OpenBMIP Standard

Version 11

Rev F

October 27, 2023

ST Engineering iDirect is a global leader in satellite communications (satcom) providing technology and solutions that enable its customers to expand their business, differentiate their services and optimize their satcom networks. Through the merger with Newtec, a recognized industry pioneer, the combined business unites over 35 years of innovation focused on solving satellite's most critical economic and technology challenges, and expands a shared commitment to shaping the future of how the world connects. The product portfolio, branded under the names iDirect and Newtec, represents the highest standards in performance, efficiency and reliability, making it possible for its customers to deliver the best satcom connectivity experience anywhere in the world. ST Engineering iDirect is the world's largest TDMA enterprise VSAT manufacturer and is the leader in key industries including broadcast, mobility and military/government.

Company Web site: http://www.idirect.net | Main Phone: 703.648.8000

TAC Contact Information: Phone: 703.648.8151 | Email: tac@idirect.net | Web site: http://support.idirect.net

iDirect Government created in 2007, is a wholly owned subsidiary of ST Engineering iDirect and was formed to better serve the U.S. government and defense communities. In 2019, iDirect Government acquired Glowlink Communications Technology, Inc.

Company Web site: http://www.idirectgov.com | Main Phone: 703.648.8118

TAC Contact Information: Phone: 703.648.8111 | Email: tac@idirectgov.com | Web site: http://-partnerportal.idirectgov.com/

Document Name: OpenBMIP_Standard_T0000683_RevF_102723.pdf

Document Number: T0000683

Revision History

The following table shows all revisions for this document. To determine if this is the latest revision, check the Technical Assistance Center (TAC) Web site. Refer to Getting Help on page xvii for TAC access information.

Table i-1. Revision History

Revision	Date	Updates
1.0-6.0	2011-2012	Internal versions.
7.0	Dec 14, 2012	First public release.
7.1	Feb 4, 2013	Minor edits: remove “optional” nature of some messages.
7.2	Feb 13, 2013	Incorporate internal and customer reviews:Major cleanup and consistency checkDelete redundant and incorrect message charts; add Table 1 as a clearer format description; add worked example.
7.3	Feb 19, 2013	Additional error codes. Additional command codes to get/set timeouts. Added a table of commands in numerical order. Clarified Band Select Logic. Cleaned up Acronyms & Glossary.
7.4	Feb 22, 2013	Clarified terminator character in string data Removed spaces from XML format
7.5	Feb 25, 2013	Internal only release.
7.6	March 12, 2013	Revised error codes. Corrected 2.2 timeout error code description. Deleted get/set timeout commands. Clarified CMD_SEQ behavior.

Table i-1. Revision History (Cont.)

7.7	March 26, 2013	Revised error codes Corrected 2.2 timeout error code description. Deleted get/set timeout commands. Clarified CMD_SEQ behavior. Revised Get Band Selection response strings.
7.8	May 19, 2013	Added GPOWERAMP and SPOWERAMP commands. Clarified file transfer timeout behavior.
7.9	July 18, 2013	Corrected codes for GPOWERAMP and SPOWERAMP commands.
7.91	Aug 27, 2013	Corrected header in Table 4 Corrected ID items in Section 3.2: BUC Part Number (BPN) , BUC Manufacturer ID (MID) , BUC Serial Number (BSN) , BUC Functional ID (FID) Added descriptions of the XML fields
7.92	Sep 04, 2013	Added Time outs: BUC vendor to specify timeout if 5-seconds impractical Added 5 RSP_CODE: Change “0x08: File receive fail” to “0x08:File transfer fail” Added “Unimplemented” RSP_CODE.3. 2.2.6 CMD_VERS Header Deconstruction: Change “RESP_CODE” to “RSP_CODE”
8	Sep 24, 2013	Change factory calibration procedure and XML file format to add detector reading with power amp disabled.
8.1	Sep 25, 2013	Corrected missing references in 2.7.
8.2	Oct 22, 2013	Deleted reference to OpenAMIP in GPOWERAMP command description.
8.3	Nov 8, 2013	Added option to compress file using zlib/gzip. Added note to select peaks & troughs of gain vs frequency curve.
8.4	Nov 11, 2013	Rephrased note to select peaks & troughs of gain vs frequency curve, and endpoints.
8.5	Nov 11, 2013	Added clarification to ADC slope.
8.6	Nov 13, 2013	Added clarification to Kermit description.

Table i-1. Revision History (Cont.)

8.7	Jan 27, 2013	Added requirement to support both SW upgrade and downgrade to 2.7.15.1.3.1 Software Upgrade Procedure. Added requirement to validate the SW package.
8.8	Apr 04, 2014	Deleted TFTP from Ethernet implementation.
8.9	Apr 11, 2014	Section 3.2 Table 5: Deprecated freq_step_size; steps are permitted to be non-uniform. Section 3.2 Table 5: Clarified detector monotonicity.
9.0	Apr 28, 2014	Section 2.6.2: Describe Ethernet operation: Assign port numbers for OpenBMIP (UDP 6001) and Kermit (TCP 1649) Section 2.7.1: Clarify that the serial number (BSN) needs to be unique, but does not need to follow the example format. Section 3.1: Raw ADC hex values are recommended to be linear to BUC power expressed in dB. Section 3.1: Add a comment on use of “extrapolation” if BUC temperature reading is outside of the BUC Cal file temperatures.
9.1	Apr 29, 2014	Section 2.2.5: Removed incorrect code for “invalid FCS”.
9.2	May 29, 2014	Section 2.7.15.2.5: Corrected file transfer diagram (“STARTFILETRANSFER” -> “SENDFILE”)
9.3	Feb 19, 2015	Section 3.2 Table 5: Added “buc_cutoff_frequency_mhz” parameter; increased <num_temps> maximum to 5.
9.4	April 1, 2015	Section 3.2 Table 5: Changed description for <buc_cutoff_frequency_mhz> for consistency and correctness against actual implementation.
A (9.5)	May 2, 2017	First release of the OpenBMIP Standard document in the iDirect Technical Publications template. Minor changes that do not affect the technical content.
B (9.6)	Aug 18, 2017	Added information about the BUC Test application and source code, and BUC Simulator and source code, to Execute Transfer (Kermit). Added support for BUC Simulator in section Update Simulator. Minor changes which do not affect the technical content.

Table i-1. Revision History (Cont.)

C (10.1)	**Aug 21, 2020**	**Calibration file format version modified; name is now <cal_ format_version>.** **Added Get Polarization and Set Polarization commands (GPOLARIZATION, SPOLARIZATION)**
D (10.2)	**Feb 19, 2021**	**New extension to BUC calibration file format to accommodate per polarization parameters.**
E (11)	**April 2, 2021**	**Added: Request IFL Calibration Permission (REQIFLCAL), IF Calibration Status (IFLCALSTATUS), and Appendix A, B, and C. Extended Status (GEXTSTATUS), multiple Transceiver Commands (Get RX LO, GRXLOSCILLATOR, through Get Power Amp Input Power, GPAINPUTPOWER).**
F (11)	**Oct 27, 2023**	**Added vendors to Appendix A: BUC Manufacturer IDs.** **Update does not affect the technical content.**

Contents

List of Figures

List of Tables

About

Purpose

This document describes the OpenBMIP administrative interface between the modem and block up converter (BUC) components of a satellite terminal. This interface is to enable calibration of the terminal's transmitted power for regulatory compliance. This enables a terminal integrator to provide a seamless installation process, with respect to transmitter power calibration. Additionally, OpenBMIP can provide device control during operations.

This standard may be tailored to the application by selection of interface type (RS-422 serial or Ethernet). It is also designed to be extensible for vendor-specific enhancements.

This interface is intended for setup, calibration, installation purposes and during operations. It is not intended for continuous operation while the BUC is transmitting live data. The messages described here are typically exchanged:

- When a BUC is first installed into a terminal
- When the terminal is first powered on
- Before the terminal begins transmission to connect to a satellite network
- On a periodic basis (typically 1-2 seconds between requests) during regular operation to verify proper BUC functionality
- As required configuration of LO and Polarization during regular operation

Audience

The intended audience for this document is an engineering team responsible for integrating a satellite modem with a Block Up Converter (BUC), or Transceiver, or a team designing a compliant device.

Contents

This document contains the following major sections:

- Introduction

  This chapter gives an introduction about OpenBMIP.

- Protocol Specification

  This chapter describes the protocol specifications, message types, and syntax.

- Calibration

  This chapter explains how the BUC is calibrated.

Standards Reference

- *IETF RFC 1055 Serial Line IP*
- *IETF RFC 1171 Point-to-Point Protocol*
- *IETF RFC 1700 Assigned Numbers*
- *IETF RFC 1952 GZIP file format specification version 4.3*
- *Kermit*
- *TFTP*

Disclaimer

Open BUC Modem Interface Protocol (OpenBMIP) was invented by iDirect.

Permission to copy and distribute this document in unmodified form is hereby granted to all without restriction. Modified forms of this document may be distributed, but only if this "legal matters" section is retained intact and provided that any document that describes a modified form of the protocol clearly states that the protocol is modified.

To the extent that iDirect has rights to control the protocol itself, iDirect grants rights to implement the protocol to all, without restriction.

Use of the trademark "OpenBMIP" to describe an unmodified implementation of this protocol is unrestricted. Use the term "modified OpenBMIP" to describe a variant of this protocol, is also unrestricted; however the document containing the term "modified OpenBMIP" refers to this document.

While iDirect, Inc. strives to make the information in this document as accurate as possible, iDirect makes no claims, promises, or guarantees about the accuracy, completeness, or adequacy of the contents, and expressly disclaims liability for errors and omissions. No warranty of any kind, whether implied, expressed, or statutory, including but not limited to the warranties of non-infringement of third party rights, title, merchantability, or fitness for a particular purpose, is given with respect to the contents of this document. iDirect, Inc. reserves the right to change or update this document at any time.

Certification

You may certify your compliance with the test suite yourself. If you do, you are free to use the trademark "OpenBMIP" freely for any product that you have certified.

Your use of the OpenBMIP trademark authorizes any OpenBMIP implementer to validate your implementation and publish the results, referring to your product by company and product name, if the implementer finds your implementation to be non-compliant. A finding of non-compliance will not be published until thirty days after the OpenBMIP member notifies you of the finding. At your option, the implementer's published finding of non-compliance will include a reference to a statement in rebuttal by you.

Getting Help

> ***NOTE:*** Not all users of BMIP are entitled to TAC support. Only iDirect customers are entitled to TAC support.

The iDirect Technical Assistance Center (TAC) and the iDirect Government Technical Assistance Center (TAC) are available to provide assistance 24 hours a day, 365 days a year. Software user guides, installation procedures, FAQs, and other documents that support iDirect and iDirect Government products are available on the respective TAC Website:

- Access the iDirect TAC Website at http://support.idirect.com
- Access the iDirect Government TAC Website at http://tac.idirectgov.com

The iDirect TAC may be contacted by telephone or email:

- Telephone: 703.648.8151
- E-mail: tac@idirect.net

The iDirect Government TAC may be contacted by telephone or email:

- Telephone: 703.648.8111
- Email: tac@idirectgov.com

iDirect and iDirect Government produce documentation that are technically accurate, easy to use, and helpful to our customers. Please assist us in improving this document by providing feedback. Send comments to:

- iDirect: techpubs@idirect.net
- iDirect Government: techpubs@idirectgov.com

For sales or product purchasing information contact iDirect Corporate Sales at the following telephone number or e-mail address:

- Telephone: 703.648.8000

E-mail: sales@idirect.net

1 Introduction

This chapter contains the following sections:

- Scope
- Terminal Overview

1.1 Scope

This document describes a monitoring and control interface between the modem and block up converter (BUC) or Transceiver components of a satellite terminal. The most important function of this interface is to enable calibration of the terminal's transmitted power for regulatory compliance.

1.1.1 Tailoring

OpenBMIP may be tailored to fit the application. For example, systems with a fixed BUC local oscillator frequency may be implemented without support for the Set Local Oscillator on page 30 command.

In order to allow different BUC or Transceiver vendors to provide the end user with a richer command set beyond those required by the monitoring and control interface, the protocol allows for the definition of vendor-specific commands (see Vendor Specific Commands on page 44).

1.2 Terminal Overview

The Modem

- Acts as a satellite modem
- Coordinates with the satellite network to provide monitoring and control functions
- Provides IF, reference frequency, and control signals to the BUC or Transceiver
- Reads gain calibration data from the BUC and from other sources
- Adjusts its IF level to produce the correct RF power level at the BUC output for satellite link operation

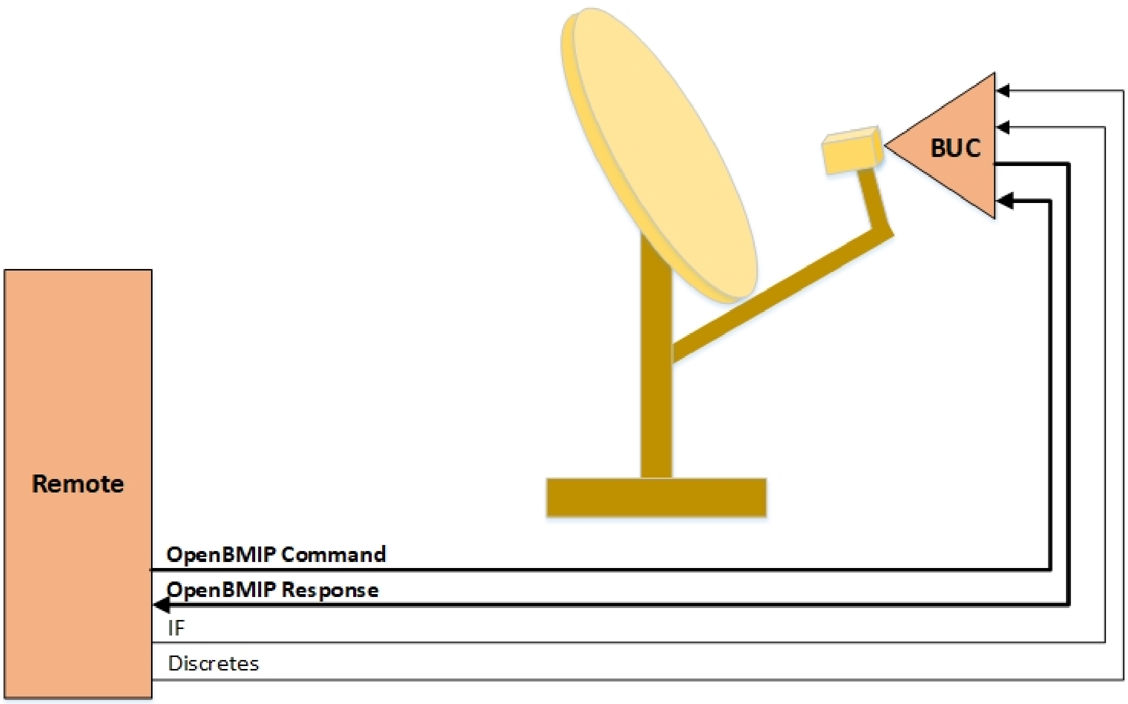


Figure 1-1. BUC and Remote (Modem) Connection

The BUC or Transceiver

- Upconverts the IF from the modem to the required satellite uplink frequency (RF)
- Amplifies the upconverted RF power with fixed gain
- Interacts with the modem to enable calibrated control of the RF output power

1.3 BUC Functional Requirements to Support OpenBMIP

A typical BUC implementation incorporates the following key functions to support calibrated operation using the OpenBMIP protocol:

- An RF RMS power detector (typically placed at the input of the final power amp, but may be implemented at another appropriate point in the signal chain to support a fully calibrated power level, from IF input to RF output of the BUC)
- A temperature sensor to support temperature correction of calibrated values
- A microcontroller with these interfaces and features:
    - One-time-programmable memory to store ID numbers
    - Factory-programmable memory (read-only in the field) to store calibration data
    - Analog to digital converters for the power detector and the temperature sensor
    - A physical layer interface (serial or Ethernet) for connection to the modem.

