Marvelmind デバイスとのデータ交換のハードウェアインターフェースおよびプロトコル

バージョン 2026.02.24
ファームウェア v7.000 以降に対応

www.marvelmind.com

目次

1. Marvelmind デバイスへの接続

Marvelmind デバイス（Modem またはモバイルビーコン Hedgehog）と通信するには、以下のいずれかのインターフェースを介して外部デバイス（ロボット、コプター、AGV など）に接続する必要があります。

1. CDC クラスの USB デバイス（Windows ではバーチャル COM ポート、Linux では ttyACM または ttyUSB ）として USB ホストに接続します。Windows ではドライバーが必要であり、Modem 用と同じドライバーを使用します。Linux では、必要なドライバーが Linux カーネルに統合されているため、ドライバーは不要です。このインターフェースでは実際の RS-232 を使用しないため、ホスト側でオープンするシリアルポートのパラメーター（ボーレート、ビット数、パリティなど）は任意の値に設定できます。
2. UART に接続します。ストリーミング用には 2 本のワイヤーをピンにはんだ付けし、双方向通信には 3 本のワイヤーが必要です。以下のハードウェアインターフェースの図を参照してください。UART トランスミッターのロジックレベルは CMOS 3.3V です。デフォルトのボーレートは 500 kbps で、Dashboard から以下のリストにより設定可能です: 4.8、9.6、19.2、38.4、57.6、115.2、500 kbps。データフォーマット: 8 ビット、パリティなし、ストップビット 1。
3. SPI に接続します。Marvelmind デバイスは SPI スレーブデバイスとして動作します。SPI のパラメーター: SPI モード 0、各バイト内の MSB が最初に送信されます。接続は SCK スピード最大 8 MHz でテスト済みです。高速（500 kHz 以上）では、配線接続の品質に注意してください。
4. RS-485 に接続します（Super-Modem または Industrial Super-Beacon のみ）。
5. I2C に接続します（Super-Beacon のみ）。
6. Wi-Fi 経由の UDP に接続します（Super-Modem の場合）、または任意のネットワーク接続（Dashboard の場合）。
7. CAN に接続します（Industrial Super-Beacon、およびリクエストにより提供される Super-Modem に対応）。

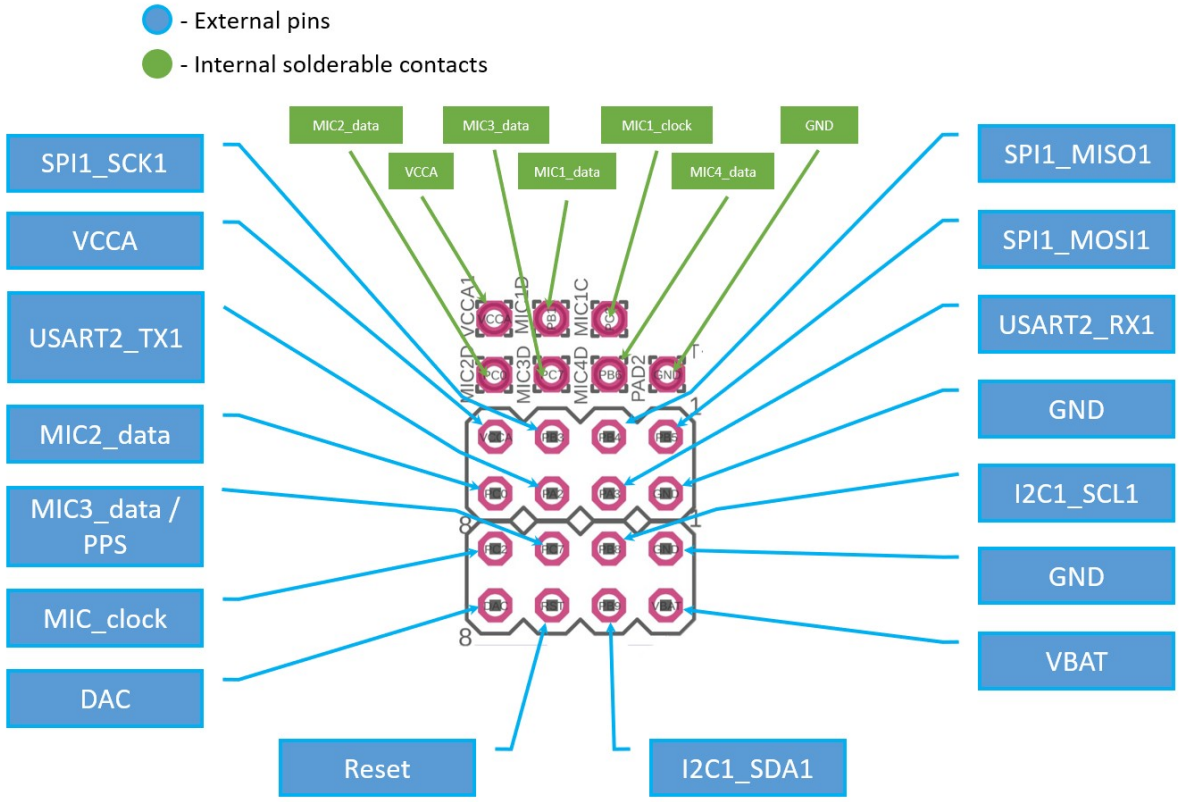
接続設定の概要:

インターフェース	ビットレート	その他の設定
USB（バーチャル UART）	UART のビットレートは適用されません。シリアルポートの速度は任意の値に設定できます。 USB フルスピード（12 Mbit/s）によるデータ転送	UART 設定は適用外
UART	4.8、9.6、19.2、38.4、57.6、115.2、500 Kbit/s Dashboard で選択可能	データビット 8 ビット、ストップビット 1 ビット、パリティなし

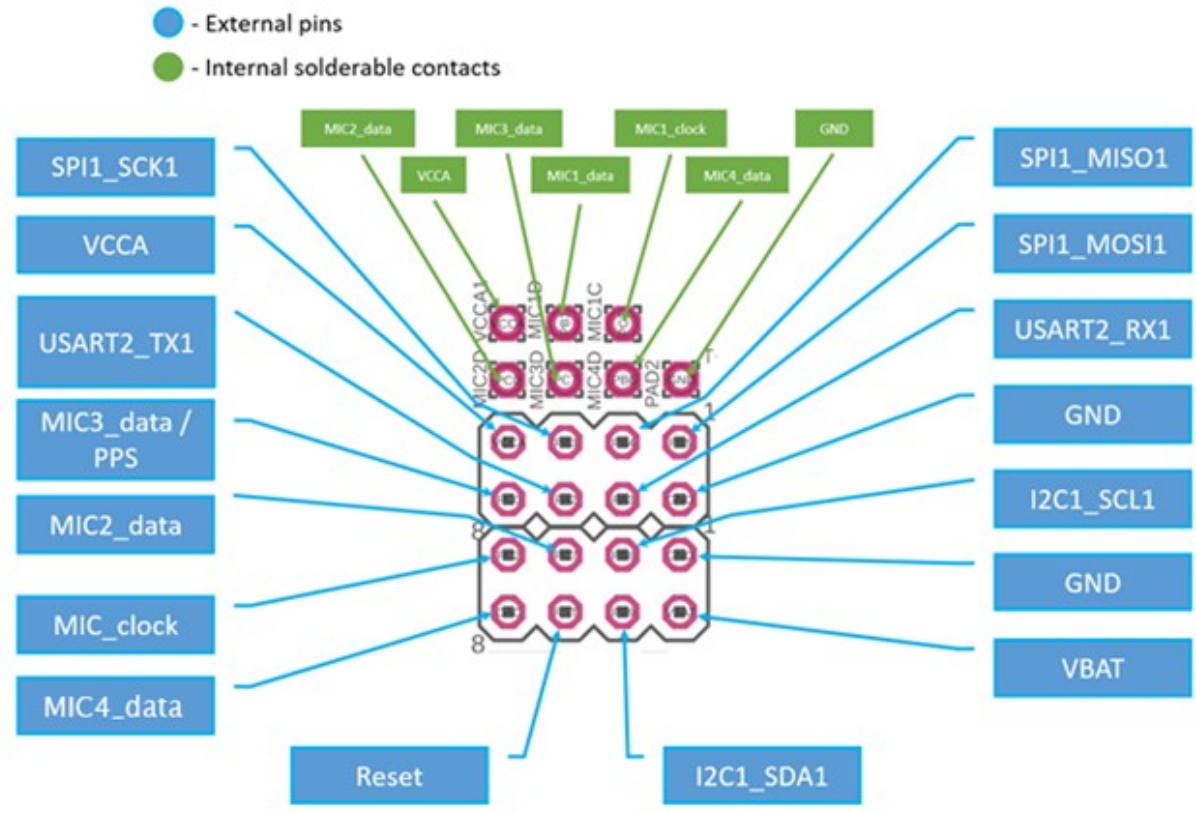
SPI	8 Mbit/s まで動作確認済み	SPI モード 0
RS-485	4.8、9.6、19.2、38.4、57.6、115.2、500 Kbit/s Dashboard で選択可能（UART と同様）	データビット 8 ビット、ストップビット 1 ビット、パリティなし
I2C	最大 400 Kbit/s	
UDP	ネットワーク接続速度に依存	
CAN	125 Kbit/s	標準フレーム

1.1 Super-Beacon の UART およびその他のインターフェース

Super-Beacon 用 4x4 ピン配列:



Super-Beacon-2 および Super-Beacon-3 用 4x4 ピン配列:

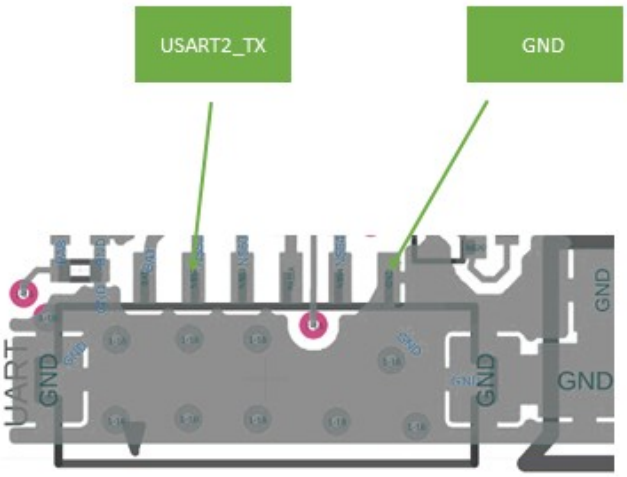


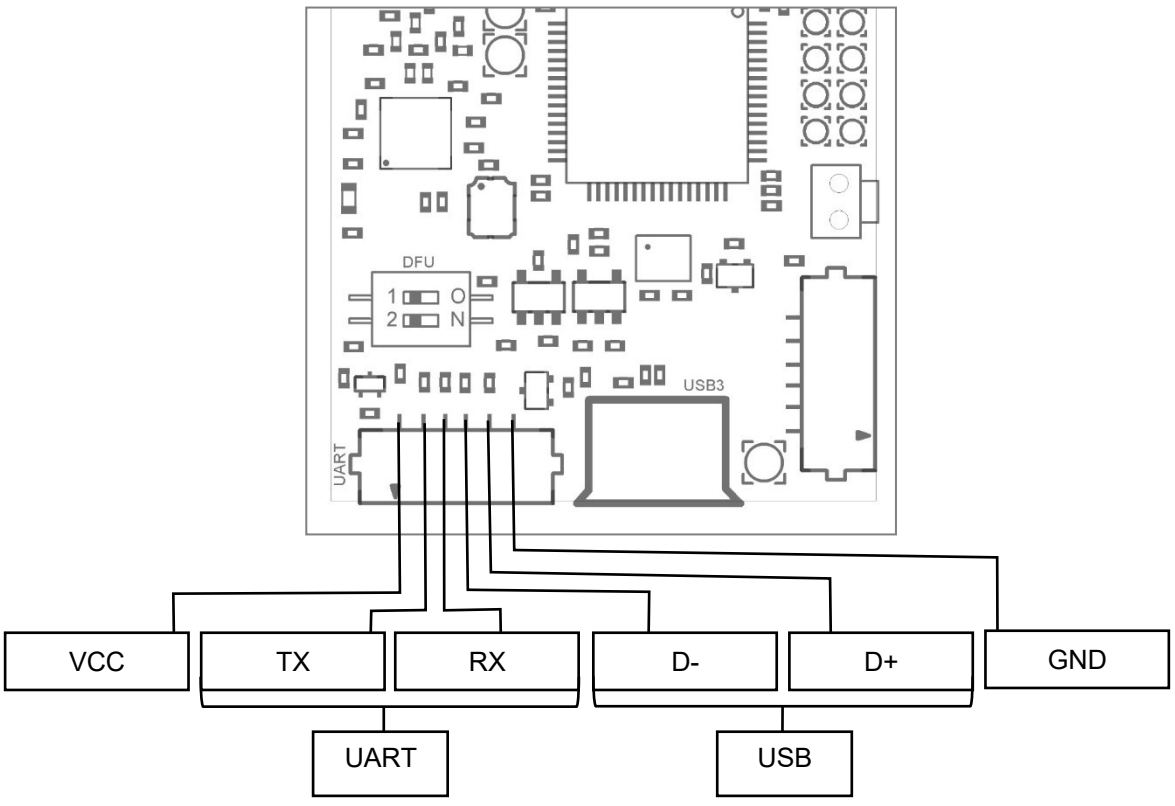
1.2 beacon Mini-RX の UART およびその他のインターフェース

正しくはんだ付けできると確信している場合にのみ使用してください。DIP-switches で beacon の電源を切ることを忘れないでください。はんだ付けが不良で beacon を破損した場合、Marvelmind チームはその責任を負いません。

beacon Mini-RX から UART データストリーミングを取得するには、基板上のピンにはんだ付けする必要があります。

- Internal solderable contacts





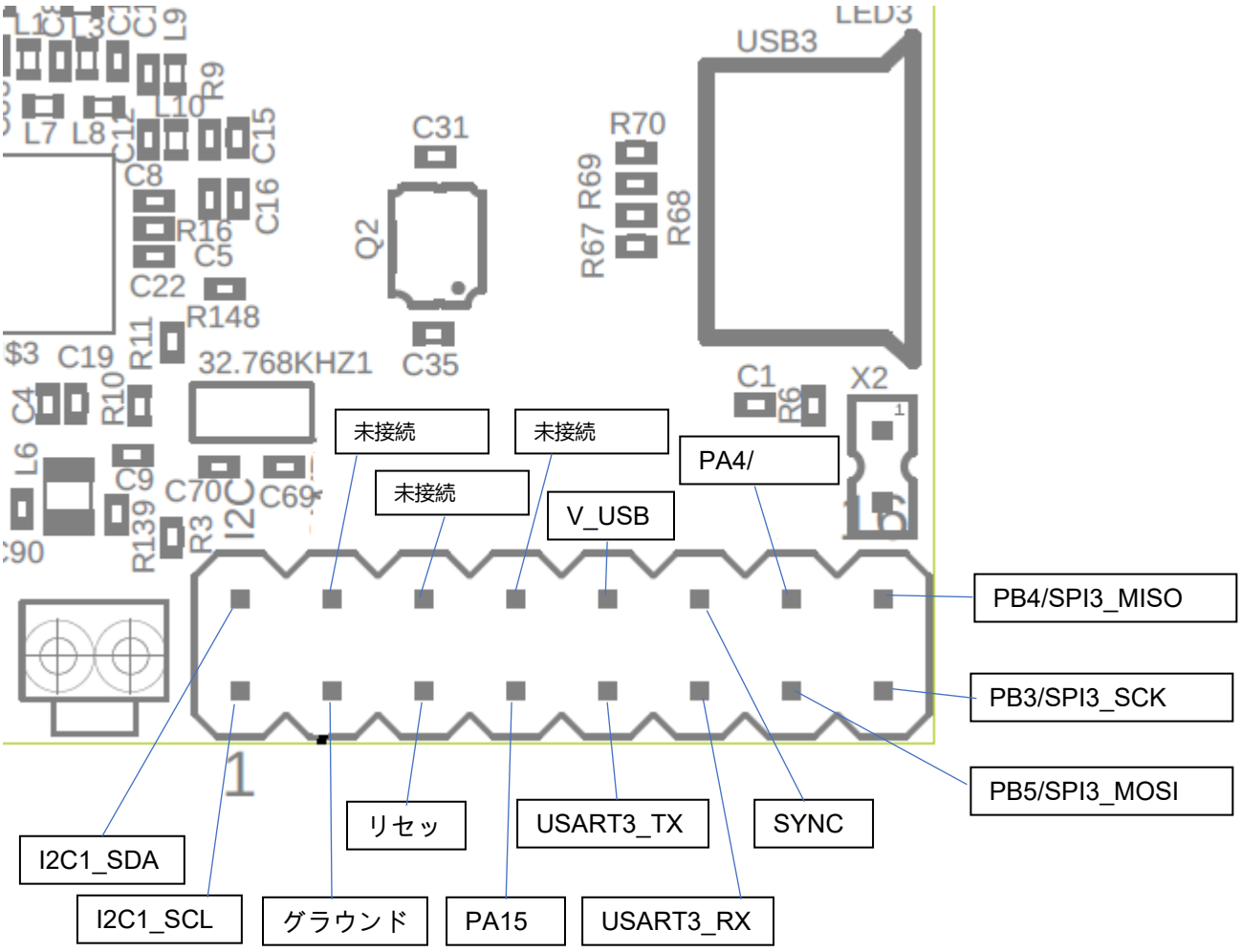
Marvelmind robotics

1.3. ビーコン Mini-TX-2 の UART およびその他のインターフェース

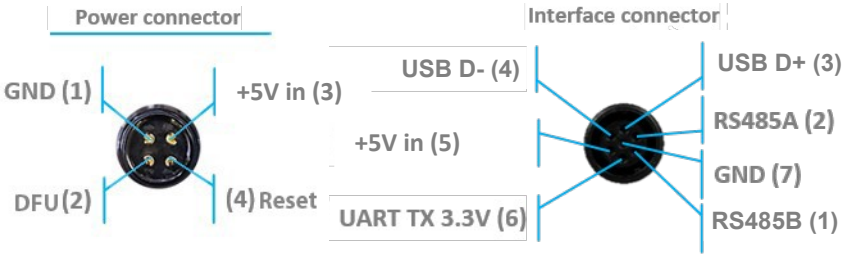
ビーコン Mini-TX-2 は、ビーコン Mini-RX と同じピン配列のコネクターを備えています。

UART ケーブルは Mini-TX-2 への接続に使用できます。

1.4. Modem HW v5.1 の UART およびその他のインターフェース



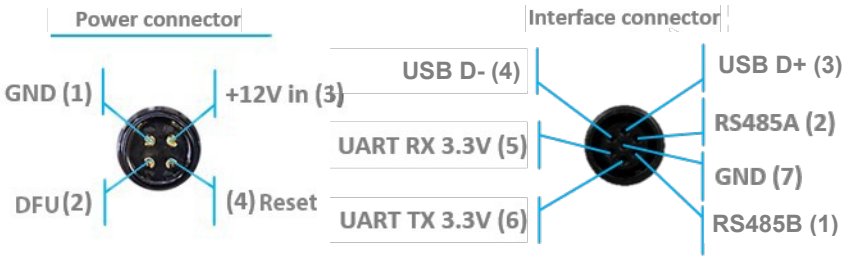
1.5. Super-Modem の UART およびその他のインターフェース



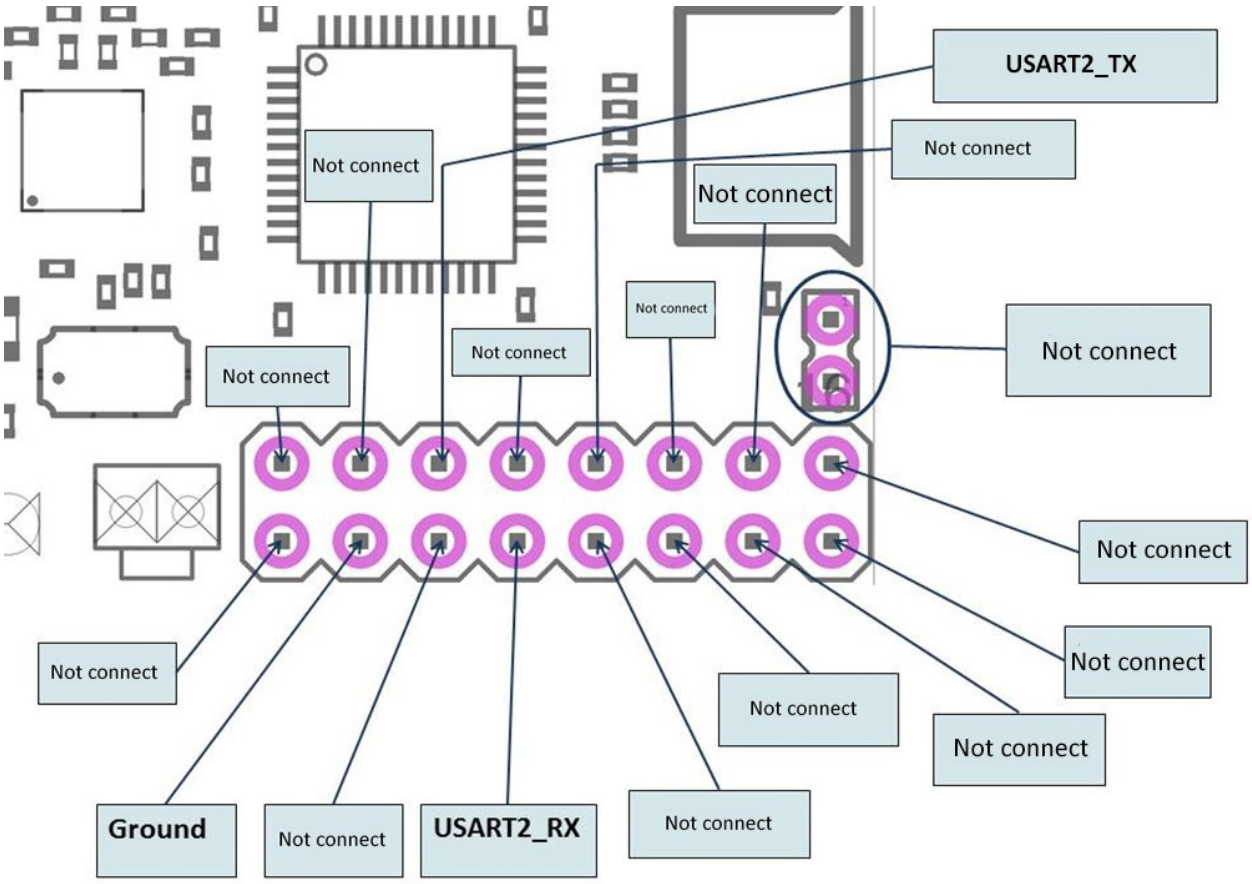
また、Super-Modem にはオンボード Wi-Fi インターフェースが搭載されています。Wi-Fi 接続の設定は UDP の章に記載されています。

- 新バージョンの Super-Modem（2023 年 6 月以降）は+5V 電源のみに対応しています。このバージョンには+12V 電源コンバーターを使用しないでください。ビーコンが破損する恐れがあります。

-



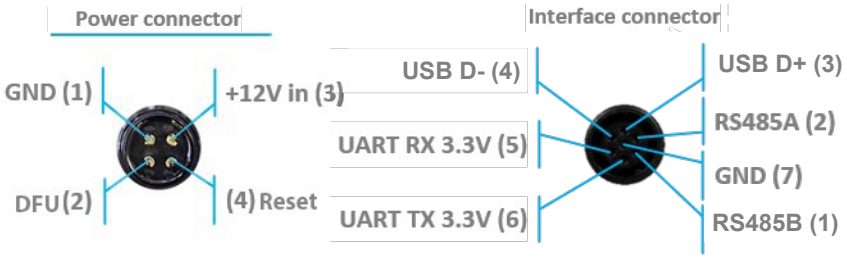
1.6. Modem HW v4.9 の UART およびその他のインターフェース



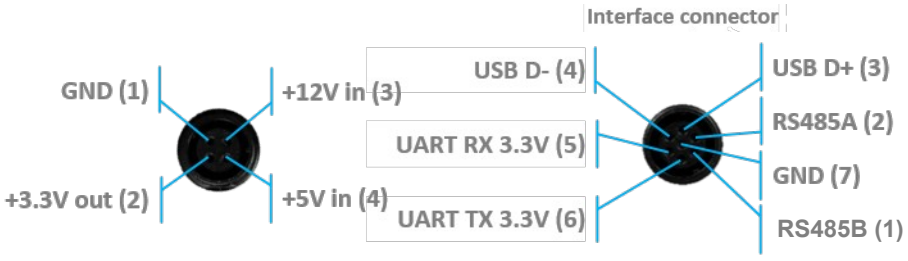
1.7. Industrial-TX、Industrial-RX、Industrial Super-Beacon の UART およびその他のインターフェース

- Industrial-TX、Industrial-RX、Industrial Super-Beacon のバージョン 2 および 3（2022 年 6 月以降）は、+5V 電源のみに対応しています。このバージョンに +12V 電源コンバーターを使用しないでください。ビーコンが破損します。

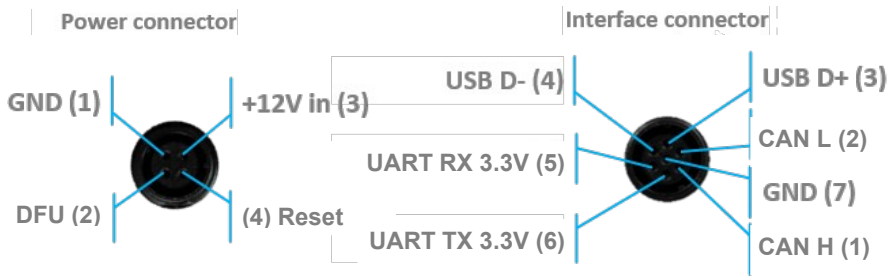
- このバージョンには UART RX がありませんが、インターフェースコネクターを電源供給として使用することが可能になりました。



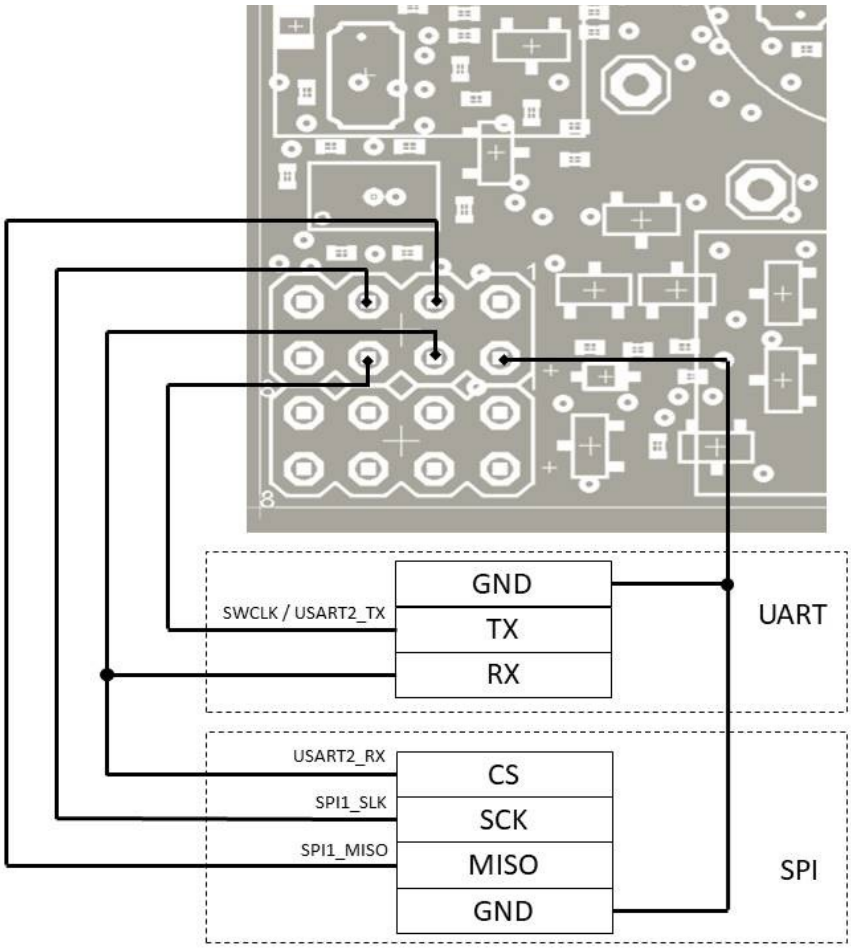
RS485 変更ピン配置（2019 年 9 月以前）



CAN 修正のピン配置

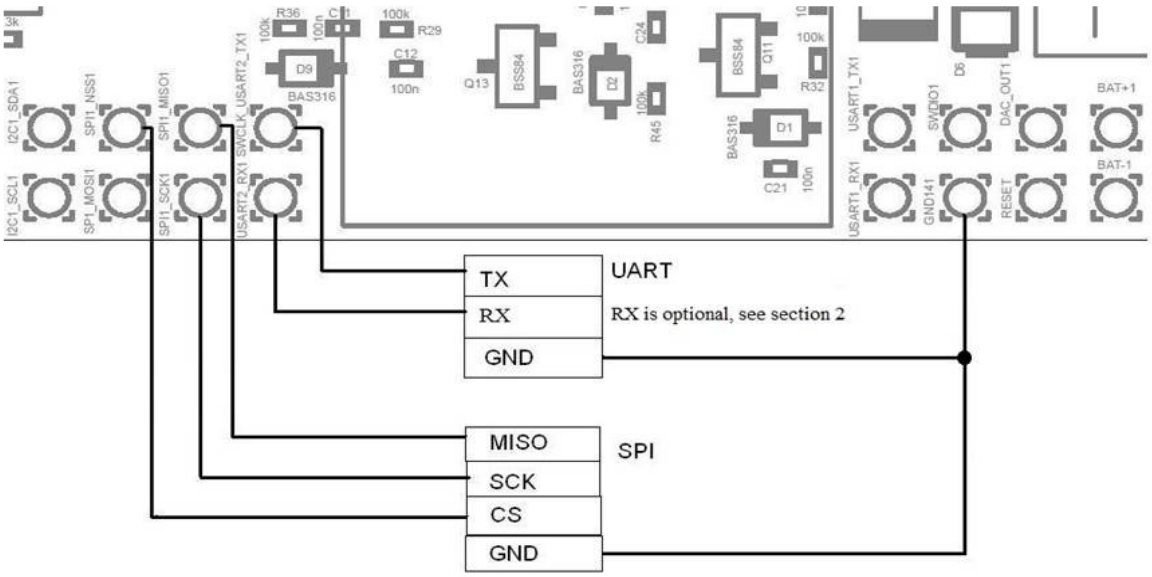


1.8. ビーコン HW v4.9 の UART および SPI インターフェース



注意：UART RX と SPI CS は同じ共有ピンを使用しています。このピンの機能（UART レシーバー、SPI チップセレクト、またはその他）は、Dashboard の「Interfaces」セクションにある「PA15 pin function」パラメーターで選択できます。

