CAN-Kommunikation mit PiXtend

- Vorbereitungen für CAN unter Linux -

- Hinzufügen eines CODESYS CANOpen Masters -

- Hinzufügen eines CODESYS CANOpen Slaves -

APP-PX-530

Stand 27.09.2017, V1.06

Qube Solutions UG (haftungsbeschränkt)
Arbachtalstr. 6, 72800 Eningen, Germany
http://www.qube-solutions.de/
http://www.pixtend.de

Versionshistorie

Version	Beschreibung	Bearbeiter
1.00	Dokument erstellt	CS
1.01	Dokument überarbeitet	CS
1.02	QS Firmenanschrift aktualisiert Informationen zur Linux-Konfiguration überarbeitet und mit wichtigen Hinweisen versehen. Rechtschreibung und Formatierung überarbeitet.	TG
1.03	Hinweis wegen neuem Namen des CAN-Overlays bei den neusten Raspbian Jessie Versionen (Linux Kernel 4.4) hinzugehügt	TG
1.05	Zeilen in /boot/config.txt überarbeitet und Hinweis auf das notwendige Leerzeichen eingefügt	TG
1.06	Dtoverlay Angaben laut den neuen Vorgaben der Raspberry Pi Foundation angepasst	TG

Inhaltsverzeichnis

1. Einleitung

Der CAN-Bus stammt ursprünglich aus dem Automobilbereich (entwickelt von der Firma Bosch). Heute wird der robuste und zuverlässige Bus auch in vielen anderen Bereichen, z.B. der industriellen Automation, eingesetzt. In der Automatisierungstechnik wird oft das CANopen Protokoll (OSI-Schicht 7 – Anwendungsschicht) verwendet, welches erfreulicherweise auch von CODESYS unterstützt und für den Raspberry Pi verfügbar ist. Es lohnt sich also den CAN-Bus und den CANopen-Stack genauer unter die Lupe zu nehmen.

Diese Anleitung beschreibt alle Schritte die notwendig sind um mit PiXtend eine CANOpen-Kommunikation unter CODESYS einzurichten.

Zunächst wird ein CAN-fähiges Linux Image vorbereitet und getestet.
Anschließend wird die CANopen-Konfiguration in CODESYS erklärt.

Diese Application-Note ist gleichermaßen für PiXtend V1.2 und V1.3 gültig.

Viele weitere Informationen, Tipps und Tricks finden Sie auch in unserem Support-Forum unter: http://www.pixtend.de/forum/

Sollten trotzdem Fragen offen bleiben, so bitten wir Sie uns per E-Mail (support@pixtend.de) in Kenntnis zu setzen. Sie erhalten schnellst möglich eine Antwort und weitere Informationen.

Die jeweils neuesten Versionen aller Dokumente und Software-Komponenten finden Sie im Download-Bereich unserer Homepage: http://www.pixtend.de/pixtend/downloads/

1.1 Voraussetzungen

Diese Anleitung setzt voraus dass Sie PiXtend bereits erfolgreich in Betrieb genommen haben und dass Sie die Grundlagen im Umgang mit Linux beherrschen, als auch mit der Bedienung von CODESYS vertraut sind.

Um die CAN-Kommunikation unter Linux zu testen muss PiXtend an einen CAN Bus mit mindestens einem aktiven CAN-Gerät angeschlossen werden. Es kann natürlich auch mit einem oder mehreren PiXtends verbunden werden.

Für die CODESYS OpenCAN Master/Slave Kommunikation werden zwei PiXtend-Boards verwendet.

1.2 Einschränkungen

Wenn der PiXtend CAN Controller verwendet wird, kann der PiXtend DAC nicht gleichzeitig verwendet werden. Die beiden Chips "teilen" sich einen SPI Chip-Select.

1.3 Haftungsausschluss

Weder Qube Solutions UG noch 3S-Smart Software Solutions können für etwaige Schäden verantwortlich gemacht werden die unter Umständen durch die Verwendung der zur Verfügung gestellten Software, Hardware, Treiber, oder der hier beschriebenen Schritte entstehen können.

1.4 Warnhinweise

PiXtend darf **nicht** in sicherheitskritschen Systemen eingesetzt werden.

Prüfen Sie vor der Verwendung die Eignung von Raspberry Pi und PiXtend für Ihre Anwendung.

2. Hardware-Verbindung zum CAN-Bus

Die korrekte Anschlussbelegung des PiXtend CAN-Ports kann dem technischen Datenblatt entnommen werden.

Beachten Sie bitte auch den Jumper für die Terminierung des CAN-Bus.

3. Vorbereitung und Test unter Linux

Damit der CAN Controller später in CODESYS verwendet werden kann, sind zunächst einige Anpassungen am Linux Betriebssystem notwendig.

Falls Sie bereits Anpassungen an Ihrem Image vorgenommen haben empfehlen wir für erste CAN-Tests eine neue SD-Karte zu erstellen.

Wir empfehlen Ihnen die Verwendung unseres "PiXtend Image CODESYS", welches Sie immer in der aktuellsten Version in unserem Download-Bereich finden.

Hinweis: Alle hier beschriebenen Schritte beziehen sich auf das Raspbian Image "Jessie" und werden als user "pi" durchgeführt.

Die Unterstützung des auf dem PiXtend verwendeten CAN-Controllers MCP2515 ist erst ab der Kernel Version 4.0.8 gewährleistet. Falls sie das "Wheezy" Image verwenden möchten muss daher zuerst auf eine Kernel Version 4.0.8 (oder neuer) geupdated werden.

3.1 Vorbereitung des SD-Karten-Image

Laden Sie das Raspbian "Jessie" Image von der offiziellen Download Seite herunter:

https://www.raspberrypi.org/downloads/raspbian/

Übertragen Sie das heruntergeladene Image mit einem Tool ihrer Wahl auf eine SD-Karte (empfohlen 8 GB). Unter Windows kann für diese Aufgabe z.B. das kostenlose Programm Win32DiskImager verwendet werden.

Booten Sie das neue Image und führen Sie die üblichen Anpassungen mittels raspi-config durch:

- Erweitern des Filesystems
- Einstellung ihrer Zeitzone
- Ländereinstellungen
- Spracheinstellungen
- Keyboard Layout
- SSH Zugriff aktivieren

Starten Sie nun das Raspberry Pi neu mit dem Befehl

```
pi@raspberrypi ~ $ sudo shutdown reboot
```

und loggen Sie sich anschließend über ssh in der Kommandozeile als user pi ein (z.B. Mit putty).

Tipp:

Verwenden Sie unser „PiXtend Image CODESYS“, so können Sie die Abschnitte 3.3 und 3.4 überspringen, da wir hier bereits alles für Sie vorbereitet haben.
Es ist lediglich eine kleine Anpassung notwendig, die in 3.2 beschrieben ist. Mit dem Unterschied, dass die Zeilen bereits vorhanden sind und Sie lediglich das #-Zeichen vor den entsprechenden Zeilen entfernen müssen.

3.2 Konfiguration des Device Tree Overlay

Damit der MCP2515 CAN-Chip des PiXtends-Boards vom Kernel bedient werden kann, sind folgende Anpassungen an der Datei */boot/config.txt* notwendig. Öffnen Sie den Editor mit dem Befehl

```
pi@raspberrypi ~ $ sudo nano /boot/config.txt
```

und fügen Sie folgende Einträge am Ende hinzu:

```
dtparam=spi=on
dtoverlay=mcp2515-can1
dtparam=oscillator=20000000
dtparam=interrupt=4
dtparam=spimaxfrequency=1000000
dtoverlay=spi-bcm2835-overlay
dtoverlay=spi-dma
dtdebug=on
```

Speichern Sie die Änderungen mit Ctrl-O und verlassen Sie den Editor mit Ctrl-X.

Starten Sie das Raspberry Pi nun neu:

```
pi@raspberrypi ~ $ sudo reboot
```

3.3 Blacklist bearbeiten

Damit das Kernel Modul „mcp251x“ beim booten noch nicht automatisch geladen wird, fügen Sie es der Blacklist hinzu.

Öffnen Sie dazu die Datei /etc/modprobe.d/raspi-blacklist.conf

```
pi@raspberrypi ~ $ sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

und fügen Sie folgende Zeile ein:

```
blacklist mcp251x
```

Die Datei ist zunächst komplett leer, kann aber dennoch verwendet werden.

Speichern Sie die Änderungen mit Ctrl-O und verlassen Sie den Editor mit Ctrl-X.

Starten Sie das Raspberry Pi nun neu:

```
pi@raspberrypi ~ $ sudo reboot
```

3.4 CAN Script anlegen

Um den PiXtend CAN-Controller zu aktivieren müssen prinzipiell folgende Aktionen durchgeführt werden:

- GPIO 24 als Ausgang konfigurieren und auf TRUE setzen (SPI Enable)
- GPIO 27 als Ausgang konfigurieren und auf TRUE setzen (SPI CS1 wird dadurch an den CAN Controller weitergereicht, der DAC wird deaktiviert)
- Das Kernel Modul "mcp251x" laden
- Das Interface *can0* konfigurieren und aktivieren

Da hierzu immer die gleichen Schritte notwendig sind, legen wir dazu ein neues Skript im Home Verzeichnis des users *pi* an:

```
pi@raspberrypi ~ $ nano activateCAN.sh
```

Fügen Sie folgende Zeilen ein:

```
#!/bin/sh
if [ ! -d "/sys/class/gpio/gpio24" ]; then
    echo "24" > /sys/class/gpio/export
fi
echo "out" > /sys/class/gpio/gpio24/direction
echo "1" > /sys/class/gpio/gpio24/value
```

```
if [ ! -d "/sys/class/gpio/gpio27" ]; then
    echo "27" > /sys/class/gpio/export
fi
echo "out" > /sys/class/gpio/gpio27/direction
echo "1" > /sys/class/gpio/gpio27/value
sleep 1
sudo modprobe mcp251x
sudo /sbin/ip link set can0 up type can bitrate 125000
sudo ip -s -d link show can0
```

Speichern Sie die Änderungen mit Ctrl-O und verlassen Sie den Editor mit Ctrl-X.

Machen Sie das Skript ausführbar, indem Sie folgenden Befehl eingeben:

```
pi@raspberrypi ~ $ chmod +x ./activateCAN.sh
```

3.5 CAN aktivieren

Wenn Sie nun das Skript mittels

```
pi@raspberrypi ~ $ sudo ./activateCAN.sh
```

aufrufen, wird der CAN Controller aktiviert.

Mit einem Aufruf von

```
pi@raspberrypi ~ $ ifconfig
```

kann kontrolliert werden ob nun ein Eintrag für ein *can0* Interface vorhanden ist.

3.6 CAN-Utilities installieren und verwenden

Installieren Sie nun die can-utils, eine Sammlung nützlicher user-space Programme um mit SocketCAN in Linux direkt auf die CAN-Schnittstelle zugreifen zu können:

```
pi@raspberrypi ~ $ sudo apt-get install can-utils
```

Mit dem Befehl

```
pi@raspberrypi ~ $ cansend can0 123#deadbeef
```

wird eine CAN Message mit ID 123 und dem Inhalt 0xdeadbeef verschickt.

Mit dem Befehl

```
pi@raspberrypi ~ $ candump can0
```

kann der gesamte Datenverkehr auf *can0* mitgelauscht werden:

Mit dem Program `cangen` können laufend CAN-Messages zu Testzwecken generiert werden.

Weitere nützliche Funktionen und Parameter der can-utils entnehmen Sie bitte den jeweiligen Hilfe Seiten.

Mit diesen Tools können Sie bereits an einer CAN Bus Kommunikation unter Linux teilnehmen.

Bitte fahren Sie mit den weiteren Schritten (Vorbereitung für CODESYS) erst fort wenn Sie mit den can-utils erfolgreich Daten empfangen und versendet haben.

4. Vorbereitungen für CODESYS

4.1 CODESYS Runtime installieren[1]

Zunächst muss die CODESYS-Runtime auf dem Raspberry Pi installiert werden.
Öffnen Sie dazu CODESYS und verwenden Sie das Menu Tools->Update Raspberry Pi

Tragen Sie in die Eingabefelder ihre Raspberry Login Datein ein.

Klicken Sie auf den "Scan" Button um das Netzwerk nach ihrem Raspberry Pi zu durchsuchen.

Wählen Sie das Raspberry Pi aus und klicken Sie auf "OK".

Die CODESYS Runtime wird nun automatisch auf das Raspberry Pi übertragen.

4.2 Start Skript anlegen

Damit das Kernel Modul für den PiXtend CAN-Controller korrekt geladen werden kann, benötigen wir ein Start Skript das folgende Aktionen in exakt dieser Reihenfolge ausführt:

- CODESYS Control stoppen
- CAN aktivieren mit Hilfe des vorher erstellen Skripts `activateCAN.sh` (Siehe Punkt 3.5)
- CODESYS Control starten

Öffnen Sie eine Konsole auf dem Raspberry Pi (z.B. via SSH mit dem Tool *putty*) und legen Sie die Datei `startPLCCAN.sh` im home-Verzeichnis an:

```
pi@raspberrypi ~ $ nano startPLCCAN.sh
```

Fügen Sie die folgenden Zeilen ein:

```
#!/bin/sh
sudo /etc/init.d/codesyscontrol stop
sudo /home/pi/activateCAN.sh
sudo /etc/init.d/codesyscontrol
```

Speichern Sie die Änderungen mit Ctrl-O und verlassen Sie den Editor mit Ctrl-X.

1 Sie können den Abschnitt 4.1 überspringen, wenn Sie unser vorbereitetes SD-Image "PiXtend Image CODESYS" verwenden. Dieses können Sie hier herunterladen: www.pixtend.de/downloads/

Machen Sie das Skript ausführbar, indem Sie folgenden Befehl eingeben:

```
pi@raspberrypi ~ $ chmod +x ./startPLCCAN.sh
```

4.3 Baudrate Skript anlegen

Legen Sie nun folgendes Baudrate Skript an:

```
pi@raspberrypi ~ $ sudo nano /root/rts_set_baud.sh
```

und fügen Sie die folgenden Zeilen ein:

```
#!/bin/sh
BITRATE=`expr $2 \\* 1000`
ifconfig $1 down
echo ip link set $1 type can bitrate $BITRATE
ip link set $1 type can bitrate $BITRATE
ifconfig $1 up
```

Speichern Sie die Änderungen mit Ctrl-O und verlassen Sie den Editor mit Ctrl-X.

Machen Sie das Skript ausführbar, indem Sie folgenden Befehl eingeben:

```
pi@raspberrypi ~ $ sudo chmod +x /root/rts_set_baud.sh
```

4.4 CODESYS Control mit PiXtend CAN-Unterstützung starten

Starten Sie das Raspberry Pi neu:

```
pi@raspberrypi ~ $ sudo reboot
```

Nach dem Bootvorgang wird die CODESYS Control Runtime automatisch gestartet.

Damit Sie CODESYS mit CAN Unterstützung verwenden können, muss nach jedem Start zusätzlich das Skript "startPLCCAN.sh" aufgerufen werden:

```
pi@raspberrypi ~ $ sudo ./startPLCCAN.sh
```

Durch den Aufruf des Skripts wird wie oben schon erwähnt zunächst die CODESYS-Runtime beendet, die GPIOs entsprechend konfiguriert, das mcp251x Kernel Modul geladen, die *can0* Schnittstelle konfiguriert, und CODESYS Control anschließend wieder

neu gestartet.

Falls Sie die CAN Unterstützung dauerhaft aktiviert haben möchten, können Sie natürlich das Start Skript automatisch bei jedem Neustart ausführen lassen, in dem Sie die Datei `/etc/rc.local` bearbeiten und folgende Zeilen **vor** `exit 0` hinzufügen:

```
# Start CODESYS Control with PiXtend CAN Support
sudo /home/pi/startPLCCAN.sh
```

Testen Sie die Konfiguration indem Sie das Raspberry Pi neu starten:

```
pi@raspberrypi ~ $ sudo reboot
```

Mit einem Aufruf von `lsmod` können Sie prüfen ob das "mcp251x" Modul nach dem Neustart geladen wurde, und mit einem Aufruf von `top` ob CODESYS Control läuft.

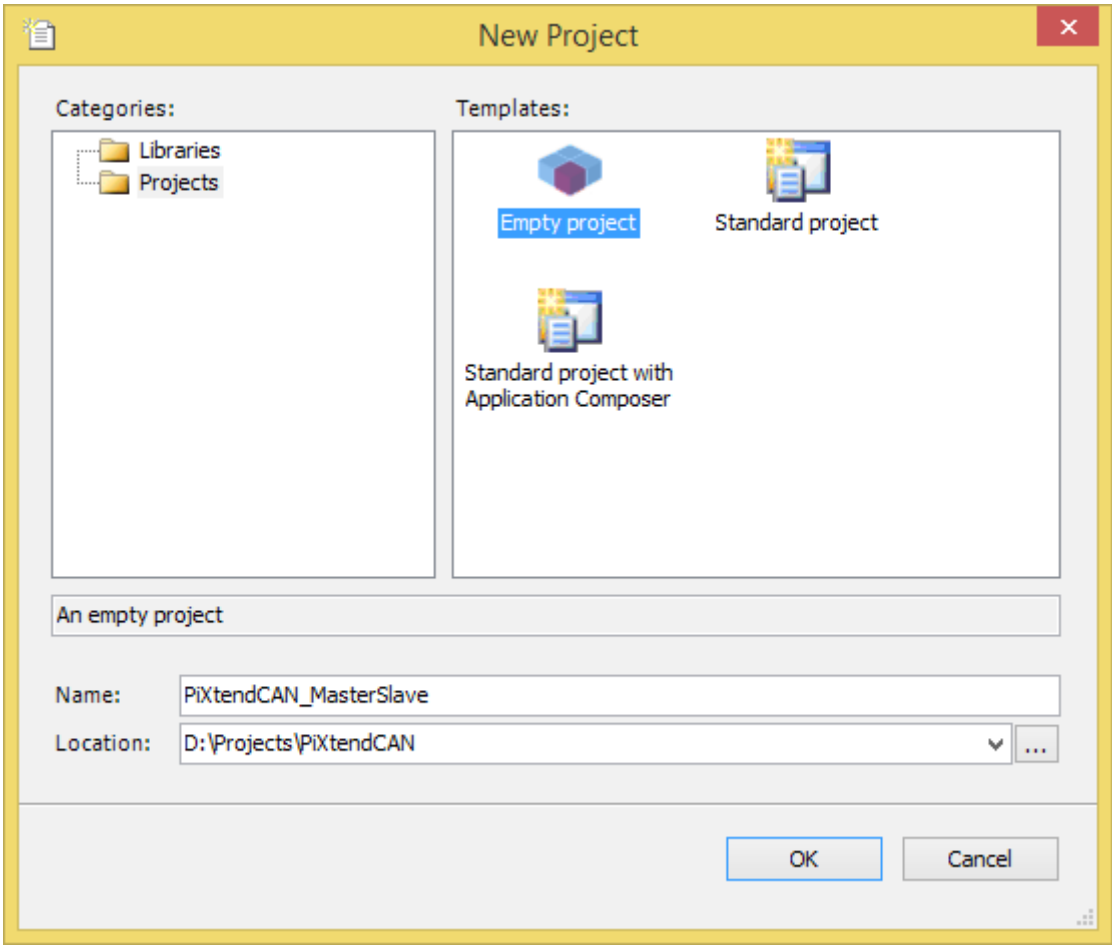
5. CODESYS Projekt mit CANopen

Wenn die Vorbereitungen und Tests unter Linux erfolgreich waren kann nun ein CANopen Test Projekt erstellt werden. Dabei werden zwei PiXtend-Geräte miteinander über CANopen kommunizieren.

Wir wird ein leeres Projekt erstellt zu dem zwei PiXtend Geräte hinzugefügt werden. Ein Gerät ist dabei der CANopen-Master, das andere der CANopen-Slave.

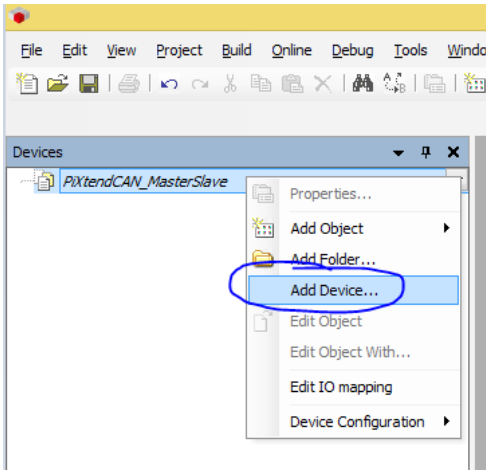
Der Einfachkeit halber wird zunächst nur 1 Byte vom Master an den Slave übertragen, das Beispiel lässt sich anschließend jedoch beliebig erweitern.

Erzeugen Sie zunächst ein leeres Projekt (Datei -> Neues Projekt -> Leeres Projekt) und nennen Sie es "PiXtendCAN_MasterSlave".

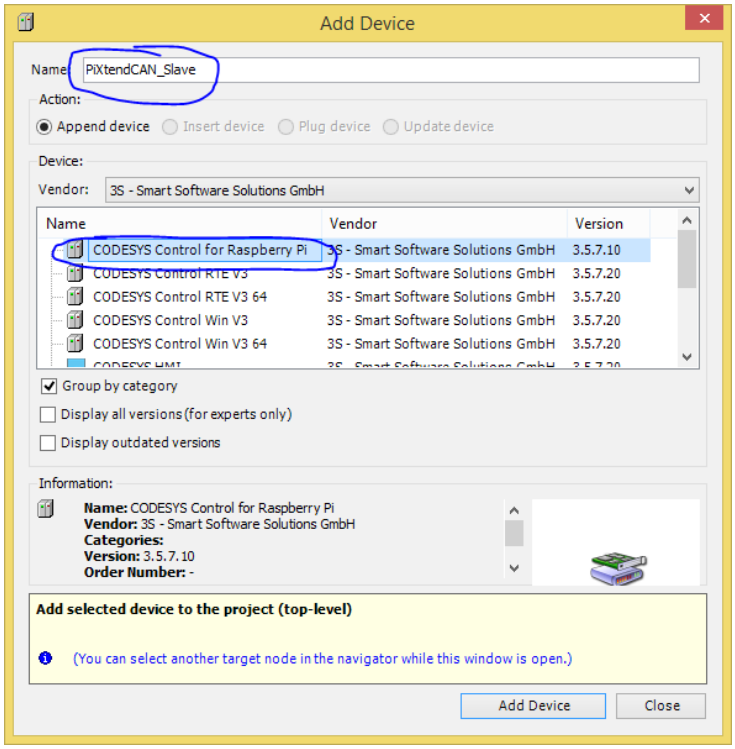


5.1 PiXtend als CAN-Slave

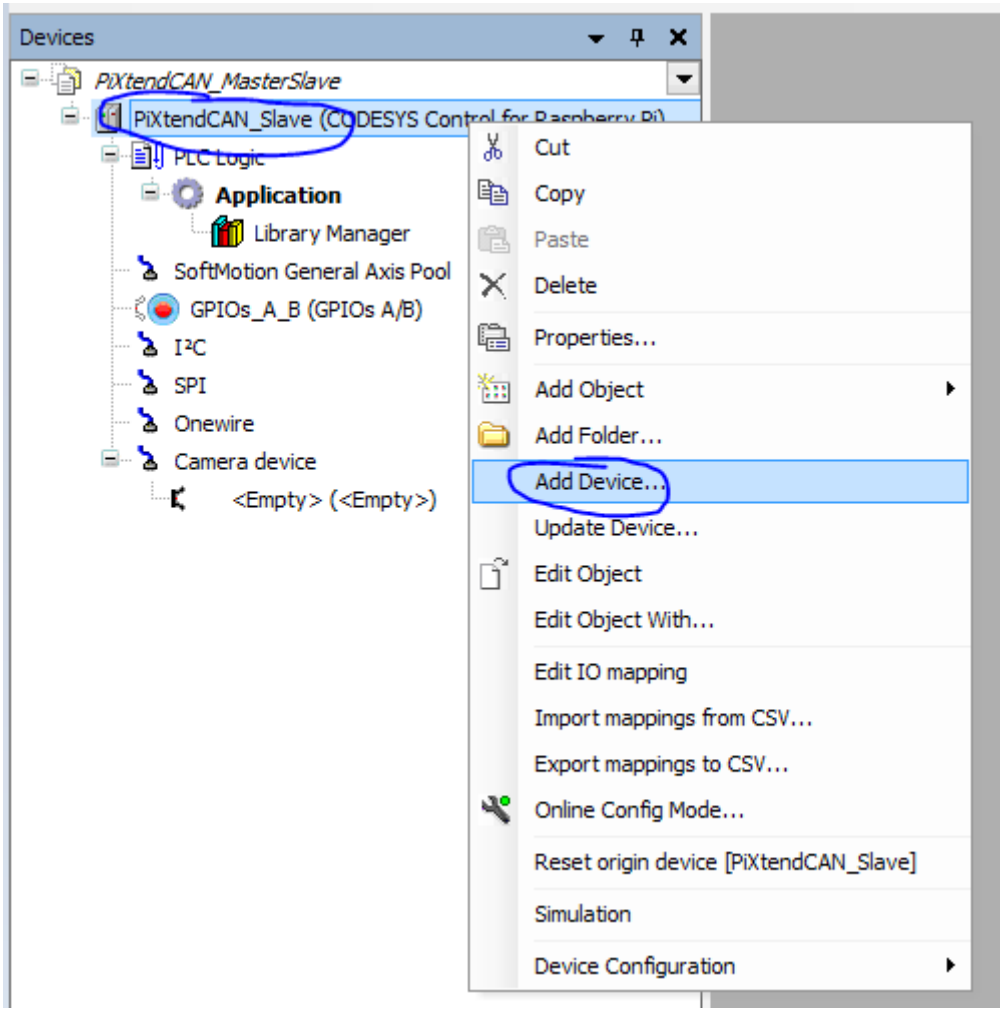
