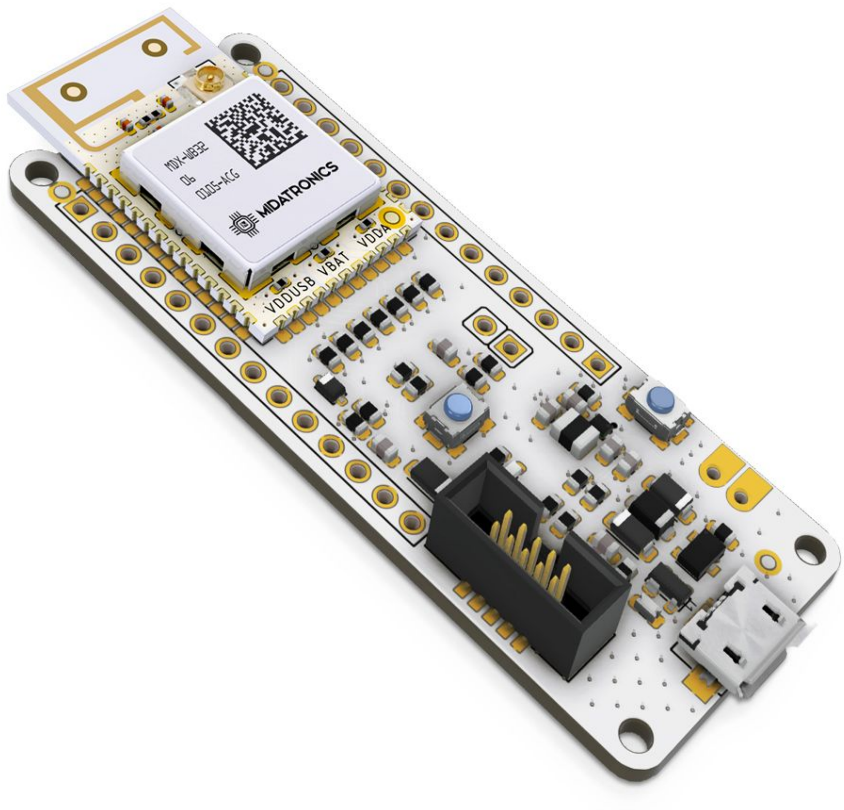

SHARKY MKR

User's Guide

MDX-MKR-STWBP-R01 : Sharky MKR PCB Ant.

MDX-MKR-STWBU-R01 : Sharky MKR uFL antenna

Outline

Illustrations

Tables

Revisions

REVISION	DATE	DESCRIPTION	STATUS	AUTHOR	REVISER
Ver 1.0	04/04/2019	First Release	Draft	info@midatronics.co m	UA-EM

Disclaimer

Midatronics S.r.l.
Via Zucchi 1 20900
Monza (Monza Brianza)
Italy

info@midatronics.com
www.midatronics.com

1 Introduction

1.1 Description

This document describes the Sharky MKR Board.

Sharky MKR board is based on the Sharky module that contains an STMicroelectronics STM32WB55CG, a dual-core MCUs with wireless support, based on an Arm® Cortex®-M4 core running at 64 MHz (application processor) plus an Arm® Cortex®-M0+ core at 32 MHz (network processor).

With two totally independent cores, this innovative architecture is optimized for real-time execution (radio-related software processing).

The STM32WB55 Bluetooth 5.0-certified device offers Mesh 1.0 software support, multiple profiles and flexibility to integrate proprietary BLE stacks.

OpenThread-certified software stack is available. The radio can also run BLE/OpenThread protocols concurrently. The embedded generic MAC allows the usage of other IEEE 802.15.4 proprietary stacks like ZigBee®, or proprietary protocols, giving even more options for connecting devices to the Internet of Things (IoT).

The board pinout is compatible with Arduino MKR boards, and can be programmed with Arduino IDE thanks to the STM32Duino project. The processor voltage is 3.3V .

Onboard SWD connector allows programming the board with STLink in-circuit debugger and programmer and Atollic/IAR/SW4STM32/Keil IDEs.

Main features

- Board size 65.90 x 25 mm
- Integrated BLE/OpenThread or IEEE 802.15.4 programmable networking stacks
- Processor Voltage: 3.3V

1.2 Kit Contents

The following items are included in the box:

- 1x Dragonfly board

1.3 Getting Started

The Dragonfly board, developed by Midatronics for Arrow Electronics, is a ready-to-use Internet of Things (IoT) hardware.

Please refer to software chapter to learn how to get started to develop your application using the Arduino IDE or STM32 Studio IDE.

2 System Overview

2.1 BLE Technology Overview

Bluetooth Low Energy (BLE) is the main feature of the Bluetooth specification v4.0 released in December 2009. BLE is a new protocol that allows for long-term operation of Bluetooth devices that transmit low volumes of data. BLE enables smaller form factors, better power optimization, and the ability to operate on a small power cell for several years.

The classic Bluetooth specification defines a uniform structure for a wide range of devices that connect to each other. Bluetooth operates primarily using ad hoc piconets. A master device controls up to seven slaves per piconet; the slaves communicate with the master device but they do not communicate with each other. However, a slave device may participate in one or more piconets, essentially a collection of devices connected via Bluetooth. A summary of classic Bluetooth topology with multiple piconets, called scatternet, can be found below.

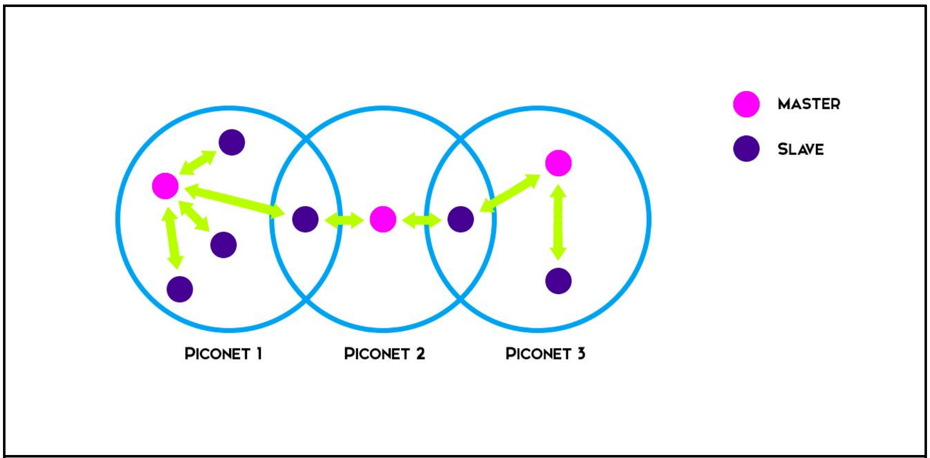

Figure 1. Bluetooth Scatternet topology

In a BLE topology, the slaves each communicate on a separate physical channel with the master. Unlike a classic Bluetooth piconet, where all slaves listen for incoming connections and therefore need to be on constant standby, a BLE slave invites connections and so is in total control of when to consume power. A BLE master, which is assumed to have less power constraints, will listen for advertisements and make connections on the back of an advertisement packet. A diagram of this can be found below.

