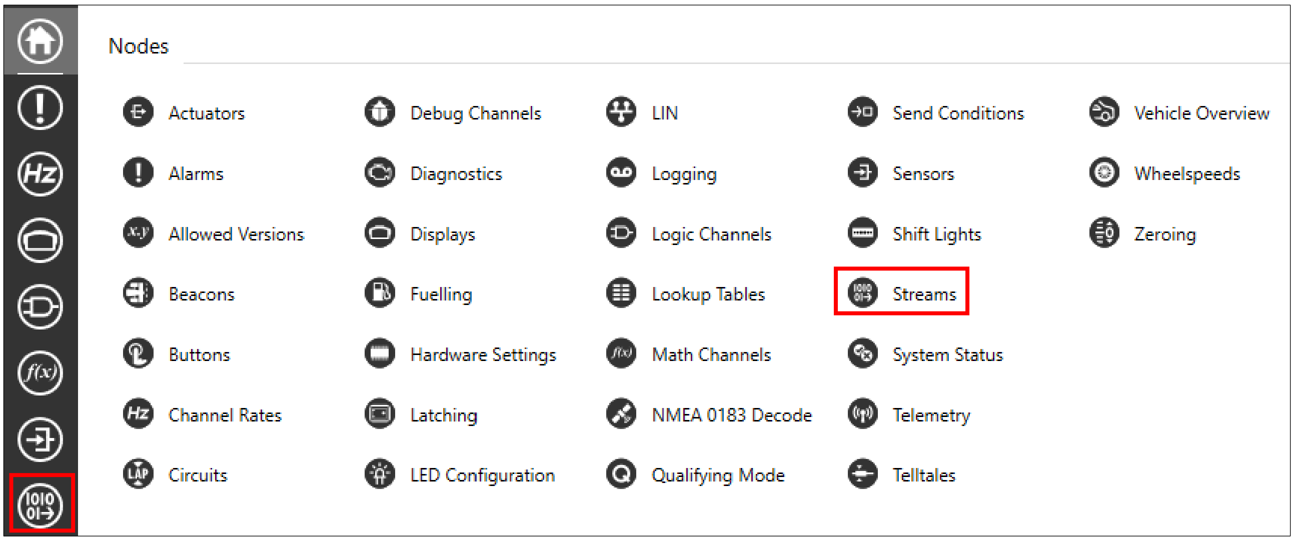
CAN streams overview

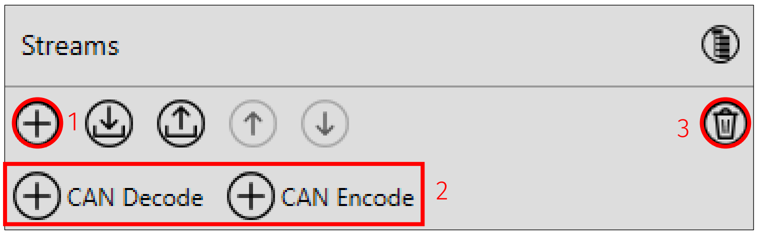
Control Area Network (CAN) is a widely used communication protocol for system integration. Most Cosworth devices support CAN communication and interfacing with both Cosworth and third-party CAN devices. The **Streams** node is used to configure CAN streams.

Create a CAN stream

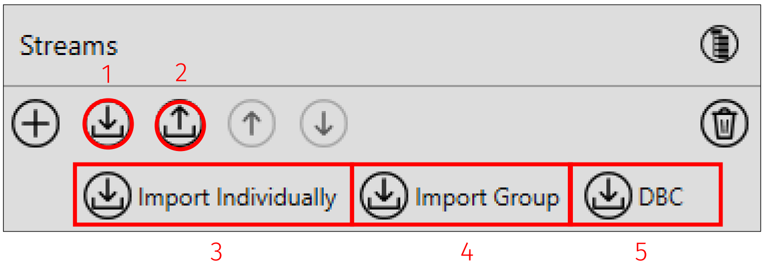
Within a setup, click on the 'Streams' option on the left side or use the **Streams** node to access all the CAN streams. The hierarchy of a CAN steam is as follows: Stream, Packets, Bytes, and Bits. Channels can be multiple Bytes or Bits.



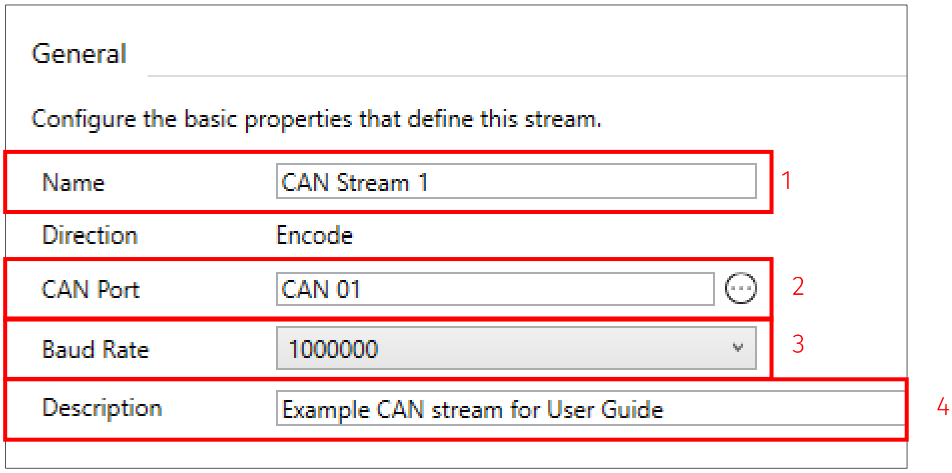
To create a new stream, hover over the **+** button (1) and select either CAN Decode or CAN Encode (2). Use the 'bin' tool to delete unwanted streams (3).



You can import (1) and export (2) streams either individually (3) or in multiples within a group (4) between existing Toolset setups in Toolset Library File (*.tlf*) format and import and export in the standard CAN DBC (*.dbc*) format (5).



When you add a new stream, the stream appears in the streams list. Configure the stream at the right in the main window. In the General section you can configure the stream name (1), CAN Port allocation (2), Baud Rate (3) and a description (4).



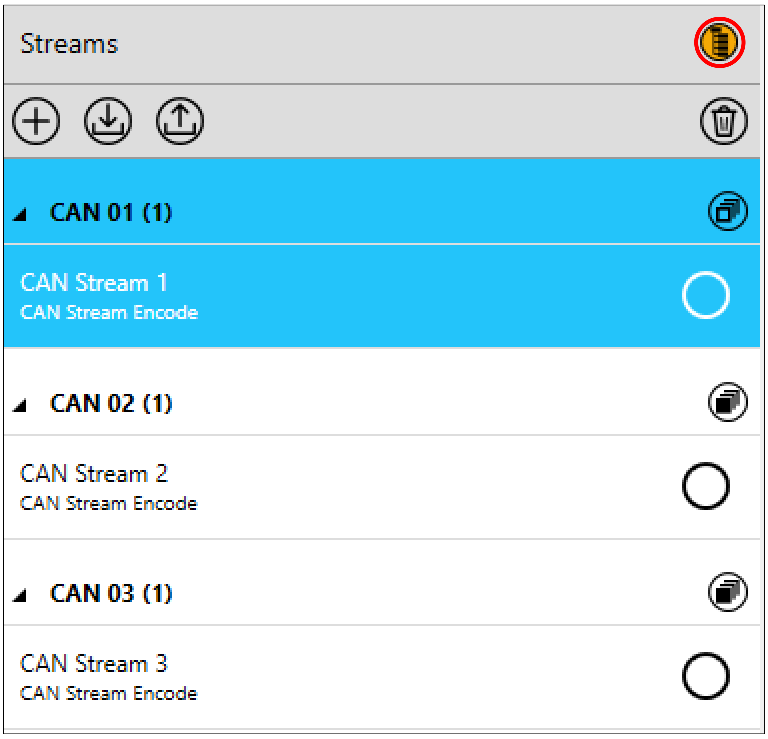
General

Configure the basic properties that define this stream.

Name	CAN Stream 1
Direction	Encode
CAN Port	CAN 01
Baud Rate	1000000
Description	Example CAN stream for User Guide

Note: You can configure the **CAN Port** name on the Hardware Settings node.

If you configure multiple streams, you can select the 'display view' option to display the **Streams** list in groups linked to the CAN port allocation for each stream.



