Product Specification

Document Number	20070319
Title	MoTeC ECU CAN receive message format
Prepared By	Chris Nelson
Revision Date	**Change History**
19/03/2007	Initial Version

Introduction

This document describes the message format required when transmitting CAN data to a MoTeC Mxx0 ECU utilising V3.31D or later firmware. Up to a maximum of 12 channels can be transmitted to the ECU.

Specification

There are several requirements for sending data to the M800 via CAN, in a format it can handle. These are as follows:

- CAN bus speed of 1MBit/s
- Standard 11-bit identifier (this can be any number, but has to be entered in decimal format in ECU Manager when setting up)
- Transmission speed up to 50Hz
- Messages should be sent in Compound form, with the first byte the identifier
- Identifiers for the Compound messages need to be 00, 01, 02, 03
- Channels should be sent as 16-bit, maximum of 3 channels per compound ID
- Channel alignment should be 'normal' - i.e. High Byte first, Low Byte second
- The offset of the first channel should be 2, i.e. the channel starts in the third bit of the message
- The MSB (most significant bit) for each channel should be used to specify whether the value is negative or positive through the use of 2's complement numbering.

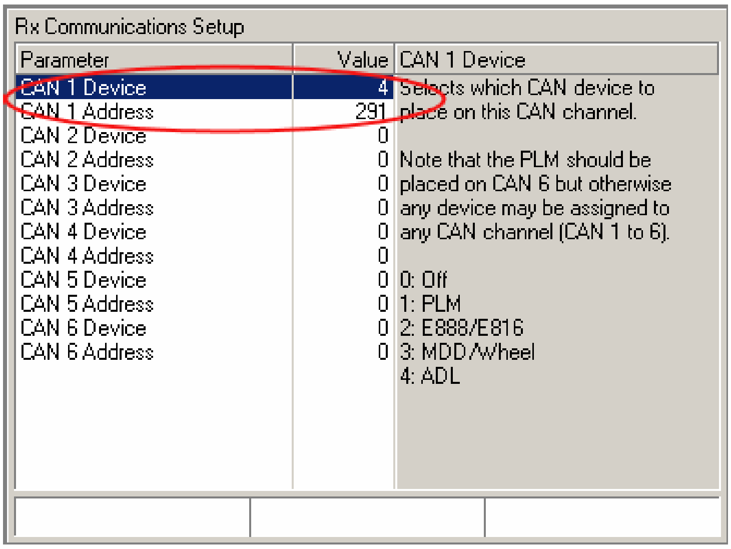
See below for a schematic of the message format required.

Byte							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Compound ID 00		Channel 1 (ADL1)		Channel 2 (ADL2)		Channel 3 (ADL3)	
Compound ID 01		Channel 4 (ADL4)		Channel 5 (ADL5)		Channel 6 (ADL6)	
Compound ID 02		Channel 7 (ADL7)		Channel 8 (ADL8)		Channel 9 (ADL9)	
Compound ID 03		Channel 10 (ADL10)		Channel 11 (ADL11)		Channel 12 (ADL12)	

See the following pages for an extract from a Technical Note explaining how to set up the reception of these channels in the ECU.

ECU Configuration

Go to 'Adjust -> General Setup -> Communications -> Receive Data Setup'



Rx Communications Setup

Parameter	Value
CAN 1 Device	4
CAN 1 Address	291
CAN 2 Device	0
CAN 2 Address	0
CAN 3 Device	0
CAN 3 Address	0
CAN 4 Device	0
CAN 4 Address	0
CAN 5 Device	0
CAN 5 Address	0
CAN 6 Device	0
CAN 6 Address	0