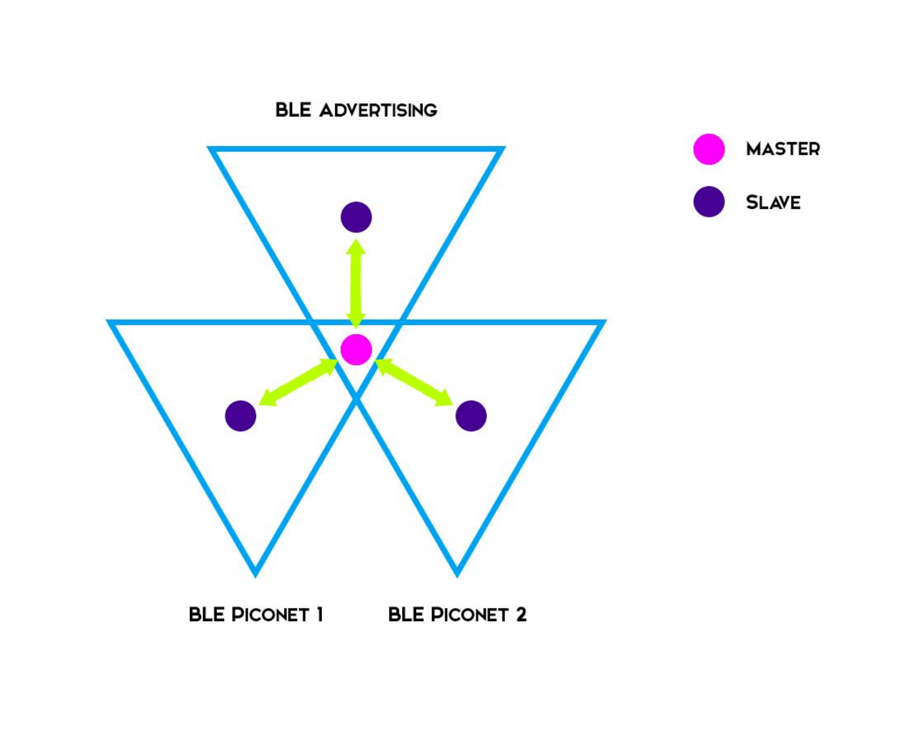

Figure 2. BLE Star-bus Topology

While BLE inherits the operating spectrum and the basic structure of the communication protocol from the classic Bluetooth protocol, BLE implements a new lightweight Link Layer that provides ultra-low power idle mode operation, fast device discovery, and reliable and secure point-to-multipoint data transfers. As a result, BLE offers substantially lower peak, average, and idle-mode power consumption than classic Bluetooth. Averaged over time, BLE consumes only 10% of the power consumed by classic Bluetooth.

In addition to its ultra-low power consumption, BLE has several unique features that set it apart from other available wireless technologies, including:

- **Interoperability**: Like classic Bluetooth devices, BLE devices follow standards set by the Bluetooth Special Interest Group (SIG), and BLE devices from different manufacturers interoperate.
- **Robustness**: BLE uses fast frequency hopping to secure a robust transmission even in the presence of other wireless technologies.
- **Ease of Use**: BLE has been developed so that it is straightforward for designers to implement it in a variety of different applications.
- **Latency**: The total time to send small chunks of data is generally fewer than 6 ms, and as low as 3 ms (compared to 100 ms with classic Bluetooth).
- **Range**: Thanks to an increased modulation index, BLE technology offers greater range (up to 200 feet and beyond, in ideal environments) than to classic Bluetooth offers.

2.2 BLE Mesh Technology overview

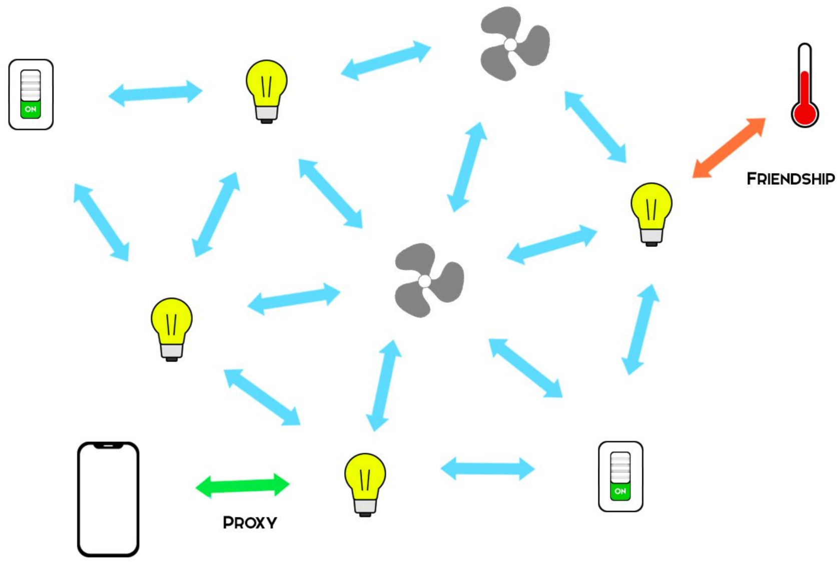

Figure 3. BLE Mesh Topology

Borrowing from the original Bluetooth specification, the Bluetooth SIG defines several profiles — specifications for how a device works in a particular application — for low energy devices. Manufacturers are expected to implement the appropriate specifications for their device in order to ensure compatibility. A device may contain implementations of multiple profiles.

Majority of current low energy application profiles is based on the generic attribute profile (GATT), a general specification for sending and receiving short pieces of data known as attributes over a low energy link. Bluetooth mesh profile is the exception to this rule as it is based on General Access Profile (GAP).

Bluetooth mesh profiles use Bluetooth Low Energy to communicate with other Bluetooth Low Energy devices in the network. Each device can pass the information forward to other Bluetooth Low Energy devices creating a "mesh" effect. For example, switching off an entire building of lights from a single smartphone.

Conceptually, the Bluetooth Mesh Standard is defined as a publish/subscribe model where publishers can publish to a certain topic and subscribers can subscribe to one or more topics of interest.

This concept is used as an inspiration for the implementation in the standard. A node in a Bluetooth Mesh network can subscribe to one or more addresses (stored in the *subscriber list*) and publish to one specific address (stored in the *publish address*).

To be able to connect these different publishers and subscribers, a mesh topology is created. The standard uses BLE advertising and scanning as an underlying technology to implement communication. To communicate in a Bluetooth Mesh network, a flooding mechanism is used. By default, a flooding mechanism ensures that each node in the network repeats incoming messages, so that they are relayed further, until the destination node is reached.

The standard uses a new type of BLE advertisement packet to communicate in a mesh network, which is only supported by devices that support both BLE and Bluetooth Mesh. Fortunately, the standard also defines a backwards compatibility feature to ensure that BLE devices which do not support Bluetooth Mesh can also be part of a Bluetooth Mesh network.

