KAMAMI

Kamod BluePill

Rev. 20241026162139
Źródło: https://wiki.kamamilabs.com/index.php/Kamod_BluePill

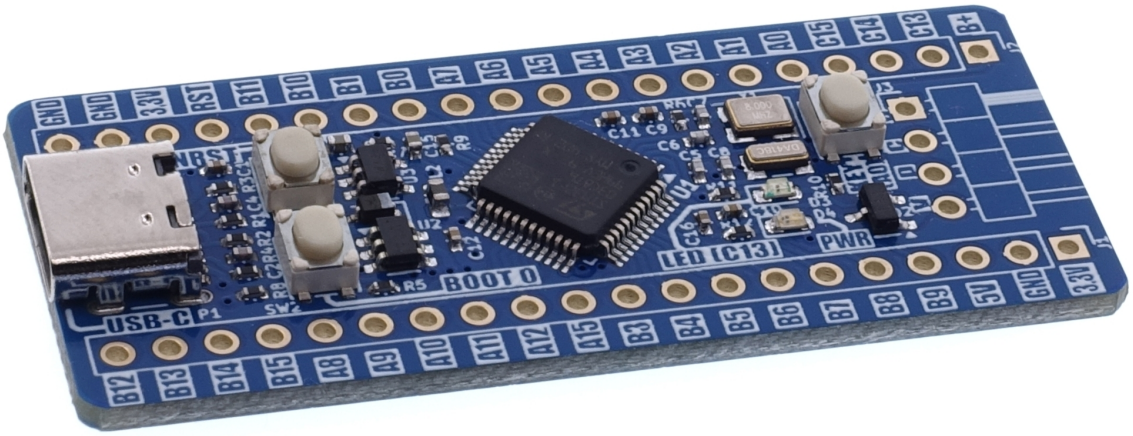
Spis treści

Description

Kamod BluePill+

Evaluation board with STM32F103C8T6 microcontroller, compatible with BluePill

The Kamod BluePill+ evaluation board contains the STM32F103C8T6 microcontroller and the components necessary to run and program it. The board is pin-compatible with the BluePill project, but has a number of improvements, including: new PCB design, USB-C connector with ESD protection, or improved power supply circuit. It can be programmed with Arduino IDE, because the system memory contains a suitable bootloader.

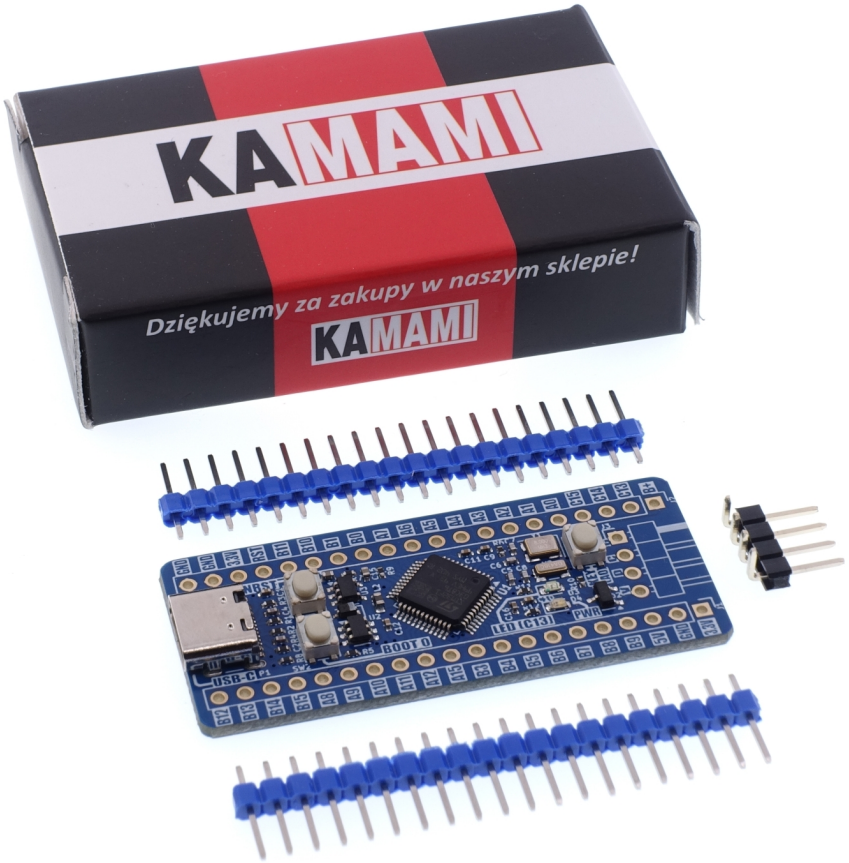


Basic features and parameters

- STM32F103C8T6 microcontroller: 64 kB Flash, 20 kB RAM, 72 MHz, 2 x 12-bit ADC, 4 timers, 2 x I2C, 2 x SPI, 3 x UART, CAN, USB, RTC
- USB-C connector, which serves as a power connector, USB communication interface and allows for microcontroller programming
- elements filtering interference and overvoltages on USB interface lines
- 32 GPIO pins and 5 V and 3.3 V power lines available on standard 2.54 mm pitch connectors
- maximum load on the 5 V line is 500 mA, while for the 3.3 V line it is 200 mA
- precise resonator clocking the microcontroller and an independent resonator for the RTC module
- possibility of connecting a battery to support the RTC module
- SWD programming/debugging interface connector
- possibility of programming via STM32CubeIDE and Arduino IDE
- board dimensions: 53.5x23 mm, height approx. 7 mm (without soldered goldpins)

