate of up to 250 kbits/s. Its channels are numbered from 11 to 26, with a 5MHz gap between two adjacent channels.

Data Link Layer

One distinct feature of the WirelessHART standard is the time synchronized data link layer. It defines a strict 10ms timeslot and utilizes TDMA technology to provide collision free and deterministic communications. The concept of superframe is introduced to group a sequence of consecutive time slots. A superframe is periodical, with the total length of the member slots as the period. All superframes in a WirelessHART network start from the ASN (absolution slot number) 0, the time when the network is first created. Each superframe then repeats itself along the time based on its period. In the WirelessHART protocol, a transaction in a time slot is described by a vector: {*frame id*, *index*, *type*, *src addr*, *dst addr*, *channel offset*} where *frame id* identifies the specific superframe; *index* is the index of the slot in the superframe; *type* indicates the type of the slot (transmit/receive/idle); *src addr* and *dst addr* are the addresses of the source device and destination device, respectively; *channel offset* provides the logical channel to be used in the transaction. To fine-tune the channel usage, the WirelessHART standard introduces the idea of channel blacklisting. Channels affected by consistent interferences could be put in the blacklist. In this way, the network administrator can disable the use of those channels in the blacklist totally. To support channel hopping, each device maintains an active channel table. Due to channel blacklisting, the table may have less than 16 entries. For a given slot and channel offset, the actual channel is determined from the formula:

$$\text{ActualChannel} = (\text{ChannelOffset} + \text{ASN}) \ \% \ \text{NumChannels}$$

The actual channel number is used as an index into the active channel table to get the physical channel number. Since the ASN is increasing constantly, the same channel offset may be mapped to different physical channels in different slots. Thus we provide channel diversity and enhance the communication reliability. Fig. 1.5 describes the overall design of the data link layer which consists of six major modules as described in the following.

Interfaces

The interface between the MAC and physical layer describes the service primitives provided by the physical layer, and the interface between the MAC and network layer defines the service primitives provided to the network layer.

Timer

Timer is a fundamental module in the WirelessHART standard. It provides accurate timing to ensure the correct operation of the system. One significant challenge is how to design the timer module and keep those 10ms timeslots in synchronization.

Communication Tables

Each network device maintains a collection of tables in the data link layer. The superframe table and link table store communication configurations created by the network manager; the neighbor table is a list of neighbor nodes that the device can reach directly; and the graph table is used to collaborate with the network layer and record routing information.

Link Scheduler

The functionality of the link scheduler is to determine the next timeslot to be serviced based on the communication schedule in the superframe table and link table. The scheduler is complicated by such factors as transaction priorities, the link changes, and the enabling and disabling of superframes. Every event that can affect link scheduling will cause the link schedule to be re-assessed.

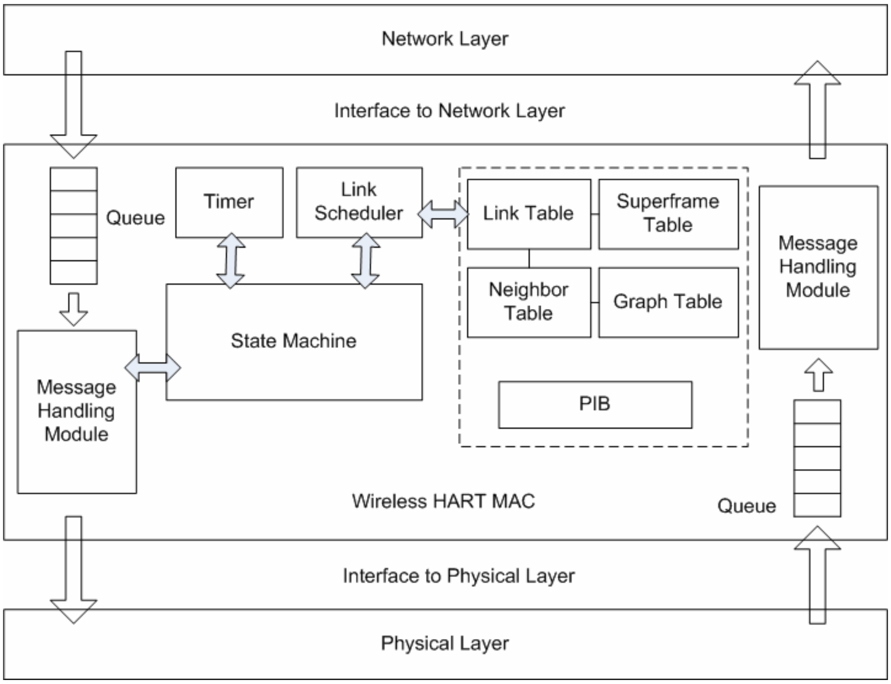


Fig. 1.5 WirelessHART Data Link Layer Architecture.

Message Handling Module

The message handling module buffers the packets from the network layer and physical layer separately.

State Machine

The state machine in the data link layer consists of three primary components: the TDMA state machine, the XMIT and RECV engines. The TDMA state machine is responsible for executing the transaction in a slot and adjusting the timer clock. The XMIT and RECV engines deal with the hardware directly, which send and receive a packet over the transceiver, respectively.

Network Layer and Transport Layer

The network layer and transport layer cooperate to provide secure and reliable end-to-end communication for network devices. Fig. 1.6 describes the overall design of the network and transport layer.

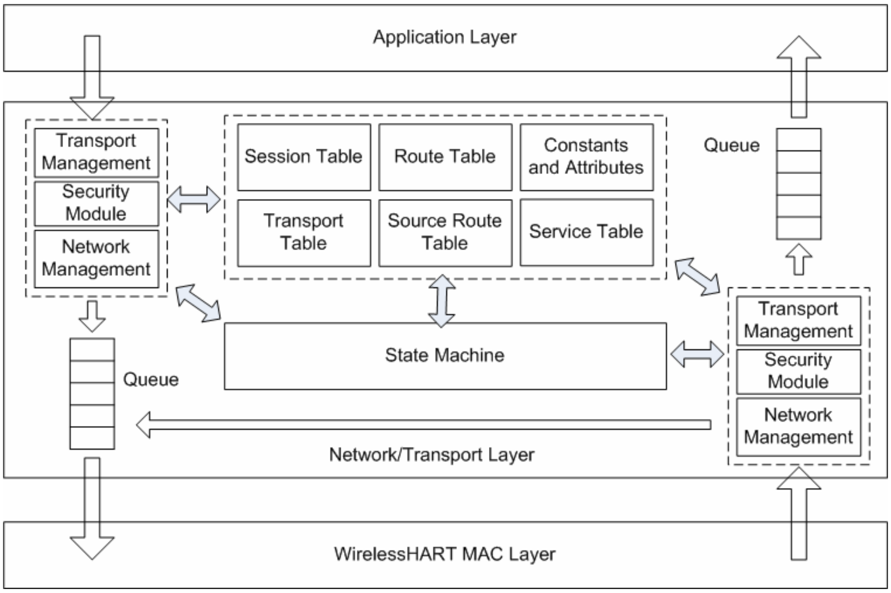


Fig. 1.6 WirelessHART Network Layer Architecture.

The basic elements of a typical WirelessHART network include: (1) Field Devices that are attached to the plant process, (2) Handheld which is a portable WirelessHART-enabled computer used to configure devices, run diagnostics, and perform calibrations, (3) A gateway that connects host applications with field devices, and (4) A network manager that is responsible for configuring the network, scheduling and managing communication between WirelessHART devices. To support the mesh communication technology, each WirelessHART device is required to be able to forward packets on behalf of other devices. There are three routing protocols defined in the WirelessHART standard:

Graph Routing

A graph is a collection of paths that connect network nodes. The paths in each graph is explicitly created by the network manager and downloaded to each individual network device. To send a packet, the source device writes a specific graph ID (determined by the destination) in the network header. All network devices on the way to the destination must be pre-configured with graph information that specifies the neighbors to which the packets may be forwarded.

Source Routing

Source routing is a supplement of the graph routing aiming at network diagnostics. To send a packet to its destination, the source device includes in the header an ordered list of devices through which the packet must travel. As the packet is routed, each routing device utilizes the next network device address in the list to determine the next hop until the destination device is reached.

Superframe Routing

Superframe routing is a special case of graph routing. In superframe routing packets are assigned to a superframe. Packets are instructed to follow the superframe en-route from the source towards the destination. With superframe routing the Graph ID is set to the Superframe ID. Since the packet follows the superframe, it is not necessary to explicitly configure graph edges.

Application Layer

Fig. 1.7 describes the overall design of the application layer. The application layer is the topmost layer in the WirelessHART standard. It defines various device commands, responses, data types and status reporting. In the WirelessHART standard, the communication between the devices and gateway is based on commands and responses. The application layer is responsible for parsing the message content, extracting the command number, executing the specified command, and generating responses.

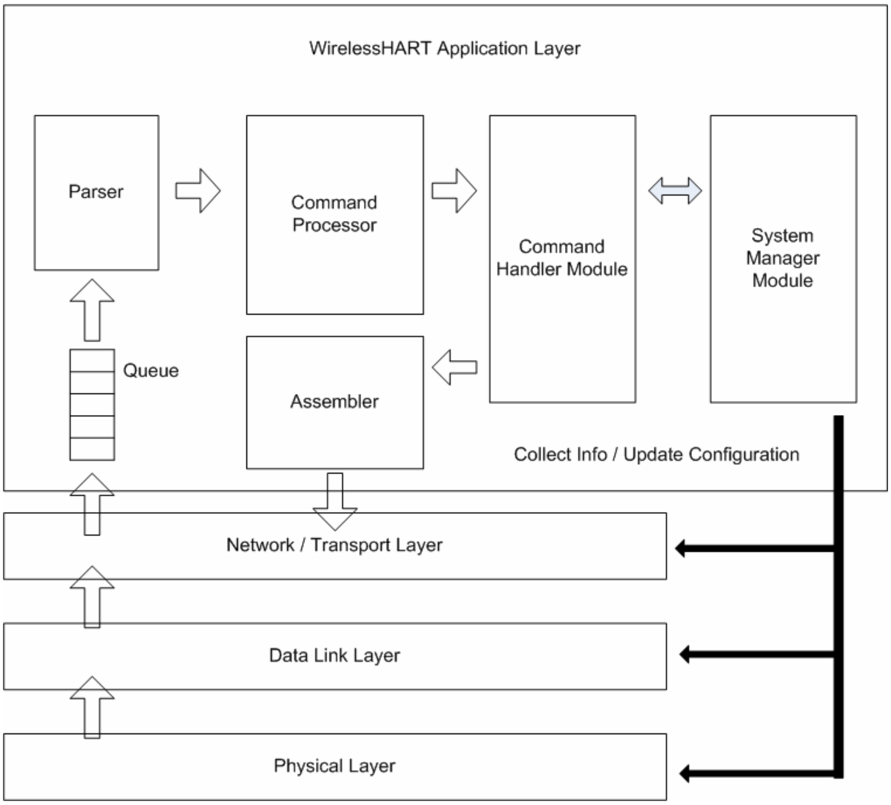


Fig. 1.7 WirelessHART Application Layer Architecture.

Security Architecture

A WirelessHART network is a secure network system. Both the MAC layer and network layer provide security services. The MAC layer provides hop-to-hop data integrity by using a combination of a cyclic redundancy check (CRC) and a Message Integrity Code (MIC). Although the CRC has limited value it is still used. Both the sender and receiver use the CCM* mode together with AES-128 as the underlying block cipher to generate and compare the MIC. The network layer employs various keys to provide confidentiality and data integrity for end-to-end connections. Four types of keys are defined in the security architecture:

- One public key which is used to generate MICs on the MAC layer when network key is not applicable.
- Network keys which are shared by all network devices and used by existing devices in the network to generate MAC MIC's.
- Join keys that are unique to each network device and is used during the joining process to authenticate the joining device with the network manager.
- Session keys that are generated by the network manager and are unique for each end-to-end connection between two network devices. They provide end-to-end confidentiality and data integrity. Session keys are further differentiated by whether they are unicast or broadcast keys. Unicast and Broadcast keys are very different in how they store and use nonce counters.

Fig. 1.8 describes the usage of these keys under two different scenarios: 1) a new network device wants to join the network and 2) an existing network device is communicating with the network manager. In the first scenario, the joining device will use the public key to generate the MIC in the MAC layer header and use the join key to generate the network layer MIC and encrypt the join request. After the joining device is authenticated, the network manager will create a session key for the device and thus establish a secure session between them. In the second scenario, on the MAC layer, the DLPDU is authenticated with the network key; on the network layer, the packet is authenticated and encrypted by the session key.

1.4 A Simple Example

Here we describe a very simple WirelessHART network, which is demonstrated with our prototype stack on a FreeScale™ platform, written in ANSI C.

The demonstration network contains one gateway, which also acts as the access point, and two devices, which are referred to as Device 1 and Device 2. In this network, Device 1 has the Gateway as its time source, and is the time source of the Device 2. The Gateway and Device 2 both maintain a 4-bit counter, with the Gateway counting upward and Device 2 counting downward. They exchange the counter values via Device 1 and show the received value on the LEDs. The first two LEDs on Device 1 display the lower two bits of the Gateway's counter and the remaining two LEDs on Device 1 shows the lower two bits of Device 2's counter. In this way, we can check the communication status between any two devices simply from the LEDs.

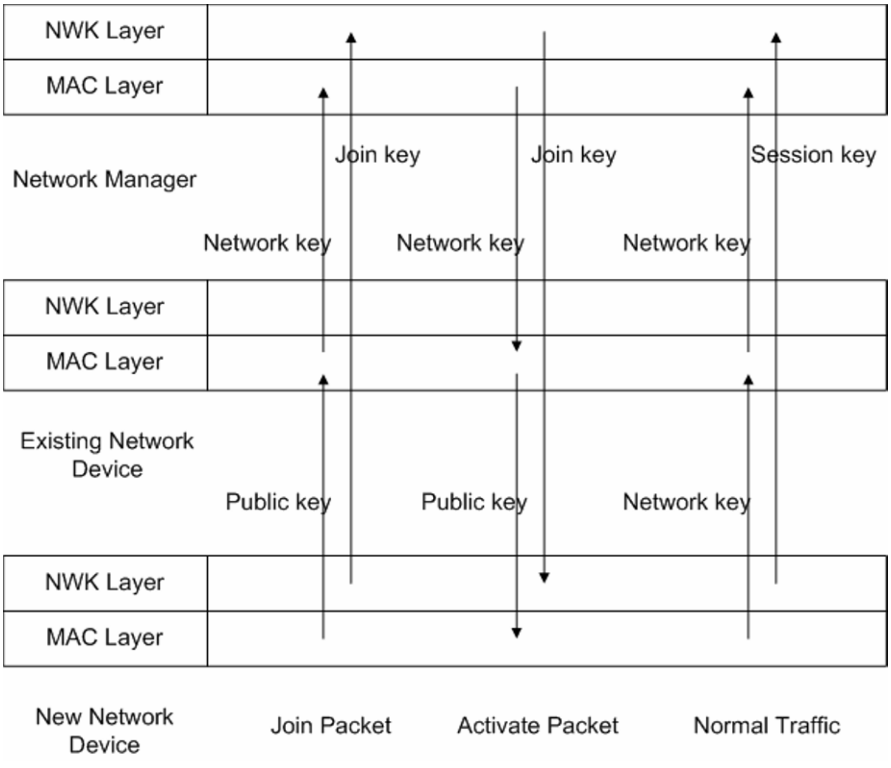


Fig. 1.8 Keying Model.

We also define an overall superframe for the network, shown in Table 1.1. Basically, this superframe regulates the time sequence of all communications. That is, the Gateway first transmits a packet to Device 2 via Device 1. Device 2 sends out its counter value after it receives the packet from the Gateway.

Table 1.1 The Superframe Configuration.

Time Slot	Link
0	Gateway -> Device 1
1	Device 1 -> Device 2
2	Device 2 -> Device 1
3	Device 1 -> Gateway

With the help of the WirelessHART package sniffer Wi-Analys™, we verify the following points.

- All two devices are synchronized to the network time source (the Gateway).
- A DLPDU is always immediately acknowledged in the same time slot.
- Both the Gateway and Device 2 get the other party's counter value via Device 1. That is, Device 1 is forwarding data packets for the two devices.

Since the WirelessHART standard works on top of IEEE 802.15.4 physical layer and adopts 802.15.4 DLPDU format, we can also capture the communication details on packet level by using any hardware sniffer and IEEE 802.15.4 protocol analyzer.

Chapter 2 Physical Layer

Abstract The physical Layer defines the electrical and physical level relationship between a node and a physical medium. It defines characteristics such as antenna, air medium, power level, timing of voltage changes, physical data rates, maximum transmission distances, etc. The WirelessHART standard is based on the IEEE 802.15.4 standard. WirelessHART physical layer is a much simplified subset of that defined in the IEEE 802.15.4 standard. This chapter lists the services that the WirelessHART physical layer provides to the upper data link layer. Please refer to the Wireless Physical Layer Specification (HCF_SPEC-65) for more details.

The WirelessHART standard builds on top of the IEEE 802.15.4 standard. With only a few modifications or restrictions, WirelessHART physical layer is a much simplified subset of that defined in IEEE 802.15.4-2006 Section 6. For any WirelessHART device:

- There are only one or two IEEE 802.15.4 messages per 10ms timeslot (broadcast messages are not acknowledged).
- The closest time between two messages is between the two messages within a timeslot, 1ms from the end of the message to the start of the acknowledgement message.
- All WirelessHART messages are IEEE 802.15.4 messages of *data* type.
- Only the 2.4GHz frequency band is defined for WirelessHART.
- Channels 11-25 can be used with the WirelessHART standard. Channel 26, which is not legal in many locales, is not supported.

In summary, the WirelessHART physical layer limits itself to transmitting and receiving IEEE 802.15.4 data messages. The noticeable items in WirelessHART physical layer are:

- *Channel hopping*. In WirelessHART physical channel is changed each transmission.
- *Transmit power*. The IEEE 802.15.4 standard is defined for personal area network with personal operating space of 10 meters. WirelessHART mesh covers a relatively larger area. All devices must provide a nominal EIRP of +10dBm (10mW) ±3dB. The transmit power is programmable from -10dBm to +10dBm. The maximum outdoor line of sight transmission distance could be 100 meters.

The targeted radio hardware component of a WirelessHART device is the commercial off the shelf chips designed for the IEEE 802.15.4 standard.

2.1 Physical Layer Services

This section lists the operation of the Service Primitive (SP).

2.1.1 Message SPs

Enabling/Disabling the Transceiver

This service primitive is used to enable/disable channels.

- ENABLE.request(state, channel)
- ENABLE.confirm(state, channel)
- ENABLE.indicate()

Clear Channel Assessment

This service primitive causes the Physical Layer to perform

- CCA.request()
- CCA.confirm(status)

Data Communication Services

These service primitives are used to exchange data packets with the Physical Layer.

- DATA.request(data)
- DATA.confirm(status, data)
- DATA.indication(rsl, data)
- ERROR.indication(status, data)

2.1.2 Management SPs

This service primitive is used to enable/disable channels.

- LOCAL_MANAGEMENT.request(service, [data])
- LOCAL_MANAGEMENT.confirm(service, status, [data])
- LOCAL_MANAGEMENT.request(service, status, [data])

  The first parameter is the requested service. The services are listed in Table 2.1.

Table 2.1 Local Device Management Commands.

Service	Data	Description
RESET	-	Initialize the physical layer
READ_TX_PWR_LEVEL	Signed-8 txPwrLevel	Read the transmit power level setting in dBm
WRITE_TX_PWR_LEVEL	Signed-8 txPwrLevel	Write the transmit power level setting in dBm
WRITE_SLEEP_STATE	Unsigned-8 slepState	Write the sleep state of the physical layer. Value is one of: {sleep awake}
WRITE_RCV_OVERFLOW _ENABLE	Boolean rcvOverflowEnable	Enable the receiver overflow error indication. This will be reported in the ERROR.indication() service status

Chapter 3 Data Link Layer

Abstract The data link layer provides the reliable means to transfer data between network nodes by detecting and possibly correcting errors that may occur in the physical layer. This layer has the important task of creating and managing data frames. There are usually two sublayers, Logical Link Control (LLC) layer and Medium Access Control (MAC) layer. LLC layer defines the service to the network layer and MAC defines how the communication medium is accessed by multiple nodes. This chapter lists the services that the WirelessHART data link layer provides to the upper network layer. It further describes in more detail the LLC and MAC layers. We describe the message format and its flow control, error detection, and security. We also describe how to maintain slot synchronization with TDMA, schedule slots to listen for packets from neighbors and propagate packets from the network layer. Please refer to the TDMA Data Link Layer Specification (HCF_SPEC-75) for more details.

Fig. 3.1 shows the scope of the data link layer.

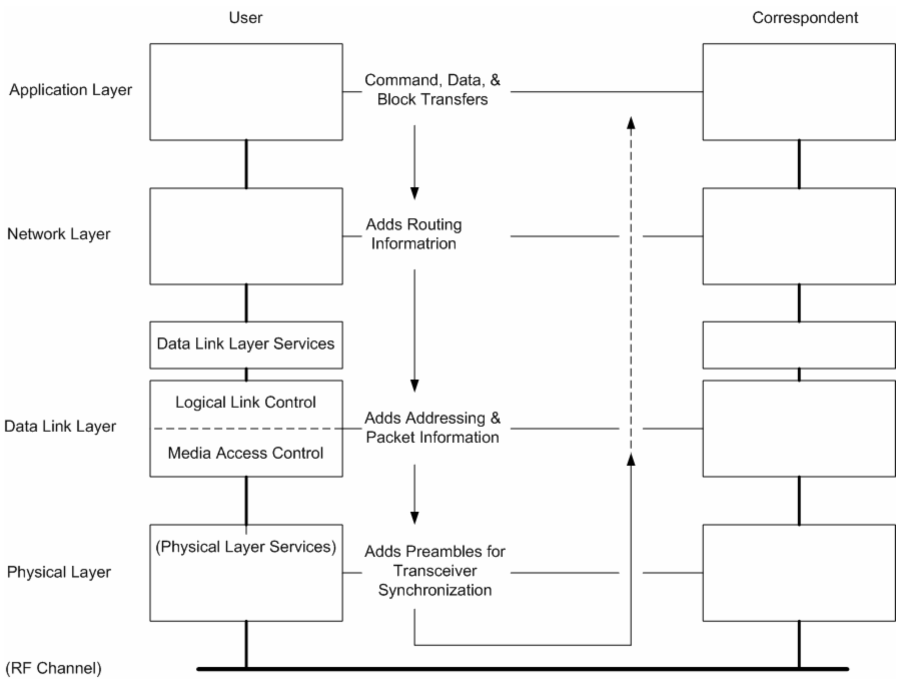

User
Correspondent
Application Layer
Command, Data, & Block Transfers
Network Layer
Adds Routing Informatrion
Data Link Layer Services
Data Link Layer
Logical Link Control
Media Access Control
Adds Addressing & Packet Information
Physical Layer
(Physical Layer Services)
Adds Preambles for Transceiver Synchronization
(RF Channel)

Fig. 3.1 Data-Link Layer Scope.

The data link layer specification includes:

- The services provided by the Data-Link Layer to the Network Layer. These services constitute a *black box* model of the Data-Link Layer requirements.
- Logical Link Control (LLC) requirements including the format of HART frames, the structure of HART device addresses; the security services used for message integrity and the error detection coding to be used.
- Media Access Control (MAC) rules ensuring that transmissions by devices occur in an orderly fashion. In other words, the MAC specifies when a device is allowed to transmit a message.
- The actual timing values required for proper operation of the MAC sub-layer. These timing values directly correspond to Physical Layer performance characteristics (e.g., Clear Channel Assessment time, Tx/Rx turnaround time).

3.1 Data Link Layer Services

3.1.1 Message SPs

Message SPs provide services supporting the basic transfer of data between devices. The data link layer must also allow multiple messages to be queued. The protocol also supports automatic retries to ensure accurate data exchange.

Transmit SPs

- FLUSH.request (handle)
- FLUSH.confirm (handle, localStatus)
- TRANSMIT.confirm (handle, localStatus)
- TRANSMIT.indicate (localStatus, priority, sourceAddress, payload)
- TRANSMIT.request (handle, payload, priority, timeout, graph)
- TRANSMIT.request (handle, payload, priority, timeout, sframe, bcast)
- TRANSMIT.request (handle, payload, priority, timeout, shortDestAddress)
- TRANSMIT.request (handle, payload, priority, timeout, longDestAddress)

The parameters to the overloaded TRANSMIT.request include:

- *handle* - The handle is supported for the convenience of the client layer. The Data-Link returns this value in the corresponding TRANSMIT.confirm.
- *payload* - The NPDU to be propagated to the destination device.
- *priority* - The packet priority as determined by the contents of the payload, from the set: {management, process-data, normal, alarm}.

- *timeout* - The maximum time to attempt packet transmission. The Network Layer should set this based on the ASN Snippet (see the Network Management Specification).
- *graph* - This parameter is only present if graph routing is to be employed. When employed, the graph indicates the neighbors that may be used as the destination for the next hop.
- *sframe* - This parameter is only present if broadcasting a message. sframe indicates the superframe whose broadcast links can be used to forward the packet.
- *bcast* - This flag indicates the NPDU must be broadcast on the indicated Superframe.
- *shortDestAddress* - This parameter indicates the nickname of the destination device that must be used for the next hop
- *longDestAddress* - This parameter indicates the Unique ID of the destination device that must be used for the next hop.

Network Event SPs

- DISCONNECT.indicate (localStatus, sourceAddress)
- PATH_FAILURE.indicate (localStatus, sourceAddress)
- ADVERTISE.indicate (localStatus, AdvertisePayload)
- NEIGHBOR.indicate (localStatus, sourceAddress, packetRSL)

Receive SPs

- RECEIVE.indicate (localStatus, packetRSL, payloadDLPDU)

3.1.2 Management SPs

Management SPs support both configuration of the data link layer and access to data link layer statistics.

- LOCAL_MANAGEMENT.request (service, [data])
- LOCAL_MANAGEMENT.confirm (service, status, [data])
- LOCAL_MANAGEMENT.indication (service, status, [data])

The first parameter is the requested service. The services are listed in Table 3.1.

Table 3.1 Local Device Management Commands.

Service	Description
RESET	Initializes the Data-Link Layer.
DISCONNECT	Disconnect from the network, cease communications
RE_JOIN	Disconnect from the network, rejoin the network, purging all

	MAC queues and clearing all MAC tables
WRITE_SUPERFRAME	Creates a new superframe.
DELETE_SUPERFRAME	Deletes an existing scheduling superframe and any associated links.
ADD_LINK	Adds a new link to another device, possibly updating the neighbor and connection tables in the process.
DELETE_LINK	Deletes an existing link, possibly updating the neighbor and connection tables in the process.
ADD_EDGE	Adds a new neighbor to the specified graph
DELETE_EDGE	Deletes a neighbor from a graph
READ_NETWORKID	Reads the ID of the network the device belongs to.
WRITE_NETWORKID	Writes the ID of the network the device belongs to.
WRITE_NETWORK_KEY	This command allows the Network Manager to write the network key on a Network Device. Keys should be protected from pilfering (e.g., by encryption)
READ_TIMEOUT _PERIODS	Read the time period values, Keep-Alive, Path-Failure, Advertise, and Discovery.
WRITE_TIMEOUT _PERIOD	Write the indicated time period value.
READ_CAPACITIES	Read maxSuperframes, maxLinks, maxNeighbors, and maxPktBuffers
READ_PRIORITY _THRESHOLD	Read the lowest priority DLPDU to be accepted from another device.
WRITE_PRIORITY _THRESHOLD	Write the lowest priority DLPDU to be accepted from another device.
READ_JOIN_PRIORITY	Read what join priority the device should advertise.
WRITE_JOIN_PRIORITY	Write what join priority the device should advertise.
READ_PROMISCUOUS _MODE	Read whether the sublayer is in “receive all” mode.
WRITE_PROMISCUOUS _MODE	Write whether the sublayer is in “receive all” mode.
READ_MAX_BACK_OFF _EXPONENT	Read the maximum value that can be assumed for the back-off exponent used in shared slots.
WRITE_MAX_BACK _OFF_EXPONENT	Write the maximum value that can be assumed for the back-off exponent used in shared slots.

3.2 Logical Link Control

3.2.1 The DLPDU

Each Data-Link packet (DLPDU) consists of the following fields:

- A single byte set to 0x41
- A 1-byte address specifier;
- The 1-byte Sequence Number;
- The 2 byte Network ID
- Destination and Source Addresses either of which can be 2 or 8-bytes long;
- A 1-byte DLPDU Specifier
- The DLL payload
- A 4-byte keyed Message Integrity Code (MIC), and
- A 2-byte ITU-T CRC16

Fig. 3.2 illustrates the basic PhPDU and DLPDU structure.

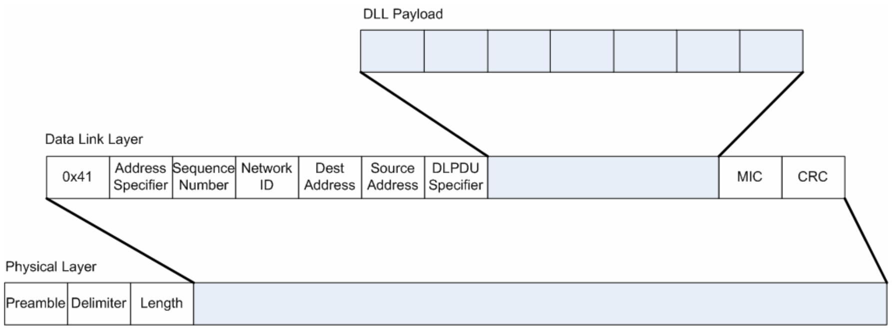


Fig. 3.2 DLPDU Structure.

DLPDU Specifier

The DLPDU specifier is defined in Fig. 3.3. It specifies the priority and type of the message, and whether the network key or the public key is used to authenticate the message.

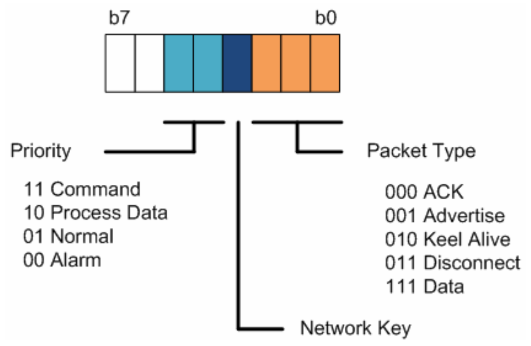


Fig. 3.3 DLPDU Specifier.

Keyed Message Integrity Code (MIC)

A keyed Message Integrity Code (MIC) is used for link-layer authentication of DLPDU. Devices shall reply only to unicast, non-acknowledgement DLPDUs that have been successfully authenticated.

3.2.2 DLPDU Types

There are five DLPDU types as specified in the DLPDU specifier:

- Data DLPDUs contain network and device data in transit to their final destination device.
- Keep-Alive DLPDUs facilitate connection maintenance between neighboring devices.
- Advertise DLPDUs provide information to neighboring devices wishing to join the network.
- Disconnect DLPDUs are used to advise neighboring devices that the device is leaving the network.
- ACK DLPDUs are the immediate link level response to receipt of the source device's transmission DLPDU.

3.2.3 DLPDU Priority and Flow Control

Four priority levels as specified in the DLPDU Specifier. Network management packets with highest command priority always propagate through the network allowing the Network Manager to keep the network operational. The flow of alarms through the network is restricted ensuring alarm floods do not disrupt network operation. Since alarms are always time-stamped, no information is lost regarding, for example, failure sequences.

Finally, all other network traffic flows through the network as buffer space and bandwidth allows. Within this network traffic, process data has priority. Operation and control of the process is second only to preventing network communication disruption.

3.2.4 Error Detection Coding and Security

A keyed MIC is used to ensure that the DLPDU originated from an approved, authenticated device. The DLPDU itself is not enciphered; rather its contents are authenticated using the MIC, which is calculated using the CCM* algorithm described in the IEEE 802.15.4 standard. This ensures that the hardware accelerator in the commercial off the shelf chips could be employed.

There are two DLL keys: the well-known key (used in advertisements and when joining the network), and the network key (used for all other transactions). The well-known key is identical for all WirelessHART devices and has a value of 7777 772E 6861 7274 636F 6D6D 2E6F 7267 hexadecimal. The well-known key is used for messages passed between the joining device and devices already part of the network.

The well-known public key is the ASCII value sequence of the 16 character string of the HART Foundation’s web address: *www.hartcomm.org*.

3.3 Media Access Control

The primary objectives of the MAC sublayer are to maintain slot synchronization, identify slots that must be serviced, listen for packets being propagated from neighbors and, in turn, propagate packets received from the network layer.

3.3.1 Slot Timing

All transactions occur in slots following specific timing requirements. Fig. 3.4 shows one slot and provides an overview of transaction timing.

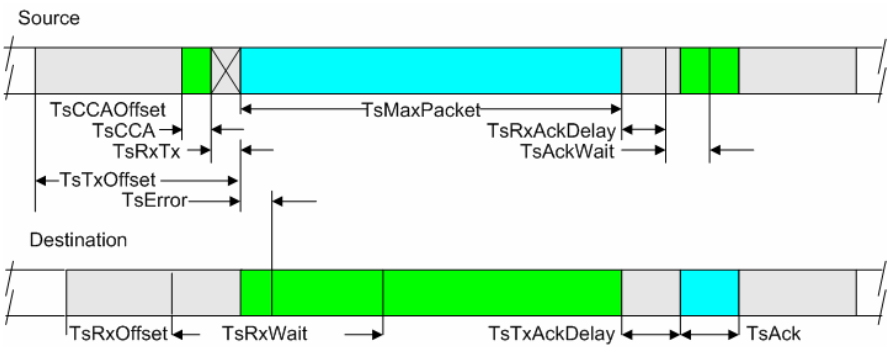


Fig. 3.4 Slot Timing.

Table 3.2 Slot Timing Symbols.

Symbol	Description
TsTxOffset	Start of the slot to start of preamble transmission.
TsRxOffset	Start of the slot to when transceiver must be listening.
TsRxWait	The minimum time to wait for start of message. This correlates to the amount of drift between the neighbors that can be tolerated and communications still be maintained.
TsError	This is the difference between the actual start of message and the ideal start of message time as perceived by the receiving device. In other words, this is how much the receiving device perceives the transmitting device to be out of sync.
TsMaxPacket	The amount of time it takes to transmit the longest possible message (includes PhL preamble, delimiter, length and DLPDU
TsTxAckDelay	End of message to start of ACK. The destination device must validate the STX, and generate an ACK during this interval. Note: Broadcast messages are not acknowledged.
TsRxAckDelay	End of message to when transceiver must be listening for ACK.
TsAckWait	The minimum time to wait for the start of an ACK.
TsAck	Time to transmit an ACK.
TsCCAOffset	Start of slot to beginning of CCA.
TsCCA	Time to perform CCA.
TsRxTx	The longer of the time it takes to switch from receive to transmit or vice versa.

3.3.2 Communication Tables and Buffers

All devices maintain a series of tables that control the communications performed by the device and collect statistics on those communications. In addition, packets are buffered as messages are received, processed and forwarded.

The tables controlling communication activities include:

- The superframe table. Multiple superframes may be configured by the network manager.
- The link table. Multiple links within a superframe are configured to specify communication with a specific neighbor or broadcast communications to all listening to the link.
- The neighbor table. The neighbor table is a list of all devices that the device may be able to communicate with.
- The graph table. Graphs are used to route messages from their source to their destination. The device does not know the entire route rather, the graph indicates the next hop destinations legal for propagating the packet onward toward its destination.

In addition to these tables, there is a packet queue that buffers messages.

3.3.3 Link Scheduling

All devices must maintain a link schedule that identifies the next slot that must be serviced. Servicing the slot consists of either listening for a new packet or propagating a packet onward through the mesh. When a slot has both a packet waiting to be propagated and receive links, propagating the packet has priority over attempting to listen for a new packet.

While, on the surface, link scheduling seems straightforward, it is complicated by transaction priorities, the modification of links, and the enabling and disabling of superframes. Each event that affects link scheduling may result in widespread reassignment of transmit links. For example, if a high priority transaction fails transmission then it must be rescheduled. Consequently, lower priority transactions may need to be deferred to a later link and their current link given up to the higher priority transaction. Effects can be even more widespread, for example, should a superframe be disabled or even deleted.

Chapter 4 Network Layer and Transport Layer

Abstract In the ISO™ OSI™ 7 level network design the network layer is responsible for network routing functions. It handles the addressing and delivery of data. The transport layer controls the reliable and timely transmission of data between two network nodes through flow control, segmentation and desegmentation, and error control. The session layer manages the dialogue, session, and connections between two network nodes. In WirelessHART standard the network layer encompasses all three OSI layers. It is the point of convergence for traditional HART Token-Passing Networks and WirelessHART TDMA-based networks. This chapter lists the services that the network layer provides to the upper application link layer. It talks about the communication traffic, routing, and security. It also describes the format of the network layer PDU. Please refer to the Network Management Specification (HCF_SPEC-85) for more details.

4.1 Overview

Fig. 4.1 shows the focus of this chapter. The thin transport layer is specified as part of the network layer that also manages the sessions. Above this resides the HART application layer that defines allowed HART data types, procedures and commands. Below this reside the two major HART disciplines: wired Token-Passing networks, and TDMA wireless communication technologies. This chapter covers the wireless part.

4.1.1 Communication Traffic

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. 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.

- Block Data Transfers establish a pipe between two nodes that allow data to be streamed between them.

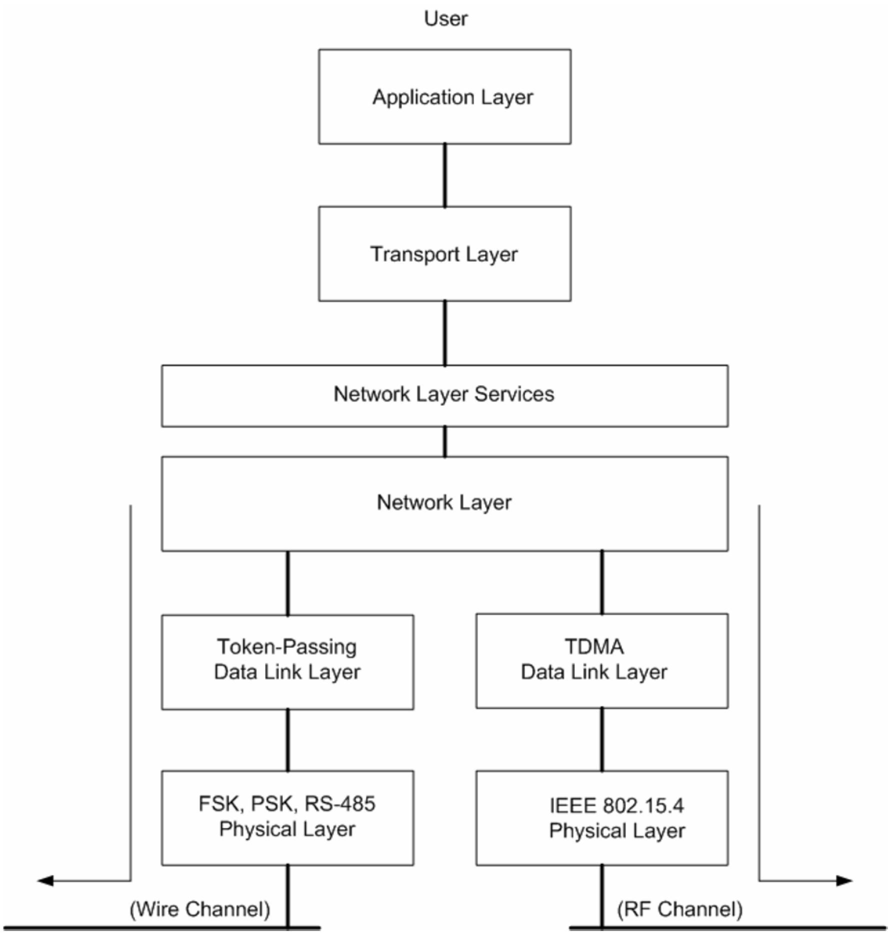


Fig. 4.1 Network Layer Scope.

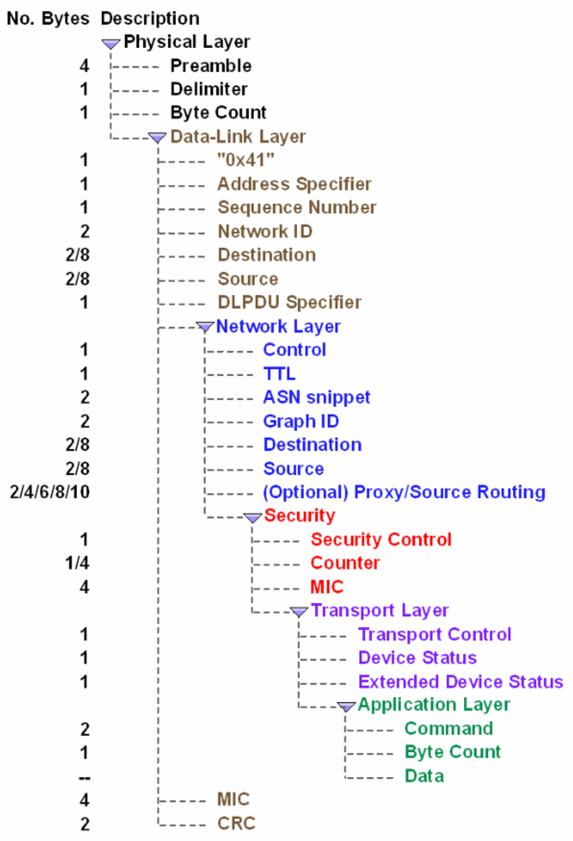
Fig.4.2 is the layout of the WirelessHART message.

4.1.2 Routing

The WirelessHART network layer supports Graph and Source routing.

- **Graph**. 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 actual route 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.



```
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
```

Fig. 4.2 Summary of PDU Format.

4.1.3 Security

End-to-end communications are managed on the Network Layer by sessions. A device may have more than one session defined for a given peer device. 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. Sessions can be established between any two devices in the network.

4.2 Network Layer Services

4.2.1 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.

- TRANSMIT.request(handle, dest, priority, timetableID, transportType, payload)

  The parameters included with the service request include:

- *handle* - 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.
    - uniqueID – the long adress of the destination device
    - nickname – the short address of the destination device
    - broadcast – the broadcast address is indicated in the NPDU and the control byte indicates 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.
- *transportType* - The transport type (see Table 4.1) indicates the Transport Layer operation requested and is used to set the Transport Byte.
- *payload* - The contents of payload parameter is transmitted to the destination device.

Table 4.1 Transport Type Codes.

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.response only). This is a broadcast announcement to all network devices. (e.g., time broadcast)	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.response only). (e.g., process data)	-	-	←Rsp

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

- TRANSMIT.indicate (handle, srcAddr, priority, transportType, payload)
- TRANSMIT.response (handle, payload)
- TRANSMIT.confirm (handle, localStatus, [payload])
- FLUSH.request (handle)
- FLUSH.confirm (handle, localStatus)

4.2.2 Network Layer Management Services

Management SPs support both configuration of the network layer and access to statistics that it gathers.

- LOCAL_MANAGEMENT.request(service, [data])
- LOCAL_MANAGEMENT.confirm(service, status, [data])

- LOCAL_MANAGEMENT.indication(service, status, [data])
  The first parameter is the requested service. The services are listed in Table 4.2.

Table 4.2 Local Device Management Commands.

Service	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.
DEL_SESSION	delete a session
ADD_ROUTE	add route to a given destination address..
DEL_ROUTE	delete route information
DEFAULT_ROUTE	set given route as default
READ_PDU_TIMEOUT	Reads the number of slots since packet's birth until it is discarded. Device checks ASN Snippet against current ASN
WRITE_PDU_TIMEOUT	Writes the number of slots since packet's birth until it is discarded. Device checks ASN Snippet against current ASN
READ_TTL	Read TTL. The value TTL is initialized to when a new packet is generated.
WRITE_TTL	Write TTL.

4.3 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.

4.3.1 Network Layer PDUs

As shown in Fig. 4.3, 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 the private communication between the NPDU's end points has not been interfered with. Finally, the NPDU payload is enciphered and contains the information being exchanged across the network.

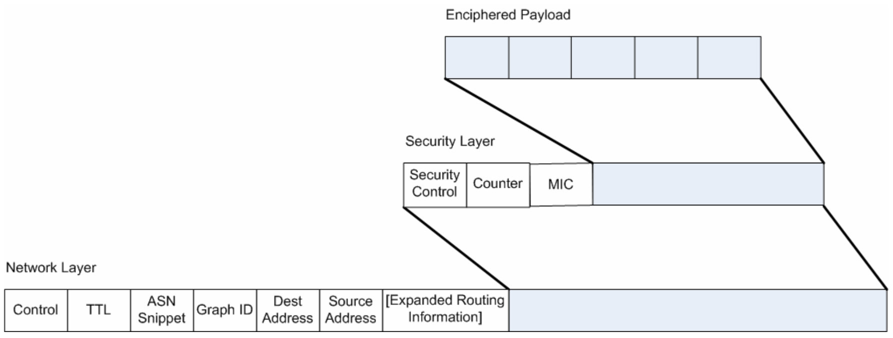


Fig. 4.3 WirelessHART NPDU Structure.

Collectively these three elements comprise the NPDU.

4.3.1.1 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.

Control Byte

The first byte in the Network PDU is the Control byte (see Fig. 4.4). 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.

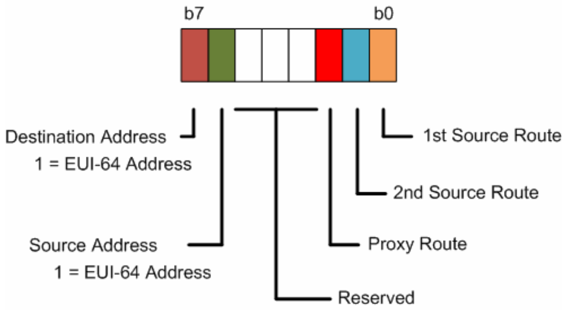


Fig. 4.4 Network Control Byte.

Bits 2-0 indicate, when set, the presence of the optional routing fields. 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.

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 is not decremented and the packet is always forwarded onward toward its final destination (i.e., TTL is infinite).

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.

If the TTL is valid then the ASN corresponding to the packet's birthday is compared to the current ASN to get the age of the packet. If the age is greater than the maxPacketAge the packet is discarded. The result of this comparison shall be provided as the timeout to the Data-Link.

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.

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.

4.3.1.2 Security Sub-Layer

The security layer header of the NPDU (see Fig. 4.3) is designed to: ensure private, unmolested communication; and allow for future possible enhancements to WirelessHART security.

As indicated in Fig. 4.5, the security control byte consists of a 4 bit enumeration (bits 0-3) that indicates the security strategy employed for this NPDU (See Common Table 53 in Specification HCF_SPEC-183). The most significant bits (bite 4-7) of the security control byte are reserved.

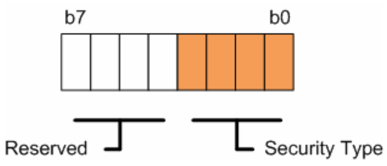


Fig. 4.5 Security Control Byte.

The Counter is the nonce counter used for running the encryption algorithm.

4.3.1.3 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.

4.3.2 Transport Layer PDU

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.

Each transport layer PDU (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

Fig. 4.6 illustrates the basic TPDU structure.

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.

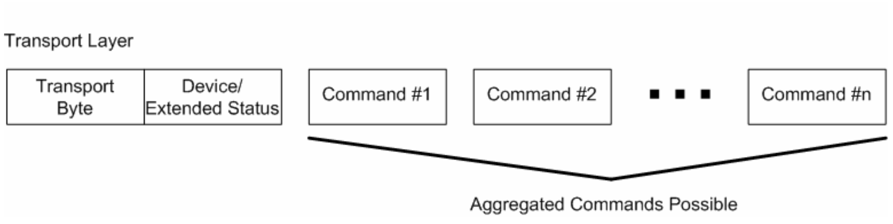


Fig. 4.6 Transport Layer.

Transport Byte

The first byte in the TPDU is the transport byte (see Fig. 4.7).

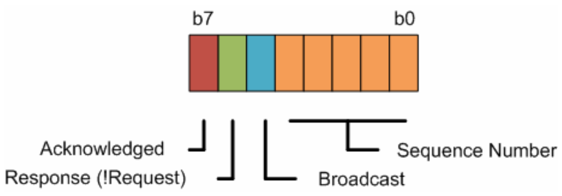


Fig. 4.7 Transport Byte.

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.

Aggregated Commands

With some limitations, the WirelessHART standard allows multiple HART commands to be transported in a single transaction. The format of commands transported over the WirelessHART network is shown in Fig. 4.8.

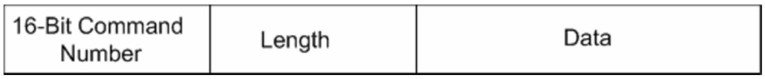


Fig. 4.8 WirelessHART Command Format.

Chapter 5 Application Layer

Abstract In the OSI network layer design the application layer is the closest layer to the end user. It provides a means for an application to access information on the network. Its functions typically include identifying communication partners, determining resource availability, and synchronizing communication. In the WirelessHART application layer, commands are the building blocks for communication. This chapter lists the services that the WirelessHART application layer provides to the host applications. It describes the dynamic and device variables, and the four host conformance classes.

The basic application layer is defined in Command Summary Specification (HCF_SPEC-99), which specifies:

- The allowable formats of data transmitted via the protocol;
- Revision rules for all field devices;
- The allocation of Commands Numbers for use by Universal, Common Practice, Device-Specific, and Device Family commands;
- The requirements for the construction of any HART command;
- The Command Status Bytes required to be returned with all command responses;
- The use of Dynamic and Field Device Variables;
- Procedures used by masters to identify field devices and manage HART networks.

These requirements provide the basis for all HART application Layer Specifications.

5.1 Application Layer Interface

The Application Layer is command based. In other words, commands from master or slave devices are the basis for HART communication and the command number, embedded in the communication, determines the content of the messages. The command number indicates the specification of a unique, unambiguous packet of Data with a fixed Byte Count.

Command Number Partitions

The command set is divided into the following classes (see Table 5.1):

- Universal Commands – A collection of commands that must be supported by all HART capable devices.

- Common Practice Commands – A collection of commands applicable to a wide range of devices. Common Practice Commands shall be supported by Devices whenever possible.
- Non-public – a special set of commands intended for factory-only use during the construction of a field device. These commands should not be used when servicing a device in the field.
- Wireless Commands – A Collection of commands to support WirelessHART products. All products supporting WirelessHART standard must implement all Wireless Commands.
- Device Family Commands – Collection of commands that allow the setup and parameterization of field devices without requiring using device specific commands or special device-specific drivers.
- Device-Specific Commands – Commands defined by the manufacturer according to the need of the field device.

Table 5.1 HART Command Number Partitions.

Command No.	Type	Description
0-30, 38, 48	Universal	See Universal Command specification.
31	Expansion flag	Used to signal the presence of a 16-bit command in the data field of the command. See Network Management Specification.
21-121 (Except 38 and 48)	Common Practice	See Common Practice Command specification
122-126	Non-Public	Intended for factory use only during the construction of the field device.
127	Reserved	-
128-253	Device-Specific	See the manufacturer's device-specific document.
254-511	Reserved	-
512-767	Additional Common Practice	See Common Practice Command specification.
768-1023	WirelessHART	See Wireless Command specification.
1024-33791	Device Family	See Device Family Command specification.
33792-64511	Reserved	-
64512-64763	Wireless Device Specific	These command numbers are reserved for use by wireless networking manufacturer specific commands.
64766-64767	Reserved	-

64768-65021	Additional Device Specific	These commands shall only be used when the command numbers in 128-253 are substantially consumed by the device.
65022-65535	Reserved	-

As shown in Table 5.2, WirelessHART commands are in the range 768-1023. These commands are used to support network management and gateway functions.

Table 5.2 WirelessHART Commands.

Category	Command No.	Description
Provisioning	768	Write Join Key
-	769	Read Join Status
-	770	Request Active Advertising
-	771	Force Join Mode
-	772	Read Join Mode Configuration
-	773	Write Network ID
-	774	Read Network ID
-	775	Write Network Tag
-	776	Read Network Tag
-	797	Write Radio Power Output
-	798	Read Radio Power Output
-	781	Read Device Nickname Address
-	962	Write Device Nickname Address
Managing Superframes and Links	783	Read Superframe List
-	965	Write Superframe
-	966	Delete Superframe
-	806	Read Handheld Superframe
-	807	Request Handheld Superframe
-	784	Read Link List
-	967	Write Link
-	968	Delete Link
-	786	Read Neighbor Property Flag
-	971	Write Neighbor Property Flag
Managing Graph and Source Routes	802	Read Route List
-	974	Write Route
-	975	Delete Route
-	785	Read Graph List
-	969	Write Graph Edge

-	970	Delete Graph Edge
-	803	Read Source Route
-	976	Write Source Route
Security	961	Write Network Key
-	782	Read Session List
-	963	Write Session
-	964	Delete Session
-	823	Request Session
-	814	Read Device List Entries
-	815	Add Device list Table Entry
-	816	Delete Device list Table Entry
-	821	Write Network Access Mode
-	822	Read Network Access Mode
Bandwidth Management	799	Request Timetable
-	800	Read Timetable List
-	801	Delete Timetable
-	973	Write Timetable
-	812	Read Packet Receive Priority
-	813	Write Packet Receive Priority
Device Management	960	Disconnect Device
-	972	Suspend Device(s)
-	777	Read Wireless Device Capabilities
-	778	Read Battery life
-	793	Write UTC Time Mapping
-	794	Read UTC Time Mapping
Network Maintenance	795	Write Time Interval
-	796	Read Time Interval
-	808	Read Packet time-to-Live
-	809	Write Packet time-to-Live
-	810	Read Join Priority
-	811	Write Join Priority
-	819	Read Back-Off Exponent
-	820	Write Back-Off Exponent
Coexistence	804	Read CCA Mode
-	805	Write CCA Mode
-	817	Read Channel Blacklist
-	818	Write Channel Blacklist
Network Health Reporting and Status	779	Report Device Health about the devices communication statistics
-	780	Report Neighbor Health List

-	787	Report Neighbor Signal Levels
-	788	Alarm “Path Down”
-	789	Alarm “Source route Failed”
-	790	Alarm “Graph route Failed”
-	791	Alarm “Transport Layer Failed”
Gateway Commands	832	Read Network Device Identity using Unique Id
-	833	Read Network Device’s Neighbor Health
-	834	Read Network Topology Information
-	835	Read Burst Message List
-	836	Flush Cached Responses for a Device
-	837	Write Update Notification Bit Mask for a Device
-	838	Read Update Notification Bit Mask for a Device
-	839	Change Notification
-	840	Read Network Device’s Statistics
-	841	Read Network Device Identity using Nickname
-	842	Write Network Device’s Scheduling Flags
-	843	Read Network Device’s Scheduling Flags
-	844	Read Network Constraints
-	845	Write Network Constraints

Command Requirements

- HART commands must be designed to be autonomous. They allow stateless operation of the device’s application layer.
- A HART command must perform one and only one of the functions: Read, Write, or Command.
- Command may contain indices allowing access to arrays or tables of data stored in a field device.
- Multi-transaction commands should only be used when a device exhausts the allowed set of device-specific commands.

Command Status Bytes

All slave response messages must return two Command Status bytes in the first two bytes of the Data field. The first byte is multiplexed and contains either the communication Status or the Response Code. The second byte always contains Field Device Status.

5.2 Dynamic and Device Variables

The HART protocol is designed to support smart field device technology and the 4-20mA Loop Current. Commands 1 and 2 return the Primary Variable (PV), Loop Current, and Percent Range. Command 3 adds to these the Secondary (SV), Tertiary (TV), and Quaternary (QV) Variables. Collectively they are called Dynamic Variables. Command 9 allows up to 8 digital values to be returned. Command 9 also includes status information as well as a timestamp indicating when these measurements were taken.

In addition, the protocol supports Device Variables for use in more sophisticated smart field devices and multi-variable field devices. Furthermore, multi-variable field devices can configure which Device Variable to connect to the current loop.

5.3 Host Conformance Classifications

Host conformance classes identify host capabilities based on the level of host functionality. This functionality encompasses the level of data access and manipulation provided by the host and includes references to Common Practice and Device Family Commands. Each conformance class includes the functions of all the lower classes. Class 1 is the minimum classification that can be claimed by any host application. Table 5.3 lists the classes.

Table 5.3 Host Conformance Classes.

Class	Description
0	The host does not meet the minimum requirements of Conformance Class 1.
1	The host can utilize cyclical process data from any field devices.
2	The host can supply the user with basic identification and configuration data about any field device.
3	The host can perform basic configuration of any device. Minimum level required to be classified as a “Generic Host”.
4	The host provides basic commissioning and calibration support for any device.
5	The host is capable of accessing all field device data items and all device-specific commands for any field device.

Chapter 6 WirelessHART Network

Abstract A WirelessHART mesh network consists of several different kinds of devices. The majority of nodes in a WirelessHART network are field devices that either collect process data or control some process. All devices are capable of routing. Some might only act as routers. The access points are the routers between the gateway and all other devices. The communication between the access point and the gateway is assumed to be robust and does not take up wireless bandwidth. The gateway is the interface between the wireless network and host applications. The gateway also serves as the bridge between the network and the network manager which controls the join, configuration, maintenance, and all other network management duties. The security manager manages the keys used at both the network and data link layers. Two special network device types are also defined by the WirelessHART standard, the adaptor and the handheld. The adaptor serves as the master in the wired HART network to provide communications to the wired devices through the wireless network. A handheld is normally carried by field personnel and used for maintenance and troubleshooting in the plant. Unlike other WirelessHART device types, the handheld is designed to be mobile and it is expected that it will connect to multiple WirelessHART networks and devices. Please refer to the Wireless Devices Specification (HCF_SPEC-290) for more details.

A WirelessHART product shall be classified as one of five different product types: field devices, adaptors, handhelds, gateway, and network manager. Additional device types will be added in future releases.

Fig. 1.2 shows a WirelessHART network connected to a plant automation network through a gateway. The plant automation network could be a TCP-based network, a remote IO system, or a bus such as PROFIBUS™ DP. The gateway is connected to the WirelessHART network through WirelessHART Access Points.

All devices directly connected to the WirelessHART network are a type of Network Device. Network Device types include Field Devices, Adaptors, Routers, Access Points and Handheld Devices.

All Network Devices transmit and receive WirelessHART packets and perform the basic functions necessary to support network formation and maintenance. All Network Devices must be able to source and sink packets and be capable of routing packets on behalf of other devices in the network.

6.1 Field Devices

Field Devices are connected to and characterize or control the Process. They are a producer and consumer of WirelessHART packets and must be capable of routing packets on behalf of other Network Devices.

Field devices are used in a wide range of applications including monitoring and controlling tank levels, monitoring emission levels and water quality, monitoring equipment health, and a wide variety of control applications. Field devices measure temperatures, pressures, flows, pH, density, composition, emission levels, vibration, etc. They are also connected to final control elements such as valves, agitators, blowers, and conveyors.

