

Development Kit content and key features

Contents

1. [Interface MCU](#)
 - 1.1. [IF Boot/Reset button](#)
 - 1.2. [Virtual COM port](#)
 - 1.3. [Interface MCU firmware \(FW\)](#)
2. [Hardware description](#)
 - 2.1. [Hardware drawings](#)
 - 2.2. [Block diagram](#)
 - 2.3. [Power supply](#)
 - 2.4. [Connector interface](#)
 - 2.5. [Buttons and LEDs](#)
 - 2.5.1. [I/O expander for buttons and LEDs](#)
 - 2.6. [32.768 kHz crystal](#)
 - 2.7. [Measuring current](#)
 - 2.8. [RF measurements](#)
 - 2.9. [Debug input](#)
 - 2.10. [Debug output](#)
 - 2.11. [NFC antenna interface](#)

Development Kit content and key features

In addition to hardware, the nRF52 Development Kit consists of firmware source code, documentation, hardware schematics, and layout files.

The key features of the development kit are:

- nRF52832 flash-based ANT/ANT+, *Bluetooth*® low energy SoC solution
- Buttons and LEDs for user interaction
- I/O interface for Arduino form factor plug-in modules
- SEGGER J-Link OB Debugger with debug out functionality
- Virtual COM Port interface via UART
- Drag-and-drop Mass Storage Device (MSD) programming
- Supporting NFC-A listen mode

For access to firmware source code, hardware schematics, and layout files, see www.nordicsemi.com.

Figure 1. 1 × nRF52 Development Kit board (PCA10040) and 1 x NFC adhesive tag

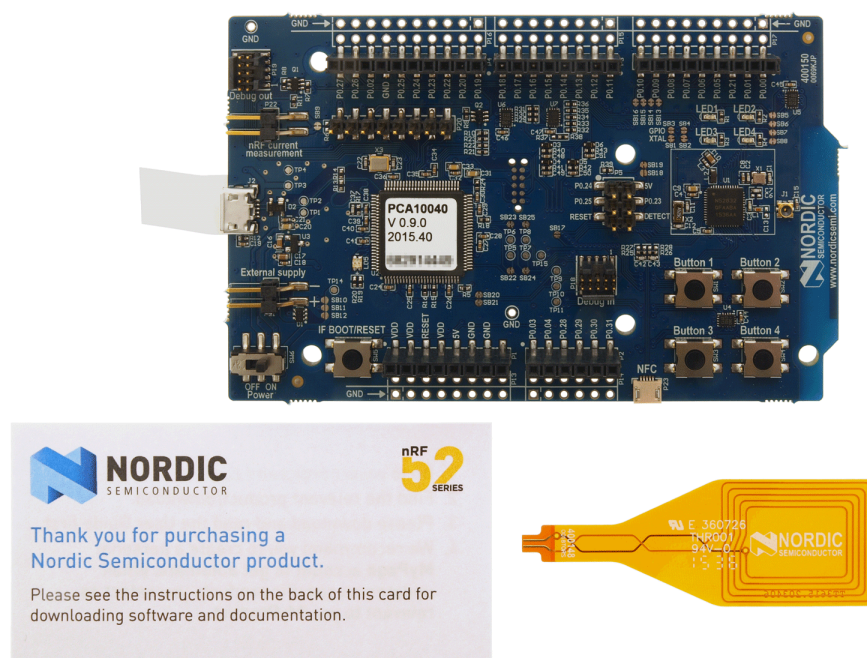



Figure 1. 1 × nRF52 Development Kit board (PCA10040) and 1 x NFC adhesive tag



Environmental Protection

Waste electrical products should not be disposed of with household waste.

Please recycle where facilities exist. Check with your local authority or retailer for recycling advice.

Interface MCU

The Interface MCU is used to control the firmware on the nRF52832 IC by the on-board SEGGER J-Link.

Hardware description

The nRF52 Development Kit board (PCA10040) can be used as a development platform for the nRF52832 device. It features an onboard programming and debugging solution.

Parent topic: [Get started](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

1. Interface MCU

The Interface MCU is used to control the firmware on the nRF52832 IC by the on-board SEGGER J-Link.

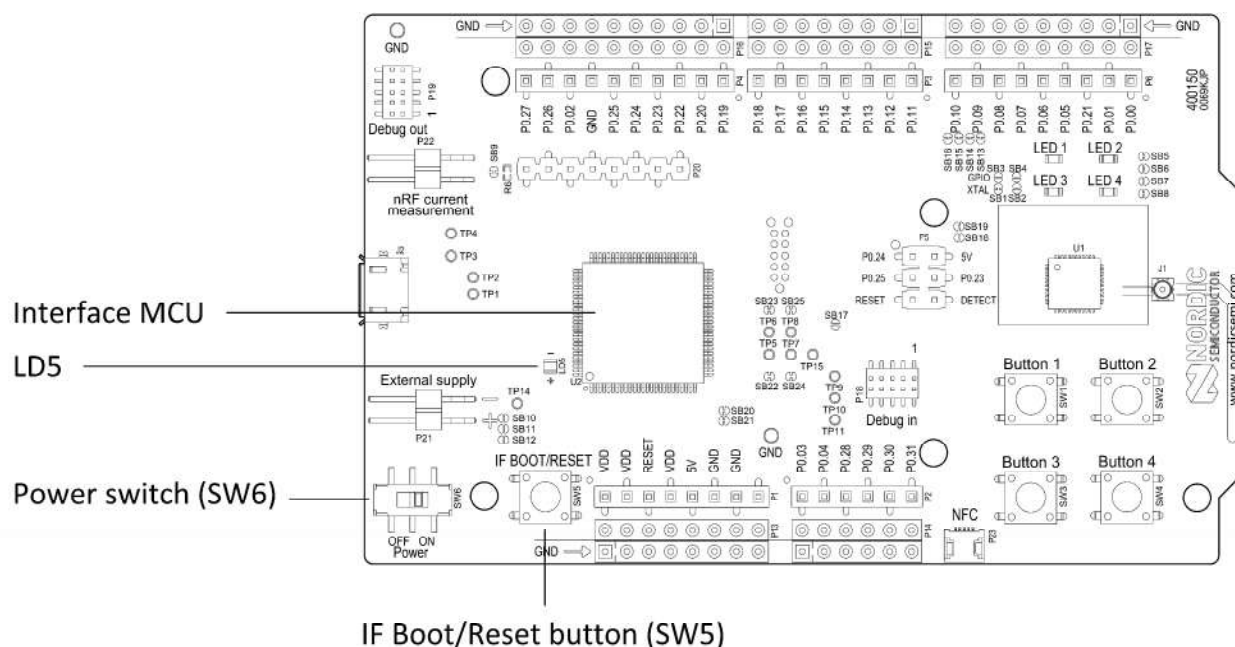


Figure 1. Interface MCU

IF Boot/Reset button

The nRF52 Development Kit board is equipped with a boot/reset button (**SW5**).

Virtual COM port

The on-board Interface MCU features a Virtual COM port via UART.

Interface MCU firmware (FW)

The on-board interface MCU is factory programmed with an mbed-compliant bootloader. This makes it possible to update the SEGGER J-Link OB interface firmware.

Parent topic: [Development Kit content and key features](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

1.1. IF Boot/Reset button

The nRF52 Development Kit board is equipped with a boot/reset button (**SW5**).

This button is connected to the Interface MCU on the board and has two functions:

- Resetting the nRF52832 device.
- Entering bootloader mode of the interface MCU.

During normal operation the button will function as a reset button for the nRF52832 device. **For this to work, pin reset on P0.21 needs to be enabled for the nRF52832 device.** The button is also used to enter the bootloader

mode of the Interface MCU. **To enter bootloader mode, keep the reset button pressed while powering up the board until LED (LD5) starts to blink.** You can power up the board either by disconnecting and reconnecting the USB cable, or toggle the power switch (SW6).

Important: Pin reset can be enabled by defining the CONFIG_GPIO_AS_PINRESET variable in the project settings. This can be done by defining the preprocessor symbol in Keil, go to: **Project > Options for Target > C/C++ > Preprocessor Symbols > Define**. Here you can add the CONFIG_GPIO_AS_PINRESET variable after NRF52.

This functionality can be removed by doing a nRFjprog --recover.

Parent topic: [Interface MCU](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

1.2. Virtual COM port

The on-board Interface MCU features a Virtual COM port via UART.

The virtual COM port has the following features:

- Flexible baudrate setting up to 1 Mbps.
- Dynamic Hardware Flow Control (HWFC) handling.
- Tri-stated UART lines when no terminal is connected.

[Table 1](#) shows an overview of the UART connections on nRF52832 and the interface MCU.

