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2026.10.15
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 5
   1.1 Super-BeaconのUARTおよびその他のインターフェース 6
   1.2 ビーコンMini-RXのUARTおよびその他のインターフェース 7
   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 8
   1.4. Modem HW v5.1のUARTおよびその他のインターフェース 9
   1.5. Super-ModemのUARTおよびその他のインターフェース 10
   1.6. Modem HW v4.9のUARTおよびその他のインターフェース 11
   1.7. Industrial-TX、Industrial-RX、Industrial Super-BeaconのUART等のインターフェース 12
   1.8. ビーコンHW v4.9のUARTおよびSPIインターフェース 13
   1.9. ビーコンHW v4.5のUARTおよびSPIインターフェース 14
2. システム要件 15
3. プロトコルの概要 17
4. UARTによる通信プロトコル 18
   4.1 ストリーミング用「Marvelmind」プロトコル 18
   4.2 ユーザーデバイスからの/へのデータ読み書きプロトコル 45
   4.3 NMEA0183通信プロトコル 50
   4.4 U-Blox通信プロトコル 59
5. USB（仮想UART）経由の通信プロトコル 60
   5.1 ストリーミング用「Marvelmind」プロトコル 60
   5.2 ユーザーデバイスからの/へのデータ読み書きプロトコル 60
   5.3 NMEA0183通信プロトコル 60
   5.4 USBインターフェース経由でのModemとのデータ交換プロトコル 61
   5.5 U-Blox通信プロトコル 88
6. RS-485経由の通信プロトコル 89
   6.1 ストリーミング用「Marvelmind」プロトコル 89
   6.2 ユーザーデバイスからの/へのデータ読み書きプロトコル 89
   6.3 NMEA0183通信プロトコル 90
7. SPI経由の通信プロトコル 91
   7.1 Hedgehog位置情報を含むパケット 91
   7.2 SPI経由のその他のデータ 91
8. I2C経由の通信プロトコル 92
   8.1 ドローン向けコンパスエミュレーション 92
   8.2 I2C経由のその他のデータ 92
9. UDP (Wi-Fi) 経由の通信プロトコル 93
   9.1 Hedgehog位置情報を含むパケット 94
   9.1.1. リアルタイムタイムスタンプ付きのHedgehog位置情報のパケット（ファームウェアv7.200+） 96

9.2. 固定ビーコン位置情報を含むパケット 97
9.3. 生IMUデータのパケット 98
9.3.1. リアルタイムタイムスタンプ付きの生IMUデータパケット (firmware v7.200+) 99
9.4. 生の距離データを含むパケット 100
9.4.1. リアルタイムタイムスタンプ付きの生距離データパケット (ファームウェアv7.200+) 101
9.5. IMUフュージョンデータを含むパケット 102
9.5.1. リアルタイムタイムスタンプ付きIMU融合データパケット (firmware v7.200+) 103
9.6. テレメトリデータを含むパケット 104
9.7. 品質情報および拡張位置データを含むパケット 105
9.8. 全ビーコンのテレメトリを含むパケット 106
9.9. NMEA0183プロトコル 107
10. CAN経由の通信プロトコル 108
10.1 ストリーミングの「Marvelmind」プロトコル 108
10.2 NMEA0183通信プロトコル 108
11. Dashboard csvログファイルのフォーマット 109
11.1 csvログファイルのフォーマット（Dashboardバージョン V7.000以降） 109
11.2 以前の形式のcsvログ(V7.000以前のDashboardまたはmodem HW v4.9) 122
12. Dashboardとモデム間のインターフェース 123
12.1 USB（仮想COMポート）インターフェース 123
12.2 UDPによるSuper-Modemへの接続 124
13. Marvelmind API 126
13.1 Windows用インストール 127
13.2 Linux用インストール 128
13.3 APIへの接続を確認します 129
13.4 Marvelmind APIライブラリの説明 130
13.5 Marvelmind API用Cサンプルの説明 188
13.6 デバイスタイプ 196
14. Marvelmind REST API 197
14.1 JSONデータの最上位フォーマット 198
14.2 モバイルビーコン位置データ項目 199
14.3 固定ビーコン位置データ項目 200
14.4 ビーコンテレメトリデータ項目 201
14.5 モバイルビーコン位置品質データ項目 202
14.6 IMUデータ項目 203
14.7 生距離データ項目 205
15. MQTTプロトコル 206
16. ユーザーデバイスとの間でのユーザーデータの送受信 207
17. RHU (Radio Head USB) 209
17.1 接続 210

17.2 Marvelmindプロトコル 211
17.3 NMEA0183プロトコル 216
18. 連絡先 217
付録1. CRC-16の計算 218
付録2. Modemからのエラー応答のフォーマット 219

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。

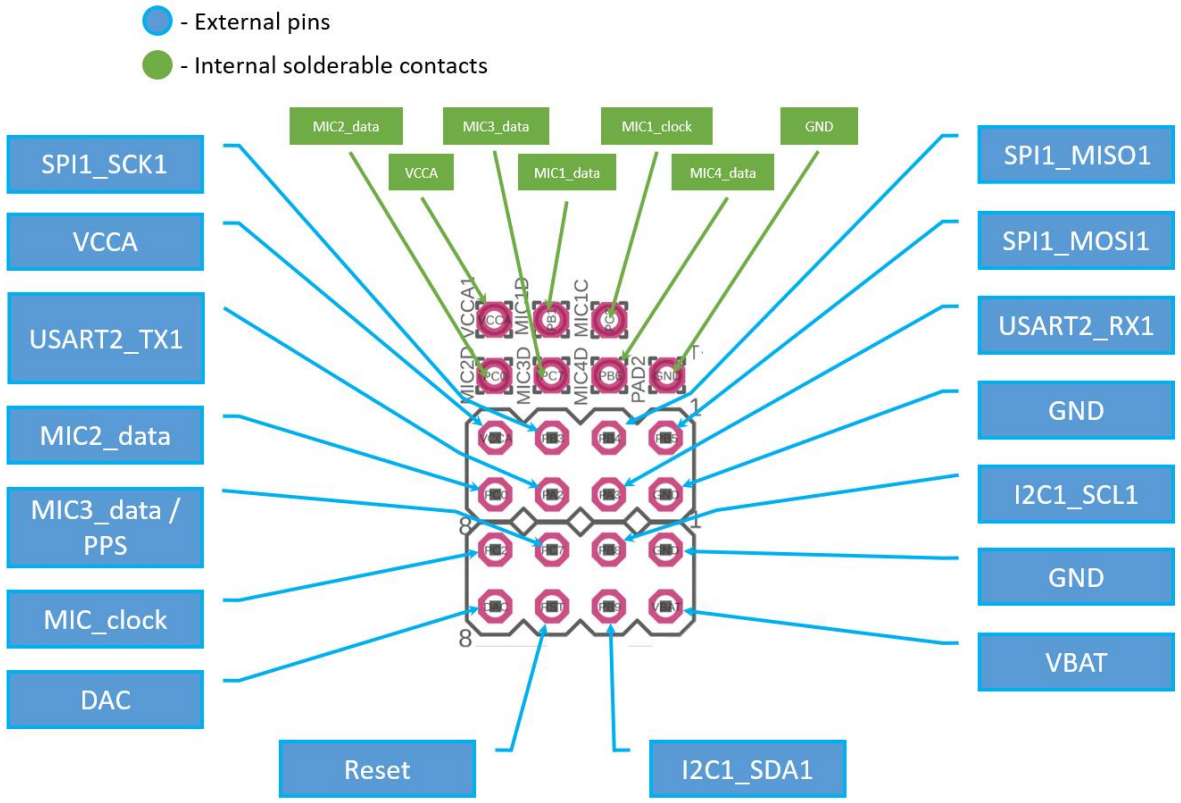
1. CDCクラスのUSBデバイス（WindowsではバーチャルCOMポート、LinuxではttyACMまたはttyUSB）としてUSBホストに接続します。Windowsではドライバーが必要であり、Modem用と同じドライバーを使用します。Linuxでは、必要なドライバーがLinuxカーネルに統合されているため、ドライバーは不要です。このインターフェースでは実際のRS-232を使用しないため、ホスト側でオープンするシリアルポートのパラメーター（ボーレート、ビット数、パリティなど）は任意の値に設定できます。
2. UARTに接続します。ストリーミング用には2本のワイヤーをピンにはんだ付けし、双方向通信には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（Super-Modem）または任意のネットワーク接続（Dashboard）に接続します。
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:

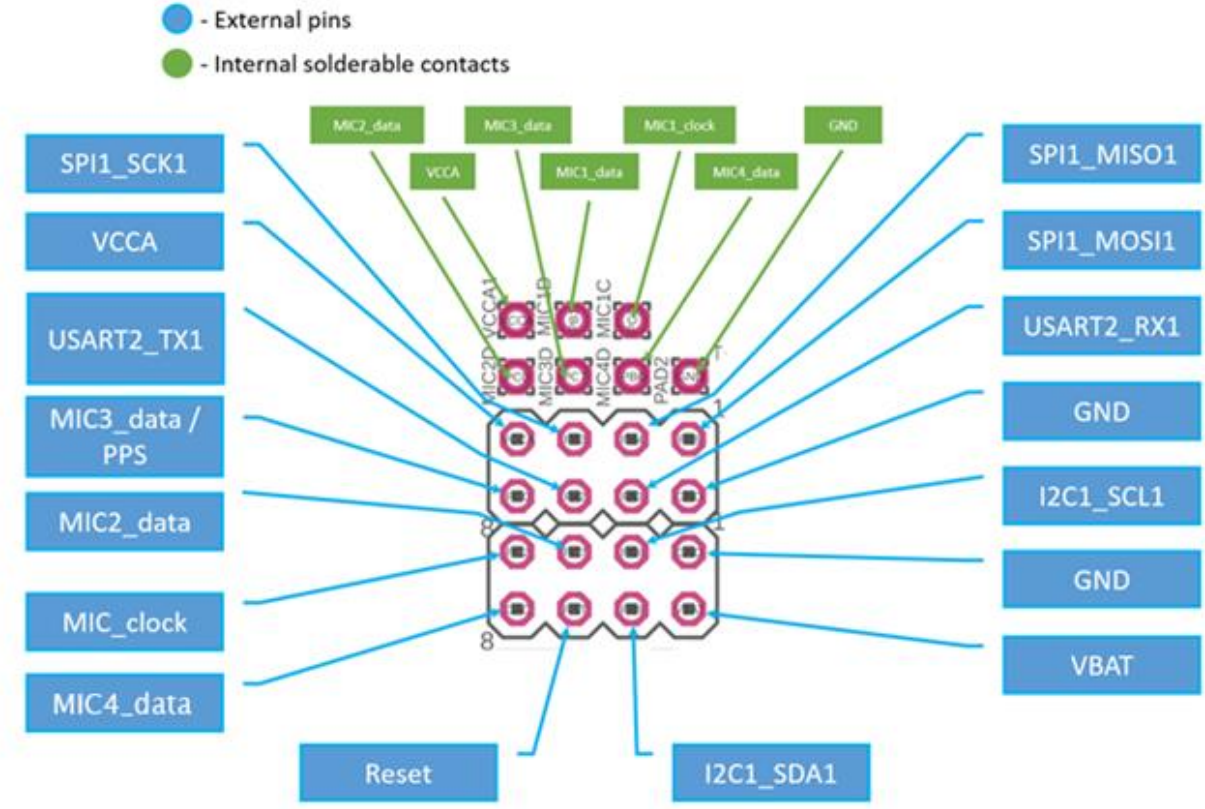
インターフェース	ビットレート	その他の設定
USB（仮想UART）	UARTのビットレートは適用されません。シリアルポートの速度は任意の値に設定できます。USBフルスピード（12 Mbit/s）によるデータ転送	UART設定は適用外
UART	4.8、9.6、19.2、38.4、57.6、115.2、500 Kbit/s Dashboardで選択可能	データ8ビット、ストップビット1、パリティなし
SPI	8 Mbit/sまで動作確認済み	SPIモード0
RS-485	4.8、9.6、19.2、38.4、57.6、115.2、500 Kbit/s Dashboardで選択可能（UARTと同様）	データ8ビット、ストップビット1、パリティなし
I^2C	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の4x4ピン配置：



Super-Beacon-2、Super-Beacon-3、Super-Beacon-4の4x4ピン配置：



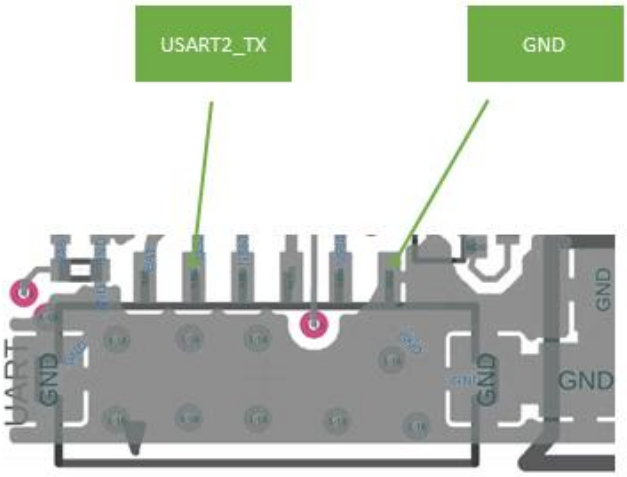
Marvelmind robotics

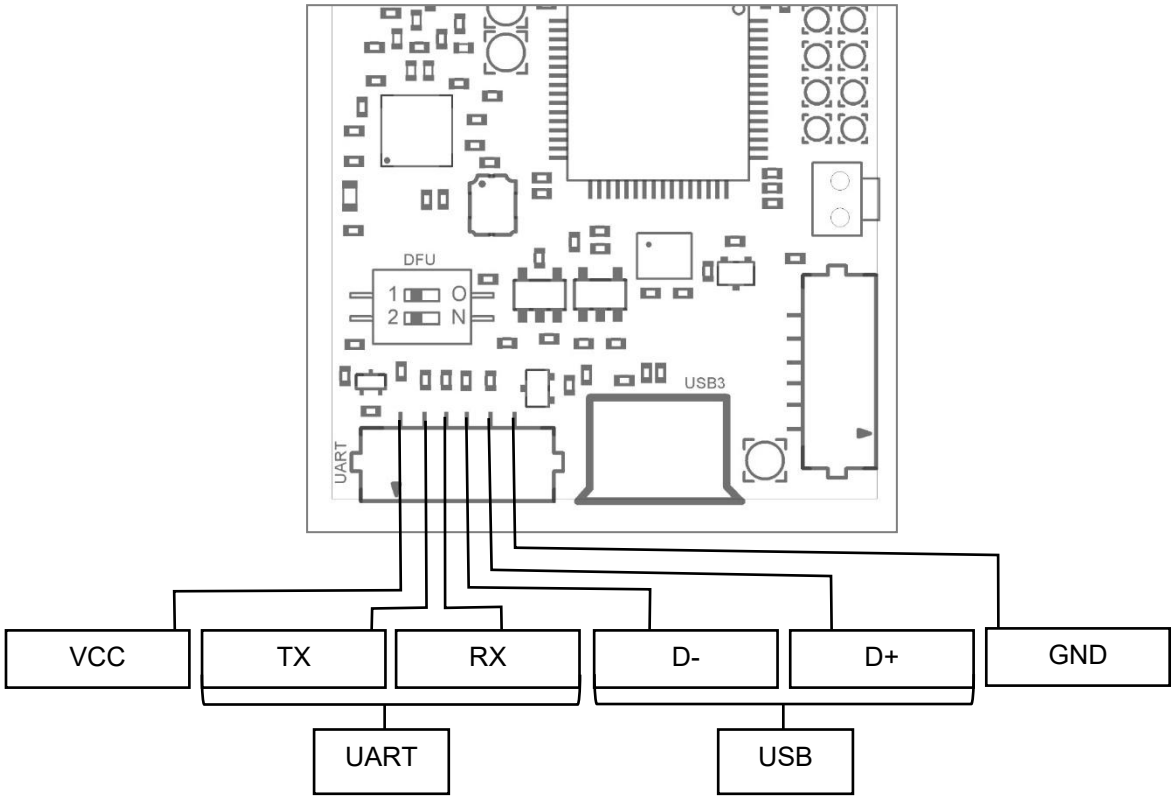
1.2　ビーコンMini-RXのUARTおよびその他のインターフェース

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。

ビーコン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





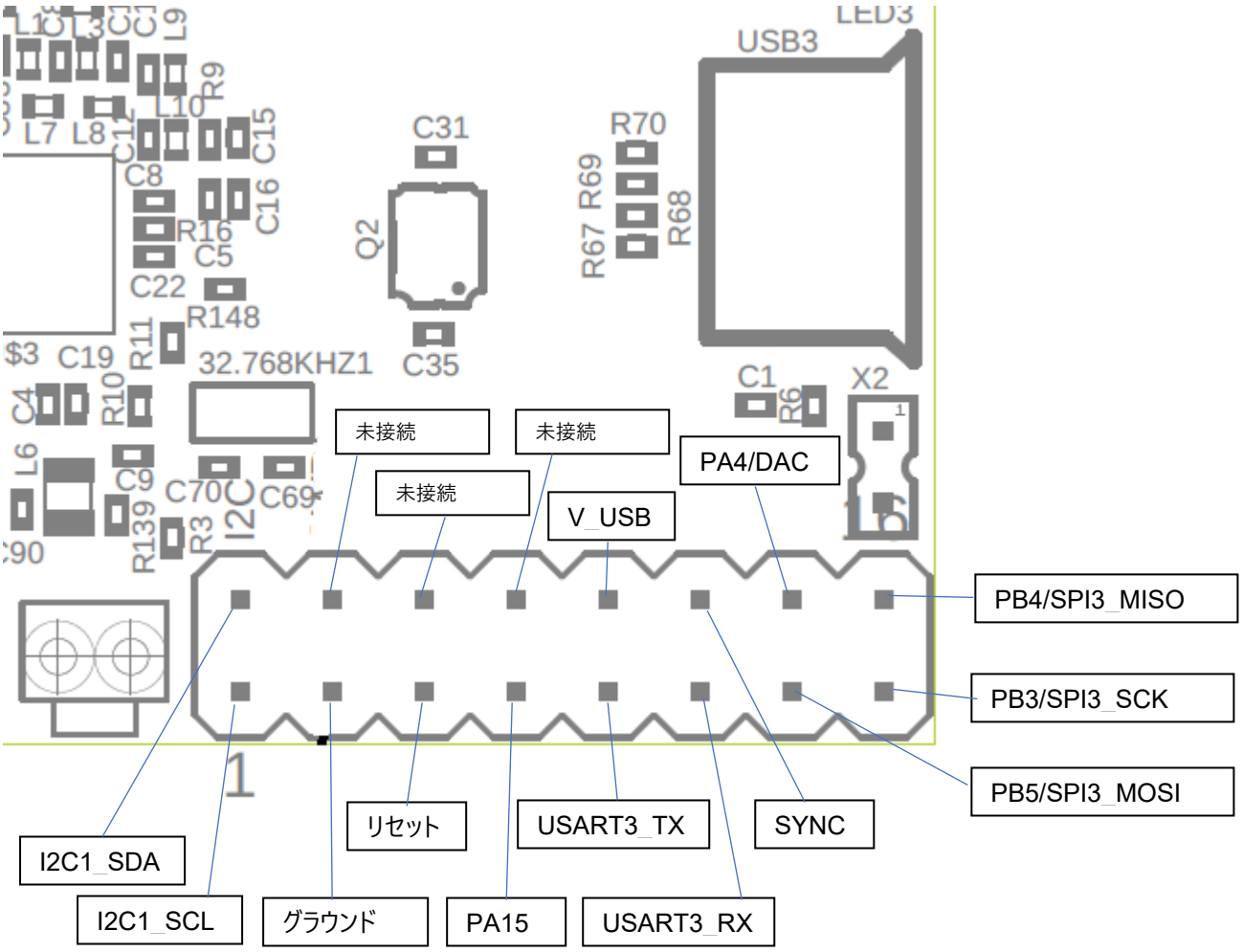
Marvelmind robotics

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と同じピン配置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。

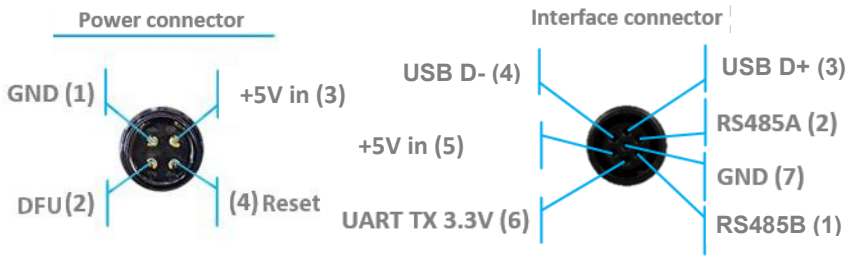
1.4. Modem HW v5.1のUARTおよびその他のインターフェース



Marvelmind robotics

1.5. Super-ModemのUARTおよびその他のインターフェース

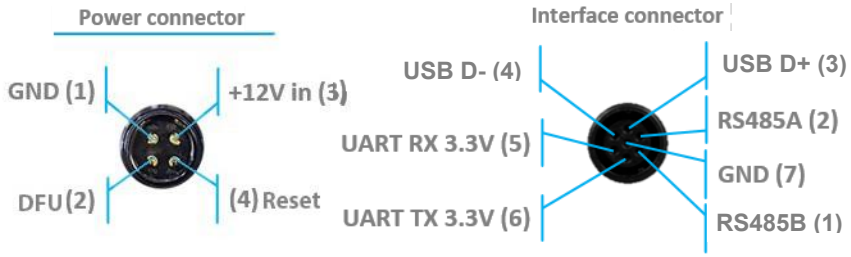
新しいSuper-Modem（2023年6月以降）のコネクタのピン配置：



Super-ModemはWi-Fiインターフェースも内蔵しています。Wi-Fi接続の設定はUDPの章で説明しています。

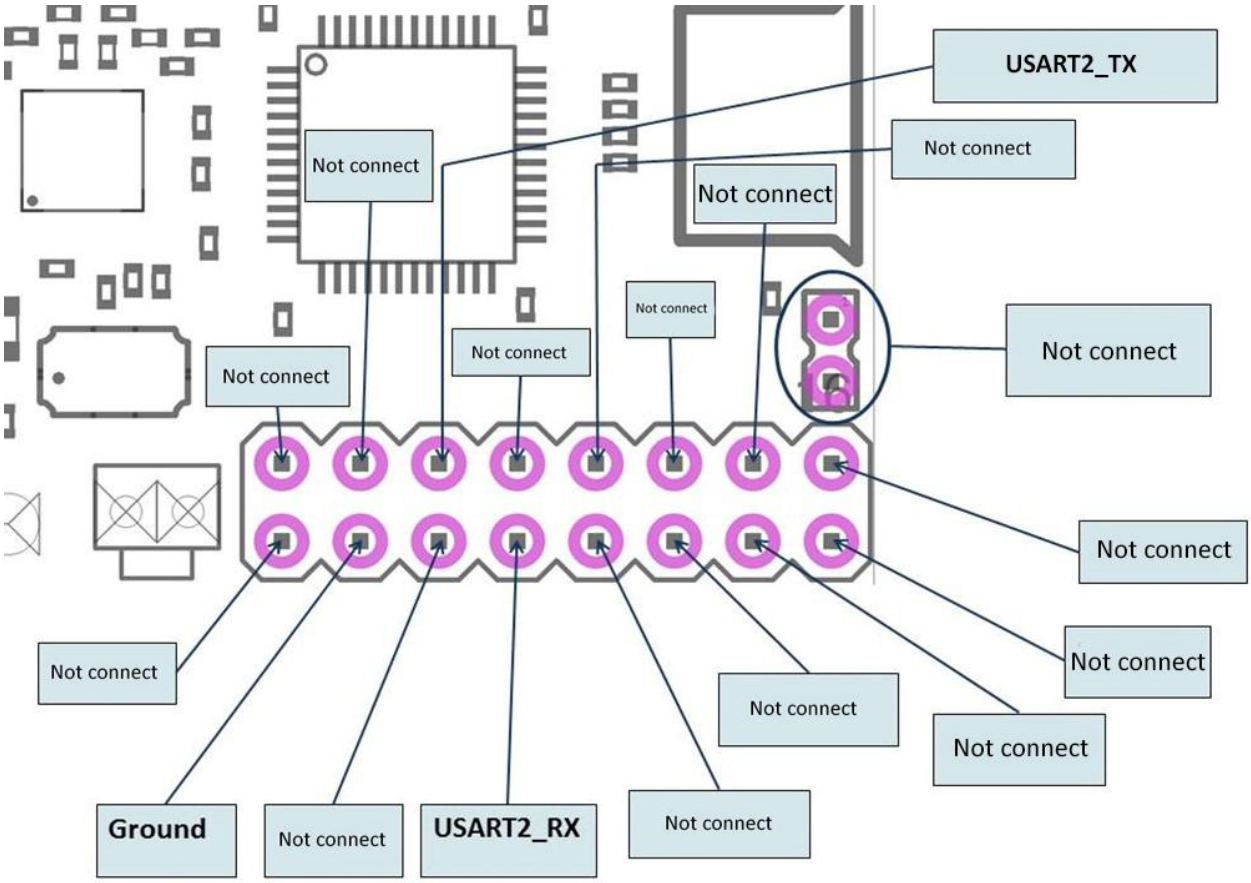
- 新しいバージョンのSuper-Modem（2023年6月以降）は**+5V**電源のみに対応しています。このバージョンに+12V電源コンバーターを使用しないでください。ビーコンが焼損します！