WirelessHART Field Devices may be line, loop, or battery powered or they may be powered in some other fashion. A Field Device is connected to the process or plant equipment. Field Devices may or may not support traditional current loop signaling. All WirelessHART field devices must have a maintenance port which is used for provisioning and local diagnostics. There is no requirement for WirelessHART devices to provide a wired 4-20mA signal.

6.1.1 General Requirements

All Field Devices must support all Universal Commands, some Common Practice Commands, and the standardized commands and procedures defined for WirelessHART Devices.

6.1.2 Maintenance Port

All Field Devices must provide a maintenance port that complies with the requirements in the Token-Passing Data Link Layer Specification. The maintenance port must also support at least one of the Physical Layers that is defined by the HART standards. This interface is used for provisioning and maintenance (e.g., to load the Join Key and Network ID or to monitor the join process).

Hand held devices and asset management applications connected to the maintenance port do not have access to the wireless network.

6.1.3 WirelessHART Device Interface

WirelessHART field devices are first and foremost HART devices. As such they must adhere to all of the requirements for HART devices *plus* they must adhere to all WirelessHART specifications. As part of device specifications they must support all Common Practice Commands.

As part of the device specification WirelessHART devices must support status and extended status. Status provides an indication of the measurement quality (level of goodness), whether the device making the measurement is healthy, and the timeliness of the measurement. Devices perform checks on hardware and software associated with the input or output and in turn reports this through the status value. The status of an output parameter is calculated to give an explicit indication of the quality of the value; good, poor, bad, or fixed. A good value may be used for control. A poor value is suspect and may not reflect the true measurement or calculated value. A bad value means that the parameter value does not reflect the true measurement, calculation, or control value. A fixed value means that the parameter value is constant and is not being updated in a periodic fashion. Status also reports additional information such as configuration changed, cold start, loop current fixed, loop current saturated, non-primary variable out of limits, and primary variable out of limits.

The Extended Device Status value provides information on whether the device has malfunctioned (Maintenance Required), a device variable is in an alarm or warning state (Device Variable Alert), or when power is critically low (Critical Power Failure). Both Device Status and Extended Device Status are included as part of burst mode communications.

Another feature in the WirelessHART standard is the timestamp that is communicated with all values. The addition of the time stamp allows applications to determine how current the parameter is, whether or not there is jitter in the measurement, and in some cases, use the time value to determine appropriate control actions using the measurement value.

Three other key features of Field Devices are burst mode, block data transfer, and event notifications. Burst mode is used to send data on an exception basis. Block data transfer is used to transfer a block of data, for example a spectral analysis, from a device to a host application. Event Notification publishes changes in the device's status, independently from data publishing supported in other Burst Mode commands.

A key feature of Field Devices is their ability to be installed and used in a wide range of applications. The actual process that the device is inserted into requires the device to be configured. As part of this configuration the device is given a tag, scaled (instrument scaling includes high scale value, low scale value, engineering units, and decimal places), and signal conditioning is applied (in the case of a valve it is important to know which direction the valve goes when the signal valve is increased). Field devices may be commissioned and calibrated in factory or at site in the instrumentation shop or once the device is installed. The set of parameters that a device supports and the methods available to the device is described by the EDD file written in EDDL. End users usually have standard parameter value templates for different types of devices and use these templates to customize devices for use in their plants (many use tools that use EDD files). In this way devices can be completely defined off-line and configured at the factory, in the shop, or prior to startup by downloading the offline configuration into the device. All that is required is to associate the unique device identifier in the configuration system and then use this as a key for the device.

Burst mode, block data transfer, maintenance & configuration, and event notifications all require network resources. The WirelessHART network contains support for devices, hosts, and the network manager to request network services and bandwidth to support these functions. Services are defined and described as timetables in the WirelessHART specifications.

6.2 Router Device

A Router Device is a Network Device that forwards packets from one Network Device to another. A Network Device that is acting as a Router Device uses its graphs and connections to decide which Neighbor Device to send packets. In general standalone routers are not required since all Network Devices must support routing. However, it may be beneficial (e.g., to extend the Network, or to save the power of a Field Devices in the network) to add additional devices to improve routing in the network. A router is not connected to the process and does not act as a gateway.

In some cases it would be better to install an additional Access Point rather than installing a Router. Access Points improve the overall throughput and redundancy of the network.

6.3 Adapter

A WirelessHART Adapter connects to an existing HART compatible field device or to several multi-dropped field devices and enables communication to all of connected field devices via a WirelessHART Network. The adapter must contain both a (wired) Token-Passing and a (wireless) TDMA interface. In other words, and unless explicitly stated otherwise, a WirelessHART Adapter must meet all requirements for wired and WirelessHART communication. Furthermore, since many of the devices connected to a WirelessHART Adapter will be of previous version levels, the Adapter must support key HART 7 capabilities for the sub-device. For example the Adapter must support a minimum of 5 burst mode messages and 2 event messages.

The WirelessHART Adapter must also allow complete access to the configuration and status of all the connected sub-devices. In addition to supporting HART communication with the connected field devices, the adapter must have no adverse impact on analog signaling.

The Adapter must support Universal and Common Practice Commands and identifies itself in Identity Command responses. In addition, the Adapter must support at least one Block Data Transfer connection.

The Adapter is responsible for negotiating network resources on behalf of itself and sub-devices. In many cases, for example when a search broadcast is sent out, the Adapter must also be capable of responding on behalf of its sub-device.

6.4 Handheld

Handheld Devices are used in the installation and maintenance of Network Devices. Handheld Devices are portable equipment operated by the plant personnel. There are four approaches to connect Handheld Devices:

- *HART Handheld or application connected through a Plant automation Network.* A plant automation network-connected Handheld Device connects to the plant automation network through some networking technology such as Wi-Fi™. This device talks to Network Devices through the gateway Device in the same fashion as external plant automation servers. To the WirelessHART network this type of handheld is just another host application.
- *HART Handheld connected through an FSK modem on the device.* In this mode the Handheld Device connects through an FSK modem directly to the device. When connected in this mode, the Handheld Device cannot talk out through the device into the WirelessHART Network.
- *WirelessHART Handheld connected to a WirelessHART Network.* A WirelessHART-connected Handheld Device is a device in the WirelessHART Network. In this mode the WirelessHART Handheld is restricted in the same way as any other device, i.e., it can only talk to the gateway and to the network manager. This mode is used to write keys into the Wireless Handheld and to view diagnostic and system health information. This will be referred to *Connected as a Network Device.*
- *WirelessHART Handheld connected to a WirelessHART Field Device.* A WirelessHART-connected Handheld Device connected over the WirelessHART Network to a WirelessHART Device is restricted to communication with the WirelessHART Device that the handheld is connected to. Special provisioning is used to ensure that the WirelessHART Handheld is restricted to one hop and one device at a time. This will be referred to *Connected as a Maintenance De-*

vice. Handheld devices connected in this manner must have their own session for use between the Handheld and the device they are connected to.

6.5 Gateway and Access Point

Gateway and Access Points connect the WirelessHART Network to a plant automation network, allowing data to flow between the two networks. The gateway Device provides host applications access to the Network Devices. A gateway Device 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. The primary function of the gateway is to cache burst data that is being reported by field devices and then return that data to host applications. Caching both improves the responsiveness of the gateway to host applications and greatly reduces network communications. With exception communications enabled it will be very common for network communications to be reduced by a factor of 10 to 20 times.

In many situations networks will have more than one access point. These multiple access points can be used to improve the effective throughput and reliability of the network. Access points communicate directly with the gateway.

Installing a wireless network includes installing wireless devices, a gateway (in some cases both a gateway and one or more access points), and a connection to a host or control system. Once the gateway, devices, and control system are configured the wireless mesh forms and communications begin.

To simplify support for redundant Access Points, every gateway has a fixed, well know address (Unique ID = 0xF981000002; Nickname = 0xF981). There is one gateway per WirelessHART network. Gateways can be redundant.

6.5.1 General Requirements

The gateway uses standard HART commands to communicate with network devices and host applications. The gateway also acts as a server and is responsible for collecting and maintaining cached data and command responses from all devices in the network. These cached responses correspond to burst messages, event notifications, and common HART command responses. These cached responses are returned immediately to host application requests. This reduces network communication load, improving power utilization and host application responsiveness.

The gateway must natively support Adapters allowing transparent access to the Adapters sub-devices. The devices connected to the Adapter can be identified by polling the Adapter for its sub-devices. The gateway uses Command 74 to determine how many sub-devices may be connected to the Adapter. Then the gateway issues Command 75 to walk through the legal combinations of polling addresses, IO Card, and Channel identifiers until all the connected sub-devices are identified.

If multiple Access Points are supplied by the gateway, the network manager will schedule communication traffic through all of them. If one of these WirelessHART Access Points fails then the network manager will adjust the schedule spreading traffic across the remaining WirelessHART Access Points. Each Access Point has its own physical and nickname Address.

Internal to the gateway, all Access Points route traffic through the gateway to either a Host Interface or the network manager.

The WirelessHART gateway must provide the network clock to other Network Devices. The clock information ripples downward from the top of the network hierarchy to the bottom.

6.5.2 Gateway Model

WirelessHART gateways connect the WirelessHART Network with other networks, such as plant automation networks, allowing HART Commands/ Responses, tunneled messages, formatted XML, and diagnostic messages to flow between the two networks. The WirelessHART Network uses the concept of a gateway to provide a single entry point into the WirelessHART Network. Access to Host interfaces is through Service Access Points. Access to the WirelessHART Network itself is provided through WirelessHART Access Points.

Gateway

The gateway provides a single entry point into the WirelessHART Network.

- It is part of the WirelessHART Field Device network.
  a. It is a device type in the WirelessHART Network.
  b. It communicates through Access Points to any Field Device in the WirelessHART Network (the gateway must have a path to every device in the WirelessHART network).
- It can communicate directly with the network manager.
- It sources time synchronization messages.
- It is a HART Device Type
  a. Described with DDL
  b. Supports HART DD
- It supports one or more Service Access Points for connecting to the automation network and plant backbone. It supports the following through these Service Access Points:
  a. Translation functions satisfying HART Commands with locally cached data. The WirelessHART gateway implements a data cache to optimize the overall performance of the WirelessHART network and improve responsiveness to host applications.

b. Tunneling functions transferring HART Commands to WirelessHART Network requests. The WirelessHART gateway can connect with the host application via various protocols (e.g., Modbus™, Profibus DP, ControlNet™, HART OPC server, proprietary, other) based on different physical layers (RS-485, Ethernet LAN, Wi-Fi, etc).

c. Optionally supports an XML-based interface.

- Compatibility

  a. The WirelessHART gateway can support existing HART commands (only to the extent that the gateway is acting as a translator or proxy).
- It provides buffering for

  a. Burst Mode.

  b. Event Notification.

  c. Cached command responses.

  d. Diagnostics.

  e. Large data transfers (several specific cases used in the development of the WirelessHART specifications, for example a valve uploading its signature information and the results from a vibration analysis are two use cases where the gateway is receiving from a device; the gateway can also perform large data/file transfers down to a device).
- It provides support for publishing variables to devices (often referred to as catch variables). In this case the WirelessHART gateway will be able to publish burst mode data that it is caching to other devices in the WirelessHART network.

The network used on the host side may consist of a variety of technologies. Most PLC, DCS or SCADA vendors utilize a proprietary network. Asset Management and Device Management companies tend to use open protocols, such as TCP/IP and one of several standard MAC/PHY layers such as IEEE 802.11™ and IEEE 802.3™.

WirelessHART Access Point

A WirelessHART Access Point is a Network Device that connects gateways into the WirelessHART Network. A WirelessHART Access Point has a WirelessHART connection on one side and an external connection on the other side – the external connection could be an Ethernet or Wi-Fi connection or a proprietary connection. The external connection is not specified by the WirelessHART standard. A WirelessHART Access Point is not directly connected to the process. WirelessHART Access Points provide the following:

- They are part of the WirelessHART Field Device network.

  a. They are a device type in the WirelessHART Network.

  b. They communicate with the gateway via dedicated link or communication port.

c. Each WirelessHART Access Point can support communication with any device to which the network manager has provided a path.

Service Access Point (SAP)

Service Access Points provide a connection to the automation network and plant backbone. They provide:

- An interface to the gateway for host systems or applications that wish to access Network Devices that are part of the WirelessHART Network. The interface provides support for accessing all wired HART Devices that are included through Adapters.
- Access to cached response messages:
  a. Burst Mode Reponses.
  b. Event Notification Reponses.
  c. Cached command responses.
- Access to diagnostics.
- Access to network manager data.
- Support for block mode data transfers (e.g., uploading results from a vibration analysis).
- Tunneling functions transferring HART Commands to WirelessHART Network and WirelessHART Device requests. Through these SAP's the gateway can connect with the host application via various protocols (e.g., Modbus, Profibus DP, ControlNet, HART OPC server, proprietary, other) based on different physical layers (RS-485, Ethernet LAN, Wi-Fi, etc).

To support Service Access Points two interface types are provided. The first directly supports HART Commands – this interface must be supported by all gateway implementations. The second supports XML-formatted commands – the XML interface is optional.

Tunneling Protocols

Gateways must also be able to support tunneling protocols. Tunneling protocols are used to relay messages between the host which is outside the WirelessHART Network and a destination device that is part of the WirelessHART Network. All WirelessHART gateways must be able to support HART and WirelessHART universal and common practice commands. They may also support manufacturer specific commands.

There are several categories of Tunneling gateways – WirelessHART, HART over Ethernet, open protocol such as TCP/IP, and Vendor Specific

Host

The Host interface is used to connect clients outside of the WirelessHART Network with the WirelessHART Network and devices in the WirelessHART Network. Host interfaces take on many forms – several common ones include:

- Ethernet-to-wireless gateway Device—A gateway device that provides a bidirectional path between industrial Ethernet Networks and the WirelessHART Network.
- Wi-Fi-to-wireless gateway device—A variation of an Ethernet-to-wireless gateway device that uses 802.11 a/b/g radio to connect to the plant's network.
- Serial-to-wireless gateway device—If plant automation servers and equipment support serial interfaces, a serial-to-wireless gateway device can be used to connect to the serial interfaces of these devices.

WirelessHART gateways must be able to cache burst mode commands, several commonly used read and write commands, and diagnostics. To take advantage of this caching, the gateway must also be able to act as a translator. As a translator, the gateway peeks at requests from the Hosts and, if the response data is cached and is current, returns the cached response messages from its real-time database. For example, if a client on a host issues HART command 0 request to a device in the network, the gateway will check to see if it has a cached command 0 response. If the gateway does not have a cached command 0 response, it will forward the command to the device and return the resulting response to the client.

The translation functions can be quite involved. They deal with network layer as well as some application layer interactions. In the network layer, the different response packet sizes have to be dealt with, and a mapping of security, priority, addresses and such is made.

Cached Response Messages

Network Status

Each Network Device maintains diagnostics. The diagnostics are periodically published via HART commands to the network manager. The network manager maintains the complete set of device and network diagnostics. Hosts can query the gateway or the network manager for network level diagnostics.

Burst Mode Command Responses

The database caches all of the burst mode response messages.

Event Notification Command Responses

The database caches all of the event notification response messages.

Cached Command Responses

The database caches the latest response messages for several commands (these commands are summarized below).

Delayed Response Command Responses

For HART request/ response commands, the gateway maintains a complete list of all outstanding commands that have been sent to devices for which a response has not yet been received in return. Delayed Response Commands must be purged if they exceed a 24 hour timeout.

6.6 Network Manager and Security Manager

The network manager is also treated as a type of Network Device. Doing so allows other HART devices to exchange HART Commands with the network manager. The network manager is described in detail in this section.

The network manager is responsible for the overall management, scheduling, and optimization of the WirelessHART Network. As part of its duties the network manager initializes and maintains network communication parameter values. The network manager provides mechanisms for devices joining and leaving the network. It is also responsible for managing dedicated and shared network resources.

The network manager communicates with devices on the WirelessHART Network through the network layer which is described in the Network Management Specification. The commands that the network manager uses to setup, monitor, and manage the overall network are described in the Common Practice Command Specification and the Wireless Commands Specification. The network manager is also responsible for collecting and maintaining diagnostics about the overall health of the network. These diagnostics are available to be reported to host-based applications. The diagnostics are also used to adapt the overall network to changing conditions.

To perform its complete set of functions the network manager needs information about the devices themselves, information about how the network is to be used, and feedback from the network on how well the network is performing. Configuration and setup information about devices is read from the devices themselves. Communication resources are requested by devices, applications, and users. Feedback on how well the network is performing is provided by the devices themselves through health reports and diagnostics.

The user (administrator/maintenance) interacts with the network manager application which generates a network management control packet to Network Devices. Network Management packets travel through the Network layer, then through the Data-Link and the Physical layer before being transmitted through the air to the destination device.

6.6.1 Core Network Functions

Network Manager

The network manager forms the WirelessHART Network, joins and configures new Network Devices, and monitors the network. The network manager uses diagnostic information to adjust the network topology – since these adjustments are made on an on-going basis the overall operation is referred to as adapting or grooming the network.

The WirelessHART Network architecture does not restrict where the network manager resides in the plant automation network. As shown in Fig. 1.2, the network manager may be co-located with the gateway in the same box or located in a completely separate physical box. There is one network manager per WirelessHART Network. A network manager may span multiple WirelessHART networks.

Connection between Network Manager and Security Manager

The Security Manager and the network manager are responsible for establishing a connection with each other, and maintaining this connection to support device join requests and establishment of sessions. The connection between the network manager and the Security Manager and the method of securing it are not described by the WirelessHART standard. The Security Manager is completely hidden from the gateways.

Security Manager

The Security Manager works with the network manager to secure the WirelessHART Network from adversarial threats to its operation. The Security Manager generates and manages the cryptographic material used by the network. It is responsible for the generation, storage, and management of keys.

The Security Manager works closely with the network manager in a server-client architecture. The Security Manager is shown separately from the network manager because it may be a centralized function in some plant automation networks, servicing more than one WirelessHART Network and in some cases other networks and applications. There is one Security Manager associated with each WirelessHART Network.

A secure connection between the network manager and Security Manager is required. This secure connection is outside the scope of the specification.

Network Diagnostics

As part of its system functions, the network manager collects network 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, reconfiguration of the network is performed while the network is operating. Network diagnostic information can be accessed through HART commands.

Network Performance

The WirelessHART Network maintains very high reliability through the use of several mechanisms including multiple paths to network devices, multiple RF channels, and multiple communication tries. If improved reliability is required, more paths can be inserted by adding additional WirelessHART Access Points and field devices. Additional devices improve path diversity. Additional WirelessHART Access Points, and devices in general, increase throughput, reduce latency, and can be used to route around potential interferers.

Time-synchronized Communication

All communication on the WirelessHART Network is time-synchronized. The basic unit of measure is a time slot which is a unit of fixed time duration commonly shared by all Network Devices in a network. The duration of a time slot is sufficient to send or receive one packet per channel and an accompanying acknowledgement, including guard-band times for network-wide synchronization. The per-channel qualification indicates that more than one communication can occur in the same time slot.

Precise time synchronization is critical to the operation of networks based on time division multiplexing. Since all communication happens in time slots, the Network Devices must have the same notion of when each time slot begins and ends, with minimal variation. The WirelessHART protocol defines mechanisms for time synchronization. In the WirelessHART Network, time propagates outward from the gateway.

Sessions

End-to-end communications are managed on the Network Layer by sessions. Each session contains information on security for a pair (or group) of network devices. All network devices will have 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 two network manager session keys. 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, a special Network Device address 0xFFFF will be used.

Connection between Gateway and Network Manager

The interface between a gateway and the network manager is not described by the WirelessHART protocol. The network manager and the gateway are responsible for establishing a secure connection with each other, and maintaining this connection to carry control and data traffic. Thus, it is not necessary for the gateway to go through the normal network device join process. Once the gateway connects to the network manager, the network manager may configure the gateway to begin advertising to other devices.

Once communication paths have been established the network manager is not involved in communications between host applications and network devices. The gateway is responsible for buffering, protocol conversions, timeouts, time clock, etc.

Scheduling

The main functions of the network manager are to schedule, monitor, manage, and optimize communication resources. The network manager combines information it has about the topology of the network, heuristics about communication requirements, and requests for communication resources from network devices and applications to generate the schedule.

6.6.2 Network Manager Requirements

The network manager is central to the overall operation of the WirelessHART Network. The network manager is responsible for forming the network, establishing routes, scheduling communication resources, monitoring the health of the network, adapting the network to on-going changes, and working with the Security Manager to allocate and manage Session Keys. The overall set of requirements is summarized below in Table 6.1.

Table 6.1 Network Manager Requirements.

Network Function	Requirement
Network Formation and Configuration	Provides logic for initializing itself and starting up the network.
-	Manages Topology. Understanding the topology of the network. Adapts the network to changes as diagnostic information is reported from devices.
-	Manages the Network Key. The Network Key is provided to the Network Manager by the Security Manager and is provided to all Network Devices. The Network Manager distributes the Network Key and changes it as required by plant security policies
	Separate Network Manager and Gateway keys are used for unicast and broadcast traffic that originating from each of them.
-	Manages Join process. The Network Manager validates devices that wish to join the network. After authenticating a Network Device, the Network Manager gives the joining Network Device the network key and four session keys 1 - Network Manager unicast session keys 2 - Network Manager broadcast session key 3 - Gateway unicast session key 4 - Gateway broadcast session key
	Devices need to be configured with NetworkId independent of the network manager in order to join properly. They need it to find the right network.
-	Assigns 16-bit Nicknames. The Network Manager assigns and manages network unique 16-bit Nicknames (network addresses) to each Network Device. The Network Manager is responsible for ensuring that the Neighbor Table inside each device is up-to-date.
-	Establishes a connection with the Gateway. Whenever the Gateway (via the Gateway's Access Points) receive messages destined for the Network Manager the Gateways forwards them to the Network Manager.
-	Configures at least one of the Gateway's Access Points to provide the network clock.
-	Manages network configuration. Maintains a full map of the network configuration, including any information about the network that has been distributed to network devices.
-	Responds to requests for network information. For example, when a host application makes a request for all of the Network Devices in the network, the Network Manager is responsible for providing the response.

Routing	Creates and manages network route. The network route is a complete map of the network.
-	Manages Neighbor tables. The Network Manager collects network statistics and neighbor table information from each device through periodic health reports. This information is used to adapt the network to changes.
-	Builds route tables for Graph routing. Graph Routing is ideal for both scheduled upstream and downstream communications. Upstream communications include process measurements and alarms. Downstream communications include SP changes to actuators.
-	Builds source route lists for Source routing.
-	Allocates communication resources to itself, gateways, and to the Devices so that the Network Manager can manage the network and so Devices can communicate.
Network Schedule	Creates Superframes. Multiple Superframes will be used to support communications at specific scan rates. Additional Superframes will be allocated to support device management and diagnostic applications which require large amounts of traffic for short periods of time.
-	Assigns links in Superframes.
-	Creates Link tables. Each Link includes exactly one slot associated with a Superframe, its type (normal, advertising, discovery, join) its options (transmit, receive, shared), neighbor information, channel offset, and the device connected to this link.
-	Activates and Deactivates Superframes in response to application demands.
-	Maintains overall WirelessHART Network diagnostic information. For example, when a Network Device has not received a packet from one of its neighbors within the KeepAliveInterval, the device sends a path-down notification to the Network Manager indicating that the path is no longer available.
Network Schedule and Channel Management	Keeps track of blacklisted channels (blacklisting is a manual operation).
-	Provides Channel Offset. The channel offset is used to calculate the channel number when channel hopping. The channel offset takes on a value of 0 to (number of channels less number of blacklisted channels).
Network Diagnostics and Adapting	Maintains record of health information about each Network Device.
-	Adapts network to changing environment and application demands. The adaptation includes updating route and schedule information.
-	Allocates communication resources as requested by Network Devices. Devices request network capacity to support burst mode, event notification, and block mode traffic. Gateways request bandwidth to support client demands. Network traffic is biased towards a particular path by increasing or decreasing the number of links through a particular device.
-	Optimizes routes and schedules in order to improve operation of the network while conserving power within devices.
-	Creation and management of Join Keys.
Security Manager	Creation and management of Session Keys.

6.6.3 Scheduling

The most important thing the network manager does is to schedule communication resources. In order to put together efficient and optimized schedules the network manager needs information about the network, information about communication requirements, and information about the capabilities of the network devices themselves. As this information is discovered the network manager adjusts the schedule until it has met the requirements of the network. The scheduler then uses feedback from the operation of the system to tune the schedule.

Schedule Requirements

The requirements for developing the schedule are summarized in Table 6.2.

Table 6.2 Scheduler Requirements.

Function	Requirement
Assumptions	Network Manager has reasonable representation of network graph.
-	Each device has been configured with a connection table.
-	Network Manager knows the update rate of each device.
-	For redundancy, a datum is configured with one transmit and a retry on one path and another retry on another path.
Constraints	Maximum number of concurrent active channels is determined by the number of enabled channels (limited by black-listing).
-	No devices can be scheduled to listen twice in a slot.
-	More than one device can transmit to the same device (e.g., A broadcast link and dedicated links to each of the listening devices can coexist.)
-	On multi-hop path, early hop must be scheduled first.
-	The supported update rates should be defined as 2^n where ‘n’ is positive or negative integer values e.g., update rate selections of ¼ 250msec, ½ 500msec, 1 sec, 2 sec, 4 sec, 8sec, 16 sec, 32 sec, and 60 sec. (or more)
-	Base Network Management and Burst Mode communications should not exceed 30% of the available communication bandwidth (100 slots/sec max).
-	Services – the network manager must take into account the service requirements.
-	The final schedule (not counting the Gateway spec) should have 50% free slots (i.e. allocated for retries, listens).
Data Superframe	The data Superframe length is determined by data scan rate.
-	Allocate slots starting with the fastest to the slowest scan rate.
-	From the furthest end device, allocate one link for each en-route Network Device to the Gateway. Allocate a 2nd dedicated slot on the same path to handle a retry. Allocate a 3rd shared slot on a separate path to handle another retry.

Management Superframe: Management	Management Superframe has priority over data Superframes.
-	Traverse the graph by breath-first search, starting from the gateway, number the devices as N_0, N_1, … $N_{n.}$
-	At a minimum, every device needs to have a slot for a Keep-Alives and there must be a corresponding shared receive on the parent side.
Management Superframe: Join Process	Join-Request - From the furthest devices, allocate one link for each en-route Network Device to the Gateway (No redundancy provided).
-	Join Response - Traverse the graph by breath-first search, allocate one link for each en-route Network Device from the Gateway to end Network Device (No redundancy provided).
-	Allocate advertise packets in each device. The number of advertise packets will be inversely related to the number of hops away from the gateway.
Management Superframe: Neighbor Discovery	Neighbor discovery. The Network Manager shall allocate discovery links common to all Network Devices. The discoveryInterval timer shall be set to enable discovery.
Management Superframe: Network Management Commands	Share the Network Management links with join requests and responses.
Command Request/Response Traffic	Allocate shared slots to meet ad-hoc request and response traffic.
Gateway Superframe	The Gateway Superframe should be allocated with a large ID value.
-	The Gateway Superframe should be 40 slots long. All slots in the Gateway’s Access Points should be allocated).
-	Schedule all unallocated slots in the Gateway. Alternate each of these slots as XMIT, RECEIVE (receive slots must be shared).
Special Purpose Superframes: High Throughput	Allocated by gateway or client to address high throughput demand to satisfy asset management and other applications. This will be allocated as a “maintenance” or “block transfer” service type.
Special Purpose Superframes: Maintenance Superframe	Allocated in the Handheld and Every Field Device. This Superframe is used to provide the Field Device and the Handheld with a high-speed connection to talk on. The Network Manager will allocate 4 slots per second (two links in each direction).

PART II WirelessHART in Depth

This part discusses the key topics in the WirelessHART standard that should be of interest to people working with it. These topics are important in different ways. They may be of topics that are helpful in understanding the specification such as TDMA and the timeslot; they may be of topics that are not clear in the specification such as the bit orders and byte orders; they may be of topics that are interesting of itself such as the origin of the public key; they may be important to the development of WirelessHART products such as the join sequence; they may also be of topics that answers general questions from the public such as security and coexistence.

Chapter 7 includes topics related to the stack.

Chapter 8 includes topics related to the network.

Chapter 9 includes generic topics that are related to the WirelessHART standard.

Each section in the chapters addresses a unique question and is written as a self-contained unit. The readers could read the sections in any order, pick any topic that is relevant at the time.

Chapter 7 An Example

Abstract In this chapter an example of wireless transmitters and actuators associated with a bio-reactor process is used to demonstrate the operation of the wireless mesh network. In this example only wireless communication with the sensors is discussed. In other words, no wireless control is considered in the design and management of the wireless network. We illustrate how the network traffic is scheduled via superframes and links. We provide result for two network topologies, the single hop example in which all sensors are directly connected to the gateway via the access point, and the multi hop example in which some sensors are connected via other sensors to the access point and the gateway.

The example is adapted from Annex C of Wireless Devices Specification (HCF_SPEC-290), which also appears in one of our publications (Nixon et al. 2008). In this chapter gateway refers to the combination of the gateway and an access point.

The bio-reactor is illustrated in Fig. 7.1.

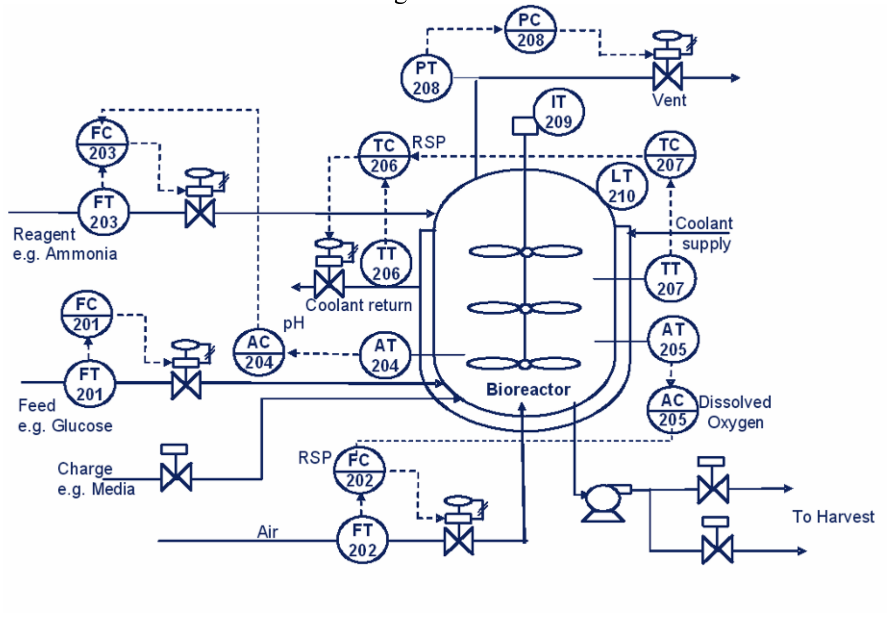


Fig. 7.1 Bio-reactor process.

The measurements, actuators, and blocking valves included in this example are summarized in Table 7.1.

Table 7.1 Instrument and Valve List for the Bio-Reactor.

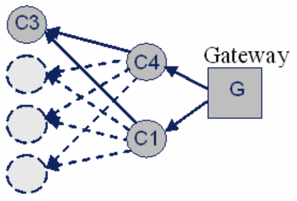
Category	DEVICE	Measurement
Measurement	C1	Reactor Level (LT210)
-	C2	Feed Flow (liquid –FT201)
-	C3	Reactor Gas Pressure (PT208)
-	C4	Reactor Temperature (TT207)
-	C5	Agitator Amps (IT209)
-	C6	Return Water Temperature (TT206)
-	C7	Reagent Flow (FT203)
-	C8	Air Flow (FT202)
-	C9	Dissolved Oxygen (AT205)
-	C10	pH (AT204)
Regulating Valve	A1	Feed Flow (FV201)
-	A2	Reagent flow (FV203)
-	A3	Coolant Flow (FV206)
-	A4	Vent Flow (FV208)
-	A5	Air flow (FV202)
Blocking Valve	B1	Charge Flow (FZ211)
-	B2	Harvest Flow (FZ212)
-	B3	Harvest Flow (FZ213)

7.1 Network Management and Host Request

The first step in the formation of the mesh is to allocate communication resources for network management. Devices wishing to join the network listen for advertisement packets and then use information in the packets to generate join requests. When a join response is received by the Network Manager, the Network Manager verifies the device's join key and either completes or rejects the join process. The only network devices that know the join key are the Network Manager and the Field Device. As part of this process, the Network Manager creates dedicated slots in the management superframe for device management and advertisement functions. A slot will be defined in the management superframe for the newly added device to send advertisement packets. Also, shared slots will be reserved in the management superframe for the new device to send and receive messages from a device responding to its advertisement packet. This new slot information is transferred into the network and reflected in all of the effected devices.

It is expected that any new device will listen for a period of time for advertisement packets and select the device with the strongest signal strength for joining the network. In most cases a device will be able to transmit a join request to the

Network Manager since a slot is dedicated for a join request. However, if two new devices happen to transmit a join request at exactly the same time using the advertised slot and channel, then the messages will collide. In this event, collision will be detected and transmission attempted after a backoff time. Thus, the join request will be spaced and will most likely be successful on the next attempt.



Device	Peer
C3	C4, C1
C4	G
C1	G

Network Management - Transmit Frame/Graph

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5	TS6	TS7	TS8	TS9	TSn
0	G Advertise	G=>*				G=>C4	C4=>C3				
1	C4 Advertise	C4=>*				G=>C1	C1=>C3				
2	C1 Advertise	C1=>*									
3	C3 Advertise	C3=>*									

Network Management - Receive Frame/Graph

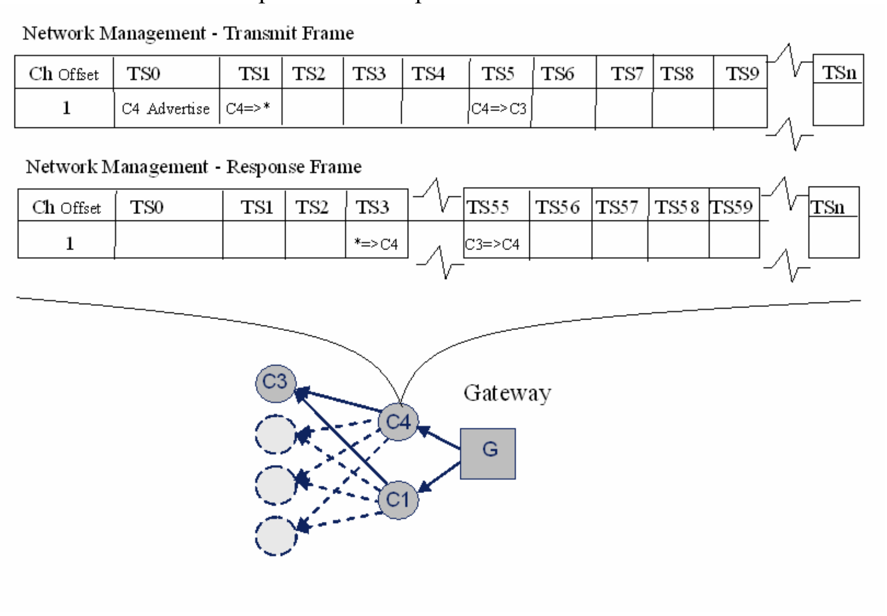
Ch Offset	TS0	TS1	TS2	TS3		TS55	TS56	TS57	TS58	TS59	TSn
0				*=>G		C3=>C1	C4=>G				
1				*=>C4		C3=>C4		C1=>G			
2				*=>C1							
3				*=>C3							

* Temporary slot reserved for communication with a new device.

Fig. 7.2 Network Management Frames.

Network devices join the network one at a time. Overall network routing will be determined by the Network Manager based on signal strength as determined by physically layout of the plant, number of hops, and traffic flow. The overall routing will be further adjusted to reflect diagnostic information, retries, etc. In establishing a schedule for management communications, it may be assumed that only one outstanding network management communications will be allowed. Thus, a device with multiple peers may be scheduled within the same slot since the transmission will only go to one device. This allows the number of slots required for scheduling and the associated power to be reduced. Scheduling of the response should take into account the maximum response time of the devices included in the network. For example, assuming a maximum response time of 0.5 seconds, the two management superframes may be illustrated as shown in Fig. 7.2 for the case where four devices have joined a network. To simplify this scenario the gateway and WirelessHART access points have been combined into the same box and labeled as "G".

Only the portions of the superframe that are directly associated with a specific Network Device will be transferred to each Network Device, as illustrated in Fig. 7.3 for device C4 in the previous example.



Network Management - Transmit Frame

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5	TS6	TS7	TS8	TS9	TSn
1	C4 Advertise	C4=>*				C4=>C3					

Network Management - Response Frame

Ch Offset	TS0	TS1	TS2	TS3	TS55	TS56	TS57	TS58	TS59	TSn
1				*=>C4	C3=>C4					

Fig. 7.3 Network Management Frames/Graph Transferred to C4.

The speed of response in transferring network management request and response is primary determine by the how often transmissions are scheduled to and from the Network Managers. In this example, the frequency of request can be regulated through the adjustment of the superframe size. For example, if the superframe size is set to 100 slots, then requests and responses will be transferred at a maximum rate of once per second with one retry (based on each slot requiring 10 ms). If the scheduled network slot communications conflicts with another superframe, then the network management communications will automatically be delayed to avoid collisions since the network superframe ID will always be set to be the highest numeric value.

7.2 Process Measurement

The primary objective of most field devices will be to provide a process measurement. The frequency at which this information is required by the process automation host is specific to the process equipment and the measurement type e.g. pressure, temperature, flow, level, and analytical. Thus, as part of the process automation host configuration, the user will configure the following information for all network devices that are accessed through the wireless gateway:

- Device Tag – which uniquely identifies the device e.g. HART Tag,
- Measurement value(s) that are to be accessed in the network device,
- How often each measurement value is to be communicated to the gateway.

For the batch bio-reactor example, the field device measurements may be configured as in Table 7.2.

Table 7.2 Measurements and Scan Rates for the Bio-Reactor.

Device	Measurement	Update Rate
C1	Reactor Level (LT210)	16 sec
C2	Feed Flow (liquid –FT201)	1 sec
C3	Reactor Gas Pressure (PT208)	1 sec
C4	Reactor Temperature (TT207)	4 sec
C5	Agitator Amps (IT209)	8 sec
C6	Return Water Temperature (TT206)	16 sec
C7	Reagent Flow (FT203)	1 sec
C8	Air Flow (FT202)	1 sec
C9	Dissolved Oxygen (AT205)	4 sec
C10	pH (AT204)	4 sec

To support the configuration of multiple superframes for the transfer of process information at different rates, the configured measurement scan rates used by the field device and the associated communication should be configured as integer multiples of the fastest update time that will be supported by field devices. For this example, the supported scan rate will be defined as 2x where x is of non-negative integer values, e.g., scan rate selections of 1 sec, 2 sec, 4 sec, 8sec, 16 sec, and 32 sec.

To avoid introducing latency into the measurement value that is communicated to the gateway, it is important that the processing of the sensor be coordinated with the slot configured for the measurement transmission, as illustrated in Fig. 7.4.

The scheduling of communications associated with process measurements included in a network can be simplified by defining a superframe for each scan period and developing the schedule by allocating slots for transmission of measurement data starting with the fastest to the slowest scan rates. In the schedule, a device may only occur once within a slot time since at any given time a device is either transmitting or receiving on one channel.

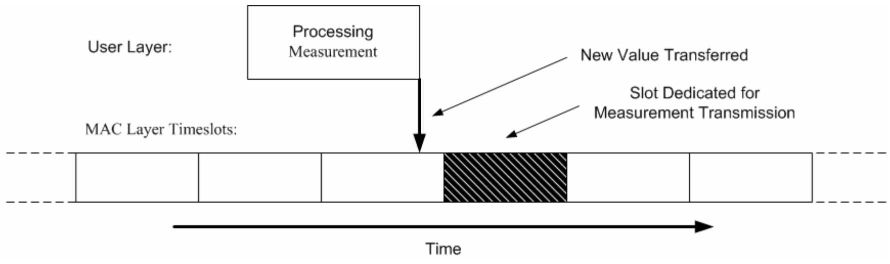


Fig. 7.4 Synchronize Measurement Processing and Transmission.

7.3 Scheduling Example – Single Hop

Using the recommended approach of defining superframes for each scan period and allocation of slots starting with the fastest to the slowest measurement, the graph associated with the batch bioreactor measurements would appear as shown in Fig. 7.5, assuming the gateway and all the network devices are only separated by 1 hop.

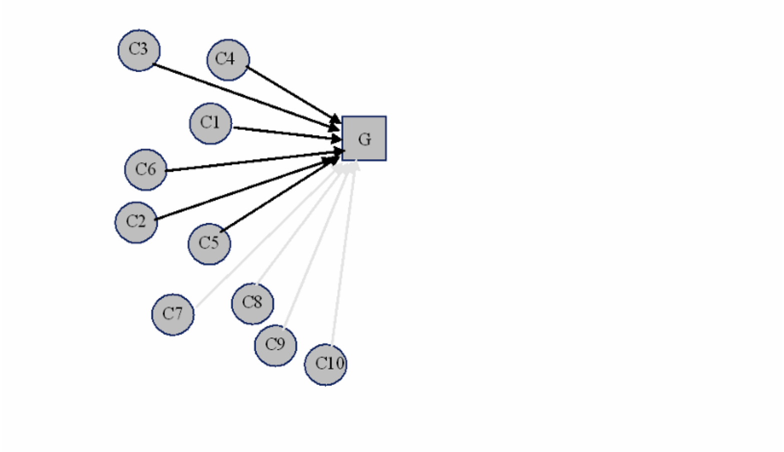
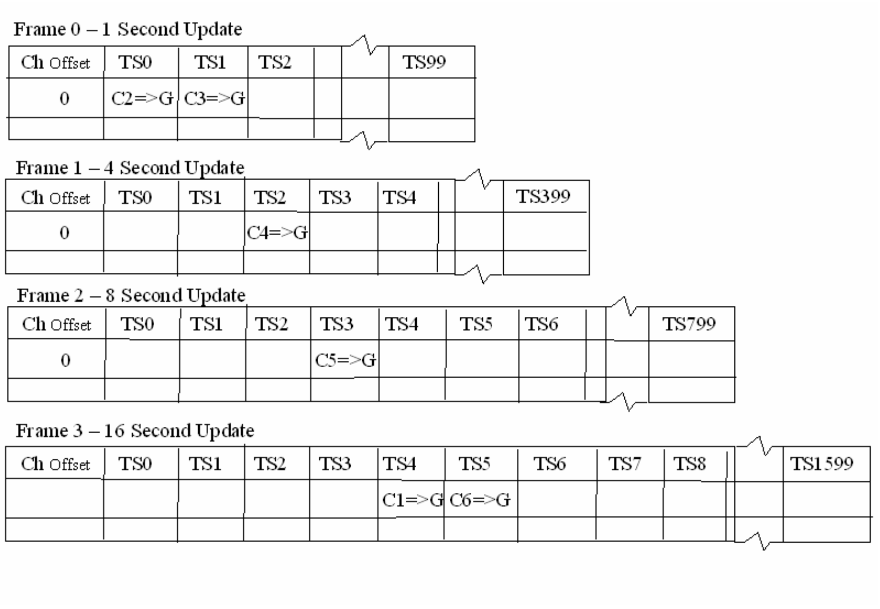


Fig. 7.5 Batch Bio-Rractor Example – Single Hop.

The schedule for this configuration is illustrated in Fig. 7.6.

To support immediate re-transmission (if required) after a failed transmission to the gateway, additional slots would be added in the schedule immediately after each transmission.


Frame 0 – 1 Second Update

Ch Offset	TS0	TS1	TS2		TS99
0	C2=>G	C3=>G			

Frame 1 – 4 Second Update

Ch Offset	TS0	TS1	TS2	TS3	TS4		TS399
0			C4=>G				

Frame 2 – 8 Second Update

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5	TS6		TS799
0				C5=>G					

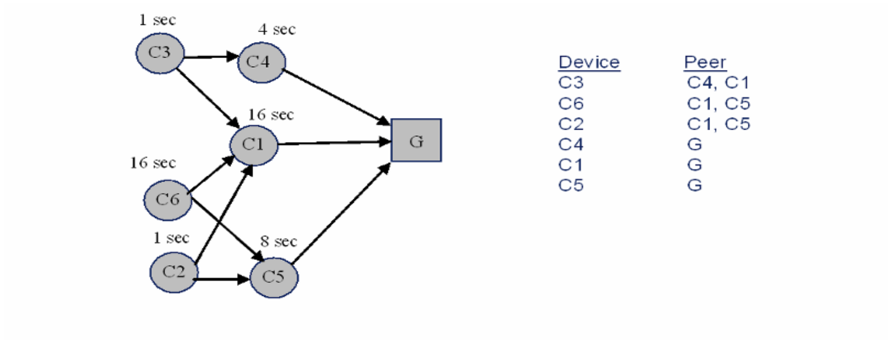
Frame 3 – 16 Second Update

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5	TS6	TS7	TS8		TS1599
					C1=>G	C6=>G					

Fig. 7.6 Multiple Frames for Different Update Rates – Single Hop.

7.4 Scheduling Example – Multiple Hop

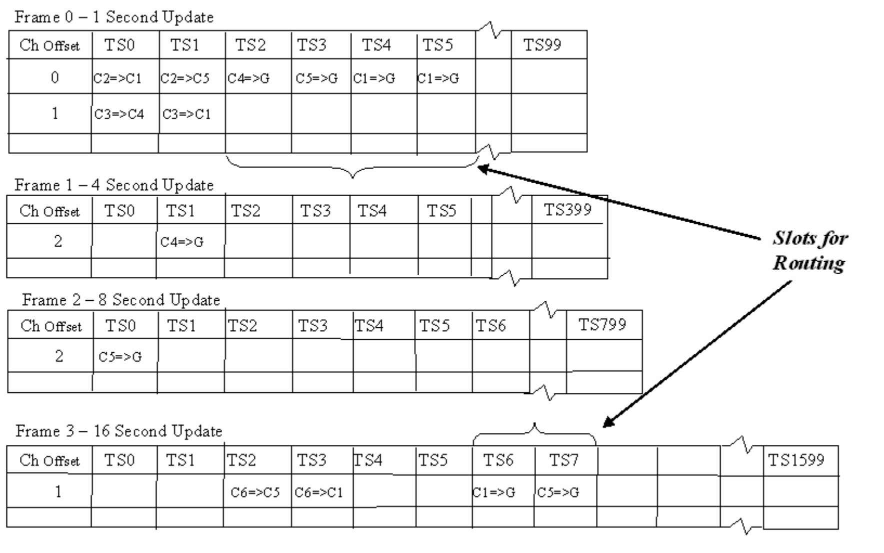
The previous example presented the ideal case where all network devices were connected through the gateway through a single hop. In many cases this will not be the case – multiple hops will be utilized.



Device	Peer
C3	C4, C1
C6	C1, C5
C2	C1, C5
C4	G
C1	G
C5	G

Fig. 7.7 Bio-Reactor Example – Multiple Hops.

A WirelessHART Network is formed through the join process. When a Network Device does not communicate directly with the gateway, then added communication slots must be reserved in the schedule for packet routing. Also, if the network is configured to allow a network device to have multiple peers, then communication slots must be defined for each peer's data plus the messages it must route. When slots are reserved to transmit to both peers, the second slot in the superframe is used for transmission only if no acknowledgement is received from its first peer in the superframe. The impact of routing and for a network device to support multiple peers is illustrated in Fig. 7.7. The resulting schedule is shown in Fig. 7.8. The data delay from device C3 to the gateway ranges from 30ms with no retry to 60ms with multiple retries. This and the resulting jitter are very small compared with the date rate of 1 second.



Frame 0 – 1 Second Update

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5		TS99
0	C2=>C1	C2=>C5	C4=>G	C5=>G	C1=>G	C1=>G		
1	C3=>C4	C3=>C1						

Frame 1 – 4 Second Update

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5			TS399
2		C4=>G							

Frame 2 – 8 Second Update

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5	TS6		TS799
2	C5=>G								

Frame 3 – 16 Second Update

Ch Offset	TS0	TS1	TS2	TS3	TS4	TS5	TS6	TS7				TS1599
1			C6=>C5	C6=>C1			C1=>G	C5=>G				

Slots for Routing

Fig. 7.8 Multiple Frames for Different Update Rates – Multiple Hops.

While retransmission for reliability has to be scheduled in advance and will therefore limit bandwidth even when retries are not needed, WirelessHART provides other means to make it up. Besides the data transmission, up to 15 channels could be simultaneously used which increases the bandwidth by 15 fold. The effective WirelessHART bandwidth is higher than that of the wired HART version.

Chapter 8 Discourses on the Stack

Abstract In this chapter we delve into details of selected topics on the stack. They are categorized into groups of physical layer, network layer, application layer, cross layers, and other topics. The topics are listed as section titles.

8.1 Physical Layer

8.1.1 Physical Channel and Maximum Bandwidth

The WirelessHART standard currently uses the 15 of the 16 channels in 2.4GHz spectrum defined by the IEEE 802.15.4 standard. While many mesh networks based on the IEEE 802.15.4 standard uses only one channel at run time, the WirelessHART network could use all 15 channels during its lifetime (since channel 26 is not legal in some locales, the decision was made to remove this channel). Furthermore, it allows multiple channels to be used at the same time. In other words, 15 pairs of devices in the same network could have 15 communications on 15 different channels in one timeslot.

We shall now calculate the maximum achievable bandwidth with a WirelessHART network. The IEEE 802.15.4 standard defines its maximum raw data rate to be 250kb/s. This is how fast the bits are transmitted. The maximum message size, i.e., the physical layer payload, is 127 bytes. In the WirelessHART standard at most one message could be transmitted in a timeslot of 10ms. So the maximum data rate per channel in a WirelessHART network is 127 bytes per 10ms, which is 12.7kB/s. With 15 channels, the achievable bandwidth in the WirelessHART network is 190.5 kB/s. This is the concurrent data rate within the network. If we have 15 access points in a network, this could also be the achievable bandwidth between the network and the outside world.

In process control the process and control data is typically transmitted exclusively one per packets and in dedicated links. So the maximum achievable rate is 15 data packets per 10ms, or 1500 data packets per second. Devices can include 1 to 8 measurements per packet along with status and units information. The packets are published using Command 9. Since most devices will contain 2 to 4 4-byte float values, and Command 9 data packet with these contains a payload of 21 to 37 bytes, the maximum useful throughput of real-time measurement data is 31.5kB/s to 55.5kB/s (15*100*21 to 15*100*37).

Note here we do not consider using one channel for different communications at the same time, which is possible if the two communications do not interfere with each other. A typical WirelessHART network is of small size, though.

8.1.2 Packet Length versus Reliability

There is an argument that the message size should be small to achieve reliable wireless delivery. The argument is that the shorter the message, the quicker it will be transmitted hence the less chance that it will be corrupted by noise. On the other hand, it is preferable to combine several commands in one message to enhance throughput. This also could improve reliability as sending commands individually will expose each of them to interference. WirelessHART takes the latter approach.

In fact message length is less a contributing factor to reliability in noisy environment. Annex E of the IEEE 802.1.4-2003 standard provides the formula of bit error rate (BER) caused by the signal noise ratio:

$$BER = \frac{8}{15} \times \frac{1}{16} \times \sum_{k=2}^{16} -1^{k} \binom{16}{k} e^{\left(20 \times SINR \times \left(\frac{1}{k} - 1\right)\right)} \tag{8.1}$$

The equation's curve is also depicted in Figure E.2 in the document. With 2dB signal noise ratio, the BER is less than 0.000001. For a maximum message of 127 bytes, there is about one bit error per 1000 messages. A few bytes' difference between packet sizes does not make much difference with regard to loss rate.

The major contributor to message loss should be incidental interferences rather than constant background noise. Even with the maximum packet length, the size of the message is still small with regard to the BER calculation.

8.1.3 Channel Hopping

The physical channel used for communication is calculated as follows. The active physical channels are saved in an array. An array index for the channel is calculated each time a timeslot is chosen for a link. The index is the remainder after dividing by the array size the number that is the summation of the ASN (Absolute Slot Number) and an offset attribute of the link. Because ASN is monotonically increasing, in theory any active channel could be selected at a certain time. However, care must be taken to make sure as much randomness as possible is achieved with channel hopping.

For example, suppose all 15 channels are in use, a superframe is of size 15, and the first link in the superframe is a normal link with offset 0, then this link will always be assigned to the same first physical channel. However, if we change the superframe size to 16, prime to the number of channels 15, then this link will be assigned to the first physical channel in the first superframe repeat, to the second physical channel in the second superframe repeat, until all 15 physical channels are assigned to in 15 superframe repeats.

Note the active channel array does not necessarily contain all physical channels. Some channels may be blacklisted. So the array index number does not necessarily equal to the physical channel sequence number.

Links are defined within a superframe. The ASN numbers for a link is always a multiple of the superframe size plus the link's offset within the superframe. If the superframe size is a multiple of the number of active channels as shown in above example, the link will always use the same physical channel. This should be avoided, especially for advertisement messages. We want to advertise on all active channels. Setting the superframe size differently, a link could end up using a subset of the active channels. Ideally, the superframe size should be prime to the number of active channels, in which a link could be used on all possible channels.

8.1.4 Health Report

The device's ability to communicate with a neighbor is a key metric in forming and grooming the mesh network. Consequently, statistics are maintained in each neighbor table entry. These include average Received Signal Level (RSL); statistics on the packets transmitted and received and the timestamp of the last communication with the neighbor. For linked neighbors, RSL is calculated using an IIR filter using the following equation:

$$RSL = RSL - (RSL / RSLDamp) + (MeasuredRSL / RSLDamp)$$

Where MeasuredRSL is the RSL for the current packet and RSLDamp is the damping factor. RSLDamp must be a power of 2 and defaults to 64. For discovered neighbors (i.e., neighbors the device does not communicate with) the highest RSL value is returned. The value is weighted towards the latest communications.

Periodically a device sends out health reports of the neighbors to the network manager. The health value is primarily the RSL of its neighbors. The value is reset after each report. This is important for the network manager to optimize the network configuration. Up to four neighbors are reported each time, however, the network manager could request more.

There are two commands to report neighbor health. The information in these two commands is exclusive and the device must report both periodically. Command 780 provides statistics for linked neighbors, i.e., there are links configured with the neighbors. Other information such as communication statistics are also reported in Command 780. Command 787 provides only the signal strength of discovered (but not linked) neighbors. A device may discover a neighbor when it hears communication in a discovery link. A new device could discover neighbors by capturing their advertisement messages. In fact the join request message from a new device includes Command 787 data. If necessary, while joining, before normal links are configured a new device also sends out keep alive messages on the join link to keep synchronized as well as collecting health information.

Currently the WirelessHART standard does not provide the means for the device to report which physical channel is down. Such information could be used by the network manger to create channel blacklist. Because channel blacklist is used in channel hopping calculation, it is a challenge to change channel blacklist while the network is already setup. Besides, the WirelessHART standard is designed to tolerate non-performing physical channels.

8.2 Data Link Layer

8.2.1 Timeslot

A WirelessHART timeslot is 10ms long. Please refer to Fig. 3.4 and the symbols defined in Table 3.2. It is important that the device interprets and implements the timing points correctly for both communication and synchronization.

Source The sender waits for TsCCAOffset then checks if the channel is clear for a time length of TsCCA. If the channel is noisy, the sender will do nothing in this timeslot and proceed to the next timeslot. If the channel is clear, the sender switches to transmission mode within time length of TsRxTx. Then it sends out the whole message. Starting from the end of the message, the sender waits for TsRxAckDelay and starts listening for acknowledgement message. If no acknowledgement comes within TsAckWait, then it considers the transmission failed and acts accordingly. Otherwise the acknowledgement is received and processed. The acknowledgement includes the reception status and a time adjustment value. If the receiver is not the time source of the sender, the time adjustment value is ignored. Otherwise the sender will use this value to adjust its clock and start the next timeslot at the same time as the receiver does.

Destination The receiver waits for TsRxOffset then starts listening on the designated channel. If no message comes in within TsRxWait, then this timeslot is considered not being used and the receiver proceeds to the next timeslot. Otherwise the sender's message is received and processed. The receiver then prepares the acknowledgement message within TsTxAckDelay, after which the message is sent out. If the sender is the receiver's time source, the receiver will adjust its clock based on the difference between the expected message arrival time and actual arrival time (TsError) and set the next timeslot starting time in accordance with the sender's.

Interpreting the timeslot figure can be complicated. Here are some comments to clarify:

- The whole message includes, in sequence, four byte preamble, one byte SFD, one byte physical header, and the physical payload. The physical header contains the length of the physical payload. The start of the message (SOM) should be considered as the start of the preamble. Since no information of the message is available at the start of preamble, the receiver has to wait a little longer to find out if the incoming message is a legitimate one or just some noise. Since in many hardware platforms the first interrupt does not come in until the message length is known, i.e., the physical header is received, so in the implementation the listening timeout value should add the extra time to receive the physical header.
- In the WirelessHART standard the network manager could send the device a command 805 to skip CCA. In this way the sender should still transmit at TsTxOffset in the timeslot. Notice that no CCA is performed by the receiver before sending out the acknowledgement message.
- CCA is useful to detect interference from outside the network. If two devices share a link and try to send, both will pass CCA and transmit, in which two messages interfere with each other. The receivers will not get respective message and no acknowledgement will be transmitted. In this case both senders will fail the transmission. This is a rare case, in practice the two devices will not be exactly synchronized. Since the two devices are not exactly synchronized the slower one will defer due to CCA failure and the faster one will successfully transmit its message.
- If the receiver is not the time source, its time adjustment value in the acknowledgement message is not used. Regardless, the device must still follow the specification and send out TsError.
- TsRxTx is the time required for the radio to switch from transmit mode to receive mode, or vice versa. Its value is copied from the IEEE 802.15.4 standard. Most of the hardware chips on the market need less than this amount of time. It is up to individual device for how to implement; the WirelessHART standard cares that the message is sent at the right time.
- The acknowledgement message is sent TsTxAckDelay after the end of the transmitted message, not fixed distance from the start of message transmission as might be suggested in the timeslot figure.
- If time source replies, it must reply at sender's expected time, not the correct time. This way we have better communication success rate.
- In the specification, TsAck's value 832μs is based on transmitting 26 bytes, the size of the whole acknowledgement message. During join procedure the new device's long address will be used, which is 8 bytes, 6 bytes more than the short address. In this case the TsAck will be 1024μs.
- The key used for an acknowledgement message should follow what is used in the data message by checking its key bit.

- Two terms use in the specification, TsTxWait and TsRxWait, refer to the same definition.
- A stack implementation should strive to meet the exact time points defined in the timeslot. If it behaves (sends, starts wait, stops wait) within the range of ±100μs, the stack implementation will be considered to have met the specification. For example, if a device acknowledges between 900μs and 1100μs after the end of the received message, the device is compliant; if the device acknowledges at either 899μs before or 1101μs after the end of the received message, the device is considered to be noncompliant with the standard and as such will not be certified.

