Powerline Transceiver Datasheet PLT V 1.50

Copyright © 2009-2013 Cypress Semiconductor Corporation. All Rights Reserved.

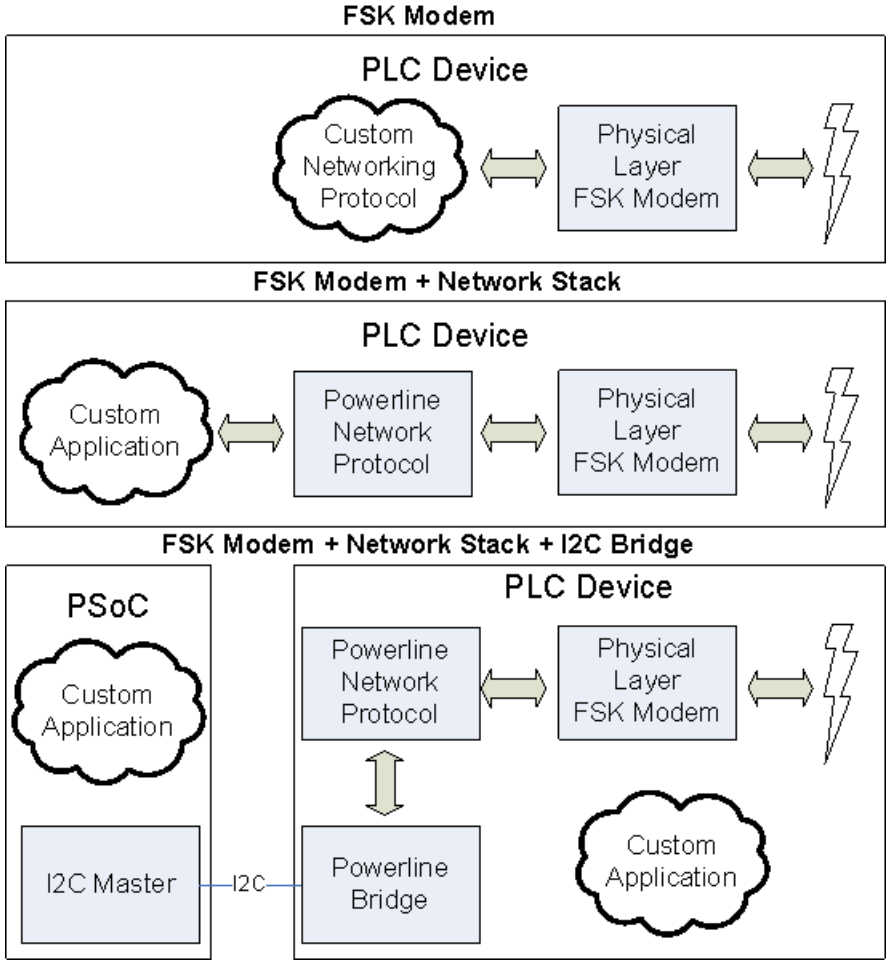
Resources	PSoC® Blocks			API Memory (Bytes)		Pins (per External I/O)
	Digital	Analog CT	Analog SC	Flash	RAM	
CY8CPLC20, CY8CLED16P01, CY8C29x66						
FSK Modem Only	8	3	6	8573	164	11
FSK Modem with Network	8	3	6	19695	323	11
FSK Modem with Network and I^2C Slave	8	3	6	19518	323	13

Features and Overview

- Choice between three implementations:
  - FSK modem only
  - FSK modem with proprietary Powerline Communication Protocol
  - FSK modem with proprietary Powerline Communication Protocol along with built-in communication to external microcontrollers through I^2C
- Bidirectional half-duplex communication
- Supports master and slave and peer-to-peer network topologies
- Multiple masters on powerline network
- Each device has a built in unique 64-bit address
- 8-bit logical addressing supports up to 256 powerline nodes
- 16-bit extended logical addressing supports up to 65536 powerline nodes
- 64-bit physical addressing supports up to 2^{64} powerline nodes
- Individual, broadcast, or group mode addressing
- Carrier Sense Multiple Access (CSMA)

The Powerline Transceiver (PLT) User Module gives a control interface within the PLC device for communicating between the application and the FSK Modem. The application may either run on the PLC device or on a separate device that communicates with the PLC device through I^2C, RS232, or SPI. The I^2C communication implementation is available as a user module option and does not require additional coding to operate.

Figure 1. Block Diagram



Quick Start

In the user module catalog, select Power Line Comm > PLT User Module and the required selection for your application.

For the FSK Modem Only

1. In the Properties Window, select the required Baud rate from 2400, 1800, 1200, and 600 bps.
2. Set the Noise Level Threshold to the required value for your application.
3. Generate the application.
4. Use the FSK Modem Transmitter sample code from the Sample Code section of this datasheet to program the main.c source file.
5. Repeat steps 1 to 4 for a second device, except use the FSK modem receiver sample code. Ensure that the baud rates are the same.

For the FSK Modem with Network

1. Generate the application.
2. Use the FSK Modem with Network Transmitter sample code given in the Sample Code section of this datasheet to program the main.c source file.
3. Repeat steps 1 and 2 for a second device, except use the FSK Modem with Network Receiver sample code.

For the FSK Modem with Network and I^{2}C Slave

1. In the Properties Window, set the I^{2}C slave address. The value must be between 1 and 127.
2. Generate the application.
3. Use the FSK Modem with Network and I^{2}C Slave sample code given in the sample code section of this datasheet to program the main.c source file.
4. Repeat steps 1 to 3 with a second device.

Functional Description

The PLT User Module implements an FSK Modem and optionally a network protocol stack and I^{2}C slave communication capability. The FSK Modem is implemented using the digital and analog resources available on the parts belonging to the PLC family. The optional Powerline Protocol used in this solution is a proprietary Protocol designed by Cypress.

Communication over a Powerline is performed using a defined protocol. The protocol supports a flexible architecture allowing full control over transmission parameters (acknowledged vs. unacknowledged, and repeated transmit), addressing formats, and address types (single node, broadcast, or group). The Protocol gives robust error checking to detect data loss.

Powerline Transmitter

The application running on the PLC device or an external device generates messages to be transmitted. These messages are embedded into a network packet for transfer over the Powerline. The packets can be generated either by the Cypress proprietary network protocol or a custom defined protocol. These packets are modulated by the FSK Modem and coupled to the Powerline by the external coupling circuit.

Powerline Receiver

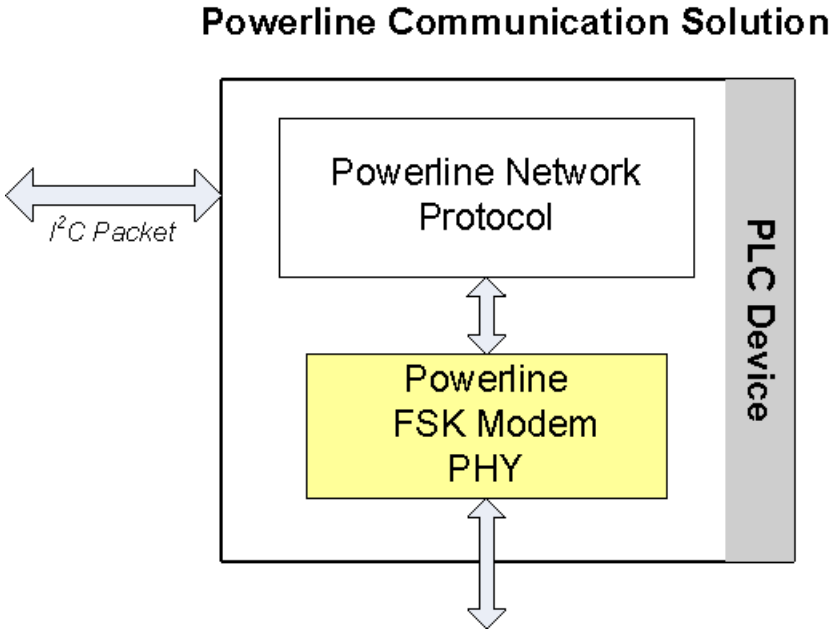
Powerline signals are received by the coupling circuit and demodulated by the FSK Modem PHY. These PLT packets are decoded by the Powerline Network Protocol.

