Poliechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki
Raport końcowy z Systemów Mikroprocesorowych

PLANTIE^{TM}

Autorzy:
KAMIL CHOIŃSKI
OSKAR STABLA

17 stycznia 2020

Spis treści

1 Ogólny opis projektu

1.1 Cel i zakres projektu

Nasz projekt ma na celu pomoc zabieganym ludziom, którzy nie mają czasu na zajmowanie się swoją ukochaną roślinką przez swój częsty brak pobytu w domu. Wystarczy dostęp do internetu, nic więcej.

Nasza koncepcja opiera się na systemie zdalnego zarządzania rośliną. Chcemy mierzyć parametry gleby i otoczenia takie jak wilgotność, nasłonecznienie. W zależności od odczytanych wartości przez płytkę rozwojową UNO połączoną z modułem ESP8266 będzie możliwe sterowanie pompką wody. Ponieważ koncepcja projeku nie jest rozległa postanowiliśmy utworzyć panel sterowania na stronie internetowej.

1.2 Kosztorys

Większość komponentów projektu została sprowadzona z chin, ze względu na ich niską cenę. Niestety czas oczekiwania na nie wydłużył się i spowodował nieoczekiwane opóźnienia w harmonogramie.

Przedmiot	Cena
Płytka rozwojowa UNO	15 PLN
Moduł ESP8266	18 PLN
Czujnik wilgotności i temperatury powietrza	5 PLN
Czujnik wilgotności gleby	5 PLN
Czujnik nasłonecznienia	5 PLN
Czujnik poziomu wody	5 PLN
Przekaźnik	5 PLN
Pompa wodna	22 PLN
Bateria 9V	2 PLN
Materiał na obudowę	5 PLN
Przewody, płytka uniwersalna, cyna, klej na gorąco	10 PLN
Suma	97 PLN

Tablica 1: Zakupione produkty

1.3 Opis podobnych rozwiązań

1.3.1 Trutina

'Trutina' z Gremon Systems pozwala na automatyczne podlewanie danej rośliny, a także na pomiary parametrów za pomocą sensorów wbudowanych w urządzenie takich jak: pyranometer, E-box, GS3 które wysyłając pomiary, kontroluje system i wybiera kiedy podlać roślinę.

System posiada także aplikację na telefon, dzięki której możemy podglądać graficznie mierzone parametry, a także kontrolować system podlewający.

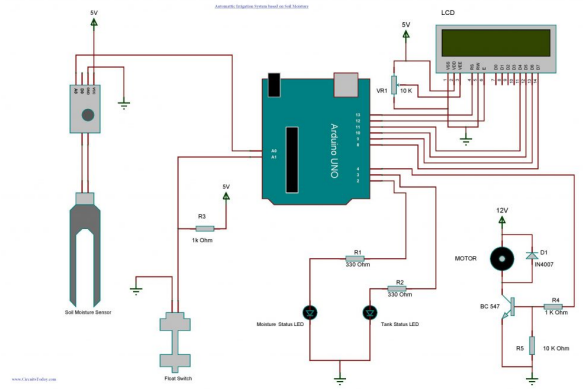
Różni sie od naszego projektu aplikacją, gdyż my otrzymujemy wszystkie pomiary na stronę internetową i samoistnym wyborem kiedy będzie włączone podlewanie. Jest to także dużo większy projekt od naszego gdyż jest przystosowany do pracy w warunkach terenowych, a nasz do bardziej domowych warunków.

Rysunek 1: Trutina [1]

1.3.2 Arduino Irrigator System

Ten projekt bazuje na płytce rozwojowej arduino UNO i ma na celu automatyzacje procesu pomiarów parametrów rośliny i jej podlewaniu. System czyta wilgotność gleby i włącza pompę wodną jeśli wilgotność spadnie poniżej pewnego poziomu. Kiedy system wykryje wartość powyżej ustalonej wyłączy pompę. System posiada także wyświetlacz LCD 16x2 na którym są wyświetlane: poziom wody w zbiorniku, status czy pompka jest włączona, wilgotność gleby.

Projekt różni się od naszego sposobem wyświetlania danych - różnica wyświetlaczy, a także brakiem zapisywania pomiarów i połączeniem z siecią Wi-Fi.



