

Graficzne programowanie mikrokontrolerów

M5Stack / UIFlow

Laboratorium 4

Sygnały analogowe – joystick i ADC

Czas trwania: 2 × 45 min**Tryb:** studia stacjonarne**Jednostka:** Katedra Elektrotechniki i Technologii Nadprzewodnikowych**Prowadzący:** dr inż. Michał Łanczont

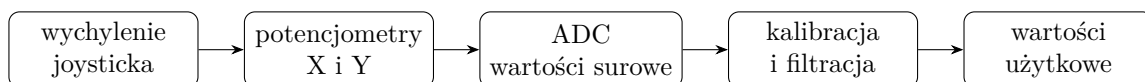
Cel ćwiczenia

Celem ćwiczenia jest poznanie sposobu reprezentacji i przetwarzania sygnałów analogowych na przykładzie joysticka płytki **DEMO BOARD**. W tym laboratorium obie osie joysticka są odczytywane **bezpośrednio przez wewnętrzny przetwornik ADC mikrokontrolera ESP32** z wykorzystaniem wybranych wejść analogowych. Student przeprowadza kalibrację, normalizację zakresu, wprowadza martwą strefę i filtrację oraz wykorzystuje przetworzone wartości do sterowania obiektem graficznym. Zewnętrzny przetwornik ADS1115 z płytki DEMO BOARD zostanie wykorzystany dopiero w Laboratorium 5 podczas omawiania magistrali I2C.

Budowa joysticka

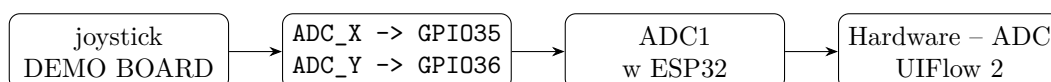
Joystick na płytce **DEMO BOARD** zawiera dwa potencjometry ustawione prostopadle. Wychylenie drążka zmienia napięcie na wyjściach osi X i Y. Są to sygnały analogowe, dlatego przed wykorzystaniem w programie muszą zostać przetworzone na liczby przez przetwornik ADC. Naciśnięcie drążka uruchamia dodatkowy przycisk Z, który jest sygnałem cyfrowym i należy traktować go niezależnie od obu osi analogowych.

Na złączu joysticka płytki sygnały są opisane jako ADC_X, ADC_Y, BTN1 i GND. W tym ćwiczeniu sygnały osi X i Y należy doprowadzić przewodami do **wejść ADC mikrokontrolera**. Zalecane jest wykorzystanie GPIO35 i GPIO36, ponieważ oba piny należą do bloku ADC1 ESP32 i są dostępne na złączu M5-Bus zarówno w Core1, jak i Core2. Przykładowe przypisanie: ADC_X -> GPIO35, ADC_Y -> GPIO36. Masa joysticka musi być połączona z masą mikrokontrolera. Sygnał BTN1 – jeżeli jest wykorzystywany – należy podłączyć do wolnego wejścia cyfrowego zgodnie ze schematem stanowiska.



Wewnętrzny przetwornik ADC mikrokontrolera

W mikrokontrolerze ESP32 wejścia analogowe są obsługiwane przez wbudowane bloki ADC. W UIFlow 2 należy wykorzystać elementy z grupy **Hardware – ADC**, utworzyć osobny obiekt ADC dla każdej osi i wskazać odpowiedni pin. W ćwiczeniu zaleca się stosowanie pinów GPIO35 i GPIO36, należących do ADC1. Unika się dzięki temu ograniczeń ADC2, który w ESP32 współdzieli zasoby z modulem Wi-Fi.



Przetwornik ADC zamienia napięcie wejściowe na liczbę. Wartość surowa zależy od rozdzielczości, tłumienia wejścia, napięcia oraz rzeczywistych parametrów potencjometru joysticka. Nie należy więc

zakładać idealnych wartości skrajnych. Wartości minimum, srodek i maksimum należy wyznaczyć eksperymentalnie dla każdego stanowiska.

W UIFlow 2 funkcja `read()` zwraca surowy wynik ADC, natomiast `read_u16()` zwraca wynik przeskalowany do zakresu 0–65535. W całym ćwiczeniu należy konsekwentnie stosować ten sam rodzaj odczytu dla obu osi.

