FB60 - <offline>

"SEND"

Nome: **Famiglia:**
Autore: **Versione:** 2.1
Versione blocco: 2
Data e ora Codice: 16/08/2011 11.42.46
Interfaccia: 30/03/2011 09.05.53
Lunghezze (blocco / codice / dati): 02172 01844 00042

Nome	Tipo di dati	Indirizzo	Valore iniziale	Commento
IN		0.0		
REQ	Bool	0.0	FALSE	Auftragsanstoß bei positiver Flanke
R	Bool	0.1	FALSE	Synchron Reset auslösen
LADDR	Int	2.0	0	logische Basisadresse des CP - entspricht der Adresse aus der HW-Konfiguration
DB_NO	Int	4.0	0	Datenbausteinnummer - Nummer des Sende-DB (>0)
DBB_NO	Int	6.0	0	Datenbytenummer - Sendedaten ab Datenbyte
LEN	Int	8.0	0	Länge des zu sendenen Telegramms in Byte (>0 und <1025)
IO_SIZE	Word	10.0	W#16#0	parametrierte IO Größe des Moduls
OUT		0.0		
DONE	Bool	12.0	FALSE	Auftrag fertig ohne Fehler, Daten gesendet, Parameter STATUS=00h
ERROR	Bool	12.1	FALSE	Auftrag fertig mit Fehler, Parameter STATUS enthält die Fehlerinformation
STATUS	Word	14.0	W#16#0	Spezifikation des Fehlers bei ERROR=1
IN_OUT		0.0		
CONTROL	Byte	16.0	B#16#0	geteilter Speicher: Sendebaustein Bit 0..3, Empfangsbaustein Bit 4-7
STAT		0.0		
bIntStatus	Byte	18.0	B#16#0	interner Bausteinstatus
bAcknowledge	Byte	19.0	B#16#0	Bit 4..7 des Eingangsbyte 0 als Quittung
bFrag	Byte	20.0	B#16#0	Fragmentanzahl
bFragCounter	Byte	21.0	B#16#0	Fragmentzähler
xFpRequest	Bool	22.0	FALSE	Hilfsbit zur Erkennung einer steigenden Flanke an REQ
xFpReset	Bool	22.1	FALSE	Hilfsbit zur Erkennung einer steigenden Flanke an R
xResetDone	Bool	22.2	FALSE	Fertigmeldung Synchron Reset (Statemachine Bit)
iLenTrailer	Int	24.0	0	Trailer Länge = Anzahl Nutzdaten im Trailer
xResetBecauseAbort	Bool	26.0	FALSE	Reset durch fallende Flanke an REQ ausgelöst
xWait	Bool	26.1	FALSE	warte nach fertigem Reset oder Auftrag mit Fehler auf FALS an EN_R und R
wStatusMem	Word	28.0	W#16#0	temporärer Zwischenspeicher für STATUS Ausgang
arInputData	Array [0..59] Of Byte	30.0		
arOutputData	Array [0..59] Of Byte	90.0		
TEMP		0.0		
anyScr	Any	0.0		
anyDst	Any	10.0		
iRetval	Int	20.0		
wTemp1	Word	22.0		
wTemp2	Word	24.0		
dwTemp1	DWord	26.0		
dwTemp2	DWord	30.0		
wLaddr	Word	34.0		

Blocco: FB60 SEND

```
History:
V1.0   erste Version
V1.1   Sprungmarke T030 -> von #bFragCounter werden nur noch die untersten 3 Bit
       in das Control-Wort geschrieben
       Sprungmarke S050 --> Korrektur der Berechnung von wTemp1
V1.2   Übergabe BLKMOV RetVal an Status
       Reset auch bei fehlerhaften Eingangsparametern möglich
       Ausgangsbits stehen nur für einen SPS-Zyklus an
       Status 0x0517 und 0x070A hinzugefügt
V1.3   neuer interner Status ERROR hinzugefügt
V2.0   Prozessdatenzugriff auf SFC14/15 umgestellt
V2.1   Längenkorrektur Destination-Pointer für Block-Move in NW8 beim Schreiben
       vom Header und Header/Trailer kombiniert
```

Segmento: 1 INIT

Sprungmarken Definition:
- Ixxx INIT
- Rxxx SCHREIBE RESET
- Sxxx START
- Txxx SENDEN
- Qxxx TLG QUITTUNG
- Axxx ABSCHLIEßEN
- Exxx ENDE

Der Buchstabe kennzeichnet den Bausteinstatus/Netzwerk. Die x kennzeichnen die Sprungmarkennummer in dezimaler Form.