8.2.2 Links

The communication type in a timeslot is defined by the links. There are four types of links. They are normal, broadcast, join, and discovery.

Normal link The normal link is the most common link. It has a source address and a destination address. The source uses this link to transmit a message to the destination.

Broadcast link A broadcast link is associated with a device, who is the sender of the link. Upon this link the device sends out unacknowledged broadcast messages. The broadcast address 0xFFFF is used as the destination address.

Join link A join link is also associated with a device. The device could be the source or the destination. These two kinds of links are advertised in the advertisement message, which contains necessary information for a new device to join through this device. A new device extracts necessary information from the advertisement and uses them to complete the join process.

Discovery link A discovery link is a special link to maintain connectivity among the devices. The primary purpose of the discovery link is to allow devices to discover new neighbors. After a random length of time a device will use a discovery link to send a keep-alive message to a neighbor that it hasn't talked with for the longest time. A discovery link defers to all other link types if the same timeslot could be used for both. If not transmitting, a device will listen on discovery links. If the keep alive is not addressed to itself, it will still listen to record the talking neighbors' communication status

In the WirelessHART standard a link is used to describe the MAC layer connection between two neighbor devices. Connection is used to describe network layer connection between two peers that could talk to each other via intermediate nodes.

Links could be shared in which multiple senders compete for a link. The back-up mechanism is in place to handle retries. In shared normal link, there is only one receiver but multiple senders. Join links are shared in the sense that multiple joining device could compete to talk to the proxy device. Discovery links are also shared in which a device could either compete to transmit in a link or listen on the same link.

Links could be added to an active superframe. But an existing link can only be removed in an inactive superframe. A link could not be updated other than being removed then added back.

8.2.3 Synchronization

Since the clocks within a device may drift slowly, two perfectly synchronized devices must readjust their clocks on an ongoing basis. This is built in by the WirelessHART standard during any communication in a timeslot.

During a communication in a timeslot, the receiver will timestamp the arrival of the sender's message, which is compared with the expected arrival time it calculated based on the timeslot specification and its own clock. The difference TsError, which is measured in µs, will be the clock asynchrony between the two devices. If the sender's message arrives earlier than what the receiver is expecting, the TsError is positive, otherwise TsError is negative. If the sender is the time source of the receiver, the receiver will subtract TsError from its own clock.

TsError will also be encoded with two bytes in the acknowledgement message. Since the sender's message must be sent in the receiver's listening window, TsError is limited by the size of the window. Two bytes are enough to encode the maximum allowable drift. If TsError is non-negative, the two bytes will be the non-negative integer presentation of TsError, otherwise the two bytes will be the non-negative integer presentation of $0xFFFF - |TsError|$.

If the receiver is the time source of the sender, the sender will add the TsError value in the acknowledgement message to its own clock for adjustment.

The ultimate time source is the gateway; and the ultimate time source that talks wirelessly is the Access Point (AP). If there is more than one AP, they must also be synchronized. The synchronization between APs and gateway is not specified. If this has to be achieved within the wireless network, then one should be the main time source and the rest should synchronize just like other devices in the network. In cases where APs are put at different corners of the network to improve throughput, they might not talk directly. In these cases the APs will still be synchronized with the GW. It is also possible to have non-time-source APs synchroniz with neighboring network devices.

8.2.4 Keep Alive Interval

Two neighboring devices synchronize with each other on every communication. If they do not have information to exchange for some time, they should exchange keep alive messages to remain synchronized. According to the specification:

> "Devices shall not require a Keep-Alive more often than once per 30 seconds while temperature is varying 2º C per minute or less. … This corresponds to approximately a compensated clock accuracy of 10ppm or better."

In this section, we shall look at how two devices maintain synchronized. In WirelessHART two devices are synchronized as long as they continuously communicate successfully within timeslots.

A message starts with a preamble of 128μs, followed by 32μs of SFD field, then followed by the data package. For CCA, we have some extra μs before the preamble. For two devices to communicate, the receiver must start listening before the sender's preamble is started (Case A) and the sender's physical header must be already transmitted before the receiver is supposed to stop listening (Case B). Suppose y is the time difference between two devices at the start of the timeslot, we have two corresponding equations:

$$y + TsRxOffset \le TsTxOffset \tag{8.2}$$

$$y + TsTxOffset + tHead \le TsRxOffset + TsRxWait \tag{8.3}$$

The two cases are depicted in Fig. 8.1. Please also refer to Fig. 3.4.

Use the default values from Table 8.1, we have $y \le 1000$ μs.

Table 8.1 2.4GHz IEEE 802.15.4 Timing and Specifications (excerpt).

Symbol	Value
tHead (preamble+SFD+PHR)	192μs
TsTxOffset	2120μs ± 100μs
TsRxOffset	1120μs ± 100μs
TsRxWait	2200μs ± 100μs

Assume that the clock accuracy of all devices are x ppm, i.e., the clock will drift x μs per second. Since the two devices may drift in opposite directions, the two devices have to synchronize (keep alive) every $y/2x$ seconds. With 10ppm clock, we have 50 second keep alive interval.

The standard defines a range of ±100μs for most of the timeslot timing data. This is to tolerate imprecision in device implementation, either in software or hardware. For example, even if a receiving device has perfect clock and is programmed to acknowledge the sender in 1ms, the actual acknowledgement message may still not be transmitted on the air exactly in 1ms. With ±100μs tolerance allowed by WirelessHART, we should further subtract 200μs from y, which gives us 40 second keep alive interval.

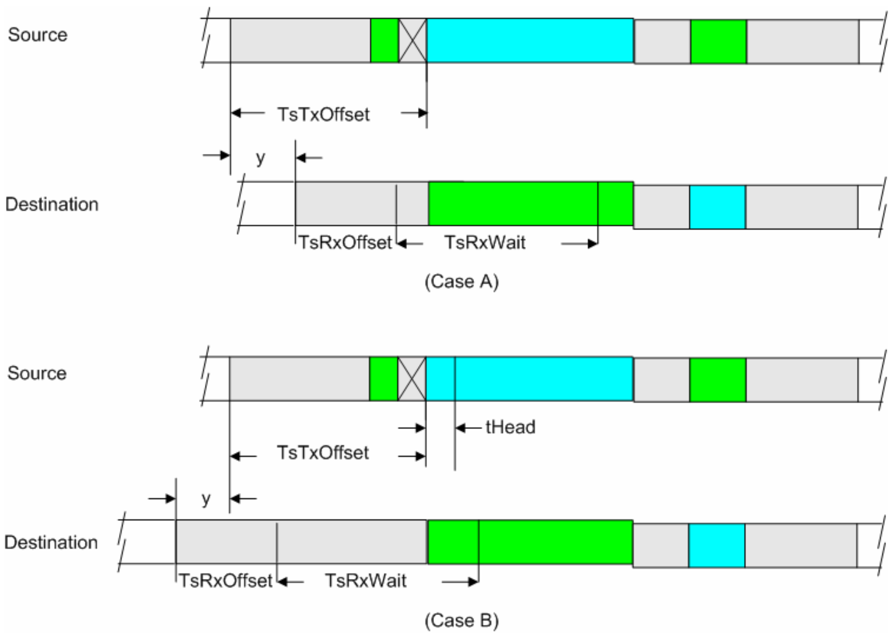


Fig. 8.1 Calculating Keep Alive Interval.

The story does not end here. In a mesh network, a device may be several hops away from the root time source. In the WirelessHART standard the network manager configures if a device is a time source of the other. The whole time source configuration forms a non-cyclical directional graph in which an incoming edge of a device is from its time source. There is at least on path from the root time source to any device in the network. If two neighboring devices are *z* hops apart on the time source path, then in the worst case they should exchange keep alive every *y/(x(z+1))* seconds. To keep the network in sync, the network manager must be careful when configure the time source graph and individual device's keep alive interval.

A joining device could use the join links to send keep alive messages before it is fully configured. Since join links are shared, the joining device must send keep alive more often in case it fails due to collisions. The network manager must also provide enough join links for this purpose, or configure the new device soon enough for it to use the regular links. If the joining device is to send keep alive messages, it might have to use the public key. The proxy device should be able to respond to them.

HART physical layer uses 2450MHz IEEE 802.15.4 physical layer, which requires a symbol rate of 62.5 ksym/s ± 40 ppm. If we are to use any IEEE 802.15.4 conformance hardware, we may have to accept 40ppm as the given.

The timing on acknowledgement message is based on the sender's time, i.e., always 1ms after the end of the sender's message. The clock drift is at most 0.002x μs in 1ms. This is not a factor as far as communication is concerned.

The clock in the device may drift uniformly. It may also drift randomly due to aging or temperature. Standard devices such as the DS32kHz from Maxim™ offer ±2ppm from 0°C to 40°C and ±7.5ppm from -40°C to 85°C. To keep the drift within required ppm, some age or temperature compensation mechanism must be built in the device. Above calculations are based on the worst case scenarios. In many cases the device should still talk even if the time between two synchronizations are larger than the calculated interval.

8.2.5 Clock Drift and Precision

There are two kinds of clock drift. One kind is that the clock is of the constant rate but the rate is slightly bigger or smaller than the standard one. For this case we could adjust by adding or subtracting some ticks periodically to compensate. This could make the clock run at the exact rate. The other kind is that the clock is slower sometimes but faster at other times. For this case we could do nothing other than letting the device constantly synchronize with its time source.

8.2.6 Broadcast Messages

If we define broadcast as one sender with multiple receivers, then the WirelessHART standard defines several different cases of broadcast.

Advertisement An advertisement message could be sent on any transmit link, on a normal, broadcast, join, or discover link. It could be sent whenever a link is available. By making use of all possible links we enable new devices to gather joining information and join quickly. The network manager could also set the advertisement interval or suspend advertising.

In the advertisement payload, the channel bitmap is not restricted to 2 bytes corresponding to the 16 channels at 2.4GHz. Future releases of WirelessHART versions will likely support more channels at other frequencies.

Broadcast A WirelessHART broadcast message is close to what broadcast means. Normally the sender uses the broadcast destination address 0xFFFF, and all devices that is configured with broadcast receiving link will listen to such messages. In the WirelessHART standard a device is allowed to send through broadcast link a message to a particular neighbor. In this case the designated device shall respond and all other listening neighbors simply ignore the message.

Discovery A discovery link could be used by any device to send out keep alive messages. The targeted receiver will acknowledge but other devices shall also listen to update neighbor health information.

8.3 Network and Transport Layer

In the ISO network model the transport layer is responsible to break up and reassemble large data packages. In the WirelessHART standard, however, large data is handled in a unique way which is defined in Specification 190, the Block Data Transfer Specification. The transport layer has only one byte as its header. The byte contains the acknowledge bit, response bit, broadcast bit, and 5 bits sequence number. Since the transport layer is simplified, it is conveniently specified in the network layer document.

8.3.1 Session and Transport Table, Who Owns Who?

The ISO network model puts a session layer on top of the transport layer, which in turn is on top of the network layer. And the transport table is managed under the network sessions. In other words, session owns the transport table. In the WirelessHART standard, however, there is no session layer and a session is defined within the network layer, which is under the transport layer. The reader needs to be aware of this when reading the Network Management Specification in places like its Figure 33.

Although the Network Manager can establish peer-to-peer sessions and routes between devices, this is not something that will be very common. In most cases routes will be established between the NM or GW and field devices. When peer-to-peer communications are used, one device will publish and the other device will catch the bursted message. An exception to this is with the handheld. The handheld is required to have a peer-to-peer session with network devices

8.3.2 The Security Layer

The WirelessHART standard uses encryption at the network layer to both encrypt the network payload and authenticate through the four MIC bytes. This is reflected conceptually with a thin security layer between the network layer and transport layer. The transport layer packet is encrypted. The security layer header contains a control byte, one or four nonce counter bytes, and four MIC bytes.

8.3.3 Broadcast and Response

In WirelessHART network broadcast messages can be used to reach all devices. Using broadcast messages both reduces the load on network resources and saves power. Changing the network key is such a command. However, broadcast links at the data link layer do not allow acknowledgement. So it is up to the network layer to make sure that all devices have received the command and respond back. What we see at the broadcaster is one request message and multiple response messages. If any device does not respond, either a new broadcast will be reissued or a confirmed unicast message will be sent. In the case of changing the network key this is important; otherwise devices that miss the broadcast request to change the network key will lose sync. Furthermore, the broadcast is sent with the broadcast session using broadcast session key. But the responses from the devices come back via the unicast session using the unicast key. Note that the sequence number in all the response messages must match that in the original broadcast message.

In some cases a broadcast messages is targeted to a single device. For example the sender wants to find a device with certain tag. In this case only one response is expected.

8.3.4 Block Data Transfer

As we have said that the WirelessHART transport layer does not segment/reassemble big data packages. This is achieved at the application layer via the block data transfer mechanism. When a device has a large amount of data to send, it first requests bandwidth for a period of time from the network manager using command 799. The network manager then allocates links for the transfer and replies to the device, telling it which route to use. The device then sends the block data via the route, one piece at a time. The receiver will assemble them back. How to break down and reassemble the block data is defined in a single document Block Data Transfer Specification (HCF_SPEC-190).

8.3.5 Transport Type Codes

Table 3 in the network layer specification (HCF_SPEC-085) deserves some words. It is duplicated in this book as Table 4.1. It defines the meaning of bit combinations of the transport byte. The commonly asked question is how to distinguish between Code 1 (Transfer Response) and Code 8 (Publish/Notify). Their bit settings are the same. This poses problem only for the receiver. The sender just sets the bits. The answer is that the receiver transport layer does not need to know and the upper application layer can tell the difference. Upon receiving such a message, the transport layer simply forward it to the application layer, which already knows if it is expecting a block data piece. We should point out that the sequence number in the transport byte could provide some clue to the transport layer. We could also guess from the priority: data publishing has high priority and block data transfer has low priority.

For other message types the transport layer needs to know what it is in order to proceed accordingly. For example upon reception of a Response-Unicast message, the transport layer must look up its matching Request-Unicast message.

8.4 Application Layer

8.4.1 Commands and Messages

The WirelessHART application layer is command based. The application payload to the network layer is a series of commands; each command contains a 2 bytes command number followed by the command itself. The command size is determined by the command number, so the receiver can successfully extract each command from the message. Basically if there are still data left in the message after extracting the current command, then it must be the start of a new command. The response message must contain the same set of commands, and the order should be the same.

Since the network manager configures a device via commands, the device's network layer forwards the commands to the application layer, which understands the commands and in turn forwards the configuration instructions back to the network layer. The network is fully under the control of its application layer which in turn is managed by the network manager.

There is one extra status byte in the beginning of the response in the command response data. In many cases the response message copies the content of the request message following this extra status byte. Since the response is usually longer than the request, it is possible that one message cannot hold all the responses. In this case no command in the request message will be processed. Instead, an overall error response message will be returned to the sender indicating the problem.

In some cases one of the commands in a set of aggregated commands will be invalid. In this case the correct behavior is to process each command in order and form the response message with a mixture of command responses that have succeeded and failed. It is important that devices respond to commands. If more bytes are left in the message after the commands are parsed, they are ignored for response messages and treated as an invalid command for request messages.

A device should only respond to gateway commands sent from the address 0xF981; a device should only respond to network manager commands sent from the address 0xF980. It is also important that command responses be returned to the device that initiated the command,

There are times when commands are variable in length. The command response messages must match.

8.4.2 Wireless verses Wired Command Formats

The original HART standard allocated only one byte for command numbers. The command format begins with one byte command number, followed by one bye command data length, followed by the command data. To account for new commands and maintain backward compatibility, it later defined a special command 31 which means the actual command number is defined in two extra bytes inserted after the command data length. So to parse a wired command message, you need to read the first byte to determine the command number. If it is 31, then find out the exact command number in the command data section.

In WirelessHART command format, the first two bytes are directly allocated for the command number, followed by the command data length, followed by the command data. We could translate the wired version to the wireless version. If the command number is not 31, then we simply set the higher command byte to 0 and copy the command data; if the command number is 31, we copy the two byte command from the data section as the first two bytes and copy the actual command data in the following. We could also translate the wireless version to the wired version. If a wireless format is of command number less than 256, we could simply remove the first 0 value byte and it becomes a wired version; if a wireless format is of command number bigger than 255, we could set the command number to 31 and set the two byte command number after the command length byte, followed by the actual command data.

The command response format in the wireless version is also different with regard to status bytes. First of all, there is a device status byte and an extended device status byte before all the response commands. They are defined in the transport layer, but probably need to be forwarded to the application layer. There are no such two bytes for the wired version. The real difference is within the command data. There is one response code byte in the head of each command response in wireless version while there is a device status byte after the response code byte in the wired version. You need to use command 0 to read the extended device status in wired version.

A command parser, given a command package, needs to be told if it is wireless or wired version. It needs to know if the second byte is also the command number byte. This might pose difficulties to third party gateway vendors as they have to look at the transmission medium or other means to find out the format of the commands coming from its clients.

8.4.3 Some Interesting Commands

In this section we describe some commands that are not explained elsewhere in this book.

Command 770 Command 770 turns on fast advertisement from the device. The single parameter is the duration of the fast advertisement. This is useful during initial network creation in which all is to be done is to make the devices join. It is also useful when a new device is put in and need to be joined. A handheld could send this command to a device, which will start fast advertisement. In the mean time, the device will forward command 770 to the network manager, which will again forwards the command to all devices.

Command 962 Command 962 sets a device with new short address. This is tricky because the short address is used for communication. In order not to cause the device to be orphaned and request rejoin, the network manager should configure all the links to and from the new address in the device and its neighbors, change the address, then delete links to and from the old address.

Commands 969, 974, and 976 Given a route ID, it could be either a source route or a graph route. Source route is configured with one command 976, which contains the ID and the address list. Graph route is configured with two commands 969 and 974. Command 969 associates a neighbor with a graph ID. Command 974 associates a route ID with a graph ID. A graph could be used for multiple purposes. For example, the up graph routing data from field devices to the gateway could be used for the gateway with one route ID; it could also be used for the network manager with another route. Normally a graph should be created before a route is associated with it. This causes people to insist that at least one Command 969 should be sent before a Command 974 is. They argue that the device must have the graph configured to accept associating a route with it. However, the specification does not preclude this, which could also be implemented. Strictly speaking there is no command to create a graph. Command 969 is to add an edge to a graph. A graph is implicitly created with the first Command 969. Command 974 should also implicitly create a graph if it does not exist yet. Only in this case the graph is created without any edges, which could be added by the subsequent 969 commands.

Command 972 Command 972 suspends a device. When instructed, a device stops communication at certain ASN and comes back at another ASN. The coming back time could not be precise as the device has to rejoin the network first.

Command 64512 Command 64512 returns about the radio manufacturer. In some cases, the wireless transceiver module is manufactured by a company other than the device manufacturer.

8.4.4 Burst Data and Delayed Response

In this section we describe the sequence of setting up publishing process data periodically to the gateway, which forwards to the host.

Certain WirelessHART commands can have delayed response in which the device responds to a command immediately indicating that it is still working on the command. It will send a final response once it is completed. During this period the device could send out intermediate progress responses; and the sender of the command could also inquire about the progress by sending out the same command again. Only a small set of commands allow delayed response.

A device will publish data according to the configuration of the control system. The device itself does not decide on how often or what to publish, which are configured through a series of commands 101 through 108. We assume that this is already configured in the device and describe what is next.

The gateway sends command 109 (Burst Mode Control) to the device asking the device to start bursting data. The device responds and starts sending out burst data immediately if there is enough bandwidth. Otherwise the gateway will get delayed response. Once the device gets bandwidth allocated by the network manager, it will send out the final response to command 109 and starts bursting data.

Here is how the device request additional bandwidth. The device sends out command 799 (Request Timetable) to the network manager, which replies with a route ID for the device to use. Usually no route is available and the device will get a delayed response. Then the network manager will add links to a new route or an existing route, which is sent to the device in the final response. The device must only use the designated route for bursting.

Note we have one delayed response command 799 enclosed in another command, command 109. However, for a single session there should be only one delayed response outstanding. A device is allowed to send other commands while waiting for a delayed response.

A device could be preconfigured with commands including command 109 before joining the network. It will then send command 799 immediately after it is joined and has normal communication with the network manager and the gateway.

Delayed response is also employed with adaptors in which all commands sent to the sub devices are forwarded by the adaptor. Because the response from the sub-device is slower through the wired network, the adaptor will first send a delayed respond back indicating that the command is been received and processed.

8.5 Topics that Cross Layers

8.5.1 The Encryption Algorithm

The WirelessHART standard adopts the CCM* encryption algorithm defined in the IEEE 802.15.4 standard. CCM* is an extension of CCM. It turns out that by the way of WirelessHART standard's adoption, it actually just uses CCM. However, the WirelessHART standard does not adopt how CCM* is applied to a message. To an IEEE 802.15.4 message parser, all WirelessHART messages do not have IEEE 802.15.4 security bit set and are not encrypted. As a matter of fact, all WirelessHART messages are data messages in the IEEE 802.15.4 standard. Instead, the payload to the IEEE 802.15.4 MAC layer, i.e., the WirelessHART data, is protected with CCM*.

The WirelessHART standard applies encryption at two places, at the data link layer for authentication and at the network layer for authentication and encryption. In order to help facilitate the development of WirelessHART products and ensure their interoperability, we give here the details on how the encryption algorithm is applied.

8.5.1.1 Symbol Definitions

The WirelessHART standard adopts the CCM* encryption algorithm defined in the IEEE 802.15.4 standard. CCM* is an extension of CCM. The symbols used in the IEEE 802.15.4 standard differ a little from those in the encryption standard (Dworkin 2004, US FIPS Publication 197). The symbols defined here comply with both standards. Undefined symbols are copied from the CCM standard. In Table 8.2 the last two columns list the values for the WirelessHART standard if applicable.

Table 8.2 CCM Encryption Symbols.

Symbol	Definition	Value for DLL	Value for NWK
a	The additional data to be authenticated but not enciphered.	-	-
c	The octet ciphertext	-	-
K	The AES encryption key.	-	-
E(K, x)	The AES encryption of a 16-octets string x with key K.	-	-
keylen	The bit length of the block cipher key.	128	128
l(a)	The octet length of a.	10 ~ 121 (TX) 13, 19, 25 (ACK)	16 ~ 49
l(m)	The octet length of m.	0	0 ~ 95
L	The octet length of the binary representation of l(m). (CCM standard calls it q)	2	2
m	The message to be enciphered.	null	-
M	The octet length of MIC.	4	4
MIC	The message integrity code. MIC is a suffix of c.	-	-
n	The octet length of the nonce.	13	13
N	The nonce octet string.	-	-
$\|$	The concatenation of two bit strings.	-	-
$\oplus$	The bitwise exclusive-OR of two bit strings of the same length.	-	-

8.5.1.2 The Encryption Algorithm

The CCM* algorithm is defined by **Keylen**, **L**, **M**, and **n**. It takes **a**, **K**, **m**, and **N** as input. And it outputs **c**. For decryption, the inputs are **a**, **K**, **c**, and **N,** and the output is **m**.

CCM* is an IEEE extension to CCM. It is stated in IEEE 802.15.4-2006 that

> As a result, if M is fixed and the value M = 0 is not allowed, then there are no additional restrictions on N, in which case the CCM* mode reduces to the CCM mode.

CCM requires that **L** + **n** = 15. The WirelessHART standard defines **n** = 13, and **L** = 2 is derived from the equation.

CCM* encryption includes 3 steps, input transformation, authentication transformation, and encryption transformation; CCM* decryption includes 2 steps, decryption transformation and authentication checking transformation.

The relevant sections in the WirelessHART standard are TDMA Data Link Layer (HCF_SPEC-075) Section 8.4 and Network Management Specification (HCF_SPEC-085) Section 8.1.3.

The relevant sections in the IEEE 802.15.4-2006 standard are Section 7.6 and Annex B.

8.5.1.3 The Encryption Algorithm at the Data Link Layer

CCM* is used in DLL to authenticate the message from the sender and, if present, the acknowledgement message from the receiver.

The WirelessHART data link layer does not encrypt the data. **c** generated from CCM* is simply the **MIC**. **MIC** is appended to the end of the plain message text to form the MAC payload defined in IEEE 802.15.4. To an IEEE 802.15.4 interpreter, every WirelessHART message is an unencrypted data message.

Like the sender of a message who calls CCM*, the receiver of a message shall call the same CCM* function with the same input. The receiver then compares the returned **MIC** with the **MIC** in the received message. If they match, the message is authenticated. Otherwise the message is invalid.

The input data

a - It is the physical layer payload without the **MIC** and CRC component:

0x41	Address Specifier	Sequence Number	Network ID →
→ Destination Address	Source Address	DLPDU Specifier	DLL Payload

For acknowledgement message the DLL Payload is:

Response Code	Time Adjustment in microseconds (MSB first)

K – It is the network key shared within the network. For devices joining the network, the following public key is used: 7777 772E 6861 7274 636F 6D6D 2E6F 7267.

m – It is empty.

N – If the source address in the message is a long address, it is:

ASN (MSB first)	Source Address (MSB first)

If the source address in the message is a short address, it is:

ASN (MSB first)	0x00	0x00	0x00 →
→ 0x00	0x00	0x00	Source Address (MSB first)

Treat CCM* as a black box algorithm

If a CCM* library is used instead of a CCM* been developed in house, the CCM* function should be selected with the values of **Keylen**, **L**, **M**, and **n** defined above. Then c returned by the CCM* function with above input data should be the **MIC**, which is put in the DLL message.

Customized WirelessHART CCM* function

The following is the WirelessHART reduction of the CCM* algorithm:
Input transformation

- Form the octet string representation L(a) as the 2-octets encoding of **l(a)**.
- Right-concatenate the octet string L(a) with the octet string **a** itself.
- Form the padded message AuthData by right-concatenating the resulting string with the smallest non-negative number of all-zero octets so that the octet string AuthData has length divisible by 16.

Authentication transformation

- Form the 16-octet $B_0 = 0x49 \| N \| 0x00 \| 0x00$.
- Parse the message AuthData as $B_1 \| B_2 \| \ldots \| B_t$, where each message block B_i is a 16-octet string.
- The CBC-MAC value X_{t+1} is defined by $X_0 := 0^{128}$; $X_{i+1} := \mathbf{E}(\mathbf{K}, X_i \oplus B_i)$ for i = 0, …, t. the string 0^{128} is the 16-octet all-zero bit string.
- The authentication tag T is the result of omitting all but the leftmost 4 octets of X_{t+1} thus computed.

Encryption transformation

- Form the 16-octet $A_0 = 0x01 \| N \| 0x00 \| 0x00$.
- Define the 16-octet encryption block S_0 by $S_0 := \mathbf{E}(\mathbf{K}, A_0)$.
- If any of the above operations has failed, then output “invalid.” Otherwise, the encrypted authentication tag U is returned as **MIC**, which is the result of XOR-ing the string consisting of the leftmost 4 octets of S_0 and the authentication tag T.

8.5.1.4 The Encryption Algorithm at the Network Layer

The network layer payload of every message is encrypted with CCM*. The payload is encrypted in the source and decrypted at the destination. Routing devices on the data path simply forward the encrypted payload.

Since the payload is encrypted, unlike in DDL, the destination device of a message calls the decryption part of CCM* to retrieve the message.

The input data

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

Control	0x00	ASN Snippet	Graph ID	Destination Address →
→Source Address	[Expanded Routing Information]	Security Control	0x00 or 0x00,0x00, 0x00,0x00	0x00,0x00, 0x00,0x00

K – It is the session key shared between the source and the destination. For devices joining the network, the join key is used.

m – It is the NPDU payload.

N – If the source address in the message is a long address, it is:

0x01 for join, 0x00 otherwise	Nonce Counter (MSB first)	Long Address (MSB first)

If the source address in the message is a short address, it is:

0x01 for join, 0x00 otherwise	Nonce Counter (MSB first)	0x00,0x00,0x00,0x00,0x00,0x00	Short Address (MSB first)

Note the Nonce Counter is the 4 byte counter from the sender. Depending on the situation, there may be 1 or 4 bytes included in the message. If only 1 byte, the received should provide the other 3 higher bytes. This means that the receiver must keep track of the rollover of the one byte in the message.

WirelessHART requires that the nonce counter in the join request and the new device's long address be used to construct the nonce for encryption of the join reply. This is different from other messages. It is intended for more security.

Treat CCM* as a black box algorithm

If a CCM* library is used instead of a CCM* been developed in house, the CCM* function should be selected with the values of **Keylen**, **L**, **M**, and **n** defined above. Then **c** returned by the CCM* function with above input data should be the encrypted payload right-concatenated with **MIC**. The encrypted payload will be put in the NPDU payload position; the 4-byte **MIC** will be put in front of the payload in NPDU. At the destination, c is form by right-concatenating the encrypted payload in the received NPDU with the 4-byte **MIC** in the NPDU. If the decryption succeeds, the returned **m** is the decrypted payload.

Customized WirelessHART encryption

The following is the WirelessHART reduction of the CCM* algorithm:
Input transformation

- Form the octet string representation L(a) as the 2-octets encoding of **l(a)**.
- Right-concatenate the octet string L(a) with the octet string **a** itself.

- Form the padded message AddAuthData by right-concatenating the resulting string with the smallest non-negative number of all-zero octets so that the octet string AddAuthData has length divisible by 16.
- Form the padded message PlaintextData by right-concatenating the octet string m with the smallest non-negative number of all-zero octets so that the octet string PlaintextData has length divisible by 16.
- Form the message AuthData consisting of the octet strings AddAuthData and PlaintextData: AuthData = AddAuthData $\|$ PlaintextData.

Authentication transformation

- Form the 16-octet B_0 = 0x49 $\|$ N $\|$ 0x00 $\|$ 0x00.
- Parse the message AuthData as $B_1 \| B_2 \| \ldots \| B_t$, where each message block B_i is a 16-octet string.
- The CBC-MAC value X_{t+1} is defined by $X_0 := 0^{128}$; $X_{i+1} := \mathbf{E}(\mathbf{K}, X_i \oplus B_i)$ for i = 0, …, t. the string 0^{128} is the 16-octet all-zero bit string.
- The authentication tag T is the result of omitting all but the leftmost 4 octets of X_{t+1} thus computed.

Encryption transformation

- Form the 16-octet A_i = 0x01 $\|$ N $\|$ 0x00 $\|$ Counter i, for i = 0, 1, 2, …
- Parse the message PlaintextData as $M_1 \| \ldots \| M_t$, where each message block M_i is a 16-octet string.
- The ciphertext blocks $C_1, \ldots, C_t$ are defined by $C_i := \mathbf{E}(\mathbf{K}, A_i) \oplus M_i$ for i = 1, 2, …, t.
- The string Ciphertext is the result of omitting all but the leftmost **l(m)** octets of the string $C_1 \| \ldots \| C_t$.
- Define the 16-octet encryption block S_0 by $S_0 := \mathbf{E}(\mathbf{K}, A_0)$.
- The encrypted authentication tag U is the result of XOR-ing the string consisting of the leftmost 4 octets of S_0 and the authentication tag T.
- If any of the above operations has failed, then output "invalid." Otherwise, output *c* to be the encrypted message Ciphertext and the encrypted authentication tag *U* to be the **MIC**.

Customized WirelessHART decryption

The following is the WirelessHART reduction of the CCM* decryption algorithm:

Decryption transformation

- Form the padded message CiphertextData by right-concatenating the encrypted payload with the smallest nonnegative number of all-zero octets so that the octet string CiphertextData has length divisible by 16.

- Use the encryption transformation in above encryption section, with as inputs the data CipherTextData and the tag U, which is **MIC** in the received message. In other words, use CipherTextData in place of PlaintextData and use U in place of T.
- Parse the output string resulting from applying this transformation as **m**$\|$T, where the rightmost string T is a 4-octet string. T is the purported authentication tag.

Authentication checking transformation

- Form the message AuthData using the input transformation in above encryption section, with as inputs the string **a** and the octet string **m** that was established in decryption transformation.
- Use the authentication transformation in above encryption section, with as input the message AuthData.
- Compare the output tag MACTag resulting from this transformation with the tag T that was established in above decryption transformation. If MACTag = T, output "valid"; otherwise, output "invalid" and stop.

Output

- If any of the above verifications has failed, then output "invalid". Otherwise, accept the decrypted octet strings **m**.

8.5.1.5 The Lifetime of an Encryption Key

CCM requires that an **N** value is used only once within a specified context. This puts a limit on the valid lifetime of an encryption key. The WirelessHART standard defines key replacement. A key will be replaced long before this limit is reached.

At DLL the key is shared among the network. The **N** contains ASN and the source address. This implies that the key could be used as long as ASN does not overflow. ASN is the 5-bytes sequence number of the 10 ms timeslot. It overflows in more than 3 centuries. There should be no concern from this aspect.

At the network layer a key is shared in a session between the source and the destination. The **N** contains the 4-bytes nonce counter and the source address. This implies that the key must be changed after nonce counter overflows, which is more than 4 billion messages. It will take more than a year if a device sends out a message in every timeslot. Like in DLL, there should be no concern from this aspect.

8.5.1.6 Incremental Execution

In the WirelessHART standard a two way communication is completed within a 10ms timeslot. At the data link layer, a receiver of a maximum-length message must authenticate the message and prepare the acknowledgement message, which must be applied CCM*, in 1ms. It will be extremely challenging for the receiver to execute this in time. Besides, the processing power of the receiver may be low; it may be powered by batteries. A WirelessHART device is supposed to be low power and long lasting.

This is specific to the WirelessHART standard. The acknowledgement mechanism at the MAC layer in the IEEE 802.15.4 standard does not involve encryption. The receiver does not need to process the content of the message to reply. And the acknowledgement message, whose content is "Frame Control + Sequence Number + FCS", is not encrypted.

The encryption method is CCM (Dworkin 2004). The standard states that

> CCM is intended for use in a packet environment, i.e., when all of the data is available in storage before CCM is applied; CCM is not designed to support partial processing or stream processing.

The main reason for this statement is that the length of **a** and **m** must be encoded first. However, due to the way the WirelessHART standard is specified, we know their lengths long before the whole message is present. We could stream process the incoming message in 16-bytes blocks.

We do not need to wait until the whole message is receive to start CCM. Once the MAC message header is received, we could start creating the first octets of B_1 in *Authentication transformation*. The MAC header contains the message length, which is need for B_1. Then as message streams in, we could create the sequence of 16-bytes B_i's, in the mean time, the created B_i's could be fed to the encryption algorithm. The CCM authentication contains a series of AES encryption executions, each on a 16 byte block. With incremental decryption, between the end of the message and start of the acknowledgement, there are only AES executions on the last B block and the A_0 block if we assume that the processor could run CCM* execution faster than the incoming data stream, which is 32μs per byte.

Next we show that we could move the execution of A_0 block outside the turnaround period as well. We pre-execute some of the steps for incoming message processing and acknowledgement message creation.

For incoming message, the receiver could create A_0 even before the message comes in. And the AES execution on A_0 could be pre-computed.

For acknowledgement message the AES execution on A_0 could be pre-computed as well. B_0 could also be pre-determined as the message length is fixed. If both the source and destination addresses in the message are short addresses, then there is only one data block B_1. Otherwise there are two data blocks B_1 and B_2. If there are two data blocks, the content of B_1 could be pre-determined. For the pre-determined blocks, AES execution could be applied to them *a priori*. In conclusion, only one B block has to be encrypted after the incoming message is processed.

More observations:

- The methods could be applied to pre-compute the sender's message as well. The sender could process the message before the actual transmission timeslot comes. The sender could use the idle timeslots for these computation-intensive executions. The ASN value of the actual transmission timeslot instead of the timeslot when the encryption is executed must be used.
- Only A_0 is used in WirelessHART DLL. In other cases we could pre-computer other A_i blocks and run AES on them before actual encryption execution. For example, this could be applied to encryption at WirelessHART network layer and other places where AES encryption is used.
- For a device that is not the time source, we could set its timeAdjust to 0. Then the only unknown value is *Response Code* who could be one of four values. There are two choices. We could pre-generate all 4 possible acknowledge messages and send one out based on the incoming message. We could also only pre-generate the success acknowledge messages. If failure response is required, the device chooses not to send out any message. In other words, use no-reply for rejection.
- We could also run CRC along with when the encrypted message is generated to further reduce the execution in the receiver's short response period.

We should point out that using incremental encryption is exactly how we succeeded in making the Wi-HTest™ tool with a FreeScale processor. The Wi-HTest tool is a component of the testing suite from HCF. The Wi-HTest tool will control and test a WirelessHART device for its conformance.

8.5.1.7 Hardware Acceleration

Most of the hardware chips on the market for the IEEE 802.15.4 standard provide encryption hardware acceleration. A WirelessHART product could use it only if they allow general CCM or AES invocation rather than encrypting an IEEE 802.15.4 message directly. If only AES is implemented in hardware, the rest of the CCM* implementation still needs to be written in software.

Since WirelessHART applies CCM* at two layers, care must be taken that the whole encryption is atomic. A CCM* most likely will access the accelerator multiple times to finish a message. If the network is in the middle of CCM* when the data link layer starts using the accelerator, the network layer has to restart.

8.5.2 Message Life Time

A message's life time is determined by several factors.

TTL TTL (Time To Live) is actually Hops To Live. It defines how many hops a message could travel, this value is saved in the network header and decrease by one through each hop. If it reaches 0 then the message is discarded.

maxPacketAge maxPacketAge is a network layer attribute preconfigured by the network manager. A device compares it with a message's age, both measured in number of timeslots. A message is discarded if it is older than maxPacketAge.

retryCount and responseTimer They are also network layer attributes preconfigured by the network manager. A message is considered dead after responseTimer. Afterwards a reincarnation is sent out again until retryCount is reached. TTL and maxPacketAge determine the lifetime of one incarnation. responseTimer should be larger than maxPacketAge. The longest life of a request from the application layer is retryCount times responseTimer.

8.5.3 Retry

The WirelessHART standard provides retry mechanism on two levels to deal with failure. The data link layer retries until the message is forwarded to the next neighbor; the network layer retries until the peer responds back. There is a time limit on both retries. Failure will be reported for timeouts.

At the data link layer, a message from the network layer is associated with a timeout value. The data link layer will try to transmit it as soon as an appropriate link is available. If transmission fails because of CCA problem, no reception of acknowledge, or rejection indicated in the acknowledgement, the data link layer will repeat in the next available link until success or timeout.

The timeout value from the network layer is calculated based on the ASN snippet in the message and maxPacketAge. The ASN value of the message creation plus maxPacketAge is the lifetime of the message. Both the ASN snippet and maxPacketAge are two bytes. Care must be taken to avoid rollover error when forming ASN from the snippet if maxPacketAge is big. The data link layer simply retries until the message is no longer allowed to live.

If the network layer is simply forwarding a message, it does not retry itself. When data link layer returns a timeout, the message is already dead. If the network layer is the source, it will retry if no response message is received for a message that requires acknowledgement. If no response is received before timeout, the network layer will retry once. It is suggested that the network layer should wait until timeout even if it is aware of the failure earlier. If retry counter is reached, the message is considered undeliverable and error is returned to the upper layer.

For broadcast messages that needs no acknowledgement, they are considered successfully transmitted as soon as they are sent over the air.

8.5.4 MSB and LSB, Big Endian and Small Endian

If you look at a full WirelessHART message, you will notice that two bytes of the device address are in different order between the data link layer and the network layer. For example, if the device address is 0x0123, in the data link header it is byte 0x01 after byte 0x23, but in the network header it is byte 0x01 followed by byte 0x23. This is because the IEEE 802.15.4 standard follows small endian and the HART standard follows big endian. Since the WirelessHART standard conforms to the IEEE 802.15.4 standard, it has to use small endian when creating the data link layer header. But any data in the data link payload follows big endian.

What is endianness? According to "The Authoritative Dictionary of IEEE Standards Terms":

> Big endian is a representation of multibyte numerical values in which bytes with greater numerical significance appear at lower memory addresses. Small endian is a representation of multibyte numerical values in which bytes with lesser numerical significance appear at lower memory addresses.

In a message, earlier bytes correspond to lower memory addresses.

Another set of more common terms to express endianness is called Most Significant Byte (MSB) First which means big endian and Least Significant Byte (LSB) First which means small endian. Sometimes they are simply called MSB and LSB.

A more detailed explanation could be found in Wikipedia™ at http://en.wikipedia.org/wiki/Endianness. Unlike IEEE standards, the HART standard is an industrial consortium. A key function is to educate people to promote the standard. Despite the controversy around the authenticity of Wikipedia, its entry on endianness is considered very useful and included in the WirelessHART documents.

Given the fact that both co-exist in the WirelessHART standard, we should be careful when formatting or reading a message. For example, ASN must be MSB. Anything within the application layer commands must follow MSB.

> Rule of Thumb: In the WirelessHART standard all data are represented MSB except those to conform to the IEEE 802.15.4 standard, which should be LSB.

This certainly presents potential traps in programming. When formatting or retrieving a multi-byte integer, the programmer must know whether MSB or LSB is applied. To make matters worse, the multi-byte integer itself may be stored MSB or LSB in the memory depending on the processor used. For example, Intel processors for PCs are small endian while most embedded processors are big endian. Applications written on Windows handle integers in the WirelesHART messages totally opposite from applications written for devices.

The bit order within a byte is also sometimes confusing. In fact the IEEE 802.15.4 standard defines LSB to be Least Significant *Bit* and uses it to describe how bits are transmitted over the air, i.e., the lease significant bit is transmitted first. For example, all WirelessHART messages are of IEEE 802.15.4 data message type. If both source address and destination address are short addresses, the 16 bit frame control field is represented in its Figure 35 as 1000001000010001 and also transmitted in this order. However, when viewed as two bytes, they are 0x41 and 0x88.

Real numbers are represented strictly according to HART standard as a sequence of bytes, no confusions here.

8.5.5 Short Address and Long Address

Conforming to the IEEE 802.15.4 standard, the WirelessHART standard supports both long and short address. The two byte short address is dynamically assigned by the network manager when a device joins the network. The eight byte long address is, like a MAC address of any networking hardware, unique per device; a device's long address is fixed and different from those of all other HART devices in the world. The long address is created by appending the 5 byte HART Unique ID to the 3 byte Organizationally Unique Identifier (OUI™) assigned to the HCF. Long address is only used at join. As soon as a device receives its short address, it should stop using the long address. There is a reason for this. No long address is used to configure any device in the network. A device only has links to short addressed neighbors. The address mapping is only kept in the network manager, who may not even preserve it. If a device receives a message from a long source address, it cannot associate the address with any known neighbors.

The network manager sends the network key and short address to the joining device in the join reply message. As a consequence we could determine whether the public key or network key is used in a message: If one address is long, public key is used for message authentication.

8.5.6 Nonce Counter and Sequence Numbers

The WirelessHART standard defines three numbers to keep track of message correspondence, the IEEE 802.15.4 data sequence number at the data link layer, the nonce counter at the network layer, and the sequence number at the transport layer. Each serves a different purpose.

8.5.6.1 The IEEE 802.15.4 Data Sequence Number

The third byte of the IEEE 802.15.4 MAC header is the data sequence number (DSN). It is randomly generated per device. A device saves it locally. Each time a message is formed, the DSN is copied to the message and incremented by one. An automatic reply message must copy the DSN from the original message. The DSN rolls over after 256 messages.

The WirelessHART standard is slightly different. The LSB of the ASN is used as the sequence number. ASN is universal in the network. Any message transmitted in a timeslot will have pre-determined value, regardless of the device. This provides some guard against interference; if a message's sequence number does not match the timeslot it is transmitted in, something must be wrong. The WirelessHART standard does not use automatic reply feature in the IEEE 802.15.4 standard. However its acknowledgement message has the matching sequence number as it is transmitted in the same timeslot. In the WirelessHART standard, the sequence number is not incremental with messages, and it rolls over after 256 timeslots, i.e., 2.56 seconds.

8.5.6.2 The Nonce Counter

The 4 byte nonce counter is defined in the network layer. Each device has one nonce counter per session.

Nonce counter is used to keep track of the message flow. Each new message within a session is associated with the current nonce counter which is incremented by one after each message formation. The receiving side keeps track of the sender's nonce counter and knows which message arrives early and which ones are still not arrived. It keeps a sliding window of 32 counter values. The sliding window is shifted forward whenever a newer counter is received. Since messages can arrive out of order or be lost completely, this sliding window algorithm is used to facilitate the communications and eliminate duplicate packets. A message with the nonce counter more than 32 earlier is discarded as obsolete.

Nonce counter is also important for security. It is used to construct the nonce, which is used to run CCM* for encryption and authentication. Four bytes of nonce counter make sure that a nonce could only repeat after 256^4 messages. Give that a timeslot is 10ms, the earliest time to repeat is after $256^4/100$ seconds, i.e., 1.3 years.

To save message overhead, during normal session communication only the LSB of the nonce counter is included in a message. This means the receiver must maintain correctly the upper three bytes of the sender's counter. There are two tasks, one is to obtain the peer's initial three upper bytes, the other is to update whenever there is a rollover of the LSB in the latest message.

A session is set up in the device by writing command 963 (Write Session) to the device. Not surprisingly, this command contains a peer nonce counter value. So a device knows its peer's initial upper three bytes. Upon receiving command 963, a device also sets its own nonce counter to 0. So the network manager or the gateway, if it is the device's peer, knows the device's initial upper three bytes as well. Currently in the WirelessHART standard all sessions have network manager or gateway at one end. If device to device session is to be used in the future, the network manger will write command 963 to each of the device. Since command 963 only sets the peer's nonce counter, in this case both commands to the two devices must set nonce counter value to 0.

During the join process, i.e., with join session, the full 4 byte nonce counter is included in the join-request and join-reply messages. In fact, the network manager must use the same value in the join reply message to guard against attacks. A device will compare the nonce counter value received in the reply message to the value it used in the request message. If they do not match, the reply message is considered invalid.

Next we discuss how a device updates its peer's upper three bytes when the LSB rolls over. Suppose the latest LSB a device has of its peer is *x* from message A when it receives a new message B with LSB *y* in it. Message B could be before or after message A depending on if *x* is before or after *y*, which could happen because of rollovers. For example, if we assume $x > y$, A could be *x-y* messages after B, or *256-(x-y)* messages before B. We have to use common sense here. A and B shouldn't be separated by too many messages, which are supposed to be out of order through the network. If *x* and *y* are separated too much in both interpretations of the order of A and B, the network is probably already malfunctioning. The detailed handling of LSB rollover could be tedious and of many approaches, but we could assume that basically of *x* is near or equal 255 and *y* is near or equal to 0, then B is after A and an LSB rollover has happened.

If a device can afford extra computations, e.g., the gateway or the network manager runs on workstations, it could simply try to run CCM* twice, one with the upper three byte unchanged and the other incremented by one. Whichever passes the authentication is the correct interpretation.

8.5.6.3 The WirelessHART Sequence Number

The lower 5 bits of the transport byte is the transport layer sequence number. According to the standard:

> 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.

The WirelessHART standard is very restrictive on sequence numbers for acknowledged traffic. Each peer keeps its own sequence number and increments it by one after each message transmission. The receiver must copy the sequence number from the received message to the acknowledgement message. It requires that a device will only accept a message whose sequence number equals one plus the sequence number received in the previous message. It is mandatory that at most there is only one outstanding acknowledged message at any time. Once it is completed, the next message could commence. The exception is for delayed response. One final delayed response could be transmitted after many other messages are completed. Even for delayed response, the first response must be received before the sender could send out the next message. The sequence number will roll over after $2^5=32$ messages.

If the sequence number is out of sync, the session will be useless and a new session must be created to reestablish communication. For example, if an acknowledgement message is lost in the network then the session is broken. In this case the sender keeps using the past sequence number whereas the receiver keeps expecting the next sequence number.

Since re-establishing a session is costly, a device could try some tricks to avoid broken sessions. For example, a sender could try different sequence numbers if a reply is not received from its peer. What it does is resending the same message with a different sequence number, until the peer replies or it times out. One candidate number that must be tried is the next sequence number to deal with the case described above. Other numbers could also be tried, for example, the ones immediately before and after the current sequence number.

On the receiving end, the device will still process the message if its sequence number is not the next one to be expected. It will reject arbitrary sequence number still, but if the number is one less than the expected, or close to it, the receiving device could just accept it and use it as the new correct sequence number. The receiver could tell that a message is legitimate if it is received repeatedly; the sender has retry mechanism at the transport layer. This will keep the session alive.

8.5.7 Timestamp and ASN Time

The WirelessHART network is fully synchronized. A timeslot is exactly 10ms; all devices know exactly the time since the creation of the network when ASN is 0. If a device knows the wall clock time of ASN 0, it knows perfectly what time it is, and we will have no problem time stamping or sorting network events. This is achieved with Command 793 (Write UTC Time Mapping), which tells the device what time it was when the network was created at ASN 0.

8.5.8 Master and Slave

In the WirelessHART network a master is the one that sends out request command and a slave is the one that responds. This follows the definition in the earlier wired standard. Normally the network manager and the gateway are the masters; the field devices simply respond. The messages from the network manager or the gateway contain command data and the messages from the field devices contain command response data. Even the join request from a new device contains command response data and is considered a response. And the join reply message actually contains three commands to the device, for which the devices sends out a command response message.

There are, however, several exceptions in which the field device is the master and the network manager is the slave. One example is Command 799 (Request Timetable). The device sends this to the network manger asking for bandwidth, and the network manager replies with a route ID for the data traffic. Another command is 770 (Request Active Advertising), which is mostly used by a handheld device.

8.5.9 Broadcast and Unicast

The network layer defines broadcast and unicast messages. The data link layer defines broadcast and unicast links. This creates interesting combinations of message transmissions, all of which are legal in the WirelessHART standard: a network layer unicast message sent on a normal link, a network layer broadcast message sent on a broadcast link, a network layer unicast message sent on a broadcast link, and a network layer broadcast message sent on a normal link. Please also refer to Table 19 in the network specification.

The peer address in the network header is unique for broadcast. For a unicast session the peer's address is used for both transmitting and receiving. For the broadcast session, the peer is either the gateway or the network manager. When receiving a broadcast, the source address is the network manager or the gateway and the destination address is 0xFFFF; when transmitting a broadcast, the source and destination addresses are still the same as in receiving.

8.6 Other Topics

8.6.1 Memory Footprint

The eWirelessHART standard is an industrial strength standard. Lots of features are included to provide sometimes conflicting requirements, such as power preservation versus real-time responsiveness. The implementation of WirelessHART stack is going to be on the higher end of the memory usage spectrum of all low power, low data rate wireless mesh networks. Some of the IEEE 802.15.4 processors on the market couldn't even afford a WirelessHART stack. For example, our experience with FreeScale's processors tells us that MC1321™ is not a candidate for WirelessHART implementation. One of the big reason is that it only has 8K

RAM, far too little. We implemented the Wi-HTest tool for WirelessHART test on the JM128™ Codefire™ processor. Even Wi-HTest does not implement the full stack, we still had to struggle to fit data section within the 20K RAM provided by the JM128 processor.

Next we shall calculate a rough memory size the WirelessHART data structures take. We shall use our Wi-HTest implementation as the reference.

Data link layer

Table 4 of the data link layer specification defines the Minimum Table and Buffer Space Requirement, which we reproduce here in Table 8.3:

Table 8.3 Minimum Table and Buffer Space Requirement.

Description	Size in the Wi-HTest Tool (bytes)	Minimum Number Required
Neighbors	21	32
Superframes	12	16
Links	21	64
Graphs	8	32
Graph-Neighbor pairs	4	128
Packet Buffers	196	16

Adding all above together, the total memory size is 6112 bytes.

Network layer

Table 14 of the network layer specification defines the Minimum Table Space Requirement, which we reproduce here in Table 8.4:

Table 8.4 Minimum Table Space Requirement.

Description	Size in the Wi-HTest Tool (bytes)	Minimum Required
Sessions	53	8
Correspondent Device	2	1 per Session
Transport	13	2 Per Session
Routes	27	8
Source-Routes	17	2
Timetables	13	16

Adding all above together, the total memory size is 1106 bytes.

More than 7K RAM is required to store the minimum number of data structures in two stack layers.

8.6.2 Key Change

For security reasons, the keys must be periodically changed. There are two keys in discussion, the network key at the data link layer and the session keys at the network layer. They are set by Command 961 and Command 963 respectively. There are normally 4 session keys per device: unicast and broadcast session keys to the network manager and unicast and broadcast session keys to the gateway. We shall discuss how the new keys come to take effect during the normal functioning of the network. The join key is only used during join and not discussed here.

8.6.2.1 Network Key at the Data Link Layer

Since the network key is shared among all the devices, the network manager must make sure that the effective ASN of the new key is well after the Command 961 is completed for all devices. Beginning at that ASN, all devices will switch to the new key.

- If the ASN in Command 961 is in the future, normal operation is performed.
- If the ASN in Command 961 is 0 or the ASN value is missing in the command, the device will switch immediately to this new key. This is mainly used during the join process in which the new device switches to the new key as soon as it has received the join reply.
- If the ASN in Command 961 is older than current ASN, the device should reply with error code 0x42, discard the command, and continue using the existing key.

8.6.2.2 Session Key at the Network Layer

Command 963 is actually writing a new session. The network manager must make sure that the effective ASN of the new session key is well after the Command 963 is completed.

- If the ASN in Command 963 is in the future, normal operation is performed.
- If the ASN in Command 963 is 0 or the ASN value is missing in the command, the device will start the new session immediately and use this new key. This is mainly used during the join process in which the new device switches to the new key as soon as it has received the join reply.
- If the ASN in Command 963 is older than current ASN, the device should reply with error code 0x42, discard the command, and continue using the existing key. This is why the network manager must set the ASN late enough.

After a device has switched to the new key, the old key should still be kept for older messages that arrive afterwards. The old key could be discarded after the maximum message lifetime. The device uses the ASN snippet in the network header to determine which key to use. There is no need to try both keys.

Chapter 9 Discourses on the Mesh Network

Abstract In this chapter we delve into some details of selected topics on the network. They are grouped into device life cycle, routing, host communication, network management, redundancy, scalability, battery consumption, interoperability, and unwanted access. The topics are listed as section titles.

9.1 The Birth of a WirelessHART Mesh

A WirelessHART network is born when the gateway and the network manager are started up; and the network is alive to the outside world when the first access point transmits the first advertisement message. Technically, the gateway initiates the first communication with the network manager requesting configuration. So the network manager is the first in the network to be alive.

In practice, though, a WirelessHART network is born at the 0 ASN, which is determined on the first AP that is the original time source of all devices. The communication among the network manager, the gateway, and the APs are not wireless and could be proprietary. So when the first AP sets its ASN to 0 is totally up to the vendor implementation, but the network manager should record the real world time of this event.

Once the first advertisement message is transmitted over the air, the network comes to existence and other devices could join.

ASN value has 5 bytes and could last for more than 3 centuries (> 348 years) without overflow.

9.2 Device Life Cycle in the Network

The first and probably the most difficult thing for a WirelessHART device is to successfully join the network. It first should search for advertisement messages on all physical channels and synchronize with the network. Then it initiates the join process, which takes several message exchanges with the network manager via the proxy device. After it is fully configured by the network manager, it could then perform its duty in the plant.

The specification defines the APIs between each layer, but it does not specify the APIs in the device join process. It's internal to the stack and could be implemented different ways.

Next we discuss each step in more detail. We can see in the following that the join process is fully controlled by the network manager, which makes all the difference between a good and a poor join experience.

9.2.1 Pre-configure the New Device

A new device must be pre-configured with the join key and the network ID. There may be more than one WirelessHART network in the area. The device needs the network ID to know which one to join to. The device also needs the first key to encrypt the initial messages. The initial messages include setting the rest of the keys to be used, which must be protected between the new device and the network manager. Command 963 (Write Session) could be used to configure the join key; Command 773 (Write Network Id) could be used to set the network ID. Although both are wireless command, they could be sent via FSK modem to the device just like wired HART command.

A new device also needs to be instructed to automatically commence join procedure upon powered up. This is achieved with Command 771 (Force Join Mode).

A device could also be pre-configured on how process data is published. This is achieved with a set of Common Practice Commands (HCF_SPEC-127). Furthermore, if we send Command 109 (Burst Mode Control) at the end, we could enable the device to automatically initiate publishing process data after join.

9.2.2 Network Device Advertise

On the network side, it needs to constantly send out advertisement messages, which are used by the new device to synchronize with the network and extract all join related information. Any device in the network could transmit advertisements, which instructs how new devices could join through the device. The advertisement message contains the following information:

- The current full ASN number
- The 4 bit join priority of the capability degree for accepting new devices. The priority values is based on 4 factors, the number of hops to the gateway, the signal strength, the battery power left, and the number of descendants. It could be a weighed summation of those four.
- The channel map about used channels

- The graph ID that should be of the graph that the device uses to send message to the network manager
- The superframes and their join links. An advertising device's transmit join link should be used exclusively for the device to transmit. It is used to transmit advertisement, join reply to the new device, writing superframe/links to the new device, and other command before superframe/links are configured. An advertising device's receive join link should be exclusive for the device to receive. It is used to receive join request, response to join reply, and other commands before the superframe/links are configured.

Advertisement message could be sent on any transmit link. This could speed up new devices joining the network.

All above are configured and controlled by the network manager. The network manager uses Command 795 (Write Timer Interval) to write the advertising interval to the device. A value of 0 means advertising as much as possible; a value of 0xFFFFFFFF means stop advertising.

Not all superframes in the device contain join links; however, a device is allowed to put superframes without join links in the advertisement msg. Not configuring any join link might not prevent a device from transmitting advertisement.

Public key is used at the data link layer for advertisement messages because the new devices listening for them do not know the network key.

The WirelessHART standard has specified the ranges and purposes of the network ID values in the data link layer specification, Section 8.1.2.

9.2.3 New Device Synchronize

The new device will continuously listen on a physical channel for a period of time, and then continue to the next channel, until all channels are exhausted, or the join is initiated. If an advertisement message is received, the active channels will be known and the new device will skip listening to black listed channels.

Statistically there is a chance that a device may not hear an advertisement message before the timeout due to the channel hopping mechanism. First, an advertisement may never get transmitted over certain active channels. Second, when a device is listening on one channel the advertisement may be transmitted on other channels. It is important to advertise as much as possible during network forming stage, especially by APs or other mainline powered devices.

The device will try to synchronize as soon as one message is received that is from the same network and not an acknowledgement. It could not synchronize with an advertisement message whose transmission time depends on the varying size of the acknowledged message.

While only the advertisement messages' information is saved by the new device, the other messages received are used to tally the neighbor communication quality of the transmitting devices.

9.2.4 Join Request

Once the device has received advertisement messages, it could start requesting join. There may be more than one device allowing join requests, for which case the new device must select the best candidate according to certain criteria.

The new device chooses the device with the highest signal strength and join priority to request join with. Signal strength should take precedence.

The join request contains three commands in response format, command 0 (Report Neighbor Signal Levels), Command 20 (Read Long Tag), and Command 787 (Report Neighbor Signal Levels). They are encrypted with the join key, which is indicated in the security control byte. So any node receives it knows it to be a join request. It is indicated as graph routing and the graph ID is copied from the advertisement message. At the data link layer the public key is used.

