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

# К У Р С О В П Р О Е К Т

Тема: Проектиране и реализация на контролер за отоплителна система

с GUI и Remote Access

|  |  |
| --- | --- |
| Ученици:  *Даниел Янев* | Научен ръководител:  *Инж. Росен Витанов* |
| *Денис Христов*  *Виктор Найчев* | *М. Инж. Кирил Митов* |

СОФИЯ

2018

**Assignment:**

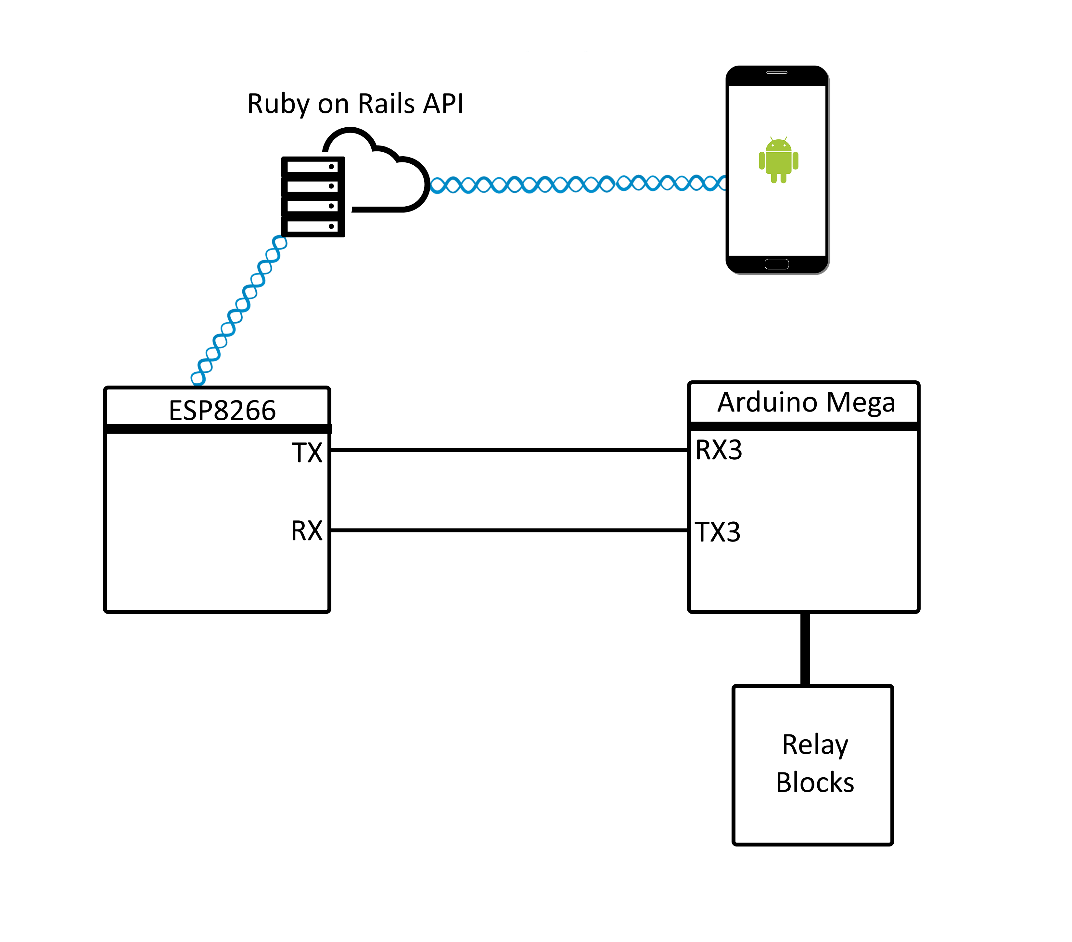
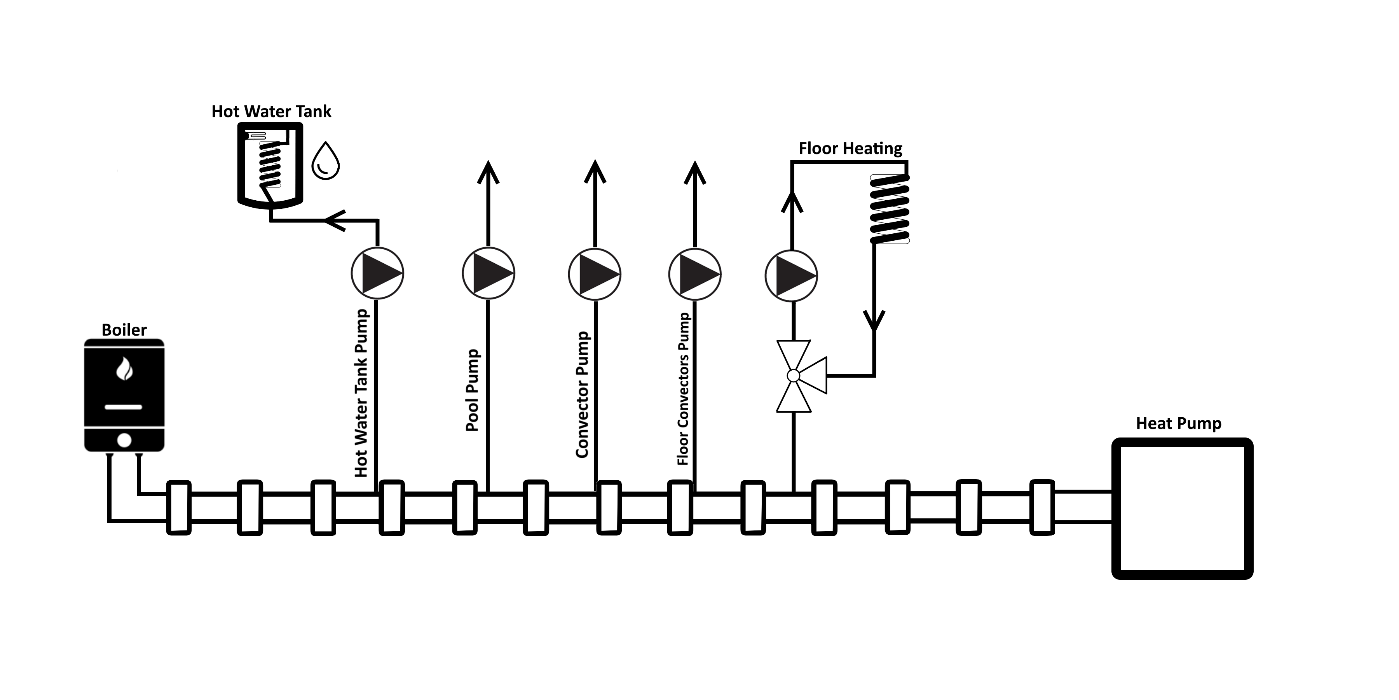
Developing an easy-to-use device for controlling complicated water-heating systems through relays, giving the end user the ability to easily manage the heating system by the provided local touchscreen display, or remotely done so by the developed mobile Android application.

