User manual evaluation kit for NLM0011/NLM0010

About this document

Scope and purpose

This document describes Infineon's evaluation kit for the latest Near Field Communication (NFC) controller NLM0011/NLM0010 for LED power supply current configuration, and how to use it.

The kit provides a quick demonstration of all features of NLM0011, evaluates filter designs and starts the development process for application software.

Intended audience

This document is intended for design and software engineers who want to improve their LED power supply current configuration application.

Table of contents

Table of contents

1 Introduction

The evaluation kits EVAL_NLM0011_DC and EVAL_NLM0011_DC_RE can be seen as a tool for design and software engineers who want to use NLM0011/NLM0010. It can be used with the "NLM-PWM" mobile app or with a Feig reader.

It provides:

- Fast demonstration of all features of NLM0011, including:
  - Parameter setting (PWM frequency, operating hours, on/off counting)
  - Programming CLO table (also memory layout for making values faster without calculation)
  - Duty cycle settings and indication via LED ramp
- Evaluation of own PWM filter
- A start for the development process for application software, e.g. a Feig reader, without using a full system

2 Main features

The kit consists of:

- NFC board 3 to 5 V (DK_NLM_ 01)
- NFC board 5 to 25 V (DK_NLM_02)
- Demo board (DK_NLM_03)
- USB stick (including mobile app for smartphone and user manual)
- USB cable (A to micro B)

Only included in option with Feig reader:

- Feig desktop reader ID CPR30-USB
- USB cable (A to mini B)

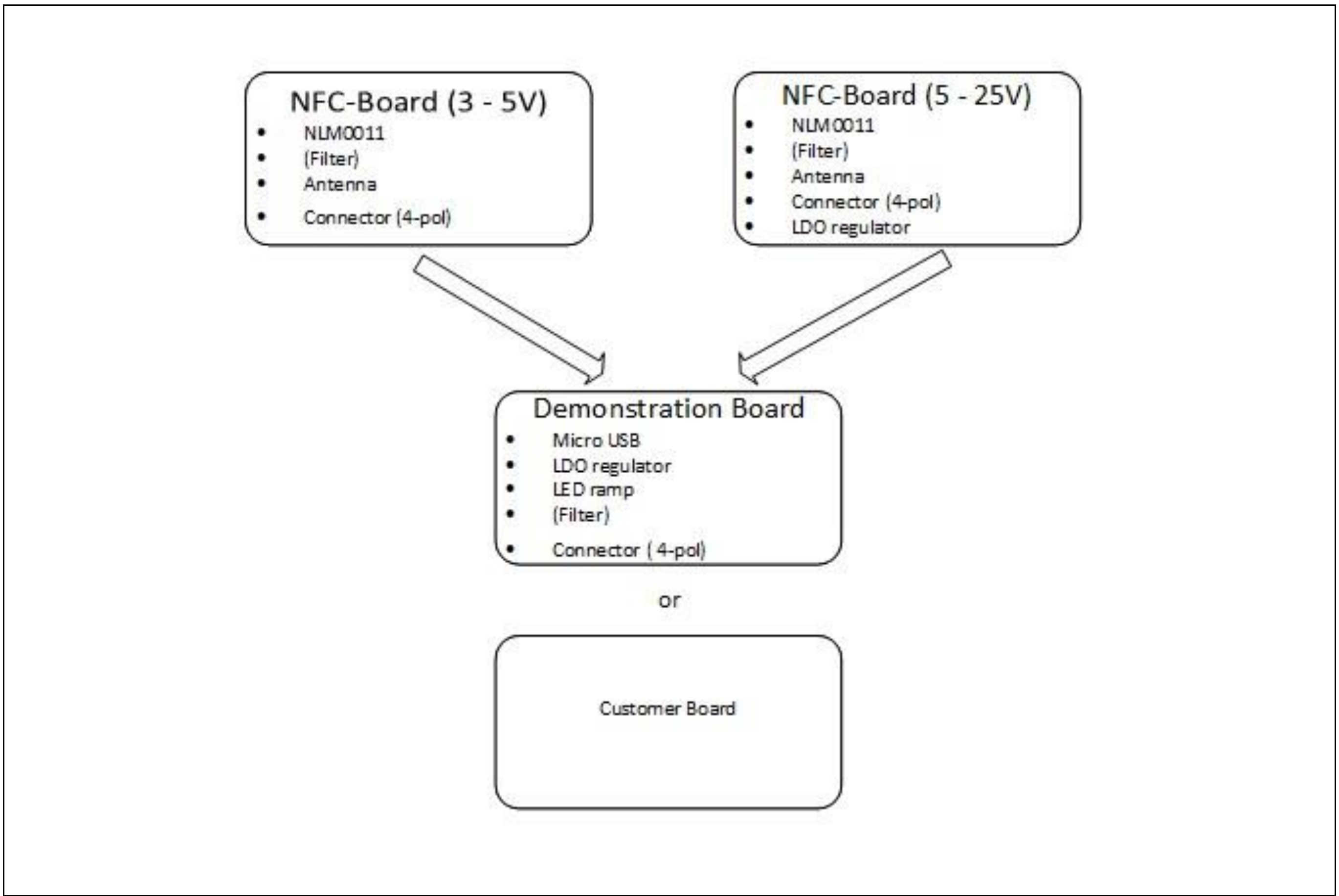

Figure 1 Structure of the evaluation kit

When the mobile app is installed on an Android smartphone, or when using the Feig reader, the parameters can be set by programming the NFC boards. According to the set parameters on NFC boards, the LED ramp on the demo board displays the duty cycle (percentage of duty cycle$_{Imax}$). It is also possible to connect the NFC boards to their own applications. For this, the corresponding NFC board must be chosen. For applications with V_{CC} = 3 to 5 V, use DK_NLM_02; for applications with V_{CC} = 5 to 25 V, use DK_NLM_01. When using the demo board, both NFC boards can be connected. So it is also possible to start the development process for Feig application software only with this evaluation kit, without using a full system.

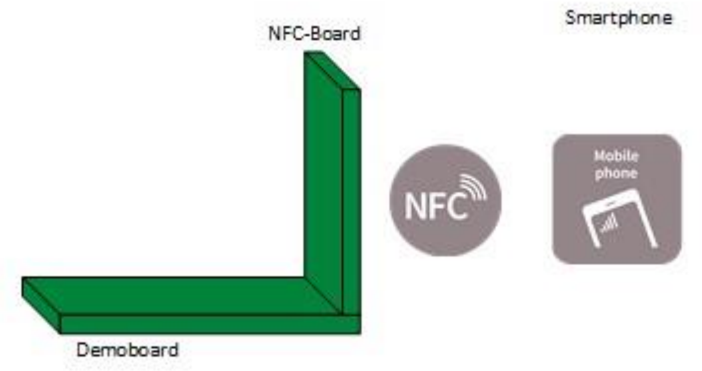

Figure 2 Set the parameters on NLM0011 via NFC and watch the LED ramp according to the set parameters (with preset filter on the demo board)

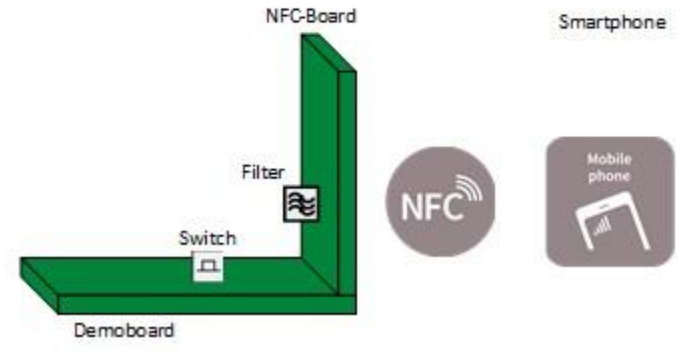

Figure 3 Set the parameters on NLM0011 and watch the LED ramp according to the set parameters (with own filter on the NFC board, filter on demo board disabled by the switch)

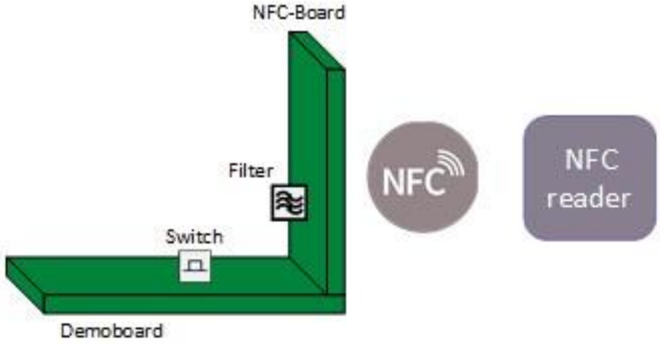

Figure 4 **Start developing process for application software for the Feig reader**

3 Mobile app

In order to use the mobile app, it must be installed on the smartphone and the NFC board must be in passive mode. That means it is not connected to V_{CC}.

