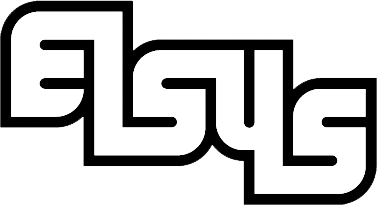
**Технологично училище**

**„Електронни системи“ към**

**Технически университет – гр. София**

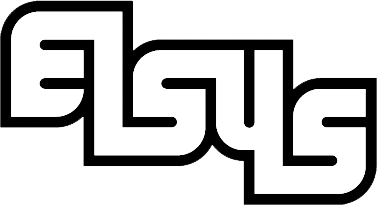
**КУРСОВА РАБОТА**

**Тема:**

**Увод във вградените микрокомпютърни системи**

**Курсист: Оценител:**

Калоян Дойчинов Енчо Шахънов

**Технологично училище**

**„Електронни системи“ към**

**Технически университет – гр. София**

**Дата на заданието:** 10.04.2022г. **Утвърждавам:**………………….

**Дата на предаване:** 10.04.2022г. /г-н Енчо Шахънов/

**ЗАДАНИЕ**

**за курсова работа**

**на ученика Калоян Стефанов Дойчинов от 10а клас**

1. **Тема:** Дистанционно управление на компютър
2. **Изисквания**
   1. Разработване на работещ хардуерен проект, включващ използването на Arduino платка
   2. Разработване на система, която управлява работното състояние на компютър и отчита данни като температура и влага
   3. Визуализиране на данни като измерената температура и влага в компютърната кутия

**Курсист:**……………………

**/Калоян Дойчинов/**

**Оценител:**………………….

/г-н Енчо Шахънов/

**Съдържание**

[**УВОД** - 4 -](#_Toc100528013)

[**ПЪРВА ГЛАВА** - 5 -](#_Toc100528014)

[**ХАРДУЕРНИ КОМПОНЕНТИ, ИЗПОЛЗВАНИ В РЕАЛИЗАЦИЯТА НА ПРОЕКТА** - 5 -](#_Toc100528015)

[**1.** **Избиране на Arduino платка** - 5 -](#_Toc100528016)

[**2.** **Избиране на компонент, чрез който ще се включва и изключва компютъра** - 6 -](#_Toc100528017)

[**3.** **Избиране на сензор за температура и влага** - 7 -](#_Toc100528018)

[**4.** **Избиране на компонент за получаване и изпращане на данни към клиента** - 8 -](#_Toc100528019)

[**5.** **Избиране на захранване** - 10 -](#_Toc100528020)

[**ВТОРА ГЛАВА** - 12 -](#_Toc100528021)

[**ЕЛЕКТРИЧЕСКА И БЛОК СХЕМИ НА ПРОЕКТА** - 12 -](#_Toc100528022)

[**1.** **Блок схеми на работа на проекта** - 12 -](#_Toc100528023)

[**2.** **Електрическа схема на проекта** - 13 -](#_Toc100528024)

[**ТРЕТА ГЛАВА** - 14 -](#_Toc100528025)

[**РЕАЛИЗАЦИЯ И РАЗРАБОТВАНЕ НА ПРОЕКТА** - 14 -](#_Toc100528026)

[**1.** **Свързване на всички компоненти в една електрическа схема** - 14 -](#_Toc100528027)

[**2.** **Описание на софтуерната част на проекта** - 14 -](#_Toc100528028)

[**2.1** **Файлова структура на проекта** - 14 -](#_Toc100528029)

[**2.2** **Софтуерът за Arduino платката** - 14 -](#_Toc100528030)

[**2.3** **Софтуерът за сървъра** - 21 -](#_Toc100528031)

[**2.4** **Софтуерът за клиента** - 21 -](#_Toc100528032)

[**3.** **Създаване на работещ модел на проекта на практика (снимки)** - 21 -](#_Toc100528033)

[**ЗАКЛЮЧЕНИЕ** - 24 -](#_Toc100528034)

[**ИЗПОЛЗВАНА ЛИТЕРАТУРА** - 25 -](#_Toc100528035)

# **УВОД**

С напредването на технологиите и вграждането им в нашия свят все повече и повече неща трябва да се контролират от далеч – компютрите на работниците в различни фирми, които искат да позволят отдалечен достъп на своите служители до хората, които трябва да достъпят някакъв важен файл, който са забравили на настолния си компютър и биха могли да се свържат безжично или до съвсем другата крайност – на хората с домашни или отдалечени DIY сървъри. Тези хора изключително много разчитат на отдалечената си връзка с подобни компютри, което ги прави изключително уязвими, когато примерно забравят да си ги включат, токът падне за няколко минути и след това да трябва някой да ги включва ръчно или просто искат да автоматизират включването на дадените компютри. Тази кросова работа има за цел да реши този проблем.