Default GPIO nRF52832	UART nRF52832	Interface MCU UART
P0.05	RTS	CTS
P0.06	TXD	RXD
P0.07	CTS	RTS
P0.08	RXD	TXD

Table 1. Relationship of UART connections on nRF52832 and Interface MCU

The UART signals are routed directly to the Interface MCU. The UART pins connected to the Interface MCU are tri-stated when no terminal is connected to the Virtual COM port on the computer.

Important: The terminal used must send a DTR signal in order to configure the UART Interface MCU pins.

The P0.05 (RTS) and P0.07 (CTS) can be used freely when HWFC is disabled on the nRF52832.

Parent topic: [Interface MCU](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

1.3. Interface MCU firmware (FW)

The on-board interface MCU is factory programmed with an mbed-compliant bootloader. This makes it possible to update the SEGGER J-Link OB interface firmware.

For more information on entering the bootloader mode, see [IF Boot/Reset button](#).

To update Interface MCU firmware, drag the Interface image (.bin) into the mounted bootloader drive on the connected computer and power cycle the board. It is also possible to download the latest SEGGER J-Link software from [SEGGER J-Link Software](#) and open a debug session in order to automatically update to the latest J-Link OB firmware version.

The J-Link OB image can be downloaded from www.nordicsemi.com.

Parent topic: [Interface MCU](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2. Hardware description

The nRF52 Development Kit board (PCA10040) can be used as a development platform for the nRF52832 device. It features an onboard programming and debugging solution.

In addition to radio communication, the nRF52832 device can communicate with a computer through a virtual COM port provided by the interface MCU.

[Hardware drawings](#)

nRF52 Development Kit hardware drawings show both sides of the PCA10040 board.

[Block diagram](#)

The nRF52 Development Kit board block diagram shows the connections between the different blocks.

[Power supply](#)

The nRF52 Development Kit board has three power options: 5 V from the USB, external power supply, and coin cell battery.

[Connector interface](#)

Access to the nRF52832 GPIOs is available from connectors **P2**, **P3**, **P4**, **P5**, and **P6**. The **P1** connector provides access to ground and power on the nRF52 Development Kit board.

[Buttons and LEDs](#)

The four buttons and four LEDs on the nRF52 Development Kit board are connected to dedicated I/Os on the nRF52832 chip.

[32.768 kHz crystal](#)

nRF52832 can use an optional 32.768 kHz crystal (X2) for higher accuracy and lower average power consumption.

[Measuring current](#)

The current drawn by the nRF52832 device can be monitored on the nRF52 Development Kit board.

RF measurements

The nRF52 Development Kit board is equipped with a small size coaxial connector (**J1**) for conducted measurements of the RF signal.

Debug input

The Debug in connector (**P18**) makes it possible to connect external debuggers for debugging while running on battery or external power supply.

Debug output

The nRF52 Development Kit board supports programming and debugging external boards. To debug an external board, connect to the Debug out connector (**P19**) with a 10 pin cable.

NFC antenna interface

The nRF52 Development Kit board supports a Near Field Communication (NFC) tag.

Parent topic: [Development Kit content and key features](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.1. Hardware drawings

nRF52 Development Kit hardware drawings show both sides of the PCA10040 board.

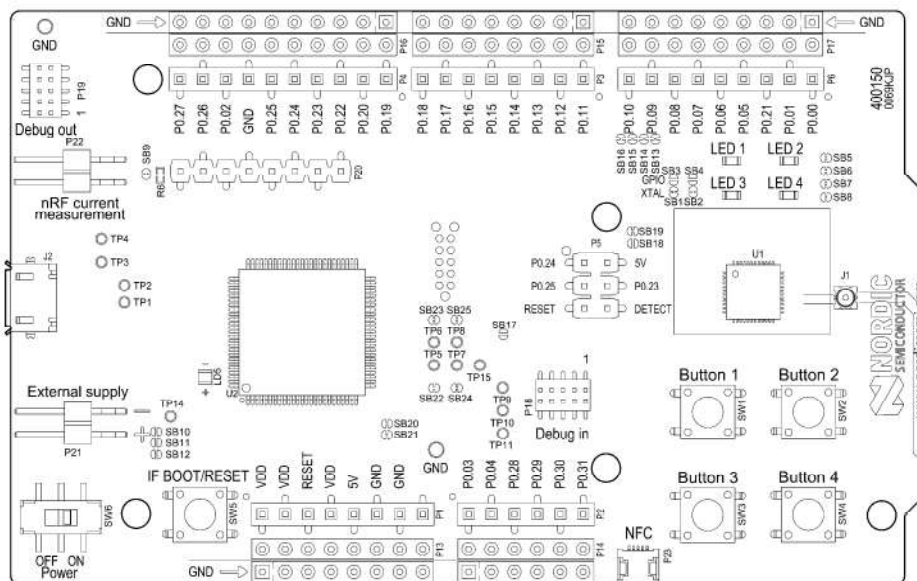


Figure 1. nRF52 Development Kit board top view

Figure 2. nRF52 Development Kit board bottom view

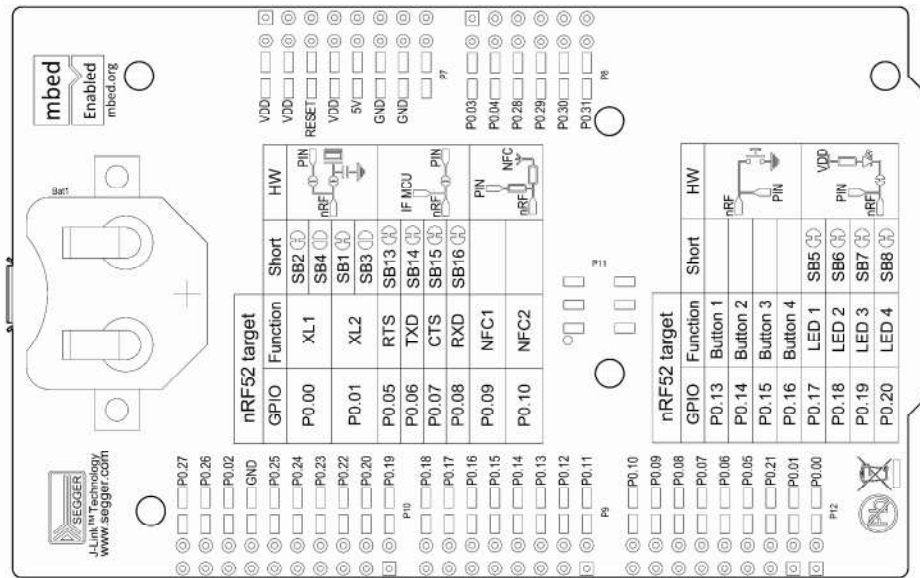


Figure 2. nRF52 Development Kit board bottom view

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.2. Block diagram

The nRF52 Development Kit board block diagram shows the connections between the different blocks.

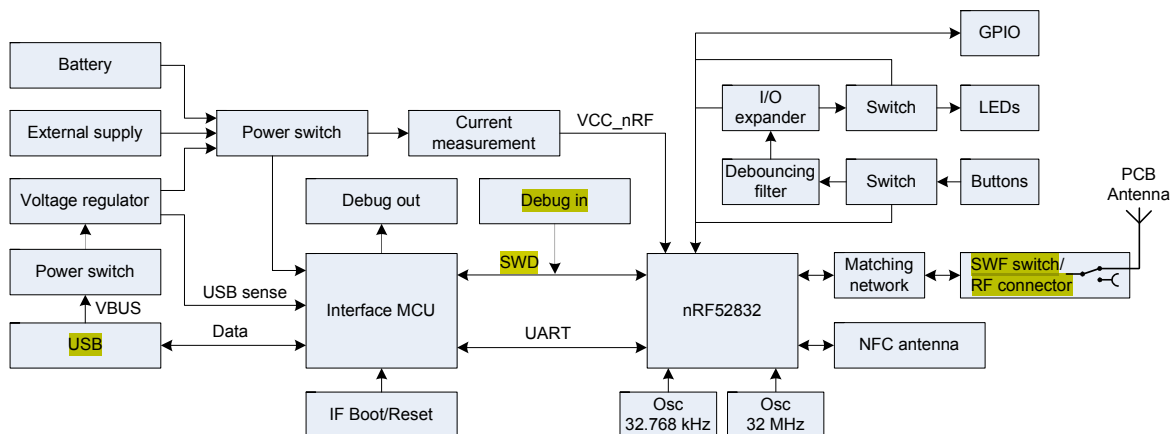


Figure 1. nRF52 Development Kit board block diagram

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.3. Power supply

The nRF52 Development Kit board has **three power options**: 5 V from the **USB**, **external** power supply, and coin cell battery.

