Politechnika Poznańska
Wydział Automatyki, Robotyki i Elektrotechniki
Systemy mikroprocesorowe

# Projekt
# Sterowanie obrotami silnika za pomocą sterownika PID

*Autor:*

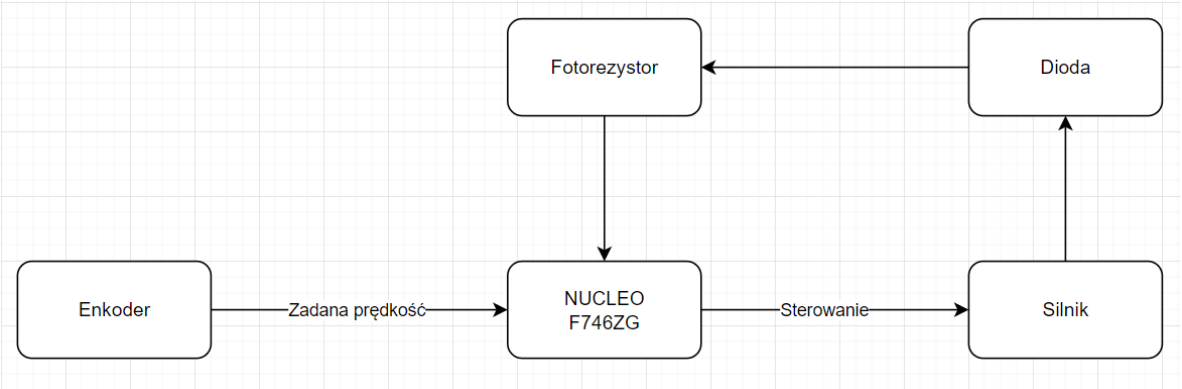
Wiktor Rosiński
144576
Paweł Kwiatkowski
139654

# 1 Opis działania

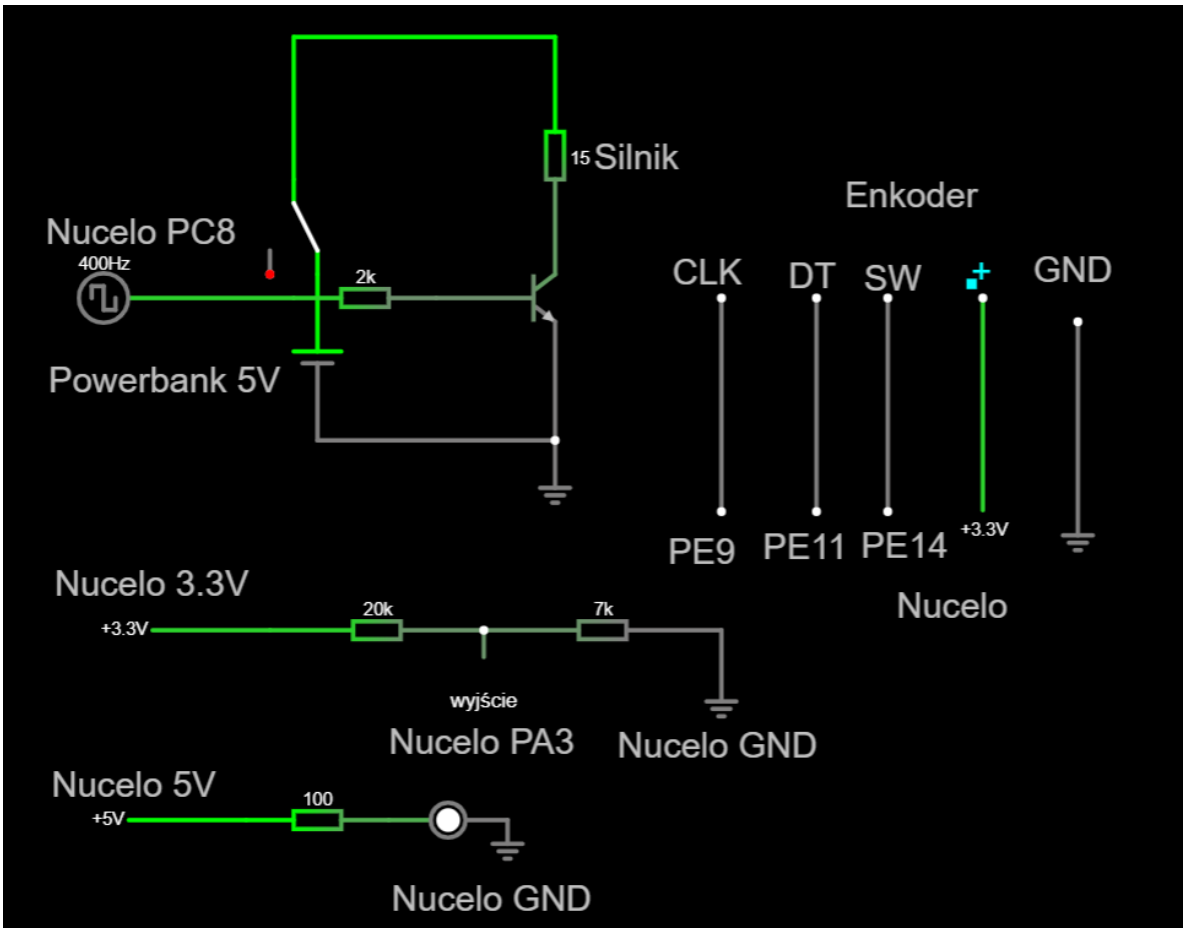
Projekt zakłada sterowanie obrotami silnika po przez sterownik PID realizowanego za pomocą modułu NUCLEO F746ZG.
Wartość prędkości zadanej ustawiamy za pomocą enkodera lub za pośrednictwem programu wykonanego w języku Phyton. Następnie moduł nukleo nastawia sterowanie na silnik by kręcił się z zadaną prędkością.
Pomiar prędkości silnika realizowany jest po przez pomiar wzrostów rezystancji na fotorezystorze spowodowany przysłonięciem diody umieszczonej za silnikiem co jeden obrót. Dodaliśmy również funkcjonalność opserwacji pracy silnika na wykresie w czasie rzeczywistym w programie w języku Phyton.
Silnik wyposażony jest w śmigło, które skonstrułowane jest tak by tylko raz na obrót było w stanie zasłonić diodę. Jest to nasza pętla ujemnego sprzężenia zwrotnego.