OpenBMIP may be implemented as either an RS-422 serial connection or a UDP connection, through the Ethernet connector.

The BUC EEPROM must have enough memory to store the following information:

- In a one-time-programmable (OTP)
    - BUC Part Number (BPN)
        - 13 bytes; unused trailing bytes filled with "x"
    - BUC Manufacturer ID (MID)
        - 2 bytes
    - BUC Serial Number (BSN) with date code and Revision; (see BUC Command Syntax on page 19)
        - 9 bytes
    - BUC Functional ID (FID) per Appendix D: , BUC Wattage Numbering
        - 6 bytes; for example: 5W, 29.0-30.0GHz, Standard:: ' 050300
- In an erasable page (with lock feature)
    - Calibration file (XML Format)
    - Check Sum

> ***NOTE:*** It is anticipated that some applications of this interface will allocate functionality to other system components. For example, the microcontroller and EEPROM might not be co-located with the BUC. Such differences should be implemented so they are transparent from the perspective of the modem.

2 Protocol Specification

This chapter contains the following sections:

2.1 Introduction

The purpose of this protocol is to provide specifications for communication between the satellite modem (client) and BUC (server) using a monitoring and control interface.

This protocol is a Client-Server protocol. The Client is the satellite modem; the Server is the BUC. Only one modem at a time is connected to a BUC. The modem always initiates communication with the BUC. The BUC never transmits without receiving a request from the modem.

2.2 Frame Description

The table below shows the structure of an OpenBMIP message.

Table 2-1. BMIP Message Format

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	TYPE	CMD_CODE						
1	RSP_CODE					CMD_VERS		
2	CMD_SEQ (high byte)							
3	CMD_SEQ (low byte)							
4	DATA_LEN (high byte) DATA_LEN = M							
5	DATA_LEN (low byte)							
6	DATA BYTE 0							
7	DATA BYTE 1							
...	...							
n-4	DATA BYTE M-1 (ASCII NULL character)							
n-3	FCS(high byte) FCS is computed from bytes 0…n-4							
n-2	FCS(low byte)							
n-1	END (0xC0)							

All message fields are binary, with the exception of the DATA field, which is NULL-terminated ASCII, including the representation of decimal or hexadecimal numbers. If DATA field length is 0, the terminating NULL character is omitted.

2.2.1 Time Outs

If the recipient receives message characters, with more than 2 seconds elapsed between characters, it should time out and assume a connection fault has occurred. In the case of the BUC (server), it should send RSP_CODE 0x02 (see RSP_CODE on page 8). There is no concept of a message queue in OpenBMIP; the client (modem) must wait for a response before sending another message to the server (BUC). This provides natural throttling of flow rate. If no response is received within 5 seconds, the client should time out and assume a connection fault. During a file transfer, if the modem does not acknowledge a message from the BUC within 5 seconds, the BUC should time out and send a message with RSP_CODE = 0x08. If 5 seconds is impractical, the BUC vendor should specify what the BUC file transfer timeout is. The same rules apply during a file transfer.

2.2.2 Empty Message

If the recipient receives an END character (0xC0), but has not received corresponding prior message characters, it should silently ignore it. This is a standard part of the SLIP protocol (see IETF RFC-1055 SLIP Protocol on page 11).

2.2.3 TYPE

One bit is used to identify whether the message is a command or response.

- 0x0: command
- 0x1: response

2.2.4 CMD_CODE

Seven bits are used; there are up to 128 command codes. Codes are assigned as follows:

- 0x00 to 0x3F: BUC Commands
- 0x40 to 0x4F: Transceiver Commands
- 0x50: Extended Status
- 0x60 to 0x7E: Custom Commands, Client side, available for third-party use

2.2.5 RSP_CODE

Five bits are used; there are up to 32 response codes. The server responds to every command with either a failure or success. Other error codes can be defined per command as required. The following values are defined:

- 0x00: Failure (of a type not in the list below)
- 0x01: Success
- 0x02: Message receipt timeout
- 0x03: Invalid command sequence number (future use)

- 0x04: Invalid FCS detected in modem command
- 0x05: Incompatible command version
- 0x06: Invalid third-party command
- 0x07: Data length and data field size mismatch
- 0x08: File transfer fail
- 0x09: BUC Input Buffer Overrun
- 0x0A: Modem Command Data Field Error
- 0x0B: Upgrade fail
- 0x1E: Unimplemented command
- 0x1F: Unrecognized command

2.2.6 CMD_VERS

Three bits are used to identify the version of the protocol. The current protocol is version 2 and subsequent versions will be one up. CMD_VERS is only incremented if there is a change to the BMIP message structure described above; CMD_VERS is not incremented with each release of the BMIP standard. If the BUC supports at least the same version as the modem, it will be compatible; otherwise it should report an error. The BUC should provide backward compatibility for versions 0 through 2.

NOTE: The CMD_VERS only refers to the BMIP message structure, and is not the same as the <cal_format_version> in the BUC Calibration file, which reflects the version of this document.

2.2.7 CMD_SEQ

Sixteen bits are used; this is the command sequence number used to match requests to responses. The command sequence number starts with 0 for the initial command and is incremented by one for each additional command. After the value reaches 0xff 0xff, it wraps to 0x00 0x00. The BUC response should echo the CMD_SEQ value from the command it is responding to. In the case of a timeout, the BUC should use the most recent received CMD_SEQ value. The BUC is not required to detect CMD_SEQ errors; this error code is reserved for future use.

2.2.8 DATA_LEN

Sixteen bits are used; this is the message length in bytes excluding the header. This is the length of the ASCII DATA field, including its NULL (0x00) terminator. If the DATA field length is 0, the NULL terminator may be included or omitted.

2.2.9 DATA

This is ASCII data; its content and length are command-dependent. It is terminated with an ASCII NULL (0x00) character, and maximum length is 247. Therefore, to send the hexadecimal value 0xABC, one sends four bytes: 65, 66, 67, 0. To send the decimal value 123, one sends four bytes: 49, 50, 51, 0. The DATA field is case insensitive, for hexadecimal values, filenames, or any other content. Only the following characters are allowed in the DATA field (in addition to the NULL terminator at the end):

{0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ,._=+-}

2.2.10 FCS

Sixteen bits are used for the Frame Check Sequence (FCS); this is computed with the algorithm used by PPP for its FCS calculation. Refer to RFC-1171 for a sample implementation (http://tools.ietf.org/html/rfc1171#page-44). The FCS is calculated from all bytes which precede the FCS.

2.3 IETF RFC-1055 SLIP Protocol

The message is packaged and transmitted using escape sequence insertion and checks in accordance with IETF RFC-1055 SLIP protocol. Refer to RFC-1055 for a sample implementation in the C programming language. This sample algorithm explains the steps to send a message:

Optionally, send an initial END (0xC0) byte to flush out any data that may have accumulated in the receiver due to line noise. The initial END byte does not count toward CMD_SEQ, DATA_LEN, or FCS values. The receiver should silently ignore an END with no corresponding message.

- Construct the message, including DATA_LEN and FCS. The DATA_LEN and FCS computed values are unaffected by the substitutions below, but they may themselves be subject to substitution.

For each byte in the message make the following substitutions:

  - 0xC0 becomes 0xDB 0xDC
  - 0xDB becomes 0xDB 0xDD
  - any other value is unchanged

- Send the final END (0xC0) byte. ***The final END byte does not count toward DATA_LEN or FCS values.***
- Apply the above substitutions after constructing the message, so that DATA_LEN and FCS are unaffected.

To receive a message:

- Receive the message, including DATA_LEN and FCS. The DATA_LEN and FCS computed values are unaffected by the substitutions below, but they may themselves be subject to substitution.

For each byte received make the following substitutions:

  - 0xDB 0xDC becomes 0xC0
  - 0xDB 0xDD becomes 0xDB
  - any other value is unchanged

- Receive the final END (0xC0) byte. ***The final END byte does not count toward DATA_LEN or FCS values.***
- Compare the FCS computed, with the FCS received; send an "Invalid FCS" response code if computed FCS does not match received FCS.

2.4 Client/Server Interaction Sequence Diagram

Figure 2-1 shows a sample interaction between the client (CM) and the server (BUC) represented as a sequence diagram. As can be seen from the sequence diagram, it is expected that the CM is always the client while the BUC is always the server.

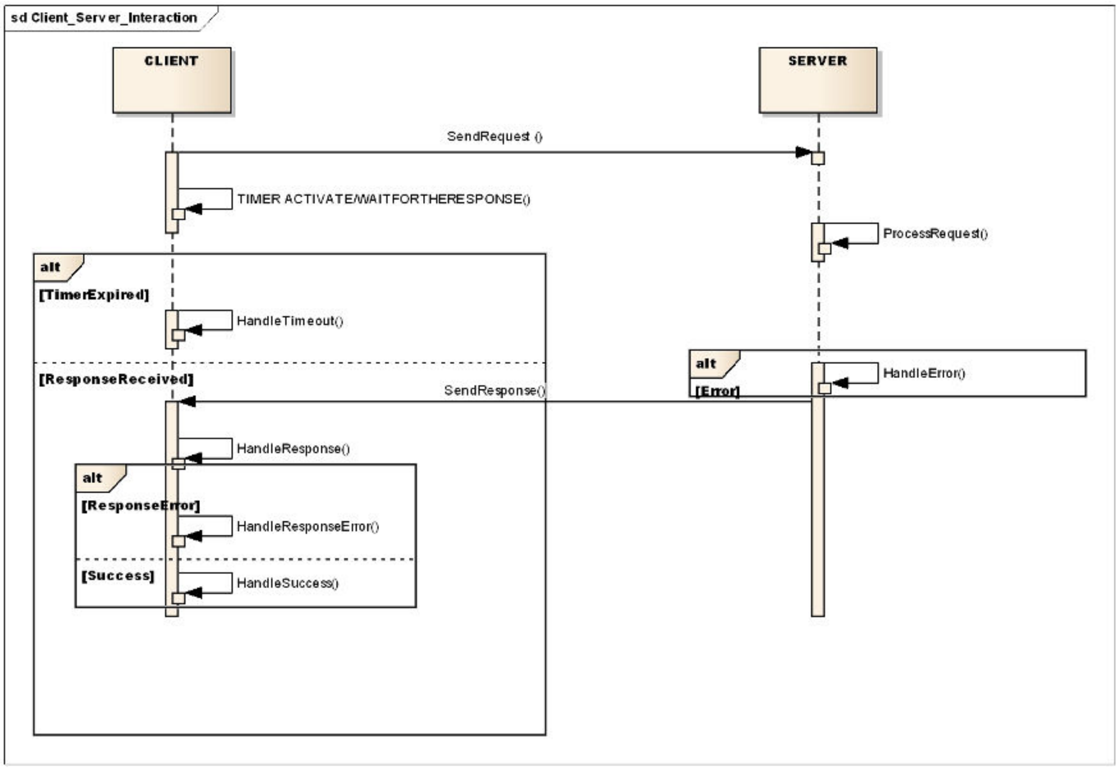


Figure 2-1. Client-Server Interaction Sequence

2.5 Example of BUC to Set IF Lowpass Harmonic Filter Selection

In this example, the modem commands the BUC to Set IF Lowpass Harmonic Filter Selection (SBANDSEL) followed by the BUC's response.