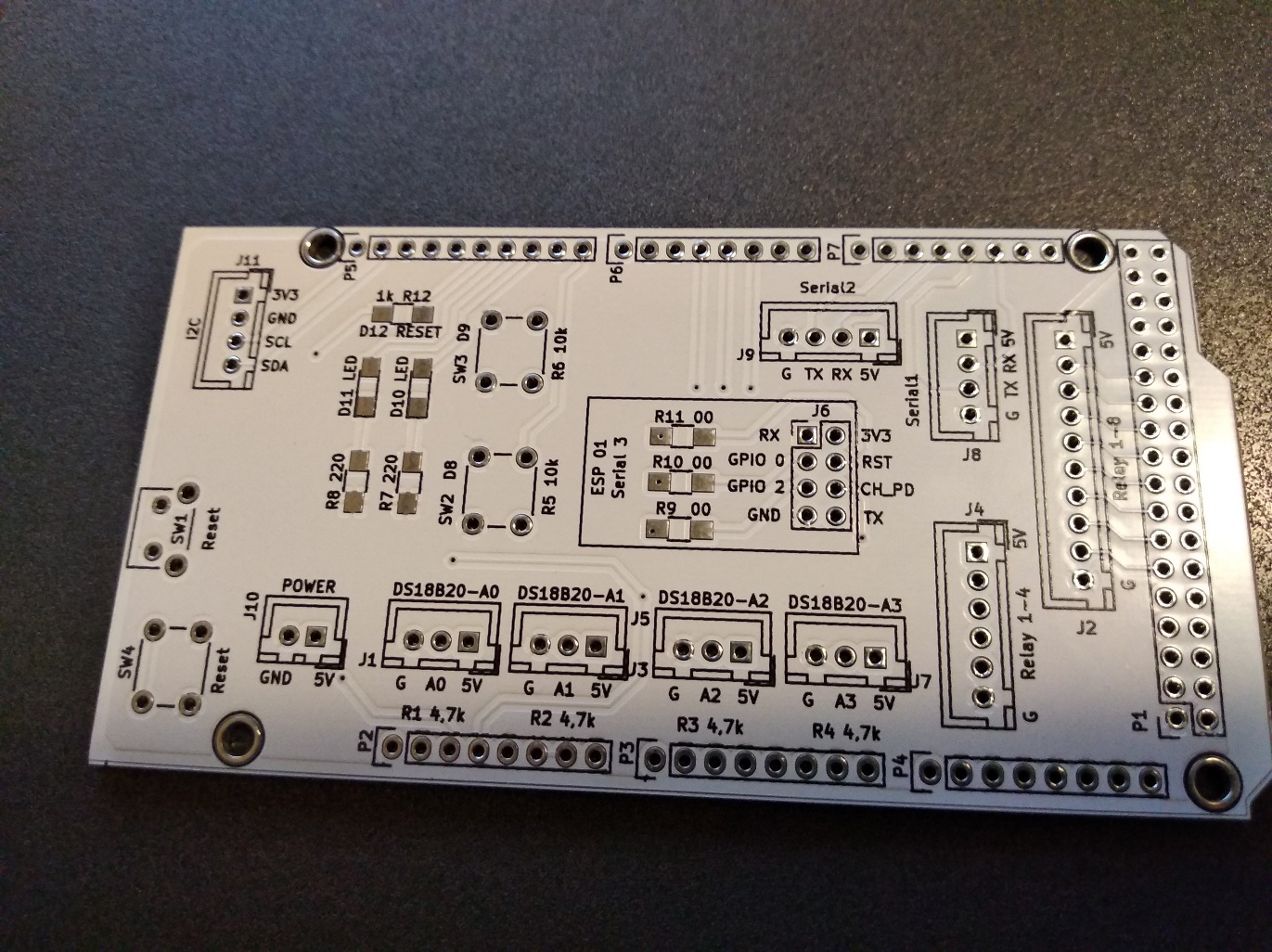
**GENERAL CHARACTERISTICS OF THE PROBLEM**