Command 787 includes the list of neighbor devices that the new device considers good candidates to join through. The device actually receiving the join request is now considered a proxy device. The join request will be sent on a join link. So the proxy's data link layer recognizes it and forwards to its network layer. The proxy must forward it using its own routing path but cannot modify the network header, which is used for encryption. This is why the new device must use the proxy's graph ID, and the proxy must have a graph route to the network manger. If a device only has a source route configured to the network manager, it must not advertise to solicit joining devices.

As soon as the join request is acknowledged at the data link layer, the new device shall be ready to transmit keep alive messages, if necessary, to keep synchronized with the network. Depending on where the join process is at, the keep alive could use the join links; these keep alives must also use the public key.

Once ready, the new device must wait a random time up to 2 minutes before sending join request. Many new devices might compete to join at the same time with a proxy.

9.2.5 Join Reply

The network manager should know beforehand the join key used by the device. The message is authenticated with this key. The network manager might also know beforehand the device's ID, in which case other devices will not be allowed to join. The device's long address could be formed from the information in Command 0 response; the network manager could also check the legitimacy of the join request message by comparing the formed long address with that in the network header of the message.

Once the new device is authenticated to join, the network manager will send back the join reply message. Join reply contains three commands, Command 961 (Write Network Key), 962 (Write Device Nickname Address), and 963 (Write Session). Join reply is actually not a response but a request command for which the new device will send the response back.

Join reply is proxy routed. The network manager selects one of the devices included in Command 787 of the join request and uses this as the proxy to construct the join reply. The message is actually sent to the proxy as if the proxy is the destination. The proxy detects the proxy bit in the message and retransmits it on the join links.

The proxy chosen by the network manager for join reply may be different from the proxy the new device sends join request to. After all, the network manager even does not know to which device the join request is sent.

The new device should keep active all the join links of all the neighbor devices it has listed in Command 787. From the join reply the new device obtains the short address and will use it hence forth, including even in the response to the join reply.

Except messages related to a handheld, a long address is only used in the join request and join reply messages.

The new device also uses the keys immediately. In the response to the join reply, it uses network key at the data link layer and unicast session key for the command responses. The implied join session is over and the join key is no longer used until the next join.

The join reply belongs to the join session; the response to the join reply belongs to the unicast session. As a consequence, the network manager must continue using the join session sequence number in the unicast session.

At this point the device is joined. One interesting observation is that the key, superframe, and link configuring commands must be forwarded to the app layer who then writes back to the network layer. The network layer does not know if the join is complete until the application layer tells it.

A device listening on a join link should be prepared to receive messages using either public key or network key. When a proxy receives a message on the join link, it could be public keyed join request, or network keyed response to join reply. When a new device receives a message on a join link, it could be public keyed join reply, or network keyed for writing superframe/links. This is why the key bit in the DLPDU specifier is important; it indicates which key is used in the message. In most cases, however, a receiver could guess which key is used in a received message from other relevant information.

9.2.6 More Configurations

Write superframe and link

After the new device is configured its own normal links, it no longer needs to use the join links for communication, proxy routing is no longer used and the term proxy ceased to exist. In fact, the only time proxy routing exists is after join request and before the normal link is configured. The new device will stop using the join transmit link once it receives a unicast transmit link; the new device will stop using the join receive link once it receives a unicast receive link. After both types of links are configured, the new device shall also remove the superframes derived from the advertisement messages if they are not configured by the network manager yet. It is recommended that the first message after join reply is writing superframes and links, which is then the only message in addition to join reply that are proxy routed. In the other direction, only join request and response to join reply are routed specially by the proxy.

The network manager configures the new device links according to the neighbor list it provided in the join request. The new device could be configured to use other devices as communicating neighbors. The proxy is not necessary the only connected neighbor of the new device. In extreme cases, the new device and proxy may not have any link configured between.

Write route to the network manager

Before the route to the network manager is configured, the new device constructs a default graph route, whose ID is the one from the advertisement message and whose forwarding neighbor is the proxy. As soon as the network manager configures the new device with a route, the default one must be removed.

Write time parameters

The network manager must also configure the time source for the new device with Command 971 (Write Neighbor Property Flag), which may not be the proxy.

In order for the new device to timestamp data or events, the network manager should also write the global time to it using Command 793 (Write UTC Time Mapping).

Write gateway session and route

While the network manager controls the network, the device performs its duty through the gateway. The final stage in preparing a device for full participation in the network is to provide the joining device with a gateway session, a route to the gateway, and graph information.

9.2.7 Keep in the Network

Once in the network, devices must take actions to remain in the network. This is achieved through synchronization which happens every message exchange. It is also achieved through constant network adjustment by the network manager based on the neighbor health information collected by all the devices.

For security purpose, the network manager also periodically changes keys. The join key could also be changed at this time, which will take effect in the next join.

9.2.8 Disconnect

During the lifetime of a network there will be times when devices must be disconnected from the network. The device, the network manager, or a handheld can initiate the disconnection. The network manager could send a disconnect command 960 to the device. Or a device could actively disconnect itself by sending a data link layer disconnect message to all its linked neighbors. This message is handled within the data link layer; the neighbors shall indirectly tell this to the network manager with health reports. A device could also die, in which case its neighbors will report lost communication to the network manager, who deduces that it is disconnected.

9.2.9 Rejoin

Once a device loses synchronization, it has to rejoin. The original specification version 7.0 provided a Re-Syching state in which the device keeps the entire configuration and could get back quickly. This was later dropped to make it simple. Once a device loses synchronization its entire configuration is removed, and the network considers it to be dead. The device should attempt joining again as if this is the first time.

9.3 Routing

A mesh network is to transfer data from the source to the sink, which is called routing. There are many ways to route data, each of which suits different purpose. The objective is to reliably deliver data on time. WirelessHART provides source routing, graph routing, and superframe routing. Proxy routing is also used when a new device joins.

A session should exist between the source and the sink to make routing meaningful. Although the WirelessHART standard allows a session between two field devices, practically all sessions are between the field devices and the network manager or the gateway. In this section we only discuss the routing between the gateway and a field device. The network manager always talks to a device through the gateway. We also use the term up, such as up link and up graph, to describe path to the gateway and the term down, such as down link and down graph, to describe the path from the gateway.

9.3.1 Source Routing

Source routing is straightforward; the source route includes the whole route path in the message. Each device in the middle simply forwards the message to the next device on the path. This is mostly used by the network manager and the gateway that know the network topology and could construct the path. For a device to use source routing, it must be configured by the network manager, who writes the whole path to the device using Command 976 (Write Source-Route). The path will be associated with the destination address by Command 974 (Write Route). The device looks up the path for the destination device. If the path is configured as a source route, it will be source routed.

9.3.2 Graph Routing

In graph routing, a device forwards a message according to the 2 bytes graph ID embedded in the message. A graph ID represents a directional graph whose vertexes are the devices and whose directional edges are the transmission links from the sender to the receiver. The graph is constructed by the network manager. Each device on the graph must receive the information about the graph ID and the address of each neighbor device to which an outgoing edge exists in the graph. A device is supposed to select the first available link that matches a graph edge to forward the message, regardless of which neighbor it is. Any device in the graph that transmits its own data on the graph also must know the address of the destination device associated with the graph.

Obviously to make it work all paths in the graph must lead to a single destination device. It is the duty of the network manager to construct the correct graph. It is also obvious that there should be no loops in the graph to avoid a message traveling endless in a loop. Without a loop, graph routing guarantees timed delivery of any messages. The standard, however, does not proscribe loops in a graph.

For example in Fig. 9.1 four devices A, B, C, and D form a graph with 5 edges. B could forward a message to either C or D. A message originated from device A will have the graph ID embedded and be routed through B, or C, or B and C in sequence to the destination D.

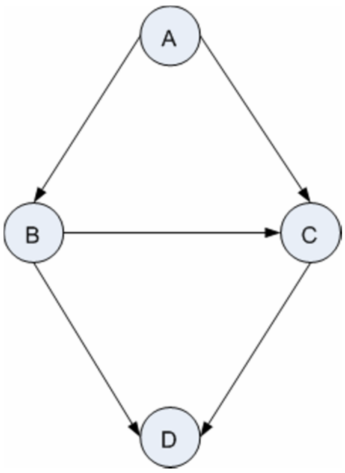


Fig. 9.1 Graph Routing.

Each field device requires a different down graph because it is the only sink. All field devices could share one up graph to the gateway. Furthermore, this shared graph could also be used to route messages to the network manager.

9.3.3 Mixed Routing

The source route and graph route information are defined in separate network header fields and could co-exists in one message. This provides good flexibility hence more chance for a successful routing. If one route hits a dead end, the routing device could pick another routing method. Besides, there are also special rules to facilitate data passage. For example, if the destination is a neighbor, a device could ignore the routing instruction and send the message directly to the neighbor. Then the questions would be do we prefer one hop to access point directly with low signal strength or multi hop with strong signal strength? All the special rules are summarized in Section 9.3.3.2 and Table 19 (Routing of Forwarded Packets) of the network layer specification.

9.3.4 Superframe Routing

A specialized form of graph routing called superframe routing was included in the standard. In superframe routing, a device forwards a message according to the superframe ID embedded in the message. The superframe is constructed by the network manager. Every device that is associated with any link in the superframe must receive the information about the superframe and the link. A device is supposed to select the first available normal transmit link in the superframe to forward the message, regardless of which neighbor it is. Any device that transmits its own data with superframe routing also must know the address of the destination device associated with the superframe.

In order not to change the message format, it is decided to make use of the graph ID field in the message for superframe ID. If the field value is no more than 255, then it is superframe routing, if it is 256 or bigger, then it is graph routing. As a consequence, a valid graph ID must always be larger than 255.

For example, if we want to use superframe routing for the 4 devices in Fig. 9.1, we could construct a superframe with only 5 kinds of links among them: from A to B, A to C, B to C, B to D, and C to D.

Converting graph routing to superframe routing

The examples about Fig. 9.1 show the equivalence of graph routing and superframe routing to some extent. For any graph routing, we could convert it into a superframe routing.

We define a superframe whose only links are normal links converted from the edges from the graph. The sender of the link will be the source of the graph edge, and the receiver of the link will be the end of the graph edge. Topologically this conversion preserves all the possible paths a message could travel.

However, the superframe routing thus derived does not match exactly with the graph routing at run time because the number of links available differ. For graph routing, any link in any superframe could be used. For superframe routing, only the links within the superframe could be used. We could estimate how many links could be used by an edge in the graph and define the same number of links in the superframe. However, the graph routing will compete for the links with other data traffic whereas superframe routing does not.

Converting superframe routing to graph routing

Similarly we could convert superframe routing to graph routing.

We construct the graph from the superframe as follows. The vertexes are the set of all the devices each of which has at least one related normal link in the superframe. For each normal link in the superframe we construct an edge from the sender to the receiver in the graph. We could keep the superframe around in which case its entire normal links could be used for the new graph routing; or we could remove the superframe and define other superframes to provide links for the new graph.

Likewise, topologically this conversion preserves all the possible paths a message could travel, but run time behavior does not match exactly.

Comparison

Graph routing and superframe routing are topologically the same. But they differ in minor ways.

- Graph routing is widely studied and easy to understand.
- Superframe routing has better isolation of data traffic. The data delivery is less affected by other traffic in the network, and vice versa. This might be better for maintaining real-time performance.
- Superframe routing on the other hand takes more device resources. A device has limited number of entries for superframes. It could not afford too many superframe routings. And it will take more computation to locate current link with more superframes.
- Both require careful management by the network manager. For both routing methods, a bad network manager may cause endless loops, dead ends, or unreachable destinations.

While superframe routing only uses the links within the superframe, source and graph routing can use links in any superframes. As long as routing methods coexists, all links are shared by different data traffic. In this way graph routing and source routing have advantage in that they have more links to use.

If in a network the only routing method is superframe routing, then all data traffic will have dedicated links, which will not be interfered with by other data traffic. And guaranteed end-to-end data delay could be calculated. Furthermore, if the routing path is redundant, guaranteed defense against single failure is also achieved.

9.3.5 Proxy Routing

We have explained in Section 9.2.4 (Join Request) that the proxy must include its up graph ID in the advertisement message, and the new device must form its messages as graph routing with this ID until it has its own routing configured. We also briefly mentioned proxy routing afterwards. Now we describe more about proxy routing.

Just like source and graph routing, the proxy routing data in the network header has its own field. The message will be routed to the proxy by the routing devices according to the source and graph fields just like normal messages that are destined to the proxy. In other words, the sender will set these routing fields exactly the same as if it is sending a normal message to the proxy. For source routing, the last address in the source routing list should be of the proxy; for graph routing, the graph ID should be of the down graph to the proxy. Only the proxy will process the proxy fields and finally forward the message to the new device. The destination address in the network header will be the new device's long address. The effect on the routing devices is that none of them will find the new deice in their neighbor list.

9.3.6 Broadcast Routing

Table 19 in the network layer specification (HCF_SPEC-85) lays down the rules for how to route a message. Here is how the network manager or gateway broadcasts a message. It will be a broadcast routing.

Form the broadcast graph

A broadcast graph is a unidirectional graph. It has one source. There is at least one path from the source to any node in the graph. For redundant graph, there must be at least two receive links for each device. The single source generates the broadcast message that will flow through all the edges of the graph. Each node in the graph will receive, process, and forward the message. The graph will be defined into a superframe.

The easiest way to form such a graph is by reversing the direction of all the edges of the shared up graph.

Configure the broadcast tree

The network manager sends the graph, i.e., the superframe, information to individual device in the graph. The link will be of broadcast type.

Form the broadcast message

The network manager or the gateway sets the message as broadcast in both the network layer and the data link layer, in other words, set the destination address as 0xFFFF. The graph ID in the network header shall be the superframe ID.

Forward the broadcast message

After receiving the message on its broadcast receive link, a field device, processes it as the recipient and reschedules to broadcast it on its broadcast transmit link.

It is not clear from the published specification, but the field device should be able to choose broadcast link in any superframe if the graph ID in the message is not a superframe ID.

9.4 Communication with the Host

The WirelessHART standard does not specify what communication medium must be used to connect the gateway and the host. Any network could be envisioned, such as Ethernet, Internet, Wi-Fi, Fieldbuss, etc. It provides a higher level XML syntax for them to exchange information. Because the WirelessHART standard is based on wired HART, the great advantage is that there is no need for the host to reinvent a brand new application solution. The host simply reuses its HART solution. What we need is to just let any underlying network to pass HART commands and command responses.

Instead of deploying new cables from the host to the gateway, we could use existing cables or, not a surprise, use wireless backbone. Next we describe a solution using deployed fieldbus networks.

To allow WirelessHART devices to be easily integrated into existing control systems at low cost, it is proposed that field devices be designed to be WirelessHART gateways. These gateway field devices would be designed to translate from the protocol used by the WirelessHART standard to the Foundation Fieldbus™ (FF) or Profibus DP protocols. These field mounted gateway devices could be wired and connected to the FF and Profibus fieldbus interfaces supported by most modern control systems. This approach would allow inexpensive gateways to be installed in the field next to the wireless transmitters associated with a process unit. The gateway device would draw power from the fieldbus segment. Using this approach, a control system that supports interfaces for FF or Profibus Fieldbus devices could interface to these gateway devices. This will allow WirelessHART devices to be easily and quickly installed in these control systems.

It is envisioned that these field mounted gateway devices could be manufactured with the same rugged and compact housing as a transmitter or similar field device, powered by the fieldbus segment and be located in the field physically close to clusters of wireless devices. These wireless gateways would be designed to contain protocol stack used by the FF or Profibus standard and also the WirelessHART stack. The FF or Profibus application layer would act as the glue between these two stacks, i.e., parameters communicated by the WirelessHART stack would be mapped to standard parameters of the function block application.

9.5. Network Management

Network manager is the center of the network. The network's performance could vary dramatically depending on how good the network manager is. In this section we list some key points related to managing the network.

9.5.1 Superframe Lengths

Communication links could be organized into superfames such as maintenance superframe, burst data superframe, join superframe, etc. Problem may occur if the-links in different superframes collide on the same timeslot. The WirelessHART standard allows this to happen and describes the priority to resolve. However, it is better not to having this happen at all. The network manager should correctly arrange the links. It could check for collision by looking at consecutive time slots for the number of the least common multiple of all superframe sizes. We know after that the pattern repeats.

A simpler approach is to choose superframe sizes from a harmonic chain (Kuo and Mok 1991). A harmonic chain is a list of numbers in which each number, except the first one, is a multiple of its preceding one. If x and y are two numbers from a harmonic chain, then either x divides y or y divides x.

With superframe sizes selected from a harmonic chain, any smaller superframe repeats within and aligns with any bigger superframe. It would be easier to generate schedules in which two superframe do not have conflicting links defined. For example, a scheduler could start from the smallest superframe. It then schedules the next smallest superframe, avoiding any time slots used by the smaller superframes. It repeats the same method with the next smallest superframe until in the end it schedules the biggest superframe.

In practice, we could select superframe sizes from the expression ab^n where a and b are two constant numbers, and n is any natural number. It could be verified that the sequence ab^0, ab^1, ab^2, ab^3, ... is a harmonic chain.

The superframe size should also be prime to the number of active channels so that a link in the superframe has the chance to be used at any physical channels. If the superframe size is a multiple of active channels, then a link will always end up in the same physical channel.

9.5.2 Allocating Bandwidth for the Host Application

Ultimately the goal of the WirelessHART network is to support host applications. Basically this means that devices will sense measurement data, communicate that information to host applications, and process requests from control applications. The network manager should allocate bandwidth for the data.

For a configured host control strategy, the scheduled links for sensor data are establish in this way: When the control strategy related to a device is downloaded to the device, i.e., telling it what burst data to report and how fast, the device will send request to the network manager, who creates links for it. WirelessHART standard does not define how to request links for control data sent to actuators. There should be a customized way for the network manager to translate the control strategy into superframe links for actuator data.

We could use the quarantine phase of the join process to do more: during start up, all devices are put in the quarantine state, then control modules are downloaded and schedule generated, then the schedule is sent to devices when all devices enter running state with final schedule. We could also use such quarantine-like state during mesh operating phase by simply disabling all its data superframes. Then we could do any maintenance work of the mesh, generate new schedule, and then bring devices back to running state again by enabling their data superframes.

9.5.3 Some Comments

Here are some comments related to network management functions:

- There is no ADD_NEIGHBOR command or DELETE_NEIGHBOR command. They are implicit, internal functions. A device is required to maintain its neighbor information based on the communication and configuration.
- To reduce latency in a multi-hop path, the network manager could try to cascade the links on the path by putting them consecutively in a superframe. There is a limit on this approach in that the link could be shared by other data traffic.
- Network manager controls the network via gateway. It is possible that one network manager application controls multiple networks.
- While the schedule should be gradually adjusted to network changes to minimize disruption to existing communication, optimization should be gradually applied as well. It was stipulated early on by the specification team that once in a while the network manager shall globally optimize the total schedule for the currently evolved network topology and application support. The network grooming criteria could be load balancing, power awareness, time scale, etc.
- The network itself is a resource and asset to the whole process plant. The network manager should represent the network to be accessed by the plant applications for network health trending, network troubleshooting, network health alerts, etc.

9.6 Redundancy

In process automation redundancy is always important. Some users put it as a prerequisite for any control system. The WirelessHART standard is fully aware of redundancy and provides ample support. Redundancy allows a system to continue functioning in spite of failure, and continuous functioning is critical in process control systems. The most common redundancy is double redundancy to guard against single failure. In other words, if two failures occur at the same time, the system might still fail. To guard against single failure, there should be one duplicate or standby for each component of the system.

9.6.1 Device Redundancy

In a WirelessHART network, failure could happen on the network manager, the gateway, or any of the field devices.

Network manager There is only one network manager per network. To provide redundancy, a standby network manager should maintain synchronization with the active. When the active is instructed to shutdown, or fails, the standby network manager will take over. Coordination of the switchover is implementation specific.

Gateway There is also only one gateway per network. To provide redundancy, a standby gateway should keep silent unless the active gateway dies or is switched over by instruction. How to coordinate the standby with the active is up to the people implementing this.

Access Point Although a vendor could implement standby access point, in the WirelessHART standard access point redundancy is realized by multiple independent access points. Each has its own direct connection to the gateway and possesses its own short address in the network. If one access point fails, data will be automatically routed through other access points.

Field devices The concept of redundant field device is beyond the scope of the WirelessHART standard (in practice redundant measurements are often used on critical loops). If two devices sample the same data, they will be seen as two totally different devices in the network, just like redundant access point. However, if a field device also routes data, its failure affects the rest of the network. Routing redundancy is used in this case, which is discussed in the next section. We could use other field device for alternative router. A powerful way to strengthen the mesh is to deploy extra devices as routers in the network.

9.6.2 Path Redundancy

Source routing provides one single path in the message. If one hop in the path fails, the message is lost. To provide redundancy, the source should, after timeout, retry a different source route.

Next we shall pay attention to redundancy in graph routing.

If any node except the sink in the graph has at least two forwarding neighbor, we could overcome single failure.

9.6.2.1 The Issue

With a directed graph, it is inherent that one link to the sink is simplex. Unidirectional is required to eliminate cycles and multiple forward links per node are required for redundancy. However, in the case of the last node to the sink, it is impossible to maintain the full redundancy for this device on the graph without creating a cycle. As a result, full redundancy is not available for data transmitted from this one device to the sink. If that simplex link goes down, the device has nowhere to send on the graph and the packet dies. The following graph (Fig. 9.2) shows an example of the issue. Notice that, no matter how many access points there are, as least one will have only one connection to the gateway without causing a cycle.

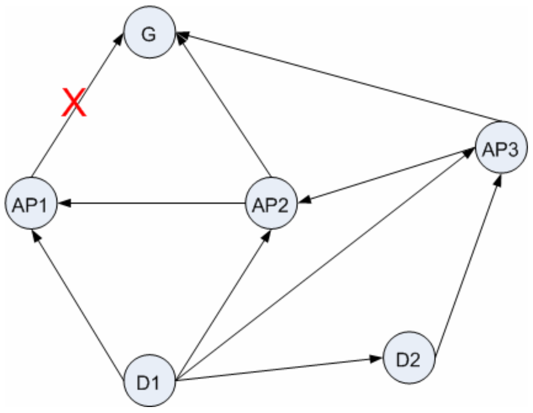


Fig. 9.2 Single Failure in a Graph.

The concept of the probabilistic scheduling is that not every send of a packet will travel the same route. This helps in that, if this issue does occur, it will probably be seen as a reduction in data update rate, not a complete loss of process insight. However, the un-deterministic nature of the schedule cannot guarantee the level of the reduced update rate. Once the network manager is aware of the down link, it can identify the problem and take corrective action.

In the case of the gateway bound up graph the issue is minimal and can even be addressed with more hardware. The gateway communicates directly with each access point over a wired network. Thus, any disruption in a link between the gateway and an access point is far less likely than disruption of a wireless link. Another benefit is that the Network Manager will be aware of the fault immediately. Thus, the Network Manager will be able to alert and take corrective action quickly. With the addition of a redundant wired network between the gateway and the access points, the situation can be eliminated. Of course if there is only one access point in the network, we are stuck again; there will be one non-access-point (NAP) node that has only one last hop to the access point.

For network with more than one access point, we could also prove that unless there is a node that talks directly to more than one access point, we face the same problem.

With regards to graph routed device-bound data, the problem still exists. The same bottle-neck occurs where at least one device will only have one link to the sink. Because of the detached location, it could be minutes before the network manager is aware of the down link. It is also recognized that device bound data is important in that it can affect control of the process. One option is to restrict device bound data to be transmitted with redundant source routing.

Note the overall discussion here assumes a well-behaved Network Manager, for which the only simplex node in a graph is this unavoidable last node to the sink. This further implies that there is at least another node in the graph that has this node and the sink as its forward neighbors in the graph.

The last statement from above paragraph could be proved. In addition, we could prove that, to provide redundancy except that simplex node, there must be at least two access points that are connected directly from at least one node.

The specification does not, however, explicitly prohibit cycles in a graph. The graph is generated by the Network Manager; and the implementation of the Network Manager is left to the systems vendors. With cycles, a data package may loop endlessly without reaching the destination. Note the specification also defines the lifetime of a data package, so the data will not loop forever, rather, it will be discard after a while, resulting in wasted bandwidth and power usages.

The specification also defines the routing rules. The following rule is important for the following discussions and quoted here (HCF_SPEC-85, Section 9.3.3.2):

> "If the Unicast NPDU destination address matches a neighbor then the NPDU must be routed directly to the neighbor… If this fails… then the packet must be Graph routed (if possible)."

This rule mitigates the impact of the problem of the simplex node: the node that has the sink and the simplex node as forward neighbors will always try the sink first. So any data traversed through this node is provided single failure protection. When the link between the simplex node and the sink fails, only data routed to the simplex from other path will be lost. A good network manager could construct the graph to avoid this case.

9.6.2.2 A Possible Solution

Let’s look at Figure 9.3. The graph is of the set of solid arrows. For destination node D, there are two immediate neighbors B and C, B’s alternative path goes through C, but C does not have alternative path to the destination. We could add the dotted arrow from C to B into the graph. This works in the WirelessHART standard as it is specified that, if the destination is the immediate neighbor, a node always try forwarding the message to the destination first. Problem arrives if the link from C to B does not exist.

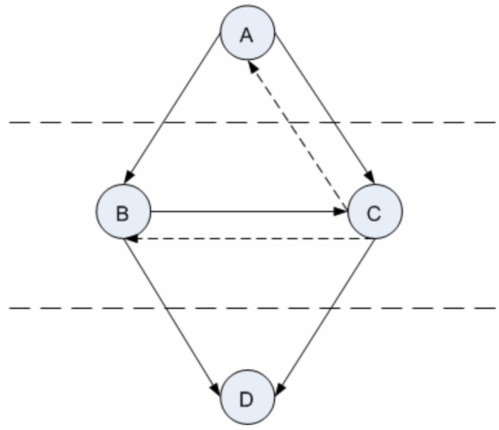


Fig. 9.3 Redundant Graph.

The solution is twofold. It takes the coordination’s of the network manage and special nodes.

First of all, the network manager tries to add direct links among the destinations immediate neighbors to create redundancy. The cycles among the immediate neighbors are allowed.

If it fails, the network manager will add links from the node without a redundant path back to the special nodes. In above figure the special node will be node A and the new link will be the dotted arrow from C to A. If there is an error for the link C to D, the message will be forwarded back to A, then to B and D.

The special node implements one feature: it records messages passed through it and to which neighbor it was forwarded. If a message is passed back to it, the special node will forward the message to another neighbor. This feature is common in many routing schemes and not difficult to implement in a WirelessHART device.

Let’s look at an example. If a message is routed to A, A randomly forwarded it to C. However the link form C to D is down. So C forwards the message back to A. A remembers this message and re-forwards it to B. B will not forward to C even though the link exists because D is its direct neighbor. So the message takes 4 hops from A to D.

By making use of special nodes, a network manager could generate graphs that is fully redundant yet provides deterministic maximum delay for a message. The network manager must be careful in creating the graph to avoid endless message loops. Once it has created the graph, it could also calculate the maximum hops a message takes from the source to the destination.

The idea could be extended to more complicated cases in that, as long as the cycles in the graph could be broken by the special nodes and there are deterministic hops for a message, the graph could be deployed. For example the backward link does not necessarily need to be from the destination's immediate neighbor. This could happen, for example, node A in above figure does not have a link to B or any other immediate neighbors of D. Then A must also have a backward link to some other node that leads to a redundant path to D.

The problem addressed here also applies to upward graphs which routes data from the field device back to the gateway, even though the gateway is connected via access points to the network and the access points are assumed to be fully connected to each other. For example, if every node that is an immediate neighbor to an access points can only talk to one access point. These nodes are in the same situation and a cycle must be created among them to provide redundant paths.

The special device does not need to remember the past messages forever. It could associate each one with its life time, and delete it from the record list once its life time has passed. According to the WirelessHART standard, if a message is still on route but past its life span, the routing nodes must drop it. So the special node does not need to worry if a message comes back after its life span.

9.6.2.3 An Alternative Solution

By further enhancing the special nodes, we could provide more power to the network manager. This additional feature in a special node is that it will send the message backward if all its forwarding neighbors are exhausted. For example, if nodes A and C in above figure are further enhances special nodes, we do not even need to add the link from C to A. C will return the message back to A and A will re-forward it to B.

This gives the network manager even more power in creating graphs. Of course it should be even more careful.

With the enhanced special nodes we could define much simpler graphs to provide deterministic yet redundant routing. For example a graph could consist of two separate single paths from the source to the destination with each node on the two paths as enhanced special node. If any link fails, the message will be returned all the way to the source, which simply retries with the other path.

Again, this feature is common in many routing schemes and not difficult to implement in a WirelessHART device.

The special node could also be enhanced to selectively choose neighbors to forward messages. For example, it could reuse the last successful neighbors; it could avoid neighbors that has problem in the past.

9.6.3 Broadcast Redundancy

Broadcasting is different that there is more than one recipient, and they do not acknowledge at the Data link layer. The sender could not tell if any receiver has received the broadcast. This is why a node must have two broadcast sources so that if it could not hear from one, it could get the broadcast from the other. We set up broadcast path by creating links in superframe. For example, one link could be configured to be the broadcast sender in one device and broadcast receiver in multiple devices.

9.7 Scalability

A WirelessHART network is small scale, low data rate, low power network. With nominal transmit power of 10dBm, the nominal non-line-of-sight receiver distance is 75 meters and line-of-sight distance is 200 meters. A typical process plant unit is about a football field, which is about 100 by 50 meters. From the size of the area, one WirelessHART network is able to cover one whole plant unit without hopping.

We calculated in Section 8.1.1 that the maximum achievable process data rate is 1.5k data items per second (31.5k to 55.5k data bytes per second). Normally one sensor provides one to four measurement and one actuator acts on one aspect for loop control. The fastest sampling rate allowed in the WirelessHART standard is 0.25 seconds. So we could afford in maximum 400 fast sampling devices. Using 40% fill factor, we have 160 devices.

One likely deployment outcome is that a plant unit may have several networks deployed, each of which has tens of nodes and located around a separate control unit. All the small networks form a cluster and controlled by the host in the control room.

9.8 Low Power Mode and Battery Life

WirelessHART devices are usually battery powered. The battery life (McCluer 2003) expectancy is important. To preserve energy, WirelessHART is designed to allow a device to enter low power mode when no communication is required. The network manager could configure the links in a device so that periodically no active links is present for a while, when the device could power down the antenna and freeze operation. It couldn’t be too long to cause losing synchronization. The keep-alive interval is the limit; a device could sleep no longer than this interval.

A battery's typical capacity is describe with unit Milliampere-Hour (mAh), i.e., how many hours with milliampere current. The electric power equals voltage times current, and the energy equals power times time. So the energy unit should be voltage-current-time. Assuming constant battery voltage, we have mAh as the conventional unit. A battery's life expectancy is then this value divided by the average current draw, and all we need is to calculate the average current usage of a device.

Batteries usually die abruptly rather than gracefully: Modern batteries are good at maintaining voltage so that the energy consumption does not decrease gracefully. With voltage U and a resistance R, the power usage rate U^2/R is a constant. Once the battery exhausts its energy, it simply dies suddenly. A defensive device shall calculate the future lifetime by measuring how much energy it has used. To do this, it measures the current drawn from it. The life left would then be the full battery energy mAh minus the used mAh.

There are three sources that draw current in a device, the processor, the antenna, and the actual device.

In this section we use a simplified example to illustrate the calculation. In this example, the device publishes every 15 seconds.

The Processor

We use MC1322™ from FreeScale as a reference. Its processor draws 3.3mA during normal operation. MC1322 has two sleep modes, hibernate and doze. They are similar except the clock precision. We have to choose doze mode, which uses external clock and consumes more current, to have a better chance of keeping devices synchronized after sleep. It draws 60μA while dozes; it draws 3.3mA when active. Assume that it is only active in the data publishing timeslot, the processor draws on average 62μA.

The Antenna

MC1322 has the system on the package; the antenna is built-in, which draws 24mA for receiving and 29mA for transmitting. We assume no external antenna here. The antenna is active, including switch time, when transmitting 43 byte data message (1568 μs) and receiving acknowledgement (1124μs). The antenna draws on average 4830μA.

The Device

The sensor device consumes power when it measures. The example we use is a temperature sensor. It takes 120ms and draws 500μA for each measurement. The device draws on average 4μA.

The Battery

We use two double-A batteries. One battery's capacity is 2000mAh.

Adding all together

For simplicity, we ignore the transition time energy usage. The batteries will last 816 hours, a little bit over a month. A process plant typically requests battery lifetime for over a year. Two double-A batteries are not enough for this simple example.

Finally we should point out that the energy provided by the battery itself varies. A lead-acid battery's life could be cut in half if operated above its nominal temperature. Other less prominent factors could also affect a battery's longevity, such as discharge frequency and duration, voltage variations, connections, leak by self-discharge, etc.

9.9 Interoperability and Interchangeability

It is important for a standard that different devices from different vendors work together. This includes interoperability in which one device communicates with another, and interchangeability in which one device could be put in place of the other without changing the network behavior.

Interoperability is essential to a standard. But interchangeability may be lacking in some standards. The WirelessHART standard mandates both and uses interoperability to encompass both:

> 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.

A device may have customized functionalities and they may even be used with customized applications. They should not interfere or replace what is defined in the WirelessHART standard and should not expect guaranteed performance for the customization over a WirelessHART network.

9.10 Unwanted Access to a WirelessHART Mesh

Before we start, let's emphasis that the WirelessHART standard is very strong on reliability and security. What we discuss next does not make WirelessHART venerable, rather it exposits on general risk faced by any wireless networks.

9.10.1 Jamming

The WirelessHART standard mainly deals with unintentional temporal jamming, which it handles pretty nicely. No wireless mesh can deal with intentional jamming in a good way.

The WirelessHART standard adopts the IEEE 802.15.4 standard and uses 2.4G Hz physical channels. The straightforward way to jam a WirelessHART network is sending noise on all 16 channels continuously. Of course, here we shall look at smart jamming; we only jam in particular points of time to render the network useless.

The fact that data link layer payload is not encrypted means that any sniffer could read the message content all the way to the network header. Of course, it still needs the network key to create fake messages, whose MIC depends on this key. The adversary knows the messages, but it does not know what application content is, which is encrypted at the network layer.

Since a WirelessHART network is fully synchronized, an adversary could disrupt the network by only jamming the channels for a very short time. The adversary first listens to synchronize with the network. This is done the same way as any new joining device. No security information is required to synchronize with any WirelessHART network. Jamming then becomes more efficient. For CCA enabled network, we only need to create noise when a device is performing CCA; for CCA disabled network, we only need to jam the preamble. Now we have a low cost and long battery (and intelligent) adversary. We only need 16 of such jammers to paralyze the mesh. Traditional jammers have to be on all the time, which is expensive.

The attack does not need to take long to disintegrate the network. Only a few time timeslots, i.e., longer than the maximum retry time, are enough to break a link, and further to get a device dropping out of the network.

Just like any new joining device, an adversary could also follow the channel hopping to jam specific links. The easiest hopping it could follow is the join links, which are read as plain text in the advertisement message. It could block any device from joining a network device.

The adversary could also infer the hopping sequence of any link. The active channel for any link is calculated using the known equation. With a 16-channel sniffer running for some time, we can identify the number of active channels on a device. Further, with known active channels and ASN, we can derive the link's channel offset. The superframe size could then be guessed from the transmission pattern between two devices. Sometimes a superframe is listed in the advertisement message, in which case the size is shown in plain text. With superframe size, the link slot number within its superframe could also be guessed.

Jamming could be a critical issue that might be a "show stopper" for the WirelessHART standard in some applications. Fortunately, in the WirelessHART standard there are methods for notifying all the nodes that jamming is occurring in other parts of the system when part of the system becomes unresponsive. For example, the network manager could make use of the channel blacklist, neighbor health reports, and path-down alarms. Because of the relatively low power and short range of 802.15.4 radios, jamming devices would have to be placed inside the factory or they would have to use very high levels of transmission power. Both of these are likely to alert plant personnel.

9.10.2 Key Discovery

There are two keys at the data link layer, the public key is public. The network key is shared among all devices. The network layer keys are unique per session, but the plain message header reveals which session is used. Unless an adversary gets the key by means outside the network, the only way to discover a key is to reverse engineer from samples. In each sample data, the nonce could be constructed from unencrypted texts. The MIC and the encrypted text are known. The task is to guess the key from the sample set. A guessed key is correct if it produces the same MIC on the encrypted text.

The WirelessHART and IEEE 802.15.4 standards use 128 bit AES key. In theory a huge sample set and a long time is required. Besides, the WirelessHART standard requires periodic key changes. This not only makes key discovery impossible, but also contaminates the sample set because the adversary does not know when the new key is used. The sample set may contain messages of different keys.

Chapter 10 Discourses in General

Abstract In this chapter we discuss general topics related to the WirelessHART standard. They are grouped into the relationship with OSI network layers, radio basics, centralized control, field survey, relationship with the IEEE 802.15.4 standard, coexistence, other fieldbuses, the limitations, security and reliability, and what it means to the user. The topics are listed as section titles.

10.1 The WirelessHART Standard and the ISO OSI Standard

All network protocols references the Open System Interconnection Reference Model (OSI) from the International Organization for Standardization (ISO). It separates the network functionalities into 7 layers. The OSI model is complete in that all network functionalities could find their counterpart in the model. It is also a great model in helping organize design of communications stack.

The complete encompassing stack and the functionality separation of each layers is also why actual implementations of protocols combine layers in the model and, in the case of the data link layer, separate out the MAC layer. Embedded system programming often trades off elegance for performance. Like other embedded applications, The WirelessHART standard does not have all 7 layers clearly delineated. These standards seek to simplify, eliminate, or cross layer optimize the features defined in the OSI model. Fig. 1.4 compares the WirelessHART stack side by side with the OSI model. The following are some factors influencing the deviation from the OSI model.

Power The power consumption is a major concern for embedded systems. Field devices are usually battery powered. Supporting a complete 7 layer stack is not the best solution.

Speed and Size The processors and memory size in embedded systems are small. A fully implemented OSI 7 layer stack is not an idea solution.

Security The OSI model was started in the 70s when security was considered important but not the most important factor. Today security has become the most important factor in networks. As a result of this many very sophisticated security technologies have been developed. The 128bit encryption and authentication algorithm CCM* is adopted by WirelessHART at both the data link layer and the network layer.

Real-Time Another important requirement in the embedded environment is the guaranteed data delivery by the network. Because of this the WirelessHART standard adopts TDMA and the whole network is synchronized. Time parameters are defined both within one stack and among the networked stacks.

As the result, most industrial network protocols are subsets of the OSI model:

Upper Layers While the lower layers are essential to in network design and implementation, fieldbuses usually simplify or completely get rid of some upper layers. Besides lowering the implementation footprint and save power consumption, most embedded stack do not need some of the features defined in the OSI model. The WirelessHART standard drops the presentation layer, rolling some of it into the application layer. It merges the session layer into the network layer and keeps a thin transport layer.

Cross-Layer Optimization The WirelessHART standard removes the transportation layer's support for data segmentation and re-concatenation. Large size data is handled by the application layer under the term of block data transfer. Consequently, the session is managed in the network layer.

Another optimization is in graph routing. Routing is defined in the OSI model's network layer. WirelessHART data link layer is also involved for graph routing. For graph routing, one device has alternative forwarding devices. Rather than letting the network layer choose which one to forward the message, the network layer sends the message to the data link layer with the graph ID. The WirelessHART standard saves the graph's forwarding device list in the data link layer. The data link layer picks the first link that talks to the neighbor that is in the forwarding list. The advantage is that if the transmission fails, the data link layer could retry on all possible forwarding devices until timeout without getting the network layer involved. Otherwise the data link layer could only retry on a single neighbor device and the network layer is left to retry alternative neighbors. In the WirelessHART standard the links are configured in the data link layer and the network layer may not know which device has the earliest link. Besides, there is message delay between the two layers. Letting the network layer decide on which neighbor to forward a message is inefficient. Even if the network layer has selected the earliest link, by the time the message reaches the data link the targeted link might already be in the past.

10.2 Radio Basics

10.2.1 Radio Basics

10.2.1.1 Electromagnetic Wave

Microwave, FM/AM radio, TV signal, cellular phone signal, light, etc., are all continuous electromagnetic waves, similar to the water waves in a pond. All the electromagnetic waves, including the light, travel at the same speed, i.e., they all travel at the speed of light. They are represented as sinusoid waves and differ by the vibrating frequencies. For waves the speed equals the multiplication of the frequency and the period. So we could also say that all the electromagnetic waves differ by their periods. Quantum-mechanic physicists teach us that the electromagnetic waves are also discrete quanta, but fortunately we do not need to understand that in our world of wireless mesh networks.

10.2.1.2 Transmit and Receive

Electromagnetic waves are continuous switching between the electric field and the magnetic field. The wave could be generated from periodic electric current change in an antenna. A radio transducer is a special electronic circuit that controls the charge and discharge on the antenna to generate different electromagnetic waves. A processor could then transmit desired waves by controlling the transducer. The antenna could also be triggered by the electromagnetic waves to generate changing current, which could be detected by the transducer. So a processor could also receive information from the electromagnetic waves. If we could embed data into the electromagnetic waves, we could then have one processor sending message wirelessly to another processor. This is exactly how it works in today's wireless applications.

The sender is characterized by the power of the wireless signal it transmits. This is defined as effective radiated power (ERP), measured in dB. ERP depends on the whole radio circuit, how much power is applied, how much is lost as heat in the circuit, etc. It also depends on how the antenna is arranged. Two antennae with the same ERP will be observed showing the same transmission capability. Commonly an antenna is omni-directional, it radiates on all directions. The direction with the strongest power is measured with antenna gain, which is the ratio of the power to the power of a theoretical isotropic antenna in the same direction. An isotropic antenna evenly transmits power in all directions. The power required for an isotropic antenna to transmit power the same as the ERP of the maximum gain direction is called equivalent isotropically radiated power (EIRP).

The receiver is characterized by its ability to pick up the sender's signal, the receiver sensitivity. This is measured with two criteria, the signal noise ratio (SNR) and the received signal strength indication (RSSI). The SNR is the power of received signal compared the power of the white noise. A good receiver is able to extract signal information in a low SNR environment. Sometimes the signal interference is not purely by white noise but from other legitimate transmissions on the same channel. This is measured in Signal to Interference-plus-Noise Ratio (SINR). The RSSI measures the power level of the received signal. A good receiver is able to extract signal information with a low RSSI.

The antenna's size and shape determines the best frequency and antenna gain distribution. The common antenna for the frequency used by a WirelessHART device is a single antenna of about 10cm long. Sometimes on board circuit are drawn as internal antenna to reduce physical size, for which the transmission range is relatively small.

10.2.1.3 Transmission Space

Now let's look at the space between the sender and the receiver. The best traveling path for the wireless signal is the straight line from the sender to the receiver, provided there is nothing blocking in the middle, i.e., there is line-of-sight (LOS) between the two. The receiver gets the best reception via LOS. In many cases, there are obstacles in or near the LOS causing reflections and signal losses. The term Non-line-of-sight (NLOS) is used to describe the existence of obstacles. The area between the sender and the receiver is divided into Fresnel zones, named after the scientist Fresnel. As shown in Figure 10.1, all Fresnel zones together shape like a prolate spheroid. Each Fresnel zone is a rotated ellipsoid layer. The sender and the receiver are at two ends of the spheroid. Different interference patterns result when an obstacle is put in different Fresnel zones. The first Fresnel Zone is the direct line between the two transceivers which, if not blocked, provides the strongest signals. The transmission space is also characterized with indoor and outdoor. Normally it is assumed NLOS exists indoor and many times LOS happens outdoor.

In a process plant, lots of equipments usually make long LOS impossible. The signal path is reflected quite often, and the receiver gets the same signal arriving multiple times in quick subsequence. This is called multipath. Multipath is fairly common. It might reduce the receiving quality because multipath signals cancel each out. We call this multipath fading. The strongest interference is often your own signal due to reflections.

The equipments in the plant not only blocks LOS, but mostly also contain metals and absorb signals. This weakens the signal, which is called path fading. There are many causes of path fading and there are many path fading models to study them. The natural fading is caused by distance. Normally the signal strength is quadruple inverse to the distance it has traveled.

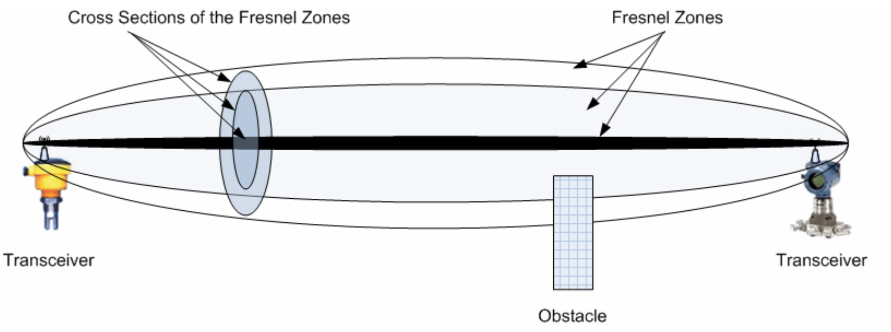


Fig. 10.1 The Fresnel Zones.

Another common interference comes from the communications of other neighboring devices, the so called near-far problem. Suppose a device has a close neighbor and a distant neighbor and the device needs or receive from the distant one. If the close one happens to be transmitting at the same time, due to the quadruple signal fading, the signal strength of the close one is much stronger than the distant one. Even the signal from the close one is like noise, the noise could be too big to be filtered out.

10.2.1.4 Encoding and Modulation

We have talked about the electromagnetic wave. But what we want is to pass information from the sender to the receiver. We attach data to the wave by manipulating the normal sinusoid wave in different ways. The receiver detects these manipulations and extracts the data. This is called modulation and demodulation. The electromagnetic wave of certain frequency being manipulated is called the carrier wave. For example, the AM radio modulates the amplitude; the FM radio modulates the frequency. We could also modulate the phase, the direction, or some combinations. AM/FM radios are analog. Digital data goes through another step, encoding and decoding. The original digital data is represented as a sequence of 0s and 1s. The encoding formats it into certain values that could be fed to the modulator. On the receiver side the decoder extracts the value spit out by the demodulator.

10.2.2 Spread Spectrum Modulation

It is expected that lots of factors could introduce errors during the whole sequence of transmitting some digital data from the sender to the receiver. One of the successful modulation methods is spectrum spreading. It uses multiple frequencies at the same time for a single communication task. Two typical ones are direct sequence spread spectrum (DSSS) and frequency hopping spread spectrum (FHSS).

A physical channel is a small band around a frequency. It is a single unit of communication medium. Different channels could be used for different communications at the same time. The spread spectrum technology uses multiple channels for a single communication. If one channel is bad, the rest will carry the communication through.

10.2.2.1 DSSS

In DSSS multiple channels are used simultaneously. The transmit power is divided among these channels, and the actual data transmitted on each channel is encoded with a pseudonoise code from the original data. The receiver uses the same pseudonoise code to combine the signal from multiply channels back to the original data. The interference on one channel will only affect its portion of the whole data. With some error recovery code, the receiver is then immune to the interference. Because a sender evenly distributes the energy among these channels with DSSS, its transmission appears more like white noise to others.

DSSS helps on the near-far problem. While the close neighbor’s transmission remains as noise, the distant device’s signal is multiplied by the number of channels used. Hence the SNR is enhanced by a multiple of the number of channels.

10.2.2.2 FHSS

FHSS does not spread its transmission power among multiple channels. Rather it pseudo-randomly picks one channel at a time among multiple channels to transmit. For example, the 8 bits of one byte could be sent on 8 different physical channels. The receiver hops channels the same way to maintain communicating. Again, because of built-in extra error correction bytes, the loss of one channel will not disrupt the overall communication.

FHSS also helps on the near-far problem. Because the close and distance devices hop channel at different sequence, in most cases they talk one different channels. In rare cases when the close device does transmit on the same channel and interfere with the distant device, the distant device simply forfeit such cases and recovers when it communicates successfully on other channels.

10.2.2.3 What the WirelessHART Standard Uses

The WirelessHART standard employs both DSSS and FHSS at different levels. DSSS is used for each message transmission and FHSS is applied on the sequence of the timeslots.

The WirelessHART standard adopts the 2.4GHz physical layer of the IEEE 802.15.4 standard, which uses a 16-ary quasi-orthogonal modulation technique. The final encoded sequence is modulated onto the radio using offset quadrature phase-shift keying (O-QPSK). The IEEE 802.15.4 standard defines 16 contiguous channels on the 2.4GHz frequency band. O-QPSK does not spread over all 16 channels and is not named DSSS, but it is DSSS-like in that it divides each channel into sub channels and spread the data among them.

The WirelessHART standard applies FHSS at a higher level. Rather than hopping channel after each bit, it hops channel after each message. To be more precise, it hops channel for each timeslot.

10.2.3 Media Access Control

Media Access Control (MAC) is the lower layer within the data link layer that controls the access to the physical layer. It controls when and how to send and receive messages.

10.2.3.1 Simplex and Duplex

A network node both transmits and receives. For many wired networks, a node could transmit and receive at the same time. They are called duplex. A node is simplex if it could only either transmit or receive but not both at any time. All WirelessHART devices are simplex devices.

10.2.3.2 CSMA and CSMA-CA

CSMA stands for Carrier Sense Multiple Access; CSMA-CA stands for CSMA with Collision Avoidance. In CSMA, a device ready to transmit first listens to the channel. If it is clear, the device then starts transmitting, otherwise it shall wait until the channel is clear again. MA in CSMA means more than one device could transmit at the same time on the same channel. In CSMA-CA the device will wait for a random period of time instead. This prevents continued collision of multiple devices all trying to transmit.

CSMA or CSMA-CA cannot solve the so called hidden node problem. If one device A could talk to two devices who could not hear each other, then both might think the channel is free and send to Device A at the same time. Both transmissions are lost because of the interference between each other at A. The two devices are hidden nodes to each other.

10.2.3.3 TDMA

TDMA stands for Time Division Multiple Access. The channel is divided into timeslots, a device uses it own assigned timeslot to transmit. The cause of collision is avoided because no more than two devices are transmitting at the same time. TDMA also solves the hidden node problem. Multiple Access here implies that the same channel is multiply accessed.

10.2.3.4 ARQ

ARQ stands for Automatic Repeat Request. It requires an acknowledgement to a transmission. The two way handshake guarantees the reception of a message. If no acknowledgement is received, the sender either retries or report failure after some retries. With ARQ a rare situation might happen in which the acknowledgement is sent but not received. In this case the sender thinks the transmission failed but the receiver thinks it is successful and acts upon the message.

10.2.3.5 What the WirelessHART Standard Uses

The IEEE 802.15.4 standard uses CSMA-CA. The WirelessHART standard adopts the IEEE 802.15.4 standard but it uses TDMA plus ARQ plus also some flavor of CSMA-CA. In process control industry guaranteed process data delivery is essential. TDMA meets the need. Each timeslot is 10ms and normally only one device per channel is configured to transmit in a timeslot. And the receiver shall send acknowledgement message back within the same timeslot.

CSMA also has its advantage for non-real-time data, especially when they have low occurrences. A WirelessHART network configures multiple access timeslots for CSMA-CA. These links are called shared links in which multiple devices could transmit and one device listens. Since all devices are synchronized, multiple devices might check the channel and find it clear at the same time. In this case they all transmit without being aware of each other. The collision is detected post priori by not receiving the acknowledgement message. If this happens, all senders will back off a random number of timeslots.

10.2.4 The Reason for 2.4GHz Band

WirelessHART network uses 2.4GHz frequency band, which is inherited from the IEEE 802.15.4 standard. With this band the effects of rain or snow are negligible within 1km. 2.4GHz is one of the ISM (industrial, scientific and medical) bands. They are unlicensed by worldwide government regulations. In other words, you do not need to pay for using the frequency. Another drawback for licensed band is that different countries have different uses for different bands. An international standard such as WirelessHART technology must be adopted as broadly as possible internationally. You cannot afford to have standard conforming devices usable in one country but not in another.

Of course, the drawback is WirelessHART network must compete with other networks in the same radio band. The good news is that advanced technology development in this field makes co-existence much easier, which we shall discuss later in this chapter.

10.3 Why Centralized Control

In centralized control a node does not generate its own schedule; rather, it executes a schedule generated by and downloaded from a central scheduler such as the base station. The node simply collects communication statistics and forwards them to the central scheduler. In distributed control a node is autonomous. It schedules its own tasks and data processing. It also processes requests from its neighbors and the host. In a centralized network, the central controller generates routing paths and distributes them to each node; in a distributed network, each node builds its own routing information by talking with each other.

The real-time nature of a process plant favors centralized control (Song et al. 2007). The WirelessHART standard defines a network manager that takes sole responsibility of managing the whole mesh. Unlike ad hoc wireless networks where a node wanders around, randomly joins and leaves, a wireless mesh in a plant usually have fixed device nodes that are put where they are supposed to sense or actuate. As a consequence, the data routing in a WirelessHART network could also be pre-determined, which makes both centralized configuration and guaranteed data delivery possible. The following points favor the centralized control in WirelessHART networks.

Run time scheduling

In centralized case, a device will receive exact data routing schedule from the network manager. Fixed time slots will be allocated to route data from the application. The network manager will also schedule the device to route data for other devices and send its own data. When not servicing the above schedules, the device is free to handle its internal tasks. In centralized case, the device will not cause any deadline misses.

If a device's schedule is determined by itself, it will treat the application's data with its own judgment. There is no longer pre-determined delay. The device may not necessarily assign higher priority to forward data of other applications. Furthermore, even if the device has dedicated time slot for data routing, it could not differentiate data from different applications. This means that applications sharing paths will interfere with each other at the path level, which poses more obstacles to meet real-time requirements.

Data path generation

Centralized management has the advantage in generating routing paths. By taking into account all possible links, the network manager could derive the best routing table for each node based on the load, number of hops, signal strength, and more importantly, deadline requirements. All the nodes need to do is to finds its neighbors and measure the signal strength with the neighbors and pass the information to the network manager.

It would be difficult if each node forms its own knowledge of the network by itself. A good path may be favored by all data transmissions and the nodes on the popular path will exhaust their battery before other nodes. People could always find good distributed routing protocols, but no matter what those protocols are, they could always be implemented on a central controller.

Device join and leave

We now look at how a node handles the come-and-go of its neighbors. Suppose before a device joins the network, the networks is running applications without any real-time violations. After the device has joined the network, it would request data transmission with certain nodes in the network. This added data traffic, if not managed correctly, could interfere with other applications and cause them to fail. In the distributed control, a neighbor of the new device may assign the new device's data the same priority as that of existing applications while it should have assigned the lowest priority to it. On the other hand, it is very difficult for the individual node to make this decision alone.

With centralized control, the newly joined device may not even be able to transmit data without admissions from the network manager. After the network manager decides that the addition of the new node will not affect existing applications, the new device could then safely be admitted into the network and transmit its data.

Distributed control is better when a device dies or voluntarily leaves the network. If the device is a leaf node, there is not much difference between the two control mechanisms. If the device is also a router, the data from its neighbors must be re-routed. Centralized control will be less responsive to re-routing demand as the new schedule would have to be updated by the central manager, while in distributed case, sensors could handle re-routing by themselves locally.

Collision avoidance

Another advantage of centralized scheduling is collision avoidance. In random channel access scheme such as CSMA, a node having data to transmit first listens on the channel, if the channel is clear, it starts transmitting. If the channel is occupied, the node has to back off and retry later. This mechanism works very well when network traffic is low. However, once many nodes try to transmit at the same time, there would be lots of collisions which may lead to missed deadlines. As for centralized scheduling, a timeslot is exclusively used by one transmission. Retry is only to deal with outside interference described in the next subsection.

Temporary interference

Temporary interference is common in wireless networks. There are many ways to mitigate this problem, such as DSSS and FHDS. However, those mechanisms could not eliminate the entire problem. We have to consider it within the scheduling policies in order to meet real-time requirements. When calculating the worst data transmission delay, we should take into account the retries and re-routing. For temporary interference, retry and re-routing requires similar work for centralized and distributed controls. In both situations, the sender will resend the packet to the same neighbor and, if unsuccessful, try the neighbor on the alternative path. Distributed control may have advantage in that it could retry or re-route according to the failure information. For example, it could choose alternative neighbor that it had the most successful transmissions in the past. A simple node with centralized control might retry the same failed route until told by the central manger otherwise.

Load distribution

In general centralized control reduces the scheduling computation in individual nodes, which in turn reduces the cost of the sensors and increases the battery life. Both real-time and non-real-time applications can benefit in this aspect.

10.4 Field Survey

Field survey could help in deploying a wireless network. There are two parts of the survey; one is to measure the existing wireless distribution in the target field, the other is to measure the path fading characteristics of the devices' signal transmission in the planned network topology. A typical process plant is a labyrinth of pipes and equipments, many of which reflect and absorb wireless signals. The field survey result of a process plant is expected to be highly customized. It helps to design and deploy a wireless network in a process plant.

Field survey also has its problems. First of all, it is inconvenient. Wireless network is supposed to simplify compare with its wired counterpart. Secondly, the plant is not fixed in stone. Although most of the process hardware does not change, other movable hardware such as material and products, vehicles, etc., could easily change the pattern captured by a spectrum analyzer. Onetime survey before the deployment may not suffice. Do we need to survey each time there is change in the plant? Thirdly, wireless devices cannot be deployed at arbitrary locations. Their locations are probably the input to rather than the output of the deployment configuration. This diminishes the value of a survey; a survey is supposed to help determine the best device locations.

The WirelessHART standard does not mandate field surveys. Device deploy locations are not subject to signal strengths or interferences. The WirelessHART standard uses the inherent mesh technology to achieve robust communications. If at any time there is certain non-redundant weak path, the solution is to add extra router devices or access points to those weak places and let the network manager to sort it out. Again, those extra devices could be put wherever it is convenient to set up.

10.5 The WirelessHART Standard and the IEEE 802.15.4 Standard

It has been repeated throughout this book that the WirelessHART standard adopts the IEEE 802.15.4 standard. Any WirelessHART message is a legitimate IEEE 802.15.4-2003 unencrypted data message. The IEEE 802.15.4-2006 standard is the latest version. In this section we summarize in more detail what is dropped, what remains, and what is changed.

10.5.1 WirelessHART Values in IEEE 802.15.4 Header Fields

The physical header of a WirelessHART message is exactly the same as that of an IEEE 802.15.4 message. Next we look at the MAC header.

Table 10.1 IEEE 802.15.4 Frame Control Fields.

IEEE 802.15.4 Name	Bits	WirelessHART Value	Comment
Frame type	$b_2b_1b_0$	001	Type is data
Security enabled	b_3	0	No 802.15.4 security
Frame pending	b_4	0	No frame pending
Ack request	b_5	0	No 802.15.4 Ack
Intra PAN	b_6	1	Within PAN
Reserved	$b_9b_8b_7$	000	-
Dest addressing mode	$b_{11}b_{10}$	10 or 11	Short or long address
Reserved	$b_{13}b_{12}$	00	Defined as frame version in 802.15.4-2006. Means compatible with 802.15.4-2003

Source addressing mode $b_{15}b_{14}$	10 or 11	Short or long address

Table 10.2 IEEE 802.15.4 MAC Header.

