A picture containing chart

Description automatically generated

Report of DeseNET Laboratory

DeSEm – Design of communicative embedded systems

Schnyder Nicolas

Delivery date: 11.01.2023

Content

[1 Introduction 3](#_Toc124365496)

[1.1 Objective 3](#_Toc124365497)

[1.2 DeseNET structure 3](#_Toc124365498)

[2 Architecture 4](#_Toc124365499)

[2.1 Multi PDU 4](#_Toc124365500)

[2.2 Joystick Application 4](#_Toc124365501)

[3 Design 5](#_Toc124365502)

[3.1 Initialization and synchronization 5](#_Toc124365503)

[3.2 Beacon reception 6](#_Toc124365504)

[4 Tests 7](#_Toc124365505)

[4.1 Scenarios 7](#_Toc124365506)

[4.2 Results on Mesh simulator 9](#_Toc124365507)

[4.3 Results on real target 10](#_Toc124365508)

[5 Conclusion 12](#_Toc124365509)

[6 Appendix 13](#_Toc124365510)

[6.1 Frame observation of Mesh simulator test series 13](#_Toc124365511)

# Introduction

## Objective

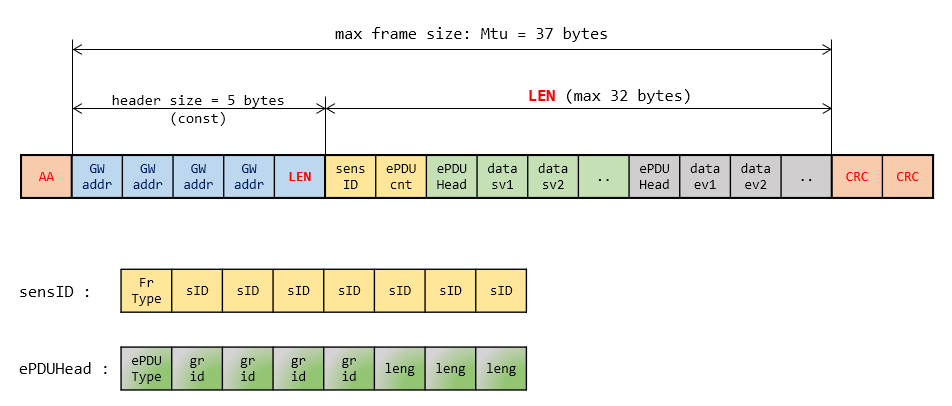
The DeseNET protocol is a real-time communication protocol, developed by the Hes-so Valais/Wallis, for a reliable and low-powered data exchange over Wi-Fi.[[1]](#footnote-2)

The objective of this laboratory is to implement the necessary software requirements, to establish a functional connection between a master gateway and a slave sensor. Thereby, the sensor consists of a 3-axis accelerometer and a 4-button joystick (up, down, left, right, centre).

To facilitate the development process a “Mesh simulator” was provided, so that the real target (a NUCLEO L476RG evaluation board) can be simulated. For both, simulator and target, several test conditions were defined and conducted to ensure the proper functioning of the proposed solution.

## DeseNET structure

The DeseNET protocol requires to send so-called MPDUs (Multi Protocol Data Units) in the following format:



Legend:

|  |  |
| --- | --- |
| AA, CRC | Unused |
| GW addr | Gateway address (const) |
| LEN | Usable data length (max 32 bytes) |
| Sens ID | Sensor ID (const) |
| ePDU cnt | Number of ePDUs (embedded PDUs) in the frame |
| ePDU head | Header of an ePDU (type, SV group or EV ID, length) |
| data sv, ev | SV or EV data |

Hence, the general objective is to create and send such frames.

# Architecture

The most part of the system was already working and handed-out to form the template of the work. Hence, only the self-made classes and their interaction with some existing classes are described here-below. The complete diagram can be found in the delivery folder in PDF format.