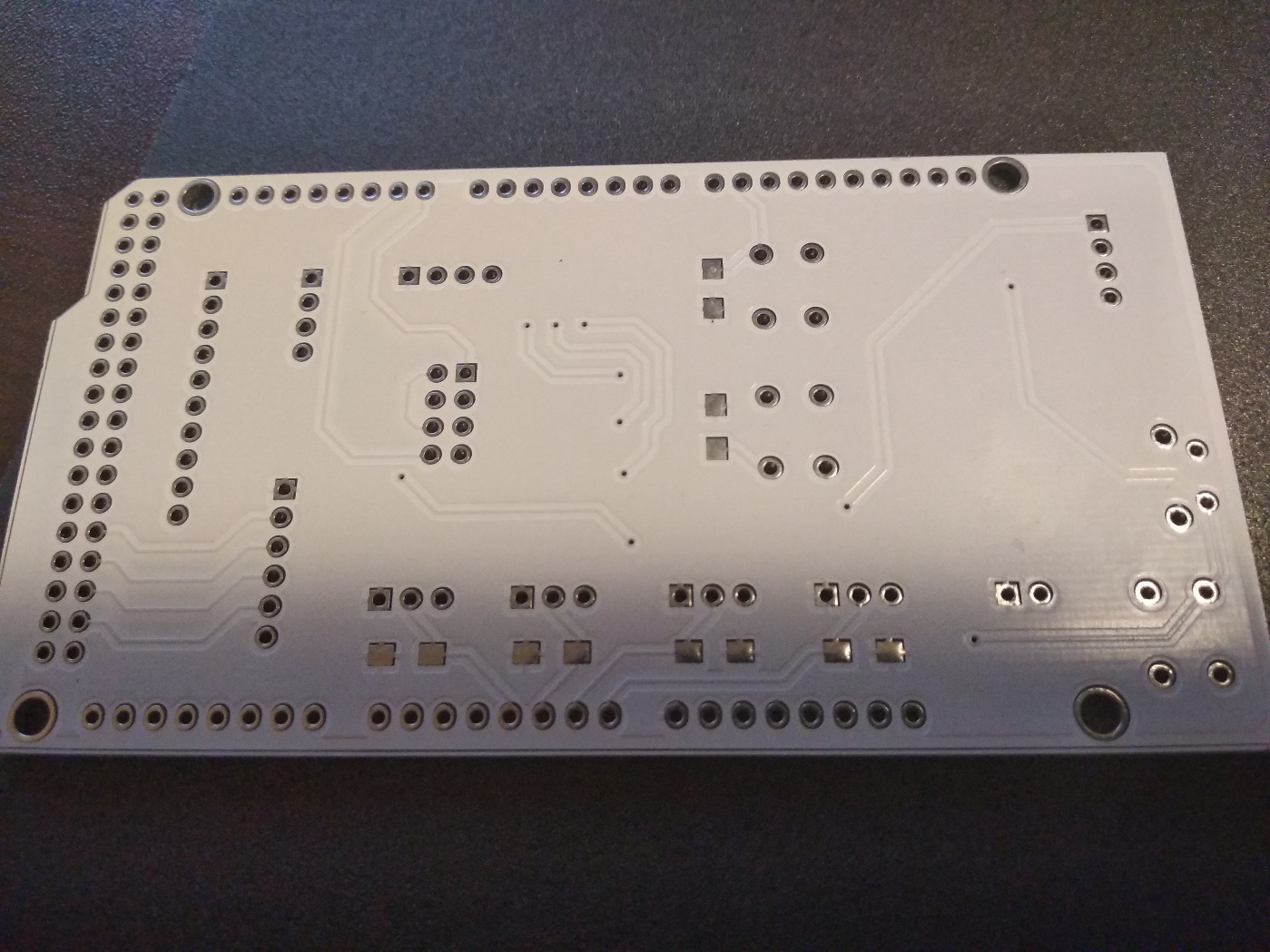
The product allows the end-user to easily and quickly control the system while being very efficient on resources such as time, electricity and water and taking fault proof into consideration.

