# PILOT™ USER GUIDE

# CONTENTS

# IMPORTANT NOTICES

Waste Electrical & Electronic Equipment (WEEE) Regulations require that any end-of-life units are returned to yellowcog limited for processing. This service is free-of-charge - please read the Recycling and Disposal section of this manual.

DISCLAIMER: Yellowcog has a continuous development programme. All products and features described in this document are subject to change without notice; this includes power consumption figures, weights and dimensions which are all approximate.

IMPORTANT: The Pilot™ must only be used with approved hardware. Using alternative hardware will **void the warranty** and may result in permanent damage to the unit.

IMPORTANT: You must read the installation guide before connecting the Pilot™ to any other equipment. Incorrect connections will **destroy the unit** and **will not be covered by the limited warranty**.

IMPORTANT: This is not a clinical or medical device. This device is not to be used for diagnosis or treatment of medical conditions.

IMPORTANT: This device is for motorsport use only.

IMPORTANT: All devices can and do fail! Long recording sessions risk losing larger amounts of data. Immediately back-up all data recorded by the connected logger to avoid loss.

# GLOSSARY

| Term | Description |
|---|---|
| 3V3 | 3.3 Volts. This references either the internal voltage of the Pilot™ or, more commonly, the 3.3 volt serial communications option. |
| ANT | Robust multi-cast 2.4GHz wireless protocol. ANT has many defined communication requirements but does not define or care for what payload data means. |
| ANT+ | Based on the ANT protocol with defined “profiles” that dictate what the payload data means. ANT+ devices are compatible with other equipment that supports the complementary profile. Secure connections are possible but rarely used. |
| Arrived | In the wireless world things are not plugged in. Arrived is used to refer to a device coming into range, or being switched on when in range etc. When a device arrives is can be connected to. See also “Departs”. |
| Bluetooth | Robust device-to-device 2.4GHz wireless protocol. The Bluetooth standard defines many roles such as “remote control” or “audio speaker” but does not do so well for fitness devices. As such, most Bluetooth protocols are proprietary and need explicit implementation. |
| Bluetooth Low Energy (BLE) | Robust multi-cast and secured device-to-device 2.4GHz wireless protocol. Functionally it is equivalent to ANT+ (which also has security). It is supported by most smart phones. |
| Bluetooth Smart | Older trademarked name for BLE. |
| Bonded | Used with Bluetooth LE to mean two devices that already have stored security keys that they’ve previously exchanged. |

| | |
|---|---|
| Connected | For wired parts, this is simply being plugged in and wired up. For wireless devices the meaning varies. This document will use connected to mean a device that is actively delivering data to the system (which may mean to some that it is linked, bound, bonded, listening etc) |
| Connector | The main connector on the Device. |
| Departs | In the wireless world things are not plugged in. "Departs" is used to refer to a device going out of range, or being switched off when in range, or batteries running out etc. When a device departs it can no longer be connected to. See also "Arrives". |
| Device | A "Device" is a distinct piece of hardware in the system. Since both Pilot™ and connected devices can be transmitters and receivers then it is often the case that they are both just "devices" with no master or slave. |
| Master | The device controlling or initiating a link. We avoid this term since the common usage is often at odds with the technical e.g. a cycle computer is in control of which sensors it connects to yet it is a slave device (the sensors are the masters). |
| Module | An individual configuration option that maps to a real-world device, an input or output method or an abstract operation. |
| Pairing | Usually Bluetooth, the act of establishing communications with another device or peripheral. |
| Panel | When used in the context of Module configuration, this is a section where certain similar properties of the Module can be configured. |
| Peripheral | Any other device (commonly wireless) that communicates with the Pilot™. |
| Pilot™ | The Pilot™ device or development equivalent. i.e. the hardware. |
| Port | The Device can have more than one input/output of the same type, for example there can be multiple serial ports. |
| Property | When used in the context of Module configuration, this refers to an individual or group of configurable settings. |

| | |
|---|---|
| Protocol | Ports can use different "languages" to communicate. A protocol is the language used and may be industry standards like JSON or proprietary or contrived. |
| Receive | The context defines what is receiving. "The Pilot™ RX pin" would mean the pin that the Pilot™ collects data on. Connector diagrams will always show the name of the equipment that sports the connector and therefore Receive/RX will refer to the pin that that device collects data on. Direction arrows show the arrow pointing in the direction of data flow. |
| RX | See Receive. |
| Slave | See Master. |
| Transmit | The context defines what is transmitting. "The Pilot™ TX pin" would mean the pin that the Pilot™ outputs data on. Connector diagrams will always show the name of the equipment that sports the connector and therefore Transmit/TX will refer to the pin that that device outputs data on. Direction arrows show the arrow pointing in the direction of data flow. |
| TX | See transmit. |
| Variant | Hardware variants are distinct versions of hardware that provide specific capabilities. Individual variants are suited to specific purposes such as development work, production monitoring or configuration flexibility. |

# GETTING STARTED

## INTRODUCTION

Yellowcog introduces the Pilot™ range for physiological data collection both on and off the track.

Pilot™ bridges the gap between driver and machine. Driver vital signs can now be gathered wirelessly and seamlessly delivered straight to the ECU, logging and telemetry systems. The Pilot™ is a small matchbox-sized device that can be located anywhere convenient in the vehicle.

To ensure that your team stays ahead of its competitors the Pilot™ collects from many different commercially available sensors: from wireless chest-strap monitors to multi-parameter medical grade devices.

# HARDWARE

## LEDS

All Pilot devices have activity LEDs but the Pilot-MUX-2V2 has a secondary (alternative) LED set to report details of the expansion radio communications. There are three LED indicators: a green CPU light, a main activity indicator which can illuminate red or yellow (or a combination of red and yellow) and an alternative activity indicator which can also illuminate red/yellow.

The following table explains meaning of the LED light sequences.

| LED | Flash State | Description |
|---|---|---|
| MAIN - CPU | Flashing Green, 1Hz | System operational |
| MAIN - CPU | Flashing Green, 4Hz | System in reprogramming mode |
| MAIN – Activity | Flashing Red, 2Hz | Bluetooth start-up mode |
| MAIN – Activity | Red, on | Missing Bluetooth connection |
| MAIN – Activity | Flashing Yellow, 1Hz | Single Bluetooth connection established |
| MAIN – Activity | Flashing Yellow, 1Hz – 'n' multi blip | 'n' Bluetooth connections established |
| ALT – Activity | Red, on | Missing ANT/BLE connection |
| ALT – Activity* | Flashing Yellow, 1Hz | Single ANT/BLE connection established |

| | | |
|---|---|---|
| ALT – Activity | Flashing Yellow, 1Hz – 'n' multi blip | 'n' ANT connections established |

*for Pilot-DAC variants the ALT Activity light is flashed to signify that the analogue converter is running. More complex patterns are not supported in this mode.

Please see the installation section for details of the specific connectors available.

# SENSORS

The Pilot™ interface wirelessly collects information from a driver-worn physiological monitor. This information is delivered in real-time to the car's on-board management unit. Data is then processed the same as other car parameters and sent out over the car's telemetry so it is available at the pit wall at all times. The Pilot™ device is capable of connecting to both wired and wireless devices. It supports multiple wired and wireless input and output standards. Most features are configurable by the user.

Parameters collected include: heart rate, ECG, R-R interval, heart rate variability, breathing rate, core temperature and g-forces. The Pilot™ interface supports fitness and medical grade monitoring devices and other devices can be added.

# PRODUCTION VERSIONS

The Pilot™ is currently supplied in a number of variants, and according to customer specifications.

| Variant | CAN | RS-232 Out | RS-232 In | Serial 3V3 Out | Serial 3V3 In | Analogue Outputs | Connector ID (Pin Count) | Customer Connector | Pilot™ Connector |
|---|---|---|---|---|---|---|---|---|---|
| MUX | Y | Y | | Opt | Opt | | AS5P-MUXA (5) | ASL606-05SN-HE | ASL006-05PN-HE |
| 232 | | Y | Y | | | | AS5S-232A (5) | ASL606-05SN-HE | ASL006-05PN-HE |
| DAC | | | | | | 8 | DE15S-DACA (15) | D15 HD P | D15 HD S |
| OVR | Opt | | | | | 8 | DE9M-PLTOVR (9)<br>DE9F-PLTANOUT (9) | DE9F (9)<br>DE9M (9) | DE9M<br>DE9F |
| CAN2 | Y | | | Opt | Opt | | DE9M-PLTCAN | DE9F | DE9M |

# DEVELOPMENT VERSIONS

| Variant | CAN | RS-232 Out | RS-232 In | Serial 3V3 Out | Serial 3V3 In | Analogue | Connector Pin Count | Customer Connector | Pilot™ Connector |
|---|---|---|---|---|---|---|---|---|---|
| 3V3 | | | | Y | Y | | 5 | ASL606-05SN-HE | ASL006-05PN-HE |
| Hybrid | Opt | Opt | Opt | Opt | Opt | <= 2 | <= 8 | Customer Spec | Customer Spec |

# INSTALLATION

## IMPORTANT!

Use the correct device Installation Guide – connecting up the device incorrectly will likely destroy it and will NOT be covered by the limited warranty. Installation Guides MUST be downloaded from the web-site, always check for an updated version. Different hardware types and version DO NOT use the same pin-outs!

## POWER SUPPLY

The Pilot™ device requires between 5 and 24 Volts to operate. It has been designed to tolerate voltage changes and noise but a stable supply free from noise will ensure optimum performance. The internal voltage regulation is protected against transient and DC changes. Damage will occur if the absolute maximum ratings are exceeded. If integrating with an existing system (such as a car CAN bus and supply) it is likely that the existing hardware is already within the limits required by the Pilot™ device. The Pilot™ device is also protected both in software and hardware against brown-out and slow rise power conditions, although operations will be interrupted the Pilot™ will perform an internal reset and resume operation as soon as conditions are stable. Noisy or supplies that drop below the required minimum voltage will result in interruptions to the collection and distribution of data.

## CAN INSTALLATIONS

Installation on a CAN bus requires adding the appropriate connector to the customer equipment loom and connecting the Pilot. The most important consideration (as with any hardware) is to ensure that the correct pin wiring is in place. In particular, damage may occur if ground and the positive supply are reversed or supply level voltages are applied to the communications pins. It should also be noted that BUS controllers (data loggers, ECUs etc.) are able to reconfigure pin functions and it should be ensured that alterations in the configuration must not revert to a case where pin voltages or functions are damaging to the device.