1.9. ビーコン HW v4.5 の UART および SPI インターフェース



2. UART による通信プロトコル

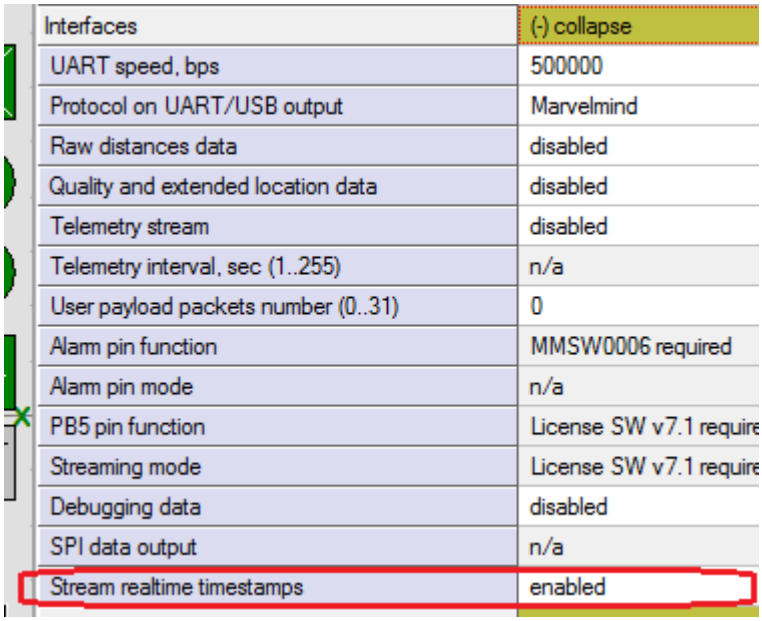
2.1　ストリーミング用'Marvelmind'プロトコル

すべてのストリーミングパケットは同一の基本構造を持つ:

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	宛先アドレス	0xff
1	1	uint8_t	パケットタイプ	0x47
2	2	uint16_t	パケット内データコード	詳細参照
4	1	uint8_t	送信データのバイト数	N
5	N	N バイト	データフィールドのコードに応じたペイロードデータ	
5+N	2	uint16_t	CRC-16（付録 1 参照）	

ソフトウェアバージョン v7.200 以降、リアルタイムタイムスタンプがデフォルトで有効になっています。これにより、パケット 0x0081、0x0083、0x0084、0x0085 が、それぞれパケット 0x0011、0x0003、0x0004、0x0005 の代わりにストリーム出力されます。

旧バージョンのソフトウェアとの互換性のためにローカルタイムスタンプを使用した旧ストリーミング形式が必要な場合は、Dashboard のデバイス設定でこのオプションを無効にすることができます。



2.1.1 Hedgehog 座標パケット

このパケットは、新しい座標が測定されるたびに、または測定に失敗するたびに送信されます。

2.1.1.1 モバイルビーコンの mm 解像度座標とリアルタイムタイムスタンプを含むパケット（ファームウェア V7.200 以降）

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：対応

Super-Modem：対応

Mini-RX（Badge、Helmet 等）：UART ケーブルを使用して対応

Mini-TX：現行の HW バージョンでは非対応

Mini-TX-2：UART ケーブルを使用して対応

Modem HW4.9：対応

Beacon HW4.9: サポート済み

Beacon HW4.5: サポート済み

タイムスタンプに関する注意事項を参照してください。

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	宛先アドレス	0xff
1	1	uint8_t	パケットの種類	0x47
2	2	uint16_t	パケット内のデータコード	0x0081
4	1	uint8_t	送信データのバイト数	N
5	8	int64_t	タイムスタンプ – Unix 時間 - 1970.01.01 00:00:00 からのミリ秒数。 Modem およびダッシュボードを持つすべてのデバイスによって同期された時刻。	
13	4	int32_t	Beacon の X 座標、mm	
17	4	int32_t	Beacon の Y 座標、mm	
21	4	int32_t	Z 座標、Beacon の高さ、mm	
25	1	uint8_t	フラグのバイト: ビット 0: 1 - 座標が利用不可。X、Y、Z フィールドのデータは使用しないこと。 ビット 1: タイムスタンプ単位インジケーター（注記参照） ビット 2: 1 - ユーザーボタンが押されている（V5.23+） ビット 3: 1 - ユーザーデバイスへのアップロード用データが利用可能、セクション 2 参照（V5.34+） ビット 4: 1 - ユーザーデバイスからのデータダウンロードを要求、セクション 2 参照（V5.34+） ビット 5: 1 - 第 2 ユーザーボタンが押されている（V5.74+）	

			ビット 6：1 - 別の Hedgehog 向けのデータ（このパケットを送信している Hedgehog と同一ではない） ビット 7：1 - ジオフェンシングゾーン外	
26	1	uint8_t	Hedgehog のアドレス	
27	2	uint16_t	ビット 0～11：XY 平面における Hedgehog ペアの向き、デシ度（0～3600） ビット 12：1 – 座標はビーコンペアの中心に対して指定；0 – 指定された Hedgehog の座標 ビット 13：1 – 向きは適用不可 ビット 14～15：予約済み（0）	
29	2	uint16_t	超音波放射から現在時刻までの経過時間、ミリ秒（V5.88+）	
31	M=N-26		オプションデータ – リストを参照	
31+M	2	uint16_t	CRC-16（付録 1 を参照）	

モバイルビーコン位置パケットのオプションデータには、以下の構造を含めることができます：

- 速度データ（7 バイト）。Dashboard のモバイルビーコン Settings のインターフェースセクションで有効にする必要があります

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	データフィールドのコード = 1 は速度ベクトルを意味する	**1**
1	2	int16_t	X 方向の速度、mm/sec	
3	2	int16_t	Y 方向の速度、mm/sec	
5	2	int16_t	Z 方向の速度、mm/sec	

2.1.1.2　モバイルビーコンの cm 分解能座標パケット

廃止されたパケット。0x0081 に置き換えられました。廃止パケットの詳細はこちらをご参照ください。

2.1.1.3　モバイルビーコンの mm 分解能座標パケット

廃止されたパケット。0x0081 に置き換えられました。廃止パケットの詳細はこちらをご参照ください。

2.1.2 固定ビーコンの座標パケット

このパケットはマップがフリーズされているときに送信され、10秒ごとに繰り返されます。

2.1.2.1 固定ビーコンの mm 分解能座標パケット、データコード 0x0012（ファームウェア V5.35 以降）

対応ハードウェア:

Super-Beacon：対応
Industrial Super-Beacon：対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、ヘルメット等）：UART ケーブルで対応
Mini-TX：現在の HW バージョンでは未対応
Mini-TX-2：UART ケーブルで対応
Modem HW4.9：対応
Beacon HW4.9：対応
Beacon HW4.5：対応

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0012
4	1	uint8_t	送信データのバイト数	1+N*14
5	1	uint8_t	パケット内の Beacon 数	N
6	1	N*14 バイト	N 個の Beacon のデータ	
6+N*14	2	uint16_t	CRC-16（付録 1 参照）	

N 個の各 Beacon のデータ構造フォーマット:

オフセット	サイズ（バイト）	型	説明
0	1	uint8_t	Beacon のアドレス
1	4	int32_t	Beacon の X 座標、mm
5	4	int32_t	ビーコンの Y 座標（mm）
9	4	int32_t	ビーコンの Z 座標（高さ）（mm）
13	1	uint8_t	ビット 0：1 = 位置情報が適用されない ビット 1…7：予約済み

2.1.2.2 固定ビーコンの cm 分解能座標を含むパケット、データコード 0x0002

廃止されたパケット。0x0012 に置き換えられました。廃止パケットの詳細はこちらを参照してください。

2.1.3 慣性センサーの生データパケット

このパケットは、新しい慣性センサーデータが利用可能になった際に送信されます。

2.1.3.1 リアルタイムタイムスタンプ付き慣性センサー生データパケット、データコード 0x0083（ファームウェア V7.200 以降）

このパケットは、新しい慣性センサーデータが利用可能になった際に送信されます。

対応ハードウェア:

Super-Beacon：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

Industrial Super-Beacon：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

Modem HW5.1：対応、システムアップデートレート（「IMU via modem」が有効な場合）

Super-Modem：対応、システムアップデートレート（「IMU via modem」が有効な場合）

Mini-RX（Badge、Helmet など）：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

UART ケーブルを使用

Mini-TX：現在の HW バージョンではサポートされていません

Mini-TX-2：サポート対応、100 Hz（'Raw inertial sensors data'が有効な場合）

Modem HW4.9：サポート対応、システム更新レート（'IMU via modem'が有効な場合）

Beacon HW4.9：サポート対応、100 Hz（'Raw inertial sensors data'が有効な場合）

Beacon HW4.5：サポート対応、100 Hz（'Raw inertial sensors data'が有効な場合）

タイムスタンプに関する注意事項を参照してください。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0083
4	1	uint8_t	送信データのバイト数	
5	36		データパケット（以下参照）	
41	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	型	説明	値
0	2	int16_t	加速度センサー、X 軸、1 mg/LSB	
2	2	int16_t	加速度センサー、Y 軸、1 mg/LSB	

4	2	int16_t	加速度センサー、Z 軸、1 mg/LSB	
6	2	int16_t	ジャイロスコープ、X 軸、0.0175 dps/LSB	
8	2	int16_t	ジャイロスコープ、Y 軸、0.0175 dps/LSB	
10	2	int16_t	ジャイロスコープ、Z 軸、0.0175 dps/LSB	
12	2	int16_t	コンパス、X 軸、1100 LSB/Gauss	
14	2	int16_t	コンパス、Y 軸、1100 LSB/Gauss	
16	2	int16_t	コンパス、Z 軸、980 LSB/Gauss	
18	1	uint8_t	ビーコンのアドレス	
19	5	5 バイト	予約済み（0）	
24	8	int64_t	タイムスタンプ – Unix 時間 - 1970.01.01 00:00:00 からのミリ秒数 Modem および Dashboard を使用する全デバイスで同期された時刻	
32	1	uint8_t	フラグ: ビット 0：1 = 加速度計データ 利用不可 ビット 1：1 = ジャイロスコープデータ利用不可 ビット 2：1 = Compass データ利用不可 ビット 3...7 – 予約済み（0）	
33	3	3 バイト	予約済み	

注意：Compass データは IMU 搭載の HW v4.9 ビーコンでのみ利用可能です。

2.1.3.2 生慣性センサーデータのパケット、データコード 0x0003

廃止済みパケット。0x0083 に置き換えられました。廃止済みパケットの詳細はこちらをご参照ください。

2.1.4 生距離データのパケット

2.1.4.1 リアルタイムタイムスタンプ付き生距離データのパケット、データコード 0x0084（ファームウェア V7.200 以降）

このパケットは、モバイルビーコンの座標を含むパケットの後に、新しい座標が計測されるか計測に失敗するたびに送信されます。

設定の「Interfaces」セクションで「raw distances data」オプションが有効になっている場合のみ利用可能です。

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：対応

Super-Modem：対応

Mini-RX（Badge、Helmet など）：UART ケーブルを使用することで対応

Mini-TX：現行 HW バージョンでは非対応

Mini-TX-2：UART ケーブルを使用することで対応

Modem HW4.9：対応

Beacon HW4.9: サポート対象

Beacon HW4.5: サポート対象

タイムスタンプに関する注意事項を参照してください。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0084
4	1	uint8_t	送信データのバイト数	
5	36		データパケット（以下を参照）	
41	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Hedgehog のアドレス	
1	6		距離アイテム 1	
7	6		距離アイテム 2	
13	6		距離アイテム 3	
19	6		距離アイテム 4	
25	8	int64_t	タイムスタンプ – ビーコン超音波送信の Unix 時刻、1970.01.01 00:00:00 からのミリ秒数 Modem および Dashboard によりすべ	

			てのデバイス間で同期された時刻	
29	2	uint16_t	超音波送信から現在時刻までの経過時間、ミリ秒単位（V5.89+）	
31	1	uint8_t	予約済み	

距離アイテムのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Beacon のアドレス（アイテムが未入力の場合は 0）	
1	4	uint32_t	Beacon までの距離（mm）	
5	1	uint8_t	ビット 0：1 = 距離が適用不可 ビット 1…7：予約済み（0）	

2.1.4.2 リアルタイムタイムスタンプ付きの候補データを含む生距離パケット、データコード 0x0094（ファームウェア V8.431+）

このパケットは、新しい座標が計測されるか計測に失敗するたびに、モバイル Beacon の座標パケットの後に送信されます。

設定の「Interfaces」セクションで「extra raw distances data」オプションが有効になっている場合にのみ使用可能です。

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：要望に応じて対応

Super-Modem：要望に応じて対応

Mini-RX（Badge、Helmet など）：UART ケーブルで対応

Mini-TX：非対応

Mini-TX-2：要望に応じて対応

Modem HW4.9：非対応

Beacon HW4.9：非対応

Beacon HW4.5：非対応

タイムスタンプに関する注意事項を参照してください。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0094
4	1	uint8_t	送信データのバイト数	
5	80		データパケット（以下参照）	
85	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Hedgehog のアドレス	
1	17		距離アイテム 1	
18	17		距離アイテム 2	
35	17		距離アイテム 3	
52	17		距離アイテム 4	
69	8	int64_t	タイムスタンプ – ビーコンの超音波送信の Unix 時刻、1970.01.01 00:00:00 からのミリ秒数。Modem およびすべてのデバイスとの時刻同期、Dashboard により同期。	
77	2	uint16_t	超音波送信から現在時刻までの経過	

			時間、ミリ秒（V5.89+）	
79	1	uint8_t	予約済み	

距離アイテムのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	ビーコンのアドレス（アイテムが未入力の場合は 0）	
1	1	uint8_t	距離候補の数	
2	4	uint32_t	ビーコンまでの距離の第 1 候補、mm	
6	1	uint8_t	第 1 候補の品質、%	
7	4	uint32_t	ビーコンまでの距離の第 2 候補、mm	
11	1	uint8_t	第 2 候補の品質、%	
12	4	uint32_t	ビーコンまでの距離の第 3 候補、mm	
16	1	uint8_t	第 3 候補の品質、%	

2.1.4.3 生距離データのパケット、データコード 0x0004

廃止されたパケットです。0x0084 に置き換えられました。廃止されたパケットの詳細はこちらをご参照ください。

2.1.5 処理済み IMU データのパケット

このパケットは、新しい慣性センサーデータが利用可能になったときに送信されます。

2.1.5.1 リアルタイムタイムスタンプ付き処理済み IMU データのパケット、データコード 0x0085（ファームウェア V7.200 以降）

このパケットは、新しい慣性センサーデータが利用可能になったときに送信されます。

対応ハードウェア:

Super-Beacon：対応、100 Hz（「Processed IMU data」が有効な場合）

Industrial Super-Beacon：対応、100 Hz（「Processed IMU data」が有効な場合）
Modem HW5.1：対応、システム更新レート（「IMU via modem」が有効な場合）

Super-Modem：対応、システム更新レート（「IMU via modem」が有効な場合）

Mini-RX（Badge、Helmet など）：対応、100 Hz（「Processed IMU data」が有効な場合）

Mini-Rx 用 UART ケーブルを使用

Mini-TX：現在の HW バージョンでは非対応
Mini-TX-2：対応、100 Hz（「Processed IMU data」が有効な場合）
Modem HW4.9：対応、システム更新レート（「IMU via modem」が有効な場合）

Beacon HW4.9：対応、100 Hz（「Processed IMU data」が有効な場合）
Beacon HW4.5：対応、100 Hz（「Processed IMU data」が有効な場合）

タイムスタンプに関する注意事項を参照してください。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0085
4	1	uint8_t	送信データのバイト数	
5	46		データパケット（下記参照）	
51	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	4	int32_t	Beacon の座標 X（フュージョン）、mm	
4	4	int32_t	Beacon の座標 Y（フュージョン）、mm	
8	4	int32_t	Beacon の座標 Z（フュージョン）、mm	
12	2	int16_t	回転クォータニオンの W フィールド	

			（角度）	
14	2	int16_t	回転クォータニオンの X フィールド（角度）	
16	2	int16_t	回転クォータニオンの Y フィールド（角度）	
18	2	int16_t	回転クォータニオンの Z フィールド（角度）	
20	2	int16_t	ビーコンの速度 X（フュージョン）、mm/s	
22	2	int16_t	ビーコンの速度 Y（フュージョン）、mm/s	
24	2	int16_t	ビーコンの速度 Z（フュージョン）、mm/s	
26	2	int16_t	ビーコンの加速度 X、mm/s2	
28	2	int16_t	ビーコンの加速度 Y、mm/s2	
30	2	int16_t	ビーコンの加速度 Z、mm/s2	
32	1	uint8_t	ビーコンのアドレス	
33	1	1 バイト	予約済み（0）	
34	8	int64_t	タイムスタンプ – Unix タイム - 1970.01.01 00:00:00 からのミリ秒数。すべてのデバイスが Modem および Dashboard と同期する時刻。	
42	1	uint8_t	フラグ: ビット 0：1 = 位置データ なし ビット 1：1 = クォータニオンデータ なし ビット 2：1 = 速度データ なし ビット 3：1 = 加速度データ なし	
43	3	3 バイト	予約済み（0）	

注：クォータニオンは 10000 の値に正規化されています。

2.1.5.2 処理済み IMU データのパケット、データコード 0x0005

廃止されたパケットで、0x0085 に置き換えられました。廃止されたパケットの詳細はこちらをご参照ください。

2.1.6 テレメトリデータのパケット（データコード 0x0006）

このパケットは、設定の「Interfaces」セクションで「Telemetry stream」オプションが有効になっている場合、位置情報更新後に送信されます。

対応ハードウェア:

Super-Beacon: サポート対象
Industrial Super-Beacon: サポート対象
Modem HW5.1: サポート対象（ファームウェア V7.000+）
Super-Modem: サポート対象（ファームウェア V7.000+）
Mini-RX（Badge、Helmet など）: UART ケーブルを使用してサポート対象
Mini-TX: 現在の HW バージョンではサポート対象外
Mini-TX-2: UART ケーブルを使用してサポート対象
Modem HW4.9: サポート対象外
Beacon HW4.9: サポート対象
Beacon HW4.5: サポート対象

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0006
4	1	uint8_t	送信データのバイト数	
5	16		データパケット（下記参照）	
21	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	2	uint16_t	バッテリー電圧、mV	
2	1	int8_t	RSSI、dBm	
3	1	uint8_t	ビーコンのアドレス	
4	12		予約済み（0）	

2.1.7 品質および拡張位置データのパケット（データコード 0x0007）

このパケットは、設定の「Interfaces」セクションで「Quality and extended location data」オプションが有効になっている場合、位置情報の更新後に送信されます。

対応ハードウェア:

Super-Beacon：対応
Industrial Super-Beacon：対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：UART ケーブルを使用して対応
Mini-TX：現在の HW バージョンでは非対応
Mini-TX-2：UART ケーブルを使用して対応
Modem HW4.9：対応（品質フィールドのみ）
Beacon HW4.9：対応（品質フィールドのみ）
Beacon HW4.5：対応（品質フィールドのみ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0007
4	1	uint8_t	送信データのバイト数	
5	16		データパケット（下記参照）	
21	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスアドレス	
1	1	uint8_t	測位品質、%	
2	1	uint8_t	0 = ジオフェンシングゾーンアラームなし 1...255 - ジオフェンシングゾーンのインデックス このフィールドには MMMSW0005 ライセンスが必要です。	
3	13		予約済み（0）	

2.1.8 廃止された機能

ここでは、旧バージョンのソフトウェアで使用され、デフォルトで他のパケットに置き換えられた廃止パケットについて説明します。

2.1.8.1 モバイルビーコンの cm 分解能座標を含むパケット

廃止パケット。0x0081 に置き換えられました。

対応ハードウェア:

Super-Beacon：対応
Industrial Super-Beacon：対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：UART ケーブルで対応
Mini-TX：現在の HW バージョンでは非対応
Mini-TX-2：UART ケーブルでサポート
Modem HW4.9：サポート対応
Beacon HW4.9：サポート対応
Beacon HW4.5：サポート対応

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	宛先アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0001
4	1	uint8_t	送信データのバイト数	0x10
5	4	uint32_t	タイムスタンプ – 最新のウェイクアップイベントからの経過時間（ミリ秒単位）による Beacon の超音波送信内部時刻。注記を参照。	
9	2	int16_t	Beacon の X 座標（cm）	
11	2	int16_t	Beacon の Y 座標（cm）	
13	2	int16_t	Z 座標（Beacon の高さ）（cm）	
15	1	uint8_t	フラグバイト: ビット 0：1 - 座標が利用不可。X、Y、Z フィールドのデータは使用しないこと。 ビット 1：タイムスタンプ単位インジケーター（注記を参照） ビット 2：1 - ユーザーボタンが押されている（V5.23 以降） ビット 3：1 - ユーザーデバイスへのアップロード用データが利用可能（セクション 2 を参照）（V5.34 以降） ビット 4：1 - ユーザーデバイスからのデータダウンロードを要求（セクション 2 を参照）（V5.34 以降） ビット 5：1 - 第 2 ユーザーボタンが押されている（V5.74 以降） ビット 6：1 - 別の Hedgehog 向けデータ（本パケットを送信している Hedgehog と同一でないもの）	

			ビット 7: 予約済み（0）	
16	1	uint8_t	Hedgehog のアドレス	
17	2	uint16_t	ビット 0…11: XY 平面における Hedgehog ペアの向き、デシ度（0…3600） ビット 12: 1 – 座標はビーコンペアの中心に対して指定; 0 – 指定されたビーコンに対する座標 ビット 13: 1 – 向きは適用不可 ビット 14…15: 予約済み（0）	
19	2	uint16_t	超音波放射から現在時刻までの経過時間、ミリ秒（V5.88+）	
21	2	uint16_t	CRC-16（付録 1 参照）	

2.1.8.2 モバイルビーコンの mm 分解能座標パケット

廃止済みパケット。0x0081 に置き換えられました。

対応ハードウェア:

Super-Beacon: 対応

Industrial Super-Beacon: 対応

Modem HW5.1: 対応

Super-Modem: 対応

Mini-Rx（Badge、Helmet など）: UART ケーブルを使用して対応

Mini-TX: 現在の HW バージョンでは非対応

Mini-TX-2: UART ケーブルを使用して対応

Modem HW4.9：サポート済み

Beacon HW4.9：サポート済み

Beacon HW4.5：サポート済み

タイムスタンプに関する注意事項を参照してください。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	宛先アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0011
4	1	uint8_t	送信データのバイト数	N
5	4	uint32_t	タイムスタンプ – ビーコンの超音波送信の内部時刻。最新のウェイクアップイベントからのミリ秒単位の時刻。注記を参照。	
9	4	int32_t	ビーコンの X 座標、mm	
13	4	int32_t	ビーコンの Y 座標、mm	
17	4	int32_t	ビーコンの Z 座標（高さ）、mm	
21	1	uint8_t	フラグのバイト: ビット 0: 1 – 座標取得不可。X、Y、Z フィールドのデータは使用不可。 ビット 1: タイムスタンプ単位インジケーター（注記参照） ビット 2: 1 – ユーザーボタンが押されている（V5.23+） ビット 3: 1 – ユーザーデバイスへのアップロード用データが利用可能。セクション 2 を参照（V5.34+） ビット 4: 1 – ユーザーデバイスからのデータダウンロードを要求。セクション 2 を参照（V5.34+） ビット 5: 1 – 第 2 ユーザーボタンが押されている（V5.74+） ビット 6: 1 – 別の Hedgehog 向けのデータ（このパケットを送信している Hedgehog とは異な	

			る) ビット 7：1 – ジオフェンシングゾーン外	
22	1	uint8_t	Hedgehog のアドレス	
23	2	uint16_t	ビット 0 ~ 11：XY 平面における Hedgehog ペアの向き、デシ度単位（0 ~ 3600） ビット 12：1 – ビーコンペアの中心に座標が指定されている；0 – 指定された Hedgehog の座標 ビット 13：1 – 向きは適用不可 ビット 14 ~ 15：予約済み（0）	
25	2	uint16_t	超音波放射から現在時刻までの経過時間、ミリ秒単位（V5.88 以降）	
27	M= N-22		オプションデータ – リストを参照	
27+M	2	uint16_t	CRC-16（付録 1 を参照）	

注意：V5.20 より前のファームウェアバージョンでは、タイムスタンプは 1/64 秒単位であり、タイムスタンプ単位インジケーター（flags バイトのビット 1）は 0 です。バージョン 5.20 以降では、タイムスタンプはミリ秒単位であり、タイムスタンプ単位インジケーターは 1 です。

2.1.8.3 固定ビーコンの cm 解像度座標を含むパケット、データコード 0x0002。

廃止されたパケット。0x0012 に置き換えられました。

対応ハードウェア:

Super-Beacon：対応
Industrial Super-Beacon：対応
Modem HW5.1：対応
Super-Modem：サポート済み
Mini-RX（Badge、Helmet など）：UART ケーブルでサポート
Mini-TX：現在の HW バージョンではサポートされていません
Mini-TX-2：UART ケーブルでサポート
Modem HW4.9：サポート済み
Beacon HW4.9：サポート済み
Beacon HW4.5：サポート済み

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0002
4	1	uint8_t	送信データのバイト数	1+N*8
5	1	uint8_t	パケット内のビーコン数	N
6	1	N*8 バイト	N 個のビーコンのデータ	
6+N*8	2	uint16_t	CRC-16（付録 1 参照）	

N 個の各ビーコンのデータ構造フォーマット:

オフセット	サイズ（バイト）	タイプ	説明
0	1	uint8_t	ビーコンのアドレス
1	2	int16_t	ビーコンの X 座標（cm）
3	2	int16_t	ビーコンの Y 座標（cm）
5	2	int16_t	Z 座標、ビーコンの高さ（cm）
7	1	uint8_t	予約済み（0）

2.1.8.4 生慣性センサーデータのパケット、データコード 0x0003

廃止されたパケット。0x0083 に置き換えられました。

対応ハードウェア:

Super-Beacon：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

Industrial Super-Beacon：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

Modem HW5.1：対応、システム更新レート（「IMU via modem」が有効な場合）

Super-Modem：対応、システム更新レート（「IMU via modem」が有効な場合）

Mini-RX（Badge、Helmet など）：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

UART ケーブル使用時

Mini-TX：現在の HW バージョンでは非対応

Mini-TX-2：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

Modem HW4.9：対応、システム更新レート（「IMU via modem」が有効な場合）

Beacon HW4.9：対応、100 Hz（「Raw inertial sensors data」が有効な場合）

Beacon HW4.5：サポート済み、100 Hz（「Raw inertial sensors data」が有効な場合）

タイムスタンプに関する注意事項を参照してください。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0003
4	1	uint8_t	送信データのバイト数	
5	32		データパケット（下記参照）	
37	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	2	int16_t	加速度計、X 軸、1 mg/LSB	
2	2	int16_t	加速度計、Y 軸、1 mg/LSB	
4	2	int16_t	加速度計、Z 軸、1 mg/LSB	
6	2	int16_t	ジャイロスコープ、X 軸、0.0175 dps/LSB	
8	2	int16_t	ジャイロスコープ、Y 軸、0.0175 dps/LSB	

10	2	int16_t	ジャイロスコープ、Z 軸、0.0175 dps/LSB	
12	2	int16_t	コンパス、X 軸、1100 LSB/Gauss	
14	2	int16_t	コンパス、Y 軸、1100 LSB/Gauss	
16	2	int16_t	コンパス、Z 軸、980 LSB/Gauss	
18	1	uint8_t	ビーコンのアドレス	
19	5	5 バイト	予約済み（0）	
24	4	uint32_t	タイムスタンプ、ms	
28	1	uint8_t	フラグ： ビット 0：1 = 加速度計データ n/a ビット 1：1 = ジャイロスコープデータ n/a ビット 2：1 = コンパスデータ n/a ビット 3...7 – 予約済み（0）	
29	3	3 バイト	予約済み	

注：コンパスデータは、IMU 搭載の HW v4.9 ビーコンでのみ利用可能です。

2.1.8.5 生距離データのパケット、データコード 0x0004

廃止されたパケット。0x0084 に置き換えられました。

対応ハードウェア：

Super-Beacon：対応
Industrial Super-Beacon：対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmets など）：UART ケーブルを使用して対応
Mini-TX：現在の HW バージョンでは非対応
Mini-TX-2：UART ケーブルを使用して対応
Modem HW4.9：対応
Beacon HW4.9：対応
Beacon HW4.5：対応

タイムスタンプに関する注意事項をご参照ください。

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0004
4	1	uint8_t	送信データのバイト数	
5	32		データパケット（下記参照）	

37	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Hedgehog のアドレス	
1	6		距離アイテム 1	
7	6		距離アイテム 2	
13	6		距離アイテム 3	
19	6		距離アイテム 4	
25	4	uint32_t	タイムスタンプ – 最新のウェイクアップイベント時点からのミリ秒単位による、ビーコン超音波放射の内部時刻 (V5.89+)	
29	2	uint16_t	超音波放射から現在時刻までの経過時間、ミリ秒 (V5.89+)	
31	1	uint8_t	予約済み	

距離アイテムのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	ビーコンのアドレス（アイテムが未入力の場合は 0）	
1	4	uint32_t	ビーコンまでの距離、mm	
5	1	uint8_t	ビット 0：1 = 距離は適用外 ビット 1…7：予約済み（0）	

2.1.8.6 処理済み IMU データのパケット、データコード 0x0005

廃止されたパケット。0x0085 に置き換えられました。

対応ハードウェア:

Super-Beacon：角度対応、100 Hz（「Processed IMU data」が有効な場合）

Industrial Super-Beacon：角度対応、100 Hz（「Processed IMU data」が有効な場合）

Modem HW5.1：対応、システム更新レート（「IMU via modem」が有効な場合）

Super-Modem：対応、システム更新レート（「IMU via modem」が有効な場合）

Mini-RX（Badge など）：角度対応、100 Hz（「Processed IMU data」が有効な場合）

Mini-RX 用 UART ケーブルを使用

Mini-TX：現在の HW バージョンでは非対応

Mini-TX-2：角度対応、100 Hz（「Processed IMU data」が有効な場合）

Modem HW4.9：対応、システム更新レート（「IMU via modem」が有効な場合）

Beacon HW4.9：対応、100 Hz（「Processed IMU data」が有効な場合）

Beacon HW4.5：対応、100 Hz（「Processed IMU data」が有効な場合）

タイムスタンプに関する注記をご参照ください。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0005
4	1	uint8_t	送信データのバイト数	
5	42		データパケット（下記参照）	
47	2	uint16_t	CRC-16（付録 1 参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	4	int32_t	ビーコンの X 座標（フュージョン）、mm	
4	4	int32_t	ビーコンの Y 座標（フュージョン）、mm	
8	4	int32_t	ビーコンの Z 座標（フュージョン）、mm	
12	2	int16_t	回転クォータニオンの W フィール	

