BusLink CR2 CAN Repeater - User Manual

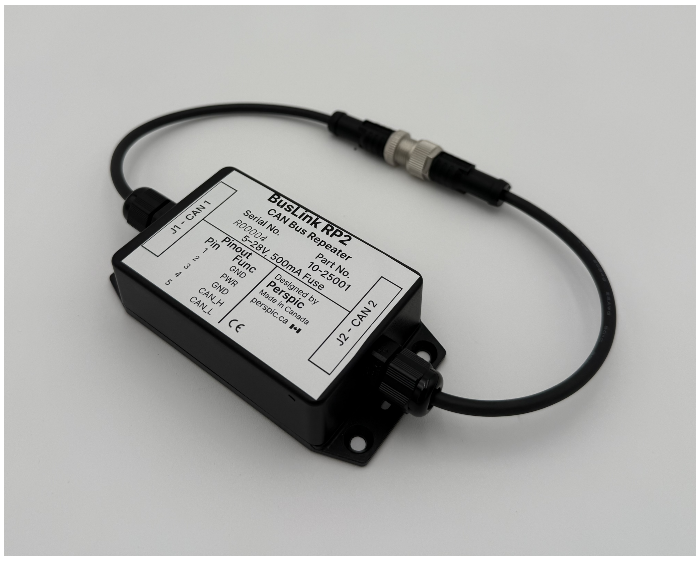


Figure 1: BusLink CR2

Overview

The Perspic BusLink CR2 (CR2 = Can Repeater, 2-channel) is a rugged CAN Bus repeater designed to extend, isolate, and stabilize communication across two CAN networks. The CR2 does not record traffic; instead, it transparently relays messages between CAN1 and CAN2 while providing fused power distribution between the two sides. This makes it ideal for network extension, segmentation, and environments where reliable message forwarding is critical.

Compact and straightforward to deploy, the BusLink CR2 offers automotive-grade durability with mounting flanges and multiple connector options. The M12 version uses two separate M12 connectors (one for CAN1, one for CAN2). The Deutsch version uses a single 6-pin connector for both buses. The standard unit is rated IP54 for protection against dust and splashing water, with an IP68 option for applications requiring full submersion resistance.

Key Features

- 2-Channel CAN Bus data repeater
- Isolates and extends CAN Bus networks
- Supports standard CAN baud rates from 5 kbit/s to 1 Mbit/s
- Fused power distribution between CAN Bus networks
- Configurable baud rates and message filtering
- Configuration via USB-C and JSON file
- Selectable 120 Ω termination (solder jumper)
- Wide input voltage range (5–28 VDC) with reverse polarity protection
- IP54 rating (splash-resistant); IP68 (submersion-resistant) optional

Applications

Network Isolation

The CR2 can be used to isolate two CAN networks, ensuring that faults such as bus pulled into dominant state, miswiring, or excessive bus load on one side do not propagate to the other. This is especially useful in systems where sensitive control modules need to be protected from noisy or unstable subsystems.

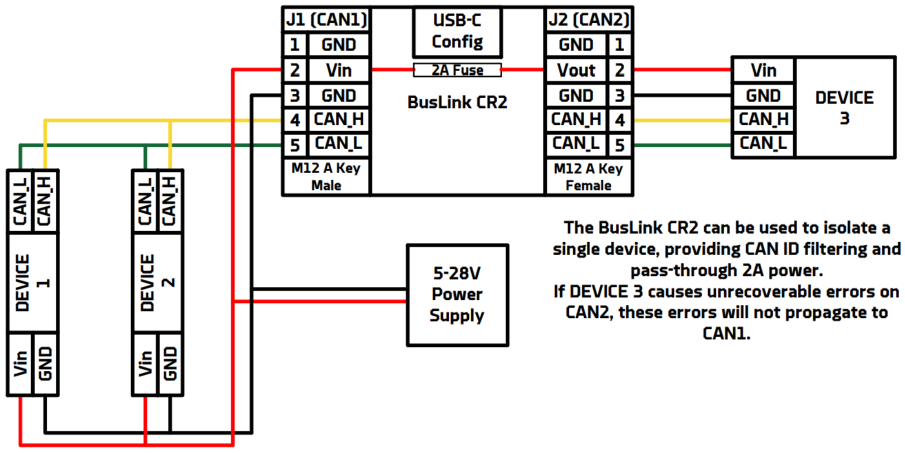


Figure 2: Network Isolation Example

Baud Rate Translation / Conversion

In many real-world deployments, not all CAN devices operate at the same baud rate. The CR2 can bridge two buses running at different speeds, allowing otherwise incompatible devices to communicate. For example, a sensor operating at 125 kbit/s can be connected to a controller running at 500 kbit/s without requiring hardware changes.