- RS485モデルのピン配置（2019年9月以降）



Marvelmind robotics

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等のインターフェース

この章は、ビーコンIndustrial Super-Beacon-Plastic、Industrial-RX With Omni、Industrial-RX、Industrial Super-Beacon Metal-25kHzに適用されます

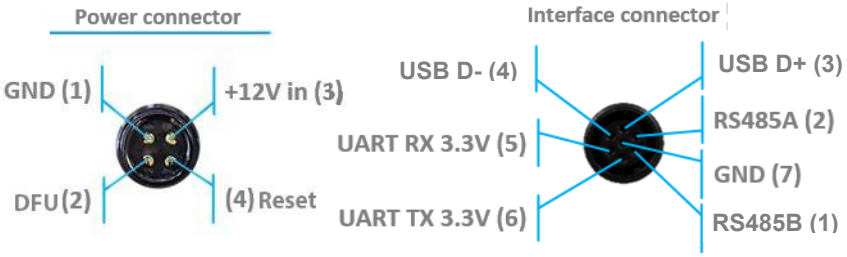
バージョン2、3、4のモデル（2022年6月以降、ステッカーの末尾が「-2」「-3」「-4」）



- Industrial-TX、Industrial-RX、Industrial Super-Beaconのバージョン2、3、4（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。ビーコンが焼損します！

- 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。

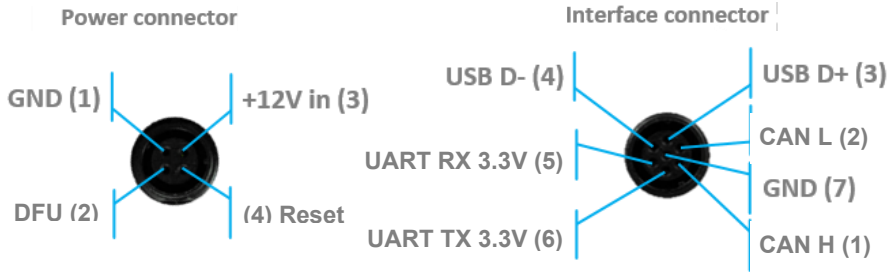
RS485モデルのピン配置（2019年9月以降）



RS485モデルのピン配置（2019年9月以前）

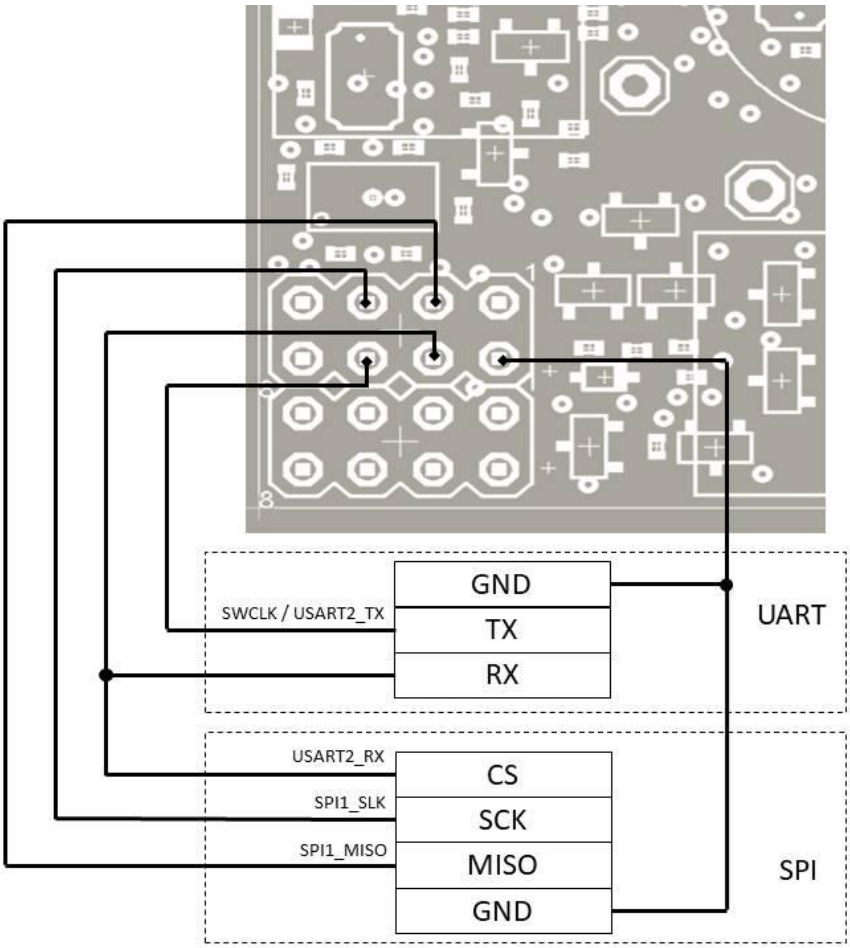


CANモデルのピン配置



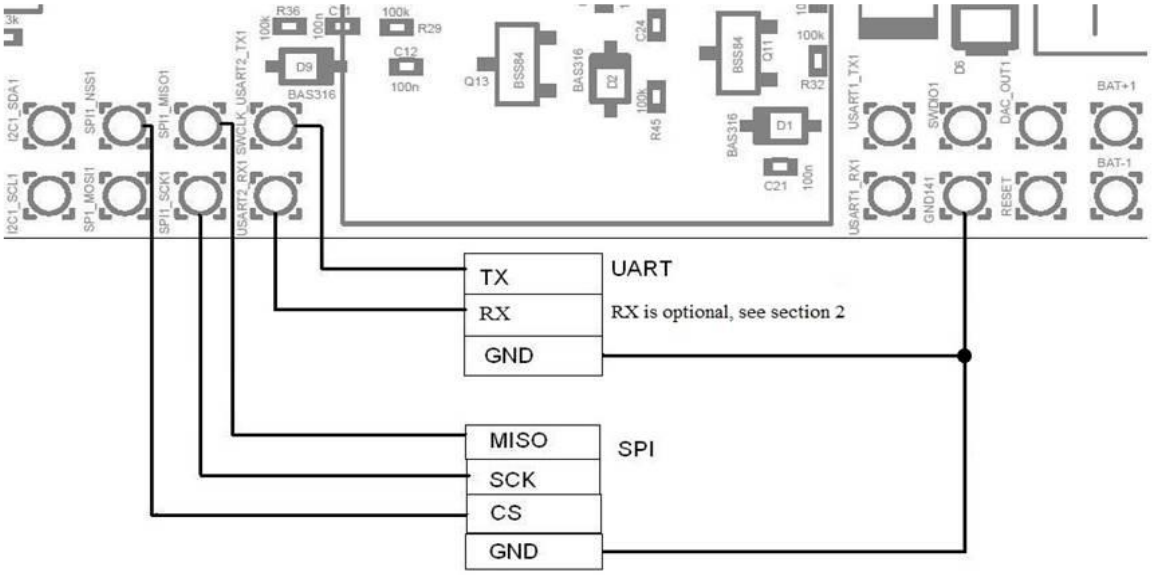
Marvelmind robotics

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



Marvelmind robotics

2. システム要件

Marvelmindナビゲーションシステムの設定と監視のためにDashboardを実行するには、以下の最小ハードウェア要件およびソフトウェア要件を満たす必要があります。

Microsoft Windowsの場合：

- OS：Windows 11 64 bit
- プロセッサ：Intel Core i5-8350Uまたは同等
- メモリ：8 GB RAM
- ディスプレイ：1920x1080ピクセル
- ストレージ：10 GBの空き容量
- 空きUSB 2.0+ポート

x86-64 PCのLinuxの場合：

- OS：Ubuntu 18.04 64 bit
- プロセッサ：Intel Core i5-4200Mまたは同等
- メモリ：4 GB RAM
- ディスプレイ：1920x1080ピクセル
- ストレージ：10 GBの空き容量
- 空きUSB 2.0+ポート

Raspberry PiのLinuxの場合：

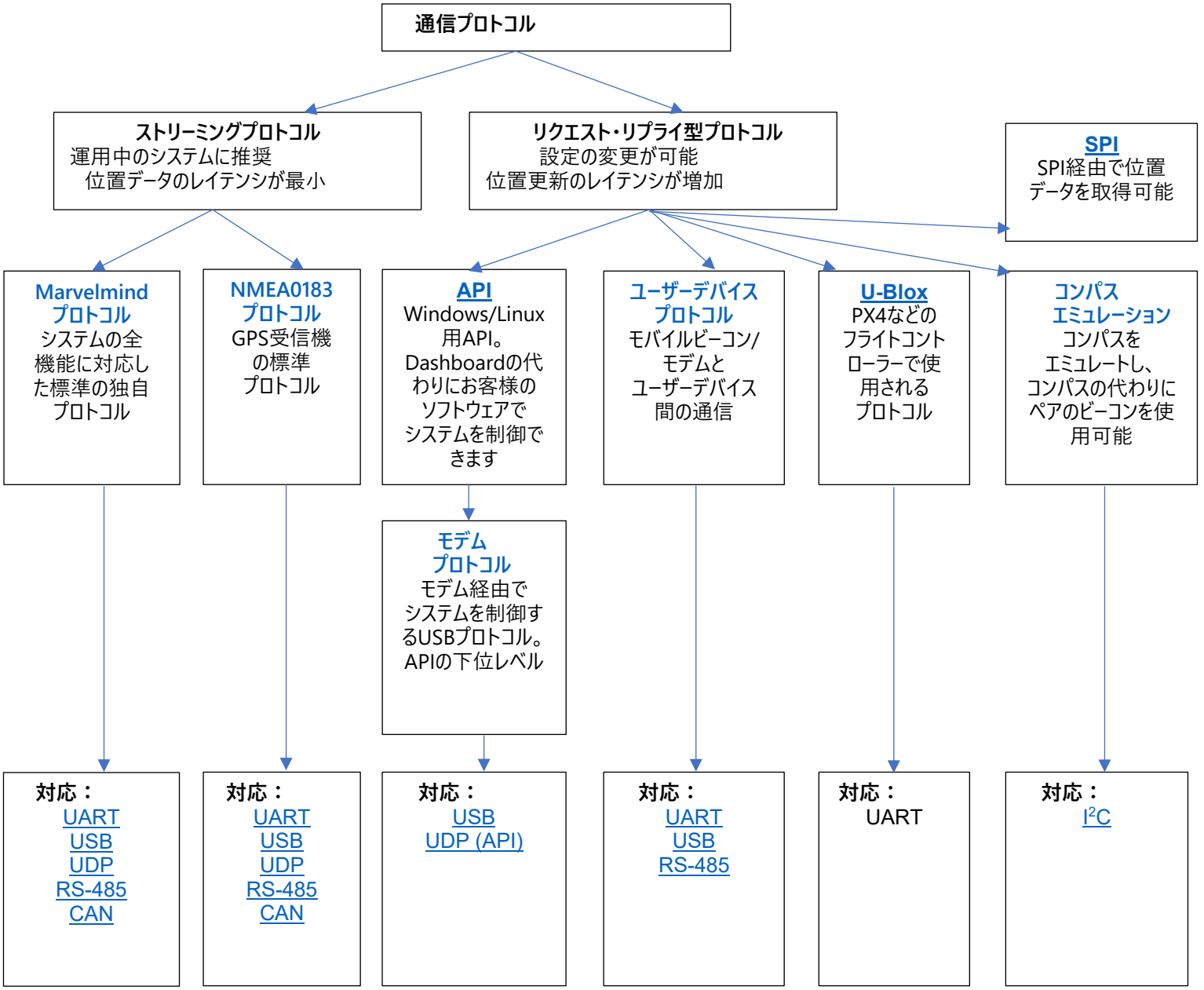
- OS：Raspberry Pi OS 64 bit
- プロセッサ：Broadcom BCM2712（Raspberry Pi 5）
- メモリ：2 GB RAM
- ディスプレイ：1920x1080ピクセル
- ストレージ：10 GBの空き容量
- 空きUSB 2.0+ポート

For MAC OS;

- OS：MacOS Tahoe 26
- 仮想マシン：VMware FusionとWindows 11 64 bit（Dashboardは仮想マシン上で動作）

- プロセッサ：Apple M5
- メモリ：16 GB RAM
- ディスプレイ：1920x1080ピクセル
- ストレージ：10 GBの空き容量
- 空きUSB 2.0+ポート

3. プロトコルの概要



Marvelmind robotics

4. UARTによる通信プロトコル

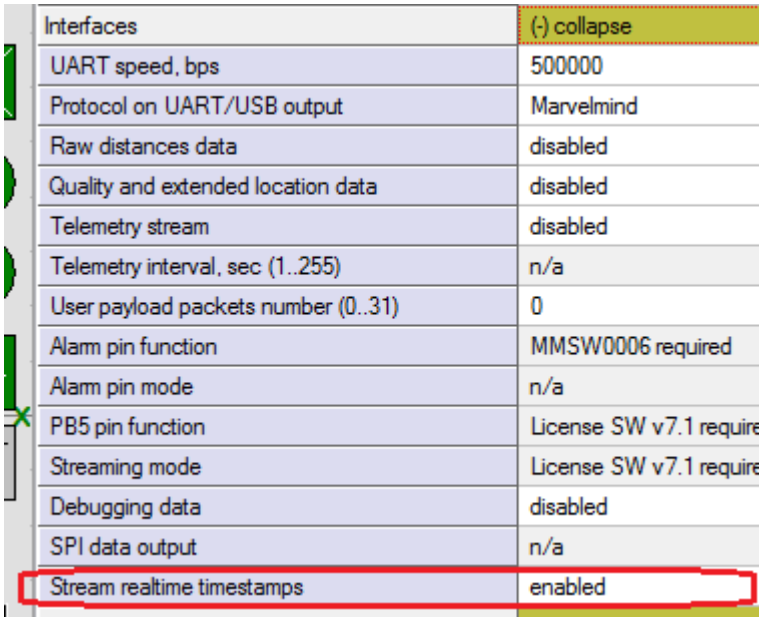
4.1 ストリーミング用「Marvelmind」プロトコル

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：

Offset	バイト数	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	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。



4.1.1 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。

4.1.1.1 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, etc.): UARTケーブルで対応
Mini-TX: 現行HWバージョンでは非対応
Mini-TX-2: UARTケーブルで対応
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参照してください。

Offset	バイト数	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Dashboard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	ビーコンのX座標、mm	
17	4	int32_t	ビーコンの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でください。 ビット1：タイムスタンプ単位（注記参照） ビット2：1 - ユーザーボタン押下（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ダウンロード要求、セクション2参照（V5.34+） ビット5：1 – 第二ユーザーボタン押下（V5.74+） ビット6：1 – 別のHedgehog向けデータ（本パケットの送信元以外） 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 - 向きは適用されない ビット14：1 – スレーブモバイルビーコンの位置データなし。向きは最大1度/secドリフトする可能性あり ビット15：reserved（0）	

Marvelmind robotics

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、次の構造を含められます：

- 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

Offset	バイト数		型	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	

4.1.1.2 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。

4.1.1.3 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。

4.1.2 固定ビーコンの座標パケット

このパケットはマップが固定されているときに送信され、10 secごとに繰り返されます。

4.1.2.1 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, Helmet, etc.): UARTケーブルで対応
Mini-TX: 現行HWバージョンでは非対応
Mini-TX-2: UARTケーブルで対応
Modem HW4.9: 対応
Beacon HW4.9: 対応
Beacon HW4.5: 対応

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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	
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：

Offset	バイト数	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	Z座標、Beaconの高さ、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：reserved

Marvelmind robotics

4.1.2.2 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。

4.1.3 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。

4.1.3.1 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」有効時)
Industrial Super-Beacon:	対応、100 Hz (「Raw inertial sensors data」有効時)
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 etc.):	対応、100 Hz (「Raw inertial sensors data」有効時) UARTケーブルを使用
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	対応、100 Hz (「Raw inertial sensors data」有効時)
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」有効時)
Beacon HW4.5:	対応、100 Hz (「Raw inertial sensors data」有効時)

タイムスタンプに関する注意事項を参照してください。

Offset	バイト数	型	説明	値
0	1	uint8_t	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	0x0083
4	1	uint8_t	送信データのバイト数	
5	36		データパケット（下記参照）	
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

Offset	バイト数	型	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	Reserved (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を持つすべてのデバイスによって同期された時刻。	
32	1	uint8_t	フラグ: ビット0：1 = 加速度計データ 利用不可	

Marvelmind robotics

			ビット1：1 = ジャイロデータ利用不可 ビット2：1 = Compassデータ利用不可 ビット3...7 – reserved（0）	
33	3	3バイト	reserved	

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。

4.1.3.2 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。

Marvelmind robotics

4.1.4 生距離データのパケット

4.1.4.1 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」で「raw distances data」が有効な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, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
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参照してください。

Offset	バイト数	型	説明	値
0	1	uint8_t	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	0x0084
4	1	uint8_t	送信データのバイト数	
5	36		データパケット（下記参照）	
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

Offset	バイト数	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	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を持つすべてのデバイスによって同期された時刻。	
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	reserved	

距離項目のフォーマット

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス（未入力時は0）	

Marvelmind robotics

1	4	uint32_t	ビーコンまでの距離、mm	
5	1	uint8_t	ビット0：1 = 距離が適用不可 ビット1...7：Reserved（0）	

Marvelmind robotics

4.1.4.2 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、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「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, etc.):	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。

Offset	バイト数	型	説明	値
0	1	uint8_t	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	0x0094
4	1	uint8_t	送信データのバイト数	
5	80		データパケット（下記参照）	
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

Offset	バイト数	型	説明	値
0	1	uint8_t	Hedgehogのアドレス	
1	17		距離項目1	
18	17		距離項目2	
35	17		距離項目3	
52	17		距離項目4	
69	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を持つすべてのデバイスによって同期された時刻。	
77	2	uint16_t	超音波送信から現在時刻までの経過時間、ミリ秒単位（V5.89+）	
79	1	uint8_t	reserved	

距離項目のフォーマット

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス（未入力時は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一候補、mm	

Marvelmind robotics

6	1	uint8_t	第一候補の品質、%	
7	4	uint32_t	ビーコンまでの距離の第二候補、mm	
11	1	uint8_t	第二候補の品質、%	
12	4	uint32_t	ビーコンまでの距離の第三候補、mm	
16	1	uint8_t	第三候補の品質、%	

4.1.4.3 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。

4.1.5 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際に送信されます。

4.1.5.1 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際に送信されます。

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, etc.):	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:	現行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。

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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

Offset	バイト数	型	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	X方向速度（フュージョン）、mm/s	
22	2	int16_t	Y方向速度（フュージョン）、mm/s	
24	2	int16_t	Z方向速度（フュージョン）、mm/s	
26	2	int16_t	ビーコンのX方向加速度、mm/s2	
28	2	int16_t	ビーコンのY方向加速度、mm/s2	
30	2	int16_t	ビーコンのZ方向加速度、mm/s2	
32	1	uint8_t	ビーコンのアドレス	
33	1	1バイト	Reserved (0)	
34	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を持つすべてのデバイスによって同期された時刻。	
42	1	uint8_t	フラグ: ビット0：1 = 位置データ利用不可	

Marvelmind robotics

			ビット1：1 = クォータニオン利用不可 ビット2：1 = 速度データ利用不可 ビット3：1 = 加速度データ利用不可	
43	3	3バイト	Reserved (0)	

注: クォータニオンは10000の値に正規化されています。

4.1.5.2 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、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参照してください。

4.1.6 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（ファームウェアV7.000+）
Super-Modem: 対応（ファームウェアV7.000+）
Mini-RX (Badge, Helmet, etc.): UARTケーブルで対応
Mini-TX: 現行HWバージョンでは非対応
Mini-TX-2: UARTケーブルで対応
Modem HW4.9: 非対応
Beacon HW4.9: 対応
Beacon HW4.5: 対応

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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

Offset	バイト数	型	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		Reserved (0)	

Marvelmind robotics

4.1.7 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, etc.): UARTケーブルで対応
Mini-TX: 現行HWバージョンでは非対応
Mini-TX-2: UARTケーブルで対応
Modem HW4.9: 対応 (qualityフィールドのみ)
Beacon HW4.9: 対応 (qualityフィールドのみ)
Beacon HW4.5: 対応 (qualityフィールドのみ)

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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

Offset	バイト数	型	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警報なし 1...255 - ジオフェンスゾーン番号 MMSW0005ライセンスが必要。	
3	13		Reserved (0)	

Marvelmind robotics

4.1.8 ポーリングリクエストに使用できる空き時間のパケット

このパケットは位置更新の後、一連のストリーミングパケットの最後に送信されます。このパケットには次のストリーミングパケットまでの時間が含まれます。この間、PCや他のデバイスはストリーミングと競合することなく、APIを使用してポーリングリクエストを送信できます。

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: 非対応
Mini-RX (Badge, Helmet, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	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	0x000a
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

Offset	バイト数	型	説明	値
0	4	uint32_t	次のストリーミングパケットまでの時間。この間バスは空いており、APIによるリクエストの送信に使用できます	
4	4	uint32_t	Reserved (0)	

Marvelmind robotics

4.1.9 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済みパケットについて説明します。

4.1.9.1 モバイルビーコンのcm分解能座標パケット

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, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
Modem HW4.9:	対応
Beacon HW4.9:	対応
Beacon HW4.5:	対応

Offset	バイト数	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	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 – ビーコンの超音波発信の内部時刻。直近のウェイクアップイベントからのミリ秒数。注記を参照してください。	
9	2	int16_t	ビーコンのX座標、cm	
11	2	int16_t	ビーコンのY座標、cm	
13	2	int16_t	Z座標、ビーコンの高さ、cm	
15	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すべきではありません。 ビット1：タイムスタンプ単位（注記参照） ビット2：1 - ユーザーボタン押下（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ダウンロード要求、セクション2参照（V5.34+） ビット5：1 – 第二ユーザーボタン押下（V5.74+） ビット6：1 – 別のHedgehog向けデータ（本パケットの送信元以外） ビット7：– reserved (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のもの; 0 – 指定されたビーコンの座標 ビット13：1 - 向きは適用されない ビット14：1 – スレーブモバイルビーコンの位置データなし。向きは最大1度/secドリフトする可能性あり ビット15：reserved	
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）	

Marvelmind robotics

4.1.9.2 モバイルビーコンのmm分解能座標パケット

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, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
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参照してください。

Offset	バイト数	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	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 – ビーコンの超音波発信の内部時刻。直近のウェイクアップイベントからのミリ秒数。注記を参照してください。	
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	Z座標、Beaconの高さ、mm	
21	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でください。 ビット1：タイムスタンプ単位（注記参照） ビット2：1 - ユーザーボタン押下（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ダウンロード要求、セクション2参照（V5.34+） ビット5：1 – 第二ユーザーボタン押下（V5.74+） ビット6：1 – 別のHedgehog向けデータ（本パケットの送信元以外） 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 (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 - 向きは適用されない ビット14～15：reserved（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（V5.88+）	
27	M= N-22		オプションデータ – リストを参照	

