UM2249

User manual

# Getting started with the STSW-BCNKT01 software package for STEVAL-BCNKT01V1 based on STM32Cube

## Introduction

The STSW-BCNKT01 firmware package for BlueCoin Starter kit provides sample projects for the development of custom applications.

Built on STM32Cube software technology, it includes all the low level drivers to manage the on-board devices and system-level interfaces.

The package comes with the Audio_SD, DataLog, AudioLoop, BLE_SampleApp and Gesture Recognition applications.

The Audio_SD application allows saving the audio captured by the on-board microphones on SD card as a common .wav file.

The DataLog application features sensor raw data streaming via USB (Virtual COM Port class) and sensor data storage on SD card.

The BLE_SampleApp provides a sample Bluetooth low energy configuration that enables BlueCoin to stream environmental sensor data; it is compatible with the STBLESensor app available for Android and iOS.

The Gesture Recognition application exploits the Time-of-Flight ranging sensors to detect the distance of a target object, and some simple gestures such as directional swipe and tap.

# 1 What is STM32Cube?

STM32Cube™ represents the STMicroelectronics initiative to make developers' lives easier by reducing development effort, time and cost. STM32Cube covers the STM32 portfolio.

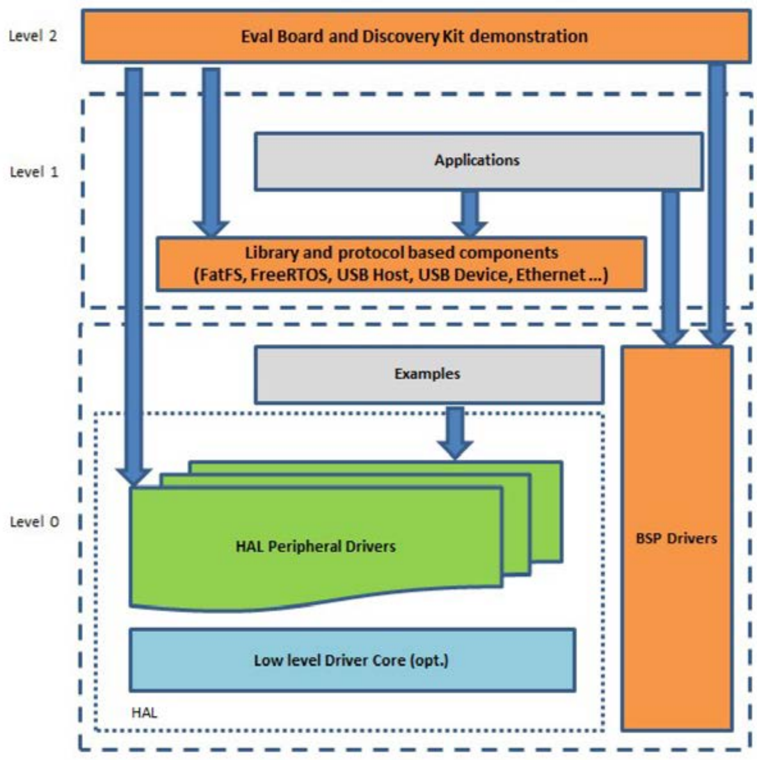
STM32Cube version 1.x includes:

- STM32CubeMX, a graphical software configuration tool that allows the generation of C initialization code using graphical wizards.
- A comprehensive embedded software platform specific to each series (such as the STM32CubeF4 for the STM32F4 series), which includes:
  - the STM32Cube HAL embedded abstraction-layer software, ensuring maximized portability across the STM32 portfolio
  - a consistent set of middleware components such as RTOS, USB, TCP/IP and graphics
  - all embedded software utilities with a full set of examples

## 1.1 STM32Cube architecture

The STM32Cube firmware solution is built around three independent levels that can easily interact with one another, as described in the diagram below.

Figure 1. Firmware architecture


Level 2
Eval Board and Discovery Kit demonstration
Level 1
Applications
Library and protocol based components
(FatFS, FreeRTOS, USB Host, USB Device, Ethernet ...)
Examples
Level 0
HAL Peripheral Drivers
BSP Drivers
Low level Driver Core (opt.)
HAL

**Level 0**: This level is divided into three sub-layers:

- Board Support Package (BSP): this layer offers a set of APIs relative to the hardware components in the hardware boards (Audio codec, IO expander, Touchscreen, SRAM driver, LCD drivers. etc…); it is based on modular architecture allowing it to be easily ported on any hardware by just implementing the low level routines. It is composed of two parts:

- Component: is the driver relative to the external device on the board and not related to the STM32, the component driver provides specific APIs to the external components of the BSP driver, and can be ported on any other board.
- BSP driver: links the component driver to a specific board and provides a set of easy to use APIs. The API naming convention is BSP_FUNCT_Action(): e.g., BSP_LED_Init(), BSP_LED_On().

- Hardware Abstraction Layer (HAL): this layer provides the low level drivers and the hardware interfacing methods to interact with the upper layers (application, libraries and stacks). It provides generic, multi-instance and function-oriented APIs to help offload user application development time by providing ready to use processes. For example, for the communication peripherals (I²C, UART, etc.) it provides APIs for peripheral initialization and configuration, data transfer management based on polling, interrupt or DMA processes, and communication error management. The HAL Drivers APIs are split in two categories: generic APIs providing common, generic functions to all the STM32 series and extension APIs which provide special, customized functions for a specific family or a specific part number.
- Basic peripheral usage examples: this layer houses the examples built around the STM32 peripherals using the HAL and BSP resources only.