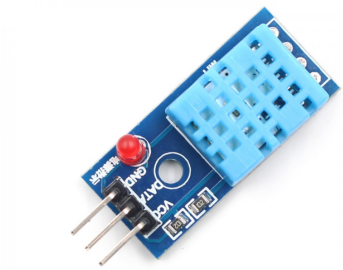
Rysunek 2: Arduino Irrigator System [2]

2 Szczegółowy opis projektu

2.1 Rozwiązania techniczne - Przegląd modułów projektu

2.1.1 Czujnik wilgotności i temperatury powietrza: DHT11

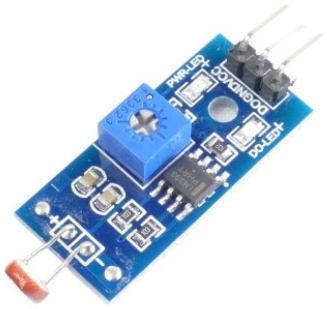
Pomiary wilgotności i temperatury powietrza są wykonywane z użyciem czujnika DHT11. Do implementacji w ArduinoIDE została wykorzystana biblioteka DHT-sensor-library, aby pomiary wykonywały się poprawnie, dzięki czemu ręczna kalibracja nie była potrzebna.



Rysunek 3: Czujnik DHT11

2.1.2 Czujnik natężenia światła

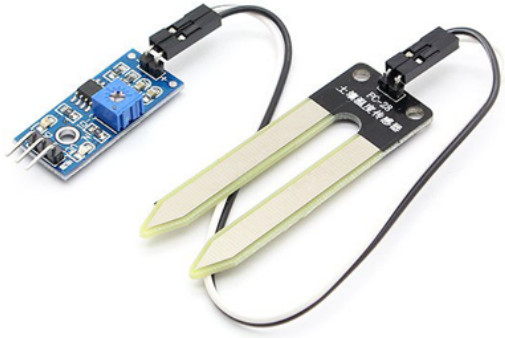
Pomiary natężenia aktualnego oświetlenia w otoczeniu rośliny są wykonywane z użyciem czujnika natężenia światła. Jego parametry były skonfigurowane na podstawie własnych doświadczeń: 0 procent - czujnik zakryty ciemnym materiałem, 100 procent - czujnik oświecony mocnym źródłem światła.



Rysunek 4: Czujnik natężenia światła

2.1.3 Czujnik wilgotności gleby FC-28

Pomiary wilgotności gleby naszej rośliny są wykonywane z użyciem czujnika wilgotności gleby FC-28. Jego parametry były skonfigurowane na podstawie własnych doświadczeń: 0 procent - czujnik w powietrzu - brak powierzchni styku, 100 procent - czujnik zanurzony w wodzie.



Rysunek 5: Czujnik FC-28

2.1.4 Czujnik poziomu wody w zbiorniku

Pomiary aktualnego poziomu wody w zbiorniku z wodą do podlewania są wykonywane z użyciem czujnika poziomu wody. Jego parametry były skonfigurowane na podstawie własnych doświadczeń: 0 procent - czujnik suchy, nie zanurzony w wodzie, 100 procent - czujnik w całości zanurzony w wodzie.

Rysunek 6: Czujnik poziomu wody

2.1.5 Wyświetlacz OLED

Zewnętrzne wyświetlanie pomiarów wilgotności gleby i powietrza jest obsługiwane przez wyświetlacz OLED zamontowany na przodzie obudowy, aby była możliwość monitorowania jej stanu, gdy przebywamy w jej otoczeniu bez potrzeby połączenia internetowego. Do poprawnego wyświetlania danych na ekranie zostały wykorzystane biblioteki AdafruitGFX.h i AdafruitSSD1306.h.

Rysunek 7: 0.96 calowy wyświetlacz OLED

2.1.6 Przekaźnik, Pompka wodna

Do układu została zamontowana samochodowa pompka do spyskiwaczy firmy TOPRAN, gdyż było to tańsze rozwiązanie niż inne rodzaje pompek. Jej napięcie znamionowe wynosi 12 [V], jednak w projekcie do jej zasilania używamy baterii 9 [V], gdyż po testach była to wystarczająca moc zasilania do naszych potrzeb. Jej okresowe zasilanie jest realizowane za pomocą przekaźnika do którego z jednej strony wpięta jest bateria i pompka, a z drugiej zasilanie 5 [V] z płytki rozwojowej UNO zasilające sam przekaźnik, GND i sygnał sterujący.

