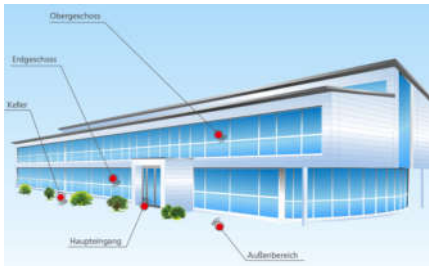


EnOcean auf dem Raspberry PI mit CODESYS



Welche Hardware wird verwendet

EnOcean Beispiel auf Raspberry PI

- ✓ Raspberry PI
- ✓ Erweiterungs-Platine EnOcean Pi Funkmodul
- ✓ PTM200 enocean Taster
- ✓ EnOcean-Modul
montieren und aktivieren (Anleitung)
- ✓ Beispiel Projekt



1. Serielle Schnittstelle aktivieren in Rasbian Stretch

- ✓ EnOcean-Modul montieren und aktivieren
- Das Modul verwendet die tty's die über GPIOs des Raspberry Pi kommen und kann über **sudo raspi-config** aktiviert werden

```
Raspberry Pi Software Configuration Tool (raspi-config)

1 Change User Password Change password for the current user
2 Network Options       Configure network settings
3 Boot Options          Configure options for start-up
4 Localisation Options  Set up language and regional settings to match your location
5 Interfacing Options   Configure connections to peripherals
6 Overclock             Configure overclocking for your Pi
7 Advanced Options      Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config    Information about this configuration tool

<Select>                <Finish>
```

```
Raspberry Pi Software Configuration Tool (raspi-config)

P1 Camera      Enable/Disable connection to the Raspberry Pi Camera
P2 SSH         Enable/Disable remote command line access to your Pi using SSH
P3 VNC         Enable/Disable graphical remote access to your Pi using RealVNC
P4 SPI         Enable/Disable automatic loading of SPI kernel module
P5 I2C         Enable/Disable automatic loading of I2C kernel module
P6 Serial      Enable/Disable shell and kernel messages on the serial connection
P7 1-Wire      Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins

<Select>                <Back>
```

```
Would you like a login shell to be accessible over
serial?

<Yes>                <No>
```

```
Would you like the serial port hardware to be enabled?

<Yes>                <No>
```

Schnittstelle aktivieren 2. Teil

2. Schnittstelle aktivieren im CODESYS Config File

- ✓ EnOcean-Modul montieren und aktivieren



- Um das Modul aus der CODESYS Applikation nutzen zu können, sind folgende Einstellungen unter Linux erforderlich:
 - COM1 in die CODESYSControl_User.cfg (diese Zeile gibt es schon in dieser Datei es muss nur „;“ am Anfang der Zeile entfernt werden)

[SysCom]

Linux.Devicefile=/dev/ttyS

<http://www.enocean.com/esp>

- 84 Seiten - uns interessiert aber nur ein ganz kleiner Teil, nämlich „Packet Type 1“.



„Packet Type 1“

1.7 Packet Type 1: RADIO_ERP1

1.7.1 Packet structure

The ERP1 radio telegram (raw data) is embedded into the ESP3 packet.
The actual user data (variable length) is a subset of the radio telegram.

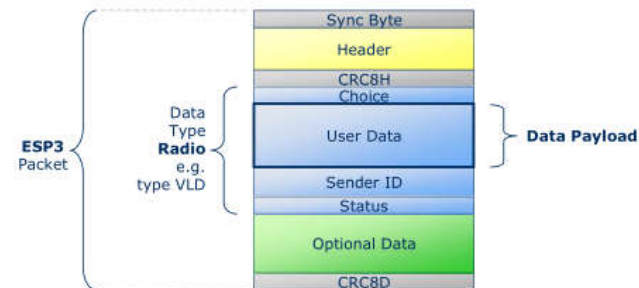


Figure 7

The following structure is applicable to all types of radio telegrams:

Group	Offset	Size	Field	Value hex	Description
-	0	1	Sync. Byte	0x55	
Header	1	2	Data Length	0xn timer	Variable length of radio telegram
	3	1	Optional Length	0x07	7 fields fixed
	4	1	Packet Type	0x01	RADIO_ERP1 = 1
-	5	1	CRC8H	0xnn	
Data	6	x	...	...	Radio telegram without checksum/CRC x = variable length / size
	6+x	1	SubTelNum	0xnn	Number of subtelegram; Send: 3 / receive: 1 ... y
Optional Data	7+x	4	Destination ID	0xxxxxxxx	Broadcast radio: FF FF FF FF ADT radio: Destination ID (= address)
	11+x	1	dBm	0xnn	Send case: FF Receive case: best RSSI value of all received subtelegrams (value decimal without minus)
	12+x	1	SecurityLevel	0x0n	0 = telegram unencrypted n = type of encryption (not supported any more)
-	13+x	1	CRC8D	0xnn	CRC8 Data byte; calculated checksum for whole byte groups: DATA and OPTIONAL_DATA