Über das Byte "byIntStatus" wird der Bausteinstatus (Statemachine) gekennzeichnet:
- 0x10 SCHREIBE RESET (Synchron Reset)
- 0x20 START (neuen Sendeauftrag starten)
- 0x30 SENDEN (Sendeauftrag bearbeiten)
- 0x40 TLG QUITTUNG (Quittierung bearbeiten)
- 0x50 ABSCHLIEßEN (Auftrag abschließen)
- 0x60 ERROR (Sendefehler und Warten auf Quittung)

```
//*********************************************************************************************
// Eingangsdaten konsistent lesen
//*********************************************************************************************
      L     #LADDR                                                      #LADDR            -- logische Basis
                                                                        adresse des CP - entspricht der Adre
                                                                        sse aus der HW-Konfiguration
      T     #wLaddr                                                     #wLaddr

//  DSTBLK zusammenbauen
      LAR1  P##anyDst           // Any Pointer Anfangsadresse
      L     W#16#1002
      T     LW [AR1,P#0.0]      // Byte 0+1: Datentyp Byte
      L     #IO_SIZE                                                    #IO_SIZE          -- parametrierte
                                                                        IO Größe des Moduls
      T     LW [AR1,P#2.0]      // Byte 2+3: Länge
      L     DINO
      T     LW [AR1,P#4.0]      // Byte 4+5: DB-Nummer
      TAR2
      UD    DW#16#FFFFFF
      L     P##arInputData
      +D
      T     LD [AR1,P#6.0]      // DWord 6: Speicherbereich + Offset im Speicherbereich

// Eingangsdaten lesen
      CALL  "DPRD_DAT"                                                  SFC14             -- Read Consisten
                                                                        t Data of a Standard DP Slave
       LADDR  :=#wLaddr                                                 #wLaddr
       RET_VAL:=#iRetval                                                #iRetval
       RECORD :=#anyDst                                                 #anyDst

// RetVal auswerten
      L     #iRetval                                                    #iRetval
      L     0
      ==I
      SPB   S001
      L     #iRetval                                                    #iRetval
      T     #STATUS             // schreibe STATUS = iRetVal DPRD_DAT   #STATUS           -- Spezifikation
                                                                        des Fehlers bei ERROR=1
      SPA   E100
S001: NOP   0


//*********************************************************************************************
// CONTROL aktualisieren
//*********************************************************************************************

      L     #CONTROL                                                    #CONTROL          -- geteilter Spei
                                                                        cher: Sendebaustein Bit 0..3, Empfan
                                                                        gsbaustein Bit 4-7
      T     #arOutputData[0]                                            #arOutputData[0]


//*********************************************************************************************
// Auftrag/Reset mit Fehler oder abgeschlossener Reset -> Handling Ausgänge
//*********************************************************************************************

      O     #REQ                                                        #REQ              -- Auftragsanstoß
                                                                         bei positiver Flanke
      O     #R                                                          #R                -- Synchron Reset
                                                                         auslösen
      U     #ERROR                                                      #ERROR            -- Auftrag fertig
                                                                         mit Fehler, Parameter STATUS enthäl
                                                                        t die Fehlerinformation
```

```
      S     #xWait                                                    #xWait             -- warte nach fer
                                                                      tigem Reset oder Auftrag mit Fehler
                                                                      auf FALS an EN_R und R

      U     #xFpReset                                                 #xFpReset          -- Hilfsbit zur E
                                                                      rkennung einer steigenden Flanke an
                                                                      R
      U     #DONE                                                     #DONE              -- Auftrag fertig
                                                                       ohne Fehler, Daten gesendet, Parame
                                                                      ter STATUS=00h
      S     #xWait                                                    #xWait             -- warte nach fer
                                                                      tigem Reset oder Auftrag mit Fehler
                                                                      auf FALS an EN_R und R

      CLR
      =     #DONE                                                     #DONE              -- Auftrag fertig
                                                                       ohne Fehler, Daten gesendet, Parame
                                                                      ter STATUS=00h
      =     #ERROR                                                    #ERROR             -- Auftrag fertig
                                                                       mit Fehler, Parameter STATUS enthäl
                                                                      t die Fehlerinformation
      L     0
      T     #STATUS                                                   #STATUS            -- Spezifikation
                                                                      des Fehlers bei ERROR=1

      UN    #REQ                                                      #REQ               -- Auftragsanstoß
                                                                       bei positiver Flanke
      UN    #R                                                        #R                 -- Synchron Reset
                                                                       auslösen
      R     #xWait                                                    #xWait             -- warte nach fer
                                                                      tigem Reset oder Auftrag mit Fehler
                                                                      auf FALS an EN_R und R

      U     #xWait                                                    #xWait             -- warte nach fer
                                                                      tigem Reset oder Auftrag mit Fehler
                                                                      auf FALS an EN_R und R
      SPB   E100


//*********************************************************************************************
// EN_R=0 während Datenempfang -> Synchron Reset anstoßen
//*********************************************************************************************

      U     #REQ                                                      #REQ               -- Auftragsanstoß
                                                                       bei positiver Flanke
      SPB   I000
      L     #bIntStatus                                               #bIntStatus        -- interner Baust
                                                                      einstatus
      L     B#16#0              // Datensendung aktiv?
      ==I
      SPB   I000                // nein
      SET
      =     #xResetBecauseAbort // ja -> REQ während Übertragung zurü  #xResetBecauseAbort -- Reset durch f
                                kgesetzt -> Synchron Reset anstoßen     allende Flanke an REQ ausgelöst
I000: NOP   0


//*********************************************************************************************
// Aufruf mit REQ=R=0 und keine fallende Flanke an REQ -> Baustein verlassen
//*********************************************************************************************

      O     #REQ                                                      #REQ               -- Auftragsanstoß
                                                                       bei positiver Flanke
      O     #R                                                        #R                 -- Synchron Reset
                                                                       auslösen
      O     #xResetBecauseAbort                                       #xResetBecauseAbort -- Reset durch f
                                                                      allende Flanke an REQ ausgelöst
      SPB   I010
// Ausgangsparameter (Ausgangsdaten bleiben unverändert) und statische Variablen zurücksetzen
      CLR
      =     #DONE               // DONE                               #DONE              -- Auftrag fertig
                                                                       ohne Fehler, Daten gesendet, Parame
                                                                      ter STATUS=00h
      =     #ERROR              // ERROR                              #ERROR             -- Auftrag fertig
                                                                       mit Fehler, Parameter STATUS enthäl
                                                                      t die Fehlerinformation
      =     #xFpRequest                                               #xFpRequest        -- Hilfsbit zur E
                                                                      rkennung einer steigenden Flanke an
                                                                      REQ
      =     #xFpReset                                                 #xFpReset          -- Hilfsbit zur E
                                                                      rkennung einer steigenden Flanke an
                                                                      R
      =     #xResetDone                                               #xResetDone        -- Fertigmeldung
                                                                      Synchron Reset (Statemachine Bit)
      =     #xResetBecauseAbort                                       #xResetBecauseAbort -- Reset durch f
                                                                      allende Flanke an REQ ausgelöst
      L     0
```

```
      T     #STATUS             // STATUS                          #STATUS           -- Spezifikation
