

**ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ
към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

ДИПЛОМНА РАБОТА

**по професия код 481020 „Системен програмист“
специалност код 4810201 „Системно програмиране“**

Тема: Система за активно следене и оценка на слънчевата енергия

Дипломант:

Мартин Георгиев Георгиев

Научен ръководител:

маг. инж. В. Деведжиев

СОФИЯ

2022



**ТЕХНОЛОГИЧНО УЧИЛИЩЕ
ЕЛЕКТРОННИ СИСТЕМИ
към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ -
СОФИЯ**

Дата на заданието: 10.11.2021 г.
Дата на предаване: 10.02.2022 г.

Утвърждавам:.....
/проф. д-р инж. Т. Василева/

ЗАДАНИЕ за дипломна работа

на ученика Мартин Георгиев Георгиев, 12^Б клас

1. Тема: Система за активно следене и оценка на слънчевата енергия
2. Изисквания:
 - 2.1. Да се проучат съществуващите системи за следене и оценка на слънчевата енергия и да се направи сравнителен анализ на възможностите им;
 - 2.2. Да се проучат основните компоненти, подходящи за създаването на система за следене и оценка на слънчевата енергия;
 - 2.3. Да се проучат възможностите за запис и анализ на информацията, получена от системата за следене и оценка на слънчевата енергия;
 - 2.4. Да се дефинират техническите изисквания към системата за следене и оценка на слънчевата енергия;
 - 2.5. Да се подберат подходящи компоненти, да се проектират електрическа схема, печатна платка и вграден софтуер за реализиране на нейните функционални възможности.
3. Съдържание: 3.1 Обзор
 3.2 Същинска част
 3.3 Приложение

Дипломант :.....

/ Мартин Георгиев /

Ръководител:.....

/ доц. д-р инж. Ст. Стефанова /

Директор:.....

/ доц. д-р инж. Ст. Стефанова /

Становище на дипломния ръководител

За периода, през който ръководя разработката на дипломната работа на Мартин Георгиев той се прояви като изключително мотивиран, съзнателен и постоянен в работата си. Ежеседмично му бяха поставяни задачи, които той изпълняваше коректно, с ентузиазъм и предлагайки свои предложения и идеи, и въпреки комплексността на темата на дипломната му работа постигна сериозни резултати. На практика по време на разработката на дипломната му работа знанията и уменията на дипломанта се разшириха както в теоретично, така и в практическо отношение, и то в широк обхват от области: практическо приложение на слънчевата енергия, проектиране на електронни и електромеханични системи, електрически схеми и печатни платки, създаване на управляваща програма за вградена микрокомпютърна система, използване на комуникационен протокол и специфични програмни приложения с цел предаване и визуализация на получените резултати и др.

Въз основа на написаното по-горе считам, че дипломантът Мартин Георгиев трябва да бъде допуснат до защита на дипломната си работа.

Предлагам за рецензент на дипломната работа маг. инж. Симеон Иванов, ръководител на отдел "Хардуер" в направление "Собствени продукти" на фирма "Смартком-България" АД

*Дата: 04.04.2022 г.
гр. София*

*Подпис:
(маг. инж. В. Деведжиев)*

СЪДЪРЖАНИЕ

Увод	7
Първа глава - Проучване на съществуващи системи за следене и оценка на слънчевата енергия и основни компоненти за изграждането им	9
1.1. Съществуващи системи	9
1.1.1. Обзор на конкретни системи	9
1.1.2. Сравнение и анализ на функциите на системите	18
1.2. Основни хардуерни компоненти	20
1.2.1. Слънчеви панели	20
1.2.2. Преобразуватели на напрежение	23
1.2.3. Задвижване	25
1.2.4. Управление	28
1.2.5. Комуникация	31
1.2.6. Индикация	34
1.2.7. Захранване	37
1.2.8. Сензори	39
1.3. Софтуерни компоненти	43
1.3.1. Управляващ софтуер	43
1.3.2. Бази данни	46
1.3.3. Визуализация	47
1.4. Механична конструкция	48
1.4.1. Материали	49
1.4.2. Изработка със CNC фреза	49
1.4.3. Изработка с 3D принтер	49
1.4.4. Използване на готови конструктивни компоненти	50
1.4.5. Специално проектирана конструкция	50
Втора глава - Основни изисквания към системата и блокови схеми	51
2.1. Основни изисквания към проектираната система	51
2.1.1. Архитектурни и комуникационни изисквания	51
2.1.2. Функционални изисквания към системата	51
2.1.3. Изисквания към конструкцията на системата	51
2.2. Обща блокова схема и функции на основните блокове	52
2.3. Схеми и описание на отделните блокове	53
2.3.1. Блок Сензори	53
2.3.2. Блок Задвижване	54
2.3.3. Блок Индикация	55

2.3.4. Блок Комуникация	56
2.3.5. Блок Управление	57
2.3.6. Блок Захранване	58
Трета глава - Проектиране на принципна електрическа схема	59
3.1. Избор на CAD система	59
3.2. Блок Сензори	59
3.2.1. Избор на компоненти	59
3.2.2. Принципна електрическа схема на Блок сензори	63
3.3. Блок Задвижване	64
3.3.1. Избор на компоненти	64
3.3.2. Принципна електрическа схема на Блок задвижване	65
3.4. Блок Индикация	66
3.4.1. Избор на компоненти	66
3.4.2. Принципна електрическа схема на Блок Индикация	69
3.5. Блок Управление	69
3.5.1. Избор на компоненти	69
3.5.2. Принципна електрическа схема на Блок Управление	71
3.6. Блок Комуникация	72
3.7. Блок Захранване	72
3.7.1. Избор на компоненти	72
3.7.2. Принципна електрическа схема на Блок Захранване	76
3.8. Обща принципна електрическа схема	77
Четвърта глава - Печатни платки	78
4.1. Корпуси на използваните компоненти	78
4.1.1. Корпус на женски 12V Barrel Jack 2.1 x 5.5 mm	78
4.1.2. Корпус на диод	79
4.1.3. Корпус на DC-DC преобразувател LM 2596	79
4.1.4. Корпус на ESP32 Wroom devkit	80
4.1.5. Корпус на ACS712 модул	80
4.1.6. Корпуси на резистори	81
4.2. Проектиране на печатната платка	81
4.2.1. Слой Keep-Out	82
4.2.2. Слой Компоненти (Бял печат)	83
4.2.3. Долен слой (Bottom Layer)	84
4.3. Окончателен вариант на проектираната печатна платка	85
4.5. Изработка на печатната платка	86
Пета глава - Управляващ софтуер	87

5.1. Избор на развойна среда	87
5.2. Обща блокова схема на софтуера	88
5.3. Блокова схема на алгоритъма за управление на вградената система	90
5.4. Блокова схема на алгоритъма работа на сървъра	91
5.5. Блокова схема на алгоритъма на приложението за връзка с потребителя	92
5.6. Програмен код на вградената система	92
5.6.1. Използвани библиотеки	93
5.6.2. Константи	94
5.6.3. Функции обработващи данните от сензорите	97
5.6.4. Функции осъществяващи комуникацията	100
5.6.5. Функция контролираща серво моторите	102
5.6.6. setup() функция	103
5.6.7. loop() функция	104
5.7. Програмен код на сървърното приложение	105
5.7.1. Модели	105
5.7.2. Връзка с базата данни	106
5.7.3. REST мрежови контролери	107
5.8. База данни	108
5.9. Софтуер за визуализация на данните	110
5.10. Потребителско интернет приложение	111
Шеста глава - Практически резултати	115
6.1. Подготвителни дейности	115
6.2. Свързване и тестване на хардуерните решения	115
6.2.1. Първоначална подготовка на компоненти	115
6.2.2. Начална работа с микроконтролера	116
6.2.3. Тестове с делители на напрежение	117
6.2.4. Тестове със серво моторите	118
6.2.5. Тестове с компонентите от блок индикация	119
6.2.6. Тестове със сензора на ток и температура	120
6.3. Тестване на софтуерните алгоритми и компоненти	120
6.3.1. Разработка и тестване на Django сървъра и база данни	121
6.3.2. Разработка и тестване на HTTP заявките от ESP32	121
6.3.3. Разработка на информационни табла в Grafana	122
6.3.4. Разработка на потребителско приложение с React и свързването му с Grafana	123
6.4. Изграждане на конструкцията	124

Заключение	126
Използвани съкращения	128
Използвана литература	131

Увод

Почти всички дейности на човека в съвременния свят са свързани с консумацията на различни видове енергия. Поради бурното развитие на технологиите в последното столетие тази консумация се е увеличила многократно. Въпреки, че темповете на увеличение на енергийната консумация на човечеството напоследък се забавят поради глобалното затопляне и негативните климатични промени на Земята, потреблението на енергия все пак продължава да се увеличава средно с 1-2% всяка година [1]. Производството на енергия следва това увеличение, при което конвенционалните енергийни източници ускоряват темповете на замърсяването и глобалното затопляне [2]. Алтернативата са възобновяемите енергийни източници, като най-голям потенциал за развитие има слънчевата енергия, получавана с помощта на фотоволтаици. През последните десетилетия се постигна многократно понижаване на цените на слънчевите панели и повишаване на КПД-то (кофициент на полезно действие) им, като тенденцията е това да продължи [3], което ги прави все по-популярни не само сред индустриалните потребители, но дори и сред по-малките домакинства. За съжаление една от най-значимите причини, заради която много потребители не се възползват от тази технология, е липсата на информация [4].

Стъпка към решаването на този проблем е изграждането на лесно достъпна база данни, съдържаща надеждни резултати за потенциала на фотоволтаичните системи в конкретни географски точки. Приблизителните изчисления на база на географска ширина и метеорологични данни (там където има такива), не могат да отчетат всички фактори и потенциални проблеми, които могат да се появят, като запрашаване, температура и охлаждане на фотоволтаиците, трудности при поддръжката, включително и такива със случаен

характер, например натрупване на листа, природни явления, инциденти и др [5].

Целта на настоящата дипломна работа е да се разработи система за активно следене и оценка на слънчевата енергия в дадена географска точка за по-продължителен период от време. По този начин ще може да се повиши ефективността на работа на системата и да се направи оценка на повишаването на тази ефективност за фотоволтаична система, инсталирана на съответното място.

За реализацията на дипломната работа първоначално ще бъде направено проучване на съществуващи подобни системи и сравнение и анализ на техните функционални параметри. Освен това ще бъдат идентифицирани основните компоненти, необходими за изграждането на такава система. Въз основа на това предварително проучване ще бъдат дефинирани основните изисквания към системата, и ще бъде извършено нейното последователно проектиране, започвайки от блоковата ѝ схема, и продължавайки с избора на компоненти, проектиране на електрическа схема, печатна платка и управляващ софтуер.

Въз основа на постигнатите практически резултати ще бъде направена оценка на изпълнението на целта на дипломната работа, основните изисквания към системата, както и на перспективите за нейното бъдещо развитие.

Първа глава - Проучване на съществуващи системи за следене и оценка на слънчевата енергия и основни компоненти за изграждането им

1.1. Съществуващи системи

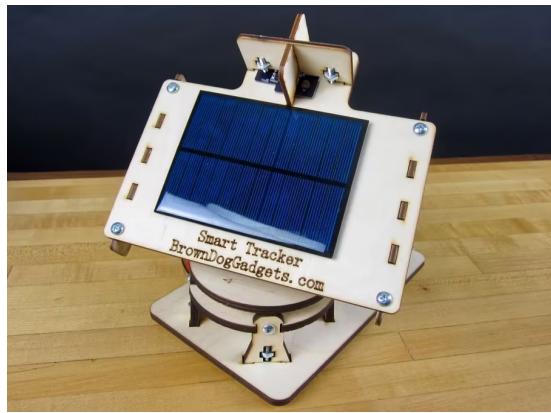
В следващите точки ще бъдат представени избрани като най-интересни като функционалност съществуващи системи, и ще бъде направен анализ и сравнение на техните възможности, въз основа на което ще се представят и съответни изводи.

1.1.1. Обзор на конкретни системи

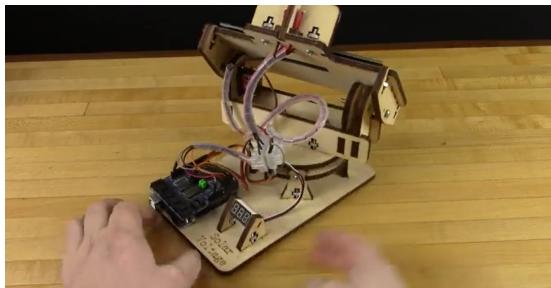
Solar Tracker V2.0

Този проект е публикуван за свободно ползване в "Project Hub" [6]. Той представлява система, чиято основна функционалност е да насочи слънчев панел към най-силният източник на светлина в реално време, като последващото развитие на проекта е оставено в ръцете на потребителите. Целта му не е толкова да е практичен, колкото да се продава като интересен, предварително проектиран, комплект за начинаещи ентузиасти, които искат да навлязат в света на вградените системи и соларните технологии. Въпреки това от него могат да се извлекат полезни идеи.

На фиг. 1.1. е даден фронтален изглед към проекта, а на фиг. 1.2. изглед откъм управлението на системата.



Фиг. 1.1. Снимка на завършения проект отпред



Фиг. 1.2. Снимка на завършения проект отзад

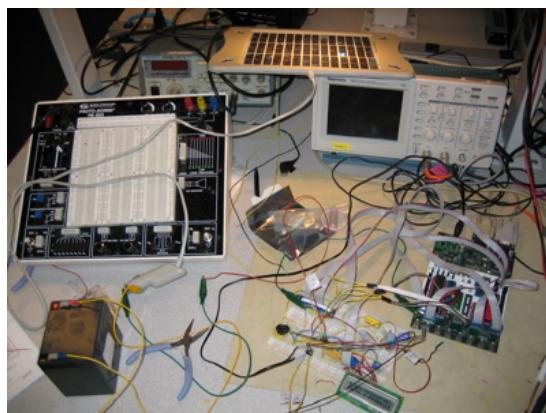
В реализацията на проекта, подвижната част се върти около статична основа. За ориентиране към светлината се разчита на светлочувствителни резистори, а управлението се извършва от микроконтролерната платформа Arduino UNO. Движението на подвижната част се извършва от серво мотори, захранвани и управлявани от споменатия микроконтролер. За по-лесен монтаж са проектирани две печатни платки - съответно за монтаж на светлиенно чувствителните резистори и на управляващата платформа. Системата се захранва от електрическата мрежа и работи при напрежение 5V DC. Кодът на управляващата програма е кратък и добре структуриран, като за управление на серво задвижването се използва библиотеката Servo.h. Самата механична конструкция е изградена от технически шперплат с дебелина четвърт инч.

Табл. 1.1. Основни технически характеристики на система Solar Tracker V2.0

Сълнчев панел:	NA (Not Available)
Ориентация към светлината:	чрез 4 фоторезистора
Измерване:	не
Управление:	Arduino Uno
Комуникация:	Arduino Uno
Интерфейс/ Протокол:	NA
Задвижване:	2 серво мотора SG90 (по две оси)
Захранване:	ел. мрежа, 5 VDC
Конструкция:	фрезован технически шперплат

Self-Powered Solar Data Logger

Проектът е публикуван в списание “CIRCUIT CELLAR” [7]. Това е система, която следи и записва силата на слънчевото лъчение, на мястото на което е инсталриана. Целта на проекта е да бъде създадено решение с приемлива цена, което да позволява на обикновените потребители да измерват слънчевото излъчване в домовете си. Системата се стреми да предостави реалистична представа за ефективната енергия, която може да се очаква от съвкупност от соларни панели, поставена на координатите, на които е проведено измерването, като взема в предвид и загубите при преобразуване на енергията. На фиг. 1.3. е показан изглед към проекта след успешно свързване на електрическите компоненти.



Фиг. 1.3. Изглед към завършения проект

Системата е проектирана и реализирана с идеята да се самоподдържа, тоест да не разчита на електрическата мрежа за захранване. Това е постигнато, като е вградена 12-волтова батерия с капацитет 5 Ah и соларният панел е свързан така, че да зарежда батерията.

Измерванията се постигат със схема, съдържаща фотодиод OPT101 на фирмата Texas Instruments (TI), и трансимпедансен усилвател с едно захранване. Сигнала се подава на микроконтролер STK500, и измерванията се записват в база данни в споменатия микроконтролер, като се пази датата на тяхното отчитане, за да бъде възможно бъдещо извлечане и обработка на данните на компютър. В системата е вграден и малък LCD экран, на който се показват измерванията в реално време и се изписват съобщения и предупреждения при нужда. Управляващият софтуер използва вградения таймер на микроконтролера, за да се постигне максимална точност на измерванията, които се извършват на всеки 30 ms.

Табл. 1.2. Основни технически характеристики на система Self-Powered Solar Data Logger

Сълнчев панел:	18.8 VDC, 3.2W
Ориентация към светлината:	фиксирана
Измерване:	чрез 1 фотодиод на фирмата TI
Управление:	микроконтролер SKT500
Комуникация:	микроконтролер SKT500
Интерфейс/ Протокол:	UART, RS-232
Задвижване:	фиксирано
Захранване:	батерия, 12 VDC, 5Ah
Конструкция:	NA

Solar Tracker with Live Data Feed - Windows IoT

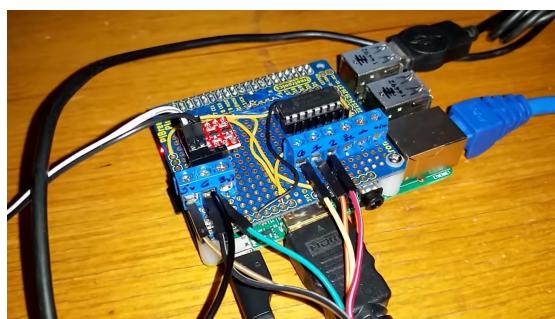
Този проект е взет от сайта “hackster.io” [8]. Дизайнът е проектиран и реализиран като упражнение по интересна за автора тема, поради което са разгърнати много функционалности. Системата е способна да

следи най-силния източник на светлина, като насочва слънчев панел към него. Проектът измерва извлечената ефективна енергия и изпраща данните за нея към свързан компютър. От там те са достъпни чрез потребителски приложен интерфейс. Освен това през приложението може ръчно да се контролира хардуера на системата.

На фиг. 1.4. е показан изглед към механичната конструкция носеща панела, а на фиг. 1.5. се вижда контролния блок със Raspberry Pi 2.



Фиг. 1.4. Снимка на проекта в процеса на разработване



Фиг. 1.5. Снимка на контролният блок

Проектът е реализиран така, че да поддържа връзка с персонален компютър. За тази цел се използват две микроконтролерни платформи. Едната е Raspberry Pi 2, която осъществява комуникацията с компютъра, а другата е Arduino UNO, която контролира двата серво мотора, използвани за задвижване на слънчевия панел. Блокът за

насочване към светлината се състои от 4 светлинно чувствителни резистора, и е фиксиран към соларния панел. Измерванията от този блок се подават на Ардуиното. За по-лесен монтаж са проектирани две печатни платки - съответно за всяка от двете микроконтролерни платформи. Системата се захранва от електрическата мрежа и работи при напрежение 5 VDC. Самата механична конструкция е изградена от метални компоненти. Данните, за енергията получена от слънчевия панел се отчитат с помощта на амперметър и вградените способности на Ардуино платформата за измерване на напрежение. Приложението, което е написано за системата, е предназначено да комуникира с хардуерния модул посредством Windows IoT core средата, инсталирана върху Raspberry Pi 2.

Табл. 1.3. Основни технически характеристики на система Solar Tracker with Live Data Feed - Windows IoT

Слънчев панел:	NA
Ориентация към светлината:	чрез 4 фоторезистора
Измерване:	амперметър ACS715 и делител на напрежение
Управление:	Raspberry Pi 2, Arduino Uno
Комуникация:	Raspberry Pi 2
Интерфейс/ Протокол:	Ethernet
Задвижване:	2 серво мотора (по две оси)
Захранване:	ел. мрежа, 5 VDC
Конструкция:	метална

Floating Sun Tracker Hydraulic Solar Panel

Този проект е публикуван в "Nevon Projects" [9]. Той е създаден с идеята да реши един от основните проблеми на фотоволтаичните системи, а именно факта, че традиционните дизайни заемат голяма площ, която би могла да се използва за други цели. Идеята на разработчиците е да се изградят модули, които могат да плават над

водната повърхност, като по този начин не заемат намаляващата свободна наземна площ. Допълнително предимство на този подход е, че соларните панели ще намаляват изпаренията от водните източници, докато биват пасивно охлаждани. Друго преимущество на непосредствената близост до водоем е улесненото почистване на слънчевите панели, което ще намали и разходите. За повишена ефективност на системата е изградено и активно ориентиране на панела към слънцето, но само по една ос. Важно е да се отбележи, че конкретната реализация е по-скоро прототип, защото не е вградена батерия или решение за съхранение или пренос на събраната енергия. От значение е също, че заради комерсиалната си насоченост като комплект, предназначен да обучи младите ентузиасти и инженери, за проекта не е дадена пълна и подробна информация за свободно ползване.

На фиг. 1.6. е показан проекта предварително визуализиран, а на фиг. 1.7. виждаме завършен прототип в действие.



Фиг. 1.6. Графичен модел на прототип



Фиг. 1.7. Снимка на завършеният проект

Проектът, бидейки нестандартен, налага нестандартни решения. Контролната електрическа схема е заснета, но не се навлиза в детайли за нея. Откриването на светлинен източник се извършва само с 2 светлинно чувствителни диода, поради факта, че насочване има само по една ос. Движението се осъществява чрез система от водни помпи и бутала, като вода се набавя от водоизточника, над който системата е поставена. Контролният блок е съсредоточен в една точка, за да се подобри устойчивостта срещу потенциални щети от водата. Механичната конструкция е изградена предимно от ПВЦ тръби и елементи, както и от допълнителни компоненти за подобряване на плаваемостта.

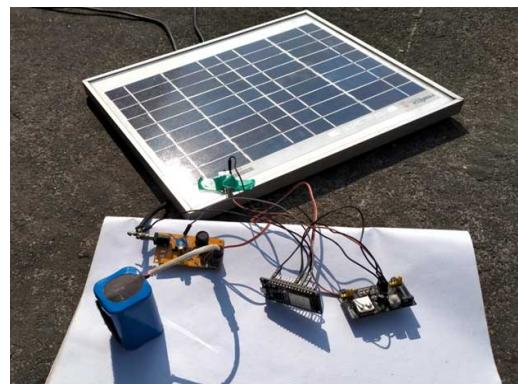
Табл. 1.4. Основни технически характеристики на система Floating Sun Tracker Hydraulic Solar Panel

Слънчев панел:	NA
Ориентация към светлината:	чрез 2 фоторезистора
Измерване:	не
Управление:	NA
Комуникация:	NA
Интерфейс/ Протокол:	NA
Задвижване:	система от водни помпи и бутала (по една ос)
Захранване:	батерия (NA)
Конструкция:	предимно PVC

IoT Based Solar Panel Power Monitoring using ESP32 and ThingSpeak

Този проект е публикуван в сайта “Circuit Digest” [10]. Неговата цел е да позволи лесно, безжично следене на електрическите показатели на отделните соларните панели в една фотоволтаична система. По този начин се очаква да се откриват повреди в слънчевите панели и да се установява на кои площи се натрупва най-много прах. В системата е вграден Maximum Power Point Tracking (MPPT) контролер, т.е.

контролер, който следи максималното производство на свързания към него соларен панел и осигурява както по-добър заряд на батерията, така и по-голяма производителност на системата като цяло. В конкретната реализация на проекта преобразуваната енергия се съхранява в батерия, която освен това захранва системата. На фиг. 1.8. е показан свързаният прототип на проекта, а на фиг. 1.9. виждаме потребителският интерфейс, осъществен с помощта на ThingSpeak.



Фиг. 1.8. Снимка на прототипа на проекта



Фиг. 1.9. Отделни екрани на потребителския интерфейс

Системата е проектирана така, че да бъде максимално компактна. Измерванията се извършват с помощта на шунтов резистор, делител на напрежение, датчик за температура и няколко аналогово-цифрови преобразувателя. Микроконтролерът ESP32 отчита измерените данни и се грижи те да достигнат до сървър, поддържан в конкретния случай на отделен преносим компютър. Преносът на данни се осъществява безжично, посредством Wi-Fi. Проектът е замислен като част от система

от слънчеви панели, захранващи голям консуматор, но за демонстративни цели, енергията се съхранява в батерия с капацитет 4Ah и напрежение 7.4V, което се конвертира на 5V. MPPT контролер управлява потока на енергия, като осигурява максимално ефективно и бързо зареждане на батерията. Изграден е и потребителски приложен интерфейс с помощта на софтуера ThingSpeak.

Табл. 1.5. Основни технически характеристики на система IoT Based Solar Panel Power Monitoring using ESP32 and ThingSpeak

Слънчев панел:	18 VDC
Ориентация към светлината:	чрез 2 фоторезистора
Измерване:	шунтов резистор, делител на напрежение, сензор за температура
Управление:	ESP32
Комуникация:	ESP32
Интерфейс/ Протокол:	Wi-Fi
Задвижване:	фиксирано
Захранване:	батерия 7.4 VDC / 4Ah, преобразувано на 5 VDC
Конструкция:	NA

1.1.2. Сравнение и анализ на функциите на системите

В таблица 1.6. по-долу е представено сравнение на параметрите и функционалните особености на системите, разгледани в т. 1.1.1.

Табл. 1.6. Сравнение на избраните проекти и възможностите им

Система → Параметър ↓	Solar Tracker V2.0	Self-Powered Solar Data Logger	Solar Tracker with Live Data Feed	Floating Sun Tracker	IoT Based Solar Panel Power Monitoring
Слънчев панел	NA	18.8V, 3.2W	NA	NA	18V
Ориентация към светлината	чрез 4 фоторезистора	фиксирана	чрез 4 фоторезистора	чрез 2 фоторезистора	фиксирана
Измерване	не	чрез 1	чрез	не	шунтов

		фотодиод на фирмата TI	амперметър ACS715 и делител на напрежение		резистор, делител на напрежение, сензор за температура
Управление	Arduino Uno	микроконтролер SKT500	Raspberry Pi 2, Arduino Uno	NA	ESP32
Интерфейс / Протокол	NA	UART, RS-232	Ethernet	NA	Wi-Fi
Задвижване	2 серво мотора	фиксирано	2 серво мотора	спец. водна система	фиксирано
Захранване	ел. мрежа, 5 VDC	батерия, 12 VDC, 5Ah	ел. мрежа, 5 VDC, 3.3 VDC	NA	батерия, 5 VDC, 4Ah
Конструкция	фрезован технически шперплат	NA	метална	предимно PVC	NA
Допълнителни функционалности	елегантна конструкция, проектът умишлено е запазен прост	захранване независимо от електрическата мрежа	графичен потребителски интерфейс с възможност за ръчно управление на системата	лека конструкция, предназначена да плава на повърхността на воден басейн	безжичен пренос на данни, графичен потребителски интерфейс, независимо захранване, MPPT контролер

Въз основа на сравнението на основните параметри на разгледаните системи, представени в таблица 1.6, могат да се направят определени коментари и изводи:

- В проектите, в които се измерва слънчевата енергия, се използват панели с 18V изходно напрежение, въпреки че това не се посочва като задължително.
- Ориентацията към светлината (слънцето) се извършва по три начина - фиксирано, с 2 или 4 фоторезистора, в зависимост от желаната точност и прецизност;

- Измерването на слънчевата енергия в някои проекти отсъства, а в останалите става по три различни начина - чрез фотодиод, амперметър или шунтов резистор;
- Множество микроконтролерни платформи са подходящи за целите на проекта. Използвани са популярни решения (Arduino, Raspberry, ESP). Основното, което ги различава е комуникационният интерфейс;
- Относно предаването на данни, преобладаващият метод е изпращане в реално време. Използвани са както безжични, така и физически преносни среди (Wi-Fi, Ethernet);
- Задвижването, където е реализирано, се постига или чрез серво мотори, или чрез хидравлична система, в зависимост от работните условията и изискванията за прецизност.
- По отношение на захранването има два основни подхода - връзка с електрическата мрежа и независимо захранване с батерия;
- Конструкциите се различават много по отношение на избрания материал. Вариациите са от дърво (шперплат) до метал и PVC, в зависимост от структурните и функционални изисквания към проекта.

