# Graficzne Programowanie Mikrokontrolerów

*Ćwiczenie 1*

## 1 Wstęp

Programowanie graficzne polega na zapisaniu w dedykowanym oprogramowaniu algorytmu programu w sposób graficzny. Realizowane jest to zazwyczaj za pomocą diagramu przepływów (LabView) lub schematu blokowego (M5Stack UIFlow). To drugie rozwiązanie wzorowane jest na schematach Nassi–Shneiderman, czyli schematach kaskadowych. Ten sposób programowania to BLOKLY. Jest on stosowany jako nakładka graficzna przy generowaniu kodu w językach interpretowanych (php, javascript, LUA, Python, Dart).

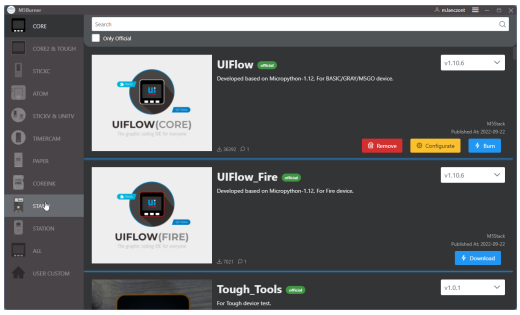
Rozwiązanie to przyjęte zostało przy projektowaniu interfejsu środowiska programistycznego dla mikrokontrolerów firmy M5Stack. Edytor umożliwia pisanie programów dla mikrokontrolerów M5Stack w języku Micro Python bezpośrednio kodem, lub za pomocą dostarczonych narzędzi graficznych.

## 2 Konfiguracja

Micro Python jest językiem interpretowanym. Kod programu jest interpretowany przez interpreter i wykonywany na mikrokontrolerze. W przypadku produktów M5Stack uruchomienie kodu programu jest możliwe na dwa sposoby:

- wykonanie kodu przez środowisko programistyczne na mikrokontrolerze, rozwiązanie stosowany na etapie projektowania aplikacji
- zapisanie kodu programu w pamięci mikrokontrolera i wykonanie interpretacji kodu przez interpreter zintegrowany w firmware'rze mikrokontrolera, do zastosowania dla gotowego produktu.

Rozwiązanie to wymaga systematycznej konserwacji firmware'u mikrokontrolera w celu zapewnienia dostępności zawsze aktualne wersji interpretera. M5Stack dostarcza dedykowanego dla swoich produktów programu konfiguracyjno-aktualizacyjnego - M5Burner. Oprogramowanie jest dostępne na stronie producenta pod adresem: https://docs.m5stack.com/en/download.




Rys. 1: M5Burner 3.0-beta: interfejs programu

Program pozwala na konfiguracje parametrów startowych i sieciowych mikrokontrolera. Pozwala na ustawienie:

1. trybu pracy (aplikacji, połączenia z środowiskiem programistycznym [usb lub internet])
2. wybór serwera m5stack do połączenia internetowego
3. konfiguracja połączenia wifi

Aktualnie dostępna jest wersja 3.0-beta z ograniczoną w stosunku do wcześniejszej wersji funkcjonalnością.

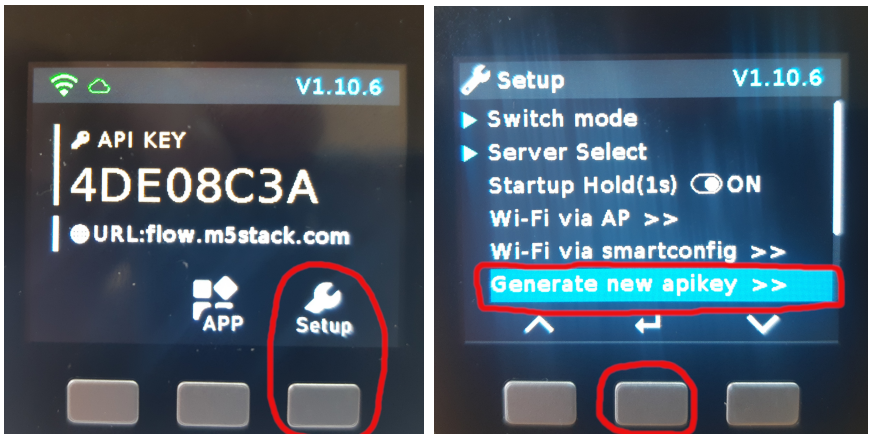
# 3 Firmware

Oprogramowanie zainstalowane (za pomocą M5Burner) na mikrokontrolerze zarządza działaniem urządzenia. Składa się z dwóch elementów. Aplikacji sterującej oraz interpretera Micro Python'a.

