CEX PDI Builder

Release 6.8.73

Embention

2023-11-20

CONTENTS

1		k Start
	1.1	Download
	1.2	Installation
2	Conf	iguration
	2.1	CEX
		2.1.1 CEX Base
		2.1.2 Status
	2.2	Sensors
		2.2.1 RPM
		2.2.2 Lidar
	2.3	
		2.3.1 GPIO
		2.3.2 PWM
		2.3.3 I/O Setup
		2.3.3.1 Tunnel
		2.3.3.2 Custom Messages
		2.3.3.3 JETI box
		2.3.4 CAN I/O
		2.3.4.1 Configuration
		2.3.4.2 CAN Telemetry
		2.3.5 Digital Input 2
		2.3.6 SCI
		2.3.7 CAN Setup
	2.4	Communications
	2.4	2.4.1 Ports 3
	2.5	Stick
	2.3	
	26	1
	2.6	Devices
		2.6.1 Jetibox
		2.6.2 Scorpion tribunus
	2.7	Arbitration
		2.7.1 Arbitration
		2.7.2 Config
		2.7.2.1 Absolute Arbitration Variables
		2.7.2.2 Relative Arbitration Variables
		2.7.2.3 Arbitration Example

		2.7.2.4 Config menu
		2.7.3 CAN Setup
3	Inteo	gration examples 57
•	3.1	CAN Isolator
	0.11	3.1.1 Filtered CAN subnet
		3.1.2 CAN tunnel
	3.2	Reading Arbitration Messages 65
	3.3	External devices
		3.3.1 Jetibox
		3.3.2 Scorpion tribunus
		3.3.3 Veronte products
		3.3.3.1 CAN communication
		3.3.3.1.1 CEX PDI Builder side
		3.3.3.1.2 1x PDI Builder side
		3.3.3.2 Commanding/Reading PWMs
		3.3.3.2.1 1x PDI Builder side
		3.3.3.2.2 CEX PDI Builder side
		3.3.3.3 Connection with Autopilot 1x via CAN
		3.3.3.4 Connection with Autopilot 4x via CAN
		3.3.3.4.1 CAN Reception IDs
		3.3.3.4.2 CAN I/O Interconnections
		3.3.3.5 GPIO Command
		3.3.3.5.1 1x PDI Builder side
		3.3.3.5.2 CEX PDI Builder side
		3.3.3.6 Reading/Sending RPMs
		3.3.3.6.1 CEX PDI Builder side
		3.3.3.6.2 1x PDI Builder side
		3.3.3.7 Serial communication
		3.3.3.7.1 CEX PDI Builder side
		3.3.3.7.2 1x PDI Builder side
4		bleshooting 109
	4.1	Migrate configuration
5	FAQ	111
	5.1	How to calculate a mask

CEXBULDER

CEX PDI Builder is an application for modifying, generating and uploading CEX PDI files.

	any Embention user manual.	
Veronte Link × + C		× v – @ @ ⊉ @ 9 46 ∏ 4
		🛱 Home Version-6.8.27 v Languages-EN v Download v 🌐
٩	Docs » Veronte Link	
Veronte Link Software installation How to use Veronte Link Sessions Troubleshooting Integration examples COM port configuration	Veronte Link Veronte Link interconnects multiple control stations and autopilot un	nits, so they can operate simultaneously.

CHAPTER

QUICK START

CEX PDI Builder is the main configuration tool to adapt a **CEX** to be suitable for a specific system, being its main goal to expand the available communication protocols. **CEX PDI Builder** includes:

- Communications: Through general purpose CAN bus, inputs and outputs and PWMs.
- Stick control signal management: Compatible with **Stick Expander**, Futaba, Jeti, FrSky and TBS. It includes custom configuration for other sticks.
- Arbitration: CEX is able to send PWM signals using arbitration in the same way Veronte Autopilot 4x does.

Once CEX has been detected on Veronte Link, install CEX PDI Builder.

1.1 Download

Once the CEX has been purchased, a GitHub release should be created for the customer with the application.

To access to the release and download the software, read the Releases section of the **Joint Collaboration Framework** manual.

1.2 Installation

To install CEX PDI Builder on Windows just execute "CEXPDIBuilder.exe" and follow the Setup Wizard instructions.

Warning: If users have any problems with the installation, please disable the antivirus and the Windows firewall. Disabling the antivirus depends on the antivirus software. Administrator rights are needed.

To disable the firewall, go to "Control Panel" \rightarrow "System and Security" \rightarrow "Windows Defender Firewall" and then, click on "Turn windows Defender Firewall on or off".

	Windows Defender Firewall					
<form></form>	– → ∨ ↑ 🔐 « System and	I Security > Windows Defender Firewall v 🖑 Search Control Panel				
<form></form>		Help protect your PC with Windows Defender Firewall				
Allow any pure failing in the pure fail in the pure f	Control Panel Home					
Treveil Come contained setting: Treveil de data Ancres de data Ancres de data Treveil de data Ancres de data Treveil de data Ancres						
Cancelle control content in the series of the series o						
Tireval no off Record eduals Avanced settings Toublehoot my network	Change notification settings	Private networks Connected V				
Rescred details Advanced settings Toublechood my network Cutomics Setting:		Guest or public networks Not connected 📀				
troublehood my network Fig. 1: Windows Defender Firewall Curronies estings ()() () ()() ()						
troublebloot my network Fig. 1: Windows Defender Firewall (v users settings v v v v v v v v v v v v v v v v v v v						
Cutomise Settings						
→						
Customise settings for each type of network You can modify the firewall settings for each type of network that you use. Private network settings Image: Setting in the interval						
You can modify the firewall settings Image: Setting Seting Seting Setting Setting Set	→ ヾ 个 🔮 > Control Pa	nel > System and Security > Windows Defender Firewall > Customise Settings	~ Ū	Search Con	trol Panel	,
You can modify the firewall settings Image: Setting Seting Seting Setting Setting Set		Customise settings for each type of network				
Private network settings Itum on Windows Defender Firewall Block all incoming connections, including those in the list of allowed applications Notify me when Windows Defender Firewall blocks a new app Windows Defender Firewall Itum on Windows Defender Firewall (not recommended) Public network settings Itum on Windows Defender Firewall Block all incoming connections, including those in the list of allowed applications Notify me when Windows Defender Firewall blocks a new app Windows Defender Firewall Block all incoming connections, including those in the list of allowed applications Notify me when Windows Defender Firewall blocks a new app Wotify me when Windows Defender Firewall (not recommended) 						
 Block all incoming connections, including those in the list of allowed applications ⊘ Notify me when Windows Defender Firewall blocks a new app ② Or un of Windows Defender Firewall (not recommended) > Or un on Windows Defender Firewall > Block all incoming connections, including those in the list of allowed applications > ○ Notify me when Windows Defender Firewall blocks a new app > ③ Or un off Windows Defender Firewall blocks a new app > ③ Or un off Windows Defender Firewall blocks a new app > ③ Or un off Windows Defender Firewall blocks a new app > ③ Or un off Windows Defender Firewall (not recommended) 						
Notify me when Windows Defender Firewall blocks a new app Image: Second Seco		Turn on Windows Defender Firewall				
 						
Public network settings I'um on Windows Defender Firewall Block all incoming connections, including those in the list of allowed applications Notify me when Windows Defender Firewall blocks a new app Wolf me when Windows Defender Firewall blocks a new app I'um off Windows Defender Firewall (not recommended) 						
 Turn on Windows Defender Firewall Block all incoming connections, including those in the list of allowed applications Notify me when Windows Defender Firewall blocks a new app Turn off Windows Defender Firewall (not recommended) 		• •				
□ Block all incoming connections, including those in the list of allowed applications □ Notify me when Windows Defender Firewall blocks a new app Image: Control of Windows Defender Firewall (not recommended)						
✓ Notify me when Windows Defender Firewall blocks a new app Image: Comparison of Windows Defender Firewall (not recommended)						
OK Cancel		🚫				
OK Cancel		• •				
OK Cancel						
OK Cancel						
OK Cancel						
OK Cancel						
OK Cancel						
OK Cancel						
OK Cancel						
		OK Cancel				

CHAPTER

TWO

CONFIGURATION



2.1.1 CEX Base

CEX is able to send information about its version and Jeti telemetry from devices connected to CEX, this is the "status message". It is possible to define the CAN Id used for those messages.

Enable the **Extended** checkbox to use the extended CAN protocol (with a 29-bit identifier), disable it to use the standard protocol (11-bit identifier).

CEX2PDI Builder - CEX2 45010 - CONNEC	ED	X
	() CEX	8 2 2 9
Cex Base Status Image: Sta	Enable Jeti <u>TX CAN ID</u> Extended CAN 33	

Fig. 1: Cex Base panel

2.1.2 Status

This option enables the periodic sending of the status message that Veronte Link uses to recognise CEX.

CEX2PD	l Builder - CEX2 45010 - CONNECT	ED	– ×
		() CEX	8 2 2 9
O	Cex Base	Enable VCP Status Message	
	Status		
2		Period 1.0 s	
•/*			
₽ 0 8 ® ∲ % C			
3			
.			

Fig. 2: Status panel

Period: Enter a desired period to send repeatedly the status message.

Note: VCP is the Veronte Communication Protocol. To know more, read the VCP user manual.



2.2.1 RPM

CEX can measure RPMs by reading froms up to four input sources:

CEX2PDI Buik	der - CEX2 45010 - CONNECTE	D			- ×
		ີ Sensor	s		8 2 2 9
	RPM	RPM 1 RPM 2 RPM 3 RPM 4	4		
	Lidar	Units	6.2831855	Custom	•
		✓ Average	5		
		Minimum	2.0E-4	s	
5 2		Maximum	0.5	s	
$\ \ \bigcirc$					

Fig. 3: RPM panel

- Units: Sensor conversion factor. It can be Custom, Radians per pulse and Pulses per cycle.
- Average: It is a filter to avoid voltage spikes. The readout of the pulse can be filtered for the output to be an average. The amount of measurements to do the average needs to be specified.
- **Minimum**: Here the minimum expected pulse period needs to be specified. This will discard spurius pulses (e.g. induced by EMI) which are smaller than this minimum pulse.
- **Maximum**: The maximum period of time allowed without capturing. If no incoming pulse is received for more than this time, the output RPMs will be 0.

A configuration example to send RPMs can be found in *Reading/Sending RPMs - Integration examples* section.

2.2.2 Lidar

The I2C bus allows the connection of several devices with different addresses to the same line via master-slave communication. At this moment, CEX supports the following Lidar devices:

- Garmin LIDAR-Lite v3: Optical distance measurement sensor with a range from 5 cm to 40 m.
- SF11 Lidar: Long range laser altimeter. Supported SF11/B and SF11/C with a range of maximum 50 m and 0.2 m to 120 m respectively.
- SF20 Lidar: OEM laser altimeter module. Supported SF20/C with a range of 0.2 m to 100 m.

CEX allows up to 5 Lidar devices to be connected to the system at the same time. The configuration menu can be seen below:

CEX2PDI Builder - CEX2 45010 - CONNECTE	D	- ×
	Sensors	8 2 8 9
RPM	Lidar I2C	
RPM Lidar Image: Constraint of the second s	✓ Lidar 1 Garmin Lida ▼ Address 0 dec Digital filter	
	Lidar 2 Garmin Lidar Lite v3 SF11 Lidar dress 0 dec Digital filter	
	Lidar 3 SF20 Lidar dress 0 dec Digital filter	
	Lidar 4 Garmin Lida Address 0 dec Digital filter	
	Lidar 5 Garmin Lida Address 0 dec Digital filter	
E		

Fig. 4: Lidar panel

After enabling the needed number of Lidar devices, configurable parameters are:

- Type of Lidar.
- Address: With an accepted value between 16 239, this is the origin address from the Lidar being configured.
- **Digital filter**: Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**.

Note: I2C address will be different for different devices make sure to define it properly by checking the manufacturer documentation.



2.3.1 GPIO

In this window, each individual GPIO (General Purpose Input/Output) behavior can be configured:

		🕂 Input /	Output		8 2 0	
GPIO	Signal		2) 10(3)	Pull-up 4	Function	5) a
PWM	I/01	GPIO 24	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
I/O Setup	1/02	GPIO 25	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
CAN I/O	1/03	GPIO 26	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
Digital Input	1/04	GPIO 27	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
SCI	PWM 1	GPIO 0	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
CAN Setup	PWM 2	GPIO 1	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 3	GPIO 2	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 4	GPIO 3	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 5	GPIO 4	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 6	GPIO 5	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 7	GPIO 6	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 8	GPIO 7	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync

Fig. 5: GPIO panel

- 1. Signal: Pin ID as described in Hardware installation Pinout section of the CEX Hardware Manual.
- 2. GPIOId: GPIO ID of the microcontroller.
- 3. IO: Define GPIO as an input or output.
- 4. **Pull-up**: Enable or disable the pull-up resistance.
- 5. **Function**: Mux 0/GPIO: GPIO, Mux 1: PWM, Mux 2 or Mux 3. These are the different functionalities that the GPIO can have, depending on the multiplexer.

Note: When users set **Function** to "Mux 1", it indicates that the corresponding pin is disabled as GPIO and enabled as PWM. Consequently, the **Enable** checkbox in the *PWM menu* for that pin should be activated automatically.

Warning: Check that the corresponding Enable checkbox in the *PWM menu* is changed.

- 6. **Qsel**: This is the "input qualification", it is used to control how the value of a GPIO is evaluated. The available options are:
 - **Sync**: The value is taken as the time it is checked (synchronously). This is the default mode of all GPIO pins.
 - 3 Samples: The value is checked 3 times and the value is only changed when the 3 times are the same.

- 6 Samples: Same as 3 samples, but checking 6 times instead of 3.
- ASync: No checks are performed. It is used when it is not used as GPIO.

2.3.2 PWM

In this panel each PWM can be configured:

CEX2PDI Builder - CEX2 45010 - CONNECTE	D	- x
	• 🖕 Input / Output	
GPIO PWM 1 N I/O Setup 2 CAN I/O 2 Digital Input 3 SCI CAN Setup 4 5 6	PWM 1 Imput / Output PWM 1 Sub id 0 Frequency 50 Mode Time Min 9.0E-4 Max 0.0021 PWM 2 PWM 3 PWM 4 PWM 5 PWM 6 PWM 7 PWM 8 PWM 8	

Fig. 6: PWM panel

Note: PWMs in CEX work in normalized mode, so when the input value is 0 the output value will be the minimum configured, and when the input value is 4095 (12 bits all with ones), the output will be the maximum configured. This approach allows usage of the maximum resolution for the commanded value.

The configuration parameters are:

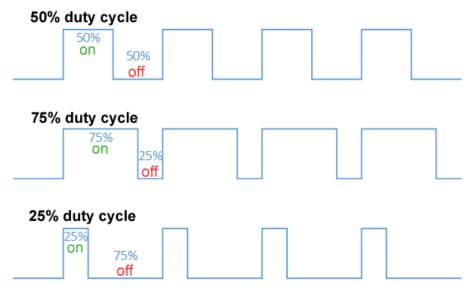
1. Enable: Defines if the PWM is enabled or not.

Note: This checkbox disables the pin as GPIO and enables it as PWM. Hence, in the *GPIO menu* the "Function" parameter shall change to "**Mux 1**".

2. **Sub id**: Identifies the sub-id of the PWM and, according to this value, determines the command to be used, using one type of CAN message or another.

- 3. Frequency: PWM output frequency. This determines the period of the pulses sent by the CEX.
- 4. Timeout: If a PWM message is not received in less than this time, the PWM will output the start value.
- 5. Start value: Value used before any PWM message arrives and on timeout.
- 6. Pulse source ID: PWM input ID [0,3], defined in the *Digital Input* panel.
- 7. Active High: Polarity high or low (high if enabled).
- 8. Mode: The avilable options are Time and Duty cycle.
 - Time: The values indicated in Min and Max parameters are expressed in time units.
 - **Duty cycle**: This option is a a different way of indicating the pulse width. Now the value indicated in **Min** and **Max** parameters is a percentage which corresponds to the relation between the pulse width over the total period of the sent signal.

So a 100% duty cycle will correspond to a signal with a constant value of 1, while a 0% duty cycle implies a constant signal with value 0. Between this two extremes, the pulse width can vary as in the examples shown in the following figure.





Note: Duty cycle percentages can be expressed in percent and per unit.

- Min: This parameter is the pulse width value that will make the servo/actuator go to its lowest position. It will be the output when the PWM message specifies 0.
- Max: This parameter is the pulse width value that will make the servo/actuator go to its highest position. It will be the output when the PWM message specifies 4095.

An example of reading PWM can be found in Commanding/Reading PWMs - Integration examples section.

2.3.3 I/O Setup

In this panel the user can establish the relationship between a determined signal with a I/O port. This allows users to configure serial inputs and outputs: external sensors, custom messages, etc.

	€ √ →	Input / Output			7 0
GPIO	Configuration				
PWM	Priority	Producer		Consumer	
I/O Setup	High C	UART-A / RS232-A	\rightarrow	Commgr port 2	00
CAN I/O Digital Input	High 🔯	UART-B / RS232-B)→	Commgr port 3	Q0
SCI	High 🔯	UART-C / RS485	\rightarrow	None	Q0
CAN Setup	High 🔯	CAN to Serial 1	→	Commgr port 1	Q0
	High 🔯	CAN to Serial 2	\rightarrow	None	Q0
	High 🔯	CAN to Serial 3	\rightarrow	None	¢¢₀
	High 🔯	Commgr port 1	→	Serial to CAN 1	¢¢
	High 🔯	Commgr port 2]→	UART-A / RS232-A	¢¢
	High 🔯	Commgrf port 3]→	UART-B / RS232-B	¢¢
	High 🔯	Commgr port 4	\rightarrow	None	Q0
	High 🔯	Commgr port 5	\rightarrow	None	Q _0
	High 🔯	Commgr port 6	\rightarrow	None	Q0
	High 🔯	Tunnel 1	\rightarrow	None	Q0
	High 🔯	Tunnel 2	\rightarrow	None	Q0
	High 🔯	Tunnel 3	\rightarrow	None	Q0
	High 🕵	CAN wrapper 0 for serial tran	\rightarrow	None	Q0

Fig. 8: I/O Setup panel

- **Priority:** Connections between I/O ports can be marked with **high priority** with this checkbox. If enabled, they will **run at high frequency: 1000 Hz**.
- **Producer:** Functions for creating and sending messages.
- Consumer: Functions for receiving and parsing messages

Firstly, users have to configure the **Producer** selecting the I/O port or information to use. Later, users have to configure the **Consumer** by clicking on an element, a new window will be displayed to select an item. The relationship between them can be unidirectional (Bind \rightarrow) or bidirectional (Bind Bidirectional \leftrightarrow), the last enables a port to receive or send information.

