NFC Memory Structure for Eco-system products

REVISION HISTORY

The following major modifications and improvements have been made to the first version of this document:

No	Who	Major Changes
1.0	MH	Submission to TWG

Important!

The original version of this document (version 1.0 from date 21.04.2020) has been created by EnOcean GmbH (Oberhaching, Germany) and has been submitted to the EnOcean Alliance Inc. as a standard Technical Specification for use by EnOcean Alliance membership according to the By Laws and Membership Agreements of the EnOcean Alliance, including the EnOcean Alliance Intellectual Property Rights (IPR) Policy, enabling open non-discriminatory use within, and improved synergies and interoperability within and beyond the EnOcean Alliance eco-system.

It is agreed between EnOcean GmbH and EnOcean Alliance Inc. that any subsequent updates of this specification created within the EnOcean Alliance (as defined by ratified and publicized by the EnOcean Alliance Board of Directors) may also be used by EnOcean GmbH in any of their market offerings, such as but not limited to modules, products, tools or systems, also including such offerings using wireless protocols not part of or supported by

participant or promoter member of the EnOcean Alliance and/or joint owner of the appropriate joint IPR.

These errata may not have been subjected to an Intellectual Property review, and as such, may contain undeclared Necessary Claims.

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 Alliance website: http://www.enocean-alliance.org

Published by EnOcean Alliance Inc.

5000 Executive Parkway, Suite 302
San Ramon, CA 94583 USA

Graham Martin

Chairman & CEO EnOcean Alliance

TABLE OF CONTENT

1 Introduction

1.1 Scope

This document describes the structure of the NFC Memory Structure of products in the EnOcean Alliance Eco-system. Its scope is to provide common specifications for all devices, which include a NFC interface for configuration / set up. The specification aims to allow a maximum of freedom, to enable different products

In an eco-system there are parties, which produce devices, and other companies, that build tools to incorporate them. The need for common definitions emerges from integrators, which are required to incorporate NFC based products from different companies into their solution. By having common definitions, the effort for integrators shall be reduced to a minimum having them to focus only on the application specific parts of the device.

Device makers complying with these standards need to present the parameters / link tables in the defined structure. The format how to describe the parameter, length, possible values etc. is provided. The meaning of the parameters, behavior of the device itself is not scope of the specification.

1.2 Target audience

This document is for device makers and companies, which incorporate these devices into their solutions.

By devices, we understand: sensors, switches, actuators, controllers and gateways.

Solutions where these devices are incorporated include: commissioning tool to set up devices (PC or Smartphone), configuration SW to set parameters (PC or Smartphone).

1.3 Document structure

The document is structured into three parts:

1) Mandatory requirements for NFC based devices – definitions that each NFC device must comply with.
2) Format definitions that should be used to describe configuration parameters / link tables etc., which are application specific and not part of the mandatory requirements.
3) Best practice examples – set of examples and recommendations on how to address specific challenges, which arise with the usage of NFC Interface for configuration.

1.4 References

EnOcean Alliance Labeling & Product ID instructions:

1. https://www.enocean-alliance.org/specifications/

2 Specification

Each device shall have a NDEF[1] field with following parameters:

- Well known type
- Text type

Structure of the NDEF fields shall comply with the EnOcean Alliance labeling instructions 1. It shall include especially these fields:

Identifier	Identifier length	Length of data excluding identifier	Value
MANDATORY:			
6P	2 characters	15 characters or 3 characters	„ENOCEANALLIANCE“ or „ENO”– identification of used standard
12Z	3 characters	XX characters	NFC Forum ID – see 1.
3C	2 characters	XX characters	Address of the EnOcean NFC header as hex number reflecting the address from NFC reader interface.
30S	3 characters	12 characters	EURID
1P	2 characters	12 characters	Product ID
Optional			
31Z	3 characters	8 characters	NFC PIN for protected area.

2.1 EnOcean NFC header

At the address specified by field “3C” the header is located. The header provides detailed information about the device NFC structure and revision.