Since CAN is a two wire bidirectional bus there is no concept of transmit and receive lines. The bus wires must still observe the correct polarity with the Pilot™ CAN-H pin and CAN-L being wired to the bus High and Low lines respectively.

The Pilot™ device does not provide bus termination. Data communications will fail if the bus is not terminated. Normal CAN bus procedures should be used to ensure that the bus has the correct termination at each end.

# RS-232 INSTALLATIONS

The Pilot™ device conforms to the RS-232 standard but does not support hardware flow control. Additionally, in the CAN specification hardware, only the transmit line for RS-232 is provided. For the full RS-232 specification Pilot™ device, both incoming (RX) and outgoing (TX) data are supported.

A wide range of baud rates and bit packing formats are supported. All well-known rates up to 115200 baud are supported. Unusual baud rates may be supported but internal clock limitations may mean that certain rates are not possible – it is sometimes possible to correct for such errors by slightly misconfiguring the baud rate of the Pilot.

# ANALOGUE INSTALLATIONS

For versions of Pilot™ that have analogue capabilities (currently the Pilot™-Mux with a PilotPlus-DAC expansion card or the Pilot™-OVR) then direct connection to data logger analogue inputs is possible. All versions with analogue output a voltage that varies between 0 volts and 3.3 volts according to the stream.

# 3V3 INSTALLATIONS – DEVELOPERS ONLY

The Pilots native internal voltage is 3.3 volts (hence 3V3) and bidirectional serial communications are available at this level. 3V3 communications use zero volts and 3V3 volts to transmit and receive the serial data. A bit ‘1’ is represented by a high (3V3) level and a ‘0’ by a low (zero volt) level. This is standard for inter-processor communications but is the opposite polarity to that of RS-232.

There is **NO PROTECTION** on these lines, they are wired directly to the processor. Noise or overvoltage on these lines will destroy the Pilot™ device. It is highly recommended that these lines are only attached to another local 3V3 microprocessor-based device. There is absolutely no warranty available when connecting to the device in this way. For further details, please contact us.

# CONFIGURATION SOFTWARE

## EMBEDDED SETUP

The configuration software (AKA “Embedded Setup”) is available to registered users from:

https://www.yellowcog.com/userportal

Embedded Setup allows the device to be reconfigured to suit the user’s preferences. Configuration is only required when altering the system to:

- Collect data from new or additional sensors.
- Collect data from replacement sensors.
- Output data in a different format.

If purchasing a kit that includes a BioHarness then it will arrive pre-configured for immediate operation.

Pilot™ comes with configuration software that can reprogram the device using a Bluetooth connection. A wide variety of configurations are possible but there are limits dictated by the available hardware connections, memory size, processor speed and licencing. It is unlikely that any of these limitations will come into play in a real-world installation.

The starting point for a configuration is to decide how the Pilot™ will output its data. This decision is made before even purchasing the device since not all device variants can support all the different configuration options.

### INSTALLATION

Download the ZIP file from the yellowcog website and extract to a folder of your choice. If updating the software it is acceptable to extract the files into an existing Embedded Setup install folder. Once the files are unzipped, run the executable “Yellowcog Embedded Setup.exe”.

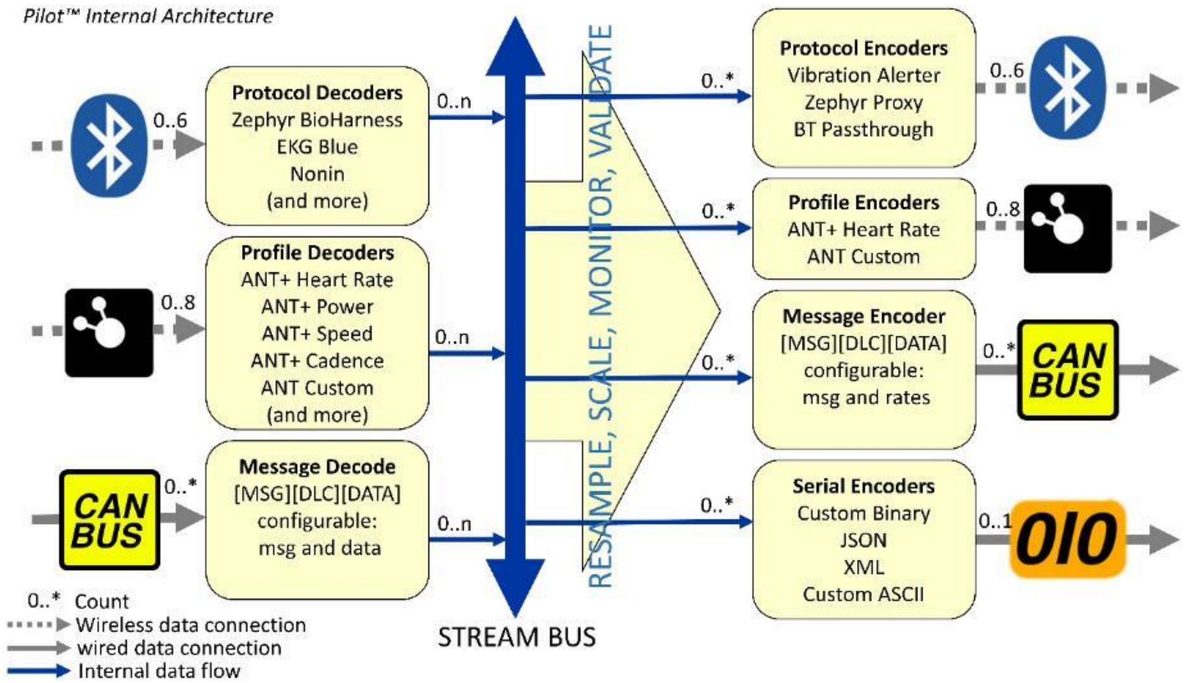
# CONFIGURATION CONCEPTS

## CONFIGURATION MODULES

To serve any practical purpose the Pilot device must have one or more configuration modules added. Each module is a stand-alone software unit that interact with other modules via streams. Configuration modules are generally either a source of information (i.e. streams of data) or a sink (i.e. they receive data and process or convert it). Some modules can be configured to be both sources and sinks (e.g. CAN bus) and others by their purpose must sink data before providing a new source stream (e.g. the Exertion analyser).

## STREAMS AND FILTERS

Streams are the fundamental mechanism for transferring data through the Pilot™ system. A stream represents a value in time that is output from one of the many module. A stream value is stored internally as a 32-bit signed integer.




### STREAM NAMES

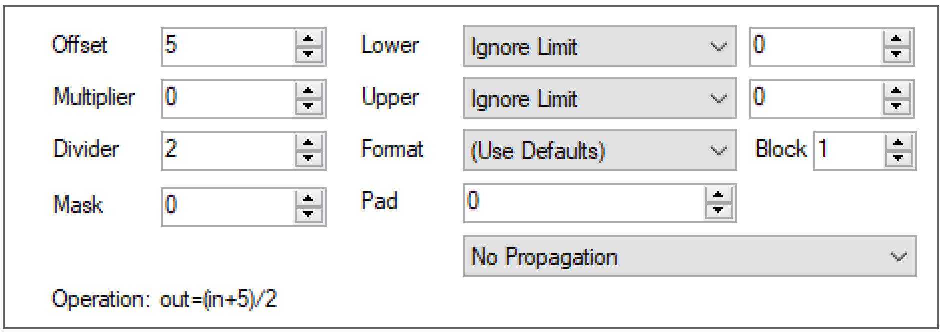
Each module that outputs a stream gives it an identifier. Identifiers make it possible to configure other modules to manipulate the stream. Most identifiers are predefined (such as for heart rate) but streams can also be used for custom data. In addition, the user can define an entirely new stream for use with their custom configurations.

### Filters

Filters are there to allow multiple streams with the identifier to be used. For example, it would be difficult to predefine how many heart rate sources there are in a system. Creating stream types called HeartRate1, HeartRate2 etc would be difficult to manage. Instead, every stream has a filter. The filter allows us to say there are x modules emitting heart rate and have each emit using their own filter. In this way, output modules can decide which of the filters they output. You could have one serial port filtering for Filter 1 and another for Filter 2 or a CAN message that emits, for example, eight different heart rates in a single package.

### Stream Maths

Most configured streams allow simple maths to be applied to a stream value either as it is read in (say, from a CAN bus) or before it is output (say, via ASCII).




Options are:

- Offset – This offset will be added to the stream value before transmission. This resultant is then passed to the multiplier.
- Multiplier – The stream output is multiplied by this value. If zero, no multiplication will be performed. The resultant is passed to the divider.
- Divider – The stream output is divided by this value. If zero, no division will be performed. The resultant is passed to the “Lower” filter.
- Lower – output values can be stopped from going below the number in the box to the right. The options are: Ignore – no operation is performed; Truncate – if the value is below the limit set it will be set to the limit value; or Invalidate – if the value is below the limit it will be set to the invalid value (reserved – always zero).

- Upper – same operation as Lower but for the high end limit.
- Format – Override the default formatting for this stream only. See Values (purple) above.
- Padding – Override the default number padding for this stream only. See Values (purple) above.
- Advanced: Propagation – this option allows a Pilot mastered CAN bus system to reflect messages up or down the hierarchy. Setting "upwards" will mean that data stored in this configured device will be reported to the Pilot master. Downwards is used on the master to explicitly ensure data is passed to one or more slaves.

### Filtered Stream Memory Requirements

Filters can be thought of in most configurations as representing devices or people. Every Pilot™ device has a fundamental hardware limit to the number of connections it can simultaneously handle. For example, the Pilot™-Mux can only make seven Bluetooth connections. All devices also have a fundamental limit on storage and memory. Since these limits exist, the software is given the same limits as the hardware. i.e. there is no simple advantage in being able to transmit eight independent heart rates if only seven can ever be gathered.

Since it is not clear in advance how a Pilot™ will be configured, the limits of filters and streams is combined i.e. Each Filter+Stream is an entity to be stored. So, "HR/1", "BR/1", "HR/2" requires three storage units. i.e. there is no penalty of how filters and streams are combined.

The current release of software allows 200 filtered streams to be configured.