# 2 Schemato gólny projektu

Schemat przedstawia logike układu i oddziałowywywanie poszczególnych komponentów na siebie.



## 3 Schemat elektryczny

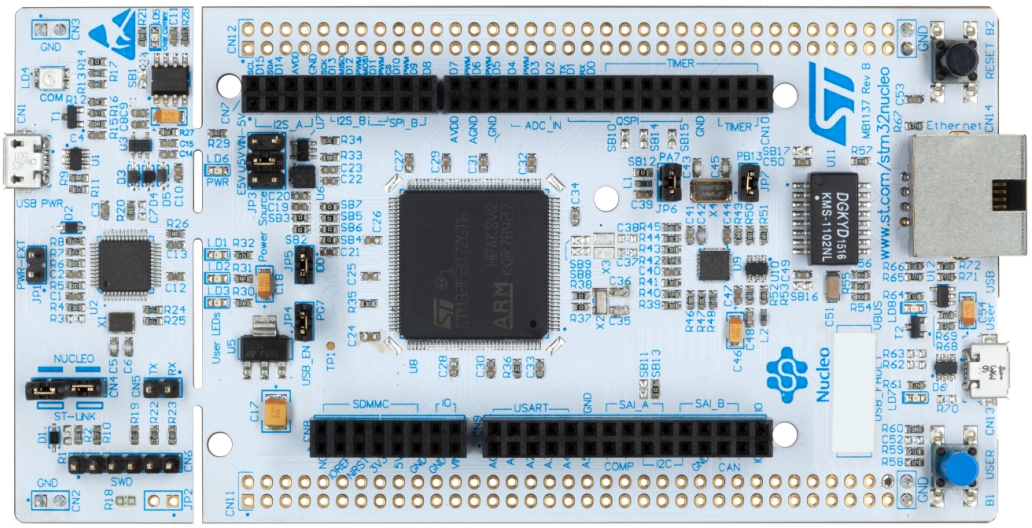



Dodatkowo można pobrać plik txt z naszego githaba i sprawdzić działanie układu w internetowym progeramie falstad.

# 4 Użyte części w projekcie

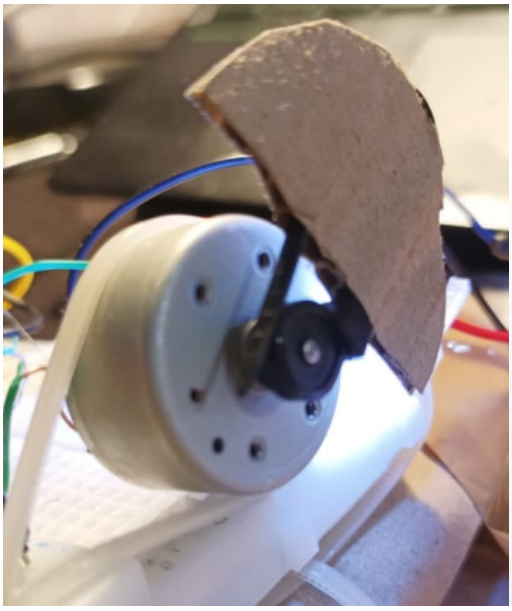
## 4.1 NUCLEO F746 ZG

Modół Nucleo jest sterownikiem jak i komunikatorem pomiędzy wszystkimi częściami naszego układu. Za jego pośrednictwem będziemy odbierać sygnały z poszczególnych części jak i nadawać sygnały sterujące. Służyć też będzie nam jako źródło zasilania. W poniższej tabeli zostały wypisane wykorzystywane piny i ich zastosowanie.



