

ENOCEAN LINK WITH PROFILE EXTENSIONS

ADDING PROFILE DEFINITIONS TO ENOCEAN LINK

11 October 2026



Observe precautions! Electrostatic sensitive devices!

Patent protected:

WO98/36395, DE 100 25 561, DE 101 50 128,
WO 2004/051591, DE 103 01 678 A1, DE 10309334,
WO 04/109236, WO 05/096482, WO 02/095707,
US 6,747,573, US 7,019,241

ENOCEAN LINK WITH PROFILE EXTENSIONS

REVISION HISTORY

The following major modifications and improvements have been made to this document:

Version	Author	Reviewer	Date	Major Changes
1.0	MH	–	2010.10	Original version
1.1	MH	–	2014.10	Content update
2.0	AZ	JB, SA	2026.10	Document maintenance

Published by EnOcean GmbH, St.-Martin-Straße 76, 81541 Munich, Germany
www.enocean.com, info@enocean.com, phone +49 (89) 6734 689-0

© EnOcean GmbH, All Rights Reserved

Important!

This information describes the type of component and shall not be considered as assured characteristics. No responsibility is assumed for possible omissions or inaccuracies. Circuitry and specifications are subject to change without notice. For the latest product specifications, refer to the EnOcean website: <http://www.enocean.com>.

As far as patents or other rights of third parties are concerned, liability is only assumed for modules, not for the described applications, processes and circuits.

EnOcean does not assume responsibility for use of modules described and limits its liability to the replacement of modules determined to be defective due to workmanship. Devices or systems containing RF components must meet the essential requirements of the local legal authorities.

The modules must not be used in any relation with equipment that supports, directly or indirectly, human health or life or with applications that can result in danger for people, animals or real value.

Components of the modules are considered and should be disposed of as hazardous waste. Local government regulations are to be observed.

Packing: Please use the recycling operators known to you.

ENOCEAN LINK WITH PROFILE EXTENSIONS

TABLE OF CONTENTS

1 Introduction..... 4

1.1 Definitions 4

1.2 References 5

2 Profile Definitions in EnOcean Link 6

2.1 EnOcean Equipment Profiles 6

2.2 Profile Factory 8

3 Adding or Changing Profiles in EnOcean Link..... 8

3.1 Example: Manufacturer Specific Profile Changes 10

LIST OF FIGURES

Figure 1 eoEEPProfile collaboration diagram 6

Figure 2 Inheritance diagram of eoD2EEPProfile (RORG = 0xD2) 7

LIST OF TABLES

Table 1 Terms and abbreviations..... 4

Table 2 Fields of the EEP_ITEM structure 7

ENOCEAN LINK WITH PROFILE EXTENSIONS

1 Introduction

EnOcean Link [1] includes the implementation of all current EnOcean Equipment Profiles (EEP) [3]. This application note is intended for use cases in which special profiles must be added to EnOcean Link, for example:

- Manufacturer specific profiles (MSC telegrams) [3]
- Profiles in development that will be submitted to the EnOcean Alliance
- Other profile extensions

When the EnOcean Alliance has approved a new profile, the profile is included in the next EnOcean Link release. This application note shows how to test and benchmark a profile with EnOcean Link before this approval.

The source code of EnOcean Link is part of the delivery. Developers can therefore adapt EnOcean Link as required. This application note describes in detail how to add or change profile implementations in EnOcean Link.

1.1 Definitions

Table 1 lists the terms and abbreviations used in this document.

Table 1 Terms and abbreviations

Term / Abbr.	Description
μC	Microcontroller (external)
4BS	4 Byte Communication (RORG 0xA5)
AES	Advanced Encryption Standard
API	Application Programming Interface
APP	Application
ASK	Amplitude Shift Keying
CBC	Cipher Block Chaining
CMAC	Cipher-based Message Authentication Code
CRC	Cyclic Redundancy Check
DATA	Payload of a radio telegram
DB	Data byte of a telegram (for example DB0, DB1)
Device	Customer end device with an integrated EnOcean radio module
DoS	Denial of Service
EEP	EnOcean Equipment Profile
EHW	Energy Harvested Wireless protocol
ERP	EnOcean Radio Protocol (ERP1 = version 1, ERP2 = version 2)
ESP3	EnOcean Serial Protocol version 3
FSK	Frequency Shift Keying
FUNC	Second part of the EEP number: basic functionality of the data content
Gateway	Module with bidirectional serial communication, connected to a host