IEEE 802.15.4 Name	Bytes	WirelessHART Value	Comment
Frame control byte 1	Lower byte	0x41	See Table 10.1
Frame control byte 2	Upper byte	0x88, 0x8C, 0xC8	See Table 10.1. There is no case for 0xCC
Sequence number	1	LSB of the ASN	Not as 802.15.4 defined.
Destination PAN identifier	0/2	2 bytes	Network ID
Destination address	0/2/8	2 or 8 bytes	Short or long address
Source PAN identifier	0/2	0 byte	Absent
Source address	0/2/8	2 or 8 bytes	Short or long address
Frame Payload part 1	1	-	DLPDU specifier
Frame Payload part 2	0..111	-	DLL payload
Frame Payload part 3	4	-	MIC
FCS	2	FCS	CRC code

10.5.2 The Security Method

The IEEE 802.15.4 standard put the encryption method CCM* at the MAC layer as an option. The WirelessHART standard uses CCM* but not the way how it is applied in the IEEE 802.15.4 standard. The WirelessHART standard adopts the unencrypted IEEE 802.15.4 version and puts the security information MIC in IEEE 802.15.4 MAC payload, see Table 10.2. The WirelessHART standard does not encrypt the payload at the data link layer; it uses CCM* to generate MIC for authentication. The payload of the data link layer is the network layer data. Since the network layer only encrypts its own payload, both the data link header and the network header are plain text to anyone who captures a WirelessHART message. The IEEE 802.15.4 standard does not define a network layer and in a WirelesHART network the routing information need not to be encrypted. This is not a big security risk because the network payload is encrypted. The encryption method, not surprisingly, is also CCM*. The authentication MIC is also included in the network header.

It is common for commercial-off-the-shelf (COTS) IEEE 802.15.4 chips to implement hardware accelerator for the CCM* algorithm. In order for WirelessHART devices to take advantage of this, the chip should expose the CCM* API directly to the software implementations. If the hardware accelerator is further embedded in another hardware implementation of the IEEE 802.15.4 MAC security processing, we are out of luck and have to implement CCM* in software.

10.5.3 Maximum MAC Payload

A maximum-size IEEE 802.15.4 message is 133 byes. The physical header is 6 bytes. So the MAC layer maximum size is 127 bytes. The minimum MAC header and footer are 9 bytes and the maximum MAC header without the security bytes is 25 bytes. So the maximum MAC payload could range between 127-25=102 bytes and 127-9=118 bytes depending on the MAC header format. However, the IEEE 802.15.4-2003 standard take the safe approach and sets the maximum value to the smaller one 102 bytes. Later the IEEE 802.15.4-2006 standard relaxed this. If a message's frame version indicates the 2006 standard, a message could have more than 102 bytes MAC payload. Please refer to Table 85 of the IEEE 802.15.4-2006 standard.

A WirelessHART message is an IEEE 802.15.4-2003 one. Its data link layer section that corresponds to IEEE 802.15.4 MAC header is 11 bytes, or 17 bytes if there is one long address. The extra WirelessHART data link layer fields that fall in the IEEE 802.15.4 MAC payload are 5 bytes. The WirelessHART standard still allows the maximum message length to be 133 bytes, which means its maximum data link payload is 127-11-5=111 bytes, or 127-17-5=105 bytes. So if a WirelessHART message has longer than 102 bytes data that corresponds to the IEEE 802.15.4 MAC payload, it is no longer an IEEE 802.15.4-2003 one. However, it is still considered legitimate by the IEEE 802.15.4-2006 standard. Upon receiving such message, an IEEE 802.15.4-2006 receiver automatically corrects the frame version bits.

10.5.4 Other Comparisons

- The intended use for the IEEE 802.15.4 standard is personal area network (PAN). The range is about 10m. The WirelessHART standard targets process plant area which is larger.
- Consequently the transmission power for an IEEE 802.15.4 device is expected to be low, 0dBm is usual. The WirelessHART standard mandates 10dBm transmitters, which is the maximum allowed in the 2.4GHz band by many governments.
- The IEEE 802.15.4 standard allows 40ppm clock drift rate. A WirelessHART network is synchronized and requires more precise clocks. 10ppm is expected.
- The WirelessHART standard never defines meanings for the reserved bit or Byte in IEEE 802.15.4 MAC header. It is decided not to touch those as the IEEE 802.15.4 standard may provide new meanings in the future release.
- All IEEE 802.15.4 data-type message transmissions are preceded with CCA. A WirelessHART message is of IEEE 802.15.4 data type. But it is an option

whether CCA is used or not. Besides, there is no CCA for the acknowledgement messages.
- The IEEE 802.15.4 standard allows communication between meshes by using different PAN ID in the addressing fields. In WirelessHART mesh the communication is confined within the network.

10.5.5 Some Added Benefits with the IEEE 802.15.4 Standard

- We are able to implement WirelessHART devices with current and future COTS chips for the IEEE 802.15.4 standard. Some estimation puts the WirelessHART market volume to be two digits of magnitude less than the IEEE 802.15.4 market.
- We are able to take advantage of hardware improvement or acceleration for IEEE 802.15.4 MAC in current and future COTS chips.
- We could use any IEEE 802.15.4 compliant sniffer to read WirelessHART messages.
- However, other than from telltale signs, a sniffer cannot distinguish a WirelessHART message from other 802.15.4 compliant messages. There is not WirelessHART signature bytes defined.

10.5.6 Beacon

In the IEEE 802.15.4 standard, an existing network sends out beacon message to broadcast its existence and a new device listens for beacon to discover existing networks and network IDs. A new device could also send out beacon request messages, which the existing network will reply with beacons. The only way to find out the PAN ID of an existing network is from its beacon messages. A new network will actively scan the channels before start a new network. It sends out beacon request; other network replies with beacon message. A device reports PAN confliction if it receives a beacon message with the same PAN ID but wrong coordinator address.

WirelessHART device does not support beacons. An IEEE 802.15.4 conforming network coordinator could not find out if a network ID is used by a WirelessHART network. In addition, it could not find out if the channel it selects is free of interference. This means an IEEE 802.15.4 network, such as a ZigBee network or another WirelessHART network, could establish a network with the same network ID where a WirelessHART network is operating. They will then confuse each other's messages.

To solve this problem, we could arm the WirelessHART access point with beacon capability. The goal is to prevent other mesh network from selecting the same network ID on any of the channels the WirelessHART network uses. Note WirelessHART network is installed in a controlled environment. The problem could be avoided manually. What we discuss here is a possible automatic solution.

In this scheme, the access point periodically sends out beacon messages on all used channels. It needs not to be equally distanced. Just make sure that within certain time period at least one beacon is sent on each channel.

The beacon messages could be sent within an idle timeslot. A timeslot selected must be the one that contains no scheduled WirelessHART communication of the mesh.

In addition, the access point could also reply to beacon request. The gateway is put in listening mode during all the time when it is not scheduled to transmit. It will spread the time among non-blacklisted channels according to any scheme. Once received a beacon request message, it will reply in an idle timeslot.

The beacon message only provides the network ID used by the WirelessHART mesh. It should not provide any other information for new devices. It should not let them to try to join the network.

Some comments:

- The beacon or reply beacon could be sent in non-idle timeslots as long as the channel is not used. To find out if a channel is not used in a timeslot, we calculate the actual channels used by all scheduled transmission in that timeslot.
- The beacon could also be sent on black-listed channels. And we could also listen for beacon on black-listed channels
- Other devices in the mesh could also have above beacon support. Since they do not have overall knowledge of the network schedule, when and on which channel they send out beacon must be coordinated by the network manager. This may be useful when the signal from the access point could not reach the whole WirelessHART mesh area.
- The WirelessHART mesh could also use beacon mechanism to check for existing mesh before starting up. It could listen for beacon messages on all channels, or request beacons on those channels. If there is another mesh with the same network ID running in the vicinity, the WirelessHART mesh may decide not to start and show the issue to the user, or blacklist that channel, or choose another network ID.
- Above scheme could also be used for other mesh with different network ID. Before starting up, it may ask the user to take remedial action if there is a mesh with different network ID but using the same channel(s) of the WirelessHART mesh. Or it could automatically blacklist the used channels.

10.5.7 Configure IEEE 802.15.4 Stack for WirelessHART Stack

In this section we do a small experiment. We look at how to use an IEEE 802.15.4 implementation to support WirelessHART protocol. Although this is not practical, the exercise shall provide more insight comparing the two standards.

If we have full access to an IEEE 802.15.4 physical layer library, we should be able to build the WirelessHART stack directly on top of it. The advantage is that the development time is reduced. The disadvantage is that we have bigger code size, bigger data size, and possibly longer execution time, which leads to shorter battery life.

Most IEEE 802.15.4 software offerings tightly couples the physical layer and MAC layer together, only the MAC layer API is exposed. We now look at how to build WirelessHART on top of this.

Like using IEEE 802.15.4 physical layer, we need to get the incoming message start time. In case we could not get it from the native chip API, we could calculate it approximately using the time when we receive MCPS-DATA.indication from the IEEE 802.15.4 MAC layer. The time delay from the end of receiving to the indication could be constant as there is not message queuing and a WirelessHART device does not use IEEE 802.15.4 encryption. The problem is that we have to measure it for different hardware.

The following table lists the settings in IEEE 802.15.4 MAC for WirelessHART MAC.

Table 10.3 WirelessHART values for IEEE 802.15.4 Fields.

IEEE 802.15.4 Field	WirelessHART Value	Comment
phyCCAMode	2	Carrier sense only
phyCurrentChannel	varies	Set at the beginning of each timeslot
macMaxCSMABackoffs	0	Disable random backup
macMinBE	0	Disable random backup
MAC payload	<102	Otherwise FrameVersion will be set to 1.
macBeaconOrder	15	Disable active beacon
macSuperframeOrder	15	Disable active beacon
macGTSPermit	FALSE	Eliminate the impact of unintentionally sent beacon messages
macAssociationPermit	FALSE	Eliminate the impact of unintentionally sent beacon messages
PANCoordinator in MLME-START.request	FALSE	Not act as coordinator
macDSN	LSB of ASN	Need to set it every time.

macSecurityEnabled	FALSE	No IEEE 802.15.4 MAC encryption
MLME-RX-ENABLE.request	FALSE whenever not communication	-
macMaxFrameRetries	0	WirelessHART retries by itself
macPromiscuousMode	FALSE	-
macRxOnWhenIdle	FALSE	Only need to listen at designated time to save energy.

10.6 Coexistence

In this section we look at the impact on a wireless network by other wireless networks in the vicinity. We talk about how a WirelessHART network accommodates other networks and also how it affects other networks.

In Section 10.5.6 we have talked about how to proactively coexist through beacon mechanism. In other words, a network intentionally avoids interference with existing ones during setup. We have also talked about how an adversary could actively disrupt a WirelessHART network in Section 9.10. In this section, we look at passive coexistence, i.e., how to live in an existing environment in which multiple peaceful wireless networks coexist.

Two radio transmissions do not interfere with each other if their carrier wave frequencies are far separated. For example, if you are listening to a radio program, you are not concerned when you turn on the light. There are two ways a wireless transmission could be interrupted. One is with close frequency proximity, in which the interferer's signal tangles with the expected signal. In this case the interferer's signal does not need to be strong. The other is with brute force in which the interfering energy level is high enough that the receiver simply couldn't pick up the signal. One good example is microwave oven. When it is operating, it blasts strong magnetic waves and all wireless gadgets nearby are affected.

A WirelessHART network operates in the 2.4GHz band. Normally it should coexist peacefully with other wireless network operating in other frequency bands. Our discussion in the following will be about other networks also operating in the 2.4GHz band.

The license-freeness of 2.4GHz band spawns great advancement of wireless technologies. We've seen lots of standards and products developed for this. The unfortunate consequence is that it is getting crowded. And the coexistence issue becomes more and more important.

10.6.1 The IEEE 802.15.4 Standard

The most effective interference to a wireless network is from another network of its own type. The WirelessHART standard is defined to allow coexistence of multiple WirelessHART networks, each of which must have a unique network ID. If two networks happen to transmit at the same time on the same physical channel, both will fail. However, with the low data rate nature and pseudo-random channel hopping, the chance of such collision is rare. Besides, retry is built in the network to deal with failures. Human intervention could also reduce collisions. For example, we could allocate different physical channels to different networks. More challenging, we could co-ordinate the schedules on multiple networks.

Some observations of two interfering IEEE 802.15.4 messages:

- Two IEEE 802.15.4 messages cannot be sent at the same time on the same physical channel. The IEEE 802.15.4 standard uses pseudo DSSS.
- Two IEEE 802.15.4 meshes could coexist if they use different physical channels.
- Two IEEE 802.15.4 meshes using the same physical channel can co-exist poorly due to message collisions.
- For IEEE 802.15.4 network, adjacent physical channels can coexist; unlike in IEEE 802.11b™ at most 3 channels out of 16 can freely co-exist.

The WirelessHART standard is based on the IEEE 802.15.4 standard. There are also other network protocols, notable ZigBee™, that are also based on the IEEE 802.15.4 standard. The coexistence issue between the two draws more interest. It is perceived that the chance is high of a ZigBee network deployed near a WirelesHART network.

ZigBee (www.zigbee.org) network uses one physical channel; ZigBee Pro provides the way to change the channel while in operation. In any case, a ZigBee network's interference to a WirelessHART network is limited – this is because WirelessHART network hops channels. Conversely, a WirelessHART network's interference to a ZigBee network is also limited as the single ZigBee channel is only visited randomly by WirelessHART. They may hear each other's message. If ZigBee PAN ID is different from WirelessHART network ID, the messages will be gracefully dropped. However, they could not automatically discover each other's existence and network ID. We discussed this in Section 10.5.6. The field engineer must make sure different network IDs are used.

10.6.2 The IEEE 802.11 Standard

So far the most prominent player in the 2.4GHz band is IEEE 802.11a/b/g/n™, also called Wi-Fi. Wi-Fi is largely deployed in residential and office environment and making inroads in process plants. It is true that the 2.4G band is getting crowded; it is also true that people in the game are aware of this.

The Wi-Fi standard has wider physical channel band and higher transmit power. Generally a Wi-Fi network is affecting a WirelessHART network rather than being affected. The Annex E of the IEEE 802.15.4 standard provides authoritative analysis on its coexistence with other IEEE standards and proposed standards. Some of the highlights are listed here:

- For the nonhopping systems, large frequency offsets allow close-proximity coexistence (less than 2 m separation), while low-frequency offsets, or co-channel interference, require separation distances in the tens of meters.
- Transmit power level is the dominant factor in co-channel interference situations.
- The effect of the interfering signal on the desired signal is assumed to be similar to additive white Gaussian noise (AWGN) in the same bandwidth.
- As long as an IEEE 802.15.4 network uses unique spreading spectrum method in a channel, other wireless transmissions are simply noise.
- The wider frequency range per 802.11n channel is equivalent to two 802.11b channels used at the same time.
- There are four IEEE 802.15.4 channels that fall in the guard bands between (or above) the three IEEE 802.11b channels (n = 15, 20, 25, 26 for North America; n = 15, 16, 21, 22 in Europe). While the energy in this guard space will not be zero, it will be lower than the energy within the channels; and operating an IEEE 802.15.4 network on one of these channels will minimize interference between systems. The wider frequency range per channel in the IEEE 802.11n standard is equivalent to two IEEE 802.11b channels used at the same time. With the expanded channel width of an IEEE 802.11n channel, those IEEE 802.15.4 channels are no longer safe.

The energy threshold in CCA could be increased to improve transmission in channels close to interfering frequencies. The interfering frequencies will not impair decoding too much; setting the energy threshold too low will unnecessarily fail CCA and stop possible successful transmissions.

All above applies to a WirelessHART network. In addition the WirelessHART standard provides many more ways to coexist: channel hopping, retry, multipath, blacklisting, etc.

There has been a lot of study on the coexistence between ZigBee and Wi-Fi networks. The results are generally positive. One way to look at the coexistence between WirelessHART and Wi-Fi networks is to consider it equivalent to between ZigBee network plus channel hopping and Wi-Fi network.

10.6.3 Other Standards

Other wireless network protocols use different modulations. Their signals do not mean anything to a WirelessHART device and simply appears as white noise. The way they interfere with or interfered by WirelessHART network is the energy level imposed on each other. Those include Bluetooth™, Wibree™, RFID, UWB, and WiMAX™ networks, walkie-talkies, cell phones, base stations, repeater, proprietary radios, etc.

Bluetooth

Bluetooth standard is an evolving technology. The latest is version 3.0, which might include IEEE 802.11 standards or UWB. Here we use version 2.0 as reference.

A Bluetooth device also operates in the 2.4GHz ISM radio band. It supports timeslots and channel hopping. However, its channel is much narrower than that of an IEEE 802.15.4 device. The Bluetooth protocol divides the band into 79 channels (each 1 MHz wide) and changes channels up to 1600 times per second. The interference between a Bluetooth device and a WirelessHART device could be considered noise to each other and could be negligible.

RFID

RFID stands for Radio-frequency identification. The main use is for an RFID reader to identify an RFID tag that is attached to something. The wireless communication is point to point and any radio frequency, including 24.GHz, could be used. The application and usage pattern is totally different from a WirelessHART network.

UWB

UWB stands for ultra-wide band. It uses a large portion of the radio spectrum at the same time to transmit data. So it is received as white noise by a WirelessHART device. UWB is considered by many standards such as WiMAX and IEEE 802.15.4a standards.

10.6.4 Coexistence Test Scenarios

There have been many coexistence experiment tests already in the 2.4 GHz ISM band. A coexistence test could cover the following items:

- Receive signal strength
- Transmission range
- Test target as interferer or interfere.
- Traffic Pattern
- Channel Selection

- Frame Length
- Antenna gain
- Path loss

10.7 The HART Standard and Other Fieldbthus Standards

Process control systems went through several generations over the years. Fieldbus is a major milestone in which field devices are connected to the host through a network bus rather than each device commanding its own connection. Fieldbuses are industrial strength. They are more restrictive and demanding. Popular network protocols such as Ethernet could not be used directly in a process plant unless it is customized. With networked devices, much new functionality is developed. For example, some fieldbuses support control-in-the-field in which control modules run in the devices and no host is involved to complete a control loop. Many diagnostics and maintenance functions are enabled with fieldbuses, which saves customers huge amount of costs. Many different fieldbus protocols have been developed, such as Foundation Fieldbus, Profitbus, DeviceNet™, Modbus, etc.

HART communications protocol is one of these fieldbuses. It was initially designed to provide maintenance support over the existing 4-20mA twisted pair wires. While other fieldbuses require replacement of previous non-fieldbus control systems, HART bus builds on top of them. You could replace a classic device with a HART device and suddenly without other hardware retrofits the host gains access to a lot of device information. The HART standard allows continued process data to be handled the old ways. Because of the HART standard's simple and evolutional nature, it boasts the largest number of deployment in the world.

Wireless represents another still ongoing generation. Other than defining a brand new wireless bus protocol, the WirelessHART standard continues its tradition by extending the HART standard with wireless capability. A WirelessHART network is a truly mesh network with the latest wireless technology development. In the mean time it smoothly inherits from wired HART network, provides easy migration of the huge HART customer base to the wireless bandwagon. We are not aware of any significant activity from any of the other wired fieldbuses to go wireless.

With WirelessHART network, process and control data can no longer be transmitted the old fashioned way outside HART network. A legacy oddity is that, in a control loop, the periodic output data sent to the actuators are embedded in Command 79, which requires a response message.

10.8 What the WirelessHART Standard Does Not

The WirelessHART standard set it scope from the start. It does not intend to be an all encompassing wireless standard. A WirelessHART network is a low power, low data rate mesh network for the process control industry. For anything beyond that, the WirelessHART standard does not intend to apply, although its technology makes it capable of many of the non-targeted applications.

The WirelessHART standard does not apply to industrial manufacturing where high data rate reaches low millisecond or sub-millisecond levels.

The WirelessHART standard does not specify how things work beyond the mesh. The noticeable omission is that it does not standardize the backbone transmission between the gateway and the host. You will not see a wireless backbone data hauling protocol from the HART standard.

The WirelessHART standard stipulates the rules of a gateway or a network manager. It does not mandate how a gateway or a network manager is implemented. It does not mandate how the network manager configures the network. The consequence is that the WirelessHART standard could be wrongly blamed for inefficiencies because of a bad network manager implementation. On the other hand, an excellent network manager could produce a superb WirelessHART mesh.

10.9 Security and Reliability

When people talk about wireless, the first concern comes up is security (McCluer 2003). This is understandable. After all, anyone anywhere can pick up wireless signals. From a technical point of view, though, what is really concerned is reliability. The latest encryption technology applies in both wired and wireless world. It does not make it much easier to break CCM* algorithm because it is applied to wireless. We could have as good security in wireless as any other network medium. Wireless security may not be better that wired security, but the perceived security problem in wireless exist in the wired world as well.

The WirelessHART CCM* algorithm is copied from the IEEE 802.15.4 one which is based on the AES algorithm. This AES algorithm is defined by the United States NIST (National Institute of Standards and Technology) FIPS (Federal Information Processing Standard) Publication 197. FIPS 197 is compliant with the NIST FIPS 140-2, required by some government agencies and corporations.

The WirelessHART standard does not specify the communication between the network and the host or other outside applications. Security concern in the area is beyond the WirelessHART network and should not be used against it.

In a WirelessHART network a white list or a blacklist of devices can be defined. A device in the white list is allowed to join the network; a device in the black list is not allowed to join the network. If both lists are defined, a device must be in the white list but not in the black list to join. This is unrelated to the blacklist channels.

In the WirelessHART network the security manger manages all the keys. The security manger itself and the communication between it and the network manager must be secured as well, which is outside the wireless network and could be applied the best and latest technology in the security field. All keys in the devices are write only.

Reliability, though, is a major shortcoming for wireless communication compared with wired communication. Environment noise, reflection, path fading, mobility, etc., are all inherent problem for wireless communication. The WirelessHART standard employs a suite of latest technologies to fight this problem.

Many terms are used to describe certain aspect of reliability, such as availability, survivability, dependability, integrity, safety, performability, etc.

The WirelessHART standard achieves high reliability through diversity:

Time

All non-broadcast messages require acknowledgement within the timeslot. Retry is employed on failure.

Space (path)

The WirelessHART network is of mesh topology, each end-to-end pair must have at least 2 live paths.

Frequency

The WirelessHART standard provides channel hopping over 16 channels.

Code and Polarity

The WirelessHART standard adopts the IEEE 802.15.4 standard, whose coding and modulation methods are reasonably reliable. We expect further enhancement in the future.

10.10 Do I Need to Know All These to Use WirelsssHART Technology?

No, just like the answer to the similar question "Do I need to know all these mechanics to drive a car?"

Many users have already reported how easy it is. They received the shipment, unpacked, installed, and turned the power on. The whole WirelessHART mesh then came alive.

The WirelessHART standard puts a high premium on easy to use. For example, as we have discussed in Section 10.4, it does not require the highly technical field survey.

PART III WirelessHART in Practice

In this part we look at the use of the WirelesHART standard.

Chapter 11 describes the test and diagnostic tools from the HART Communication Foundation, the Wi-Analys tool and the Wi-HTest tool. The authors themselves led the development of the Wi-HTest tool, which will be described in detail here.

Chapter 12 suggests ways to quickly bring a traditional wired HART device into the wireless world.

Chapter 13 provides some comments on WirelessHART product development based on the authors' own development experience.

Chapter 14 suggests how to best make use of WirelessHART devices and network.

Chapter 11 Test and Diagnostic Tools

Abstract WirelessHART conformance testing requires test specifications and test tools. The test tools include the Wi-HTest tool, the Wi-Analys tool, and a post process suite for analyzing the packets captured by the Wi-Analys tool. The post processing utilities also summarize test results and generate the final compliance report. The Wi-Analys tool is a passive packet sniffer that captures all WirelessHART messages for analysis; The Wi-HTest tool provides a combination of network management, gateway, access point, script execution, and communications. The Wi-HTest tool joins the device under test then supports test script execution by allowing the test to introduce perturbations. The Network communications are captured by the Wi-Analys tool and sent to the post-processor for evaluation. Combined the number of wired and wireless tests totals over 200. These tests cover hundreds of different testing scenarios and thousands of different failure points. Each test script entails an execution of the Wi-HTest tool. To better manage the overall test process, special messages intended for the Wi-Analys tool are emitted as part of the Wi-HTest scripts.

To assure the standard compliance of HART products (including WirelessHART devices), which is key to vendors' success in the market, and help them avoid expensive product rework in the field and technical support costs, since 1995 the HCF has operated a rigorous quality assurance program and all HART devices must be thoroughly tested and registered with the HCF. As part of this program, HCF develops detailed test specifications and test tools for HART standards. Along with the release of WirelessHART standard, HCF has also released the Wi-HTest tool, an extension of the original HTest tool to support WirelessHART quality assurance especially focusing on the timing compliance test. HCF has also released a specific sniffer called Wi-Analys tool for real-time monitoring of the WirelessHART network and will release a post process suite for analyzing the packets captured by the Wi-Analys tool and generating the final compliance report. All these three tools together provide a complete compliance verification environment for WirelessHART devices.

The Wi-Analys and Wi-HTest tools could also be used for other purpose, from vendors' development to customers' deployment. The Wi-Analys and Wi-HTest tools each by itself could be used as a standalone tool.

11.1 The Wi-Analys Tool

The Wi-Analys tool, the Compliance Verification Receiver, is a WirelessHART message sniffer. Its product name is HCF_KIT-190™. It allows device companies to fully test and analyze their WirelessHART radio transmissions. It listens simultaneously on all 16 channels and captures all IEEE 802.15.4 conformant messages. If the message is a legitimate WirelessHART message, the fields of each stack layer, from the physical layer all the way up to the application layer, could be extracted and displayed in columns.

The Wi-Analys tool consists of a radio receiver box and a software suite running on a workstation.

Fig. 11.1 is the radio receiver box. It has only one physical antenna that is used for all the channels. Each received message will be time-stamped upon reception with the internal clock. The clock runs on a crystal of 10ppm. It is connected via the USB cable to a Windows workstation.

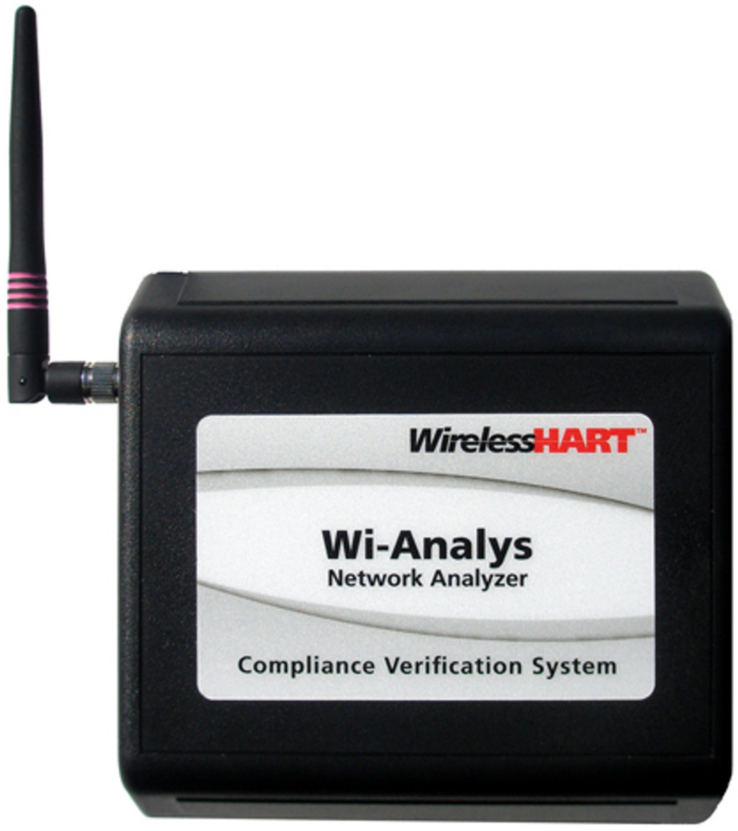
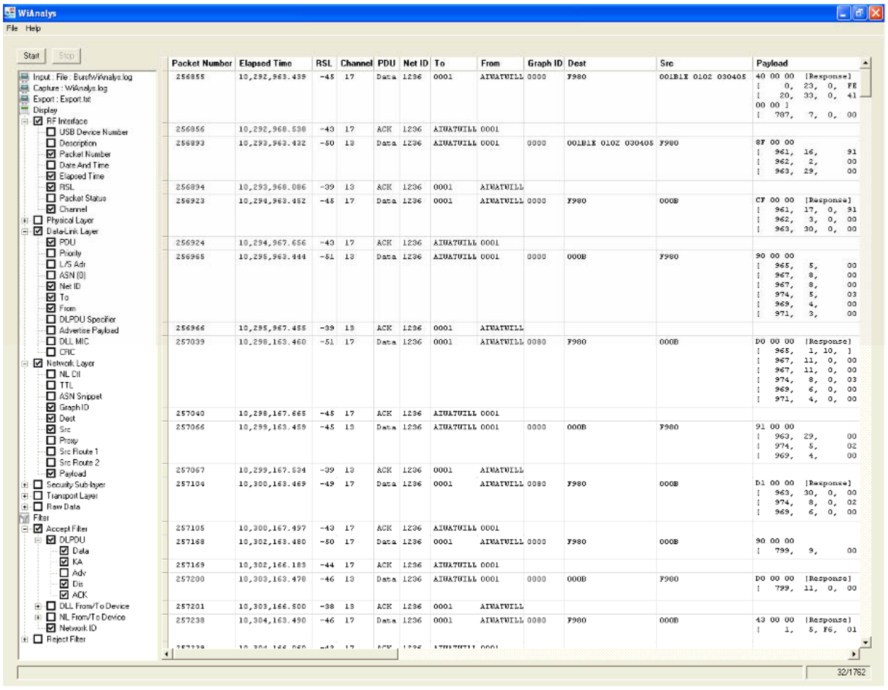


Fig. 11.1 Wi-Analys.

The software on the PC is a window displaying the messages. The Wi-Analys tool could capture live messages or replay saved logs. For live messages, its maximum capturing speed is up to 1000 messages per second. Messages on all physical channels are captured simultaneously. It could also display and export selected messages with selected data columns. The Wi-Analys tool also shows some statistics of the collected messages.

There are many ways to filter out messages not to display. A user could select which kind of messages to display or which not to display. The most useful filters for us are the network ID filter and the advertisement filter. Many times we have more than one WirelessHART network in our development environment. By filtering out messages not belonging to ours, we are able to concentrate on the work we do. When we are developing, our Wi-Analys tool is inundated with advertisement messages. By filtering out those messages, we could concentrate on the real communications on display.



- Input : File : Burst\WiAnalys.log
- Capture : WiAnalys.log
- Export : Export.txt
- Display
  - [✓] RF Interface
    - [ ] USB Device Number
    - [ ] Description
    - [✓] Packet Number
    - [ ] Date And Time
    - [✓] Elapsed Time
    - [✓] RSL
    - [ ] Packet Status
    - [✓] Channel
  - [ ] Physical Layer
  - [✓] Data-Link Layer
    - [✓] PDU
    - [ ] Priority
    - [ ] L/S Adr
    - [ ] ASN (0)
    - [✓] Net ID
    - [✓] To
    - [✓] From
    - [ ] DLPDU Specifier
    - [ ] Advertise Payload
    - [ ] DLL MIC
    - [ ] CRC
  - [✓] Network Layer
    - [ ] NL Ctl
    - [ ] TTL
    - [ ] ASN Snippet
    - [✓] Graph ID
    - [✓] Dest
    - [✓] Src
    - [ ] Proxy
    - [ ] Src Route 1
    - [ ] Src Route 2
    - [✓] Payload
  - [ ] Security Sub-layer
  - [ ] Transport Layer
  - [ ] Raw Data
- Filter
  - [✓] Accept Filter
    - [✓] DLPDU
      - [✓] Data
      - [✓] KA
      - [ ] Adv
      - [✓] Dis
      - [✓] ACK
    - [ ] DLL From/To Device
    - [ ] NL From/To Device
    - [✓] Network ID
  - [ ] Reject Filter

Packet Number	Elapsed Time	RSL	Channel	PDU	Net ID	To	From	Graph ID	Dest	Src	Payload
256855	10,292,963.439	-45	17	Data	1236	0001	AIWATWILL	0000	F980	001B1E 0102 030405	40 00 00 [Response] [0, 23, 0, FE [20, 33, 0, 41 00 00] [787, 7, 0, 00
256856	10,292,968.538	-43	17	ACK	1236	AIUATUILL	0001				
256893	10,293,963.432	-50	13	Data	1236	AIUATUILL	0001	0000	001B1E 0102 030405	F980	8F 00 00 [961, 16, 91 [962, 2, 00 [963, 29, 00
256894	10,293,968.086	-39	13	ACK	1236	0001	AIWATWILL				
256923	10,294,963.462	-45	17	Data	1236	0001	AIWATWILL	0000	F980	000B	CF 00 00 [Response] [961, 17, 0, 91 [962, 3, 0, 00 [963, 30, 0, 00
256924	10,294,967.656	-43	17	ACK	1236	AIUATUILL	0001				
256965	10,295,963.444	-51	13	Data	1236	AIUATUILL	0001	0000	000B	F980	90 00 00 [965, 5, 00 [967, 8, 00 [967, 8, 00 [974, 5, 03 [969, 4, 00 [971, 3, 00
256966	10,295,967.455	-39	13	ACK	1236	0001	AIWATWILL				
257039	10,298,163.460	-51	17	Data	1236	0001	AIWATWILL	0080	F980	000B	D0 00 00 [Response] [965, 1, 10,] [967, 11, 0, 00 [967, 11, 0, 00 [974, 8, 0, 03 [969, 6, 0, 00 [971, 4, 0, 00
257040	10,298,167.665	-45	17	ACK	1236	AIUATUILL	0001				
257066	10,299,163.459	-45	13	Data	1236	AIUATUILL	0001	0000	000B	F980	91 00 00 [963, 29, 00 [974, 5, 02 [969, 4, 00
257067	10,299,167.534	-39	13	ACK	1236	0001	AIWATWILL				
257104	10,300,163.469	-49	17	Data	1236	0001	AIWATWILL	0080	F980	000B	D1 00 00 [Response] [963, 30, 0, 00 [974, 8, 0, 02 [969, 6, 0, 00
257105	10,300,167.497	-43	17	ACK	1236	AIUATUILL	0001				
257168	10,302,163.480	-50	17	Data	1236	0001	AIWATWILL	0000	F980	000B	90 00 00 [799, 9, 00
257169	10,302,166.183	-44	17	ACK	1236	AIUATUILL	0001				
257200	10,303,163.478	-46	13	Data	1236	AIUATUILL	0001	0000	000B	F980	D0 00 00 [Response] [799, 11, 0, 00
257201	10,303,166.500	-38	13	ACK	1236	0001	AIWATWILL				
257238	10,304,163.490	-46	17	Data	1236	0001	AIWATWILL	0080	F980	000B	43 00 00 [Response] [1, 5, F6, 01

32/1762

Fig. 11.2 Wi-Analys Screen Capture.

The Wi-Analys tool accepts security keys through user input and intelligently decodes the messages from the Wi-HTest tool. These messages could be the ones intended to the Wi-Analys tool, or legitimate WirelessHART commands from the

Wi-HTest tool to the device that assign keys to the device. The Wi-Analys tool will use the keys it possesses to authenticate or decrypt. Messages that fail will be indicated with different colors.

Fig. 11.2 is a screen capture of the Wi-Analys software display window.

Table 11.1 lists the data fields defined in the Wi-Analys tool.

Table 11.1 Wi-Analys Display Fields.

Protocol Layer	Name	Explanation	Example: Join Reply Message
RF Interface	USB Device Number	The radio receiver box serial number	115077
-	Description	IEEE 802.15.4 message type	802.15.4-DATA
-	Packet Number	-	680
-	Date And Time	Raw timestamp	2009-09-20 03:01:57.031
-	Elapsed Time	Precise timestamp	348,453.681
-	RSL	Received Signal Level	-50
-	Packet Status	-	0x0000
-	Channel	Physical channel	11
Physical Layer	Byte Count	-	110
Data Link Layer	PDU	WirelessHART message type	Data
-	Priority	WirelessHART message priority	Cmd
-	L/S Adr	-	88
-	ASN (0)	Lowest ASN byte (sequence number)	B0
-	Net ID	-	1236
-	To	Data link layer destination address	0005
-	From	Data link layer source address	0001
-	DLPDU Specifier	-	37
-	Advertise Payload	If it is an advertisement message	-
-	DLL MIC	Data link layer MIC	CB2A0763
-	CRC	-	FBEB
Network Layer	NL Ctl	Control Byte	85
-	TTL	-	7E
-	ASN Snippet	-	684C
-	Graph ID	-	0000

-	Dest	Network layer destination address	001B1E2659000000
-	Src	Network layer source address	F980
-	Proxy	Proxy address if proxy routing	0001
-	Src Route 1	Source routing table 1	000100050004FFFF
-	Src Route 2	Source routing table 2	-
Application Layer	Payload	Network payload	8F 00 00 *[961, 16, 0D 32 23 40 7E 44 48 5D 65 D9 B8 DB 0B 92 C1 B9]*[962, 2, 00 04]*[963, 29, 00 F9 80 F9 80 00 00 01 00 00 00 00 9B 9D 57 11 C4 5C D4 CA 2A 10 07 DE 5C D2 28 01 00]
Security Layer	SL Ctl	Security control byte	Join
-	SL Cnt	Nonce counter	0000087C
-	SL MIC	Network link layer MIC	37DE8837
Transport Layer	TL Ctl	Transport layer control byte	8F
-	TL Status	Device status	00
-	TL ExStatus	Extended device status	00
Raw Data	Cipher Text	Original message byte stream. First byte is the stream length	6E4188….F6C4….CB2A0763FBEB
-	Clear Text	Message byte stream with network layer payload deciphered.	6E4188….F980….CB2A0763FBEB

11.2 The Wi-HTest Tool

The Wi-HTest tool (Han et al. 2009), the WirelessHART Test System, is designed to work with the HART Test System HTest tool and the Wi-Analys tool to verify conformance of WirelessHART devices. Its product name is HCF_KIT-193™. According to HCF:

> The Wireless Test System is critical to the development and compliance testing of WirelessHART devices, Manufacturers must use this system to test a WirelessHART device prior to submitting the device for Foundation registration.

HCF tests the compliance of wired HART products using the HTest tool which is the general purpose HART master software for PC with HART modem. The HTest tool uses CINT (root.cern.ch/twiki/bin/view/ROOT/CINT), a simple interpreted script language, to build, send, receive and display HART messages. By evaluating the correctness of the response data, the compliance of the Device Under Test (DUT) can be verified. However, along with the release of WirelessHART standard and rapid emergence of WirelessHART devices in the market, it's necessary for the HCF to extend the HTest tool to support wireless commands testing and assure the compliance of WirelessHART devices.

As a test suite specifically designed for wireless real-time communication protocol, the Wi-HTest tool has two inherent functionalities. First, it is able to construct the test packets in real-time manner and send them to the DUT through IEEE 802.15.4 radio with accurate timing. Second, the Wi-HTest tool can capture the response packets from the DUT along with its timing information. These two functionalities enable the Wi-HTest tool to not only check the correctness of the response data but also verify the timing of the response.

The eWi-HTest tool is aimed to automate the execution of test cases defined in the WirelessHART test specification. More specifically, the Wi-HTest tool provides the stimulus (good and bad) necessary to exercise operations of the DUT. The test cases for verifying the compliance of WirelessHART devices can be roughly classified into two different test scenarios, the device joining process and normal data communication. In the joining process, the Wi-HTest tool coordinates with the DUT through a sequence of message exchanges and verifies whether the DUT can join the WirelessHART network successfully; While during the normal data communication, by executing specific test scripts, the Wi-HTest tool transmits either correct data packets or manipulated packets by injecting fault data. By evaluating DUT's corresponding behaviors in the presence of good and ill-behaved stimuli, the DUT's compliance can be assessed.

11.2.1 WirelessHART Test Specification and Test Scripts

In developing a WirelessHART compatible field device, a variety of informal ad-hoc testing and formal tests are to be performed. HCF simplifies this testing effort by supplying a test specification to developers. This test specification must be completed along with the test report prior to product release and product registration with HCF. The test specification provides clear test requirements and reduces the number of test plans that must be developed by the manufacturer. They can be used early in the development effort to informally verify functionality during implementation and are a useful part of a regression testing program as the field device is maintained and enhanced. Furthermore, the test specification clarifies ambiguities in the protocol and is the final, definitive authority when interpreting the protocol.

This test specification uses a quasi *black box* approach to confirming compliance with WirelessHART standard. The complete set of tests consists of the following five phases and each phase contains multiple test descriptions that are intended for sequential execution.

- Boot-Strap Tests
- Single Correspondent Tests
- Multiple Correspondent Tests
- Multi-channel-selection Tests
- Stress Tests

The Boot-Strap Tests try to perform an audit of the command set implemented in the DUT through either maintenance port or wireless connection. They also configure the DUT to put it in an initialized state and test its joining process into a specific WirelessHART network.

Based on the successful completion of all Boot-Strap tests, the single correspondent tests focus on the scenario where the DUT interacts directly with a Network Manager and Gateway. This series of tests demonstrate that a wireless Field Device properly requests admission to the wireless network; accepts commands that condition its operation in the wireless network, including commands with a deferred execution; and operates synchronously with the peer device. Different from the single correspondent tests, Multiple Correspondent Tests demonstrate that a wireless device properly interacts with multiple peers, including inferring information about those peers from received messages.

Multi-channel-selection Tests extend the Multiple Correspondent Tests by demonstrating that a wireless device properly selects among multiple potential radio channels on which its schedule permits operation.

Finally, the Stress Tests combine all the prior tests into a single random sequence that serves to demonstrate continuous device operation in a simulated field environment. The primary purpose of this phase is to provide confidence that the device will reliably interoperate in a real-world environment.

According to the WirelessHART test specification, we write the test scripts for each test case within different test phases. Test scripts are small, narrowly-focused test applications. They are taken as input to feed in the Wi-HTest tool for establishing the test environments, generating proper test packets and conducting the compliance tests. Typically a test script composes of two parts: The test configuration and the test body. The test configuration section initializes the Network Manager and Gateway and sets up the equipment and various test parameters. The test body consists of a sequence of small test steps. Each test step generates or manipulates a WirelessHART data packet by calling related libraries implemented in the Wi-HTest tool.

11.2.2 Wi-HTest Architecture

11.2.2.1 Overview

The WirelessHART standard is an industrial strength real-time protocol and we have a procedure/plan to verify its data correctness and timing compliance with the Wi-HTest test suite. The Wi-HTest tool extends the HTest tool, the traditional compliance test tool for HART devices by supporting wireless commands to stress WirelessHART-enabled devices.

The Wi-HTest tool consists of two components: the Wi-HTest Host and RF Interface. Fig. 11.3 is the picture of the tool. The box on the left is a PC box running Ubuntu Linux. The box on the right is the RF interface. Fig. 11.4 presents the high level architecture of the Wi-HTest test suite.

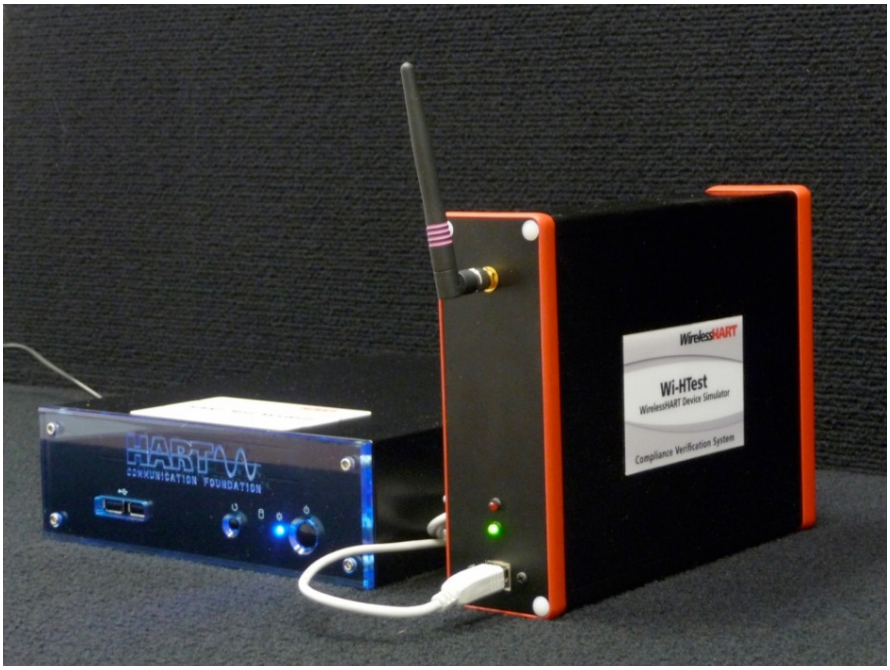


Fig. 11.3 Wi-HTest Tool.

The Wi-HTest Host is responsible for overall control and execution of the input test scripts. According to the specific requirements of the scripts, it generates either correct packets or manipulates the packets by injecting fault data into either the command payload or the headers of different layers. Wi-HTest Host then sends these packets to the RF Interface for transmission through the IEEE 802.15.4 radio. The RF Interface is a real-time embedded pseudo-stack. It is responsible for low-level, time-critical communications to and from the DUT using its onboard wireless transceiver. Responses from DUT are forwarded back to the Wi-HTest Host and captured by the Wi-Analys tool for post processing. Finally, the post process suite reads in the log recorded by the Wi-Analys tool (especially the timing information of the response packets) and generates the corresponding compliance report for the DUT. The details of the system design of the Wi-HTest tool are discussed in the following sections.

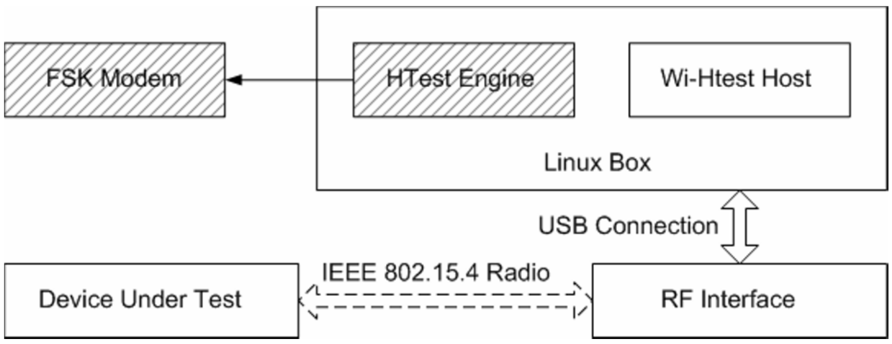


Fig. 11.4 Wi-HTest High Level Architecture.

11.2.2.2 Host Architecture

The Wi-HTest Host is implemented on a Linux box and consists of three major modules: the RF Interface driver, network layer library, and the Wi-HTest engine. The architecture of the Wi-HTest Host is depicted in Fig. 11.5. These modules coordinate closely to achieve the following functionalities:

1. Read in the test scripts as input and set up corresponding equipments and test environments.
2. According to the requirements of the test script, generate corresponding command payload and assemble it with proper network layer header. If necessary, inject designated errors into the network payload and further inform the data link layer in which fields its header should be manipulated.
3. Transmit the control information and data packets to the MAC layer for transmission and wait for the response from the DUT.

We describe the details of each module as follows.

RF Interface driver:

The driver between the Wi-HTest Host and the RF Interface uses a simple private protocol for communication through USB (see Fig. 11.4). The protocol provides basic framing functions, such as preambles, delimiter, frame control and CRC error detection.

There are three types of commands carried over the USB cable: commands from the host Linux box to configure the RF Interface (Type I), commands to relay data packets between the Wi-HTest Host and the DUT (Type II), and commands from the RF Interface to the Linux box to update certain data structure in the Host (Type III). Basically, Type I commands are mostly WirelessHART commands. An example is Command 965 which is used to write superframes to the RF Interface. There is only one Type II command defined, Command 64513, to denote the standard network to data link layer data request or data link layer to network layer data indication. Currently there is only one Type III command, command 64518, used by the RF Interface to update its ASN (Absolute Slot Number) to the Linux box. This provides Wi-HTest Host a rough ASN snippet to fill in the network layer header. More Type III commands are introduced to provide the data sharing between the Host and the RF Interface.

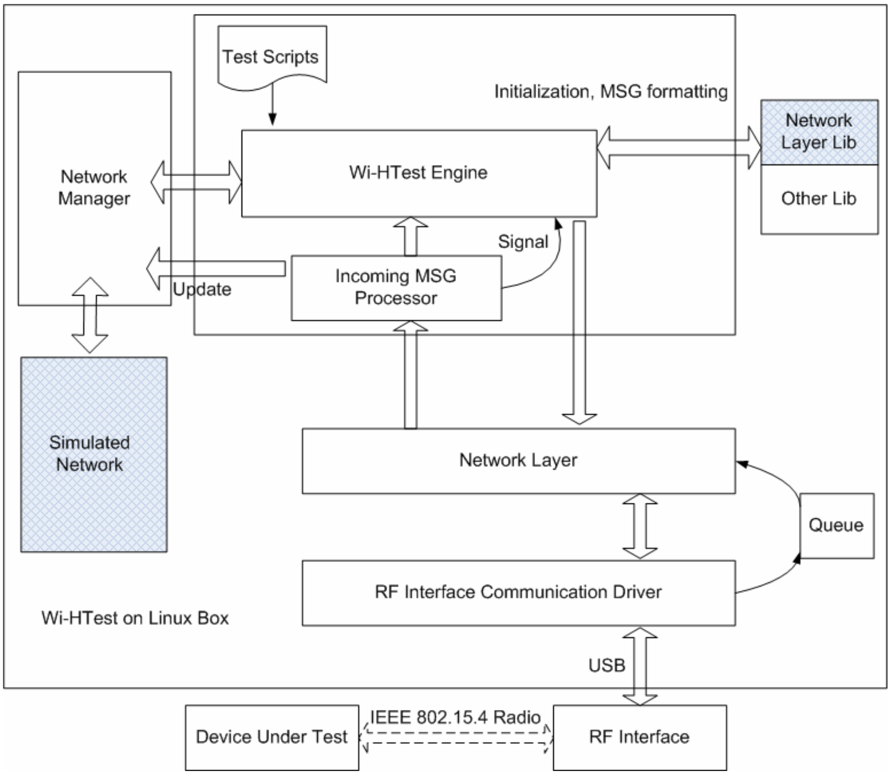


Fig. 11.5 Wi-HTest Host Architecture.

The frame counter field guarantees the reliable communication between the Wi-HTest Host and RF Interface. Each end of the communication keeps two frame counters, one for itself and one for the other end. Each time a device sends a frame, its own frame counter is incremented by one. When it receives a frame, the counter in the incoming frame is compared to the frame counter for the sender. If they are equal, the frame is expected. Otherwise, the frame is discarded silently. Whenever a correct frame is received, the receiver increments the frame counter for the sender by one.

The Network Layer on Wi-HTest Host:

The network layer is constructed as a library along with an independent receiving thread in Wi-HTest Host. The network layer library provides function calls to construct or manipulate the data payload and packet header while the receiving thread handles the response packet back from the MAC layer or processes received unsolicited messages.

We separate the network layer from the RF interface and put it on the Wi-HTest Host for three practical reasons. First, compared with the MAC layer, most of the operations on network layer are not time critical. Given the limited memory and MCU resources, and stringent timing requirements on the RF Interface, putting the network layer on the Wi-HTest Host can save more resources for implementing Wi-HTest-specific modules on the RF Interface. Second, with the network layer on the Wi-HTest Host, it is more direct and convenient for the test scripts to inject fault data into the WirelessHART command payload, the network layer header, and even the MAC layer header. On the Wi-HTest Host, these operations can now be achieved by calling corresponding functions directly. Otherwise, the test script has to convey these control information to the RF through various interface messages; At last, putting the network layer on the Host provides us the possibility and flexibility to simulate virtual devices and form a virtual network for multiple correspondent tests.

The network layer library provides plenty of useful function calls for supporting various network operations. For example, based on given parameters, a set of functions are used to construct network header while another function set is for parsing existing network layer packets. The library also contains functions for network layer initialization and configuration, building interface messages, encrypting, decrypting and authenticating network layer packets, and maintaining various communication tables in the network layer.

In each test case, when a transmit command is read in from the test script, the test engine in Wi-HTest Host assembles either correct or manipulated network layer packets by calling related functions from the network layer library and forwards it to the RF driver for transmission; the independent receiving thread continuously monitors the incoming queue and handles different types of packets by calling corresponding callback functions in the incoming message processor. If the message is a response that the test engine is waiting for, it is stored in a shared buffer and the test engine is woken up for processing. For other cases, the message will be either discarded or used to update relevant data structures. An example of such messages is the neighbor health report which is generated periodically by each device to update its status to its neighbors.

To provide the tester complete control over the transmitted packet, a critical feature provided by the network layer library is the bitwise packet manipulation. It allows the test cases to change any field of the network packets including the header and the command payload. Furthermore, the interface between the network layer and the MAC layer enables us to specify which fields of the MAC layer header will be manipulated and how. In our implementation, all fields in the MAC layer header has a corresponding bit in the bitmap variable bitMapHdrs-Manipulated. For example, bit 0 corresponds to byte 0 of the MAC header. A set bit in the variable bitMapHdrsManipulated denotes that the corresponding byte in the header should be manipulated using the information in the interface message from the network layer.

To support multiple correspondent tests which focus on evaluating network layer operations, another important module named simulated network is introduced in Wi-HTest. Cooperating with the network manager, this module can simulate a virtual network for a single DUT by carefully configuring each virtual device. In this way, the DUT is tricked to believe that it is operating in a real WirelessHART network with multiple devices and then various network layer tests can be performed on the device.

Wi-HTest Test Engine:

The heart of the Wi-HTest Host is a Wi-HTest engine which is a C++ program executed in the CINT environment. CINT is a C/C++ interpreter aimed at processing C/C++ scripts with reduced compile and link cycle. An important component of the test engine is the script library. This library provides a bunch of supporting functions to help generate various test scripts.

The main functionality of the test engine is to read in the test scripts and generate corresponding wireless commands for the DUT. It then passes the commands through the network layer, assembles it with required header and sends it to the RF Interface. The test engine then waits for the response from the DUT and verifies its correctness. On the other hand, the test engine will maintain timers for various timeout defined in the test script. If a timeout is reached while no expected response is received from the DUT, the test engine will report corresponding error messages.

11.2.2.3 RF Interface Design

For the Wi-HTest Host to be able to talk to the DUT following the WirelessHART protocol, we connect a RF Interface through a USB cable to the Host. The RF Interface works as a bridge between the Wi-HTest Host and the DUT. Fig. 11.6 illustrates the overall architecture of the RF Interface.

Hardware Platform:

We implement the RF Interface using a Coldfire V1 toolkit from FreeScale. This hardware has the following features:

- Up to 50.33 MHz ColdFire V1 core.
- Up to 128 KB of flash memory and up to 16K RAM with security circuit.
- Supports four low-power modes.
- On-board Logic Analyzer and Virtual Serial Port.
- USB device mode and host mode support with Mini-AB USB connector.
- 8 User LEDs and 5 Push Buttons.

The hardware platform is powerful enough to meet the stringent timing requirements defined in the WirelessHART specification.

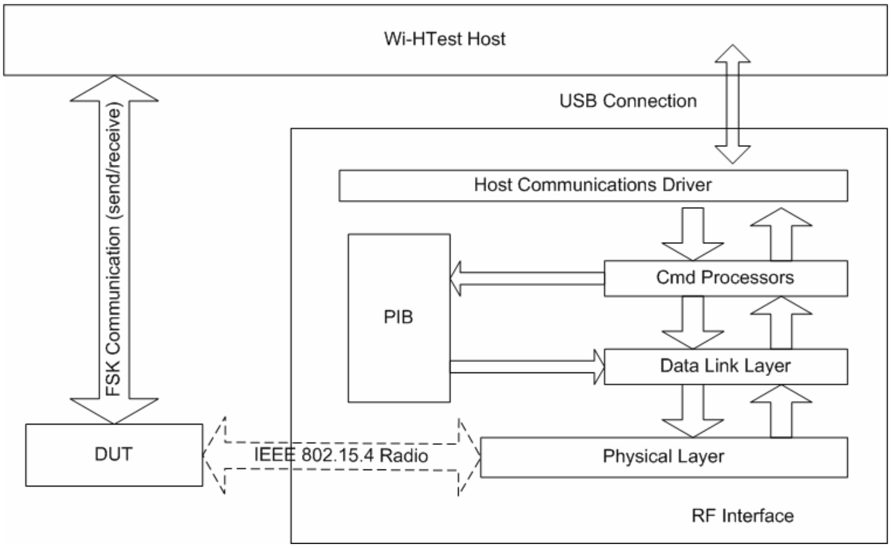


Fig. 11.6 Wi-HTest Architecture on RF Interface.

Real-time Embedded Pseudo Stack:

The data link layer and physical layer on the RF Interface are collectively called the Pseudo Stack as the network layer is separated and put on the Wi-HTest Host. The pseudo stack is fully compliant with the WirelessHART specification, which means it must meet the stringent timing requirements defined in the specification. To address this strict time synchronization issue, we use several solutions.

First of all, enciphering a frame and deciphering its acknowledgement (using AES-128) can take the stack out of synchronization. In order to speed up the enciphering/deciphering process, we take a streaming strategy. For example, if it is in a receiving slot, the stack starts to decipher the incoming frame when the first 16 bytes are received. In this way, the calculation intensive security checking can always be finished in time. Secondly, the interrupt handler is kept as simple as possible. Only those time critical jobs are put in the handler. Non-time-critical jobs are deferred and would be processed at appropriate time. Thirdly, we give the MAC layer the highest scheduling priority. Every time there is a frame to be processed, the MAC layer can preempt all the other tasks and get the MCU. There could be several jobs waiting at any time in the MAC layer. We further prioritize these jobs to keep the stack in synchronization. For instance, transmitting a frame has priority over accepting a frame from the command processor.

The Command Processor:

As the RF interface is basically a wireless access point for the Wi-HTest Host, all test commands are transmitted via it and all test responses are captured by the RF Interface. All commands and responses are handled in the command processor, the core of the RF Interface.

There can be three types of messages on the USB cable:

- Board configuration commands: The commands issued by the Host to configure the RF Interface.
- Test commands: The commands issued by the Host to be transmitted by the RF Interface to the DUT.
- Test responses: The responses from the DUT to the Host.

We use the simple communication protocol to differentiate the messages and provide error detection and retries. During initialization, the RF Interface waits for configuration commands from the Host. After it is set up properly, the RF Interface can accept test commands and receive responses from the DUT. If the commands are test commands, it encapsulates the command in a WirelessHART MAC frame and passes it to the data link layer, which would transmit the packet over the air to the DUT. In the other direction, every time the RF interface receives a response from the DUT (through the PHY and MAC layers), the RF Interface sends the response through the USB connection to the Wi-HTest Host.