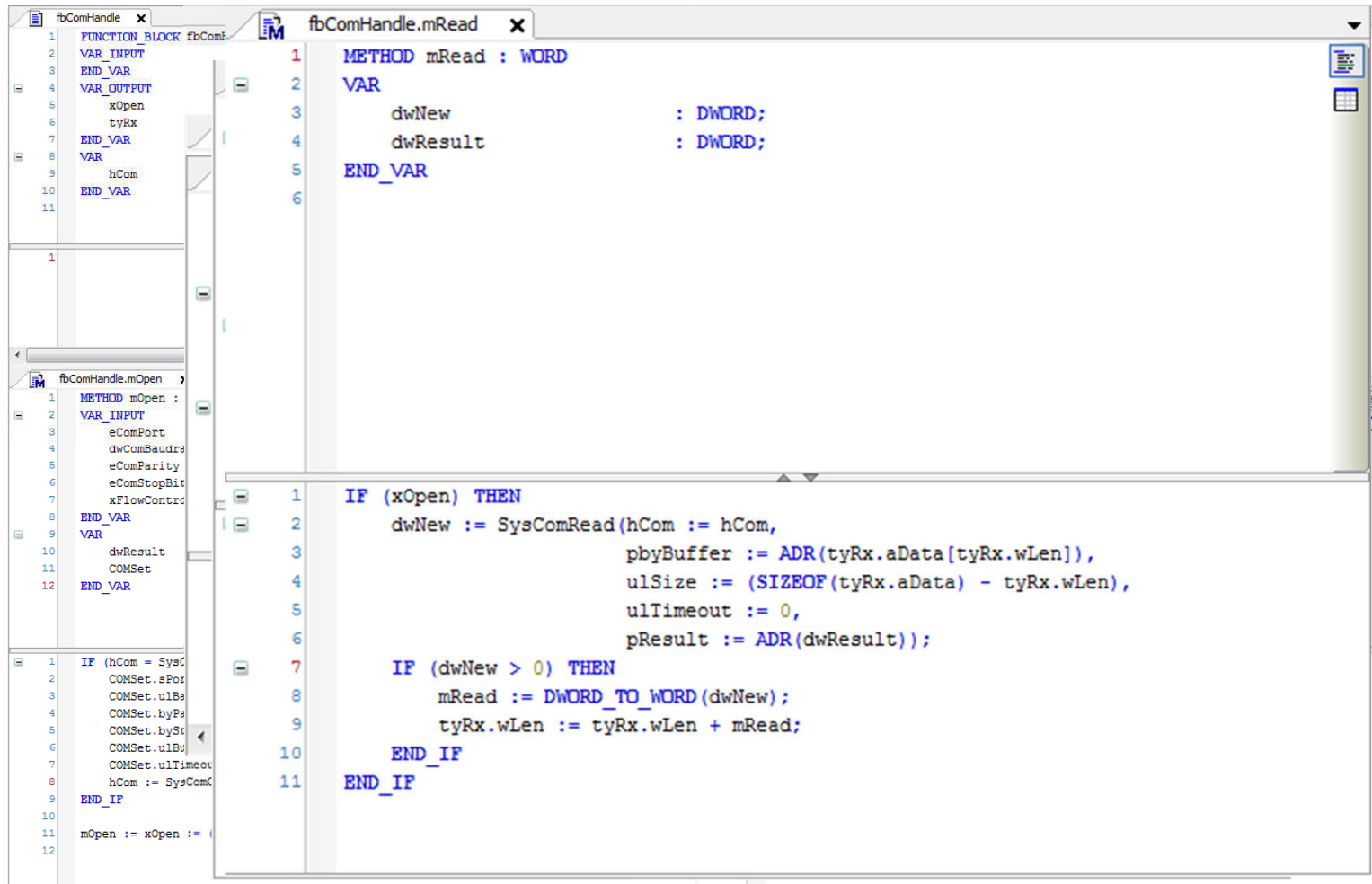
Table 4

Was ist zu tun?

Wir brauchen:

- FB, der mir die Arbeit mit der COM abnimmt
- FB, der empfangene Daten versteht und interpretieren kann
- FBs für die unterschiedliche Funktionen, die wir realisieren wollen, z. B. Steuerung von Licht oder Jalousie
- Und natürlich den gut verdienenden Manager, der den Überblick behält und Aufgaben zu delegieren weiß.