# ADVANCED CONFIGURATION CONCEPTS

## Groups

Groups can be used to allow/refuse more than one source of data per group. If two wireless devices both have their Group et to "1" then even if two eligible heart rate straps are in range, the first to connect means that the other can't. This is useful in situations where only one person at a time should be monitored, or multiple people but not two of the same type at a time. E.g. driver and co-driver should be monitored at the same time but only one of each is ever permitted. For a yacht, it may be that tens can be monitored but you want to avoid each crew's backup device if the primary is already in use.

Leaving a group at zero means this setting is ignored but setting it to non-zero means that only one configured item per group can connect. i.e. if you have five devices all in group zero then if all the devices are present. If they are in a non-zero group then only one per group can be connected to at any one time.

Non-zero group numbers have no numerical meaning beyond associating devices together. i.e. Group 1 behaves as, say, Group 55. There is no interrelation or priority between groups, they are separate entities.

## LATE BINDING

Normally, a source of data, such as a heart rate monitor, is configured in the system with a Default Filter value that makes it clear where the data came from. i.e. if you have two people to monitor, you create two heart rate configuration modules, set one Default Filter to “0” and the next to “1”. Elsewhere in the system you can output each stream of data by using their filter valuers to define where they appear.

However, sometimes it is too limiting to work in this way. Late Binding tells the system that there is a set of kit that as and when it is connected to will be assigned a new Default Filter value. This allows more devices to be configured and downloaded than you might expect (or require) to operate simultaneously. There reasons include:

- You want a pool of more hardware than the Pilot can technically connect to. i.e. the hardware limit for Bluetooth connections on a Pilot-MUX is seven, but you might want to have ten available devices.
- You want to output a simpler output (e.g. CAN) message. i.e. you may want two devices (people) to be collected from but only one standard CAN message containing HR, BR, Serial Number etc. Late binding makes this simple.

From the above, you could, for example, have:

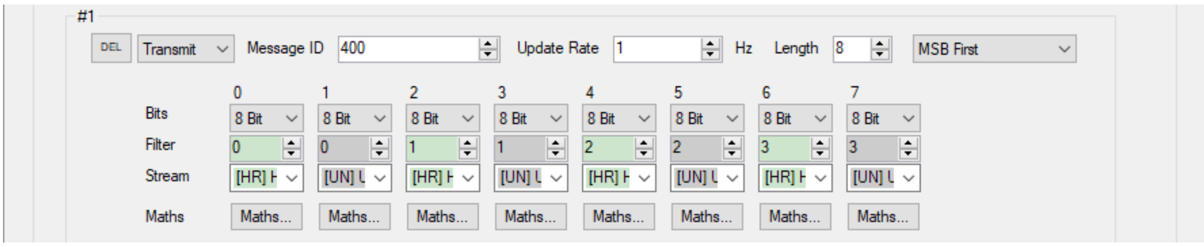
- 10 Bluetooth devices in the configuration and the first seven to arrive (the fundamental Bluetooth hardware limit) will each be assigned to its own filter so that it can be output elsewhere.
- Each Late Binding device must be described to the system. This is done by selecting the Late Binding Flag checkbox on each source device wireless settings. The flag is then used by the system to decide how to treat the devices outputted data.

Legacy: Previously, the Late Binding flag was signalled by adding 128 (technically, logically OR-ing) to the Default Filter value. This has now been replaced by the Late Binding checkbox.

Note: If you have a set of devices and each is powered and connects then the filter order will incremental. If you power multiple devices before they have all connected there is no way to predict which device will get which late binding filter slot, this is due to the way the wireless protocols work – the first device that *can* connect is not always the first device to be found.

### LATE BINDING EXAMPLE

We have six heart rate monitors, with serial numbers 100 to 106. We want a single CAN message that transmits heart rate and the serial number of the heart rate monitor. We configure a CAN message as:

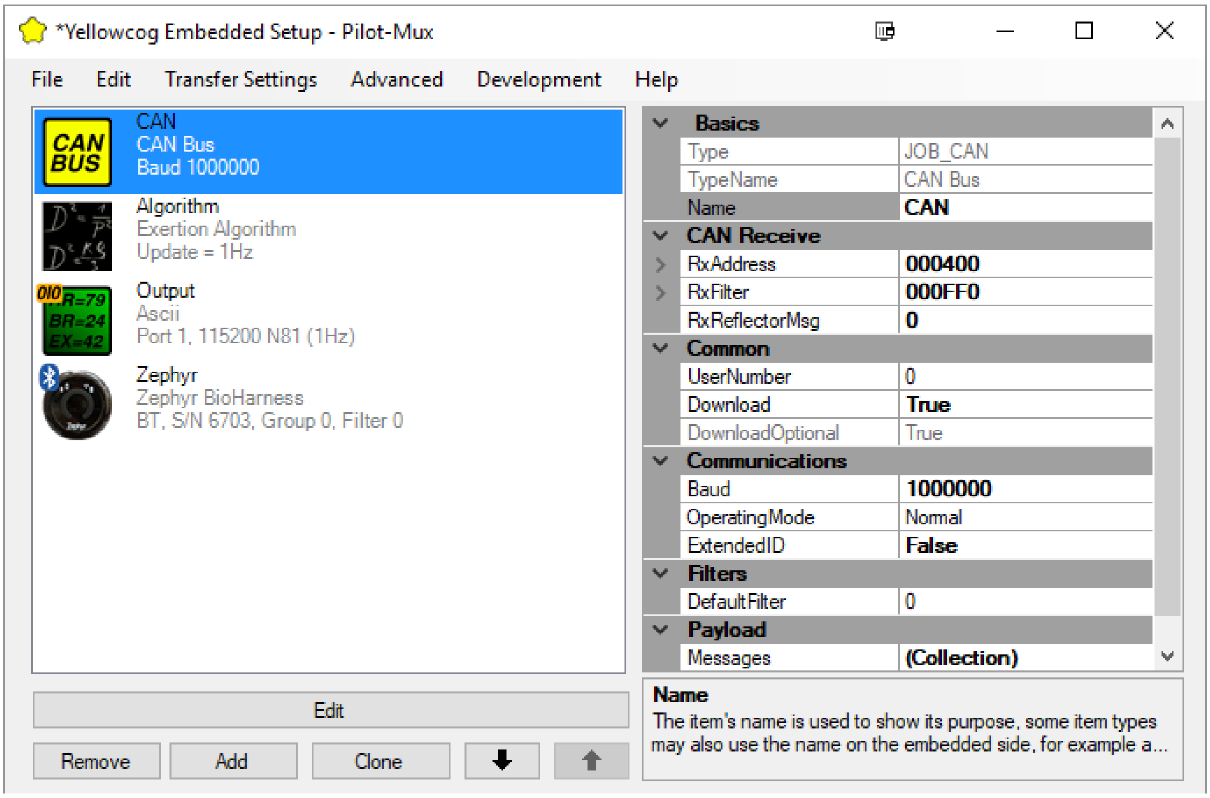



The following table imagines the heart rate monitors connecting and disconnecting over time in the order given. Note how #200 and #300 swap position in the message from Time 7. This is because at Time 7 when #300 reconnects, it takes the first free slot, equally at Time 8, #200 takes the (now higher) first free slot.

| Time | Action | Byte 0 | Byte 1 | Byte 2 | Byte 3 | Byte 4 | Byte 5 | Byte 6 | Byte 7 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | #100 Connects | HR for #100 | 100 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | #200 Connects | HR for #100 | 100 | HR for #200 | 200 | 0 | 0 | 0 | 0 |
| 3 | #300 Connects | HR for #100 | 100 | HR for #200 | 200 | HR for #300 | 300 | 0 | 0 |
| 4 | #400 Connects | HR for #100 | 100 | HR for #200 | 200 | HR for #300 | 300 | HR for #400 | 400 |
| 5 | #200 Disconnects | HR for #100 | 100 | 0 | 0 | HR for #300 | 300 | HR for #400 | 400 |
| 6 | #300 Disconnects | HR for #100 | 100 | 0 | 0 | 0 | 0 | HR for #400 | 400 |
| 7 | #300 Reconnects | HR for #100 | 100 | HR for #300 | 300 | 0 | 0 | HR for #400 | 400 |
| 8 | #200 Reconnects | HR for #100 | 100 | HR for #300 | 300 | HR for #200 | 200 | HR for #400 | 400 |

# MAIN EDITOR

The main editor screen allows different configuration modules to be added, edited, removed and downloaded to the hardware. On the left side is a list of configured modules. This list should, at a minimum, normally contain an output module (such as CAN or RS-232) and one or more input devices (such as a heart rate strap).




The editor is grouped into the:

- Menu strip – a set of standard Windows-style menu items.
- Module List – a list of configurable items or "modules".
- Properties List – a quick overview and editor for the highlighted module.
- Operations Buttons – Buttons required to add, remove and edit the modules.
- Feedback Window – feedback and error messages list (not shown).

The module list depends on what has been added to the system. By highlighting a module you can see the properties on the right-hand side. The properties editor is intended to give you an overview but can also be used to directly alter the configuration of each module. The properties editor is supplemented by a much richer editor screen that is available either by double-clicking the module or by highlighting the module and clicking “Edit”.

# GETTING STARTED

## OPENING A DEFAULT CONFIGURATION

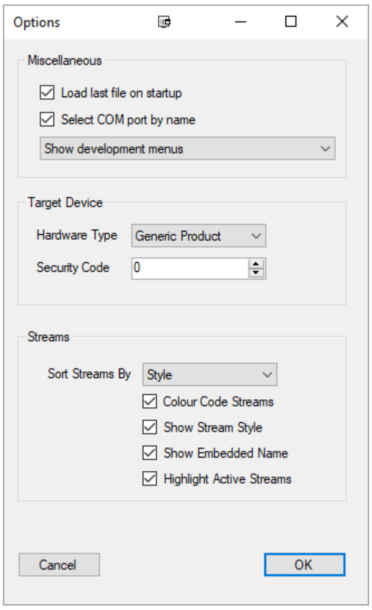
Embedded Setup comes with a folder called “Settings” in its installation path. Open the most appropriate to your hardware.

Note: It is **always** better to prove that the device as-shipped is operating according to the default setup. Making changes to the device before it has been seen to work leaves the user in the position of not know where to look for issues. e.g. is it the configuration of the Pilot, the cabling or the downstream system cabling or its configuration.

Yellowcog configuration files are ASCII in format but not generally editable by hand. They are saved as “ycd” files and best edited in the Embedded Setup program. It can be useful to associate “ycd” files with the setup program using the Window “Always open with” feature.