                                                                   des Fehlers bei ERROR=1
      T     #bIntStatus                                           #bIntStatus       -- interner Baust
                                                                   einstatus
      T     #bAcknowledge                                         #bAcknowledge     -- Bit 4..7 des E
                                                                   ingangsbyte 0 als Quittung
      T     #bFrag                                                #bFrag            -- Fragmentanzahl
      T     #bFragCounter                                         #bFragCounter     -- Fragmentzähler
      T     #iLenTrailer                                          #iLenTrailer      -- Trailer Länge
                                                                   = Anzahl Nutzdaten im Trailer
      SPA   E100                // Baustein verlassen
I010: NOP   0



//*********************************************************************************************
// Eingangsparameter prüfen
//*********************************************************************************************

// Reset auch bei ungültigen Eingangsparametern möglich
      U     #R                                                    #R                -- Synchron Reset
                                                                    auslösen
      ON    #REQ                                                  #REQ              -- Auftragsanstoß
                                                                    bei positiver Flanke
      SPB   I024

// DB_NO prüfen
      L     #DB_NO                                                #DB_NO            -- Datenbausteinn
                                                                   ummer - Nummer des Sende-DB (>0)
      L     0
      <>I                       // Fehler bei DB_NO=0
      SPB   I020
      L     W#16#301
      T     #STATUS             // schreibe STATUS 0x0301         #STATUS           -- Spezifikation
                                                                   des Fehlers bei ERROR=1
      SPA   E100                // Baustein verlassen
I020: NOP   0
// LEN prüfen
      L     #LEN                                                  #LEN              -- Länge des zu s
                                                                   endenen Telegramms in Byte (>0 und <
                                                                   1025)
      L     0
      ==I                       // Fehler bei LEN=0
      SPB   I021
      L     #LEN                                                  #LEN              -- Länge des zu s
                                                                   endenen Telegramms in Byte (>0 und <
                                                                   1025)
      L     1024
      >I                        // Fehler bei LEN>1024
      SPBN  I022
I021: NOP   0
      L     W#16#517
      T     #STATUS             // schreibe STATUS 0x0517         #STATUS           -- Spezifikation
                                                                   des Fehlers bei ERROR=1
      SPA   E100                // Baustein verlassen
I022: NOP   0
// IO_SIZE prüfen
      L     #IO_SIZE                                              #IO_SIZE          -- parametrierte
                                                                   IO Größe des Moduls
      L     0
      ==I                       // Fehler bei IO_SIZE=0
      SPB   I023
      L     #IO_SIZE                                              #IO_SIZE          -- parametrierte
                                                                   IO Größe des Moduls
      L     60
      >I                        // Fehler bei IO_SIZE>60
      SPBN  I024
I023: NOP   0
      L     W#16#202
      T     #STATUS             // schreibe STATUS 0x0202         #STATUS           -- Spezifikation
                                                                   des Fehlers bei ERROR=1
      SPA   E100                // Baustein verlassen
I024: NOP   0
// keine Fehler an den Eingangsparametern -> STATUS zurücksetzen
      L     W#16#0
      T     #STATUS             // schreibe STATUS 0x0000         #STATUS           -- Spezifikation
                                                                   des Fehlers bei ERROR=1



//*********************************************************************************************
// Aufruf mit R=1 (Reset immer möglich, unabhängig von REQ)
//*********************************************************************************************

      O     #R                  // R gesetzt?                     #R                -- Synchron Reset
                                                                    auslösen
      O     #xResetBecauseAbort // oder Reset durch fallende Flanke an  #xResetBecauseAbort -- Reset durch f
                                 REQ?                              allende Flanke an REQ ausgelöst
      SPBN  I030                // nein -> Flankenmerker R zurücksetzen
```

```
      U     #xFpReset           // Synchon Reset bereits gesartet?      #xFpReset          -- Hilfsbit zur E
                                                                       rkennung einer steigenden Flanke an
                                                                       R
      SPB   I031                // ja -> Reset bearbeiten
// neuen Reset anstoßen
      L     B#16#10             // Int_Status = SCHREIBE RESET
      T     #bIntStatus                                              #bIntStatus        -- interner Baust
                                                                       einstatus
      SET
      =     #xFpReset                                                #xFpReset          -- Hilfsbit zur E
                                                                       rkennung einer steigenden Flanke an
                                                                       R
      SPA   I031
I030: NOP   0
      CLR
      =     #xFpReset           // Flankenmerker R zurücksetzen      #xFpReset          -- Hilfsbit zur E
                                                                       rkennung einer steigenden Flanke an
                                                                       R
I031: NOP   0



//*********************************************************************************************
// Aufruf mit REQ=1 (Senden nur mit R=0 möglich)
//*********************************************************************************************

      U     #xFpReset           // Reset angestoßen?                 #xFpReset          -- Hilfsbit zur E
                                                                       rkennung einer steigenden Flanke an
                                                                       R
      SPB   I040                // ja -> Synchron Reset bearbeiten
      UN    #REQ                // REQ gesetzt?                      #REQ               -- Auftragsanstoß
                                                                        bei positiver Flanke
      SPB   E100                // nein -> Baustein verlassen
      U     #DONE               // Auftrag bereits bearbeitet        #DONE              -- Auftrag fertig
                                                                        ohne Fehler, Daten gesendet, Parame
                                                                       ter STATUS=00h
      SPB   E100                // Baustein verlassen
// neuen Auftrag anstoßen oder aktuellen Auftrag bearbeiten
      U     #xFpRequest                                              #xFpRequest        -- Hilfsbit zur E
                                                                       rkennung einer steigenden Flanke an
                                                                       REQ
      SPB   I040
      L     B#16#20             // Int_Status = START
      T     #bIntStatus                                              #bIntStatus        -- interner Baust
                                                                       einstatus
      SET
      =     #xFpRequest                                              #xFpRequest        -- Hilfsbit zur E
                                                                       rkennung einer steigenden Flanke an
                                                                       REQ
I040: NOP   0



//*********************************************************************************************
// Quittierungsbyte generieren
//*********************************************************************************************

      L     #arInputData[0]                                          #arInputData[0]
      SRW   4
      T     #bAcknowledge       // Quittierungsbyte = Bit 4..7 von Ein #bAcknowledge      -- Bit 4..7 des E
                                gangsbyte 0                          ingangsbyte 0 als Quittung



//*********************************************************************************************
// Bausteinstatus auswerten
//*********************************************************************************************

// Int_Status = SCHREIBE RESET
      L     B#16#10
      L     #bIntStatus                                              #bIntStatus        -- interner Baust
                                                                       einstatus
      ==I
      SPB   R000
// Int_Status = START
      L     B#16#20
      L     #bIntStatus                                              #bIntStatus        -- interner Baust
                                                                       einstatus
      ==I
      SPB   S000
// Int_Status = SENDEN
      L     B#16#30
      L     #bIntStatus                                              #bIntStatus        -- interner Baust
                                                                       einstatus
      ==I
      SPB   T000
// Int_Status = TLQ QUITTUNG
      L     B#16#40
      L     #bIntStatus                                              #bIntStatus        -- interner Baust
                                                                       einstatus
```

```
      ==I
      SPB   Q000
// Int_Status = ABSCHLIEßEN
      L     B#16#50
      L     #bIntStatus                                             #bIntStatus        -- interner Baust
                                                                    einstatus
      ==I
      SPB   A000
// Int_Status = ERROR
      L     B#16#60
      L     #bIntStatus                                             #bIntStatus        -- interner Baust
                                                                    einstatus
      ==I
      SPB   ER00
// Int_Status nicht definiert = ENDE
      SPA   E100
```

