NILE (I540M0L8)

BT 5.4 + ZB/Thread + NFC-A Standalone Module

NILE DVK User Guide

Version 1.0

For additional Information, please contact info@ivativ.com

Confidential – Ivativ, Inc

Table of Contents

Table of Figures

List of Tables

Revision history

Date	Version	Author	Description
Dec 23, 19	0.9.5	Sridhar J	Pre-release version
Nov 3, 21	0.9.7	Mahesh K	Initial release
Sep 23, 24	1.0	Venkatesh k	Production & website release

1. Overview

The NILE DVK is a sophisticated and comprehensive development kit which includes hardware, software and documentation required to evaluate all the features supported by the NILE SoC

The key features of the development kit are:

- Supports the leading IoT connectivity protocols –Bluetooth 5® Low Energy, BLE Mesh and Thread, Zigbee.
- Buttons and LEDs for user interaction
- I/O interface for Arduino form factor plug-in modules
- SEGGER J-Link Onboard (**OB**) programmer, debugger with debug out functionality
- Support to bypass the OB debugger and directly connect J-Link to NILE SoC via Debug IN port
- UART interface through virtual COM port
- USB interface
- Flash memory
- Drag-and-drop Mass Storage Device (MSD) programming
- Supporting NFC-A Listen Mode
- Support for different power sources such as Battery, USB and an external power supply
- Solder bridges /Jumpers to measure current consumption / profiling

2. Requirements

Before you start, check that you have the required hardware and software.

2.1 Hardware requirements

- Personal computer (PC) or Laptop
- Micro-USB 2.0 cable
- NILE DVK

2.2 Software requirements

- Operating System: Windows 7, Windows 8, Windows 10 , MacOS , or Linux
- SEGGER J-Link Software
- nRF Command Line Tools

3. Hardware setup

3.1 NILE DVK

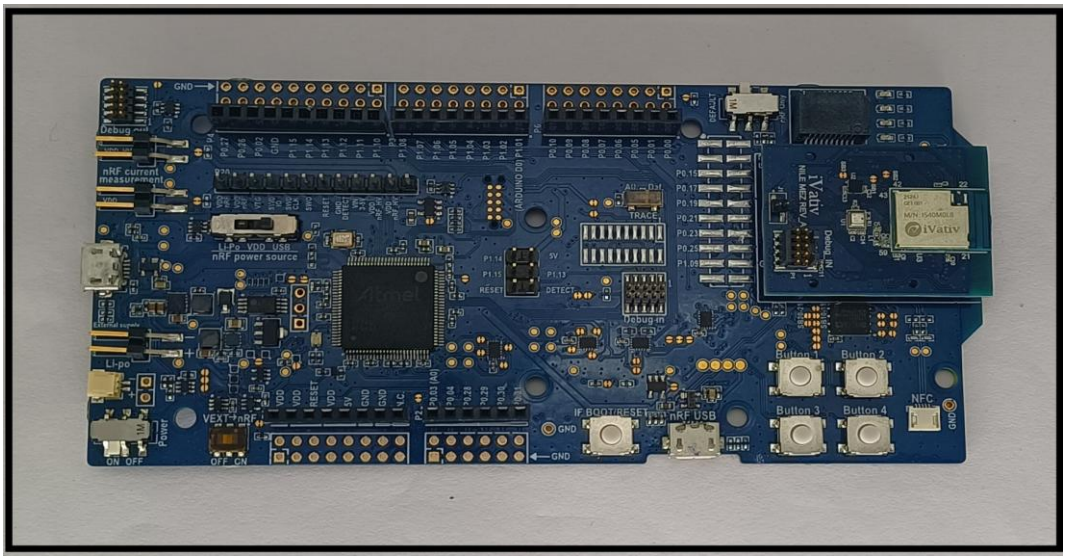

Figure 1: NILE DVK

3.2 Getting started

Before you start developing, set up the hardware and download the required software

- Unpack the Development Kit
- Make sure kit contains NILE DVK Board and Micro-USB 2.0 cable

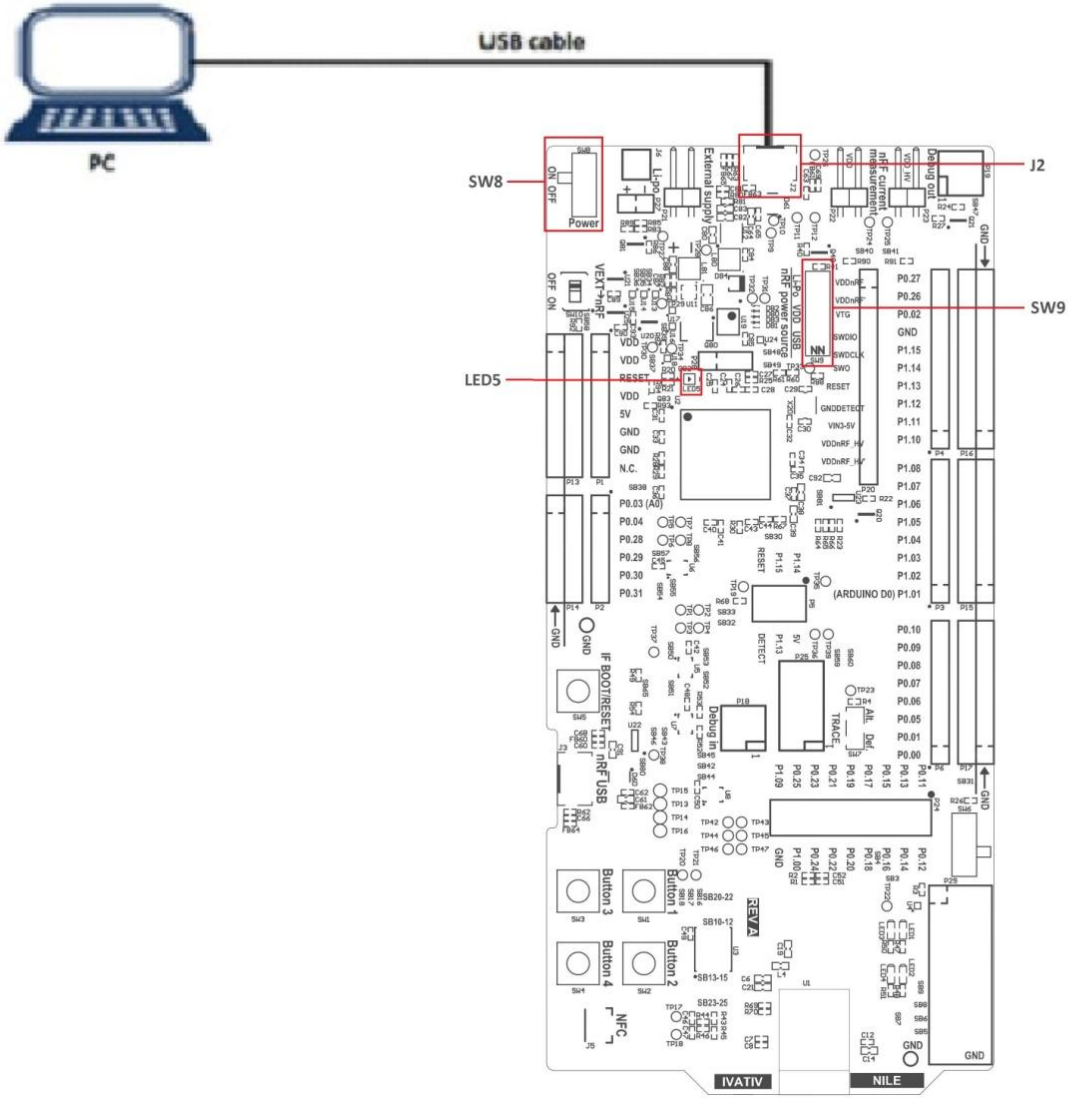

Figure 2: Connecting NILE DVK to PC

Follow the steps to set up the kit:

1. Install SEGGER Embedded Studio if not installed already (See 'Installing SES' in Chapter 5)
2. To power up the DK:
    a. Connect a micro-USB 2.0 cable to the USB connector **J2** on the NILE DVK and the other end to one of your PC's USB. In addition to providing power to the DVK this USB connection supports module programming.
    b. Set power source switch **SW9** to VDD.
    c. Set power switch **SW8** to the ON position.
    d. A pop-up may appear, it can be ignored.
    e. Check that **LED5** powered ON.

f. Make sure NILE setup completion by opening My Computer -> This PC to check that the NILE DVK has appeared as a removable drive named "J-LINK".
This allows you to program the chip on the DVK.

3.3 Windows serial terminal set-up

1. Connect the DVK to PC by plugging the Micro-USB cable at **J2** on DVK.
2. Open a serial terminal (Putty, Docklight etc.) and set the following console settings:

- **Serial Port detection:** Open Device Manager-> Ports (COM & LPT). Select the one which will display similar to **Jlink CDC UART Port (COMXX)**
- **Baud rate:** 115200
- **Hardware Flow control CTS/RTS:** Disable
- **Parity:** No
- **Data bits:** 8
- **Stop bits:** 1

4. Hardware Description

4.1 Interface MCU

The interface MCU is used to program and debug the NILE module. The interface MCU on the DVK runs the SEGGER J-Link OB interface firmware and is used to program and debug the firmware of the SoC.

