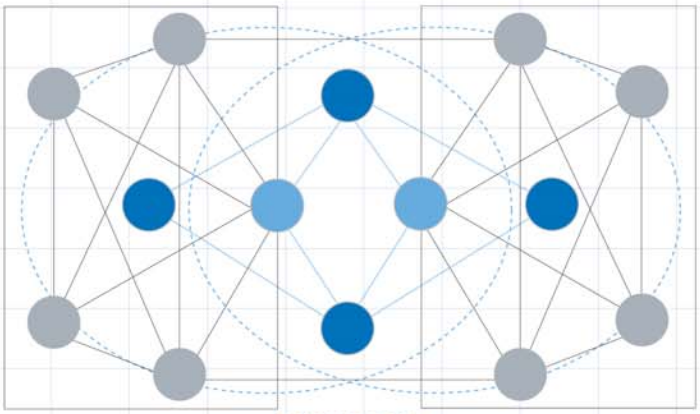
INSTEON

The Details



SMARTHOME technology

August 11, 2005

Table of Contents

Change Log

Date	Author	Change
07-28-05	Paul Darbee	Release for comment.
08-11-05	Paul Darbee	Release for publication.

Introduction

A TV automatically turns on the surround sound amplifier, a smart microwave oven downloads new cooking recipes, a thermostat automatically changes to its energy saving setpoint when the security system is enabled, bathroom floors and towel racks heat up when the bath runs, an email alert goes out when there is water in the basement. When did the Jetson-style home of the future become a reality? When INSTEON™—the new technology standard for advanced home control—arrived. INSTEON enables product developers to create these distinctive solutions for homeowners, and others yet unimagined, by delivering on the promise of a truly connected 'smart home.'

INSTEON is a cost-effective dual band network technology optimized for home management and control. INSTEON-networked Electronic Home Improvement™ products can interact with one another, and with people, in new ways that will improve the comfort, safety, convenience and value of homes around the world.

This white paper is for those who wish to learn more about INSTEON—why it was developed, how it works, and how to use it to create networked products for the modern home.

Developers of INSTEON-enabled products should not be too concerned about the level of detail presented here—a low-level 'INSTEON Engine' and higher-level interface software shield the programmer from having to remember it all. But by looking at the details, readers will gain an appreciation for the underlying motivations for INSTEON, and recognize why it is the right technology for home-control networking in the twenty-first century.

In this White Paper

INSTEON Overview

INSTEON enables simple, low-cost devices to be networked together using the powerline, radio frequency (RF), or both. All INSTEON devices are peers, meaning that any device can transmit, receive, or repeat[1] other messages, without requiring a master controller or complex routing software. Adding more devices makes an INSTEON network more robust, by virtue of a simple protocol for communication retransmissions and retries. On the powerline, INSTEON devices are compatible[2] with legacy X10 devices.

This section explains why INSTEON has these properties and explains them further without going into the details.

In this section

Why INSTEON?

INSTEON is the creation of Smarthome, the world's leading authority on home automation. With its three divisions, Smarthome.com, "the Amazon of electronic home improvement"; Smarthome Design, delivering best-in-class home control products; and Smarthome Technology, the pioneering architects of INSTEON, Smarthome is uniquely positioned with a global distribution channel, product development and manufacturing, and ongoing technology innovation to support third-party developers.

But why did Smarthome undertake the complex task of creating an entirely new home-control networking technology in the first place?

Smarthome has been a leading supplier of devices and equipment to home automation installers and enthusiasts since 1992. Now selling over 5,000 products into more than 130 countries, Smarthome has first-hand experience dealing directly with people all over the world who have installed lighting control, whole-house automation, security and surveillance systems, pet care devices, gadgets, and home entertainment equipment. Over the years, by talking to thousands of customers through its person-to-person customer support operation, Smarthome has become increasingly concerned about the mismatch between the dream of living in a responsive, aware, automated home and the reality of existing home-control technologies.

Today's homes are stuffed with high-tech appliances, entertainment gear, computers, and communications gadgets. Utilities, such as electricity, lighting, plumbing, heating and air conditioning are so much a part of modern life that they almost go unnoticed. But these systems and devices all act independently of each other—there still is nothing that can link them all together. Houses don't know that people live in them. Lights happily burn when no one needs them, HVAC is insensitive to the location and comfort of people, pipes can burst without anyone being notified, and sprinklers dutifully water the lawn even while it's raining.

For a collection of independent objects to behave with a unified purpose, the objects must be able to communicate with each other. When they do, new, sometimes-unpredictable properties often emerge. In biology, animals emerged when nervous systems evolved. The Internet emerged when telecommunications linked computers together. The global economy emerges from transactions involving a staggering amount of communication. But there is no such communicating infrastructure in our homes out of which we might expect new levels of comfort, safety and convenience to emerge. There is nothing we use routinely in our homes that links our light switches or our door locks, for instance, to our PCs or our remote controls.

It's not that such systems don't exist at all. Just as there were automobiles for decades before Henry Ford made cars available to everyone, there are now and have been for some time systems that can perform home automation tasks. On the high end, all kinds of customized systems are available for the affluent, just as the rich could buy a Stanley Steamer or a Hupmobile in the late 1800s. At the low end, X10 powerline signaling technology has been around since the 1970s, but its early adoption is its limiting factor—it is too unreliable and inflexible to be useful as an infrastructure network.

Smarthome is a major distributor of devices that use X10 signaling. In 1997, aware of the reliability problems its customers were having with X10 devices available at the time, Smarthome developed and began manufacturing its own 'Linc' series of

improved X10 devices, including controllers, dimmers, switches, computer interfaces and signal boosters. Despite the enhanced performance enjoyed by Linc products, it was still mostly do-it-yourselfers and hobbyists who were buying and installing them.

Smarthome knew that a far more robust and flexible networking standard would have to replace X10 before a truly intelligent home could emerge. Smarthome wanted a technology that would meet the simplicity, reliability, and cost expectations of the masses—mainstream consumers who want immediate benefits, not toys.

In 2001, Smarthome's engineers were well aware of efforts by others to bring about the home of the future. The aging X10 protocol was simply too limiting with its tiny command set and unacknowledged, 'press and pray' signaling over the powerline. CEBus had tried to be everything to everybody, suffering from high cost due to overdesign by a committee of engineers. Although CEBus did become an official standard (EIA-600), developers did not incorporate it into real-world products.

Radio-only communication protocols, such as Z-Wave and ZigBee, not only required complex routing strategies and a confusing array of different types of network masters, slaves, and other modules, but radio alone might not be reliable enough when installed in metal switch junction boxes or other RF-blocking locations.

BlueTooth radio has too short a range, WiFi radio is too expensive, and high-speed powerline protocols are far too complex to be built into commodity products such as light switches, door locks, or thermostats. Overall, it seemed that everything proposed or available was too overdesigned and therefore would cost too much to become a commodity for the masses in the global economy.

So, in 2001, Smarthome decided to take its destiny into its own hands and set out to specify an ideal home control network, one that would be simple, robust and inexpensive enough to link everything to everything else. INSTEON was born.

Hallmarks of INSTEON

These are the project pillars that Smarthome decided upon to guide the development of INSTEON. Products networked with INSTEON had to be:

Instantly Responsive

INSTEON devices respond to commands with no perceptible delay. INSTEON's signaling speed is optimized for home control—fast enough for quick response, while still allowing reliable networking using low-cost components.

Easy to Install

Installation in existing homes does not require any new wiring, because INSTEON products communicate over powerline wires or they use the airwaves. Users never have to deal with network enrollment issues because all INSTEON devices have an ID number pre-loaded at the factory—INSTEON devices join the network as soon as they're powered up.

Simple to Use

Getting one INSTEON device to control another is very simple—just press and hold a button on each device for 10 seconds, and they're linked. Because commands are confirmed, INSTEON products can provide instant feedback to the user, making them straightforward to use and 'guest friendly.'

Reliable

An INSTEON network becomes more robust and reliable as it is expanded because every INSTEON device repeats[1] messages received from other INSTEON devices. Dual band communications using both the powerline and the airwaves ensures that there are multiple pathways for messages to travel.

Affordable

INSTEON software is simple and compact, because all INSTEON devices send and receive messages in exactly the same way, without requiring a special network controller or complex routing algorithms. The cost of networking products with INSTEON is held to at an absolute minimum because INSTEON is designed specifically for home control applications, and not for transporting large amounts of data.

Compatible with X10

INSTEON and X10 signals can coexist with each other on the powerline without mutual interference. Designers are free to create hybrid INSTEON/X10 products that operate equally well in both environments, allowing current users of legacy X10 products to easily upgrade to INSTEON without making their investment in X10 obsolete.

INSTEON Specifications

The most important property of INSTEON is its no-frills simplicity.

INSTEON messages are fixed in length and synchronized to the AC powerline zero crossings. They do not contain routing information beyond a source and destination address. INSTEON is reliable and affordable because it is optimized for command and control, not high-speed data transport. INSTEON allows infrastructure devices like light switches and sensors to be networked together in large numbers, at low cost. INSTEON stands on its own, but can also bridge to other networks, such as WiFi LANs, the Internet, telephony, and entertainment distribution systems. Such bridging allows INSTEON to be part of very sophisticated integrated home control environments.

The following table shows the main features of INSTEON at a glance.

INSTEON Property	Specification	
Network	Dual band (RF and powerline) Peer-to-Peer Mesh Topology Unsupervised No routing tables	
Protocol	All devices are two-way Repeaters[1] Messages acknowledged Retry if not acknowledged Synchronized to powerline	
X10 Compatibility[2]	INSTEON devices can send and receive X10 commands INSTEON devices do not repeat or amplify X10	
Data Rate	Instantaneous	13,165 bits/sec
	Sustained	2,880 bits/sec
Message Types	Standard	10 Bytes
	Extended	24 Bytes
Message Format, Bytes	From Address	3
	To Address	3
	Flags	1
	Command	2
	User Data	14 (Extended Messages only)
	Message Integrity	1
Devices Supported	Unique IDs	16,777,216
	Device Types	65,536
	Commands	65,536
	Groups per Device	256
	Members within a Group	Limited only by memory
INSTEON Engine Memory Requirements	RAM	80 Bytes
	ROM	3 KBytes

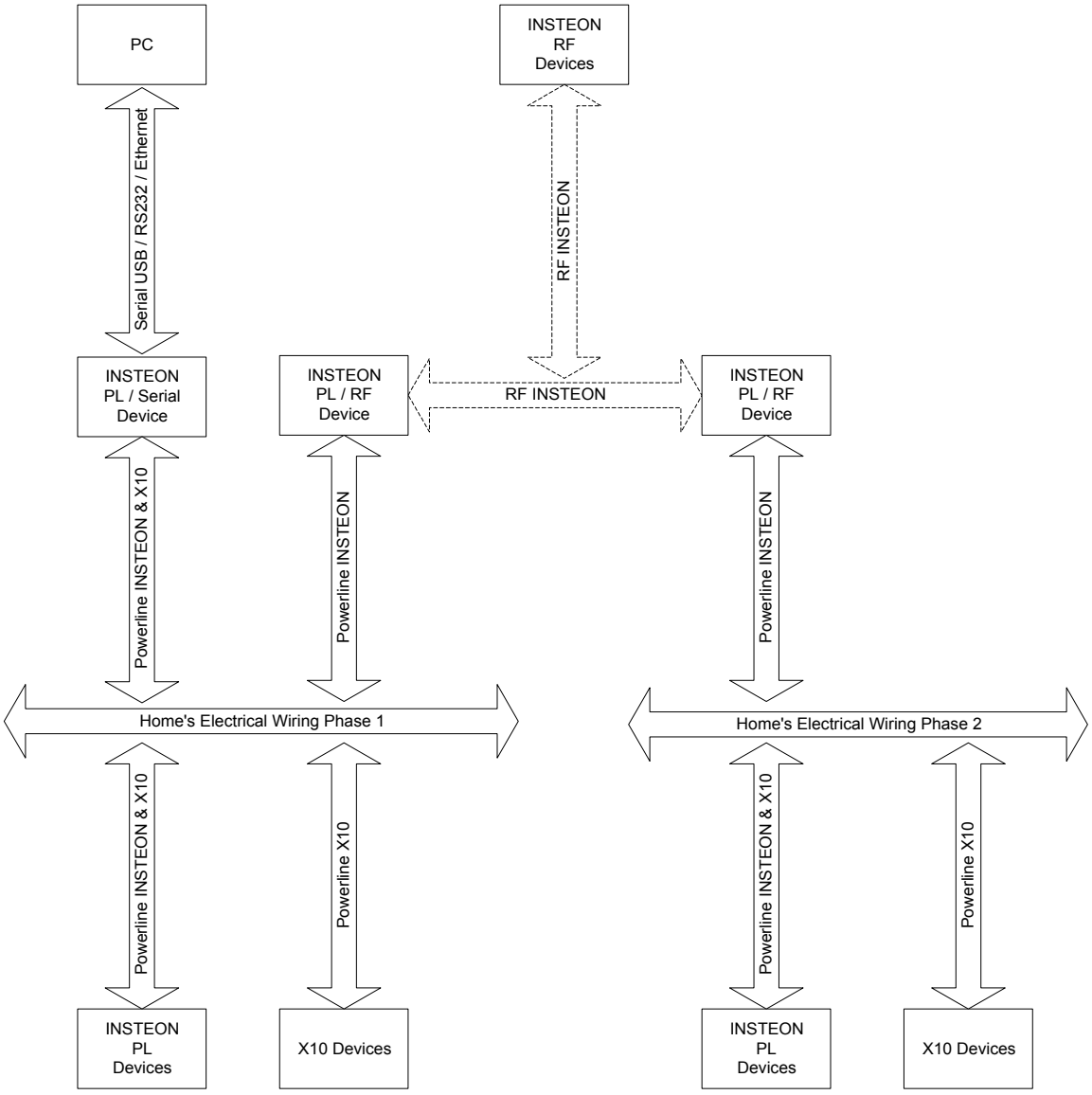
INSTEON Property	Specification	
Typical Application (Light Switch, Lamp Dimmer) Memory Requirements	RAM	256 Bytes
	EEPROM	256 Bytes
	Flash	7 Kbytes
Device Installation	Plug-in Wire-in Battery Operated	
Device Setup	Plug-n-Tap™ manual linking PC or Controller	
Security	Physical device possession Address masking Encrypted message payloads	
Application Development	IDE (Integrated Development Environment) SALad interpreted language Software and Hardware Development Kits	
Powerline Physical Layer	Frequency	131.65 KHz
	Modulation	BPSK
	Min Transmit Level	3.16 Vpp into 5 Ohms
	Min Receive Level	10 mV
	Phase Bridging	INSTEON RF or hardware
RF Physical Layer	Frequency	904 MHz
	Modulation	FSK
	Sensitivity	-103 dbm
	Range	150 ft unobstructed line-of-sight

INSTEON Fundamentals

This section covers *INSTEON Device Communication*, *INSTEON Message Repeating*, and *INSTEON Peer-to-Peer Networking*.