Datenstruktur „tyComBuffer“ und Funktionsbaustein „fbComHandle“



The screenshot displays the CODESYS IDE interface. On the left, the 'fbComHandle' function block is defined with the following structure:

```

1 FUNCTION_BLOCK fbComHandle
2   VAR_INPUT
3   END_VAR
4   VAR_OUTPUT
5     xOpen
6     tyRx
7   END_VAR
8   VAR
9     hCom
10  END_VAR
11
12

```

Below this, the 'fbComHandle.mOpen' method is shown:

```

1 METHOD mOpen :
2   VAR_INPUT
3     eComPort
4     dwComBaudrate
5     eComParity
6     eComStopBit
7     xFlowControl
8   END_VAR
9   VAR
10    dwResult
11    COMSet
12  END_VAR

```

The main editor shows the 'fbComHandle.mRead' method:

```

1 METHOD mRead : WORD
2   VAR
3     dwNew          : DWORD;
4     dwResult       : DWORD;
5   END_VAR
6
7   IF (xOpen) THEN
8     dwNew := SysComRead(hCom := hCom,
9                        pbyBuffer := ADR(tyRx.aData[tyRx.wLen]),
10                       ulSize := (SIZEOF(tyRx.aData) - tyRx.wLen),
11                       ulTimeout := 0,
12                       pResult := ADR(dwResult));
13
14     IF (dwNew > 0) THEN
15       mRead := DWORD_TO_WORD(dwNew);
16       tyRx.wLen := tyRx.wLen + mRead;
17     END_IF
18   END_IF
19

```

Wie geht es jetzt mit den empfangenen Daten weiter?

- Welche EnOcean-Nutzdaten benötigen wir?
- Dann legen wir uns eine passende Datenstruktur an.
- Wie viele Zeilen Code braucht man wohl, um aus den seriell empfangenden Bytes die EnOcean-Nutzdaten zu extrahieren?
- Wie synchronisiert man sich auf dem Empfangsteam auf?

BH1

Diesen Satz verstehe ich nicht.

Birgit Hess; 18.01.2016

Nutzdaten und zugehörige Datenstruktur „tyEnOceanMSG“

RADIO (VLD)

Group	Offset	Size	Field	Value hex	Description
Data	6	1	R-ORG	0xD2	Radio type VLD = D2
	7	x	User Data	0xnn...0xnn	1 ... 14 byte data payload
	7+x	4	Sender ID	0xnnnnnnnn	Unique device sender ID
	11+x	1	Status	0xnn	Telegram control bits – used in case of repeating, switch telegram encapsulation, checksum type identification

Table 5

```

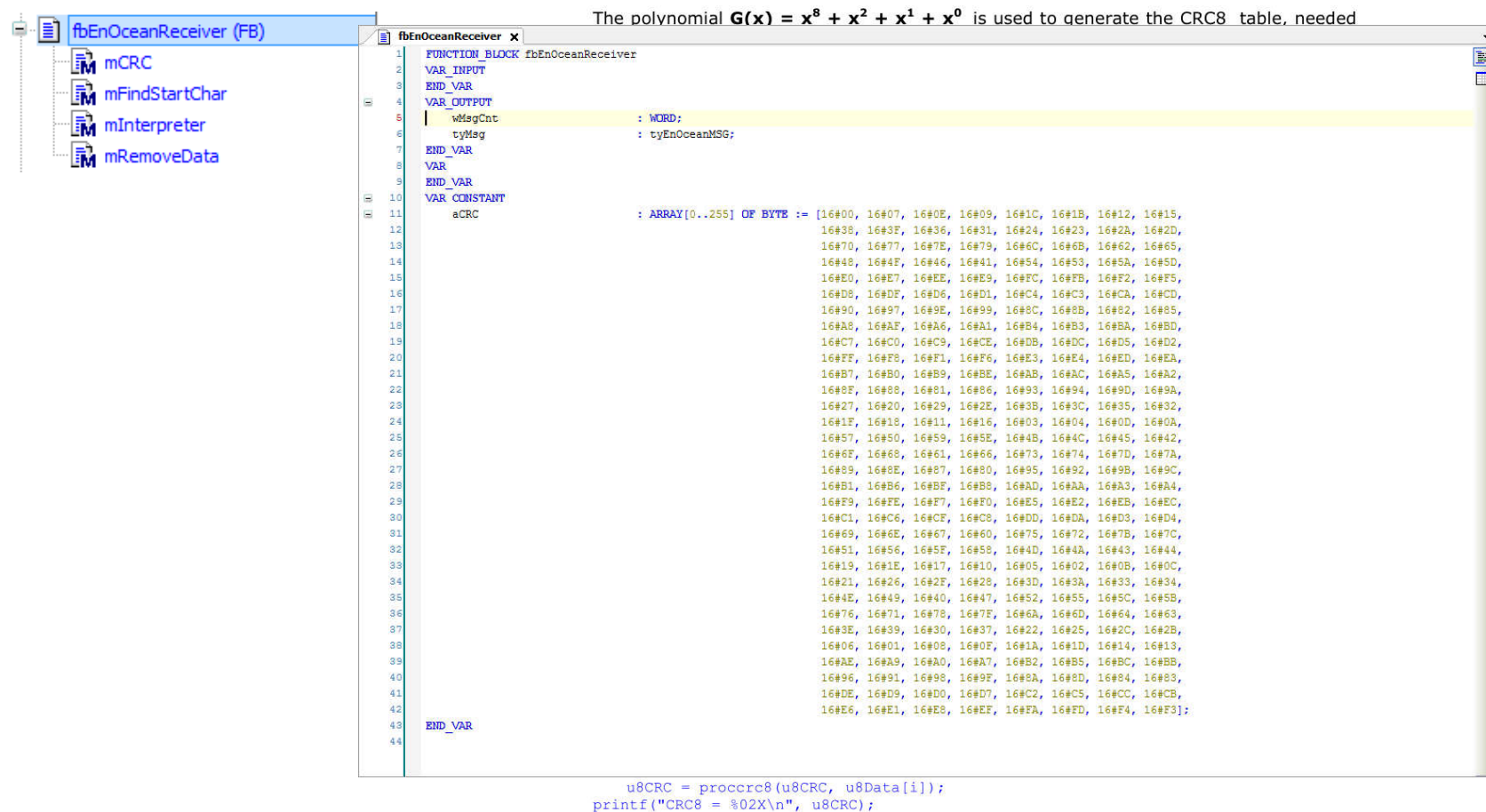
1  TYPE tyEnOceanMSG :
2  STRUCT
3      byPacketType          : BYTE; // Typ des Pakets
4      byRadioType           : BYTE; // Radio type VLD = D2
5      byStatus              : BYTE; // Telegram control bits
6      byUserDataLen         : BYTE; // Nums of UserData depending on the message-typ
7      aUserData             : ARRAY[0..31] OF BYTE; // User Date
8      dwSenderID            : DWORD; // Unique device sender ID
9  END_STRUCT
10 END_TYPE
11

