



Applikationsbeispiel

Beispiele für verschiedene Zugriffe auf Modbus Register (Konfiguration über ABC Config Tool)

Haftungsausschluß

Die Schaltungen in diesem Dokument werden zu Amateurzwecken und ohne Rücksicht auf die Patentlage mitgeteilt. Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt.

Wir haben den Inhalt dieses Dokumentes auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Dennoch können Abweichungen nicht ausgeschlossen werden, so dass wir für die vollständige Übereinstimmung keine Gewähr übernehmen. Die Angaben in diesem Dokument werden jedoch regelmäßig überprüft. Notwendige Korrekturen sind in den nachfolgenden Versionen erhalten. Für Verbesserungsvorschläge sind wir dankbar.

© Copyright by HMS GmbH. All rights reserved.

Hinweis: Dieses Dokument ersetzt nicht die offiziellen Handbücher und Dokumentationen, die in den aktuellsten Versionen unter www.anybus.de zur Verfügung stehen.

Erstellt	Version	Name	Kommentar
07.2010	0.1	DOW	Erstentwurf
08.2010	1.0	DOW	kleinere Korrekturen

Inhalt:

1. ZUSAMMENFASSUNG.....	3
2. VORBEREITUNG	3
3. ERKLÄRUNG DER BEISPIELKONFIGURATION	4
3.1 STANDARDFUNKTION	4
3.2 TRANSPARENTES ÜBERGEBEN DER ZU LESENDEN ADRESSE	5
3.3 MASKIEREN DER GELESENEN WERTE.....	6
4. ANHANG	6
4.1 WEITERE DOKUMENTE.....	7

1. Zusammenfassung

Dieses Dokument soll einen Überblick über die Anwendung der vorbereiteten Modbus-Befehle des ABC Config Tools unter verschiedenen Voraussetzungen geben.

Der Communicator kann als Modbus-RTU-Master bis zu 50 Befehle ausführen (z.B. 20x Read Input Registers + 30x Preset Single Register). In den meisten Fällen ist dies vollkommen ausreichend.

Werden diese Befehle mit Standardaufbau und -funktionalität verwendet, so sind die Adressen der zu bearbeitenden Register sowie der Umfang der übergebenen Daten immer gleich. Dies wird im ersten Beispiel beschrieben.

Wenn die maximal 50 Befehle nicht ausreichen, um alle gewünschten Register bzw. Slaves ansprechen zu können, bietet das Konfigurationstool die Möglichkeit den Aufbau der vorgefertigten Befehle zu modifizieren. Solche Modifikationen dienen der Reduzierung der Anzahl benötigter Befehle oder ermöglichen eine flexiblere Nutzung des Communicators, wenn z.B. die benötigte Funktionalität nicht von vorne herein feststeht.

Zwei mögliche Modifikationen werden an den Beispielen „Transparente Übergabe der zu lesenden Adresse“ sowie „Maskieren der gelesenen Werte“ aufgezeigt.

Alle Beispiele dieses Dokumentes werden auf Basis des Befehls „Read Holding Registers“ (Funktionscode 0x03) erläutert. Sie können jedoch auch auf andere Befehle adaptiert werden.

2. Vorbereitung

Im Rahmen dieses Dokumentes wird davon ausgegangen, dass die Verkabelung sowie die Konfiguration der seriellen Schnittstelle bereits vorgenommen wurden.

Weitere Hinweise zum Vorgehen an dieser Stelle entnehmen Sie bitte der Dokumentation zu Ihrem Anybus Communicator.