2.5.1 Construction of Set IF Lowpass Harmonic Filter Selection Command

Objective of message: Set IF Lowpass Harmonic Filter Selection to low band

2.5.2 Command Raw Byte Sequence

Table 2-2. Command Raw Byte Sequence

Header	**Data**	**FCS**	**END**
0x7 0x0 0x0 0x1 0x0 0x8	0x6c 0x6f 0x77 0x62 0x61 0x6e 0x64 0x0	0x5a 0x73	0xc0

2.5.3 Header Deconstruction by Field

Table 2-3. Header Deconstruction by Field

FIELD	**DATA**
TYPE: 0 =>	COMMAND
CMD_CODE: 7 =>	SBANDSEL
RSP_CODE:	0
CMD_VERS:	0
CMD_SEQ:	1
DATA_LEN:	8

2.5.4 Header Deconstruction by Bit

Table 2-4. Header Deconstruction by Bit

Byte	b7	b6	b5	b4	b3	b2	b1	b0
0	**Type**	**CMD_CODE**						
	0	0	0	0	0	1	1	1
1	**RSP_CODE**					**CMD_VERS**		
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**
2	**CMD_SEQ (high byte)**							
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**
3	**CMD_SEQ (low byte)**							
	0	**0**	**0**	**0**	**0**	**0**	**0**	**1**
4	**DATA_LEN (high byte)**							
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**
5	**DATA_LEN (low byte)**							
	0	**0**	**0**	**0**	**1**	**0**	**0**	**0**

2.5.5 Data Payload

lowband + NULL => {'l','o','w','b','a','n','d',0}

2.5.6 Frame Check Sequence

0x5a73

2.5.7 Expected BUC Response Raw Byte Sequence

Table 2-5. Expected BUC Response Raw Byte Sequence

Header	Data	FCS	END
0x87 0x8 0x0 0x1 0x0 0x0		0xee 0x00	0xc0

2.5.8 Expected BUC Response Header Deconstruction by Field

Table 2-6. Expected BUC Response Header Deconstruction by Field

FIELD	DATA
TYPE: 1 =>	RESPONSE
CMD_CODE: 7 =>	SBANDSEL
RSP_CODE:	1
CMD_VERS:	0
CMD_SEQ:	1
DATA_LEN:	0

2.5.9 Header Deconstruction by Bit

Table 2-7. Header Deconstruction by Bit

Byte	b7	b6	b5	b4	b3	b2	b1	b0
0	**Type**	**CMD_CODE**						
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**
1	**RSP_CODE**					**CMD_VERS**		
	0	**0**	**0**	**0**	**1**	**0**	**0**	**0**
2	**CMD_SEQ (high byte)**							
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**
3	**CMD_SEQ (low byte)**							
	0	**0**	**0**	**0**	**0**	**0**	**0**	**1**
4	**DATA_LEN (high byte)**							
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**
5	**DATA_LEN (low byte)**							
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**

2.5.10 Data Payload

Empty

2.5.11 Frame Check Sequence

0xee00

2.6 Physical Layer

OpenBMIP may be used with either serial RS-422 or Ethernet as the physical layer.

2.6.1 Serial RS-422 Mode

Physical layer data transmission.

2.6.2 Communication Parameters

This mode is used for the data transmission at the physical layer; settings are:

Table 2-8. Serial Communication Settings

FIELD	DATA
Baud Rate	38400
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	None
Handshaking	None

2.6.3 Data Ordering

For serial transfers, bits are sent beginning with Bit 0 (LSB) for any given byte. So, in order of transmission, a message is sent over the serial interface as follows:

(Byte 0, Bit 0), (Byte 0, Bit 1), ... , (Byte 0, Bit 7), (Byte 1, Bit 0), ..., (Byte n-1, Bit 7)

2.6.4 Ethernet Mode

The purpose of this protocol is to provide specification for communication between the modem (client) and BUC Proxy (server) using an Ethernet Interface. The BUC Proxy will communicate with the BUC and provide responses to the modem.

The Modem sends OpenBMIP messages to the BUC's IPv4 address (defined in a separate terminal-specific document), on UDP port 6001. For file transfers, the modem sends either (i) Kermit Over TCP/IP commands to the BUC's IPv4 address on TCP port 1649, or (ii) TFTP. The BUC accepts a connection from one client at a time. The modem always initiates the connection, similar to the Serial protocol.

The OpenBMIP commands described in BUC Command Syntax on page 19 will be sent via UDP. Each UDP packet contains a single command or a response. Figure 2-2 shows the UDP packet structure:

Ethernet Header	IP Header	UDP Header	OpenBMIP Frame

Figure 2-2. UDP Ethernet Packet Format

The OpenBMIP Frame format in Figure 2-2 is specified in Frame Description on page 7. The interaction sequence between the client and server has been specified in Client/Server Interaction Sequence Diagram on page 12. All commands operate the same as with a serial connection; except for those related to file transfer (see BUC Command Syntax on page 19).

2.7 BUC Command Syntax

The following sections detail individual commands and their responses. The DATA and Response DATA are shown as example strings enclosed in curly braces: {sample}. Each string should also include a NULL terminator, not shown. If the Command DATA or Response DATA string is blank, the NULL terminator may be included or omitted.

Table 2-9. Commands in Numerical Order

CMD_CODE	Section	Function Name
1 (0x01)	Get IF Lowpass Harmonic Filter Selection on page 24	GFAULTSTATUS
2 (0x02)	Get PA Power on page 23	GPAPOWER
3 (0x03)	Get Product ID on page 20	GPRODUCTID
4 (0x04)	Get Heartbeat on page 23	GHEARTBEAT
5 (0x05)	Get Local Oscillator on page 29	GLOSCILLATOR
6 (0x06)	Get IF Lowpass Harmonic Filter Selection on page 24	GBANDSEL
7 (0x07)	Set IF Lowpass Harmonic Filter Selection on page 26	SBANDSEL
8 (0x08)	Set Local Oscillator on page 30	SLOSCILLATOR
9 (0x09)	Reset Status Latch on page 33	RSTATUSLATCH
10 (0x0A)	Send File on page 36	SENDFILE
11 (0x0B)	File Transfer Status on page 37	STATUSFILETRANSFER
12 (0x0C)	Receive File on page 36	RECEIVEFILE
13 (0x0D)	Get Software Version on page 33	GSOFTWAREVER
14 (0x0E)	Get Calibration File on page 34	GCALIBFILE
15 (0x0F)	Set Calibration File on page 35	SCALIBFILE
16 (0x10)	Prepare for Upgrade on page 35	PREP_UPGRADE
17 (0x11)	BUC Reset on page 34	BUC_RST
18 (0x12)	Get Latched Fault Status on page 32	GLFAULTSTATUS
19 (0x13)	Get Power Amp State on page 27	GPOWERAMP

Table 2-9. Commands in Numerical Order (Cont.)

20 (0x14)	Set Power Amp State on page 28	**SPOWERAMP**
21 (0x15)	Update Simulator on page 38	**UPDATESIM**
22 (0x16)	Get Polarization on page 39	**GPOLARIZATION**
23 (0x17)	Set Polarization on page 40	**SPOLARIZATION**
24 (0x18)	Request IFL Calibration Permission on page 41	**REQIFLCAL**
25 (0x19)	IF Calibration Status on page 42	**IFLCALSTATUS**

2.7.1 Get Product ID

This command is used to query the BUC EEPROM contents. The contents of the response from the BUC follows the format and response information as described below.

Table 2-10. Get Product ID

FIELD	DATA
Command Description	Echo manufacturer name and BUC model number
CMD_CODE	3
Functional Name	GPRODUCTID
Command DATA	{}
Response DATA	{BPN=1234567890xxxMID=mmBSN=A12345A33,FID=050300}

The BUC identifiers are:

- BUC Part Number (BPN)
  - 13 bytes; unused trailing bytes filled with "x"
- BUC Manufacturer ID (MID) assigned by iDirect
  - 2 bytes
- BUC Serial Number (BSN) **Must be unique;** see suggested example in **Figure 2-3**.
  - 9 bytes
- BUC Functional ID (FID) per Client/Server Interaction Sequence Diagram on page 12
  - 6 bytes; Example: 5W, 29.0-30.0GHz, Standard:: --> 050300

For more information, see BUC Wattage Numbering on page 79.

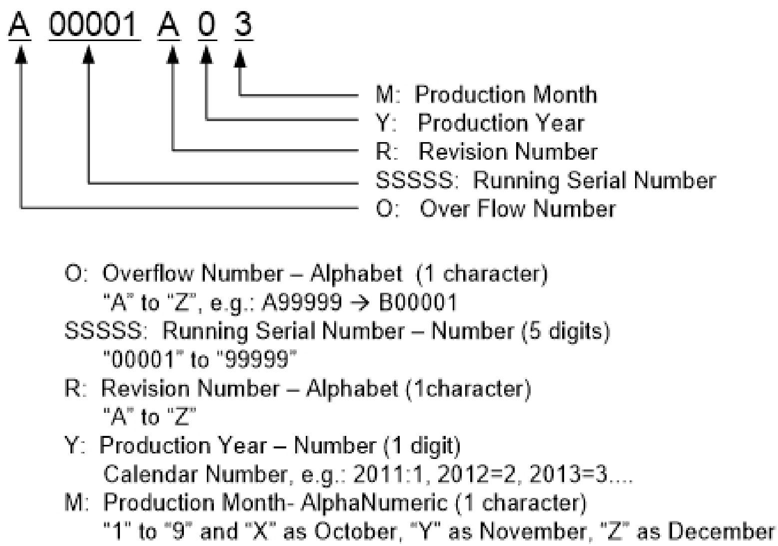
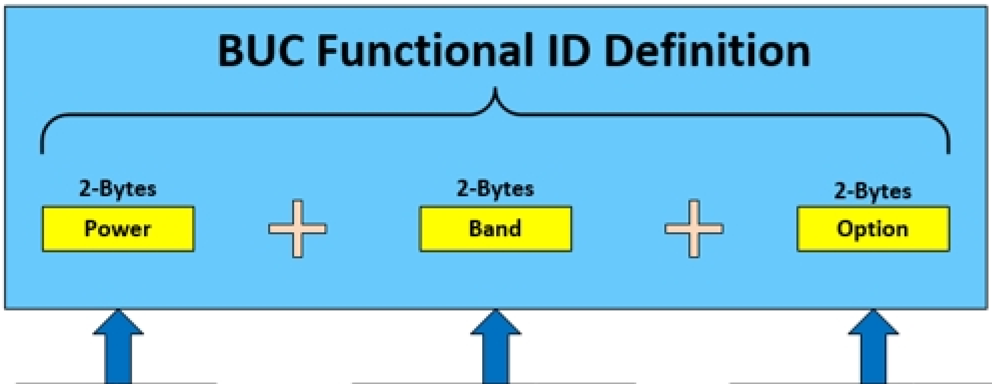


Figure 2-3. BUC Serial Number with Embedded Date Code and Revision

	NOTE: Codes for the FID 'Power' field are detailed in BUC Wattage Numbering on page 79



Power	Code
1W	01
2W	02
3W	03
...	
9W	09
10W	10
...	
16W	16
17W	17
...	
99W	99
100W	A0
101W	A1
...	
0.5W	H0
1.5W	H1
2.5W	H2
...	
9.5W	H9
10.5W	HA
...	
16.5W	I0
17.5W	I1
...	
99.5W	N3
100.5W	N4
101.5W	N5
...	

Frequency Range	Code
29.0 - 29.5 GHz	01
29.5 - 30.0 GHz	02
29.0 - 30.0 GHz	03
30.0 - 31.0 GHz	04
29.5 - 31.0 GHz	05
29.0 - 30.5 GHz	06
29.0 - 31.0 GHz	07
28.0 - 30.0 GHz	08
27.5 - 30.0 GHz	09
...	

Option	Code
Single Band Single Polarization	00
Single Band Dual Polarization	01
Multiple Band Single Polarization	02
Multiple Band Dual Polarization	03
...	

Figure 2-4. BUC Functional ID

2.7.2 Get PA Power

This command will be used for PA power purposes.

Table 2-11. Get PA Power

FIELD	DATA
Command Description	Echo raw power in terms of digital voltage with the resolution as specified in the calibration file.
CMD_CODE	2
Functional Name	GPAPOWER
Command DATA	{}
Response DATA	{ABC} ASCII representation of hex value from A to D converter.

NOTE: This function should operate correctly even if the final power amp stage is disabled by the Keyline discrete or by the Set Power Amp State command.

2.7.3 Get Heartbeat

This command will be used for heartbeat purposes.

Table 2-12. Get Heartbeat

FIELD	DATA
Command Description	Echo S/N, temperature, and fault status. (Manufacturing Date code may be added, but is optional.)
CMD_CODE	4
Functional Name	GHEARTBEAT
Command DATA	{}
Response DATA	{BSN=A12345A33, TEMP=+48.8,FSTATUS=lolock, fannormal,outputmute, normaltemp}

2.7.4 Get IF Lowpass Harmonic Filter Selection

This command is used to identify method(s) used to set the BUC IF Harmonic Lowpass Filter (if supported).

Table 2-13. Get IF Lowpass Harmonic Filter Selection

FIELD	DATA
Command Description	Get IF Harmonic Filter Selection
CMD_CODE	6
Functional Name	GBANDSEL
Command DATA	{}
Response DATA	{lowband,gpio_0,serial_0,tone_1}

The IF Lowpass Filter Selection can be controlled via multiple different interfaces:

- GPIO (RS-422)
- OpenBMIP (RS-422 or Ethernet) (called “serial” in response data for backwards compatibility)
- 27MHz Tone (IFL Cable)

If single interface is used, functionality is clear. However, in some installations, multiple interfaces could be connected to the BUC and the control states need explaining:

- If the GPIO input is unused, the GPIO defaults to 0, highband/wideband, lowpass filter bypassed
- If no OpenBMIP command is received (or it is rejected), “serial” defaults to highband/wideband, lowpass filter bypassed
- If no 27MHz tone is received (or is unused), tone defaults to highband/wideband, lowpass filter bypassed
- If any input requires lowband, and the BUC supports the Lowpass Harmonic Filter, the BUC operates in lowband, lowpass filter inserted

Table 2-14. IF Lowpass Harmonic Filter Select Logic

State No.	GPIO	OpenBMIP	27MHz Tone	Response DATA
0	0 or unused	highband or unused	absent or unused	highband,gpio_0,serial_0,tone_0
1	0	highband	present	lowband,gpio_0,serial_0,tone_1
2	0	lowband	absent	lowband,gpio_0,serial_1, tone_0
3	0	lowband	present	lowband,gpio_0,serial_1,tone_1
4	1	highband	absent	lowband,gpio_1,serial_0,tone_0
5	1	highband	present	present lowband,gpio_1,serial_ 0,tone_1
6	1	lowband	absent	lowband,gpio_1,serial_1,tone_0
7	1	lowband	present	lowband,gpio_1,serial_1,tone_1

2.7.5 Set IF Lowpass Harmonic Filter Selection

This command is used to select the BUC IF Harmonic Lowpass Filter function.

