

simplyCAN

USB-to-CAN-Adapter

BENUTZERHANDBUCH

4.01.0001.12001 1.0 de-DE DEUTSCH



Wichtige Benutzerinformation

Haftung

Dieses Dokument wurde mit größter Sorgfalt erstellt. Bitte informieren Sie HMS Industrial Networks über Ungenauigkeiten oder Versäumnisse. Die Daten und Illustrationen in diesem Dokument sind nicht verbindlich. Wir, HMS Industrial Networks, behalten uns das Recht vor, unsere Produkte gemäß unseres Grundsatzes der kontinuierlichen Produktentwicklung zu modifizieren. Die Informationen in diesem Dokument können ohne Vorankündigung geändert werden und sollten als nicht bindend für HMS Industrial Networks angesehen werden. HMS Industrial Networks übernimmt keine Verantwortung für mögliche Fehler in diesem Dokument.

Es gibt viele Anwendungsmöglichkeiten für dieses Produkt. Die für die Anwendung des Produkts Verantwortlichen müssen sicherstellen, dass alle notwendigen Schritte getroffen wurden, um sicherzustellen, dass die Anwendung alle Anforderungen bezüglich Durchführung und Sicherheit, einschließlich aller zutreffenden Gesetze, Vorschriften, Normen und Standards entspricht.

HMS Industrial Networks übernimmt in keinem Fall die Haftung oder Verantwortung für Probleme, die entstehen könnten, durch die Nutzung undokumentierter Funktionen, zeitlichen Ablauf, oder durch funktionelle Nebeneffekte außerhalb des dokumentierten Umfangs dieses Produkts. Die Effekte, verursacht durch jegliche direkte oder indirekte Verwendung solcher Aspekte des Produkts sind nicht definiert und könnten beispielsweise Kompatibilitätsprobleme und Stabilitätsprobleme beinhalten.

Die Beispiele und Illustrationen in diesem Dokument sind ausschließlich zum Zweck der Veranschaulichung enthalten. Aufgrund der vielen Variablen und Anforderungen, die mit jeder einzelnen Implementierung verbunden sind, kann HMS Industrial Networks keine Verantwortung übernehmen für die tatsächliche Verwendung basierend auf diesen Beispielen und Illustrationen.

Schutz- und Urheberrechte

HMS Industrial Networks besitzt die Schutz- und Urheberrechte für die Technologie, die in dem, in diesem Dokument beschriebenen, Produkt integriert ist. Diese Schutz- und Urheberrechte können Patente und schwebende Patentanmeldungen in den USA und anderen Ländern beinhalten.

Inhaltsverzeichnis

Seite

1	Benutzerführung	3
1.1	Zielgruppe	3
1.2	Dokumenthistorie	3
1.3	Eingetragene Warenzeichen.....	3
1.4	Konventionen	4
2	Sicherheitsanweisungen	5
2.1	Informationen zur EMV	5
2.2	Allgemeine Sicherheitshinweise	5
2.3	Bestimmungsgemäße Verwendung	5
3	Lieferumfang	5
4	Produktbeschreibung	6
5	Installation	7
6	Betrieb	8
6.1	simplyCAN Busmonitor	8
6.2	USB LED	9
6.3	CAN LED	9
7	Zusätzliche Komponenten	10
7.1	CAN-Bus-Abschluss.....	10
8	Technische Daten	11
9	Fehlerbehebung	11
10	Reinigung	12
11	Support/Hardware zurücksenden	12
11.1	Support	12
11.2	Hardware zurücksenden	12

12	Entsorgung	12
13	API-Dokumentation	13
13.1	API-Funktionen	13
13.1.1	simply_open	13
13.1.2	simply_close	13
13.1.3	simply_initialize_can	13
13.1.4	simply_identify	14
13.1.5	simply_start_can	14
13.1.6	simply_stop_can	15
13.1.7	simply_reset_can	15
13.1.8	simply_can_status	16
13.1.9	simply_set_filter	17
13.1.10	simply_receive	18
13.1.11	simply_send	18
13.1.12	simply_get_last_error	19
13.2	Status-Diagramm	20
A	Konformitätserklärungen.....	21
A.1	EMV Konformitätserklärung (CE).....	21
A.2	Entsorgung und Recycling	21

1 Benutzerführung

Bitte lesen Sie das Handbuch sorgfältig. Verwenden Sie das Produkt erst, wenn Sie das Handbuch verstanden haben.

1.1 Zielgruppe

Dieses Handbuch richtet sich an geschultes Personal, das vertraut ist mit CAN und den geltenden Richtlinien. Der Inhalt des Handbuchs muss allen Personen, die autorisiert sind, das Produkt zu verwenden oder zu betreiben, zugänglich gemacht werden.