```

Struktur, Variablen und Konstanten von „fbEnOceanReceiver“

2.3 CRC8 calculation

The polynomial $G(x) = x^8 + x^2 + x^1 + x^0$ is used to generate the CRC8 table, needed



```

1 FUNCTION_BLOCK fbEnOceanReceiver
2   VAR_INPUT
3   END_VAR
4   VAR_OUTPUT
5   | wMsgCnt          : WORD;
6   | tyMsg            : tyEnOceanMSG;
7   END_VAR
8   VAR
9   END_VAR
10  VAR CONSTANT
11  | aCRC              : ARRAY[0..255] OF BYTE := [16#00, 16#07, 16#0E, 16#09, 16#1C, 16#1B, 16#12, 16#15,
12  | 16#38, 16#3F, 16#36, 16#31, 16#24, 16#23, 16#2A, 16#2D,
13  | 16#70, 16#77, 16#7E, 16#79, 16#6C, 16#6B, 16#62, 16#65,
14  | 16#48, 16#4F, 16#46, 16#41, 16#54, 16#53, 16#5A, 16#5D,
15  | 16#E0, 16#E7, 16#EE, 16#E9, 16#FC, 16#FB, 16#F2, 16#F5,
16  | 16#D8, 16#DF, 16#D6, 16#D1, 16#C4, 16#C3, 16#CA, 16#CD,
17  | 16#90, 16#97, 16#9E, 16#99, 16#8C, 16#8B, 16#82, 16#85,
18  | 16#AB, 16#AE, 16#A6, 16#A1, 16#B4, 16#B3, 16#BA, 16#BD,
19  | 16#C7, 16#C0, 16#C9, 16#CE, 16#DB, 16#DC, 16#DS, 16#D2,
20  | 16#FF, 16#F8, 16#F1, 16#FE, 16#E3, 16#E4, 16#ED, 16#EA,
21  | 16#B7, 16#B0, 16#B9, 16#BE, 16#AB, 16#AC, 16#AS, 16#A2,
22  | 16#BF, 16#B8, 16#B1, 16#BE, 16#93, 16#94, 16#9D, 16#9A,
23  | 16#27, 16#20, 16#29, 16#2E, 16#3B, 16#3C, 16#35, 16#32,
24  | 16#1F, 16#18, 16#11, 16#16, 16#03, 16#04, 16#0D, 16#0A,
25  | 16#57, 16#50, 16#59, 16#5E, 16#4B, 16#4C, 16#45, 16#42,
26  | 16#6F, 16#68, 16#61, 16#66, 16#73, 16#74, 16#7D, 16#7A,
27  | 16#39, 16#3E, 16#37, 16#30, 16#95, 16#92, 16#9B, 16#9C,
28  | 16#D1, 16#D6, 16#DF, 16#DB, 16#AD, 16#AA, 16#A3, 16#A4,
29  | 16#F9, 16#FE, 16#F7, 16#F0, 16#E5, 16#E2, 16#EB, 16#EC,
30  | 16#C1, 16#C6, 16#CF, 16#CB, 16#DD, 16#DA, 16#D3, 16#D4,
31  | 16#69, 16#6E, 16#67, 16#60, 16#75, 16#72, 16#7B, 16#7C,
32  | 16#51, 16#56, 16#5F, 16#58, 16#4D, 16#4A, 16#43, 16#44,
33  | 16#19, 16#1E, 16#17, 16#10, 16#05, 16#02, 16#0B, 16#0C,
34  | 16#21, 16#26, 16#2F, 16#28, 16#3D, 16#3A, 16#33, 16#34,
35  | 16#4E, 16#49, 16#40, 16#47, 16#52, 16#55, 16#5C, 16#5B,
36  | 16#76, 16#71, 16#78, 16#7F, 16#6A, 16#6D, 16#64, 16#63,
37  | 16#3E, 16#39, 16#30, 16#37, 16#22, 16#25, 16#2C, 16#2B,
38  | 16#06, 16#01, 16#08, 16#0F, 16#1A, 16#1D, 16#14, 16#13,
39  | 16#2E, 16#29, 16#20, 16#27, 16#12, 16#15, 16#1C, 16#1B,
40  | 16#96, 16#91, 16#98, 16#9F, 16#8A, 16#8D, 16#84, 16#83,
41  | 16#DE, 16#D9, 16#D0, 16#D7, 16#CA, 16#CD, 16#C4, 16#C3,
42  | 16#E6, 16#E1, 16#E8, 16#EF, 16#FA, 16#FD, 16#F4, 16#F3];
43  END_VAR
44
45  u8CRC = proccrc8(u8CRC, u8Data[i]);
46  printf("CRC8 = %02X\n", u8CRC);