Program sterujący odpowiada za sposób działania mikrokontrolera, konfigurowanie jego wybranych parametrów: Urządzenie może po uruchomieniu wejść w jedne z trzech dostępnych trybów pracy:

1. tryb aplikacji - uruchamiany jest kod ostatnio wgranego programu
2. tryb USB - uruchamiany jest tryb łączności szeregowej.
   (a) konfigurowanie, wgrywanie firmware'u i aplikacji (M5Burner, Arduino IDE)
   (b) wykonywanie przez interpreter Micro Python'a programu na mikrokontrolerze
   (c) Tworzenie i testowanie kodu w UIFlow-Coding IDE
3. tryb internetowy - zdalny dostęp do mikrokontrolera poprzez wybrany serwer pośredniczący, uruchamianie i wgrywanie programów tworzonych w UIFLow Web IDE

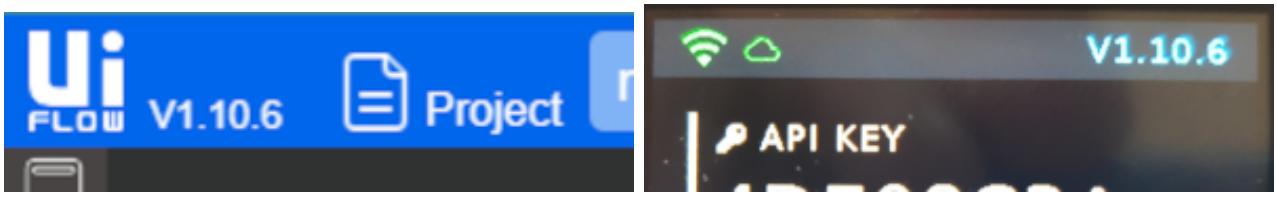
W trybie USB i internet możliwe jest przejście do panelu konfiguracyjnego. W trybie tym na początku każdych zajęć **konieczne będzie aktywowanie nowego klucza API** dzięki któremu realizowany będzie dostęp zdalny do mikrokontrolera.




Rys. 2: Generowanie nowego klucza API

# 4 UIFlow Web IDE

Wydaje się że M5Stack zaprzestał rozwijania stacjonarnej wersji środowiska programistycznego dla swoich mikrokontrolerów (UIFlow-Coding IDE). W ramach zajęć projektowych z "Graficznego programowania mikrokontrolerów" podstawowym środowiskiem programowania będzie UIFlow Web IDE. Po uruchomieniu środowiska należy upewnić się że na mikrokontrolerze zainstalowana jest aktualna wersja firmware'u. Przy braku zgodności zalecana jest aktualizacja oprogramowanie na mikrokontrolerze do aktualnej wersji.




Rys. 3: Określanie wersji firmware'u w aplikacji i na mikrokontrolerze

Środowisko dostarcza bloki wykonawcze realizujace określone zadania.

- Podstawowy zestaw bloków realizuje zadania związane z elementomi składni języka Micro Python, związanymi z tworzeniem i przetwarzaniem danych, operacjami wejścia-wyjścia oraz GUI.
- Bloki związane z warstwą sprzętową zaimplementowaną w wybranym modelu mikrokontrolera.
- Grupa bloków realizująca zaawansowane zadania.
- Realizacja usług związaną z chmurą IoT.
- bloki realizujące właściwości dostarczane przez elementy zewnętrzne (moduły, rozszerzenia itp), także dostarczane przez firmy zewnętrzne.
- narzędzia tworzenia i dodawania własnych bloków w dedykowanym środowisku w oparciu o kod pisany w języku Micro Python

Wraz z rozwojem środowiska i oferty sprzętowej dodawane są nowe funkcjonalności dla istniejących bloków, nowe bloki lub całe grupy bloków realizujące nowe funkcjonalności. Z środowiskiem programowania zintegrowany jest system pomocy, omawiający większość dostępnych bloków wraz z licznymi przykładami obrazującymi podstawowe funkcjonalności.