Segmento: 2 SCHREIBE RESET

```
R000: NOP   0


//*********************************************************************************************
// "Reset abschließen" prüfen
//*********************************************************************************************

      U     #xResetDone                                             #xResetDone        -- Fertigmeldung
                                                                    Synchron Reset (Statemachine Bit)
      SPB   R010                // Quittung 0x08 prüfen
      SPA   R020                // Quittung 0x0C prüfen


R010: NOP   0


//*********************************************************************************************
// Quittung 0x08 prüfen
//*********************************************************************************************

      L     #bAcknowledge                                           #bAcknowledge      -- Bit 4..7 des E
                                                                    ingangsbyte 0 als Quittung
      L     B#16#8
      <>I
      SPB   R011
      L     B#16#0              // Int_Status = zurücksetzen
      T     #bIntStatus                                             #bIntStatus        -- interner Baust
                                                                    einstatus
      CLR
      =     #xResetDone                                             #xResetDone        -- Fertigmeldung
                                                                    Synchron Reset (Statemachine Bit)
      =     #xResetBecauseAbort                                     #xResetBecauseAbort -- Reset durch f
                                                                    allende Flanke an REQ ausgelöst
R011: NOP   0

      SPA   E100                // Baustein verlassen


R020: NOP   0


//*********************************************************************************************
// Quittung 0x0C prüfen
//*********************************************************************************************

      L     #bAcknowledge                                           #bAcknowledge      -- Bit 4..7 des E
                                                                    ingangsbyte 0 als Quittung
      L     B#16#C
      <>I
      SPB   R021
// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                #CONTROL           -- geteilter Spei
                                                                    cher: Sendebaustein Bit 0..3, Empfan
                                                                    gsbaustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#8              // Synchron Reset quittieren
      OW
      T     #arOutputData[0]                                        #arOutputData[0]
      SET
```

```
      =     #xResetDone          // "Reset abschließen" setzen           #xResetDone        -- Fertigmeldung
                                                                          Synchron Reset (Statemachine Bit)
      =     #DONE                // DONE setzen                          #DONE              -- Auftrag fertig
                                                                           ohne Fehler, Daten gesendet, Parame
                                                                          ter STATUS=00h
      SPA   R022
R021: NOP   0
// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                     #CONTROL           -- geteilter Spei
                                                                          cher: Sendebaustein Bit 0..3, Empfan
                                                                          gsbaustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#B               // Synchron Reset Quittierung anfordern
      OW
      T     #arOutputData[0]                                             #arOutputData[0]
R022: NOP   0
      SPA   E100                 // Baustein verlassen
```