2.3 Thread Technology overview

Thread is a secure, wireless mesh networking protocol. The Thread stack is an open standard that is built upon a collection of existing Institute for Electrical and Electronics Engineers (IEEE) and Internet Engineering Task Force (IETF) standards.

The Thread stack supports IPv6 addresses and provides low-cost bridging to other IP networks and is optimized for low-power/battery-backed operation, and wireless device-to-device communication. The Thread stack is designed specifically for Connected Home applications where IP-based networking is desired and a variety of application layers can be used on the stack.

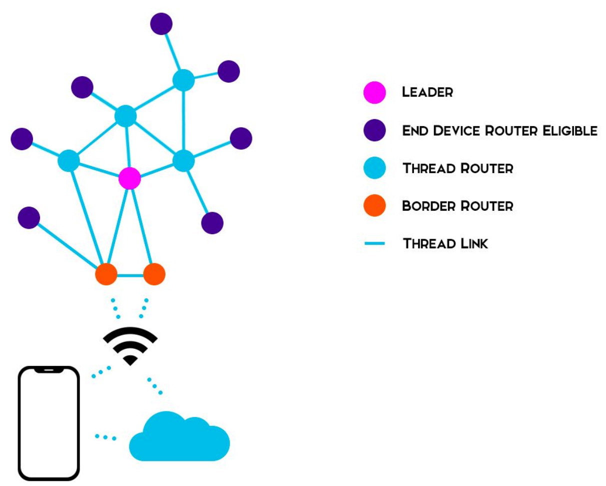

Figure 4. Thread Network Architecture

These are the general characteristics of the Thread stack focused on the Connected Home:

- **Simple network installation, start-up, and operation**: The Thread stack supports several network topologies. Installation is simple using a smartphone, tablet, or computer. Product installation codes are used to ensure only authorized devices can join the network. The simple protocols for forming and joining networks allow systems to self-configure and fix routing problems as they occur.
- **Secure**: Devices do not join the network unless authorized and all communications are encrypted and secure. Security is provided at the network layer and can be at the application layer. All Thread networks are encrypted using a smartphone-era authenticationscheme and Advanced Encryption Standard (AES) encryption. The security used in Thread networks is stronger than other wireless standards the Thread Group has evaluated.
- **Small and large networks**: Home networks vary from several to hundreds of devices. The networking layer is designed to optimize the network operation based on the expected use.
- **Range**: Typical devices provide sufficient range to cover a normal home. Readily available designs with power amplifiers extend the range substantially. A distributed spread spectrum is used at the Physical Layer (PHY) to be more immune to interference.
- **No single point of failure**: The Thread stack is designed to provide secure and reliable operations even with the failure or loss of individual devices.
- **Low power**: Devices efficiently communicate to deliver an enhanced user experience with years of expected life under normal battery conditions. Devices can typically operate for several years on AA type batteries using suitable duty cycles.
- **Cost-effective**: Compatible chipsets and software stacks from multiple vendors are priced for mass deployment, and designed from the ground up to have extremely low-power consumption. Typical home products run in the connected home include: normally powered (lighting, appliances, HVAC, fans), powered or battery-operated (thermostats, smoke detectors, CO and CO_2 detectors, security systems), and normally battery-operated (door sensors, window sensors, motion sensors, door locks).

2.5 STM32WB Wireless System-on-Chip

The Sharky MKR Board is based on STMicroelectronics STM32WB55CG, a dual-core MCUs with wireless support based on an Arm® Cortex®-M4 core running at 64 MHz (application processor) plus an Arm® Cortex®-M0+ core at 32 MHz (network processor).

The STM32WB platform is an evolution of the well-known market-leading STM32L4 ultra-low-power series of MCUs. It provides the same digital and analog peripherals suitable for applications requiring extended battery life and complex functionalities.

STM32WB proposes a variety of communication assets, a practical crystal-less USB2.0 FS interface, audio support, an LCD driver, up to 72 GPIOs, an integrated SMPS for power consumption optimization, and multiple low-power modes to maximize battery life.

On top of wireless and ultra-low-power aspects, a particular focus was placed on embedding security hardware functions such as a 256-bit AES, PCROP, JTAG Fuse, PKA (elliptic curve encryption engine), and Root Secure Services (RSS). The RSS allows authenticating OTA communications, regardless of the radio stack or application.

For more informations on STM32WB visit the following site:

https://www.st.com/en/microcontrollers/stm32wb-series.html?querycriteria=productId=SS1961

