PRIME Host Message Protocol Specification

Version 1.0000
December 18, 2012

Revision History

Revision	Status	Date	Comment
1.0	Final	12/18/2012	Finial Revision

Table of Contents

1.0 Introduction

This document describes the Application Specific Interface (APIs) during run time of TI Power Line Communication Device. The API can be implemented in a host MCU (or on a PC) that also implements some custom application features and requires power line communications.
The messaging protocol works on hardware peripheral interfaces such as UART, SPI, USB, Ethernet, etc.

1.1 System Overview

The default PLC UART settings are:

Baud Rate = 57600
Data Bits = 8
Stop Bits = One
Parity = None
Handshake = None
RTS Enable = true

2.0 Message Formats and APIs

2.1 Message Formats

Figure 1 shows the message format that is used for host to PLC communications. Little Endian format (least significant octet first, most significant octet last) is used. All messages are 16-bit aligned. Messages are padded with 0 to 1 octets if necessary to meet the 16-bit alignment. A pad octet contains value '0' (zero).

Messages are formatted as: message header + message body + (padding) as described in Figure 1.

Message Header	Message Body	(0-1) padding octets

Figure 1 Message Format

Figure 2 shows the message header format.

Offset	Length Octets	bits							
		7	6	5	4	3	2	1	0
0	1	Message Type							
1	1	ORG	RESV			SEQ			
2	1	Message Payload Length LSB							
3	1	Message Payload Length MSB							
4	2	Message Header CRC16							
6	2	Message Payload CRC16							

Figure 2 Message Header Format

Field	Description
Message Type	Message Type identifier.
ORG	Message origination. A value of 1 represents the message originated from the host, and value of 0 represents the message originated from the receiver.
RESV	Reserved bits, ignored.
SEQ	Sequence Number. This is the 4 bit message sequence number, starting with 0 and incrementing by 1 for each message sequence. The Sequence Number should be set to 0 unless specified in the message definition.
Message Payload Length	A 16 bit number representing the length of the message payload plus the length of the Message Header CRC16 and Message Payload CRC16 (a message with no payload will have a length of 4). The maximum message payload length supported is 1500Bytes.
Message Header CRC16	The CRC16 of the message header, from the Message Type to the Message Payload Length, but not including the Message Header CRC or the Message Payload CRC. ***A value of 0 denotes no CRC16 present***. See CRC16 implementation in the appendix for CRC details.
Message Payload CRC16	The CRC16 of the message payload data. ***A value of 0 denotes no CRC16 present***. See CRC16 implementation in the appendix for CRC details.

Figure 3 Message Header Fields Description

2.2 API List

Table 1 defines the host APIs supported by the TI PLC device.

Message Type	PRIME Standard	Description
0x00	DATA TRANSFER	Application specific Data Transfer messages
0x01	GET_SYSTEM_INFO	Get system (HW/SW) info
0x02	GET_PHY_PIB	Get PHY PIB attributes from PLC device
0x03	GET_MAC_PIB	Get MAC PIB attributes from PLC device
0x04	SET_INFO	Set system configuration parameters to PLC device
0x05	SHUTDOWN	Reset PLC device
0x06	SETUP_ALARM	Setup alarm notifications
0x07	ALARM	Alarm Notification
0x0c	LOAD_SYSTEM_CONFIG	Load system configuration data
0x0d	SET_MAC_PIB	Set MAC PIB attributes for PLC device
0x0e	CLEAR_PHY_PIB	Clear PHY PIB attributes.
0x0f	CLEAR_MAC_PIB	Clear MAC PIB attributes.
0x10	ATTACH	PRIME CL-432 Establish Request and Confirm
0x11	DETACH	PRIME CL-432 Release Request and Confirm
0x13	FIRMWARE_UPGRADE	FW Upgrade process.
0x14	Get Info	Gets miscellaneous PLC data
0x15 - 0x2F	Reserved	
0x30 - 0x3F	Reserved	
0x80 - 0xfe	Firmware Flash messages	
0xff	Reserved	

Table 1 TI PLC Device Host Commands/APIs

2.3 API Specifications

2.3.1 DATA_TRANSFER – Message Type 0x00

The DATA_TRANSFER message is used to transfer IEC61334-4-32 LLC Layer data to/from the host. DATA_TRANSFER message can be either initiated by host (TX) or by PLC (RX).

1. Message Sequence for Data Transmit from host to PLC in Push Mode:

DATA_TRANSFER.Request: (message from host to PLC)
DATA_TRANSFER.Request is used to command the PLC to transmit an L_SDU to the media.

Length Octets	Bits 7	6	5	4	3	2	1	0
2	Mode							
1	Destination LSAP							
1	Source LSAP							
2	Destination Address							
4	reserved							
var	Data							

Field	Description
Mode	A value of 0x01 defines this as a DATA_TRANSFER.Request
Destination LSAP	The Destination LSAP where the data is being sent to.
Source LSAP	The Source LSAP where the data is originating from.
Destination Address	The 12-bit Destination Address where the data is being sent to.
reserved	Reserved field.
Data	L_SDU data to be sent.

DATA_TRANSFER.Confirm: (message from PLC to host)

The DATA_TRANSFER.Confirm is used to indicate to the host the status of the L_SDU transmission.

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	Mode							
1	Destination LSAP							
1	Source LSAP							
2	Destination Address							
4	reserved							
2	Status							

Field	Description
Mode	A value of 0x02 defines this as a DATA_TRANSFER.Confirm
Destination LSAP	The Destination LSAP where the data was sent to.
Source LSAP	The Source LSAP where the data originated from.
Destination Address	The 12-bit Destination Address where the data was sent to.
reserved	Reserved field.
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

2. Message Sequence for Data Receive initiated by PLC:

DATA_TRANSFER.Indication: (message from PLC to host)

DATA_TRANSFER.Indication is used to indicate an L_SDU has been received and passed to the host.

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Mode							
1	Destination LSAP							
1	Source LSAP							
2	Destination Address							
4	reserved							
2	Source Address							
4	reserved							
var	Data							

Field	Description
Mode	A value of 0x03 defines this as a DATA_TRANSFER.Indication
Destination LSAP	The Destination LSAP where the data was sent to.
Source LSAP	The Source LSAP where the data originated from.
Destination Address	The 12-bit Destination Address where the data was sent to.
reserved	Reserved field.
Source Address	The 12-bit Source Address where the data was sent from.
reserved	Reserved field.
Data	L_SDU data that was sent.

2.3.2 GET SYTSTEM INFO – Message Type 0x01

The GET_SYSTEM_INFO message is used to retrieve information from the PLC device. It is always initiated by the host.

GET_SYSTEM_INFO Request:

This message does not have message body.

GET_SYSTEM_INFO Response:

Len	Offset	PRIME (LLC Mode)
4	0	Firmware Version
2	4	Device Serial Number Length
16	6	Device Serial Number
1	22	Device Type
1	23	Device Mode
2	24	Hardware Revision
6	26	EUI/MAC Address
1	32	Port Assignments
1	33	PHY Settings
6	34	Reserved
1	40	ARQ and PAC Flags
1	41	Security Profile
2	42	Reserved
2	44	Reserved
2	46	Reserved
48		Total Bytes in message

