CAN Expansion Module V1

Document Number		PR035_CANExpansionModuleV1_12JAN2023	
Title		CAN Expansion Module V1	
Team Member / Engineer		Raheel Farouk	
Creation Date		12 JAN 2023	
Prepared By	**Date**	**Checked By**	**Change History**
Raheel Farouk	12JAN2023	-	Initial Version
Raheel Farouk	16JAN2023	-	Added Endianness and Cleaned up formatting
Raheel Farouk	06MAR2023	-	Removed slip angle sensor and added Aux2
Patrick Murphy	09MAR2023	-	Added Descriptions of data and Setup information

Purpose of Module

The purpose of the CAN Expansion Module is to convert desired analog sensor inputs into digital data on the CAN bus.

Setup Guide

Setting up the Expansion Module requires a CAN ID for the expansion module and the sensors to be plugged into their corresponding spots on the module. The MoTeC firmware will decide between what input will be used for each function, for example brake temperature will select the brake temperature input but you will be able to choose between front and rear, left and right.

Board Configuration

The board is currently based on the Canduino V3.1, an atmega328P with an attached MCP2515 and MCP2551 to allow communication onto a CAN network. More information on this can be found in the following locations.

https://www.tindie.com/products/massivegroup/canduino-v31-atmega328p-with-can-bus/

https://github.com/MassiveButDynamic/canduino-v3.1

Can Library Configuration

While the Canduino recommendation for Arduino libraries is the Arduino-mcp2515 library, the current library used is the mcp_can library which can be found in the following link. This is intended to have compatibility between the MCP2515 and the MCP25625 chips.

https://github.com/coryjfowler/MCP_CAN_lib

Analog Sensor Wiring

Sensor Name	PIN
Brake Temperature	ADC 0
DPOS	ADC 1
AUX 1	ADC 2
AUX 2	

Module Specifications

Messaging Structure

SPI Message structure

Endianness	Big Endian	
Message Length	8	
Byte	**Contents***	
Byte0	Brake Temp (9-2)	
Byte1	Brake Temp (1-0)	DPOS (9-4)
Byte2	DPOS (3-0)	Aux 1 (9-6)
Byte3	Aux 1 (5-0)	Aux 2 (9-8)
Byte4	Aux 2 (7-0)	
Byte5	Firmware Version	
Byte6	Compound ID	
Byte7	Counter	

*Note: The numbers in brackets are the bit numbers of the data stored in the byte. For example if Brake temp is a 10bit value, then the most significant first 8 bits of brake temp, Bits 9 to 2 will be stored in byte 1, then the remaining 2 bits will be stored in byte 1.

Can Message Structure of Data

Base CAN ID:	0xf0		
Data Name*	**Length (Bits)**	**Position in Message**	
		Start	**End**
Brake Temp	10	0	9
DPOS	10	10	19
Aux 1	10	20	29
Aux 2	10	30	39
Firmware Version	8	40	47
Compound	8	48	55
Counter	4	56	59

*Note: that the Data will effectively be sent as unsigned integers with values in the range of 0 to 2^(length of bits). Any conversion of this information currently needs to be performed on the receiving end of the data.

Additional Notes

Auxiliary sensor attached to wind speed or steering position.

Description of Firmware Version

The firmware version corresponds to the current version of the Module. This is represented in format the format of vXX.X. In the software this is programmed as a uint8_t type. To achieve the decimal position the firmware version must be 100 times the current version. For example if the current version of the module is v0.1 then the firmware version shall be 100*0.1 or 100.

Description of Compound

Compound is unimplemented. To be used if the length of data the module transmits exceeds 8 bytes of information to allow more data extraction from the same CAN ID.

Description of Counter

Counter is a 4bit value used to determine if loss of transmission occurs. If observing transmission from the board this value shall go from 0 to 15 and rollover, if any value is missed while this is occurring then a loss of transmission has occurred and should be investigated.