To install the mobile app:

- Connect the smartphone to a PC
- Move “NLM-PWM.apk” from the USB stick to the phone

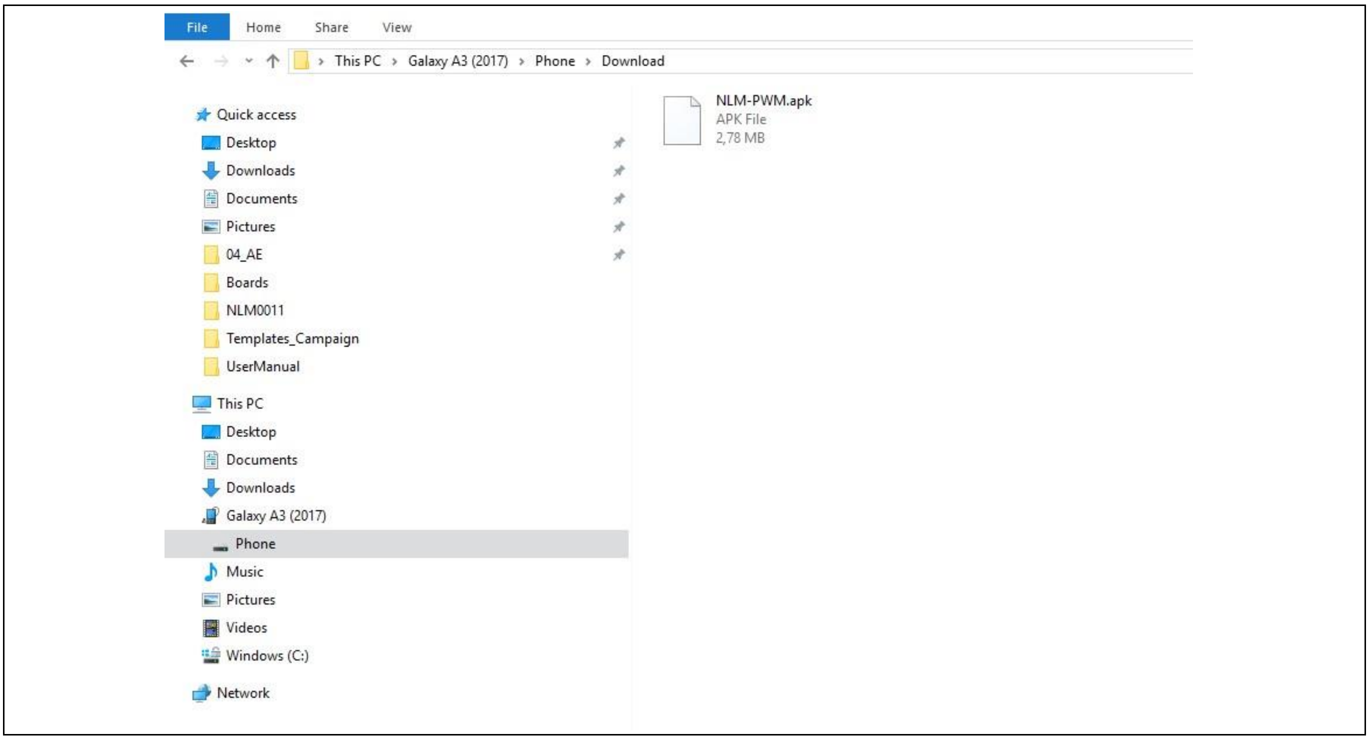

Figure 5 Moving “NLM-PWM.apk” to phone

- Open file on phone
- Press “install” on the phone

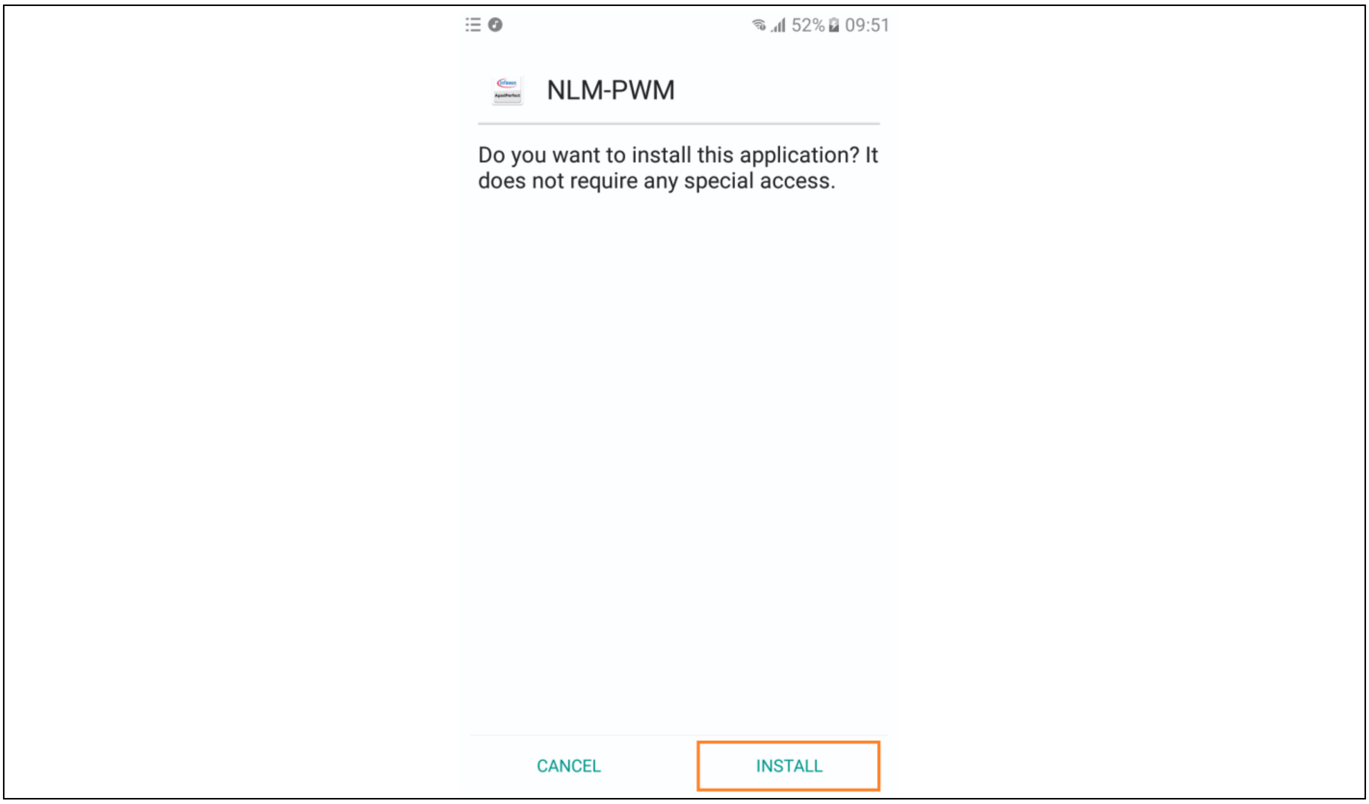

Figure 6 Confirm installation of mobile app

- Go to "Setting > Connections"
- Activate "NFC and Payment"

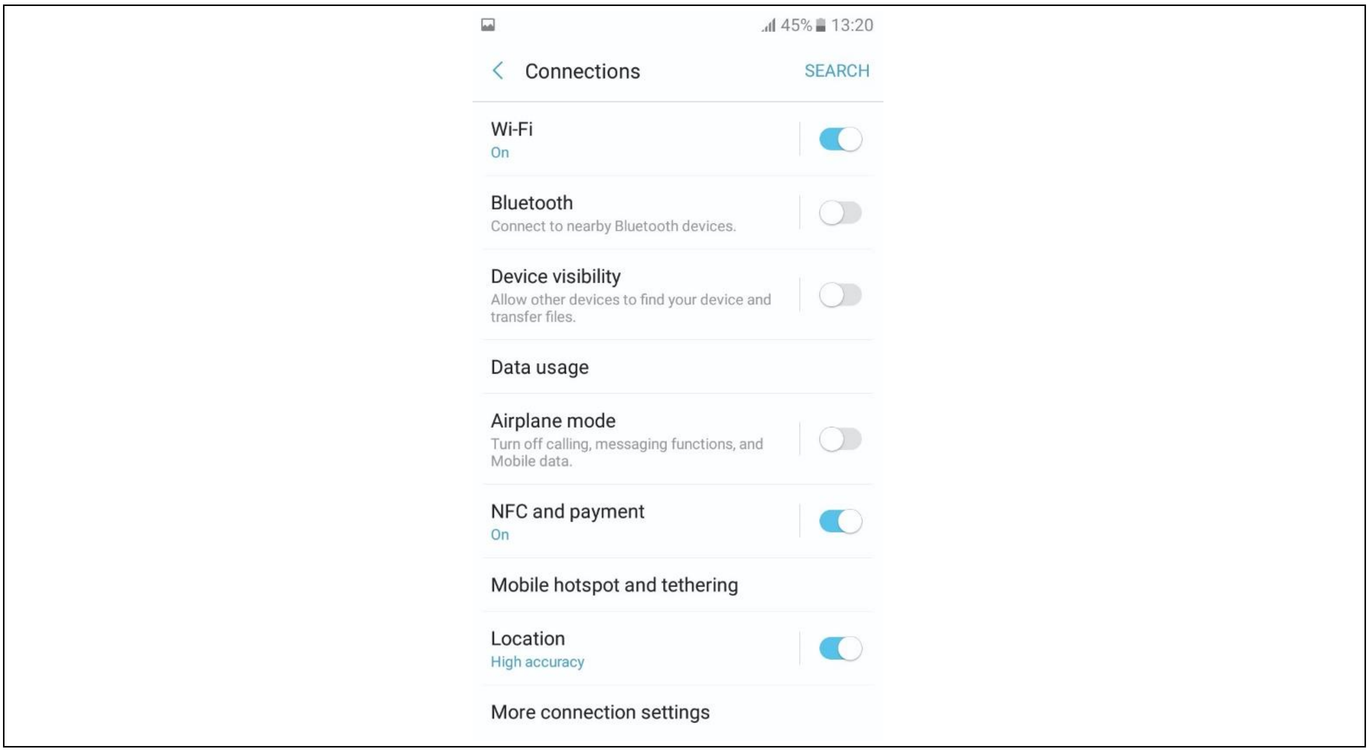

Figure 7 Activating NFC

- Start mobile app

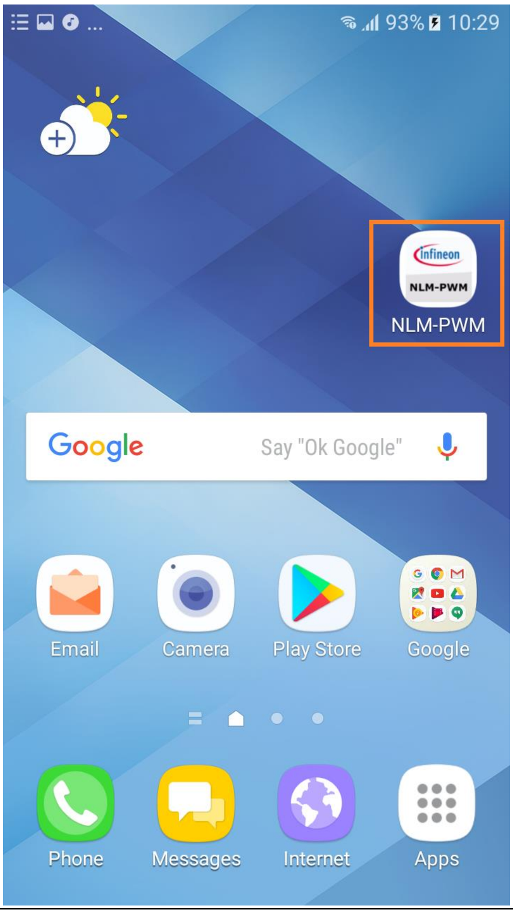

Figure 8 Start mobile app

Now the mobile app is ready for use.