# **ПЪРВА ГЛАВА**

# **ХАРДУЕРНИ КОМПОНЕНТИ, ИЗПОЛЗВАНИ В РЕАЛИЗАЦИЯТА НА ПРОЕКТА**

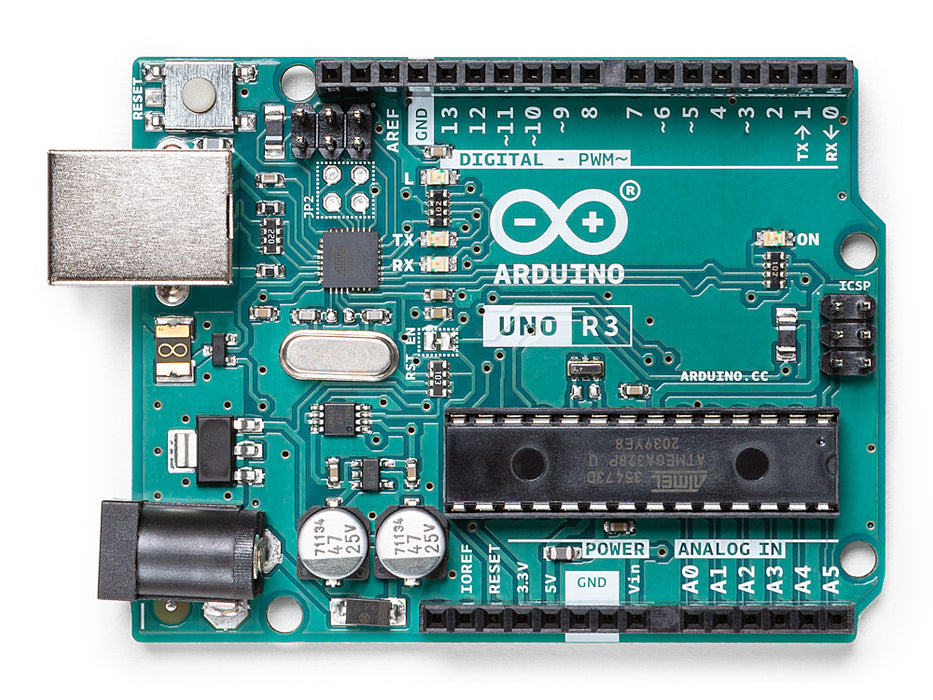
# **Избиране на Arduino платка**

Arduino Uno R3

Arduino Uno е най–достъпната, удобна и лесна за програмиране и употреба платка за хора с малко или без никакъв опит в програмирането на хардуер. Това е най–разпространената Arduino платка и поради тази причина има изключително много информация за това как работи и как трябва да се програмира в интернет. Освен това заедно с платката е предоставена и лесна и интерактивна среда за програмиране – Arduino IDE, осигуряваща всичко необходимо откъм допълнителни библиотеки и софтуер свързани с програмирането на платката – именно тези фактори я прави изключително добър избор за начинаещи – от малки до големи.

Като заради горепосочените причини в този проект ще бъде използвана Arduino Uno Rev. 3 (*фиг. 1.1а*), която е най-новата разновидност на Arduino Uno. Аrduino Uno R3 е базирана на микроконтролера ATmega328P и има 16 MHz резонатор. Микроконтолерът не е запоен за платката и това позволява да бъде сменен много лесно в случай, че се случи нещо или просто искаме да ползваме Arduino платката като USB Bus.