Detailed Description

The PLC device consists of two main functional components:

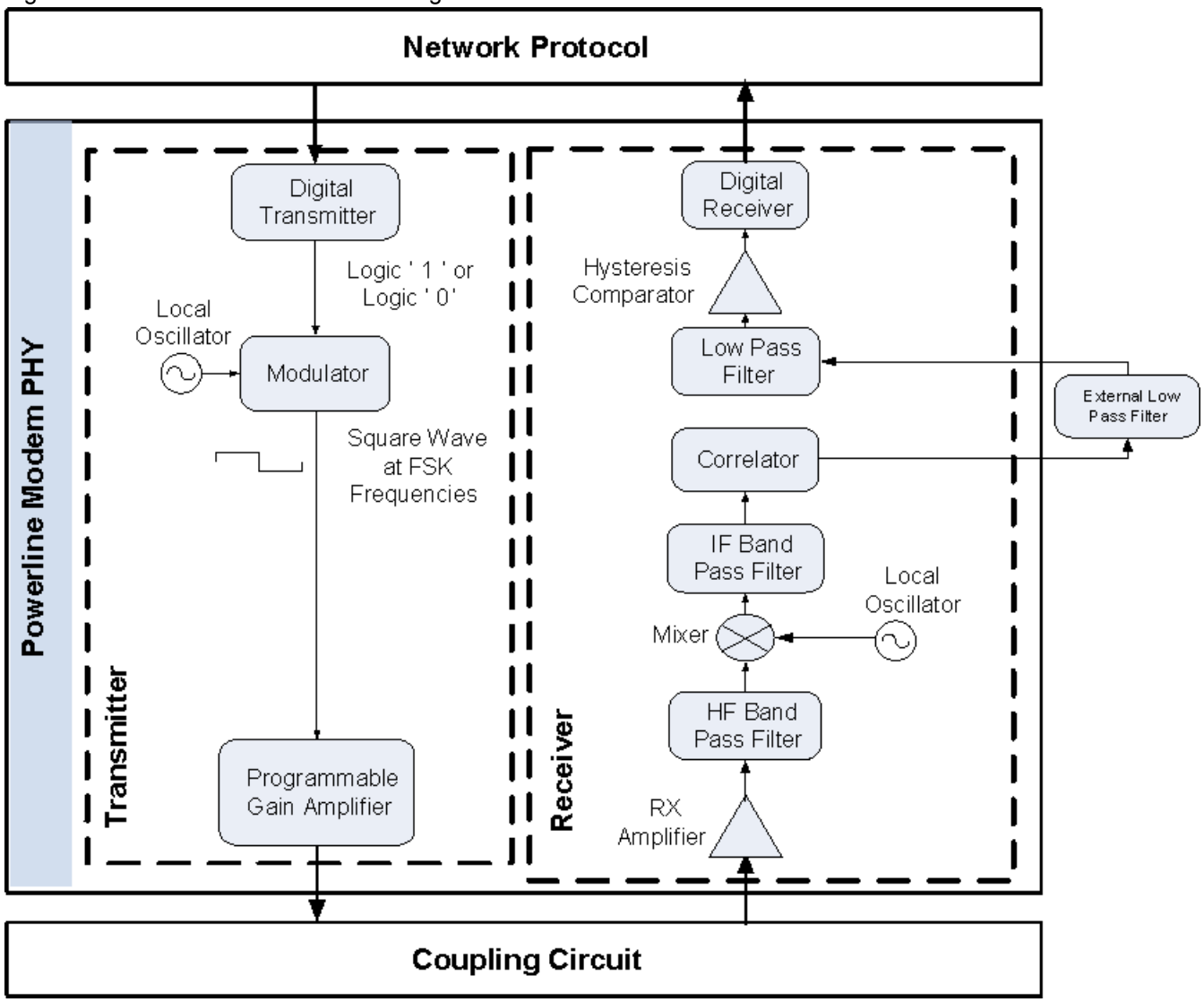
- Powerline Modem PHY
- Powerline Network Protocol

Figure 2. The Physical Layer of a PLC Solution



The physical layer of Cypress's PLC solution is implemented using an FSK modem that enables half duplex communication on a Powerline. This modem supports raw data rates up to 2400 bps.

Figure 3. FSK Modem PHY Block Diagram



Transmitter Section

Digital data from the network layer is serialized by the digital transmitter and fed as input to the modulator. The modulator divides the local oscillator frequency by a definite factor depending on whether the input data is high level logic ‘1’ or low level logic ‘0’. It then generates a sine wave at 133.3 kHz (Logic ‘0’) or 131.8 kHz (Logic ‘1’), which is fed to the Programmable Gain Amplifier to generate FSK modulated signals. The logic ‘1’ frequency can also be configured as 130.4 kHz for wider FSK bandwidth.

Receiver Section

The incoming FSK signal from the Powerline is input to a programmable gain receive amplifier. The amplified signal is fed to the High Frequency (HF) Band Pass Filter that filters out-of-band frequency components and outputs a filtered signal within the desired spectrum of 125 kHz to 140 kHz for further demodulation. The Mixer block multiplies the filtered FSK signals with a locally generated signal to produce heterodyned frequencies.

The Intermediate Frequency (IF) Band Pass Filters further remove out-of-band noise as required for further demodulation. This signal is fed to the correlator which produces a DC component (consisting of Logic ‘1’ and ‘0’) and a higher frequency component.

The output of the correlator is fed to a Low Pass FIlter (LPF) that outputs only the demodulated digital data at the specified baud rate and suppresses all other higher frequency components generated in the correlation process. The output of the LPF is digitized by the hysteresis comparator. This eliminates the effects of correlator delay and false logic triggers due to noise. The Digital Receiver deserializes this data and outputs to the Network Layer for interpretation.

Coupling Circuit Reference Design

The coupling circuit couples low voltage signals from the PLC device to the Powerline. The topology of this circuit is determined by the voltage on the Powerline and design constraints mandated by Powerline usage regulations.

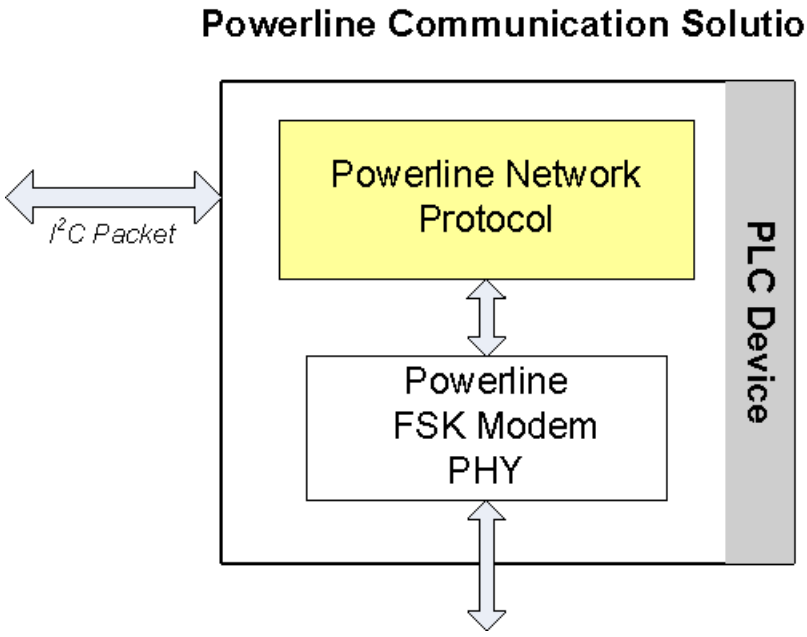
Cypress gives reference designs for a range of Powerline voltages such as 110 V AC, 240 V AC, 12 V DC, 24 V DC, 12 V AC, and 24 V AC. The 110 V AC and 240 V AC designs comply with the following Powerline usage regulations:

- FCC part 15 for North America
- EN50065-1:2001

Powerline Network Protocol

Cypress's Powerline optimized Network Protocol performs the functions of the data link and network layers in an ISO/OSI Equivalent Model.

Figure 4. Powerline Communication Solution



The Network Protocol implemented on the PLC device chip supports the following features:

- Bidirectional half-duplex communication
- Supports master and slave and peer-to-peer network topologies
- Multiple masters on Powerline network
- 8-bit logical addressing supports up to 256 Powerline nodes
- 16-bit extended logical addressing supports up to 65536 Powerline nodes
- 64-bit physical addressing supports up to 2^{64} Powerline nodes
- Individual, broadcast, or group mode addressing
- Carrier Sense Multiple Access (CSMA)
- Full control over transmission parameters:
  - Acknowledged
  - Unacknowledged
  - Repeated transmit

CSMA and Timing Parameters

- CSMA: When band-in-use detection is enabled, the protocol gives the random selection of a period between 85 and 115 ms (out of 7 possible values in this range) in which the band in use detector must indicate that the line is not in use, before attempting a transmission. After completing a transmission when band-in-use is enabled for the system, the application should wait 125ms before the next transmission.
- Band-In-Use (BIU): A Band-In-Use detector, as defined under CENELEC EN 50065-1, is active whenever a signal that exceeds 86 dBuVrms anywhere in the range 131.5 kHz to 133.5 kHz is present for at least 4 ms. This threshold can be configured for different end-system applications not requiring CENELEC compliance.The modem tries to retransmit after every 85 to 115 ms when the Band is in use. The Transmitter times out after 1.1 seconds to 3.5 seconds (depending on the noise on the Powerline) and generates an interrupt to indicate that the transmitter was unable to acquire the Powerline.
- For non-CENELEC compliant systems, the BIU interval can be modified. The BIU Time Interval (min) parameter sets the minimum possible interval period. The BIU Time Interval (span) parameter sets the span from the lowest possible interval to the highest possible interval. For example a BIU Time Interval (min) of 10ms and BIU Time Interval (span) of 15ms will generate a random selection of a period between 10 and 25ms (out of 7 possible values in this range). Note that the BIU Time Interval (span) of 5ms does not generate a random period out of 7 possible values. Also the BIU Timeout can be changed so that the transmitter times out on the first band-in-use detection. Refer to the Timing_Config register for how to set these properties.

Powerline Transceiver Packet

The Powerline Network Protocol defines a Powerline Transceiver (PLT) packet structure, which is used for data transfer between nodes across the Powerline. Packet formation and data transmission across the Powerline network is implemented internally in the PLC device.

A PLT Packet is apportioned into a variable length header (minimum 6 bytes to maximum 20 bytes, depending on address type), a variable length payload (minimum 0 bytes to maximum 31 bytes), and a packet CRC byte.

This packet (preceded by a one byte preamble "0xAB") is then transmitted by the Powerline Modem PHY and the external coupling circuit across the Powerline.