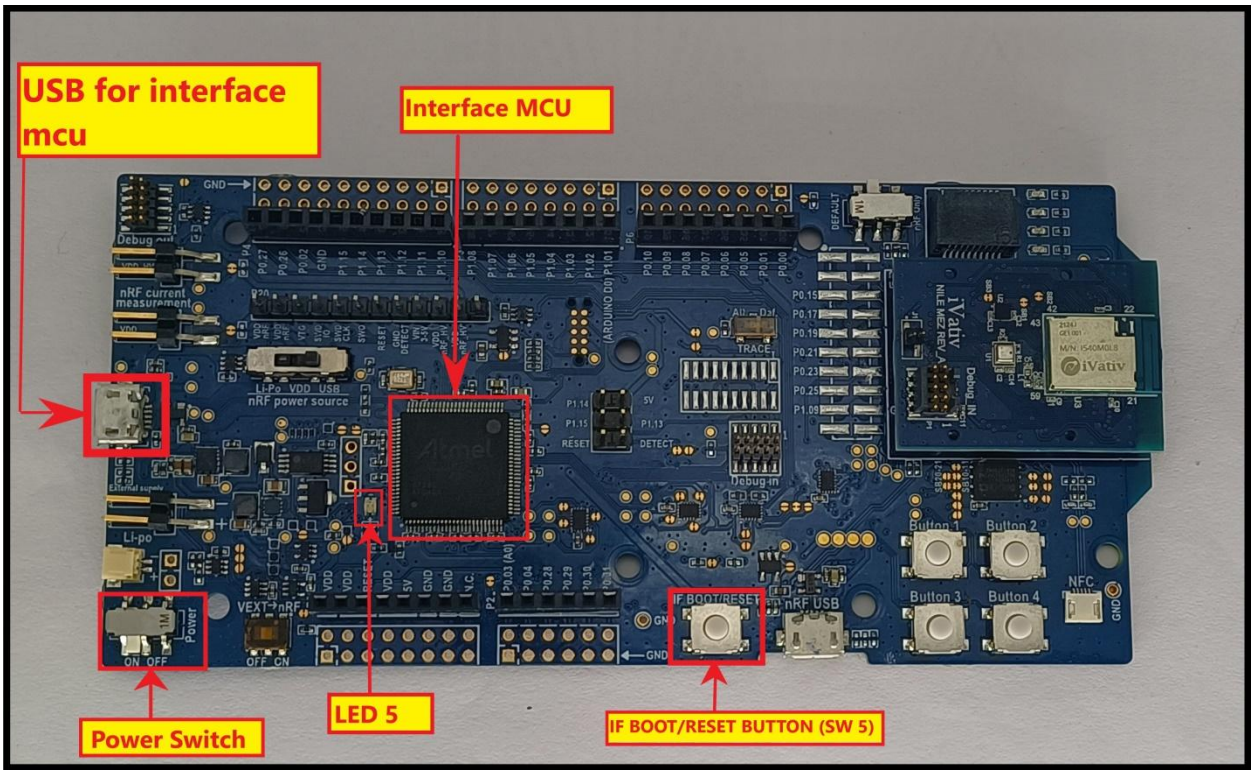

Figure 3: Interface MCU

4.1.1 IF Boot/Reset button

The NILE DVK is equipped with a Reset button (**SW5**).

This button is connected to the interface MCU on the DVK and has two functions:

- Resetting the NILE SoC.
- Entering bootloader mode of the interface MCU.

During normal operation the button will function as a reset button for the nRF52840 SoC. For this to work, pin reset on P0.18 needs to be enabled in the SoC.

The button is also used to enter the bootloader mode of the interface MCU. To enter the bootloader mode, keep the reset button pressed while powering up the DK until LED5 starts to blink. You can power

up the DK either by disconnecting and reconnecting the USB cable or by toggling the power switch (**SW8**).

Note:
Pin reset can be enabled by adding the CONFIG_GPIO_AS_PINRESET variable to the compiler preprocessor macros. The way of doing this depends on the *IDE*/toolchain in use:

- When using SEGGER Embedded Studio, go to **Project > Edit Options > Code > Preprocessor > Preprocessor Definitions** and add the CONFIG_GPIO_AS_PINRESET variable.
- When using Keil, go to **Project** > **Options for Target** > **C/C++** > **Preprocessor Symbols** > **Define** and add the CONFIG_GPIO_AS_PINRESET variable.

If your program does not enable pin reset, this functionality can also be enabled on an already programmed device by calling `nrfjprog.exe` with argument `--pinresetenable`. To disable pinreset again, reprogram with `--chiperase`.

The following changes to the solder bridges will give more flexibility for reset:
Shorting **SB44** will connect the RESET pin in the Arduino interface to the reset pin (**P0.18**) of nRF52840
When interface MCU is used (Connecting to J2 using USB):

• Shorting **SB46** will connect the RESET pin in the Arduino interface to the BOOT input of the interface MCU.

• Shorting **SB43** will connect the RESET pin in the Arduino interface to the IF Boot/Reset **button**.

When the interface MCU is not used (nRF_only mode or no connection to J2):

• Cutting **SB42** will disconnect the IF Boot/Reset button from the reset pin (**P0.18**) of nRF52840.

• Shorting **SB45** will connect the RESET pin in the Arduino interface to the reset pin (**P0.18**) of nRF52840.

4.1.2 Virtual COM port

The onboard interface MCU features a UART interface through a virtual COM port.

The virtual COM port has the following features:

- Flexible baud rate setting up to 1 Mbps. Baud rate 921 600 is not supported through the virtual COM port.
- Dynamic Hardware Flow Control (HWFC) handling.
- Tri-stated UART lines when no terminal is connected.

The table below shows an overview of the UART connections on NILE SoC and the interface MCU.

GPIO	UART connection
P0.05	RTS
P0.06	TXD
P0.07	CTS
P0.08	RXD

Table 1: Relationship of UART connections on NILE 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.

Note: The terminal software used must send a Data Terminal Ready (DTR) signal to configure the UART interface MCU pins.

The P0.05 (Request to Send (RTS)) and P0.07 (Clear to Send (CTS)) can be used freely when HWFC is disabled on the SoC.

4.1.2.1 Dynamic HWFC handling

When the interface MCU receives a DTR signal from a terminal, it performs automatic HWFC detection. Automatic HWFC detection is done by driving **P0.07** (CTS) from the interface MCU and evaluating the state of **P0.05** (RTS) when the first data is sent or received. If the state of **P0.05** (RTS) is high, HWFC is assumed not to be used. If HWFC is not detected, both CTS and RTS can be used freely by the nRF application. After a power-on reset of the interface MCU, all UART lines are tri-stated when no terminal is connected to the virtual COM port. Due to the dynamic HWFC handling, if HWFC has been used and detected, **P0.07** (CTS) will be driven by the interface MCU until a power-on reset has been performed or until a new DTR signal is received and the detection is redone. To ensure that the UART lines are not affected by the interface MCU, the solder bridges for these signals can be cut and later resoldered if needed. This might be necessary if UART without HWFC is needed while **P0.05** (RTS) and **P0.07** (CTS) are used for other purposes.

4.1.3 Mass Storage Device (MSD)

The interface MCU features an MSD. This makes the development kit appear as an external drive on your computer.

This drive can be used for drag-and-drop programming. However, files cannot be stored on this drive. By copying a HEX file to the drive, the interface MCU will program the file to the device.

Note:

- Windows might try to defragment the *MSD* part of the interface MCU. If this happens, the interface MCU will disconnect and be unresponsive. To return to normal operation, the *DK* must be power cycled.
- Your antivirus software might try to scan the *MSD* part of the interface MCU. Some antivirus programs trigger a false positive alert in one of the files and quarantine the unit. If this happens, the interface MCU will become unresponsive.
- If the computer is set up to boot from USB, it can try to boot from the *DK* if the *DK* is connected during boot. This can be avoided by unplugging the *DK* before a computer restart, or changing the boot sequence of the computer.

You can also disable the MSD of the kit by using the **msddisable** command in J-Link Commander. To enable, use the **msdenable** command. These commands take effect after a power cycle of the DVK and stay this way until changed again.

4.2 Power supply

The NILE DVK board has multiple power supply options.

The power options are:

- USB connector J2 for the interface MCU (5 V)
- USB connector J3 for the nRF52840 (5 V)

- Lithium polymer (Li-Po) battery connectors **J6** or **P27** (2.5–5.0 V)
- VIN 3–5 pin on P20 (3.0–5.0 V)
- External supply on P21 (1.7–3.6 V)
- Coin cell battery

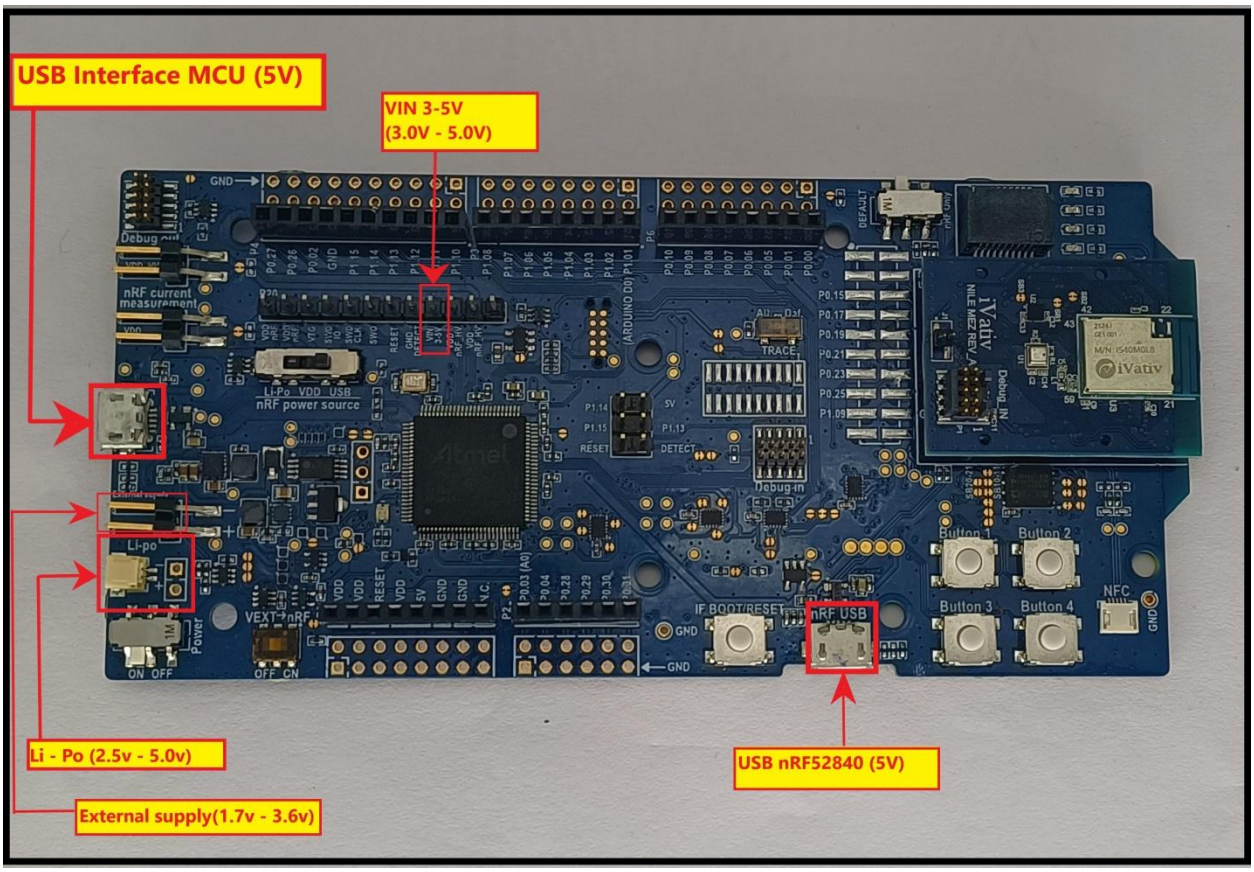

Figure 4: Power supply options (front view of DVK)

