**Analysis Report 26.9.2019-9.10-2019**

Emre Özincegedik

The project had some alterations due to Arduino Uno’s low capacity of analog pins and no storage.

The problem with storage isn’t about the data, but rather the time. Displaying the time on screen and the time of power shortage requires that the Arduino must know the time beforehand and should not be affected by it’s own power shortage. For example changing the battery would mean that Arduino would lose it’s power and the timer would be reset at that time. Arduino knows the time only through it’s embedded code. The embedded code can’t be changed after production so the timer should be independent from Arduino’s power.

Solution: New component will be added to components list whose only purpose is to keep track of time. Timer module requires it’s own power. This module will be used to display the current time and during the Arduino’s power shortage it will run on its own.

Although the timer module solved a problem, it brought another problem with it!

While project is defined the required components and their corresponding pin numbers made me conclude that Arduino Uno would be inefficient for this project. Reason is:

* Arduino Uno has 6 analog pins and 13 digital pins
* Each lamp status requires 1 analog pin (2 total)
* OLED Screen Requires 2 analog pins
* Water Sensor Requires 1 analog pin
* Electricity Sensor requires 1 analog pin
* The new Timer sensor requires 1 analog pin

These don’t count the pins that the sensors may additionally need. Only the need for analog pins.

Total analog pins needed: 7

Arduino Uno’s supporting analog pin: 6

**Why is analog pin so important while Arduino Uno has 15 other digital pins?**

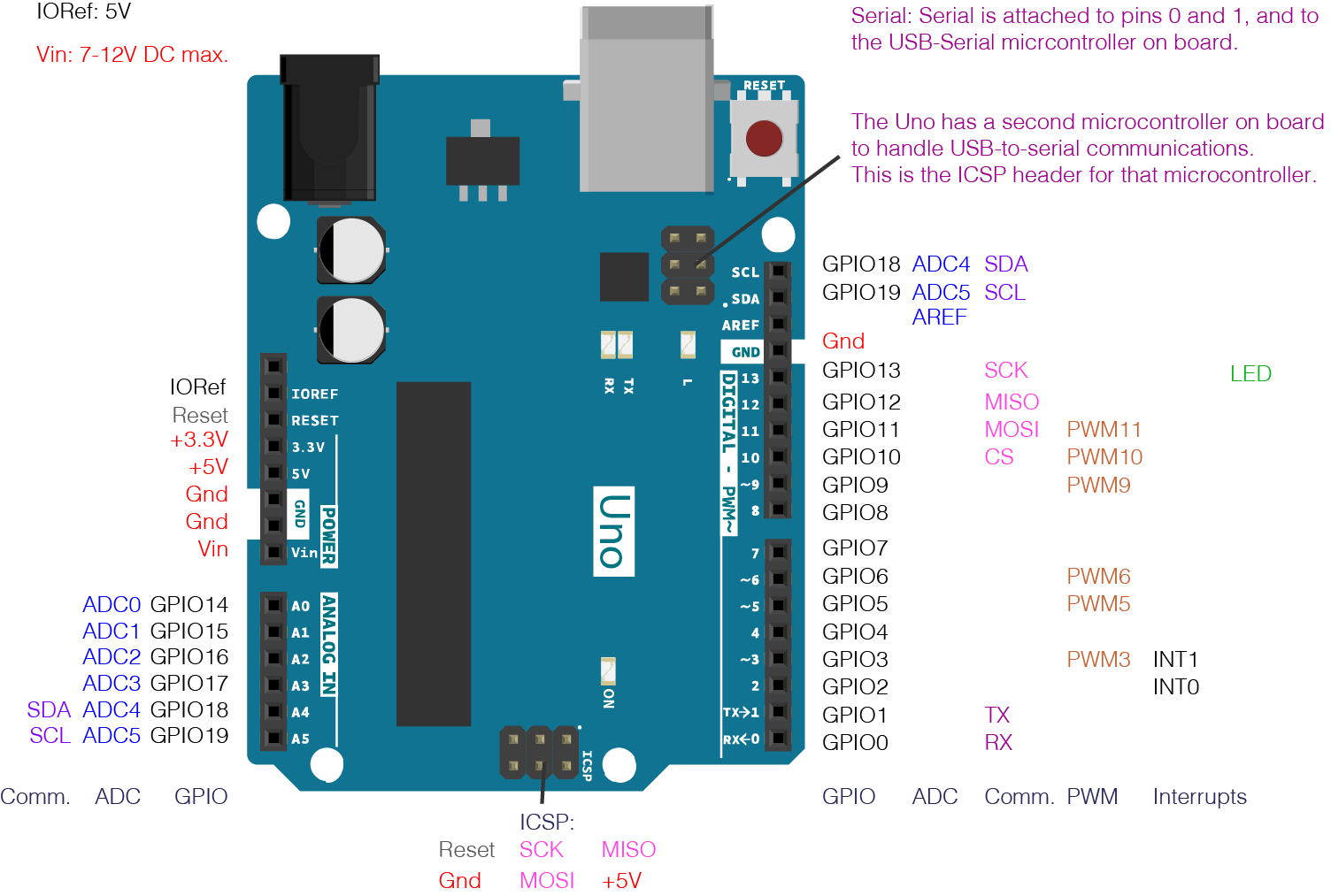
Digital pins can either be 5v or 0v for reading and writing voltage (1 or 0)