1.2. Основни хардуерни компоненти

Във всяка от предстоящите точки ще бъде представен кратък списък от сходни компоненти за реализацията на проекта като в заключение ще се направи анализ на техните предимства и недостатъци.

1.2.1. Слънчеви панели

Поради това, че в проектите, разгледани по-горе, в които се измерва слънчевата енергия, преобладават слънчеви панели с 18 V изходно напрежение, които позволяват по-голяма гъвкавост при избора на преобразувател и батерия, ще бъдат разгледани именно такъв тип

слънчеви панели. Основните параметри, които трябва да се вземат предвид при този избор, са изходното напрежение, максималният ток и мощността (и трите трябва да са в диапазона на преобразувателя на напрежение, включен след панела), както и размерите, площта и теглото, от които от своя страна зависи възможността за създаването на подходяща конструкция. По-долу са разгледани основните налични на пазара панели, подходящи за системата.

Solar panel CL-SM5P



Фиг. 1.10 Изглед към Solar panel CL-SM5P

Табл. 1.7. Технически параметри на Solar panel CL-SM5P [11]

Изходно напрежение:	18.2 V
Мощност (при силно слънце):	5 W
Номинален ток:	0.28 A
Размери/ площ:	251x186x18 mm/ 4.66 dm ²
Ефективност:	1.07 W/dm ²
Тегло:	0.640 kg
Цена:	24.90 лв

Соларен панел, фотоволтаичен elektronikabg.com-30401288



Фиг. 1.11 Изглед към панел elektronikabg.com-30401288

Табл. 1.8. Технически параметри на соларен панел elektronikabg.com-30401288 [12]

Изходно напрежение:	17.64 V
Мощност (при силно слънце):	5 W
Номинален ток:	0.28 A
Размери/ площ:	270x250x17 mm / 6.75 dm ²
Ефективност:	0.74 W/dm ²
Тегло:	0.850 kg
Цена:	25.00 лв

Монокристален фотоволтаичен панел

solarhouse.bg-12345-015-1



Фиг. 1.12 Изглед към панел solarhouse.bg-12345-015-1

Табл. 1.9. Технически параметри на монокристален фотоволтаичен панел solarhouse.bg-12345-015-1 [13]

Изходно напрежение:	18.00 V
Мощност (при силно слънце):	5 W
Номинален ток:	0.28 A
Размери/ площ:	230x205x18 mm / 4.715 dm ²
Ефективност:	1.05 W/dm ²
Тегло:	0.600 kg
Цена:	16.90 лв

Табл. 1.10. Съпоставка на разгледаните продукти

Продукт→ Параметър ↓	CL-SM5P	elektronikabg.com- 30401288	solarhouse.bg- 12345-015-1
Изходно напрежение:	18.2 V	17.64 V	18.00 V
Мощност (при силно слънце):	5 W	5 W	5 W
Номинален ток:	0.28 A	0.28 A	0.28 A

Размери/ площ:	251x186x18 mm / 4.66 dm ²	270x250x17 mm / 6.75 dm ²	230x205x18 mm / 4.715 dm ²
Ефективност:	1.07 W/dm ²	0.74 W/dm ²	1.05 W/dm ²
Тегло:	0.640 kg	0.850 kg	0.600 kg
Цена:	24.90 лв	25.00 лв	16.90 лв

След сравнение на трите продукта може да се направи извода, че соларен панел CL-SM5P има най-висока ефективност, второ, има най-добър баланс между тегло и структурна здравина, и освен това е по-средата по отношение на тегло и цена спрямо останалите два панела, но панела предлаган от solarhouse.bg е най-достъпен, преди да се вземе в предвид цената на доставка.

1.2.2. Преобразуватели на напрежение

Преобразувател на напрежение ще се използва във системата във всеки случай, за да предостави връзка между компонентите нуждаещи се от различно захранващо напрежение, за да функционират.

DC/DC понижаващ конвертор BX. 3,2-40V, ИЗХ. 1,25-35V / 20W



Фиг. 1.13 Изглед към DC/DC понижаващ конвертор BX. 3,2-40V, ИЗХ. 1,25-35V / 20W

Табл. 1.11. Технически параметри на DC/DC понижаващ конвертор BX. 3,2-40V, ИЗХ. 1,25-35V / 20W [14]

Входно напрежение:	3.2 - 40 VDC
Изходи:	1.25 - 35 VDC
Максимален ток:	3 A
Цена:	3.90 лв

Модул DC/DC AC/DC конвертор понижаващ 12V 2A



Фиг. 1.14 Изглед към Модул DC/DC AC/DC конвертор понижаващ 12V 2A

Табл. 1.12. Технически параметри на Модул DC/DC AC/DC конвертор понижаващ 12V 2A [15]

Входно напрежение:	40-95 VDC; 40-70 VAC
Изходи:	12 VDC
Максимален ток:	2 A
Цена:	14.40 лв

Табл. 1.13. Съпоставка на разгледаните продукти

Продукт→ Параметър ↓	DC/DC конвертор BX. 3,2-40V, ИЗХ. 1,25-35V	DC/DC AC/DC конвертор понижаващ 12V 2A [15]
Входно напрежение:	3.2 - 40 VDC	40-95 VDC; 40-70 VAC
Изходи:	1.25 - 35 VDC	12 VDC
Максимален ток:	3 A	2 A
Цена:	3.90 лв	14.40 лв

От сравнението на преобразувателите в горната таблица може да се направи извода, че най-подходящ от практическа гледна точка е DC/DC понижаващ конвертор BX. 3,2-40V, ИЗХ. 1,25-35V [14], тъй като

той притежава необходимата функционалност и е достатъчно компактен.

1.2.3. Задвижване

От точка 1.1. може да заключим, че задвижваща система от два серво мотора е най-популярният и гъвкав избор. Най-често срещани са малки електромотори, поради факта, че са достъпни, имат ниска консумация и успяват да предоставят достатъчно сила. Имайки предвид, че предпочитаният соларен панел е с тегло 0.640 kg, без да се включва масата на конструкцията, по-мощен избор би се оказал ключов.

Серво SG90



Фиг. 1.15 Изглед към servo SG90

Табл. 1.14. Технически параметри на servo SG90 [16]

Работно напрежение:	4.8 - 6 VDC
Въртящ момент:	1.6 kg/cm
Размери:	22.6mm x 12.1mm x 22.5mm
Цена:	3.52 лв

Стъпков мотор 17HS4401



Фиг. 1.16 Изглед към стъпков мотор 17HS4401

Табл. 1.15. Технически параметри на стъпков мотор 17HS4401 [17]

Работно напрежение:	3.6 - 36 VDC
Въртящ момент:	4.3 kg/cm
Размери:	42mm x 42mm x 38mm
Цена:	18.26 лв (без доставка)

Servo MG996



Фиг. 1.17 Изглед към Servo MG996

Табл. 1.16. Технически параметри на Servo MG996 [18]

Работно напрежение:	4.8 - 6 VDC
Въртящ момент:	13 kg/cm
Размери:	40mm x 19mm x 43mm
Цена:	9.82 лв

Табл. 1.17. Съпоставка на разгледаните продукти

Продукт→ Параметър ↓	SG90	стъпков мотор 17HS4401	MG996
Работно напрежение:	4.8 - 6 VDC	3.6 - 36 VDC	4.8 - 6 VDC
Въртящ момент:	1.6 kg/cm	4.3 kg/cm	13 kg/cm
Размери:	22.6mm x 12.1mm x 22.5mm	42mm x 42mm x 38mm	40mm x 19mm x 43mm
Цена:	3.52 лв	18.26 лв (без доставка)	9.82 лв

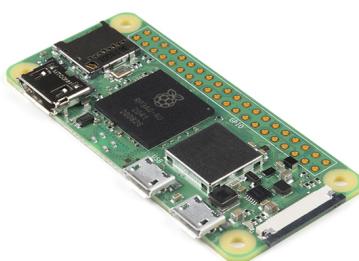
Сравнението на трите серво мотора показва, че за модел MG996 предлага неколкократно по-голям въртящ момент, на достъпна цена, но стъпковият мотор прелага по-голяма прецизност, а SG90 е най-широко разпространен.

1.2.4. Управление

За един IoT проект изборът на подходяща управляваща платформа за разработка е може би най-важен, тъй като тя трябва да може да обработва необходимото количество данни, да формира съответните управляващи команди, както и да изпраща отчетените резултати до сървър чрез съответния протокол за връзка. По-долу са разгледани три подходящи за управлението на системата платформи, които ще бъдат сравнени по своята функционалност и технически параметри.

Raspberry Pi Zero 2 W

Raspberry Pi микроконтролерите, като цяло са много бързи, надеждни и лесни за ползване. На практика имат всички функции на един конвенционален личен компютър, като основната е поддръжката на операционна система. Zero 2 W е сред най-малките и достъпни продукти на компанията.



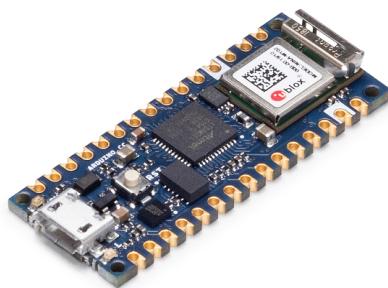
Фиг. 1.18 Изглед към Raspberry Pi Zero 2 W

Табл. 1.18. Технически параметри на Raspberry Pi Zero 2 W [19]

Захранване:	5 VDC
Процесор:	Broadcom BCM2710A1, четириядрен 64-bit SoC (Arm Cortex-A53 @ 1GHz)
Оперативна памет:	512 MB LPDDR2
Консумация (икономичен режим):	~90 mA
Периферни портове:	mini HDMI, 2 x micro USB, 28 GPIO порта
Безжична комуникация:	Wi-Fi 2,4 GHz, Bluetooth 4.2, BLE
Цена:	45.0 лв

Arduino Nano 33 IoT

Arduino е най-широко разпространената и поддържана платформа за разработка на вградени проекти и прототипи сред ентузиастите. Този конкретен модел се отличава с малкия си размер и безжична комуникация.



Фиг. 1.19 Изглед към Arduino Nano 33 IoT

Табл. 1.19. Технически параметри на Arduino Nano 33 IoT [20]

Захранване:	5 VDC
Процесор:	SAMD21 Cortex®-M0+ 32bit low power ARM MCU (48 MHz)
Оперативна памет:	32 kB
Консумация (икономичен режим):	~20 mA
Периферни портове:	micro USB, 22 GPIO порта
Безжична комуникация:	Wi-Fi 2,4 GHz, Bluetooth 4.2, BLE
Цена:	50.0 лв

ESP32 NodeMCU-32S

Платформата ESP е известна с високото представяне и много функции на ниска цена. Тя споделя една и съща развойна среда с Arduino и също се поддържат много библиотеки и материали за нея.



Фиг. 1.20 Изглед към ESP32 NodeMCU-32S

Табл. 1.20. Технически параметри на ESP32 NodeMCU-32S [21]

Захранване:	5 VDC
Процесор:	Single or Dual-Core 32-bit LX6 Microprocessor (240 MHz)
Оперативна памет:	520 kB
Консумация (икономичен режим):	~20 mA
Периферни портове:	micro USB, 29 GPIO порта
Безжична комуникация:	Wi-Fi 2,4 GHz, Bluetooth 4.2, BLE
Цена:	24.0 лв

Табл. 1.21. Съпоставка на разгледаните продукти

Продукт→ Параметър ↓	Raspberry Pi Zero 2 W	Arduino Nano 33 IoT	ESP32 NodeMCU-32S
Захранване:	5 VDC	5 VDC	5 VDC
Процесор:	Broadcom BCM2710A1, четириядрен 64-bit (1 GHz)	SAMD21 Cortex®-M0+ 32bit low power ARM MCU (48 MHz)	Single or Dual-Core 32-bit LX6 Microprocessor (240 MHz)
Оперативна памет:	512 MB LPDDR2	32 kB	520 kB
Консумация (икономичен режим):	~90 mA	~20 mA	~20 mA
Периферни портове:	mini HDMI, 2 x micro USB, 28 GPIO порта	micro USB, 22 GPIO порта	micro USB, 29 GPIO порта
Безжична комуникация:	Wi-Fi 2,4GHz, Bluetooth 4.2, BLE	Wi-Fi 2,4GHz, Bluetooth 4.2, BLE	Wi-Fi 2,4GHz, Bluetooth 4.2, BLE
Цена:	45.0 лв	50.0 лв	24.0 лв

След оглед и сравнение на трите популярни микроконтролерни платформи, може да се заключи, че ESP32 е най-подходящ за проекта, тъй като предлага средно голяма изчислителна мощност и най-много GPIO порта, на най-ниска цена.

1.2.5. Комуникация

Съществуват различни IoT комуникационни технологии, интерфейси и протоколи, които биха могли да се използват успешно в системата. Те имат различни характеристики и функционални възможности, което допълнително усложнява най-подходящия за конкретния случай избор. По-долу са разгледани някои от най-популярните технологии за комуникация.

WiFi (802.11n)

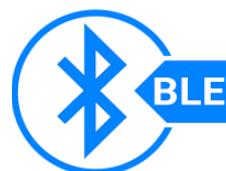


Едно от предимствата на WiFi е, че това е доказана и стандартизирана технология, която вече присъства в много сгради и обществени зони. Следващото му предимство е, че избраният микроконтролер го поддържа и дори има вграден комуникационен модул. [22]

Табл. 1.22. Характеристики на WiFi

Обхват:	70 m (закрито) / 250 m (открито)
Скорост:	70-600 Mb/s
Консумация:	120 mA (ESP32 using WiFi)
Сигурност:	висока
Цена:	(на практика безплатен)

BLE (Bluetooth 4.2)



BLE е версия на Bluetooth с по-ниска електрическа консумация, която го прави подходящ за IoT системи. Друго негово предимство е, че, с негова помощ, сравнително бързо се изграждат мрежи, с възможност за

разрастване. Също така, избраният микроконтролер го поддържа и има вграден комуникационен модул. Недостатък е, че няма разпространени шлюзове към интернет мрежата за BLE. [22]

Табл. 1.23. Характеристики на BLE

Обхват:	50-100 м
Скорост:	125 kb/s - 3 Mb/s
Консумация:	70 mA (ESP32 using BLE)
Сигурност:	средна
Цена:	налага се закупуването на BLE интернет шлюз (30 лв)

LoRa



Предимствата на LoRa са големият ѝ обхват при много ниска консумация, двустранна комуникация и сигурност. Недостатъците включват слабо разпространена инфраструктура, нужда от отделни модули за комуникацията и трудна настройка. [22][23]

Табл. 1.24. Характеристики на LoRa

Обхват:	2-5 km (в населено място) / 15 km (извън населено място)
Скорост:	10-50 kb/s
Консумация:	5 mA
Сигурност:	висока
Цена:	в зависимост от наличната инфраструктура

Mobile (NB-IoT)



NB-IoT използва съществуващата LTE инфраструктура, което го прави най-разпространеният от досега изброените протоколи. Има голям обхват и ниска консумация. Недостатъците му са висока цена, нуждата от отделни модули за комуникацията и трудна настройка. [23]

Табл. 1.25. Характеристики на WiFi

Обхват:	1 km (в населено място) / 10 km (извън населено място)
Скорост:	26-62 kb/s
Консумация:	5 mA
Сигурност:	висока
Цена:	~5lv / 10MB

Табл. 1.26. Съпоставка на разгледаните протоколи

Протокол→ Параметър ↓	WiFi (802.11n)	BLE (Bluetooth 4.2)	LoRa	NB-IoT
Обхват:	70 m (закрито) / 250 m (открито)	50-100 m	2-5 km (в населено място) / 15 km (извън населено място)	1 km (в населено място) / 10 km (извън населено място)
Скорост:	70-600 Mb/s	125 kb/s - 3 Mb/s	10-50 kb/s	26-62 kb/s
Консумация:	120 mA (ESP32 using WiFi)	70 mA (ESP32 using BLE)	5 mA	5 mA
Сигурност:	висока	средна	висока	висока
Цена:	(на практика бесплатен)	налага се закупуването на BLE интернет шлюз (30 lv)	в зависимост от наличната инфраструктура	~5lv / 10MB

Въз основа на сравнението на параметрите в таблицата по-горе може да се направи извода, че поради широката си разпространеност, относителна простота и липсата на допълнителни разходи за връзка

към Интернет, WiFi протоколът е най-подходящ за комуникация при разработването на прототип на системата.

1.2.6. Индикация

В този сегмент ще бъдат разгледани предложения за физически индикатори, които биха били монтирани на физическия модул на системата, с цел лесно, бързо, надеждно и прегледно показване на важна информация.

Светодиоди



Фиг. 1.21. Изображения показващи светодиоди

Предимствата на светодиодите са тяхната ниска цена, лесно въвеждане и използване в системата и способността им за много ясна и нагледна индикация на приста информация. Светодиодите обикновено консумират между 5 и 20mA.

LCD

Тези екрани са широко достъпни, и добре познати. Предлагат добра гъвкавост при изобразяването на информация, на достъпни цени. Имат ниска консумация - 40mA за стандартен 16x2 экран.



Фиг. 1.22. Изглед към символен LCD

Табл. 1.27. Характеристики на Дисплей LCD 16x2 - син [25]

Захранване:	5 VDC
Резолюция:	2 реда x 16 символа (5x7 px)
Консумация:	40 mA
Интерфейс:	8 информационни пина
Цена:	8.00 лв

OLED еcran

По-modерната технология на OLED екраните предлага висок контраст и ниска консумация, като типично тези екрани имат и по-голяма резолюция в по-малко пространство.



Фиг. 1.23. Изглед към OLED еcran

Табл. 1.28. Характеристики на Дисплей OLED 0.96" 128x64, SSD1306 [26]

Захранване:	5 VDC
Резолюция:	64x128 px
Консумация:	25 mA
Интерфейс:	I ² C, SPI
Цена:	15.00 лв

Еcran с "електронно мастило" (e-ink)

Като по-тясно разпространена технология, тези екрани не са особено достъпни, но предлагат много ниска консумация.



Фиг. 1.24. Изглед към мастилен экран

Табл. 1.29. Характеристики на E-Ink /e-Paper module/ [27]

Захранване:	3-5 VDC
Резолюция:	200x200 px
Консумация:	поддръжка: 0,004mA, опресняване: 85mA (еднократно)
Интерфейс:	SPI
Цена:	33.00 лв

Табл. 1.30. Съпоставка на разгледаните продукти

Продукт Параметър ↓	LCD	OLED еcran	Мастилен еcran
Захранване:	5 VDC	5 VDC	3-5 VDC
Резолюция:	2 реда x 16 символа (5x7 px)	64x128 px	200x200 px
Консумация:	40 mA	25 mA	поддръжка: 0,004mA, опресняване: 85mA (еднократно)
Интерфейс:	8 информационни пина	I ² C, SPI	SPI
Цена:	8.00 лв	15.00 лв	33.00 лв

Въз основата на направеното по-горе сравнение, комбинация от светодиоди и дисплей се очертава като най-подходящото решение. LCD дисплеят, макар и не толкова ефикасен е най-достъпен от трите, докато OLED екранът предлага интересен баланс от качества, функции и цена. Също може да се направи извода, че предимствата на экрана с

“електронно мастило”, трудно оправдават сравнително високата му цена.

1.2.7. Захранване

Захранващият блок в една система зависи от желаната функционалност и цялостната концепция на проекта. В случая възможните варианти са основно два - единият е захранващ блок с батерия, зареждана от слънчеви панел, или захранващ блок, свързан с електрическата мрежа. Разбира се двата варианта могат да се обединят в един комбиниран захранващ блок, което има някои предимства, напр. по-голяма гъвкавост, но и известни недостатъци - по-голяма сложност и цена. Двата варианта ще бъдат разгледани по-долу самостоятелно / независимо един от друг.

Мрежово захранване

Този тип захранване може да се интегрира сравнително лесно в проекта, но необходимостта от физическа връзка с електрическата мрежа е недостатък, който би ограничил сериозно обхвата, в който системата може да функционира. За да се реализира захранването е нужен преобразувател на напрежение, който да намалява променливото напрежение на електрическата мрежа, 220 VAC, до по-ниско, и го преобразува и стабилизира в постоянно, напр. 5 VDC, което е подходящо за захранване на компонентите на системата. Пример за такъв преобразувател е даден на фигура 1.26. Друг недостатък на този подход е зависимостта от стабилността и надеждността на електрическата мрежа.



Фиг. 1.25. Изглед към AC-DC Buck Converter AC 220v to 5v [28]

Батерийно захранване

При захранване с батерия споменатите по-горе недостатъци на мрежовото захранване се избягват. От своя страна обаче недостатък при този подход е неизбежният разряд на батерията и евентуалното ѝ пълно изтощение. В конкретния случай този проблем се решава като енергията, генерирана от слънчевия панел, зарежда и се съхранява в батерията, което позволява на теория (при достатъчно слънчеви дни и часове) постоянно батерийно захранване. Примерна батерия, подходяща за захранване на системата, е дадена на фигура 1.27.



Фиг. 1.26. Изглед към пакет от пет батерии, модел 18650 5V

Комбинирано захранване

Както се спомена по-горе, всеки от двата варианта има известни предимства и недостатъци.

Възможен е и трети - комбиниран вариант, както бе споменато в началото, който, макар и по-сложен и скъп, би осигурил по-голяма сигурност и надеждност на захранването на системата.

Най-практичният начин за реализацията на тази идея би бил добавяне на порт към батерийното решение, позволяващ зареждане на батерията, посредством физическа връзка. Комбинираният подход ще елиминира риска от твърде дълъг период без достатъчна слънчева светлина за зареждането на батерията, като освен това запазва автономността и гъвкавостта на батерийният вариант, без да прави проекта зависим от електрическата мрежа.

1.2.8. Сензори

Целта на проекта преди всичко е да се оцени ефективността на преобразуване на получената от слънчевия панел енергия в зависимост от това дали той е статичен или следи динамично слънцето. Основните параметри, които могат да бъдат проследени, са извлечената от панела енергия и температурата на същия. За целта трябва да се реализира амперметър, волтметър и температурен сензор. По този начин ще се получава детайлна информация за работата на слънчевия панел. За насочване на динамичният слънчев панел се налага и използването на сензори за светлина.

Амперметър / сензор на ток

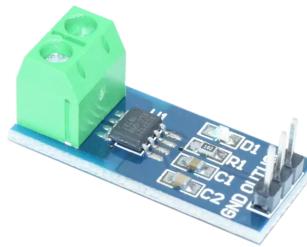
За измерване на потока на генерирания ток може да се подхodi по два начина. Първият е измерване на пад на напрежение върху шунтов резистор, какъвто е показан на фигура 1.28., поставен последователно на положителния извод на слънчевият панел, и се измерва напрежението върху него. Вторият начин предполага използването на амперметър, който да отчита и изпраща данни за измерения ток на микроконтролера. Такъв сензор е показан на фигура 1.29.



Фиг. 1.27. Изглед към шунтов резистор

Табл. 1.31. Технически параметри на шунтов резистор [29]

Номинален ток:	20 A
Номинален пад на напрежение:	60 mV
Клас на точност:	0.5 (0.5%)
Цена:	11.90 лв



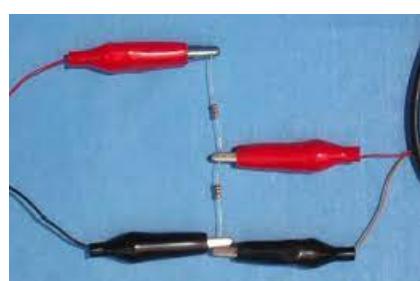
Фиг. 1.28. Изглед към сензор за ток ACS712

Табл. 1.32. Технически параметри на сензор за ток ACS712 [30]

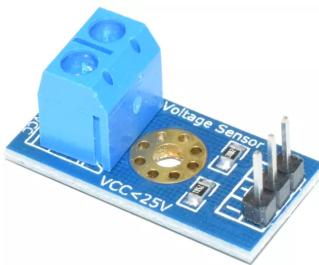
Захранване:	5 V
Номинален ток:	25 A
Стойност на вътрешния резистор:	1.2 mOhm
Максимално напр-е на изолация	2100 V _{RMS}
Изход:	2.5 - 5 VDC аналогов сигнал
Грешка:	1.5 %
Цена:	6.30 лв.

Волтметър / сензор на напрежение

Измерването на напрежение предполага използването на двата възможни подхода, а именно използването на схема с делител на напрежение (фигура 1.30.) или използването на сензор, който измерва напрежението и подава информацията за него на микроконтролер. Такъв сензор е показан на фигура 1.31.



Фиг. 1.29. Изглед към делител на напрежение



Фиг. 1.30. Изглед към сензор за напрежение

Табл. 1.33. Технически параметри на сензор за напрежение [31]

Захранване:	5 V
Измервателен обхват:	0-25 V
Цена:	2.60 лв

Сензор на температура

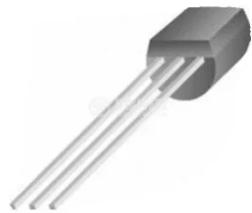
Съществуват много решения за измерване на температура. Те могат да бъдат както безконтактни така и контактни, но в конкретния случай първото би усложнило и осъщели проекта без да носи практически предимства. Контактните сензори варират от обикновени терморезистори до готови сензори за температура, като ценовата разлика е пренебрежима. Пример за терморезистор е разгледан показан на фигура 1.32., а на фигура 1.33. може да се види сензор за температура.



Фиг. 1.31. Изглед към терморезистор

Табл. 1.34. Технически параметри на конкретен терморезистор [32]

Съпротивление:	100 kOhm
Топлинни загуби(25°C):	500 mW
Толеранс:	5%
Работен обхват:	-40 °C до +125 °C
Цена:	1,90 лв



Фиг. 1.32. Изглед към температурен сензор IC LM335AZ

Табл. 1.35. Технически параметри на температурен сензор IC LM335AZ [33]

Захранване:	5-40 V
Обхват на измерване:	-40 °C до +100 °C
Грешка:	± 0.5 °C;
Сигнал:	10mV / C;
Цена:	2.90 лв

Фоторезистори

Фоторезисторите са светочувствителни резистори, които често се използват в проектирането на електронни схеми, когато е необходимо да се следи светлинен източник или да се реагира на наличието на светлина. Въпреки че за целта могат да се използват и други електронни компоненти, като например фотодиоди и фототранзистори, фоторезисторите са по-подходящи подходящи в много случаи, тъй като осигуряват голяма промяна в съпротивлението при малки промени в нивото на светлината. С оглед на тяхната ниска цена/евтино производство, както и лесно използване, фоторезисторите имат много приложения. [34]



Фиг. 1.33. Изглед към светлинно чувствителен резистор

1.3. Софтуерни компоненти

Софтуерът за подобен проект включва в себе си различни компоненти, най-важният от които е непосредствено инсталираният в паметта на микроконтролера управляващ софтуер (фърмуер). Някои от останалите компоненти на софтуера са свързани например с поддръжка на безжичната връзка и реализиране на потребителски интерфейс.