Table 2 System Configuration Info Field Definitions

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Firmware Version</td><td>A 32-bit value representing the firmware version, in the format of <code>MAJOR.MINOR.REVISION.BUILD</code>.</td></tr>
<tr><td>Serial Number Length</td><td>The length of valid octets in the 16 octet serial number.</td></tr>
<tr><td>Serial Number</td><td>A 16 octet value representing the device serial number.</td></tr>
<tr><td>Device Type</td><td>The device type. The following are the supported device types:
<table>
<tr><td>Device Type</td><td>Description</td></tr>
<tr><td>1</td><td>PRIME IEC-432-LLC convergence</td></tr>
<tr><td>2</td><td>PRIME IP convergence</td></tr>
<tr><td>3</td><td>G3</td></tr>
<tr><td>4</td><td>FLEX</td></tr>
<tr><td>5</td><td>FLEX Lite</td></tr>
<tr><td>6</td><td>FLEX OFDM</td></tr>
<tr><td>7</td><td>Reserved</td></tr>
<tr><td>8</td><td>P1901.2</td></tr>
</table></td></tr>
<tr><td>Device Mode</td><td>Device Mode dependanting on the Device Type is as follows:
<table>
<tr><td>Device Mode</td><td>Value</td><td>Description</td></tr>
<tr><td>Normal Mode</td><td>0</td><td>Device is operating in normal mode</td></tr>
<tr><td>AppEmu Mode</td><td>1</td><td>Device is operating in Embedded AppEMU MAC mode</td></tr>
<tr><td>LLC Mode</td><td>2</td><td>Device is operating in Embedded AppEMU LLC mode</td></tr>
<tr><td>P2P Mode</td><td>3</td><td>Device is operating in Point-to-Point test mode</td></tr>
<tr><td>Reserved</td><td>4</td><td></td></tr>
<tr><td>Reserved</td><td>5</td><td></td></tr>
<tr><td>Reserved</td><td>7</td><td></td></tr>
</table></td></tr>
<tr><td>Hardware Revision</td><td>The hardware revision. A value of zero (0) represents pre Rev D hardware, otherwise the revision is the hardware revision (Rev D would be indicated by 68 (0x44, ‘D’)).

Note: TI Rev D EVM has an onboard EEPROM, older hardware does not.</td></tr>
<tr><td>EUI</td><td>Service Node EUI or MAC address</td></tr>
</table>

<table>
<tr><td rowspan="3">Port Assignments</td><td colspan="8"><table><tr><td></td><td></td><td></td><td></td><td></td><td>Diag Port</td><td></td><td>Data Port</td></tr></table></td></tr>
<tr><td colspan="8">Data Port (for application data) designation.

Value Description
0 SCI-A
1 SCI-B</td></tr>
<tr><td colspan="8">Diagnostics Port (for diagnostic data) designation.

Value Description
0 SCI-A
1 SCI-B</td></tr>
<tr><td rowspan="3">PHY Settings</td><td colspan="8"><table><tr><td></td><td></td><td>Auto Mode</td><td></td><td></td><td>PRM</td><td></td><td></td></tr></table></td></tr>
<tr><td colspan="8">Auto Mode:
Flag that controls whether, in Device Normal Mode (Device Mode 0) the device attempts to automatically register and connect on the network, or if the Host manually controls the start with CL_Establish and CL_Release.

Value Description
0 Manual Mode (default)
1 Automatic Mode</td></tr>
<tr><td colspan="8">PRM:

Value Description
0 PRM off (default)
1 PRM on</td></tr>
<tr><td rowspan="2">ARQ and PAC Flags</td><td colspan="8"><table><tr><td></td><td></td><td></td><td></td><td></td><td></td><td>PAC</td><td>ARQ</td></tr></table>
MAC ARQ:

Value Description
0 MAC ARQ disabled (default)
1 MAC ARQ enabled</td></tr>
<tr><td colspan="8">MAC PAC:

Value Description
0 MAC PAC disabled (default)
1 MAC PAC enabled</td></tr>
<tr><td>Security Profile</td><td colspan="8">MAC default security profile</td></tr>
</table>

2.3.3 GET_PHY_PIB – Message Type 0x02

Message Type 0x02

The GET_PHY_PIB message is used by host to retrieve PLC PHY PIB information on the device. This message is always initiated by the host. An array of PHY PIB IDs are sent to the device, and the device will reply will an array of PHY PIB TLVs.

Message Sequence initiated by host:

GET_PHY_PIB.Request:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Attribute ID							
	[. . .]							

Field	Description
Attribute ID	One or more Attribute IDs, as defined in Tables 1 - 2 (Section 6.1.1.1 - 6.1.1.2) of the PRIME Spec v1.3E.

GET_PHY_PIB.Reply:

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	Status							
2	Attribute ID							
2	Attribute Length							
var	Attribute Value							
	[. . .]							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.
Attribute ID	Attribute ID, as defined in Tables 1 - 2 (Section 6.1.1.1 - 6.1.1.2) of the PRIME Spec v1.3E.
Attribute Length	Attribute Length for the ID, as defined in Tables 1 - 2 (Section 6.1.1.1 - 6.1.1.2) of the PRIME Spec v1.3E.
Attribute Value	Attribute Value for the ID, as defined in Tables 1 - 2 (Section 6.1.1.1 - 6.1.1.2) of the PRIME Spec v1.3E.

2.3.4 GET_MAC_PIB – Message Type 0x03

The GET_MAC_PIB message is used by host to retrieve PLC MAC PIB information on the device. This message is always initiated by the host. An array of MAC PIB IDs are sent to the device, and the device will reply will an array of MAC PIB TLVs.

Message Sequence initiated by host for PRIME:

GET_MAC_PIB.Request:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Attribute ID							
	[. . .]							

Field	Description
Attribute ID	One or more Attribute IDs, as defined in Tables 3 - 7 (Section 6.1.2.1 – 6.1.2.4) of the PRIME Spec v1.3E.

GET_MAC_PIB Reply:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Status							
2	Attribute ID							
2	Attribute Length							
var	Attribute Value							
	[. . .]							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.
Attribute ID	Attribute ID, as defined in Tables 3 - 7 (Section 6.1.2.1 – 6.1.2.4) of the PRIME Spec v1.3E.
Attribute Length	Attribute Length for the ID, as defined in Tables 3 - 7 (Section 6.1.2.1 – 6.1.2.4) of the PRIME Spec v1.3E.
Attribute Value	Attribute Value for the ID, as defined in Tables 3 - 7 (Section 6.1.2.1 – 6.1.2.4) of the PRIME Spec v1.3E.

The following PRIME MAC PIBs are TI specific and not defined in PRIME Spec v1.3E

Attribute Name	**Size (bytes)**	**Id**	**Units**	**Allowed values**
Piggyback ARQ	1	0x001F	80 ms	Values should be 0, 1, 2, 3, 4, and 5 in 80 millisecond steps for piggyback ARQ waittime. 0 means not enabled. Default is 1.
SAR Size	2	0x8001	Bytes	Segmentation size can be set to: 64, 128, 192, 256. Default = 64
Control Packet Priority	2	0x8002	N/A	0, 1, 2, 3. Zero is the highest priority. The default = 1

2.3.5 SET_INFO – Message Type 0x04

The SET_INFO message is used by host to set system parameters in the PLC device. This message is always initiated from host.

Message Sequence initiated by host:

SET_INFO.Request:

Length Octets	bits							
	7	6	5	4	3	2	1	0
2	INFO_Type							
2	INFO_Length							
var	INFO_Value							

Field	Description
INFO_Type	Type of information being set.
INFO_Length	Length of INFO_Value, in octets.
INFO_Value	Value depending on the INFO_Type.

INFO_Type - TLV IDs

Id	*Description*
0x0000	PRIME PHY TX Parameters
0x0001	PRIME PHY RX Parameters
0x0002	Reserved
0x0003	Reserved
0x0004	Vendor and Product Id's
0x0005	Reserved
0x0006	Reserved
0x0007	Reserved
0x0008	Reserved
0x0009	TX PHY Attenuation / Gain Parameters
0x000A	Reserved

INFO TYPE - TLV:

0x0000 – PRIME PHY TX Parameters:

Length Octets	bits							
	7	6	5	4	3	2	1	0
1	reserved							
1						PRM	Resv	FEC
2	Modulation							
2	TX Level							

Field	Description
FEC	A value of 1 for FEC ON, value of 0 for FEC OFF.
PRM	A value of 1 to set PHY to PRM mode, value of 0 for non designated mode. Note that if PRM mode is enabled, Modulation and TX Level are ignored.
Modulation	TX Modulation: Value Description 0x0000 DBPSK 0x0001 DQPSK 0x0002 D8PSK 0x0003 Reserved 0x0004 Reserved
TX Level	Transmission Level, 0-7 (refer to PRIME spec v1.3).

