Application Note

Modbus-TCP Kommunikation

Verbindung mehrerer PiXtend-Baugruppen
Einrichten von Modbus-Master & Slave unter CODESYS
I/O-Daten austauschen & Anzeigen

Modbus [1]

APP-PX-560

Stand 24.08.2016, V1.01

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, Texte & Bilder eingefügt	TO
1.01	- Dokument überarbeitet, Titelbild aktualisiert - Einleitungskapitel erstellt - Tipps & Hinweise eingefügt	TG

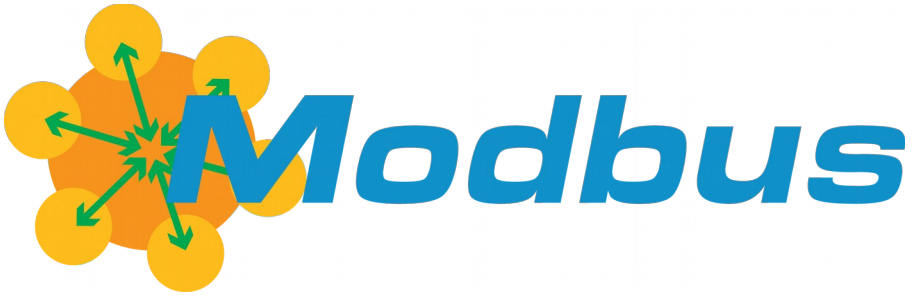
Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

Das Modbus[1]-Protokoll hat sich in den letzten Jahren zu einer Art De-facto-Standard in der industriellen Automation entwickelt. Modbus basiert auf einer Master/Slave-Architektur. Es gibt also einen Master und mehrere Slaves.

Als Übertragungsmedium kann RS232 / RS485 (Modbus RTU) oder Ethernet (Modbus TCP) dienen. Wir schauen uns in dieser AppNote das moderne Modbus TCP genauer an.



[1]

Bei Modbus TCP haben Sie viele Vorteile. Heute hat jeder Hauhalt, jede Firma und Bildungseinrichtung ein Ethernet-basiertes Netzwerk. Egal ob per Kabel oder W-LAN, die Infrastrutur besteht. Mit Modbus TCP können wir genau diese Infrastruktur nutzen, ohne weitere Kabel zu verlegen. Ein großer Vorteil für eine Haus Automatisierung (Smart Home).

In dieser App-Note zeigen wir Ihnen, wie Sie zwei PiXtend-Boards per Modbus TCP miteinander verbinden und Daten austauschen lassen. Ein Gerät fungiert als Master, das zweite als Slave.
Der Master kann die Ausgänge des Slave setzen und Eingänge lesen. Der Slave dient also als E/A-Erweiterung, die bei einer Home Automation sinnvoll ist und problemlos per WLAN angebunden werden kann.

Für die komfortable Bedienung bauen wir auch noch eine kleine Web-Visualisierung auf.

Dann bleibt nur noch zu sagen:

„Viel Spaß beim nachvollziehen der folgenden Schritte und bei Ihrem Automatisierungs-Projekt mit PiXtend!“

1 „ Modbus“ und das zugehörige Logo sind geschützte Markenzeichen der Modbus Organization, Inc. - www.modbus.org ;

1.1 Voraussetzungen & Vorbereitung

Es wird vorausgesetzt, dass Sie bereits mit CODESYS und PiXtend gearbeitet haben und den grundlegende Umgang beherrschen. Ansonsten möchten wir Sie gerne zunächst auf unsere Einführungsdokumente hinweisen:

App-Note: PiXtend mit CODESYS – Installation der Softwarekomponenten
App-Note: PiXtend mit CODESYS – Projekt erstellen

Hardware:

- PC mit CODESYS mit Ethernet oder WLAN-Schnittstelle
- 2x PiXtend V1.2 / V1.3, jeweils mit Raspberry Pi
- Netzwerkverbindung aller Geräte per WLAN/Ethernet

Software:

- CODESYS V3 Programmierumgebung auf dem PC
- *PiXtend Image CODESYS* aus unserem Download-Bereich
  - aufgespielt auf beiden SD-Karten der Raspberry Pi Computern
- Zwei Standard-Projekte für PiXtend vorbereitet (laut AppNote)

Eines der Standard-Projekte wird mit dem Namen *Modbus_Master* und das andere mit dem Namen *Modbus_Slave* versehen.

Erst wenn diese beiden Projekte angelegt bzw. vorbereitet wurden, sollten Sie mit den nächsten Schritten im Kapitel *2. Modbus Master Programm* fortgefahren.

Die IP-Adressen von beiden Raspberry Pis im Netzwerk müssen bekannt sein. Am Besten notieren Sie sich diese im Vorhinein.

1.2 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.

1.3 Sicherheitshinweise

PiXtend darf nicht in sicherheitskritischen Systemen eingesetzt werden.

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

2. Modbus Master Programm

Als erstes offenen wir das Projekt, welchem wir den Namen *Modbus_Master* gegeben haben. Im nächsten Schritt wird ein *Ethernet Adapter* eingefügt. Mit einem Rechtsklick im Gerätebaum auf *Device* (*CODESYS Control for Raspberry Pi*) und dann *Gerät anhängen* kann der Ethernet-Adapter eingefügt werden, wie in Abbildung 1 gezeigt.