3.1 Write parameters

When starting the mobile app for the first time, it looks like this:

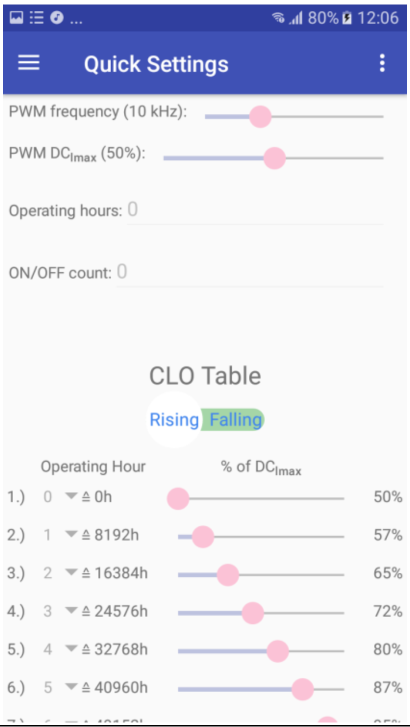

Figure 9 Preset parameters

Now the app is in “read mode”. To change to “write mode” tap in the top right-hand corner and deactivate “read only”.

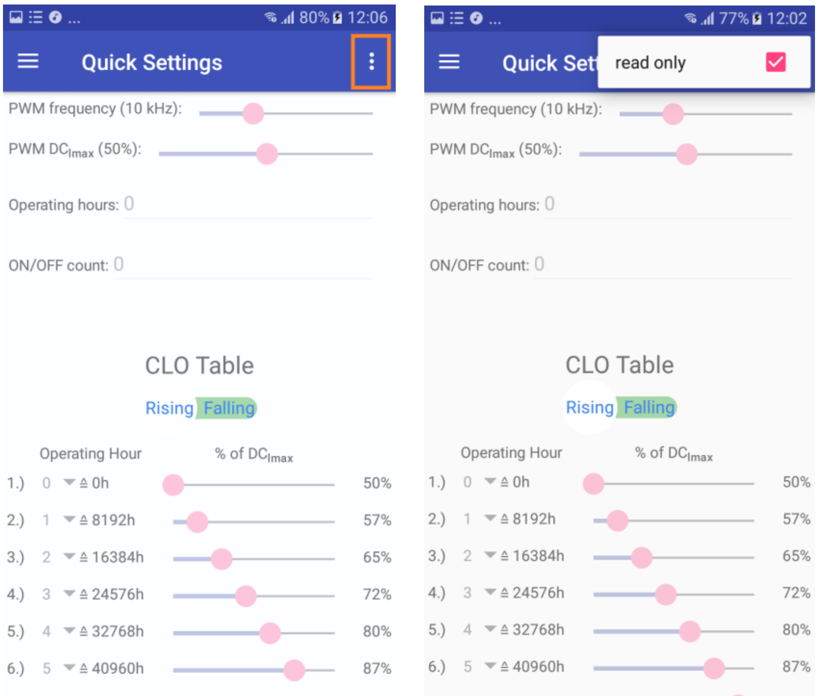

Figure 10 Change mode

Now the parameters can be set for:

Mobile app

- PWM frequency
- PWM DC_{Imax}
- Operating hours
- On/off count
- CLO table (8 points for operating hours and percentage of DC_{Imax})
  - The value for operating hours has to be set as multiples of 8192 in a range from 0 to 15
  - The value for percentage of DC_{Imax} has to be set in a range from 50 to 100 percent for rising curve and from 0 to 50 percent for falling curve
  - The slider must be set according to the chosen curve (falling or rising)

All parameters, exept for access codes, can be updated with this app.

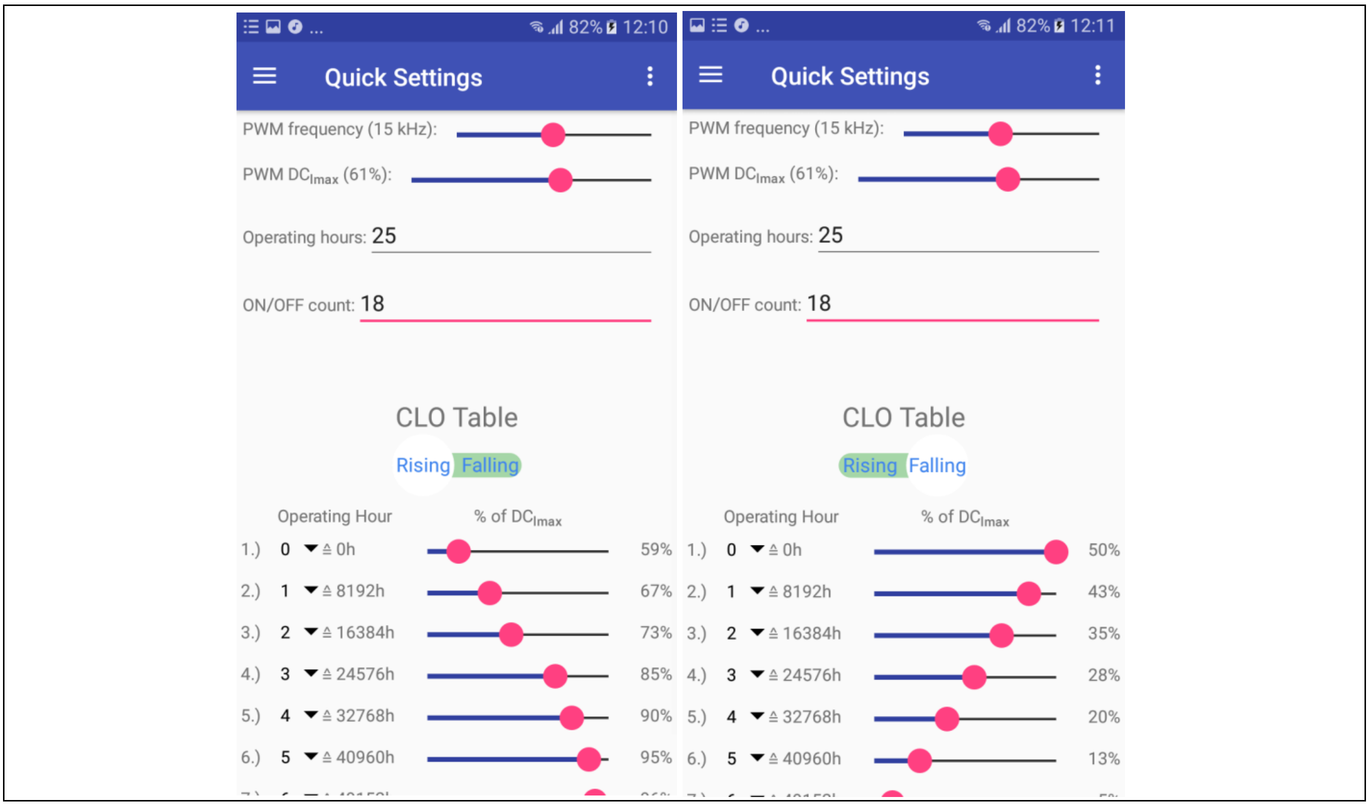

Figure 11 Set parameters

When all parameters are set in the app:

- Turn on the phone's speaker.
- Position the phone close to the external NFC antenna.
- Move slowly up–down and left–right to test the antenna position inside the phone.
- When two antennas make contact, a short beep will be emitted.

Mobile app

- Watch the bottom line of the mobile app, where an indication of connection failure or success will pop up.
- If it is successful, the parameters entered in the mobile app are stored in EEPROM of NLM0011.
- If it fails, move the phone away and close to the antenna again to establish a new NFC connection.

The antenna position varies from phone to phone.

3.2 Read parameters

To read the EEPROM, change the mode again to “read mode” by tapping the right-hand corner of the display and activating “read only”.

Position the phone close to the external antenna again and wait until the NFC is successfully completed. The EEPROM data of NLM0011 is now read out successfully.

If it fails, move the phone away again and then close to the antenna to establish a new NFC connection.

The EEPROM data is visible in the “Quick Settings” window. The EEPROM structure with data can be shown in “Memory Layout”.

