DM Device Integration Guide

Version 1.5 - 4 November 2020

Version History

Version	Date	Changes
1.0	11th September 2019	Version 1
1.1	27th September 2019	Fixed Numbering
1.2	13th December 2019	Added Trip Movement Trips
1.3	6th March 2020	Increased recommended receive buffer size
1.4	2nd April 2020	Corrected Yabby Wi-Fi Log Reasons
1.5	12th Oct 2020	Added Falcon/Eagle task upload information Updated Links Added BLE section.

1 INTRODUCTION

This guide is to be read in conjunction with the standard Digital Matter integration documents which consist of:

- DMT Data Fields Document
- DMT Log Reasons
- DMT Device Integration – Flexi Record (TCP Integration)
- DMT JSON Device Integration (HTTP Integration)
- Direct 3rd Party Server Integration (Device direct to 3rd Party Server)

The purpose of this guide is to provide the reader with an understanding of the operation of Digital Matter cellular devices, along with some payload examples and other notes.

The article Choosing An Integration Strategy sets the scene for the different integration options. This document applies to JSON HTTP integration via OEM; TCP integration via OEM; and Direct TCP Integration bypassing OEM.

The integration documents provide a complete description of what payloads the device will send, but knowledge of why and when these will be sent by the device will allow users to best take advantage of smart device features in their end software platform.

Notes:

- The operation of the device can be greatly affected by the system parameters in use. Unless otherwise stated, any reference to the operation of devices refers to a device using default parameters.

- Certain behaviour that only occurs at start-up may not be covered in this guide if it is not part of the regular operation of the device. Though in general most devices will wake, acquire a fix, heartbeat and then sleep.
- In general, parameter changes will not change device payloads or structure greatly - but why and when they are sent could vary
- This document is written with the latest firmware at the time of writing in mind. If you are experiencing unexpected behaviour – update your firmware to the latest version and re-test. If you are still unsure contact DM Support.
- Any payload examples given in this document are given purely as a guide to assist integrators in their understanding of device operation and

Other relevant documentation for integrators - available at support.digitalmatter.com

Some key resources:

- **Cellular Payload Decoder**
  This payload decoder is written in JavaScript and the code may be inspected and used freely as example code.
- Getting Started Guides for each product - each of these guides contains default settings + default I/O mappings.
- Wired Devices - Overview of Operation
- Battery Powered Devices - Overview of Operation
- OEM Getting Started Guide

2 KEY CONCEPTS

2.1 Log Reasons and Data Fields

A log reason is why a record was logged by the device. E.g. a GPS point might be logged because a trip started, or ended, and it is useful to distinguish between these two events to demarcate trips.

Another example is that a point might be logged due to an input change.

The integration document covers many log reasons and data fields. However not all are used by all devices, some are deprecated, and others may only be sent when specific (non-default) settings are applied to devices.

Not all data fields need to be integrated. When integration, systems mut be able to accommodate any unknown fields. They should be ignored by the software platform and allow regular operation to continue.

2.2 Log Reasons Per Device

For a table showing which Log Reasons are used by which devices see:

Appendix A – Log Reason Table

2.3 Data Fields Per Device

For a table showing which Data Fields are used by each device see:

Appendix B – Data Fields Per Device

Not all fields will be sent by each device. Some fields, e.g. High-G will not be sent as default unless the device is configured to send these fields.

As mentioned in the integration documents, any unknown fields should be ignored. The record header will contain the Field ID and length of this data field – that number of bytes can be ignored, meaning any new fields will not cause any server issues.

For both the Log Reasons and Data Fields tables – simply use them as a starting point. Device firmware updates may change them in future.

2.4 Default Input Mappings

Default input mappings for each device are available on the DM Knowledge Base.

This will fill in any gaps for Digital and Analogue inputs sent in these fields which may not be detailed in the other integration documents.

2.5 Device Status Flags

As mentioned later in this document, each device has common status flags, send in Field 2 (digital input data) indicating key states.

Device Status Flags:

- b0 = trip status. 1 = “in trip”
- b1 = internal battery good
- b2 = external power good
- b3 = connected to GSM
- b4 = shunting power from battery
- b5 = external power enabled
- b6 = tamper alert
- b7 = recovery mode active

They can be used for a variety of uses such as low battery alerts, detecting towing trips, power removed alerts – they are common against device types so general alerts can be implemented.

