
**AKADEMIA GÓRNICZO-HUTNICZA**

Dokumentacja do projektu

# Układ sterujący do inteligentnego domu

Prowadzący:
Dr.inż Roman Rumian

Wykonali:
Rafał Kościej
Michal Kubas

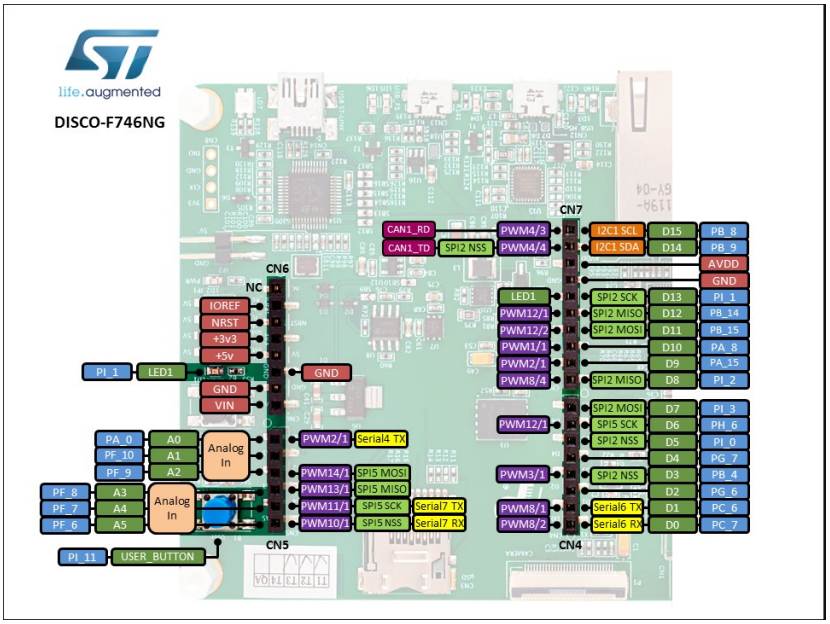
# 1.Opis projektu

Przy tworzeniu projektu przyświecała nam idea stworzenie technologii, która będzie nam ułatwiać codziennie życie. Będzie ułatwiać czynności które nie wymagają wielkich nakładów umysłowych, a jednak ich redukcja, dodałaby nam kilku minut więcej z naszymi bliskimi, co we współczesnym zabieganym świecie jest czymś najbardziej wartościowym. Pomysłem, na który wpadliśmy jest projekt sterownika inteligentnego domu. Nasz sterownik potrafi sterować 2 źródłami światła oraz roletami w pokoju. Do tego dołączyliśmy czujnik temperatury, który na bieżąco wyświetla temperaturę oraz wilgotność. Dostosowywanie światła w zależności od pory dnia, jest pomocne jeszcze w jednym aspekcie, mianowicie, w ochronie środowiska czyli naszej planety. Regulacja na jaką pozwala projekt, może w skali rocznej przynosić zyski dla naszego portfela oraz dla naszych najbliższych w postaci mniej zanieczyszczonego środowiska.

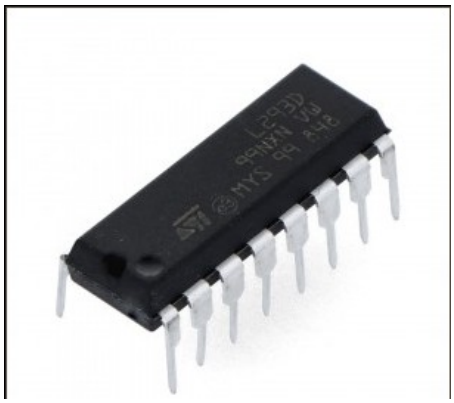
# 2. Urządzenia użyte w projekcie

a) Płytki STM32F746G – Disco – Jest to płytka z rdzeniem ARM Cortex-M7, 340kB pamięci SRAM, 1 MB pamięci flash, Wyświetlacz 4,3 cala LCD-TFT WQVGA 480x272 pikseli z pojemnościowym touch-panelem.

