Vehicle Surface Pressure Sensor V1

Document Number		PR035_VehicleSurfacePressureSensor_31MAY2023	
Title		Vehicle-Surface-Pressure-Sensor-V1	
Team Member / Engineer		Patrick Murphy	
Creation Date		31 MAY 2023	
Firmware Version		1.0	
Prepared By	**Date**	**Checked By**	**Change History**
Patrick Murphy	31MAY2023	-	Initial Version

Purpose of System

The purpose of the Vehicle Surface Air Pressure Sensor (VSAPS) is to get multipoint surface air pressure on a wing from digital sensors and convert that to a usable number on the CAN bus.

There are two parts to the system, the Sensor board and the Interface board. Currently for any system there are 8 Sensor boards to 1 interface board max.

Sensor Board

The sensor board is based around the BMP581 pressure sensor with an SPI interface to connect to the board. This board requires external 3.3V power to work.

Interface Board

The Interface Board is based around the ATMEGA328P with 2 level shifters, one for Chip Select (CS) pins and one for SPI communications. The TXB0108 series shifter works for Chip Select, however for working implementation the shifter needs replaced with a different logic level converter. The interface board features a set of wires coming out of the wing that are for reprogramming the board aswell as debugging.

Setup Guide

Setting up the VSAPS 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 Wiring

The Chip Select Pins to enable the Level shifters and the CAN chip were wired to pins 19(ADC6) and 22(ADC7) of the 328p board, unfortunately these are not digital outputs, so pins 23 and 24 were used, which are the ADC0 and ADC1 pins respectively. This was done by manually soldering a wire from the ADC pin holes to the VIAs used for the other shifter then cutting out the MCU trace connecting to it.

The LVL-2 shifter was replaced using an externally driven Bi-Directional logic level converter – Bi-Directional, from spark fun. To do this all that was done was to splice the ICSP programming wires into the level shifter.

Board Configuration
To program the Interface Board simply plug in an ICSP programmer and set the device as an Arduino Nano. Debugging can be done using a serial programmer.

Can Library
The library used to communicate with the MCP2515 as recommended by the Canduino is the Arduino-mcp2515 library. Previously the MCP_CAN library by coryjfowler was used however issues were encountered relating to significant transmit errors on the bus so this is no longer used.

Messaging Structure

SPI Message structure

Endianness	Big Endian		
Message Length	8		
Byte	**Contents*,****		
Byte0	Counter (3-0)	Compound ID (2-0)	Pressure (23)
Byte1	Pressure A (22-15)		
Byte2	Pressure A (14-7)		
Byte3	Pressure A (6-5)	Pressure B (23-18)	
Byte4	Pressure B (17-10)		
Byte5	Pressure B (9-6)	Pressure C (23-21)	
Byte6	Pressure C (20-13)		
Byte7	Pressure C (12-5)		

*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.

** Note: Pressure A, B, and C are just meant to distinguish the data sets from each other.

Can Message Structure of Data

Base CAN ID:	0x760 ; +1 ; +2 ; +3		
Data Name*	**Length (Bits)**	**Position in Message**	
		Start	**End**
Counter	4	0	3
Compound ID	3	4	6
Pressure A	19	7	25
Pressure B	19	26	44
Pressure C	19	45	63

*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

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

Because we are trying to fit more than 64 bits of data using a singular CAN ID, we need to have distinguishing bit(s) to know what the data means. For example in for a device that only transmits CAN ID 0x760, if we are sending planning on sending 3 sets of data that are 24 bits long each, that's 72 bits total, larger than the CAN data length so we send the first two sets of 24 with an distinguishing number ie bit 0 =0 to say message 1 for data set 1 and 2, then bit 0=1 to say message 2 for data set 3. Refer to the CAN message excel document for more reference if needed.