Fügen Sie dem Projekt nun ein neues Gerät hinzu, indem Sie im Projektbaum auf das Projekt rechts-klicken und "Gerät hinzufügen" auswählen.



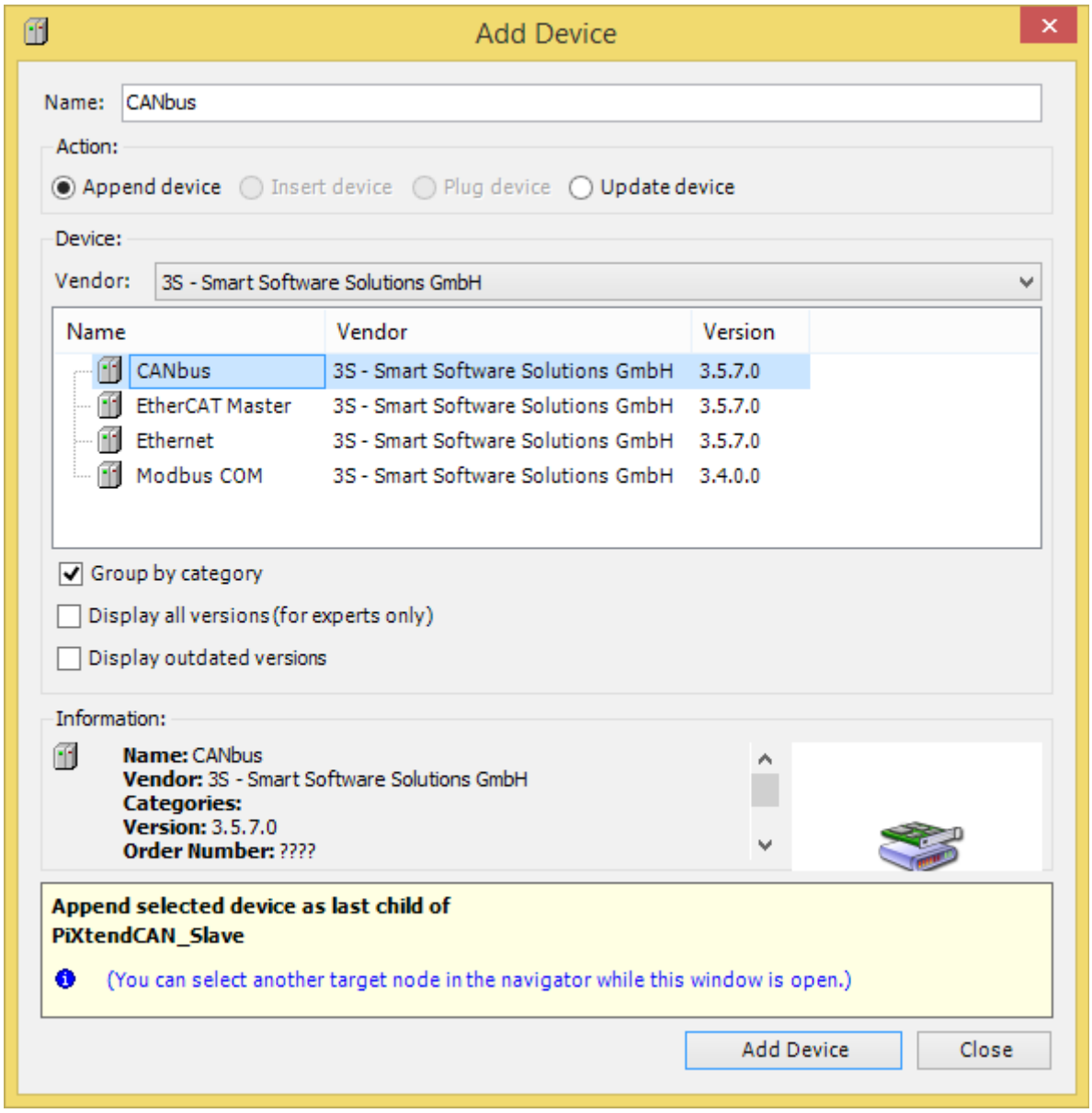
Wählen Sie nun "CODESYS Control for Raspberry Pi" als Gerät aus, geben Sie ihm den Namen "PiXtendCAN_Slave" und klicken Sie auf hinzufügen.



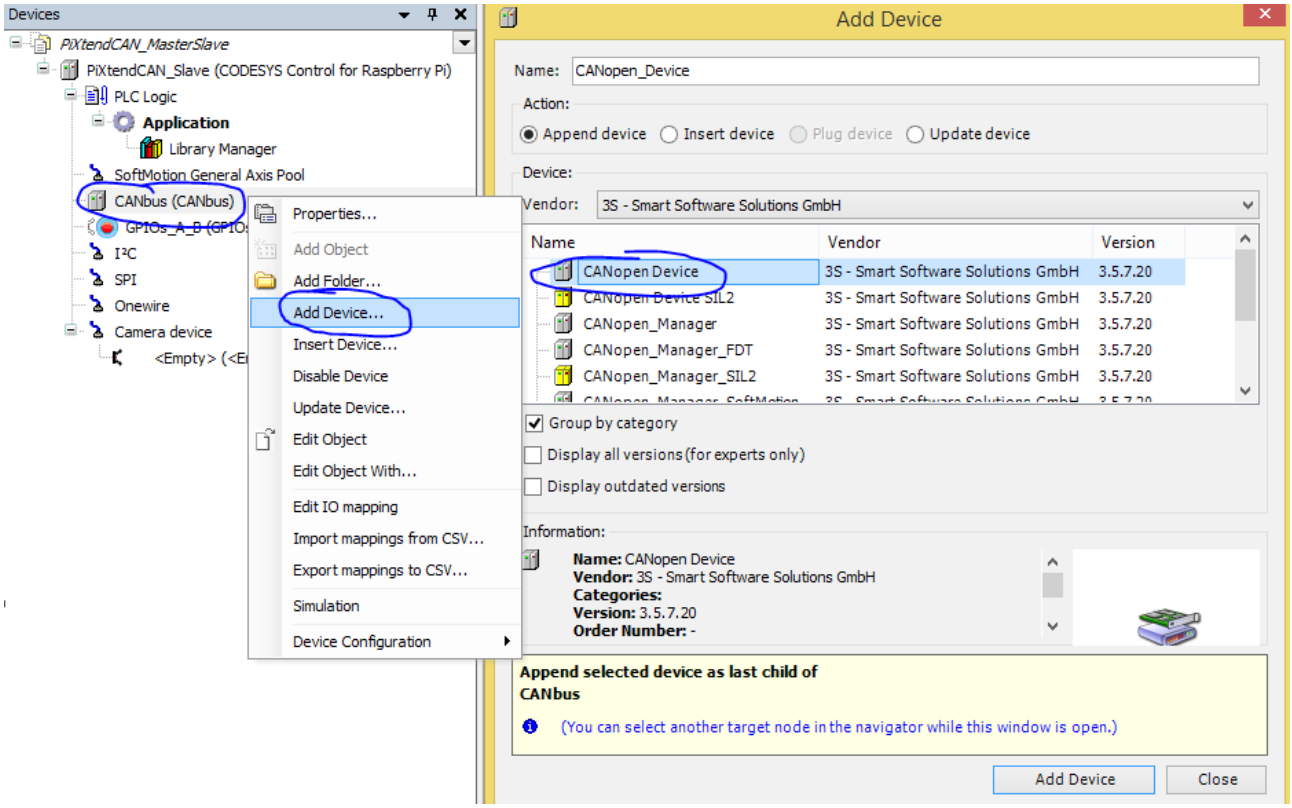
Machen Sie nun im Projektbaum einen Rechtsklick auf das soeben hinzugefügte "PiXtendCAN_Slave" Gerät:



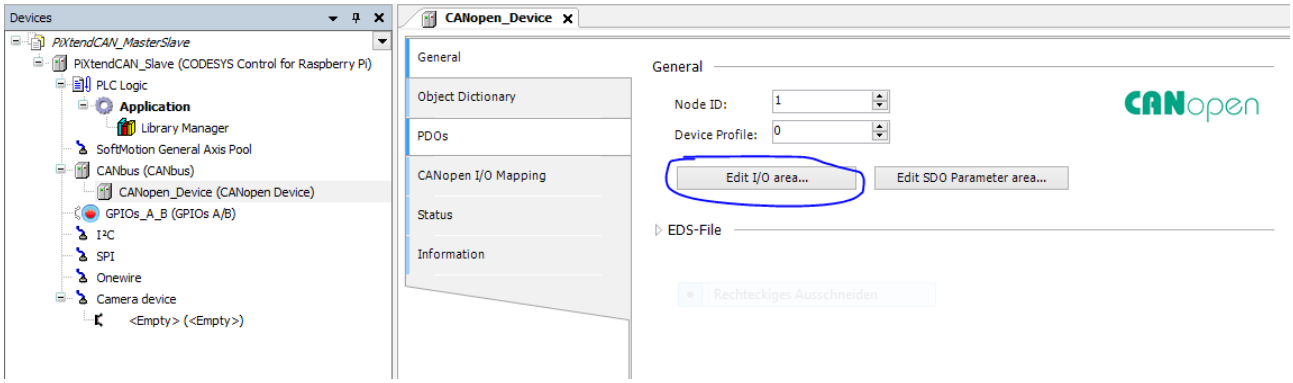
Im Dialog wählen Sie bitte "CANbus" aus und fügen das Gerät hinzu.



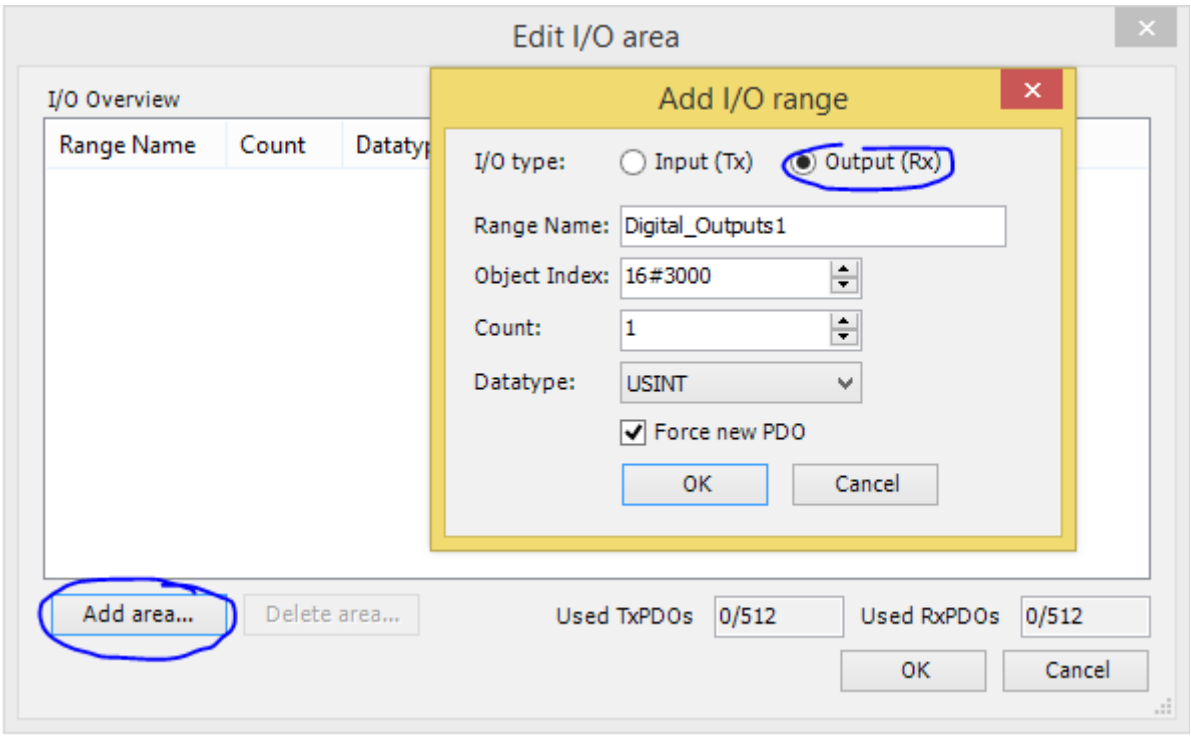
Machen Sie nun einen Rechtsklick auf das soeben hinzugefügte "CANbus" Gerät und fügen Sie ein "CANopen Device" hinzu.



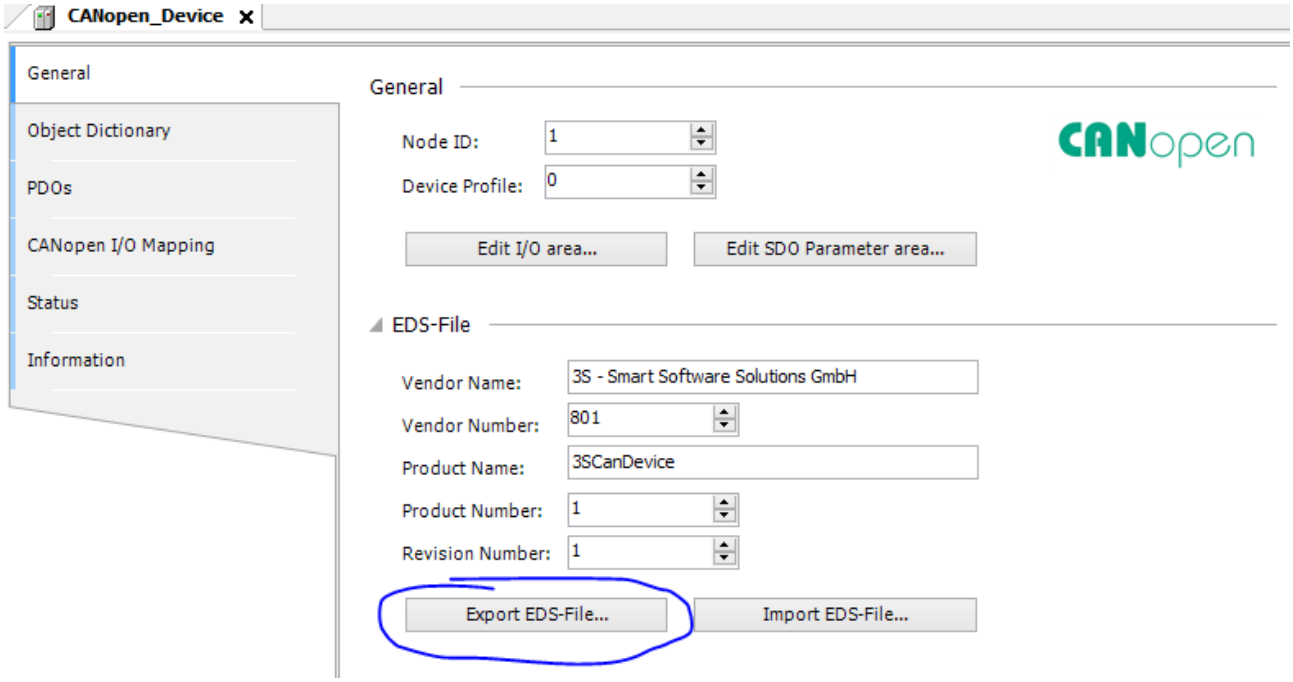
Doppelklicken Sie nun im Projektbaum auf das hinzugefügte CANopen-Gerät und klicken Sie anschließend auf den Button "E/A-Bereich bearbeiten"