1.2 Dokumenthistorie

Version	Datum	Beschreibung
1.0	März 2019	Erste Version

1.3 Eingetragene Warenzeichen

Ixxat® ist ein registriertes Warenzeichen von HMS Industrial Networks AB. Alle anderen erwähnten Warenzeichen sind Eigentum der jeweiligen Inhaber.

1.4 Konventionen

Handlungsaufforderungen und Resultate sind wie folgt dargestellt:

- ▶ Handlungsaufforderung 1
- ▶ Handlungsaufforderung 2
 - Ergebnis 1
 - Ergebnis 2

Listen sind wie folgt dargestellt:

- Listenpunkt 1
- Listenpunkt 2


Fette Schriftart wird verwendet, um interaktive Teile darzustellen, wie Anschlüsse und Schalter der Hardware oder Menüs und Buttons in einer grafischen Benutzeroberfläche.

Diese Schriftart wird verwendet, um Programmcode und andere Arten von Dateninput und -output wie Konfigurationsskripte darzustellen.


Dies ist ein Querverweis innerhalb dieses Dokuments: [Konventionen, S. 4](#)


Dies ist ein externer Link (URL): www.hms-networks.com


Warnhinweise sind wie folgt dargestellt:


	<p>Quelle der Gefahr!</p> <p>Konsequenzen bei Nichtbeachtung.</p> <p>Maßnahmen um Gefahr zu vermeiden.</p>
---	--

Warnsignale und Signalworte sind abhängig vom Level der Gefahr verwendet.

	<p><i>Dies ist eine zusätzliche Information, die Installation oder Betrieb vereinfachen kann.</i></p>
---	---

	<p>Diese Anweisung muss befolgt werden, um Gefahr reduzierter Funktionen und/oder Sachbeschädigung oder Netzwerk-Sicherheitsrisiken zu vermeiden.</p>
---	---

	<p>Vorsicht!</p> <p>Diese Anweisung muss befolgt werden, um Gefahr von Verletzungen zu vermeiden.</p>
---	--

	<p>ACHTUNG!</p> <p>Diese Anweisung muss befolgt werden, um Gefahr von schweren Verletzungen und Lebensgefahr zu vermeiden.</p>
---	---

2 Sicherheitsanweisungen

2.1 Informationen zur EMV



Gefahr von Interferenzen mit Radio- oder Fernsehgeräten bei Einsatz in Büro- oder Wohnbereich! Das Produkt ist ein Gerät der Klasse B.

Ausschließlich beiliegendes Zubehör oder HMS-Zubehör, bestimmt für die Verwendung mit dem Gerät, verwenden. Ausschließlich abgeschirmte Kabel verwenden.

Sicherstellen, dass Schirm der Schnittstelle auf Gerätesteckern und Gegenstelle aufliegt.

2.2 Allgemeine Sicherheitshinweise

- ▶ Produkt vor Nässe und Feuchtigkeit schützen.
- ▶ Produkt vor zu heißer oder kalter Temperatur schützen (siehe *Technische Daten, S. 11*).
- ▶ Produkt vor offenen Flammen und Feuer schützen.
- ▶ Produkt nicht lackieren oder bemalen.
- ▶ Produkt nicht modifizieren oder auseinanderbauen. Service ausschließlich durch HMS Industrial Networks durchführen lassen.
- ▶ Produkt staubfrei und trocken lagern.

2.3 Bestimmungsgemäße Verwendung

Das Gerät wird verwendet, um Computersysteme an CAN-Netzwerke anzubinden, um Daten auszutauschen zum Beispiel, um ein Gerät über CAN zu konfigurieren oder um Diagnosedaten zu lesen. Das simplyCAN ist bestimmt für den Anschluss an einen Computer über eine USB-Schnittstelle.

3 Lieferumfang

Im Lieferumfang enthalten:

- simplyCAN Gerät

Folgendes kann von www.simplycan.info heruntergeladen werden:

- simplyCAN Busmonitor
- Installationsdatei *setup.bat* (notwendig für Windows 7)
- Programmier-API
- Programmierbeispiele
- Benutzerhandbuch

CAN-Bus-Abschluss kann separat bestellt werden.

4 Produktbeschreibung

Das simplyCAN ist ein aktiver USB-Adapter, der es dem User ermöglicht einen Computer mit einem CAN-Netzwerk zu verbinden, um Netzwerk-Traffic zu überwachen und mit anderen Netzwerk-Geräten zu interagieren. Das simplyCAN ist durch die einfache Installation und der einfach zu verwendenden CAN-Programmierschnittstelle ein Plug&Play-Gerät.