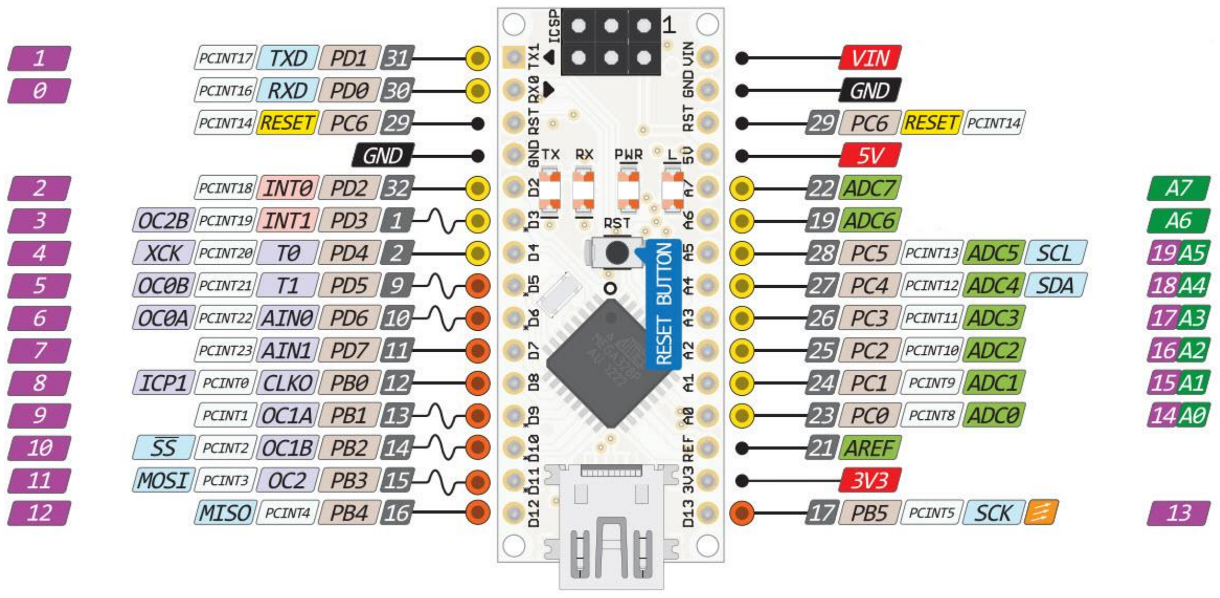
Description of Counter

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.

Arduino Nano Programming to MCU Pin diagram

The below diagram shows the Arduino Nano which is relevant to programming the VSAPS interface board and correlating the pinout of the MCU to I/O in programming.

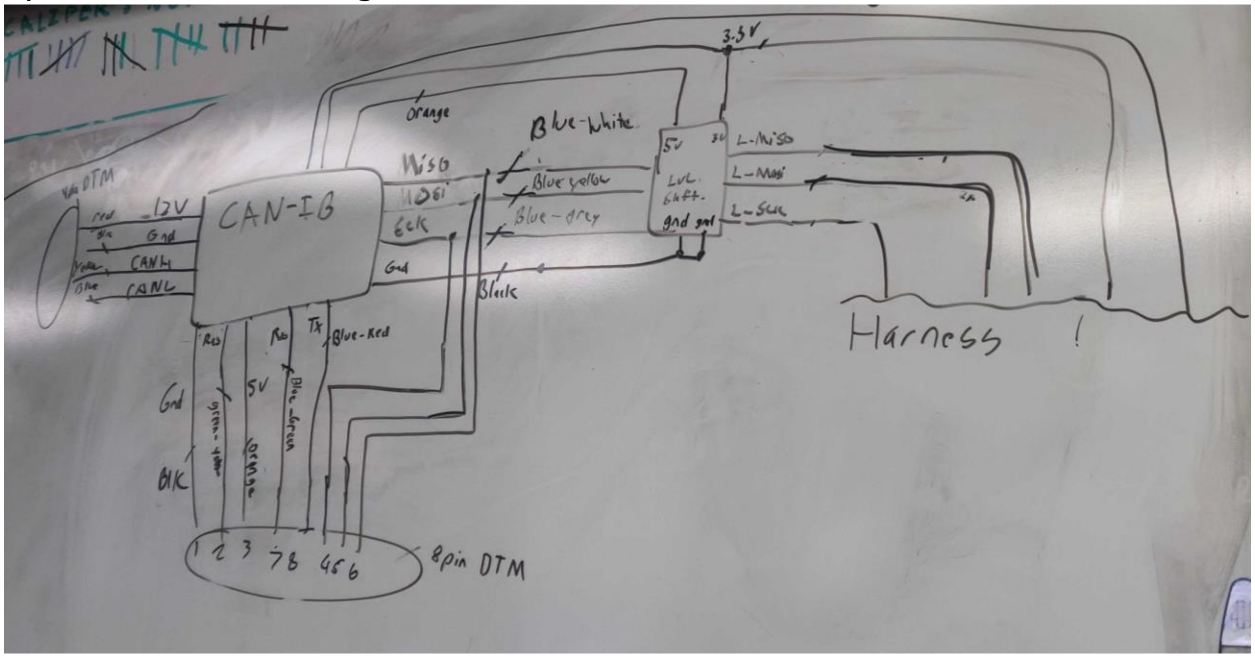
The numbers in Purple are the Arduino Nano pinout # to be used when programming, the inner numbers in grey are the MCU pinout, used for wiring the board. So for ADC5 it would be MCU pin 28, and Arduino pinout # 19. So in code it would look like " Digitalwrite(19,LOW);" to set the digital pin low.



ADC Pins

For the ATMEGA328P there are 7 ADC pins which are multiplexed to a single ADC at up to a 10bit accuracy, however ADC 6 and 7 are ADC only and not digital pins.

System Connection Diagram



DTM
red
12V
Blk
Gnd
Yellow
CANH
Blue
CANL
CAN-IB
Miso
MOSI
Sck
Gnd
Orange
Blue-White
Blue yellow
Blue-grey
Black
3.3V
5V
3V
Lvl. Shft.
gnd gnd
L-Miso
L-Mosi
L-Sck
Harness
!
Rx
Rx
Tx
Blue-Red
Gnd
Blk
green-yellow
5V
orange
Blue-Green
1 2 3 7 8 4 5 6
8pin DTM