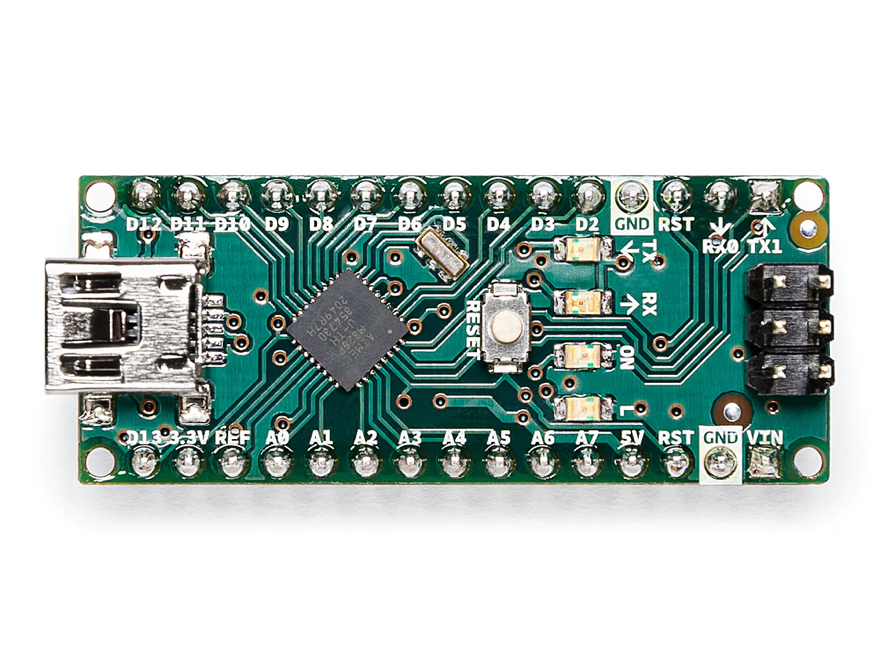
Тя има общо 14 цифрови пина, които могат да служат както за входове, така и за изходи. Освен това има и 6 аналогови, които обаче не са използвани в този проект. Те обаче все пак могат да бъдат използвани и като цифрови, обаче приемането и изпращането на цифрова информация чрез тях става по-бавно, отколкото при използването на цифровите.



*Фиг. 1.1а – Arduino Uno Rev. 3*

Arduino Nano

Друго отлична платка за начинаещи е Arduino Nano (*фиг. 1.1б*). Главната разлика между тази платка и Arduino Uno Rev 3 е, че заема сравнително малка площ, която би била прекрасна също и за този проект, въпреки че би била малко по-неудобна за използване по време на инсталация.



*Фиг. 1.1б – Arduino Nano*

# **Избиране на компонент, чрез който ще се включва и изключва компютъра**

Релето е електрически задвижван ключ. Те обикновено използват електромагнит (намотка), за да управляват вътрешния си механичен превключващ механизъм (контакти). Когато релейният контакт е отворен, това ще включи захранването за верига, когато бобината е активирана.