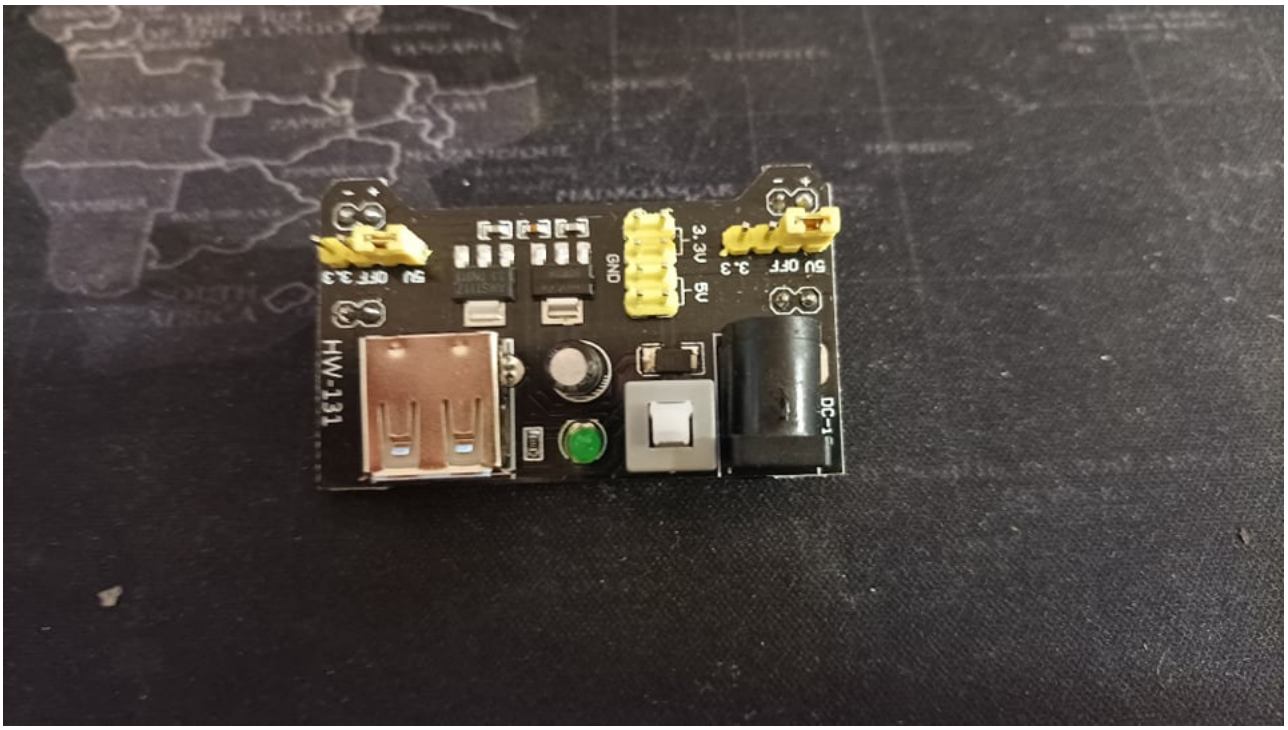

| Nazwa i numer pinu | zastosowanie |
|---|---|
| PC8 | Podajemy napięcie na bazę tranzystora |
| 3V | zasilanie enkodera i fotorezystora |
| 5V | zasila diodę |
| GND | Używane do urziemienia oraz diody |
| PA3 | Sczytuje sygnał z fotorezystora |
| GND | Uziemienie silnika, tranzystora |
| PE9 | Opsługuje wejście CLK enkodera |
| PE11 | Opsługuje wejście DT enkodera |
| PE14 | Opsługuje wejście SW enkodrea |

## 4.2 Silnik




Obiekt regulacji.
Silnik został wymontowany z odtwarzacza CD jego numer seryjny to: 3C146312.
Staraliśmy się znaleźć dokumentacje do niego lub podobny model i jedyne co znaleźliśmy to silnik o numerze seryjnym: rc300-ft-08800. Próbowaliśmy na podstawie dołączonych do niego danych zamodelować nasz silnik jednak nie dawało to współmiernych wyników. Dlatego samodzielnie staraliśmy sie wyznaczyć jego wartości i model.
Wartość rezystancji silnika to 15 Ω w stanie spoczynku.