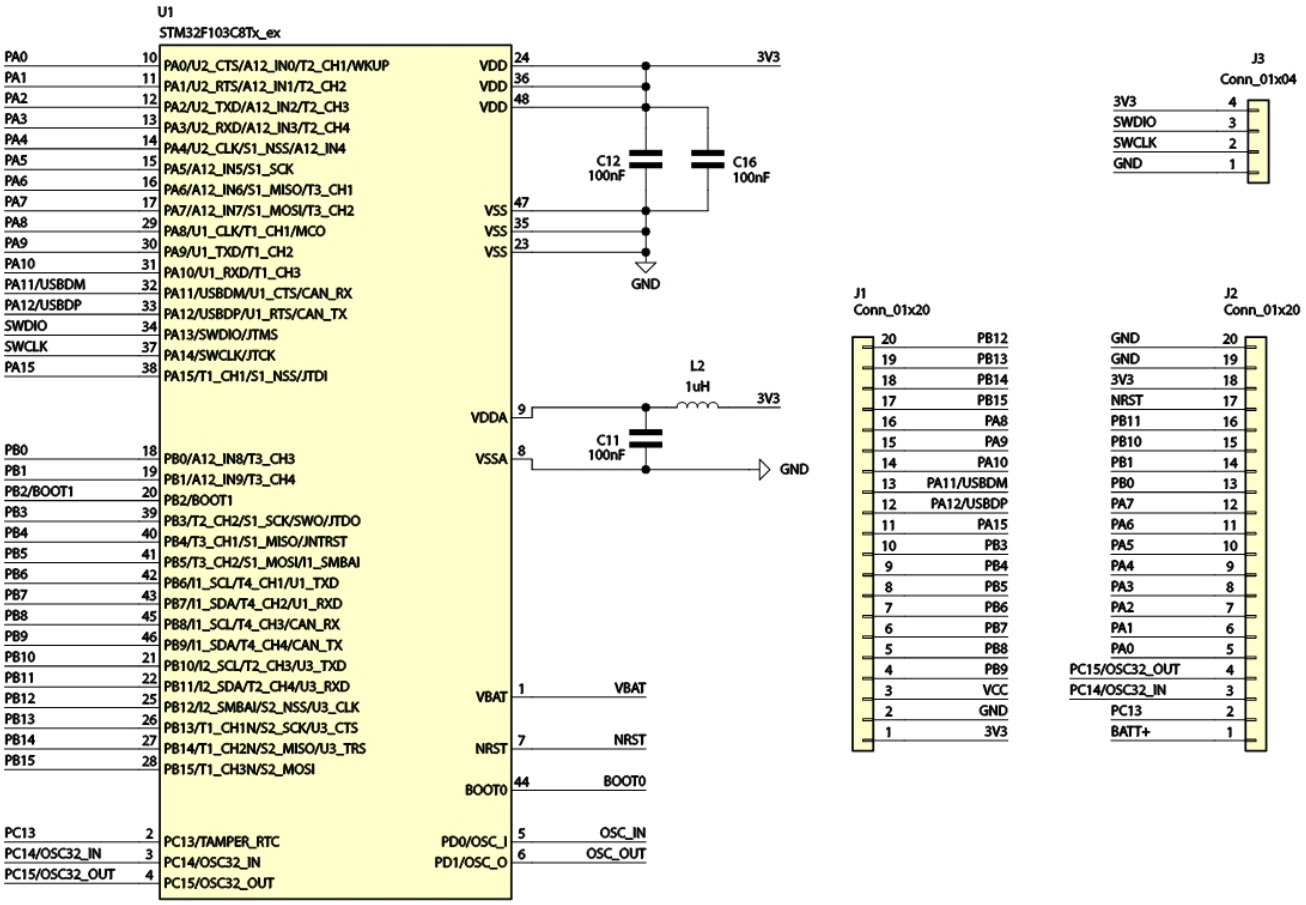
Standard equipment

Code	Description
Kamod BluePill+	• Assembled and started module, with bootloader uploaded • 2 x straight goldpin header 20-pin raster 2.54 mm • 1 x angled goldpin header 4-pin raster 2.54 mm

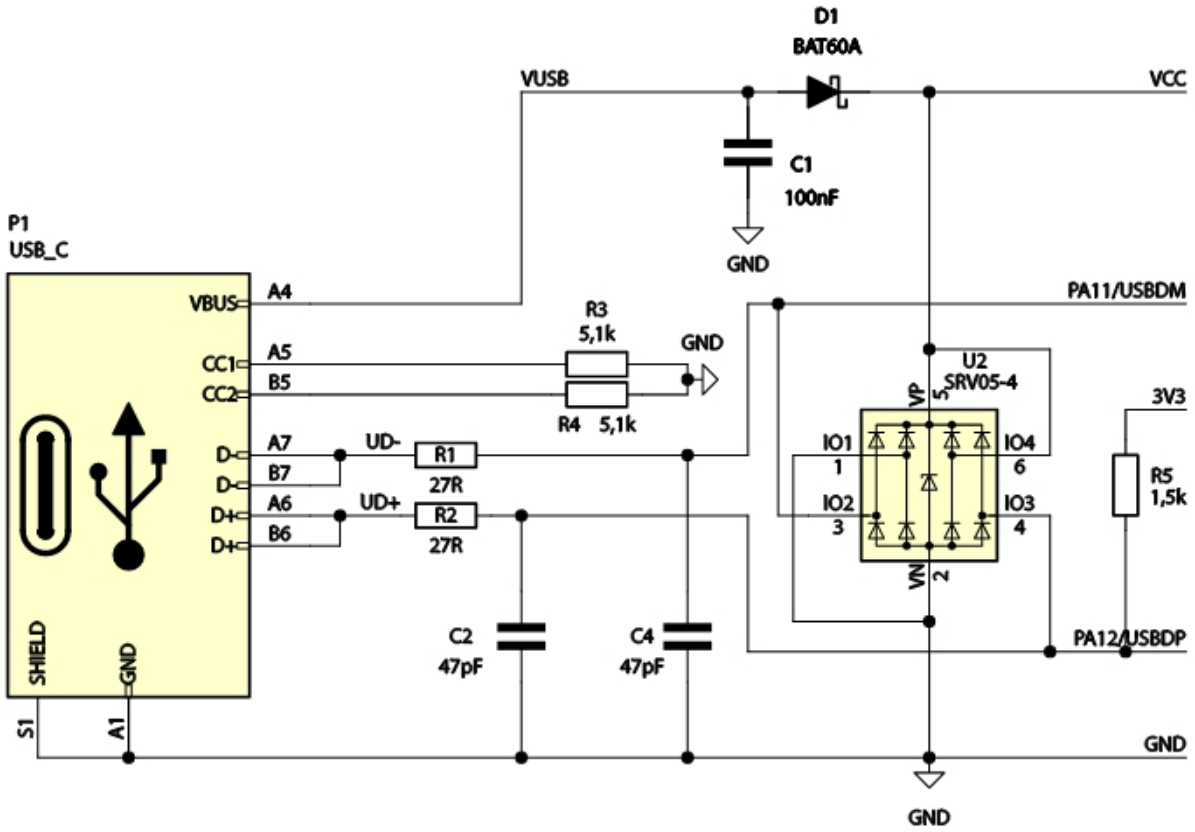


Electrical diagram

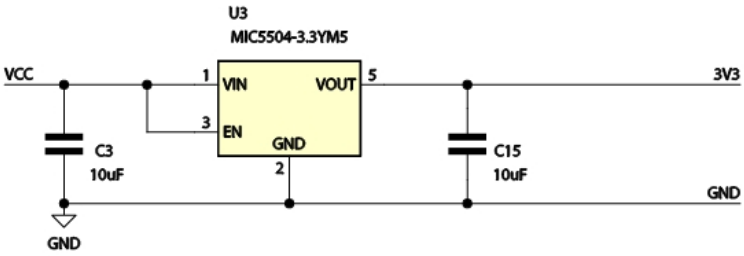
Microcontroller and GPIO and SWD connectors