Funktionen

- USB 1.1 Full-Speed (12 MBit/s)
- 1 x CAN-High-Speed-Kanal entsprechend ISO 11898-2
- D-Sub-9 Feldbusanschluss, Pinbelegung entsprechend CiA 303-1
- USB-Kabel mit Steckertyp A



Windows: Das simplyCAN ist getestet mit Windows 7 (32 Bit und 64 Bit) und Windows 10 (64 Bit).

Linux: Das simplyCAN ist getestet mit Ubuntu 14.04 mit Linux Kernel-Version 4.4 und Ubuntu 18.04 mit Linux Kernel-Version 4.15.



Um auf die USB-Schnittstelle zuzugreifen, sind möglicherweise Administratorrechte notwendig.

5 Installation



Unzureichende Spannungsversorgung!

Gerät direkt mit Computer oder über Hubs mit externer Spannungsversorgung anschließen, um ausreichende Spannungsversorgung sicherzustellen. Verlängerungskabel können Verbindungsprobleme verursachen.



USB-Schnittstelle ist Hot-Plug fähig!

Es ist möglich das Gerät während des Betriebs einzustecken und auszustecken.

Bei Windows 10 und Linux wird die USB-Schnittstelle beim Einstecken automatisch installiert, ohne Installation eines Treibers.

- ▶ simplyCAN Zip-Datei von www.simplycan.info herunterladen und Datei entpacken.
- ▶ Bei Windows 7 auf *setup.bat* doppelklicken, um Treiber für Windows 7 zu installieren.



*Bei Windows 10 wird die COM-Schnittstelle für das simplyCAN nach Einstecken als **USB Serial Device (COMx)** im Gerätemanager angezeigt. Um das simplyCAN als **lxxat simplyCAN (COMx)** im Gerätemanager anzuzeigen, auch bei Windows 10 die Datei *setup.bat* ausführen.*

- ▶ USB-Anschluss in USB-Steckplatz des Computers stecken.
 - Hardware wird automatisch erkannt und installiert.
 - USB LED blinkt grün.
- ▶ Falls notwendig, Busabschlusswiderstand installieren (siehe [CAN-Bus-Abschluss](#), S. 10).
- ▶ CAN-Feldbusanschluss mit CAN-Feldbus verbinden.
- ▶ simplyCAN Busmonitor starten (siehe [Betrieb](#), S. 8).

Anschlüsse

Der Schirm des USB-Kabels ist über einen 100 nF Kondensator mit der Masse verbunden. Der Schirm des CAN-Anschlusses ist über einen 1 MΩ Widerstand und einen 10 nF Kondensator mit der Masse der CAN-Ankopplung verbunden. USB_shield ist über einen 4,7 nF Kondensator mit CAN_shield verbunden.



Für höchste Störfestigkeit die Schirme der CAN-Kabel direkt mit Gerätemasse verbinden.

