Beispieldokumentation

Deutsche Beschreibung

BEISPIELBESCHREIBUNG

Inhalt
Dieses Beispiel basiert auf dem allgemeinen Modbus-TCP-Beispiel und konfiguriert ein Energieerfassungs-gerät (UMD96) und liest dessen Messwerte aus.
Die Messwerte werden anschließend umformatiert, um zur bestehenden INSEVIS-Energiemessbaugruppe kompatibel zu sein.

Das Modbus-TCP-Interface des UMD96
Das Modbus-TCP Interface des UMD96 bildet alle Prozessdaten als Fließpunktzahlen (32 oder 64 Bit) in Input-Registern paarweise bzw. in 4er-Gruppen ab. Die Konfigurationsdaten werden in Holding-Register abgebildet.

S7-Programm
FB1 (aus dem allgemeinen Modbus-TCP-Beispiel) dient als Modbus-TCP client Treiberbaustein und nutzt die Systembausteine zum Senden und Empfangen über TCP/IP und bleibt **unverändert**.
Als Parameter werden VerbindungsID-Nummer, Knotennummer (UID), Modbus- Kommando (function code 1, 2, 3, 4, 6, 15 oder 16) und Nutzdatenpointer übergeben.

```
CALL  "ModbusTCP_Client" , "tcp"      // FB1 mit DB1 als Instanz-DB
  R         :=FALSE                   // Reset-Eingang, einmalig nach Hochlauf setzen
  ConnectID:=1                        //  VerbindungsID-Nummer aus ConfigStage
  UID       :=                        //  obsolete Knotennummer aus ModbusRTU
  CMD       :=B#16#2                  // Modbus Kommando (funktion code 1,2,3,4,6,15,16)
  Index     :=0                       // Register bzw. Bit-Adresse (0...65535)
  LEN       :=256          // Anzahl zu übertragender Register (1..125) bzw. Bits (1..2000)
  DATA      :="Daten_TCP".Inputs      // ANY-Pointer auf Nutzdatenbereich
  DONE      :="tcp1_done"             // TRUE wenn erfolgreich
  ERROR     :="tcp1_error"            // TRUE bei Fehler
  ErrSrc    :="tcp1_ErrorSrc"         // Fehlercode s. Tabelle
  ErrStatus:="tcp1_ErrCode"
```

Der Aufruf muss wiederholt werden, bis DONE oder ERROR zurückgegeben wird.
Die Längenangabe im Pointer DATA wird zum Kopieren der Nutzdaten benutzt und muss zu den Datenpaketen passen (Beim Senden wird sonst ggf. der Sendepuffer unvollständig gefüllt oder andere Werte überschrieben. Beim Empfang werden ggf. andere Nutzdaten überschrieben)
Alle lokalen Variablen des Kommunikationstreiberbausteins FB1 sowie Sende- und Empfangsdaten liegen in dem zugehörigen Instanzdatenbaustein.

FC1 stellt die Modbus-Kundenapplikation dar und **wurde** wie folgt **angepasst:**
Da die relevanten Prozessdaten verstreut liegen, erfolgen mehrere Kommunikationszyklen auf Datengruppen.
Im NW3 wird (einmalig nach dem Hochlauf) die Konfiguration über function code 10_{hex} geschrieben.
Im NW4 werden Statusregister gelesen, in NW5 die Spannungswerte L1-L3, im NW6 die Stromwerte L1-LN, in NM7 die Leistungsfaktoren, in NW8 die Wirk- und Scheinleistungen und in NW9 die Energiezähler.
In NW10 wird der Sprungverteiler auf 1 zurückgesetzt, um den Reset-State nicht zyklisch auszuführen. Der Setup-State wird über Bit M 96.0 verriegelt und kann bei Bedarf aktiviert werden.

FC11 formatiert die erfassten Messwerte um:
- alle 32-Bit-REAL-Daten werden in das spezielle Integer-Format der INSEVIS-Energiemessbaugruppe konvertiert (1 Dezimalstelle)
- die relevanten Energiezähler werden über einen Bibliotheksbaustein von 64 Bit REAL in 32 Bit Integer konvertiert und danach auf 1 Dezimalstelle umgerechnet. Dabei verringert sich der nutzbare Wertebereich um Faktor 100.
- Da das UMD96 nur Blind und Wirkarbeit bereitstellt, wird die Scheinarbeit berechnet.