Table 2-15. Set IF Lowpass Harmonic Filter Selection

FIELD	DATA
Command Description	Set IF Harmonic Filter Selection
CMD_CODE	7
Function Name	SBANDSEL
Command DATA	{highband}
Response DATA	{}

The Lowpass Harmonic Filter Section has two states:

- Highband/Wideband: lowpass harmonic filter bypassed, use upper half of the IF frequency range. Can use entire band, if harmonics are not an issue
- Lowband: lowpass harmonic filter inserted, use lower half of the IF frequency range

An example of the lowpass harmonic filter functionality is indicated below.

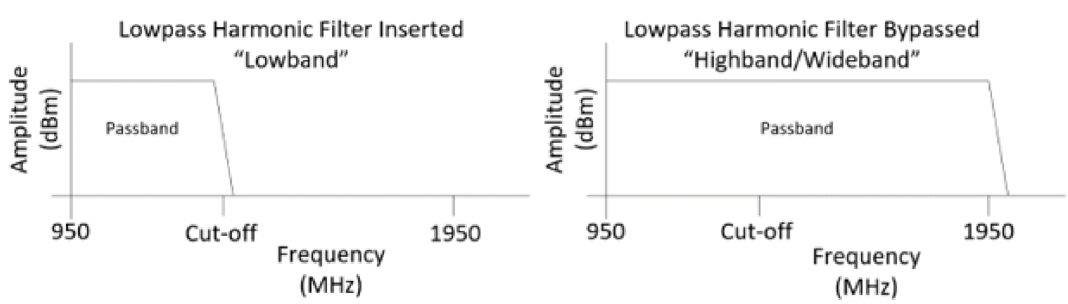


Figure 2-5. Example Lowpass Harmonic Filter Operation

If this command is sent to a multi-band BUC that supports this feature, all bands will be configured with the same Lowpass Harmonic Filter selection. If the multi-band BUC does not support the Lowpass Harmonic Filter, then it would respond with an RSP code of preferably "Unrecognized Command" or alternately "Unimplemented Command". This is the same expected performance of other OpenBMIP commands that a BUC doesn't recognize or support.

It should be noted that this is only one of the methods by which the Lowpass Harmonic Filter can be commanded. The BUC may also support selection by Tone or using a discrete signal (GPIO). The selection criterion to be followed by the BUC when one or more of these methods are used is as explained inGet IF Lowpass Harmonic Filter Selection on page 24.

2.7.6 Get Power Amp State

This command is used to identify the method the power amplifier is controlled and the current state of the power amplifier.

Table 2-16. Get PowerAmp State

FIELD	DATA
Command Description	Get state of final power amplifier
CMD_CODE	19
Functional Name	GPOWERAMP
Command DATA	{}
Response DATA	{enabled, gpio_0,BMIP_0}

In the response data,

- enabled/disabled indicates the actual state of the Power Amp
- gpio_0/1 indicates the PA has been commanded to be On (0) or Off (1) via Keyline GPIO.
- BMIP_0/1 indicates the PA has been commanded to be On (0) or Off (1) via BMIP.

For keyline enabled BUCs where the GPIO discrete line has been left disconnected, the response should report a predictable default state (gpio_0 or gpio_1).

For BUCs with no keyline support, the keyline indicator should default to gpio_0 (PA On).

Explanation of control states:

- BUC PA disabled by OpenBMIP command: it remains disabled until re-enabled by another command, regardless of the state of the discrete signal
- BUC PA disabled by discrete signal: the PA is disabled, regardless of any commands received.
- If the PA is disabled by any inputs, it remains disabled until all inputs re-enable it.

Table 2-17. Power Amp Logic

State No.	Control Source		Response DATA
	Keyline	OpenBMIP	
0	PA On	enabled or unspecified	enabled,gpio_0,BMIP_0
1	PA On	disabled	disabled,gpio_0,BMIP_1
2	PA Standby	enabled	disabled,gpio_1,BMIP_0
3	PA Standby	disabled	disabled,gpio_1,BMIP_1

2.7.7 Set Power Amp State

This command is used to enable or disable the final power amp in the BUC, using the OpenBMIP interface. It should be noted that this is only one of the methods by which power amp state can be commanded. The BUC may also support selection by OpenAMIP (in terminals configured with a common controller for the BUC and the antenna) or using a discrete signal (GPIO) known as “Keyline.” The selection criterion to be followed by the BUC when one or more of these methods are used is as explained in Get PA Power on page 23.

Table 2-18. Set Power Amp State

FIELD	DATA
Command Description	Set state of final power amplifier
CMD_CODE	20
Functional Name	SPOWERAMP
Command DATA	{enabled}
Response DATA	{}

2.7.8 Get Local Oscillator

This command is used to get the Local Oscillator of the band where the BUC supports more than one band, and selection is controlled via OpenBMIP. For single or multi-band BUCs which use OpenAMIP to select the Band, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses.

Table 2-19. Get Local Oscillator

FIELD	DATA
Command Description	Get the Local Oscillator Frequency
CMD_CODE	5
Functional Name	GLOSCILLATOR
Command DATA	{}
Response DATA	{28.05GHz}

2.7.9 Set Local Oscillator

For single or multi-band BUCs which use OpenAMIP to select the Band, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses.

> ***NOTE:*** The {lock/unlock} in 'Response DATA' is optional, and describes whether the LO has locked to the new band specified in the 'Command DATA' field. This response is to indicate lock status to the modem more quickly, and remove the need for subsequent query; the BUC may briefly delay its response until lock/unlock is determined, but the delay should not interrupt the regular Client/Server communication flow described in Client/Server Interaction Sequence Diagram on page 12.

Table 2-20. Set Local Oscillator

FIELD	DATA
Command Description	Set the Local Oscillator Frequency
CMD_CODE	8
Functional Name	SLOSCILLATOR
Command DATA	{28.05GHz}
Response DATA	{lock/unlock} or {}

2.7.10 Get Fault Status

This command is used to set the Local Oscillator of the band where the BUC supports more than one band, and selection is controlled via OpenBMIP. This command is used for getting the current fault status.

Table 2-21. Get Fault Status

FIELD	DATA
Command Description	Display the fault status within the BUC: display current faults and latched faults
CMD_CODE	1
Functional Name	GFAULTSTATUS
Command DATA	{}
Response DATA	{lolock, fannormal,outputmuteinternal,normaltemp,fault} Any valid comma-separated combination of: lolock/lounlock, fannormal/fastandbv/fanfault outputnormal/outputoverdriven/outputmute/outputmuteinternal, normaltemp/overtemp, fault1, fault2, ..., fault8

NOTE: "fanstandby" means the fan(s) are purposely off to self-warm the BUC at low ambient temperature.

outputmute should represent normal operational mute states, whether a result of being instructed to mute by command or keyline, or muting while reconfiguring LO. **outputmuteinternal** is intended to represent faults, where the BUC has muted itself as a result of some internal condition. Vendor may use one of {fault1, ... ,fault8} to specify the cause of that internal fault.

"fault*n*" strings are not mutually exclusive, and represent custom faults as defined by the device vendor.

2.7.11 Get Latched Fault Status

This command is used to report historical faults. For status definitions, see on page 30. The following faults should be latched: lounlock, fanfault, outputoverdriven, overtemp.

Table 2-22. Get Latched Fault Status

FIELD	DATA
Command Description	Request the latched faults within the BUC. Request only latched faults, that is, faults that have previously occurred but have not been cleared.
CMD_CODE	18
Function Name	GLFAULTSTATUS
Command DATA	{}
Response DATA	{outputoverdriven} Any valid comma-separated combination of: lounlock, fanstandby/fanfault, outputoverdriven/outputmuteinternal, overtemp, fault1, fault2, fault3, fault4, fault5, fault6, fault7, fault8

2.7.12 Reset Status Latch

This command resets all the Latched Fault Status, see Get Latched Fault Status on page 32. Any fault event will be latched and readable, until the BUC receives this command. If the fault condition is persistent, this command will have no effect.

Table 2-23. Reset Status Latch

FIELD	DATA
Command Description	Clear all latched status
CMD_CODE	9
Function Name	RSTATUSLATCH
Command DATA	{}
Response DATA	{}

2.7.13 Get Software Version

This command will be used for getting the software version.

Table 2-24. Get Software Version

FIELD	DATA
Command Description	Get the version number
CMD_CODE	13
Function Name	GSOFTWAREVER
Command DATA	{}
Response DATA	{03.0123}

2.7.14 BUC Reset

This command will be used for resetting the BUC.

Table 2-25. BUC Reset

FIELD	DATA
Command Description	Reset the BUC
CMD_CODE	17
Function Name	BUC_RST
Command DATA	{}
Response DATA	{}

2.7.15 Get Calibration File

This command will be used for BUC preparation sending purposes.

Table 2-26. Get Calibration File

FIELD	DATA
Command Description	Prepare BUC to send the calibration file
CMD_CODE	14
Function Name	GCALIBFILE
Command DATA	{}
Response DATA	{buc.cal.xml, 6123} file name and file size (in bytes)

NOTE: After response, perform the file transfer (See BUC Command Syntax on page 19)

2.7.16 Set Calibration File

This command will be used for BUC preparation reception purposes.

Table 2-27. Set Calibration File

FIELD	DATA
Command Description	Prepare BUC to receive the calibration file
CMD_CODE	15
Function Name	SCALIBFILE
Command DATA	{buc.cal,xml,6123} file name and file size (in bytes)
Response DATA	{}

NOTE: After response, perform the file transfer. (See BUC Command Syntax on page 19.)

2.7.17 Prepare for Upgrade

This command will be used for BUC software upgrade purposes.

Table 2-28. Prepare for Upgrade

FIELD	DATA
Command Description	Prepare BUC to receive a software upgrade
CMD_CODE	16
Functional Name	PREP_UPRADE
Command DATA	{buc.sw,127383} file name and file size (in bytes)
Response DATA	{}

NOTE: After response, perform the file transfer (See BUC Command Syntax on page 19)

2.7.18 Send File

This command will be used for starting the file transfer.

Table 2-29. Send File

FIELD	DATA
Command Description	Start the file transfer
CMD_CODE	10
Functional Name	SENDFILE
Command DATA	{buc.cal.xml, 6123} file name and file size (in bytes)
Response DATA	{}

2.7.19 Receive File

This command will be used for receiving file purposes.

Table 2-30. Receive File

FIELD	DATA
Command Description	Receive a file
CMD_CODE	12
Functional Name	RECEIVEFILE
Command DATA	{buc.cal.xml, 6123} file name and file size (in bytes)
Response DATA	{}

2.7.20 File Transfer Status

The purpose of this command is for the modem to confirm the success or failure (1 or 0) of the file transfer initiated with SENDFILE (CMD_CODE 10) Send File on page 36 or RECEIVEFILE (CMD_CODE 12) Receive File on page 36. The BUC may choose to close the file transfer port in response to this command.

Table 2-31. File Transfer Status

FIELD	DATA
Command Description	Status of the latest file transfer transaction
CMD_CODE	11
Function Name	STATUSFILETRANSFER
Command DATA	{buc.cal.xml} file name
Response DATA	{}

2.7.21 Update Simulator

A software-only simulator is available in order to provide for software development and system testing without requiring an actual BUC. This simulator does not have any method for measuring the transmit output power and frequency or for monitoring the control lines from each remote. Because of this, a method by which the simulator can be informed of the remote state must be provided. This message provides everything the simulator needs in order to test One Touch Calibration (OTC) and normal BUC operation.

Table 2-32. Update Simulator

FIELD	DATA
Command Description	Update the simulator
CMD_CODE	21
Function Name	UPDATESIM
Command DATA	{1200.0,-25.0,BPSK,1,1,0} Command data fields in order: IFL Frequency (MHz), Transmit power output (dBm), Modulation (BPSK or QPSK) Keyline enable state (1=enable, 0=disable) Reference enable state (1=enabled, 0=disabled) Tone enabled state (1=lowband, 0=highband)
Response DATA	{ABC} ASCII representation of hex value from A to D converter

2.7.22 Get Polarization

This command is only used for BUCs with an internal polarization switch. For devices without an internal polarization switch, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses.

NOTE: Polarization data is in upper case.

Table 2-33. Get Polarization

FIELD	DATA
Command Description	Get the Polarization Settings
CMD_CODE	22
Function Name	GPOLARIZATION
Command DATA	{}
Response DATA	ASCII {V,H,R,L}

2.7.23 Set Polarization

This command is only used for setting BUCs with an internal polarization switch. For devices without an internal polarization switch, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses.

> ***NOTE:*** Polarization data is in upper case.
>
> The {V, H, R, or L} in 'Response DATA' is optional, and describes whether the BUC has successfully changed to the new polarization specified in the 'Command DATA' field. The intention of this response is to indicate polarization status to the modem more quickly, and remove the need for subsequent query; the BUC may briefly delay its response until the polarization change is confirmed, but the delay should not interrupt the regular Client/Server communication flow described in Client/Server Interaction Sequence Diagram on page 12.

Table 2-34. Set Polarization

FIELD	DATA
Command Description	Set Polarization
CMD_CODE	23
Function Name	SPOLARIZATION
Command DATA	ASCII {V,H,R,L}
Response DATA	ASCII {V,H.R, or L} or {}

2.7.24 Request IFL Calibration Permission

In some Terminals an external controller may be acting as proxy to the BUC interface (i.e. KANDU in Aero Terminal, or Antenna controller in Maritime Terminal), that external controller may drive the IFL calibration process. This command (REQIFLCAL) and the subsequent updates (CALSTATUS) are intended for

In such Terminals, the modem requests permission to initiate IFL calibration (GETIFLCALPERMISSION), and follows with IFL calibration updates (CALSTATUS).

Table 2-35. Request IFL Calibration

FIELD	DATA
Command Description	Request to initate (IFL) Calibration
CMD_CODE	24
Function Name	REQIFLCAL
Command DATA	{status, cause}status {iflok,iflerror},cause=error string
Response DATA	{allowed/notallowed}

The command data has 2 fields:

The status field describes whether IFL calibration is needed. This field can have 2 possible values:

- iflok all validity checks regarding the IFL calibration were passed.
- iflerror problem with the IFL validity checks. Need to do IFL calibration.

The cause field describes the reason the Modem is asking to perform IFL calibration. This string is a free string intended for debug/logging purposes.