(a) Przekaźnik [8]

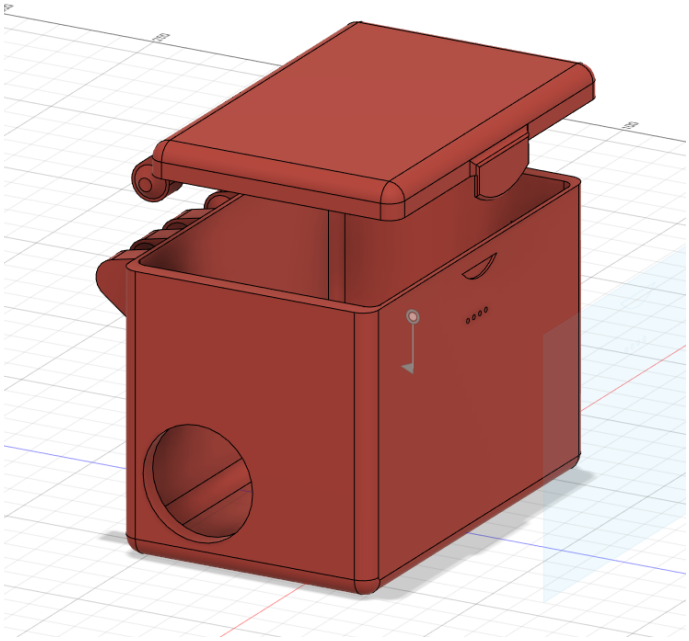
(b) Pompa wodna [9]

(c) Bateria 9V [10]

Rysunek 8: Komponenty pompki

2.1.7 Obudowa

Obudowa na komponenty naszego projektu została zaprojektowana w programie Fusion360. Model ten został "pocięty" (przygotowany do druku) przy użyciu programu Cura i wydrukowany na drukarce 3D Ultimaker 2+. Został użyty materiał do drukowania o nazwie PLA (Polilaktyd) i dysza drukująca wielkości 0.8mm, co jak na standardy drukowania 3D jest dosyć dużo jednak nie była wymagana duża dokładność. Prędkość drukowania elementów była ustawiona na 40 [mm/s] i chłodzenie na 100 [%] Czas drukowania pokrywki i pudełka wyniósł 12 godzin, nie licząc nieudanych druków i czasu spędzonego na dostrajanie drukarki.



Rysunek 9: Model obudowy (Fusion360)

2.2 Szczególne rozwiązania warstwy programowej

2.2.1 Płytka rozwojowa UNO

Realizacja oczekiwania płytki rozwojowej UNO na sygnał sterujący z ESP8266 została wykonana poprzez ciągłe sprawdzanie czy mamy coś w jej buforze. Gdy wykryjemy że jest coś do odczyania zczytujemy ten znak. Dane które są odbierane będą zawarte w wskaźnikach początku "<" i końca ">" danej komendy, dzięki temu możemy wykryć kiedy zaczyna się nowa instrukcja i wyczyścić pozostałości jakie mogły pozostać w buforze, a także wykryć zakończenie danej instrukcji. Zapobiega to "gubieniu danych"i otrzymywaniu powielonych instrukcji.

Część kodu płytki rozwojowej UNO odbierająca dane

```
while (Serial.available())
{
   char serialChar = Serial.read();
   if (serialChar == '<')
   {
      serialMessage = "";
      serialMessage += serialChar;
   }
   else if (serialChar == '>')
   {
      stringComplete = true;
      serialMessage += serialChar;
   }
   else
   {
      if ((serialChar >= 48 && serialChar <= 122) || serialChar == ',')
      serialMessage += serialChar;
   }}
   if( stringComplete == true)
```

Rysunek 10: Płytka rozwojowa UNO

2.2.2 ESP8266

Połączenie płytki ESP8266 z Internetem jest realizowane z pomocą biblioteki ESP8266WiFi.h która ułatwia nam połączenie poprzez dodanie klasy WiFiClient, która przyjmując argumenty nazwy sieci Wi-Fi i hasła łączy się automatycznie z internetem. Gdy już uzyskamy połączenie z siecią Wi-Fi łączymy się z naszym serwerem na wcześniej zadeklarowanym IP i porcie za pomocą metody stworzonej z użyciem biblioteki WebSocketClient.h. Aby ułatwić testowanie po każdym nieudanym połaczeniu restertujemy ESP, aby próbowało cały czas się połączyć.

Tak jak w płytce rozwojowej UNO, odbieranie i wysyłanie danych jest realizowane we wskaźnikach początku "<" i końca ">".