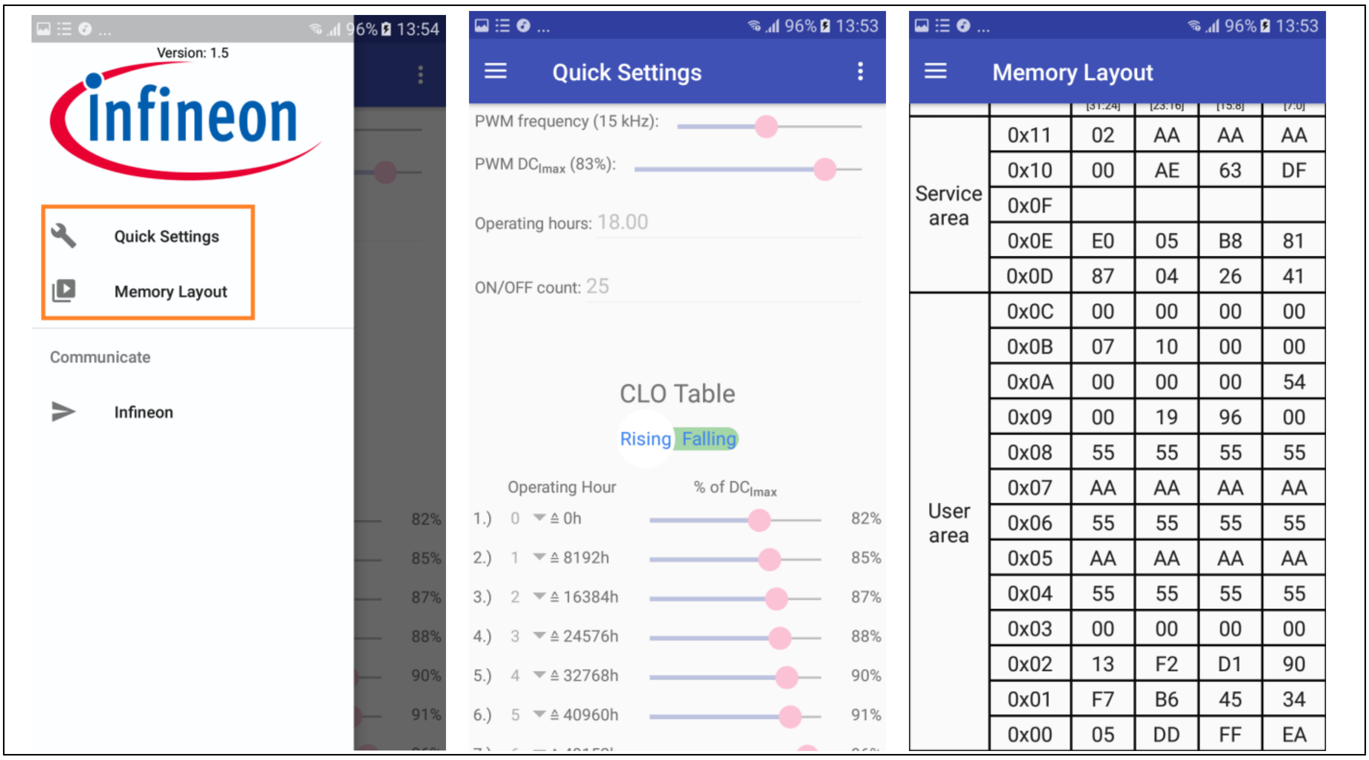

Figure 12 Main menu, quick settings, memory layout

In this way the mobile app can also be used for getting the hexadecimal code for the values in CLO.

Attention: ***Because of internal roundings, it is not possible to set exactly 100 percent for PWM DC_{Imax} <u>in the mobile app</u>. The highest value that can be set is 99 percent.***

4 Operational conditions and parameters

4.1 NFC board 3 to 5 V (DK_NLM_02)

Table 1 Operational conditions DK_NLM_02

Parameter	Symbol	Limit values			Unit	Note
		Min.	Typ.	Max.		
Board supply	V_{CC}	3		5	V	Power supply
PWM output	PWM	2.7	2.8	2.9	V	Unfiltered PWM output
PWM frequency	f_{PWM}	1		30	kHz	Frequency of PWM output
Ground	GND	0		0	V	System GND
Ambient temperature	T	-40		+85	°C	Ambient temperature in active and passive mode

4.2 NFC board 5 – 25 V (DK_NLM_01)

Table 2 Operational conditions DK_NLM_01

Parameter	Symbol	Limit values			Unit	Note
		Min.	Typ.	Max.		
Board supply	V_{CC}	5		25	V	Power supply
PWM output	PWM	2.7	2.8	2.9	V	Unfiltered PWM output
PWM frequency	f_{PWM}	1		30	kHz	Frequency of PWM output
Ground	GND	0		0	V	System GND
Ambient temperature	T	-40		+85	°C	Ambient temperature in active and passive mode

4.3 Demo board (DK_NLM_03)

Table 3 Operational conditions DK_NLM_03

Parameter	Symbol	Limit values			Unit	Note
		Min.	Typ.	Max.		
Board supply	V_{CC}	4.5	5	5.5	V	Power supply
Ground	GND	0		0	V	System GND
Ambient temperature	T	-20		+65	°C	Ambient temperature in active and passive mode

5 Evaluation boards

To use the kit, install the mobile app on an Android smartphone (chapter 3) or use the Feig reader.

5.1 NFC boards

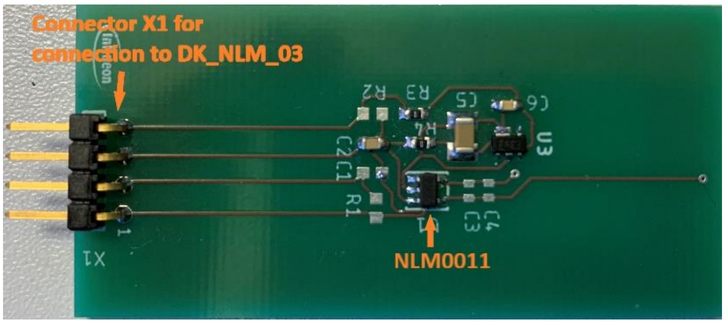

Figure 13 Bottom view of DK_NLM_01

Figure 14 Top view of DK_NLM_01 with NFC antenna

Both NFC boards include the NLM0011 chip and the antenna for NFC. Also the connector for connection to the demo board or own applications; the pin-out is shown below.

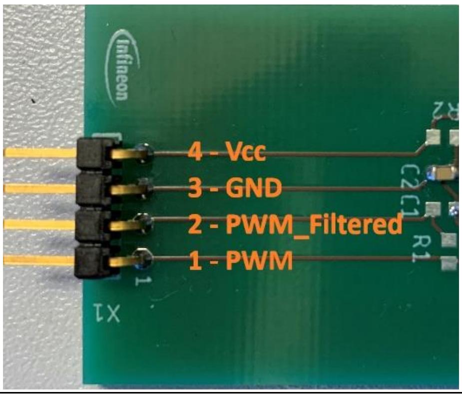

Figure 15 Pin-out of X1 NFC boards

The PWM_Filtered signal can only be measured on pin 2 of the NFC board, when an own filter (R1 and C1) is soldered onto the board, because the preset filter is mounted on the demo board.

Additionally, on the DK_NLM_01 board the components for the Low Drop-Out (LDO) regulator are mounted, to enable V_{CC} voltage from 5 to 25 V.

The EEPROM of the NFC boards can be read and written by mobile app as shown in chapter 3, or by Feig reader.

The read/write process is only possible in passive mode. That means the board is not connected to V_{CC}. This can be done by disconnecting the NFC board from the demo board or own applications, or by switching off the demo board.

5.2 Demo board (DK_NLM_03)

Features:

- Connector X1 for connection to NFC boards
- Dip-switch S1 for enabling own filter/preset filter of PWM signal
- On/off switch S2
- Micro-USB connector X2 for powering the demo board
- 8 LEDs for duty cycle indication (0 to 80 percent)

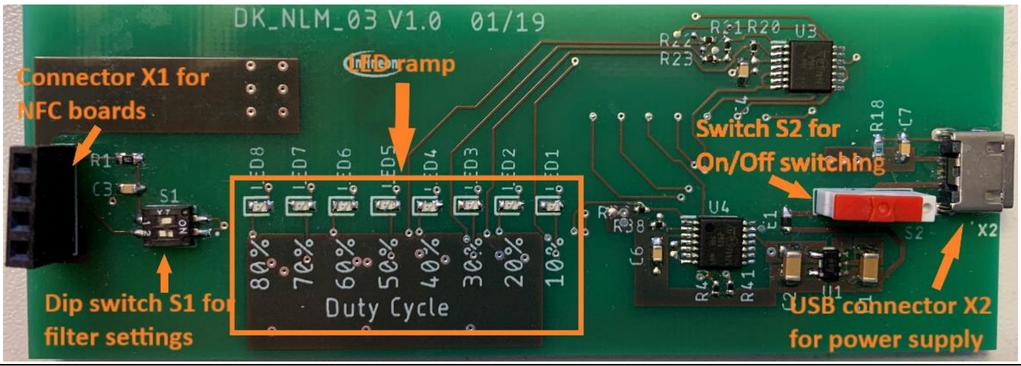

Figure 16 Top view DK_NLM_03

To indicate the set duty cycle, write all parameters to EEPROM on the NFC board and connect it to the demo board. Also connect the demo board via USB cable to a standard USB 5 V power supply (e.g. a laptop). To use the preset filter (R1, C3) on the demo board, the dip-switch must be set as shown below:

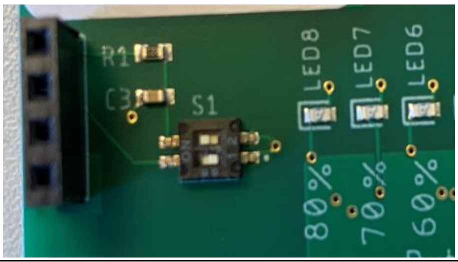

Figure 17 Dip-switch S1 setting for preset filter

The preset filter is dimensioned for PWM frequencies higher than 12.9 kHz (R1 = 22 kHz, C3 = 56 nF). In case an own filter is enabled, set position 1 of S1 to off and position 2 to on. Now the preset filter is disabled and an own filter can be soldered in on R1 and C1 on the NFC board.

Switch on the demo board by sliding switch S2 to the on position, as shown below:

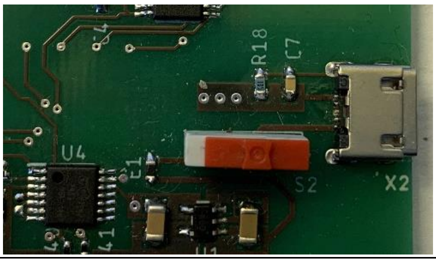

Figure 18 Switch S2 on position

The LEDs show the set duty cycle, a percentage of DC_{Imax}.

