MetOcean Data Systems

iBCN

Interface Control Document

Version 1.3

March 2016

VERSION HISTORY

Version #	Author	Date	Notes
0.0	J. Beaudoin	10/28/2013	Initial Release
0.1	C. Tosuner	11/20/2013	Limited scope for the first release of the FW/SW
0.2	C. Tosuner	11/25/2013	Updated the mode configuration and device operation
0.3	J.Beaudoin	12/09/2013	Updated Event Data table and Event Data descriptions
0.4	C. Tosuner	12/17/2013	Added Extended Diagnostics command and Event Data definitions
0.5	C. Tosuner	1/7/2014	Indicated units for Voltage, Temperature Corrected GPS Timeout Event data
1.0	J. Beaudoin	8/1/2014	TD 14-014 Release
1.1	J. Beaudoin	1/8/2015	Update for iBCN-12K release. GPS Sat Check options.
1.2	J. Beaudoin	9/24/2015	Update to support new feature of only transmitting latest position report rather than all new records Support for Iridium data pass-through for custom messaging
1.3	J. Gavel	31/3/2016	Updates to support changes made in ECP 102101. - Commands for “Reset Reporting Index” behaviour - Events for “Reset Reporting Index” behavior

TABLE OF CONTENTS

1 Introduction

1.1 Purpose

This document is intended to contain the necessary details to communicate with the iBCN, describing the interface between the hardware and the software.

The intended audience of the iBCN Interface Control Document is technical integrators looking to communicate and decode messages with the iBCN.

2 Communication Protocol

The serial communication with the device is based on a binary protocol over various interface paths. The protocol is structured into a multi-layer scheme to optimize communication across the various interface paths.

2.1 Layer 1 – Physical Layer

Direct Serial Interface – This interface consists of RS232 or Virtual Com Port (via USB) connection with the device. The default baud rate is 19200 baud. The Direct Serial Interface requires packet framing to encapsulate the data packets as well as CRC error checking, therefore, Layer 2 – Link Layer is used for this interface.

Bluetooth Interface – This interface consists of a transparent serial connection using the Serial Port Profile (SPP) over Bluetooth. The host connecting device (ex. PC) is required to establish a Bluetooth connection with the device that results in a Virtual Com Port being available on the host. The connection baud rate is determined by the host connecting device upon connection, the range of baud rates supported is 9,600 to 57,600. This Bluetooth Serial Interface requires packet framing to encapsulate the data packets as well as CRC error checking. Layer 2 – Link Layer is used for this interface.

Iridium Interface – This interface consists of Short Burst Data (SBD) data packets being transferred between the user and the device. Since SBD is a packet based protocol, the purpose of Layer 2 – Link Layer is redundant and not used. The Iridium interface uses Layer 3 – Application Data Layer directly. Messages may be transmitted to the device through Iridium SBD services such as Email and Direct IP. Messages are received from the device through Email and/or Direct IP depending on modem provisioning.

2.2 Layer 2 – Link Layer

The Link Layer provides packet framing for the data communication. The packet frame is enclosed by Start of Frame (SOF) and End of Frame (EOF) Indicators. Immediately following the Start of Frame indicator is the packet data length. With this information, the End of Frame indicator is received in a predicable number of bytes. A CRC is included within this layer to provide error-checking. A valid Link Layer packet should satisfy the following requirements:

1. Start of Frame and End of Frame indicators received
2. Packet Data Length results in the EOF indicator received at the correct position
3. CRC calculation of received bytes matches the CRC value contained in the packet

Upon validation of a Link Layer packet, the Packet Data may be processed. Data not satisfying these requirements should be discarded and ignored.

