AN13996

How to use the low-power features of the PN76 family NFC controller

Rev. 5.0 — 18 November 2025 Application note

Document information

Information	Content
Keywords	PN76, PN7642, LPCD, ULP, ULPCD, ultra low-power card detection, standby, low-power
Abstract	This application note explains and shows how to use different low-power modes on the PN76 family NFC controllers using the NFC low-power modes example of the SDK.

1 Introduction

The PN76 family has many options to save power. This document highlights some of the most used functions and features. The goal is that the user understands the purpose of the different lower power modes and knows how to configure and run them, in the "nfc_low_power_modes" example. As this application note is tightly coupled to the SDK example "***nfc_low_power_modes"***, slight changes in different versions are to be expected, but the fundamentals stay the same. The SDK version used for this application note is v02.15.004.

As not every function is discussed within this document, it is highly recommended to read the data sheet, user manual, and PN7642 NFC controller user API documentation.

The LPCD and ULPCD calibration, configuration, parameters, and technical background are explained in detail in the PN7642 design-in recommendations.

1.1 Environment

The following tools, hardware, and software have been used:

- PNEV7642A development board
  - Firmware version: v02.06
- MCUXpresso IDE
  - Version: v24.12.148
- MCUXpresso PN7642 SDK
  - Version: v02.15.004
- Debugger (MCU-Link, SEGGER J-Link, ...)
  - Tool: SEGGER J-Link Base
- SEGGER J-Link RTT Viewer
  - SEGGER SDK v8.30

This document does not explain how to install, configure, or set up the PNEV7642 development board and its environment. For getting started with the PNEV7642 development board, read the PN76 family evaluation board quick start guide ref.[6]. The PN7642 is updated to FW v02.06 by using SDK v02.15.004.

1.2 Debugging

When the chip enters ultra low-power modes, the debugger loses connection, in most cases without any immediate warning.

The SDK offers the option to use UART as debug output instead of Semihost (console within the IDE). For the low-power examples, it is recommended to use UART and the J-Link Section 6 "RTT Viewer". The RTT viewer is easy to reconnect after the connection has been lost due to going to low power and subsequent waking up.

2 Low-power modes

Lower-power modes described within this Application note:

- Low-power card detection (LPCD) : Software driven measurement of I/Q channels.
- Standby : Interface states and AO_RAM is retained.
- Ultra low-power card detection (ULPCD) : Highest current saving card detection mode.
- Ultra low-power standby : Highest current saving standby mode.

2.1 Low-power card detection (LPCD)

The low-power card detection (LPCD) of the PN7642 is a single API which issues an I/Q channel measurement. The LPCD is software-implemented and runs while the microcontroller is active.

In contrast to NFC frontend readers, for example PN5190, this API will **not** set the PN7642 into standby, but directly return the measurement result:

Table 1. Results

Result	Description
PN76_STATUS_NO_TAG_DETECTED	If Tag is not detected during LPCD..
PN76_STATUS_TAG_DETECTED.	On Tag detection.
PN76_STATUS_EXTERNAL_FIELD.	In case of external RF detection during LPCD. Refer to PN76_Status_t for error codes details.

Calibration and measurement execution are described in the ref.[1].

2.1.1 LPCD API

All APIs and their description can be found in ref.[3].

◆ PN76_Sys_PCRM_LPCD_Start()

PN76_Status_t PN76_Sys_PCRM_LPCD_Start (**PN76_PCRM_LPCD_Mode_t** tLpcdMode)

This function Runs the SW LPCD.

Parameters

tLpcdMode Reference mode or measurement mode

Return values

PN76_STATUS_NO_TAG_DETECTED	If Tag is not detected during LPCD.
PN76_STATUS_TAG_DETECTED.	On Tag detection
PN76_STATUS_EXTERNAL_FIELD.	In case of external RF detected during LPCD. Please refer to **PN76_Status_t** for error codes details

Figure 1. LPCD Start API

2.2 Standby

In standby, all interface states and AO_RAM (Always On RAM) content is retained. The PMU operates in a low-power state and a wake-up counter clock is available. Multiple wake-up sources can be selected. See the API description for more information.

2.2.1 Standby API

All APIs and their description can be found in ref.[3].

◆ PN76_Sys_PCRM_EnterStandby()

```
PN76_Status_t PN76_Sys_PCRM_EnterStandby ( PN76_PCRM_LowPower_WakeUp_Config_t eWakeUpSource,
                                           uint32_t                           dwWakeUpCntValue
                                         )
```

Switch into standby mode.

Note

The function will never return if IC is switched into standby was successful.

In case the function returns, the standby prevention can be found in register **PCRM_SYS_BOOT2_STS (0xC5)**. This register can be read using **PN76_Sys_ReadRegister()**.

Once the IC enters into standby, the configuration and values of GPIO0s will persist.

Parameters

eWakeUpSource	Selection of wake-up source in Enum **PN76_PCRM_LowPower_WakeUp_Config_t**.
dwWakeUpCntValue	10 bit wake-up counter value: resolution=2,63ms -> max time slot=2,57s. This is used only in case of TIMER as wakeup.

Return values

PN76_STATUS_PARAMETER_ERROR	Incase no wakeup Source is selected
PN76_STATUS_LOW_POWER_ENTRY_PREVENTED.	Incase the Standby Entry is prevented. The reason for the prevention can be read from register **PCRM_SYS_BOOT2_STS (0xC5)**. This register can be read using **PN76_Sys_ReadRegister()**.

Figure 2. PN76_Sys_PCRM_EnterStandby

2.3 Ultra low-power card detection (ULPCD)

The ultra low-power card detection (ULPCD) offers the highest current saving. In ULPCD mode, the only wake-up sources to exit from the card detection loop are:

- Section 2.3.2.1 "GPIO 3 Abort"
- Section 2.3.2.2 "Card detected"
- Section 2.3.2.3 "External RF detected" if enabled.

Note: *The ULPCD* ***cannot*** *be used together with the DC-DC function. The PNEV7642 development board is designed to operate without DC-DC. The inductor for DC-DC can remain, and VUP is per default set (J1) to be supplied by the 5V board supply.*

For more information about the ULPCD capabilities and description, see the data sheet ref.[1], user manual ref.[2], and PN7642 design-in recommendations ref.[4].

The usage of ULPCD is explained in Section 5.4 "Option 5: ULPCD Calibration".

2.3.1 ULPCD API

All APIs and their description can be found in ref.[3].

The ULPCD API belongs to the Power Control and Reset Management (PCRM) system services:

◆ PN76_Sys_PCRM_EnterULPCD()

```
PN76_Status_t PN76_Sys_PCRM_EnterULPCD ( PN76_PCRM_ULPCD_Config_t bUlpcdConfig,
                                         uint16_t                 dwWakeUpCntValue,
                                         uint8_t                  bUlpdetEnable
                                       )
```

Switch into Ultra Low Power Card Calibration/Detection mode.

Parameters

bUlpcdConfig	Selection of ULPCD Configuration as in Enum **PN76_PCRM_ULPCD_Config_t**.
dwWakeUpCntValue	12 bit wake-up counter value: resolution=1ms -> max time slot=4.096s. This is interval defines the wakeup time during calibration phase and time between RF pings during card detection.
bUlpdetEnable	Ultra low power RF detection enabled/Disabled. Value to be TRUE means 1 for ULPDET Enable or FALSE means 0 for ULPDET Disabled. Any other value is not considered for ULPDET .

Return values

PN76_STATUS_SUCCESS	: SUCCESS is never returned as the system enters into Ultra low power mode in case of successful entry to ULP mode.
PN76_STATUS_PARAMETER_ERROR	Incase invalid wakeup source is selected or wake-up interval is more than 12bits

Figure 3. API: Enter ULPCD

2.3.2 Wake-up reasons in ULPCD

- Section 2.3.2.1 "GPIO 3 Abort"
- Section 2.3.2.2 "Card detected"
- Section 2.3.2.3 "External RF detected" if enabled.

2.3.2.1 GPIO 3 Abort

In Ultra-Low-Power-Card-Detection (ULPCD) the GPIO 3 can be used to abort it from an external source (host, power management unit, etc..). An ultralow frequency oscillator drives a wake-up counter, which triggers a periodic activation.

While the GPIO 3 can be raised at any time, the check if it has been raised is only done after the wake-up counter elapsed. This results in inconsistent latency from GPIO 3 edge to actual wake-up. The latency can vary from the *standby time maximum + 1 ms* to a theoretical minimum of 1 ms.

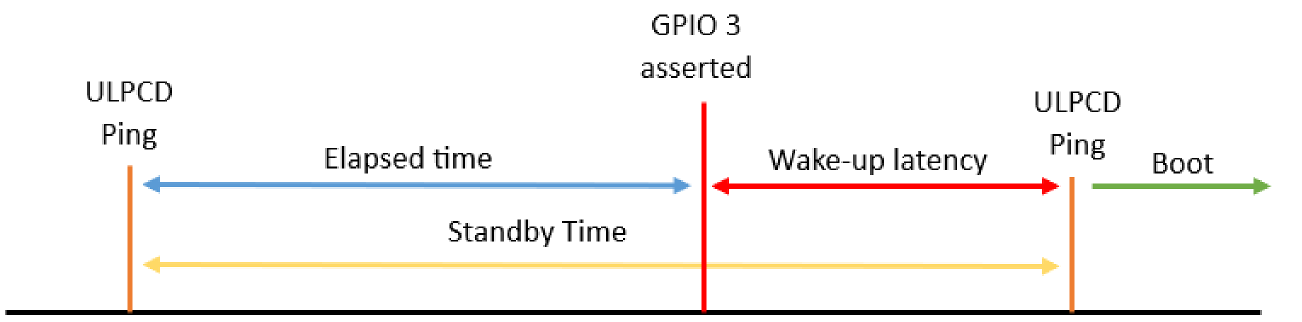


Figure 4. GPIO3 wake-up time variance

ULPCD Ping: A short RF pulse measuring the RSSI value.

Standby Time: The configured time between the wake-ups.

GPIO3: Depending on the configured polarity either a falling or rising edge on GPIO 3 of the PN7642.

Wake-up latency: Time from GPIO 3 trigger to actual PN7642 boot.

The wake-up time is: → **Wake-up latency** = Standby Time - Elapsed time

GPIO 3 is checked every time when the card detection is started. If GPIO 3 is still high, for example, from a previous abort and has not been deasserted, the ULPCD is aborted again. Best practice is to assert GPIO 3 for a short time, long enough to have a stable logical high, and deasserted afterward.