Rysunek 11: ESP8266

2.2.3 Panel sterowania przez Internet

3 Realizacja projektu

3.1 Problematyka projektu

Przed rozpoczęciem prac nad projektem utworzyliśmy tabele w programie internetowym Trello, na których zapisaliśmy rzeczy potrzebne do zrobienia, a także podział kto jaką cześcią projektu się zajmie. Najtrudniejszą częścią wydawała nam się implementacja połączeń pomiędzy płyką rozwojową UNO, ESP8266, a serwerem, więc postanowiliśmy zacząć od tego, a dopiero później przejść do następnych aspektów projektu. Wyzwaniem dla nas była również współpraca nad tak rozległym projektem, gdyż większość prac jakie do tej pory wykonywaliśmy nie byly takie rozległe.

3.2 Sposób wykonania

Postanowiliśmy połączyć płytkę rozwojową UNO obsługującą pompkę i monitorującą wartości naszych czujników z ESP8266, aby była możliwość sterowania poprzez Internet.

Płytka rozwojowa UNO cały czas czeka na przychodzące dane które będą zawarte w wskaźnikach początku "<" i końca ">" danej komendy, aby wykonać określoną czynność lub odesłać pomiary wykonane przez czujnik wilgotności i temperatury do ESP8266, które jest połączone z UNO poprzez piny RX i TX. UNO odsyła swoje komendy również w wskaźnikach początku i końca, aby zapobiec "gubieniu" danych.

ESP8266 jest naszym tzw. "mostem" i w każdym momencie czeka na dane czy to z UNO czy ze strony internetowej i w zależności od kogo dane otrzymuje przerzuca je do danego odbiorcy. Program na ESP jest zrealizowany z użyciem bibliotek ESP8266WiFi, WebSocketClient, SoftwareSerial, które kolejno służą do: umożliwieniu połączenia się ESP do sieci WiFi zadeklarowanej w naszym kodzie, połączeniu się poprzez websocket jako klient do naszego serwera postawionego na laptopie, połączenia się poprzez wyprowadzenie RX i TX do płytki rozwojowej UNO.

Serwer do którego łączy się ESP8266 działa na laptopie. Połączenie jest realizowane z użyciem websocketów. Dzięki takiemu rozwiązaniu możemy wpisać adres na którym działa nasz serwer do przegladarki i zobaczyć stronę, dzięki której możemy kontrolować nasze urządzenie. Kontrola odbywa się za pomocą przycisków na tej stronie, które po naciśnięciu wysyłają daną komendę do podłączonych klientów - w tym przypadku naszego ESP8266. Dane pomiarowe przychodzące do serwera są zapisywane w stworzonej na potrzeby projektu bazie danych i przedstawiane graficznie w postaci wykresów na stronie.

3.3 Problemy napotkane podczas realizacji

3.3.1 Połączenie między UNO, a ESP8266

Próbowaliśmy różnych typów połączeń jak I2C, albo SPI jednak poprawność testowych danych jakie otrzymywaliśmy była nikła - albo dostawaliśmy tylko cześć danych albo były one w różny sposób powielone. Rozwiazaliśmy ten problem poprzez użycie wskaźników początku "<" i końca ">" przy wysyłaniu danej komendy, aby wykonać określoną czynność lub odesłać pomiary.

3.3.2 Projektowanie i drukowanie obudowy

Dużym problemem była niewiedza związana z wielkością jaką powinna mieć obudowa, jednak zdecydowaliśmy że będzie ona większa niż będziemy potrzebować, aby umożliwić nam łatwe rozbudowanie projektu w przyszłości.

Rysunek 12: Nieudany druk pokrywki

Przy początkach drukowania popełniliśmy błąd złej początkowej wysokości druku co spowodowało że model był drukowany w powietrzu a co za tym idzie druk się nie powiódł. Kalibracja drukarki 3D naprawiła nasz problem i druk został wykonany poprawnie

4 Harmonogram i podział obowiązków

4.1 Podział obowiązków

Kamil zajął się xxxxxx Oskar zajął się xxxxxxx

Budowa projektu na płytce stykowej, testowanie osobnych czujników i modułów projektu.

Implementacja połączeń między płytką rozwojową, a ESP8266.

Stworzenie serwera i panelu sterowania.

Baza danych i graficzne ich przedstawienie.

Implementacja połączeń między ESP8266, a serwerem.

Projekt i druk obudowy.

Przeniesienie projektu na płytkę uniwersalną i włożenie do obudowy.

