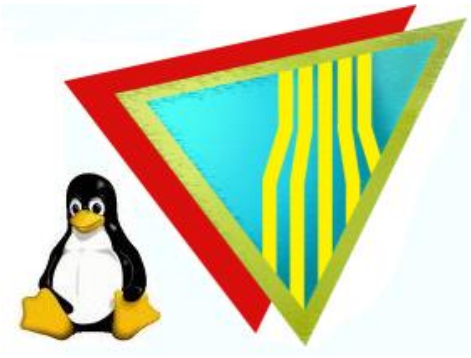
Dokumentation
CanFox API



Author: **Andreas Herbertz**

Erstellt: **18.09.2012**

Sontheim Industrie Elektronik GmbH
Georg-Krug-Straße 2, D-87437 Kempten
Telefon: (+49) 0831.575900-0
Fax: (+49) 0831.575900-72
Email: info@s-i-e.de

Index

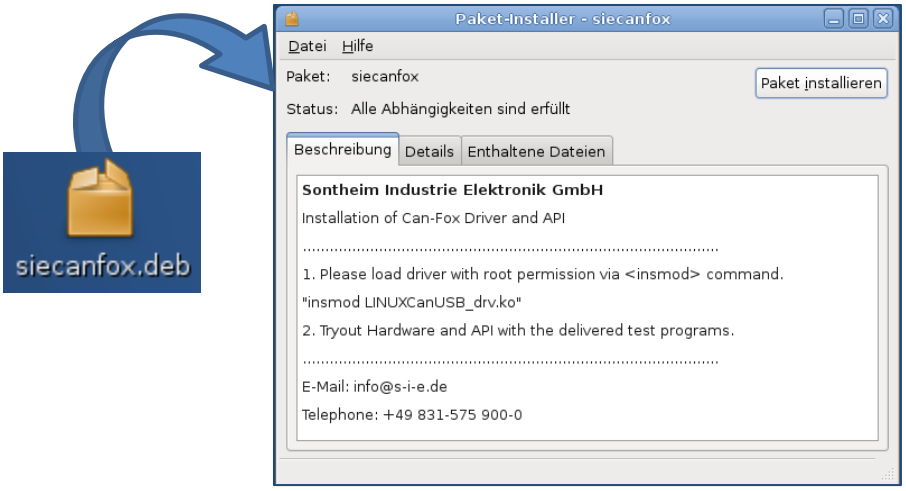
1. Einleitung

Dieses Dokument soll kurz die wesentlichen Grundlagen zum CanFox und dessen API erläutern. Des Weiteren wird die Installation des Installationspaketes beschrieben, so wie die Nutzung des Testprogrammes und der API.

2. Debian Paket Installation

Debian Pakete (*.deb) dienen als Softwareinstallation auf Debian-basierten Systemen. Sie beinhalten die zu installierende Software in komprimierter Form.

Die Pakete können über die GUI mit dem Debian-Installer (***Gdebi***) installiert werden oder über ein Kommando in der Konsole.



(Gdebi – zum installieren von Paketen)

Die Konsole kann über folgendes Kommando das Paket installieren.

```
dpkg –i siecanfox.deb
```

3. RPM Paket Installation

Falls Sie eine RedHat-Distribution von Linux nutzen können Sie das mitgelieferte RPM-Paket für die Installation nutzen.

Die Konsole kann über folgendes Kommando das Paket installieren.

```
rpm –i siecanfox-0.0.1-5.i386.rpm
```

4. Treiber laden

Der Treiber für den CanFox muss separat in den Kernel geladen werden.

Den Treiber finden Sie im Verzeichnis ***/usr/share/siecanfox/driver*** .

Hierzu benötigen Sie root-Rechte um mit folgendem Kommando den Treiber zu laden.

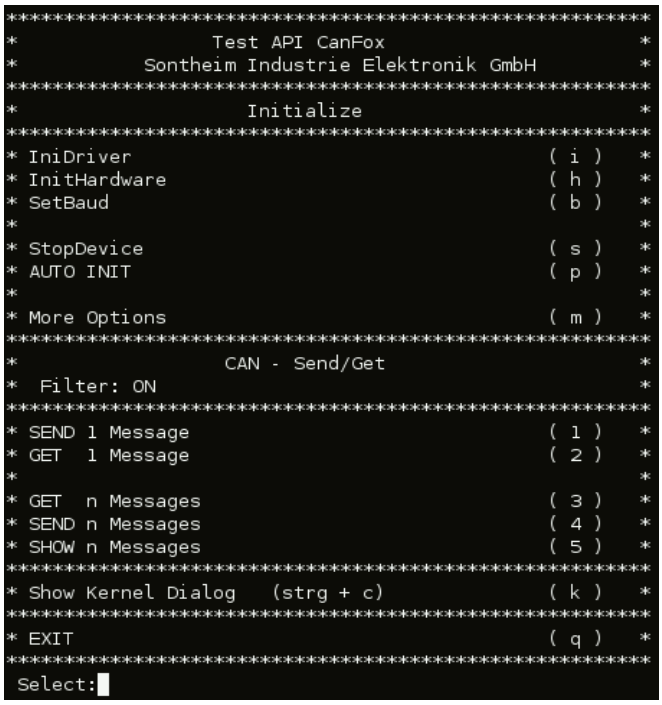
```
insmod LINUXCanUSB_Drv.ko
depmod -a
```