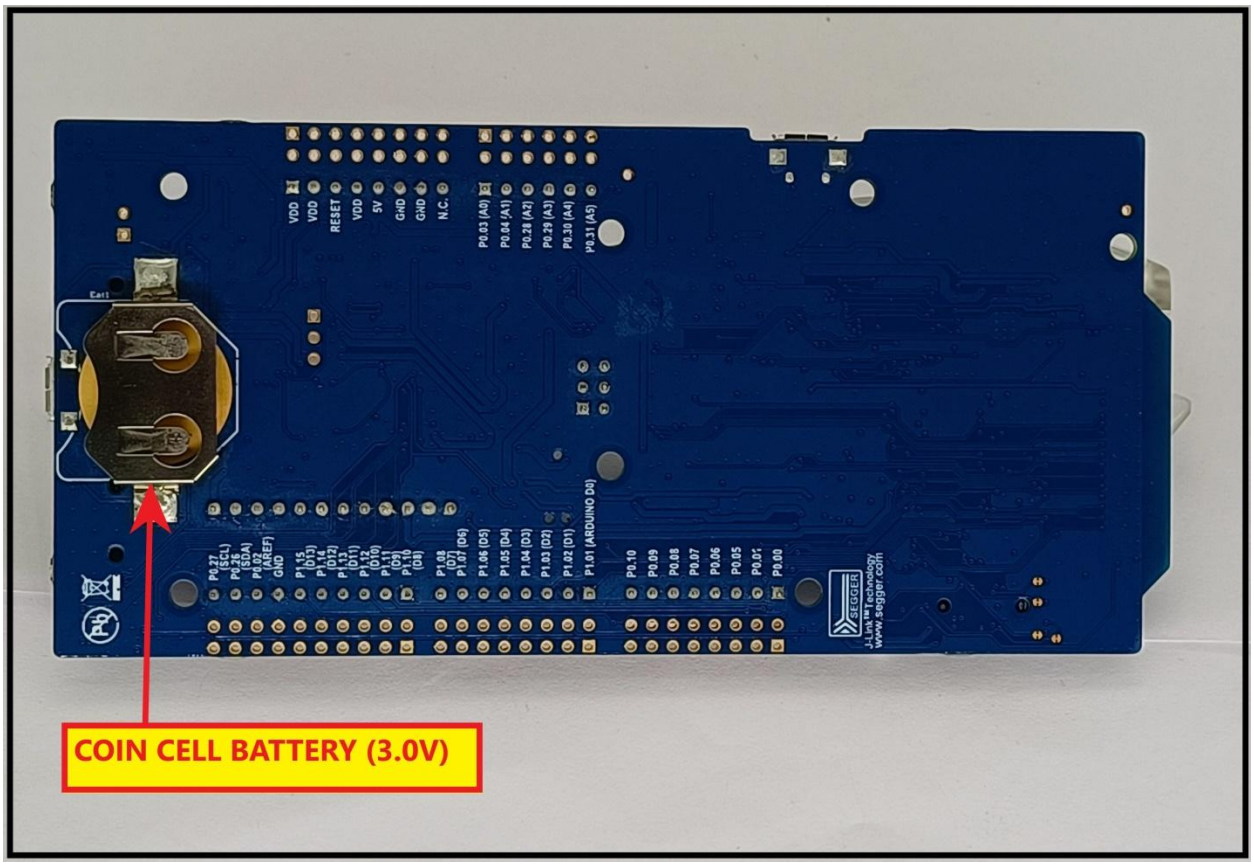

Figure 5: Power supply options (back view of DVK)

4.2.1 5 V power sources

The NILE DVK has 5V boost regulator.

It gives a stable 5 V output from four possible sources.

- USB connector **J2** for the interface MCU
- USB connector **J3** for the nRF52840
- Lithium polymer (Li-Po) battery connectors **J6** or **P27** (2.5–5.0 V)
- VIN (3 – 5 V) pin on **P20**

Each of these sources has a reverse protection diode to prevent current flowing in the wrong direction if multiple sources are connected at the same time.

4.2.2 VDD Power sources

The main supply (VDD) can be sourced from the 5 V domain, external power supply, and coin cell battery.

For the 5 V domain, there are two regulators, one fixed 3 V buck regulator and one voltage follower regulator that follows the VDD_nRF voltage. The coin cell battery and external power supply are not regulated.

- 5 V domain:
  - Fixed 3 V buck regulator
  - VDD_nRF voltage follower
- External power supply
- Coin cell battery

For more information about power sources, see section NILE nRF power source.

The power sources are routed through a set of load switches, which is controlled by logic to give the correct priority of the power sources.

If the high voltage regulator of the nRF52840 is used, the DK will be supplied from the VDD_nRF voltage follower regardless of the state of the other power sources.

The power switches work in the way that the body diode of the internal transistor powers the VSUPPLY net, which supplies the gates controlling the enable signal of the switches. If 5 V is present, the switches for external supply and battery are disabled. If external supply is present, the switch for the battery is disabled.

4.2.3 Interface MCU power

Interface MCU is powered off when J2 USB is not connected or when SW6 is switched to nRF Only mode.

4.2.4 NILE power source

The NILE DVK has a power source switch (**SW9**) for selecting between three power sources for the NILE SoC. The three positions of the switch are:

- VDD (default)
- Li-Po
- USB (J3)

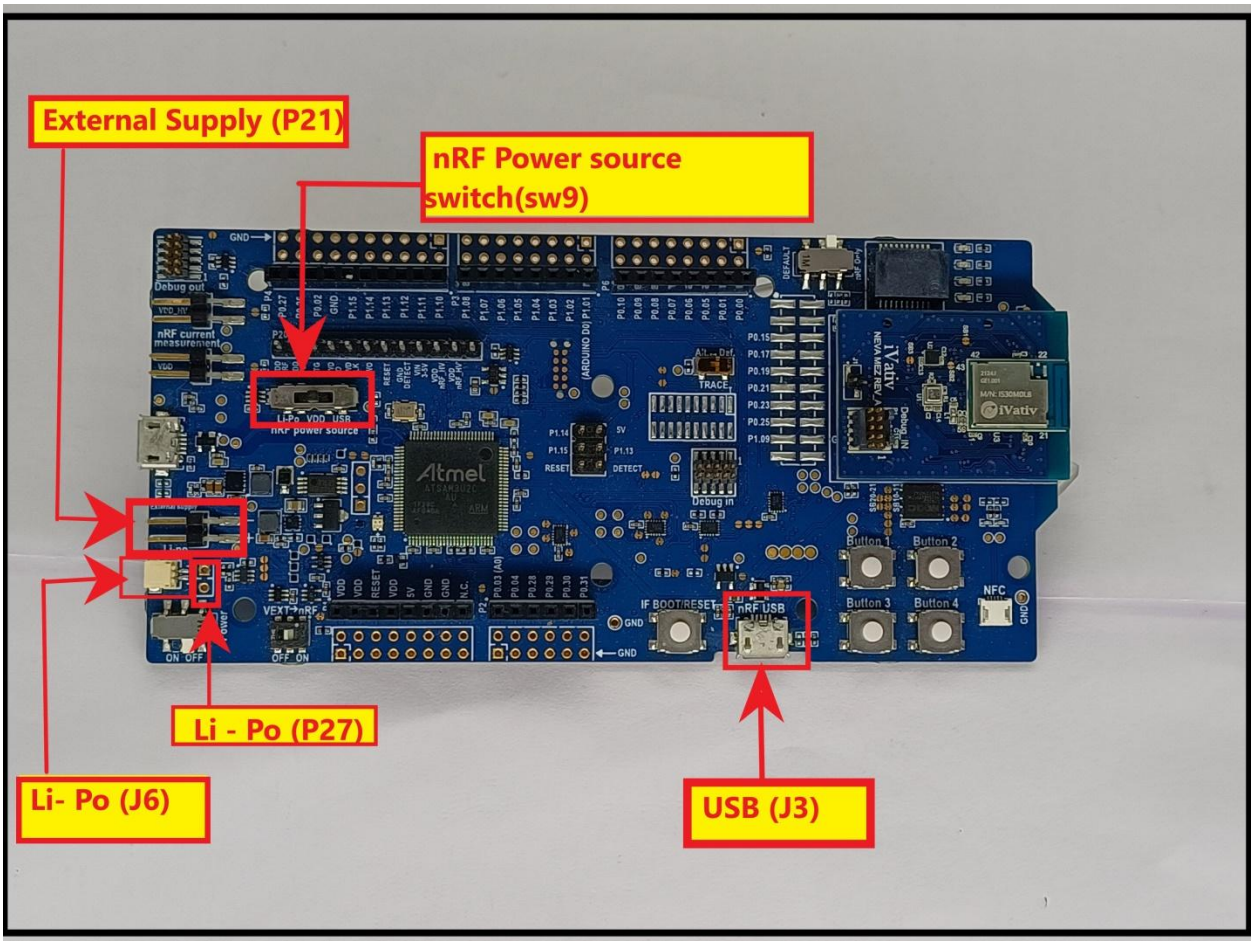

Figure 6: NILE nRF power source switch

The NILE SoC has a high voltage buck regulator that can support up to 5 V input.

In the **VDD** position, the SoC is powered either from the onboard buck regulator, coin cell battery, or external supply (**P21**).

In the **Li-Po** position, the high voltage regulator of the SoC is supplied directly from the Li-Po battery connectors (**J6** or**P27**).

In the **USB** position, the USB high voltage regulator gets power from the NILE USB connector (**J3**).

Li-Po input supply can be used for this switch kept at both VDD and Li-Po position

4.2.5 NILE direct supply

It is possible to power the SoC directly from a source without powering the rest of the DVK from the same source.

