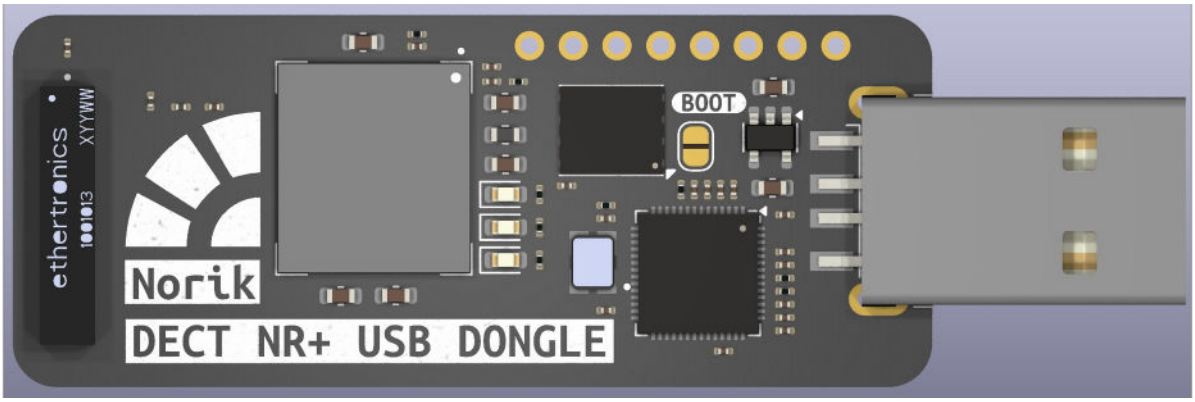
DECT NR+ USB Dongle



Overview

DECT NR+ USB Dongle is based on nRF9151 SiP. It can serve as an NR+ child node or a USB-powered gateway. It comes pre-flashed with console shell firmware, allowing for easy evaluation of DECT NR+ features. On the other hand, developers can freely re-flash the device with Zephyr examples available in the Nordic's SDKs to prototype, flash and debug DECT NR+ applications with minimal setup.

Its small form factor, built-in programmer, and full radio support make it ideal for anyone working on DECT NR+ connectivity—whether building new devices or testing deployments in the field.

Features

- **nRF9151 SiP**
  - Arm Cortex-M33 @ 64 MHz
  - 256 KB SRAM, 1 MB flash
  - DECT NR+ bands: 1, 2, 9, 22
  - Arm TrustZone + CryptoCell security
- **Built-in Debugger**
  - Raspberry Pi RP2040 as CMSIS-DAP programmer
- **Connectivity**
  - USB-A connector
  - USB-to-Serial + DFU (Device Firmware Upgrade)
  - No cables required
  - 6 user GPIOs
- **RF**
  - Onboard omnidirectional chip antenna