1.3.1. Управляващ софтуер

Съществуват два основни подхода при разработката на управляващ софтуер за вградени микрокомпютърни системи (ВМКС). Първият е базиран на използването на операционна система, която да упражнява контрол над ресурсите на ВМКС, докато вторият предполага конфигуриране на ресурсите на контролера на ниско (хардуерно) ниво [35].

Операционна система за вградени системи (Embedded OS)

Този вид операционни системи (ОС) са проектирани да бъдат компактни, надеждни и ефективни при използване на ресурсите на управляващия модул, а именно паметта и изчислителната му мощност. Много функции, които стандартните ОС за настолни компютри предоставят, са премахнати, защото не са нужни за специализираните функции, които вградените проекти изпълняват.

Тези ОС изпълняват вградени функции и приложения, представляващи софтуер, който се инсталира постоянно в устройството, за да изпълнява специфичен набор от задачи.

Предимството на този подход е, че предлага лесен, надежден и ефективен начин за бързо менажиране на по-сложен софтуер [36], но налага сравнително мощн микроконтролер и се инициализира по-сложно.

Bare Metal Programming, т.е. директно програмиране (без ОС)

Този вид програмиране и управление на ВМКС означава създаване на приложение, управляващо директно хардуера, без използване на допълнителен интерфейс между приложението и хардуера, както и без допълнителни софтуерни модули, управляващи едновременното изпълнение на различни приложения. Това означава писане на приложения с директен достъп до хардуерните регистри и паметта на микроконтролера [37]. Този подход се отличава с голяма ефективност, лесна инициализация и се препоръчва при писането на относително прости еднозадачни приложения.

Езици за програмиране:

За създаването на управляващия софтуер ще е нужен програмен език, предлагащ добра гъвкавост и продуктивност. За софтуера, свързан със сървърни приложения и бази данни обаче най-често се налага използването на език, различен от този, използван за вградения управляващ софтуер.

C/C++



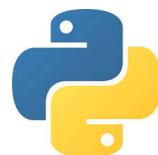
Със C/C++ се създава много компактен, бърз и лек за изпълнение код и той вече е езикът по избор за 95% от програмистите в света на вградените технологии [38]. Той е изпитан във времето, широко разпространен и поддържан и за него са предоставени ненадминат брой материали и помощни средства в Интернет пространството.

Lua



Lua е ефективен, лек, скриптов език, който може да се използва за вградени системи. Той комбинира прост процедурен синтаксис с конструкции за описание на данни. Lua се въвежда динамично, работи чрез интерпретиране на байт код, има автоматично управление на паметта с постепенно събиране на боклука, което го прави подходящ за конфигуриране, скриптове и бързо прототипиране [39].

Phyton



Python може да бъде най-добър, когато се използва като комуникационен посредник между потребителя и вградената система. Изпращането на съобщения чрез Python към или от вградена система позволява на потребителя да автоматизира тестването [39]. Python може да се използва и за получаване на данни от вградени системи, които могат да се съхраняват за анализ с помощта на база данни. За целта е необходимо използването на допълнителни разширения на езика, подобни на Django или Flask.

JavaScript (React)

Последната брънка във веригата е потребителският интерфейс. Javascript се налага като основният програмен език в света на мобилните приложения. React Native е утвърдена като водещата JavaScript framework (програмна структура) за мобилни приложения. React Native е бърз, добре поддържан и много рентабилен. В резултат

на това той продължава да расте в популярност поради способността си да работи с всяка платформа едновременно. [40]

Интегрирана среда за разработка (IDE):

Технически, създаването на софтуер за една вградена система е възможно във всяка интегрирана среда за разработка, но най-разпространената, заради семплия си интерфейс е Arduino IDE. Освен това много популярен избор като среда за разработка е Visual Studio Code, която поддържа голямо разнообразие от езици и разширения за тях.

1.3.2. Бази данни

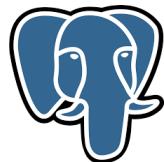
Избраната база данни трябва да бъде бърза и лесно разширяема. Поради факта, че ще се съхраняват еднотипни данни, няма специфични изисквания за нейната гъвкавост [41].

InfluxDB



InfluxDB е специално създадена платформа за данни от времеви серии. Тя е проектирана да обработва данни от милиони точки и стотици източници на данни от IoT устройства и среди за наблюдение. Тази база данни не е релационна. За разлика от релационните бази данни, InfluxDB е по-малко структурирана/ограничена във формат и по този начин позволява по-голяма гъвкавост и адаптивност. Тя е много разпространена и използвана както сред ентузиасти, така и сред комерсиални потребители [42].

PostGreSQL



Това е мощна обектно-релационна, свободна за ползване база данни, която използва и разширява SQL езика. PostgreSQL идва с много функции, целящи да помогнат на разработчиците да изграждат приложения, на администраторите - да защитят целостта на данните и да изградят устойчиви на грешки среди с гъвкаво управление на данните, без значение какъв е техния обем. Базата данни позволява дефиниране на собствени типове данни, създаване на персонализирани функции, както и изпълнение на код от различни езици за програмиране [43].

1.3.3. Визуализация

Визуализацията на данните, получени от извършените измервания, е ключова част при представянето им на потребителя [44]. Прегледният и логичен начин на представяне на данните може лесно и бързо да доведе до извършването на анализ и формирането на съответни изводи. По долу разгледаните технологии могат да се имплементират в потребителското приложение, с точно тази цел.

Grafana



Това е свободен за ползване софтуер, позволяващ автоматично изготвяне на графично обобщение на различни части от информация, която в конкретния случай ще се черпи от база данни. Grafana не изиска приемане на данни в бекенд хранилище или база данни на доставчик. Вместо това Grafana използва уникален подход за предоставяне на single plane of glass plane (единична еcranна плоскост), като обединява съществуващите ви данни, където и да са, и ги визуализира, както бъде настроена от потребителя [45].

Kibana



Kibana е инструмент, позволяващ анализиране на данни с визуализации. Софтуерът позволява използването на предварително конфигурирани табла за визуализация и управление на различни източници на данни, като ги внедрява в един потребителски интерфейс и дори насярчава "еластично търсене" на информация. Kibana е интегриран в множество продукти на пазара и позволява проследяване на натоварването на заявките и начина, по който заявките преминават през вашите приложения [46].

1.4. Механична конструкция

Конструкцията на проекта трябва да бъде здрава и стабилна, но освен това трябва да позволява движението на активните елементи, което предполага избраните материали да не бъдат твърде тежки.

1.4.1. Материали

Добра комбинация от здравина, твърдост и цена предлагат дървеният материал, по специално шперплат, както и вариациите на полимерни продукти, иначе казано пластмаси. В допълнение, споменатите материали са сравнително лесни за обработка и предлагат свобода при разработването на дизайна. От друга страна металът и изделията изградени от него предлагат по-висока здравина и позволяват по-прецизен дизайн. Като негови недостатъци могат да се считат по-високата му цена, както и факта, че е по-труден за обработка. [47].

1.4.2. Изработка със CNC фреза

CNC (Computer Numerical Control, или Компютърно Цифрово управление) фрезоването е производствена техника за ниско до средно по количество производство. Тя е доста често срещана техника за създаване на детайли. Процесът, който тези машини използват, е "отнемаш", тоест от един блок материал се извайва желаната форма, като по този начин се постига по-здрав продукт на цената на повече загубен материал [48].

1.4.3. Изработка с 3D принтер

3D принтирането е бързо развиваща се технология с много предимства пред традиционните производствени методи. То обаче има своите проблеми, включително че 3D печатането не може да се използва ефективно за масово производство и е ограничено по скорост и достъпност. Освен това тя използва "добавящ" процес при изработването на продукт, като така дава ненадмината гъвкавост при изработка на сложни елементи, но за сметка на това може работи само със специален полимерен материал. Изработените продукти са

по-крехки от тези, създадени със CNC машина, но не се хаби никакъв излишен материал [48].

1.4.4. Използване на готови конструктивни компоненти

Поради липсата на аналогични комерсиални проекти, както и осъдният набор от подобни прототипи, разработени от ентузиасти, готови детайли за конструкции, подходящи за конкретния проект, се срещат рядко.

1.4.5. Специално проектирана конструкция

За повечето елементи на механичната конструкцията на проекта се налага разработка на нов дизайн, или да се използва публично достъпен такъв. Проектът разгледан в точка 1.1.1., Solar Tracker V2.0, реализира добра конструкция и предлага свободни за използване файлове със скиците на използваните от тях конструктивни елементи.

Втора глава - Основни изисквания към системата и блокови схеми

2.1. Основни изисквания към проектираната система

В тази точка ще бъдат представени основните архитектурни, функционални и конструктивни изисквания към системата.

2.1.1. Архитектурни и комуникационни изисквания

- Основните компоненти на системата трябва да включват електронен блок за управление и комуникация, слънчев панел, сензорен блок, механична конструкция и захранващ блок.
- За реализиране на комуникацията трябва да се реализират подходящи сървър, мобилно приложение и комуникация между тях и управляващият блок на системата.

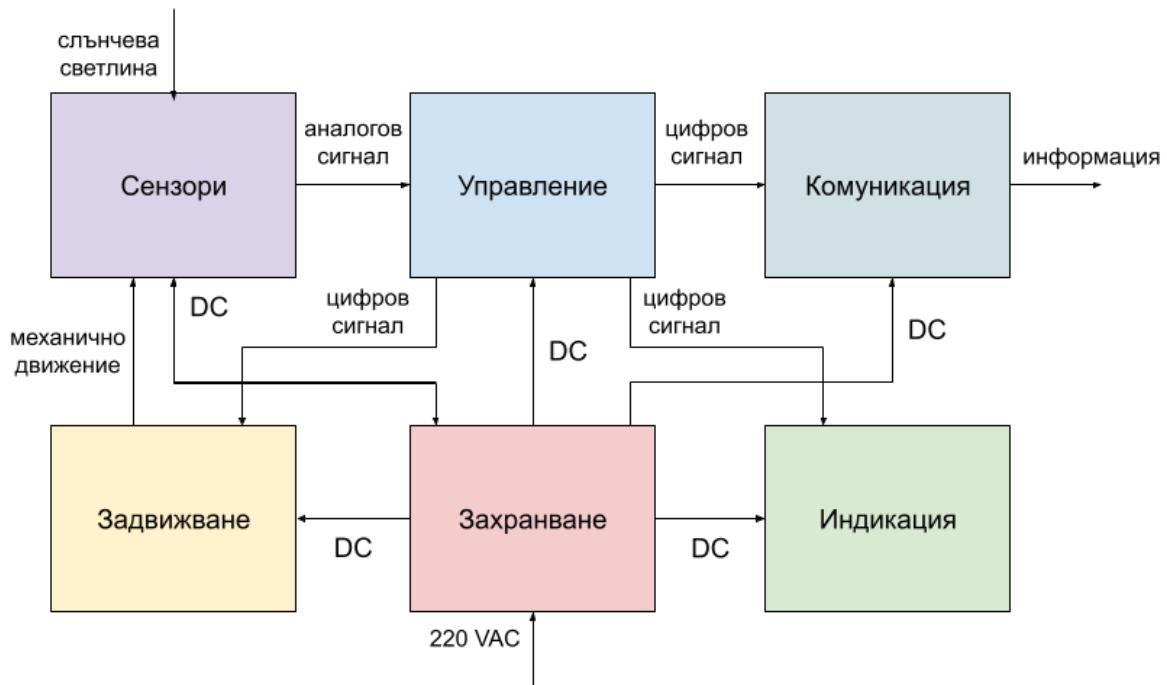
2.1.2. Функционални изисквания към системата

- Да определя посоката, от която падат слънчевите лъчи, и съответно да насочва слънчевия панел директно към слънцето.
- Да измерва параметрите на енергията получена с помощта на слънчев панел.
- Да може да обработва и визуализира информацията за измерванията в подходящ за потребителя вид.
- Да притежава мрежово захранване, както и резервно батерийно захранване в случай на отпадане на мрежовото.

2.1.3. Изисквания към конструкцията на системата

- Да осигурява стабилна основа за хардуерните компоненти и да не нарушава работата на комуникационния блок.
- Да позволява движение с две степени на свобода на слънчевия панел с цел насочването му към слънцето.

2.2. Обща блокова схема и функции на основните блокове



Фиг. 2.1. Обща блокова схема

На фигура 2.1. се вижда общата блокова схема и принципните връзките между различните блокове.

С помощта на Блок Сензори се подава информация за определяне на посоката на слънчевите лъчи, както и за енергията, преобразувана от слънчевия панел.

Функцията на Блок Задвижване е да насочва слънчевия панел в посоката, определена от сензорите на светлина.

Целта на Блок Индикация е да предоставя данни за работата на системата.

Чрез Блок Комуникация се осъществява връзката между Блок Управление и външната комуникационна мрежа.

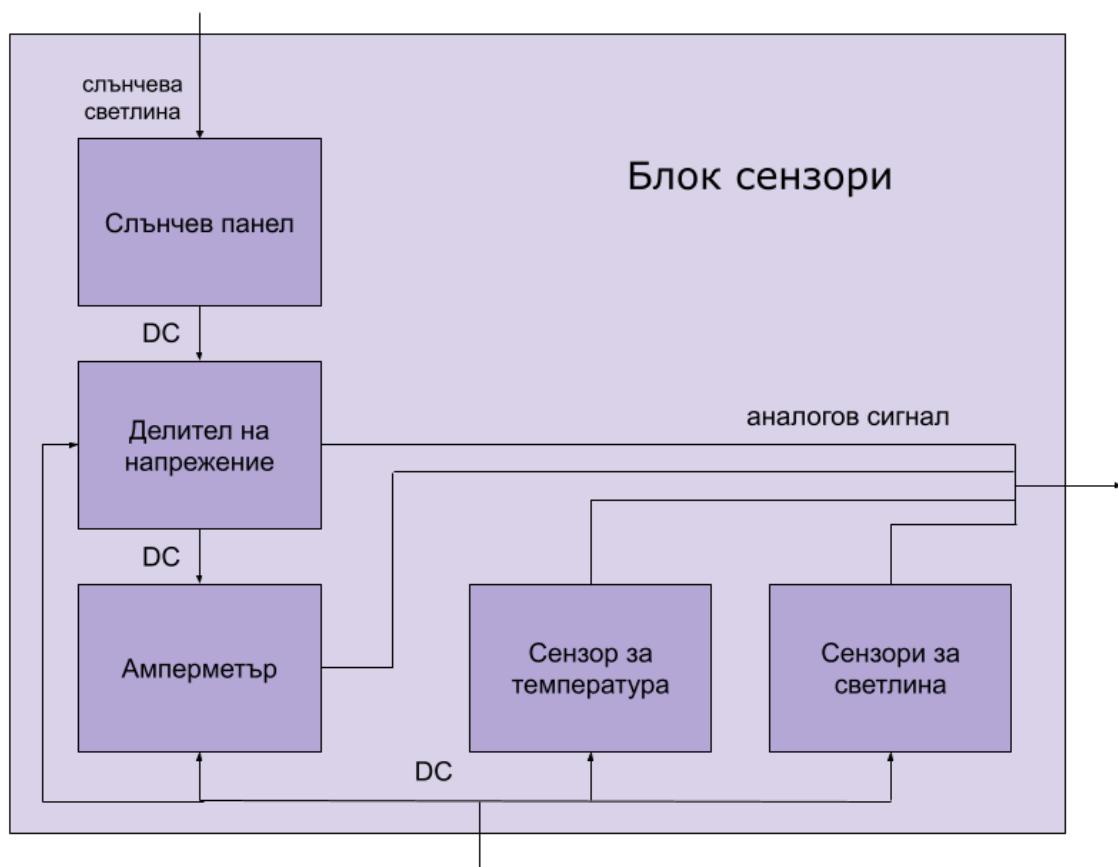
Блок Управление обработва получената от Блок Сензори информация и въз основа на алгоритъма му на работа управлява Блок Задвижване, Блок Комуникация и Блок Комуникация.

Основната функция на Блок Захранване е да предоставя стабилно електрическо захранване на останалите блокове и да съхранява енергията, получена от соларният панел.

2.3. Схеми и описание на отделните блокове

В настоящата точка са разгледани отделните подблокове, от които е съставен всеки блок в общата блокова схема Фиг. 2.1.

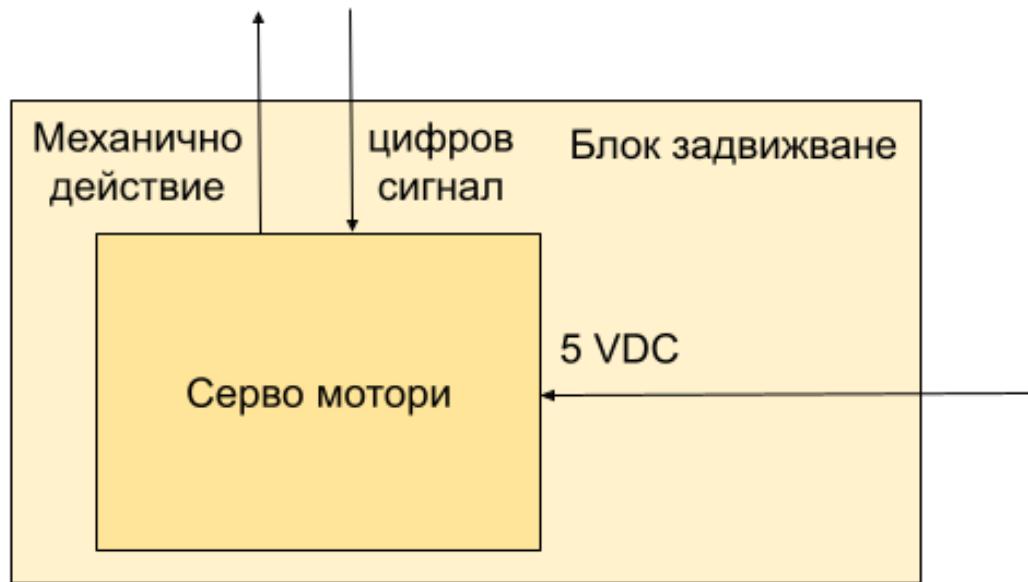
2.3.1. Блок Сензори



Фиг. 2.2. Блок Сензори

На фигура 2.2. е показан Блок Захранване и основните изграждащи го елементи. Основна роля играе слънчевият панел, като произвежданите от него напрежение и ток биват измерени чрез делител на напрежение и амперметър. Сензорът за температура е монтиран в непосредствена близост до слънчевият панел, като целта е да измерва неговата температура. Сензорите на светлина, които служат за определяне на посоката, от която падат слънчевите лъчи, съдържат делители на напрежение, състоящи се от нормален резистор и светлинно зависим резистор. Всички сензори комуникират с управляващия блок посредством аналогови сигнали.

2.3.2. Блок Задвижване

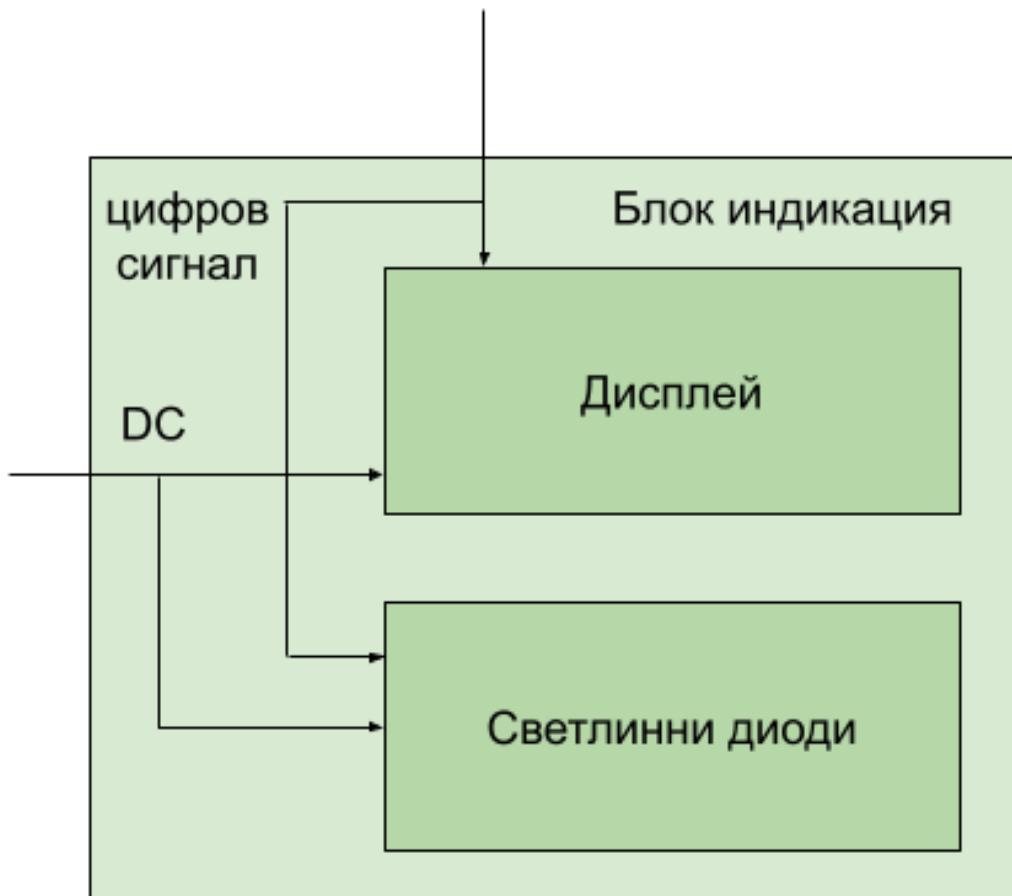


Фиг. 2.3. Блок Задвижване

Блок Задвижване е показан на горната фигура 2.3. Той се състои от два идентични серво мотора, получаващи захранване от захранващия

блок. Моторите са управлявани посредством цифрови сигнали, идващи от Блок Управление.

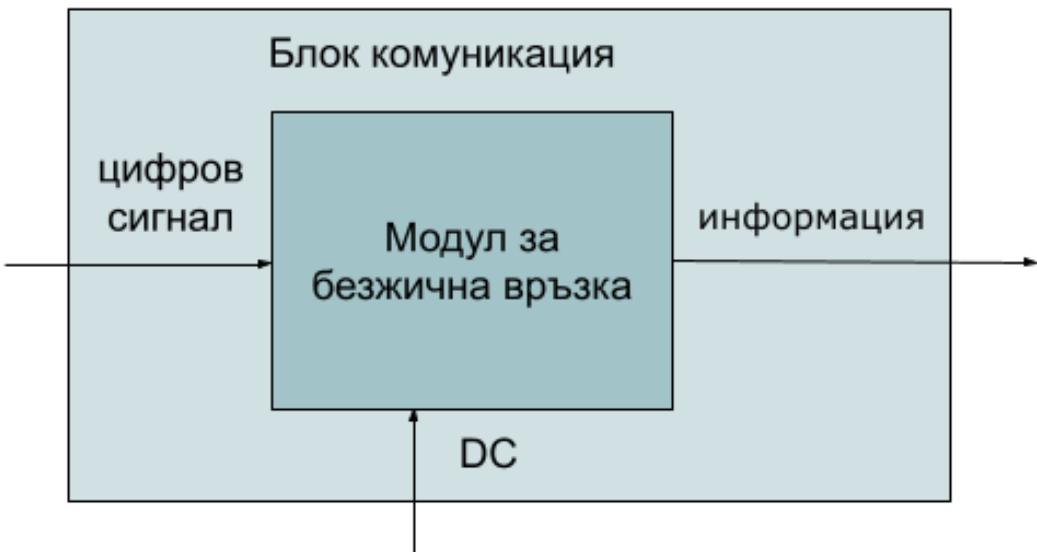
2.3.3. Блок Индикация



Фиг. 2.4. Блок Индикация

На фигура 2.4 е показана вътрешната структура на Блок Индикация. В него има два блока - блок дисплей, и блок светодиоди. И двата блока се захранват с 5 VDC, и се управляват от Блок Управление с цифрови сигнали.

2.3.4. Блок Комуникация



Фиг. 2.6. Блок Комуникация

Блок Комуникация представлява модул, позволяващ безжичен пренос на данни, интерпретиращ сигнали, получени от Блок Управление и предаващ ги на външна безжична комуникационна мрежа. Чрез нея данните от Блок Управление трябва да достигнат до сървър, на който те се съхраняват и визуализират, и който притежава интерфейс за достъп на потребители до него.

2.3.5. Блок Управление

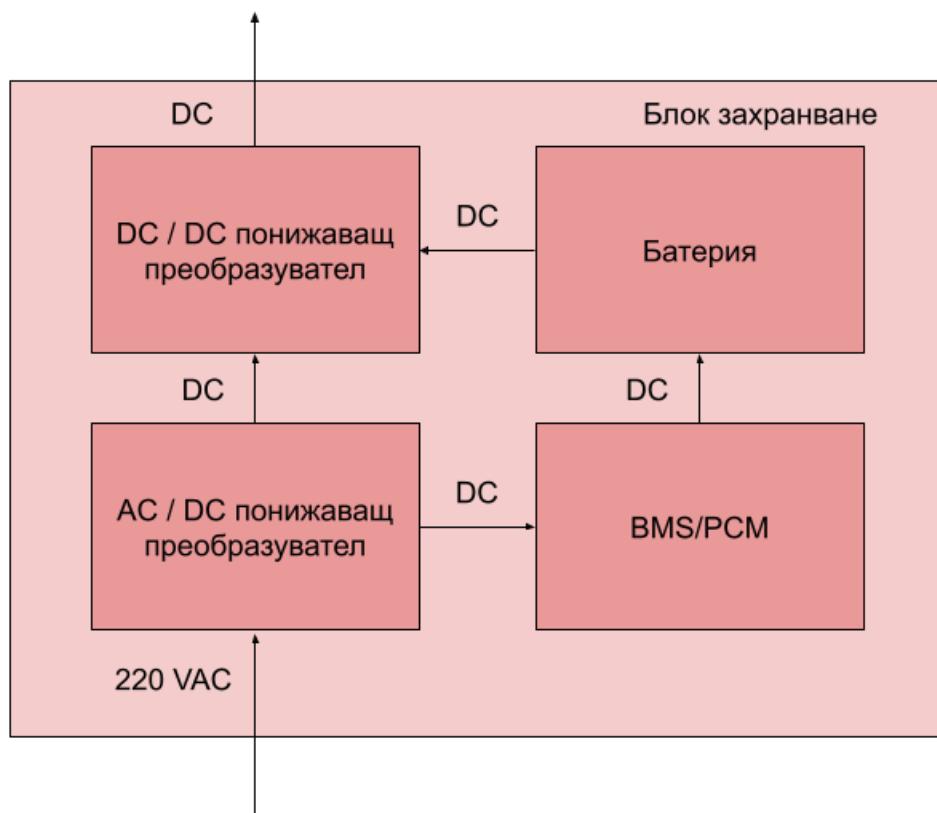


Фиг. 2.5. Блок Управление

На фигура 2.5. е показана структурата на Блок Управление, която съдържа опростена схема на микроконтролер със съответните му основни блокове. Показани са и различните по предназначение групи изводи. Захранващите изводи се свързват към Блок Захранване, за да осигурят източник на енергия за микроконтролера.