Typowy tor przetwarzania ma postać:

GPIO ADC → ADC ESP32 → kalibracja → filtracja → normalizacja → martwa strefa →
GUI/sterowanie

Dla każdej osi należy wyznaczyć co najmniej trzy wartości: minimum, środek i maksimum. Ponieważ położenie środkowe nie musi znajdować się dokładnie w połowie zakresu, korzystniejsze jest osobne skalowanie obu połówek charakterystyki:

```
if surowa >= srodek:
    wartosc = mapuj(surowa, srodek, maksimum, 0, 100)
else:
    wartosc = mapuj(surowa, minimum, srodek, -100, 0)
wartosc = ogranicz(wartosc, -100, 100)
```

Takie przeliczenie pozwala uzyskać wartość bliską 0 w położeniu spoczynkowym nawet wtedy, gdy rzeczywiste zakresy po obu stronach środka nie są symetryczne.

Martwa strefa i filtracja

Wartość joysticka w spoczynku może nieznacznie zmieniać się z powodu tolerancji potencjometrów, szumu pomiarowego i ograniczonej rozdzielczości układu. Z tego powodu wokół środka stosuje się **martwą strefę**. Jeżeli wartość znormalizowana znajduje się np. w zakresie $-10 \dots +10$, można przyjąć wartość 0.

```
if abs(wartosc) <= martwa_strefa:
    wartosc = 0
```

Drugim sposobem ograniczenia wahań jest filtracja. W tym ćwiczeniu wystarczy prosta średnia ruchoma z kilku ostatnich próbek. Większa liczba próbek poprawia wygładzenie, ale zwiększa opóźnienie reakcji.

Przebieg ćwiczenia

Tor przetwarzania joysticka

X/Y → GPIO35/36 → ADC ESP32 → kalibracja → filtr → normalizacja → martwa
strefa → sterowanie

Zadanie 1. Odczyt surowych wartości joysticka

Połącz ADC_X joysticka z GPIO35, ADC_Y z GPIO36 oraz wspólną masę. W UIFlow 2 utwórz dwa obiekty **Hardware** – **ADC** i odczytaj obie osie joysticka. Wyświetl ich surowe wartości na LCD. Zanotuj wskazania dla położenia środkowego oraz dla skrajnych wychyleń w lewo, prawo, górę i dół. Jeżeli w dalszej części wykorzystujesz przycisk Z, odczytuj go osobnym wejściem cyfrowym.

Sposób realizacji i weryfikacja

1. Sprawdź, czy oś X jest rzeczywiście odczytywana z GPIO35, a oś Y z GPIO36.
2. Dla każdej osi zanotuj **minimum**, **srodek** i **maksimum**.
3. Sprawdź, czy wzrost wartości odpowiada ruchowi w prawo i w górę; jeżeli nie, zanotuj konieczność odwrócenia znaku danej osi.
4. Odczyt przycisku Z prezentuj oddzielnie od wartości analogowych.

Warunek odbioru: W sprawozdaniu powinny znaleźć się rzeczywiście zmierzone wartości kalibracyjne obu osi.

Zadanie 2. Kalibracja i normalizacja zakresu

Przelicz obie osie na zakres od -100 do $+100$, wykorzystując osobno minimum, srodek i maksimum każdej osi. Położenie spoczynkowe powinno odpowiadać wartości możliwie bliskiej 0.

Sposób realizacji i weryfikacja

1. Zastosuj osobne parametry kalibracji dla X i Y.
2. Dla wartości poniżej środka mapuj zakres **minimum-srodek** na $-100 \dots 0$.
3. Dla wartości powyżej środka mapuj zakres **srodek-maksimum** na $0 \dots 100$.
4. Po przeliczeniu ogranicz wynik do zakresu $-100 \dots 100$.

Zadanie 3. Martwa strefa

Dodaj martwą strefę wokół środka. Porównaj wartości 0, 5, 10 i 15 w skali znormalizowanej i wybierz najmniejszą wartość, dla której po zwolnieniu joysticka obie osie stabilnie wracają do zera.

```
if abs(x) <= martwa_strefa: x = 0
if abs(y) <= martwa_strefa: y = 0
```

Zadanie 4. Filtracja pomiarów

Zaimplementuj średnią ruchomą dla obu osi. Porównaj działanie dla liczby próbek $N = 1$, $N = 4$ i $N = 8$. Obserwuj zarówno poziom wahań w spoczynku, jak i opóźnienie przy szybkim wychyleniu joysticka.

```
bufor.append(probka)
if len(bufor) > N: usun_najstarsza()
filtrowana = sum(bufor) / len(bufor)
```

Weryfikacja

Wybierz jedną wartość N do dalszej części ćwiczenia i krótko uzasadnij wybór: filtr powinien zmniejszać wahania, ale nie może powodować wyraźnie uciążliwego opóźnienia sterowania.