Table 1. Powerline Transceiver (PLT) Packet Structure

Byte Offset	Bit Offset							
	7	6	5	4	3	2	1	0
0x00	SA Type	DA Type		Service Type	RSVD	RSVD	Response	RSVD
0x01	Destination Address (8-bit Logical, 16-bit Extended Logical or 64-bit Physical)							
0x02	Source Address (8-bit Logical, 16-bit Extended Logical or 64-bit Physical)							
0x03	Command							
0x04	RSVD			Payload Length				
0x05	Seq Num				Powerline Packet Header CRC			

Byte Offset	Bit Offset							
	7	6	5	4	3	2	1	0
0x06	Payload (0 to 31 Bytes)							
0x07	Powerline Transceiver Packet CRC							

Packet Header

The Packet Header comprises the first six bytes of the packet when 1-byte logical addressing is used. When 8-byte physical addressing is used, the source and destination addresses each contain eight bytes. In this case, the header can consist of a maximum of 20 bytes. Unused fields marked RSVD are for future expansion and are transmitted as bit 0.

Table 2. Powerline Transceiver (PLT) Packet Header

Field Name	Number of Bits	Tag	Description
SA Type	1	Source Address Type	0 - Logical Addressing 1- Physical Addressing
DA Type	2	Destination Address Type	00 - Logical Addressing 01 - Group Addressing 10 - Physical Addressing 11 - Invalid
Service Type	1		0 - Unacknowledged Messaging 1 - Acknowledged Messaging
Seq Num	4	Sequence Number	Four bit Unique Identifier for each packet between source and destination
Header CRC	4		Four bit CRC Value. This enables the receiver to suspend receiving the rest of the packet if its header is corrupted

Payload

The packet payload has a length of 0 to 31 bytes. Payload content is user defined.

Packet CRC

The last byte of the packet is an 8-Bit CRC value used to check packet data integrity. This CRC calculation includes the header and payload portions of the packet and is in addition to the Powerline Packet Header CRC.

Sequence Numbering

The sequence number is increased for every new unique packet transmitted. If in acknowledged mode and an acknowledgment is not received for a given packet, that packet is retransmitted (if TX_Retry > 0) with the same sequence number. If in unacknowledged mode, the packet is transmitted (TX_Retry + 1) times with the same sequence number.

If the receiver receives consecutive packets from the same source address with the same sequence number and packet CRC, it does not notify the host of the reception of the duplicate packet. If in acknowledged mode, it still sends an acknowledgment so that the transmitter knows that the packet was received.

Addressing

The logical address of the PLC node is set by the local application or by a remote node on the Powerline. Every PLC node also has a unique 64-bit physical address.

Group Membership

Group Membership enables you to multicast messages to select groups. The PLC device supports two types of Group Addressing:

Single Group Membership: The Network protocol supports up to 256 different groups on the network in this mode. In this mode, each PLC node can only be part of a single group. For example, multiple PLC nodes can be part of Group 131.

Multiple Group Membership: The Network protocol supports eight different groups in this mode and each PLC node can be a part of multiple groups. For example, a single PLC node can be a part of Group 3, Group 4, and Group 7 at the same time.

Both these modes can also be used together for Group membership. For example, a single PLC node can be a part of Group 131 and also multiple groups such as Group 3, Group 4, and Group 7.

The Group membership ID for broadcasting messages to all nodes in the network is 0x00.

The Service Type is always set to Unacknowledgment Mode in Group Addressing Mode. This is to avoid Acknowledgment flooding on the Powerline during multicast.

PLC Device Memory Map

The following table gives detailed information on PLC device memory locations. This information can be used for Application development on an external host controller. Several PLC Commands are instantiated from the Powerline Network Protocol based on which memory location is written into. The Memory Map is accessible by all PLT User Module options. However, for the FSK Modem only user module option, only addresses 0x30 through 0x34 and 0x6A through 0x72 are available.

Table 3. PLC Device Memory Map

Offset	Register Name	I²C Access[a]	7	6	5	4	3	2	1	0
0x00	INT_Enable	RW	INT_Clear	INT_Polarity	INT_UnableToTX	INT_TX_NO_ACK	INT_TX_NO_RESP	INT_RX_Packet_Dropped	INT_RX_Data_Available	INT_TX_Data_Sent
0x01	Local_LA_LSB	RW	8-bit Logical Address / LSB for extended 16-bit address							
0x02	Local_LA_MSB	RW	MSB for 16-bit Extended Address							
0x03	Local_Group	RW	8-bit Group Address							
0x04	Local_Group_Hot	RW	One Hot Encoded (for example. if byte = 0b00010001, then member of groups #5 and #1)							
0x05	PLC_Mode	RW	TX_Enable	RX_Enable	Lock_Configuration	Disable_BIU	RX_Override	Set_Ext_Address	Promiscuous_MASK	Promiscuous_CRC_MASK
0x06	TX_Message_Length	RW	Send_Message	Reserved		Payload_Length_MASK				
0x07	TX_Config	RW	TX_SA_Type	TX_DA_Type		TX_Service_Type	TX_Retry			
0x08	TX_DA	RW	Remote Node Destination Address (8 bytes)							
0x10	TX_CommandID	RW	TX Command ID							
0x11	TX_Data	RW	TX Data (31 bytes)							
0x30	Threshold_Noise	RW	Reserved	Auto_BIU_Threshold	Reserved		BIU_Threshold_Constant			
0x31	Modem_Config	RW	Reserved	Modem_TXDelay		Reserved	Modem_FSKBW	Reserved	Modem_BPS	
0x32	TX_Gain	RW	Reserved				TX_Gain_Mask			
0x33	RX_Gain	RW	Reserved					RX_Gain_Mask		
0x34	Timing_Config	RW	BIU_Interval_Min		BIU_Interval_Span		BIU_Timeout	Reserved	ACK_Timeout	
0x35-0x3F	Reserved	RW	Reserved							
0x40	RX_Message_INFO	RW	New_RX_Msg	RX_DA_Type	RX_SA_Type	RX_Msg_Length				

Offset	Register Name	I^2C Access[a]	7	6	5	4	3	2	1	0
0x41	RX_SA	R	Remote Node Source Address(8 Bytes)							
0x49	RX_CommandID	R	RX Command ID							
0x4a	RX_Data	R	RX Data (31 bytes)							
0x69	INT_Status	R	Status_Value_Change	Reserved	Status_UnableToTX	Status_TX_NO_ACK	Status_TX_NO_RESP	Status_RX_Packet_Dropped	Status_RX_Data_Available	Status_TX_Data_Sent
0x6A	Local_PA	R	Physical Address (8 bytes), "0x6A -> MSB"							
0x72	Local_FW	R	Version Number							

a. The Access column is only applicable for the "FSK Modem + Network Stack + I2C Bridge" user module option.

The following table describes the fields specified in the previous table. The default value for each field is '0' unless otherwise noted.

Table 4. Memory Field Description

Field Name	Number of Bits	Description
INT_Enable (Interrupt Config Register 0x00) for the PLT_HostInterrupt_ISR interrupt service routine		
INT_Clear	1	0 - INT Cleared (W) 1 - INT Triggered (Set Internally) Note The user should set this bit to Logic 0 after reading the INT_Status register. This will clear the INT_Status register, except for Status_RX_Packet_Dropped and Status_RX_Data_Available.
INT_Polarity	1	0 - Active High 1 - Active Low
INT_UnableToTX	1	Enable Interrupt for BIU Timeout and the Modem is unable to Transmit, if Disable BIU = 0
INT_TX_NO_ACK	1	Enable Interrupt for no acknowledgment received after if Service Type = 1 (ACK Mode)
INT_TX_NO_RESP	1	Enable Interrupt for No Response Received
INT_RX_Packet_Dropped	1	Enable Interrupt when RX Packet is dropped because RX Buffer is full
INT_RX_Data_Available	1	Enable Interrupt when RX buffer has new data
INT_TX_Data_Sent	1	Enable Interrupt when TX data is sent successfully
PLC_Mode Register (0x05)		
TX_Enable	1	0 - TX Disabled (Can send ACKs only) 1 - TX Enabled

Field Name	Number of Bits	Description
RX_Enable	1	0 - RX Disabled (Can Receive ACKs only) 1 - RX Enabled
Lock_Configuration	1	0 - Allow Remote Access to change config (TX Enable, Ext Address, Disable BIU, Threshold Value, Logical Address, Group Membership) 1 - Lock Remote Access to change config
Disable_BIU	1	0 - Enables Band-In-Use 1 - Disables Band-In-Use
RX_Override	1	0 - If RX Buffer is full, new RX Message is dropped 1 - If RX Buffer is full, new RX Message overwrites RX Buffer
Set_Ext_Address	1	0 - 8-bit Addressing Mode 1 - Extended 16-bit Addressing Mode Note: This mode must be the same in all the devices in the network
Promiscuous_MASK	1	0 - Drops the RX Message if Destination Address does not match the Local Address 1- Ignores Destination Address match and accepts all CRC-verified RX Messages
Promiscuous_CRC_MASK	1	0 - Drops the RX Message if CRC fails 1- Ignores CRC and accepts all RX Messages if Destination Address matches Local Address
TX_Message_Length Register (0x06)		
Send_Message	1	(For the "FSK Modem Only + Network Stack + I2C Bridge" user module option only.) 0 - Transmitter is idle. Automatically cleared after each Transmit 1 - Triggers the transmitter to send a message across the Powerline Note: The registers TX Config, TX Destination Address, TX Command ID and TX Data need to be set before you set this bit to Logic 1
Payload_Length_MASK	5	5-bit value for variable payload length. The payload length can vary from 0 to 31.
TX_Configuration Register (0x07)		
TX_SA_Type TX_SA_Type_Log TX_SA_Type_Phy	1	Mask 0 - Logical Address 1 - Physical Address
TX_DA_Type TX_DA_Type_Log TX_DA_Type_Grp TX_DA_Type_Phy	2	Mask 00 - Logical Address 01 - Group Address 10 - Physical Address
TX_Service_Type	1	0 - Unacknowledgement mode 1 - Acknowledgement Mode