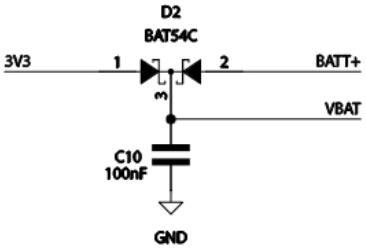


USB interface

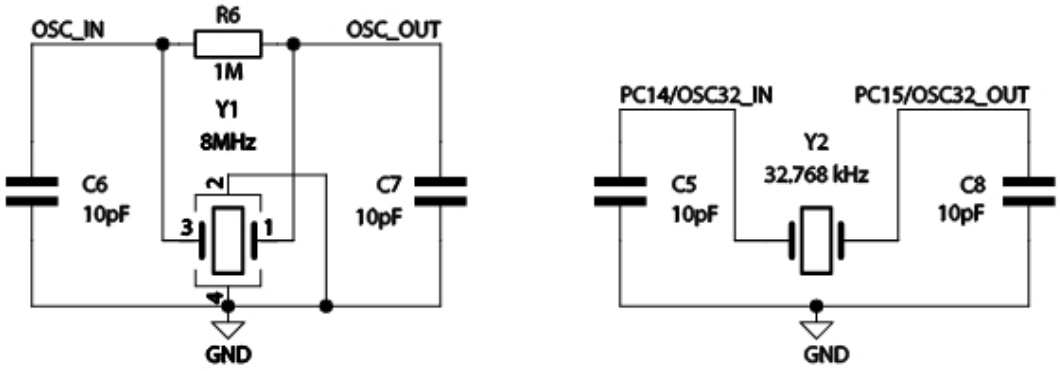


Power supply circuit

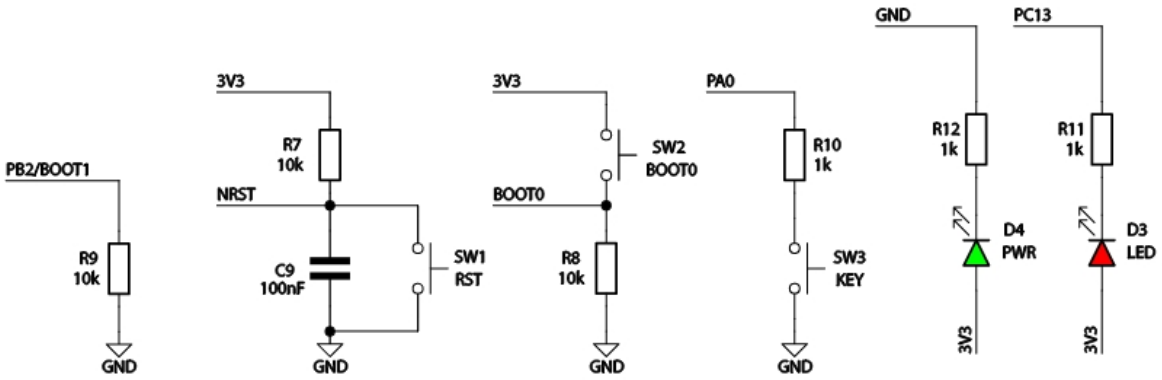




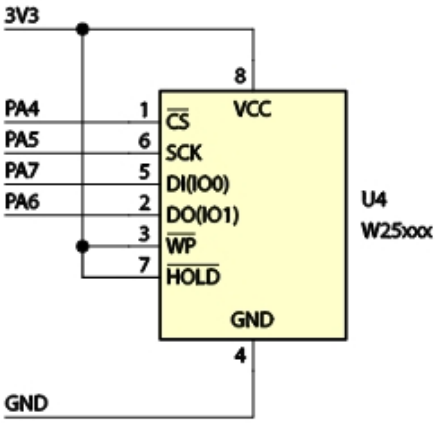
Microcontroller clock speed



Additional elements



External memory


3V3
8
VCC
PA4
1
CS
PA5
6
SCK
PA7
5
DI(IO0)
PA6
2
DO(IO1)
3
WP
7
HOLD
GND
4
GND
U4
W25xxx

Power

Connector	Function
USB-C J1, J2	• Supplies 5 V power to the module • Allows 5 V power and provides 3.3 V

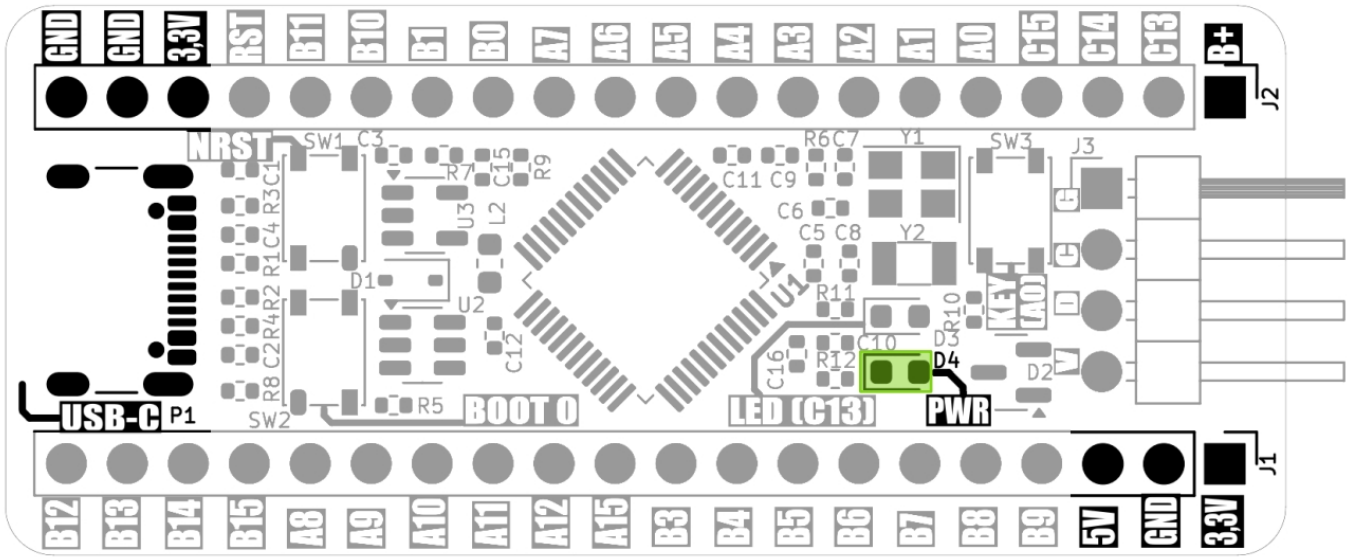
The **Kamod BluePill+** evaluation board can be powered in two ways:

- via the appropriate contacts of J1 and J2 connectors,
- via the USB-C connector.

A power source with a voltage in the range of 4.5...5.5 V and an efficiency of min. 100 mA should be connected to the contacts marked ' ***5V*** (plus) and ***GND*** (minus) on connectors J1 and/or J2. Then, on the contact marked ***3.3V***, a stabilized voltage of 3.3 V is available, which also powers the microcontroller. The presence of 3.3 V is indicated by the LED marked ***PWR***' lighting up.

