



# KAmoD BlackPill 411 (PL)



Rev. 20241015115047

Źródło: [https://wiki.kamamilabs.com/index.php/KAmoD\\_BlackPill\\_411\\_\(PL\)](https://wiki.kamamilabs.com/index.php/KAmoD_BlackPill_411_(PL))

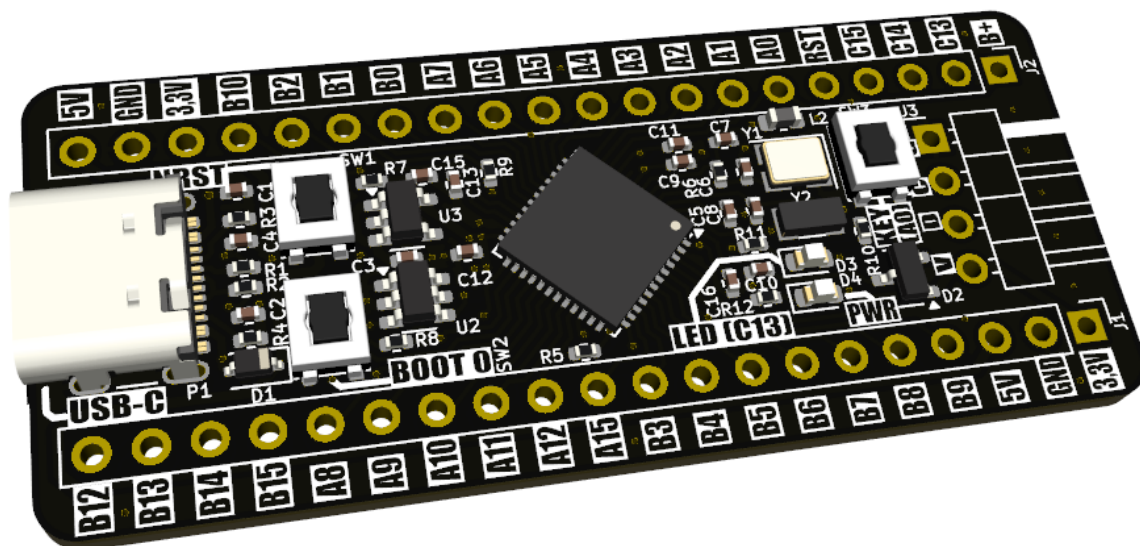
**Spis treści**

Opis .....	1
Podstawowe cechy i parametry .....	2
Wyposażenie standardowe .....	3
Schemat elektryczny .....	4
Zasilanie .....	8
Interfejs USB .....	9
Interfejs programowania/debugowania SWD .....	10
Programowanie mikrokontrolera .....	11
Dodatkowe elementy - dioda LED oraz przycisk .....	14
Złącza GPIO .....	15
Wymiary .....	16
Program testowy .....	17
Linki .....	22

## Opis

### Płytko ewaluacyjna z mikrokontrolerem STM32F411CEU, kompatybilna z BlackPill

Płytko ewaluacyjna **KAmo BlackPill 411** zawiera mikrokontroler STM32F411CEU oraz elementy niezbędne do jego uruchomienia i programowania. Płytko jest kompatybilna pod względem wyprowadzeń z projektem BlackPill, jednak jest to zupełnie nowa konstrukcja, wyposażona w złącze USB-C z zabezpieczeniem ESD i zmodyfikowany obwód zasilania. Może być programowana z Arduino IDE, ponieważ w pamięci układu znajduje się odpowiedni bootloader.

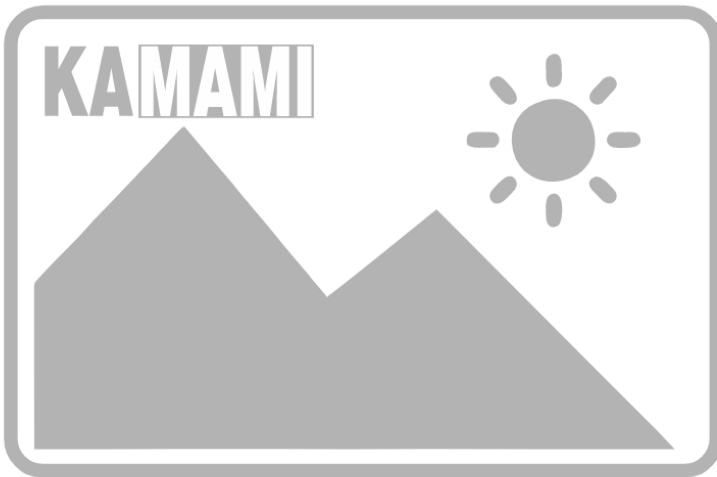


## Podstawowe cechy i parametry

- mikrokontroler STM32F411CEU6: Cortex-M4, 512 kB Flash, 128 kB RAM, 100 MHz, 1 x ADC 12-bitowy, 7 timerów, 3 x I2C, 2 x SPI, 3 x UART, USB OTG FS, RTC
- złącze USB-C służy jako złącze zasilające, interfejs komunikacyjny USB oraz pozwala na programowanie mikrokontrolera
- zawiera elementy filtrujące zakłócenia i przepięcia na liniach interfejsu USB
- 32 wyprowadzenia GPIO oraz linie zasilania 5 V i 3,3 V dostępne na standardowych złączach o rastrze 2,54 mm
- maksymalne obciążenie linii 5 V wynosi 500 mA, natomiast dla linii 3,3 V wynosi 200 mA
- precyzyjny rezonator taktujący mikrokontroler oraz niezależny rezonator dla modułu RTC
- możliwość dołączenia baterii podtrzymującej działanie modułu RTC
- złącze interfejsu programowania/debugowania SWD
- możliwość programowania poprzez STM32CubeProgrammer oraz Arduino IDE
- wymiary płytki: 53,5x23 mm, wysokość ok. 7 mm (bez wlutowanych szpilek goldpin)