Field Name	Number of Bits	Description
TX_Retry	4	4-bit value for variable TX Retry Count
TX Destination Address Register (0x08 - 0x0F)		
8-bit Logical Address		0x08
16-bit Logical Address		0x08 - LSB 0x09 - MSB
64-bit Physical Address		0x08 - MSB | 0x0F - LSB
Threshold Noise Register (0x30)		
BIU_Threshold_Constant	4	Sets the threshold for the Band In Use Detection. 000 - 70 dBuVrms 001 - 75 dBuVrms 010 - 80 dBuVrms 011 - 87 dBuVrms (default) 100 - 90 dBuVrms 101 - 93 dBuVrms 110 - 96 dBuVrms 111 - 99 dBuVrms 1000 - 103 dBuVrms 1001 - 106 dBuVrms 1010 - 109 dBuVrms 1011 - 112 dBuVrms 1100 - 115 dBuVrms 1101 - 118 dBuVrms
FSK Configuration Register (0x31)		
Modem_TXDelay Modem_TXDelay_7ms Modem_TXDelay_13ms Modem_TXDelay_19ms Modem_TXDelay_25ms	2	Mask 00 - 7ms 01 - 13 ms 10 - 19 ms 11 - 25 ms
Modem_FSKBW Modem_FSKBW_1_5M Modem_FSKBW_3M	1	Mask 0 - Logic '0' - 133.3 kHz Logic '1' - 131.8 kHz 1 - Logic '0' - 133.3 kHz Logic '1' - 130.4 kHz (default)
Modem_BPS Modem_BPS_600[a] Modem_BPS_1200 Modem_BPS_1800 Modem_BPS_2400	2	Mask 00 - 600 bps 01 - 1200 bps 10 - 1800 bps 11 - 2400 bps (default)
Transmit Gain Register (0x32)		

Field Name	Number of Bits	Description
TX_Gain	4	The following are the output AC voltage swing for the given settings: 0000 - 55 mVp-p 0001 - 75 mVp-p 0010 - 100 mVp-p 0011 - 125 mVp-p 0100 - 180 mVp-p 0101 - 250 mVp-p 0110 - 360 mVp-p 0111 - 480 mVp-p 1000 - 660 mVp-p 1001 - 900 mVp-p 1010 - 1.25 Vp-p 1011 - 1.55 Vp-p (default) 1100 - 2.25 Vp-p 1101 - 3.00 Vp-p 1110 - 3.50 Vp-p 1111 - Reserved
RX_Gain Register (0x33)		
RX_Gain	3	The following values are the minimum RX input sensitivity for the given settings: 000 - 5 mVrms (default) 001 - 5 mVrms 010 - 2.5 mVrms 011 - 1.25 mVrms 100 - 600 µVrms 101 - 350 µVrms 110 - 250 µVrms 111 - 125 µVrms
Timing_Config Register (0x34)		
BIU_Interval_Min BIU_Interval_Min_85ms BIU_Interval_Min_50ms BIU_Interval_Min_20ms BIU_Interval_Min_10ms	2	Mask 00 – 85ms 01 – 50ms 10 – 20ms 11 – 10ms
BIU_Interval_Span BIU_Interval_Span_30ms BIU_Interval_Span_15ms BIU_Interval_Span_5ms	2	Mask 00 – 30ms (7 random values) 01 – 15ms (7 random values) 10 – 5ms (1 fixed value. Not recommended to use for more than one device on the network)
BIU_Timeout	1	0 – 1.1s Timeout 1 – Timeout on first BIU detection
ACK_Timeout ACK_Timeout_500ms ACK_Timeout_Auto100ms ACK_Timeout_Auto50ms ACK_Timeout_Auto20ms	2	Mask 00 – 500ms 01 – Auto + 100ms 10 – Auto + 50ms 11 – Auto + 20ms

Field Name	Number of Bits	Description
RX Message Information Register (0x40)		
New_RX_Msg	1	0 - No Packet received 1 - New Packet received Note: The user should set this bit to Logic 0 after reading the RX Message. This allows the device to receive a new RX message
RX_DA_Type	1	0 - Logical / Physical Addressing 1 - Group Addressing
RX_SA_Type	1	0 - Logical Address 1 - Physical Address
RX_Msg_Length	5	5-bit value for variable payload length. The payload length can vary from 0 to 31.
RX Source Address Register (0x41 - 0x48)		
8-bit Logical Address		0x41
16-bit Logical Address		0x41 - LSB 0x42 - MSB
64-bit Physical Address		0x41 - MSB | 0x48 - LSB
Status change Register (0x69) Note: When the user sets INT_Clear to '0', every bit in this register (except Status_RX_Packet_Dropped and Status_RX_Data_Available) are cleared to '0'. When the user sets New_RX_MSG to '0', the Status_Value_Change, Status_RX_Packet_Dropped, and Status_RX_Data_Available bits are cleared to '0'.		
Status_Value_Change	1	0 - No Change 1 - Change
Status_UnableToTX	1	0 - No BIU Timeout 1- BIU Timeout and the Modem is unable to Transmit, if Disable BIU = 0
Status_TX_NO_ACK	1	If Service Type = 1 (ACK Mode) 0 - ACK Received (when TX Data sent = 1) 1 - No ACK received (when TX Data sent = 0) Note: The timeout window for receiving the ACK is set by the ACK_Timeout property in the Timing_Config register.

Field Name	Number of Bits	Description
Status_TX_NO_RESP	1	0 - Response Received (when TX Data sent = 1) 1 - No Response Received (when TX Data sent = 0) Note: The timeout window for receiving Responses is 1.5s
Status_RX_Packet_Dropped	1	If RX Overwrite = 0 0 - No RX Packet is dropped 1- RX Packet is dropped because RX Buffer is full
Status_RX_Data_Available	1	0 - No new data available in RX buffer 1- RX buffer has new data available
Status_TX_Data_Sent	1	0 - No TX data sent 1- TX data sent successfully

a. To ensure that the receiver has sufficient time to start up and read the first byte, the transmit delay parameter (Modem_TXDelay) should be set to >= 18 ms for 600 bps and >= 12 ms for 1200 bps. For 1800 bps and 2400 bps, the delay can be set to any value.

External Host Application

The application residing on the external host microcontroller has direct access to the local PLC memory over any of the available digital communication protocols (I2C, RS232, SPI). This communication enables the host controller to instantiate several PLC functions by reading or writing to the appropriate memory locations in the PLC chip. In this way, the host application can configure the PLC device, read status and configuration information, and transmit data to remote Powerline nodes. The I^2C communication implementation is available as a user module option and requires no additional coding to operate. The slave I^2C address is a configurable parameter and can be set to any value from 1 to 127.

Remote Commands

In addition to sending normal data over the Powerline, the PLT User Module can also send (and request) control information to (and from) another node on the network. The type of remote command to transmit is set by the TX_CommandID register and when received, is stored in the RX_CommandID register.

When a control command (Command ID = 0x01 - 0x08 and 0x0C - 0x0F) is received, the protocol will automatically process the packet (if Lock_Configuration is '0'), respond to the initiator, and notify the host of the successful transmission and reception.

When the send data command (ID 0x09) or request for data command (ID 0x0A) is received, the protocol will reply with an acknowledgment packet (if TX_Service_Type = '1'), and notify the host of the new received data. If the initiator doesn't receive the acknowledgment packet within time defined by the ACK_Timeout property, it will notify the host of the no acknowledgment received condition. The ACK_Timeout property can be a fixed 500ms, or can be based on the expected time to receive the acknowledgment packet plus a buffer. For example, the "Auto + 20ms" property will set the timeout to be 20ms longer than the time it would normally take to receive the acknowledgement. This gives the receiver extra time to perform other functions in between checking for a received message. The calculation is based on the length of the acknowledgment packet, the baud rate, and the Modem_TXDelay.

When a response command (ID 0x0B) is received by the initiator within 1.5s of sending the request for data command, the protocol will notify the host of the successful transmission and reception. If the response command is not received by the initiator within 1.5s, it will notify the host of the no response received condition.

The host is notified by updating the appropriate values in the INT_Status register (including Status_Value_Change) and calling the PLT_HostInterrupt_ISR interrupt service routine (if the corresponding bit is set in the INT_Enable register).

The command IDs 0x30-0xff can be used for custom commands that would be processed by the external host (for example, set an LED color, get a temperature/voltage reading).

The available remote commands are described in the following table with the respective Command IDs.