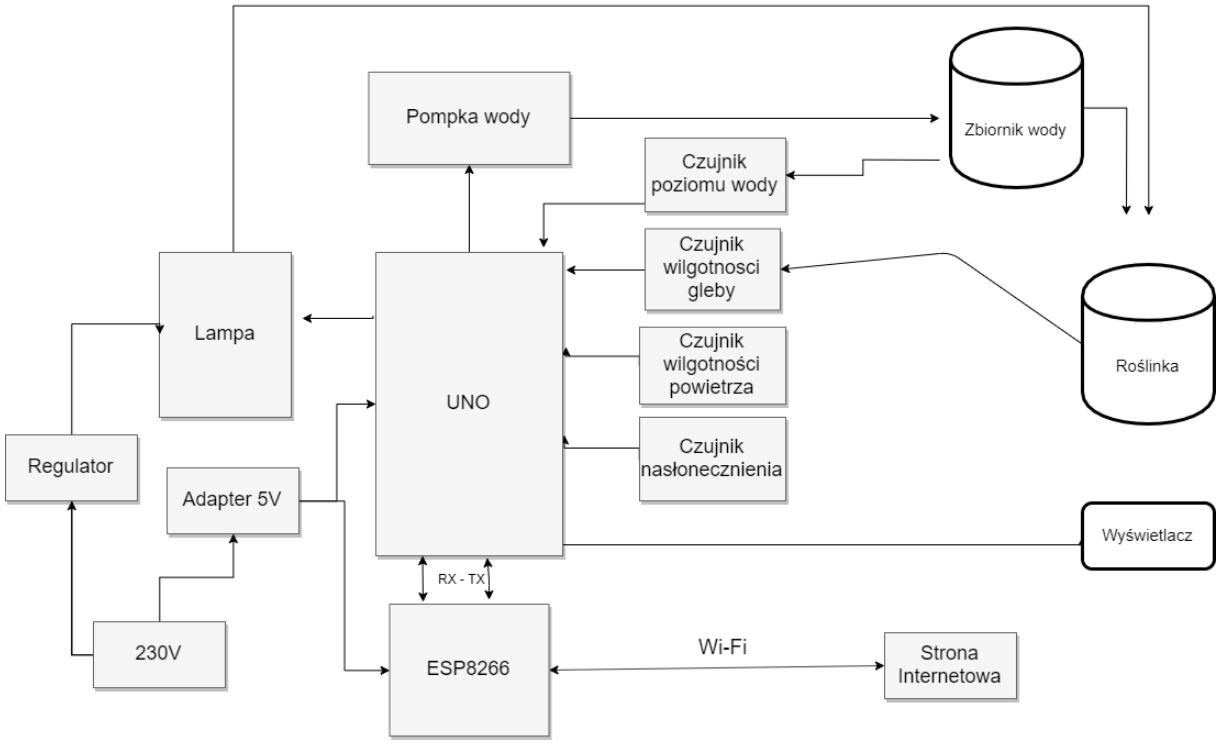
Testowanie

x

x

4.2 Wstępny schemat blokowy

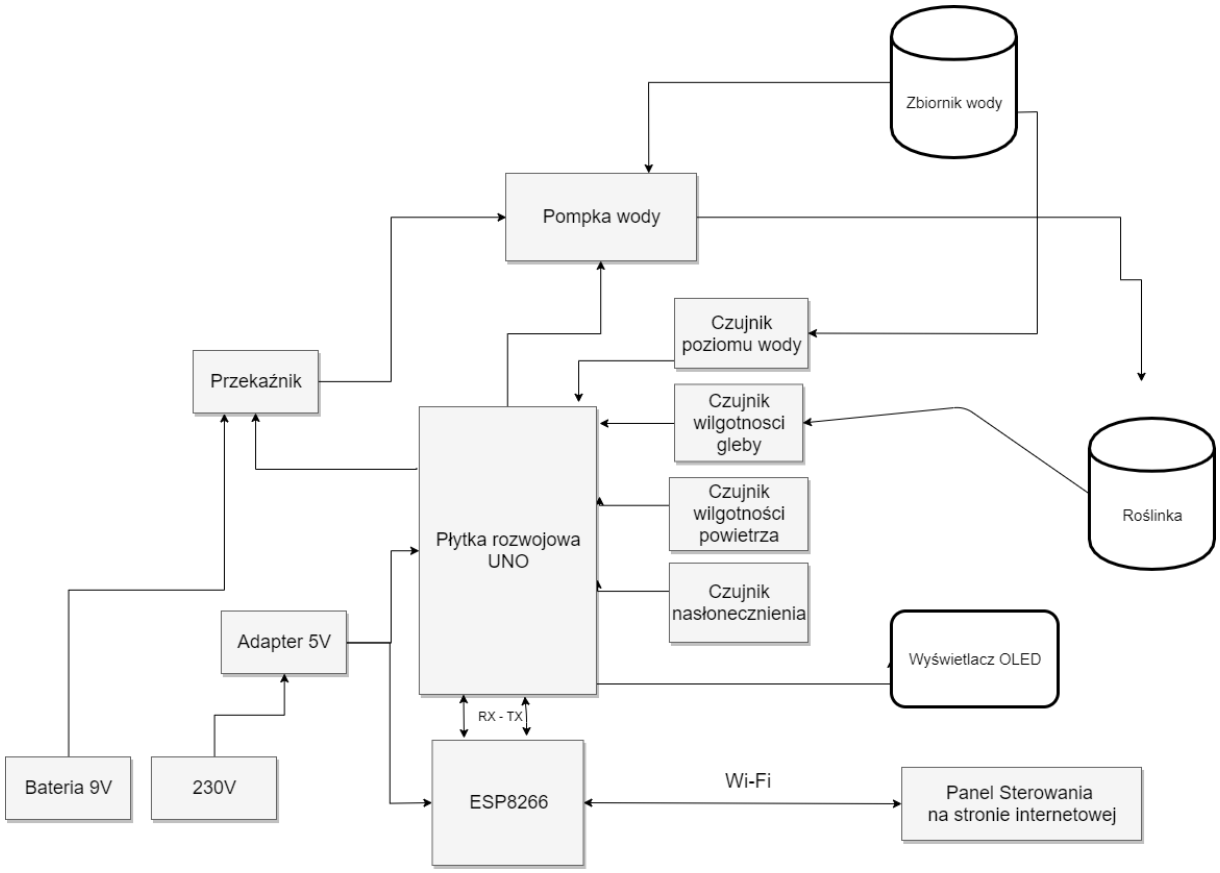
Przed rozpoczęciem pracy nad projektem stworzyliśmy schemat za pomocą programu draw.io, którego staraliśmy się trzymać jednak ze względu na pewne komplikacje w trakcie realizacji pewne aspekty zostały zmienione co zostało przedstawione na finalnym schemacie blokowym.



Rysunek 13: Wstępny schemat blokowy

4.3 Końcowy schemat blokowy