GPIO	Configuration					
PWM	Priority	Producer	_	Consumer		
I/O Setup	High 🔅	UART-A / RS232-A	>	Commgr port 2	Bind	▶ UART-B
CAN I/O	High 🔯	UART-B / RS232-B		Commgr port 3	Bind Bidirectional	UART-C
Digital Input	High 🔯	UART-C / RS485		None	Clear	Serial to CAN 1
SCI	High 🕵	CAN to Serial 1		Commgr port 1	08	Serial to CAN 2
CAN Setup	High 0	CAN to Serial 2		None	08	Serial to CAN 3
						Tribunus ESC
	High 🕵	CAN to Serial 3		None	06	Lift - MCU
	High 🕵	Commgr port 1		Serial to CAN 1	¢\$	JETI box
	High 🕵	Commgr port 2		UART-A / RS232-A	¢\$	JETI telemetry 1
	High 🕵	Commgrf port 3		UART-B / RS232-B	08	JETI telemetry 2
	High 🕵	Commgr port 4		None	08	Commgr port 1
	High 🕵	Commgr port 5		None	00	Commgr port 3
						Commgr port 4
	High 🕸	Commgr port 6		None	06	Commgr port 5
	High 👯	Tunnel 1		None	08	Commgr port 6
	High 🛱	Tunnel 2		None	¢\$	Tunnel 1
	High 🔯	Tunnel 3		None	08	Tunnel 2
	High 🕵	CAN wrapper 1 for serial tran		None	08	Tunnel 3
		CANI		N		Serial CAN unwrapp

Y1 splitter consumer

Y2	splitter	consumer

GPIO	Configuration							
PWM	Priority	Producer	Co	Consumer				
I/O Setup	High 👯	UART-A / RS232-A	•	Commgr port 2	Bind	► î		
CAN I/O	High 👯	UART-B / RS232-B	+	Commgr port 3	Bind Bidirectional	▶ UART-A		
Digital Input	High 🔯	UART-C / RS485	•	None	Clear	UART-B		
SCI CAN Setup	High 🗱	CAN to Serial 1	•	Commgr port 1	OC .	UART-C		
	High 🔯	CAN to Serial 2	•	None	06	Lift - MCU		
	High 🗱	CAN to Serial 3	•	None	00			
	High 🔯	Commgr port 1	•	Serial to CAN 1	06			
	High 🔯	Commgr port 2	+	UART-A / RS232-A	00			
	High 🗱	Commgrf port 3	+	UART-B / RS232-B	00			
	High 🔯	Commgr port 4	+	None	00			
	High 🔯	Commgr port 5	+	None	00			
	High 🔯	Commgr port 6	+	None	00	U		
	High 🔯	Tunnel 1	+	None	00			
	High 🔯	Tunnel 2	•	None	06			
	High 🔯	Tunnel 3	•	None	00			
	High 🗱	CAN wrapper 1 for serial tran	•	None	06			
	Contraction Contraction	CANL		NI	- the the	~		

Fig. 9: I/O Setup consumer options

The following I/O ports are available:

Field	Description
UART-A / RS232-A	Serial Port 232
UART-B / RS232-B	Serial Port 232
UART-C / RS485	Serial Port 485
Commgr port	COM Manager ports send and receive VCP messages. VCP is the protocol used by
	Veronte products to communicate.
	For more information on this, read the VCP user manual
Tunnel	Creates a bidirectional brigde between two devices, see <i>Tunnel</i>
Custom message	This allows user to send/receive a serial custom message, see Custom message
producer/consumer	
Y splitter producer/consumer	Used to split a signal into 2
CAN to serial / Serial to CAN	Serial to CAN sends serial streams via CAN Bus / CAN to serial undoes the
	transformation 'Serial to CAN'
CAN wrapper for serial	CAN wrapper sends CAN streams over a serial Bus / Serial CAN unwrapper
transmission / Serial CAN	undoes this transformation
unwrapper	
Tribunus ESC	Reads telemetry data from the Tribunus ESCs by connecting it to one of the serial
	ports
Lift - MCU	Created for communication with a Lift MCU
JETI box	Simulates a Jetibox to read telemetry from legacy Jeti devices, see JETI box
JETI telemetry	Reads telemetry from Jeti devices

More information about some elements can be found in the following sections.

2.3.3.1 Tunnel

It is possible to configure a Tunnel which is a bidirectional bridge between units that communicate to each other sharing information about an external device connected to the Serial or Digital port.

Let's consider the following image.

CEX2 PD	1 Builder	•	Input / Output			*	- × 0 = 0		
Ö	gpio pwm	Configuration Priority	Producer		Consumer			0	
	I/O Setup	High 😡	UART-A / RS232-A	→	Tunnel 1	QC -	Veronte ID	Broadcast	-
	CAN I/O	High 🕫	UART-B / RS232-B	\rightarrow	Commgr port 3	08	Parser	No protocol	•
	Digital Input SCI	High 🔯	UART-C / RS485	→	None	08	Destination tunnel	Tunnel producer 1	•
P	CAN Setup	High 🔯	CAN to Serial 1	→	Commgr port 1	00	Time between messages	0.01	S
5 2		High 🔯	CAN to Serial 2	→	None	00	Bytes to send	22	byte
0		High 🔯	CAN to Serial 3	\rightarrow	None	00			
		High 🔯	Commgr port 1	\rightarrow	Serial to CAN 1	00			
÷.		High 😡	Commgr port 2	→	UART-A / RS232-A	00			
		High 😡	Commgrf port 3	→	UART-B / RS232-B	00			
		High 😡	Commgr port 4	\rightarrow	None	00			
		High 😡	Commgr port 5	\rightarrow	None	00			
		High 🗘	Commgr port 6	\rightarrow	None	00			
		High 🗘	Tunnel 1	\rightarrow	None	00			
		High 🗘	Tunnel 2	\rightarrow	None	00			
		High 🔯	Tunnel 3	\rightarrow	None	00			
		High 🔯	CAN wrapper 0 for serial tran	\rightarrow	None	00			
		High 82	CAN wranner 1 for cerial tran		None	458	~		

Fig. 10: Tunnel configuration

In the image above there is a device connected to the **UART-A/RS232-A** (**Producer**) and there is a **Tunnel** (**Consumer**) which sends that information to all units on the network (Broadcast). On the other hand, another Veronte device has to be configured to receive the signal sent by another device. In that case the **Producer** will be **Tunnel** and **Consumer** will be the **port or destination tunnel where the device is connected**.

The option available when configuring Tunnel as Consumer are:

- Veronte ID: Select the address that will receive the information.
 - App 2: Veronte applications address.
 - Broadcast: All units on the network. Select this option for a generic configuration.
 - CEX2 XXXXX: Address of a specific unit, it can be a CEX, a 1x, a 4x, etc.
- Parser: The user can choose protocol to parse message data. The options available are:
 - No protocol
 - RTCM3
 - CANserial
- **Destination tunnel**: Number of port is used to avoid mistakes and identify a Tunnel when using more than one, *Tunnel 1, 2 and 3* are available.
- Time between messages.
- Bytes to send: Sets the message size to send.

When configuring **Tunnel as Producer** (i.e. on the unit that receives the information), no configuration is required. It is only necessary to connect it to a Consumer, usually to a serial port.

2.3.3.2 Custom Messages

Warning:

- CEX has a serial limitation shared with all Custom Messages:
 - Maximum number of vectors (fieldset): 100
 - Maximum number of fields: 1000
- Each Custom message **consumer** has a limit of 32 fields.

It is possible to configure the messages sent/received through the serial port and its conversion to system variables by selecting the option **Custom message producer/consumer** and configuring the I/O port.

	• •	Input / Output		l	* 0 *
GPIO	Configuration	1			
PWM	Priority	Producer		Consumer	- 14
I/O Setup	High 🗱	CAN to Serial 2	\rightarrow	None	Q0
CAN I/O	📄 High 🗱	CAN to Serial 3	\rightarrow	None	Q_0^0
Digital Input SCI	🗌 High 🙀	Commgr port 1	→	Serial to CAN 1	Q0
CAN Setup	High 🔯	Commgr port 2	\rightarrow	UART-A / RS232-A	Q ₀ ⁰
	🗌 High 🔯	Commgrf port 3]→[UART-B / RS232-B	00
	High 🔯	Commgr port 4	\rightarrow	None	00
	High 🔯	Commgr port 5	\rightarrow	None	₽ ⁰
	High 🔯	Commgr port 6	\rightarrow	None	$\dot{\Omega}^{0}_{0}$
	🗌 High 🔯	Tunnel 1)→[Custom message consumer 1	$\mathbf{Q}_{\mathbf{Q}}^{\mathbf{Q}}$
	High 🔯	Tunnel 2	\rightarrow	None	00
	High 🔯	Tunnel 3	\rightarrow	None	Q_0^0
	High 🔯	CAN wrapper 0 for serial tran	\rightarrow	None	00
	High 🔯	CAN wrapper 1 for serial tran	\rightarrow	None	Q0
	High 🗰	Custom message producer 0	\rightarrow	None	Q0
	🗌 High 🙀	Custom message producer 1]→[UART-C / RS485	Θ_0^0
	High 🗱	Lift - MCU	\rightarrow	None	Q ₀ ⁰
	High 100	Y0 splitter producer 0	i → ī	None	de la

Fig. 11: Serial Custom Messages

In the image above can be seen two possible configurations using a Custom Message. The 'red' one is configured to receive a determined message from Tunnel 1 and the 'green' is used to send a Custom Message through a UART-C/RS485 serial port. It is also possible to use the same Custom Message for both tasks if the bidirectional relationship is selected (the arrow indicates this, \leftrightarrow).

To configure a Custom message, the user must follow the next steps:

1. Press the configuration button (

icon) and another window will be displayed.

[Consumer] Cons	umer Custom mes	ssage consume	r 1										- ;
- 50	Little endian	Time out	Time to Idle	Bit ID: 0	Checksum 0	Matcher 0	Skip O	Variable O	ASCII 0	Position 0	Occupancy	₽	÷
				Memory u		6204 hut-							
				wemory u	sage: 0 / T	0304 Dyte	3						

In this window press the "+" icon to add a custom message.

Fig. 12: Serial Custom Message configuration

2. When it is already added, the following options are available to configure a custom message:

[Producer] Producer Custom messag	ge producer 1							- ×
Little endian	Period Delay 1.0 s 0.0	Checksum M s 0	latcher Skip O O	Variable 0	ASCII 0	Position 0	Occupancy	+

Fig. 13: Custom Message producer configuration

[Consumer] Consumer Custom me	ssage consume	r 1									- x
Little endian	Time out 1.0 s	Time to Idle	Bit ID: 0	Checksum 0	Matcher 0	Skip O	Variable 0	ASCII 0	Position 0	Occupancy	+



• Endianness: Depending on the order in which the device issue the message, it is possible to select:

- Big endian: Set the value from left to right.
- Little endian: Set the value from right to left.
- Mixed endian: Some devices use this format. If users need to configure it, please contact the support team (create a ticket in the customer's Joint Collaboration Framework; for more information, see Tickets section of the JCF manual).
- Period/Time out: This option has a dual role depending on if it is used to transmit or receive data.
 - Period Producer: It is the inverse of the send frequency.
 - **Time out Consumer**: This is the threshold time between receptions to consider that the message is not being received correctly.
- Delay/Time to Idle: This option has a dual role depending on if it is used to transmit or receive data.
 - **Delay Producer**: It is a delay applied before sending the message. This serves to send messages with the same period without overloading the Serial bus.
 - Time to Idle Consumer: This is the time CEX waits before discarding partially parsed bytes.
- **Bit ID**: This option is only available when a message is configured as **Consumer**. The user bit selected in Bit ID box will be true if the message is being received correctly.

Warning: Pay attention that the user bit selected in Bit ID is not in use for another task.

3. To create the structure of the message, click on the **edit message button** () icon) and then press the "+" icon to add fields to it. The following type of messages are available to configure a structure: **Variable**, **Checksum**, **Matcher**, **Skip**, **Parse ASCII** and **Position**.

The configuration of each structure is covered in Custom Messages section of the 1x PDI Builder manual.

Warning: Before configuring any message, user has to know the structure it has to have according to the device that is connected to the port. Each device may have a different message structure when it sends or receives information.

2.3.3.3 JETI box

JETIBOX is a universal communication terminal which can be used with any JETI products.

JETIBOX operates as a two-way terminal, showing all data stored in the JETIBOX Compatible product. With the use of **four buttons**, the user can browse the menu and set the selected values to take advantage of the full capability of JETIBOX Compatible products.



Fig. 15: JETI box device

To simulate it, it is necessary to link the specific **JETI box consumer** to a serial port:

	- •	Input / Output			T 0
GPIO	Configuration				
PWM	Priority	Producer		Consumer	
/O Setup	High 🔅	UART-A / RS232-A	\rightarrow	JETI box	Q ⁰
CAN I/O	High 🔯	UART-B / RS232-B	\rightarrow	None	00
Digital Input	High 🔯	UART-C / RS485	\rightarrow	None	00
CAN Setup	High 🛱	CAN to Serial 1	\rightarrow	None	Ø0
	High 🕰	CAN to Serial 2	\rightarrow	None	Q0
	High 🕰	CAN to Serial 3	\rightarrow	None	Q0
	High 🕰	Commgr port 1	\rightarrow	None	Q0
	High 🕰	Commgr port 2		UART-B / RS232-B	Q0
	High 🕰	Commgrf port 3	\rightarrow	None	Q0
	High 🛱	Commgr port 4	\rightarrow	None	Q0
	High 🕰	Commgr port 5	\rightarrow	None	Q0
	High 🚭	Commgr port 6	\rightarrow	None	¢¢
	High 🕰	Tunnel 1	→	None	Q0
	High 🚭	Tunnel 2	\rightarrow	None	Ø0
	High 🔯	Tunnel 3	\rightarrow	None	Q0

Fig. 16: JETI box Consumer

Then, the sequence to retrieve the data shall be configured by clicking on the configuration button (

• To add a Custom Message, just clik on the "+" icon:

CEX2 PE	DI Builder					-	- ×
		•🕁 Inp	ut / Output			* • •	0
0	GPIO	Configuration					
	PWM	Priority	Producer	Consun	ner		
2	I/O Setup	High Q ₀ ^o	UART-A / RS232-A	→	JETI box		â
•	[Consumer] Consumer JETI box						- ×
	🚥 📑 🗍 🛛 Little endian	Command Jeti Box Nop 👻	Checksum Matcher	Skip Variable ASCII 0 0 0	Position Occupancy 0		÷
630							
© \$ 0							
H							
-							
			Memory usag	e: 0 / 16384 bytes			

Fig. 17: JETI box Configuration

The following parameters can be configured in this pop-up window:

- Endianness: Depending on the order in which the device issue the message, it is possible to select:
 - * **Big endian**: Set the value from left to right.
 - * Little endian: Set the value from right to left.
 - * **Mixed endian**: Some devices use this format. If users need to configure it, please contact the support team (create a ticket in the customer's **Joint Collaboration Framework**; for more information, see Tickets section of the JCF manual).
- Command: Here the user can select between Jeti box Left, Jeti box Down, Jeti box Up, Jeti box Right or Jeti box Nop. These correspond to the four buttons on the physical JETIBOX (see image above), except the Jeti box Nop that is only for simulating a "wait".

For example, to read the **Actual Voltage** of a **Jeti MasterSpin 220** the Consumer must be configured with a series of custom messages (use **Big endian** in all messages).

		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
- 🖳 🗆	Big endian	Jeti Box Down	*	0	2	8	0	0	0		
		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
• 🖳 🗍	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command			Matcher	Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
n a		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
n -		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
اللا 😴	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
	D : 1	Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
n ar		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
n N N		Command		Checksum		Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Nop	*	0	1	8	0	1	0		

Fig. 18: JETI box example

Below is a small example with the configuration of one of these messages, the full example can be found in *Jetibox* - *Integration examples* section.

- 1. Expected text: "CONTROLLER TYPE MasterSpin 220~"
 - Command: Jeti Box Down
 - Matcher(32) "CONT" 0x434F4E54 (1129270868)
 - Skip(24*8) 192
 - Matcher(32) "220~" 0x3232307E (842150014)

[Consumer] Consun	ner JETI box											- ×
- 🗐 🗖	Big endian	Comr Jeti Box D		Checksum 0	Matcher 2	Skip 8	Variable 0	ASCII 0	Position 0	Occupancy		î 🕂
			own •	0	2	0	0	U	0	L	^	
	Hate Mate	her x434F4E54									^+	
Value	Bits		Mask									
1129270868	32	4294967295		dec								
Bits	🗘 Skip	24										
24												
	🕂 Skip	24										
	Skip	24										
	Skip	24										
	Skip	24										
	Skip	24										
	Skip	24										
	skip	24										
	♣ Matc	her x3232307E										
Value	Bits		Mask									U
842150014	32	4294967295		dec							~	~
				Me	mory usa	ge: 256	/ 16384 b	ytes				

Fig. 19: JETI box custom message example

2.3.4 CAN I/O

Warning: Remember that for the reception of CAN messages, Mailboxes need to be configured accordingly.

A CAN (Controller Area Network) Bus is a robust standard communication protocol for vehicles widely used in the aviation sector. CEX is fitted with two CAN buses that can be configured independently.

The structure of a CAN message can be seen in the following image:

•	CAN MESSAGE								
1 bit	29 bit	1 bit	6 bit	0-64	16 bit	2 bit	7 bit		
SOF Start Of Frame	CAN-ID Extended identifier T	RTR Remote ransmissie Request	Control	Data	CRC Cyclic Redundancy Check	ACK Acknow- ledge	EOF End Of Frame		

Fig. 20: CAN message structure

Only the ID is introduced in the system, the rest of the message layout is already coded. The data field is the one build by the user to send, and parsed when received.

For more information on the CAN Bus protocol, see CAN Bus protocol section of the CEX Software manual.

The baud rate of both CAN buses can be configured in the CAN Setup panel.

2.3.4.1 Configuration

This menu allows the configuration of the CAN inputs and outputs.