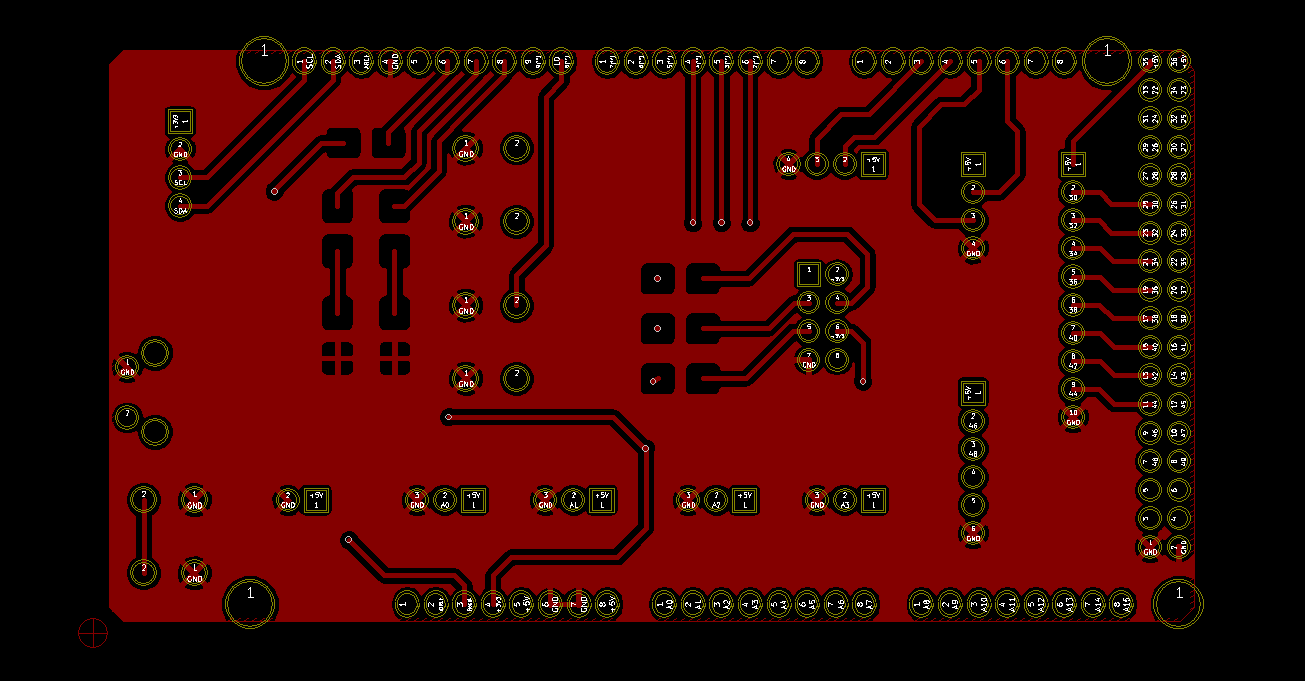
It’s especially useful when manging the heating system of a distanced house in an underpopulated location. Using this system frees you from the responsibility to go there and manually manage everything and allows you to quickly setup your heating system just in time before you arrive as if you were just there yesterday.

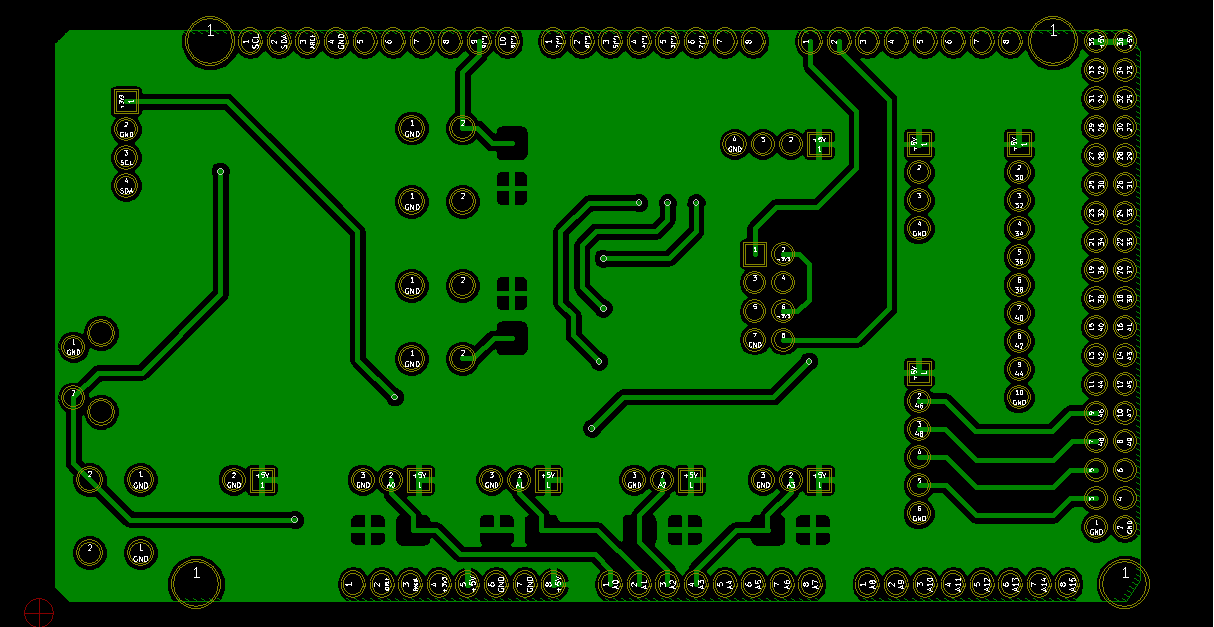
# Chapter I. Function Call Tree, Shield, Modularity, Principal Heating Schema, Memory Management, Controller communication, Libraries.