The external source can be connected to the external supply connector **(P21)** and the VEXT->nRF switch **(SW10)** put in the ON position. The nRF power source switch **(SW9)** must be in the VDD position, the external supply voltage should not be exceed 1.7 – 3.6V

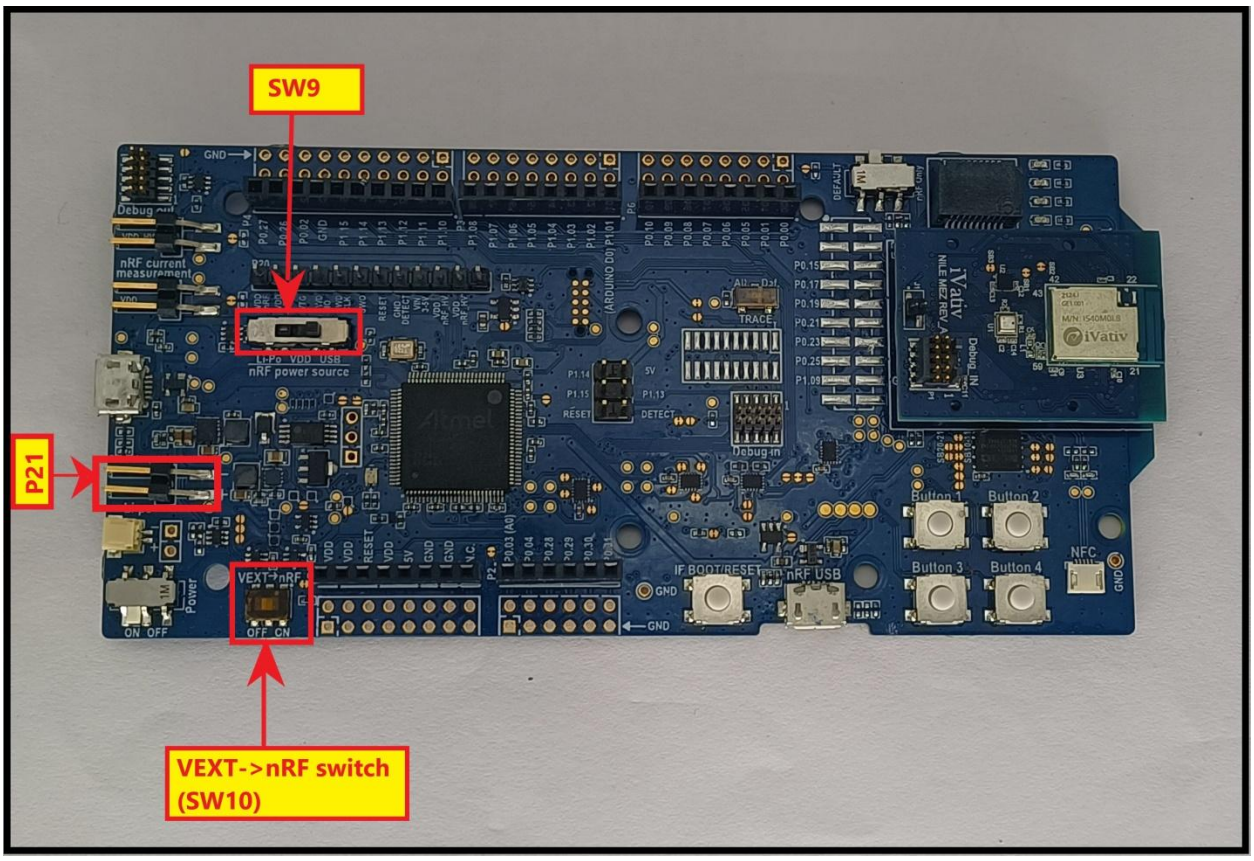

Figure 7: VEXT->nRF switch (SW10)

Since it is only the NILE SoC that is supplied from this source, it is recommended to supply the VDD domain from a different source to prevent the pins of the SoC to be connected to unpowered devices.

4.3 Operating modes

The NILE DVK has various modes of operation.

4.3.1 USB detect

In this mode Interface MCU will be powered on and detects when a USB cable is attached to **J2**.

4.3.2 nRF only mode

The nRF only mode disconnects the power supply of the interface MCU, the external memory, and the LEDs as well as disconnects the signal lines between the NILE SoC and the interface MCU using analog switches.

This is done to isolate the chip on the DK as much as possible, and can be of use when measuring currents on low-power applications.

The power supply of the external memory can be changed to maintain operation in the nRF only mode.

As shown in figure the switch is in default mode (right side). To set it to nRF only mode, toggle the switch to other side (left side).

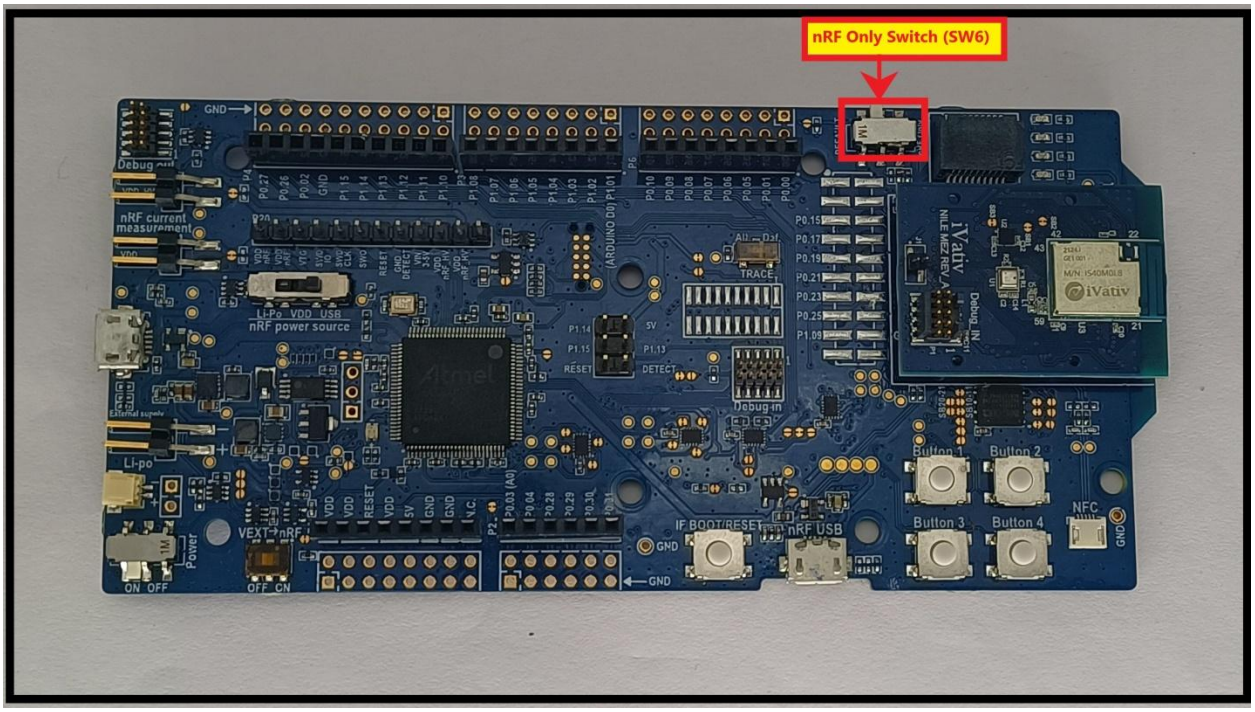

Figure 8: nRF only switch (SW6)

4.4 External memory

The nRF52840 DK has a 64 Mb external flash memory. The memory is a multi-I/O memory supporting both regular SPI and Quad SPI.

The memory is connected to the chip using the following GPIOs:

GPIO	Flash memory pin	Solder bridge for memory use(default: shorted)	Solder bridge for GPIO use(default: open)
P0.17	CS	SB13	SB23
P0.19	SCLK	SB11	SB21
P0.20	SIO_0/SI	SB12	SB22
P0.21	SIO_1/SO	SB14	SB24
P0.22	SIO_2/WP	SB15	SB25
P0.23	SIO_3/HOLD	SB10	SB20

Table 2: Flash memory GPIO usage and connecting solder bridges

To use the GPIOs for a purpose other than the onboard external memory and have them available on the P24 connector, six solder bridges (SB10–SB15) must be cut and six solder bridges (SB20–SB25) must be shorted. See the following figure for details.

Note: If debugging the QSPI communication is needed, the SB20–SB25 can be shorted without cutting SB10–SB15, but the pins should not be driven externally

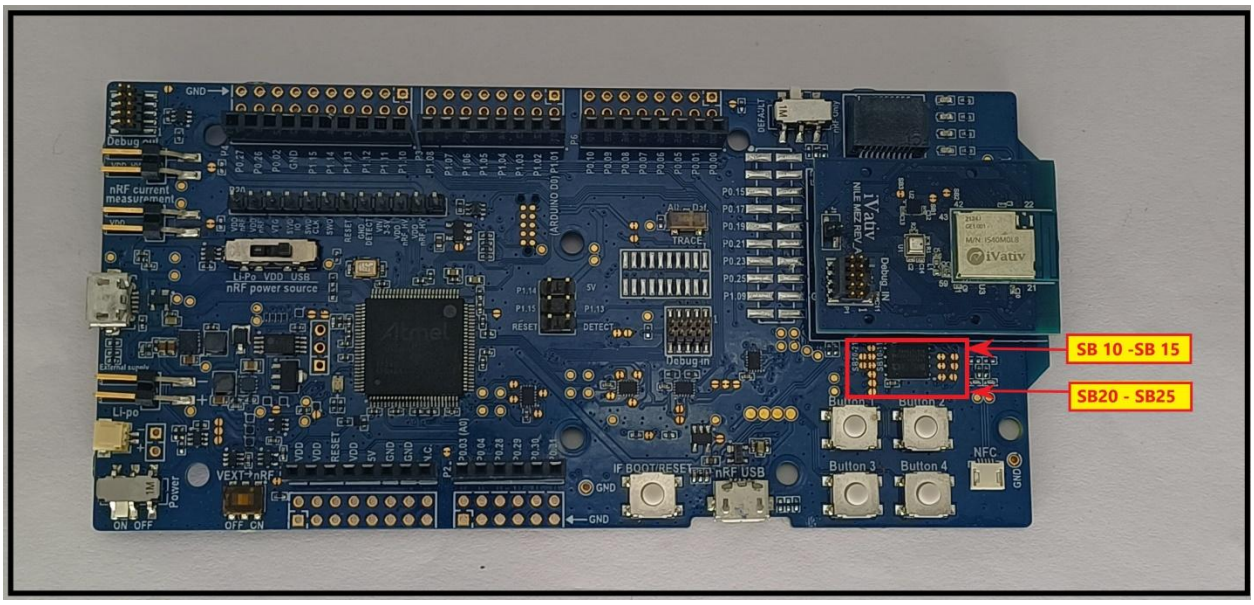

Figure 9: Configuring GPIOs for external memory

By default, the power supply of the external memory is coming from the VDD domain and it is controlled by the nRF only switch (SW6). In the nRF only mode, there are two optional power sources for keeping the external memory powered, VDD and VDD_nRF. If VDD_nRF is selected, the power consumption of the external memory will be added to the nRF52840 current measured on P22 or P23. See the following table for configuration:

Power source	Solder bridge	Default state
VDD_PER	SB16	Shorted
VDD	SB17	Open
VDD_nRF	SB18	Open

Table 3: Flash memory power source configuration

4.5 Connector interface

Access to the NILE DVK GPIOs is available from connectors **P2, P3, P4, P6 and P24**

The **P1** connector provides access to ground and power on the NILE DVK.