<u>Example:</u>

App settings, written to EEPROM of NLM0011 in passive mode:

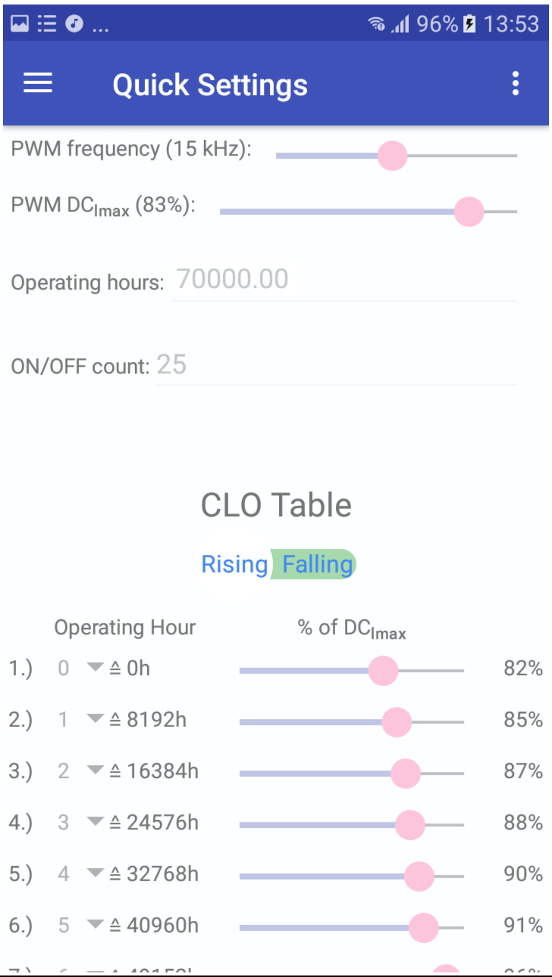

Figure 19 App settings

Because of the set parameters (operating hours = 70000 -> set point 1 in CLO), the percentage of DC_{Imax} can be calculated as shown:

$$\% \; of \; DC_{Imax} = \frac{PWM \; DC_{Imax}}{100\%} \cdot \% \; of \; DC_{Imax}$$

$$\% \; of \; DC_{Imax} = \frac{83\%}{100\%} \cdot 82\% = 68{,}06\%$$

That means that on the demo board, seven LEDs should be lit.

In this case, the demo board LED indication looks like this:

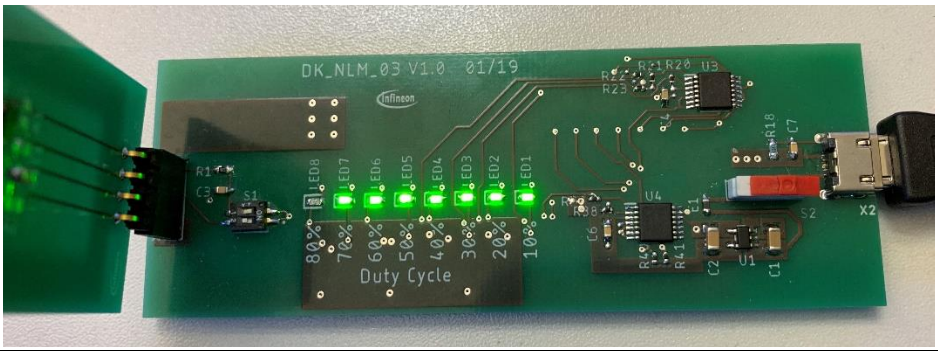

Figure 20 LED indication for our example

Attention: ***When connecting the NFC board to the demo board, make sure that pin 1 of the NFC board is connected to pin 1 of the demo board (pin 1 is marked on each board).***