INSTEON Device Communication

Devices communicate with each other using the INSTEON protocol over the air via radio frequency (RF) and over the powerline as illustrated below.



Electrical power is most commonly distributed to homes in North America as split-phase 220-volt alternating current (220 VAC). At the main electrical junction box to the home, the single three-wire 220 VAC powerline is split into a pair of two-wire

110 VAC powerlines, known as Phase 1 and Phase 2. Phase 1 wiring usually powers half the circuits in the home, and Phase 2 powers the other half.

INSTEON devices communicate with each other over the powerline using the INSTEON Powerline protocol, which will be described in detail below (see *INSTEON Messages* and *INSTEON Signaling Details*).

Existing X10 devices also communicate over the powerline using the X10 protocol. The INSTEON Powerline protocol is compatible[2] with the X10 protocol, meaning that designers can create INSTEON devices that can also listen and talk to X10 devices. X10 devices, however, are insensitive to the INSTEON Powerline protocol.

INSTEON devices containing RF hardware may optionally communicate with other INSTEON RF devices using the INSTEON RF protocol.

INSTEON devices that can use *both* the INSTEON Powerline protocol and the INSTEON RF protocol solve a significant problem encountered by devices that can only communicate via the powerline. Powerline signals originating on the opposite powerline phase from a powerline receiver are severely attenuated, because there is no direct circuit connection for them to travel over.

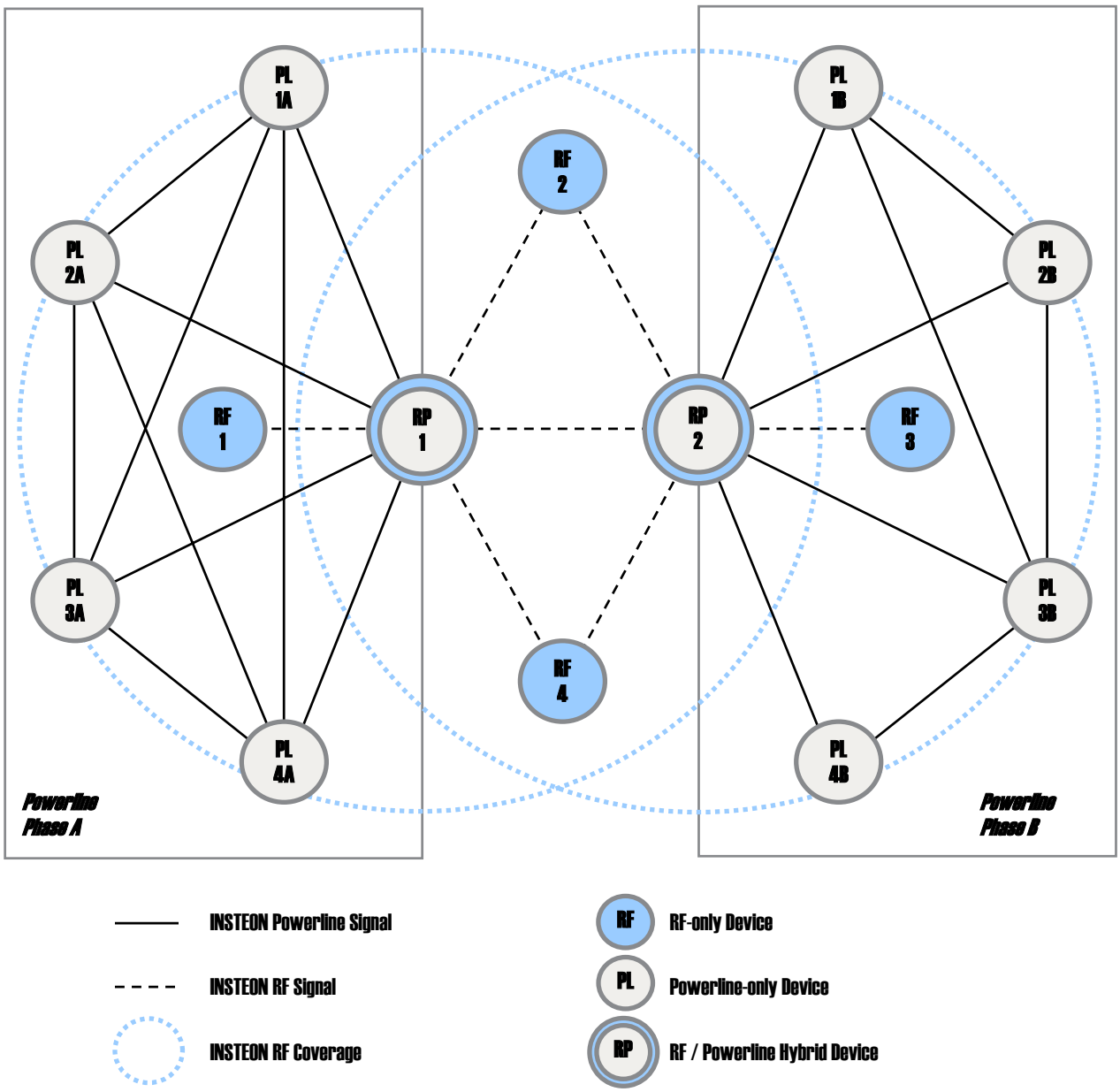
A traditional solution to this problem is to connect a signal coupling device between the powerline phases, either by hardwiring it in at a junction box or by plugging it into a 220 VAC outlet. INSTEON automatically solves the powerline phase coupling problem through the use of INSTEON devices capable of both powerline and RF messaging. INSTEON RF messaging bridges the powerline phases whenever at least one INSTEON PL/RF device is installed on each powerline phase.

When suitably equipped with a dedicated serial interface, such as USB, RS232, or Ethernet, INSTEON devices can also interface with computers and other digital equipment. In the figure above, an INSTEON device is shown communicating with a PC using a serial link.

Serial communications can bridge networks of INSTEON devices to otherwise incompatible networks of devices in a home, to computers, to other nodes on a local-area network (LAN), or to the global Internet. Such connections to outside resources allow networks of INSTEON devices to exhibit complex, adaptive, people-pleasing behaviors. INSTEON devices capable of running downloadable SALad Applications (see *SALad Applications*) can be upgraded to perform very sophisticated functions, including functions not envisioned at the time of manufacture or installation.

INSTEON Message Repeating

The figure below shows how network reliability improves when additional INSTEON devices are added. The drawing shows INSTEON devices that communicate by powerline-only (PL), RF-only (RF), and both (RP).



Every INSTEON device is capable of repeating[1] INSTEON messages. They will do this automatically as soon as they are powered up—they do not need to be specially installed using some network setup procedure. Adding more devices increases the number of available pathways for messages to travel. This *path diversity* results in a higher probability that a message will arrive at its intended destination, so the more devices in an INSTEON network, the better.

As an example, suppose RF device RF1 desires to send a message to RF3, but RF3 is out of range. The message will still get through, however, because devices within range of RF1, say RP1 and RF2, will receive the message and retransmit it to other devices within range of themselves. In the drawing, RP1 might reach RF2, RP2, and RF4, and devices RP2 and RF1 might be within range of the intended recipient, RF3. Therefore, there are many ways for a message to travel: RF1 to RF2 to RF3 (1 retransmission), RF1 to RP1 to RP2 to RF3 (2 retransmissions), and RF1 to RP1 to RF2 to RP2 to RF3 (3 retransmissions) are some examples.

On the powerline, path diversity has a similar beneficial effect. For example, the drawing shows powerline device PL1B without a direct communication path to device PL4B. In the real world, this might occur because of signal attenuation problems or because a direct path through the electric wiring does not exist. But a message from PL1B will still reach PL4B by taking a path through RP2 (1 retransmission), through PL2B to RP2 (2 retransmissions), or through PL2B to RP2 to PL3B (3 retransmissions).

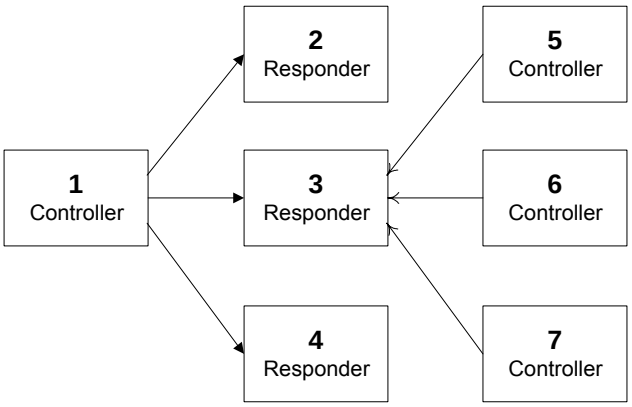
The figure also shows how messages can travel among powerline devices that are installed on different phases of a home's wiring. To accomplish phase bridging, at least one INSTEON hybrid RF/powerline device must be installed on each powerline phase. In the drawing, hybrid device RP1 is installed on phase A and RP2 is installed on phase B. Direct RF paths between RP1 to RP2, or indirect paths using RF2 or RF4 (1 retransmission) allow messages to propagate between the powerline phases, even though there is no direct electrical connection.

With all devices repeating messages, there must be some mechanism for limiting the number of times that a message may be retransmitted, or else messages might propagate forever within the network. Network saturation by repeating messages is known as a 'data storm.' The INSTEON protocol avoids this problem by limiting the maximum number times an individual message may be retransmitted to three (see *INSTEON Message Hopping*).

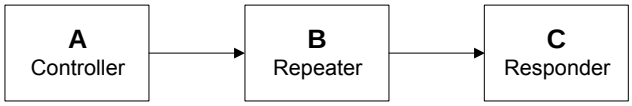
INSTEON Peer-to-Peer Networking

All INSTEON devices are peers, meaning that any device can act as a Controller (sending messages), Responder (receiving messages), or Repeater[1] (relaying messages).

This relationship is illustrated in the figure below, where INSTEON device **1**, acting as a Controller, sends messages to multiple INSTEON devices **2**, **3**, and **4** acting as Responders. Multiple INSTEON devices **5**, **6**, and **7** acting as Controllers can also send messages to a single INSTEON device **3** acting as a Responder.



Any INSTEON device can repeat[1] messages, as with device **B**, below, which is shown relaying a message from device **A** acting as a Controller to device **C** acting as a Responder.



INSTEON Applications and Devices

INSTEON technology can be built into many different kinds of equipment, and it can interface with all kinds of non-INSTEON equipment via communications bridges. Applications for INSTEON-enabled equipment are diverse.

A partial list of applications includes:

- Home management, incorporating sensors, heating ventilating and air-conditioning (HVAC), appliances, lighting, and security systems
- Audio/video (A/V) remote control, with tie-ins to home management
- Energy management
- Remote monitoring via the Internet
- Interoperation with voice recognition and response, cameras and other sensors

Products that can be improved using INSTEON technology include:

- Electrical devices such as plug-in or wire-in dimmers, switches, or outlets
- Home appliances
- Annunciators, thermostats, access controllers
- Pool/spa and irrigation controllers
- Environmental, device status, motion, room occupancy, or contact sensors
- PC, touchscreen, keypad, handheld, or keyfob controllers

Using bridges to other networking standards, INSTEON devices can interoperate with other devices communicating using WiFi (IEEE 802.11), BlueTooth (IEEE 802.15.1), ZigBee (IEEE 802.15.4), Z-Wave, WiMax (IEEE 802.16), HomePlug, HomeRF, Intellon, Echelon Lonworks, CEBus (EIA-600), or other future technology.

A network of INSTEON devices with at least one device having USB, RS232, or Ethernet communications capabilities can connect to a PC or to the Internet. New software can be downloaded to many INSTEON devices (see *SALad Applications*), making them capable of being upgraded with new capabilities in the future. An infrastructure of low-cost, reliable, upgradeable INSTEON-enabled devices capable of being connected to the larger world can bring so many benefits to people that it is not possible to foresee them all.

INSTEON Messages

INSTEON devices communicate by sending messages to one another. In the interest of maximum simplicity, there are only two kinds of INSTEON messages: 10-byte Standard messages and 24-byte Extended messages. The only difference between the two is that Extended messages carry 14 bytes of arbitrary User Data. They both carry a From Address, a To Address, a Flag byte, two Command bytes, and a Message Integrity byte.

In this section

INSTEON Message Structure
Gives the details about the contents of the various fields in INSTEON messages.

INSTEON Message Summary
Gives a single table showing the usage of all of the fields in all possible INSTEON message types. Recaps the usage of all of the different message types.

INSTEON Message Repetition
Explains how all INSTEON devices engage in retransmitting each other's messages so that an INSTEON network will become more reliable as more devices are added.

INSTEON Message Structure

INSTEON devices communicate with each other by sending fixed-length messages. This section describes the two *Message Lengths* (Standard and Extended) and explains the contents of the *Message Fields* within the messages.

Message Lengths

There are only two kinds of INSTEON messages, 10-byte Standard Length Messages and 24-byte Extended Length Messages.

The only difference between the two is that the Extended message contains 14 User Data Bytes not found in the Standard message. The remaining information fields for both types of message are identical.

INSTEON Standard Message – 10 Bytes				
3 Bytes	3 Bytes	1 Byte	2 Bytes	1 Byte
From Address	To Address	Flags	Command 1, 2	CRC[3]

INSTEON Extended Message – 24 Bytes					
3 Bytes	3 Bytes	1 Byte	2 Bytes	14 Bytes	1 Byte
From Address	To Address	Flags	Command 1, 2	User Data	CRC[3]

Standard Message

Standard messages are designed for direct command and control. The payload is just two bytes, Command 1 and Command 2.