A standard USB power source with a capacity of at least 100 mA should be connected to the USB-C connector. Then, on the 5V contact of connector J1, a voltage of close to 5 V is available (relative to the ground marked ***GND***). A small voltage drop (approx. 0.5 V) occurs on the Schottky diode, which allows current to flow in the direction from the USB-C connector to the board, but blocks current flow in the opposite direction - to the USB-C connector. Thanks to this, you can safely connect power in various configurations - USB and/or J1, J2 contacts.

There is a contact marked ***B+***' on the J2 connector. Together with the ground ***GND*** this is the power input from the battery supporting the RTC clock (integrated with the microcontroller). The backup battery voltage should be in the range of 1.8...3.6 V. Detailed information on the operation of the RTC module can be found in the documentation for the STM32F103C8T6 microcontroller.



USB interface

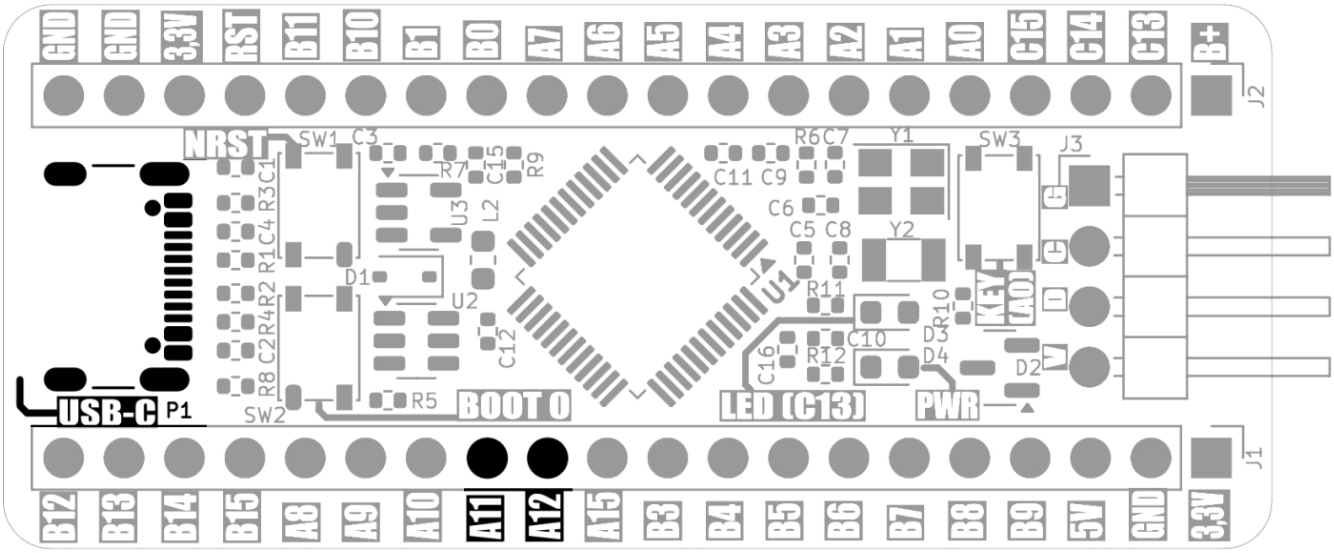
Connector	Ffunction
USB-C	• Provides 5V power to the module • Can implement USB 2.0 FS Peripheral interface • Allows programming of microcontroller Flash memory

The USB-C connector is an easy way to provide power to the Kamod BluePill+ board. In addition, the STM32F103C8T6 microcontroller has an integrated USB 2.0 FS interface controller, which can be used for communication, e.g. in VCP (*Virtual COM Port*) or CDC (*Communication Device Class*) mode.

The board has been configured in such a way that it allows the interface to operate in Peripheral mode, it is not prepared to work in Host mode. The board also contains elements that filter possible interference and overvoltages on the USB interface lines, which ensure its stable operation.

Additional functionality of the USB-C connector is the ability to program the microcontroller's Flash memory. This requires an appropriate bootloader, which is loaded into the Kamod BluePill+ module microcontroller along with the test program.

The USB interface lines (DP and DM) are also GPIO ports PA12 and PA11 of the microcontroller. If we use USB, the contacts marked *A12* and *A11* cannot perform any other function - they must remain unconnected.



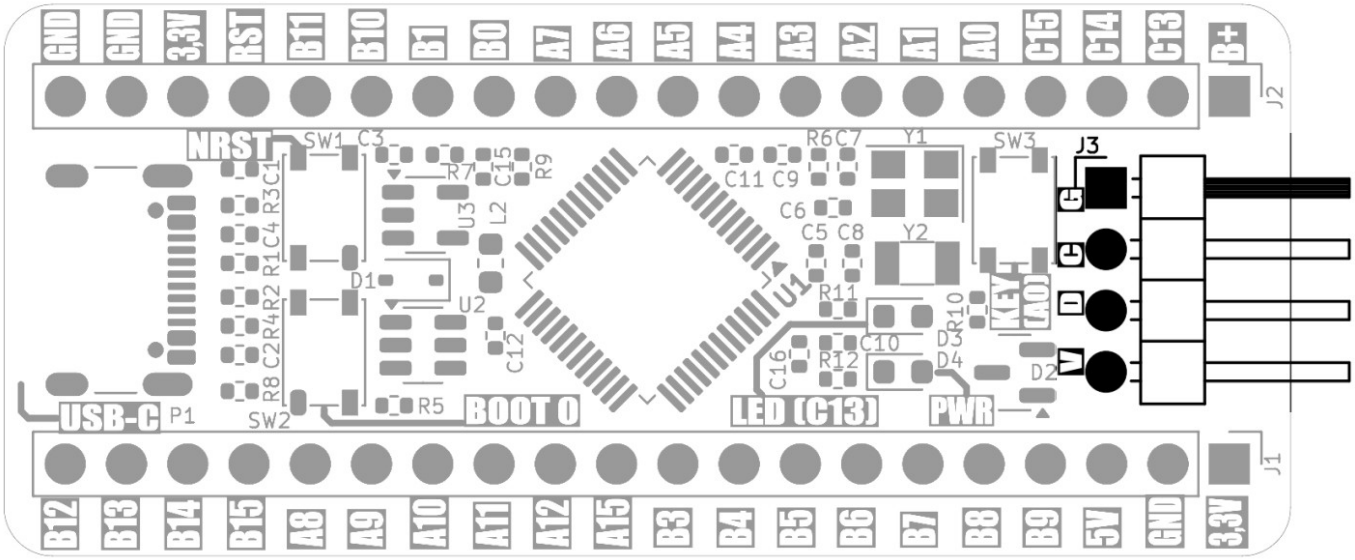
SWD programming/debugging interface

Connector	Function
J3	• SWD (*Single Wire Debug*) interface allows for programming the microcontroller's Flash memory and tracking the program's operation (debugging). Requires the connection of an external programmer/debugger, e.g. STLINK-V2 or STLINK-V3MINIE.