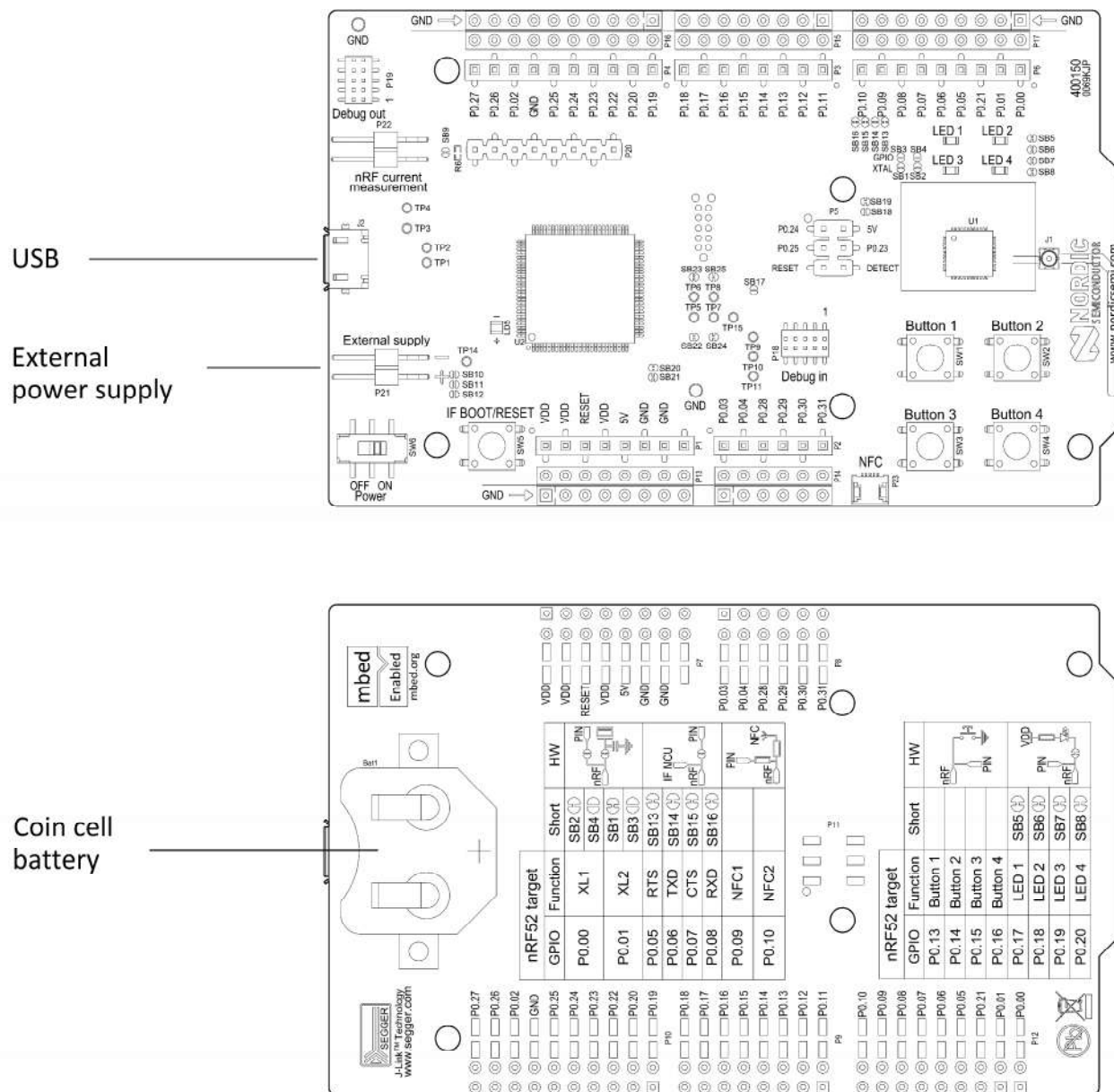


Figure 1. Power supply options

The 5 V from the USB is regulated down to **3.3 V** through an on-board voltage regulator. **The battery and external power supply are not regulated.** The power sources are routed through a set of diodes (**D1A**, **D1B**, and **D1C**) for reverse voltage protection, where the circuit is supplied from the source with the highest voltage.

Important: When USB is not powered, the Interface MCU is in dormant state and will draw an additional current of $\sim 20 \mu\text{A}$ in order to maintain the reset button functionality. This will affect board current consumption, but not the nRF52832 current measurements, as described in [Measuring current](#).

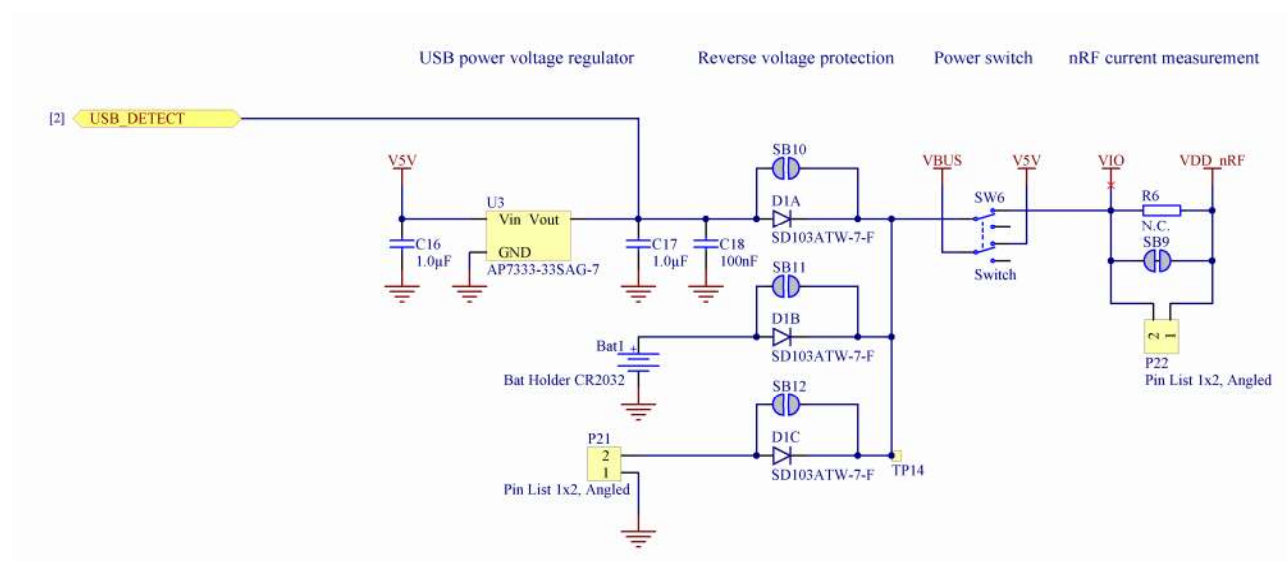


Figure 2. Power supply circuitry

The reverse voltage protection diodes will add a voltage drop to the supply voltage of the circuit. To avoid this voltage drop, **the diodes can be bypassed by shorting one or more solder bridges.**

Power source	Protection bypass	Voltage level
USB	SB10	3.3 V
Coin-cell battery	SB11	Battery
External supply	SB12	1.7 V - 3.6 V

Table 1. Protection diode bypass solder bridges

Figure 3. Protection diode bypass solder bridges

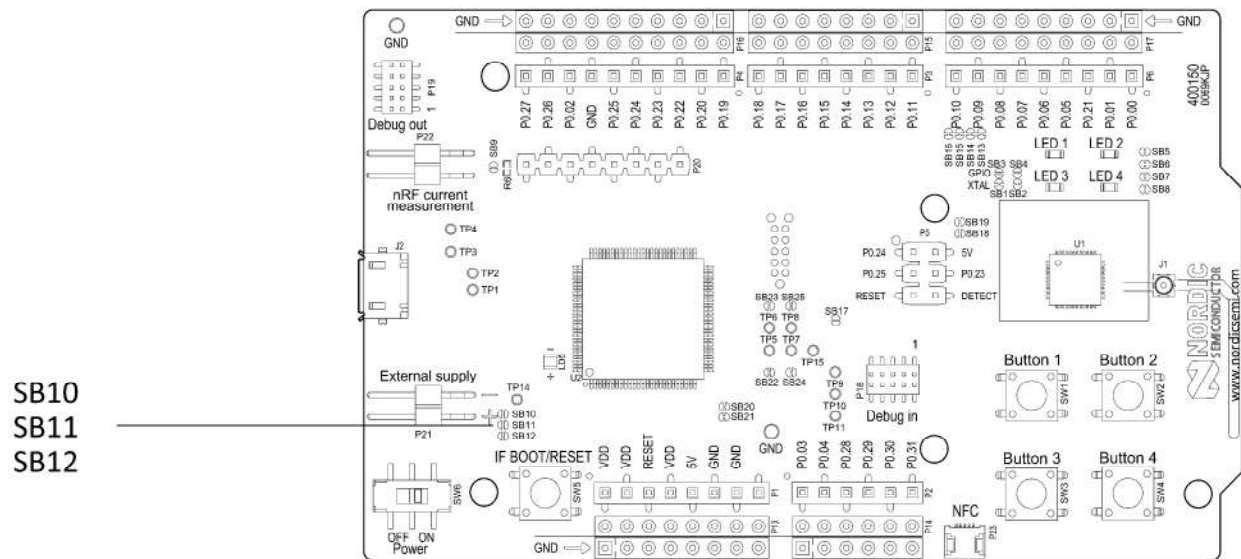


Figure 3. Protection diode bypass solder bridges

Important: Connect only one power source at the time. Shorting the solder bridges removes the reverse voltage protection.