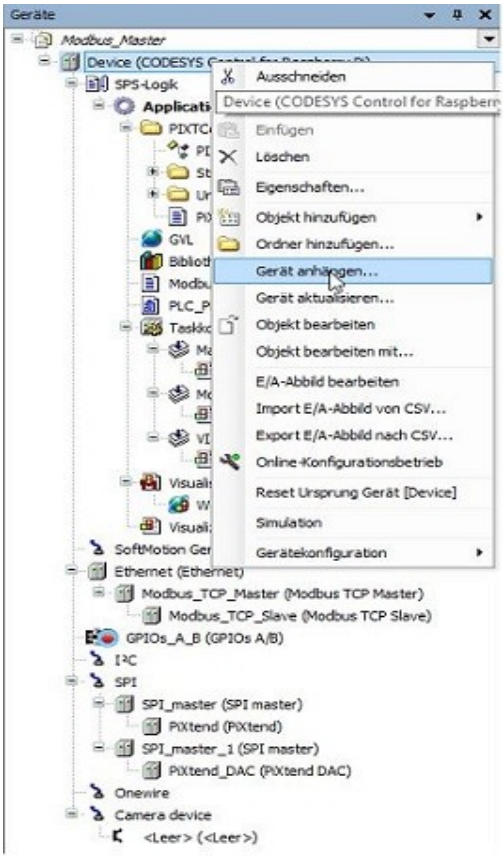


Abbildung 1: Ethernet-Adapter anhängen

Nun wird unter *Ethernet Adapter* das *Ethernet* Gerät ausgewählt.

Über den Button *Gerät anhängen* wird der Ethernet-Adapter an die Device angehängt.

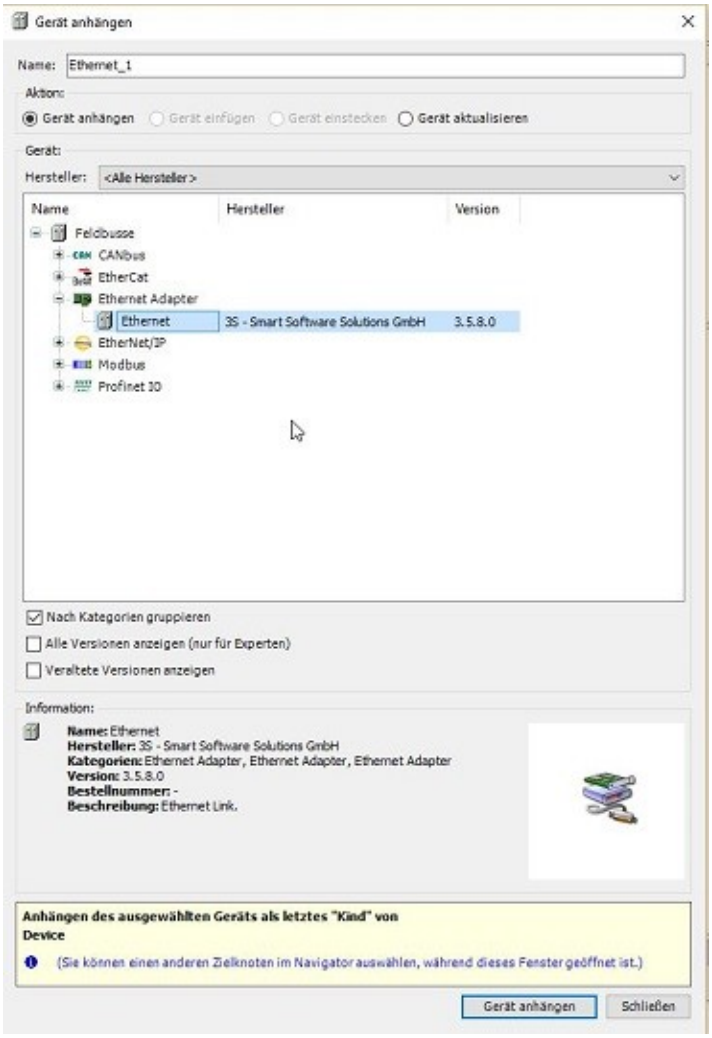


Abbildung 2: Ethernet Adapter anhängen

Der soeben eingehängte Ethernet-Adapter muss noch konfiguriert werden. Wir doppelklichen auf *Ethernet* und landen im Reiter *Allgemein*. Als Schnittstelle sollte hier (*eth0*) eingetragen werden (die Ethernet-Schnittstelle des Raspberry Pi). Falls Sie einen WLAN-Dongle oder das im RPi 3 B integrierte WLAN verwenden möchten, so tragen Sie im Feld *Netzwerkschnittstelle* die entsprechende Bezeichnung des WLANs ein. In der Regel ist das *wlan0*.

Tipp: Wenn Sie mit dem Gerät verbunden sind, kann CODESYS die Schnittstellen auch über den „...“ Button suchen.

Nun fügen wir noch zwei weitere Geräte ein, Modus TCP Master & Slave.

Zuerst mit Rechtsklick auf den *Ethernet* Adapter der jetzt im Gerätebaum zu sehen ist. Dann wieder über Gerät anhängen im Reiter *Modbus / Modbus TCP Master* das *Modbus TCP Master* Gerät auswählen. Zuletzt muss auf die gleiche Weise ein *Modbus TCP Slave* an den *Modbus TCP Master* angehängt werden.