Segmento: 3 START

```
S000: NOP   0



//*********************************************************************************************
// Quittung 0x08 prüfen
//*********************************************************************************************

      L     #bAcknowledge                                                #bAcknowledge      -- Bit 4..7 des Ein
                                                                          gangsbyte 0 als Quittung
      L     B#16#8
      ==I
      SPB   S010
      L     W#16#202
      T     #STATUS              // schreibe STATUS 0x0202               #STATUS            -- Spezifikation de
                                                                          s Fehlers bei ERROR=1
      SPA   E100                 // Baustein verlassen
S010: NOP   0



//*********************************************************************************************
// Fragmentanzahl berechnen und prüfen
//*********************************************************************************************

// Zwischenwert 1 = Dividend
      L     #LEN                                                         #LEN               -- Länge des zu sen
                                                                          denen Telegramms in Byte (>0 und <1025
                                                                          )
      L     3
      +I
      T     #wTemp1                                                      #wTemp1
// Zwischenwert 2 = Divisor
      L     #IO_SIZE                                                     #IO_SIZE           -- parametrierte IO
                                                                           Größe des Moduls
      L     1
      -I
      T     #wTemp2                                                      #wTemp2
// Dividend / Divisor -> Quotient ohne Divisionsrest
      L     #wTemp1                                                      #wTemp1
      L     #wTemp2                                                      #wTemp2
      /D
      T     #bFrag               // Fragmentanzahl                       #bFrag             -- Fragmentanzahl
// Dividend / Divisor -> Divisionsrest = Länge des Trailer
      L     #wTemp1                                                      #wTemp1
      L     #wTemp2                                                      #wTemp2
      MOD
      T     #iLenTrailer         // setze Trailerlänge = Divisionsrest   #iLenTrailer       -- Trailer Länge =
                                                                          Anzahl Nutzdaten im Trailer
      L     0
      ==I
      SPB   S020
      L     #bFrag                                                       #bFrag             -- Fragmentanzahl
      L     1
      +I
      T     #bFrag               // inkrementiere Fragmentanzahl bei vorh #bFrag             -- Fragmentanzahl
                                 andenem Divisionsrest um 1
      SPA   S021
S020: NOP   0
      L     #IO_SIZE                                                     #IO_SIZE           -- parametrierte IO
                                                                           Größe des Moduls
      DEC   1
      T     #iLenTrailer         // setze Trailerlänge = IoSize-1 Byte be #iLenTrailer       -- Trailer Länge =
                                 i bei Divisionsrest = 0                  Anzahl Nutzdaten im Trailer
```

```
S021: NOP   0
// Fragmentanzahl=1 ?
      L     #bFrag                                                  #bFrag            -- Fragmentanzahl
      L     1
      ==I
      SPB   S030              // Fragmentanzahl=1 -> schreibe Trailer/Header kombiniert
      SPA   S040              // Fragmentanzahl>1 -> schreibe Header



S030: NOP   0



//*********************************************************************************************
// Schreibe Header/Trailer kombiniert
//*********************************************************************************************

// DONE zurücksetzen
      CLR
      =     #DONE                                                   #DONE             -- Auftrag fertig o
                                                                                      hne Fehler, Daten gesendet, Parameter
                                                                                      STATUS=00h
// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                #CONTROL          -- geteilter Speich
                                                                                      er: Sendebaustein Bit 0..3, Empfangsba
                                                                                      ustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#A            // Datensendung über RS232 vorbereiten -> Header/Trailer kombiniert
      OW
      T     #arOutputData[0]                                        #arOutputData[0]
// Int_Status = TLQ QUITTUNG setzen
      L     B#16#40
      T     #bIntStatus       // Int_Status = TLQ QUITTUNG          #bIntStatus       -- interner Baustei
                                                                                      nstatus
      SPA   S050              // gemeinsame Sendedaten Header/Trailer kombiniert / Header generieren



S040: NOP   0



//*********************************************************************************************
// Schreibe Header
//*********************************************************************************************

// DONE zurücksetzen
      CLR
      =     #DONE                                                   #DONE             -- Auftrag fertig o
                                                                                      hne Fehler, Daten gesendet, Parameter
                                                                                      STATUS=00h
// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                #CONTROL          -- geteilter Speich
                                                                                      er: Sendebaustein Bit 0..3, Empfangsba
                                                                                      ustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#9            // Datensendung über RS232 vorbereiten -> Header
      OW
      T     #arOutputData[0]                                        #arOutputData[0]
// Fragmentzähler auf 1 setzen
      L     1
      T     #bFragCounter                                           #bFragCounter     -- Fragmentzähler
// Int_Status = SENDEN setzen
      L     B#16#30
      T     #bIntStatus       // Int_Status = SENDEN                #bIntStatus       -- interner Baustei
                                                                                      nstatus
      SPA   S050              // gemeinsame Sendedaten Header/Trailer kombiniert / Header generieren



S050: NOP   0



//*********************************************************************************************
// gemeinsame Sendedaten Header/Trailer kombiniert / Header generieren
//*********************************************************************************************

// Ausgangsbyte 1 = 0x00
      L     0
      T     #arOutputData[1]                                        #arOutputData[1]
// Ausgangsbyte 2+3 = LEN
      L     #LEN                                                    #LEN              -- Länge des zu sen
                                                                                      denen Telegramms in Byte (>0 und <1025
                                                                                      )
      SRW   8
      T     #arOutputData[2]                                        #arOutputData[2]
```

```
      L     #LEN                                                    #LEN              -- Länge des zu sen
                                                                    denen Telegramms in Byte (>0 und <1025
                                                                    )
      UW    W#16#FF
      T     #arOutputData[3]                                        #arOutputData[3]

// Ausgangsbyte 4-16 = SRCBLK zusammenbauen
      L     #IO_SIZE                                                #IO_SIZE          -- parametrierte IO
                                                                     Größe des Moduls
      L     4                 // Header Länge
      -I
      T     #wTemp1           // Byte 2+3: Länge = LEN              #wTemp1
      L     #DBB_NO                                                 #DBB_NO           -- Datenbytenummer
                                                                    - Sendedaten ab Datenbyte
      SLD   3
      T     #dwTemp1          // Byte 8+9: Offset im Sende-DB       #dwTemp1
// Ausgangsbyte 4-16 = DSTBLK zusammenbauen
      L     #IO_SIZE                                                #IO_SIZE          -- parametrierte IO
                                                                     Größe des Moduls
      L     4                 // 4 Byte Header nicht mitkopieren
      -I
      T     #wTemp2                                                 #wTemp2
      L     32
      T     #dwTemp2          // Byte 8+9: Offset im Ausgangsdatenbere  #dwTemp2
                              ich ist 4 Byte = 32 Bit
// STATUS schreiben
      L     W#16#8181
      T     #STATUS           // schreibe STATUS 0x8181             #STATUS           -- Spezifikation de
                                                                    s Fehlers bei ERROR=1
      SPA   E000              // Daten kopieren
```