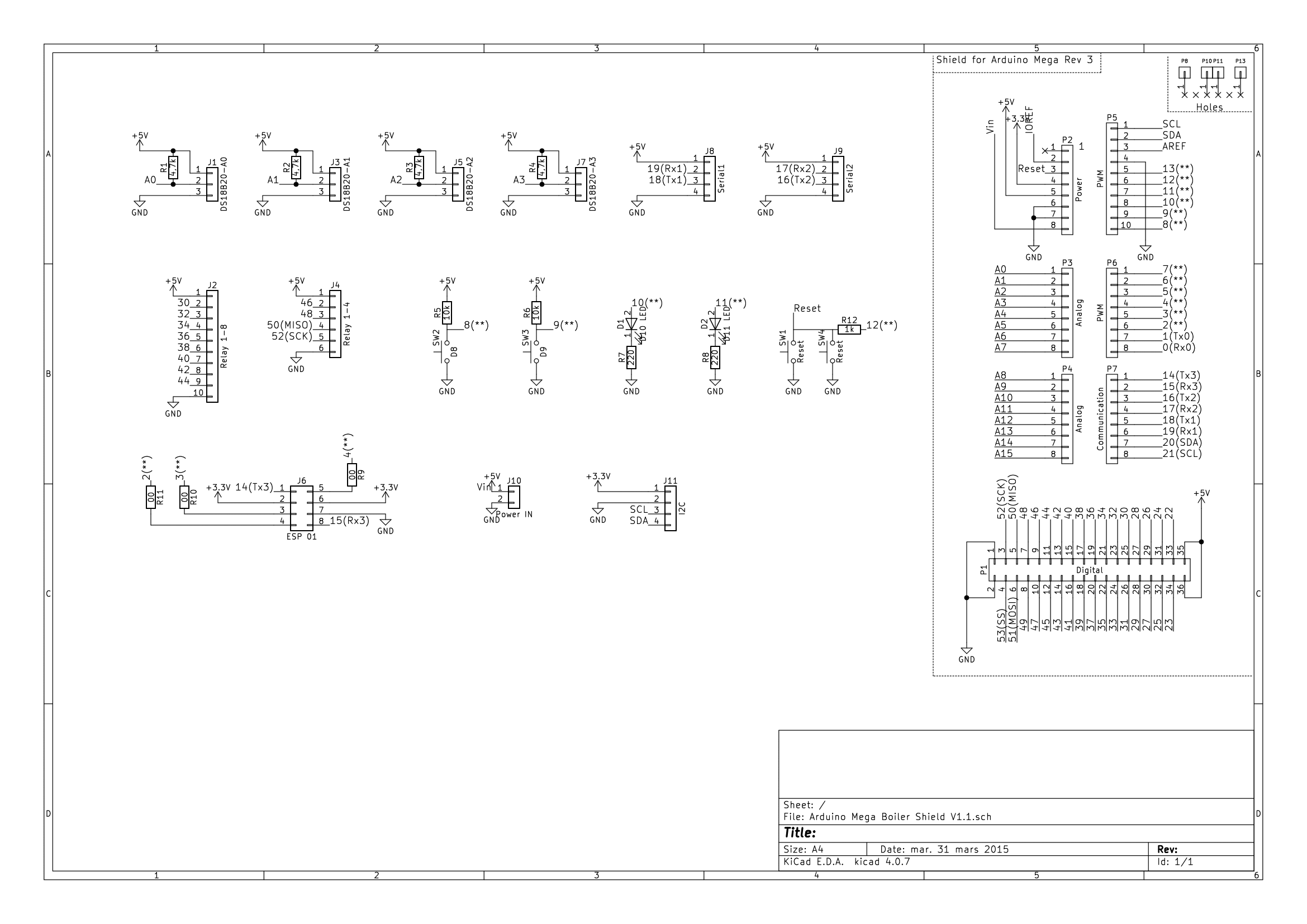
1. **Simple Logical Scheme**
2. **Principal Heating Schema**
3. **Two layer PCB Mega Shield**
   1. Up to four DS18B20 Temperature Sensors
   2. One I2C Connection
   3. Power Connector
   4. Two Serial Connectors
   5. One 4 relay block connector
   6. One 8 relay block connector
   7. Two reset buttons for chip reset
   8. Option for additional ESP8266 Chip