			ド（角度）	
14	2	int16_t	回転クォータニオンの X フィールド（角度）	
16	2	int16_t	回転クォータニオンの Y フィールド（角度）	
18	2	int16_t	回転クォータニオンの Z フィールド（角度）	
20	2	int16_t	ビーコンの X 方向速度（フュージョン）、mm/s	
22	2	int16_t	Beacon の Y 方向速度（フュージョン）、mm/s	
24	2	int16_t	Beacon の Z 方向速度（フュージョン）、mm/s	
26	2	int16_t	Beacon の X 方向加速度、mm/s2	
28	2	int16_t	Beacon の Y 方向加速度、mm/s2	
30	2	int16_t	Beacon の Z 方向加速度、mm/s2	
32	1	uint8_t	Beacon のアドレス	
33	1	1 byte	予約済み（0）	
34	4	uint32_t	タイムスタンプ、ms	
38	1	uint8_t	フラグ： ビット 0：1 = 位置データ利用不可 ビット 1：1 = クォータニオンデータ利用不可 ビット 2: 1 = 速度データ n/a ビット 3: 1 = 加速度データ n/a ビット 4...7 – 予約済み (0)	
39	3	3 バイト	予約済み (0)	

注意: クォータニオンは 10000 の値に正規化されています

2.2 ユーザーデバイスへのデータ読み書きプロトコル

2.2.1 ユーザーデバイスからのデータ送信

対応ハードウェア:

Super-Beacon: 対応

Industrial Super-Beacon: 対応

Modem HW5.1: 対応

Super-Modem: 対応

Mini-RX (Badge、Helmet 等): UART ケーブルを使用して対応

Mini-TX: 現在の HW バージョンでは非対応

Mini-TX-2: UART ケーブルを使用して対応

Modem HW4.9: 非対応

Beacon HW4.9: 対応

Beacon HW4.5: 対応

ユーザーデバイスが Marvelmind システムを介してデータを送信する必要がある場合、以下のフレームを送信する必要があります:

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	宛先アドレス	0x00
1	1	uint8_t	パケットのタイプ	0x49
2	2	uint16_t	パケット内のデータコード	0x0200
4	1	uint8_t	送信データのバイト数	N
5	N	N バイト	ペイロードデータ	
5+N	2	uint16_t	CRC-16（付録 1 参照）	

データは、Hedgehog のダッシュボード設定の「Interfaces」セクションにある「User payload data size」で定義されたサイズの部分に分割され、無線経由で Modem に送信されます。これらの部分の送信レートは、Hedgehog のアップデートレート Hedgehog のバッファサイズは 128 バイトです。バッファのオーバーフローを防ぐため、この点に注意してください。

2.2.2 ユーザーデバイスへのデータ書き込み

このパケットは、Marvelmind デバイス（Modem またはモバイル Beacon）からユーザーデバイスへ送信されます。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	宛先アドレス	0xff
1	1	uint8_t	パケットタイプ	0x4a
2	2	uint16_t	パケット内データコード	0x0200... 0x02ff
4	1	uint8_t	送信データのバイト数	N
5	N	N バイト	ペイロードデータ	
5+N	2	uint16_t	CRC-16（付録 1 参照）	

このコマンドでは、0x200 ~ 0x2ff のデータコードが予約済みです。

ユーザーデバイスがリクエストの処理に成功した場合、以下の形式でレスポンスを送信する必要があります:

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Hedgehog のアドレス（ストリーミングの 0x0001 または 0x0011 パケットから取得可能）	
1	1	uint8_t	パケットのタイプ	0x4a
2	2	uint16_t	パケット内データのコード	0x0200... 0x02ff
4	2	uint16_t	CRC-16（付録 1 参照）	

ユーザーデバイスがリクエストの処理に失敗した場合、以下の形式でレスポンスを送信します:

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Hedgehog のアドレス（ストリーミングの 0x0001 パケットから取得）	
1	1	uint8_t	パケットのタイプ	0xca
2	2	uint16_t	要求されたデータのコード	0x0200... 0x02ff
4	1	uint8_t	エラーコード（注記参照）	1
5	2	uint16_t	CRC-16（付録 1 参照）	

以下のセクションでは、特定のデータ書き込みリクエストについて説明します。

注記：ユーザーデバイスが Hedgehog からのリクエストを処理できない場合、以下のいずれかのエラーコードを含む返答を送信する必要があります:

1 - リクエスト内の「パケットタイプ」フィールドが不明
2 - リクエスト内の「データコード」フィールドが不明
3 - リクエスト内のペイロードデータが不正
6 - デバイスがビジー状態のため、現在要求されたデータを取得できない

2.2.2.1 移動経路の書き込みリクエスト

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：要望に応じて対応

Super-Modem：要望に応じて対応

Mini-RX（Badge、Helmet 等）：UART ケーブルを使用して対応

Mini-TX：現在の HW バージョンではサポートされていません

Mini-TX-2：UART ケーブルでサポートされています

Modem HW4.9：サポートされていません

Beacon HW4.9：サポートされています

Beacon HW4.5：サポートされています

このパケットには、基本動作の 1 つのコマンドが含まれています。Marvelmind デバイスは、パス内の基本動作に関するすべてのコマンドを順番に送信します。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	宛先アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x4a
2	2	uint16_t	パケット内のデータコード	0x201
4	1	uint8_t	送信データのバイト数	0x0c
5	12	12 バイト	ペイロードデータ	
17	2	uint16_t	CRC-16（付録 1 参照）	

ペイロードデータのフォーマット:

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	基本動作のタイプ: 0 - 前進 1 - 後退 2 - 右回転（時計回り） 3 - 左回転（反時計回り） 4 - 一時停止 5 - プログラムを最初から繰り返す 6 - 指定ポイントへ移動する 7 - 速度を設定する	
1	1	uint8_t	この基本動作のインデックス（0 が最初）	
2	1	uint8_t	基本動作の総数	
3	2	int16_t	動作のパラメータ: タイプ 0; 1 - 移動距離、cm タイプ 2; 3 - 回転角度、度 タイプ 4: 一時停止時間、ms タイプ 6: X 目標座標、cm タイプ 7: 速度、%	

5	2	int16_t	動作のパラメータ： タイプ 6: Y 目標座標、cm	
7	2	int16_t	動作のパラメータ： タイプ 6: Z ターゲット座標、cm	
9	3	3 バイト	予約済み（0）	

2.2.2.2 ゾーン書き込みリクエスト

対応ハードウェア:

Super-Beacon：対応
Industrial Super-Beacon：対応
Modem HW5.1：要望に応じて対応
Super-Modem：要望に応じて対応
Mini-RX（Badge、Helmet 等）：UART ケーブルを使用して対応
Mini-TX：現在の HW バージョンでは非対応
Mini-TX-2：UART ケーブルを使用して対応
Modem HW4.9：非対応
Beacon HW4.9：対応
Beacon HW4.5：対応

このパケットには、ゾーンリストのシーケンスの 1 つのアイテムが含まれています。
Marvelmind デバイスは、ゾーンリストのすべてのコマンドを順次送信します。

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	宛先アドレス	0xff
1	1	uint8_t	パケットの種類	0x4a
2	2	uint16_t	パケット内データのコード	0x202
4	1	uint8_t	送信データのバイト数	0x25
5	37	37 bytes	ペイロードデータ	
42	2	uint16_t	CRC-16（付録 1 参照）	

ペイロードデータのフォーマット:

オフセット	サイズ（bytes）	タイプ	説明	値
0	1	uint8_t	ゾーンのインデックス	
1	1	uint8_t	ゾーンポリゴンの頂点数（N）	
2	1	uint8_t	このパケット内の最初の頂点のインデックス：M=0…N-1	
3	1	uint8_t	フラグ: ビット 0：1 = サービス不可ゾーン ビット 1：1 = 走行不可ゾーン ビット 2：1 = 反転ゾーン ビット 3：1 = アクティブゾーン ビット 4…7：予約済み（0）	
4	1	uint8_t	ゾーン数	
5	32	4x8 バイト	ゾーンポリゴンの最大 4 頂点（下記参照）	

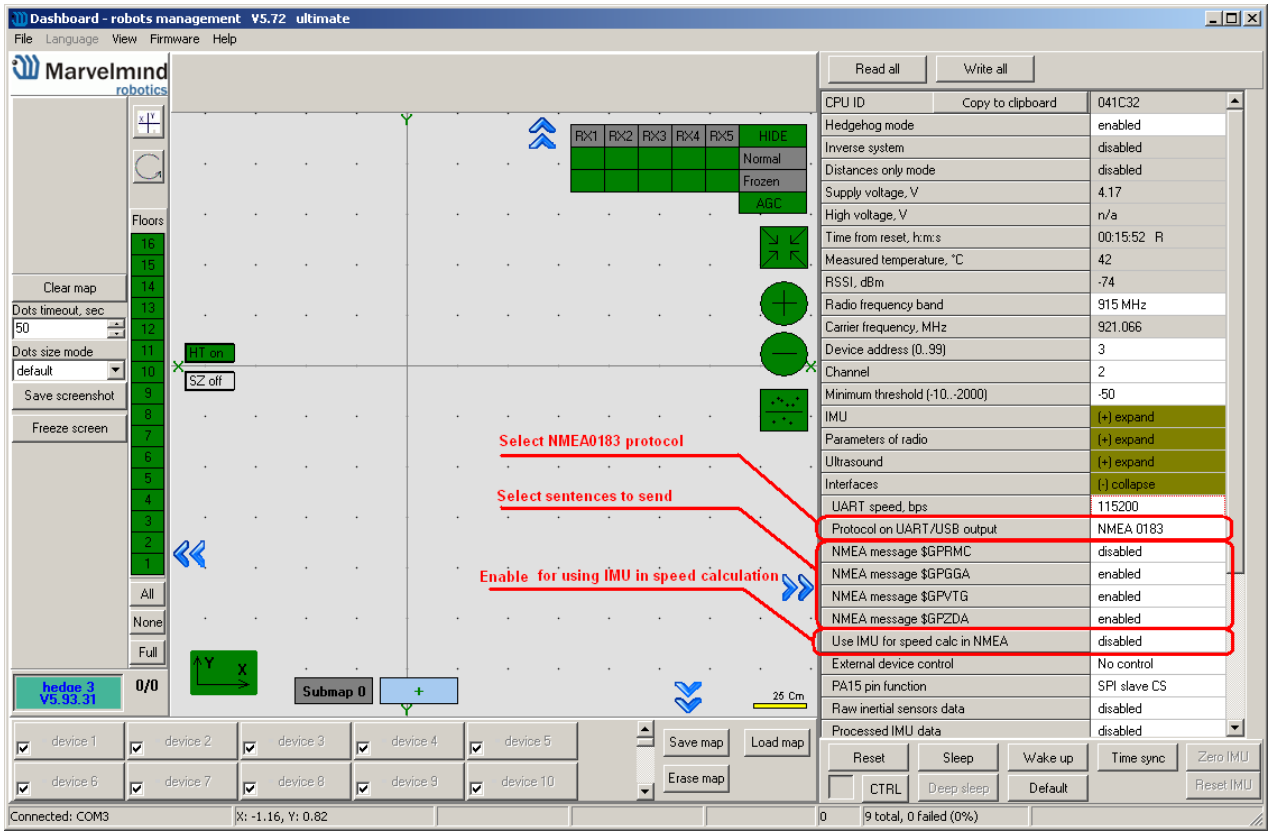
ペイロードデータのフォーマット:

オフセット	サイズ（バイト）	タイプ	説明	値

ト				
0	4	int32_t	ポイントのX座標（mm）	
4	4	int32_t	ポイントのY座標（mm）	

2.3 NMEA0183 通信プロトコル

モバイルビーコンは、UART および USB（仮想 UART）インターフェースを介して、一部の NMEA0183 センテンスを出力することができます。以下のスクリーンショットに示すように、Dashboard で NMEA プロトコルをデバイスで有効にする必要があります:



デバイスは、位置情報の更新を受信するたびに、有効化されたすべてのメッセージを送信します。

モバイルビーコン（Hedgehog）から NMEA データを取得するには、以下のいずれかのインターフェースを使用して外部デバイス（ロボット、コプター、AGV など）に接続する必要があります:

1. CDC クラスの USB デバイス（Windows では COM ポート、Linux では ttyUSB または ttyACM）として USB ホストに接続します。Windows では、Modem 用と同じドライバーが必要です。Linux では、必要なドライバーが Linux カーネルに統合されているため、ほとんどの場合ドライバーは不要です。このインターフェースでは実際の RS-232 を使用しないため、ホスト側で開くシリアルポートのパラメーター（ボーレート、ビット数、パリティなど）は任意に設定できます。
2. Hedgehog のピンに 2 本のワイヤーをはんだ付けすることで、Hedgehog の UART に接続します。以下のビーコンインターフェースの画像を参照してください。位置データを出力するには、GND と USART2_TX の 2 本のワイヤーを接続するだけで十分です。UART トランスミッターのロジックレベルは CMOS 3.3V です。デフォルトのボーレートは 500 kbps で、Dashboard から以下のリストより設定可能です（上記画像の「UART speed, bps」パラメーターを参照）： 4.8、9.6、19.2、38.4、57.6、115.2、500 kbps。データフォーマット: 8 ビット、パリティなし、ストップビット 1。

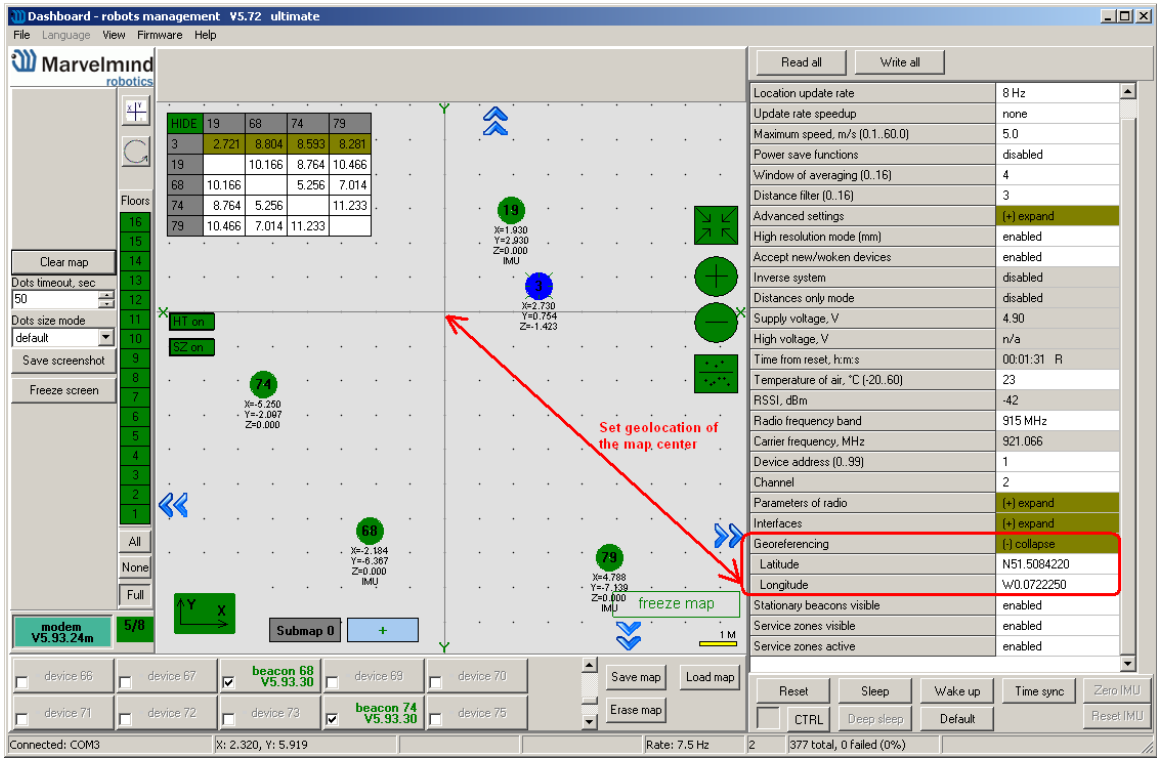
13.11.

2.3.1 座標変換の一般的な規則

Marvelmind システムは、直交デカルト座標系（X、Y、Z）の形式で位置を計測します。Z はほとんどの場合、高度を表します。GPS 座標への変換には、以下の規則が使用されます:

- Z 軸は上方向を向いており、Z 座標は海抜高度を意味します。
- Y 軸は北方向を向いており、Y は Latitude（緯度）を表します。
- X 軸は東方向を向いているため、X は X 座標（経度）を表します。
- 点（X= 0, Y= 0）は、ジオリファレンス点に基づく GPS 座標を持ちます（デフォルト：北緯 0°、西経 0°）。

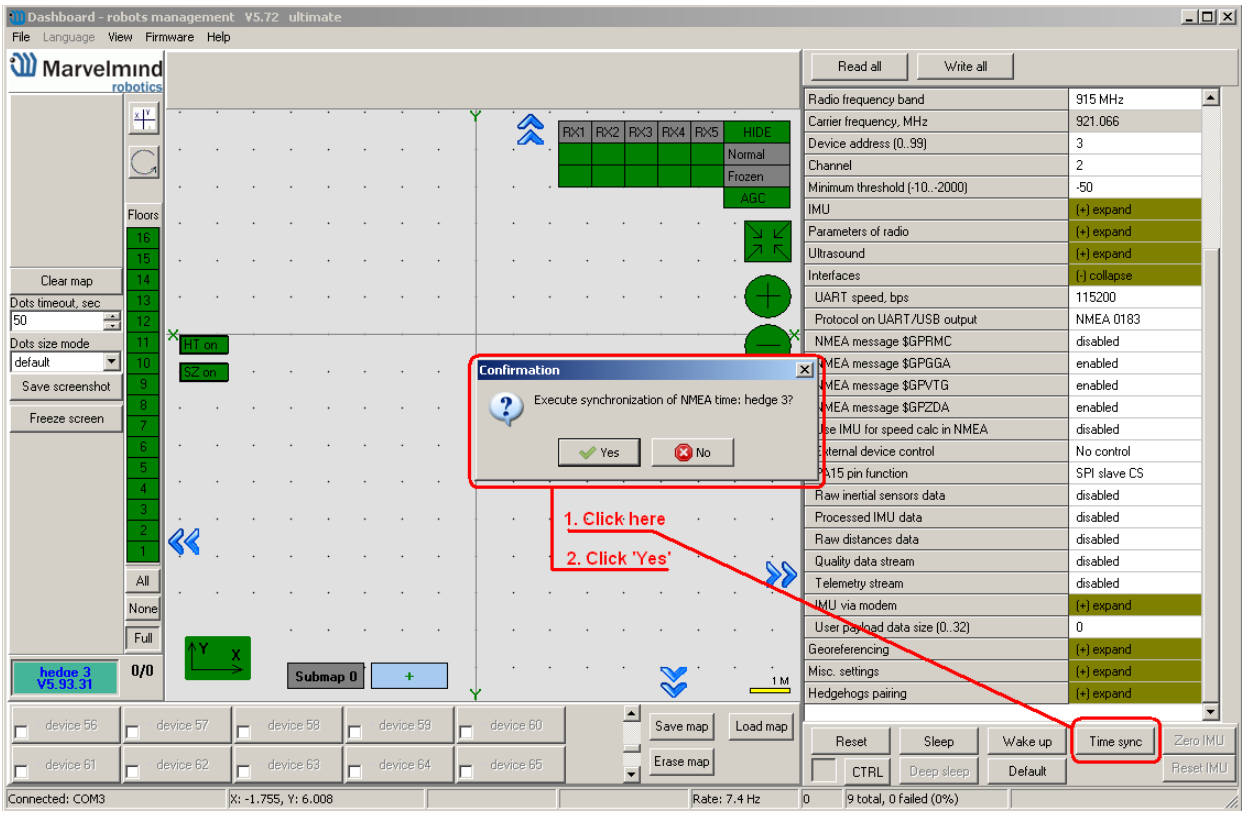
ジオリファレンス座標は、スクリーンショットに示されているように設定できます。



GPS 座標は、指定されたジオリファレンス点および WGS-84 地球モデルに基づいて計算されます。詳細は以下の通りです: Lat= Lat_ref + y*9.013373 ここで、Lat - 緯度（マイクロ度）Lat_ref - ジオリファレンス緯度（マイクロ度）y - Marvelmind システムにおける y 座標（メートル）Long= Long_ref + x*8.98315/cos(Lat_ref/1000000)Long - 経度（マイクロ度）Long_ref - ジオリファレンス経度（マイクロ度）Lat_ref - ジオリファレンス緯度（マイクロ度）x - Marvelmind システムにおける x 座標（メートル）

2.3.2 時刻に関する一般的な取り決め

電源投入後、モバイルビーコンは 2016.08.01 00:00:00 を起点として時刻をカウントします。ユーザーは、以下のスクリーンショットに示されているように、コンピュータの時計と時刻を同期できます。



2.3.3 「NMEA0183」メッセージの実装に関する説明

NMEA 0183 メッセージは、ASCII コードのテキストフレームであり、カンマで区切られた複数の部分で構成され、行末で終了します。行末の前に、各メッセージは「*」記号で終わり、その後に NMEA 0183 標準に基づいて計算されたチェックサムの 2 文字が続きます。
NMEA 0183 メッセージの各部分は、特定のパラメータを表します。
以下に、サポートされているすべてのメッセージおよびパラメータフィールドの説明を示します。
メッセージ形式は、2002 年 1 月 1 日付け NMEA 0183 標準バージョン 3.01 に基づいています。

1. $GPRMC - 推奨最小限の特定 GNSS データ

サポートされているハードウェア:

Super-Beacon：サポート済み

Industrial Super-Beacon：サポート済み

Modem HW5.1：サポート済み（ SW V7.000 以降 ）

Super-Modem：サポート済み（ SW V7.000 以降 ）

Mini-RX（ Badge、Helmet など ）：UART ケーブルを使用してサポート済み

Mini-TX：現在の HW バージョンではサポートされていません

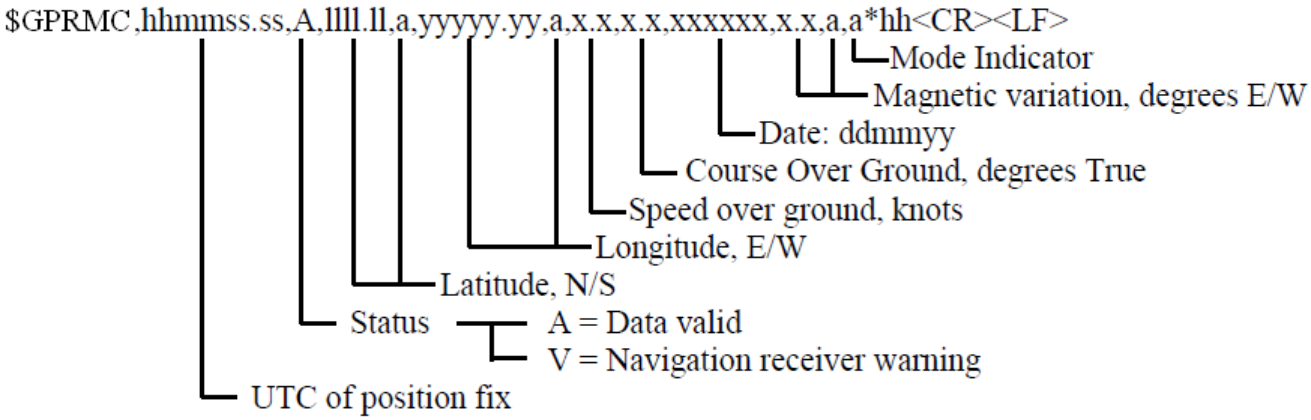
Mini-TX-2：UART ケーブルを使用してサポート済み

Modem HW4.9：非対応

Beacon HW4.9：対応

Beacon HW4.5：対応

NMEA 0183 規格の一般フォーマット:



フィールド実装の説明:

1.1. '$GPRMC' – メッセージタイプの識別子

1.2. 'hhmmss.ss' – UTC 測位時刻

一般的な取り決めに従い、時刻はデフォルトの 2016.01.01 を起点としてカウントされるか、またはコンピュータのクロックと同期されます。

1.3. 'A' – ステータス

最後の測位更新が正常に完了した場合、'A'の値が送信されます。
最後の測位更新においてエラーが発生した場合、'V'の値が送信されます。

1.4. 'llll.llllll,a' – 緯度、N/S

一般的な取り決め（上記参照）に従い、緯度はジオリファレンス位置に対する Y 座標に対応します。緯度は分の小数部 6 桁で表され、2mm 以下の分解能を実現します。

1.5. 'yyyyy.yyyyyy,a' – 経度、E/W

一般的な取り決め（上記参照）に従い、経度はジオリファレンス位置に対するX座標に対応します。経度は分の小数部6桁で表され、2mm以下の分解能を実現します。

1.6. 'x.x' – 対地速度、ノット

Marvelmindシステムは座標を計測し、速度は座標の変化に基づいてフィルタリングを適用しながら算出されます。オプションとして、速度計算にIMUフュージョンを使用することもできます。

1.7. 'xxxxxx' – 日付：ddmmyy

一般的な取り決めに従い、時刻はデフォルトの2016.01.01を起点としてカウントされるか、またはコンピュータのクロックと同期されます。

1.8. 'x.x,a' – 磁気偏差

このパラメータ値は常にnullフィールドです。

1.9. 'a' - モードインジケータ

最後の位置更新が成功した場合、'A'値（自律モード）が送信されます
最後の位置更新でエラーが発生した場合、'N'値（データ無効）が送信されます

2. $GPGGA - Global Positioning System Fix Data

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：対応（SW V7.000以降）

Super-Modem：対応（SW V7.000以降）

Mini-RX（Badge、Helmetなど）：Mini-Rx用UARTケーブルで対応

Mini-TX：現在のHWバージョンでは非対応

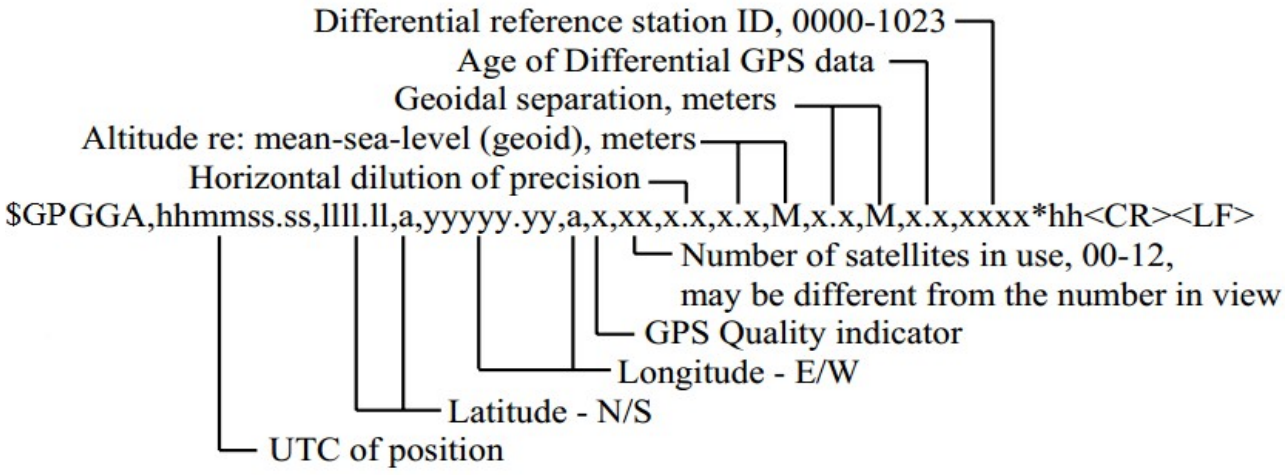
Mini-TX-2：対応

Modem HW4.9：非対応

Beacon HW4.9：対応

Beacon HW4.5：対応

NMEA 0183規格の一般フォーマット:



フィールド実装の説明:

2.1. '$GPGGA' – メッセージタイプの識別子

2.2. 'hhmmss.ss' – UTC位置フィックス

一般的な規定に従い、時刻はデフォルトの2016.01.01から計算されるか、コンピュータの時計と同期されます。

2.3. 'llll.llllll,a' – 緯度、N/S

一般的な規定に従い（上記参照）、緯度はジオリファレンス位置に対するＹ座標に対応します。緯度は分の小数部６桁で表示され、２mm 以下の分解能が得られます。

2.4. 'yyyyy.yyyyyy,a' – 経度、E/W
一般的な規定に従い（上記参照）、経度はジオリファレンス位置に対するＸ座標に対応します。経度は分の小数部６桁で表示され、２mm 以下の分解能が得られます。

2.5. 'x' – GPS 品質インジケータ
直前の位置更新が成功した場合、'1'（GPS SPS モード、有効なフィックス）の値が送信されます。

直前の位置更新でエラーが発生した場合、'0'（フィックス利用不可または無効）の値が送信されます。

2.6. 'xx' – 使用中の衛星数
現在の実装では常に'08'です。

2.7. 'x.x' – 水平精度劣化度（HDOP）
現在の実装では常に'1.2'です。

2.8. 'x.x, M' – 平均海面（ジオイド）を基準とした高度、メートル
これは一般的な規定に従うＺ座標に対応します。

2.9. 'x.x, M' – ジオイド分離、メートル
常に'0.0, M'の値が送信されます。

2.10. 'x.x' – 差分 GPS データの経過時間
このパラメータの値は常に null フィールドであり、DGPS は使用されません。

2.11. 'xxxx' – 差分基準局 ID
このパラメータの値は常に null フィールドです。

3. $GPVTG - 対地針路および対地速度

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：対応（SW V7.000 以降）

Super-Modem：対応（SW V7.000 以降）

Mini-RX（Badge、Helmet など）：Mini-Rx 用 UART ケーブルを使用して対応

Mini-TX：現行 HW バージョンでは非対応

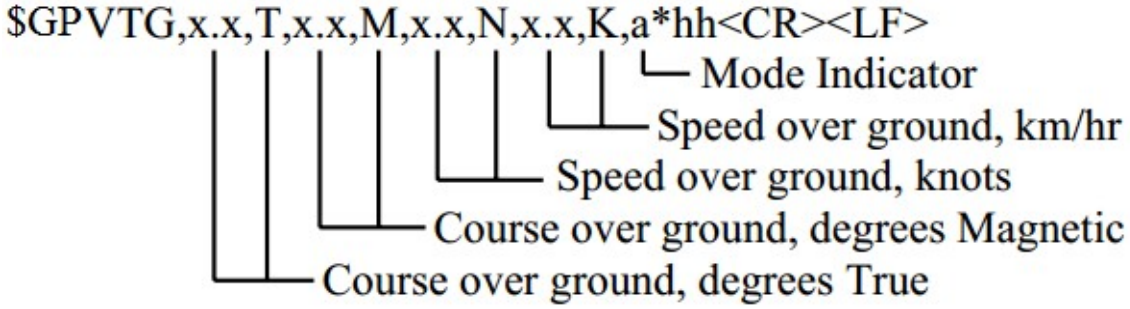
Mini-TX-2：対応

Modem HW4.9：非対応

Beacon HW4.9：対応

Beacon HW4.5：対応

NMEA 0183 規格の一般フォーマット:



フィールド実装の説明:

3.1. '$GPVTG' – メッセージタイプの識別子

3.2. 'x.x, T' – 対地針路、真方位（度）
NMEA 規格によると、針路は速度ベクトルと北方向の間の角度です。上記の一般的な取り決めに示されているように、Y 軸が北方向として使用されます。