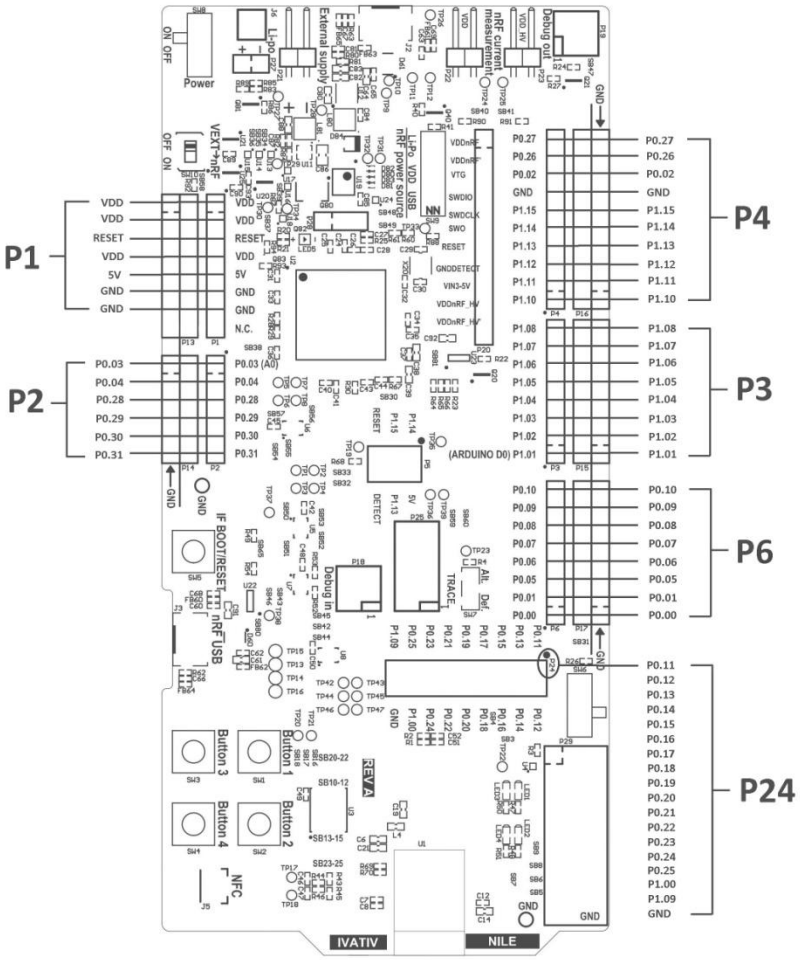

Figure 10: NILE DVK connectors

Note:

Some pins have default settings:

P0.05, **P0.06**, **P0.07**, and **P0.08** are used by the UART connected to the interface MCU.

P0.09 and **P0.10** are by default used by NFC1 and NFC2. For more information, see NFC antenna interface

P0.11–**P0.16** and **P0.24**–**P0.25** are by default connected to the buttons.

P0.17 and **P0.19**–**P0.23** are by default connected to the external memory.

Header P24

It can be used to access GPIOs other than GPIOs mentioned in Figure 9, but make sure you need to modify *solder bridge* default configurations. This header is not mounted by default.

When the NILE DVK is used as a shield together with an Arduino standard motherboard, the Arduino signals are routed as shown in the figure below.

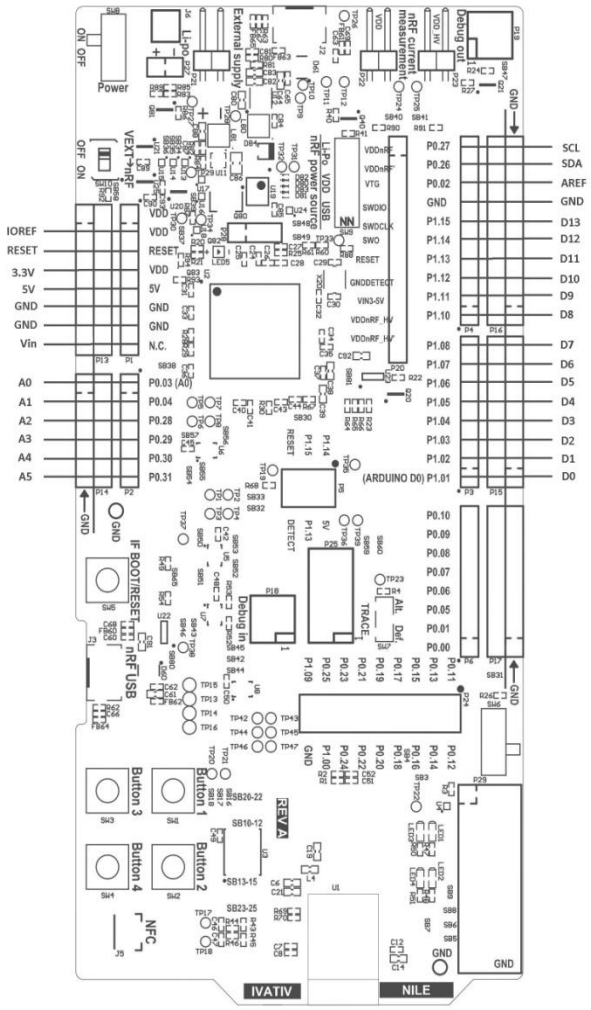

Figure 11: Arduino signals routing on the NILE DVK

4.5.1 Mapping of analog pins

The table shows the mapping between GPIO pins, analog inputs, and the corresponding Arduino analog input naming.

GPIO	Analog input	Arduino naming
P0.03	AIN1	A0
P0.04	AIN2	A1
P0.28	AIN4	A2
P0.29	AIN5	A3
P0.30	AIN6	A4
P0.31	AIN7	A5

Table 4: Mapping of analog pins

4.6 Buttons and LEDs

The four buttons and four LEDs on the NILE DVK are connected to the dedicated GPIOs on the NILE SoC.

Part	GPIO	Solder bridge
Button 1	P0.11	-
Button 2	P0.12	-
Button 3	P0.24	-
Button 4	P0.25	-
LED 1	P0.13	SB5
LED 2	P0.14	SB6
Led 3	P0.15	SB7
LED 4	P0.16	SB8

Table 5: Button and LED connection

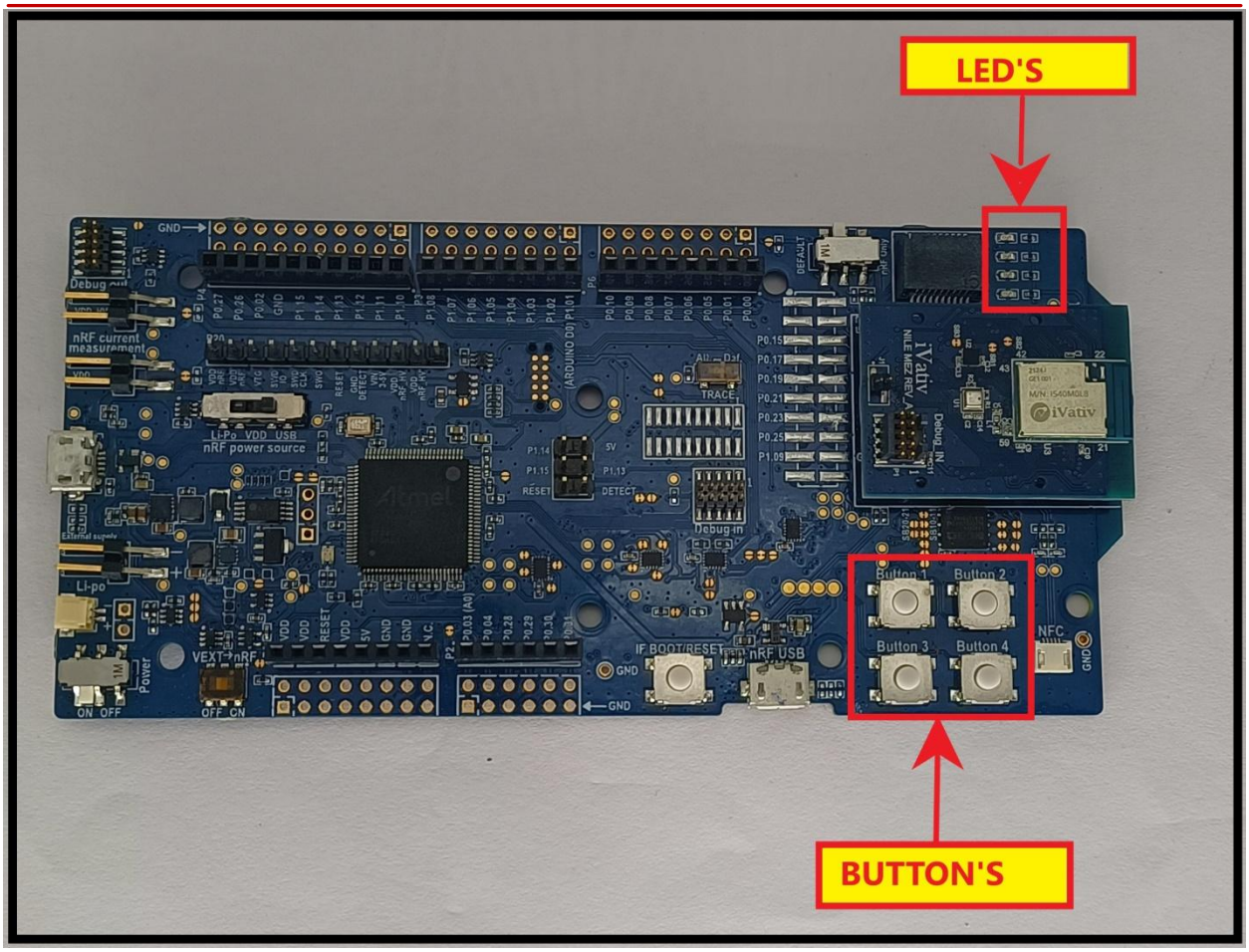

Figure 12: Buttons and LED's

If P0.13–P0.16 are needed elsewhere, the LEDs can be disconnected by cutting the short on **SB5–SB8.**

The buttons are active low, meaning that the input will be connected to ground when the button is activated. The buttons have no external pull-up resistor, and therefore, to use the buttons, the **P0.11**, **P0.12**, **P0.24**, **P0.25** 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.

4.7 Debug input and trace

The Debug in connector (P18) makes it possible to connect external debuggers for debugging when the interface MCU USB cable is not connected or the DK is in nRF only mode.