0x0001 – PRIME PHY RX Parameters:

Length	bits							
Octets	7	6	5	4	3	2	1	0
1	reserved							
1							Resv	AGC
2	Gain Value							

Field	Description
Gain Value	Manual set gain value of 0 – 7 if AGC = 0. If AGC = 1, this field is ignored.
AGC	Automatic Gain Control. A value of 1 for AGC ON, value of 0 for AGC OFF

0x0004 – Vendor/Product ID:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Vendor ID							
16	Product ID							

Field	Description
Vendor ID	Vendor ID (assigned by Prime Alliance for PRIME)
Product ID	Vendor Specifics Identifier in ASCII, terminating with a NULL character

0x0009 – TX PHY Attenuation/Gain Parameters:

Length	bits							
Octets	7	6	5	4	3	2	1	0
1	TX Gain / Attenuation							
1	TX PGA Attenuation							

Field	Description
TX Gain / Attenuation	The range is from -6 dB (-60 or 0xC4) to 3dB (30 or 0x1E) in 0.1 dB steps
TX PGA Attenuation	0 = 0 dB 1 = -3 dB 2 = -6 dB

SET_INFO Reply:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Status							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

2.3.6 SHUT_DOWN – Message Type 0x05

The Shutdown message is used to shutdown or reset the device.

Message Sequence initiated by host

SHUT_DOWN Request:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Reset_Type							

Field	Description
Reset_Type	The type of shutdown or reset to perform. Value — Description 0x0000 — Soft reset 0x0001 — Soft shutdown

SHUT_DOWN Reply:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Status							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

2.3.7 SETUP_ALARM – Message Type 0x06

The SETUP_ALARM message is used by host to set up alarm or event notifications from device to the host.

Message Sequence initiated by host:

SETUP_ALARM Request:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Alarm Type							
2	Alarm Length							
var	Alarm Value							
	. . .							

Field	Description
Alarm Type	The type of the alarm that is set. Value — Description 0x0001 — Network Deregistered 0x0002 — Reserved 0x0003 — Reserved 0x0004 — Network Registration Started 0x0005 — Network Registration Complete 0x0040 — PHY received PPDU header CRC fail detected 0x0041 — PHY received PPDU header syntax error detected 0x0080 — Reserved 0x0081 — Reserved 0x00c0 — CL Alarm 0x00ff — General Error
Alarm Length	If SET_ALARM requires additional data, this defines the length of the data (in octets). For this message the value is one.
Alarm Value	Additional SET_ALARM data.

ALARM VALUE:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
1								CLR

Field	Description
Clear	Set to 1 to clear the ALARM SETUP, i.e., the host no longer wants the ALARM notification.

SET_ALARM.Reply:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Status							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

2.3.8 ALARM – Message Type 0x07

The ALARM message is used by PLC to notify the host of an ALARM previously requested by the SETUP_ALARM message. It is an ALARM_INDICATION messaging that contains the alarm type as well as specific alarm information, in TLV (type, length and value) format.

Message Sequence initiated by device:

ALARM Message Sequence (initiated by PLC):

ALARM:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Alarm Type							
2	Alarm Length							
Var	Alarm Value							

Field	Description
Alarm Type	The type of the alarm that is requested. Value — Description 0x0001 — Network Deregistered 0x0002 — Reserved 0x0003 — Reserved 0x0004 — Network Registration Started 0x0005 — Network Registration Complete 0x0040 — PHY received PPDU header CRC fail detected 0x0041 — PHY received PPDU header syntax error detected 0x0080 — Reserved 0x0081 — Reserved 0x00c0 — CL Alarm 0x00ff — General Error
Alarm Length	If an ALARM_INDICATION has a value to return, this defines the length of the value (in octets).
Alarm Value	See the following definitions.

ALARM VALUE:

0x0001 – Network Deregistered:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Reason							

Field	Description
Reason	The reason for the deregistration 0x00 – deregistration initiated by remote node 0x01 – deregistration initiated by local node (connectivity problem)

0x0004 – Network Register Started
0x0005 – Network Register Complete
0x0040 – PHY received PPDU header CRC fail detected:
0x0041 – PHY received PPDU header syntax error detected:
No additional data.

0x00c0 – CL Alarm:

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	CL Alarm							
var	Alarm Data							

Field	Description
Alarm	CL Alarm. Type – Description 0x0001 – CL Attach Success (only valid when auto mode is set) 0x0002 – CL Detach (connection close)
Alarm Data	For Type = 0x0001 Alarm data is as following. Refer 2.3.13 for field definitions.

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	Status							
2	Device Identifier Length							
var	Device Identifier							
2	Source Address							
2	Base Address							

For Type = 0x0002 Alarm data is as following. Refer 2.3.14 for field definitions.

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	Status							
2	Destination Address							

0x00ff – General Error:

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	Error							
var	Information							

Field	Description
Error	Error Code. Type – Description 0x0001 – The port receiving HCT messages has discarded data. 0x0003 – An HCT message header includes unsupported feature. 0x0004 – An HCT message has been discarded by the device port. 0x0008 – An HCT message has invalid message format.
Information	Extended error information, dependant on the error. **Note that type 0x0003 and 0x0008 does not include "Information" field.**

0x0001 – PORT_DISCARDED:

The port receiving messages has discarded data. This is normally due to a port communications error.

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Number of octets discarded							

Field	Description
Number of octets discarded	Number of octets discarded from the port, usually due to the attempt to resync to the message framing or other port error.

0x0004 – MESSAGE_DISCARDED:

The port receiving messages has discarded a message. This is normally due to a communications error.

Length	Bits							
Octets	7	6	5	4	3	2	1	0
1	Message Type							
1	ORG	RPY	RESV		SEQ			
2	Reason							

Field	Description
Message Type	Discarded Message Type.
ORG, RPY, RESV, SEQ	Discarded message ORG, RPY, RESV and SEQ.
Reason	The reason for the discard. The list of status codes can be found at Table 4 Message Status Code Definitions.

2.3.9 LOAD_SYSTEM_CONFIG – Message Type 0x0C

The LOAD_SYSTEM_CONFIG message is used by the host to load system configuration parameters at the start of the system. After system starts (registered, etc), this message is not allowed.

Message Sequence initiated by host:

LOAD_SYSTEM_CONFIG Request:

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	Config Type							
2	Config Length							
Var	Configuration							
	[…]							

Field	Description
Config Type	The type of configuration. The following are the supported types: Value — Description 0x0001 — Port designation 0x0003 — System Configuration 0x0004 — PHY Configuration 0x0005 — MAC Configuration 0x0006 — Reserved 0x0007 — PRIME LLC Configuration 0x0008 — Reserved 0x0009 — Reserved 0x000A — Reserved 0x000B — Reserved
Config Length	Length of configuration.
Configuration	Configuration, of length Config_Length.

0x0001 – Port designation:

Length	bits							
Octets	7	6	5	4	3	2	1	0
1					Diag Port		Data Port	

Field	Description
Data Port	Data Port designation. Value – Description 0 – SCI-A 1 – SCI-B
Diag Port	Diagnostics Port designation. Value – Description 0 – SCI-A 1 – SCI-B

0x0003 – System Configuration:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Serial Number Length							
16	Serial Number							
6	EUI							
1	Device Mode							
1			Auto Mode					

Field	Description
Serial Number Length	Service Node Serial Number Length. This specifies how many octets in the 16 octet Serial Number are valid.
Serial Number	Service Node Serial Number.
EUI	Service Node EUI
Device Mode	Refer Table 2 System Configuration Info Field Definitions
Auto Mode	Flag that controls whether, in Device Normal Mode (Device Mode 0) the device attempts to automatically register and connect on the network, or if the Host manually controls the start with ATTACH and DETACH.