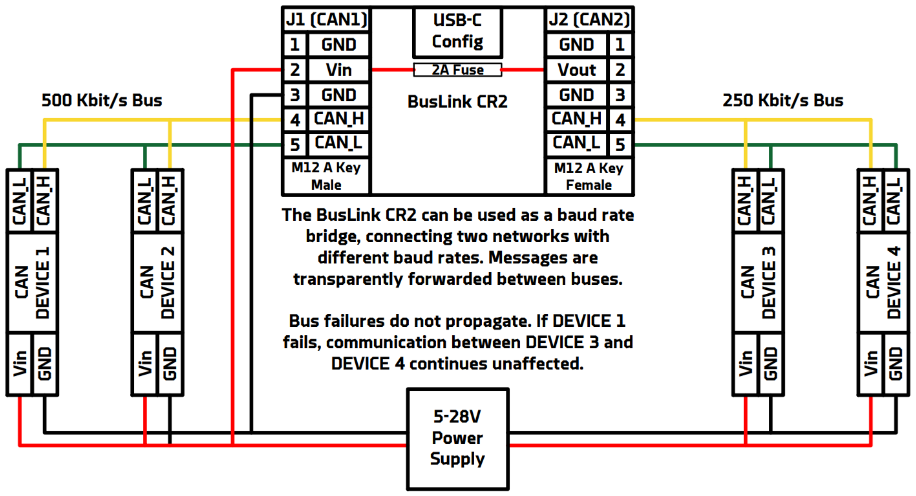


Figure 3: Baud Rate Differential Example

CAN ID Filtering

The CR2 can selectively forward messages based on CAN ID, allowing only relevant traffic to pass between networks. This helps reduce bus congestion, improve determinism, and shield nodes from unnecessary data. Common applications include isolating diagnostic traffic, filtering out chatter from high-traffic modules, or exposing only a subset of IDs to a third-party tool.

View CAN Data on a Computer

When connected via USB, the CR2 can act as a live interface to PC software such as SavvyCAN. This allows engineers to monitor, log, and analyze CAN traffic in real time without needing a separate dedicated CAN adapter. It effectively doubles as both a repeater and a diagnostic interface, making it highly versatile for lab testing and field troubleshooting.

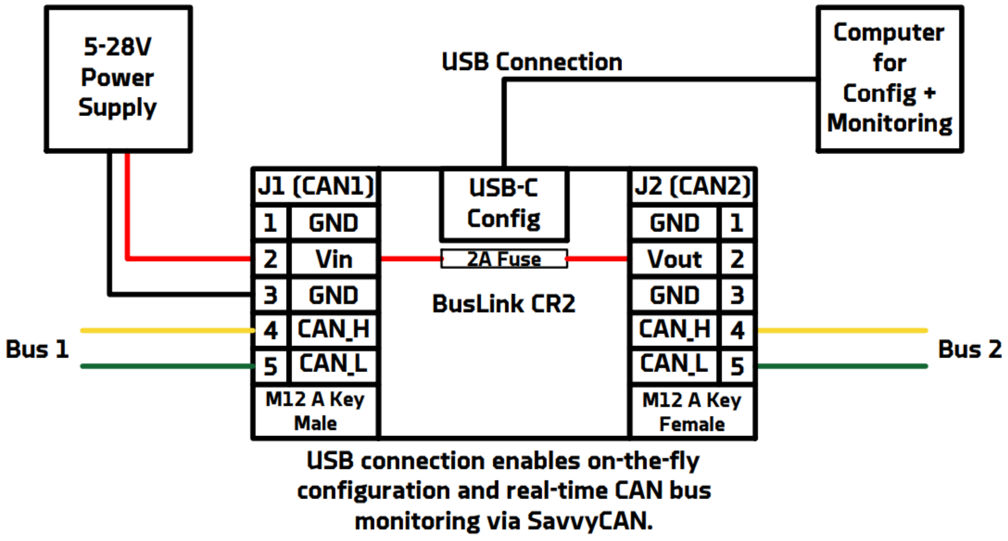


Figure 4: Computer Connection Example

Device Specifications

Electrical

- *Input Voltage*: 5-28 V DC
- *Current Draw*: 30-60 mA
- *CAN Baud Rate*: 5, 10, 50, 100, 125, 250, 500, 800, 1000 Kbit/s (configured over USB Console)
- *CAN Filters*: Up to 128 standard (11-bit) and 64 extended (29-bit) filters per bus
- *CAN Resistors*: Optional 120 Ω termination resistors (solder jumpers)
- *USB Interface*: USB-C for configuration and console debugging
- *Power Protection*: Reverse polarity protection for internal circuitry and 2A in-line fuse between CAN1 and CAN2 on V+ line