## OPTIONS WINDOW

Arguably, the best place to start configuring a system is to go to the options screen to get things behaving the way you want. To call up the options window, go to File->Options.



## Miscellaneous

#### *Load Last File On Startup*

Avoid the need to keep finding the last file being worked on.

#### *Select COM port by name*

COM port names and numbers are used to communicate over a hard-wired serial cables or Bluetooth (See the Bluetooth Download section for more details).

If this option is checked, Embedded Setup will ask Windows for a list of valid ports and their names, these names will be shown in dialogues that use a serial connection. However, there are some devices that misbehave in Windows. If you cannot find the serial port you expect then uncheck this option to allow the setting of the COM number in the dialogues.

## Target Device

#### *Hardware Type*

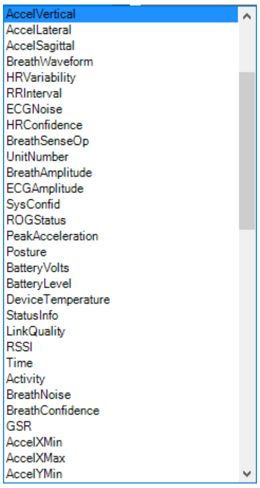
You can set the basic type of hardware you have in this option to limit the configurable modules that will be presented as available. Modules that are not supported by your hardware are shown faded, to help when selecting items. Custom hardware, or more advanced users, should leave this as “Generic Product”.

#### *Security Code*

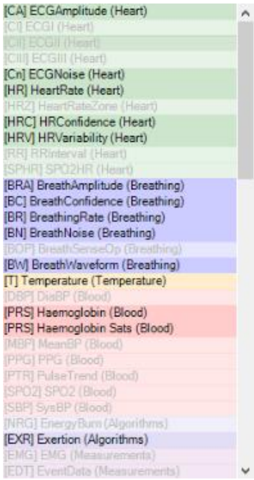
Advanced feature: If your device was shipped with an embedded security code then set this here. This must be left at zero for all standard devices.

## Streams

A number of options are given to help with displaying streams in the various dialogues. An example of the difference between all options off and all options on is shown below.




*All options off*




*All options on*

### *Sort Streams by Style*

Streams often need to be visually selected in the editor, particularly for items that output data. This option groups streams in dropdowns etc into general categories such as "heart", "blood", "breathing" to make finding the correct stream easier.

### *Colour Code Streams*

Like the Sort Streams by Style option, this shows streams in dropdowns with a colour to indicate their general style e.g. green for heart, red for blood and blue for breathing.

### *Show Stream Style*

An alternative or addition to the colour coding is to show the style in brackets after the name.

### *Show Embedded Name*

The Embedded Name is a very brief version of the streams descriptive name. For example, ASCII data for heart rate is by default sent with the identifier "HR". It can be useful to set this option to select using these embedded names.

### *Highlight Active Streams*

Configurations can be complex and there can be many modules capable of outputting different data on different streams and filters. To help in finding, and to provide feedback on the correctness of a configuration, the program highlights that are, or may be under the correct circumstances, active in the system.

For example, if you have a heart rate strap with the Default Filter set to 2 and then configure some sort of output device with a single stream being output. When you look at the filter and stream drop-down, you'll see that Heart Rate is shown in bold only when the filter is set to 2 as well. Conversely if the filter is set to another value then Heart Rate will be greyed out to show that there is no known source for this data. The system can only highlight what is configured to be possible sources of data, obviously in the real system, should a device never connect then there will never be that source of data.

# DOWNLOADING A CONFIGURATION

## SERIAL NUMBER

Some methods of download allow for "multi-drop" protocols. In general, leave the Serial Number settings (if it is even shown) as zero. Using zero means the serial number is not used, non-zero number introduce addressing of data and/or encryption so only a specific device will cooperate with the download procedure.

## FINDING THE CORRECT SERIAL PORT

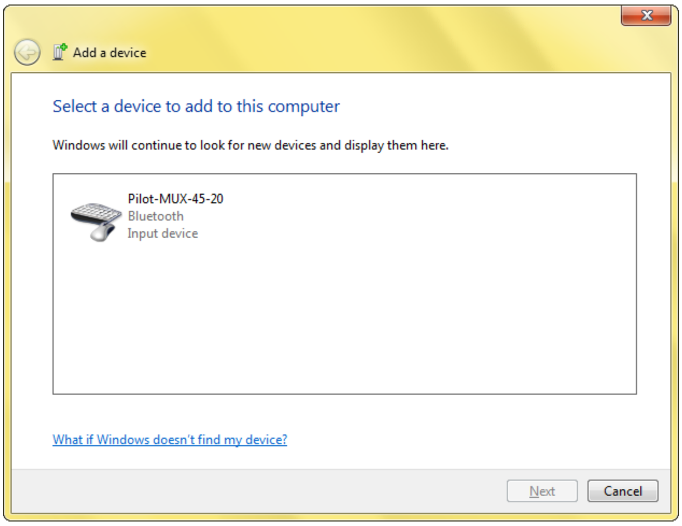
Devices that are capable of connection to a PC via a programmer cable can be downloaded to directly. Plug in the programmer cable (along with the secondary USB cable if fitted) and then use Windows's Devices and Printers panel to ascertain which serial COM number has been assigned. Then just set this COM port number in the download window.

## DOWNLOADING VIA BLUETOOTH

### ADDING PILOT™ AS A BLUETOOTH DEVICE

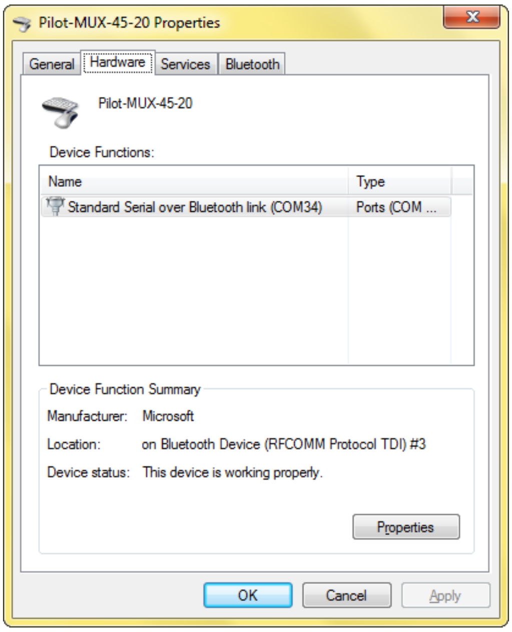
To download a configuration to a Pilot™ device via Bluetooth (i.e. not BLE) it is necessary to add the device via the Windows "Bluetooth Devices" control panel window. Calling up this window varies according to the version of Windows but is normally accessible via the control panel or via the small Bluetooth icon in the system tray.

Ensure the Pilot™ is powered and in-range of the PC and click "Add a device" or the equivalent. Note: The procedure is quicker if the Pilot™ is not connected to other Bluetooth sensors. In Windows 10, select "Add a Bluetooth Device" and choose the type as "Bluetooth". A similar dialogue to this shows in most versions of Windows.




Wait until Windows finds the Pilot device and displays its name and click OK. If prompted for a password/PIN enter, without quotes, “0000” (or “1234” for older devices). Now wait for Windows to install the default drivers.

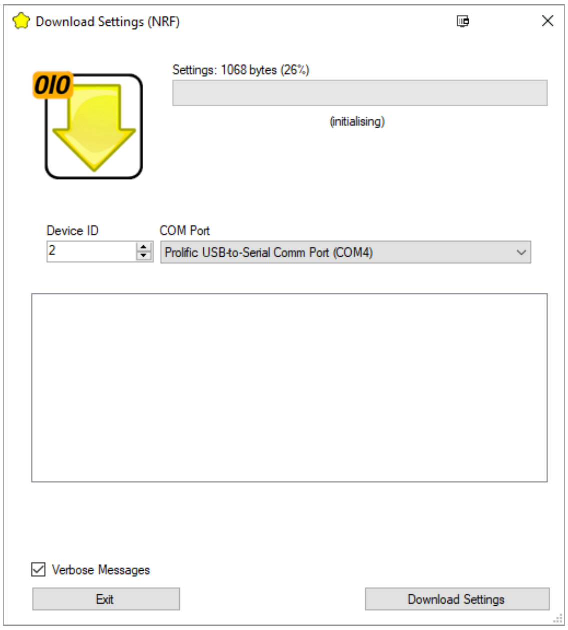
Once Windows reports everything is installed the Pilot™ device will be listed by name in the Bluetooth Devices control panel screen. Right-click on the icon and select “Properties” then the “Hardware” tab. Note: Microsoft make this harder by moving the options each version; on later versions of Windows (e.g. Windows 10/2018+) you may need to find a link on the Bluetooth screen that reads “More Bluetooth Options”. Make a note of the ***outgoing*** serial port listed, in this example it is COM34.




You may add as many Pilot™ devices as you wish. Note that Windows will maintain the serial port (COM number) for the device but if there are problems with connecting to the Pilot™ it may be worth rechecking the above window.

## Serial/Bluetooth Download Window

In the main editor go to “Transfer-> to Device over Serial/Bluetooth”. Set the COM Port to the corresponding Pilot™ Bluetooth serial port number as identified in the preceding “Adding Bluetooth Devices” section.




Click "Update Settings" and the software will proceed to establish a connection with the Pilot™, reboot the Pilot™ into its reprogramming mode, download the settings, validate the update and then wait for the command to restart the device with the new settings. Restarting the device can be done through the software or by repowering the Pilot™.

Due to the way Bluetooth and Windows interact there are sometimes delays and timeouts. The configuration software my wait for some time as it attempts to establish a connection. Some time-outs are actually expected. As the Pilot™ switches modes it must be reconnected to. If the COM Port was chosen correctly and the Pilot™ is powered and within range, the settings will download and a screen similar to the following will appear.

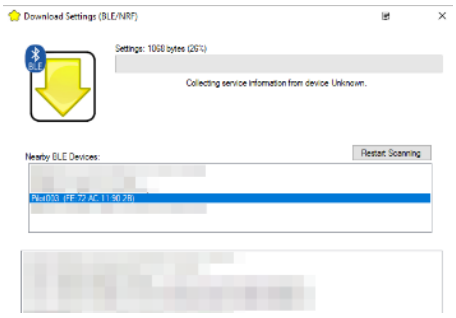
NOTE: Experience shows the best way to operate using Windows's Bluetooth is to open the download window, power cycle the Pilot and click download as soon as the Pilot's CPU light start blinking rapidly. This means the Pilot is first connected in the start-up phase and makes wireless connections from Windows more reliable.