Some examples for this command:

- status = iflok, cause=na
- status = iflerror, cause=BUC Cal file is not valid
- status = iflerror, cause=IFL Cable loss parameter is zero
- status = iflerror, cause=Modem Output power calibration file is missing
- status = iflerror, cause=BUC gain file Heartbeat SN mismatch

The response data can have 2 possible values:

- allowed Modem is permitted to start the IFL calibration process.
- notallowed Modem is not permitted to start the IFL calibration process.

2.7.25 IF Calibration Status

The purpose of the command for the Modem to update the BUC about the Modem's IFL calibration process. The command will be sent when the calibration process has been authorized by the external controller (BUC proxy) and after modem has sent Request IFL Calibration Permission and received no errors in response.

Table 2-36. IF Calibration Status

FIELD	DATA
Command Description	Update on IFL calibraion process
CMD_CODE	25
Function Name	IFLCALSTATUS
Command DATA	{status, error, ifl1, ifl2, ifl3, ifl4, ifl5, percent}where, status=[0,1.or 2],error=error code (defined below),string,ifl=value1;value2;value3; value4;value5,percent=% of completion
Response DATA	{}

The status field describes the Modem IFL calibration status. This field can have 3 possible values:

0 – the IFL calibration process ended with error.

1 – the IFL calibration process ended successfully.

2 – the IFL calibration process is running.

The error field describes the reason the Modem failed the IFL calibration process. This string is a free string intended for debug/logging purposes. Examples for error strings:

- BUC Cal File No Reply on name message
- BUC Cal File Fail to download
- BUC Cal File Invalid format
- BUC Cal File Missing frequencies
- BUC Cal File Missing serial number
- BUC Cal File Heartbeat Serial number mismatch
- No reply on Heartbeat message

- No reply on ADC message
- Attenuation too low to satisfy maximum operating point
- Invalid cable loss values
- Unknown error

The ifl value fields will reflect the calculated IFL cable loss in dB when the calibration process ended successfully. While the modem is in calibration or in case the calibration process ended with error, the ifl values will be set to zero.

value1 – the cable loss calculated in frequency of 950MHz

value2 – the cable loss calculated in frequency of 1300MHz

value3 – the cable loss calculated in frequency of 1600MHz

value4 – the cable loss calculated in frequency of 1950MHz

value5 – the cable loss calculated in frequency of 1950MHz

The percent field describe the percentage completion of the IFL calibration process. The percent value is between 0 and 100, with 100 indicating successful completion. If the calibration process has failed, the value will reflect the percent completion before the error occurred.

During the IFL calibration the percent value will be incremented according to some internal software logic so the user will be able to verify that the process has not stalled.

Some examples for this command:

Calibration report command after the Modem finishes the calibration process successfully:

status=1,error=na,ifl1=15.3,ifl2=14.7,ifl3=13.9,ifl4=14.9,ifl5=15.0,percent=100

Calibration report command after the Modem failed to finish the calibration process:

status=0,error=BUC Cal File Invalid format,ifl1=0,ifl2=0,if3=0,ifl4=0,ifl5=0,percent=5

Calibration report command while the Modem is running the calibration process:

status=2,error=na,ifl1=0,ifl2=0,if3=0,ifl4=0,ifl5=0,percent=35

2.7.26 Extended Status

Query device for all status messages and configurations with a single command. This returns a string of key-value pairs primarily for logging/reporting, defined in detail in Extended Status on page 69.

Table 2-37. Extended Status

FIELD	DATA
Command Description	Return extended status message.
CMD_CODE	80 (0x50)
Function Name	GEXTSTATUS
Command DATA	{}
Response DATA	{}

2.7.27 Vendor Specific Commands

In order to allow different BUC vendors to provide the end user with a richer command set beyond those required by the monitoring and control interface, the protocol allows for the definition of vendor-specific commands.

All command codes 0x60 to 0x7E shall be reserved for use by the vendor. The Manufacturer_ID field is the binary representation of the MANUFACTURER_ID assigned to the vendor. This ID is as specified in Section 2.7.1. The vendor is free to use any internal format for the commands in the vendor specific portion after the first two fields, as shown in the command format. It is suggested that these vendor-specific commands be provided as ASCII strings.

Example Client Message:

CMD_CODE (0x60 to 0x7E)	MANUFACTURER_ID	CMD_LEN	Vendor-specific Command (MAX LEN = 255 Bytes) (Preferably an ASCII String)

Example Server Message (in response to Vendor Specific Client Message):

CMD_CODE (0xE0 to 0xFE)	MANUFACTURER_ID	CMD_LEN	Vendor-specific Command (MAX LEN = 255 Bytes) (Preferably an ASCII String)

2.8 Transceiver Command Syntax

While the goals of the OpenBMIP protocol are primarily the command and control of BUCs, increasing numbers of transceivers are being offered by vendors. For such devices, a limited set of additional OpenBMIP -format commands are defined below to allow for the transceiver to be controlled and managed via a single interface. For devices without receive (Rx) functionality, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses too all the commands below.

Table 2-38. Commands in Numerical Order

CMD_CODE	Section	Function Name
64 (0x40)	Get RX Local Oscillator (LO) on page 45	GRXLOSCILLATOR
65 (0x41)	Set RX Local Oscillator (LO) on page 46	SRXLOSCILLATOR
66 (0x42)	Get RX Polarization on page 46	GRXPOLARISATION
67 (0x43)	Set RX Polarization on page 47	SRXPOLARISATION
68 (0x44)	Get RX State on page 47	GRXSTATE
69 (0x45)	Get RX Fault Status on page 48	GRXFAULTSTATUS
70 (0x46)	Get RX Latched Faults on page 48	GRXSTATUSLATCH
71 (0x47)	Reset RX Latched Faults on page 49	RRXSTATUSLATCH
72 (0x48)	Get Power Amplifier Forward Power on page 49	GRXPAFWDP
73 (0x49)	Get Power Amplifier Input Power on page 50	GRXPAINPUTPOWER

2.8.1 Get RX Local Oscillator (LO)

Query the device for the currently configured receiver Local Oscillator (LO) frequency in GHz. For devices using OpenAMIP to select the LO frequency, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses.

Table 2-39. Get RX Local Oscillator (LO)

FIELD	DATA
Command Description	Get the configured RX LO frequency (GHz)
CMD_CODE	64 (0x40)
Function Name	GRXLOSCILLATOR
Command DATA	{}
Response DATA	{18.25GHz}

2.8.2 Set RX Local Oscillator (LO)

Configure the receiver Local Oscillator (LO) frequency in GHz.

Table 2-40. Set RX Local Oscillator (LO)

FIELD	DATA
Command Description	Set the configured RX LO frequency (GHz)
CMD_CODE	65 (0x41)
Function Name	SRXLOSCILLATOR
Command DATA	{18.25GHz}
Response DATA	{lock/unlock} or {}

2.8.3 Get RX Polarization

Get the currently configured Rx polarization. For single-band devices or those which use OpenAMIP to select the LO frequency, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses.

Table 2-41. Get RX Polarization

FIELD	DATA
Command Description	Get the configured RX Polarization
CMD_CODE	66 (0x42)
Function Name	GRXPOLARIZATION
Command DATA	{}
Response DATA	ASCII {V, H, R, L}

2.8.4 Set RX Polarization

This command is only used for setting BUCs with an internal Rx polarization switch. For devices without an internal Rx polarization switch, "Unimplemented Command" (0x1E) or "Unrecognized Command" (0x1F) are acceptable responses.

Table 2-42. Set RX Polarization

FIELD	DATA
Command Description	Get the configured RX Polarization
CMD_CODE	67 (0x43)
Function Name	SRXPOLARIZATION
Command DATA	ASCII {V,H,R,L}
Response DATA	ASCII {V,H,R,L} or {}

2.8.5 Get RX State

This command is used to query the current state of the transceiver's receiver.

Table 2-43. Get RX State

FIELD	DATA
Command Description	Get the state of the RX
CMD_CODE	68 (0x44)
Function Name	GRXSTATE
Command DATA	{}
Response DATA	{ok} or {fault}

2.8.6 Get RX Fault Status

This command queries the device for any currently faults/status for the active Rx chain.

Table 2-44. Get RX Fault Status

FIELD	DATA
Command Description	Get currently active Rx faults
CMD_CODE	69 (0x45}
Function Name	GRXFAULTSTATUS
Command DATA	{}
Response DATA	{fault1,fault2,...fault_n}Fault string options include,ok, externalRefernceError,voltageFault,overCurrentFault,overTempertureFault,noConnection Custom fault strings may be added if not in the list above.

2.8.7 Get RX Latched Faults

This command queries the device for the current forward power in dBm of the power amplifier (if supported).

Table 2-45. Get RX Latched Faults

FIELD	DATA
Command Description	Get currently active Rx faults
CMD_ CODE	70 (0x46)
Function Name	GLRXFAULTSTATUS
Command DATA	{}
Response DATA	{fault1,fault2,...,fault_n} Fault string options include,ok,externalReferenceError,voltageFault,toneFault,overCurrentFault,overTemperatureFault,noConnection custom fault strings may be added if not in the list above.

2.8.8 Reset RX Latched Faults

Reset latched receiver chain faults.

Table 2-46. Reset RX Latched Faults

FIELD	DATA
Command Description	Clear all lached receiver/LNB faults
CMD_CODE	71 (0x47)
Function Name	RRXSTATUSLATCHED
Command DATA	{}
Repsonse DATA	{}

2.8.9 Get Power Amplifier Forward Power

This command queries the device for the current forward power in dBm of the power amplifier (if supported).

Table 2-47. Get Power Amplifier Input Power

FIELD	DATA
Command Description	Return power amplifier Forward power in dBmResponse format Float: (nn.nn) 00.00 to +99.99
CMD_CODE	72 (0x48)
Function Name	GPAFWDPOWER
Command DATA	{}
Response DATA	{43.50dBm}

2.8.10 Get Power Amplifier Input Power

This command queries the device for the current input power in dBm of the power amplifier (if supported).

Table 2-48. Get Power Ampliffier Input Power

FIELD	DATA
Command Description	Return power amplifier Input power in dBmRespnse format Float: (+/-nn.nn) -99.99 to +99.99
CMD_CODE	73 (0x49)
Function Name	GPAPOWERINPUT
Command DATA	{}
Response DATA	{-12.50dBm}

2.9 File Transfers

2.9.1 File Transfer Types

There are two file types to be transferred: calibration files in XML format (see Factory Calibration Format on page 60) and software upgrade files in vendor-defined format. A file transfer is always initiated by the modem, using GCALIBFILE (see File Transfers on page 51), SCALIBFILE (see File Transfers on page 51), or PREP_UPGRADE (see File Transfers on page 51). Next, the file transfer is performed (see File Transfers on page 51).

File names consist only of the characters allowed in the DATA field (see DATA on page 10), excluding comma. Filenames do not include any path name. Filenames are case-insensitive.

2.9.2 File Transfer Methods

Implementations may support either or both of Kermit and TFTP for transfer of calibration file from BUC to the modem.

Kermit

File Transfer uses Kermit, a combined network and serial communication software protocol offering a consistent, transport-independent, cross-platform approach to connection establishment, terminal sessions, file transfer, file management, character-set translation, numeric and alphanumeric paging, and automation of file transfer and management, dialogs, and communication tasks through its built-in scripting language.

Not all implementations of Kermit in the field behave identically. To ensure compatibility with iDirect terminals, iDirect has produced an application to test the OpenBMIP and Kermit performance of a BUC. This application functions correctly with basic Kermit implementations as well as the more complex additions to the original design. The BUC Test application is available from iDirect along with its source code. A BUC simulator is also available along with its source code.

TFTP

File Transfer may use TFTP instead of (or in addition to) Kermit. TFTP version supported should cover RFC 1350, plus subsequent amendments and updates through RFC 7440.

2.9.3 File Transfer Sequence, Modem to BUC

```
modem -------------- [SENDFILE] ----------->BUC

modem <-------------- [SENDFILE, 1] -----------BUC

modem -------------- [Kermit -s powercal.xml or TFTP put powercal.xml] ----------->BUC

modem -------------- [STATUSFILETRANSFER] ----------->BUC

modem <-------------- [STATUSFILETRANSFER, 1] -----------BUC
```

2.9.4 File Transfer Sequence, BUC to Modem

```
modem -------------- [RECEIVEFILE] ----------->BUC

modem <-------------- [RECEIVEFILE, 1] -----------BUC

modem -------------- [Kermit -r powercal.xml or TFTP get powercal.xml] ----------->BUC

modem -------------- [STATUSFILETRANSFER] ----------->BUC

modem <-------------- [STATUSFILETRANSFER, 1] -----------BUC
```

2.9.5 Calibration File Procedure

During commissioning of a terminal, the modem requires the BUC calibration file and it must be transferred from the BUC to the modem. The OpenBMIP protocol allows for transfer of this file over Kermit or TFTP. The modem software must take measures to ensure that the calibration file is usable, e.g. a calibration file with missing mandatory or invalid parameters may not be usable. A successful Kermit or TFTP transfer is necessary but not sufficient. The modem software must ensure that the payload of the Kermit or TFTP transfer is valid for the model of modem and that using the file will allow the modem to start in a valid operating mode.

The protocol supports retrieving the BUC's calibration file from the BUC to the modem. In order to retrieve the calibration file, the following procedure is performed:

- Modem sends the GCALIBFILE command to the BUC.
- BUC responds with file name and size in bytes
- Modem sends RECEIVEFILE command with the file name and size
- BUC responds success/failure of the command
- BUC starts a file transfer of the file with the name specified in the previous command. Protocol includes indication of completion of file transfer.
- Modem sends STATUSFILETRANSFER to BUC.
- BUC responds with response code that indicates success or failure of the transfer:
- On successful transfer, the modem will continue its normal operation

2.9.6 Software Upgrade Procedure

Both upgrades and downgrades must be supported. The BUC software must take measures to ensure that an invalid software image may not be used for a software upgrade. A successful Kermit or TFTP transfer is necessary but not sufficient. The BUC software must ensure that the payload of the Kermit or TFTP transfer is valid for the model of BUC being upgraded and that using that image will allow the BUC to start in a valid operating mode.