Marvelmind robotics

27+M	2	uint16_t	CRC-16（付録1参照）	

注：V5.20より前のファームウェアでは、タイムスタンプの単位は1/64 secで、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。バージョン5.20以降では、タイムスタンプの単位はミリ秒で、タイムスタンプ単位インジケーターは1です。

4.1.9.3 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。

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, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
Modem HW4.9:	対応
Beacon HW4.9:	対応
Beacon HW4.5:	対応

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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Beaconのデータ構造フォーマット：

Offset	バイト数	型	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	Reserved (0)

Marvelmind robotics

4.1.9.4 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」有効時)
Industrial Super-Beacon:	対応、100 Hz (「Raw inertial sensors data」有効時)
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, etc.):	対応、100 Hz (「Raw inertial sensors data」有効時) UARTケーブルを使用
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	対応、100 Hz (「Raw inertial sensors data」有効時)
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」有効時)
Beacon HW4.5:	対応、100 Hz (「Raw inertial sensors data」有効時)

タイムスタンプに関する注意事項を参照してください。

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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

Offset	バイト数	型	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	Reserved (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 利用不可 ビット1：1 = ジャイロデータ利用不可 ビット2：1 = Compassデータ利用不可 ビット3...7 – reserved（0）	
29	3	3バイト	reserved	

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。

Marvelmind robotics

4.1.9.5 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, Helmet, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
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参照してください。

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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

Offset	バイト数	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 – ビーコンの超音波送信の内部時刻、直近のウェイクアップイベントからの経過時間（ミリ秒）(V5.89+)。	
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	reserved	

距離項目のフォーマット

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス（未入力時は0）	
1	4	uint32_t	ビーコンまでの距離、mm	
5	1	uint8_t	ビット0：1 = 距離が適用不可 ビット1...7：Reserved（0）	

Marvelmind robotics

4.1.9.6 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, etc.):	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:	現行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意事項を参照してください。

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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

Offset	バイト数	型	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	X方向速度（フュージョン）、mm/s	
22	2	int16_t	Y方向速度（フュージョン）、mm/s	
24	2	int16_t	Z方向速度（フュージョン）、mm/s	
26	2	int16_t	ビーコンのX方向加速度、mm/s2	
28	2	int16_t	ビーコンのY方向加速度、mm/s2	
30	2	int16_t	ビーコンのZ方向加速度、mm/s2	
32	1	uint8_t	ビーコンのアドレス	
33	1	1バイト	Reserved (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 = クォータニオン利用不可 ビット2：1 = 速度データ利用不可 ビット3：1 = 加速度データ利用不可 ビット4...7 – reserved (0)	
39	3	3バイト	Reserved (0)	

注: クォータニオンは10000の値に正規化されます

Marvelmind robotics

4.2 ユーザーデバイスからの/へのデータ読み書きプロトコル

4.2.1 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, etc.): UARTケーブルで対応
Mini-TX: 現行HWバージョンでは非対応
Mini-TX-2: UARTケーブルで対応
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システム経由でデータを送信する場合は、次のフレームを送信します：

Offset	バイト数	型	説明	値
0	1	uint8_t	宛先アドレス	0x00
1	1	uint8_t	パケットの種類	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用のDashboard設定の「Interfaces」セクションで「User payload data size」として定義されたサイズのパーツで、無線によりmodemへ送信されます。これらのパーツの送信レートは、hedgehogの更新レートに等しくなります。hedgehog内のバッファサイズは128バイトです。バッファオーバーフローを避けるため、この点に注意してください。

Marvelmind robotics

4.2.2 ユーザーデバイスからモバイルビーコンへのメッセージ/コマンドの送信

対応ハードウェア:

Super-Beacon: 対応
Industrial Super-Beacon: 対応
Modem HW5.1: 非対応
Super-Modem: 非対応
Mini-RX (Badge, Helmet, etc.): UARTケーブルで対応
Mini-TX: 現行HWバージョンでは非対応
Mini-TX-2: UARTケーブルで対応
Modem HW4.9: 非対応
Beacon HW4.9: 非対応
Beacon HW4.5: 非対応

ユーザーデバイスがモバイルビーコンにメッセージを送信する場合は、次のフレームを送信します：

Offset	バイト数	型	説明	値
0	1	uint8_t	宛先アドレス	0x00
1	1	uint8_t	パケットの種類	0x49
2	2	uint16_t	パケット内のデータコード	0x0205
4	1	uint8_t	送信データのバイト数	4
5	1	uint8_t	メッセージの種類： 0 – フォークリフトの状態 1 – モバイルビーコンのソフトリセットを実行 2 –「zero IMU」（ジャイロスコープの キャリブレーション）を実行 3...255 - reserved	
6	1	uint8_t	メッセージの種類 == 0の場合： 0 – フォークリフトはトレーラーの外 1 – フォークリフトはトレーラーの中	
7	2	2バイト	Reserved (0)	
9	2	uint16_t	CRC-16（付録1参照）	

Marvelmind robotics

4.2.3 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またはモバイルビーコン）からユーザーデバイスへ送信されます。

Offset	バイト数	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	0x4a
2	2	uint16_t	パケット内のデータコード	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から0x2ffまでのデータコードがreservedされています。

ユーザーデバイスがリクエストを正常に処理した場合、以下の形式で応答を送信する必要があります：

Offset	バイト数	型	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パケットで取得可）	
1	1	uint8_t	パケットの種類	0x4a
2	2	uint16_t	パケット内のデータコード	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、以下の形式で応答を送信します：

Offset	バイト数	型	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	パケットの種類	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要求について説明します。

注記：ユーザーデバイスがhedgehogからの要求を処理できない場合、以下のエラーコードのいずれかを含む応答を送信する必要があります：

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

4.2.3.1 移動経路の書き込み要求

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, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
Modem HW4.9:	非対応
Beacon HW4.9:	対応
Beacon HW4.5:	対応

このパケットには基本動作のコマンドが一つ含まれます。Marvelmindデバイスは、経路内のすべての基本動作のコマンドを順に送信します。

Offset	バイト数	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	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:

Offset	バイト数	型	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 - 最初からプログラムを繰り返す 6 - 指定した位置へ移動する 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目標座標、cm	
9	3	3バイト	Reserved (0)	

Marvelmind robotics

4.2.3.2 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, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
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項目が一つ含まれます。Marvelmindデバイスは、ゾーンリストのすべてのコマンドを順に送信します。

Offset	バイト数	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	0x4a
2	2	uint16_t	パケット内のデータコード	0x202
4	1	uint8_t	送信データのバイト数	0x25
5	37	37バイト	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:

Offset	バイト数	型	説明	値
0	1	uint8_t	ゾーンのインデックス	
1	1	uint8_t	ゾーンポリゴン内の点の数 (N)	
2	1	uint8_t	本パケット内の最初の点のインデックス: M=0...N-1	
3	1	uint8_t	フラグ: ビット0：1 = 非サービスゾーン ビット1：1 = 走行禁止ゾーン 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：reserved (0)	
4	1	uint8_t	ゾーンの数	
5	32	4x8バイト	ゾーンポリゴンの最大4点 (下記参照)	

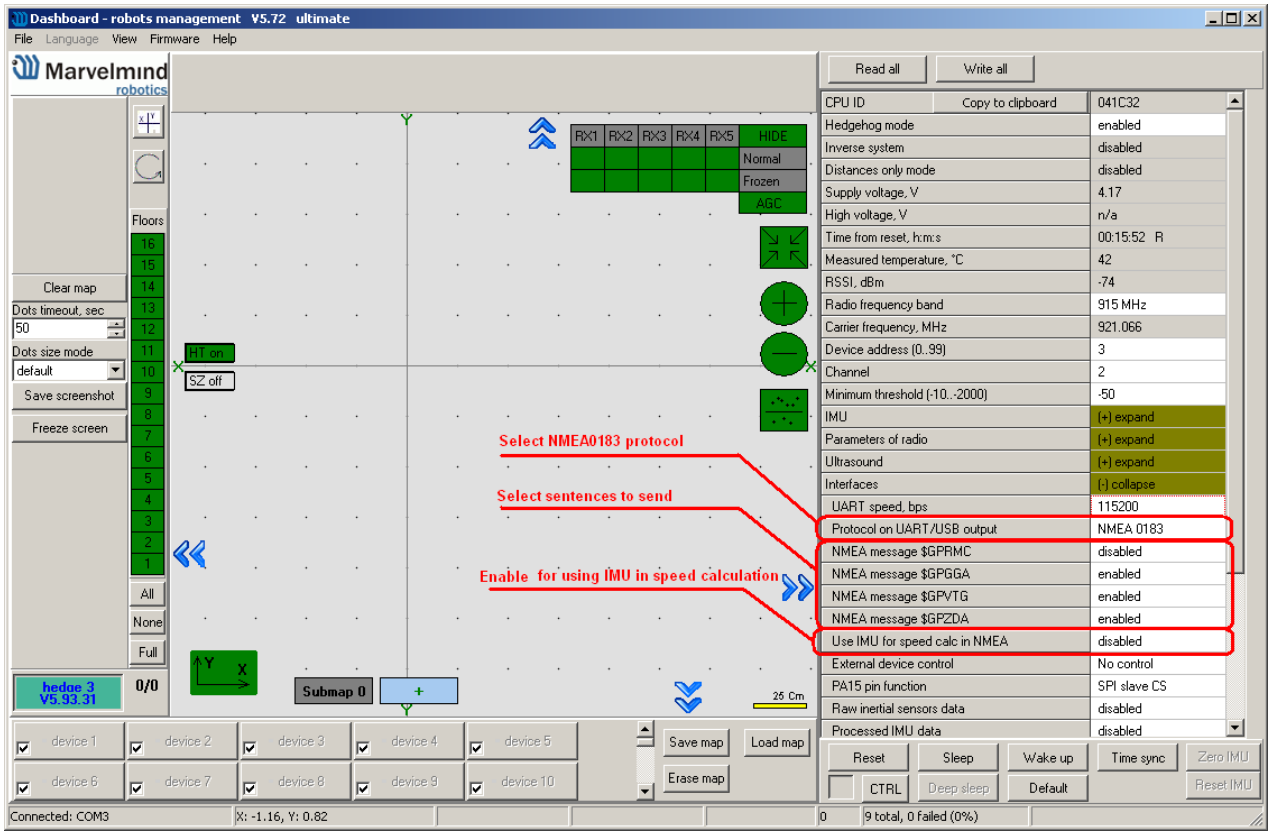
ペイロードデータのフォーマット:

Offset	バイト数	型	説明	値
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	

Marvelmind robotics

4.3 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、NMEA0183センテンスの一部を出力できます。次のスクリーンショットに示すように、Dashboardでデバイスの NMEAプロトコルを有効にする必要があります:



デバイスは、更新された位置を受信するたびに、有効になっている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、以下のインターフェースのいずれか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では仮想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ではドライバーが必要です（モデムと同じドライバー）。Linuxでは、必要なドライバーがLinuxカーネルに組み込まれているため、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は使用しないため、ホストで開くシリアルポートのパラメータ（ボーレート、ビット数、パリティなど）は任意です。
2. Hedgehog上のUARTに接続する - ピンへの2本のワイヤーのはんだ付けが必要です。下のビーコンインターフェースの図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送信機の論理レベルはCMOS 3.3Vです。デフォルトのボーレートは500 kbpsで、Dashboardから次のリスト（4.8、9.6、19.2、38.4、57.6、115.2、500 kbps）で設定可能です（上の図のパラメータ「UART speed, bps」を参照）。データ形式:8ビット、パリティなし、1ストップビット。

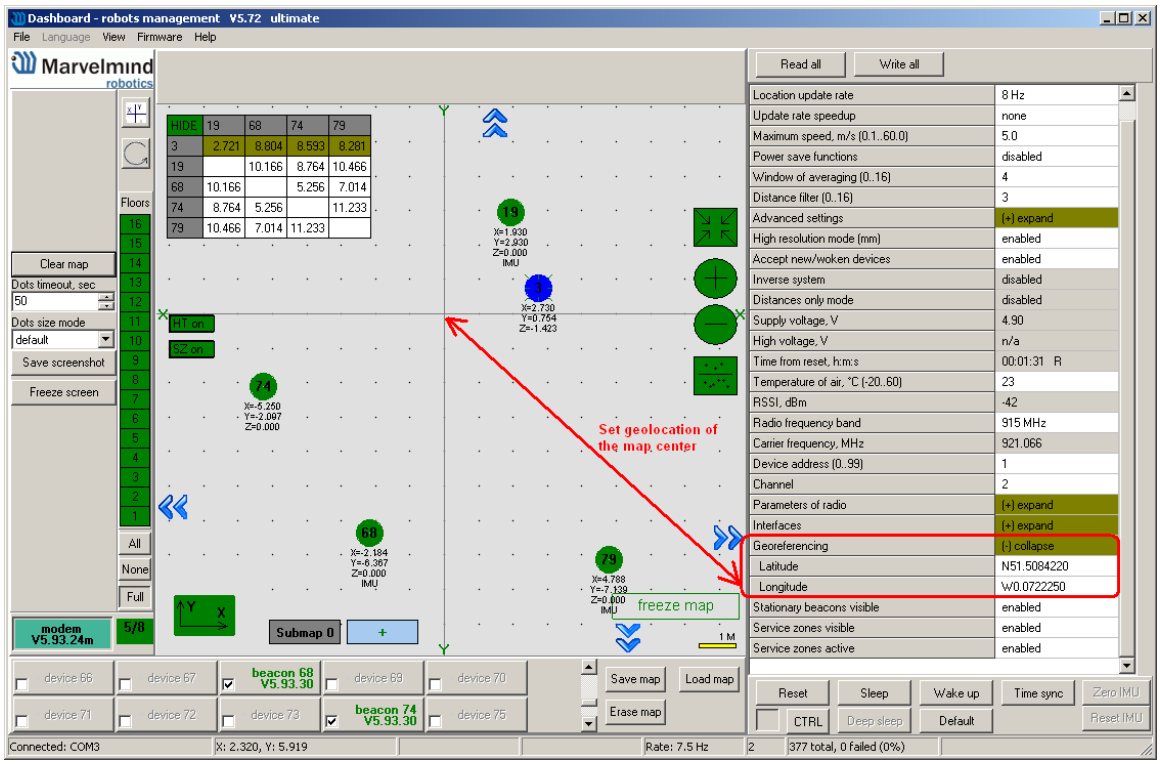
4.3.1 座標変換に関する一般的な取り決め

Marvelmindシステムは、直交座標系（X, Y, Z）の形で位置を測定します。ここでZはほとんどの場合、高さを表します。GPS座標への変換には、以下の取り決めが使用されます:

- Z軸は上向きで、Z座標は海抜高度を表します;
- Y軸は北を向いており、Yは緯度です;

- X軸は東方向を指すため、Xは経度です;
- ポイント(X= 0, Y= 0)は、ジオリファレンスポイントに従ったGPS座標を持ちます(デフォルトでは:北緯0°、西経0°');

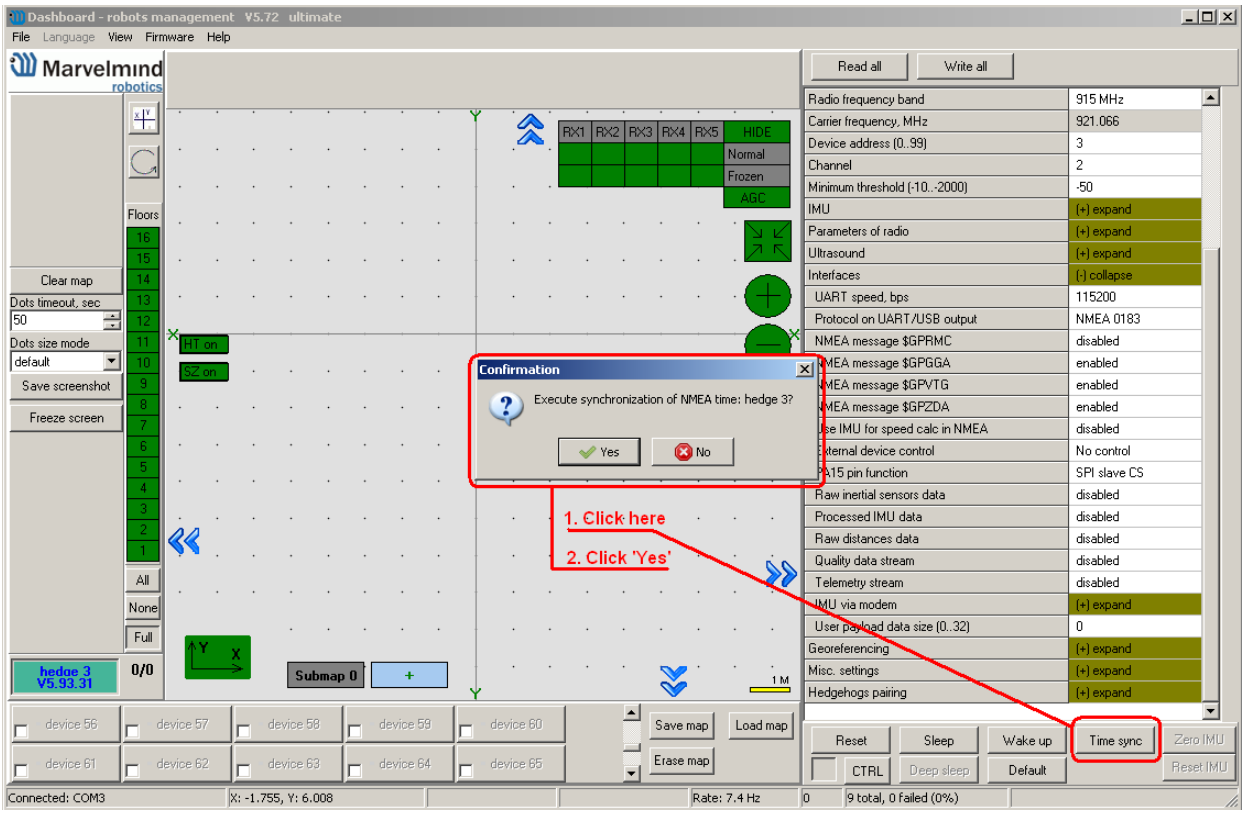
ジオリファレンス座標は、スクリーンショットに示されているように設定できます:



GPS座標は、指定されたジオリファレンスポイントとWGS-84地球モデルに従って計算されます。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

4.3.2 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から時間のカウントを開始します。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、コンピュータのクロックと時刻を同期させることができます。



4.3.3 「NMEA0183」メッセージ実装の説明

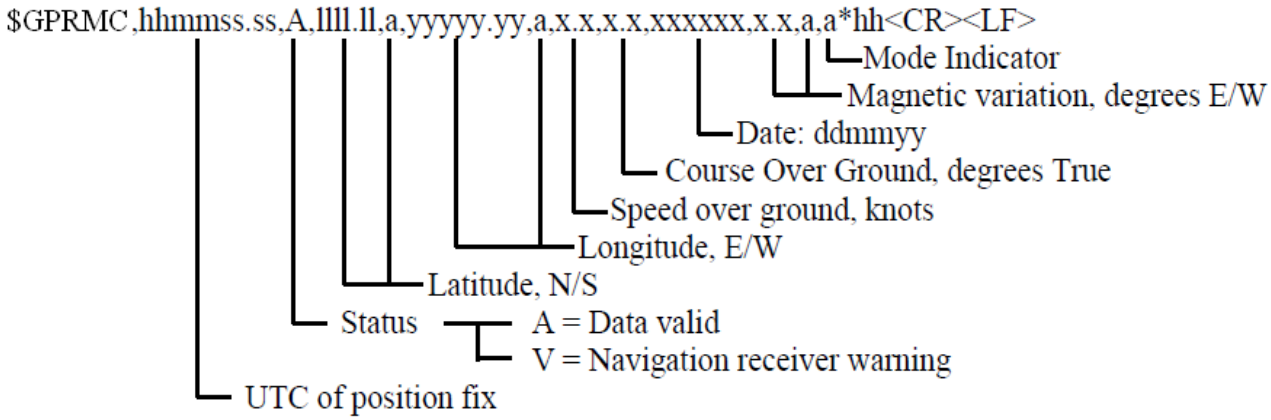
NMEA 0183メッセージは、カンマで区切られた複数の部分からなり、行末で終わるASCIIコードのテキストフレームです。各メッセージは行末の前に「*」記号で終わり、その後にNMEA 0183規格に従って計算された二文字のチェックサムが続きます。
NMEA 0183メッセージの各部分は、特定のパラメータを表します。
以下は、サポートされているすべてのメッセージとパラメータフィールドの説明です。
メッセージのフォーマットは、NMEA 0183規格バージョン3.01（2002年1月1日）に基づいています。

1. $GPRMC - 推奨最小特定GNSSデータ

対応ハードウェア:

Super-Beacon:	対応
Industrial Super-Beacon:	対応
Modem HW5.1:	サポート（SW V7.000以降）
Super-Modem:	サポート（SW V7.000以降）
Mini-RX (Badge, Helmet, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
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:

1.1. '$GPRMC' – メッセージタイプの指定

1.2. 'hhmmss.ss' – UTC位置測定時刻

一般的な取り決めに従い、時刻はデフォルトで2016.01.01から計測されるか、コンピューターのクロックと同期されます。

1.3. 'A' – ステータス

前回の位置更新が成功した場合、'A'値が送信されます
前回の位置更新でエラーが発生した場合、'V'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、緯度は地理参照位置に対するY座標に対応します。緯度は分の小数部を6桁で表現し、2 mm以下の分解能を提供します。

1.5. 'yyyyy.yyyyyy,a' – 経度、E/W

一般的な取り決め(上記参照)に従い、経度は地理参照位置に対するX座標に対応します。経度は分の小数部を6桁で表現し、2 mm以下の分解能を提供します。

1.6. 'x.x' – 対地速度、ノット

Marvelmindシステムは座標を測定します。速度は座標の動的変化から、一定のフィルタリングを適用して計算されます。オプションで、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、時刻はデフォルトで2016.01.01から計測されるか、コンピューター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

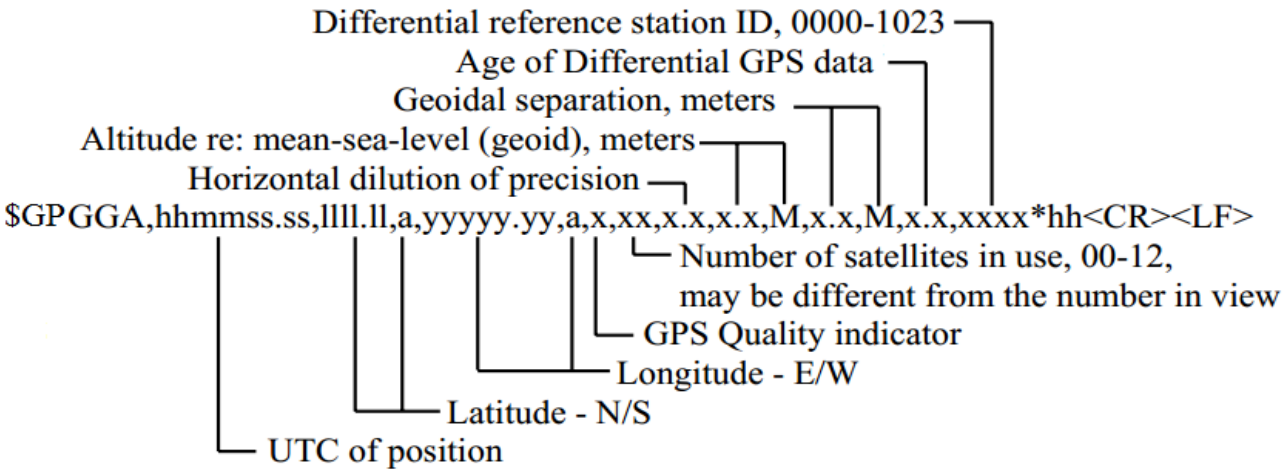
最後の位置更新が成功した場合、'A'値（自律モード）が送信されます
最後の位置更新でエラーが発生した場合、'N'値（データ無効）が送信されます

2. $GPGGA - Global Positioning System Fix Data（全地球測位システム測位データ）

対応ハードウェア:

Super-Beacon:	対応
Industrial Super-Beacon:	対応
Modem HW5.1:	サポート（SW V7.000以降）
Super-Modem:	サポート（SW V7.000以降）
Mini-RX (Badge, Helmet, etc.):	Mini-Rx用UARTケーブルで対応
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:

2.1. '$GPGGA' – メッセージタイプの指定

2.2. 'hhmmss.ss' – UTC位置測位

一般的な取り決めに従い、時刻はデフォルトで2016.01.01から計測されるか、コンピューターのクロックと同期されます。

2.3. '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桁で表現され、これにより2 mm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桁で表現され、これにより2 mm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は '1'（GPS SPSモード、fix有効）の値が送信されます。

直前の位置更新で何らかのエラーが発生した場合は '0'（Fix利用不可または無効）の値が送信されます。

Marvelmind robotics

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' – 水平精度低下率
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）に対する高度、メートル
これは一般的な取り決めに従い、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' – ディファレンシャルGPSデータの経過時間
このパラメータ値は常にnullフィールドであり、DGPSは使用されません。

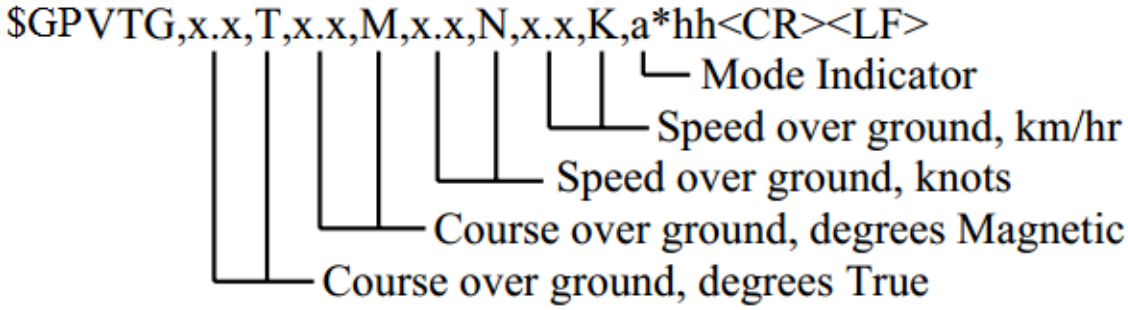
2.11. 'xxxx' – ディファレンシャル基準局ID
このパラメータ値は常にnullフィールドです。

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:	サポート（SW V7.000以降）
Super-Modem:	サポート（SW V7.000以降）
Mini-RX (Badge, Helmet, etc.):	Mini-Rx用UARTケーブルで対応
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標準によると、針路は速度ベクトルと北方向との間の角度です。上記の一般的な取り決めに示されるように、Y軸を北方向としています。

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

3.4. 'x.x, N' – 対地速度、ノット
Marvelmindシステムは座標を測定します。速度は座標の動的変化から、一定のフィルタリングを適用して計算されます。オプションで、速度計算にIMUフュージョンを使用することもできます。

3.5. 'x.x, K' – 対地速度、km/hr
単位が異なるだけで、同じ速度です

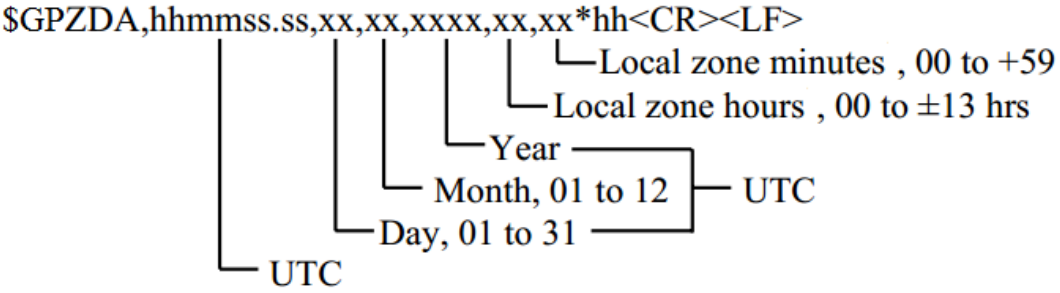
3.6. 'a' – モードインジケータ
最後の位置更新が成功した場合、'A'値（自律モード）が送信されます
最後の位置更新でエラーが発生した場合、'N'値（データ無効）が送信されます

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:	サポート（SW V7.000以降）
Super-Modem:	サポート（SW V7.000以降）
Mini-RX (Badge, Helmet, etc.):	UARTケーブルで対応
Mini-TX:	現行HWバージョンでは非対応
Mini-TX-2:	UARTケーブルで対応
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：



一般的な取り決めに従い、時刻はデフォルトで2016.01.01から計測されるか、コンピューターの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指定

4.1. 'hhmmss.ss' – UTC

時刻（時、分、秒）。

4.2. 'xx' – 日、01から31

日。

4.3. 'xx' – 月、01から12

月。

4.4. 'xxxx' – 年

年。

4.4. 'xx – ローカルゾーン（時）

ローカルゾーンは常に「00」時です。

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。

Marvelmind robotics

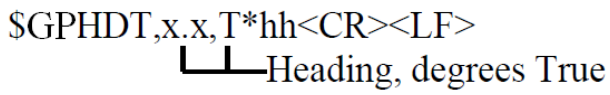
5. $GPHDT – 方位

対応ハードウェア:

Super-Beacon:	対応
Industrial Super-Beacon:	対応
Modem HW5.1:	サポート（SW V7.000以降）
Super-Modem:	サポート（SW V7.000以降）
Mini-RX (Badge, Helmet, etc.):	UARTケーブルで対応
Mini-TX:	非対応
Mini-TX-2:	UARTケーブルで対応
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規格の一般フォーマット：



5.1. '$GPHDT' – メッセージタイプの指定

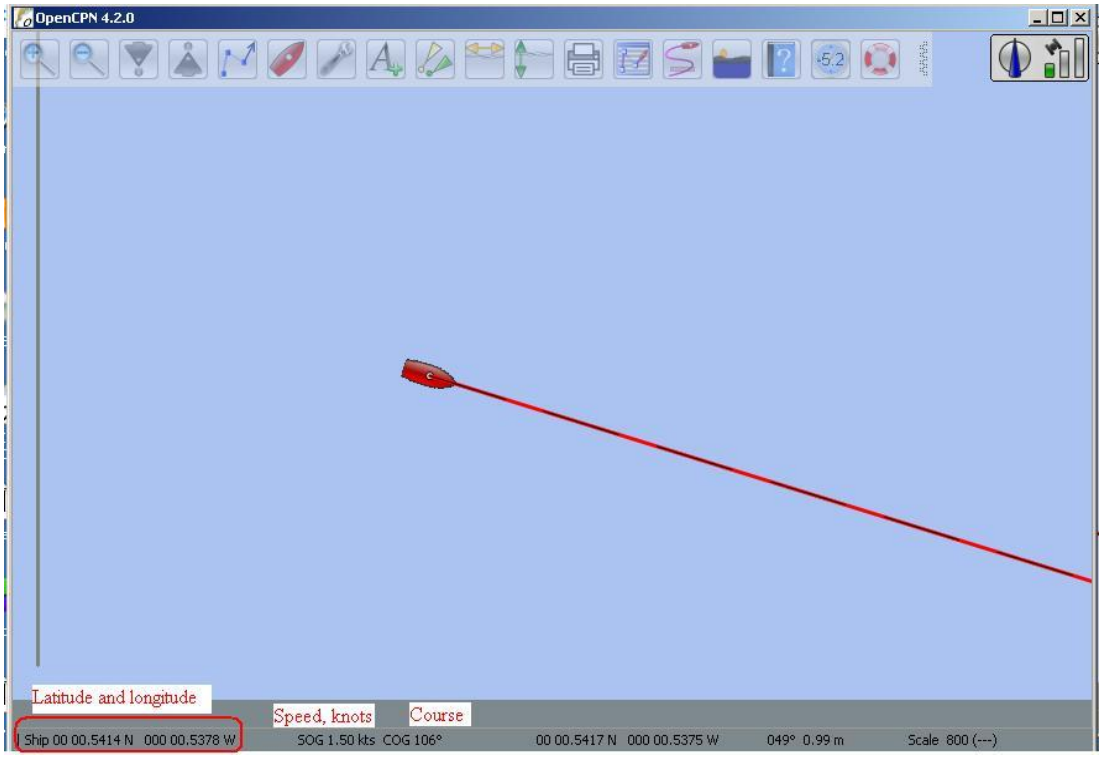
5.2. 'x.x, T' – 方位（真北基準、度）

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の融合を使用して計算された方位角です。

Marvelmind robotics

4.3.4 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MS Windows上で動作する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す。



Marvelmind robotics

4.4 U-Blox通信プロトコル

MMSW0003ライセンスが有効なモバイルビーコンは、次のU-bloxフレームを出力します：
- UBX-NAV-DOP
- UBX-NAV-SOL
- UBX-NAV-PVT

詳細はプロトコルの説明を参照してください：
https://content.u-blox.com/sites/default/files/products/documents/u-blox8-M8_ReceiverDescrProtSpec_UBX-13003221.pdf

UBX-NAV-PVTには、通常のGPS受信機と同様にLATとLONの両方および時刻が含まれます。

データは、モバイルビーコンのUARTまたはUSB（USB上の仮想UART）出力から読み取れます。
UARTを使用する場合は、Dashboardのモバイルビーコンの「Interfaces」セクションの設定と同じボーレートに合わせた受信機を使用してください。
UARTフレームは8N1（データ8ビット、パリティなし、ストップビット1）です。

5. USB（仮想UART）経由の通信プロトコル

5.1　ストリーミング用「Marvelmind」プロトコル

UARTの該当セクションのすべてのパケットは、USB（仮想UART）でも利用できます。

これらのデータは「UART Cable for Mini-Rx」なしでもmini-TX、mini-RXで利用可能です。

このプロトコルに従ってMarvelmindデバイスが何らかのリクエストを受信した場合、ストリーミングは5秒間停止します。

5.2　ユーザーデバイスからの/へのデータ読み書きプロトコル

UARTの該当セクションのすべてのパケットは、USB（仮想UART）でも利用できます。

これらのデータは「UART Cable for Mini-Rx」なしでもmini-TX、mini-RXで利用可能です。

5.3　NMEA0183通信プロトコル

UARTの該当セクションのすべてのパケットは、USB（仮想UART）でも利用できます。

これらのデータは「UART Cable for Mini-Rx」なしでもmini-TX、mini-RXで利用可能です。

このプロトコルに従ってMarvelmindデバイスが何らかのリクエストを受信した場合、ストリーミングは5秒間停止します。

5.4 USBインターフェース経由での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で説明するMarvelmind APIで使用されます。

ModemはCDCクラスのUSBデバイスとしてUSBホストに接続されます(Windowsでは仮想COMポート、LinuxではttyUSBまたはttyACM)。

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(ボーレート、ビット数、パリティなど)は任意で構いません。

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)

5.4.1 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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	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	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）

Offset	バイト数	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	0x64
3	100 (0x64)	100バイト	データ構造(下記参照)	
103	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(100バイト)

Offset	バイト数	説明
0	96 (6*16)	モデムが受信した直近六個の座標構造体（下記参照）
96	1	フラグのバイト: ビット0...1：reserved 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：reserved
97	3	Reserved

座標構造体のフォーマット(16バイト)

Offset	バイト数	説明
0	1	デバイスのアドレス
1	4	座標X、mm (int32_t)
5	4	座標Y、mm (int32_t)
9	4	座標Z、mm (int32_t)
13	1	フラグのバイト: ビット0：1 – 有効な座標なし（Dashboardでの赤モード） ビット1：1 – 固定マップ上の一時的なモバイルビーコン（青モード） ビット2：1 – Hedgehogの位置特定に使用されるビーコン
14	2	Reserved (0)

Marvelmind robotics

5.4.2 Modem構成の読み取り/書き込み

5.4.2.1 Modem構成の読み取り（ファームウェアV5.30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, etc.):	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形式（ホストからmodemへ）

Offset	バイト数	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	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）

Offset	バイト数	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。

Marvelmind robotics

5.4.2.2 Modem構成の書き込み

警告！Modem構成を書き込むには、構成を読み取り、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む必要があります。構造体内の他のバイトは変更しないでください。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形式（ホストからmodemへ）

Offset	バイト数	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	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	データ送信のバイト数	0x30
7	0x30	構造体	データ構造（以下参照）	
0x37	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）

Offset	バイト数	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	0x10
2	2	uint16_t	データコード	0x5000
4	2	uint16_t	reserved	
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。

Marvelmind robotics

5.4.2.3 Modem構成データの構造

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を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、許可されていない変更はModemの動作を低下させる可能性があります。

Offset	バイト数	型	説明
0	20	20バイト	説明なし
20	1	int8_t	空気温度設定Vt（符号付き）。 Temperature is (Vt+23) °C
21	1	uint8_t	マップ座標X=0, Y=0となるビーコンのアドレス
22	4	4バイト	説明なし
26	1	uint8_t	マップ座標X>0, Y=0となるビーコンのアドレス
27	1	uint8_t	マップ座標Y>0となるビーコンのアドレス
28	1	uint8_t	制御フラグ: ビット0：説明なし ビット1：1 - モバイルビーコンの移動フィルタリングを有効化 ビット2：説明なし 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：説明なし ビット5：1 = マップ全体のミラーリング ビット6：1 = 省電力モード（省電力はすべてのサブマップがフリーズされている場合のみ動作します） ビット7：未説明
29	2	2バイト	説明なし
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	説明なし

Marvelmind robotics

5.4.3 サブマップ構成の読み取り/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, etc.): 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

5.4.3.1 サブマップ構成の読み取り（firmware V5.30以降）

リクエストフレームの形式（ホストからmodemへ）

Offset	バイト数	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	2	uint16_t	パケット内のデータコード	0x60XX（XXはサブマップ番号）
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）

Offset	バイト数	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	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。

Marvelmind robotics

5.4.3.2 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。変更するとモデム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形式（ホストからmodemへ）

Offset	バイト数	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	0x10
2	2	uint16_t	パケット内のデータコード	0x60XX（XXはサブマップ番号）
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	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）

Offset	バイト数	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	0x10
2	2	uint16_t	データコード	0x5000
4	2	uint16_t	reserved	
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。

Marvelmind robotics

5.4.3.3 サブマップ構成データの構造

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を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、許可されていない変更はModemの動作を低下させる可能性があります。

Offset	バイト数	型	説明
0	1	uint8_t	サブマップ構築の開始ビーコンのアドレス
1	1	uint8_t	制御ワード: ビット0：1 - サブマップ凍結済み (freeze submap) ビット1：1 - ビーコンがHedgehogより高位置にある ビット2...4：説明なし ビット5：1 - サブマップのミラーリング ビット6...7：説明なし
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 bytes	説明なし
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	サブマップの回転角、センチ度
22	58	58 bytes	説明なし

Marvelmind robotics

5.4.4 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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	型	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	0x10
2	2	uint16_t	パケット内のデータコード	0xb006
4	2	uint16_t	アクセスモード	ウェイクの場合: 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リセットで復帰) 2 – 標準スリープからの復帰 3…255 - reserved	0…2
12	3	3バイト	reserved	
15	2	uint16_t	CRC-16（付録1参照）	

復帰コマンドの応答フレーム形式（モデムからホストへ）

Offset	バイト数	型	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	0x10
2	2	uint16_t	データコード	0xb006
4	2	uint16_t	reserved	
6	2	uint16_t	CRC-16（付録を参照）	

スリープコマンドに対する応答フレームの形式（modemからホストへ）

Offset	バイト数	型	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 (modemの応答)	0x7f
2	2	uint16_t	データコード	0xb006
4	2	uint16_t	reserved	
6	2	uint16_t	バイト0…5のCRC-16 (付録1を参照)	

Marvelmind robotics

8	1	uint8_t	デバイスのアドレス	0x01…0xfe
9	1	uint8_t	パケットの種類	0x10
10	2	uint16_t	データコード	0xb006
12	2	uint16_t	reserved	
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。

5.4.5 デバイスのアドレス設定

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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	型	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	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）	

5.4.6 測定された生の距離の読み取り

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, etc.): 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

このコマンドには二つのモードがあります：

- データコード0x4000 – 直近八個の距離を読み取ります。応答フレームには、リクエスト時点での直近8個の測定距離が含まれます
- データコード0x4001の場合 – すべての距離をフレームごとに読み取ります。次の各要求に対する応答フレームには、次の8つの保存された測定距離が含まれます。距離テーブル全体が送信されると、最初から再開されます

リクエストフレームの形式（ホストからmodemへ）

Offset	バイト数	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	2	uint16_t	パケット内のデータコード	0x4000か 0x4001
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）

Offset	バイト数	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	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）

Offset	バイト数	説明
0	32 (8*4)	八個の生距離データ構造体（下記参照）
32	8	Reserved

距離構造のフォーマット（4バイト）

Offset	バイト数	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	デバイス間の測定距離、mm (uint16_t)

Marvelmind robotics

5.4.7 ビーコンの状態の読み取り (ファームウェアV5.33+)

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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	型	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	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）	

応答フレームのフォーマット（modemからホストへ）

Offset	バイト数	型	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	データ送信のバイト数	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:

Offset	バイト数	型	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ー）。 R>128の場合、RSSI (dBm) = ((R-256)/2) – 74 If R<=128, RSSI (dBm) = (R/2) - 74
5	1	uint8_t	説明なし
6	1	int8_t	測定温度Vt（符号付き）。温度は(Vt+23) °Cです
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、まもなくスリープに移行 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入ります
9	23	23バイト	説明なし

Marvelmind robotics

5.4.8 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, etc.): 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

5.4.8.1 ロボット制御コマンド

リクエストフレームの形式（ホストからmodemへ）

Offset	バイト数	型	説明	値
0	1	uint8_t	ロボットのアドレス	0x01...0xfe
1	1	uint8_t	パケットの種類	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	データ送信のバイト数	0x10
7	16 (0x10) バイト	uint8_t	ロボット制御データ(下記参照)	
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）

Offset	バイト数	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 (modemの応答)	0x7f
2	2	uint16_t	データコード	0x1000
4	2	uint16_t	reserved	
6	2	uint16_t	バイト0...5のCRC-16（付録1を参照）	
8	1	uint8_t	ロボットのアドレス	0x01...0xfe
9	1	uint8_t	パケットの種類（ロボット応答）	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:

Offset	バイト数	型	説明
0	1	uint8_t	制御モード: 0 - 制御なし（待機モード） 1 - モーター電源制御 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 - 一時停止後の移動継続
1	1	uint8_t	操作コード: 0 - 前進 1 - 後退

Marvelmind robotics

			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 - 移動プログラムを最初から繰り返す（モード3用） 6 - 座標で指定された地点へ移動（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：モーターの電源投入、% 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の番号（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：プログラム内の総ステップ数。
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	Reserved (0)

この複雑なコマンドに関するいくつかのコメント。
ロボット制御構造体のバイト0で指定する、ロボット制御の主なモードは三つあります：

- 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、主にテスト用途で使用されます。モード1では、ロボットは選択されたモーター出力で前進、後退、左回転、または右回転します。モード2では同じ動作を行いますが、選択された速度を維持するように出力を調整します。パワーまたは速度は構造体のバイト2で設定され、移動の種類はバイト1で設定されます。

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継続するための特別なコマンドです。

複雑な軌道上を移動するための主なモードはモード3です。
基本動作のシーケンスをロボットにプログラムでき、その組み合わせで軌跡を構成します。
シーケンスの各項目は、この種類のコマンド一つで送信します。各コマンドには、ロボット制御構造体のバイト2に現在のステップ番号を、バイト5に総ステップ数を格納します。

ロボット制御構造体のバイト1には、プリミティブ移動の種類が指定されます。プリミティブ移動のパラメータは、「プログラム用データ」(バイト3～4) および「プログラム用追加データ」(バイト6～7) フィールドで指定されます。

したがって、以下のプリミティブが利用可能です：

- 指定された距離だけ前進する；
- 指定された距離だけ後退する；
- 指定された角度だけ時計回りに回転する；
- 指定された角度だけ反時計回りに回転する；
- 指定され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開する（ループ移動用）；
- Marvelmindナビゲーションシステムの座標で指定された点（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後、プログラムの実行を開始します。プログラム実行後、ロボットは停止します。ただし、プログラムに操作コード5（最初から繰り返し）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をアップロードするまで、プログラムはループを永続的に繰り返し実行します。

5.4.8.2 ロボットを停止

リクエストフレームの形式（ホストからmodemへ）

Offset	バイト数	型	説明	値
0	1	uint8_t	ロボットのアドレス	0x01…0xfe
1	1	uint8_t	パケットの種類	0x10
2	2	uint16_t	パケット内のデータコード	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バイト	4バイト	Reserved (0)	0
11	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）

Offset	バイト数	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 (modemの応答)	0x7f
2	2	uint16_t	データコード	0x403
4	2	uint16_t	reserved	
6	2	uint16_t	バイト0…5のCRC-16 (付録1を参照)	
8	1	uint8_t	ロボットのアドレス	0x01…0xfe
9	1	uint8_t	パケットの種類（ロボット応答）	0x10
10	2	uint16_t	データコード	0x403
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。

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。

Marvelmind robotics

5.4.9 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+）

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, etc.): 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

5.4.9.1 デバイス制御設定の読み取り

リクエストフレームの形式（ホストからmodemへ）

Offset	バイト数	型	説明	値
0	1	uint8_t	デバイスアドレス（ビーコン/モデム）	0x01...0xfeか0xff
1	1	uint8_t	パケットの種類	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からホストへ）

Offset	バイト数	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を参照）。リクエストがmodem（0xff）に送信された場合のみ該当	
11	2	uint16_t	バイト0...10のCRC-16(付録1参照)	
リクエストがビーコンに送信された場合、以下のデータが受信されます				
13	1	uint8_t	デバイスのアドレス	0x01...0xfe
14	1	uint8_t	パケットの種類	0x03
15	1	uint8_t	データ送信のバイト数	0x08/0x10
16	16	構造体	データ構造（以下を参照）。	
32	2	uint16_t	バイト12～22の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。

Marvelmind robotics

5.4.9.2 デバイス制御設定の書き込み

リクエストフレームの形式（ホストからmodemへ）

Offset	バイト数	型	説明	値
0	1	uint8_t	デバイスアドレス（ビーコン/モデム）	0x01…0xfeか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	0x1201
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	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）

Offset	バイト数	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beacon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を参照)	
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	パケットの種類	0x10
10	2	uint16_t	データコード	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。

Marvelmind robotics

5.4.9.3 制御設定ペイロードデータのフォーマット

Offset	バイト数	型	説明
0	1	uint8_t	フラグ: ビット0...5：説明なし、常にゼロにしてください！ ビット6：0 - 据置型ビーコンモード、1 - Hedgehogモード ビット7：reserved (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: reserved
2	1	uint8_t	Reserved (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: reserved ビット4...6：無線バンド: 0: 433 MHz 1: 868 MHz 2: 915 MHz 3: 315 MHz 4...7: reserved ビット7：reserved
4	1	uint8_t	UART/USB出力のタイプ: 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：reserved (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：生の加速度計データ ビット5：生のジャイロデータ ビット6：生のコンパスデータ ビット7：0 = IMUフュージョン送信、1 = 生IMU送信
8	1	uint8_t	ビット0...6：テレメトリストリーミングの間隔 (0 = ストリームなし)

Marvelmind robotics

			ビット7：reserved (0)
9	1	uint8_t	ビット0：速度計算にIMUを使用 ビット1…7 – reserved (0)
10	6	6バイト	Reserved (0)

警告！無線で接続されているビーコンの無線プロファイルを変更すると、無線接続が失われます。プロファイルを切り替える必要がある場合は、すべてのビーコンの無線プロファイルを順番に切り替えた後、Modemの無線プロファイルを切り替えてください。数秒後には、すべてのビーコンが新しい無線プロファイルで利用可能になります。

5.4.10　ネットワーク内のデバイス一覧の読み取り（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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	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	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）

Offset	バイト数	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	0x72
3	1	uint8_t	ネットワーク内のデバイス総数 (K)	
4	112	112バイト	0～16個のネットワーク内デバイス情報構造体、下記の説明を参照	
116	1	uint8_t	Reserved	0x00
117	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。

