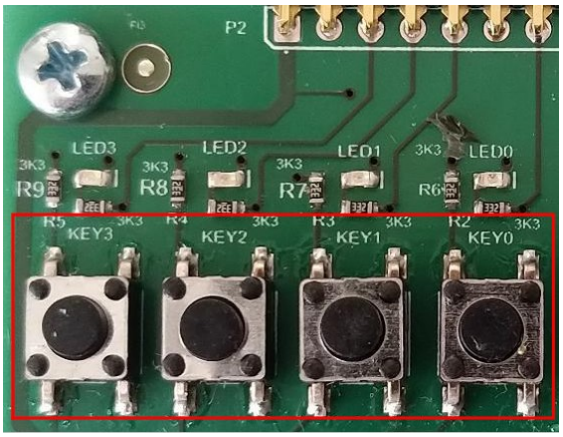
Wprowadzenie

Płytka mikrokontrolera zestawu sprzętowego została wyposażona, między innymi, w cztery przyciski (Key0.. Key3).



Odczyt przycisków odbywa się z użyciem funkcji `ButRead()`. Jako argument funkcja przyjmuje numer przycisku do odczytania. Jeżeli dany przycisk jest naciśnięty funkcja zwraca `True`, jw przeciwnym razie zwraca `False`. Podobnie jak funkcja do sterowania ledami funkcja `ButRead()` zostały zdefiniowana w module `peripherals`.

Wymagane wiadomości do wykonania zadań: tkod binarny, kod „1 z x”, typ „lista”, pętla „for”, iterowanie po liście

Funkcaj ButRead - testy

Zadanie 1

1. Napisać program, który będzie 5 razy na sekundę wyświetlał stan wszystkich przycisków. Przykładowo:
   ```
   False True True False # kolejność przycisków jak na zdjęciu
   False True True False
   …
   ```
   Podpowiedź: funkcja print może przyjąć dowloną liczbę argumentów
2. Wstawić **cały** kod z pliku `main.py`.

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

while(True):
    print(ButRead(3),ButRead(2),ButRead(1),ButRead(0))
    sleep(0.20)
```

Zadanie 2

1. Napisać program, który będzie zapalał\gasił diode 0 jeżeli przycisk 0 jest naciśnięty\zwolniony. Usunąć opóźnienie z kodu
2. Wstawić kod z pliku `main.py` bez importów

```
while(True):
    if ButRead(0):
        LedPoint(0)
    else:
        LedPoint(None)
```

Zadanie 3

1. Zmodyfikować poprzedni program tak aby:
    - zapalał diode 0 wtedy i tylko wtedy jeżeli wciśniete są tylko oba skrajne przyciski
    - zapalał diode 1 wtedy i tylko wtedy jeżeli wciśniete są tylko oba wewnętrzne przyciski
2. Wstawić kod z pliku `main.py` bez importów

```
while(True):

    if ButRead(0) and KeyRead(3):
        LedPoint(0)
    elif ButRead(1) and KeyRead(2):
        LedPoint(0)
    else:
        LedPoint(None)
```

Zadanie 4

1. Zmodyfikować poprzedni program tak aby realizował transkode kodu binarnego na 1 z 4 Dla uproszczenia naciśniecie\zwolnienie przycisku oraz świecenie\zgasznie diody reprezentują cyfry 1\0

Przyciski (kod binarny)	Diody (kod 1 z 4)
Key1|Key0	Led3|Key2|Led1|Led0
00	0001
01	0010
10	0100
11	1000

2. Wstawić kod z pliku `main.py` bez importów

```
while(True):
    if not ButRead(1) and not KeyRead(0):
        LedPoint(0)
    elif not ButRead(1) and KeyRead(0):
        LedPoint(1)
    elif ButRead(1) and not KeyRead(0):
        LedPoint(2)
    elif ButRead(1) and KeyRead(0):
        LedPoint(3)
```

Zadanie 5

1. **Dodać** do programu obsługę przycisku 3 polegającą na tym, że jeżeli przycisk jest wciśnięty to diody migają zamiast świeci. Miganie powinno odbywać się z częstotliwością 10 Hz.
2. Wstawić kod z pliku `main.py` bez importów

```
while(True):
    if not ButRead(1) and not KeyRead(0):
      LedPoint(0)
    elif not ButRead(1) and KeyRead(0):
      LedPoint(1)
    elif ButRead(1) and not KeyRead(0):
      LedPoint(2)
    elif ButRead(1) and KeyRead(0):
      LedPoint(3)

    sleep(0.05)

    if ButRead(3):
      LedPoint(None)

    sleep(0.05)
```

Zadanie 6

Wymagne wiadomości: konwersje typów, jwane i niejawne -operacje arytmetyczne na zmiennych logicznych

Napisac program który w sposób ciągły będzie zapalał tyle diod ile jest naciśniętych przycisków.

Użyć najbardziej odpowiedniej ze zdefiniowanych do tąd funkcji do sterowania ledami.

a) dowolna implementacja

```
while(True):
    i = ButRead(0)+ButRead(1)+ButRead(2)+ButRead(3)
    LedBar(i)
```

b) implementacja z jednokrotnym wystapieniem w kodzie funkcji `ButRead`

```
while(True):
    j = 0
    for i in range(4):
        j = j + ButRead(i)
    LedBar(j)
```

Funkcaj ReadKeyboard – definicja

Zadanie 7

1. Zdefiniować funkcję `ReadKeyboard()`, która będzie zwracać numer naciśniętego przycisku w momencie wywołania funkcji.

   Jeżeli żaden przycisk nie jest naciśnięty funkcaj powinna zwracać `None`

   Jeżeli naciśnieych jest więcej niż jeden przycisk funkcja powinna zwracać numer pierwszego naciśnietego przycisku licząc od zerowego.

2. Funkcja powinna:

   - używać sekwencji `if-elif-elif ...`
   - być zdefiniowana w pliku `main.c`
   - `zostać` przetestowana kodem jak poniżej.