The SWD interface has been led out to the J3 pin connector. The signals are described as follows:

- G – system ground,
- C – SWCLK clock signal,
- D – SWDIO data signal,
- V – 3.3 V power line.

The signals should be connected to the same signals on the programmer/debugger connector. Sometimes SWCLK is also marked as TCK, while SWDIO is also marked as TMS. The programmer does not supply power to the Kamod BluePill+ board, the power should be connected to the USB-C connector or J1/J2 pins.



Microcontroller programming

Component	Function
SW1 – NRST	• Forcing the microcontroller to zero state
SW2 – BOOT0	• Starting the factory microcontroller bootloader
J3 – SWD	• SWD (Single Wire Debug) programming interface
A9, A10 – UART	• UART (TXD, RXD) programming interface

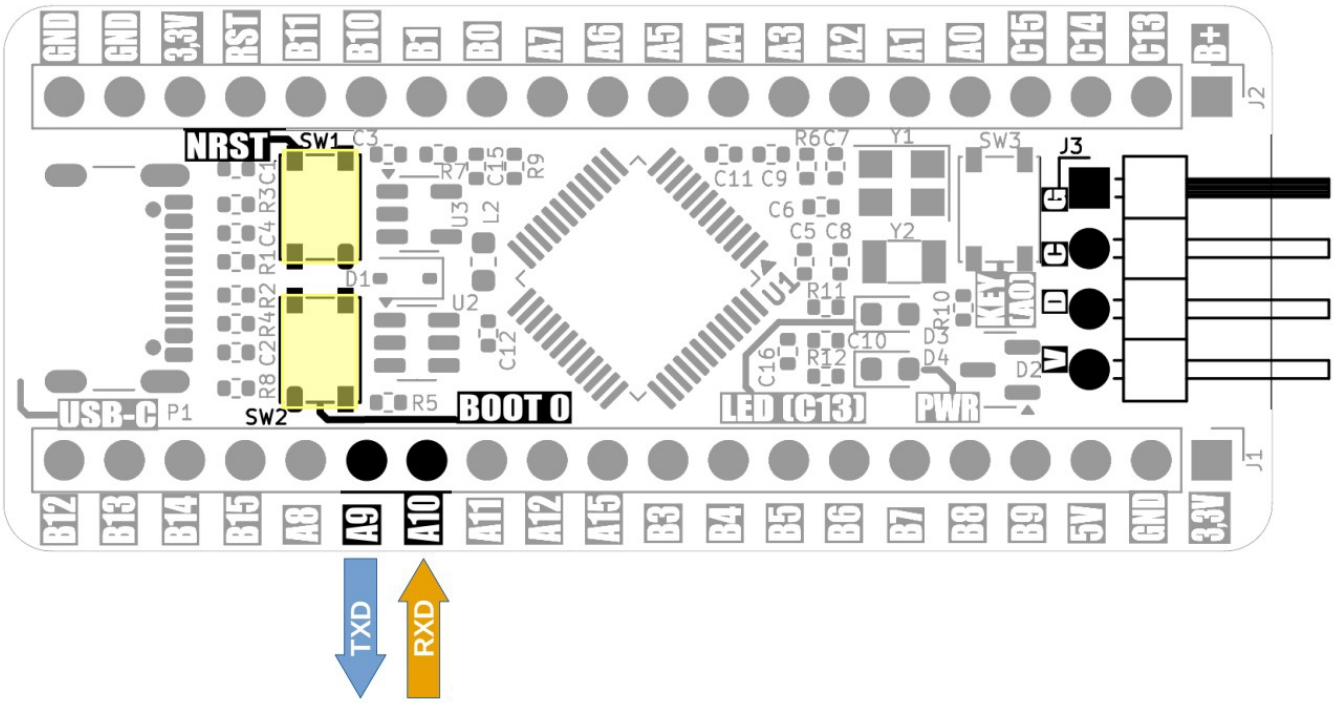
A special area of the microcontroller's memory contains software that allows programming its program memory - this is the so-called bootloader prepared by the microcontroller manufacturer. To start the bootloader, perform the following sequence of actions, with the board's power supply connected:

1. Press and hold the NRTS button
2. Press and hold the BOOT 0 button while holding the NRST button
3. Release the NRTS button while holding the BOOT 0 button
4. Release the BOOT 0 button

Starting the bootloader activates two programming interfaces:

- SWD, which was described in the previous chapter and allows programming the microcontroller using a programmer/debugger, e.g. STLINK-V2 or STLINK-V3MINIE.
- UART, which allows you to use a regular USB-UART converter as a programmer, e.g. KAmod USB-UART-mini. Then you need to connect the UART interface lines to the A9 and A10 pins, where:
  - A9 – the TXD output of the microcontroller should be connected to the RXD input of the USB-UART converter,
  - A10 – the RXD input of the microcontroller should be connected to the TXD output of the USB-UART converter.

Now you can program the microcontroller via SWD or UART using the STM32CubeProgrammer, which is available on the official STMicroelectronics website: https://www.st.com/en/development-tools/stm32cubeprog.html