Der Gerätebaum sollte nun wie folgt aussehen:

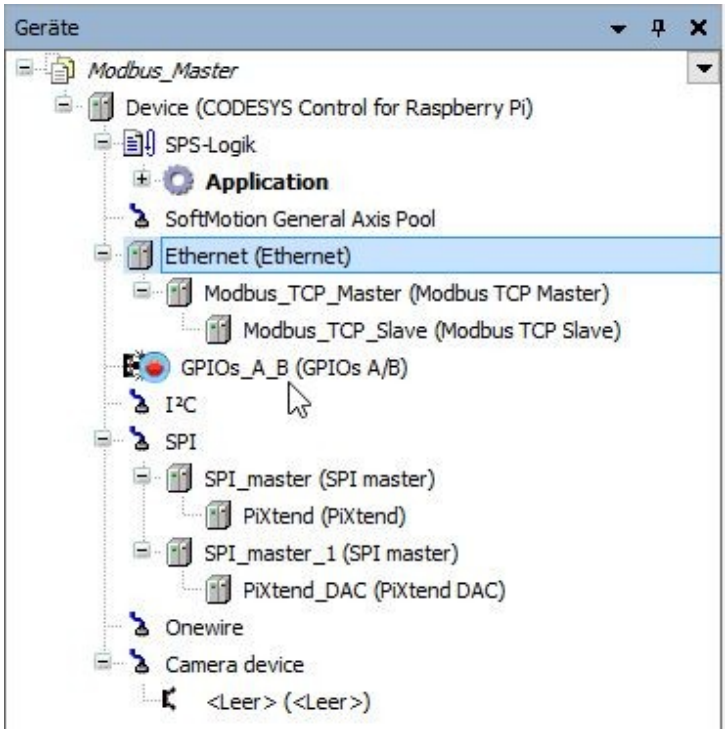


Abbildung 3: Gerätebaum

Bevor die Geräte-Parameter eingestellt werden, legen wir zuerst die Variablen an, die wir für das Auslesen der digitalen Ein- und Ausgänge brauchen. Diese werden in die GVL (Globale Variablen Liste) geschrieben.

Rechtsklick auf *Application → Objekt hinzufügen → Globale Variablenliste*

Evtl. haben Sie die GVL bei der Vorbereitung der beiden Projekte bereits angelegt. In diesem Fall können wir die Ein- und Ausgänge einfach zur bisherigen GVL hinzufügen. Die folgenden Variablen können Sie einfach per Copy-Paste aus diesem Dokument in CODESYS übernehmen:

//Eingänge

xDI0_Slave1 : BOOL;
xDI1_Slave1 : BOOL;
xDI2_Slave1 : BOOL;
xDI3_Slave1 : BOOL;
xDI4_Slave1 : BOOL;
xDI5_Slave1 : BOOL;
xDI6_Slave1 : BOOL;
xDI7_Slave1 : BOOL;

//Ausgänge

xDO0_Slave1 : BOOL;
xDO1_Slave1 : BOOL;
xDO2_Slave1 : BOOL;
xDO3_Slave1 : BOOL;
xDO4_Slave1 : BOOL;
xDO5_Slave1 : BOOL;

Nach dem Einfügen der Ein- und Ausgangsvariablen sollte die GVL in etwa so aussehen:

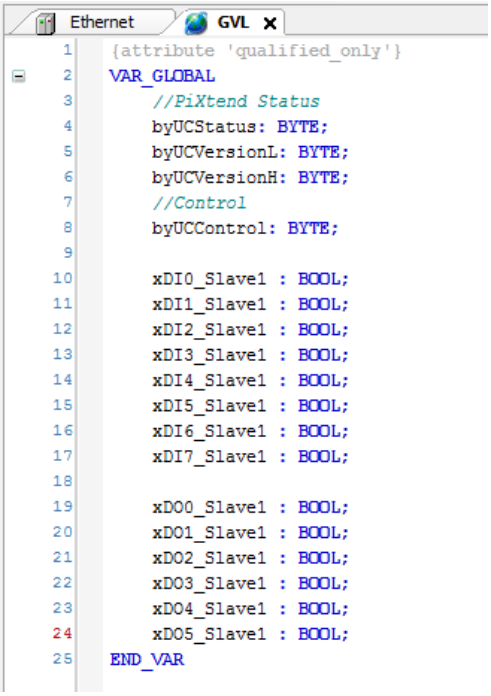


Abbildung 4: GVL Liste erweitert

Im nächsten Schritt müssen die Parameter des Slaves anpasst werden. Zu den Einstellungen kommen Sie mit einem Doppelklick auf die *Modbus_TCP_Slave* Device. Es öffnet sich folgendes Menü:

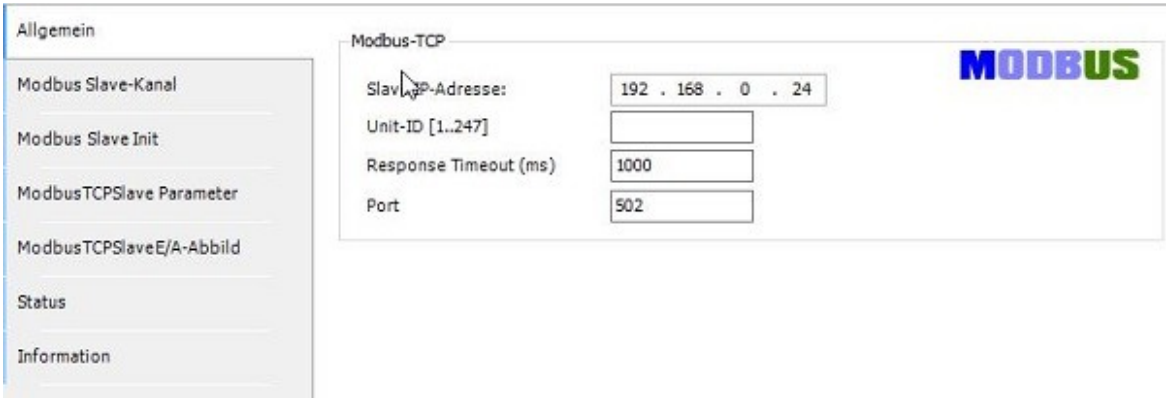


Abbildung 5: Modbus-TCP-Einstellungen

Im Reiter *Allgemein* wird die IP-Adresse des Raspberry Pi eingetragen, welcher der Slave sein wird. Der Port ist schon eingetragen. Der Port 502 wird standardmäßig für den Modbus TCP verwendet.