• 🗘 Input / Output									
gpio PWM	Configura	ation	CAN Telemetry 1 CAN Telemetry	2 CAN	Telemetry 3				
I/O Setup	Priority High	Q ⁰ ₆	Application Processor	→	Consumer CAN Output Filter 2	¢\$			
CAN I/O									
Digital Input	High	00	CAN Input Filter 1		CAN to Serial 1	\$ \$			
SCI	High High	Q_0^0	CAN Input Filter 2	\rightarrow	Application Processor	Q ⁰			
CAN Setup	High	Q^0_0	CAN Input Filter 3	\rightarrow	None	Q0			
	🔲 High	Q^0_0	CAN Input Filter 4	\rightarrow	None	Q0			
	🗌 High	Q_0^0	CAN Input Filter 5	\rightarrow	None	Q0			
	🗌 High	Q^0_0	CAN Input Filter 6	\rightarrow	None	¢¢			
	🗌 High	Q^0_0	CAN unwrapper 1	\rightarrow	None	¢¢			
	🗌 High	Q^0_0	CAN unwrapper 2	\rightarrow	None	Ø\$			
	🗌 High	Q^0_0	Serial to CAN 1	→	CAN Output Filter 1	Ø0			
	🗌 High	Q^0_0	Serial to CAN 2	\rightarrow	None	¢¢			
	🔲 High	Q^0_0	Serial to CAN 3	\rightarrow	None	Ø6			
	🗌 High	Q^0_0	CAN Custom Producer 1 Tele	\rightarrow	None	¢¢			
	🗌 High	Q^0_0	CAN Custom Producer 2 Tele	\rightarrow	None	OC C			
	High	00	CAN Custom Producer 3 Tele	\rightarrow	None	08			

Fig. 21: CAN configuration panel

In this menu, the user can find the same 'columns' (Priority, Producer and Consumer) as in the I/O Setup panel.

Warning: In CAN, while the specified period is not guaranteed in the Low state, in the High state it is.

However, only those **messages that are critical for external devices** should be set as **high priority**, as this may disrupt the proper functioning of CEX.

On the one hand, CEX has the **producers** shown below:

- **Application Processor**: Sends a specific set of information, the "status message". This message is composed as: version (major, minor and revision), address (serial number), system bit error, system power up bit error, PDI bit error, memory allocation bit, fily system bit error, CAN A bit error, CAN B bit error, arbiter enabled and arbiter status.
- CAN Input Filter: CAN input filters. Those CAN messages received in one filter can no longer be received in

subsequent filters. The following parameters need to be configured by clicking on the icon:

Port	CAN A	*
ld	8	
Mask	2044	dec
Filter type	Standard	-

Fig. 22: CAN Input Filter configuration

- **Port**: It is required to configure the CAN bus from which it listens, the user can choose between *CAN A*, *CAN B* or *BOTH*.
- Id: CAN Id must be set and it is used to identify messages. The value set has to be decimal format.
- Mask: A CAN Id mask can be set to filter messages. The mask marks which bits of the message id (in binary) are matched.

For example, to admit standard Ids (11 bits) from 8 to 11 (100 to 111 in binary) the user should set the mask to binary 1111111100, that is 2044 in decimal.

Warning: Make sure that mask is set properly to be able to receive the desired CAN messages.

The mask should be 11 bits for Standard frame format and 29 bits for Extended frame format.

More information about this can be found in *How to calculate a mask - FAQ* section of this manual.

- Filter type: The options available are *Standard* (frame format with a 11-bit identifier), *Extended* (frame format with a 29-bit identifier) and *Both*.
- CAN unwrapper: This undoes the 'CAN serial wrapper' action, it has to be connected to *I/O Setup* consumer (*Serial CAN unwrapper*).

• Serial to CAN: Serial messages via CAN output, it has to be connected to *I/O Setup* consumer. It can be configured in the configuration button (icon), a pop-up window will appear:



Fig. 23: Serial to CAN configuration

- Id: CAN Id must be set and it is used to identify messages. The value set has to be decimal format.
- **Extended**: If enabled, the frame format will be this, 'Extended', i.e. with a 29-bit identifier. Otherwise, the frame format 'Standard' (11-bit identifier) is set by default.
- Time out: This is the threshold time between receptions to consider that it is not being received correctly.
- **CAN Custom Producer Telemetry**: Telemetry messages sent via CAN (such as CAN custom messages on Veronte Autopilot 1x). They are configured in the next section, see *CAN Telemetry*.

On the other hand, the **consumers** are the following:

- Application Processor: Receives a specific set of information sent by Veronte Autopilot 1x or Arbiter.
- CAN Output Filter: CAN output filters. The user can choose between CAN A, CAN B or BOTH in the

configuration button (icon).



Fig. 24: CAN Output Filter configuration

- CAN serial wrapper: CAN messages via serial output, it has to be connected to *I/O Setup* producer (CAN wrapper for serial transmission).
- CAN GPIO consumer: CAN messages from 1x autopilot for GPIO inputs. An example of how to implement it can be found in *GPIO Command Integration examples* section.
- CAN to Serial: This undoes the 'Serial to CAN' action, it has to be connected to I/O Setup producer.
- CAN Custom Consumer Telemetry: Telemetry messages received via CAN (such as CAN custom messages on Veronte Autopilot 1x). They are configured in the next section, see *CAN Telemetry*.

2.3.4.2 CAN Telemetry

In the CAN Telemetry tabs (there are 3 available), the user chooses the telemetry to be sent/received via the CAN buses. The following elements can be configured:

- **TX Ini**: Used to configure transmitted messages that are only sent once at the beginning of the operation (sent when CEX boots up). They can be used to initialize some devices.
- **TX**: Used to configure transmitted messages.
- **RX**: Used to configure the reception messages (where they are stored).

Warning:

- The maximum capacity of a CAN message is 64 bits (8 bytes), so to send more information it must be divided into several messages.
- CEX has a CAN limitation of 40 TX Ini messages per Custom, 40 TX messages per Custom and 80 RX messages per Custom.

In addition, there is a limit shared with all Customs:

- Maximum number of vectors (fieldset): 160
- Maximum number of fields: 1280

CEX2 PDI Builder		- x
	•∕• Input / Output	2020
GPIO PWM I/O Setup CAN I/O Digital Input SCI CAN Setup CAN Setup	Configuration CAN Telemetry 1 CAN Telemetry 2 CAN Telemetry 3	

Fig. 25: CAN Telemetry panel

Since this panel works in a similar way to the CAN Custom Message configuration in the **1x PDI Builder** software, the explanation to configure the telemetry messages via CAN can be found in the Custom Messages - Input/Output section of the **1x PDI Builder** user manual.

An example of the sending of **CAN telemetry messages** can be found in the *CAN communication - Integration examples* section.

2.3.5 Digital Input

CEX digital signal inputs can be used to measure pulse count, pulse widths and PPM signals from a RC radio. Each source shall be connected to the desired consumer to allow measurements.

	•	🕂 Input / Outp	out		(± 0	9
GPIO	Configuration	on					
		Producer		Consumer) (- M)		
I/O Setup	00	CAP 1	\rightarrow	Pulse 1	Q_0^0		
CAN I/O	\$	CAP 2	\rightarrow	RPM 4	Q0		
	¢\$	CAP 3		RPM 1	Ø0		
SCI CAN Setup		CAP 4		Pulse 4	00		

Fig. 26: Digital Input panel

In this menu, the user can find the same 'columns' (Producer and Consumer) as in the I/O Setup panel.

The process to configure a device can be done as follows:

1. Select and configure a **Producer**. There are 4 possible producers: CAP 1 - 4.

Press on the configuration button (icon) and a new pop-up window will show.

CEX2 PDI Builder	ৰ্শ্ব Input / Output		 • • • •
GPIO PWM I/O Setup CAN I/O Digital Input © CAN Setup CAN Setup T	Configuration	RPM 4	

Fig. 27: Digital Input - Producer

In the pop-up window, users can check:

- That this producer is **enabled**.
- Which **pin** this CAP is associated to and, therefore, to which device is connected. They are associated in this way: **CAP 1** with **I/O1**, **CAP 2** with **I/O2**, **CAP 3** with **I/O3** and **CAP 4** with **I/O4**.
- How pulses are read and transformed into a digital signal (how they are processed). That can be configured with the **Edge detection** option.

By clicking on the drop-down menu, the following options can be selected:

	Enabled		
30	Edge detection	First falling edge 🔹 👻	
	7	First rising edge	
3465		First falling edge	γ

Fig. 28: Digital Input - Edge detection options

- First rising edge: With this option, when the rise of the pulse is detected, the data will start to be stored. Recommended when consumer is **PPM** or **Pulse**.
- First falling edge: With this option, when the fall of the pulse is detected, the data will start to be stored.

CEX2 PDI Builde	er			
				_

Note: By clicking on the marked arrows, it can also be configured as desired, getting another scheme.

		🕰 Input / Output		*	0 2 0
0	GPIO	Configuration			
	PWM	Producer	Consumer		
2	I/O Setup	Enabled	None	Q0	î
•	CAN I/O	₩\$ I/01	None	OC C	
א ⊷ ©	Digital Input SCI	Edge detection First falling edge	None	Q0	
P	CAN Setup		None	Ø\$	

2. Click on the **Bind** button to select the type of **Consumer**, it is possible to choose among a PPM 0 (Stick PPM), RPM 1-4 or Pulse 1-4.

(2 PDI	Builder	•4	🖶 Input / Outp	out		(± 0 6
	GPIO	Configuration	on				
	PWM		Producer		Consumer		
]	I/O Setup	\$\$°	CAP 1	\rightarrow	Pulse 1	Bind	PPM0
	CAN I/O	00	CAP 2	\rightarrow	RPM 4	Clear	RPM 1
	Digital Input		CAP 3		RPM 1	00	RPM 2
	SCI		CAP 4	\rightarrow	Pulse 4	¢\$	RPM 3
	CAN Setup	346 <u>0</u>	CAP 4		Puise 4	2 ⁴⁶ Q	RPM 4
							Pulse 2
							Pulse 3
							Pulse 4
]							

Fig. 29: Digital Input - Consumer

- **PPM 0** selected: PPM is configured in the *Stick panel*.
- **RPM 1-4** selected: The variables in which the information read here is stored are '**RPM 1-4**'. For more information on the configuration of RPM, see the *RPM section*.
- Pulse 1-4 selected: The variables in which the information read here is stored are 'Captured pulse 1-4'.

It is possible to configure it cliking on the **configuration button** (icon):

CEX2 PDI Builder	•< ⁴ → Input /	Output		* 0	- × 0 4	
GPIO PWM I/O Setup CAN I/O Digital Input Image: Can Setup CAN Setup Image: Can	Configuration Producer ©© CAP 1 ©© CAP 2 ©© CAP 3 ©© CAP 4		Consumer Pulse 1 None None None	1.1 1.0 0.9 0.8 0.7 0.6 0.5 0.4 0.5 0.4 0.3 0.2 0.4 0.3 0.2 0.1 0.0	Positive pulse duration	s y 0.0 - y 1 -

Fig. 30: Digital Input - Pulse

In the pop-up window, users will find the following options for configuration:

- Mode:
 - * Positive pulse duration: The period of the pulse is obtained. It takes the time in 'High' state.
 - * Negative pulse duration: The period of the pulse is obtained. It takes the time in 'Low' state.



Fig. 31: Positive/Negative pulse duration

* Positive duty cycle: The duty cycle of the pulse is obtained. It takes the time in 'High' state.

* Negative duty cycle: The duty cycle of the pulse is obtained. It takes the time in 'Low' state.

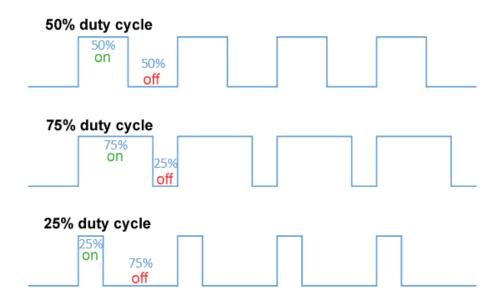


Fig. 32: Positive/Negative duty cycle

- **Time out**: This defines the time to consider that no signal is received.
- Function: Here the user can customise a function to handle the values. Normally, a function is set with the points [0,0] and [1,1], so no transformation is applied, input = output. However, the user can configure it as desired.

Example

Let's imagine that **First rising edge** has been selected as the wrap option in Producer and the pulse that CEX has to read is a square signal with a period of 2 seconds and a duty cycle of 25% (see image below).

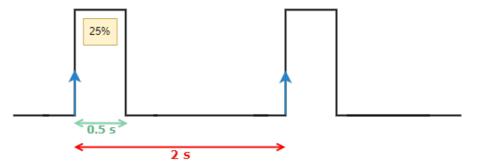


Fig. 33: Signal generated

On the other hand, if **Positive pulse duration** is selected as Consumer and it is configured as in the previous image (*Digital Input - Pulse*), the value obtained in the variable **Captured pulse** (*Captured pulse 1* in the following example) will be **0.50s**, this is because it is the period of the "Positive pulse" of that pulse.

However, if **Positive duty cycle** is selected as Consumer, the value obtained in the variable **Captured pulse** (*Captured pulse 2* in the following example) will be **0.25**, this is because it is the positive duty cycle of that pulse.

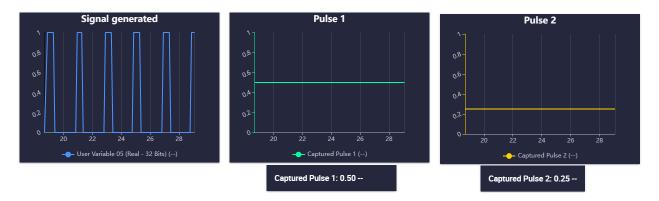


Fig. 34: Digital Input example

2.3.6 SCI

CEX can use up to three serial peripherals (SCI A, SCI B and SCI C). Serial ports A, B and C parameters can be edited in this menu to fit the serial protocol requirements.

CEX2PD	l Builder - CEX2 45010 - CONNEC	TED	- ×
		•∕⊶ Input / Output	
	GPIO PWM I/O Setup CAN I/O Digital Input SCI CAN Setup	SCI A SCI B SCI C Baudrate 115200 Length 8 Stop 1 Parity Disabled Use address mode	

Fig. 35: SCI Configuration

- Baudrate: This specifies how fast data is sent over a serial line.
- Length: This defines the number of data bits in each character: 4 to 8 bits.

- Stop: Number of stop bits sent at the end of every character: 1, 1.5, 2.
- **Parity**: Is a method of detecting errors in transmission. When parity is used with a serial port, an extra data bit is sent with each data character, arranged so that the number of 1 bits in each character, including the parity bit. **Disabled**, **odd** or **even**.
- Use address mode: 9-bit data framing uses the bit typically associated with parity error detection to identify address messages. Sent serial data that does not have the address bit set will be ignored (unless the device had previously identified an address message associated with it). This option can be disabled or enabled.

Note: SCI A corresponds to port RS232 A, SCI B to port RS232 B and SCI C to port RS485.

2.3.7 CAN Setup

In this screen users can configure the baudrate and the reception mailboxes of each CAN Bus (CAN A or CAN B)

Since **CEX** is going to receive data on the CAN Bus, it is mandatory to configure a certain number of mailboxes to store that data until CEX reads it. A mailbox can be configured for multiple CAN message IDs as long as the mask is configured correctly and these messages are **sent spaced out** with enough time between them to allow the high priority core to read each one individually. More information on masks can be found in *How to calculate a mask - FAQ* section of this manual.

Warning: Since CEX PDI Builder allows up tp 32 mailboxes, users should make sure to leave at least one mailbox free for transmission (TX).

If any mailbox is **full** and another message arrives, the **new** message is **discarded**.

			•🕁 Inp	ու / Ծակ	Jui		8 1 8 5
GPIO	CAN	4 C/	ANB				
PWM I/O Setup	Units	ID	DEC 👻	Units M	ask BIN	 Baudrate 1000000.0]
CAN I/O	Mailt	oxes	reserved RX:	16 Mailbo	xes available	TX: 16	
Digital Input	#		Mailboxes		ID	Mask	+
sci	1				1302	1111111111	
CAN Setup							
	2		4		8	1111111111	
	3		4		9	1111111111	
	4		4		10	1111111111	

Fig. 36: CAN Setup panel

More information about Mailboxes can be found in the Mailboxes section of the 1x PDI Builder user manual.



2.4.1 Ports

Ports configuration allows the user to configure which communication ports (Commgr Ports in *I/O Setup*) will be used for communication. When using the **Route** feature, CEX can be configured to **route** VCP messages for an external Veronte device with a known address (ID) through a given port.

Builder - CEX2 45010 - CONNECTED	- ×
	8 2 8 9
Ports	+
Forward + Route + PORT 1 PORT 2 PORT 3 PORT 4 PORT 4 PORT 5 PORT 5 PORT 6	
	Ports Routing 1 Forward PORT 1 PORT 2 PORT 3 PORT 4 PORT 4

Fig. 37: Ports menu

Each port can be configured as either one of the following options:

- Forward: Any messages generated by this unit (i.e. Telemetry or response messages to certain commands) will be sent through these ports.
- **Route**: Any messages received at any Commgr Port with the defined address will be re-sent through the defined port. It is possible to route several addresses through the same port, but is not possible to route the same address through several ports. Only the first configured port will be used. Routing also applies to messages generated by the unit for the defined address.

Note: The same port cannot be used as Forward and Route at the same time.

It is possible to define up to 4 routing setups, which can be switched unsing the Ports action of the Automations menu of **1x PDI Builder**. Routing 1 will always be selected by default when booting CEX.



The wired transmitter is configured through the following tabs, which cover the following content:

- Setting of the transmitter's parameters.
- Definition of exponential response-curves for the desired channels.
- Trimming of the channels' neutral position.
- Setting of the data receiving port on the autopilot.

2.5.1 PPM

This tab provides the options to configure CEX to read a PPM radio controller to control the platform.

CEX2 PD)l Builder					×
			😨 Stick			
Ö	PPM Exponential	Trim Output				
_	Brand Custom	▼ Model	- Chan	inels 16		
ھ ج	Pulse polarity	Positive Negative	Sync time	0.004 s		
	Min pulse	2.5E-4 s	Max pulse	5.0E-4 s		
(P)	Position					
650	Min accepted	8.0E-4 s	Max accepted	0.0022 s		
$\begin{tabular}{ c c c c } \hline \hline \end{tabular}$	Min value encoded	9.0E-4 s	Max value encoded	0.0021 s		
	Channel(DisabledE	nabledFilter) 1 2	3 4 5 6 7 8 9 10 1	1 12 13 14 15 16		
.	Non linear low pas	s filter				
	Min delta	0.0	Max delta	1000.0		
	Min delta alpha	1.0	Max delta alpha	0.02		

Fig. 38: PPM panel

• **Brand**, **Model** and **Channels**: **CEX PDI Builder** has been configured to provide the user with the expected parameters to configure different transmitters models.

Brand	Models	Channels
Futaba	8J/10J/12K/14SG	8 (for 8J and 10J)
		12 (for 12K and 14SG)
	T18SZ	8
Jeti	DC 16/DC 24	16
FrSky	Taranis X9D	8
	Horus X12S	8
TBS	Crossfire	8
Embention	Stick Expander	16
Custom	-	-

- Custom: If the user's transmitter is not among those mentioned above, choose this option and replace the parameter values with the appropriate ones.
- Pulse polarity: Indicates the pulse polarity:
 - **Positive**: Default signal is low and goes up to high.
 - Negative: Default signal is high and goes down to low.
- Sync time: Minimum time on the PPM output till the next frame. It tells the receiver to reset its channel counter.
- **Minimum/Maximum pulse**: Pulse length, it depends on the system and it is a constant value (usually 0.2-0.5 ms).
- Position
 - Minimum/Maximum accepted: Pulse length accepted for each channel. Standard for R/C servos uses a pulse of 1 ms for the maximum position at one end, 1.5 ms for the midpoint and 2 ms for the maximum position at the opposite end.
 - **Minimum/Maximum encoded**: If there is noise and the signal is varying around the minimum/maximum values accepted, CEX will encode those values to the ones set here. For instance, a pulse length between 0.8-0.9 ms will be considered as one of 0.9 ms.
 - Channels: Sets the number of channels accepted. Besides, it is possible to Disable/Enable/Filter each channel individually.
- Non linear low pass filter
 - Minimum/Maximum delta: Default parameters are recommended.
 - Minimum/Maximum delta alpha: Default parameters are recommended.