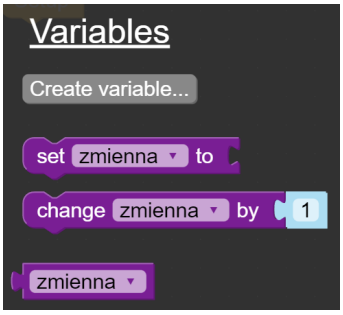
## 4.1 Podstawowe bloki wykonawcze

Składnia algorytmu implementowanego w BLOKLY jest w znacznym stopniu przeniesieniem kodu pisanego w Micro Python'ie do postaci graficznej. Jest to jednak związane z koniecznością przyjęcia pewnych ograniczeń. Nie każdy kod napisany w Micro Python'ie można przenieść do postaci kodu BLOKLY.

### 4.1.1 Zmienne

W UIFlow można tworzyć tylko zmienne globalne. Typ zmiennej dobierany jest automatycznie na podstawie aktualnie przechowywanej w niej wartości. Powoduje to że tworząc kod aplikacji trzeba zwracać szczególną uwagę na właściwe zarządzanie zmiennymi. W środowisku można przetwarzać za pomocą zmiennych wartości podstawowych typów:

- logicznego,
- całkowitego,
- rzeczywistego,
- logicznego,
- daty i czasu
- typów złożonych listy i mapy.




Rys. 4: Bloki przetwarzania zmiennej

Po dodaniu do kodu aplikacji nazwy zmiennej aktywowany jest dostęp do bloków związanych z jej przetwarzaniem. Bloki przetwarzania określonych typów zmiennych dostępne są w odpowiadających im zakładkach, matematycznej, logicznej, tekstu, listy, mapy i o ile dostępny jest moduł RTC bloki daty i czasu. Złożone dane - lista (tablica) i mapa (kontener) pozwalają na przechowywanie zestawów danych indeksowanych liczbowo lub za pomocą unikalnych nazw.

### 4.1.2 Bloki sterujące

Sterowanie wykonaniem kodu realizowane może być przez dwie grupy bloków, instrukcji warunkowych i operacji logicznych oraz pętli. Grupa bloków logicznych poza typem danych `BOOLEAN` udostępnia bloki BLOKLY realizujące:

- instrukcje warunkowe proste i złożone - `if else, when`

- instrukcje wyboru - `case, test`
- testowania - `try`

Test logiczny na którym opierają się instrukcje warunkowe można budować w oparciu o:

- wynik działania funkcji logicznej - zwracającej wartość typu `BOOLEAN`
- warunku prostego (=,<,>,≤, ≥, ≠)
- warunków złożonych, w oparciu o operatory logiczne (`and, or, not`)

Drugą grupę stanowią instrukcje pętli (LOOPS). Dostarcza ona bloków odpowiadających podstawowym konstrukcjom:

- pętla `for` wykonująca kod w niej ujęty po elementach tablicy stanowiącej jej argument.
- pętla `count` - jest odpowiednikiem klasycznej pętli for znanej z C/C++
- pętla `repeat` - występująca w trzech formach:
  - powtórzenie wykonywane jest określoną liczbę razy
  - kod powtarzany jest tak długa jak spełniony jest warunek - typ `while`
  - kod powtarzany jest dopóki nie zostanie spełniony warunek - typ `until`

Dodatkowo dostępny jest blok realizujący znane z języków C/C++ instrukcje przerwania pętli `break` i `continue`.

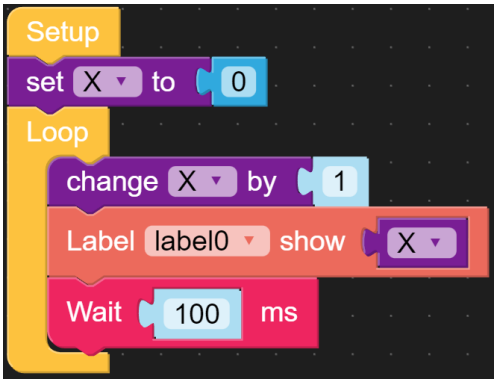
### 4.1.3 Zdarzenia

Zdarzenia są blokami kodu wykonywanymi w określonych chwilach czasowych lub w odpowiedzi na działanie użytkownika bądź otoczenia. Są zgrupowanie w grupie **EVENT**. Podstawowym blokiem z tej grupy występującym w każdym programie jest blok startowy **SETUP**. Bloki po nim wstawione wykonywane są zawsze w momencie uruchomienia programu. Drugim blokiem z tej grupy jest blok pętli głównej, nieskończonej **LOOP**.