Аналого-цифровият преобразувател (АЦП) обработва сигналите от Блок Сензори. ШИМ (Широчинно Импулсна Модулация) изводите се използват за контрол на задвижващият блок. Цифровите изводи служат за управление на Блок Индикация. Микроконтролерът управлява цялата система в съответствие с получените външни сигнали и заложения алгоритъм за обработката им.

2.3.6. Блок Захранване



Фиг. 2.7. Блок Захранване

Основните елементи на Блок Захранване са AC/DC преобразувател от 220 VAC към 5 VDC, който ще може да зареди батерията при необходимост, Блок с MPPT функционалност, с помощта на който ще се извлича максимално количество енергия от соларния панел, и Блок Батерия, в която се съхранява енергията. Третият елемент представлява преобразувател на напрежение.

Трета глава - Проектиране на принципна електрическа схема

3.1. Избор на CAD система

CircuitMaker

Това е бесплатна софтуерна CAD система, предназначена за широката дизайнърска общност, предоставяща услуги за проектиране на електрически схеми печатни платки, както и за 3D визуализация. CircuitMaker PCB Design Editor има всички функции, необходими за проектиране на електрически схеми и печатни платки, без ограничение за броя на слоевете или площта на платката. Продуктът поддържа голяма база данни от компоненти и множество функции, улесняващи работния процес, като например push-and-shove маршрутизиране и лесно споделяне на проекти. [49]

3.2. Блок Сензори

3.2.1. Избор на компоненти

За вграждане в блок сензори са избрани слънчев панел модел CL-SM5P, модул за измерване на ток използваш сензор ACS712, аналогов температурен сензор KY-013, стандартни фоторезистори за измерване на разлики с осветеността с цел насочване на слънчевият панел директно към източника на светлина и делител на напрежение за измерване на напрежение извън пределите на вграденият АДЦ на микроконтролера.

Слънчев панел CL-SM5P

След сравнение на продуктите в точка 1.2.1., може да се направи извода, че соларен панел CL-SM5P е най-подходящ за проекта поради това, че предлага най-висока ефективност, най-добър баланс между тегло и структурна здравина, по средата е по отношение на тегло и

цена спрямо останалите два панела и е лесен за набавяне вземайки предвид, че се предлага от надежден доставчик.

Тип панел: поликристален

Системно напрежение: 18.2 V

Мощност: 5 W

Максимален ток: 0.28 A

Габаритни размери: 251x186x18 mm

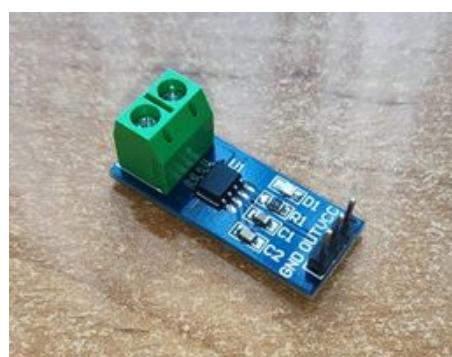
Сензор за ток ACS712

ACS712 представлява икономичен и прецизен амперметър за измерване на променлив или постоянен ток в промишлеността, търговски и комуникационни системи. Устройството се състои от прецизен, линеен сензор използваш ефекта на Хол, с ниско изместване, образуващ верига с меден проводящ път, разположен близо до повърхност на матрицата. Минаващият ток, протичащ през тази верига генерира магнитно поле, което се усеща от интегрираният сензор и го преобразувана в пропорционално напрежение, което се подава на комуникационния изход. [30]

Работно напрежение: 5V

Диапазон на измерване: 0-25 VDC

Точност на измерване $\pm 1,5\%$



Фиг. 3.2. Изглед към конкретно използвани амперметър ACS712

Делител на напрежение

Делителят на напрежението е проста схема, която превръща голямо напрежение в по-малко. Използвайки само два последователни резистора и входно напрежение, можем да създадем изходно напрежение, което е малка част от входното. Делителите на напрежението са едни от най-основните схеми в електрониката. Добре направен делител на напрежение консумира пренебрежимо количество енергия. По този начин ще бъде удобно да се измери напрежението на изхода на слънчевият панел. В конкретния случай ще бъде използван делител на напрежение състоящ се от два резистора със съпротивления съответно $10\text{ k}\Omega$ и $100\text{ k}\Omega$ в съотношение 1 към 10 за да се позволи на вграденият АДЦ на микроконтролера, приемащ стойности до 3.3V да чете стойности до съответно 36V.

Аналогов температурен сензор KY-013

Аналоговият температурен сензор KY-013 може да измерва температурата на околната среда въз основа на съпротивлението на термистора на платката. Съвместим е с популярни електронни платформи като Arduino и ESP32. Този модул се състои от NTC (Negative Temperature Coefficient) термистор, $10\text{ k}\Omega$ резистор и 3 мъжки щифта.
[51]

Работно напрежение: 5V

Температурен диапазон на измерване: -55°C до 125°C

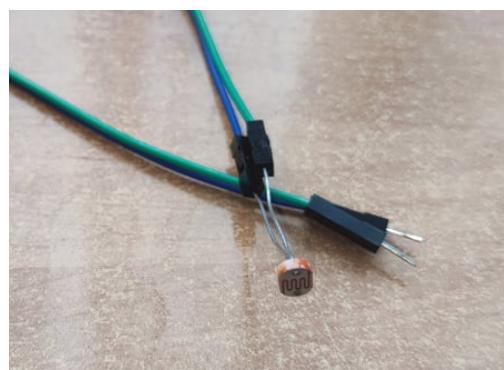
Точност на измерване $\pm 0,5^{\circ}\text{C}$



Фиг. 3.3. Изглед към конкретно използваният температурен сензор
KY-013

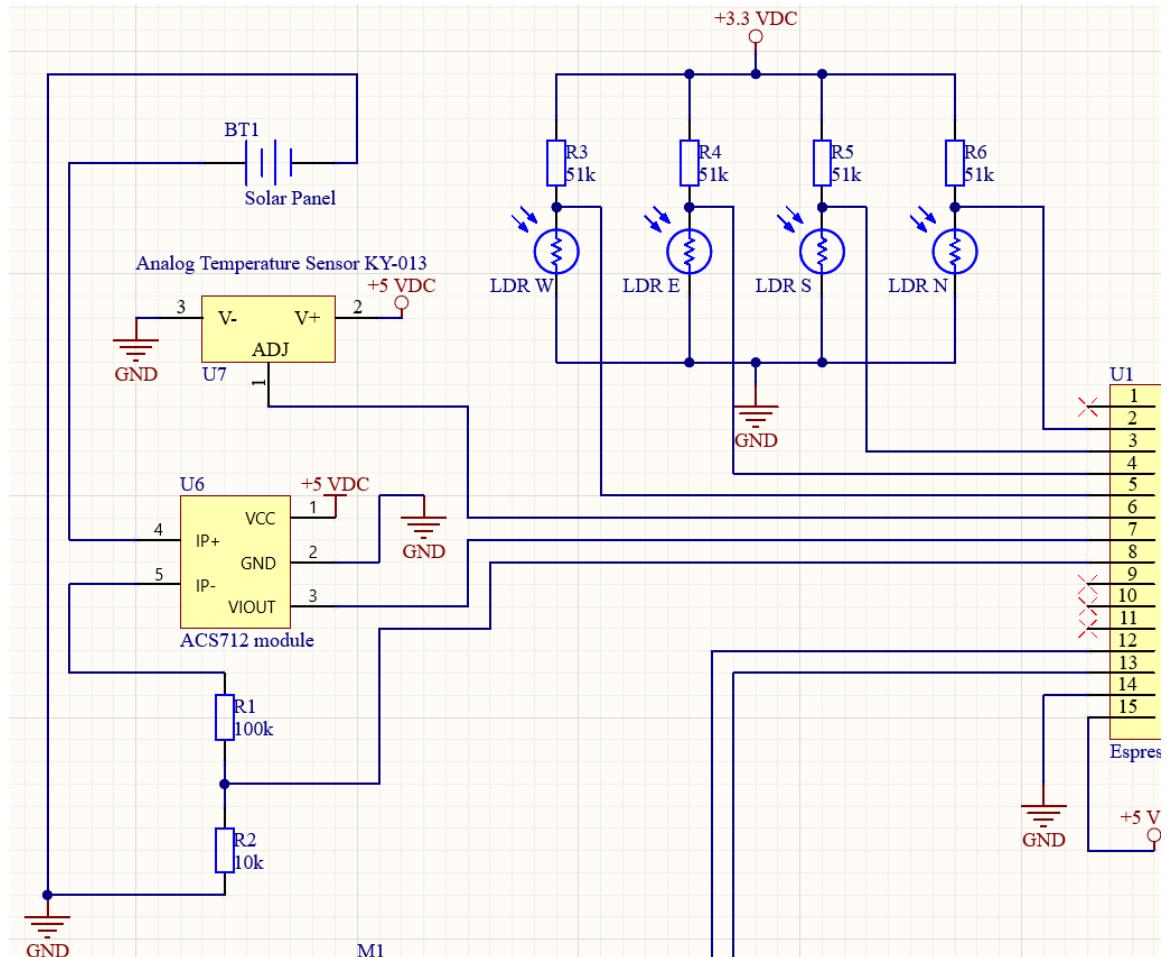
Фоторезистори

На тъмно фоторезисторът може да има съпротивление до няколко мегаома, докато на светлина той може да има съпротивление до няколкостотин ома. Ако падащата светлина върху фоторезистор надвиши определена честота, фотоните, погълнати от полупроводника, дават на свързаните електрони достатъчно енергия, за да скочат в лентата на проводимост. Получените свободни електрони (и техните дупки партньори) провеждат електричество, като по този начин намаляват съпротивлението. Диапазонът на съпротивлението и чувствителността на фоторезистора могат да се различават значително при различните компоненти. За измерване на съпротивлението на фоторезистор (съответно и нивото на осветеност) може да се използват обикновен резистор с познато съпротивление, формиращ делител на напрежение със фоторезистора в комбинация със микроконтролер с АЦП, който да чете променящият се пад на напрежение върху схемата.



Фиг. 3.4. Изглед към конкретно използван фоторезистор

3.2.2. Принципна електрическа схема на Блок сензори



Фиг. 3.5. Принципна електрическа схема на Блок Сензори

На показаната по-горе фигура 3.5 може да се видят последователно свързаните слънчев панел, сензор за ток и делител на напрежение, позволяващи измерването на електрическите параметри на фотоволтаика. Температурният сензор е свързан самостоятелно. Горе в дясното се вижда и блокът от делители на напрежение, формирани от обикновени резистори и фоторезистори, чиято цел е определяне на релативният ъгъл, под който падат светлинните лъчи.

3.3. Блок Задвижване

3.3.1. Избор на компоненти

Серво мотори от тип MG996R са избрани за задвижване, заради техният висок въртящ момент и достъпност.

Серво MG996R

Това цифрово servo с висок въртящ момент разполага с метална предавка, което води до допълнително висок въртящ момент от 10 кг в малка опаковка. MG996R е по същество подобрена версия на известен servo MG995 и разполага с подобрена удароустойчивост и преработена печатна платка и IC система за управление, което го прави много по-точен от предшественика си. Предавката и мотора също са подобрени, за да подобрят мъртвата честотна лента и центрирането.

Уредът идва в комплект с 30 см проводник и 3-пинов 'S' тип женски конектор, който пасва на повечето приемници. Това стандартно servo с висок въртящ момент може да се върти на приблизително 120 градуса (60 във всяка посока). Можете да използвате всеки servo код, хардуер или библиотека, за да управлявате тези servo, така че е чудесно за начинаещи. Серво MG996R също идва със стандартна селекция от пластмасови приставки позволяващи известна гъвкавост при проектиране на конструкцията. [18]

Тегло: 55 гр

Размери: 40,7 x 19,7 x 42,9 mm

Въртящ момент: 9,4 kgf·cm (4,8 V), 11 kgf·cm (6 V)

Работна скорост: 0,17 s/60° (4,8 V), 0,14 s/60° (6 V)

Работно напрежение: 4,8 V до 7,2 V

Работен ток 500 mA – 900 mA (6V)

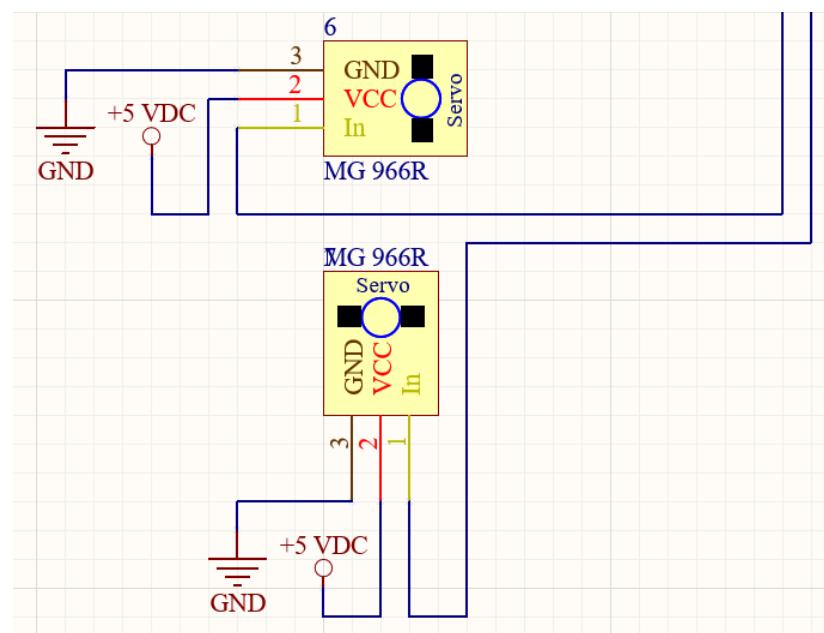
Температурен диапазон: 0 °C – 55 °C

ШИМ дължина на вълната: 20ms



Фиг. 3.6. Изглед към конкретно използван серво мотор MG996R

3.3.2. Принципна електрическа схема на Блок задвижване



Фиг. 3.7. Принципна електрическа схема на Блок Задвижване

На фигура 3.7 са показани двата сервомотора образуващи блок задвижване. Моторите са захранени и проводници служещи за тяхното управление са прокарани до микроконтролера.

3.4. Блок Индикация

3.4.1. Избор на компоненти

За индикация са избрани комбинация от светодиоди и малък LCD. За да се опрости схемата е избран и I²C адаптер за экрана.

Светодиоди (LED) 3mm

Светодиодът е полупроводниково устройство, което излъчва светлина, когато през него преминава електрически ток. Светлината се произвежда, когато частиците, които носят тока (известни като електрони и дупки), се комбинират заедно в полупроводниковия материал. те предлагат сравнително ярък източник на светлина с ниска консумация и са много достъпни.

Конкрето избраните светодиоди не могат да бъдат директно свързани към източник на захранване от порядъка на 3.3 V. Нужно е резистор да бъде свързан последователно към всеки един от тях. Съпротивлението на тези се резистори се пресмята по формулата:

$$R = \frac{U_{max} - U_{led}}{I_{led}}$$

Където R е неизвестното съпротивление на резистора, което търсим.

U max е напрежението между извод 3.3V на и извод 18 на микроконтролера, което е 3.3V.

I led за червен 3mm светодиод е 18 mA стандартно.

U led за червен 3mm светодиод е 1.8V стандартно.

I led за зелен 3mm светодиод е 20 mA стандартно.

U led за зелен 3mm светодиод е 2.2V стандартно.

Използвайки посочената формула получаваме:

R червен светодиод = 83Ω , като най-близка стандартна стойност е 82.5Ω .

R зелен светодиод = 55Ω , като най-близка стандартна стойност е 54.9Ω .



Фиг. 3.8. Изглед към конкретно използваните светодиоди

LCD 1602

LCD1602 или течнокристален дисплей с 1602 символа е вид точков матричен модул за показване на букви, цифри и знаци и т.н. Състои се от матрични позиции 5×7 . Всяка позиция може да показва един знак. Има разстояние от точка между два знака и интервал между редовете, като по този начин се разделят знаците и редовете. Модел 1602 означава, че показва 2 реда от 16 знака. [26]

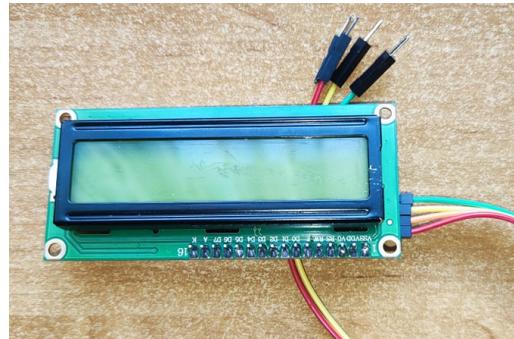
Формат на дисплея: 16 знака x 2 реда

Входни данни: наличен 4-битов или 8-битов интерфейс

Шрифт на дисплея: 5 x 8 точки

Захранване: Единично захранване ($5V \pm 10\%$)

Схема на управление: 1/16 Duty, 1/5 Bias



Фиг. 3.9. Изглед към конкретно използваният LCD 1602

I²C LCD адаптер

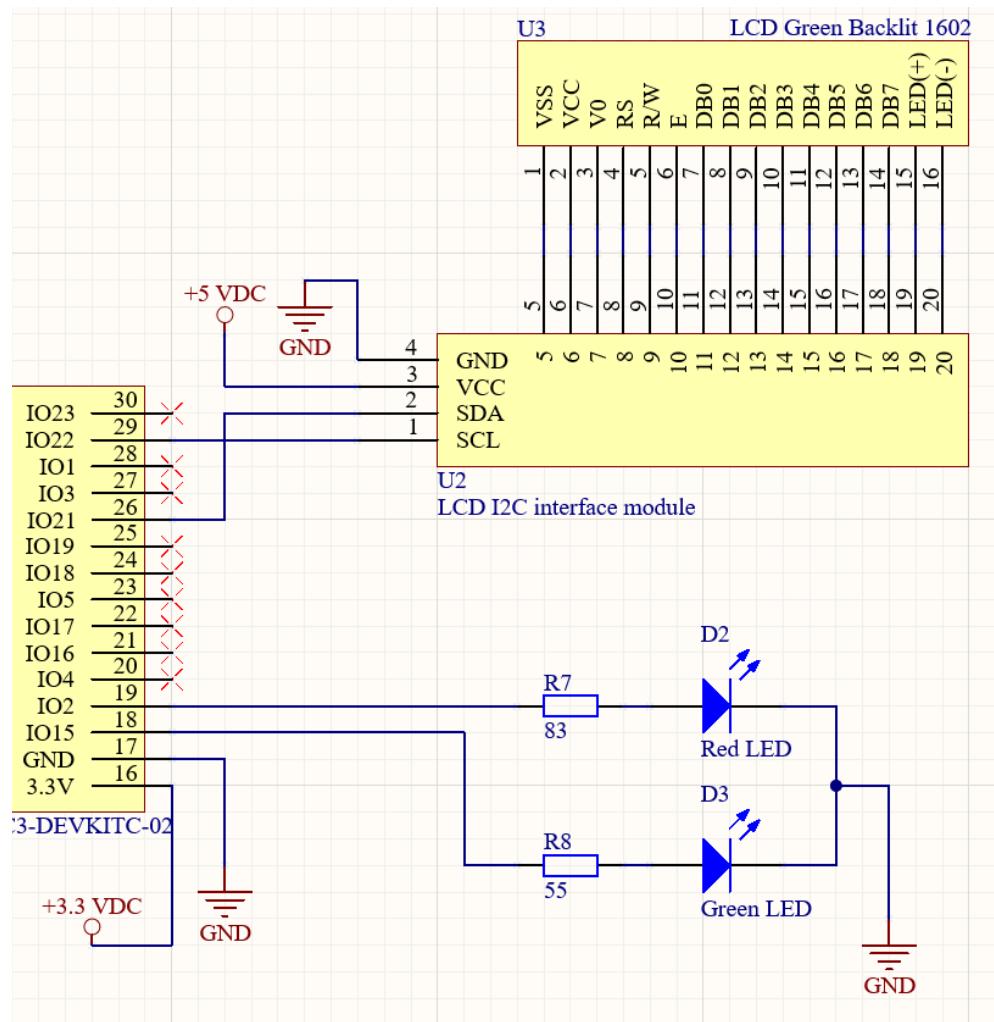
I²C (Inter-Integrated Circuit) LCD адаптер е устройство, съдържащо микроконтролер PCF8574 чип. Този микроконтролер е I/O разширител, който комуникира с друг чип на микроконтролера с двупроводен комуникационен протокол. Използвайки този адаптер, всеки може да управлява 16x2 LCD само с два проводника (SDA, SCL), което спестява много щифтове на използваният микроконтролер. Модулът същ има вграден потенциометър за контрол на контраста на LCD дисплея. I²C (между интегрална връзка) е сериен, цифров протокол за пренос на информация. Той използва връзка тип господар-подчинен (master-slave), което го прави полуудуплексен. I²C предлага лесен и надежден начин за еднопосочна комуникация между вградени модули и други. [51]

Захранващо напрежение: 2.5 - 6V DC;



Фиг. 3.10. Изглед към конкретно използваният I²C LCD адаптер

3.4.2. Принципна електрическа схема на Блок Индикация



Фиг. 3.11. Принципна електрическа схема на Блок Индикация

На фигура 3.11 се виждат използваните LCD и I²C адаптер, който го захранва и контролира, както и двата диода контролирани директно от микроконтролера.

3.5. Блок Управление

3.5.1. Избор на компоненти

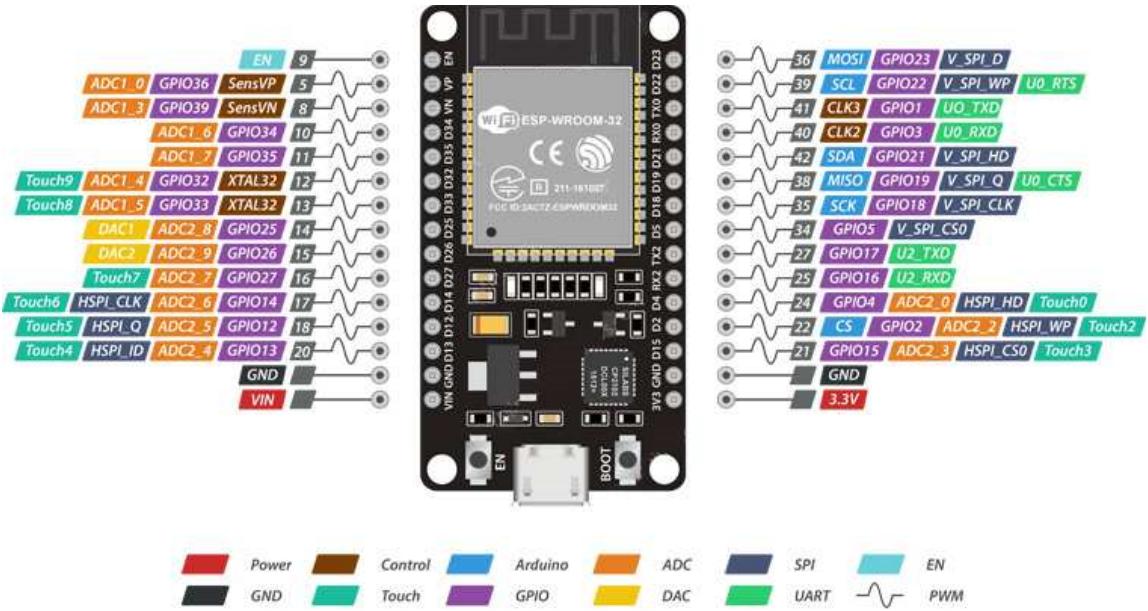
За управляваща платформа е избран ESP-Wroom-32 devkit, заради многобройните си изводи с вграден АДЦ и вграден WiFi комуникационен модул.

ESP-Wroom-32 devkit

ESP32-WROOM-32 е мощен, общ Wi-Fi+BT+BLE микроконтролерен модул, който е насочен към голямо разнообразие от приложения, вариращи от сензорни мрежи с ниска мощност до най-взискателните задачи, като гласово кодиране, стрийминг на музика и MP3 декодиране. В основата на този модул е чипът ESP32-D0WDQ6. Вграденият чип е проектиран да бъде мащабируем и адаптивен. Има две ядра на процесора, които могат да се управляват индивидуално, а тактовата честота на процесора е регулируема от 80 MHz до 240 MHz. Чипът има и копроцесор с ниска мощност, който може да се използва вместо процесора за пестене на енергия, при изпълнение на задачи, които не изискват много изчислителна мощност, като наблюдение на периферни устройства. ESP32 интегрира богат набор от периферни устройства, вариращи от капацитивни сензори за докосване, сензори на Хол, SD интерфейс на картата, Ethernet, високоскоростен SPI, UART, I²S и I²C.

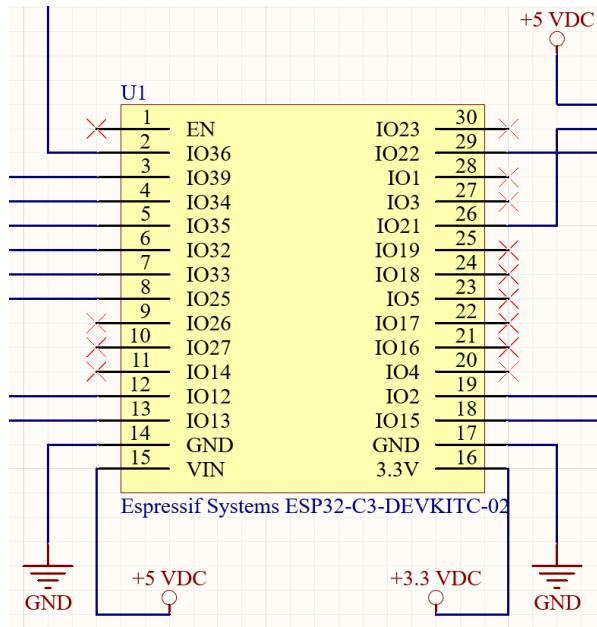


Фиг. 3.12. Изглед към конкретно използваният ESP-Wroom-32



Фиг. 3.13. Обозначения и функции на изводите на ESP-Wroom-32

3.5.2. Принципна електрическа схема на Блок Управление



Фиг. 3.14. Принципна електрическа схема на Блок Управление

На фигурата 3.14 показана горе може да се види схематичният модел на микроконтролер ESP32 Wroom devkit.

3.6. Блок Комуникация

Относно комуникационният блок, тряба да се отбележи, че избраният микроконтролер ESP-Wroom-32 разполага с вградена антена и поддържа безжична комуникация чрез протоколите Bluetooth версия 4.2 и WiFi 802.11n. Разработени са множество публично достъпни софтуерни библиотеки за лесена инициализация и настройка на преноса на данни от и към микроконтролерната платформа.

3.7. Блок Захранване

3.7.1. Избор на компоненти

За хибридното захранване са избрани три батерии тип 18650, поради надеждността, широкото разпространение и факта, че при последователно свързване образуват верига с общо напрежение 12V, което позволява да бъдат зареждани с комбинация от защитна управляваща платка и преобразувател на напрежение 220 VAC / 12 VDC. При този подход се налага и използването на понижаващ преобразувател на напрежение 12 VDC / 5 VDC, за да се осигури подходящо работно напрежение на останалите компоненти в схемата.

Батерии 18650

Батерията 18650 е литиево-йонна батерия. Името произлиза от специфичните размери на батерията: 18mm x 65mm. Батерията 18650 има напрежение 3.7V, но то може да варира между 2.8V и 4.2V в зависимост от това какъв процент от заряда ѝ е наличен, и има капацитет между 2200mAh и 3500mAh. Тези батерии се използват във фенерчета, лаптопи, електроника и дори някои електрически автомобили поради тяхната надеждност, дълго време на работа, способност да се презареждат стотици пъти. Тези батерии са относително достъпни и множество спомагателни модули и материали са налични онлайн.