GPIO 3 abort cannot be disabled. If this option shall not be used, the pin shall be tied to ground.

2.3.2.2 Card detected

An object is detected to be in the proximity of the reader, when the measured RSSI differs from the reference RSSI by more than a configurable threshold.

In the measurement phase, the card detection activity is performed autonomously by the hardware at configurable time intervals (standby time). This configuration is passed as a parameter in the ULPCD API. The RSSI value is measured and compared against the reference value stored at the calibration phase.

Note: *For ULPCD, only RSSI is considered, not I/Q.*

2.3.2.3 External RF detected

Waking up from ultra low power card detection due to the detection of an external RF-Field.

When calling the ULPCD API "*PN76_Sys_PCRM_EnterULPCD(…)*" the third parameter "*bUlpdetEnable*" determines if an external RF-Field wakes up the PN7642 or not.

If the value is not 0, the RF level detector is enabled while in ULPCD. Any detection of an external RF-Field causes the immediate wake-up from ULPCD with the boot reason "BOOT_ULPCD_RX_ULPDET" (see register PCRM_SYS_BOOT1_STS 0xC6).

The low-power mode example prompts the following text: `Boot from ULPCD due to External RF detection`

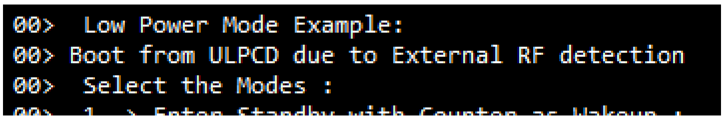


Figure 5. Boot due to external RF detection

2.4 Ultra low-power standby

In the ultra low-power (ULP) standby mode, everything is turned off except a timer based on an ultra low frequency oscillator (ULFO). In this mode, only two options are available for wake-up:

1. Wake-up by time-out (resolution: 1 ms, min time-out = 1 ms, max time-out 4096 ms)
   a. An external RF field is ignored and does not cause a wake-up.
2. External RF field detected or wake up by time-out.
   a. The wake-up timer is mandatory.

This mode is typically used if users want to save as much power as possible and waking up in intervals, or triggered by an external RF-Field, is sufficient.

The main difference of ULP standby to ULPCD (ultra low-power card detection) is that there is no card detection cycle and a wake-up timer is mandatory.

2.4.1 ULP Standby API

All APIs and their description can be found in ref.[3].

The ULP Standby API belongs to the PCRM (Power Control and Reset Management) System Services:

◆ PN76_Sys_PCRM_EnterUlpStandby()

```
PN76_Status_t PN76_Sys_PCRM_EnterUlpStandby ( PN76_PCRM_UltraLowPower_WakeUp_Config_t  wWakeupSource,
                                              uint16_t                                 dwWakeUpCntValue
                                            )
```

Switch into Ultra Low Power standby mode.

Parameters

wWakeupSource	Selection of wake-up source in Enum **PN76_PCRM_UltraLowPower_WakeUp_Config_t**.
dwWakeUpCntValue	12 bit wake-up counter value: resolution=1ms -> max time slot=4.096s. This is used as a wakeup time interval when wakeup source is TIMER.

Note

It is mandatory to have TIMER as wakeup source in all cases.
If external RF is configured as wakeup source, the IC wakes up if there is no external RF within the wakeup counter interval.
Higher priority for wakeup is always the Wakeup counter.

Return values

PN76_STATUS_SUCCESS	: SUCCESS is never returned as the system enters into Ultra low power in case of successful entry to ULP mode.
PN76_STATUS_PARAMETER_ERROR	In case invalid wakeup source is selected or wake-up interval is more than 12bits

Figure 6. API: Enter ULP Standby

Table 2. EnterUlpStandby wake-up sources

wWakeupSource	**Behavior**
E_PN76_PCRM_ULP_WAKEUP_SOURCE_TIMER	The chip will enter ULP standby and wake-up after the timer, set by "dwWakeUpCntValue", expires. An external RF-Field is ignored.
E_PN76_PCRM_ULP_WAKEUP_SOURCE_RFFIELD	The chip will enter ULP standby and wake-up either after the timer expires or an external RF-Field is detected.

2.4.2 Wake-up reasons in ULP standby

2.4.2.1 Wake-up timer

The wake-up timer can be set in the resolution of 1 ms from a minimum of 1 ms up to a maximum of 4096 ms (4.096 seconds). In ULP only a very small part of the hardware is running to achieve the maximum in power saving.

After the timer elapses, driven by ULFO, the PN76 wakes up and the user shall check the register PCRM_SYS_BOOT1_STS, see Section 3, to determine the wake-up reason.

2.4.2.2 External RF detected

If the wake-up source "E_PN76_PCRM_ULP_WAKEUP_SOURCE_RFFIELD " is enabled, the RF level detector is enabled while in ULP standby. Any detection of an external RF-Field causes the immediate wake-up from ULP standby with the boot reason "BOOT_RX_ULPDET" (see register PCRM_SYS_BOOT1_STS 0xC6).

3 Boot reasons

Before the example, *nfc_low_power_modes*, prompts the options to exercise different low-power modes, the boot reason of the IC is checked. This check is done in the method `BootLowPowerCheck()` and serves multiple purposes.

First, the boot reason is evaluated to decide which further actions must be taken. An example is going to ULPCD card detection, after the calibration cycle. Second, to display the boot reason in the debug output.

The cause of boot can be retrieved by reading the register `PCRM_SYS_BOOT1_STS`.

Table 3. PCRM_SYS_BOOT1_STS_REG bit fields

Bits	Field	Access	Reset Value	Description
0	BOOT_POR	r	0x0	Bootup Reason
1	BOOT_RXPROT	r	0x0	Bootup Reason
2	BOOT_VUPDET	r	0x0	Bootup Reason
3	BOOT_TEMP	r	0x0	Bootup Reason
4	BOOT_WUC	r	0x0	Bootup Reason
5	BOOT_VDDIO_START	r	0x0	Bootup Reason this is valid if STBY/SUSPEND entered with VDDIO LOSS
6	BOOT_VDDIO_LOSS	r	0x0	Bootup Reason
7	BOOT_PVDDLDO_ OVERCURRENT	r	0x0	Bootup Reason
8	RESERVED	-	0x0	Reserved
9	RESERVED	-	0x0	Reserved
10	BOOT_ULPCD_GPADC_ READY_TIMEOUT	r	0x0	ULPCD exit as GPADC READY is not asserted from GPADC Analog
11	BOOT_RX_ULPDET	r	0x0	Bootup Reason
12	BOOT_LPDET	r	0x0	Bootup Reason
13	BOOT_GPIO0	r	0x0	Bootup Reason
14	BOOT_GPIO1	r	0x0	Bootup Reason
15	BOOT_GPIO2	r	0x0	Bootup Reason
16	BOOT_GPIO3	r	0x0	Bootup Reason
17	RESERVED	r	0x0	Reserved
18	RESERVED	r	0x0	Reserved
19	BOOT_I2C	r	0x0	Bootup Reason
20	BOOT_SPI	r	0x0	Bootup Reason
21	WAKEUP_RX_ULP	r	0x0	Indicates if VEN is masked due to wake-up with rx_ulp. This bit is cleared with ULPDET_WKUP_VEN_ MASK_CLR
22	BOOT_ULPCD_RX_ULPDET	r	0x0	RX ULPDET resulted in boot in ULPCD mode
23	RESERVED	r	0x0	Reserved
24	BOOT_USB	r	0x0	Bootup Reason
25	BOOT_ULPCD_LDO_ VDDPA_OVERCURRENT	r	0x0	ULPCD LDO VDDPA overcurrent
26	BOOT_ULP_STANDBY	r	0x0	ULPCD Standby
27	BOOT_ULPCD_GPIO_ ABORT	r	0x0	ULPCD GPIO Abort

Table 3. PCRM_SYS_BOOT1_STS_REG bit fields...*continued*

Bits	Field	Access	Reset Value	Description
28	BOOT_ULPCD_ CALIBRATION_DONE	r	0x0	ULPCD calibration complete
29	BOOT_ULPCD_CARD_ DETECT	r	0x0	ULPCD card detect
30	BOOT_ULPCD_CLKDET_ ERROR	r	0x0	ULPCD CLK_DET error
31	BOOT_ULPCD_XTAL_ TIMEOUT	r	0x0	ULPCD exit due to XTAL timeout

3.1 GPADC_READY_TIMEOUT

The `BOOT_ULPCD_GPADC_READY_TIMEOUT` in `PCRM_SYS_BOOT1_STS_REG` is set in case the GPADC in ULPCD was not ready in time. This can happen due to various reasons. In particular in firmware lower than v02.05, the timings of GPADC are set stricter than needed to account for the variety of crystals on the market together with manufacturing spread.

In firmware v02.05, the GPADC timings have been adjusted. For for more information, refer to RN00257 ref.[7] and AN14518 ref.[8].

3.2 GPADC_ERROR (0x001E)

Application using ULPCD gets an error during the call *phNfcLib_Init()* as *GPADC_ERROR (0x001E)* after the IC woke up due to either ULPCD_CALIBRATION or ULPCD_DETECTION operation. Recovery from failure shall require a hard reset of the IC. This can be achieved by either calling PN76_Sys_Hal_HardReset() or VEN toggle.

Refer to the following documentation for more information, refer to ref.[9].

4 EEPROM read/writes and IC configuration

Caution: EEPROM writes must be used with extreme care, due to limited write cycles (100k) and potential for corruption if power is lost during the operation! A reset or power loss during an EEPROM write can corrupt the currently written EEPROM page (512 bytes). If an EEPROM write is performed make sure that stable power is available and no reset is performed until the write is complete.

The *"nfc_low_power_mode"* example requires some basic configuration of the power domain and HFATT value. From SDK v02.15.003 onwards, the example itself will not execute any EEPROM writes and configurations. It will inform the user if conflicting or unexpected configuration is found. The user is responsible to ensure that the PN7642 is properly configured. To assist with that task, a new example, namely "*NFC_Config*", is introduced. This further outlines the importance of separating IC configuration and business logic.

```
Configurations Example:
Select the Modes :
1 -> Set ULPCD Power Configuration :
2 -> Read ULPCD HFATT and save to EEPROM:
3 -> Reset Power Configuration:
4 -> Toggle Random UID Configuration:
5 -> Configure Autocoll Parameters :
6 -> Set RFLD and NFCLD Configuration : 1
1
ULPCD Power Configuration success!!
```

Figure 7. Example "NFC Config" execution

Note: *This application note, at the current version, focuses on the usage of the example and the code integration. Detailed calibration and configuration of ULPCD is explained in ref.[4] chapter 4.2 "ULPCD".*

4.1 Power configuration

See ref.[4] Section 2.1 "DC-DC and TXLDO" for more details.