## 4.3 Moduł zasilający do płytek stykowych MB102

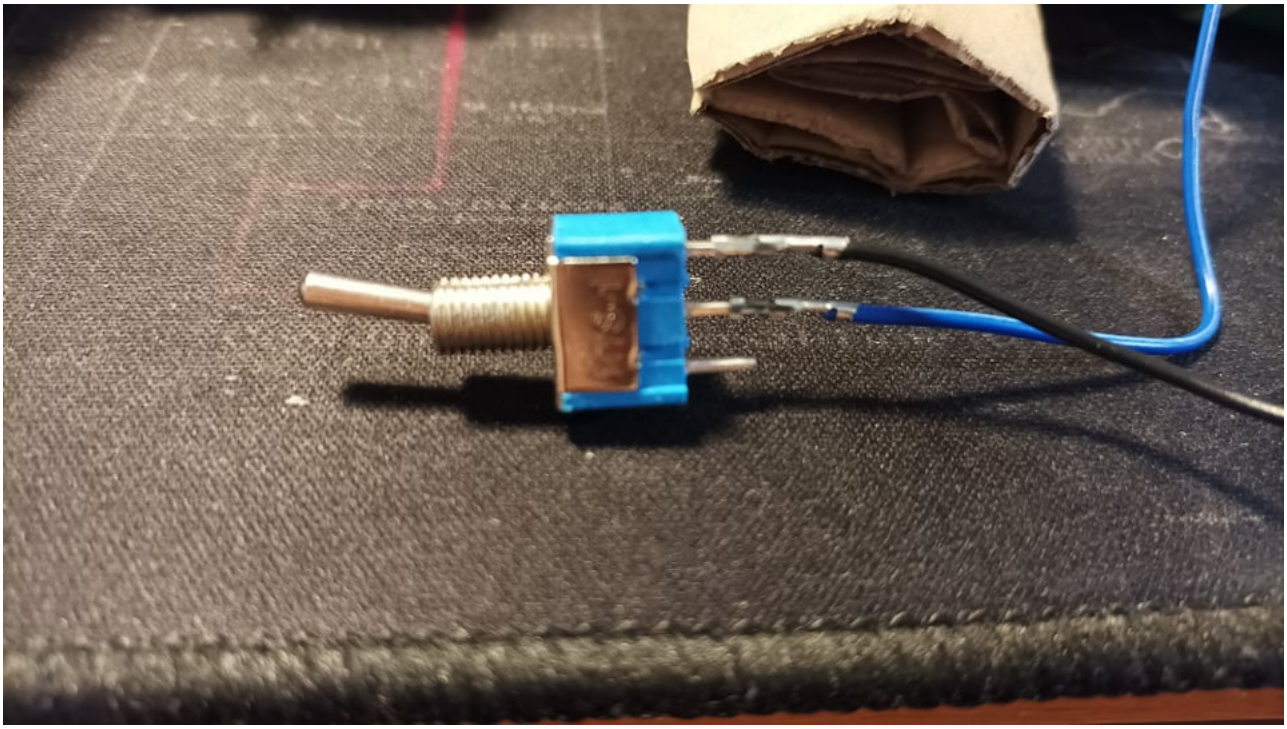



Początkowo silnik zasilany był z baterji 9V jednak z powodu użytkowania z czasem bateria się rozładowywała zdecydowaliśmy się na zamiane bateri na moduł zasilający. Jest on bardziej efektywny i ekonomiczny w przypadku sterowania i ciągłej pracy z układem.

## 4.4 Enkoder z przyciskiem - Iduino SE055


Enkoderem zadajemy prędkość obrotową silnika.

## 4.5 Przełącznik



Używany jako włącznik silnika i wyłącznik. Działa jako zabezpieczenie w przypadku niepożądanego stanu silnika można automatycznie wyłączyć dopływ prądu dochodzącego do silnika.
Używaliśmy go również by sprawdzać działąnie innych części programu bez uruchamiania silnika.