Pin Functions

M12 A-Key Style

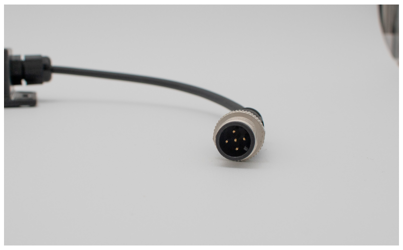


Figure 5: M12 Male Plug

Pin Number	Function
1	GND
2	V+
3	GND
4	CAN_H
5	CAN_L

TE Example: T4110001051-000
CAN1 - Left Connector is M12 Male (Pins)
CAN2 - Right Connector is M12 Female (Sockets)

6-Pin Deutsch/T Style

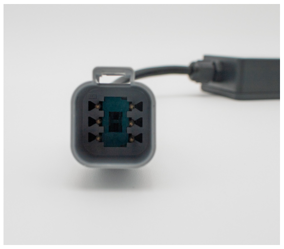


Figure 6: Deutsch/AT 6 Pin Male Plug

Pin Number	Function
1	CAN1_H
2	CAN1_L
3	VCC/PWR
4	GND
5	CAN2_L
6	CAN2_H

AMP Example: AT06-6S (DT06-6S)
NOTE: CAN_L and CAN_H pins are swapped compared to M12 version

Mechanical Specifications

Mechanical - IP54 ABS Enclosure (Standard)

Dimension	Measure (Pigtail)
Width	5.8 cm
length	13 cm
height	3 cm
Weight	110 g
Cable Length	15 cm
IP Rating	IP54
Hole Pattern	Rectangular 4x 5mm countersunk through holes
Temperature	-20 to 80 C
Humidity	0 to 95% (Non-Condensing)

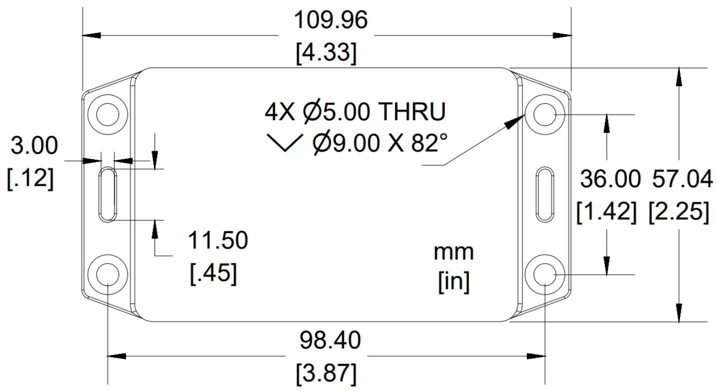


Figure 7: IP54 Flange Mounting Pattern

Mechanical - IP68 Polycarbonate Enclosure

Dimension	Measure (Pigtail)
Width	9 cm
length	13 cm
height	3.4 cm
Weight	200 g
Cable Length	15 cm
IP Rating	IP68
Hole Pattern	Rectangular 4x 5mm countersunk through holes
Temperature	-40 to 85 C
Humidity	0 to 100%

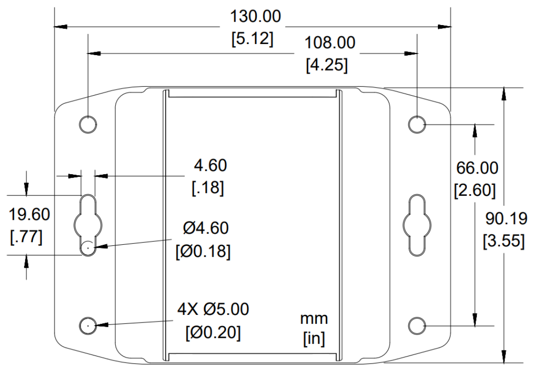


Figure 8: IP68 Flange Mounting Pattern

Device Operation

Quick Start Guide

1. Determine Desired Behavior

Use Case	CAN1 Baud	CAN2 Baud	Filtering	Configuration Required?
Simple bridge	500k	500k	None	No (default)
Baud rate conversion	e.g. 250k	e.g. 500k	None	Yes
ID filtering	Any	Any	Custom	Yes
Network isolation	Any	Any	Optional	As Required