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.4. Connector interface

Access to the nRF52832 GPIOs is available from connectors P2, P3, P4, P5, and P6. The P1 connector provides access to ground and power on the nRF52 Development Kit board.

Figure 1. nRF52 Development Kit board connectors

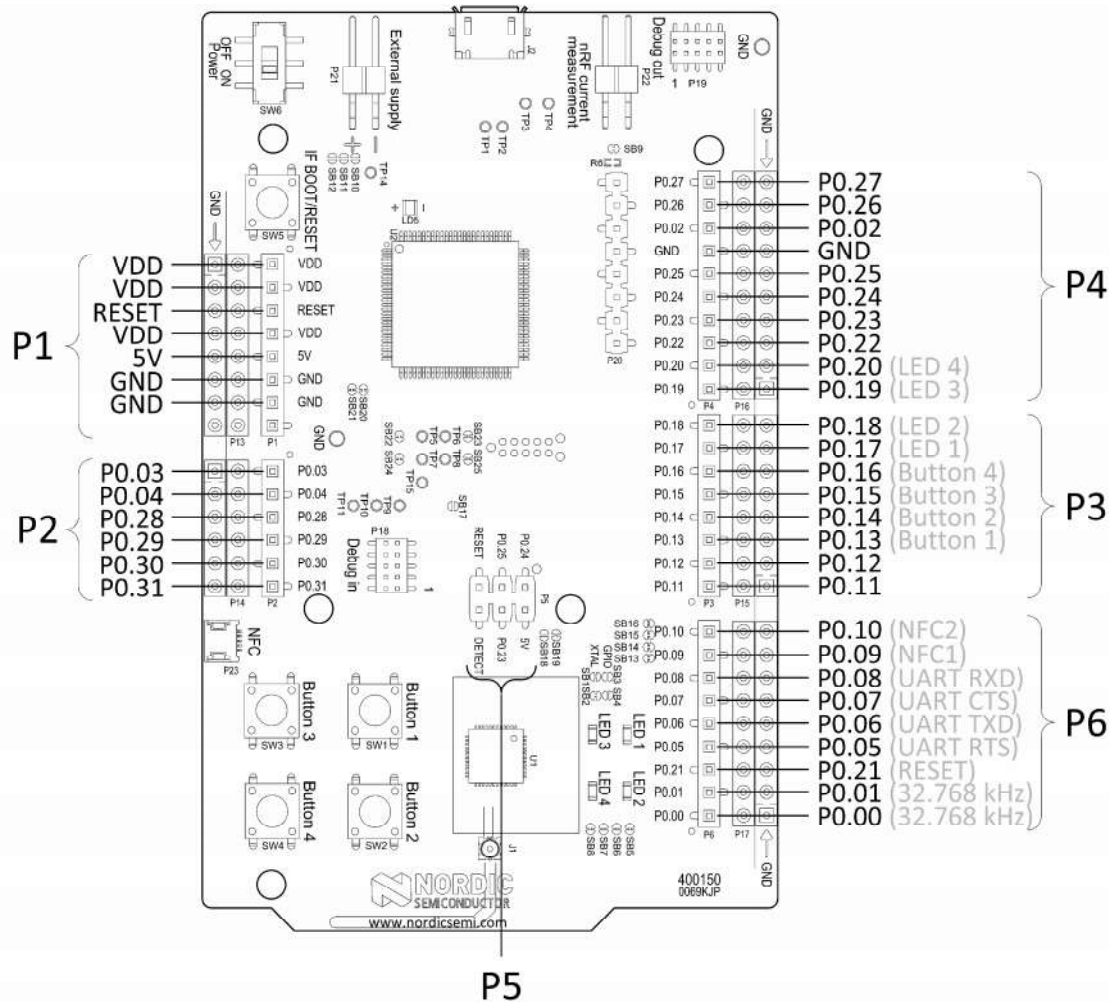


Figure 1. nRF52 Development Kit board connectors

The signals are also available on connectors **P7**, **P8**, **P9**, **P10**, **P11**, and **P12**, which are on the bottom side of the board. By mounting pin lists on the connector footprints, the nRF52 Development Kit board can be used as a shield for Arduino motherboards or other boards that follow the Arduino standard.

For easy access to GPIO, power, and ground, the signals can also be found on the through-hole connectors **P13**–**P17**.

Important: Some pins have default settings.

- P0.00 and P0.01 are by default used for the 32 kHz crystal and are not available on the connectors. For more information, see Section [32.768 kHz crystal](#).
- P0.05, P0.06, P0.07, and P0.08 are by default used by the UART connected to the Interface MCU. For more information, see Section [Virtual COM port](#).
- P0.09 and P0.10 are by default used by NFC1 and NFC2. For more information, see Section [NFC antenna interface](#).
- P0.13–P0.20 are by default connected to the buttons and LEDs. For more information, see Section [Buttons and LEDs](#).

When the nRF52 Development Kit board is used as a shield together with an Arduino standard motherboard, the Arduino signals are routed as shown in [Figure 2](#).

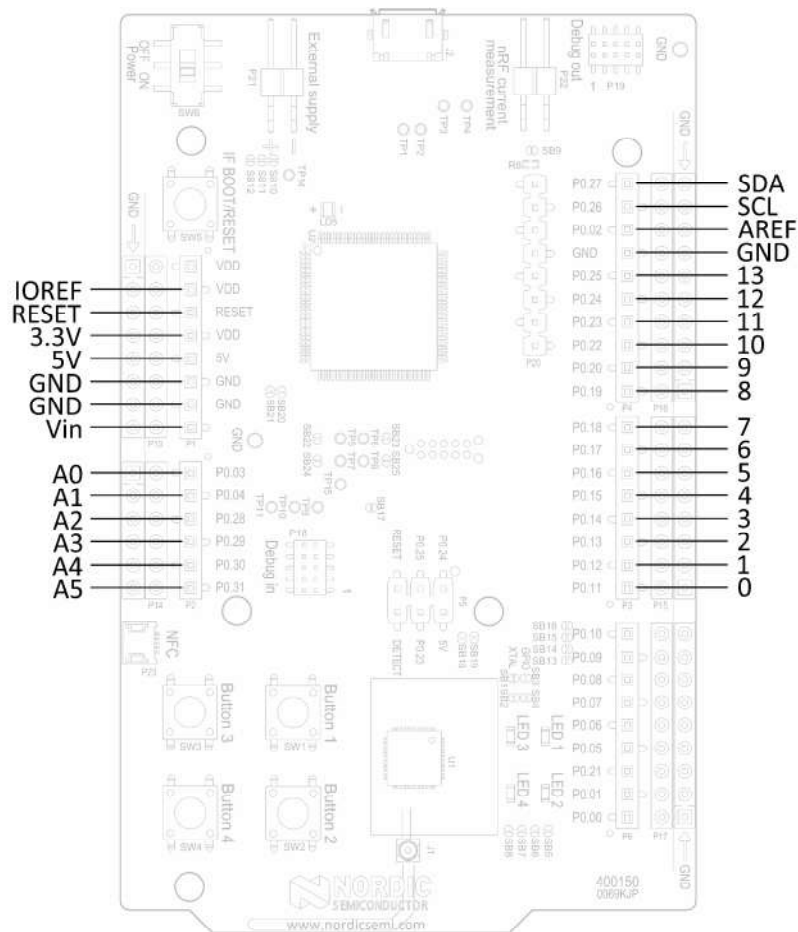


Figure 2. Arduino signals routing on the nRF52 Development Kit board

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.5. Buttons and LEDs

The four buttons and four LEDs on the nRF52 Development Kit board are connected to dedicated I/Os on the nRF52832 chip.

Part	GPIO	Short
Button 1	P0.13	-
Button 2	P0.14	-
Button 3	P0.15	-

Part	GPIO	Short
Button 4	P0.16	-
LED 1	P0.17	SB5
LED 2	P0.18	SB6
LED 3	P0.19	SB7
LED 4	P0.20	SB8

Table 1. Button and LED connection

If GPIO P0.17–P0.20 are needed elsewhere, the LEDs can be disconnected by cutting the short on **SB5–SB8**, see [Figure 1](#). The LEDs and buttons can also be disconnected by using the I/O extender as described in [I/O expander for buttons and LEDs](#).