Im nächsten Reiter *Modbus Slave-Kanal* werden nun zwei Kanäle angelegt. Einen um die Eingänge des Slaves zu lesen und den anderen um dessen Ausgänge zu setzen. Rechts unten im Fenster kann über die Taste *Kanal hinzufügen...* ein neuer Kanal eingefügt werden. Dabei wird folgendes Fenster geöffnet:

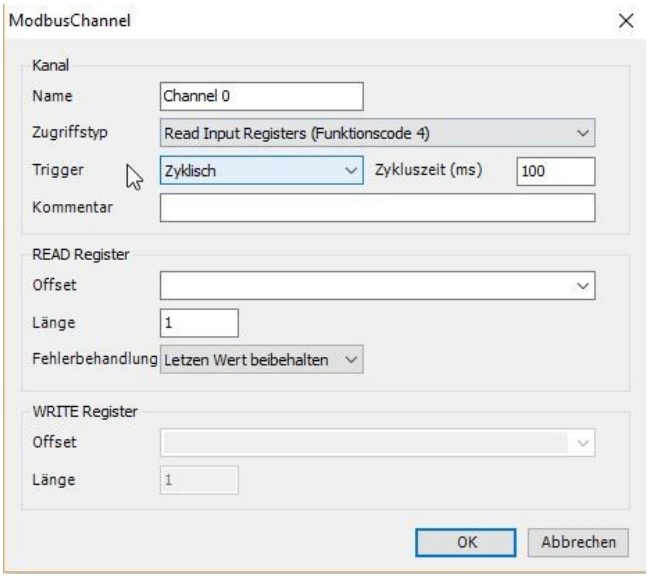


Abbildung 6: Kanal Einstellungen

Hier muss nur der gewünschte Name des Kanals ausgewählt und bei *Zugriffstyp → Read Input Registers (Funktionscode 4)* ausgewählt werden. Dieser Kanal dient dem Lesen der digitalen Eingänge des Slaves. Mit OK Bestätigen.

Nun fügen wir einen weiteren Kanal ein, in dem wir den ganzen Vorgang wiederholen. Nur wird dieses Mal beim *Zugriffstyp → Write Multiple Registers (Funktionscode 16)* ausgewählt.

Es sind jetzt zwei Kanäle im Reiter *Modbus Slave-Kanal* sichtbar:

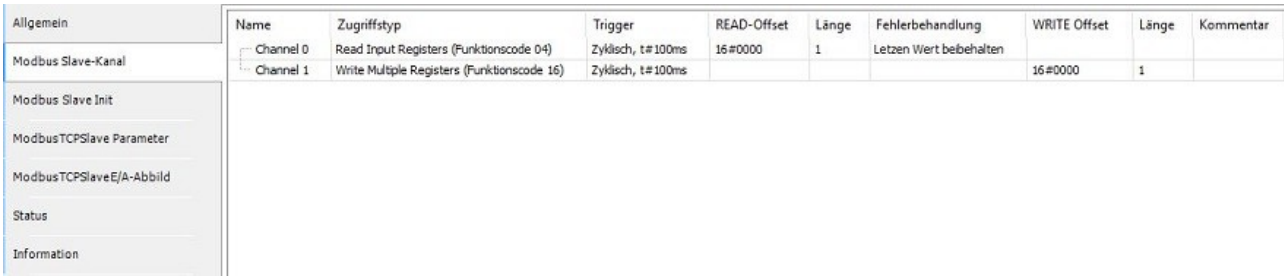


Abbildung 7: Kanal Einstellungen fertig

Anschließend wird auf den Reiter *ModbusTCPSlave E/A-Abbild* gewechselt. Dort sind die zuvor deklarierten Variablen einzutragen. Abbildung 8 zeigt wie es aussehen sollte wenn Sie fertig sind:

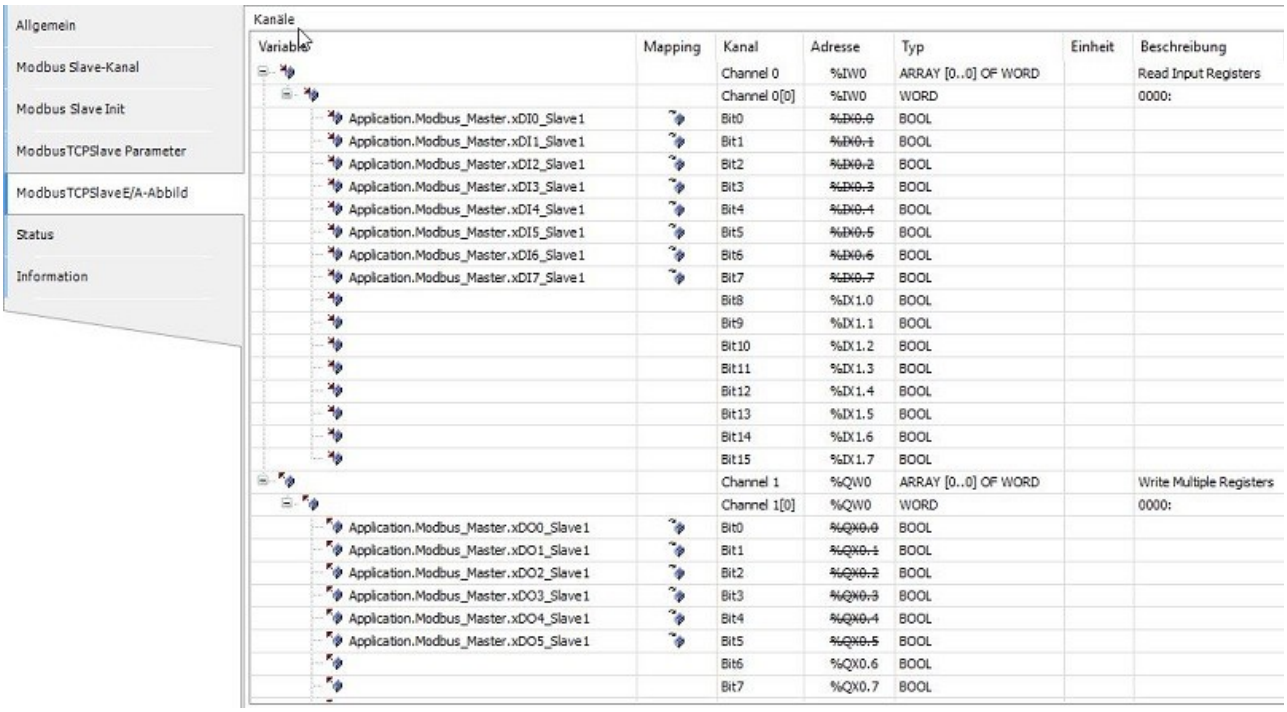


Abbildung 8: E/A-Abbild fertig eingestellt

Die digitalen Eingänge werden bei Channel 0 (Read Inputs Registers) eingetragen, die digitalen Ausgänge bei *Channel 1 (Write Multiple Register)*.

Rechts unten im Reiter *ModbusTCPE/A-Abbild* stellen Sie bei Variablen aktualisieren folgende Einstellung im Drop-Down-Menü ein: *Aktiviert 1, Buszyklus-Task*.

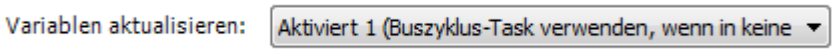


Abbildung 9: Buszyklus aktivieren

Wichtig: Ohne diese Einstellung findet später keine Datenübertragung statt!

Eine Einstellung nehmen wir nun noch am *Modbus_TCP_Master* Gerät im Reiter *Allgemein* vor:

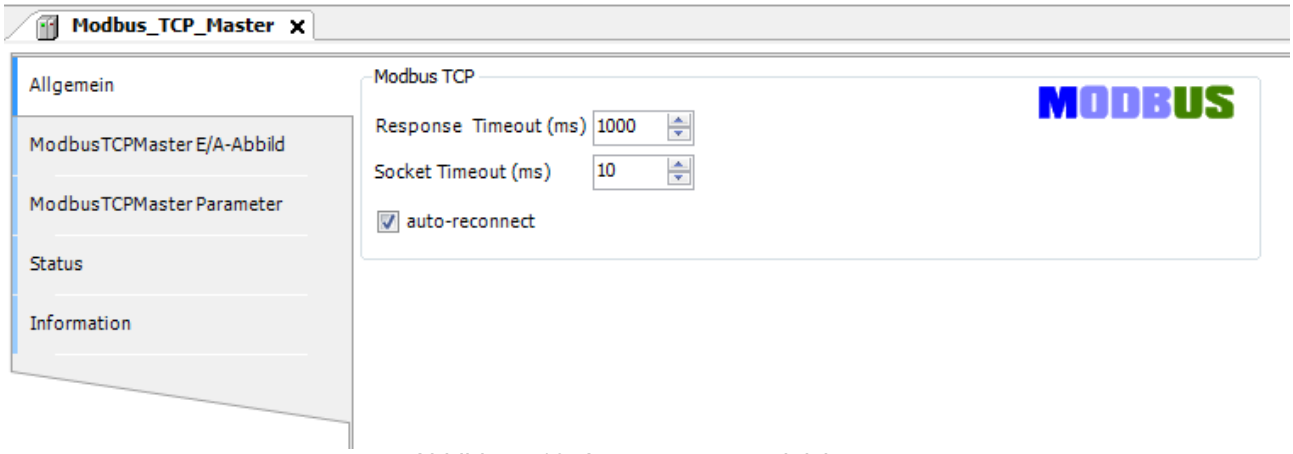


Abbildung 10: Auto-reconnect aktivieren

Die CheckBox *auto-reconnect* sollte aktiviert werden. Ansonsten stellt der Modbus-Master nach einem Bus-Error den Betrieb ein. Ein solcher Bus-Error kann schon dann auftreten, wenn der Slave innerhalb der *Response Time* nicht antwortet (das passiert z.B. wenn der Slave einen Reboot macht oder wir ein neues Programm aufspielen).

Damit ist das Master-Programm nahezu abgeschlossen.

Zum Schluss erstellen wir noch eine kleine Visualisierung, damit wir später die Eingänge des Slaves anzeigen und dessen Ausgänge komfortabel schalten können.

Falls Sie noch keine Visualisierung angelegt haben, so können Sie dies nun folgendermaßen tun:

Rechtsklick auf *Application* → *Objekt hinzufügen* → *VisualizationManager*

Der *Visualizationmanager* legt automatisch ein weiteres Objekt, die *WebVisu*, an. Nun aber noch die eigentliche Visualisierung:

Rechtsklick auf *Application* → *Objekt hinzufügen* → *Visualization*

Damit die gerade erstellte Visualisierung auch als Webvisu angezeigt wird, klicken Sie im Gerätebaum Auf *WebVisu* und tragen im obersten Feld „*Startvisualisierung:*“ den Namen Ihrer Visualisierung ein (Standard-Name ist: *Visualization*).