Table 5. Remote Commands

Command ID	Command Name	Description	Payload (TX Data)	Response (RX Data)
0x01	SetRemote_TXEnable	Sets the TX Enable bit in the PLC Mode Register. Rest of the PLC Mode register is unaffected	0 - Disable Remote TX 1 - Enable Remote TX	If Remote Lock Config = 0, Response = 00 (Success) If Remote Lock Config = 1, Response = 01 (Denied)
0x03	SetRemote_ExtendedAddr	Set the Addressing to Extended Addressing Mode	0 - Disable Extended Addressing 1 - Enable Extended Addressing	If Remote Lock Config = 0, Response = 00 (Success) If Remote Lock Config = 1, Response = 01 (Denied)
0x04	SetRemote_LogicalAddr	Assigns the specified logical address to the remote PLC node	If Ext Address = 0, Payload = 8-bit Logical Address If Ext Address = 1, Payload = 16-bit Logical Address	If Remote Lock Config = 0, Response = 00 (Success) If Remote Lock Config = 1, Response = 01 (Denied)
0x05	GetRemote_LogicalAddr	Get the Logical Address of the remote PLC node	None	If Remote TX Enable = 0, Response = None If Remote TX Enable = 1, {If Ext Address = 0, Response = 8-bit Logical Address If Ext Address = 1, Response = 16-bit Logical Address}
0x06	GetRemote_PhysicalAddr	Get the Physical Address of the remote PLC node	None	If Remote TX Enable = 0, Response = None If Remote TX Enable = 1, Response = 64-bit Physical Address
0x07	GetRemote_State	Request PLC_Mode Register content from a Remote PLC node	None	If Remote TX Enable = 0, Response = None If Remote TX Enable = 1, Response = Remote PLC Mode register

Command ID	Command Name	Description	Payload (TX Data)	Response (RX Data)
0x08	GetRemote_Version	Get the Version Number of the Remote Node	None	If TX Enable = 0, Response = None If TX Enable = 1, Response = Remote Version register
0x09	SendRemote_Data	Transmit data to a Remote Node.	Payload = Local TX Data	If Local Service Type = 0, Response = None If Local Service Type = 1, Response = ACK
0x0A	RequestRemote_Data	Request data from a Remote Node	Payload = Local TX Data	If Local Service Type = 1, Response = ACK. Then, the remote node host must send a ResponseRemote_Data command. The response must be completely transmitted within 1.5s of receiving the request. Otherwise, the requesting node will time out.
0x0B	ResponseRemote_Data	Transmit response data to a Remote Node.	Payload = Local TX Data	None
0x0C	SetRemote_BIU	Enables/Disables BIU functionality at the remote node	0 - Enable Remote BIU 1 - Disable Remote BIU	If Remote Lock Config = 0, Response = 00 (Success) If Remote Lock Config = 1, Response = 01 (Denied)
0x0D	SetRemote_Threshold Value	Sets the Threshold Value at the Remote node	3-bit Remote Threshold Value	If Remote Lock Config = 0, Response = 00 (Success) If Remote Lock Config = 1, Response = 01 (Denied)

Command ID	Command Name	Description	Payload (TX Data)	Response (RX Data)
0x0E	SetRemote_GroupMembership	Sets the Group Membership of the Remote node	Byte0 - Remote SIngle Group Membership Address Byte1- Remote Multiple Group Membership Address	If Remote Lock Config = 0, Response = 00 (Success) If Remote Lock Config = 1, Response = 01 (Denied)
0x0F	GetRemote_GroupMembership	Gets the Group Membership of the Remote node	None	If Remote TX Enable = 0, Response = None If Remote TX Enable = 1, Response = Byte0 - Remote SIngle Group Membership Address Byte1- Remote Multiple Group Membership Address
0x10 - 0x2F	Reserved			
0x30 - 0xFF	User Defined Command Set			

DC and AC Electrical Characteristics

The following values are indicative of expected performance and based on initial characterization data. Unless otherwise specified in the following tables, TA = 25°C, Vdd = 5.0V, Power HIGH, OpAmp Bias HIGH, output referenced to Analog Ground = VCC/2.

Table 6. PLT DC Electrical Characteristics

Parameter	Typical	Limit	Units	Conditions and Notes
FSK_OUT DC Voltage	$V_{DD}/2$	--	V	
FSK_IN DC Voltage	$V_{DD}/2$	--	V	
Operating Current				
TX Mode	30	--	mA	
RX Mode	41	--	mA	

Table 7. PLT AC Electrical Characteristics

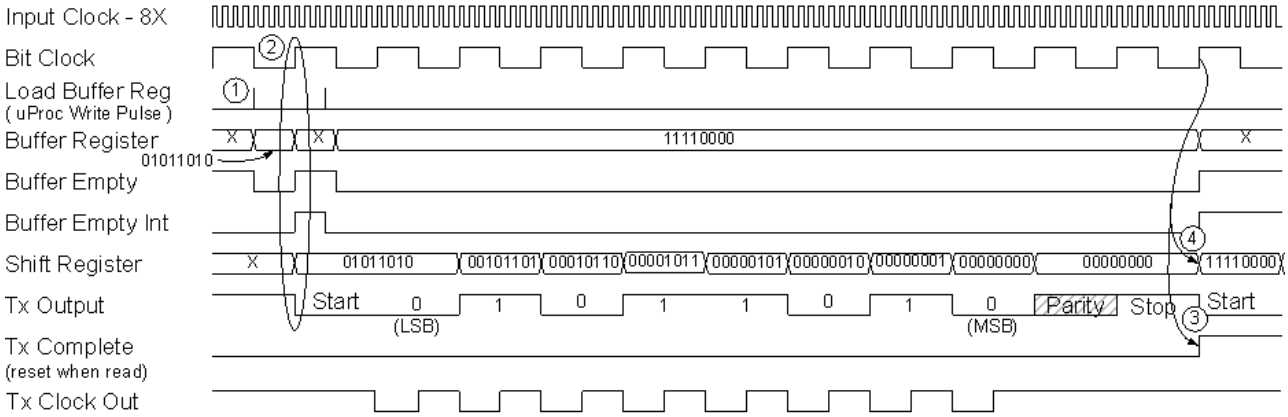
Parameter	Typical	Limit	Units	Conditions and Notes
FSK_OUT Second Harmonic (Fundamental = 125 mVp-p)	-32	--	dB_C	
FSK_OUT Third Harmonic (Fundamental = 125 mVp-p)	-9	--	dB_C	
FSK_OUT Second Harmonic (Fundamental = 1.55 Vp-p)	-34	--	dB_C	
FSK_OUT Third Harmonic (Fundamental = 1.55 Vp-p)	-15	--	dB_C	
Maximum FSK_IN Signal	V_{DD}	--	Vp-p	

Timing

FSK Modem

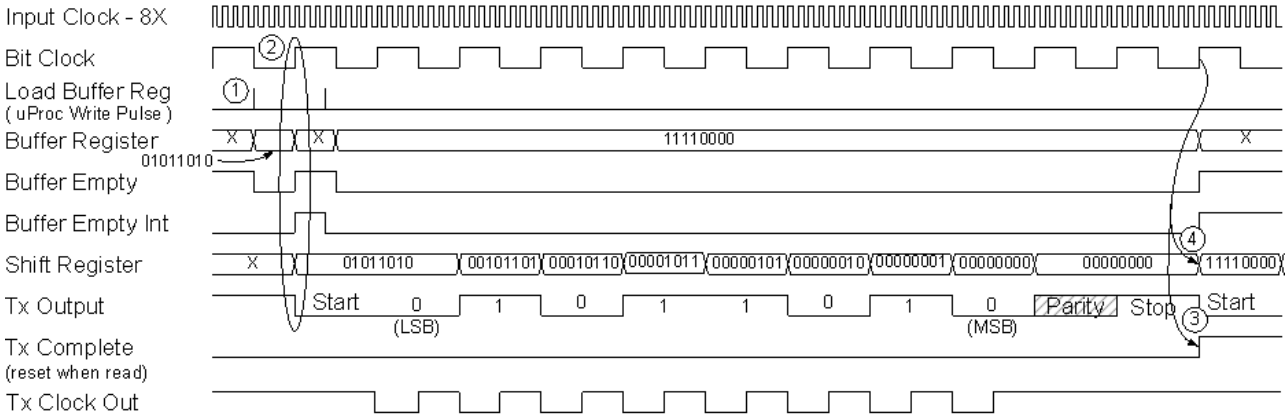
The following RX Timing diagram illustrates the operation of the RX component of the FSK Modem for the PLC User Module.

Figure 5. Figure 5. RX Timing Diagram



The following TX Timing diagram illustrates the operation of the TX component of the FSK Modem for the PLC User Module.

Figure 6. Figure 6. TX Timing Diagram



Placement

The PLT User Module occupies eight digital blocks, three analog CT blocks, six analog SC blocks, and the PLC modem block. Because the user module needs to use specific columns and rows for internal connections, no alternative placement is available.