0x0004 – PRIME PHY configuration:

Length	bits							
Octets	7	6	5	4	3	2	1	0
1						PRM	PHY Mode	

Field	Description
PHY Mode	PPDU Header Mode. Value – Description 0 – Prime Mode (default) 1 – Reserved
PRM Flag	PRM Off or On Value – Description 0 – PRM Off (default) 1 – PRM On

0x0005 – PRIME MAC configuration:

Length	bits							
Octets	7	6	5	4	3	2	1	0
6	Reserved							
1	MAC_Default_Security_Profile							
1							PAC	ARQ

Field	Description
MAC_Default_Security_Profile	The default MAC security profile. Valid range: 0-255
ARQ	A value of 1 represents that MAC ARQ is enabled by default, a value of 0 indicates MAC ARQ is disabled by default.
PAC	A value of 1 represents that MAC PAC is enabled by default, a value of 0 indicates MAC PAC is disabled by default.

0x0007 – LLC configuration:

Configuration Data:

Length Octets	bits							
	7	6	5	4	3	2	1	0
2	LLC Source LSAP							
2	LLC Default Destination LSAP							
2	LLC Default Destination Node Address							

Field	Description
LLC Source LSAP	The Source LSAP where the data is originating from.
LLC Default Destination LSAP	The Destination LSAP where the data is being sent to.
LLC Default Destination Node Address	The 12-bit Destination Address where the data is being sent to.

LOAD_SYSTEM_CONFIG Reply:

Length Octets	Bits							
	7	6	5	4	3	2	1	0
2	Status							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

2.3.10 SET_MAC_PIB – Message Type 0x0D

The SET_MAC_PIB message is used to set PLC PIB information on the device. This message is always initiated from host. An array of PIB TLV's is sent to the device.

Message Sequence initiated by host for PRIME:

SET_ PIB Request:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Attribute ID							
2	Attribute Length							
var	Attribute Value							
	[. . .]							

Field	Description
Attribute ID	Attribute ID, as defined in Tables 1 - 7 (Section 6.1.1.1 - 6.1.1.2, 6.1.2.1 – 6.1.2.4, 6.1.2.5) of the PRIME Spec v1.3E.
Attribute Length	Attribute Length for the ID, as defined in Tables 1 - 7 (Section 6.1.1.1 - 6.1.1.2, 6.1.2.1 – 6.1.2.4, 6.1.2.5) of the PRIME Spec v1.3E.
Attribute Value	Attribute Value for the ID, as defined in Tables 1 - 7 (Section 6.1.1.1 - 6.1.1.2, 6.1.2.1 – 6.1.2.4, 6.1.2.5) of the PRIME Spec v1.3E.

SET_ PIB Reply:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Status							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

The following PRIME MAC PIBs are TI specific and not found in PRIME Spec v1.3E.

Attribute Name	Size (bytes)	Id	Units	Allowed values
Piggyback ARQ	1	0x001F	80 ms	Values should be 0, 1, 2, 3, 4, and 5 in steps of 80ms. 0 means no piggyback ARQ. Default = 1
SAR Size	2	0x8001	Bytes	64, 128, 192, 256. Default = 64
Control Packet Priority	2	0x8002	N/A	0, 1, 2, 3. Zero is the highest priority. The default = 1

2.3.11 CLEAR_PHY_PIB – Message Type 0x0E

Message Type 0x0e

The CLEAR_PHY_PIB message is used by host to clear certain PLC PHY PIB information on the device. This message is always initiated by the host. An array of Attribute IDs is used to list the PHY PIBs to be cleared.

Message Sequence initiated by host:

CLEAR_PHY_PIB Request:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Attribute ID							
	[. . .]							

CLEAR_PHY_PIB Reply:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Status							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

2.3.12 CLEAR_MAC_PIB – Message Type 0x0F

The CLEAR_MAC_PIB message is used by host to clear certain PLC MAC PIB information on the device. This message is always initiated by the host. An array of Attribute IDs is used to list the MAC PIBs to be cleared.

Message Sequence initiated by host for PRIME:

CLEAR_MAC_PIB Request:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Attribute ID							
	[. . .]							

Field	Description
Attribute ID	Attribute ID, as defined in Tables 3 - 7 (Section 6.1.2.1 – 6.1.2.4) of the PRIME Spec v1.3E.

CLEAR_MAC_PIB Reply:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Status							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.

2.3.13 ATTACH – Message Type 0x10

The ATTACH message is used by the host to initiate the process of registering the device identifier with the Base Node and the Base Node allocates the destination address to the service node session.

Message Sequence initiated by host:

ATTACH.Request:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Device Identifier Length							
var	Device Identifier							

Field	Description
Device Identifier Length	Length of the Device Identifier.
Device Identifier	Equivalent to COSEM “logical device name” which is defined as an octet-string of up to the Device Identifier Length.

ATTACH.Confirm:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Status							
2	Device Identifier Length							
var	Device Identifier							
2	Source Address							
2	Base Address							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.
Device Identifier Length	Length of the Device Identifier.
Device Identifier	Used to identify which ATTACH.request this ATTACH.confirm is for, of length Device Identifier Length..
Source Address	The address assigned to the 4-32 CL session by the base node (12-bit)
Base Address	Base node address (12-bit)

2.3.14 DETACH – Message Type 0x11

The DETACH message is used by host to leave the network, close the convergence layer session and any resources it may hold.

Message Sequence initiated by host:

DETACH.request:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Destination Address							

Field	Description
Destination Address	The address allocated to the service node 4-32 session by the base node.

DETACH.confirm:

Length	bits							
Octets	7	6	5	4	3	2	1	0
2	Status							
2	Destination Address							

Field	Description
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.
Destination Address	The address allocated to the service node 4-32 session by the base node.

2.3.15 FW_UPGRADE – Message Type 0x13

The follow describes the message sequence of the FW_UPGRADE and its Message Payload Definitions.

Message Sequence initiated by PLC:

1. FIRMWARE_UPGRADE.preperation_request: PLC to host

Length Octets	Bits 7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0001							
2	Page Size							
4	Image Size							
4	Image CRC							

Field	Description
Page Size	The PLC receives a firmware image in fixed sized pages (except for the last page). The page size can be one of 32, 64, 128 or 192 bytes. The page size value is provided to the PLC by the base node at the beginning of a firmware upgrade session.
Image Size	The size of the "upgrade" image in bytes. This is also provided to the PLC by the base node at the beginning of a firmware upgrade session.
Image CRC	CRC of the "upgrade" image. This is also provided to the PLC by the base node at the beginning of a firmware upgrade session.

2. FIRMWARE_UPGRADE.preperation_reply: host to PLC

Length Octets	Bits 7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0001							
2	Status							

Field	Description
Status	0x0000 - SUCCESS 0x0004 - PAYLOAD_CRC_ERROR 0x6001 - Host does not support firmware upgrade

3. FIRMWARE_UPGRADE.data_indicate: PLC to host

Length Octets	Bits 7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0000							
4	Page Index							
Var	Page contents							

Field	Description
Page Index	The position of this page within the firmware image.
Page Contents	The contents of the page as an array of bytes

4. FIRMWARE_UPGRADE.data_reply: host to PLC

Length Octets	Bits 7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0000							
2	Status							
4	Page Index							
2	CRC Valid Flag							
4	CRC							

Field	Description
Status	0x0000 - SUCCESS 0x0004 - PAYLOAD_CRC_ERROR 0x6001 - Host does not support firmware upgrade 0x6006 - Host not prepared for firmware upgrade 0x6007 – Length of the “page contents” field is invalid.
Page Index	The position of this page within the firmware image.
CRC Valid Flag	1 – host set CRC field. 0 – CRC field not valid
CRC	CRC calculated over received FW image. Only valid when CRC Valid Flag is set to non-zero.

Step 3. Step 4 will repeat until the last page of the FW is received.

5. FIRMWARE_UPGRADE.crc_request: PLC to Host
   This message requests the host to report the CRC status.

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0002							

6. FIRMWARE_UPGRADE.crc_reply: Host to PLC

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0002							
2	Status							
4	CRC							

Field	**Description**
Status	0x0000 - SUCCESS 0x6001 - Host does not support firmware upgrade 0x6002 - Host does not have an "upgrade" image (to flash and reboot) 0x6003 - "Upgrade" image is incomplete 0x6004 - "Upgrade" image did not pass CRC check 0x6005 - "Upgrade" image is bad/Incompatible
CRC	CRC of the "upgrade" image as calculated by the host.