The figure below shows the shape of the PPM signal.

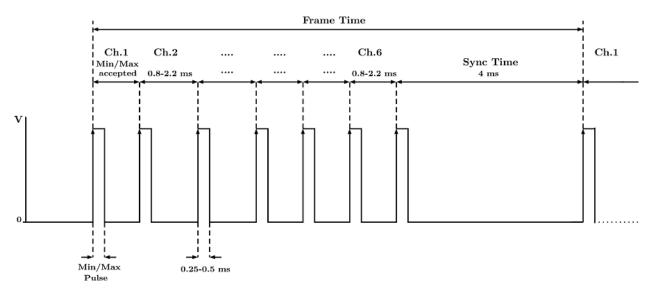


Fig. 39: PPM signal

2.5.2 Exponential

The exponential tab allows the user to define an exponential stick response for every channel.

The allowed inputs range from 0 to 1 and there is a graph showing the generated response curve, as can be seen in the figure below.

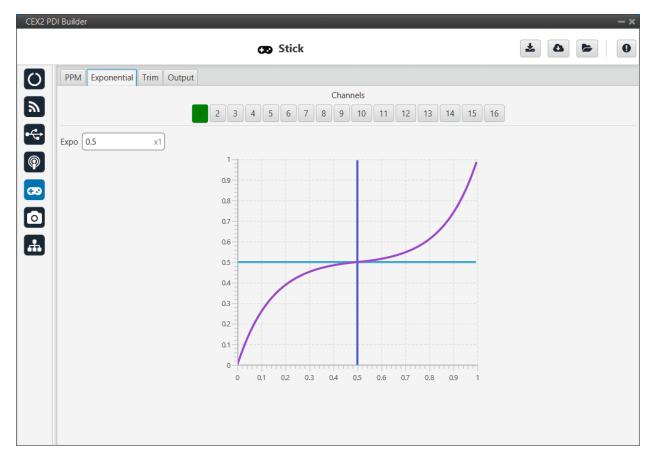


Fig. 40: Exponential panel

The X axis of the graph corresponds to the stick input and the Y axis is the result of applying the exponential function to that stick input.

2.5.3 Trim

By enabling the **Avanced** option, the user can set the expected trim values manually. The user should have a deep knowledge on its transmitter if this option is selected.

Finally, on the right hand side, the **Reset** button puts every parameter back to 0.

CEX2 PD	01 Builder											- ×
					69	Stic	k				70	0
0	PPM Expo	onential	Trim Output	:								
2	Advanc	ed							Reset			
		Trir	m Range	Neutral		Tri	m	Range	Neutral			
≈ ∳ ©	Channel 1	0.0	0.6	Zero	Channel 9	0.0		0.7	Zero			
P	Channel 2	0.0	0.7	Zero	Channel 10	0.0		0.7	Zero			
53	Channel 3	0.0	0.7	Zero	Channel 11	0.0		0.7	Zero			
$\begin{tabular}{ c c c c c } \hline \hline \end{tabular}$	Channel 4	0.0	0.7	Zero	Channel 12	0.0		0.7	Zero			
	Channel 5	0.0	0.7	Zero	Channel 13	0.0		0.7	Zero			
	Channel 6	0.0	0.7	Zero	Channel 14	0.0		0.7	Zero			
	Channel 7	0.0	0.7	Zero	Channel 15	0.0		0.7	Zero			
	Channel 8	0.0	0.7	Zero	Channel 16	0.0		0.7	Zero			

Fig. 41: Trim panel

2.5.4 Output

In this menu the user sets the receiving port and process the incoming commands. Once the stick has been configured, the commands have to be sent to the air unit.

CEX2 PE	DI Builder	X
	CD Stick	4 4 4
Ö	PPM Exponential Trim Output	
<u></u>	Enable Initial 1 Port 0	
	Remote UAV Broadcast Min period 0.02 s Delta 10 Max period 0.2 s	
<u>∞</u> ⊙	Enable Initial 1 Port 0	
#	Remote UAV Min period 0.0 s Delta 0 Max period 0.0 s	

Fig. 42: Output panel

In this menu, the following parameters can be configured:

- Enable.
- **Initial Channel at destination**: The user indicates to which channel of the air autopilot will be sent the first channel received in the ground unit. The channels arrive at the platform in order and without spaces between them.

For example, if at the GND channels 6, 7, 8, 9 and 10 are enabled, the AIR will receive channels 1, 2, 3, 4 and 5. Therefore channel 6 of the stick will be channel 1 in the AIR configuration.

- **Port**: If more than one transmitter is configured, each transmitter must be configured on a different port. This has to match the port set on the air unit.
- Remote: It has to be enabled if the user wants to allow the delivery of the commands to the platform.
 - UAV: The address of the UAV that receive the commands has to be indicated. The following options are available:
 - * App 2: Veronte applications address.
 - * Broadcast: The commands are sent to all units on the network. We recommend this option.
 - * Veronte v4.X XXXX: The address of a specific air unit.
 - Min period: As the period is the inverse of the frequency, this is the maximum frequency. Therefore, to give the pilot more control, this is the frequency that is set when the stick is commanding. We recommend 0.02 s.

- Max period: As the period is the inverse of the frequency, this is the minimum frequency. Thus, to free up bandwidth, this is the frequency that is set when the stick is idle. We recommend 0.2 s.
- **Delta**: This parameter determines whether the frequency is set to the minimum or maximum period set above.

If CEX detects a **change above** the **delta value**, the frequency goes to the maximum frequency (minimum period). While if the **changes are less than this value**, it switches to the **minimum frequency** (maximum period). We recommend **10**.

Note: An example of how to configure a stick can be found in the Stick - Integration examples section of the **1x PDI Builder** user manual.



2.6.1 Jetibox

CEX can simulate a Jetibox to read telemetry from legacy Jeti devices.

CEX2PDI	Builder - CEX2 45010 - CONNEC	TED Devices	- ×
)	Jetibox Scorpion tribunus	Enable SCI Port SCI A	

Fig. 43: Jetibox panel

• Enable: It can be enabled or disabled by the user.

• SCI Port: A SCI port must be selected. The available options are: SCI A, SCI B or SCI C.

Besides, a configuration is needed in SCI and I/O Setup panels.

An example of this can be seen in the *Jetibox* - *Integration examples* section.

Note: The serial port will be totally reserved for this, so it will not be usable to other things and the I/O Setup affecting it wil be ignored.

2.6.2 Scorpion tribunus

CEX can read telemetry from Tribunus ESCs by connecting it to one of its serial ports.

CEX2PD	Builder - CEX2 45010 - CONNECT	ED		- ×
		Dev	vices	
\bigcirc	Jetibox	Enable		
	Scorpion tribunus	Telemetry period	0.02	
		MCU Telemetry 1 period	0.05	
•				
		MCU Telemetry 2 period	0.05	
P		PWM ID	0	
620				
-				
Ŧ				

Fig. 44: Scorpion tribunus panel

The following parameters are configurable:

- Enable: It can be enabled or disabled by the user.
- Telemetry period.
- MCU telemetry 1/2 period: Telemetry period for MCU devices.
- PWM ID: PWM ID associated to the Scorpion Tribunus.

The serial port has to be configured in the *I/O Setup* panel.

An example of this can be seen in the Scorpion tribunus - Integration examples section.

Note: The serial port will be totally reserved for this, so it will not be usable to other things and the I/O Setup affecting it will be ignored.



The arbitration algorithm in Veronte is based on a **scoring system**. Each autopilot must send continuously a set of arbitration variables that will be used by the arbiter in order to calculate the score for each unit. Then, based on the scores and the current arbitration mode, the arbiter will choose to keep the current selected autopilot, or switching to one of the other units.

A **Score** is a 32 bit, single precission, floating-point value. This parameter is first computed resulting in a range between **0** and **0xFFFFFFFF**, where **0** is a perfect score (*Score*'). To achieve a better understanding, *Score*' is converted to a value comprehended between 0 an 100 (*Score*), being **100** the best possible score in the end.

Scores are calculated using the **arbitration variables** received from each autopilot at their dedicated addresses. After receiving the value, the following formula is used to compute the score for their respective unit:

$$score'_{i} = \frac{\sum_{j}^{N} min\left[\left(\frac{x_{i,j} - \mu_{j}}{d_{j,max}}\right)^{2}, 1\right] \cdot w_{j}}{\sum_{j}^{N} w_{j}}$$

$$score_i = 100 \cdot (1 - score'_i)$$

Where:

j	Variable index
$x_{i,j}$	<i>j</i> variable from autopilot <i>i</i>
μ_j	<i>j</i> variable reference
$d_{j,max}$	Maximum allowed error
w_j	Weight of <i>j</i> variable
N	Number of variables
$score'_i$	Score' of unit <i>i</i>
$score_i$	Score of unit <i>i</i>

Tip: The Arbitration weights should be used to increase or decrease the relevance that a certain arbitration variable has over the calculation of the score.

Any variable in Veronte can be used as an arbitration variable. Depending on the platform, operation, application etc. the more relevant variables can be selected for its use as arbitration references.

2.7.1 Arbitration

CEX is able to output PWMs using arbitration in the same way **Veronte autopilot 4x** does. This functionality has to be enabled as follows:

CEX2PD	l Builder - CEX2 45010 - CONNECTED		- ×
		🕂 Arbitration	
	Arbitration Config CAN Setup CAN Setup Master arbitration RX CAN ID Extended CAN		

Fig. 45: Arbitration panel

Master arbitration RX CAN Id is exclusive for CEX when it is used alongside a Veronte Autopilot 4x. If **enabled**, when an arbitration message is received from the Arbiter of the 4x autopilot (**with the CAN Id configured here**), the selected autopilot will be updated according to the data received.

2.7.2 Config

2.7.2.1 Absolute Arbitration Variables

Absolute arbitration variables are indicators that are **inherently good** or **bad**, and so they are added directly to the score.

Examples of absolute arbitration variables are Link Quality, GNSS accuracy or warnings such as Sensors error or Position not fixed.

2.7.2.2 Relative Arbitration Variables

Relative arbitration variables are **not inherently good** or **bad**, and hence need to be compared against the other autopilots in order to calculate its score contribution.

The contribution to the score from a relative arbitration variable will be its **Deviation** from the **Average** of the same variable from each autopilot.

Examples of relative arbitration variables are Attitude, Position, measurements from sensors, etc.

2.7.2.3 Arbitration Example

Autopilot	Var.	Veronte		Туре	$x_{i,j}$	μ_j	w_j	Relative	$Score'_i$	$Score_i$
	Nº	Variable					_	score		
1	1	Roll		Relative	0.12	0.096	1	0.054	0.043	95.67
	2	Pitch		Relative	0.30	0.283	1	0.027		
	3	GNSS	1	Absolute	1.7		0.01	0.004		
		Accuracy								
2	1	Roll		Relative	0.10	0.096	1	0.0011	0.0039	99.60
	2	Pitch		Relative	0.28	0.283	1	0.0011		
	3	GNSS	1	Absolute	1.9		0.01	0.005		
		Accuracy								
3	1	Roll		Relative	0.07	0.096	1	0.071	0.046	95.39
	2	Pitch		Relative	0.27	0.283	1	0.017		
	3	GNSS	1	Absolute	1.5		0.01	0.0036		
		Accuracy								

In the above example, **AP2** is considered to be **the best**, since it has the **highest total score**. Even though its **GNSS accuracy** is the worst of all 3, its values for pitch and roll are the ones with the lower deviation from the mean.

2.7.2.4 Config menu

In this panel the parameters of the arbiter algorithm are set.

CEX2PE	DI Builder - CEX2 45010 - CONNECTED	- ×
	🚠 Arbitration	8 2 6 9
Ö	Arbitration Config CAN Setup	
	Arbiter	
ھ ج	Preferred Autopilot 1	
	Method Always best 2	
P	Holding CAP time 0.1 s 3	
œ	Hysteresis 0.001 x1 4	
$\ \ \bigcirc$	Use external autopilot 5	
#	Variables (7)	
	- Variable 1 Max error: 0.0 Weight: 1.0 boolute: 0 + 6	
	Variable 2 Max error: 0.0 Weight: 1.0 Absolute: 0	
	Variable 3 Max error: 0.0 Weight: 1.0 🗹 Absolute: 0 9	

Fig. 46: Config panel

- 1. **Preferred.** The preferred autopilot will be chosen in case of a score draw. **Fixed while ok** mode will always select this autopilot first.
- 2. Method. The method of arbitration can be chosen by the user. The available options are:
 - Always best. It chooses always the best autopilot.
 - **Change if worst.** The arbiter will only switch if the currently selected autopilot has the worst score. In that case, it will switch to the one with the best score.
 - **Round robin control.** Using the **Holding CAP** time parameter, the **arbiter** will periodically switch between autopilots. This mode is meant for testing purposes only.
 - Fixed. Arbitration is disabled and one autopilot is selected. In this mode Autopilot 4x will behave as an Autopilot 1x.
 - Fixed 0. Autopilot 1 is selected.
 - Fixed 1. Autopilot 2 is selected.
 - Fixed 2. Autopilot 3 is selected.
 - **Fixed while ok.** This mode does not take into account scores. In this mode, the **Preferred** autopilot will be selected by default. A switch will only happen if the current autopilot is considered **Dead**.
- 3. Holding CAP time. Amount of time needed from last switch in order to allow a new switch.
- 4. **Hysteresis.** When comparing scores, the difference between them needs to be bigger than this proportional value, in order to assess scores. The difference is proportional to the score of the selected autopilot.

i.e. If current selected autopilot is the number 1, arbitration mode is **Always best**, hysteresis is **0.5** and score for AP 1 is **0.3**, AP 2 will need a score lower than **0.15** in order to be selected.

- 5. Enable arbitration of external autopilot.
- 6. Add variables.
- 7. Variable Max error. Arbitration maximum error for each variable.
- 8. Variable Weight. Arbitration weight for each variable.
- 9. Variable Absolute. If it is enabled, it will indicate that it is an absolute variable. The value set here will be used to compare the variables in order to choose the best autopilot.

2.7.3 CAN Setup

Here users can configure the receiving CAN Ids for each of the 3 possible Veronte Autopilots 1x that are sending data to CEX. Therefore, to send data from any of them to the CEX, these Ids must be specified.

Note: If arbitration is not enabled, only the configuration of the Autopilot 1 will be used.

CEX2PDI	Builder - CEX2 45010 - CONNECTED	
	Arbitration Config CAN Setup	

Fig. 47: CAN Setup panel

1. Send status and Period: The user can enable status sending and set a status message period.

Note: The *Send Status Period* must be set to **0.2 s**, otherwise, the arbitration algorithm will force the selection of autopilot 1.

- 2. Send score and Period: Score sending can be enabled and a period of scoring messages can be set.
- 3. Status message ID: CAN ID to which Status and Score messages are sent when there is no External Arbitration and therefore CEX works as arbiter.
 - **Extended**: If enabled, the frame format will be this, '**Extended**', i.e. with a 29-bit identifier. Otherwise, the frame format 'Standard' (11-bit identifier) is set by default.
- 4. CAN IDs used for the reception of autopilot 4x arbitration messages for each AP.
 - **Extended**: If enabled, the frame format will be this, '**Extended**', i.e. with a 29-bit identifier. Otherwise, the frame format 'Standard' (11-bit identifier) is set by default.

Once the installation is finished, open CEX PDI Builder and select the unit.

EX2 PDI Builder - (v.6.8.73)			-		×
CEX2					
Build PDI to configure your CEX2					
	.	CEV	2		
🛨 Upload PDI		en CEX			
Upload PDI to the CEX2, this option can't be undone	Open PD	l online and	l work with it		
		_			
Not connected		Not	selected		*
		Not s	selected		
		CEX2	45010 (6.8.73))	

Fig. 48: CEX ID

If it is correctly connected, it should appear in Normal mode, as shown in the figure below, or Maintenance mode.

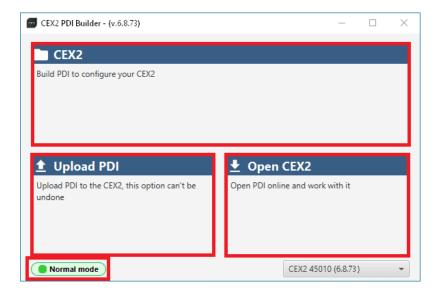


Fig. 49: CEX PDI Builder

CEX unit can also appears as: Maintenance mode (loaded with errors) or Normal mode - Disconnected.

Note: Maintenance mode (loaded with errors) appears when something is wrong in the configuration.

The user can access now to 3 configuration options:

- **CEX**: It allows the user to work with **offline** configurations. A previously exported CEX PDI can be opened and modified or it is possible to build a new one from the default configuration.
- Upload PDI: A previously exported CEX PDI configuration can be imported to the linked CEX.

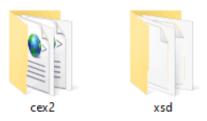
Warning: When a configuration is loaded into CEX with a version older than the software version being used, an **automatic migration** from the configuration version to the software version being used will be performed.

For more information on this, see section *Migrate configuration -> Troubleshooting* in this manual.

• **Open CEX**: By clicking on this option, CEX PDI Builder configuration menu opens with the configuration (the PDI files) loaded in the CEX. Then, the user can modify it **online**.

Note: PDI files are CEX configuration files. These files allow for modular control with improved version management. These PDI files are split in 2 folders. Each folder hold several .xml files:

- cex2: This contains the configuration of the CEX. All the control system and its parameters are stored here.
- **xsd**: This folder holds .xsd files. An XSD file is a definition file specifying the elements and attributes that can be part of an XML document. This ensures that data is properly interpreted, and errors are caught, resulting in appropriate XML validation. Users should never delete, replace or modify it.





Finally, click on 'CEX' to edit a configuration offline or 'Open CEX' to open the configuration and start editing it online.

Note: When CEX PDI is open, the unit changes to **Maintenance mode**. In order for this to be possible, the user must first accept the confirmation panel below.