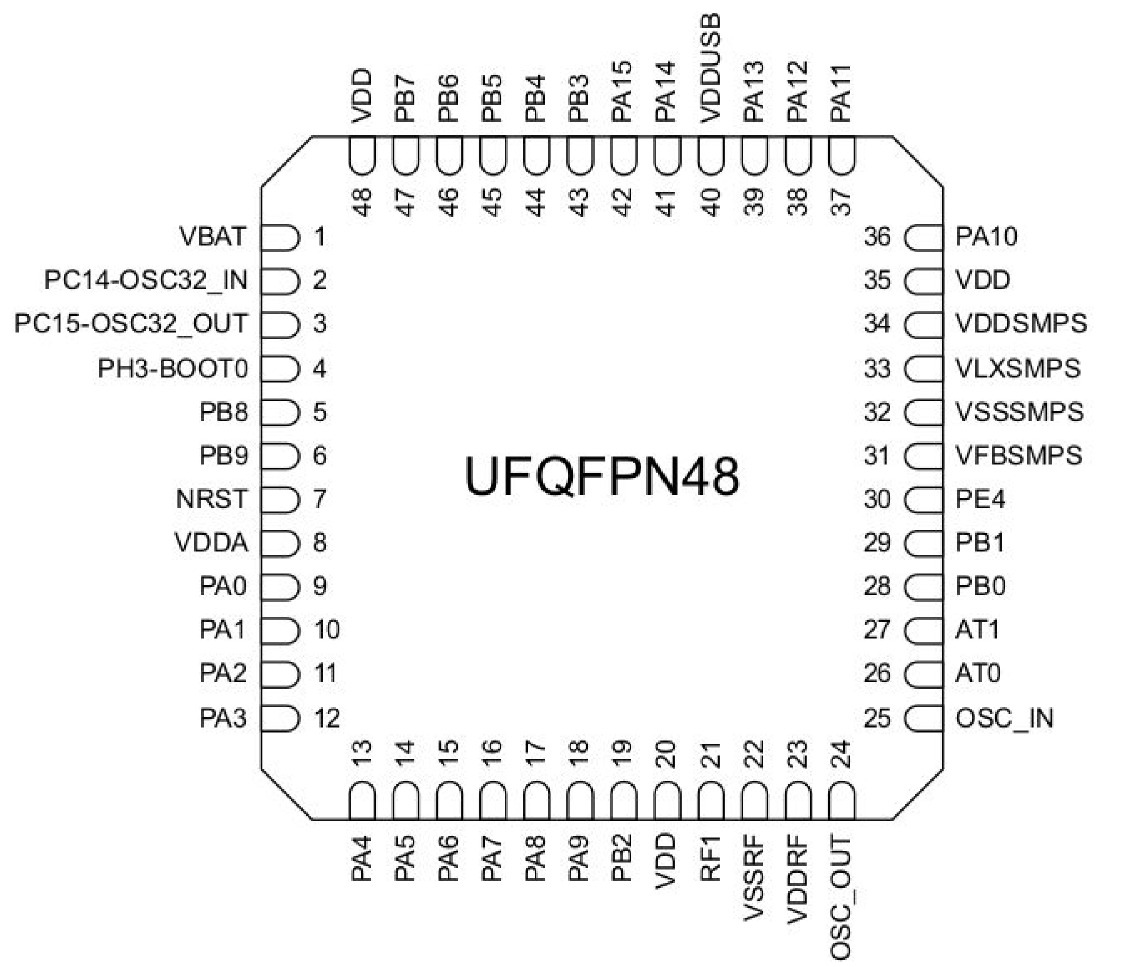

Figure 5. STM32WB55CG pinout

2.6 Block Diagram

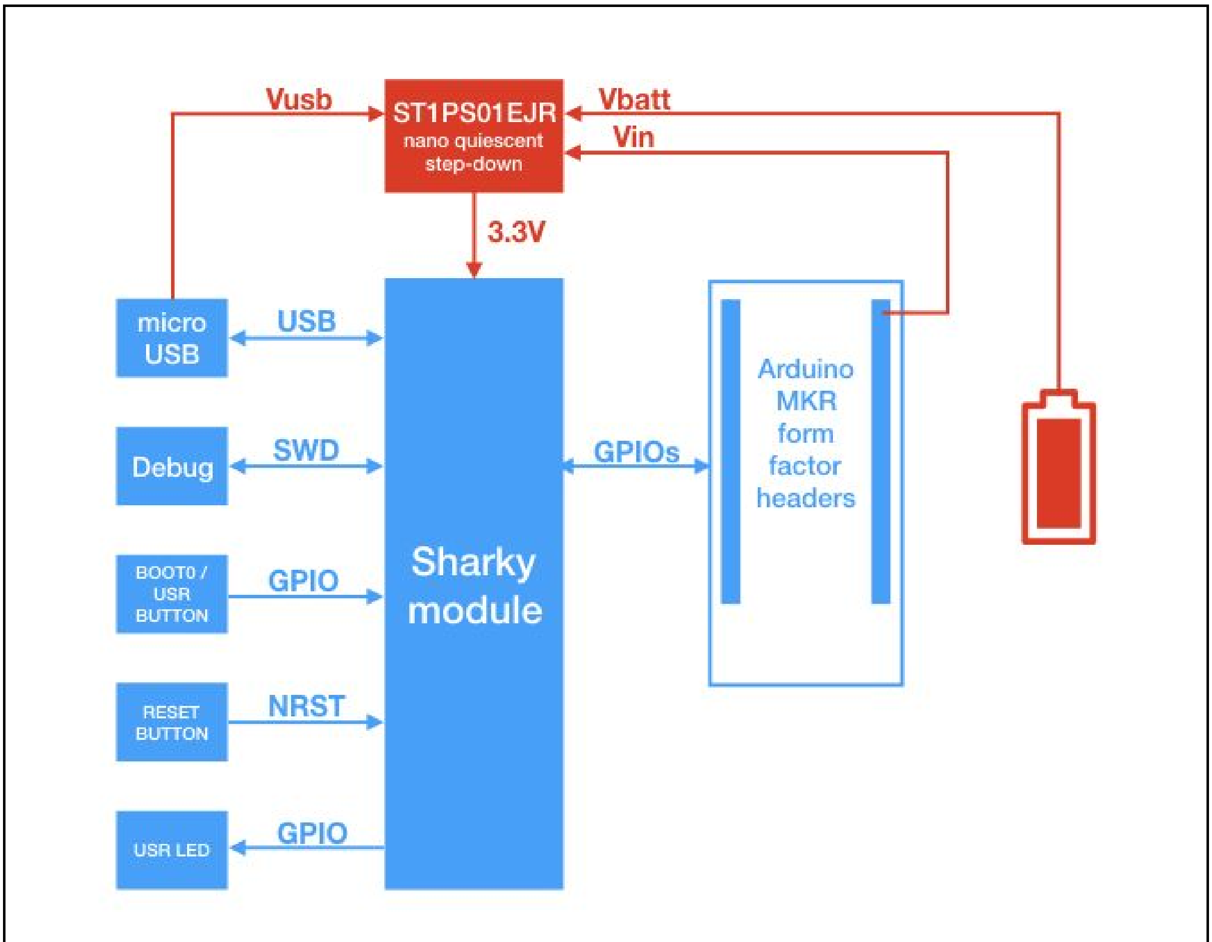

Figure 6. Skarky MKR Block Diagram

2.7 Board Specifications

Characteristics	Value
CPU Clock Speed	64 MHz
Flash Memory	1 Mbyte
SRAM	256 KByte
Connector	1 USB 1 SWD Debugger 1 battery

	Arduino MKR compatible pinout
Board supply voltage	3.3 V to 5.5 V DC
Operating Voltage	3.3 V (*)
Operating Temperature	-40 °C to +85 °C
Dimensions	65.90 x 25 mm
RoHS status	Compliant

Table 1. Board Specifications

(*) All digital I/O refer to this level

2.8 Sharky Module Block Diagram

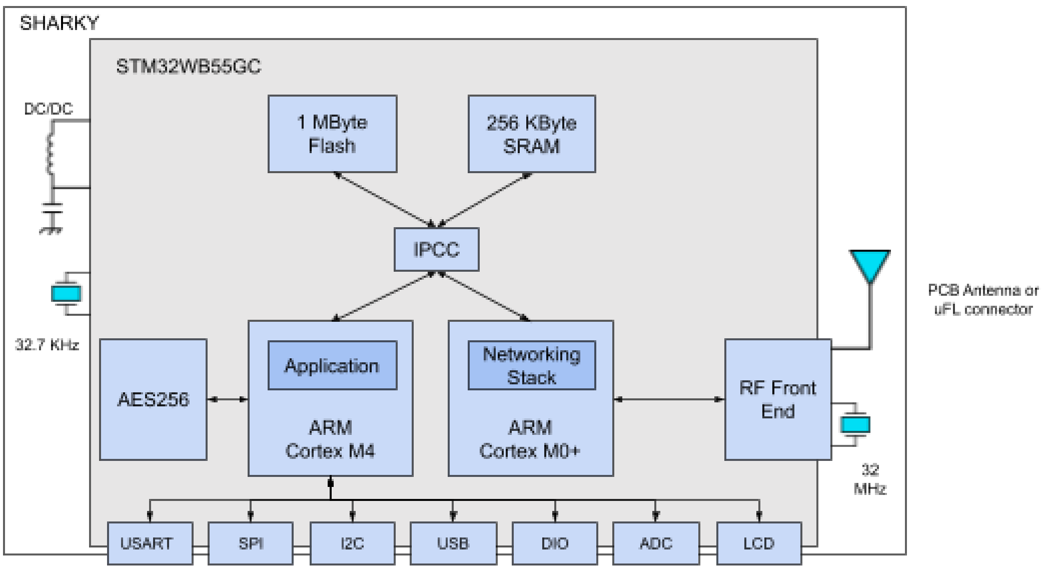

Figure 7. Sharky Module with PCB Antenna / or uFL connector

3 Connectors

This chapter gives you an overview of the Sharky MKR board connectivity.