Rozkład pinów:

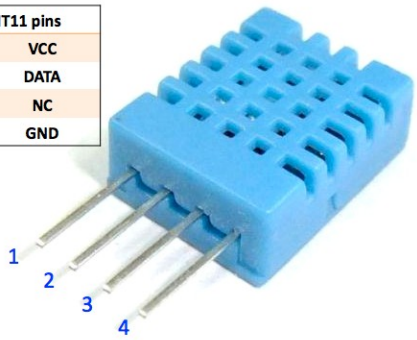


b) mostek L293D – dwukanałowy sterownik silników.



c) silnik DC
d) diody led (imitujące nasze światło np w salonie)
e) czujnik temperatury i wilgotności DHT11

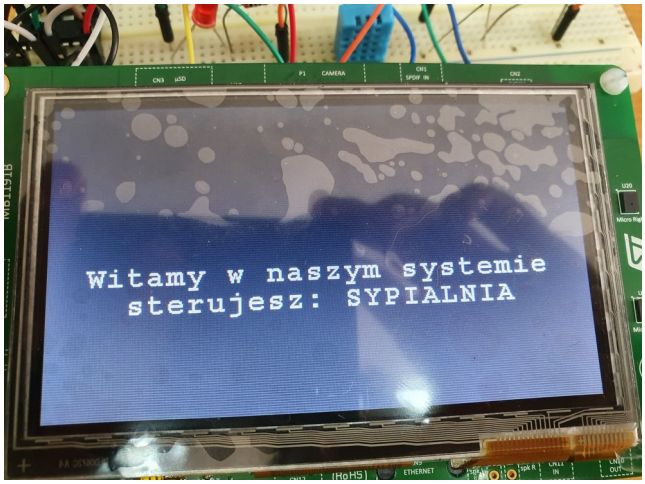
| DHT11 pins | |
|---|---|
| 1 | VCC |
| 2 | DATA |
| 3 | NC |
| 4 | GND |



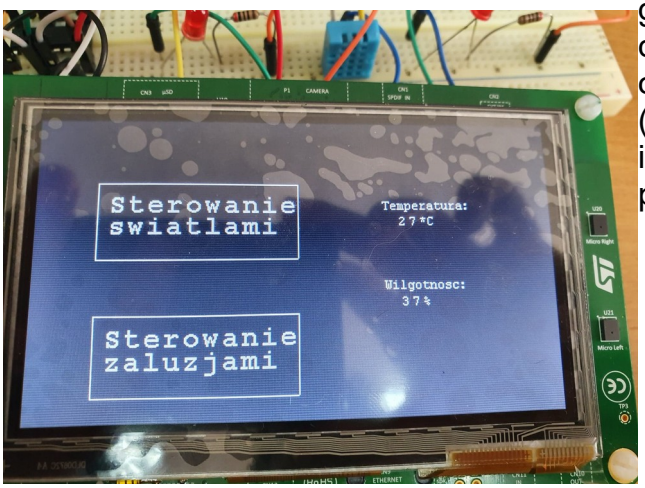

f) przewody, rezystory

# 3. Instrukcja użytkowania

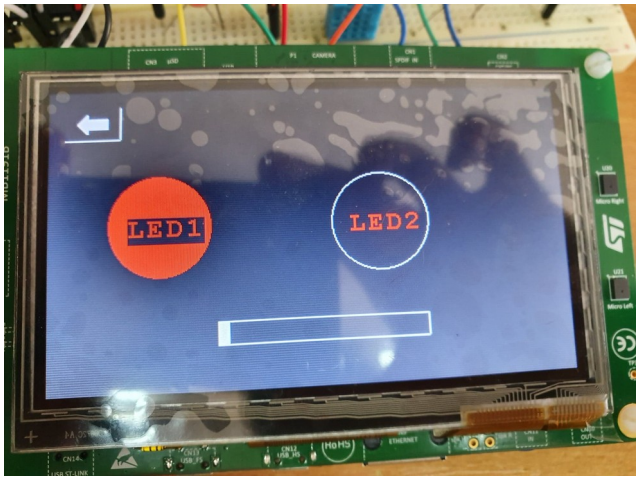
Na samym wstępie wita nas panel sterowania:



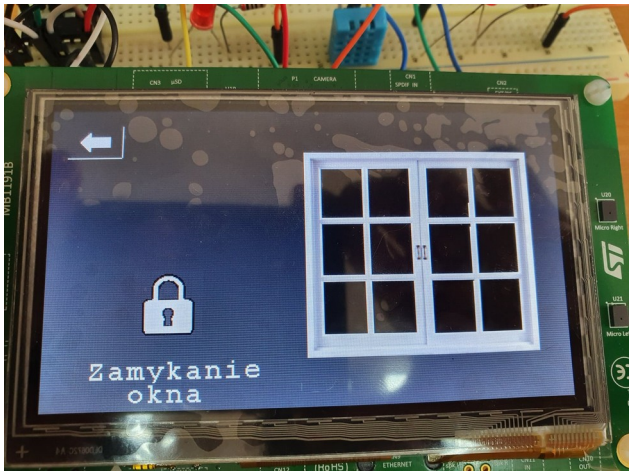

Następnie wyświetlają się dostępne opcje. Możemy sterować oświetleniem lub żaluzjami w naszym domu.W panelu głównym wyświetla się dodatkowa informacja odnośnie temperatury (w stopniach Celsjusza) i wilgotności (w procentach).



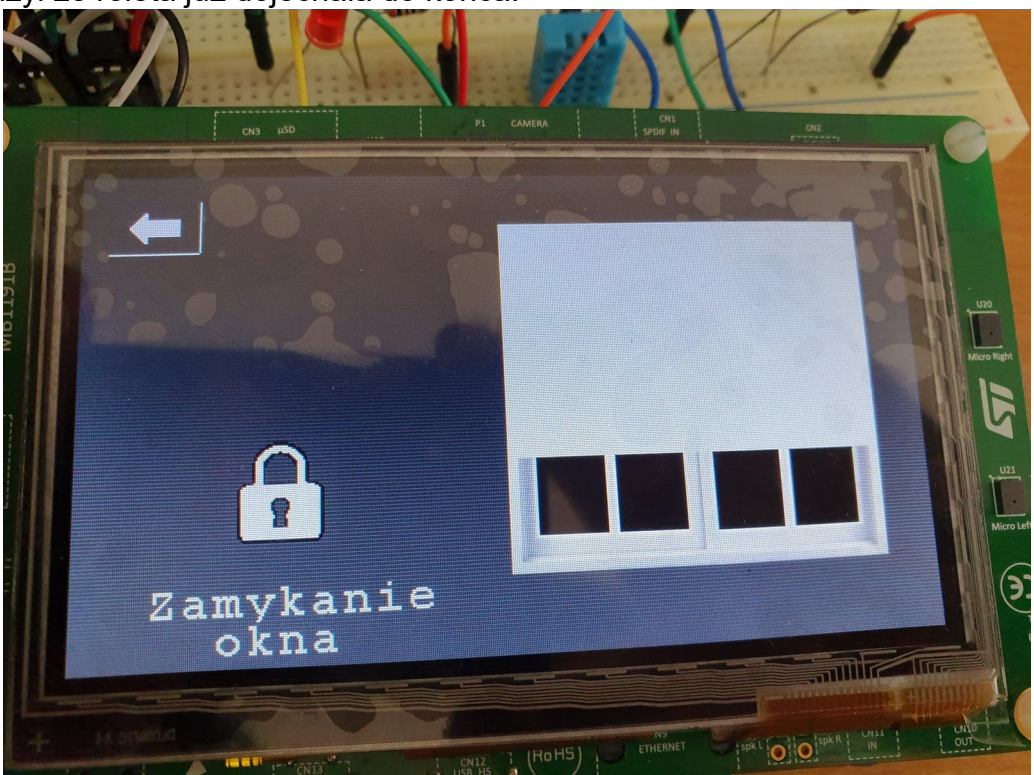

Po wejściu w zakładkę “sterowanie światłem” wyświetlają się nam 2 przyciski, którymi wybieramy którą diodą chcemy sterować. Poziom światła wybranej diody możemy regulować za pomocą slidera. Przy przełączeniu się na drugą diodę, poziom światła na drugiej pozostaje niezmieniony. Diody podłączone są do pinów z możliwością sterowania PWM.



Po powrocie do menu głównego wchodząc w opcję “sterowanie żaluzjami” wyświetla nam się projekt okna. W lewym dolnym rogu mamy kłódkę. Jest ona informacją czy okno jest otwarte lub zamknięte. Jest to zabezpieczeniem przed tym, gdyby użytkownik nie wiedział, czy dane okno jest otwarte czy zamknięte.