W grupie **EVENT** dostępne są jeszcze dwie rodziny bloków:

a) bloki zegara systemowego - wykonujące wskazany kod w sterowalnych odstępach czasu

b) bloki zdarzeń z przyciskami (fizycznymi) - wykonujący wskazany kod w odpowiedzi na wybraną konfigurację naciśnięć przycisków




Rys. 5: Główne bloki typu EVENT programu - SETUP i LOOP

Pierwsza grupa składa się z czterech przycisków pozwalających zdefiniowanie bloku kodu zegara oraz uruchomienia i terowania pracą zegara. Dopiero po umieszczeniu na pulpicie bloków z kodem zegara możliwe będzie umieszczenie bloku startowego i bloków sterujących jego pracą. Blok zegara może być uruchomiony tylko raz `ONE_SHOT` lub cyklicznie `PERIODIC`. Bloki sterujące pozwalają na zmianę trybu pracy, czasu pomiędzy wywołaniami lub zatrzymanie zdarzenia.

Druga grupa dostarcza narzędzi umożliwiających reakcję programu na zdarzenia związane z fizycznymi przyciskami umiejscowionymi na mikrokontrolerze. Zaprogramować można reakcję na cztery rodzaje zdarzeń:


Rys. 6: Zdarzenie powiązane z przyciskiem **A**

I. `wasPressed` - na naciśnięcie wybranego przycisku

II. `wasRelesed` - na zwolnienie wybranego przycisku

III. `longPress` - na przytrzymanie wybranego przycisku

IV. `wasDoublePress` - na podwójne wciśnięcie wybranego przycisku

Możliwe jest także zaprogramowanie zdarzenia wywołanego naciśnięciem dwóch wskazanych przycisków. Podobne zdarzenie można zaprogramować dla wizualnego przycisku na ekranie dotykowym mikrokontrolera M5Core2 dla dwóch pierwszych typów reakcji. Zdarzenia można zaprogramować dla wszystkich interaktywnych obiektów graficznego interfejsu użytkownika.

Na chwilę obecną nie zaimplementowano do środowiska zdarzeń związanych z zmianą stanów na portach mikrokontrolerów M5Stack.

## 4.2 Połączenie z mikrokontrolerem

Z mikrokontrolerami M5Stack w środowisku programowania graficznego UIFlow można połączyć się na dwa sposoby:

- ▷ port usb-c, mozliwe do wykonania w stacjonarnej wersji oprogramowania
- ▷ przez serwer pośredniczący za pośrednictwem klucza API dla wersji Web oprogramowania

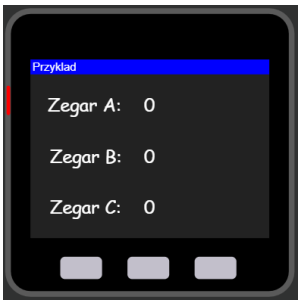

Rys. 7: Okno połączenia zdalnego z mikrokontrolerem identyfikowanym kluczem API

Firma M5Stack w rozwoju stawia na programowanie mikrokontrolerów za pomocą aplikacji Web. Podejście takie że programować można praktycznie z każdego urządzenia posiadającego przeglądarkę internetową i dostęp do sieci. Niestety podejście takie wiąże się z kilkoma problemami. Zarówno mikrokontroler jak i jednostka programująca muszą mieć dostęp do internetu. Ponadto należy mieć świadomość że do mikrokontrolera wysyłany jest program w otwartym kodzie poprzez chiński serwer pośredniczący. W przypadku zastosowań amatorskich i edukacyjnych nie stanowi to większego problemu, ale przy zastosowaniach komercyjnych staje się to poważnym problemem. Do celów komercyjnych bezpieczniejsza jest wersja stacjonarna UIFlow, aczkolwiek tutaj tez na mikrokontrolerze zostanie umieszczony kod aplikacji w języku Micro Python, a więc będzie dostępny do podglądu i analizy. Dlatego w takich przypadkach kod programu lepiej jest opracować w środowisku klasycznego programowania np.: Arduino IDE i na mikrokontroler wysłać skompilowaną, wykonywalną wersję aplikacji.