Pinbelegung D-Sub-9

Signal	Pin Nr.
CAN-High	7
CAN-Low	2
CAN-GND	3, 6

6 Betrieb

6.1 simplyCAN Busmonitor

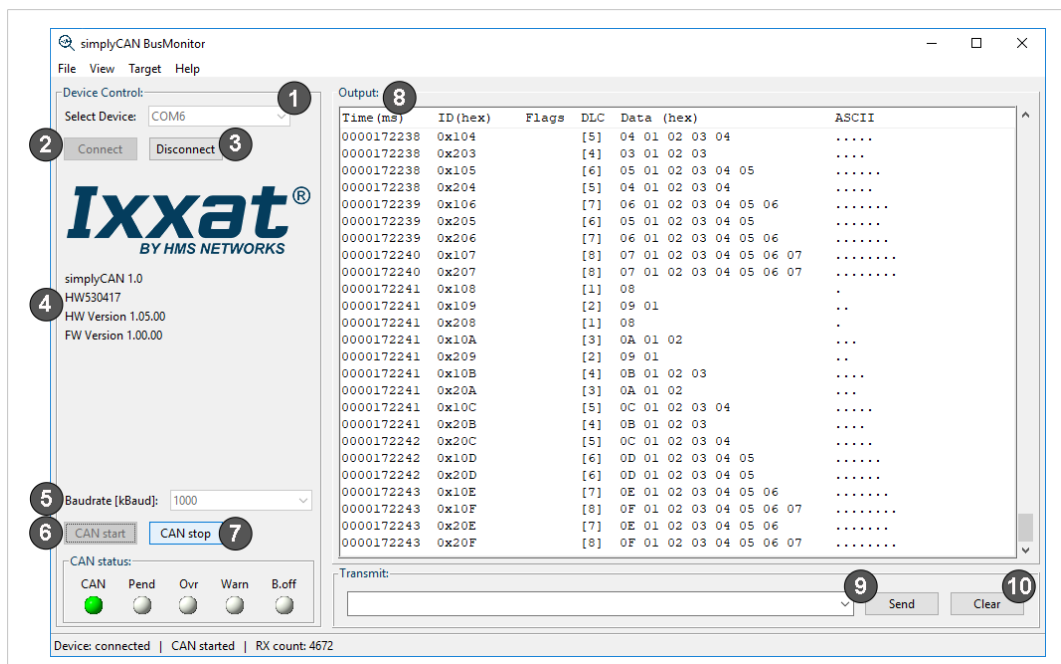


Fig. 1 simplyCAN Busmonitor

- ▶ simplyCAN Busmonitor starten.
 - Wenn ein simplyCAN mit dem Computer verbunden ist, wird das Gerät automatisch gewählt und verbunden (1).
- ▶ Wenn mehrere simplyCAN mit dem Computer verbunden sind, gewünschtes Gerät wählen (1) und Button **Connect** (2) klicken.
 - Informationen zum Gerät werden angezeigt (4).



Der simplyCAN Busmonitor kann mehrmals geöffnet werden, um mehrere simplyCAN Geräte gleichzeitig zu verbinden.

- ▶ Um Gerät zu wechseln, Button **Disconnect** (3) klicken, Gerät in Auswahlliste **Select device** (1) wählen und Button **Connect** (2) klicken.
- ▶ Gewünschte CiA-Baudrate wählen (5).
- ▶ Um Kommunikation zu starten, Button **CAN start** (6) klicken.
 - CAN-Nachrichten werden im Fenster **Output** (8) angezeigt.
 - In gesendeten Nachrichten ist der Zeitstempel 0 und das Flag **S** ist angezeigt.
- ▶ Um Nachricht zu senden, Nachricht in Zeile **Transmit** (9) eingeben (siehe [Sende-Nachrichten](#), S. 9 für weitere Informationen).
- ▶ Button **Send** (9) klicken.
 - Wenn eingegebene Nachricht gültig ist, wird die Nachricht gesendet.
 - Wenn eingegebene Nachricht ungültig ist, wird Fehlnachricht **Syntax error** und eine Beschreibung des Nachrichtenformats angezeigt.
- ▶ Kommunikation mit Button **CAN stop** (7) stoppen.

- Output-Fenster mit Button **Clear (10)** leeren.



Bei einer Buslast über 65 % ist Datenverlust möglich. Datenverlust wird der Applikation durch die **Ovr** LED im simplyCAN Busmonitor signalisiert.

Sende-Nachrichten

Syntax: **<id> [R] [E] [<data>...]**

- **id**: Identifier (dezimal oder hexadezimal)
- **R**: Remote Transmit Request der Nachricht
- **E**: Nachricht in Extended Frame Format (29 Bit)
- **data**: Datenbytes der Nachricht (dezimal oder hexadezimal), in RTR-Nachrichten enthält erstes Datenbyte den DLC



Falls hexadezimale Werte verwendet werden, müssen diese mit 0x beginnen.

Beispiel: 256 in dez ist 0x100 in hex

Beispiele

Nachricht im simplyCAN Busmonitor	Beschreibung
0x100 0x11 0x22 0x3 0x44	11-Bit-Nachricht mit ID 100 (hex) und 4 Datenbytes
0x1FE1200 E 1 2 3 4 5 6 7 8	29-Bit-Nachricht mit ID 1FE1200 (hex) und 8 Datenbytes
123 R 8	11-Bit-Remote-Frame mit ID 123 und DLC=8
0x1FE1200 R 8	29-Bit-Remote-Frame mit ID 1FE1200 (hex) und DLC=8

6.2 USB LED

Die USB LED zeigt den Status der USB-Kommunikation.

LED-Status	Beschreibung	Bemerkungen
Aus	Ausgeschaltet	Keine Spannungsversorgung oder Gerät defekt
Grün blinkend	Keine aktive Verbindung	Gerät betriebsbereit, simplyCAN Busmonitor oder API müssen gestartet werden, um Gerät zu verwenden
Grün	Aktive Verbindung	Gerät in Betrieb

6.3 CAN LED

CAN LED zeigt den Status der CAN-Kommunikation.

LED-Status	Beschreibung	Bemerkungen
Aus	Keine Kommunikation	Keine Kommunikation, Gerät nicht mit CAN verbunden.
Grün blinkend	Kommunikation OK	LED wird mit jeder Nachricht getriggert.
Rot blinkend	Kommunikation mit Fehlern	Controller ist in Status <i>Error Warning</i> oder in Status <i>Error Passive</i> , Kommunikation ist möglich.
Rot	Bus Off	Controller ist in Status <i>Bus Off</i> , keine Kommunikation möglich.

7 Zusätzliche Komponenten

7.1 CAN-Bus-Abschluss

Im Gerät ist kein Bus-Abschlusswiderstand für den CAN-Bus vorhanden. HMS Industrial Networks bietet einen Bus-Abschlusswiderstand als Durchführungsstecker an.

