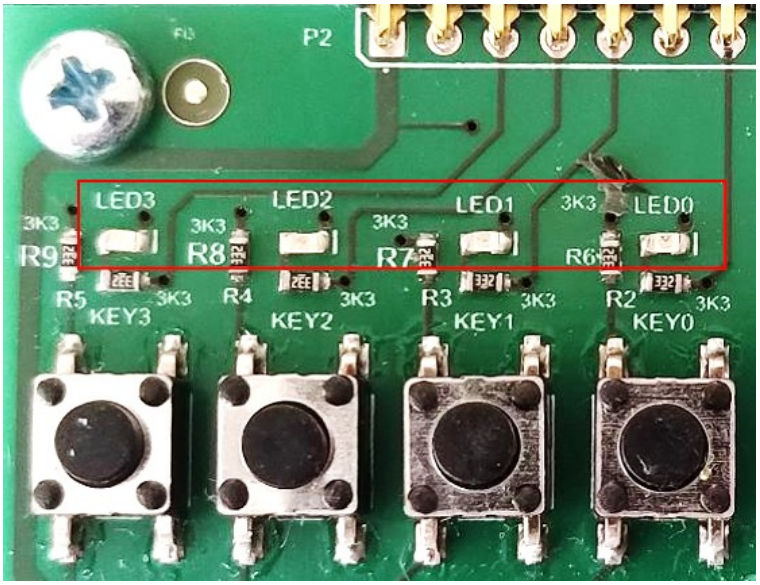
Wprowadzenie

Płytka mikrokontrolera zestawu sprzętowego została wyposażona, między innymi, w cztery diody LED.

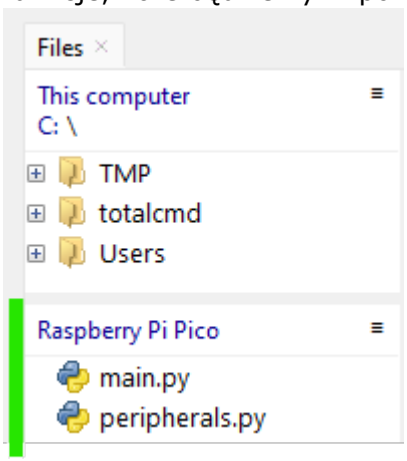


Sterowanie diodami odbywa się z użyciem funkcji do zapalania\gaszenia diody o określonym numerze. Są to funkcje `LedSet(pos)` \ `LedClr(pos)`. Numer diody podaje się w argumencie wywołania funkcji.

Opisane wyżej funkcje zostały zdefiniowane w osobnym pliku źródłowym nazywanym dalej modułem peryferiów.

Aby móc użyć wspomnianych funkcji w programie:

- w katalogu z plikiem programu musi znajdować się również plik z modułem, zawierającym funkcje, kóre będziemy importować



- przed wywołaniem funkcji muszą one zostać zaimportowana z modułu peryferiów
  `from peripherals import LedSet, LedSet`

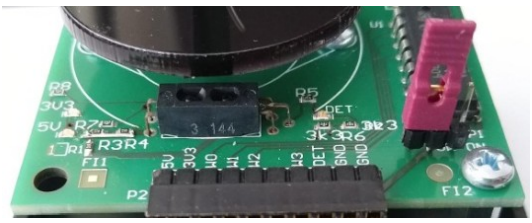
Poniżej pokazano program, który zapala a następnie gasi diodę o numerze 1

```
from peripherals import LedSet, LedClr

LedSet(1)
LedClr(1)
```

Wiadomości wymagane do wykonania zadań : zmienne liczbowe, stałe logiczne, operator przypisania, operator porównania, pętla while

UWAGA:
Przed rozpoczeciem wykonywania zadań upewnić się, że zworka załaczania silnika krokowego jest wpięta jak na obrazku.
Powinna być wpięta po lewej stronie złacza



Zadanie 1

Wymagane wiadomości:

- stałe liczbowe
- funkcje wbudowane: „print”
- funkcje inne:
  - importowanie, wywołanie, argumenty
  - sterowanie ledami: `LedSet, LedClr`

1. Skopiować do katalogu na mikrokontrolerze (Raspberry Pi Pico) plik z modułem peripherals (`peripherals.py`). Źródło pliku wskaże prowadzący.

2. Jeżeli to konieczne stworzyć i zapisać w katalogu mikrokontrolera plik `main.py` z poniższą zawartością

```
from peripherals import LedSet, LedClr

LedSet(2)
LedClr(2)

print('Done !')
```

3. Uruchomić program.
W oknie powłoki powinien pojawić się napis „Done !”

```
Shell ×
>>> %Run -c $EDITOR_CONTENT

  Done !

>>>
```

4. Dlaczego dioda się nie zaświeciła?

Zadanie 2

Wymagane wiadomości: generowanie opóźnienia (funkcja `sleep`)

1. Wstawić po wywołaniu funkcji LedSet opóźnienie jednosekundowe opóźnienie
   W tym celu użyć funkcji `sleep`, która jako argument wywołania przyjmuje czas wyrażony w sekundach.
   Nie zapomnieć o konieczności importu funkcji. Znajduje się ona w module time, będącym jednym z „gotowych modułow” MicroPythona)
2. Uruchomić program. Dioda powinna zaświecić się na 1 sekundę a następnie zgasnąć.
3. Wstawić kod programu do poniższej ramki

```
from peripherals import LedSet, LedClr
from time import sleep

LedSet(1)
sleep(1)
LedClr(1)

print('Done')
```

Zadanie 3

Zmodyfikować poprzedni program tak aby zapalał diody po kolei od 0 do 3 a następnie gasił od 3 do 0.