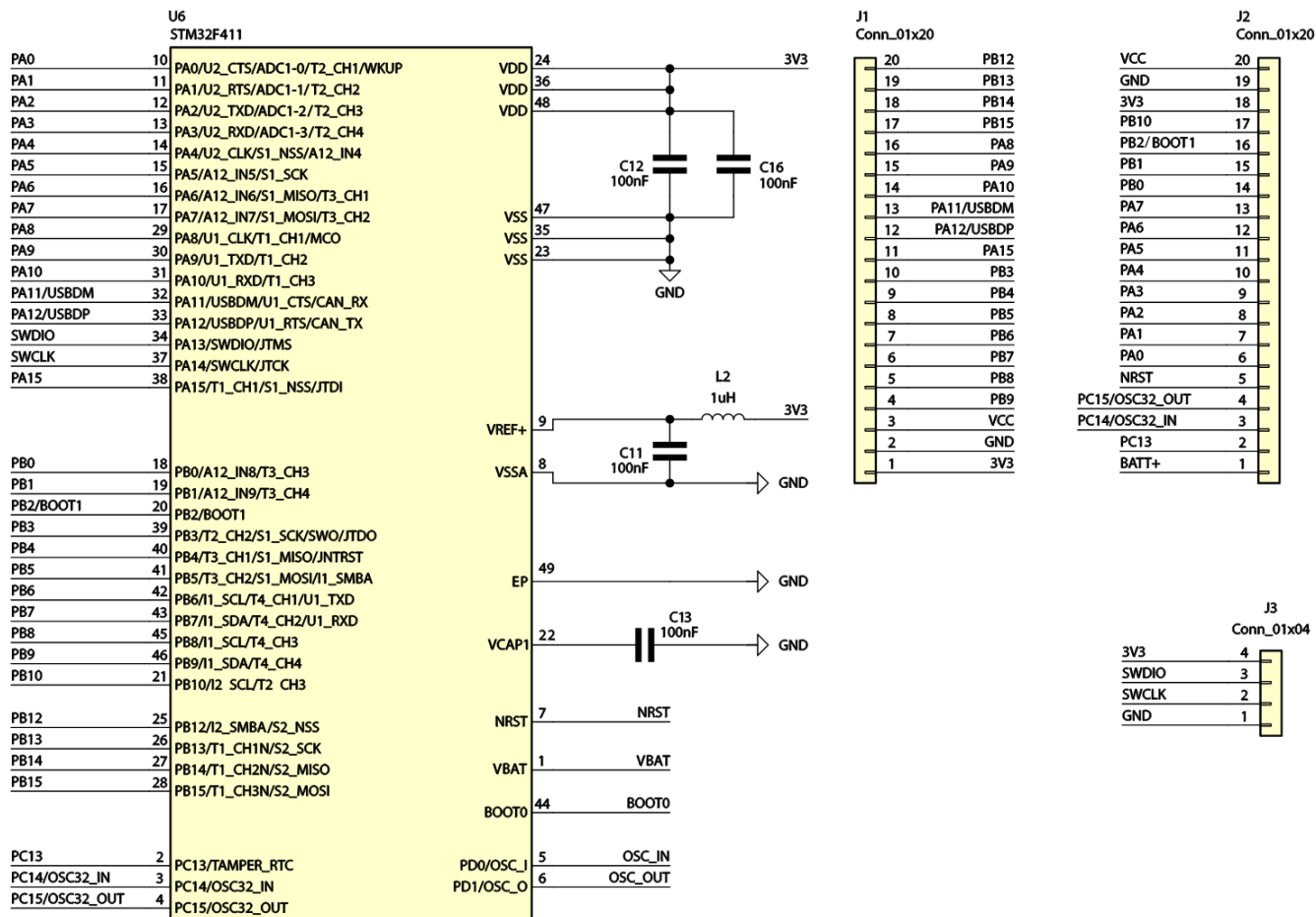
## Wyposażenie standardowe

Kod	Opis
<b>KAmo</b> <b>BlackPill 411</b>	<ul style="list-style-type: none"><li>• Zmontowany i uruchomiony moduł, z wgranym bootloaderem</li><li>• 2 x prosta listwa goldpin 20-pin raster 2,54 mm</li><li>• 1 x kątowna listwa goldpin 4-pin raster 2,54 mm</li></ul>