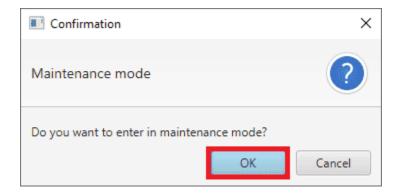


Fig. 51: Enter maintenance mode - Confirmation panel

SEX2 PDI Builder - (v.6.8.73)		×	CEX2PDI Builder - CE	EX2 45010 - CONNECTED	
CEX2				CEX2 PDI Builder	
Build PDI to configure your CEX2			Ö		
			<u></u>		
			•		
1 Upload PDI	生 Open CEX2				
Upload PDI to the CEX2, this option can't be	Open PDI online and work with it		5		
undone			$\begin{tabular}{ c c c c c } \hline \hline \end{tabular}$		
			Æ		
Maintenance mode	CEX2 45010 (6.8.71)	-			

Fig. 52: Open CEX

The different 'buttons' that can be seen in the initial menu of the CEX PDI builder are explained below.

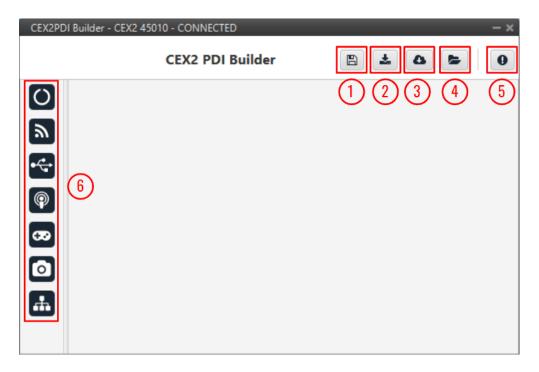


Fig. 53: Initial menu

1. **Save PDI**: After changes are done, press on the save button to apply the changes in the PDI files. While saving, a percentage of saving process is displayed.

In order to save the configuration in the CEX unit it is necessary to **RESET**, therefore the **CEX PDI Builder** software will **close**. For this reason, the user must accept the following panels:

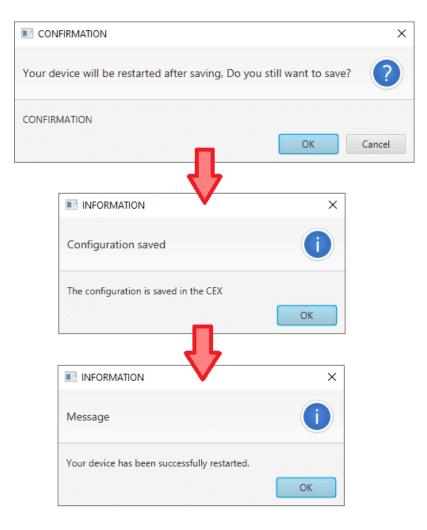


Fig. 54: Confirmation panel - Save

Danger: As CEX is reset, it is not advisable to save changes during flight tests.

Note: This button will only appear if a CEX unit is connected, i.e. when working offline this button will not be available.

- 2. **Export PDI**: After modifying a configuration, press the export button to store the configuration in the local storage. Users can store this configuration in an empty folder or in the folder where the previously imported configuration is stored. With the latter option, the "original" configuration will be overwritten by the one with the new changes.
- 3. **Import PDI from repo**: The user can import a configuration file from the repo and modify it. After that, if the save button is pressed, this configuration will be uploaded in the CEX.
- 4. **Import PDI from local storage**: The user can import a configuration file from the local storage and modify it. After that, if the save button is pressed, this configuration will be loaded into the CEX.
- 5. Feedback: Users can report a problem they have encountered by creating an issue in their own 'Joint Collaboration Framework'. The 'Download' button downloads a zipped folder with the current CEX

configuration and more information needed for Embention to resolve the issue. It is advisable to attach this folder when creating the issue.

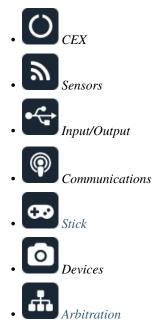
Note: The user's 'Joint Collaboration Framework' is simply a Github repository for each customer.

If the user has any questions about this Joint Collaboration Framework, please see Joint Collaboration Framework user manual or contact sales@embention.com.

FeedBack	_	1
Ser	nd us your feedback	
Repo:*	Your repo	
Title:		
🛓 Dow	vnload 👩 Send	

Fig. 55: Feedback

6. These are the different functions of CEX. They are explained in the following sections.



CHAPTER

THREE

INTEGRATION EXAMPLES

3.1 CAN Isolator

CEX can operate as a CAN bus isolator, since it manages both CAN buses as desired. The system has a built-in microcontroller; which manages CAN buses in real-time. Both buses are not electrically connected, in consequence, electrical signals that do not follow the CAN protocol will be isolated.

The functionalities of a CAN isolator are the following:

- CAN input filter
- Filtered CAN subnet
- CAN tunnel

The following diagram summarizes the behavior of CEX as a CAN bus isolator:

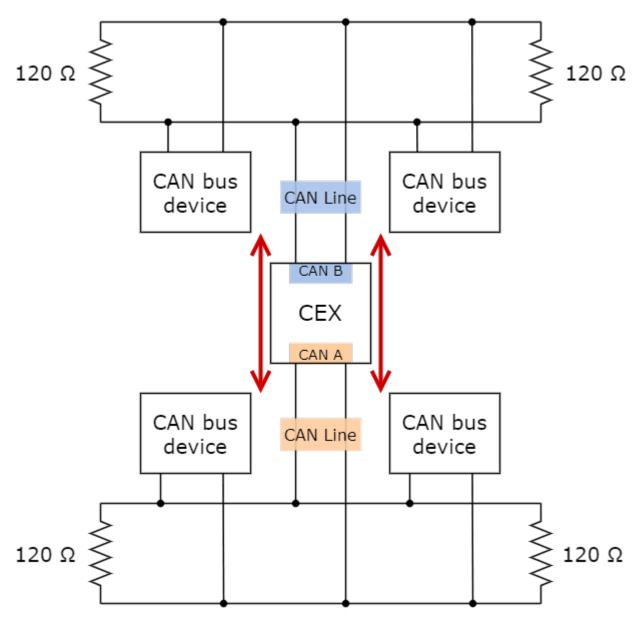


Fig. 1: Subnets CAN diagram

3.1.1 Filtered CAN subnet

With **CEX** it is possible to isolate CAN nets by **filtering certain messages** from one CAN line and only transmitting specific information through CEX to the other one.

In this example, only a certain range of CAN IDs will be allowed to cross from one CAN line to the other using the two CAN ports of **CEX**. Specifically, information will pass **from CAN B** of CEX **to CAN A** of CEX.

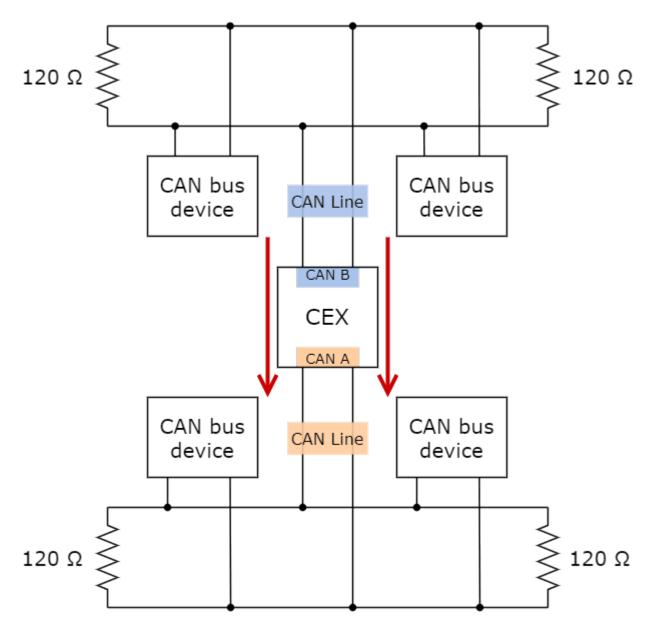


Fig. 2: Subnets CAN example - From CAN B to CAN A

The allowed range will be from **0x550** to **0x55F**.

1. Create a new mailbox entry for CAN B.

To do this, go to Input/Output menu \rightarrow CAN Setup panel \rightarrow CAN B tab.

Assign some of the mailboxes to it and set the ID to **0x550**. This example uses an ID expressed in **hexadecimal** notation.

CEX2 PDI Builder				🗙
	•€+ Input	t / Output		4 6 4 0
GPIO PWM I/O Setup CAN I/O Digital Input SCI CAN Setup CAN Setup CAN Setup	CANA CANB Units ID HEX - Mailboxes reserved RX:	Units Mask BIN 8 Mailboxes available 1	Baudrate 500000.0	
	<)	3

Fig. 3: Filtered CAN subnet - Mailbox configuration

2. Set a Mask which will ignore the last 4 bits.

Note: Although the ID has been entered in hexadecimal notation, note that the Mask has been set in **binary** format.

CEX2 PDI Builder		🚓 Input	: / Outpu	ıt		- ± 4 5 9
GPIO PWM I/O Setup CAN I/O	CANA (Units ID Mailboxe	HEX -		ask BIN es available	Baudrate 500000.0 TX: 24	
Digital Input SCI	#	Mailboxes 4	Extended	ID 516	Mask	+
I/O Setup CAN I/O Digital Input SCI CAN Setup Image: CAN Setup Image: CAN Setup	2	4		550	11111110000	

Fig. 4: Filtered CAN subnet - Mailbox mask configuration

3. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow Configuration tab.

Configure CAN Input Filter 2 on CAN B, with the same settings as the Mailbox.

Important: This menu only allows **decimal numbers**. For this example the Id 0x550, is represented as **1360** in decimal format, and the Mask is also represented in decimal format as **2032**.

GPIO		n CAN Telemetry 1 CAN Telem		
PWM I/O Setup	Priority High	Producer	CAN Output Filter 2	Q ^o
CAN I/O			CAR Output Intel 2	
Digital Input	High 💭		CAN to Serial 1	00
sci	🗌 High 📫		Application Processor	00
CAN Setup	📃 High 🕻	Id 1360	None	08
	🗌 High 🕻	Mask 2032	dec	OS .
	🔲 High	Filter type Standard	▼ None	08
	🗌 High 🕻	CAN Input Filter 6		¢\$
	🔲 High 🚺	CAN unwrapper 1	None	¢\$
	High	CAN unwrapper 2	None	O\$
	🗌 High 🕏	Serial to CAN 1	CAN Output Filter 1	00
	🔲 High 🚺	Serial to CAN 2	None	00
	🔲 High 🚺	Serial to CAN 3	None	00
	High	CAN Custom Producer 1 Te	Ie	08
	High 🚺	CAN Custom Producer 2 Te	Ie	00
	High 🗘	CAN Custom Producer 3 Te	e> None	08

Fig. 5: Filtered CAN subnet - CAN Input Filter configuration

4. Bind CAN Output Filter 2 to CAN Input Filter 2 and, configure the CAN Output Filter to CAN A.

GPIO	Configuration	CAN Telemetry 1 CAN Telemet	y 2 CAN	Telemetry 3			
PWM	Priority	Producer		Consumer			
I/O Setup	High 🗰	Application Processor	\rightarrow	None	00		
CAN I/O	High 🗱	CAN Input Filter 1	\rightarrow	CAN to Serial 1	Q0		
Digital Input SCI	🗌 Higł 🗮	CAN Input Filter 2	→	CAN Output Filter 2	\$	Port	CAN A
CAN Setup	🗌 High 🗱	CAN Input Filter 3	\rightarrow	None	00		
	🗌 High 🗰	CAN Input Filter 4	\rightarrow	None	00		
	🗌 High 🙀	CAN Input Filter 5	\rightarrow	None	00		
	🗌 High 🙀	CAN Input Filter 6	\rightarrow	None	00		
	High 🗱	CAN unwrapper 1	\rightarrow	None	00		
	High 🗱	CAN unwrapper 2	\rightarrow	None	00		
	🗌 High 🛱	Serial to CAN 1	-	CAN Output Filter 1	\dot{Q}^0_0		
	🗌 High 🙀	Serial to CAN 2	\rightarrow	None	Q0		
	🗌 High 🙀	Serial to CAN 3	\rightarrow	None	Q0		
	🗌 High 🔯	CAN Custom Producer 1 Tele	\rightarrow	None	Q0		
	High 🗰	CAN Custom Producer 2 Tele	\rightarrow	None	00		
	High 🔯	CAN Custom Producer 3 Tele	\rightarrow	None	00		

Fig. 6: Filtered CAN subnet - CAN Output Filter configuration

3.1.2 CAN tunnel

CAN tunnel is a specific type of message forwarding; where messages are tunneled from one CAN port of CEX to the other.

In this example, a transparent tunnel will be created. So, any messages received on **Interface A** will be sent through **Interface B**:

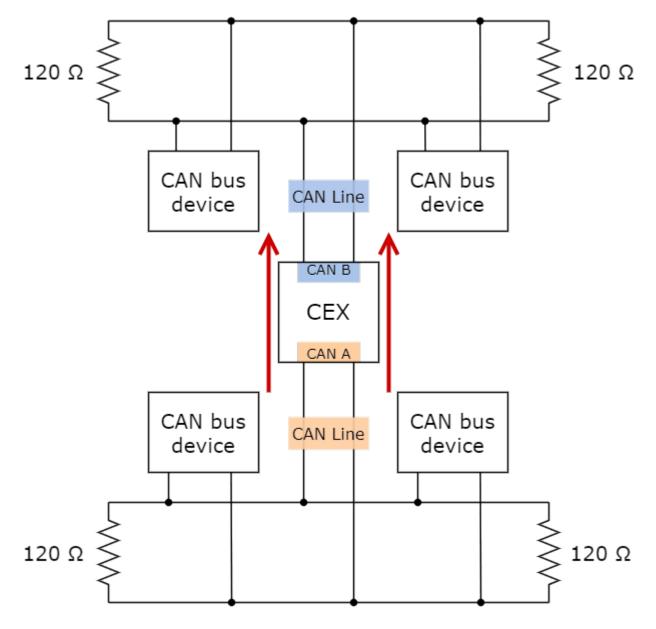


Fig. 7: Subnets CAN example - From CAN A to CAN B

Optionally, the mailboxes can be equally distributed to support both standard and extended CAN IDs.

1. Create a new mailbox entry for Interface A.

To do this, go to Input/Output menu \rightarrow CAN Setup panel \rightarrow CAN A tab.

Assign half of the mailboxes to it and set a **Mask** of 0.

CEX2 PD	l Builder		×
		• 🖕 Input / Output	* • • •
3- 0 8 ® 1/2 %	GPIO PWM I/O Setup CAN I/O Digital Input SCI CAN Setup	Imput / Output CANA_CANB Units ID Units ID Units ID Units ID Units ID Units ID Units Mask BIN< Baudrate Mailboxes reserved RX: 20 Mailboxes available TX: 12 # Mailboxes 1 4 1302 1111111111 2 8 0 3 8 0	
		x () x	

Fig. 8: CAN tunnel - Mailbox configuration

2. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow Configuration tab.

Configure CAN Input Filter 2 on CAN A, with CAN id 0, a Mask of 0 and Both as frame format type.

GPIO	Configuratio	CAN Telemetr	y 1 CAN Telemetry 2	CAN Tel	lemetry 3	
PWM	Priority	Prod		Co	onsumer	
I/O Setup	High K	X ^p Applica	tion Processor	→	None	O ₀
CAN I/O Digital Input	📄 High				CAN to Serial 1	08
SCI	📄 High 📢	Port	CAN A	-	AN Output Filter 2	¢\$
CAN Setup	High 🕻	Ko Id	0		None	00
	📃 High 🕻	K Mask	0	dec	None	08
	High 🕻	Filter type	Both	-	None	CC ⁰
	High 🕻	CAN	Input Filter 6	→	None	00
	High 📢	K CAN	unwrapper 1	→	None	08
	High 🕻	Ka CAN	unwrapper 2	→	None	00
	🗌 High 📢	X ^a Seri	al to CAN 1	→	CAN Output Filter 1	Q0
	High 🕻	Seri-	al to CAN 2	→	None	OC .
	High 📢	Seri-	al to CAN 3	→	None	08
	High 📢	K CAN Custor	m Producer 1 Tele	→	None	08
	🗌 High 📢	CAN Custor	m Producer 2 Tele	→	None	08
	High 🐇	CAN Custor	n Producer 3 Tele	→	None	00

Fig. 9: CAN tunnel - CAN Input Filter configuration

3. Bind CAN Output Filter 2 to CAN Input Filter 2 and, configure the CAN Output Filter to CAN B.

GPIO	Configuration	CAN Telemetry 1 CAN Telemetry	2 CAN	Telemetry 3			
PWM	Priority	Producer	Consumer				
I/O Setup	High 😋	Application Processor	\rightarrow	None	Q ⁰		
CAN I/O Digital Input	High 🕸	CAN Input Filter 1	\rightarrow	CAN to Serial 1	00		
SCI	High 🛱	CAN Input Filter 2	\rightarrow	CAN Output Filter 2	00	Port	CAN B
CAN Setup	High 🛱	CAN Input Filter 3	\rightarrow	None	08		
	High 🔀	CAN Input Filter 4	\rightarrow	None	Q0		
	High 🔀	CAN Input Filter 5	\rightarrow	None	Q0		
	High 🛱	CAN Input Filter 6	\rightarrow	None	08		
	High Q	CAN unwrapper 1	\rightarrow	None	08		
	High 00	CAN unwrapper 2	\rightarrow	None	08		
	High 🔯	Serial to CAN 1	\rightarrow	CAN Output Filter 1	Q0		
	High 🛱	Serial to CAN 2	\rightarrow	None	08		
	🗌 High 🔀	Serial to CAN 3	\rightarrow	None	00		
	High 🔯	CAN Custom Producer 1 Tele	\rightarrow	None	00		
	High 🔯	CAN Custom Producer 2 Tele	\rightarrow	None	00		
	High QS	CAN Custom Producer 3 Tele	\rightarrow	None	08		

Fig. 10: CAN tunnel - CAN Output Filter configuration

3.2 Reading Arbitration Messages

Note that those messages are generated only if arbitration and the messages are enabled in CEX.

Arbiter will send its telemetry in little endian format, using its CAN-TX ID.

The appropriate arbitration message format is described in CAN Bus protocol -> Arbitration section of the CEX Software Manual.