The serial protocol supports upgrading the software on the BUC. In order to perform a software upgrade of the BUC, the following procedure is performed:

- CM sends the PREP_UPGRADE command that specifies the filename used for upgrade and also its size in bytes.
- The BUC responds (see RSP_CODE on page 8) with RSP_CODE = 1 if it is ready to accept the upgrade; 0 if not ready.
- If a RSP_CODE = 0 is received, the satellite modem retries the upgrade after a configurable timeout (default 30s).
- If a RSP_CODE = 1 is received, the satellite modem starts a file transfer of the file with the name specified in the previous command.
- If the file transfer fails, the CM retries the file transfer after a configurable timeout (default 30s).
- If the file transfer succeeds, the satellite modem resets the BUC using the BUC Reset command.
- The BUC is expected to boot with the new software version after the reset.
- The satellite modem will restart communications with the BUC after a configurable timeout (default 60s).

3 Calibration

This chapter contains the following sections:

3.1 Factory Calibration Algorithm

The calibration procedure as indicated below should be run by the BUC manufacturer. The BUC output power is factory-calibrated with IF stimulus at several temperature points, for example: minimum operating temperature, maximum operating temperature, and nominal or average operating temperature. For example, if the MOP_STEP_SIZE is 4 dB, the BUC would be calibrated at Maximum Output Power (MOP), [MOP-4dB], and [MOP-8dB] for better linearity interpolation by the modem. For maximum interpolation accuracy, the raw hex bits power reading from the ADC should be linear to BUC output power expressed in dBm. The resultant calibration data is stored so the modem can electronically retrieve it via OpenBMIP query.

> ***NOTE:*** If the BUC supports an IF lowpass harmonic filter, the filter should be configured to its lowpass setting for all frequencies less than or equal to the transition frequency. The filter should be configured to its high-band (or wide-band) setting for all frequencies greater than the transition frequency. This will ensure consistent operation with the modem.

The BUC Calibration process is shown in **Figure 3-1**. Measurements are made at each point identified by target power level "pow", modulation type "m", frequency "f", polarization “pol”, local oscillator “l” and temperature "t". At each measurement point, power is externally measured at the IF input, and at the RF output. Additionally, the BUC's internal detector analog to digital converter (ADC) is read with the BUC power amp enabled, and with the BUC power amp disabled. Finally, the BUC's internal temperature sensor (which should be thermally close to the power amp and power detector) is read. It is desirable to read the power detector and the temperature sensor as rapidly as possible (allowing for detector settling time) to minimize temperature changes between the associated readings.

The measurement temperatures should be selected to include any peaks or troughs in the gain versus temperature curve, to minimize error in interpolation. In all cases, the endpoints (minimum and maximum operating temperatures) should be included, but data will be linearly extrapolated if temperatures beyond those in the CAL file are reported by the BUC.

The following cases are anticipated:

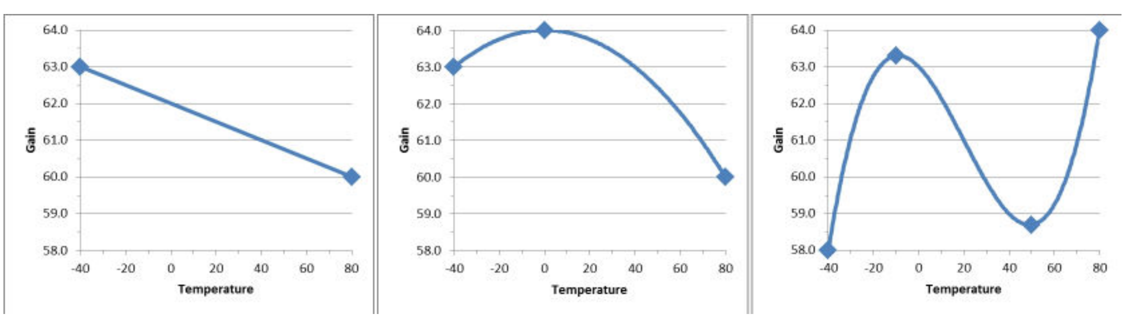


Figure 3-1. Typical Gain vs Temperature for 2, 3, and 4 Points

NOTE: The temperatures t=1, t=2, and t=3 indicated in Figure 3-2 are for reference; systems may calibrate at the temperatures appropriate for the application. The nominal Maximum Output Power (MOP) indicated in the figure is for reference; systems may implement the MOP appropriate to the application. Other details, such as the frequency calibration points chosen, may be tailored for the application. The file format described in Factory Calibration Format on page 60 must be used in all implementations.

3.2 Multiple Local Oscillator Values

If the BUC supports multiple local oscillator ("loscillator" or LO) settings, each possible setting will be associated with one block of data (a "loscillator_band") in the calibration file. The file should include frequencies stepped across the entirety of each loscillator band.

The format is intended to be partially back-compatible with OpenBMIP versions up to 9.5. Remotes supporting up to v9.5 will ignore the loscillator_bands and assume all content applies to a single band. Remotes supporting v10.0 or later will use loscillator_bands if present, but will default to a configured (in the remote) single LO value if loscillator_band XML tags are absent. As in previous versions of OpenBMIP, all frequencies represented IF, except for loscillator frequencies.

If loscillator_band XML tags are present, the remote must always send the SLOSCILLATOR command when configuring the terminal to operate in a given band, or to be calibrated in a given band.

The flowchart below shows stepping through the loscillator_bands, although manufacturers are free to adopt their own process. It is assumed that the manufacturer will stabilize the test chamber at one temperature, and step through all bands and frequencies before changing the temperature and repeating all bands and frequencies.

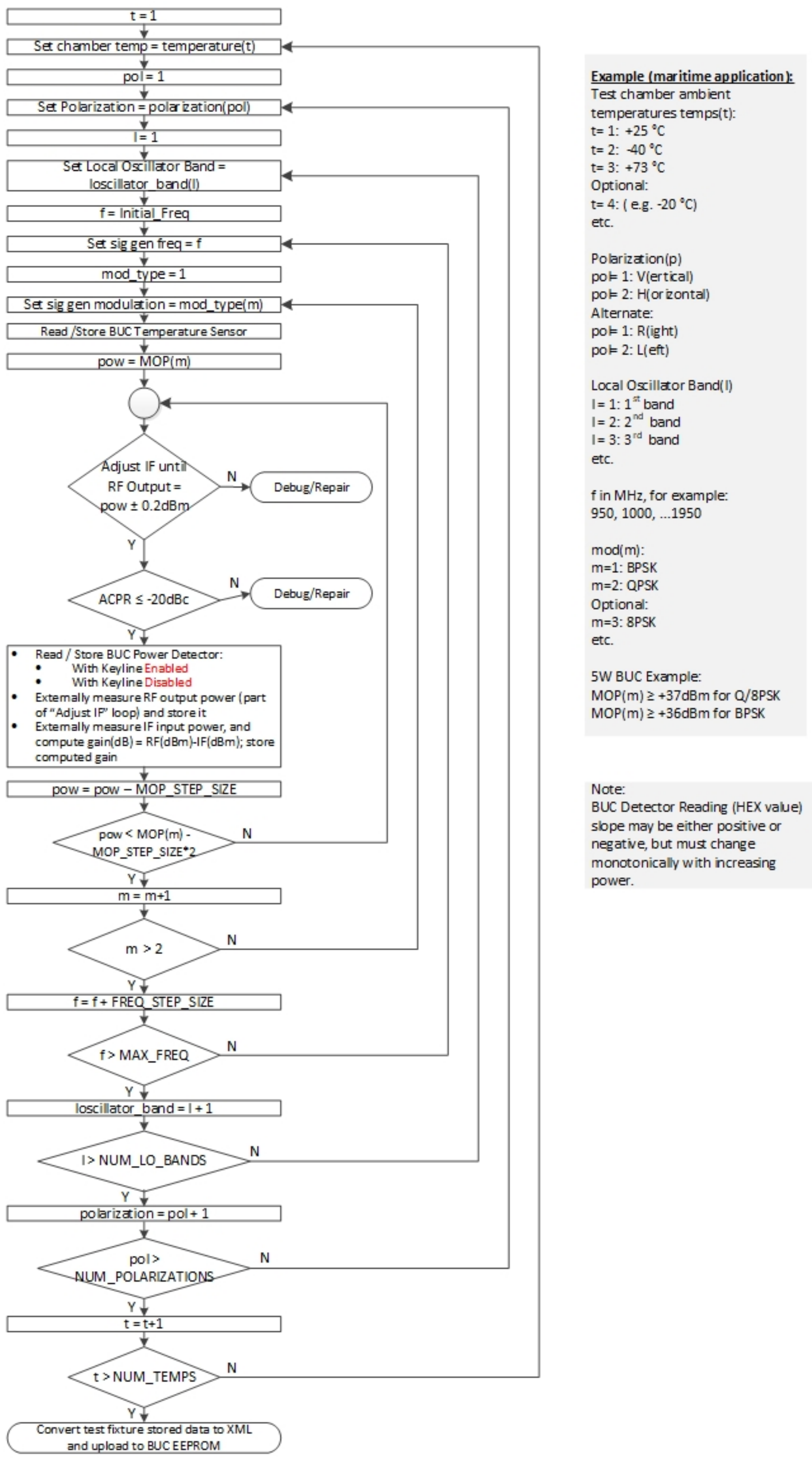


Figure 3-2. Sample BUC Factory Calibration Process

3.3 Factory Calibration Format

The calibration file that will be retrieved from the BUC by the CM should bind to the following xml file format. The BUC manufacturer may optionally choose to compress the calibration file in GZIP format (see IETF RFC 1952, listed in the Standards References). The filename shall end with ".xml" if it is XML and ".gz" if it is a GZIP-compressed file containing the XML file.

In the sample calibration file shown below, only the characters enclosed within the demarcation characters need to be presented as-is. The demarcation characters are "<>" and "</>" respectively. The other information that is presented in this sample calibration file format are all "values" and not string constants. These need to be replaced with the actual measured values of that particular variable. The "values" are indicated as SOME_VARIABLE_NAME in the following format. These can be decimals, floats, or strings as required.

For <buc_cal_file> element information, see below:

Table 3-1. Factory Calibration File: Element <buc_cal_file>

Element: <buc_cal_file>	Format	Marks beginning and end of file content
Name of Value		Example Value Units
<cal_format_version>	11.0	Same as version of this document. If not present, version is assumed to be compatible with B (9.5). Cal Format Version 10.1 adds multiband capability.
<vendor_id>	99	BUC Manufacturer ID (MID) unique in network. UC Manufacturer ID (MID) unique in network. Assigned MID values are indicated in Section Appendix A.
<model_number>	1234567890xxx	BUC Part Number (BPN)
<serial_number>	A12345A33	BUC Serial Number (BSN)
<functional_id>	050300	BUC Functional ID (FID)
<num_polarizations>	2	Number of <polarization_setting> Elements below. Each <polarization_setting> Element identifies the <polarization> for which subsequent calibration data applies. Note that a set of calibration values may apply to multiple polarizations: (i.e <polarization>RL</polarization>). If this is the case, they are contained in a single <polarization_setting>, and <num_polarizations>=1
<num_loscillator_bands>	4	Number of local oscillator bands, maximum 8

Table 3-1. Factory Calibration File: Element <buc_cal_file> (Cont.)

<pow_det_ settling_time>	5	milliseconds (ms), power detector settling time
<polarization_ setting>	Element	Marks beginning and end of data for one polarization setting

For <polarization_setting> element information, see below:

Table 3-2. Factory Calibration File: Element <polarization_setting>

Element: <polarization_ setting>	Format	Marks beginning and end of data for one polarization settings
Name of Value		Example Value Units
<polarization>	L	Polarization(s). For BUCs with no ability to set polarization, <num_polar-izations>=1, and this <polarization> parameter may be omitted. For BUCs with the ability to set polarization, <num_polar-izations>=1 or more, and <polarization>={L, R, H and/or V}, describing the polarizations for which calibration data in this <polarization_setting> applies.
<loscillator_band>	Element	Marks the beginning and end of data for one local oscillator setting.

For <loscillator_band> element information, see below:

Table 3-3. Factory Calibration File: Element <loscillator_band>

Element: <loscillator_ band>	Format	Example Value Units
Name of Value		
<loscillator>	28.05	GHz, local oscillator frequency
<num_symbol_rates>	1	Num of symbol rates (<symbol_rate_setting> Elements) for which calibration data are available.
<symbol_rate_setting>	Element	Marks the beginning and end of data for one symbol rate.

For <symbol_rate_setting> element information, see below:

Table 3-4. Factory Calibration File: Element <symbol_rate_setting>

Element: <symbol_rate_ setting> Name of Value	Format	Format Example Value Units
<symbol_rate>	1	Symbol rate (MSps) that was used during BUC factor calibration for calibration data in this <symbol_rate_setting> Element.
<max_operating_point>	33.0	dBm (for QPSK; nominal)
<back_off_step_size>	4	dB; difference between power steps
<initial_freq>	950	MHz
<freq_step_size>	50	NOTE: freq_step_size is deprecated; the modem will ignore this value. Because each frequency is explicitly listed the file is permitted to contain non-uniform steps. One typical use for this feature is to add finer steps to accommodate discontinu- ities in the frequency response, such as at the band select filter breakpoint.
<max_freq>	1950	MHz
<buc_cutoff_frequency_ mhz>	1350.0	MHz; Decision point for the BUC's internal lowpass harmonic filter. When transmitting an IF frequency less than or equal to this value, the remote will set the filter to lowband; frequency greater than this value will set the BUC's filter to highband. If this value is absent, the remote will assume:mean(max_freq, min_freq) – 100MHzIf the BUC does not have a switchable filter, this value can be set to equal <max_freq>.
<adc_res>	12	Bits; 12 bits is currently supported
<adc_slope>	-1	-1 or +1. “+1” means that a higher power output will produce a higher ADC number. Slope may be mildly nonlinear but must be monotonic, over the calibrated range (MOP to MOP-x, where MOP-x represents the lowest power level in the BUC calibration file), plus a margin of +0.75/-2.0dB.
<num_temps>	3	Number of temperature points, maximum 5.

Table 3-4. Factory Calibration File: Element <symbol_rate_setting> (Cont.)