Although the Wi-HTest tool is just a test instigator, DUT behaves as if it is joining and controlled by a real WirelessHART network. In this sense the network manager, the gateway, and the access point are tightly combined together in Wi-HTest as a single unit. Wi-HTest is even more than that; it also supports various testing functionalities such as fault injections.

To pass FCC test, the Wi-HTest tool could also generate continuous wave. Continuous wave is basically a test mode using transmit without modulation. This could also be used to generate interference on a physical channel. The script sends Command 64529 to the RF interface to enable or disable this.

11.3 The Post Process Suite

The Wi-Analys tool captures and logs all the wireless traffic. The logs are post-processed by the post process suite to check if all the messages conform to the standard. The benefit of the log files is two folded. They serve as the raw data for ultimate compliance check; they also serve as the defense evidence when Wi-HTest is blamed for DUT's failure to pass the test.

The post process suite judges the successfulness of the compliance test. For each test case, a post process program reads the log file and analyzes it. Depending on the purpose of each test case, it will check the sequence of the messages the DUT transmitted, the transmission time points, the relationship of the messages, the content of the messages, etc. If all satisfy the standard, the test case is passed. Otherwise the place where the standard is violated will be reported.

The log file produced by the W-Analys tool is in plain text. So people could manually check the correctness of a DUT's interaction with the Wi-HTest tool. A device could be certified even before the post process suite is ready.

The Wi-HTest tool, the Wi-Analys tool, and the post process suite comprise the complete compliance verification environment. In this environment, a device typically goes through the following steps to be certified by HCF. A complete test is composed of a set of test cases. The Wi-HTest tool runs each test case with the Wi-Analys tool capturing the messages through the whole test case period. While the Wi-HTest tool could declare if a device has passed certain test cases, a post process program per case will analyze the corresponding log file to check if the device has strictly followed the standard during the test, especially satisfied the stringent timing requirements. The device is certified if it passes all the test cases.

Chapter 12 A Fast Approach to Equip a HART Device with WirelessHART Capability

Abstract The WirelessHART standard extends the traditional HART standard with wireless capabilities. It is important for existing HART devices to support WirelessHART. The WirelessHART adapter is specially defined to connect wired HART device to host applications via the WirelessHART network. In this chapter we follow the principle of a WirelessHART adapter and suggest a fast solution to equip HART devices with wireless capability without much change to the existing HART devices. This could serve as an intermediate solution before a fully integrated WirelessHART device is developed. In many cases this might also be the final solution for the HART device vendors.

12.1 The WirelessHART Adapter

The WirelessHART standard specifies a device type called an adapter. The adapter is not just a wire replacement but an intelligent HART-enabled device and powerful system integration tool. The adapter joins a WirelessHART network as an independent device; it also connects to traditional wired HART network as a master device. All devices on the wired HART network could communicate through the WirelessHART network via the adapter. They could be individually addressed by the host applications as sub-devices of the adapter. The adapter provides a cost efficient connection for integrating the intelligent capabilities of HART devices into control and asset management systems.

12.2 A WirelessHART Adapterlite

We propose adapter like component that is attached to any traditional HART device. Let's call it adapterlite. The adapterlite is piggybacked to a HART device; together they behave as a WirelessHART device. There is no change to the HART device; it is connected to the adapterlite via the HART cable. The adapterlite serves as its master on this internal two-device wired HART network. To the outside the adapterlite presents itself as the HART device in the wired or wireless HART network.

It is clear at this point that an adapterlite is composed of the following parts:

- An FSK port connectable to the HART device. This serves to control the HART device as the master.
- A WirelessHART stack and the antenna. This serves to present the new formed WirelessHART device in the wireless network.
- Another FSK port exposed to the outside. This serves to expose the wired HART interface of the newly formed WirelessHART device.

This is how the adapterlite behaves:

Upon startup, the adapterlite will communicate with the HART device and retrieve necessary device information such as device ID, tag, etc. And then the adapterlite will take on the identity of the device.

The adapterlite then waits to be initiated either through the external FSK port or by joining a wireless network.

The adapterlite will handle all wireless and wired HART commands. When handling a command, the adapterlite may, depending on necessity, send corresponding command to the HART device for the response or use different commands to retrieve any information. Commands initiated from the wired device will also be forwarded out as if it is initiated from the adapterlite.

If the wired device supports HART version 7, then the adapterlite is the simplest. It could forward most of the commands in both directions without much modification. If the wired device is of an earlier version, then the adapterlite will take up the new functionalities defined in version 7.

We could develop an adapterlite from scratch. Or we could develop one directly from a standard WirelessHART adapter. There is no change to the hardware, simple modify the software in an adapter in a few places:

- Only recognize one slave device on the network.
- Take over the identity of the slave device instead of recording it as a sub-device.
- Rather than acting as the message router between the host and the sub devices respond to the host directly, treat the device as internal component that serves as the data source.

A WirelessHART adapter is capable of handling devices with earlier HART versions. Hence the adapterlite derived from an adapter automatically handles older version devices. Two important features of the adapterlite approach:

- There is no change to the HART device, by attaching the adapterlite to it, we make it a WirelessHART device.
- The Adapterlite is universal in that it could be attached to any HART device.

Once there is a need to develop the fully integrated WirelessHART device from the wired HART device, the adapterlite could be part of the solution by removing the internal FSK connection and combining the circuit board of the device and the adapterlite into one board and keep most of the software, e.g., the stack, in the adapterlite.

Chapter 13 Development Recommendations

Abstract Embedded development is much harder than PC based application development. This chapter collects some of our experiences in the process of forming and experimenting with the WirelessHART standard. The hands-on experiences are drawn from our development of the Wi-HTest tool, which essentially supports all the functionalities required of a WirelessHART device. We shall discuss the need for a real-time OS, how to timestamp the received message, the API between stack layers, a timer module to support timeslots, hardware considerations, and miscellaneous comments.

13.1 OS or No OS

A PC needs an operating system (OS) before any application can run. The OS provides the execution environment so that an application only needs to carry out the function it is supposed to perform such as editing an article, drawing pictures, etc. A general purpose OS has big memory foot print because it should support all those needed in the environment. As a result, it also slows down the performances of the applications.

In an embedded system normally there is only one application running for one purpose. The OS serves this only application. To some extent the OS and the application together is one single software piece. Since an application will not need all the services provided by an OS and the embedded platform has limited memory space and speed, most applications include a subset of the OS. In addition, timing requirements in the embedded world forces the OS to provide extra real-time support. Any embedded OS is a real-time OS (RTOS). It is reasonable to develop a WirelessHART stack with some kind of RTOS.

An operating system is a human invention to facilitate application development. It is quite possible for an application not to make use of any service provided by an OS, especially the embedded applications. The ZigBee toolkit provided by FreeScale is such an example. The projects generated from its BeeKit™ tool does not link with any OS library. After power up initialization, the application enters an endless loop; each loop selects a task and runs it to the completion. There is merit to implement WirelessHART stack without an OS. Firstly it saves memory and execution time. Secondly it only requires limited OS support for a WirelessHART solution. To go without any OS, people develop a WirelessHART solution must have enough computer science background to replace those required OS services.

13.2 Timestamping Incoming Messages

WirelessHART data link layer uses TDMA. To maintain good synchronization, it is critical that we have precise timestamps of message arrivals. There was some confusion while the standard was developed about when to timestamp a message. Do we timestamp before the preamble, after the preamble, after the delimiter, or after the physical header? This question is actually an implementation question. The standard only needs to pick one point and use it consistently through the specification. Any implementation could derive the timestamp at any point in the message from the timestamp of its interrupt handler; the message bytes arrive at the constant speed of 250k bytes per second. Of course, we have to know, for each hardware platform, what the interrupt is triggered by, the reception of the preamble, or the reception of the message length, or others. For example, the FreeScale MC1320 radio board could be configured to interrupt at many places of an incoming message, the reception of the message size byte in the physical header, the end of the message, and every reception of 2 message bytes. We could timestamp any one of the interrupts and calculate the arrival time of any byte in the message. To get the timestamp value, we simply read the clock in the beginning of the interrupt handler. MC1320 board itself also saves its tick counter value in an internal register upon the reception of a message. So another way to get the timestamp with MC1320 board is to read that register before the next message comes.

13.3 Realizing Network Layers in the Stack

Each layer in a stack could be implemented as a library or a task. In the former approach each stack layer is a collection of APIs (Application Program Interfaces). Upper layer calls the lower layer API which in turn calls the even lower layer API. There could be one task handling the transmission and another one handling incoming messages. This approach is very efficient for small size implementations. WirelessHART stack, however, is relatively more complex compared with other similar stacks such as ZigBee stack. It is at the higher end of the low power wireless mesh. We recommend implementing each network layer as a task. The communication between layers is realized by message passing between tasks. The lower layer has higher priority due to its closeness to time constrains. This approach matches closely with the interfaces defined between WirelssHART network layers.

13.4 API between Adjacent Network Layers in the Stack

The standard has defined the message APIs, i.e., service primitives (SP), between network layers. This is important to delineate the boundaries between the layers and what information is passed through. However, they are internal to a stack and other devices or host does not know about the exact implementation. So an implementation could define those APIs differently to meet particular needs. For example, people may want to pass extra statistic or limiting information through. In fact, the APIs defined in the standard could only be used as a reference. Some of the required detail information is not defined in the standard. For example:

- The WirelessHART standard does not define the PIB (PAN Information Base) attributes in explicit tables as the IEEE 802.15.4 standard does. Certain attributes could be implemented either in the data link layer or the network layer, especially those related to the join procedure. Some attributes must be accessed by both layers.
- Join procedure itself involves close cooperation between the network layer and the data link layer. As a matter of fact, the data link layer's responsibility for join is described in Section 9.4.2 of the network layer specification.
- In the TRANSMIT.indicate primitive in the data link layer specification, the meaning of the localStatus parameter is defined but the bit assignment is not. Besides, some implementation may want to add the handle parameter.
- In the data link layer specification the primitive RECEIVE.indicate(localStatus, packetRSL, payloadDLPDU) could be used for tools like a Sniffer. To implement this, a lot of other issues must be considered: when do we set the radio in the listening mode? How do we not filter out messages with arbitrary destination address? How does the upper layer parse the data link layer package payloadDLPDU?
- The physical layer specification API does not say how the incoming messages' timestamp is obtained. If the only interface to the physical layer is the ones defined there, then the data link layer could only read the clock in the beginning of ENABLE.indicate() or DATA.indicate(rsl, data) implementation as the timestamp, which is not precise due to delays. A practical implementation needs to define the API as ENABLE.indicate(timestamp) or DATA.indicate(rsl, data, timestamp), or add a new management API to read the timestamp of the last received message.

13.5 A Timer Module

In a WirelessHART network all devices must strictly obey the 10 ms time slot rule. The time slots start from the creation of the mesh network. All devices are synchronized and prescheduled to behave in each time slot, idling, transmitting, or receiving. Within each transmission or receive slot, a device must act promptly at different time points in order to have successful communication and time synchronization. Because the distance between time points are short, the action executions are heavy, and the processor is less powerful due to energy consumption constraint, traditional way of setting the timer one at a time may not work. In this section we describe how we implement this in the Wi-HTest. We create a standard-conscious timer module that generates a series of time interrupts automatically according to the timeslot type (Song et al. 2008). In addition, we divide actions into synchronous and asynchronous parts. The synchronous part is executed within the interrupt handler.

13.5.1 Timeslots in the WirelessHART Standard

Please refer to Fig.3.4 for the timing sequence of a timeslot. The top one shows that of the sender; the bottom one shows that of the receiver. A device is prescheduled to send, received, or idle in each timeslot.

If a node is scheduled to send, it first waits for TxCCAOffset. Then it starts CCA followed by the transmission of the message. Afterwards it will start listening to the acknowledgement from the receiver TsRxAckDelay after the end of message transmission. The listening will timeout in time length of TsAckWait.

If a node is scheduled to receive, it waits for TsRxOffset, and then starts listening. If no message coming in, it stops listening after TsRxWait. If a message comes in, it transmits acknowledgement TsTxAckDelay after the end of the incoming message.

The timeslots of communicating devices must be synchronized. The timing sequence within a timeslot must also be synchronized so that two devices could successfully talk, taking into considerations of clock drifts.

13.5.2 Standard-Conscious Timer Module

Fig. 13.1 shows the components of this design. Instead of setting next timeouts based on current upper layer execution, the timer module knows about the standard's timing requirement and manages them with a small footprint to suit low power processors. Only heavy duty computation is left to the upper layer to run asynchronously.

Hardware Timer:

The hardware timer is supported in any hardware platforms. It interrupts after the specified number of ticks expires from the counter. To support the Wireless HART standard, it should have the resolution of microseconds.

WirelessHART Timer Module:

It is the core component of this design. It is built on top of the Hardware Timer. When a timeout interrupt occurs, it checks its state for the semantic of the timeout, executes some code, if necessary sends a timeout event to the upper MAC layer, then sets the next timeout.

If the timeout indicates a timeslot start, the slot type is checked first, if it is an idle slot, the timer module will set the next timeout to the next time slot 10ms later.

If the next time slot is a transmit slot. The timer module sends out an event to the upper layer and sets the next timeout to be TsCCAOffset later. Upon timeout it will issue a CCA and start transmitting a message. Then it sets the next timeout to be TxMaxPacket+TsRxAckDelay after the start of transmission for listening to reply. If no message is available, it will set the timeout to be the next timeslot. If there is message, the next timeout will be adjusted to TsRxAckDelay after the end of the message. When timeout for listening to reply, it will put the antenna into receiving mode and set the final timeout to be TsAckWait later, upon which it will set the next timeout to be the next timeslot. There will be error handlings if no acknowledge is received. If the acknowledge is received before the timeout, we could directly reset the timeout value to the next timeslot.

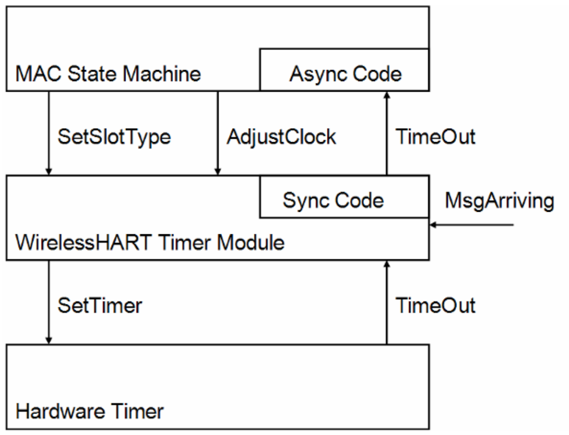


Fig. 13.1. WirelessHART Timer Module.

If the next time slot is a receive slot. The timer module sends out an event to the upper layer and sets the next timeout to be TsRxOffset later. Upon timeout it will start listening for transmission and set the next timeout to be TsRxWait later. If no message is coming upon timeout, it will set the timeout to be the next time-slot, otherwise it will set the next timeout to be TxMaxPacket+TsTxAckDelay after the start of the incoming message. This timeout value will be adjusted, once the incoming message size is known, to TsTxAckDelay after the end of the message. At time out the acknowledgement message is transmitted and the next timeout is set to be the next timeslot.

The Timer Module also receives interrupt when the message starts coming in. It will record the timestamp, which is used to synchronize devices.

MAC State Machine:

This is the MAC layer implementation that is responsible to conforming to the WirelessHART timeslot timing requirement. Because the majority of work is moved to the Timer Module, the MAC State Machine only need to set the next slot type, adjust the local clock, and execute asynchronous part of the timer code.

One of the jobs the MAC State Machine must do is to tell the Timer Module the type of future timeslots. Not doing this beforehand will result in a default idle timeslot. Also the timer module needs to be instructed to adjust its clock. The adjustment involves many factors including the incoming message timestamp. The timer module itself alone could not determine if and how much clock should be adjusted.

Between any two consecutive timeouts a node must perform some tasks. Some of them must be done immediately, some may be postponed. We divide them into synchronous and asynchronous parts. The synchronous part is within the timer module, it is executed within the timeout interrupt handler. Some of the tasks within this part are to set the next timeout, set transceiver state, start transmitting message, execute some non-critical but short tasks, etc. Upon each timeout the timer module will send and event to the MAC State Machine with the timeout type. The state machine could then asynchronously execute the rest of the tasks required between two consecutive time points. They may not be completed before the next timeout. Examples of such tasks are decryption, invoking link scheduler, prepare acknowledgement, etc. If a task is finished too late for the node to work correctly, error handling will be employed and the current timeslot may be abandoned, which, however, will not affect the timing coordination between two talking nodes.

13.5.3 The Implementation

The Timer Module could be implemented fairly efficiently. The following is a sample code section in C++:

```
mCurrentTimeoutType = mTimeoutType[mCurrentSlotType][mTimeoutIndex];

/* Set the next timeout*/
if (mTimeoutIndex == 0) { /* A new time slot started */
```

```
      mASN++;
      mASNTime = mCmpTime;
      mRxTime = mASNTime + mInterval_TsTxOffset; // expected receive time.
      mGetRxStart = FALSE;
      mCurrentSlotType = mNextSlotType;
      mNextSlotType = eMacTimerIdleSlot;
      mTimeoutIndex = mTimeoutsPerType[mCurrentSlotType] - 1;
      mCmpTime = mASNTime + mTimeoutInterval[mCurrentSlotType][mTimeoutIndex];
    } else {
      mTimeoutIndex--;
      mCmpTime = mASNTime + mTimeoutInterval[mCurrentSlotType][mTimeoutIndex];
      if (mTimeoutIndex == 0) {
        if (mPositiveAdjust) {
          mCmpTime += mAdjustTime;
        } else {
          mCmpTime -= mAdjustTime;
        }
        mAdjustTime = 0;
      } else if ((mTimeoutIndex == 1) && (mCurrentSlotType == eMacTimerReceiveSlot)) {
        mCmpTime = mRxTime + mInterval_TsMaxPacket + mInterval_TsTxAckDelay; //
ACK's sent time is based on sender's time.
      }
    }
    TMR2InitCompareRegister(mCmpTime);

    /* notify Mac */
    mpMacTimerCallBackFunction(mCurrentTimeoutType);
```

The method could be applied to other similar standards where lower powerful processor has to meet tight timing requirements.

If a device is too slow to complete its job correctly in a timeslot, its appeared timing behavior to the other devices will still conform to the WirelessHART standard.

The partition of the work between synchronous and asynchronous components could be decided based on how powerful the processor is.

13.6 Hardware Considerations

Among low-power low data rate mesh networks, WirelessHART is at the high end with regard to memory and power usage. We have calculated in Section 8.6.1 that more than 7K RAM is required just to store the minimum number of data structures defined in the data link and network layers.

We do not have precise numbers for code size of a complete WirelessHART stack. As reference we provide the code space of the Wi-HTest's access point, which is implemented on the FreeScale JM128 board.

Table 13.1 Wi-HTest AP Code Size.

Code Component	Code Size (Kilo Bytes)
Stack	40
RTOS	7
System Library	11
CDC-USB driver	11
Encryption Engine	20
Boot Loader	3

The stack code includes the physical layer, the data link layer, and a thin application layer. Note there is no network layer here. We implemented the encryption engine in software. It should not require 20K if hardware encryption accelerator is used. Most of the IEEE 802.15.4 chips on the market provide such hardware support.

If we ignore the time consumed running encryption and decryption, most of the embedded processors are fast enough for WirelessHART stack. The Wi-HTest access point uses FreeScale JM128 CodeFire V1 processor which is a 32 bit processor running at 28 Mhz. Since we use the software encryption engine, we have to run it incrementally, which is sufficient enough to meet the need of the Wi-HTest.

The WirelessHART standard intentionally uses the IEEE 802.15.4 standard. The big advantage is that the commercial of the shelf radio chips for the IEEE80 2.15.4 standard could be directly used for WirelessHART development, and no quality problem is to be worried about. In addition, the packages with those radio chips also come with the hardware encryption engine.

The most common commercial solution is combining the radio and the processor together, either system-on-chip such as CC2430™ from Texas Instrument (http://www.ti.com) or system-on-package such as MC1321 and MC1322 from FreeScale (http://www.freescale.com). For these chips we have to be careful because usually the processor power and memory size is limited. It is hard to implement a full WirelessHART stack with such products that were on the market before the WirelessHART standard was released.

13.7 Miscellaneous Comments

A hypothetical use of the Wi-HTest and Wi-Analys Tools

- Initially the device under development was able to receive advertisement messages and send out join request. However the join request was never received by the proxy device.
- After the device is placed with the Wi-HTest tool, it is noticed via the Wi-Analys tool that the device sent out join request about 2 ms too early. It turned out that the timestamp adjustment was off. After the adjustment in the device, the HTest-W tool acknowledged the join request at the MAC layer.
- But now the Wi-HTest tool did not reply to the join request. It turned out that the join request message format was wrong. After this fix the join reply was sent out by Wi-HTest.
- But the device did not acknowledge the join reply at the MAC layer. It turned out that after sending out join request, the device timed out before the join reply arrived. After adjusting the time out value, the device also received the join reply and forwarded to the application upper layer.
- Now the developers started implementing the code to handle the join reply.

Timing is everything

The radio is only turned on in transmitting or receiving timeslots. Within the timeslot, the radio only listens in a small window. This is good to preserve battery, but makes development difficult. In many times a device does not handle a message simply because it is not listening when the message is streamed in. While we could debug the device for incoming messages, it is often very frustrating when your device under development sends out a legitimate message but the receiving end is silent. You check the Wi-Analys and do not see anything wrong with your one message, except, after excruciating looking around, you noticed that the timestamp recorded by Wi-Analys shows that your message is off.

How to measure internal hardware delay

To make sure the transmission is right on time, we need to initiate the transmission a short time before the expected time to counter for the delay from the call to the transmit function to when the message is actually in the air. This delay could be measured with the help of the transmit interrupt which is generated when transmission starts. Record the time when the interrupt comes in; compare it with the time when transmission is called. The difference is the delay.

To measure receiving delay, we could make use of the Wi-Analys tool. The receiving delay is the time difference between the actual message arrival time in the air and when the incoming message is timestamped. In the transmission timeslot, the time distance between when the transmission function is called and when the acknowledge message is timestamped consists of three parts: the internal transmission delay, the actual time distance between the transmitted message and acknowledgement message, and the internal receiving delay. Assume that we have already measured the transmission delay. We could calculate the middle part from the Wi-Analys log. So we could derive the receiving delay.

The clock skew algorithm

The time points in a timeslot is realized with the processor's own clock, whose one tick may not exactly divide 1µs, the time unit in the WirelessHART timeslot. For example, the JM board we use for the Wi-HTest tool runs at 24MHz. Its timer tick is scaled by 16 to 1.5MHz. So we use every 1.5 ticks of the timer to represent 1µs to trigger the WirelessHART counter. To adjust for small clock drift, we need to periodically increment or decrement the WirelessHART counter. The algorithm we use is to occasionally change the counter by one at the beginning of a timeslot. We use two numbers to select the timeslot, the long counter and the short counter. The algorithm restarts after every long counter number of timeslots. In each repeat, we count the timeslots. When it reached the short counter, we choose the timeslot to adjust and then restart the short counter. The algorithm is simple, versatile, and easy to implement. It also avoids the potential risk of missing a time interruption counter value that happens to be skipped by the adjustment.

The JM128 DEMO board we developed with has various onboard crystals and we were able to fine tune all of them to progress at the very precise rate. The final Wi-HTest tool product uses a high precision oscillator and we no longer need to tune each board.

The algorithm now serves as a support for false injection tests. The Wi-HTest tool could intentionally skew faster or slower to test the DUT's ability to keep synchronized with its time source. A test script could set the long period, short period, and the skew direction through the provided function.

Other comments

- We could use timeslot as the unit for upper layers' clocks or timers.
- When building advertisement messages, make sure that the join links' directions are set right. The device's own transmit join link should be set as receiver link in the message, and vice versa. The link definition in the message is defined for the joining devices.
- Unless you are working on channel hopping, you could blacklist all but one channel to help debug.
- For two neighboring devices A and B, if device B is configured with device A as the router, make sure there is always working link between them when configure B's links. Also for any link, it's safe to configure the receiver device before the sending device.

Chapter 14 Deployment Recommendations

Abstract Millions of HART field devices installed in the world have rich digital field data ready that is not being collected by the host. In many cases these devices are not connected to the plant asset management systems. In these cases wireless offers a cost-effective means to leverage the intelligence and diagnostic capabilities inside smart devices. A complete WirelessHART solution provides the bridge. It's easy to install; the rich data provided by the devices are routed to the host user with high reliability; and, with standard interfaces, no special software is required to seamlessly integrate the rich data into the hosts to validate quality information. This also offers a perfect opportunity to easily and safely test wireless technology in non-critical monitoring and control applications.

14.1 Scope the Network

Engineers of existing plants should assess their HART-enabled field devices and control loops, and determine where wireless connectivity would have the greatest impact with minimal risk. Every process facility in every industry is different in design, but there are many common features that allow us to apply best practices for network design. While a network could be small of a few devices, too few devices could not maximize the power unleashed by this new technology. On the other hand, too big a network will drag down the performance of the network. For a large facility such as refinery or chemical plant, the network should be scoped to a single process unit. For vertically arranged facilities such as power plants or factories, the self-organizing network should be scoped to a single floor. Good engineering practice will generally assign a gateway to each plant area and/or unit operation, similar to how automation system controllers and I/O systems are done today.

14.2 Design the Network

WirelessHART network could be configured similar to configuring wired HART network. The gateway is the remote I/O system connecting wireless devices and adaptors to DCSs, PLCs, and other plant automation systems. And the access points are the I/O modules of the gateway. The gateway will have one or more access points that connect wireless devices to the gateway. The WirelessHART network starts with the gateway and access points at one end and the field devices at the other. Access points can be geographically dispersed from the gateway electronics and in general should be located near the devices they will connect.

HCF suggested a simple formula to determine the number of devices that could be connected, directly or indirectly, to one access point, which is called the access point loading:

$$\text{NumDevices} = \text{Average update period (AUP)} \times 25$$

For example, with an average reporting rate of once per 1 second, 25 devices can be connected with an access point; with an average reporting rate of once per 10 second, 250 devices can be connected with an access point.

This criterion is similar to that we use for any traditional I/O: Do not crowd the I/O. When in doubt, use more access points. With little additional cost of access points, this increases the number of alternative network paths and makes the network more robust.

HCF also suggests the following formula to estimate the average bandwidth consumed by a WirelessHART network:

$$\text{Bandwidth Consumed} = \text{NumDevices} \times (0.0001\% + (0.02\%/\text{AUP}))$$

Where the 0.0001% is used for overhead (network health reports and the like), and the 0.02% is for data publishing and other network traffic.

For example, 100 devices at average update interval of 1 second will consume 2.01% of total Bandwidth; 1500 devices at average update interval of 60 second will consume 0.65% of total Bandwidth.

These are very conservative estimates that do not account for the size of the area. For large networks, devices farther apart could use the same channel at the same time. And the bandwidth usages in one place will be less than the total network bandwidth due to the distance fading of the radio energy. With this formula the user could estimate whether the WirelessHART network will coexist well with other 2.4GHz networks in the same plant area.

A huge advantage of WirelessHART is its adapter. The adapter can be located anywhere along the current loop from the device to the I/O module. This allows any existing HART devices to be adapted to wireless, but it also frees the designer from having to locate the device in a better reception area instead of where the device physically should belong. Suppose a pressure transmitter must be located at the centerline of a large steel vessel that is mounted only a foot above the floor. Instead of having to remotely mount the transducer from the WirelessHART device, making for very long impulse lines, or remotely mount the antenna from the WirelessHART device, we could install a wired HART device which is remotely connected to a WirelessHART adapter. This would be a simpler and less costly solution.

14.3 Deploy the Network

WirelessHART networks are easier and less costly to install than traditional wired HART systems simply because there are no wires to pull through cable trays. You simply locate where the devices should be mounted in the plant and install them, then install the access points and gateway. Occasionally throw in some routing devices. After the devices have joined the network, simply configure them and they start running. It's as easy as a traditional HART 4-20mA installation, with the same tools and know-how. Once the gateway, access points and devices are installed, the wireless mesh forms and communications begin.

The WirelessHART devices are also preconfigured the same way as the wired HART devices are. The key difference is that WirelessHART devices require the entry of a join key and a network ID. The device uses the network ID to determine which advertisement message to join through (There could be more than one wireless network within range). Once the device has heard one or more neighbors, it uses the join key to send a join request to the network manager. The network manager in-turn authenticates the device and incorporates the device into the network.

The WirelessHART network's self-organizing feature is a proven wireless solution for the process industry, in part because they are so easy to plan, commission, and install. Unlike other wireless field network solutions, such as line-of-sight or point-to-point networks, self-organizing networks don't require detailed site surveys or specialized equipment to implement. They are also much easier to expand. Planning, in advance and for the future using best practices, is the key to successful implementation.

In small networks, the gateway should be located in the center of the network. For large networks or applications that require the gateway to be mounted inside a control room or rack room with remote access points, it is best practice to build the initial network around the location of the access points. Then the network can be expanded to reach remote areas of the process unit. These practices will provide a solid foundation on which to expand the network.

The gateway provides the connection between host applications and control systems and the devices that make up the wireless network. The gateway is responsible for collecting and maintaining cached response messages from all devices in the network. These cached response messages originate from burst mode commands, event notifications, and commonly communicated HART commands. These cached response messages are returned as responses to host-based application requests. The gateway also has additional built-in functionality to support adapters that allow transparent access to the devices connected to the adapter.

In many cases the gateway connects to the mesh network through multiple access points. When multiple access points are in-use the network manager will schedule communication traffic through all of them. If one of these access points fails then the network manager will adjust the schedule by spreading traffic across the remaining access points.

An example of a gateway is the Rosemount 1420 Wireless Gateway (Fig. 14.1). The 1420 includes both the security manager as well as the network manager. The 1420, through a web browser, provides an interface for users to setup security, run diagnostics, and configuration information about the wireless mesh. The gateway is also the entry point for host and DCS systems to access wireless device data. The 1420 enables integration with Modbus, OPC, TCP/IP via Ethernet or serial connections.

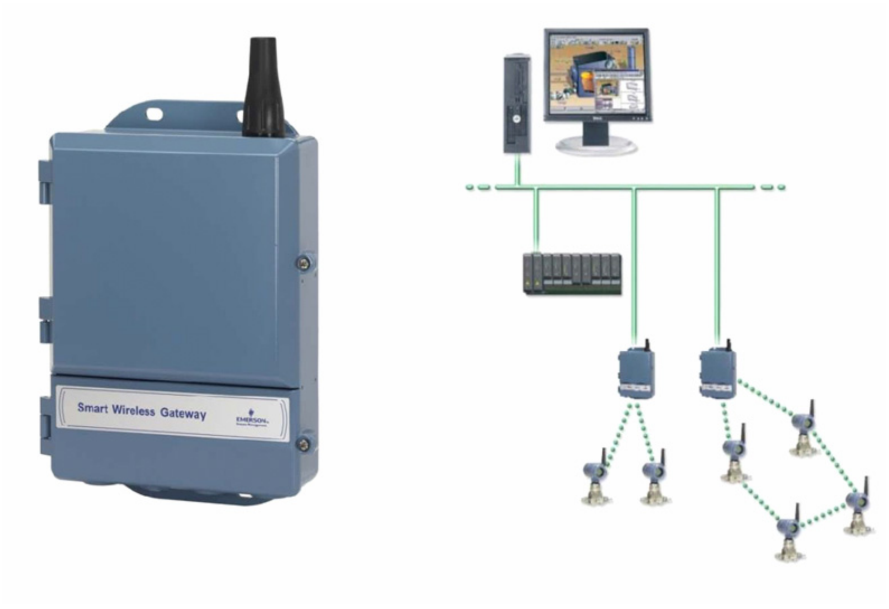


Fig. 14.1. Connecting the Gateway to the Control System.

In many situations the Gateway connects to the control system through native interfaces and shows up as a part of the overall configuration system (Fig. 14.2). In other cases industry standard mechanisms such as Modbus and OPC can be used.

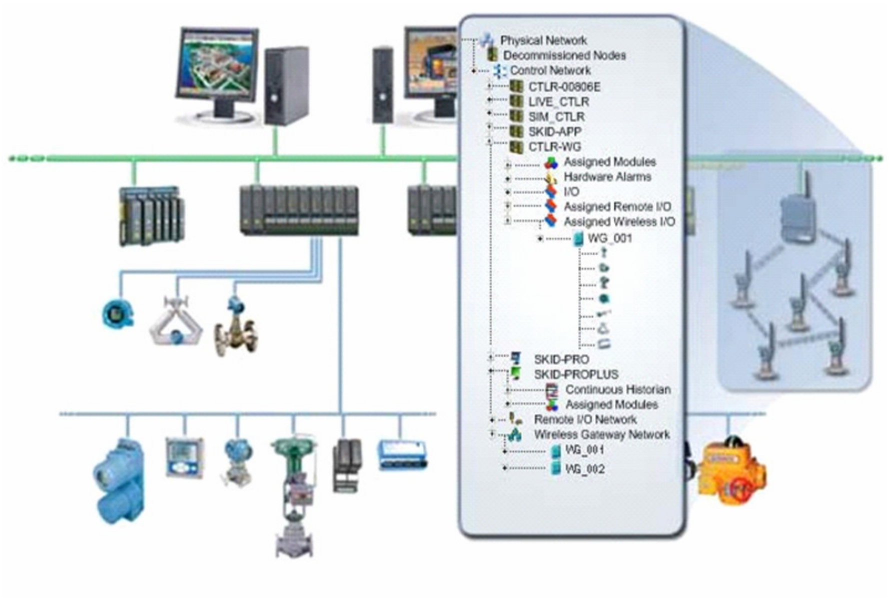


Fig. 14.2. Integration with DCS System.

14.4 More Comments

The following comments are also important to consider in the installation of the WirelessHART network.

- For security, the network manager must be pre-configured with the information of the potential devices in the network. This information will be used to check the security and authentication during the device join sequence. The Join key is mandatory to decipher the join request payload. To be secure, each device should have a different join key. In this way the network manager must also store the device ID or device long tag, both are included in the join request message. Device ID is unique among all HART devices; using device ID to authenticate the device will add extra work in the network manager when we replace a bad device.
- When the network manager configures a unicast link between two devices, it is important that no message be sent by the sender before the receiver has been configured the receive link.
- Device ID could be pre-recorded in network manager together with the join key and the network ID. This is more secure in that only the designated device could be put in the designated location. But it may cause problem for installation

and replacement. How does engineer know which device is supposed to be installed where? What really matters is the right device type installed in the right place. Maybe the network manger should record, before join, the device tag instead, which indicates the location of the device. To assist in the installation of the device the handheld that is being used must be capable of installing join keys into field devices.

- A WirelessHART network requires high device clock precision to remain synchronized. However, not all devices will have high caliber in this regard. The network manager should use shorter keep-alive interval for such devices.
- Deploying access points in pairs will provide better redundancy for the reasons we described in Section 9.6.2.
- Although a WirelessHART network does not require site survey, it is helpful nonetheless. For example, we could use it to find the best locations for routing devices that result in the strongest signal strength and also uses the least routing devices.
- If no new device is expected to join the mesh, the devices could stop sending advertisement messages. This avoids revealing the current ASN and other mesh information to any sniffing device.
- Each device has its own join links. It is possible that two neighboring devices have their join links share the same timeslot but on different physical channels. When configuring links, the network manager should avoid this situation as a new device will have trouble deciding whose join link to follow.

PART IV WirelessHART in the Bigger Picture

In this part we look at the WirelessHART standard in the bigger picture of the status of wireless in the industrial process automation.

Chapter 15 explains why the WirelessHART standard is here and it is ready now, and why it has the great chance of being successful.

Chapter 16 talks about the challenges of applying wireless to the real-time process control. It provides the authors' study of adapting the classical PID control method with wireless communications.

Chapter 17 describes the research opportunities generated by the process control industry's adoption of wireless technology.

Chapter 18 looks at what is next for wireless and the WirelessHART standard, including applications, products, and standardizations.

Chapter 15 Why WirelessHART

Abstract There is a need for wireless at the field level of process plants and the WirelessHART standard meets the need quite well. We shall talk about the need throughout Part IV. In this chapter we show why WirelessHART mesh is the ideal wireless approach for connecting field devices. The WirelessHART standard is based on proven solutions; it embraces the best technology; and it has an easy adoption path.

15.1 The WirelessHART Standard Is Based on Proven Solutions

The WirelessHART standard was released in September 2007. It was based on years of solid pilot experimentations. Years before the standard committee members started drafting the specification, real wireless mesh applications in real process plant had been tried out. It was the success of those experimentations that lead to the birth of the WirelessHART standard. All the lessons and experiences are incorporated into the standard. For example, some of the critical findings during the experimentations are:

- Power consumption is a very critical performance parameter. Users expect years of running time of any field devices before maintenance is required. This requires powerful enough batteries as well as smart management of power usages.
- Without high reliability, work processes won't change. Enough has been said about reliability. Basically wireless has to be as reliable as its wired counterpart before it could gain any traction.
- The number of devices within a wireless network will grow over time. Wireless technology not only replaces wire, but also unleashes a whole new set of potential applications. Very large networks of wireless devices are not far in the future.
- Customers will drive for faster update rates. Any wireless solution must keep this in its design to have a lasting lifetime.
- Point-to-point communications are unreliable in a plant. It is very difficult to position devices to maintain line-of-sight all the time in a process plant. The best wireless communication has to be via mesh topology, which provides multipath redundancies.
- Any mesh must be able to coexist in a crowded and noisy environment. WirelessHART is based on this assumption.

- Site surveys are costly and have a limited chance of success. Two talking devices deployed based on the survey could easily lose communication due to plant dynamics.
- Collision based networks do not scale up. The WirelessHART standard replaces CSMA in the IEEE 802.15.4 standard with TDMA based on this finding.

Customer trials taught us a lot about what works, what does not, what is important, and what is not in the dynamic industrial real-world environment.

15.2 The WirelessHART Standard Embraces the Best Technology

The WirelessHART standard is a backward compatible, cost-effective, common sense approach to wireless communication that supports industry requirements for a simple, reliable, and secure wireless communication technology. The WirelessHART standard is also the HART standard. It could be applied to all process field device applications. These include plant and equipment management, alarms and events. They also include monitoring on level, flow, pressure, temperature, vibration gas detection, etc. either fast or slow. Where appropriate, the WirelessHART network could also be applied to closed loop controls. The WirelessHART solution has the support of major process automation vendors. From process control perspective, it offers flexibility, scalability, and interoperability.

For wireless perspective, the WirelessHART standard includes the best wireless technologies development in low power, low data rate networks. A WirelessHART network is a self-organizing mesh network. It focuses on field device communications. Some of the highlights are:

Mesh Topology

The simplest topology is a star. A tree topology is more versatile but difficult. The ultimate topology is a mesh, which is complex yet powerful. The WirelessHART mesh is a flexible, ad-hoc network organization. There is no hierarchy among the field devices; all devices have the same network capabilities. In other words, there are no RFDs (Reduced Function Devices) or FFDs (Full Function Devices) to complicate matters. Every device that sources and sinks packets is a router. The WirelessHART standard allows adaptation to different applications and environments. There are no special engineering or installation requirements. And it could cover a large geographic area.

Secure

The WirelessHART standard uses industry-standard AES-128 ciphers and keys at both the data link layer and the network layer. It provides both authentication and encryption.

Reliable

Because it uses mesh topology, the WirelessHART network provides redundant paths between the gateway and the field devices. Messages can be routed around interference and obstacles. The network could be configured to tolerate at least single failure. Because of the nature of a mesh, the more devices we have, the more reliable the network is.

Other technologies adopted for reliable communications include frequency hopping, clear channel assessment that is enabled by default, settable transmit-power that is 10dBm by default, black listing of channels, and prioritized communications that ensures satisfying monitoring and control needs.

All these technologies also help in reliable coexistence. A WirelessHART network is tolerant to interferences; it is hard to disrupt. It also minimizes interference to others.

Dynamic Bandwidth Allocation

The WirelessHART standard economically utilizes the communications for battery longevity. Devices request fixed bandwidth for publishing process and control data. They can also request additional bandwidth temporarily, e.g., to transfer vibration spectra. We also could ramp up and down bandwidth usage when performing maintenance on a field device.

Self Organizing and Self Healing

In the WirelessHART network all devices provide statistics to network manager about signal levels, communication reliability with neighbors, and detection of new neighbors. The network manager grooms connections and routes, changes connections and routes for reliability, adds interconnection to make more resilient, and flattens network for increased speed and lower power. If an obstruction is introduced into the mesh network, devices will automatically find the best alternative communication path. The paths will then be readjusted. Information continues to flow. All these are handled by the network manager while the plant applications, without any awareness of the network manager, talk continuously through the gateway with all the field devices.

15.3 The WirelessHART Standard Has an Easy Adoption Path

The use of the latest wireless technologies makes the WirelessHART network easy to be applied, such as the mesh topology and self-organizing self healing nature discussed in above section. Basing the standard on pilot experimentation and customer inputs also makes the HART standard easy for the users. Furthermore, the WirelessHART device’s advantages lies in the fact that it is still a HART device.

The WirelessHART working group has set its Keep-It-Simple philosophy from the start - simple to configure, install, support, and maintain.

The HART standard is the most prevalent process communication protocol measured in the number of devices installed. There are about 30 million HART-enabled devices in the field as of mid-2008. Any technology build around HART standard has a fast track to be adopted by and benefit the customers.

The WirelessHART standard is part of the HART standard. It leverages heavily the existing HART technology, the infrastructure and practices. Its entire application layer follows the HART standard. With HART command structure a WirelessHART device is compatible with HART-enabled control systems and DDL. An existing DD-enabled application or tool could access it just like accessing a normal HART device. A WirelessHART network can do whatever wired HART network can do and more. This enables transparent integration of wireless and wired devices into plant operation. User can add wireless technology to existing installations or brand new ones. Since the HART standard supports an entire range of measurement and control field devices and systems, so is the WirelessHART standard.

While wireless technology is disrupting, adopting the WirelessHART standard via the HART standard is a smooth transition. As a matter of fact, utilizing a WirelessHART adaptor, a traditional HART device could talk to a traditional HART application and the communication is routed through a WirelessHART mesh.

By maintaining the same HART user experience, we save a lot of upfront cost in training. Rather than intensive infrastructure building for a brand new standard, WirelessHART technology allows organic growth from exiting HART installations. The fact that millions of HART-enabled field devices are already installed in the field makes this even more appealing. To adopt WirelessHART technology, a customer could first try it with limited cost, learn about it, then gradually expand it, and finally fully adopt it. The customer could configure the WirelessHART device with the same familiar tool that is used to configure the wired HART device.

It is estimated that only about 25 percent of the total wired HART devices installed are digitally connected so that users can fully leverage the diagnostic information carried by the digital signal. The WirelessHART standard offers a quick and cost effective means of connecting field devices and other assets to plant networks. Without change to existing installations, we could add wireless capability to installed HART devices, which then form a mesh network and all the diagnostics information could then be carried by the mesh to be used by the host application. In the mean time the process data in the devices are still carried in analog via the 4-20mA cable.

For brand new installations, we could cut the 4-20mA wire and make the HART device truly wireless. They will be battery powered, solar powered, process powered, ambient energy powered, etc. The process data will then be digitally transmitted through the mesh. Even for all these changes in the field, the host application in the control room could still be traditional HART one. One reason the HART standard has the most installation than other fieldbuses is its simple installation. WirelessHART installation is even simpler. After the gateway and access points are set up, the devices automatically join, not even the need to deploy 4-20mA cables.

With the capability of mixing both old and new devices, the HART standard enables the customers to add wireless wherever needed and they are comfortable with.

The adoption of the IEEE 802.15.4 standard by the WirelessHART standard also lowers the entry cost. There are already lots of chip manufacturers for the IEEE 802.15.4 standard. This translates to relatively simple and low cost WirelessHART solutions. Using of the 2.4 GHz frequency also means WirelessHART applications could be deployed world-wide.

Chapter 16 Wireless and Real-Time Industrial Process Control

Abstract There are different kinds of applications running in a process plant, ranging from auxiliary to critical. They are categorized into three groups, Class C applications for monitoring, Class B applications for control, and Class A applications for safety. As a new technology, wireless sees its early deployment in monitoring rather than controlling. And the early wireless devices are sensors rather than actuators. Most of the wireless network applications at the device level are sensing for logging and monitoring. Although work is underway to apply wireless to control applications, wide spread use of wireless in control applications and safety systems is still some time off. While wireless technology needs to advance to tackle the industrial real-time challenges, the traditional process control applications also deserve a second look. In this chapter we discuss the challenges of applying wireless in process control in general, and our preliminary study on adapting closed-loop control for wireless communications.

16.1 Challenges of Wireless Control

There has been tremendous interest in the research and development of wireless technology. In general wireless for the control industry is not a new topic. Many industry organizations, such as WINA™, ZigBee, and ISA™, have been pushing wireless technology for years. There is a set of well-known and agreed upon challenges that we have to overcome to apply wireless to industry control, such as security, robustness, delay, and power. After touching on those concerns briefly, we will analyze the challenges in applying control over process sensor networks. While using sensor network for process monitoring has been studied extensively (Akyildiz et al. 2002, Callaway and Callaway 2003, Caro 2004, Chen et al. 2004, Culler et al. 2004, Elbatt et al. 2006, Krishnamurthy et al. 2005, Nixon et al. 2004, Nixon et al. 2005, Sheldon et al. 2005, Soldati et al. 2008, Song et al. 2007, Vieira et al. 2003, Zhang et al. 2009) we are now seeing early adopters testing wireless in control applications (Chen et al. 2005, Hieb 2003). While people agree this will eventually happen, a continued long term effort from both academia and industry is required. Several areas that could be collaborated on are listed here.

16.1.1 Process Control Networks and their Wireless Counterparts

There are three levels of networks in a typical process control systems (Blevins et al. 2002) as shown in Fig. 16.1. Fig. 16.2 depicts its projected wireless counterpart. At the bottom of Fig. 16.1 are control networks that physically manage the plant process. The controllers are connected with the devices, including both sensors and actuators, via the control networks. The controller reads data from the sensors and writes data to the actuators. The network protocols are usually industry standards that provide real-time support and have high predictability and reliability. The range is short and the data size is small. The wireless network at this level is usually called sensor network. We shall address its challenges in Section 16.1.2.

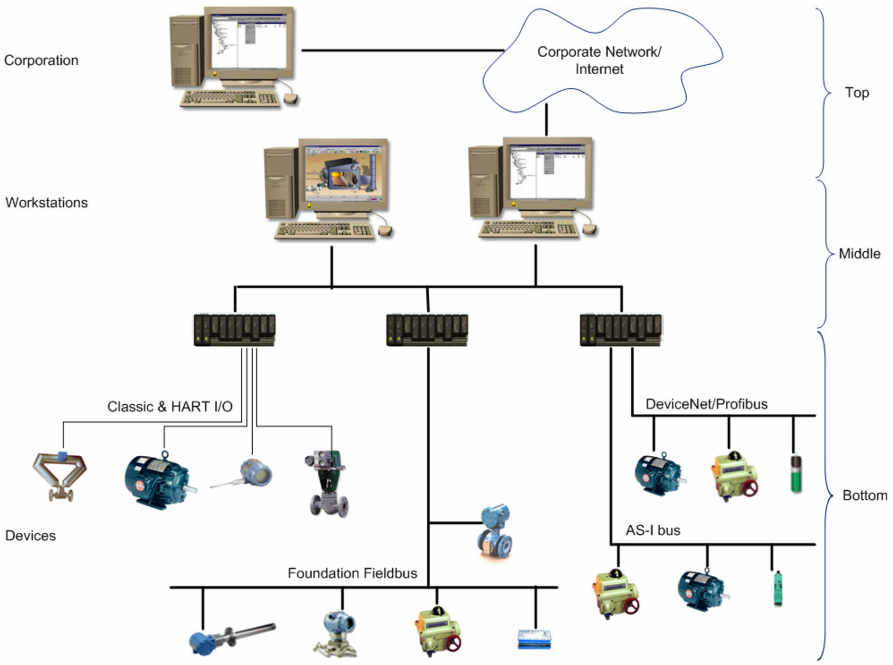


Fig. 16.1 A Process Control System.

We call the middle-level network an area control network. It connects controllers that control devices in the field and workstations that interface with the user. Area control networks carry user interaction data for configuration, control command, monitoring, and diagnostics. It has less timing requirements, but still needs good reliability. The range is longer and the data size is bigger than for the sensor network. Area control networks can be proprietary protocol that utilizes industry communication standards or use industry standards such as Ethernet. Since it is not immediately connected to the field devices, we might use commercial wireless networks as its replacement. Challenges for wireless network at this level are mostly the same as those for commercial networks.

At the top level is the corporate office network. The corporate network is always connected through one or more firewalls. The corporate network provides connections to corporate systems like accounting, inventory, management decision systems, etc. Its wireless counterpart is the commercial wireless network. There are no special wireless challenges with regard to process control at this level. Of course, connecting control networks to office networks poses security concerns.

Challenges for wireless applications are well documented; challenges for wireless applications for process control are also studied extensively. Some of the issues related to wireless become more important for process control, such as security, robustness, and power.

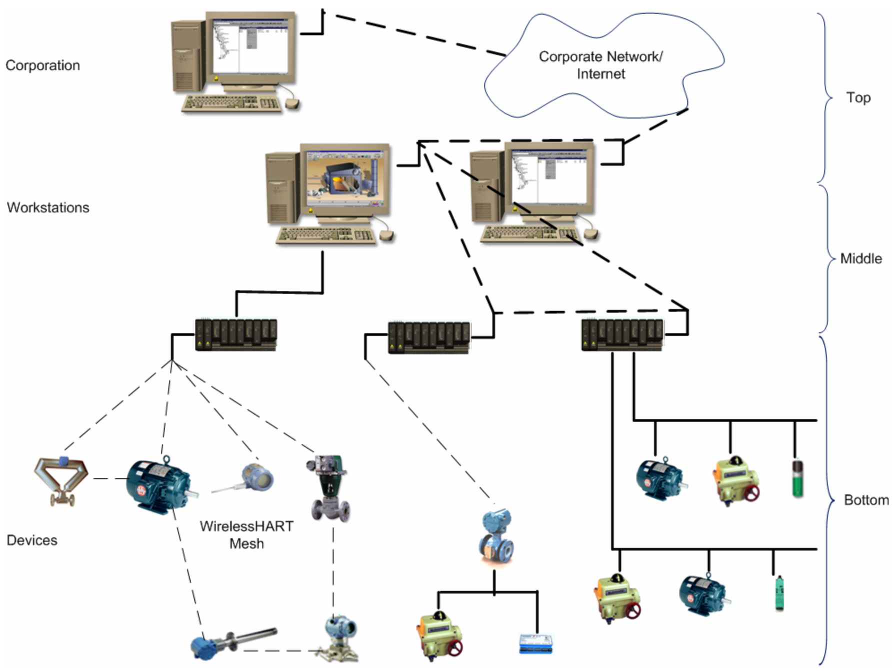


Fig. 16.2 A Wireless Process Control System.

Security becomes more and more important for social reasons. Connecting the control system to the Web aggravates the concern. According to some study (Weiss 2005), Control systems are susceptible to attack because they weren't designed to meet cyber threats;

and,

More than 60 identified (though not publicly documented) real-world cases have occurred where electronic means have impacted the control systems' reliable operations.

Wireless security was one of the biggest topics during the ISA 2005 Expo. ISA's SP99 Committee has defined a common set of information security requirements for control systems that users and vendors alike can reference.

Robustness, including reliability and safety, is a concern because the interference in the process field and the consequence of a failure are much worse. Robustness in many environments that are common in a plant may require a powerful antenna, but higher transmission power poses danger in inflammable space in addition to longer interference ranges. Battery replacement is also more difficult in a plant environment.

16.1.2 Process Control with Sensor Networks

Sensor network studies concentrate on system monitoring. The majority of wireless systems in the field play an auxiliary role to the existing control system. They collect additional data that the control system does not provide. The term *sensor network* unintentionally limited its scope to sensing only. To perform control we need actuators; we need networks of sensors and actuators. Using sensor network to control process poses technical challenges; early adopters are now testing control over wireless applications.

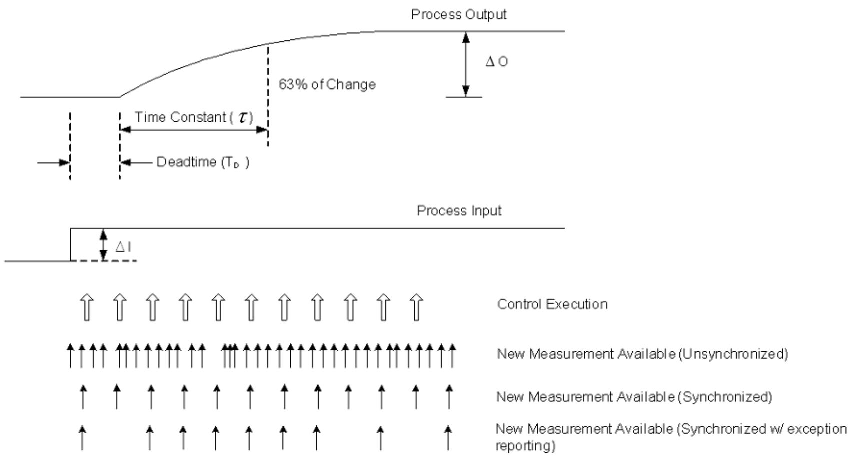

Process Output
ΔO
63% of Change
Time Constant (τ)
Deadtime (T_D)
Process Input
ΔI
Control Execution
New Measurement Available (Unsynchronized)
New Measurement Available (Synchronized)
New Measurement Available (Synchronized w/ exception reporting)

Fig. 16.3 Control Data Sampling Rate.

Utilizing wireless communication to provide measurements that will be used in closed loop control presents many technical challenges. To minimize control variation, the typical rule of thumb is that feedback control should be executed 4 to 10 times faster than the process response time, process time constant plus deadtime. Since the measurement system is often unsynchronized with control, measurement values are sampled much faster than the process responds – in many cases multi-loop controllers over-sample measurements by a factor of 2 to 10 times. This is illustrated in Fig. 16.3.

To reduce transmitter power consumption, it is desirable to minimize how often a measurement value is communicated. By synchronizing measurement and control execution, as is also shown in Fig. 16.3, it is possible to eliminate the need to over sample and over-communicate measurements.

The benefit of replacing wires in a process plant is huge and people will always want to apply control over wireless. It shouldn't be hard for future wireless technology to achieve the same level of functionality as current fieldbuses. We believe advances will come sooner if both the industry and academia spend more effort tackling wireless control.

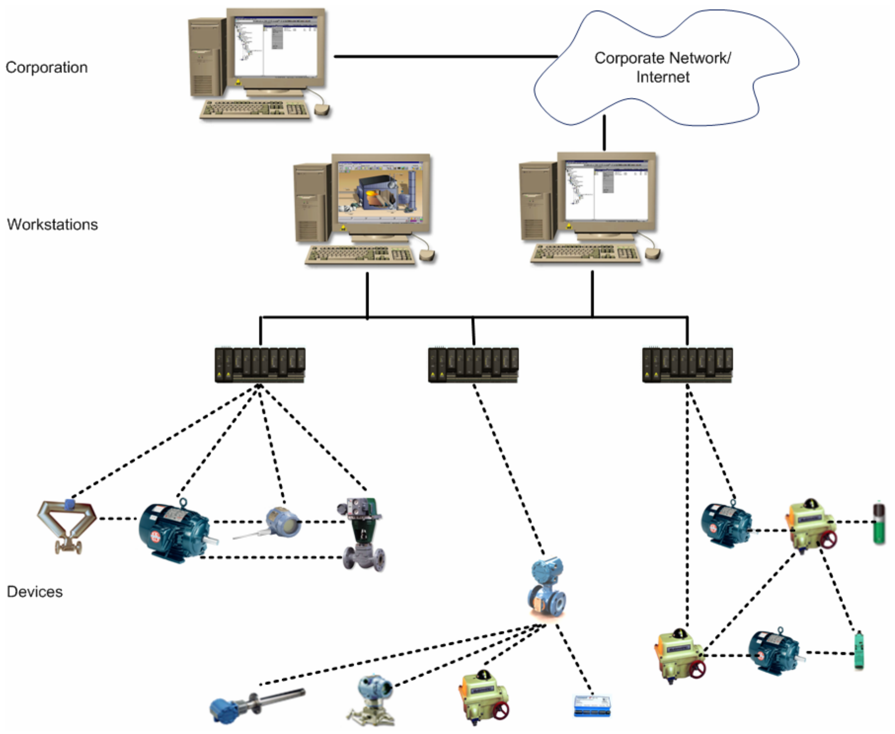


Fig. 16.4 Sensor Networks in Process Control Systems (Large System).

Fig. 16.4 illustrates a possible sensor network in a large control system; Fig. 16.5 illustrates a possible small control system. Let's look at the fieldbuses they replace. To achieve process control, the fieldbus network communication and associated control function and scheduling are designed to be fully deterministic. There is no traffic interference or variation in control organization. Ongoing work in network power usage and communication optimization will continue for some time as companies looking for the best scheduling algorithms.

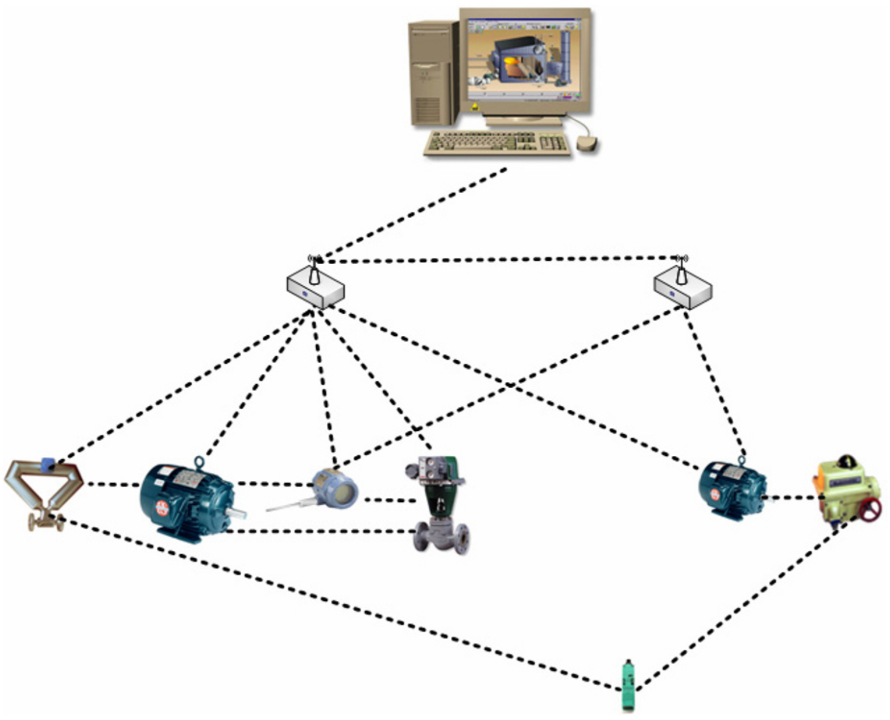


Fig. 16.5 Sensor Networks in Process Control Systems (Small System).