Data		**Bits**	**Contents**
From Address		24	Message Originator's address
To Address		24	For Direct messages: Intended Recipient's address For Broadcast messages: Device Type, Subtype, Firmware Version For Group Broadcast messages: Group Number [0 - 255]
Message Flags	Message Type	1	Broadcast/NAK
		1	Group
		1	Acknowledge
	Extended Flag	1	**0 (Zero) for Standard messages**
	Hops Left	2	Counted down on each retransmission
	Max Hops	2	Maximum number of retransmissions allowed
Command 1		8	Command to execute
Command 2		8	
CRC[3]		8	Cyclic Redundancy Check

Extended Message

In addition to the same fields found in Standard messages, Extended messages carry 14 bytes of arbitrary User Data for downloads, uploads, encryption, and advanced applications.

Data		Bits	Contents
From Address		24	Message Originator's address
To Address		24	For Direct messages: Intended Recipient's address For Broadcast messages: Device Type, Subtype, Firmware Version For Group Broadcast messages: Group Number [0 - 255]
Message Flags	Message Type	1	Broadcast/NAK
		1	Group
		1	Acknowledge
	Extended Flag	1	**1 (One) for Extended messages**
	Hops Left	2	Counted down on each retransmission
	Max Hops	2	Maximum number of retransmissions allowed
Command 1		8	Command to execute
Command 2		8	
User Data 1		8	User defined data
User Data 2		8	
User Data 3		8	
User Data 4		8	
User Data 5		8	
User Data 6		8	
User Data 7		8	
User Data 8		8	
User Data 9		8	
User Data 10		8	
User Data 11		8	
User Data 12		8	
User Data 13		8	
User Data 14		8	
CRC[3]		8	Cyclic Redundancy Check

Message Fields

All INSTEON messages contain source and destination *Device Addresses*, a *Message Flags* byte, a 2-byte *Command 1 and 2* payload, and a *Message Integrity Byte*. INSTEON Extended messages also carry 14 bytes of *User Data*.

Device Addresses

The first field in an INSTEON message is the *From Address*, a 24-bit (3-byte) number that uniquely identifies the INSTEON device originating the message being sent. There are 16,777,216 possible INSTEON devices identifiable by a 3-byte number. This number can be thought of as an ID Code or, equivalently, as an address for an INSTEON device. During manufacture, a unique ID Code is stored in each device in nonvolatile memory.

The second field in an INSTEON message is the *To Address*, also a 24-bit (3-byte) number. Most INSTEON messages are of the *Direct*, or *Point-to-Point* (P2P), type, where the intended recipient is another single, unique INSTEON device.

If the message is indeed Direct (as determined by the Flags Byte), the To Address contains the 3-byte unique ID Code for the intended recipient. However, INSTEON messages can also be sent to all recipients within range, as *Broadcast* messages, or they can be sent to all members of a group of devices, as *Group Broadcast* messages. In the case of Broadcast messages, the To Address field contains a 2-byte *Device Type* and a *Firmware Version* byte. For Group Broadcast messages, the To Address field contains a Group Number. Group Numbers only range from 0 to 255, given by one byte, so the two most-significant bytes of the three-byte field will be zero.

Message Flags

The third field in an INSTEON message, the Message Flags byte, not only signifies the Message Type but it also contains other information about the message. The three most-significant bits, the Broadcast/NAK flag (bit 7), the Group flag (bit 6), and the ACK flag (bit 5) together indicate the Message Type. Message Types will be explained in more detail in the next section (see *Message Types*). Bit 4, the Extended flag, is set to one if the message is an Extended message, i.e. contains 14 User Data bytes, or else it is set to zero if the message is a Standard message. The low nibble contains two two-bit fields, Hops Left (bits 3 and 2) and Max Hops (bits 1 and 0). These two fields control message retransmission as explained below (see *Message Retransmission Flags*).

The table below enumerates the meaning of the bit fields in the Message Flags byte. The Broadcast/NAK flag (bit 7, the most-significant byte), the Group flag (bit 6), and the ACK flag (bit 5) together denote the eight possible Message Types.

Bit Position	Flag	Meaning
Bit 7 (Broadcast /NAK) (MSB)	Message Type	100 = Broadcast Message 000 = Direct Message 001 = ACK of Direct Message 101 = NAK of Direct Message 110 = Group Broadcast Message 010 = Group Cleanup Direct Message 011 = ACK of Group Cleanup Direct Message 111 = NAK of Group Cleanup Direct Message
Bit 6 (Group)		
Bit 5 (Acknowledge)		
Bit 4	Extended	1 = Extended Message 0 = Standard Message
Bit 3	Hops Left	00 = 0 message retransmissions remaining 01 = 1 message retransmission remaining 10 = 2 message retransmissions remaining 11 = 3 message retransmissions remaining
Bit 2		
Bit 1	Max Hops	00 = Do not retransmit this message 01 = Retransmit this message 1 time maximum 10 = Retransmit this message 2 times maximum 11 = Retransmit this message 3 times maximum
Bit 0 (LSB)		

Message Type Flags

There are eight possible INSTEON *Message Types* given by the three *Message Type Flag Bits*.

Message Types

To fully understand the eight Message Types, consider that there are four basic classes of INSTEON messages: *Broadcast*, *Group Broadcast*, *Direct*, and *Acknowledge*.

Broadcast messages contain general information with no specific destination. Directed to the community of all devices within range, they are used extensively during device linking (see *Device Identification Broadcast*, below). Broadcast messages are not acknowledged.

Group Broadcast messages are directed to a group of devices that have previously been linked to the message originator (see *INSTEON Groups*, below). Group Broadcast messages are not acknowledged directly. They only exist as a means for speeding up the response to a command intended for multiple devices. After sending a Group Broadcast message to a group of devices, the message originator then sends a Direct 'Group Cleanup' message to each member of the group individually, and waits for an acknowledgement back from each device.

Direct messages, also referred to as Point-to-Point (P2P) messages, are intended for a single specific recipient. The recipient responds to Direct messages by returning an Acknowledge message.

Acknowledge messages (ACK or NAK) are messages from the recipient to the message originator in response to a Direct message. There is no acknowledgement

to a Broadcast or Group Broadcast message. An ACK or NAK message may contain status information from the acknowledging device.

Message Type Flag Bits

The Broadcast/NAK flag will be set whenever the message is a Broadcast message or a Group Broadcast message. In those two cases the Acknowledge flag will be clear. If the Acknowledge flag is set, the message is an Acknowledge message. In that case the Broadcast/NAK flag will be set when the Acknowledge message is a NAK, and it will be clear when the Acknowledge message is an ACK.

The Group flag will be set to indicate that the message is a Group Broadcast message or part of a Group Cleanup conversation. This flag will be clear for general Broadcast messages and Direct conversations.

Now all eight Message Types can be enumerated as follows, where the three-bit field is given in the order Bit 7, Bit 6, Bit 5.

- Broadcast messages are Message Type 100.
- Direct (P2P) messages are 000.
- An ACK of a Direct message is 001
- A NAK of a Direct message is 101
- A Group Broadcast message is 110.
- Group Broadcasts are followed up by a series of Group Cleanup Direct messages of Message Type 010 to each member of the group.
- Each recipient of a Group Cleanup Direct message will return an acknowledgement with a Group Cleanup ACK of Message Type 011 or a Group Cleanup NAK of Message Type 111.

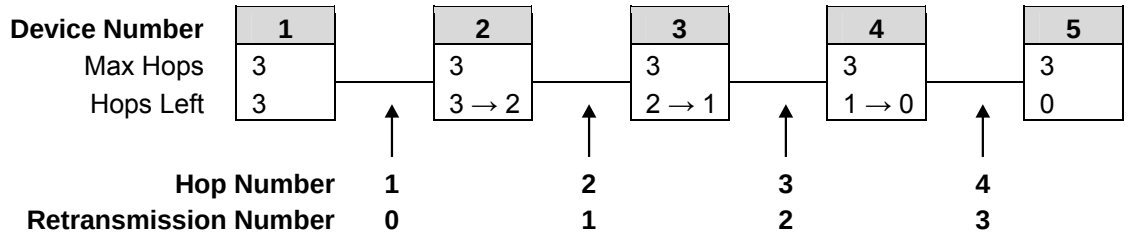
Extended Message Flag

Bit 4 is the Extended Message flag. This flag is set for 24-byte Extended messages that contain a 14-byte User Data field, and the flag is clear for 10-byte Standard messages that do not contain User Data.

Message Retransmission Flags

The remaining two flag fields, Max Hops and Hops Left, manage message retransmission. As described above, all INSTEON devices are capable of repeating[1] messages by receiving and retransmitting them. Without a mechanism for limiting the number of times a message can be retransmitted, an uncontrolled 'data storm' of endlessly repeated messages could saturate the network. To solve this problem, INSTEON message originators set the 2-bit Max Hops field to a value of 0, 1, 2, or 3, and they also set the 2-bit Hops Left field to the same value. A Max Hops value of zero tells other devices within range not to retransmit the message. A higher Max Hops value tells devices receiving the message to retransmit it depending on the Hops Left field. If the Hops Left value is one or more, the receiving device decrements the Hops Left value by one, then retransmits the message with the new Hops Left value. Devices that receive a message with a Hops Left value of zero will not retransmit that message. Also, a device that is the intended recipient of a message will not retransmit the message, no matter what the Hops Left value is. See *INSTEON Message Hopping* for more information.

Note that the designator 'Max Hops' really means maximum *retransmissions* allowed. All INSTEON messages 'hop' at least once, so the value in the Max Hops field is one less than the number of times a message actually hops from one device to another. Since the maximum value in this field is three, there can be four actual hops, consisting of the original transmission and three retransmissions. Four hops can span a chain of five devices. This situation is shown schematically below.



Command 1 and 2

The fourth field in an INSTEON message is a two-byte Command, made up of *Command 1* and *Command 2*. The usage of this field depends on the Message Type as explained below (see *INSTEON Commands*).

User Data

Only if the message is an Extended message, with the Extended Flag set to one, will it contain the fourteen-byte *User Data* field. User Data can be arbitrarily defined.

If more than 14 bytes of User Data need to be transmitted, multiple INSTEON Extended messages will have to be sent. Users can define a packetizing method for their data so that a receiving device can reliably reassemble long messages. Encrypting User Data can provide private, secure communications for sensitive applications such as security systems. See *INSTEON Extended Messages*, below, for more information.

Message Integrity Byte

The last field in an INSTEON message is a one-byte CRC, or Cyclic Redundancy Check. The INSTEON transmitting device computes the CRC over all the bytes in a message beginning with the From Address. INSTEON uses a software-implemented 7-bit linear-feedback shift register with taps at the two most-significant bits. The CRC covers 9 bytes for Standard messages and 23 bytes for Extended messages. An INSTEON receiving device computes its own CRC over the same message bytes as it receives them. If the message is corrupt, the receiver's CRC will not match the transmitted CRC.

Firmware in the INSTEON Engine handles the CRC byte automatically, appending it to messages that it sends, and comparing it within messages that it receives. Applications post messages to and receive messages from the INSTEON Engine without the CRC byte being appended.

Detection of message integrity allows for highly reliable, verified communications. The INSTEON ACK/NAK (acknowledge, non-acknowledge) closed-loop messaging protocol based on this detection method is described below (see *INSTEON Message Retrying*).

INSTEON Message Summary

The table below summarizes all the fields in every type of INSTEON message. Standard messages are enumerated at the top and Extended messages are enumerated at the bottom. The figure clearly shows that the only difference between Standard and Extended messages is that the Extended Flag is clear for Standard messages and set for Extended messages, and Extended messages possess a 14-byte User Data field. The From Address, the To Address, the Message Flags, and the CRCs are as explained above.

Message		3 Bytes			3 Bytes			1 Byte						1 Byte	1 Byte	1 Byte
		From Address			To Address			Message Flags						Cmd 1	Cmd 2	CRC[3]
								Type			X	HL	MH			
Standard	Broadcast	ID1_2	ID1_1	ID1_0	Type	Subtype	Revision	1	0	0	0	Message retransmissions left	Maximum message retransmissions allowed	Broadcast Cmd		CRC
	Broadcast Group	ID1_2	ID1_1	ID1_0	0	0	Group #	1	1	0	0			Gp Cmd	Param	CRC
	P2P Cleanup	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	1	0	0			Gp Cmd	Group #	CRC
	P2P Cleanup ACK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	1	1	0			Gp Cmd	Status	CRC
	P2P Cleanup NAK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	1	1	1	0			Gp Cmd	Reason	CRC
	P2P	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	0	0	0			Direct Cmd		CRC
	P2P ACK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	0	1	0			ACK Status		CRC
	P2P NAK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	1	0	1	0			NAK Reason		CRC
															14 Bytes	1 Byte
															D1 ⇒ D14	CRC[3]
Extended	Broadcast	ID1_2	ID1_1	ID1_0	Type	Subtype	Revision	1	0	0	1			Broadcast Cmd		D1 ⇒ D14	CRC
	Broadcast Group	ID1_2	ID1_1	ID1_0	0	0	Group #	1	1	0	1			Gp Cmd	Param	D1 ⇒ D14	CRC
	P2P Cleanup	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	1	0	1			Gp Cmd	Group #	D1 ⇒ D14	CRC
	P2P Cleanup ACK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	1	1	1			Gp Cmd	Status	D1 ⇒ D14	CRC
	P2P Cleanup NAK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	1	1	1	1			Gp Cmd	Reason	D1 ⇒ D14	CRC
	P2P	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	0	0	1			Direct Cmd		D1 ⇒ D14	CRC
	P2P ACK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	0	0	1	1			ACK Status		D1 ⇒ D14	CRC
	P2P NAK	ID1_2	ID1_1	ID1_0	ID2_2	ID2_1	ID2_0	1	0	1	1			NAK Reason		D1 ⇒ D14	CRC
								Broadcast / NAK	Group	Acknowledge	Extended	Hops Left, 2 bits	Max Hops, 2 bits				

The Command 1 and Command 2 fields contain different information for each of the eight types of INSTEON messages. In the case of Broadcast messages, the two fields together contain a 2-byte command chosen from a possible 65,536 commands suitable for sending to all devices at once. For example, a device can identify itself to other devices by sending a *Set Button Pushed* Broadcast command (see *Device Identification Broadcast*). Every receiving device contains a database of Broadcast commands that it is capable of executing.

In the case of Point-to-Point (Direct) messages, the two Command fields together comprise a 2-byte command chosen from a possible 65,536 commands suitable for sending to a single device. For example, a Direct command could tell a lamp control device to turn on the lamp plugged into it. Every receiving device contains a database of Direct commands that it is capable of executing (see *INSTEON Commands*).