3 ASSET LOCATION AND TRIP TRACKING

While devices may use different methods to track trips (i.e. physical ignition wire vs GPS movement) In general the logs and data fields that they send to the server are similar. The frequency and reason such logs are sent is determined by system parameters.

When a device enters a trip, Bit 0 of the status flags (sent in field ID 2 – Digital Data) will be set on to ‘1’.

All devices also treat bit 0 of the digital inputs as their “Ignition input” whether they have a wired ignition or not. Most will set this input active when they enter a trip, more detail on this later.

3.1 Default Settings/Trip Detection

Consult the Getting Started Guide for each product to see default settings for each device.

Knowledge Base

3.2 Overview of Operation + Trip Detection

Trips can be detected via the following methods.

1. Physical Ignition Input (used on powered devices)
2. GPS Movement – See “Standard Tracking Mode” in this article. Most battery powered devices use this method, and Powered Devices also use this method in addition to the ignition input by default.
3. Run Detect – for powered devices based on external voltage changes
4. Jostle Trips – Based on accelerometer movement – available on battery powered devices.

4 MOVEMENT EXAMPLES

Powered Devices

Powered devices use either Physical Ignition, GPS Movement, or Run Detect to begin a trip. By default, Ignition and GPS movement trips are enabled, run detect is not. The exception is the Bolt which has no physical ignition input – so only movement trips are enabled by default.

Battery Powered Devices

In general battery powered devices detect trips by method 3 (movement trips).

The exception is the Yabby which always uses Jostle Trips as its sole tracking mode. All other battery powered devices can have this mode set as an option.

For this example, we will consider the movement of the device in the following way:

- Device is stationary (out of trip) for a period
- Device starts a trip (begins to move)
- Device on the move for some time
- Device ends a trip.

4.1.1 Device is stationary

Please see key documents:

- Wired Devices - Overview of Operation
- Battery Powered Devices - Overview of Operation

4.1.1.1 Powered Devices

Every 60 min by default for most devices (or whatever the heartbeat parameter is set to) the device will send a heartbeat:

- Log Reason 11 – Heartbeat
- Fields 0, 2 and 6 – GPS + Digital and Analogue Data

On heartbeats, powered devices turn on their GPS to freshen the data however they **do not send any new fixes** to prevent GPS wander.

This will result in the following example scenario:

0945

- Trip ends, device gets a GPS fix (timestamp 0945)

1045

- Heartbeat time of 1045 (RTC Datetime at beginning of data record)
- GPS timestamp of 0945 (GPS UTC DateTime in data field 0)

Subsequent heartbeats will keep sending the same GPS time. As soon as the device is on the move again new fixes will be sent – so excellent tracking is achieved – it is just important to note this behaviour.

4.1.1.2 Battery Powered Devices

Battery powered devices **do send a new fix each heartbeat**

The record and GPS times should match in most cases **(unless a fix fails, in which case the last known fix is sent)**

4.1.2 Device begins a trip

For more information on how this is determined, check the knowledge base for the relevant product.

4.1.2.1 Ignition Trip – General case for powered devices

Devices with a wired ignition will go into trip after the ignition line becomes active.

The following 2 records are sent, in this order.

Record 1:

- Log Reason 9 – Digital Input Change
- Digital Input 0 will now be set on to '1'
- Fields 0, 2 and 6

Record 2:

- Log Reason 1 – Start of Trip
- Trip Status Flag is set on.
- Fields 0, 2, 6 (GPS + Inputs).
- The GPS lat/long of the last trip's end position will be used

4.1.2.2 Movement Trip

Used by battery powered devices Remora/Remora2, Oyster/Oyster2 and Falcon as the default setting.

Powered devices - Dart/Dart2, G62, G100, Bolt.

On default settings, the device must cross a movement threshold (250m default) before a trip is registered. The following 2 records will be sent together, in the below order.

Record 1:

- Log Reason 1 – Start of Trip
- Fields 0, 2, 6 (GPS + Inputs).
- The GPS lat/long of the last trip's end position will be used

Record 2

- Log Reason 2 – Distance Travelled
- Fields 0, 2, 6 (GPS + Inputs) – new GPS point

In this scenario, the trip was started because enough (>250m) distance had been covered. Once the trip starts, the device will upload.

Due to variances in GPS along with this threshold, a trip is not actually detected to have started exactly where the device last stopped.

The device sends the 2 records above so that end and start points line up in the end software platform. Otherwise there would be gaps between the end point and new start point, which doesn't line up with reality or look particularly pleasing in the platform.