**Level 1**: This level is divided into two sub-layers:

- Middleware components: set of libraries covering USB Host and Device Libraries, STemWin, FreeRTOS, FatFS, LwIP, and PolarSSL. Horizontal interaction among the components in this layer is performed directly by calling the feature APIs, while vertical interaction with low-level drivers is managed by specific callbacks and static macros implemented in the library system call interface. For example, FatFs implements the disk I/O driver to access a microSD drive or USB Mass Storage Class.
- Examples based on the middleware components: each middleware component comes with one or more examples (or applications) showing how to use it. Integration examples that use several middleware components are provided as well.

**Level 2**: This level is a single layer with a global, real-time and graphical demonstration based on the middleware service layer, the low level abstraction layer and basic peripheral usage applications for board-based functions.

# 2 STSW-BCNKT01 software expansion for STM32Cube

## 2.1 Overview

This software package expands the functionality of the STM32Cube platform.

The key features of the package are:

- Complete firmware suite to build applications on BlueCoin development kit using:
  - STM32F446 high-performance microcontroller
  - motion sensors
  - microphone array
  - pressure sensor
  - temperature sensor
  - proximity sensors
- Based on STM32Cube, the consistent and complete embedded software for STM32 that maximizes portability between all STM32 series and frees the user from dependency issues
- An Audio_SD application which allows saving the audio captured by the on-board microphone on SD card as a common .wav file
- A DataLog application which allows the real-time transmission of all sensor data to a PC via serial port or to save/log sensor data to file on an SD card
- An AudioLoop application which sends audio signals acquired by the microphones to USB (Audio IN Class) and to an on-board DAC via I²S interface
- A Gesture Recognition application to configure Time-of-Flight ranging sensors and gesture detection middleware.
- A BLE_SampleApp which provides an example of Bluetooth low energy configuration
- A third party FAT file system middleware for small embedded systems
- Freely available in source code from www.st.com

This software enables data acquisition from different sensors like motion sensors, environmental sensors, and audio sensors via I²C or I²S.

Exploiting the capabilities of an included VCP USB driver, the device is recognized as a Virtual COM Port by Microsoft Windows or any Unix-like system.

You can also save the data on an SD card if plugged to the relevant connector.

## 2.2 Architecture

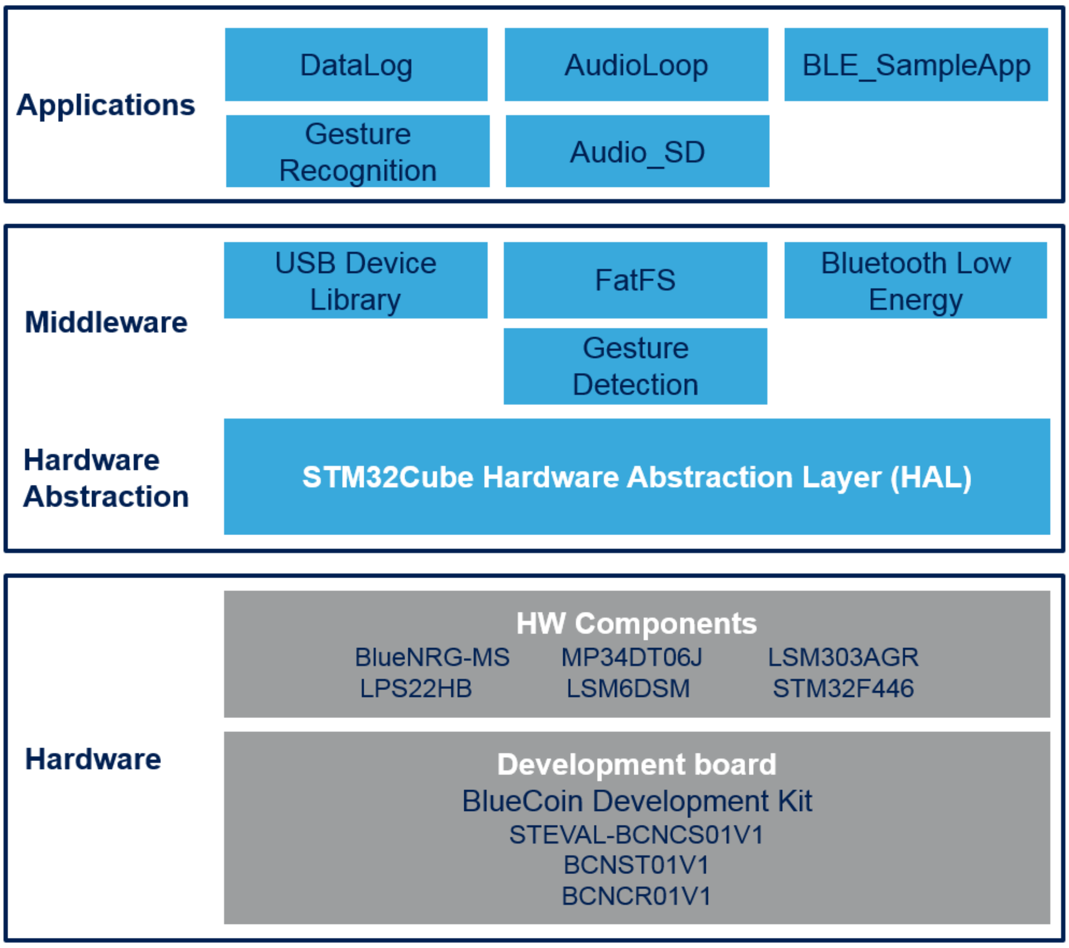
This software is a fully compliant expansion for STM32Cube enabling development of applications using digital MEMS microphones.