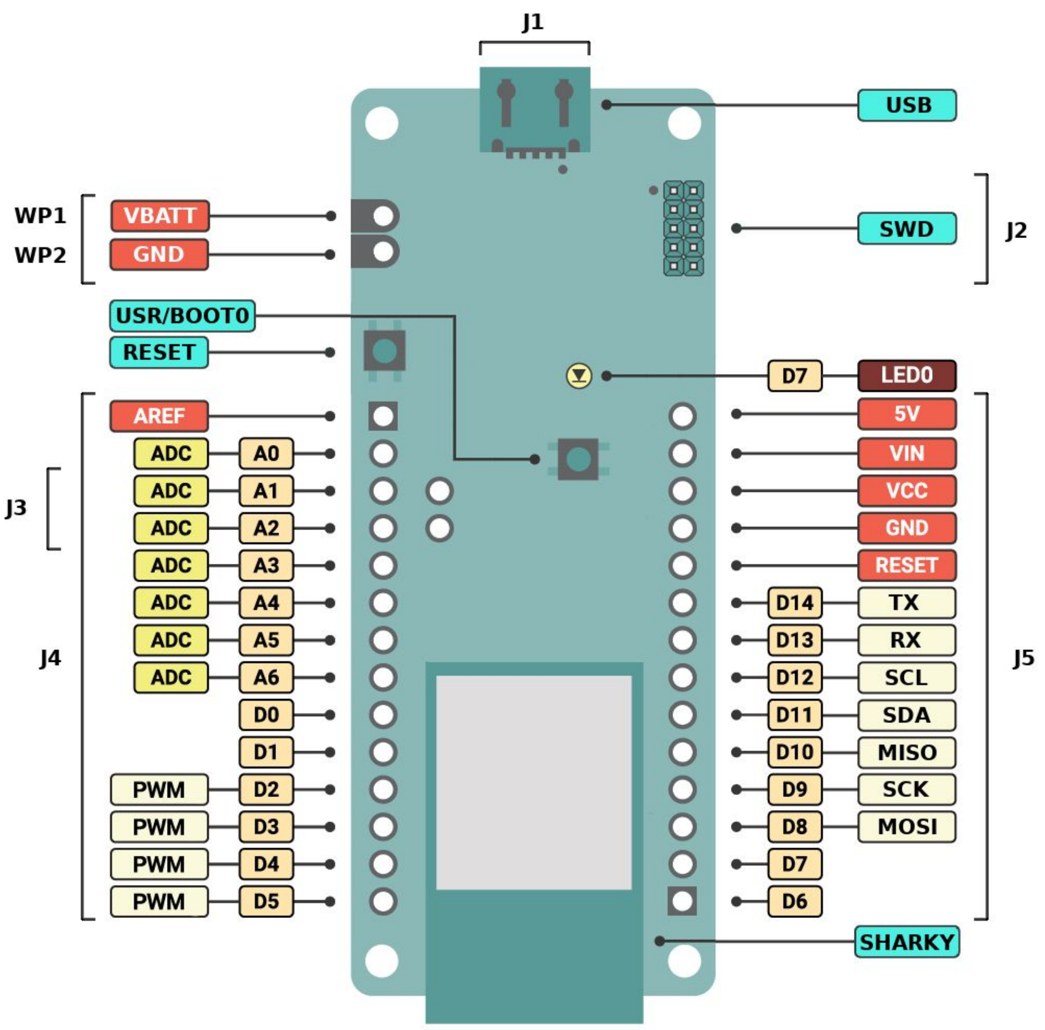

Figure 8. Sharky MKR board pinout

3.1 Arduino MKR Connectors

The connectors J4 and J5 provide the user with a standard Arduino MKR shield slot as listed below.

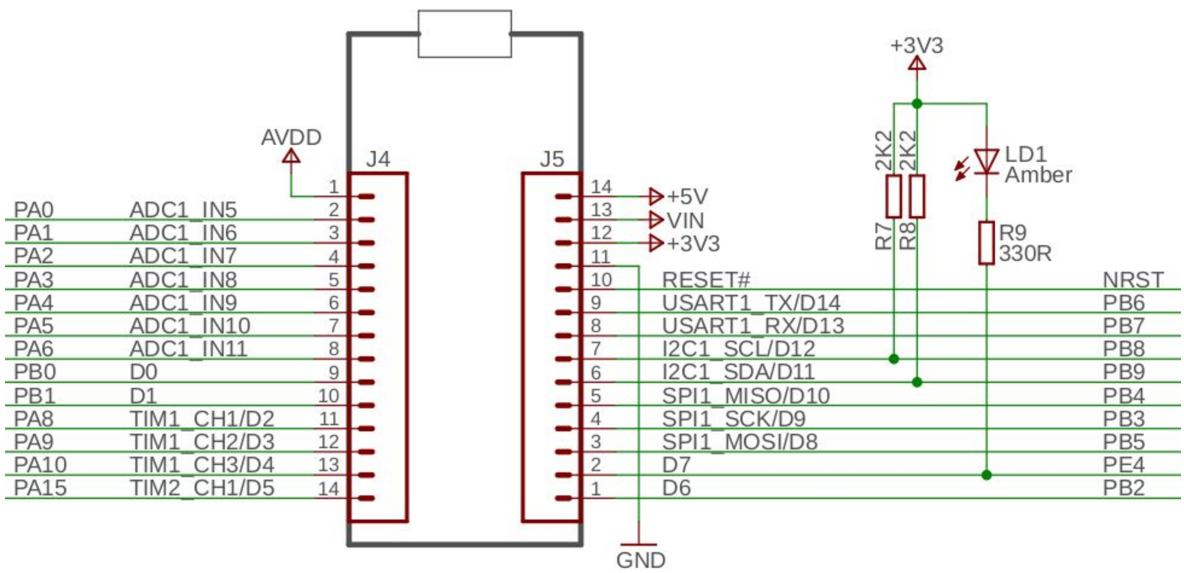

Figure 9. Sharky MKR Connectors

Conn	Arduino	STM	Description	Sharky Module pin
J4-1			VREF+	VDDA**
J4-2	**A0**	**PA0**	ADC1_IN5/A0	J27
J4-3	**A1**	**PA1**	ADC1_IN6/A1	J28
J4-4	**A2**	**PA2**	ADC1_IN7/A2	J29
J4-5	**A3**	**PA3**	ADC1_IN8/A3	J31
J4-6	**A4**	**PA4**	ADC1_IN9/A4	J32