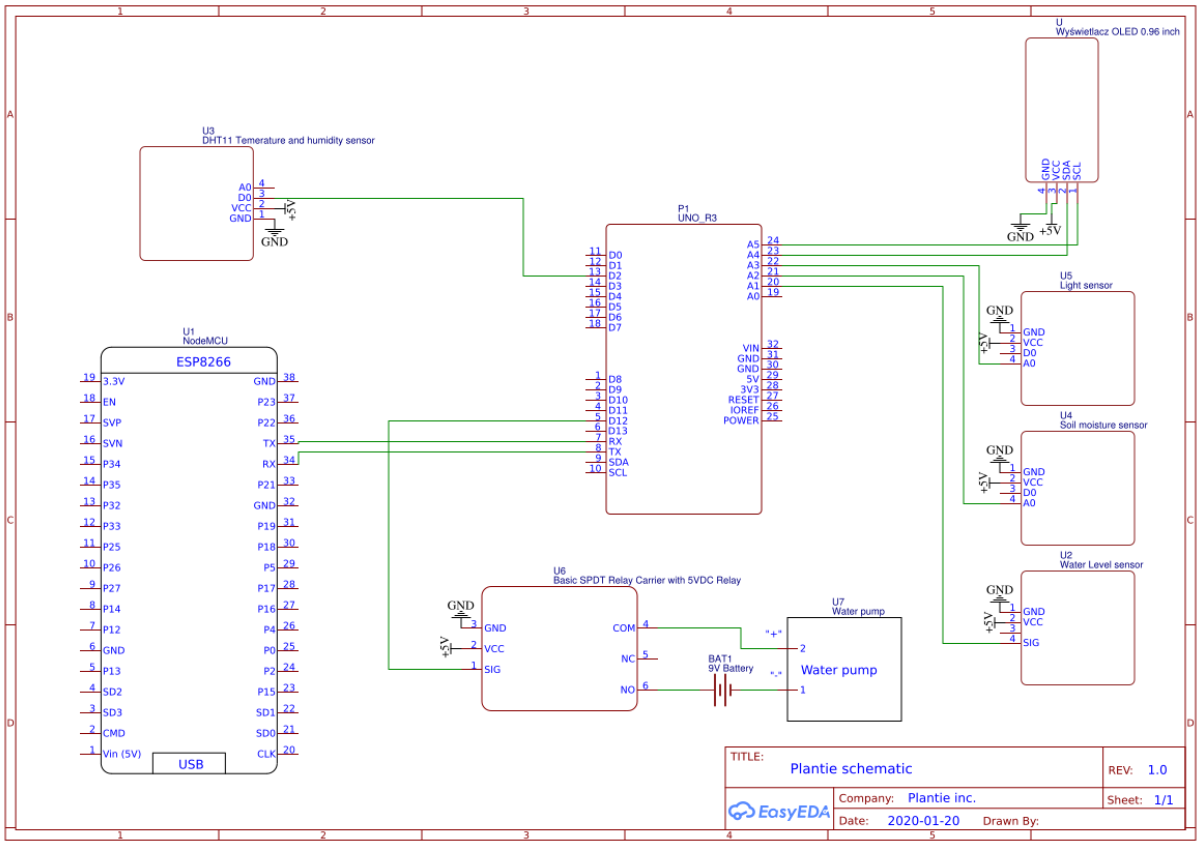
Można zaobserwować że nasz schemat zmienił się jednak jego zmiany nie są duże, a koncepcja projektu została zachowana. Zmiany zostały wprowadzone w trakcie budowy projektu ze względu na problemy z implementacją oświetlenia, a także poprzez postanowienie zasilania pompki wodnej z użyciem baterii 9V, gdyż zasilanie płytki rozwojowej UNO nie było wystarczające do dostatecznie dużego poboru wody.