The following device resources are used by the user module:

Global Resources

- Sleep Timer (512 kHz)
- VC1 (6)
- VC2 (2 when PLT baud rate is 2400 bps, or 1800 bps, 3 when 1200 bps or 600 bps)
- VC3 (104 when PLT baud rate is 2400 bps or 1200 bps, 140 when 1800 bps, 208 when 600 bps)
- 32K_Select (External)
- Analog Power (SC On/Ref High)
- Ref Mux (Can be "(Vdd/2) +/- BandGap" or "(Vdd/2) +/- (Vdd/2)")
- AgndBypass (Enable)
- Opamp Bias (High)
- A_Buff_Power (High)

Pin Use

Table 8. Port Pins Used By the PLT User Module

Port	Use	Description
Port_0_0	Reserved	PLT_Transmitter_Interconnect. Internal transmitter signal
Port_0_3	Analog Output	PLT_FSK_OUT. Transmit FSK output
Port_0_5	Reserved	PLT_Receiver_Interconnect
Port_0_6	Analog Input	PLT_FSK_IN. Receive FSK input
For External Crystal		
Port_1_0	Output	PLT_XTAL_OUT
Port_1_1	Input	PLT_XTAL_IN
Port_1_3	I/O	PLT_XTAL_STABILITY. Connect a 0.1 µF capacitor between the pin and VSS.
FSK Modem with Network and I^2C Slave		
Port_1_5	Open Drain Low	PLT_SDA. I^2C Data
Port_1_7	Open Drain Low	PLT_SCL I^2C Clock

In addition, the following external connections are required for proper function of the user module:

Table 9. External Connections Required By the PLT User Module

Port	Use	Description
Port_2_0		PLT_RXCOMPPOS. Receiver Compensation Output. Should be connected to external f3dB = 7.5 kHz low pass filter circuitry.
Port_2_2		PLT_RXCOMPNEG. Receiver Compensation Input. Input from the low pass filter circuitry.
Port_2_4	External AGND	PLT_AGND. Connect to Analog Ground through a 1.0 µF capacitor.
Port_2_7	Digital Output - Strong Slow	PLT_TX_SHUTDOWN. This pin is set LOW when transmitting. It is set HIGH when not transmitting.

Digital Row Interconnects

Because of the dedicated use of digital blocks 0 and 1, you must not modify the following connections (doing so causes the user module to fail).

- Row_0_Input_0 - Row_0_Input_3
- Row_0_Output_0 - Row_0_Output_3
- Row_1_Input_0 - Row_1_Output_3
- Row_1_Output_0 - Row_1_Output_3
- Row_0_Broadcast
- Row_1_Broadcast

Global Interconnects

The following global interconnects are dedicated to the PLT User Module. Do not modify them.

- GlobalOutEven_0
- GlobalOutEven_1
- GlobalInEven_0 (GlobalInEven_0 is connected to GlobalOutEven_0)

Analog Interconnects

Because of the dedicated use of analog blocks 0, 1, and 3, do not modify these connections:

- AnalogColumn_Clock_0
- AnalogColumn_InputMUX_0
- AnalogOutBus_0
- AnalogOutBuf_0
- AnalogLUT_0
- AnalogColumn_Clock_1
- AnalogColumn_InputMUX_1
- AnalogColumn_InputSelect_1
- AnalogOutBus_1
- AnalogOutBuf_1
- AnalogLUT_1
- AnalogColumn_Clock_3
- AnalogOutBus_3
- AnalogOutBuf_3
- AnalogLUT_3

Dynamic Reconfiguration

If the PLT User Module is used with multiple Loadable Configurations in a project, one of the following steps must be done for the PLT User Module to function properly:

1. Follow the guidelines in Application Note AN2104 - PSoC Dynamic Reconfiguration.
2. Place the PLT User Module in an overlay Loadable Configuration (not the base configuration).

Parameters and Resources

Not all parameters are available with all selection options. The following table shows which parameters are available depending on selection options.

Parameter	FSK Modem Only	FSK Modem with Network	FSK Modem with Network and I^2C Slave
Baud rate	●	●	
Tx Gain	●	●	
Rx Gain	●	●	
Noise Level Threshold	●	●	
FSK Bandwidth	●	●	
Modem_TXDelay	●	●	
BIU Time Interval (min)	●	●	
BIU Time Interval (span)	●	●	
BIU Timeout	●	●	
Band in Use		●	
ACK Timeout		●	
BIU Timeout Interrupt		●	
No ACK Received Interrupt		●	
No Response Received Interrupt		●	
Rx Packet Dropped Interrupt		●	
Rx Data Available Interrupt		●	
Tx Data Sent Interrupt		●	
Addressing mode		●	
Logical Address of Node		●	
Transmit Enable		●	

Parameter	FSK Modem Only	FSK Modem with Network	FSK Modem with Network and I²C Slave
Transmit Source Address Type		●	
Transmit Destination Address Type		●	
Transmit Service Type		●	
Transmit Retry Count		●	
Transmit Payload Length		●	
Receive Enable		●	
Lock Configuration		●	
Rx Overwrite		●	
Destination Address Verification		●	
CRC Messages Verification		●	
Single Group Membership ID		●	
Multiple Group Membership ID		●	
I2C Slave Address			●

Baud rate

Sets the baud rate for the Powerline Communication PHY. You can configure the baud rate to 600, 1200, 1800, or 2400 bits per second. To ensure that the receiver has sufficient time to start up and read the first byte, the transmit delay parameter (Modem_TXDelay) should be set to >= 19 ms for 600 bps and >= 13 ms for 1200 bps. For 1800 bps and 2400 bps, the delay can be set to any value.

TX Gain

Sets the gain of the transmitter's programmable gain amplifier.

RX Gain

Sets the gain of the receiver’s programmable gain amplifier. This in turn determines the minimum input sensitivity for the receiver.

Noise Level Threshold

Sets the Noise Level Threshold for Brand In Use (BIU) detection.

FSK Bandwidth

Sets the separation of the FSK signals representing logic '1' and logic '0'. Can either be set to a deviation of ~1.5 kHz or ~3 kHz. The logic '0' frequency is always 133.3 kHz. The logic '1' frequency can be configured to either 131.8 kHz or 130.4 kHz.

This parameter is sometimes referred to as FSK Deviation.

Modem_TXDelay

Sets the amount of delay from when the transmission is initiated to when the data starts being output from the Transmit Programmable Gain Amplifier. This gives the external circuitry time to set up. The TX_Shutdown signal (Port2[7]) is set before the delay.

BIU Time Interval (min)

Sets the minimum possible interval period that the Band In Use (BIU) detector will wait to see if the line is clear. For CENELEC compliant systems, this should be set to 85ms, which is the default value.

BIU Time Interval (span)

Sets the span of the interval period that the BIU detector will wait to see if the line is clear. Therefore, the maximum interval period is the BIU Time Interval (min) plus the BIU Time Interval (span). When the span is 15 or 30ms, the BIU detector will choose a random value between the minimum and maximum intervals for the detection period. When the span is 5ms, there will only be one fixed value that is used. For CENELEC compliant systems, the span should be set to 30ms, which is the default value.

BIU Timeout

Sets the condition for timing out the BIU detector. It can be either on the first BIU detection of a signal or if the BIU detector can't acquire the line after 1.1 seconds. (This value can be up to 3.5s depending on the amount of noise on the line).

Band in Use

Enables the Band In Use (BIU) option.

ACK Timeout

Sets the amount of time that the device will wait for an acknowledgment after it completes the transmission of a packet. This only applies when the Service Type is acknowledgment mode. The time can be a fixed 500ms, or can be based on the expected time to receive the acknowledgment packet plus a buffer. For example, the "Auto + 20ms" property will set the timeout to be 20ms longer than the time it would normally take to receive the acknowledgement. This gives the receiver extra time to perform other functions in between checking for a received message. The calculation is based on the length of the acknowledgment packet, the baud rate, and the Modem_TXDelay.

BIU Timeout Interrupt

Enables the BIU Timeout interrupt. This interrupt fires when the BIU times out or transmission is attempted when the transmitter is disabled. The interrupt service routine (ISR), PLT_HostInterrupt_ISR, is located in PLT_1INT.asm.

No ACK Received Interrupt

Enables an interrupt that fires when the Service Type is set to ACK Mode and no acknowledgment is received within the time set by the ACK_Timeout property in the Timing_Config register. The interrupt service routine (ISR), PLT_HostInterrupt_ISR, is located in PLT_1INT.asm.

No Response Received Interrupt

Enables an interrupt that fires when no response is received within 1.5s of transmitting (Refer to the Remote Commands section for details). The interrupt service routine (ISR), PLT_HostInterrupt_ISR, is located in PLT_1INT.asm.

Rx Packet Dropped Interrupt

Enables an interrupt that fires when a receive packet is dropped because the Rx buffer is full and the Rx Overwrite property is disabled. The interrupt service routine (ISR), PLT_HostInterrupt_ISR, is located in PLT_1INT.asm.

Rx Data Available Interrupt

Enables an interrupt that fires when the receive buffer has new data. The interrupt service routine (ISR), PLT_HostInterrupt_ISR, is located in PLT_1INT.asm.

Tx Data Sent Interrupt

Enables an interrupt that fires when transmit data is sent successfully. The interrupt service routine (ISR), PLT_HostInterrupt_ISR, is located in PLT_1INT.asm.

Addressing Mode

Sets the logical addressing length to 8-bit or 16-bit.

Logical Address of Node

Set the logical address for the Powerline node. The available addresses vary depending on whether 8-bit or 16-bit addressing is used.

Transmit Enable

Enables Transmit Mode operation. When transmit mode is disabled, no new messages are transmitted. Acknowledgments are transmitted, regardless of Transmit Enable.

Transmit Source Address Type

Sets the Transmit SA Type. Choose Logical Address or Physical Address. You cannot change this parameter if Transmit Enable is set to Disable.

Transmit Destination Address Type

Sets the Transmit DA Type. Choose from Logical Address, Group Address, and Physical Address. This parameter is not editable if the Transmit Enable option is set to Disable.

Transmit Service Type

Sets the Transmit Acknowledgment Mode. Transmissions are acknowledged in acknowledgement mode. This parameter is not editable if the Transmit Enable option is set to Disable.