J4-7	**A5**	**PA5**	ADC1_IN10/A5	J33
J4-8	**A6**	**PA6**	ADC1_IN11/A6	J34
J4-9	**D0**	**PB0**	D0	J2
J4-10	**D1**	**PB1**	D1	J3
J4-11	**D2**	**PA8**	TIM1_CH1/D2	J36
J4-12	**D3**	**PA9**	TIM1_CH2/D3	J37
J4-13	**D4**	**PA10**	TIM1_CH3/D4	J7
J4-14	**D5**	**PA15**	TIM2_CH1/D5	J10
J5-1	**D6**	**PB2**	D6	J38
J5-2	**D7**	**PE4**	D7/LED0	J4
J5-3	**D8**	**PB5**	SPI1_MOSI/D8	J13
J5-4	**D9**	**PB3**	SPI1_SCK/D9	J11
J5-5	**D10**	**PB4**	SPI1_MISO/D10	J12
J5-6	**D11**	**PB9**	I2C1_SDA/D11	J25
J5-7	**D12**	**PB8**	I2C1_SCL/D12	J23
J5-8	**D13**	**PB7**	USART1_RX/D13	J20
J5-9	**D14**	**PB6**	USART1_TX/D14	J15
J5-10	**RESET**	7-NRST	RESET*	
J5-11	**GND**		GND	
J5-12	**VCC**		3V3	
J5-13	**VIN**		VIN	
J5-14	**5V**		5V	

Table 2: Sharky MKR pinout

* see reset circuit in DBG connector

** On UFQFPN48 VDDA is connected to VREF+.

3.2 J1 USB connector

The board is equipped with an USB (J1) Full-Speed (12 Mbps) device port on J1 connector. The Sharky MKR board can be powered through this interface.

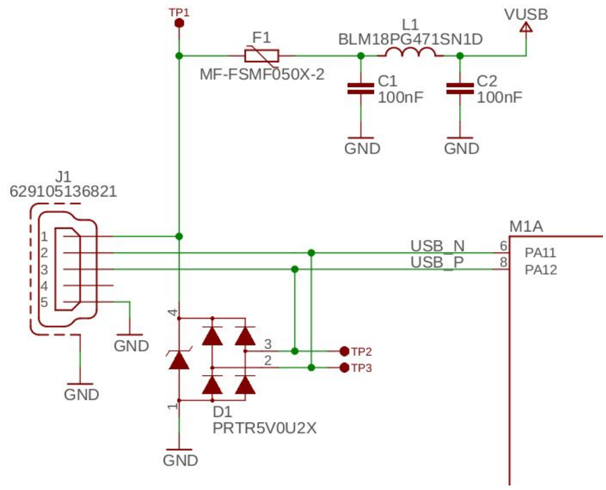

Figure 10. USB Interface connector

3.3 J2 SWD/Debug Connector

The Sharky MKR board features an on-board SWD Connector (J2) that can be used to program and debug the microcontroller.

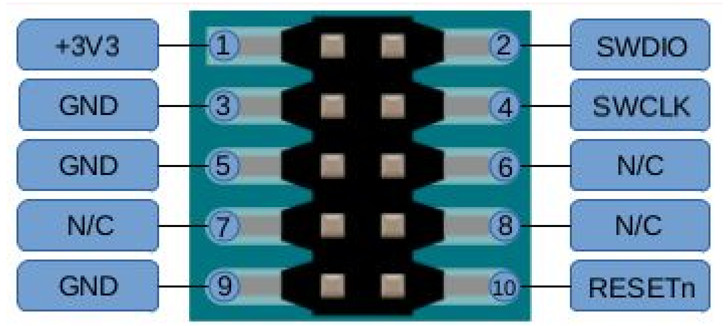

Figure 11. SWD Connector Pinout

Conn	STM	Description	Sharky Module pin
J2-1		3V3	
J2-2	**PA13**	JTSM-SWDIO	J5
J2-3		GND	
J2-4	**PA14**	JTCK-SWCLK	J9
J2-5		GND	
J2-6		NC	
J2-7		NC	
J2-8		NC	
J2-9		GND	
J2-10	**Pin 7**	NRST	J30

Table 3: SWD connector pinout

3.4 J3 VBATT Voltage Sense

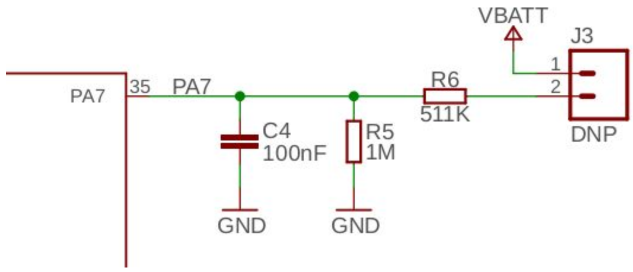

Figure 12. VBATT voltage sense

VBATT sensing is disabled by default to minimize current consumption. It can be enabled connecting J3 pins and configuring pin PA7 as analog input.

4 Usage

This chapter describes how to connect, configure and interact with the Sharky MKR board.

4.1 Power Supply

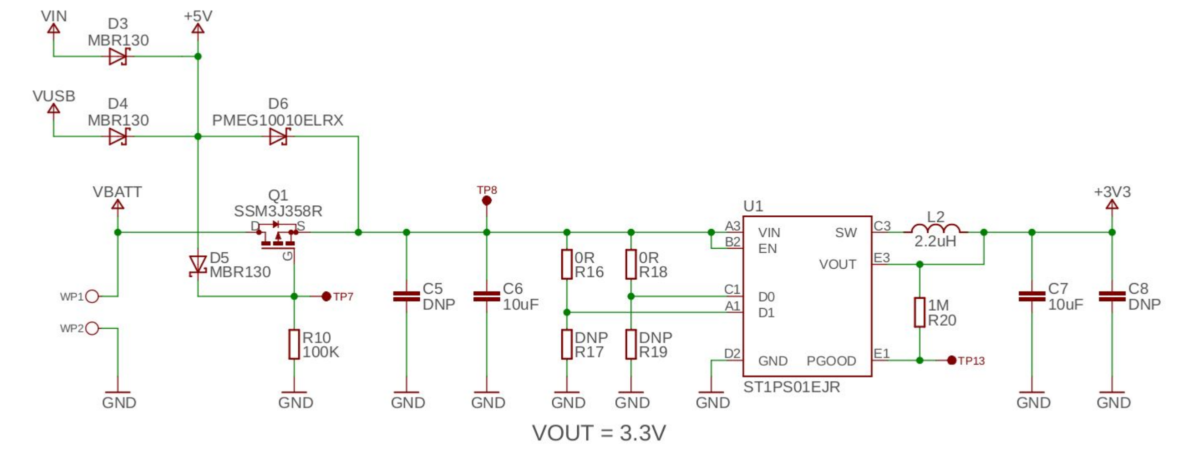

Figure 13. Sharky MKR Board Power Supply

Sharky MKR has an onboard ST1PS01 400 mA nano-quiescent synchronous step-down converter. The output voltage of the converter is 3.3V.