```
while(True):
      print(ReadKeyboard())
      sleep(0.2)
```

3. Wstawić kod z pliku `main.py` bez importów

```
def ReadKeyboard():
    if ButRead(0):
        return 0
    elif ButRead(1):
        return 1
    elif ButRead(2):
        return 2
    elif ButRead(3):
        return 3
    else:
        return None

while(True):
    print(ReadKeyboard())
    sleep(0.2)
```

Zadanie 8

1. Ograniczyć powtarzanie się kodu w funkcji `ReadKeyboard()` używając pętli `for` oraz funkcji `range()`.
2. Przenieśc funkcję `ReadKeyboard()` do biblioteki `my_peri`
3. Wstawić kod z pliku `main.py` bez importów

```
def ReadKeyboard():
    for i in range(4):
        if ButRead(i):
            return i
    return None

while(True):
    print(ReadKeyboard())
    sleep(0.2)
```

4. Wstawić kod modułu my_peri

```
from peripherals import LedSet, LedClr,
ButRead

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

def LedBar(led_nr):
    for i in range(4):
        LedClr(i)
    for i in range(led_nr):
        LedSet(i)

def ReadKeyboard():
    for i in 0,1,2,3:
        if ButRead(i):
            return i
    return None
```

Funkcja ReadKeyboard – testy

Zadanie 9

1. Napisać program, który będzie zapalał diodę o numerze takim jak numer naciśniętego przycisku Jeżeli nie nacisnięto żadnego przycisku wszystkie diody powiny być zgaszone.

2. Wstawić kod z pliku `main.py` bez importów

```
while(True):
    LedPoint(ReadKeyboard())
```

Zadanie 10

Zmodyfikować poprzedni program tak aby zwolnienie przycisku nie powodowało żadnej akcji.

Innymi słowy zawsze powinien się świecić ten led którego przycisk ostatnio nacisnieto.

```
while(True):
    but = ReadKeyboard()
    if but != None:
        LedPoint(but)
```

Zadanie 11

1. Zmodyfikować poprzedni program tak aby zapalała się dioda o numerze przciwnym do numeru naciśniętego przycisku (0→3, 1→2, 2→1, 3->0).

2. Wstawić kod z pliku `main.py` bez importów

```
while(True):
    but = ReadKeyboard()
    if but != None:
        LedPoint(3-but)
```

Funkcja ReadKeyboard – testy – wykrywanie zmiany

Zadanie 12

Napisać program, który każde naciśnieciu dowolnego przycisku spowoduje pojedyncze wydrukowanie na konsoli napisu

a) „but”.

b) „but [numer przycisku]”, czyli przykładowo „but 3”

Program powinie posiadać dwie zmienne, służące do przechowywania obecnego (zmienna but_current) i poprzedniego (but_previous) stanu klawiatury.

Wydruk na konsolę pownien mieć miejsce jeżeli aktualnie przycisk jest naciśnięty a poprzednio był zwolniiny

Odczyt przycisków powinien odbywac się z częstotliwością 10 Hz.

```
while(True):
    but_current = ReadKeyboard()
    if (but_current != None) and (but_prev == None):
        print('but', but_current)
    but_prev = but_current
    sleep(0.1)
```

Zadanie 13

Zmodyfikować poprzedni program tak aby zliczał liczbę naciśnieć przycisków.

a) Zawartośc licznika powinna być przentowana na konsoli przy każdym naciśnięciu

```
while(True):
    but_current = ReadKeyboard()
    if (but_current != None) and (but_prev == None):
        print('but', but_current)
    but_prev = but_current
    sleep(0.1)
```

b) Zawartość licznika powinn być prezentowana na ledach z użyciem funkcji LedBar, przy czym liczba w liczniku pownna być ograniczona funkcją modulo 5

```
counter = 0
while(True):
    but_current = ReadKeyboard()
    if (but_current != None) and (but_prev == None):
        counter = (counter + 1)%5
        LedBar(counter)
    but_prev = but_current
    sleep(0.1)
```

c) Zmodyfikować podpunkt b) tak aby

- przycisk 0 powodował dekrementację licznika a przycisk 1 jego inkrementację
- wartość licznika nie wychodziła poza zakres 0-4

```
counter = 0
while(True):
    but_current = ReadKeyboard()
    if (but_current != None) and (but_prev == None):

        if but_current == 0:
            counter = counter - 1
        if but_current == 1:
            counter = counter + 1

        if counter < 0:
            counter = 0
        elif  counter > 4:
            counter = 4

        print(counter)
        LedBar(counter)

    but_prev = but_current
    sleep(0.1)
```