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）

Offset	バイト数	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

Marvelmind robotics

3	1	ビット0...5：デバイスのタイプ： 10: 車輪型ロボット 12: クローラー型ロボット 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ビーコン 36: Mini TXビーコン (HW V5.07) 37: Industrial-TXビーコン 41: Industrial-RXビーコン 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

Marvelmind robotics

5.4.11 ファームウェアのリーディングバージョン

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, etc.): 対応
Mini-TX: 対応
Mini-TX-2: 対応
Modem HW4.9: 対応
Beacon HW4.9: 対応
Beacon HW4.5: 対応

リクエストフレームの形式（ホストからmodemへ）

Offset	バイト数	型	説明	値
0	1	uint8_t	デバイスのアドレス	0xff
1	1	uint8_t	パケットの種類	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からホストへ）

Offset	バイト数	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	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	Reserved	
8	1	uint8_t	デバイスタイプID	
9	2	uint16_t	Reserved	
11	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。

Marvelmind robotics

5.4.12 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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	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	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からホストへ）

Offset	バイト数	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	Reserved (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、以下の構造のレコードのシーケンスです:

Offset	バイト数	型	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	

Marvelmind robotics

5.4.13　手動でのデバイス位置の書き込み

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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス	0x01…0xfe
1	1	uint8_t	パケットの種類	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	構造体	データ構造（以下参照）	
39	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）

Offset	バイト数	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	0x10
2	2	uint16_t	データコード	0x5003
4	2	uint16_t	reserved	
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。

データ構造の形式：

Offset	バイト数	型	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	説明なし	0xff
13	4	int32_t	説明なし	0
17	4	int32_t	説明なし	0
21	4	int32_t	説明なし	0
25	1	uint8_t	説明なし	0x02
26	6	6バイト	Reserved	0

Marvelmind robotics

5.4.14　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, etc.): 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形式（ホストからmodemへ）

Offset	バイト数	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	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	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）

Offset	バイト数	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	0x10
2	2	uint16_t	データコード	0x4003
4	2	uint16_t	reserved	
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。

データ構造の形式：

Offset	バイト数	型	説明	値
0	1	uint8_t	一番目のビーコンのアドレス	
1	1	uint8_t	二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	Reserved	0

Marvelmind robotics

5.5 U-Blox通信プロトコル

UARTの該当セクションのすべてのパケットは、USB（仮想UART）でも利用できます。

6. RS-485経由の通信プロトコル

6.1 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, etc.):	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の該当セクションのすべてのパケットは、RS-485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。

6.2 ユーザーデバイスからの/へ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:	要望に応じて対応
Modem HW5.1:	非対応
Super-Modem:	要望に応じて対応
Mini-RX (Badge, Helmet, etc.):	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用の該当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RS-485ハードウェアを搭載している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。

6.3 NMEA0183通信プロトコル

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:	サポート（SW V7.000以降）
Mini-RX (Badge, Helmet, etc.):	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の該当セクションのすべてのパケットは、RS-485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。

7. SPI経由の通信プロトコル

7.1 Hedgehog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, etc.):	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に対応します。Modem HW5.1はハードウェアによるSPIサポートを備えており、ソフトウェアサポートは要望に応じて追加可能です。

7.2 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, etc.):	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他のデータパケットのサポートは、Super-Beaconおよびmodem HW5.1向けに要求に応じて追加可能です

8. I2C経由の通信プロトコル

8.1 ドローン向けコンパスエミュレーション

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, etc.):	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

ペアのSuper-Beaconは、I2CでPX4またはArdupilotに接続し、より安定した高精度のコンパスとして使用できます。

IST8310コンパスをエミュレートします。

詳細はこちらのドキュメントを参照してください。

8.2 I2C経由のその他のデータ

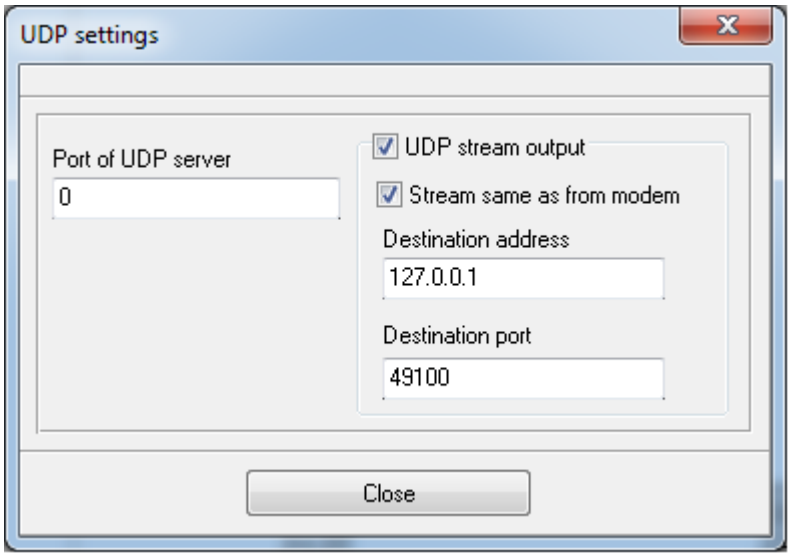
対応ハードウェア:

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, etc.):	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他のデータパケットのサポートは、Super-Beaconおよびmodem HW5.1向けに要求に応じて追加可能です

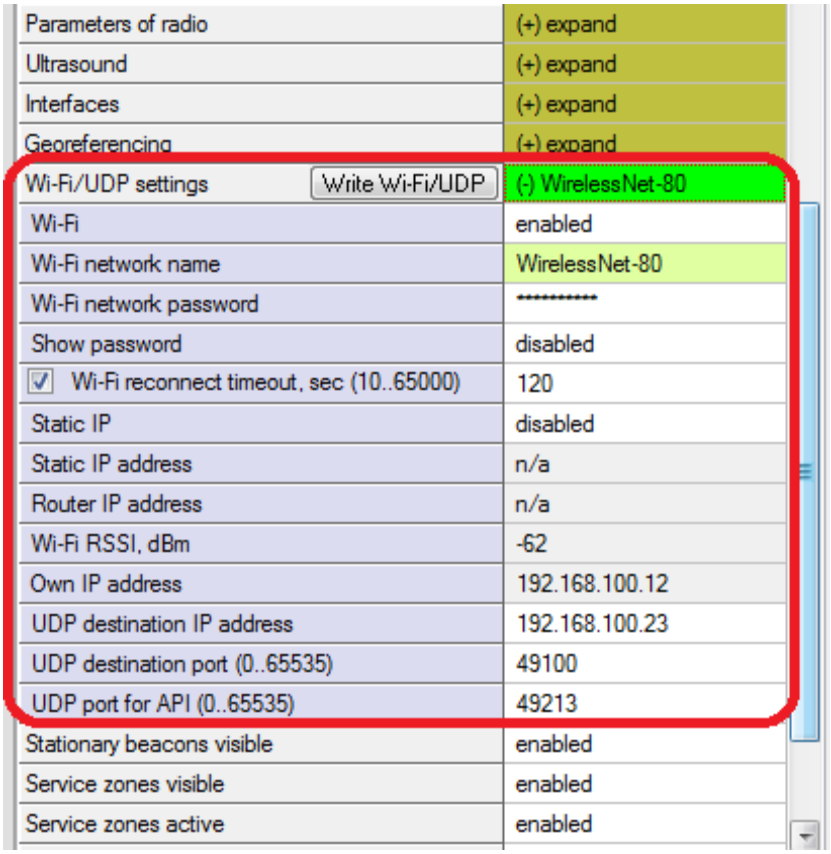
9. UDP (Wi-Fi) 経由の通信プロトコル

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介してUDP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File/UDP settingsから調整できます:



Super-Modemは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説明するその他のデータをストリーミングすることが可能です。

WiFiネットワークおよびUDPストリーミングの設定は、DashboardのSuper-Modem設定から行うことが可能です:



Marvelmind robotics

9.1 Hedgehog位置情報を含むパケット

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, etc.): 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

Offset	サイズ	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットの種類	0x47
2	2	uint16_t	パケット内のデータコード	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 (ミリ秒単位)	
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でください。 ビット1：タイムスタンプ単位（注記参照） ビット2...6：reserved (0) ビット7：– 1 – ジオフェンシングゾーン外	
22	1	uint8_t	Reserved (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の座標 ビット13...15：reserved (0)	
25	2	Reserved (0)		
27	M=N-22		オプションのデータフィールド – リストを参照	
27+M	2	Reserved (0)		

注：V6.290より前のDashboardおよびSuper-Modemでは、タイムスタンプの単位は1/64 secで、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。バージョンV6.290以降では、タイムスタンプの単位はミリ秒で、タイムスタンプ単位インジケーターは1です。

Marvelmind robotics

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、次の構造を含められます：

- 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

Offset	バイト数	型	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	

9.1.1. リアルタイムタイムスタンプ付きのHedgehog位置情報のパケット（ファームウェアv7.200+）

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, etc.): 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

Offset	サイズ	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットの種類	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 – 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	ビーコンのX座標、mm	
17	4	int32_t	ビーコンのY座標、mm	
21	4	int32_t	ビーコンのZ座標、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でください。 ビット1：タイムスタンプ単位（注記参照） ビット2...6：reserved (0) ビット7：– 1 – ジオフェンシングゾーン外	
26	1	uint8_t	Reserved (0)	
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...15：reserved (0)	
29	2	Reserved (0)		
31	M=N-26		オプションのデータフィールド – リストを参照	
31+M	2	Reserved (0)		

Marvelmind robotics

9.2. 固定ビーコン位置情報を含むパケット

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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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個の各Beaconのデータ構造フォーマット：

Offset	バイト数	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	Reserved (0)

Marvelmind robotics

9.3. 生IMUデータのパケット

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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットの種類	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	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	Reserved (0)	
29	4	uint32_t	タイムスタンプ、ms	
33	8	8バイト	reserved	

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。

Marvelmind robotics

9.3.1. リアルタイムタイムスタンプ付きの生IMUデータパケット (firmware v7.200+)

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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットの種類	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	Reserved (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。ModemおよびDashboardを持つすべてのデバイスによって同期された時刻。	
37	8	8バイト	reserved	

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。

Marvelmind robotics

9.4. 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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	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	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

Offset	バイト数	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 – ビーコンの超音波送信の内部時刻、直近のウェイクアップイベントからの経過時間（ミリ秒）(V5.89+)。	
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	reserved	

距離項目のフォーマット

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス（未入力時は0）	
1	4	uint32_t	ビーコンまでの距離、mm	
5	1	uint8_t	Reserved (0)	

Marvelmind robotics

9.4.1. リアルタイムタイムスタンプ付きの生距離データパケット (ファームウェアv7.200+)

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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	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	0x0084
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

Offset	バイト数	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	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を持つすべてのデバイスによって同期された時刻。	
33	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+）	
35	1	uint8_t	reserved	

距離項目のフォーマット

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス（未入力時は0）	
1	4	uint32_t	ビーコンまでの距離、mm	
5	1	uint8_t	Reserved (0)	

Marvelmind robotics

9.5. 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, Helmet, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットの種類	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	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	X方向速度（フュージョン）、mm/s	
27	2	int16_t	Y方向速度（フュージョン）、mm/s	
29	2	int16_t	Z方向速度（フュージョン）、mm/s	
31	2	int16_t	ビーコンのX方向加速度、mm/s2	
33	2	int16_t	ビーコンのY方向加速度、mm/s2	
35	2	int16_t	ビーコンのZ方向加速度、mm/s2	
37	1	uint8_t	ビーコンのアドレス	
38	1	1バイト	Reserved (0)	
39	4	uint32_t	タイムスタンプ、ms	
43	4	4バイト	Reserved (0)	

注: クォータニオンは10000の値に正規化されています。

Marvelmind robotics

9.5.1. リアルタイムタイムスタンプ付きIMU融合データパケット (firmware v7.200+)

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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットの種類	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	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	X方向速度（フュージョン）、mm/s	
27	2	int16_t	Y方向速度（フュージョン）、mm/s	
29	2	int16_t	Z方向速度（フュージョン）、mm/s	
31	2	int16_t	ビーコンのX方向加速度、mm/s2	
33	2	int16_t	ビーコンのY方向加速度、mm/s2	
35	2	int16_t	ビーコンのZ方向加速度、mm/s2	
37	1	uint8_t	ビーコンのアドレス	
38	1	1バイト	Reserved (0)	
39	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。モデムとDashboardによりすべてのデバイスで同期された時刻	
47	4	4バイト	Reserved (0)	

注: クォータニオンは10000の値に正規化されています。

Marvelmind robotics

9.6. テレメトリ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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
1	1	uint8_t	パケットの種類	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	2	uint16_t	バッテリー電圧、mV	
7	1	int8_t	RSSI、dBm	
8	13		Reserved (0)	

Marvelmind robotics

9.7. 品質情報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, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	アドレス	
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	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警報なし 1...255 - ジオフェンスゾーン番号 MMSW0005ライセンスが必要。	
8	13		Reserved (0)	

Marvelmind robotics

9.8. 全ビーコンのテレメトリを含むパケット

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(SSMファームウェア内)
Mini-RX (Badge, Helmet, etc.): 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

Offset	バイト数	型	説明	値
0	1	uint8_t	アドレス	0xff
1	1	uint8_t	パケットの種類	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個のビーコンのテレメトリ（下表参照）	

ビーコンテレメトリ項目のフォーマット

Offset	バイト数	型	説明	値
0	1	uint8_t	ビーコンのアドレス	
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	Reserved (0)	

Marvelmind robotics

9.9. NMEA0183プロトコル

対応ハードウェア/ソフトウェア：

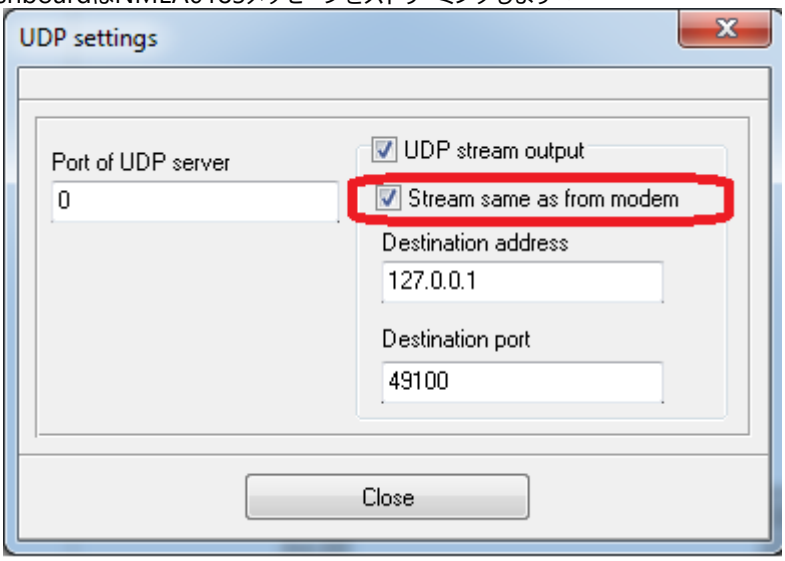
Super-Beacon: 非対応
Industrial Super-Beacon: 非対応
Modem HW5.1: 非対応
Super-Modem: 対応
Mini-RX (Badge, Helmet, etc.): 非対応
Mini-TX: 非対応
Mini-TX-2: 非対応
Modem HW4.9: 非対応
Beacon HW4.9: 非対応
Beacon HW4.5: 非対応
Dashboardソフトウェア： 対応

パケットのフォーマット

Offset	バイト数	型	説明	値
0	N	Nバイト	NMEA0183メッセージ(こちらの説明を参照)	
N	1	uint8_t	モバイルビーコンのアドレス	

備考:

- DashboardのSuper-Modem設定のinterfacesセクションでNMEA0183プロトコルを選択すると、Super-ModemはUDPでNMEA0183メッセージを送信します
- モデムの設定のinterfacesセクションでNMEA0183プロトコルが選択され、Dashboardの Settings/UDP Settingsメニューで下のスクリーンショットの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：



Marvelmind robotics

10. CAN経由の通信プロトコル

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:	要望に応じて対応
Mini-RX (Badge, Helmet, etc.):	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、要望に応じてSuper-ModemおよびIndustrial Super-Beaconに搭載可能です。CANを搭載した場合、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**。

10.1 ストリーミングの「Marvelmind」プロトコル

UARTストリーミングの該当章で説明しているパケットは、CANフレームid **0x10**でCAN経由でも送信されます。各CANフレームには1～8バイトのデータを格納できます。データのバイト数はCANフレームの**DLC**フィールドで指定されます。

データは生のストリームとして送信されるため、CANフレームに、ある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。複数のCANフレームを受信し、そ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じ方法で処理してください。

10.2 NMEA0183通信プロトコル

UARTストリーミングの該当章で説明しているパケットは、CANフレームid **0x11**でCAN経由でも送信されます。各CANフレームには1～8バイトのデータを格納できます。データのバイト数はCANフレームの**DLC**フィールドで指定されます。

データは生のストリームとして送信されるため、CANフレームに、ある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。複数のCANフレームを受信し、そ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じ方法で処理してください。

11. Dashboard csvログファイルのフォーマット

Dashboardは、固定ビーコンとモバイルビーコンの位置などのデータを、Dashboardのディレクトリ内の「log」フォルダにあるcsvログファイルに保存します。バージョンV7.000からログのフォーマットが変更されました。以前のフォーマットはモデムHW v4.9用にのみ残されています。

11.1 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、各イベントが一行のCSVとしてログに記録され、イベントごとに行のフォーマットが異なります。ただし、行の先頭部分はすべての種類の行で共通です。

以下はcsvログファイルの数行の例です：

```
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
```

行の共通部分は最初の3つのフィールドで構成されます：

「T2021_11_04__173001_581」– この行のデータのタイムスタンプ：2021.11.04, 17:30:01.581；
「user」– ユーザー名（将来用のreserved。将来のバージョンでは、Dashboardはユーザーのログインに対応する予定です）；
「41」– 行タイプのID。異なる行タイプは、以降のフィールドで異なるフォーマットを持ちます。

データフィールドには共通の特殊コードがあります：

「**nl**」– no license（ライセンスなし）。このフィールドを埋めるには何らかのライセンスが必要です；
「na」– not applicable（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。たとえば、モバイルビーコンの位置特定に成功しなかった場合、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、さまざまな行タイプの説明を記載します。

Marvelmind robotics

11.1.1 行タイプID 01 – マップファイルへのリンク

この行は、マップファイルの自動保存時、またはユーザーが「Save map」ボタンで保存した時に記録されます。

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 - 行タイプ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

11.1.2 行タイプ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Modemプロトコル設定が「Marvelmind」の場合に記録されます。

Marvelmindプロトコルには様々な種類のレコードがあり、それらはログファイルの異なる行に対応しています。以降のサブ章で説明します。

11.1.2.1 Hedgehog位置 (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 - 行タイプID（Marvelmindプロトコルストリーミング）
3	17 (0x0011) – Hedgehog位置のデータコード 129 (0x0081) – hedgehog位置のデータコード（リアルタイムタイムスタンプ付き）
4	Hedgehogアドレス
5	Hedgehog X座標、メートル
6	Hedgehog Y座標、メートル
7	Hedgehog 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でください。 ビット1...6：reserved ビット7：– 1 – ジオフェンシングゾーン外
9	ヨー角とフラグ: ビット0...11：hedgehogペアのヨー角、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の座標
10	時間シフト、ms。超音波発信からこの行内の位置計算までに経過した時間

Marvelmind robotics

11.1.2.2 固定ビーコン位置 (41 18)

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	18 (0x0012) – 固定ビーコン位置のデータコード
4	固定ビーコンアドレス
5	ビーコンX座標（メートル）
6	ビーコンY座標（メートル）
7	ビーコンZ座標（メートル）
8	Reservedフィールド

11.1.2.3 Hedgehogから固定ビーコンまでの生の距離 (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) – 生の距離のデータコード 132 (0x0084) – 生距離のデータコード（リアルタイムタイムスタンプ付き）
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）、下記参照
6+N*2+1	時間シフト（ms）。超音波放射から距離測定までに経過した時間。

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

Marvelmind robotics

11.1.2.4 生IMUデータ (41 3) または (41 131)

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IMUデータのデータコード 131 (0x0083) – 生IMUデータのデータコード（Hedgehogでリアルタイムタイムスタンプ有効時）
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

11.1.2.5 IMU融合データ (41 5) または (41 133)

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融合データのデータコード 133 (0x0085) – IMU融合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（融合）、メートル
6	ビーコンのY座標（融合）、メートル
7	ビーコンのZ座標（融合）、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	X方向速度（フュージョン）、mm/s
13	Y方向速度（フュージョン）、mm/s
14	Z方向速度（フュージョン）、mm/s
15	ビーコンのX方向加速度、mm/s2
16	ビーコンのY方向加速度、mm/s2
17	ビーコンのZ方向加速度、mm/s2

Marvelmind robotics

11.1.2.6 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

11.1.2.7 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 - 行タイプ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）

11.1.3 行タイプ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 - 行タイプ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形式によるフィールドの順序（NMEA0183レコードもカンマ区切り値形式を持つ）

11.1.4 行タイプ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の設定のinterfacesセクションでペイロードデータサイズがゼロ以外に設定され、
ユーザーデバイスがhedgehogのUSBまたはUART経由で何らかのデータを送信した場合に記録されます。

また、一部のMarvelmindデバイス（ロボットv100やBoxieなど）でもペイロードデータを利用できます。

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	43 - 行タイプID（ユーザーペイロード）
3	Hedgehogのアドレス
4、5など	カンマ区切りのペイロードデータバイトのシーケンス（各フィールドは1バイト）

11.1.4.1 Robot v100のペイロードテレメトリデータ

行の一般的な形式は、ユーザーペイロードデータ形式に対応します。

データバイト（行の四番目のフィールド以降）は、以下に示す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、下位バイトから配置されます（リトルエンディアン形式）。

Robot v100テレメトリレコードN3：

Offset	バイト数	型	説明	値
0	2	uint16_t	レコードID	0x3003
2	2*12		12個のlidarの距離（各2バイト） 各lidarデータは以下の形式を持ちます： 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 Status = 0 – 距離が測定されています Status <> 0 – 距離が測定されていません	
26	1	uint8_t	一般的なlidarステータス： ビット0：1 – lidarの読み取りに成功 0 – lidarの読み取り失敗 ビット1...7 – reserved (0)	
27	1	uint8_t	ロボット状態: 0: ロボットは正常に停止しています 1: ロボットは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 - ロボットバッテリー電圧。 V= (RV/10) + 20 Volts	
29	2	int16_t	ロボット供給電流、x10 mA	N

Marvelmind robotics

			値が負の場合、ロボットバッテリーはこの電流で充電中です。	
31	1	uint8_t	残りのバッテリー容量、%	
32	2	uint16_t	ロボット供給電力、Watt	
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	1	uint8_t	左モーターの出力、%	
38	1	uint8_t	右モーターの出力、%	
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 – reserved ビット4：1 – 超音波トラッキングエラー ビット5...7 - reserved	
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：

Offset	バイト数	型	説明	値
0	2	uint16_t	レコードID	0x3004
2	2*12		12個のlidarの距離（各2バイト） 各lidarデータは以下の形式を持ちます： 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 Status = 0 – 距離が測定されています Status <> 0 – 距離が測定されていません	
26	1	uint8_t	一般的なlidarステータス： ビット0：1 – lidarの読み取りに成功 0 – lidarの読み取り失敗 ビット1...7 – reserved (0)	
27	1	uint8_t	ロボット状態: 0: ロボットは正常に停止しています 1: ロボットは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	移動プログラム内の現在の項目のインデックス	
29	1	uint8_t	移動プログラム内の項目の総数	
30	1	uint8_t	Reserved	
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	アラームの原因となったlidarの現在の測定距離、x2cm	
34	3		Reserved	
36	2	int16_t	ロボットX位置、cm	
38	2	int16_t	ロボットY位置、cm	

11.1.4.2 Robot Boxieのペイロードテレメトリデータ

行の一般的な形式は、ユーザーペイロードデータ形式に対応します。

データバイト（行の四番目のフィールド以降）は、以下に示す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、下位バイトから配置されます（リトルエンディアン形式）。