Low-power modes such as ULP or ULPCD require the DC/DC to be disabled.

Note: *Disabling the DC-DC requires the supply of VUP as well as related EEPROM settings (DCDC_POWER_CONFIG, 0x00). Wrong settings in combination with certain power configurations might destroy the PN7642!*

To disable the DC/DC the EEPROM parameter "DCDC_PWR_CONFIG (0x0000)" must be written.

Further, it is recommended to set the TX LDO Start VDDPA: TXLDO_VDDPA_CONFIG (0x0006).

All this is also done by running the example *NFC_Config* with option "1" (Set ULPCD Power Configuration).

4.2 ULPCD HFATT value

See ref.[4] Section 4.2.1 "HF attenuator value" for more details.

For the correct working of the ULPCD the HF attenuator value must be set once (!) in the EEPROM. The HF attenuator value guarantees the optimum RSSI range, even though the VDDPA might be reduced.

The best way to derive the optimum HF attenuator value is to enable the RF-Field once, using Type-A 106 settings, and forcing the VDDPA to the same value as set in ULPCD_VDDPA_CTRL. Let the RF-Field settle for a short time (for example, 1 ms) before reading CLIF_RXCTRL_STATUS (0x28) - [8:3] RXCTRL_HF_ATT_VAL.

The read out value must be written to ULPCD_AGC_HFATT_CTRL (0x63A, bits[0:7]). To enable the usage of this value in ULPCD the highest bit, bit 7, must be set to "1".

All this is also done by the SDK example *NFC_Config* by executing option "2" (Read ULPCD HFATT and save to EEPROM).

4.2.1 HFATT value code snippets

All relevant code can be read in the method "*ULPCD_Get_HFATT_Value(…)*" of the *"NFC_Config*" example.

1. Load the RF config for Type-A 106:

```
/* Apply Reader Mode Type A settings. */
    wStatus = PN76_Sys_LoadRfConfiguration(E_PN76_LOADRF_TX_ISO14443A_106,
 E_PN76_LOADRF_RX_ISO14443A_106);
    PN76_CHECK_SUCCESS(wStatus);
```

2. Turn on the RF-Field and let it settle:

```
/* Turn ON the RF Field. */
    wStatus = PN76_Sys_FieldON(0);
    PN76_CHECK_SUCCESS(wStatus);

    /* Wait for sometime to stabilize RF ON before reading HF ATT value. */
    LowPowerMode_CommonWait(1500);
```

3. Read the HF attenuator value:

```
/* Read HF ATT value. */
    dwReadReg_Val = PN76_Sys_ReadRegister(CLIF_RXCTRL_STATUS);

    /* Get the HF ATT Value and store in EEPROM to be used for ULPCD  */
    bHFATT_Val = (uint8_t)((dwReadReg_Val & CLIF_RXCTRL_STATUS_RXCTRL_HF_ATT_VAL_MASK)
 >>
                            CLIF_RXCTRL_STATUS_RXCTRL_HF_ATT_VAL_POS);
```

4. Mask the highest bit to enable this value in ULPCD:

```
bHFATT_Val |= 0x80;
```

5. Write new HF attenuator value to ULPCD_AGC_HFATT_CTRL:

```
wStatus = PN76_WriteEeprom(&bHFATT_Val, (uint32_t)PN76_ULPCD_CONFIG->Agc_Hfatt_Ctrl,
0x01U,
                                   E_PN76_EEPROM_SECURE_LIB_CONFIG);
        PN76_CHECK_SUCCESS(wStatus);
```

4.3 RFLD and NFCLD calibration

The PN7642 implements an RF level detector (RFLD) and NFC level detector (NFCLD) which allows to detect the presence of an external RF field.

RF Level Detector:

The purpose of the RFLD function is to detect any signal at 13.56 MHz.

During low-power card detection (LPCD), the RF level detector (RFLD) acts as a wake-up source from power-saving mode.

During ultra low-power card detection (ULPCD), a specific ultra low-power RF level detector is used as RF level detector(RFLD). This can be enabled as a wake-up source.

NFC Level Detector:

The NFC Level detector (NFCLD) is used during full power mode. The NFCLD function is required by the NFC Forum to support the "RF collision avoidance".

The sensitivity of the NFCLD sensor can be configured by EEPROM register to meet the NFC Forum requirements.

It can be used as well in card mode to detect an external field.

Calibration:

In ULPCD one wake-up source can be the detection of an external RF-Field. It is recommended to calibrate the RFLD and NFCLD first. The example "*NFC_Config"* offers the option for calibration.

For calibration the PN7642 offers the following API: **PN76_Sys_Configure_RFLD_NFCLD()**

Re-Calibration:

A recalibration is only done if the new values are different from the existing values. It is recommended to reset the following values to ensure a proper recalibration:

1. PN76_EXT_RF_DETECTION_HW->dwNFCLD_OnOffThreshold
   a. location: 0x0597
   b. E_PN76_EEPROM_SECURE_LIB_CONFIG
   c. reset value: 0x00000000
2. PN76_EXT_RF_DETECTION_HW->dwLPDET_AnaCtrl_Wkup
   a. location: 0x05A5
   b. E_PN76_EEPROM_SECURE_LIB_CONFIG
   c. reset value: 0x00000000
3. PN76_EXT_RF_DETECT_INT_INPUT->wNFCLD_RFLD_Valid_Bit
   a. location: 0x05C5
   b. E_PN76_EEPROM_SECURE_LIB_CONFIG
   c. reset value: 0x00

By checking *NFCLD_RFLD_VALID (0x05C5)* the user can make sure that the calibration is done properly.

4.3.1 Reset EEPROM values code snippet

The following code snippet will read and reset the EEPROM values necessary for recalibration:

```
uint8_t bReadEepromVal[4] = {0};
uint8_t bResetValue[4] = {0};

/*
 * location: 0x0597,
 * E_PN76_EEPROM_SECURE_LIB_CONFIG,
 * 4-bytes
 * PN76_EXT_RF_DETECTION_HW->dwNFCLD_OnOffThreshold = 0x00000000;
 * */
status = PN76_ReadEeprom((uint8_t *)bReadEepromVal, (uint32_t)PN76_EXT_RF_DETECTION_HW-
>dwNFCLD_OnOffThreshold, 0x04U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
APP_PRINTF("\nBefore - dwNFCLD_OnOffThreshold: %08x", (uint32_t *)bReadEepromVal[0]);
status = PN76_WriteEeprom(bResetValue, (uint32_t)PN76_EXT_RF_DETECTION_HW-
>dwNFCLD_OnOffThreshold, 0x04U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
status = PN76_ReadEeprom(bReadEepromVal, (uint32_t)PN76_EXT_RF_DETECTION_HW-
>dwNFCLD_OnOffThreshold, 0x04U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
APP_PRINTF("\nAfter - dwNFCLD_OnOffThreshold: %08x", (uint32_t *)bReadEepromVal[0]);

/*
 * location: 0x05A5,
 * E_PN76_EEPROM_SECURE_LIB_CONFIG,
 * 4-bytes
 * PN76_EXT_RF_DETECTION_HW->dwLPDET_AnaCtrl_Wkup = 0x00000000;
 * */
status = PN76_ReadEeprom(bReadEepromVal, (uint32_t)PN76_EXT_RF_DETECTION_HW-
>dwLPDET_AnaCtrl_Wkup, 0x04U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
APP_PRINTF("\nBefore - dwLPDET_AnaCtrl_Wkup: %08x", (uint32_t *)bReadEepromVal[0]);
status = PN76_WriteEeprom(bResetValue, (uint32_t)PN76_EXT_RF_DETECTION_HW-
>dwLPDET_AnaCtrl_Wkup, 0x04U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
status = PN76_ReadEeprom(bReadEepromVal, (uint32_t)PN76_EXT_RF_DETECTION_HW-
>dwLPDET_AnaCtrl_Wkup, 0x04U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
APP_PRINTF("\nAfter - dwLPDET_AnaCtrl_Wkup: %08x", (uint32_t *)bReadEepromVal[0]);

/*
 * location: 0x05C5,
 * E_PN76_EEPROM_SECURE_LIB_CONFIG,
 * 1-byte
 * PN76_EXT_RF_DETECT_INT_INPUT->wNFCLD_RFLD_Valid_Bit = 0x00;
 * */
status = PN76_ReadEeprom(bReadEepromVal, (uint32_t)PN76_EXT_RF_DETECT_INT_INPUT-
>wNFCLD_RFLD_Valid_Bit, 0x01U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
APP_PRINTF("\nBefore - wNFCLD_RFLD_Valid_Bit: %02x", bReadEepromVal[0]);
status = PN76_WriteEeprom(&bResetValue[0], (uint32_t)PN76_EXT_RF_DETECT_INT_INPUT-
>wNFCLD_RFLD_Valid_Bit, 0x01U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
status = PN76_ReadEeprom(bReadEepromVal, (uint32_t)PN76_EXT_RF_DETECT_INT_INPUT-
>wNFCLD_RFLD_Valid_Bit, 0x01U, E_PN76_EEPROM_SECURE_LIB_CONFIG);
APP_PRINTF("\nAfter - wNFCLD_RFLD_Valid_Bit: %02x", bReadEepromVal[0]);
```

5 NFC low-power mode example

The MCUXpresso SDK (see ref.[5]) has a low-power mode example called `nfc_low_power_mode`, which includes different low-power modes. The following chapters give detailed information about the example, and highlights the APIs and their usage.

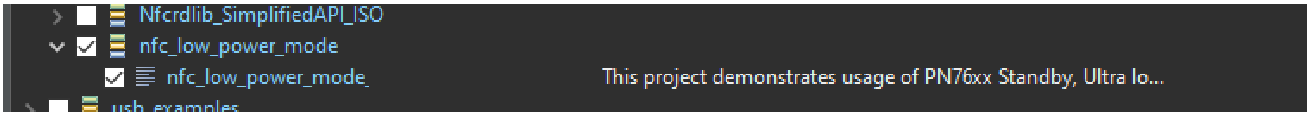


Figure 8. MCUXpresso low-power mode example

The purpose of the example is to showcase certain functionality. For product purposes, it is recommended for users to familiarize themselves with the used APIs and alter the example to their needs. Transfer the necessary additions to the application code.