Transmit Retry Count

Sets the transmitter retry count. In Acknowledgment mode, sets the maximum transmit retries if no acknowledgment is received. In Unacknowledgment mode, sets how many times the transmitter retransmits the same packet. This parameter is not editable if the Transmit Enable option is set to Disable.

Transmit Payload Length

Sets the Payload Length from 0 to 31 bytes. This parameter is not editable if the Transmit Enable option is set to Disable.

Receive Enable

Enables Receive Mode. When receive mode is disabled, no new messages are accepted.

Lock Configuration

Allows remote access to change the configuration.

Rx Overwrite

Enables receive buffer overwrite mode. When enabled, newly received messages overwrite the Rx Buffer when it contains an unread packet. When disabled and the Rx Buffer contains an unread packet, the newly received message is dropped. To clear the RX buffer, the New_RX_Msg must be set to '0'.

Destination Address Verification

Allows the receiver to ignore destination address verification. When the setting is Do Not Ignore the received packet is dropped if it does not match the local address. When set to Ignore, the received packet is accepted regardless of the local address.

CRC Messages Verification

Allows the receiver to ignore cyclic redundancy check message verification. When the setting is Do Not Ignore the packet is dropped if the 8-bit packet CRC fails.

Single Group Membership ID

Sets the group number when the network is placed in 'single group membership' mode.

Multiple Group Membership ID

Sets the group number when network is placed in 'multiple group membership' mode. This parameter is an 8-bit value and is expressed in binary. Each instance of 1 in the number indicates that the group is member of the group corresponding to the binary position of the number. For example, if the value is 01000001, then the particular node is part of the 1st and 7th group.

I2C Slave Address

Can be any value from 0x01 to 0x7F.

Application Programming Interface

The Application Programming Interface (API) routines are given as part of the user module to allow the designer to deal with the module at a higher level. This section specifies the interface to each function together with related constants given by the include files.

The PLT User Module uses precompiled libraries. The default name of the user module is PLT_1 and must not be changed. The precompiled names of the APIs are exactly as shown in the datasheet.

Note

** In this, as in all user module APIs, the values of the A and X register may be altered by calling an API function. It is the responsibility of the calling function to preserve the values of A and X before the call if those values are required after the call. This "registers are volatile" policy was selected for efficiency reasons and has been in force since version 1.0 of PSoC Designer. The C compiler automatically takes care of this requirement. Assembly language programmers must ensure their code observes the policy, too. Though some user module API functions may leave A and X unchanged, there is no guarantee they may do so in the future.

For Large Memory Model devices, it is also the caller's responsibility to preserve any value in the CUR_PP, IDX_PP, MVR_PP, and MVW_PP registers. Even though some of these registers may not be modified now, there is no guarantee that will remain the case in future releases.

PLT_Start

Description:

Before the Powerline Transceiver User Module can be used, it must first be initialized by calling this function.

The functions executed by this API depend on whether you chose Modem Only, Modem with Network, or Modem with Network and I^2C Slave when you placed the user module. The FSK Modem is initialized in all cases. The Networking Protocol Layer is not initialized when the FSK Modem option is chosen. The I^2C Bridge Protocol Layer and associated hardware is initialized only when the FSK Modem + Protocol Stack + I2C Bridge option is selected.

C Prototype:

```
void PLT_Start(void);
```

Assembly:

```
lcall PLT_Start
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_Stop

Description:

This function disables all digital and analog blocks that were initialized to create the FSK modem.

C Prototype:

```
void PLT_Stop(void);
```

Assembly:

```
lcall PLT_Stop
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_Restart

Description:

This function starts the PLT blocks, but retains the current values in the PLT_Memory_Array. It should only be used after PLT_Start has been called at least once.

C Prototype:

```
void PLT_Restart(void);
```

Assembly:

```
lcall PLT_Restart
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_DisableInt

Description:

This function disables all interrupts that are used by PLT.

C Prototype:

```
void PLT_DisableInt(void);
```

Assembly:

```
lcall PLT_DisableInt
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_EnableInt

Description:

This function enables the interrupts that are used by PLT.

C Prototype:

```
void PLT_EnableInt(void);
```

Assembly:

```
lcall PLT_EnableInt
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT FSK Modem Only API Commands

PLT_SwitchToReceiver

Description:

Configures the PSoC device to be an FSK receiver. Automatically stops the FSK transmitter if it is already loaded.

C Prototype:

```
void PLT_SwitchToReceiver(void);
```

Assembly:

```
lcall PLT_SwitchToReceiver
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_SwitchToTransmitter

Description:

Configures the PSoC device to be an FSK transmitter. Automatically stops the FSK receiver if it is already loaded.

C Prototype:

```
void PLT_SwitchToTransmitter (void);
```

Assembly:

```
lcall PLT_SwitchToTransmitter
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_SendData

Description:

The function gives control to the FSK transmitter (if not already started) and sends the data over the powerline.

C Prototype:

```
BOOL PLT_SendData (BYTE *bPacketBuffer, BYTE bPktLen);
```

Assembly:

```
mov A, bPktLen
push A
mov A, >_bPacketBuffer
push A
mov A, <_bPacketBuffer
push A
lcall PLT_SendData
```

Parameters:

bPacketBuffer: A pointer to the first byte of the data to be transmitted.

bPktLen: The length of the data to be transmitted in bytes.

Return Value:

Returns a one if the transmit request was processed. Returns a zero if the transmit request was not processed.

Side Effects:

See Note ** at the beginning of the API section.

PLT_AcquirePowerline

Description:

Checks whether the powerline is in use. Returns a one if the powerline is clear for transmission. A zero return indicates the line is in use.

C Prototype:

```
BOOL PLT_AcquirePowerline(void);
```

Assembly:

```
lcall PLT_AcquirePowerline
mov [bIsAcquired], A
```

Parameters:

None

Return Value:

Returns a one, passed in the accumulator, if the powerline is clear for transmission. A zero return indicates the line is in use.

Side Effects:

See Note ** at the beginning of the API section.

PLT_RXEnableInt

Description:

Enables the receiver interrupts by setting the appropriate interrupt enable bits.

C Prototype:

```
void PLT_RXEnableInt(void);
```

Assembly:

```
lcall PLT_RXEnableInt
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_RXDisableInt

Description:

Disables the receiver interrupts by clearing the appropriate interrupt enable bits.

C Prototype:

```
void PLT_DisableInt(void);
```

Assembly:

```
lcall PLT_DisableInt
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_bReadRxStatus

Description:

Returns the contents of the receiver control register.

C Prototype:

```
BYTE PLT_bReadRxStatus(void);
```

Assembly:

```
lcall PLT_bReadRxStatus
mov [bRdStatus], A
```

Parameters:

None

Return Value:

Returns the status byte read, passed in the accumulator. Use the following defined masks, to test for specific status conditions. Note that masks can be OR'ed together to check for combined conditions.

RX Status Masks	Value
PLT_RX_ACTIVE	0x10
PLT_RX_COMPLETE	0x08
PLT_RX_PARITY_ERROR	0x80
PLT_RX_OVERRUN_ERROR	0x40
PLT_RX_FRAMING_ERROR	0x20
PLT_RX_NO_ERROR	0xE0

Side Effects:

See Note ** at the beginning of the API section.

PLT_bReadRxData

Description:

Reads the received data byte from the Rx Buffer Register.

C Prototype:

```
BYTE bReadRxData (void);
```

Assembly:

```
lcall bReadRxData
mov [bRxData], A
```

Parameters:

None

Return Value:

Data byte received and passed in the Accumulator.

Side Effects:

See Note ** at the beginning of the API section.

PLT_ AutoSetBIUThreshold

Description:

Sets the Band In Use Noise Threshold (BIU_Threshold_Constant) to a value higher than the noise on the Powerline.

C Prototype:

```
void PLT_AutoSetBIUThreshold(void);
```

Assembly:

```
lcall PLT_AutoSetBIUThreshold
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

FSK Modem with Network API Commands

PLT_SendMsg

Description:

This function informs the network protocol that a message is ready to be transmitted over the Powerline. The PLT_Poll function must still be called to transmit the message when the line is available.

C Prototype:

```
BOOL PLT_SendMsg(void);
```

Assembly:

```
lcall PLT_SendMsg
mov [bSendStatus], A
```

Parameters:

None

Return Value:

Returns a one, passed in the accumulator, if the function was successful, a zero otherwise.

Side Effects:

See Note ** at the beginning of the API section.

PLT_Poll

Description:

This function runs through a predefined state machine for data packet transmission and reception. It reacts to settings in the memory map array. When attempting to transmit a packet, this function should be called until the status for the transmitter is updated in the INT_Status byte.

When in idle mode and waiting for a received message, this function should be called often enough so that a new message can be detected by the network protocol before another message is received.

C Prototype:

```
void PLT_Poll(void);
```

Assembly:

```
lcall _PLT_Poll
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_ AutoSetBIUThreshold

Description:

Sets the Band In Use Noise Threshold (BIU_Threshold_Constant) to a value higher than the noise on the Powerline.

C Prototype:

```
void PLT_AutoSetBIUThreshold(void);
```

Assembly:

```
lcall PLT_AutoSetBIUThreshold
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

FSK Modem with Network and I2C Slave APIs

PLT_Poll

Description:

This function runs through a predefined state machine for data packet transmission and reception. It reacts to settings in the memory map array. When attempting to transmit a packet, this function should be called until the status for the transmitter is updated in the INT_Status byte.