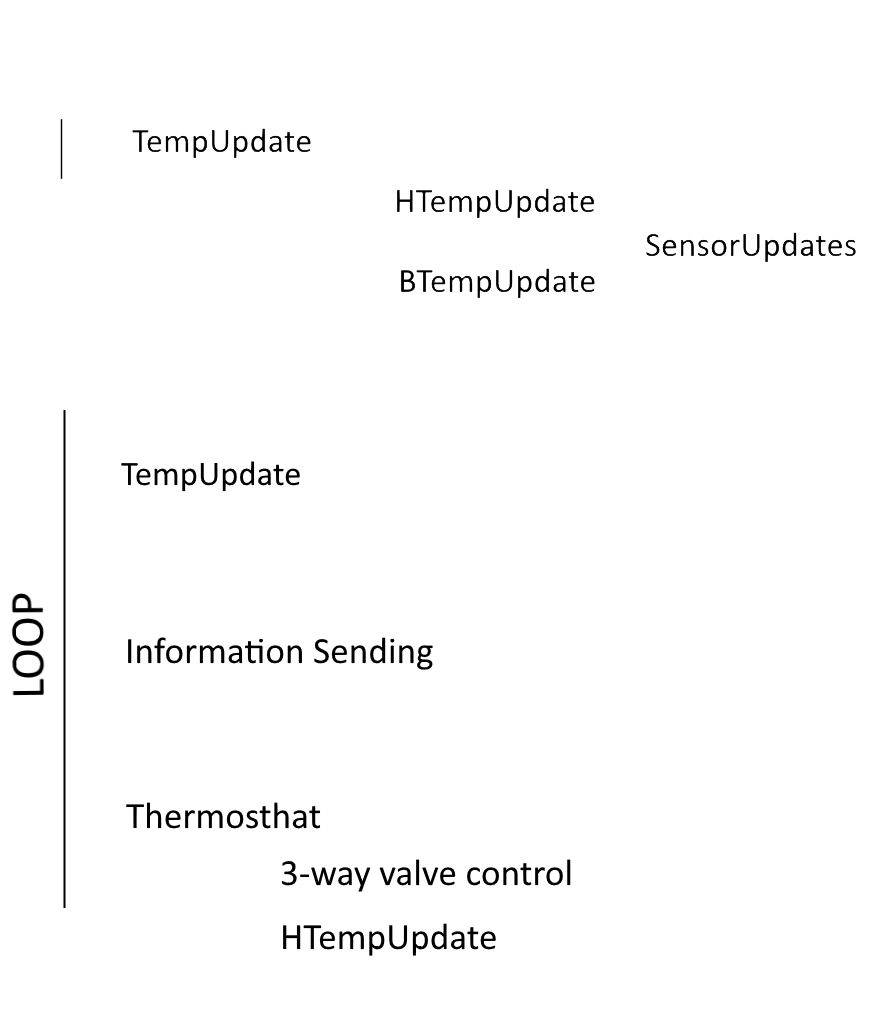


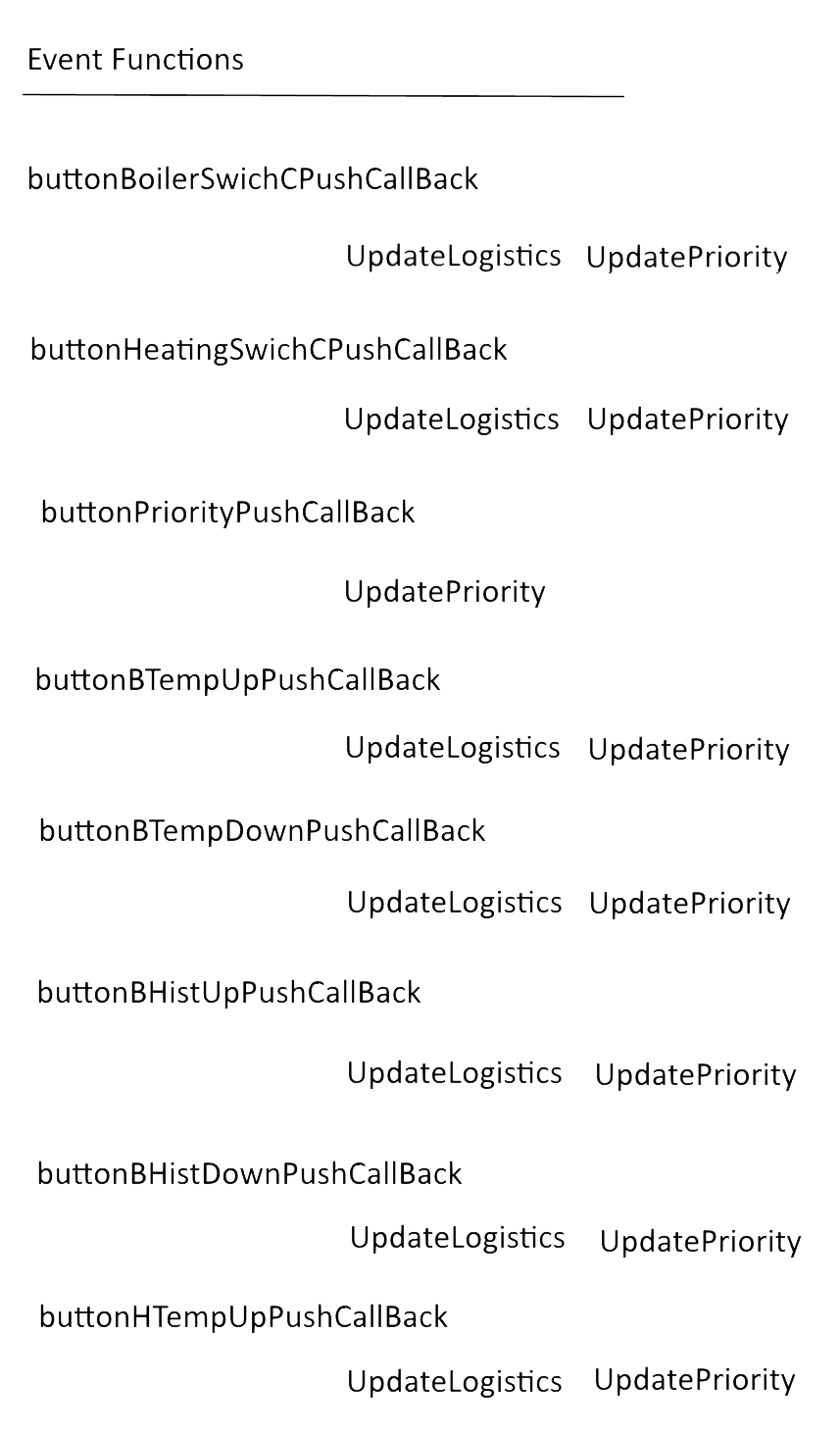


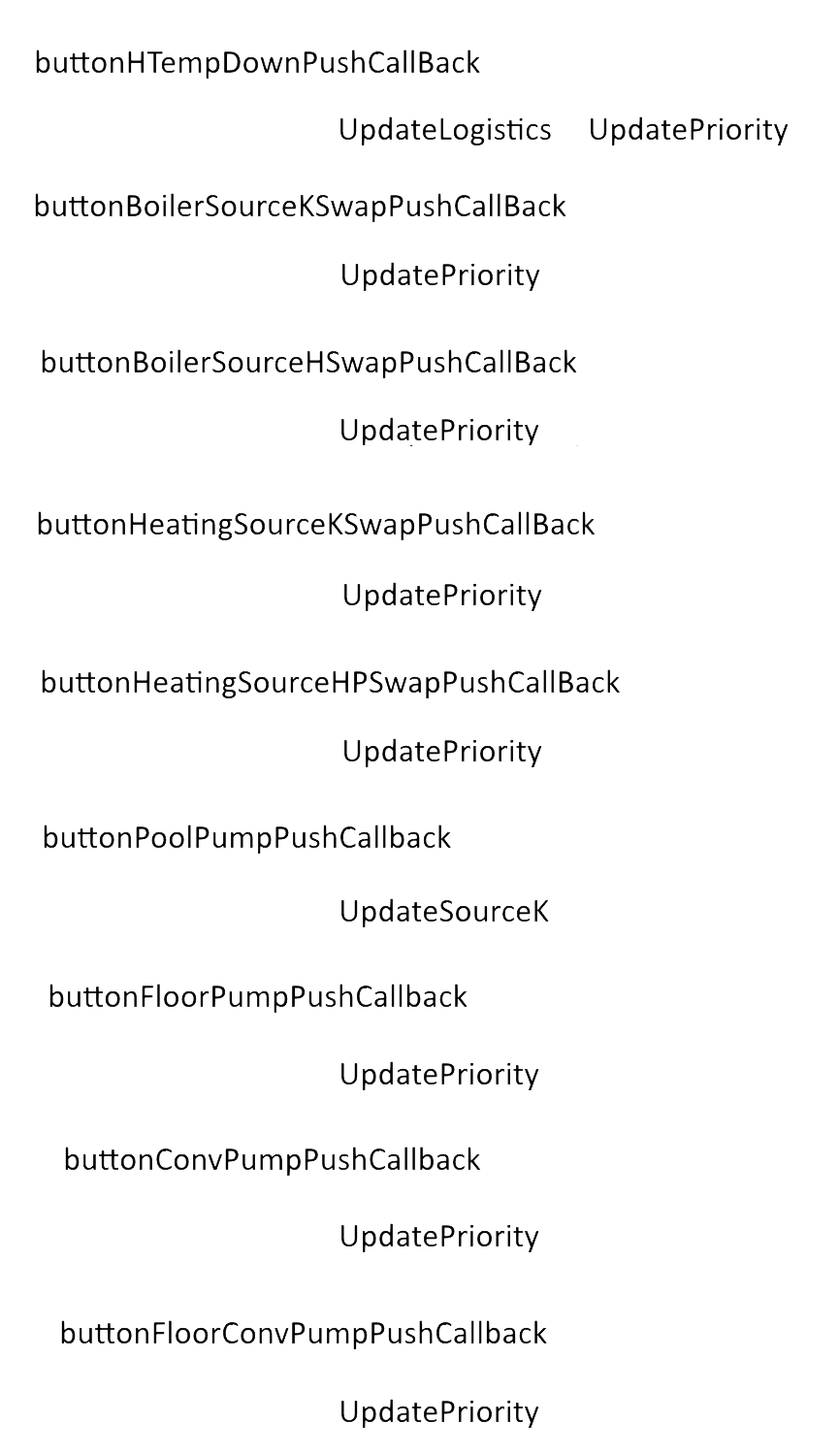
*Front Mask*

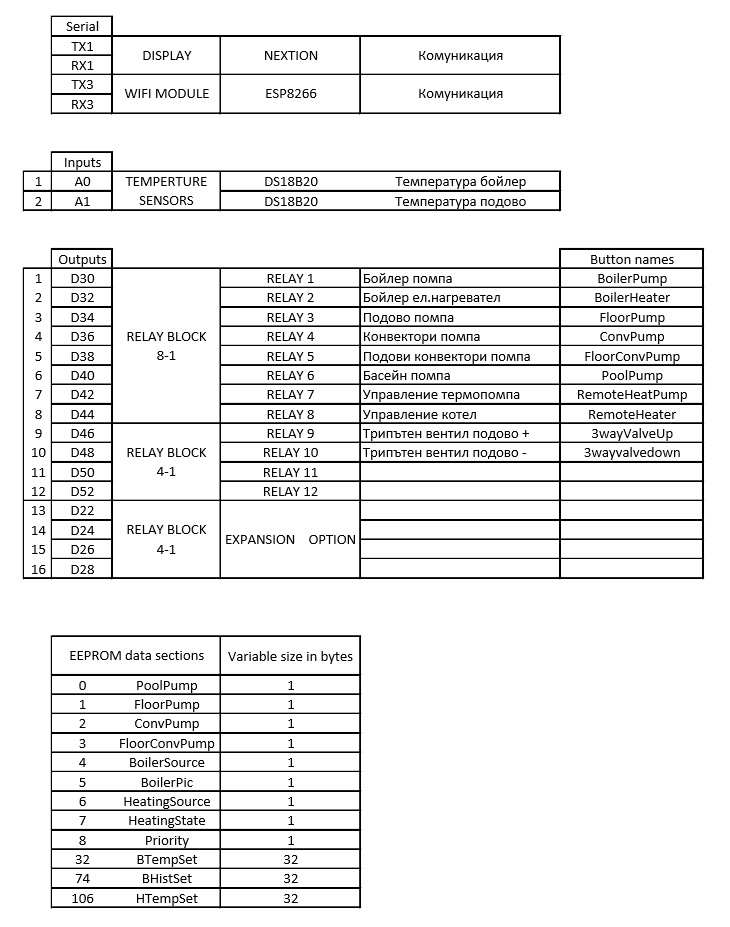
*Back Mask*

**

1. **Function Call Tree**





1. **Memory and pin layout**
2. **Controller Communication**

The Arduino Mega and the ESP module communicate through regular serial. The serialEvent interrupt function in each controller parses the messages.

The main functionality of the ESP module is to communicate with the **API** via **HTTP** **requests**. While the Mega is responsible for building the logic and managing the hardware. Both devises have acknowledgement whether or not a message has been parsed successfully.

Every 10 seconds the ESP module attempts to fetch information from the server via a **GET request** at “[server\_ulr]/users/[user\_token]”. Every