## Multi PDU

Diagram

Description automatically generated

## Joystick Application

Diagram

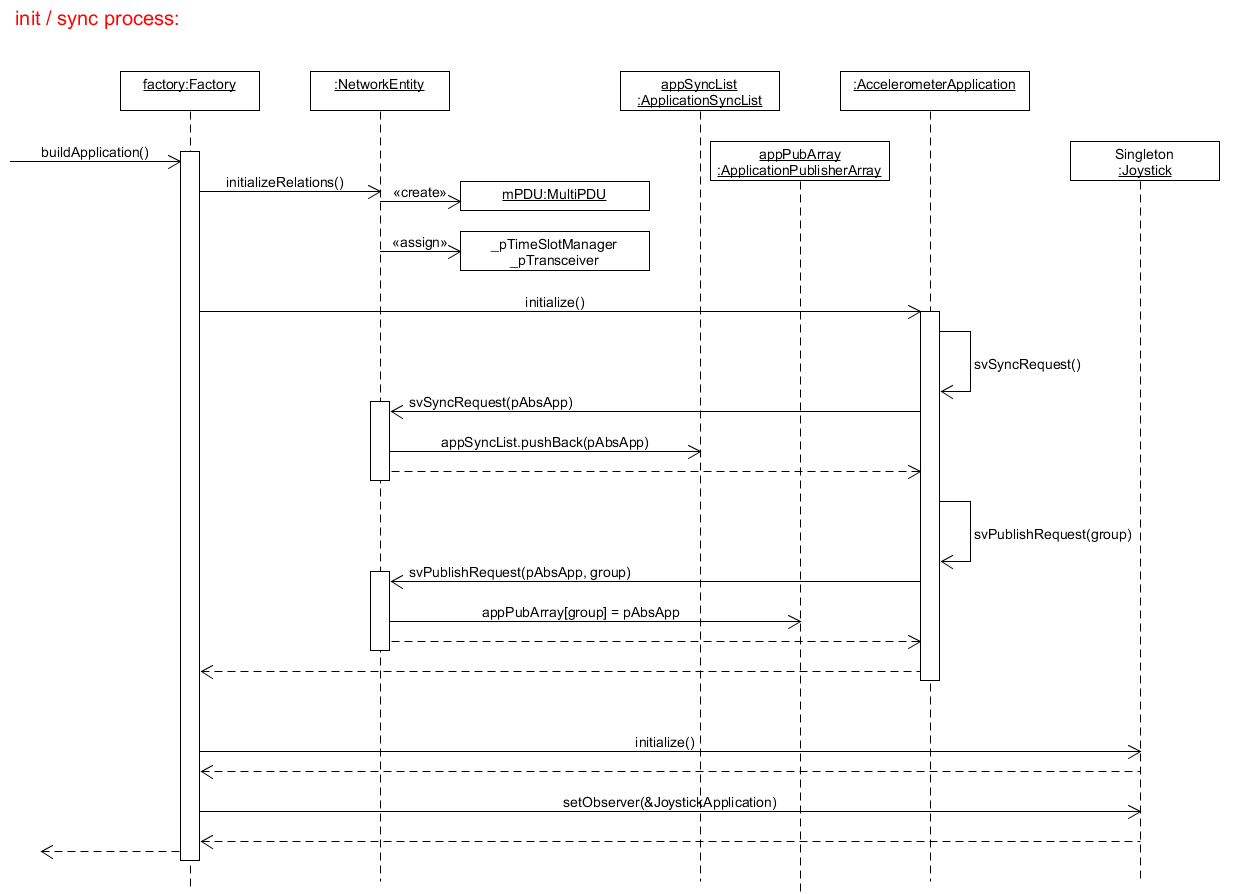
Description automatically generated

# Design

The following sub-chapters give a brief description on how the corresponding values are acquired and transmit via the DeseNET protocol.

## Initialization and synchronization

A general view of the initialization process is shown in the sequence diagram below. The corresponding diagram in PDF can be found in the delivery folder.



## Beacon reception

The principle of the main sequence is shown in the flowcharts below. Thereby, the chart on the left describes the algorithm when a frame is received by the method “onReceive” of the NetworkEntity. The chart on the right shows the process to send the MPDU frame by the means of the “onTimeSlotSignal” method, which is called at a timer event, launched in “onReceive”.

The corresponding detailed sequence diagram in PDF can be found in the delivery folder.