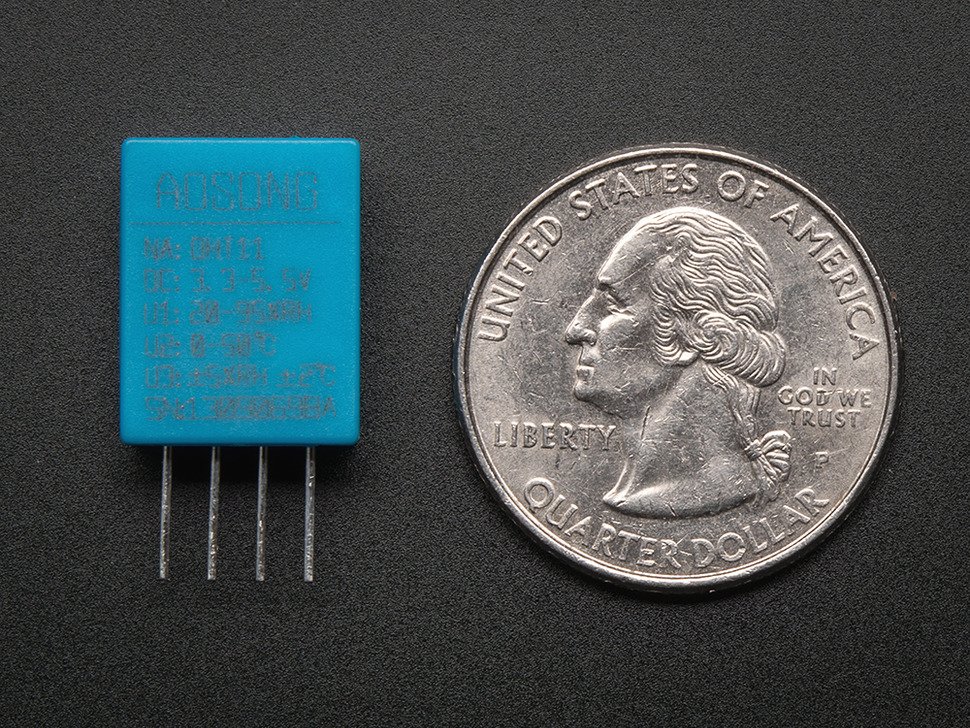
*Фиг. 1.2 - Реле*

# **Избиране на сензор за температура и влага**

DHT11 срещу DHT22

DHT11 е изключително прост и изгоден на цена цифров сензор за измерване температура и влажност. Той използва капацитивен сензор за влажност и термистор за измерване на околния въздух и изпраща сигнал върху цифровите пинове за данни, правейки го още по-лесен за използване, като ни съкращава от изчисляването по АЦП уравнението. Единственият реален недостатък на този сензор е, че може да се получават нови данни от него само веднъж на всеки 2 секунди, което в някои ситуации може да е фатално.

В сравнение с DHT22, този сензор е по-малко прецизен, по-малко точен и работи в по-малък диапазон от температура/влажност, но в същото време е сравнително по-малък и с по-ниска цена.



*Фиг. 1.3 – DHT11 и американски четвърт долар*

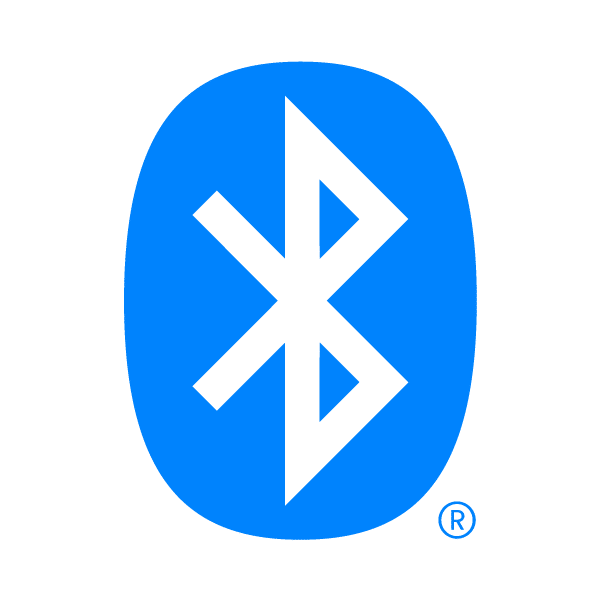
# **Избиране на компонент за получаване и изпращане на данни към клиента**

Протокол за връзка

Както Bluetooth, така и Wi-Fi се използват за осигуряване на безжична комуникация чрез радиосигнали. Най-голямата разлика между Bluetooth и Wi-Fi е, че Bluetooth всъщност е свикнал да свързва устройства с малък обсег за споделяне на информация, докато Wi-Fi се използва за предоставяне на високоскоростен достъп до мрежата / интернет.

Wi-Fi осигурява висока степен на информация, тъй като скоростта на интернет е жизненоважен проблем.

|  |  |
| --- | --- |
| Bluetooth | Wi-Fi |
| Bluetooth консумира малко енергия. | Докато Wi-Fi консумира висока мощност. |
| Сигурността на Bluetooth е по-ниска в сравнение с Wi-Fi. | Докато Wi-Fi осигурява по-добра сигурност от Bluetooth. |
| Bluetooth е по-малко гъвкаво означава, че се поддържат ограничени потребители | Докато Wi-Fi поддържа голям брой потребители. |
| Обхватът на радиосигнала на Bluetooth е десет метра. | Докато Wi-Fi обхватът може е достигне десетки метри, зависейки от честотната му лента. |
| Bluetooth изисква ниска честотна лента. | А пък Wi-Fi изисква висока честотна лента. |
| Bluetooth е т.нар. PAN – Personal Area Network – което е прекрасно за „смарт“ дистанционно, което се намира в стаята, но в нашия случай – не. | Wi-Fi може да се свърже към т.нар. WAN – Wide Area Network – което означава, че се свързва към интернет, правейки го така, че да може да го контролираме независимо къде се намираме. |

*Фиг. 1.4а – Bluetooth и Wi-Fi*

Wi-Fi модули

Има много ESP Wi-Fi модели, затова е важно да изберем най-подходящият Почти всички Wi-Fi модули от семейството на ESP ще свършат добре работата, но всеки от тях има свои собствени характеристики и спецификации.