Contents

Recommended Operating Conditions

Parameter	MIN	TYP	MAX	UNITS
Operating Temperature	-40	25	85	°C
VDD		3.3		V

Dimensions

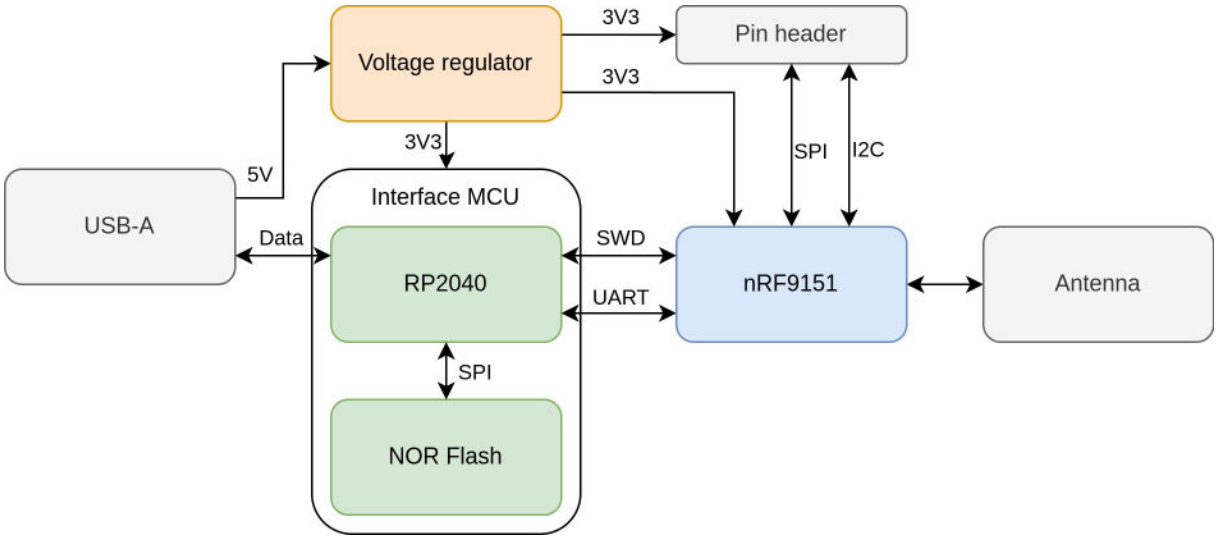
Parameter	Value
Height	21.59 mm
Width	68.55 mm

User Available Pins

#	Label	Description	Device-tree node
1	GND	Ground power output	-
2	VDD	GPIO power output	-
3	P0.10/SPI_CLK	SPI CLK pin	spi3
4	P0.08/SPI_MISO	SPI MISO pin	spi3
5	P0.09/SPI_MOSI	SPI MOSI pin	spi3
6	P0.07/SPI_CS	SPI CS pin	spi3
7	P0.06/I2C_SDA	I2C SDA pin	i2c0
8	P0.05/I2C_SCL	I2C SCL pin	i2c0

NOTE: GPIO pins are not 5V tolerant.

Block diagram



DECT NR+ Console shell

DECT NR+ USB Dongle is pre-flashed with DECT NR+ Shell sample from Nordic Semiconductor. It demonstrates how to setup a DECT NR+ applications with DECT PHY firmware. The following documentation is summary of DECT NR+ Shell sample documentation. For more information see Nordic's website.

DECT related command structure:

```
dect
      activate
      deactivate
      sett
      radio_mode
      rssi_scan
      ping
      perf
      rf_tool
      status
      rx
      mac
            beacon_scan
            beacon_start
            beacon_stop
            rach_tx
            associate
            dissociate
            status
```

Application settings

DeSh command: `dect sett`

Some of the main DeSh command parameters can be saved into settings that are persistent between sessions. The settings are stored in the persistent storage and loaded when the application starts.

Examples

See the usage and read the current settings:

```
dect sett -?
dect sett -r
```

Reset settings to their default values:

```
dect sett --reset
```

Change the default TX power:

```
dect sett --tx_pwr -16
```

Change the default band to 2:

```
dect sett --b -2
```

RSSI measurement

DeSh command: `dect rssi_scan`

Execute RSSI measurement/scan.

Examples

Execute RSSI measurement with default parameters:

```
dect rssi_scan
```

Execute longer (5000 ms on each channel) RSSI measurement on all channels on a set band:

```
dect rssi_scan -c 0 -t 5000
```

Stop RSSI measurement:

```
dect rssi_scan stop
```

Execute longer RSSI measurements on all permitted channels, and instead of default high/low RSSI value-based verdict, use subslot count-based verdict for BUSY/POSSIBLE/FREE and as an end result, see the verdict for the best channel:

```
dect rssi_scan -c 0 --verdict_type_count -t 3000 -a
```

Execute longer RSSI measurements on specific channel, and instead of default high/low RSSI value-based verdict, use subslot count-based verdict for BUSY/POSSIBLE/FREE. Additionally, print BUSY/POSSIBLE measurements:

```
dect rssi_scan -c 1661 --verdict_type_count_details -t 6500
```

Ping command

DeSh command: `dect ping`

Ping is a tool for testing the reachability of a neighbor host running with DeSh ping server. You can also use it to measure the round-trip time, demonstrate HARQ and for range testing.

