INERTIAL NAVIGATION SYSTEM (INS)

Revision 0.0.1.41 (May 2023)

Firmware Version = 41

Brief:

This device is a multi-functional sensor that merges two main components:

- An Inertial Measurement Unit (IMU) that has a 3-axis accelerometer and a 3-axis gyroscope. The accelerometer measures changes in velocity along three perpendicular axes (x, y, and z), and the gyroscope detects changes in rotational position (pitch, roll, and yaw) along the same axes.
- A Global Navigation Satellite System (GNSS) receiver, which communicates with satellites to determine the sensor's location on Earth.

What makes this sensor unique is its onboard sensor fusion algorithm. This sophisticated software combines the different measurements from the IMU and the GNSS to produce very accurate and fast updates on the sensor's position, orientation, and movement.

The sensor fusion algorithm also corrects for biases (systematic errors) in the accelerometer and gyroscope measurements, ensuring even more accurate data.

Moreover, the device can calculate the velocities in the sensor's (or the vehicle's) own frame of reference, which can be extremely useful for understanding and controlling its movement.

Purpose:

The main objective with this sensor is to put cutting-edge, high-performance sensor fusion algorithms into the hands of motorsport integrators and motorsport engineers without having to deal with sensor calibration, embedded systems, or understanding the sensitivity of MEMS IMU’s, specifically.

Functional Overview:

The sensor fusion algorithm that runs on this sensor is an Extended Kalman Filter (EKF). An EKF is a filter that combines data from multiple sensors and provides a best guess estimate of the position, orientation, and velocity of the vehicle.

There are issues that become apparent when trying to use a GNSS by itself, or an IMU by itself to determine the position and orientation of a vehicle. We will highlight the issues of each, below:

GNSS / GPS:

The Global Navigation Satellite System uses satellites in low orbit of the planet with known positions to triangulate the position of the GNSS receiver. GNSS receivers can produce global position updates with high absolute accuracy (within the context of the size of the planet). The issue is that they cannot generate high-rate position updates due to how the satellite signals are transmitted from the satellite to the planet surface. There are also many environmental scenarios that can cause position errors or even loss of position altogether.

IMU:

Inertial Measurement Units use small force measurement sensors to measure acceleration and angular velocity. IMU's can produce relative movement measurements at a very high rate. The issue is that the measurements are subject to propagation drift very quickly.

GNSS + IMU:

We now know that GNSS receivers will provide high absolute accuracy but have slow update rates. We also know that IMU's will have very low absolute accuracy over time but have a very high update rate. An EKF will combine these two and have them complement each other to provide high absolute accuracy and a high update rate.

GENERAL USAGE / INSTALLATION:

COORDINATE FRAME:

This sensor uses a "right hand rule" coordinate frame. You must place the sensor with the X axis pointing forward. This will place the sensor in a Forward Right Down frame (FRD). This means that the sensor will produce data where:

Axis	Direction	Gyroscope Axis	Acceleration Axis
X+	Forward	Roll Rate	Longitudinal Accel
Y+	Right	Pitch Rate	Lateral Accel
Z+	Down	Yaw Rate	Vertical Accel

GENERAL INSTALLATION NOTES:

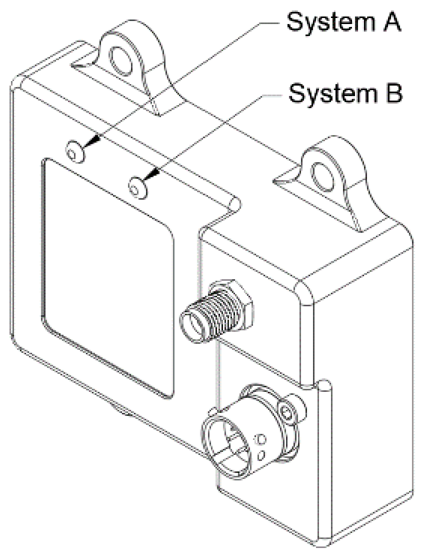
- **It is imperative that the INS must not be installed on a CAN bus with active traffic on the 0x006 CAN ID.**
- It is of the **utmost importance** to isolate the sensor from chassis vibrations to the extent that it does not introduce error into the orientation measurements.
    - An example of poor placement would be placing the sensor in the middle of a floor pan without any structural bracing beneath it. The floor pan, in this case, is likely to act like a spring causing the accelerometer to saturate and produce poor results. A more logical place to put the sensor is mounted on a rigid part of the chassis (I.E., the roll cage, or mounted to a structural member that is rigid).
- It is **highly recommended** to keep the sensor away from sources of excess heat. While the internal accelerometer and gyroscope are calibrated over a broad temperature range (-40 to 80 C), it will help the stability of the estimates if the temperature is relatively constant around 25-30 C.
- It is **highly recommended** to specify at the time of order the 3D translation of the GNSS antenna to the sensor body. This will help make the data more accurate. By default, the sensor is programmed to have

the GNSS antenna mounted 1 meter above the sensor body with no lateral translation (X / Y). This 3D translation is now configurable via CAN as of firmware version 8 (See CAN Definitions (Input)).

- To produce accurate measurements the sensor will need an initialization period after start-up to generate accurate yaw estimation and body frame velocity measurements. This generally will occur after a short period of driving (in practice, 1 warm up lap is generally adequate). If faster convergence time is required, the user can disable the AHRS system (See CAN Definitions (Input)).
- **If only forward velocity is the desired output from the sensor, this does not require initialization as it is calculated from the magnitude of the X and Y velocity measurements.**
- **If the user wishes to have the output data in a different reference frame, a hardcoded transformation can be written in firmware specific to the customer's needs.** Please contact us for more information.