Streams

◢ CAN 01 (1)

CAN Stream 1
CAN Stream Encode

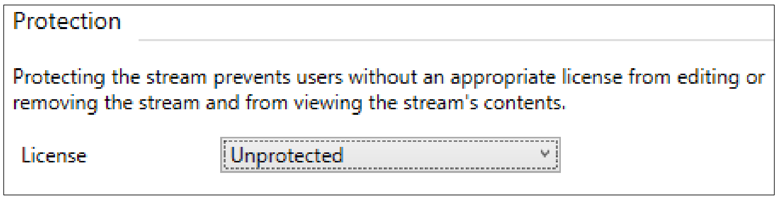
◢ CAN 02 (1)

CAN Stream 2
CAN Stream Encode

◢ CAN 03 (1)

CAN Stream 3
CAN Stream Encode

In the **Protection** section you can add permissions to the stream via a dongle to allow you to edit or view the stream content. This feature is typically used by manufacturers to lock their setups, streams, and so on.

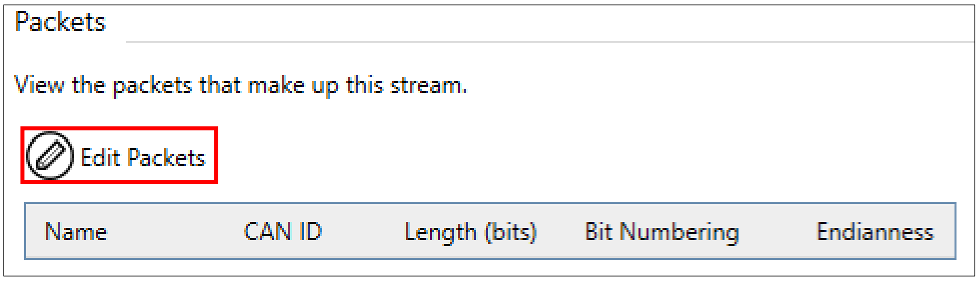


Protection

Protecting the stream prevents users without an appropriate license from editing or removing the stream and from viewing the stream's contents.

License: Unprotected

The **Packets** section hosts each packet in the stream. Click **Edit Packets** to begin to create a packet.

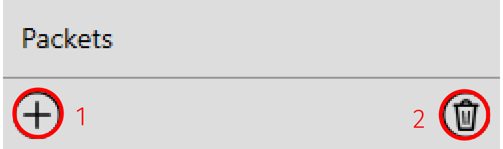


Packets

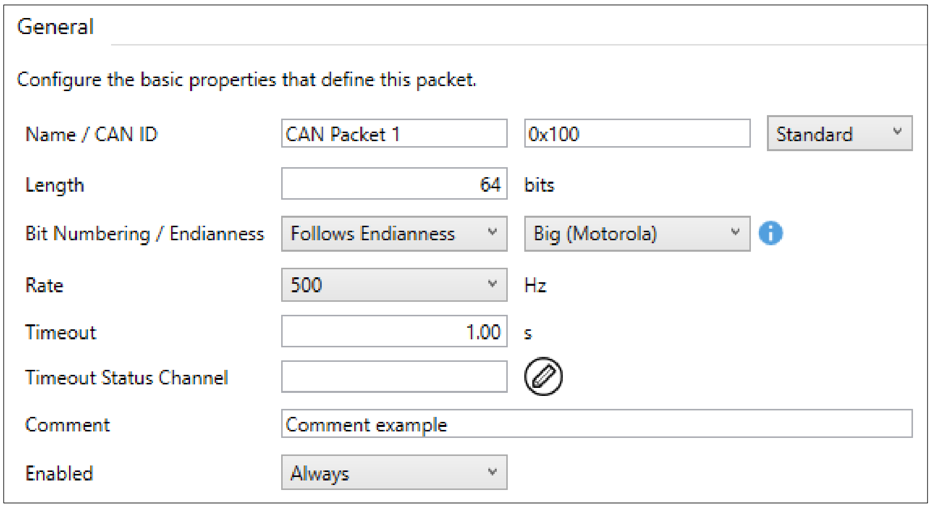
View the packets that make up this stream.

Name	CAN ID	Length (bits)	Bit Numbering	Endianness

Click the **+** button at the top left to create a new packet. This populates the list below with a new packet. Use the ‘bin’ icon to remove a selected packet Click the + button (1) to add a new packet. Use the ‘bin’ tool to delete an unwanted packet (2).



Once a packet is added, you can complete configuration from the main window. In the **General** section you can edit the **Name/CAN ID**, **Length**, **Bit Numbering/Endianness**, **Rate**, **Timeout**, **Timeout Status Channel**, and **Comment**. The **CAN ID** needs to be unique on for that specific CAN Port. The **Length** has a maximum of 64 bits. You can change **Endianness** according to preference, but it needs to correspond to the opposing Encode/Decode. For more information about Endianness/Bit Numbering methodologies click the **i** option. The **Rate** selected is the limit for all channels in that packet. The **Timeout** option is used to set an interval after which the Encode/Decode triggers the Timeout Behaviour (explained later in this section). You can use the **Timeout Status Channel** to generate a channel that records the status of a timeout.



General

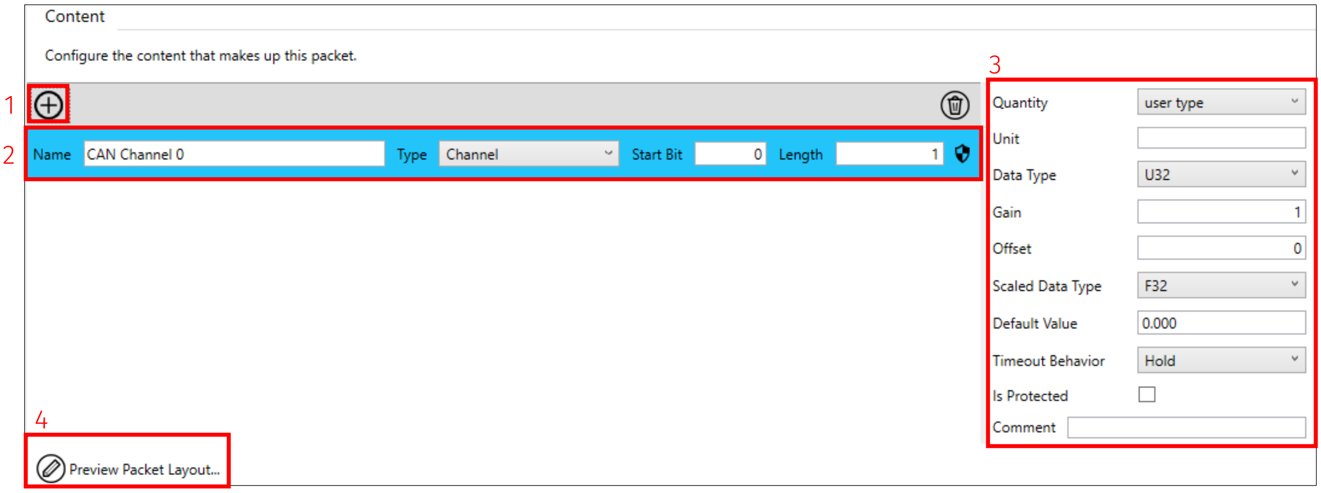
Configure the basic properties that define this packet.

Name / CAN ID	CAN Packet 1	0x100	Standard
Length	64	bits	
Bit Numbering / Endianness	Follows Endianness	Big (Motorola)	
Rate	500	Hz	
Timeout	1.00	s	
Timeout Status Channel			
Comment	Comment example		
Enabled	Always		

To add a channel to the packet, click the **+** icon (1) in the **Channels** section. This adds a channel to the list (2) and populates the channel configurations (3) to the right. The channel list contains the following information: **Name**, **Type**, **Start Bit**, and **Length**.

Note: Channel names are limited to 24 characters, including spaces.

Note: It is best to use a math channel when dealing with complex order of operations for Gain/Offset.