ENOCEAN LINK WITH PROFILE EXTENSIONS

Term / Abbr.	Description
GP	Generic Profiles
ID	Unique module identification number
KEY	Specific parameter used to encrypt, decrypt or transform DATA
MAC	Message Authentication Code
MAN_ID	Manufacturer ID
MSB	Most Significant Byte
MSC	Manufacturer Specific Communication
PSK	Pre-shared Key
PTM	Pushbutton Transmitter Module
RLC	Rolling Code
RORG	Radio organization (R-ORG): message parameter that identifies the telegram type, first part of the EEP number
RPS	Repeated Switch Communication (RORG 0xF6)
SLF	Security Level Format, specifies which security parameters are used
TXID	ID of a transmitter
TYPE	Third part of the EEP number: type of the device
VAES	Variable AES
VLD	Variable Length Data (RORG 0xD2)

1.2 References

- [1] EnOcean Link, product page: enoclean.com/en/product/enoclean-link
- [2] EnOcean Link User Manual (API reference): enoclean.com/.../support/enoclean-link
- [3] EnOcean Equipment Profiles (EEP) Specification: enoclean-alliance.org/eeep
- [4] Generic Profiles (GP) Specification: enoclean-alliance.org/gp
- [5] EnOcean Radio Protocol 1 (ERP1): enoclean.com/.../EnOceanRadioProtocol1.pdf
- [6] EnOcean Serial Protocol 3 (ESP3): enoclean.com/esp
- [7] Gateway Controller, product page: enoclean.com/en/product/gateway-controller
- [8] EnOcean Wireless Standard: enoclean-alliance.org/.../enoclean-wireless-standard
- [9] Security of EnOcean Radio Networks Specification: enoclean-alliance.org/security
- [10] EnOcean Application Notes: enoclean.com/en/support/application-notes
- [11] Factory design pattern: oodeesign.com/factory-pattern

ENOCLEAN LINK WITH PROFILE EXTENSIONS

2 Profile Definitions in EnOcean Link

In EnOcean Link, the term profile definition covers:

- EnOcean Equipment Profiles (EEP) [3]
- Generic Profiles (GP) [4]

Generic Profiles already use a self-descriptive language. Each GP device is specific, but it is described in an abstract way. EnOcean Link can therefore interpret GP devices without additional changes. Changes to the Generic Profiles language itself are complex and are not part of this application note.

2.1 EnOcean Equipment Profiles

This application note focuses on extensions of EnOcean Equipment Profiles (EEP). An EEP is identified by three numbers:

- RORG
- FUNC
- TYPE

An EEP defines how the data of a received telegram is parsed to get the application values. Bidirectional profiles also define the structure of the transmitted telegrams. In EnOcean Link, profiles are represented by classes. The base class for all EEPs is [eoEEPProfile](#) [2]. Figure 1 shows the collaboration diagram of this class.

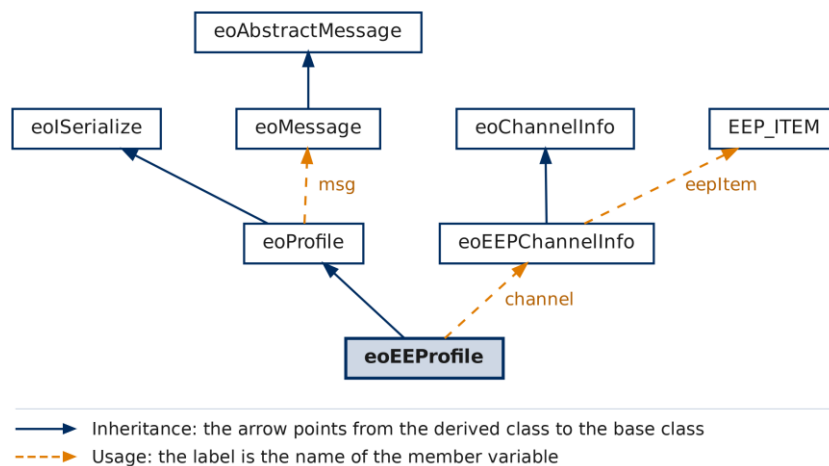


Figure 1 `eoEEPProfile` collaboration diagram

The profile classes are first grouped by their RORG, for example:

- RORG = RPS (0xF6): [eoF6EEPProfile](#)
- RORG = 4BS (0xA5): [eoA5EEPProfile](#)
- RORG = VLD (0xD2): [eoD2EEPProfile](#)

The profiles are then grouped by their FUNC, for example:

- RORG = 0xA5, FUNC = 0x05: [eoEEP_A505xx](#)
- RORG = 0xA5, FUNC = 0x02: [eoEEP_A502xx](#)

ENOCLEAN LINK WITH PROFILE EXTENSIONS

These FUNC classes inherit from the RORG classes listed above. The specific profile is implemented in the FUNC class. Figure 2 shows the inheritance diagram for RORG = 0xD2.

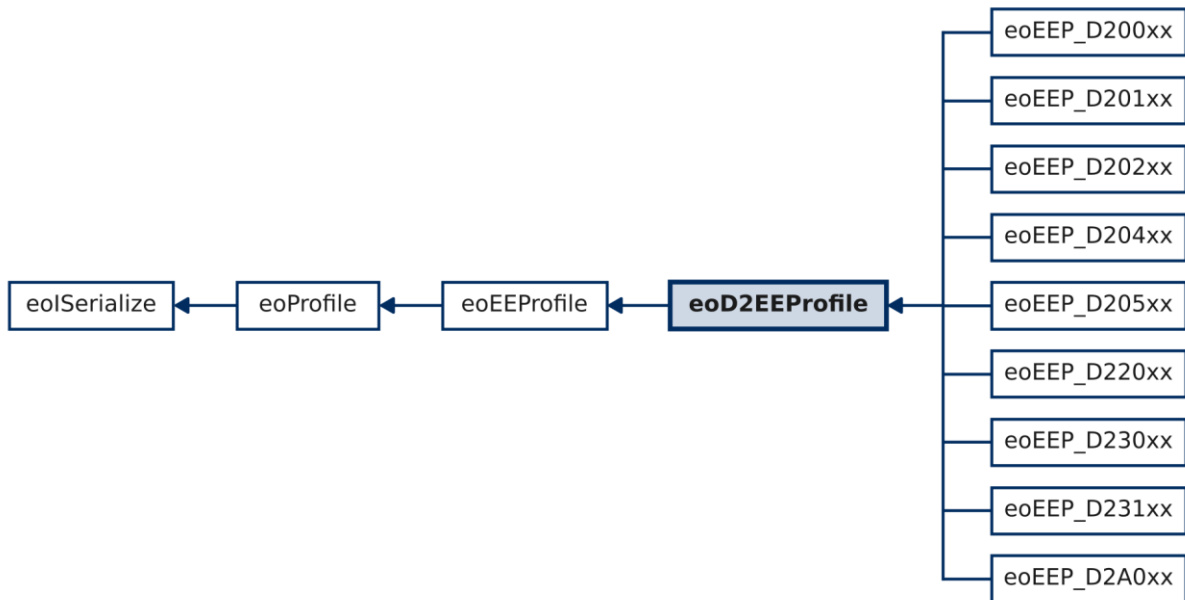


Figure 2 Inheritance diagram of `eoD2EEPProfile` (RORG = 0xD2)

The `GetChannel()` function gives access to the values of a received telegram. The channels are parsed with a predefined matrix of parameters, for example bit offset, resolution, scaling and physical unit. Each entry of this matrix is an [EEP_ITEM](#) structure. Table 2 describes its fields.

Table 2 Fields of the `EEP_ITEM` structure

Field	Data type	Description
exist	bool	Channel is available (true) or not used (false)
bitoffs	uint16_t	Offset of the data value in the telegram, in bits
bitsize	uint16_t	Size of the data value in the telegram, in bits
rangeMin	uint64_t	Minimum raw value in the telegram
rangeMax	uint64_t	Maximum raw value in the telegram
scaleMin	double	Application value that matches rangeMin
scaleMax	double	Application value that matches rangeMax
type	CHANNEL_TYPE	Channel type, for example S_VOLTAGE, S_LUMINANCE or S_TEMP
index	uint8_t	Subchannel index, only needed when two channels have the same CHANNEL_TYPE

About 70% of all EEPs can be represented and parsed with the `EEP_ITEM` matrix. The matrix closely follows the EEP XML file provided by the EnOcean Alliance [3].

