

INSTYTUT INFORMATYKI

Wydział IEiT AGH



Ćwiczenie laboratoryjne

Raspberry Pi

Nazwa kodowa: **rpi-intro**. Wersja **20231120.0**

Ada Brzoza

ada.brzoza@agh.edu.pl

Wstęp

1 Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z wybranymi zagadnieniami tworzenia oprogramowania dla komputerów jednopłytkowych korzystając z wysokopoziomowego języka programowania.

2 Materiały

Wymienione poniżej materiały są dostępne w laboratorium, w którym prowadzone są zajęcia z platformą Raspberry Pi. W przypadku samodzielnej lub zdalnej realizacji ćwiczenia należy zaopatrzyć się w te nie we własnym zakresie. Potrzebujemy następujące elementy:

1. komputer umożliwiający na wgranie obrazu dysku na kartę pamięci SD, np. Disks (gnome-disks), lub odpowiednik,
2. karta pamięci SD (SDHC), najlepiej o pojemności 16 GB lub większej przy zalecanej klasie szybkości C10 lub UHS-1 (wolniejsze karty także powinny zadziałać, jednak spowodują niepotrzebnie spowolnienie działania systemu),
3. czytnik kart SD (wbudowany w komputer lub z interfejsem USB),
4. komputer Raspberry Pi 3B lub 4B (dla jednej z tych wersji Raspberry Pi został opracowany załączony do materiałów obraz systemu, jednak jeśli zaczynamy od instalacji nowego obrazu, można użyć właściwie dowolnego modelu Raspberry Pi z mikroprocesorem aplikacyjnym),
5. monitor komputerowy z interfejsem HDMI – raczej dla ułatwienia, ponieważ z Raspberry Pi można łączyć się także przez SSH,
6. klawiatura i mysz z interfejsami USB (alternatywnie można uzyskać dostęp do Raspberry Pi przez interfejs sieciowy i SSH, jednak jest to nieco trudniejsze niż podejście opisane w niniejszej instrukcji),
7. płytki stykowe typu breadboard – wystarczy breadboard o niewielkich rozmiarach,
8. dioda LED przewlekana,
9. przycisk chwilowy typu tact-switch,
10. rezystor 1 k Ω ,
11. trzy przewody 1-pinowe wtyk-gniazdo do płytek breadboard,
12. przewód USB A-microB do zapewnienia zasilania Raspberry Pi z komputera – alternatywnie można użyć zasilacza-ładowarki 5 V o wysokiej wydajności prądowej (przynajmniej 2 A = 2000 mA), z wtyczką USB-microB.

3 Plan wykonania ćwiczenia

Wykonanie ćwiczenia będzie podzielone na trzy etapy. W pierwszym etapie przygotujemy system operacyjny Raspberry Pi OS na karcie pamięci SD i uruchomimy Raspberry Pi. W drugiej części dołączymy do komputera Raspberry Pi prosty układ elektroniczny, który oprogramujemy w części trzeciej.

Część 1. Przygotowanie nośnika z systemem operacyjnym

Platforma Raspberry Pi została zaprojektowana tak, aby cały system mógł znajdować się na dołączanej do płytki karcie pamięci SDHC. Kompletny obraz systemu operacyjnego można pobrać z oficjalnej strony Raspberry Pi: <https://www.raspberrypi.org/>.

Raspberry Pi OS jest systemem dystrybuowanym w formie obrazu, który możemy wgrać na kartę przy pomocy powszechnie dostępnych narzędzi. Do tego ćwiczenia oraz kolejnych dostarczono specjalnie przygotowany obraz systemu Raspberry Pi OS z zainstalowanymi dodatkowymi narzędziami, przydatnymi w czasie prac w laboratorium. Dla osób zainteresowanych zalecane jest samodzielne przejście przez instalację systemu na karcie pamięci z dostępnego obrazu. Narzędziem pozwalającym na zainstalowanie obrazu systemu operacyjnego dla Raspberry Pi na karcie SDHC może być np. Gnome Disks dostępne na komputerach w laboratorium.