3.3. 'x.x, M' – 対地針路、磁気方位（度）
現在の実装では、磁気針路は真針路と同一です。

3.4. 'x.x, N' – 対地速度、ノット
Marvelmind システムは座標を計測し、速度は座標のダイナミクスに一定のフィルタリングを適用することで算出されます。オプションとして、速度計算に IMU フュージョンを使用することができます。

3.5. 'x.x, K' – 対地速度、km/hr
別の単位で表した同じ速度です

3.6. 'a' – モードインジケーター
最後の位置更新が成功した場合、'A'値（自律モード）が送信されます
最後の位置更新でエラーが発生した場合、'N'値（データ無効）が送信されます

4. $GPZDA – 時刻と日付

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：対応（SW V7.000 以降）

Super-Modem：対応（SW V7.000 以降）

Mini-RX（Badge、Helmet など）：UART ケーブルを使用することで対応

Mini-TX：現在の HW バージョンでは非対応

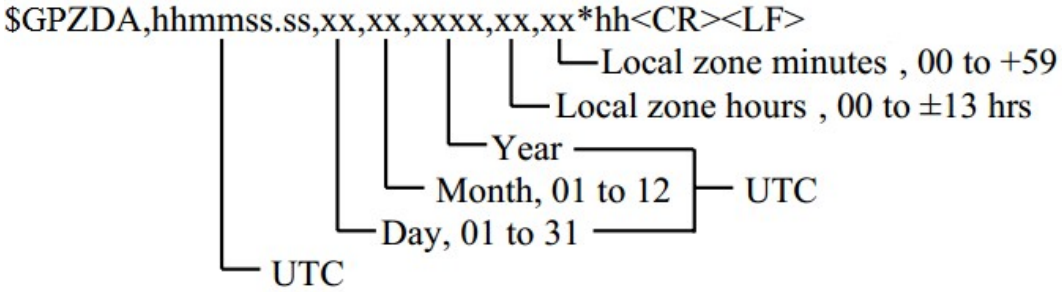
Mini-TX-2：UART ケーブルを使用することで対応

Modem HW4.9：非対応

Beacon HW4.9：対応

Beacon HW4.5：対応

NMEA 0183 標準の一般フォーマット:



一般的な規約に従い、時刻はデフォルトの 2016.01.01 から起算されるか、コンピュータークロックと同期されます。

フィールド実装の説明:

4.1. '$GPZDA' – メッセージタイプの識別子

4.1. 'hhmmss.ss' – UTC

時刻（時、分、秒）。

4.2. 'xx' – 日、01～31

日。

4.3. 'xx' – 月、01 ~ 12

月。

4.4. 'xxxx' – 年

年。

4.4. 'xx' – ローカルゾーン時間

ローカルゾーンは常に「00」時間です。

4.5. 'xx' – ローカルゾーン分

ローカルゾーンは常に「00」分です。

5. $GPHDT – 方位角（Heading）

対応ハードウェア:

Super-Beacon：対応

Industrial Super-Beacon：対応

Modem HW5.1：対応（SW V7.000 以降）

Super-Modem：対応（SW V7.000 以降）

Mini-RX（Badge、ヘルメット等）：UART ケーブルでサポート

Mini-TX：非対応

Mini-TX-2：UART ケーブルでサポート

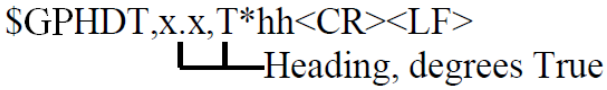
Modem HW4.9：非対応

Beacon HW4.9：非対応

Beacon HW4.5：非対応

このパケットのストリーミングを有効にするには、MMSW0002 ライセンスが必要です。

NMEA 0183 標準の汎用フォーマット:



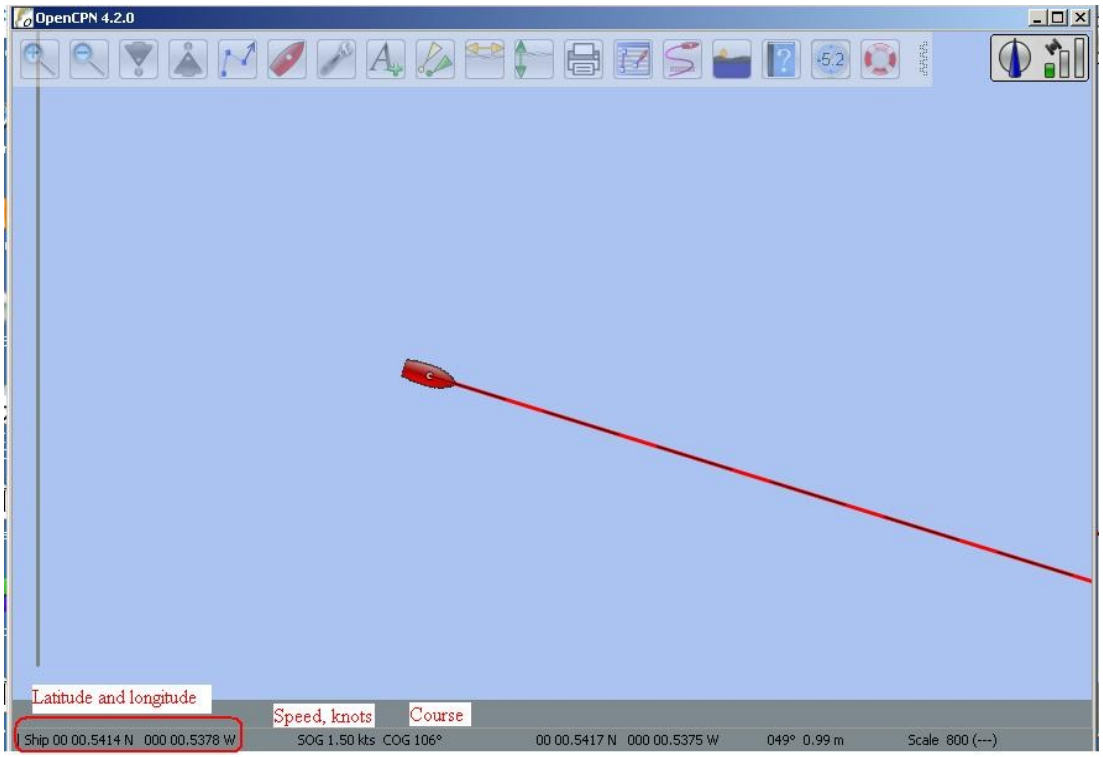
5.1. '$GPHDT' – メッセージタイプの指定

5.2. 'x.x, T' – 真方位の方向角（度）

これは、ペアビーコンまたはペアマイクロフォン機能とジャイロスコープとのフュージョンを使用して算出された方位角です。

2.3.4 NMEA データ受信の例

次のスクリーンショットは、USB（仮想 COM ポート）経由で OpenCPN ソフトウェアに接続されたモバイルビーコンから受信したデータの例であり、MS Windows 上のコンピューターで実行されています。



3. USB（仮想 UART）経由の通信プロトコル

3.1　ストリーミング用'Marvelmind'プロトコル

UART の対応セクションに記載されているすべてのパケットは、USB（仮想 UART）経由でも利用可能です。

これらのデータは、mini-TX および'UART Cable for Mini-Rx'なしの mini-RX でも利用可能です。

Marvelmind デバイスがこのプロトコルに従ったリクエストを受信した場合、ストリーミングは 5 秒間停止されます。

3.2 ユーザーデバイスとのデータ読み書きプロトコル

UART の対応セクションに記載されているすべてのパケットは、USB（仮想 UART）経由でも利用可能です。

これらのデータは、'UART Cable for Mini-Rx'なしの Mini-TX および Mini-RX でも利用可能であることに注意してください。

3.3 NMEA0183 通信プロトコル

UART の対応セクションに記載されているすべてのパケットは、USB（仮想 UART）経由でも利用可能です。

これらのデータは、'UART Cable for Mini-Rx'なしの Mini-TX および Mini-RX でも利用可能であることに注意してください。

Marvelmind デバイスがこのプロトコルによるリクエストを USB から受信した場合、ストリーミングは 5 秒間停止されます。

3.4 USB インターフェースを介した Modem とのデータ交換プロトコル

このプロトコルは、Dashboard ソフトウェアおよび次章に記載されている Marvelmind API で使用されます。

Modem は CDC クラスの USB デバイス（Windows ではバーチャル COM ポート、Linux では ttyUSB または ttyACM）として USB ホストに接続します。

このインターフェースでは実際の RS-232 は使用されないため、ホスト側で設定するシリアルポートのパラメーター（ボーレート、ビット数、パリティなど）は任意の値で構いません。

データはバイナリ形式です。

USB 経由で接続されたデバイスの「ネットワークアドレス」は 0xff です。

マルチバイト数値は下位バイトから送信されます（リトルエンディアン形式）。

3.4.1 最新座標パックの読み取り（ファームウェア V5.13+）

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2:非対応
Modem HW4.9:対応
Beacon HW4.9:非対応
Beacon HW4.5:非対応

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x4110
4	2	uint16_t	アクセスモード	0x0000
6	2	uint16_t	CRC-16（付録 1 参照）	0xc004

応答フレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットタイプ	0x03
2	1	uint8_t	送信データのバイト数	0x64
3	100 (0x64)	100 バイト	データ構造（以下参照）	
103	2	uint16_t	CRC-16（付録 1 参照）	

エラー返信のフォーマットは付録 2 に記載されています。

データフィールドのフォーマット（100 バイト）

オフセット	サイズ（バイト）	説明
0	96 (6*16)	Modem が受信した最新 6 件の座標構造体（以下参照）
96	1	フラグバイト: ビット 0...1：予約済み ビット 2：1 = ユーザーデータ利用可能（セクション 12 参照） ビット 3...7：予約済み
97	3	予約済み

座標構造体のフォーマット（16 バイト）

オフセット	サイズ（バイト）	説明

ト		
0	1	デバイスのアドレス
1	4	X 座標、mm (int32_t)
5	4	Y 座標、mm (int32_t)
9	4	Z 座標、mm (int32_t)
13	1	フラグのバイト: Bit 0: 1 – 有効な座標なし（Dashboard 上の赤色モード） Bit 1: 1 – フリーズされたマップ上の一時的なモバイルビーコン（青色モード） Bit 2: 1 – ビーコンは Hedgehog の測位に使用されている
14	2	予約済み（0）

3.4.2 Modem 設定の読み取り/書き込み

3.4.2.1 Modem 設定の読み取り（ ファームウェア V5.30+ ）

対応ハードウェア:

Super-Beacon: 非対応

Industrial Super-Beacon: 非対応

Modem HW5.1: 対応

Super-Modem: 対応

Mini-RX (Badge、Helmet など): 非対応

Mini-TX: 非対応

Mini-TX-2: 非対応

Modem HW4.9：サポート対象

Beacon HW4.9：サポート対象外

Beacon HW4.5：サポート対象外

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x5000
4	2	uint16_t	アクセスモード	0x0000
6	2	uint16_t	CRC-16（付録 1 参照）	0x0550

応答フレームのフォーマット（Modem からホストへ A へ）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットの種類	0x03
2	1	uint8_t	データ送信のバイト数	0x30
3	0x30	構造体	データ構造（以下参照）	
0x33	2	uint16_t	CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

3.4.2.2 Modem 設定の書き込み

警告！ Modem 設定を書き込むには、まず設定を読み込み、次のセクションで説明するデータフィールドを設定してから書き込みを行ってください。構造体内の他のバイトは変更しないでください。変更すると Modem の動作が低下する可能性があります。

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス S	0xff
1	1	uint8_t	パケットの種類	0x10
2	2	uint16_t	パケット内のデータコード	0x5000
4	2	uint16_t	アクセスモード	0x0000
6	1	uint8_t	データ転送のバイト数	0x30
7	0x30	structure	データ構造（以下を参照）	
0x37	2	uint16_t	CRC-16（付録 1 を参照）	

アンサーフレームのフォーマット（Modem からホストへ H）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	データのコード	0x5000
4	2	uint16_t	予約済み	
6	2	uint16_t	CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

3.4.2.3 Modem 設定データの構造

データ構造の多くのフィールドは説明されていません。これらのフィールドは変更しないでください。これらは Dashboard プログラムからシステムを調整するために使用されており、無断で変更すると Modem の動作が低下する可能性があります。

オフセット	サイズ（バイト）	タイプ	説明
0	20	20 バイト	未説明
20	1	int8_t	気温設定 Vt（符号付き） Temperature is (Vt+23) °C
21	1	uint8_t	マップ座標 X=0、Y=0 を持つべきビーコンのアドレス
22	4	4 バイト	未説明
26	1	uint8_t	マップ座標 X>0、Y=0 を持つべきビーコンのアドレス
27	1	uint8_t	Y>0 のマップ座標を持つべきビーコンのアドレス
28	1	uint8_t	制御フラグ： ビット 0：未説明 ビット 1：1 - モバイルビーコンの移動フィルタリング有効 ビット 2：未説明 ビット 3：1 - 高解像度モード（座標を cm ではなく mm で出力） ビット 4：未説明 ビット 5: 1 = 全マップのミラーリング ビット 6: 1 = 省電力モード（省電力機能はすべてのサブマップがフリーズされている場合にのみ動作します） ビット 7: 未説明
29	2	2 バイト	未説明
31	1	uint8_t	N、Hedgehog 座標の取得最大周波数を決定します F(N)= 2^(N-1) Hz、N= 0…4、 F(5)= 12 Hz、F(6)= 16 Hz、F(7)= 16+（最大）
32	16	16 バイト	未説明

3.4.3 サブマップ設定の読み取り/書き込み

対応ハードウェア:

Super-Beacon: 非対応
Industrial Super-Beacon: 非対応
Modem HW5.1: 対応
Super-Modem: 対応
Mini-RX（Badge、Helmet など）: 非対応
Mini-TX: 非対応
Mini-TX-2: 非対応
Modem HW4.9：サポート対象
Beacon HW4.9：サポート対象外
Beacon HW4.5：サポート対象外

3.4.3.1 サブマップ設定の読み取り（ファームウェア V5.30 以降）

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x60XX（XX はサブマップの番号）
4	2	uint16_t	アクセスモード	0x0000
6	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホストへの Modem）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットの種類	0x03
2	1	uint8_t	データ転送のバイト数	0x50 (80)
3	80	構造体	データ構造（以下参照）	
83	2	uint16_t	CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

3.4.3.2 サブマップ設定の書き込み（ファームウェア V5.30+）

警告! サブマップ設定を書き込むには、設定を読み取り、次のセクションに記載されているデータフィールドを設定してから書き込む必要があります。構造内の他のバイトは変更しないでください。変更すると Modem の動作が低下する可能性があります。

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内のデータコード	0x60XX（XX はサブマップの番号）
4	2	uint16_t	アクセスモード	0x0000
6	1	uint8_t	データ転送のバイト数	0x50 (80)
7	80	構造体	データ構造（以下参照）	
87	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	データのコード	0x5000
4	2	uint16_t	予約済み	
6	2	uint16_t	CRC-16（付録 1 を参照）	

エラー応答のフォーマットは付録 2 に記載されています。

3.4.3.3 サブマップ設定データの構造

データ構造の多くのフィールドは説明されていません。これらのフィールドは変更しないでください！これらは Dashboard プログラムによる調整システムに使用されており、無断で変更すると Modem の動作が低下する可能性があります。

オフセット	サイズ（バイト）	タイプ	説明
0	1	uint8_t	サブマップ構築のための開始ビーコンのアドレス
1	1	uint8_t	コントロールワード: ビット 0：1 - サブマップがフリーズされている（サブマップをフリーズ） ビット 1：1 - ビーコンは Hedgehog より高い位置にある ビット 2...4：未説明 ビット 5：1 - サブマップのミラーリング ビット 6...7：未説明
2	1	uint8_t	距離の制限: ビット 0...6：手動距離制限（ビット 7 = 1 の場合） ビット 7：0 - 自動制限、1 = 手動
3	13	13 バイト	未説明
16	2	int16_t	サブマップの X シフト、cm
18	2	int16_t	サブマップの Y シフト、cm
20	2	uint16_t	サブマップの回転、センチ度
22	58	58 バイト	未説明

3.4.4 デバイスのスリープ/ウェイクアップ

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内データのコード	0xb006
4	2	uint16_t	アクセスモード	ウェイク用: 0x0002 その他：0x0001
6	1	uint8_t	データ送信のバイト数	0x08
7	1	uint8_t	パスワード、バイト 0	0x2d
8	1	uint8_t	パスワード、バイト 1	0x94
9	1	uint8_t	パスワード、バイト 2	0x5e
10	1	uint8_t	パスワード、バイト 3	0x81
11	1	uint8_t	コマンド: 0 – 標準スリープ 1 – ディープスリープ（HW リセットのみで起動） 2 – 標準スリープからの起動 3…255 - 予約済み	0…2
12	3	3 バイト	予約済み	
15	2	uint16_t	CRC-16（付録 1 参照）	

起動コマンドに対する応答フレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ	0x10

2	2	uint16_t	データのコード	0xb006
4	2	uint16_t	予約済み	
6	2	uint16_t	CRC-16（付録参照）	

スリープコマンドに対する応答フレームのフォーマット（Modem からホストへ Er)

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ（Modem 返信）	0x7f
2	2	uint16_t	データコード	0xb006
4	2	uint16_t	予約済み	
6	2	uint16_t	バイト 0…5 の CRC-16（付録 1 参照）	
8	1	uint8_t	デバイスのアドレス	0x01…0xfe
9	1	uint8_t	パケットのタイプ	0x10
10	2	uint16_t	データのコード	0xb006
12	2	uint16_t	予約済み	
14	2	uint16_t	バイト 8…13 の CRC-16（付録参照）	

エラー応答のフォーマットは付録 2 に記載されています。

3.4.5 デバイスアドレスの設定

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内のデータコード	0x0101
4	2	uint16_t	アクセスモード	0x0000
6	1	uint8_t	データ送信のバイト数	0x02
7	1	uint8_t	データ項目のコード（アドレス）	0x00
8	1	uint8_t	デバイスの新しいアドレス	
9	2	uint16_t	CRC-16（付録 1 参照）	

3.4.6 計測済み生距離データの読み取り

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

このコマンドは 2 つのモードでアクセス可能です:

- データコード 0x4000 の場合 – 最新の 8 つの距離を読み取ります。応答フレームには、リクエスト時点から測定された最新の 8 つの距離が含まれます
- データコード 0x4001 の場合 – 全距離をフレームごとに読み取ります。次のリクエストごとの応答フレームには、保存された測定距離の次の 8 つが含まれます。距離テーブルの全データが送信されると、最初から繰り返します

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x4000 または 0x4001
4	2	uint16_t	アクセスモード	0x0000
6	2	uint16_t	CRC-16（付録 1 参照）	

アンサーフレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	1	uint8_t	送信データのバイト数	0x28
3	40 (0x28)	40 バイト	データ構造（下記参照）	
43	2	uint16_t	CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

データフィールドのフォーマット（40 バイト）

オフセット	サイズ（バイト）	説明
0	32 (8*4)	8 つの生距離構造体（下記参照）
32	8	予約済み

距離構造体のフォーマット（4 バイト）

オフセット	サイズ（バイト）	説明
0	1	超音波受信機のアドレス
1	1	超音波送信機のアドレス
2	2	デバイス間の計測距離、mm（uint16_t）

3.4.7　ビーコンの状態読み取り（ファームウェア V5.33 以降）

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x0003
4	2	uint16_t	アクセスモード	0x0002
6	2	uint16_t	CRC-16（付録 1 参照）	

アンサーフレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス	0x01…0xfe
1	1	uint8_t	パケットの種類	0x03
2	1	uint8_t	データ転送のバイト数	0x20
3	32	32 バイト	データ構造（下記参照）	
35	2	uint16_t	CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

データフィールドのフォーマット:

オフセット	サイズ（バイト）	タイプ	説明
0	4	uint32_t	リセットまたはウェイクアップからの動作時間（秒）
4	1	uint8_t	R、無線 RSSI レジスタ値（受信信号強度インジケータ）。 R>128 の場合、RSSI (dBm) = ((R-256)/2) – 74 R<=128 の場合、RSSI (dBm) = (R/2) - 74
5	1	uint8_t	説明なし
6	1	int8_t	Measured temperature Vt (signed). Temperature is (Vt+23) °C
7	2	uint16_t	ビット 0…11：電源電圧、mV ビット 12…13：説明なし ビット 14：1：低電力、デバイスは短時間後にスリープ状態に移行します ビット 15：1：非常に低電力、デバイスは短時間後にディー

			プスリープ状態に移行します
9	23	23 バイト	説明なし

3.4.8 Marvelmind ロボット制御コマンド

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX (Badge、Helmet 等)：非対応
Mini-TX:サポートされていない
Mini-TX-2:サポートされていない
Modem HW4.9:サポートされている
Beacon HW4.9:サポートされていない
Beacon HW4.5:サポートされていない

3.4.8.1 ロボット制御コマンド

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	ロボットのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内のデータコード	0x1000
4	2	uint16_t	アクセスモード	0x0001
6	1	uint8_t	データ転送のバイト数	0x10
7	16 (0x10) バイト	uint8_t	ロボット制御データ（以下参照）	
23	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ（Modem 返答）	0x7f
2	2	uint16_t	データコード	0x1000
4	2	uint16_t	予約済み	
6	2	uint16_t	バイト 0…5 の CRC-16（付録 1 参照）	
8	1	uint8_t	ロボットのアドレス	0x01…0xfe
9	1	uint8_t	パケットのタイプ（ロボット返答）	0x10
10	2	uint16_t	データコード	0x1000
12	2	uint16_t	reserved	
14	2	uint16_t	バイト 8 ~ 13 の CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

ロボット制御データのフォーマット:

オフセット	サイズ	タイプ	説明

	（バイト）		
0	1	uint8_t	制御モード: 0 - 制御なし（待機モード） 1 - モーター電力制御 2 - 速度制御 3 - 移動プログラムの書き込み 4 - 移動プログラムの一時停止 5 - 一時停止後の移動再開
1	1	uint8_t	操作コード: 0 - 前進 1 - 後退 2 - 時計回りに回転 3 - 反時計回りに回転 4 - 指定時間の一時停止（モード 3 用） 5 - 開始から移動プログラムを繰り返す（モード 3 用） 6 - 座標で指定したポイントへ移動（モード 3 用） 7 - 移動速度の設定（モード 3 用）
2	1	uint8_t	制御バイト 1: モード 1 の場合: モーターへの電力供給、% モード 2 の場合: 移動速度、% モード 3 の場合: プログラムステップ番号（ゼロから開始）
3	2	Int16_t	プログラム用データ（モード 3）: 操作コード 0 または 1: 移動距離、cm 操作コード 2 または 3: 回転角度、度 操作コード 4: 一時停止時間、ms 操作コード 6: 移動目標の X 座標、cm 操作コード 7: 移動速度、%
5	1	uint8_t	モード 3 の場合: プログラムの総ステップ数。
6	2	int16_t	プログラム用追加データ（モード 3）: 動作コード 6: 移動目標の Y 座標、cm
8	2	int16_t	動作コード 6: 移動目標の Z 座標、cm
10	6	6 バイト	予約済み（0）

この複雑なコマンドに関するコメント。
ロボット制御構造のバイト 0 に指定された、ロボット制御の 3 つの主要モードがあります:

- パワー制御（モード 1）
- 速度制御（モード 2）
- プログラムによる移動（モード 3）

モード 1 およびモード 2 は、主にテスト目的で使用されます。モード 1 では、ロボットはモータの選択したパワーで前進、後退、左回転、または右回転します。モード 2 では、ロボットは同様の動作を行いますが、選択した速度を維持するようにパワーを調整します。パワーまたは速度は構造体のバイト 2 に設定し、移動の種類はバイト 1 に設定します。

モード 4 およびモード 5 は、プログラム実行中の移動を一時停止し、一時停止後に移動を再開するための特殊コマンドです。

複雑な軌道上を移動するための主要モードはモード 3 です。
このモードでは、ロボットに一連の基本動作をプログラムすることができ、それらを組み合わせることで軌道が構成されます。シーケンスの各項目は、このタイ

プのコマンドを 1 つ送信することで指定します。各コマンドには、ロボット制御構造のバイト 2 に現在のステップ番号、バイト 5 に総ステップ数を含める必要があります。

ロボット制御構造のバイト 1 には、基本移動の種類を指定します。基本移動のパラメータは、「プログラム用データ」フィールド（バイト 3…4）および「プログラム用追加データ」フィールド（バイト 6…7）に指定します。

利用可能な基本動作は以下のとおりです:

- 指定した距離だけ前進する;
- 指定した距離だけ後退する;
- 指定した角度だけ時計回りに回転する;
- 指定した角度だけ反時計回りに回転する;
- 指定した時間だけ一時停止する;
- 移動プログラムを最初の項目から再起動する（ループ動作用）;
- Marvelmind ナビゲーションシステムの座標で指定した点（X, Y）へ移動する;
- 移動速度を変更する。

ロボットはプリミティブのシーケンスを受信した後、プログラムの実行を開始する。プログラムの実行が完了するとロボットは停止する。ただし、プログラムに操作コード 5（先頭から繰り返し）の項目が含まれている場合、停止コマンドを受信するか新しいプログラムがアップロードされるまで、プログラムは無限にループを繰り返す。

3.4.8.2 ロボットを停止させる

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	ロボットのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内データのコード	0x403
4	2	uint16_t	アクセスモード	0x0001
6	1	uint8_t	データ送信のバイト数	0x04
7	4 bytes	4 bytes	予約済み (0)	0
11	2	uint16_t	CRC-16 (付録 1 を参照)	

アンサーフレームのフォーマット (Modem からホストへ)

オフセット	サイズ (bytes)	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ（Modem 応答）	0x7f
2	2	uint16_t	データコード	0x403
4	2	uint16_t	予約済み	
6	2	uint16_t	バイト 0…5 の CRC-16（付録 1 参照）	
8	1	uint8_t	ロボットのアドレス	0x01…0xfe
9	1	uint8_t	パケットのタイプ（ロボット応答）	0x10
10	2	uint16_t	データコード	0x403
12	2	uint16_t	予約済み	
14	2	uint16_t	バイト 8 ~ 13 の CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

このコマンドは、ロボットの移動または移動プログラムの実行を単純に終了させます。ロボットは停止し、新しいコマンドを待機します。

3.4.9 デバイス制御設定の読み取り/書き込み（ファームウェア V6.01 以降）

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9: サポートされていません
Beacon HW4.5: サポートされていません

3.4.9.1 デバイス制御設定の読み取り

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス（beacon/Modem）	0x01…0xfe または 0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x1201
4	2	uint16_t	アクセスモード	0x0001
6	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホスト PC）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットタイプ	リクエストが Modem に送信された場合は 0x03 リクエストが Beacon に送信された場合は 0x7f
2	1	uint8_t	データ送信のバイト数	0x10
3	16	構造体	データ構造（セクション 9.3 参照） リクエストが Modem（0xff）に送信された場合のみ適用	
11	2	uint16_t	バイト 0...10 の CRC-16（付録 1 参照）	
リクエストが Beacon に送信された場合、以下のデータが受信される				
13	1	uint8_t	デバイスのアドレス	0x01...0xfe
14	1	uint8_t	パケットのタイプ	0x03
15	1	uint8_t	データ転送のバイト数	0x08/0x10
16	16	structure	データ構造（以下参照）	
32	2	uint16_t	バイト 12...22 の CRC-16（付録 1	

			参照)	

エラー応答のフォーマットは付録 2 に記載されています。

3.4.9.2 デバイス制御設定の書き込み

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス（beacon/Modem）	0x01…0xfe または 0xff
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内のデータコード	0x1201
4	2	uint16_t	アクセスモード	0x0001
6	1	uint8_t	データ転送のバイト数	0x10
7	16	構造体	データ構造（以下参照）	
23	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットの種類	リクエストが Modem に送信された場合は 0x03 リクエストがビーコンに送信された場合は 0x7f
2	2	uint16_t	データコード	0x1201
4	2	uint16_t	reserved	
6	2	uint16_t	バイト 0 ~ 5 の CRC-16（付録 1 参照）	
リクエストがビーコン（アドレス 0x01…0xfe）に送信された場合、以下のデータが受信されます				
8	1	uint8_t	デバイスのアドレス	0x01…0xfe
9	1	uint8_t	パケットのタイプ	0x10
10	2	uint16_t	データのコード	0x1201
12	2	uint16_t	reserved	
14	2	uint16_t	バイト 8 ~ 13 の CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

3.4.9.3 制御設定ペイロードデータのフォーマット

オフセット	サイズ（バイト）	タイプ	説明
0	1	uint8_t	フラグ： ビット 0...5：未定義、常にゼロにすること！ ビット 6：0 - 固定ビーコンモード、1 - Hedgehog モード ビット 7：予約済み（0）
1	1	uint8_t	UART ボーレート設定： 0：500000 bps（デフォルト値） 1：4800 bps 2：9600 bps 3：19200 bps 4：38400 bps 5：57600 bps 6：115200 bps 7...255：予約済み
2	1	uint8_t	予約済み (0)
3	1	uint8_t	ビット 0...3: 無線プロファイル: 0: 38.4 kbps 1: 150 kbps 2: 500 kbps 3...7: 予約済み ビット 4...6: 無線帯域: 0: 433 MHz 1: 868 MHz 2: 915 MHz 3: 315 MHz 4...7: 予約済み ビット 7: 予約済み
4	1	uint8_t	UART/USB 出力タイプ： 0: Marvelmind プロトコル 1: NMEA0183
5	1	uint8_t	NMEA0183 モードで送信する NMEA フレームのマスク： ビット 0：1 - $GPRMC フレームを送信 ビット 1：1 - $GPGGA フレームを送信 ビット 2：1 - $GPVTG フレームを送信 ビット 3：1 - $GPZDA フレームを送信 ビット 4...7：予約済み（0）
6	1	uint8_t	この Hedgehog から Modem へ送信するユーザーペイロードデータのバイト数（0...32）
7	1	uint8_t	「IMU via modem」モードで Modem へ送信する IMU データのマスク： ビット 0：IMU フュージョン位置 ビット 1：クォータニオン ビット 2：速度 ビット 3：加速度

			ビット 4：生加速度計データ ビット 5：生ジャイロデータ ビット 6：生コンパスデータ ビット 7：0 = IMU フュージョンを送信、1 = 生 IMU データを送信
8	1	uint8_t	ビット 0...6：ストリーミングテレメトリの間隔（0 = ストリームなし） ビット 7: 予約済み (0)
9	1	uint8_t	ビット 0: 速度計算に IMU を使用する ビット 1...7 – 予約済み (0)
10	6	6 バイト	予約済み (0)

警告! ラジオ接続されている Beacon のラジオプロファイルを変更すると、ラジオ接続が切断されます。プロファイルを切り替える必要がある場合は、すべての Beacon のラジオプロファイルを順番に切り替えてから、Modem のラジオプロファイルを切り替えてください。数秒後に、すべての Beacon が新しいラジオプロファイルで利用可能になります。

3.4.10 ネットワーク内のデバイスリストの読み取り（ファームウェア V6.01 以降）

対応ハードウェア:

Super-Beacon: 非対応

Industrial Super-Beacon: 非対応

Modem HW5.1: 対応

Super-Modem: 対応

Mini-RX (Badge、Helmet など): 非対応

Mini-TX: 非対応

Mini-TX-2: 非対応

Modem HW4.9: 対応

Beacon HW4.9: 非対応

Beacon HW4.5: 非対応

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x31xx（xx はデバイスグループの番号）
4	2	uint16_t	アクセスモード	0x0000
6	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホストへ D）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	1	uint8_t	送信データのバイト数	0x72
3	1	uint8_t	ネットワーク内のデバイスの総数（K）	
4	112	112 バイト	ネットワーク内のデバイスに関する情報の構造体（0～16 個）、下記の説明を参照	
116	1	uint8_t	予約済み	0x00
117	2	uint16_t	CRC-16（付録 1 を参照）	

エラー応答のフォーマットは付録 2 に記載されています。

ネットワーク内のデバイスに関するデータのフォーマット（7 バイト）

オフセット	サイズ（バイト）	説明
0	1	デバイスのアドレス（0x01...0xfe）
1	1	ファームウェアのメジャーバージョン
2	1	ファームウェアのマイナーバージョン
3	1	ビット 0...5：デバイスのタイプ： 10：ホイールロボット 12：クローラーロボット 22：Beacon HW V4.5 23：Beacon HW V4.5（Hedgehog モード） 24：Modem（HW V4.5/4.9） 30：Beacon HW V4.9 31：Beacon HW V4.9（Hedgehog モード） 32：Mini-RX beacon 36: Mini TX beacon（HW V5.07） 37: Industrial-TX beacon 41: Industrial-RX beacon 42: Super-Beacon 43: Super-Beacon（Hedgehog モード） 44: Industrial Super-Beacon 45: Industrial Super-Beacon（Hedgehog モード） ビット 6: 1 - このアドレスを持つデバイスが複数存在する ビット 7: 1 - スリープモード
4	1	ファームウェアのセカンドマイナーバージョン

3.4.11　ファームウェアのバージョン読み取り

対応ハードウェア:

Super-Beacon: 対応
Industrial Super-Beacon: 対応
Modem HW5.1: 対応
Super-Modem: 対応
Mini-RX（Badge、ヘルメット等）: 対応
Mini-TX: 対応
Mini-TX-2: 対応
Modem HW4.9: 対応
Beacon HW4.9: サポート済み
Beacon HW4.5: サポート済み

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	デバイスのアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0xfe00
4	2	uint16_t	アクセスモード	0x0000
6	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホスト PC）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットの種別	0x03
2	1	uint8_t	送信データのバイト数	0x08
3	1	uint8_t	ファームウェアのマイナーバージョン	
4	1	uint8_t	ファームウェアのメジャーバージョン	
5	3	3 バイト	予約済み	
8	1	uint8_t	デバイスタイプ ID	
9	2	uint16_t	予約済み	
11	2	uint16_t	CRC-16（付録 1 を参照）	

エラー返答のフォーマットは付録 2 に記載されています。

3.4.12 ユーザーデータの読み取り

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9: サポートされていません
Beacon HW4.5: サポートされていません

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x03
2	2	uint16_t	パケット内のデータコード	0x0004
4	2	uint16_t	アクセスモード	0x0000
6	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホスト HE へ）

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットの種類	0x03
2	1	uint8_t	送信データのバイト数	0x84
3	1	uint8_t	ユーザーデータの合計サイズ	
4	3	3 バイト	予約済み (0)	
7	128	uint8_t	Hedgehog からのユーザーデータ	
135	2	uint16_t	CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

Hedgehog からのユーザーデータは、以下の構造を持つレコードのシーケンスです:

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Hedgehog のアドレス	H
1	1	uint8_t	Hedgehog からのユーザーデータのバイト数	M
2	M	uint8_t	Hedgehog H からの M バイトのデータ	

3.4.13　デバイス位置の手動書き込み

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	ビーコンのアドレス	0x01…0xfe
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内のデータコード	0x5003
4	2	uint16_t	アクセスモード	0x0002
6	1	uint8_t	データ送信のバイト数	0x20
7	32	structure	データ構造（以下を参照）	
39	2	uint16_t	CRC-16（付録 1 を参照）	

応答フレームのフォーマット（Modem からホストへ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	データのコード	0x5003
4	2	uint16_t	予約済み	
6	2	uint16_t	CRC-16（付録 1 参照）	

エラー応答のフォーマットは付録 2 に記載されています。

データ構造のフォーマット:

オフセット	サイズ（バイト）	タイプ	説明	値
0	4	int32_t	X 座標、mm	
4	4	int32_t	Y 座標、mm	
8	4	int32_t	Z 座標、mm	
12	1	uint8_t	未説明	0xff
13	4	int32_t	未説明	0

17	4	int32_t	未説明	0
21	4	int32_t	未説明	0
25	1	uint8_t	未説明	0x02
26	6	6 バイト	予約済み	0

3.4.14　ビーコン間の手動距離の書き込み

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

リクエストフレームのフォーマット（ホストから Modem へ）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	パケット内のデータコード	0x4003
4	2	uint16_t	アクセスモード	0x0000
6	1	uint8_t	データ送信のバイト数	0x10
7	16	structure	データ構造（以下参照）	
23	2	uint16_t	CRC-16（付録 1 参照）	

応答フレームのフォーマット（Modem からホストへ CRC-16）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	0x10
2	2	uint16_t	データのコード	0x4003
4	2	uint16_t	予約済み	
6	2	uint16_t	CRC-16（付録 1 参照）	

エラー返信のフォーマットは付録 2 に記載されています。

データ構造のフォーマット:

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	最初のビーコンのアドレス	
1	1	uint8_t	2 番目のビーコンのアドレス	
2	4	uint32_t	ビーコン間の距離（mm）	
6	10	10 バイト	予約済み	0

4. RS-485 経由の通信プロトコル

4.1　ストリーミング用'Marvelmind'プロトコル

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

UART の対応セクションに記載されているすべてのパケットは、RS-485 経由でも利用可能です。

これらのデータは、RS-485 ハードウェアを搭載した Super-Modem および Industrial Super-Beacon でのみ利用可能であることに注意してください。

4.2 ユーザーデバイスとのデータ読み書きプロトコル

対応ハードウェア:

Super-Beacon：非対応
Industrial Super-Beacon：要問い合わせ
Modem HW5.1：非対応
Super-Modem：要問い合わせ
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9: サポートされていません
Beacon HW4.5: サポートされていません

UART の対応セクションに記載されているすべてのパケットは、要求に応じて実装できます。

これらのデータは Super-Modem および Industrial Super-Beacon でのみ利用可能です。これらの機器は RS-485 ハードウェアを搭載しているためです。

4.3 NMEA0183 通信プロトコル

対応ハードウェア:

Super-Beacon: サポートされていません
Industrial Super-Beacon: サポートされています
Modem HW5.1: サポートされていません
Super-Modem: サポートされています（SW V7.000 以降）
Mini-RX（Badge、Helmet など）: サポートされていません
Mini-TX: サポートされていません
Mini-TX-2: サポートされていません
Modem HW4.9: サポートされていません
Beacon HW4.9: サポートされていません
Beacon HW4.5: サポートされていません

UART の対応セクションに記載されているすべてのパケットは、RS-485 経由でも利用できます。

これらのデータは Super-Modem および Industrial Super-Beacon でのみ利用可能です。これらの機器は RS-485 ハードウェアを搭載しているためです。

5. SPI 経由の通信プロトコル

5.1 Hedgehog の位置情報を含むパケット

対応ハードウェア:

Super-Beacon：対応
Industrial Super-Beacon：非対応
Modem HW5.1：要望に応じて対応
Super-Modem：非対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：対応
Beacon HW4.5：対応

Super-Beacon、Beacon HW4.9、および Beacon HW4.5 は SPI スレーブデバイスとして動作し、Hedgehog の位置データを含むパケットの読み取りをサポートします。Modem HW5.1 はハードウェア SPI をサポートしており、ソフトウェアサポートは要望に応じて追加可能です。

5.2 SPI 経由のその他のデータ

対応ハードウェア:

Super-Beacon：要望に応じて対応
Industrial Super-Beacon：非対応
Modem HW5.1：要望に応じて対応
Super-Modem：非対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

第 2 章に記載されているその他のデータパケットのサポートは、Super-Beacon および Modem HW5.1 に対してご要望に応じて追加可能です

6. I2C を介した通信プロトコル

6.1 PX4 搭載ドローン向け Compress エミュレーション

対応ハードウェア:

Super-Beacon：対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：非対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

ペアリングされた Super-Beacon は、I2C 経由で PX4 に接続することで、より安定した高精度なコンパスとして機能します。

これを利用するには MMSW0003 ライセンスを購入する必要があります。

6.2 I2C 経由のその他のデータ

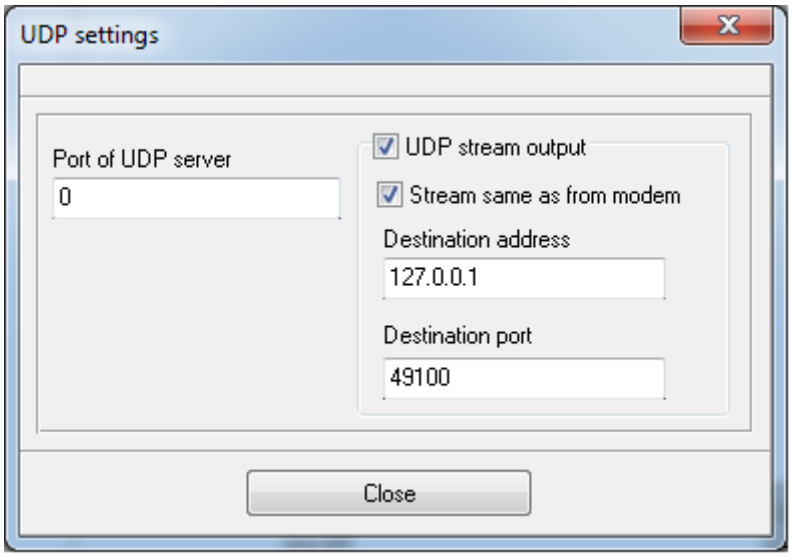
対応ハードウェア:

Super-Beacon：要求に応じて対応
Industrial Super-Beacon：非対応
Modem HW5.1：要求に応じて対応
Super-Modem：非対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

第 2 章に記載されているその他のデータパケットのサポートは、Super-Beacon および Modem HW5.1 に対して要求に応じて追加可能です

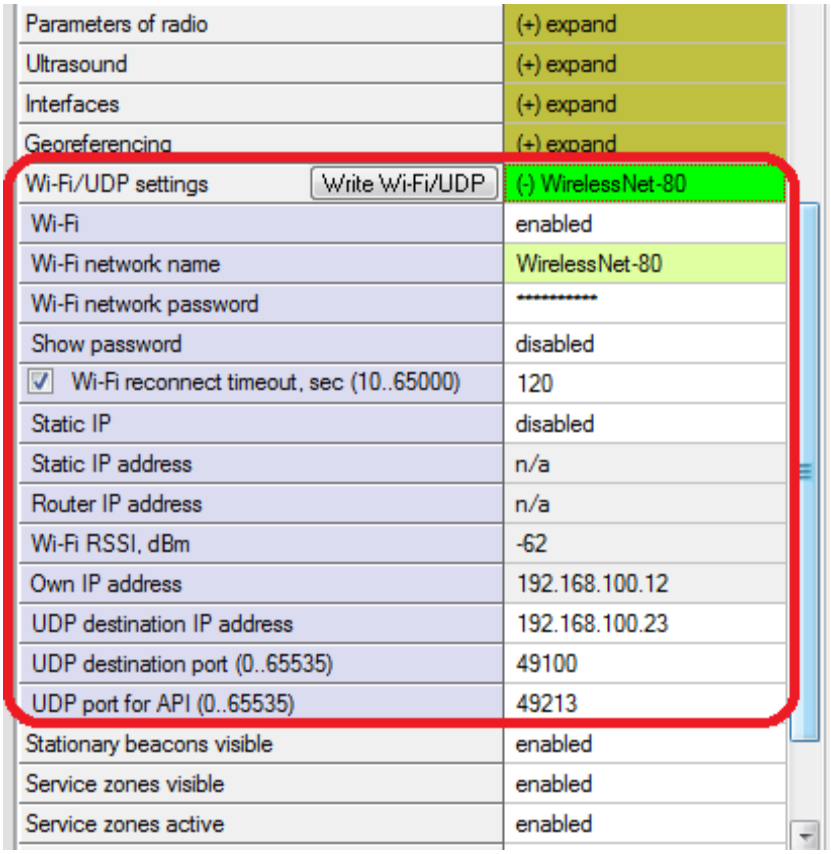
7. UDP（Wi-Fi）経由の通信プロトコル

Dashboard ソフトウェアは、Dashboard が動作している PC のネットワークインターフェースを通じて UDP 経由でデータを送信できます。宛先 IP アドレス / ポートは、メニューの「File/UDP settings」から設定可能です:



Super-Modem はオンボード Wi-Fi を搭載しており、モバイルビーコンの位置情報および以下に記載されるその他のデータをストリーミングすることができます。

Wi-Fi ネットワークおよび UDP ストリーミングの設定は、Dashboard 上の Super-Modem 設定から行うことができます:



7.1 Hedgehog 位置情報パケット

対応ハードウェア / ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応
Dashboard ソフトウェア：対応

タイムスタンプに関する注意事項をご参照ください。

パケットのフォーマット

オフセット	サイズ	タイプ	説明	値
0	1	uint8_t	Beacon のアドレス	
1	1	uint8_t	パケットタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0011
4	1	uint8_t	データサイズ（バイト）	N
5	4	uint32_t	タイムスタンプ – 座標受信時点における Dashboard/Super-Modem の起動からの経過時間（ミリ秒）	
9	4	int32_t	ビーコンの X 座標（mm）	
13	4	int32_t	ビーコンの Y 座標（mm）	
17	4	int32_t	ビーコンの Z 座標（mm）	
21	1	uint8_t	フラグバイト: ビット 0：1 - 座標利用不可。X、Y、Z フィールドのデータは使用しないこと。 ビット 1：タイムスタンプ単位インジケーター（注記参照） ビット 2...6：予約済み（0） ビット 7：– 1 – ジオフェンシングゾーン外	
22	1	uint8_t	予約済み（0）	
23	2	uint16_t	ビット 0...11：XY 平面における Hedgehog ペアの向き、デシ度（0...3600） ビット 12：1 – 座標はビーコンペアの中心に対して指定；0 – 指定された Hedgehog に対する座標 ビット 13...15：予約済み（0）	

25	2	予約済み（0）		
27	M=N-22		オプションデータフィールド – リストを参照	
27+M	2	予約済み（0）		

注意：V6.290 より前の Dashboard および Super-Modem バージョンでは、タイムスタンプは 1/64 秒単位であり、タイムスタンプ単位インジケーター（フラグバイトのビット 1）は 0 です。V6.290 以降のバージョンでは、タイムスタンプはミリ秒単位であり、タイムスタンプ単位インジケーターは 1 です。

モバイルビーコン位置パケットのオプションデータには、以下の構造体を含めることができます:

- 速度データ（7 バイト）。Dashboard のモバイルビーコン Settings のインターフェースセクションで有効にする必要があります

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	データフィールドのコード = 1 は速度ベクトルを意味する	**1**
1	2	int16_t	X 方向の速度、mm/sec	
3	2	int16_t	Y 方向の速度、mm/sec	
5	2	int16_t	Z 方向の速度、mm/sec	

7.1.1. リアルタイムタイムスタンプ付き Hedgehog 位置情報パケット（ファームウェア v7.200 以降）

対応ハードウェア/ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5:サポートされていません
Dashboard ソフトウェア:サポートされています

タイムスタンプに関する注意事項をご参照ください。

パケットのフォーマット

オフセット	サイズ	タイプ	説明	値
0	1	uint8_t	Beacon のアドレス	
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0081
4	1	uint8_t	データサイズ（バイト）	N
5	8	int64_t	タイムスタンプ – Beacon 超音波放射の Unix 時刻、1970.01.01 00:00:00 からのミリ秒数。 時刻は、Modem および Dashboard を持つすべてのデバイスで同期されます。	
13	4	int32_t	Beacon の X 座標（mm）	
17	4	int32_t	Beacon の Y 座標（mm）	
21	4	int32_t	Beacon の Z 座標（mm）	
25	1	uint8_t	フラグバイト: Bit 0: 1 – 座標が利用不可。X、Y、Z フィールドのデータは使用しないこと。 Bit 1: タイムスタンプ単位インジケーター（注記参照） Bit 2...6: 予約済み（0） Bit 7: 1 – ジオフェンシングゾーン外	
26	1	uint8_t	予約済み（0）	
27	2	uint16_t	Bit 0...11: XY 平面における Hedgehog ペアの向き、デシ度（0...3600） Bit 12: 1 – 座標は Beacon ペアの中心に対して指定；0 – 指定された Hedgehog に対する座標 ビット 13...15: 予約済み (0)	

29	2	予約済み (0)		
31	M=N-26		オプションデータフィールド – リストを参照	
31+M	2	予約済み (0)		

7.2. 固定ビーコン位置情報パケット

対応ハードウェア/ソフトウェア:

Super-Beacon: 非対応

Industrial Super-Beacon: 非対応

Modem HW5.1: 非対応

Super-Modem: 対応

Mini-RX (Badge、Helmet 等): 非対応

Mini-TX: 非対応

Mini-TX-2: 非対応

Modem HW4.9: 非対応

Beacon HW4.9: 非対応

Beacon HW4.5: 非対応

Dashboard ソフトウェア: 対応

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0012
4	1	uint8_t	送信データのバイト数	1+N*14
5	1	uint8_t	パケット内のビーコン数	N
6	1	N*14 バイト	N 個のビーコンのデータ	

N 個のビーコンそれぞれのデータ構造のフォーマット:

オフセット	サイズ（バイト）	型	説明
0	1	uint8_t	ビーコンのアドレス
1	4	int32_t	ビーコンの X 座標、mm
5	4	int32_t	ビーコンの Y 座標、mm
9	4	int32_t	ビーコンの Z 座標、mm
13	1	uint8_t	予約済み（0）

7.3. 生の IMU データを含むパケット

対応 HW/SW:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応
Dashboard ソフトウェア：対応

タイムスタンプに関する注記をご参照ください。

パケットのフォーマット

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Beacon の Address	
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0003
4	1	uint8_t	送信データのバイト数	
5	2	int16_t	加速度計、X 軸、1 mg/LSB	
7	2	int16_t	加速度計、Y 軸、1 mg/LSB	
9	2	int16_t	加速度計、Z 軸、1 mg/LSB	
11	2	int16_t	ジャイロスコープ、X 軸、0.0175 dps/LSB	
13	2	int16_t	ジャイロスコープ、Y 軸、0.0175 dps/LSB	
15	2	int16_t	ジャイロスコープ、Z 軸、0.0175 dps/LSB	
17	2	int16_t	コンパス、X 軸、1100 LSB/Gauss	
19	2	int16_t	コンパス、Y 軸、1100 LSB/Gauss	
21	2	int16_t	コンパス、Z 軸、980 LSB/Gauss	
23	1	uint8_t	ビーコンのアドレス	
24	5	5 バイト	予約済み（0）	
29	4	uint32_t	タイムスタンプ、ms	
33	8	8 バイト	予約済み	

注：コンパスデータは IMU 搭載の HW v4.9 ビーコンでのみ利用可能です。

7.3.1. リアルタイムタイムスタンプ付き生 IMU データパケット（ファームウェア v7.200 以降）

対応ハードウェア/ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX (Badge, Helmet, など):非対応
Mini-TX:非対応
Mini-TX-2:非対応
Modem HW4.9:非対応
Beacon HW4.9:非対応
Beacon HW4.5:非対応
Dashboard ソフトウェア:対応

タイムスタンプに関する注意事項をご参照ください。

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Beacon のアドレス	
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0083
4	1	uint8_t	送信データのバイト数	
5	2	int16_t	加速度計、X 軸、1 mg/LSB	
7	2	int16_t	加速度計、Y 軸、1 mg/LSB	
9	2	int16_t	加速度計、Z 軸、1 mg/LSB	
11	2	int16_t	ジャイロスコープ、X 軸、0.0175 dps/LSB	
13	2	int16_t	ジャイロスコープ、Y 軸、0.0175 dps/LSB	
15	2	int16_t	ジャイロスコープ、Z 軸、0.0175 dps/LSB	
17	2	int16_t	コンパス、X 軸、1100 LSB/Gauss	
19	2	int16_t	コンパス、Y 軸、1100 LSB/Gauss	
21	2	int16_t	コンパス、Z 軸、980 LSB/Gauss	
23	1	uint8_t	ビーコンのアドレス	
24	5	5 バイト	予約済み（0）	
29	8	int64_t	タイムスタンプ – Unix 時間、1970.01.01 00:00:00 からのミリ秒数 モデムおよび Dashboard を使用して全デバイスで同期された時刻	
37	8	8 バイト	予約済み	

注意：コンパスデータは IMU 搭載の HW v4.9 ビーコンでのみ利用可能です。

7.4. 生の距離データを含むパケット

対応ハードウェア/ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2:サポートされていません
Modem HW4.9:サポートされていません
Beacon HW4.9:サポートされていません
Beacon HW4.5:サポートされていません
Dashboard ソフトウェア:サポートされています

タイムスタンプに関する注意事項をご参照ください。

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0004
4	1	uint8_t	送信データのバイト数	
5	32		データパケット（下記参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Hedgehog のアドレス	
1	6		距離項目 1	
7	6		距離項目 2	
13	6		距離項目 3	
19	6		距離項目 4	
25	4	uint32_t	タイムスタンプ – 最新のウェイクアップイベント時点からの経過時間（ミリ秒単位）による、ビーコン超音波放射の内部時刻（V5.89+）	
29	2	uint16_t	超音波放射から現在時刻までの経過時間、ミリ秒（V5.89+）	
31	1	uint8_t	予約済み	

距離アイテムのフォーマット

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	Beacon のアドレス（アイテムが未	

			入力の場合は 0)	
1	4	uint32_t	Beacon までの距離（mm）	
5	1	uint8_t	予約済み（0）	

7.4.1. リアルタイムタイムスタンプ付き生距離データのパケット（ファームウェア v7.200 以降）

対応ハードウェア / ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX (Badge, Helmet, など): 非対応
Mini-TX: 非対応
Mini-TX-2: 非対応
Modem HW4.9: 非対応
Beacon HW4.9: 非対応
Beacon HW4.5: 非対応
Dashboard ソフトウェア: 対応

タイムスタンプに関する注意事項を参照してください。

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内のデータコード	0x0084
4	1	uint8_t	送信データのバイト数	
5	32		データパケット（以下参照）	

データパケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Hedgehog のアドレス	
1	6		距離項目 1	
7	6		距離項目 2	
13	6		距離項目 3	
19	6		距離項目 4	
25	8	int64_t	タイムスタンプ – ビーコン超音波放射の Unix 時刻、1970.01.01 00:00:00 からのミリ秒数 モデムおよび Dashboard によってすべてのデバイスと同期された時刻。	
33	2	uint16_t	超音波放射から現在時刻までの経過時間、ミリ秒 (V5.89+)	
35	1	uint8_t	予約済み	

距離アイテムのフォーマット

オフ	サイズ（バ	タイプ	説明	値

セット	イト)			
0	1	uint8_t	ビーコンのアドレス（アイテムが未入力の場合は 0)	
1	4	uint32_t	ビーコンまでの距離、mm	
5	1	uint8_t	予約済み（0)	

7.5. IMU フュージョンデータを含むパケット

対応ハードウェア/ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helm など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応
Dashboard ソフトウェア：対応

タイムスタンプに関する注意事項をご参照ください。

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Beacon のアドレス	
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0005
4	1	uint8_t	送信データのバイト数	
5	4	int32_t	ビーコンの X 座標（フュージョン）、mm	
9	4	int32_t	ビーコンの Y 座標（フュージョン）、mm	
13	4	int32_t	ビーコンの Z 座標（フュージョン）、mm	
17	2	int16_t	回転クォータニオンの W フィールド	
19	2	int16_t	回転クォータニオンの X フィールド	
21	2	int16_t	回転クォータニオンの Y フィールド	
23	2	int16_t	回転クォータニオンの Z フィールド	
25	2	int16_t	Beacon の速度 X（フュージョン）、mm/s	
27	2	int16_t	Beacon の速度 Y（フュージョン）、mm/s	
29	2	int16_t	Beacon の速度 Z（フュージョン）、mm/s	
31	2	int16_t	Beacon の加速度 X、mm/s2	
33	2	int16_t	Beacon の加速度 Y、mm/s2	
35	2	int16_t	Beacon の加速度 Z、mm/s2	
37	1	uint8_t	Beacon のアドレス	
38	1	1 バイト	予約済み（0）	
39	4	uint32_t	タイムスタンプ、ms	

43	4	4 バイト	予約済み（0）	

注：クォータニオンは 10000 の値に正規化されています。

7.5.1. リアルタイムタイムスタンプ付き IMU フュージョンデータパケット（ファームウェア v7.200 以降）

対応ハードウェア/ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応
Dashboard ソフトウェア：対応

タイムスタンプに関する注記をご参照ください。

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Beacon のアドレス	
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0085
4	1	uint8_t	送信データのバイト数	
5	4	int32_t	Beacon の X 座標（フュージョン）、mm	
9	4	int32_t	Beacon の Y 座標（フュージョン）、mm	
13	4	int32_t	Beacon の Z 座標（フュージョン）、mm	
17	2	int16_t	回転クォータニオンの W フィールド	
19	2	int16_t	回転クォータニオンの X フィールド	
21	2	int16_t	回転クォータニオンの Y フィールド	
23	2	int16_t	回転クォータニオンの Z フィールド	
25	2	int16_t	ビーコンの速度 X（フュージョン）、mm/s	
27	2	int16_t	ビーコンの速度 Y（フュージョン）、mm/s	
29	2	int16_t	ビーコンの速度 Z（フュージョン）、mm/s	
31	2	int16_t	ビーコンの加速度 X、mm/s2	
33	2	int16_t	ビーコンの加速度 Y、mm/s2	
35	2	int16_t	ビーコンの加速度 Z、mm/s2	
37	1	uint8_t	ビーコンのアドレス	
38	1	1 バイト	予約済み（0）	

39	8	int64_t	タイムスタンプ – Unix タイム、1970.01.01 00:00:00 からのミリ秒数 すべてのデバイスが Modem および Dashboard と同期した時刻	
47	4	4 バイト	予約済み（0）	

注意：クォータニオンは 10000 の値に正規化されています。

7.6. テレメトリデータを含むパケット

対応 HW/SW:

Super-Beacon：非対応

Industrial Super-Beacon：非対応

Modem HW5.1：非対応

Super-Modem：対応

Mini-RX（Badge、Helmet など）：非対応

Mini-TX：非対応

Mini-TX-2：非対応

Modem HW4.9：非対応

Beacon HW4.9：非対応

Beacon HW4.5：非対応

Dashboard ソフトウェア：サポート対象

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0006
4	1	uint8_t	送信データのバイト数	
5	2	uint16_t	バッテリー電圧、mV	
7	1	int8_t	RSSI, dBm	
8	13		予約済み (0)	

7.7. 品質および拡張位置データを含むパケット

対応ハードウェア/ソフトウェア:

Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応
Dashboard ソフトウェア：対応

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	アドレス	
1	1	uint8_t	パケットのタイプ	0x47
2	2	uint16_t	パケット内データのコード	0x0007
4	1	uint8_t	送信データのバイト数	
5	1	uint8_t	デバイスアドレス	
6	1	uint8_t	測位品質（%）	
7	1	uint8_t	0 = ジオフェンシングゾーンアラームなし 1…255 - ジオフェンシングゾーンのインデックス このフィールドには MMSW0005 ライセンスが必要です。	
8	13		予約済み（0）	

7.8. 全 Beacon のテレメトリパケット

サポートされているハードウェア/ソフトウェア:

Super-Beacon：非対応

Industrial Super-Beacon：非対応

Modem HW5.1：非対応

Super-Modem：対応（SSM ファームウェアにて）

Mini-RX（Badge、Helmet など）：非対応

Mini-TX：非対応

Mini-TX-2：非対応

Modem HW4.9：非対応

Beacon HW4.9：非対応

Beacon HW4.5：非対応

Dashboard ソフトウェア：要望に応じて対応

パケットのフォーマット

オフセット	サイズ（バイト）	型	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットのタイプ	0x48
2	2	uint16_t	パケット内のデータコード	0x2001
4	2	uint16_t	送信データのバイト数	
6	N*10		N 個の Beacon のテレメトリ（下表参照）	

Beacon テレメトリアイテムの Format

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Beacon の Address	
1	2	uint16_t	電源電圧、mV	
3	1	int8_t	RSSI、dBm	
4	4	uint32_t	最後のデータ更新からの経過時間、sec	
8	2	uint16_t	予約済み（0）	

7.9. NMEA0183 プロトコル

対応ハードウェア/ソフトウェア:

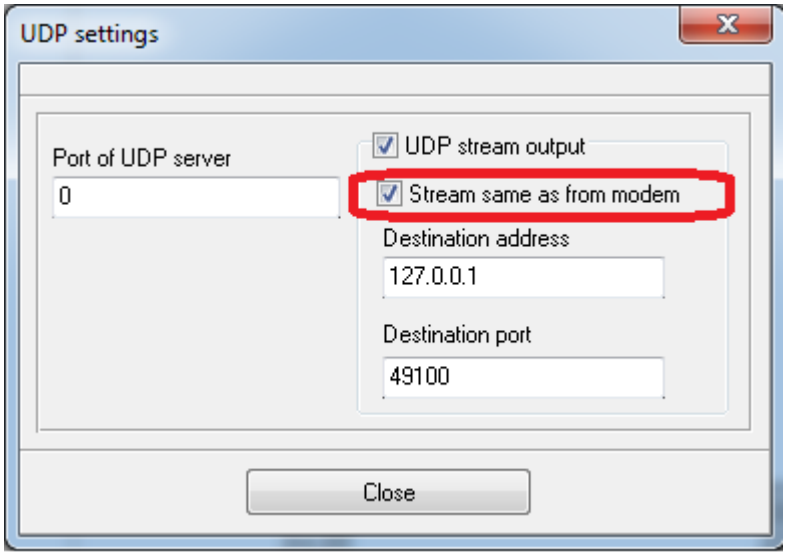
Super-Beacon：非対応
Industrial Super-Beacon：非対応
Modem HW5.1：非対応
Super-Modem：対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応
Dashboard ソフトウェア：サポート済み

パケットのフォーマット

オフセット	サイズ（バイト）	タイプ	説明	値
0	N	N バイト	NMEA0183 メッセージ（説明はこちらを参照）	
N	1	uint8_t	モバイルビーコンのアドレス	

注記:

- Super-Modem の Dashboard 設定のインターフェースセクションで NMEA0183 プロトコルが選択されている場合、Super-Modem は UDP 経由で NMEA0183 メッセージをストリーミングします
- モデムの設定のインターフェースセクションで NMEA0183 プロトコルが選択され、かつ Dashboard の Settings/UDP Settings メニューで以下のスクリーンショットに示すオプションが選択されている場合、Dashboard は NMEA0183 メッセージをストリーミングします：