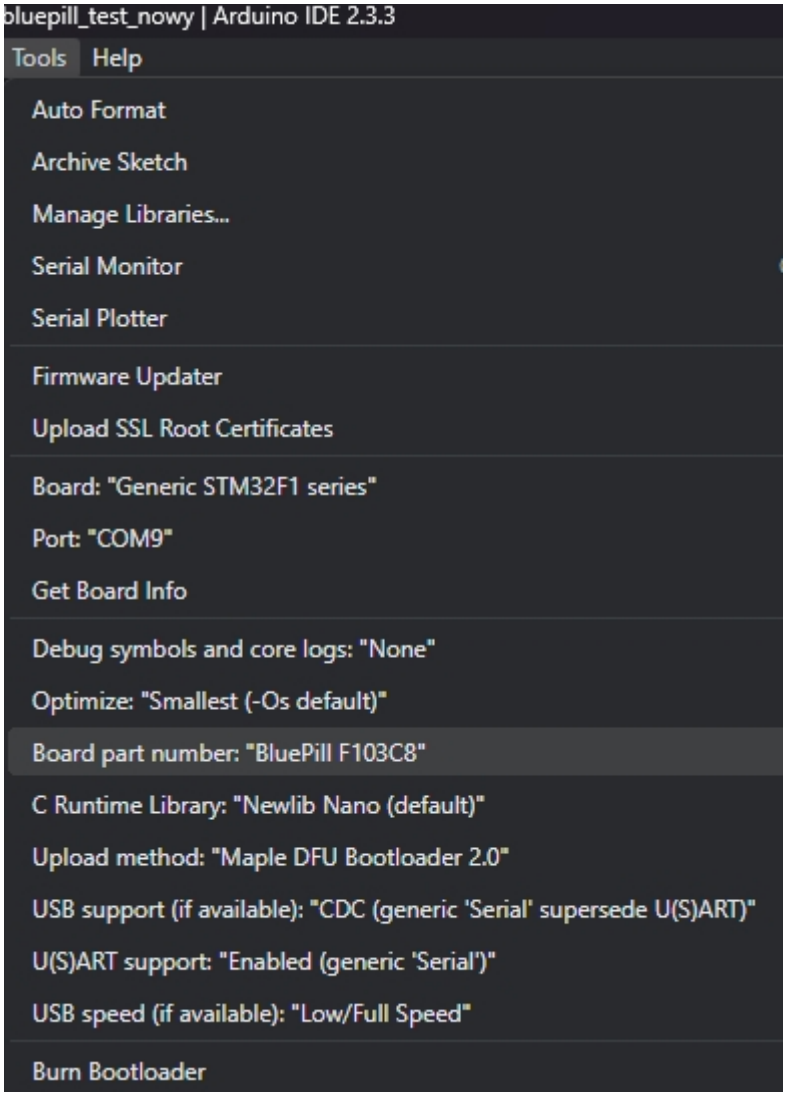


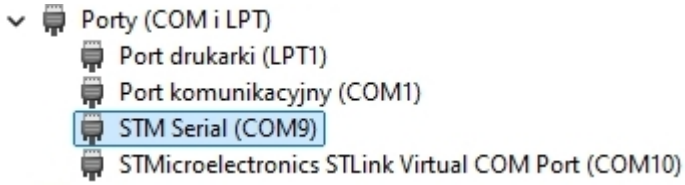
The **Kamod BluePill+** board also allows you to program the microcontroller via the USB interface using the Arduino IDE. This is possible because, along with the test program, an additional bootloader is placed in the program memory for communication and programming in the Arduino environment.duino.Then you should:

- add the following address to the sources of the board manager (Preferences) in the Arduino IDE:https://github.com/stm32duino/BoardManagerFiles/raw/main/package_stmicroelectronics_index.json
- via the board manager (Boards Manager) install the package "STM32 MCU based boards"
- install STM32CubeProgrammer (https://www.st.com/en/development-tools/stm32cubeprog.html)

Then, after creating a new sketch, in the Tools tab:

- select the board: *BluePill F103C8,*
- set the Upload method to: *Maple DFU Bootloader 2.0,*
- USB support should be set to: CDC *(generic Serial),*
- U(S)ART support should be set to: Enabled *(generic Serial),*
- select STM Serial communication port *(COMxx).*


bluepill_test_nowy | Arduino IDE 2.3.3
Tools
Help
Auto Format
Archive Sketch
Manage Libraries...
Serial Monitor
Serial Plotter
Firmware Updater
Upload SSL Root Certificates
Board: "Generic STM32F1 series"
Port: "COM9"
Get Board Info
Debug symbols and core logs: "None"
Optimize: "Smallest (-Os default)"
Board part number: "BluePill F103C8"
C Runtime Library: "Newlib Nano (default)"
Upload method: "Maple DFU Bootloader 2.0"
USB support (if available): "CDC (generic 'Serial' supersede U(S)ART)"
U(S)ART support: "Enabled (generic 'Serial')"
USB speed (if available): "Low/Full Speed"
Burn Bootloader


Porty (COM i LPT)
Port drukarki (LPT1)
Port komunikacyjny (COM1)
STM Serial (COM9)
STMicroelectronics STLink Virtual COM Port (COM10)

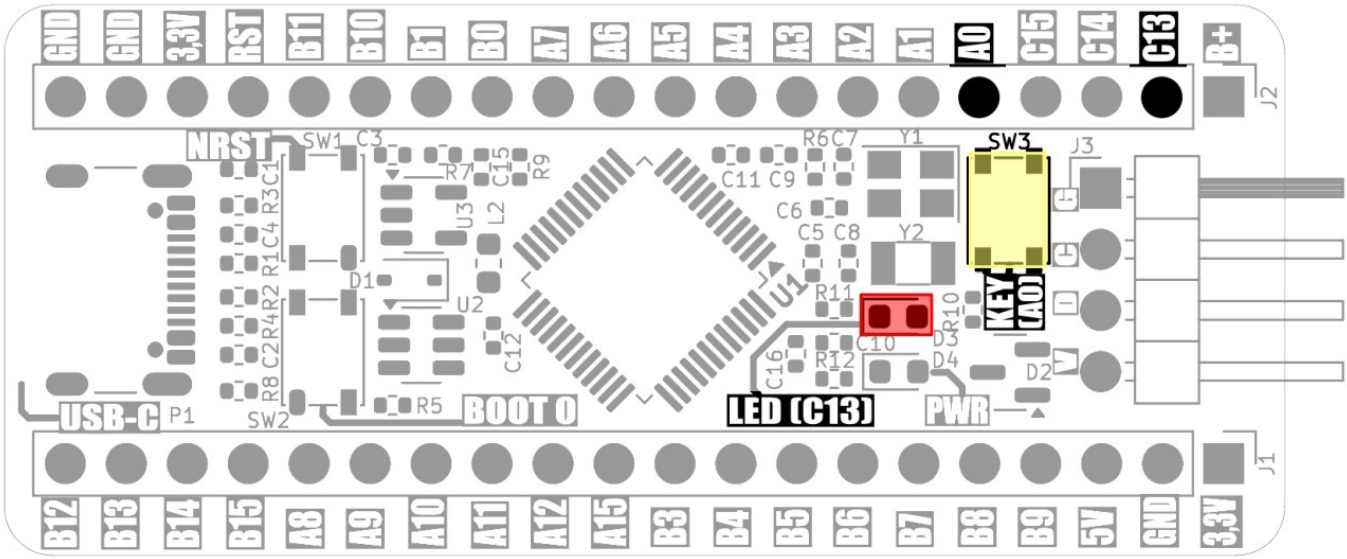
Additional elements – LED and button

Component	Function
D3 – LED (C13)	• LED connected to port PC13, active in L state
SW3 – KEY (A0)	• Microswitch connected to port PA0, active in L state

The **Kamod BluePill+** board contains additional elements – LED and microswitch, which can be used in the target application.

The LED is connected to port PC13, it lights up when the state is low.

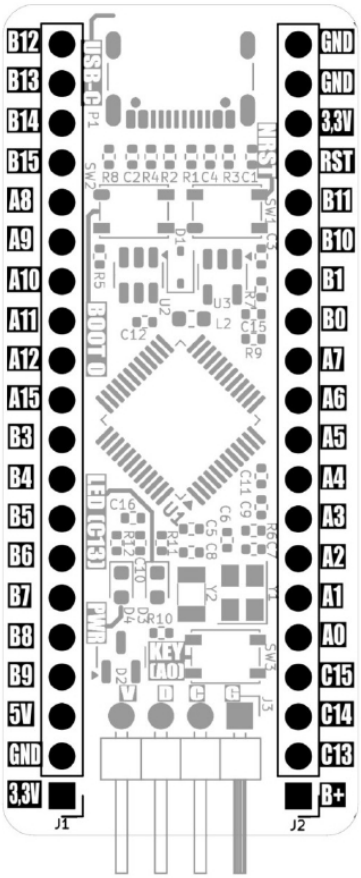
The microswitch is connected to port PA0, pressing it forces the state to be low on this port. The button is connected via a 1k resistor, so there is no possibility of damaging port PA0 configured as an output.