Temporary interference Sensor networks use the open air as communication medium. Whatever happens in open air can cause interference to the data transmission. Events such as the weather, people and things moving around, and other wireless signals can interfere with transmissions. Temporary interference impacts timely data transmission which directly challenges the objective of real-time process control

Permanent interference Once a fieldbus is deployed, it will function throughout the lifetime of the control system. A deployed wireless control network however, must be re-configured through its life cycle. The communication between two nodes may change permanently due to addition or removal of field devices that are not related to the wireless network.

Power usage Power issues related directly to real-time control include the contingency handling of power outage, data transmission delay variance due to battery levels, etc.

Beyond duplicating what fieldbus could do, sensor networks open up new possibilities and hence new challenges for real time control. One of the next steps is to connect the sensor network directly to the user workstation instead of the dedicated controller. For small systems there could be no controllers, and the devices talk directly to the workstations which also run user applications. In the future, every operating system should have some flavor of a real-time operating system. As shown in Fig. 16.4, for large system, the controllers are still required for control coordination and data processing and servicing, e.g., alarm management and history data collection.

Fieldbus and sensor network differ in many ways. While sensor network is inherently less robust, it compensates with multipath capability. Also lower cost and higher communication speed than that are available with fieldbus networks mean more devices could be deployed and more process data could be monitored and controlled. To simply mimic fieldbus, sensor network must synchronize all the nodes so that data could be time-stamped with the global clock, a central node must coordinate all the data traffic. This increases the complexity of a sensor, which goes against the other goals of sensor network: to reduce cost, reduce energy consumption, and have plenty of small sensors/actuators.

A better question could be asked: how could sensor network achieve the same level or even better process control compared with fieldbus? Is there an approach different from that of the fieldbus? To answer this question, we should understand what the goal of process control is.

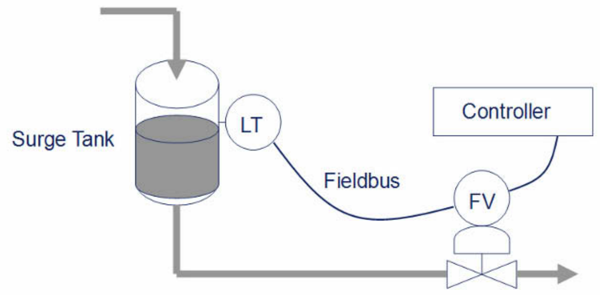


Fig. 16.6 Tank Level Control with Fieldbus.

Figure 16.6 is an example of a plant process and a fieldbus solution. The input pipe fills the tank at a rate based on the upstream process operation. The goal is to keep the tank fluid within a level range by controlling the output flow. The controller receives tank level value via the fieldbus from the level transmitter and sends valve position value via the fieldbus to the valve. In many process plants, significant loss will occur if the tank level exceeds upper or lower range limits.

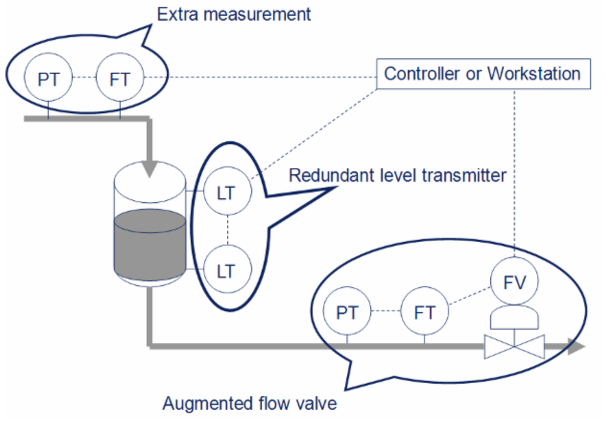


Fig. 16.7 Tank Level Control with Sensor Network.

Figure 16.7 applies sensor network to achieve the same goal. The challenge is if we could beat the performance of the system in Figure 16.7 by using more sensors in field devices and make use of other current and future technologies of sensor network. Additional or redundant measurements can be built into the next generation of field devices, e.g., flow and upstream pressure measurement as part of the control valve. Maybe a new control paradigm could emerge.

People are pushing hard to adopt wireless for process control. At present wireless applications in process control are mainly for monitoring purposes. Issues and challenges are also well-known and agreed upon in this area. However, the long term challenge lies in control with sensor network. We think there is significant benefit in the industry and academia continuing to work together to address these problems.

16.2 Improving PID Control with Unreliable Communications

For standard industrial process control systems, blocks in a control loop are executed periodically. That is, the sampling data is fed to control blocks by sensors at a fixed rate. At the same rate, control blocks do some calculations on the inputs and send out commands to actuators. This paradigm works well with reliable wireline networks, such as Fieldbus and PROFIBUS. However, it is a justified assumption that part of the wired networks will be replaced by wireless networks. With wireless networks, we shall expect intermittent communication losses. In this section, we first identify the poor dynamic response of the standard PID algorithms in the case of lost communications. An enhanced PID algorithm is proposed to improve the dynamic response under these conditions. When there is no communication loss, the enhanced PID block acts exactly the same as a standard PID block. Lost data is compensated by the integral component in the enhanced PID block. When communications are reestablished, the derivative component in the enhanced PID block eliminates possible spikes in the output. We evaluate the enhanced PID algorithm under several wireless scenarios. The results demonstrate the advantages of the enhanced PID algorithm (Chen et al. 2006).

16.2.1 The Control Loop

A sensor provides measurements and status of a physical property, e.g. the flow in a pipe associated with the process. Based on the measurement from sensors, the controller determines any adjustment in the actuators that is needed to maintain the process at a target value, i.e., the setpoint. The control loop is executed periodically at a rate fast enough to correct any unwanted deviations in the process.

Therefore, unreliable communications presents a very challenging problem for standard control paradigms. Consider a PID block with inputs from a wireless channel. Suppose the inputs are lost at time t_1 and reestablished at time t_2. The derivative component of the PID would cause a spike in the output at t_2. Also, from t_1 to t_2, the reset component may wind up based on the error that existed at time t_1.

In this study, we focus on the most widely used control blocks – PID blocks. We revise the calculation of the integral and derivative components of PID algorithms by detecting missed communications and compensate for it proactively. Simulation results on several wireless scenarios prove the superiority of the enhanced PID algorithm.

16.2.2 The Standard PID Algorithm

PID is the most widely used control algorithm in industrial process control. As shown in Fig. 16.8, the controller compares the process variable (PV) with a reference setpoint (SP). The error is then processed to calculate a new output to bring the PV back to its desired SP.

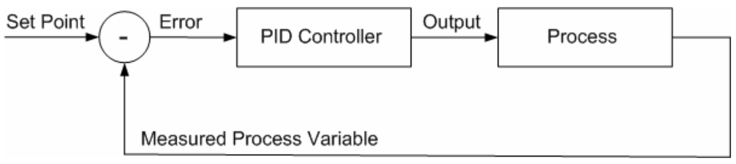


Fig. 16.8 PID Block.

PID stands for *proportional, integral, and derivative* components of the algorithm. Each of the three components performs a different task and has a different effect on the functioning of a system. Their outputs are summed up to produce the system output.

Though there are many variations of PID algorithms, in its non-interacting form without rate limiting and all actions based on error, the equation for standard PID algorithm is

$$Output = K_P\left[e(t) + K_I\int e(t)d(t) + K_D\frac{de(t)}{dt}\right]$$

K_P, K_I and K_D are the proportional, integral and derivative gains, respectively.

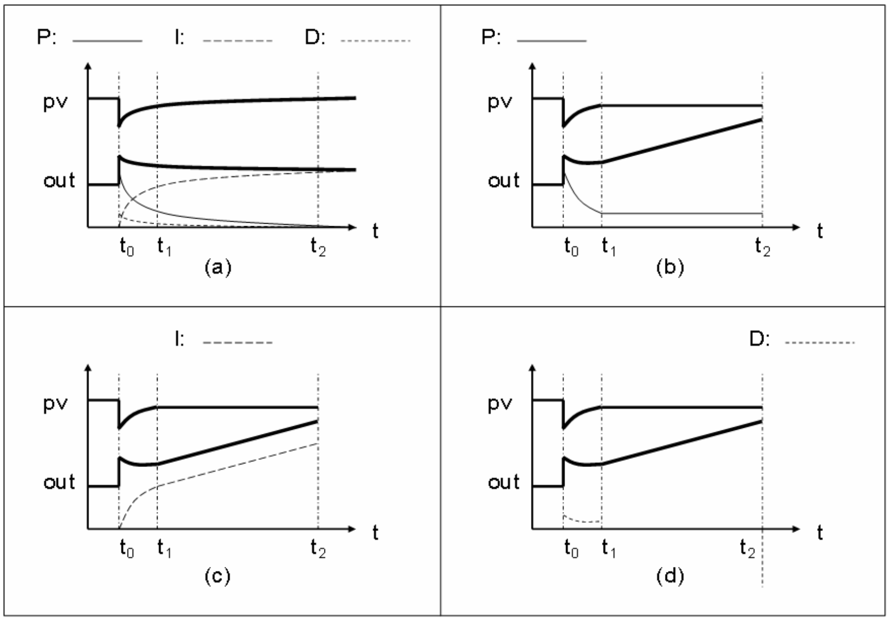


Fig. 16.9 Standard PID Block with Lost Input.

In its digital form, the software implementation of the PID algorithm is based on the sampled data for its PV being provided on a periodic basis.

When there is no communication loss, PID, once configured for the process it controls, keeps the process in a steady state. Fig. 16.9(a) shows the PID reaction to a process disturbance.

Before time t_0 the PID output (*out*) is kept at a constant value to maintain *pv* at the *sp* value. At t_0 a drop of PV is observed due to process disturbances. To correct the drop, the PID increases its output. At time t_2, *pv* comes back at *sp*, and *out* is stabilized at some value that is slightly bigger than its original value to compensate the disturbance. As is shown in Fig. 16.9(a), *out* is the sum of three parts: P, I, and D.

16.2.2.1 Input Communication Lost

Now consider what would happen to each component of *out* if the communication from the sensor input is lost between times t_1 and t_2. For the PID block, the measured *pv* value remains the same as at time t_1 during this period, shown in Fig. 16.9(b,c,d).

Fig. 16.9(b) shows the proportional gain P. Since the measured *pv* and *sp* remains the same, the proportional gain is constant from t_1 to t_2. Fig. 16.9(c) shows the integral part I. Since *pv* and *sp* remains the same, the error remains the same, so the integral part is a linear increasing line from t_1 to t_2. Fig. 16.9(d) shows the derivative part D. Since *pv* and *sp* remains the same, the error remains the same, so the derivative stays 0 from t_1 to t_2. As a result, *out* of the PID block is a linear increasing line from t_1 to t_2, shown in Fig. 16.9(b,c,d). This destabilizes the process. The longer the communication is lost, the bigger deviation *pv* is from *sp*.

Once the communication is reestablished at t_2, PID is back to normal. However, since the derivative part calculated at time t_2 is based on the difference of measured *pv*'s between t_2 and one period before t_2, we shall expect a spike for the derivative part. This is because the *pv* value at t_1 is used as the *pv* value at one period before t_2. The value of *pv* at t_2 could be significantly different from the *pv* value at t_1. Since *pv* changes between time t_1 and t_2, we expect the derivative spike even bigger. Due to sudden changes of the proportional and derivative parts, the value of *out* will have a big impulse before and after t_2.

16.2.2.2 Output Communication Lost

We continue to analyze how standard PID behaves if output communication is lost between t_1 and t_2. Here we assume there are no other disturbances and input communication is OK.

Since t_1 the actuator will stay with the *out* value of t_1 until t_2 when a new *out* value is received from the PID. This will cause the measured *pv* to eventually reach *sp* and then overshoot a little bit, shown in Fig. 16.10(b,c,d). The P, I, and D components are all calculated based on the current pv, shown in Fig. 16.10(b,c,d). This is as good as we could expect from PID. The only drawback is that the actuator gets a bump in the *out* value, from the one calculated at t_1 to the one calculated at t_2.

16.2.2.3 Both Input and Output Communication Lost

When both input and output communications are lost, the PID behaves the same as when only input communication is lost in Fig. 16.9. The only difference is that when communication is reestablished at t_2, the actual *pv* is different. In this case *pv* strays less as the actuator output stays constant. Similarly, the actuator receives a bump in the *out* value.

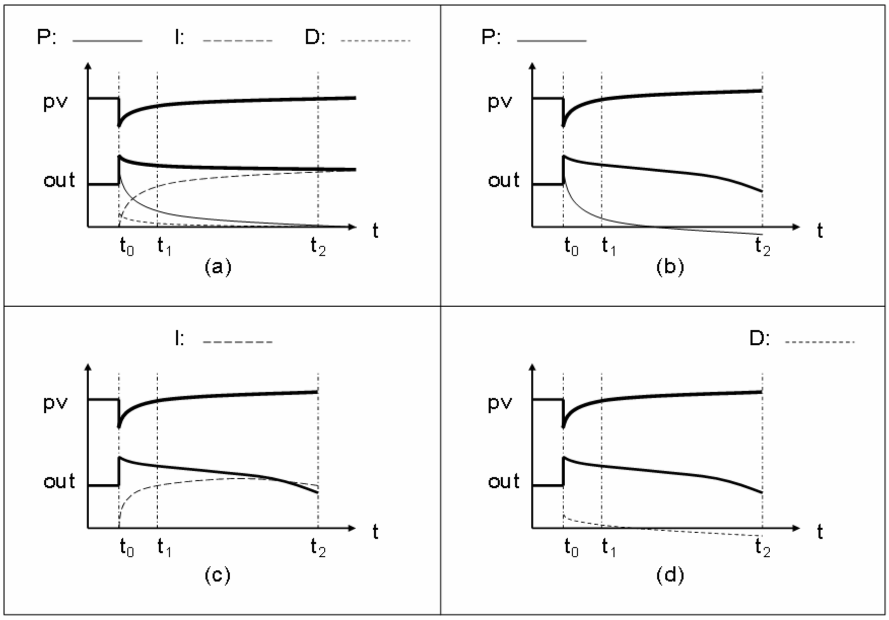


Fig. 16.10 Standard PID Block with Lost Output.

16.2.3 The Enhanced PID Algorithm

The underlying assumption in the digital implementation of the PID algorithm above is that the algorithm is executed on a periodic basis. When the input containing the measurement is lost, the calculated reset action may not be appropriate. When later on a new measurement gets through, the calculated derivative action may produce a spike in the output. If a PID block continues to execute using the last process variable, the output will continue to move based on the reset tuning and error between the last measured process variable and the setpoint. If the control block is only executed when a new measurement is communicated, then this could delay control response to setpoint changes and feedforward action on measured disturbances. Also, when control is executed, calculating the reset contribution based on the scheduled period of execution or on the time since the last contribution may result in changes that increase process variability.

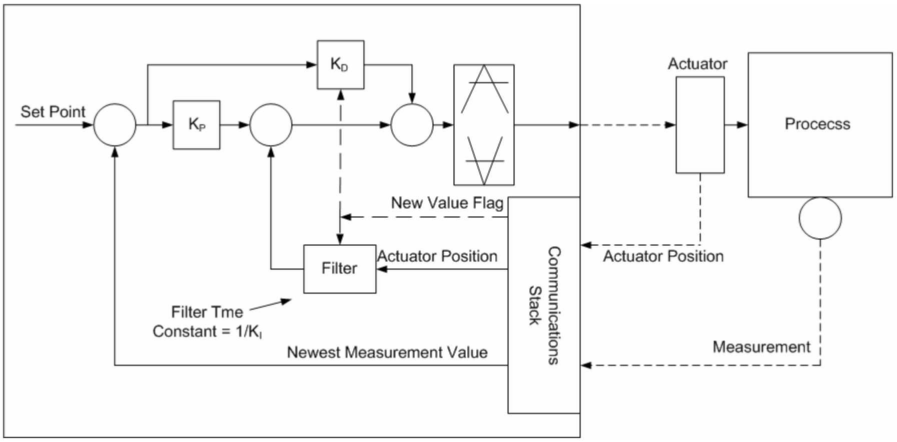


Fig. 16.11 The enhanced PID algorithm application.

To provide best control when measurements are not updated on a periodic basis, the PID may be restructured to reflect the reset and derivative contributions for the expected process response since the last measurement update. One means of doing this is illustrated in Fig. 16.11.

As shown in Fig. 16.11, the reset/rate contributions (integral/derivative parts of the PID) are determined based on the use of a new value flag from the communications stack. To account for the process response, the filter output is calculated in the following manner when a new measurement is received:

$$F_N = F_{N-1} + \left(O_{N-1} - F_{N-1}\right) * \left(1 - e^{\frac{-\Delta T}{T_{Reset}}}\right)$$

Where F_N = New filter output; F_{N-1} = Filter output for last execution; O_{N-1} = Controller output for last execution; and ΔT = Elapsed time since a new value was communicated.

Since the last communicated actuator position as reflected in the feedback of actuator position is used in the integral calculation, this automatically compensates for any loss in the output communicated to the downstream element.

The derivative part for this example (rate limiting not applied) is determined by the following equation:

$$O_D = K_D \bullet \frac{e_N - e_{N-1}}{\Delta T}$$

Where e_N = current error; e_{N-1} = last error; ΔT = Elapsed time since a new value was communicated; and O_D = controller derivative term.

Consider the contribution of the derivative part when the inputs are lost for several periods. When the communication is reestablished, $e_N - e_{N-1}$ in the equation above would be the same for the original and modified algorithms. However, for the standard PID algorithm, the divisor in the derivative part would be the period, while that in the new algorithm is the elapsed time between two successfully received measurements. It is obvious that the modified algorithm would produce smaller derivative action than the standard PID algorithm.

There are two major problems with standard PID algorithm when dealing with communication losses: continued execution during communication loss and sudden output change when communication is reestablished. The enhanced PID algorithm solves these problems by only computing the integral and derivative components when communication is established and incorporating actuator feedback into the reset calculation.

16.2.4 Experiments and Results

We carried out several experiments to validate the algorithm. First, we prove that the new algorithm will produce the same result as the regular algorithm when the communication is reliable. Then, the revised algorithm is shown to have better performance than the existing PID algorithm when communications are unreliable.

16.2.4.1 Experimental Setup

We create two simple PID control loops, as shown in Fig. 16.12.

PROC_1 and PROC_2 are two identical processes, each of which consists of a second order process with a delay of 1 second and time constants of 6 seconds and 3 seconds.

The modified PID algorithm is implemented in PIDPLUS, and PID2 is a standard PID block. The parameters for PID2 are determined by testing it with a tuning application, which suggests a gain of 0.85, a reset of 10.71 and a rate of 1.71. Then the tuning parameters of PIDPLUS are set the same as PID2. PIDPLUS is configured to utilize the BKCAL_IN value for the reset components.

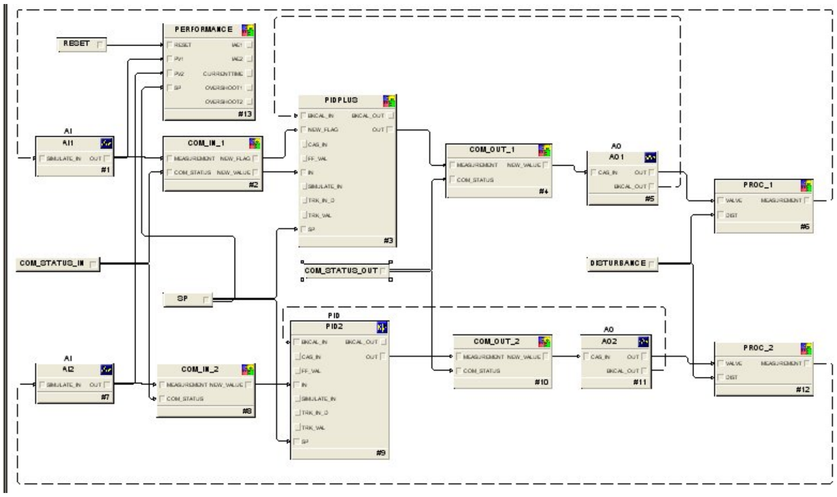


Fig. 16.12 Experimental Setup.

The process variable communication is simulated by the COM_IN_1 and COM_IN_2 block, which are controlled by COM_STATUS_IN. If COM_STATUS_IN is set to 1, block COM_IN_1 and COM_IN_2 relay measurements accurately. Otherwise, the two blocks drop the measurements. The same logic is applied to the output using COM_OUT_1, COM_OUT_2 and COM_STATUS_OUT.

By changing the external setpoint and introducing some disturbances that impact each PID and associated process equally, we can evaluate the performance of the two PID blocks. The performance of the two PID blocks is collected in the PERFORMANCE block. The metrics used here is Integral Absolute Error (IAE).

The scan rate for all blocks is set to 0.2 second. Initially, the uncontrolled disturbance (DISTURBANCE) to the processes is 20.

16.2.4.2 Reliable Communications

To simulate the case of reliable communications, we set both COM_STATUS_IN and COM_STATUS_OUT to 1. SP is changed from 50 to 60. The result is shown in the left part of Fig. 16.13 (from time 11:10 to 11:11).

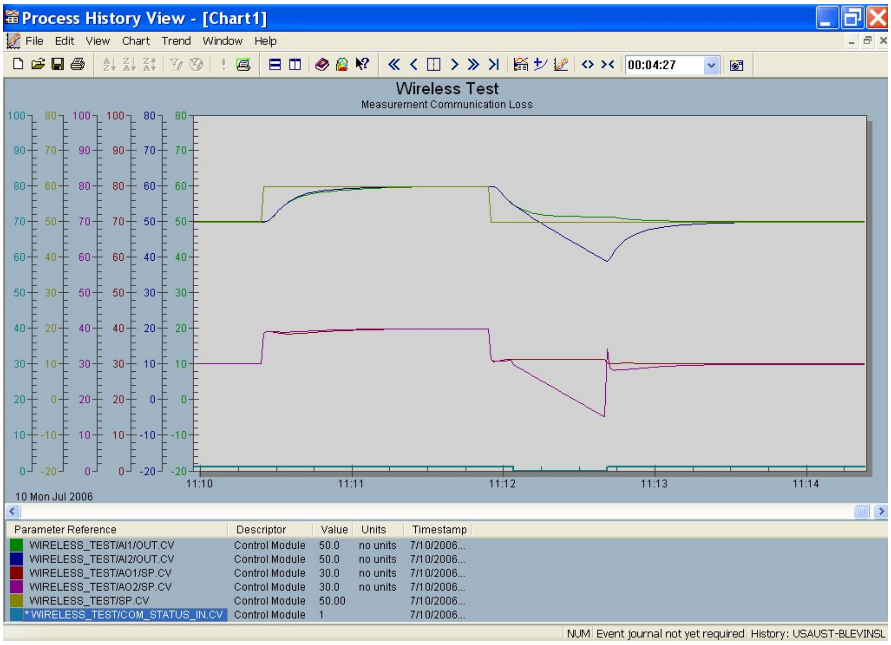
The curve of AI1/OUT.CV matches nicely with that of AI2/OUT.CV, and AO1/SP.CV matches nicely with AO2/SP.CV. Therefore, we conclude that the two PID blocks work in the same way when the communication is reliable.

16.2.4.3 Unreliable Communications

To study the effect of unreliable communications on the two PID blocks, we consider two cases: unreliable input and unreliable output.

Unreliable Input

During the period of lost inputs, the last communicated process variable is maintained and used in the PID blocks. We first experiment with changing setpoints. The result is shown in the right part of Fig. 16.13, where SP is decreased from 60 to 50.


Process History View - [Chart1]
File Edit View Chart Trend Window Help
00:04:27
Wireless Test
Measurement Communication Loss
11:10
11:11
11:12
11:13
11:14
10 Mon Jul 2006

Parameter Reference	Descriptor	Value	Units	Timestamp
🟩 WIRELESS_TEST/AI1/OUT.CV	Control Module	50.0	no units	7/10/2006...
🟦 WIRELESS_TEST/AI2/OUT.CV	Control Module	50.0	no units	7/10/2006...
🟥 WIRELESS_TEST/AO1/SP.CV	Control Module	30.0	no units	7/10/2006...
🟪 WIRELESS_TEST/AO2/SP.CV	Control Module	30.0	no units	7/10/2006...
🟨 WIRELESS_TEST/SP.CV	Control Module	50.00		7/10/2006...
🟦 *WIRELESS_TEST/COM_STATUS_IN.CV	Control Module	1		7/10/2006...

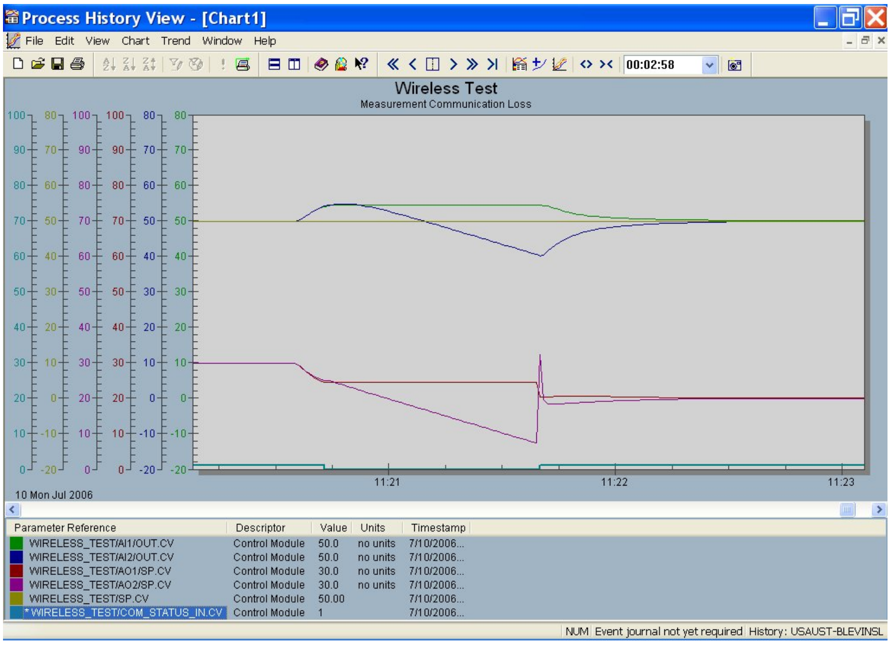
NUM Event journal not yet required History: USAUST-BLEVINSL

Fig. 16.13 Lost Inputs coupled with a setpoint change.

When the input channel is shut down, the error between the setpoint and process variable fed to PID blocks is a constant. For the standard PID block, PID2, the integral part would keep integrating, which explains the linear decreasing of AI2/OUT and AO2/SP. However, as it has a flag for missed communications, PIDPLUS would simply freeze the reset component during loss of communication, which explains the level-off in AO1/SP during lost communications. Since AO1 is given a constant value, the process variable of PROC_1 approaches the setpoint gradually, as shown by AI1/OUT.

When communications are re-established, the input process variables to PIDPLUS/PID2 reflect the true measurement provided by AI1/AI2, respectively. For PID2, AI2/OUT is very low compared to the setpoint, which causes a sharp spike in AO2 by the derivative part of PID2 at the moment communications are re-established. For PIDPLUS, AI1/OUT is close to the setpoint, and that small deviation is further evened out by the divisor used in the derivative part of the new algorithm. Thus both AI1 and AO1 transit to their steady states smoothly.

The different behaviors of the two PID bocks are further proved by the performance data. In duration of 121 seconds, the IAE for PIDPLUS is 169, while that for PID2 is 372.


Process History View - [Chart1]
File Edit View Chart Trend Window Help
00:02:58
Wireless Test
Measurement Communication Loss
11:21
11:22
11:23
10 Mon Jul 2006
NUM Event journal not yet required History: USAUST-BLEVINSL

	Parameter Reference	Descriptor	Value	Units	Timestamp
🟩	WIRELESS_TEST/AI1/OUT.CV	Control Module	50.0	no units	7/10/2006...
🟦	WIRELESS_TEST/AI2/OUT.CV	Control Module	50.0	no units	7/10/2006...
🟥	WIRELESS_TEST/AO1/SP.CV	Control Module	30.0	no units	7/10/2006...
🟪	WIRELESS_TEST/AO2/SP.CV	Control Module	30.0	no units	7/10/2006...
🟫	WIRELESS_TEST/SP.CV	Control Module	50.00		7/10/2006...
🟦	*WIRELESS_TEST/COM_STATUS_IN.CV	Control Module	1		7/10/2006...

Fig. 16.14 Lost Inputs coupled with unmeasured disturbances.

We also test the two PID blocks with unmeasured disturbances. The DISTURBANCE is increased from 20 to 30. The result curves are shown in Fig. 16.14. The behavior of PID2 is the same as in Fig. 16.13, which can be explained by the same reason for Fig. 16.13. For PIDPLUS, the reset component is maintained constant during the loss of communications. Thus AO1 is kept the same output value, which in turn produces the same AI1 value. When the input communications resume, PIDPLUS starts to change its output (AO1/SP) to bring AI1/OUT back to the setpoint. During the time (196 seconds), the IAE for PIDPLUS is 333, and that for PID2 is 366.

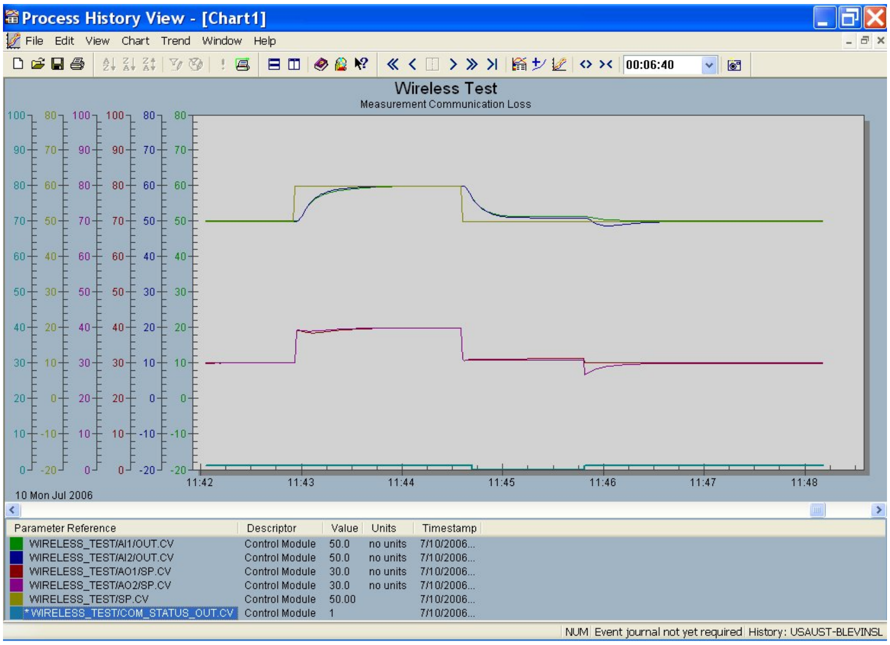
Unreliable Output

In this case, we examine the behaviors of the two PID blocks in two scenarios too: setpoint changes and uncontrolled disturbances.

For setpoint changes, we first change the setpoint from 50 to 60, when the communication is reliable. Then, after the processes settle at the setpoint 60, the setpoint is changed back to 50, and the output channels are cut off. Fig. 16.15 shows the resulting curves.

When the communication is lost, the outputs of PIDPLUS/PID2 are equal. Since the two controlled processes are the same, the values of AI1.OUT and AI2.OUT follow the same curve. When the communication is reestablished, the input errors for PIDPLUS and PID2 are the same. However, the divisor in the derivative part of PIDPLUS is much bigger than that in PID2, which explains the sharper spike in the curve for AO2/OUT. During the transition period, the IAE for PIDPLUS is 190, while that for PID2 is 196.

Again, we test the two blocks by introducing uncontrolled disturbance to the processes. The DISTURBANCE is changed from 20 to 30. The results are shown in Fig. 16.16.

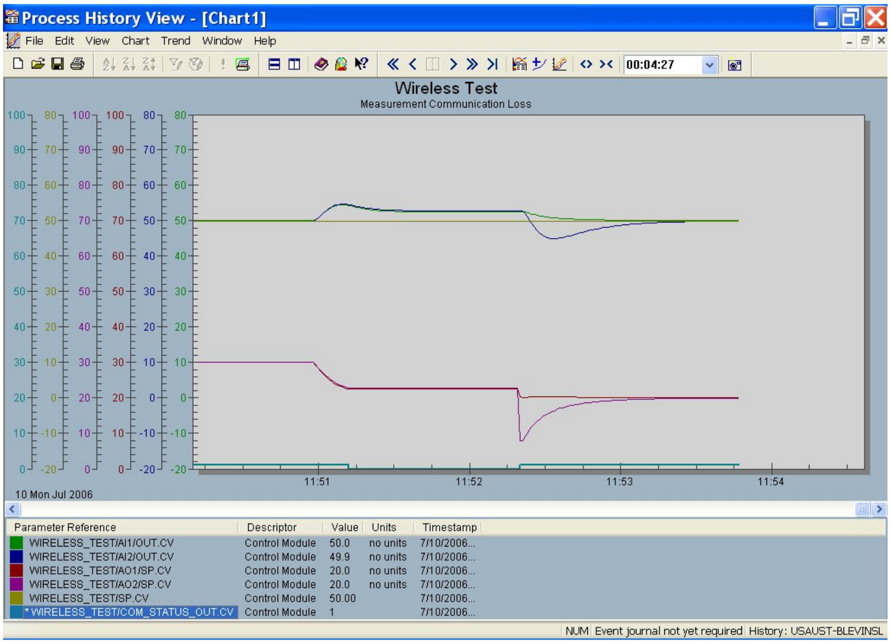

Process History View - [Chart1]
File Edit View Chart Trend Window Help
00:06:40
Wireless Test
Measurement Communication Loss
11:42
11:43
11:44
11:45
11:46
11:47
11:48
10 Mon Jul 2006

Parameter Reference	Descriptor	Value	Units	Timestamp
🟩 WIRELESS_TEST/AI1/OUT.CV	Control Module	50.0	no units	7/10/2006...
🟦 WIRELESS_TEST/AI2/OUT.CV	Control Module	50.0	no units	7/10/2006...
🟥 WIRELESS_TEST/AO1/SP.CV	Control Module	30.0	no units	7/10/2006...
🟪 WIRELESS_TEST/AO2/SP.CV	Control Module	30.0	no units	7/10/2006...
🟨 WIRELESS_TEST/SP.CV	Control Module	50.00		7/10/2006...
🟦 *WIRELESS_TEST/COM_STATUS_OUT.CV	Control Module	1		7/10/2006...

NUM | Event journal not yet required | History: USAUST-BLEVINSL

Fig. 16.15 Missed Outputs with a setpoint change.

The curves in Fig. 16.16 look similar to their counterparts in Fig. 16.15 and they can be explained the same way as above. We also notice that the improvement of PIDPLUS over PID2 is more pronounced in Fig. 16.16 than in Fig. 16.15. This is because at the time of communication reestablishment, the input errors to the PID blocks are bigger in Fig. 16.16 than in Fig. 16.15. During the transition period (158 seconds) in Fig. 16.16, the IAE for PIDPLUS is 267, while that for PID2 is 388.


Process History View - [Chart1]
File Edit View Chart Trend Window Help
00:04:27
Wireless Test
Measurement Communication Loss
11:51
11:52
11:53
11:54
10 Mon Jul 2006
NUM
Event journal not yet required
History: USAUST-BLEVINSL

	Parameter Reference	Descriptor	Value	Units	Timestamp
🟩	WIRELESS_TEST/AI1/OUT.CV	Control Module	50.0	no units	7/10/2006...
🟦	WIRELESS_TEST/AI2/OUT.CV	Control Module	49.9	no units	7/10/2006...
🟥	WIRELESS_TEST/AO1/SP.CV	Control Module	20.0	no units	7/10/2006...
🟪	WIRELESS_TEST/AO2/SP.CV	Control Module	20.0	no units	7/10/2006...
🟨	WIRELESS_TEST/SP.CV	Control Module	50.00		7/10/2006...
🟦	* WIRELESS_TEST/COM_STATUS_OUT.CV	Control Module	1		7/10/2006...

Fig. 16.16 Missed Outputs with unmeasured disturbances.

16.2.5 Active Traffic Reduction to Increase Battery Life

We have discussed extending PID to deal with unreliable communications. This could happen with a wireless network. The other side of the coin with wireless network is the need to reduce traffic to save battery power. For this we could actively stop transmitting data as long as control performance is no affected (Chen et al. 2005). We could stop transmission if the value does not change or changes little. It could be perceived as lost data by the enhanced PID, but we know the value does not have big change compared with truly lost communication when process value might change a lot.

To reduce transmitter power consumption, it is desirable to minimize how often a measurement value is communicated. However, to avoid the restrictions of synchronizing the measurement value with the control, most multi-loop controller are designed to over-sample the measurement by a factor of 2 to 10 times as illustrated in Fig. 16.3.

By synchronizing measure and control execution, as done in fieldbus devices based on Foundation Fieldbus, then it is possible to eliminate the need to over sample the measurement. However, if the traditional approach is taken in scheduling control 2 to 10 times faster than the process response, then the power consumption associated with the transmission of the measurement value may be excessive for all but the slowest types of process. Slowing down the control execution to reduce the power consumption associated with communication may increase control variability when the process is characterized by frequent unmeasured disturbances.

Ideally the power consumption could be minimized by only transmitting the measurement value only as often as required to allow control action to correct for unmeasured disturbances or changes in operation point. For example, one approach to minimize the power consumed in communicating new measurement values is to design the transmitter and wireless communication according to the following Rules for Transmitting a New Measurement Value:

1. The transmitter will periodically sample the measurement 4-10x faster than the process response time.
2. If the magnitude of the difference between the new measurement value and the last communicated measurement value is greater that a specified resolution or if the time since the last communication exceeds a refresh time then the new value will be communicated.

For the lag process used for this example, the number of communications during the duration of the test was reduced by over 96 % when the rules for wireless communication as outlined above are followed. The impact of non-periodic measurement updates on control performance is minimized through the use of the modified PI algorithm for wireless communication. The difference in control performance is shown below in terms of IAE for periodic measurement update vs. non-periodic.

Table 16.1 Control Performance Difference.

Communications/Control	Number of Communications	IAE
Periodic/standard PI controller	692	123
Update Using communication Rules/ PI controller for Wireless	25	159

The power that must be supplied by the transmitter for data transmission can be significantly reduced when the communication rules and the PID controller modifications are used with wireless transmitters. This reduction in power requirement increases the number of control applications that may be addressed using wireless transmitters.

16.2.6 Comments

Using actuator feedback for smooth output transition is not a new idea. The Foundation Fieldbus standard already defines back-calculate-in/out links to smoothen transition during start up. Certain control systems continue to use this link after startup.

PID blocks in modern control systems have status flags for data values. In this way a data value could be tagged with communication lost status. Some systems provide fail-safe mechanism by making use of this flag. For example, Foundation Fieldbus standard allows a limited number of communication failures, after which error is declared and the block enters failure state. This in turn forces the actual block mode to manual during communication failure. This approach alleviates the problem associated with communication loss but does not eliminate them. The proposed modifications to the PID to compensate for communication loss differs in that we explicitly address communication failures and take advantage of the related information.

Current control designs assume periodic samplings. However, this assumption does not hold in a wireless environment. The enhanced algorithm acts in the same manner as the standard PID algorithm when the communication is reliable. When it detects any communication lost, this algorithm can smoothen possible spikes in the output.

Chapter 17 Research in Real-Time Wireless Mesh Networks

Abstract The network community has done a lot of research work on wireless mesh networks. Many of them are theoretical. They are validated through simulations or in limited test environments. Wireless mesh in the industrial settings, especially in a process plant, is getting more and more interests. Different from mobile ad hoc meshes in offices or homes, a wireless mesh in a process plant must satisfy real-time requirements, sometimes even hard real-time requirements in which missing a deadline may have catastrophic consequences. Real-time or embedded applications are considered far more difficult than PC based applications. Likewise, real-time wireless network is also very difficult. The time is ripe for the research into industrial wireless mesh networks. Many topics throughout this book are worth studying by the academia. This chapter will re-emphasize some of them. Wireless technology adds value to process control industry. In turn, having the process control industry embrace wireless technology accelerates wireless technology development. The process industrial setting provides a real world test environment for any theories and ideas developed over the past decade by the network research community.

17.1 Real-Time Systems

A real-time application is characterized not by its speed, but by deterministic behavior. Normally process control loops run with the period of a fraction of a second or seconds. The key requirement is that a complete control algorithm is executed within each period. A fast but late response is worse than a slow one who meets the deadline. The speed of a discrete manufacturing control system may run with microsecond cycles. A process control system is slow compared to it, but the real-time issues in a process control system are no less important or complex.

A real-time system is normally continuously performing the same job repetitively; each recurrence must be finished in time. This is well captured with the task models in the real-time research community. A real-time task T is a 3-tuple $\{C, D, P\}$, where C is the execution time, D is the relative deadline, and P is the period. At the beginning of each period P, the task T requests an execution of length C that should be finished within D time units. Each request is called a job. Normally $D <= P$. The task T fails if any of its jobs misses its deadline. A real-time task set S is a set of n real-time tasks $T_1, T_2, ..., T_n$. A task set S is schedulable by a scheduling policy on a single processor if no job of any task will miss dead-

D. Chen et al., *WirelessHART™: Real-Time Mesh Network for Industrial Automation*, 223

line under the control of this policy. *S* is feasible if *S* is schedulable by at least one scheduling policy. There are also other real-time task models. For example, a task may have fixed initial start time. In some simple versions a task's deadline equals its period. For continuous real-time applications, a task is usually repetitive, hence the period in the task definition. *P* is sometimes also called the minimum separation time to model tasks whose jobs are not exactly periodic. Many schedulability results apply even if *P* is the minimum separation time.

Research on real-time task scheduling on a single processor is considered mature now (Chen et al. 1997, Chen 1999, Chen et al. 2003, Kuo and Mok 1991, Liu and Layland 1973, Sha et al. 1990, Sha et al. 1994, Stankovic et al. 1998). Many well known scheduling policies, such as Earliest-Deadline-First (EDF), Rate-Monotonic-Algorithm (RMA), and Priority-Ceiling-Protocol (PCP), have already found their ways in real industrial applications. There has also been works done on multi-processor systems in which a task could be selected to run on any processor in the system.

17.2 Selected Topics

Security

It is said that security aspect could be a big show stopper for wireless in the process industry. Because of this WirelessHART incorporates an extensive set of security features that are described in Chapter 8. To some extent this is not fair as nowadays security could be a show stopper for any process industrial applications. Any security concerns could find its wired counterpart. The only difference is the cost and level of severity. When we address the security issues, we should address the security about the whole process plant in which wireless communication is just a component. Two best defenses of the wired control system are its proprietary nature and total disconnection from the Internet. Being proprietary makes a system less penetrateable than standard ones; and physical disconnection avoids any online hackings.

Admittedly, wireless makes security research more urgent and more interesting. For example, how to implement a public-private key with limited memory size, using elliptic encryption algorithm? How to perform a complex encryption algorithm with limited process power?

End-to-End Delay

A real-time application running on a distributed network could be modeled as one real-time task set per node plus real-time data communications among the nodes. The real-time data communication between a source and destination could be considered as one real-time task in the source and one task in the destination, plus the synchronization requirement. Each communication task has an execution time, a relative deadline, and a period. The execution time is the transmission time of the communication. The destination must be in the listening mode when the source is transmitting. Fortunately, there are always ways to schedule the task set once we divide the problem into individual scheduling problems in each node. Since mainstream distribute networks such as Internet adopt best-effort data transmission mechanisms, there are few, if any, real-time applications on distributed networks. It is hard to provide guaranteed data delivery on best effort networks.

Another challenge lies in the case when the source and the destination are not direct neighbors. In this case, all nodes on the path from the source to the destination should deliver the data to the next hop timely so that the total delay is no more than the relative deadline of the communication. Since each node on the path acts independent of one other, it is hard for the nodes down the path to dynamically adjust to the accumulated delay from earlier nodes. An easier way is to pre-configure each node's delay contribution. One possible solution is to pre-assign individual delays on each node. For example, in the RSVP protocol, the bandwidth and delay request is passed from the source to the destination. Each intermediate node returns with the range of commitment it could provide. Then the final allotment on each node is calculated and assigned. To enable real-time applications, some sort of centralized control is necessary.

There have been extensive research works on finding data paths that meet QoS requirements such as delay bounds. Those results could be applied to either centralized or distributed networks. For example, in different heuristic Delay-Constrained Least-Cost (DCLC) unicast routing algorithms the path from source to node is determined by breadth-first or depth-first search on the network graph. DCLC algorithms would be very costly if it is carried out online. For periodic real-time applications, we could use DCLC algorithms offline and apply the result online during application execution.

Reliability

Reliability issues in wireless networks are more prominent than in wired networks. The WirelessHART standard employs various ways to mitigate the problem. Please refer to Section 10.9. More interesting problems will emerge with the wide adoption of wireless technology in the process plant.

Failure Semantics

Recovery from faults is also an interesting research topic now that failure is inevitable. We need to define the failure semantics in the wireless mesh that provides a theoretical framework for peacefully living with errors.

Statistics

There are many uncertainties in wireless mesh networks and statistics approach to handle this is actively studied in the research community. The process control applications add another real-time dimension to the equation, how to achieve guarantees out of uncertainties.

Mesh in mesh

There has been work on large scale wireless sensor networks. Mesh in mesh, i.e., clustering a set of small size mesh networks is a natural approach in a process plant. The WirelessHART network works best for fixed assets and Wi-Fi mesh works best for mobile workers. We could form WirelessHART gateways into a Wi-Fi mesh network, which takes advantage of the best of both worlds. Researches on this kind of hierarchical topology benefit the industry directly.

Battery Consideration

There are two research fields for battery usage in a wireless sensor, one is the battery design itself, and the other is how to maximize battery usage.

Chapter 18 Future of Wireless and the WirelessHART Standard

Abstract This chapter consists of four sections. In section 18.1 we discuss areas where a wireless application has advantages and where wireless applications could be used to replace existing wired applications. We also list new applications that have not been addressed, or have been poorly addressed, with wired applications. In section 18.2 we discuss location awareness applications and techniques. Location tracking will be a major component of wireless plant applications in the future. A location-determination application in the WirelessHART standard is also described. Section 18.3 talks about cyberphysics for which wireless sensor network could be the immediate interface to the physical world. And the last Section 18.4 reveals the future directions of wireless and offers some thoughts on how the WirelessHART standard will evolve to meet these directions.

18.1 Wireless Sensor Network in Process Automation

Of tens of millions of HART devices in the field, only a small portion of them are digitally connected to the host to provide supporting data besides process data. With the WirelessHART standard, users can add wireless technology to existing installations and new builds. The wireless solution offers flexibility, scalability, and interoperability between field devices, control systems, and asset management packages. It connects small sensors through low power, low data rate wireless communication. With the vast amount of new environmental data, we are able to explore unlimited potential for enhancing the quality of process control. Users are given a clear path to add wireless connectivity to their operations, and are able to harness the information embedded in intelligent field devices to improve productivity.

Looking forward, wireless could be used as the replacement for wired counterpart in the plant; it could also be used in brand new applications. Next we discuss areas where a wireless application helps and wireless products in general.

18.1.1 Wireless Mesh Applications

Asset

Equipment failures and the associated maintenance cost time, money, and reduce plant throughput and availability. Small problems, if undetected, can escalate into big problems causing significant damages, turning a small maintenance task into a major repair, reducing asset life or requiring replacement. With wireless mesh, asset health could be monitored constantly. Small problem could be found and eliminated early, extending the life of assets.

Efficiency

Increasing plant efficiency reduces material costs and can increase plant throughput. Small increases in efficiency can bring a substantial improvement in plant profits. Smart wireless solutions can cost-effectively enable additional measurements needed to optimize plant efficiency. These include points that were not possible to monitor in the past, such as points on moving or rotating equipment, or points in harsh environmental conditions.

Maintenance

Maintaining plant assets is expansive. Most plants still rely on preventative or reactive maintenance practices even though these practices reduce the availability of plant assets. Smart wireless solutions can help move to predictive and proactive maintenance practices.

Safety

Smart wireless solutions could allow wireless monitoring of safety equipment such as eyewash stations so that action can be taken sooner, should an incident occur. We could also wirelessly monitor readings in hazardous areas of the plant, reducing personnel safety risks.

Environment

Wireless technology can minimize the hazard, clean-up, and expense of environmental releases by providing fast notification and accurate logging of an environmental excursion, should it occur.

18.1.2 Wireless Products

Let's look at the path that the process data travels from the device all the way to the end user. Wireless products for process control exist from the edge of the devices to the data-receiving workstations. Field devices could have built-in wireless transmitters, or connected by wire to a wireless transmission node, which may talk wirelessly to a controller/workstation, or to another wireless node connected by wire to the controller/workstation. Wireless products could be used to communicate among controllers and workstations. Yet more wireless products could be used to communicate from a control system to the outside world.

The wireless products at the control unit level should serve short distance with high reliability. They have to endure harsh environment. They may have low power requirement. The wireless products at the control area network need to serve high data rate and long distance. It still requires sufficient reliability, although not as high as in the control unit. At this level, security concern starts showing up as the airwave spans a much larger area. Beyond the control system, we assume any conventional wireless products could be used to transmit data to and from the control workstations. Of course, any concern with conventional products maybe an even bigger concern here as we are talking about connected process control system.

Hardware products could be standalone wireless transmitters. They could also be devices and controllers equipped with wireless transmitters. Companies could also sell wireless parts such as antennas and accessories. Software products could be communication stacks, device drivers, and complete systems. Companies could also provide complete solutions or help customers setting up a wireless system.

For customers, the main selecting criteria would be if the product suits the need. Others are product cost, development cost, power consumption, transmission distance, communication reliability (mesh vs. peer to peer), bandwidth, business maturity, project risk, network protocol, handheld support, large system deployment, interoperability, installation tool support, troubleshooting tool support, international support, etc.

One question needs to be answered in the future is how a wireless product meets the hard real-time requirement of a process control system, especially at the control network level

Wireless Accessories

These include hardware parts such as antennae, cables, accessories, PC-based wireless cards, remote sensor interface, etc.

Wireless Devices

These will be the most common wireless products on the market.

Gateway

Gateway acts as the intermediaries between the devices and the outside. There are two kinds of gateways. One kind connects to the devices by wire using existing wired network protocols and in turn communicates wirelessly to the host or another intermediate gateway connected to the host. The transmission distance ranges from hundreds of feet to miles. The other kind is part of a wireless mesh network and in turn communicates by wire or wirelessly to the host. WirelesSHART gateway is the latter.

The gateway could be as simple as a modem that translates data from one protocol to another. It could also be a data access point that combines several devices together and acts as a single wireless source. More sophisticated gateways are called wireless bridges, node modules, or remote terminals. They run software, could be configured, and process data. They provide product designer with a transparent and easy to design wireless communication link.

Wireless Mesh

The trend is standardized mesh networks such as WirelessHART meshes.

Wireless Control System

This will be the ultimate wireless product for process control.

Wireless Service

These include radio frequency site survey, installation, start-up and commissioning, and training. Others are sensors and data collection tools for diagnostics, condition-based maintenance, asset management, situation awareness, operational readiness, and security.

18.2 Location Awareness

Location awareness deserves special attention because it is important for plant personnel tracking. When incident occurs the promptness of locating and rescuing people makes a lot of difference. With wireless mesh deployed throughout the plant, a person could carry a handheld or a tag that interacts with the mesh devices which functions to track the handheld, hence the person.

We believe location awareness will be a major component of wireless plant applications in the future.

18.2.1 Location Awareness Techniques

Localization problem is considered as one of top 10 networking problems. While GPS is widely used for such purpose, it is costly and does not work indoor. Hence, non-GPS location awareness has recently received a great deal of attention.

18.2.1.1 Theoretical Principles of Localization

There are two localization schemes: range-based localization and range-free localization. Since the latter cares more about the relationship between nodes rather than the accurate location, it won't be discussed here.

In general, a positioning system consists of two components: reference points, whose coordinates are known; the other is the relationship between the sensors and the reference points. Those reference points are often called beacons or anchor nodes. The relationship between the sensors and the reference points can be studied with three techniques:

RSSI

RSSI (Received Signal Strength Indication) makes use of signal strength decaying models to estimate the distance between the sender and receiver. The problem of RSSI is when noise or obstruction is present, its accuracy will be affected. It is pointed out that the accuracy of RSSI is expected to be about 10 feet under indoor environment.

TDOA

TDOA (Time Difference of Arrival) makes use of the signal propagation speed which is more robust than the signal attenuation feature used by RSSI. However, since radio signal processing is too fast for most current crystal timer chips, acoustic devices are instead employed in many localization systems. Also, TDOA requires time synchronization between the sender and receiver which is very demanding in some cases.

AOA

AOA (Angle of Arrival) is a method for determining the direction of propagation of a radio-frequency wave incident on an antenna array. This approach requires more than one antenna and computes the direction based on the time differences between antennae.

18.2.1.2 Properties of Location Techniques

Physical Position and Symbolic Location Information

Physical position refers to global position, like 47°39’17” S. Symbolic location information encompasses abstract ideas of where something is, like office 5.302.

Absolute versus Relative Locations

Absolution location system uses a shared reference grid for all located objects, while in relative system each node can have its own frame of reference.

Localized Location Computation Capability (LLC)

Some systems require that a node should compute its own location while other systems utilize a central server to assign coordinates to nodes.

Accuracy and Precision

It is defined as the locating accuracy or the error ratio between estimated location and accrual position.

Scale

Scalability is concerned with the coverage. While a wireless mesh may be small, localization techniques should be able to cross meshes.

Cost

The cost could be estimates as the ratio of anchor nodes. Computational complexity of a technique also affects the cost.

18.2.1.3 Localization Techniques

Some of the techniques developed by the research community:

- Utilizing interesting environmental features to locate the nodes. It abstracts temperature, humidity, ambient noise, spectrum energy, received signal strength to recognize the location. (Chen et al. 2007)
- Making full use of connectivity information. Since connectivity implies both connected (positive) and non-connected (negative), it can divide the space into two. Using these constraints, a possible area for a node can be determined. (Guha et al. 2005)
- Locating mobile nodes. There are several mobile patterns and we could apply sequential Monte Carlo localization method. Mobility can improve the accuracy and reduce the costs of localization. (Hu et al. 2004)
- An iterative method to minimize the least-square errors of localizing. Iterative algorithms are popular in localization. Since some statistical models do not have analytical solution, iterative programming becomes the only one way to solve it. However, iterative algorithms are more complex than linear programming. It may not suitable for sensors with low capability. (Liu et al. 2006)
- Adopting an iterative algorithm for how to find a network deployment for a set of nodes. Since such problem is NP-hard, we could use approximation: weak deployment and strong deployment. The two deployments give out different upper and lower bounds on the uncertainty of localizing nodes. (Basu et al. 2006)
- Using mobile nodes as beacon nodes. The receiver needs not time synchronization with the sender (beacon node). By using TDOA, it can locate the node accurately. (Luo et al. 2006)
- Utilizing prior location knowledge of sensor groups and developing a statistical model to estimate node location. Sometimes the location of sensors can be got before deploying. However, sensors might not be deployed at the designed place but sensors can contact its neighbors to estimate the location by probability methods. (Fang et al. 2005)

18.2.2 A WirelessHART Location-Determination Application

In this section a location-aware application on WirelessHART network is presented (Zhu et al. 2009). It is completely software based and needs no modifications of field devices. Thus, it is applicable to all WirelessHART networks. The key ideas behind locating mobile users are based on the following two techniques: the comparison of two-way received signal strengths and choosing best parameters of radio propagation model that uses training data collected beforehand. In this approach, the first step is to select a radio propagation model and then collect enough training data to train the model. Then, during the localization procedure, both the mobile user, i.e., the handheld device or badge, and the field devices send *neighbor heath report* to the network manager. These neighbor health reports contain both device ID information and receive signal strength levels. As the worker moves around the plant, the mobile device will detect field devices around them. Similarly the field devices will be able to detect the presence of the worker. Frequently, there are more than three devices in close proximity to the worker. As the devices and the worker's handheld or badge transfer information to the network manager, the network manager in turn makes use of this information to filter out the untrustworthy received signal strength indications. The easiest way to do this is to compare the health report from a field device with that from the handheld device. If the two match, both can serve as good measures of distance. If the difference is bigger than a certain threshold, the pair will not be considered. Since the network manager knows the deployment of all the field devices, it is easy for the network manager to locate the node accurately by using well-trained localization model.

Although WirelessHART defines many kinds of services at the application layer, it currently does not specify how to provide location awareness support. As such, there is no way to locate the devices or workers by issuing the standard commands defined by the WirelessHART protocol. While it is possible to add device-specific hardware and commands to the devices, we choose not to add extra gears and acoustic features. The only distance indication that we shall rely on is the received signal strength information provided by the physical layer. In theory, a handheld device or badge can compute its location completely by itself through trilateration as field devices are usually extensively deployed and the handheld device can sense many devices around it. However, such solution has several problems in WirelessHART network. Firstly, for security concerns, no two devices in a WirelessHART network can easily talk directly to each other. So, even if the handheld device knows the distance from a field device by sensing the signal strength, it cannot directly inquire about the location of the field device. Of course, the handheld device may request the location information from the network manager. However, the WirelessHART standard has not defined any command for location information and as stated earlier, we do not want to modify the devices in our solution. Secondly, the handheld device may receive more than enough distance indications. However, such redundancy may not help to improve the accuracy and may even cause confusion because the handheld or badge does not know which distance indication is untrustworthy. The uncertainty in signal strength surely must filter out the *bad* indications. The handheld will need to select the neighbor devices to use in its calculations from a list of possible candidates. Our solutions meet these challenges as follows:

- In order to improve the precision of localization, we need to derive accurate parameters for radio propagation model. In our approach, a survey is taken to collect enough training data in the first step. Then, with the training data, we calculate the best parameters by using the least-square method.

- The WirelesHART standard requires every device (including the handheld device) to send out keep alive messages to its neighbors periodically (normally 30 seconds; however, it can be set by the network manager) and the receivers should send acknowledgements back. Through such bidirectional communications, each device can get the signal strength of the other end.
- Locating a handheld device or badge is done by the network manager. Since neighbor health reports are sent periodically (15 minutes by default, also, it can be set by the network manger) to the network manager, the network manager can collect enough information for locating the handheld device. In fact, the network manager can request the neighbor health report actively. As stated before, redundancy will not confuse the network manager. This is because while a field device can report the signal strength from the handheld device, the handheld device can do the same independently. Thus, the network manager can match the two reports. If the difference between two signal strength values is bigger than a certain threshold, the pair of reports will be discarded. Otherwise, the network manager will average the two and put it into trained propagation model. In this way, we can filter out some *bad* indications caused by random noise.