3.3 External devices

The step-by-step instructions for the following external devices are explained in detail in the following sections:

- Jetibox
- Scorpion tribunus
- Veronte products

3.3.1 Jetibox

To enable CEX to simulate a Jetibox to read telemetry from legacy Jeti devices, the following steps need to be taken:

1. Go to Devices menu \rightarrow **Jetibox panel**.

Enable it and select a **SCI Port**, in this example SCI A is selected:

CEX2 PDI Builder		- ×
	D evices	* & = 0
 ✓ Jetibox Scorpion tribunus ✓ <	SCI Port SCI A	

Fig. 11: Jetibox - Devices configuration

2. Go to Input/Output menu \rightarrow **SCI panel**.

Configure the serial port (the one selected in step 1) as follows:

- Baudrate: 9800
- Length: 8
- Stop: 2 stop bits
- Parity: Odd
- Use address mode: Enabled

CEX2 PE	01 Builder			– × –					
	•Carl Input / Output								
\bigcirc	GPIO	SCI A SCI B SCI C							
	PWM								
2	I/O Setup	Functionality							
	CAN I/O								
	Digital Input	Baudrate 9800	-						
0	SCI	Length 8	-						
	CAN Setup	Stop 2	•						
		Parity Odd	-						
6		Use address mode							
		U ose address mode							
.fh									

Fig. 12: Jetibox - SCI configuration

Note: The serial port will be totally reserved for this, so it will not be usable to other things and the IO Manager affecting it will be ignored.

3. Go to Input/Output menu \rightarrow I/O Setup panel.

Link the specific Jetibox I/O consumer to the port configured in step 2:

GPIO PWM	Configuration Priority	Configuration Priority Producer Consumer				
I/O Setup	High 🔅		\rightarrow	JETI box	Q ^o	
CAN I/O	High 🗱		\rightarrow	None	00	
Digital Input SCI	High 🗱		\rightarrow	None	OC C	
CAN Setup	High 🔯		\rightarrow	None	OC	
	High 🗱	CAN to Serial 2	\rightarrow	None	OC	
	High 🛱	CAN to Serial 3	\rightarrow	None	QC .	
	High 🛱		\rightarrow	None	Q0	
	🗌 High 🔯	Commgr port 2		UART-B / RS232-B	Q0	
	High 🔯	Commgrf port 3	\rightarrow	None	Q0	
	High 🔯	Commgr port 4	\rightarrow	None	Q ₀	
	High 🔯	Commgr port 5	\rightarrow	None	Q0	
	High 🔯	Commgr port 6	\rightarrow	None	Q ₀	
	High 🔯	Tunnel 1	\rightarrow	None	Q0	
	High 🔯	Tunnel 2	\rightarrow	None	Q0	
	High 🔯	B Tunnel 3	\rightarrow	None	Q ₀	
	High 🛱	CAN wrapper 0 for serial tra	n	None	00	

Fig. 13: Jetibox - Consumer

4. Configure the Jetibox I/O consumer to retrieve the data. Click on the configuration button

 \mathbf{Q}^{o}_{o}

CEX2 PE	DI Builder					-	- ×
		•🛟 Inp	ut / Output			205	0
ال	GPIO	Configuration					
	PWM	Priority	Producer	Consur	ner		
2	I/O Setup	High 🔅	UART-A / RS232-A	→	JETI box		î
•	[Consumer] Consumer JETI box						- ×
	🚥 📑 🗍 🛛 Little endian	Command Jeti Box Nop 👻	Checksum Matcher 0 0	Skip Variable ASCII 0 0 0	Position Occupancy		•
630			,				
÷ © 8 ⊚							
<u>н</u>							
			Memory usag	ge: 0 / 16384 bytes			

Fig. 14: Jetibox - Consumer configuration

Below is an example of the custom messages needed to read the Actual Voltage from a Jeti MasterSpin 220 (use **Big** endian in all messages):

Consumer] Con	sumer JETI box										
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
	Big endian	Jeti Box Down	*	0	2	8	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
- 📑 🗖	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
- 💷 🗆	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
- 📑 🗖	Big endian	Jeti Box Down		0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
• 🖳 🗍	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
- 📑 🗖	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
- 💷 🗆	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
- 💷 🗍	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
- 🖳 🔲	Big endian	Jeti Box Down	*	0	1	0	0	0	0		
		Command		Checksum	Matcher	Skip	Variable	ASCII	Position	Occupancy	
• 💷 🗆 I	Big endian	Jeti Box Nop	*	0	1	8	0	1	0		

Fig. 15: **JETI box example**

- 1. Expected text: "CONTROLLER TYPE MasterSpin 220~"
 - Command: Jeti Box Down
 - Matcher(32) "CONT" 0x434F4E54 (1129270868)
 - Skip(24*8) 192
 - Matcher(32) "220~" 0x3232307E (842150014)

[Consumer] Consumer JETI box		- ×
- III Big endian Jeti Box Down	Checksum Matcher Skip Variable ASCII Position	Occupancy
		^ +
— 📑 🗍 🕂 Matcher x434F4E54		
Value Bits Mask		
1129270868 32 4294967295	dec	
Bits		
24		
— 📑 🗍 💠 Skip 24		
— 🗐 🕂 Skip 24		
— 📑 🗇 💠 Skip 24		
— 📑 🗇 💠 Skip 24		
— 📑 🗇 💠 Skip 24		
— 📑 🗇 💠 Skip 24		
— 📑 🗍 🕂 Skip 24		
— 📑 🗍 💠 Matcher x3232307E		
Value Bits Mask		
842150014 32 4294967295	dec	•
	Memory usage: 256 / 16384 bytes	

Fig. 16: JETI box custom message example

- 2. Expected text: "MeasureOrSetting MEASURE ~"
 - Command: Jeti Box Down
 - Matcher(32) "Meas" 0x4D656173 (1298489715)
- 3. Expected text: "Max Temperature"...
 - Command: Jeti Box Down
 - Matcher(32) "Max" 0x4D617820 (1298233376)
- 4. Expected text: "Min Temperature"...
 - Command: Jeti Box Down
 - Matcher(32) "Min" 0x4D696E20 (1298755104)
- 5. Expected text: "Actual Temperatu"...
 - Command: Jeti Box Down
 - Matcher(32) "Actu" 0x41637475 (1097036917)
- 6. Expected text: "MaxCurrent"...
 - Command: Jeti Box Down

- Matcher(32) "MaxC" 0x4D617843 (1298233411)
- 7. Expected text: "MinCurrent"...
 - Command: Jeti Box Down
 - Matcher(32) "MinC" 0x4D696E43 (1298755139)
- 8. Expected text: "Max Voltage"...
 - Command: Jeti Box Down
 - Matcher(32) "Max" 0x4D617820 (1298233376)
- 9. Expected text: "Min Voltage"...
 - Command: Jeti Box Down
 - Matcher(32) "Min" 0x4D696E20 (1298755104)
- 10. Expected text: "Actual Voltage 11,86 V "
 - Command: Jeti Box Nop
 - Matcher(32) "Actu" 0x41637475 (1097036917)
 - Skip(12*8) 96
 - Parse ascii: int(2), decimal(2), separartor(`,')

[Consumer] Consumer JETI box			- ×
- E Big endian Jeti Box Down	• 0 1 0	0 0 0	^ 🕇
	Checksum Matcher Skip	Variable ASCII Position Occupancy	
🗕 🤯 🗍 🛛 Big endian 🛛 Jeti Box Nop	• 0 1 8	0 1 0	
— 📑 🗍 🕂 Matcher x41637475			^ +
Value Bits Mask			
1097036917 32 4294967295	dec		
- 📑 🗍 🕂 Skip 12			
Bits			
12			
— 📑 🗍 💠 Skip 12			
— 📑 🗍 💠 Skip 12			
— 📪 🗍 💠 Skip 12			
— 🗊 🗇 💠 Skip 12			
- 🗐 🕂 Skip 12			
— 🗐 🗍 🕂 Skip 12			
— 🗐 🕂 Skip 12			
		1	
Variable Char in inter part	Char in decimal part Division char		
· · ·		-	
User Variabl 2	2 .		` ↓
	Memory usage: 0 /	16384 bytes	

Fig. 17: JETI box last custom message example

3.3.2 Scorpion tribunus

As CEX can read telemetry from Tribunus ESCs by connecting it to one of its serial ports, the following steps are necessary to configure it:

1. Go to Devices menu \rightarrow **Scorpion tribunus** and enable it.

CEX2 PDI Builder			~~ X
	O Devi	ces	
Jetibox Scorpion tribunus	Enable		
	Telemetry period	0.02	
	MCU Telemetry 1 period	0.05	
	MCU Telemetry 2 period	0.05	
	PWM ID	0	
Jetibox Scorpion tribunus Image: Constraint of the second			

Fig. 18: Scorpion tribunus - Devices configuration

2. Go to Input/Output menu \rightarrow I/O Setup panel.

Link the specific Tribunus ESC consumer to the desired port:

l Builder	• ~ +	nput / Output			± 0 Þ
GPIO PWM	Configuration			-	
I/O Setup	Priority	Producer UART-A / RS232-A		Consumer Commgr port 2	00
CAN I/O	High	UART-B / RS232-B		Commgr port 3	OC .
Digital Input SCI	High 😂	UART-C / RS485]→[Tribunus ESC	Q ⁰
CAN Setup	High 🔯	CAN to Serial 1	\rightarrow	Commgr port 1	Q ₀
	High 🔯	CAN to Serial 2	\rightarrow	None	Ø₀ ^o
	High 🔯	CAN to Serial 3	\rightarrow	None	O ₀
	High 🔯	Commgr port 1]→	Serial to CAN 1	O ₀
	High 🔯	Commgr port 2]→	UART-A / RS232-A	O ₀
	High 🔯	Commgrf port 3	→	UART-B / RS232-B	O ₀
	High 🔯	Commgr port 4	\rightarrow	None	00
	High 🔯	Commgr port 5	\rightarrow	None	O ₀
	High 🔯	Commgr port 6	\rightarrow	None	OC .
	High 🔯	Tunnel 1	\rightarrow	None	OC .
	High 😡	Tunnel 2	\rightarrow	None	OS
	High 🔯	Tunnel 3	\rightarrow	None	00
	High 🔯	CAN wrapper 0 for serial tran	\rightarrow	None	00
	High 100	CAN wrapper 1 for serial trap		None	68

Fig. 19: Scorpion tribunus - I/O Setup configuration

3.3.3 Veronte products

3.3.3.1 CAN communication

CEX can send telemetry via CAN.

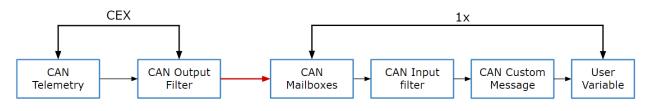


Fig. 20: Communication diagram 1x-CEX

The user must follow the steps below to learn how to enable this function and also how to configure an Autopilot 1x to read this telemetry:

3.3.3.1.1 CEX PDI Builder side

1. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow Configuration tab.

Connect a CAN Custom Producer Telemetry to a CAN Output Filter as follows:

SPIO	Configura	ation	CAN Telemetry 1 CAN Telemetry	2 CAN	Telemetry 3	
WM	Priority		Producer		Consumer	
/O Setup	High	00	Application Processor	\rightarrow	CAN Output Filter 2	00
CAN I/O	High	Q^0_0	CAN Input Filter 1	\rightarrow	CAN to Serial 1	OC .
Digital Input SCI	🗌 High	Q^0_0	CAN Input Filter 2	→	Application Processor	Q0
CAN Setup	High	Q_0^0	CAN Input Filter 3	\rightarrow	None	00
	High	Q_0^0	CAN Input Filter 4	\rightarrow	None	O ₀
	High	Q_0^0	CAN Input Filter 5	\rightarrow	None	Q0
	High	Q^0_0	CAN Input Filter 6	\rightarrow	None	Q0
	- High	Q^0_0	CAN unwrapper 1	\rightarrow	None	00
	High	Q^0_0	CAN unwrapper 2	\rightarrow	None	00
	High	Q_0^0	Serial to CAN 1]→[CAN Output Filter 1	Q0
	📃 High	Q^0_0	Serial to CAN 2	\rightarrow	None	00
	High	\mathbf{Q}_0^0	Serial to CAN 3	\rightarrow	None	Q0
	🔜 High	Ω_0^0	CAN Custom Producer 1 Tele	→	CAN Output Filter 3	Q ⁰
	High	Q_0^0	CAN Custom Producer 2 Tele	\rightarrow	None	00
	High	00	CAN Custom Producer 3 Tele	\rightarrow	None	08

Fig. 21: CAN I/O Custom Telemetry

2. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow CAN Telemetry tab 1 (as the Producer is CAN Custom Producer 1 Telemetry).

Select the fields to send in **TX** or **TX** Ini, as it is a Producer. More information on the configuration of Telemetry messages can be found in the *CAN Telemetry section*.

Note: To configure the telemetry received on a **CAN Custom Consumer Telemetry**, the fields to be read must be configured in **RX**.

CEX2 PDI Builder		- ×
	• 😭 Input / Output	± 4 5 0
GPIO PWM I/O Setup CAN I/O Digital Input CI CAN Setup	Configuration CAN Telemetry 1 CAN Telemetry 2 CAN Telemetry 3 TX Ini TX EXT Can ID: 90 Little endian Period 1.0 s KX KX Memory usage: 0 / 16384 bytes	

Fig. 22: CAN Custom Telemetry

For example, a telemetry messsage with a variable set to ID 90:

CAN message ID: 9 Checksum Matcher		Occupancy			- ×
0 0	0 1 0 0				
					+
Varia PWI	able Compressi	on Decimals End	ode/Decode Encode Min Max	Decode Min Max	

Fig. 23: CAN Custom Telemetry example

3.3.3.1.2 1x PDI Builder side

3. Go to UI menu \rightarrow Variables panel \rightarrow **Real Vars tab**.

Rename a User Variable that will be used to store the value received from CEX:

onte v4.8 👻		🚯 UI		-	4 6 5
Operation variables Variables	Bits	Insigned Real Vars Featur	res		
Geoid	Searc	h Default name	Custom name	Show unit	Init value (SI)
3	3100	User Variable 01 (Real	CEX PWM 1	Sustom Type ()	
	3101	User Variable 02 (Real		Custom Type ()	
	3102 3103	User Variable 03 (Real User Variable 04 (Real		Custom Type () Custom Type ()	
	3104	User Variable 05 (Real		Custom Type ()	
3	3105	User Variable 06 (Real User Variable 07 (Real		Custom Type () Custom Type ()	
	3107	User Variable 08 (Real		Custom Type ()	
)	3108	User Variable 09 (Real		Custom Type ()	
	3109	User Variable 10 (Real User Variable 11 (Real		Custom Type () Custom Type ()	
<u>1</u>	3111	User Variable 12 (Real		Custom Type ()	
1	3112	User Variable 13 (Real		Custom Type ()	

Fig. 24: 1x PDI Builder - User Variable renamed

4. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Mailboxes tab.

Configure the mailbox to receive the message with ID 90 in **1x PDI Builder**:

/eronte v4.8 👻	•🕁 Input / Output	
CAN Setup	Configuration Custom message 1 Custom message 2 Custom message 3 Mailboxes	
Digital Input	CAN A CAN B	
a	Units ID DEC - Units Mask BIN - Baudrate 1000000.0	
~	Mailboxes reserved RX: 6 Mailboxes available TX: 26	
	# Mailboxes Extended ID Mask + 1 5 1301 1111111111	
*	2 — 1 90 1111111111	
2		
2		
20		
×		

Fig. 25: 1x PDI Builder - Mailbox configuration

5. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Configuration tab.

Connect an Input filter with the right CAN ID to a Custom message consumer:

l Builder			enuvzenuvzenuvz						
		• 🙀 Input / Output							
I/O Setup	Configuration	Custom message 1 Custom m	essage 2	Custom message 3 Mailboxe	s				
CAN Setup	Priority	Producer		Consumer					
Digital Input	High 🕸	Serial to CAN 1	\rightarrow	Output filter 1	Q_0°				
	High 🛱	Serial to CAN 2	\rightarrow	None	¢¢				
	High 🛱	Serial to CAN 3	\rightarrow	None	¢¢₀				
	High 🔯	Serial to CAN 4	\rightarrow	None	Q0				
	High 🔯	Serial to CAN 5	\rightarrow	None	Q0				
	High 🛱	Serial to CAN 6	\rightarrow	None	¢₀°				
	High 🔅	CAN custom message 1	\rightarrow	Output filter 2	Q ^o				
	High 🔅	CAN custom message 2	\rightarrow	Output filter 3	$\dot{\mathbf{Q}}^{o}_{o}$				
	High 🗘	CAN custom message 3	\rightarrow	None	Q ^o				
	High 🔯	Input filter 1	\rightarrow	CAN to serial 1	¢¢₀				
	High 🛱	Input filter 2	\rightarrow	None	¢¢				
	High 🛱	Input filter 3	\rightarrow	Custom message 1	¢8				
	High 😋	Input filter 4	\rightarrow	None	Q ₀ ^o				
	High 🛱	Input filter 5	\rightarrow	None	Q0				
	High 🛱	Input filter 6	\rightarrow	None	Q ⁰ ₀				
	High 🔯	CAN unwrapper 1	\rightarrow	None	Q0				
	High 88	CAN unwranner 2		None	2550				

Fig. 26: 1x PDI Builder - Input filter

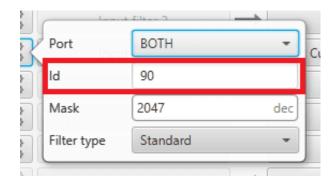


Fig. 27: 1x PDI Builder - Input filter configuration

6. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Custom message 1 tab (as Custom Message 1 has been selected as consumer).

Configure the reading of the message and store the received value in the user variable renamed above:

Veronte	PDI Builder		- ×
Veronte	v4.8 •	• 🛟 Input / Output	•
0	I/O Setup CAN Setup	Configuration Custom message 1 Custom message 2 Custom message 3 Mailboxes	
	Digital Input	 TX Ini TX 	
<u></u>		▼ RX	
		EXT Can ID: 90 Little endian Time out 1.0 s Bit ID: 0	+
	CAN message ID: 90 Checksum Matcher Skip Variabl		- ×
P	0 0 0 1	0 0	+
∞	Variable	Compression Decimals Encode/Decode Encode Decode Min Max Min Max	
	CEX PWM 1 Un	■ Compress • 0 1 0.0 0.0 0 0	

Fig. 28: 1x PDI Builder - Custom Message configuration

3.3.3.2 Commanding/Reading PWMs

The appropriate PWM message format is described in CAN Bus protocol -> Command PWMs of the **CEX Software Manuals**.

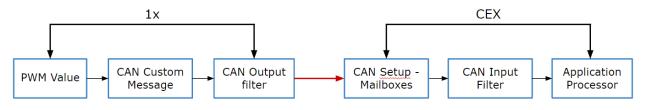


Fig. 29: Communication diagram 1x-CEX

The user can follow the following steps to achieve PWM commanding from 1x to CEX:

3.3.3.2.1 1x PDI Builder side

1. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Configuration tab.