Zachować 1 sekundowe opóźnienia.

Wstawić kod programu do poniższej ramki. Pominąć instrukcje importu.

```
from peripherals import LedSet, LedClr
from time import sleep

LedSet(0)
sleep(1)

LedSet(1)
sleep(1)

LedSet(2)
sleep(1)

LedSet(3)
sleep(1)

LedClr(3)
sleep(1)

LedClr(2)
sleep(1)

LedClr(1)
sleep(1)

LedClr(0)
sleep(1)

print('Done')
```

Zadanie 4

Wymagane wiadomości: pętla while, zmienne liczbowe

Zmodyfikować poprzedni program tak aby zamigał diodą o numerze 2 trzy razy a następnie zgasił diodę (_*_*_*_)

W tym celu użyć pętli `while`

Wstawić kod programu do poniższej ramki. Pominąć instrukcje importu.

```
i = 0
while(i<3):
    LedSet(2)
    sleep(1)
    LedClr(2)
    sleep(1)
    i = i+1
```

Zadanie 5

Wymagane wiadomości: stałe logiczne

Zmodyfikować poprzedni program tak aby:

- zapalił diodę 1
- zapalał i gasił diodę 2 w nieskończoność (pętla nieskończona),
  przy czym dioda powinna pozostać:
  - zostać zapalona przez 10 milisekund
  - zgaszona przez 90 milisekund

Podpowiedź: funkcja sleep przyjumuje jako argument również liczby z przecinkiem

Sprawdzić czy dida zachowuje się zgodnie z przwidywaniem.

Wstawić kod programu do poniższej ramki. Pominąć instrukcje importu.

```
LedSet(1)
while(True):
    LedSet(2)
    sleep(0.01)
    LedClr(2)
    sleep(0.09)
```

Zadanie 6

Skrócić pięciokrotnie czasy świecenia i zgaszenia diody 2

Po czym poznać że dioda pulsuje?: ...

Wstawić kod programu do poniższej ramki. Pominąć instrukcje importu.

```
LedSet(1)
while(True):
    LedSet(2)
    sleep(0.002)
    LedClr(2)
    sleep(0.0018)
```

Funkcja `LedPoint` – definicja

Zadanie 7

Wymagane wiadomości: definowanie funkcji, przekzywanie argumentów do funkcji

1. Zdefiniować funkcję `LedPoint()`, której zadaniem jest zapalać diodę o numerze podanym w argumencie wywołania funkcji. Pozostałe diody powinny zostać zagaszne.

   Podanie jako argument wywołania wartości `None` powinno gasić wszystkie diody.

2. Przetestować funkcję poniższym kodem

```
i = 0
while(i<4):
    LedPoint(i)
    i += 1
    sleep(1)
LedPoint(None)
```

3. Wstawić kod programu do poniższej ramki. Pominąć instrukcje importu.

```
def LedPoint(pos):
    for i in range(4):
        LedClr(i)
    if pos != None:
        LedSet(pos)

i = 0
while(i<4):
    LedPoint(i)
    i += 1
    sleep(1)
LedPoint(None)
```

Zadanie 8

Wymagane wiadomości: tworzenie bibliotek\modułów

1. Stworzyć moduł `my_peri` i przenieść do niego funkcję `LedPoint`. W tym celu:
   - stworzyć plik o nazwie my_peri.py
   - wstawić do niego kod funkcji `LedPoint` oraz importy funkcji niezbędnych do jej działania
   - zapisać plik na dysku mikrokontrolera
2. W pliku `main.py` zastąpić definicję funkcji `LedPoint` jej importem
3. Sprawdzić działanie programu
4. Wstawić kod modułu `peri`

```
from peripherals import LedSet, LedClr

def LedPoint(pos):
    for i in range(4):
        LedClr(i)
    if pos != None:
        LedSet(pos)
```

5. Wstawić **cały** kod z pliku `main.py`.

```
from my_peri import LedPoint
from time import sleep

i = 0
while(i<4):
    LedPoint(i)
    i += 1
    sleep(1)
LedPoint(None)
```