Segmento: 4 SENDEN

```
T000: NOP   0


//*********************************************************************************************
// Quittung = letzte Fragmentnummer prüfen
//*********************************************************************************************

// Fragmentnummer steht in Bit0..2 der Quittierung
      L     #bAcknowledge                                           #bAcknowledge     -- Bit 4..7 des Ein
                                                                    gangsbyte 0 als Quittung
      L     B#16#7
      UW
      T     #wTemp1                                                 #wTemp1
// Fragmentnummer = Fragmentzähler & 0x07
      L     #bFragCounter                                           #bFragCounter     -- Fragmentzähler
      L     B#16#7
      UW
      T     #wTemp2                                                 #wTemp2
// vergleiche Quittierung mit Fragmentnummer
      L     #wTemp1                                                 #wTemp1
      L     #wTemp2                                                 #wTemp2
      ==I
      SPB   T010
      SPA   E100              // Baustein verlassen
T010: NOP   0
// Fragmentzähler um 1 inkrementieren
      L     #bFragCounter                                           #bFragCounter     -- Fragmentzähler
      L     1
      +I
      T     #bFragCounter                                           #bFragCounter     -- Fragmentzähler



//*********************************************************************************************
// Fragmentzähler = Fragmentanzahl ?
//*********************************************************************************************

      L     #bFrag                                                  #bFrag            -- Fragmentanzahl
      L     #bFragCounter                                           #bFragCounter     -- Fragmentzähler
      ==I
      SPB   T020              // schreibe Trailer
      SPA   T030              // schreibe Fragment



T020: NOP   0


//*********************************************************************************************
// Schreibe Trailer
```

```
//*********************************************************************************************

// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                    #CONTROL           -- geteilter Speich
                                                                       er: Sendebaustein Bit 0..3, Empfangsba
                                                                       ustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#A            // Datensendung über RS232 vorbereiten -> Trailer
      OW
      T     #arOutputData[0]                                          #arOutputData[0]
//  Sendedaten Trailer generieren
      L     #iLenTrailer                                              #iLenTrailer       -- Trailer Länge =
                                                                       Anzahl Nutzdaten im Trailer
      T     #wTemp1           // Byte 2+3: Trailer Länge              #wTemp1
      T     #wTemp2           // Byte 2+3: Trailer Länge              #wTemp2
// Int_Status = TLQ QUITTUNG setzen
      L     B#16#40
      T     #bIntStatus       // Int_Status = TLQ QUITTUNG            #bIntStatus        -- interner Baustei
                                                                       nstatus
      SPA   T050              // gemeinsame Sendedaten Trailer/Fragment generieren



T030: NOP   0



//*********************************************************************************************
// Schreibe Fragment
//*********************************************************************************************

// Ausgangsbyte 0 = Fragmentnummer = Fragmentzähler & 0x07

      L     #CONTROL          // Datensendung über RS232 vorbereiten -  #CONTROL           -- geteilter Speich
                              > Fragment                               er: Sendebaustein Bit 0..3, Empfangsba
                                                                       ustein Bit 4-7
      L     B#16#F0
      UW
      L     #bFragCounter                                             #bFragCounter      -- Fragmentzähler
      SLW   13
      SRW   13
      OW
      L     B#16#F7           // Ausmaskieren des Steuerbits
      UW
      T     #arOutputData[0]                                          #arOutputData[0]

//  Sendedaten Fragment generieren
      L     #IO_SIZE                                                  #IO_SIZE           -- parametrierte IO
                                                                       Größe des Moduls
      L     1
      -I
      T     #wTemp1           // Byte 2+3: Fragmentlänge = IO_SIZE - 1  #wTemp1
                               Byte
      T     #wTemp2           // Byte 2+3: Fragmentlänge = IO_SIZE - 1  #wTemp2
                               Byte
      SPA   T050              // gemeinsame Sendedaten Trailer/Fragment generieren



T050: NOP   0



//*********************************************************************************************
// gemeinsame Sendedaten Trailer/Fragment generieren
//*********************************************************************************************

      L     #bFragCounter                                             #bFragCounter      -- Fragmentzähler
      L     2
      -D
      L     #IO_SIZE                                                  #IO_SIZE           -- parametrierte IO
                                                                       Größe des Moduls
      DEC   1
      *D
      L     #IO_SIZE                                                  #IO_SIZE           -- parametrierte IO
                                                                       Größe des Moduls
      +D
      L     #DBB_NO                                                   #DBB_NO            -- Datenbytenummer
                                                                       - Sendedaten ab Datenbyte
      +D
      L     4
      -D
      SLD   3
      T     #dwTemp1          // Byte 8+9: Offset im Sende-DB         #dwTemp1
      L     8
      T     #dwTemp2          // Byte 8+9: Offset im Ausgangsdatenbere  #dwTemp2
                              ich ist 1 Byte = 8 Bit
      SPA   E000
```