Rysunek 14: Końcowy schemat blokowy

4.4 Schemat elektryczny

Z użyciem programu EasyEDA został utworzony schemat elektryczny projektu.



Rysunek 15: Końcowy schemat blokowy

4.5 Harmonogram

4.5.1 4.11

Przeanalizowanie schematów płytki rozwojowej UNO, modułu komunikacyjnego ESP8266, czujnika wilgotności gleby, czujnika wilgotności powietrza, czujnika nasłonecznienia oraz rozwiązanie techniczne doświetlania rośliny. Stworzenie schematu elektrycznego gotowego projektu. Testowanie poprawności działania posiadanych czujników w warunkach domowych. Implementacja komunikacji z modułem ESP8266. Dopasowywanie czasów działania. Dokupienie brakujących komponentów gotowego projektu.

4.5.2 25.11

Realizacja połączeń elektrycznych na płytce stykowej i ostateczne testowanie poprawności dzia- łania. Przeniesienie projektu na płytkę uniwersalną. Przygotowanie ewentualnej obudowy i realiza- cja montażu systemu doświetlania. Stworzenie prezentacji na zajęcia

4.5.3 16.12

Wstępna prezentacja gotowego projektu i ocena błędów.

4.5.4 20.01

Ostateczna prezentacja z poprawką błędów.

4.6 Zgodność z harmonogramem

Okazało się że założony harmonogram był zbyt optymistyczny, założyliśmy za dużo rzeczy do zrobienia w początkowej części harmonogramu, a nie ustaliliśmy dużo na końcową część. Jednak dzięki temu mieliśmy czas na dokończenie zaplanowanych aspektów projektu i nie przekroczeniu danego czasu na ukończenie projektu.

4.6.1 4.11

Rozpoczęliśmy prace nad naszym projektem. Postanowiliśmy skorzystać z aplikacji internetowej Trello do utworzenia tabel, które pozwoliły by nam na lepsze zarządzanie podziałem prac nad projektem.

Przeanalizowaliśmy schematy płytki rozwojowej UNO, modułu komunikacyjnego ESP8266, czujnika wilgotności gleby, czujnika wilgotności powietrza oraz czujnika nasłonecznienia. Przeanalizowaliśmy działanie każdego z czujników poprzez wykonanie przykładowych kodów na płytce rozwojowej UNO. Dowiedzieliśmy się jakich komponentów projektu nam brakuje i zamówiliśmy je w wybranych sklepach. Stworzyliśmy wstępny schemat blokowy projektu, aby ukazać działanie poszczególnych elementów zawartych w naszym projekcie. Na ten moment postanowiliśmy nie tworzyć schematu elektrycznego, gdyż nie wiedzieliśmy jak dokładnie zrealizujemy poszczególne połączenia.

Ze względu na obszerność testowania i planowanie realizacji planu działania na nadchodzące tygodnie nie zajęliśmy się komunikacją z modułem ESP8266, ani problemem związanym z doświetlaniem naszej rośliny.