In the interest of maximum system reliability, the INSTEON protocol requires that Direct messages be acknowledged. A receiving device can issue an acknowledgement of successful communication and completion of a task, i.e. an ACK, or it can issue a NAK to indicate some kind of failure. If a receiving device fails to send an ACK or a NAK back to the originating device, the originating device will retry the message (see *INSTEON Message Retrying*).

To respond with an ACK or a NAK, firmware in a receiving device swaps the From Address and the To Address in the message it received, and sets the Message Type bits to 001 for an ACK or 101 for a NAK. Depending on the command received in the Command fields, the receiving device composes a two-byte status response code for an ACK or else a two-byte reason code for a NAK, which it inserts in the Command fields. For example, if a lamp dimmer receives a command to set the lamp to a certain brightness level, issued as a Set Brightness code in the Command 1 field and the desired brightness level as one of 256 values in the Command 2 field, the dimmer will respond with an ACK message containing the same two bytes in the Command fields to indicate successful execution of the command.

The remaining INSTEON message types are for dealing with groups of devices (see *INSTEON Groups*). Group Broadcast messages exist as a performance enhancement. While it is true that all the members of a group of devices could be sent individual Direct messages with the same command (to turn on, for example), it would take a noticeable amount of time for all the messages to be transmitted in sequence. The members of the group would not execute the command all at once, but rather in the order received. INSTEON solves this problem by first sending a Group Broadcast message, then following it up with individual Direct 'Group Cleanup' messages.

Group Broadcast messages contain a Group Number in the To Address field, a one-byte Group Command in the Command 1 field, and an optional one-byte parameter in the Command 2 field. During the Direct Group Cleanup messages that will follow, the Group Command will be sent in the Command 1 field and the Group Number will be sent in the Command 2 field. These are both one-byte fields, so there can only be 256 Group Commands and only 256 Group Numbers. This is a reasonable limitation given that Group Broadcasts only need to be used where rapid, synchronous response of multiple devices is an issue. In any case, the numerical limitation can be overcome by using Extended messages and embedding additional commands or group membership criteria in the User Data field.

Recipients of a Group Broadcast message check the Group Number in the To Address field against their own group memberships recorded in a Link Database (see *INSTEON Link Database*). This database, stored in nonvolatile memory, is established during a prior group enrollment, or *linking*, process (see *Methods for Linking INSTEON Devices*). If the recipient is a member of the Group being broadcast to, it executes the command in the Command 1 field. Since the Group Command only occupies one byte, the other byte in field can be a parameter or a subcommand.

Group Broadcast command recipients can expect a Direct individually-addressed Group Cleanup message to follow. If the recipient has already executed the Group Command, it will not execute the command a second time. However, if the recipient missed the Group Broadcast command for any reason, it will not have executed it, so it will execute the command after receiving the Direct Group Cleanup message.

After receiving the Direct Group Cleanup message and executing the Group Command, the recipient device will respond with a Group Cleanup ACK message, or else, if something went wrong, it will respond with a Group Cleanup NAK message. In both cases the Command 1 field will contain the same one-byte Group Command received during the Direct Group Cleanup message. The other byte in the Command 2 field will contain a one-byte ACK Status code in the case of an ACK, or a one-byte NAK Reason code in the case of a NAK. These one-byte codes are a subset of the corresponding two-byte codes used in Direct ACK and Direct NAK messages.

INSTEON Message Repetition

To maximize communications reliability, the INSTEON messaging protocol includes two kinds of message repetition: message hopping and message retrying.

INSTEON Message Hopping is the mechanism whereby INSTEON devices, all of which can retransmit INSTEON messages, aid each other in delivering a message from a message originator to a message recipient.

INSTEON Message Retrying occurs when the originator of a Direct message does not receive a proper acknowledgement message from the intended recipient.

INSTEON Message Hopping

In order to improve reliability, the INSTEON messaging protocol includes message retransmission, or hopping. Hopping enables other INSTEON devices, all of which can repeat[1] messages, to help relay a message from an originator to a recipient.

When INSTEON devices repeat messages, multiple devices can end up simulcasting the same message, meaning that they can repeat the same message at the same time. To ensure that simulcasting is synchronous (so that multiple devices do not jam each other), INSTEON devices adhere to specific rules given below (see *Timeslot Synchronization*).

Message Hopping Control

Two 2-bit fields in the Message Flags byte manage INSTEON message hopping (see *Message Retransmission Flags*, above). One field, *Max Hops*, contains the maximum number of hops, and the other, *Hops Left,* contains the number of hops remaining.

To avoid 'data storms' of endless repetition, messages can be retransmitted a maximum of three times only. A message originator sets the Max Hops for a message. The larger the number of Max Hops, the longer the message will take to complete being sent, whether or not the recipient hears the message early.

If the Hops Left field in a message is nonzero, every device that hears the message synchronously repeats it, thus increasing the signal strength, path diversity, and range of the message. An INSTEON device that repeats a message decrements Hops Left before retransmitting it. When a device receives a message with zero Hops Left, it does not retransmit the message.

Timeslot Synchronization

There is a specific pattern of transmissions, retransmissions and acknowledgements that occurs when an INSTEON message is sent, as shown in the examples below.

An INSTEON message on the powerline occupies either six or thirteen zero crossing periods, depending on whether the message is Standard or Extended. This message transmission time, six or thirteen powerline half-cycles, is called a *timeslot* in the following discussion. See *Message Timeslots*, below, for more details.

During a single timeslot, an INSTEON message can be transmitted, retransmitted, or acknowledged. The entire process of communicating an INSTEON message, which may involve retransmissions and acknowledgements, will occur over integer multiples of timeslots.

The following examples show how INSTEON messages propagate in a number of common scenarios. The examples use these symbols:

Legend		
	T	Transmission by Message Originator
	R	Message Retransmission
	A	Acknowledgement by Intended Recipient
	C	Confirmation received by Message Originator
	L	Listening State
	W	Waiting State

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 1	**0**	Sender	**T**							

Example 1, the simplest, shows a Broadcast message with a Max Hops of zero (no retransmissions). The **T** indicates that the Sender has originated and transmitted a single message. There is no acknowledgement that intended recipients have heard the message. The message required one timeslot of six or thirteen powerline zero crossings to complete.

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 2	**1**	Sender	**T**							
		Repeater 1	L	**R**						

Example 2 shows a Broadcast message with a Max Hops of one. Max Hops can range from zero to three as explained above. The Sender transmits a Broadcast message as signified by the **T**. Another INSTEON device, functioning as a Repeater, listens to the message, as signified by an **L**, and then retransmits it in the next timeslot as indicated by the **R**.

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 3	**3**	Sender	**T**	L	L	L	L			
		Repeater 1	L	**R**	L	**R**	L			
		Repeater 2	L	L	**R**	L	L			
		Repeater 3	L	L	L	**R**	L			

Up to three retransmissions are possible with a message. **Example 3** shows the progression of the message involving an originating Sender and three repeating devices, with a Max Hops of three. Example 3 assumes that the range between Repeaters is such that only adjacent Repeaters can hear each other, and that only Repeater 1 can hear the Sender. Note that the Sender will not retransmit its own message.

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 4	**0**	Sender	**T**	**C**						
		Recipient	L	**A**						

When a Sender transmits a Direct (Point-to-Point) message, it expects an acknowledgement from the Recipient. **Example 4** shows what happens if the Max Hops value is zero. The **A** designates the timeslot in which the Recipient acknowledges receipt of the Direct message. The **C** shows the timeslot when the Sender finds that the message is confirmed.

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 5	**1**	Sender	**T**	L	L	**C**				
		Repeater 1	L	**R**	L	**R**				
		Recipient	L	L	**A**	L				

When Max Hops is set to one, a Direct message propagates as shown in **Example 5**. Repeater 1 will retransmit both the original Direct message and the acknowledgement from the Recipient.

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 6	**1**	Sender	**T**	L	**C**	W				
		Repeater 1	L	**R**	L	**R**				
		Recipient	L	W	**A**	L				

If Max Hops is set to one, but no retransmission is needed because the Recipient is within range of the Sender, messages flow as shown in **Example 6**. The **W** in the Sender and Recipient rows indicates a wait. The Recipient immediately hears the Sender since it is within range. However, the Recipient must wait one timeslot before sending its acknowledgement, because it is possible that a repeating device will be retransmitting the Sender's message. Repeater 1 is shown doing just that in the example, although the Recipient would still have to wait even if no Repeaters were present. Only when all of the *possible* retransmissions of the Sender's message are complete, can the Recipient send its acknowledgement. Being within range, the Sender hears the acknowledgement immediately, but it must also wait until possible retransmissions of the acknowledgement are finished before it can send another message.

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 7	3	Sender	**T**	L	L	L	L	L	L	**C**
		Repeater 1	L	**R**	L	**R**	L	**R**	L	**R**
		Repeater 2	L	L	**R**	L	L	L	**R**	L
		Repeater 3	L	L	L	**R**	L	**R**	L	**R**
		Recipient	L	L	L	L	**A**	L	**R**	L

Example 7 shows what happens when Max Hops is three and three retransmissions are in fact needed for the message to reach the Recipient. Note that if the Sender or Recipient were to hear the other's message earlier than shown, it still must wait until Max Hops timeslots have occurred after the message was originated before being free to send its own message. If devices did not wait, they would jam each other by sending different messages in the same timeslot. A device can calculate how many timeslots have passed prior to receiving a message by subtracting the Hops Left number in the received message from the Max Hops number.

All seven of the above examples are given again in the table below in order to show the patterns more clearly.

	Max Hops	Timeslot	1	2	3	4	5	6	7	8
Example 1	**0**	Sender	**T**							
Example 2	**1**	Sender	**T**							
		Repeater 1	L	**R**						
Example 3	**3**	Sender	**T**	L	L	L	L			
		Repeater 1	L	**R**	L	**R**	L			
		Repeater 2	L	L	**R**	L	L			
		Repeater 3	L	L	L	**R**	L			
Example 4	**0**	Sender	**T**	**C**						
		Recipient	L	**A**						
Example 5	**1**	Sender	**T**	L	L	**C**				
		Repeater 1	L	**R**	L	**R**				
		Recipient	L	L	**A**	L				
Example 6	**1**	Sender	**T**	L	**C**	W				
		Repeater 1	L	**R**	L	**R**				
		Recipient	L	W	**A**	L				
Example 7	**3**	Sender	**T**	L	L	L	L	L	L	**C**
		Repeater 1	L	**R**	L	**R**	L	**R**	L	**R**
		Repeater 2	L	L	**R**	L	L	L	**R**	L
		Repeater 3	L	L	L	**R**	L	**R**	L	**R**
		Recipient	L	L	L	L	**A**	L	**R**	L

Legend	
T	Transmission by Message Originator
R	Message Retransmission
A	Acknowledgement by Intended Recipient
C	Confirmation received by Message Originator
L	Listening State
W	Waiting State

INSTEON Message Retrying

If the originator of an INSTEON Direct message does not receive an acknowledgement from the intended recipient, the message originator will automatically try resending the message up to five times.

In case a message did not get through because Max Hops was set too low, each time the message originator retries a message, it also increases Max Hops up to the limit of three. A larger number of Max Hops can achieve greater range for the message by allowing more devices to retransmit it.

Firmware in the INSTEON Engine handles message retrying. After using the INSTEON Engine to send a Direct message, applications will either receive the expected acknowledgement message or an indication that the intended recipient did not receive the Direct message after five retries.

Because message retrying is automatic, it is important to unlink INSTEON Responder devices from INSTEON Controller devices when a linked device is removed from an INSTEON network. See *INSTEON Link Database*, below, for more information.

INSTEON Signaling Details

This section gives complete information about how the data in INSTEON messages actually travels over the powerline or the airwaves. Unlike other mesh networks, INSTEON does not elaborately route its traffic in order to avoid data collisions—instead, INSTEON devices *simulcast* according to simple rules explained below. Simulcasting by multiple devices is made possible because INSTEON references a global clock, the powerline zero crossing.

In this section

INSTEON Packet Structure

This section describes *Powerline Packets* and *RF Packets*.

Powerline Packets

Messages sent over the powerline are broken up into packets, with each packet sent in conjunction with a zero crossing of the AC voltage on the powerline. Standard Messages use five packets and Extended Messages use eleven packets, as shown below.

Standard Message – 5 Packets

120 total bits = 15 bytes
84 Data bits = 10½ bytes, 10 usable

SP	BP	BP	BP	BP

Extended Message - 11 Packets

264 total bits = 33 bytes
192 Data bits = 24 bytes

SP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP

A Start Packet appears as the first packet in an INSTEON message, as shown by the symbol **SP** in both the Standard and Extended Messages. The remaining packets in a message are Body Packets, as shown by the symbols **BP**.

Each packet contains 24 bits of information, but the information is interpreted in two different ways, as shown below.

SP Start Packet

1	0	1	0	1	0	1	0	1	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x
8 Sync bits								4 Start Code bits				12 Data bits											

BP Body Packet

1	0	1	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
2 Sync bits		4 Start Code bits				18 Data Bits																	

Powerline packets begin with a series of *Sync Bits*. There are eight Sync Bits in a Start Packet and there are two Sync Bits in a Body Packet. The alternating pattern of ones and zeros allows the receiver to detect the presence of a signal.

Following the Sync Bits are four *Start Code Bits*. The 1001 pattern indicates to the receiver that Data bits will follow.

The remaining bits in a packet are *Data Bits*. There are twelve Data Bits in a Start Packet, and there are eighteen Data Bits in a Body Packet.

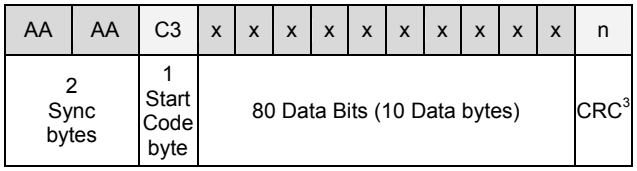
The total number of Data Bits in a Standard Message is 84, or 10½ bytes. The last four data bits in a Standard Message are ignored, so the usable data is 10 bytes. The total number of Data Bits in an Extended Message is 192, or 24 bytes.

RF Packets

The figure below shows the contents of INSTEON messages sent using RF. Because INSTEON RF messaging is much faster than powerline messaging, there is no need to break up RF messages into smaller packets. An RF Standard message and an RF Extended message are both shown. In both cases the message begins with two Sync Bytes followed by one Start Code Byte. RF Standard messages contain 10 Data Bytes (80 bits), and RF Extended messages contain 24 Data Bytes (192 bits).

