**System Engineering document to keep track of progress, changes and implementations**

Now working in the nRF connect SDK version 2.6.1

V2.6.1 supports Bootloader, DFU and FOTA.

UART mcubootloader was tested and confirmed to work, RS485 bootloader was not functional, attempts were made to edit the source code but no progress was made.

BLE Bootloader and DFU OTA was tested and confirmed to work. This method was chosen to use on both versions of the probe to improve efficiency and ease of use.

Lesson8 exercise1, was loaded onto the DK and a test program blinkyplain was built and the app\_update.bin file loaded onto my phone and sent to the board using the nRF device manager application, this worked and the updated application was updated and ran.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No progress |  | Basic Functionality |  | 100% Working |
|  |  |  |  |  |

|  |  |
| --- | --- |
| Item | Progress |
| Bootloader(BLE) | Working |
| UART | Working |
| Counter | Working |
| ADC | Working |
| Internal Temp Sensor | Working |
| BLE Manufacturers advertising packet | Working |
| Timers | Working |
| Power management |  |
| External HFclock | Busy…… |
| RTC | Working |
| PPI | Working |

Next steps:

Work through the new SDK exercises on the website

Turn Bluetooth off, then switch it on with a button and see if I can connect to it and the download a new version of code.(blinky+BLE not working)(peripheral\_uart+onOff)

Testing:

Make a program that is different to the FOTA DFU example, but that is still Bluetooth capable, and that can then be updated with another image, so basically two updates over the air. Done and done. Interesting rollbacks and protections, need to figure out how to swap between the two images.

|  |
| --- |
|  |
| Next step, used the sample BLE\_Peripheral Uart example and added in these 2 lines of configuration into the prj.conf file. Both codes were able to become the default image used and not revert after cycling the power. |
| CONFIG\_BOOTLOADER\_MCU=y  CONFIG\_NCS\_SAMPLE\_MCUMGR\_BT\_OTA\_DFU=y |

|  |  |
| --- | --- |
| Option | Outcome |
| Test and Confirm | Bootloader temporarily runs the new firmware image, and if it is successful marks it as confirmed, meaning the next time the bootloader is called if it is marked as confirmed it will be run. |
| Test only | Bootloader temporarily runs the new firmware image, when it is reset, it reverts back to the original firmware loaded onto the device. |
| Confirm only | The new firmware image is immediately confirmed without testing it and becomes the default on the next reboot. |
| No revert | Basically the same as above. |

Make a base layer code version: BLE on and off + Logging to the serial monitor

End requirements of the two versions wrapped into one: 6-level ADC & 6-level counter reading plus battery reading, plus RS485 communication/special character to get . Battery reading, plus internal temperature reading, plus counter reading, periodically advertised on BLE using the manufacturers packet.

Th3IronByron25Work

|  |  |
| --- | --- |
| Bootloader different terminal commands |  |
| **West** |  |
| west build -b nrf52dk\_nrf52832 |  |
| west build -t pristine |  |
| west flash |  |
|  |  |
| **Mcumgr** |  |
| mcumgr conn add testRS485 type=serial connstring="COMX,baud=115200,mtu=512" |  |
| mcumgr -c testRS485 image list |  |
| mcumgr -c testRS485 image upload build/zephyr/app\_update.bin |  |
| mcumgr -c testRS485 reset |  |

Tips and tricks while working with nRF connect SDK

|  |
| --- |
| Remember to disable cts to free up pin 0.07 |
| Do not use kprint to print to the serial as this is a blocking, rather use the logging function |
|  |
| Must still activate the external high frequency clock |
| Must use PPI to use purely hardware to count |
| Must use the internal temperature sensor |

Asynchronous UART

|  |
| --- |
| Add the following into prj.conf  CONFIG\_SERIAL=y  CONFIG\_UART\_ASYNC\_API=y |
| Header file include in the source code  #include <zephyr/drivers/uart.h> |
|  |
|  |
| CONFIG\_UART\_0\_ASYNC=y  CONFIG\_UART\_0\_NRF\_HW\_ASYNC=y  CONFIG\_UART\_0\_NRF\_HW\_ASYNC\_TIMER=2  CONFIG\_NRFX\_TIMER2=y |
| To change the Hardware flow control pins, I set them in the visual device tree manager to pins 9 and 10 which are unused, this allowed pin 7 to be used for DIR control for the RS485 implementation |
| fedf |

Setting up logging

|  |  |
| --- | --- |
| #include <zephyr/logging/log.h> | |
| CONFIG\_LOG=y  CONFIG\_SERIAL=y  CONFIG\_LOG=y  CONFIG\_LOG\_MODE\_IMMEDIATE=y | Add into prj.conf file |
| LOG\_MODULE\_REGISTER(booran, LOG\_LEVEL\_DBG); | |
|  | |

SAADC

|  |  |
| --- | --- |
| Typical sample time is 2us | Can use two reference voltages:  Internal reference  And VDD as reference |
|  |  |
|  |  |
|  | |
|  | |
|  | |

Timers

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

Add a counter

|  |  |
| --- | --- |
| Add this into prj.conf:  CONFIG\_GPIO=n  CONFIG\_LOG=y  CONFIG\_LOG\_PROCESS\_THREAD\_SLEEP\_MS=100  CONFIG\_NRFX\_TIMER1=y  CONFIG\_NRFX\_TIMER2=y  CONFIG\_LOG\_PROCESS\_THREAD=y |  |
|  |  |
|  |  |