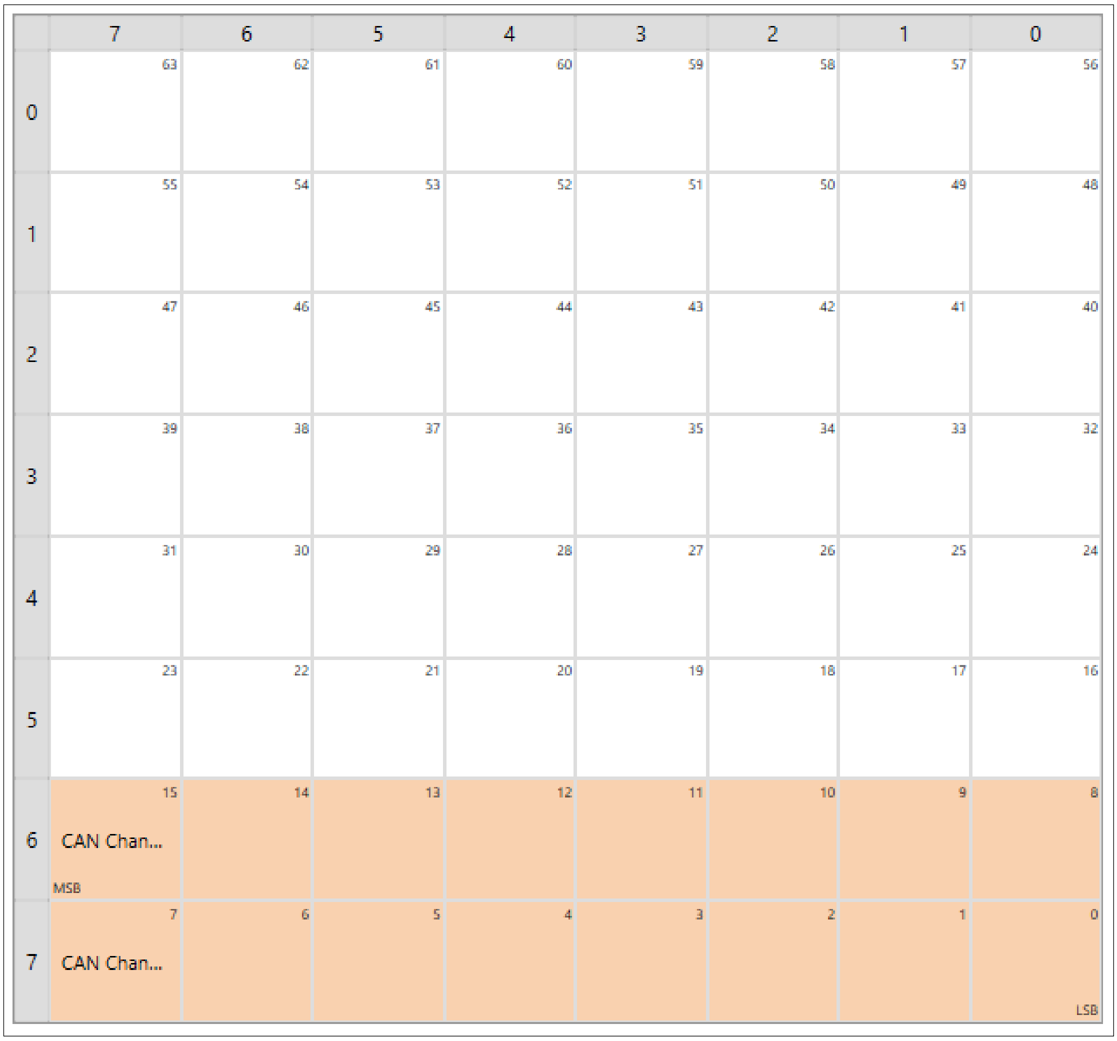
Content

Configure the content that makes up this packet.

Name CAN Channel 0 Type Channel Start Bit 0 Length 1

Quantity	user type
Unit	
Data Type	U32
Gain	1
Offset	0
Scaled Data Type	F32
Default Value	0.000
Timeout Behavior	Hold
Is Protected	□
Comment	

Click **Preview Packet Layout** (4), a visual representation of the packet layout is displayed. This enables an easier visualisation of the bit allocation of items within the packet. LSB means least significant bit and MSB means most significant bit (see below).

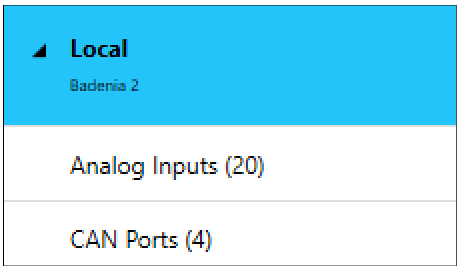


	7	6	5	4	3	2	1	0
0	63	62	61	60	59	58	57	56
1	55	54	53	52	51	50	49	48
2	47	46	45	44	43	42	41	40
3	39	38	37	36	35	34	33	32
4	31	30	29	28	27	26	25	24
5	23	22	21	20	19	18	17	16
6	15 CAN Chan... MSB	14	13	12	11	10	9	8
7	7 CAN Chan...	6	5	4	3	2	1	0 LSB

Hardware limitations

Each CAN Bus has a limit to the number of messages it can send or receive. Click on the Hardware Settings node to see details about CAN bus usage.

On the left side, a list is populated. Select **CAN Ports > Local** to view information about port usage.

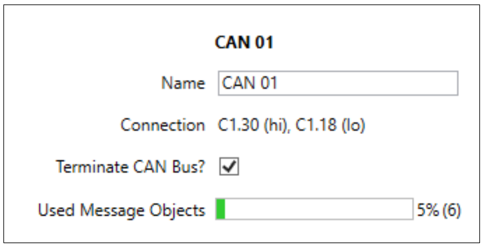


◢ **Local**
Badenia 2

Analog Inputs (20)

CAN Ports (4)

This displays each CAN Port and its usage. It is recommended that you keep the utilization at 80% or below. Oversaturating the CAN bus can lead to CAN errors and potentially missed data.

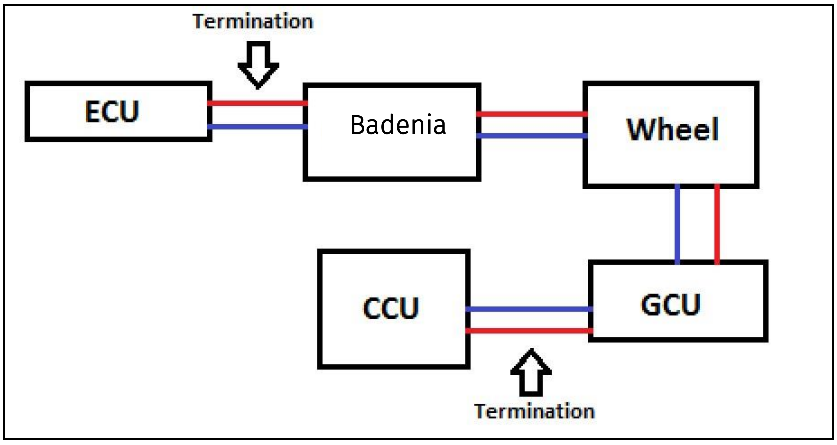


CAN 01

Name	CAN 01
Connection	C1.30 (hi), C1.18 (lo)
Terminate CAN Bus?	☑
Used Message Objects	5% (6)

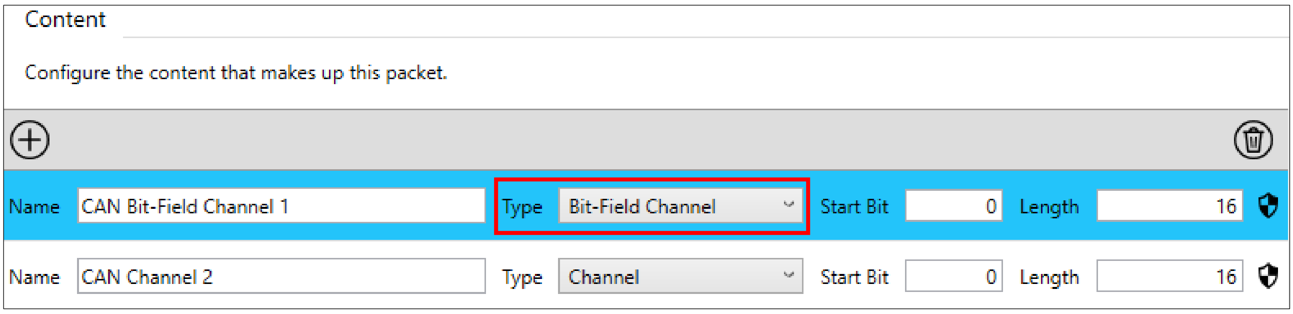
You can use the **Terminate CAN Bus** option to apply a software selectable CAN bus termination. This acts as a physical termination for that bus, if required.

This is a complete CAN Bus setup that has two physical terminations.



Implement a Bit-masked CAN channel

Bit-masking a CAN channel can be useful when information from each bit is beneficial.
When you create a CAN Channel there is an option to add a Bitmask. The Bitmask ANDs with the channel and returns the bit entries that are true. To create a Bit-masked CAN Channel, create a normal channel as described above, but set the **Type** to *Bit-Field Channel*.

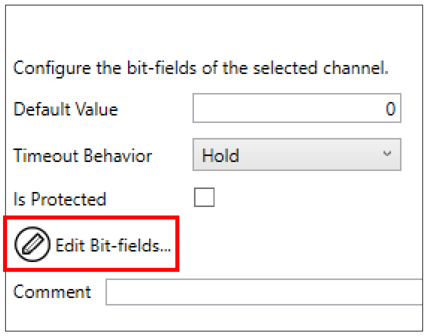


Content

Configure the content that makes up this packet.