Note that during reprogramming, and while the Pilot™ waits for a restart, the green CPU light will be flashing rapidly. This confirms it is in reprogramming mode.

## DOWNLOADING VIA BLUETOOTH LOW ENERGY (BLE)

For BLE based devices (e.g. Pilot-CAN2), go to "Transfer->to Device using Bluetooth LE". As soon as the Bluetooth LE download window opens, it will begin scanning for nearby devices.

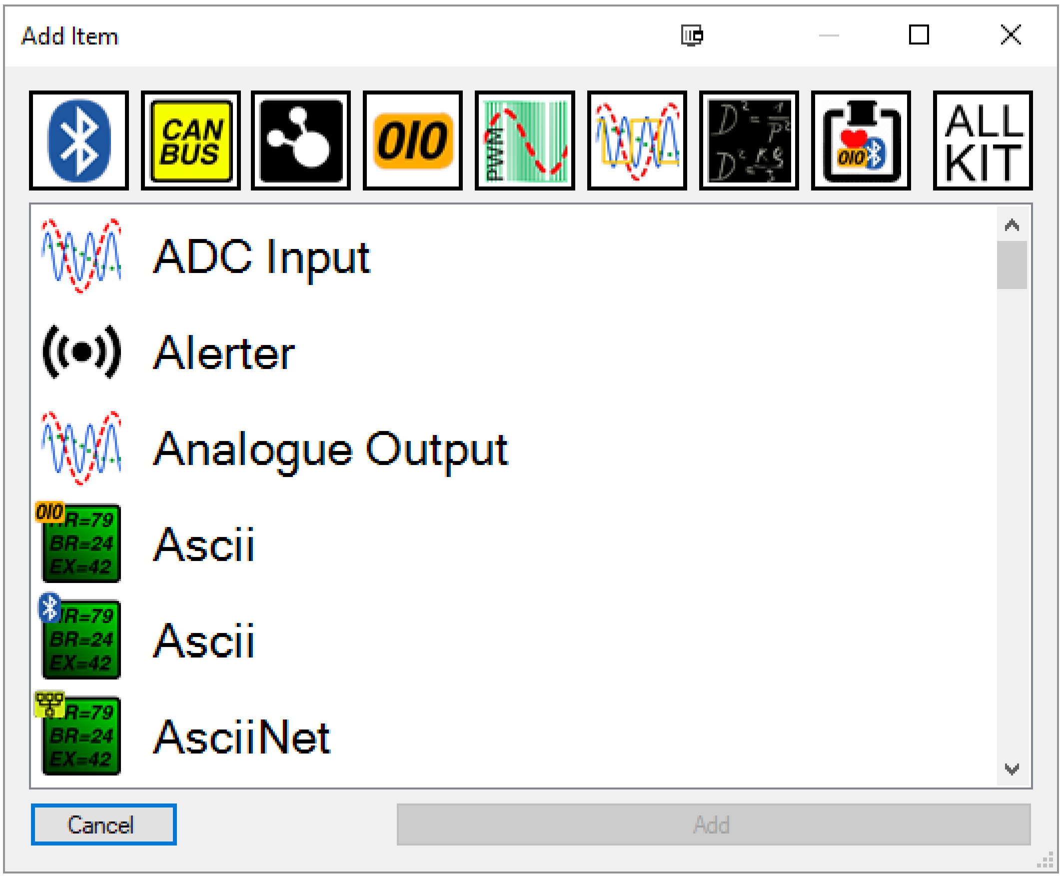
When you click on the required device then scanning stops. It can be restarted using the button. Once you are happy with your selection then click download, the device may need to reboot into the bootloader but this will be handled automatically.



# CREATING A NEW CONFIGURATION

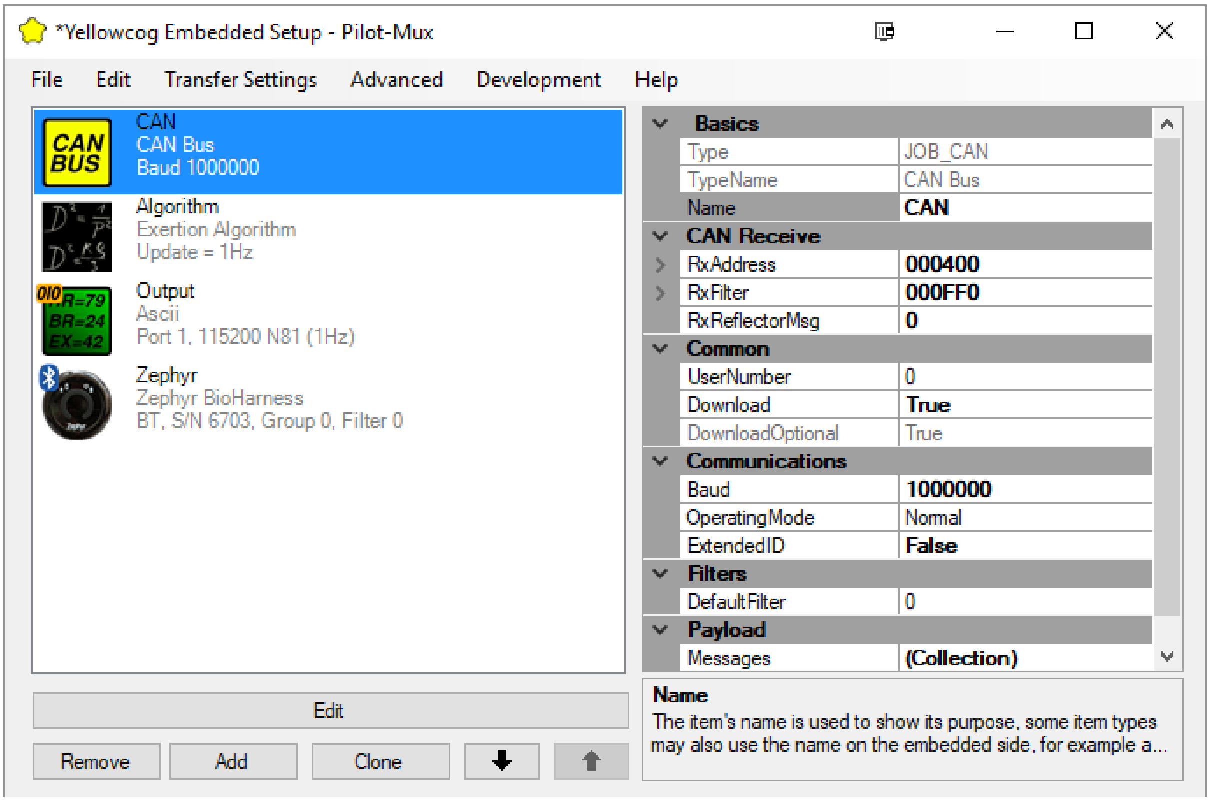
To create a new configuration, go to the File→New. This will create a blank configuration. As per best practice, this new configuration should be saved to a file name before adding any modules. Use “File→Save As…”.

The next step is to add a module. Click “Add” below the module list or go to menu Edit→Add and the following screen will appear. The icons along the top filter the list of modules to aid finding the required one. In the top right is the “All Kit” filter. This swaps between, when highlighted, all the available options and, when not highlighted, only those options available to you. The options available to you are those in the current Import File. The import file allows you to keep a record of the kit you actually have rather than entering details of the kit each time other changes are made.



Use the filters where necessary and high light the module you wish to add, then click 'Add'. The new module will now appear in the module list on the main screen and will be highlighted.

For example, a suitable setup would be to add a CAN module along with a Zephyr BioHarness, giving us the following:

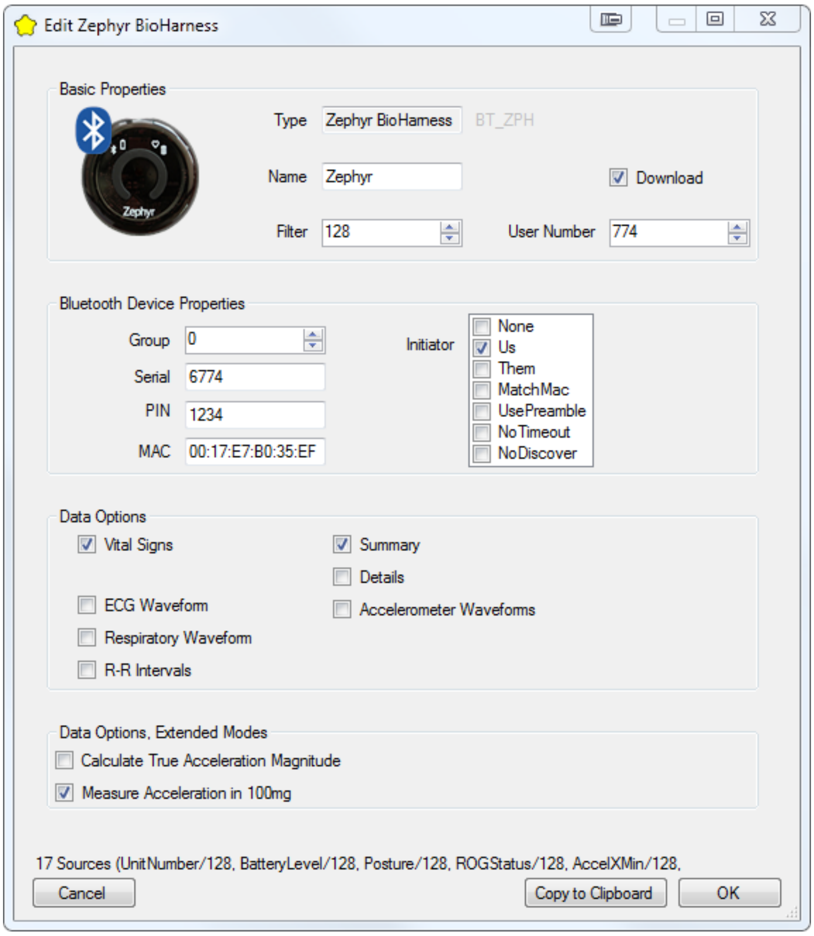



It is then necessary to configure each module so that they function as required.