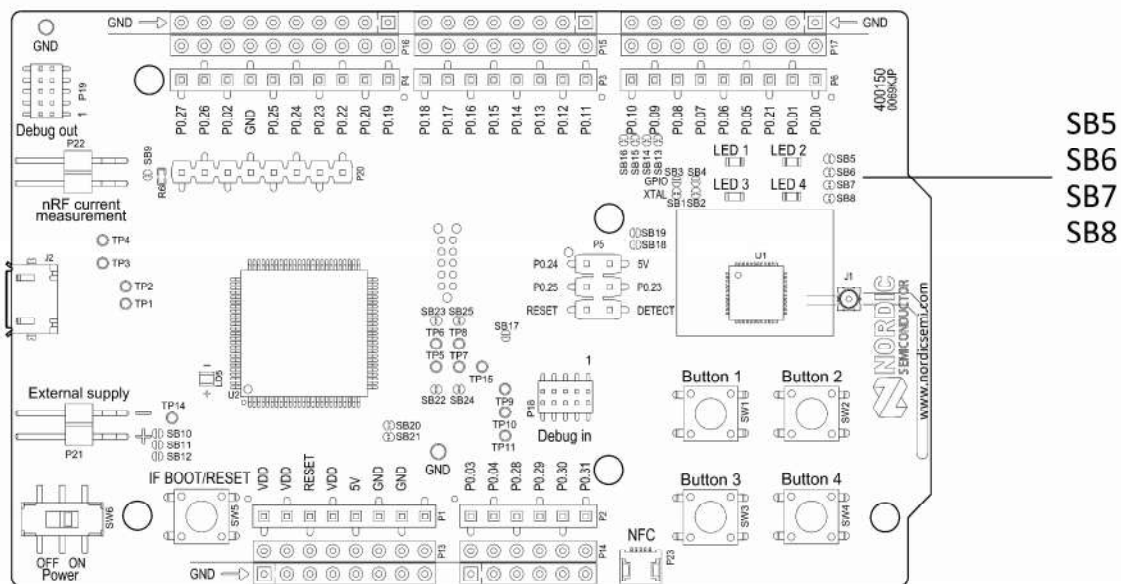


Figure 1. Disconnecting the LEDs

The buttons are active low, meaning the input will be connected to ground when the button is activated. The buttons have no external pull-up resistor, so to use the buttons the P0.13–P0.16 pins must be configured as an input with an internal pull-up resistor.

The LEDs are active low, meaning that writing a logical zero ('0') to the output pin will illuminate the LED.

Figure 2. Button and LED configuration

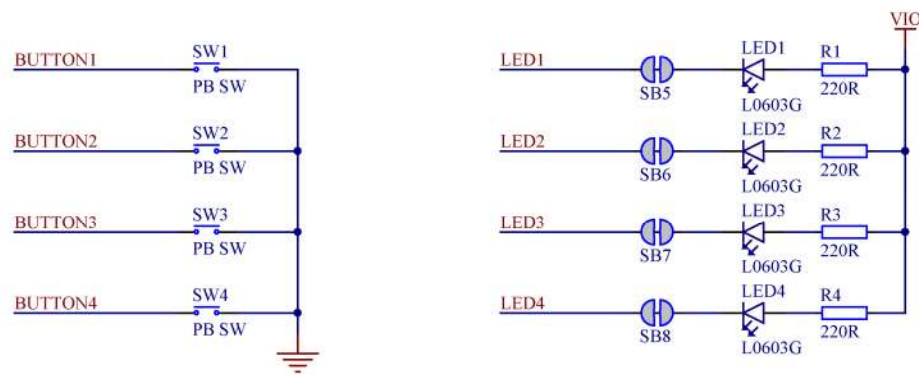


Figure 2. Button and LED configuration

I/O expander for buttons and LEDs

The nRF52 Development Kit board has an I/O expander to avoid conflicts with boards that follow the Arduino standard, the on-board GPIOs for the buttons and LEDs would otherwise possibly conflict with such boards.

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.5.1. I/O expander for buttons and LEDs

The nRF52 Development Kit board has an I/O expander to avoid conflicts with boards that follow the Arduino standard, the on-board GPIOs for the buttons and LEDs would otherwise possibly conflict with such boards.

GPIO	Part	Arduino signal
P0.13	Button 1	2
P0.14	Button 2	3
P0.15	Button 3	4
P0.16	Button 4	5
P0.17	LED 1	6
P0.18	LED 2	7
P0.19	LED 3	8
P0.20	LED 4	9

Table 1. GPIO connection

The I/O expander will release these GPIOs for general use when the nRF52 Development Kit is used together with boards that follows the Arduino standard. The I/O expander can be permanently enabled by shorting solder bridge

SB18 or permanently disabled by cutting the shorting track on **SB19**. You must also short **SB18** when cutting **SB19** for full compatibility with the Arduino standard.

The I/O extender can be temporarily enabled by connecting SHIELD DETECT to ground.

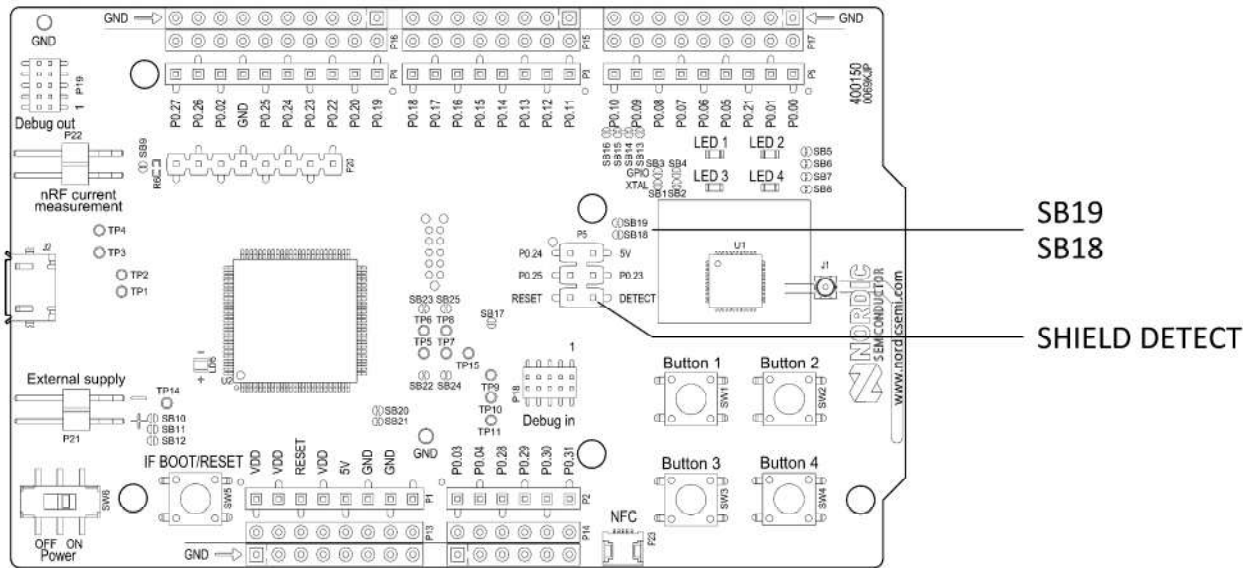


Figure 1. Enable or disable I/Os for Arduino standard

In addition to the buttons and LEDs, the following GPIOs are used for the I/O expander:

I/O expander signal	GPIO
/INT	P0.17
SDA	P0.26
SCL	P0.27

Table 2. I/O expander connection

Figure 2. I/O expander schematic

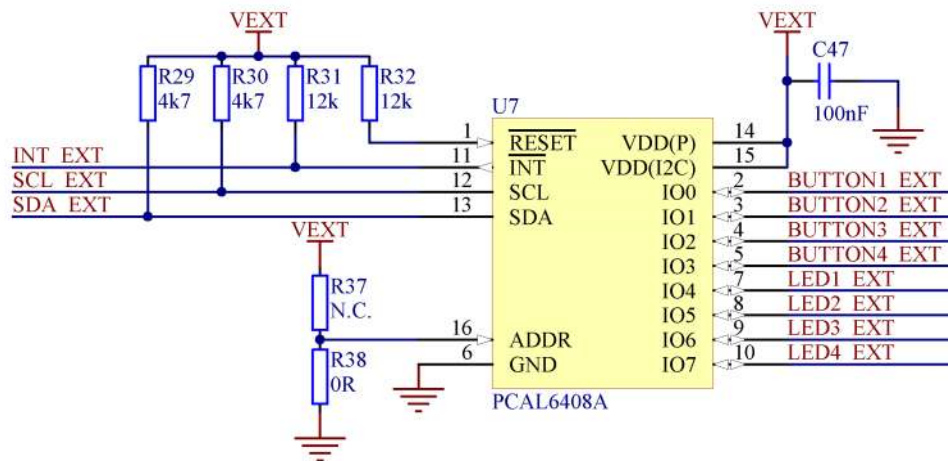


Figure 2. I/O expander schematic

Important: SW debouncing should not be needed when using the I/O expander. Each button on the nRF52 Development Kit board is equipped with a debouncing filter.