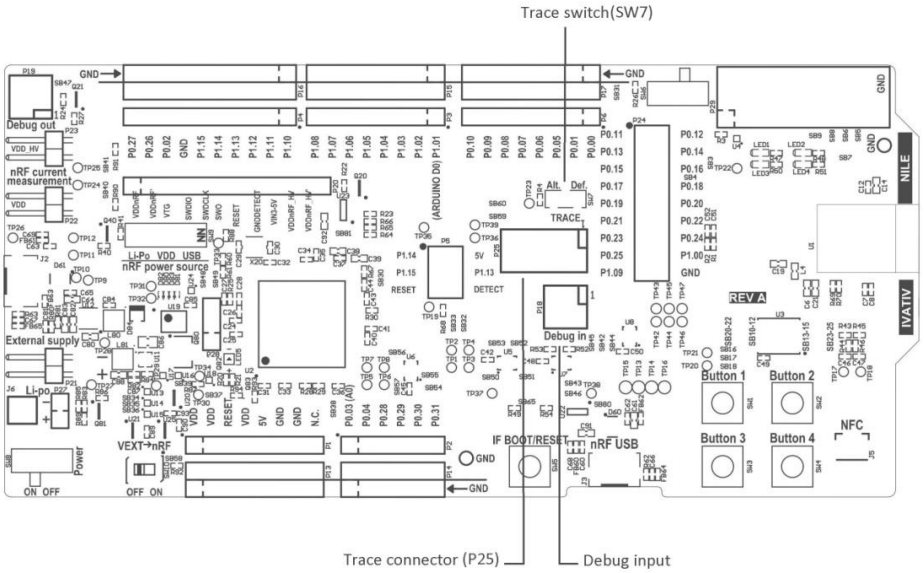

Figure 13:Debug input and trace connectors

For trace, a footprint for a 20-pin connector is available (P25). If trace functionality is required, it is possible to mount a 2×10 pin 1.27 mm pitch surface mount pin header. Some of the trace pins are by default used for other functionality on the DK. By sliding the TRACE switch (SW7) from Def. to Alt., the functionality is moved to other GPIOs. See the following table:

GPIO	Trace	Default use	Optional GPIO
P0.07	TRACECLK	UART CTS	P0.04
P1.00	TRACEDATA[0]		
P0.11	TRACEDATA[1]	Button 1	P1.07
P0.12	TRACEDATA[2]	Button 2	P1.08
P1.09	TRACEDATA[3]		

Table 6: Default and Trace GPIOs

The reference voltage for the debug input and trace is by default connected to VDD_nRF'. This can be connected to the VDD by cutting SB60 and soldering SB59.

4.8 Debug output

The NILE DVK supports programming and debugging external boards which support SWD interface. To debug an external board with SEGGER J-Link OB IF , connect to the Debug-out connector (**P19**) with a 10-pin cable.

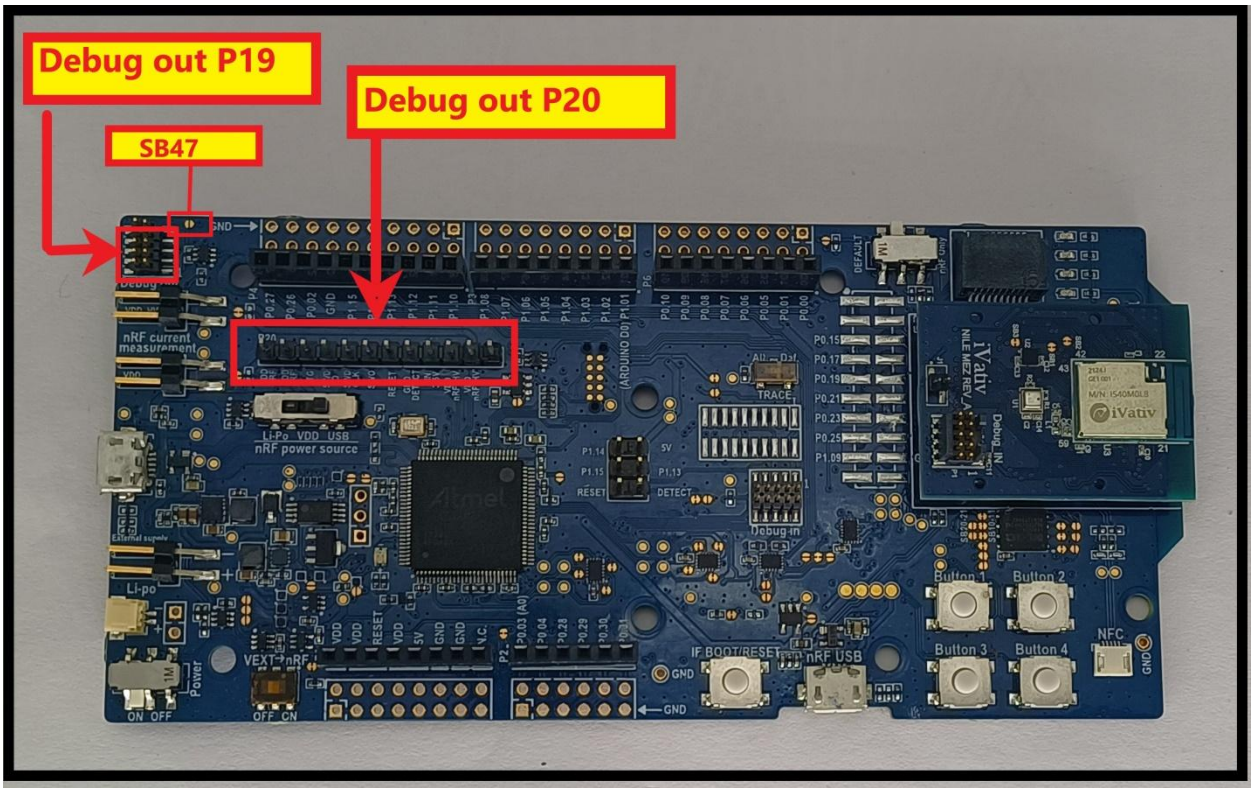

Figure 14: Debug output connector

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 onboard NILE SoC.

Note: The voltage supported by external debugging/programming is VDD voltage. Normally, this is 3 V when running from USB, but if the onboard nRF52840 SoC is supplied from either USB or Li-Ion, the nRF power source switch (**SW9**) is in either Li-Po or USB position, and VDD can be set by the nRF52840 firmware. Make sure the voltage level of the external board matches the VDD of the nRF52840 DK.

You can also use P20 as a debug out connection to program shield-mounted targets. For both **P19** and **P20**, the interface MCU will detect the supply voltage on the mounted shield and program/debug the target.

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

If it is inconvenient to have a separate power supply on the external board, the NILE DVK can supply power through the Debug out connector (**P19**). To enable this, short the solder bridge (**SB47)**. Please note that as long as **SB47** is shorted, it is not possible to program the onboard NILE SoC even if the external board is unplugged.

4.9 NFC antenna interface

The NILE DVK supports a Near Field Communication (NFC) tag.

NFC-A listen mode operation is supported on the NILE SoC. The NFC antenna input is available on connector **J5** on the NILE DVK.

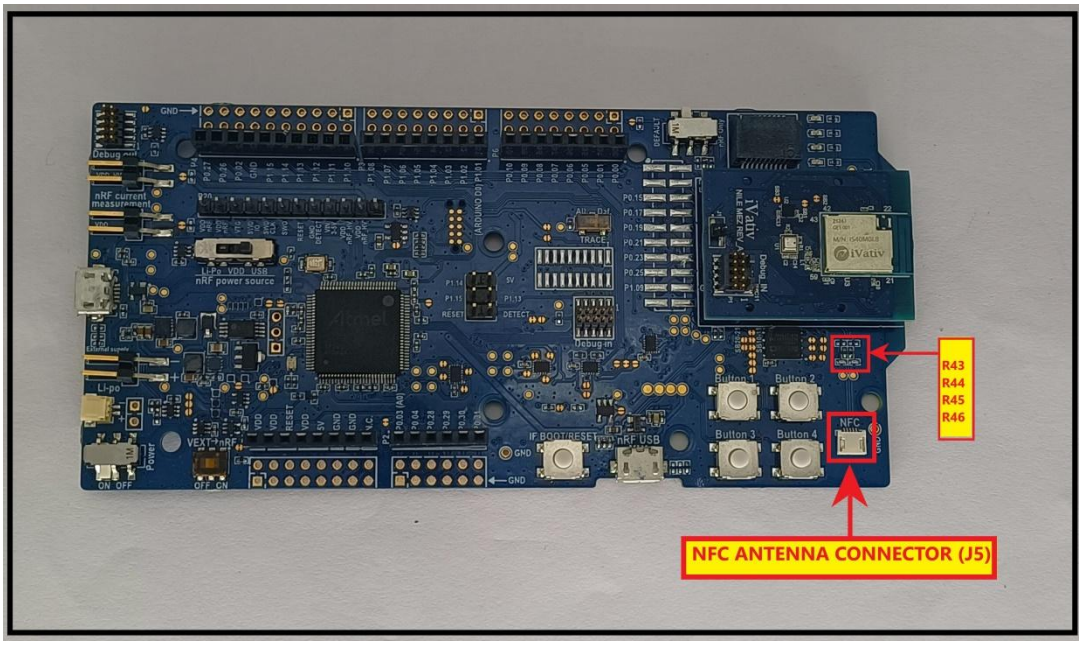

Figure 15: NFC antenna connector

NFC uses two pins , **L24** (NFC1) and **J24** (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.

Note : The NFC pins are enabled by default.

NFC can be disabled and GPIOs enabled by defining the CONFIG_NFCT_PINS_AS_GPIOS variable in the project settings. The way of doing this depends on the *IDE*/toolchain in use:

- When using SEGGER Embedded Studio, go to **Project** > **Edit Options** > **Code** > **Preprocessor** > **Preprocessor Definitions** and add the CONFIG_NFCT_PINS_AS_GPIOS variable.
- When using Keil, go to **Project** > **Options for Target** > **C/C++** > **Preprocessor Symbols** > **Define** and add the CONFIG_NFCT_PINS_AS_GPIOS variable.

Pins **L24** and **J24** are by default configured to use the NFC antenna, but if they are needed as normal GPIOs, **R44** and **R46** must be NC and **R43** and **R45** must be shorted by **0R**.

5. Software

5.1 Software

Software is distributed in open CPU form, which uses Nordic semiconductor's nRF5 SDK

In the Nordic semiconductor's nRF5 SDK, you can find precompiled application firmware examples

https://www.nordicsemi.com/Products/Development-software/nrf5-sdk/downloadHere we provide links for technical documentation. Please try to develop using latest releases.