GPIO connectors

Connector	Function
J1, J2	• 20-pin connectors with a 2.54 mm pitch, to which the microcontroller's GPIO ports and 5 V and 3.3 V power supply are connected

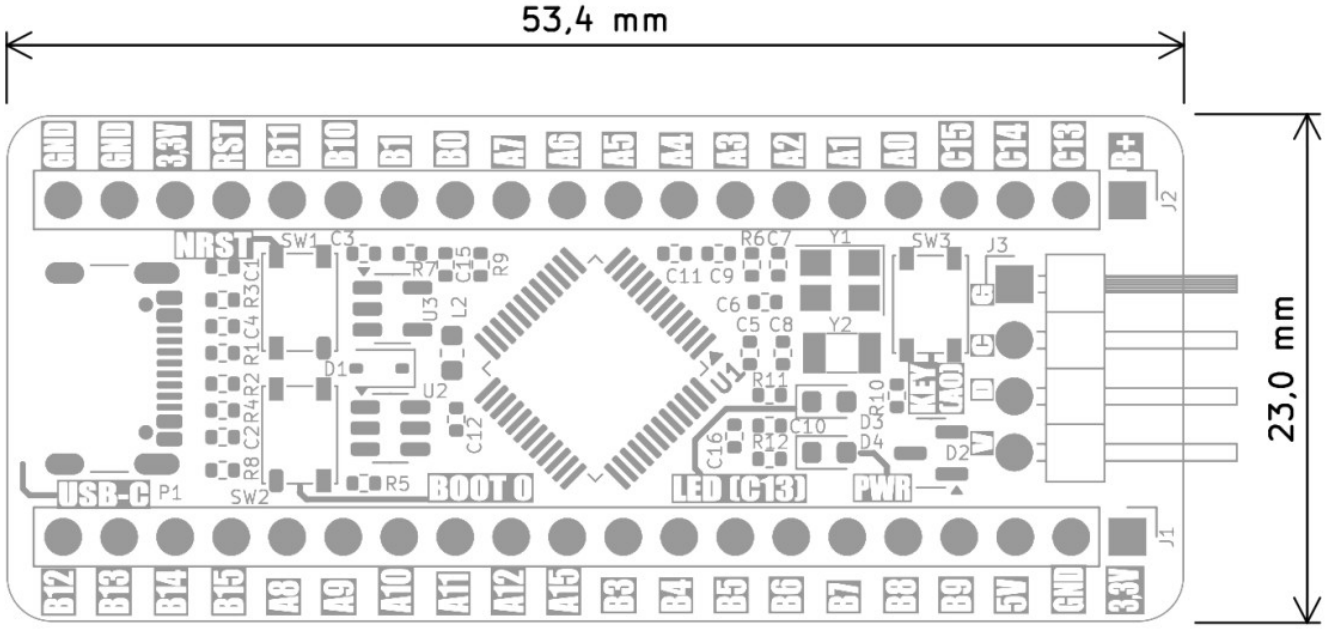
GPIO connectors J1 and J2 contain 20 pins each, to which the 5 V, 3.3 V power supply lines, GND ground and the GPIO pins of the microcontroller module are connected. The exact description of the pins and their additional functions are shown in the drawing and table below:



J1		J2
PB12 (SPI2_NSS)		GND
PB13 (SPI2_SCK)		GND
PB14 (SPI2_MISO)		3,3 V
PB15 (SPI2_MOSI)		RESET
PA8 (TIMER1_CH1)		(UART3_RXD) PB11
PA9 (UART1_TXD)		(UART3_TXD) PB10
PA10 (UART1_RXD)		(ADC12_IN9) PB1
PA11 (USB_DM)		(ADC12_IN8) PB0
PA12 (USB_DP)		(ADC12_IN7) PA7
PA15 (SPI1_NSS)		(ADC12_IN6) PA6
PB3 (SPI1_SCK)		(ADC12_IN5) PA5
PB4 (SPI1_MISO)		(ADC12_IN4) PA4
PB5 (SPI1_MOSI)		(TIMER2_CH4) PA3
PB6 (I2C1_SCL)		(TIMER2_CH3) PA2
PB7 (I2C1_SDA)		(TIMER2_CH2) PA1
PB8 (CAN_RX)		(TIMER2_CH1) PA0
PB9 (CAN_TX)		(OSC_RTC_O) PC15
5 V		(OSC_RTC_I) PC14
GND		(LED_D3) PC13
3,3 V		RTC_BATTERY +

Dimensions

The dimensions of the **Kamod BluePill+** board are 53.5x23 mm, and the height is approx. 7 mm (without soldered goldpins).