<band_cal_flag>	1	Flag indicating that this band is preferred for IFL calibration. Any value other than 1, or if the band_cal_ flag is absent, indicates that the band is not preferred for calibration.
<temperature*n*>	Element	Marks the beginning and end of data for one temperature measurement

For <temperature*n*> element information, see below:

Table 3-5. Factory Calibration File: Element <temperature*n*>

Element: <temperature*n*>	Format	Marks beginning and end of data for one set of temperature measurements
Name of Value		Example Value Units
<modulation_setting>	Element	Marks the beginning and end of data for one set of measurements at a given modulation

For <modulation_setting> element information, see below:

Table 3-6. Factory Calibration File: Element <modulation_setting>

Element: <modulation_ setting>	Format	Marks beginning and end of data for one set of measurements at a given modulation
Name of Value		Example Value Units
<mod_type>	1	1 (BPSK), 2 (QPSK), 3 (8PSK), 4(16PSK), 5(32PSK), 6 (16QAM), 7(32QAM); 8(32APSK), 9(64QAM), 10 (64APSK), 11(128APSK), 12(128QAM), 13(256APSK), 14 (256QAM). 1 and 2 must be supported.
<pow_set>	Element	Marks beginning and end of data for one frequency. (All input power levels for one frequency)

For <pow_set> element information, see below:

Table 3-7. Factory Calibration File: Element <pow_set>

Element: <pow_set>	Format	Marks beginning and end of data for one frequency. (all input power levels for one frequency)
Name of Value		Example Value Units
<freq>	950	MHz, IF input
<pout>	33.0	dBm, externally measured output **NOTE:** <pout>, <pout_det>, <pout_det_dis> and <gain> form a set of parameters that are repeated within the <pow_set> Element for each unique <pout> calibration point. Three <pout> calibration points are required: at the <maximum_operating_point> (MOP) defined in the <symbol_rate_settings> Element, and two additional calibration points at (MOP-MOP_STEP_SIZE) and (MOP-2*MOP_STEP_SIZE). Provision of additional calibration points should be coordinated between the manufacturers.
<pout_det>	7F5	Raw hex bits from ADC, PA enabled
<pout_det_dis>	82E	Raw hex bits from ADC, PA disabled
<gain>	62.4	dB, Externally measured gain
<buc_temp>	57.6	Degrees C near PA & detector

```xml
<buc_cal_file>
    <cal_format_version>11.0</cal_format_version>
    <vendor_id>MID</vendor_id>
    <model_number>BPN</model_number>
    <serial_number>BSN</serial_number>
    <functional_id>FID</functional_id>
    <num_polarizations>NUM_POLARIZATIONS</num_polarizations>
    <num_loscillator_bands>NUM_LO_BANDS</num_loscillator_bands>
    <pow_det_settling_time>POW_DET_SETTLING_TIME</pow_det_settling_time>
    <polarization_setting>
        <polarization>POLARIZATION(L, R, H, and/or V)</polarization>
        <loscillator_band>
            <loscillator>LO_FREQUENCY_1</loscillator>
            <num_symbol_rates>NUM_SYM_RATES</num_symbol_rates>
            <symbol_rate_setting>
                <symbol_rate>SYMBOL_RATE</symbol_rate>
                <max_operating_point>MOP</max_operating_point>
                <back_off_step_size>MOP_STEP_SIZE</back_off_step_size>
                <initial_freq>INI_FREQ</initial_freq>
                <freq_step_size>FREQ_STEP_SIZE</freq_step_size>
                <max_freq>MAX_FREQ</max_freq>
                <buc_cutoff_frequency_mhz>CUTOFF</buc_cutoff_frequency_mhz>
                <adc_res>ADC_RES_IN_BITS</adc_res>
                <adc_slope>ADC_SLOPE</adc_slope>
                <num_temps>NUM_TEMPS</num_temps>
                <band_cal_flag>BAND_CAL_FLAG</band_cal_flag>
                <temperature1>
                    <modulation_setting>
                        <mod_type>1</mod_type>
                        <pow_set>
                            <freq>INITIAL_FREQ</freq>
                            <pout>MOP</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE*2</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <buc_temp>BUC_TEMPERATURE</buc_temp>
                        </pow_set>
                    </modulation_setting>
                </temperature1>
            </symbol_rate_setting>
        </loscillator_band>
    </polarization_setting>
</buc_cal_file>
```

Appendix A: BUC Manufacturer IDs

BUC Manufacturer IDs are assigned by iDirect.

Manufacturers intending to produce a product supporting the BMIP protocol must contact iDirect to be assigned a MID.

Table A-1. BUC Manufacturer IDs

Manufacturer	MID
Honeywell	03
Astronics	04
NJRC	10
CPI	11
ViaSat	12
Actox	13
Xicom	14
Teledyne	15
Mission Microwave	16
Kymeta	17
Norsat	18
Tampa Microwave	19
Agilis	20
Acorde	21
RevGo	22
Intellian	23

Table A-1. BUC Manufacturer IDs (Cont.)

Satcom Direct	24
Actia	25
Ball Aerospace	26
MTI (Microelectronics Technology Inc.)	27
Honeywell	30
Skyware	40
Wavestream	50
L3-GCS	60
EMS	70
Rugged Logic	80
SAGE	90
iDirect	99

NOTE: New vendors wishing to be assigned a Manufacturer ID (MID) should contact iDirect TAC (tac@idirect.net) for assistance.

Appendix B: Extended Status

Table B-1. Extended Status

Parameter (Key)	Description	Data Type
txState	BUC / Transmitter status This should include any faults or conditions that are present with the BUC	llist {string} ({status-1,status-2, … status-n}) Defined options: {ok, transmitterOn, externalMute, mute, externalReferenceFault, highForwardPowerFault, lowForwardPowerFault, internalFault, overTemperatureFault, fanFault, noConnection} or custom string

Table B-1. Extended Status (Cont.)

bucLatchedFault	**BUC / Transmitter Latched Fault** **Latched faults within the BUC**	**list {string}** **({fault-1,fault-2, … fault-n})** **Defined options:** **{externalReferenceFault,** **highForwardPowerFault,** **lowForwardPowerFault,** **internalFault,** **overTemperatureFault,** **fanFault,** **noConnection}** **or custom string**
bucFwdPower	**BUC / Transmitter forward power**	**dBm (float, nn.nn)** **00.00 to 99.99**
bucInputPower	**BUC / Transmitter input power (if supported)**	**dBm (float, +/-nn.nn)** **-99.99 to +99.99**
bucAtt	**BUC / Transmitter atten-uation setting**	**dB (float, nn.nn)** **00.00 to 99.99**
bucTemp	**BUC / Transceiver Tem-perature**	**degrees Celcius (float, +/-nnn.nn)** **-999.99 to 999.99**
bucFanSpeed	**BUC / Transceiver Fan Speed**	**Revolutions per minute (integer nnnnnn)** **0 to 999999**

Table B-1. Extended Status (Cont.)

lnbState	**This should include any faults or conditions that are present with the LNB**	**list {string}** **{status-1,status-2, … status-n}** **Defined options:** **{ok,** **externalReferenceError,** **voltageFault,** **toneFault,** **overCurrentFault,** **internalFault,** **overTemperatureFault,** **noConnection}** **or custom string**
lnbLatchedfault	**Latched faults within the LNB**	**list {string}** **{status-1,status-2, … status-n}** **Defined options:** **{externalReferenceError,** **voltageFault,** **toneFault,** **overCurrentFault,** **internalFault,** **overTemperatureFault,** **noConnection}** **or custom string**
lnbLOSetting	**LNB Local Oscillator Setting** **Receive Local Oscillator frequency configuration**	**MHz (integer, nnnnn)** **00000 to 99999**

Table B-1. Extended Status (Cont.)

bucLOSetting	**BUC Local Oscillator Setting** **Transmit Local Oscillator frequency configuration**	**MHz (integer, nnnnn)** **00000 to 99999**
bucMan	**Transceiver Manufacturer** **The BUC manufacturer**	**string** **custom string**
bucPN	**BUC/Transceiver Part Number** **The manufacturer part number**	**string** **custom string**
bucSN	**BUC/Transceiver Serial Number** **The manufacturers SN for the BUC**	**string** **custom string**
bucID	**BUC/Transceiver ID** **ID / hostname assigned to the BUC**	**string** **custom string**
bucIFFilter	**BUC IF Lowpass Harmonic filter setting** **get the BUC IF lowpass harmonic filter setting**	**string** **Defined options:** **{lowband,** **gpio_0,** **serial_0,** **tone_1}**

Appendix C: BUC Calibration (Sample)

A generic sample BUC Calibration file is shown on the next few pages.

> ***NOTE:*** The Vendor_ID, MODEL_NUMBER, SERIAL NUMBER, FUNCT_ID definition/format can be found in BUC Command Syntax on page 19. All items in UPPER CASE will be replaced with specific values.

```xml
<buc_cal_file>
    <cal_format_version>11.0 </cal_format_version>
    <vendor_id>MID</vendor_id>
    <model_number>BPN</model_number>
    <serial_number>BSN</serial_number>
    <functional_id>FID</functional_id>
    <num_polarizations>NUM_POLARIZATIONS</num_polarizations>
    <num_loscillator_bands>NUM_LO_BANDS</num_loscillator_bands>
    <pow_det_settling_time>POW_DET_SETTLING_TIME</pow_det_settling_time>
    <polarization_setting>
        <polarization>POLARIZATION(L,R,H and/or V)</polarization>
        <loscillator_band>
            <loscillator>LO_FREQUENCY_1</loscillator>
            <num_symbol_rates>NUM_SYM_RATES</num_sym_rates>
            <symbol_rate_setting>
                <symbol_rate>SYMBOL_RATE</symbol_rate>
                <max_operating_point>MOP</max_operating_point>
                <back_off_step_size>MOP_STEP_SIZE</back_off_step_size>
                <initial_freq>INI_FREQ</initial_freq>
                <freq_step_size>FREQ_STEP_SIZE</freq_step_size>
                <max_freq>MAX_FREQ</max_freq>
                <buc_cutoff_frequency_mhz>CUTOFF</buc_cutoff_frequency_mhz>
                <adc_res>ADC_RES_IN_BITS</adc_res>
                <adc_slope>ADC_SLOPE</adc_slope>
                <num_temps>NUM_TEMPS</num_temps>
                <band_cal_flag>BAND_CAL_FLAG</band_cal_flag>
                <temperature1>
                    <modulation_setting>
                        <mod_type>1</mod_type>
                        <pow_set>
                            <freq>INITIAL_FREQ</freq>
                            <pout>MOP</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE*2</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <buc_temp>BUC_TEMPERATURE</buc_temp>
                        </pow_set>
                        <pow_set>
                            <freq>INITIAL_FREQ+FREQ_STEP_SIZE</freq>
                            <pout>MOP</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
```

```
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE*2</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <buc_temp>BUC_TEMPERATURE</buc_temp>
            </pow_set>
            <pow_set>
                <freq>MAX_FREQ</freq>
                <pout>MOP</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE*2</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <buc_temp>BUC_TEMPERATURE</buc_temp>
            </pow_set>
        </modulation_setting>
        <modulation_setting>
            <mod_type>2</mod_type>
            <pow_set>
                <freq>INITIAL_FREQ</freq>
                <pout>MOP</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE*2</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <buc_temp>BUC_TEMPERATURE</buc_temp>
            </pow_set>
            <pow_set>
                <freq>INITIAL_FREQ+ FREQ_STEP_SIZE</freq>
                <pout>MOP</pout>
                <pout_det>DET_ADC_HEX</pout_det>
```

```
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <pout>MOP-MOP_STEP_SIZE</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <pout>MOP-MOP_STEP_SIZE*2</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <buc_temp>BUC_TEMPERATURE</buc_temp>
        </pow_set>
        <pow_set>
            <freq>INITIAL_FREQ+ FREQ_STEP_SIZE*2</freq>
            <pout>MOP</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <pout>MOP-MOP_STEP_SIZE</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <pout>MOP-MOP_STEP_SIZE*2</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <buc_temp>BUC_TEMPERATURE</buc_temp>
        </pow_set>
        <pow_set>
            <freq>INITIAL_FREQ+FREQ_STEP_SIZE*19</freq>
            <pout>MOP</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <pout>MOP-MOP_STEP_SIZE</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <pout>MOP-MOP_STEP_SIZE*2</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
            <buc_temp>BUC_TEMPERATURE</buc_temp>
        </pow_set>
        <pow_set>
            <freq>MAX_FREQ</freq>
            <pout>MOP</pout>
            <pout_det>DET_ADC_HEX</pout_det>
            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
            <gain>GAIN_IN_DB</gain>
```

```
                <pout>MOP-MOP_STEP_SIZE</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE*2</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <buc_temp>BUC_TEMPERATURE</buc_temp>
            </pow_set>
        </modulation>
    </temperature1>
    <temperature2>
        <modulation_setting>
            <mod_type>1</mod_type>
            <pow_set>
                <freq>INITIAL_FREQ</freq>
                <pout>MOP</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE*2</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <buc_temp>BUC_TEMPERATURE</buc_temp>
            </pow_set>
            <pow_set>
                <freq>MAX_FREQ</freq>
                <pout>MOP</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <pout>MOP-MOP_STEP_SIZE*2</pout>
                <pout_det>DET_ADC_HEX</pout_det>
                <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                <gain>GAIN_IN_DB</gain>
                <buc_temp>BUC_TEMPERATURE</buc_temp>
            </pow_set>
        </modulation_setting>
        <modulation_setting>
            <mod_type>2</mod_type>
```

```
                        <pow_set>
                            <freq>INITIAL_FREQ </freq>
                            <pout>MOP</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE*2</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <buc_temp>BUC_TEMPERATURE</buc_temp>
                        </pow_set>
                        <pow_set>
                            <freq>MAX_FREQ</freq>
                            <pout>MOP</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <pout>MOP-MOP_STEP_SIZE*2</pout>
                            <pout_det>DET_ADC_HEX</pout_det>
                            <pout_det_dis>DET_ADC_HEX</pout_det_dis>
                            <gain>GAIN_IN_DB</gain>
                            <buc_temp>BUC_TEMPERATURE</buc_temp>
                        </pow_set>
                    </modulation_setting>
                </temperature2>
            </symbol_rate_setting>
        </loscillator_band>
    </polarization_setting>
</buc_cal_file>
```