Example 1: basic usage

On both client and server side – scan for a free channel:

```
dect rssi_scan -c 0
```

Server side: Set a unique transmission ID and start ping server on manually chosen free channel:

```
dect sett -t 39
dect ping -s --channel 1677
```

Client side: Start basic pinging:

```
dect ping -c --s_tx_id 39 --channel 1677
```

Client side: Max amount of data in one request with custom parameters:

```
dect ping -c --s_tx_id 39 --c_tx_pwr 7 -i 3 -t 2000 -l 4 --c_tx_mcs 4 --
c_count 10 --channel 1677
```

Server or client side: Stop ping server:

```
dect ping stop
```

DECT NR+ PHY MAC

DeSh command: `dect mac`

This command demonstrates a basic sample of the DECT NR+ MAC layer on top of PHY API. It is based on MAC specification ETSI TS 103 636-4. With this command, you can start a cluster beacon, scan for beacons, associate/dissociate, and send data to the beacon random access RX window.

This is just an initial basic sample and not a full MAC implementation. It does not support all the features of the DECT NR+ MAC layer and is not fully compliant with the MAC specification. For example, cluster beaconing with RA allocation and LMS implementations overall are not what is required by the specification.

The following abbreviations from MAC specification are used in the examples:

- FT: Fixed Termination point
- PT: Portable Termination point

Example: starting of cluster beacon and sending RA data to it

FT/Beacon device - Start periodic cluster beacon TX on default band 1 and on the first free channel:

```
desh:~$ dect sett --reset
desh:~$ dect sett -t 1234

dect common settings saved
desh:~$ dect mac beacon_start
Beacon starting
RSSI scan started.
RSSI scan duration: scan_time_ms 2010 (subslots 9648)
------------------------------------------------------------------------------
RSSI scanning results (meas #1 mdm time 2335226066):
channel                              1657
total scanning count                 201
highest RSSI                         -94
lowest RSSI                          -104
Subslot count based results:
total subslots: 9648
free subslots: 9648, possible subslots: 0, busy subslots: 0
not measured subslots: 0, saturated subslots: 0
Final verdict FREE based on SCAN_SUITABLE 75%:
   free: 100.00%, possible: 100.00%
------------------------------------------------------------------------------
…
------------------------------------------------------------------------------
RSSI scan done. Found 11 free, 0 possible and 0 busy channels.
   Best channel: 1659
   Final verdict: FREE
   Free subslots: 9648
   Possible subslots: 0
   Busy subslots: 0
Scheduled beacon TX: interval 2000ms, tx pwr 0 dbm, channel 1659, payload PDU
byte count: 50
Channel 1659 was chosen for the beacon.
Beacon TX started.
```

FT/Beacon device - Check MAC status:

```
desh:~$ dect mac status
dect-phy-mac status:
Cluster beacon status:
   Beacon running:               yes
   Beacon channel:               1659
   Beacon tx power:              0 dBm
   Beacon interval:              2000 ms
   Beacon payload PDU byte count: 50
Client status:
Neighbor list status:
```

FT/Beacon device - Check generated long and short RD IDs from settings:

```
desh:~$ dect sett -r
Common settings:
network id (32bit)............................305419896 (0x12345678)
transmitter id (long RD ID)...................1234 (0x000004d2)
short RD ID...................................27462 (0x6b46)
band number...................................1
```

PT/client side - Scan the beacon:

```
desh:~$ dect sett --reset
desh:~$ dect sett -t 1245
desh:~$ dect mac beacon_scan -c 1659
-------------------------------------------------------------------------------
Beacon scan started.
Starting RX: channel 1659, rssi_level 0, duration 4 secs.
-------------------------------------------------------------------------------
…
-------------------------------------------------------------------------------
RX DONE.
```