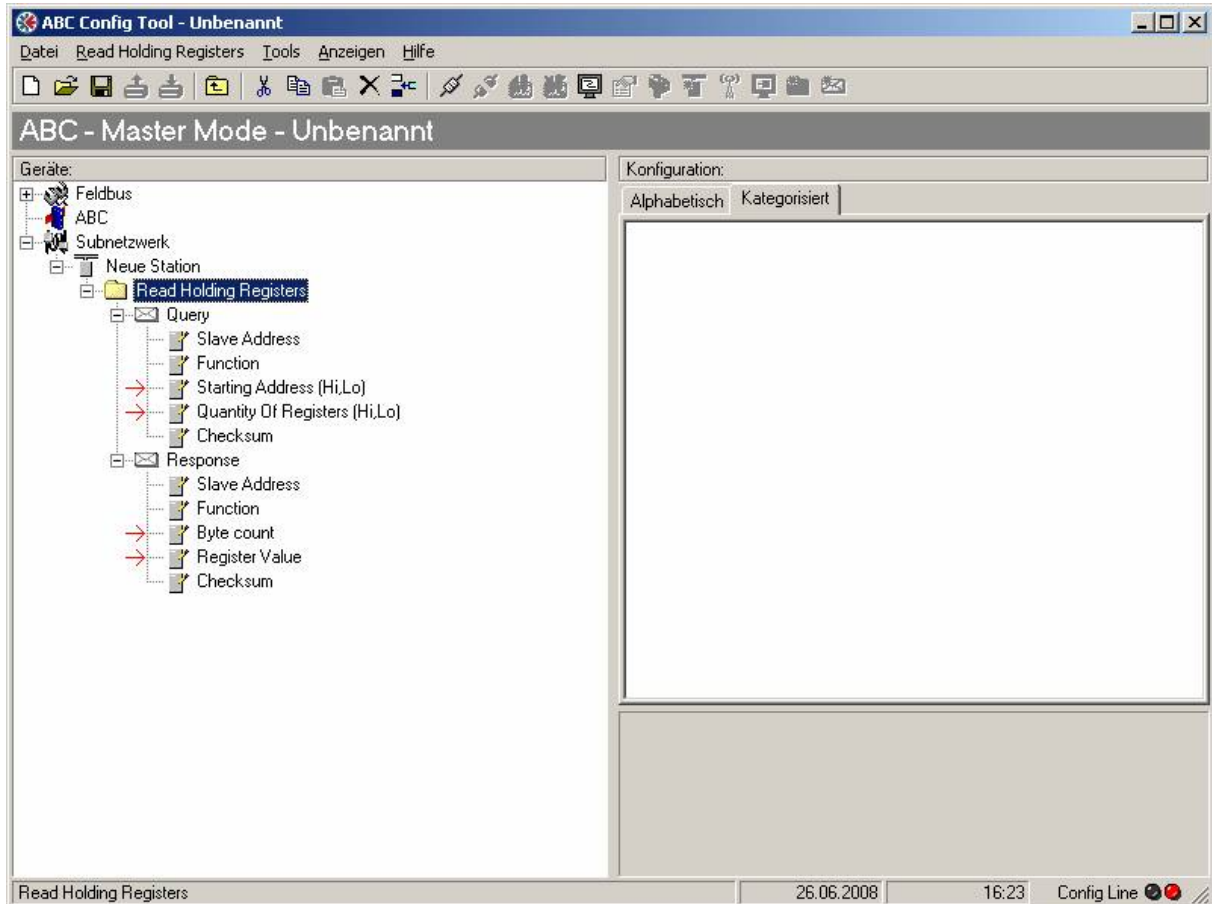
Die Adresse des Modbus-Slaves wird im ABC Config Tool unter dem Eintrag „Neue Station“ eingestellt. In diesem Beispiel wurde sie auf dem Standardwert 1 belassen.

3. Erklärung der Beispielkonfiguration

3.1 Standardfunktion

Wenn in die bisher leere Konfiguration ein „Read Holding Register“-Befehl eingefügt wird, erhält man folgendes Bild:

(Das Einfügen eines Befehls geschieht über einen Rechtsklick auf die „Neue Station“ und die Auswahl des Punktes „Befehl einfügen“.)



Die mit Pfeil markierten Teile müssen manuell angepasst werden um eine Standard-Funktion zu gewährleisten.

Dabei werden die Parameter

- „Starting Address“ (z.B.: =200=0xC8),
- „Quantity Of Registers“ (z.B.: =10=0xA) und
- „Byte count“ (z.B.: =20=0x14)

auf feste Werte gesetzt. Diese ergeben sich anhand der Dokumentation des Modbus-Slaves.