Segmento: 5 TLQ QUITTUNG

```
Q000: NOP   0


//*****************************************************************************************
// Quittung 0x0A prüfen
//*****************************************************************************************

      L     #bAcknowledge                                                 #bAcknowledge      -- Bit 4..7 des Ein
                                                                          gangsbyte 0 als Quittung
      L     B#16#A
      ==I
      SPB   Q010
      SPA   E100              // Baustein verlassen
Q010: NOP   0
// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                    #CONTROL           -- geteilter Speich
                                                                          er: Sendebaustein Bit 0..3, Empfangsba
                                                                          ustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#8            // Daten über RS232 senden
      OW
      T     #arOutputData[0]                                            #arOutputData[0]
// Int_Status = ABSCHLIEßEN setzen und Baustein verlassen
      L     B#16#50
      T     #bIntStatus       // Int_Status = ABSCHLIEßEN               #bIntStatus        -- interner Baustei
                                                                          nstatus
      SPA   E100              // Baustein verlassen
```

Segmento: 6 ERROR

```
ER00: NOP   0

      L     #bAcknowledge                                               #bAcknowledge      -- Bit 4..7 des Ein
                                                                          gangsbyte 0 als Quittung
      L     B#16#8
      ==I
      SPB   ER10
      SPA   E100

//*****************************************************************************************
// Quittung 0x08 prüfen
//*****************************************************************************************

ER10: NOP   0
// STATUS schreiben
      L     #wStatusMem                                                 #wStatusMem        -- temporärer Zwisc
                                                                          henspeicher für STATUS Ausgang
      T     #STATUS           // schreibe zwischengespeicherten STATUS  #STATUS            -- Spezifikation de
                                                                          s Fehlers bei ERROR=1
// Int_Status zurücksetzen und Baustein verlassen
      L     B#16#0
      T     #bIntStatus       // Int_Status zurücksetzen                #bIntStatus        -- interner Baustei
                                                                          nstatus
      L     #CONTROL                                                    #CONTROL           -- geteilter Speich
                                                                          er: Sendebaustein Bit 0..3, Empfangsba
                                                                          ustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#8            // CP Ruhezustand quittieren
      OW
      T     #arOutputData[0]                                            #arOutputData[0]
      SPA   E100              // Baustein verlassen
```

Segmento: 7 ABSCHLIEßEN

```
A000: NOP   0

      L     #bAcknowledge                                           #bAcknowledge      -- Bit 4..7 des Ein
                                                                    gangsbyte 0 als Quittung
      L     B#16#8
      ==I
      SPB   A010
      L     #bAcknowledge                                           #bAcknowledge      -- Bit 4..7 des Ein
                                                                    gangsbyte 0 als Quittung
      L     B#16#D
      ==I
      SPB   A020
      L     #bAcknowledge                                           #bAcknowledge      -- Bit 4..7 des Ein
                                                                    gangsbyte 0 als Quittung
      L     B#16#E
      ==I
      SPB   A030
      SPA   E100              // Baustein verlassen


//*****************************************************************************************
// Quittung 0x08 prüfen
//*****************************************************************************************

A010: NOP   0
// STATUS schreiben
      L     W#16#0
      T     #STATUS           // schreibe STATUS 0x0000                #STATUS            -- Spezifikation de
                                                                    s Fehlers bei ERROR=1
// DONE setzen
      SET
      =     #DONE                                                   #DONE              -- Auftrag fertig o
                                                                    hne Fehler, Daten gesendet, Parameter
                                                                    STATUS=00h
// Int_Status zurücksetzen und Baustein verlassen
      L     B#16#0
      T     #bIntStatus       // Int_Status zurücksetzen             #bIntStatus        -- interner Baustei
                                                                    nstatus
      SPA   E100              // Baustein verlassen


//*****************************************************************************************
// Quittung 0x0D prüfen
//*****************************************************************************************

A020: NOP   0
// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                #CONTROL           -- geteilter Speich
                                                                    er: Sendebaustein Bit 0..3, Empfangsba
                                                                    ustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#D            // signalisiere ungültige Länge
      OW
      T     #arOutputData[0]                                        #arOutputData[0]
// Fehlerstatus zwischenspeichern
      L     W#16#517
      T     #wStatusMem                                             #wStatusMem        -- temporärer Zwisc
                                                                    henspeicher für STATUS Ausgang
// Int_Status = ERROR setzen und Baustein verlassen
      L     B#16#60
      T     #bIntStatus       // Int_Status = ERROR                  #bIntStatus        -- interner Baustei
                                                                    nstatus
      SPA   E100              // Baustein verlassen


//*****************************************************************************************
// Quittung 0x0E prüfen
//*****************************************************************************************

A030: NOP   0
// Ausgangsbyte 0 = CONTROL
      L     #CONTROL                                                #CONTROL           -- geteilter Speich
                                                                    er: Sendebaustein Bit 0..3, Empfangsba
                                                                    ustein Bit 4-7
      L     B#16#F0
      UW
      L     B#16#E            // signalisiere Übertragung fehlgeschlagen
      OW
      T     #arOutputData[0]                                        #arOutputData[0]
// Fehlerstatus zwischenspeichern
      L     W#16#70A
      T     #wStatusMem                                             #wStatusMem        -- temporärer Zwisc
                                                                    henspeicher für STATUS Ausgang
// Int_Status = ERROR setzen und Baustein verlassen
```

```
      L     B#16#60
      T     #bIntStatus       // Int_Status = ERROR                  #bIntStatus        -- interner Baustei
                                                                     nstatus
      SPA   E100              // Baustein verlassen
```