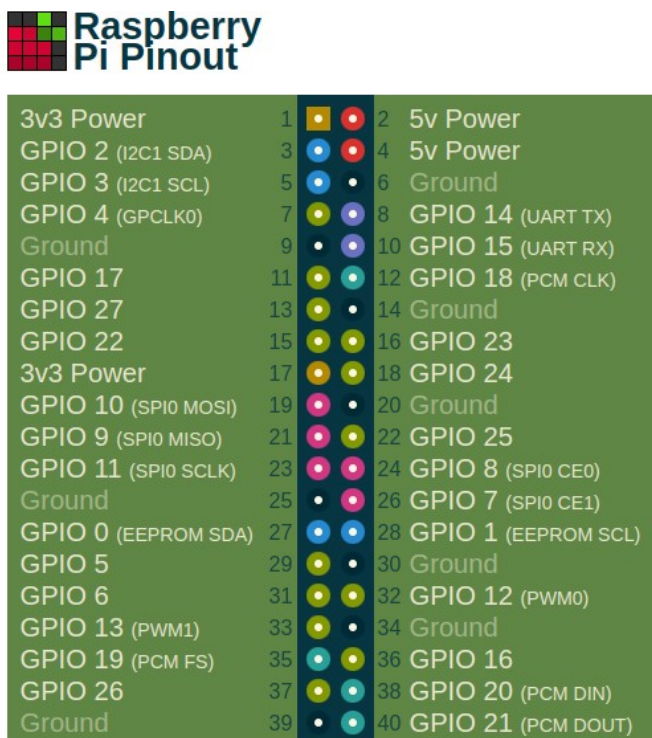
Po wgraniu pliku obrazu systemu Raspberry Pi OS karta SD powinna być gotowa do przełożenia do komputera Raspberry Pi. Po włożeniu przygotowanej karty możemy podłączyć komputer Raspberry Pi: do monitora, klawiatury, myszy, opcjonalnie interfejsu Ethernet i – na koniec – zasilania. Chwilę po włączeniu zasilania system Raspberry Pi OS powinien być gotowy do użycia. Jak można się przekonać, jest to system o przyjaznym, graficznym interfejsie użytkownika z zainstalowanymi licznymi narzędziami, w tym nawet przeglądarką internetową.

W dalszych częściach ćwiczenia dowiemy się, jak zastosować złącze rozszerzeń dostępne na platformie Raspberry Pi. Umożliwia ono łatwą realizację bardzo wielu urządzeń oraz ciekawych projektów wymagających interakcji z prostymi układami zewnętrznymi.

Część 2. Podłączenie prostego sprzętu do złącza rozszerzeń komputera Raspberry Pi

Na początek zestawmy prosty układ testowy z diodą LED oraz włącznikiem chwilowym. Oprogramowaniem tego układu zajmiemy się w następnej kolejności. **Pamiętajmy, aby wykonywać tę część ćwiczenia przy wyłączonym napięciu zasilania komputera Raspberry Pi.**

Widok złącza rozszerzeń wraz z opisem wyprowadzeń znajdziemy na stronie <http://pinout.xyz/>. Dla wygody poniżej został zamieszczony zrzut ekranu strony głównej pinout.xyz, jednak warto również zajrzeć na tę stronę bezpośrednio, ponieważ diagram jest interaktywny ;)



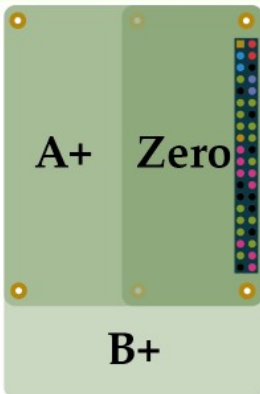
Raspberry Pi Pinout

Pin	Function	Pin	Function
1	3v3 Power	2	5v Power
3	GPIO 2 (I2C1 SDA)	4	5v Power
5	GPIO 3 (I2C1 SCL)	6	Ground
7	GPIO 4 (GPCLK0)	8	GPIO 14 (UART TX)
9	Ground	10	GPIO 15 (UART RX)
11	GPIO 17	12	GPIO 18 (PCM CLK)
13	GPIO 27	14	Ground
15	GPIO 22	16	GPIO 23
17	3v3 Power	18	GPIO 24
19	GPIO 10 (SPI0 MOSI)	20	Ground
21	GPIO 9 (SPI0 MISO)	22	GPIO 25
23	GPIO 11 (SPI0 SCLK)	24	GPIO 8 (SPI0 CE0)
25	Ground	26	GPIO 7 (SPI0 CE1)
27	GPIO 0 (EEPROM SDA)	28	GPIO 1 (EEPROM SCL)
29	GPIO 5	30	Ground
31	GPIO 6	32	GPIO 12 (PWM0)
33	GPIO 13 (PWM1)	34	Ground
35	GPIO 19 (PCM FS)	36	GPIO 16
37	GPIO 26	38	GPIO 20 (PCM DIN)
39	Ground	40	GPIO 21 (PCM DOUT)

Legend

Orientate your Pi with the GPIO on the right and the HDMI port(s) on the left.

