# Transmission Structure & Format

# ETM “9140/Delta” Class of IoT Cellular Edge Devices

ETM manufacture a range of cellular enabled edge devices used to monitor and log analogue, digital and pulse data from connected I/O.

The following document describes the procedure for interpreting data transmissions sent back by ETM devices over the cellular data network, to a remote server. An intended audience is 3rd party software platform integrators.

It provides a summary of transmission structure as well as relevant network, time and storage information.

This document applies to the following device models:

* **ETM 9140 + I/O Board** 1st Gen
* **Delta Black** DIN Mount Industrial Package
* **Delta Blue** Outdoor IP-Rated Battery PoweredPackage

## Data String Syntax Overview

Delta class products feature an internal cellular module which provides a WWAN connection to two IP address groups.

There are several popular data string formats that are used to push data to a server, from the different services running on the device. These are collectively referred to in ETM syntax as #(Hash) strings. Each is assigned a unique numeric identifier as well as a local network port.

Listed below are some popular data string formats, though there are several others.

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | **Function** | **TCP Port Options** | **UDP Port Options** |
| #1 | Telemetry Data | 13 | - |
| #9 | Network alarms | 2200 | 2200 |
| #21 | Logged data with raw  analogue signals in mV | 2150 | 2150 |
| #22 | Logged data with analogue signals convert to eng/sci units | 2150 | 2150 |

Data strings are sent as a raw text transmission over a TCP/IP socket connection, from a local port on the device IP, to a nominated server IP address and port number.

UDP is also supported. These settings can be customised using the configuration tool.

Typically, ETM products connect to our own hosted IoT dashboard, EWO (ETM Web Office) which utilitses the transmission formats listed above.

The rest of this document focuses on the **#21** transmission format, which is the most commonly used. Future revisions to this document will cover additional formats, including: **#1, #3, #5, #9 & #22**, eventually building a complete index.

# #21 Transmission Format

## Purpose:

The #21 type string is used for transmission of logged data records from device memory to remote servers, typically on a fixed interval. Analogue values are unscaled, presented simply in millivolts.

## Data String Example:

Below is an unpacking of a typical #21 transmission from an ETM field device, configured with the name “TestRTUDevice”. The transmission header is underlined for clarity. The lines that follow are the most recent log entries, to be delivered to the server in this transmission.

**#21,1,TestRTUDevice,1,175544,7,14,**

**1174100,0,150,50,1307,1175,1,2498,**

**1174200,0,160,52,1315,1193,1,2498,**

**1174300,0,168,51,1314,1190,1,2498,**

**1174400,0,170,55,1310,1176,1,2498,**

**1174500,0,174,54,1316,1194,1,2498,**

**1174600,1,181,53,1313,1182,1,2498,**

**1174700,1,183,52,1307,1176,1,2498,**

**1174800,1,185,54,1320,1194,1,2498,**

**1174900,1,195,56,1311,1186,1,2498,**

**1175000,0,201,55,1320,1200,1,2498,**

**1175100,0,208,54,1315,1188,1,2498,**

**1175200,0,212,53,1319,1192,1,2498,**

**1175300,0,217,52,1311,1188,1,2498,**

**1175400,0,224,55,1303,1169,1,2498,**

## Header format description:

|  |  |  |
| --- | --- | --- |
| **Position** | **Example** | **Explanation** |
| ***1*** | **“#21”** | Identifies the transmission type. |
| ***2*** | **"1"** | Refers to the send event type, with “1” denoting a periodic transmission, which occurs at a regular fixed interval. |
| ***3*** | **“TestRTU**  **Device”** | User defined device identifier. |
| ***4*** | **“1”** | Represents the day counter for the send date, as counted from a reference date, stored in the device memory. Indexed from 1, where 1 refers to the reference date itself. Hence in this example, the transmission was sent on the first date of the modem’s operation. Your server will need to know the reference date used by the modem, to calculate an absolute send date for each #21 transmission. See date calculation section on next page, for more details. |
| ***5*** | **“175544”** | The send time, in the format: hhmmss |
| ***6*** | **“7”** | Defines the number of active I/O channels associated with each record/block in the payload. Channels can be enabled/disabled using the modem configuration tool. |
| ***7*** | **“14”** | A count of the total number of records/blocks included in this transmission payload. In this example, you will observe 14 consecutive rows/records that follow the header. |

## Payload format description:

The payload is comprised of chronologically sequential records/blocks that share an identical structure. In this example, you will observe 14 records/blocks, consisting of time stamps and 7 channel measurements. They are logged at 1 min intervals, between 5:41pm – 5:54pm.

Each record starts with a timestamp in the format: **Dhhmmss**, where D represents the day count from the reference date, as described earlier. The reference date is required to determine the absolute date for each entry, using the relative day counter. See date calculation section below, for more details.

Subsequent fields in each record represent the various channel levels/measurements, at the time of logging. The 1st channel is configured for digital/dry contact input, so it can record

**1** **or 0** values only

The 2nd channel is configured for aggregate pulse counting, up to a maximum value of 4294967295counts, after which an arithmetic overflow will occur. Assuming an average pulse input frequency of 20Hz, this allows for approx. 6 years and 9 months of continuous pulse monitoring. Max pulse input frequency 25Hz.

The 3rd channel is configured for pulse frequency measurement. The pulse frequency is specified in **pulses/min**, not in Hz. It is determined by the difference between the present pulse count and the last recorded pulse count, divided by the time interval in between, where:

(Current Pulse Count – Previous Pulse Count) / Logging Interval = X pulses/min

The 4th and 5th channels are configured as analogue inputs, with the measured values provided in millivolts, ranging between **0 – 2500mV**. This guarantees the flexibility to perform any required conversions/calculations on the server side, to derive the process/scientific units you are interested in.

The remaining 6th and 7th channels have been left disconnected/unused. Channel 6 is configured as a digital input and is permanently reading 1, whereas channel 7 is configured as an analogue input and is permanently reading close to 2500mV. This is expected behaviour for disconnected channels, due to internal pullup characteristics of the device input pins. In this example, 7 sets of channel readings are logged and reported back to the server at each interval, despite the last 2 channels being left unused/redundant. It is recommended to disable unused channels, in order to reduce the length of the payload, conserving network data use. This change can be made using the modem configuration tool.

The overall length of the #21 transmission string depends on 3 variables: the number of channels recorded per log entry, the logging interval and the transmission interval. As you increase the logging frequency and decrease the transmission frequency, you can expect the overall size/length of the #21 string to increase. There will be a greater number of new records/blocks waiting in memory, to be sent to the server at each transmission time. Enabling additional channels also increases the length of the string.

## Date calculation

To calculate an absolute date for a particular transmission or log entry, you must use a date offset calculation function that can apply the day counter against the reference date. Any reference date can be programmed into the modem, using the configuration tool. **Default reference date:** **01/01/2005**.Alternatively, the modem can periodically transmit its reference date to the server as part of the separate #1 telemetry string, not described here. Max value for the day counter is 32767, permitting approx. **89 years of continuous operation** before overflow.

## Memory & logging

The modem will keep track of which records in its memory bank have already been sent back to the server. Each #21 transmission will only contain new records/blocks, written to the device memory since the last transmission time. In other words, the transmission of logged data is differential; the device will not retransmit the entire contents of its memory log each time it connects to the server. It is up to you to recombine/reassemble this data on the server-side to create a chronologically complete log file. Note that by default, the modem logging memory is circular and will eventually be overwritten, once it has become full.