A picture containing timeline

Description automatically generated

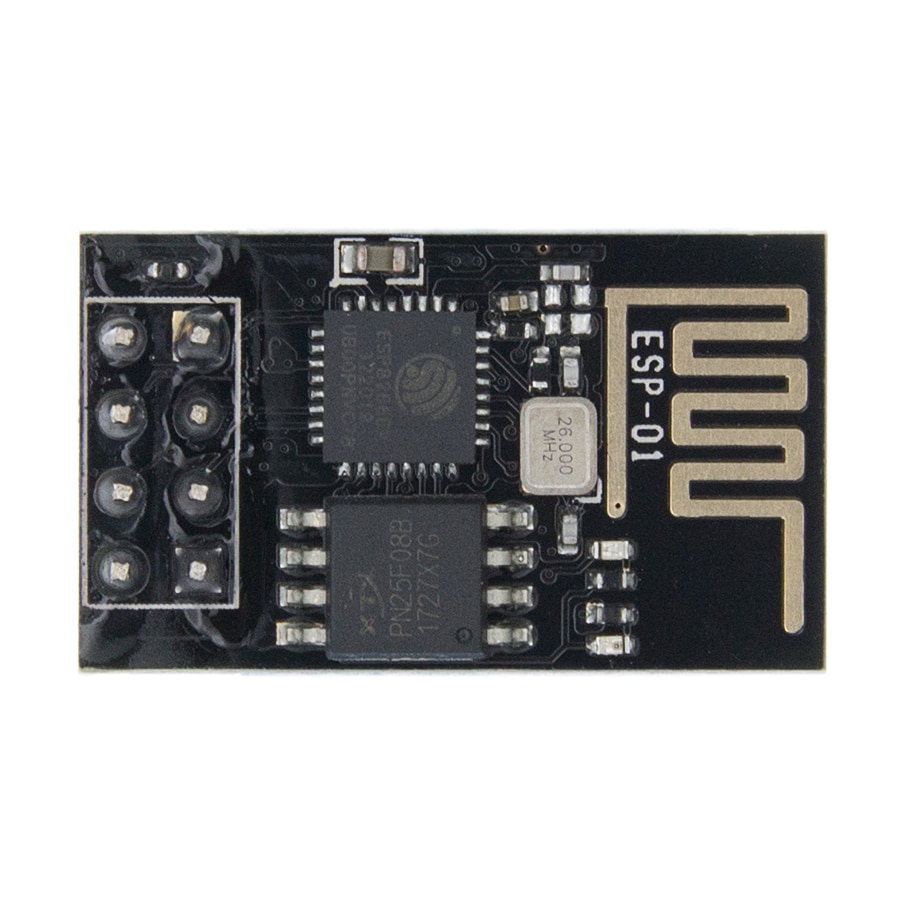
*Фиг. 1.4б – ESP семейството*

ESP-01 (ESP8266)

Според нуждите на този проект ESP8266-01 е най-подходящият поради много причини. ESP-01 е най-популярният Wi-Fi модул, поради ниската си цена. В резултат на това и огромна онлайн общност, в която може да се намери отговор на почти всеки въпрос или проблем, с който може човек да се сблъска.

ESP8266 ESP-01 или само ESP-01 е Wi-Fi модул, който позволява на микроконтролерите като Arduino Uno и др. достъп до Wi-Fi мрежа. Този модул е самостоятелен SоC (System On a Chip), който не се нуждае непременно от микроконтролер за манипулиране на входове и изходи, както обикновено правите с Arduino, например, защото ESP-01 действа като малък компютър. По този начин можем да дадем достъп до интернет на микроконтролер, както Wi-Fi щитът прави на Arduino, или можем просто да програмираме ESP8266 не само да има достъп до Wi-Fi мрежа, но и да действа като микроконтролер. Това прави ESP8266 много гъвкав и може да ви спести малко пари и място във вашите проекти.