Abbildung 11: Visualisierung

Um die Eingänge anzuzeigen wird z.B. das Element *Lampe* verwendet und um die Ausgänge zu schalten das *CheckBox* Element. Jedem Element wird eine entsprechende Variable zugewiesen. Siehe Abbildung 12:

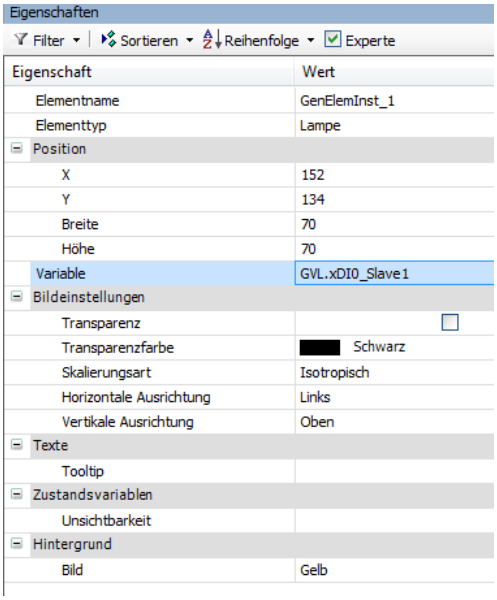


Abbildung 12: Variable einer Lampe zuweisen

Gleichermaßen wird auch mit den CheckBox-Elementen verfahren. Hier werden die Ausgänge *DO 0* (*xDO0_Slave1*) bis *DO 5* (*xDO5_Slave1*) als Variablen eingetragen.

Somit ist das Programm für den Modbus-TCP Master fertiggestellt und wir können mit dem Slave Programm weitermachen.

Tipp: Die Texte wie „Ausgänge“, „Eingänge“ und die Beschriftungen der Lampen können wir mit dem Element *Textfeld* hinzufügen. Das Textfeld hat standardmäßig einen dünnen Rahmen um den Text. Um diesen Rahmen zu entfernen werfen Sie einen Blick in das Eigenschaften-Fenster des Textfelds. Unter *Aussehen* kann die *Linienart* auf *Unsichtbar* eingestellt werden. Der Rahmen wir dann nicht länger angezeigt.

3. Modbus Slave Programm

Wie im Master Programm muss auch hier zunächst ein *Ethernet Adapter* eingefügt werden. An den *Ethernet Adapter* wird hier allerdings nur eine *ModbusTCP Slave Device* angehängt:

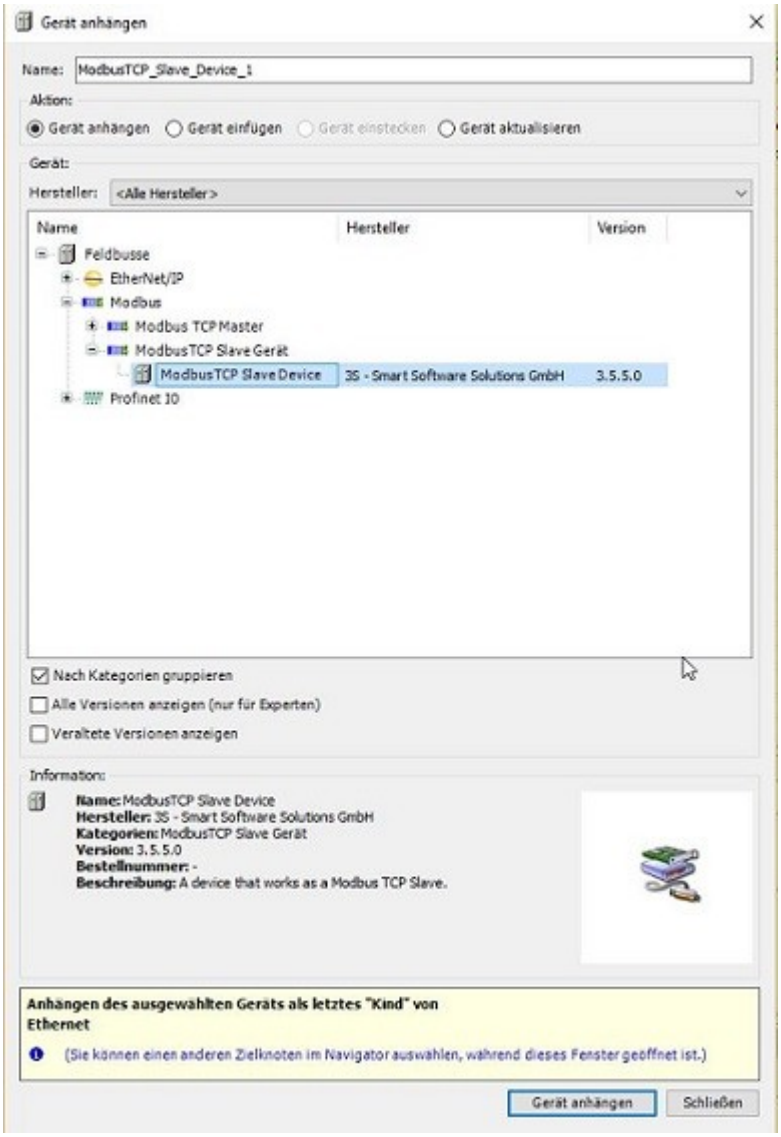


Abbildung 13: ModbusTCP Slave einfügen

Wir legen in diesem einfachen Beispiel keinen Programmcode an, sondern verbinden die Modbus-Daten direkt mit den PiXtend Ein- und Ausgängen des Slave-Gerätes.

Der Einfachheit halber benutzen wir im Slave Programm die gleichen Variablennamen für die Ein- und Ausgänge, wie wir sie bereits im Master Programm genannt haben. Das müsste aber nicht zwingend so sein.