This is dictated by the *Assumed Start Point Range* parameter (default of 10000m) – found under the *Movement Trips* tab in OEM.

4.1.2.3 Run Detect Trip

If Run Detect is set up and mapped to DI 0, a change in external voltage – in accordance with the thresholds set in parameters, will begin a trip

The record sent is

- Log Reason 1 – Start of Trip
- Fields 0, 2, 6 (GPS + Inputs).
- The GPS lat/long of the last known position will be used

4.1.2.4 Jostle Trip

Records as per a movement trip (see section above). However the reason the trip has started is due to accelerometer movement, rather than GPS movement.

4.2 Device on the move

Every log period during a trip (typically 2 min for battery powered devices and 60s for powered) the device logs a GPS point.

For battery powered devices - every 30 min, all of these records will be uploaded together in a batch, (15 records).

Powered devices remain connected to the server during a trip and upload each point immediately.

Log Reason 3 – Elapsed Time

- Fields 0, 2, 6 (GPS + Inputs)

Other log reasons that may be sent during a trip by powered devices include:

3 - Elapsed time, 4- Speed change, 5 - Heading change, 6 - Distance travelled, 7 - Stationary

These logs allow accurate vehicle paths to be tracked (e.g. around corners) without having to set a high update rate.

Data would arrive in the form as above with the relevant log reason.

4.2.1 Trip ends

For devices that use multiple methods to detect trips, the trip will only end via the method it was started.

E.g. for a powered tracker, using both physical ignition + movement detection, if an ignition trip starts, the trip won't end until the ignition is off.

If a movement trip starts, then the ignition is turned on, turning the ignition off won't end the trip, only when movement ends will the trip end.

4.2.1.1 Movement Trip End

A movement trip will end if:

- No GPS movement has been detected for *Trip End Time* (300s, 5 min by default)
- GPS has been lost for *Trip End Time, GPS Lost* (300s, 5 min by default)

Any “Elapsed Time” records which have been logged and not yet uploaded will be sent. Followed by an End of Trip Record.

Logged GPS Positions

- Records as per 4.2

End of Trip Record

- Log Reason 2 – End of Trip
- Fields 0, 2, 6, 15

4.2.1.2 Ignition Trip End

Record 1:

- Log Reason 9 – Digital Input Change
- Digital Input 0 will now be set off to ‘0’
- Fields 0, 2 and 6

Record 2:

- Log Reason 1 – End of Trip
- Fields 0, 2, 6 (GPS + Inputs).

5 DETECTING TOWING TRIPS

Consult this article for information:

[Transported Trips - What Are They?]

Sometimes assets such as excavators, generators etc will be towed. It is important to be able to distinguish the difference between an asset being towed, and when it is ‘active’ i.e. ignition is on and working, to properly track hours of utilisation/run hours.

As mentioned in the article, the way to effectively track run hours is to set up your device’s parameters such that the Ignition input is only active when the device is ‘running’ i.e. engine on.

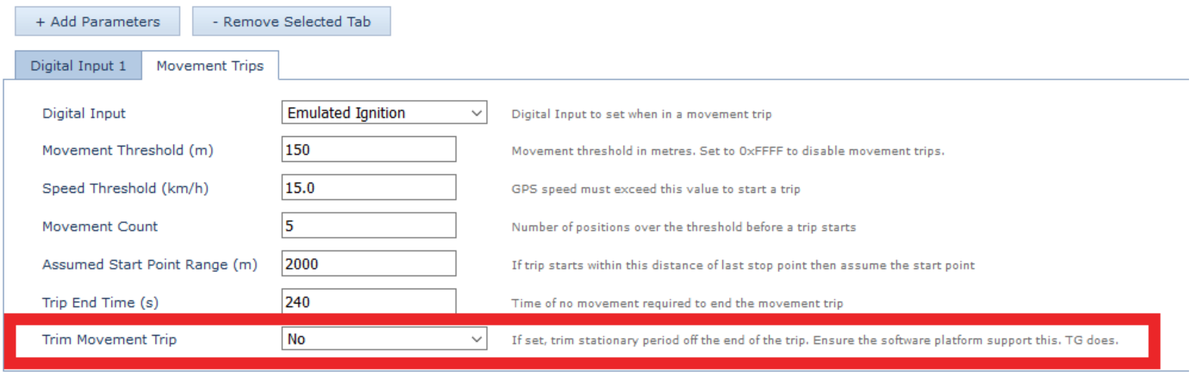
Then you have the following states:

B0 of status flags (trip state)	Digital input 0 (ignition input)	Meaning
Inactive	Inactive	Device is not moving, out of trip
Inactive	Active	Invalid Combination
Active	Inactive	Device is being towed, don’t increment run hours/odo

Active	Active	Device is being used, increment run hours and odometer

6 TRIM MOVEMENT TRIPS

Some devices have a *Trim Movement Trip* parameter



This relates to data sent in Field 15

FID = 15: Device Trip Type and Data

Length = 3

Offset	Data Type	Length	Description	Unit
0	BYTE	1	**Device Trip Type** This is the reason that the trip was started by the device. 0 = NONE 1 = Ignition 2 = Movement 3 = Run Detect 4 and up = reserved	
1	UINT16	2	**Movement trip trimming amount** Valid only for Device Trip Type = 2 Indicates the number of seconds that should be trimmed off the movement trip	seconds

Implementation detail:
Logged as a field on the Digital Input change when the ignition is turned off on a movement trip.

This field is sent at the end of a movement trip, and the *Trim Movement Trip* parameter works as follows

When set to NO:
When a movement trip ends, the *Movement Trip Trimming Amount* value will always be 0

When set to YES:
When a movement trip ends, the *Movement Trip Trimming Amount* will be equal to the *Trip End Time*.

Then the software platform can deduct this time from the trip length, and trip end time if necessary. This gives a truer indication of how long a device was **actually** moving for, as the trip end timeout always increases the length of a trip by this amount.

7 BEHAVIOUR WHEN IN POOR GPS RECEPTION

If a GPS fix fails, typically due to poor signal, but at times due to incorrect parameter setup, **the last valid fix will be reported**.

As an example, consider a device set to report every 2 minutes during a trip, which fails some fixes – the logs will appear as:

- 0900 – In trip, GPS fix successful
  - Log Reason 3, Elapsed Time
  - Record Time of 0900 (as this was when it was taken) in record header
  - GPS UTC datetime in Field 0 of 0900
- 0902 – In trip, fix fails
  - Log Reason 3, Elapsed Time
  - Record time of 0902
  - GPS Data will be the same as sent for the previous record, time of 0900
- 0904 – In Trip, fix fails
  - Log Reason 3, Elapsed Time
  - Record Time 0904
  - GPS Time of 0900, same lat/long as the 0900 valid fix
- 0906 – Valid Fix
  - Log Reason 3, Elapsed Time
  - Record Time and GPS Time of 0906.

8 BEHAVIOUR WHEN UNABLE TO CONNECT

Digital Matter Cellular devices store all their records in their internal memory until they receive a commit response from the server to indicate that these records have been saved.

If a device cannot connect to the network it will not receive a commit response so no records will be lost, and they will be uploaded when the device can next connect.

On each upload attempt a device will try to upload all records which are stored in memory.

The record time can be compared to the time a message is received on the server to determine transmission delays.

9 POWER AND BATTERY MONITORING

9.1 Powered Devices

Most powered devices have both an external power input and an internal LiPo backup battery (exceptions exist such as the Bolt).

This can be monitored via:

1. Analogue 1 (Field 6) Displays the internal battery voltage. This is the Li-Po Battery for powered units, or the primary cells for battery powered devices
2. Analogue 2 (in Field 6) will display the external voltage in mV x 10
3. Bit 1 of the Digital Status Flags (Internal Battery Good)
4. Bit 2 of the Digital Status Flags (External Power Good)
5. Log Reason 15 – External Power Change. If a record with this log reason is received, external power has been removed or connected (check the above flag to distinguish)
6. Log Reason 16 – System Power Monitoring. This record indicates the internal battery has dropped below a certain threshold.

The device firmware will set these flags to 1 if ‘good’ and 0 if not. The levels are selected at a value appropriate for the device type, **so all system alerts can simply be run off this flag, across multiple device types if desired.**

9.2 Battery Powered Devices

Digital Matter battery powered devices utilise a wide range of battery types and chemistries. For some battery types and devices, predicting battery life is difficult based on voltage alone. Lithium batteries typically maintain a relatively consistent voltage until they are nearly drained. Making low battery alerts difficult.

Most new (2018 onward) battery powered devices (Oyster2, Remora2, Falcon) are fitted with a battery Meter (coulomb counter) to track the energy used by the device. This allows for superior battery level estimates in comparison to using the measured voltage alone.