5. Testprogramm

Das Paket (***siecanfox.deb***) beinhaltet das Testprogramm „**SIE_CanFox_Test**“ um die Hardware (CanFox) und die ebenfalls mitgelieferte Shared Library (***libSIE_CANMTAPI.so***) auf deren Funktionalitäten zu testen.

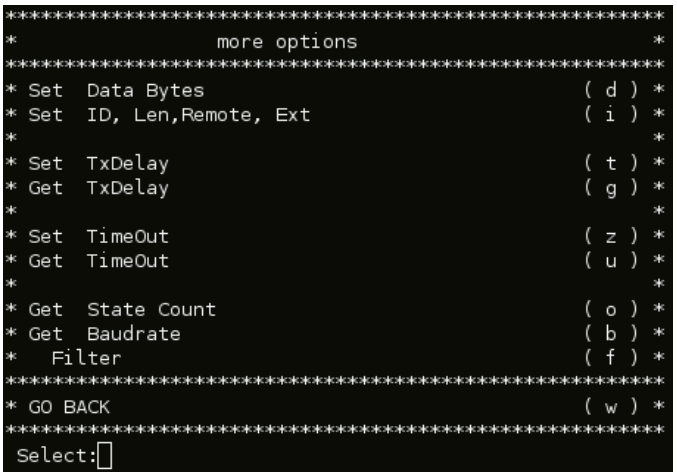
Sie finden das Testprogramm im Verzeichnis ***/usr/share/siecanfox/test_application*** .

Bevor Sie das Testprogramm starten muss der Treiber geladen werden!



(Testprogramm - SIE_CanFox_Test)

Auswahl	Name	Beschreibung	Aufruf API
i	***InitDriver***	Laden der Shared Library, Treiber initialisieren	*canStartAPI()*
h	***InitHardware***	Firmware starten, Speicher allokieren	*canOpen()*
b	***SetBaud***	Channel öffnen, Baudrate setzten	*canSetBaudrate()*
s	***StopDevice***	Channel schließen	*canClose()*
p	***AUTO INIT***	Initialisiert automatisch CanFox mit Baudrate 250 kbits.	*canStartAPI()* *+ canOpen()* *+ canSetBaudrate(2)*
m	***More Options***	Weitere Konfigurationen anzeigen.	
1	***SEND 1 Message***	Sendet eine einzelne Nachricht.	*canWrite()*
2	***GET 1 Message***	Empfängt eine einzelne Nachricht.	*canRead()*
3	***GET n Messages***	Empfängen und speichern mehrerer Nachrichten.	*canRead()*
4	***SEND n Messages***	Sendet 5 Nachrichten.	*canWrite()*
5	***SHOW n Messages***	Zeigt die empfangenen Nachrichten.	
k	***Show Kernel D…***	Zeigt Kernelkommunikation, Treiber aufrufe, etc. (Beenden der Anzeige : *strg + c*)	
q	***EXIT***	Beendet das Testprogramm.	



(*weitere Optionen des Testprogramms – SIE_CanFox_Test*)

Auswahl	Name	Beschreibung	Aufruf API
d	***Set Data Bytes***	Daten Bytes mit Werten belegen.	
i	***Set ID, Len, Rem…***	ID, Datenlänge, etc. festlegen.	
t	***Set TxDelay***	TxDelay konfigurieren.	*canSetTxDelay()*
g	***Get TxDelay***	TxDelay anzeigen.	*canGetTxDelay()*
z	***Set TimeOut***	Timeout für canRead festlegen.	*canSetTimeout()*
u	***Get TimeOut***	Aktuelles Timeout anzeigen.	*canGetTimeout()*
o	***Get State Count***	Abfragen der Counter. (z.B. TxMsgCounter)	*canGetStateCounter()*
b	***Get Baudrate***	Aktuelle Baudrate anzeigen.	*canGetBaudrate()*
f	***Filter***	Eine ID als Filter setzten.	
w	***GO BACK***	In das Hauptmenü zurückkehren.	