2. Connect Hardware

- Connect CAN1 (M12 Male) to first network: CAN_H, CAN_L, GND
- Connect CAN2 (M12 Female) to second network: CAN_H, CAN_L, GND
- Apply 5–28V DC power to either CAN1 or CAN2 V+ pin
- Ensure proper bus termination (120Ω at each physical end of the bus)

3. Verify Operation

- Green LED (power): ON indicates internal power is good
- Status LED: Green = normal operation, Blue = bus warning, Red = fault

4. Configure (if needed)

- Connect USB-C to a PC
- Open a terminal (PuTTY, Tera Term) at any baud rate (USB is virtual COM port, baud rate does not matter)
- Use `can 1 baud <rate>` and `can 2 baud <rate>` to set baud rates
- Use `status` to verify bus communication

Default Configuration

The BusLink CR2 comes pre-configured with the following default settings:

- CAN1+CAN2 Baud Rate: 500 Kbit/s
- CAN1+CAN2 Mode: Normal
- Accept all frames (standard and extended)
- No filtering (all messages accepted)
- CAN Passthrough Mode: Enabled

USB Connection

The BusLink CR2 has an onboard USB-C connector that provides console services over USB. When connected to a Windows PC, it will appear as a USB Composite Device with a COM port.

- USB Vendor ID 0xAD50
- USB Product ID: 0x60C4.

USB Console

The USB device console allows you to configure settings and view status. Simply connect the USB-C cable to a computer, and the device will appear as a console device. Use a console program like Tera Term or PuTTY to access menus and configure the CAN Bus.

Command **help**, **h** or **?**: Show help message

Command **version** or **v**: Print the software version

Command **status** or **stat**: Show advanced CAN Bus status

Command **config** or **cfg**: Show or set CAN Bus configuration as JSON string (See Configuration command section)

Command **passthrough** or **pt**: Show or Toggle CAN Passthrough mode. E.G. `passthrough` (show), `passthrough on` (set), `passthrough off` (unset)

Command **led** or **l**: Show or set LED status. E.g. `led` (show), `led r on`, `led g off`, `led b tog`

Command **can** or **c**: Show or set CAN Bus configuration. (See CAN command section)

Command **reboot** or **rb**: Reboot the device Command **xE7**: Send this special character to toggle CAN Serial setup (GVRET mode)

Up to 10 previous commands can be accessed by pressing the up arrow key.

Configuration Command

The configuration command allows you to view or set the CAN Bus configuration using a JSON string. The configuration is stored on the local memory after each change and is loaded on boot.

config or **config show**: Print the current configuration JSON String.

config set: Set the configuration using a JSON string. After typing `config set` and pressing enter, paste the JSON string. Once the device receives the final closing bracket, the device will validate the JSON and apply the new configuration.

NOTE: we recommend modifying the configuration using the `can` command first, then copying the current configuration using `config show` before uploading it to the device. This ensures a valid configuration. Direct editing of the JSON file is possible, but the JSON must be valid.

config reset: Reset the configuration to factory defaults.

Example configuration JSON string:

```json
{
    "can1": {
        "bus_name": "can_1",
        "baudrate": 500,
        "accept_standard_non_matching_frames": true,
        "accept_extended_non_matching_frames": true,
        "bus_enabled": true,
        "listen_only": false,
        "scan_enable": false,
        "standard_filters": [],
        "extended_filters": []
    },
    "can2": {
        "bus_name": "can_2",
        "baudrate": 500,
        "accept_standard_non_matching_frames": true,
        "accept_extended_non_matching_frames": true,
        "bus_enabled": true,
        "listen_only": false,
        "scan_enable": false,
        "standard_filters": [],
        "extended_filters": []
    }
}
```

CAN Command

The CAN command allows you to view or set individual CAN Bus settings.

Note: shortcuts for CAN command arguments are available and can stack. For example, `can 1 baud 500` can be shortened to `c 1 b 500`.

can help or **can h** or **can ?** - Show help message for CAN commands.

can settings or **can s** or **c settings** or **c s** - Show a summary of the current CAN Bus settings.

can <1|2> baud or **can <1|2> b** - Show or set the baud rate for CAN Bus 1 or 2. Supported baud rates are: 5, 10, 50, 100, 125, 250, 500, 800, 1000 (in Kbit/s).
Example: `can 1 baud 500` or `c 2 b 125`

can <1|2> sensitivity or **can <1|2> sens** - Show or set the bus degrade sensitivity for CAN Bus 1 or 2. Supported options are:

- `none` - No automatic degrade
- `low` - Low sensitivity, degrade into "CAN_SUSPECT" state only (disable retransmits on transmit errors)
- `medium` - Medium sensitivity, degrade into "CAN_DEGRADED" state (listen-only mode)
- `high` - High sensitivity (default), degrade into "CAN_OFFLINE" state (disable all tx and rx, put can into sleep mode until recovery)

Sensitivity controls how aggressively the device reacts to CAN errors and how far it will automatically degrade bus activity to protect the network. Higher sensitivity detects smaller bursts of protocol errors or rising error counters sooner and allows escalation to stricter behaviors: High may fully take the bus OFFLINE (no TX/RX) until attempting recovery; Medium will at most enter listen-only (DEGRADED) to observe traffic without transmitting; Low only limits retransmissions briefly (SUSPECT); None disables automatic degradation and keeps normal operation regardless of errors.

Choose a higher sensitivity for noisy or safety-critical networks where it's preferable to back off quickly rather than risk adding load to a faulted bus. Use lower sensitivity on stable networks or in scenarios where maintaining transmission through transient disturbances is more important than aggressive protection, for example when a tool or node is actively scanning baud rates. Sensitivity is set per bus and can be adjusted live via the CLI or saved in the configuration JSON.

can <1|2> mode or **can <1|2> m** - Show or set the mode for CAN Bus 1 or 2. Supported modes are:

- `normal` - Normal mode (default)
- `listen` - Listen-only mode

can <1|2> accept or **can <1|2> a** - Show or set the acceptance filter for CAN Bus 1 or 2. Supported options are:

- `std` - Accept only standard (11-bit) CAN IDs
- `ext` - Accept only extended (29-bit) CAN IDs
- `both` - Accept both standard and extended CAN IDs (default)

NOTE: Accept refers to "non-matching" frames based on filters. If filters are configured they will take precedence over this setting. All frames that do not match a filter will be accepted or rejected based on this setting. See filter diagram below for flow chart on how filtering works.

can <1|2> filter or **can <1|2> f** - Show or set the CAN ID filtering for CAN Bus 1 or 2.

can <1|2> filter list - Print all filters for the selected CAN Bus.

can <1|2> filter add - Add a new filter to the selected CAN Bus.

- `<frame type>` - `std` for standard (11-bit) CAN IDs, `ext` for extended (29-bit) CAN IDs
- `<filter style>` - `range` to match all frames between ID1 and ID2, `dual` to match ID1 or ID2, `classic` for a mask-based filter
- `<id1>` - The first ID in the filter (for `range` ID1 must be less than ID2, for `dual` ID1 and ID2 can be in any order, for `classic` this is the filter ID)
- `<id2>` - The second ID in the filter (for `classic` this is the mask)
- `<action>` - `accept` to accept matching frames, `reject` to reject matching, `accept-priority` to accept and set highest priority (0) for matching frames

Example: `can 1 filter add std range 0x123 0x456 reject` Short form example: `c 2 f add s r 0x123 0x456 r`

can <1|2> filter delete - Remove a filter from the selected CAN Bus by its index number. The index number can be found using `can <1|2> filter list`.

can <1|2> filter clear - Remove all filters of the specified frame type from the selected CAN Bus. `<frame type>` can be `std`, `ext` or `all`.

> **Note:** Each CAN bus supports a maximum of 128 standard (11-bit) filters and 64 extended (29-bit) filters. Use `can <1|2> filter list` to see current filter usage.

Filtering Diagram

The diagram below illustrates how CAN ID filtering works. Filters are evaluated in the order they are listed, and the first matching filter determines the action taken. If no filters match, the default acceptance setting is applied.