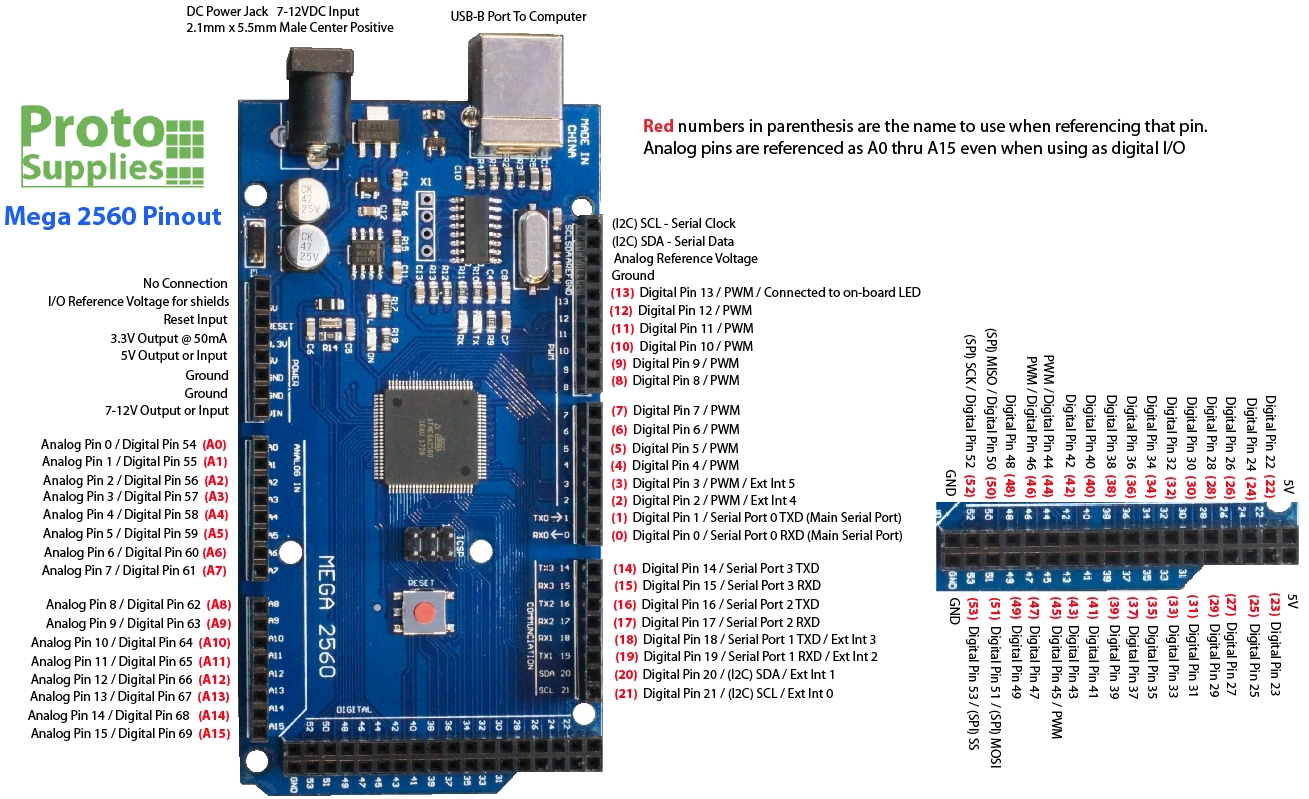
But the analog pins can read or write voltage between 0V and 5v, for example 3,2V

Some components may look like they may not need analog pins (for example lamp status) The reason they are used in analog pin instead of digital pin is that simple change in the voltage will result that the digital pin will change from 1 to 0 or vice versa. The relay will be functional after 3 Volts but Arduino can’t decide the 3V with a digital pin.

It is decided that the analog pin number must be increased to support the data flow. So the board is now Arduino Mega instead of Arduino Uno.