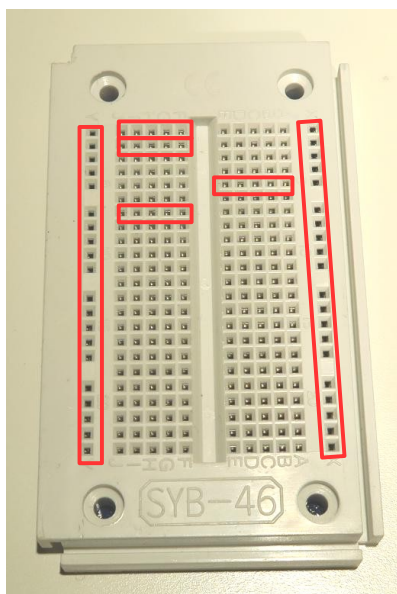
- GPIO (General Purpose IO)
- SPI (Serial Peripheral Interface)
- I²C (Inter-integrated Circuit)
- UART (Universal Asynchronous Receiver/Transmitter)
- PCM (Pulse Code Modulation)
- Ground
- 5V (Power)
- 3.3V (Power)



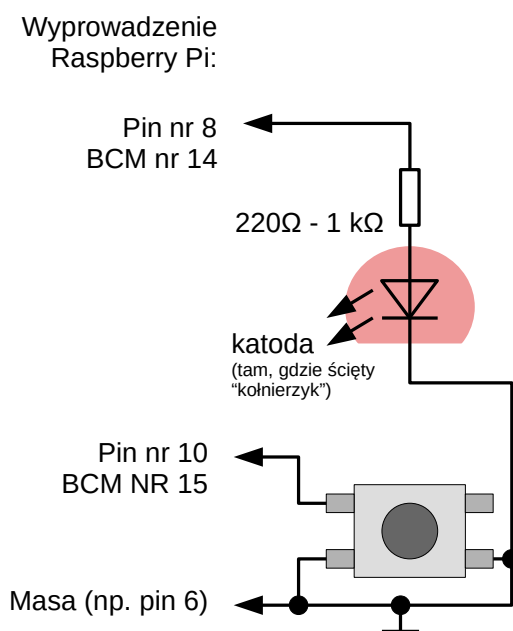
W ramach ćwiczenia spróbujemy podłączyć:

- diodę **LED** do wyprowadzenia **nr 8** złącza GPIO (alternatywne oznaczenie BCM14);
- **przycisk chwilowy** (tzw. *tact switch*) do wyprowadzenia **nr 10** złącza GPIO (alternatywne oznaczenie BCM15).

Do podłączenia przycisku i diody LED można użyć dostępnych płytek typu „breadboard”. Typowe płytki typu breadboard mają wykonane wewnętrzne połączenia, z których kilka zostało zaznaczone na poniższym rysunku. Skrajnych, pionowych grup często używa się do dystrybucji zasilania.



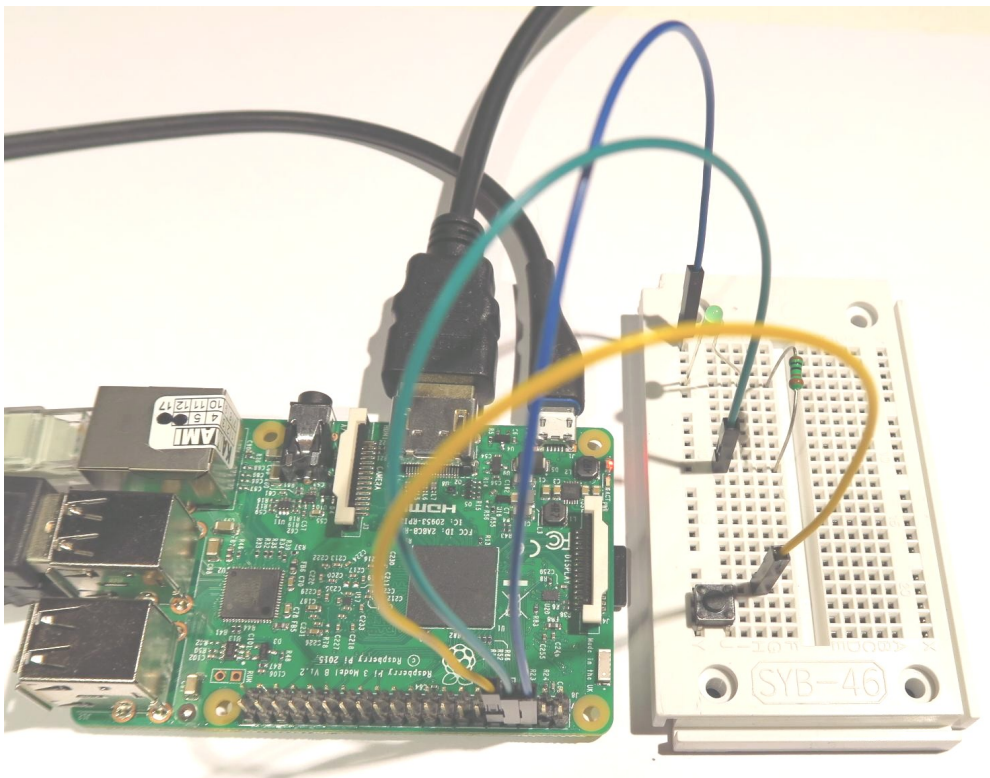
W ramach ćwiczenia zmontujemy na płytce stykowej bardzo prosty układ składający się ze wspomnianych elementów: przycisku i diody LED wg poniższego schematu.