Robot Boxieテレメトリレコード N1：

Offset	バイト数	型	説明	値
0	2	uint16_t	レコードID	0x3101
2	2*12		12個のlidarの距離（各2バイト） 各lidarデータは以下の形式を持ちます： 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 Status = 0 – 距離が測定されています Status <> 0 – 距離が測定されていません	
26	1	uint8_t	一般的なlidarステータス： ビット0：1 – lidarの読み取りに成功 0 – lidarの読み取り失敗 ビット1...7 – reserved (0)	
27	1	uint8_t	ロボット状態: 0: ロボットは正常に停止しています 1: ロボットは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を意味します	
29	2	int16_t	ロボット供給電流、x10 mA 例えば、値123は1.230 Aを意味します	
31	1	uint8_t	Reserved	
32	1	uint8_t	左モーターの出力、%	
33	1	uint8_t	右モーターの出力、%	
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 – reserved ビット2：1 – 移動プログラム実行中 ビット3：1 – 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 - reserved	
44	2	int16_t	ロボットのY座標、cm	

Marvelmind robotics

Robot Boxieテレメトリレコード N3：

Offset	バイト数	型	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フィルタを使用して計算されたロボットのX座標、cm	
10	2	int16_t	EKFフィルタを使用して計算されたロボットのY座標、cm	
12	2	int16_t	Reserved	
13	1	uint8_t	一般的なlidarステータス： ビット0：1 – lidarの読み取りに成功 0 – lidarの読み取り失敗 ビット1...7 – reserved (0)	
14	1	uint8_t	ロボット状態: 0: ロボットは正常に停止しています 1: ロボットは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	
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が0、二番目が1など）	
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 –「Run forever」オプション ビット1...7 - reserved	
18	1	uint8_t	0 – lidarによるアラームなし 1...12 - アラームを出したlidarの番号	
19	1	uint8_t	lidarによる最小アラーム距離、x2cm	
20	1	uint8_t	アラームの原因となったlidarの現在の測定距離、x2cm	
21	2	int16_t	ロボット上のペアリングされた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	ロボットの目標方向と現在の向きの偏差角度（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バイト	Reserved	

11.1.5 行タイプID 44 – Dashboardのリアルタイムプレイヤーの位置

この行にはMMSW0005ライセンスが必要です。

このラインは、リアルタイムプレーヤーが有効になっている場合にHedgehog用に記録されます。
リアルタイムプレーヤーは100 Hzの位置データを提供します。

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	44 - ラインタイプID(リアルタイムプレーヤー位置)
3	Hedgehogのアドレス
4	Reservedフィールド
5	Hedgehog X座標、メートル
6	Hedgehog Y座標、メートル
7	Hedgehog Z座標、メートル

Marvelmind robotics

11.1.6 ラインタイプID 55 – 短縮ビーコンテレメトリ

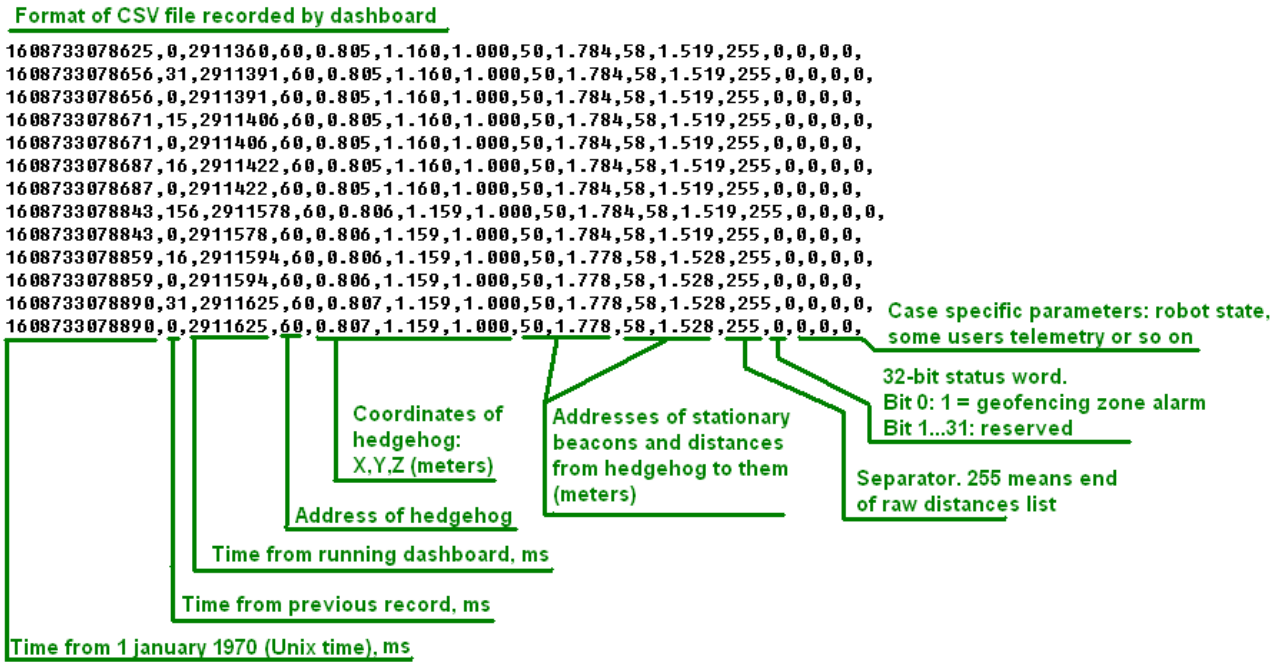
このラインには、SSM(Super Super-Modem)から受信したビーコンテレメトリが含まれます。

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	55- ラインタイプID(短縮テレメトリ)
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の補数コードでのRSSI:R0 R0が128未満の場合、RSSI= R0 dBm (R0 >= 128) の場合、RSSI= R0-256 dBm
7	Reserved
8	Reserved

11.2 以前のcsvログのフォーマット（V7.000より前のDashboardまたはモデムHW v4.9）

以下は、ログファイルの旧形式を示す図です:



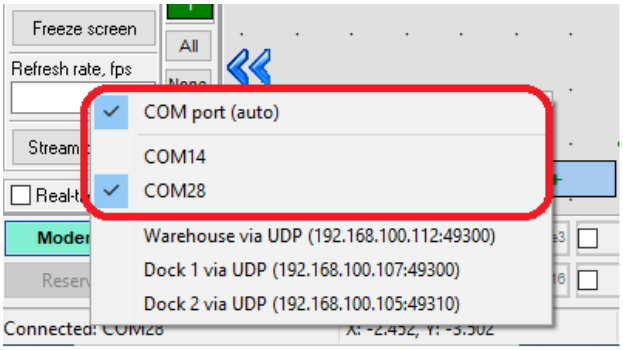
Marvelmind robotics

12. Dashboardとモデム間のインターフェース

12.1 USB（仮想COMポート）インターフェース

Marvelmind Dashboardとモデムおよびビーコンの標準の接続はUSBインターフェースです。Dashboardはシステムの COMポートを自動的にスキャンし、接続されたモデムまたはビーコンを見つけます。

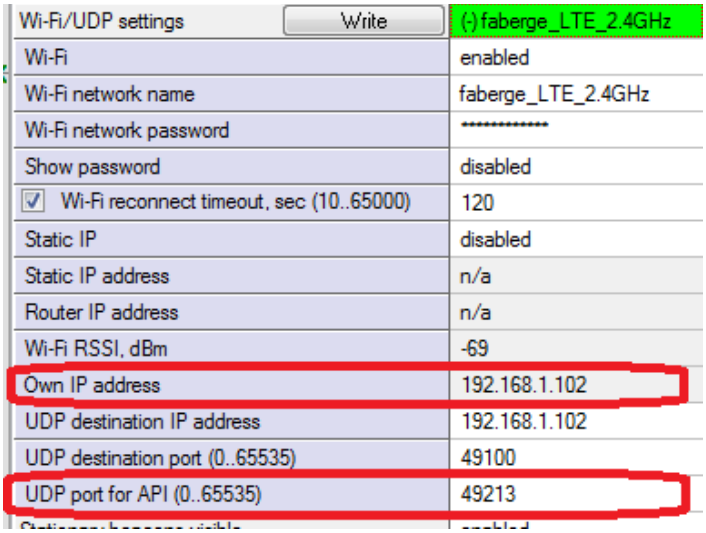
特定のCOMポートを使用する必要がある場合は、Dashboardの左下にあるCOMポートのコンテキストメニューで選択できます。ポートの自動スキャンモードに戻すには、メニューで「COM port (auto)」を選択します：



12.2 UDPによるSuper-Modemへの接続

Super-ModemはWi-Fiインターフェースを備えており、DashboardはUSBの代わりにUDPで通信できます。

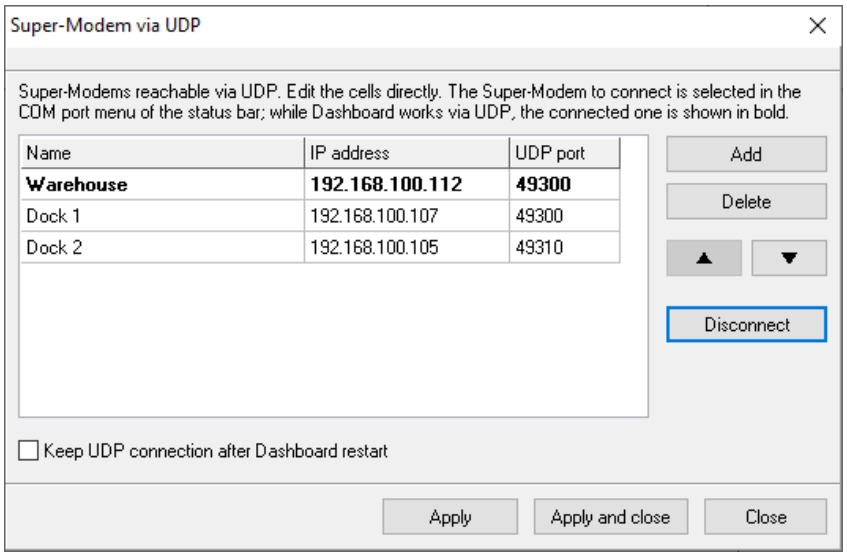
Super-ModemのIPアドレスはSuper-Modemの設定に表示されます。通信用のUDPポートはユーザーが指定できます：



Wi-Fi/UDP settings	Write	(-) faberge_LTE_2.4GHz
Wi-Fi		enabled
Wi-Fi network name		faberge_LTE_2.4GHz
Wi-Fi network password		**********
Show password		disabled
Wi-Fi reconnect timeout, sec (10..65000)		120
Static IP		disabled
Static IP address		n/a
Router IP address		n/a
Wi-Fi RSSI, dBm		-69
Own IP address		192.168.1.102
UDP destination IP address		192.168.1.102
UDP destination port (0..65535)		49100
UDP port for API (0..65535)		49213

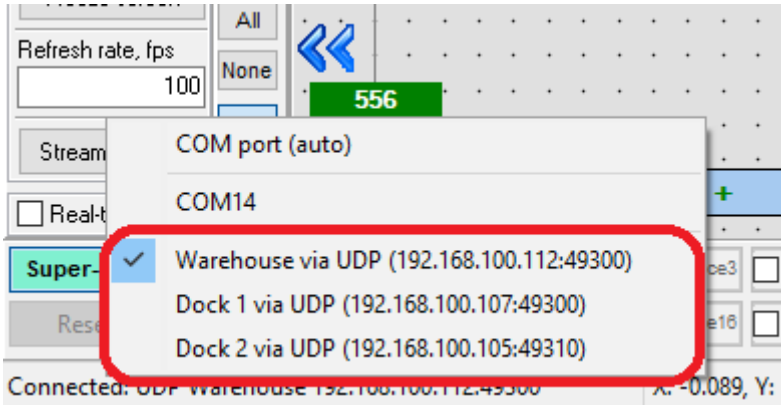
UDPでSuper-Modemと通信するには、Super-Modem用のMMSW0014ライセンスが必要です。

Dashboardでは、メニュー「Settings/Super-Modem via UDP」で送信先のIPアドレスとUDPポートを指定できます。また、各モデムの設置場所がわかるように、Super-Modemの名前も指定できます：



Super-Modemのリストは、Dashboardの左下にある通信ポートのコンテキストメニューに表示され、そこで接続するSuper-Modemを選択できます：

Marvelmind robotics


All
Refresh rate, fps
100
None
556
Stream
COM port (auto)
COM14
Warehouse via UDP (192.168.100.112:49300)
Dock 1 via UDP (192.168.100.107:49300)
Dock 2 via UDP (192.168.100.105:49310)
Super-
Connected: UDP Warehouse
X: -0.089, Y:

13. 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にインターフェースを提供します。APIは、MS Windows用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)用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)経由でmodemに接続し、modem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言語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関数の呼び出しを含んでいます。

このサンプルは、ユーザーのソフトウェア開発の基礎として、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に移植するためのベースとして使用できます。

テスト環境:

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+

13.1 Windows用インストール

- Marvelmind APIソフトウェアパッケージをダウンロードしてください。Dashboard APIおよびサンプルソフトウェアを、プログラムで使用するディレクトリにコピーしてください。Windows版のサンプルには事前にビルドされた実行ファイルが含まれており、APIソフトウェアパッケージ内の 'windows' ディレクトリから 'mm_api_example.exe' をすぐに実行できます。

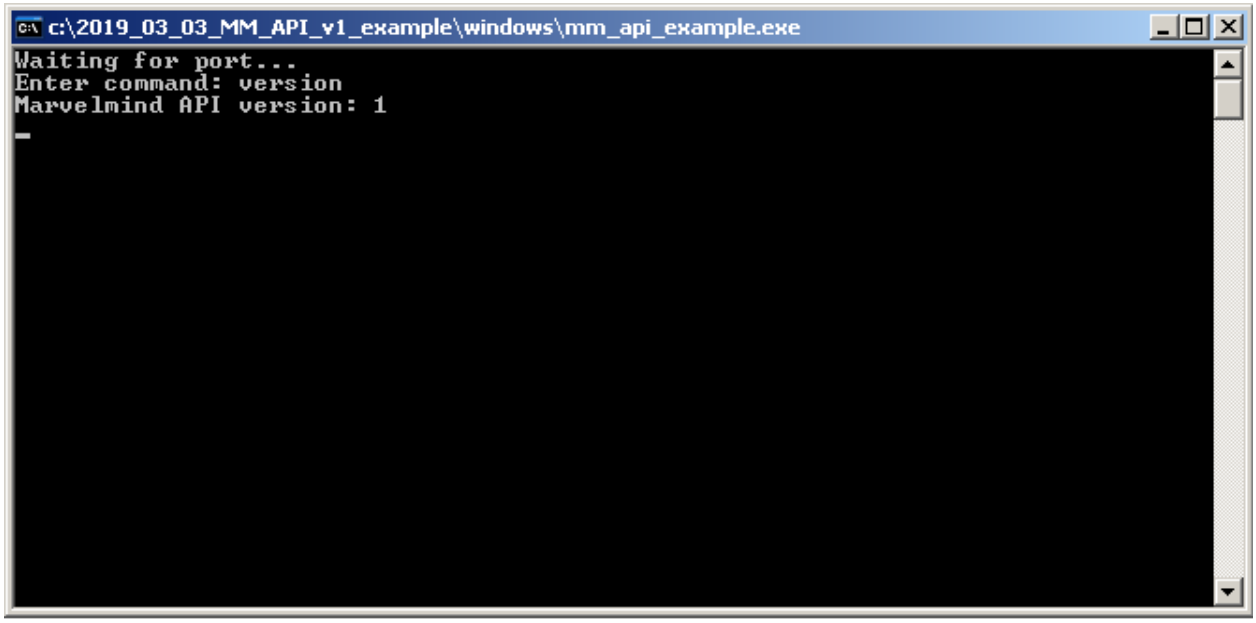
13.2 Linux用インストール

- 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、プログラムで使用するディレクトリにコピーします。Linux版は、**x86**（IntelまたはAMDのCPUを搭載したほとんどのノートパソコン）と**arm**（Raspberry PIなどのシングルボードコンピューター）の二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ます
- 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 **/usr/local/lib**ディレクトリにコピーしてください。**libdashapi.so**があるディレクトリで開いたターミナルで**sudo cp libdashapi.so /usr/local/lib**コマンドを実行します。その後、ターミナルで**sudo ldconfig**を実行してください。
- ユーザーを**dialout**グループに追加して、シリアルポートへのアクセス権限を付与する必要がある場合があります:
  - ターミナルで実行: **sudo adduser $USER dialout**
  - **/etc/udev/rules.d**ディレクトリに、以下の内容の「**99-tty.rules**」ファイルを追加します:

    ```
    #Marvelmind serial port rules
    KERNEL==”ttyACM0”,GROUP=”dialout”,MODE=”666”
    ```
- サンプルソフトウェアをビルドします – パッケージに含まれる「source」ディレクトリで開いたターミナルで「make all」を実行します
- ターミナルで「./mm_api_example」と入力してサンプルを実行します

13.3 APIへの接続を確認します

サンプルソフトウェアの実行後、ターミナルで「space」ボタンを押し、コマンド「version」を入力して Enterを押します。APIのバージョンが表示されれば、APIライブラリと通信できています。



13.4 Marvelmind APIライブラリの説明

APIは、MS Windows用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）用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。本ドキュメントのこのセクションでは、これらすべての関数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、ほとんどの複雑なデータ構造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含まれています。サンプルソフトウェアの「marvelmind_api.c」ファイルでは、メモリプールとC構造体のフィールド間のデータ移動の実装を確認できます。

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

型	バイト数	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の補数形式の符号付き整数値、-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（メモリ内のアドレス）。 32ビット環境では4バイト、64ビット環境では8バイト。

各関数の説明には、その関数が利用可能なAPIバージョンのセットが含まれています。新しいAPIバージョンでは、Marvelmindシステムの新機能に対応するより多くの関数がサポートされます。現在、Dashboardにあるすべてのビジネスが API経由で利用できるわけではないため、より多くの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と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	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	V1+	なし
最新の位置データを取得	V1+	なし
最新の位置データを取得（角度付き）	V3+	なし
ビーコンの位置を設定	V3+	MMSW0005
ビーコン間の距離を設定	V4+	MMSW0005
最新の生距離データを取得	V1+	なし
hedgehogの高さを取得	V4+	なし
hedgehogの高さを設定	V4+	MMSW0005
サブマップ内の固定ビーコンの高さを取得	V4+	なし
サブマップ内の固定ビーコンの高さを設定	V4+	MMSW0005
位置更新レート設定を取得	V1+	なし
位置更新レート設定を設定	V1+	MMSW0005
サブマップを追加	V1+	MMSW0005
サブマップを削除	V1+	MMSW0005
サブマップをフリーズ	V1+	MMSW0005
サブマップのフリーズを解除	V1+	MMSW0005
サブマップ設定を取得	V1+	なし
サブマップ設定を設定	V1+	MMSW0005
マップをフリーズ	V4+	MMSW0005
マップのフリーズを解除	V4+	MMSW0005
ビーコンの超音波設定を取得	V1+	なし
ビーコンの超音波設定を設定	V1+	MMSW0005
マップを消去	V1+	MMSW0005
デバイスを初期設定にリセット	V1+	MMSW0005
ビーコンを軸に接続	V2+	MMSW0005
モデムの設定メモリダンプを読み取る	V3+	MMSW0005
モデムの設定メモリダンプを書き込む	V3+	MMSW0005
モデムから空気の温度設定を取得	V3+	なし
モデムに空気の温度設定を設定	V3+	なし
デバイスのソフトウェアリセット	V3+	なし
ビーコンのリアルタイムプレーヤー設定を取得	V6+	なし
ビーコンのリアルタイムプレーヤー設定を行う	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行う	V6+	MMSW0005
位置更新モードを取得	V6+	なし
位置更新モードを設定	V6+	MMSW0022
トリガー位置更新のコマンド	V6+	MMSW0022
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を送信	V9+	MMSW0005
汎用ユーザーペイロードデータを取得	V9+	MMSW0005
手動距離測定用コマンドを送信	V9+	MMSW0011
modemからストリーミングデータを取得	V9+	なし
デバイスタイプがmodemかどうかを確認	V1+	なし
デバイスタイプが固定ビーコンかどうかを確認	V1+	なし
デバイスタイプがhedgehogであるかを確認する	V1+	なし

13.4.1.1 Marvelmind APIライブラリのバージョンを取得

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関数が利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言語での宣言： bool mm_api_version(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た false – 実行エラー

パラメータ:

型	説明
void *	格納先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。

型	説明
uint32_t	APIライブラリのバージョン

13.4.1.2 最後のエラーを取得

APIライブラリでの最終操作のステータスを読み取り、エラーの原因を区別します。

関数名： **mm_get_last_error**
C言語での宣言： 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 – 実行エラー

パラメータ:

型	説明
void *	格納先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。

型	説明
uint32_t	最後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

Marvelmind robotics

13.4.1.3 シリアルポートを開く

Marvelmindデバイス（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。APIがすべてのシリアルポートを検索し、Marvelmindデバイスに対応す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言語での宣言： 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

Marvelmind robotics

13.4.1.4 指定された名前で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。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言語での宣言： bool mm_open_port_by_name(void *pdata);
対応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 - ゼロで終端されるASCII文字列 (ASCIIZ)

Marvelmind robotics

13.4.1.4.1. UDPポートを開く

USBの代わりにUDP経由でSuper-Modemとの通信を確立できます。

関数名： **mm_open_port_udp**
C言語での宣言： bool mm_open_port_udp(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、UDPポートが開かれた false – 実行エラー

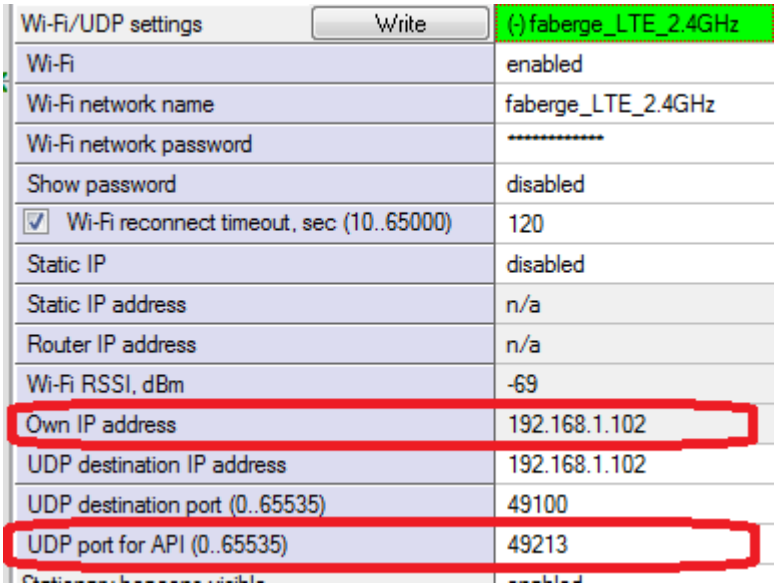
パラメータ:

型	説明
void *	UDP設定の構造体へのポインタ (下記参照)

ポインタが指すデータの構造:

型	説明
uint16_t	接続するUDPポート
uint16_t	通信のタイムアウト（ms）
uint16_t	reserved
最大255バイト	IPアドレス — ゼロで終端されたASCII文字列（ASCIIZ）

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



Marvelmind robotics

13.4.1.5 シリアルポートを閉じる

以前にmm_open_port関数で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 – 関数が正常に実行され、ポートを閉じた false – 実行エラー

パラメータ: なし

Marvelmind robotics

13.4.1.6 Marvelmindデバイスのバージョンと CPU IDを取得

バージョンと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はデバイス個体固有のIDです。

関数名： **mm_get_device_version_and_id**
C言語での宣言： bool mm_get_device_version_and_id (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、バージョンとCPU IDを取得 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 *	格納先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	ファームウェアのメジャーバージョン（例：V6.07aの場合「6」）
uint8_t	ファームウェアのマイナーバージョン（例：V6.07aの場合「7」）
uint8_t	ファームウェアのセカンドマイナーバージョン（例：V6.07aの場合「1」）
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やデバイスのステッカーに表示される形式のCPU IDになります。