Segmento: 8 ENDE

```
E000: NOP   0


//*********************************************************************************************
// Daten kopieren
//*********************************************************************************************

//  SRCBLK zusammenbauen
      LAR1  P##anyScr         // Any Pointer Anfangsadresse
      L     W#16#1002
      T     LW [AR1,P#0.0]    // Byte 0+1: Datentyp Byte
      L     #wTemp1                                               #wTemp1
      T     LW [AR1,P#2.0]    // Byte 2+3: Länge
      L     #DB_NO                                                #DB_NO             -- Datenbausteinnum
                                                                  mer - Nummer des Sende-DB (>0)
      T     LW [AR1,P#4.0]    // Byte 4+5: Sende-DB Nummer
      L     #dwTemp1                                              #dwTemp1
      T     LD [AR1,P#6.0]    // Byte 7+8+9: Offset im Sende-DB
      L     B#16#84
      T     LB [AR1,P#6.0]    // Byte 6: Speicherbereich DB
//  DSTBLK zusammenbauen
      LAR1  P##anyDst         // Any Pointer Anfangsadresse
      L     W#16#1002
      T     LW [AR1,P#0.0]    // Byte 0+1: Datentyp Byte
      L     #wTemp2                                               #wTemp2
      T     LW [AR1,P#2.0]    // Byte 2+3: Länge
      L     DINO
      T     LW [AR1,P#4.0]    // Byte 4+5: DB-Nummer
      TAR2
      UD    DW#16#FFFFFF
      L     P##arOutputData
      +D
      L     #dwTemp2                                              #dwTemp2
      +D
      T     LD [AR1,P#6.0]    // DWord 6: Speicherbereich + Offset im Speicherbereich

// Sendedaten auf Ausgangsdaten umkopieren (nicht verwendete Ausgangsdaten behalten alten Wert)
      CALL  "BLKMOV"                                              SFC20              -- Copy Variables
       SRCBLK :=#anyScr                                           #anyScr
       RET_VAL:=#iRetval                                          #iRetval
       DSTBLK :=#anyDst                                           #anyDst

// RetVal auswerten
      L     #iRetval                                              #iRetval
      L     0
      ==I
      SPB   E001
      L     #iRetval                                              #iRetval
      T     #STATUS           // schreibe STATUS = iRetVal BLKMOV  #STATUS            -- Spezifikation de
                                                                  s Fehlers bei ERROR=1
E001: NOP   0


E100: NOP   0

//*********************************************************************************************
// Ausgangsdaten konsistent schreiben
//*********************************************************************************************
      L     #LADDR                                                #LADDR             -- logische Basisad
                                                                  resse des CP - entspricht der Adresse
                                                                  aus der HW-Konfiguration
      T     #wLaddr                                               #wLaddr

//  DSTBLK zusammenbauen
      LAR1  P##anyScr         // Any Pointer Anfangsadresse
      L     W#16#1002
      T     LW [AR1,P#0.0]    // Byte 0+1: Datentyp Byte
      L     #IO_SIZE                                              #IO_SIZE           -- parametrierte IO
                                                                   Größe des Moduls
      T     LW [AR1,P#2.0]    // Byte 2+3: Länge
      L     DINO
      T     LW [AR1,P#4.0]    // Byte 4+5: DB-Nummer
      TAR2
      UD    DW#16#FFFFFF
      L     P##arOutputData
      +D
      T     LD [AR1,P#6.0]    // DWord 6: Speicherbereich + Offset im Speicherbereich
```

```
// Ausgangsdaten schreiben
      CALL  "DPWR_DAT"                                                  SFC15             -- Write Consistent
                                                                         Data to a Standard DP Slave
       LADDR  :=#wLaddr                                                 #wLaddr
       RECORD :=#anyScr                                                 #anyScr
       RET_VAL:=#iRetval                                                #iRetval

// RetVal auswerten
      L     #iRetval                                                    #iRetval
      L     0
      ==I
      SPB   S002
      L     #iRetval                                                    #iRetval
      T     #STATUS           // schreibe STATUS = iRetVal DPWR_DAT     #STATUS           -- Spezifikation de
                                                                        s Fehlers bei ERROR=1
S002: NOP   0



//*********************************************************************************************
// schreibe CONTROL
//*********************************************************************************************

// schreibe CONTROL
      L     #arOutputData[0]  // CONTROL (Bit 0..3)                   #arOutputData[0]
      T     #CONTROL                                                  #CONTROL          -- geteilter Speich
                                                                        er: Sendebaustein Bit 0..3, Empfangsba
                                                                        ustein Bit 4-7



//*********************************************************************************************
// ERROR Bit steuern und auf BIE (Statuswort, Bit 8) abbilden
//*********************************************************************************************

// ERROR steuern
      L     #STATUS                                                   #STATUS           -- Spezifikation de
                                                                        s Fehlers bei ERROR=1
      L     B#16#0            // STATUS = 0x0000 -> kein Fehler
      ==I
      SPB   E110
      L     #STATUS                                                   #STATUS           -- Spezifikation de
                                                                        s Fehlers bei ERROR=1
      L     W#16#8181         // STATUS = 0x8181 -> Senden
      ==I
      SPB   E110
      SET
      =     #ERROR            // STATUS = Fehler -> ERROR setzen      #ERROR            -- Auftrag fertig m
                                                                        it Fehler, Parameter STATUS enthält di
                                                                        e Fehlerinformation
      SPA   E111
E110: NOP   0
      CLR
      =     #ERROR            // STATUS = kein Fehler -> ERROR zurücks #ERROR            -- Auftrag fertig m
                               etzen                                    it Fehler, Parameter STATUS enthält di
                                                                        e Fehlerinformation
E111: NOP   0
// BIE steuern
      UN    #ERROR                                                    #ERROR            -- Auftrag fertig m
                                                                        it Fehler, Parameter STATUS enthält di
                                                                        e Fehlerinformation
      SAVE                    // BIE=VKE
```