Connect a CAN custom message to an Output filter to send the message:

O Setup	Configuratio	on Custom message	e 1 Custom me	essage 2	Custom message 3 Mailboxe	s
V Setup	Priority	Produc	er	_	Consumer	
al Input	High 🚦	Serial t	o CAN 1	\rightarrow	Output filter 1	\mathbf{Q}^{o}_{o}
	High	Serial t	o CAN 2	\rightarrow	None	Q ₀ ⁰
	High	Serial t	o CAN 3	\rightarrow	None	Q ₀ ^o
	High	Serial t	o CAN 4	\rightarrow	None	Q ₀ ^o
	High	Serial t	o CAN 5	\rightarrow	None	Q_0^0
	High	Serial t	O CAN 6	\rightarrow	None	Q_0^0
	High	CAN custo	m message 1	\rightarrow	Output filter 2	Q^0_0
	High 3	CAN custo	m message 2	→	Output filter 3	\mathbf{Q}_{0}^{0}
	High 🖁	CAN custo	m message 3	\rightarrow	None	Q ₀
	High	nput	filter 1	\rightarrow	CAN to serial 1	Q ₀ ⁰
	High	nput	filter 2	→	Custom message 1	Q_0^0
	High	🗱 Input	filter 3	\rightarrow	Custom message 2	Q_0^0
	High	Input	filter 4	\rightarrow	None	Q_0^0
	High	input	filter 5	\rightarrow	None	Q_0^0
	High	Input	filter 6	\rightarrow	None	Q ₀ ⁰
	High	CAN un	wrapper 1	\rightarrow	None	00
		AND CAN UP			News	atelli

Fig. 30: PWM Command - CAN Custom Message

2. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Custom Message 2 tab (as Custom Message 2 has been selected as producer).

Build a CAN custom message using the PWM variable and the appropriate Message format. CAN ID is arbitrary, for this example ID 100 will be used:

Veronte PDI Builder		- ×
Veronte v4.8 👻	• 🔂 Input / Output	0
CAN Setup	Configuration Custom message 1 Custom message 2 Custom message 3 Mailboxes	
Digital Input	► TX Ini ▼ TX	
CAN message ID: 100 Checksum Matcher Skip Vari	EXT Can ID: 100 Little endian Period 0.01 s	+
CAN message ID: 100 Checksum Matcher Skip Vari	able ASCII Position Occupancy	- ×
		+
	255 dec	
🗰 — 📑 🗍 🕂 PWM 1	Encode Decode	
Variable PWM 1	Compression Bits Encode/Decode Min Max Min Max Compress - Bits Unsigned - 12 1 0.0 1.0 0 4095	
8		
X U		
< [>

Fig. 31: PWM Command - CAN Custom Message configuration

Message format:

- Matcher (2) [8 bits]
 - PWM 1 [12 bits]
 - PWM 2 [12 bits]
 - PWM 3 [12 bits]
 - PWM 4 [12 bits]
- Matcher (3) [8 bits]
 - PWM 5 [12 bits]
 - PWM 6 [12 bits]
 - PWM 7 [12 bits]
 - PWM 8 [12 bits]
- Each PWM variable has to be set using the following format:
 - Variable: PWM X
 - Compression: Compress Bits Unsigned

- **Bits**: 12
- Encode: Min=0.0 / Max=1.0
- Decode: Min=0 / Max=4095

3.3.3.2.2 CEX PDI Builder side

3. Go to Input/Output menu \rightarrow CAN Setup panel.

Create the mailbox to receive the new message (ID 100):

CEX2 PDI Builder			- ×
	•र⊶ Input / Output		
GPIO PWM I/O Setup CAN I/O Digital Input SCI CAN Setup CAN Setup I/O I/O I/O I/O I/O I/O Digital Input SCI I/O I/O	CANA CANB Units ID DEC - Units Mask Mailboxes reserved RX: 5 Mailboxes ava	ailable TX: 27 ID Mask 2 1111111111	
	< [>

Fig. 32: PWM Command- Mailbox configuration

4. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow Configuration tab.

Connect an Input filter with the right CAN ID to the Application Processor consumer:

GPIO	Configurat	tion	CAN Telemetry 1 CAN Telemetry	y 2 CAN	Telemetry 3	
PWM	Priority		Producer		Consumer	
/O Setup	High	\dot{Q}^{0}_{0}	Application Processor	\rightarrow	None	00
CAN I/O	High	Q^0_0	CAN Input Filter 1	\rightarrow	Application Processor	00
Digital Input SCI	High	\mathbf{Q}^{o}_{o}	CAN Input Filter 2	\rightarrow	CAN to Serial 1	
CAN Setup	High	\dot{Q}^{0}_{0}	CAN Input Filter 3	\rightarrow	None	Q0
	High	\dot{Q}^0_0	CAN Input Filter 4	\rightarrow	None	Q0
	High	\dot{Q}^{0}_{0}	CAN Input Filter 5	\rightarrow	None	
	High	\dot{Q}^{0}_{0}	CAN Input Filter 6	\rightarrow	None	Q0
	High	Q_0^0	CAN unwrapper 1	\rightarrow	None	Ø8
	High	Q_0^0	CAN unwrapper 2	\rightarrow	None	Q0
	High	Q^0_0	Serial to CAN 1]→	CAN Output Filter 2	Q_{0}^{0}
	High	Q^0_0	Serial to CAN 2	\rightarrow	None	
	High	Q^0_0	Serial to CAN 3	\rightarrow	None	
	High	Q^{0}_{0}	CAN Custom Producer 1 Tele	\rightarrow	None	Q0
	High	Q^{0}_{0}	CAN Custom Producer 2 Tele	\rightarrow	None	
	High	100	CAN Custom Producer 3 Tele	\rightarrow	None	00

Fig. 33: PWM Command - Input filter

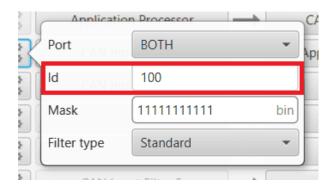


Fig. 34: PWM Command - Input filter configuration

 Go to Input/Output menu → PWM panel → PWM 1 tab (as PWM 1 has been selected in 1x PDI Builder). Configure parameters for PWM 1:

CEX2 PD	l Builder						X
		•< <mark>-</mark> Input	t / Output			*	•
O	GPIO	VWM 1					
	PWM			PWM 1			
2	I/O Setup	Enable					
€÷	CAN I/O	Sub id	0	✓ Active H	ligh		
	Digital Input	Frequency	50	Mode	Time	•	
P	SCI CAN Setup	Timeout	0.1	Min	0.001	s	
679	CAN Setup	Start value	0.001				
		Pulse source ID	0	Max	0.002	s	
\mathbf{O}							
		► PWM 2					
Ē		PWM 3					
		► PWM 4					
		PWM 5					
		► PWM 6					
		PWM 7					
		► PWM 8					

Fig. 35: PWM Command - PWM configuration

6. Finally, export the configuration and save the changes.

3.3.3.3 Connection with Autopilot 1x via CAN

No configuration is necessary in CEX, the configuration for communication with 1x is set by default.

The configuration required in **1x PDI Builder** to communicate with CEX via CAN is explained in the CEX/MEX - Integration Examples section of the **1x PDI Builder** user manual.

3.3.3.4 Connection with Autopilot 4x via CAN

3.3.3.4.1 CAN Reception IDs

First, users can setup the recepcion CAN Ids for each of the 3 possible Veronte Autopilots 1x sending data to CEX in the **Arbitration menu**.

CEX2 PDI	Builder 🚣 Arbitration	- × • • •
8 9 19	Arbitration Arbitration Config CAN Setup Send status Period 0.25 Send score Period 3.0 Status message ID 255 Extended Autopilot 1 Extended ID 8 Autopilot 2 Extended ID 9	
	Autopilot 3 Extended ID 10	

Fig. 36: Arbitration - CAN Ids of autopilots

If **arbitration is not enabled**, and therefore only a 1x autopilot is being used, no CAN Ids need to be configured here.

3.3.3.4.2 CAN I/O Interconnections

Once the CAN IDs are set, users shall configure:

- The Input Filters going to be used (as communication with CEX has to be always through a Filter).
- The connection between input filters and data Consumers.

SPIO	Configuration	CAN Telemetry 1 CAN Telemetry	2 CAN Telemetry 3	
PWM	Priority	Producer	Consumer	
/O Setup	High 🕸	Application Processor	CAN Output Filter 2	00
CAN I/O	High 🕵	CAN Input Filter 1	CAN to Serial 1	00
Digital Input SCI	🗌 High 🔯	CAN Input Filter 2	Application Processor	Q0
CAN Setup	📄 High 🚺	CAN Input Filter 3	None	00°
	High ପ	CAN Input Filter 4	None	00 O
	High 🗘	CAN Input Filter 5	None	00
	🗌 High 🚺	CAN Input Filter 6	None	Q ₀
	High	CAN unwrapper 1	None	00°
	High	CAN unwrapper 2	None	00°
	High 🔯	Serial to CAN 1	CAN Output Filter 1	Q0
	High 🗘	Serial to CAN 2	None	Q0
	High 🗘	Serial to CAN 3	None	00°
	High	CAN Custom Producer 1 Tele	None	Q0
	High 🔯	CAN Custom Producer 2 Tele	None	Q0
	High 🔯	CAN Custom Producer 3 Tele	None	08

Fig. 37: CAN I/O interconnections

For more information on CAN I/O configuration, see section CAN I/O of this manual.

Next step is to connect each of the desired data Producers to an **Output Filter**, and configure both the Producer and the Output Filter:

SPIO	Configura	tion	CAN Telemetry 1 CAN Telemetry	2 CAN	Telemetry 3			
WM	Priority		Producer		Consumer			
/O Setup	High	Q_0^0	Application Processor		CAN Output Filter 2	Q ₀	Port	CAN B
CAN I/O Digital Input	High	Q_0^0	CAN Input Filter 1	\rightarrow	CAN to Serial 1	Q_0^0		
ici	High	Q^0_0	CAN Input Filter 2	\rightarrow	Application Processor	Q_0^0		
CAN Setup	High 📄	Q^0_0	CAN Input Filter 3	\rightarrow	None	Q_0^0		
	High	Q^0_0	CAN Input Filter 4	\rightarrow	None	Q ₀		
	High	Q^0_0	CAN Input Filter 5	\rightarrow	None	Q0		
	High	Q^0_0	CAN Input Filter 6	\rightarrow	None	Q0		
	High	Ω^0_0	CAN unwrapper 1	\rightarrow	None	Q0		
	High	Q^0_0	CAN unwrapper 2	\rightarrow	None	Q0		
	High	Q^0_0	Serial to CAN 1	→	CAN Output Filter 1	Q ₀		
	High	Q^0_0	Serial to CAN 2	\rightarrow	None	Q ₀ ^o		
	High	Q^0_0	Serial to CAN 3	\rightarrow	None	Q ₀ ^o		
	High	Q^0_0	CAN Custom Producer 1 Tele	\rightarrow	None	Q ₀ ⁰		
	High	Q^0_0	CAN Custom Producer 2 Tele	\rightarrow	None	00		
	High	10	CAN Custom Producer 3 Tele	\rightarrow	None	Q0		



3.3.3.5 GPIO Command

The following are the steps to send a GPIO command from the Veronte Autopilot 1x, receive it at CEX and process it, so that CEX carries out the command.

Warning: Remember that for the reception of CAN messages, Mailboxes need to be configured accordingly.

GPIO Command is very similar to PWM command with a few exceptions.

3.3.3.5.1 1x PDI Builder side

1. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Configuration tab.

Connect, as Producer, a 'CAN GPIO remote' to an Output filter:

Setup	Configura	ation	Custom message 1	Custom mess	age 2	Custom message 3	Mailboxes	
l Setup	Priority	T Q	Producer			Consumer		Q
tal Input	High	00	Serial to C	AN 4	→Ì	None		Q ⁰
	High	00	Serial to C	AN 5	→Ì	None		Q ⁰
	High	00	Serial to C	AN 6	→Ì	None		00
	High	00	CAN custom m	essage 1	→Ì	None		Q^0_0
	High	Q^0_0	CAN custom m	essage 2	→Ì	None		Q^0_0
	High	Q^0_0	CAN custom m	essage 3	→[None		Q^0_0
	High	\mathbf{Q}_{0}^{0}	Input filte	er 1	→[CAN to seria	11	Q^0_0
	High	\mathbf{Q}_0^0	Input filte	er 2	\rightarrow	None		Q^0_0
	High	\mathbf{Q}_0^0	Input filte	er 3	→[None		Q^0_0
	High	\mathbf{Q}_0^0	Input filte	er 4	→[None		Q^0_0
	High	\mathbf{Q}_0^0	Input filte	er 5	→[None		Q^0_0
	High	\mathbf{Q}_0^0	Input filte	er 6	→[None		Q^0_0
	High	Q^0_0	CAN unwraj	oper 1	→[None		Q^0_0
	High	Q^0_0	CAN unwraj	oper 2	\rightarrow	None		$\dot{\Omega}^{0}_{0}$

Fig. 39: GPIO Command - 1x CAN Setup

CAN GPIO remote must be configured. For more information on its configuration, see 1x PDI Builder manual -> CAN Setup.

Period	0.1	S
ld of the generat CAN messages	40 de	
Destination	Value	ĵ
Virtual 01	Virtual 01	•
Virtual 02	Virtual 01	-
Virtual 03		-
Virtual 04	Virtual 01	* ~

Fig. 40: GPIO Command - 1x CAN Setup configuration

2. Go to Automations menu.

GPIO must be activated using an 'Output action':

Veronte PDI Builder		- ×
Veronte v4.8 💌	C Automations	
✓ 1 - GPIO Command Image: Second s	GPIO Command Events AND OR NOT 1 - GPIO activated 1 - Activate GPIO	+
	Delay 0.0 s Perio	odical: Off
Image: Constraint of the second s	Virtual 1 Off On Pulse Pulse off Time 0.0 s	

Fig. 41: GPIO Command - 1x Automation configuration

3.3.3.5.2 CEX PDI Builder side

3. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow Configuration tab.

Finally, connect an Input filter to the CAN GPIO consumer:

SPIO	Configuratio	n CAN Telemetry 1 CAN Telem	etry 2 CAN	Telemetry 3	
NWW	Priority	Producer		Consumer	
/O Setup	High 🗘	Application Processor	\rightarrow	None	\$\$°
CAN I/O	🗌 High 🚺	CAN Input Filter 1	\rightarrow	Application Processor	00
Digital Input SCI	🗌 High 🚺	CAN Input Filter 2	→	CAN GPIO consumer	00
CAN Setup	🗌 High 🗘	CAN Input Filter 3	\rightarrow	CAN to Serial 2	00
	🗌 High 🚺	CAN Input Filter 4	\rightarrow	None	Q0
	🗌 High 🚺	CAN Input Filter 5	\rightarrow	None	Q0
	🗌 High 🚺	CAN Input Filter 6	\rightarrow	None	Q0
	High 🗘	CAN unwrapper 1	\rightarrow	None	Q0
	High 🗘	CAN unwrapper 2	\rightarrow	None	Q0
	🗌 High 🚺	Serial to CAN 1	\rightarrow	CAN Output Filter 2	Q ₀
	🗌 High 🚺	Serial to CAN 2	\rightarrow	CAN Output Filter 3	Q0
	🗌 High 🚺	Serial to CAN 3	\rightarrow	None	Q0
	High 🔅	CAN Custom Producer 1 Tele		CAN Output Filter 1	Q ₀
	High 🗘	CAN Custom Producer 2 Tel	\rightarrow	None	Q0
	🗌 High 🔹	CAN Custom Producer 3 Tele	\rightarrow	None	00

Fig. 42: GPIO Command - CAN I/O

The Id must match the one configured in the 1x PDI Builder as Output filter:

CANL		
Port	вотн	-
ld CONTR	40	
Mask	2047	dec
Filter type	Standard	•

Fig. 43: GPIO Command - Input filter configuration

3.3.3.6 Reading/Sending RPMs

This section presents the steps to follow to read RPMs.

3.3.3.6.1 CEX PDI Builder side

1. Go to Input/Output menu \rightarrow **GPIO panel**.

RPM can be read on the available digital inputs I/O 1-4. The chosen pin needs to be configured as "GPIO as input". In the example shown here, I/O1 is chosen.

GPIO	Signal	GPIOId	10	Pull-up	Function	(
PWM	I/O1	GPIO 24	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
I/O Setup	1/02	GPIO 25	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
CAN I/O	I/O3	GPIO 26	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
Digital Input	I/O4	GPIO 27	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
SCI	PWM 1	GPIO 0	GPIO as input	Pull-up disabled	Mux 1	Sync
CAN Setup	PWM 2	GPIO 1	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 3	GPIO 2	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 4	GPIO 3	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 5	GPIO 4	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 6	GPIO 5	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 7	GPIO 6	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync
	PWM 8	GPIO 7	GPIO as input	Pull-up disabled	Mux 0 / GPIO	Sync

Fig. 44:	Reading	RPMs -	GPIO	configuration
1 18. 1 11	iteaung.		0110	comparation

2. Go to Input/Output menu \rightarrow **Digital Input panel**.

There are 4 possible producers: CAP 1-4. One needs to be chosen and linked to one of the RPMs consumers (RPMs 1-4). For more information on Digital input configuration, see section *Digital Input* of this manual.

Then, select an edge detection option: "First rising edge" or "First falling edge".

	نو	🗘 Input / Out	put		4	. .	-
GPIO PWM	Configuratio						
		Producer		Consumer	(
I/O Setup	\$ \$	CAP 1	\rightarrow	RPM 1	Q0		
CAN I/O	00	CAP 2	\rightarrow	None	Q0		
	¢\$	CAP 3	\rightarrow	None	Q0		
SCI CAN Setup	Q ₀	CAP 4	\rightarrow	None	Q0		

Fig. 45: Reading RPMs - Digital Input

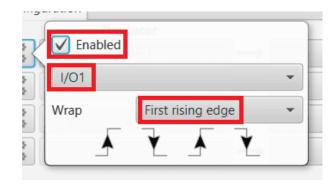


Fig. 46: Reading RPMs - Digital Input configuration

3. Go to Sensors menu \rightarrow RPM panel \rightarrow RPM 1 tab (as RPM 1 has been selected in the Digital Input panel).

Here the expected pulse needs to be defined. For more information on RPM configuration, see section *RPM* of this manual.

CEX2 PDI Builder			~~ X
	ີລ Sensors		2 2 5 0
RPM Lidar Co Image: Strate st	Sensors RPM 1 RPM 2 RPM 3 RPM Units Average Minimum Maximum	4 32 Pulses per cycle 5 1.0E-4 s 0.5 s	

Fig. 47: Reading RPMs - RPM configuration

4. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow Configuration tab.