UDP settings
Port of UDP server
0
UDP stream output
Stream same as from modem
Destination address
127.0.0.1
Destination port
49100
Close

8. CAN 経由の通信プロトコル

サポートされているハードウェア:

Super-Beacon：サポートされていません
Industrial Super-Beacon：サポート済み
Modem HW5.1：非対応
Super-Modem：要求に応じて対応
Mini-RX（Badge、Helmet など）：非対応
Mini-TX：非対応
Mini-TX-2：非対応
Modem HW4.9：非対応
Beacon HW4.9：非対応
Beacon HW4.5：非対応

CAN ハードウェアサポートは、要請により Super-Modem および Industrial Super-Beacon に搭載可能です。CAN が搭載されている場合、RS-485 は使用できません。

CAN のパラメータ:

ボーレート: 125 kbps

フレームフォーマット: 標準

8.1. ストリーミングの「Marvelmind」プロトコル

UART ストリーミングに関する該当章に記載されているパケットは、CAN フレーム ID 0x10 を使用して CAN でも送信されます。各 CAN フレームには 1〜8 バイトのデータを含めることができます。データバイト数は CAN フレームの DLC フィールドで指定されます。

データはロウストリームとして送信されるため、CAN フレームには 1 つのデータパケットの末尾と次のパケットの先頭が含まれる場合があります。ユーザーは複数の CAN フレームを受信し、それらのデータフィールドをバッファに格納して、UART から受信したデータと同様の方法で処理する必要があります。

8.2. NMEA0183 通信プロトコル

UART ストリーミングに関する該当章に記載されているパケットは、CAN フレーム ID 0x11 を使用して CAN でも送信されます。各 CAN フレームには 1～8 バイトのデータを含めることができます。データバイト数は CAN フレームの DLC フィールドで指定されます。

データはロウストリームとして送信されるため、CAN フレームには 1 つのデータパケットの末尾と次のパケットの先頭が含まれる場合があります。ユーザーは複数の CAN フレームを受信し、それらのデータフィールドをバッファに格納して、同様の方法で処理する必要があります。

9. Dashboard CSV ログファイルのフォーマット

Dashboard は、固定ビーコンおよびモバイルビーコンの位置情報などのデータを、Dashboard ディレクトリの「log」フォルダに保存された CSV ログファイルに記録します。バージョン V7.000 以降、ログのフォーマットが変更されました。旧フォーマットは Modem HW v4.9 のみで引き続き使用されます。

9.1. csv ログファイルのフォーマット（Dashboard バージョン V7.000+）

Dashboard バージョン V7.000+ の csv ログファイルでは、各イベントが 1 つの CSV 行としてログに記録され、イベントの種類によって行のフォーマットが異なります。ただし、すべての行タイプで行の先頭部分は共通です。

csv ログファイルの数行の例を以下に示します：

```
T2021_11_04__173001_581,user,41,17,14,4.675,2.714,0.250,2,975,100
T2021_11_04__173001_581,user,41,17,15,4.665,2.708,0.250,2,975,114
T2021_11_04__173001_581,user,41,17,26,4.073,1.987,0.250,2,3462,128
T2021_11_04__173001_581,user,41,17,27,4.075,1.987,0.250,2,3462,141
T2021_11_04__173001_581,user,41,17,28,3.588,1.979,0.250,2,3496,155
T2021_11_04__173001_581,user,41,17,29,3.592,1.978,0.250,2,3496,169
T2021_11_04__173001_701,user,43,15,nl
T2021_11_04__173001_728,user,43,27,nl
T2021_11_04__173001_756,user,43,29,nl
```

行の共通部分には、最初の 3 つのフィールドが含まれます：

「T2021_11_04__173001_581」– この行のデータに対するタイムスタンプ：2021.11.04、17:30:01.581；
「user」– ユーザー名（将来のために予約済み）。将来のバージョンでは、Dashboard はユーザーログインをサポートする予定です；
「41」– 行タイプの ID。行タイプが異なると、以降のフィールドのフォーマットも異なります。

データフィールドには、共通の特殊コードがいくつかあります：

「nl」– ライセンスなし。このフィールドに値を入力するには、特定のライセンスが必要です；
「na」– 該当なし。このフィールドに関連するデータがありません。例えば、モバイルビーコンの測位に失敗した場合、X、Y、Z 座標のフィールドには「na」が入ります。

以降の章では、各行タイプの説明を記載します。

9.1.1. ラインタイプ ID 01 – マップファイルへのリンク

この行は、マップファイルが自動的に保存されたとき、またはユーザーが「Save map」ボタンを押したときに記録されます。

行のフィールド:

N	**フィールドの説明**
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	01 - ラインタイプ ID（マップファイルへのリンク）
3	その時点で保存されたマップファイルの名前

9.1.2. ラインタイプ ID 41 – Marvelmind プロトコルストリーミングレコード

この行は、「Interfaces」セクションのモデムプロトコル設定が「Marvelmind」の場合に記録されます。

Marvelmind プロトコルにはさまざまな種類のレコードがあり、それぞれがログファイル内の異なる行に対応しています。詳細は以降のサブチャプターで説明します。

9.1.2.1. Hedgehog の位置情報 (41 17) または (41 129)

行のフィールド:

N	**フィールドの説明**
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	41 - ラインタイプ ID（Marvelmind プロトコルストリーミング）
3	17 (0x0011) – Hedgehog 位置情報のデータコード 129 (0x0081) – Hedgehog の位置データコード（リアルタイムタイムスタンプストリーミング付き）
4	Hedgehog アドレス
5	Hedgehog の X 座標（メートル）
6	Hedgehog の Y 座標（メートル）
7	Hedgehog の Z 座標（メートル）
8	フラグ: ビット 0：1 - 座標が利用不可。X、Y、Z フィールドのデータは使用しないこと。 ビット 1...6：予約済み ビット 7：1 - ジオフェンシングゾーン外
9	ヨー角およびフラグ: ビット 0...11：Hedgehog ペアの Yaw 角、デシ度（0...3600） ビット 12：1 - 座標はビーコンペアの中心に対して指定；0 - 指定された Hedgehog に対する座標
10	タイムシフト（ms）。超音波の発信から当該ラインの位置算出までに経過した時間

9.1.2.2. 固定ビーコン位置（41 18）

ラインのフィールド:

N	フィールドの説明
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	41 - ラインタイプ ID（Marvelmind プロトコルストリーミング）
3	18 (0x0012) – 固定ビーコン位置のデータコード
4	固定ビーコンのアドレス
5	ビーコンの X 座標（メートル）
6	ビーコンの Y 座標（メートル）
7	ビーコンの Z 座標（メートル）
8	予約フィールド

9.1.2.3. Hedgehog から Stationary Beacon への Raw 距離 (41 4) または (41 132)

行のフィールド:

N	フィールドの説明
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	41 - 行タイプ ID（Marvelmind プロトコルストリーミング）
3	4 (0x0004) – Raw 距離のデータコード 132 (0x0084) – Raw 距離のデータコード（リアルタイムタイムスタンプストリーミング付き）
4	Hedgehog のアドレス
5	N – 行内の距離の数
6	N 個の距離サブレコード（2*N フィールド）、以下を参照
6+N*2+1	時間シフト（ms）。超音波放射から距離測定までの経過時間

距離サブレコードのフィールド:

0	固定ビーコンのアドレス
1	固定ビーコンまでの距離

9.1.2.4. 生の IMU データ（41 3）または（41 131）

この行には MMSW0005 ライセンスが必要です。

行のフィールド:

N	フィールドの説明
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	41 - 行タイプ ID（Marvelmind プロトコルストリーミング）
3	3（0x0003）- 生の IMU データのデータコード 131（0x0083）- 生の IMU データのデータコード（Hedgehog のリアルタイムタイムスタンプが有効）
4	Hedgehog のアドレス
5	加速度計、X 軸、1 mg/LSB
6	加速度計、Y 軸、1 mg/LSB
7	加速度計、Z 軸、1 mg/LSB
8	ジャイロスコープ、X 軸、0.0175 dps/LSB
9	ジャイロスコープ、Y 軸、0.0175 dps/LSB
10	ジャイロスコープ、Z 軸、0.0175 dps/LSB
11	コンパス、X 軸、1100 LSB/Gauss
12	コンパス、Y 軸、1100 LSB/Gauss
13	コンパス、Z 軸、980 LSB/Gauss

9.1.2.5. IMU フュージョンデータ (41 5) または (41 133)

この行には MMSW0005 ライセンスが必要です。

行のフィールド:

N	フィールドの説明
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	41 - 行タイプ ID（Marvelmind プロトコルストリーミング）
3	5 (0x0005) – IMU フュージョンデータのデータコード 133 (0x0085) – IMU フュージョンデータのデータコード（Hedgehog のリアルタイムタイムスタンプ有効時）
4	Hedgehog のアドレス
5	ビーコンの X 座標（フュージョン）、メートル
6	ビーコンの Y 座標（フュージョン）、メートル
7	ビーコンの Z 座標（フュージョン）、メートル
8	回転クォータニオンの W フィールド
9	回転クォータニオンの X フィールド
10	回転クォータニオンの Y フィールド
11	回転クォータニオンの Z フィールド
12	ビーコンの速度 X（フュージョン）、mm/s
13	ビーコンの速度 Y（フュージョン）、mm/s
14	ビーコンの速度 Z（フュージョン）、mm/s
15	ビーコンの加速度 X、mm/s2
16	ビーコンの加速度 Y、mm/s2
17	ビーコンの加速度 Z、mm/s2

9.1.2.6. テレメトリデータ（41 6）

行のフィールド:

N	フィールドの説明
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	41 - 行タイプ ID（Marvelmind プロトコルストリーミング）
3	6（0x0006）– テレメトリデータのデータコード
4	ビーコンのアドレス
5	供給電圧、V
6	RSSI、dBm

9.1.2.7. 品質および拡張位置データ（41 7）

行のフィールド:

N	フィールドの説明
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	41 - ラインタイプ ID（Marvelmind プロトコルストリーミング）
3	7 (0x0007) – 品質および拡張位置データのデータコード
4	Hedgehog のアドレス
5	位置の品質（%）
6	ジオフェンシングゾーンの番号（このフィールドには MMSW0005 ライセンスが必要です）

9.1.3. ラインタイプ ID 42 – NMEA0183 ストリーミングレコード

この行には MMSW0005 ライセンスが必要です。

この行は、「Interfaces」セクションの Modem プロトコル設定が「NMEA0183」の場合に記録されます。

行のフィールド:

N	**フィールドの説明**
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	42 - ラインタイプ ID（NMEA0183 プロトコルストリーミング）
3	Hedgehog のアドレス
4、5 など	NMEA0183 フォーマットに従ったフィールドのシーケンス（NMEA0183 レコードもカンマ区切り値フォーマットを使用）

9.1.4. ラインタイプ ID 43 – Hedgehog を介して送信されるユーザーペイロードデータ

この行には MMSW0005 ライセンスが必要です。

この行は、Hedgehog が設定のインターフェースセクションでゼロ以外のペイロードデータサイズを有効にしており、ユーザーデバイスが Hedgehog の USB または UART 経由でデータを送信した場合に記録されます。

また、ペイロードデータはロボット v100 や Boxie など、一部の Marvelmind デバイスでも利用可能です。

行のフィールド:

N	**フィールドの説明**
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	43 - ラインタイプ ID（ユーザーペイロード）
3	Hedgehog のアドレス
4、5 など	ペイロードデータのカンマ区切りバイトのシーケンス（各フィールドは 1 バイト）

9.1.4.1. ロボット v100 のペイロードテレメトリデータ

行の一般フォーマットはユーザーペイロードデータフォーマットに対応しています。

データバイト（行の 4 番目のフィールドから開始）は、以下に説明するフォーマットのデータレコードを形成します。

マルチバイト値はロウバイトから配置されます（リトルエンディアンフォーマット）。

ロボット v100 テレメトリレコード N3:

オフセット	サイズ（バイト）	タイプ	説明	値
0	2	uint16_t	レコード ID	0x3003
2	2*12		12 個の LiDAR による距離（LiDAR 1 つあたり 2 バイト） 各 LiDAR データは以下のフォーマットを持つ: ビット 0...11 – LiDAR による距離（mm） ビット 12...15 – 距離測定ステータス ステータス = 0 – 距離が測定済み ステータス <> 0 – 距離が未測定	
26	1	uint8_t	LiDAR 全体のステータス: Bit0: 1 – LiDAR の読み取り成功 0 – LiDAR の読み取り失敗 ビット 1...7 – 予約済み（0）	
27	1	uint8_t	ロボットの状態:	

			0: ロボットは正常に停止中 1: ロボットはいずれかのアラームにより停止中 2: ロボットは自律走行中 3: ロボットは充電中	
28	1	uint8_t	RV - ロボットのバッテリー電圧。 V= (RV/10) + 20 ボルト	
29	2	int16_t	ロボットの供給電流、x10 mA 値が負の場合、ロボットのバッテリーはこの電流で充電中です。	N
31	1	uint8_t	残バッテリー容量、%	
32	2	uint16_t	ロボットの供給電力、ワット	
34	2	uint16_t	左ホイールの速度、mm/s	
36	2	uint16_t	右ホイールの速度、mm/s	
37	1	uint8_t	左モーターへの出力、%	
38	1	uint8_t	右モーターへの出力、%	
39	2	uint16_t	目標速度、mm/s	
41	2	int16_t	ロボットの X 座標、cm	
43	1	uint8_t	ステータスフラグ: ビット 0...3 – 予約済み ビット 4: 1 – 超音波トラッキングエラー ビット 5...7 - 予約済み	
44	2	int16_t	ロボットの Y 座標、cm	

ロボット v100 テレメトリー レコード N4:

オフセット	サイズ（バイト）	タイプ	説明	値
0	2	uint16_t	レコード ID	0x3004
2	2*12		12 個の LiDAR による距離（LiDAR 1 個あたり 2 バイト） 各 LiDAR データは以下のフォーマットを持ちます: ビット 0...11 – LiDAR による距離、mm ビット 12...15 – 距離計測ステータス ステータス = 0 – 距離計測済み ステータス <> 0 – 距離未計測	
26	1	uint8_t	LiDAR 全体ステータス: Bit0: 1 – LiDAR 読み取り成功 0 – LiDAR 読み取り失敗 ビット 1...7 – 予約済み（0）	
27	1	uint8_t	ロボットの状態: 0: ロボットは正常停止中 1: ロボットはいずれかのアラームにより停止中 2: ロボットが自律移動中 3: ロボットが充電中	
28	1	uint8_t	移動プログラムにおける現在の項目のインデックス	
29	1	uint8_t	移動プログラムの項目の総数	

30	1	uint8_t	予約済み	
31	1	uint8_t	LiDAR による障害物の方向: 0 – なし 1 – 前方 2 – 左 3 – 右	
32	1	uint8_t	LiDAR による最小アラーム距離、x2cm	
33	1	uint8_t	アラームを引き起こした現在の測定 LiDAR 距離、x2cm	
34	3		予約済み	
36	2	int16_t	ロボット X 座標、cm	
38	2	int16_t	ロボット Y 座標、cm	

9.1.4.2. ロボット Boxie のペイロードテレメトリデータ

行の一般フォーマットはユーザーペイロードデータフォーマットに対応しています。

データバイト（行の 4 番目のフィールドから始まる）は、以下に説明するフォーマットのデータレコードを構成します。

マルチバイト値はロウバイトから配置されます（リトルエンディアン形式）。

ロボット Boxie テレメトリレコード N1:

オフセット	サイズ（バイト）	タイプ	説明	値
0	2	uint16_t	レコード ID	0x3101
2	2*12		12 個の LiDAR による距離（LiDAR 1 個につき 2 バイト） 各 LiDAR データの形式は以下のとおりです: ビット 0…11 – LiDAR による距離、mm ビット 12…15 – 距離測定ステータス ステータス = 0 – 距離が測定されている ステータス <> 0 – 距離が測定されていない	
26	1	uint8_t	LiDAR 全体のステータス: ビット 0: 1 – LiDAR の読み取り成功 0 – LiDAR の読み取り失敗 ビット 1…7 – 予約済み（0）	
27	1	uint8_t	ロボットの状態: 0: ロボットは正常停止中 1: ロボットはいずれかのアラームにより停止中 2: ロボットは自律移動中 3: ロボットは充電中	
28	1	uint8_t	RV – ロボットのバッテリー電圧、x100 mV。 例えば、値 118 は 11.8V を意味する	
29	2	int16_t	ロボットの供給電流、x10 mA 例えば、値 123 は 1.230 A を意味する	

31	1	uint8_t	予約済み	
32	1	uint8_t	左モーターへの電力供給、%	
33	1	uint8_t	右モーターへの電力供給、%	
34	2	uint16_t	左車輪の速度、mm/s	
36	2	uint16_t	右車輪の速度、mm/s	
37	2	int16_t	左モーターのオドメトリによる走行距離、cm	
39	2	int16_t	右モーターのオドメトリによる走行距離、cm	
41	2	int16_t	ロボットのX座標、cm	
43	1	uint8_t	ステータスフラグ: ビット 0...1 – 予約済み ビット 2: 1 – 移動プログラムが実行中 ビット 2: 1 – 移動が一時停止中 ビット 4: 1 – 超音波トラッキングエラー ビット 5...7 - 予約済み	
44	2	int16_t	ロボットのY座標、cm	

ロボット Boxie テレメトリレコード N3:

オフセット	サイズ（バイト）	タイプ	説明	値
0	2	uint16_t	レコード ID	0x3103
2	2	int16_t	角度制御 PID レギュレータの現在の「P」値	
4	2	int16_t	角度制御 PID レギュレータの現在の「I」値	
6	2	int16_t	角度制御 PID レギュレータの現在の「D」値	
8	2	int16_t	EKF フィルターを使用して算出されたロボットの X 座標（cm）	
10	2	int16_t	EKF フィルターを使用して算出されたロボットの Y 座標（cm）	
12	2	int16_t	予約済み	
13	1	uint8_t	LiDAR 全体のステータス： Bit0: 1 – LiDAR 読み取り成功 0 – LiDAR 読み取り失敗 Bit 1...7 – 予約済み（0）	
14	1	uint8_t	ロボットの状態： 0: ロボットは正常停止中 1: 何らかのアラームによりロボットが停止中 2: ロボットは自律移動中 3: ロボットは充電中	
15	1	uint8_t	現在の移動ステップのインデックス（最初のウェイポイントは 0、次は 1、以下同様）	
16	1	uint8_t	現在のプログラムにおける総移動ステップ数	
17	1	uint8_t	移動フラグ： ビット 0：1 –「永続実行」オプション ビット 1...7 - 予約済み	
18	1	uint8_t	0 – LiDAR によるアラームなし 1...12 - アラームを発生させた LiDAR のインデックス	
19	1	uint8_t	LiDAR による最小アラーム距離、x2cm	
20	1	uint8_t	アラームを発生させた現在の測定 LiDAR 距離、x2cm	
21	2	int16_t	ロボット上のペアビーコンによる角度、度	
23	1	uint8_t	ロボットの目標速度（ユーザー設定）、cm/s	
24	2	int16_t	ロボットの X 座標、cm	
26	2	int16_t	ロボットの Y 座標、cm	
28	2	int16_t	Reserved	
30	2	int16_t	Reserved	
32	2	int16_t	目標方向とロボットの現在の向きの間の偏差角度（度）	
34	1	uint8_t	Reserved	

35	2	int16_t	融合されたロボットの向き角度（度）	
36	5	5 bytes	Reserved	

9.1.5. ライン タイプ ID 44 – Dashboard リアルタイム プレイヤー位置

このラインには MMSW0005 ライセンスが必要です。

このラインは、リアルタイム プレイヤーが有効な場合に Hedgehog 用として記録されます。リアルタイム プレイヤーは 100 Hz の位置データを提供します。

ラインのフィールド:

N	**フィールドの説明**
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	44 - ラインタイプ ID（リアルタイムプレイヤー位置）
3	Hedgehog のアドレス
4	予約フィールド
5	Hedgehog の X 座標（メートル）
6	Hedgehog の Y 座標（メートル）
7	Hedgehog の Z 座標（メートル）

9.1.6. ラインタイプ ID 55 – ショートビーコンテレメトリー

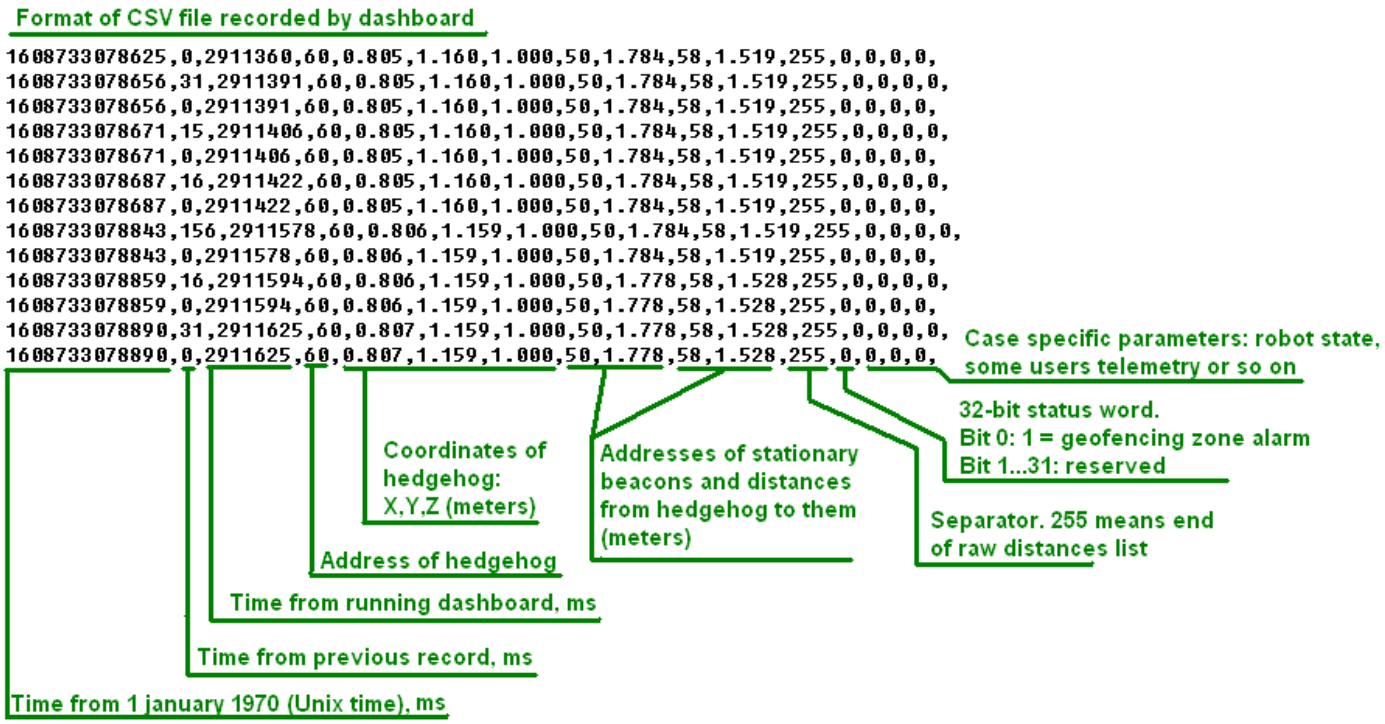
この行には、SSM（Super Super-Modem）から受信したビーコンテレメトリーが含まれています。

行のフィールド:

N	フィールドの説明
0	タイムスタンプ（共通フィールド）
1	ユーザー名（共通フィールド）
2	55 - ラインタイプ ID（ショートテレメトリー）
3	ビーコンのアドレス
4	ビーコン電源電圧の下位バイト：V0
5	ビーコン電源電圧の上位バイト：V1 ビーコン電源電圧は V = V0+V1*256 mV
6	2 の補数コードによる RSSI：R0 R0 が 128 未満の場合、RSSI = R0 dBm (R0 >= 128) の場合、RSSI = R0-256 dBm
7	予約済み
8	予約済み

9.2. csv ログの旧フォーマット（V7.000 以前の Dashboard、またはモデム HW v4.9）

ログファイルの旧フォーマットを示す図を以下に示します:



10. Marvelmind API

Marvelmind API ライブラリは、Marvelmind Dashboard ソフトウェアで使用され、ユーザーのソフトウェアへのインターフェースを提供します。API は、MS Windows 向けのダイナミックリンクライブラリ（DLL）および Linux（x86 および ARM プラットフォーム）向けの共有ライブラリとして提供されます。API は USB（仮想シリアルポート）を介してモデムに接続し、モデムとの通信プロトコルを実装します。

API ライブラリに加えて、ソフトウェアパッケージには C のサンプルソフトウェアが含まれており、これは API のテストに使用され、すべての API 関数の呼び出しが含まれています。

このサンプルは、ユーザーソフトウェアの開発の基礎として、また API ライブラリインターフェース（ファイル 'marvelmind_api.c'）を他のプログラミング言語へ移植する際の基礎として使用できます。

動作確認済み環境:

1. MS Windows 10；CPU：Intel Core i5
2. Ubuntu 20.04；CPU：Intel Core i5
3. Raspbian （2018-11-13-raspbian-stretch-full）；プラットフォーム： Raspberry Pi 3 Model B+

10.1. Windows へのインストール

- Marvelmind API ソフトウェアパッケージをダウンロードします。Dashboard API およびサンプルソフトウェアを、プログラムに使用するディレクトリにコピーします。Windows バージョンのサンプルにはビルド済みの実行ファイルが付属しているため、API ソフトウェアパッケージに含まれる 'windows' ディレクトリから 'mm_api_example.exe' をすぐに実行できます。

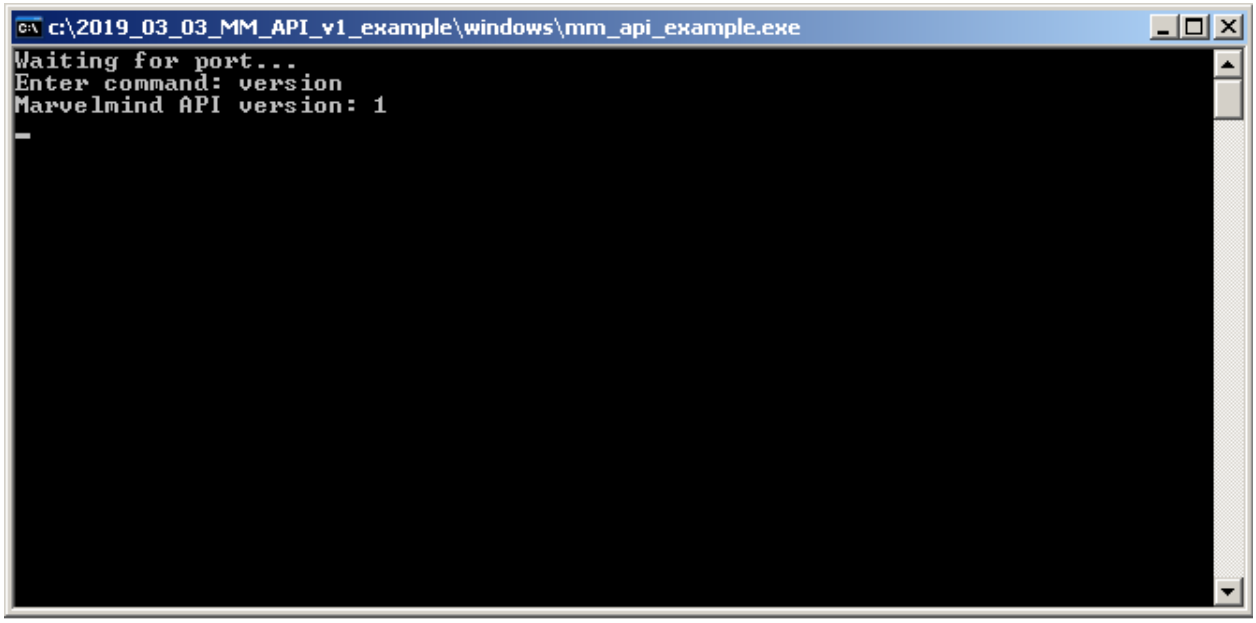
10.2. Linux へのインストール

- Marvelmind API ソフトウェアパッケージをダウンロードします。Dashboard API を、プログラムに使用するディレクトリにコピーします。Linux バージョンは、x86（Intel または AMD CPU を搭載したほとんどのノートパソコン）と arm（例：Raspberry PI などのシングルボードコンピュータ）の 2 つのハードウェアプラットフォーム向けに提供されていることに注意してください。
- お使いのプラットフォームに対応するライブラリ libdashapi.so を、libdashapi.so が存在するディレクトリで開いたターミナルにて、コマンド sudo cp libdashapi.so /usr/local/lib を実行してディレクトリ /usr/local/lib にコピーします。その後、ターミナルで sudo ldconfig を実行します。
- シリアルポートへのアクセス権を付与するために、ユーザーを dialout グループに追加する必要がある場合があります：
  - ターミナルで次のコマンドを実行します：sudo adduser $USER dialout
  - ディレクトリ /etc/udev/rules.d に、以下の内容を持つファイル「99-tty.rules」を追加してください：

    ```
    #Marvelmind serial port rules
    KERNEL=="ttyACM0",GROUP="dialout",MODE="666"
    ```
- サンプルソフトウェアをビルドする — パッケージに付属する「source」ディレクトリで開いたターミナルで「make all」を実行してください
- ターミナルで「./mm_api_example」と入力してサンプルを実行してください

10.3. API への接続を確認する

サンプルソフトウェアを起動した後、ターミナルで「Space」キーを押し、コマンド「version」を入力して Enter キーを押してください。サンプルソフトウェアが API のバージョンを表示した場合、API ライブラリとの通信が可能です。



10.4. Marvelmind API ライブラリの説明

API は、MS Windows 向けのダイナミックリンクライブラリ（DLL）、および Linux（x86 および ARM プラットフォーム）向けの共有ライブラリとして提供されます。このライブラリには、コンピューターの USB ポートに接続された Modem を介して Marvelmind システムを監視・制御するための関数セットが含まれています。本セクションでは、これらの関数すべてについて説明します。

異なるプログラミング言語との互換性を高めるため、複雑なデータ構造のほとんどは、メモリへの型なしポインタを介して渡されます。関数の説明には、メモリプール内の各データフィールドのオフセットが記載されています。サンプルソフトウェアに含まれる「marvelmind_api.c」ファイルには、メモリプールと C 構造体のフィールド間でデータを移動する実装例を確認できます。

説明内のパラメータの型は C 構文で示されています。各型の説明を以下に示します:

型	サイズ（バイト）	説明
bool	1	ブール型。ゼロは false、ゼロ以外は true を意味します
uint8_t	1	符号なし整数値、0…255
int8_t	1	2 の補数形式による符号付き整数値、-128…127
uint16_t	2	符号なし整数値、0…65535
int16_t	2	2 の補数形式の符号付き整数値、-32768…32767
uint32_t	4	符号なし整数値、0…4294967295
int32_t	4	2 の補数形式の符号付き整数値、 -2147483648…2147483647
void *	4/8	メモリポインタ（メモリ上のアドレス）。 32 ビットプラットフォームでは 4 バイト、64 ビットプラットフォームでは 8 バイト。