Name	CAN Bit-Field Channel 1	Type	Bit-Field Channel	Start Bit	0	Length	16
Name	CAN Channel 2	Type	Channel	Start Bit	0	Length	16

Select **Edit Bit-fields** to edit the Bit-fields.



Configure the bit-fields of the selected channel.

Default Value 0

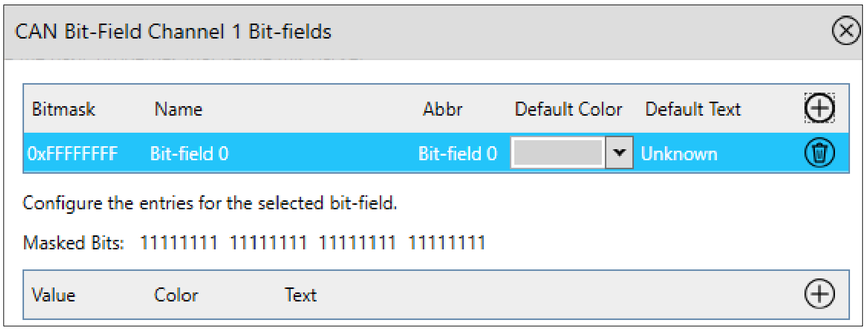
Timeout Behavior Hold

Is Protected □

Edit Bit-fields...

Comment

A window is displayed where you can click the **+** icon to add a Bitmask. The top section relates to the Bitmask and the bottom section relates to bit-specific actions.



CAN Bit-Field Channel 1 Bit-fields

Bitmask	Name	Abbr	Default Color	Default Text
0xFFFFFFFF	Bit-field 0	Bit-field 0	▢	Unknown

Configure the entries for the selected bit-field.

Masked Bits: 11111111 11111111 11111111 11111111

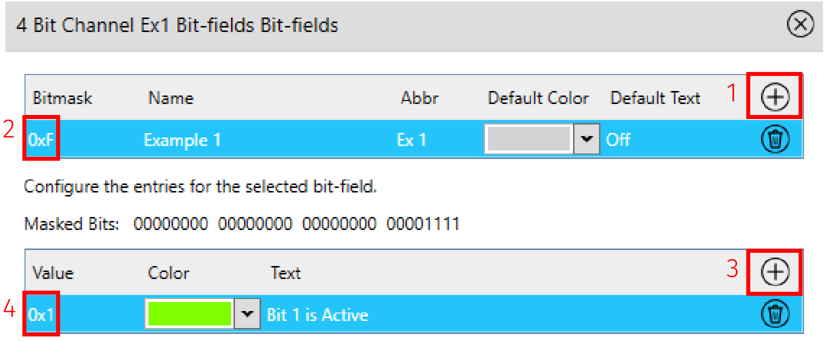
Value	Color	Text

The top section shows the actual **Bitmask**, **Name**, **Abbr**(eviation), **Default Color**, and **Default Text**. The Bitmask is in Hexadecimal. In the bottom section there is each entry for the specified bits. There are two ways to configure Bit-fields and each depend on which view you want to see in live data (see examples below).

Example 1

In this example, a 4 Bit-masked channel is created, consisting of 1 Bitmask and 5 entries. Click the **+** button (1) to create a Bitmask and change it to the desired value (here 0xF (2)). Click the lower + button to add a bit entry (3). The value (4) needs to equal the bit value being requested.

Note: The value needs to be in Hexadecimal. The first entry added is active when bit 1 in the channel is 1.



4 Bit Channel Ex1 Bit-fields Bit-fields

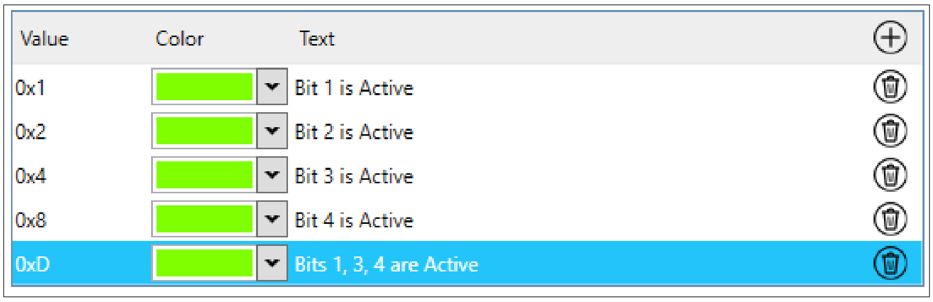
Bitmask	Name	Abbr	Default Color	Default Text
0xF	Example 1	Ex 1	▢	Off

Configure the entries for the selected bit-field.

Masked Bits: 00000000 00000000 00000000 00001111

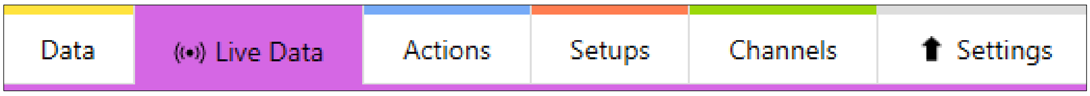
Value	Color	Text
0x1	🟩	Bit 1 is Active

Four more entries are added. Value 0x2 activates when Bit 2 is active. Value 0x4 activates when Bit 3 is Active. Value 0x8 activates when Bit 4 is active, and Value 0xD activates when Bits 1, 3, 4 are Active.

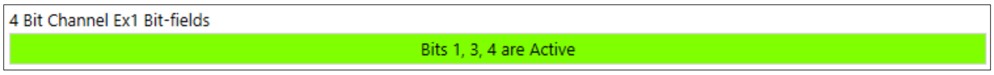


Value	Color	Text
0x1	🟩	Bit 1 is Active
0x2	🟩	Bit 2 is Active
0x4	🟩	Bit 3 is Active
0x8	🟩	Bit 4 is Active
0xD	🟩	Bits 1, 3, 4 are Active

You can view the output of this channel on the **Live Data** tab.



Add the channel 4 Bit Channel Ex1 Bit-fields (from above) to the **Live Data** page. The channel is displayed as seen below. The default value of this channel is set to 13 (0xD). Since there is no data coming across, the value reverts to the default.



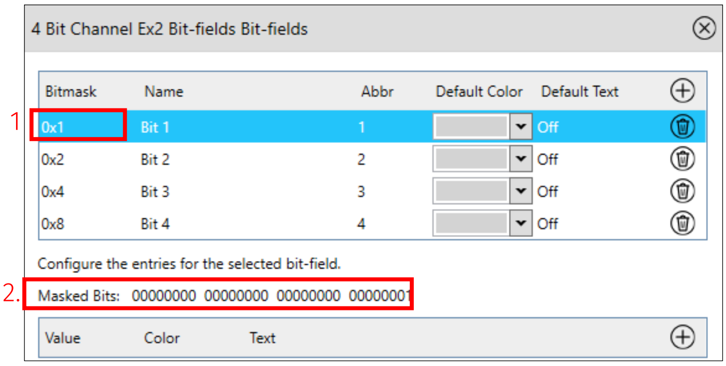
4 Bit Channel Ex1 Bit-fields

Bits 1, 3, 4 are Active

Example 2

In this example, a 4 Bit-masked channel is created. There are 4 Bitmasks and 1 entry per Bitmask. Create 4 Bitmasks, but make each of them specific to each bit. The first mask (1) has a value of 0x1. This creates a mask for bit 1. (2)

Note: No Bitmasks can overlap.



4 Bit Channel Ex2 Bit-fields Bit-fields

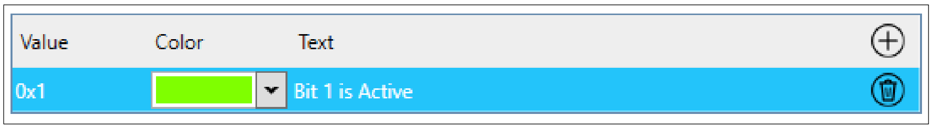
Bitmask	Name	Abbr	Default Color	Default Text
0x1	Bit 1	1	⬜	Off
0x2	Bit 2	2	⬜	Off
0x4	Bit 3	3	⬜	Off
0x8	Bit 4	4	⬜	Off

Configure the entries for the selected bit-field.

Masked Bits: 00000000 00000000 00000000 00000001

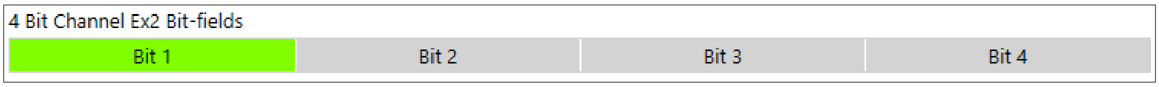
Value	Color	Text

For each mask, you must dd an entry that has the same value as the mask. For the first mask, an entry with the value of 0x1 is added.