GENERAL USAGE NOTES:

There are two LEDs on the outside of the sensor that can help indicate the status of the sensor. If the sensor is powered, the following states will indicate how the sensor is operating.



The **System A** LED is defined by the following states:

- Red indicates that the system is online and publishing data via CAN.
- Blue indicates that the system has an adequate GNSS fix and that the system is initializing.
- Green indicates that the system has a good GNSS fix and that the Kalman filter has converged on a valid solution and that the sensor is publishing accurate pose estimates and slip angle.

If the sensor is powered and there is no illumination, this will require support from Obsidian Group. Please email for support sander@obsidianeng.com.

The **System B** LED is defined by the following states:

- Blue indicates that the sensor has adequate power and that the on board electronics are OK.
- Purple indicates that there is an issue with the CAN bus that the sensor is connected to.
  - Incorrect CAN termination resistance

    - Incorrect CAN wiring
    - Incorrect CAN baud rate
- Cyan flashing at 1hz corresponds to the 1PPS signal from the internal GNSS receiver.

If the sensor is powered and there is no illumination, this will require support from Obsidian Group. Please email for support sander@obsidianeng.com.

INS SPECIFICATIONS:

Note: All sensors have individual thermal calibration between -40 - 80 C

Note: All sensors have internal calibrations for bias, scale factor, and misalignment

Accelerometer:

Range	+/- 16 G
Bias Stability	< 0.04 mg
Linearity	< 0.5 % FS
Bandwidth	260 Hz
Cross Axis Sensitivity	+/- 0.05 deg

Gyroscope:

Range	2000 deg/sec
Bias Stability	< 10 deg / hour
Linearity	< 0.1 % FS
Bandwidth	256 Hz
Cross Axis Sensitivity	< 0.05 deg

GNSS Receiver:

Type	72 Channel, Single Frequency
Constellations	GPS, GLONASS, Galileo, BeiDou, QZSS, SBAS
TTF (Cold / Hot (typical))	< 60 Seconds / 1 Second
ITAR Limit	Altitude 50,000 m and Speed of 500 m/s

Sensor Fusion Calculation:

Rate	400 Hz (Position, Velocity, Orientation, Acceleration, Gyroscope) 100 Hz (MoTeC GPS Simulation 0x680, 0x681, 0x682, 0x683) 50 Hz (Emtron Pty Ltd GPS Simulation 0x28A, 0x28B, 0x28C)
CAN	1Mbps (Default) (500kbps and 250kbps optional)

Electrical:

Input Voltage	5V (Min) / 24V (Max)
Power Consumption	75 mA @ 12V (0.84W)

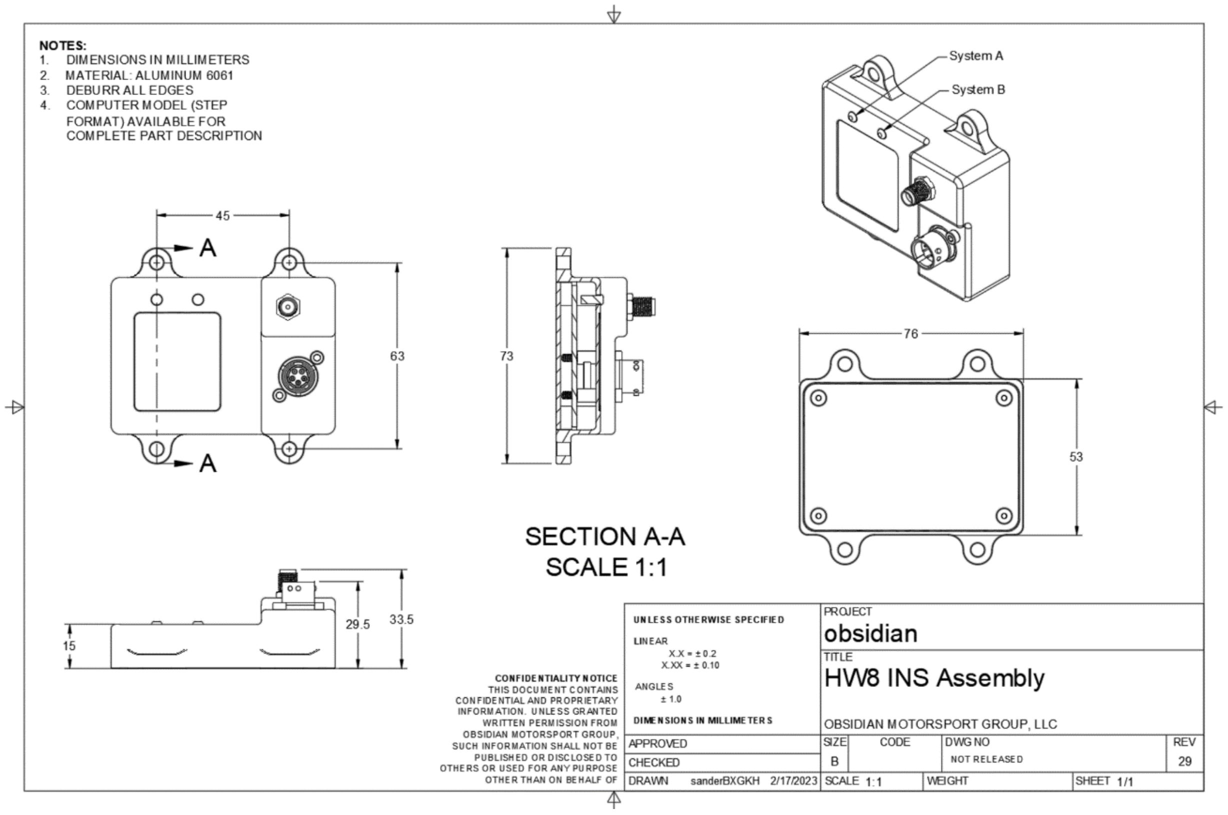
Electrical Pinout:

Mating Connector	Deutsch Autosport ASL606-05SN
Pin 1	Ground

Pin 2	n/c
Pin 3	Power
Pin 4	CAN L
Pin 5	CAN H

Mechanical:

Dimensions (L x W x H)	76 x 70 x 33 (mm)
Material	6061 T6 Anodized Aluminum
Weight	


NOTES:
1. DIMENSIONS IN MILLIMETERS
2. MATERIAL: ALUMINUM 6061
3. DEBURR ALL EDGES
4. COMPUTER MODEL (STEP FORMAT) AVAILABLE FOR COMPLETE PART DESCRIPTION
System A
System B
45
A
A
63
73
76
53
SECTION A-A
SCALE 1:1
15
29.5
33.5
CONFIDENTIALITY NOTICE THIS DOCUMENT CONTAINS CONFIDENTIAL AND PROPRIETARY INFORMATION. UNLESS GRANTED WRITTEN PERMISSION FROM OBSIDIAN MOTORSPORT GROUP, SUCH INFORMATION SHALL NOT BE PUBLISHED OR DISCLOSED TO OTHERS OR USED FOR ANY PURPOSE OTHER THAN ON BEHALF OF
UNLESS OTHERWISE SPECIFIED
LINEAR
X.X = ± 0.2
X.XX = ± 0.10
ANGLES
± 1.0
DIMENSIONS IN MILLIMETERS
APPROVED
CHECKED
DRAWN sanderBXGKH 2/17/2023
PROJECT
obsidian
TITLE
HW8 INS Assembly
OBSIDIAN MOTORSPORT GROUP, LLC
SIZE B
CODE
DWG NO NOT RELEASED
REV 29
SCALE 1:1
WEIGHT
SHEET 1/1

CHANNEL DESCRIPTIONS:

This sensor produces data in 10 different categories:

- Orientation
- Acceleration (Accelerometer)
- Body Frame Acceleration (Gravity Compensated Accelerometer)
- Angular Velocities (Gyroscope)
- Body Frame Velocities
- Global Position
- Slip Angle
- Global Time (UTC)
- Race Time and Race Distance
- State Enumerations

ORIENTATION:

Orientation refers to the angles of rotation about the three principal axes of the system. The three angles are typically referred to as roll, pitch, and yaw:

Roll (rotation around the X-axis): This is the tilt to the left or right (wing up or down in the case of an aircraft).

Pitch (rotation around the Y-axis): This is the tilt forwards or backwards (nose up or down in the case of an aircraft).

Yaw (rotation around the Z-axis): This is the rotation to the left or right (nose left or right in the case of an aircraft), analogous to turning in a circle left or right.

These angles are with respect to an Earth-fixed frame. This essentially means that a roll and pitch measurement of 0 degrees will indicate that the sensor is exactly perpendicular to gravity, while a yaw measurement of 0 means that the sensor is pointing exactly north.

RAW ACCELERATION:

Accelerometers measure the rate of change of velocity along the three axes of the body frame (x, y, and z). These measurements are made in response to both the movement of the vehicle **and the force of gravity**.

Raw acceleration measurements from the INS include both these components:

- The actual acceleration of the vehicle as a result of propulsion, braking, or changes in direction.
- The acceleration as a result of gravity, which always points downwards to the center of the Earth.

Raw acceleration is, in practice, generally not very useful for vehicle dynamics evaluation. Raw acceleration values are only provided to the user to provide a gravity reference if the user requires.

BODY FRAME ACCELERATION:

Body frame acceleration measurements represent the actual movement that the vehicle is experiencing **independent of orientation or gravity**.

To isolate the acceleration caused by the vehicle's movement, the gravity component needs to be removed from the raw acceleration measurements. The gravity component is not simply a constant 9.81 m/s^2 (1G) to be subtracted, because the direction of the gravity vector changes as the vehicle changes its pitch, roll, and yaw.

This correction is necessary because you typically want to know how the vehicle is moving through space, not how it is being affected by gravity. The gravity component is essentially a distraction that needs to be removed.

These measurements are, in practice, most useful for vehicle dynamics evaluation.

ANGULAR VELOCITY:

The angular velocity measurements represent the velocity at which the sensor is rotating around the three axes of the inertial frame / body frame.

Angular velocity is a vector quantity, meaning it has both magnitude (speed of rotation) and direction (axis of rotation). It is generally represented along three perpendicular axes – roll (X-axis), pitch (Y-axis), and yaw (Z-axis), corresponding to the three dimensions of space:

Gyro X or **Roll Rate (X-axis):** The angular velocity around the X-axis is the rate of change of the roll angle, which describes the vehicle's motion of tilting side to side.

Gyro Y or **Pitch Rate (Y-axis):** The angular velocity around the Y-axis is the rate of change of the pitch angle, which describes the vehicle's motion of tilting forward and backward.

Gyro Z or **Yaw Rate (Z-axis):** The angular velocity around the Z-axis is the rate of change of the yaw angle, which describes the vehicle's motion of turning left and right.