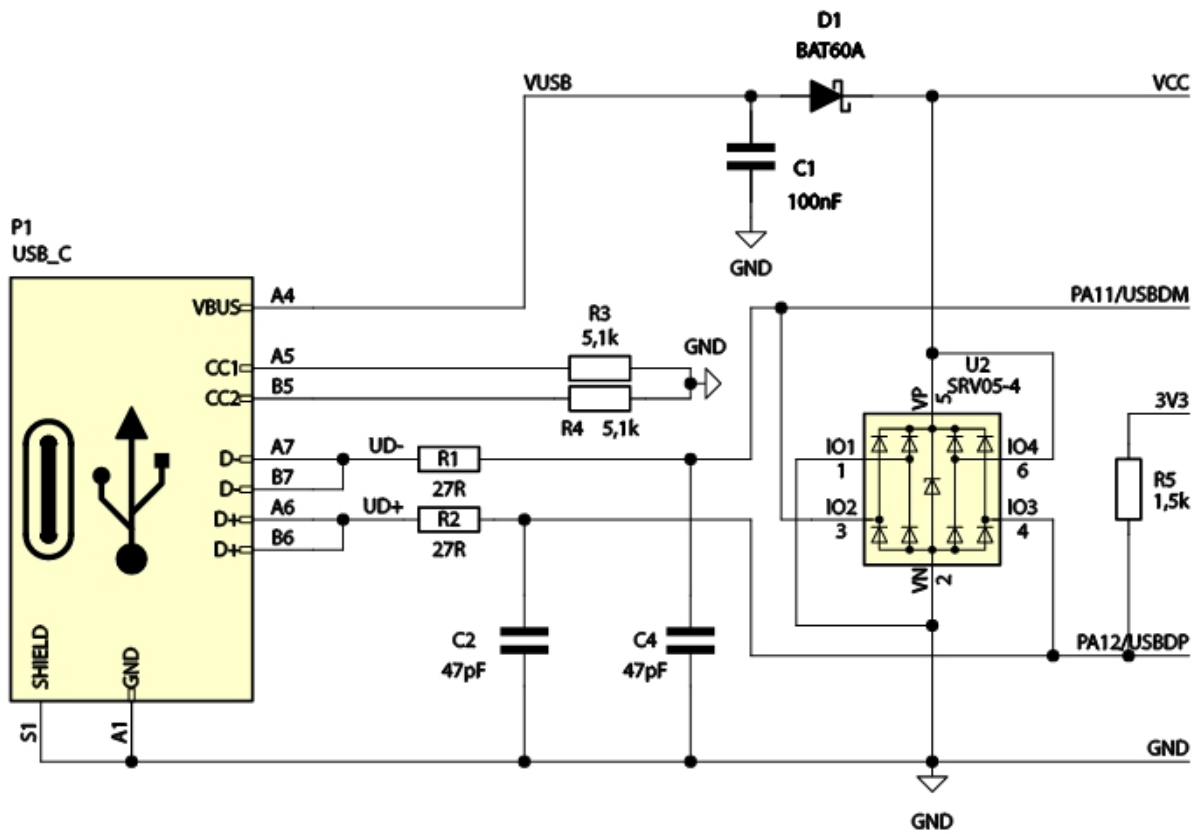


## Schemat elektryczny

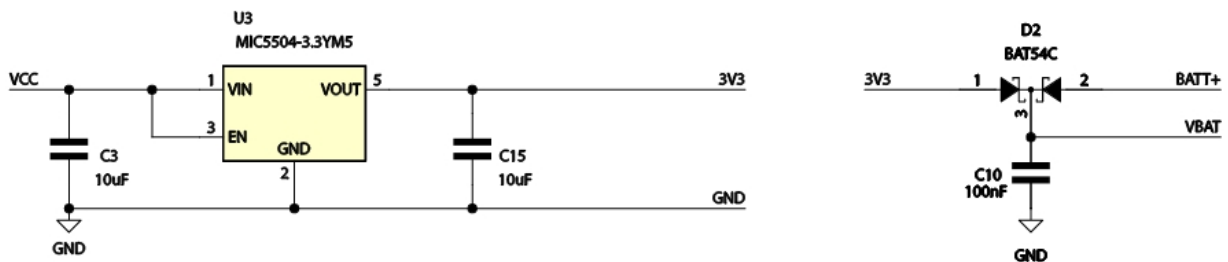
Mikrokontroler oraz złącza GPIO i SWD



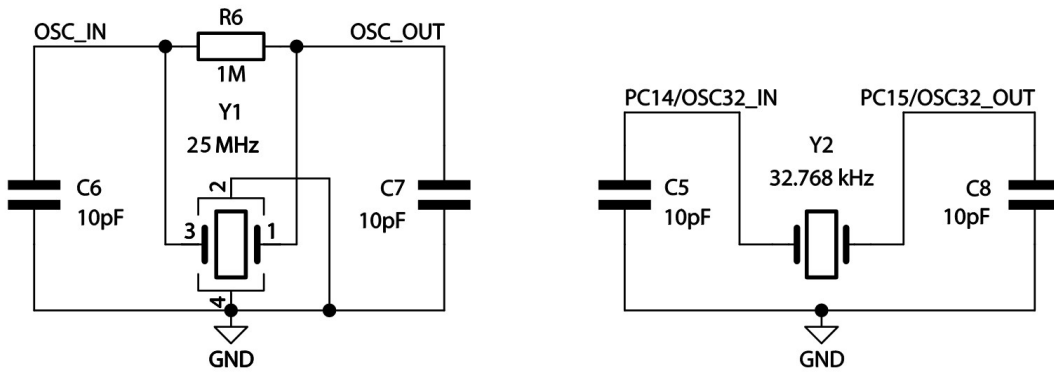
Interfejs USB



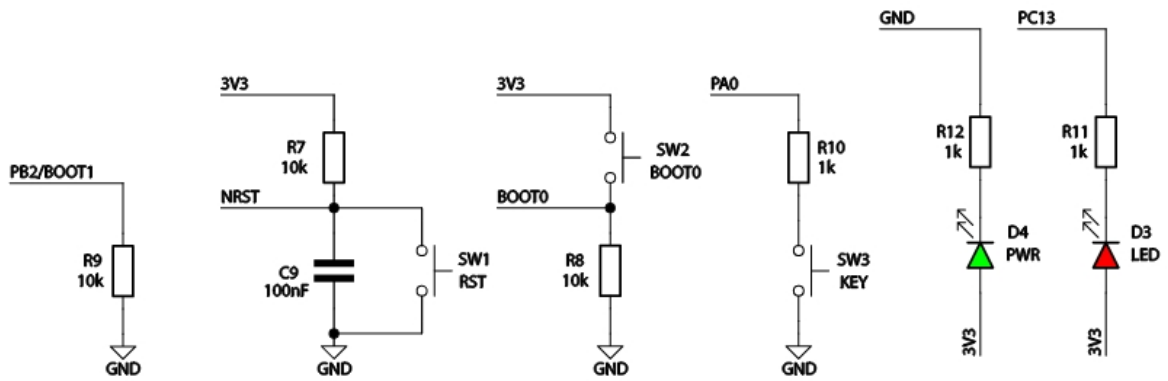
Obwód zasilania



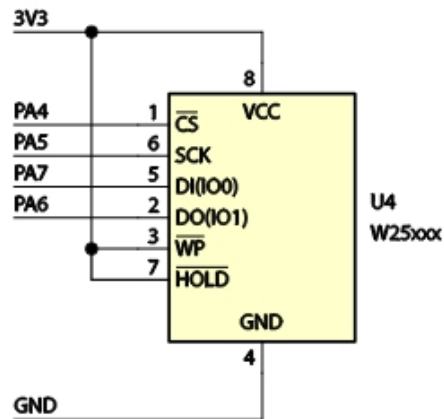
Taktowanie mikrokontrolera



Elementy dodatkowe



Zewnętrzna pamięć







## Zasilanie

Złącze	Funkcja
<b>USB-C</b> J1, J2	<ul style="list-style-type: none"> <li>Doprowadza zasilanie 5 V do modułu</li> <li>Pozwala doprowadzić zasilanie 5 V oraz udostępnić napięcie 3,3 V</li> </ul>

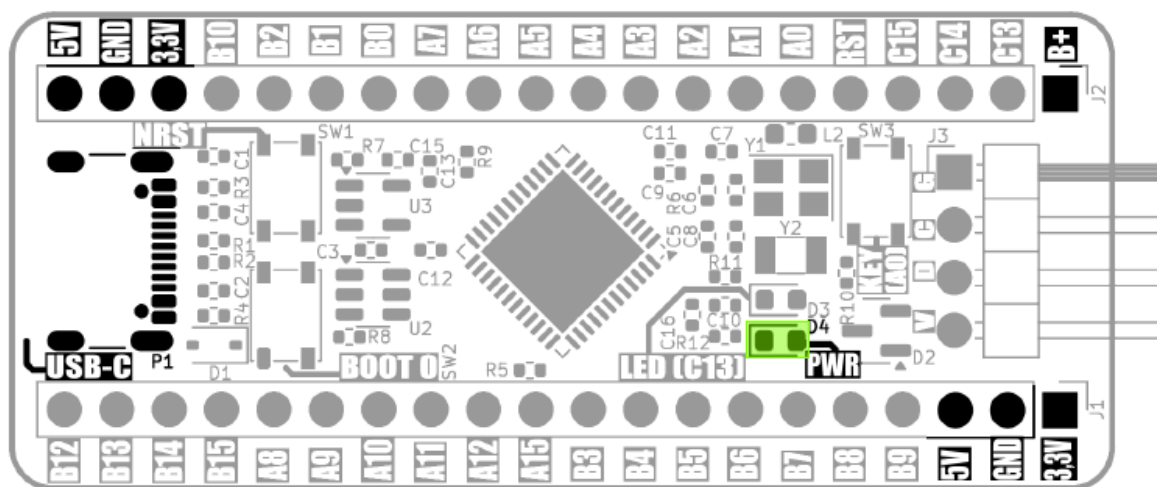
Płytki ewaluacyjnej **KAmoD BlackPill 411** może być zasilana na dwa sposoby:

- poprzez odpowiednie styki złączy J1 i J2,
- poprzez złącze USB-C.

Źródło zasilania o napięciu z zakresu 4,5...5,5 V i wydajności min. 100 mA należy dołączyć do styków oznaczonych **5V** (plus) oraz **GND** (minus) na złączach J1 i/lub J2. Wtedy na styku oznaczonym **3,3V** dostępne jest stabilizowane napięcie o wartości 3,3 V, które również zasila mikrokontroler. Obecność napięcia 3,3 V jest sygnalizowana świeceniem diody LED oznaczonej **PWR**.

Do złącza USB-C należy dołączyć standardowe źródło zasilania USB o wydajności min. 100 mA. Wtedy na styku 5V złącza J1 dostępne jest napięcie o wartości bliskiej 5 V (względem masy oznaczonej **GND**). Niewielki spadek napięcia (ok 0,5 V) występuje na diodzie Schottky'ego, która umożliwi przepływ prądu w kierunku ze złącza USB-C do płytki, ale blokuje przepływ prądu w kierunku przeciwnym - do złącza USB-C. Dzięki temu można bezpiecznie dołączać zasilanie w różnych konfiguracjach - USB i/lub styki J1, J2.

Na złączu J2 znajduje się styk oznaczony **B+**. Wraz z masą **GND** jest to wejście zasilania z baterii podtrzymującej działanie zegara RTC (zintegrowanego z mikrokontrolerem). Napięcie baterii podtrzymującej powinno zawierać się w przedziale 1,65...3,6 V.. Dokładne informacje na temat działania modułu RTC można znaleźć w dokumentacji mikrokontrolera STM32F411CEU.