Parent topic: [Buttons and LEDs](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.6. 32.768 kHz crystal

nRF52832 can use an optional 32.768 kHz crystal (X2) for higher accuracy and lower average power consumption.

On the nRF52 Development Kit board, P0.00 and P0.01 are by default used for the 32.768 kHz crystal and are not available as a GPIO on the connectors.

Important: When using ANT/ANT+, the 32.768 kHz crystal (X2) is required for correct operation.

If P0.00 and P0.01 are needed as normal I/Os, the 32.768 kHz crystal can be disconnected and the GPIO routed to the connectors. Cut the shorting track on **SB1** and **SB2**, and solder **SB3** and **SB4**. See [Figure 1](#) for reference.

Figure 1. Configuring P0.00 and P0.01

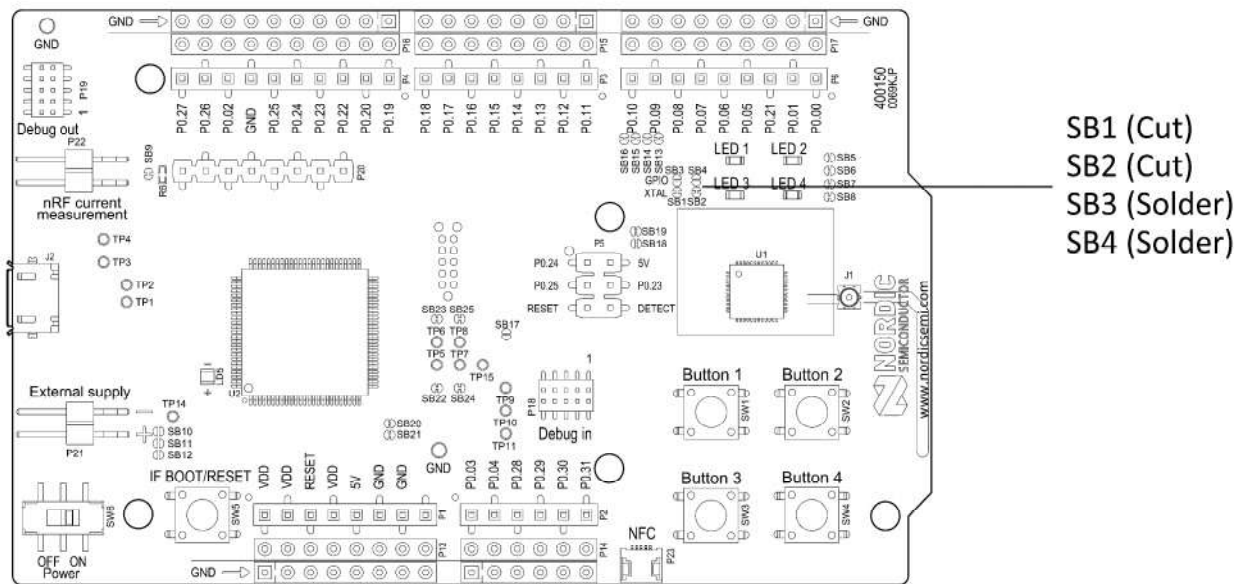


Figure 1. Configuring P0.00 and P0.01

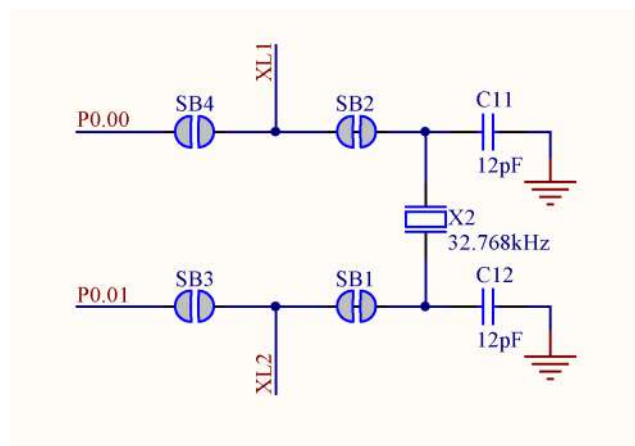


Figure 2. 32.768 kHz crystal and SB1 to SB4 schematic

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.7. Measuring current

The current drawn by the nRF52832 device can be monitored on the nRF52 Development Kit board.

To measure the current, you must first prepare the board by cutting the shorting of solder bridge **SB9**. There are two ways of measuring the current consumption: using an ampere-meter or an oscilloscope.

1. Ampere-meter:

- a. Connect an ampere-meter between the pins of connector **P22**. This will monitor the current directly.

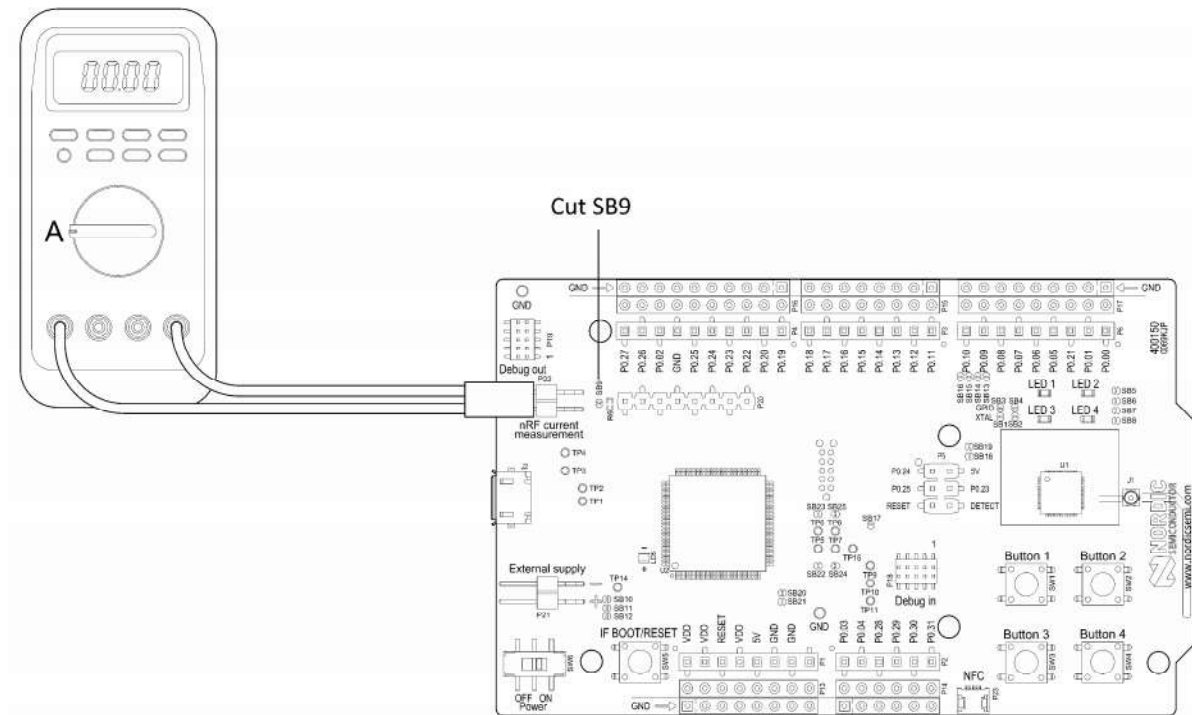


Figure 1. Current measurement with ampere-meter

2. Oscilloscope:

- a. Mount a resistor on the footprint for **R6**. The resistor should not be larger than 10 Ω .
- b. Connect an oscilloscope in differential mode or similar with two probes on the pins of the **P22** connector.
- c. Measure the voltage drop. The voltage drop will be proportional to the current consumption. For example, if a 10 Ω resistor is chosen, 10 mV equals 1 mA.