BODY FRAME VELOCITY:

The velocities in the body frame refer to the speed and direction of an object's movement, as it relates to its own local coordinate system.

The body frame is a reference frame that is fixed to the vehicle, and it moves and rotates with the vehicle. Its axes are defined as follows:

Velocity X (Forward): This axis points forward, out of the vehicle's nose.

Velocity Y (Rightward): This axis points to the right, out of the vehicle's right side.

Velocity Z (Downward): This axis points downward, out of the vehicle's bottom.

Forward Velocity (Magnitude of X and Y): This is an equivalent to a ground speed channel. This is calculated from the magnitude of INS Body Velocity X and INS Body Velocity Y. This is given by:

$$forward\_velocity[m/s] \ = \ sqrt(velocity\_x[m/s] \ \hat{} \ 2 \ + \ velocity\_y[m/s] \ \hat{} \ 2)$$

GLOBAL POSITION (LATITUDE, LONGITUDE, ALTITUDE)

The global position is with respect to a local tangent frame on the earth’s surface (using the WGS84 ellipsoid) and is represented as decimal degrees in Latitude and Longitude.

Altitude is represented as height above the WGS84 ellipsoid **and will vary from other GNSS receivers that represent altitude as height above the Mean Sea Level (MSL)**.

SLIP ANGLE

The slip angle measurement is the ratio of lateral body frame velocity (**Velocity Y**) to longitudinal body frame velocity (**Velocity X**). This can be represented as:

$$slip\_angle[rad] \ = \ arctan(velocity\_y[m/s] \ / \ velocity\_y[m/s])$$

Slip angle will only be published from the INS if the **Filter State** value is equal to 2 and longitudinal body frame velocity (vX) is greater than 0.1 m/s. Otherwise the slip angle value will be set to 0.0.

TIME AND DATE

The time measurements provided by the INS are representative of the time and date in UTC (Universal Time Coordinated). In practice, UTC lines up with the Greenwich Mean Time (GMT) zone.

The time and date will be accurate when the INS Time Valid value is equal to 1.

RACE TIME AND RACE DISTANCE

The INS will automatically start a timer and a velocity integrator that will provide distance traveled during a period of forward movement.

The distance channel is integrated from **Forward Velocity**.

Without any user intervention, the INS will start this time and distance calculation module as soon as the vehicle is traveling at a speed of 1 m/s. This module will reset as soon as the vehicle has decelerated to a speed of less than 0.8 m/s.

The user also has the option to manually initiate the timer and integration module using a CAN command which is outlined in the **0x40 Dead Reckoning Enable / Disable** section of the CAN Definitions (Input) chapter.

- The channel limits of INS Race Time are 655.35 seconds.
- The channel limits of INS Race Distance are 655.35 meters.

STATE ENUMERATIONS

The state enumerations are provided to inform the user about the quality of the estimates that the INS is providing.

- **Tracked Satellites** represents the number of satellites that are being used in the solution estimate.
    - In practice, this should be no less than 8 satellites when the antenna has good sky view.
- **GNSS Fix Type** represents the type of GNSS fix.
    - In practice, you will want to see a value of 4 in normal operating conditions.
        - **0 == No GNSS Fix**
        - **1 == Time Fix Only**
        - **2 == 2D Fix Only**
        - **3 == 3D Fix**
        - **4 == Differential Fix**
- **Filter State** represents the state of the main on-board Kalman filter that is generating the acceleration / velocity / position / orientation estimates.
    - In practice, you will want to make sure you see a value of 2 in normal operating conditions after the filter has converged*
        - **0 == Not Tracking** (normal at startup)
        - **1 == Aligning** (normal during the filter convergence period)
        - **2 == Tracking** (normal after filter has converged)
        - **3 == Loss of GNSS** (normal if there is a brief GNSS outage due to environmental conditions)
- **Baud Rate Enum** represents the current CAN baud rate of the sensor.
    - **1 == 1Mbps** (default)
    - **2 == 500Kbps**
    - **3 == 250Kbps**
- **Transmission Rate Enum** represents the current transmitted rate of the fastest transmitted channels in the highest priority channel group.
    - **1 == 400 Hz**
    - **2 == 200 Hz** (default)
    - **3 == 100 Hz**
    - **4 == 50 Hz**

- **5 == 25 Hz**
- **6 == 10 Hz**

*The INS must have a brief period of driving with a bit of straight road driving, left turns, and right turns to reach filter convergence. In practice, this is achieved in about half of a sighting lap on a normal racing circuit.

CAN DEFINITIONS (OUTPUT):

Note: This sensor should be connected to a 1Mbps CAN bus. Since the data rate and precision is so high, the data payload transmits roughly ~60% of the bus load of a 1Mbps bus by itself. Take note of this when integrating the sensor onto a bus with other high bandwidth devices.

Note: As of Version 15, the transmission rate and the CAN Baud rate are user selectable using CAN configuration commands. ***The data rate must be reduced from 400 Hz to AT LEAST 200Hz if the CAN Baud rate is moved from 1mbps to 500kbps.***

Note: The values, units, identifiers, baud rate, and update rates are configurable at time of order. Please contact sander@obsidianeng.com for more information.

CAN Standard	CAN 2.0A (11-bit identifiers)
Bit Rate	1Mbps (default)
Byte Order	Big-Endian (MoTeC “Normal”) **Note: All references in this document are Big Endian**
Termination Resistor	Internal hardware DIP switch on SN > 50

CAN Message Structures:

This table represents a byte-wise arrangement of the CAN messages that the INS produces.