Instructions for installing MCUXpresso, importing the SDK and its examples can be found in the PN76 family evaluation board quick start guide ref.[6].

Running the low-power mode example presents the following options:

```
00>  Low Power Mode Example:
00>  Select the Modes :
00>  1 -> Enter Standby with Counter as Wakeup :
00>  2 -> Enter Standby with External RF as Wakeup :
00>  3 -> Enter Ultra Low Power Standby with Counter as Wakeup :
00>  4 -> Enter Ultra Low Power Standby with External RF as Wakeup :
00>  5 -> Enter ULPCD Calibration :
00>  6 -> Enter ULPCD Detection :
00>  7 -> Enter LPCD Calibration :
00>  8 -> Enter LPCD Detection:
```

Figure 9. Low-power mode example options

The following chapters explain Section "Option 3: Enter ultra low-power standby with counter as wake-up", Section "Option 4: Enter ultra low-power standby with external RF as wake-up", and Section "Option 5: ULPCD Calibration" in detail, which APIs are used, the expected behavior, and how to apply these options to the user's application code.

5.1 Option 1+2: Enter Standby

Option 1 and 2 shows how the PN7642 can be set in standby with different wake-up sources. To enter standby, the API "*PN76_Sys_PCRM_EnterStandby(..)*" with the wake-up sources of choice is called.

```
        /* Ensuring LPCD is disabled */
        PCRM_CLEARBITN(PCRM_GPREG6, PHHAL_HW_PN76XX_GPREG_6_LPCD_ENABLED_POS);
        wStatus = PN76_Sys_PCRM_EnterStandby((PN76_PCRM_LowPower_WakeUp_Config_t)wWakeUpSource, wWakeupCnt);
        if(wStatus)
        {
            APP_PRINTF("\n Standby Entry FAILED!!!!! : ");
        }
```

Figure 10. EnterStandby API usage

Option 1:

The only wake-up source selected is the timer "*SMU_WAKEUP_SOURCE_TIMER*". There are many more wake-up sources as choices available (see the API description for more details), which can be easily selected by adding to or replacing the current choice.

```
00>
00>  Low Power Mode Example:
00>  Boot after Standby with WakeUp counter expiry
00>  Select the Modes :
00>  1 -> Enter Standby with Counter as Wakeup :
00>  2 -> Enter Standby with External RF as Wakeup :
00>  3 -> Enter Ultra Low Power Standby with Counter as Wakeup :
```

Figure 11. Exit standby due to counter expiry

After expiry, the PN7642 will wake up with the boot reason set to "*PCRM_SYS_BOOT1_STS_BOOT_WUC_MASK*".

Option 2:

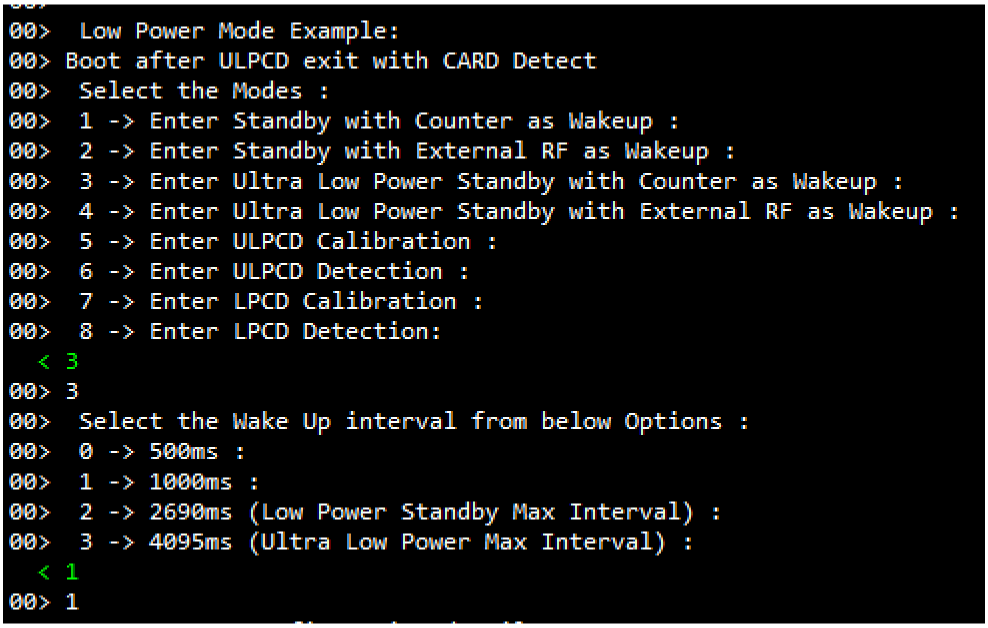
The wake-up source is the external RF-Field. The PN7642 will not wake-up due to counter or anything else.

```
00>
00>  Low Power Mode Example:
00>  Boot from Standby due to External RF detection
00>  Select the Modes :
00>  1 -> Enter Standby with Counter as Wakeup :
00>  2 -> Enter Standby with External RF as Wakeup :
```

Figure 12. Exit standby due to external RF-Field

5.2 Option 3: Enter ultra low-power standby with counter as wake-up

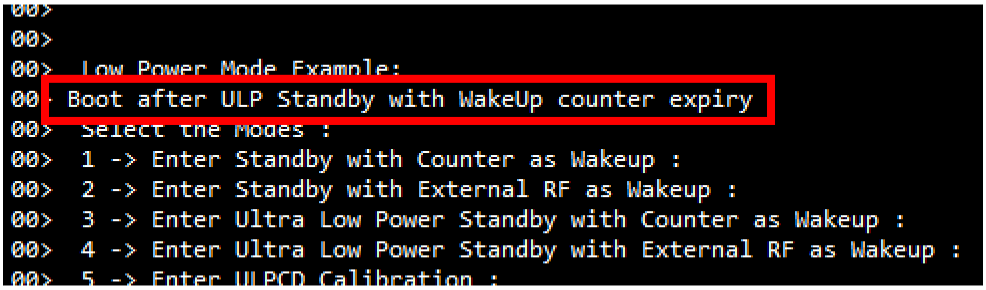
Option 3 of the low-power modes example sets the chip into ultra low-power standby. The wake-up counter can be set to predefined options (in Figure 13, the wake-up counter "1" represents 1 second):



```
00>  Low Power Mode Example:
00> Boot after ULPCD exit with CARD Detect
00>  Select the Modes :
00>  1 -> Enter Standby with Counter as Wakeup :
00>  2 -> Enter Standby with External RF as Wakeup :
00>  3 -> Enter Ultra Low Power Standby with Counter as Wakeup :
00>  4 -> Enter Ultra Low Power Standby with External RF as Wakeup :
00>  5 -> Enter ULPCD Calibration :
00>  6 -> Enter ULPCD Detection :
00>  7 -> Enter LPCD Calibration :
00>  8 -> Enter LPCD Detection:
  < 3
00> 3
00>  Select the Wake Up interval from below Options :
00>  0 -> 500ms :
00>  1 -> 1000ms :
00>  2 -> 2690ms (Low Power Standby Max Interval) :
00>  3 -> 4095ms (Ultra Low Power Max Interval) :
  < 1
00> 1
```

Figure 13. Option 3 execution

After the interval selection, it immediately enters low-power mode. After 1 second, the chip wakes up again. If we reconnect the RTT viewer, we can see that the boot reason is wake-up counter expiry (see Figure 14):



```
00>
00>  Low Power Mode Example:
00> Boot after ULP Standby with WakeUp counter expiry
00>  Select the Modes :
00>  1 -> Enter Standby with Counter as Wakeup :
00>  2 -> Enter Standby with External RF as Wakeup :
00>  3 -> Enter Ultra Low Power Standby with Counter as Wakeup :
00>  4 -> Enter Ultra Low Power Standby with External RF as Wakeup :
00>  5 -> Enter ULPCD Calibration :
```

Figure 14. Boot reason counter expiry

5.2.1 Code flow

The code flow for this option is rather simple. In the *switch block* the case `ULP_STANDBY_WITH_WAKEUP_COUNTER_MODE` is taken, the method `Demo_UltraLowPowerStandby(…)` with the chosen interval is called.

The method `Demo_UltraLowPowerStandby(…)` is simple as well; only calling the system service API from the PN76 to enter ultra low-power standby: `PN76_Sys_PCRM_EnterUlpStandby(…)`

```
PN76_Status_t Demo_UltraLowPowerStandby(uint16_t wWakeUpSource, uint16_t wWakeUpIntervalInMs)
{
    PN76_Status_t wStatus = PN76_STATUS_SUCCESS;

    do{
       PN76_Sys_PCRM_EnterUlpStandby((PN76_PCRM_UltraLowPower_WakeUp_Config_t)wWakeUpSource, wWakeUpIntervalInMs);

         APP_PRINTF("\r\n Ultra Low Power Standby Entry FAILED!!!!! : ");

       wStatus = PN76_STATUS_INTERNAL_ERROR;
     } while (0);

     return wStatus;
}
```

Figure 15. API PCRM EnterUlpStandby

5.3 Option 4: Enter ultra low-power standby with external RF as wake-up

This option is calling the ultra low-power standby method with the configuration parameter set to `E_PN76_PCRM_ULP_WAKEUP_SOURCE_RFFIELD`. This sets the chip into *ultra low-power standby* with wake-up either by timer expiry or external RF field.

See Section 2.4.1 for more details of the API.