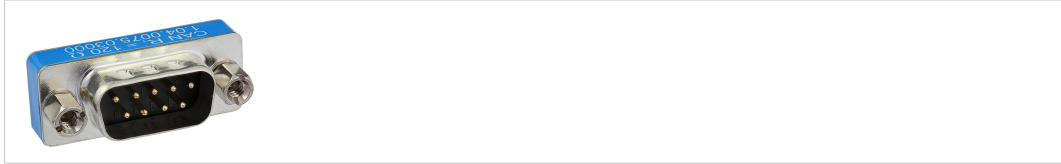


Fig. 2 CAN-Bus-Abschlusswiderstand

- Für Bestellinformationen siehe www.ixxat.com.

8 Technische Daten

USB-Schnittstelle	USB 1.1, Full-Speed (12 MBit/s)
CAN-Bitraten	10 kbit/s bis 1 Mbit/s, nur CiA-Bitraten sind unterstützt: 10, 20, 50, 125, 250, 500, 800, 1000
CAN-Transceiver	TI SN65HVD251
CAN-Bus-Abschluss	Keiner
Abmessungen	80 x 50 x 22 mm
Gewicht	Circa 100 g
Spannungsversorgung	Via USB, 5 V DC/100 mA
Galvanische Trennung	800 V DC/500 V AC für 1 min
Betriebstemperatur	-20 bis +70 °C
Lagerungstemperatur	-40 bis +85 °C
Relative Feuchtigkeit	10 % bis 95 %, keine Kondensation
Gehäusematerial	ABS Kunststoff
Schutzklasse	IP40

9 Fehlerbehebung

USB LED ist aus nach Verbinden.

Keine Spannungsversorgung oder Gerät defekt

- ▶ Sicherstellen, dass Gerät korrekt mit USB-Anschluss verbunden ist.
- ▶ Gerät direkt mit Computer oder über Hubs mit externer Spannungsversorgung anschließen.

Verlängerungskabel wird verwendet und Gerät funktioniert nicht.

Verlängerungskabel können Verbindungsprobleme verursachen.

- ▶ Verlängerungskabel entfernen.
- ▶ Gerät direkt mit Computer oder über USB-Hubs mit externer Spannungsversorgung anschließen.

10 Reinigung

- ▶ Gerät von Spannungsversorgung trennen.
- ▶ Schmutz mit weichem, chemisch unbehandeltem, trockenen Tuch entfernen.

11 Support/Hardware zurücksenden

Folgende Informationen im Support-Bereich auf www.ixxat.com beachten:

- Informationen zu Produkten
- FAQ-Listen
- Installationshinweise
- aktuelle Produktversionen
- Updates

11.1 Support

- ▶ Bei Problemen mit dem Produkt oder bei Support-Bedarf, auf www.ixxat.com/support Support anfragen.
- ▶ Wenn notwendig telefonische Support-Kontakte auf www.ixxat.com nutzen.

11.2 Hardware zurücksenden

- ▶ Formular für Gewährleistung und Reparaturen auf www.ixxat.com ausfüllen.
- ▶ RMA-Nummer (Return Material Authorization) ausdrucken.
- ▶ Produkt sorgfältig und ESD-geschützt verpacken, wenn möglich Originalverpackung verwenden.
- ▶ RMA-Nummer beilegen.
- ▶ Weitere Informationen auf www.ixxat.com beachten.
- ▶ Hardware zurücksenden.

12 Entsorgung

- ▶ Produkt entsprechend nationaler Gesetze und Vorschriften entsorgen.
- ▶ Weitere Hinweise zu Entsorgung von Produkten auf www.ixxat.com beachten.

13 API-Dokumentation

13.1 API-Funktionen



Um die Verwendung der Funktionen zu zeigen, sind Beispiele in C, C# und Python auf www.simplycan.info verfügbar.

13.1.1 simply_open

Öffnet die serielle Kommunikationsschnittstelle. Nachrichtenfilter des CAN-Controllers ist für alle Nachrichtenidentifizier geöffnet.

```
bool simply_open(char *serial_port);
```

Parameter

Parameter	Dir.	Beschreibung
<i>serial_port</i>	[in]	Name des seriellen Kommunikationsports (z. B. <i>COM1</i> oder <i>/dev/ttyACM0</i>). simplyCAN Busmonitor verwenden, um zu erkennen mit welchem seriellen COM-Port das simplyCAN verbunden ist. Mit Windows ist es auch möglich den Gerätemanager und mit Linux den Befehl <code>ls -l /dev/serial/by-id</code> zu verwenden.