7. FIRMWARE_UPGRADE.flash_request: PLC to Host
   This message requests the host to flash the received FW image to the PLC sub-system.

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0003							

8. FIRMWARE_UPGRADE.flash_reply: Host to PLC
   This message indicates Host is ready to flash the newly received image to the PLC sub-system.

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0003							
2	Status							

Field	Description
Status	0x0000 - SUCCESS 0x6001 - Host does not support firmware upgrade 0x6002 - Host does not have an "upgrade" image (to flash and reboot) 0x6003 - "Upgrade" image is incomplete 0x6004 - "Upgrade" image did not pass CRC check 0x6005 - "Upgrade" image is bad/Incompatible

After the upgraded FW is flashed and restarted, the following message sequence is used by the PLC to request the host to mark the "upgrade" image as the "active" image. Depending on the implementation, the image marked as "active" till this point can be deleted or archived. Note that the host will no longer have an "upgrade" image after this request has been handled (since it has been marked "active").

9. FIRMWARE_UPGRADE.activate_request: PLC to Host

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0004							

10. FIRMWARE_UPGRADE.activate_reply: Host to PLC

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x0004							
2	Status							

Field	Description
Status	0x0000 - SUCCESS 0x6001 - Host does not support firmware upgrade 0x6002 - Host does not have an "upgrade" image. 0x6003 - "Upgrade" image is incomplete 0x6004 - "Upgrade" image did not pass CRC check 0x6005 - "Upgrade" image is bad/Incompatible

During the process of firmware upgrade, it is possible that the PLC Base Node to initiate a "Abort" to the upgrade. PLC will indicate to the host that the currently running firmware upgrade process has been aborted. The host is required to delete the "upgrade" image if it has one and do any required clean ups.

11. FIRMWARE_UPGRADE.abort_request:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x000F							
2	Flag							

Field	Description
Flag	0 - Do not flash the "active" image and do not reboot the PLC 1 - Flash the "active" image and reboot the PLC

12. FIRMWARE_UPGRADE.abort_reply:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Message Sub-Type = 0x000F							
2	Status							

Field	Description
Status	0x0000 - Success 0x0004 - PAYLOAD_CRC_ERROR 0x6001 - Host does not support firmware upgrade 0x6002 - Host does not have an "upgrade" image

2.3.16 GET_INFO – Message Type 0x14

The GET_INFO message is used by host to retrieve certain system parameters in the PLC device. This message is always initiated from host.
A GET_INFO TLV ID is sent to the device, and the device will reply the GET_INFO TLVs.

GET_INFO Request:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Get Info TLV ID							

GET_INFO Reply:

Length	Bits							
Octets	7	6	5	4	3	2	1	0
2	Status							
2	TLV ID							
2	TLV Length							
var	TLV Value							

Field	***Description***
Status	The status of the request. The list of status codes can be found at Table 4 Message Status Code Definitions.
TLV ID	
TLV Length	TLV Length for the ID, as defined in the following tables
TLV Value	TLV Value for the ID, as defined in the following tables

TLV IDs

Id	***Description***
0x0000 – 0x0001	Reserved
0x0002	Reserved
0x0003	Reserved
0x0004	Vendor and Product Id's
0x0005	Reserved
0x0006	Reserved
0x0007	Reserved
0x0008	Reserved
0x0009	TX PHY Attenuation / Gain
0x000A	Get Firmware CRC32 value

0x0004 – Vendor and Product Id's

Length Octets	bits							
	7	6	5	4	3	2	1	0
2	TLV Id = 0x0004							
2	Length = 18							
2	Vendor Id							
16	Product Id							

Field	***Description***
TLV Id	0x0004
Length	18
Vendor Id	Vendor Id, UINT16 value
Product Id	Product Id – ASCII characters, null terminated The max number of characters is 15

0x0009 – TX PHY Attenuation/Gain Parameters:

Length Octets	bits							
	7	6	5	4	3	2	1	0
1	TX Gain / Attenuation							
1	TX PGA Attenuation							

Field	**Description**
TX Gain / Attenuation	The range is from -60 to (0xC4) to 30 (0x 1E) in 0.1 dB steps
TX PGA Attenuation	0 = 0 dB 1 = -3 dB 2 = -6 dB
Product ID	Vendor Specifics Identifier in ASCII, terminating with a NULL character

0x000A – Firmware CRC32

Length	bits							
Octets	7	6	5	4	3	2	1	0
4	Firmware CRC32 value							

Field	Description
Firmware CRC 32 value	The CRC32 of the current DSP firmware

A CRC32 value of 0xFFFFFFFF means that no CRC32 was loaded with the DSP firmware.

2.4 Firmware Flash Messages

For FW upgrade the host needs to support the following APIs.

2.4.1 Flash Firmware Image

2.4.1.1 Flash Firmware Process

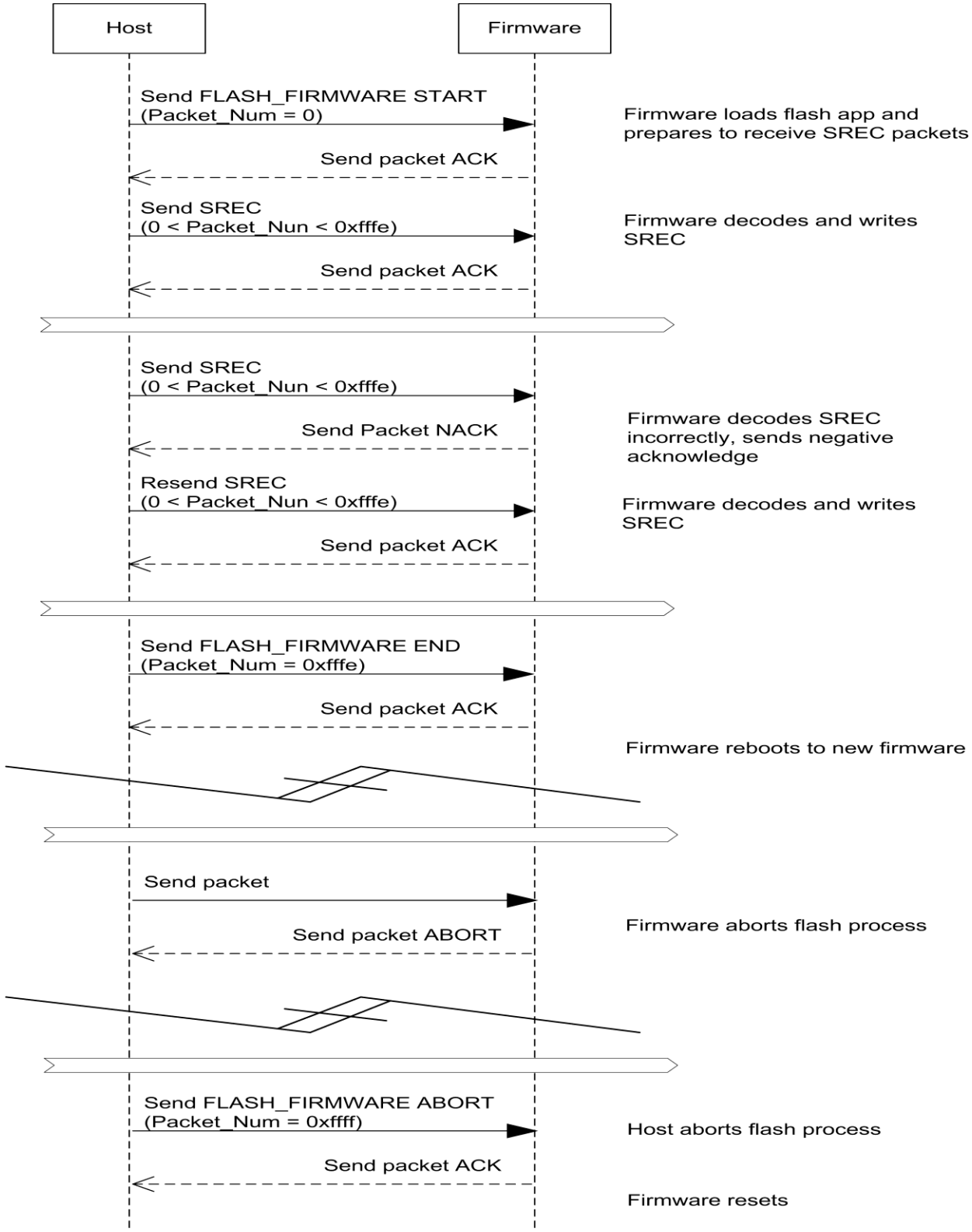
A process to flash firmware image is in place to provide an easy firmware update process as well as help minimize error conditions. Each packet sent by the host is acknowledged by the PLC before the host can send the next packet. The current packet can be resent by the host if a negative acknowledgement is received from the device.

