

Podstawy Systemów Mikroprocesorowych

Laboratorium

Laboratorium 1: Wejścia i wyjścia mikrokontrolera — GPIO, ADC i LCD

Autorzy: Dariusz Tefelski, Angelika Tefelska

Spis treści

1	Wprowadzenie	1
2	Wyposażenie stanowiska	1
3	Zadanie 1: Więcej niż jedna dioda	2
4	Zadanie 2 - sterowanie przyciskami	3
5	Zadanie 3: Pierwszy sygnał analogowy — fotorezystor	3
6	Zadanie 4: Prezentacja wyników na wskaźniku LED	5
7	Zadanie 5: Wyświetlacz LCD 16x2	6
8	Zadanie końcowe: Miernik oświetlenia	7
9	Podsumowanie	8

1 Wprowadzenie

Cel ćwiczenia



Celem ćwiczenia jest rozwinięcie umiejętności pracy z wejściami i wyjściami mikrokontrolera oraz zbudowanie pierwszego prostego systemu kontrolno-pomiarowego. Po wykonaniu ćwiczenia student powinien:

- potrafić obsłużyć kilka wejść i wyjść cyfrowych,
- rozróżniać sygnał cyfrowy i analogowy,
- rozumieć zasadę działania dzielnika napięcia,
- odczytać sygnał analogowy za pomocą wbudowanego przetwornika ADC,
- przeliczyć wynik pomiaru ADC na napięcie,
- wykorzystać wyświetlacz LCD z użyciem dostarczonej biblioteki,
- połączyć czujnik, element wykonawczy i interfejs użytkownika w jeden program.

Na poprzednich zajęciach uruchomiono stanowisko, skonfigurowano podstawowe piny GPIO oraz wykonano pierwszą obserwację sygnału za pomocą oscyloskopu. W tym laboratorium będziemy wykorzystywać większą liczbę wejść i wyjść oraz po raz pierwszy odczytamy sygnał analogowy.

Zakres tego laboratorium



W tym ćwiczeniu ADC traktujemy przede wszystkim jako narzędzie pomiarowe. Szczegółowe zagadnienia dotyczące rozdzielczości, kwantyzacji, błędów przetwarzania i częstotliwości próbkowania zostaną omówione w późniejszym laboratorium poświęconym ADC, DAC i PWM.

2 Wyposażenie stanowiska

Do wykonania ćwiczenia potrzebne są:

- płytką dydaktyczną z Raspberry Pi Pico 2 W,
- przewód USB, przewody połączeniowe,
- **dodatkową diodę LED**
- osiem diod LED i przyciski dostępne na płytce dydaktycznej,
- **fotorezystor,**
- **rezystor 10 k Ω i 330 Ω ,**
- wyświetlacz alfanumeryczny LCD 16x2 znajdujący się na płytce,
- Analog Discovery II.

Uwaga



Raspberry Pi Pico 2 W pracuje z logiką 3,3 V. Do wejść GPIO oraz ADC nie wolno podawać napięcia 5 V. Dzielnik z fotorezystorem zasilaj wyłącznie z linii 3,3 V.

3 Zadanie 1: Więcej niż jedna dioda

Na poprzednich zajęciach sterowano pojedynczą diodą LED. W praktycznych systemach mikrokontrolerowych często wykorzystuje się jednocześnie wiele wejść i wyjść. W tym zadaniu utworzymy prosty wskaźnik złożony z ośmiu diod.

Połącz osiem diod LED z pinami GP6–GP13. Diody na płytce dydaktycznej są sterowane stanem niskim: wartość 0 zapala diodę, a wartość 1 ją gasi.

Zadanie 1 — Linijka LED



Uruchom poniższy program i sprawdź jego działanie.

```
1 from machine import Pin
2 from utime import sleep
3
4 leds = [Pin(pin, Pin.OUT, value=1) for pin in range(6, 14)
5         ]
6
7 def show_level(level):
8     for i, led in enumerate(leds):
9         if i < level:
10             led.value(0)
11         else:
12             led.value(1)
13
14 while True:
15     for level in range(9):
16         show_level(level)
17         sleep(0.2)
```

Następnie zmodyfikuj program tak, aby linijka po osiągnięciu wartości maksymalnej zmniejszała poziom z powrotem do zera.

Stan początkowy pinu — argument value



Przy tworzeniu obiektu klasy Pin można od razu określić stan początkowy wyjścia. Zapis:

```
Pin(pin, Pin.OUT, value=1)
```

oznacza, że wybrany pin zostaje skonfigurowany jako wyjście, a następnie usta-

 wiony w stanie wysokim (1). W tym ćwiczeniu jest to szczególnie wygodne, ponieważ diody LED są sterowane stanem niskim — ustawienie `value=1` powoduje więc, że po uruchomieniu programu wszystkie diody są początkowo zgaszone.