Follow https://infocenter.nordicsemi.com/index.jsp?topic=%2Fstruct_nrf5gs%2Fstruct%2Fnrf5gs_sw_dev.html&cp=1_1 for developing and programming application

Follow https://infocenter.nordicsemi.com/index.jsp?topic=%2Fstruct_sdk%2Fstruct%2Fsdk.html&cp=5 to facilitate firmware development for different wireless protocols

Follow https://infocenter.nordicsemi.com/index.jsp?topic=%2Fsdk_nrf5_v17.0.2%2Fexamples_ble.html&cp=7_1_4_2for developing BLE wireless protocol stack based GATT applications

5.2 Related documentation

In addition to the information in this document, you may need to consult other documents

- NILE Data Sheet
- Refer to the documents section in the https://ivativ.com/product/nile/

5.3 Installing SES (SEGGER Embedded Studio)

The SDK given by Nordic has some ways to test the application, in which SEGGER Embedded Studio is one method

- Go to SEGGER Embedded Studio https://www.segger.com/downloads/embedded-studio/
- Nordic SoC is based on ARM architecture. Select the download option in the **Embedded Studio for ARM**. Download application based on the type of your system architecture (32 or 64 bit) for Windows.

5.4 Downloading SEGGER J-Link software

Follow the steps to download the J-Link software.

- Get the J-Link software from https://www.segger.com/downloads/jlink/
- Go to "Software and documentation pack"
- Click on "Click for Downloads"
- Select the download based on your OS version 32 bit or 64 bit

5.5 Installing nRF command line tools

Install the nRF command line tools if not installed, by following the steps at https://www.nordicsemi.com/Products/Development-tools/nRF-Command-Line-Tools/Download#infotabs

5.6 Application development

After you have set up the development kit and installed the tool chain, choose depending on which networking protocol you want to use.

For nRF5 SDK for Bluetooth® Low Energy (nRF5 Series devices), see *nRF5 SDK Getting Started*.

For nRF5 SDK for Mesh (nRF5 Series devices), see *Getting Started with Mesh*.

For nRF5 SDK for Thread and Zigbee, see *Getting Started with Thread and Zigbee*.

See also *Software development Getting Started Guides* for guidance for the main Integrated Development Environment (IDE).

5.7 Running examples

Here we provide the example how to use SES (SEGGER Embedded Studio) tool chain.

After installing the SEGGER Embedded Studio and nRF5 SDK, go to the path of the SDK. To open and execute peripheral example applications in SDK as follows:

nRF5 SDK(latest version)-->Examples-->BLE peripheral -->example (select-any) -> pca10056 (board type) --> s140 (softdevice) -->ses -->application.emprojectfile.

Double click on the application.project extension file, it will directly lead you to the SEGGER Embedded Studio, where you can edit and load the image to NILE module flash.

6. Measuring current

The current drawn by the NILE SoC can be monitored using the NILE DVK.

Current can be measured using various test instruments. Examples of test equipment are the following:

- Power analyzer
- Oscilloscope
- Ampere-meter
- Power profiler kit

Measuring power with Ampere meter will be described in this document.

Note: In order to measure using the PPK you need to mount 12 pin single lane male headers or bring out the pins on P20 header.

The NILE DVK has two possible power supplies, VDD (1.7–3.6 V) and VDDH (2.5–5.5 V). The NILE DVK is prepared for measuring current on both domains. Only the VDD domain current measurement is described here, but the approach is the same with the VDDH supply. See the following table for the corresponding components:

Component	VDD	VDDH
Measurement connector	P22	P23
Solder bridge	SB40	SB41
Series resistor	R90	R91

Table 7: Components for current measurement on VDD and VDDH

Note:

When measuring the current consumption:

It is not recommended to use a USB connector to power the DVK during current measurements. However, when measuring current on an application using the USB interface of the NILE SoC, the USB must be connected. It is recommended to power the DVK from a coin cell battery, external power supply on connector **P21** (1.7–3.6 V), or through the Li-Po connector **J6** or **P27** (2.5–5.0 V).

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

After programming the NILE SoC, the USB for the interface MCU must be disconnected.

For more information on current measurement, see **'NILE Power measurement'** App note.

6.1 Preparing the development kit

To measure current, you must first prepare the DVK.

The suggested configurations actually split the power domains for the NILE SoC and the rest of the DVK.

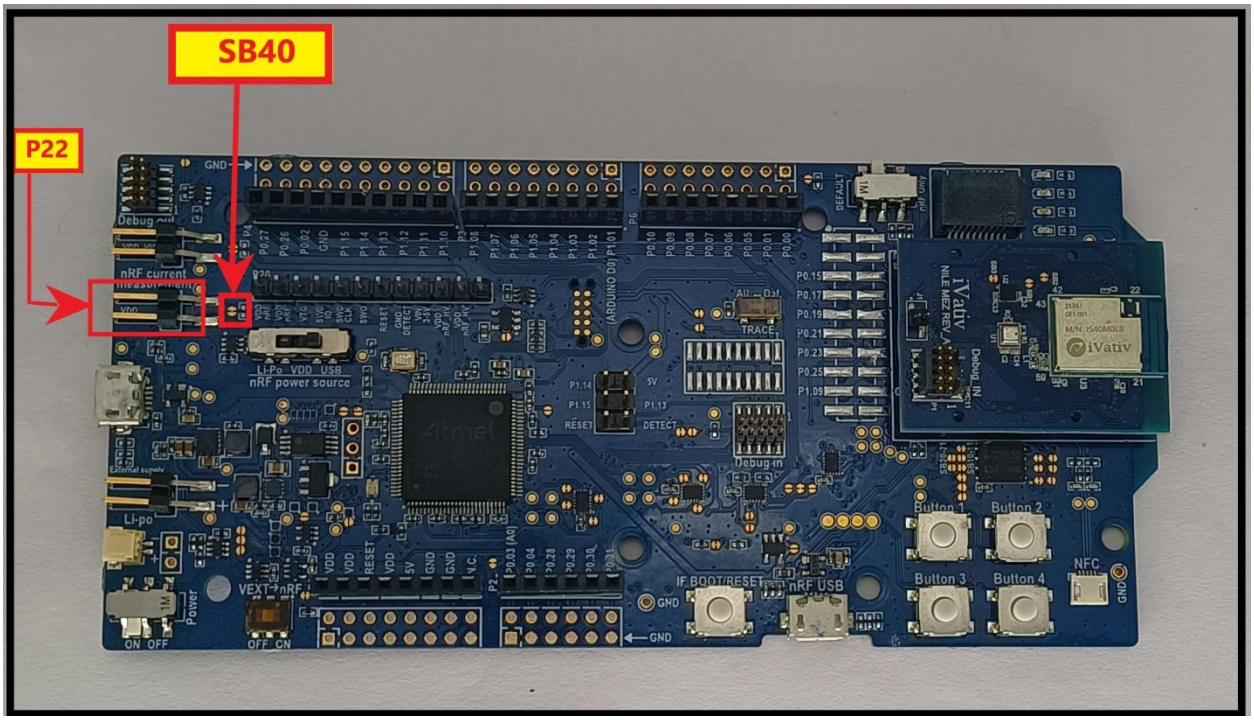

Figure 16: Preparing the DVK for current measurements

*To put **P22** in series with the load, cut the PCB track shorting solder bridge **SB40.***

- To restore normal kit function after measurement, solder **SB40** or apply a jumper on **P22.**
- To reprogram the NILE SoC while the DVK is prepared for current measurements, remove measurement devices from **P22**, and then connect the USB cable.

6.2 Using an ampere-meter for current measurement

The average current drawn by the NILE SoC can be measured using an ampere-meter. This method will monitor the current in series with the nRF device.

Make sure you have prepared the development kit as described in section preparing the development kit mentioned above.

Connect an ampere-meter between the pins of connector **P22** as shown in the figure below.

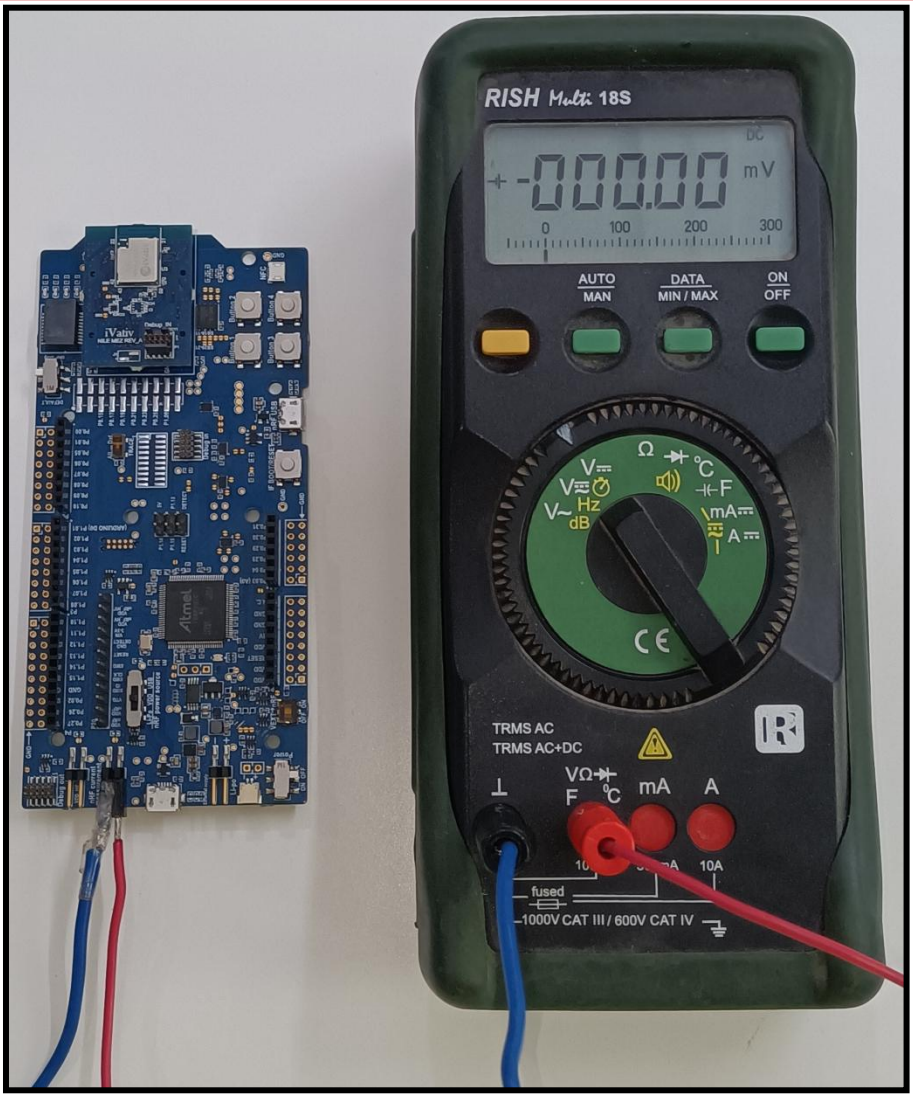

Figure 17: Current measurement with an ampere-meter

Note: An ampere-meter will measure the average current drawn by the NILE SoC if:

The NILE SoC is in a state where it draws a constant current, or, the activity on the device changing load current, like BLE connection events, is repeated continuously and has a short cycle time (less than 100 ms) so that the ampere-meter will average whole load cycles and not parts of the cycle.