<SOF> <PACKET_DATA_LENGTH> <PACKET_DATA> <CRC> <EOF>

<SOF>	0xAA 0x55	2 byte start of frame indicator
<PACKET_DATA_LENGTH>	Number of bytes in <PACKET_DATA>	2 bytes
<PACKET_DATA>	Payload Data	<PACKET_DATA_LENGTH> number of bytes
<CRC>	Based on CRC-CCITT (0xFFFF) calculation of data bytes within <PACKET_DATA_LENGTH> and <PACKET_DATA> fields	2 bytes
<EOF>	0xFF 0xCC	2 byte end of frame indicator

Notes:

- The total length of a Link 2 packet is <PACKET_DATA_LENGTH> + 8 bytes.
- Byte ordering of the transmitted data is Big Endian (Most Significant Byte transmitted first).

2.3 Layer 3 – Application Data Layer

The Application Data Layer provides the payload data to be processed. To provide a flexible system that allows various packets of different types to be transmitted to and from the device, a small header is used to identify the packet contents. Packet contents are pre-defined and documented, see Section 3.0 Command Definitions.

The Application Data Layer further defines the <PACKET_DATA> field of Layer 2. For interfaces not using Layer 2 (such as Iridium SBD), Layer 3 – Application Data Layer is used directly as the payload contents.

<CMD_TYPE> <CMD_SUB_TYPE> <SEQ_NUM> <PAYLOAD>

<PACKET_DATA>	**<CMD_TYPE>**	Bit 7 (MSB): Reserved for encryption Bits 0-6: define CMD_TYPE (see Section 3.0)
	<CMD_SUB_TYPE>	Bit 7 (MSB): Command (0) / Response (1) Bits 0-6: defines CMD_SUB_TYPE (see Section 3.0)
	<SEQ_NUM>	8-bit sequence number (0 to 255)
	<PAYLOAD>	N byte payload as defined by the <CMD_TYPE>, <CMD_SUB_TYPE> pair

Notes:

- The sequence number is used for individual packet identification. The sequence number will be echoed in the response packet. This field may be used by software tools to match up responses with the command packets.
- During packet processing, the CMD_TYPE and CMD_SUB_TYPE will be parsed and a response packet generated. The CMD_TYPE will be echoed as received and the CMD_SUB_TYPE will be echoed with the MSB set. All packets from the device will have the MSB of the CMD_SUB_TYPE set to indicate packet direction (ie. Response).
- As with Layer 2, byte ordering of the all multi-byte fields within the data packet is Big Endian (Most Significant Byte transmitted first).

3 Command Definitions

This section defines the payload contents of the supported command/response pairs to the device. This section only addresses the contents of the Application Data Layer. Depending on the Physical Layer used during communication, Link Layer data may also be required.

3.1 Asynchronous Messaging

During normal operation, the device will transmit periodically based on the operating configuration. Position reports and Alerts are transmitted autonomously from the device and do not require a command packet.

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Single Position Report	0x01	0x81	T_EVENT_REPORT (16 bytes)
Multiple Position Report	0x01	0xFD	n x T_EVENT_REPORT (n x 16 bytes)

Notes:

- The MSB of the <CMD_TYPE> field is reserved to denote the Payload contents are encrypted.
- The <SEQ_NUM> field is included as part of the Layer 3 packet data but is undefined for Asynchronous Messaging and therefore not shown in detail in the above Command Definitions.
- There are other asynchronous messages related to the Log Memory (downloading and clearing). See Section 3.2.4 Log Memory Commands for details.

3.2 Synchronous Messaging

The following messages are only transmitted upon request. These messages are responses to a polled command. The polled command takes on the same CMD_TYPE, CMD_SUB_TYPE of the desired response and the payload will be dependent on the command. For commands that write data to the device, the payload will be transmitted to the device. For command that request data from the device, the payload will be included in the response.

Note for the following command definition tables, the MSB of the <CMD_TYPE> field is reserved to denote the Payload contents are encrypted.