```

Funktionsbaustein „fbEnOceanReceiver“ die Methode „mInterpreter“ BH2

```

1  METHOD mInterpreter : BOOL
2  VAR_IN_OUT
3      tyBuffer          : tyComBuffer;
4  END_VAR
5  VAR
6      wIxCRC8D          : WORD;
7      wDataLen          : WORD; // Länge der Daten im Paket
8      byOptLen          : BYTE; // Länge der Optionalen-Daten im Paket
9      byCRC8H, byCRC8D  : BYTE; // CRC des Header und des Data
10 END_VAR

1  IF (mFindStartChar(tyBuffer)) THEN // Telegramm-Anfang suchen und prüfen ob mehr als 5 Zeichen empfangen wurden
2      byCRC8H := mCRC(ADR(tyBuffer.aData[1]), 4);
3      IF (byCRC8H = tyBuffer.aData[5]) THEN // Header Checksumme ist IO
4          wDataLen := (BYTE_TO_WORD(tyBuffer.aData[1]) * 16#0100) + tyBuffer.aData[2]; // Länge der Daten im Paket
5          byOptLen := tyBuffer.aData[3]; // Länge der Optionalen-Daten im Paket
6          tyMsg.byPacketType := tyBuffer.aData[4];
7          wIxCRC8D := (wDataLen + byOptLen) + 6; // +6 weil Daten 6 Byte hinter dem Startzeichen beginnen
8          IF (tyBuffer.wLen >= wIxCRC8D) THEN // Alle Zeichen bis zum Daten CRC empfangen ?
9              byCRC8D := mCRC(ADR(tyBuffer.aData[6]), WORD_TO_INT(wDataLen + byOptLen));
10             IF (byCRC8D = tyBuffer.aData[wIxCRC8D]) THEN // Daten Checksumme ist IO
11                 wMsgCnt := wMsgCnt + 1; // Zähler für die korrekten Telegramme incrementieren
12                 IF (tyMsg.byPacketType = 1) THEN
13                     IF (wDataLen >= 6) THEN // 6 ergibt sich aus 1 Byte für den Radio-Typ, 4 Byte für die Sender ID und 1 Byte für den Status
14                         tyMsg.byRadioTyp := tyBuffer.aData[6];
15                         IF (wDataLen > 6) THEN
16                             tyMsg.byUserDataLen := WORD_TO_BYTE(wDataLen - 6);
17                             SysMemCpy(ADR(tyMsg.aUserData), ADR(tyBuffer.aData[7]), tyMsg.byUserDataLen);
18                         ELSE
19                             tyMsg.byUserDataLen := 0;
20                         END_IF
21                         SysMemCpy(ADR(tyMsg.dwSenderId), ADR(tyBuffer.aData[7 + tyMsg.byUserDataLen]), 4);
22                         tyMsg.byStatus := tyBuffer.aData[11 + tyMsg.byUserDataLen];
23                         mInterpreter := TRUE;
24                     END_IF
25                 END_IF
26                 mRemoveData(wIxCRC8D + 1, tyBuffer);
27             END_IF
28         ELSE // Fehler in der Daten Checksumme => beim nächsten Aufruf wird ein neuer Telegrammanfang gesucht
29             mRemoveData(wCount := 1, tyBuffer);
30         END_IF
31     ELSE // Fehler in der Header Checksumme => beim nächsten Aufruf wird ein neuer Telegrammanfang gesucht
32         mRemoveData(wCount := 1, tyBuffer);
33     END_IF
34 END_IF
35

```

BH2

Verstehe ich nicht. In welchem Zusammenhang stehen die beiden Begriffe? Kommt da ein Komma dazwischen, oder vielleicht ein Doppelpunkt?
Ganz ohne Satzzeichen geht nicht.