各関数の説明には、その関数が使用可能な API バージョンのセットが含まれています。新しい API バージョンは、Marvelmind システムの新機能に対応するより多くの関数をサポートします。現在、Dashboard で利用可能なすべての機能が API を通じて利用できるわけではありませんので、追加の API 関数が必要な場合は info@marvelmind.com までお問い合わせください。

サポートされている関数の一覧:

関数	API バージョン	必要なライセンス
Marvelmind API ライブラリのバージョンを取得	V1+	なし
最後のエラーを取得	V6+	なし
シリアルポートを開こうとする	V1+	なし
指定された名前でシリアルポートを開こうとする	V2+	なし
UDP ポートを開こうとする	V9+	なし
シリアルポートを閉じる	V1+	なし
Marvelmind デバイスのバージョンおよび CPU ID を取得	V1+	なし
デバイスのリストを取得	V1+	なし
デバイスをウェイクアップする	V1+	なし
デバイスをスリープ状態にする	V1+	なし
ビーコンからテレメトリデータを取得する	V1+	なし
最新の位置データを取得する	V1+	なし
最新の位置データを取得する（角度付き）	V3+	なし
ビーコンの位置を設定する	V3+	MMSW0005
ビーコン間の距離を設定する	V4+	MMSW0005
最新の生距離データを取得する	V1+	none
Hedgehog の高さを取得する	V4+	none
Hedgehog の高さを設定する	V4+	MMSW0005
サブマップ内の固定ビーコンの高さを取得する	V4+	none
サブマップ内の固定ビーコンの高さを設定する	V4+	MMSW0005
位置更新レートの設定を取得する	V1+	なし
位置更新レート設定の設定	V1+	MMSW0005
サブマップの追加	V1+	MMSW0005
サブマップの削除	V1+	MMSW0005
サブマップのフリーズ	V1+	MMSW0005
サブマップのフリーズ解除	V1+	MMSW0005
サブマップ設定の取得	V1+	なし
サブマップ設定の設定	V1+	MMSW0005
マップのフリーズ	V4+	MMSW0005
マップのフリーズ解除	V4+	MMSW0005
ビーコンの超音波設定を取得	V1+	none
ビーコンの超音波設定を適用	V1+	MMSW0005
マップの消去	V1+	MMSW0005
デバイスをデフォルト設定にリセット	V1+	MMSW0005
ビーコンを軸に接続する	V2+	MMSW0005
Modem の設定メモリダンプを読み取る	V3+	MMSW0005
Modem の設定メモリダンプを書き込む	V3+	MMSW0005
Modem から気温設定を取得する	V3+	none
Modem に気温設定を書き込む	V3+	none
デバイスのソフトウェアリセット	V3+	none
ビーコンのリアルタイムプレイヤー設定を取得する	V6+	なし
ビーコンリアルタイムプレーヤー設定を設定する	V6+	MMSW0005
ジオリファレンス設定を取得する	V6+	なし
ジオリファレンス設定を設定する	V6+	MMSW0005
位置更新モードを取得する	V6+	なし
位置更新モードを設定する	V6+	MMSW0005
位置更新コマンド	V6+	MMSW0005

ビーコンのジオフェンシングアラーム状態を設定する	V9+	MMSW0005 MMSW0006
汎用ユーザーペイロードデータの送信	V9+	MMSW0005
汎用ユーザーペイロードデータの取得	V9+	MMSW0005
手動距離測定コマンドの送信	V9+	MMSW0011
Modem からのストリーミングデータの取得	V9+	none
デバイスタイプが Modem であるかどうかの確認	V1+	none
デバイスタイプが固定ビーコンであるかどうかの確認	V1+	なし
デバイスタイプが Hedgehog であるか確認する	V1+	なし

10.4.1. Marvelmind API ライブラリのバージョンを取得する

API ライブラリのバージョンを読み取ります。このバージョンのライブラリで必要な機能が使用可能であることを確認するために必要です。

関数名: mm_api_version
C での宣言: bool mm_api_version(void *pdata);
使用可能な API バージョン: V1+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行中にエラーが発生した

パラメータ:

型	説明
void *	入力するデータへのポインタ

ポインタを介して返されるデータの構造体

型	説明
uint32_t	API ライブラリのバージョン

10.4.2. 最後のエラーを取得する

API ライブラリとの最後の操作のステータスを読み取り、エラーの原因を特定します。

関数名: mm_get_last_error
C での宣言: bool mm_get_last_error(void *pdata);
対応 API バージョン: V6+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行中にエラーが発生した

パラメータ:

型	説明
void *	データを格納するためのポインタ

ポインタを介して返されるデータの構造体

型	説明
uint32_t	直前の操作のステータス: 0: 操作が正常に実行された 1: 通信エラー 2: シリアルポートのオープンエラー 3: ライセンスが必要

10.4.3. シリアルポートを開く

Marvelmind デバイス（Modem またはビーコン）が USB（仮想シリアルポート）経由で接続されているポートを開きます。API がすべてのシリアルポートを検索し、Marvelmind デバイスに対応しているかどうかを確認するため、シリアルポート名を指定する必要はありません。

関数名：mm_open_port
C での宣言：bool mm_open_port ();
対応 API バージョン：V1+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、ポートが開かれた false – 実行エラー

パラメータ: なし

10.4.4. 指定された名前でシリアルポートを開く

Marvelmind デバイス Marvelmind（Modem または Beacon）が USB（仮想シリアルポート）経由で接続されているポートを開きます。関数は指定された名前のポートを開こうとします。

関数名: mm_open_port_by_name
C での宣言: bool mm_open_port_by_name(void *pdata);
利用可能な API バージョン: V2+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、ポートが開かれた false – 実行エラー

パラメータ:

型	説明
void *	シリアルポート名へのポインタ – ゼロで終端された ASCII 文字列（ASCIIZ）

10.4.4.1. UDP ポートを開く

USB の代わりに UDP を使用して Super-Modem との UDP 通信を確立することができます。

関数名: mm_open_port_udp
C での宣言: bool mm_open_port_udp(void *pdata);
利用可能な API バージョン: V9+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、UDP ポートが開かれた false – 実行エラー

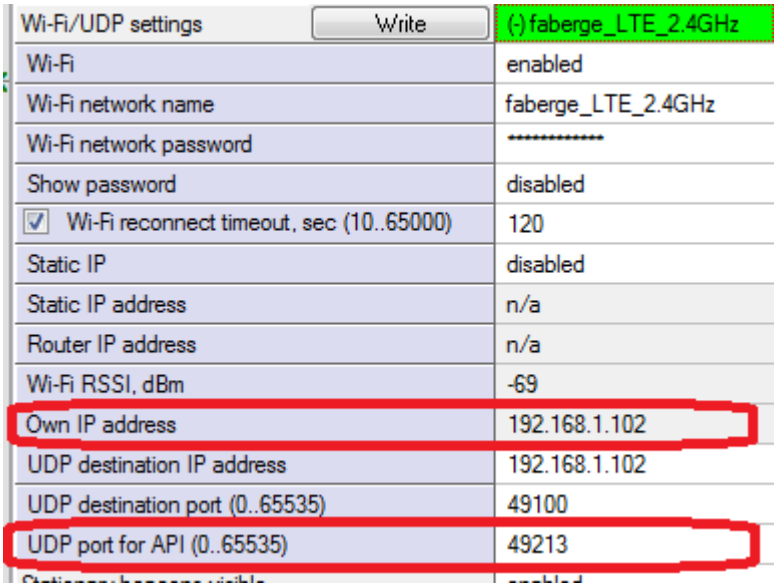
パラメータ:

型	説明
void *	UDP 設定の構造体へのポインタ（以下を参照）

ポインタが指すデータの構造体:

型	説明
uint16_t	接続する UDP ポート
uint16_t	通信タイムアウト（ms）
uint16_t	予約済み
最大 255 バイト	IP アドレス – ゼロで終端される ASCII 文字列（ASCIIZ）

IP アドレスおよび UDP ポートは、Super-Modem の設定と一致している必要があります（以下のスクリーンショットを参照）。



10.4.5. シリアルポートを閉じる

mm_open_port 関数によって以前に開かれたポートを閉じます。
関数名：mm_close_port
C 言語での宣言：bool mm_close_port ();
対応 API バージョン：V1+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数は正常に実行され、ポートは閉じられました false – 実行エラー

パラメータ：なし

10.4.6. Marvelmind デバイスのバージョンおよび CPU ID の取得

バージョンおよび CPU ID を読み取ります。バージョンにはファームウェアのバージョンおよびデバイスハードウェアの種類に関する情報が含まれます。CPU ID はデバイス個体の固有 ID です。

関数名：mm_get_device_version_and_id
C での宣言：bool mm_get_device_version_and_id (uint8_t address, void *pdata);

利用可能な API バージョン：V1+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数は正常に実行され、バージョンおよび CPU ID データが取得されました false – 実行エラー

パラメータ:

型	説明
uint8_t	Marvelmind デバイスのアドレス（1…254）
void *	データを格納するためのポインタ

ポインタ経由で返されるデータの構造:

型	説明
uint8_t	ファームウェアのメジャーバージョン（例："6"、バージョン V6.07a の場合）
uint8_t	ファームウェアのマイナーバージョン（例："7"、バージョン V6.07a の場合）
uint8_t	ファームウェアのセカンドマイナーバージョン（例："1"、バージョン V6.07a の場合）
uint8_t	デバイスタイプ ID（付録参照）。
uint8_t	ファームウェアオプション（TBD）。
uint32_t	CPU ID。この値を 16 進数で表示すると、Dashboard およびデバイスのステッカーに ID 表示される形式の CPU ID が得られます。

10.4.7. デバイス一覧の取得

Modem が認識している Marvelmind デバイスの一覧を読み取ります。この一覧には、スリープ状態のデバイスを含め、Modem のネットワークに無線接続されているすべてのデバイスが含まれます。

関数名: mm_get_devices_list
C での宣言: bool mm_get_devices_list (void *pdata);
対応 API バージョン: V1+

必要ライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、デバイスのリストが取得された false – 実行エラー

パラメータ:

型	説明
void *	入力されるデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
uint8_t	リスト内の後続デバイスの数（N）
N*9 バイト	N 個のデバイス構造体のシーケンス（次の表に記載）

リスト内の各デバイスの構造体:

タイプ	説明
uint8_t	デバイスのアドレス
bool	true = アドレス重複 — 同じアドレスを持つデバイスが 2 台以上検出された false = アドレス重複なし
bool	true = デバイスはスリープ中 false = デバイスはスリープ中でない
uint8_t	ファームウェアのメジャーバージョン（例：バージョン V6.07a の場合は「6」）
uint8_t	ファームウェアのマイナーバージョン（例：バージョン V6.07a の場合は「7」）
uint8_t	ファームウェアの第 2 マイナーバージョン（例：バージョン V6.07a の場合は「1」）
uint8_t	デバイスタイプ ID（付録を参照）。
uint8_t	ファームウェアオプション（TBD）。
uint8_t	フラグ: ビット 0：1 – デバイス接続完了 – デバイスが接続を確認済み 0 – デバイスからの確認待ち（Dashboard の「Connecting...」に相当）。 ビット 1...7 - TBD

10.4.8. デバイスのウェイク

指定したデバイスにウェイクコマンドを送信します。ウェイクコマンドが送信され、該当デバイスが存在する場合、そのデバイスは数秒以内に Modem に接続し、デバイスリストに表示されます。

関数名：mm_wake_device
C 言語での宣言：bool mm_wake_device (uint8_t address);
対応 API バージョン：V1+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、ウェイクコマンドが送信された false – 実行エラー

パラメータ:

型	説明
uint8_t	1…254 - スリープ解除する Marvelmind デバイスのアドレス 0 – 全デバイスをスリープ解除

10.4.9. デバイスをスリープ状態に移行

既存のデバイスをスリープ状態に移行します。

関数名：mm_send_to_sleep_device
C 言語での宣言：bool mm_send_to_sleep_device (uint8_t address);
対応 API バージョン：V1+

必要ライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、スリープコマンドが送信された false – 実行エラー

パラメータ:

型	説明
uint8_t	1…254 - スリープさせる Marvelmind デバイスのアドレス 0 – 全デバイスをスリープ状態に移行

10.4.10. Beacon からテレメトリデータを取得

Marvelmind の Beacon のテレメトリデータを読み取ります。

関数名: mm_get_beacon_telemetry
C での宣言: bool mm_get_beacon_telemetry (uint8_t address, void *pdata);
対応 API バージョン: V1+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、テレメトリが取得された false – 実行エラー

パラメータ:

型	説明
uint8_t	Marvelmind ビーコンのアドレス（1…254）
void *	データを格納するためのポインタ

ポインタを介して返されるデータの構造体:

型	説明
uint32_t	ビーコンの動作時間（秒）（リセットまたは起動からの経過時間）
int8_t	RSSI、dBm – 無線信号強度
int8_t	測定温度、°C
uint16_t	電源電圧、mV
16 バイト	予約済み（0）

10.4.11. 最新の位置データの取得

Modem から最新の更新済み座標パックを読み取ります。また、利用可能な場合はユーザーペイロードデータも読み取ります。

関数名：mm_get_last_locations
C 言語での宣言：bool mm_get_last_locations(void *pdata);
利用可能な API バージョン：V1+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、位置データが取得された false – 実行エラー

パラメータ:

型	説明
void *	入力するデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
18*6 bytes	最後に更新された座標の 6 つの 18 バイトデータ構造（下表参照）
bool	true – 読み取り可能な新しい生の距離データが存在する
5 バイト	TBD
uint8_t	ユーザーペイロードデータサイズ (M)
M バイト	ユーザーペイロードデータ

各位置データ項目の構造:

タイプ	説明
uint8_t	デバイスのアドレス (1…254) 0 - このデータ項目は未入力
uint8_t	ヘッドインデックス (TBD)
int32_t	X 座標、mm
int32_t	Y 座標、mm
int32_t	Z 座標、mm
uint8_t	ステータスフラグ（TBD）
uint8_t	測位品質、0…100%
uint8_t	TBD
uint8_t	TBD

10.4.12. 最新の位置データの取得（角度付き）

Modem から最新の更新済み座標パックを読み取ります（ペアリングされたビーコンの角度付き）。利用可能な場合はユーザーペイロードデータも読み取ります。

関数名: mm_get_last_locations2
C での宣言: bool mm_get_last_locations2(void *pdata);
対応 API バージョン: V3+
必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、位置データが取得されました false – 実行エラー

パラメータ:

型	説明
void *	データへのポインタ（入力用）

ポインタ経由で返されるデータの構造:

型	説明
20*6 バイト	最後に更新された座標の 20 バイトデータ構造体×6 個（下表参照）
bool	true – 新しい生の距離データの読み取りが可能
5 バイト	TBD
uint8_t	ユーザーペイロードデータサイズ（M）
M バイト	ユーザーペイロードデータ

各位置データ項目の構造:

型	説明
uint8_t	デバイスのアドレス（1…254） 0 - このデータ項目は未入力
uint8_t	ヘッドインデックス（TBD）
int32_t	X 座標、mm
int32_t	Y 座標、mm
int32_t	Z 座標、mm
uint8_t	ステータスフラグ（TBD）
uint8_t	測位品質、0…100%
uint8_t	TBD
uint8_t	TBD
uint16_t	ビット 0…11 – 回転角度（1/10 度単位）（ペアビーコン機能が有効な場合） ビット 12 – 1 = 角度取得不可 ビット 13…15 - 予約済み

10.4.13. ビーコンの位置を設定する

指定されたビーコンの位置を手動で設定する。

関数名: mm_set_beacon_location
C 言語での宣言: bool mm_set_beacon_location (uint8_t address, void *pdata);
対応 API バージョン: V3+
必要なライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、位置情報が更新された false – 実行中にエラーが発生した

パラメータ:

型	説明
uint8_t	ビーコンのアドレス
void *	位置データを含むバッファへのポインタ

ポインタによるデータ構造（関数呼び出し前に入力する必要があります）：

型	説明
int32_t	ビーコンの新しい X 座標（mm）
int32_t	ビーコンの新しい Y 座標（mm）
int32_t	ビーコンの新しい Z 座標（mm）

10.4.14. ビーコン間の距離の設定

ビーコン間の距離の手動設定。

関数名：mm_set_beacons_distance
C での宣言：bool mm_set_beacons_distance (void *pdata);
対応 API バージョン：V4+
必要なライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、距離が書き込まれた false – 実行エラー

パラメータ:

型	説明
void *	距離データを含むバッファへのポインタ

ポインタが指すデータの構造（関数呼び出し前に入力する必要があります）：

型	説明
uint8_t	1 つ目のビーコンのアドレス
uint8_t	2 つ目のビーコンのアドレス
int32_t	ビーコン間の距離、mm

10.4.15. 最新の生距離データの取得

Modem から最新の更新された生距離パックを読み取ります。

関数名: mm_get_last_distances
C での宣言: bool mm_get_last_distances(void *pdata);
対応 API バージョン: V1+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、生距離データが取得された false – 実行エラー

パラメータ:

型	説明
void *	データ格納先へのポインタ

ポインタ経由で返されるデータの構造:

型	説明
uint8_t	生距離データ項目の数 (N) リクエストごとの生距離の最大数は 16 です: N<=16
9*N バイト	最後の生距離の N 個の 9 バイトデータ構造体（下表参照）

各生距離データ項目の構造:

型	説明
uint8_t	超音波 RX デバイスのアドレス（1…254） 0 - このデータ項目は未設定
uint8_t	RX Head インデックス（TBD）
uint8_t	超音波 TX デバイスのアドレス（1…254） 0 - このデータ項目は未設定
uint8_t	TX Head インデックス（TBD）
uint32_t	TX デバイスから RX デバイスまでの距離（mm）
uint8_t	TBD

10.4.16. Hedgehog の高さを取得する

モバイルビーコン（Hedgehog）の高さを返します。

関数名：mm_get_hedge_height
C での宣言：bool mm_get_hedge_height (uint8_t address, void *pdata);
利用可能な API バージョン：V4+
必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、高さが返された false – 実行エラー

パラメータ:

型	説明
uint8_t	Hedgehog の Address
void *	高さデータを含むバッファへのポインタ

ポインタが指すデータの構造:

タイプ	説明
int32_t	Hedgehog の高さ（mm）

10.4.17. Hedgehog の高さを設定する

モバイルビーコン（Hedgehog）の高さを設定します。

関数名: mm_set_hedge_height
C での宣言: bool mm_set_hedge_height (uint8_t address, void *pdata);
利用可能な API バージョン: V4+
必要なライセンス: MMSW0005

戻り値:

タイプ	説明
bool	true – 関数が正常に実行され、高さが変更された false – 実行エラー

パラメータ:

タイプ	説明
uint8_t	Hedgehog のアドレス
void *	高さデータを含むバッファへのポインタ

ポインタが指すデータの構造体（関数呼び出し前に設定する必要があります）:

型	説明
int32_t	Hedgehog の高さ（mm）

10.4.18. サブマップ内の固定ビーコンの高さを取得する

サブマップ内の固定ビーコンの高さを返します。

関数名: mm_get_beacon_height
C での宣言: bool mm_get_beacon_height (uint8_t address, void *pdata);
対応 API バージョン: V4+
必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、高さが返された false – 実行エラー

パラメータ:

型	説明
uint8_t	Beacon のアドレス
void *	高さデータを含むバッファへのポインタ

ポインタが指すデータの構造:

型	説明
uint8_t	Submap ID（関数呼び出し前に設定する必要があります）
int32_t	Beacon の高さ（mm）

10.4.19. Submap における固定 Beacon の Beacon 高さを設定する

Submap における固定 Beacon の高さを設定します。

関数名: mm_set_beacon_height
C 言語での宣言: bool mm_set_beacon_height (uint8_t address, void *pdata);
対応 API バージョン: V4+
必要ライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、高さが変更された false – 実行エラー

パラメータ:

型	説明
uint8_t	ビーコンのアドレス
void *	高さデータを含むバッファへのポインタ

ポインタが指すデータの構造体（関数呼び出し前に設定する必要があります）:

型	説明
uint8_t	サブマップ ID
int32_t	ビーコンの高さ、mm

10.4.20. 位置更新レート設定の取得

Modem から位置更新レート設定を読み取ります。

関数名: mm_get_update_rate_setting
C での宣言: bool mm_get_update_rate_setting (void *pdata);

対応 API バージョン: V1+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、更新レートが取得された false – 実行エラー

パラメータ:

型	説明
void *	入力されるデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
uint32_t	位置更新レートの設定（mHz 単位）。1 Hz の場合は 1000、16 Hz の場合は 16000、0.05 Hz モードの場合は 50 が返されます。

10.4.21. 位置更新レートの設定を行う

位置更新レートの設定を Modem に書き込みます。

関数名: mm_set_update_rate_setting
C での宣言: bool mm_set_update_rate_setting (void *pdata);

利用可能な API バージョン: V1 以降

必要なライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、更新レートが変更されました false – 実行中にエラーが発生しました

パラメータ:

型	説明
void *	データへのポインタ

ポインタが指すデータの構造（関数呼び出し前に設定する必要があります）:

型	説明
uint32_t	位置更新レートの設定（mHz 単位）。1 Hz の場合は 1000、16 Hz の場合は 16000、0.05 Hz モードの場合は 50 が返されます。システムは、0.05 Hz、0.1 Hz、0.2 Hz、0.5 Hz、1 Hz、2 Hz、4 Hz、8 Hz、12 Hz、16 Hz、16+ Hz のシリーズから、指定された更新レートに最も近い値を使用します。

10.4.22. submap の追加

新しい submap を追加します。

関数名：mm_add_submap
C 言語での宣言：bool mm_add_submap (uint8_t submapId);
対応 API バージョン：V1+

必要なライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、submap が追加されました false – 実行エラー

パラメータ:

型	説明
uint8_t	追加する Submap ID（0…254）

10.4.23. Submap の削除

既存の Submap を削除します。

関数名：mm_delete_submap
C での宣言：bool mm_delete_submap (uint8_t submapId);
利用可能な API バージョン：V1+

必要なライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、Submap が削除された false – 実行エラー

パラメータ:

型	説明
uint8_t	削除する Submap ID（0…254）

10.4.24. Submap のフリーズ

Submap をフリーズします。

関数名: mm_freeze_submap
C での宣言: bool mm_freeze_submap (uint8_t submapId);
対応 API バージョン: V1+

必要ライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、サブマップがフリーズされた false – 実行エラー

パラメータ:

型	説明
uint8_t	フリーズするサブマップ ID（0…254）

10.4.25. サブマップのフリーズ解除

サブマップのフリーズを解除します。

関数名: mm_unfreeze_submap
C での宣言: bool mm_unfreeze_submap (uint8_t submapId);
対応 API バージョン: V1+

必要なライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、サブマップのフリーズが解除された false – 実行エラー

パラメータ:

型	説明
uint8_t	フリーズを解除するサブマップ ID (0…254)

10.4.26. サブマップ設定の取得

Modem からサブマップ設定を読み取る。

関数名: mm_get_submap_settings
C での宣言: bool mm_get_submap_settings (uint8_t submapId , void *pdata);
対応 API バージョン: V1+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、サブマップ設定が取得された false – 実行エラー

パラメータ:

型	説明
uint8_t	サブマップ ID（0...254）
void *	入力されるデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
uint8_t	三辺測量の開始 Beacon
uint8_t	Beacon の開始セット、Beacon 1
uint8_t	Beacon の開始セット、Beacon 2
uint8_t	ビーコンの開始セット、ビーコン 3
uint8_t	ビーコンの開始セット、ビーコン 4
bool	true = 3D ナビゲーション有効
bool	true = サブマップは Z 座標のみに使用
bool	true = 距離制限を手動設定 false = 距離制限を自動設定
uint8_t	最大距離、メートル（手動距離制限の場合）
int16_t	サブマップ X シフト、cm
int16_t	サブマップ Y シフト、cm
int16_t	サブマップ Z シフト、cm
uint16_t	サブマップの回転、センチ度
int16_t	平面回転クォータニオン、W（クォータニオンは 10000 に正規化）
int16_t	平面回転クォータニオン、X
int16_t	平面回転クォータニオン、Y
int16_t	平面回転クォータニオン、Z
int16_t	サービスゾーンの厚さ、cm
int16_t	2D モードにおける Hedge の高さ
bool	true = サブマップはフリーズされている
bool	true = サブマップはロックされている
bool	true = 固定ビーコンはモバイルより高い位置にある
bool	true = サブマップが左右反転されている
4 バイト	サブマップ内の Beacon のアドレスリスト（0 = なし）
8 バイト	近隣サブマップの ID リスト（255 = なし）
uint8_t	サービスゾーンポリゴンの頂点数（P）
P*4 バイト	サービスゾーンポリゴン頂点の構造体リスト（以下参照）

サービスゾーンポリゴン頂点の構造体:

型	説明
int16_t	X、cm
int16_t	Y、cm

10.4.27. サブマップ設定の書き込み

サブマップ設定を Modem に書き込みます。

関数名：mm_set_submap_settings
C 言語での宣言：bool mm_set_submap_settings (uint8_t submapId , void *pdata);

利用可能な API バージョン：V1+

必要なライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、サブマップ設定が変更された false – 実行エラー

パラメータ:

型	説明
uint8_t	サブマップ ID（0...254）
void *	書き込むデータへのポインタ （「get submap settings」関数を参照)

10.4.28. マップのフリーズ

サブマップをフリーズします。

関数名：mm_freeze_map
C での宣言：bool mm_freeze_map ();
利用可能な API バージョン：V4+

必要なライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、マップがフリーズされた false – 実行エラー

10.4.29. マップのフリーズ解除

サブマップをフリーズします。

関数名: mm_unfreeze_map
C 言語での宣言: bool mm_freeze_map ();
対応 API バージョン: V4+

必要なライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、マップのフリーズが解除された false – 実行エラー

10.4.30. ビーコンの超音波設定の取得

指定されたビーコンから超音波設定を読み取ります。

関数名: mm_get_ultrasound_settings
C での宣言: bool mm_get_ultrasound_settings (uint8_t address , void *pdata);
対応 API バージョン: V1+

必要なライセンス: なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、超音波設定が取得されました false – 実行エラー

パラメータ:

型	説明
uint8_t	ビーコンのアドレス (1…254)
void *	格納されるデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
uint16_t	超音波 TX の周波数（DSP RX 専用ビーコンには適用されない）
uint8_t	TX ペリオド数（DSP RX 専用ビーコンには適用されない）
bool	true = RX に AGC を使用する false = RX に手動ゲインを使用する
uint16_t	手動ゲイン値（0…4000）
bool	true = センサー RX1 が通常モードで有効
bool	true = センサー RX2 が通常モードで有効
bool	true = センサー RX3 が通常モードで有効
bool	true = センサー RX4 が通常モードで有効
bool	true = センサー RX5 が通常モードで有効
bool	true = センサー RX1 がフローズンモードで有効になっている
bool	true = センサー RX2 がフローズンモードで有効になっている
bool	true = センサー RX3 がフローズンモードで有効になっている
bool	true = センサー RX4 がフローズンモードで有効になっている
bool	true = センサー RX5 がフローズンモードで有効になっている
uint8_t	DSP RX フィルターのインデックス（DSP ビーコンにのみ関連） 0 = 19 kHz 1 = 25 kHz 2 = 31 kHz 3 = 37 kHz 4 = 45 kHz

10.4.31. ビーコンの超音波設定を行う

指定したビーコンに超音波設定を書き込む。

関数名: mm_set_ultrasound_settings
C での宣言：bool mm_set_ultrasound_settings (uint8_t address, void *pdata);
対応 API バージョン：V1+

必要ライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、超音波設定が変更された false – 実行エラー

パラメータ:

型	説明
uint8_t	ビーコンのアドレス（1…254）
void *	書き込むデータへのポインタ（「超音波設定の取得」関数を参照）。

10.4.32. マップの消去

Modem 内のマップを消去 – すべてのサブマップ（サブマップ 0 を除く）を削除し、サブマップ 0 を初期状態にリセットし、ネットワークから接続されているすべてのビーコンを削除する。

関数名：mm_erase_map
C での宣言：bool mm_erase_map ();
対応 API バージョン：V1+

必要ライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、マップが消去された false – 実行エラー

パラメータ：なし

10.4.33. デバイスをデフォルト設定にリセット

デバイスをデフォルト設定（無線、超音波など）にリセットする。

関数名：mm_set_default_settings
C 言語での宣言：bool mm_set_default_settings (uint8_t address);
対応 API バージョン：V1+

必要ライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、デバイスがデフォルト設定にリセットされた false – 実行エラー

パラメータ:

型	説明
uint8_t	デバイスのアドレス（1…254） 255 – USB 経由で接続されたデバイスをデフォルトにリセット

10.4.34. ビーコンを軸に接続する

選択したビーコンが軸上に位置するようにマップをシフトする。

関数名：mm_beacons_to_axes
C での宣言： bool mm_beacons_to_axes (uint8_t address_0, uint8_t address_x, uint8_t address_y);
対応 API バージョン：V2+

必要ライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、マップがシフトされた false – 実行エラー

パラメータ:

タイプ	説明
uint8_t	address_0 – 中心（X=0, Y=0）に配置するビーコンのアドレス
uint8_t	address_x – X 軸方向（Y=0）に配置するビーコンのアドレス
uint8_t	address_y – Y の正方向（Y>0）に配置するビーコンのアドレス

10.4.35. Modem の設定メモリのダンプ読み取り

Modem の設定メモリのダンプを読み取ります。Modem の設定および保存されたマップの保存が可能です。

関数名: mm_read_flash_dump
C 言語での宣言: bool mm_read_flash_dump(uint32_t offset, uint32_t size, void *pdata);

対応 API バージョン: V3+
必要なライセンス: MMSW0005

戻り値:

タイプ	説明
bool	true – 関数が正常に実行され、ダンプが読み取られた false – 実行エラー

パラメータ:

型	説明
uint32_t	offset – 設定メモリの先頭からのオフセット（バイト単位)
uint32_t	size – 読み取るデータのサイズ（バイト単位)
void *	pdata – データ受信用ユーザーバッファへのポインタ

10.4.36. Modem の設定メモリのダンプを書き込む

Modem の設定メモリにデータダンプを書き込みます。Modem の設定およびマップを復元することができます。

関数名: mm_write_flash_dump

C 言語での宣言: bool mm_write_flash_dump(uint32_t offset, uint32_t size, void *pdata);

対応 API バージョン: V3+

必要ライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行され、ダンプが書き込まれた false – 実行エラー

パラメータ :

型	説明
uint32_t	offset – 設定メモリの先頭からのオフセット（バイト単位） 正常に動作するには、offset は 4096 バイトのページ境界に揃えてください（値は 0、4096、8192 など）。
uint32_t	size – 書き込むデータのサイズ（バイト単位）
void *	pdata – データが格納されたユーザーバッファへのポインタ

注意：設定を書き込んだ後は、新しい設定を適用し上書きを防ぐため、 Modem のソフトウェアリセット（mm_reset_device(255)）を実行することを推奨します。