- **(msb)** represents the “beginning” of the can message. This is also referred to as “most significant bit”.
- **(scalar <x>)** represents the scalar used to bring the message in to a floating-point representation.
- **(unit <x>)** represents the unit of the channel.

ID	**Byte 0**	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Default Rate (hz)**
0x790	yaw (msb, scalar 1e-2, unit deg)	yaw	pitch (msb, scalar 1e-2, unit deg)	pitch	roll (msb, scalar 1e-2, unit deg)	roll	version	serial number	200
0x791	gyro x (msb, scalar 1e-7, unit rad/sec)	gyro x	gyro x	gyro x	gyro y (msb, scalar 1e-7, unit rad/sec)	gyro y	gyro y	gyro y	200
0x792	gyro z (msb, scalar 1e-7, unit rad/sec)	gyro z	gyro z	gyro z	body accel x (msb, scalar 1e-7, unit m/sec/sec)	body accel x	body accel x	body accel x	200
0x793	body accel y	body accel y	body accel y	body accel y	body accel z	body accel z	body accel z	body accel z	200

	(msb, scalar 1e-7, unit m/sec/sec)				(msb, scalar 1e-7, unit m/sec/sec)				
0x794	latitude (msb, scalar 1e-7, unit decimal deg)	latitude	latitude	latitude	longitude (msb, scalar 1e-7, unit decimal deg)	longitude	longitude	longitude	200
0x795	altitude (msb, scalar 1e-5, unit m)	altitude	altitude	altitude	barometric pressure (msb, scalar 1e-1, unit kpa)	barometric pressure	temp (msb, scalar 1e-1, unit C)	temp	200
0x796	velocity x (msb, scalar 1e-7, unit m/s)	velocity x	velocity x	velocity x	velocity y (msb, scalar 1e-7, unit m/s)	velocity y	velocity y	velocity y	200
0x797	velocity z (msb, scalar 1e-7, unit m/s)	velocity z	velocity z	velocity z	forward velocity (msb, scalar 1e-7, unit m/s)	forward velocity	forward velocity	forward velocity	200
0x798	year	month	day	hour	minute	second	millisecond (msb, scalar 0, unit ms)	millisecond	200
0x799	tracked satellites	GNSS fix type	filter state	vel uncert	yaw uncert	roll uncert	pitch uncert	time valid state	200
0x79A	sprint distance (msb, scalar 1e-2, unit m)	sprint distance	sprint race time (msb, scalar 1e-2, unit sec)	sprint race time	slip angle (msb, scalar 1e-2, unit deg)	slip angle	dead reckoning enable state		200
0x79B	raw accel x (msb, scalar 1e-2, unit m/s/s)	raw accel x	raw accel y (msb, scalar 1e-2, unit m/s/s)	raw accel y	raw accel z (msb, scalar 1e-2, unit m/s/s)	raw accel z	transmission rate enum	baud rate enum	200
0x79C	quaternion x (msb, scalar 1e-4, unit none)	quaternion x	quaternion y (msb, scalar 1e-4, unit none)	quaternion y	quaternion z (msb, scalar 1e-4, unit none)	quaternion z	quaternion w (msb, scalar 1e-4, unit none)	quaternion w	200
0x79D	0xFA **See CAN Input Section**	user rotation received flag	rotation x (msb, scalar 1e-3, unit radian)	rotation x	rotation y (msb, scalar 1e-3, unit radian)	rotation y	rotation z (msb, scalar 1e-3, unit radian)	rotation z	1
0x79E	0x14 **See CAN Input Section**	user antenna translation received flag	antenna translation x (msb, scalar 1e-3, unit m)	antenna translation x	antenna translation y (msb, scalar 1e-3, unit m)	antenna translation y	antenna translation z (msb, scalar 1e-3, unit m)	antenna translation z	1

0x79F	0x11 **See CAN Input Section**	user accel / gyro filter request flag	accel filter window	gyro filter window					1

CAN DEFINITIONS (INPUT):

The items in this section are available for advanced users that wish to configure the sensor output to suit their specific needs.

It is imperative that this sensor must not be installed on a bus with active traffic on the 0x006 CAN ID. This ID is used for sensor configuration and will lead to unpredictable behavior if random traffic enables or disables some of these toggles.

AN EXAMPLE:

The general idea behind these user inputs is that all messages that are sent from the user to the sensor will be transmitted on the **0x006** ID. The general message structure will look like this:

ID	Byte 0	Byte 1	Byte 2	Byte 3
0x006	0x06	0x66	0x10 (Parameter selection byte)	0x00 (Parameter toggle byte)

Generally, byte 2 (aside from the **0xFA Reference Frame Rotation**) is a descriptor of which function you are trying to access. Bytes 3 - 7 are for user entered values or toggles (on/off). In the example above, sending this message will **DISABLE (0x00)** the Adaptive Filter functionality of the system and produce better orientation estimates in some cases. If you wished to re-enable Adaptive Filter functionality, you would simply exchange the 0x00 in byte 3 for an 0x01. This will **ENABLE (0x01)** the Adaptive Filter.

This area will be under constant update and development as new user input functions are added. The following will describe all the current user configuration parameters and their uses.

0X10 (ADAPTIVE FILTERING / ENABLED BY DEFAULT)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Type
0x006	0x06	0x66	0x10	0x00 == Disabled 0x01 == Enabled	Toggle

This setting will only be needed in the most extreme vibration conditions. The Adaptive Filtering that is done on-board the sensor is used to make the orientation estimates in most cases more reliable. However, it can have adverse effects on the orientation data in some cases. This allows the user to disable it if they wish. **In general, this field should be left alone unless instructed to change this by Obsidian Group**.