Zauważmy kilka ważnych rzeczy.

1. Podłączając diodę LED do mikrokontrolera (jak w Raspberry Pi) lub układu cyfrowego, warto pamiętać o rezystorze szeregowym ograniczającym prąd diody. Obecnie, przy logice pracującej przy napięciach 5 V lub 3,3 V, może to być rezystor o wartości nawet 1 k Ω . Zapewni on co prawda bardzo mały prąd dla diody LED (rzędu pojedynczych mA), jednak przy nowoczesnych diodach LED będzie i tak wystarczający do wyraźnego zaświecenia diody, a z drugiej strony mały prąd diody sprzyja energooszczędności.
2. Katoda przewlekanej, okrągłej diody LED najczęściej (ale nie zawsze) oznaczana jest przez ścięty fragment kołnierza obudowy. Katodę diody LED podłączamy do niższego potencjału – tutaj do masy.
3. Typowe włączniki chwilowe *tact switch* (*tactile switch*) mają 4 wyprowadzenia. Wyprowadzenia te połączone są wewnątrz włącznika parami. Przyjęło się, że we właściwym układzie także w miarę możliwości łączy się je parami zgodnie z połączeniami wewnętrznymi.
4. W przypadku Raspberry Pi mamy możliwość włączenia rezystorów podciągających oraz ściągających wejście odpowiednio do stanu wysokiego albo niskiego. Po włączeniu rezystora podciągającego, „domyślnym” stanem będzie stan wysoki. Ta opcja ma zastosowanie do schematu z powyższego rysunku. Przycisk będzie mógł natomiast ten stan zmienić na niski zwierając wyprowadzenie silnie do masy.

Widok układu zmontowanego na płytce stykowej i dołączonego do Raspberry Pi przedstawiono na poniższym rysunku.



Część 3. Oprogramowanie interfejsu GPIO w języku Python

Zacznijmy od uruchomienia podstawowych narzędzi.

Gdy korzystamy ze środowiska graficznego, wystarczy uruchomić dowolny edytor tekstowy oferowany przez środowisko graficzne lub z terminala. Dla wygody warto rozważyć edytor podświetlający składnię. W przygotowanym obrazie systemu Raspberry Pi OS mamy dostępny np. Geany. Możemy znaleźć Geany w odpowiedniku „menu start” (logo Raspberry Pi) → sekcja *Programming*. Można również doinstalować własny edytor, jeśli nie zajmie to zbyt dużo czasu.

Oprócz tego potrzebujemy terminala, z którego będzie można uruchamiać pisane skrypty. Terminal możemy uruchomić zarówno „klikając” w GUI lub używając skrótu klawiszowego Ctrl + Alt + T. W terminalu rozpoczynamy pracę od katalogu domowego. Warto rozważyć założenie własnego podkatalogu o unikalnej nazwie, w którym będziemy pracować:

```
mkdir [jakaś nasza unikalna nazwa]
```

Nawigujemy do katalogu, w którym chcemy pracować używając

```
cd [założony właśnie katalog]
```

Następnie tworzymy pusty plik o nazwie *blink.py*:

```
touch blink.py
```

Otwieramy plik przy pomocy wybranego edytora i wpisujemy treść.