Selecting option 4 and interval "2" (which provides more time to manually place something on the antenna that emits an RF-Field and would therefore wake the chip) will put the chip into ULP standby (see Figure 16):

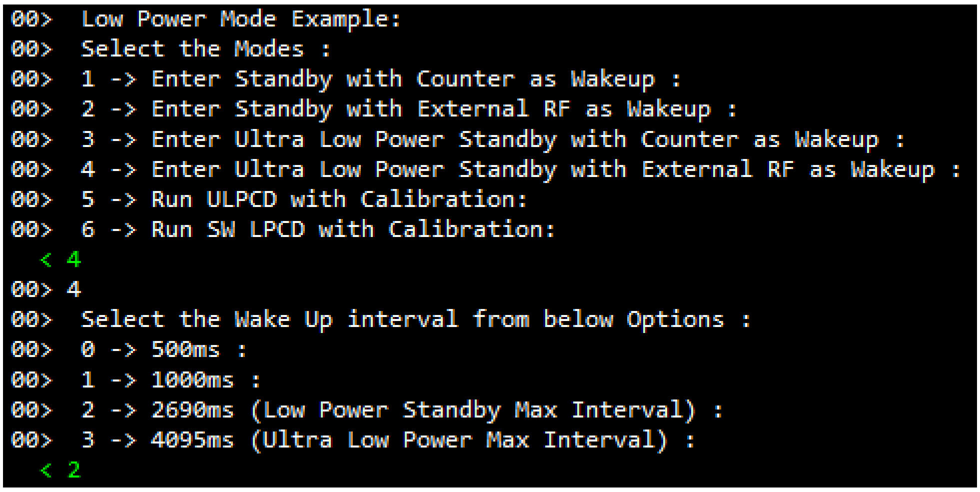


Figure 16. Option 4 execution

Bringing a device, which emits an RF-Field (for example, cell phone), into the antennas area wakes up the chip with the boot reason of external RF detected (see Figure 17, do not forget to reconnect the RTT viewer as connection is lost when entering ULP):

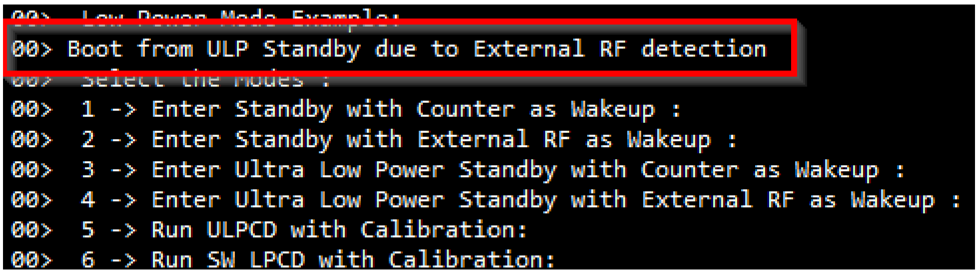


Figure 17. Boot due to external RF

5.4 Option 5: ULPCD Calibration

This option calibrates ULPCD. For the ULPCD calibration, some additional parameters, such as power configuration and HFATT value, must be configured first. Those configurations involve EEPROM writes and is outsourced to a new example called "*NFC_Config*".

Note: *Detailed calibration and configuration of ULPCD is explained in ref.[4] Section 4.2 "ULPCD". In addition, the example NFC_Config can be read to see how the configuration is done.*

To make the PNEV7642 ready for ULPCD, run the example "*NFC_Config*" with the options [1] "Set ULPCD Power Configuration" and [2] "Read ULPCD HFATT and save to EEPROM". This only has to be done once. After that, the ULPCD example can be run as many times as wanted.

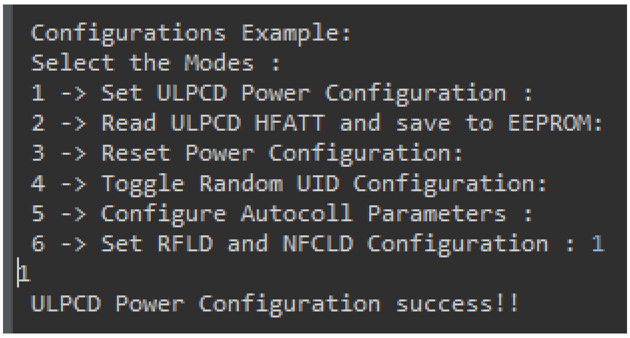


Figure 18. NFC Config example for ULPCD

By executing the "*nfc_low_power_mode"* example with ULPCD calibration, the PN7642 enters ULPCD for calibration. Debugger connection will be lost as any ULP (Ultra Low Power) mode causes detachment of the debugger.

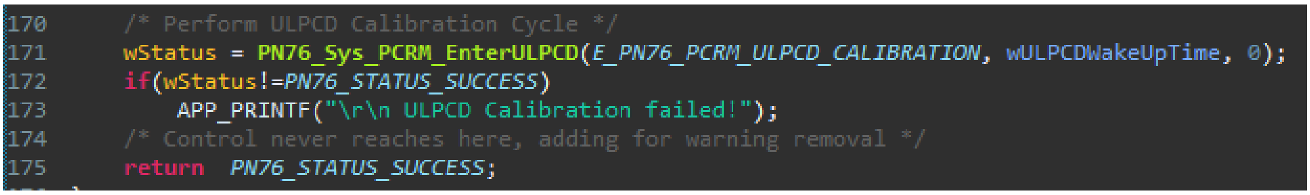


Figure 19. ULPCD Calibration call

After calibration is done, the PN7642 will boot up with the boot reason "*PCRM_SYS_BOOT1_STS_BOOT_ULPCD_CALIBRATION_DONE_MASK*". The boot reason is checked in the method "*BootLowPowerCheck(..)*". If you reconnect the RTT viewer the "ULPCD calibration is done !!" message should appear.

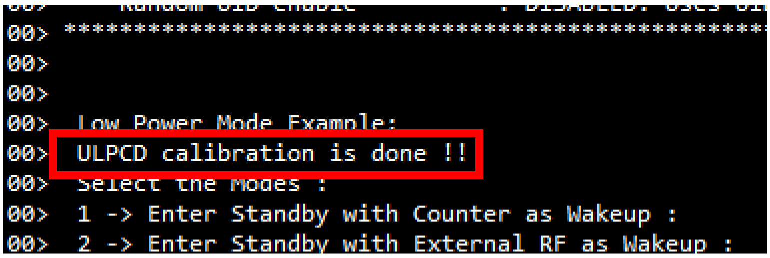


Figure 20. ULPCD calibration done

5.5 Option 6: ULPCD Detection

Before the ULPCD detection can be performed make sure to perform the calibration (see Section 5.4 "Option 5: ULPCD Calibration").

```
PN76_Status_t ULPCD_Demo(uint16_t wLPCDWakeUpTime)
{
    phStatus_t wStatus;
    uint8_t bReadEEPROM_Val=0;
    uint8_t bUlpcdHFATTCalibDone=0;
    /*Disable SW_LPCD*/
    PCRM_CLEARBITN(PCRM_GPREG6, PHHAL_HW_PN76XX_GPREG_6_LPCD_ENABLED_POS);
    /*Check Power config from EEPROM */
    wStatus = PN76_ReadEeprom(&bReadEEPROM_Val,(uint32_t )PN76_USER_PMU->PwrConfig, 0x01U,E_PN76_EEPROM_SECURE_LIB_CONFIG)

    PN76_CHECK_SUCCESS(wStatus);

    if (bReadEEPROM_Val != LOWPOWERMODE_EXAMPLE_ULPCD_VUP_TO_VBAT_CONFIG_VAL)
    {
        APP_PRINTF("\r\n Wrong Power Config for ULPCD");
        return PN76_STATUS_FAILED;
    }

    wStatus = PN76_ReadEeprom(&bUlpcdHFATTCalibDone,(uint32_t )PN76_ULPCD_CALIB_STATUS->bCalibDone, 0x01U,
                              E_PN76_EEPROM_USER_AREA);

    PN76_CHECK_SUCCESS(wStatus);

    if(bUlpcdHFATTCalibDone==0)
    {
        APP_PRINTF("\r\nHFATT calibration is not done!! ULPCD Calib aborted!");
        return PN76_STATUS_FAILED;
    }

    /* Perform ULPCD Detection Cycle */
    wStatus = PN76_Sys_PCRM_EnterULPCD(E_PN76_PCRM_ULPCD_CARD_DETECTION, wLPCDWakeUpTime, 0);

    PN76_CHECK_SUCCESS(wStatus);
    /* Control never reaches here, adding for warning removal */
    return  PN76_STATUS_SUCCESS;
}
```

Figure 21. ULPCD Detection

Before entering ULPCD, some parameters are checked to ensure the power config is correct and calibration has been performed. After that, "*PN76_Sys_PCRM_EnterULPCD(…)*" is called to enter ULPCD. Per default the external RF detection is disabled. To enable it, change the third parameter to "1".

On the PNEV7642 evaluation board, the user can observe the blue LED pulse on every ~320 ms (ULPCD RF field cycle). During this time, the RF field is turned on for a short period. The RF-On duration can be configured to check for any potential detuning.

Inspecting the RF field with an oscilloscope shows a short pulse every ~320 ms (the time between ULPCD pings depends on the entered configuration).

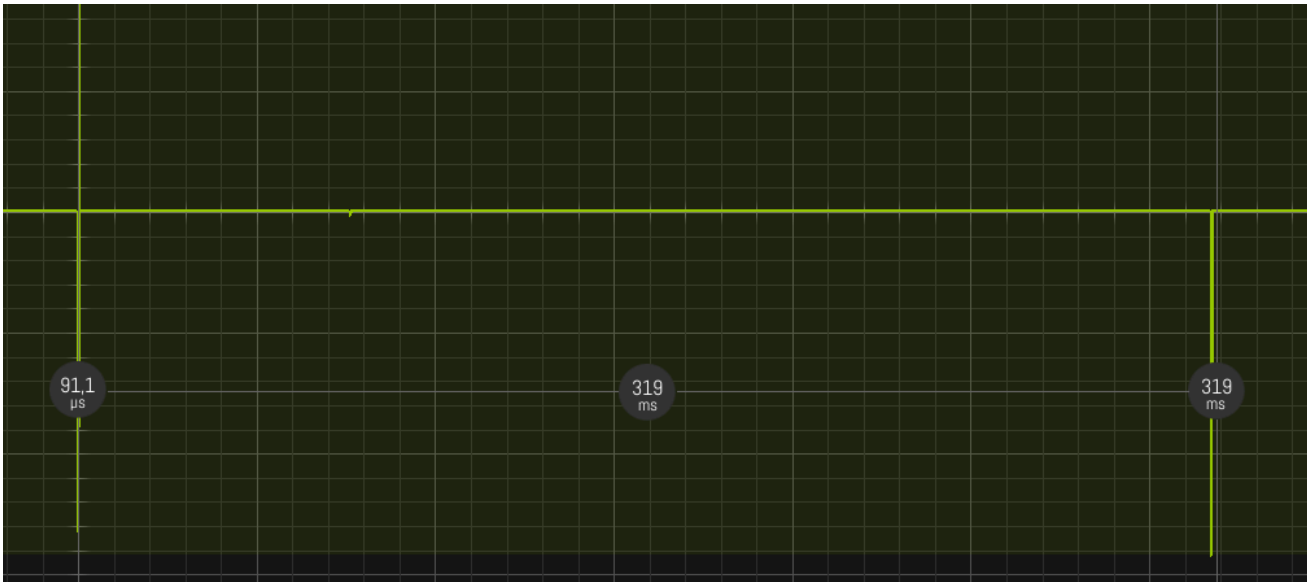


Figure 22. ULPCD oscilloscope inspection

If users bring a card into the RF field, the chip wakes up with the boot reason "*PCRM_SYS_BOOT1_STS_ BOOT_ULPCD_CARD_DETECT_MASK*". Reconnect the RTT viewer to see the output since the connection is lost when entering ULPCD.