18.3 Cyber-Physical Systems and WirelessHART Systems

The application of wireless technology to process automation is a significant advance in the long march of industrialization from the invention of the steam engine to today's Internet-powered enterprises. Yet, we have only begun to see the dawn of a new age of technological advances that will change society in profound ways. The success of the Internet is founded on its ability to disseminate and integrate information on a global scale. The Internet enables the *mashing* of various information services such that new and more powerful services can be delivered to the consumers. In the future, the services that will be mashed will include not only information services but physical services that require advanced process control technologies. One such example is the emergence of tele-medicine including tele-surgery whereby surgeons will perform surgical procedures on patients from afar. Another example is tele-home-assistance whereby intelligent haptic devices will be operated from afar to perform household chores for the elderly and the infirm. In the intelligent highway system of the future, cars equipped with automated driving assistants will be able to avoid accidents by modulating the human driver's behavior in hazardous situations. The term Cyber-Physical System (CPS) has been coined to draw focus to the new types of technologies and services that are enabled by the integration of information and physical control technologies. CPSs are physical and engineered systems whose operations are monitored, coordinated, controlled and integrated by a computing and communication core (Stankovic et al. 2005).

Wireless process automation is an essential component in the technological development of cyber-physical systems. The WirelessHART standard is well suited for implementing many CPSs inasmuch as the WirelessHART standard is a technology for implementing control applications. Unlike business applications, control applications have far more stringent requirements on the determinism of response, tolerance for noisy environments and security against deliberate attacks. Real-time control systems must have highly predictable response times by design and this cannot in general be guaranteed by non-deterministic protocols such as Wi-Fi or Bluetooth. Because the WirelessHART standard has been designed to operate in a noisy environment such as factories running heavy machineries, it is robust against data corruption in communication channels. The WirelessHART standard has also been designed to withstand certain types of deliberate attacks. However, the development of cyber-physical systems presents new challenges because CPSs are different from the traditional information systems (IS) and physical control systems (PCS) in that there are interactions between the information processing elements and the physical control elements that neither IS nor PCS alone may entail. In computational terms, CPS presents new failure modes that have not been anticipated by the designer of control systems or information processing systems. An attacker who understands these interactions may be able to exploit the new failure modes to cause major system upsets. In particular, the use of wireless technology as the infrastructure for communication and control may open up unforeseen gaps for attackers to mount unconventional attacks. New models and techniques must be developed to counter these attacks. In the following, we shall briefly survey the security landscape that has implications on the deployment of wireless technology such as the WirelessHART standard in the process automation industry. By doing so, we hope to raise the awareness of both academic and industry researchers and practitioners for information security as we deploy the new technology.

We can think of the attacks as strategies to disrupt the coordination between the physical state information flow and the command and control information flow of the CPS. For example, in the Smart Grid of the future, reliable phase measurements of the power generators are critical for the safe and efficient control of the power grid. An attacker of the Smart Grid may insert itself in the information flow infrastructure of the control network and thereby able to create havoc. This can be done by, for example, jamming the wireless control signals at crucial moments and at different points of the control network, or worse still, an attacker that has succeeded in impersonating the identity of a transmitter can mount Byzantine attacks on the Smart Grid (Byzantine attacks are information warfare attacks where the attacker may take over some of the control computers in a communication network and pass misleading information to other computers by impersonating the healthy computers). The resulting erroneous phase measurements could cause serious damage if appropriate counter measures are not taken. While there has been significant work in applying defenses against Byzantine attacks in the literature, all the solutions must depend on the assumption that not all the nodes in the network are compromised simultaneously and that at least some fraction of the nodes (above a lower bound) must function correctly in order to defend against the attack. As such, there is no defense that we are aware of that is effective against common-mode attacks inasmuch an attacker that can exploit a common vulnerability to compromise every node that can potentially mount an attack to simultaneously compromise all the nodes and disable all the control network elements. This can happen if for example, a bad version of the control software has been downloaded into every node of the system waiting to become activated on command of the attacker. This type of attack can in theory overcome all known defenses against Byzantine attacks.

On the other hand, there are characteristics of CPSs that are advantageous to the defender. Some of these characteristics are:

1. The dynamics of the physical system under control is determined by the laws of physics that no attacker can alter.
2. The rates of change of the physical state variables of the process under control may be slow in comparison to the rate of information flow in the control system, and this is usually the case with mechanical and chemical systems.
3. The value of information in a CPS is usually a function of time and may degrade substantially with the passage of time.

An implication of (1) is that the defender has a source of information that of itself is not subject to subversion by a human adversary through cyber attacks alone, although the information may be compromised during transmission and transformation. If we can devise security checks that use only information directly from the physical process measurements, then such checks cannot be subverted by an adversary. These checks can be the foundation of the secure computing base of the system.

An implication of (2) is the possibility for the defender to maintain through measurements and real-time simulation techniques an approximate but sufficiently accurate global view of the physical state of the system or at least be able to know within bounded time if sufficient accuracy has been lost, so that the defender can know that the state information cannot be trusted. This is analogous to the late Flaviu Cristian's approach in system design vis-a-vis his timed asynchronous model of computation (Cristian and Fetzer 1999). The issue is how we can effectively leverage the relatively slower rate of change in the physical state to increase the probability of attack detection. We think that this is possible unless the attacker can corrupt physical state measurements on a global scale and in a synchronized fashion. In any case, it should be possible to design detection systems that build on local measurements in a hierarchical fashion and on different time scales that would make it extremely difficult for an attacker to synchronize an attack.

An implication of (3) is that the attacker has limited time to exploit the information s/he has obtained about the physical system state. Therefore the protection of state variable information does not have to be permanent. Specifically, it is only necessary for the defender to deny the attacker the information s/he seeks for as long as the information has time value. This should open up new ways to look at the design of information protection methods and architecture.

We believe that the crucial point in mounting an effective defense is in being able to correlate between global control information and local physical state measurements in time, and in switching the defensive strategy faster than the attacker can incur unacceptable physical damage to the CPS. This assumes that certain real-time constraints can be monitored either by the information flow infrastructure itself or by correlation to physical state measurements. Contrary to common wisdom, we do not regard hard timing constraints as an vulnerability in the design of secure systems. We take the view that hard timing constraints define the integrity of a system so that failure to meet a hard timing constraint means that system integrity has been breached, but any single timing failure should not result in total system failure. This approach to design secure systems is akin to the late Flaviu Cristian's *timed asynchronous system* design philosophy. The bedrock of this approach is that time synchronization failure is detectable. In the case of CPS, the physical measurements performed by locally controlled physical devices can be hardened, say by hardware means, to detect timing failures as exhibited by any unexpected behavior of the physical processes under control as a function of time. A successful attacker must now coordinate the timing of the attack on the physical measurement system in order to mask the attack. This race against time turns the timing constraint requirements into an advantage for the defender inasmuch as the physics of the processes under control help define what is normal behavior against which anomalous system behavior can be compared. This gives us hope that we can design low-overhead intrusion detection and prevention systems that can defend against new classes of denial-of-service attacks such as allergy attacks (Chung and Mok 2007) which would otherwise have been very difficult to defend against.

The other half of an effective defense against attacks is the design of recovery mechanisms, given that some attacks will succeed. Central to the notion of recovery is the idea of failure semantics. Here the critical insight is that a system should be designed to function according to specification not only under normal conditions but that the system should also function with high probability in some specific way when a failure occurs. For CPS systems, the traditional failure mode classification is not satisfactory inasmuch as the classification was devised with computer (hardware) systems in mind. For CPS systems, we need a classification system that is more specific with respect to the interaction between the physical components of the system under control and the computer and communications components that control them. For example, we should be able to take advantage of the fact that mechanical components usually fail on a time scale in seconds) that is large compared with the reaction time of computer systems (much less than a second) to define a CPS-specific failure classification by taking into account the amount of time that is available for defensive actions. We may for example add a time dimension to the failure mode classification by including explicitly a time-to-failure parameter. There are also other dimensions that a CPS-specific failure mode classification scheme can exploit by focusing on the application domain specifics. For example, when a traffic light fails, we do not stop all traffic but instead we put the lights in a blinking-red mode. This signals to all the cars entering the intersection to follow a pre-agreed protocol, namely, the intersection should be regarded as being regulated by all-way stop signs. The blinking red lights thus impose a specific type of behavior on all traffic. This suggests a way to generalize failure mode semantics: we can define failure semantics in terms of protocols that the failed system must follow. These protocols can be formalized by the plethora of techniques from computer science, hybrid systems and other branches of engineering. In other words, we can define failure modes by the protocols that a failed system must follow and these protocols can be application-specific as well as have a time dimension.

No defense is perfect in that the effectiveness of a defense is necessarily relative to the assumptions made about the damage that can be caused by an attacker. In CPS systems, the damage to the system can be quantified by the behavioral deviation from a healthy system. Since the speed at which an effective attack on a CPS system can be mounted is also limited by the physics of the system under control, this allows the defense to define and enforce application-specific failure semantics that will aid in the design of secure systems.

We believe that secure CPS design can benefit from the assumption that system failures are detectable by local physical measurements and that detection of security attacks can be predicated on the detection of such failures. Given this assumption, the defense against security attacks becomes a discipline of designing systems with well chosen physical failure semantics and the timely detection of such failures. What are required are design principles for specifying physically induced timing requirements and the tools for monitoring these constraints (Woo and Mok 2007). The monitoring facility should be incorporated into the secure computing base of any CPS system.

The security problem can only be solved by the close collaboration between the designers and implementers of process control protocols and the end-user community. Toward this end, there is considerable research done at universities and industry to strengthen WirelessHART networks against deliberate information attacks through rigorous testing and advanced design ideas (Song et al. 2008). However, there is still a lot of work that needs to be done including:

1. Collaboration by between academia with the utility industry to understand and exploit the domain knowledge that is required to formulate security checks directly from the sensors to provide a first line of defense against cyber attacks.

2. Creating an information system architecture to exploit the correlation of process and timing information of both the IS and PCS sides of the utility system, including real-time simulation facilities.
3. Engineering domain-specific real-time database services that are resilient to cyber attacks, especially denial-of-service attacks.
4. Understanding quantitatively the tradeoff between the cost of timely detection and the damage that a utility can sustain as a function of time-to-detection and the time to execute counter-measures.
5. Investigating what is appropriate failure semantics that can minimize damage as well as the time to recovery.
6. Exploring information protection methods and architectures that can exploit the fact that the attacker can have only a bounded time window in which to synchronize and mask an attack before the attacker's data becomes stale and loses its value, e.g., by inventing light-weight cryptographic techniques that can guarantee short-term protection for any new data.
7. Understanding and exploiting the technical characteristics of wireless protocols and technologies for process control to facilitate implementing points 1-6 above.

18.4 What's next for the WirelessHART Standard?

It would be easy to make predictions about all of the great things that wireless technologies could do in manufacturing and automation environments. However, we won't do that. The reality is that the WirelessHART standard is simply a continuation of long series of progressions that started over 20 years ago when the HART standard was first released. The HART standard was designed from the beginning to support suppliers and end users working together to address real needs in real plants. During this progression the device infrastructure has evolved to support devices covering a wider range of needs. So the real questions are, "What are the business drivers and how will the device infrastructure evolve to support these business drivers?" From this we can easily predict future enhancements. The discussion will start with a brief discussion of business direction followed by directions in device technology. Specific areas that are likely candidates for standardization will then be discussed.

Looking at the first question, "What are the business drivers?" All business performance is based on value that can be generated from its assets. These assets range from people and materials, to intellectual content to physical properties. Plants are becoming much more integrated with business systems. These plants operate too much tighter requirements, are expected to be able to adjust production schedules in real-time to changes in conditions and orders, and are much more regulated. Achieving these objectives requires a much greater understanding of the process, improved understanding of the state of the equipment in the plant, and far better data analysis techniques. The people operating these plants will likely hold degrees, and in many cases advanced degrees. This leads to the second questions, "How will the device infrastructure evolve to support these business drivers?"

The answer to this second question must be considered in several parts. Gaining process insight involves increased measurements, providing more diagnostics on the devices providing the measurements, providing diagnostics on the process that the devices part of, and moving things that were in the past done manually on-line. The first release of the WirelessHART standard went a long way towards making it possible to both reach advanced measurements and diagnostics that are already in devices today and to cost effective measure many things in the past that were difficult to reach. In the first case many plant infrastructures today are ill-equipped to report advanced diagnostics. Wireless allows these measurements to be communicated on an alternative infrastructure. In other cases the type of equipment, for example rotating equipment, made it difficult to take measurements. It is a lot easier to attach devices to this kind of equipment and let the wireless infrastructure take care of the communications. In still other cases where state-of-the-art was manual measurement, wireless makes it cost effective to periodically take these measurements and communicate them. An example of this is equipment health and monitoring. New devices are being designed and built to measure vibration and communicate signals values and diagnostics back on-line centralized systems.

New devices often include advanced diagnostics that can diagnose the health of the device and in many cases, the health of the process that the device is connected to. It is not uncommon for these latest devices to include diagnostics that can detect plugged lines, burner flame instability, agitator loss, wet gas, orifice wear, leaks, and cavitations. These devices tell the user how well they are operating and when they need maintenance.

To realize the benefits of these smart devices there needs to be very good integration with applications and control systems. To facilitate this integration HART incorporated features such as status on measured values, time stamps, event latching and confirmation, block data transfer, and in the case of HART 7, a complete wireless communication system that tunes itself to match the control and communication demands of the control system. Better integration results in fewer trips into the plant, a much better controlled process, and significantly less downtime.

Many new devices include multiple measurements, for example a level measurement device may include both a flow measurement as well as discrete values providing high and low level indications. These hybrid devices are supported by a new discrete specification. Other new devices will offer asset tracking capabilities. These requirements will be addressed using location measurements and techniques.

An area of WirelessHART technology that was incorporated in the first release of the specification was the wireless handheld. The wireless handheld was designed to support workers in the plant who wanted direct access to devices and equipment. This area of the specification will be enhanced.

Many of the technologies that the WirelessHART standard is built on will continue to evolve. For example, new radio technologies will be released and new frequency ranges will open up. The standard is not locked into the 16 channels described by 802.15.4 – the standard supports up to 64 channels. The physical layer in the WirelessHART standard can be easily replaced in the future as new radios are released.

Another area that is likely to evolve is how devices are provisioned. In some cases it would be advantageous to be able to provision devices over the area. The original release of the WirelessHART standard considered this and left the door open for public key encryption techniques to be added at a later date.

So what about control over wireless? The truth is that early adopters are already hard at work testing control over wireless. These first installations have already proven that good control performance can be maintained with wireless infrastructure. As improved wireless network scheduling techniques and better battery technologies are developed, control over wireless will be adopted by more and more plants.

18.4.1 Discrete Devices and Values

In most plants in the process industry, discrete measurements and actuation play an important role in the overall plant control systems and operations. In some industry segments, over 80% of the total input-output count in a control system may be discrete. Traditionally, these discrete requirements have been addressed through the use of discrete input-output cards contained in the control system or though PLC's that interface to the process control systems. In many cases these discretes are either part of existing measurement devices that are already in the plant or in support of these devices. For example, level and pressure instruments often include associated limit switches and diagnostics that should ideally be communicated as discrete values.

Discrete values can be represented either as simple discrete values or through more complex state information. In the discrete case which includes limit switches and on/off devices, the value is simple 1 or 0. In more complex cases the state requires multiple values, for example a valve could be represented as a series of states such as open, closed, opening, or closing.

In the simple on/off case values can be communicated as blocks of discrete values or bits. A single status can be associated with a range of values. In the more case, the value holding the state of the on/off valve or other device should be communicated along with its status and time stamp.

This overall expansion to include discrete values and status information represents HART standard's continued evolution as it addresses more measurements and a wider range of plant floor needs.

18.4.2 Location

Location technology can be used to track personnel and assets on the plant floor. Tracking of personnel on the floor is critical in process plants for personal safety because of the high risk involved due to toxic or inflammable chemicals involved. Tracking of assets makes it easier for personnel on the plant floor to locate materials and equipment. The initial WirelessHART specifications already included much of the functionality required to support location applications; for example all devices have a consistent sense of time, must be able to advertise, route messages, track signal strength, publish health reports, and route messages using proxy addresses. With the first release of WirelessHART specifications it was possible to perform off-line location calculations. What is missing is a standardized way for devices in the field providing location information to communicate with location applications that running in a control room or on a handheld device.

To address these requirements a future release of the WirelessHART standard will add standardized commands for location tracking devices to talk to location applications. These commands will make it possible to locate where assets are in the plant. Another enhancement will be adding sessions allowing tracking devices to communicate to location applications. The content of these communications will be fully encrypted and secured.

18.4.3 Handhelds

Handhelds are used in the installation and maintenance of devices. They will also be used to host location applications. In all cases these handhelds must not be allowed to join the wireless network until they have been authenticated. Once authenticated all communications between the handheld and devices on the network must be secured.

During the development of the initial release of the WirelessHART standard there was considerable discussion around limiting the communications of wireless handhelds. This limitation was supported by setting up a session, a route, a superframe, and links between the handheld and the wireless device being connected to. At this point all communications would be restricted to between the handheld and the device.

As part of the location tracking discussion it will be necessary to expand the scope of the handheld. With location applications it will be necessary to use the handheld to locate assets and communicate with control room operators. In these cases a special session will be set up allowing messages to be securely exchanged between the handheld and control room and location applications.

18.4.4 Public/Private Key for Handheld

The initial release of the WirelessHART specifications includes an extensive set of security features designed to ensure that devices cannot join until they are authenticated. Once authenticated all communications are private. One of the features discussed, but not specified, was over-the-air provisioning. To allow for this capability the original design left room in the standard to specify and utilize additional security schemes.

Looking ahead one of the features that will be added is public key encryption. Several technologies are available for this; one of the most promising being considered is Elliptic Curve Diffie-Hellman (ECDH). Before this, or some other technique, can be accepted it is important to understand the technology as well as the resource requirements (memory and CPU) of the implementation. More work is underway.

18.4.5 Control over Wireless

Utilizing wireless as an infrastructure for closed loop control presents many technical challenges for device manufacturers. Most multi-loop controllers are designed to over-sample the measurement by a factor of 2 to 10. Also, to minimize control variation, the typical rule of thumb is that feedback control should be executed 4 to 10 times faster than the process response time, process time constant plus process delay. In wireless where battery life is critical, the goal is to communicate as few times as possible. While this at first appears to be a huge barrier, there is a path forward.

In WirelessHART the approach taken was to allow the devices to sample as often as necessary but communicate only when values change. Using this approach the communication overhead on the network is significantly reduced. In real plant the use of these techniques resulted in communication reductions of 10 to 30 folds. So what about the control side?

To provide best control when a measurement is not updated on periodic basis, the PID may be restructured to reflect the reset contribute for the expected process response since the last measurement update. Some of our work in this direction is presented in Section 16.2.

PART V Appendices

Chapter 19 lists WirelessHART attributes that are time related and describes all the message fields. The readers could use it as a quick reference guide.

Chapter 20 collects the symbols and abbreviations used in the standard and this book.

Chapter 21 collects the definitions of the technical terms used in the standard and this book.

Chapter 22 lists the references.

Chapter 19 Attribute and Field Values

19.1 Comments on Message Field Values

Most of the device attribute values could be set by the network manager. Here we list some default values when they are not configured.

Table 19.1 Time related parameters.

attribute	meaning	Default value
ActiveSearchShedTime	Max amount of time to stay in active search mode while joining. After this interval lapses the device transitions to passive search mode.	4000 seconds
advertiseInterval	Time period specifying the transmission of Advertise DLPDUs	None. Need to be as fast as possible during mesh forming stage.
AdWaitTimeout	The amount of time to wait while attempting to receive additional Advertisements	30 seconds
BcastReplyTime	Maximum amount of time to reply to a broadcast message	60 seconds
ChannelSearchTime	The amount of time to stay on a given channel while listening for Advertise packets	400 milliseconds
DefaultTTL	Packet life limit in hops. Specifies the number of hops a packet can travel before being discarded	32
discoveryInterval	Time period specifying the interval bounding the random transmission of Keep Alive DLPDUs on Discovery links.	None
HealthReportTime	Period at which to publish health reports	15minutes
JoinRspTimeout	Join response timeout	keepAliveInterval
keepAliveInterval	Interval during which a node must successfully communicate with each linked neighbor. Any	30 seconds

	DLPDU received from the neighbor resets the Keep-Alive timer for that neighbor.	
maxJoinRetries	Join retry limit	5
maxReplyTime	Used to trigger retires by Transport Layer	30 seconds
maxPacketAge	Maximum number of slots a packet can live while hopping the mesh.	300 seconds
maxRetries	Number of retries used by the Transport Layer when performing an acknowledged packet transmission	5
minAdsNeeded	Preferred number of different Advertisements before issuing join request	3
PassiveCycleTime	When in passive search mode, the period over which the device cycles between sleeping and listening. The sleep interval equals the PassiveCycleTime minus the PassiveWakeTime.	600 seconds
PassiveWakeTime	When in passive search mode, the amount of time to be awake listening for the network	6.5 seconds
pathFailInterval	Interval of unsuccessful communication with a given neighbor, indicating a path failure	none

19.2 WirelessHART Message Fields

The WirelessHART standard adopts the IEEE 802.15.4 standard at the lower layers. Some of the IEEE 802.15.4 message fields are fixed or limited for WirelessHART messages. In addition, not all WirelessHART message fields are required for all message types.

Table 19.2 Physical Layer Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Preamble	4	Conform to IEEE
-	Delimiter	1	Conform to IEEE
-	Byte count	1	Conform to IEEE
Upper Layer	Payload	*	Conform to IEEE

Table 19.3 Data Link Layer Data Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Header	6	Conform to IEEE
Data Link	Frame Control Byte 1	1	0x41. Conform to IEEE
-	Address Specifier	1	0x88, 0x8C, 0xC8, or 0xCC. Conform to IEEE
-	Sequence Number	1	LSB of ASN. Conform to IEEE
-	Network ID	2	Conform to IEEE
-	Destination	2/8	Conform to IEEE
-	Source	2/8	Conform to IEEE
-	DLPDU Specifier	1	-
Upper Layer	Payload	*	-
Data Link	MIC	4	-
-	CRC	2	Conform to IEEE

Table 19.4 Data Link Layer Acknowledgement Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Header	6	Conform to IEEE
Data Link	Frame Control Byte 1	1	0x41. Conform to IEEE
-	Address Specifier	1	0x88, 0x8C, 0xC8, or 0xCC. Conform to IEEE
-	Sequence Number	1	LSB of ASN. Conform to IEEE
-	Network ID	2	Conform to IEEE
-	Destination	2/8	Conform to IEEE
-	Source	2/8	Conform to IEEE
-	DLPDU Specifier	1	Depend on received message
-	Status	1	-
-	Time Adjustment	2	-
-	MIC	4	-
-	CRC	2	Conform to IEEE

Table 19.5 Data Link Layer Advertisement Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Header	6	Conform to IEEE
Data Link	Frame Control Byte 1	1	0x41. Conform to IEEE
-	Address Specifier	1	0x88. Conform to IEEE
-	Sequence Number	1	LSB of ASN. Conform to IEEE
-	Network ID	2	Conform to IEEE

-	Destination	2	0xFFFF. Conform to IEEE
-	Source	2	Conform to IEEE
-	DLPDU Specifier	1	0x31 or 0x11
-	ASN	5	-
-	Join Control	1	-
-	Channel Map Size	1	-
-	Channel Map	*	-
-	Graph ID	2	The graph ID this device will use for the new device
-	Number of superframes	1	-
-	Superframe details	*	-
-	MIC	4	-
-	CRC	2	Conform to IEEE

Table 19.6 Data Link Layer Keep Alive Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Header	6	Conform to IEEE
Data Link	Frame Control Byte 1	1	0x41. Conform to IEEE
-	Address Specifier	1	0x88. Conform to IEEE
-	Sequence Number	1	LSB of ASN. Conform to IEEE
-	Network ID	2	Conform to IEEE
-	Destination	2	0xFFFF.Conform to IEEE
-	Source	2/8	8 if new device without short address. Conform to IEEE
-	DLPDU Specifier	1	0x3A or 0x1A. 0x32 or 0x12 if new device without short address.
-	MIC	4	-
-	CRC	2	Conform to IEEE

Table 19.7 Data Link Layer Disconnect Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Header	6	Conform to IEEE
Data Link	Frame Control Byte 1	1	0x41. Conform to IEEE
-	Address Specifier	1	0x88. Conform to IEEE
-	Sequence Number	1	LSB of ASN. Conform to IEEE
-	Network ID	2	Conform to IEEE
-	Destination	2	0xFFFF. Conform to IEEE
-	Source	2	Conform to IEEE
-	DLPDU Specifier	1	0x3B or 0x1B

-	MIC	4	-
-	CRC	2	Conform to IEEE

Table 19.8 Network Layer Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Header	6	Conform to IEEE
Data Link	Header	*	-
Network	Control	1	-
-	TTL	1	-
-	ASN Snippet	2	-
-	Graph ID	2	-
-	Destination	2/8	-
-	Source	2/8	-
-	(Optional) Proxy/Source Routing	2/4/6/8/10	-
Network/Security	Security Control	1	-
-	Counter	1/4	-
-	MIC	4	-
Transport	Transport Control	1	Include sequence number
-	Device Status	1	-
-	Extended Device Status	1	-
Upper Layer	Payload	*	-
Data Link	FOOTER	6	-

Table 19.9 Application Layer Message Format.

Network layer	Name	Number of bytes	Comments
Physical	Header	6	Conform to IEEE
Data Link	Header	*	-
Network	Header	*	-
Application	Command	2	-
-	Byte Count	1	-
-	Data	*	These 3 fields could be repeated for multiple commands
Data Link	FOOTER	6	-

Chapter 20 Symbols and Abbreviations

Table 20.1 Symbols and Abbreviations.

Name	Definition
802.15.4	IEEE STD 802.15.4-2006 in general. When referring to the Physical Layer it refers to the 2.4GHz DSSS Physical Layer employing OQPSK modulation.
ACK	Acknowledge
AE	Application Entity
AES	Advanced Encryption Standard
AL	Application Layer
AOA	Angle of Arrival
AP	Application Process
API	Application Program Interface
APDU	Application Protocol Data Unit
APO	Application Object
AR	Application Relationship
AREP	Application Relationship Endpoint
ARPM	Application Relationship Protocol Machine
ARQ	Automatic Repeat Request
ASCII	American Standard Code for Information Interchange
ASE	Application Service Element
ASN	Absolute Slot Number
AWGN	Additive White Gaussian Noise
BACK	Burst Acknowledge
BER	Bit Error Rate
CBC-MAC	Cipher Block Chaining Message Authentication Code
CCA	Clear Channel Assessment
CCM	Counter with CBC-MAC (mode of operation)
CCM*	extension of CCM
Cnf	Confirmation
COTS	Commercial Off The Shelf
CPS	Cyber-Physical System
CPU	Central Process Unit
CRC	Cyclic Redundancy Check

CSMA	Carrier Sense Multiple Access
CSMA-CA	CSMA with Collision Avoidance
dB	Relative Power Decibels
dBi	dBi is used to express the gain of an antenna in decibels. The terminal letter 'I' indicates that the gain is relative to an isotropic antenna.
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
DCS	Distributed Control System
DDL	Device Description Language
DL-	Data-Link Layer (as a prefix)
DLE	DL-Entity (the local active instance of the data-link layer)
DLL	Data-Link Layer
DLM	DL-Management
DLMS	DL-Management Service
DLPDU	Data-Link Protocol Data Unit (i.e., a Data-Link Layer packet)
DLS	DL-Service
DLSDU	DL-Service-Data-Unit
DR	Delayed Response
DRM	Delayed Response Mechanism
DSN	Data Sequence Number
DSSS	Direct Sequence Spread Spectrum
DUT	Device Under Test
ECDH	Elliptic Curve Diffie-Hellman
EDD	Electronic Device Description
EDDL	Electronic Device Description Language
EDF	Earliest Deadline First
EIRP	Effective Isotropic Radiated Power
ERP	Effective Radiated Power
EUI-64	Extended Unique Identifier (64 bits long)
FAL	Fieldbus Application Layer
FCC	Federal Communications Commission
FF	Foundation Fieldbus
FFD	Full Function Device
FSK	Frequency Shift Keyed
FTA	Field Termination Assembly
FHSS	Frequency Hopping Spread Spectrum
FSMP	FAL Service Protocol Machine
HART	Highway Addressable Remote Transducer
HCF	HART Communication Foundation

IAE	Integral Absolute Error
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	The Institute of Electrical and Electronics Engineers
Ind	Indication
IO	Input Output
I/O	Input Output
IS	Information System
ISO	International Organization for Standardization
ISM	Industry, Scientific, Medical frequency bands
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector
LLC	Logical Link Control
LED	Light Emitting Diode
LoS	Line of Sight, an unobstructed distance between a transmitter and a receiver
LSB	Least Significant Byte. The LSB is always the last byte transmitted over a HART data link
MAC	Medium Access Control.
MIC	Message Integrity Code
MSB	Most Significant Byte. The MSB is always the first byte transmitted over a HART data link.
NL	Network Layer
NLOS	Non-Line-of-Sight
NPDU	Network PDU
OLE	Object Linking and Embedding
OPC	OLE for Process Control
OQPSK	Offset Quadrature Phase Shift Keying
OS	Operating System
OSI	Open System Interconnection
OUI	Organizationally Unique Identifier
PAN	Personal Area Network
PCP	Priority Ceiling Protocol
PCS	Physical Control System
PER	Packet Error Rate
PDU	Protocol Data Unit. The packet of information being communicated.
PhPDU	Physical Layer Protocol Data Unit (i.e., a Physical Layer packet)
PHY	Physical Layer
PIB	PAN Information Base
PID	Proportional, Integral, and Derivative
PLC	Programmable Logic Controller
PPDU	PhPDU
ppm	parts per million

PSK	Phase Shift Keyed
PV	Primary Variable
QV	Quaternary Variable
RF	Radio Frequency
RFD	Reduced Function Device
RAM	Random Access Memory
RFID	Radio-Frequency Identification
RMA	Rate Monotonic Algorithm
RSL	Received Signal Level. The signal level (in dBm) at a receiver input terminal.
RSSI	Received Signal Strength Indication
RTOS	Real-Time OS
SAP	Service Access Point
SCADA	Supervisory Control And Data Acquisition
SFD	Start of Frame Delimiter
SINR	Signal to Interference-Plus-Noise Ratio
SNR	Signal to Noise Ratio
SOM	Start of the Message
SP	Service Primitive
STX	Start of a transaction. An STX is used to convey a Network layer packet (an NPDU) from one node to an adjacent node.
SV	Secondary Variable
TDMA	Time Division Multiple Access
TDOA	Time Difference of Arrival
TER	Transaction Error Rate
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
TPDU	Transport PDU
TTL	Time To Live
TV	Tertiary Variable
UTC	Coordinated Universal Time
UWB	Ultra Wideband
WHA	WirelessHART Adapter
WHD	WirelessHART Device
WINA	Wireless Industrial Networking Alliance
XML	eXtensible Markup Language

Chapter 21 Definitions

4-20mA A point-to-point or multi-drop circuit mainly used in the process automation field to transmit signals from instruments and sensors in the field to a controller. It sends an analog signal from 4 to 20 mA that represents 0 to 100% of some process variable. As a current loop signal, 4-20 mA also powers the sensor transmitter on the same wire pair, and 4-20mA provides more resistance to interference than a voltage-based line.

(http://www.pcmag.com/encyclopedia_term/0,2542,t=4-20+mA&i=37097,00.asp)

Absolute slot number count of all slots that have occurred since the network was formed NOTE This number always increments and is never reset to any fixed value or zero. Its current value is always the sequence number of the current slot. Its maximum value is (2^{40} - 1).

Acknowledge explicit data-link response to the successful reception of a directed, non-broadcast DLPDU from a DLE source device and the second DLPDU of a two-DLPDU transaction.

Adapter network device that enables connection and communication with a field device which does not have a direct connection to a Type 20 network

Advertise DLPDU invite new devices into the network NOTE When a device wishes to join a network, it listens for these DLPDUs and then uses the information in the DLPDU to synchronize with the network and initiate the join process.

Analog channel continuously varying electrical signal connecting a field device to the remainder of the data acquisition or control system NOTE 1 Some field devices support multiple analog channels (input or output) NOTE 2 Each analog channel transmits a single dynamic variable to or from the field device.

Antenna Gain The apparent power gain resulting from the antenna capability of concentrating power in a given direction.

Application function or data structure for which data is consumed or produced

Application layer interoperability capability of application entities to perform coordinated and cooperative operations using the services of the FAL

Application object Object class that manages and provides the run time exchange of messages across the network and within the network device. NOTE: Multiple types of application object classes may be defined.

Application process part of a distributed application on a network, which is located on one device and unambiguously addressed

Application process identifier distinguishes multiple application processes used in a device

Application process object component of an application process that is identifiable and accessible through an FAL application relationship NOTE Application process object definitions are composed of a set of values for the attributes of their class (see the definition for Application Process Object Class Definition). Application process object definitions may be accessed remotely using the services of the FAL Object Management ASE. FAL Object Management services can be used to load or update object definitions, to read object definitions, and to dynamically create and delete application objects and their corresponding definitions.

Application process object class class of application process objects defined in terms of the set of their network-accessible attributes and services

Application relationship cooperative association between two or more application-entity-invocations for the purpose of exchange of information and coordination of their joint operation NOTE This relationship is activated either by the exchange of application-protocol-data-units or as a result of pre-configuration activities.

Application relationship application service element application-service-element that provides the exclusive means for establishing and terminating all application relationships

Application relationship endpoint context and behaviour of an application relationship as seen and maintained by one of the application processes involved in the application relationship NOTE Each application process involved in the application relationship maintains its own application relationship endpoint.

Absolute Slot Number The count of all slots that have occurred since the network was formed and always contains the number of the current slot. The Absolute Slot Number is only incremented and must never be reset.

ASN time time represented as absolute slot number; any instance in time is equal to absolute slot number at that instance

Assailant The device generating interference.

Attribute description of an externally visible characteristic or feature of an object. NOTE: The attributes of an object contain information about variable portions of an object. Typically, they provide status information or govern the operation of an object. Attributes may also affect the behavior of an object. Attributes are divided into class attributes and instance attributes

Behavior indication of how the object responds to particular events NOTE Its description includes the relationship between attribute values and services.

Broadcast the process of sending a PDU to all devices that are connected to the network and are able to receive the transmission

Broadcast address broadcast address is used by a master to send a command to all devices NOTE The broadcast address is five octets long and has all zeros as the value.

Burst-mode device This is a device which provides a digital response carrying measurement or other data, at regular intervals, without the data being specifically requested, i.e., this device normally functions as an independently broadcast device. A Burst-Mode Device is defined to be a Slave Device with burst capability (hence the use of the word "mode" in describing the device type). When such a mode is enabled, the Slave Device is said to "be in burst mode".

Busy device is active on different tasks and cannot execute a command at the time. NOTE: A device indicates busy by returning response code 32 when allowed by the command specification. The requested command is not executed if a busy response is returned.

Byte 8-bits, sometimes called an Octet.

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

Channel blacklisting method of eliminating an RF channel from usage

Channel hopping regular change of transmit or receive frequency to combat interference and fading

Channel offset link-specific value provided by the network manager that is used to calculate the channel to use when channel hopping

Class set of objects, all of which represent the same kind of system component. NOTE: A class is a generalization of the object; a template for defining variables and methods. All objects in a class are identical in form and behavior, but usually contain different data in their attributes.

Class attributes attribute that is shared by all objects within the same class

Class code unique identifier assigned to each object class

Class specific service service defined by a particular object class to perform a required function which is not performed by a common service NOTE A class specific object is unique to the object class which defines it.

Clear channel assessment avoid initiating a transaction while the RF channel is in use. NOTE: It is performed by listening to the channel prior to sending the first

DLPDU of a transaction. If signal is detected, then the transaction is deferred to a later TDMA slot.

Client a) an object which uses the services of another (server) object to perform a task b) an initiator of a message to which a server reacts, such as the role of an AR endpoint in which it issues confirmed service request APDUs to a single AR endpoint acting as a server

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).

Connection A data structure associated with graph routing that contains an ordered pair of Network Devices.

Conveyance path unidirectional flow of APDUs across an application relationship

Cyclic term used to describe events which repeat in a regular and repetitive manner

Data-Link Layer Layer 2 in the OSI Basic Reference Model. This layer is responsible for the error-free communication of data. The Data-Link Layer defines the message structure, error detection strategy and bus arbitration rules.

Dedicated AR AR used directly by the FAL User NOTE On Dedicated ARs, only the FAL Header and the user data are transferred.

Device any entity containing an implementation of Type 20 fieldbus

Device ID serial number for a device NOTE The manufacturer is required to assigned unique value for every device that has the identical values for Manufacturer ID and Device Type.

Device profile collection of device dependent information and functionality providing consistency between similar devices of the same device type

Device type manufacturer's type of a device NOTE The value of this attribute is assigned by the manufacturer. Its value specifies the set of commands and data objects supported by the device. The manufacturer is required to assigned unique value to each type of the device. An example for a device type name could be its product name.

Device variable uniquely defined data item within a Field Device that is always associated with cyclical process information NOTE A device variable's value varies in response to changes and variations in the process to which the device is connected.

Discovery process of finding a new neighbour by listening the special DLPDU that indicates the presence of the device

Discovery link the link that is used for transmitting and receiving the special DLPDU that indicates the presence of the device

Dynamic variable connection between a process and an analog channel. NOTE: A device may contain primary, secondary, tertiary, and quaternary variables. These are collectively called the dynamic variables.

Endpoint one of the communicating entities involved in a connection

Error discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition

Error code identification of a specific type of error within an error class

Field device network device that is connected to the process or to a plant equipment NOTE It directly connects to the sensor or actuator or performs process control function and it is directly connected to the physical layer of Type 20.

Frame A Data-Link Layer "packet" which contains the header and trailer information required by the physical medium. That is, Network Layer packets are encapsulated to become frames.

Frequency channel allocation of the frequency spectrum in a given frequency range

Gateway network device containing at least one host interface such as serial or Ethernet, acting as ingress or an egress point enabling communication between host applications and field devices

Graph routing structure that forms a directed end-to-end connection between network devices

Graph ID identifier used to indicate a specific graph entry

Handheld host application residing on a portable device

HCF enumeration enumeration which is controlled and maintained by the HART Communications Foundation (HCF)

Hop movement of a packet directly between two adjacent neighbors in one network transaction without the participation of any other node in the network. Also used to denote the function of changing channels

Host One of (possibly) several applications that can be executed sequentially or simultaneously on a master.

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.

Join process by which a network device is authenticated and allowed to participate in the network NOTE A device is considered Joined when it has the network key, a network manager session and a normal (not join) superframe and links.

Join key security key that is used to start the join process

Latency time it takes for a packet to cross a network connection, from sender to receiver

Lease A lease is an agreement between the host and the WirelessHART Gateway to share a resource for a future period of time; after which the resources can be reallocated for other purpose.

Link full communication specification between adjacent devices in a network and it includes the communication parameters necessary to move a DLPDU one hop. NOTE: A Link is a function of source and destination address pairing, slot and channel offset assignment, direction of communication, dedicated or shared communication, and type. Links are assigned to superframes as part of the scheduling process.

Link Margin The difference between the power of a received signal and the sensitivity of the receiver. Typically, this determines the viability of a link. Around 10 dB of margin is desirable for a reliable link.

Logical link control higher of the two data-link layer levels NOTE This level handles error detection, flow control, framing, and addressing.

Long tag 32-character restricted ISO Latin-1 string used to identify a field device

Loop current value measured by a mA in series with the field device. NOTE: The loop current is a near DC analog 4-20 mA signal used to communicate a single value between the control system and the field device. Voltage mode devices use “Volts DC” as their engineering units where “loop current” values are used.

Maintenance port interface in the network device that is used for provisioning. NOTE: That means to write the join key, network ID and to monitor the join process

Management information network-accessible information that supports managing the operation of the fieldbus system, including the application layer NOTE Managing includes functions such as controlling, monitoring, and diagnosing.

Manufacturer ID 2 octet enumeration identifying the manufacturer that produced a device NOTE A manufacturer is required to use the value assigned to it and is not permitted to use the value assigned to another manufacturer.

Master a device that initiates communication activity by sending request APDU to a device and expecting a response PDU

Medium access control lower of the two data-link layer levels NOTE This level controls the access to the communication channel.

Multicast sending of one packet to more than one device in the network using only one transmission

Neighbor adjacent node in the network such that the receive signal level (RSL) from it suggests that the communication is possible in at least one direction

Neighbor table list of neighbors that a DLE has knowledge of NOTE It also stores the properties of the neighbor.

Network series of nodes connected by some type of communication medium NOTE The connection paths between any pair of nodes can include repeaters, routers and gateways.

Network device device with a direct physical layer connection to the network. NOTE: Each network device has a unique address that is used in communication with the device. Network devices include field device, access point (i.e. gateway), adapter and handheld.

Network ID identifier used to indicate a network to which all inter-communicating devices are connected. NOTE: A device connected to one network can not send a PDU to another device connected to a different network.

Network manager entity that 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 network. NOTE: There is one and only one network manager per instance of Type 20 network. Although the network manager need not have a direct physical layer connection, it still has a Unique address.

Nickname device identifier that is unique within the network to which the device is connected. NOTE: The 2-octet nickname is assigned and managed by the network manager connected to the operational network to which the device is connected.

Node addressable logical or physical device attached to the network

Nonce non-repeating number constructed so as to be unique to the current packet and to ensure that old communications cannot be reused in replay attacks. NOTE: Nonce is necessary for maintaining packet secrecy and providing sender authenticity and packet integrity.

Omni-directional Antenna An antenna with a radiation pattern that, when viewed from above, is equally strong in all directions (i.e., the antenna sends or receives signals equally well in all directions)

Packet the formatted, aggregated bits that are transmitted together in time across the physical medium

Packet Error Rate Average number packets (in percent) transmitted but not received correctly.

Payload data contents of a data message that is being transmitted

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 Basic Reference Model. The Physical Layer is responsible for transmission of the raw bit stream and defines the physical (e.g., mechanical, electrical) connections and signaling parameters for devices.

Polling address integer used to identify a device NOTE The polling address is used to construct a one-octet address.

Pre-defined AR endpoint AR endpoint that is defined locally within a device without use of the create service NOTE Pre-defined ARs that are not pre-established are established before being used.

Pre-established AR endpoint AR endpoint that is placed in an established state during configuration of the AEs that control its endpoints

Receiver Sensitivity The minimum input signal required to produce a PER of less than 1% with a PPDU 20 bytes long (IEEE).

Route ID identifier used to indicate a specific route

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

Server <communication> role of an AREP in which it returns a confirmed service response APDU to the client that initiated the request

Server <system> an object which provides services to another (client) object

Service operation or function than an object and/or object class performs upon request from another object and/or object class. NOTE: A set of common services is defined and provisions for the definition of object-specific services are provided. Object-specific services are those which are defined by a particular object class to perform a required function which is not performed by a common service.

Service Session An agreement between a Client and a WirelessHART Gateway that services shall be provided to the Client by the Gateway.

Session ID identifier used to indicate a specific session entry

Slave a device that initiates communication activity only after it receives a request PDU from a master device and it is required to send a response to that request

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

Stream A reliable, virtual connection between two Application Layer functions. A stream delivers exactly the same sequence of data bytes to the destination device that the sender communicates to the source device transport layer.

Superframe collection of slots repeating at a constant rate; each slot has a link associated with it

Superframe ID identifier used to indicate a specific superframe entry

Tag 8-character ASCII string used to identify a field device

Throughput The effective data transfer rate of the network.

Time division multiple access medium access control technique that uses time slots where communications between devices can occur. NOTE: It provides collision free, deterministic communications.

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

Transaction exchange of related, consecutive frames between two peer medium access control entities, required for a successful transmission. NOTE: A transaction consists of either (a) a single PhPDU transmission from a source device, or (b) one PhPDU from the source device followed by a second, link-level acknowledgement PhPDU from the destination device.

Transport Layer The Transport Layer provides reliable data transfer between two deices. Communication is transparent in that detailed low level knowledge of the communication is not required.

Unicast sending of a packet to a single device in the network

Unique ID concatenation of the 2-octet Expanded Device type code and the 3-octet device identifier. NOTE: The Expanded Device type code is allocated by the HCF (HART Communication Foundation). Each device manufactured with the same Device type code assigns a unique device identifier to every instance of the device.

UTF-8 UTF-8 (8-bit UCS/Unicode Transformation Format) is a variable-length character encoding for Unicode.

Chapter 22 References

22.1 HART 7 Protocol Specifications

1. HART Field Communication Protocol Specification (HCF_SPEC-13)
2. FSK Physical Layer Specification (HCF_SPEC-54)
3. C8PSK Physical Layer Specification (HCF_SPEC-60)
4. Wireless Physical Layer Specification (HCF_SPEC-65)
5. TDMA Data Link Layer (HCF_SPEC-75)
6. Data Link Layer Specification (HCF_SPEC-81)
7. Network Management Specification (HCF_SPEC-85)
8. Command Summary Specification (HCF_SPEC-99)
9. Universal Command Specification (HCF_SPEC-127)
10. Common Practice Command Specification (HCF_SPEC-151)
11. Wireless Command Specification (HCF_SPEC-155)
12. Device Families Command Specification (HCF_SPEC-160)
13. Temperature Device Family Specification (HCF_SPEC-160-4)
14. PID Device Family Specification (HCF_SPEC-160-7)
15. Common Tables (HCF_SPEC-183)
16. Block Data Transfer Specification (HCF_SPEC-190)
17. Discrete Applications Specification (HCF_SPEC-285)
18. Wireless Devices Specification (HCF_SPEC-290)
19. Command Specific Response Code Definitions (HCF_SPEC-307)
20. Field Device Specific Specification Template (HCF_LIT-18)
21. Field Device Specific Specification Template (Sample) (HCF_LIT-18)
22. HART Slave Data Link Layer, Test Specification (HCF_TEST-1)
23. HART Physical Layer Test Procedure (HCF_TEST-2)
24. HART Slave Application Layer Universal Command Test Specification (HCF_TEST-3)
25. Application Layer Common Practice Command Test Specification (HCF_TEST-4)

Among these, the following specifications are new for wireless:
Wireless Physical Layer Specification (HCF_SPEC-65)
TDMA Data Link Layer (HCF_SPEC-75)
Network Management Specification (HCF_SPEC-85)
Wireless Command Specification (HCF_SPEC-155)
Wireless Devices Specification (HCF_SPEC-290)

22.2 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
Approved IEEE 802.15.4 Transceivers. HCF_LIT-088

22.3 Related Documents Cited by HART

The following are applicable IEEE documents:

IEEE STD 802.15.4-2006. Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). 2006
Diffie W, Hellman M (1979) Privacy and Authentication: An Introduction to Cryptography. Proceedings of the IEEE, Vol. 67 No. 3, pp 397-427

In addition, the application of the IEEE Extended Unique Identifier (EUI-64) and Organizationally Unique Identifier (OUI) can be found at:

IEEE. (accessed 1 August, 2007). IEEE Registration Authority - Tutorials. IEEE Standards Association, http://standards.ieee.org/regauth/tutorials.html.

The following document provides additional information about and algorithms for the 16 bit ITU-T CRC (also known as CRC16).

Simpson W, Editor (1993) PPP in HDLC Framing. RFC 1549, http://www.ietf.org/rfc/rfc1549.txt, IETF 1993.

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

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

On byte ordering

Wikipedia contributors (accessed 9 February, 2007) Endianness. Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/w/index.php?title=Endianness&oldid=105787173.
Cohen D (accessed 9 February, 2007) On Holy Wars And A Plea For Peace, DAV's Endian FAQ http://www.rdrop.com/~cary/html/endian_faq.html.

The following reference provides additional information about and algorithms for the CCM* Mode algorithm used in conjunction with AES-128 cipher for security.

Dworkin M (2004) Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality. National Institute of Standards and Technology. Special Publication 800-38C.

The following reference describes communication specification techniques.

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

The following reference describes the methods for specifying state transition diagrams.

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

22.4 Other References

Publications used by this book.

Journal and Conference articles

1. Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E (2002) A survey on sensor networks. IEEE Communication Magazine.
2. Basu A, Gao J, Mitchell JSB, Sabhnani G (2006) Distributed localization using noisy distance and angle information. MobiHOC
3. Bavier A, Feamster N, Huang M, Peterson L, Rexford J (2006) In vini veritas: realistic and controlled network experimentation. SIGCOMM, pp. 3–14, ACM.
4. Chen D, Mok AK, Baruah SK (1997) On Modeling Real-time Task Systems. Lecture Notes in Computer Science - Lectures on Embedded Systems, v(1494).
5. Chen D, Mok AK, Kuo T-W (2003) Utilization Bound Re-visited. IEEE Transactions on Computers.
6. Chen D, Nixon M, Aneweer T, Shepard R, Blevins T, McMillan G, Mok AK (2005) Similarity-based Traffic Reduction to Increase Battery Life in a Wireless Process Control Network. ISA EXPO, Chicago, USA.
7. Chen D, Nixon M, Aneweer T, Shepard R, Burr K, Mok AK (2005) Wireless Process Control Products from ISA 2004. International Workshop on Wireless and Industrial Automation.
8. Chen D, Nixon M, Aneweer T, Shepard R, Mok AK (2004) Middleware for Wireless Process Control Systems. Architectures for Cooperative Embedded Real-Time Systems Workshop.
9. Chen S, Chen Y, Trappe W (2007) Exploiting Environmental Properties for Wireless Localization. MobiCom.
10. Chung SP, Mok A (2007) Advanced Allergy Attacks: Does a Corpus Really Help? RAID, LNCS 4637, 236-255.
11. Cristian F, Fetzer C (1999) The Timed Asynchronous System Model. IEEE Transactions on Parallel and Distributed Systems, 10(6), June, 642-657.
12. Culler D, Estrin D, Srivastava M (2004) Overview of Sensor Networks, Computer.
13. Diffie W, Hellman M (March 1979) Privacy and Authentication: An Introduction to Cryptography. Proceedings of the IEEE, Vol. 67 No. 3, pp 397-427.
14. Elbatt T, Saraydar C, Ames M, Talty T (2006) Potential for Intra-Vehicle Wireless Automotive Sensor Networks. IEEE Sarnoff Symposium, pp. 1-4.
15. Elnahrawy E, Li X, Martin R (2004) The Limits of Localization Using Signal Strength: A Comparative Study. IEEE SECON.
16. Fang L, Du W, Ning P (2005) A beacon-less location discovery scheme for wireless sensor networks. Infocm.
17. Guha S, Murty RN, Sirer EG (2005) Sextant: A Unified Framework for Node and Event Localization in Sensor Networks. MobiHoc.

18. Hightower J, Boriello G (2001) Location Systems for Ubiquitous Computing. IEEE Computer, vol. 34, no. 8, pp.57–66.
19. Han S, Song J, Zhu X, Mok AK, Chen D, Nixon M, Pratt W, Gondhalekar V (2009) WiHTest: Compliance Test Suite for Diagnosing Devices in Real-Time WirelessHART Network. RTAS.
20. Hu L, Evans D (2004) Localization for Mobile Sensor Networks. MobiCom.
21. Krishnamurthy L, Adler R, Buonadonna P, Chhabra J, Flanigan M, Kushalnagar N, Nachman L, Yarvis M (2005) Design and deployment of industrial sensor networks: experiences from a semiconductor plant and the north sea. SenSys, pp. 64–75.
22. Kuo T-W, Mok AK (1991) Load Adjustment in Adaptive Real-Time Systems. IEEE Real-Time Systems Symposium.
23. Liu CL, Layland JW (1973) Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. Journal of ACM.
24. Liu J, Zhang Y, Zhao F (2006) Robust distributed node localization with error management. MobiHOC.
25. Liu W, Lou W, Fang Y (2005) An efficient quality of service routing algorithm for delay-sensitive applications. Computer Networks, v(47).
26. Luo J, Shukla HV, Hubaux JP (2006) Non-Interactive Location Surveying for Sensor Networks with Mobility-Differentiated ToA. INFOCOM.
27. McCluer S (2003) Wanted: Real World Battery Life Prediction. American Power Conversion Coporation.
28. Nixon M, Chen D, Blevins T, Mok AK (2008) Meeting Control Performance over a Wireless Mesh Network. CASE.
29. Nixon M, Shepard R, Bennett B, Chen D, Mok AK (2004) A Framework to Transmit Process Control Data over Commercial Wireless Networks. ISA Technical Conference.
30. Nixon M, Shepard R, Mok AK, Bennett B, Chen D (2005) Process Control Adopts Wireless. InTech Magazine, 52(2).
31. Peng C, Shen G, Zhang Y, Li Y, Tan K (2007) BeepBeep: A High Accuracy Acoustic Ranging System using COTS Mobile Devices. ACM SenSys.
32. Sha L, Rajkumar R, Lehoczky J (1990) Priority Inheritance Protocols: An Approach to Real-Time System Synchronization. IEEE Trans. on Computers, 39(9), 1175-1185.
33. Sha L, Rajkumar R, Sathaye S (1994) Generalized Rate-Monotonic Scheduling Theory: A Framework for Developing Real-Time Systems. In Proceeding of the IEEE. Vol. 82. No. 1, pp. 68-82.
34. Sheldon M, Chen D, Nixon M, Mok AK (2005) A Practical Approach to Deploy Large Scale Wireless Sensor Networks, Workshop on Resource Provisioning and Management in Sensor Networks.
35. Soldati P, Zhang H, Johansson M (2008) Deadline-constrained transmission scheduling and data evacuation in wirelessHART networks. Technical Report, Automatic Control Lab, School of Electrical Engineering, Royal Institute of Technology, Sweden.
36. Song J, Han S, Mok AK, Chen D, Lucas M, Nixon M, Pratt W (2008) Wirelesshart: Applying wireless technology in real-time industrial process control. RTAS, pp. 377–386.
37. Song J, Han S, Mok AK, Chen D, Nixon M (2007) A study of process data transmission scheduling in wireless mesh networks. ISA EXPO Technical Conference.
38. Song J, Han S, Mok AK, Chen D, Nixon M (2007) Centralized Control of Wireless Sensor Networks for Real-Time Applications, 7th IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems.
39. Song J, Han S, Zhu X, Mok AK, Chen D, Nixon M (2008) Demonstration of a complete wirelesshart network. SenSys demo.
40. Song J, Mok AK, Chen D, Nixon M (2006) Challenges of Wireless Control in Process Industry. Workshop on Research Directions for Security and Networking in Critical Real-Time and Embedded Systems.

41. Song J, Mok AK, Chen D, Nixon M (2006) Using Real-Time Logic Synthesis Tool to Achieve Process Control over Wireless Sensor Networks. RTCSA.
42. Song J, Mok AK, Chen D, Nixon M, Blevins T, Wojsznis W (2006) Improving pid control with unreliable communications. ISA EXPO Technical Conference.
43. Stankovic J, Lee I, Mok A, Rajkumar R (2005) Opportunities and Obligations for Physical Computing Systems. IEEE Computer, 38(11), November, 23-31.
44. Stankovic J, Spuri M, Ramamritham K, Buttazzo G (1998) Deadline Scheduling For Real-Time Systems: EDF and Related Algorithms. Kluwer Academic Publishers, Boston.
45. Thonet G, Allard-Jacquin P, Colle P (2008) ZigBee - WiFi Coexistence - White Paper and Test Report, Schneider Electric.
46. Vieira MAM, da Silva DC Jr, Coelho CN Jr, da Mata JM (2003) Survey on wireless sensor network devices. Emerging Technologies and Factory Automation.
47. Weiss JM (2005) Cyber Security Meets Plant Politics. InTech Magazine.
48. Woo H, Mok K (2007) Real-Time Monitoring of Uncertain Data Streams using Probabilistic Similarity. RTSS.
49. Zhang H, Soldati P, Johansson M (2009) Efficient Link Scheduling and Channel Hopping for Convergecast in WirelessHART Networks. Technical Report, Automatic Control Lab, School of Electrical Engineering, Royal Institute of Technology, Sweden.
50. Zhang H, Soldati P, Johansson M (2009) Optimal Link Scheduling and Channel Assignment for Convergecast in Linear WirelessHART Networks. Technical Report, Automatic Control Lab, School of Electrical Engineering, Royal Institute of Technology, Sweden.
51. Zhu X, Dong W, Mok AK, Han S, Song J, Chen D, Nixon M (2009) A Location-determination Application in WirelessHART. RTCSA.

Books

1. Blevins T, McMillan G, Wojsznis W, Brown M (2002) Advanced Control Unleashed: Plant Performance Management for Optimum Benefit. ISA Press.
2. Callaway EH Jr, Callaway EH (2003) Wireless Sensor Networks: Architectures and Protocols. CRC Press.
3. Caro D (2004) Wireless Networks for Industrial Automation, ISA Press.
4. Chen D (1999) Real-Time Data Management in the Distributed Environment. Ph.D. Thesis, the University of Texas at Austin.
5. Gutierrez J, Callaway E, Barrett R (Jan 1, 2007) Low-Rate Personal Area Networks: Enabling Wireless Sensors with IEEE 802.15.4. 2nd edn.
6. Halsall F (1992) Data Communications, Computer Networks and Open Systems. 3rd edn. Addison Wesley.
7. Hieb B (2003) Developing a Small Wireless Control Network. Master's Thesis, the University of Texas at Austin.

Standards and Organizations

1. (November 2001) Advanced Encryption Standard (AES), U. S. FIPS Publication 197, DoC/NIST.
2. (1993) Automatic Controller Dynamic Specification. EnTech Control Engineering Inc., http://www.emersonprocess.com/entechcontrol/download/publications/control.pdf, Version 1.0.
3. Bluetooth, www.bluetooth.com/bluetooth
4. Electronic Device Description Language, http://www.eddl.org
5. Foundation Fieldbus, www.fieldbus.org
6. HART Foundation, www.hartcomm.org
7. IEEE 802.11 Task Group, grouper.ieee.org/groups/802/11
8. IEEE 802.15.4 WPAN Task Group, www.ieee802.org/15/pub/TG4.html
9. IEEE wireless standards, http://standards.ieee.org/wireless
10. The Instrumentation, Systems and Automation Society (ISA), www.isa.org

11. ISO 7498-1 Information Processing Systems - OSI Reference Model - The Basic Model
12. OPC Standard, www.opcfoundation.org
13. Profibus Standard, www.profibus.org
14. Wi-Fi Alliance, www.wi-fi.org
15. Wireless Industrial Networking Alliance (WINA), www.wina.org
16. ZigBee Alliance, www.zigbee.org

Online documents

1. Azimuth systems inc, www.azimuthsystems.com.
2. Chipcon Products, www.chipcon.com
3. CINT, root.cern.ch/twiki/bin/view/ROOT/CINT
4. DeltaV digital control system, www.easydeltav.com
5. DEMOJM Board, www.pemicro.com/fixedlinks/demoqetoolkit.cfm
6. FreeScaleMC1321,www.freescale.com/webapp/sps/site/prod_summary.jsp?code=1321xEVK
7. FreeScale MC1322, http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=1322x_Dev_Kits&parentCode=MC13224V
8. Freescale Coldfire, www.freescale.com/coldfire
9. Robustness tester for bluetooth, www.codenomicon.com
10. ZigBee Automated Compliance Test, www.seasolve.com

Index