6 Schematic

6.1 NFC board DK_NLM_01

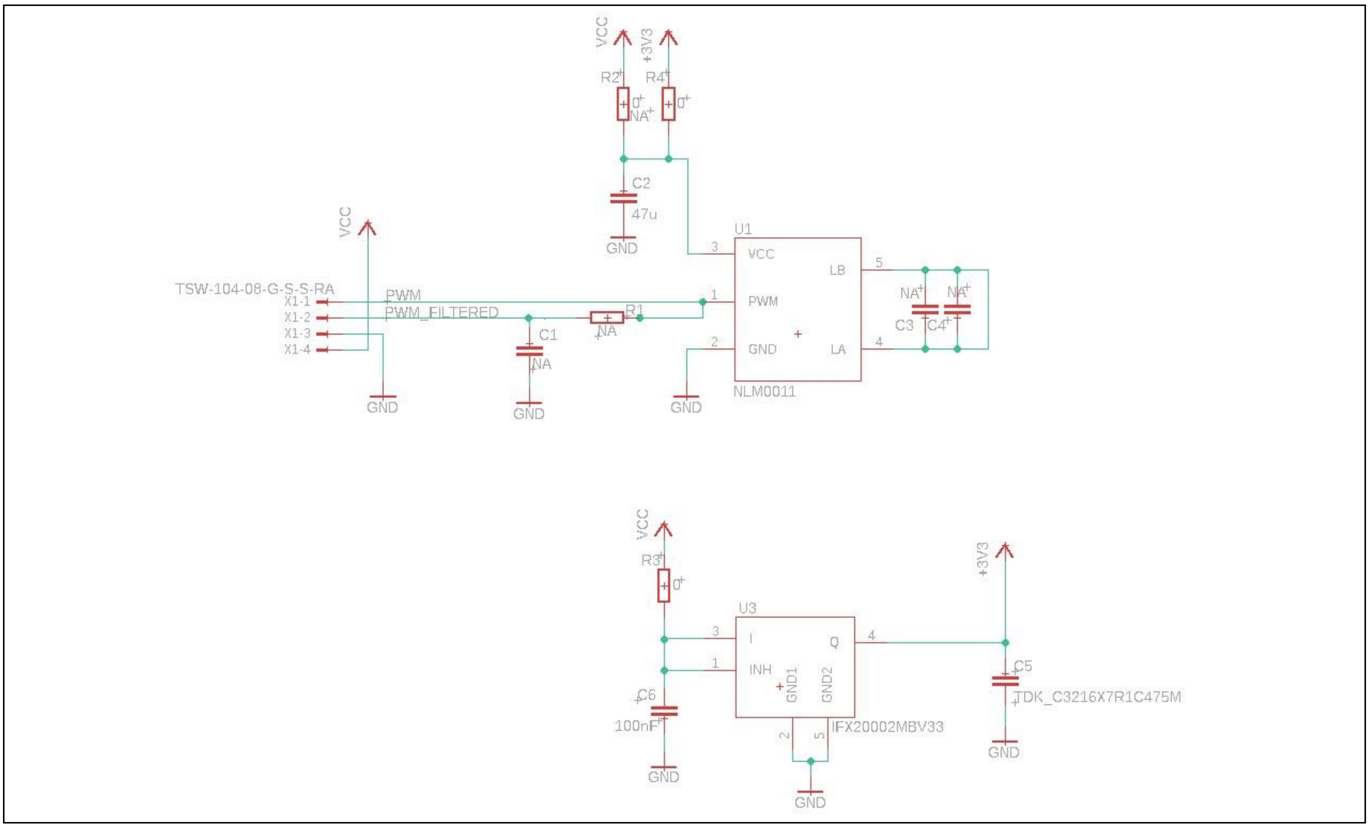

Figure 21 Schematic of NFC board DK_NLM_01

6.2 NFC board DK_NLM_02

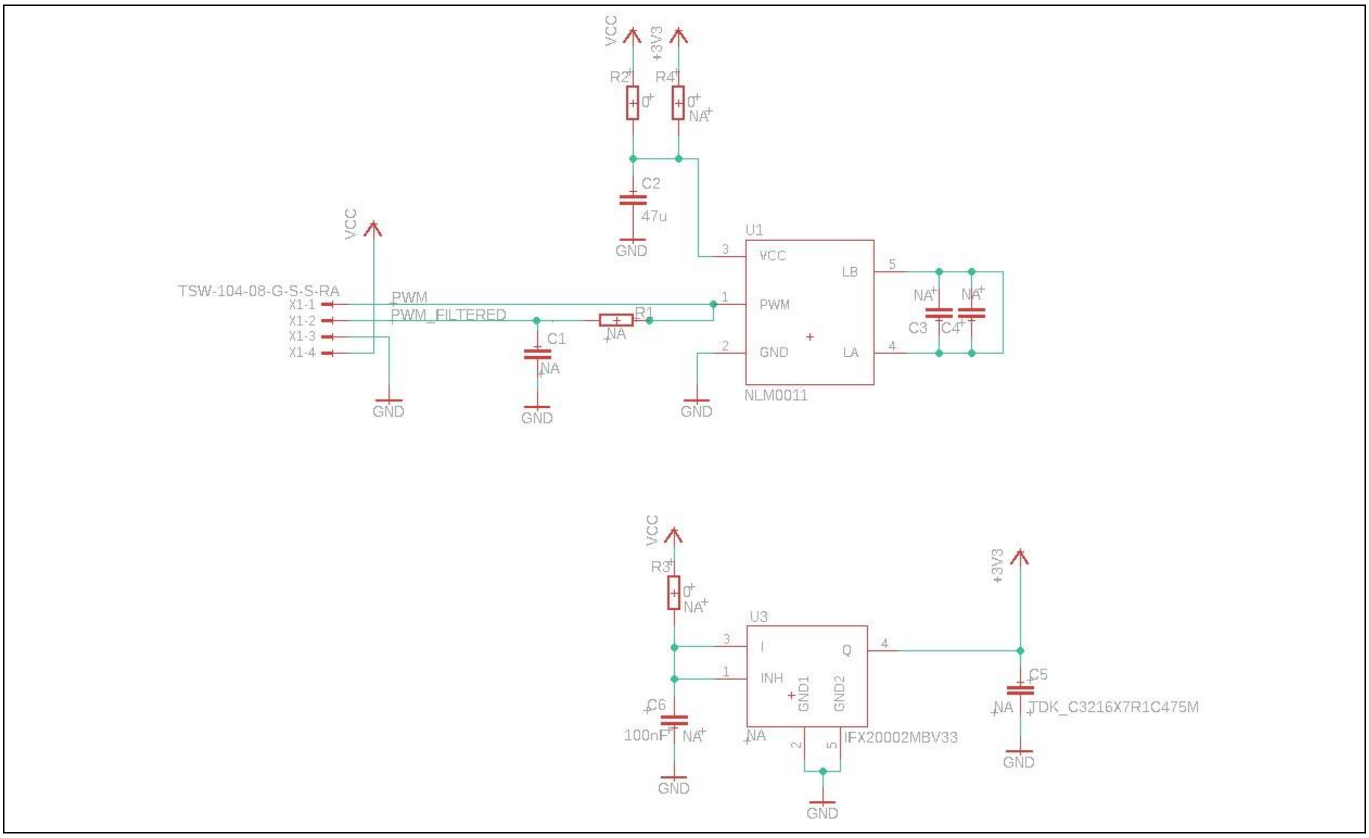

Figure 22 Schematic of NFC board DK_NLM_02

6.3 Demo board DK_NLM_03

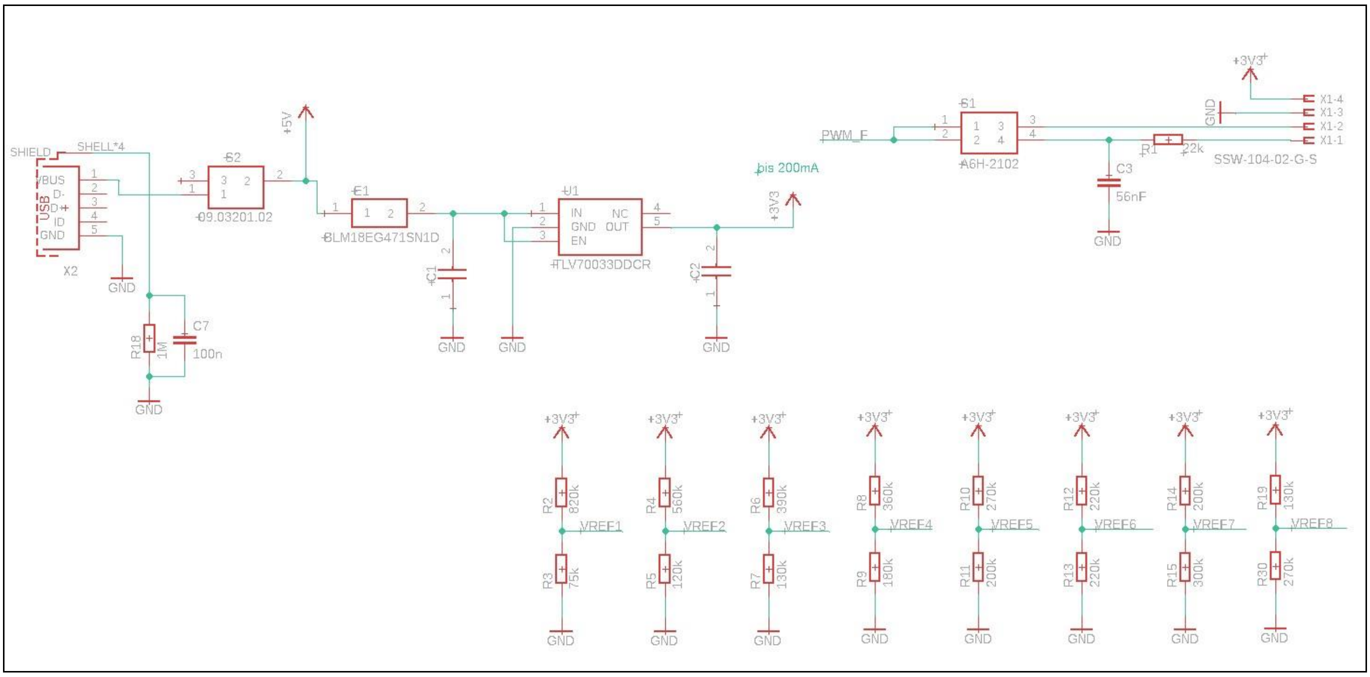

Figure 23 Schematic of demo board DK_NLM_03

Schematic

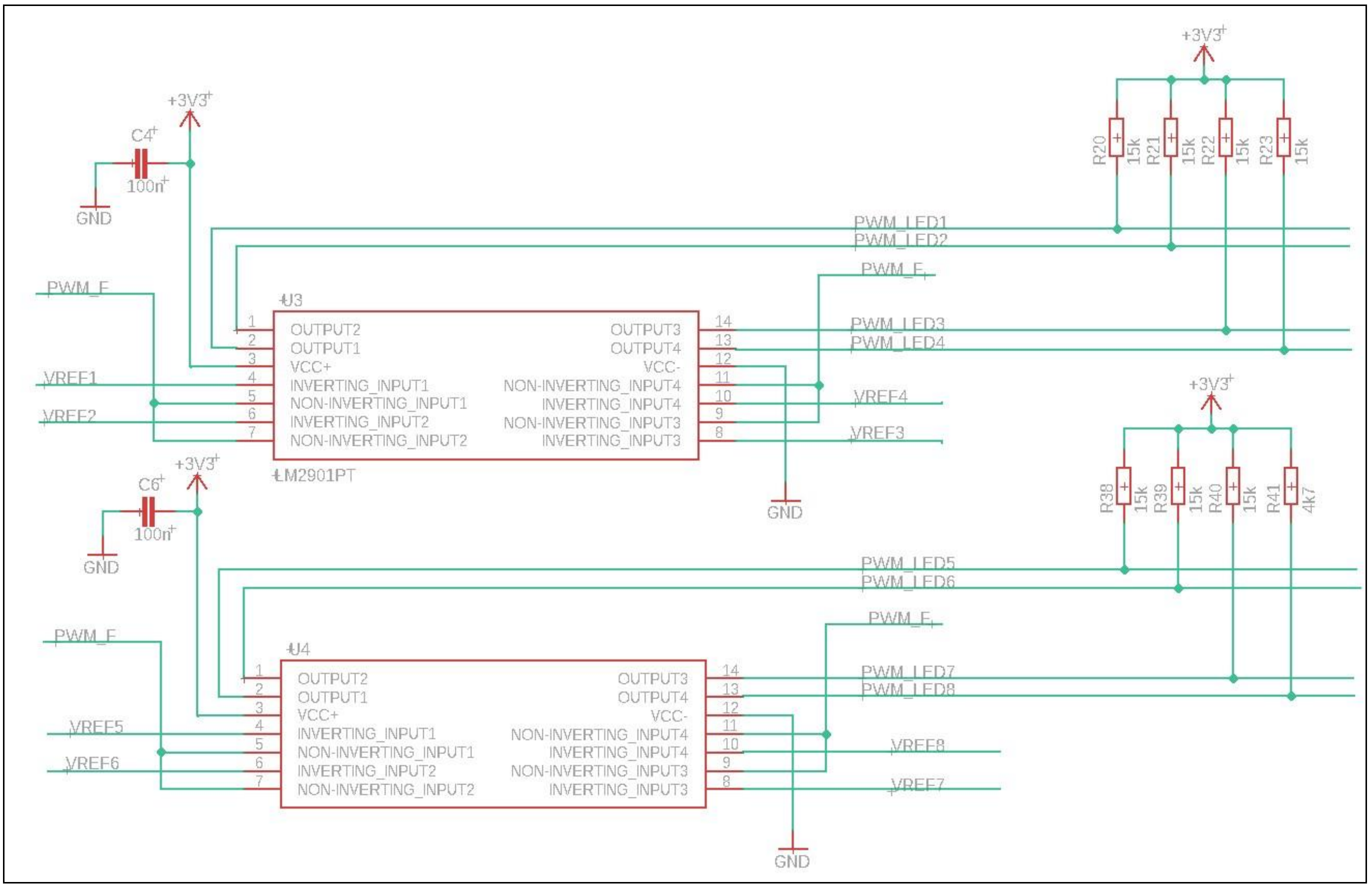

Figure 24 **Schematic of demo board DK_NLM_03**

Schematic

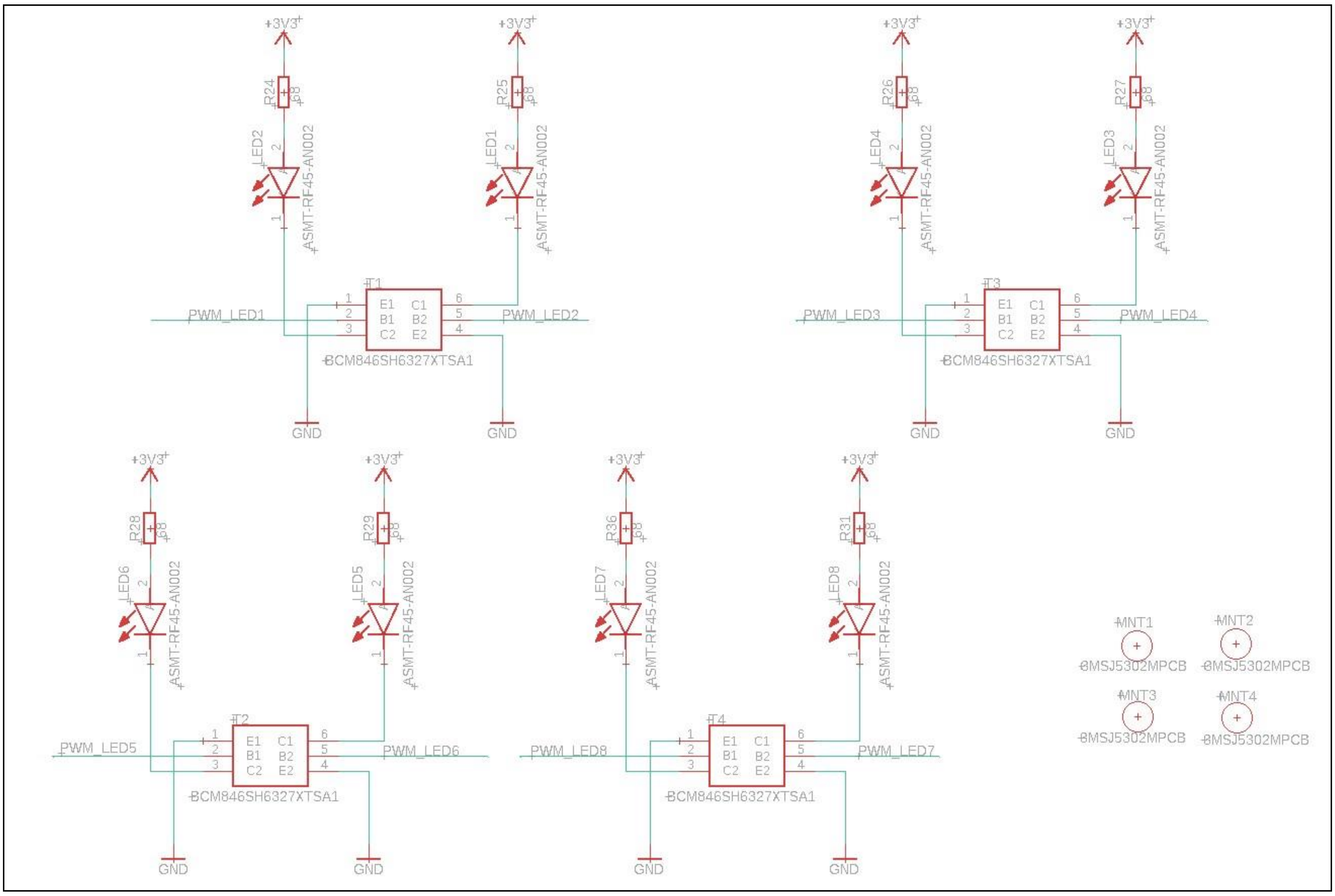

Figure 25 Schematic of demo board DK_NLM_03

7 Layout

7.1 NFC boards DK_NLM_01 and DK_NLM_02

Figure 26 Top layer of DK_NLM_01

Figure 27 Top layer of DK_NLM_02

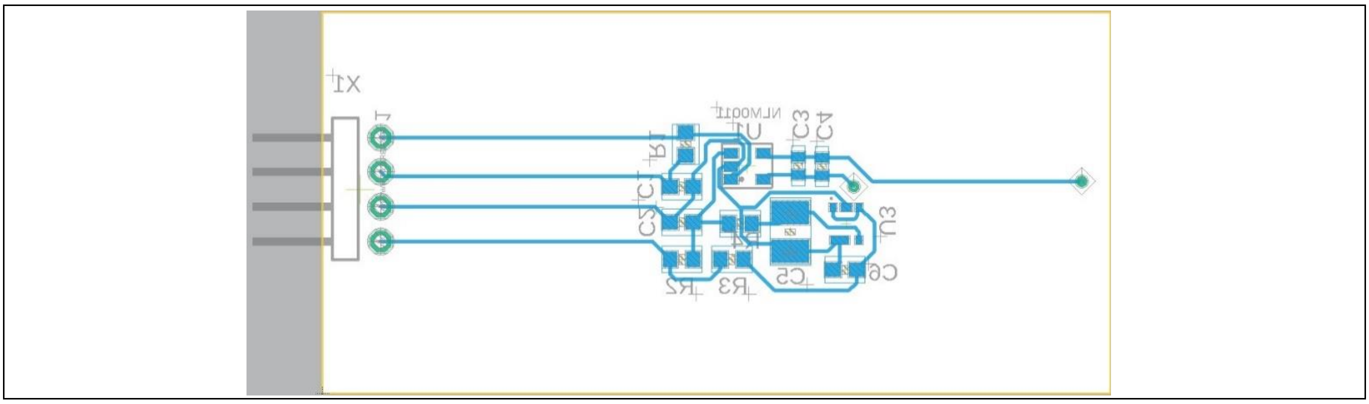

Figure 28 Bottom layer of DK_NLM_01 and DK_NLM_02

7.2 Demo board DK_NLM_03

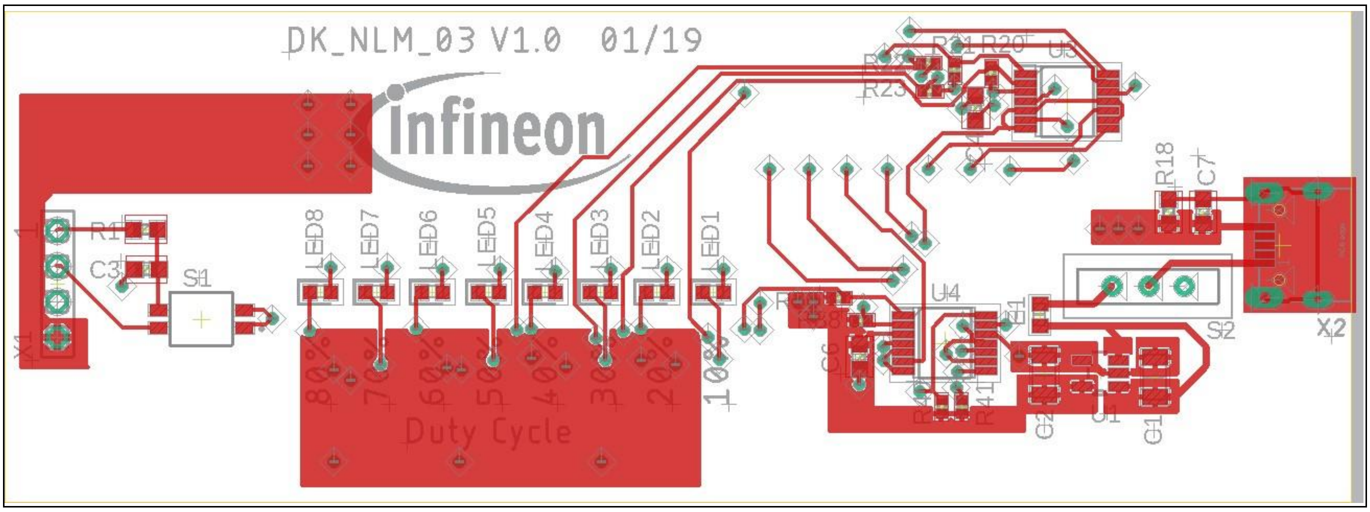

Figure 29 Top layer of DK_NLM_03

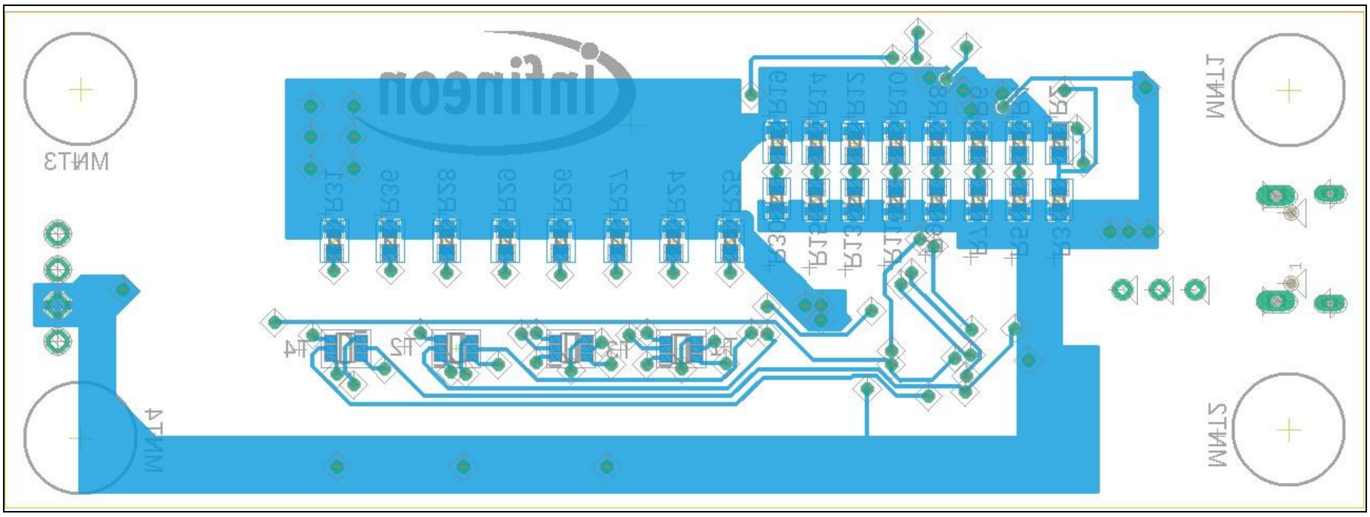