Zadanie 5. Kierunek i intensywność wychylenia

Na podstawie znormalizowanych wartości X i Y określ kierunek wychylenia joysticka. Jeżeli obie osie są równe 0 po zastosowaniu martwej strefy, wyświetl **ŚRODEK**. W przeciwnym przypadku wykorzystaj oś o większej wartości bezwzględnej do rozpoznania kierunku **LEWO**, **PRAWO**, **GÓRA** lub **DÓŁ**. Dodatkowo wyznacz intensywność wychylenia w zakresie $0 \dots 100$.

```
if x == 0 and y == 0:
    kierunek = SRODEK
else if abs(x) >= abs(y):
    if x > 0: kierunek = PRAWO
```

```
    else:        kierunek = LEWO
else:
    if y > 0: kierunek = GORA
    else:      kierunek = DOL

intensywnosc = max(abs(x), abs(y))
```

Zadanie 6. Kursor sterowany joystickiem

Narysuj na ekranie punkt lub mały prostokąt. Położenie albo prędkość ruchu obiektu ma zależeć od obu osi joysticka. Ogranicz współrzędne tak, aby obiekt nie wychodził poza ekran.

Sposób realizacji i weryfikacja

1. Używaj wartości po kalibracji, filtracji i zastosowaniu martwej strefy – nie steruj kursorem bezpośrednio surowym odczytem ADC.
2. Przelicz $-100 \dots 100$ na zmianę współrzędnych kursora albo bezpośrednią pozycję na ekranie.
3. Ogranicz współrzędne do wymiarów ekranu pomniejszonych o rozmiar obiektu.
4. Po zwolnieniu joysticka kursor w trybie prędkościowym nie powinien samoczynnie dryfować.

```
kursor_x += x / skala
kursor_y += y / skala
kursor_x = ogranicz(kursor_x, 0, szerokosc-szerokosc_obiektu)
kursor_y = ogranicz(kursor_y, 0, wysokosc-wysokosc_obiektu)
```

Wymagany rezultat

Joystick dostarcza stabilnych, skalibrowanych wartości X i Y w zakresie $-100 \dots 100$. Program poprawnie wykrywa położenie środkowe, kierunek i intensywność wychylenia, a obiekt graficzny reaguje płynnie i nie dryfuje po zwolnieniu joysticka.

Zadanie końcowe

Integracja zadań etapowych

Zadanie końcowe wykonaj przez połączenie wszystkich etapów toru pomiarowego. Surowe wartości z zadania 1 przechodzą przez kalibrację z zadania 2, filtrację z zadania 4 oraz martwą strefę z zadania 3. Dopiero tak przygotowane wartości mają być użyte do rozpoznawania kierunku i sterowania kursorem.

Zbuduj **graficzny manipulator dwuwymiarowy**. Na ekranie wyświetlaj surowe wartości ADC, wartości po filtracji, wartości znormalizowane X/Y, rozpoznany kierunek oraz intensywność wychylenia. Joystick steruje kursorem. Naciśnięcie przycisku Z przełącza pomiędzy dwoma trybami:

- **POŁOŻENIE** – wychylenie joysticka wyznacza bezpośrednio pozycję kursora względem środka ekranu,
- **PRĘDKOŚĆ** – wychylenie określa kierunek i szybkość przesuwania kursora.

Po zwolnieniu joysticka w trybie **PRĘDKOŚĆ** kursor ma pozostać nieruchomy.

Rozszerzenie: dodaj procedurę ponownej kalibracji środka uruchamianą dłuższym przytrzymaniem przycisku Z.

Kryteria zaliczenia

- poprawne podłączenie osi joysticka do GPIO35 i GPIO36 oraz odczyt przez wewnętrzny ADC ESP32,
- wyznaczenie i zastosowanie parametrów kalibracji,
- poprawna normalizacja zakresu do $-100 \dots 100$,
- zastosowanie martwej strefy i prostej filtracji,
- określenie kierunku i intensywności wychylenia,
- płynne sterowanie obiektem graficznym bez dryfu w położeniu spoczynkowym.

Sprawozdanie

Podstawą zaliczenia ćwiczenia jest sprawozdanie dotyczące wykonania zadania końcowego. Zadania etapowe są zaliczane na zasadzie demonstracji działania programu na stanowisku. Sprawozdanie należy przesłać w formie pliku **.pdf* oraz pliku projektu programu do weryfikacji. W dokumencie należy przedstawić:

- kod programu w UIFlow,
- tabelę wartości kalibracyjnych dla osi X i Y,
- porównanie działania dla wybranych wartości martwej strefy i długości filtru,
- zdjęcia działającego stanowiska lub czytelne zrzuty ekranu,
- krótki opis toru przetwarzania wraz ze schematem blokowym lub pseudokodem,
- opis sposobu testowania programu,
- wnioski.