The board can be powered by three different power supplies sources:

- 5V on micro USB connector
- VBATT pin, from 3.3 V to 5.5 V.
- VIN pin, form 3.3V to 5.5V

4.3 Reset Button

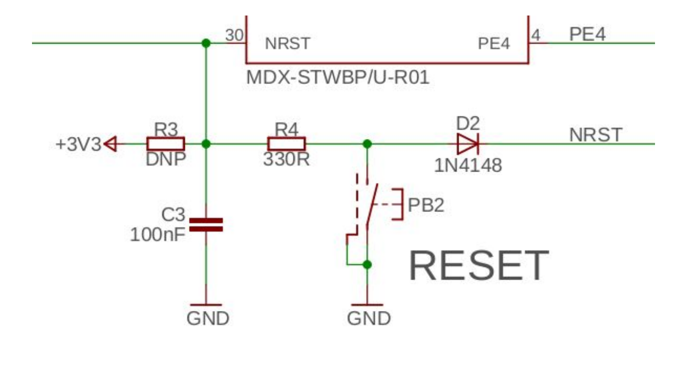

Figure 14. Reset button circuit

Push the button to reset the MCU

4.4 USR/BOOT0 button

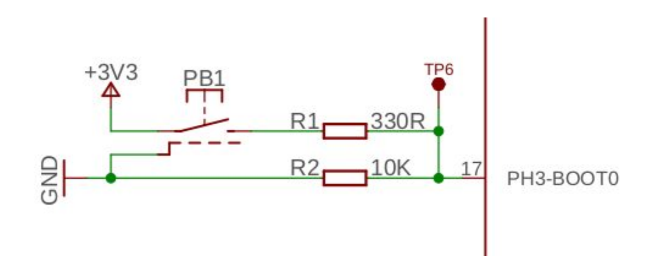

Figure 15. USR/BOOT0 button circuit

The USR/BOOT0 button can be pressed during reset to load the internal MCU bootloader. After reset it can be used as a standard input button.

4.5 LED

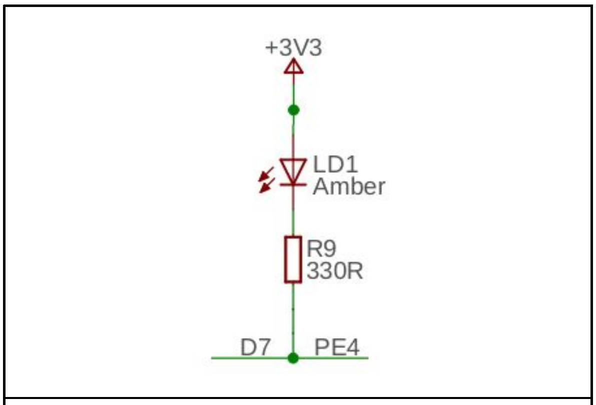

Figure 16. LED Circuit

The LD1 led is connected to the PE4 pin of the M4 core. Set the pin low to lit the LED.

5 Board Layout

The following pictures show the dimensions of the Sharky MKR Board.

6 Firmware Upload

The STM32WB SoC inside the Sharky module has 2 cores that share the same FLASH and SRAM addresses:

- M0+ core for embedded communication stack
- M4 core for user application

The module is delivered with BLE communication stack firmware installed. Thread and other stacks can be installed by the user.

6.1 FW upload to M4 core

The GUI application for flashing firmware is STM32CubeProgrammer, available for Windows, Linux and MacOS operating systems. It can be downloaded from ST at:

https://www.st.com/en/development-tools/stm32cubeprog.html

The firmware for the M4 CPU can be uploaded:

- Using an STLink V2 or V3 device connected to the SWD interface
- Using the embedded ROM Bootloader that is selected by rising the BOOT0 pin and setting nBOOT0 and nBOOT1 option bytes. In this case the firmware can be uploaded via USB or UART

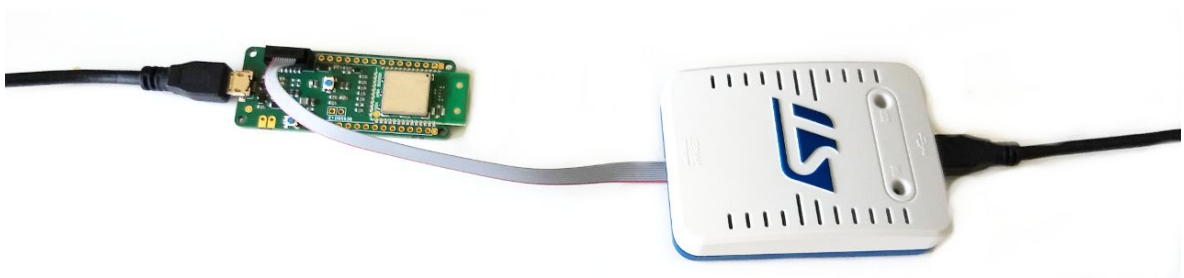

Figure 17. Board connections with STLink

6.2 FW upload to M0+ core

The M0+ firmware cannot be uploaded using STLink programmer, only the internal bootloader is allowed to update the firmware.

ST provides the package en.stm32cubewb.zip (download from: https://my.st.com/content/my_st_com/en/products/embedded-software/mcu-mpu-embedded-software/stm32-embedded-software/stm32cube-mcu-mpu-packages/stm32cubewb.license=1554117969251.product=STM32CubeWB.version=1.0.0.html#tools-software)

With the compiled staks:

- stm32wb5x_BLE_Stack_fw.bin
  - Full BLE Stack 5.0 certified : Link Layer, HCI, L2CAP, ATT, SM, GAP and GATT database
  - BT SIG Certification listing : Declaration ID D042164
- stm32wb5x_BLE_HCILayer_fw.bin
  - HCI Layer only mode 5.0 certified : Link Layer, HCI
  - BT SIG Certification listing : Declaration ID D042213
- stm32wb5x_Thread_FTD_fw.bin
  - Full Thread Device certified v1.1
  - To be used for Leader / Router / End Device Thread role (full features excepting Border Router)
- stm32wb5x_Thread_MTD_fw.bin
  - Minimal Thread Device certified v1.1
  - To be used for End Device and Sleepy End Device Thread role
- stm32wb5x_BLE_Thread_fw.bin
  - Static Concurrent Mode BLE Thread
  - Supports Full BLE Stack 5.0 certified and Full Thread Device certified v1.1
- stm32wb5x_Mac_802_15_4_fw.bin
  - MAC API is based on latest official IEEE Std 802.15.4-2011
  - To be used for MAC FFD and RFD devices
- stm32wb5x_rfmonitor_phy802_15_4_fw.bin
  - Dedicated firmware binary to be used with STM32CubeMonitor-RF application.
  - Refer to STM32CubeMonitor-RF User Manual (UM2288) to get application details.