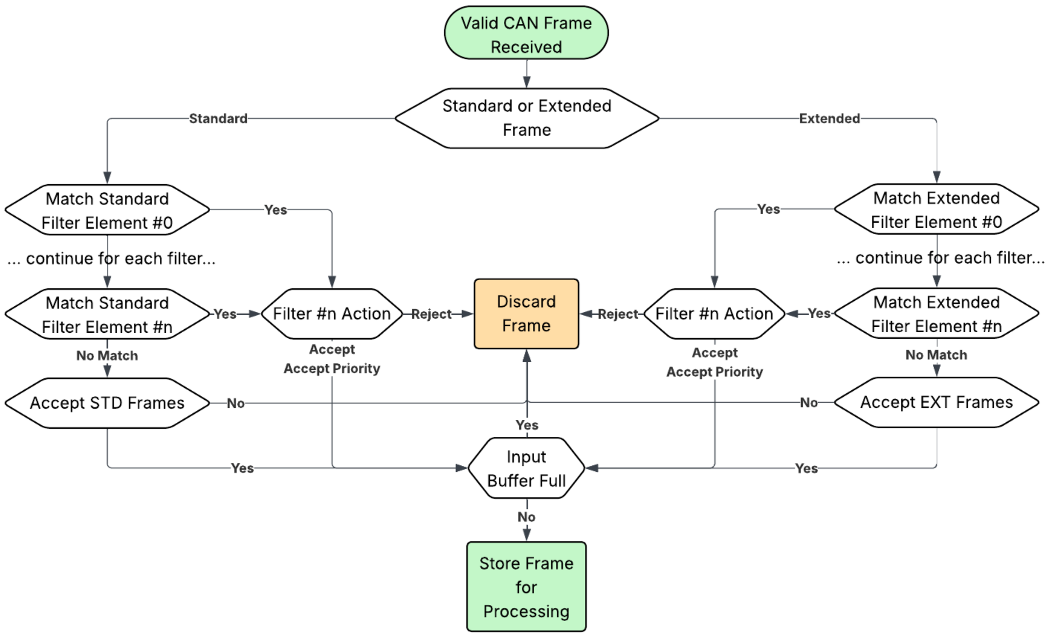
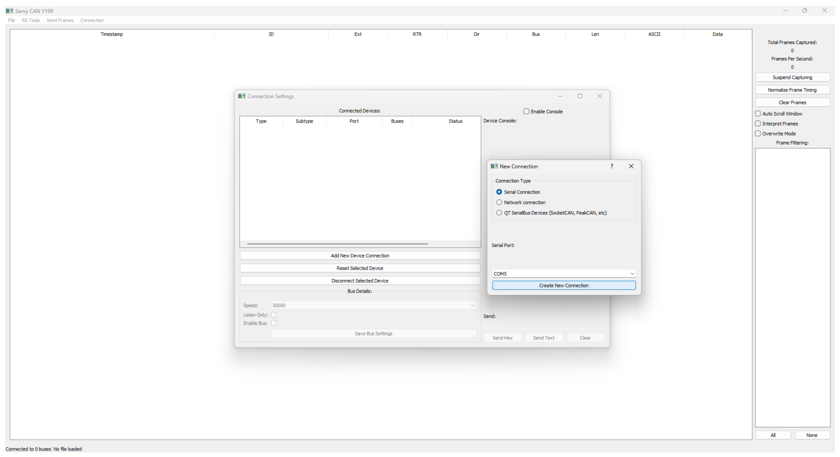

Valid CAN Frame Received
Standard or Extended Frame
Standard
Extended
Match Standard Filter Element #0
Yes
... continue for each filter...
Match Standard Filter Element #n
Yes
Filter #n Action
Reject
Accept
Accept Priority
Discard Frame
Reject
Filter #n Action
Accept
Accept Priority
Yes
Match Extended Filter Element #0
... continue for each filter...
Match Extended Filter Element #n
Yes
No Match
Accept STD Frames
No
No Match
Accept EXT Frames
No
Yes
Input Buffer Full
Yes
Yes
No
Store Frame for Processing

Figure 9: Filtering Diagram

Can Serial Setup

The USB Console can be function as a *GVRET* device and can be used with SavvyCAN.



Connection Type: Serial Connection

Serial Port: COM5

Speed: 50000

Figure 10: SavvyCan Connection

More information: GVRET Github

Notes for GVRET mode: - CAN Bus numbers are 0 (Bus 1) and 1 (Bus 2) - Bus speeds must be configured before connecting - Console is locked while in GVRET mode

Advanced Device Operation Modes

Viewing CAN Data on the Console

Once the device busses are configured, CAN data can be printed to the USB console to troubleshoot connection issues. Send commands `can 1 print on` and `can 2 print on` to toggle printing of raw CAN messages to the console.

Passthrough Mode

On the repeater CAN Passthrough Mode is ON by default and resets to ON every reboot. If disabling of passthrough mode is required run command `passthrough off` to disable passthrough.

Performance and System Behavior

Bus Errors

If the repeater detects protocol errors, it will incrementally move the the CAN hardware into warning and degraded states to prevent bus arbitration errors for other nodes.

CAN_OK: No errors detected.

CAN_SUSPECT: 3 errors within 50ms, transmit error count > 96 or receive error count > 50. In this mode the CAN hardware will transmit new frames once and will not retry if transmission fails.

CAN_DEGRADED: 10 errors within 50ms, transmit error count > 128 or receive error count > 100. In this mode the CAN hardware will be in listen-only mode and will not transmit any frames or respond with any ack bits.