Value	Color	Text
0x1	🟩	Bit 1 is Active

Do this for each corresponding Bitmask. Go to the **Live Data** tab to view the channel. The channel is displayed as seen below. This method of Bit-masking is useful when several bits in a channel have independent importance. The default value of this channel is set to 1 (0x1). Since there is no data coming across, the value reverts to the default.


4 Bit Channel Ex2 Bit-fields

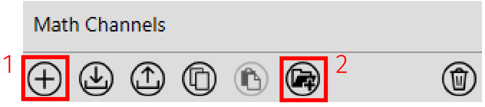
Bit 1	Bit 2	Bit 3	Bit 4

Implement multiplexed CAN channels

You can use multiplexed CAN Channels to compress additional data into a single CAN packet/stream CAN packet.

To create a Multiplexed CAN Encode, two math channels are needed to locate and associate the CAN Channels in the packet. The first math channel is the Indexor. To create a math channel, click on the **Math Channels** node.

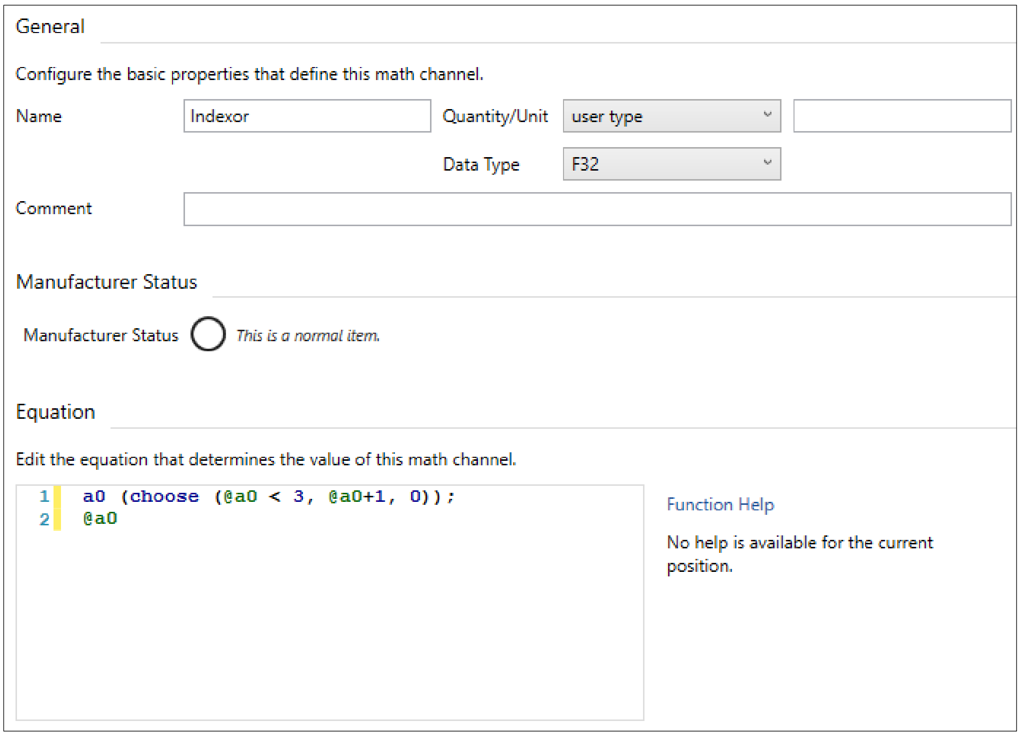
Click the **+** button (1) at the top left to create a new math channel. By default, this is added to the <Default> group of channels. To add channels to a group, highlight the required channels, click ‘add group’ (2), and name the group.



The first math channel is named Indexor. This is a counter and records the number of channels being multiplexed. In this example, three channels are being multiplexed. The equation required is:

```
 a0 (choose (@a0 < 3, @a0+1, 0));
@a0
```

Set the rate to 100Hz and this channel counts to 2 from 0 at 100 increments per second.



General

Configure the basic properties that define this math channel.

Name	Indexor	Quantity/Unit	user type	
		Data Type	F32	
Comment				

Manufacturer Status

Manufacturer Status ○ *This is a normal item.*

Equation

Edit the equation that determines the value of this math channel.

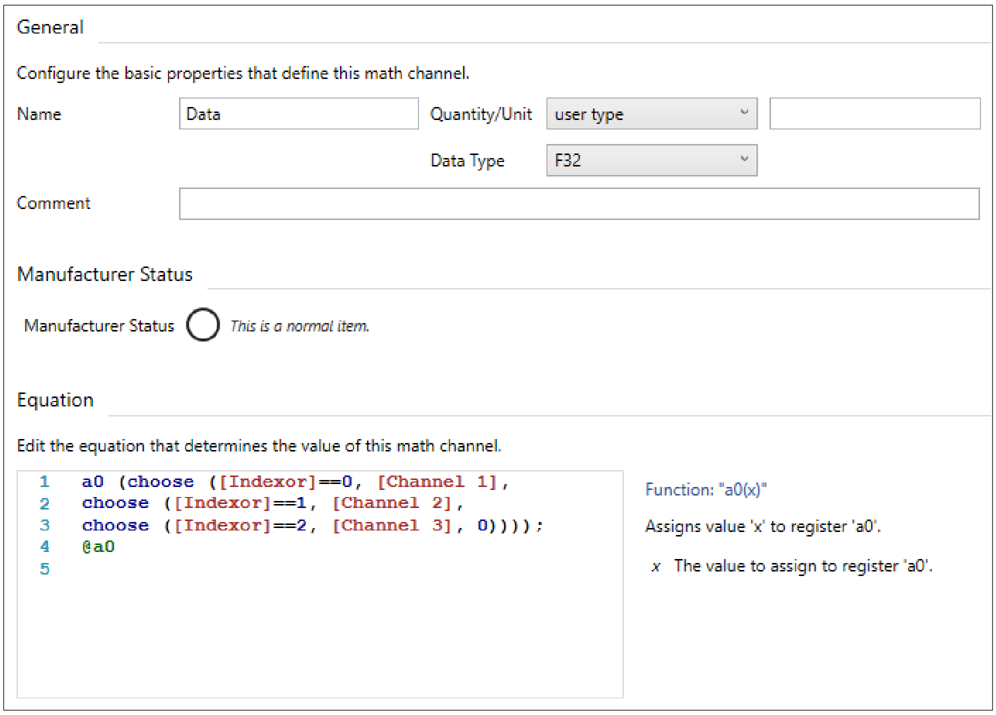
```
a0 (choose (@a0 < 3, @a0+1, 0));
@a0
```

Function Help

No help is available for the current position.

The second channel needed is used to correlate the value from the Indexor to the channel being sent. In the example code below, when the Indexor is equal to 0, the output is Channel 1. When it is 1, the output is Channel 2, and so.

```
a0 (choose ([Indexor]==0, [Channel 1],
choose ([Indexor]==1, [Channel 2],
 choose ([Indexor]==2, [Channel 3], 0)));
@a0
```



General

Configure the basic properties that define this math channel.

Name	Data	Quantity/Unit	user type	
		Data Type	F32	
Comment				

Manufacturer Status

Manufacturer Status ○ *This is a normal item.*

Equation

Edit the equation that determines the value of this math channel.

```
a0 (choose ([Indexor]==0, [Channel 1],
choose ([Indexor]==1, [Channel 2],
choose ([Indexor]==2, [Channel 3], 0)));
@a0

```

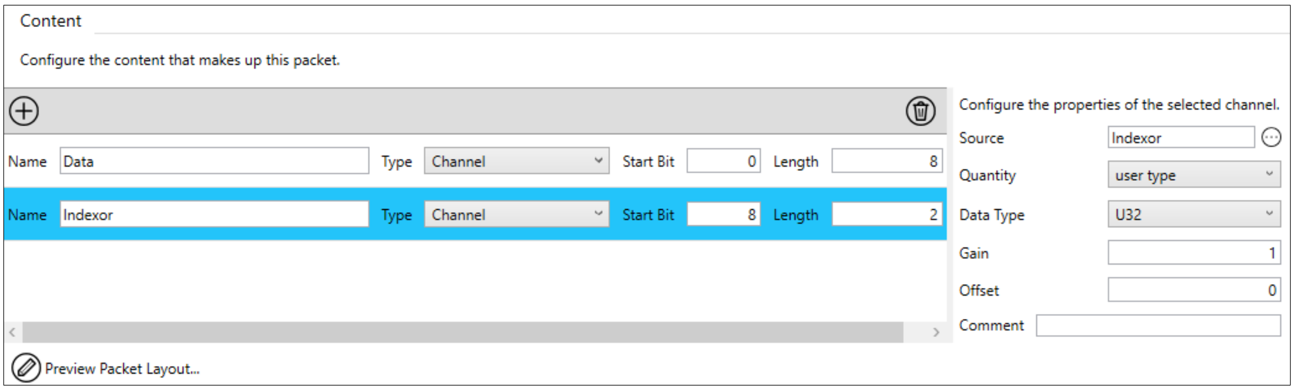
Function: "a0(x)"

Assigns value 'x' to register 'a0'.

x The value to assign to register 'a0'.