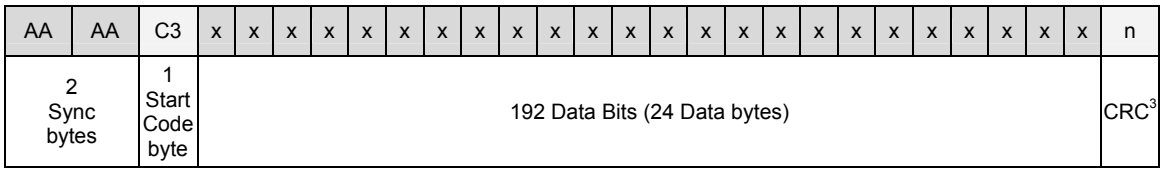
RF Standard Message – 1 Packet

112 total bits = 14 bytes
80 Data bits = 10 bytes



RF Extended Message – 1 Packet

224 total bits = 28 bytes
192 Data bits = 24 bytes



INSTEON Signaling

This section explains bit encoding for powerline and RF transmission, packet synchronization to the powerline, X10 compatibility[2], message timeslots, and data rates.

Powerline Signaling

INSTEON devices communicate on the powerline by adding a signal to the powerline voltage. In the United States, powerline voltage is nominally 110 VAC RMS, alternating at 60 Hz.

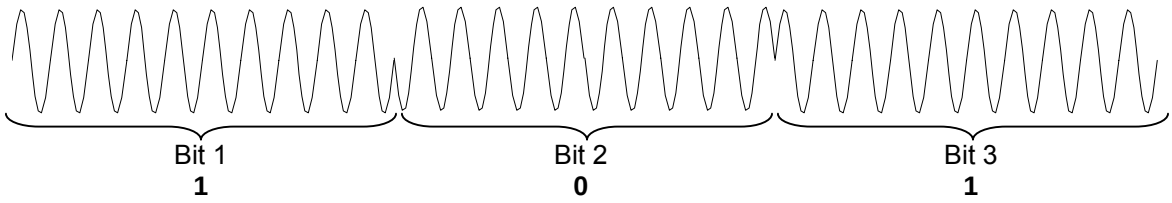
An INSTEON powerline signal uses a carrier frequency of 131.65 KHz, with a nominal amplitude of 4.64 volts peak-to-peak into a 5 ohm load. In practice, the impedance of powerlines varies widely, depending on the powerline configuration and what is plugged into it, so measured INSTEON powerline signals can vary from sub-millivolt to more than 5 volts.

INSTEON data is modulated onto the 131.65 KHz carrier using binary phase-shift keying, or BPSK, chosen for reliable performance in the presence of noise.

The bytes in an INSTEON powerline message are transmitted most-significant byte first, and the bits are transmitted most-significant bit first.

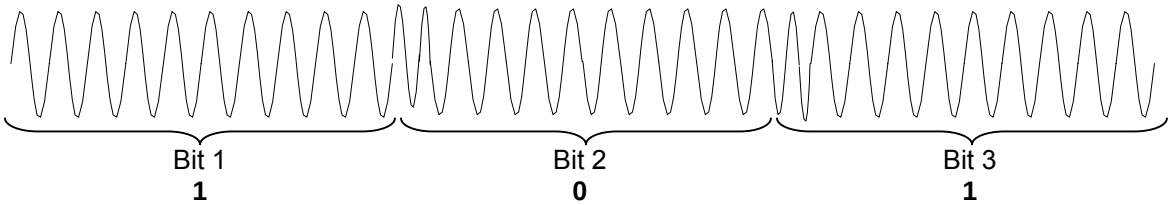
BPSK Modulation

The figure below shows an INSTEON 131.65 KHz powerline carrier signal with alternating binary phase-shift keying (BPSK) bit modulation.



INSTEON uses 10 cycles of carrier for each bit. Bit 1, interpreted as a one, begins with a positive-going carrier cycle. Bit 2, interpreted as a zero, begins with a negative-going carrier cycle. Bit 3 begins with a positive-going carrier cycle, so it is interpreted as a one. Note that the sense of the bit interpretations is arbitrary. That is, ones and zeros could be reversed as long as the interpretation is consistent. Phase transitions only occur when a bitstream changes from a zero to a one or from a one to a zero. A one followed by another one, or a zero followed by another zero, will not cause a phase transition. This type of coding is known as NRZ, or non-return to zero.

Note the abrupt phase transitions of 180 degrees at the bit boundaries. Abrupt phase transitions introduce troublesome high-frequency components into the signal's spectrum. Phase-locked detectors can have trouble tracking such a signal. To solve this problem, INSTEON uses a gradual phase change to reduce the unwanted frequency components.

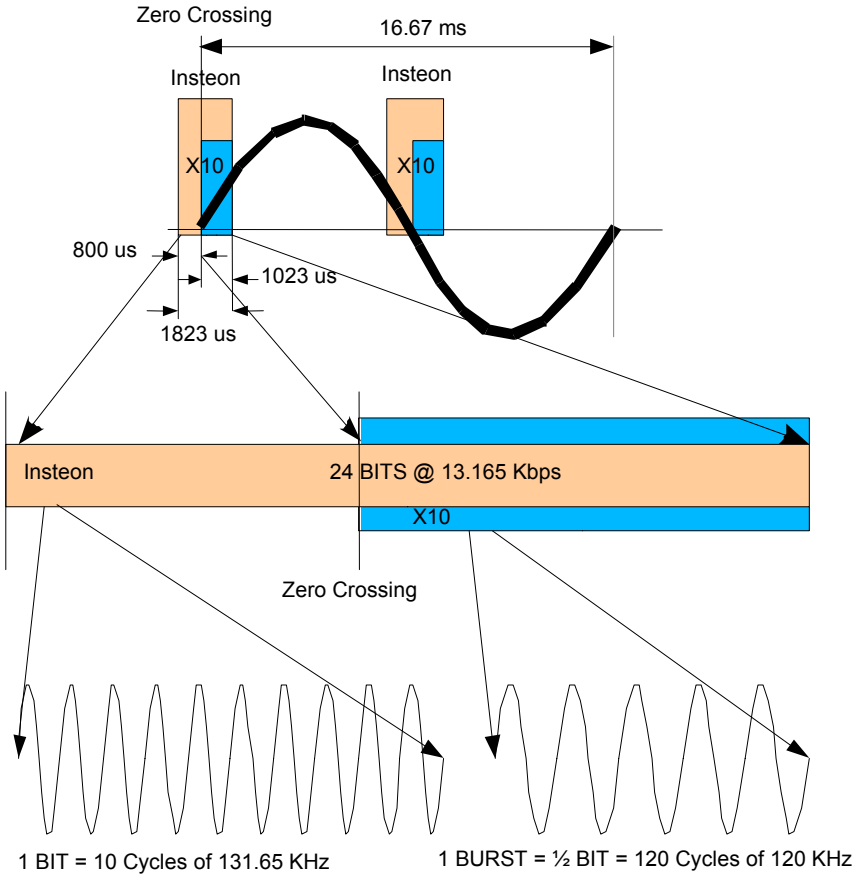


The figure above shows the same BPSK signal with gradual phase shifting. The transmitter introduces the phase change by inserting 1.5 cycles of carrier at 1.5 times the 131.65 KHz frequency. Thus, in the time taken by one cycle of 131.65 KHz, three half-cycles of carrier will have occurred, so the phase of the carrier will be reversed at the end of the period due to the odd number of half-cycles. Note the smooth transitions between the bits.

Packet Timing

All INSTEON powerline packets contain 24 bits. Since a bit takes 10 cycles of 131.65 KHz carrier, there are 240 cycles of carrier in an INSTEON packet. An INSTEON powerline packet therefore lasts 1.823 milliseconds.

The powerline environment is notorious for uncontrolled noise, especially high-amplitude spikes caused by motors, dimmers and compact fluorescent lighting. This noise is minimal during the time that the current on the powerline reverses direction, a time known as the powerline zero crossing. Therefore, INSTEON packets are transmitted during the zero crossing quiet time, as shown in the figure below.



The top of the figure shows a single powerline cycle, which possesses two zero crossings. An INSTEON packet is shown at each zero crossing. INSTEON packets begin 800 microseconds before a zero crossing and last until 1023 microseconds after the zero crossing.

X10 Compatibility

The figure also shows how X10 signals are applied to the powerline. X10 is the signaling method used by many devices already deployed on powerlines around the world. Compatibility[2] with this existing population of legacy X10 devices is an important feature of INSTEON. At a minimum, X10 compatibility means that INSTEON and X10 signals can coexist with each other, but compatibility also allows

designers to create hybrid devices that can send and receive both INSTEON and X10 signals.

The X10 signal uses a burst of approximately 120 cycles of 120 KHz carrier beginning at the powerline zero crossing and lasting about 1000 microseconds. A burst followed by no burst signifies an X10 one bit and no burst followed by a burst signifies an X10 zero bit. An X10 message begins with two bursts in a row followed by a one bit, followed by nine data bits. The figure shows an X10 burst at each of the two zero crossings.

The X10 specification also allows for two copies of the zero crossing burst located one-third and two-thirds of the way through a half-cycle of power. These points correspond to the zero crossings of the other two phases of three-phase power. INSTEON is insensitive to those additional X10 bursts and does not transmit them when sending X10.

The middle of the figure shows an expanded view of an INSTEON packet with an X10 burst superimposed. The X10 signal begins at the zero crossing, 800 microseconds after the beginning of the INSTEON packet. Both signals end at approximately the same time, 1023 microseconds after the zero crossing.

INSTEON devices achieve compatibility with X10 by listening for an INSTEON signal beginning 800 microseconds before the zero crossing. INSTEON receivers implemented in software can be very sensitive, but at the cost of having to receive a substantial portion of a packet before being able to validate that a true INSTEON packet is being received. Reliable validation may not occur until as much as 450 microseconds after the zero crossing, although an INSTEON device will still begin listening for a possible X10 burst right at the zero crossing. If at the 450-microsecond mark the INSTEON receiver validates that it is not receiving an INSTEON packet, but that there *is* an X10 burst present, the INSTEON receiver will switch to X10 mode and listen for a complete X10 message over the next 11 powerline cycles. If instead the INSTEON device detects that it is receiving an INSTEON packet, it will remain in INSTEON mode and not listen for X10 until it receives the rest of the complete INSTEON message.

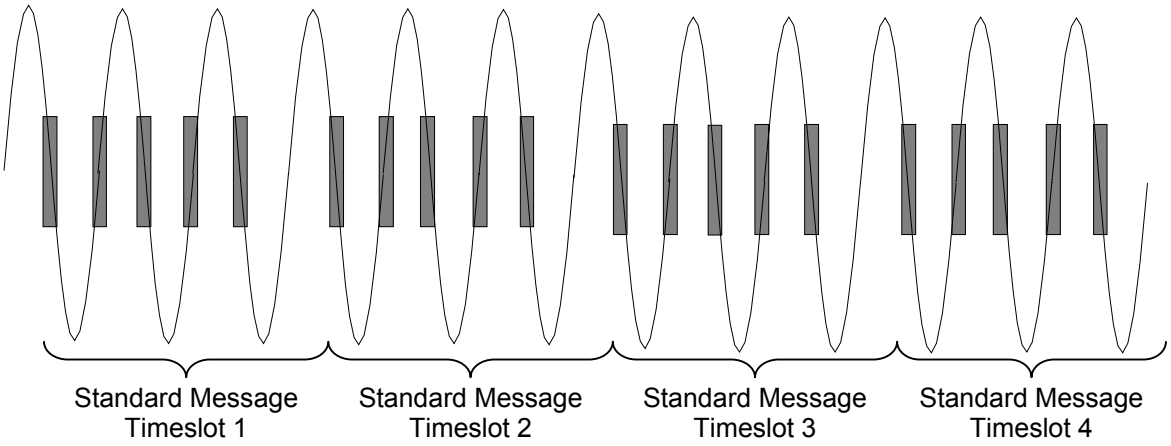
The bottom of the figure shows that the raw bitrate for INSTEON is much faster for INSTEON than for X10. An INSTEON bit requires ten cycles of 131.65 KHz carrier, or 75.96 microseconds, whereas an X10 bit requires *two* 120-cycle bursts of 120 KHz. One X10 burst takes 1000 microseconds, but since each X10 burst is sent at a zero crossing, it takes 16,667 microseconds to send the two bursts in a bit, resulting in a sustained bitrate of 60 bits per second. INSTEON packets consist of 24 bits, and an INSTEON packet can be sent during each zero crossing, so the nominal raw sustained bitrate for INSTEON is 2880 bits per second, 48 times faster than X10. Note that this nominal INSTEON bitrate must be derated to account for packet and message overhead, as well as message retransmissions. See *INSTEON Powerline Data Rates*, below, for details.

Message Timeslots

To allow time for potential retransmission of a message by INSTEON RF devices, an INSTEON transmitter waits for one additional zero crossing after sending a Standard message, or for two zero crossings after sending an Extended message. Therefore, the total number of zero crossings needed to send a Standard message is 6, or 13 for an Extended message. This number, 6 or 13, constitutes an INSTEON message timeslot.

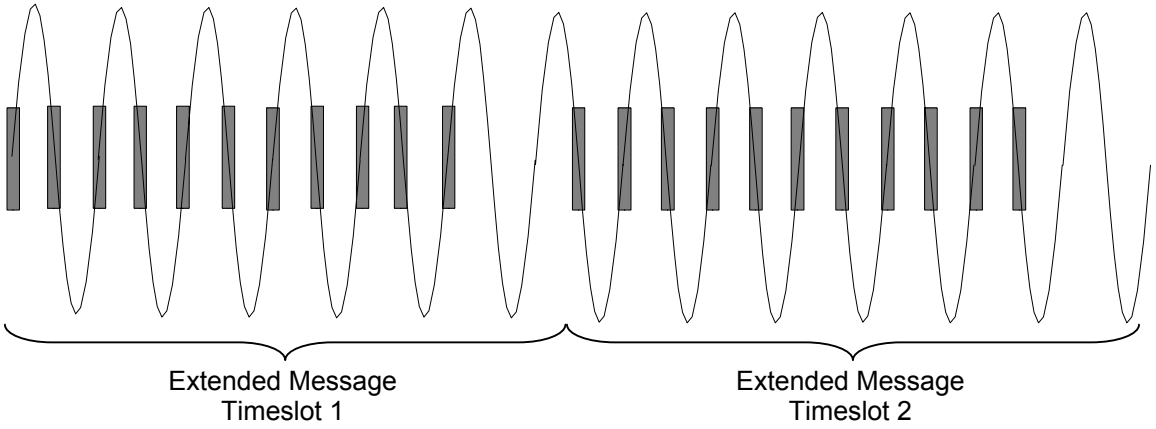
Standard Message Timeslots

The figure below shows a series of 5-packet Standard INSTEON messages being sent on the powerline. INSTEON transmitters wait for one zero crossing after each Standard message before sending another message, so the Standard message timeslot is 6 zero crossings, or 50 milliseconds, in length.