10.4.37. デバイスの再起動（ソフトリセット）

指定されたデバイスに対してソフトウェアリセットを実行します。

関数名：mm_reset_device

C での宣言：bool mm_reset_device (uint8_t address);

対応 API バージョン：V3+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、デバイスがリセット中 false – 実行エラー

パラメータ:

型	説明
uint8_t	デバイスのアドレス（1…254） 255 – USB 経由で接続されたデバイスのソフトウェアリセット

10.4.38. Modem から気温設定を読み取る

Modem から気温設定（摂氏）を読み取ります。

関数名：mm_get_air_temperature

C での宣言：bool mm_get_air_temperature (void *pdata);

利用可能な API バージョン：V3 以上

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、温度が返されました false – 実行エラー

pdata ポインタを介して返されるデータの構造:

型	説明
int8_t	気温（摂氏）

10.4.39. 気温設定を Modem に書き込む

Modem に気温設定（摂氏）を設定します。

関数名：mm_set_air_temperature

C での宣言：bool mm_set_air_temperature (void *pdata);

対応 API バージョン：V3+

必要ライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行され、温度が書き込まれました false – 実行エラー

pdata ポインタを介してユーザーが提供するデータの構造:

型	説明
int8_t	気温（摂氏）

10.4.40. Beacon のリアルタイムプレイヤー設定を取得する

Beacon のリアルタイムプレイヤー設定を読み取ります。

関数名：mm_get_realtime_player_settings

C 言語での宣言：bool mm_get_realtime_player_settings (uint8_t address, void *pdata);

対応 API バージョン：V6 以降

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
uint8_t	address - ビーコンのアドレス（1…254）
void *	pdata - 入力するデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
bool	true = リアルタイムプレーヤーが有効
uint8_t	処理するリアルタイムプレーヤーの前方サンプル数
uint8_t	処理するリアルタイムプレーヤーの後方サンプル数
uint8_t	予約済み（0）
uint8_t	予約済み（0）

10.4.41. ビーコンのリアルタイムプレーヤー設定の設定

ビーコンのリアルタイムプレーヤー設定をセットアップします。

関数名：mm_set_realtime_player_settings

C 言語での宣言：bool mm_set_realtime_player_settings (uint8_t address, void *pdata);

対応 API バージョン：V6+

必要ライセンス：MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
uint8_t	address - ビーコンのアドレス（1…254）
void *	pdata - 書き込むデータへのポインタ（「Get beacon real-time player settings」関数を参照）

10.4.42. ジオリファレンス設定の取得

ジオリファレンス設定（Marvelmind マップの点（X=0, Y=0）のジオロケーション）を読み取ります。

関数名：mm_get_georeferencing_settings

C での宣言：bool mm_get_georeferencing_settings (void *pdata);

対応 API バージョン：V6+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
void *	pdata - 入力されるデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
int32_t	緯度、x10-7 度
int32_t	経度、x10-7 度

10.4.43. ジオリファレンス設定の設定

Marvelmind マップの(X=0, Y=0)地点のジオロケーションに関するジオリファレンス設定をセットアップします。

関数名: mm_set_georeferencing_settings

C での宣言: bool mm_set_georeferencing_settings (void *pdata);

利用可能な API バージョン: V6+

必要なライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行中にエラーが発生した

パラメータ:

型	説明
void *	pdata – 書き込むデータへのポインタ（「Get georeferencing settings」関数を参照)

10.4.44. 位置更新モードの取得

モバイルビーコンの位置更新の現在のモードを読み取ります。

関数名: mm_get_update_position_mode

C での宣言: bool mm_get_update_position_mode (void *pdata);

対応 API バージョン： V6 以上

必要ライセンス： なし

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
void *	pdata – 入力されるデータへのポインタ

ポインタ経由で返されるデータの構造:

型	説明
uint8_t	モバイルビーコンの位置更新モード: 0 – 位置の自動更新（デフォルトモード） 1 – 次の更新サイクルにおけるユーザーリクエストによる位置更新 2 – ユーザーリクエストにより位置情報を即時更新
7 バイト	予約済み (0)

10.4.45. 位置更新モードの設定

モバイルビーコンの位置更新モードを設定します。

関数名: mm_set_update_position_mode

C 言語での宣言: bool mm_set_update_position_mode (void *pdata);

対応 API バージョン: V6+

必要ライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
void *	pdata - 書き込むデータへのポインタ（関数 'Get mode of updating positions' を参照)

10.4.46. 位置更新コマンド

モバイルビーコンの位置を更新するコマンドを送信します（更新モードが自動でない場合）。

関数名: mm_set_update_position_command

C での宣言: bool mm_set_update_position_command (void *pdata);

対応 API バージョン: V6+

必要ライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
void *	pdata – 書き込むデータへのポインタ

ポインタが指すデータの構造:

型	説明
8 バイト	予約済み（0）

10.4.47. Beacon のジオフェンシングアラーム状態を設定する

Super-Beacon のアラームピンにアラーム State 設定するコマンドを送信する。

ピンのアラーム状態は、Dashboard の設定の「Interfaces」セクションにある「Alarm pin mode」設定で指定できます（MMSW0006 ライセンスが有効化されている場合）。

関数名：mm_set_alarm_state

C での宣言：bool mm_set_alarm_state (uint8_t address, void *pdata);

対応 API バージョン：V9 以降

必要なライセンス：MMSW0005、MMSW0006

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行中にエラーが発生した

パラメータ:

型	説明
uint8_t	address – Beacon のアドレス（1 ~ 254）
void *	pdata - 書き込むデータへのポインタ

ポインタによるデータの構造:

型	説明
uint8_t	アラームピンモード: 0 – ピンはジオフェンシングステータスに従って自動制御される 1 – ピンは手動制御 – アラーム状態なし 2 – ピンは手動制御 – アラーム状態
uint8_t	ジオフェンシングゾーンインデックス – ビーコンがアラーム状態でストリーム出力するジオフェンシングゾーンの番号
6 バイト	予約済み (0)

10.4.48. 汎用ユーザーペイロードデータの送信

汎用ユーザーペイロードデータを送信します。API がモデムに接続されている場合、データは指定されたモバイルビーコンの UART/USB ポートを介して送信されます。API がモバイルビーコンに接続されている場合、データはモデムの UART/USB ポートを介して送信されます。受信したデータは、受信 API 関数、Arduino サンプル、ROS およびその他のソフトウェアによってリモート側で利用可能です。

関数名: mm_send_user_payload_data

C での宣言: bool mm_send_user_payload_data (uint8_t address, void *pdata);

対応 API バージョン: V9+

必要ライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
uint8_t	address - ビーコンのアドレス（1…254）（API がモデムに接続されている場合） API がビーコンに直接接続されている場合は n/a
void *	pdata - 書き込むデータへのポインタ

ポインタが指すデータの構造:

型	説明
uint8_t	送信するデータのサイズ
256 bytes	送信するための汎用データバッファ

10.4.49. 汎用ユーザーペイロードデータの取得

送信 API 関数、Arduino、ROS またはその他のユーザーソフトウェアによって送信された汎用ユーザーペイロードデータを受信します。API が Modem に接続されている場合、この関数はモバイルビーコンの UART/USB ポート経由で送信されたデータを受信できます。API がモバイルビーコンに接続されている場合、この関数は Modem 経由で送信されたデータを受信できます。

関数名: mm_get_user_payload_data

C での宣言: bool mm_get_user_payload_data (void *pdata);

対応 API バージョン: V9+

必要なライセンス: MMSW0005

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメーター:

型	説明
void *	pdata – 受信するデータへのポインター

ポインターによって受信されるデータの構造:

型	説明
uint8_t	address – ビーコンのアドレス
int64_t	データ送信のタイムスタンプ – 01.01.1970 からのミリ秒数（Unix 時間）
uint8_t	送信するデータのサイズ
256 bytes	受信データのバッファ

10.4.50. 手動距離測定のコマンドを送信する

指定されたビーコンからシステム内の他のビーコンまでの距離を測定するコマンドを送信します。現在のソフトウェアバージョンでは、IA（逆アーキテクチャ）でサポートされています。

関数名: mm_send_distances_measurement_command

C での宣言: bool mm_send_distances_measurement_command (void *pdata);

対応 API バージョン: V9+

必要ライセンス: MMSW0011

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行エラー

パラメータ:

型	説明
void *	pdata - 送信するデータへのポインタ

ポインタが指すデータの構造:

型	説明
uint8_t	モード: 0 – 自動 1 – 手動（このコマンドによる）
uint8_t	ビーコンのアドレス
uint32_t	測定する最大距離、mm
8 バイト	予約済み

10.4.51. Modem からのストリーミングデータを取得する

前述の形式で Modem のストリーミングデータを読み取ります。

関数名：mm_get_stream_data

C での宣言：bool mm_get_stream_data (void *pdata);

対応 API バージョン：V9+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 関数が正常に実行された false – 実行中にエラーが発生した

パラメータ:

型	説明
void *	pdata – 送信するデータへのポインタ

ポインタが指すデータの構造:

型	説明
uint8_t	この応答におけるストリームレコードの数（0...16）
138*16 バイト	138 バイトの 16 個のストリーミングレコード（以下参照）
8 バイト	予約済み

ストリームレコードの構造:

型	説明
uint8_t	レコードサイズ（バイト）
uint8_t	レコードタイプ。Dashboard ログファイルの「line type」と同じ値。 例えば、41 は Marvelmind プロトコルデータを意味する
8 バイト	予約済み
128 バイト	ストリームレコードデータ

10.4.52. デバイスタイプが Modem であるかどうかを確認する

指定されたデバイスタイプが Modem に対応するかどうかを確認する。

関数名：mm_device_is_modem
C での宣言：bool mm_device_is_modem (uint8_t deviceType);
利用可能な API バージョン：V1+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 指定されたデバイスタイプが Modem に対応する

パラメータ:

型	説明
uint8_t	チェックするデバイスタイプ

10.4.53. デバイスタイプが固定ビーコンであるかどうかを確認する

指定されたデバイスタイプが固定ビーコンに対応するかどうかを確認します。

関数名：mm_device_is_beacon
C での宣言：bool mm_device_is_beacon (uint8_t deviceType);
利用可能な API バージョン：V1+

必要なライセンス：なし

戻り値:

型	説明
bool	true – 指定されたデバイスタイプが固定ビーコンに対応している

パラメータ:

型	説明
uint8_t	チェック対象のデバイスタイプ

10.4.54. デバイスタイプが Hedgehog であるかどうかを確認する

指定されたデバイスタイプが Hedgehog に対応しているかどうかを確認します。

関数名： mm_device_is_hedgehog
C 言語での宣言：bool mm_device_is_hedgehog (uint8_t deviceType);
利用可能な API バージョン：V1+

必要なライセンス： なし

戻り値:

型	説明
bool	true – 指定されたデバイスタイプが Hedgehog に対応している

パラメータ:

型	説明
uint8_t	チェックするデバイスタイプ

10.5. Marvelmind API の C サンプルの説明

C サンプルは Marvelmind API のテストに使用され、ユーザーアプリケーション構築の基盤として利用できます。

C サンプルはコンソールアプリケーションであ D す。以下のプラットフォームでテスト済みです：

- CPU：Intel Core 2 Duo、OS：MS Windows XP；
- CPU：Intel Core i5、OS：Linux Ubuntu 16.04；
- Raspberry Pi 3 Model B+、OS：Raspbian（2018-11-13-raspbian-stretch-full）

Windows プラットフォームでは、サンプルは CodeBlocks IDE を使用してビルドされたため、CodeBlocks プロジェクトファイルが含まれています。

Linux プラットフォームでは、サンプルは make ユーティリティを使用してビルドされたため、対応する makefile が含まれています。

サンプルには以下のモジュールが含まれています：

ファイル名	説明
main.c	main()関数を含むモジュール。サンプル関数の呼び出しおよびシンプルなコマンドラインインターフェースを実装します。
marvelmind_example.c marvelmind_example.h	marvelmindStart() – サンプルの初期化 marvelmindFinish() – API との作業終了後に呼び出される marvelmindCycle() – メインループから頻繁に呼び出される また、モジュールにはユーザーが入力したコマンドを処理するための関数が複数含まれています。
marvelmind_api.c marvelmind_api.h	marvelmindAPILoad() – API ライブラリを読み込む marvelmindAPIFree() – API ライブラリが使用するメモリを解放する API ライブラリとの通信に関するすべての関数。
marvelmind_devices.c marvelmind_devices.h	「デバイスリストの取得」コマンドを呼び出すことで Modem から取得したビーコンのリストをサポートします。各ビーコンには、その位置情報および他のビーコンとの距離に関するデータが含まれています。
marvelmind_pos.c marvelmind_pos.h	最新の位置データおよび最新の生の距離データを読み取り、デバイスリスト内のこれらのデータを更新します。
marvelmind_utils.c marvelmind_utils.h	他のモジュールで使用されるヘルパー関数群。

サンプルの動作方法：

1. 成功するまでシリアルポートのオープンを試みる
2. ポートが開かれると、プログラムは USB 経由で接続されたデバイスのバージョンを読み取ります。これが Modem である場合、プログラムは次のステップの実行を続けます。
3. Modem に接続されると、プログラムは 1 Hz のレートでデバイスリストをリードします。デバイスリストは marvelmind_devices.c モジュールに現在保存されているものと比較され、変更が検出された場合は marvelmind_devices.c 内のリストが更新されます。すべての変更はコンソールに出力されます。
4. Modem に接続されると、プログラムは 20 Hz のレートで最新の位置データを読み取ります。新しい生距離データのフラグが設定されている場合、プログラムは最新の生距離を読み取ります。プログラムは marvelmind_devices.c のデバイスリスト内のデータと位置および距離を比較し、変更があればデータを更新します。変更されたすべてのデータはコンソールに出力されます。

5. プログラムが最新の位置データを 10 回取得できない場合、ポートを閉じてステップ 1 に戻り、再度ポートを開こうとします。ポートの再オープンは、Modem が USB から切断され、再接続された場合に対応するために必要です。
6. ユーザーが「スペース」ボタンを押すと、プログラムは「Enter command: 」というメッセージを表示し、ユーザーのコマンドを待ちます。ほとんどの API 関数はユーザーコマンドによって呼び出されます。詳細は以下を参照してください。

ユーザーコマンド:

プログラムの実行中にユーザーが「スペース」ボタンを押すと、プログラムは「Enter command: 」というメッセージを表示します。ユーザーはキーボードでコマンドを入力し、Enter キーを押す必要があります。

以下の表にすべてのユーザーコマンドの形式を示します:

コマンドグループ	説明
API バージョン	コマンドの形式: **version** 動作: API ライブラリのバージョンを出力します。
プログラムの終了	コマンドの形式: **quit** 動作: プログラムの実行を終了します。
スリープ/ウェイク	コマンドの形式: **wake <address>** 動作: ウェイクコマンドを実行します。 例: **wake 5 - デバイス 5 をウェイクさせるコマンドを送信する** **wake 0 - すべてのデバイスをウェイクさせるコマンドを送信する**
	コマンドの形式: **sleep <address>** 動作: スリープコマンドの送信を実行します。 例: **sleep 5 - デバイス 5 をスリープ状態にする** **sleep 0 - すべてのデバイスをスリープ状態にする**
デフォルト	コマンドの形式: **default <address>** 動作: デフォルト設定へのリセットコマンドを実行します。 例: **default 5 - デバイス 5 のデフォルト設定を行う**
テレメトリの読み取り	コマンドの形式: **tele <address>** 動作: ビーコンのテレメトリデータを読み取り、表示する。 例: **tele 5 - ビーコン 5 のテレメトリを読み取り、表示する**
サブマップコマンド	コマンドの形式: **submap add <submapId>**

	動作: 指定されたサブマップ ID のサブマップを追加するコマンドを実行する。 例: **submap add 1 - サブマップ 1 を追加する**
	コマンドの形式: **submap delete <submapId>** 動作: 指定されたサブマップ ID のサブマップを削除するコマンドを実行する。 例: **submap delete 1 - サブマップ 1 を削除する**
	コマンドの形式: **submap freeze <submapId>** アクション: 指定したサブマップ ID のサブマップをフリーズするコマンドを実行します。 例: **submap freeze 0 - サブマップ 0 をフリーズする**
	コマンドの形式: **submap unfreeze <submapId>** アクション: 指定したサブマップ ID のサブマップのフリーズを解除するコマンドを実行します。 例: **submap unfreeze 0 - サブマップ 0 のフリーズを解除する**
	コマンドの形式: **submap get <submapId>** アクション: 指定したサブマップ ID のサブマップの設定を取得するコマンドを実行します。 例: **submap get 0 - サブマップ 0 の設定を取得して表示する**
	コマンドの形式: **submap testset <submapId>** アクション: 指定された submap ID を持つ submap の設定を行うコマンドを実行します。プログラムはコマンドのテスト用に事前定義された設定を書き込みます。サンプルコードを参照してください。 例: **submap testset 0 - submap 0 の設定を変更する**
Map コマンド	コマンドの形式: **map erase** アクション: マップ消去コマンドを実行します。 例:

	map erase - Modem 内のマップを消去する
	コマンドの形式: **map freeze** アクション: マップフリーズコマンドを実行します。 例: **map freeze - マップをフリーズする**
	コマンドの形式: **map unfreeze** アクション: マップフリーズ解除コマンドを実行します。 例: **map unfreeze - マップのフリーズを解除する**
更新レートコマンド	コマンドの形式: **rate get** アクション: 更新レート設定の読み取りコマンドを実行します。 例: **rate get - 更新レート設定を読み取り、表示する**
	コマンドの形式: **rate set <value>** アクション: 更新レート設定の変更コマンドを実行します。値は Hz 単位で指定します。 例: **rate set 0.5 - 更新レートを 0.5 Hz に設定する**
超音波コマンド	コマンドの形式: **usound get <address>** アクション: 指定されたビーコンの超音波設定を読み取ります。 例: **usound get 5 - ビーコン 5 の超音波設定を読み取り、表示する**
	コマンドのフォーマット: **usound testset <address>** アクション: 指定されたビーコンの超音波設定を書き込みます。プログラムはコマンドのテスト用に事前定義された設定を書き込みます。サンプルコードを参照してください。 例: **usound testset 5 - ビーコン 5 の超音波設定を変更する**
軸接続コマンド	コマンドのフォーマット: **axes <address_0> <address_x> <address_y>** アクション: ビーコンを軸に接続するコマンドを実行します。 例: **axes 3 4 5 - ビーコン 3 を X=0、Y=0 に設定し、**

	ビーコン 4 を X 軸方向（Y=0）、ビーコン 5 を X 軸より上（Y>0）に設定する
Modem から設定メモリダンプを読み取る	コマンドのフォーマット: **read_dump <offset> <size>** アクション: Modem の設定メモリのダンプ読み取りコマンドを実行します。 例: **read_dump 0 1000 - 設定メモリの先頭から最初の 1000 バイトを読み取る**
設定メモリのテストダンプを Modem に書き込む	コマンドの形式: **write_dump_test <offset> <size>** アクション: Modem の設定メモリのダンプ書き込みコマンドを実行します。 例: **write_dump_test 0 1000 - 設定メモリの先頭から最初の 1000 バイトをテストパターンで埋める**
デバイスのソフトウェアリセット	コマンドの形式: **reset <address>** アクション: ソフトウェアリセットコマンドを実行します。 例: **reset 255 - USB 経由で接続されたデバイスのソフトウェアリセットを実行する**
気温コマンド	コマンドの形式: **temperature get** アクション: Modem から空気の温度設定を読み取る処理を実行する 例: **temperature get 空気設定の超音波温度を読み取り、表示する**
	コマンドの形式: **temperature set <value>** アクション: Modem へ空気の温度設定を書き込む処理を実行する 例: **temperature set 30 空気の温度設定を摂氏 30 度に設定する**
ビーコンの位置を設定する	コマンドの形式: **setloc <address> <X> <Y> <Z>** アクション: ビーコンの位置設定コマンドを実行する。X、Y、Z はメートル単位の座標である。 例: **setloc 12 1.51 3.45 2.0 - ビーコン 12 の位置を X= 1.51**

	m、Y= 3.45 m、Z= 2.0 m に設定する
ビーコン間の距離を設定する	コマンドのフォーマット: **setdist <address1> <address2> <distance>** アクション: ビーコン間の距離設定コマンドを実行します。Address1 および address2 はビーコンのアドレスです。Distance はメートル単位の距離です。 例: **setdist 12 13 16.5 - ビーコン 12 と 13 の間の距離を 16.5 メートルに設定します**
高さコマンド	コマンドのフォーマット: **height_h get <address>** アクション: Hedgehog の高さ取得コマンドを実行します。Address は Hedgehog のアドレスです。 例: **height_h get 15 - Hedgehog 15 の高さを読み取り、表示します**
	コマンドのフォーマット: **height_h set <address> <height>** アクション: Hedgehog の高さ設定コマンドを実行します。Address は Hedgehog のアドレスです。Height はメートル単位です。 例: **height_h set 15 2.5 - Hedgehog 15 の高さを 2.5 メートルに設定します**
	コマンドのフォーマット: **height_b get <address> <submap_id>** アクション: 固定 Beacon の高さ取得コマンドを実行します。address は Beacon のアドレスです。submap_id は Beacon が属するサブマップの ID です。 例: **height_b get 12 0 - サブマップ 0 内の固定 Beacon 12 の高さを読み取り、表示します。**
	コマンドの形式: **height_b set <address> <submap_id> <height>** アクション: 固定 Beacon の高さ設定コマンドを実行します。address は Hedgehog のアドレスです。submap_id は Beacon が属するサブマップの ID です。height はメートル単位で指定します。 例: **height_b set 12 0 5.1 - サブマップ 0 内の Beacon 12 の高さを 5.1 メートルに設定します。**
リアルタイムプレーヤーコマンド	コマンドの形式: **rtp get <address>** アクション: リアルタイムプレーヤー設定取得コマンドを実行しま

	す。address は Beacon のアドレスです。 例: **rtp get 15 - Beacon 15 のリアルタイムプレーヤー設定を読み取り、表示します。**
	コマンドの形式: **rtp testset <address>** アクション: リアルタイムプレイヤー設定コマンドの設定を実行します。アドレスはビーコンのアドレスです。プログラムはコマンドのテスト用にいくつかの事前定義された設定を書き込みます。サンプルコードを参照してください。 例: **rtp testset 15 - ビーコン 15 のリアルタイムプレイヤーテスト設定をセットアップする**
ジオリファレンスコマンド	コマンドの形式: **georef get** アクション: ジオリファレンス設定取得コマンドを実行します。 例: **georef get - ジオリファレンス設定を読み取り、表示する**
	コマンドの形式: **georef set <latitude> <longitude>** アクション: ジオリファレンス設定書き込みコマンドを実行します。 例: **georef set 10 20 – 緯度 10 度、経度 20 度のジオリファレンスを書き込む**
アップデートモードコマンド	コマンドの形式: **update_mode get** アクション: 位置更新モード取得コマンドを実行します。 例: **update_mode get - 位置更新モードを読み取り、表示します**
	コマンドの形式: **update_mode set <mode>** アクション: 位置更新モード設定コマンドを実行します。 例: **update_mode set 0 - 位置更新の自動モードを設定します**
	コマンドの形式: **update** アクション: 位置更新コマンドを実行します。 例: **update - 現在のモードに従ってモバイルビーコンの位置を更新します**
ジオフェンシングアラーム状態を設定します	コマンドの形式: **alarm <address> <mode> <zone>** アクション: ジオフェンシングアラーム状態設定コマンドを実行します。

	例: **alarm 10 2 5 - ビーコン n10 にジオフェンシングゾーン番号 5 のジオフェンシングアラーム信号を設定する**
ユーザーペイロードコマンド	コマンドの形式: **payload read <address>** 動作: ユーザーペイロードデータ取得コマンドを実行します。 例: **payload read - 任意のビーコン/Modem からユーザーペイロードを読み取る** 例: **payload read 10 - ビーコン n10 からユーザーペイロードを読み取る**
	コマンドの形式: **payload write <address>** 動作: ユーザーペイロードデータ送信コマンドを実行します。 例: **payload write 10 - ビーコン n10 にテストペイロードデータを書き込む** テストパターンは 100 から始まる 40 バイトです: 100, 101, ..., 139
手動距離測定コマンド	コマンドのフォーマット: **distance <manual/auto> <address> <max distance>** アクション: 手動距離測定コマンドを実行する。 例: **distance manual 10 5 – ビーコン 10 から他のビーコンまでの距離を測定、最大距離 5 メートル** 例: **distance manual 10 – ビーコン 10 から他のビーコンまでの距離を測定、最大距離 30 メートル（デフォルト）** 例: **distance auto - 自動距離測定モードに戻る**

10.6. デバイスタイプ

特定デバイスの「Device type ID」値の一覧:

Device type ID	デバイスの説明
22	Beacon HW V4.5
23	Beacon HW V4.5（Hedgehog モード）
24	Modem HW V4.9
30	Beacon HW V4.9
31	Beacon HW V4.9（Hedgehog モード）
32	Beacon Mini-RX
36	Beacon Mini-TX
37	Beacon-TX-IP67
41	Beacon industrial-RX
42	Super-Beacon
43	Super-Beacon（Hedgehog モード）
44	Industrial Super-Beacon
45	Industrial Super-Beacon （Hedgehog モード）
46	Super-Modem
48	Modem HW V5.1

デバイスタイプ ID は、デバイスリストおよびデバイスバージョン読み取りコマンドから取得できます。

11. ユーザーデバイスへのユーザーデータの送受信

Marvelmind は、Marvelmind システムを通じてユーザーデータを伝送するためのさまざまな方法をサポートしています。

- Modem の UART または USB を介してデータを送信し、モバイルビーコンの UART または USB を介して受信する
- モバイルビーコンの UART または USB を介してデータを送信し、Modem の UART または USB を介して受信する

Super-Modem は UDP を介したユーザーデータの送受信もサポートしています。

データ伝送のプロトコルについては、本ドキュメントの前のセクションで説明しています。

- ユーザーデバイスへのデータ伝送およびユーザーデバイスからのデータ伝送プロトコル
- 送受信データのための API 関数

Marvelmind は通信用のソフトウェアサンプルをさまざまな形式で提供しています。

サンプル	Arduino (UART)	PC / Raspberry Pi (USB)				PC / Raspberry Pi (UDP Super-Modem)
		API	C	Python	ROS/ROS2	C example
ユーザーデバイス → beacon←	+	+	+	+	+	n/a
Modem → ユーザーデバイス→	+	+	+	+	+	+
ユーザーデバイス → beacon→	+	+	-	-	+	n/a
Modem → ユーザーデバイス←	+	+	-	-	+	+

example の全リスト:

- ユーザーデータの送受信用 Arduino サンプルは、Marvelmind ソフトウェアパッケージのフォルダー「01_Common_Indoor_positioning_SW/ 06_Examples/ arduino」に収録されています。「hedgehog_sample_uart_user_data_receive_v2」はユーザーデータの受信用、「hedgehog_sample_uart_user_data_send_v2」はユーザーデータの送信用です。
- API 通信 example は、Marvelmind ソフトウェアパッケージのフォルダー「01_Common_Indoor_positioning_SW/ 05_API/example_source」（ソースコード）および「01_Common_Indoor_positioning_SW/ 05_API/example_bin_win32」（Windows 用バイナリ）に収録されています。データの送受信は、本ドキュメントに記載の方法で呼び出すことができます。
- ストリーミングデータ受信用の C example は、Marvelmind ソフトウェアパッケージのフォルダー「01_Common_Indoor_positioning_SW/ 06_Examples/ c」に収録されています。また、この example は GitHub のリポジトリでも公開されています。この example は、ユーザーデータを含む、モバイル beacon または Modem から受信したすべてのデータを出力します。
- ストリーミングデータを受信するための Python サンプルは、Marvelmind ソフトウェアパッケージのフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ python」に

収録されています。また、このサンプルは GitHub のリポジトリでも入手可能です。このサンプルは、ユーザーデータを含む、モバイルビーコンまたは Modem から受信したすべてのデータを表示するものです。

- ストリーミングデータを受信するための ROS パッケージサンプルは、Marvelmind ソフトウェアパッケージのフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ ROS」に収録されています。また、このパッケージはリポジトリでも入手可能です。ROS パッケージを使用することで、API を通じてユーザーデータの受信および送信が可能です。詳細はドキュメントを参照してください。
- ストリーミングデータを受信するための ROS-2 パッケージサンプルは、Marvelmind ソフトウェアパッケージのフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ ROS2」に収録されています。また、このパッケージはリポジトリでも入手可能です。ROS パッケージを使用することで、API を通じてユーザーデータの受信および送信が可能です。詳細はドキュメントを参照してください。
- UDP 経由でデータを受信するための C サンプルは、Marvelmind ソフトウェアパッケージのフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ c」に収録されています。このサンプルは、ユーザーデータを含む、Super-Modem または Dashboard から UDP 経由で受信したすべてのデータを表示するものです。UDP 経由でのユーザーデータの送信は、USB の代わりに UDP で Super-Modem へ API を使用して接続する場合、API を通じて実行できます。

12. お問い合わせ

追加サポートが必要な場合は、info@marvelmind.com までお問い合わせください。

付録 1. CRC-16 の計算

チェックサムには CRC-16 を使用します。N バイトフレームの末尾 2 バイトには、フレームの先頭(N-2)バイトに適用した CRC-16 の値が格納されます。データの検証を行うには、N バイトのフレーム全体に CRC-16 を適用することができ、結果の値はゼロになる必要があります。

以下は「C」言語によるアルゴリズムの実装です:

```
typedef uint16_t ModbusCrc;

typedef union {
	uint16_t w;
	struct{
		uint8_t lo;
		uint8_t hi;
	} b;
	uint8_t bs[2];
} Bytes;

static ModbusCrc modbusCalcCrc(const void *buf, uint16_t length)
{
	uint8_t *arr = (uint8_t *)buf;
	Bytes crc;

	crc.w = 0xffff;

	while(length--){
		char i;
		bool odd;

		crc.b.lo ^= *arr++;
		for(i = 0; i< 8; i++){
			odd = crc.w& 0x01;
			crc.w>>= 1;
			if (odd)
				crc.w ^= 0xa001;
		}
	}
	return (ModbusCrc) crc.w;
}
```

付録 2. Modem からのエラー応答のフォーマット

エラーフレームのフォーマット（Modem からホスト PC 向け）

オフセット	サイズ（バイト）	タイプ	説明	値
0	1	uint8_t	Modem のアドレス	0xff
1	1	uint8_t	パケットのタイプ	
2	1	uint8_t	エラーコード	
3	2	uint16_t	CRC-16（付録 1 参照）	

エラーパケットのタイプは、リクエストフレームのパケットタイプに上位ビットを加えたものです。例えば、リクエストのパケットタイプが 0x03 の場合、エラーパケットのタイプは 0x83 となります。
エラーコードは以下のいずれかになります:

1 – リクエスト内の不明なパケットタイプ
2 – リクエスト内の不明なデータコード
3 – リクエストのデータフィールドにエラーあり
6 – デバイスがビジー状態
10 – リモートデバイスからのエラーメッセージ
11 – リモートデバイスからの応答タイムアウト