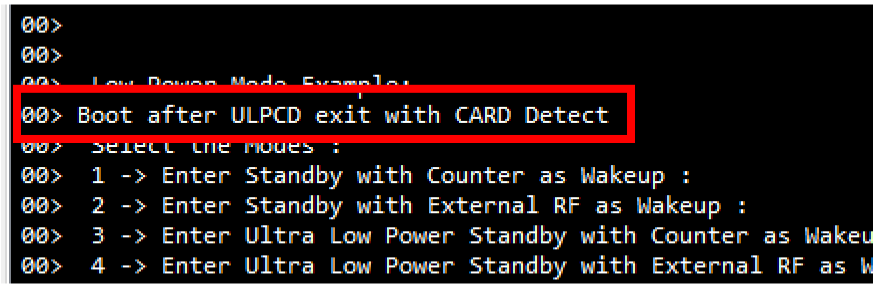


Figure 23. ULPCD card detected

5.5.1 Abort with GPIO3

Another option to exit ULPCD mode is abort via GPIO3. The development board can have multiple configurations. The option shown below is not the most elegant, but is one that works without regard for the board configuration.

To abort via GPIO3, we have to apply +3.3 V to GPIO3.

Use a jumper wire to connect J47.4 (P3V3_BRD) to pin 30 on the module board:

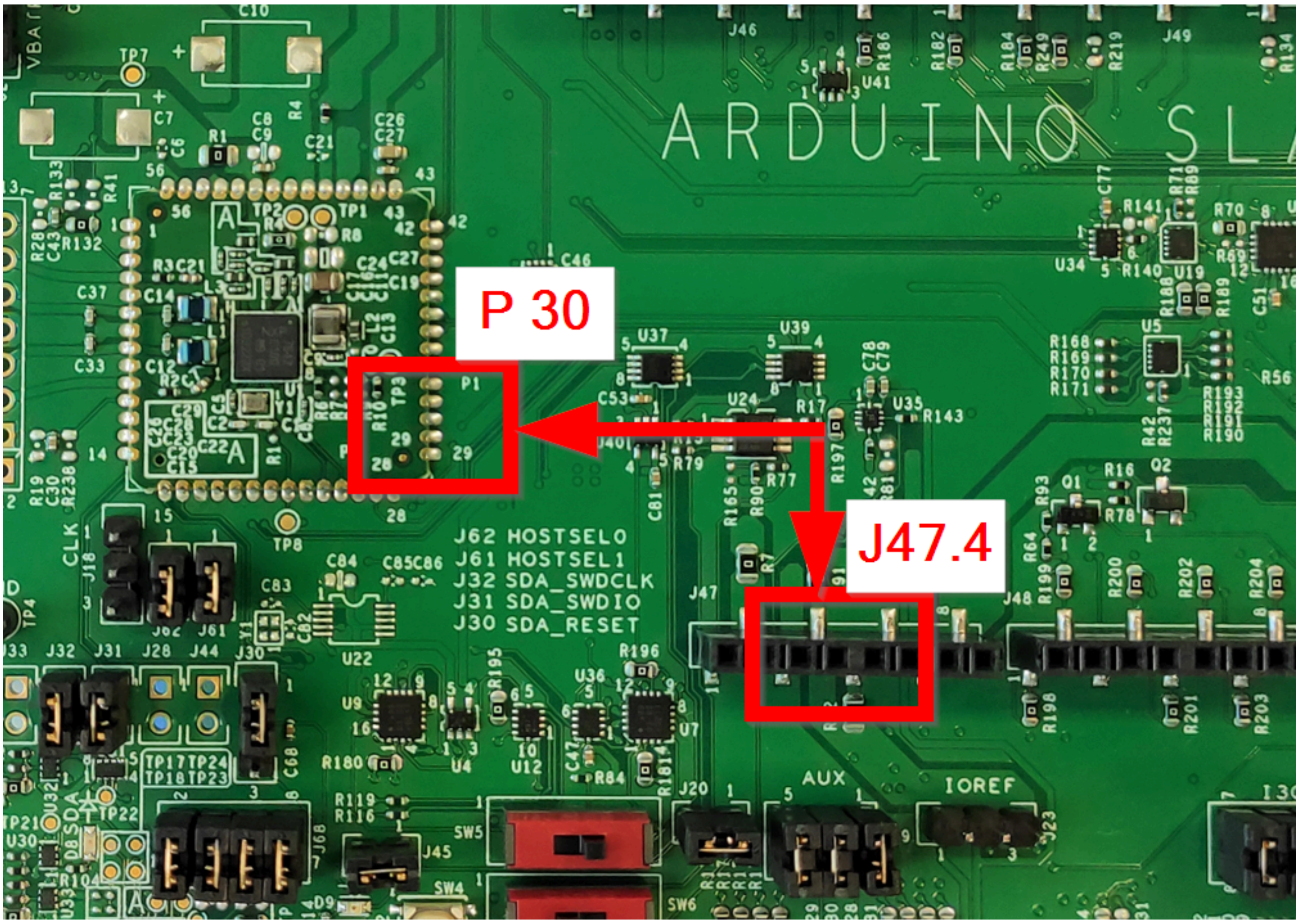


Figure 24. GPIO3 abort wiring

P.30 is the second pin of the right row (above 29).

If the chip is woken up, the green LED is turned on.

5.6 Option 7: LPCD Calibration

The low-power card detection (LPCD) section of the PN7642 Datasheet ref.[1] outlines how the LPCD APIs should be used for calibration and detection. Option 7 of the low-power mode example performs an LPCD calibration.

For a more detailed explanation of LPCD and its calibration including adjustments such as the average samples and I/Q channel threshold, see ref.[4].

```
uint32_t LPCD_Calib(void)
{
    PN76_Status_t wStatus = PN76_STATUS_INTERNAL_ERROR;
    uint32_t dwReference=0;

    /* Enable LPCD in GPREG6 */
    PCRM_SETBITN(PCRM_GPREG6, PHHAL_HW_PN76XX_GPREG_6_LPCD_ENABLED_POS);
    wStatus = PN76_Sys_PCRM_LPCD_Start(E_PN76_LPCD_REFERENCE_BOOT);
    if(wStatus != PN76_STATUS_NO_TAG_DETECTED)
    {
        if(wStatus == PN76_STATUS_EXTERNAL_FIELD )
        {
            APP_PRINTF("\n External Field Detected. LPCD Failed !!!!! : ");
                return 0;
        }
    }
    PN76_Sys_PCRM_LPCD_Start(E_PN76_LPCD_MEASUREMENT_BOOT);
    dwReference =  PN76_Sys_PCRM_GetLpcdReference();
    PCRM_CLEARBITN(PCRM_GPREG6, PHHAL_HW_PN76XX_GPREG_6_LPCD_ENABLED_POS);
     return dwReference ;
}
```

Figure 25. LPCD Calibration method

LPCD Initialization

The first call, at line 319, "*PN76_Sys_PCRM_LPCD_Start*(*E_PN76_LPCD_REFERENCE_BOOT*);" is purely to initialize the LPCD and will not take any reference value at this point.

LPCD Calibration

The second call, at line 328, "*PN76_Sys_PCRM_LPCD_Start(E_PN76_LPCD_MEASUREMENT_BOOT);*" will perform an actual measurement and store the measured I and Q channel values in the internal AO_RAM. Calling "*PN76_Sys_PCRM_GetLpcdReference()*" retrieves the measured values. This is for information only and has no effect on the actual LPCD functionality.

Usage of standby

Depending on the antenna matching, layout, surroundings and LPCD configuration, the LPCD can be very sensitive. Ideally, the calibration is performed as close to the actual LPCD detection loop as possible. If the LPCD detection loop includes standby, it is advised to also use the same standby at LPCD calibration. This can be done between the execution of "*PN76_Sys_PCRM_LPCD_Start*(*E_PN76_LPCD_REFERENCE_BOOT*);" and "*PN76_Sys_PCRM_LPCD_Start(E_PN76_LPCD_MEASUREMENT_BOOT);*". This will ensure that the transmitter and receiver have the same amount of time to settle between the calls as in the actual detection loop.

How to use standby is explained in Section 5.1 "Option 1+2: Enter Standby".

PCRM_GPREG6

This GPREG (General-Purpose Register) is not related to the LPCD functionality but is used in the *nfc_low_power_mode* example to carry the Enabled/Disabled status of LPCD throughout the examples execution.

5.7 Option 8: LPCD Detection

Before the LPCD detection is executed make sure that LPCD has been calibrated (see Section 5.6).

In the method "*Demo_SW_LPCD(uint16_t wWakeUpIntervalMs)*" the "*PCRM_GREP6"* register will be set to retain the information if LPCD is enabled or disabled over standby. Afterwards, the PN7642 is set to standby, by calling "*PN76_Sys_PCRM_EnterStandby(…)*", for an x-amount of milliseconds. The wake-up reasons are timer elapsed (*E_PN76_PCRM_WAKEUP_SOURCE_TIMER*) or external RF field detected (*E_PN76_PCRM_WAKEUP_SOURCE_RFFIELD*).

```
PN76_Status_t Demo_SW_LPCD(uint16_t wWakeUpIntervalMs)
    {
    PN76_Status_t wStatus = PN76_STATUS_INTERNAL_ERROR;
    uint16_t wWakeupCnt ;

    /* Enable LPCD in GPREG6 */
    PCRM_SETBITN(PCRM_GPREG6, PHHAL_HW_PN76XX_GPREG_6_LPCD_ENABLED_POS);
    /* Enter Low power mode to ensure on wakeup card detection occurs. */
    wWakeupCnt = CalcWakeupCtrValue(wWakeUpIntervalMs);
    if(wWakeupCnt == 0xFFFF)
    {
      return PN76_STATUS_PARAMETER_ERROR;
    }
    wStatus = PN76_Sys_PCRM_EnterStandby(((PN76_PCRM_LowPower_WakeUp_Config_t)(E_PN76_PCRM_WAKEUP_SOURCE_TIMER |
                                          E_PN76_PCRM_WAKEUP_SOURCE_RFFIELD)),
                                          wWakeupCnt);
    if(wStatus)
     {
       APP_PRINTF("\n Standby Entry FAILED!!!!! : ");
     }
    return wStatus;
 }
```

Figure 26. Demo_SW_LPCD method

After the timer is elapsed or an external RF field is detected, the PN7642 will boot up. The boot reason expected is either "*PCRM_SYS_BOOT1_STS_BOOT_WUC_MASK*", because the timer has elapsed, or "*PCRM_SYS_BOOT1_STS_BOOT_LPDET_MASK*", because an external RF field is detected.