Освен това е много лесен за използване с платката Arduino, тъй като е сериен wifi модул, той може да комуникира с платката Arduino през серийната комуникация. Той има вграден микроконтролер, което означава, че можете да го използвате като самостоятелен микроконтролер и wifi модул в едно комбо, което е супер готино. Има вградени два GPIO. Само с 1,5 $ ще получите всички тези функции, което всъщност е супер готино.



*Фиг. 1.4в*

# **Избиране на захранване**

След свързване на ESP-01 се установява как Arduino Uno не може да подава достатъчно захранване, чрез 3.3V си захранване. Като това може да се провери с мултицет, който измерва спад в 3.3V напрежение до около 2.2V ~ 2.4V. Заради това трябва да се сдобием със външно захранване. В нашия случай ще ни е най-удобно да си вземем бредборд захранване, което може чрез шунтиране на дадени негови пинове да го надстроим да подава или 3.3V, или 5V, зависейки за коя захранваща линия го надстроим. Бредборд захранването, използвано в този проект, е , което може да бъде захранено или с Mini USB -B, или чрез букса (5.5мм/2.1мм), като чрез вградени светодиоди може да се види дали получава захранване.



*Фиг. 1.5 – бредборд захранване*

Следователно този проект се захранва от 2 USB-та:

* USB-B на Arduino платката
* Mini USB-B на бредборд захранването

# **ВТОРА ГЛАВА**

# **ЕЛЕКТРИЧЕСКА И БЛОК СХЕМИ НА ПРОЕКТА**

# **Блок схеми на работа на проекта**

Diagram

Description automatically generated

*Фиг. 2.1*

# **Електрическа схема на проекта**

Diagram, schematic

Description automatically generated

*Фиг. 2.2*

# **ТРЕТА ГЛАВА**

# **РЕАЛИЗАЦИЯ И РАЗРАБОТВАНЕ НА ПРОЕКТА**

# **Свързване на всички компоненти в една електрическа схема**

Електрическата схема може да бъде видяна на *фиг. 2.2*

# **Описание на софтуерната част на проекта**

# **Файлова структура на проекта**

След като изтеглите проекта, ще видите дадените директории:

* hardware – тук се намират файловете, които съдържат кода, който се изпълнява върху Arduino платката
* web-app – тук се намират всички файлове за уеб клиента, написан на React.js
  + src – тук се намира цялата функционалност и дизайн на уеб приложението
  + public – тук се намира шаблонът, върху който се „рисува“ уеб приложението
* server – тук се намира целия код за express.js сървъра
* doc – оттук може да се изтегли / прочете тази документация
* assets
  + img – тук се намират всички снимки на проекта, както и използвани в readme.md файла в корена на тази файлова система
  + schematics – тук се намират всички схеми на проекта

# **Софтуерът за Arduino платката**

Първото нещо, с което ще започнем, са библиотеките (фиг. 3.2а), използвани в този проект. Като тук са следвани всички норми за писане на добре четлив код, като библиотеките са качени най-отгоре на *hardware.ino* файла. Библиотеките, които са използвани са:

* Arduino JSON – използвана за изпращане на данни като температура и влажност, като това улеснява процеса на клиентския софтуер, като пакетира данните в JSON обект.
* DHT и DHT\_U – това са библиотеките на DHT, които включват изключително важните функции, с които стартираме работата на DHT11, както и да четем информацията от него.
  + *„DHT dht(DHTPIN, DHTYPE);“* и *„dht.begin();“* - функциите за инициализиране и стартиране на DHT11 или DHT22
  + *dht.readHumidity();* и *dht.readTemperature();* - функциите за четене на влажност и температура от цифровите пинове
* Software Serial -е библиотека, която позволява серийна комуникация с цифров порт, различен от серийния порт.

A screenshot of a computer

Description automatically generated

*Фиг. 3.2а*

Собственоръчно направената локална библиотека „arduino\_secrets.h“ се използва за съхраняване на поверителна информация като Wi-Fi пароли, като в този случай се използва и за името на Wi-Fi не само неговата парола.