Funkcja `LedPoint` – testy

Zadanie 9

Wymagane wiadomości: listy, pętla for, iterowanie po listach z użyciem pętli for

Jednokrotnie przesuniecie punktu od Led0 do Led3 i zagaszenie

a) w dowolny sposób

```
from my_peri import LedPoint
from time import sleep

LedPoint(0)
Sleep(1)
LedPoint(1)
sleep(1)
LedPoint(2)
sleep(1)
LedPoint(3)
sleep(1)
LedPoint(None)
```

b) z dokładnie dwoma wystąpieniami funkcji Led* w kodzie (użyć pętli while)

```
i = 0
while(i<4):
    LedPoint(i)
    i += 1
    sleep(1)
LedPoint(None)
```

c) podpunkt b) ale z użyciem: pętli for i listy

```
for i in 0,1,2,3:
    LedPoint(i)
    sleep(1)
LedPoint(None)
```

d) podpunkt c) ale z jednym wystąpieniami funkcji Led* w kodzie

```
for i in 0,1,2,3:
    LedPoint(i)
    sleep(1)
LedPoint(None)
```

Zadanie 10

Jednokrotne przesuniecie punktu od Led 0..3,3..0 i zgaszenie

a) w dowolny sposób

```
for i in 0,1,2,3:
    LedPoint(i)
    sleep(1)
for i in 3,2,1,0:
    LedPoint(i)
    sleep(1)
```

b)

- z użciem jednej pętli i z jednym wystąpieniami funkcji Led* w kodzie
- pętla powinna być typu while
- bez użycia listy

```
i = 0
while(i<8):
    if i < 4:
        j = i
    else:
        j = 3-i
    LedPoint(j)
    i = i+1
    sleep(0.5)
```

Zadanie 11

Wymagane wiadomości: instrukcja warunkowa if-elif-else

Ciągłe przesuwanie punktu Led (0..3,3..0,0..3 itd.)

a) z trzema pętlami

```
while(True):
    for i in 0,1,2,3:
        LedPoint(i)
        sleep(0.5)
    for i in 3,2,1,0:
        LedPoint(i)
        sleep(0.5)
```

b) z jedną pętlą

```
i = 0
j = 1

while(True):

    if i == 0:
        j = 1
    elif i == 3:
        j = -1

    i = i + j

    LedPoint(i)
    sleep(0.5)
```

Zadanie 12

Ciągłe ustawianie punktu Led na pozycje: 0,3,1,2,zgaszone

a) z dwoma pętlami

```
while(True):
    for i in 0,3,1,2,None:
        LedPoint(i)
        sleep(0.5)
```

Funkcja `LedBar` – definicja i test

Zadanie 13

a) Zdefiniować funkcję `LedBar(led_nr)`, której zadaniem jest zapalanie liczby diód dpodanej w argumnecie.
Wstawić funkcje do biblioteki my_peri

```
led_nr : led3..led0
   0   : _ _ _ _
   1   : _ _ _ *
   2   : _ _ * *
   3   : _ * * *
   4   : * * * *
```

b) Napisać program testujący funkcję dla wszystkich sensownych wartości argumentów funkcji.
Program umieścić w pętli nieskończonej.

```
while(True):
    for i in 0,1,2,3,4:
        LedBar(i)
        sleep(0.5)
```

Kontrola wypełnienia (PWM, ang. Pulse With Modulation)

Zadanie 14

Napisać program który bedzie cyklicznie gasił i zapalał diodę 0.

Nalezy używyć funkcji LedSet i LedClr.

Częstotliwość pętli głownej powinna wynośić 50 Hz.

Dioda powinna byc zapalona i zgaszona przez taki sam okres czasu.

Poruszać energicznie płytką przed oczami w lewo i prawo. Co widać?

```
while(True):
    sleep(0.01)
    LedSet(0)
    sleep(0.01)
    LedClr(0)
```

Zadanie 15

Zmodyfikowac poprzedni program tak aby::

- dodatkowo na poczatku, tj. przed pętlą główną, zapalał diodę 1

- stosunek czasu świecenia do całego okresu pętli wynosił do 1/20

Uwaga: należy zachować poprzednia częstotliwość (okres) pętli

```
LedSet(1)
while(True):
    LedSet(0)
    sleep(0.001)
    LedClr(0)
    sleep(0.019)
```

Funkcja `LedDimm`

Zadanie 16

Zdefinować funkcję LedDimm(pwm,time), która zaswieci diodę zero z wypełnieniem podanym w pierwszym argumencie (pwm) przez czas podany w drugim argumencie (time).

Wypełnienie powinno byc wyrażone w procentach a czas w sekundach.

Należy uzyć pętli for.

Program testowy powiniem w nieskończoność zmieniać wypełnienie świecenia leda w sposób podany poniżej: 5,10,15,20,15,10.

Zmiana powinna nastepować co 0.5 sekundy

Nalezy użyc po jedej pętli while i for.

```
def LedDimm(pwm,time):
    time_on = (pwm/100)*0.020
    time_off = 0.020-time_on
    for i in range(50*time):
        LedSet(0)
        sleep(time_on)
        LedClr(0)
        sleep(time_off)

while(True):
    for pwm in 5,10,15,20,15,10:
        LedDimm(pwm,0.5)
```