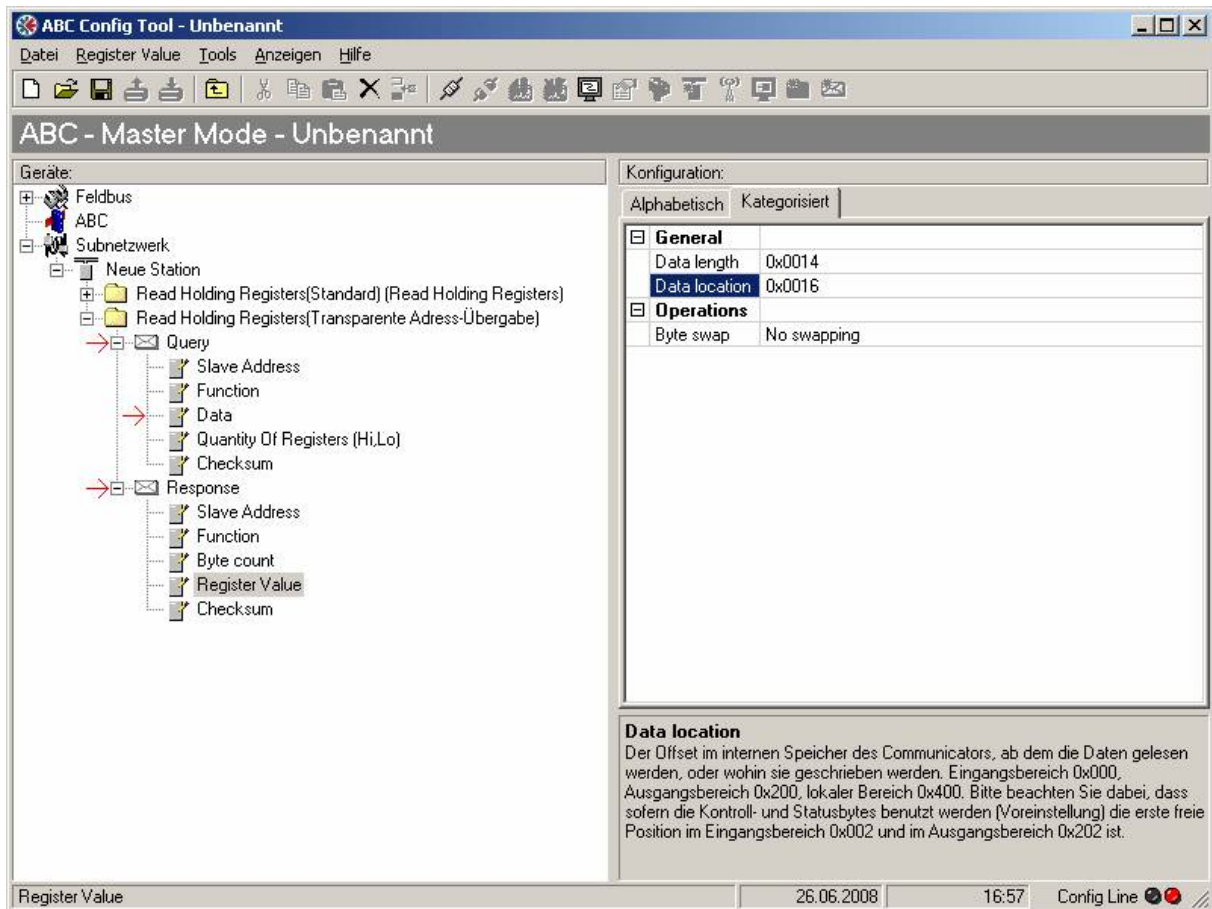
Die Eingabe der Werte kann entweder dezimal oder hexadezimal (mit vorgestelltem 0x) erfolgen.

Dabei ist „Byte count“ auf den doppelten Wert von „Quantity Of Registers“ zu setzen, da ein Modbus-Register 2 Byte hat.

„Register Value“ repräsentiert die eigentlichen Daten. Hier ist die Länge gleich dem „Byte count“ zu setzen“. Die Adresse ist gemäß dem gewünschten Speicher-Mapping zu wählen, wobei 0x0000 den Beginn des Datenblocks darstellt, der zur SPS übertragen wird (Eingangsbereich des übergeordneten Masters).

Bei dieser Art der Konfiguration ist die Anordnung der Daten auf der übergeordneten Steuerung immer gleich. Daher bedeutet die Auswertung den minimalen Programmieraufwand für den SPS-Programmierer.