Fügen Sie nun einen neuen Bereich hinzu und wählen Sie "Ausgang (Rx)". Die restlichen Felder können unverändert bleiben da der Master zunächst nur ein einzelnes USINT (BYTE) an den Slave schicken soll.



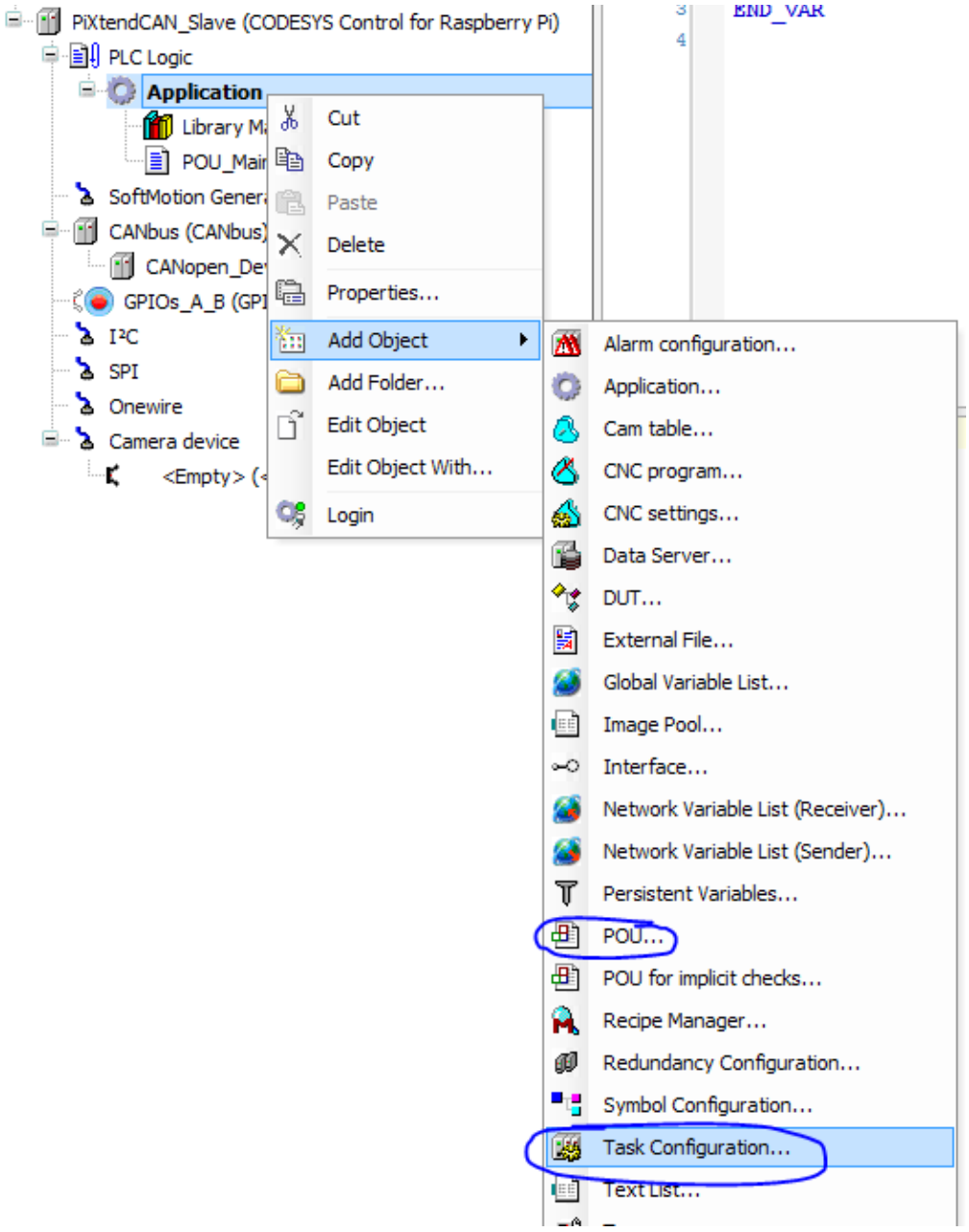
Bestätigen Sie und schließen Sie den Dialog. Klappen Sie nun den Bereich "EDS-Datei" auf und exportieren Sie das soeben erzeugte Gerät als EDS-Datei.



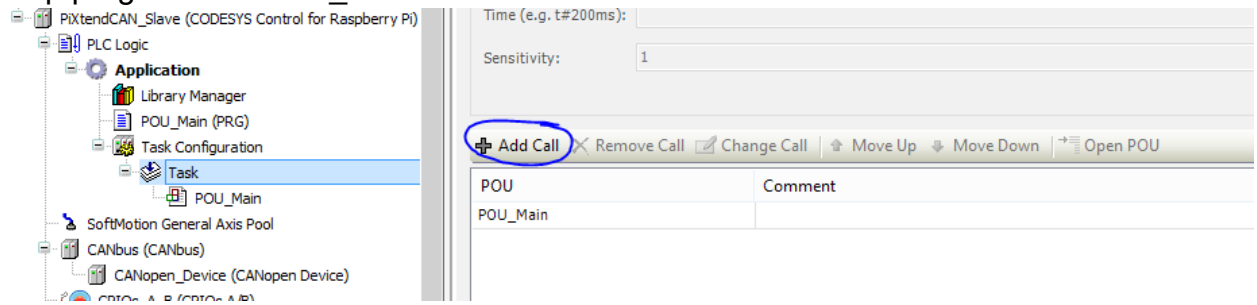
Speichern Sie die EDS Datei im Projektordner ab.

Das EDS File enthält alle Informationen über das soeben erzeugte CAN Gerät und wird später wieder benötigt um den CAN Master damit zu konfigurieren.

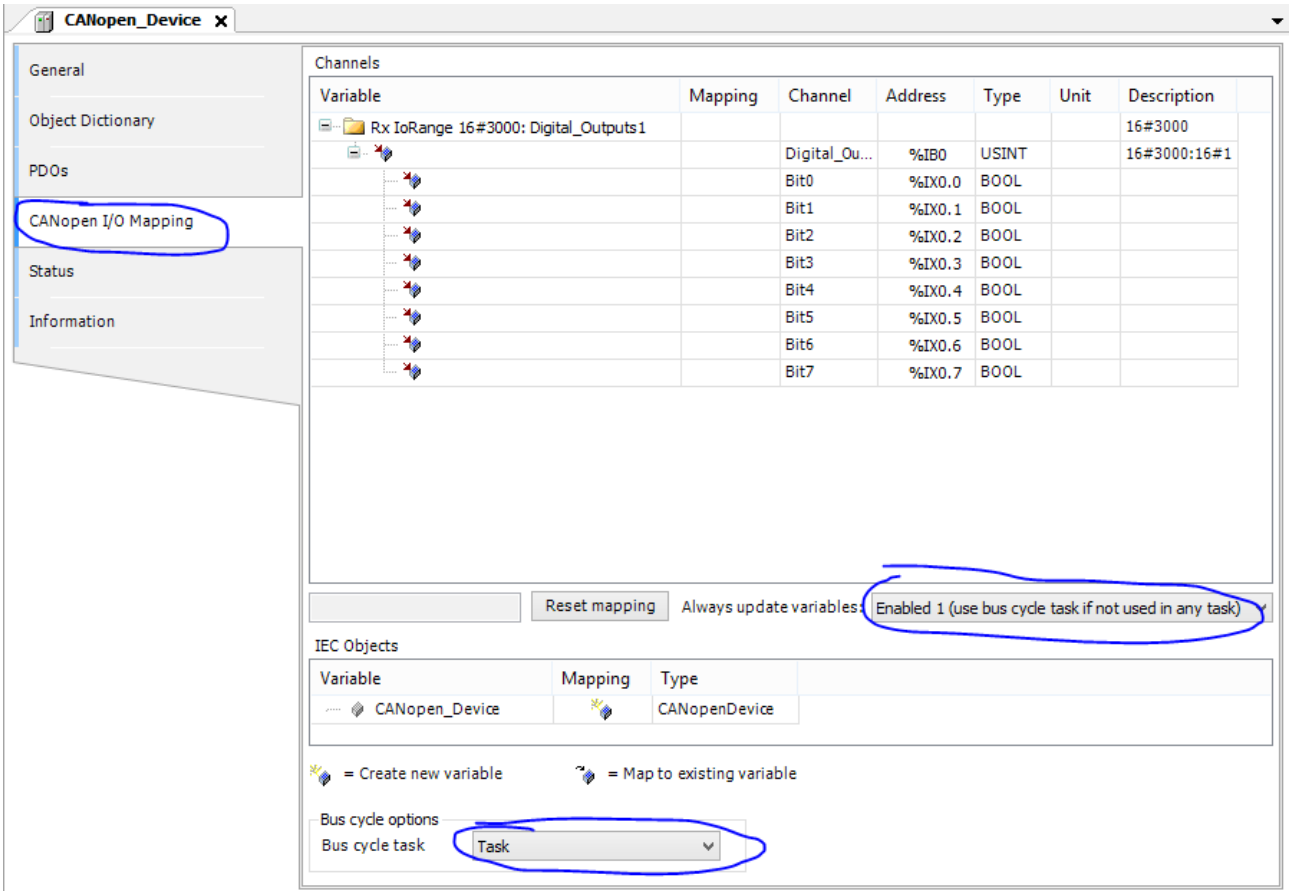
Fügen Sie nun der Applikation noch eine Taskkonfiguration hinzu sowie eine POU namens "POU_Main".



Öffnen Sie die Taskkonfiguration und fügen Sie dem Task einen Aufruf des Hauptprogrammes POU_Main hinzu:

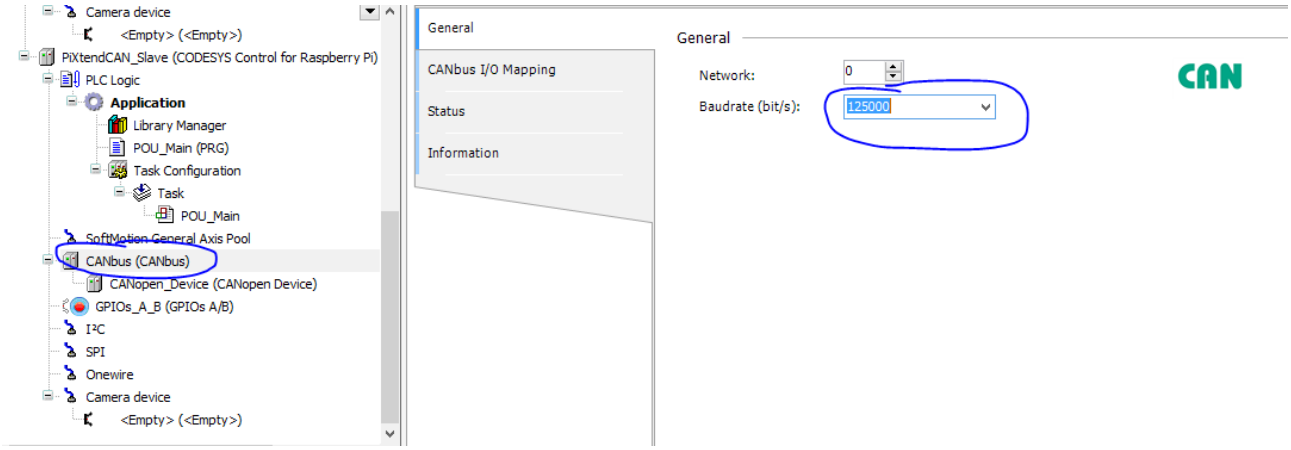


Wechseln Sie wieder zum CANopen_Device und ändern Sie dort die Einstellungen für die Aktualisierung der Variablen unter dem Reiter "CANopen E/A-Abbild":