PT/client side - As an alternative to the previous command, and if you do not know in which channel the beacon is running, you can scan all channels in a set band:

```
desh:~$ dect mac beacon_scan -c 0
```

PT/client side - Check that the scanned beacon is found from neighbor list:

```
desh:~$ dect mac status
dect-phy-mac status:
Cluster beacon status:
   Beacon running: no
Client status:
Neighbor list status:
   Neighbor 1:
      network ID (24bit MSB): 1193046 (0x123456)
      network ID (8bit LSB):  120 (0x78)
      network ID (32bit):     305419896 (0x12345678)
      long RD ID:             1234
      short RD ID:            27462
      channel:                1659
      Last seen:              12964 msecs ago
      Background scan status:
         Running: false
```

PT/client side - Send association request to the scanned beacon:

```
desh:~$ dect mac associate -t 1234
Sending association_req to FT 1234's random access resource
Scheduled random access data TX/RX:
…
RX for Association Response completed
```

PT/client side - Check that the client is associated with the device with long RD ID 1234 and a background scan is running:

```
desh:~$ dect mac status
dect-phy-mac status:
Cluster beacon status:
   Beacon running: no
Client status:
   Association #1: long RD ID 1234
Neighbor list status:
   Neighbor 1:
      network ID (24bit MSB): 1193046 (0x123456)
      network ID (8bit LSB):  120 (0x78)
```

```
      network ID (32bit):     305419896 (0x12345678)
      long RD ID:             1234
      short RD ID:            64945
      channel:                1671
      Last seen:              1428 msecs ago
      Last timestamp: 16878096625 mdm ticks
      Metrics:
         Scan info updated count:            1
         Scan started ok count:              1
         Scan start fail count:              0
         Scan info time shift updated count: 0
         Scan info time shift last value:    0
```

PT/client side - Send RA data to the scanned beacon:

```
desh:~$ dect mac rach_tx -t 1234 -d "TAPPARA!"
Sending data TAPPARA! to FT 1234's random access resource
Scheduled random access data TX:
target long rd id 1234 (0x000004d2), short rd id 27462 (0x6b46),
target 32bit nw id 305419896 (0x12345678), tx pwr 0 dbm,
channel 1659, payload PDU byte count: 17,
beacon interval 2000, frame time 29319898225, beacon received 17016336625
Client TX to RACH started.
Client data TX completed.
```

FT/Beacon device - Observe that data was received:

```
PCC received (stf start time 32017011258): status: "valid - PDC can be
received", snr 91, RSSI-2 -123 (RSSI -61)
phy header: short nw id 120 (0x78), transmitter id 27761
receiver id: 27462
len 0, MCS 0, TX pwr: 0 dBm
PDC received (stf start time 32017011258): snr 98, RSSI-2 -123 (RSSI -61), len
17
DECT NR+ MAC PDU:
MAC header:
   Version: 0
   Security: MAC security is not used
   Type: DATA MAC PDU header
      Reset: yes
      Seq Nbr: 1
SDU 1:
   MAC MUX header:
      IE type: User plane data - flow 1
      Payload length: 10
      DLC IE type: Data: DLC Service type 0 without a routing header (0x01)
      Received data, len 9, payload as ascii string print:
         TAPPARA!
SDU 2:
   MAC MUX header:
      IE type: Padding (0 byte)
      Payload length: 0
      Received padding data, len 0, payload is not printed
```

PT/client side - Send JSON-formatted periodic RA data in 10-second intervals with the current modem temperature to the scanned beacon:

```
desh:~$ dect mac rach_tx -t 1234 -d "Data from device 1245" -i 10 -j
```

PT/client side - Stop periodic RA data sending:

```
desh:~$ dect mac rach_tx stop
```