Diagram

Description automatically generated

# Tests

## Scenarios

To validate the implemented solution, the test scenarios in the table below were defined. Thereby, the demo sensor formed the reference for a correct behaviour and the general frame structure is like in   
§ 1.2.

|  |  |  |
| --- | --- | --- |
| **#** | **Test** | **Expected Result** |
| 1 | Send single Beacon for sampled accelerometer values  Objective:  Test correct data and correct reception time | **Header:**  LEN = 0x09  **ID + Cnt:**  sensID = 0x87 (Type = 1, ID = 7)  ePDUcnt = 0x01  **SV ePDU:**  ePDU Head = 0x16   (Type = 0, SVgroup = 2, Length = 6)  dataSV = 0xC406Ca00C703 (top left corner)  **EV ePDU:**  n/a  **Timing:**  Response time btw 400 [ms] and 450 [ms] |
| 2 | Send single Beacon for single event data  Objective:  Test correct data and correct reception time | **Header:**  LEN = 0x0D  **ID + Cnt:**  sensID = 0x87 (Type = 1, ID = 7)  ePDUcnt = 0x03  **SV ePDU:**  n/a  **EV ePDU:**  ePDU Head #1 = 0xC1  (Type = 1, EV ID = 8, Length = 1)  dataEV #1 = 0x20 (centre button pressed)  ePDU Head #2 = 0xC1  (Type = 1, EV ID = 8, Length = 1)  dataEV #2 = 0x01 (button released)  **Timing:**  Response time btw 400 [ms] and 450 [ms] |
|  |  |  |
| 3 | Send single Beacon for multiple event data without sampled values  Objective:  Test correct event data without over-charging the MPDU and that the SV group subscription works | **Header:**  LEN = 0x20 (max)  **ID + Cnt:**  sensID = 0x87 (Type = 1, ID = 7)  ePDUcnt = 0x0F  **SV ePDU:**  n/a (unsubscribed)  **EV ePDU:**  ePDU Head #odd = 0xC1  (Type = 1, EV ID = 8, Length = 1)  dataEV #1 = 0x20 (centre button pressed)  ePDU Head #even = 0xC1  (Type = 1, EV ID = 8, Length = 1)  dataEV #2 = 0x01 (button released)  **Timing:**  Response time btw 400 [ms] and 450 [ms] |
| 4 | Send beacons continuously for sampled accelerometer values  Objective:  Test correct SV data multiple times | Same as #1 (additionally check multiple slot durations)  Qualitative check by moving the sensor window as well. |
| 5 | Send beacons continuously for event data  Objective:  Check correct data EV data by pressing each button | **Header:**  LEN = 0x0C  **ID + Cnt:**  sensID = 0x87 (Type = 1, ID = 7)  ePDUcnt = 0x02  **SV ePDU:**  n/a  **EV ePDU:**  ePDU Head #1 = 0xC1  (Type = 1, EV ID = 8, Length = 1)  dataEV #1 = 0x20 (centre button pressed)  dataEV #2 = 0x04 (right button pressed)  dataEV #3 = 0x10 (down button pressed)  dataEV #4 = 0x02 (left button pressed)  dataEV #5 = 0x08 (up button pressed)  **Timing:**  Response time btw 400 [ms] and 450 [ms] |

## Results on Mesh simulator

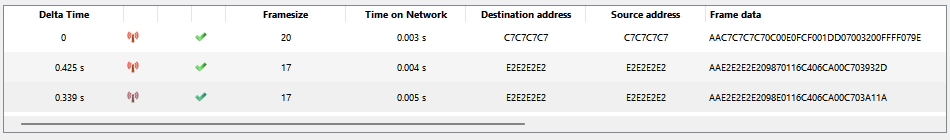
|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Test** | **Verdict** | **Remark** |
| 1 | Send single Beacon for sampled accelerometer values | OK | -- |
| 2 | Send single Beacon for single event data | OK | -- |
| 3 | Send single Beacon for multiple event data | OK | -- |
| 4 | Send beacons continuously for sampled accelerometer values | OK | -- |
| 5 | Send beacons continuously for event data | OK | -- |

As a qualitative verification, the following screenshot shows the comparison between the demo sensor (ID # 14) and the tested sensor (ID # 7). The detailed test data can be found in Appendix § 6.1.