Checking those boot reasons is done in the method "*BootLowPowerCheck()*".

In case the reason is timer elapsed and LPCD is enabled (*PCRM_GPREG6* set), an LPCD measurement ping will be issued:

```
    if (dwBootRegVal & PCRM_SYS_BOOT1_STS_BOOT_WUC_MASK)
    {
      if(PCRM_TESTBITN(PCRM_GPREG6, PHHAL_HW_PN76XX_GPREG_6_LPCD_ENABLED_POS))
      {
         wStatus = PN76_Sys_PCRM_LPCD_Start(E_PN76_LPCD_MEASUREMENT_BOOT);
         if(wStatus == PN76_STATUS_NO_TAG_DETECTED)
         {
            PN76_Sys_PCRM_ReEnterStandby();
         }
         else if(wStatus == PN76_STATUS_TAG_DETECTED)
         {
            APP_PRINTF("\nBoot after LPCD with Card Detect ");
         }
         else if(wStatus == PN76_STATUS_EXTERNAL_FIELD)
         {
            APP_PRINTF("\nBoot after LPCD due to External RF detection ");
         }
         else
         {
            APP_PRINTF("\n LPCD Failed with Unexpected Status ");
         }
      }
      else
      {
          /* If the GPREG6 has ZERO then its standby wakeup with counter */
          APP_PRINTF("\r\nBoot after Standby with WakeUp counter expiry ");
    }
   }
```

Figure 27. LPCD measurement after boot

If no tag has been detected, the PN7642 will re-enter standby, by calling the API "*PN76_Sys_PCRM_ReEnterStandby()*". This closes the detection loop. The PN7642 will again wake up after time elapse or external RF field detected and again issue an LPCD measurement ping.

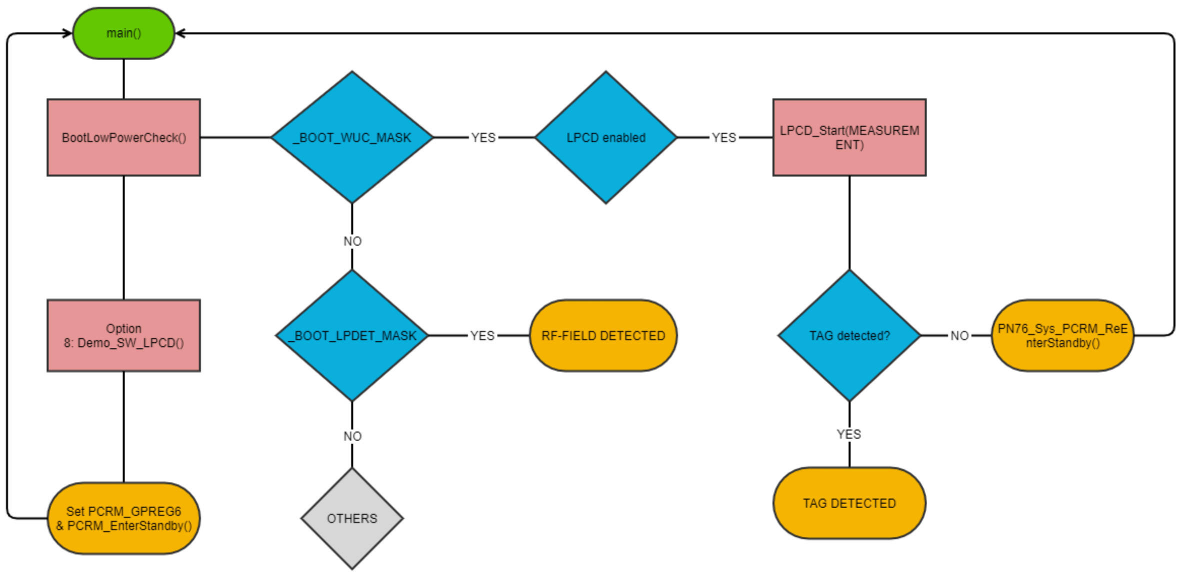


Figure 28. LPCD detection flowchart

6 RTT Viewer

The J-Link RTT Viewer is a GUI tool from SEGGER, which attaches to either an existing J-Link connection of a debugger or opening a connection on its own. Instead of having the IDE console as input and output, the J-Link RTT Viewer is used. This option is more lightweight, easier to reattach.

At the import of the example, you can choose "UART" or "Semihost". UART represents the RTT Viewer option and Semihost the output in the IDE console. You can switch the debug console at any time, using the quick settings: Quick Settings → SDK Debug Console → Semihost/UART

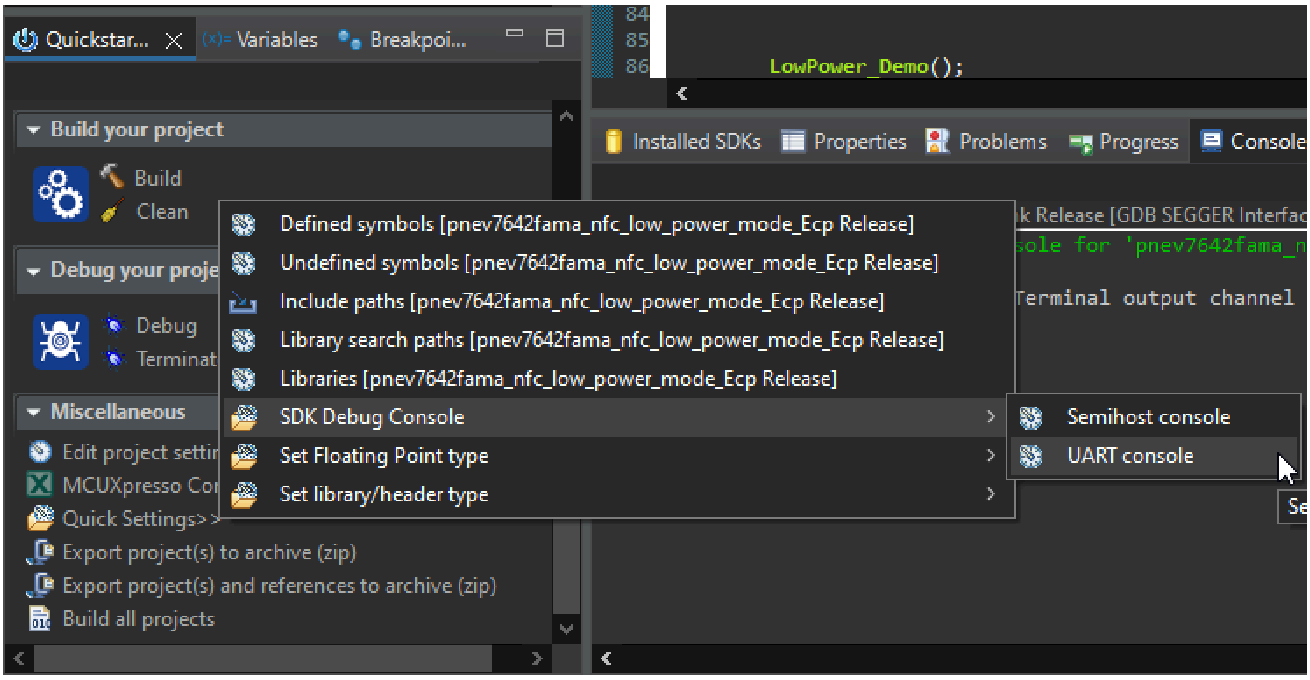


Figure 29. MCUXpresso Debug Console Option

The configuration to attach to the PN7642 is as below:

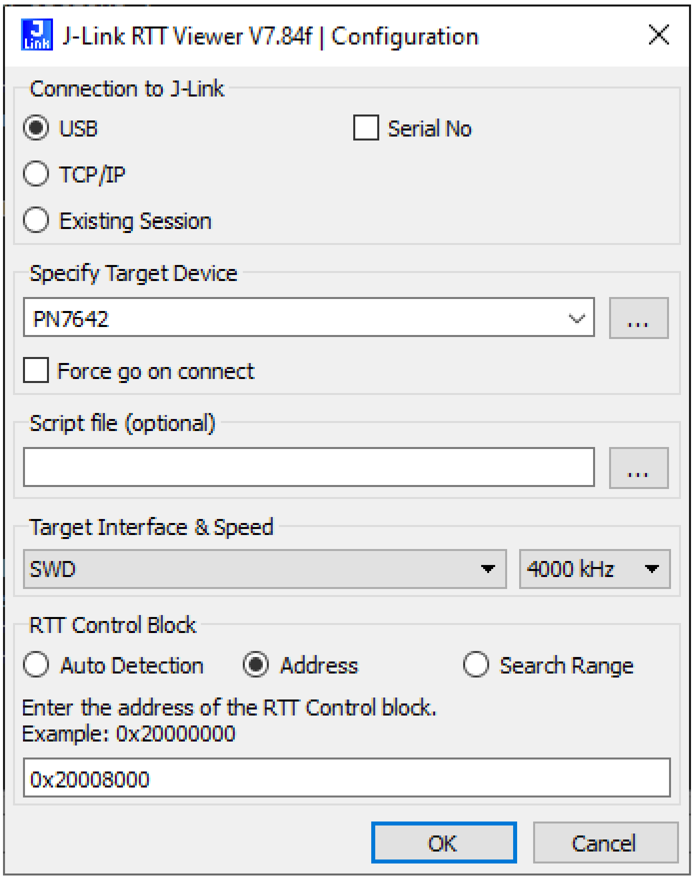


Figure 30. J-Link RTT Viewer configuration

The address of the RTT control block is: **0x20008000**

7 Tips, tricks, and FAQ

Q: What can I do if the chip is cycling through ULP Standby and I cannot connect with a debugger anymore?