CAN_OFFLINE: CAN Hardware went to "BUS OFF" state, Transmit error count > 254, or receive error count > 126. In this mode the CAN hardware will be in sleep mode and will not transmit or receive any frames.

During *CAN_SUSPECT* and *CAN_DEGRADED* states, incoming messages are stored in a 255-message software buffer per bus. This allows the device to continue receiving traffic while transmission is limited, and queued messages will be forwarded once the bus recovers to normal operation.

The device will attempt to recover from degraded states automatically as follows.

From *CAN_OFFLINE*: Wait between 0.25 seconds incrementing to 8 seconds, then attempt to reinitialize the CAN hardware in *CAN_DEGRADED* mode. If the bus remains degraded after retries, it will retry every 8 seconds until the bus recovers.

From *CAN_DEGRADED*: Wait 1.5 seconds, then attempt to reinitialize the CAN hardware in *CAN_SUSPECT* mode. If the bus remains degraded after retries, it will retry every 1.5 seconds until the bus recovers.

From *CAN_SUSPECT*: Wait 0.5 seconds, then attempt to reinitialize the CAN hardware in *CAN_OK* mode. If the bus remains suspect, it will retry every 0.5 second until the bus recovers.

Traffic Load Imbalance

When CAN1 and CAN2 have significantly different traffic loads (for example, a high-traffic 1 Mbit/s bus relaying to a slower 125 kbit/s bus), messages are buffered as much as possible. If the buffer fills (255 messages per bus), the newest incoming frames will be discarded until space becomes available.

Relay Latency

The delay from end of message reception to start of transmission on the opposite bus is approximately 25 µs ±2 µs.

LED Status Lights

- Green LED (beside power connector) - Indicates that internal device power is on.
- Status LED (beside USB-C Connector) - Indicates device runtime status:
  - Red - Configuration Failed or one CAN bus is in *CAN_OFFLINE* state
  - Green - Device is operating normally
  - Blue - one CAN bus is in *CAN_SUSPECT* or *CAN_DEGRADED* state

CAN Termination Resistor Jumpers

Termination resistors are installed and enabled by default. To remove termination resistance cut the trace between the resistor and the jumper. To re-enable termination solder a small blob of solder between the resistor and the jumper.

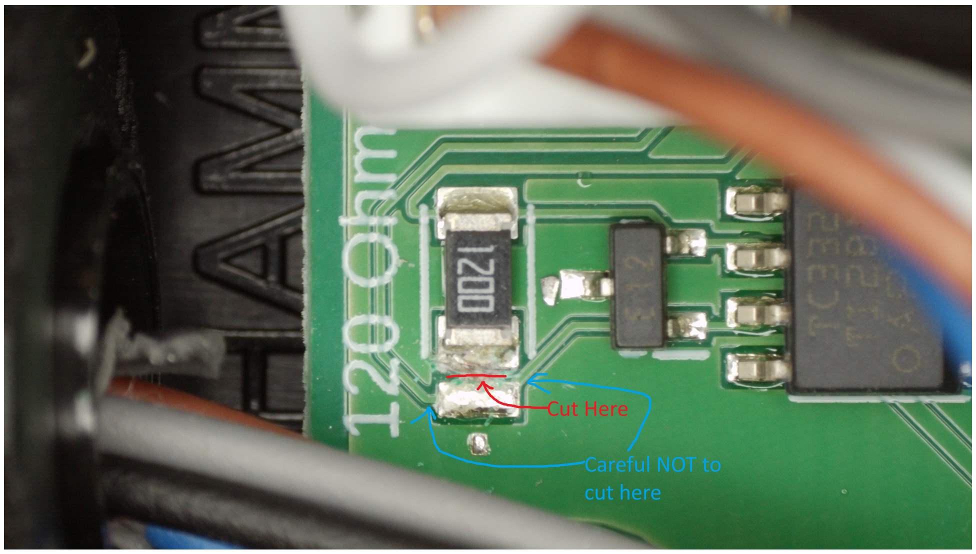


Figure 11: CAN Termination Resistors

Tests and Certifications

RF Emissions: Tested per CISPR22 Edition 5.2 2006-03 Class A
EMC: Tested per IEC 61000-4-2 Edition 3.2 2010-04 Level 2
ESD Immunity: Tested per IEC 61000-4-2 Edition 2.0 2008-12 Class 1

Part Numbers

Part Number	IP Rating	Wiring Option	Temperature Range	Connector
10-25001	IP54	15 cm pigtail	-20 to 80 C	M12 5-Pin x2
10-25022	IP68	15 cm pigtail	-40 to 85 C	M12 5-Pin x2

Visit our website at perspic.ca for the latest part numbers and configurations.