Connect a **CAN Custom Producer Telemetry** to a **CAN Output Filter** as follows (in this example, the **RPM1** variable is sent via CAN B bus of the CEX):

PIO	Configurat	tion	CAN Telemetry 1 CAN Telemetry	2 CAN	Telemetry 3			
WM	Priority		Producer		Consumer			
/O Setup	High High	Q6	Application Processor	\rightarrow	None	00		Â
AN I/O	High 🗌	Q_0^0	CAN Input Filter 1	\rightarrow	Application Processor	Q ₀		
igital Input Cl	High	Q _0	CAN Input Filter 2]→[CAN to Serial 1	Q0		
AN Setup	High High	Q _0	CAN Input Filter 3	\rightarrow	None	Q0		
	High 🗌	Q _0	CAN Input Filter 4	\rightarrow	None	O ₀		
	High 🗌	Q _0	CAN Input Filter 5	\rightarrow	None	Q0		
	High 🗌	Q _0	CAN Input Filter 6	\rightarrow	None	Q0		
	High	Q0	CAN unwrapper 1	\rightarrow	None	Q ₀ ⁰		
	High	Q0	CAN unwrapper 2	\rightarrow	None	Q ₀ ⁰		
	High	Q_0^0	Serial to CAN 1		CAN Output Filter 2	Q_0^0		
	High 🗌	00	Serial to CAN 2	\rightarrow	None	Q ₀		
	High 🗌	Q _0	Serial to CAN 3	\rightarrow	None	Q0		
	High	00	CAN Custom Producer 1 Tele	→	CAN Output Filter 1	¢¢ <	Port	CAN B
	High	Q0	CAN Custom Producer 2 Tele	\rightarrow	None	00	·	
	High 🗌	Q ⁰	CAN Custom Producer 3 Tele	\rightarrow	None	Q ₀ ⁰		

Fig. 48: Reading RPMs - CAN Custom Telemetry

5. Go to Input/Output menu \rightarrow CAN I/O panel \rightarrow CAN Telemetry tab 1 (as the Producer is CAN Custom Producer 1 Telemetry).

A new telemetry message needs to be created with its correspondent ID, endianness and period.

- Can ID: 1200.
- Endianness: Little.
- Period: 0.01 s.

In the telemetry message one of CEX's variables needs to be selected. As **RPM 1** has been chosen as consumer in the **Digital Input**, the variable to be sent is **RPM1**.

CEX2 PDI Builder	ৰ্ণ্য Input / Output	- × • • • • •
	Configuration CAN Telemetry 1 CAN Telemetry 2 CAN Telemetry 3 TX Ini TX EXT Can ID: 1200 Little endian Period 0.01 able ASCI Position Occupancy	s] + - ×
		+

Fig. 49: Reading RPMs - CAN Custom Telemetry configuration

More information on the configuration of Telemetry messages can be found in the CAN Telemetry section.

3.3.3.6.2 1x PDI Builder side

6. Go to UI menu \rightarrow Variables panel \rightarrow **Real Vars tab**.

Rename a Real User Variable (32 bits) that will be used to store the value received from CEX:

eronte PDI Builder eronte v4.8 💌		₆ UI			- • • • • •
Operation variables	Bits	Jnsigned Real Vars Featur	es		
Geoid	Searc	h			
	Id	Default name	Custom name	Show unit	Init value (SI)
	3228	User Variable 129 (Real		Custom Type ()	
3	3229	User Variable 130 (Real	CEX RPM 1	Custom Type ()	
3	3230	User Variable 131 (Real		Custom Type ()	
*	3231	User Variable 132 (Real		Custom Type ()	
	3232	User Variable 133 (Real		Custom Type ()	
	3233	User Variable 134 (Real		Custom Type ()	
	3234	User Variable 135 (Real		Custom Type ()	
3	3235	User Variable 136 (Real		Custom Type ()	
	3236	User Variable 137 (Real		Custom Type ()	
	3237	User Variable 138 (Real		Custom Type ()	
	3238	User Variable 139 (Real		Custom Type ()	
	3239	User Variable 140 (Real		Custom Type ()	
	3240	User Variable 141 (Real		Custom Type ()	
	3241	User Variable 142 (Real		Custom Type ()	

Fig. 50: 1x PDI Builder - User Variable renamed

7. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Mailboxes tab.

Configure some mailboxes to receive the message with ID 1200:

Veronte PDI Builder Veronte v4.8	-
I/O Setup CAN Setup Digital Input	Configuration Custom message 1 Custom message 2 Custom message 3 Mailboxes CAN A CAN B
	Units ID DEC Units Mask BIN Baudrate 1000000.0 Mailboxes reserved RX: 8 Mailboxes available TX: 24
	# Mailboxes Extended ID Mask
	1 4 1301 1111111111 2 4 1200 1111111111
(
8	
8 9 8 9	

Fig. 51: 1x PDI Builder - Mailbox configuration

8. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Configuration tab.

Connect an **Input filter** to a **Custom message consumer**. Both CAN buses of the autopilot 1x can be used, as well as normal IDs and extended IDs:

-		•🕁 Input / O	utput				± (
) Setup	Configuration	Custom message 1	Custom message	e 2 Cu	istom message 3	Mailboxes	
Setup	Priority	Producer		(Consumer		
al Input	High 🕸	Serial to C	AN 1	→	Output filter	1	Q_0^0
	High 🔯	Serial to C	AN 2	→	Output filter	2	\mathbf{Q}_{0}^{0}
	High 🔯	Serial to C	AN 3	→	None		Q^0_0
	High 🗘	Serial to C	AN 4	•	None		Q^0_0
	High 🗘	Serial to C	AN 5	→	None		Q^0_0
	High 🗘	Serial to C	AN 6	•	None		Q^0_0
	High 🔯	CAN custom n	nessage 1	→	None		Q^0_0
	High 🔯	CAN custom n	nessage 2	+	Output filter	3	$\dot{\mathbf{Q}}^0_0$
	High 🔯	CAN custom n	nessage 3	→	None		00
	High 🛱	Input filt	er 1	→	CAN to seria	11	Q^0_0
	High 🛱	Input filt	er 2	→	None		\mathbf{Q}_0^0
	High High	Input filt	er 3	→	Custom messa	ge 1	Q^0_0
	High 🛱	Input filt	er 4	→	None		Q^0_0
	High 🕸	Input filt	er 5	•	None		Q^0_0
	High 🕸	Input filt	er б	•	None		Q^0_0
	High 🔯	CAN unwra	pper 1	→	None		00
	High 68	CAN UNWER		-	None		20

Fig. 52: 1x PDI Builder - Input filter

Port	BOTH	
ld	0	
Mask	0	de
Filter type	Both	-

Fig. 53: 1x PDI Builder - Input filter configuration

9. Go to Input/Output menu \rightarrow CAN Setup panel \rightarrow Custom message 1 tab (as Custom Message 1 has been selected as consumer).

Configure the reading of the message and store the received value in the user variable renamed above:

Veronte Veronte	PDI Builder v4.8 👻	•ᡩ Input / Output	X
	I/O Setup CAN Setup Digital Input	Configuration Custom message 1 Custom message 2 Custom message 3 Mailboxes TX Ini TX RX EXT Can ID: 1200 Little endian Time out 1.0 s Bit ID: 0	+
		00 kip Variable ASCII Position Occupancy 0 1 0 0	- ×
Ⅲ ⊙			
89 X V			

Fig. 54: 1x PDI Builder - Custom Message configuration

3.3.3.7 Serial communication

The user can send information through the serial ports of the CEX. For this purpose, please note the following explanation:

3.3.3.7.1 CEX PDI Builder side

1. I/O Setup configuration

- Reception of serial information on UART-A (RS232-A) (producer) is stored in Serial to CAN 2 (consumer).
- Transmission of serial information is sent using the CAN to Serial 2 (producer) via UART-A (consumer).

('Serial to CAN' and 'CAN to Serial' are explained in the I/O section of this manual, click here to access it)

	¢∕ _{v∎} +	Input / Output			7 0
GPIO	Configuration				
PWM	Priority	Producer		Consumer	
I/O Setup	High 🔇	UART-A / RS232-A	\rightarrow	Serial to CAN 2	00
CAN I/O	High 🕸	UART-B / RS232-B	\rightarrow	None	00
Digital Input SCI	High 🕸	UART-C / RS485	\rightarrow	None	Q0
CAN Setup	High 🔯	CAN to Serial 1	\rightarrow	Commgr port 1	Q0
	High 🔯	CAN to Serial 2	→	UART-A / RS232-A	00
	High 🕸	CAN to Serial 3	\rightarrow	None	Q0
	High 🕸	Commgr port 1	\rightarrow	Serial to CAN 1	Q0
	High 🕸	Commgr port 2	\rightarrow	None	Q0
	High 🕸	Commgrf port 3	\rightarrow	None	Q0
	High 🕸	Commgr port 4	\rightarrow	None	00
	High 🔯	Commgr port 5	\rightarrow	None	00
	High 🕼	Commgr port 6	\rightarrow	None	Q0
	High 🔯	Tunnel 1	\rightarrow	None	00
	High 🔯	Tunnel 2	\rightarrow	None	Q0
	High 🔯	Tunnel 3	\rightarrow	None	Q0
	High 🗱	CAN wrapper 0 for serial tran	\rightarrow	None	00

Fig. 55: Serial - I/O Setup

2. CAN I/O configuration

- The information that will be **sent via serial port UART-A** is going to be received on the CEX through its **CAN B** port. A **mask Id of 51** is added to the **Input filter**. The incoming information (from Veronte Autopilot 1x) is processed in '**CAN to Serial 2**'.
- The information **coming from port UART-A** and processed as '**Serial to CAN 2**' is going to be linked to an **Output filter**. The information of '**Serial to CAN 2**' is going to be sent via **CAN B** with a **mask ID of 50** (to be read by Veronte Autopilot 1x).

SPIO	Configuratio	on CA	N Telemetry 1 CAN Telemetry	2 CAN	Telemetry 3			
WM	Priority	-	Producer		Consumer			
/O Setup	High 🗘		Application Processor	\rightarrow	None	0°		â
AN I/O	🗌 High 🗘	X;	CAN Input Filter 1)→	Application Processor	Q ^o		
vigital Input Cl	🗌 High 🚺	X°	CAN Input Filter 2	\rightarrow	CAN to Serial 1	Q0		
AN Setup	🗌 High 🗯	Xo	CAN Input Filter 3]→[CAN to Serial 2	Q0		
	🗌 High 🚺	X	CAN Input Filter 4	\rightarrow	None	Q0		
	🗌 High 🚺	Xo	CAN Input Filter 5	\rightarrow	None	Q ₀ ^o		
	🗌 High 🚺	Xo	CAN Input Filter б	\rightarrow	None	Q ₀ ^o		
	High 0	X ^o	CAN unwrapper 1	\rightarrow	None	Q ₀		
	High 0	X ^o	CAN unwrapper 2	\rightarrow	None	Q ₀		
	🗌 High 🚺	X0	Serial to CAN 1	\rightarrow	CAN Output Filter 2	Q ₀ ⁰		
	🗌 High 🔯	X	Serial to CAN 2	→	CAN Output Filter 3	Q° (Port	CAN B
	🗌 High 🕻	X	Serial to CAN 3	\rightarrow	None	00		
	High 0	X ^o	CAN Custom Producer 1 Tele]→[CAN Output Filter 1	Q _0		
	High 🗘	X ^D _D	CAN Custom Producer 2 Tele	\rightarrow	None	Q ₀ ^o		
	High 0	2	CAN Custom Producer 3 Tele	\rightarrow	None	00		

Fig. 56: Serial - CAN I/O

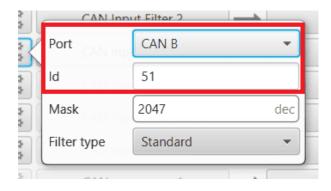


Fig. 57: Serial - Input filter configuration



Fig. 58: Serial - Serial to CAN configuration

3. CAN Setup (mailboxes) configuration

	•	🕁 Input	: / Outpu	ıt		2050
GPIO DWM	CANA CAN	NB				
I/O Setup	Units ID	DEC 🔹	Units M	ask BIN	▼ Baudrate 1000000.0	
CAN I/O	Mailboxes re	eserved RX:	8 Mailbox	es available T	X: 24	
Digital Input	#	Mailboxes	Extended	ID	Mask	
	1 ****** 4	4		1302	1111111111	
CAN Setup	2 2 200000 2	4		51	1111111111	
	< () :	
	CAN I/O	PWM Units ID I I/O Setup Mailboxes m CAN I/O Mailboxes m Digital Input # SCI 1 CAN Setup 2 ON Setup 1 I I I I I I I I I I I I I I I I I I I I I I I II I II I III I IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	PWM I/O Setup CAN I/O Digital Input SCI CAN Setup 2 4 2 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 1 1 1 1 1 1 1 2 1 1 2 1 2 1 2 1 2 1 2 1 2 1 1 1 1 2 1 2 1 2 1 2 1 2 1 2	PWM I/O Setup CAN I/O Digital Input SCI CAN Setup 2 2 3 4 1 2 3 4 1 2 3 3 4 1 4 1 4 1 4 1 4 1 <td>PWM I/O Setup CAN I/O Digital Input SCI CAN Setup CAN Setup 2 4 1302 2 4 51 2 4 51</td> <td>PWM I/O Setup CAN I/O Digital Input SCI CAN Setup CAN Setup CAN Setup I = 4 I = 1302 I = 4 I = 4 I = 1302 I =</td>	PWM I/O Setup CAN I/O Digital Input SCI CAN Setup CAN Setup 2 4 1302 2 4 51 2 4 51	PWM I/O Setup CAN I/O Digital Input SCI CAN Setup CAN Setup CAN Setup I = 4 I = 1302 I = 4 I = 4 I = 1302 I =

Mailboxes need to be defined for the reception of CAN messages. In the example above, mailboxes for ID 51 need to be added on CAN B port.

Fig. 59: Serial - CAN Setup configuration

3.3.3.7.2 1x PDI Builder side

4. I/O Setup configuration

On the **I/O Setup**, link an '**RS Custom Message**' to a '**Serial to CAN**' with the serial data that the autopilot is going to send to CEX.

Then, link a 'CAN to Serial' to another 'RS Custom Message' with the expected serial messages that the CEX will receive in the selected serial port.

I/O Setup CAN Setup	Configura						
	Priority		Producer No custom message 2		Consumer	70	
Digital Input	📃 High	\mathbf{Q}_0^0	RS custom message 3	\rightarrow	Serial to CAN 2	Q0	
	High	00	Tunnel 1	\rightarrow	None	Q ⁰	
	High	Q^0_0	Tunnel 2	\rightarrow	None	¢₀	
	High	Q^0_0	Tunnel 3	\rightarrow	None	Q0	
	High	Q^0_0	GPS 1 RTCM	\rightarrow	None	¢₀	
	High	Q^0_0	GPS 2 RTCM	\rightarrow	None	Ф ⁰ ₀	
	High	Q^0_0	External HMR2300 magneto	\rightarrow	None	Φ.	
	High	Q^0_0	Y1 splitter A	\rightarrow	None	Q0	
	High	Q^0_0	Y1 splitter B	\rightarrow	None	Q ^o	
	High	Q^0_0	Y2 splitter A	\rightarrow	None	Q ⁰ ₀	
	High	Q^0_0	Y2 splitter B	\rightarrow	None	Q ⁰	
	High	Q^0_0	Y3 splitter A	\rightarrow	None	Q ⁰ ₀	
	High	Q^0_0	Y3 splitter B	\rightarrow	None	Q ⁰	
	High	Q^0_0	Iridium	\rightarrow	None	Q0	
	High	Q^0_0	Unescape port	\rightarrow	None	Q0	

Fig. 60: Serial - 1x I/O Setup configuration

5. CAN Setup configuration

As for the **CAN I/O**, the same IDs employed in CEX for the Input and Output filters are going to be employed on Veronte Autopilot 1x side, but they need to be inverted.

Therefore, the **Input filter** linked to the chosen 'CAN to Serial' needs to have **ID 50**. And the **Output filter** linked to the chosen 'Serial to CAN' will have **ID 51**.

r		• 🕁 Input / O	utput		*
etup	Configuratio		Custom message 2	Custom message 3	Mailboxes
Setup	Priority	Producer		Consumer	
Input	🗌 High 🗯		CAN 1 -	Output filter	
	🗌 High 🗘	Serial to C	CAN 2	Output filter a	2 Ø ₀ ⁰
	High 🕻	Serial to C	CAN 3	None	¢\$
	🗌 High 🚺	Serial to C	CAN 4	None	00
	High 🚺	Serial to C	CAN 5	None	Q ₀ ^o
	🗌 High 🚺	😮 Serial to C	CAN 6	None	Q0
	High	CAN custom r	message 1 🔶	None	Q ₀
	High 🚺	CAN custom r	message 2 🔶	None	Q0
	High 🗘	CAN custom r	message 3 🔶	None	Q0
	🗌 High 🗘	lnput filt	ter 1	CAN to serial	1 00
	🗌 High 🚺	C Input filt	ter 2 🔶	None	Q ₀ ^o
	High	C Input filt	ter 3 🔶	None	00
	🗌 High 🚺	C Input filt	ter 4	None	Q ₀
	🗌 High 🚺	C Input filt	ter 5	None	Q0
	🗌 High 🚺	C Input filt	ter 6	None	Q0
	High 🗘	CAN unwra	apper 1 🔶	None	Q0
	High 1	CAN UDWIT	nner?	None	50

Fig. 61: Serial - 1x CAN Seup



Fig. 62: Serial - 1x Serial to CAN configuration

Port	BOTH	-
ld	50	
Mask	2047	de
Filter type	Standard	

Fig. 63: Serial - 1x Input filter configuration

6. Mailboxes configuration

Some mailboxes with ID 50 will have to be created on whichever chosen reception CAN bus.

Veronte PDI Builder Veronte v4.8 👻	- ×
I/O Setup CAN Setup Digital Input	Configuration Custom message 1 Custom message 2 Custom message 3 Mailboxes CAN A CAN B
I/O Setup CAN Setup Digital Input Image: Setup Image: Setup <td< th=""><td>Units ID DEC Units Mask BIN Baudrate 1000000.0 Mailboxes reserved RX: 8 Mailboxes available TX: 24</td></td<>	Units ID DEC Units Mask BIN Baudrate 1000000.0 Mailboxes reserved RX: 8 Mailboxes available TX: 24
	# Mailboxes Extended ID Mask
臣	1 — 4 1301 111111111
¢ °	2 — 4 50 1111111111
8	
B	
© 	

Fig. 64: Serial - 1x Mailboxes configuration

CHAPTER

FOUR

TROUBLESHOOTING

4.1 Migrate configuration

Warning: When performing automatic migration from a previous version to the current version of the software, errors may occur.

It is then the responsibility of the user to check the subsequent result.

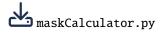
CHAPTER

FIVE

FAQ

5.1 How to calculate a mask

This section attaches a python program that allows users to easily calculate their mask in standard or extended frame format by simply entering the CAN Ids as a **vector**. In addition, this program also converts each Id entered into binary.



An example of the execution of this program is shown below:



Fig. 1: Example of maskCalculator program