Add Bootloader(BLE) //nRF Connect SDK build files

|  |  |
| --- | --- |
| Add these into the prj.conf file:  CONFIG\_NCS\_SAMPLES\_DEFAULTS=y  CONFIG\_BT=y  CONFIG\_BT\_PERIPHERAL=y  CONFIG\_BT\_DEVICE\_NAME="Gentick\_RS+Boot"  CONFIG\_BOOTLOADER\_MCUBOOT=y  CONFIG\_NCS\_SAMPLE\_MCUMGR\_BT\_OTA\_DFU=y  CONFIG\_GPIO=y  CONFIG\_BOARD\_ENABLE\_DCDC=n  CONFIG\_RESET\_ON\_FATAL\_ERROR=y  CONFIG\_GPIO\_AS\_PINRESET=n  CONFIG\_CLOCK\_CONTROL\_NRF\_K32SRC\_RC=y  CONFIG\_CLOCK\_CONTROL\_NRF\_K32SRC\_XTAL=n  CONFIG\_CLOCK\_CONTROL\_NRF\_K32SRC\_RC\_CALIBRATION=y  CONFIG\_CLOCK\_CONTROL\_NRF\_K32SRC\_500PPM=y  CONFIG\_SERIAL=y  CONFIG\_UART\_ASYNC\_API=y  CONFIG\_UART\_0\_ASYNC=y  CONFIG\_UART\_0\_NRF\_HW\_ASYNC=y  CONFIG\_UART\_0\_NRF\_HW\_ASYNC\_TIMER=2  CONFIG\_NRFX\_TIMER2=y # I think this is optional  CONFIG\_LOG=y  CONFIG\_LOG\_MODE\_IMMEDIATE=y |  |
| Add these includes into the main.c:  #include <zephyr/types.h>  #include <stddef.h>  #include <string.h>  #include <zephyr/sys/printk.h>  #include <zephyr/kernel.h>  #include <zephyr/bluetooth/bluetooth.h>  #include <zephyr/bluetooth/hci.h>  #include <zephyr/bluetooth/conn.h>  #include <zephyr/settings/settings.h> |  |
| Add these files into the project:  child\_image  Kconfig  Kconfig.sysbuild  nrf52dk\_nrf52832.overlay |  |
|  |  |

Future improvements:

|  |  |
| --- | --- |
| DFU FOTA:  Understand why the new image is not being confirmed.  Current fix:  Test application and then force confirm for new software versions. |  |
| TIMER for Counters:  Not initializing when used in conjunction with the BLE stack.  Current fix:  Use the k-timer (kernel timers). |  |
| HFXO:  BLE not releasing control of the external high freq clock  Current fix:  Nothing |  |
|  |  |

Github useful resources

|  |  |
| --- | --- |
| Ncs-nrfx-timer | <https://github.com/too1/ncs-nrfx-timer-example> |
| Ncs-ppi-counter  NOT WORKING!!!!! | <https://github.com/too1/ncs-nrfx-pulse-count-example> |
| Instructions on PPI v2.6.1 | <https://developer.nordicsemi.com/nRF_Connect_SDK/doc-legacy/2.6.1/nrf/libraries/debug/ppi_trace.html> |
|  |  |
| Storing data on nvs ect | <https://www.youtube.com/watch?v=Np8O-MjAeGU> |
| Youtube video that showed how to setup a GPIO pin | <https://www.youtube.com/watch?v=rvDdev18PAI> |
|  |  |

24/01/2025

Can load the new code onto the PCB boards, not sure if the code is functional, need to create a custom build configuration to load the relevant code onto it.

28/01/2025

Got the new nRF connect SDK v2.6.1 code to run on the PCB, it was the DCDC configuration in the prj.conf file, similar story to get the BLE working, and then to get the bootloader and FOTA to work using v2.6.1 on the PCBs, a child image file was needed with the DCDC configuration enabled aswell.

29/01/2025

Got the PCB to run mcuboot code with ble, that uses uart to switch the ble back on and allow FOTA, and the process can just keep being repeated.

30/01/2025

Got the RTC2 working, although I am stuck using the 0,125ms ticks, this is due to the fact that the prescaler is a 12-bit value meaning the max value it can be is 4096 which equates to 0,125ms ticks.

Got the internal temperature sensor working.

Got BLE manufacturers data to broadcast.

Got multiple SAADC channels reading, using the Zephyr API.

31/01/2025

Got all 7 ADC channels working for the large probe. Got the 7 ADC readings using functions and streamlined.

Got the HFXCLK active and the counter using pin 0.05 accurately.

06/02/2025

Checked all 6 counters on the PCB and they were verified to work correctly, it was tested using the code Boot+RS485+Cnt.

Found bug with the BarebonesBoot, it would revert after a reset. Will investigate why and rectify the error.

07/02/2025

Reset when switching enable pins high, brownout failure.

11/02/2025

Fixed the brownout issue by adding a 470uF capacitor to the 3V3 regulator.

17/02/2025

Went back to UART\_RS485 and it worked, then pasted in the UART new method code and that worked, now it is all working. RS485 with the roman still drops some communications, checking on termite now, but seems to be sending the correct reply message(Not sure the cause of the problem yet).

18/02/2025

Fixed the UART problem, was timing of the readings

Got the small probe taking readings and advertising over uart