Set the rate to 'Inherit rate from Indexor' and this math channel uses the same rate as the channel Indexor. The channels that are requested need to match the rate of the Indexor/Data channels.

Create a new CAN Stream Encode and add two channels to the packet. The source for the channels is the two math channels created above. In this example, data starts at 0 and is 8 bits long. The Indexor starts at 8 and is 2 bits long. The **Length** needs to be long enough to support the number of states. If the number of states is 3, then the length needs to be 2 bits (3 in binary is 11).



Content

Configure the content that makes up this packet.

Name	Type	Start Bit	Length
Data	Channel	0	8
Indexor	Channel	8	2

Configure the properties of the selected channel.

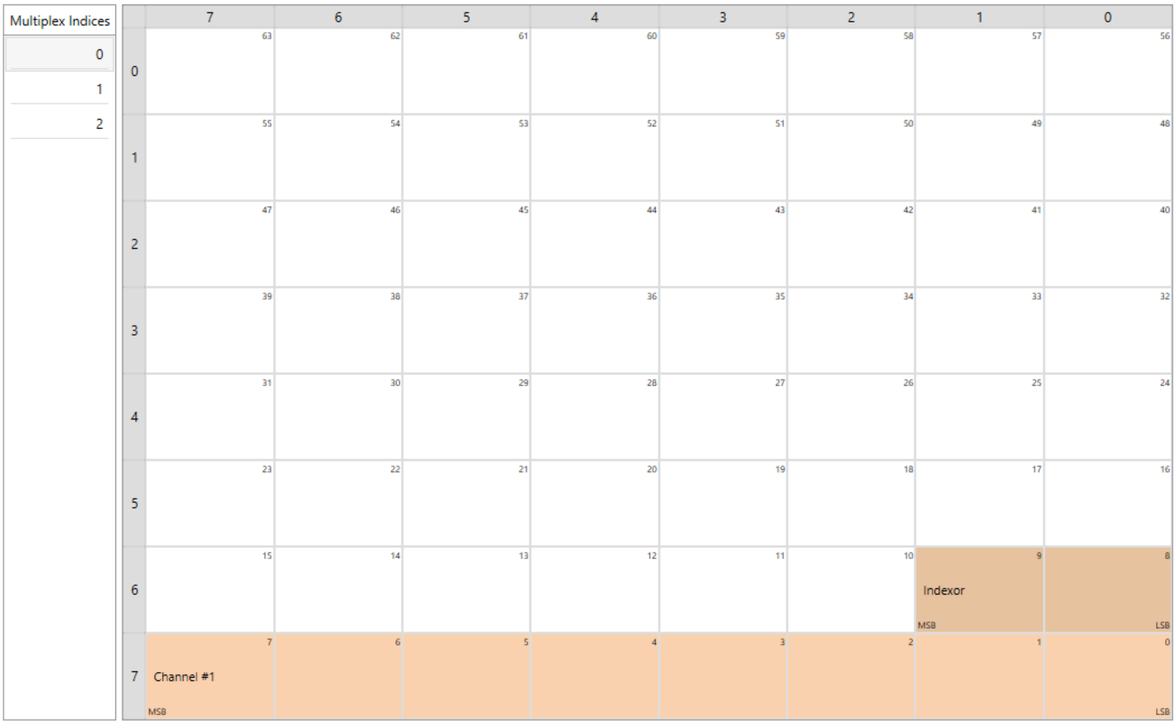
Source	Indexor
Quantity	user type
Data Type	U32
Gain	1
Offset	0
Comment	

When this is done, the channels added to the Data math channel are transmitted across CAN in a multiplexed message.

To create a Multiplexed CAN Decode, create a CAN packet and add an Indexor channel. Under channel type, select 'Indexor'. To the right, enter the **Number of States**. For this example, this 3. Select the **Start Bit** and the **Length**.

Next add the required channels. Create a new channel and set its type as 'Multiplexed Region'. In this example, add Channel #1, Channel #2, and Channel #3. The **Start Bit** and **Length** are the same for all three. Each channel will have its own Mux Index (0-2).

Click on **Preview Packet Layout** in the **Channels** section to view the packet layout. On the left, the Mux Indices are displayed. Click on an index to display each Mux and the layout for that packet.



Multiplex Indices

0

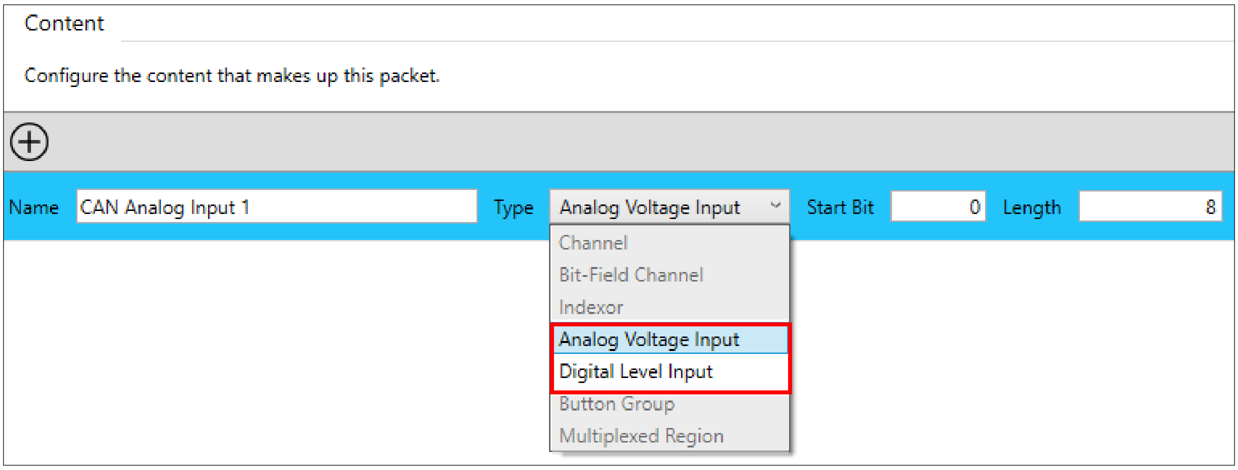
1

2

	7	6	5	4	3	2	1	0
0	63	62	61	60	59	58	57	56
1	55	54	53	52	51	50	49	48
2	47	46	45	44	43	42	41	40
3	39	38	37	36	35	34	33	32
4	31	30	29	28	27	26	25	24
5	23	22	21	20	19	18	17	16
6	15	14	13	12	11	10	9 Indexor MSB	8 LSB
7	7 Channel #1 MSB	6	5	4	3	2	1	0 LSB

Virtual analog and digital inputs

You can use CAN messages as virtual analog and digital sensor inputs. In other words, when the message within a packet is configured as either an 'Analog Voltage Input' or 'Digital Level Input' then the value transmitted over CAN is interpreted as an analog or digital input, respectively.

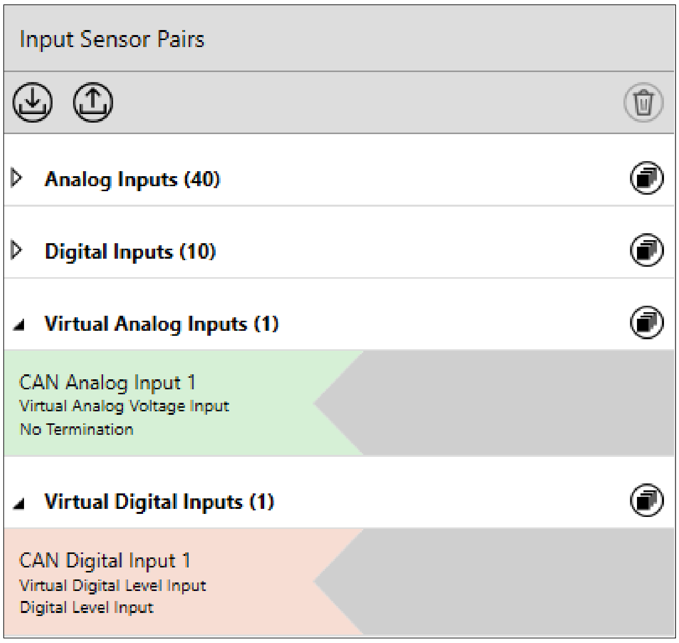


Content

Configure the content that makes up this packet.