Номинален капацитет 2200 mAh (0.2 C (440mA), до 2.75 V разряд)
Минимално гарантиран капацитет 2100 mAh (0.2C (420mA), до 2.75 V разряд)
Напрежение на заряд: 4.2 ±0.05 V
Номинално напрежение: 3.67V
Метод на заряд CC-CV (с контрол по ток и напрежение)
Заряден ток (номинален ток на зареждане): 1100 mA
Максимален ток на зареждане : 2200 mA
Време за заряд (стандартно): 3 часа
Бърз заряд: 2.5 часа
Максимален ток на разряд 4400mA (при 25 °C)
Критично ниво на разряд до 2.75 V
Тегло 44.0 гр.
Дължина: 65.00 mm
Диаметър : 18.40 mm



Фиг. 3.15. Изглед към конкретно използваните батерии 18650

BMS/PCM модул (*Battery Management System / Protection Circuit module*)

Конкретно използваният модул се използва за зареждане/разреждане на литиеви батерии свързани паралелно. Той може да издържи непрекъснат ток на зареждане 20A, непрекъснат ток на разреждане

40A, моментен ток под 80A. Първото ниво на контрол и защита, което модулът предлага е балансиране и оптимизиране на напреженията на клетките, с което гарантира, че при зареждане на две клетки с леко разминаващи се параметри няма да се стигне до твърде голямо презареждане на едната батерия. Второто ниво на управлението е модул на защитна верига (PCM), който защитава клетки за високи/ниски напрежения и токове по време на зареждане и разреждане.

Напрежение на зареждане: 12.6V-13V

Максимален работен ток: 60A (условията за разсейване на топлината са много добри)

Продължителен ток на зареждане (горна граница): 20A

Продължителен разряден ток (горна граница): 40

Зашита от свръхток: 125A

Откриване на презареждане: 12.8V (за 3 групи батерии)

Освобождаване на презареждане: 12,3V (за 3 групи батерии)

Откриване на свръх разреждане: 2,4V (за 1 група батерии)

Освобождаване на свръх разреждане: 3.0V (за 1 група батерии)

Зашита от късо съединение: Да

Работна температура: -40°C ~ +85°C

Условия на съхранение: -40°C ~ +125°C



Фиг. 3.16. Изглед към конкретно използваният BMS/PCM модул

AC/DC преобразувател

Използван е мултифункционален преобразувател на напрежение от 220 VAC към 12 VDC със букса тип 2.1mm X 5.5mm. Преобразувателят поддържа изходен ток до 3 A, което е достатъчно за проекта.



Фиг. 3.17. Изглед към конкретно използваният BMS/PCM модул

DC/DC преобразувател LM2596

Регулаторите от серията LM2596 са монолитни интегрални схеми, които осигуряват всички активни функции за понижаващ (buck) превключващ регулатор, способен да поддържа товар от 3 A, което съвпада с избраният AC/DC преобразувател на напрежение по-горе. Тези устройства се предлагат с регулируемо изходно напрежение. Те изискват минимален брой външни компоненти, лесни за използване и включват вътрешна компенсация на честотата и осцилатор с фиксирана честота. Серията LM2596 работи с честота на превключване от 150 kHz, което позволява филтър с по-малък размер компоненти от това, което би било необходимо с превключващи регулатори с по-ниска честота. [53]

Входно напрежение: 3,2 ÷ 40V DC;

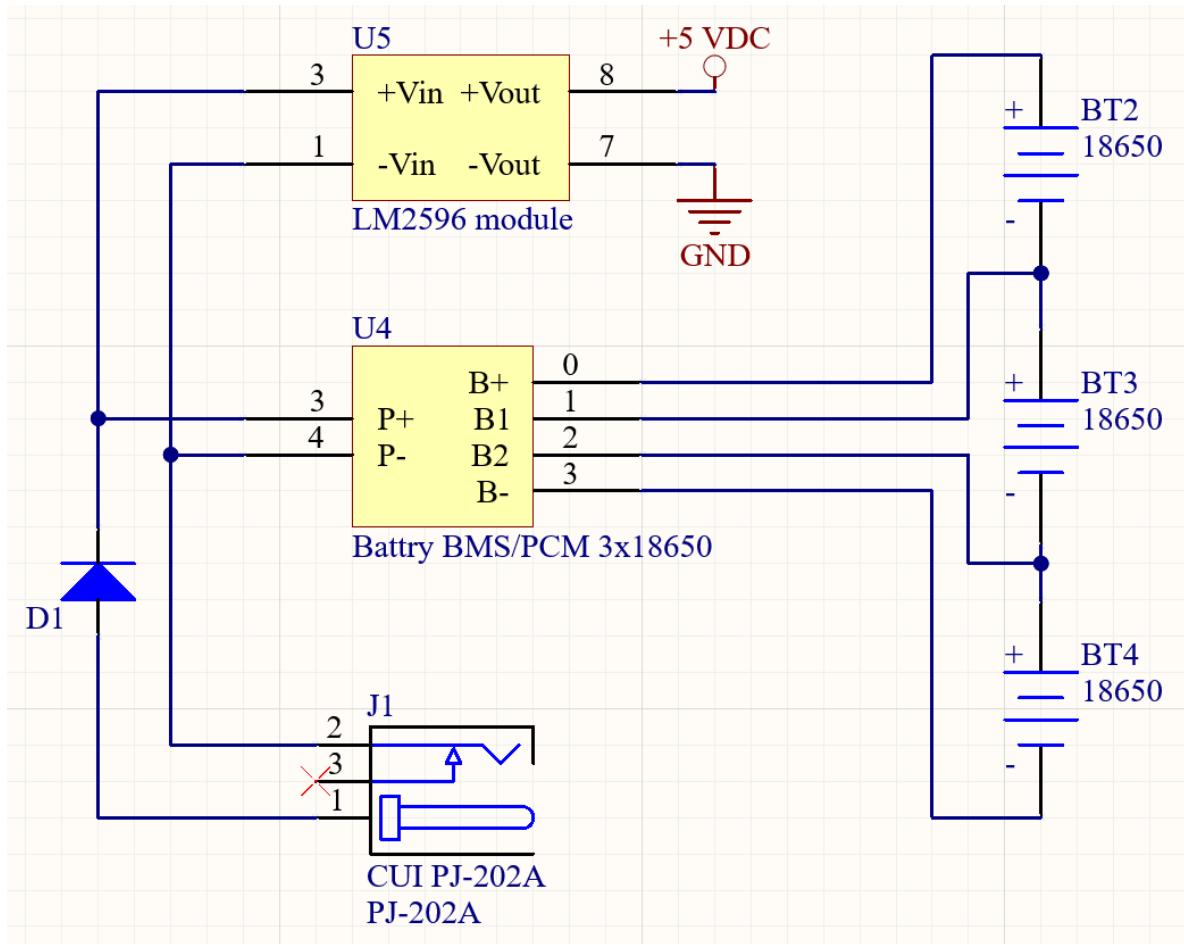
Изходно напрежение: 1,25 ÷ 35V;

Максимален изходен ток: 3A(максимален товар 20W);



Фиг. 3.18. Изглед към конкретно използваният LM2596

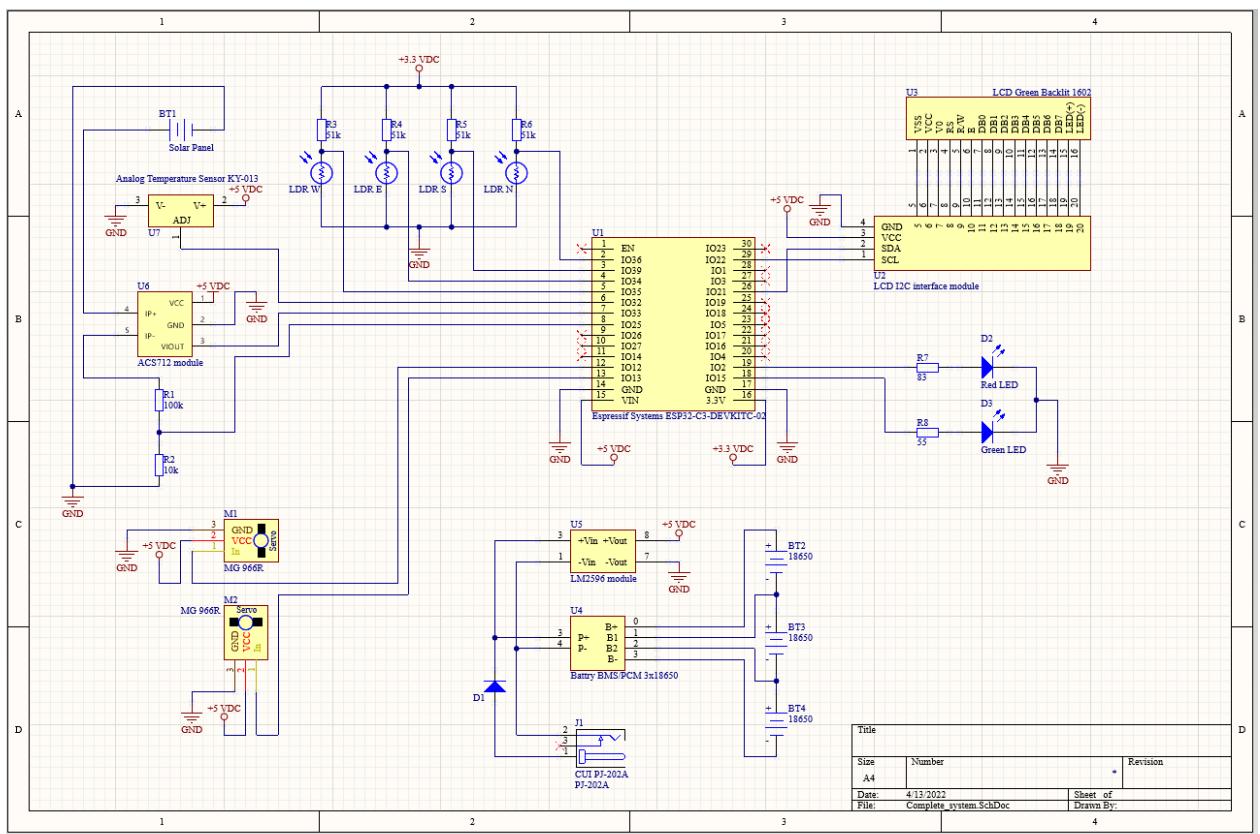
3.7.2. Принципна електрическа схема на Блок Захранване



Фиг. 3.19. Принципна електрическа схема на Блок Захранване

На фигура 3.19 вдясно са показани батериите, свързани последовательно за да предоставят приблизително 12VDC. Те са свързани с BSM/PCM модула. Женски 5.5 x 2.1 mm barrel порт е добавен за да се включва външното 12-волтово захранване и е добавен диод за защита на веригата. Заедно батериите и захранването са свързани към понижващия преобразувател на напрежение който превръща 12VDC в подходящи за системата 5VDC.

3.8. Обща принципна електрическа схема



Фиг. 3.20. Обща принципна електрическа схема

На фигура 3.20 е показана цялата принципна електрическа схема на проекта. Всеки блок е свързан към централната микроконтролерна платформа.

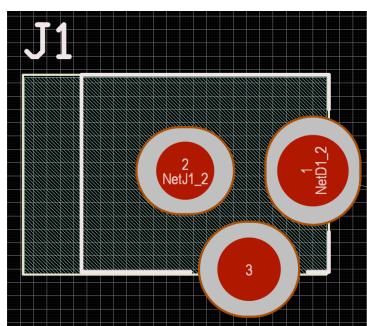
Четвърта глава - Печатни платки

В тази глава ще бъде разгледан процеса на проектиране на печатната платка, както и основните му характеристики и етапи на изпълнение.

4.1. Корпуси на използваните компоненти

По-долу са разгледани корпусите на използваните компоненти, техните размери и разположение на изводите им.

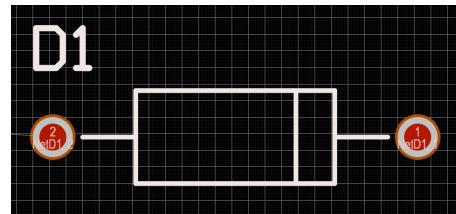
4.1.1. Корпус на женски 12V Barrel Jack 2.1 x 5.5 mm



Фиг. 4.1. Изображение на корпуса на Женски 12V цилиндричен конектор 2.1 x 5.5 mm

На фиг. 4.1 е показан корпуса на използваният в системата 12V конектор. Извод 1 е използван като "топъл" проводник, тоест той е свързан към положителния извод на външният захранващ преобразувател. Извод 2 е свързан към земята и отрицателният извод на външното захранване. Извод 3 може да се използва за определяне на това дали в конектора е вкаран мъжки такъв, но в конкретния случай той не се използва. Допълнително предимство на този извод е, че придава структурна здравина на закрепването на компонента, който на практиката е подложен на натоварване при всяко включване и изключване.

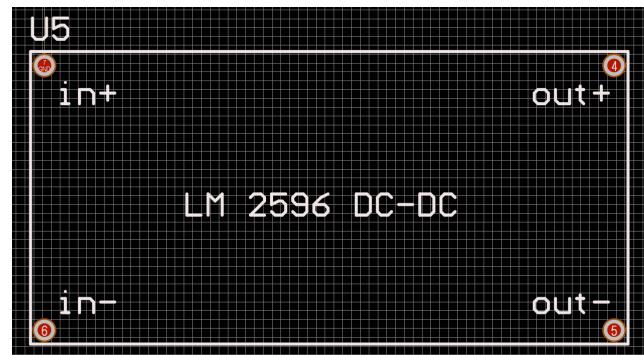
4.1.2. Корпус на диод



Фиг. 4.2. Изображение на корпуса на защитния диода

На фигура 4.2 е показан корпуса използваният диод. извод едно представлява катода, а извод 2 е анода на диода.

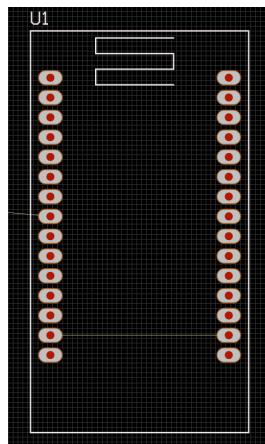
4.1.3. Корпус на DC-DC преобразувател LM 2596



Фиг. 4.3. Изображение на корпуса на DC-DC преобразувателя LM 2596

Корпусът на модула, преобразувател на напрежение LM 2596 е показан на фиг. 4.3. В горния ляв ъгъл е положителния вход, приемащ +12VDC. В горен десен ъгъл е положителният изход, предоставящ +5VDC. Долните изводи са заземени.

4.1.4. Корпус на ESP32 Wroom devkit



Фиг. 4.4. Изображение на корпуса на ESP32 Wroom devkit

На фигура 4.4 е показан корпуса на конкретно използваният микроконтролер. Способностите на изводите на ESP32 и връзките с останалите компоненти са разгледани по-обширно в глава трета.

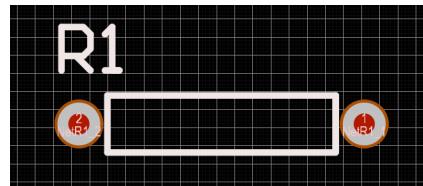
4.1.5. Корпус на ACS712 модул



Фиг. 4.5. Изображение на корпуса на ACS712 модул

На фигура 4.5 е показан корпуса на модул за измерване на ток със сензор ACS712. Горният десен извод се използва за захранване на сензора, долният десен е земя, средният се използва за предаване на информация. В ляво са изводите, към които се свързва товарът, който трябва да бъде измерен.

4.1.6. Корпуси на резистори



Фиг. 4.6. Изображение на корпуса на резистор

На фигура 4.6 е показан корпуса на един от използваните резистори.

4.2. Проектиране на печатната платка

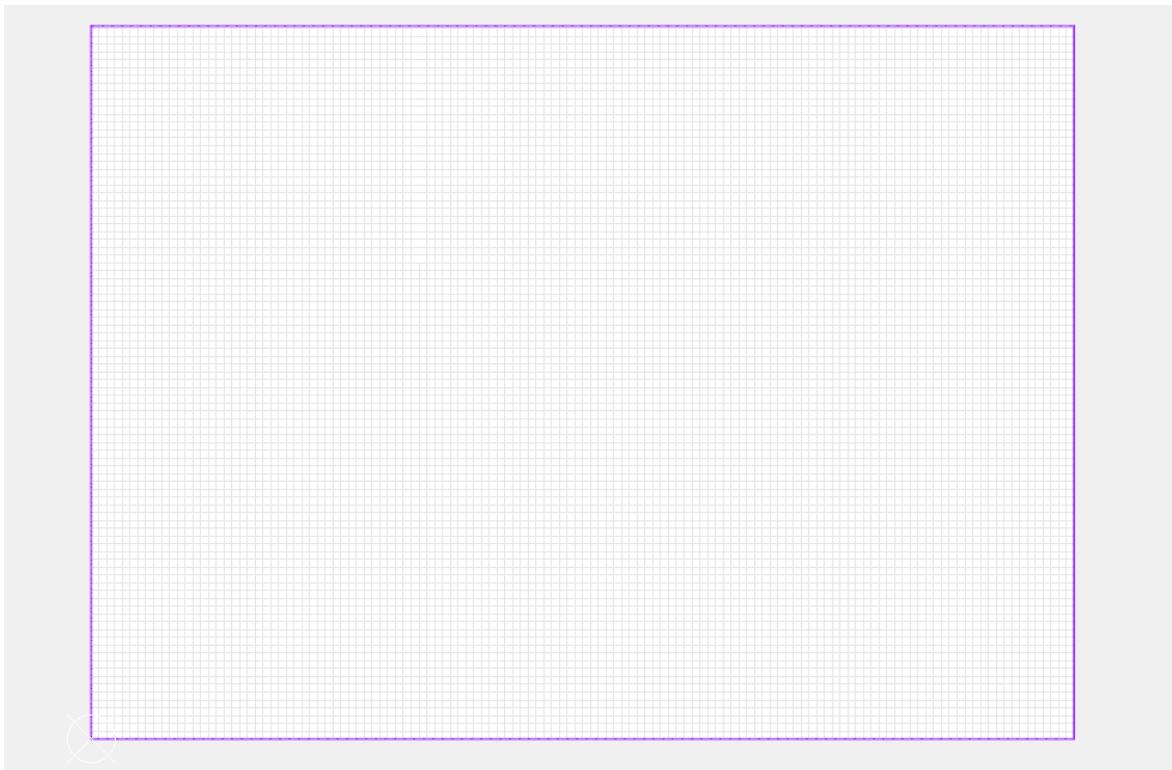
В тази точка предстои да бъдат разгледани отделните слоеве на проектираната печатна платка.

Основни изисквания и параметри на проектираната печатна платка

Проектираната платка трябва да:

- съответства на разработените в трета глава принципни електрически схеми;
- е реализирана с използване на ТНО (Through-hole) компоненти, за да се улесни монтажа;
- бъде с подходящи за конструкцията и използвани компоненти размери, да бъде рационално проектирана и да има отвори за закрепване;
- бъде възможно най-компактна, за да се намали обема на конструкцията и цената на производство;
- бъде еднослойна, за да се улесни производството и намали цената;
- отделните захранващи и сигнални писти трябва да са достатъчно широки, но не твърде големи;

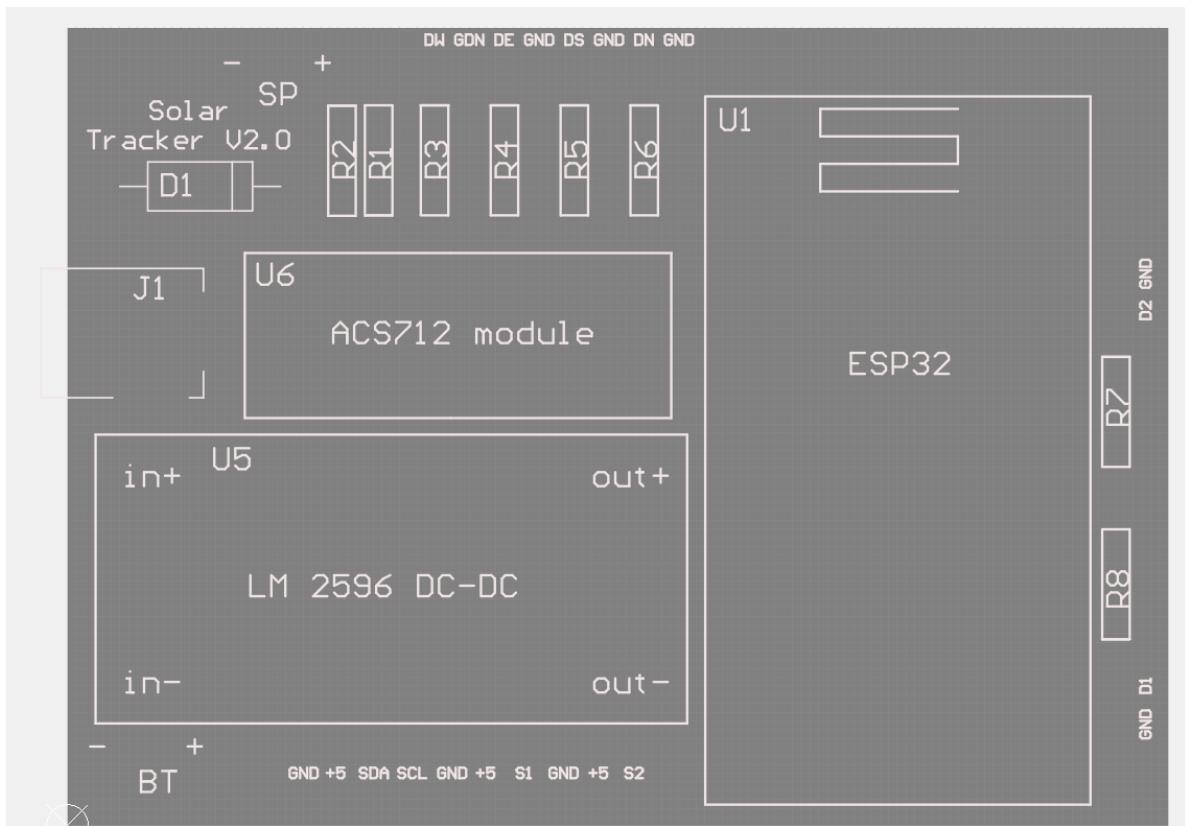
4.2.1. Слой Keep-Out



Фиг. 4.7. Изглед към печатната платка, показващ Keep-Out слой

На този слой са показани очертанията на платката, тоест формата на текстолитната плоскост, която ще представлява крайният продукт. Размерите на печатната платка са сведени до точните числа 80mm x 58mm, за да се избегнат грешки при закръгляне.

4.2.2. Слой Компоненти (Бял печат)



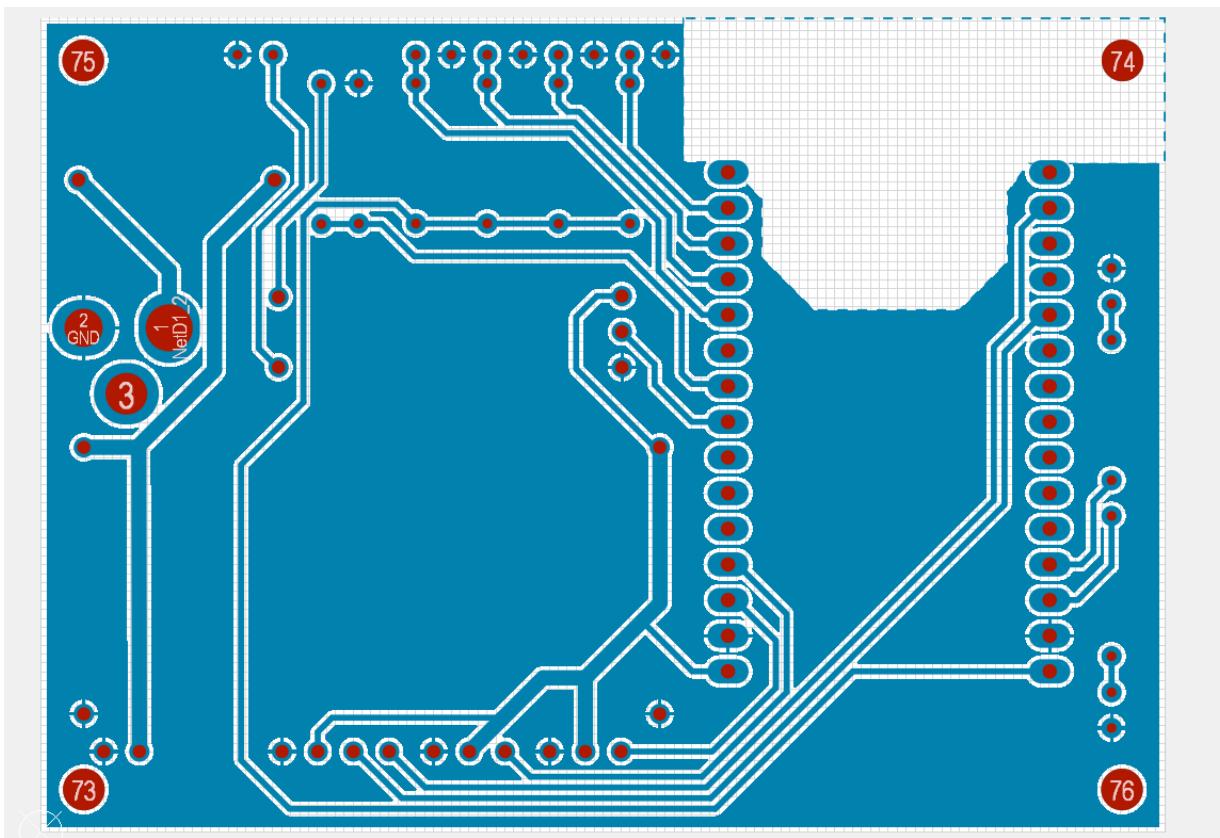
Фиг. 4.8. Изглед към печатната платка, показващ слой Компоненти, или т.нар. "Бял печат" (Top Silkscreen)

На фигура 4.8 се вижда най-горният слой на платката, бял печат или Top Silkscreen. В бяло са очертани местата на който ще се поставят модулите ESP32 devkit, ACS712, LM2596, женски 12V barrel jack 2.1 x 5.5 mm както и резистори номерирани от от едно до осем и един диод. Именувани са изводите за свързване на останалите компоненти, чието интегриране в платката е невъзможно или непрактично. Те представляват изводи за свързване на:

батерията, екрана, двата сервомотора, двата светодиода, четирите фоторезистора, и соларният панел.

В горният ляв ъгъл е предоставена информация за името и версията на проекта.