Figure 30 Bottom layer of DK_NLM_03

8 Bill of Materials (BOM)

8.1 NFC board DK_NLM_01

Table 4 BOM of DK_NLM_01

Item number	Quantity	Part reference	Value	Package	Description
1	2	R3, R4	0	0603	Resistor
2	1	C6	100 nF	0603	Capacitor
3	1	C2	47 µF	0603	Capacitor
4	1	U3	IFX20002MBV33	SCT595-5	LDO
5	1	U1	NLM0011	SOT23-5	NLM0011 Infineon
6	1	C5	47 pF	1206	TDK_C3216X7R1C475M
7	1	X1	4		TSW-104-08-G-S-S-RA Samtec

8.2 NFC board DK_NLM_02

Table 5 BOM of DK_NLM_02

Item number	Quantity	Part reference	Value	Package	Description
1	1	R2	0	0603	Resistor
2	1	C2	47 µF	0603	Capacitor
3	1	U1	NLM0011	SOT23-5	NLM0011 Infineon
4	1	X1	4		TSW-104-08-G-S-S-RA Samtec

8.3 Demo board DK_NLM_03

Table 6

Item number	Quantity	Part reference	Value	Package	Description
1	1	X2			629105150521 Würth, horizontal Micro-USB
2	1	S2			09.03201.02 EOZ, two-way through-hole SIP switch
3	3	C4, C6, C7	100 nF	0603	Capacitor
4	1	C3	56 nF	0603	Capacitor