CAN 1 Device

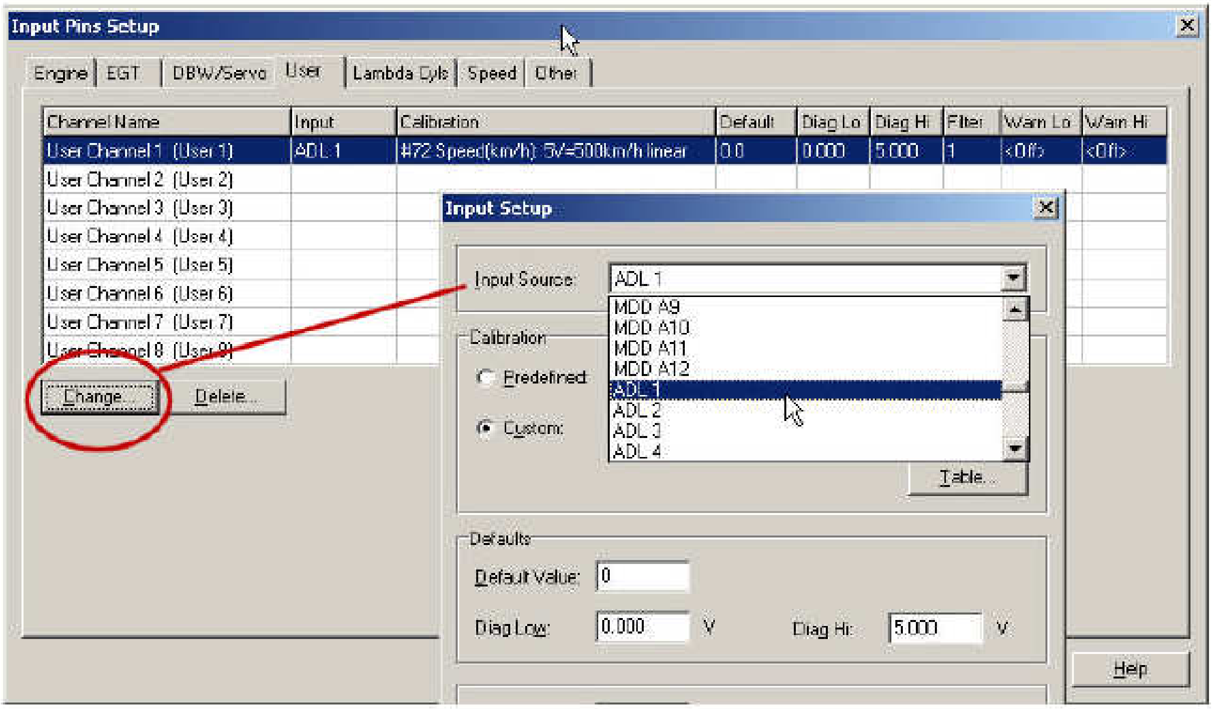
Selects which CAN device to place on this CAN channel.

Note that the PLM should be placed on CAN 6 but otherwise any device may be assigned to any CAN channel (CAN 1 to 6).

0: Off
1: PLM
2: E888/E816
3: MDD/Wheel
4: ADL

The CAN device is always 4 (ADL). The address will be the decimal value of the Base Address used in the Dash Manager setup. For this example, 291. (ie: 0x123 = 291 dec)

Select '*Adjust -> Sensor Setup -> Input Setup...*' to assign each ADL input source to a channel and calibrate it. Pick the channel that you wish to bring over and then click on '*Change*' to select the input source – the ADL in our case – then enter a calibration:



Channel Name	Input	Calibration	Default	Diag Lo	Diag Hi	Filter	Warn Lo	Warn Hi
User Channel 1 (User 1)	ADL 1	#72 Speed(km/h) 5V=500km/h linear	0.0	0.000	5.000	1	<Off>	<Off>
User Channel 2 (User 2)								
User Channel 3 (User 3)								
User Channel 4 (User 4)								
User Channel 5 (User 5)								
User Channel 6 (User 6)								
User Channel 7 (User 7)								
User Channel 8 (User 8)								

Input Source: ADL 1

Calibration

○ Predefined

● Custom:

Defaults

Default Value: 0

Diag Low: 0.000 V Diag Hi: 5.000 V

Here, User Channel 1 is setup to receive *ADL 1*, which is *Wheel Speed - Front* in our example.

Send the configuration to the ECU and/or reset the ECU. The *User 1* channel should now be displaying the value as received from the Dash.

Calibration

Choosing a calibration: most of the time you simply want a one-to-one linear calibration for channels received over CAN. In this example, the Base Resolution of ADL 1 (Wheel Speed - Front) was noted as being 0.1km/h in Dash Manager - in other words 1 decimal place. In practice this means a value, such as 100.0 km/h, is transmitted as 1000 (simply ignore the decimal point). We could then use a custom calibration to achieve the desired mapping (see table below).

An Input value of 1.000 (ECU Manager is incorrectly asking for a Voltage, just ignore the decimal point and treat 1.000 as 1000) corresponds to 100.0 km/h. Adding a second set of points gives us a straight-line that the ECU will interpolate/extrapolate along.