IMPORTANT: This setting **will not persist** after a power cycle unless you have used the **0x66 (WRITE SETTINGS TO MEMORY)** function.

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X11 (ACCELEROMETER / GYROSCOPE DATA OUTPUT FILTER / DISABLED BY DEFAULT)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Type
0x006	0x06	0x66	0x11	0x01 == Enabled 0x00 == Disabled	Accel Filter Window Size **Example: 0x20 == Window Size of 32**	Gyro Filter Window Size **Example: 0x10 == Window Size of 16**	Toggle + 8-bit unsigned ints

The accelerometer data can sometimes be hard to view in the absolute sense due to the noise in the signal data. If the user wishes to apply a moving average filter (boxcar) across the data, the sensor can do this filtering on-board. The filter window is also configurable by the user.

The maximum filter window size is 120 samples. The raw IMU rate of the sensor is 400hz. If a window size of 40 is used, this would be the rough equivalent of a 10hz moving average filter. In practice, I would not suggest exceeding a window size of 20-30. If values are too high, it may result in over-filtering.

If the filter is disabled with the 0x00 flag on byte 3, the sensor will ignore any data from bytes 4 - 7.

IMPORTANT: Sending these filter parameters will cause a brief delay in transmitted data while the filter is enabled.

IMPORTANT: These parameters will persist in the future until the filter parameters are disabled by sending this message again and setting byte 3 to **0x00** (Disabled).

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X12 (AHRS CONTROL / ENABLED BY DEFAULT)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Type
0x006	0x06	0x66	0x12	0x01 == Enabled	Toggle

				0x02 == Disabled	

AHRS stands for Attitude and Heading Reference System. This system attempts to use the magnetometer in the sensor to estimate the current heading of the vehicle even when it is stationary before the system has achieved a GNSS fix. In practice, the magnetometer can be confused by noisy environments and there are times where disabling this system will yield quicker filter convergence times.

This module can only be disabled after it has been enabled.

IMPORTANT: This setting **will not persist** after a power cycle unless you have used the **0x66 (WRITE SETTINGS TO MEMORY)** function.

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X13 (ADAPTIVE FILTER PARAMETER TUNING / PRE-CONFIGURED BY DEFAULT)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Type
0x006	0x06	0x66	0x13	atX (bits 0 – 3) atY (bits 4 – 7)	atZ (bits 0 – 3) afX (bits 4 – 7)	afY (bits 0 – 3) afZ (bits 4 – 7)	4-bit unsigned ints

The adaptive filter parameter tuning allows the end user to adjust some of the pre-filtering that is done on the raw signaling before the data passes through the EKF. **In general, this field should be left alone unless instructed to change this by Obsidian Group**.

IMPORTANT: This setting **will not persist** after a power cycle unless you have used the **0x66 (WRITE SETTINGS TO MEMORY)** function.

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X14 (IMU TO ANTENNA LEVER ARM TRANSLATION / PRE-CONFIGURED TO 0.75M ABOVE IMU)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Type
0x006	0x06	0x66	0x14	Translation X (xT) **(Example:**	Translation Y (yT) **(Example:**	Translation Z (zT)	8-bit signed int scaled by 10

				0xF3 == -1.3m)	**0xE6 == -2.6m)**	**(Example: 0x10 == 1.6m)**	

The IMU to Antenna Lever Arm Translation is an important aspect of any IMU + GNSS system. This translation informs the EKF the correlation of movement that the antenna might see to the movement that the IMU observes. It is, by default, configured such that the GNSS antenna will be mounted 0.75m directly above the IMU. This default configuration assumes no lateral translation. If the installation of the antenna requires that the translation be something other than this default, this must be programmed into the INS using this input.

Regarding the example above:

The coordinate frame of the INS uses a "Forward, Right, Down" convention which means that the X axis is positive in the forward direction, the Y axis is positive in the right direction, and the Z axis is positive in the down direction. If we use the values in the example above, that will place the antenna 1.3 meters **behind** the INS, 2.6 meters to the **left** of the INS, and 1.6 meters **below** the INS. Obviously this is an implausible example (being that you wouldn't generally want to mount the antenna below the INS!).

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X20 (TRANSMISSION RATE CONTROL)

ID	**Byte 0**	**Byte 1**	**Byte 2**	**Byte 3**	**Type**
0x006	0x06	0x66	0x20	0x01 == 400Hz 0x02 == 200Hz 0x03 == 100Hz 0x04 == 50Hz 0x05 == 25Hz 0x06 == 10Hz	Toggle

The transmission rate control directly controls the rate at which CAN messages are published from the sensor. This does not have any impact on the calculation speed of the internal EKF.

You will only be able to change the transmission rate **once** per power cycle. This is to make sure that the sensor does not enter any unpredictable behavior if there are many rate changes requested at any one time.

IMPORTANT: This setting will be written to the internal memory of the device after it has been set. This means that at the next power up, it will maintain the transmission rate that you have set.

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X21 (CAN BAUD RATE CONTROL)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Type
0x006	0x06	0x66	0x21	0x01 == 1Mbps 0x02 == 500Kbps 0x03 == 250Kbps	Toggle

This toggle will control the baud rate of the CAN bus. Once this command has been set, it will take effect at the next power cycle.

