|  | **AKADEMIA GÓRNICZO-HUTNICZA** |
| --- | --- |

Dokumentacja do projektu

**Biblioteka do obsługi układu ADS1115 za pomocą stm32**

z przedmiotu

**Języki Programowania Obiektowego**

Elektronika i Telekomunikacja, rok III

*Karol Stefanik*

piątek 9:45

prowadzący: Jakub Zimnol

08.01.2025

# Ogólny opis biblioteki

Biblioteka służy do obsługi układu ADS1115. Jest napisana w C++ na mikrokontroler

STM32F303RET6 i na nim była testowana. W przypadku chęci używania jej z innym mikrokontrolerem STM32 można spróbować zaincludować inną odpowiednią dla wybranego mikrokontrolera bibliotekę HAL. Wspomniany include znajduje się w linii 33 w pliku ADS1115.hpp.



1. **Struktura biblioteki**

Klasa główna znajduje się w przestrzeni nazw ks.

# Opis dostępnych metod

**void set\_os(OS os)** - ustawia bit OS.

Dostępne argumenty:

START\_CONV

NO\_EFFECT

**void set\_mux(MUX mux)** - ustawia multiplekser ADS1115

Dostępne argumenty:

A0\_A1 - ustawia AINp = A0, AINn = A1

A0\_A3 - ustawia AINp = A0, AINn = A3

A1\_A3 - ustawia AINp = A1, AINn = A3

A2\_A3 - ustawia AINp = A2, AINn = A3

A0\_GND - ustawia AINp = A0, AINn = GND

A1\_GND - ustawia AINp = A1, AINn = GND

A2\_GND - ustawia AINp = A2, AINn = GND

A3\_GND - ustawia AINp = A3, AINn = GND

**MUX get\_mux()** - zwraca ustawienie multipleksera.

Zwracane wartości:

0 - dla AINp = A0, AINn = A1

1 - dla AINp = A0, AINn = A3

2 - dla AINp = A1, AINn = A3

3 - dla AINp = A2, AINn = A3

4 - dla AINp = A0, AINn = GND

5 - dla AINp = A1, AINn = GND

6 - dla AINp = A2, AINn = GND

7 - dla AINp = A3, AINn = GND

**void set\_gain(GAIN gain)** - ustawia konfigurację wzmacniacza

Dostępne argumenty:

PGA\_6\_144

PGA\_4\_096

PGA\_2\_048

PGA\_1\_024

PGA\_0\_512

PGA\_0\_256

**GAIN get\_gain()** - zwraca konfigurację wzmacniacza

Zwracane wartości:

0 - dla FS = +-6.144 V

1 - dla FS = +-4.096 V

2 - dla FS = +-2.048 V

3 - dla FS = +-1.024 V

4 - dla FS = +-0.512 V

5 - dla FS = +-0.256 V

**void set\_ADS\_mode(ADS\_MODE mode)** - ustawia tryb działania ADS1115

Dostępne argumenty:

CONTINUOUS,

SINGLE\_SHOT

**ADS\_MODE get\_ADS\_mode()**

Zwracane wartości:

0 - dla trybu pojedynczej konwersji

1 - dla trybu ciągłej konwersji

**void set\_data\_rate(DATA\_RATE data\_rate)** - ustawia częstotliwość próbkowania

Dostępne argumenty:

DR\_8

DR\_16

DR\_32

DR\_64

DR\_128

DR\_250

DR\_475

DR\_860

**DATA\_RATE get\_data\_rate()**

Zwracane wartości:

0 - dla f próbkowania = 8 SPS

1 - dla f próbkowania = 16 SPS

2 - dla f próbkowania = 32 SPS

3 - dla f próbkowania = 64 SPS

4 - dla f próbkowania = 128 SPS

5 - dla f próbkowania = 250 SPS

6 - dla f próbkowania = 475 SPS

7 - dla f próbkowania = 860 SPS

**void set\_comp\_mode(COMP\_MODE comp\_mode)**

Dostępne argumenty:

DEFAULT

WINDOW

**COMP\_MODE get\_comp\_mode()**

Zwracane wartości:

0 - dla trybu komparatora klasycznego

1 - dla trybu komparatora okienkowego

**void set\_comp\_polarity(COMP\_POLARITY comp\_polarity)**

Dostępne argumenty:

ACTIVE\_LOW

ACTIVE\_HIGH

**COMP\_POLARITY get\_comp\_polarity()**

Zwracane wartości:

0 - dla aktywowania komparatora stanem niskim

1 - dla aktywowania komparatora stanem wysokim

**void set\_comp\_latch(COMP\_LATCH comp\_latch)**

Dostępne argumenty:

NON\_LATCHING

LATCHING

**COMP\_LATCH get\_comp\_latch()**

Zwracane wartości:

0 - dla nie zatrzaskującego komparatora

1 - dla zatrzaskującego komparatora

**void set\_comp\_que(COMP\_QUE comp\_que)**

Dostępne argumenty:

ASSERT\_AFTER\_1\_CONV

ASSERT\_AFTER\_2\_CONV

ASSERT\_AFTER\_4\_CONV

DISABLE\_COMP

**COMP\_QUE get\_comp\_que()**

Zwracane wartości:

0 - dla ustawienia wartości na ALERT pinie po 1 konwersji

1 - dla ustawienia wartości na ALERT pinie po 2 konwersjach

2 - dla ustawienia wartości na ALERT pinie po 4 konwersjach

3 - dla wyłączenia komparatora

**void upload\_config()** - wysyła konfigurację do ADS1115

**int16\_t get\_conversion()** - zwraca 16 bitową wartość zawartą aktualnie w rejestrze konwersyjnym ADS1115

**set\_low\_threshold(int16\_t threshold)** - ustawia dolną granicę komparatora

Argument - wartość 16 bitowa(wartość = napięcie graniczne/ full scale voltage \* 32768)

**set\_high\_threshold(int16\_t threshold)** - ustawia górną granicę komparatora

Argument - wartość 16 bitowa(wartość = napięcie graniczne/ full scale voltage \*32768)