## Interfejs USB

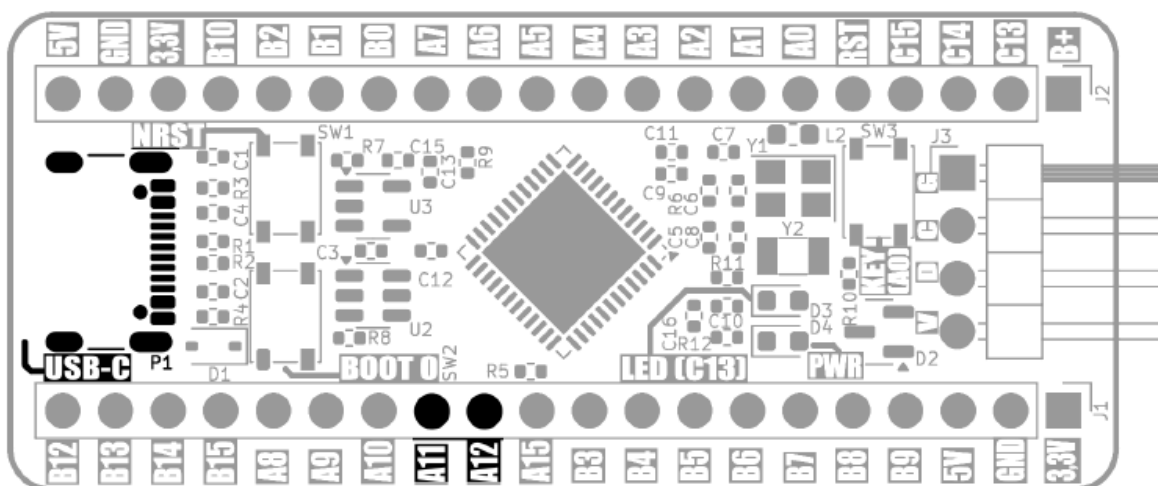
Złącze	Funkcja
<b>USB-C</b>	<ul style="list-style-type: none"> <li>Doprowadza zasilanie 5 V do modułu</li> <li>Może realizować interfejs USB 2.0 FS Peripheral</li> <li>Pozwala programować pamięć Flash mikrokontrolera</li> </ul>

Złącze USB-C jest łatwym sposobem na doprowadzenie zasilania do płytki **KAmoD BlackPill 411**. Ponadto mikrokontroler STM32F411CEU ma zintegrowany kontroler interfejsu USB 2.0 FS, który może służyć do komunikacji np. w trybie VCP (*Virtual COM Port*) lub CDC (*Communication Device Class*).

Płytkę została skonfigurowana w taki sposób, że umożliwia działanie interfejsu w trybie Peripheral, nie jest przygotowana do pracy w trybie Host (OTG). Na płytce znajdują się również elementy filtrujące ewentualne zakłócenia i przepięcia na liniach interfejsu USB, które zapewniają jego stabilne działanie.

Dodatkową funkcjonalnością złącza USB-C jest możliwość programowania pamięci Flash mikrokontrolera. Odpowiedni bootloader jest ładowany do pamięci nieulotnej (ROM) mikrokontrolera na etapie produkcji.

Linie interfejsu USB (DP oraz DM) są jednocześnie portami GPIO PA12 oraz PA11 mikrokontrolera. Jeżeli korzystamy z USB to styki oznaczone A12 i A11 nie mogą pełnić żadnej innej funkcji - muszą pozostać niepodłączone.



## Interfejs programowania/debugowania SWD

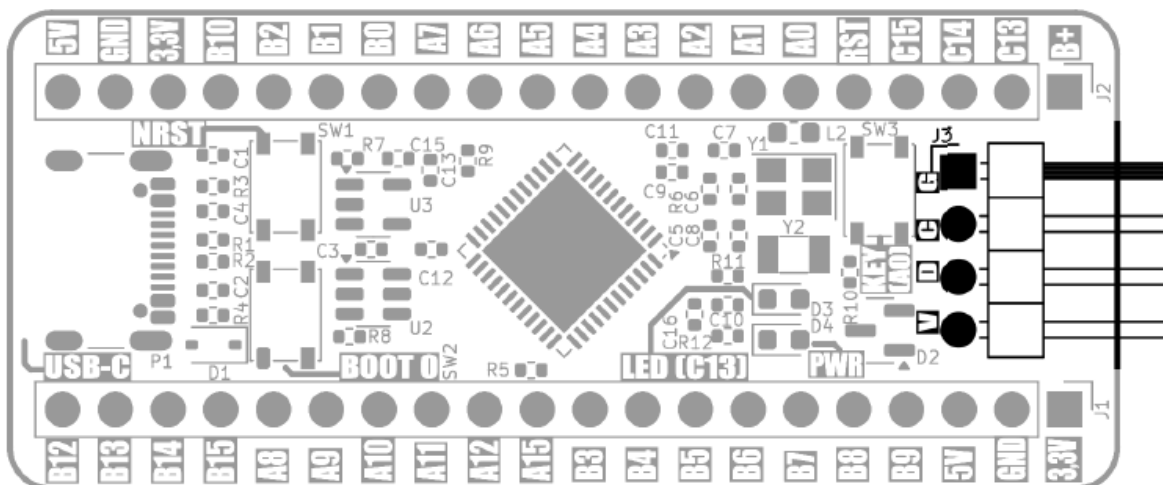
Złącze	Funkcja
J3	• Wyprowadzony interfejs SWD ( <i>Single Wire Debug</i> ) z sygnałami SWDIO oraz SWCLK

Interfejs SWD (*Single Wire Debug*) pozwala na programowanie pamięci Flash mikrokontrolera oraz śledzenie działania programu (debugowanie). Wymaga dołączenia zewnętrznego programatora/debuggera np. STLINK-V2 lub STLINK-V3MINIE.

Interfejs SWD został wyprowadzony na złącze szpilkowe J3. Sygnały zostały opisane w następujący sposób:

- G – masa układu,
- C – sygnał taktujący SWCLK,
- D – sygnał danych SWDIO,
- V – linia zasilania 3,3 V.

Sygnały należy połączyć z takimi samymi sygnałami na złączu programatora/debuggera. Czasami SWCLK jest oznaczony również jako TCK, natomiast SWDIO jest jednocześnie oznaczony jako TMS. Programator nie dostarcza zasilania do płytki **KAmoD BlackPill 411**, zasilanie należy dołączyć do złącza USB-C lub styków J1/J2.