Birgit Hess; 18.01.2016

„fbEnOceanReceiver“ die „privaten“ Methoden BH3

```

fbEnOceanReceiver.mFindStartChar
1 METHOD PRIVATE mFindStartChar : BOOL
2 VAR_IN_OUT
3   tyBuffer          : tyComBuffer;
4 END_VAR
5 VAR
6   i                  : WORD;
7 END_VAR
8
9 IF (tyBuffer.wLen > 5) THEN // es müssen mehr als 5 Zeichen empfangen worden sein
10  IF (tyBuffer.aData[0] = 16#55) THEN // ist das erste Zeichen schon ein 16#55?
11    mFindStartChar := TRUE; // wenn ja, kann die Funktion sofort beendet werden
12  ELSE
13    FOR i := 1 TO tyBuffer.wLen DO // sonst den Empfangsbuffer durchsuchen
14      IF (tyBuffer.aData[i] = 16#55) THEN
15        mRemoveData(i, tyBuffer); // Alle Zeichen vor dem 16#55 löschen
16        mFindStartChar := (tyBuffer.wLen > 5); // es müssen trotzdem mehr als 5 Zeichen sein
17      END_IF
18    END_FOR
19  END_IF
20 END_IF

fbEnOceanReceiver.mCRC
1 METHOD PRIVATE mCRC : BYTE
2 VAR_INPUT
3   pData              : POINTER TO ARRAY[0..255] OF BYTE;
4   iLen                : INT;
5 END_VAR
6 VAR
7   i                  : INT;
8 END_VAR
9
10 WHILE i < iLen DO
11   mCRC := aCRC[mCRC XOR pData[i]];
12   i := i + 1;
13 END_WHILE

fbEnOceanReceiver.mRemoveData
1 METHOD PRIVATE mRemoveData : BOOL
2 VAR_INPUT
3   wCount              : WORD;
4 END_VAR
5 VAR_IN_OUT
6   tyBuffer            : tyComBuffer;
7 END_VAR
8
9 IF (wCount > 0) THEN // es sollen Daten aus dem Buffer gelöscht werden
10  IF (wCount >= tyBuffer.wLen) THEN
11    SysMemSet(ADR(tyBuffer), 0, sizeof(tyBuffer));
12  ELSE
13    tyBuffer.wLen := tyBuffer.wLen - wCount;
14    SysMemMove(ADR(tyBuffer.aData[0]), ADR(tyBuffer.aData[wCount]), tyBuffer.wLen);
15  END_IF
16 END_IF

```

s. Kommentar S. 13: Wir brauchen ein Satzzeichen/eine Verbindung zwischen den beiden Begriffen.