ENOCAN LINK WITH PROFILE EXTENSIONS

Experience with profile parsing shows that the remaining 30% of the profiles use special parsing logic that does not fit into the abstract matrix structure. For these profiles, EnOcean Link uses specific parsing routines that follow the definitions of the EnOcean Alliance members. These routines are implemented as profile-specific source code in the profile class.

Parsing is therefore based on one of the following:

- `EEP_ITEM` matrix: common parsing routine, defined in the `eoEEPProfile` class
- Profile specific source code: implemented in the class of the respective profile

2.2 Profile Factory

A specific profile instance is assigned to a device instance with a pointer. The profile is created when a teach-in telegram is received, or it is created manually. For details, see the EnOcean Link user manual [2]:

- [eoTeachInModule.h file reference](#)
- [eoProfileFactory class reference](#)

The `eoProfileFactory` class follows the factory design pattern [11]. Its `CreateProfile()` function is called during teach-in. This function creates the matching profile and returns it. To use new profiles during teach-in, add them to the `eoProfileFactory` class.

3 Adding or Changing Profiles in EnOcean Link

The following options are available to add a profile:

- Add MSC profiles (purely manufacturer specific profiles)
- Add profiles with a new FUNC and a new TYPE for an existing RORG
- Add profiles with a new TYPE for an existing FUNC and RORG
- Change or extend existing profiles with a manufacturer specific identification (MAN_ID)

For MSC profiles, it is recommended to define your own profile classes instead of extending the existing classes, because MSC telegrams are purely manufacturer specific.

All other use cases can easily be implemented by extending the existing classes or by adding new classes.

If the new profile supports one of the teach-in procedures defined in the EEP specification [3], it is only necessary to add the profile to the `eoProfileFactory` class.

First, include the header file of the new class, if it is not already included in an existing file:

```
...
#include "eoEEP_D205xx.h"
#include "eoEEP_D220xx.h"
#include "eoEEP_D230xx.h"
#include "eoEEP_D231xx.h"
...
#include "eoEEP_NEW_PROFILE.h"
```

ENOCEAN LINK WITH PROFILE EXTENSIONS

Then add the profile to the switch/case statement of the respective RORG:

- For MSC profiles, add the profile to the `CreateProfile(const eoMessage &msg)` function.

```
eoProfile* eoProfileFactory::CreateProfile(const eoMessage &msg)
{
    switch (msg.RORG)
    {
        case RORG_4BS:
            if ((msg.data[3] & 0x88) == 0x80) // Is Learn telegram?
            {
                uint8_t rorg = RORG_4BS;
                uint8_t func = msg.data[0] >> 2;
                uint8_t type = ((msg.data[0] & 3) << 5) | (msg.data[1] >> 3);

                eoProfile *profile =
                    eoProfileFactory::CreateProfile(rorg, func, type);
                if (profile != NULL)
                    profile->manufacturer = (msg.data[1] & 7) << 8 | msg.data[2];

                return profile;
            }
            break;
        ...
        case RORG_MSC:
            // OWN MSC HANDLING
        ...
    }
}
```

- For a new FUNC, add the required create call to the `CreateProfile(const uint8_t rorg, const uint8_t func, const uint8_t type)` function.

```
eoProfile* eoProfileFactory::CreateProfile(const uint8_t rorg,
                                           const uint8_t func,
                                           const uint8_t type)
{
    eoProfile *eep = NULL;
    switch (rorg)
    {
        ...
        case RORG_4BS:
            switch (func)
            {
                case 0x02:
                    eep = new eoEEP_A502xx();
                    break;
                case 0x04:
                    eep = new eoEEP_A504xx();
                    break;
                case 0x05:
                    eep = new eoEEP_A505xx();
                    break;
                case 0x06:
                    eep = new eoEEP_A506xx();
                    break;
                case 0xXX:
                    eep = new eoEEP_A5XXxx(); // NEW FUNC
                    break; // CREATE NEW FUNC
            }
        ...
    }
}
```

ENOCEAN LINK WITH PROFILE EXTENSIONS

The TYPE is selected later by the `SetType()` function of the specific profile class. This function is inherited from the [eoProfile](#) class.