When in idle mode and waiting for a received message, this function should be called often enough so that a new message can be detected by the network protocol before another message is received.

C Prototype:

```
void PLT_Poll(void);
```

Assembly:

```
lcall _PLT_Poll
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

PLT_Check_I2C_Activity

Description:

This function allows the external host application to interface with the PLC device through I^2C communication. You can change the protocol parameters in the memory map array through I^2C read/writes. This function checks if the memory map has been updated and implements the changes (for example, clearing the interrupt, clearing the RX buffer). This function should be called often enough to meet the needs of the external host.

C Prototype:

```
void PLT_Check_I2C_Activity(void);
```

Assembly:

```
lcall PLT_Check_I2C_Activity
```

Parameters:

None

Return Value:

None

Side Effects:

See Note ** at the beginning of the API section.

Interrupt Service Routines (ISR)

This section describes the available Interrupt Service Routines (ISR). Any ISRs that are not listed in this section should not be modified.

Software ISRs Available in all User Module Options

Note These software ISRs are not true ISRs as they do not interrupt program execution. They are called by the PLT User Module code.

PLT_BIU_Active_ISR

Description:

Called when a Band-In-Use detect condition occurs.

PLT_BIU_Complete_ISR

Description:

After a Band-In-Use detect condition occurs, this is called when a new attempt is made to acquire the powerline.

PLT_TX_Active_ISR

Description:

Called when the transmitter initiates transmission on the powerline.

PLT_TX_Complete_ISR

Description:

Called when the transmitter completes transmission on the powerline.

Software ISRs Available in the FSK Modem with Network Stack and FSK Modem with Network Stack and I2C Slave User Module Options

Note These software ISRs are not true ISRs as they do not interrupt program execution. They are called by the PLT User Module code.

PLT_HostInterrupt_ISR

Description:

This ISR is called when at least one of six possible conditions occurs. These conditions are described in the INT_Enable register of the PLT Device Memory Map and in the "Interrupt" properties located in the Parameters and Resources Section. Each condition can be enabled or disabled individually.

PLT_RX_Active_ISR

Description:

Called when the receiver receives a new byte of data from the powerline.

PLT_RX_Complete_ISR

Description:

Called when the receiver has stored the new byte of data from the powerline.

Hardware ISR Available in the FSK Modem only User Module Option

PLT_DIG2_ISR

Description:

Called when a new byte is received from the powerline. This ISR is enabled by calling PLT_RXEnableInt and disabled by calling PLT_RXDisableInt.

Sample Firmware Source Code

Note The HI-TECH compiler does not support assembly language only projects. The HI-TECH compiler supports only C projects. C and assembly language files are both supported in C projects. The ImageCraft compiler supports assembly language projects.

Modem

The following sample code configures a PLC transmitter:

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>            // part specific constants and macros
#include "PSoCAPI.h"        // PSoC API definitions for all user modules

void main(void)
{
    // This code mimics a packet of payload two bytes 0xAA and 0xBB sent to
    // a destination address of 0x01 using the same packet structure as the
    // network protocol.
   BYTE aTx_Data[10] = {0xab,0x10,0x01,0x02,0x09,0x02,0x01,0xaa,0xbb,0x34};
    PLT_Start();
    //Check whether the powerline is free
    if (PLT_AcquirePowerline())
    {
        PLT_SendData(aTx_Data,0x0a);    //Transmit the packet
    }
}
```

The following sample code configures a PLC receiver:

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>            // part specific constants and macros
#include "PSoCAPI.h"        // PSoC API definitions for all user modules

void main(void)
{
// This code waits for a packet from a fixed function device and checks
    // whether the first byte is 0xAB and proceeds to read further bytes.
    // This adheres to the same structure as a packet sent from the network
    // layer.
    BYTE bStatus,bRx_Data;
    PLT_Start();
    PLT_RXEnableInt();
    do                    //This loop prevents false triggers
    {
do                //Loop until data is received by the RX Block
        {
        // Check whether the receive register is full
            bStatus = PLT_bReadRxStatus();
            bStatus &= PLT_RX_COMPLETE;        //Break if so..
        }while (bStatus == 0x00);
bRx_Data = PLT_bReadRxData();    //Read received data
    } while(bRx_Data!=0xab);         //Exit the loop only if the byte is 0xAB
    // Loop until the next byte is received. Use the same loop structure for
    // reading the rest of the packet.
    do
{
        bStatus = PLT_bReadRxStatus(); //Get the status of the receiver
        bStatus &= PLT_RX_COMPLETE;       //Check receive register full
    }while (bStatus == 0x00);            //Break if so..
    // Read the next byte from the packet.
    bRx_Data = PLT_bReadRxData();
}
```

FSK Modem with Network

The following sample code configures a PLC transmitter with the network protocol:

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h> // part specific constants and macros
#include "PSoCAPI.h" // PSoC API definitions for all user modules

void main(void)
{
   BYTE bData = 0x00; // Variable used to determine the packet data
   PLT_Start(); // Start the Powerline Transceiver
   PLT_Memory_Array[Local_LA_LSB] = 0x02; // Set local node parameters
   PLT_Memory_Array[PLC_Mode] |= (TX_Enable | RX_Enable );
   PLT_Memory_Array[TX_Config] |= TX_Service_Type; // Set destination node parameters
   PLT_Memory_Array[TX_DA] = 0x01;
   PLT_Memory_Array[TX_CommandID] = CMD_SENDMSG;
   PLT_Memory_Array[TX_Data] = bData;
   PLT_Memory_Array[TX_Message_Length] = 0x01;
   do //Continously send packets to address 0x01 incrementing the data in the payload
by one
   {
     PLT_SendMsg(); // Prepare the message to be transmitted
     do
     {
       PLT_Poll(); // Run the network protocol to send the message
       if (PLT_Memory_Array[INT_Status] & Status_Value_Change) // Check if the status
was updated
       {
         if (PLT_Memory_Array[INT_Status] & Status_UnableToTX)
         {
                                                                  // Add code here to
process why the transmitter is currently busy
         }
         if (PLT_Memory_Array[INT_Status] & Status_TX_NO_ACK)
         {
                                                       // Add code here to process
why there was no ACK received
         }
         if (PLT_Memory_Array[INT_Status] & Status_TX_NO_RESP)
         {
                                                     // Add code here to process
why there was no response received
         }
         if (PLT_Memory_Array[INT_Status] & Status_TX_Data_Sent)
         {
                                                    // Add code here to process
what to do after a packet was sent successfully
         }
         PLT_Memory_Array[INT_Enable] &= ~INT_Clear; // Clear the INT_Status register
         break;
       }
```

```
     } while (1);
     bData++; //Change the value of the first byte of the packet
     PLT_Memory_Array[TX_Data] = bData;
   } while (1); // Send a new packet
}
```

The following sample code configures a PLC receiver with the network protocol:

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h> // part specific constants and macros
#include "PSoCAPI.h" // PSoC API definitions for all user modules

void main(void)
{
   PLT_Start(); // Start the Powerline Transceiver
   PLT_Memory_Array[Local_LA_LSB] = 0x01; // Set the local node parameters
   PLT_Memory_Array[PLC_Mode] |= (TX_Enable | RX_Enable | RX_Override);
   do
   {
     PLT_Poll(); // Run the network protocol
     if (PLT_Memory_Array[INT_Status] & Status_Value_Change) // Only enter if there is
a change in the status register
     {
       if (PLT_Memory_Array[INT_Status] & Status_RX_Packet_Dropped)
       {
                                              // Add code here to process why a packet
was dropped
       }
       if (PLT_Memory_Array[INT_Status] & Status_RX_Data_Available)
        {
                                           // Add code here to process what to do after
a packet was received successfully
       }
      PLT_Memory_Array[RX_Message_INFO] &= ~New_RX_Msg; // Clear the received bits in
the status registers
       PLT_Memory_Array[INT_Enable] &= ~INT_Clear; // Clear the status changed bit in
the status register
     }
   }while(1);
}
```

FSK Modem with Network and I²C Slave

The following sample code can be run with an external host application. This same sample code applies for both a transmitter and a receiver.

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all user modules


void main(void)
{
    //Initialize the PLT module
        PLT_Start();
    do
    {
        //Act on any I2C activity
PLT_Check_I2C_Activity();
        PLT_Poll();
    }while(1);
}
```

Version History

Version	Originator	Description
1.1	FRE	1. Added BIU Interval (Min), BIU Interval (Max), BIU Timeout, and ACK Timeout properties. 2. Added PLT_Restart, PLT_DisableInt, and PLT_EnableInt APIs. 3. Added six noise threshold values for BIU detection. 4. This user module now works with Shadow Registers for compatibility with other user modules.
1.20	FRE	Added wizard help button and file.
1.30	FRE	Transferred the PLTID from the "AREA text" to the "AREA lit" to fix code compression issues.
1.40	DHA	1. Updated area declarations to support Imagecraft optimization. 2. Fixed support for watchdog timer. 3. Reduced the variability of the BIU timeout and BIU detection thresholds. 4. Added support to allow the system to receive PLC messages while acquiring the powerline.
1.50	DHA	1. Added CY8C29x66 device support.

Note PSoC Designer 5.1 introduces a Version History in all user module datasheets. This section documents high level descriptions of the differences between the current and previous user module versions.

Copyright © 2009-2013 Cypress Semiconductor Corporation. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC Designer™ and Programmable System-on-Chip™ are trademarks and PSoC® is a registered trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.