It consists of following fields:

Offset (byte)	Size (byte)	Data	Description	Valid Range	Scale	Unit
0	1	Start of header	Constant.	0xE0		
1	1	Length	Length indication of complete header.	1..255	1..255	x
2	1	Version	Version definition of header.	Enumeration: 0x00: Reserved		

1 https://nfc-forum.org/our-work/specifications-and-application-documents/specifications/nfc-forum-technical-specifications/

				0x01: Version 1 defined 0x02 – 0xFF: Reserved
3	2	Man ID	Assigned Manufacturer ID.	See Man ID Specification.
5	3	NFC Struct ID	Manufacturer assigned ID to reference the NFC Structure he applied in the product.	Enumeration: Manufacturer specific. Assigned and maintained by the manufacturer.
8	1	RevisionA	Revision of the products. One revision must be specified. If only one revision is listed, all previous revisions are fully supported.	Enumeration: 0x01 – 0xFD: Manufacturer specific. Assigned and maintained by the manufacturer. 0x00, 0xFE, 0xFF: Reserved
6	1 * X	Revision_Other	Older revisions of the product. Older revision is last field with highest offset.	Enumeration: 0x01 – 0xFD: Manufacturer specific. Assigned and maintained by the manufacturer. 0x00, 0xFE, 0xFF: Reserved
6+ 1*x	1	Revision Block End	Constant signalize END of revision other block might follow later	Enumeration: 0xFE – Constant

X = count of revisions – 1. The size of X (number of revisions) is manufacturer specific

Reference implementation on NT3H2111 example (16 byte block):

Address		0 4 8 12	1 5 9 13	2 6 10 14	3 7 11 15
DEC	HEC				
44	2C	0xE0	Length	Version	
45	2D	Man ID	NFC Struct ID		
46	2E	Rev 2	Rev 1	0xFE	
47	2F				

2.2 Revision tracking

A NFC Struct ID refers as one number to the complete remaining structure of the remaining NFC Memory. Since the Manufacturer ID is present, each manufacturer manages the field of NFC Struct ID himself. With small changes only a revision needs to be updated. One revision needs to be always present. Additional revision can be defined e.g. product variants. The latest revision is in the first field. If only one revision is listed all previous revision are supported. If a list of revision is present, the order is descending.

A revision represents the state of one specific NFC Struct ID. Revision can be used also to define custom variants of a product (e.g. a customer variant with special parameter).

Revision tracking provides a great tool in control of backward compatibility and ensuring forward compatibility on the tool side. Following rules apply:

- The memory structure must be compatible to every revision it is listing.
- Changes between revisions shall only be extending reserved fields or extending lists, not redefining existing structures.
- Only additions to fields are allowed and no subtractions.
- If changes are not backward compatible to any previous revisions, then a new NFC Struct ID shall be defined and the revision restarted.
- Most recent revision number is defined on the first entry.
- Revision starts at 1 and by doing new updated it is incremented by 1.
- Each revision represents a complete and final state of the NFC Structure.
- Logical or functional dependencies between revisions shall not be present.

3 NFC Memory - Form for description

The manufacturer defines the NFC structure of the products. He does not need to apply or get permission to use a specific field from the EnOcean Alliance. The complete definition and management of the remaining NFC Memory is up to the manufacturer.

The manufacturer needs to provide documentation about fields and structures he is using, the meaning, and possible values. To support common description and interoperability we provide below a structure and format definition for description.

By using one common description for the structure, it is easier for integrators to understand new products.

Description structure as seen from an NFC interface

Page DEC	Page HEC	0 4 8 12	1 5 9 13	2 6 10 14	3 7 11 15
0	0x0				
1	0x1				
2	0x2				
3	0x3				
4	0x4				
5	0x5				
6	0x6				
7	0x7				
8	0x8				
9	0x9				
10	0xA				
11	0xB				

Describe meaning – lists

Offset (byte)	Size (byte)	Data	Description	Valid Range	Scale	Unit
0	4			Enumeration:		
4	…			….	…	…

4 Best practice examples - containers

Below definitions are provided to support compatibility of devices in the eco-system and make design-in of such devices by the system integrator easier. They are listed as best practice examples but are not having a specification (requirement) character yet.

The description is separated into abstract containers – all of them are located in the same EEPROM memory of the NFC.

4.1 Semaphore

A semaphore is a special container, which does not have application data but helps to organize the communication flow between the NFC Configuration tool and the application.

A semaphore can include:

- Access flag
- Checksum for structural integrity
- Revision of the tool which executed the write operation

4.1.1 Access flag

Since the application might be offline and does not realize changes in EEPROM memory there is a strong need to signalize to the application that an EEPROM write action has been executed. This way the application can check for modifications by only reading the access flag byte and not the complete memory.

The EEPROM memory can be separated into several logical containers and the containers might have additionally own subgroups. An access flags can be present for each container or subgroup.

Access flag makes sense only for writeable parameters via NFC interface.