Jak działa funkcja `enumerate()`?

 Funkcja `enumerate()` pozwala podczas przechodzenia po elementach listy otrzymać jednocześnie **numer elementu** oraz **sam element**. Domyślnie numerowanie rozpoczyna się od zera.

W instrukcji:

```
for i, led in enumerate(leds):
```

zmienna `i` przyjmuje kolejno wartości od 0 do 7, natomiast zmienna `led` wskazuje odpowiadający jej obiekt `Pin` z listy `leds`.

4 Zadanie 2 - sterowanie przyciskami

Zadanie 2 — Sterowanie przyciskami

 Podłącz dwa przyciski do pinów GP14 i GP15 i skonfiguruj je z rezystorami *pull-up*. Jeden przycisk powinien zwiększać poziom liniiki LED, a drugi go zmniejszać.

Wartość poziomu powinna pozostawać w zakresie od 0 do 8.

Na tym etapie możesz wykorzystać odpytywanie przycisków w pętli oraz krótkie opóźnienie po wykryciu naciśnięcia. W kolejnym laboratorium zastąpimy takie rozwiązanie przerwaniami.

Zastanów się

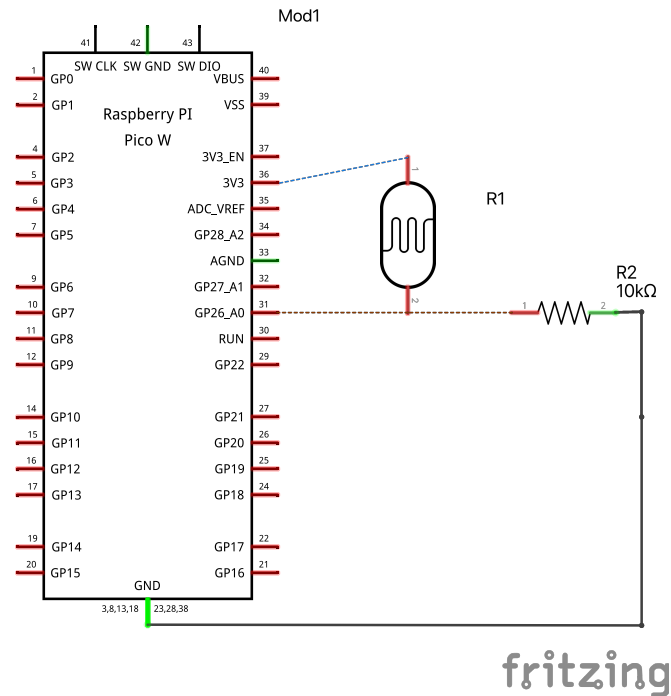
 Co stanie się, jeśli po wykryciu przycisku nie zastosujesz żadnego opóźnienia? Czy jedno fizyczne naciśnięcie musi odpowiadać dokładnie jednej zmianie stanu w programie?

5 Zadanie 3: Pierwszy sygnał analogowy — fotorezystor

Sygnał cyfrowy reprezentujemy za pomocą dwóch stanów logicznych: 0 bądź 1. W wielu pomiarach interesuje nas jednak napięcie, które może przyjmować dowolną wartość z pewnego zakresu. Do przetworzenia napięcia na wartość cyfrową służy przetwornik analogowo-cyfrowy ADC (*Analog-to-Digital Converter*).

Raspberry Pi Pico 2 W udostępnia wejścia ADC między innymi na pinach GP26, GP27 i GP28. W tym ćwiczeniu wykorzystamy GP26 (ADC0).

Fotorezystor (LDR) zmienia rezystancję wraz ze zmianą oświetlenia. Mikrokontroler nie mierzy rezystancji bezpośrednio, dlatego wykorzystamy dzielnik napięcia (patrz rysunek 1).



Rysunek 1: Podłączenie fotorezystora.

Dla takiego połączenia napięcie na wejściu ADC można zapisać jako:

$$U_{ADC} = 3.3 \text{ V} \cdot \frac{R}{R + R_{LDR}}, \quad (1)$$

gdzie $R = 10 \text{ k}\Omega$. Przy silniejszym oświetleniu rezystancja typowego fotorezystora maleje, dlatego w przedstawionym układzie napięcie na wejściu ADC rośnie.