13.4.1.7 デバイスのリストを取得

Modemに認識されているMarvelmindデバイスのリストを読み取ります。このリストには、スリープ中のデバイスを含め、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言語での宣言： bool mm_get_devices_list (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、デバイスのリストが取得されました false – 実行エラー

パラメータ:

型	説明
void *	格納先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数 (N)
N*9 バイト	次の表に記載されているN個のデバイス構造のシーケンス

リスト内の各デバイスの構造：

型	説明
uint8_t	デバイスのアドレス
bool	true = アドレス重複 - 同じアドレスのデバイスが1台より多く見つかった false = 重複アドレスではない
bool	true = デバイスはスリープ状態 false = デバイスはスリープ状態ではない
uint8_t	ファームウェアのメジャーバージョン（例：V6.07aの場合「6」）
uint8_t	ファームウェアのマイナーバージョン（例：V6.07aの場合「7」）
uint8_t	ファームウェアのセカンドマイナーバージョン（例：V6.07aの場合「1」）
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）。
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 0 – デバイスからの確認待ち（Dashboardの「Connecting...」と同様）。 ビット1...7 - TBD

Marvelmind robotics

13.4.1.8 デバイスをウェイクアップ

指定したデバイスをウェイクするコマンドを送信します。ウェイクコマンドが送信され、該当するデバイスが存在する場合、そのデバイスは数秒以内にModemに接続され、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ました false – 実行エラー

パラメータ:

型	説明
uint8_t	1...254 - ウェイクするMarvelmindデバイスのアドレス 0 – すべてのデバイスをウェイク

13.4.1.9 デバイスをスリープ状態にする

既存のデバイスをスリープ状態にします。

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、スリープコマンドが送信されました 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送信

13.4.1.10 ビーコンからテレメトリデータを取得

Marvelmindビーコンのテレメトリデータを読み取ります。

関数名： **mm_get_beacon_telemetry**
C言語での宣言： 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 *	格納先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	ビーコンの稼働時間（秒）（リセットまたはウェイクアップからの時間）。
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	Reserved (0)

13.4.1.11 最新の位置データを取得

モデムから最新の座標パックと、あればユーザーペイロードデータを読み取ります。

関数名： **mm_get_last_locations**
C言語での宣言： bool mm_get_last_locations(void *pdata);
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、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 *	格納先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 バイト	最後に更新された座標の18バイトデータ構造×6個、下表参照
bool	true – 新しい生距離データが読み取り可能です
5バイト	TBD
uint8_t	ユーザーペイロードデータサイズ (M)
M bytes	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

13.4.1.12 最新の位置データを取得（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言語での宣言： 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 *	格納先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
20*6バイト bool	最後に更新された座標の20バイトデータ構造体×6個、下表を参照 true – 新しい生距離データが読み取り可能です
5バイト	TBD
uint8_t	ユーザーペイロードデータサイズ (M)
M bytes	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 = 角度利用不可 ビット13...15 - reserved

Marvelmind robotics

13.4.1.13 ビーコンの位置を設定

指定されたビーコンの位置を手動で設定します。

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、位置が更新されました 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
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

13.4.1.14 ビーコン間の距離を設定

ビーコン間距離の手動設定。

関数名： **mm_set_beacons_distance**
C言語での宣言： 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、距離が書き込まれました false – 実行エラー

パラメータ:

型	説明
void *	距離データの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
uint8_t	一番目のビーコンのアドレス
uint8_t	二番目のビーコンのアドレス
int32_t	ビーコン間の距離 (mm)

13.4.1.15 最新の生距離データを取得

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言語での宣言： 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、生の距離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 *	格納先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	生の距離データ項目の数 (N)。 リクエストあたりの生距離の最大数は16です: N<=16
9*N バイト	最新の生距離のN個の9バイトデータ構造。以下の表を参照

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入力
uint8_t	RXヘッドインデックス (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入力
uint8_t	TXヘッドインデックス (TBD)
uint32_t	TXデバイスからRXデバイスまでの距離、mm
uint8_t	TBD

13.4.1.16 hedgehogの高さを取得

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言語での宣言： bool mm_get_hedge_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、高さが返される false – 実行エラー

パラメータ:

型	説明
uint8_t	Hedgehogのアドレス
void *	高さデータを格納するバッファへのポインタ

ポインタによるデータ構造:

型	説明
int32_t	Hedgehogの高さ、mm

Marvelmind robotics

13.4.1.17 hedgehogの高さを設定

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言語での宣言： bool mm_set_hedge_height (uint8_t address, 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、高さが変更されました false – 実行エラー

パラメータ:

型	説明
uint8_t	Hedgehogのアドレス
void *	高さデータを格納する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	Hedgehogの高さ、mm

13.4.1.18 サブマップ内の固定ビーコンの高さを取得

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言語での宣言： 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、高さが返される 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 *	高さデータを格納するバッファへのポインタ

ポインタによるデータ構造:

型	説明
uint8_t	サブマップID、関数呼び出し前に入力する必要があります
int32_t	ビーコンの高さ (mm)

13.4.1.19 サブマップ内の静止ビーコンの高さを設定

サブマップ内の静止ビーコン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、高さが変更されました 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
void *	高さデータを格納する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
uint8_t	サブマップID
int32_t	ビーコンの高さ (mm)

13.4.1.20 位置更新レート設定を取得

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言語での宣言： 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ました false – 実行エラー

パラメータ:

型	説明
void *	格納先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値（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。

13.4.1.21 位置更新レート設定を設定

位置更新レート設定をmodemに書き込みます。

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

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、更新レートが変更されました false – 実行エラー

パラメータ:

型	説明
void *	データへのポインタ

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
uint32_t	位置更新レートの設定値（mHz単位）。1 Hzでは1000、16 Hzでは16000、0.05 Hzモードでは50が返されます。システムは、指定された更新レートを次の値のいずれかに丸めて使用します：0.05 Hz、0.1 Hz、0.2 Hz、0.5Hz、1 Hz、2 Hz、4 Hz、8 Hz、12 Hz、16 Hz、16+Hz。

13.4.1.22 サブマップを追加

新しいサブマップを追加します。

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、サブマップが追加されました false – 実行エラー

パラメータ:

型	説明
uint8_t	追加するサブマップID（0...254）

Marvelmind robotics

13.4.1.23 サブマップを削除

既存のサブマップを削除します。

関数名： **mm_delete_submap**
C言語での宣言： bool mm_delete_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、サブマップが削除されました false – 実行エラー

パラメータ:

型	説明
uint8_t	削除するサブマップID（0...254）

Marvelmind robotics

13.4.1.24 サブマップをフリーズ

サブマップをフリーズします。

関数名： **mm_freeze_submap**
C言語での宣言： bool mm_freeze_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、サブマップがフリーズされました 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)

Marvelmind robotics

13.4.1.25 サブマップのフリーズを解除

サブマップのフリーズを解除します。

関数名： **mm_unfreeze_submap**
C言語での宣言： 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ました false – 実行エラー

パラメータ:

型	説明
uint8_t	フリーズ解除するサブマップID (0...254)

13.4.1.26 サブマップ設定を取得

modemからサブマップ設定を読み取ります。

関数名： **mm_get_submap_settings**
C言語での宣言： 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ました 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 *	格納先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	開始ビーコンの三辺測量
uint8_t	開始ビーコンセット、ビーコン1
uint8_t	開始ビーコンセット、ビーコン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座標にのみ使用
bool	true = 手動制限距離 false = 自動制限距離
uint8_t	最大距離、メートル（手動制限距離用）
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モードでのHedgeの高さ
bool	true = サブマップは固定されている
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 = サブマップがミラーリングされています
4バイト	サブマップ内のビーコンのアドレスリスト（0 = なし）
8バイト	近接サブマップの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構造のリスト（下記参照）

Marvelmind robotics

サービスゾーンポリゴン頂点の構造：

型	説明
int16_t	X, cm
int16_t	Y, cm

Marvelmind robotics

13.4.1.27 サブマップ設定を設定

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);
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、サブマップ設定が変更されました 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）。

13.4.1.28 マップをフリーズ

サブマップをフリーズします。

関数名： **mm_freeze_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、マップをフリーズ false – 実行エラー

13.4.1.29 マップのフリーズを解除

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ました false – 実行エラー

Marvelmind robotics

13.4.1.30 ビーコンの超音波設定を取得

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言語での宣言： 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 *	格納先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専用ビーコンには該当しません)
uint8_t	TX周期の数(DSP RX専用ビーコンには該当しません)
bool	true = RXにAGCを使用 false = RXに手動ゲインを使用
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がフリーズモードで有効
bool	true = センサーRX2がフリーズモードで有効
bool	true = センサーRX3がフリーズモードで有効
bool	true = センサーRX4がフリーズモードで有効
bool	true = センサーRX5がフリーズモードで有効
uint8_t	DSP RXフィルターのインデックス（DSPビーコンにのみ該当） 0 = 19 kHz 1 = 25 kHz 2 = 31 kHz 3 = 37 kHz 4 = 45 kHz

13.4.1.31 ビーコンの超音波設定を設定

指定したビーコンに超音波設定を書き込みます。

関数名： **mm_set_ultrasound_settings**
C言語での宣言： 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、超音波設定が変更されました 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 *	書き込むデータへのポインタ(「get ultrasonic settings」関数を参照)。

Marvelmind robotics

13.4.1.32 マップを消去

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

関数名： **mm_erase_map**
C言語での宣言： bool mm_erase_map ();
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、マップが消去されました false – 実行エラー

パラメータ: なし

Marvelmind robotics

13.4.1.33 デバイスを初期設定にリセット

デバイスをデフォルト設定（無線、超音波など）にリセットします。

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、デバイスがデフォルト設定にリセットされました 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

Marvelmind robotics

13.4.1.34 ビーコンを軸に接続

選択したビーコンが軸上に配置されるようにマップをシフトします。

関数名： **mm_beacons_to_axes**
C言語での宣言： 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、マップがシフトされた false – 実行エラー

パラメータ:

型	説明
uint8_t	address_0 – 中心（X=0、Y=0）に配置されるビーコンのアドレス
uint8_t	address_x – X軸上（Y=0）に配置されるビーコンのアドレス
uint8_t	address_y – Yの正方向（Y>0）に配置されるビーコンのアドレス

13.4.1.35 Modemの設定メモリのダンプを読み取る

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を保存できます。

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:

型	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 – データ受信用のユーザーバッファへのポインタ

13.4.1.36 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の設定と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、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） 正しく動作させるため、オフセットは4096バイトページにアラインする必要があります（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 – データを格納したユーザーバッファへのポインタ

注意: 設定の書き込み後、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)**）を推奨します。

13.4.1.37 デバイスの再起動（ソフトリセット）

指定し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言語での宣言： 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、デバイスをリセット中 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経由で接続されているデバイスのソフトウェアリセット

Marvelmind robotics

13.4.1.38 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言語での宣言： bool mm_get_air_temperature (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、温度が返されます false – 実行エラー

pdataポインタ経由で返されるデータの構造:

型	説明
int8_t	空気の温度（摂氏）

Marvelmind robotics

13.4.1.39 空気温度設定をmodemに書き込む

modemに空気温度設定（摂氏）を設定します。

関数名： **mm_set_air_temperature**

C言語での宣言： bool mm_set_air_temperature (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、温度が書き込まれました false – 実行エラー

ユーザーがpdataポインタ経由で提供するデータの構造:

型	説明
int8_t	空気の温度（摂氏）

Marvelmind robotics

13.4.1.40 ビーコンのリアルタイムプレーヤー設定を取得

ビーコンのリアルタイムプレー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+

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 = リアルタイムプレイヤーが有効
uint8_t	処理するリアルタイムプレイヤーの前方サンプル数
uint8_t	処理するリアルタイムプレイヤーの後方サンプル数
uint8_t	Reserved (0)
uint8_t	Reserved (0)

Marvelmind robotics

13.4.1.41 ビーコンのリアルタイムプレーヤー設定を行う

ビーコンのリアルタイムプレイ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:

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）

Marvelmind robotics

13.4.1.42 ジオリファレンス設定を取得する

ジオリファレンス設定（Marvelmindマップの点（X=0, Y=0）の地理的位置）を読み取ります。

関数名： **mm_get_georeferencing_settings**

C言語での宣言： 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
int32_t	緯度、x10-7度
int32_t	経度、x10-7度

Marvelmind robotics

13.4.1.43 ジオリファレンス設定を行う

ジオリファレンス設定をセットアップします（Marvelmindマップの点（X=0, Y=0）の地理的位置）。

関数名： **mm_set_georeferencing_settings**

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

対応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（「Get georeferencing settings」関数を参照）

Marvelmind robotics

13.4.1.44 位置更新モードを取得

モバイルビーコンの現在の位置更新モードを読み取ります。

関数名： **mm_get_update_position_mode**

C言語での宣言： bool mm_get_update_position_mode (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
uint8_t	モバイルビーコンの位置更新モード： 0 – 自動位置更新（デフォルトモード） 1 – reserved 2 – トリガー位置更新（API、Dashboardのボタン、またはモデムHW v5.1のピンによるユーザーのリクエストで位置を更新）
7バイト	Reserved (0)

Marvelmind robotics

13.4.1.45 位置更新モードを設定

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+

必要なライセンス：MMSW0022

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「位置更新モードの取得」を参照）

Marvelmind robotics

13.4.1.46 トリガー位置更新のコマンド

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

対応APIバージョン：V6+

必要なライセンス：MMSW0022

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

ポインタによるデータ構造:

型	説明
8バイト	Reserved (0)

Marvelmind robotics

13.4.1.47. ビーコンのジオフェンシングアラーム状態を設定する

ビーコンのアラームピンにアラーム状態を設定するコマンドを送信します（Super-Beacon用）。

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

対応APIバージョン：V9+

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

ポインタによるデータ構造:

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的に制御されます 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 – アラーム状態でビーコンが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	Reserved (0)

Marvelmind robotics

13.4.1.48. 汎用ユーザーペイロードデータを送信

汎用ユーザーペイロードデータを送信します。APIがModem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ModemのUART/USBポートを介して送信されます。受信データは、受信API関数、Arduinoのサンプル、ROS、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言語での宣言： 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:

型	説明
uint8_t	address - APIがModemに接続されている場合、ビーコンのアドレス (1...254) APIがビーコンに接続されている場合は該当なし
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構造:

型	説明
uint8_t	送信するデータのサイズ
256バイト	送信する汎用データバッファ

Marvelmind robotics

13.4.2. 汎用ユーザーペイロードデータを取得

送信APIファンクション、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言語での宣言： 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:

型	説明
void *	pdata - 受信するデータへのポインタ

ポインタ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バイト	受信データのバッファ

Marvelmind robotics

13.4.3. 手動距離測定用コマンドを送信

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。現行バージョンのソフトウェアでは、**IA** (Inverse architecture) でサポートされています。

関数名： **mm_send_distances_measurement_command**

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

対応APIバージョン：V9+

必要なライセンス: 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

ポインタによるデータ構造:

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	Reserved

13.4.4. modemからストリーミングデータを取得

前述の形式でモデム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言語での宣言： 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

ポインタによるデータ構造:

型	説明
uint8_t	この応答に含まれるストリームレコードの数 (0...16)
138*16 バイト	16個のストリーミングレコード、各138バイト (下記参照)
8バイト	Reserved

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のログファイル内の「line type」と同じ値。 例えば、41はMarvelmindプロトコルデータを意味します
8バイト	Reserved
128バイト	ストリームレコードデータ

Marvelmind robotics

13.4.5. デバイスタイプがModemかどうかを確認

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します。

関数名： **mm_device_is_modem**
C言語での宣言： bool mm_device_is_modem (uint8_t deviceType);
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 – 指定されたデバイスタイプがmodemに該当します

パラメータ:

型	説明
uint8_t	チェックするデバイスタイプ

Marvelmind robotics

13.4.6. デバイスタイプが固定ビーコンかどうかを確認

指定されたデバイスタイプが固定ビーコンに該当するかどうかを確認します。

関数名： **mm_device_is_beacon**
C言語での宣言： bool mm_device_is_beacon (uint8_t deviceType);
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 – 指定されたデバイスタイプは固定ビーコンに対応します

パラメータ:

型	説明
uint8_t	チェックするデバイスタイプ

Marvelmind robotics

13.4.7. デバイスタイプがHedgehogかどうかをチェック

指定されたデバイスタイプがHedgehogに対応するかどうかをチェックします。

関数名： **mm_device_is_hedgehog**
C言語での宣言： bool mm_device_is_hedgehog (uint8_t deviceType);
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 – 指定されたデバイスタイプはHedgehogに対応します

パラメータ:

型	説明
uint8_t	チェックするデバイスタイプ

Marvelmind robotics

13.5 Marvelmind API用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礎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CodeBlocks IDEでビルドされており、サンプルには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makeユーティリティを使用してビルドされており、サンプルにはそのための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。サンプルの関数を呼び出し、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で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いくつかの関数が含まれています。
marvelmind_api.c marvelmind_api.h	marvelmindAPILoad() – APIライブラリを読み込む marvelmindAPIFree() – APIライブラリが使用していたメモリを解放する 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	'get devices list'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、その位置および他のビーコンまでの距離に関するデータが含まれます。
marvelmind_pos.c marvelmind_pos.h	最新の位置データと最新の生距離データを読み込みます。これらのデータをデバイスリスト内で更新します。
marvelmind_utils.c marvelmind_utils.h	他のモジュールで使用されるいくつかの補助関数。

サンプルの動作方法:

1. 成功するまでシリアルポートを開こうとします
2. ポートが開かれると、プログラムはUSB経由で接続されている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読み取ります。デバイスリストは、現在marvelmind_devices.cモジュールに保存されているものと比較され、変更が検出された場合、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raw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raw距離を読み取ります。プログラムは、位置および距離をmarvelmind_devices.cのデバイスリスト内のデータと比較し、変更されている場合はデータを更新します。すべての変更された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、再接続された場合のために必要です

Marvelmind robotics

6. ユーザーが「**space**」ボタンを押すと、プログラムは「Enter command: 」メッセージを表示し、ユーザー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「**space**」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**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
プログラムを終了する	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にウェイクコマンドを送信 **wake 0** - すべてのデバイスにウェイクコマンドを送信
	コマンドの形式: **sleep <address>** 動作: スリープ送信コマンドを実行します。 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にデフォルト設定を適用
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、表示します。 例: **tele 5** - ビーコン5のテレメトリを読み取り、表示
サブマップコマンド	コマンドの形式: **submap add <submapId>** 動作: 指定したIDのサブマップを追加するコマンドを実行します。 例: **submap add 1** - サブマップ1を追加

Marvelmind robotics

	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します。 例: **submap delete 1** - サブマップ1を削除
	コマンドの形式: **submap freeze <submapId>** 動作: 指定されたサブマップIDのサブマップを凍結するコマンドを実行します。 例: **submap freeze 0** - サブマップ0を凍結
	コマンドの形式: **submap unfreeze <submapId>** 動作: 指定されたサブマップIDのサブマップの凍結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の凍結を解除
	コマンドの形式: **submap get <submapId>** 動作: 指定されたサブマップ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
	コマンドの形式: **submap testset <submapId>** 動作: 指定したsubmap IDの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: **submap testset 0** - submap 0の設定を変更
マップコマンド	コマンドの形式: **map erase** 動作: 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
	コマンドの形式: **map freeze** 動作: マップ凍結コマンドを実行します。 例:

Marvelmind robotics

	map freeze - マップを凍結
	コマンドの形式: **map unfreeze** 動作: マップの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
更新レートコマンド	コマンドの形式: **rate get** 動作: 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って表示
	コマンドの形式: **rate set <value>** 動作: 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
超音波コマンド	コマンドの形式: **usound get <address>** 動作: 指定されたビーコンの超音波設定の読み取りを実行します。 例: **usound get 5** - ビーコン5の超音波設定を読み取って表示する
	コマンドの形式: **usound testset <address>** 動作: 指定されたビーコンの超音波設定の書き込みを実行します。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: **usound testset 5** - ビーコン5の超音波設定を変更する
軸への接続コマンド	コマンドの形式: **axes <address_0> <address_x> <address_y>** 動作: 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；ビーコン4をX軸上（Y=0）に、ビーコン5をX軸より上（Y>0）に設定する

Modemから設定メモリのダンプを読み取る	コマンドの形式: **read_dump <offset> <size>** 動作: modemの設定メモリの読み出しダンプ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バイトを読み出します
modemへ設定メモリのテストダンプを書き込む	コマンドの形式: **write_dump_test <offset> <size>** 動作: modemの設定メモリの書き込みダンプ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ます
デバイスのソフトウェアリセット	コマンドの形式: **reset <address>** 動作: 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に対してソフトウェアリセットを実行します
空気温度コマンド	コマンドの形式: **temperature get** 動作: モデムから空気温度設定の読み取りを実行 例: temperature **get** 超音波用の空気温度設定を読み取り表示
	コマンドの形式: **temperature set <value>** 動作: モデムへ空気温度設定の書き込みを実行 例: temperature set 30 空気温度設定を30°Cに設定
ビーコンの位置を設定	コマンドの形式: **setloc <address> <X > <Y > <Z>** 動作: ビーコンの位置設定コマンドを実行します。X、Y、Zはメートル単位の座標です。 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

Marvelmind robotics

ビーコン間の距離を設定	コマンドの形式: **setdist <address1> <address2 > <distance>** 動作: ビーコン間の距離を設定するコマンドを実行します。Address1と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	コマンドの形式: **height_h get <address>** 動作: Hedgeの高さを取得する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
	コマンドの形式: **height_h set <address> <height>** 動作: Hedgeの高さを設定するコマンドを実行します。Addressはhedgehogのアドレスです。高さ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
	コマンドの形式: **height_b get <address> <submap_id>** 動作: 静止ビーコンの高さ取得コマンドを実行します。Addressはビーコンのアドレスです。Submap_idはビーコン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内の静止ビーコン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>** 動作: 固定ビーコンの高さ設定コマンドを実行します。Addressは固定ビーコンのアドレスです。Submap_idはビーコンが属するサブマップのIDです。高さはメートル単位 例: **height_b set 12 0 5.1** - サブマップ0内のビーコン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>** 動作: 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はビーコンのアドレスです。 例:

Marvelmind robotics

	rtp get 15 - ビーコン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>** 動作: set real-time player settingsコマンドを実行します。Addressはビーコンのアドレスです。プログラムはコマンドテスト用の定義済み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のテスト用リアルタイムプレイヤー設定をセットアップします
ジオリファレンス コマンド	コマンドの形式: **georef get** 動作: get georeferencing settings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
	コマンドの形式: **georef set <latitude> <longitude>** 動作: set georeferencing settings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みます
アップデートモード コマンド	コマンドの形式: **update_mode get** 動作: 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って表示します
	コマンドの形式: **update_mode set <mode>** 動作: 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** 動作: 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の設定	コマンドの形式: **alarm <address> <mode> <zone>** 動作: 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
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/モデムのペイロードを読み取り 例: **payload read 10** - ビーコンn10からペイロードを読み取り
	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を書込みテストパターンは100から始まる40バイト：100,101,..., 139
手動距離測定コマンド	コマンドの形式: **distance <manual/auto> <address> <max distance>** 動作: 手動距離測定コマンドを実行します。 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します。最大距離は5メートルです 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します。最大距離は30メートル(デフォルト)です 例: **distance auto** - 自動距離測定モードに戻ります

13.6　デバイスタイプ

以下は特定デバイスの「Device type ID」値のリストです:

デバイスタイプ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
42	Super-Beacon、SuperBeacon-2、Super-Beacon-3
43	Super-Beacon、SuperBeacon-2、Super-Beacon-3（hedgehogモード）
44	Industrial Super-Beacon、 Industrial Super-Beacon-2、 Industrial Super-Beacon-3、 Industrial RX、 Industrial RX-2、 Industrial RX-3
45	Industrial Super-Beacon（hedgehogモード）、 Industrial Super-Beacon-2（hedgehogモード）、 Industrial Super-Beacon-3（hedgehogモード）、 Industrial RX（hedgehogモード）、 Industrial RX-2（hedgehogモード）、 Industrial RX-3（hedgehogモード）
46	Super-Modem
48	Modem HW V5.1
56	Super-Beacon-4
57	Super-Beacon-4（hedgehogモード）
58	Industrial Super-Beacon -4、 Industrial RX-4
59	Industrial Super-Beacon -4（hedgehogモード）、 Industrial RX-4（hedgehogモード）