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.2 simply_close

Schließt die serielle Kommunikation und setzt den CAN-Controller zurück.

```
bool simply_close(void);
```

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.3 simply_initialize_can

Initialisiert den CAN-Controller.

```
bool simply_initialize_can(uint16_t bitrate);
```

Parameter

Parameter	Dir.	Beschreibung
<i>bitrate</i>	[in]	CAN-Bitrate als Ganzzahl (integer), mögliche Werte: 10, 20, 50, 125, 250, 500, 800, 1000

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.4 simply_identify

Liefert Firmware- und Hardware-Information des simplyCAN.

```
bool simply_identify(identification_t *p_identification);
```

Parameter

Parameter	Dir.	Beschreibung
<i>p_identification</i>	[out]	Pointer zur Identification Structure

Identification Structure

```
typedef struct _identification {
    uint8_t fw_version[8];
        // Zero terminated firmware version string e.g. "1.00.00"
    uint8_t hw_version[8];
        // Zero terminated hardware version string e.g. "1.00.00"
    uint8_t product_version[8];
        // Zero terminated product version string e.g. "1.00.00"
    uint8_t product_string[30];
        // Zero terminated product string e.g. "simplyCAN 1.0"
    uint8_t serial_number[9];
        // Zero terminated serial number e.g. "HW123456"
} identification_t;
```

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.5 simply_start_can

Startet den CAN-Controller. Setzt den CAN-Controller in Modus Running und leert die CAN-Nachrichten-FIFOs. In Modus Running können CAN-Nachrichten gesendet und empfangen werden.

```
bool simply_start_can(void);
```

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.6 `simply_stop_can`

Stoppt den CAN-Controller. Setzt den CAN-Controller in Modus *init*. Setzt den Nachrichtenfilter des CAN-Controllers nicht zurück. CAN-Controller nur stoppen wenn Flag `CAN_STATUS_PENDING` nicht gesetzt ist.

```
bool simply_stop_can(void);
```

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

Bemerkungen

Um sicherzustellen, dass alle Nachrichten gesendet sind bevor der CAN-Controller gestoppt wird, CAN-Status lesen bis Flag `CAN_STATUS_PENDING` nicht mehr gesetzt ist.

13.1.7 `simply_reset_can`

Setzt den CAN-Controller zurück (Hardware-Reset) und leert den Nachrichtenfilter (offen für alle Nachrichtenidentifizier). Setzt den CAN-Controller in Modus *init*.

```
bool simply_reset_can(void);
```

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.8 simply_can_status

Liefert den Status des CAN-Controllers.

```
bool simply_can_status(can_sts_t *can_sts);
```

Parameter

Parameter	Dir.	Beschreibung
<i>can_sts</i>	[out]	Status als Bit-kodierter Wert (siehe CAN Status Structure)

CAN Status Structure

```
typedef struct _can_sts {  
    uint16_t sts;  
        // bit coded status flags (see CAN status definitions)  
    uint16_t tx_free;  
        // number of free elements in CAN message tx fifo  
} can_sts_t;
```

CAN Status Definitionen

```
/* CAN status definitions */  
#define CAN_STATUS_RUNNING          (0x01)  
#define CAN_STATUS_RESET           (0x02)  
#define CAN_STATUS_BUSOFF          (0x04)  
#define CAN_STATUS_ERRORSTATUS     (0x08)  
#define CAN_STATUS_RXOVERRUN       (0x10)  
#define CAN_STATUS_TXOVERRUN       (0x20)  
#define CAN_STATUS_PENDING         (0x40)
```

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.9 simply_set_filter

Setzt den 11- oder 29-Bit-Nachrichtenfilter des CAN-Controllers. Um den 29-Bit-Nachrichtenfilter zu setzen, muss das MSB im Parameter value gesetzt sein.

```
bool simply_set_filter(uint32_t mask, uint32_t value);
```

Parameter

Parameter	Dir.	Beschreibung
mask	[in]	11- oder 29-Bit CAN-Nachrichten-Identifizier Mask
value	[in]	11- oder 29-Bit CAN-Nachrichten-Identifizier Value, MSB setzen um 29-Bit-Nachrichtenfilter zu setzen

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, simply_get_last_error für weitere Informationen aufrufen.

Bemerkung

Mit dem Mask/Value-Filter (für 11-Bit und 29-Bit-Identifizier) können mögliche gültige Identifizier basierend auf Bitmasken definiert werden.

Binäre Darstellung von Mask:

- Binäre Positionen mit Wert 1 sind relevant für den Filter.
- Binäre Positionen mit Wert 0 sind nicht relevant für den Filter.

Binäre Darstellung von Value:

- Definiert die Werte für die Positionen, die in Mask als relevant (1) markiert sind.
- Werte in Positionen, die als nicht relevant (0) markiert sind, werden ignoriert.

Die folgende Formel zeigt die Voraussetzung unter der ein Identifizier den Filter passiert.