Methods for monitoring battery status for battery powered devices:

1. Bit 1 of the Digital Status Flags

   This flag is quite powerful as the device firmware considers the issues discussed above when setting or clearing this flag. These are discussed in the articles linked below.

   For example, devices with a coulomb counter will set this flag to ‘0’ when the measured percentage is below 15%.

   An alert can be set up when this flag is set to 0 across all devices if desired.
2. Analogue 1 displays the internal battery voltage. As discussed this is not highly accurate, but the battery capacity estimate articles below can be used to set rough alerts.
3. Analogue 6 for devices with a battery meter (Oyster2, Remora2, Falcon, Eagle) is the battery percentage.

   Simply set alerts based on your desired remaining %.

For more information see Battery Life, Battery Capacity Estimate and Low Battery Flag

10 RECOVERY MODE

Digital Matter devices are designed to intelligently vary their upload rate. For battery powered devices this helps conserve battery life, and for powered devices ensures that data use is kept low, whilst enough detail is provided for accurate tracking.

In the unfortunate event that an asset is stolen, devices can be set to “Recovery Mode” also, which is designed to set the device to increase its update rate, and live track so the asset can be recovered.

In recovery mode, the update rate will increase, with the key difference being while in trip, battery powered device will attempt to remain connected to the server. Normally they will open a connection to upload data, then turn off their modem.

Recovery mode can be set via OEM Server or via ASYNC message as per the integration documents.

Bit 7 of the Digital Input field’s status flags will reflect the current recovery mode status. Additionally a Log Reasons 36 and 37 indicate recovery mode being turned on or off.

11 HIGH-G EVENT DETECTION

See HERE for information about how to configure High-G Event detection. This is available on the Oyster2, Remora2, Yabby GPS, Yabby Wi-Fi and Falcon

Records will come through as described in the article except analogues 18, 19 and 20 will not be populated with High-G data, that data can be found in Field 24.

- Log Reason 46 – High-G Event
- Field 0 (GPS) – Location of Event
- Field 2 (Digital Data)
- Field 6 (Analogue Data)
- Field 24 (High-G)

12 RUN HOURS AND ODOMETER – ACCUMULATION ON DEVICE

This is discussed in 5 - Detecting Towing Trips.

As mentioned devices should be set up such that run hours and odometer values can be calculated based on points where DI 0 (ignition) is active.

To re-iterate this, in most cases, software platforms are required to calculate the run hours (time with ignition on) and odometer (distance travelled with ignition on) manually by summing the time and distance between points.

However some DM powered devices have an on-board run hour and odometer feature. Currently the Dart2, G100, Bolt and G62 have this feature. Setup and operation is covered here:

Odometer and Run Hours for 3rd Party Servers

The device will send ODO data optionally at the end of each trip, or each data point, in fields 26 and 27. In general sending at the end of trip is preferred as it uses less data and generally live, mid trip odometer readings aren't required.

13 IDLE MONITORING

There are multiple ways to detect idling for devices that use a wired ignition or run detect.

Either:

- On the server, when the speed is 0, and the ignition input (DI 0) is active. There needs to be some time threshold to prevent waiting at traffic lights triggering the idle condition.
- On Device

DM devices can set a digital input active to indicate idling is occurring – set up and operation here:

- Set up Idle Monitoring (general case/example)
- Idle Monitoring on the Bolt (example using Run Detect rather than physical ignition)

14 THE GPS DATA FIELD

14.1 GPS Fix Age

This article addresses 3 different times often seen in the DM protocol. The difference between the time in the GPS Data Field and the record header describes the age of the fix relative to when the record was logged.

14.2 GPS Field Units

Some of the units used in the GPS Data field may seem arbitrary, but they generally match what the device gets from the GNSS engine. They are space efficient, aiming to use the bytes as efficiently as possible, without making the decoding too onerous. Here are some tips on how to interpret them.