The software is based on the STM32CubeHAL hardware abstraction layer for the STM32 microcontroller and extends STM32Cube with a board support package (BSP) for the microphones expansion board and some middleware components for audio processing and USB communication with a PC.

The software layers used by the application software to access and use the microphone expansion board are:

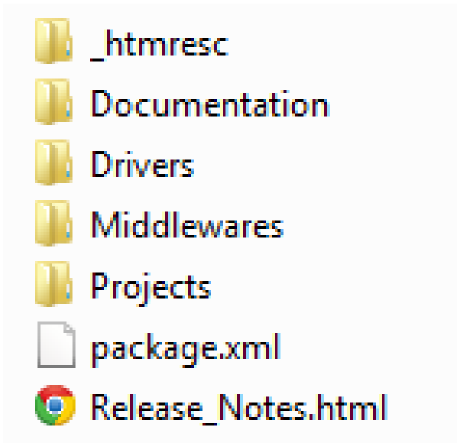
- **STM32Cube HAL layer**: provides a generic, multi-instance, simple set of APIs (application programming interfaces) to interact with the upper layers (application, libraries and stacks). It is composed of generic and extension APIs. It is built directly around a generic architecture and allows layers built on it (like the middleware layer) to implement their functions without depending on specific hardware configurations for a given microcontroller unit (MCU). This structure improves library code reusability and guarantees an easy portability on other devices.
- **Board support package (BSP) layer**: contains the software to support the STM32 Nucleo board peripherals apart from the MCU.

Figure 2. **STSW-BCNKT01 software architecture**



## 2.3 Folder structure

Figure 3. **STSW-BCNKT01 package folder structure**


_htmresc
Documentation
Drivers
Middlewares
Projects
package.xml
Release_Notes.html

The following folders are included in the software package:

- Documentation: contains a compiled HTML file generated from the source code, detailing the software components and APIs.

- Drivers: contains the board-specific HAL drivers for each supported board or hardware platform, including the on-board components and the CMSIS vendor-independent hardware abstraction layer for the ARM Cortex-M processor series.
- Middlewares: contains ST libraries for Virtual COM Port USB driver and the third party library FatFS.
- Projects: contains DataLog and AudioLoop sample applications, which can be evaluated through the IAR Embedded Workbench for ARM (IAR-EWARM), RealView Microcontroller Development Kit (MDK-ARM-STM32) and System Workbench for STM32 (SW4STM32) development environments.

## 2.4 APIs

Detailed technical information fully describing the APIs available to the user can be found in a compiled HTML file located inside the software package Documentation folder.

## 2.5 STSW-BCNKT01 software expansion applications

### 2.5.1 Audio_SD

The Audio_SD application allows saving the audio captured by the on-board microphone on SD card as a common .wav file.

After reset, the firmware:

- configures HAL and clocks
- configures SD card access, based on FatFS middleware
- configures sensors and microphone
- starts audio acquisition

To open the file and start saving the audio file, you have to push the SW2 button on the BlueCoin: the LED blinking means the .wav file has been created and the recording has been initialized.

To stop the acquisition, you have to push the BlueCoin button once again.

The LED stops blinking, that is the *SensorTile_Log_N000.wav* audio file has been correctly saved on the SD card.

You can now repeat the operations and restart the data logging on another file (*BlueCoin_Log_N001.wav*).

### 2.5.2 The DataLog

The DataLog application has two operating modes that can be selected at compile time by changing the `LoggingInterface` variable in main.c

- LoggingInterface = USB_Datalog: sensors raw data streaming via USB (Virtual COM Port class)
- LoggingInterface = SDCARD_Datalog: sensors raw data storage on SD card.

After reset, the firmware:

1. configures HAL and clocks
2. initializes the USB peripheral or the SDIO for SD card access
3. creates the threads and activate FreeRTOS scheduler

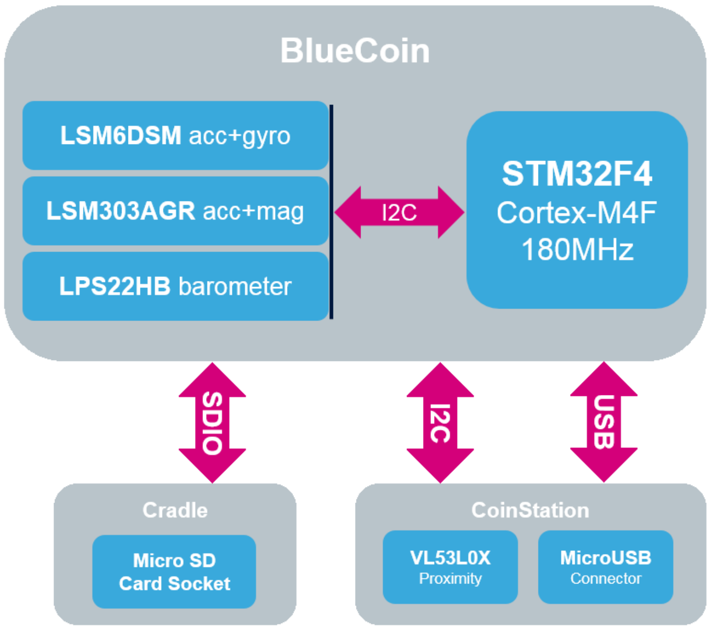
The GetData_Thread and WriteData_Thread threads are scheduled by FreeRTOS with different priorities and communicate with each other through a message queue:

- GetData_Thread: a high priority task that configures the sensors, reads data at a given frequency and pushes the new data in the queue. An OS timer triggers the execution of the thread at a given frequency.
- WriteData_Thread: a low priority task that configures the SD card and writes the sensor data as soon as they are available in the queue.