6. API

Um auf die Funktionen der API zugreifen zu können fügen Sie Ihrem Programm die Header-Datei „CanFox_API.h“ bei.

```
#include “CanFox_API.h“                    // Zugriff auf API
…
```

Sie finden die Header-Datei im Verzeichnis ***/usr/share/siecanfox/include*** .

7. Funktionen

7.1. canAPIStart

Funktion	**canAPIStart**
Beschreibung	Initialisiert die Funktionen der Shared Library, legt die Gerätedatei an mit welcher der Treiber seine Aufrufe behandelt. Desweiteren wird auch versucht den Treiber in den Kernel zu laden.

Syntax	SIE_int32	**canAPIStart** (void);
Parameter	void	Benötigt keine Parameter.
Return	SIE_int32	Liefert SUCCESS zurück wenn Shared Library geöffnet werden konnte.

7.2. canOpen

Funktion	**canOpen**
Beschreibung	Initialisiert die Hardware mit der Firmware und allokiert Speicher für den CanFox.

Syntax	SIE_int32	**canOpen** (void);
Parameter	void	Benötigt keine Parameter.
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.3. canSetBaudrate

Funktion	**canSetBaudrate**
Beschreibung	Öffnet den Channel zum CanFox und legt die Baudrate fest. Zuvor müssen die Funktionen canAPIStart und canOpen aufgerufen werden.

Syntax	SIE_int32	**canSetBaudrate** (SIE_uint32 Baud);
Parameter	SIE_uint32	Wert der die Baudrate festlegt. (1 bis 4)
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.4. canGetBaudrate

Funktion	**canGetBaudrate**
Beschreibung	Liefert aktuell eingestellte Baudrate zurück welche auf dem CanFox eingestellt wurde.

Syntax	SIE_int32	**canGetBaudrate** (SIE_uint32* Baud);
Parameter	SIE_uint32	Pointer welcher den Wert der aktuellen Baudrate erhält.
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.5. canWrite

Funktion	**canWrite**
Beschreibung	Sendet eine Can-Nachricht an den CanFox. Dazu muss ein Objekt von TCMSG erzeugt werden und mit entsprechenden Werten initialisiert werden. Ein weiterer Pointer von SIE_uint32 legt die Anzahl der zu sendenden Nachrichten fest.

Syntax	SIE_int32	**canWrite** (TCMSG* buffer SIE_uint32* msgcount);
Parameter	TCMSG*	Zu sendende Nachricht.
	SIE_uint32*	Anzahl der zu sendenden Nachrichten .
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.6. canRead

Funktion	**canRead**
Beschreibung	Funktion die eine Can-Nachricht speichert die vom CanFox empfangen wurde.

Syntax	SIE_int32	**canRead** (TCMSG* msg);
Parameter	TCMSG*	Pointer der eine Nachricht abspeichert.
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.7. canClose

Funktion	**canClose**
Beschreibung	Funktion beendet den Channel zum CanFox. CAN-Led off.

Syntax	SIE_int32	**canClose** (void);
Parameter	void	Benötigt keine Parameter.
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.8. canGetStateCounter

Funktion	**canGetStateCounter**
Beschreibung	Funktion liefert die Counter-Werte zurück. (Bsp. TxMsgCounter, RxByteCounter, FrameCounter, ErrorState, etc.)

Syntax	SIE_int32	**canGetStateCounter** (T_COUNTER* Counter);
Parameter	T_COUNTER	Pointer erhält die aktuellen Werte.
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.1. canSetTimeout

Funktion	**canSetTimeout**
Beschreibung	Festlegen des Timeouts. Nach erreichen des Timeouts wird die Funktion canRead beendet.

Syntax	SIE_int32	**canSetTimeout** (SIE_int32 Timeout);
Parameter	SIE_int32	Wert für Timeout. (Wert = ca. sec.)
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

7.1. canGetTimeout

Funktion	**canGetTimeout**
Beschreibung	Gibt den Wert des Timeouts zurück.

Syntax	SIE_int32	**canGetTimeout** (SIE_int32* Timeout);
Parameter	SIE_int32*	Pointer erhält den aktuellen Timeout Wert.
Return	SIE_int32	Liefert SUCCESS bei Erfolg zurück. Sonst NO_SUCCESS.

8. Baudraten

Folgende Baudraten werden vom CanFox unterstützt und werden über folgende Parameter konfiguriert.

Baudrate	***Parameter***
100 kbits	1
250 kbits	2
500 kbits	3
1.000 kbits	4

9. T_CMSG

Stuktur für Can-Nachrichten.

T_CMSG		
SIE_int32	***i32Id***	ID
unsigned char	***byLen***	Anzahl Daten Bytes
unsigned char	***byMsgLost***	Message Lost Flag
unsigned char	***byExtended***	Extended Flag
unsigned char	***byRemote***	Remote Flag
unsigned char	***abyData[0..7]***	max. 8 Daten Bytes
SIE_uint32	***ui32Tstamp***	Time Stamp

10. Beispiele

Die folgenden Beispiele sollen veranschaulichen wie die Funktionen der API aufgerufen werden können.