3.2.1 Status Commands

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Get Identification	0x03	0x01 (command)	None
	0x03	0x81 (response)	T_IDENTITY (128 bytes)
Get Diagnostic Status	0x03	0x02 (command)	None
	0x03	0x82 (response)	T_STATUS (13 bytes)
Get Extended Diagnostics	0x03	0x04 (command)	None
	0x03	0x84 (response)	T_EXTENDED_STATUS (83 bytes)
Request Position Report (Last Known Fix)	0x03	0x10 (command)	None
	0x03	0x90 (response)	T_POSITION_REPORT (16 bytes)
Request Position Report (New Fix) (Desirable)	0x03	0x11 (command)	None
	0x03	0x91 (response)	T_POSITION_REPORT (16 bytes)
Read Date Time	0x03	0x20 (command)	None
	0x03	0xA0 (response)	DATE_TIME (4 bytes)
Write Date Time	0x03	0x21 (command)	DATE_TIME (4 bytes)
	0x03	0xA1 (response)	Acknowledgement: 0x00 = Success 0xFF = Error

3.2.2 Configuration Commands

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Read Config Mode	0x05	0x01 (command)	Index (1 byte) 0 to (MAX_MODES-1) = read index 0xFF = read all modes
	0x05	0x81 (response)	Index (1 byte) T_MODE (all 0xFF if Index invalid)
Write Config Mode	0x05	0x11 (command)	Index (1 byte) (0xFF for all modes) T_MODE
	0x05	0x91 (response)	Acknowledgement: 0x00 = Success 0xFF = Error
Set Debug Output Level	0x05	0x20 (command)	Debug Level 0x00 = Disable 0x01 to 0x03 = Valid Debug Levels
	0x05	0xA0 (response)	Acknowledgement: 0x00 = Success 0xFF = Error

3.2.3 Bootloader Commands

The Bootloader commands are not fully implemented pending development of a robust bootloader capable of caching and verifying an executable on the device prior to initiating device programming.

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Start Bootloader Process	0x07	0x04 (command)	None
	0x07	0x84 (response)	Acknowledgement: 0x00 = Success 0xFF = Error

3.2.4 Data Pass-through Commands

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Send Iridium Message	0x03	0x30 (command)	Message (up to 337 bytes)
	0x03	0xB0 (response)	Acknowledgement: 0x00 = Message ready to be sent 0x01 = Transmission successful 0xFF = Error
Receive Iridium Message	n/a	n/a	n/a
	0x03	0xB1 (response)	Message (up to 337 bytes)

The data pass-through commands allow for custom payloads to be sent and received by the iBCN allowing software tools to use the iBCN as a data transfer device in addition to its beacon functionality.

With a local connection (direct serial or Bluetooth) established to the iBCN, the user may package custom payloads to be sent via Iridium using the "Send Iridium Message" command. The immediate response indicating "Ready to be sent" or "Error" indicates whether the message has been successfully queued for transmission. The actual Iridium transmission will be attempted when the device is ready to send (which may be delayed due to other messages already queued for transmission, coverage, etc). Upon successful Iridium transmission of this packet, another response will follow indicating "Success" or "Error" of the transmission.

The iBCN processes all incoming messages and handles the commands according to the <CMD_TYPE> and <CMD_SUB_TYPE> fields. The handling of the "Receive Iridium Message" command results in the complete message (<CMD_TYPE>, <CMD_SUB_TYPE>, and <PAYLOAD>) to be forwarded to the local connection (direct serial or Bluetooth).

3.2.5 Log Memory Commands

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Get Log Memory Status	0x09	0x01 (command)	None
	0x09	0x81 (response)	T_LOG_STATUS (16 bytes)
Start Download All	0x09	0x10 (command)	None
	0x09	0x90 (response)	Acknowledgement (1 byte): 0x00 = Success 0xFF = Error Start Index (4 bytes) End Index (4 bytes)
Start Download New Records Since Last Download	0x09	0x11 (command)	None
	0x09	0x91 (response)	Acknowledgement (1 byte): 0x00 = Success 0xFF = Error Start Index (4 bytes) End Index (4 bytes)
Stop Download	0x09	0x20	None
	0x09	0xA0	Acknowledgement (1 byte): 0x00 = Success 0xFF = Error
Clear Memory Log	0x09	0x30	None
	0x09	0xB0	Acknowledgement of Start (1 byte): 0x00 = Success 0xFF = Error
Reset Reporting Index	0x09	0x40	None
	0x09	0xC0	Acknowledgement (1 byte): 0x00 = Success 0xFF = Error

Notes:

- The Download and Clear Memory commands are time consuming tasks and operate on internal state machines within the device. These functions may result in multiple response packets as described in the next section.