W przypadku sterowania roletami używamy interaktywnej rolety wyświetlającej się na oknie która jest specjalnie zaprojektowanym slajderem. Interaktywną roletę możemy przesuwać w góre lub w dół co powoduje włączenie mechanizmu podnoszącego lub opuszczającego rolety. W dogłębnej analizie, przesuwanie slidera powoduje wysłanie sygnału do silnika który kręci się w odpowiednią stronę. Im wyżej lub niżej użytkownik podnosi slider, tym dłużej kręci się silnik. Oczywiście powstaje pytanie, co gdy użytkownik będzie trzymał palec cały czas w jednym miejscu? Zatroszczyliśmy się o to. Mechanizm polega na zliczaniu absolutnej różnicy pomiędzy poprzednim stanem a aktualnym. Przez co trzymanie palca w jednym miejscu przez dłuższą chwilę, nie spowoduje dalszego działania rolety. Jest to pewnego typu zabezpieczenie, gdyby użytkownik zapatrzył się I nie zauważył że roleta już dojechała do końca.



# 4. Opis kodu.

Podczas tworzenia projektu używaliśmy następujących bibliotek:

-stm32hal - wysokopoziomowy interfejs do części sprzętowej mikrokontrolera
-stm32discovery - zestaw bibliotek do naszego modelu płytki
-stm32discoverylcd - zestaw bibliotek odpowiedzialnych za działanie wyświetlacza LCD
-stm32discoveryts - zestaw bibliotek do uruchomienia ekranu dotykowego
-open, closed, lock_open, lock_closed, back - pliki graficzne przekonwertowane do języka C

Do konwertowania grafik z rozszerzeniem .png do kodu w języku C użyliśmy programu LCD image converter wraz z szablonem png_to_C.tmpl (znajduje sie w folderze grafiki).

Funkcje odpowiedzialne za sterowaniem jasnością ledów:

**-void led1**
**-void led2**

Do płynnej zmiany napięcia na wyjściu użyliśmy PWM uruchomionego na timerach TIM1 i TIM2.

Okres odliczania w naszym przypadku to 65535. Mnożymy tą wartość razy zmienną przyjmującą wartości od 0 do 1. Zmienna ta jest różnicą pomiędzy aktualnym położeniem slidera a jego początkiem na podstawie współrzędnych ekranu.

Obsługa czujnika DHT11 wykonywana jest za pomocą funkcji:

**-void sensor_start**
**-void check_response**
**-uint8_t read_data**

Funkcja **delay_us** odpowiada za generowanie opóźnienia w mikrosekundach. Używamy tego do obliczania czasu trwania stanów generowanych przez czujnik DHT11 opisanych niżej. Funkcja była niezbędna gdyż **Hal_delay** nie obsługuje przedrostków czasu mniejszych niż mili.

Zasada działania czujnika DHT11 wygląda następująco:

aby zainicjować czujnik, musimy najpierw ustawić linię danych w stan niski na około 18 ms. - **sensor_start**

Po tym DHT11 pozostanie w stanie niskim przez kolejne 80 us, a następnie przełączy się w stan wysoki dla 80 us. Po wykonaniu tej czynności czujnik zostanie zainicjowany i rozpocznie transmisję. - **check_response**

Teraz DHT11 wyśle 40 bitów danych. Transmisja każdego bitu rozpoczyna się od poziomu niskiego napięcia, który trwa 50us, długość następnego sygnału wysokiego określa, czy bit ma wartość „1”, czy „0”.
Jeśli długość poziomu wysokiego napięcia wynosi około 26-28 us, bit ma wartość „0”.
A jeśli długość wynosi około 70 us, to bit = „1”. - **read_data**

40 bitów wysłanych przez DHT11 ma następujące dane: 8-bitowe całkowe dane Wilgotności + 8-bitowe dziesiętne dane Wilgotności + 8-bitowe całkowe dane Temperatury + 8-bitowe dziesiętne dane Temperatury + 8-bitowa suma kontrolna
Jeśli transmisja danych jest prawidłowa, suma kontrolna powinna być ostatnim 8 bitem sumy 8-bitowych całkowych danych Wilgotności + 8-bitowych dziesiętnych danych Wilgotności + 8-bitowych całkowych danych Temperatury + 8 bitowych dziesiętnych danych Temperatury”.