An Example: If the sensor is up and running on a 1mbps bus, and you send a CAN command of the following:

ID	Byte 0	Byte 1	Byte 2	Byte 3
0x006	0x06	0x66	0x21	0x02

This will force the device to start up **at the next power cycle** in a 500kbps mode. **IT IS IMPERATIVE that you keep in mind that it is not possible to use this sensor at 400 Hz mode on a 500kbps CAN bus.** You must reduce the transmission rate (see directly above) to a minimum of 200 Hz.

IMPORTANT: This setting will be written to the internal memory of the device after it has been set. This means that at the next power up, it will maintain the baud rate that you have set.

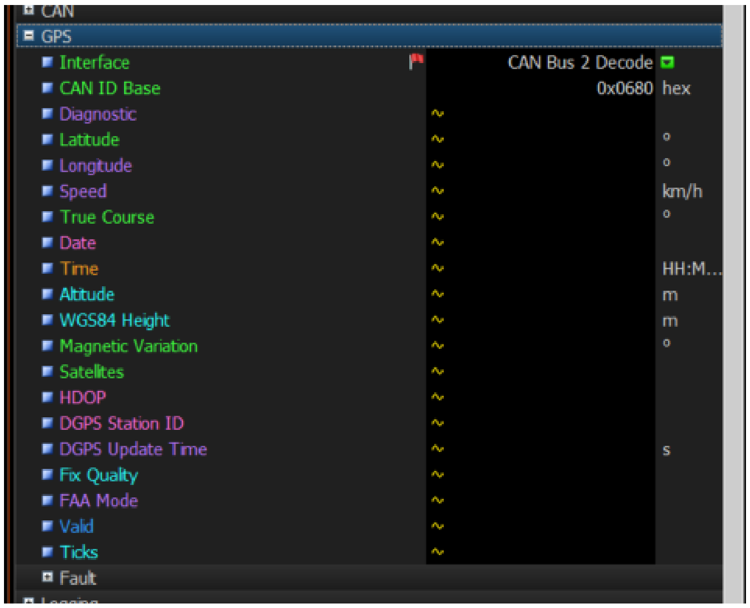
IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X30 (GPS SIMULATION OPTION)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Type
0x006	0x06	0x66	0x30	0x00 == Disabled 0x01 == MoTeC GPS Simulation (Default) 0x02 == Emtron GPS Simulation	Toggle

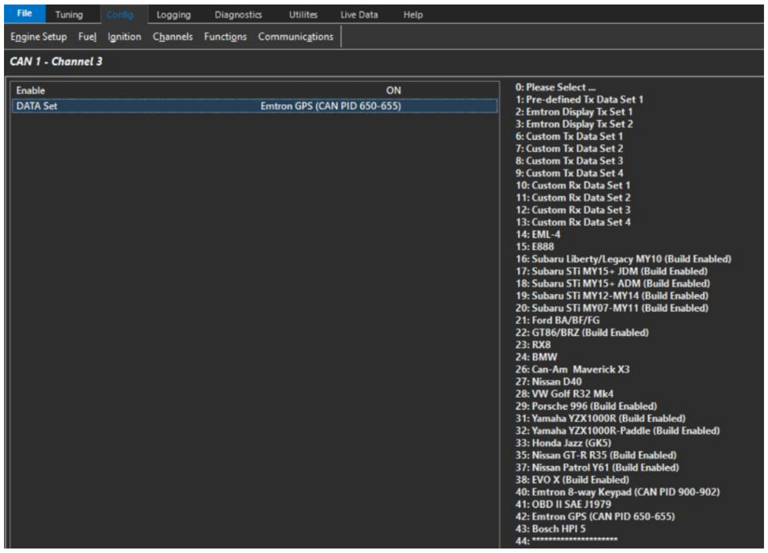
This will toggle the various GPS simulation options to directly integrate with devices from MoTeC or Emtron Pty Ltd. If enabled, the sensor will generate data in the format that these devices are expecting to receive from their own GPS receiver units.

MoTeC M1 GPS Receive should be configured as follows:



```
CAN
GPS
  Interface                CAN Bus 2 Decode
  CAN ID Base              0x0680  hex
  Diagnostic
  Latitude                         °
  Longitude                        °
  Speed                            km/h
  True Course                      °
  Date
  Time                             HH:M...
  Altitude                         m
  WGS84 Height                     m
  Magnetic Variation               °
  Satellites
  HDOP
  DGPS Station ID
  DGPS Update Time                 s
  Fix Quality
  FAA Mode
  Valid
  Ticks
  Fault
```

Emtron GPS Receive should be configured as follows:



```
CAN 1 - Channel 3

Enable                                   ON
DATA Set            Emtron GPS (CAN PID 650-655)

0: Please Select ...
1: Pre-defined Tx Data Set 1
2: Emtron Display Tx Set 1
3: Emtron Display Tx Set 2
6: Custom Tx Data Set 1
7: Custom Tx Data Set 2
8: Custom Tx Data Set 3
9: Custom Tx Data Set 4
10: Custom Rx Data Set 1
11: Custom Rx Data Set 2
12: Custom Rx Data Set 3
13: Custom Rx Data Set 4
14: EML-4
15: E888
16: Subaru Liberty/Legacy MY10 (Build Enabled)
17: Subaru STi MY15+ JDM (Build Enabled)
18: Subaru STi MY15+ ADM (Build Enabled)
19: Subaru STi MY12-MY14 (Build Enabled)
20: Subaru STi MY07-MY11 (Build Enabled)
21: Ford BA/BF/FG
22: GT86/BRZ (Build Enabled)
23: RX8
24: BMW
26: Can-Am  Maverick X3
27: Nissan D40
28: VW Golf R32 Mk4
29: Porsche 996 (Build Enabled)
31: Yamaha YZX1000R (Build Enabled)
32: Yamaha YZX1000R-Paddle (Build Enabled)
33: Honda Jazz (GK5)
35: Nissan GT-R R35 (Build Enabled)
37: Nissan Patrol Y61 (Build Enabled)
38: EVO X (Build Enabled)
40: Emtron 8-way Keypad (CAN PID 900-902)
41: OBD II SAE J1979
42: Emtron GPS (CAN PID 650-655)
43: Bosch HPI 5
44: **********************
```

IMPORTANT: This setting will be written to the internal memory of the device after it has been set. This means that at the next power up, it will maintain the GPS simulation option that you have set.