Extended Message Timeslots

The next figure shows a series of 11-packet Extended INSTEON messages being sent on the powerline. INSTEON transmitters wait for two zero crossings after each Extended message before sending another message, so the Extended message timeslot is 13 zero crossings, or 108.33 milliseconds, in length.



INSTEON Powerline Data Rates

INSTEON Standard messages contain 120 raw data bits and require 6 zero crossings, or 50 milliseconds to send. Extended messages contain 264 raw data bits and require 13 zero crossings, or 108.33 milliseconds to send. Therefore, the actual raw bitrate for INSTEON is 2400 bits per second for Standard messages, or 2437 bits per second for Extended messages, instead of the 2880 bits per second it would be without waiting for the extra zero crossings.

INSTEON Standard messages contain 9 bytes (72 bits) of usable data, not counting packet sync and start code bits, nor the message CRC byte. Extended messages contain 23 bytes (184 bits) of usable data using the same criteria. Therefore, the bitrates for usable data are further reduced to 1440 bits per second for Standard messages and 1698 bits per second for Extended messages. If one only counts the 14 bytes (112 bits) of User Data in Extended messages, the User Data bitrate is 1034 bits per second.

These data rates assume that messages are sent with Max Hops set to zero and that there are no message retries. They also do not take into account the time it takes for a message to be acknowledged. The table below shows net data rates when multiple hops and message acknowledgement are taken into account. To account for retries, divide the given data rates by one plus the number of retries (up to a maximum of 5 possible retries).

Condition			Bits per Second		
Max Hops	**ACK**	**Retries**	**Standard Message Usable Data**	**Extended Message Usable Data**	**Extended Message User Data Only**
0	No	0	1440	1698	1034
1	No	0	720	849	517
2	No	0	480	566	345
3	No	0	360	425	259
0	Yes	0	720	849	517
1	Yes	0	360	425	259
2	Yes	0	240	283	173
3	Yes	0	180	213	130

RF Signaling

RF INSTEON devices can send and receive the same messages that appear on the powerline. Unlike powerline messages, however, messages sent by RF are not broken up into smaller packets sent at powerline zero crossings, but instead are sent whole, as was shown in the section *RF Packets*. As with powerline, there are two RF message lengths: Standard 10-byte messages and Extended 24-byte messages.

The table below gives the specifications for INSTEON RF signaling.

RF Specification	Value
Center Frequency	904 MHz
Data Encoding Method	Manchester
Modulation Method	FSK
FSK Deviation	64 KHz
FSK Symbol Rate	76,800 symbols per second
Data Rate	38,400 bits per second
Range	150 feet outdoors

The center frequency lies in the band 902 to 924 MHz, which is permitted for unlicensed operation in the United States. Each bit is Manchester encoded, meaning that two symbols are sent for each bit. A one-symbol followed by a zero-symbol designates a one-bit, and a zero-symbol followed by a one-symbol designates a zero-bit. Symbols are modulated onto the carrier using frequency-shift keying (FSK), where a zero-symbol modulates the carrier half the FSK deviation frequency downward and a one-symbol modulates the carrier half the FSK deviation frequency upward. The FSK deviation frequency chosen for INSTEON is 64 KHz. Symbols are modulated onto the carrier at 76,800 symbols per second, resulting in a raw data rata of half that, or 38,400 bits per second. The typical range for free-space reception is 150 feet, which is reduced in the presence of walls and other RF energy absorbers.

INSTEON devices transmit data with the most-significant bit sent first. Referring to the figures below, RF messages begin with two sync bytes consisting of AAAA in hexadecimal, followed by a start code byte of C3 in hexadecimal. Ten data bytes follow in Standard messages, or twenty-four data bytes in Extended messages. The last data byte in a message is a CRC[3] over the data bytes as explained above (see *Message Integrity Byte*).

RF Standard Message – 1 Packet

112 total bits = 14 bytes
80 Data bits = 10 bytes

AA	AA	C3	x	x	x	x	x	x	x	x	x	x	n
2 Sync bytes		1 Start Code byte	80 Data Bits (10 Data bytes)										CRC[3]

RF Extended Message – 1 Packet

224 total bits = 28 bytes
192 Data bits = 24 bytes

AA	AA	C3	x	x	x	x	x	x	x	X	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	n
2 Sync bytes		1 Start Code byte	192 Data Bits (24 Data bytes)																								CRC[3]

It takes 2.708 milliseconds to send a 104-bit Standard message, and 5.625 milliseconds to send a 216-bit Extended message. Zero crossings on the powerline occur every 8.333 milliseconds, so a Standard or Extended RF message can be sent during one powerline half-cycle. The waiting times after sending powerline messages, as shown in the section *Powerline Packets*, are to allow sufficient time for INSTEON RF devices, if present, to retransmit a powerline message.

Simulcasting

By following the above rules for message propagation, INSTEON systems achieve a marked increase in the reliability of communications. The reason is that multiple INSTEON devices can transmit the same message at the same time within a given timeslot. INSTEON devices within range of each other thus "help each other out." Most networking protocols for shared physical media prohibit multiple devices from simultaneously transmitting within the same band by adopting complex routing algorithms. In contrast, INSTEON turns what is usually a problem into a benefit by ensuring that devices transmitting simultaneously will be sending the same messages in synchrony with each other.

Powerline Simulcasting

One might think that multiple INSTEON devices transmitting on the powerline could easily cancel each other out rather than boost each other. In practice, even if one were *trying* to nullify one signal with another, signal cancellation by multiple devices would be extremely difficult to arrange. The reason is that for two signals to cancel at a given receiver, the two transmitters would have to send carriers such that the receiver would see them as exactly equal in amplitude and very nearly 180 degrees out of phase. The probability of this situation occurring and persisting for extended periods is low.

The crystals used on typical INSTEON devices to generate the powerline carrier frequency of 131.65 KHz run independently of each other with a frequency tolerance of a few tenths of a percent. Phase relationships among multiple powerline carriers therefore will drift, although slowly with respect to the 1823 microsecond duration of an INSTEON packet. Even if the phases of two transmitters happened to cancel, it is very unlikely that the amplitudes would also be equal at the location of a receiver, so a receiver would very likely still see some signal even in the worst-case transient phase relationship. INSTEON receivers have a wide dynamic range, from millivolts to five volts or so, which will allow them to track signals even if they fade temporarily. Adding more transmitters reduces the probability of signal cancellation even more. Rather, the probability that the sum of all the signals will increase in signal strength becomes much greater with source diversity.

The INSTEON powerline carrier is modulated using binary phase-shift keying (BPSK), meaning that receivers are looking for 180-degree phase shifts in the carrier to detect changes in a string of bits from a one to a zero or vice-versa. Multiple transmitters, regardless of the absolute phase of their carriers, will produce signals whose sum still possesses 180-degree phase reversals at bit-change boundaries, so long as their relative carrier frequencies do not shift more than a few degrees over a packet time. Of course, bit timings for each transmitter need to be fairly well locked, so INSTEON transmitters are synchronized to powerline zero crossings. An INSTEON bit lasts for ten cycles of the 131.65 KHz powerline carrier, or 76 microseconds. The powerline zero crossing detector should be accurate within one or two carrier periods so that bits received from multiple transmitters will overlay each other.

In practice, multiple INSTEON powerline transmitters simulcasting the same message will improve the strength of the powerline signal throughout a building.

RF Simulcasting

Since RF signaling is used as an extension to powerline signaling, it also is based on simulcasting. However, because of the short wavelength of 900 MHz RF carrier signals, standing wave interference patterns can form where the RF carrier signal is reduced, even when the carrier and data are ideally synchronized.

As with powerline, for a cancellation to occur, two carriers must be 180 degrees out of phase and the amplitudes must be the same. Perfect cancellation is practically impossible to obtain. In general, two co-located carriers on the same frequency with random phase relationships and the same antenna polarization will sum to a power level greater than that of just one transmitter 67% of the time. As one of the transmitters is moved away from a receiver, the probability of cancellation drops further because the signal amplitudes will be unequal. As the number of transmitters increases, the probability of cancellation becomes nearly zero.

Mobile INSTEON RF devices, such as handheld controllers, are battery operated. To conserve power, mobile devices are not configured as RF Repeaters, but only as message originators, so RF simulcasting is not an issue for them. INSTEON devices that do repeat RF messages are attached to the powerline, so most of them will not be moved around after initial setup. During setup, such RF devices can be located, and their antennas adjusted, so that no signal cancellation occurs. With the location of the transmitters fixed, the non-canceling configuration will be maintained indefinitely.

RF/Powerline Synchronization

INSTEON RF devices attached to the powerline use the zero crossing for message synchronization. These devices receive INSTEON messages synchronously on the powerline, synchronously via RF from RF Repeaters, or possibly asynchronously via RF from mobile RF devices.

Messages that need to be retransmitted will have a Hops Left count greater than zero. If the INSTEON device receives such a message from the powerline, it will first retransmit the message using RF as soon as it has received the last packet of the powerline message, then it will retransmit the message on the powerline in the next timeslot. If the device receives the message via RF, it will first retransmit the message on the powerline in the next timeslot, then it will retransmit the message using RF immediately after sending the last packet of the powerline message. In this way, RF message received asynchronously will be resynchronized to the powerline zero crossing at the earliest opportunity.

INSTEON Network Usage

INSTEON messaging technology can be used in many different ways in many kinds of devices. To properly utilize the full set of possible INSTEON message types, devices must share a common set of specific, preassigned number values for the one- and two-byte Commands, two-byte Device Types, one-byte Device Attributes, one- or two-byte ACK statuses, and one- or two-byte NAK reasons. Smarthome maintains the database of allowable values for these parameters.

Because INSTEON devices are individually preassigned a three-byte Address at the time of manufacture, complex procedures for assigning network addresses in the field are not needed. Instead, INSTEON devices are logically linked together in the field using a simple, uniform procedure.

INSTEON Extended messages allow programmers to devise all kinds of meanings for the User Data that can be exchanged among devices. For example, many INSTEON devices include an interpreter for an application language, called SALad, which is compiled into token strings and downloaded into devices using Extended messages. Also, secure messaging can be implemented by sending encrypted payloads in Extended messages.

In this section

INSTEON Commands

INSTEON's simplicity stems from the fact that Standard messages are all 10 bytes in length, and they contain just two payload bytes, Command 1 and Command 2.

Smarthome maintains the table of possible INSTEON Commands. This table is currently very sparsely populated. Designers who wish to create INSTEON devices that implement new Commands should contact Smarthome at info@insteon.net.

The basic rules for handling INSTEON Commands depend on whether a device is currently acting as a Controller or a Responder.

Controllers have a repertoire of Commands that they can send, usually set by the firmware in the device. Examples for a lighting controller might include *On*, *Off*, *Bright*, *Dim*, *Fast On*, and *Fast Off*. Obviously, a Controller can only send the Commands it knows about, and no others.

Responders likewise have a repertoire of Commands that they can act upon. For example, a lamp dimmer's firmware might contain procedures to respond to *On*, *Off*, *Bright*, and *Dim* Commands, but not *Fast On* and *Fast Off*. A Responder will only act on the Commands *it* knows about, and no others.

Smarthome maintains a cross-reference between Device Classes and Commands that a device must implement for INSTEON conformance certification. Contact Smarthome at info@insteon.net for more information.

Command 1

Command 1 holds an 8-bit number representing the INSTEON Primary Command to execute.

Command 2

The interpretation of the Command 2 field depends on the Primary Command in the Command 1 field.

Parameter

The Command 2 field can be a parameter for the Primary Command. For example, Command 0x11 (*On*) has a parameter in Command 2 ranging from 0x00 to 0xFF representing the On-Level.

Subcommand

Command 2 can act as a Subcommand for certain blocks of Primary Commands. Taken together, the 2-byte Primary-plus-Subcommands allow for expansion of the command space.

Group Number

For Groups of linked devices, Controllers first send a Group Broadcast message containing a Primary Command to all devices in the Group at once. Responders in the Group will execute the Primary Command right away, but they will not reply with an acknowledgement. To ensure reliability, a Controller follows up a Group Broadcast with a Group Cleanup message sent individually to each member of the Group. In Group Cleanup messages Command 2 contains the Group Number.

Acknowledgement

Command 2 can return data to the sending device in an acknowledgement message. For example, an ACK message responding to Commands *On*, *Off*, *Bright*, *Dim*, *Fast On*, *Fast Off*, *Start Manual Change*, and *Stop Manual Change* will hold the On-Level in the Command 2 field. If one of these Commands is sent and the response is a NAK message, then the Command 2 field will hold the NAK reason code.

INSTEON Device Classes

The number of different kinds of devices that can be connected to an INSTEON network is virtually unlimited. Rather than relying on an elaborate scheme for discovery of device capabilities, INSTEON's designers opted for a very simple, yet expandable method for devices to identify themselves—they Broadcast a *Set Button Pressed* message containing device classification information. This information, the *Device Category*, *Device Descriptor*, *Device Attributes*, and *Firmware Revision*, appears in fixed-length fields within the Broadcast message.

Device Identification Broadcast

INSTEON devices identify themselves to other devices on an INSTEON network by sending a *Set Button Pressed* Broadcast message. This message contains a number of fields that describe the product type and capabilities.

A *Device Type* field, containing the *Device Category* and *Device Descriptor*, appears in the most significant 2 bytes of the To Address field, followed by the *Firmware Revision* in the least significant byte. A *Device Attributes* byte appears in the Command 2 field.