To flash the firmware follow the instructions (from the file "Release_Notes.html" in /STM32Cube_FW_WB_V1.0.0/Projects/STM32WB_Copro_Wireless_Binaries extracted from en.stm32cubewb.zip package:

- STEP 1: Use STM32CubeProgrammer
  - Version 1.4 or higher.
  - It gives access to Firmware Upgrade Service (FUS) (AN 5185) through Bootloader.
  - It is currently available as Command Line Interface (CLI) mode.
- STEP 2: Access to Bootloader USB Interface (system flash)
  - Boot mode selected by Boot0 pin set to VDD (check option bytes nBOOT0 and nBOOT1 are set)

        - Keep user button pressed during reboot
- STEP 3 : Delete current wireless stack :
    - *STM32_Programmer_CLI.exe -c port=usb1 -fwdelete*
- STEP 4 : Download new wireless stack :
    - *STM32_Programmer_CLI.exe -c port=usb1 -fwupgrade [Wireless_Coprocessor_Binary] [Install address] firstinstall=1*
- Please check **Binary Install Address Table** for Install@ parameter depending of the binary.
- STEP 5 : Revert STEP 2 procedure to put back device in normal mode.

Detailed informations and instructions for STM32CubeProgrammer in the manual: https://www.st.com/content/ccc/resource/technical/document/user_manual/group0/76/3e/bd/0d/cf/4d/45/25/DM00403500/files/DM00403500.pdf/jcr:content/translations/en.DM00403500.pdf

Binary Install Address and version : Provides Install address for the targeted binary to be used in "STEP 4" of flash procedure.

Wireless Processor Binary	Install address	Version	Date
stm32wb5x_BLE_Stack_fw.bin	0x080CB000	v1.0.0	02/06/2019
stm32wb5x_BLE_HCILayer_fw.bin	0x080CD000	v1.0.0	02/06/2019
stm32wb5x_Thread_FTD_fw.bin	0x0809F000	v1.0.0	02/06/2019
stm32wb5x_Thread_MTD_fw.bin	0x080B5000	v1.0.0	02/06/2019
stm32wb5x_BLE_Thread_fw.bin	0x08079000	v1.0.0	02/06/2019
stm32wb5x_Mac_802_15_4_fw.bin	0x080E5000	v1.0.0	02/06/2019
stm32wb5x_rfmonitor_phy802_15_4_fw.bin	0x080EA000	v1.0.0	02/06/2019

Table 4: Binary install address and version

7 Software Development

7.1 Atollic TrueStudio IDE

The firmware can be developed and uploaded with STLink V2 or V3 device using:

- STM32CubeMX v5.10 or superior code generator that can be downloaded from STMicroelectronics website at https://www.st.com/en/development-tools/stm32cubemx.html
- Atollic TrueSTUDIO for STM32 free compiler based on Eclipse IDE that can be downloaded from STMicroelectronics website at https://atollic.com/

The developed application runs on the M4 core and interfaces to the communication stack on M0+ core using the communication functions provided by ST.

IN STM32CubeMX tool selsct the STM32WB55CGUx MCU from MCU Selector.

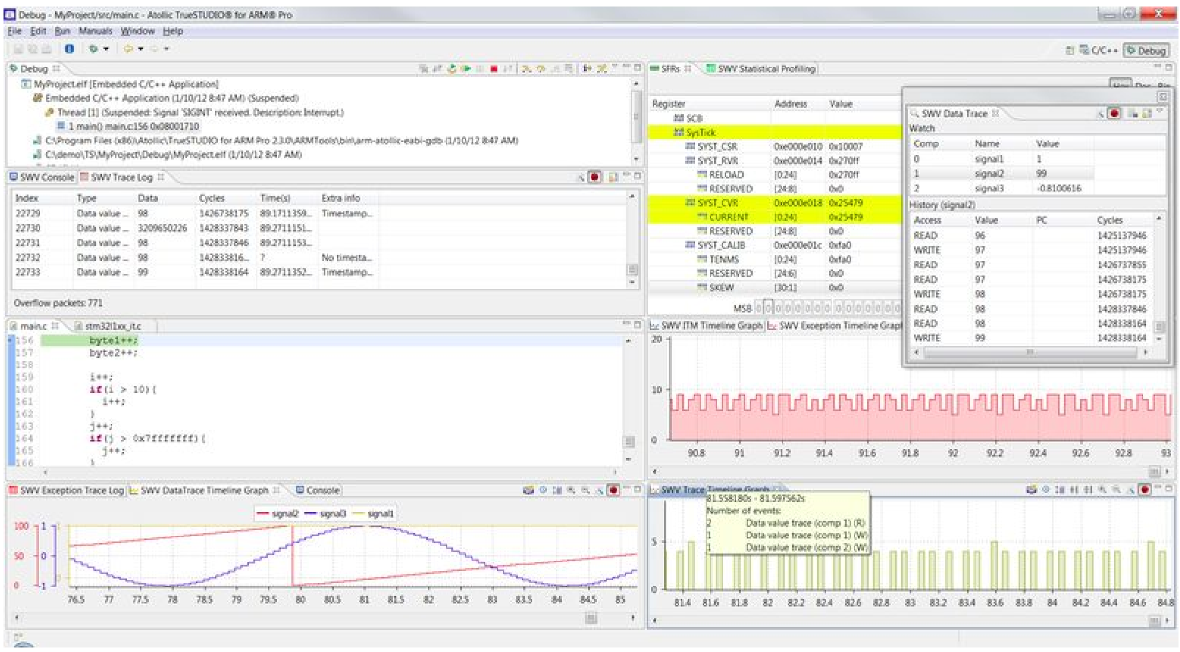

Figure 18. Atollic TrueStudio for STM32 IDE

In order to develop a custom firmware to be uploaded to the Sharky Module the following tools are necessary:

- A Windows/Linux/MacOS PC
- STM32CubeMX

- Atollic TrueSTUDIO for STM32
- STLink V2 or V3 device
  https://www.st.com/content/st_com/en/products/development-tools/hardware-development-tools/hardware-development-tools-for-stm32/st-link-v2.html

The ST-LINK/V2 is an in-circuit debugger and programmer for the STM8 and STM32 microcontroller families. The single wire interface module (SWIM) and JTAG/serial wire debugging (SWD) interfaces are used to communicate with any STM8 or STM32 microcontroller located on an application board.

7.3 Arduino IDE

8 References and Useful Links

8.1 Data Sheets and documents

- https://www.st.com/en/microcontrollers/stm32wb-series.html?querycriteria=productId=SS1961
- https://www.st.com/resource/en/data_brief/stm32wb55cg.pdf
- https://www.st.com/resource/en/datasheet/stm32wb55cg.pdf
- https://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32-wireless-mcus/stm32wb-series/stm32wbx5/stm32wb55cg.html

8.2 Tools

- https://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32-wireless-mcus/stm32wb-series/stm32wbx5/stm32wb55cg.html#sw-tools-scroll

8.3 WebSites

- http://www.midatronics.com
- https://www.st.com
- https://atollic.com/