These priority differences ensure that the application does not lose sensor data even when writing operations on the SD Card take longer than the sampling period. If this happens, the WriteData_Thread is suspended by the scheduler to allow the execution of the GetData_Thread with the correct timing.

If SD card mode is selected, a double-tap on the board is needed to start logging data to BlueCoin_Log_N000.tsv and another double-tap to stop. Subsequent operations restarts data logging to a new file (e.g., BlueCoin_Log_N001.tsv).

**Figure 4. DataLog application block diagram**



### 2.5.3 AudioLoop

The AudioLoop application sends audio signals acquired by the microphone via I²S and USB interfaces, allowing the user to play these sounds on speakers or headphones, or record them on a host PC.

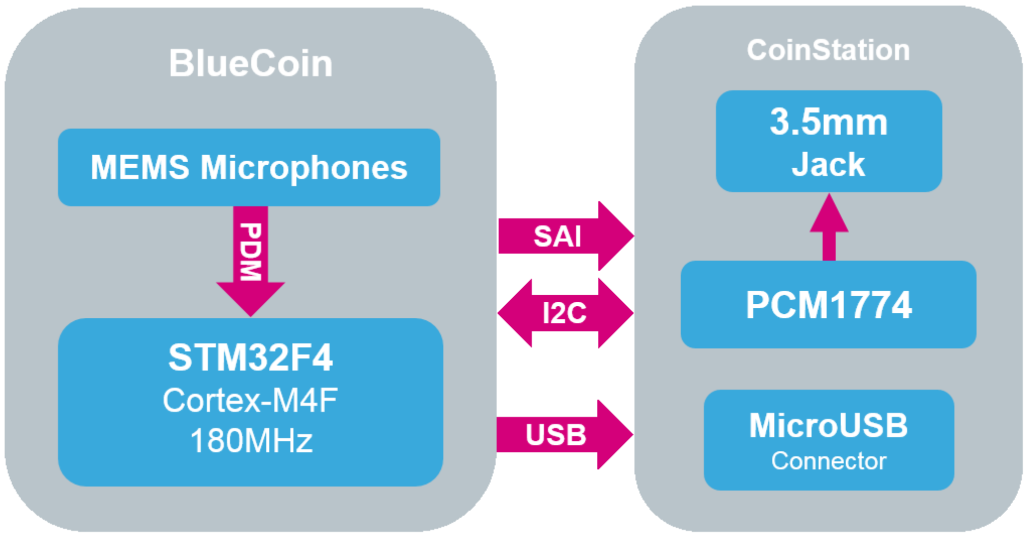
Following reset, the firmware:

1. configures HAL and clocks
2. configures LED1
3. initializes the USB peripheral
4. configures the STM32 serial audio interface peripheral (SAI) in I²S mode and the external DAC
5. configures PDM to PCM algorithm; the BSP_AUDIO_IN_ClockConfig function (defined as weak in the BSP) is redefined as an empty function in main.c as the clock and PLL configuration has already been completed in the BSP_AUDIO_OUT_Init function.
6. starts audio acquisition

In this application the main loop is empty as all the operations needed to copy the audio stream acquired from the microphone to the serial audio interface are executed in the DMA interrupt.

For this reason, the `AudioProcess()` function is called by `BSP_AUDIO_IN_TransferComplete_CallBack()` and `BSP_AUDIO_IN_HalfTransfer_CallBack()`.

Figure 5. **AudioLoop application diagram**



### 2.5.3.1 *Microphone acquisition process*

A digital MEMS microphone can be acquired via different peripherals like SPI, I²S, GPIO. It requires an input clock and it outputs a PDM stream at the same frequency of the input clock. This PDM stream is further filtered and decimated for conversion into PCM standard for audio transmission.

Two different digital MEMS microphones can be connected on the same data line, configuring the first to generate valid data on the rising edge of the clock and the other on the falling edge, by setting the L/R pin of each microphone accordingly.

On the BlueCoin (STEVAL-BCNCS01V1), the microphone output signals are acquired via I²S peripheral, which generates the precise clock needed and reads the PDM signal on the clock line rising and falling edge.

The acquired signal is then sent from PDM to PCM algorithm to generate a standard PCM stream.

An additional software high pass filtering stage removes any DC offset in the output stream.

DMA is adopted to reduce the MCU load.

## 2.5.4 BLE_SampleApp

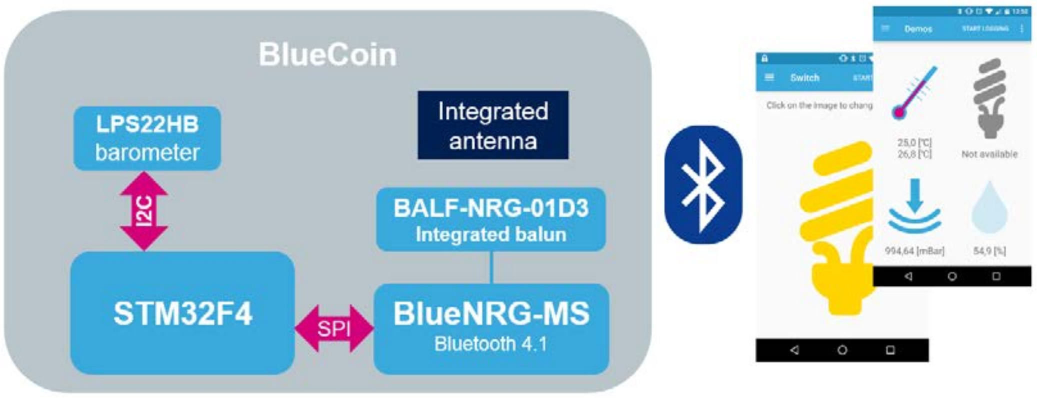
The BLE_SampleApp provides an example of Bluetooth Low Energy configuration that enables the BlueCoin to stream environmental sensor data.