Name: CAN Analog Input 1 | Type: Analog Voltage Input | Start Bit: 0 | Length: 8

Once the CAN message is configured as either an 'Analog Voltage Input' or 'Digital Level Input', they appear as a virtual input on the Sensors node. You can then configure the virtual sensor and calibrate it like a standard analog or digital input (see Setups – Sensors).



Input Sensor Pairs

Analog Inputs (40)

Digital Inputs (10)

Virtual Analog Inputs (1)

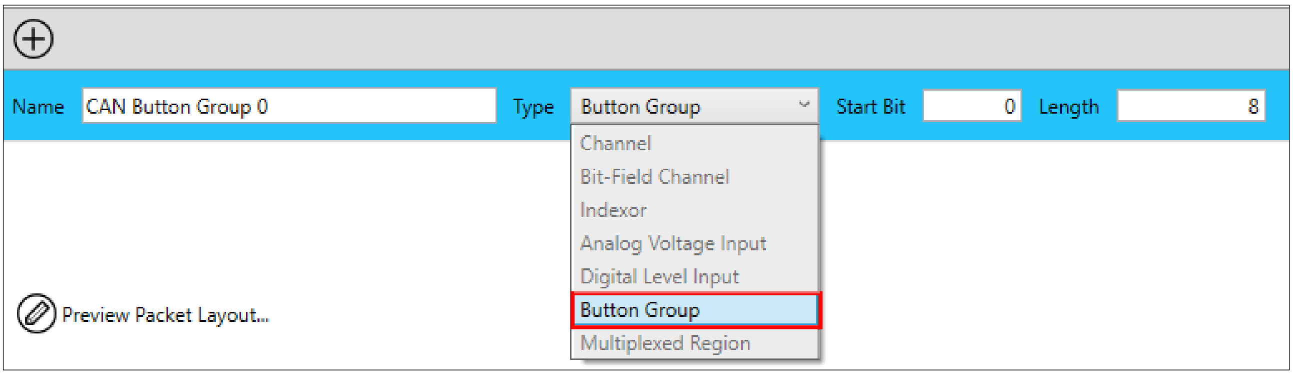
CAN Analog Input 1
Virtual Analog Voltage Input
No Termination

Virtual Digital Inputs (1)

CAN Digital Input 1
Virtual Digital Level Input
Digital Level Input

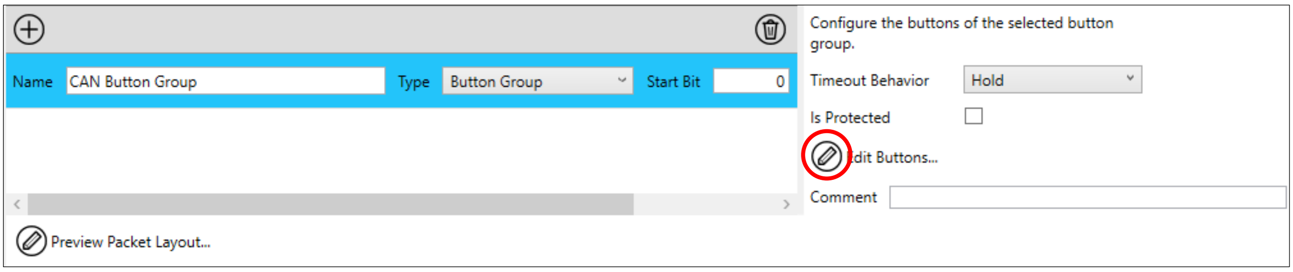
Buttons groups

You can also configure CAN messages to be interpreted as button presses. For example, bits within an 8-bit message can be configured to act as different button press types (for example clicked, click latched, held, and so on).



Name: CAN Button Group 0 | Type: Button Group | Start Bit: 0 | Length: 8

Once a CAN message is configured to be a ‘Button Group’, click **Edit Buttons...** to add buttons.



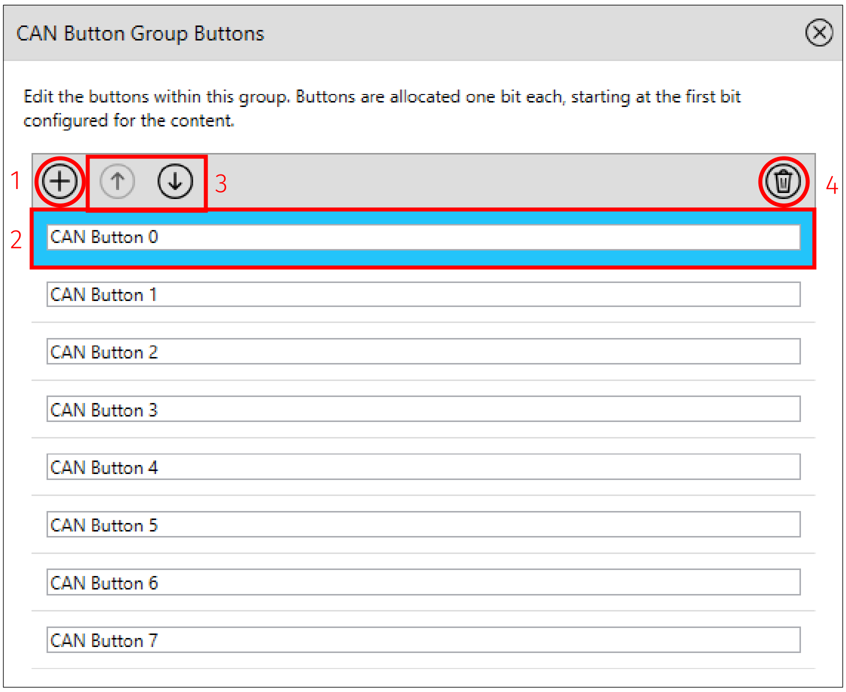
Name: CAN Button Group | Type: Button Group | Start Bit: 0

Configure the buttons of the selected button group.

Timeout Behavior: Hold

Is Protected ☐

You can then use the + button (1) to add button inputs and name them in the text box (2.) Use the ‘reorder arrows’ (3) to change the button order, Use the ‘bin’ tool to delete buttons (4).



CAN Button Group Buttons

Edit the buttons within this group. Buttons are allocated one bit each, starting at the first bit configured for the content.

Refer to Setups – Buttons for more information about how to configure a CAN button.

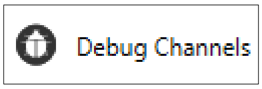
CAN CHP2 debug channels

<CAN Device Name>_RxPackets	Increments by one when a packet received is read (that is, at the receive rate)
<CAN Device Name>_TxPackets	Increments by one when a packet being sent is written (that is. at the transmit rate)
<CAN Device Name>_BusOffCount	Increments each time a device is detected in the bus off state and is restarted
<CAN Device Name>_BusState	Bit encoded see below: • Bit 0 : Error State - Set when the device is ERROR_ACTIVE (as per the CAN specification) • Bit 1 : Bus Warn - Set when bus is heavily disturbed (as per CAN spec one of the CAN Error counters exceeds 96) • Bit 2 : Bus Off - Module is in the bus off state Cleared when the device is ERROR_PASSIVE (as per the CAN specification)

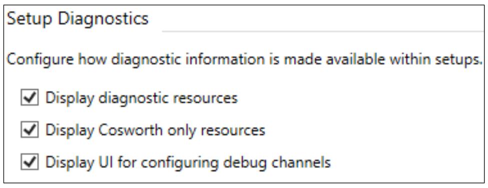
<CAN Device Name>_LastErrorCode	Numerically encoded see below: • Value = 0 - Received or transmitted a packet successfully • Value = 1 - Bit Stuff Error – More than 5 equal bits in sequence received where this is not allowed • Value = 2 - Format Error - A fixed format part of a received frame has the wrong format • Value = 3 - No ACK – A frame transmitted has not been acknowledged by another node • Value = 4 - Dominant Bit Error – During the transmission of a packet (except for the arbitration field) the device wanted to send a recessive level, but the monitored bus level was dominant • Value = 5 - Recessive Bit Error – During the transmission of a packet, or acknowledge, or active error flag, or overload flag the device wanted to send a dominant bit, but the monitored bus level was recessive • Value = 6 - Received CRC error – The CRC of the packet received does not match the calculated CRC • Value = 7 - No CAN Bus errors detected since last update

Debug channels setup

To enable these channels first create them in the setup. On the **Debug Channels** node, select **Advanced**.