*Важно е да се спомене да не се качват публично паролите на Wi-Fi, а пък ако се стигне до там – git reset –hard винаги е опция.*

Съвет от автора на курсовата работа по личен опит с този проект

Text

Description automatically generated

*Фиг. 3.2б*

Отново спазвайки правилата за писане на чист и подреден код, пишем всичките ни константи и/или глобални променливи отгоре на файла, но под библиотеките. Като тези глобални константни променливи са пиновете на релето и DHT11 както и типът на DHT сензорът – DHT11. Другите глобални променливи са Wi-Fi името и паролата, които взимаме от *arduino\_secrets.h*, както и променливите, които са полезни по време на отстраняване на грешки.

След тях инициализираме глобално комуникацията с ESP-01 (ESP8266), DHT11 сензорът и създаваме JSON буфера, в който се ще се съхраняват данните за изпращане преди самото изпращане, като поддържа до 100 DHT сензора.

Text

Description automatically generated

*Фиг. 3.2в*

Тук накратко може да се видят декларациите на функциите (извън *setup* и *loop*), които са дефинирани след тях, поради улеснена четимост на кода.

Text

Description automatically generated

*Фиг. 3.2г*

В setup() се стартира всичко нужно за проекта – което реално представлява, че този код се изпълнява само веднъж при стартиране на Arduino платката или при качването на нов код. Инициализираме пина на релето, както и за всеки случай го слагаме на стойността му по подразбиране – HIGH, тъй като го ползваме във режим на „нормално отворена връзка“, тъй като компютрите се включват и изключват при натискане на бутон, който се зачита, като затворена връзка, за около (под) 1 секунда и под насила – за да пренебрегнем софтуерът на компютъра – около 7 секунди – което ще бъде разгледано по-надолу. След това се инициализира DHT сензорът, а след него започва серийната комуникация на Arduino-то и на ESP-01 на baudrate oт 115 200. След това се вика функцията esp\_setup(), която се използва за стартиране на ESP-01 и задаване на неговите променливи чрез изпращане на команди.

Text

Description automatically generated

*Фиг. 3.2г*

В *loop()* функцията (*фиг. 3.2е*) има най-много за обяснение, но тук ще бъде обяснено по-обобщено и не по редове, докато на снимката може да се видят всички коментари ред по ред на английски език за улеснение на интернационалният читател.

Първоначално започва с проверка дали ESP-01 е налично и работещо (дали работи правилно се проверява в по-надолу в кода). След това се задава отговор по подразбиране при заявка на ардуиното от клиента. Чрез следващата проверка „if(esp.find("+IPD,"))“ се проверява дали ESP-01 получава заявка от клиент. Ако е получило взема номера на връзката и след това проверява каква команда му е дадена, като тя може да бъде една от следните 3:

* 1 – включване / изключване на компютъра – релето шунтира пиновете на дънната платка за 1.5 секунди и след това се изпраща *ОК* на клиента
* 2 – вземане на данни от сензора – температура и влажност – след това се изпращат данните към клиента
* 3 – изключване насила – независимо от софтуерът на компютъра – релето шунтира пиновете на дънната платка за 7 секунди и след това се изпраща *ОК* на клиента

След това връзката приключва като успешна, а пък ако не – като не неуспешна.

Text

Description automatically generated

*Фиг. 3.2е*

Text

Description automatically generated

*Фиг. 3.2ж*

Text

Description automatically generated

*Фиг. 3.2з*

Text

Description automatically generated

*Фиг. 3.2и*

# **Софтуерът за сървъра**

# **Софтуерът за клиента**

# **Създаване на работещ модел на проекта на практика (снимки)**

A picture containing text, electronics

Description automatically generatedA picture containing electronics

Description automatically generatedA picture containing electronics

Description automatically generatedA picture containing electronics

Description automatically generated

# **ЗАКЛЮЧЕНИЕ**

В описаната курсова работа са изпълнени всички заложени изисквания, включително функционални изисквания към хардуерна реализация, софтуерна реализация и разработено устройство. Създадена е работоспособна система, тествана при неблагоприятни условия, преминала през няколко итерации на подобрение. Бъдещото развитие включва добавяне на английска документация и сензор за звук.

# **ИЗПОЛЗВАНА ЛИТЕРАТУРА**

Arduino Uno - <https://docs.arduino.cc/hardware/uno-rev3>

Кратка информация за продуктите - <https://elimex.bg>

DHT11 & DHT22 – <https://www.adafruit.com/product/386>

ESP - <https://www.esp8266.com>