If you add a new TYPE to an existing FUNC, add the `EEP_ITEM` matrix entries for this TYPE to the class implementation (cpp file). The `EEP_ITEM` matrix for RORG = 0xA5 and FUNC = 0x06 ([eoEEP_A506xx](#)) is:

```
const EEP_ITEM listA506xx[numOfProfiles][numOfChan] =
{
// exist, bitoffs, bitsize, rangeMin, rangeMax, scaleMin, scaleMax, type, index
//TYPE:00 - DOES NOT EXIST, ONLY PLACE HOLDER
{
//Temperature
{ false, 16, 8, 255, 0, -40.0, 00.0, S_RES, 0 },
{ false, 16, 8, 255, 0, -40.0, 00.0, S_RES, 0 },
{ false, 16, 8, 255, 0, -40.0, 00.0, S_RES, 0 }, },
//TYPE:01
{
{ true, 0, 8, 0, 255, 0, 5.1F, S_VOLTAGE, 0 }, //Voltage
{ true, 8, 8, 0, 255, 300, 30000, S_LUMINANCE, 0 }, //Light
{ true, 16, 8, 0, 255, 600, 60000, S_LUMINANCE, 0 }, //Light
},
//TYPE:02
{
{ true, 0, 8, 0, 255, 0, 5.1F, S_VOLTAGE, 0 }, //Voltage
{ true, 8, 8, 0, 255, 0, 510, S_LUMINANCE, 0 }, //Light
{ true, 16, 8, 0, 255, 0, 1020, S_LUMINANCE, 0 }, //Light
},
//TYPE:03
{
{ true, 0, 8, 0, 255, 0, 5.1F, S_VOLTAGE, 0 }, //Voltage
{ true, 8, 10, 0, 1000, 0, 1000, S_LUMINANCE, 0 }, //Light
{ false, 16, 8, 255, 0, -40.0, 00.0, S_RES, 0 }, }, },
};
```

If the profile needs special parsing logic, add this source code to the `SetValue()` function of the profile class. Otherwise, the inherited `SetValue()` function is called, and the values are parsed according to the `EEP_ITEM` matrix.

Manufacturer-specific changes of existing profiles can also be implemented in the profile class, for example in the `SetValue()` or `SetType()` function. In this case, check the manufacturer attribute of the [eoProfile](#) class.

3.1 Example: Manufacturer Specific Profile Changes

The source files `eoEEP_A508xx.cpp` and `eoEEP_A506xx.cpp` provided with this application note contain a reference implementation of manufacturer-specific changes of existing profiles by Eltako. In this implementation, the `SetType()` function replaces the `EEP_ITEM` matrix entries when the manufacturer ID is 0x00D.

Occupancy sensor FBH55, FBH63

- EEP: A5-08-01
- The teach-in telegram includes the manufacturer ID 0x00D.

ENOCEAN LINK WITH PROFILE EXTENSIONS

The following changes were made:

- Illumination scaling extended from 0 to 510 lux to 0 to 2048 lux
- No temperature value in DB1
- No occupancy button in DB0, bit 0

Illumination sensor FAH60

- EEP: A5-06-01
- The teach-in telegram includes the manufacturer ID 0x00D.

The following changes were made:

- DB3: illuminance 0 to 100 lux, linear, n = 0x00 to 0x64 (valid when DB2 = 0x00)
- DB2: illuminance 300 to 30,000 lux, linear, n = 0x00 to 0xFF
- DB1: not used

ENOCEAN LINK WITH PROFILE EXTENSIONS

DISCLAIMER

The information provided in this document describes typical features of the EnOcean radio transmission system and should not be misunderstood as specified operating characteristics. No liability is assumed for errors and / or omissions. We reserve the right to make changes without prior notice. For the latest documentation visit the EnOcean website at **www.enocean.com**.