10.1. Beispiel: canOpen

```
iRet = canOpen();
```

10.2. Beispiel: canSetBaudrate

```
iRet = canSetBaudrate(2);                    // Baudrate 250 kbits
```

10.3. Beispiel: canWrite

```
SIE_uint32 msgcount = 1;
T_CMSG Buffer[100];
Buffer[0].i32Id = 45;                        // Initialisierung
Buffer[0].abyData[0] = 'A';
…
iRet = canWrite(Buffer, msgcount);           // Senden der Nachricht
```

10.4. Beispiel: canRead

```
T_CMSG  msg = NULL;
iRet = canRead(&msg);                  // Nachricht empfangen

Buffer[10].i32Id        = (SIE_int32) msg->i32Id;          // Kopieren der Nachricht
…
Buffer[10].abyData[7] = (unsigned char) msg->abyData[7];
```

10.5. Beispiel: T_CMSG

Initialisierung einer Can-Nachricht.

```
T_CMSG Buffer[100];
Buffer[0].i32Id              =  34;
Buffer[0].byLen              =   8;
Buffer[0].byMsgLost          =   0;
Buffer[0].byExtended         =   0;
Buffer[0].byRemote           =   0;
Buffer[0].abyData[0]         = 50;
…
Buffer[0].abyData[7]         = 50;
Buffer[0].ui32Tstamp         =   0;
```

11. Error-Codes

Return value = 0 : success
Return value > 0 : error

Error	Value	Description
SUCCESS	*0*	everything is ok
NO_SUCCESS	*0xFFFF*	something went wrong
RX_TIMEOUT	*-1*	timeout while reading
TX_TIMEOUT	*-2*	timeout while writing
NOT_AVAILABLE	*-3*	this functionality is not available
INVALID_PARAMETER	*-7*	invalid parameter
INVALID_HANDLE	*-8*	invalid handle
TOO_MANY_HANDLES	*-9*	to many handles
INIT_ERROR	*-10*	error while initialization
CHANNEL_NOT_INITIALIZED	*-14*	channel is not initialized -> set baudrate first
TOO_MANY_J2534_RANGES	*-19*	too many J2534 ranges set
TOO_MANY_J2534_2_FILTER	*-20*	too many J2534_2 filter set
NO_OWNER_RIGHTS	*-22*	no owner rights -> can't change
ALL_HANDLES_USED	*-30*	all handles are in use
GET_MUTEX_FAILED	*-104*	getting mutex failed
SETBAUDRATE_TIMEOUT	*-107*	timeout while setting baudrate
ARBITRATION_TIMEOUT	*-108*	timeout while getting arbitration
API_NOT_RUNNING	*-111*	API current not running
BUFFER_FULL	*-113*	buffer is full
NO_HARDWARE_FOUND	*-299*	hardware not present in system
CHANNEL_NOT_AVAILABLE	*-300*	this channel is not available
PCI_ERROR	*-400*	error pending to pci
BASE_INTERFACE	*-1000*	referenced the wrong function (basefunction)
END_OF_FILE	*500*	end of file or a stream
NOT_SUPPORTED	*600*	no more free space on disk to perform
CREATE_NODE_FAILED	*-1001*	create node on Linux failed
LOAD_MODULE_FAILED	*-1002*	load driver module failed
DEVICE_IS_OPEN	*-1003*	device is currently open
DEVICE_IS_NOT_OPEN	*-1004*	device is currently not open

12. Verzeichnisse und Dateien

*/usr/share/**siecanfox**/*
*/**documents**/*
*/documents/**Sontheim_CanFox_Documentation_eng.pdf***
*/documents/**Sontheim_CanFox_Dokumentation_de.pdf***
*/**driver**/*
*/driver/**LINUXCanUSB_Drv.ko***
*/**include**/*
*/include/**CanFox_MTAPI.h***
*/include/**HwStructs.h***
*/include/**SIE_Errors.h***
*/include/**SIE_Structs.h***
*/include/**SIE_Typedefs.h***
*/**test_application**/*
*/test_application/**Makefile***
*/test_application/**SIE_CanFox_Test***
*/test_application/**SIE_CanFox_Test.c***
*/test_application/**SIE_CanFox_Test.o***

*/usr/lib/**libSIE_CANMTAPI.so***

*/usr/include/**SIE_CanFox**/*
*/**CanFox_MTAPI.h***
*/**HwStructs.h***
*/**SIE_Errors.h***
*/**SIE_Structs.h***
*/**SIE_Typedefs.h***