Hinweise
Programmvorbereitung:
SimaticManager:
In der Hardwarekonfiguration eine S7-300-2PNDP anlegen und die
IP-Adresse der SPS unter PN-IO einstellen,
das S7-Programm aus dem Beispielprojekt einfügen
TIA-Portal:
In der Hardwarekonfiguration eine S7-300-2PNDP anlegen und die IP-Adresse der SPS einstellen,
AWL-Quelldatei aus dem Beispielverzeichnis importieren

IP-Adressen:
Per default ist das UMD96 auf DHCP konfiguriert, die automatisch zugewiesene IP-Adresse kann über Parameter 15 einfach abgefragt werden und muss dann im ConfigStage-Projekt nachgetragen werden. Für Testzwecke ist das ok, aber für feste Implementierungen könnte das problematisch werden.

RÜCKMELDUNGEN
Möchten Sie Erweiterungswünsche oder Fehler zu diesen Beispielen melden oder wollen Sie anderen eigene Bespielprogramme kostenlos zur Verfügung stellen? ***Bitte informieren Sie uns unter info@insevis.de***
Gern werden Ihre Programme -auf Wunsch mit Benennung des Autors- allen INSEVIS- Kunden zur Verfügung gestellt.

English description

TERMS OF USE
The use of this sample programs is allowed only under acceptance of following conditions by the user:
The present software which is for guidance only aims at providing customers with sampling information regarding their S7-programs in order to save time. As a result, INSEVIS shall not be held liable for any direct, indirect or consequential damages respect to any claims arising from the content of such software and/or the use made by customers of this sampling information contained herin in connection with their own programs.

SAMPLE DESCRIPTION

Abstract
This example transfers the generic Modbus TCP client demo to the energy measurement device UMD96. After reading measurement data are converted to be compatible to the structure of INSEVIS EMESS.

The Modbus-TCP-Interface of UMD96
The Modbus-TCP Interface maps all measurement process data (32 or 64 bit real) into pairs or quadruples of input register. Configuration data are mapped into holding register.

S7-program
FB1 works as Modbus-TCP client driver and handles the system calls for send and receive via TCP/IP and is **no to change**.
It uses the parameters connection ID-number, node-number (UID), Modbus-command (function code 1, 2, 3, 4, 6, 15 or 16) and payload data pointer.

```
CALL  "ModbusTCP_Client" , "tcp"      // FB1 + DB1 as Instance-DB
  R         :=FALSE                   // Reset-input, to set once while start up
  ConnectID:=1                        //  connection ID-number from ConfigStage
  UID       :=                        //  obsolete node number from ModbusRTU
```

```
CMD       :=B#16#2                    // Modbus command (function code 1,2,3,4,6,15,16)
Index     :=0                         // Register rsp. Bit-address (0...65535)
LEN       :=256                 // number of register (1..125) or bits (1..2000) to transfer
DATA      :="Daten_TCP".Inputs        // ANY-Pointer payload data area
DONE      :="tcp1_done"               // TRUE if well done
ERROR     :="tcp1_error"              // TRUE in case of trouble
ErrSrc    :="tcp1_ErrorSrc"           // error code s. table
ErrStatus:="tcp1_ErrCode"
```

The call of FB1 must be repeated until DONE or ERROR are returned.
The length information of Pointer DATA will be used to copy data and must match to the used data packets. (Otherwise sending the buffer could be filled incomplete or other data are overwritten.)
All local data are kept in the instance data block

FC1 represents the Modbus customers application and **was adapted as follows:**
Due to relevant process data are scattered, several communication cycles handles groups of data.
NW 3 writes (once after start up) configuration data via function code 10_{hex}. NW 4 reads status information, NW 5 reads voltages L1-L3, NW 6 reads currents L1-L3, NW 7 reads the power factors, NW 8 active and apparent power values, NW 9 energy meter data. NW 10 resets the state to begin of cyclic work in NW 3.
Writing setup data is locked by bit M 96.0 to be enabled on request.

FC11 does the conversion:
- all 32 bit read data are stored as integer, multiplied by 10 to get 1 decimal digit (EMESS format)
- energy counter are converted by a library function from 64 bit real into 32 bit integer, then scaled to 1/10 kW respectively kVA (EMESS format). Thereby the range of output value is reduced by 100 too.
- due to UMD96 only provides active and reactive energies, apparent energy is calculated

Hints
- Preparation before S7 program download:
  SimaticManager – hardware configuration:
  insert a S7-300-2PNDP and setup IP-address of PLC in PN-IO,
  copy and paste the S7-program of the example project
  TIA-Portal:
  hardware configuration: insert a S7-300-2PNDP and setup IP-address of PLC,
  import the AWL source file from example directory

-IP address issue:
UMD96 is configured by default to DHCP, the obtained IP address can be read by parameter 15 and must be supplemented in ConfigStage project. This is easy for quick start up, but doubtful in a fixed installation.

FEEDBACK
Do you want to inform us about necessary increments or errors or do you want to provide us with your sample programs to offer it for free to all customers? ***Please inform us at info@insevis.de***
Gladly we would provide your program -if you wish with the authors name- to all other customers of INSEVIS.