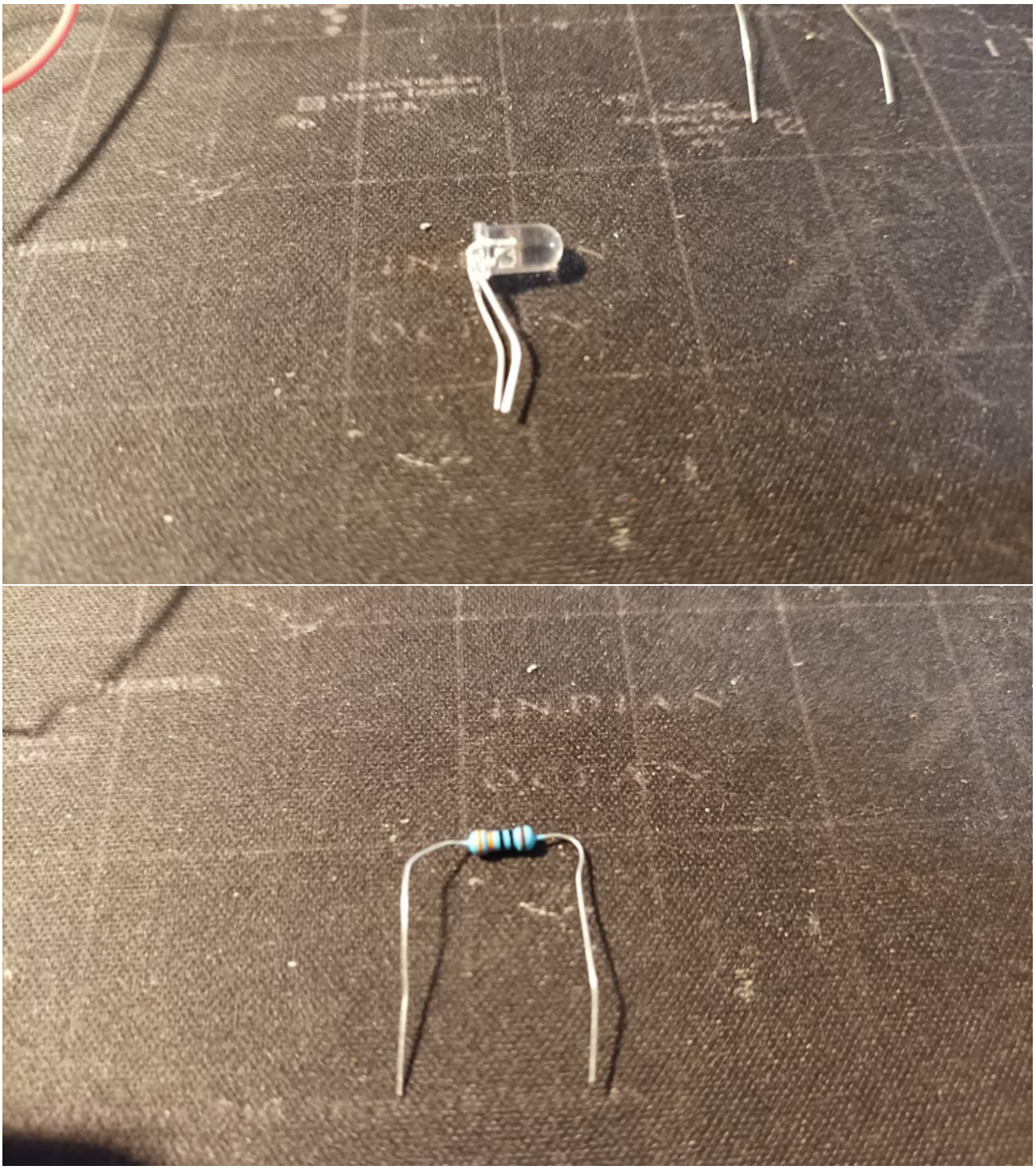
### 4.6 Tranzystor S9013


Używany do sterowania silnikiem.

### 4.7 Fotorezystor GL5528

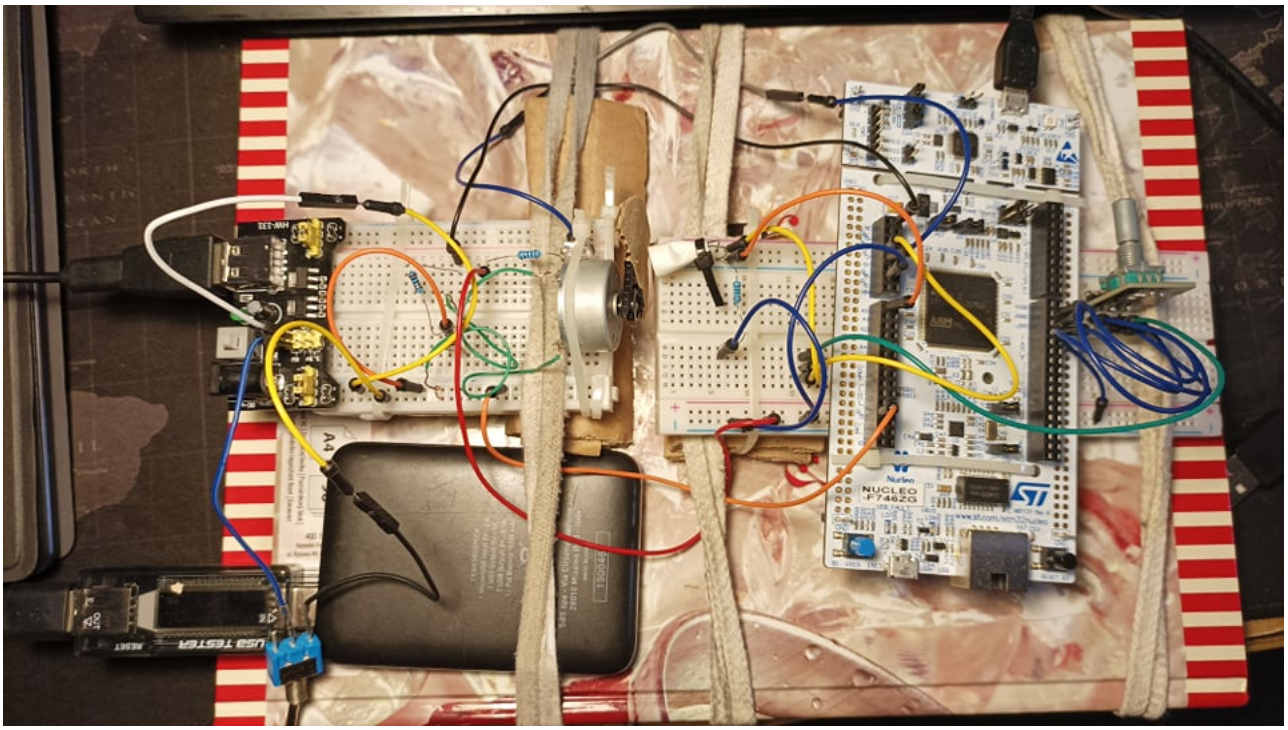
Przy jego pomocy mierzymy zadaną prędkość obrotów. Konkretniej w programie szczytujemy skoki rezystancji w chwili gdy silnik zasłoni diodę śmigłem.
Rozważaiśmy również użycie miernika światła jednak układ działał przez to za wolno z tego względu zdecydowaliśmy się na dzielnik napięcia z fotorezystorem.
Dodatkowo by zredukować szum pomiarowy skupiliśmy pomiar jasności za pomocą przesłony. Zakres rezystancji możliwej do uzyskania fotorezystorze to od 10 000 Ω do 20 000 Ω