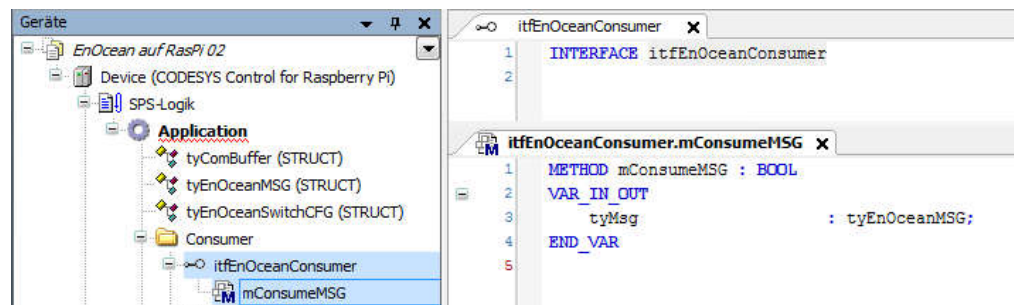
Birgit Hess; 18.01.2016

Zum Test ein paar Bausteine, die per EnOcean gesteuert werden

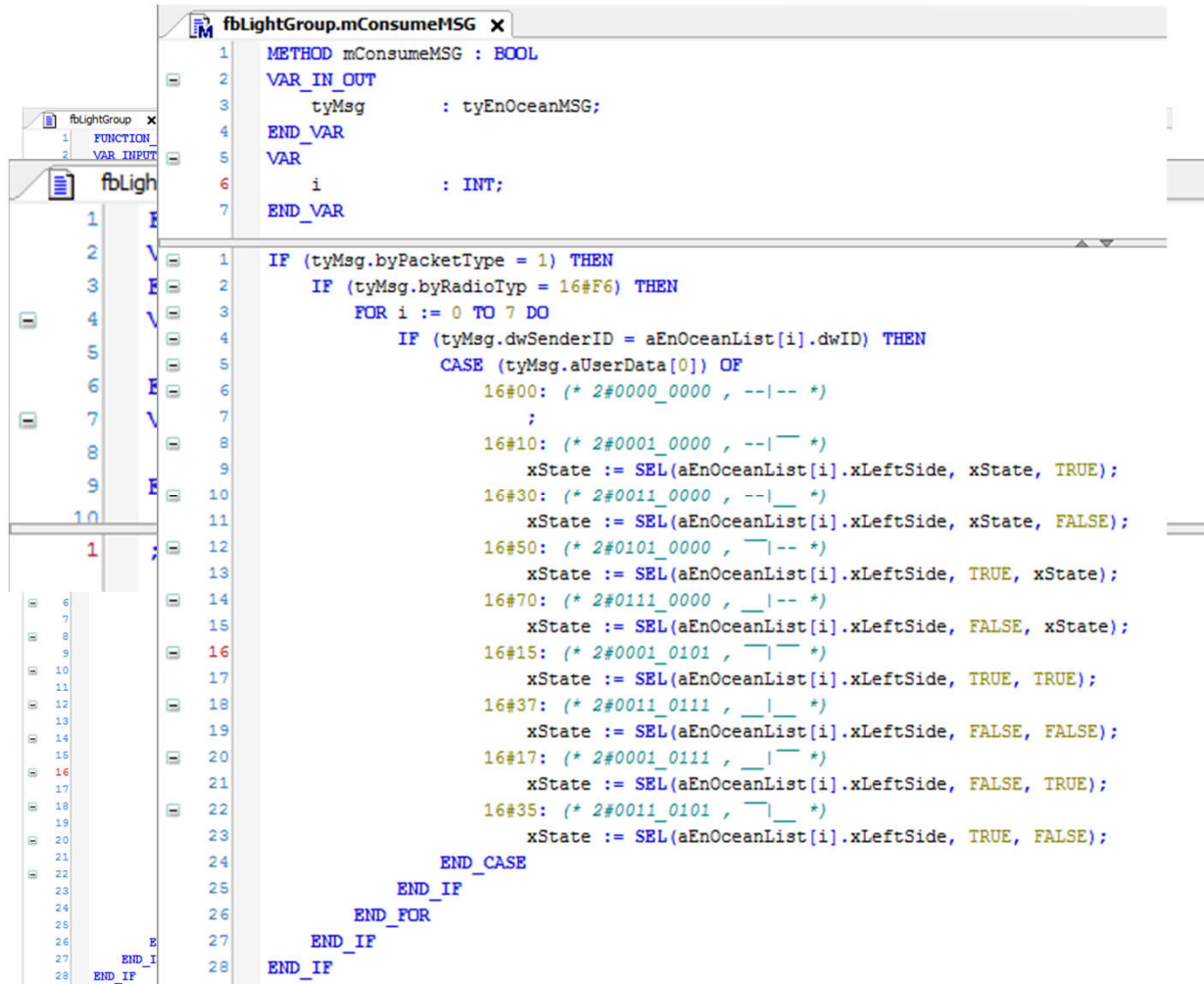
- Ideale Anwendung zur Demonstration der OOP (Objektorientierte Programmierung).
- Warum?
Weil ganz verschiedene Systeme EnOcean-Nachrichten verarbeiten können.
- Beispiele gefällig?
 - Lichtsteuerungen
 - Jalousiensteuerungen
 - Temperatur-/Klimaregelungen
 - Raum-/Gebäudeüberwachungen
 - ... u. v. a. m.
- Wie hilft OOP dabei?

Das Interface „itfEnOceanConsumer“

- Wir benutzen nur die einfachste Form der OOP – das Interface.
- Hiermit lassen sich unterschiedlichste Funktionsbausteine mit einer gemeinsamen Schnittstelle ausstatten und dann ganz einfach in einem ARRAY oder einer Liste verwalten.