Zadanie 3 — Odczyt ADC



Zbuduj dzielnik napięcia zgodnie ze schematem z rysunku 1 i uruchom program:

```
1 from machine import ADC
2 from utime import sleep_ms
3
4 ldr = ADC(26)
5
6 while True:
7     raw = ldr.read_u16()
8     voltage = raw * 3.3 / 65535
```



```
print(raw, voltage)
sleep_ms(200)
```

Sprawdź wskazania dla co najmniej trzech warunków:

1. normalne oświetlenie stanowiska,
2. fotorezystor zasłonięty dłonią,
3. fotorezystor oświetlony z bliska latarką telefonu.

Zanotuj wartości napięcia dla trzech przypadków. Czy kierunek zmian jest zgodny z przewidywaniem wynikającym z dzielnika napięcia?

Interpretacja



Funkcja `read_u16()` zwraca wynik przeskalowany do zakresu od 0 do 65535. Nie oznacza to, że fizyczny przetwornik w Pico ma rozdzielczość 16 bitów lecz ma 12-bitów. Szczegółowo wrócimy do tego zagadnienia w laboratorium dotyczącym ADC i DAC.

Pomiar 1.1 — Napięcie z dzielnika



Podłącz sondę oscyloskopu do GP26, a masę sondy do GND. Ustaw sprzężenie DC.

Zmień oświetlenie fotorezystora i obserwuj zmianę poziomu napięcia. Dla wybranego poziomu oświetlenia porównaj:

- napięcie odczytane z oscyloskopu,
- napięcie obliczone w programie MicroPython.

Nie oczekuj idealnej zgodności. Zastanów się, jakie mogą być źródła różnicy.

6 Zadanie 4: Prezentacja wyników na wskaźniku LED

W poprzednich zadaniach linijka LED była sterowana zmienną ustawianą w programie lub przyciskami. Teraz jej poziom będzie zależał od sygnału analogowego.

Zadanie 3.1 — Wskaźnik oświetlenia



Połącz kod obsługi fotorezystora z funkcją `show_level()` z zadania 1. Napisz program, który:

1. cyklicznie odczytuje wartość z ADC,
2. przelicza ją na poziom od 0 do 8,
3. pokazuje ten poziom na linijce LED.

Przykładowe przeliczenie może mieć postać:

```
1 level = raw * 8 // 65535
```

Sprawdź, czy pełny zakres ośmiu diod jest wykorzystywany w warunkach panujących w sali. Jeśli nie, zaproponuj lepszy sposób przeskalowania wyniku.

Skalowanie pomiaru



Czujnik bardzo rzadko wykorzystuje dokładnie cały teoretyczny zakres ADC. W praktyce często wykonuje się kalibrację i mapuje zakres rzeczywiście obserwowanych wartości na zakres użyteczny w aplikacji.

7 Zadanie 5: Wyświetlacz LCD 16x2

Płytką dydaktyczną zawiera alfanumeryczny wyświetlacz LCD 16x2 zgodny z popularnym kontrolerem HD44780. Wyświetlacz jest podłączany równolegle do mikrokontrolera i w tym kursie będzie używany w trybie 4-bitowym.

W standardowej konfiguracji stanowiska przyjmujemy następujące połączenia:

LCD	Raspberry Pi Pico 2 W
E	GP0
RS	GP1
D4	GP2
D5	GP3
D6	GP4
D7	GP5

Na tych zajęciach nie będziemy implementować niskopoziomowej obsługi kontrolera LCD (zainteresowanych zachęcamy do przeanalizowania biblioteki). Otrzymasz gotowy plik `lcd_hd44780.py`, który zawiera inicjalizację i transmisję danych. Twoim zadaniem jest wykorzystanie biblioteki jako gotowego komponentu.

Konstruktor klasy LCD pozwala jawnie podać numery GPIO, do których podłączone są sygnały wyświetlacza. Dzięki temu już w kodzie programu widać, z których zasobów mikrokontrolera korzysta LCD. Numery podawane w konstruktorze są numerami GPIO (GPx), a nie numerami fizycznych wyprowadzeń modułu Pico.

Zadanie 4.1 — Pierwszy tekst na LCD



Skopiuj dostarczony plik `lcd_hd44780.py` do pamięci Raspberry Pi Pico. W tym celu wybierz z menu edytora *Thonny*: *View* → *Files*. A następnie wyszukaj plik `lcd_hd44780.py` w lewym okienku z katalogami i wybierz symbol trzech kresek a następnie opcję *Upload to*.