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.

0X40 (DEAD RECKONING ENABLE / DISABLE)

ID	Byte 0	Byte 1	Byte 2	Byte 3	Type

0x006	0x06	0x66	0x40	0x00 == Disabled 0x01 == Enabled	Toggle

This field can enable the calculation of the Race Distance (m) and Race Time (s) channels that are transmitted by the INS on CAN ID 0x79A. This toggle allows the end user to start the dead reckoning process and enable the Race Distance (m) and Race Time (s) channels. An intuitive example of this use is to set **byte 3** to 0x01 when a drag race car has let off the staging brake (or transmission brake, in the case of a car with a torque converter).

IMPORTANT: By default, the INS is set to start the dead reckoning process when the vehicle's forward velocity is greater than 0.75 m/s. You can preempt this by setting **byte 3** to 0x01 before the vehicle has reached 0.75 m/s forward velocity.

0X66 (WRITE SETTINGS TO MEMORY)

NOTE: USE THIS ONLY IF YOU ARE SURE THAT YOUR SETTINGS ARE VALID.

ID	**Byte 0**	**Byte 2**	**Byte 3**	**Type**
0x006	0x06	0x66	0x66	Toggle

This field will allow the end user to write the previously configured settings to the non-volatile memory (NVM) of the sensor. Once you have sent this command using the bytes listed above, the settings will be written to the sensor, and they will persist after a power cycle.

IMPORTANT: Sending this command will result in a brief delay of transmitted data from the INS.

0X67 (WRITE DEFAULTS TO MEMORY)

ID	**Byte 0**	**Byte 2**	**Byte 3**	**Type**
0x006	0x06	0x66	0x67	Toggle

This field will allow the end user to recover the default settings that were configured on the sensor at time of shipment. If there is any confusion about settings or strange behavior of the sensor, this command will reset all of the parameters that could have been adjusted back to their defaults.

IMPORTANT: Sending this command will result in a brief delay of transmitted data from the INS.

0XFA (REFERENCE FRAME ROTATION)

NOTE: IF YOU ARE APPLYING MORE THAN ONE ROTATION, BE SURE TO UNDERSTAND THE ORDER OF ROTATIONS. THE ORDER OF ROTATIONS IS [X (ROLL), Y (PITCH), Z (YAW)]

ID	**Byte 0**	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Type**

0x006	0xFA	Roll (X) Rotation (msb)	Roll (X) Rotation	Pitch (Y) Rotation (msb)	Pitch (Y) Rotation	Yaw (Z) Rotation (msb)	Yaw (Z) Rotation	Radian Int scaled by 1e3

This field will allow the end user to rotate the body frame to align with the vehicle frame to cope with misalignments in the installation of the sensor in the vehicle. Practically speaking, this will be needed if the sensor is not mounted in the car so that the X-axis is pointing precisely forward, and that the Z-axis is pointing precisely down.

The rotation values for X, Y and Z are designed to be sent to the INS as radians that are converted to integers and scaled by 1000.0. This allows for a minimum and maximum rotation of -32.768 and 32.767 radians respectively.

An example:

Say you need to correct the misalignment in the Z axis (Yaw) by a factor of 3 degrees:

- Convert 3 degrees to radians by multiplying by pi (3.14159) and multiplying by 180. This will result in 0.5236 radian.
- Multiply 0.5236 radians by 1000. This will result in 52.36.
- Convert 52.36 to an integer. This will result in 52.
- Convert 52 to a hexadecimal number. This will result in 0x34.
- The resulting data you would send to the INS would be:
    - [id / b0 / b1 / b2 / b3 / b4 / b5 / b6 / b7]
    - [0x006 / 0xFA / 0x00 / 0x00 / 0x00 / 0x00 / 0x00 / 0x34 / 0x00]

Now let's say you need to correct the misalignment in the Z axis (Yaw) by a factor of -2 degrees:

- Convert -2 degrees to radians by multiplying by pi (3.14159) and multiplying by 180. This will result in -0.034 radians.
- Multiply -0.034 radians by 1000. This will result in -34.90 radians.
- Convert -34.90 to an integer. This will result in -34, however, you can round up to -35.
- Convert -35 to a hexadecimal number.
    - Subtract -35 from 65536. This will result in 65501.
    - Convert 65501 to a hexadecimal number. This will result in 0xFFDD.
- The resulting data you would send to the INS would be:
    - [id / b0 / b1 / b2 / b3 / b4 / b5 / b6 / b7]
    - [0x006 / 0xFA / 0x00 / 0x00 / 0x00 / 0x00 / 0xFF / 0xDD]

IMPORTANT: Sending this command will result in a brief delay of transmitted data from the INS.

IMPORTANT: This rotation will be written to the non-volatile memory of the INS and as such will only need to be set once after installation.

IMPORTANT: This can only be set once per power cycle. This is to avoid spamming or incorrect configuration causing unknown behavior.