Die GVL wird genau so angelegt bzw. erweitert, wie im Master Programm:

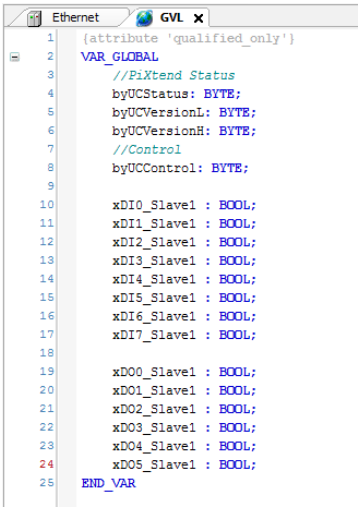


Abbildung 14: GVL des Slave

Nun kommen wir zum PiXtend-Device. Hier weisen wir die Eingänge und Ausgänge entsprechend zu, wie Abbildung 15 zeigt.

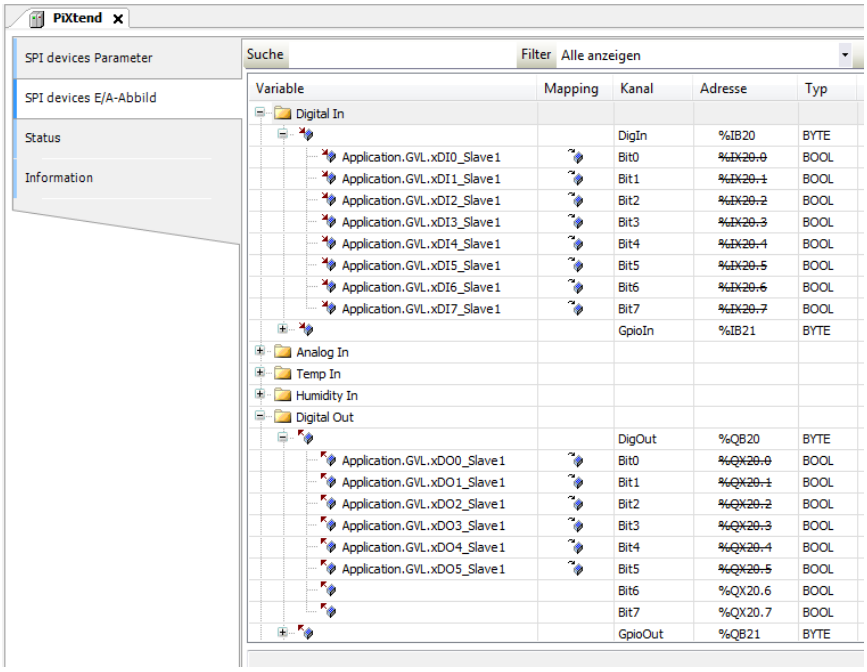


Abbildung 15: PiXtend-Device - E/A Konfiguration

Zu guter Letzt widmen wir uns dem Modbus TCP Slave Device.

Wichtig: Was auf der Master-Seite ein Ausgang ist, hier auf der Slave-Seite ein Eingang. Lassen Sie sich dadurch bitte nicht verwirren.

Wir konfigurieren also die DOs (Ausgänge) auf das erste Eingangsbyte unseres Slave.

Ausgänge:

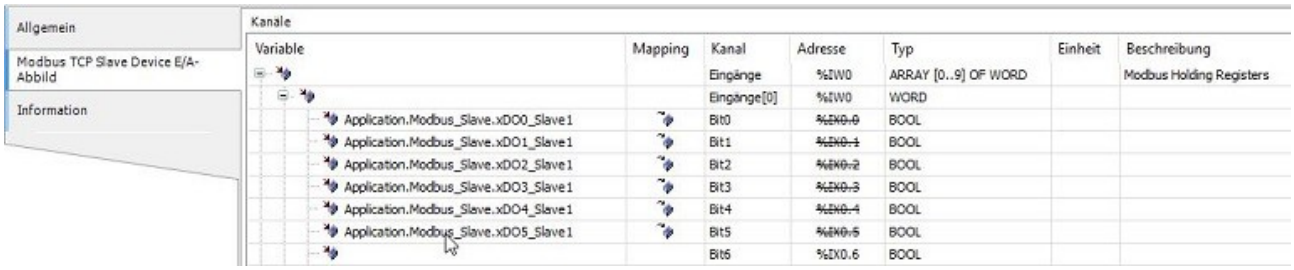


Abbildung 16: ModbusTCP Slave Programmcode

Die DIs (Eingänge) legen wir auf das erste Ausgangsbyte des Slave.

Eingänge:

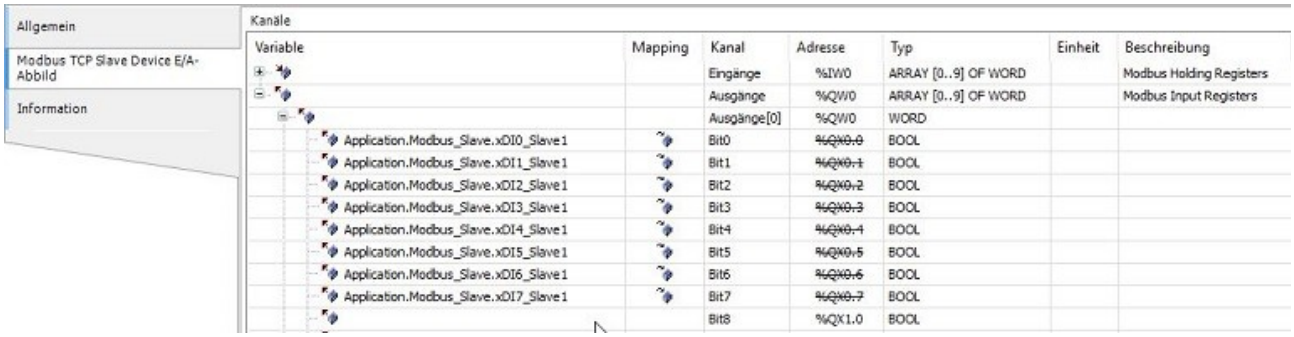


Abbildung 17: ModbusTCP Slave Programmcode

Im E/A-Abbild wird auch hier die *Buszyklus* und *Variablen aktualisieren* Einstellung vorgenommen, da ansonsten keine Daten ausgetauscht werden:

Abbildung 18: Buszyklus und Variablen Aktualisierung beim Slave

Der Bus-Zyklus wird auf Ihren Standard-Task (*MainTask*) eingestellt.