3.2 Transparentes Übergeben der zu lesenden Adresse



Um mit nur einem konfigurierten Befehl mehrere verschiedene Register lesen zu können, werden bei dem zweiten „Read Holding Register“-Befehl die mit Pfeil markierten Teile anders gehandhabt als bei einer Standard-Transaktion.

Der „Update Mode“ der Query wird auf „Change of state on trigger“ gesetzt. Die „Trigger byte address“ wird in den Bereich ab 0x0200 gemappt (im Beispiel 0x202). Dies ist der Bereich, der von der SPS beschrieben wird (Ausgangsbereich des übergeordneten Masters). Dadurch wird der Befehl erst gesendet, wenn der Inhalt des Trigger-Bytes geändert wird.

Bei der Response wird das „Trigger byte“ auf Enabled gesetzt und seine Adresse in den Eingangsbereich der Steuerung gemappt (im Beispiel 0x0015). Wenn die Antwort vom Communicator empfangen wird, wird das Trigger-Byte vom Communicator manipuliert. Daher kann die Steuerung erkennen, dass der Befehl ausgeführt wurde und sie den nächsten senden kann.

Die „Starting Address“ wird gelöscht und durch ein Daten-Objekt ersetzt. Dieses Daten-Objekt hat eine Länge von zwei Byte und wird in den Ausgangsbereich der Steuerung gemappt, so dass dieser Wert vom Feldbus-Master geschrieben wird (im Beispiel 0x0200).

Auf diese Weise kann ein Befehl sehr flexibel eingesetzt werden, da jede Modbus-Registeradresse gelesen werden kann. Auf der anderen Seite bedeutet dieses Vorgehen einen Mehraufwand für den SPS-Programmierer, da der Communicator ohne Trigger der SPS nichts tut.

Anmerkung: Wenn man Objekte aus dem automatisch generierten Modbus-Befehl entfernt oder welche hinzufügt, wird der Befehl vom ABC Config Tool nicht mehr als solcher gekennzeichnet (In unserem Beispiel verschwindet das (Read Holding Registers) hinter dem vergebenen Namen des Befehls.). Dies tut der Funktionalität bei korrektem Vorgehen jedoch keinen Abbruch.

3.3 Maskieren der gelesenen Werte

Eine weitere Möglichkeit, die Anzahl benötigter Befehle zu reduzieren, ist das Ausmaskieren der Antwort. Dazu muss der Slave das Lesen mehrerer Register mit einem Befehl unterstützen.

Bei einem weiteren eingefügten Befehl wird die Query verwendet wie im Standard. Es werden aber 10 Register auf einmal gelesen. Von diesen sind nur das erste, das fünfte und sechste sowie das neunte und zehnte interessant.