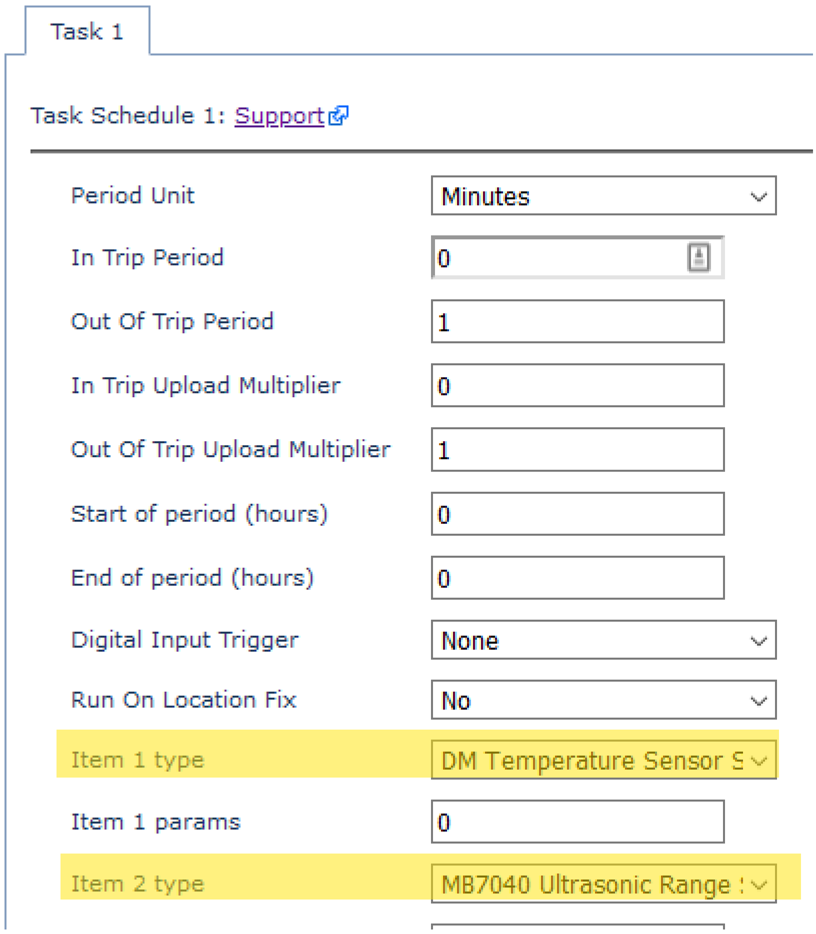
- Latitude/Longitude: simply divide by 10’000’000 to get a recognisable position in degrees.
- 2D ground speed: sent in cm/s. If you capture data in the OEM Server, it is shown in cm/s. Eg. Spd:1572 means 56.592km/h. To convert cm/s -> km/h, X * 3600 / 100’000.
- PDOP is sent x10. If you capture data in the OEM Server, it is shown in PDOPx10. Eg. PDOP:11 means a PDOP of 1.1

15 EAGLE AND FALCON TASK UPLOADS

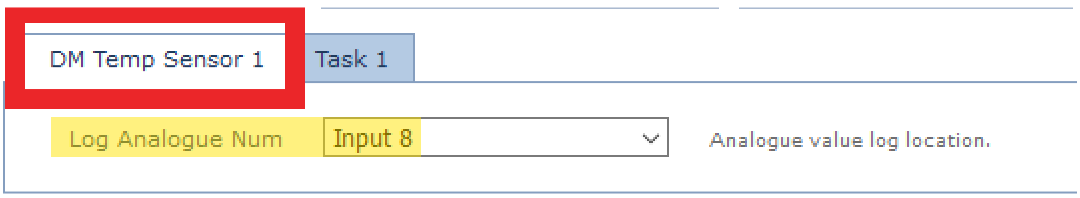
15.1 Where are my sensor readings reported?

Sensor readings will be reported in the Analogue Data Fields – FID 6 and FID 7.

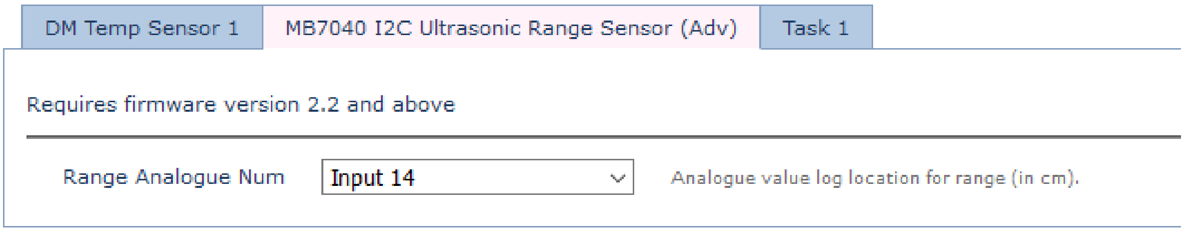
Most sensor types will be by default mapped to a particular analogue but may be remapped in system parameters. For example if we configured readings from these 2 sensors: a DM Temp Probe and MB7040 ultrasonic sensor:



We can expect the temp reading to be in Analogue 8



And the MB7040 sensor in analogue 14

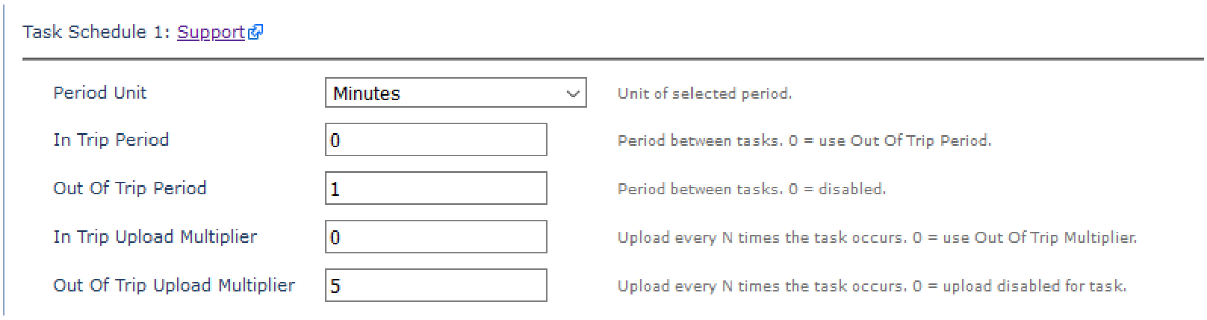


We can find these by adding the relevant parameter tabs.

15.2 Falcon and Eagle Measurement Uploads/Log Reasons

When configuring tasks, we have the option to define up to 2 tasks, each with an in trip/out of trip **period**, and **upload** multiplier. This is explained in further detail HERE

Each **measurement** will generate a log with log reason 25, and each **upload** will generate a log with log reason 25. E.g. consider these settings:



Every 5 minutes, we would get a batch of records (say beginning from 1000)

1001: Log Reason 25 + Data Fields

1002: Log Reason 25 + Data Fields

1003: Log Reason 25 + Data Fields

1004: Log Reason 25 + Data Fields

1005: Log Reason 25 + Data Fields

1005: Log Reason 11 (Heartbeat) + Data Fields.

The reason this happens is that to complete the upload, the device runs a standard heartbeat routine.

If we had the Upload Multiplier set to 1, we upload every measurement, so each time we would get 2 logs. Log Reason 25 + Log Reason 11. There is no way to eliminate either of these logs.

16 TCP INTEGRATION SPECIFIC NOTES

This section describes some of the common pitfalls and misunderstandings encountered when doing a TCP integration.

16.1 One socket per device

The protocol opens one socket per device, even when the devices are communicating via the OEM Server. The device will start with the Hello message, to identify itself. Thereafter, data on that socket applies to that device.

16.2 Socket Closure Request

When setup in Direct to 3rd Party mode, some devices may send the socket closure request. The server should respond by simply and quickly closing that socket. This is a work around for slow socket closures from the device side on some networks. It is implemented to save battery power.

16.3 Multiple records per upload

Devices may pack many data records into one data record upload message.

For example, the Data Record Upload header may specify a payload data length of 183 bytes. Looking at the first record header, the record length may be 61 bytes (which covers the header, seq number, RTC date time, log reason, and various fields). There are another 122 bytes to process in this upload. There will be another record header specifying a length, of perhaps 61 bytes, covering a full record. This leaves a final 61 bytes, which is another full record.

This example illustrates how 3 records could be sent in 1 Data Record Upload message.

16.4 Multiple Data Record Uploads before a Commit

A device may send many Data Record Upload messages (which may contain many separate records) before sending a Commit Request. The commit request is sent on:

- All records have been sent
- Or, a backlog of 100 records have been uploaded, and an intermediate commit request is being sent, and must be responded to with a Commit Response, before more records will be uploaded.

Most DM devices will send a maximum of 100 records before committing. This may contain debug record, which may be filtered out by the OEM Server, so 100 records may not reach the 3rd party.

16.5 How big should my receive buffers be?

The devices will generally not send more than 2048 bytes at a time in 1 message. However, many of these 2048 byte messages will be shoved into the TCP socket in quick succession, and may be delivered to the 3rd party in rapid succession. TCP will handle the flow control, meaning that the server should not have to deal with more than 2048 bytes at a time. We generally suggest making your buffers at least 4096 bytes in size.

17 APPENDIX A – LOG REASON TABLE

The table below lists log reasons that may be sent by each device.