Wählen Sie die Option "Aktiviert 1 – Buszyklustask verwenden wenn in keinem Task verwendet" sowie "Task" als Buszyklus-Task.

Abschließend sollte noch die Baudrate für den CAN Bus auf 125000 reduziert werden:

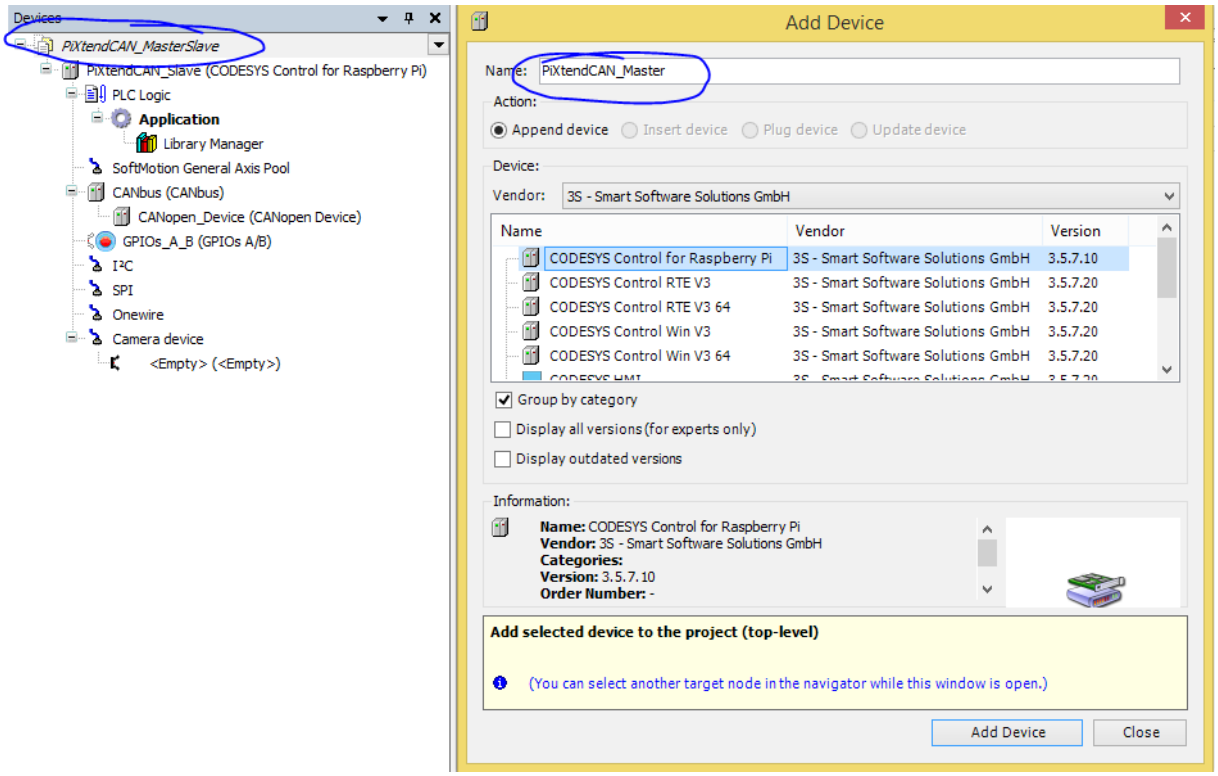


Damit ist die Konfiguration des Slaves abgeschlossen.

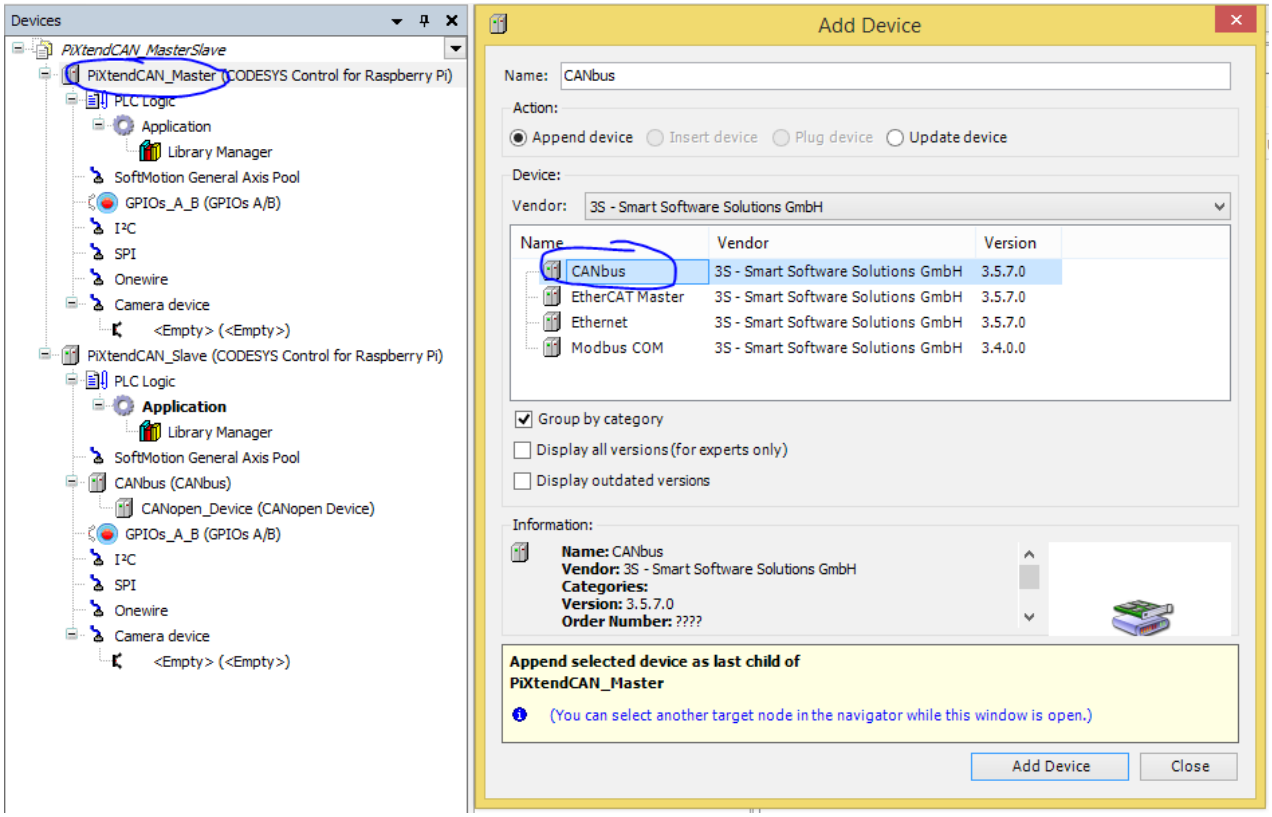
5.2 PiXtend als CANopen-Master

Nun fügen wir dem Projekt ein weiteres PiXtend Gerät hinzu das als CANopen-Master fungiert.

Führen Sie dazu wieder einen Rechtsklick auf das Projekt "PiXtendCAN_MasterSlave" im Projektbaum aus, und fügen Sie ein weiteres "CODESYS Control for Raspberry Pi" Gerät mit dem Namen "PiXtendCAN_Master" hinzu:

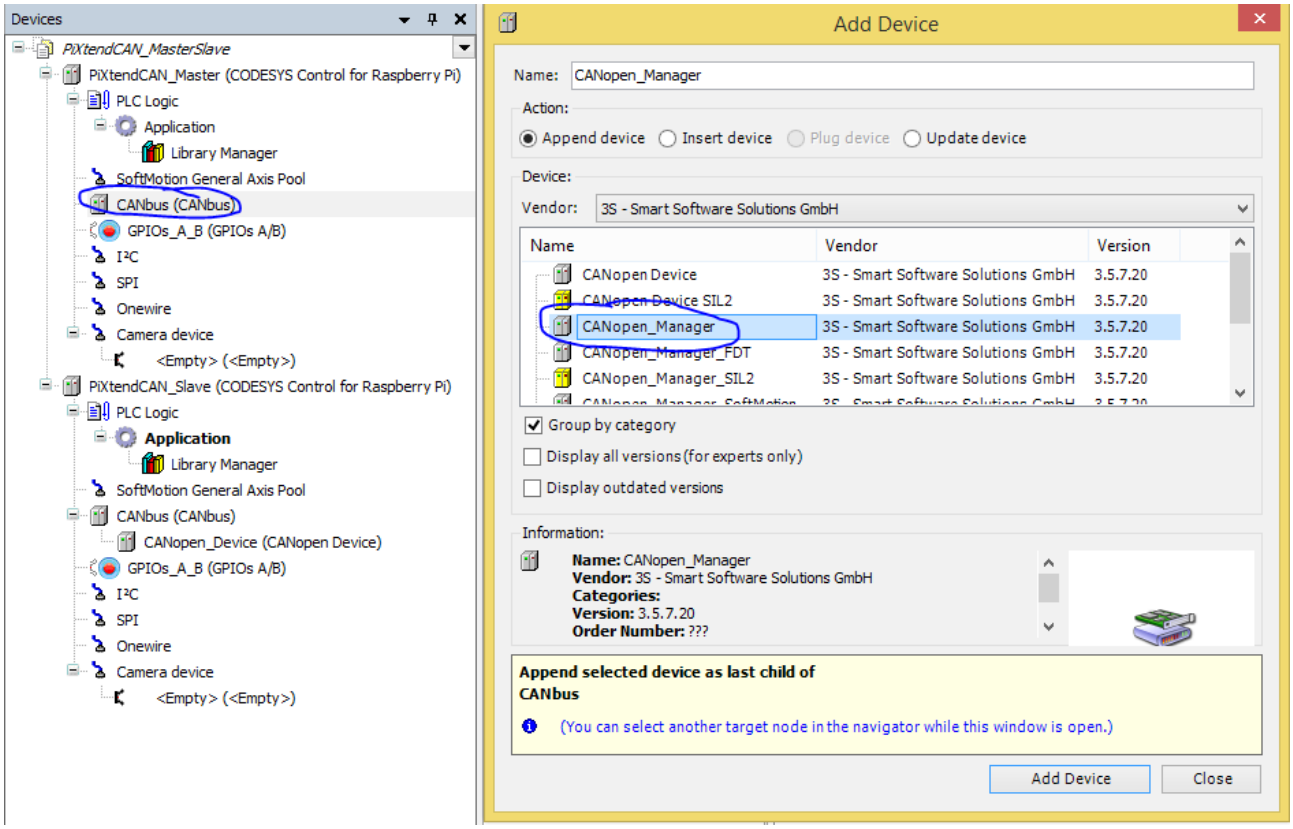


Fügen Sie dem "PiXtendCAN_Master" wieder einen CANbus hinzu:

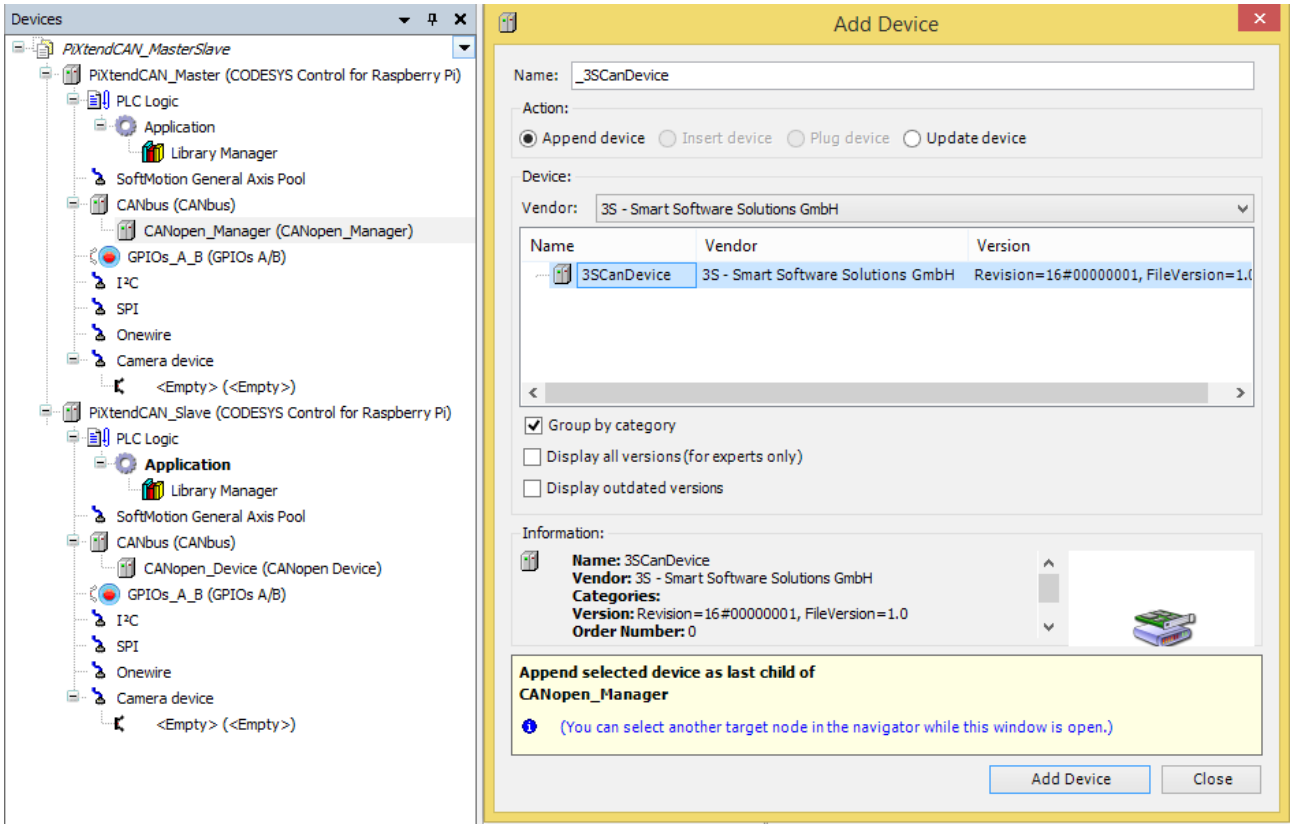


Reduzieren Sie die Baudrate des CANbus auf 125000, bzw auf die gleiche Einstellung die beim Slave gewählt wurde.

Machen Sie nun einen Rechtsklick auf den soeben hinzugefügten "CANbus" Eintrag und fügen Sie diesmal einen "CANOpenManager" hinzu:



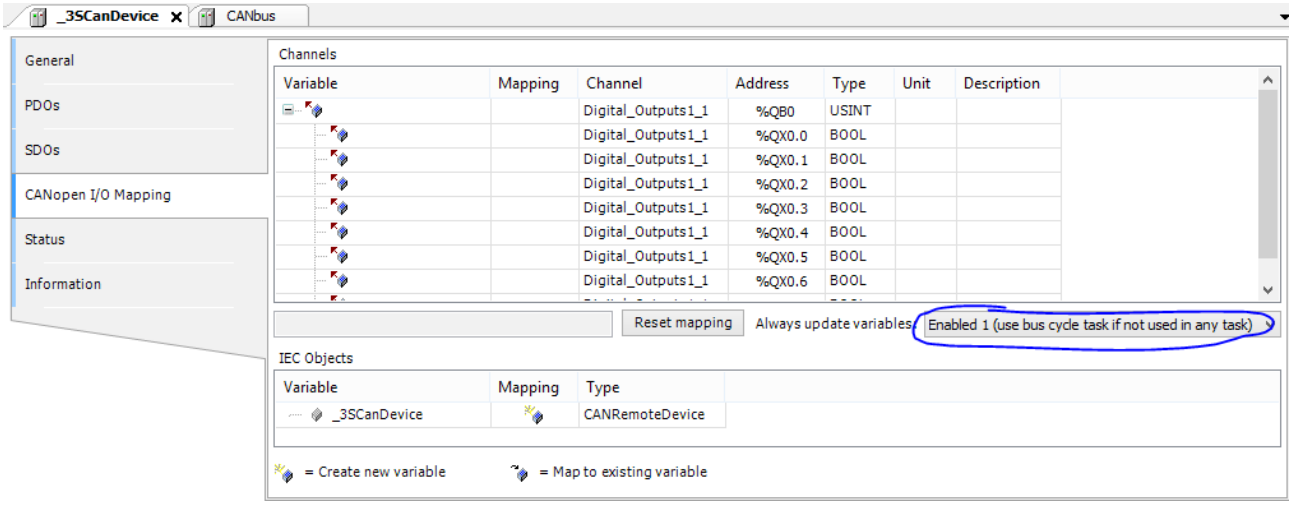
Wählen Sie nun den soeben hinzugefügten CANopen_Manager aus und fügen Sie wieder mit einem Rechtsklick das von ihnen vorher neu erzeugte "3SCanDevice" hinzu.



Hinweis: Sollte das Gerät nicht in der Liste vorgeschlagen werden, so muss die Gerätebeschreibungsdatei möglicherweise über das Geräte-Repository nachinstalliert werden. Öffnen Sie dazu das Geräte-Repository, klicken Sie auf den Button "Installieren" und wählen Sie die für den Slave erzeugte .eds Datei aus.

Erzeugen Sie wieder eine Taskkonfiguration und eine "POU_Main" als Hauptprogramm für die Applikation und fügen Sie dem Task einen Aufruf der "POU_Main" hinzu (gleiches Vorgehen wie beim Slave).

Öffnen Sie nun die E/A Konfiguration des Gerätes "_3SCanDevice" und wählen Sie den Eintrag "Aktiviert 1 – Buszyklustask verwenden wenn in keinem Task verwendet"



5.3 Programm Download und Test

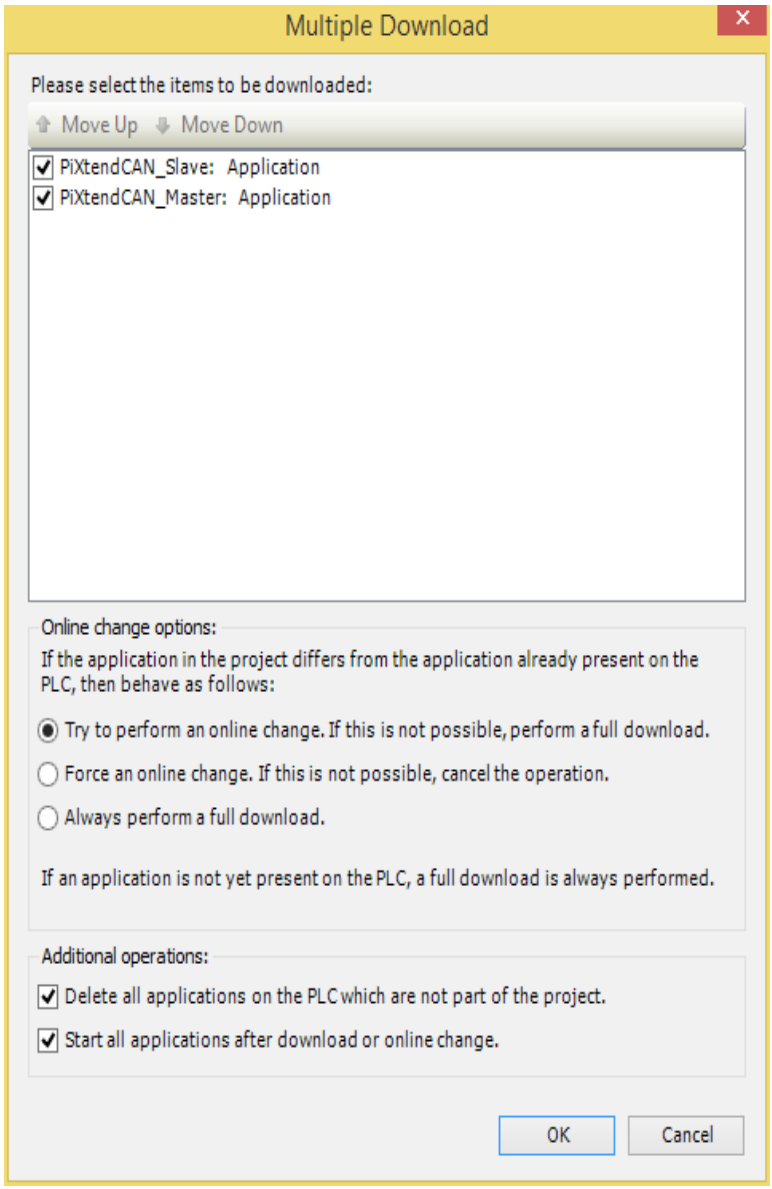