Note: If this node is not available on the setup, enable it on the **Settings** tab under **Setup Diagnostics**.



Setup Diagnostics

Configure how diagnostic information is made available within setups.

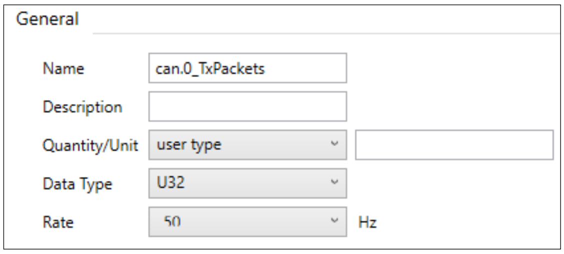
☑ Display diagnostic resources

☑ Display Cosworth only resources

☑ Display UI for configuring debug channels

Click the **+** button to create a channel. The **Name** must match the name used in the above table (<CAN Device Name> = can.0 (for CAN port 1), <CAN Device Name> = can.1 (for CAN port 2), and so on).

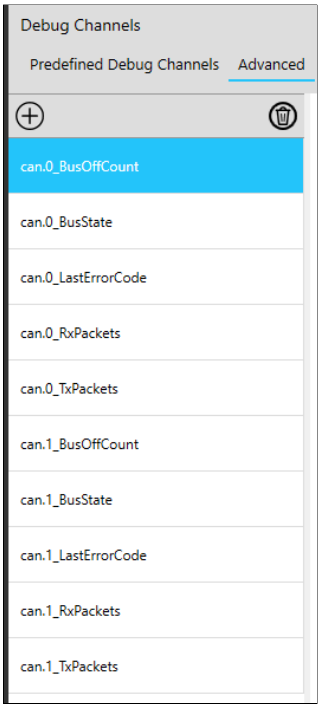
Set **Quantity/User** to 'user type', **Data Type** to'U32', and **Rate** to '5Hz', as shown below.



General

Name	can.0_TxPackets	
Description		
Quantity/Unit	user type	
Data Type	U32	
Rate	50	Hz

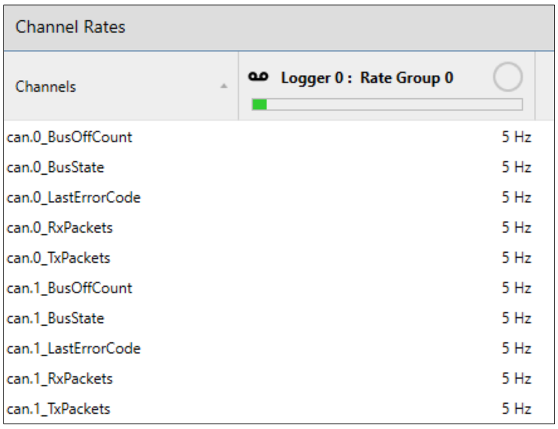
For two CAN ports there should be these channels.



Debug Channels

- can.0_BusOffCount
- can.0_BusState
- can.0_LastErrorCode
- can.0_RxPackets
- can.0_TxPackets
- can.1_BusOffCount
- can.1_BusState
- can.1_LastErrorCode
- can.1_RxPackets
- can.1_TxPackets

The channels are then available to log:



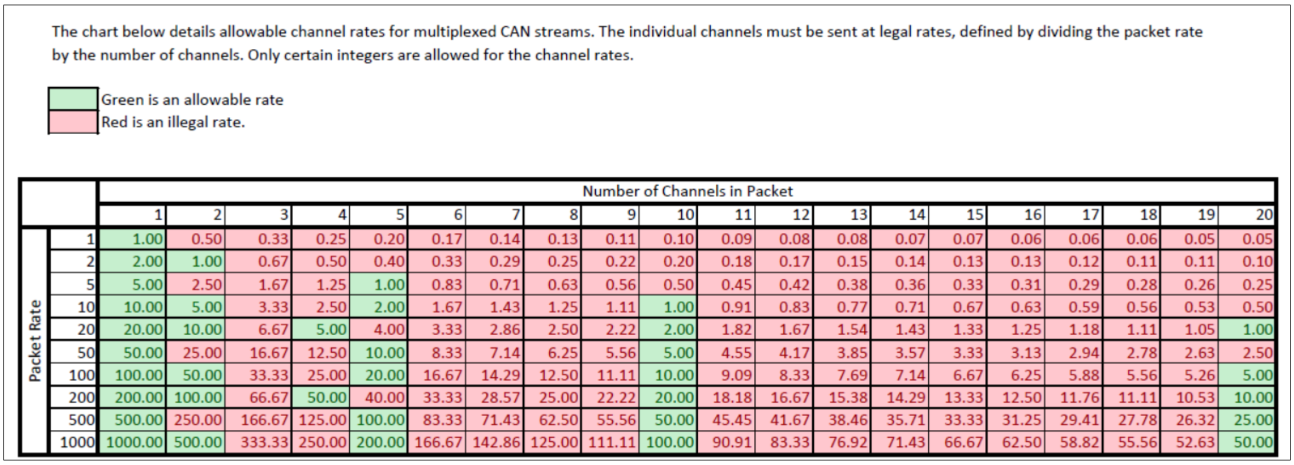
Channel Rates

Channels	Logger 0 : Rate Group 0
can.0_BusOffCount	5 Hz
can.0_BusState	5 Hz
can.0_LastErrorCode	5 Hz
can.0_RxPackets	5 Hz
can.0_TxPackets	5 Hz
can.1_BusOffCount	5 Hz
can.1_BusState	5 Hz
can.1_LastErrorCode	5 Hz
can.1_RxPackets	5 Hz
can.1_TxPackets	5 Hz

Note: When You create the channel, it might display as empty. To save the setup, close it, and then re-open it.

Allowable rates

When multiplexing, only certain rates are allowed for the packet. This value is dependent on the number of multiplexing states. This table defines what is allowed in Toolset for all of Cosworth devices.



The chart below details allowable channel rates for multiplexed CAN streams. The individual channels must be sent at legal rates, defined by dividing the packet rate by the number of channels. Only certain integers are allowed for the channel rates.

🟩 Green is an allowable rate
🟥 Red is an illegal rate.

		Number of Channels in Packet																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Packet Rate	1	🟩 1.00	0.50	0.33	0.25	0.20	0.17	0.14	0.13	0.11	0.10	0.09	0.08	0.08	0.07	0.07	0.06	0.06	0.06	0.05	0.05
	2	🟩 2.00	🟩 1.00	0.67	0.50	0.40	0.33	0.29	0.25	0.22	0.20	0.18	0.17	0.15	0.14	0.13	0.13	0.12	0.11	0.11	0.10
	5	🟩 5.00	2.50	1.67	1.25	🟩 1.00	0.83	0.71	0.63	0.56	0.50	0.45	0.42	0.38	0.36	0.33	0.31	0.29	0.28	0.26	0.25
	10	🟩 10.00	🟩 5.00	3.33	2.50	🟩 2.00	1.67	1.43	1.25	1.11	🟩 1.00	0.91	0.83	0.77	0.71	0.67	0.63	0.59	0.56	0.53	0.50
	20	🟩 20.00	🟩 10.00	6.67	🟩 5.00	4.00	3.33	2.86	2.50	2.22	🟩 2.00	1.82	1.67	1.54	1.43	1.33	1.25	1.18	1.11	1.05	🟩 1.00
	50	🟩 50.00	25.00	16.67	12.50	🟩 10.00	8.33	7.14	6.25	5.56	🟩 5.00	4.55	4.17	3.85	3.57	3.33	3.13	2.94	2.78	2.63	2.50
	100	🟩 100.00	🟩 50.00	33.33	25.00	🟩 20.00	16.67	14.29	12.50	11.11	🟩 10.00	9.09	8.33	7.69	7.14	6.67	6.25	5.88	5.56	5.26	🟩 5.00
	200	🟩 200.00	🟩 100.00	66.67	🟩 50.00	40.00	33.33	28.57	25.00	22.22	🟩 20.00	18.18	16.67	15.38	14.29	13.33	12.50	11.76	11.11	10.53	🟩 10.00
	500	🟩 500.00	250.00	166.67	125.00	🟩 100.00	83.33	71.43	62.50	55.56	🟩 50.00	45.45	41.67	38.46	35.71	33.33	31.25	29.41	27.78	26.32	🟩 25.00
	1000	🟩 1000.00	🟩 500.00	333.33	250.00	🟩 200.00	166.67	142.86	125.00	111.11	🟩 100.00	90.91	83.33	76.92	71.43	66.67	62.50	58.82	55.56	52.63	🟩 50.00