PT/client side - Send association release to the scanned beacon:

```
desh:~$ dect mac dissociate -t 1234
Sending association release to FT 1234's random access resource
Scheduled random access data TX/RX:
target long rd id 1234 (0x000004d2), short rd id 27462 (0x6b46),
target 32bit nw id 305419896 (0x12345678), tx pwr 0 dbm,
channel 1659, payload PDU byte count: 17,
beacon interval 2000, frame time 34434778225, beacon received 17016336625
Association Release TX started.
TX for Association Release completed.
```

FT/Beacon device - Observe that the association release message was received:

```
PCC received (stf start time 37131891398): status: "valid - PDC can be
received", snr 93, RSSI-2 -123 (RSSI -61)
phy header: short nw id 120 (0x78), transmitter id 27761
receiver id: 27462
len 0, MCS 0, TX pwr: 0 dBm
PDC received (stf start time 37131891398): snr 94, RSSI-2 -123 (RSSI -61), len
17
DECT NR+ MAC PDU:
MAC header:
   Version: 0
   Security: MAC security is not used
   Type: Unicast Header
      Reset: yes
      Seq Nbr: 2
      Receiver: 1234 (0x000004d2)
      Transmitter: 38 (0x00000026)
SDU 1:
   MAC MUX header:
      IE type: Association Release message
      Payload length: 1
      Received Association Release message:
      Release Cause:  Connection termination. (value: 0)
SDU 2:
   MAC MUX header:
      IE type: Padding
      Payload length: 1
      Received padding data, len 1, payload is not printed
```

PT/client side - Check that no associations and background scans are running:

```
desh:~$ dect mac status
```

FT/Beacon device - Stop the beacon:

```
desh:~$ dect mac beacon_stop
Stopping beacon.
Beacon TX stopped, cause: User Initiated.
```

Example: two devices sending data to each other

FT/Beacon device 1 - Start periodic cluster beacon TX on default band 1 and on the first free channel:

```
desh:~$ dect sett --reset
desh:~$ dect sett -t 1
desh:~$ dect mac beacon_start
...
Channel 1671 was chosen for the beacon.
Beacon TX started.
```

FT/Beacon device 2 - Scan the beacon from device 1:

```
desh:~$ dect sett --reset
desh:~$ dect sett -t 2
desh:~$ dect mac beacon_scan -c 1671
```

FT/Beacon device 2 - Start periodic cluster beacon TX on default band 1 and on the first free channel:

```
desh:~$ dect mac beacon_start
...
Channel 1675 was chosen for the beacon.
Beacon TX started.
```

FT/Beacon device 1 - Scan the beacon from device 2 by using special force:

```
desh:~$ dect mac beacon_scan -c 1675 -f
```

FT/Beacon device 1 - Send association request to device 2:

```
desh:~$ dect mac associate -t 2
```

FT/Beacon device 2 - Send association request to device 1:

```
desh:~$ dect mac associate -t 1
```

FT/Beacon device 1 - Send JSON-formatted periodic RA data in 10-second intervals with the current modem temperature to the device 2:

```
desh:~$ dect mac rach_tx -t 2 -d "Data from device 1" -i 10 -j
```

FT/Beacon device 2 - Send JSON-formatted periodic RA data in 10-second intervals with the current modem temperature to the device 1:

```
desh:~$ dect mac rach_tx -t 1 -d "Data from device 2" -i 10 -j
```

FT/Beacon devices 1 & 2 - Stop periodic RA data sending:

```
desh:~$ dect mac rach_tx stop
```

FT/Beacon device 1 - Send association release to the device 2:

```
desh:~$ dect mac dissociate -t 2
```

FT/Beacon device 2 - Send association release to the device 1:

```
desh:~$ dect mac dissociate -t 1
```

FT/Beacon devices 1 & 2 - Stop the beacon:

```
desh:~$ dect mac beacon_stop
```

Revision history

Revision number	Date	Changes
1.0	22.4.2025	Initial release