# Microcontroller Connections

Any microcontroller that has the ability to interface with all of the listed sensors can be used, but, at the moment (August 2016), only two test programs were pushed on this repository:

* **“Food\_Computer\_Draft\_V2”** sketch: For Arduino based boards (such as Uno, Mega, Nano, etc).
* **“FoodComputerARM-alpha”** program: For mbed-enabled ARM-based boards (tested only on Nucleo F401RE) – uses.

Feel free to use any other suitable microcontroller as needed.

Main tasks of the microcontroller are reading all sensors and controlling relays via UART.

No intelligence implemented yet (such as fail-safe, auto scheduling, RTC based timer, closed-loop control, etc). All controls are manually controlled.

Actuators were controlled by Relay modules.

Sensors Used:

* [Carbon Dioxide Sensor](http://www.co2meter.com/products/cozir-0-2-co2-sensor): COZIR Ambient 2/5/10K CO2 Sensor [UART]
* [Air Humidity and Temperature Sensor](https://www.seeedstudio.com/item_detail.html?p_id=838): Grove - Temperature&Humidity Sensor Pro **(DHT22)** [AM2302]
* [Light Intensity Sensor](https://www.seeedstudio.com/item_detail.html?p_id=1281): Grove - Digital Light Sensor **(TSL2561)** [I2C]
* [Water pH Sensor](http://www.dfrobot.com/index.php?route=product/product&product_id=1025#.V5s-2fkrKHt): Gravity: Analog pH Meter Kit [ADC]
* [Temperature compensated Water Conductivity Sensor](http://www.dfrobot.com/index.php?route=product/product&product_id=1123#.V5tCnvkrKHv): Gravity:Analog Electrical Conductivity Meter (With [Temperature Compensation](http://www.dfrobot.com/index.php?route=product/product&product_id=689#.V5tD0_krKHs)) [ADC for EC and 1-Wire for DS18B20 thermometer]

**Both analogue sensors have voltage divider (from 5V to 3.3V) hidden in cables (heat shrink tube) to avoid damaging 3.3V ARM board (Arduino runs on 5V). The resistors used were 1% tolerance and 3.3kR, 100R and 1.6kR all in series.**

Physical Pinouts:

Both Arduino and mbed uses Arduino’s pin convention, A and D stands for Analog and Digital pin respectively.

(Compatible with “Food\_Computer\_Draft\_V2” and “FoodComputerARM-alpha”)

A0 – Water pH Sensor (Analog) – 5V

A1 – Water Electrical Conductivity Sensor (Analog) – 5V

A2 – Shell Switch (Digital) – 5V or 3.3V on ARM

A3 – Window Switch (Digital) – 5V or 3.3V on ARM

D0 – Reserved for USB/UART communication (RX)

D1 – Reserved for USB/UART communication (TX)

D2 – Air Humidity and Temperature Sensor (DHT22) – 5V

D3 – Water Temperature Sensor (OneWire) – 5V

D4 – Relay0 (Active Low) – AC – Grow Light - 240V

D5 – Relay1 (Active Low) – AC – Water Pump - 240V

D6 – Relay2 (Active Low) – AC – Air Pump - 240V

D7 – Relay3 (Active Low) – AC – Air Pump - 240V

D8 – Relay4 (Active Low) – DC – Circulation Fan - 12V

D9 – Relay5 (Active Low) – DC – Ventilation Fan - 12V

D10 – Relay6 (Active Low) – DC – Strip Lights - 12V

D11 – Relay7 (Active Low) – DC – Air Humidifier - 5V

D12 – CO2 Sensor (RX) - **3.3V!**

D13 – CO2 Sensor (TX) - **3.3V!**

D14 – Light intensity sensor (SDA) – 5V

D15 – Light intensity sensor (SCL) – 5V

Connections were done via [Grove Shield v2](http://wiki.seeedstudio.com/wiki/Grove_-_Base_shield_v2).

If troubleshooting the connection via the connectors on the shield, the silkscreens on the PCB represent:

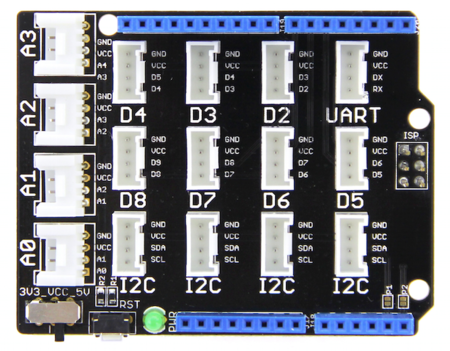


Image credit: Google image search

Vcc switch was set 5V. RST button is useful especially if raspberry pi or the Modbus library stuck.