It is compatible with the STBLESensor app available for Android and iOS.

After resetting, the firmware:

- configures HAL and clocks
- configures and disables sensor chip select pins
- initializes the target platform:
  - USB peripheral (for debugging)
  - LED1
  - environmental sensors
- initializes the Bluetooth Low Energy stack
- initializes the Bluetooth Low Energy services
- initializes timers
- starts the main loop:
  - LED management
  - BLE event management
  - environmental sensors data management

Figure 6. **BLE_SampleApp application diagram**



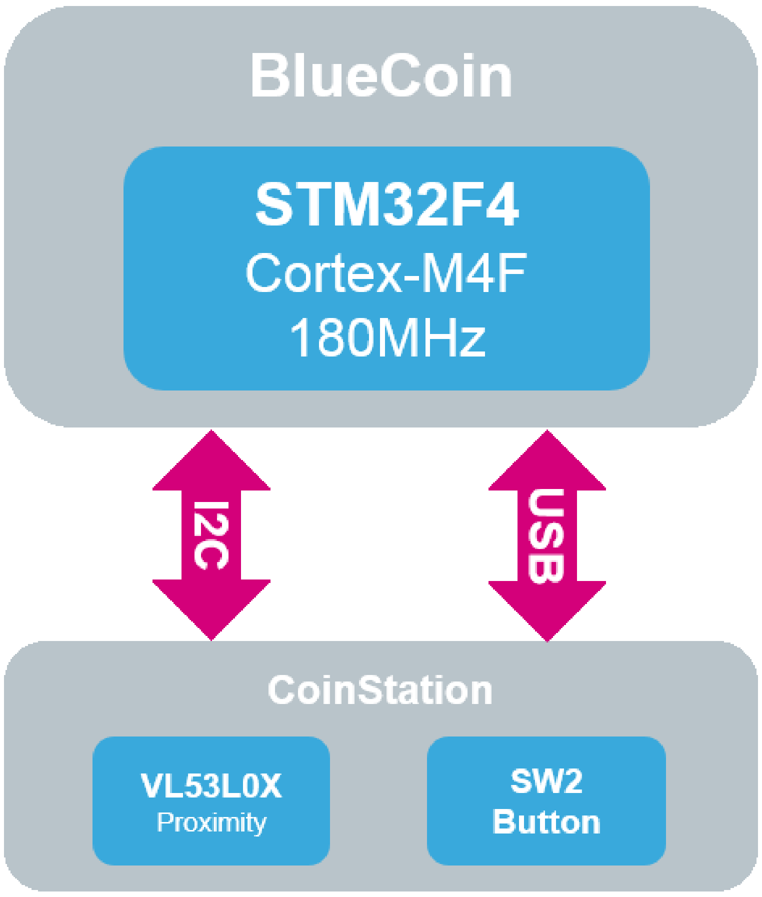
## 2.5.5 Gesture Recognition

The Gesture Recognition application exploits the Time-of-Flight sensors to provide an example of a very accurate range sensing, directional swipe and tap detection.

After resetting, the firmware:

- configures HAL and clocks
- initializes the LEDs and push buttons
- initializes the Time of Filght sensors
- initializes Gesture Detection Middleware:
  - tap detection
  - directional swipe detection
- starts the main loop:
  - push the SW2 button to switch between two demos:
    - range sensing with LED interface
    - Gesture Detection: swipe right/left and tap

Figure 7. **Gesture Recognition application diagram**



# 3 System setup guide

## 3.1 Hardware configuration and board setup

The following figures show the hardware configurations:

Figure 8. BlueCoin and CoinStation

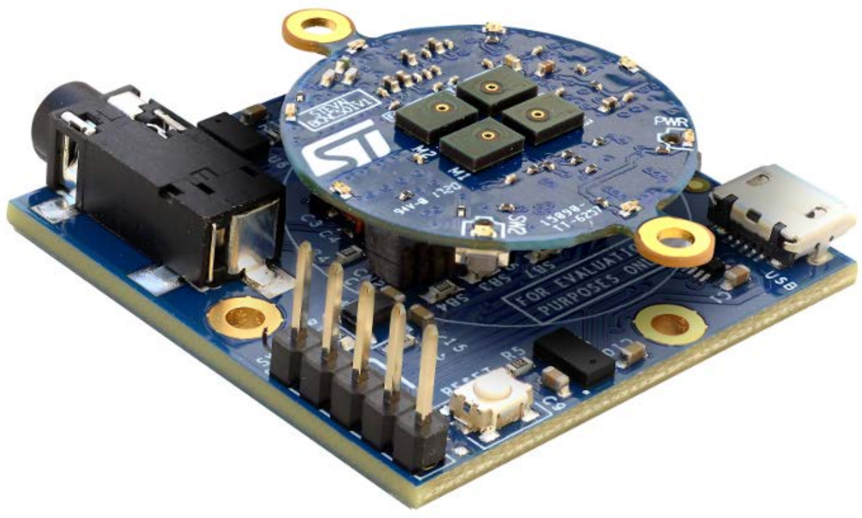
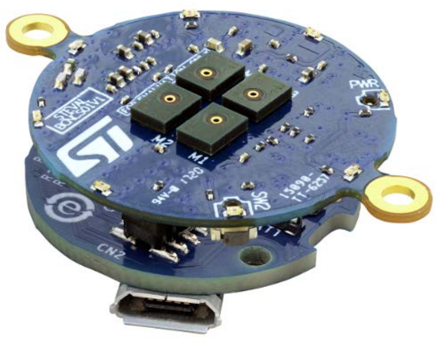