A: Usually this happens if your application has no reason to stay active and directly goes into ULP Standby again. The easiest solution, on the development board, would be to bring the chip into USB Mass-Storage and replace the application with some other, which is not using any low-power modes (for example, LED Blinky).

Alternatively you can try to catch the chip in its active state. Hold reset, then release the reset and press connect of the debugger at the same time.

Q: I cannot use the J-Link RTT viewer because I do not have a SEGGER J-Link, what can I do?

A: If you have a MCU-Link Pro or LPC-Link2 instead, you can flash J-Link Lite on them. With J-Link Lite, you can use the RTT viewer. Another option can be to use semihost and indicate the boot reason with LEDs.

Q: How to visualize the RF-Field?

A: You can use the probe of the oscilloscope and connect the Ground clamp to the tip of the probe itself. By placing this loop on the antenna, you should be able to trigger easily on a rising edge to capture the RF-Field. Depending on your chosen interval time, I would adjust the time axis to your chosen interval x 4.

Q: How to retain data over ULPCD?

The PN76 has no RAM section that is not deleted during ULP/ULPCD. This decision has been made to minimize the current consumption as much as possible. The ALV (always live) RAM section is also turned off in these ultra-low-power modes. Although it can be used for other standby and sleep modes.

The PN7642 flash and EEPROM write endurance are minimum 100k cycles. As long as your code writes infrequently and you can ensure that you do not exceed 100k writes, EEPROM, and flash should be acceptable.

Q: Why does the ULPCD example not wake-up from an external RF-Field?

The example calls the EnterULPCD API per default without enabling the RFLD option. To change this go to the method "ULPCD_Demo" and change the third parameter to "1".

```
static PN76_Status_t ULPCD_Demo(uint16_t wLPCDWakeUpTime)
{
    phStatus_t wStatus;

    /* Perform ULPCD Detection Cycle */
    wStatus = PN76_Sys_PCRM_EnterULPCD(E_PN76_PCRM_ULPCD_CARD_DETECTION, wLPCDWakeUpTime, 1);
    PN76_CHECK_SUCCESS(wStatus);
    /* Control never reaches here, adding for warning removal */
    return  PN76_STATUS_SUCCESS;
}
```

Figure 31. Enable RFLD at the ULPCD example

Q: Why do I see ≈ 5.6 ms glitches during ULPCD on GPIOs 0-3?

PN7642 contains a state latch that holds the last state of an I/O pins (including GPIO 0-3) in ULP Standby mode. However, during the active phase of **ULPCD**, specifically during the **RF Ping** operation, there exists a timing window of approximately **5.6 milliseconds** in which the I/O pin states are exclusively controlled by ULPCD hardware and a configuration shown in Table 4 is applied to these I/O pins. User modifications to I/O pin states are not permitted during this period.

If the customer design includes external pull-up resistors such as those shown on GPIO0 and GPIO2 in the figure. This configuration forms a voltage divider with the internal 50 kΩ pull-down resistors. As a result, a voltage drop may be observed on these GPIOs during the hardware-controlled period. Customers are required take this into account during the product design-in phase.

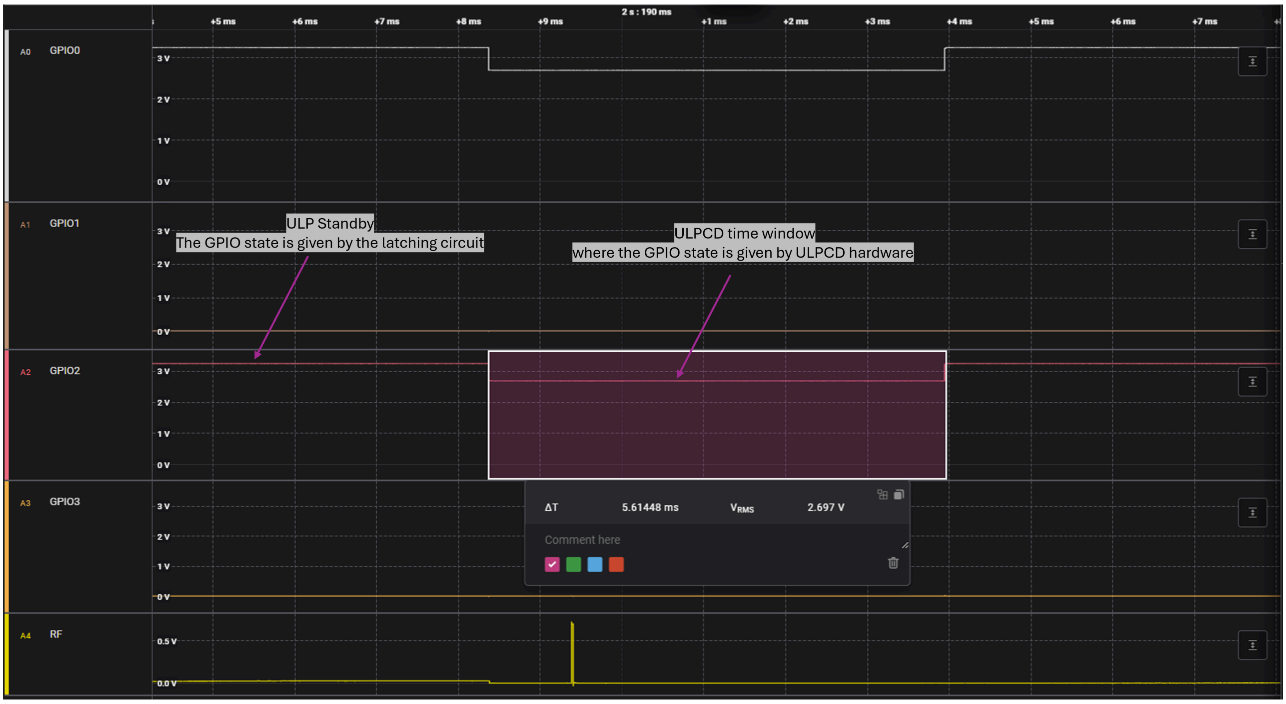


Figure 32. GPIO states in ULPCD

Table 4. State of I/O pins in the ULPCD hardware-controlled period

I/O pin name	I/O State
AUX3	HiZ
AUX2	HiZ
AUX1	HiZ
ATX_D	Pullup
ATX_C	Pull-up
ATX_B	HiZ
ATX_A	HiZ
GPIO0-3	Pull-down (50 kΩ)
GPIO4-5	HiZ
IRQ	Pull-down (50 kΩ)

8 Abbreviations

Table 5. Abbreviations

Acronym	Description
AO_RAM	Always On RAM
GPREG	General Purpose Register
LPCD	Low-power card detection
NFC	Near Field Communication
PCRM	Power clock reset management
RFLD	Radio frequency level detector
SDK	Software development kit
ULFO	Ultra low-frequency oscillator
ULP	Ultra low-power
ULPCD	Ultra low-power card detection

9 References

[1] Data sheet – PN7642 – Single chip solution with high performance NFC reader, customizable MCU and security toolbox (link)
[2] User manual – UM11566 – PN76 family NFC open controller (link)
[3] Resource – PN7642 NFC controller user API documentation (part of the MCUXpresso SDK link)
[4] Application note – AN14138 – PN7642 design-in recommendations (link)
[5] Software – PN7642 MCUXpresso SDK (link)
[6] Application note – AN13134 – PN76 family evaluation board quick start guide (link)
[7] Release Note – RN00257 – PN7642 firmware release note (link)
[8] Application note – AN14518 – Crystal Oscillator Design Guide (link)
[9] Errata information – ES_PN7642 – Errata information PN7642 (link)

10 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2023-2025 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

11 Revision history

Table 6. Revision history

Document ID	Release date	Description
AN13996 v.5.0	18 November 2025	Editorial changes. • Section 3.2 "GPADC_ERROR (0x001E)": added. • Section 7 "Tips, tricks, and FAQ": updated. • Section 9 "References": updated.
AN13996 v.4.0	27 May 2025	Editorial changes. • Section 1.1 "Environment": updated. • Section 4 "EEPROM read/writes and IC configuration": updated. • Section 4.1 "Power configuration" added. • Section 4.2 "ULPCD HFATT value" added. • Section 4.3 "RFLD and NFCLD calibration" added.
AN13996 v.3.0	18 February 2025	Editorial and structural changes. • Section 1.1 "Environment": updated. • Section 2.1 "Low-power card detection (LPCD)": added. • Section 2.2 "Standby": added. • Section 2.3.2 "Wake-up reasons in ULPCD": updated. • Section 3.1 "GPADC_READY_TIMEOUT": added. • Section 4 "EEPROM read/writes and IC configuration": updated. • Section 5.1 "Option 1+2: Enter Standby": added. • Section 5.4 "Option 5: ULPCD Calibration": updated. • Section 5.5 "Option 6: ULPCD Detection": added. • Section 5.6 "Option 7: LPCD Calibration": added. • Section 5.7 "Option 8: LPCD Detection": added. • Section 9 "References": updated. • Section 10 "Note about the source code in the document ": updated.
AN13996 v.2.0	14 October 2024	Editorial changes. • Section 1 "Introduction": updated. • Section 2 "Low-power modes": updated. • Section 2.3 "Ultra low-power card detection (ULPCD)": updated. • Section 2.3.1 "ULPCD API": updated. • Section 2.4.1 "ULP Standby API": updated. • Wake-up reasons: added. • Section 4 "EEPROM read/writes and IC configuration": added. • Section 7 "Tips, tricks, and FAQ": updated. • Section 8 "Abbreviations": added. • Section 9 "References": updated.
AN13996 v.1.0	05 July 2023	• Initial version.

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Licenses

Purchase of NXP ICs with NFC technology — Purchase of an NXP Semiconductors IC that complies with one of the Near Field Communication (NFC) standards ISO/IEC 18092 and ISO/IEC 21481 does not convey an implied license under any patent right infringed by implementation of any of those standards. Purchase of NXP Semiconductors IC does not include a license to any NXP patent (or other IP right) covering combinations of those products with other products, whether hardware or software.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

EdgeVerse — is a trademark of NXP B.V.

J-Link — is a trademark of SEGGER Microcontroller GmbH.

SEGGER Embedded Studio — is a trademark of SEGGER Microcontroller GmbH.

Tables

Figures

Contents