It is recommended that you use a true RMS ampere-meter.

7. RF measurements

The NILE DVK is assembled with MHF4 for conducting measurements of the RF signal using a spectrum analyzer.

A test probe is available with a standard SMA connection on the other end for connecting instruments (the test probe is not included with the kit).

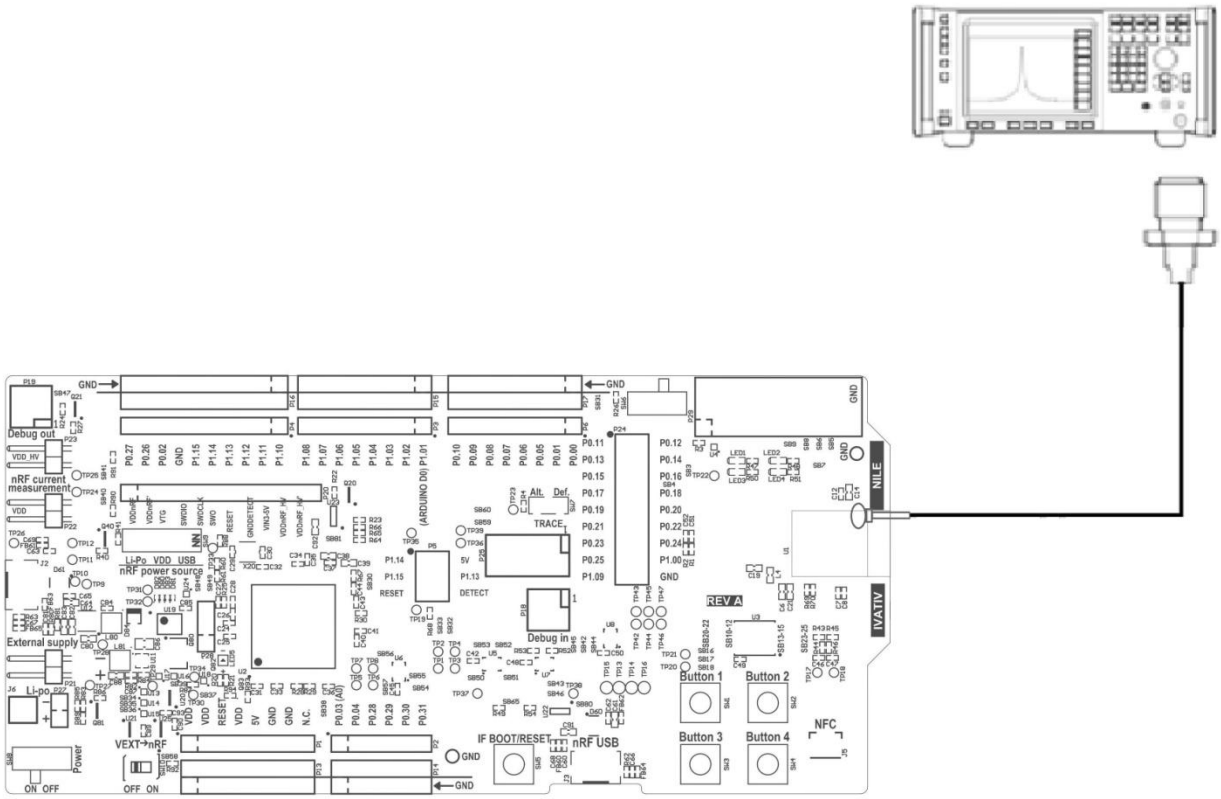

Figure 18: Connecting a spectrum analyzer

The connector and test probe will add loss to the RF signal, which should be taken into account when measuring, see the following table:

Frequency (MHz)	Loss (dB)
2440	1.0

Table 8: Typical loss in connector and test probe

8. Solder bridge configuration

Solder bridge	Default	Function
SB5	Closed	Cut to disconnect LED1
SB6	Closed	Cut to disconnect LED2
SB7	Closed	Cut to disconnect LED3
SB8	Closed	Cut to disconnect LED4
SB9	Open	Short to bypass peripheral power switch
SB10	Closed	Cut to disconnect the QSPI memory from P0.23
SB11	Closed	Cut to disconnect the QSPI memory from P0.19
SB12	Closed	Cut to disconnect the QSPI memory from P0.20
SB13	Closed	Cut to disconnect the QSPI memory from P0.17
SB14	Closed	Cut to disconnect the QSPI memory from P0.21
SB15	Closed	Cut to disconnect the QSPI memory from P0.22
SB16	Closed	Cut to disconnect QSPI memory power supply from VDD_PER
SB17	Open	Short to connect QSPI memory power supply to VDD
SB18	Open	Short to connect QSPI memory power supply to VDD_Nrf
SB20	Open	Short to enable P0.23 as a normal GPIO
SB21	Open	Short to enable P0.19 as a normal GPIO
SB22	Open	Short to enable P0.20 as a normal GPIO
SB23	Open	Short to enable P0.17 as a normal GPIO
SB24	Open	Short to enable P0.21 as a normal GPIO
SB25	Open	Short to enable P0.22 as a normal GPIO
SB30	Open	Short to reset the interface MCU
SB31	Open	Short to bypass the USB detect switch

SB32	Open	Short to permanently enable the I2C pull-up resistors
SB33	Closed	Cut to permanently disable the I2C pull-up resistors
SB34	Open	Short to bypass the power switch on the USB power
SB35	Open	Short to bypass the power switch on the coin cell battery power
SB36	Open	Short to bypass the power switch on the external supply power
SB37	Open	Short to bypass the interface MCU power switch
SB38	Closed	Cut to disable VDD power to the Arduino interface
SB39	Open	Short to bypass the power switch for regulator, coin cell, or external supply
SB40	Closed	Cut for current measurements of the VDD_nRF
SB41	Closed	Cut for current measurements of the VDD_nRF_HV
SB42	Closed	Cut to disconnect IF Boot/Reset button from nRF52840 reset pin when the interface MCU is disconnected
SB43	Open	Short to connect IF Boot/Reset button to RESET pin on the Arduino Interface
SB44	Open	Short to connect the RESET pin on the Arduino interface to the nRF52840 reset pin
SB45	Open	Short to connect the RESET pin on the Arduino interface to the interface nRF52840 reset pin when the interface MCU is disconnected
SB46	Open	Short to connect the RESET pin on the Arduino interface to the interface MCU Boot when the interface MCU is disconnected
SB47	Open	Short to enable power supply of the external device when using the debug out connector
SB48	Open	Short to bypass the interface MCU USB power switch
SB49	Open	Short to connect VDD_UTMI to VDD_SAM
SB50	Closed	Cut to disconnect the nRF52840 CTS line from the signal switch and interface MCU
SB51	Closed	Cut to disconnect the nRF52840 RTS line from the signal switch and interface MCU
SB52	Closed	Cut to disconnect the nRF52840 RxD line from the signal switch and the interface MCU
SB53	Closed	Cut to disconnect the nRF52840 TxD line from the signal switch and interface MCU
SB54	Closed	Cut to disconnect the nRF52840 SWDIO line from the signal switch and interface MCU
SB55	Closed	Cut to disconnect the nRF52840 SWDCLK line from the signal switch and interface MCU

SB56	Closed	Cut to disconnect the nRF52840 RESET line from the signal switch and interface MCU
SB57	Closed	Cut to disconnect the nRF52840 SWO line from the signal switch and the interface MCU
SB58	Closed	Cut to disconnect voltage follower from external supply when SW10 is in ON position
SB59	Open	Solder to connect debug in and trace reference voltage to VDD
SB60	Closed	Cut to disconnect debug in and trace reference voltage from VDD_nRF
SB65	Closed	Cut to disable the pull-up resistor of the IMCU_BOOT line
SB80	Open	Short to bypass the power switch for the VBUS of nRF52840
SB81	Open	Short to bypass the power switch for VDD_HV of nRF52840

Table 9: Solder bridge configurations

9. Glossary

The glossary contains terms and acronyms that are used in this document.

Clear to Send (CTS)

In flow control, the receiving end is ready and telling the far end to start sending.

Data Terminal Ready (DTR)

A control signal in RS-232 serial communications transmitted from data terminal equipment, such as a computer, to data communications equipment.

Development Kit (DVK)

Development platform used for application development

Hardware Flow Control (HWFC)

A handshaking mechanism used to prevent an overflow of bytes in modems. It is utilizing two dedicated pins on the RS-232 connector, Request to Send (RTS) and Clear to Send (CTS).

Integrated Development Environment (IDE)

A software application that provides facilities for software development.

Mass Storage Device (MSD)

Any storage device that makes it possible to store and port large amounts of data in a permanent and machine-readable fashion.

Near Field Communication (NFC)

A standards-based short-range wireless connectivity technology that enables two electronic devices to establish communication by bringing them close to each other.

NFC-A Listen Mode

Initial mode of an NFC Forum Device when it does not generate a carrier. The device listens for the remote field of another device.

Operational Amplifier (op-amp)

A high-gain voltage amplifier that has a differential input and, usually, a single output.

Receive Data (RXD)

A signal line in a serial interface that receives data from another device.

Request to Send (RTS)

In flow control, the transmitting end is ready and requesting the far end for a permission to transfer data.

Root Mean Square (RMS)

An RMS meter calculates the equivalent direct current (DC) value of an alternating current (AC) waveform. A true-RMS meter can accurately measure both pure waves and the more complex nonsinusoidal waves.

Subminiature Version A (SMA) Connector

A semi-precision coaxial RF connector for coaxial cables with a screw-type coupling mechanism.

System on Chip (SoC)

A microchip that integrates all the necessary electronic circuits and components of a computer or other electronic systems on a single integrated circuit.

Transmit Data (TXD)

A signal line in a serial interface that transmits data to another device.

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

iVativ:

I540M0L8-2L-DVK