The host can send firmware image as either SREC text format or can convert to binary format, sent as 16 bit words.

The process is as follows:

1. The host initiates the firmware flash process by sending the device a START packet. The payload is a 32bit file length (this should be the total number of bytes that will be sent).
2. The device sends an acknowledgement.
3. The host sends firmware packets (see payload type) to the device
4. For each packet, the device sends an acknowledgement.
5. The host completes the process by sending an END packet.
6. The device acknowledges the END, and reboots to the new firmware.

The following figure shows the firmware flash process.


Host
Firmware
Send FLASH_FIRMWARE START
(Packet_Num = 0)
Firmware loads flash app and prepares to receive SREC packets
Send packet ACK
Send SREC
(0 < Packet_Nun < 0xfffe)
Firmware decodes and writes SREC
Send packet ACK
Send SREC
(0 < Packet_Nun < 0xfffe)
Firmware decodes SREC incorrectly, sends negative acknowledge
Send Packet NACK
Resend SREC
(0 < Packet_Nun < 0xfffe)
Firmware decodes and writes SREC
Send packet ACK
Send FLASH_FIRMWARE END
(Packet_Num = 0xfffe)
Send packet ACK
Firmware reboots to new firmware
Send packet
Firmware aborts flash process
Send packet ABORT
Send FLASH_FIRMWARE ABORT
(Packet_Num = 0xffff)
Host aborts flash process
Send packet ACK
Firmware resets

2.4.1.2 Flash Image Messages

This diagnostic message is used to flash a firmware image to PLC. It should be noted that the message is used to allow host application to flash the new PLC image (either received from PLC network or other network) to the PLC sub-sysem.

Message sequence:

Firmware_Flash.request message

```
MSG_TYP = 0x94;
MSG_ORG = '1';  // originated from host
MSG_RPY = '0';  // reply message needed
MSG_BDY =
  uint16_t HeaderCRC16;  // CRC16 of the APE message header
  uint16_t PayloadCRC16; // CRC16 of Payload, excluding
                         //   HeaderCRC16 and PayloadCRC16
  uint16_t PacketNum;    // Packet Number, in the form of:
                         //   0x0000: Start (BOF)
                         //   0xfffe: End (EOF)
                         //   0xffff: Abort
                         //   0x0001-0xfffd: Packet Number
  uint16_t PayloadType;  // Payload Type
                         //   0x0000: SREC
                         //   0x0001: binary
  uint8_t Payload[0..256]; // Firmware File packet
```

Firmware_Flash.reply message

```
MSG_TYP = 0x94;
MSG_ORG = '0';  // originated from PLC
MSG_RPY = '1';  // end of message sequence
MSG_BDY =
  uint16_t HeaderCRC16;  // CRC16 of the host message header and PacketNum
  uint16_t PayloadCRC16; // CRC16 of Payload, excluding
                         //   HeaderCRC16 and PayloadCRC16
  uint16_t PacketNum;    // Packet Number or request
  uint16_t Status;
```

Possible status values are:

Status Code	Value	Comment	Action
SUCCESS	0	Success	Send next packet or other action depending on the current state of the firmware flash
ABORT	1	Abort Flash	If the action flash has not started then abort and do not send the flash packets. If the flash memory has been erased then the previous version of code would have to be sent to recover the modem.
BADCRC	2	Bad CRC	Resend the last packet
MISSING	3	Missing packet	A missing firmware packet was detected. Send this packet only.
ERASE_PENDING	4	The flash erase process has started	No action, status message only
ERASE_COMPLETE	5	Flash erase complete	No action, status message only. The modem will not begin saving the flash data in memory until this state has been reported.
UTILITY_MISSING	6	Flash utility missing	Abort the flash. The utility necessary for the modem to flash itself was not found and the modem can not be flashed.

2.4.1.3 FW Image Binary Payload Format

The payload for the Binary payload type is defined as follows:

```
  uint32_t Address;           // memory location address
  uint16_t Length;            // length of binary data
  uint16_t Payload[0..128];   // binary data
```

2.4.1.4 Upload Firmware Image Request

The upload firmware image messages will request the firmware image from the PLC. The PLC must be in DFU mode before the image can be transferred.

Upload Firmware Image request message

```
MSG_TYP = 0x99;
MSG_ORG = '1';  // originated from host
MSG_RPY = '0';  // reply message needed
MSG_BDY =
  Uint16_t payload_length// length of the message including crc's
  uint16_t HeaderCRC16;  // CRC16 of the APE message header
  uint16_t PayloadCRC16; // CRC16 of Payload, excluding
                         //   HeaderCRC16 and PayloadCRC16
  uint16_t PacketNum;    // Packet Number, in the form of:
```

```
                          //   0x0000: Start (BOF)
                          //   0xfffe: End (EOF)
                          //   0xffff: Abort
                          //   0x0001-0xfffd: Packet Number
    uint16_t PayloadType // Payload Type but PLC always uses binary
                          // (0x01) regardless of value
```

2.4.1.5 Upload Firmware Image Reply

Upload_Firmware_Flash reply message – Packet Number equal to zero

```
MSG_TYP = 0x99;
MSG_ORG = '0';  // originated from PLC
MSG_RPY = '1';  // end of message sequence
MSG_BDY =
  uint16_t payload_length  // length of the message including crc's
  uint16_t HeaderCRC16;    // CRC16 of the host message header and PacketNum
  uint16_t PayloadCRC16;   // CRC16 of Payload, excluding
                           //   HeaderCRC16 and PayloadCRC16
  uint16_t PacketNum;      // Packet Number or request
  uint16_t Status;         // Status = 0 for good request, see codes below

  uint16_t NumberOfBlocks  // Number of packets to upload
  uint16_t Block_Size      // Size of the packets, the last packet may be
                           // smaller
  Uint32_t NumberOfWords   // Image size to upload in words
```

Upload_Firmware_Flash reply message – Packet Number greater than zero

```
MSG_TYP = 0x99;
MSG_ORG = '0';  // originated from PLC
MSG_RPY = '1';  // end of message sequence
MSG_BDY =
  uint16_t payload_length  // length of the message including crc's
  uint16_t HeaderCRC16;    // CRC16 of the host message header and PacketNum
  uint16_t PayloadCRC16;   // CRC16 of Payload, excluding
                           //   HeaderCRC16 and PayloadCRC16
  uint16_t PacketNum;      // Packet Number or request
  uint16_t Status;         // Status = 0 for good request, see codes below
  uint32_t Address;        // Starting address of data block
  uint16_t Length;         // Length of data block
  uint8_t  Data;           // image data
```

Possible status values are:

Status Code	Value	Comment	Action
SUCCESS	0	Success	Send next packet or other action depending on the current state of the firmware flash
ABORT	1	Abort Flash	If the action flash has not started then abort and do not send the flash packets. If the flash memory has been erased then the previous version of code would have to be sent to recover the modem.
BADCRC	2	Bad CRC	Resend the last packet
MISSING	3	Missing packet	A missing firmware packet was detected. Send this packet only.
ERASE_PENDING	4	The flash erase process has started	No action, status message only
ERASE_COMPLETE	5	Flash erase complete	No action, status message only. The modem will not begin saving the flash data in memory until this state has been reported.
UTILITY_MISSING	6	Flash utility missing	Abort the flash. The utility necessary for the modem to flash itself was not found and the modem cannot be flashed.
NOT_SUPPORTED_MSG	7	This message is not supported	DFU cannot handle this message. It is also used as indication that device is in DFU mode
ERASE_ERROR	8	Has problem to erase Flash	
NO_PLC_IN_FLASH	9	Valid PLC firmware is not detected in Flash	You cannot upload the PLC firmware
LAST_PLC_RECORD	10	This is last PLC record from Flash	After receiving this packet, you should not send Flash Upload message to DFU
OUT_RANGE	11	Block number is out of range	Block number is out of valid range

2.5 Blob Read and Write Messages

2.5.1 Read Blob

This message is sent by PLC to host to read a block of binary data (Blob) from host's storage: RAM , NVRAM or flash into the PLC's local memory or vice versa. It is used to retrieve the system non-volatile information stored on host for PLC. The blob is identified by the *blob_id*. Each blob has the following definition:

```
BLOB =
  uint16_t blob_id;    // blob identifie
  uint16_t blob_total_len;  // total blob length in bytes
  uint8_t  blob_data[blob_len]; // blob data in this message
```

Note: There are no header or body CRC values in this message.