### 4.8 Rezystory oraz Diody



Dioda używana do pomiaru prędkości.
Rezystory ustawiają napięcie na
bazie tranzystorze (2 000 Ω),
dzielnika napięć przy fotorezystorze (7 000 Ω)
oraz prąd na diodzie(108 Ω).

### 4.9 Połączony układ



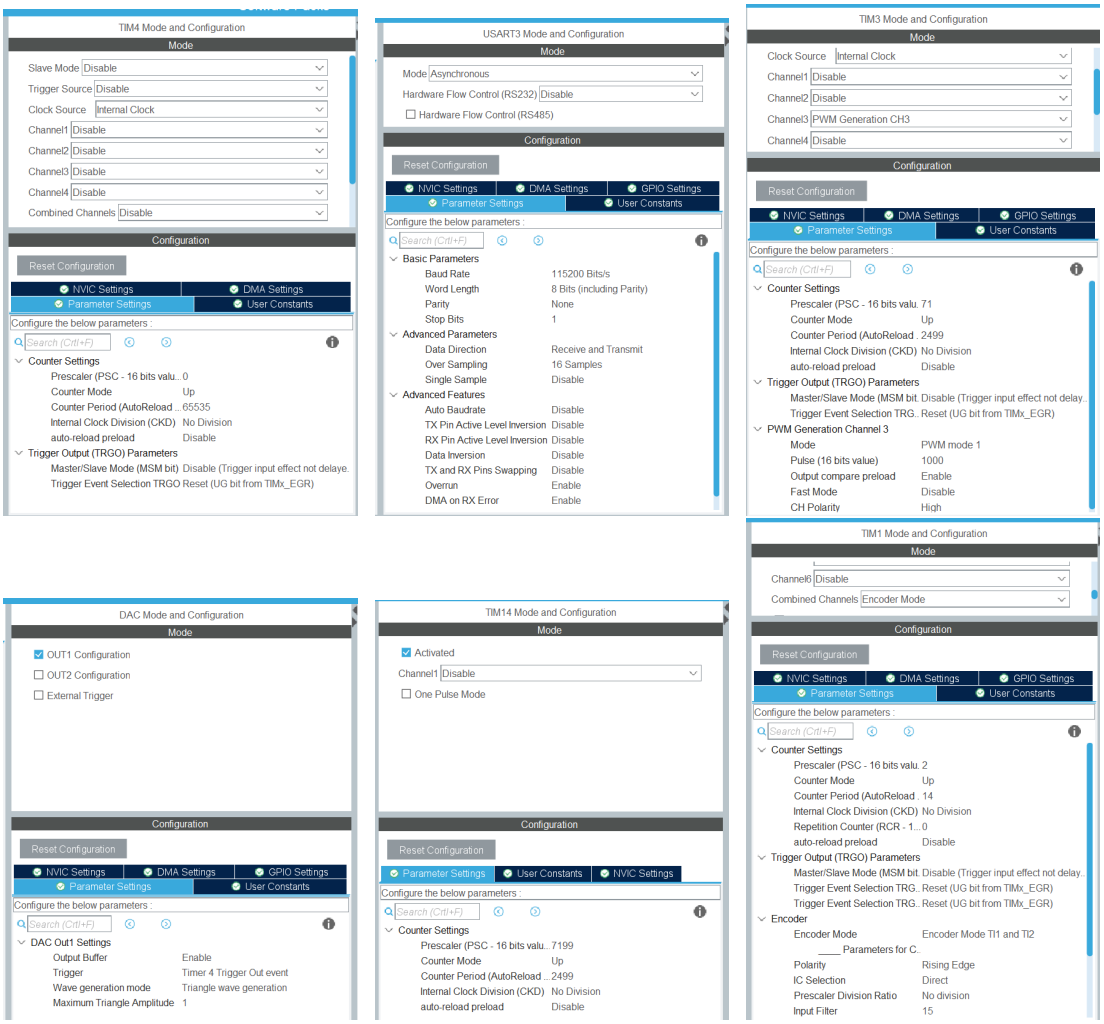

Omówienie układu zprezentacją jego działania można obejrzeć w niniejszym linku.