- if (value & mask) == (identifizier & mask) dann ist Identifizier gültig

Beispiel 11-Bit-Identifizier

	hex	bin
Value	0x700	0111:0000:0000
Mask	0x700	0111:0000:0000
Ergebnis	0x700	0111:0000:0000
Jeder Identifizier zwischen 0x700 und 0x7FF passiert den Filter, da nur die ersten 3 Bit der Mask als relevant markiert sind.		

Beispiel 29-Bit-Identifizier

	hex	bin
Value	0x90003344	1001:0000:0000:0000:0011:0011:0100:0100
Mask	0x1F00FFFF	0001:1111:0000:0000:1111:1111:1111:1111
Ergebnis	0x10003344	0001:0000:0000:0000:0011:0011:0100:0100
256 Identifizier zwischen 0x10003344 und 0x10FF3344 passieren den Filter, bei welchen die letzten zwei Bytes 0x3344 sind.		

Um 29-Bit-Nachrichten zu erlauben den Filter zu passieren, muss das MSB im Parameter value gesetzt sein.

Weitere Beispiele

Value	Mask	Gültige Nachrichten, die den Filter passieren
0x100	0x7FF	0x100
0x100	0x700	0x100–0x1FF
0x000	0x000	0x000–0x7FF

13.1.10 simply_receive

Empfängt eine einzelne CAN-Nachricht.

```
int8_t simply_receive(can_msg_t *can_msg);
```

Parameter

Parameter	Dir.	Beschreibung
<i>can_msg</i>	[out]	Pointer zur CAN Message Structure in welche die empfangene CAN-Nachricht kopiert wird

CAN Message Structure

```
typedef struct _can_msg {  
    uint32_t timestamp;           // in milliseconds  
    uint32_t ident;              // MSB=1: extended frame  
    uint8_t dlc;                 // MSB=1: remote frame  
    uint8_t payload[8];  
} can_msg_t;
```

Rückgabewert

Rückgabewert	Beschreibung
1	Nachricht empfangen
0	Keine Nachricht in Empfangsqueue verfügbar
-1	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

13.1.11 simply_send

Schreibt eine CAN-Nachricht in den Sende-FIFO. Um zu prüfen, ob Nachricht gesendet ist, CAN-Status mit [simply_can_status](#) abfragen.

```
bool simply_send(can_msg_t *can_msg);
```

Parameter

Parameter	Dir.	Beschreibung
<i>can_msg</i>	[in]	Pointer zur zu sendenden CAN-Nachricht (siehe CAN Message Structure)

Rückgabewert

Rückgabewert	Beschreibung
true	Erfolgreiche Ausführung
false	Fehler aufgetreten, <code>simply_get_last_error</code> für weitere Informationen aufrufen.

Bemerkung

Mit `simply_send` werden CAN-Nachrichten nicht sofort gesendet, sondern in den Sende-FIFO geschrieben. Mit [simply_can_status](#) prüfen, ob Nachricht gesendet ist. Wenn Flag `CAN_STATUS_PENDING` gesetzt ist, ist die Nachricht noch nicht gesendet. CAN-Status abfragen bis Flag `CAN_STATUS_PENDING` nicht mehr gesetzt ist.

13.1.12 `simply_get_last_error`

Liefert den letzten Fehlercode. Nach Lesen des Fehlercodes mit `simply_get_last_error` wird der Fehlercode auf 0 gesetzt. Jeder Fehler kann nur einmal gelesen werden.

```
int16_t simply_get_last_error(void);
```

Rückgabewert

Rückgabewert	Fehler	Beschreibung
0	<code>SIMPLY_S_NO_ERROR</code>	Kein Fehler
-1	<code>SIMPLY_E_SERIAL_OPEN</code>	Serieller Port kann nicht geöffnet werden.
-2	<code>SIMPLY_E_SERIAL_ACCESS</code>	Zugriff auf seriellen Port verweigert.
-3	<code>SIMPLY_E_SERIAL_CLOSED</code>	Serieller Kommunikationsport geschlossen.
-4	<code>SIMPLY_E_SERIAL_COMM</code>	Fehler serielle Kommunikation
-5	<code>SIMPLY_E_CMND_REQ_UNKNOWN</code>	Befehl auf Gerät unbekannt
-6	<code>SIMPLY_E_CMND_RESP_TIMEOUT</code>	Timeout für Befehl-Rückgabe erreicht
-7	<code>SIMPLY_E_CMND_RESP_UNEXPECTED</code>	Unerwartete Befehl-Rückgabe empfangen
-8	<code>SIMPLY_E_CMND_RESP_ERROR</code>	Fehler Befehl-Rückgabe
-9	<code>SIMPLY_E_INVALID_PROTOCOL_VERSION</code>	Ungültige Version simplyCAN-Protokoll
-10	<code>SIMPLY_E_INVALID_FW_VERSION</code>	Ungültige Version Geräte-Firmware
-11	<code>SIMPLY_E_INVALID_PRODUCT_STRING</code>	Ungültiger simplyCAN Product String
-12	<code>SIMPLY_E_CAN_INVALID_STATE</code>	Ungültiger CAN-Status
-13	<code>SIMPLY_E_CAN_INVALID_BAUDRATE</code>	Ungültige CAN-Baudrate
-14	<code>SIMPLY_E_TX_BUSY</code>	Nachricht nicht gesendet, Tx busy
-15	<code>SIMPLY_E_API_BUSY</code>	API ist busy.