The following 2 packet types are transmitted from the device during Log Memory operations to complete the requested task.

1) Any of the Start Download commands initiate the download state machine to provide data from the log memory to the user. Records are bundled into packets based on the T_RECORDS_PACKET structure and transmitted to the user until the download state machine is complete. Downloads will typically result in many transmitted Download Packets.

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Download Packet	0x09	0xFF	T_RECORDS_PACKET (Record Index MSB Set on last packet of download)

2) The Clear Memory command initiates a state machine to manage the internal log memory and complete the task. The internal log memory takes about 2 to 4 minutes to erase, so an additional status message is generated upon completion of Clearing Memory. During the 2 to 4 minutes while the Log Memory is being cleared, it is unavailable for record storage and downloading.

Description	<CMD_TYPE>	<CMD_SUB_TYPE>	<PAYLOAD>
Clear Memory Complete	0x09	0xFE	Acknowledgement of Complete (1 byte): 0x00 = Success 0xFF = Error

Download Operation

Get Log Memory Status
Start Download with option of All or New Records
Acknowledge last record of downloaded block (delivery to Iridium or response from local connection)
Stop Download or Wait until Download completes

4 Events

4.1 Event Log Codes

The following codes are defined to uniquely identify events within the Log Memory. Each record in the Log contains a Timestamp as well as an Event Code. Based on the Event Code, the remaining Event Data bytes of the record are defined.

<Timestamp> <Event Code> <Event Data>

Timestamp: 4 byte long integer representing GPS time, the number of seconds since Jan 6, 1980.

01-01-2000 00:00:00 = 630720000

01-01-2013 00:00:00 = 1041033600

Note: GPS Time differs from UTC time by the number of GPS leap seconds, currently 16 seconds.

Event Code	Event Description	Event Data	Notes
1	Reserved		
2	Power Up	T_EVENT_DATA_POWER_UP	
3	Battery Status	T_EVENT_DATA_BATTERY_STATUS	
4	Reserved		
5	Reserved		
6	Reserved		
7	Config Update	T_EVENT_DATA_CONFIG_UPDATE	
8	Reserved		
9	GPS Timeout	T_EVENT_DATA_GPS_TIMEOUT	
10	Iridium Timeout	n/a	
11	Reserved		
12	Reserved		
13	Reserved		
14	Iridium Data Pass-through Message sent	Size of Iridium Message Sent	
15	Time Set	T_EVENT_DATA_TIME_SET	
16	GPS Position requested by user	T_EVENT_DATA_GPS_POSITION_REQUEST	
17	Reserved		
18	Reserved		
19	Reserved		
20	Iridium Data Pass-through Message received	Size of Iridium Message Received	
21	Reset Reporting Index Request	T_EVENT_DATA_CONFIG_UPDATE	When a request to reset the reporting index is made, this event is stored. The source of the message is stored with the event.
22	Reporting Index Reset		when the reporting index has been reset, this event is stored
128+ (MSB set)	GPS Position	See T_POSITION_REPORT (subset of T_EVENT_REPORT)	

5 Data Types and Structures

The following primitive data types are defined within the system:

```
s8;                 // 8-bit signed value
u8;                 // 8-bit unsigned value
s16;                // 16-bit signed value
u16;                // 16-bit unsigned value
s32;                // 32-bit signed value
u32;                // 32-bit unsigned value
char;               // 8-bit ASCii character, same as u8
BOOL;               // TRUE/FALSE indicator, same as u8
DATE_TIME;          // u32, seconds since Jan 6, 1980 or encoded h:m:s d/m/y
```

