Tactile Cube

IDE setup, programming, getting everything running

# Programming cable

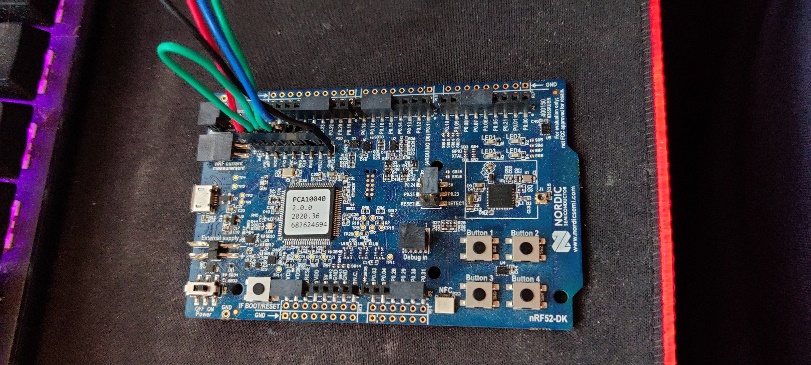
As mentioned in the electronics section, this project uses direct access to the microcontroller flash memory. This was done so I could avoid the boot loading process which would take ages with the rapid versions during development and this type of programming gives access to one of the greatest tools in programming: a debugger.

Figure 1 Location of the programming pins on the development kit

# Setting up the environment

The IDE used was Segger Embedded Studio with the free Nordic license.

## Getting the license

<https://license.segger.com/Nordic.cgi>

Fill out the form of this link, and a code will be generated. This code is necessary later when the IDE is installed.

## Development Environment

<https://www.segger.com/products/development-tools/embedded-studio/>

Install this software. Latest version should be fine.

When running this IDE it will ask for a license, you can install the Nordic License you got.

## Setting up the project

Due to long paths, to get the best experience from SEGGER studio, copy the project folder into the root of partitions. The project folder in this documentation should run, even the shortcut file if copied into C://.

If the shortcut does not work, you can still read where it points to understand where to find the file that opens the project with the SEGGER STUDIO.

The project folder is also a git repository that does not save the SDK, only the parts of the project that matters. **Try not to overwrite SDK files! It will make the git repository does not save those changes, nor will it be portable.**

# Programming

Make sure the programming cables are correct!!! Nothing should go wrong if the connectors are not connected correctly apart from the JTAG not working, but still to avoid possible shorts and magic smoke escapees verify connections.

The current version of the code should compile as it is. F7 builds the code. To program the microcontroller either do a debugging, or Target/Download. Please note you might have to manually restart the microcontroller, so a power cycling, or connecting the JTAG and issuing a reset command might be used as well.

# Improvements

There are so many improvements that can be done here:

## Game/Demo:

* Add more functionalities to the cube like simple sensor testing or visualisation modes, maybe a simple decoration mode that are changeable through BLE/USB.
* Add more difficulty to the demo in utilising the accelerometer and gyroscope
* Add logging to the BLE and/or USB about what the users are doing for an accompanying application for phone/computer
* Make an accompanying application that keeps track of high scores, allows for on the go changes of the settings etc.
* Add the ability to skip the sequence playback animation when the user just lost a life.
* Display already known elements faster since after 10 steps it gets too slow. Only prolong the last step.
* Sanitize BLE data to disallow illegal inputs.
* BLE display current remaining lives. USB poll command.

# Firmware:

* Add a bootloader (Nordic has good documentations on it)
* Add a command to recalibration over BLE/USB without having to power cycle the cube
* Test if the heat is the cause of the change in magnetic recordings from the LED.
* LED flickering might be a programming erros, since seemingly only the first bits of the colour codes seem to go wrong. (not applicable if the used LEDs will be APA102 LEDS)

## Power monitoring

Provided means are added to measure the battery voltage, temperature and states of the battery management circuit.

* Detect charging, battery voltage, temperature
* It is not advised to turn off the cube and plug in the USB power. Detect if the USB is plugged in but not the batteries to turn power hungry elements off. Since I do not think the microcontroller’s LDO is strong enough
* Add a low power flickering to the cube telling the user to charge the cube.
* Display battery data over the BLE and add polling commands to the USB