PLC can use this message to read in the configuration file that is stored in the flash or NVRAM after system is booted.

Message sequence:

read_blob request message: PLC to host

```
MSG_TYP = 0x96;
MSG_ORG = '0';  // 0: originated from PLC, 1: originated from host
MSG_RPY = '0';  // need reply message
MSG_BDY =
  uint16_t blob_id; // blob id to identify the blob
  uint16_t blob_len; // length in bytes of blob to read
  uint16_t blob_addr; //start address of the blob, only exist when blob_id=4
```

read_blob reply message: host to PLC

```
MSG_TYP = 0x96;
MSG_ORG = '1';  // 0: originated from PLC, 1: originated from Host
MSG_RPY = '1';  // end of message sequence (if no blob fragmentation)
MSG_BDY =
  uint16_t blob_id;  // blob id to uniquely identify the blob
  uint16_t blob_len; // total length in bytes of the blob data, if blob_len =
0, the blob data is either absent or not available
  uint8_t  blob_data[blob_len]; // when blob_id = 4, 1st 32-bit specifies the
blob start address.
```

The following table shows the receiver block id definitions.

Blob_id	Name
0	Configuration file
1	FW upgrade status info
2	DMEM_data_blob
3	PMEM_data_blob
4	Addressed Data blob

Table 3 blob_id Definitions

The data structure or definition of the configuration file is in host message specification.

When blob_id = 4, the 1st 32-bit in blob_data[blob_len] contains the address of the data that resides in PLC or host system. This allow host or PLC to perform random data read from address specified.

2.5.2 Write Blob

This message is sent by PLC to Host to write a block of binary data (Blob) into host's RAM, NVRAM or flash for storage purpose or vice versa. This message is being used to store the system non-volatile information to the host when there is no local NVR such as EEPROM. Message sequence:

write_blob request message:

```
MSG_TYP = 0x97;
MSG_ORG = '0';  // originated from PLC
MSG_RPY = '0';  // one block data write.
MSG_BDY =
  uint16_t blob_id;   // blob identifier refer Table 2
  uint16_t blob_total_len;  // total blob length in bytes
  uint8_t  blob_data[blob_len]; // when blob_id=4, 1st 32-bit specifies the
blob start address
```

Note: There are no header or body CRC values in this message.

write_blob reply message:

```
MSG_TYP = 0x97;
MSG_ORG = '1';  // originated from Host
MSG_RPY = '1';  // no reply message needed
MSG_BDY =
  uint16_t status;
```

Possible status values are:

Status Code	Value	Comments
OK	0	write_blob operation successful
ERR_HOST_MEM_FULL	14	there is no memory space in host that can perform this operation
ERR_CRC_CHECKSUM	15	there is CRC or check sum error in the blob data received

3.0 Appendix

3.1 Message Status Code Definition

Status Code	Value	Comments
OK	0x0000	Message request success
GENERAL_ERROR_CODES	0x0xxx	General Message Error Codes
INVALID_PARAMETER	0x0002	Invalid message parameter(s)
FEATURE_UNSUPPORTED	0x0003	Feature is not supported
PAYLOAD_CRC_ERROR	0x0004	Payload CRC error
NO_EEPROM	0x0005	No EEPROM on the device
HEADER_CRC_ERROR	0x0006	Message Header CRC error
INVALID_HANDLE	0x0007	The handle is invalid
INVALID_FORMAT	0x0008	The message format is incorrect
RPY_NOT_BLOCKED	0x000a	An RPY unblock msg (see DATA_TRANSFER) was recv'd but not currently blocked.
INVALID_MESSAGE_LENGTH	0x000b	The message length is invali
ERR_NO_HANDLER	0x00f3	No handler for the message type
PENDING	0x00fc	The operation is pending
ERR_TIMEOUT	0x00fd	The operation timed-out
ERR_NO_MEMORY	0x00fe	Out of memory
GENERAL_FAILURE	0x00ff	General failure or error
PHY_ERROR_CODES	0x1xxx	PHY error codes
MAC_ERROR_CODES	0x2xxx	MAC error codes
CL_ERROR_CODES	0x3xxx	CL error codes
LLC_ERROR_CODES	0x4xxx	LLC error codes
FU_ERROR_CODES	0x6xxxx	Firmware upgrade error codes

Table 4 Message Status Code Definitions

3.2 CRC16 implementation

The Cyclic-Redundancy Check 16bit (CRC16) implementation is a polynomial of (x^16 + x^12 + x^5 + x^0) and is calculated with a seed value of 0 and is not inverted upon completion. A sample implementation is as follows:

```
static unsigned short crctable[256] = {
  0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7,
  0x8108, 0x9129, 0xa14a, 0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef,
  0x1231, 0x0210, 0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
  0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c, 0xf3ff, 0xe3de,
  0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485,
  0xa56a, 0xb54b, 0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
  0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6, 0x5695, 0x46b4,
  0xb75b, 0xa77a, 0x9719, 0x8738, 0xf7df, 0xe7fe, 0xd79d, 0xc7bc,
  0x48c4, 0x58e5, 0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
  0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969, 0xa90a, 0xb92b,
  0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a12,
  0xdbfd, 0xcbdc, 0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
  0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03, 0x0c60, 0x1c41,
  0xedae, 0xfd8f, 0xcdec, 0xddcd, 0xad2a, 0xbd0b, 0x8d68, 0x9d49,
  0x7e97, 0x6eb6, 0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
  0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a, 0x9f59, 0x8f78,
  0x9188, 0x81a9, 0xb1ca, 0xa1eb, 0xd10c, 0xc12d, 0xf14e, 0xe16f,
  0x1080, 0x00a1, 0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
  0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c, 0xe37f, 0xf35e,
  0x02b1, 0x1290, 0x22f3, 0x32d2, 0x4235, 0x5214, 0x6277, 0x7256,
  0xb5ea, 0xa5cb, 0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
  0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,
  0xa7db, 0xb7fa, 0x8799, 0x97b8, 0xe75f, 0xf77e, 0xc71d, 0xd73c,
  0x26d3, 0x36f2, 0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
  0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9, 0xb98a, 0xa9ab,
  0x5844, 0x4865, 0x7806, 0x6827, 0x18c0, 0x08e1, 0x3882, 0x28a3,
  0xcb7d, 0xdb5c, 0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
  0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0, 0x2ab3, 0x3a92,
  0xfd2e, 0xed0f, 0xdd6c, 0xcd4d, 0xbdaa, 0xad8b, 0x9de8, 0x8dc9,
  0x7c26, 0x6c07, 0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
  0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba, 0x8fd9, 0x9ff8,
  0x6e17, 0x7e36, 0x4e55, 0x5e74, 0x2e93, 0x3eb2, 0x0ed1, 0x1ef0
};

void CRC16_UpdateChecksum(unsigned short *pcrcvalue, const void *data, int length)
{
  unsigned short crc;
  const unsigned char *buf = (const unsigned char *) data;

  crc = *pcrcvalue;
  while (length--)
  {
    crc = (crc << 8) ^ crctable[(crc >> 8) ^ *buf++];
  }
  *pcrcvalue = crc;
}

unsigned short CRC16_BlockChecksum(const void *data, int length)
{
  unsigned short crc;

  crc = 0;
  CRC16_UpdateChecksum(&crc, data, length);
  return crc;
}
```