# 5 Program

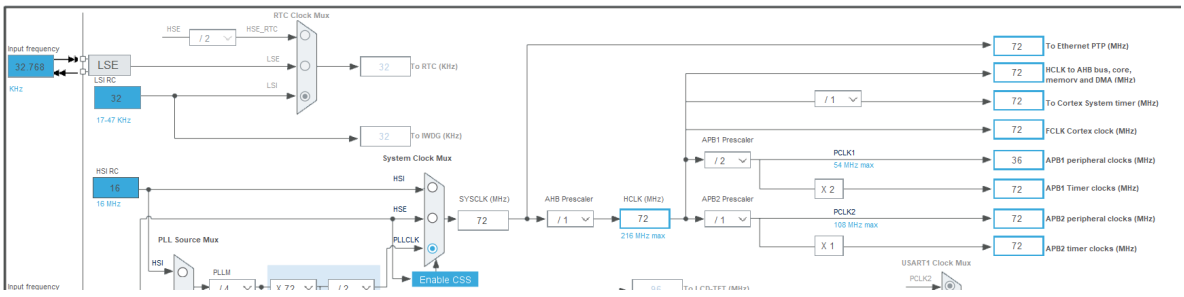
Link do githaba z omawianymi matariały dostempny jest w tym linku.
W poniższych sekcjach omawiamy koncepcje i zastosowanie napisanego kodu. Wszystkie poniższe opisywane funkcjowalności znajdują się tam w znacznie opszerniejszej wersji i po pobraniu można przetestować ich działanie w praktyce (do czego serdecznie zachęcamy).

## 5.1 Program wgrany do modułu NUCLEO

### 5.1.1 Ustawienia programu STM




### 5.1.2 Nastawy zegarów modułu NUCLEO



### 5.1.3 Funkcje

```
/* USER CODE BEGIN 4 */
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{

  if (htim == &htim14 )
  {
        obrotys=(obroty*10+obroty_last)/2;
        e=impulsy-obroty;
        obroty_last=obrotys;
        obroty=0;

    e_sum=e_sump+(e - e_last);
    e_sump=e_sum;
    if(impulsy==0)
    {
        u_next=0;
    }
    else
    {
        u_next =465+( kp*e + ki*e_sum*(dt/2) + kd*(e - e_last)/dt);
    }
    e_last=e;
    if(u_next<0)
    {
        u_next=0;
    }
    else if(u_next>2499)
    {
        u_next=2499;
    }

    u_next=(u_next+u_last1+u_last2+u_last3)/4;



    u_last3=u_last2;
    u_last2=u_last1;
    u_last1=u_next;
         __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, u_next);
    sprintf(msg,"%d %d ", obrotys,impulsy);
        HAL_UART_Transmit(&huart3, (uint8_t*)msg, strlen(msg), 1000);
  }
}

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{

        if (huart == &huart3)
        {
            HAL_UART_Receive_IT(&huart3, key, 2);
            koka=(key[0]-48)*10+(key[1]-48);
            __HAL_TIM_SetCounter(&htim1,koka);

        }
}
/* USER CODE END 4 */
```

Implementacja sterowania PID zaimplementowana została w funkcji void HAL TIM PeriodElapsedCallback(TIM HandleTypeDef *htim). Jak widać zaimplementowany jest tu klasyczny układ regulacji na podstawie podstawowego równania opisującego regupator PID z filtrem FIR by uzyskać dokładniejsze sterowanie.

Funkcja void HAL UART RxCpltCallback(UART HandleTypeDef *huart) służy do odbierania wartości zadanej z programu Phyton