13.2 Status-Diagramm

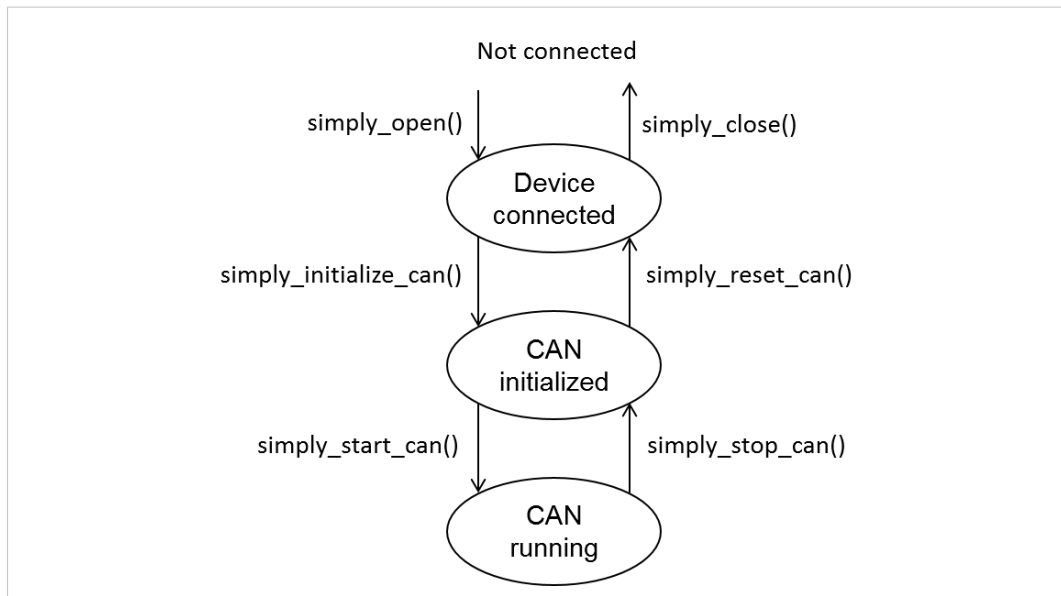


Fig. 3 simplyCAN Status

Funktionsaufrufe und zugehörige gültige Status

Funktion	Gültige Status
<code>simply_open()</code>	Nicht verbunden
<code>simply_close()</code>	Gerät verbunden, CAN initialisiert
<code>simply_initialize_can()</code>	Gerät verbunden, CAN initialisiert
<code>simply_reset_can()</code>	Gerät verbunden, CAN initialisiert, CAN running
<code>simply_start_can()</code>	CAN initialisiert
<code>simply_stop_can()</code>	Gerät verbunden, CAN initialisiert, CAN running
<code>simply_receive()</code>	CAN initialisiert, CAN running
<code>simply_send()</code>	CAN running
<code>simply_get_last_error()</code>	Alle Status
<code>simply_can_status()</code>	Gerät verbunden, CAN initialisiert, CAN running
<code>simply_identify()</code>	Gerät verbunden, CAN initialisiert, CAN running
<code>simply_set_filter()</code>	CAN initialisiert

A Konformitätserklärungen

A.1 EMV Konformitätserklärung (CE)



Dieses Produkt entspricht der EG Richtlinie über die elektromagnetische Verträglichkeit. Weitere Informationen und die Konformitätserklärung finden Sie unter www.ixxat.com.

A.2 Entsorgung und Recycling



Sie müssen dieses Produkt ordnungsgemäß entsprechend lokaler Gesetze und Richtlinien entsorgen. Weil dieses Produkt elektronische Komponenten enthält, muss es getrennt von Haushaltsmüll entsorgt werden. Bei Altprodukten kontaktieren Sie lokale Behörden, um über Entsorgungs- und Recyclingmöglichkeiten informiert zu werden, oder geben Sie es einfach bei ihrem lokalen HMS-Geschäft ab, oder senden Sie es an HMS zurück.

Für weitere Informationen siehe www.hms-networks.com.