Figure 9. BlueCoin and Cradle expansion



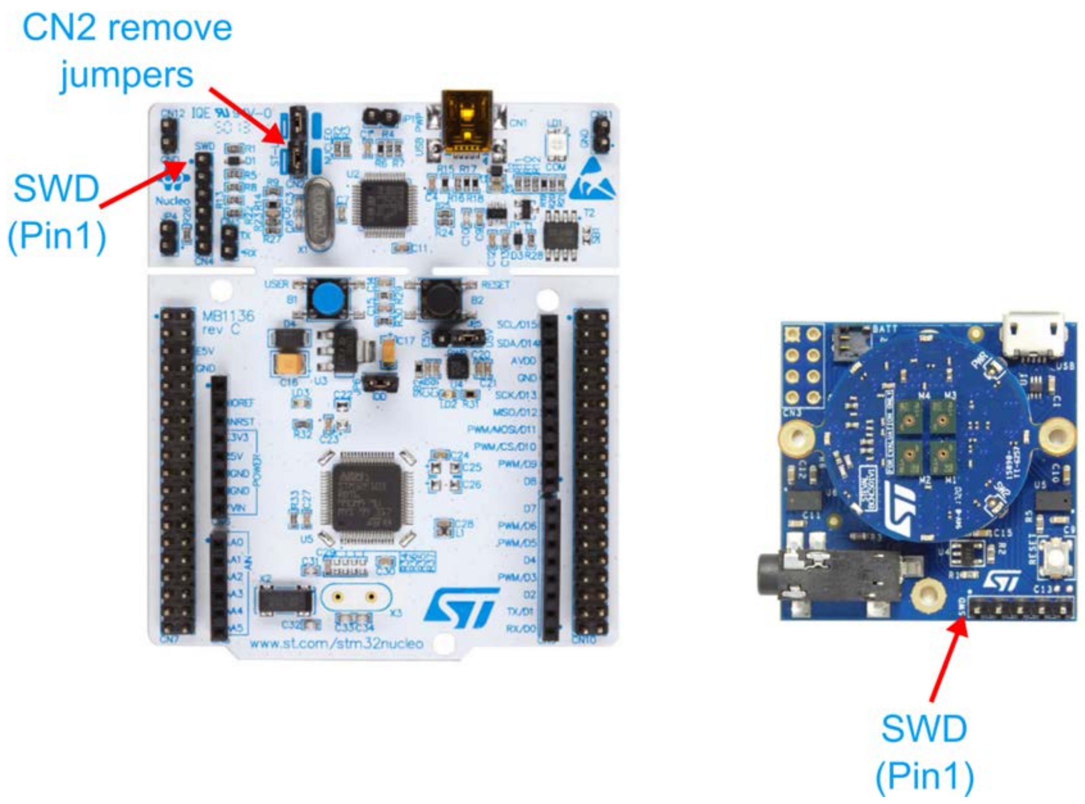
To run the software samples on the BlueCoin hardware (STEVAL-BCNKT01V1):

**Step 1.** Connect an external ST-LINK to the cradle SWD connector (A five-pin flat cable is provided in the BlueCoin Kit package)
The easiest way is to get an STM32 Nucleo board, which includes an ST-LINK V2.1 programmer.

**Step 2.** Ensure that CN2 jumpers are OFF and connect (with correct polarity) your STM32 Nucleo board to the CoinStation with the cable provided.

As shown below, Pin 1 is marked by a dot on the PCB silkscreen.

**Figure 10. STM32 Nucleo board, CoinStation SWD connectors**



**Step 3.** Select one of the package applications

**Step 4.** Open the package through one of the supported IDEs

**Step 5.** Download the code onto the board

## 3.2 Software and hardware compatibility

### 3.2.1 Datalog

#### 3.2.1.1 *USB mode*

BlueCoin

The DataLog application in USB mode can run on both hardware configurations (see Figure 8. BlueCoin and CoinStation and Figure 9. BlueCoin and Cradle expansion), as the only interface required is the USB connector on both cradles.

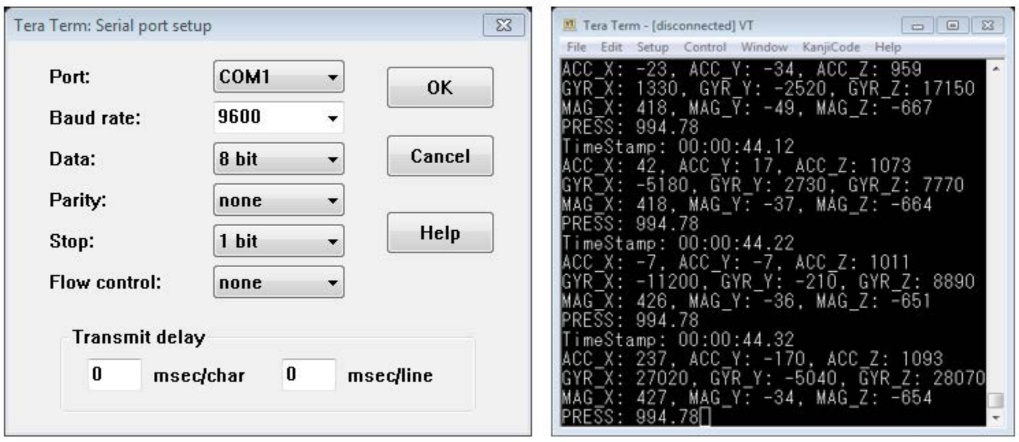
**Step 1.** Connect the board to a PC with a micro-USB cable:
it is recognized as a Virtual COM Port.
You can download the Windows driver from the ST website at VCP driver.