Test program

The simplified code of the test program is below, it can be compiled in the Arduino environment.

```
#include <STM32RTC.h>

#define LED_PIN    PC13
#define SW_PIN     PA0


int message_period = 0;
int config_state = 0;
int ports_state = 0;
int pindex = 0;


STM32RTC& rtc = STM32RTC::getInstance();

void setup() {

 Serial.begin(115200);
 pinMode(LED_PIN, OUTPUT);
 pinMode(SW_PIN, INPUT_PULLUP);
 rtc.begin(); // initialize RTC 24H format

}

// the loop function runs over and over again forever
void loop() {

 //digitalWrite(LED_PIN, HIGH);
 if (digitalRead(SW_PIN) == HIGH){
   if ((ports_state&4) > 0){
     digitalWrite(LED_PIN, HIGH);
   } else {
     digitalWrite(LED_PIN, LOW);
   }

   if (config_state > 0){
     IO2input();
     config_state = 0;
   }

 } else {
   if ((ports_state&1) > 0){
     digitalWrite(LED_PIN, HIGH);
   } else {
     digitalWrite(LED_PIN, LOW);
   }

   if (config_state == 0){
     IO2output();
     config_state = 1;
   }



   if (config_state == 1){
     IO2low();
     IOwriteIndexed(pindex, 1);
     pindex++;
```

```
      if (pindex > 26) pindex = 1;
    }
  }

  // check if plugged into a host
  if (message_period >= 11){
    message_period = 0;

    int tmt = rtc.getMinutes();
    Serial.print("KAmod BluePill+ RTC=");
    if (tmt < 10) Serial.print("0");
    Serial.print(tmt);

    tmt = rtc.getSeconds();
    Serial.print(":");
    if (tmt < 10) Serial.print("0");
    Serial.println(tmt);
  }

  delay(200);
  message_period++;
  ports_state++;

}

void IO2output(){

  //pinMode(PA0, OUTPUT); //switch
  pinMode(PA1, OUTPUT);
  pinMode(PA2, OUTPUT);
  pinMode(PA3, OUTPUT);
  pinMode(PA4, OUTPUT);
  pinMode(PA5, OUTPUT);
  pinMode(PA6, OUTPUT);
  pinMode(PA7, OUTPUT);

  pinMode(PA8, OUTPUT);
  pinMode(PA9, OUTPUT);
  pinMode(PA10, OUTPUT);
  pinMode(PA15, OUTPUT);

  pinMode(PB0, OUTPUT);
  pinMode(PB1, OUTPUT);

  pinMode(PB3, OUTPUT);
  pinMode(PB4, OUTPUT);
  pinMode(PB5, OUTPUT);
  pinMode(PB6, OUTPUT);
  pinMode(PB7, OUTPUT);
  pinMode(PB8, OUTPUT);
  pinMode(PB9, OUTPUT);

  pinMode(PB10, OUTPUT);
  pinMode(PB11, OUTPUT);

  pinMode(PB12, OUTPUT);
  pinMode(PB13, OUTPUT);
  pinMode(PB14, OUTPUT);
  pinMode(PB15, OUTPUT);
  //pinMode(PC13, OUTPUT);  //led

}
```

```
void IO2input(){

 //pinMode(PA0, INPUT); //switch
 pinMode(PA1, INPUT);
 pinMode(PA2, INPUT);
 pinMode(PA3, INPUT);
 pinMode(PA4, INPUT);
 pinMode(PA5, INPUT);
 pinMode(PA6, INPUT);
 pinMode(PA7, INPUT);

 pinMode(PA8, INPUT);
 pinMode(PA9, INPUT);
 pinMode(PA10, INPUT);
 pinMode(PA15, INPUT);

 pinMode(PB0, INPUT);
 pinMode(PB1, INPUT);

 pinMode(PB3, INPUT);
 pinMode(PB4, INPUT);
 pinMode(PB5, INPUT);
 pinMode(PB6, INPUT);
 pinMode(PB7, INPUT);
 pinMode(PB8, INPUT);
 pinMode(PB9, INPUT);

 pinMode(PB10, INPUT);
 pinMode(PB11, INPUT);

 pinMode(PB12, INPUT);
 pinMode(PB13, INPUT);
 pinMode(PB14, INPUT);
 pinMode(PB15, INPUT);

 //pinMode(PC13, INPUT);  //led

}

void IO2low(){

 //digitalWrite(PA0, LOW); //switch
 digitalWrite(PA1, LOW);
 digitalWrite(PA2, LOW);
 digitalWrite(PA3, LOW);
 digitalWrite(PA4, LOW);
 digitalWrite(PA5, LOW);
 digitalWrite(PA6, LOW);
 digitalWrite(PA7, LOW);

 digitalWrite(PA8, LOW);
 digitalWrite(PA9, LOW);
 digitalWrite(PA10, LOW);
 digitalWrite(PA15, LOW);

 digitalWrite(PB0, LOW);
 digitalWrite(PB1, LOW);

 digitalWrite(PB3, LOW);
 digitalWrite(PB4, LOW);
 digitalWrite(PB5, LOW);
 digitalWrite(PB6, LOW);
 digitalWrite(PB7, LOW);
```

```
 digitalWrite(PB8, LOW);
 digitalWrite(PB9, LOW);

 digitalWrite(PB10, LOW);
 digitalWrite(PB11, LOW);

 digitalWrite(PB12, LOW);
 digitalWrite(PB13, LOW);
 digitalWrite(PB14, LOW);
 digitalWrite(PB15, LOW);

 //digitalWrite(PC13, LOW);

}

void IO2high(){

 //digitalWrite(PA0, HIGH); //switch
 digitalWrite(PA1, HIGH);
 digitalWrite(PA2, HIGH);
 digitalWrite(PA3, HIGH);
 digitalWrite(PA4, HIGH);
 digitalWrite(PA5, HIGH);
 digitalWrite(PA6, HIGH);
 digitalWrite(PA7, HIGH);

 digitalWrite(PA8, HIGH);
 digitalWrite(PA9, HIGH);
 digitalWrite(PA10, HIGH);
 digitalWrite(PA15, HIGH);

 digitalWrite(PB0, HIGH);
 digitalWrite(PB1, HIGH);

 digitalWrite(PB3, HIGH);
 digitalWrite(PB4, HIGH);
 digitalWrite(PB5, HIGH);
 digitalWrite(PB6, HIGH);
 digitalWrite(PB7, HIGH);
 digitalWrite(PB8, HIGH);
 digitalWrite(PB9, HIGH);

 digitalWrite(PB10, HIGH);
 digitalWrite(PB11, HIGH);

 digitalWrite(PB12, HIGH);
 digitalWrite(PB13, HIGH);
 digitalWrite(PB14, HIGH);
 digitalWrite(PB15, HIGH);

 //digitalWrite(PC13, HIGH);

}

void IOwriteIndexed(int index, int state){

 int pp = 1;

 switch (index){
   //case 0: pp = PA0; break;
   case 1: pp = PA1; break;
   case 2: pp = PA2; break;
   case 3: pp = PA3; break;
   case 4: pp = PA4; break;
```

```
    case 5: pp = PA5; break;
    case 6: pp = PA6; break;
    case 7: pp = PA7; break;

    case 8: pp = PB0; break;
    case 9: pp = PB1; break;
    case 10: pp = PB10; break;
    case 11: pp = PB11; break;

    case 12: pp = PB12; break;
    case 13: pp = PB13; break;
    case 14: pp = PB14; break;
    case 15: pp = PB15; break;

    case 16: pp = PA8; break;
    case 17: pp = PA9; break;
    case 18: pp = PA10; break;

    //case 19: pp = PA11; break
    //case 20: pp = PA12; break

    case 19: pp = PA15; break;

    case 20: pp = PB3; break;
    case 21: pp = PB4; break;
    case 22: pp = PB5; break;
    case 23: pp = PB6; break;

    case 24: pp = PB7; break;
    case 25: pp = PB8; break;
    case 26: pp = PB9; break;
  }

  pinMode(pp, OUTPUT);
  if (state > 0){
    digitalWrite(pp, HIGH);
  } else {
    digitalWrite(pp, LOW);
  }

}
```

The program works on starting the serial port based on the USB interface and the RTC clock module and cyclically sending time information via this serial port.

The program is run by slow flashing of the D3 LED. When the KEY (SW3) button is pressed, all ports the high state will be set and the LED will start flashing quickly.

Links

- CAD model (STEP)
- Catalog card of the STM32F103C8T6 system
- Arduino test program
- Test project STM32CUBE_IDE
- STLINK-V3MINIE - compact programmer/debugger for STM32
- STM32CubeProgrammer
- KAmod USB-UART-mini - Miniature USB-UART converter

BTC Korporacja
05-120 Legionowo
ul. Lwowska 5
tel.: (22) 767-36-20
faks: (22) 767-36-33
e-mail:
sprzedaz@kamami.pl
https://kamami.pl