Figure 2. Current measurement with oscilloscope

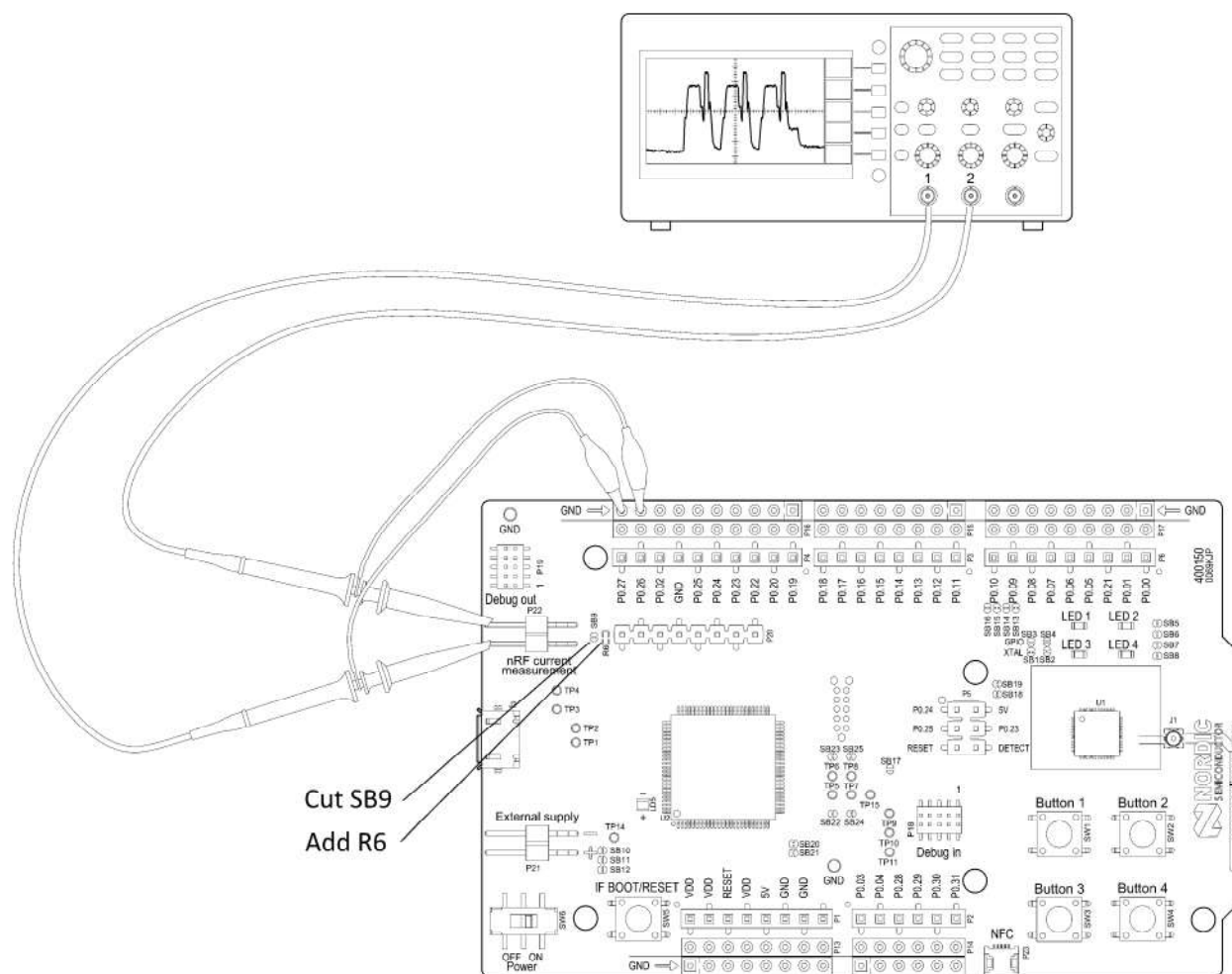


Figure 2. Current measurement with oscilloscope

Important: The current measurements will become unreliable when a serial terminal is connected to the Virtual COM port.

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.8. RF measurements

The nRF52 Development Kit board is equipped with a small size coaxial connector (**J1**) for conducted measurements of the RF signal.

The connector is of SWF type from Murata (part no. MM8130-2600) with an internal switch. By default, when there is no cable attached, the RF signal is routed to the on-board PCB trace antenna.

A test probe is available from Murata (part no. MXHS83QE3000) with a standard SMA connection on the other end for connecting instruments. When connecting the test probe, the internal switch in the SWF connector will disconnect the PCB antenna and connect the RF signal from the nRF52832 device to the test probe.

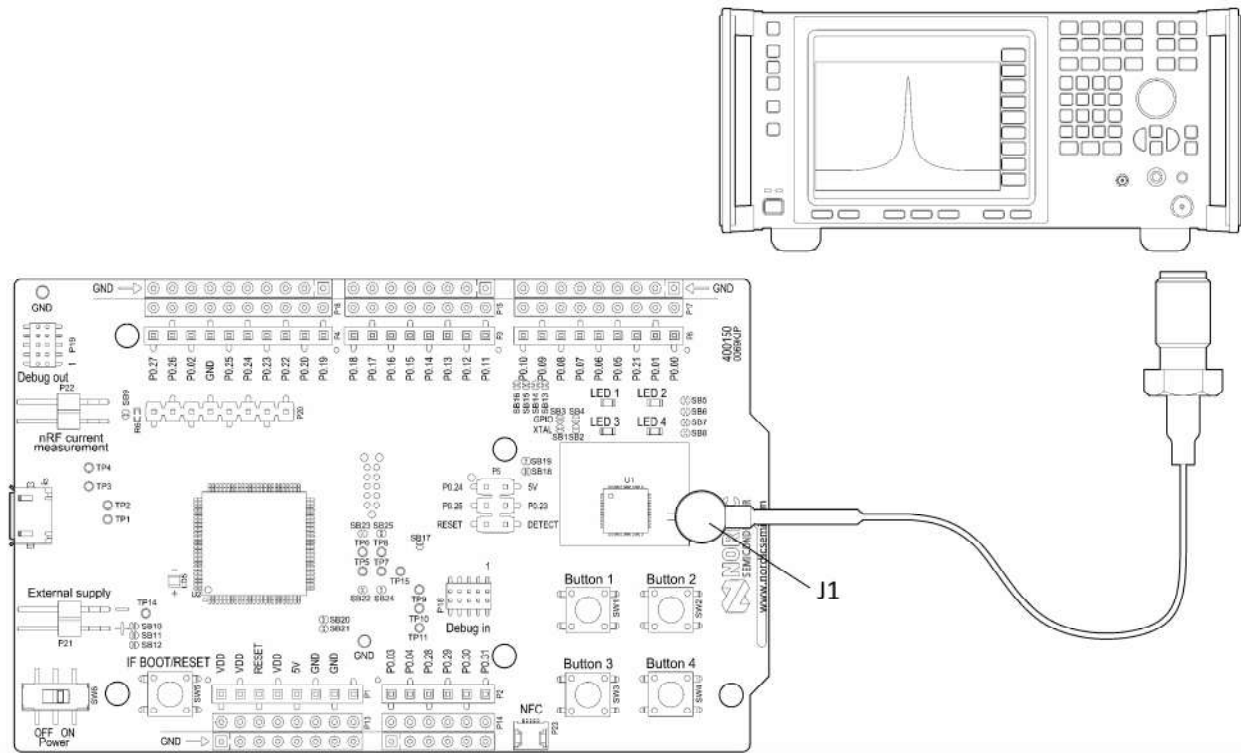


Figure 1. Connecting a spectrum analyzer

The connector and test probe will add loss to the RF signal which should be taken into account when doing measurements, see [Table 1](#).

Frequency (MHz)	Loss (dB)
2440	1.0
4880	1.7
7320	2.6

Table 1. Typical loss in connector and test probe

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.9. Debug input

The Debug in connector (P18) makes it possible to connect external debuggers for debugging while running on battery or external power supply.

外部调试器，调试PCA10040上面的nRF52832

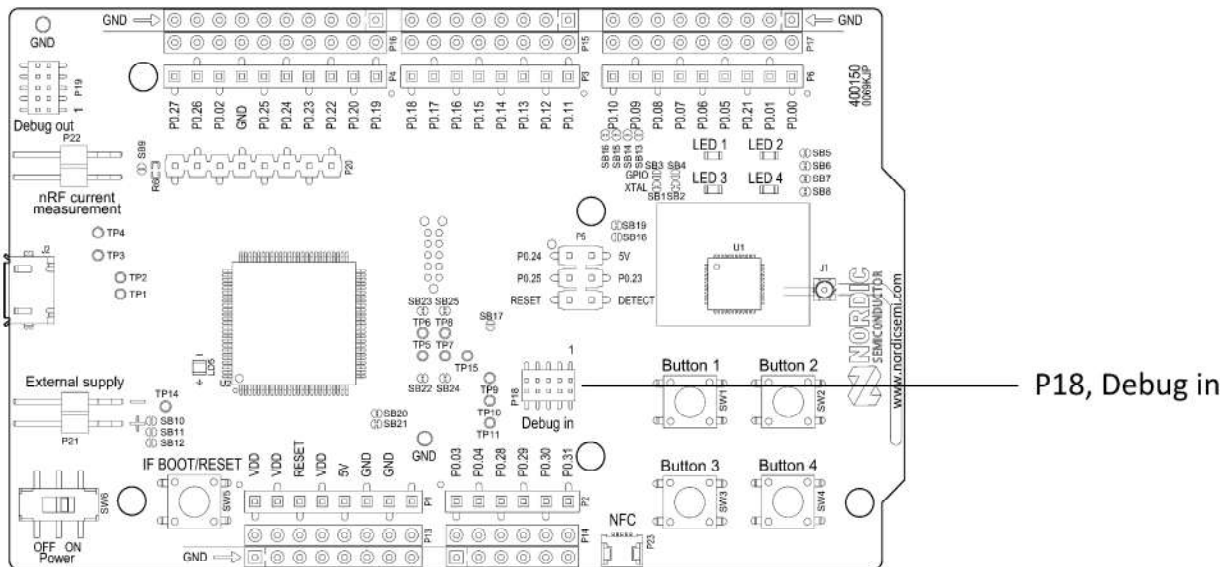


Figure 1. Debug input connector

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.10. Debug output

The nRF52 Development Kit board supports programming and debugging external boards. To debug an external board, connect to the Debug out connector (P19) with a 10 pin cable.