**Step 2.** The board configures the sensors and starts streaming data to the PC.
You can see the incoming data via any COM Port terminal software like Putty or Tera Term.

**Step 3.** Check the correct COM Port number on your PC.

**Step 4.** Use the following configuration:

**Figure 11. Serial Port configuration and received data example**


Tera Term: Serial port setup
OK
Cancel
Help
Tera Term - [disconnected] VT
File Edit Setup Control Window KanjiCode Help

|  |  |
|---|---|
| Port: | COM1 |
| Baud rate: | 9600 |
| Data: | 8 bit |
| Parity: | none |
| Stop: | 1 bit |
| Flow control: | none |

Transmit delay
0 msec/char 0 msec/line

```
ACC_X: -23, ACC_Y: -34, ACC_Z: 959
GYR_X: 1330, GYR_Y: -2520, GYR_Z: 17150
MAG_X: 418, MAG_Y: -49, MAG_Z: -667
PRESS: 994.78
TimeStamp: 00:00:44.12
ACC_X: 42, ACC_Y: 17, ACC_Z: 1073
GYR_X: -5180, GYR_Y: 2730, GYR_Z: 7770
MAG_X: 418, MAG_Y: -37, MAG_Z: -664
PRESS: 994.78
TimeStamp: 00:00:44.22
ACC_X: -7, ACC_Y: -7, ACC_Z: 1011
GYR_X: -11200, GYR_Y: -210, GYR_Z: 8890
MAG_X: 426, MAG_Y: -36, MAG_Z: -651
PRESS: 994.78
TimeStamp: 00:00:44.32
ACC_X: 237, ACC_Y: -170, ACC_Z: 1093
GYR_X: 27020, GYR_Y: -5040, GYR_Z: 28070
MAG_X: 427, MAG_Y: -34, MAG_Z: -654
PRESS: 994.78
```

#### 3.2.1.2 *SD card mode*

The DataLog application in SD card mode only runs on the BlueCoin Cradle hardware.
The SD card is not provided in the package: you have to use an SDHC (Secure Digital High Capacity) Class 10 card and it must be formatted with the FAT32 file system (eg. by using SD Formatter tool).

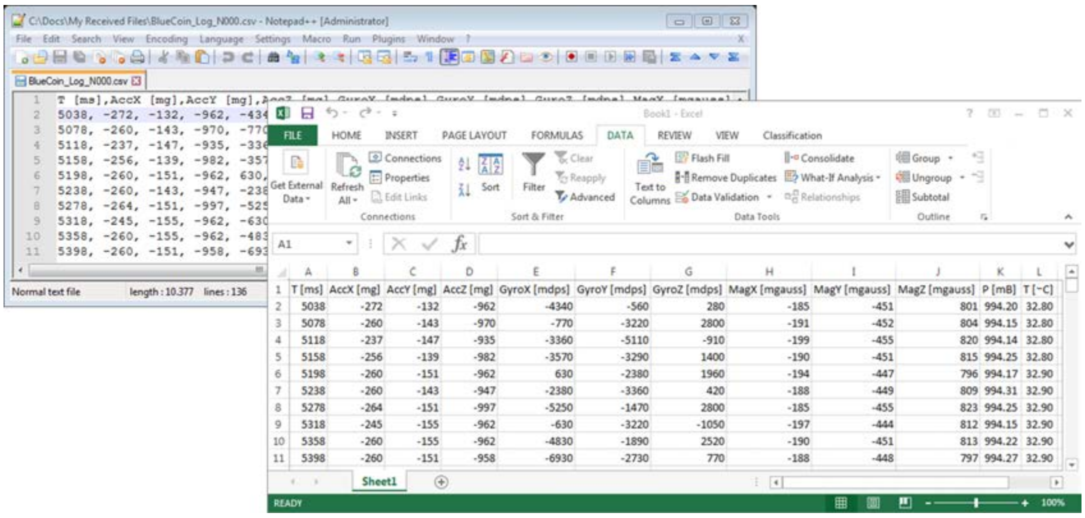
**Step 1.** Insert the card and power the board via the switch.
The acquisition begins when the SW2 button is pressed; push it again to stop.

**Step 2.** Press the SW2 button to start the acquisition

**Step 3.** Press the button again to stop

**Step 4.** Connect the SD card to a PC and open any of the .csv (comma separated values) files to verify the recorded data.