3. Inbetriebnahme

Das Modbus Slave Programm ist damit abgeschlossen und nun wird es spannend!

Sie können die Programme nun auf die beiden Geräte aufspielen. CODESYS kann auf dem PC mehrfach geöffnet werden (mehrere Programm-Instanzen), so dass Sie ein Fenster für den Master und eines für den Slave zur gleichen Zeit geöffnet haben können.

Wichtig: Achten Sie darauf, dass Sie das jeweilige Programm auf das richtige Gerät aufspielen. Sie erinnern sich daran, dass wir im Master Programm die IP-Adresse des Slaves angegeben haben. Ein Vertauschen der Geräte würde dazu führen, dass keine Kommunikation zustande kommt.

Es ist eine gute Vorgehensweise den Geräten verschiedene Namen zu geben:

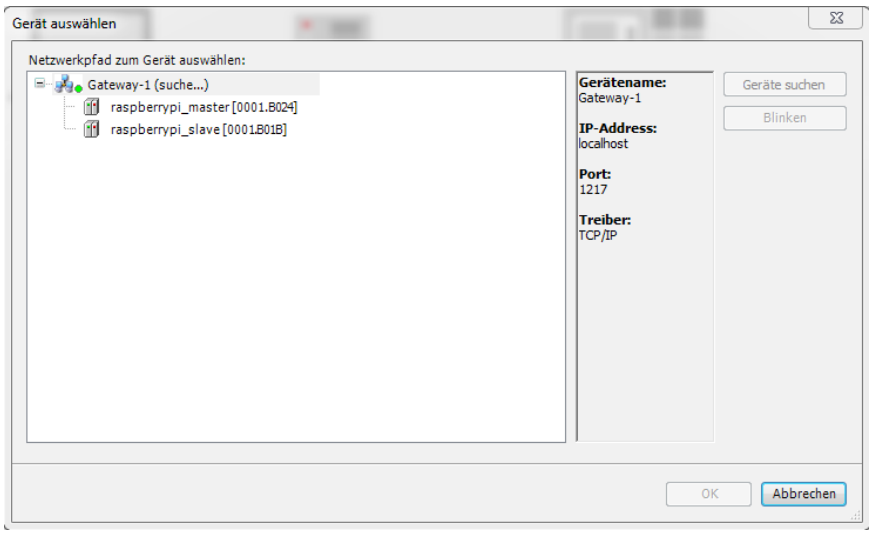


Abbildung 19: Geräte in CODESYS sauber unterscheiden

Ein zuvor ausgewähltes Gerät kann sehr einfach umbenannt werden:

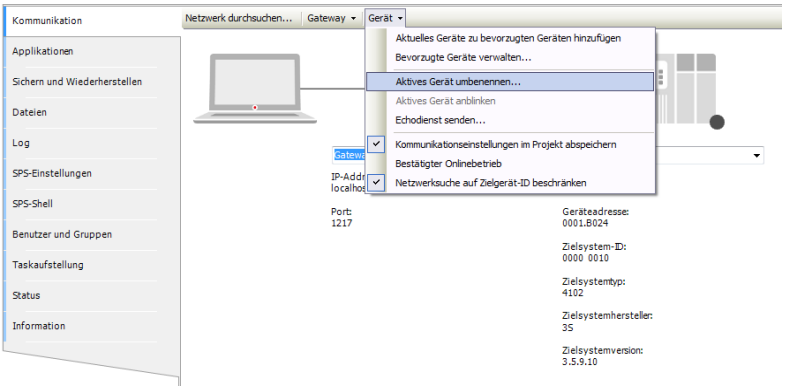


Abbildung 20: Raspberry Pi umbenennen

Wenn Sie keine Fehler in die Programme eingebaut haben, so müssen sowohl auf Master-, als auch auf Slave-Seite die Modbus-Geräte mit einem grünen Symbol gekennzeichnet sein. Dies bedeutet, dass der Bus fehlerfrei läuft.

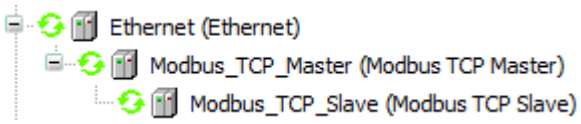


Abbildung 21: Modbus läuft

Sie können auf die Webvisu des Masters zugreifen und dort die Ausgänge des Slaves manipulieren. Ebenso werden Ihnen mit den Lampen die Eingänge des Slaves angezeigt.

Es handelt sich hier um ein sehr einfaches Beispiel. Natürlich können Sie Modbus TCP auch zur Übertragung von weitaus mehr Daten verwenden. Es müssen auch nicht nur I/O-Daten sein. Sie könnten beliebige Daten übertragen.

Auch können Sie noch weitere Modbus-Geräte (Slaves) ansprechen. Es gibt eine Vielzahl an Geräten und E/A-Erweiterungen für Modbus TCP.

Wenn es mal nicht funktioniert:

Wir laden Sie zu einer Diskussion und dem Erfahrungsaustausch in unserem Forum (http://www.pixtend.de/forum/) ein. Sollte Ihnen hier nicht innerhalb von 2 Tagen geholfen werden, so wenden Sie sich bitte an unseren Support: support@pixtend.de