We can see the new CAN item and the default, configured, properties on the right. Each module type is different, but all share a set of inherited properties that means that once the basics are understood, each extra module type only requires a little more understanding. Each module type will be dealt with in the next section.

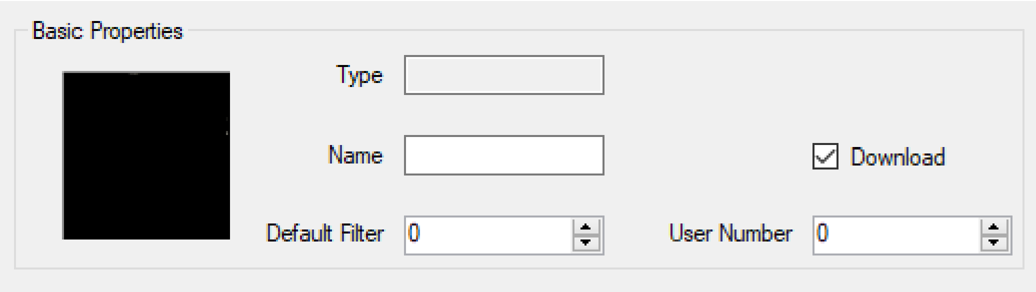
# MODULE CONFIGURATION

Below is an example of the configuration window for the Zephyr BioHarness module. Note that it is arranged into distinct, named panels such as "Basic Properties", "Bluetooth Device Properties" and so on.




Modules share many similarities. All modules share the “Basic Properties” configuration panel. Whereas, for example, all Bluetooth based modules will have the “Bluetooth Device Properties” panel.

## BASIC PROPERTIES




All modules have this panel. Options are:

- Type – is determined when first adding the module and governs which module properties can be configured.
- Name – used to identify the module in the module list but may also play a part on the embedded device, for example it may be required for it to appear in the outputted data.
- Default Filter – see Streams and Filters.
- User Number – This can be used to identify this module elsewhere in the system. For example, by outputting this number whenever the device is connected. In general it can be left as zero.

## PERIODIC TASK PROPERTIES

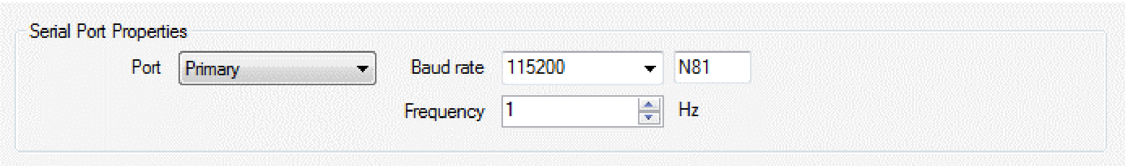

For items that do not share properties with the more numerous serial, ANT or Bluetooth types, this gives the single option of:

- Frequency – the rate in Hertz that the module will perform its operations, whether that is perform a calculation or some other task.

Note some modules will carry out some of their operations faster than this underlying rate. For example, a module may perform idle tasks at the rate specified here but may still react within microseconds to certain events and store these for processing at the periodic rate.

N.B. It is tempting to up the period frequency to "get the most" out of the system. Caution should be used since the Pilot has finite resources and making this value too high may negatively impact other modules. Best results are achieved by setting this to the *absolute minimum* required to achieve the given task.

## SERIAL PORT PROPERTIES

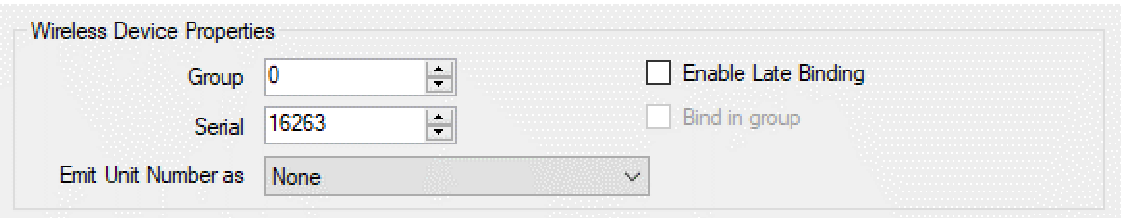



This panel is shared by all modules that input or output serial data. Options are:

- Port – almost always to be set as Primary since this is the port wired to the device's connector. The secondary or higher ports are only available in custom units.
- Baud rate – all standard rates are supported up to 115200. Contact support if other rates are required.
- Bit options – Reserved, currently only "N81" is supported. Contact support for information.
- Frequency – the rate in Hertz that the port will perform an update. An update may or may not result in data being transmitted or processed, depending on the protocols settings.

## WIRELESS DEVICE PROPERTIES

All wireless devices share some common properties.



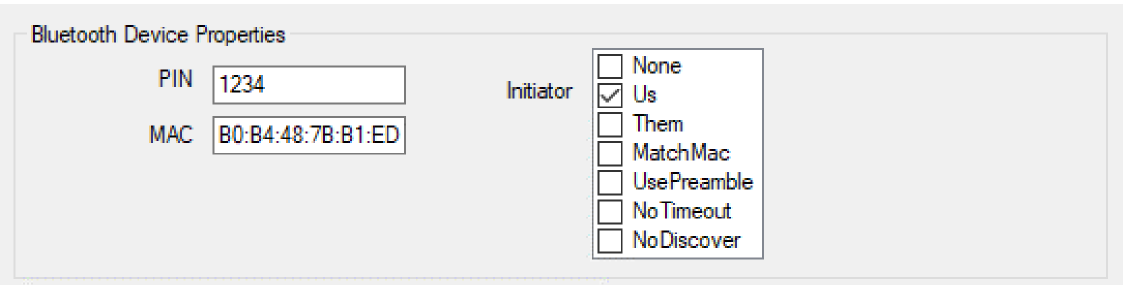

- Group – see Advanced Configuration Techniques.
- Enable Late Binding – see Advanced Configuration Techniques.
- Bind in group – see Advanced Configuration Techniques.
- Serial – the serial number of the peripheral. It is not always necessary to have this set correctly but it assists in identification and pairing.

- Emit Unit Number as – in order to identify a device in subsequent output data it is useful to have the module transmit a stream called Unit Number. The number can be one of:
  - Configured Serial Number – if this device is connected then this module will output whatever serial number you have entered into the config. If you have used a wildcard (serial number set to zero) then this option is not useful.
  - Unit Number – this is an arbitrary configured value for this module. This is useful if the proper serial number is not wanted in the output data e.g. you would like your devices called “1, 2, 3…” not their true serials which may be opaque e.g. “43252, 243352… etc”
  - Remote Serial Number – most devices will transmit their true serial number back, either via the transport protocol or in a separate message. As such, this option can be used with a configured serial number of zero but when the device connects the actual identity of the remote device is available. Note for ANT, this is the 16-bit serial number, for Bluetooth (+LE) this is the low 16-bits of the MAC address.

## BLUETOOTH DEVICES

There are to fundamental types: Bluetooth and Bluetooth Low Energy (BLE) and they generally compatible. In fact, worse, they are close enough that some devices may behave as though they are one or the other or both but will not actually allow the transfer of data. E.g. the device may show it has received a connection but will immediately close it. A few devices do genuinely support both protocols but even then they might not transmit the same data on both. It is important to know what the device really supports.

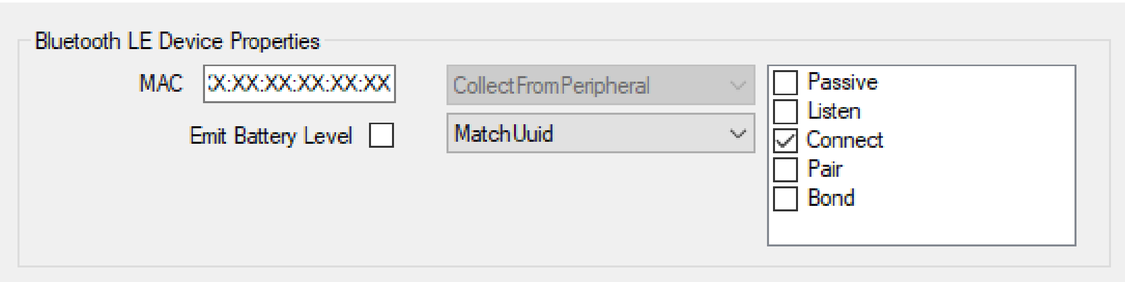
## BLUETOOTH PROPERTIES




Bluetooth devices share the same basic pattern. All require the fundamental communication details to allow a connection to be made between the Pilot™ device and the Bluetooth peripheral. Options are:

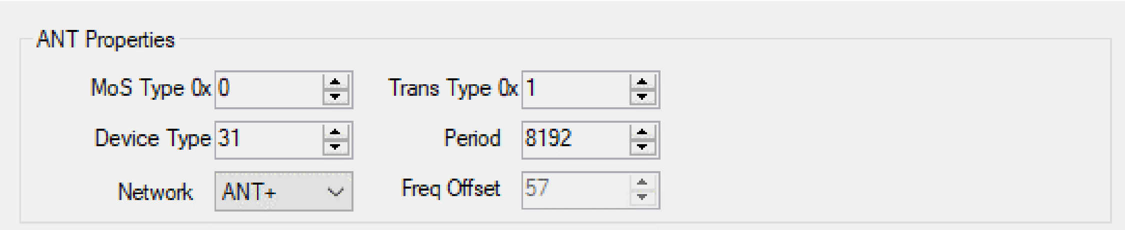
- PIN – specific to the protocol or individual piece of hardware.
- MAC – this is unique to every Bluetooth device and controls which will be accepted. The field can accept the wildcard character “*” or “X” which both represent “don’t care” i.e. if the MAC is set to “XX:XX:XX:XX:XX:01”, pairing will be attempted with any device whose address ends in “01”. It is recommended that full, explicit addresses are used whenever possible to avoid unwanted pairing attempts.
- Initiator – this not normally editable and controls who initiates the Bluetooth pairing/connection. Consider as a reserved feature.