## Programowanie mikrokontrolera

Komponent	Funkcja
<b>SW1 - NRST</b>	• Wymuszenie stanu wyzerowania mikrokontrolera
<b>SW2 - BOOT0</b>	• Uruchomienie fabrycznego bootloadera mikrokontrolera
<b>J3 - SWD</b>	• Interfejs programowania SWD (Single Wire Debug)
<b>A9, A10 - UART</b>	• Interfejs programowania UART (TXD, RXD)

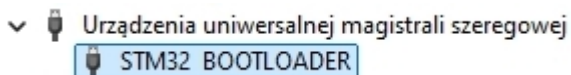
W specjalnym obszarze pamięci mikrokontrolera znajduje się oprogramowanie umożliwiające zaprogramowanie jego pamięci programu - jest to tzw. bootloader przygotowany przez producenta mikrokontrolera. Aby uruchomić bootloader należy wykonać określoną sekwencję czynności, przy podłączonym zasilaniu płytki:

1. Przycisnąć i trzymać przycisk NRST
2. Przycisnąć i trzymać przycisk BOOT 0 trzymając wciśnięty przycisk NRST
3. Zwolnić przycisk NRST, trzymając wciśnięty przycisk BOOT 0
4. Zwolnić przycisk BOOT 0

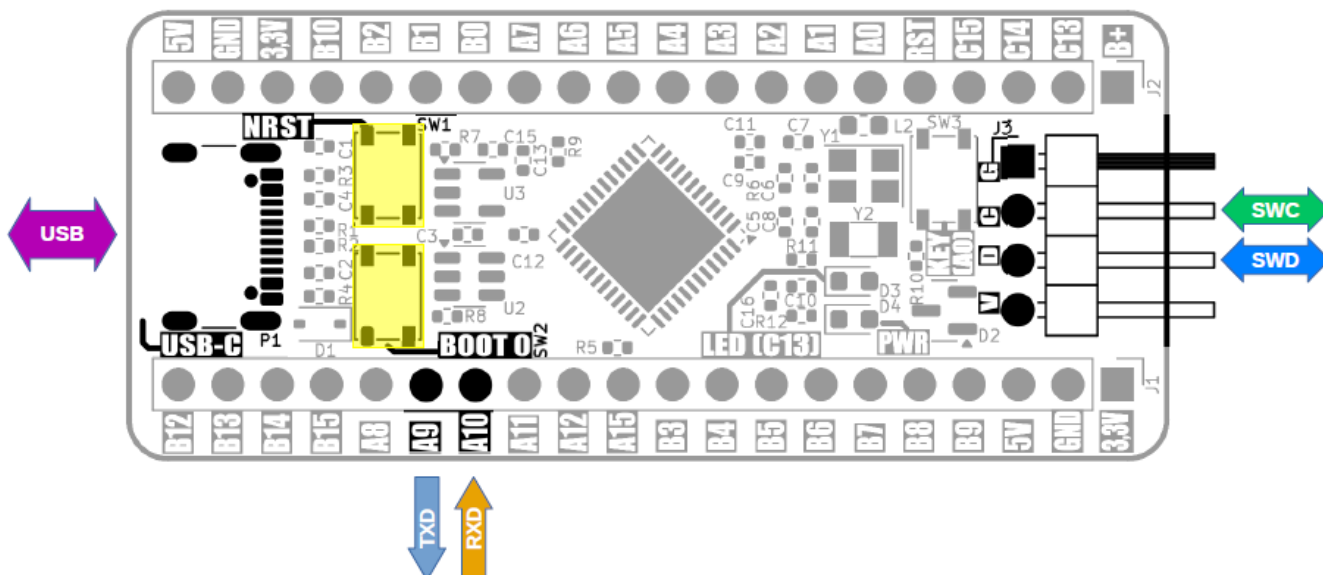
Inną metodą uruchomienia bootloadera jest podłączenie zasilania płytki przy wciśniętym przycisku BOOT 0.

Uruchomienie bootloadera aktywuje dwa interfejsy programujące:

- **SWD**, który został opisany we wcześniejszym rozdziale i pozwala na programowanie mikrokontrolera z użyciem programatora/debuggera np. STLINK-V2 lub STLINK-V3MINIE.
- **UART**, który pozwala na użycie zwykłego konwertera USB-UART w roli programatora, np. [KAmod USB-UART-mini](#). Należy wtedy podłączyć do styków A9 oraz A10 linie interfejsu UART, gdzie:
  - A9 - wyjście TXD mikrokontrolera należy podłączyć do wejścia RXD konwertera USB-UART,
  - A10 - wejście RXD mikrokontrolera należy podłączyć do wyjścia TXD konwertera USB-UART.
- **USB**, który pozwala na bezpośrednie programowanie mikrokontrolera podłączonego do komputera PC. Po prawidłowym zainicjowaniu interfejsu mikrokontroler zostanie rozpoznany jako urządzenie STM32 BOOTLOADER:



Ze względu na pewne uwarunkowania techniczne prawidłowe działanie bootloadera z interfejsem USB w układzie STM32F411CEU jest możliwe tylko wtedy, gdy układ ma temperaturę zbliżoną do 25°C. Tylko w takich warunkach możliwe jest wystarczająco precyzyjne działanie zintegrowanego generatora taktującego. Mikrokontroler można programować poprzez jeden z interfejsów: SWD, UART lub USB za pomocą aplikacji STM32CubeProgrammer, który jest dostępny na oficjalnej stronie STMicroelectronics: <https://www.st.com/en/development-tools/stm32cubeprog.html>



Płytki **KAmoD BlackPill 411** umożliwia programowanie mikrokontrolera poprzez interfejs USB z użyciem Arduino IDE. Należy wtedy:

- dodać następujący adres do źródeł menedżera płytek (*Preferences*) w Arduino IDE:  
[https://github.com/stm32duino/BoardManagerFiles/raw/main/package\\_stmicroelectronics\\_index.json](https://github.com/stm32duino/BoardManagerFiles/raw/main/package_stmicroelectronics_index.json)
- poprzez menedżer płytek (*Boards Manager*) zainstalować pakiet „*STM32 MCU based boards*”
- zainstalować STM32CubeProgrammer (<https://www.st.com/en/development-tools/stm32cubeprog.html>)

Następnie, po utworzeniu nowego szkicu, w zakładce Tools:

- należy wybrać płytkę: *BlackPill F411CC*,
- należy ustawić *Upload method* na: *STM32CubeProgrammer(DFU)*,
- *USB support* należy ustawić na: *CDC (generic Serial)*,
- *U(S)ART support* należy ustawić na: *Enabled (generic Serial)*

Blackpill\_411 | 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 STM32F4 series"

Port: "COM9"

Get Board Info

Debug symbols and core logs: "None"

Optimize: "Smallest (-Os default)"

Board part number: "BlackPill F411CE"

C Runtime Library: "Newlib Nano (default)"

Upload method: "STM32CubeProgrammer (DFU)"

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

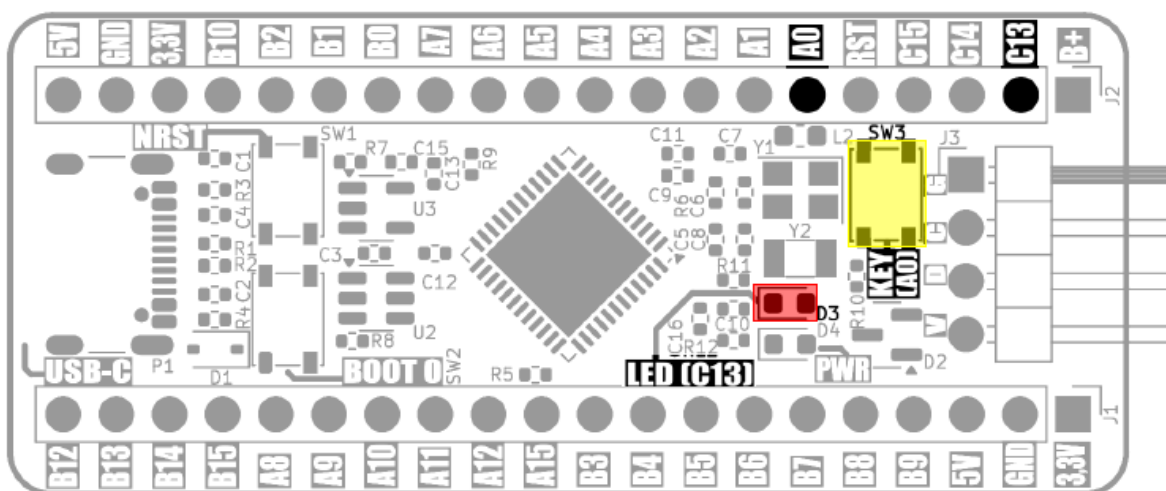
## Dodatkowe elementy - dioda LED oraz przycisk

Komponent	Funkcja
<b>D3 - LED (C13)</b>	• Dioda LED dołączona do portu PC13, aktywna w stanie L
<b>SW3 - KEY (A0)</b>	• Mikroprzycisk dołączony do portu PA0, aktywny w stanie L

Płytki **KAmoD BlackPill 411** zawiera elementy dodatkowe - diodę LED i mikroprzycisk, które mogą być użyte w docelowej aplikacji.

Dioda LED jest podłączona do portu PC13, jej zaświecenie następuje przy stanie niskim.

Mikroprzycisk jest dołączony do portu PA0, jego wciśnięcie wymusza stan niski na tym porcie. Przycisk jest dołączony poprzez rezystor 1k, zatem nie ma możliwości uszkodzenia portu PA0 skonfigurowanego jako wyjście.

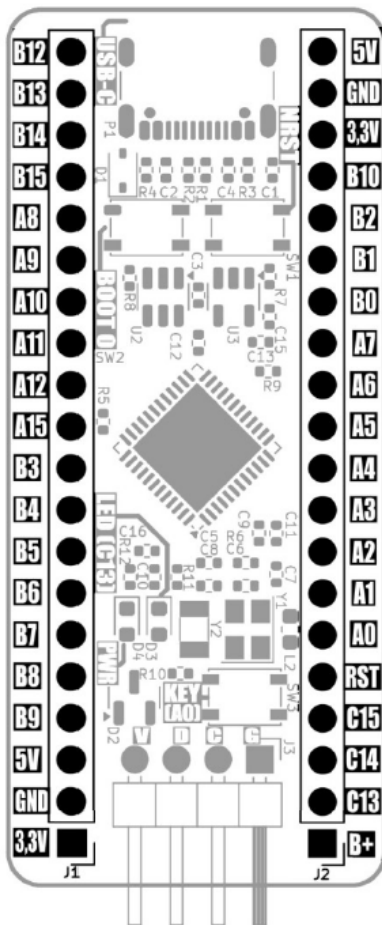




## Złącza GPIO

Złącze	Funkcja
J1, J2	• Złącza o 20 stykach z rastrem 2,54 mm, do których doprowadzone są porty GPIO mikrokontrolera oraz zasilanie 5 V i 3,3 V