Funkcja main() zawiera linie odpowiedzialna za inicjalizację peryferiów, uruchomienie timerów, uruchomienie i określenie bloku pamięci dla ekranu lcd. Ponadto niezbędne funkcje i zmienne do wyświetlania odpowiednich plansz na wyświetlaczu.

**void DisplayButton1 -** odpowiada za “wypełnienie” przycisku leda kolorem odpowiadającym jego barwie
**void welcome -** ekran powitalny
**void DrawHome -** ekran podstawowy z menu wyboru światła/zasłony oraz wskazaniem czujnika
**void DrawBlinds -** ekran wyświetlający okno sterowania zasłonami
**void DrawLed -** ekran wyświetlający okno sterowania światłami
**void sensor -** odpowiada za uruchomienie i pobranie danych z czujnika DHT11 oraz wyświetlanie ich na wyświetlaczu - na ekranie domowym jako wyższy layout.

W pętli while uruchamiamy ekran powitalny i czekamy na wydarzenie wykrycia dotyku następnie uruchamia się pobranie danych z czujnika oraz sprawdzenie czy miejsce dotknięcia odpowiada miejscom przycisków. Jeśli dotkniemy w obszarze znajdowania się któregoś z wirtualnych przełączników następuje zmiana ekranu i narysowanie tego odpowiadającego elementom, którymi chcemy sterować. Na każdym menu znajdują się podobne funkcje wykrywania dotyku i miejsca dotknięcia.

# 5. Problemy

Przy każdym projekcie człowiek natrafia na wiele problemów, jednak to właśnie problemy uczą nas najwięcej, albo chociaż pokazują nam drogę którą nie powinniśmy iść. Jednym z podstawowych problemów było integracja panelu dotykowego z pinami wyjściowymi, tak aby nasze gesty wykonywane na panelu były odwzorowane na fizyczny świat sterowania.

Kolejnym problem, dość istotnym z puktu widzenia programistów, był problemy z kompilacją kodu. Częste błędy i ostrzeżenia przyniosły nam dużo teoretycznej jak I praktycznej wiedzy na temat kompilacji języka C, czy też  o tym, w jaki sposób program jest wgrywany do naszego mikrokontrolera I później wykonywany. Mimo żmudnego procesu, uważamy, że był to dobry grunt do zdobywania wiedzy o systemach wbudowanych I architekturze ARM.

# 6. Dalszy rozwój

W naszym planie I zamyśle rozwoju chcemy dodać więcej funkcjonalności, sterowanie większą ilością światła, dodanie większej ilości silników, aby można było sterować większą ilością urządzeń w domu. Chcielibyśmy także zaimplementować różnego rodzaju sensory (np: zbliżeniowe) do lepszej automatyzacji naszych procesów. Celem końcowym jaki sobie postawiliśmy, będzie dodanie sensora wykrywającego gesty, aby za pomocą gestów można było sterować naszym domem.

Kolejnym pomysłem jest też wykorzystanie złącza Ethernet, aby stworzyć sieć, z którą będziemy mogli się łączyć I sterować naszymi urządzeniami z każdego miejsca w domu oraz spoza niego.

Stworzenie menu, aby było przyjemniejsze dla użytkownika oraz bardziej intuicyjne będzie jednym z naszych głównych priorytetów w najbliższym czasie.

Ostatnim etapem będzie także poprawa bezpieczeństwa, to znaczy, zaimplementowanie rozwiązań kryptograficznych ( jak np klucze symetryczne AES lub rodzina szyfrów hashujących SHA ), dzięki którym niepowołane osoby nie dostaną się do sterowania naszym domem.

# 7. Podsumowanie

Projekt pozwolił nam lepiej zaznajomić się z tematyką mikrokontrolerów, szczególnie z rodziną ARM. Ciągle szkolenie, szukanie informacji pozwoliło nam lepiej poznać samą platformę jak I metody programowania tych niepozornie małych urządzeń w których drzemie wielka siła. Chcielibyśmy w przyszłości lepiej rozwinąć nasz pomysł oraz poświęcić tej tematyce więcej czasu, aby dalej ją zgłębiać I kiedyś, po wielu próbach otrzymać dzieło, przy którym będziemy mogli powiedzieć, że udało się zaspokoić nasze nieposkromione ambicje. Elektronika po raz kolejny dała nam lekcję pokory, ale czujemy, że dała nam również  pewien kredyt zaufania, który będziemy chcieli jak najszybciej wykorzystać.