4.6.2 25.11

Do tego dnia zajmowaliśmy się komunikacją między płytką rozwojową UNO, ESP8266, a stroną internetową. Napotkaliśmy wiele problemów co nieplanowanie wydłużyło nam pracę na tym etapie projektu. Udało nam się uzyskać stabilne połączenie między naszymi urządzeniami. Zrealizowaliśmy połączenia UNO, ESP8266, a także podłączyliśmy do układu czujnik wilgotności i temperatury powietrza i na płytce stykowej po czym przetestowaliśmy poprawność działania.

Przeniesienie projektu na płytkę uniwersalną okazało się niepraktyczne na tym etapie projektu, gdyż nie próbowaliśmy jeszcze połączyć ze sobą wszystkich modułów projektu i finalny kod do wgrania na UNO i ESP8266 nie był gotowy. Z powodu niewiedzy jak dokładnie będzie wyglądało nasze urządzenie na płytce uniwersalnej nie przygotowaliśmy planowanej obudowy. Nie zrealizowaliśmy również systemu doświetlania.

4.6.3 16.12

Podłączyliśmy ze sobą wszystkie czujniki na płytce uniwersalnej i testowaliśmy poprawność działania. Zajęliśmy się projektowaniem obudowy w programie Fusion360, abyśmy mogli ją wydrukować na drukarce 3D i włożyć do obudowy komponenty naszego projektu. Wydrukowaliśmy obudowę i wsadziliśmy tam wszystkie moduły projektu.

4.6.4 20.01

Stworzyliśmy techniczno-reklamową prezentację projektu, a także rozległy raport końcowy z projektu.

5 Podsumowanie

5.1 Osiągnięte cele

To our surprise, the project in general was more time consuming that we could ever expect. The lack of knowledge makes it hard to even get one, because we did not really know what to look for. Thankfully, the Internet nowadays provides lots of resources and tutorials that helped us in comming up with solutions and finding out why they do not work. In conclusion we are proud of the outcome, not everything went as expected but we managed to combine software and hardware solutions and make them work together. The project forced us to think outside of the box as there were neither imposed technologies nor the way to write our code. It was good practise to come up with something of our own, develop that one idea and bring it to the end.

5.2 Pomysły na rozwój projektu

Pomysłów na dalszy rozwój projektu jest wiele, ponieważ w trakcie prac przychodziło nam do głowy ich dużo jednak postanowiliśmy je tymczasowo zignorować i spróbować zrealizować postawione przez nas na początku cele. Nie zrealizowaliśmy również jednego aspektu projektu co też stało się naszym planem na przyszłość.

Oświetlenie roślinki przy wykryciu za małej ilości światła.

Automatyzacja procesu podlewania po zczytaniu ustawionej dolnej granicy czujnika wiglotności gleby.

Aplikacja umozliwiająca podgląd parametrów roślinki i zdalne sterowanie.

Domena i działanie projektu w chmurze bez potrzeby włączania serwera na laptopie. Do tego było by potrzebne przeniesienie naszej bazy danych na bazę danych w chmurze (np. Azure), a także implementacja ESP8266 jako serwera.

Literatura

[1] Trutina z Gremon Systems gremonsystems.com/blog-en/things-you-didnt-know-about-automatic-watering-systems/

[2] Arduino Irrigator System http://www.circuitstoday.com/arduino-irrigation-plant-watering-using-soil-moisture-sensor

[3] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1993.

[4] Ivan Grokhotkov *ESP8266 Arduino Core's documentation*. 2017

[5] Christian Klippel, Peter Andersson, Peter Lerup *ESP8266 core for Arduino*. 2017

[6] guy *w3schools.com*. 1999-2019

[7] Albert Einstein. *Zur Elektrodynamik bewegter Körper*. (German) [*On the electrodynamics of moving bodies*]. Annalen der Physik, 322(10):891–921, 1905.

[8] Relay Module , opencircuit.shop/Product/Relay-Module-1-relay.-5V

[9] Pompa spryskiwacza TOPRAN , www.autoczescionline24.pl/topran-2723484.html

[10] Bateria alkaliczna 9V *VARTA*, www.elfadistrelec.pl/pl/alkaliczne-bateria-alkaliczna-9v-6lr61-varta-industrial-9v/p/16901614

[11] Knuth: Computers and Typesetting,
`http://www-cs-faculty.stanford.edu/~uno/abcde.html`