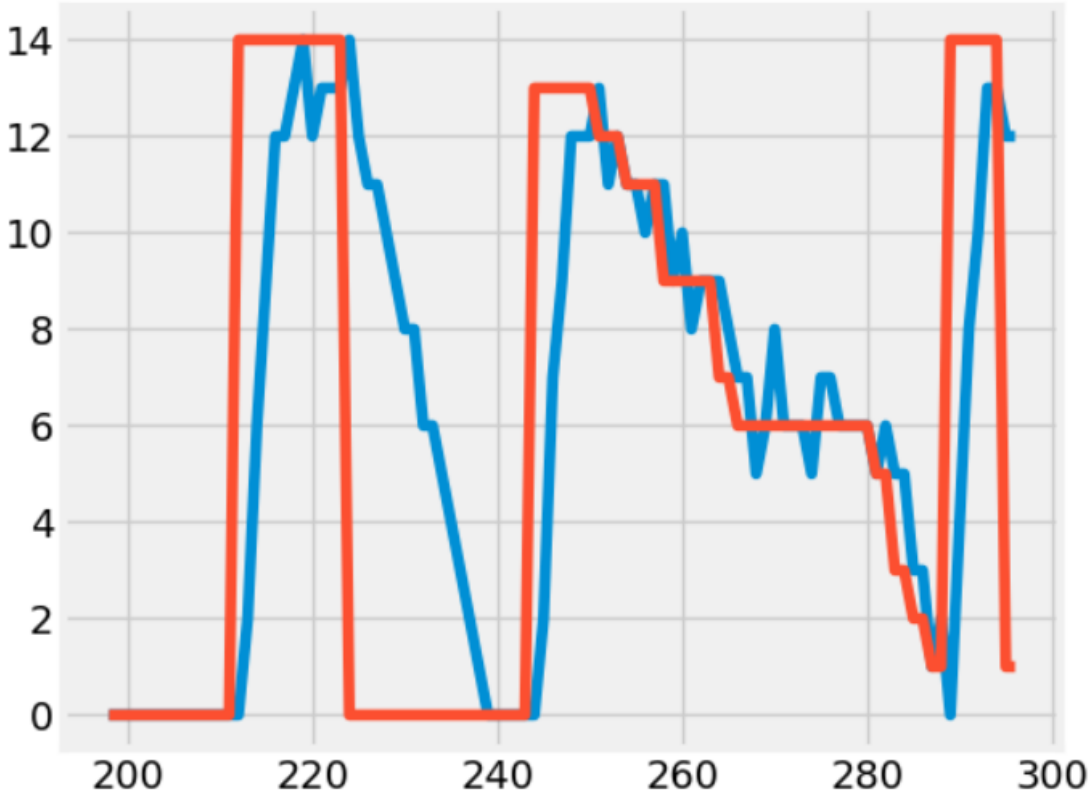
### 5.1.4 Sprawdzenie czy wystąpił obrót

```
159     /* USER CODE BEGIN WHILE */
160     while (1)
161     {
162       /* USER CODE END WHILE */
163
164       /* USER CODE BEGIN 3 */
165           impulsy = __HAL_TIM_GET_COUNTER(&htim1);
166           HAL_ADC_Start(&hadc1);
167           HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
168           AdcValue = HAL_ADC_GetValue(&hadc1);
169           HAL_Delay(1);
170           wyniki[i]=AdcValue;
171           if(wyniki[(i-1)%(rozmiar_dan+1)]!=0)
172           {
173                   for(i2 = 1; i2 < rozmiar_dan; i2++)
174                   {
175                           if(obr==0){
176                                   if((wyniki[abs((i-i2)%(rozmiar_dan+1))]-min_proc1)>wyniki[i])
177                                   {
178                                           okolica=wyniki[abs((i-i2)%(rozmiar_dan+1))];
179                                           obr=1;
180                                           break;
181                                   }
182                           }
183                           else if(obr==1)
184                           {
185                                   if((wyniki[abs((i-i2)%(rozmiar_dan+1))]+min_proc1)<wyniki[i])
186                                   {
187                                           obr=2;
188                                           memset(wyniki, 0, 1000 * sizeof(int));
189                                           break;
190                                   }
191                           }
192                           else
193                           {
194                                   if(okolica<wyniki[i]+200 )
195                                   {
196                                           obr=0;
197                                           i=-1;
198                                           obroty++;
199                                           memset(wyniki, 0, 1000 * sizeof(int));
200                                           break;
201                                   }
202                           }
203                   }
204           }
205           i=(i+1)%(rozmiar_dan+1);
206     }
207     /* USER CODE END 3 */
208   }
```

W whylu zaimplementowaliśmy zlicznnie obrotów silnika na podstawie skoków wartości rezystancji fotorezystora. Dodatkowo dodaliśmy warunki by program drobnych spadków jasności wywołanych ruchem otoczenia nie wpływał na zliczanie obrotów.

## 5.2 Program wyświetlający prędkość obrotową silnika z pomiarów

Program po uruchomieniu sczytuje prędkość obrotową silnika z modułu NUCLEO i wizualizuje dane w postaci wykresu. Wykres na osi y ilość obrotów zaś w osi x czas. Po uruchomieniu programu widoczne są na wykresie oscylacje. Wynikają one z szumu pomiarowego. Jednak nie utrudniają one oceny poprawnego działania układu regulacji.
Za pośrednictwem programu możemy równiesz nastawić wartość zadaną.



# 6 Model w Regulatora

Przugotowaliśmy równiesz prosty model działania naszego układu regulacji. Został on wykonany w programie Simulink. Można go pobrać z załączonego githaba i sprawdzić za jego pośrednictwem czy działanie układu żeczywistego działa zgodnie z założeniami.