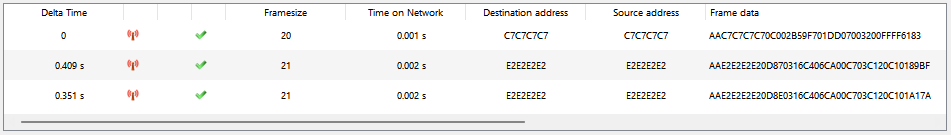
Text

Description automatically generated

**Accelerometer values comparison (top left corner)**



**Joystick Values comparison (pressed and released centre button)**



**General remark**

Sometimes, the sent responses of the tested sensor and the demo sensor were slightly out of range (around 10 ms too late). This can be explained by the fact that in simulation mode, the operation system of the used PC works sometimes too slow.

## Results on real target

Because the sent and received frames cannot be displayed with the real target, the correct functionality is verified with the displayed icons of the testbench.

**Accelerometer SV data**

*Nucleo board still on table → OK*

Graphical user interface, text, application, email

Description automatically generated

*Moving board in different positions → OK*

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

*Unsubscribe accelerometer SV group (#2) and moving board → OK*

Graphical user interface, text, application, email

Description automatically generated

⟹ Accelerometer values freeze due to unsubscription

**Joystick EV data**

*Single button check → OK*

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

*Multiple button presses→ OK*

Graphical user interface, text, application, email

Description automatically generated

*Checking time limit → OK*

*Graphical user interface, text, application, email

Description automatically generated*

⟹ Even by spamming the button, the time slot was respected for most of the time

**General Remark**

On the real system, the error message “No MPDU as a response to the beacon from 7” shows up sometimes. This error is due to the air as a transmission medium, which is perturbed by other devices.

The MPDU frame was never over-charged, hence it showed always coherent data with a delay of maximal 2 beacon cycles.

# Conclusion

The embedded software solution, proposed in this document, showed a solid functioning. It fulfilled the requirements of the test scenarios in the simulated environment, as well as on the real target.

Nevertheless, some small issues could have been observed during the tests under real conditions. This was due to the perturbations in the air. However, the Nucleo board never lagged, froze or got stuck. This shows that a satisfying solution has been developed.

# Appendix

## Frame observation of Mesh simulator test series

### Test 1: SV accelerometer data with single beacon

Graphical user interface, application, table, Excel

Description automatically generated

Graphical user interface, application, Excel

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface

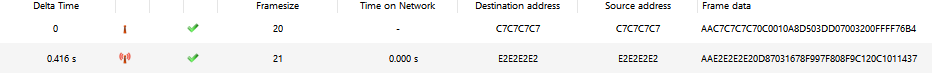
Description automatically generated

### Test 2: Single EV joystick data with single beacon

*Press + Release centre button*

Graphical user interface, text, application, email

Description automatically generated



### Test 3: Multiple EV joystick data with single beacon (SV group 2 disabled)

MAX\_EVELM\_NBR = 30 // maximal event number to avoid spam

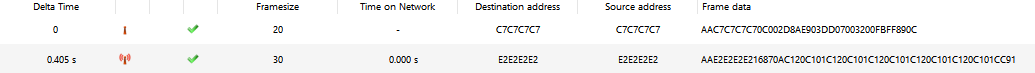
CUTOFF\_EVELM\_NBR = 4 // number of elements to be cut after having reached MAX\_EVELM\_NBR

*Press the centre button 5 times*

⟹ The captured data is sufficient to take place into one single buffer

Graphical user interface, text, application, email

Description automatically generated



*Press the centre button 15 times*

⟹ The captured data is sufficient to take place into exactly two buffers.

Because of press and release: 2 \* 15 Events = 30 Events = MAX\_EVELM\_NBR

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface

Description automatically generated

*Press the centre button 16 times*

⟹ The captured data is sufficient to take place into exactly two buffers and the “spam delimiter” resizes the event list down to 26 elements (30 Events – CUTOFF\_EVELM\_NBR)

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface

Description automatically generated

### Test 4: SV accelerometer data with multiple beacons

*Time Slot Duration: 40 ms*

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated with low confidence

*Time Slot Duration: 30 ms*

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

### Test 5: EV joystick data with multiple beacons

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface

Description automatically generated

1. <https://moodle.msengineering.ch/pluginfile.php/228006/mod_resource/content/1/Desem2Definition.pdf> (accessed 05.01.2023) [↑](#footnote-ref-2)