The following data structures are defined within the system:

```
T_EVENT_REPORT
{
  DATE_TIME timestamp;  // timestamp of event report
  u8 event_code;        // MSB clear to indicate Event Report, remaining bits specify which event
  u8 event_data[11];    // data bytes dependent upon event code
}

T_POSITION_REPORT
{
  DATE_TIME timestamp;  // timestamp of position report
  u8 event_code;        // MSB set to indicate Position Report, remaining bits available to use
                        // bit 7: set to indicate Position Report
                        // bit 6: motion bit to indicate whether in motion or not
                        // bit 5:3 – Number of satellites (0-7 map to 3-10+)
                        // bit 2:0 – HDOP (0-7 map to 0-3.5+)
  u32 latitude;         // fixed-point integer of degrees latitude scaled by 1,000,000 and offset by +90.0
  u32 longitude;        // fixed-point integer of degrees longitude scaled by 1,000,000 and offset by +180.0
  u8 speed;             // meters per second
  u8 fix_accuracy;      // meters
  u8 time_to_fix;       // seconds
}

T_RECORDS_PACKET
{
  u32 record_index;
  T_EVENT_REPORT records[12];
}
```

T_LOG_STATUS

```
{
  u32 record_count;
  u32 last_record_downloaded;
  DATE_TIME first_record_timestamp;
  DATE_TIME last_record_timestamp;
}
```

T_STATUS

```
{
  DATE_TIME time;
  u16 battery_voltage;      // voltage (V) x 10
  s8 temperature;           // in degrees Celsius
  u8 reserved1;
  u8 gps_status;
  u8 reserved2[4];
}
```

T_EXTENDED_STATUS

```
{
  u8 reserved1[32];
  u8 imei[15];
  u8 reserved2[36];
}
```

T_MODE

```
{
  u32 reserved;
  u32 fixInterval; // In seconds
  u16 Reserved;
  u16 mailboxCheckInterval; // In number of fixes
  u8 surfacingSetting;      // surfacing settings:      0 = Always On,
                            //                          1 = Conductivity Water Sensor,
                            //                          2 = GPS Sat Check @ 5 min,
                            //                          3 = GPS Sat Check @ 30 min,
                            //                          4 = GPS Sat Check @ 1hour,
                            //                          5 = GPS Sat Check @ 3 hours
  u16 transmitIntervalFlags;  //  Flags: 0 = Transmit 'New' Records from Record Log each transmitInterval
                              //         1 = Transmit Only Latest Position Report each transmitInterval
  u16 transmitInterval; // In number of fixes
}
```

T_IDENTITY

```
{
 u8 model_number[24];   // MMI-513 or MMI-613
 u32 serial_number;     //
 u8 fw_rev[4];          // major.minor.revision
 u8 iridium_fw_rev[16]; // ASCii string - TAxxxxxx
 u8 gps_fw_rev[80];     // ASCii string – 80 characters
}
```

T_EVENT_DATA_POWER_UP

```
{
 u8 reset_reason;
}
```

T_EVENT_DATA_BATTERY_STATUS

```
{
 u16 voltage;           // voltage (V) x 10
 s8 temperature;        // in degrees Celsius
}
```

T_EVENT_DATA_CONFIG_UPDATE

```
{
 u8 source;             // 0 = internal, 1 = Iridium OTA, 2 = reserved, 3 = Bluetooth
}
```

T_EVENT_DATA_GPS_TIMEOUT

```
{
 u32 latitude;          // fixed-point integer of degrees latitude scaled by 1,000,000 and offset by +90.0
 u32 longitude;         // fixed-point integer of degrees longitude scaled by 1,000,000 and offset by +180.0
 u8 speed;              // meters per second
 u8 fix_accuracy;       // meters
 u8 num_satellites;
}
```

T_EVENT_DATA_TIME_SET

```
{
 u8 source;             // 0 = GPS, 1 - user
 u32 offset;
}
```

T_EVENT_DATA_GPS_POSITION_REQUEST

```
{
 u8 source;             // 0 = reserved, 1 = command
}
```