用PCA10040上面的interface MCU调试非PCA10040板子

Figure 1. Debug output connector

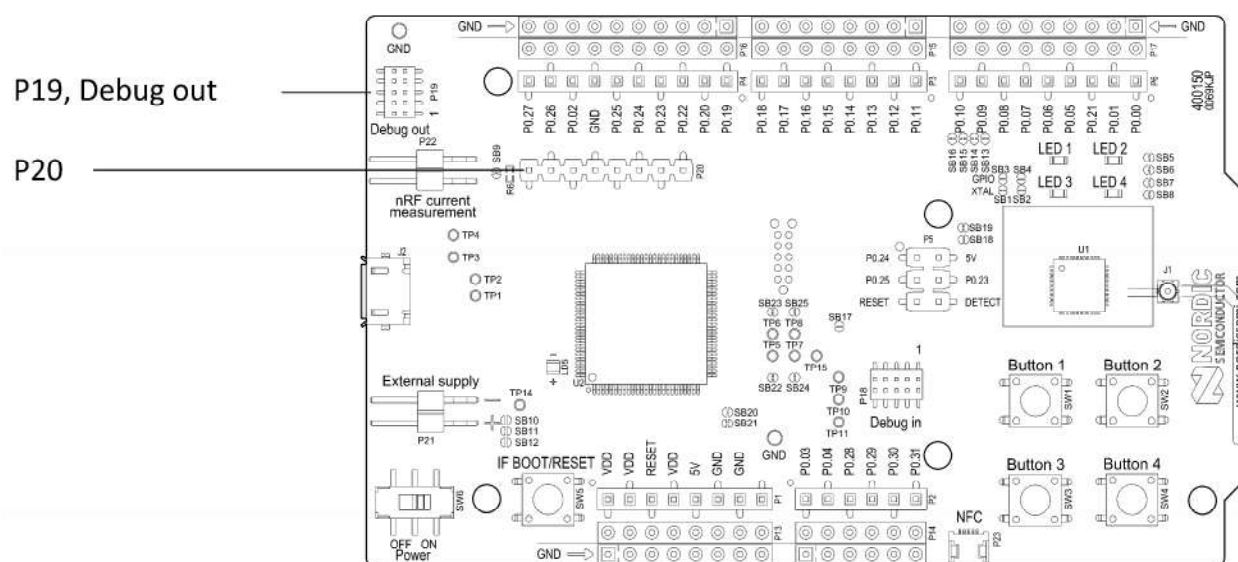


Figure 1. Debug output connector

When the external board is powered, the interface MCU will detect the supply voltage of the board and program/debug the target chip on the external board instead of the on-board nRF52832.

Important: The voltage supported by external debugging/programming is 3.0 V.

You can also use P20 as a debug out connection to program shield mounted targets. For the Debug out header (P19), the Interface MCU will detect the supply voltage on the mounted shield and program/debug the shield target.

If the Interface MCU detects target power on both P19 and P20, it will by default program/debug the target connected to P19.

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).

2.11. NFC antenna interface

The nRF52 Development Kit board supports a Near Field Communication (NFC) tag.

NFC-A listen mode operation is supported on nRF52832. The NFC antenna input is available on connector P23 on the nRF52 Development Kit board.

Figure 1. NFC antenna connector

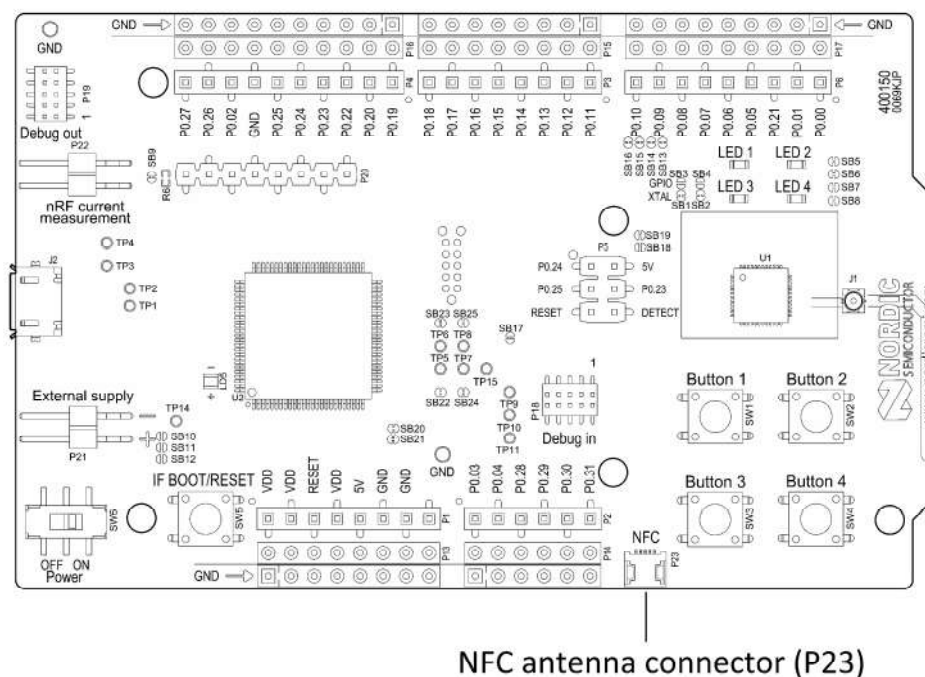


Figure 1. NFC antenna connector

NFC uses two pins, pin 11 (**NFC1**) and pin 12 (**NFC2**) to connect the antenna. These pins are shared with GPIOs (**P0.09** and **P0.10**) and the PROTECT field in the NFCPINS register in UICR defines the usage of these pins and their protection level against abnormal voltages. The content of the NFCPINS register is reloaded at every reset.

Important: The NFC pins are enabled by default. NFC can be disabled and GPIOs enabled by defining the CONFIG_NFCT_PINS_AS_GPIO variable in the project settings. This can be done by defining the preprocessor symbol in Keil, go to: **Project > Options for Target > C/C++ > Preprocessor Symbols > Define**. Here you can add the CONFIG_NFCT_PINS_AS_GPIO variable after NRF52.

This functionality can be removed by doing a nRFjprog --recover.

Pin 11 and pin 12 are by default configured to use the NFC antenna, but if pin 11 and pin 12 are needed as normal GPIOs, **R25** and **R26** must be relocated to **R27** and **R28**.

Figure 2. NFC input schematic

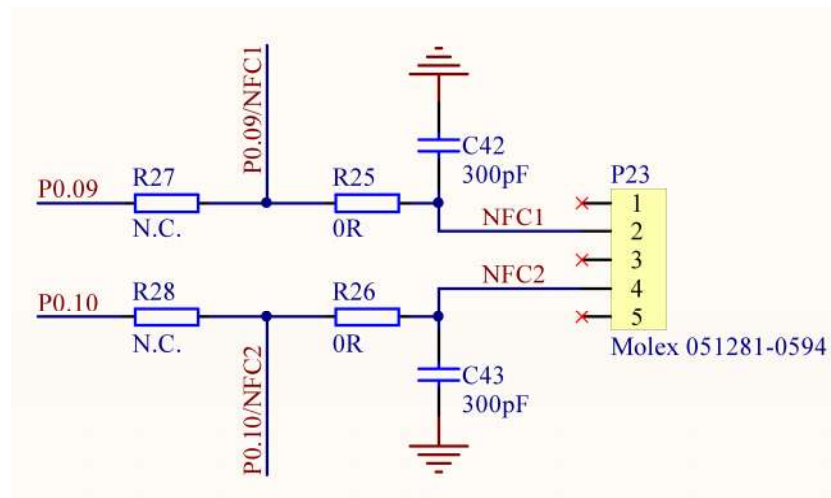


Figure 2. NFC input schematic

Parent topic: [Hardware description](#)

This document was last updated on 2016-01-20.

Please send us your [feedback](#) about the documentation! For technical questions, visit the [Nordic Developer Zone](#).