<table>
<tr><th colspan="6">INSTEON Set Button Pressed Broadcast Message</th></tr>
<tr><th>From Address</th><th colspan="3">To Address</th><th>Message Flags</th><th>Command 1</th><th>Command 2</th></tr>
<tr><td>3 bytes</td><td colspan="3">3 bytes</td><td>1 byte</td><td>1 byte</td><td>1 byte</td></tr>
<tr><td rowspan="4"></td><td colspan="2">2 bytes</td><td>1 byte</td><td rowspan="4">1000xxxx</td><td rowspan="4">Set Button Pressed (0x01)</td><td rowspan="4">Device Attributes</td></tr>
<tr><td colspan="2">Device Type</td><td rowspan="3">Firmware Revision</td></tr>
<tr><td>4 bits</td><td>12 bits</td></tr>
<tr><td>Device Category</td><td>Device Descriptor</td></tr>
</table>

Device Type

Each INSTEON device contains a 2-byte Device Type identifier consisting of a 4-bit Device Category and a 12-bit Device Descriptor. This assignment allows up to 16 Device Categories and 4096 Device Descriptors per category, for a total of 65,536 different possible Device Types.

Device Type – 2 Bytes	
Device Category – 4 bits	**Device Descriptor** – 12 bits
16 possible 0x0 to 0xF	4096 possible per Device Category 0x000 to 0xFFF

Device Category

This 4-bit field within the Device Type designates the broad class of devices to which an INSTEON-enabled product belongs. At present, only one Device Category, which can be thought of as 'Initial INSTEON Devices,' is defined. Smarthome will assign other Device Categories as needed.

Device Category	Value	Meaning
1	0x0	Includes first-release INSTEON devices
2 to 16	0x1 to 0xF	Available for future use

Device Descriptor

This 12-bit field within the Device Type is different for each Device Category. Smarthome assigns these numbers to device manufacturers. Currently, Device Descriptors are being assigned sequentially. Designers who wish to develop INSTEON-enabled products should contact Smarthome at info@insteon.net for more information.

Device Attributes

When an INSTEON device identifies itself by sending out a *Set Button Pressed* Broadcast message, the message includes a Device Attributes byte in the Command 2 field. Although currently unused, this byte can contain individual bit flags that could be interpreted differently for each Device Type.

Firmware Revision

An INSTEON device's firmware revision number appears in the least significant byte of the To Address field of a *Set Button Pressed* Broadcast message. The high nibble (4 bits) gives the major revision number and the low nibble gives the minor revision.

INSTEON Device Linking

When a user adds a new device to an INSTEON network, the newcomer device joins the network automatically, in the sense that it can hear INSTEON messages and will repeat[1] them automatically according to the INSTEON protocol. So, no user intervention is needed to establish an INSTEON network of communicating devices.

However, for one INSTEON device to control other INSTEON devices, the devices must be logically linked together. INSTEON provides two very simple methods for linking devices—manual linking using button pushes, and electronic linking using INSTEON messages.

INSTEON Groups

During linking, users create associations between events that can occur in an INSTEON Controller, such as a button press or a timed event, and the actions of a Group of one or more Responders. This section defines *Groups and Links* and gives *Examples of Groups*.

Groups and Links

A Group is a set of logical Links between INSTEON devices. A Link is an association between a Controller and a Responder or Responders. Controllers originate Groups, and Responders join Groups.

Internally, in a Link Database maintained by INSTEON devices, a Group ID consists of 4 bytes—the 3-byte address of the Controller, and a 1-byte Group Number. A Controller assigns Group Numbers as needed to the various physical or logical events that it supports. For example, a single press of a certain button could send commands to one Group, and a double press of the same button could send commands to another Group. The Controller determines which commands are sent to which Groups.

A Group can have one or many members, limited only by the memory available for the Link Database.

Examples of Groups

A device configured as a wall switch with a paddle could be designed to support one, two, or three Groups, as shown in the following examples.

One Group		
Controller Event	**Group**	**Action of Group Responders**
Tap Top	1	Turn On
Tap Bottom	1	Turn Off
Hold Top	1	Brighten
Hold Bottom	1	Dim

Two Groups		
Controller Event	**Group**	**Action of Group Responders**
Tap Top	1	Turn On
Tap Top Again	1	Turn Off
Tap Bottom	2	Turn On
Tap Bottom Again	2	Turn Off

Three Groups		
Controller Event	**Group**	**Action of Group Responders**
Tap Top	1	Turn On
Tap Bottom	1	Turn Off
Double Tap Top	2	Turn On
Double Tap Bottom	2	Turn Off
Triple Tap Top	3	Turn On
Triple Tap Bottom	3	Turn Off

Methods for Linking INSTEON Devices

There are two ways to create links among INSTEON devices, *Manual Linking* and *Electronic Linking*. This section also gives an *Example of an INSTEON Linking Session*.

Manual Linking

Easy setup is very important for products sold to a mass market. INSTEON devices can be linked together very simply:

- Push and hold for 10 seconds the button that will control an INSTEON device.
- Push and hold a button on the INSTEON device to be controlled.

This kind of manual linking implements a form of security. Devices cannot be probed by sending messages to discover their addresses—a user must have physical possession of INSTEON devices in order to link them together.

Designers are free to add to this basic linking procedure. For example, when multiple devices are being linked to a single button on a Controller, a *multilink* mode could enable a user to avoid having to press and hold the button for 10 seconds for each new device.

There must also be procedures to unlink devices from a button, and ways to clear links from buttons in case devices linked to them are lost or broken. See the *INSTEON Link Database* section below for more information on this point.

Electronic Linking

As the example below shows (see *Example of an INSTEON Linking Session*), linking is actually accomplished by sending INSTEON messages, so a PC or other device can create links among devices if the device addresses are known and if devices can respond to the necessary commands.

To maintain security, PC-INSTEON interface devices such as Smarthome's PowerLinc™ V2 Controller (PLC) mask the two high bytes of the address fields in INSTEON messages received from unknown devices. Devices are only known if there is a link to the device stored in the Link Database of the PLC, or if the message's To Address matches that of the PLC. Such links must have been previously established by manual button pushing or else by manually typing in the addresses of linked devices (see *Masking Non-linked Network Traffic*, below).

Example of an INSTEON Linking Session

This section outlines the message exchange that occurs when a Controller and Responder set up a link relationship. In this scenario, a Smarthome ControLinc™ V2 is the Controller, and a Smarthome LampLinc™ V2 is the Responder. Numbers are in hexadecimal.

Message 1	**ControLinc: "I'm looking for Group members"**		
`00 00 CC 00 04 0C 8F 01 00`	ControLinc, with address of 00 00 CC, sends a *Set Button Pressed* Broadcast message indicating it is now listening for Responders to be added to Group 1.		
	From Address		00 00 CC (ControLinc)
	To Address	Device Type	00 0A (ControLinc)
		Firmware Version	0C
	Flags		8F (Broadcast Message, 3 Max Hops, 3 Hops Left)
	Command 1		01 (*Set Button Pressed*)
	Command 2	Device Attributes	00 (Not used)

Message 2	**LampLinc: "I'll join your Group"**		
`00 00 AA 01 00 30 8F 01 00`	LampLinc, with address of 00 00 AA, sends a *Set Button Pressed* Broadcast message. When the ControLinc hears this, it will respond with a message to join Group 1.		
	From Address		00 00 AA (LampLinc)
	To Address	Device Type	00 02 (LampLinc)
		Firmware Version	30
	Flags		8F (Broadcast Message, 3 Max Hops, 3 Hops Left)
	Command 1		01 (*Set Button Pressed*)
	Command 2	Device Attributes	00 (Not used)

Message 3	**ControLinc: "Okay, join Group 1"**	
`00 00 CC 00 00 AA 0F 01 01`	ControLinc (00 00 CC) sends message to LampLinc (00 00 AA) to join Group 1.	
	From Address	00 00 CC (ControLinc)
	To Address	00 00 AA (LampLinc)
	Flags	0F (Direct Message, 3 Max Hops, 3 Hops Left)
	Command 1	01 (*Assign to Group*)
	Command 2	01 (Group 1)

Message 4	LampLinc: "I joined Group 1"	
00 00 AA 00 00 CC 2F 01 01	LampLinc (00 00 31) sends ACK to ControLinc (00 00 10).	
	From Address	00 00 AA (LampLinc)
	To Address	00 00 CC (ControLinc)
	Flags	2F (ACK of Direct Message, 3 Max Hops, 3 Hops Left)
	Command 1	01 (*Assign to Group*)
	Command 2	01 (Group 1)

Example of INSTEON Group Conversation

This example illustrates how messages are passed from device to device in a group. In this scenario, a Smarthome ControLinc™ V2 linked to two Smarthome LampLinc™ V2 Dimmers in Group 1 commands them to turn on. Numbers are in hexadecimal.

Note that the Group Broadcast message (which both LampLinc Dimmers should respond to immediately) is followed by an acknowledged Group Cleanup message to each LampLinc Dimmer (in case they didn't get the Broadcast).

Message 1	ControLinc: "Group 1, turn on"		
00 00 CC 00 00 01 CF 11 00	ControLinc, with address of 00 00 CC, sends a Group Broadcast message to Group 1, with a command of *On*.		
	From Address		00 00 CC (ControLinc)
	To Address	Unused	00 00
		Group Number	01
	Flags		CF (Group Broadcast Message, 3 Max Hops, 3 Hops Left)
	Command 1		11 (*On*)
	Command 2		00 (Unused)

Message 2	ControLinc: "LampLinc A, turn on"		
00 00 CC 00 00 AA 4F 01 00	ControLinc (00 00 CC) sends a Group Cleanup message to LampLinc A (00 00 AA) in Group 1, with a command of *On*.		
	From Address		00 00 CC (ControLinc)
	To Address		00 00 AA (LampLinc A)
	Flags		4F (Group Cleanup Message, 3 Max Hops, 3 Hops Left)
	Command 1		11 (*On*)
	Command 2	Group Number	01

Message 3	LampLinc A: "I turned on"		
00 00 AA 00 00 CC 2F 01 01	LampLinc A (00 00 AA) sends ACK to ControLinc (00 00 CC).		
	From Address		00 00 AA (LampLinc A)
	To Address		00 00 CC (ControLinc)
	Flags		2F (ACK of Direct Message, 3 Max Hops, 3 Hops Left)
	Command 1		11 (*On*)
	Command 2	Group Number	01

Message 4	ControLinc: "LampLinc B, turn on"		
00 00 CC 00 00 BB 4F 01 00	ControLinc (00 00 CC) sends a Group Cleanup message to LampLinc B (00 00 BB) in Group 1, with a command of *On*.		
	From Address		00 00 CC (ControLinc)
	To Address		00 00 BB (LampLinc B)
	Flags		4F (Group Cleanup Message, 3 Max Hops, 3 Hops Left)
	Command 1		11 (*On*)
	Command 2	Group Number	01

Message 5	LampLinc B: "I turned on"		
00 00 BB 00 00 CC 2F 01 01	LampLinc B (00 00 BB) sends ACK to ControLinc (00 00 CC).		
	From Address		00 00 BB (LampLinc B)
	To Address		00 00 CC (ControLinc)
	Flags		2F (ACK of Direct Message, 3 Max Hops, 3 Hops Left)
	Command 1		11 (*On*)
	Command 2	Group Number	01

An INSTEON Controller will send Group Cleanup commands to all Responder devices in a Group, unless other INSTEON traffic interrupts the cleanup, in which case the Group Cleanups will stop.

INSTEON Link Database

Every INSTEON device stores a Link Database in nonvolatile memory, representing Controller/Responder relationships with other INSTEON devices. Controllers know which Responders they are linked to, and Responders know which Controllers they are linked to. Link data is therefore distributed among devices in an INSTEON network.

If a Controller is linked to a Responder, and the Responder is removed from the network without updating the Controller's Link Database, then the Controller will retry messages intended for the missing Responder. The retries, which are guaranteed to fail, will add unnecessary traffic to the network. It is therefore very important for users to unlink INSTEON Responder devices from Controllers when unused Responders are removed. Unlinking is normally accomplished in the same way as linking—press and hold a button on the Controller, then press and hold a button on the Responder.

Because lost or broken Responder devices cannot be unlinked using a manual unlinking procedure, Controllers must also have an independent method for unlinking missing Responders. Providing a 'factory reset' procedure for a single Controller button, or for the entire Controller all at once, is common.

When a Controller is removed from the network, it should likewise be unlinked from all of its Responder devices before removal, or else the Link Databases in the Responders will be cluttered up with obsolete links. A 'factory reset' should be provided for Responder devices for this purpose.

INSTEON Extended Messages

Designers are free to devise all kinds of meanings for the User Data that can be exchanged among devices using INSTEON Extended messages. For example, many INSTEON devices include an interpreter for an application language, called SALad, which is compiled into token strings. SALad token string programs can be downloaded into devices (and they can also be debugged) using Extended messages (see *SALad Applications*, below).

For applications that must be secure, such as door locks and security systems, Extended messages can contain encrypted data. See *Encryption within Extended Messages*, below for more information.

If an application needs to transport more than 14 bytes of User Data, then it can use multiple Extended messages. Each Extended message can act as a packet, with the complete User Data reassembled after all packets are received. Bear in mind, however, that INSTEON was not designed for unrestricted data communications—instead, it is optimized for simplicity and reliability. Designers are encouraged to minimize the use of INSTEON to handle large amounts of data.

INSTEON Security

INSTEON network security is maintained at two levels. *Linking Control* ensures that users cannot create links that would allow them to control their neighbors' INSTEON devices, even though those devices may be repeating each other's messages. *Encryption within Extended Messages* permits completely secure communications for applications that require it.

Linking Control

INSTEON enforces Linking Control by requiring that users have *Physical Possession of Devices* in order to create links, and by *Masking Non-linked Network Traffic* when messages are relayed outside the INSTEON network itself.

Physical Possession of Devices

Firmware in INSTEON devices prohibits them from identifying themselves to other devices unless a user physically presses a button on the device. That is why the Command in the network identification Broadcast message is called *Set Button Pressed*. As shown above in the section *Example of an INSTEON Linking Session*, a user has to push buttons on both the Controller device and the Responder device in order to establish a link between them. A Responder will not act on commands from an unlinked Controller.

Linking by sending INSTEON messages requires knowledge of the 3-byte addresses of INSTEON devices. These addresses, unique for each device, are assigned at the factory and displayed on printed labels attached to the device. Users who have physical possession of a device can read the device address from the label and manually enter it when prompted by a computer program.