Item number	Quantity	Part reference	Value	Package	Description
5	1	R5	120 kΩ 1%	0603	Resistor
6	2	R7, R19	130 kΩ 1%	0603	Resistor
7	7	R20, R21, R22, R23, R38, R39. R40	15 kΩ 1 percent	0402	Resistor
8	1	R9	180 kΩ 1%	0603	Resistor
9	1	R18	1 MΩ 1%	0603	Resistor
10	2	R11, R14	200 kΩ 1%	0603	Resistor
11	2	R12, R13	220 kΩ 1%	0603	Resistor
12	2	R10, R30	270 kΩ 1%	0603	Resistor
13	1	R15	300 kΩ 1%	0603	Resistor
14	1	R8	360 kΩ 1%	0603	Resistor
15	1	R6	390 kΩ 1%	0603	Resistor
16	4	MNT1, MNT2, MNT3, MNT4			3MSJ5302MPCB 3M, Bumpon
17	1	R1	22 kΩ 1%	0603	Resistor
18	1	R41	4.7 kΩ 1%	0402	Resistor
19	1	R4	560 kΩ 1%	0603	Resistor
20	8	R24, R25, R26, R27, R28, R29, R31, R36	68 Ω 1%	0603	Resistor
21	1	R3	75 kΩ 1%	0603	Resistor
22	1	R2	820 kΩ 1%	0603	Resistor
23	1	S1			A6H-2102 Omron, two-way surface-mount dip-switch
24	8	LED1, LED2, LED3, LED4, LED5, LED6, LED7, LED8		0603	ASMT-RF45-AN002 Avago, chip LED green
25	4	T1, T2, T3, T4		SOT65	BCM846SH6327XTSA1 Infineon, bipolar transistor
26	1	E1		0603	BLM18EG471SN1D Murata, ferrite 470 R 500 mA
27	2	C1, C2	1 µF 16 V	1206	C1206X105K4RACAUTO Kemet, MLCC
28	2	U3, U4		SOP65	LM2901PT ST, quad comparator
29	1	X1			SSW-104-02-G-S Samtec, through-hole connector
30	1	U1		SOT95	TLV70033DDCR TI, LDO regulator

9 Feig reader application software

To program NLM0011/NLM0010, a NFC reader device is required. It could be a mobile phone with built-in NFC interface or a professional NFC reader device. For industrial applications, like LED lighting, typically a professional NFC reader device is used for reasons of ease of use, long sensing distance and highly reliable operation. The professional NFC reader device is connected to a PC and is controlled by PC software (NFC reader application software). The NFC reader application software is developed either by the vendor of the LED power supply, or the luminaire maker. Typically, vendors of the LED power supply provide their application software to OEM customers.

NFC readers from Feig GmbH are certified by the Module-Driver Interface Special Interest Group (MD-SIG). Using the Feig desktop reader ID CPR30-USB, we developed an application demo based on Feig application development SDK.

9.1 Command types and their execution method

NLM0011/NLM0010 support following nine NFC commands: Inventory, Stay Quiet, Read Single Block, Write Single Block, Select, Reset to Ready, Write AFI, Lock AFI, and Write Byte.

Using Feig SDK, a NFC command can be executed by using either the SendProtocol() function or the SendTransparent() function. The three commands below must be executed with the SendTransparent() function:

Read Single Block, Write Single Block, Write Byte.

Execution of other commands – Stay Quiet, Select, Reset to Ready, Write AFI, Lock AFI – are recommend to use the Send Protocol() function.

9.2 Code examples

Below two examples using transparent function and protocol function are developed for the Feig NFC reader. For complete source codes, please check the source code file provided. If you are using NFC readers from other vendors, please refer to the corresponding SDK .

9.2.1 Inventory command

In this example, the sendProtocol() function is used to execute “inventory” command. The benefit of using the sendProtocol() function is that a short command string can be used.

```
public int inventory(FedmIscTagHandler_Result result) throws
FePortDriverException, FeReaderDriverException, FedmException,
AgedPerfectException
{
        FeIscProtocol localFeIscProtocol = new FeIscProtocol();
        FedmIscReader      iscReader      = reader.getReaderImpl();
        FedmIscTagHandler iscTagHandler = tagHandler.getTagHandlerImpl();
        String cmd = new String();
        String uid = new String();
        uid         = iscTagHandler.getUid();

        result.data    = null;
```

```
        //build up NFC command string
         cmd = cmd + "01";                  // Inventory command
         cmd = cmd + "00";              // Mode: 00-nonaddressed only

//NFC command execution
         String str = iscReader.sendProtocol((byte)0xB0, cmd);

        //Exception handling
if (str.length() == 0) {
             throw new
FedmException(iscReader.getErrorText(Fedm.ERROR_NO_DATA),
Fedm.ERROR_NO_DATA);
         }


if(reader.getLastStatus() == (byte)0x95) {
             // we have a VICC error
             throw new
AgedPerfectException(AgedPerfectErrorCode.getErrorText(FeHexConvert.hexString
ToInteger(str)),
                         FeHexConvert.hexStringToInteger(str));
         } else if(iscReader.getLastStatus() == 0) {
             // we have a feedback from the Reader
             result.data        = new byte[8];
             result.data[0]     = (byte)localFeIscProtocol.recData[1];
             result.data[1]     = (byte)localFeIscProtocol.recData[2];
             result.data[2]     = (byte)localFeIscProtocol.recData[3];
             result.data[3]     = (byte)localFeIscProtocol.recData[4];
             result.data[4]     = (byte)localFeIscProtocol.recData[5];
             result.data[5]     = (byte)localFeIscProtocol.recData[6];
             result.data[6]     = (byte)localFeIscProtocol.recData[7];
             result.data[7]     = (byte)localFeIscProtocol.recData[8];
             String TrTypeStr = str.substring(2,4);
             String DsfidStr  = str.substring(4,6);
             String uidStr    = str.substring(6);
             jProtocol.append("TR-TYPE: " + TrTypeStr + "; DSFID: " + DsfidStr
+ "; UID: " + uidStr);
         }
         return reader.getLastStatus();
     }
```

9.2.2 Read single block command

In this example, the sendTransparent() function is used to execute the “read single block” command. The benefit of using the sendTransparent() function is that each bit in the command string can be defined. With good software architecture design, it can enable NFC vendor-independent or less-dependent application software development – the command string is identical and independent to the vendor of the NFC reader used.

```
public int readSingleBlock (byte address, byte db_n, FedmIscTagHandler_Result
result)
            throws FePortDriverException, FeReaderDriverException,
FedmException, AgedPerfectException {
        FeIscProtocol localFeIscProtocol = new FeIscProtocol();
        FedmIscReader iscReader          = reader.getReaderImpl();
        FedmIscTagHandler iscTagHandler  = tagHandler.getTagHandlerImpl();
        String cmd         = new String();
        String uid;
        String uidRev = new String();
        uid = iscTagHandler.getUid();
        for (int i=0; i < 15; i=i+2) {
            uidRev = uidRev + uid.substring(14-i, 16-i);
        }
        result.data    = null;


       //build up NFC command string
        cmd = cmd + "02";              // STX
        if (iscTagHandler.nonAddressedMode) {
            cmd = cmd + "000F";      // ALENGTH w/o UID plus CRC: 15
        }
        else {
            cmd = cmd + "0017";      // ALENGTH w UID and CRC: 23
        }
        cmd = cmd + "FF";              // COM-ADR
        cmd = cmd + "BF";              // CMD (Reader)
        cmd = cmd + "01";              // Mode 1: read request
        cmd = cmd + "0038";            // RSP-length in bits w/o SOF and EOF
        cmd = cmd + "0000";            // RSP-delay in multiples of 590ns
0x0000=0x021F=320,4us
        if (iscTagHandler.nonAddressedMode) {
            cmd = cmd + "12";        // flags: selected, high data-rate,
single sub-carrier
        } else {
```

```
            cmd = cmd + "22";        // flags: addressed, high data-rate,
single sub-carrier
        }
        cmd = cmd + "20";            // ReadSingleBlock command
        if (!iscTagHandler.nonAddressedMode) {
            cmd = cmd + uidRev;      // UID reverse order
        }
        cmd = cmd + FeHexConvert.byteToHexString(address);      // Address

       //NFC command execution
String str = iscReader.sendTransparent(cmd, true);

//Exception handling
        localFeIscProtocol.recProtocol =
FeHexConvert.hexStringToByteArray(str);
        int i = iscReader.splitRecProtocol(localFeIscProtocol);
        if (i == (int)0x95) {
            int ISOErrorCode = localFeIscProtocol.recData[0];
            throw new
AgedPerfectException(AgedPerfectErrorCode.getErrorText(ISOErrorCode),
ISOErrorCode);
        }
        else if (i == 0) {
            result.data         = new byte[4];
            result.data[0]      = (byte)localFeIscProtocol.recData[4];
            result.data[1]      = (byte)localFeIscProtocol.recData[3];
            result.data[2]      = (byte)localFeIscProtocol.recData[2];
            result.data[3]      = (byte)localFeIscProtocol.recData[1];
            // for debug only:
            String DataStr      = new String();
            DataStr = DataStr +
FeHexConvert.byteToHexString((byte)localFeIscProtocol.recData[4]);
            DataStr = DataStr +
FeHexConvert.byteToHexString((byte)localFeIscProtocol.recData[3]);
            DataStr = DataStr +
FeHexConvert.byteToHexString((byte)localFeIscProtocol.recData[2]);
            DataStr = DataStr +
FeHexConvert.byteToHexString((byte)localFeIscProtocol.recData[1]);
            // for debug:
            //jProtocol.append("Read-Data: " + DataStr + "\n");
        }
        else {
```

```
            throw new
FedmException(iscReader.getErrorText(Fedm.ERROR_NO_DATA),
Fedm.ERROR_NO_DATA);
        }
        return reader.getLastStatus();
    }
```

9.3 Feig reader software and documents

USB drivers, documents, firmware and demo programs can be downloaded from the Feig download area at https://www.feig.de/en/login/. Login data can be found in the installation manual, which is delivered together with the Feig reader.

10 General information

10.1 Restrictions

This evaluation kit offers limited features that enable you to evaluate and test Infineon products. The evaluation kit is not an end product (or finished appliance), nor is it intended or authorized by Infineon to be integrated into end products. You are not authorized to use the evaluation kit in any production system.

10.2 Additional information

For further information, refer to: http://www.infineon.com/

Revision history

Document version	Date of release	Description of changes
1.0	06.03.19	Initial version of document
1.1	19.07.19	Changed all pictures of mobile app. Added falling curve.
1.2	20.08.19	Minor changes in wording
1.3	28.08.19	Update Figure 18
1.4	16.12.19	Update chapter 3

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2019-12-16

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference
UM_1902_PL39_1904_160819