6 User Configuration

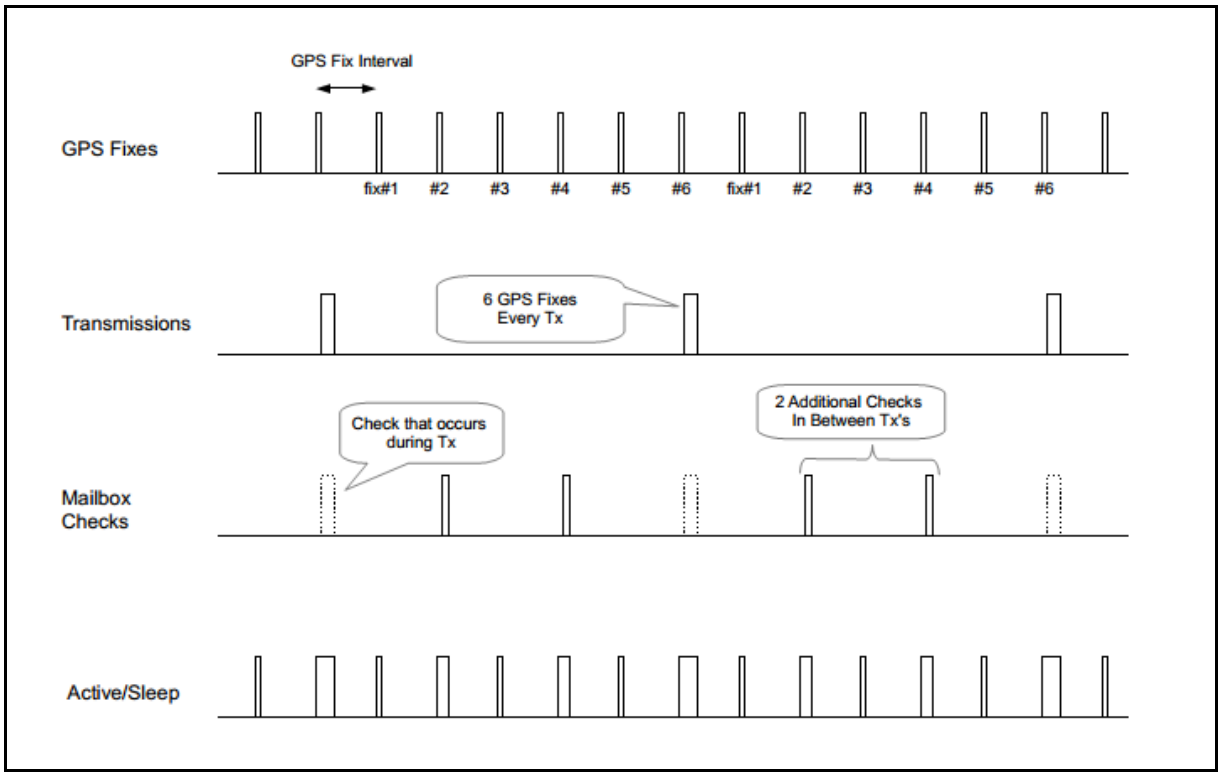
The user configuration is a collection of data structures that provide flexibility for the user to configure the device for various operating profiles.

6.1 Operating Mode

Operating mode is defined by the structure **T_MODE** as follows:

- fixInterval: interval to acquire GPS fixes to log specified in seconds, 0 = disabled.
- transmitInterval: interval to transmit Position Reports specified in number of fixes, 0 = disabled.
- mailboxCheckInterval: interval to check for incoming messages specified in number of fixes, 0 = mailbox check only during transmissions.

The following diagram illustrates the device operation during an example scenario.



In this example:

- GPS interval is set to 1 hour.
- Transmission interval is set to every 6 fixes: they occur every 6 hours.
- Mailbox check is set to every 2 fixes: they occur every 2 hours; 1 check during transmission then 2 additional mailbox checks between transmissions.