UART – 5V power to relay module 1

D2- Air Humidity and Temperature Sensor (DHT22)

D3 - Water Temperature Sensor (OneWire)

D4 – 5V power to relay module 2

A0 – pH Sensor

A1 – EC Sensor

A2 – Windows Switch Sensor

A3 – Shell Switch Sensor

I2C (any port would do) – Light Intensity Sensor

To reset the microcontroller board via Raspberry Pi, an open-collector npn switch was connected to GPIO BCM16 (or GPIO16 or pin 36) on the Pi.

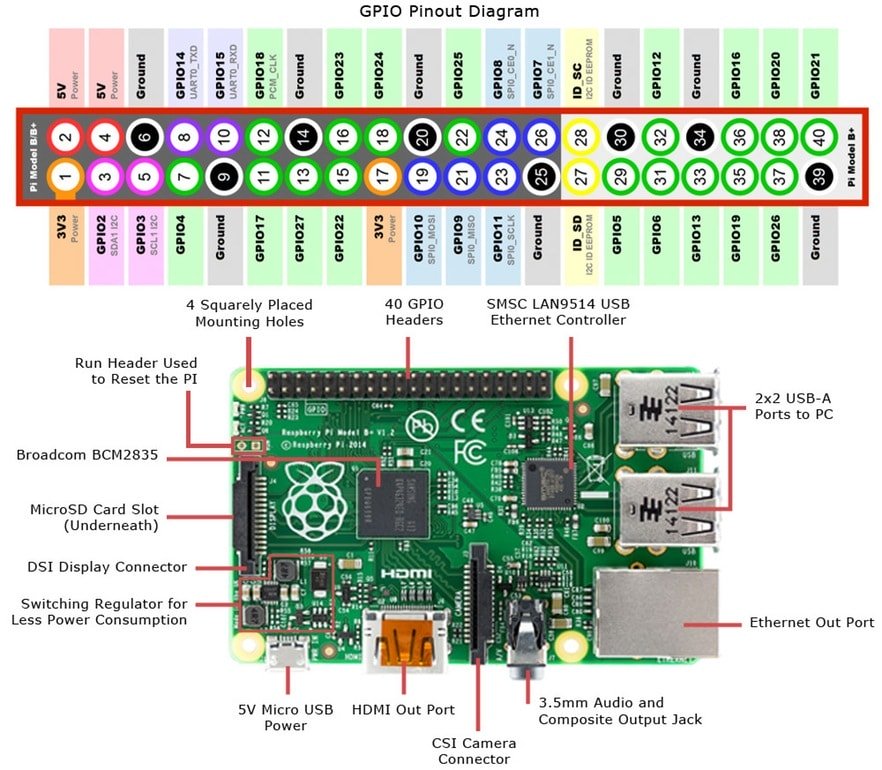


Image credit: google image search

# Developed Firmwares:

As mentioned above, there are two firmwares listed in this repo:

**Food\_Computer\_Draft\_V2**

Is a simple Arduino based “sketch”. It comes with simple Python PyQt based GUI that sends simple non-standard ascii-commands through USB/UART. Sensor readings were sent in CSV format.

The sketch is nothing but a combined libraries that read all attached sensors and send it to COM port.

**FoodComputerARM-alpha**

Developed using mbed compiler. Coded to be controlled by 9600 baud-rate MODBUS-RTU (credit: Sam). Sensor readings can be extracted by using Modbus [register read command](http://www.simplymodbus.ca/FC03.htm).

The Nucleo-401RE board is configured as Modbus slave and its address is 01.

The register used are as follows:

|  |  |  |
| --- | --- | --- |
| Holding Register Address | Stored Data | Comment |
| 01 | Carbon Dioxide Sensor (the sensor is also at 9600bps) | PPM Value |
| 02 | Air Humidity Sensor (DHT22 protocol) | In % (scaled to x100) |
| 03 | Air Temperature (via humidity sensor) | In Celcius (scaled to x100) |
| 04 | Water Temperature (OneWire) | In Celcius (scaled to x100) |
| 05 | Water Electrical Conductivity (Analogue) | In mS/cm (scaled to x100) |
| 06 | Water pH Level (Analogue) | In pH (scaled to x100) |
| 07 | Light Intensity (I2C) | Lux? |
| 08 | Window Switch | Simple True/False |
| 09 | Shell Switch (should be used for water level instead?) | Simple True/False |
| 10 | Relays (should be changed in future so that individual relays has its own register) | Use this register to control relays (first bit is for first relay, second bit is for second relay, and so on) |

The Modbus library works by periodically checks for request every 100ms. Meaning, any data transactions must be no faster than 100ms. This can be modified to run on separate thread, but, other library incompatibly issues will arise. Future revision should use a more robust Modbus library (UART interrupts?).

# Developed GUIs (PyQT-based)

* Simple UART
* Simple MQTT
* Simple Node-Red web UI (for ARM and Arduino)

(to be continued…)