Struktur „tyEnOceanSwitchCFG“ und FB „fbLightGroup“



```

1  METHOD mConsumeMSG : BOOL
2  VAR_IN_OUT
3      tyMsg      : tyEnOceanMSG;
4  END_VAR
5  VAR
6      i          : INT;
7  END_VAR

1  IF (tyMsg.byPacketType = 1) THEN
2      IF (tyMsg.byRadioTyp = 16#F6) THEN
3          FOR i := 0 TO 7 DO
4              IF (tyMsg.dwSenderId = aEnOceanList[i].dwID) THEN
5                  CASE (tyMsg.aUserData[0]) OF
6                      16#00: (* 2#0000_0000 , --|-- *)
7                          ;
8                      16#10: (* 2#0001_0000 , --|__ *)
9                          xState := SEL(aEnOceanList[i].xLeftSide, xState, TRUE);
10                     16#30: (* 2#0011_0000 , --|__ *)
11                         xState := SEL(aEnOceanList[i].xLeftSide, xState, FALSE);
12                     16#50: (* 2#0101_0000 , _|-- *)
13                         xState := SEL(aEnOceanList[i].xLeftSide, TRUE, xState);
14                     16#70: (* 2#0111_0000 , _|-- *)
15                         xState := SEL(aEnOceanList[i].xLeftSide, FALSE, xState);
16                     16#15: (* 2#0001_0101 , _|-- *)
17                         xState := SEL(aEnOceanList[i].xLeftSide, TRUE, TRUE);
18                     16#37: (* 2#0011_0111 , _|-- *)
19                         xState := SEL(aEnOceanList[i].xLeftSide, FALSE, FALSE);
20                     16#17: (* 2#0001_0111 , _|-- *)
21                         xState := SEL(aEnOceanList[i].xLeftSide, FALSE, TRUE);
22                     16#35: (* 2#0011_0101 , _|-- *)
23                         xState := SEL(aEnOceanList[i].xLeftSide, TRUE, FALSE);
24                 END_CASE
25             END_IF
26         END_FOR
27     END_IF
28 END_IF

```

Der Manager „fbEnOceanManager“ (macht nix selbst und bekommt am meisten Kohle)

```

fbEnOceanManager
1  FUNCTION_BLOCK fbEnOceanManager
2  VAR_INPUT
3      eComPort          : SysCom.COM_Ports;
4  END_VAR
5  VAR_OUTPUT
6  END_VAR
7  VAR
8      fbCom              : fbComHandle;
9      fbEOReceiver       : fbEnOceanReceiver;
10     aConsumer           : ARRAY[1..LAST_CONSUMER_IX] OF itfEnOceanConsumer;
11 END_VAR

fbEnOceanManager.mRegistConsumer
1  METHOD mRegistConsumer : INT // Positive Zahlen geben den Index in der internen Liste wieder, negaive einen eventuellen Fehler
2  VAR_INPUT
3      itfConsumer        : itfEnOceanConsumer;
4  END_VAR
5  VAR
6      i                  : INT;
7  END_VAR

1  FOR i := 1 TO LAST_CONSUMER_IX DO
2      IF (aConsumer[i] = itfConsumer) THEN // Consumer bereits in der Liste?
3          mRegistConsumer := i;
4          EXIT;
5      ELSIF (aConsumer[i] = 0) THEN // Erster freier Platz in der Consumer-Liste
6          aConsumer[i] := itfConsumer;
7          mRegistConsumer := iConsumerCnt := i;
8          EXIT;
9      END_IF
10 END_FOR
11
14 END_IF
15 END_IF
16 END_IF
17
18 END_IF
19 END_FOR

```

Ein kleines Programm zur Inbetriebnahme “PLC_PRG”

- Jetzt kommt der große Augenblick!
Bitte alles festschnallen!
- Wir nehmen die Software in Betrieb und
benötigen hierfür ein kleines PLC_PRG.

Ein kleines Programm zur Inbetriebnahme "PLC_PRG"

```
PLC_PRG x
1  PROGRAM PLC_PRG
2  VAR
3      fbLight001          : fbLightGroup := (aEnOceanList := [(dwID := 16#025E92100, xLeftSide := TRUE),
4                          (dwID := 16#014253662, xLeftSide := FALSE)]);
5      fbLight002          : fbLightGroup;
6      fbLight003          : fbLightGroup;
7
8      fbEOManager          : fbEnOceanManager;
9
10     xInit                : BOOL;
11 END_VAR

1  IF (xInit) THEN
2      fbEOManager(eComPort := SysCom.COM_Ports.SYS_COMPORT1);
3  ELSE
4      fbEOManager.mRegistConsumer(fbLight001);
5      fbEOManager.mRegistConsumer(fbLight002);
6      fbEOManager.mRegistConsumer(fbLight003);
7      xInit := TRUE;
8  END_IF
```

Schalter ID konfigurieren im Projekt

ID des Schalters einstellen

```

1  PROGRAM PLC_PRG
2  VAR
3      fbLight001          : fbLightGroup := (aEnOceanList := [(dwID := 16#DC752400, xLeftSide := TRUE),
4                          (dwID := 16#014257662, xLeftSide := FALSE)]);
5      fbLight002          : fbLightGroup := (aEnOceanList := [(dwID := 16#DC752400, xLeftSide := FALSE)]);
6      fbSwitch001         : fbSwitch    := (EnOceanID := 16#DC752400);
7      fbEOManager         : fbEnOceanManager;
8
9      xInit               : BOOL;
10 END_VAR

1  IF (xInit) THEN
2      fbEOManager(eComPort := SysCom.COM_Ports.SYS_COMPORT1);
3  ELSE
4      fbEOManager.mRegisterConsumer(fbSwitch001);
5      fbEOManager.mRegisterConsumer(fbLight001);
6      fbEOManager.mRegisterConsumer(fbLight002);
7      xInit := TRUE;
8  END_IF

```