4.2.3. Долен слой (Bottom Layer)

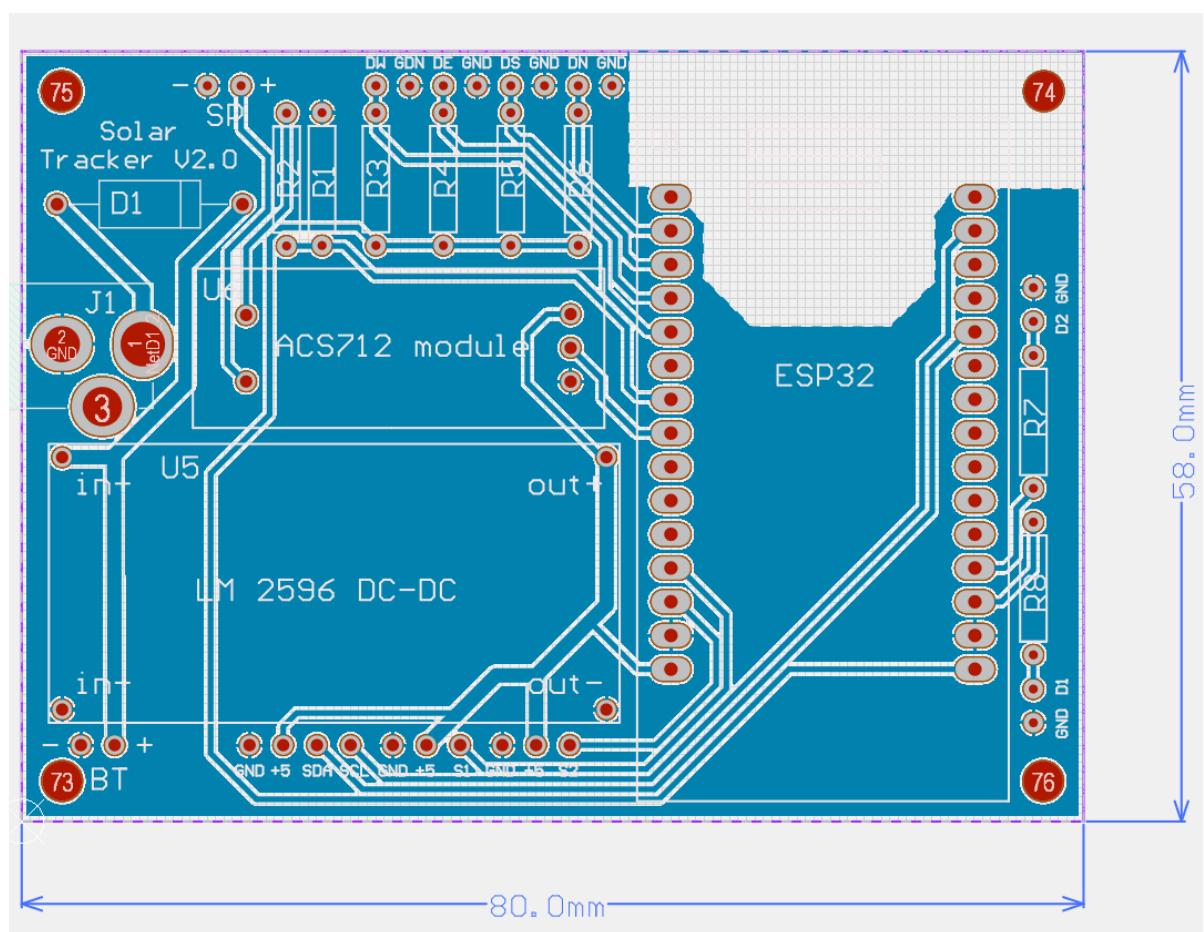


Фиг. 4.9. Изглед към печатната платка, показващ Долен слой
(Bottom layer)

Както се вижда на фигура 4.9, при проектирането на долният проводящ слой са спазени връзките между компонентите показани в глава 3 като пистите са оразмерени, за да позволяват безопасното протичане на ток без прегряване. Стандартната ширина използвана е 20 mil, което се равнява на 0.512 mm, позволяващи протичането на до 2 A, многократно надвишаващо потенциалния ток, който се предвижда да преминава през тях, там където писта с такава ширина е поставена. Най-широката писта е 46 mil или 1.1684 mm, за да позволи протичането на до 3 A, което е и капацитетът на понижаващия преобразувател на напрежение.

По цялата площ на платката, с изключение на пистите, ТНО отворите и площта под антената на микроконтролера е запазен медният слой като е заземен, което опростява схемата и осигурява допълнителна защита. Умишлено е избегнато прокарването на писти под антената за комуникация, за да не се нарушава предаването на информация и цялостта на сигнала както от и към антената така и по медните писти. За монтаж са предвидени четири отвора с диаметър 3 mm в четирите края на платката, които не са метализирани или заземени.

4.3. Окончателен вариант на проектираната печатна платка

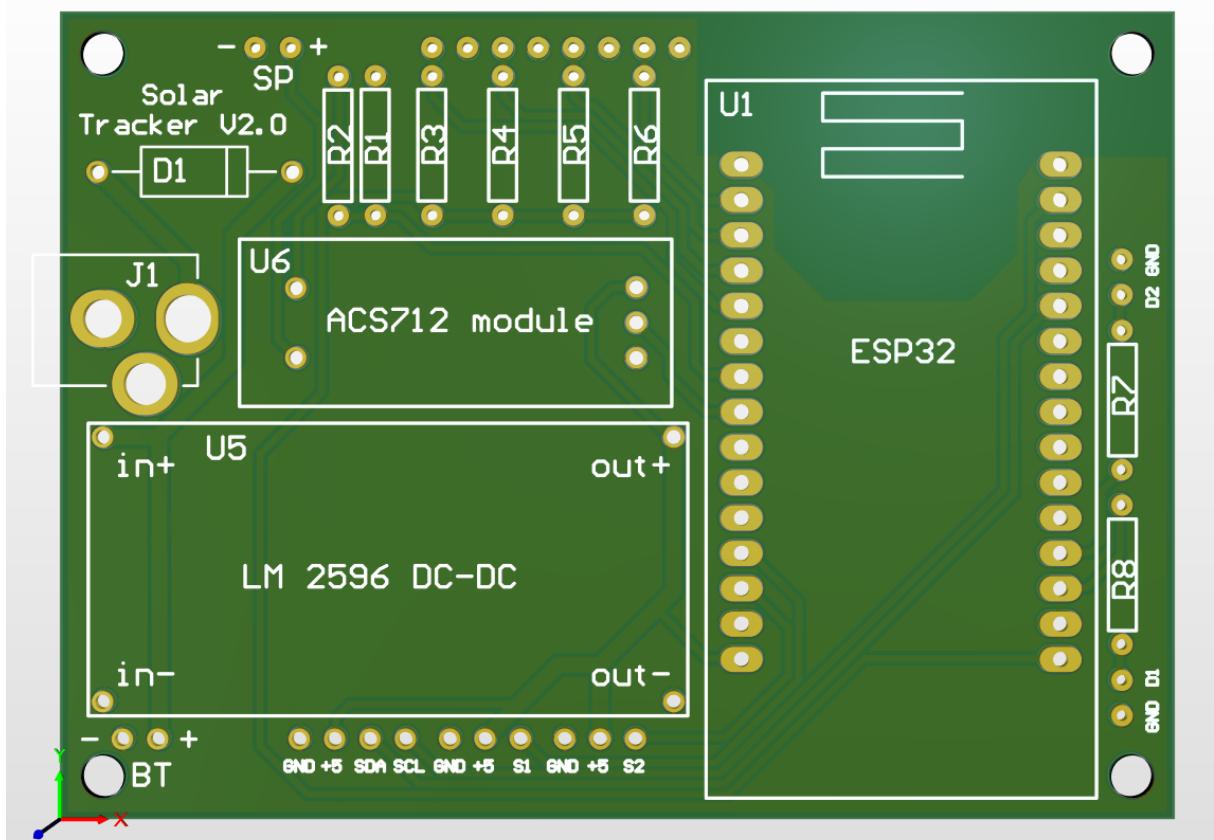


Фиг. 4.10. Изглед към печатната платка със всички слоеве

От фигура 4.10 е видимо, че печатната пратката е в много компактен формат, като се има в предвид, че модулите са сравнително обемни.

Въпреки затрудненията, в модела е използван само един проводящ слой. С оглед на постигнатите резултати може да се направи извода, че изискванията са постигнати.

На фигура 4.11 може да се види екран от триизмерната визуализация на платката.



Фиг. 4.11. Визуализация на модела на печатната платка в преден план

4.5. Изработване на печатната платка

За изработването на конкретната печатна платка се налага обработка на създаденият с помощта на Circuitmaker файл от тип *.CMPcbDoc, която предполага генерирането на гербер файлове. Това е стандартният файлов формат, с който работят повечето компании и машини за изработка на печатни платки.

Пета глава - Управляващ софтуер

В глава пета ще бъде разгледана структурата на разработеният софтуер и значимите детайли в програмния код, за управление на системата и пренос и обработка на данните. Хранилището на програмният код може да бъде открыто [тук](#) [52].

5.1. Избор на развойна среда

Arduino IDE

Arduino IDE е свободен за ползване софтуер проектиран от Arduino.cc и използван главно за писане, компилиране и качване на код в почти всички модули Arduino и ESP. Той е достъпен за всички популярни операционни системи и работи на платформата Java, която идва с вградени функции и команди, които играят жизненоважна роля при отстраняване на грешки, редактиране и компилиране на кода.

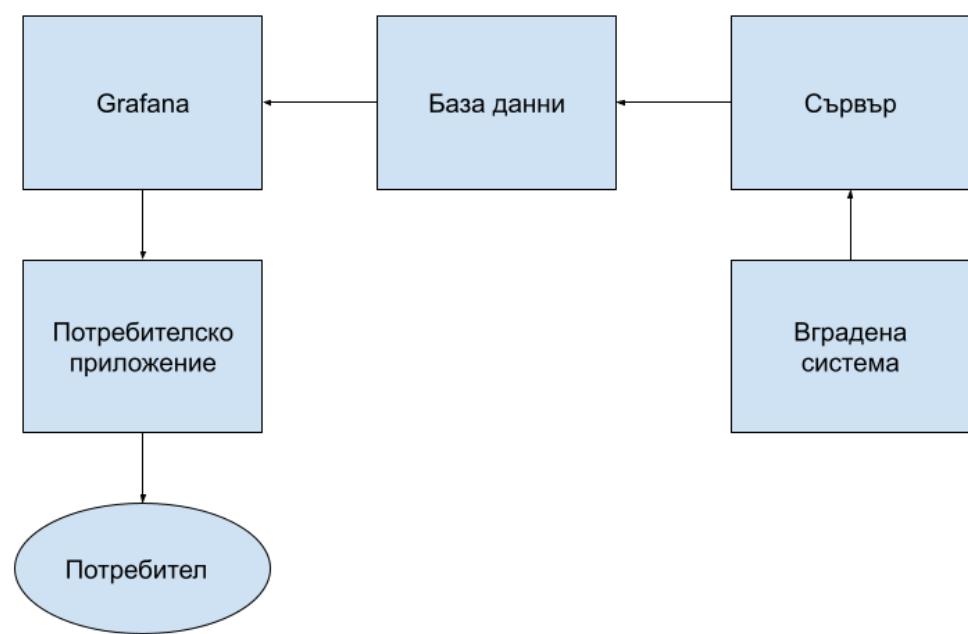
Основният код, известен също като скица, създаден на платформата IDE, в крайна сметка ще генерира шестнадесетичен файл, който след това се прехвърля и качва в контролера на дъската. IDE средата съдържа основно две основни части: редактор и компилатор, където първият се използва за писане на необходимия код, а по-късно се използва за компилиране и качване на кода в дадения модул Arduino. Тази среда поддържа както езика C, така и C++.

Visual Studio Code

Visual Studio Code е лек, но мощен редактор на изходен код и е достъпен за Windows, macOS и Linux. Той има богата екосистема от разширения за множество езици като C++, C#, Java, Python, PHP, Go, JavaScript, TypeScript и Node.js. На първо място, това е сравнително прост редактор, който изпълнява задачи бързо. Със семпъл и добре подреден дизайн той позволява минимизиране на изгубеното време и бърза работа. Visual Studio Code е разработен от Microsoft и е много

надежден. Той е бърз и може да се справи с много разнообразни задачи, като бързи ангажименти в Git или отваряне и сортиране на съдържанието на множество папки. Популярността на VS Code бележи стремителен ръст. Редакторът има вграден терминал, както и вградена поддръжка на Git. Функцията „IntelliSense“ предлага автоматично довършване на код и данни за параметрите на функциите и известни имена на променливи.

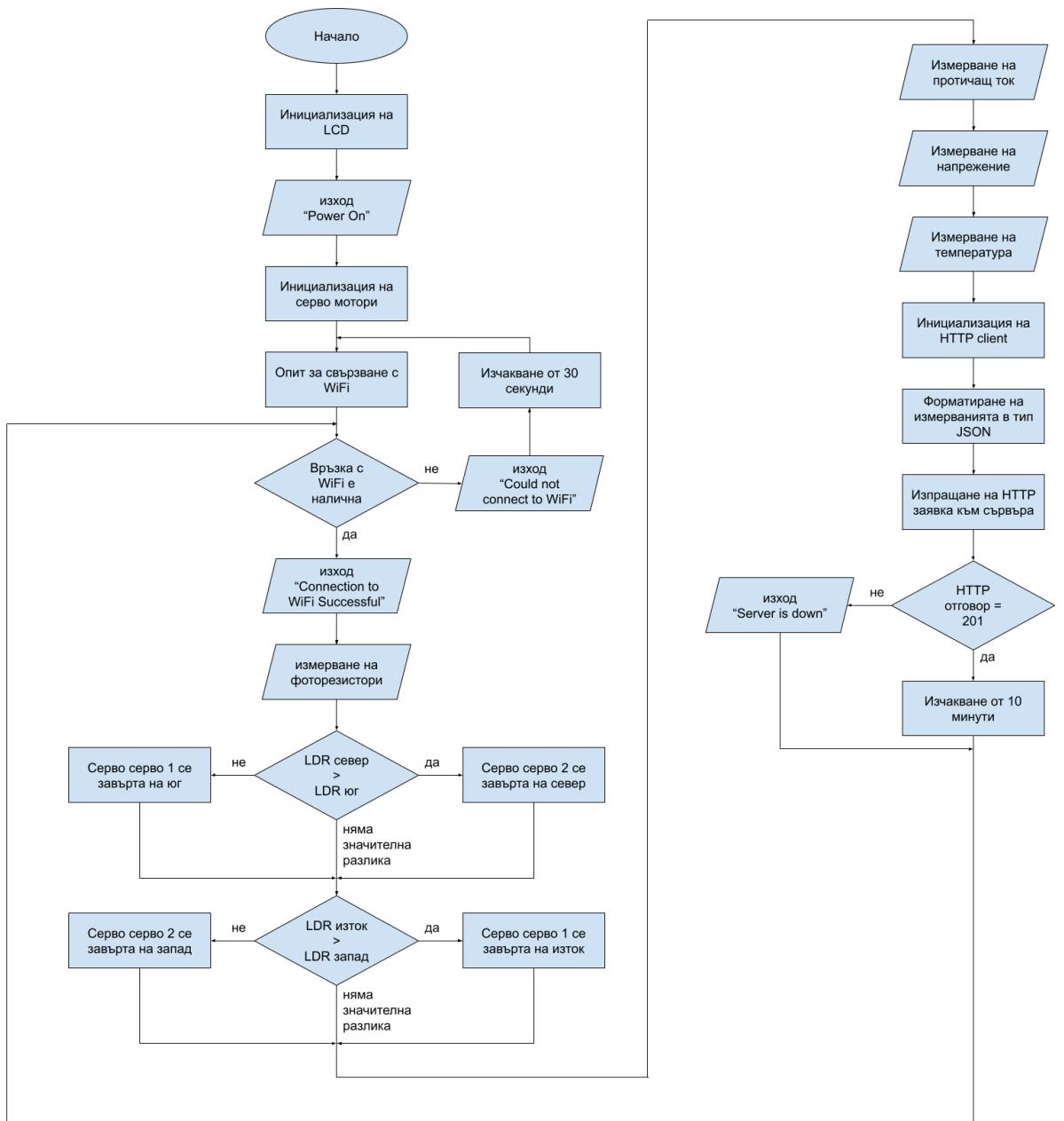
5.2. Обща блокова схема на софтуера



На фигура 5.1 са показани основните софтуерни компоненти в системата. Всеки един от блоковете функционира самостоятелно, което прави архитектурата на системата много надеждна и лесна за поддръжка и обслужване в случай на неизправност. Информацията под формата на измервания се извлича от сензорите във вградената система след което се изпращат на сървъра посредством HTTP заявка,

който ги обработва и записва в базата данни, от там те се взимат и обработват от Grafana, която ги предоставя на потребителското приложение по формата на структурирани графики, от където потребителят може да ги достъпи.

5.3. Блокова схема на алгоритъма за управление на вградената система



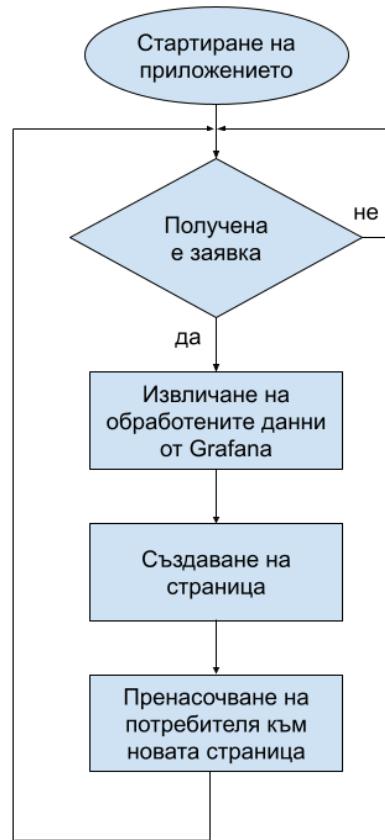
Фиг. 5.2. Блокова схема на алгоритъма за управление на вградената система

5.4. Блокова схема на алгоритъма работа на сървъра



Фиг. 5.3. Блокова схема на алгоритъма на сървъра

5.5. Блокова схема на алгоритъма на приложението за връзка с потребителя



Фиг. 5.4. Блокова схема на алгоритъма на приложението

5.6. Програмен код на вградената система

В тази точка предстои да бъдат разгледани конкретни части от програмният код управляващ микроконтролерна платформа (ESP-Wroom-32) на вградената система, заедно със съответния контекст и разяснения. Използваният програмен език е C++ със добавени допълнителни функции стандартни при разработката на управляващ софтуер за Arduino и ESP системи.

5.6.1. Използвани библиотеки

На фигура 5.5 е показана извадка от кода на която се виждат използваните библиотеки в програмният код на микроконтролера.

```
1 #include <ESP32Servo.h>
2 #include <LiquidCrystal_I2C.h>
3 #include <Wire.h>
4 #include <WiFi.h>
5 #include <HTTPClient.h>
```

Фиг. 5.5. Използвани библиотеки

ESP32Servo.h

Тази библиотека се опитва да реплицира семантиката на Arduino Servo библиотеката за ESP32, с две (по избор) допълнения. Двете нови функции разкриват способността на таймерите ESP32 PWM да променят ширината на таймера.

Wire.h

Тази библиотека позволява комуникация с I2C / TWI (Two-Wire Interface) устройства. Библиотеката наследява от функциите Stream, което я прави съвместима с други библиотеки за четене / запис.

LiquidCrystal_I2C.h

Библиотеката LiquidCrystal_I2C работи в комбинация с библиотеката Wire.h. API (Application Programming Interface) или иначе казано наборът от функции и техните наименования е много подобен (но не е идентичен) с API, използван от библиотеката Arduino LiquidCrystal за текстови LCD дисплеи с паралелен интерфейс.

WiFi.h

WiFi.h библиотеката осигуряват поддръжка за конфигуриране и наблюдение на функционалността на ESP32 Wi-Fi мрежа, както и конфигурации за множество режими.

HTTPClient.h

Това е библиотека за лесно извършване на HTTP GET, POST и PUT заявки към уеб сървър. Работи с всеки клас, извлечен от Client - така че превключването между Ethernet, WiFi и GSMClient изисква минимални промени в кода.

5.6.2. Константи

Част от използваните константи са показани на фигура 5.6 и фигура 5.7.

```
20 // set the LCD number of columns and rows
21 #define lcdColumns 16
22 #define lcdRows 2
23 // To get display address, run an I2C scanner sketch, default is 0x27
24 LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);
25
26 // Set voltage divider pin
27 #define voltage_sensor_pin 26
28
29 // Set temperature sensor pin
30 #define temperature_sensor_pin 25
31 // value of R1_temperature_sensor on temperature sensor board
32 #define R1_temperature_sensor 10000
33 //steinhart-hart coeficients for thermistor
34 #define c1 0.001129148
35 #define c2 0.000234125
36 #define c3 0.0000000876741
```

Фиг. 5.6. Екран, показващ константите свързани с LCD и температурният сензор

На фигура 5.6. може да се видят константите определящи вида на използваният LCD, както и инициализирането на обекта, който го репрезентира в програмата, показано на реда номериран 24.

LIiquidCrystal_I2C() е функция която приема I²C адреса, броят колони и редове на дисплея. Адресът на този вид дисплей е "0x27" по подразбиране, което беше потвърдено от предварително направен тест. На фигурата, на ред 27 е показано, че средният извод на делителят на напрежение се очаква да бъде свързан към GPIO извод номер 26 на микроконтролера. След това са показани желаният извод на който да се свърже аналоговият изход на температурния сензор както и константите "R1_temperature_sensor", която отговаря на съпротивлението на резисторът образуващ делител на напрежение със термистора, и коефициентите на Стайнхард за термистор.

```

38 // Set amperemeter pin
39 #define amperemeter_pin 35
40 // amperemeter ACS712 (version up to 5A) raises output voltage with 185mV per
41 // points_per_amp = 0.185 / 3.34 * 4096, where 3.34 is the work voltage of e:
42 #define points_per_ampere 270
43 // ACS712 output range starts from 2.5V, that means 2.5V (read as 3065 points)
44 // 3065 is adjusted to 2820 to level inaccuracies of the system
45 #define offset_in_points 2820
46
47 // Set pin numbers for LDR voltage dividers
48 #define ldr_pin_1 33
49 #define ldr_pin_2 32
50 #define ldr_pin_3 34
51 #define ldr_pin_4 35
52 // Since LDRs aren't precisely equal, different multipliers will be used to ]
53 #define ldr_multiplier_1 1
54 #define ldr_multiplier_2 1
55 #define ldr_multiplier_3 1.1
56 #define ldr_multiplier_4 0.9
57
58 //constants required for measure_resistance()
59 #define R1 51000 // the resistance of the known resistor
60 #define Vin 3.34 // the voltage across both resistors

```

Фиг. 5.7. Екран, показващ константите свързани със амперметъра и
фоторезисторите

На фигура 5.7 са показани използвани при четене на данните от амперметърът константи. “points per ampere” показва каква промяна в изхода на дванайсет-битовият АДЦ на микроконтролера съответства на един ампер, измерен от амперметъра. “offset_in_points” е числото нужно да се извади от проченият от АДЦ резултат, за да се нулира правилно измерването. След това са показани изводите на които се очаква да бъдат свързани делителите на напрежение използвани за измерване на фоторезисторите, както и константите нужни за изравняване на показанията, използването на които се налага поради

присъщата на фоторезисторите разлика в съпротивлениято. "R1" е стойността на резисторите използвани при измерването на фоторезисторите, докато "Vin" е пада на напрежение върху всеки индивидуален делител на напрежение.

5.6.3. Функции обработващи данните от сензорите

```
192 // function reseiving number of pin connected to Vout of voltage divider using
193 int measure_resistance(int input_pin){
194     int raw = analogRead(input_pin); // Get input from ADC pin (ranging from 0-
195     if(raw){
196         float buffer = raw * Vin;
197         float Vout = buffer/ 4095; // Calculate voltage drop on second resistor,
198         buffer = (Vin/Vout) -1;
199         int R2 = R1 / buffer; // Calculate resistance knowing that R1 is 51K ohms
200         //Serial.println(String(R2));
201         return R2;
202     }
203     return -1; // If analogRead() didn't yield proper value return -1
204 }
```

Фиг. 5.8. Екран показващ функция measure_resistance()

Функцията measure_resistance(), показана на фигура 5.8 е предназначена да измерва съпротивлението на фоторезистор свързан в делител на напрежение с резистор със съпротивление "R1" и общ пад на напрежение върху делителя от "Vin" волта. Тя използва АДЦ на посоченият извод "input_pin", който се очаква да е свързан към средният извод на делителя на напрежение. Изходното напрежение се изчислява (означено с "Vout") след което то се използва за да се пресметне съпротивлението на фоторезисторът.

```

206 // function that measures the input of ACS712 amperemeter connected to amperemeter
207 float measure_current(){
208     float input = analogRead(amperemeter_pin);
209     float current = (input - offset_in_points) / points_per_ampere;
210     return current > 0 ? current : 0;
211 }

```

Фиг. 5.9. Екран показващ функция measure_current()

Функцията measure_current(), показана на фигура 5.8 служи за разчитане на показанията на амперметър ACS712, като връщената единица е амperi. Тя приема показанията на АДЦ на извод "amperemeter_pin", свързан към аналоговият изход на сензора. Поради това, че сензорите от този тип започват да дават показания от 2.5 V нагоре, тоест аналогов сигнал еквивалентен на 2.5 V означава, че ток не протича през амперметъра, се налага да се извади константата "offset_in_points" от променливата "input". След това резултатът се дели на константата "points_per_ampere", която е показва колко единици отговарят на един ампер, за да се получи крайният резултат.

```

220 // function that measures voltage up to 33V using voltage divider comprised of two resistors
221 float measure_voltage(){
222     float input = analogRead(voltage_sensor_pin);
223     return input / voltage_division_constant; // voltage_division_constant is defined in the code
224 }

```

Фиг. 5.10. Екран показващ функция measure_voltage()

Функцията measure_voltage() се използва, за да измерва напрежение до 33 V с помощта на делител на напрежение състоящ се от резистори със стойност 100 KOhm и 10 KOhm. След измерване на напрежението използвайки АДЦ на извод с номер "voltage_sensor_pin", резултатът се

разделя на константата “voltage_division_constant”, прецизната стойност на която е определена чрез практически тестове.

```
226 // function that measures the input of temperature sensor KY-013 and returns degrees
227 float measure_temperature() {
228     int input = analogRead(temperature_sensor_pin);
229     float R2 = R1_temperature_sensor * (4096.0 / (float)input - 1); //calculate resis:
230     float logR2 = log(R2);
231     float T = (1.0 / (c1 + c2*logR2 + c3*logR2*logR2*logR2)); // temperature in Kelv:
232     T = T - 273.15; //convert Kelvin to Celcius
233     //Serial.println("Temperature: " + String(T) + " C");
234     return T;
235 }
```

Фиг. 5.11. Екран показващ функция measure_temperature()

Целта на функцията measure_temperature() е да интерпретира показанията на температурен сензор KY-013. След измерване на сигнала на аналоговия изход на сензора, свързан към извод с номер “temperature_sensor_pin” на микроконтроллера, използвайки АДЦ на съответния извод, се пресмята съпротивлението на термистора на сензора. С помощта на константите на Стайнхард се пресмята температурата в мерна единица келвини след това се

5.6.4. Функции осъществяващи комуникацията

```
114 // Function that creates and sends json with the measured data and sends it to the specific
115 int send_data() {
116     if(WIFI.status() == WL_CONNECTED) { //Check WiFi connection status
117
118         // Get measurement from ammeter
119         float current = measure_current();
120         // Get measurement from voltage divider
121         float voltage = measure_voltage();
122         // Get measurement from temperature sensor
123         float temperature = measure_temperature();
124
125         // Take multiple measurements
126         for(int i = 0; i<4; i++){
127             delay(50);
128             current += measure_current();
129             voltage += measure_voltage();
130             temperature += measure_temperature();
131         }
132         // Average the measurements and for greater accuracy
133         current /= 5;
134         voltage /= 5;
135         temperature /= 5;
136
137         // Create HTTP client and point it to the address of the server
138         HTTPClient http;
139         http.begin(server);
140
141         // Create and post a JSON object using the HTTP client
142         // First it's specified that the request body contains JSON
143         http.addHeader("Content-Type", "application/json");
144         // Then the JSON is created and immediately the request is sent
145         int httpResponseCode = http.POST("{\"current\":" + String(current) +
146                                         ", \"voltage\":" + String(voltage) +
147                                         ", \"temperature\":" + String(temperature) + "}");
148
149         // Check server response
150         if(httpResponseCode<0) {
151             String response = http.getString(); //Get the response to the request
152             Serial.println(httpResponseCode); //Print return code
153             //Serial.println(response); //Print request answer
154         } else {
155             Serial.print("Error on sending POST: ");
156             Serial.println(httpResponseCode);
157         }
158         http.end(); //Free resources
159         return 0;
160
161     } else{ // If WiFi status is compromised print message on the LCD
162         lcd.print_lack_connection();
163         Serial.println("Error in WiFi connection");
164         delay(3000);
165         connect_to_WIFI();
166     }
167 }
```

Фиг. 5.12. Екран показващ функция send_data()

Целта на функцията send_data() е да запише измерванията на сензорите, след което да създаде HTTP клиент и да изпрати заявка

съдържаща измерените данни до сървъра. Функцията започва като проверява дали все още има връзката с WiFi мрежата, след това прави множество измервания, използвайки разгледаните във точка 5.6.3 функции, и пресмята средната стойност за повишена точност на данните. Създава се HTTP клиент с помощта на библиотеката `HTTPClient.h` и информацията, структурирана като тип JSON (JavaScript Object Notation) се изпраща на предварително уточнения адрес на сървъра, записан в константата "server". Последно отговора на заявката се приема и в зависимост от него се показва съответното съобщение на LCD экрана. При липса на връзка с WiFi се извиква функцията `connect_to_WiFi()`.

```
167 // function that tries to connect to WiFi network with the given credentials
168 int connect_to_WiFi(){
169     // Try connection to local WiFi network using the specified credentials
170     WiFi.begin(ssid, password);
171
172     for(int i=0; i<10; i++) { // Continiously check for connection
173         // If connection is sucessful return 0 (success)
174         if(WiFi.status() != WL_CONNECTED){
175             // Show the connection was sucessful on the LCD
176             lcd_print_connected();
177             return 0;
178         }
179         delay(1000); // Delay is added to make sure the network doesn't mista
180         lcd_print_connecting();
181     }
182     // Show the connection failed on the LCD
183     lcd_print_no_connection()
184     // If connection is sucessful return -1 (failed)
185     return -1;
186 }
```

Фиг. 5.13. Екран показващ функция `connect_to_WiFi()`

Целта на функцията `connect_to_WiFi()` е да реализира WiFi връзка с предварително зададени име на мрежа и парола, записани съответно в "ssid" и "password". Програмата използва класа "WiFi" от библиотеката

WiFi.h, като WiFi.begin() инициализира връзката, след което в продължение на 10 секунди се проверява дали свързването е било безпроблемно. При успех се показва съответното съобщение на дисплея и се връща резултат 0 (успех), в противен случай резултатът е -1 (провал), и се показва съответстващо съобщение на LCD.