デバイスタイプidは、デバイスリストの取得コマンドとデバイスバージョンの読み取りコマンドで取得できます。

Marvelmind robotics

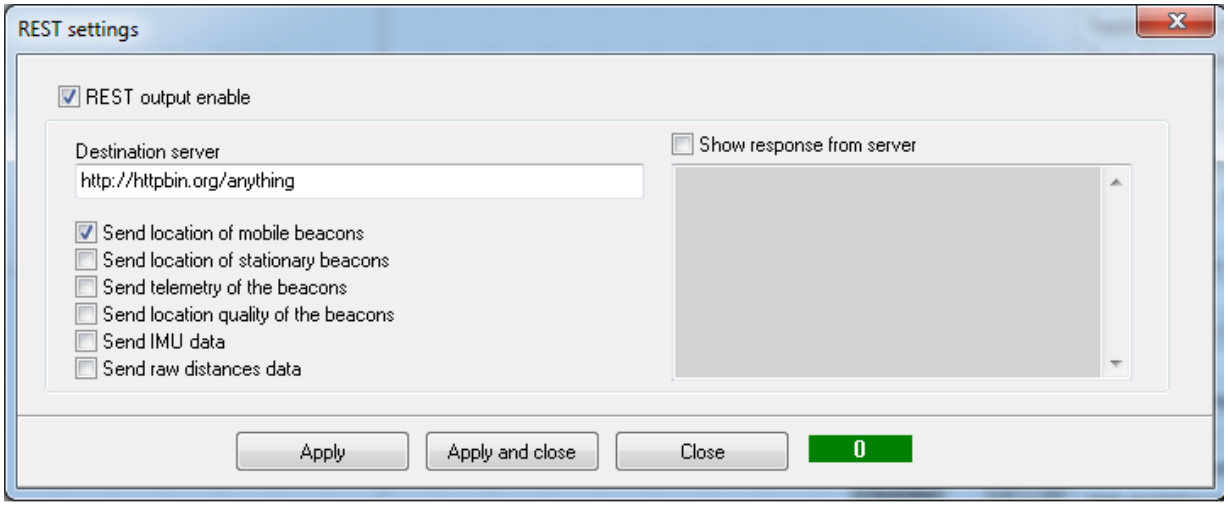
14. Marvelmind REST API

Marvelmindシステムは、位置などのデータをREST API経由で指定のhttpサーバーに送信できます。

REST APIは、システムの次のコンポーネントで利用できます：

Dashboard:	対応
Super-Modem:	要望に応じて対応

DashboardのREST APIの設定は、メニュー「Settings/REST settings」から行えます：



デフォルトの送信先サーバー（http://httpbin.org/anything）はリクエストをそのままエコーで返すだけなので、テストに使用できます。チェックボックス「Show response from server」をオンにすると、サーバーからの最後の応答を確認できます。

node.jsで書かれたこのサーバー側のサンプルを、お客様のhttpサーバーにインストールすることもできます。このサンプルもリクエストのエコーを返し、お客様のアプリケーションのベースとして使用できます。

Marvelmind REST APIの基本的な考え方：

- クライアントアプリケーション（DashboardまたはSuper-Modem）は、指定されたサーバーにhttp POSTリクエストを送信します。
- サーバーへのアクセスのレイテンシが大きくなる可能性があるため、クライアントはデータの各部分（位置更新など）を個別のパケットとしては送信しません。データはクライアント側でバッファされ、1 secに一回、または1 Kbyteを超えるデータが溜まった時点でサーバーに送信されます。そのため、一つのRESTリクエストに複数のデータ項目の配列が含まれる場合があります。
- データはJSON形式で送信されます

以下に、このAPIのJSONデータフォーマットを説明します。

Marvelmind robotics

14.1 JSONデータの最上位フォーマット

最上位では、JSONデータはバッファされたデータ項目の配列です：

```
{
  “Marvelmind_data”: [
     <データ項目1>,
     <データ項目2>,
     ……..
    <データ項目n>
  ]
}
```

各データ項目も、以降のセクションで説明するいずれかのフォーマットのJSON構造です。

14.2 モバイルビーコン位置データ項目

モバイルビーコンの位置データのフォーマットは次のとおりです：

```
{
  “HedgePos”: {
     <パラメータ1>,
     ……..
     <パラメータn>
  }
}
```

モバイルビーコンの位置のパラメータを表に示します：

名前	型	説明
“Address”	integer	モバイルビーコンのアドレス
“Timestamp”	integer	位置更新のタイムスタンプ。 “IsRealtime”がtrueの場合、このタイムスタンプはUnix時間（1970.01.01 00:00:00からのミリ秒数）を意味します “IsRealtime”がfalseの場合、このタイムスタンプはモバイルビーコンのリセットからの時間（ミリ秒）を意味します
“IsRealtime”	boolean	タイムスタンプの説明を参照
“X_mm”	integer	位置のX座標、mm
“Y_mm”	integer	位置のY座標、mm
“Z_mm”	integer	位置のZ座標、mm
“Flags”	integer	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でください。 ビット1：常に1 ビット2：1 - ユーザーボタンが押されている ビット3：1 - ユーザーデバイスへアップロードするデータあり ビット4：1 - ユーザーデバイスからデータのダウンロードを要求 ビット5：1 – 二番目のユーザーボタンが押されている ビット6：未使用 ビット7：– 1 – ジオフェンシングゾーン外
“Angle_deg”	integer	ペアのモバイルビーコンの場合 – 方位角、度

Marvelmind robotics

14.3 固定ビーコン位置データ項目

固定ビーコンの位置データのフォーマットは次のとおりです：

```
{
  “StationaryBeaconPos”: {
     <パラメータ1>,
     ……..
     <パラメータn>
  }
}
```

固定ビーコンの位置のパラメータを表に示します：

名前	型	説明
“Address”	integer	固定ビーコンのアドレス
“X_mm”	integer	位置のX座標、mm
“Y_mm”	integer	位置のY座標、mm
“Z_mm”	integer	位置のZ座標、mm

Marvelmind robotics

14.4 ビーコンテレメトリデータ項目

モバイルビーコンまたは固定ビーコンのテレメトリのフォーマットは次のとおりです：

```
{
  “BeaconTelemetry”: {
     <パラメータ1>,
     ……..
     <パラメータn>
  }
}
```

テレメトリのパラメータを表に示します：

名前	型	説明
“Address”	integer	ビーコンのアドレス
“Vbat_mv”	integer	ビーコンのバッテリー電圧、mV
“RSSI_dBm”	integer	RSSI（受信無線信号強度）レベル、dBm

Marvelmind robotics

14.5 モバイルビーコン位置品質データ項目

モバイルビーコンの位置品質のフォーマットは次のとおりです：

```
{
  “LocationQuality”: {
     <パラメータ1>,
     ……..
     <パラメータn>
  }
}
```

位置品質のパラメータを表に示します：

名前	型	説明
“Address”	integer	モバイルビーコンのアドレス
“Quality”	integer	品質

Marvelmind robotics

14.6 IMUデータ項目

モバイルビーコンのIMUデータのフォーマットは次のとおりです：

```
{
  “IMUData”: {
     <パラメータ1>,
     ……..
     <パラメータn>
  }
}
```

このパケットには、生IMUデータまたはIMUフュージョンデータが含まれます。

生IMUデータのパラメータを表に示します：

名前	型	説明
“Address”	integer	モバイルビーコンのアドレス
“Timestamp”	integer	IMUデータ更新のタイムスタンプ。 “IsRealtime”がtrueの場合、このタイムスタンプはUnix時間（1970.01.01 00:00:00からのミリ秒数）を意味します “IsRealtime”がfalseの場合、このタイムスタンプはモバイルビーコンのリセットからの時間（ミリ秒）を意味します
“IsRealtime”	boolean	タイムスタンプの説明を参照
“Accel_x”	integer	加速度計、X軸、1 mg/LSB
“Accel_y”	integer	加速度計、Y軸、1 mg/LSB
“Accel_z”	integer	加速度計、Z軸、1 mg/LSB
“Gyro_x”	integer	ジャイロスコープ、X軸、0.0175 dps/LSB
“Gyro_y”	integer	ジャイロスコープ、Y軸、0.0175 dps/LSB
“Gyro_z”	integer	ジャイロスコープ、Z軸、0.0175 dps/LSB
“Compass_x”	integer	コンパス、X軸、1100 LSB/Gauss
“Compass_y”	integer	コンパス、Y軸、1100 LSB/Gauss
“Compass_z”	integer	コンパス、Z軸、980 LSB/Gauss

コンパスデータは、IMU搭載のHW v4.9ビーコンでのみ利用できます。

IMUフュージョンデータのパラメータを表に示します：

名前	型	説明
“Address”	integer	モバイルビーコンのアドレス
“Timestamp”	integer	IMUデータ更新のタイムスタンプ。 “IsRealtime”がtrueの場合、このタイムスタンプはUnix時間（1970.01.01 00:00:00からのミリ秒数）を意味します “IsRealtime”がfalseの場合、このタイムスタンプはモバイルビーコンのリセットからの時間（ミリ秒）を意味します
“IsRealtime”	boolean	タイムスタンプの説明を参照
“Pos_x”	integer	ビーコンのX座標（フュージョン）、mm
“Pos_y”	integer	ビーコンのY座標（フュージョン）、mm
“Pos_z”	integer	ビーコンのZ座標（フュージョン）、mm

Marvelmind robotics

“Quaternion_w”	integer	回転クォータニオンのWフィールド（角度）
“Quaternion _x”	integer	回転クォータニオンのXフィールド（角度）
“Quaternion _y”	integer	回転クォータニオンのYフィールド（角度）
“Quaternion _z”	integer	回転クォータニオンのZフィールド（角度）
“Velocity_x”	integer	X方向速度（フュージョン）、mm/s
“Velocity_y”	integer	Y方向速度（フュージョン）、mm/s
“Velocity_z”	integer	Z方向速度（フュージョン）、mm/s

IMUフュージョンの位置と速度は、IMU搭載のHW v4.9ビーコンでのみ利用できます。

14.7 生距離データ項目

モバイルビーコンの生距離データのフォーマットは次のとおりです：

{

“RawDistances”: {

<パラメータ1>,

……..

<パラメータn>

}

}

生IMUデータのパラメータを表に示します：

名前	型	説明
“HedgeAddress”	integer	モバイルビーコンのアドレス
“Distances”	array	固定ビーコンまでの距離の配列： [<距離項目1>, …… <距離項目m>]

各距離項目のパラメータを表に示します

名前	型	説明
“Address”	integer	固定ビーコンのアドレス
“Distance_mm”	integer	モバイルビーコンから固定ビーコンまでの距離、mm

Marvelmind robotics

15. MQTTプロトコル

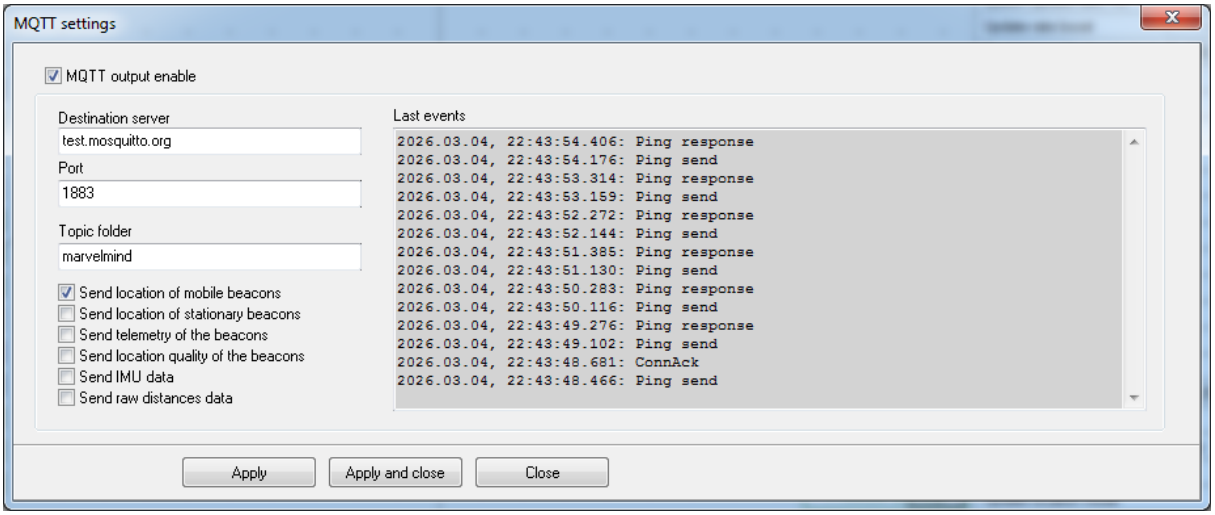
Marvelmindシステムは、位置などのデータをMQTTプロトコルで指定のMQTTブローカーにパブリッシュできます。

MQTTプロトコルは、システムの次のコンポーネントで利用できます：

Dashboard: 対応

Super-Modem: 要望に応じて対応

DashboardのMQTTの設定は、メニュー「Settings/MQTT settings」から行えます：



デフォルトの送信先サーバー（ブローカー）はtest.mosquitto.orgです。

パブリッシュ先のポートとトピックも指定できます。

Marvelmindは、パブリッシュされたデータを受信するサブスクライバーのサンプル（**python**で記述）を提供しています。このサンプルは上記のブローカーとトピックの設定に合わせてあり、受信したすべてのデータをコンソールに出力します。

Marvelmind MQTTプロトコルの基本的な考え方はREST APIと同じです：

- クライアントアプリケーション（DashboardまたはSuper-Modem）は、更新されたデータを指定のブローカーとトピックにパブリッシュします。
- サーバーへのアクセスのレイテンシが大きくなる可能性があるため、クライアントはデータの各部分（位置更新など）を個別のパケットとしては送信しません。データはクライアント側でバッファされ、1 secに一回、または1 Kbyteを超えるデータが溜まった時点でブローカーに送信されます。そのため、一つのMQTTリクエストに複数のデータ項目の配列が含まれる場合があります。
- データはJSON形式で送信されます

MQTTで送信されるJSONデータのフォーマットは、REST APIと同じです。

Marvelmind robotics

16. ユーザーデバイスとの間でのユーザーデータの送受信

Marvelmindは、Marvelmindシステムを介したユーザーデータ伝送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経由でデータを送信し、モバイルビーコンのUARTまたはUSB経由で受信する
- モバイルビーコンのUARTまたはUSB経由でデータを送信し、ModemのUARTまたはUSB経由で受信する。

Super-ModemはUDP経由でのユーザーデータの送受信もサポートしています。

データ伝送プロトコルは、本書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およびユーザーデバイスからのデータ伝送プロトコル
- データの送受信のためのAPI関数

Marvelmindは通信のためのさまざまなソフトウェア例を提供しています。

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

サンプルの全リスト：

- 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『01_Common_Indoor_positioning_SW/ 06_Examples/ arduino』に配置されています。『hedgehog_sample_uart_user_data_receive_v2』はユーザーデータ受信用、『hedgehog_sample_uart_user_data_send_v2』はユーザーデータ送信用です。
- API通信サンプルは、Marvelmindソフトウェアパッケージのフォルダ『01_Common_Indoor_positioning_SW/ 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）に配置されています。データの送信または受信は、本書に記載の通り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』に配置されています。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単純に表示します。
- ストリーミングデータ受信用のPythonサンプルは、Marvelmindソフトウェアパッケージ内のフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ python」に配置されています。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、モバイルビーコンまたはモデムから受信したすべてのデータ（ユーザーデータを含む）を単純に出力します。
- ストリーミングデータ受信用のROSパッケージサンプルは、Marvelmindソフトウェアパッケージ内のフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ ROS」に配置されています。また、このパッケージはリポジトリでも入手可能です。このROSパッケージを使用すると、APIを介し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受信用のROS-2パッケージサンプルは、Marvelmindソフトウェアパッケージ内のフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ ROS2」に配置されています。また、このパッケージはリポジトリでも入手可能です。このROSパッケージを使用すると、APIを介し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内のフォルダ「01_Common_Indoor_positioning_SW/ 06_Examples/ c」に配置されています。このサンプルは、Super-ModemまたはDashboardからUDP経由で受信したすべてのデータ（ユーザーデータを含む）を単純に出力します。USBの代わりにUDP経由でSuper-Modemに接続するためにAPIを使用する場合、APIを介してUDPでユーザーデータを送信することも可能です。

17. RHU (Radio Head USB)

RHUは専用ファームウェアを搭載したModem HW v5.1です。モバイルビーコンが無線でストリーミングするデータを受信し、USB（仮想UART）とUARTに中継します。

RHUはメインのモデムおよびMarvelmindナビゲーションシステムのチャネルとは別の無線チャネルで動作するため、ナビゲーションシステムの無線通信を妨げません。これにより、モバイルビーコンは高いレート（~80 Hz）かつ最小のレイテンシで位置データを無線送信できます。

RHUの主な目的は、モバイルビーコン上のIMUセンサーフュージョンによる位置更新を受信することです。詳細は、アーキテクチャ比較ドキュメントの「Split modem」アーキテクチャを参照してください。

RHUは、モバイルビーコンの設定に応じて、MarvelmindプロトコルまたはNMEA0183プロトコルでデータを出力します。

17.1 接続

USB（仮想**UART**）。RHUはモデムと同じ方法でUSBホストに接続します（この章を参照）。ホスト側のシリアルポートのパラメータ（ボーレート、ビット数、パリティ）は任意です。

UART。UARTはModem HW v5.1のコネクタにあります（この章を参照）。データを受信するには、GroundとUSART3_TXの2本の線を接続するだけで十分です。RHUはUART経由ではデータを受信しません。UART送信器の論理レベルはCMOS 3.3 Vです。デフォルトのボーレートは500 kbpsで、Dashboard（RHUのパラメータ「UART speed, bps」）で次の中から設定できます：4.8、9.6、19.2、38.4、57.6、115.2、500 kbps。データ形式：8ビット、パリティなし、ストップビット1。

USBとUARTは同じデータを出力します。DashboardをUSBでRHUに接続している場合、RHUのUSBはDashboardが使用します。この場合は、ユーザーデバイスをRHUのUARTに接続してください。

17.2 Marvelmindプロトコル

すべてのパケットは、ストリーミングパケットの一般的な構造に従います（この章を参照）。RHUは次のパケットを送信します：

データコード	パケット	レート
0x0085	モバイルビーコンの位置と向き（IMUセンサーフュージョン）	~80 Hz
0x0086	RHUが復元したモバイルビーコンの位置	パケット0x0085が失われた場合のみ
0x008A	モバイルビーコンの超音波による位置	超音波の更新レート
0x1116	RHUのテレメトリ	1 Hz

17.2.1 IMUセンサーフュージョンデータのパケット、データコード0x0085

このパケットは、モバイルビーコンがUART/USBで送信するパケット0x0085と同じ構造ですが（この章を参照）、次の点が異なります：

Offset	バイト数	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	0x0085
4	1	uint8_t	送信データのバイト数	46
5	4	int32_t	モバイルビーコンのX座標（フュージョン）、mm	
9	4	int32_t	モバイルビーコンのY座標（フュージョン）、mm	
13	4	int32_t	モバイルビーコンの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	モバイルビーコンのX方向速度（フュージョン）、mm/s	
27	2	int16_t	モバイルビーコンのY方向速度（フュージョン）、mm/s	
29	2	int16_t	モバイルビーコンのZ方向速度（フュージョン）、mm/s	
31	6	6バイト	Reserved	
37	1	uint8_t	モバイルビーコンのアドレス	
38	1	1バイト	Reserved	
39	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。モデムとDashboardによりすべてのデバイスで同期された時刻	
47	1	uint8_t	フラグ: ビット0 = 位置なし（座標と速度は0） ビット2 = 速度なし ビット5 = モバイルビーコンは静止中	

Marvelmind robotics

			その他のビットはreserved	
48	1	1バイト	Reserved	
49	2	int16_t	モバイルビーコンの方位（縦軸の方向）、0.1度単位	
51	2	uint16_t	CRC-16（付録1参照）	

注: クォータニオンは10000の値に正規化されています。

Marvelmind robotics

17.2.2 復元された位置のパケット、データコード0x0086

モバイルビーコンのパケット0x0085が無線上で失われた場合、RHUは代わりに、それまでのパケットから外挿した位置を含むパケット0x0086を送信します。

Offset	バイト数	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	0x0086
4	1	uint8_t	送信データのバイト数	46
5	46	46バイト	パケット0x0085と同じデータ。座標X、Y、ZはRHUが外挿した値、タイムスタンプは失われたパケットのもの	
51	2	uint16_t	CRC-16（付録1参照）	

Marvelmind robotics

17.2.3 超音波による位置のパケット、データコード0x008A

モバイルビーコンは、超音波の測定サイクルごとにこのパケットを送信します。超音波による距離のみ（IMUなし）から計算した、現在と前回のサイクルの位置が含まれます。

Offset	バイト数	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	0x008A
4	1	uint8_t	送信データのバイト数	40
5	1	uint8_t	モバイルビーコンのアドレス	
6	8	int64_t	測定のタイムスタンプ – Unix時間、ms	
14	4	int32_t	モバイルビーコンのX座標、mm	
18	4	int32_t	モバイルビーコンのY座標、mm	
22	4	int32_t	モバイルビーコンのZ座標、mm	
26	2	uint16_t	推定位置誤差（sigma）、mm。0 = 不明	
28	1	uint8_t	フラグ: ビット0 = 位置が有効（0の場合、座標は0） ビット1 = 位置はトラッキングフィルターで処理済み（それ以外は生データ） ビット2 = 位置は測定値ではなく予測値 ビット3 = 使用した距離が3未満 ビット4 = 高信頼度	
29	1	uint8_t	測位品質、0..100 %	
30	2	uint16_t	前回の位置の経過時間：前回と今回の測定の間隔、ms。0 = 前回の位置なし	
32	4	int32_t	前回のX座標、mm	
36	4	int32_t	前回のY座標、mm	
40	4	int32_t	前回のZ座標、mm	
44	1	uint8_t	前回の位置のフラグ（オフセット28と同じ）	
45	2	uint16_t	CRC-16（付録1参照）	

前回の位置により、無線上でパケットが一つ失われても測定値は失われません。

Marvelmind robotics

17.2.4 RHUテレメトリのパケット、データコード0x1116

Offset	バイト数	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	0x1116
4	1	uint8_t	送信データのバイト数	20
5	1	uint8_t	RHUの無線アドレス	
6	2	uint16_t	電源電圧、mV	
8	4	uint32_t	電源投入からの時間、秒	
12	1	int8_t	最後に受信した無線パケットのRSSI。 RSSI, dBm = 値 − 81	
13	1	uint8_t	直近一秒間に受信した無線パケット数	
14	5	5バイト	Reserved	
19	1	uint8_t	直近一秒間のCRCエラーの無線パケット数	
20	1	uint8_t	直近一秒間の短すぎる無線パケット数	
21	2	2バイト	Reserved	
23	1	uint8_t	直近一秒間のリンク品質指標（LQI）の最小値、0..127（小さいほど良好）。 0x7f = パケットなし	
24	1	uint8_t	直近一秒間のLQIの平均値	
25	2	uint16_t	CRC-16（付録1参照）	

Marvelmind robotics

17.3 NMEA0183プロトコル

モバイルビーコンの設定でNMEA0183プロトコルが選択されている場合（この章を参照）、RHUはモバイルビーコンで有効にしたNMEA0183メッセージを、そこで説明しているフォーマットで出力します。

メッセージをフル更新レート（~80 Hz）でUART出力するには、RHUのUART速度を500 kbpsに設定してください。それより低い速度では、メッセージの一部が失われます。

18. 連絡先

追加のサポートについては、ご質問をinfo@marvelmind.comまでお送りください

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バイトのフレームの最後の二バイトには、フレームの最初の(N-2)バイトに適用したCRC-16が格納されます。データを確認するには、Nバイトのフレーム全体にCRC-16を適用します。結果の値はゼロになります。

以下は「C」言語での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;
}
```


Marvelmind robotics

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からホストへ）

Offset	バイト数	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	
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最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になります。
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のエラー
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

Marvelmind robotics