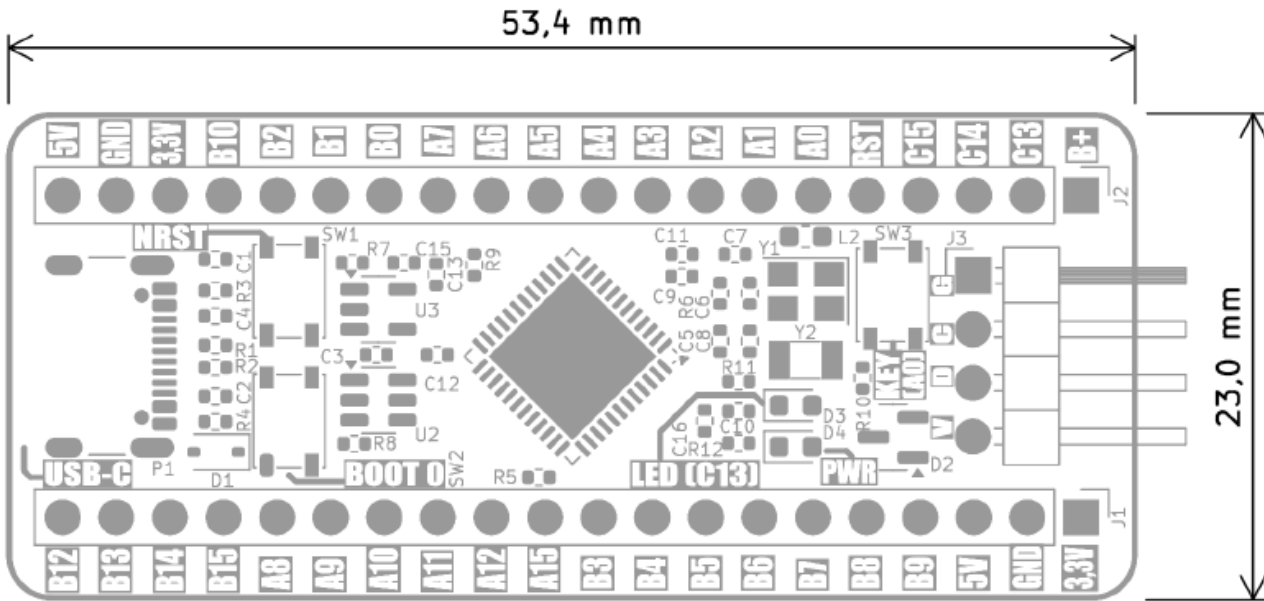
Złącza GPIO J1 i J2 zawierają po 20 szpilek, do których doprowadzone są linie zasilania 5 V, 3,3 V, masa GND oraz wyprowadzenia GPIO modułu mikrokontrolera. Porty GPIO tolerują napięcie 5 V (5 V tolerant). Dokładny opis wyprowadzeń oraz niektóre ich dodatkowe funkcje pokazuje rysunek i tabela poniżej.



J1	J2
PB12 (SPI2_NSS)	5 V
PB13 (SPI2_SCK)	GND
PB14 (SPI2_MISO)	3,3 V
PB15 (SPI2_MOSI)	PB10
PA8 (TIMER1_CH1)	(BOOT1) PB2
PA9 (UART1_TXD)	(ADC1_IN9) PB1
PA10 (UART1_RXD)	(ADC1_IN8) PB0
PA11 (USB_DM)	(ADC1_IN7) PA7
PA12 (USB_DP)	(ADC1_IN6) PA6
PA15 (SPI1_NSS)	(ADC1_IN5) PA5
PB3 (SPI1_SCK)	(ADC1_IN4) PA4
PB4 (SPI1_MISO)	(TIMER2_CH4) PA3
PB5 (SPI1_MOSI)	(TIMER2_CH3) PA2
PB6 (I2C1_SCL)	(TIMER2_CH2) PA1
PB7 (I2C1_SDA)	(TIMER2_CH1) PA0
PB8 (TIM4_CH3)	RESET
PB9 (TIM4_CH4)	(OSC_RTC_O) PC15
5 V	(OSC_RTC_I) PC14
GND	(LED_D3) PC13
3,3 V	RTC_BATTERY +

## Wymiary

Wymiary płytki **KAmoD BlackPill 411** to 53,5x23 mm, a wysokość wynosi ok. 7 mm (bez wlutowanych szpilek goldpin).



## Program testowy

Uproszczony kod programu testowego znajduje się poniżej, można go skompilować w środowisku Arduino.

```
#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 BlackPill411 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(PB2, 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(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(PB2, 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(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(PB2, 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(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(PB2, 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(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 = PB2; break;
case 11: pp = PB10; 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);
}
}

```

Działanie programu polega na uruchomieniu portu szeregowego na bazie interfejsu USB oraz modułu zegara RTC i cyklicznym wysyłaniu informacji o czasie poprzez ten port szeregowy.

Działaniu programu towarzyszy powolne miganie diody LED D3. Gdy zostanie naciśnięty przycisk KEY (SW3) to na wszystkich portach kolejno będzie ustawiany stan wysoki oraz dioda LED zacznie pulsować szybko.

## Linki

- [Model CAD \(STEP\)](#)
- [Karta katalogowa układu STM32F411CEU](#)
- [Program testowy Arduino](#)
- [Projekt testowy STM32CUBE\\_IDE](#)
- [STLINK-V3MINIE - kompaktowy programator/debuger dla STM32](#)
- [STM32CubeProgrammer](#)
- [KAmod USB-UART-mini - Miniaturowy konwerter USB-UART](#)





Zastrzegamy prawo do wprowadzania zmian bez uprzedzenia.

Oferowane przez nas płytki drukowane mogą się różnić od prezentowanej w dokumentacji, przy czym zmianom nie ulegają jej właściwości użytkowe.

BTC Korporacja gwarantuje zgodność produktu ze specyfikacją.

BTC Korporacja nie ponosi odpowiedzialności za jakiegokolwiek szkody powstałe bezpośrednio lub pośrednio w wyniku użycia lub nieprawidłowego działania produktu.

BTC Korporacja zastrzega sobie prawo do modyfikacji niniejszej dokumentacji bez uprzedzenia.