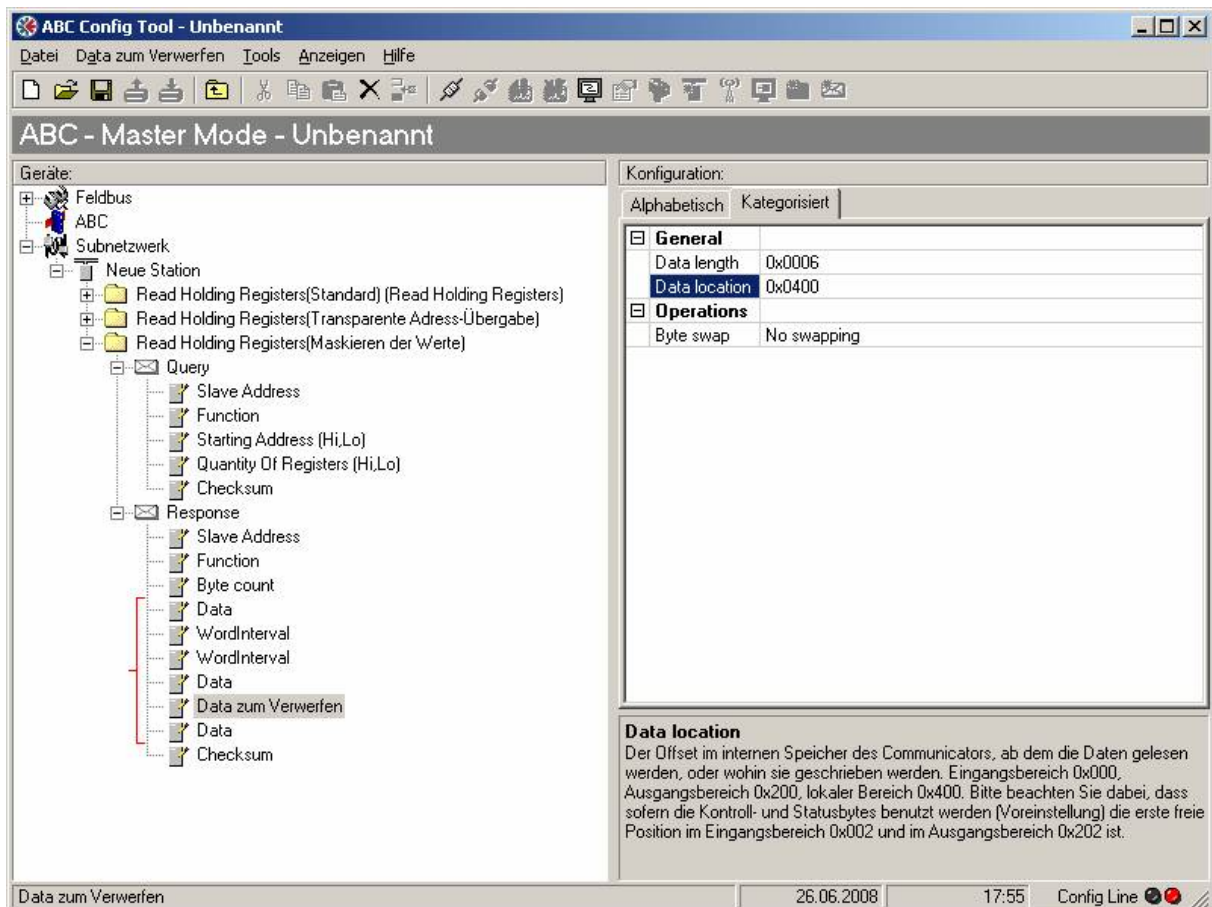
Um die überflüssigen Informationen nicht über den Feldbus übertragen zu müssen, wird die Response manipuliert.

Das Objekt „Register Value“ wird durch eine Gruppe von Objekten ersetzt. Die interessanten Register werden durch Datenobjekte abgebildet. Die restlichen Register können auf zwei Wege maskiert werden.

Eine Möglichkeit ist die Verwendung eines Wortes mit Grenzen (dies wird über den Punkt „Wort einfügen, Grenzen“ eingefügt). Diese Worte werden beim Prüfen des Telegramms als gültig erkannt, wenn ihr Wert sich innerhalb der Grenzen befindet. Daher ist die untere Grenze auf 0x0000 und die obere Grenze auf 0xFFFF gesetzt.

Für größere Bereiche, die ausmaskiert werden sollen kann auch ein Datenobjekt verwendet werden, das in den Bereich von 0x0400 aufwärts gemappt wird. Dieser ist ein interner Bereich des Communicators, der nicht zum Bus übertragen wird.

Das Ganze könnte dann so aussehen:



Beachtet werden muss hierbei aber, dass die Gesamtlänge der Daten und Maskierungsobjekte (Rote Markierung in der Grafik) genau der erwarteten Datenmenge entspricht (Byte count).

Der Vorteil dieses Vorgehens liegt darin, dass der Communicator sich aus Sicht der SPS verhält, als wäre er mit Standard-Befehlen konfiguriert. „Erkauft“ wird dieser Vorteil durch die geringere Flexibilität gegenüber der transparenten Übergabe der Registeradresse.

4. Anhang

4.1 weitere Dokumente

Die kompletten User Manuals zur Anybus Communicator Familie finden Sie im Supportbereich unserer Webseite (http://www.anybus.de/support/support_home.shtml) unter Downloads.

Außerdem können sie im Bereich Entwicklungsunterstützung weitere ergänzende Dokumente finden. Diese ersetzen NICHT die User-Manuals!!

Neben der Dokumentation zum Communicator benötigen Sie auf jeden Fall eine Dokumentation, die Informationen zur Kommunikationsschnittstelle Ihres seriellen Gerätes enthält.