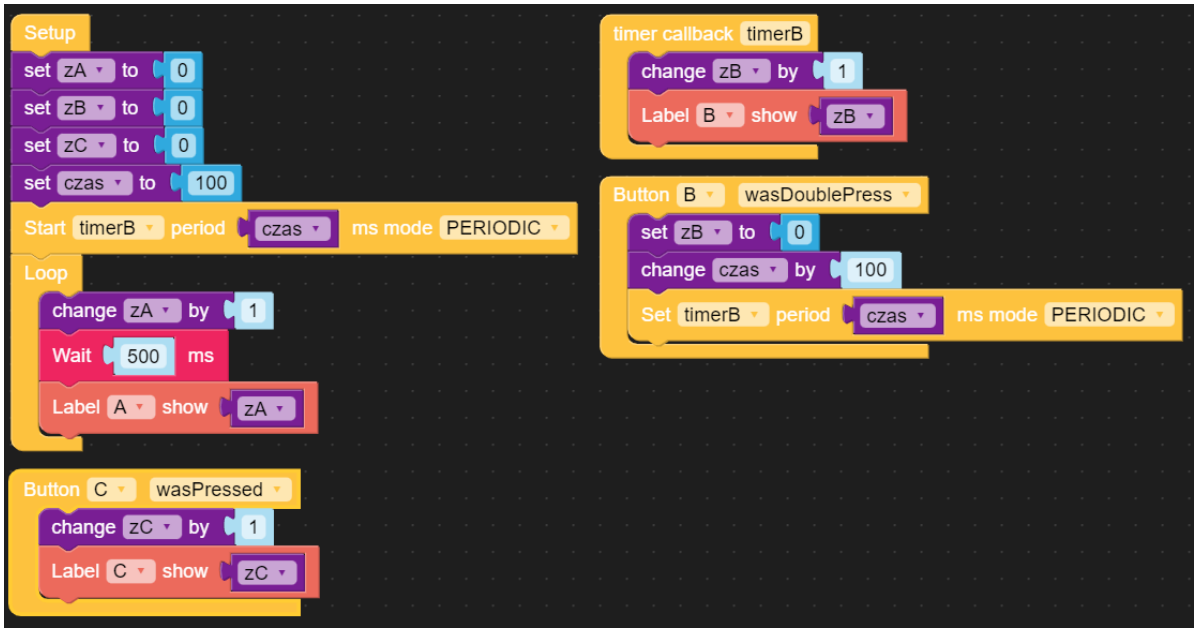
# 5 Przykład

Program demonstruje działanie zdarzeń zegara i przycisku oraz pętli głównej (nieskończonej) aplikacji. Aplikacja wyświetla na ekranie trzy liczniki:

1. pętli głównej `A`
2. zegara `B`
3. przycisku `C`




Rys. 8: Interfejs użytkownika aplikacji




Rys. 9: Kod graficzny aplikacji

# 6 Zadanie

Napisać programy realizujące poniższe zadanie. Programy napisać wykorzystując programowanie funkcyjne.

1. Napisać program wyświetlający odliczanie co sekundę od ustawionego przez użytkownika czasu.
2. Podawany czas jest określony przez liczbę godzi, minut i sekund (`HH:MM:SS`).
3. Po zakończeniu odliczania odgrywany jest sygnał dźwiękowy (melodyjka, np. gama) lub wyświetlana grafika (png, jpg, do 50kB /320x240px/).
4. Interfejs użytkownika umożliwia zatrzymanie i wznowienie odliczania oraz wyzerowania, ustawienia czasu odliczania i uruchomienia odliczania.

UWAGI:

*Odliczanie czasu (jednej sekundy) najefektywniej realizować za pomocą TIMER'a. Oparcie procesu odliczania o bloki WAIT może spowodować problemy z prawidłowym działaniem kodu.*

*Do obliczania upływu czasu najlepiej realizować na okresie czasu przeliczonym na sekundy, przeliczanym do wyświetlania na HH:MM:SS.*