

Enfora MT product range – AT Command Set– A simple overview.

This simple overview is in no way meant to be used as a substitute for the extensive & comprehensive Enfora 'AT Command Reference Set' issued for their AVL products. It's meant more as a quick reference for the beginner using the AT commands to compose scripts to control the Enfora range of products, specifically GSM/GPS 'Integrated platforms'.

The Enfora published 'Quick Start or User Guides' are the place we recommend all new integrators begin. Below, however, are a few handy AT commands commonly used:

- AT&F - Sets all TA (Terminal Adaptor, i.e. MT-GL/Mini etc) parameters back to factory defaults. Note: See the end of each AT Command reference set to define which parameters are not affected (AT\$FRIEND etc).
- AT&W – Saves the current scripted configuration to non volatile memory.
- AT\$MDMID – Sets unit ID, if left the ID will remain the unit's unique IMEI number
- AT\$EVDEL – Deletes a named event group.
- AT\$EVDELA – Deletes all Events
- AT\$EVENT – Lists all Events currently in memory.

For a data via UDP (GPRS):

- AT\$Friend - Remote servers IP address
- AT\$UDPAPI – Remote servers Port Number
- AT+CGDCONT - GPRS Network APN name
- AT%CGPCO – GPRS APN Username & Password (If required)
- AT\$AREG – 'Normally' =2 for automatic GPRS registration.
- AT\$HOSTIF – 'Normally'=0, in conjunction with AT\$AREG allows dial up connection

Note: If you require to change GPRS APN details when the MT is powered with SIM then issuing an AT\$AREG=0 will 'de-register' the modem (as will AT+CFUN=0) to allow you to change these details. Useful at the beginning of any script to allow ease of loading....just remember to set back to 2 for full registration either in HyperTerminal or via your script (usually one of the last commands)

For data via SMS/email:

- AT\$SMSDA – SMS Destination address; required by the MT to send and receive SMS.

Simple script writing

Scripts are simply lines of grouped AT Commands that achieve the desired functionality. Generally this will mean an Input Event and an Output Event for example:

AT\$EVENT=7,1,12,1,1

7 – Event Number, identifies Events in the same group

1 – Event Type, basically the input trigger, Transition (0), Occurrence (1), Input/AND (2) or Output (3)

12 –Event Category, In this case input Event category 12 (Timer1)

1 – Param1, as defined in the Event category, in this case must be 1

1 – Param2, as defined in the Event category, in this case must be 1

AT\$EVENT=7,3,40,2,8392710

7 - Event group, 3 - Event Type 'Output', 40 – Output Event (Generate and Transmit 1 UDP message to first IP address listed in \$FRIEND & Port number listed in \$UDPAPI, command based on Param1 & 2 values), 2 - Param1, 8392710 - Param2

Param1 - simply the message identifier so '2' in this case

Param2 - is a decimal representation of the Param2 bit field that defines what is sent in the message, so 8392710 = 100000000001000000000110 starting on the right Bit0 =0 (send all data as ASCII). The only bits set here are Bits 1,2,12 & 23.

Bit1 - Add Param1 to outgoing message (2 in this case)

Bit2 - Add \$MDMID to outgoing message

Bit12 – Add NMEA \$GPRMC Message

Bit23 – Add battery level.

Note: Bits may or may not be applicable to all the MT family, here; for instance, Bit23 is only applicable to Mini MT. The usage of a Bit field means the manufacturer can add functionality but retain the overall structure of the Event engine.....

So, to define the output message you requires, simply chose the bits you requires, and then convert the bit string from binary to decimal.

Summary.

With AT\$EVTIM1=60 the above script will send a UDP message every 60 seconds as Timer 1 expires. Timer1 is the input Event, the output Event being message send.

Note: With reference to the Mini MT, a 'wake event' must have occurred to allow this script to run. Externally powering the unit will achieve this as will scripting its motion sensor, as below.

More advanced scripting.

In the above a simple input Event (Timer1 expiring) causes an output Event to occur (send a message via UDP).

You'll soon find the need to add to this simple input/output structure. Take the above for instance; do we really want the MT to send every 60 seconds regardless? Unlikely, so ideally we'd like send a message every x seconds when the unit is in motion. The Mini MT has a motion sensor which we can make use of here. The MT-GL or uMT do not so motion will be define when velocity exceeds x MPH....so GPS velocity is used rather than motion or Ignition Input will be the trigger.

This is where the Event Type '2' (Input) becomes useful, so to make sure Mini only sends when in motion we can use Transition/Occurrence/Input Event 62 (Motion Status) as here:

```
AT$EVENT=7,2,62,1,1
```

So, with Timer1 expired and Motion Status as 'moving' a UDP message will be sent.

The motion sensor will require configuration in the script, this is well documented in the Mini MT AT Command Reference but here it is again.

(Wake up any time there is motion)

```
AT$WAKEENBL=4
```

(Stay awake for at least five minutes)

```
AT$WAKETIME=300
```

(Must detect no motion for 120 consecutive seconds before declaring unit stopped)

```
AT$MOTTRANS=120 (default)
```

In practice, the input Event trigger for the more AVL focused products with IO will be Ignition high as the main Trigger, with AND commands being velocity or GPS valid inputs.