

Sprawozdanie laboratorium 2

Marcin Jarczewski 330234

Bartosz Jasiński 318777

Cel i przebieg laboratorium

Celem drugiego laboratorium było zapoznanie się ze środowiskiem OpenWR, sposobem obsługi GPIO za pomocą sysfs, oraz prostymi układami wejścia/wyjścia. Dzięki zajęciom udało nam się poszerzyć swoją wiedzę z tego zakresu materiału.

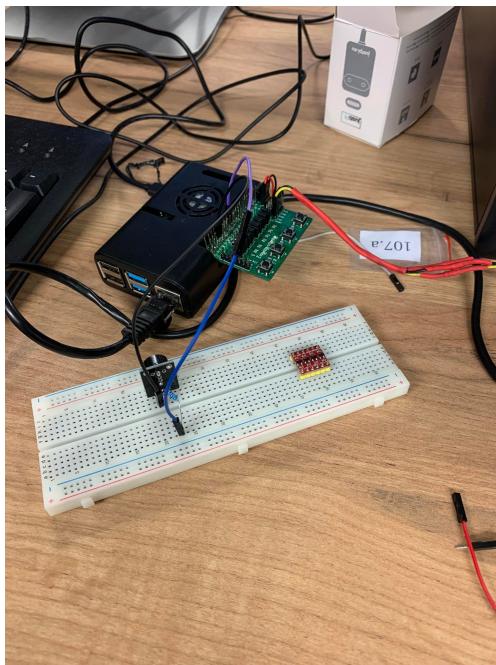
Podczas laboratorium wykonaliśmy zadania numer 1, 2, 3 i 4, które zostały zatwierdzone przez Prowadzącego laboratorium. Zadania numer 5 nie udało nam się wykonać podczas zajęć.

Biblioteki

Przy realizacji zadań laboratoryjnych wykorzystaliśmy biblioteki:

- math
- time
- gpio4

Zadanie 1: GPIO - wyjście dla LED



- Było to pierwsze zadanie wykonane podczas laboratorium, dlatego przed podłączeniem zasilania skonsultowaliśmy z Prowadzącym laboratorium poprawność utworzonego przez nas układu.
- Pierwsze zadanie polegało na napisaniu programu, dzięki któremu 10 krotnie włączy i wyłączy się LED na adapterze do RPi.
- W zadaniu wykorzystaliśmy diodę LED na pinie numer 27

```
import gpio4
import time
gpio27 = gpio4.SysfsGPIO(27)
gpio27.export = True # use the pin
gpio27.direction = 'out' # set direction to output

for _ in range(10):
    gpio27.value = 0 # set value to low
    time.sleep(0.2)
    gpio27.value = 1 # set value to high
    time.sleep(1)

gpio27.export = False # cleanup
```

Zadanie 2: GPIO - wyjście dla LED z płynną zmianą jasności

- Drugie zadanie polegało na napisaniu programu, który przez 10 sekund będzie w płynny sposób zmieniał jasność LED na adapterze do RPi. Trudność w tym zadaniu polegała na płynnej zmianie, wykorzystując przykładowo funkcje sinus. Ostatecznie udało się wykonać zadanie, uzależniając duty_cycle od zmieniającej się wartości sinusa w danej chwili, generując tym samym sygnał PWM o wskazanych w poleceniu parametrach.
- W zadaniu wykorzystaliśmy diodę LED na pinie numer 27

```

import math
import time
import gpio4

gpio27 = gpio4.SysfsGPIO(27)
gpio27.export = True # use the pin
gpio27.direction = 'out' # set direction to output

start = time.time()

def duty_cycle_pwm():
    duty_cycle = 0
    cnt = 0
    period = 0.02 # s
    while True:
        gpio27.value = 1 # set value to high
        time.sleep(period * duty_cycle)
        gpio27.value = 0 # set value to low
        time.sleep(period * (1 - duty_cycle))

        cnt += 0.2
        duty_cycle = (math.sin(cnt) + 1) / 2

        if time.time() - start >= 10:
            break

duty_cycle_pwm()

gpio27.export = False # cleanup

```

Zadanie 3: GPIO - wejście

- Zadanie numer 3 polegało na utworzeniu programu, dzięki któremu po naciśnięciu przycisku na RPI zapali się dioda LED
- W tym zadaniu, wykorzystaliśmy LED na pinie numer 18 i przycisk na pinie numer 27.

```

import gpio4

gpio27 = gpio4.SysfsGPIO(27)
gpio27.export = True # use the pin
gpio27.direction = 'out' # set direction to output
gpio27.value = 0

gpio18 = gpio4.SysfsGPIO(18)
gpio18.export = True # use the pin
gpio18.direction = 'in' # set direction to output

while True:
    if gpio18.value == 0:
        gpio27.value = 1
    elif gpio18.value == 1:
        gpio27.value = 0

    gpio27.value = 0
    gpio27.export = False # cleanup
    gpio18.export = False # cleanup

```

Zadanie 4: GPIO - wyjście PWM, buzzer pasywny

- Zadanie numer 4 polegało na wygenerowaniu sygnału PWN o zmiennej częstotliwości i stałym wypełnieniu 50%
- Program generuje kolejne dźwięki gamy C-dur w 2 oktawach.
- W zadaniu wykorzystaliśmy głośnik na pinie numer 9

```

import math
import time
import gpio4

gpio9 = gpio4.SysfsGPIO(9)
gpio9.export = True # use the pin
gpio9.direction = 'out' # set direction to output

start = time.process_time()

sounds = [
    261,
    294,
    330,
    349,
    392,
    440,
    494,

    523,
    587,
    659,
]

```

```
698,  
784,  
880,  
987  
]  
  
def play(st):  
    gpio9.value = 1 # set value to high  
    time.sleep((1/st))  
    gpio9.value = 0 # set value to high  
    time.sleep((1/st))  
  
def duty_cycle_pwm():  
    duty_cycle = 0.5  
    for sound in sounds:  
        curr_time = time.process_time()  
  
        for _ in range(200):  
            play(sound)  
  
        if time.process_time() - start >= 10:  
            break  
  
    duty_cycle_pwm()  
  
    gpio9.export = False # cleanup
```

Zadanie 5: Akcesoria do wyboru

- Zadania piątego nie udało się wykonać podczas laboratorium.