1. Dobrą praktyką może być wskazanie na samym początku interpretera, którego będziemy używać (bez tego też jednak zadziała ;)):
#!/bin/python3.9
2. Do eksperymentów przyda nam się dodanie biblioteki time,
import time
dzięki której będzie można realizować opóźnienia czasowe w naszym skrypcie.
3. Aby dostać się do interfejsu GPIO możemy skorzystać z gotowego modułu RPi.GPIO pisząc np.
import RPi.GPIO as GPIO
4. Aby uniknąć wyświetlania ostrzeżeń, które nie są teraz dla nas istotne, możemy napisać:
GPIO.setwarnings(False)
5. Do poszczególnych wyprowadzeń GPIO można odnosić się używając numerów wyprowadzeń na złączu (BOARD) lub nazw wyprowadzeń mikrokontrolera (BCM). Tutaj użyjemy numeracji wyprowadzeń wg numerów wyprowadzeń złącza (BOARD):
GPIO.setmode(GPIO.BOARD)

(Gdybyśmy chcieli używać numerów wyprowadzeń mikrokontrolera, należałoby tutaj wpisać **GPIO.BCM**)

6. Konfiguracja wyprowadzeń odbywa się przez wywołanie metody *GPIO.setup*. I tak, jeśli chcemy skonfigurować np. wyprowadzenie 8 jako wyjście cyfrowe, wystarczy że napiszemy

```
GPIO.setup(8,GPIO.OUT)
```

Konfiguracja wyprowadzenia jako wejście cyfrowe odbywa się analogicznie, tym razem jednak warto podać kierunek działania aktywnego rezystora ustalającego domyślny stan logiczny (podciągającego lub ściągającego). Jeśli używane do przycisku wyprowadzenie 10 będzie miało pracować z rezystorem podciągającym (opcja *PUD_UP*), to napiszemy

```
GPIO.setup(10,GPIO.IN,GPIO.PUD_UP)
```

7. Do zmiany stanu logicznego wyprowadzenia skonfigurowanego jako wyjście można użyć metody *GPIO.output* podając jako argumenty: numer wyprowadzenia i stan logiczny. Możemy np. włączyć diodę LED na naszej płytce stykowej poleceniem

```
GPIO.output(8,1)
```

Możemy także ją wyłączyć poleceniem

```
GPIO.output(8,0)
```

8. Błyskanie diodą LED można zrealizować choćby w nieskończonej pętli *while*, np. tak jak we fragmencie poniżej (uwaga: tutaj „→” oznacza wcięcie tekstu zrealizowane przy pomocy spacji lub tab, dzięki któremu zaznaczamy kod wewnątrz instrukcji warunkowych, pętli, etc.):

```
while True:  
→GPIO.output(8,1)  
→time.sleep(0.5)  
→GPIO.output(8,0)  
→time.sleep(0.5)
```

9. Wreszcie prosty sposób odczytu stanu wejściowego uzyskamy wywołując metodę *GPIO.input*, np. dla wcześniej podłączonego przycisku:

```
GPIO.input(10)
```

10. Odczyt stanu wyprowadzenia można wykonać w prostym poleceniu warunkowym *if* lub *if-else*, np.:

```
if GPIO.input(10) == 0:  
→print('przycisk wcisniety')  
else:  
→print('przycisk zwolniony')
```

W ramach pierwszej samodzielnej pracy można np.

1. napisać skrypt, w którym każde wciśnięcie przycisku będzie odwracało stan diody LED,
2. podłączyć funkcję callback wypisującą na konsoli komunikat za każdym razem, gdy przycisk będzie wciskany.

Jeśli zamierzamy pisać własne aplikacje korzystające z niskopoziomowych zasobów sprzętowych Raspberry Pi, to dla wygody lub tymczasowo można rozważyć uruchamianie ich z podniesionymi uprawnieniami (sudo), jednak dla bezpieczeństwa staramy się unikać uruchamiania aplikacji w taki sposób, kiedy tylko jest to możliwe.

Zastosowana tutaj prosta biblioteka to:

<https://pypi.org/project/RPi.GPIO/> ,

a jej dokumentacja dostępna jest tutaj:

https://sourceforge.net/p/raspberry-gpio-python/wiki/browse_pages/

Modułem zapewniającym jeszcze bardziej intuicyjny i abstrakcyjny interfejs jest np. `gpiozero`, który został użyty w tutorialu na oficjalnych stronach Raspberry Pi:

<https://projects.raspberrypi.org/en/projects/physical-computing/3>