5.6.5. Функция контролираща серво моторите

```
237 // function that measures and compares the resistance of the two pairs of LDRs
238 // and moves the servos until the LDRs have the same resistance, which means
239 void control_servo(){
240     // Get measurement for LDR1
241     int ldr1 = measure_resistance(ldr_pin_1) * ldr_multiplier_1;
242     //Serial.println("LDR 1: " + String(ldr1));
243     // Get measurement for LDR2
244     int ldr2 = measure_resistance(ldr_pin_2) * ldr_multiplier_2;
245     //Serial.println("LDR 2: " + String(ldr2));
246
247     // Compare LDR1 and LDR2 and move the servo towards the brighter side
248     if( (ldr1 - ldr2) > (ldr1 + ldr2)/2 * 0.2 ){
249         if(servo_position_1 < 180) servo_position_1++;
250     }
251     if( (ldr2 - ldr1) > (ldr1 + ldr2)/2 * 0.2 ){
252         if(servo_position_1 > 0) servo_position_1--;
253     }
254     // Move servo one
255     servo_1.write(servo_position_1);
256
257     // Get measurement for LDR1
258     int ldr3 = measure_resistance(ldr_pin_3) * ldr_multiplier_3;
259     //Serial.println("LDR 3: " + String(ldr3));
260     // Get measurement for LDR2
261     int ldr4 = measure_resistance(ldr_pin_4) * ldr_multiplier_4;
262     //Serial.println("LDR 4: " + String(ldr4));
```

Фиг. 5.14. Екран показващ функция control_servos()

Функцията control_servos() използва сервомоторите и показанията на фоторезисторите, за да насочи слънчевият панел директно към слънчевата светлина. В началото се използва вее разгледатаната функция measure_resistance(), за да се определят съпротивленията на фоторезисторите образуващи двойка север-юг, след което се сравняват и серво моторът отговарящ на тази двойка се мести в посоката на

фоторезистора с по-малко съпротивление. Аналогични действия се предприемат и за двойката фоторезистори изток-запад и вторият серво мотор.

5.6.6. `setup()` функция

```
189 void setup() {
190   Serial.begin(115200);
191
192   // This line is required to bypass a bug with the LCD where random characters are di-
193   // It lowers the clock from it's default value of 100 000 to lesser 10 000
194   Wire.setClock(10000);
195   // initialise LCD
196   lcd.init();
197   // turn on LCD backlight
198   // lcd.backlight();
199   // set cursor to first column, first row
200   lcd.setCursor(0, 0);
201   // print message
202   lcd.print("Power On :D");
203
204   delay(4000);    //Delay needed before calling the WiFi.begin
205   connect_to_WiFi();
206
207   // Allow allocation of all timers. This is required to controll the servos
208   ESP32PWM::allocateTimer(0);
209   ESP32PWM::allocateTimer(1);
210   ESP32PWM::allocateTimer(2);
211   ESP32PWM::allocateTimer(3);
212   // Standard 50 hs servo (20ms wavelenght), this is the required frequency for the MK
213   servo_1.setPeriodHertz(50);
214   servo_2.setPeriodHertz(50);
215   // Attaches the servo on pin 18 to the servo object
216   // 500 and 4000 specify the range (reach) of the servo which in reality is 120 degrees
217   servo_1.attach(servo_pin_1, 500, 4000);
218   servo_2.attach(servo_pin_2, 500, 4000);
219 }
```

Фиг. 5.15. Екран показващ функция `setup()`

Стандартната функция `setup()` е ключова в използваният от Arduino и ESP модел за програмиране без ОС. Функцията `setup()` се изпълнява еднократно в началото на програмата. Типично се използва за инициализация на променливи. В конкретния случай се

инициализират Serial каналът, който предоставя дневник на случващите се събития. Ред Wire.setClock(10000); гарантира безпроблемен пренос на данни към екрана, тъй като стойността по подразбиране (100 000), се оказа проблемна при тестовете и не позволяваше правилната работа на дисплея. Използваният LCD се инициализира и се изпраща съответното съобщение. След кратко изчакване, налагашо се, за да се подсигури безпроблемна работа на микроконтролера, се изпълнява функцията connect_to_WiFi(). Инициализацията на таймерите на микроконтролера се налага, за да могат да бъдат контролирани серво моторите. Накрая се уточнява честотата на която работят моторите (50 Hz, което означава дължина на вълната 20mS) и изводите към който са свързани.

5.6.7. loop() функция

```
297 void loop() {  
298     // Compare LDRs and move the servo if necessary  
299     controll_servo();  
300  
301     // Send an HTTP POST request every 10 minutes  
302     if ((millis() - last_time) > timer_delay){  
303         send_data();  
304         last_time = millis();  
305     }  
306     delay(100);  
307 }
```

Фиг. 5.16. Екран показващ функция loop()

На фигура 5.16 е показана функция loop(), която е стандартна и задължителна за управлението на платформи като ESP и тези от семейството на Arduino, без ОС. След първоначалното еднократно извикване на setup(), функцията loop() се изпълнява циклично, докато не възникне външно или вътрешно прекъсване. В конкретния случай функцията controll_servo() се изпълнява на всяко повторение на loop(),

което позволява бързо пренасочване на соларният панел, подходящо за демонстративни цели. Функцията millis() в комбинация с променливата "last_time" и константата "timer_delay" се използват, за да ограничат извикването на функцията send_data() да интервал от "timer_delay" милисекунди.

5.7. Програмен код на сървърното приложение

В следващата точка ще бъде разгледан програмният код на приложението имащо за цел да приема, обработва и записва в базата данни, информацията изпратена от вградената система. При разработката му е използван Django, който представлява популярен фреймуърк на езика Python. Django е от високо ниво, което насырчава бързото развитие и чист, прагматичен дизайн. Той идва с много предварително разработени функционалности, позволява лесно разширяване и е свободен за ползване. Конкретният Django проект поддържа само едно приложение, наречено Django_Postgre_Server.

5.7.1. Модели

```
1  from django.db import models
2
3  class Reading(models.Model):
4      current = models.FloatField()
5      voltage = models.FloatField()
6      wattage = models.FloatField()
7      temperature = models.FloatField()
8      time = models.DateTimeField(auto_now_add=True)
```

Фиг. 5.17. Екран показващ модел Reading

Приложението използва един модел наречен Reading, който съответства на таблица esp_data_collector_reading в базата данни и има пет полета, като current съхранява стойността на измереният ток, voltage на напрежението, wattage на мощността, temperature на

температурата и time момента в който заявката съдържаща данните, изпратена от вградената система е получена на сървъра.

5.7.2. Връзка с базата данни

```
85  # Database
86  # https://docs.djangoproject.com/en/4.0/ref/settings/#databases
87
88 DATABASES = {
89     'default': {
90         'ENGINE': 'django.db.backends.postgresql_psycopg2',
91         'NAME': 'Solar_Data',
92         'USER': 'postgres',
93         'PASSWORD': 'postgres',
94         'HOST': '127.0.0.1',
95         'PORT': '5432',
96     }
97 }
```

Фиг. 5.18. Екран показващ параметрите на базата данни

На фигура 5.18 е показана частта от файл Django_Postgre_server/settings.py съдържаща данните нужни на приложението за установяване на връзка с базата данни.

5.7.3. REST мрежови контролери

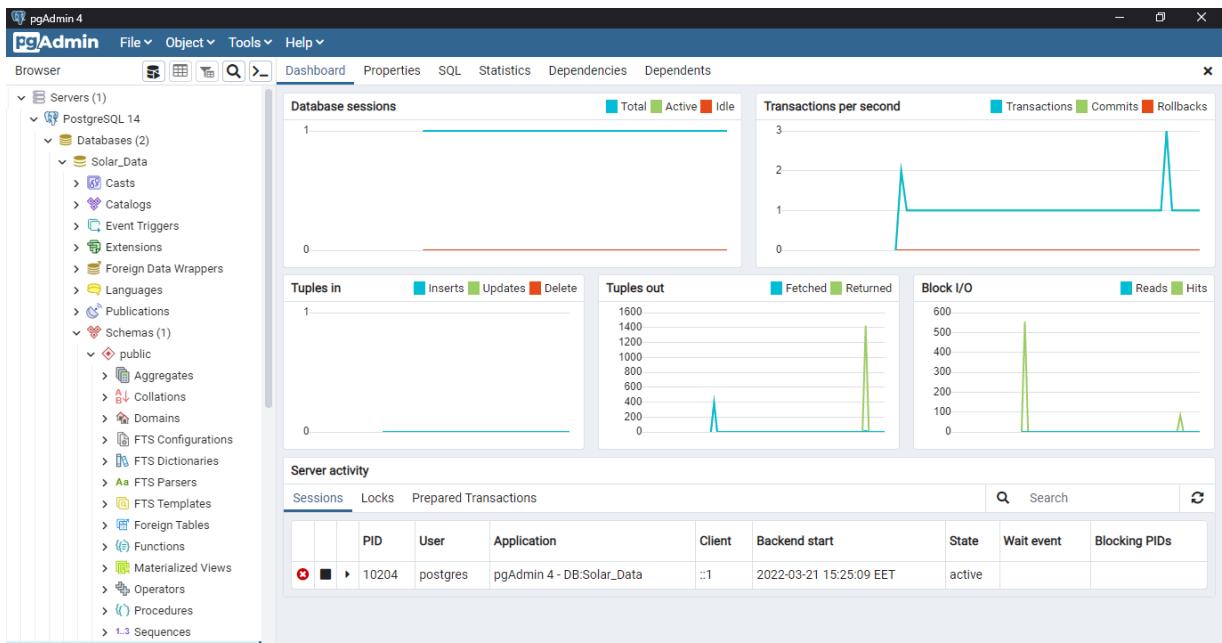
```
1  from django.http import JsonResponse
2  import json
3  from esp_data_collector.models import Reading
4  from django.views.decorators.csrf import csrf_exempt
5
6  @csrf_exempt
7  def handle_esp_reading(request):
8      if request.method == "POST":
9          try:
10              current = json.loads(request.body)["current"]
11              voltage = json.loads(request.body)["voltage"]
12              temperature = json.loads(request.body)["temperature"]
13              wattage = round( current * voltage, 2)
14
15              my_reading = Reading()
16              my_reading.current = current
17              my_reading.voltage = voltage
18              my_reading.wattage = wattage
19              my_reading.temperature = temperature
20              my_reading.save()
21
22          return JsonResponse({"message" : "created"}, status=201)
23      except:
24          return JsonResponse({"message" : "failed"}, status=500)
25      else:
26          return JsonResponse({"message" : "failed"}, status=500)
```

Фиг. 5.19. Екран показващ контролер handle_esp_reading

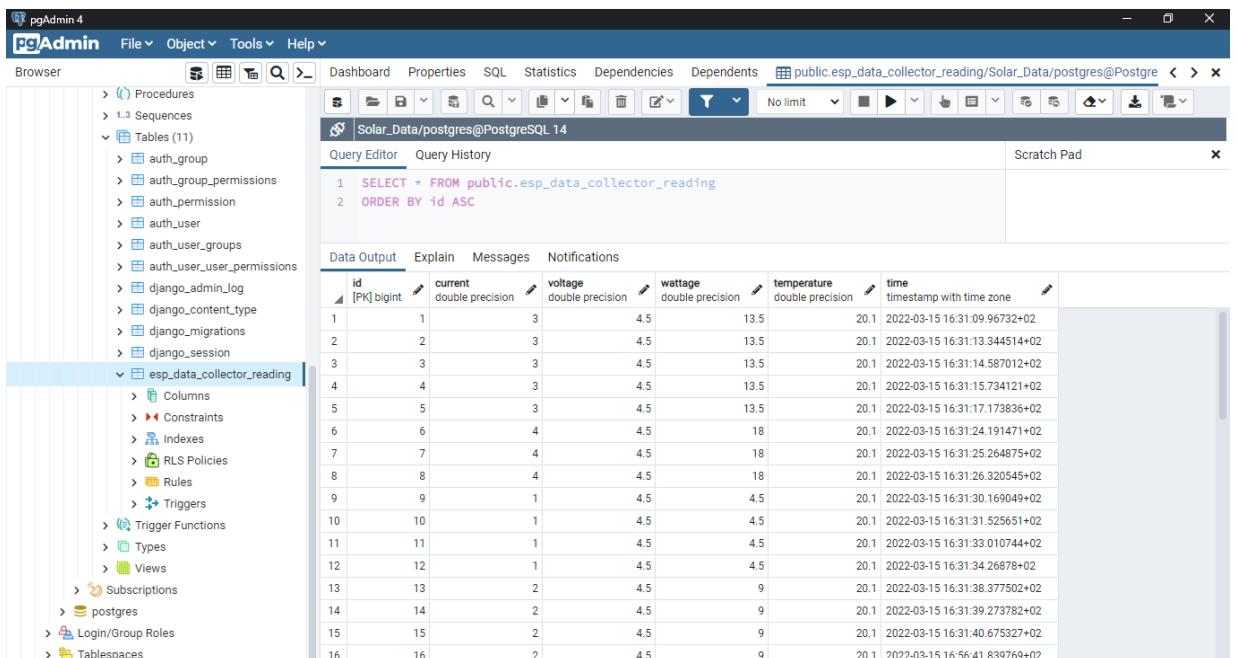
В Django приложението оперира един мрежови контролер, наречен handle_esp_reading. Анотацията @csrf_exempt предотвратява изискването на CSRF (Cross-Site Request Forgery) маркер, което се налага, за да се позволи приемането на POST заявки от микроконтролерната платформа ESP32. В началото на функцията се проверява дали заявката е от тип POST, след което се прави опит да се извлекат данните от тялото на заявката, за които се очаква да бъдат форматирани под тип JSON. При успех се прави нов запис в базата данни, и се връща съобщение "created" с код на отговор 201, в противен случай - съобщение "failed" с код на отговор 500.

5.8. База данни

За базата данни е използван PostgreSQL и графичният интерфейс pgAdmin 4, който е показан на фигура 5.20. PostgreSQL е гъвкава система за управление на релационна база данни с отворен код с функции, предназначени да отговорят на промените в работните натоварвания, от единични машини до складове за данни до уеб услуги с много едновременни потребители. PostgreSQL използва и разширява SQL (оттук и името) и е широко разширим до редица случаи на употреба извън обикновените транзакционни данни. PostgreSQL е релационна база данни. Като такава, това е лист от връзки между кортежи, представляващи обекти (като документи и хора) и връзки (като авторство). Връзките съдържат атрибути с фиксиран тип, представляващи свойства на обект (като заглавие) заедно с първичен ключ. Типовете атрибути могат да бъдат или атомни (като цяло число, плаваща запетая или булеви), или структурирани (като масив или процедура). Конкретната база данни е наречена Solar_Data и се състой от стандартните помощни таблици нужни за функционирането на Django, както и една наречена esp_data_collector_reading, показана на фигура 5.21, в която се съдържат записаните измерени параметри на вградената система, съответстващи на показания в точка 5.7.1 модел.



Фиг. 5.20. Екран показващ информационно табло създадено от pgAdmin 4



Фиг. 5.21. Екран показващ таблица esp_data_collector_reading

5.9. Софтуер за визуализация на данните

Представянето на данните по разбираем и усвоим начин е от голямо значение, поради тази причина е използван софтуера Grafana. Тя представлява услуга, която се изпълнява на отделен сървър и комуникира чрез HTTP заявки. Информационните табла на Grafana са мощни инструменти за анализ и визуализация с отворен код, които се състоят от множество отделни панели, подредени в мрежа. Панелите взаимодействват с конфигурирани източници на данни, включително (но не само) PostgreSQL, AWS CloudWatch, Microsoft SQL сървър, Prometheus, MySQL, InfluxDB и много други. Grafana е проектирана така, че всеки панел е свързан с източник на данни. Процесът на настройка на информационното табло на Grafana и интегрирането му с различни източници на данни е лесен и са налични много шаблони. Тъй като информационните табла на Grafana поддържат множество панели в една мрежа, можете да визуализирате резултати от множество източници на данни едновременно, в реално време. Създаденото информационно табло обслужващо потребителският интерфейс на проекта и панелите от който се състои са показани на фигура 5.22.



Фиг. 5.22. Екран показващ информационно табло Basic Graphics в графичния интерфейс на Grafana

5.10. Потребителско интернет приложение

За създаването на потребителски интерфейс е използван фреймуъркът React на езика JavaScript. React наследява създаването на компоненти на потребителския интерфейс за многократна употреба, които представлят данни, които се променят с течение на времето. Много хора използват React като V в MVC архитектурата. React абстрагира DOM (Document Object Model), предлагайки по-опростен модел на програмиране и по-добра производителност. React внедрява еднопосочен реактивен поток от данни, което намалява шаблона и е по-интуитивно от традиционното обвързване на данни. Той използва виртуален DOM, който е обект на JavaScript. Това подобрява производителността на приложенията, тъй като виртуалният DOM на JavaScript е по-бърз от обикновения DOM. Моделите на компоненти и данни подобряват четливостта, което помага да се поддържат по-големи приложения.

```
import './App.css';

function App() {
  return (
    <div className="App" >

      <h1 id="Title">Advanced Solar Data Collector</h1>
      <h2 id="Sub-Title">Brighter Tomorrow</h2>

      <div className="Rectangle">

        <iframe id="PowerWeek" className="Dash-Board" src="http://localhost:3000/d-solo/9X9S_DEnz/basic-gra...
        <iframe className="Dash-Board" src="http://localhost:3000/d-solo/9X9S_DEnz/basic-graphics?orgId=18...
        <iframe className="Dash-Board" src="http://localhost:3000/d-solo/9X9S_DEnz/basic-graphics?orgId=18...
        <iframe className="Dash-Board" src="http://localhost:3000/d-solo/9X9S_DEnz/basic-graphics?orgId=18...
        <iframe className="Dash-Board" src="http://localhost:3000/d-solo/9X9S_DEnz/basic-graphics?orgId=18...

      </div>
    </div>
  );
}

export default App;
```

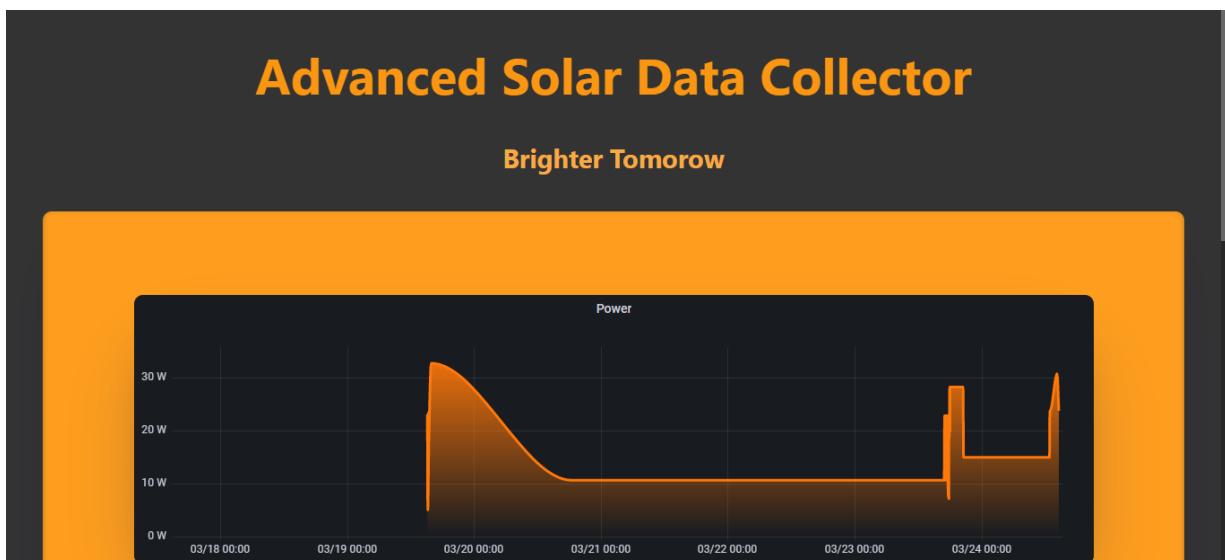
Фиг. 5.23. Екран показващ част от основният компонент в приложението

На фигура 5.23 е показан главният компонент в React приложението. Тагът `<iframe>` позволява лесно вграждане на Grafana панели в потребителският интерфейс. На фигура 5.24 е показан стилизиращия файл от тип `*.css` на компонента.

```
17  box-shadow: □rgba(50, 50, 93, 0.25) 0px 50px 100px -20px,
18  □rgba(0, 0, 0, 0.3) 0px 30px 60px -30px,
19  □rgba(10, 37, 64, 0.35) 0px -2px 6px 0px inset;
20 }
21
22 .Rectangle {
23   width: 94%;
24   height: 10%;
25   margin-top: 3%;
26   margin-bottom: 5%;
27   margin-left: 3%;
28   margin-right: 3%;
29   padding-top: 5%;
30   padding-bottom: 5%;
31
32   border-radius: 10px;
33   background-color: ■rgb(255, 158, 30);
34   box-shadow: □rgba(50, 50, 93, 0.25) 0px 50px 100px -20px,
35  □rgba(0, 0, 0, 0.3) 0px 30px 60px -30px,
36  □rgba(10, 37, 64, 0.35) 0px -2px 6px 0px inset;
37 }
38
39 #Title {
40   color: ■rgb(250, 150, 18);
41   font-size: 350%;
42 }
43
44 #Sub-Title {
45   color: ■rgb(248, 170, 69);
46   font-size: 180%;
47   margin-bottom: 2%;
48 }
```

Фиг. 5.24. Екран показващ част от стилизиращия файл на компонента

Резултатите от работата са показани на фигури 5.25 и 5.26.



Фиг. 5.25. Екран показващ част от потребителското приложение



Фиг. 5.26. Екран показващ част от графиките в потребителското приложение

Шеста глава - Практически резултати

6.1. Подготвителни дейности

Предварително бяха изтеглени и инсталирани софтуерните продукти нужни за реализирането на проекта. Това са програмите Visual Studio Code, Arduino IDE, Circuitmaker, pgAdmin 4, Grafana, PostgreSQL, както и C++, Python, Django, Node.js и React и софтуерните библиотеки посочени във файл requirements.txt. Програмният код на системата се съхранява в хранилище на GitHub [52]. Всеки от изброените продукти е свободен за използване с обучителни и некомерсиални цели.

След задълбочено проучване на конкуренцията и подобните проекти и вариантите за разработка на проекта бяха закупени нужните хардуерни компоненти, предимно от български доставчици.

6.2. Свързване и тестване на хардуерните решения

В тази точка ще бъде разгledан процеса и особеностите по изграждането на функционираща вградена система.

6.2.1. Първоначална подготовка на компоненти

Първоначалната задача беше запояването на рейка, каквато е показана на фигура 6.1, към ESP32 модула, поради причината, че беше набевен такъв който нямаше запоена рейка фабрично. Наложи се и запояването на I²C адаптера към LCD экрана, защото дисплеят не разполага с гнезда за пинове, което оставя само този вариант като възможен. Запоени бяха и батериите в последователна верига. На фигура 6.2 е показана снимка направена по време на процеса по запояването.



Фиг. 6.1. Снимка показваща използваната рейка



Фиг. 6.2. Кадър от работния процес по запояване

6.2.2. Начална работа с микроконтролера

Редно е да се отбележи, че работата върху хардуерните компоненти и връзки се изпълняваше паралелно с разработката на софтуерът за контрол на вградената система.