Figure 12. Log file opened with a text editor (left) or Excel (right)



```
T [ms],AccX [mg],AccY [mg],
5038, -272, -132, -962, -434
5078, -260, -143, -970, -770
5118, -237, -147, -935, -336
5158, -256, -139, -982, -357
5198, -260, -151, -962, 630,
5238, -260, -143, -947, -238
5278, -264, -151, -997, -525
5318, -245, -155, -962, -630
5358, -260, -155, -962, -483
5398, -260, -151, -958, -693
```

|  | A | B | C | D | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | T [ms] | AccX [mg] | AccY [mg] | AccZ [mg] | GyroX [mdps] | GyroY [mdps] | GyroZ [mdps] | MagX [mgauss] | MagY [mgauss] | MagZ [mgauss] | P [mB] | T [~C] |
| 2 | 5038 | -272 | -132 | -962 | -4340 | -560 | 280 | -185 | -451 | 801 | 994.20 | 32.80 |
| 3 | 5078 | -260 | -143 | -970 | -770 | -3220 | 2800 | -191 | -452 | 804 | 994.15 | 32.80 |
| 4 | 5118 | -237 | -147 | -935 | -3360 | -5110 | -910 | -199 | -455 | 820 | 994.14 | 32.80 |
| 5 | 5158 | -256 | -139 | -982 | -3570 | -3290 | 1400 | -190 | -451 | 815 | 994.25 | 32.80 |
| 6 | 5198 | -260 | -151 | -962 | 630 | -2380 | 1960 | -194 | -447 | 796 | 994.17 | 32.90 |
| 7 | 5238 | -260 | -143 | -947 | -2380 | -3360 | 420 | -188 | -449 | 809 | 994.31 | 32.90 |
| 8 | 5278 | -264 | -151 | -997 | -5250 | -1470 | 2800 | -185 | -455 | 823 | 994.25 | 32.90 |
| 9 | 5318 | -245 | -155 | -962 | -630 | -3220 | -1050 | -197 | -444 | 812 | 994.15 | 32.90 |
| 10 | 5358 | -260 | -155 | -962 | -4830 | -1890 | 2520 | -190 | -451 | 813 | 994.22 | 32.90 |
| 11 | 5398 | -260 | -151 | -958 | -6930 | -2730 | 770 | -188 | -448 | 797 | 994.27 | 32.90 |

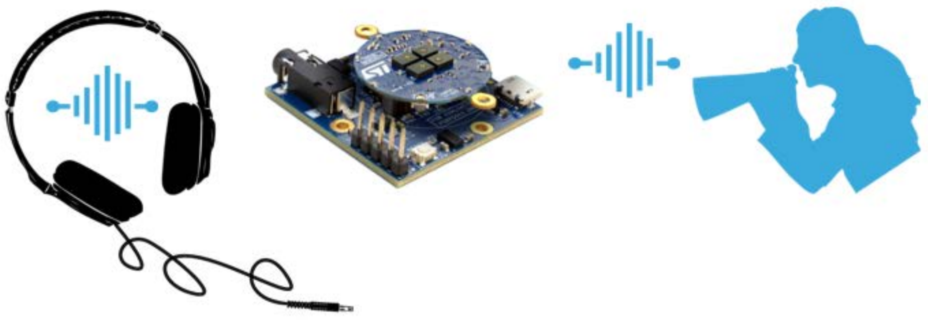
C:\Docs\My Received Files\BlueCoin_Log_N000.csv - Notepad++ [Administrator]
BlueCoin_Log_N000.csv
Normal text file
length : 10.377 lines : 136
Book1 - Excel
FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW Classification
Sheet1
READY

## 3.2.2 AudioLoop

The AudioLoop application only runs on the CoinStation as the Audio DAC and the 3.5 mm jack are not present on the small Cradle board.

Once downloaded, the application configures the microphone and the audio DAC. The sound acquired by the microphone is then played on the connected headphones or loudspeaker.

Figure 13. Audioloop application diagram



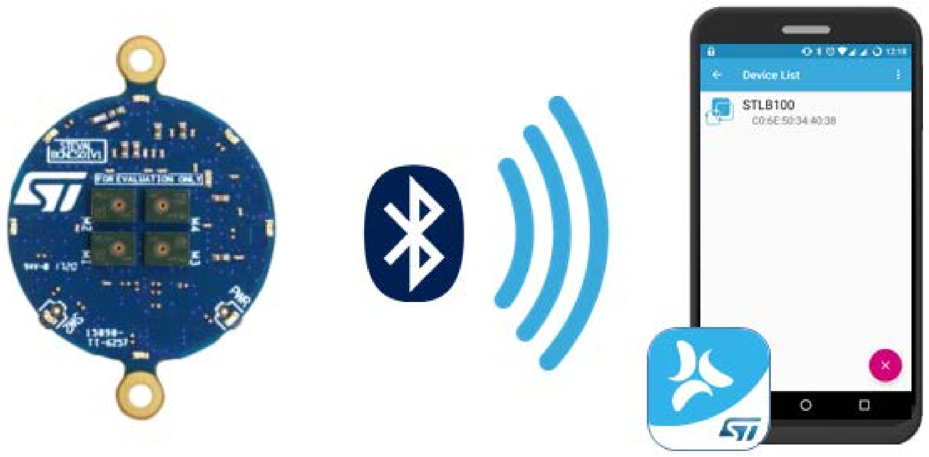
## 3.2.3 BLE_SampleApp

The BLE_SampleApp can run on both hardware configurations.

Once downloaded, a BlueCoin LED starts blinking: the device is waiting for a connection via Bluetooth.

Open STBLESensor app on your Android or iOS smartphone and use it to connect to the BlueCoin.

Figure 14. BLE_SampleApp BLE connection through BlueMS app



### 3.2.4 GestureDetect

The GestureDetect application only runs on the CoinStation as the Time-of-Flight sensors are not present on the small Cradle board.

## Revision history

Table 1. Document revision history

| Date | Version | Changes |
|---|---|---|
| 12-Jul-2017 | 1 | Initial release. |
| 05-Mar-2019 | 2 | Updated Introduction, Section 2.1 Overview and Figure 2. STSW-BCNKT01 software architecture.<br>Added Section 2.5.1 Audio_SD. |

## Contents

# List of figures