Many have been deprecated or are used by older devices not in this table.

Log Reason	Description	Bolt	Dart2	Falcon	G62	Oyster2	Remora2	Yabby GPS	Yabby Wi-Fi
0	Reserved								
1	Start of trip	✓	✓	✓	✓	✓	✓	✓	
2	End of trip	✓	✓	✓	✓	✓	✓	✓	
3	Elapsed time	✓	✓	✓	✓	✓	✓	✓	
4	Speed change	✓	✓	✓	✓				
5	Heading change	✓	✓	✓	✓				
6	Distance travelled	✓	✓	✓	✓	✓	✓	✓	
7	Maximum speed (not used)								
8	Stationary	✓	✓	✓	✓				
9	Digital Input Changed	✓	✓	✓	✓		✓		
10	Digital Output Changed		✓	✓	✓				
11	Heartbeat	✓	✓	✓	✓	✓	✓	✓	✓
12	Harsh Brake	✓	✓		✓				
13	Harsh Acceleration	✓	✓		✓				
14	Harsh Cornering	✓	✓		✓				
15	External Power Change	✓	✓		✓				
16	System Power Monitoring		✓		✓				
17	Driver ID Tag Read		✓		✓				
18	Over speed								
19	Fuel sensor record								
20	Towing Alert (not used)								
21	Debug	✓	✓	✓	✓	✓	✓	✓	✓
22	SDI-12 sensor data								
23	Accident	✓	✓		✓				
24	Accident Data	✓	✓		✓				
25	Sensor value elapsed time			✓			✓		

Log Reason	Description	Bolt	Dart2	Falcon	G62	Oyster2	Remora2	Yabby GPS	Yabby Wi-Fi
26	Sensor value change			✓					
27	Sensor alarm			✓					
28	Rain Gauge Tipped								
29	Tamper Alert								
30	BLOB notification (not used)								
31	Time and Attendance								
32	Trip Restart								
33	Tag Gained (not used)								
34	Tag Update (not used)								
35	Tag Lost (not used)								
36	Recovery Mode On	✓	✓	✓	✓	✓	✓	✓	✓
37	Recovery Mode Off	✓	✓	✓	✓	✓	✓	✓	✓
38	Immobiliser On		✓	✓	✓				
39	Immobiliser Off		✓	✓	✓				
40	Garmin FMI Stop Response								
41	Lone Worker Alarm								
42	Device Counters	✓	✓	✓	✓	✓	✓	✓	✓
43	Connected Device Data								
44	Entered Geo-Fence		✓		✓	✓	✓	✓	
45	Exited Geo-Fence		✓		✓	✓	✓	✓	
46	High-G Event			✓		✓	✓	✓	✓
47	Third party data record (no reason)								
48	Duress								
49	Cell Tower Connection								
50	Bluetooth Tag Data (update, found, lost reason covered in tag data)						✓		

18 APPENDIX B – DATA FIELDS PER DEVICE

FIELD ID	Description	Bolt	Dart2	Falcon	G62	Oyster2	Remora2	Yabby GPS	Yabby Wi-Fi
0	GPS	✓	✓	✓	✓	✓	✓	✓	
1	Debug Event	✓	✓	✓	✓	✓	✓	✓	
2	Digital Data	✓	✓	✓	✓	✓	✓	✓	
3	Driver ID		✓		✓				
4	SDI-12 Device Identification								
5	SDI-12 Measurement								
6	IN16 Analogue Data	✓	✓	✓	✓	✓	✓	✓	
7	INT32 Analogue Data	✓	✓	✓	✓	✓	✓	✓	
8	BLOB								
9	Device 3rd Party ASYNC								
10	Project Code								
11	Trip Type Code								
12	Console Data								
13	RF								
14	RF								
15	Device Trip Type and Data	✓	✓	✓	✓	✓	✓	✓	
16	Garmin FMI Stop								
17	Accident Data	✓	✓		✓				
18	Acc Trace Header								
19	Acc Trace Samples								
20	V5 RF Message								
21	Profiling Counters			✓		✓	✓	✓	✓
22	Hand Held Radio GPS								
23	Deprecated								
24	High G Event			✓		✓	✓	✓	✓
25	Wi-Fi Scan			✓					✓
26	Trip Odo, Run Hrs	✓	✓		✓				
27	Device ODO, Run Hrs	✓	✓		✓				
28	Cell Scan			✓					✓
29	BLE List						✓		
30	BLE Individual Data						✓		