Appendix D: BUC Wattage Numbering

BUC Wattage Numbering table:

Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code
1W	01	17W	17	33W	33	49W	49	65W	65	81W	81	97W	97	113W	AD	129W	BE	145W	CE
2W	02	18W	18	34W	34	50W	50	66W	66	82W	82	98W	98	114W	AE	130W	BF	146W	CF
3W	03	19W	19	35W	35	51W	51	67W	67	83W	83	99W	99	115W	AF	131W	C0	147W	D0
4W	04	20W	20	36W	36	52W	52	68W	68	84W	84	100W	A0	116W	B1	132W	C1	148W	D1
5W	05	21W	21	37W	37	53W	53	69W	69	85W	85	101W	A1	117W	B2	133W	C2	149W	D2
6W	06	22W	22	38W	38	54W	54	70W	70	86W	86	102W	A2	118W	B3	134W	C3	150W	D3
7W	07	23W	23	39W	39	55W	55	71W	71	87W	87	103W	A3	119W	B4	135W	C4	151W	D4
8W	08	24W	24	40W	40	56W	56	72W	72	88W	88	104W	A4	120W	B5	136W	C5	152W	D5
9W	09	25W	25	41W	41	57W	57	73W	73	89W	89	105W	A5	121W	B6	137W	C6	153W	D6
10W	10	26W	26	42W	42	58W	58	74W	74	90W	90	106W	A6	122W	B7	138W	C7	154W	D7
11W	11	27W	27	43W	43	59W	59	75W	75	91W	91	107W	A7	123W	B8	139W	C8	155W	D8
12W	12	28W	28	44W	44	60W	60	76W	76	92W	92	108W	A8	124W	B9	140W	C9	156W	D9
13W	13	29W	29	45W	45	61W	61	77W	77	93W	93	109W	A9	125W	BA	141W	CA	157W	DA
14W	14	30W	30	46W	46	62W	62	78W	78	94W	94	110W	AA	126W	BB	142W	CB	158W	DB
15W	15	31W	31	47W	47	63W	63	79W	79	95W	95	111W	AB	127W	BC	143W	CC	159W	DC
16W	16	32W	32	48W	48	64W	64	80W	80	96W	96	112W	AC	128W	BD	144W	CD	160W	DD

Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code	Power	Code
0.5W	H0	16.5W	I0	32.5W	J0	48.5W	K0	64.5W	L0	80.5W	M0	96.5W	N0	112.5W	O0	128.5W	P0	144.5W	Q0
1.5W	H1	17.5W	I1	33.5W	J1	49.5W	K1	65.5W	L1	81.5W	M1	97.5W	N1	113.5W	O1	129.5W	P1	145.5W	Q1
2.5W	H2	18.5W	I2	34.5W	J2	50.5W	K2	66.5W	L2	82.5W	M2	98.5W	N2	114.5W	O2	130.5W	P2	146.5W	Q2
3.5W	H3	19.5W	I3	35.5W	J3	51.5W	K3	67.5W	L3	83.5W	M3	99.5W	N3	115.5W	O3	131.5W	P3	147.5W	Q3
4.5W	H4	20.5W	I4	36.5W	J4	52.5W	K4	68.5W	L4	84.5W	M4	100.5W	N4	116.5W	O4	132.5W	P4	148.5W	Q4
5.5W	H5	21.5W	I5	37.5W	J5	53.5W	K5	69.5W	L5	85.5W	M5	101.5W	N5	117.5W	O5	133.5W	P5	149.5W	Q5
6.5W	H6	22.5W	I6	38.5W	J6	54.5W	K6	70.5W	L6	86.5W	M6	102.5W	N6	118.5W	O6	134.5W	P6	150.5W	Q6
7.5W	H7	23.5W	I7	39.5W	J7	55.5W	K7	71.5W	L7	87.5W	M7	103.5W	N7	119.5W	O7	135.5W	P7	151.5W	Q7
8.5W	H8	24.5W	I8	40.5W	J8	56.5W	K8	72.5W	L8	88.5W	M8	104.5W	N8	120.5W	O8	136.5W	P8	152.5W	Q8
9.5W	H9	25.5W	I9	41.5W	J9	57.5W	K9	73.5W	L9	89.5W	M9	105.5W	N9	121.5W	O9	137.5W	P9	153.5W	Q9
10.5W	HA	26.5W	IA	42.5W	JA	58.5W	KA	74.5W	LA	90.5W	MA	106.5W	NA	122.5W	OA	138.5W	PA	154.5W	QA
11.5W	HB	27.5W	IB	43.5W	JB	59.5W	KB	75.5W	LB	91.5W	MB	107.5W	NB	123.5W	OB	139.5W	PB	155.5W	QB
12.5W	HC	28.5W	IC	44.5W	JC	60.5W	KC	76.5W	LC	92.5W	MC	108.5W	NC	124.5W	OC	140.5W	PC	156.5W	QC
13.5W	HD	29.5W	ID	45.5W	JD	61.5W	KD	77.5W	LD	93.5W	MD	109.5W	ND	125.5W	OD	141.5W	PD	157.5W	QD
14.5W	HE	30.5W	IE	46.5W	JE	62.5W	KE	78.5W	LE	94.5W	ME	110.5W	NE	126.5W	OE	142.5W	PE	158.5W	QE
15.5W	HF	31.5W	IF	47.5W	JF	63.5W	KF	79.5W	LF	95.5W	MF	111.5W	NF	127.5W	OF	143.5W	PF	159.5W	QF

Appendix E: Glossary

2D 16-State

Type of Forward Error Correction coding available on iDirect inbound carriers in DVB-S2 networks. 2D 16-State coding can provide better link margins, improved IP throughput and faster acquisition than Turbo Product Coding.

ABS

See Automatic Beam Selection (ABS).

ACM

See Adaptive Coding and Modulation (ACM).

Acquisition

The process whereby the satellite modem synchronizes its bursts with the upstream TDMA frame timing and joins an iDirect network.

Adaptive Coding and Modulation (ACM)

Adaptive Coding and Modulation. A method of applying coding to a data stream in DVB-S2 networks in which every BBFrame can be transmitted on a different MODCOD.

Antenna

Antenna used by satellite operator to communicate with satellite

Automatic Beam Selection (ABS)

An iDirect feature that automates the process by which roaming remotes select which network to join and automatically lock on to the associated outbound carrier. Also known as Automatic Beam Switching.

Beam

The physical footprint from a satellite Antenna onto the ground.

Burst Time Plan (BTP)

Slot allocation message sent to remote modems to indicate when each remote can transmit on the TDMA upstream carriers.

Carrier

A single modulated RF signal carrying information.

CCM

See Constant Coding and Modulation (CCM).

Constant Coding and Modulation (CCM)

A method of applying coding in a DVB-S2 data stream in which every BBFrame is transmitted at the same MODCOD.

Channel

A fixed section of bandwidth on the feeded link which is mapped to a Beam. The mapping is dynamic. Channels are bidirectional, with equal Inbound and Outbound bandwidths.

Comms-on-the-MOVE (COTM)

iDirect’s mobile remote feature.

COTM

See Comms-on-the-MOVE (COTM).

Downstream Carrier

Synonymous with Outbound Carrier. The satellite carrier transmitted from the hub to the remote satellite modem.

DVB-S2

A set of open standards for satellite digital broadcasting. DVB-S2 is an extension to the widely-used DVB-S standard and was introduced in March 2005.

Forward Error Correction

A schema for detecting and correcting transmission errors, at the cost of some additional bandwidth, to minimize the need for retransmission of packets across the satellite link.

Frequency Hopping

The ability of iDirect remotes to switch between TDMA carriers within an inroute group when transmitting to the hub.

Inbound Carrier

Synonymous with Upstream Carrier. The carrier transmitted from the remote satellite modem to the hub.

Indoor Unit (IDU)

The satellite modem and indoor devices (in contrast to Outdoor Unit or ODU).

Information Rate

The rate of transmission of user data over an upstream or downstream carrier including IP headers and iDirect overhead.

Inroute

A TDMA Upstream Carrier.

Inroute Group

A group of inroutes shared by a set of remotes in an iDirect network. Typically, a remote can frequency hop among the TDMA carriers within its inroute group.

LDCP

Low Density Parity Coding. The error correction coding scheme used in DVB-S2 networks.

MODCOD

The combinations of Modulation Types and Error Coding schemes supported on a satellite channel. The higher the modulation the greater the number of bits per symbol (or bits per Hz).

Outbound Carrier

See Downstream Carrier.

Outroute

See Outbound Carrier.

Quality of Service

Agreements on minimum and maximum throughput under normal conditions; plus what constitutes abnormal conditions.

Radio Frequency System

Provides a link between Antenna and Intermediate Frequency (IF). Its main function is to provide high power amplification, low noise amplification and up/down conversion between RF and IF.

Rain Fade

Adverse conditions, especially weather, can cause transmission and reception at a ground station to degrade (dramatic decrease of C/N). For satellite terminals, this requires a larger share of the available Inroute bandwidth (uplink) and a generic increase in the bandwidth per bit expended (downlink). For a SAS, Rain Fade is avoided by switching to the secondary SAS.

Roaming Remote

iDirect mobile remotes that use the Global NMS feature to “roam” from network to network around the globe. Roaming remotes are not constrained to a single location or limited to any geographic region.

SCPC

See Single Channel Per Carrier (SCPC).

Satellite Terminal

An integrated maritime (or other) system to be installed as a unit. It comprises a radome, Antenna, radio frequency subsystem, Antenna control mechanics and electronics, and the modem.

Single Channel Per Carrier (SCPC)

User data is transmitted to the satellite continuously on a single satellite carrier and can be received by a single location (point-to-point link) or multiple locations (point-to-multipoint link).

Spread Spectrum

A transmission technique in which a pseudo-noise (PN) code is employed as a modulation waveform to “spread” the signal energy over a bandwidth much greater than the signal information bandwidth.

Symbol Rate

The number of symbols that are transmitted in one second. From the symbol rate, calculate the bandwidth (total number of bits per second) by multiplying the bits per symbol by the symbol rate.

TDMA

See Time Division Multiple Access (TDMA).

Time Division Multiple Access (TDMA)

A type of over-the-air multiplexing by which two or more channels of information are transmitted simultaneously over the same link by allocating different time slots within TDMA frames for the transmission of each channel.

Transmission Rate

A measure of the speed of all over-the-air data. This includes the user data (Information Rate), iDirect overhead, and FEC encoding bits.

Upstream Carrier

Synonymous with Inbound Carrier. The carrier transmitted from the remote satellite modem to the hub.

Appendix F: Acronyms and Abbreviations

NOTE: This list is specifically for OpenBMIP and OpenAMIP.

- 0...9 -

16APSK

Sixteen Amplitude and Phase Shift Keying

8PSK

Eight Phase Shift Keying

- A -

A-TDMA

Adaptive Time Division Multiple Access

ABS

Automatic Beam Switching

AC

Alternating Current

ACM

Adaptive Coding and Modulation

ACS

Antenna Control System

APSK

Amplitude and Phase-shift keying

AZ

Azimuth

- B -

BB

BaseBand

BIST

Built-In Self-Test

BPN

BUC Part Number

BPSK

Binary Phase Shift Keying

BSN

BUC Serial Number

BUC

Block Up Converter

- C -

C/N

Carrier to Noise ratio

CPE

Customer Premise Equipment

CRC

Cyclic Redundancy Check

- D -

DAC

Digital to Analog Converter

dB

deciBel

dBi

deciBel isotropic

dBm

deciBel milli-Watt

dBW

deciBel Watt

DC

Direct Current

DVB-S2

Digital Video Broadcasting over Satellite, Second Generation

- E -

EIRP

Effective Isotropic Radiated Power

Eb/N0

Bit Energy to Noise Power Spectral Density ratio

EEPROM

Electrically Erasable Programmable Read-Only Memory

EL

Elevation

EMC

ElectroMagnetic Compatibility

EMI

ElectroMagnetic Interference

ETSI

European Telecommunications Standards Institute

- F-

FCC

Federal Communication Commission

FEC

Forward Error Correction

FID

Functional ID

- G -

G/T

Gain over Temperature

GHz

GigaHertz

GPIO

General-Purpose Input/Output

GPS

Global Positioning System

- I -

IBIT

Initiated Built In Test

IEC

International Electrotechnical Commission

IFL

Inter-Facility Link

IF

Intermediate-frequency

IP

Ingress Protection

IP

Internet Protocol

IR

Infrared

- J -

- K -

kbpskilobit per second

kHz

kilohertz

KRFU

Ku/Ka-band Radio Frequency Unit

ksps

kilosymbol per second

- L -

LAN

Local Area Network

LDP

CLow-Density Parity Coding

LNB

Low Noise Block Converter

LOS

Loss of Signal

- M -

Mbps

Megabits per second

Mcps

Megachips per second

MF-TDMA

Multi-Frequency TDMA

MHz

Megahertz

MID

Manufacturer ID

MIL-STD

US Military Standard

MODCOD

Modulation and Coding

Msps

Mega Symbols per Second

- N -

NF

Noise Figure

NORNot OR

- O -

OAE

Outside Antenna Equipment

ODU

Outdoor Unit

OEM

Original Equipment Manufactuer

OpenAMIP

Open Antenna-Modem Interface Protocol

OTA

Over The Air

OTP

One Time Programmable

- P -

PA

Power Amplifier

PLL

Phased Locked Loop

PSK

Phase Shift Keying

PSU

Power Supply Unit

- Q -

QPSK

Quadrature Phase Shift Keying

- R -

RF

Radio Frequency

RMS

Root Mean Square

ROM

Read-Only Memory

RSSI

Receive Signal Strength Indication

RTP

Real-Time Protocol

Rx or RX

Receive

- S -

SCPC

Single Channel Per Carrier

SNR

Signal to Noise Ratio

SSB

Single Side Band

- T -

TBD

To Be Determined

TBS

To Be Supplied

TDMA

Time Division Multiple Access

Tx or TX

Transmit

ST Engineering iDirect

13861 Sunrise Valley Drive, Suite 300
Herndon, VA 20171-6126
+1 703.648.8000
+1 866.345.0983
www.idirect.net