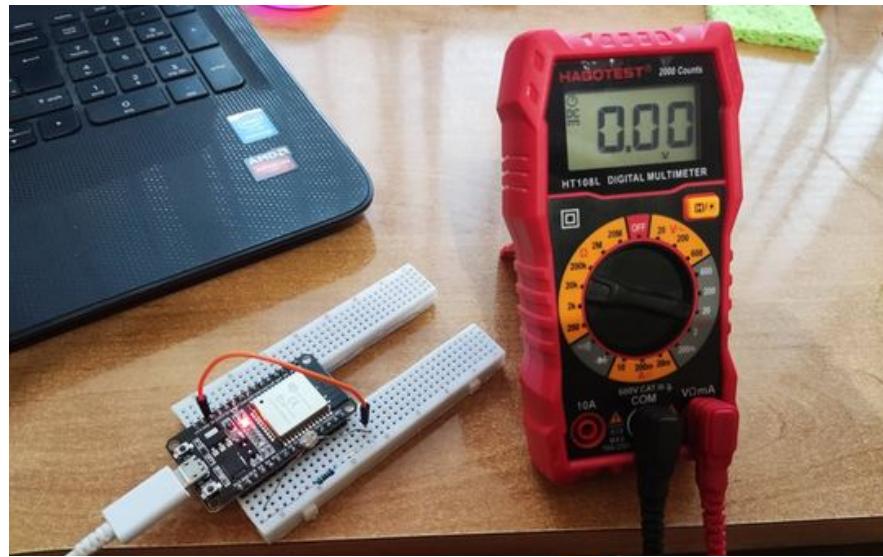
За разработка на програмен код и програмиране на ESP32 посредством Arduino IDE се налага кратка конфигурация на развойната среда. Налага се да се посочи линк към файл съдържащ инструкции за качване на програмни файлове от компютър на ESP32, посредством Arduino IDE. Този файл е публично достъпен в GitHub. След

конфигурирането бяха заредени и качени на микроконтролера базови примери от Arduino IDE с цел да се провери неговата изправност.

6.2.3. Тестове с делители на напрежение

Редно е да се отбележи, че експериментите бяхи проведени с помощта на борд за размножаване на електрическите връзки без запояване (solderless breadboard) и конектори с накрайници наподобяващи рейка, за многократна употреба (jumper wires). Това е стандартен подход при разработката на вградени системи, който позволява лесно бързо свързване на компонентите и безпроблемно променяне на връзките.

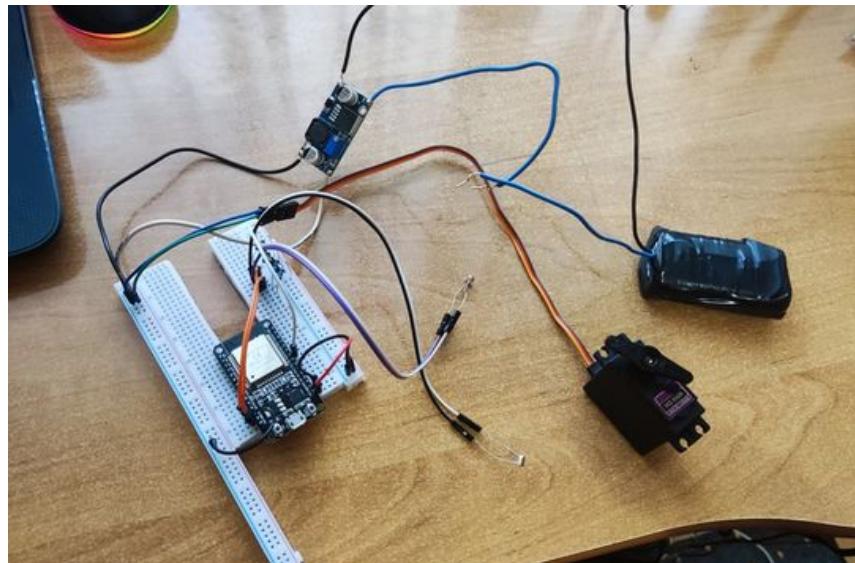
С тези експерименти се разработиха прототипите на делителите на напрежение нужни за измерване на фоторезисторите, чрез сравняването на които ще се открива източника на светлина, и главният делител на напрежение, който позволява измерването на напрежението на слънчевият панел, което само по себе си надвишава обхвата на вградените АДЦ на микроконтролера. На фигура 6.3 е показан кадър от работният процес. Чрез практически тестове беше установено, че фоторезисторите имат значително различни съпротивления при еднакво осветяване, което е характерно за този тип компоненти, и бяха подбрани най-дближаващите се по стойност такива, което в комбинация със софтуерни корекции решава проблема.



Фиг. 6.3. Кадър от процеса по тестване делителите на напрежение и
фоторезисторите

6.2.4. Тестове със серво моторите

За контрол на серво мотори изпозвайки ESP32 се налага изтеглянето на библиотека `ESP32Servo.h`, която предоставя нужния интерфейс за управление. След успешното измерване на съпротивлението на фоторезисторите беше разработена функция, която да ги сравнява по двойки и да завърта свързаните сервомотори съответно според това кой от двата фоторезистора има по-малко съпротивление, иначе казано е по-осветен. За захранването на сервомоторите се наложи свързването изграждането на прототип на захранването, поради причината, че микроконтролерът няма капацитет да им осигури захранване безопасно. На фигура 6.4 е показан кадър от работния процес.



Фиг. 6.4. Кадър от процеса по тестване на серво мотор в комбинация със фоторезистори

6.2.5. Тестове с компонентите от блок индикация

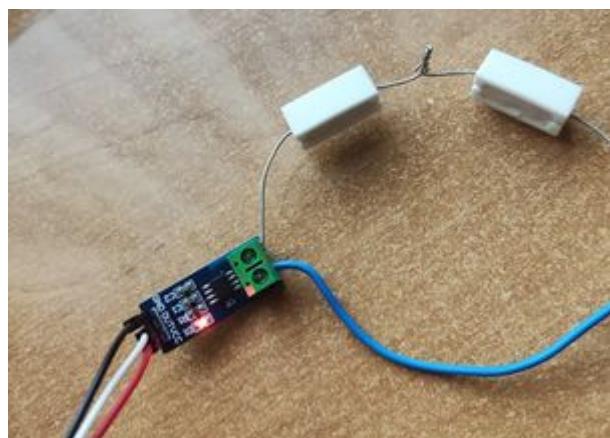
За да бъде контролиран LCD еcran с помощта на I²C адаптер се налага изтеглянето на библиотека LiquidCrystal_I2C.h. Командите са аналогични на тези използвани дори без I²C преходник, което направи работата лесна. При практическите тестове възника проблем при, който не се принтираха символи на экрана. Причината за него беше твърде високата честота на бодовете на изпращане на информация, която е зададена по подразбиране. След намаляването ѝ дисплеят работеше правилно както е показано на фигура 6.5. Функции за принтиране принтиране на конкретни съобщения бяха разработени. При свързването и управлението на светодиодите не възникнаха проблеми.



Фиг. 6.5. Кадър, показващ работещият LCD

6.2.6. Тестове със сензора на ток и температура

След направено проучване започнаха експериментите със сензора за ток и температура. Амперметърът беше тестван с помощта на източник на стабилно напрежение и показаните на фигура 6.6 резистори с ниско съпротивление и висока мощност. След практическите тестове към формулата за пресмятане на тока бяха направени леки изменения за да се компенсират неточностите в системата. Сензорът за температура беше тестван в стая с температура на въздуха от 22°C, като показа задоволителна точност.



Фиг. 6.6. Кадър, показващ сензора на ток по време на тестване

6.3. Тестване на софтуерните алгоритми и компоненти

В тази точка е разгледан процеса по разработка и свързване на софтуерните компоненти

6.3.1. Разработка и тестване на Django сървъра и базата данни

Първоначално бяха инициализирани базов Django проект и празна PostgreSQL база данни. След изтеглянето и инсталацирането на нужните библиотеки за сървъра, се пристъпи към свързването на базата данни и Django проекта. След това бяха създадени моделите и бяха мигрирани към базата, където по моделите се генерираха таблици. Накрая беше направен контролер, който да приема заявки на конкретен адрес, който да е достъпен за микроконтролера. Тестове бяха проведени с помощта на софтуерът Postman, който позволява изпращане на HTTP заявки с максимален контрол върху променливите. На фигъра 6.7 може да се види дневник на работещият сървър.

```
(venv) D:\S_tasks\DR_Martin_Georgiev_2022\Django_Postgre_Server>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

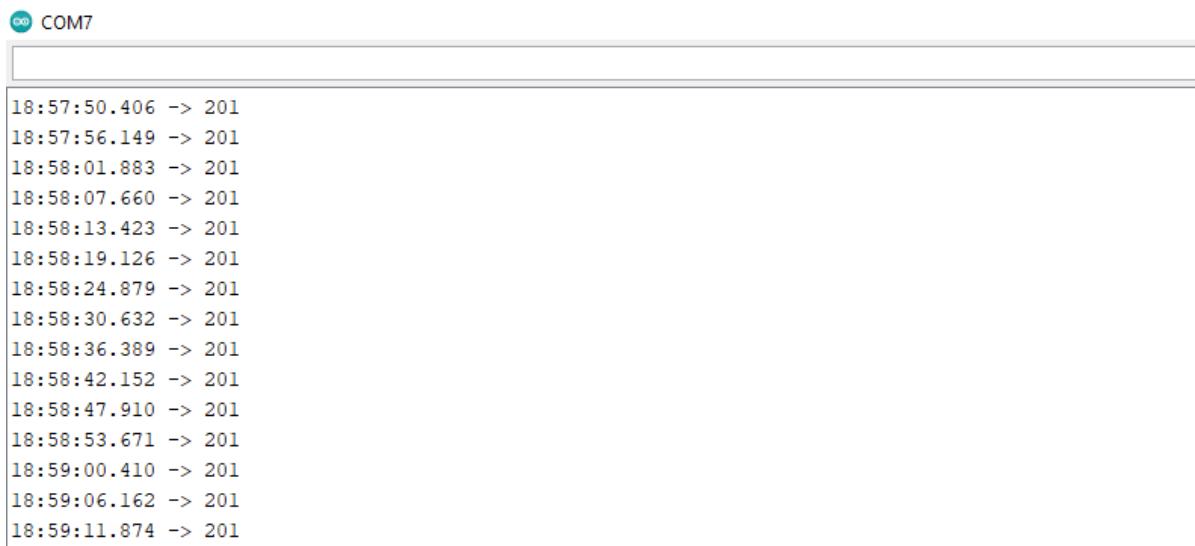
System check identified no issues (0 silenced).
March 30, 2022 - 18:18:54
Django version 4.0.3, using settings 'Django_Postgre_Server.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[30/Mar/2022 18:20:16] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:21] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:24] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:26] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:28] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:29] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:31] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:32] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:34] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:36] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:37] "POST /esp_handler HTTP/1.1" 201 22
[30/Mar/2022 18:20:39] "POST /esp_handler HTTP/1.1" 201 22
```

Фиг. 6.7. Екран показващ дневник на получените от сървъра заявки

6.3.2. Разработка и тестване на HTTP заявките от ESP32

За свързване към интернет и изпращане на HTTP заявки с помощта на комуникационния модул на ESP32 се наложи използването на библиотеки WiFi.h и HTTPClient.h. Първоначалните проблеми при пакетирането на данните в JSON бяха решени след направено

проучване. Възникналият проблем с изпращане на неуспешни заявки бе решен след като Django сървъра бе отворен за заявки идващи от локалната мрежа. С решаването на тези проблеми, получените данни от измерванията могат да достигат до сървъра и от там да се записват в базата данни.



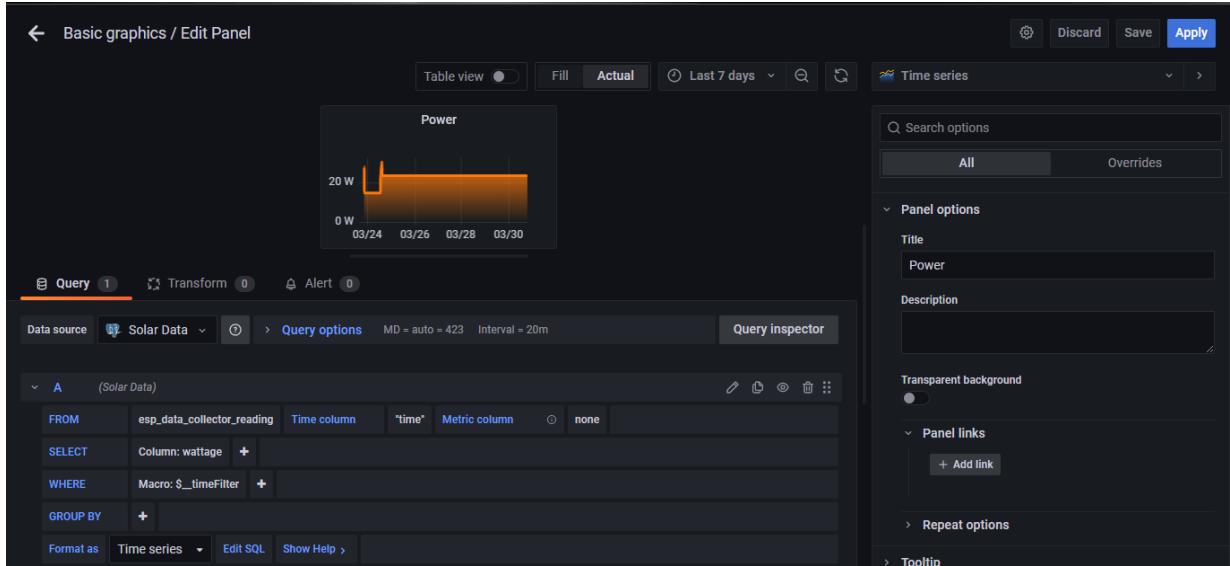
```
COM7
18:57:50.406 -> 201
18:57:56.149 -> 201
18:58:01.883 -> 201
18:58:07.660 -> 201
18:58:13.423 -> 201
18:58:19.126 -> 201
18:58:24.879 -> 201
18:58:30.632 -> 201
18:58:36.389 -> 201
18:58:42.152 -> 201
18:58:47.910 -> 201
18:58:53.671 -> 201
18:59:00.410 -> 201
18:59:06.162 -> 201
18:59:11.874 -> 201
```

Фиг. 6.8. Екран показващ дневник на отговорите на изпратените от ESP32 заявки

6.3.3. Разработка на информационни табла в Grafana

След кратко проучване започна работата по създаването на информационните табла в Grafana. След инициирането на връзката между базата данни и Grafana започна процесът включващ избиране на подходящ дизайн за панелите, кои данни и по какъв начин да бъдат групирани. Във всеки панел от таблото се наложи изграждането на SQL заявка която да взима нужната информация, което беше улеснено от интерфейса на Grafana, както е показано на фигура 6.9. С помощта на Системата за Географска Фотоволтаична Информация на ЕС [53] бе изчислена базова линия за теоретичното количество енергия на квадратен метър което би трябвало да се получава на нашата

географска ширина. След създаването на панелите те бяха стилизиирани.



Фиг. 6.9. Екран показващ средата за конфигуриране на панели в Grafana

6.3.4. Разработка на потребителско приложение с React и свързването му с Grafana

Първоначално бяха инициализирани нов React проект и стандартното приложение по подразбиране. Ненужни файлове бяха премахнати за опростяване на файловата структура и бяха избрани име и икона на таба на приложението. След експерименти с вграждането на Grafana панели в приложението бе установено, че най-удачният подход е изпъзването на таг `<iframe>`, но за целта бяха добавени допълнителни аргументи към генерираният от Grafana HTTP линк, които позволяват автоматично опресняване на панелите и разширяване на показания времеви диапазон. Последно беше направена стилизацията на страницата. На фигура 6.10. се вижда екран от конзолата след стартирането на приложението.

```
Starting the development server...
Compiled successfully!

You can now view react_frontend in the browser.

  Local:          http://localhost:3001
  On Your Network: http://192.168.0.13:3001

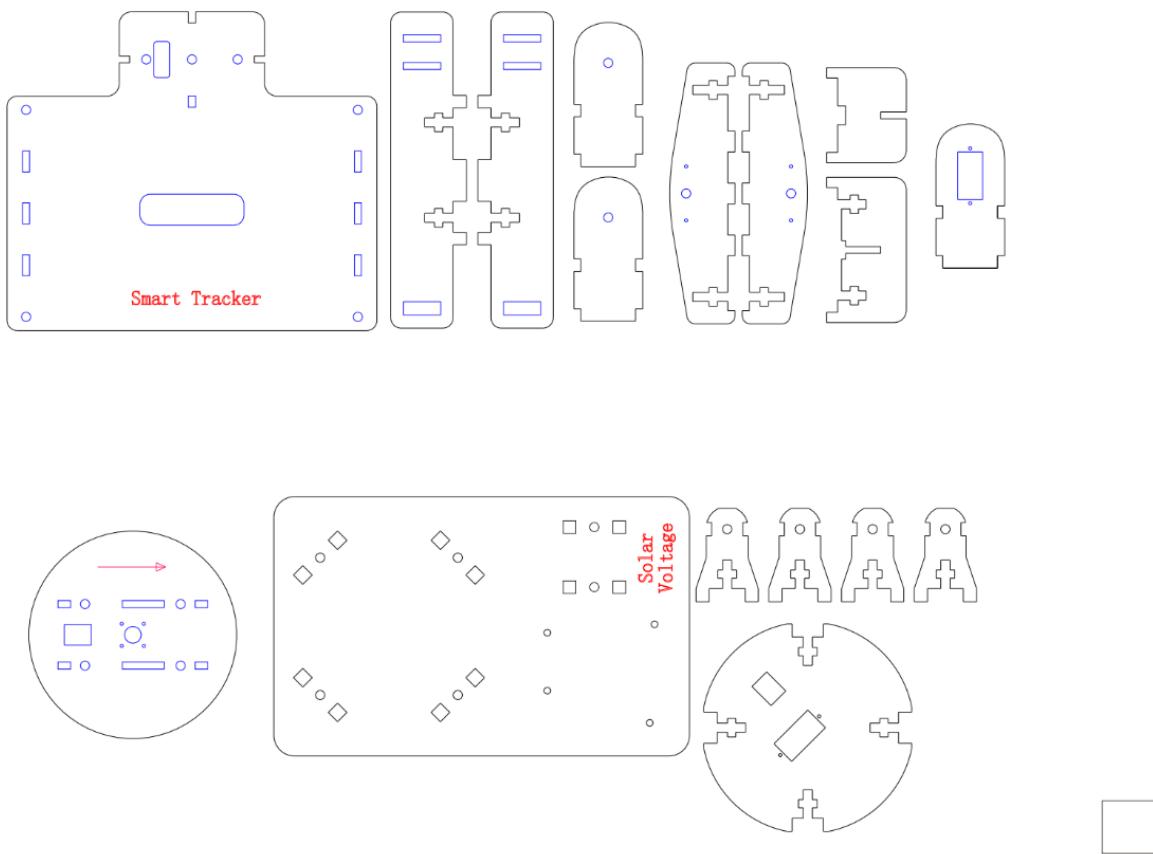
Note that the development build is not optimized.
To create a production build, use npm run build.

asset static/js/bundle.js 1.47 MiB [emitted] (name: main) 1 related asset
asset index.html 398 bytes [emitted]
asset asset-manifest.json 190 bytes [emitted]
cached modules 1.36 MiB [cached] 104 modules
runtime modules 28.2 KiB 13 modules
webpack 5.70.0 compiled successfully in 8800 ms
```

Фиг. 6.10. Екран от конзолата показващ съобщенията след стартирането на React приложението

6.4. Изграждане на конструкцията

За разработката на конструкцията беше взаимстван публично достъпен, свободен за ползване дизайн, като корекции бяха необходими, за да се компенсира за използването на различни компоненти. Първата необходима промяна беше преобразуването на файла от тип SVG в тип DXF, поради проблеми с мащабирането. След което бяха оразмерени и редактирани някой от компонентите, за да съвпаднат с използваните по-големи соларен панел и задвижващи серво мотори. На фигура 6.11 е показан чертежът на компонентите в като долния десен ъгъл е даден квадрат със страна един инч за справка.



Фиг. 6.11. Чертеж показващ компонентите на конструкцията

Заключение

В настоящата дипломна работа е разработена система за активно следене и оценка на слънчевата енергия, чиято цел е да събира информация за енергията, получена от слънцето чрез използване на соларен панел, насочван към слънцето. Тази информация се съхранява за бъдещ анализ и сравнение с други системи от този вид, като е предвидено да се визуализира за потребителите в подходящ графичен вид.

В увода на дипломната работа е поставена целта ѝ и са набелязани начините, по които тя ще се реализира. В глава първа е направено целенасочено проучване на съществуващите подобни системи и са структурирани както основните им предимства, така и най-често използваните технологии и компоненти.

С оглед на направеното проучване, във втора глава са поставени архитектурните и функционални изисквания към системата.

Дефинирани са и както общата блокова схема на проекта, така и схемите на отделните блокове, включващи отделните функционалните им особености и връзките между тях.

В глава трета е направен обоснован избор на необходимите компоненти според техните характеристики и са синтезирани принципните електрически схеми на отделните блокове и на цялата система. Проектирана е и печатната платка на системата, като са дефинирани и спазвани определени основни изисквания към нея.

В пета глава е представена общата структура на софтуера на системата и неговите компоненти, отговарящи за управление на вградената система, както и на получаването, съхраняването и визуализацията на информацията за получената слънчева енергия.

Глава шеста описва работния процес по практическото създаването на системата, тестването на хардуерните компоненти, свързването на

електрическите схеми и синтезирането и тестването на софтуерните компоненти и връзките между тях.

С оглед на всичко описано по-горе може да се направи извода, че заданието и основната цел на дипломната работа са изпълнени.

Като бъдещо развитие на проекта може да се реализира IoT мрежа от системи от този вид, която да предоставя по-комплексна информация за слънчевото лъчение и разликите в КПД между обикновените системи и тези, които следят активно слънцето. Възможно е да се осъществи хардуерна и софтуерна поддръжка на протокол за комуникация с по-голям обхват на действие, което ще позволи да се правят измервания в точки, до които не е изградена комуникационна инфраструктура. Може да се разработи система за прецизна калибрация и нивелация на продукта, с цел определяне на най-ефективният ъгъл за следене на слънцето. Възможно е да се добавят и допълнителни сензори, измерващи слънчевата радиация и UV лъчи, за да се получи по-разширена картина на получаваната слънчева енергия. Допълнително би могло да се помисли и за система за почистване на слънчевият панел, за да се поддържа постоянно максимален КПД.

Използвани съкращения

ENG

IoT - Internet of Things

TI - Texas Instruments

MPPT - Maximum Power Point Tracking

NA - Not Available (No information)

PWM - Pulse-width modulation

PVC - Polyvinyl Chloride

SoC - System on a Chip

RAM - Random Access Memory

LPDDR2 - Low-Power Double Data Rate

GPIO - General Purpose Input Output

DC - Direct Current

VDC - Volts of Direct Current

AC - Alternating current

VAC - Volts of Alternating Current

WiFi - Wireless Fidelity

BLE - Bluetooth Low Energy

LoRa - Long Range

NB-IoT - Narrow Band IoT

LED - Light Emitting Diode

LCD - Liquid crystal display

OLED - Organic Light Emitting Diode

OS - Operating System

RTOS - Real Time Operating System

IDE - Integrated Development Environment

CNC - Computer Numerical Controlled

NTC - Negative Temperature Coefficient

I2C - Inter-Integrated Circuit

BMS - Battery Management System
PCM - Protection Circuit Module
THO - Through-Hole
API - Application Programming Interface
TWI - Two-Wire Interface
MVC - Model-View-Controller
HTTP - HyperText Transfer Protocol
CSRF - Cross-Site Request Forgery
DOM - Document Object Model

BG

КПД - коефициент на полезно действие
IoT - интернет на нещата
TI - компания "Тексас Инструментс"
MPPT - следене на точката с максимална изходна мощност
НА - няма достъп, няма информация
PWM - широчинно импулсна модулация (ШИМ)
PVC - Поливинилхлорид
SoC - Система на чип
RAM - оперативна памет
LPDDR2 - двоен поток на данни с ниска консумация 2 генерация (вид RAM)
GPIO - входно-изходни с общо предназначение (относно пинове)
DC - Постоянен ток
VDC - Волтове постоянен ток
AC - Променлив ток
VAC - Волтове променлив ток
LED - Светодиод
LCD - Течноокристален дисплей
OLED - Органичен светодиод

OS - Операционна система
ОС - Операционна Система
IDE - Интегрирана среда за разработка
ВМКС - Вградени Микроконтролерни Системи
CNC - Компютърно Цифрово управление
NTC - Негативен коефициент на температура
I2C - Между интегрална връзка
BMS - Система за управление на батерията
PCM - Модул за предпазване на веригата
THO - метод, при който компонентите са закрепени към платката през дупки
API - Програмен интерфейс
TWI - Двупроводен интерфейс
MVC - Модел-Изглед-Контролер
HTTP - Протокол за прехвърляне на хипертекст
CSRF - Фалшифициране на между сайтови заявки
DOM - Документен обектен модел

Използвана литература

1. [Energy Production and Consumption](#)
2. [Ranking Renewable and Fossil Fuels on Global Warming Potential](#)
3. [Cost of solar in 2021](#)
4. [The Five Main Reasons Homeowners Don't Buy Solar](#)
5. [Common Roof Solar Panel Problems](#)
6. [Solar Tracker V2.0](#)
7. [Self-Powered Solar Data Logger](#)
8. [Solar Tracker with Live Data Feed - Windows IoT](#)
9. [Floating Sun Tracker Hydraulic Solar Panel](#)
10. [IoT Solar Panel Power Monitoring using ESP32 and ThingSpeak](#)
11. [Solar panel CL-SM5P, 5W, 0.28A, 251x186x18mm](#)
12. [Соларен панел, фотоволтаичен elektronikabg.com-30401288](#)
13. [Монокристален Фотоволтаичен Панел solarhouse.bg-12345-015-1](#)
14. [Понижаваш преобразувател на напрежение LM2596](#)
15. [Модул DC/DC AC/DC конвертор понижаващ 12V 2A](#)
16. [Classic servos 9g SG90 / MG90S](#)
17. [Twotrees Stepper Motor Nema 17](#)
18. [13KG Servos Digital MG996](#)
19. [Raspberry Pi Zero 2 W](#)
20. [Arduino Nano 33 IoT](#)
21. [NodeMCU-32S - WiFi, Dual mode Bluetooth, ESP32 CPU](#)
22. [Which IoT Communication Protocol is Suitable for your Project?](#)
23. [Cellular IoT: What is the difference between NB-IoT and LTE-M?](#)
24. [WiFi for IoT](#)
25. [Дисплей LCD 16x2 - син](#)
26. [Дисплей OLED 0.96" 128x64 - white, SSD1306](#)
27. [Дисплей E-Ink 1,54" 200x200, SPI /e-Paper module/](#)
28. [AC-DC Buck Converter AC 220v to 5v](#)

29. [Шунт VL-2, 20A, 60mV](#)
30. [Сензор за измерване на ток ACS712](#)
31. [СЕНЗОР ЗА НАПРЕЖЕНИЕ 0-25V](#)
32. [Thermistor, NTC, 100 kOhm, 2.54 mm](#)
33. [Temperature sensor IC LM335AZ](#)
34. [Light Dependent Resistor LDR: Photoresistor](#)
35. [What is an Embedded OS? – Design Support](#)
36. [Definition - Embedded Application](#)
37. [Bare-metal and RTOS Based Embedded Systems](#)
38. [Python vs. C/C++ in embedded systems](#)
39. [Lua: about](#)
40. [10 Popular Javascript Frameworks for Mobile App Development](#)
41. [Relational vs. Non-Relational Database: Pros & Cons](#)
42. [InfluxDB](#)
43. [PostgreSQL](#)
44. [What Makes a Good User Experience?](#)
45. [Grafana](#)
46. [Kibana](#)
47. [Types of Materials](#)
48. [3D Printing vs CNC Machining: Which is best for prototyping?](#)
49. [CircuitMaker](#)
50. [Аналогов температурен сензор KY-013](#)
51. [I2C адаптер за LCD](#)
52. [Линк към хранилището с програмен код](#)
53. [Сайт за Географска Фотоволтаична Информация](#)