3.3 CRC 32 implementation

The Cyclic-Redundancy Check 32bit (CRC32) implementation is calculated with a seed value of 0 and is not inverted upon completion. A sample implementation is as follows:

```
static unsigned long crctable[256] =
{
0x00000000, 0x04C11DB7, 0x09823B6E, 0x0D4326D9, 0x130476DC, 0x17C56B6B, 0x1A864DB2, 0x1E475005,
0x2608EDB8, 0x22C9F00F, 0x2F8AD6D6, 0x2B4BCB61, 0x350C9B64, 0x31CD86D3, 0x3C8EA00A, 0x384FBDBD,
0x4C11DB70, 0x48D0C6C7, 0x4593E01E, 0x4152FDA9, 0x5F15ADAC, 0x5BD4B01B, 0x569796C2, 0x52568B75,
0x6A1936C8, 0x6ED82B7F, 0x639B0DA6, 0x675A1011, 0x791D4014, 0x7DDC5DA3, 0x709F7B7A, 0x745E66CD,
0x9823B6E0, 0x9CE2AB57, 0x91A18D8E, 0x95609039, 0x8B27C03C, 0x8FE6DD8B, 0x82A5FB52, 0x8664E6E5,
0xBE2B5B58, 0xBAEA46EF, 0xB7A96036, 0xB3687D81, 0xAD2F2D84, 0xA9EE3033, 0xA4AD16EA, 0xA06C0B5D,
0xD4326D90, 0xD0F37027, 0xDDB056FE, 0xD9714B49, 0xC7361B4C, 0xC3F706FB, 0xCEB42022, 0xCA753D95,
0xF23A8028, 0xF6FB9D9F, 0xFBB8BB46, 0xFF79A6F1, 0xE13EF6F4, 0xE5FFEB43, 0xE8BCCD9A, 0xEC7DD02D,
0x34867077, 0x30476DC0, 0x3D044B19, 0x39C556AE, 0x278206AB, 0x23431B1C, 0x2E003DC5, 0x2AC12072,
0x128E9DCF, 0x164F8078, 0x1B0CA6A1, 0x1FCDBB16, 0x018AEB13, 0x054BF6A4, 0x0808D07D, 0x0CC9CDCA,
0x7897AB07, 0x7C56B6B0, 0x71159069, 0x75D48DDE, 0x6B93DDDB, 0x6F52C06C, 0x6211E6B5, 0x66D0FB02,
0x5E9F46BF, 0x5A5E5B08, 0x571D7DD1, 0x53DC6066, 0x4D9B3063, 0x495A2DD4, 0x44190B0D, 0x40D816BA,
0xACA5C697, 0xA864DB20, 0xA527FDF9, 0xA1E6E04E, 0xBFA1B04B, 0xBB60ADFC, 0xB6238B25, 0xB2E29692,
0x8AAD2B2F, 0x8E6C3698, 0x832F1041, 0x87EE0DF6, 0x99A95DF3, 0x9D684044, 0x902B669D, 0x94EA7B2A,
0xE0B41DE7, 0xE4750050, 0xE9362689, 0xEDF73B3E, 0xF3B06B3B, 0xF771768C, 0xFA325055, 0xFEF34DE2,
0xC6BCF05F, 0xC27DEDE8, 0xCF3ECB31, 0xCBFFD686, 0xD5B88683, 0xD1799B34, 0xDC3ABDED, 0xD8FBA05A,
0x690CE0EE, 0x6DCDFD59, 0x608EDB80, 0x644FC637, 0x7A089632, 0x7EC98B85, 0x738AAD5C, 0x774BB0EB,
0x4F040D56, 0x4BC510E1, 0x46863638, 0x42472B8F, 0x5C007B8A, 0x58C1663D, 0x558240E4, 0x51435D53,
0x251D3B9E, 0x21DC2629, 0x2C9F00F0, 0x285E1D47, 0x36194D42, 0x32D850F5, 0x3F9B762C, 0x3B5A6B9B,
0x0315D626, 0x07D4CB91, 0x0A97ED48, 0x0E56F0FF, 0x1011A0FA, 0x14D0BD4D, 0x19939B94, 0x1D528623,
0xF12F560E, 0xF5EE4BB9, 0xF8AD6D60, 0xFC6C70D7, 0xE22B20D2, 0xE6EA3D65, 0xEBA91BBC, 0xEF68060B,
0xD727BBB6, 0xD3E6A601, 0xDEA580D8, 0xDA649D6F, 0xC423CD6A, 0xC0E2D0DD, 0xCDA1F604, 0xC960EBB3,
0xBD3E8D7E, 0xB9FF90C9, 0xB4BCB610, 0xB07DABA7, 0xAE3AFBA2, 0xAAFBE615, 0xA7B8C0CC, 0xA379DD7B,
0x9B3660C6, 0x9FF77D71, 0x92B45BA8, 0x9675461F, 0x8832161A, 0x8CF30BAD, 0x81B02D74, 0x857130C3,
0x5D8A9099, 0x594B8D2E, 0x5408ABF7, 0x50C9B640, 0x4E8EE645, 0x4A4FFBF2, 0x470CDD2B, 0x43CDC09C,
0x7B827D21, 0x7F436096, 0x7200464F, 0x76C15BF8, 0x68860BFD, 0x6C47164A, 0x61043093, 0x65C52D24,
0x119B4BE9, 0x155A565E, 0x18197087, 0x1CD86D30, 0x029F3D35, 0x065E2082, 0x0B1D065B, 0x0FDC1BEC,
0x3793A651, 0x3352BBE6, 0x3E119D3F, 0x3AD08088, 0x2497D08D, 0x2056CD3A, 0x2D15EBE3, 0x29D4F654,
0xC5A92679, 0xC1683BCE, 0xCC2B1D17, 0xC8EA00A0, 0xD6AD50A5, 0xD26C4D12, 0xDF2F6BCB, 0xDBEE767C,
0xE3A1CBC1, 0xE760D676, 0xEA23F0AF, 0xEEE2ED18, 0xF0A5BD1D, 0xF464A0AA, 0xF9278673, 0xFDE69BC4,
0x89B8FD09, 0x8D79E0BE, 0x803AC667, 0x84FBDBD0, 0x9ABC8BD5, 0x9E7D9662, 0x933EB0BB, 0x97FFAD0C,
0xAFB010B1, 0xAB710D06, 0xA6322BDF, 0xA2F33668, 0xBCB4666D, 0xB8757BDA, 0xB5365D03, 0xB1F740B4,
};

/// <summary>
/// Calculates the checksum of the input data array.
/// </summary>
/// <param name="data">The input data array.</param>
/// <param name="length">The length of the data array.</param>
/// <returns>Returns the CRC32 value</returns>
/// <remarks></remarks>
public static UInt32 CalculateChecksum(byte[] data, int length)
{
   UInt32 crc = CRC32_INIT_VALUE;  // equals zero
   for (int index = 0; index < length; index++)
   {
      crc = getCRC32_byte(crc, data[index]);
   }
   return crc;
}
```

```
/// <summary>
/// Calculates the CRC32 from the input byte.
/// </summary>
/// <param name="input_crc32_accum">The input_crc32_accum.</param>
/// <param name="data">The data.</param>
/// <returns>Returns the crc value</returns>
/// <remarks></remarks>
public static UInt32 getCRC32_byte(UInt32 input_crc32_accum, byte data)
{
   UInt32 i;
   UInt32 crc32_accum;

   crc32_accum = input_crc32_accum;
   i = ( crc32_accum >> 24 ) ^ ((data));
   crc32_accum = (crc32_accum << 8) ^ crctable[i];

   return crc32_accum;
}
```