Masking Non-linked Network Traffic

As described in the section *Interfacing to an INSTEON Network*, below, there can be many kinds of INSTEON devices, called Bridges, that connect an INSTEON network to the outside world. But since an INSTEON Bridge is itself just another INSTEON device, it must be linked to other devices on the INSTEON network in order to exchange messages with them. A user must establish these links in the same way as for any other INSTEON device—by pushing buttons or by typing in addresses.

Smarthome's PowerLinc™ Controller (PLC) is an example of an INSTEON-certified Bridge device that monitors INSTEON traffic and relays it to a computer via a serial link. For security, the PLC's firmware masks the two high-bytes in the address fields of INSTEON messages unless the traffic is from an INSTEON device already linked to the PLC, or the traffic is from a device that already knows the address of the PLC. In this way, software can take into account the existence of INSTEON traffic without users being able to discover the addresses of devices that they never had physical access to.

To avoid 'spoofing,' where an attacker poses as someone else (by causing the PLC to send messages with bogus From Addresses), the PLC's firmware always inserts the true PLC ID number in the From Address field of messages that it sends.

Encryption within Extended Messages

For applications such as door locks and security systems, INSTEON Extended messages can contain encrypted payloads. Possible encryption methods include rolling-code, managed-key, and public-key algorithms. In keeping with INSTEON's hallmark of simplicity, rolling-code encryption, as used by garage door openers and radio keyfobs for cars, is the method preferred by Smarthome. The encryption method that will be certified as the INSTEON standard is currently under development.

INSTEON Application Development

INSTEON, with its no-nonsense emphasis on simplicity, reliability, and low cost, is optimized as an *infrastructure* network. Common devices in the home, such as light switches, door locks, thermostats, clocks, and entertainment systems currently do not communicate with one another. INSTEON can change all that.

When devices are networked together, there is a potential for coordinated, adaptive behavior that can bring a new, higher level of comfort, safety, and convenience to living. But networking devices together cannot by itself change the behavior of the devices. It is application-level software, created by developers, that transforms a network of previously unrelated devices into a coordinated, adaptive, lifestyle-enhancing system.

There are two basic kinds of applications that developers can create for INSTEON-networked devices: External Applications and Internal Applications.

External Applications run on a computing device such as a PC or PDA. A special type of INSTEON module called an INSTEON *Bridge* connects the computing device to an INSTEON network. *Manager Apps* are External Applications that exchange INSTEON messages directly with INSTEON devices via a Bridge.

Internal Applications run on INSTEON devices themselves. Smarthome has developed an embedded language interpreter, called SALad, which resides in the firmware of SALad-enabled INSTEON devices. Developers can create and debug *SALad Apps* in a Smarthome Integrated Development Environment (IDE) that communicates with INSTEON devices via an INSTEON Bridge. Devices running SALad Apps can exhibit very sophisticated behavior. Moreover, devices that have already been installed in the home can be upgraded by downloading new SALad Apps to them. With INSTEON upgradeability, the world of home control can dynamically adapt to people's expectations and needs as the marketplace evolves.

In this section

Interfacing to an INSTEON Network

An INSTEON device that connects an INSTEON network to the outside world is called an *INSTEON Bridge*. There can be many kinds of INSTEON Bridges. One kind, an INSTEON-to-Serial Bridge, connects an INSTEON network to a computing device like a PC, a PDA, or to a dedicated user interface device with a serial port. Another Bridge, INSTEON-to-IP, connects an INSTEON network to a LAN or the Internet, either with wires (like Ethernet) or wirelessly (like WiFi). Still other INSTEON Bridges could connect to other networks such as wired or wireless telephony, Bluetooth, ZigBee, WiMax, or whatever else emerges in the future.

The Smarthome PowerLinc Controller

The PowerLinc™ V2 Controller (PLC) from Smarthome is an example of an INSTEON-to-Serial Bridge for connecting an INSTEON network to a computing device. PLCs are currently available with either a USB or an RS232 serial interface. An Ethernet interface, for connecting to a LAN or the Internet, is under development.

Using the PLC, application developers can create high-level user interfaces to devices on an INSTEON network. *Manager Apps* are External Applications that run on a computing device and use the PLC to directly send and receive INSTEON messages to INSTEON devices. *SALad Apps* are Internal Applications that run on SALad-enabled INSTEON devices themselves. The PLC is a SALad-enabled INSTEON device, having a SALad language interpreter embedded in its firmware.

As shipped by Smarthome, the PLC contains a 1200-byte SALad program called CoreApp that performs a number of useful functions:

- When CoreApp receives messages from INSTEON devices, it sends them to the computing device via its serial port, and when it receives INSTEON-formatted messages from the computing device via the serial port, it sends them out over the INSTEON network.
- CoreApp handles linking to other INSTEON devices and maintains a Link Database.
- CoreApp is event-driven, meaning that it can send messages to the computing device based on the time of day or other occurrences.
- CoreApp can send and receive X10 commands.

Source code for CoreApp is available to developers to modify for their own purposes. Once programmed with an appropriately modified SALad App, the PLC can operate on its own without being connected to a computing device.

As described in the section *Masking Non-linked Network Traffic*, the PLC hides the full addresses contained within INSTEON messages that it sees, unless the messages are from devices that it is already linked to. In particular, SALad Apps that the PLC may be running cannot discover the addresses of previously unknown INSTEON devices, so a hacker cannot write a SALad App that violates INSTEON security protocols.

Manager Applications

An INSTEON Manager App is an External Application program that runs on a computing device, like a PC or PDA, connected to an INSTEON network via an INSTEON Bridge. Manager Apps can provide sophisticated user interfaces for INSTEON devices, they can interact in complex ways with the outside world, and they can orchestrate system behaviors that bring real lifestyle benefits to people.

A Manager App exchanges INSTEON messages directly with INSTEON devices, so it must contain a software module that can translate between a user's intentions and the rules for composing and parsing INSTEON messages.

An example of a Manager App that encapsulates these functions is Smarthome's *Device Manager* (DM), a Windows program that connects to an INSTEON network via a PowerLinc™ Controller (PLC). DM handles all the intricacies involved with sending and receiving INSTEON messages via a PLC. To the outside world, it exposes an interface that developers can connect their own custom top-level application layers to.

This topmost layer, often a user interface, communicates with DM using the Internet HTTP protocol or Microsoft's ActiveX, so it can run on an Internet browser or within a Windows program. DM and the top layer communicate using a simple text-based scripting language developed by Smarthome called *Home Network Language™* (HNL).

DM allows designers to concentrate on rapid application development of their end products without having to deal directly with INSTEON messaging issues. Product developers are encouraged to contact Smarthome at info@insteon.net for more information about acquiring and using DM.

SALad Applications

SALad is a language interpreter embedded in the firmware of SALad-enabled INSTEON devices (see *SALad Overview*). By writing and debugging SALad programs in Smarthome's *SALad Integrated Development Environment*, developers can create INSTEON Internal Applications that run directly on SALad-enabled devices.

SALad Overview

Because the SALad instruction set is small and addressing modes for the instructions are highly symmetrical, SALad programs run fast and SALad object code is very compact.

SALad is event driven. Events are triggered when a device receives an INSTEON message, a user pushes a button, a timer expires, an X10 command is received, and so forth. As events occur, firmware in a SALad-enabled device posts event handles to an event queue, and then starts the SALad program. The SALad program determines what action to take based on the event that started it.

SALad programs can be downloaded into nonvolatile memory of INSTEON devices using the INSTEON network itself, or via a serial link if the device has one. SALad also contains a small debugger that allows programs to be started, stopped, and single-stepped directly over the INSTEON network.

SALad programming mostly consists of writing event handlers. By following examples in the INSTEON Software Development Kit, or by modifying Smarthome's CoreApp (see *The Smarthome PowerLinc Controller*), developers can rapidly create INSTEON devices with wide-ranging capabilities. For more information about the SALad Language, contact Smarthome at info@insteon.net.

SALad Integrated Development Environment

The SALad Integrated Development Environment (IDE) is a comprehensive, user-friendly tool for creating and debugging Internal Applications that run directly on SALad-enabled INSTEON devices. Using this tool, programmers can write, compile, download, and debug SALad programs without ever having to leave the IDE. The IDE is a Windows program that connects to an INSTEON network using a Smarthome PowerLinc™ Controller (see *The Smarthome PowerLinc Controller*).

The SALad IDE includes:

- A SALad Compiler that reads SALad language files and writes SALad object code, error listings, and variable maps
- A communications module that can download SALad object code to an INSTEON device via USB, RS-232, or the INSTEON network itself
- A multiple-file, color-contextual source code editor that automatically compiles SALad programs on the fly
- Code templates for common tasks
- A real-time debugger based upon instantaneous feedback from a SALad-enabled device
- A program tracer

- An interactive device conversation window for sending and receiving INSTEON, X10, or ASCII messages
- A raw data window
- A PLC simulator for writing and debugging SALad Apps without actually being connected to an INSTEON network
- INSTEON device diagnostics
- INSTEON network diagnostics
- A device Link Database manager
- A program listing formatter

INSTEON Developer's Kits

Smarthome is committed to making the development process as easy as possible for those who create products that can profit from INSTEON networking. For designers who will be crafting new INSTEON devices, adding INSTEON networking to existing devices, or developing External Applications for a network of INSTEON devices, Smarthome offers both a Software Developer's Kit (SDK) and a series of Hardware Development Modules, as well as extensive technical support.

Software Developer's Kit

To encourage as many developers as possible to join the community of INSTEON product creators, Smarthome offers a comprehensive Software Developer's Kit (SDK) for $99. The INSTEON SDK includes:

- The INSTEON Integrated Development Environment (IDE)
- A Smarthome PowerLinc™ V2 Controller (PLC) with either a USB or RS232 serial interface
- A Smarthome LampLinc™ V2 Dimmer module
- An inclusive INSTEON Developer's Guide in both text and compiled help formats
- Access to technical support and peer networking on the INSTEON Internet Forum
- Source code to the SALad CoreApp that runs on the PLC
- Sample SALad Applications
- Version maps for product upgrades
- Header files

Hardware Development Modules

Smarthome will be releasing a series of Hardware Development Modules. The module that is currently available is an isolated powerline module ($99), to be followed soon by a non-isolated powerline module and an RF development module.

The isolated powerline module is essentially a PowerLinc™ Controller (PLC) with an extender board that has a prototyping area and a hardware interface to internal circuitry, including the microcontroller. With this module, designers can build and debug hardware interfaces to controllers, sensors, or actuators that connect to an INSTEON network. The isolated power supply for this module ensures that no dangerous voltages are exposed.

The non-isolated version of the powerline development module is only intended for those experts who are developing products that must achieve the lowest possible cost while still communicating over the powerline. To reduce the part count, the power supply is directly connected to the 110-volt mains, so potentially lethal voltages are exposed. Use of this module requires signing a liability waiver.

The RF development module contains a special RF daughter board extending from a PLC. With this module, developers can create products that communicate via RF, and only optionally communicate via the powerline. RF-only devices can be battery operated, so this module is particularly designed with developers of handheld INSTEON devices in mind.

Conclusion

"Everything should be made as simple as possible, but not simpler."

Albert Einstein (1879-1955)

Electronic Home Improvement™ is poised to become a major industry of the twenty-first century. Three-quarters of homes in the U.S. now have computers and over two-thirds of those are connected to the Internet. WiFi wireless networking is in 17% of homes. High-def TV is gaining momentum. But light switches, door locks, thermostats, smoke detectors, and security sensors cannot talk to one another. Without an infrastructure networking technology, there can be no hope for greater comfort, safety, convenience, and value brought about through interactivity. Homes will remain unaware that people live in them.

For a technology to be adopted as infrastructure, it must be simple, affordable, and reliable. Not all technology that gets developed gets used. Sadly, a common pitfall for new technology is overdesign—engineers just can't resist putting in all the latest wizardry. But with added performance, cost goes up and ease-of-use goes down.

Simplicity is the principal asset of INSTEON. Installation is simple—INSTEON uses existing house wiring or the airwaves to carry messages. INSTEON needs no network controller—all devices are peers. Messages are not routed—they are simulcast. Device addresses are assigned at the factory—users don't have to deal with network enrollment. Device linking is easy—just press a button on each device and they're linked.

Simplicity ensures reliability and low-cost. INSTEON is not intended to transport lots of data at high speed—reliable command and control is what it excels at. INSTEON firmware, because it is simple, can run on the smallest microcontrollers using very little memory—and that means the lowest-possible cost.

Developing applications for INSTEON-networked devices is also simple. Designers do not have to worry about the details of sending and receiving INSTEON messages, as laid out above, because those functions are handled in firmware. Application developers can use a simple scripting interface (Home Network Language™) and Smarthome's Device Manager to further simplify the interface to a network of INSTEON devices. Designers who wish to create new kinds of INSTEON devices can write the software for them using Smarthome's Integrated Development Environment and the SALad embedded language.

Although INSTEON is simple, that simplicity is never a limiting factor, because INSTEON Bridge devices can connect to outside resources such as computers, the Internet, and other networks whenever needed. SALad-enabled INSTEON devices can be upgraded at any time by downloading new SALad programs. Networks of INSTEON devices can evolve as the marketplace does.

Smarthome's mission is to make life more convenient, safe and fun. INSTEON provides the infrastructure that can make that dream come true. Anyone can now create products that interact with each other, and with us, in remarkable new ways. What an interesting world it will be!

NOTES

1. Battery operated INSTEON RF devices, such as security sensors and handheld remote controls, must conserve power. Accordingly, they may optionally be configured so that they do not retransmit INSTEON messages from other INSTEON devices, but act as message originators only. Such devices can nevertheless both transmit and receive INSTEON messages, in order to allow simple setup procedures and to ensure network reliability. See *RF Simulcasting* for more information.
2. At a minimum, X10 compatibility means that INSTEON and X10 signals can coexist with each other on the powerline without mutual interference. INSTEON-only powerline devices do not retransmit or amplify X10 signals. But X10 compatibility also means that designers are free to create hybrid INSTEON/X10 devices that operate equally well in both environments. By purchasing such hybrid devices, current users of legacy X10 products can easily upgrade to INSTEON without making their X10 investment obsolete. See *X10 Compatibility* for more information.
3. Firmware in the INSTEON Engine handles the CRC byte automatically, appending it to messages that it sends, and comparing it within messages that it receives. Applications post messages to and receive messages from the INSTEON Engine without the CRC byte being appended. See *Message Integrity Byte* for more information.