## BLUETOOTH LOW ENERGY (BLE) PROPERTIES




- Connection Style – For the connection method (Passive, Listen etc), select only one option unless you are forcing more specific and technical behaviour. Generic devices should be tried with 'Connect' initially, then 'Pair' if they support it.
- MAC – this is unique to every Bluetooth LE device and controls which will be accepted. The field can accept the wildcard character “*” or “X” which both represent “don’t care” i.e. if the MAC is set to “XX:XX:XX:XX:XX:01”, pairing will be attempted with any device whose address ends in “01”. It is recommended that full, explicit addresses are used whenever possible to avoid unwanted pairing attempts. IMPORTANT: Some Bluetooth LE devices (e.g. iPhone) will select a new random address each time they enable BLE – this makes setting a MAC unusable. Wearable devices do not generally do this. The MAC is not needed if the Match Mode is set to UUID.
- Match Mode – BLE devices expose their UUID which is a generally accepted description of what features they support. Use UUID match to connect to e.g. heart rate devices, without having to give a MAC.
- Emit Battery Level – this will periodically transmit the battery level through the Pilot system if and only if the remote device correctly reports it.

## ANT PROPERTIES




Most of the properties in this section should be left at their defaults if you are using a consumer sensor. If you are providing connectivity to your own custom device then the properties will need to be altered.

Available options are:

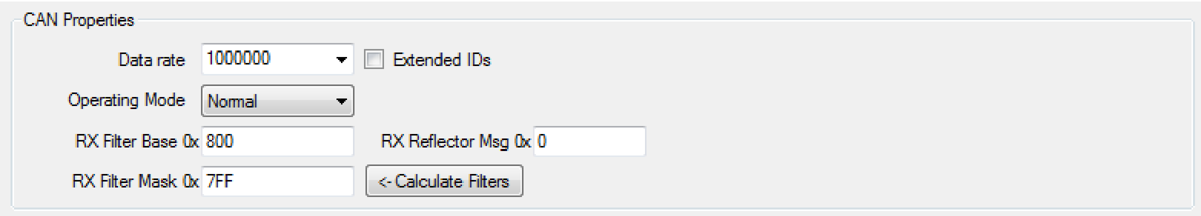
- Serial Number – set this to the serial number of the target peripheral or to zero for wildcard.

The remaining options should only be changed for custom sensors, standard sensors will cease to function if these options are changed; they are:

- MoS Type – Master or Slave type.
- Device Type – the number must match the peripherals device type.
- Network – Controls the Frequency Offset.
- Trans Type – this is specific to the way pairing is achieved.
- Period – the period between data exchanged in ticks of 1/32768 of a second. I.e. a period of 8102 means data is exchanged 4.04 times per second. Altering this from the sensors default will cause sporadic data transfer.
- Frequency Offset – defined by the network and is the offset from the base GHz frequency. Altering this will not alter the sensor frequency and will stop communications.

# CAN Module

## CAN Properties



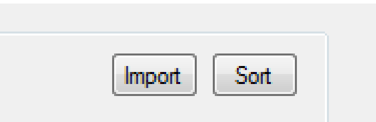

The CAN bus must be configured to match the requirements of the host CAN bus. The options are:

- Data rate – fundamental to interoperability, the rate, in Hertz, that communications occurs at. All common data rates are supported.
- Extended IDs – the CAN bus specification was altered to allow a larger range of addresses to be used. To use the extended mode this checkbox should be set. It is still common to use the original mode. The only way to discover the required mode is to check the mode of the host bus via its configuration software or specification documentation etc.
- Operating Mode – this should be set to “normal”. Other modes include “Listen”, if for example no data is to be transmitted.
- Rx Filter Base – Used to define the first message number in the range Pilot™ will receive (Rx) and process.
- Rx Filter Mask – Combined with the Rx Filter Base to control processing.

The Calculate Filters button uses the currently configured receive streams to calculate the optimum receive filter. Having the wrong filter would result in missing one or all incoming packets.

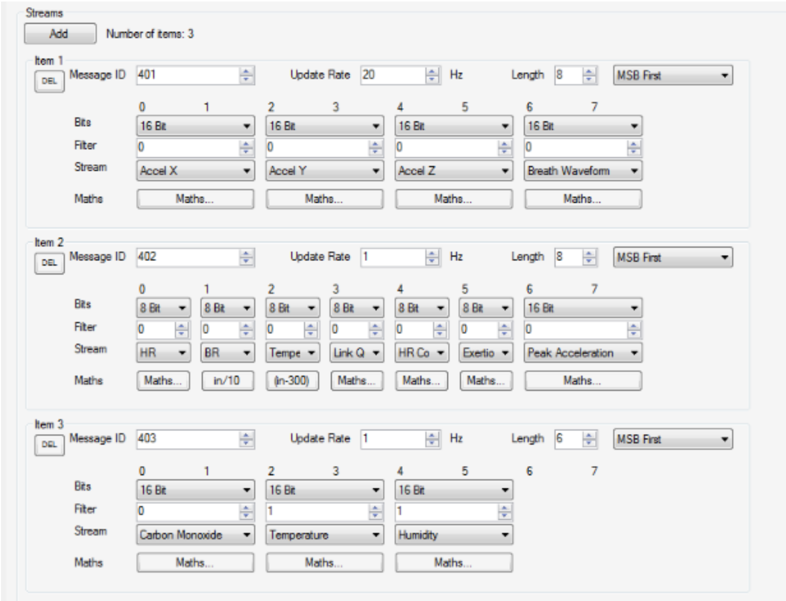
## Helpers

There are a few helper options presented in the top right of the streams section:



- Import – opens a File Open dialogue and will read in a DBC file. Not all settings are imported but the main set of streams and CAN formatting are imported. IMPORTANT: the stream names in the DBC file must conform to the naming conventions of the Pilot™ systems. i.e. Heart Rate will be mapped to the standard HEARTRATE stream.
- Sort – Sorts the messages into ascending CAN message ID order.

## CAN STREAMS




The output of the CAN module is in two tiers: messages and their payloads. See Streams and Filters for information on the fundamentals of data exchange.

Messages can be added with the main "Add" button. This will create a blank message that can then be configured to contain the required data. If a message is no longer required then the "DEL" button will remove that message.

The main options for each message are:

- Message ID – this is the ID that the host bus is expecting to receive. CAN is a prioritised protocol meaning that the value determines which messages are more important – the lower the number the higher the priority. This means that in a system with existing communications it is important to set the message priority such that it does not interfere with more important data. On a closed bus (no other peripherals other than host and Pilot™) it is acceptable to assign arbitrary numbers.

- Update Rate – the frequency that the Pilot™ outputs the message.
- Length – this is controlled by the payload. The length can be changed using this numeric box between the limits (zero – empty message, to 8 – the CAN maximum). This value will also update to complement changes made to the payload section below it.
- MSB/LSB First – defines the bit ordering. MSB puts the most significant bit of the number first i.e. the highest bit is the first to be transmitted onto the bus.
- Show Maths – if maths operations on each stream are required then checking this box will reveal the editors. Uncheck to save on screen space while editing.
- Bits – Each stream loaded into the message can be set to the required number of bits. Currently only multiples of 8 bits can be set. Set the required bits for each stream. If increasing the number of bits, the other streams will be pushed along the packet. If decreasing the number of bits the other streams will be pulled back. If you wish to reduce the width but leave the other packets it is necessary to configure in a VOID stream of the required width.
- Filter – See Streams and Filters.
- Stream – See Streams and Filters.
- Maths – click this to reveal the Maths pop-up window

## CAN/Serial Probe

This module allows some CAN to Serial to CAN messaging to be achieved. For example, it can turn the hardware into a Serial to CAN converter allowing any CAN message to be received or transmitted from a serial window on a host machine.

NOTE: The CAN/Serial Probe module MUST be configured to have the correct messages in the configuration. The required definition of the messages depends on the operational mode and facilitates the internal operations.

There are two operational modes:

### Serial ASCII Protocol ON

In this mode any CAN message can be transmitted or received. A custom ASCII protocol is used that has the following format:

```
#ID,DLC,A,B...<cr>
```

Where:

ID = CAN message ID

DLC = Length of CAN message (0 to 8)

A,B,… = individual data bytes

So, for example, sending “100,2,1,2” would transmit a CAN message with ID 0x100, and two data bytes being 0x01 and 0x02.

In order to transmit on CAN you must configure a TX message in the configuration – because any message ID can be sent, with any payload, the configuration of the TX message should be left at ID=0x0 with no data defined.

In order to receive from the CAN bus a RX message must be defined configuration – because any message ID can be received, with any payload, the configuration of the RX message should be left at ID=0x0 with no data defined.

### SERIAL ASCII PROTOCOL OFF

In this mode, any character received on the serial port will be encoded into a CAN message. If you send the string YELLOW into the serial port then a CAN message will be emitted as:

[TX:ID][6][‘Y’][‘E’][‘L’][‘L’][‘O’][‘W’]

If you send “YELLOWCOG” there will not be enough room in one standard eight-byte CAN message so two messages will be emitted, as such:

[TX:ID][8][‘Y’][‘E’][‘L’][‘L’][‘O’][‘W’][‘C’][‘O’]

[TX:ID][1][‘G’]

The exact way the message is split will depend on the (unpredictable) timing of the Pilot hardware. As long as the data rates are sustainable then no byte will be missed.

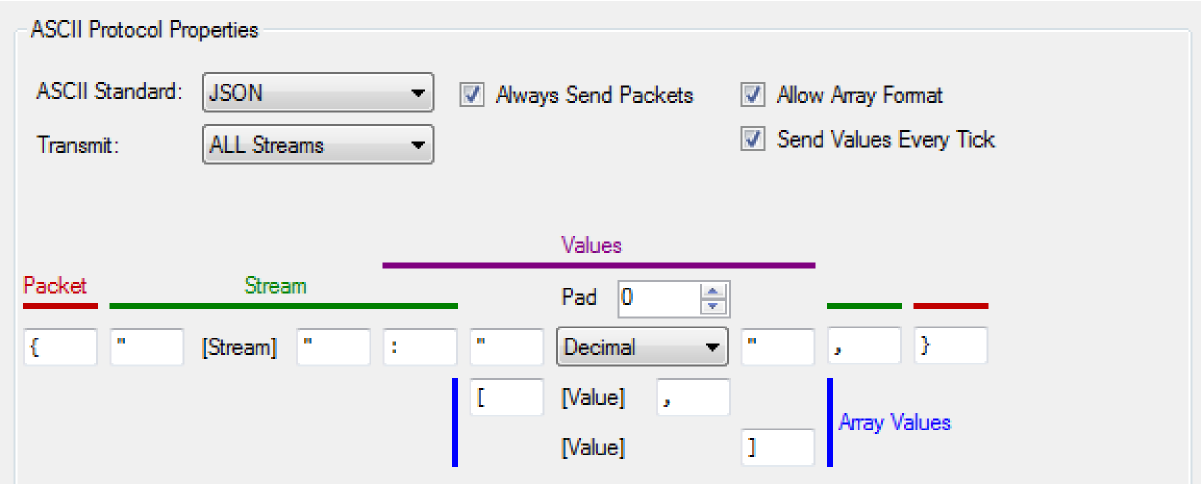
For received CAN messages the direction is reversed. Any data received as the payload on the configured receive CAN ID will be transmitted out the serial port as the same raw binary data. e.g. if the CAN message is comprised of ASCII viewable characters then you will get out will be readable text.