**Arduino Uno pin diagram:**



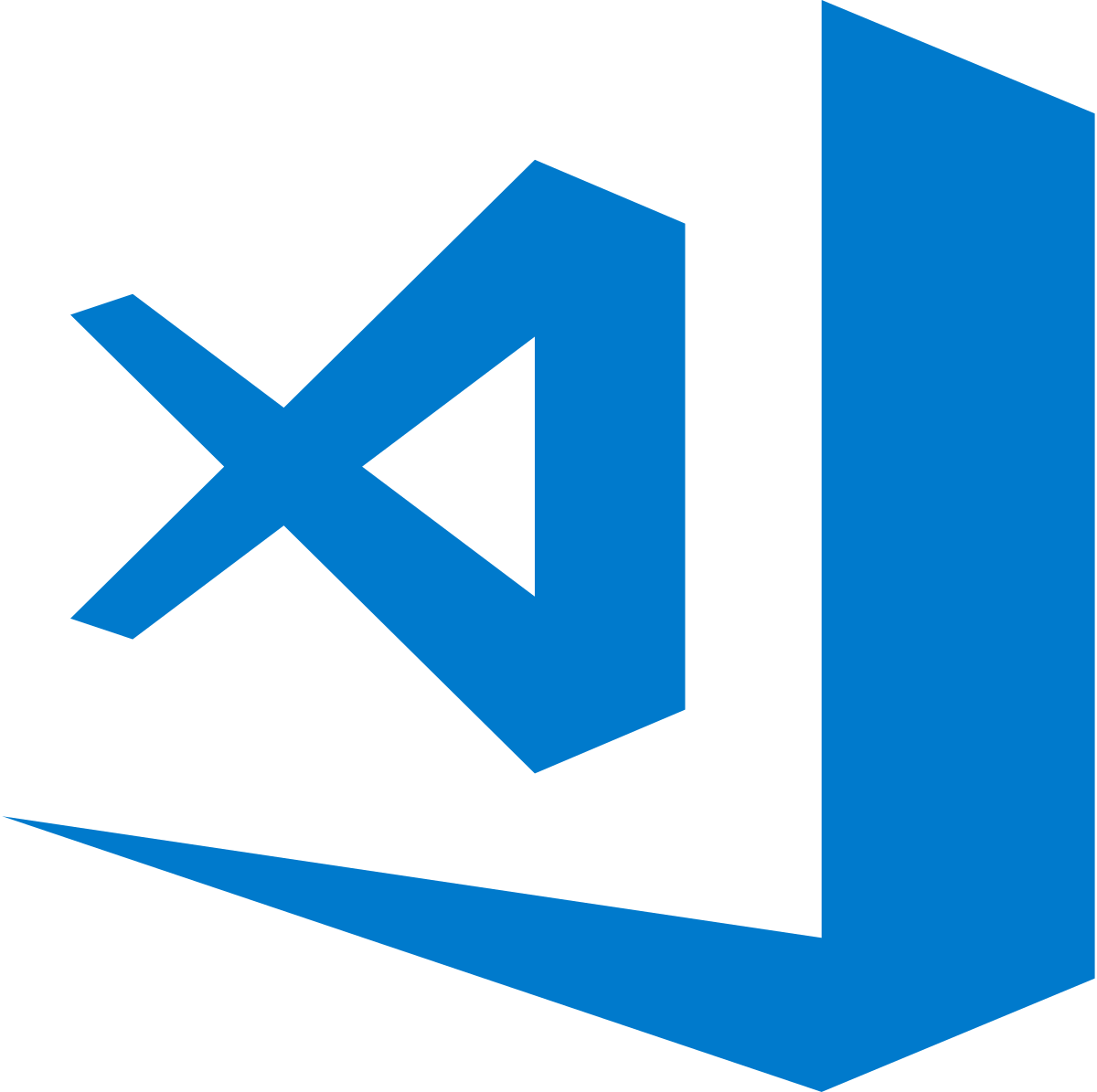
**Arduino Mega supports 16 analog pins. Arduino Mega pin diagram:**

While searching for a solution for the analog pin problem, it was decided the board would change. Among the choices for a board with more pins, another board became candidate, micro python. Micro python uses python language unlike Arduino, which uses C++. Since the board is decided to be Arduino Mega, the programming language is also decided to be C++.

**Design Report 9.10.2019-17.10.2019**

It was decided that the working platform would be Arduino Integrated Development Environment (IDE). After toying with the IDE, I concluded that it is not as efficient for writing. Why? Because Arduino IDE doesn’t have auto completion, which is a big deal breaker for slowing down the writing process a lot. Also it gives the code’s syntax errors at compilation time, not at the writing time, which will slow down the testing phase even further.

New alternative is for Arduino IDE is Visual Studio Code text editor with PlatformIO extension. This extension is specifically designed for Arduino development. Although Visual Studio Code makes writing the code significantly faster, Arduino IDE’s library manager can not be ignored. Therefore the two will be used together in the code writing phase.



Arduino IDE Visual Studio Code

Order Component Report (15.10.2019):

All the modules except time module has arrived. Timer module was added later due to alterations so it hasn’t arrived yet. The coding for that module may be rescheduled (maybe at the ending days of writing alghorithm) But it wouldn’t affect most of the project since only 2 things are depended on the time.