The application is responsible for reading the access flags as soon possible. It needs to review the changes and if the changes are valid it needs to use the new parameters for its operation. Additionally, the access flags shall be set to the “initial value” afterwards.

4.1.2 Checksum

The NFC Protocol itself has included integrity checks, but it might still occur that a write multi-pages write operation is interrupted and at least one page write operation was not executed / finished. In this case the device NFC memory has an undefined stage as not all parameters were written correctly.

To detect issues in write operation a CRC calculated over the complete container shall be written in a separate page. By counting the CRC of the container and comparing it to the one in semaphore the application can know if the structural integrity is still present.

4.1.3 Revision of tool

In the NFC Header the revision of the current product is stored. This way the tool knows which revision the device has and if multiple revisions are specified the tool can select which revision it supports. The device on the other hand does not know with what revision knowledge was the tool configuring.

After a write operation of a NFC tool, it can include inside the semaphore container the revision number which the tool used as baseline at the configuration action. Then the device will be informed about the tool revision and can optionally execute added actions e.g. if a lower revision was used by the tool the device might set the "newer" parameters which were not considered by the tool since it used an earlier revision.

If the revision does not have to be used

4.1.4 Semaphore example

In Structure below more semaphores are used since there can be several containers and it is better practice to use a separate semaphore for each of them.

		0	1	2	3
		4	5	6	7
Address		8	9	10	11
DEC	HEC	12	13	14	15
4	0x4	FLAG	REVISION_TOOL	CRC16	
5	0x5	FLAG	REVISION_TOOL	CRC16	
6	0x6	FLAG	REVISION_TOOL	CRC16	
7	0x7				

Lists

Offset (byte)	Size (byte)	Data	Description	Valid Range	Scale	Unit
0	1	FLAG	Signal if something changed in the container. After read the application resets the flag.	Enumeration: 0x01 Change pending 0x02 NO changes pending & no information about last write available. 0x03 No changes pending & Last write was accepted by the application 0x04 No changes pending & last write failed.		

1	1	REVISION_TOOL	Revision of the product which the tools supports	Enumeration: 0x01 – 0xFD Revision 0x00,0xFE,0xFF Reserved
2	2	CRC16	Checksum of the container	Enumeration: 0x0000 – 0xFFFF CRC16

4.2 Identification of the tool

Additionally, to the revision of the tool inside the semaphore we define a string container where additional information about the Tool executing the NFC operations with the end device is stored.

Existence and location of the "Installation string" container shall be connected with the specific Struct ID and specified inside the product documentation.

Inside the String following information should be included:

1. Local Date / time of the write operation.
2. Unambiguous tool name
3. Tool revision

Optionally additional information about the user, device, location and environment where the operation was executed can be stored.

The String shall be encoded in UTF-8.

The String shall be formed with the respect of the labeling specification 1.

Following fields should be used:

Identifier	Identifier length	Length of data excluding identifier	Value
MANDATORY:			
22D	3 characters	12 characters	Date and time. YYYYMMDDHHMM (24 hour clock - UTC).
10S	3 characters	XX characters	Unambiguous tool name.
16S	3 characters	XX characters	Version Number, e.g., Software Version.

5 Implementation aspects

5.1 Dual EEPROM Architecture

As a final application can have one or multiple EEPROM memories. An additional EEPROM memory e.g. EEPROM in the application processor can have different processing logic. The EEPROM in the NFC Interface can be used only for storage of the values or also as working memory.

It is important to consider that the end user shall only see and know about the NFC interface. If internally values are copied to an internal EEPROM, the user manual needs to describe e.g. that after a change of values the Product needs some seconds for an update and blinks to confirm an update.

5.2 Final Product ID

In case the product is an intermediate product and another manufacturer is using the intermediate as part of his final product the final manufacturer has to have a possibility to change the PRODUCT ID located in the NDEF Field. e.g. module PTM 210.

Since the final product has additional modules, parts additionally to the intermediate product, a change of the Product ID of the intermediate product to the final product ID, which references the final product, is a very recommended approach.

5.2.1 Example approach

The Product ID can be entered as a parameter in the configuration interface container, which can be protected by the NFC password. The application can the change the NDEF field from within the application and update the Product ID. This way the NDEF field can stay read only and other values in the NDEF field are not subject of a potential harmful or unwanted alteration.

5.3 User defined message

A user message can be included inside the device e.g. to indicate positioning instructions “switch second flow, kitchen” then this message should be part of the free to read NDEF field of the device. This way the personal doing only part of the installation do not need to have the NFC Pin and so the full access to the device.