In order to transmit on CAN you must configure a TX message in the configuration – the ID should be the CAN ID you want the raw data mirrored to, the payload should be empty.

In order to receive from the CAN bus a RX message must be defined configuration – the ID should be the CAN ID you want the raw data mirrored from, the payload should be empty.

## ASCII Module

### ASCII Protocol Properties



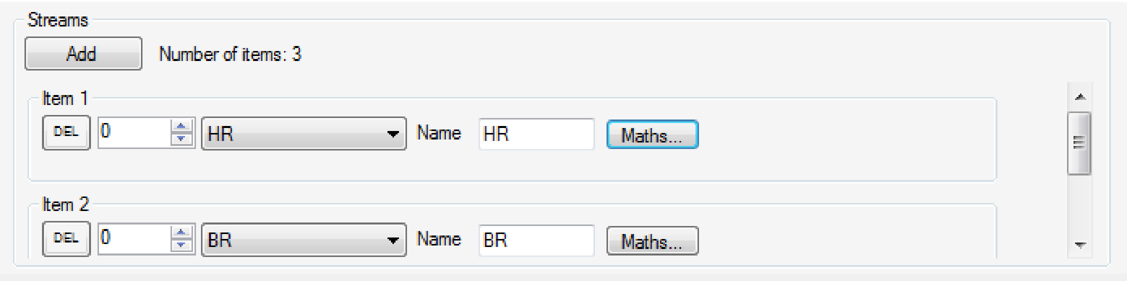

The ASCII protocol provides a way of quickly configuring human-readable serial data. The main options are:

- ASCII Standard – some built in standards are provided for quick configuration. JSON is the default and provides industry compatible and readable data output.
- Always Send Packets – if ticked, the protocol will produce a data packet even if there is no data to transmit. For example, in JSON this would mean you would get the open/close brackets each tick even if they contain no data.
- Allow Array Format – higher frequency data can be transmitted in a block, an array. If this is not desired or permitted by the receiver, uncheck this box.
- Transmit: All Streams – when checked, the protocol will transmit ALL streams available in the system. This could result in a large amount of data. Using this option saves having to individually configure each stream. Any configured streams will override the default output. Configured streams can either specify the exact Filtered Stream (and therefore only apply to it) or use the Late Binding Flag (and apply to all matching streams but a wildcard filter).
- Send Values Every Tick – when checked, the current value will be resent even if no new value has been acquired by the system.

- Packet (red) – defines the character string used to identify the start and end of the packet.
- Stream (green) – defines the character strings used to delineate the stream identifier and stream/value separator.
- Values (purple) – defines the numerical padding (e.g. with padding of four the value twelve is transmitted as "0012"), the format of the number (decimal "12", hexadecimal "AB" or Float "12.5") and the character strings used to delineate the value.
- Array Values (blue) – defines how multiple values will be combined and what contains the array elements.

The feedback panel shows an example of how data will be transmitted with the current settings (it does not reflect the actual streams that may be configured or available).

### STREAMS




This provides a list of all the streams to transmit. NOTE: if no streams are configured the system will default to transmitting all streams available. Since there is no per-stream configuration it is not possible to name or apply maths to each stream.

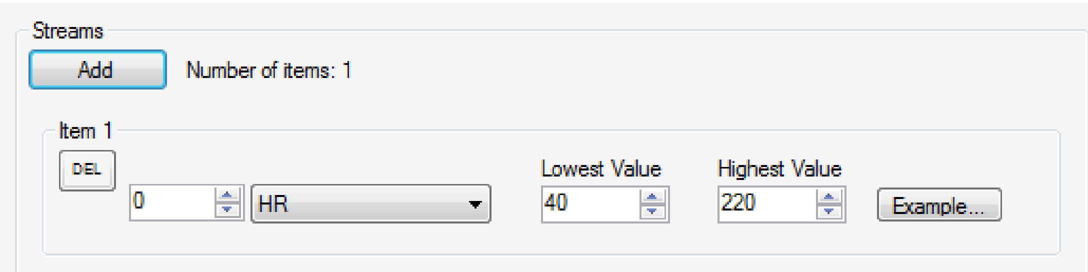
To add streams, click "Add". To delete existing streams, click the "DEL" by the unwanted stream. Stream options are:

- Filter – unlabelled numerical box to the left of the stream combo. See Streams and Filters.
- Stream – see Streams and Filters.
- Name – this is the name that will output whenever data is transmitted for this stream.
- Maths – click this to reveal the Maths pop-up window:

## Analogue Output Module

The analogue output option requires a special version of Pilot hardware fitted with a Digital to Analogue Converter daughter board or the standalone Pilot™-OVR. Standard MUX versions do not support analogue; you must have a module labelled as PilotPlus-DAC or Pilot™-OVR to use these features.

### Streams




Eight output streams are available from the Pilot-DAC. Each numbered item is configured in the same way and each relates to the numbered output of the Pilot-DAC connector respectively.

There are three main values to be set:

- Update frequency – set in the periodic properties, 18Hz is recommended.
- Filter/stream pair – this must be set to the type of data required for this output.
- Lowest Value – stream values at or below this limit will result in a zero volt output level.
- Highest Value – stream values at or above this limit will result in the maximum voltage output, which by default is 3.3volts.

### Stream/Voltage Formulae

The formula for calculating the output voltage for a given stream value is:

$$V_{out} = V_{max} \cdot \frac{S_{now} - S_{lo}}{S_{hi} - S_{lo}}$$

Where:

$S_{now}$ is the current stream value.

$V_{out}$ is the current output voltage.
$V_{max}$ is the maximum voltage, which is always 3.3 Volts.
$S_{lo}$ is the stream's Lowest Value
$S_{hi}$ is the stream's Highest Value.

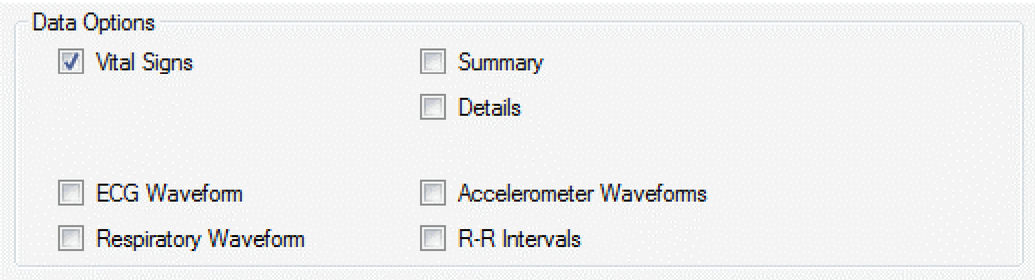
The formula for calculating the stream's value from a measured voltage is:

$$S_{now} = \left\{ \frac{V_{out}.\,(S_{hi} - S_{lo})}{V_{max}} \right\} + S_{lo}$$

### TECHNICAL NOTE

The update frequency controls how often the DAC's eight channels are updated. Setting too low a value may mean that brief transients are lost. Setting a too high value may mean that the DAC updates its output faster than the incoming data. Since data comes in discrete packets from most devices then this has the effect of producing bursts of varying data with stead-state periods in between.

## ZEPHYR BIOHARNESS




The Zephyr BioHarness can be instructed to send varying amount of data. You may wish to set fewer options if they will not be output and would therefore be redundant. See BioHarness documentation for exact data meaning. Options are:

- Vital Signs – heart rate, breathing rate, activity, peak acceleration, core temperature and heart rate variability.
- ECG Waveform – the 250Hz ECG waveform.
- Respiratory Waveform – the approx. 25Hz chest expansion waveform.
- Summary – unit number, battery level, posture, ROG Status, RSSI and Accelerometer XYZ minimums and maximums.

- Details – time stamp, battery voltage, breathing amplitude, breath noise, BR confidence, ECG amplitude, ECG noise, system confidence, GSR, device temperature, status info, link quality and transmit power.
- Accelerometer waveforms – the XYZ waveforms, when "Use 100mg" is set then these values are scaled to those units; with the option unset then the units are from its internal accelerometer. With "Use 100mg" set then a BioModule at rest should show near zero on two channels and +/- 10 (0x0A) on the third i.e. 0, 0, 1g.
- R-R Intervals – R to R intervals are reported as a alternating sign record of the precise RR intervals. The sign shows when a RR value is genuinely new. See Zephyr documentation for details.

### CorScience EKG Blue

There are no additional options required for this device.

### Nonin

There are no additional options required for this device.

## GPS (NMEA)

There are no additional options required for this device.

## GPS (XML / GPX)

There are no additional options required for this device.

# LEGAL

## LIMITED WARRANTY

Yellowcog Limited warrants to the consumer/purchaser of its hardware that the product will be free from defects in material or workmanship for one year from the date of purchase. Please keep proof of purchase. Motorsport is an aggressive environment and failures will occur that could not be countered even with best practices in manufacturing. Yellowcog's warranty only applies where defects can be shown to be caused by substandard engineering and production. Over-voltages and other destructive actions are not covered by warranty.

## RECYCLING AND DISPOSAL

Once Waste Electrical & Electronic Equipment (WEEE) reaches the end of its useful life it should be treated, recycled and disposed of in an environmentally sound manner in accordance with the WEEE Regulations. This means that it should be treated by an Approved Authorised Treatment Facility (AATF). It is the responsibility of the producer of WEEE to finance the costs of this unless an alternative arrangement has been made. Yellowcog has made arrangements to collect and treat the Electrical & Electronic Equipment that we have supplied to you once the equipment has come to the end of its life.

If you require free collection of yellowcog supplied WEEE from your business, please contact yellowcog at:

WEEE Collection
Yellowcog
The Aviary
New Odiham Road
Alton
Hampshire
UK
Tel: 07525133409
E-mail: weee@yellowcog.com

# NOTICES