Na podstawie tabeli połączeń zainicjalizuj wyświetlacz, podając w konstruktorze numery GPIO:

```
1 from lcd_hd44780 import LCD
2
3 lcd = LCD(rs=1, e=0, d4=2, d5=3, d6=4, d7=5)
4
5 lcd.clear()
6 lcd.write_line("PSM - Lab 1", 0)
```



```
lcd.write_line("Raspberry Pi Pico", 1)
```

Sprawdź, co dzieje się po wpisaniu tekstu dłuższego niż 16 znaków. Zmień program tak, aby na pierwszej linii pojawiał się Twój numer stanowiska, a na drugiej dowolny krótki komunikat.

Zadanie 4.2 — Wynik pomiaru na LCD



Zmodyfikuj program obsługujący fotorezystor tak, aby LCD wyświetlał na bieżąco wynik pomiaru, np.:

ADC: 42150

U: 2.12 V

Nie zapisuj całego programu od początku. Wykorzystaj funkcje utworzone wcześniej i połącz je w jedną aplikację.

8 Zadanie końcowe: Miernik oświetlenia

Na zakończenie połączymy poznane elementy w pierwszy kompletny system pomiarowo-sterujący.

Zadanie 5.1 — Miernik oświetlenia



Zaprojektuj program spełniający następującą specyfikację:

1. program cyklicznie odczytuje sygnał z fotorezystora,
2. na pierwszej linii LCD wyświetlane jest napięcie zmierzone przez ADC,
3. na drugiej linii LCD wyświetlany jest jeden z komunikatów: **CIEMNO**, **NORMALNIE** lub **JASNO**,
4. poziom oświetlenia jest dodatkowo przedstawiany za pomocą linijki LED,
5. wartości progowe pomiędzy trzema stanami dobierz eksperymentalnie do warunków panujących w sali.

Program powinien być podzielony na funkcje. Unikaj powtarzania tych samych fragmentów kodu w kilku miejscach.

Zadanie 5.2 — Automatyczne oświetlenie



Rozbuduj system o dodatkową, zewnętrzną diodę LED pełniącą rolę lampy. Program powinien działać w trzech trybach:

- **AUTO** — dioda jest sterowana na podstawie pomiaru z fotorezystora,
- **ON** — dioda jest zawsze włączona,
- **OFF** — dioda jest zawsze wyłączona.

Wybrany przycisk powinien przełączać kolejno tryby **AUTO** → **ON** → **OFF** → **AUTO**. Aktualny tryb wyświetlaj na LCD.

Na tym etapie zastosuj odpytywanie przycisku w pętli. W Laboratorium 2 wrócimy do tego samego problemu i pokażemy, jak obsługiwać zdarzenia za

 pomocą przerwań.

Rozszerzenie dla chętnych — Histereza

 Jeżeli oświetlenie znajduje się blisko wartości progowej, dioda w trybie AUTO może szybko zmieniać stan. Zastosuj dwa różne progi: jeden do włączania, a drugi do wyłączania lampy.
Wyjaśnij, dlaczego takie rozwiązanie stabilizuje działanie układu.

9 Podsumowanie

Po tych zajęciach powinieneś umieć



- obsłużyć kilka wejść i wyjść GPIO,
- utworzyć prosty dzielnik napięcia,
- odczytać wartość analogową za pomocą ADC,
- przeliczyć wynik ADC na napięcie,
- wykorzystać oscyloskop do weryfikacji napięcia,
- wyświetlić dane na LCD za pomocą dostarczonej biblioteki,
- połączyć czujnik, wyświetlacz, przyciski i diody w jeden program.

Uwaga



Przed opuszczeniem stanowiska zatrzymaj program, wyczyść pamięć (patrz niżej) i uporządkuj przewody oraz elementy używane podczas ćwiczenia.

Wyczyść mikrokontroler w ten sam sposób co na laboratoriach nr 0:

1. Pobierz plik `flash_nuke.uf2` z sekcji *Resetting flash memory* dokumentacji Raspberry Pi:
<https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html#reset-flash-memory>
2. Ustaw przełącznik zasilania używanego modułu Pico w pozycji **OFF**.
3. Naciśnij i przytrzymaj przycisk **BOOTSEL**, a następnie włącz zasilanie modułu przełącznikiem **ON**.
4. Po pojawieniu się dysku USB zwolnij przycisk **BOOTSEL**.
5. Skopiuj plik `flash_nuke.uf2` na wykryty dysk. Mikrokontroler automatycznie uruchomi program czyszczący i wyczyści pamięć flash.
6. **Nie wgrywaj ponownie MicroPythona.** Następna grupa sama go wgra przed rozpoczęciem laboratoriów.