device has a unique user-generated **user token** and a **controller token** which is built in the ESP module and is unique for every microcontroller. After a successful request the body of the request is fetched and sent to the Mega controller. The body contains a JSON specified by the API Documentation [here](https://github.com/DanyYanev/Boiler_Control_Server). If the parsing is successful the Mega controller will send an acknowledgement message.

The ESP won’t send data to be parsed if the current running configuration is the same as the one fetched from the server.

After a local change is made the Mega will prepare a new JSON with the changes.

Every 5 seconds if a change is made to the configuration, the updated JSON will be parsed to the ESP which will then wrap it in a **PUT request** and send it to the server. If an update is successful an acknowledgement message will be sent to the MEGA and the JSON object will be flushed. Else the MEGA will continue to send the updates every 5 seconds.

All required variables for the logic to be built are saved in EEPROM. In case of power loss, on startup all variables are loaded from the local non-volatile memory.

1. **Libraries used**
   1. [**Nextion Arduino Library**](https://github.com/DanyYanev/Boiler_Control_Server)
   2. [**Arduino JSON**](https://github.com/bblanchon/ArduinoJson)
   3. [**OneWire**](https://playground.arduino.cc/Learning/OneWire)
   4. [**EEPROM**](https://www.arduino.cc/en/Reference/EEPROM)
   5. [**ESP8266 HTTP Client**](https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266HTTPClient)
   6. [**Software Serial**](https://www.arduino.cc/en/Reference/softwareSerial)