Don’t see the configuration you need? Contact Us for specialty configurations and custom connectors

Warranty and Returns

1-Year Limited Warranty

Perspic Inc warrants that this product will be free from defects in materials and workmanship under normal use for one (1) year from the date of delivery to the original purchaser. This warranty is non-transferable.

What's Covered:

- Manufacturing defects in material or assembly
- Defects resulting from improper workmanship under normal usage conditions

What's Not Covered:

- Damage due to misuse, abuse, mishandling, or improper installation
- Normal wear and tear, cosmetic blemishes, or environmental damage (e.g., corrosion, humidity, extreme temperatures beyond rated specifications)
- Modifications or repairs not authorized by Perspic Inc
- Damage resulting from accidents, neglect, or force majeure events
- Third-party components not manufactured by Perspic Inc

Warranty Claims

To make a warranty claim:

1. Contact Perspic Inc in writing or via email within the warranty period
2. Provide proof of purchase (e.g., invoice or order confirmation)
3. Return the product for evaluation if requested (shipping at client's expense)

Perspic Inc will, at its discretion, repair, replace, or issue a credit for any product found defective under this warranty. Replacement parts or products are warranted only for the remainder of the original warranty period.

Returns

If you are not satisfied with your purchase, you may request a return within 30 days of delivery. Items must be unused, in original condition, and in original packaging. Contact support@perspic.ca with your order number and reason for return.

Non-returnable items: Customized or special-order items, final sale or clearance items.

Return shipping: Customers are responsible for return shipping costs unless the item was damaged or incorrect. Use a trackable shipping service; Perspic Inc is not responsible for items lost in return transit.

Limitation of Liability

Perspic Inc shall not be liable for incidental, indirect, or consequential damages, including loss of profit, downtime, or damage to other equipment. This warranty is the sole remedy and replaces all other warranties, express or implied, including implied warranties of merchantability or fitness for a particular purpose.

This warranty is governed by the laws of the Province of Ontario, Canada.

Definitions

- **CAN (Controller Area Network)** – A robust vehicle bus standard designed to allow micro-controllers and devices to communicate without a host computer.
- **Baud Rate** – The speed at which data is transmitted over the CAN network, measured in Kbit/s (e.g., 250 Kbit/s, 500 Kbit/s).
- **CAN Frame** – A single message sent on the CAN bus, containing an ID, data length, and data payload.
- **Standard CAN ID (11-bit)** – The shorter CAN identifier format, using 11 bits for the message ID.
- **Extended CAN ID (29-bit)** – The longer CAN identifier format, using 29 bits for the message ID.
- **DLC (Data Length Code)** – Specifies the number of bytes in a CAN frame's data payload (0-8 bytes for CAN 2.0, up to 64 bytes for CAN-FD).
- **Termination Resistor** – A 120Ω resistor placed at both ends of a CAN bus to prevent signal reflections.
- **Listen-Only Mode** – A mode where the CAN repeater passively listens to messages without transmitting or acknowledging them.
- **Passthrough Mode** – A mode where messages received on one CAN bus are retransmitted to the second CAN bus.
- **CSV (Comma-Separated Values)** – A simple file format for logging CAN messages, readable by spreadsheet software.
- **RTC (Real-Time Clock)** – A hardware clock used to timestamp CAN messages, retaining time even when powered off.
- **JSON (JavaScript Object Notation)** – A text-based configuration file format used for device settings.
- **GVRET (General Vehicle Reverse Engineering Tool)** – A protocol used by software like SavvyCAN for real-time CAN data viewing.
- **SavvyCAN** – A software tool for viewing, analyzing, and sending CAN messages using compatible hardware.
- **IP54 Rating** – Indicates the device is protected against limited dust ingress and splashing water.
- **IP68 Rating** – Indicates the device is fully protected against dust ingress and long-term water submersion.
- **USB Console** – A serial communication interface used for configuring and monitoring the CAN repeater.
- **Power Cycling** – The process of turning the device off and on to reset or apply new settings.

Version Changelog

V1.0.0

Initial Release Version

V1.0.1

- Added `can <1|2> sensitivity` command and associated behavior to set bus degrade sensitivity.

V1.0.3

- Updated pinout table for M12 A-Key connector to correct power and ground pins.
- Updated default configuration to set CAN Passthrough Mode to Enabled.
- Update USB Console commands
- Add example diagrams