Öffnen Sie nacheinander die Kommunikationseinstellungen für die beiden Geräte "PiXtendCAN_Master" und "PiXtendCAN" und wählen Sie die entsprechenden Raspberry Pi Geräte aus auf die der Download erfolgen soll.

Anschließend können Sie im Hauptmenu "Online" die Funktion "Mehrfacher Download" verwenden um die Applikationen gleichzeitig auf die entsprechenden Controller zu laden.

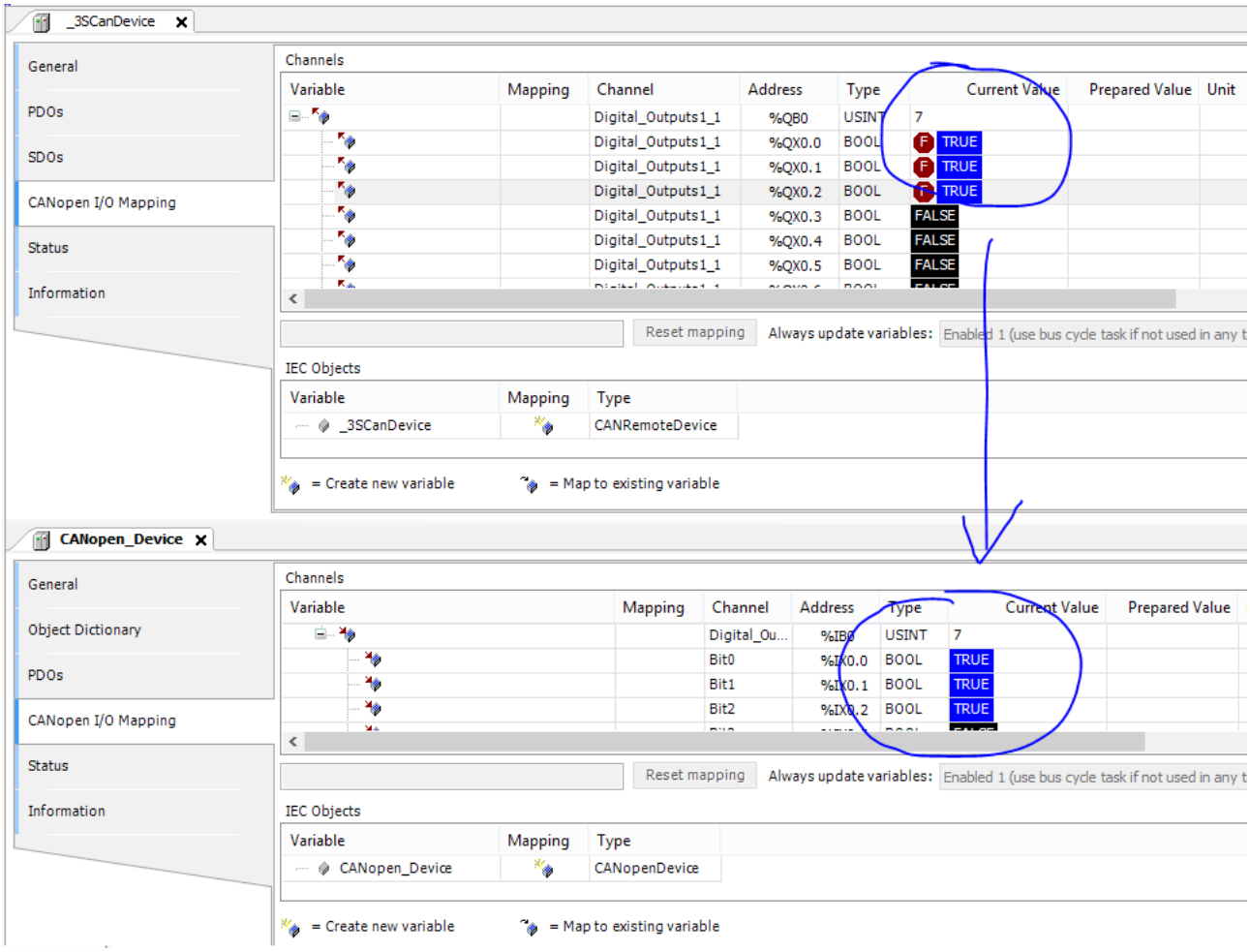
Hinweis: Sobald sich mehrere Applikationen im Projektbaum befinden so kann mit einem Rechtklick auf die Applikation -> "Aktive Applikation setzen" die gewünschte Applikation ausgewählt werden.

Gehen Sie nacheinander Online zu den Applikationen und starten Sie diese.

Wenn alles richtig konfiguriert wurde und beide Geräte korrekt miteinander verbunden sind werden die CAN-Einträge im Projektbaum grün markiert dargestellt.



Wenn Sie nun den E/A Bereiche des CAN Masters öffnen und die Werte mit einem Force verändern, so werden die Werte automatisch vom Master an den Slave übertragen:



Natürlich könnte die Variable nun noch gemapped werden und z.B. direkt im Hauptprogramm verwendet werden.