HOME MONITORING SYSTEM

David Wachenschwanz

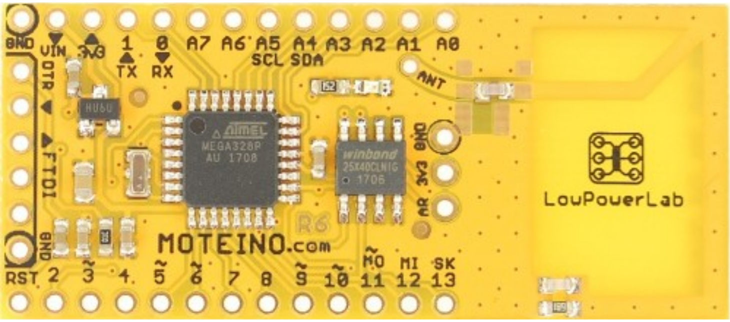
Fall 2017 UCSC IoT x413

Home Monitoring System Description

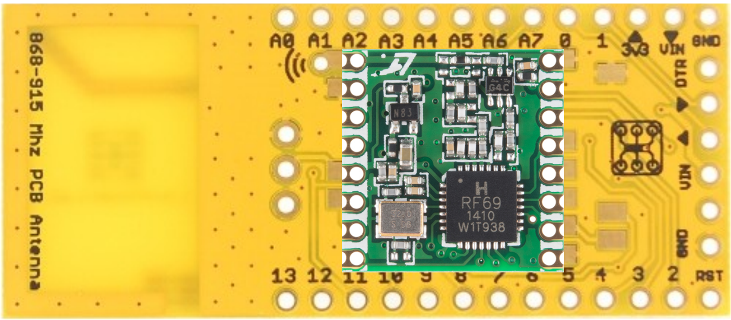
- **Multiple Inexpensive “dumb” Sensors Throughout Home Including Outdoors**
  - 9 Sensors Used in Demo
  - Each sensor contains Moteino (Atmel Atmega328 MCU), HopeRF RFM69HCW ISM Transceiver, Bosch BME280 and 18650 PCB-Protectd 3.7V Li-Ion Battery (2600 mAh)
  - Transmits Temperature, Pressure, Humidity & Battery Voltage ~ Once-per-Minute, Otherwise goes into Deep Sleep Mode to Preserve Battery Life
  - Programmed in C++ Using Arduino Development Environment
- **1st Raspberry Pi 3 Gateway**
  - HopeRF RFM69HCW Transceiver connected to SPI0 for received data from “dumb” sensors
  - **MQTT Gateway**: C program running as daemon to receive data from “dumb” sensors and then publish data by MQTT over internal network. RFM69HCW connected by SPI0 and GPIO lines for interrupts
- **2nd Raspberry Pi 3 Gateway**
  - **MQTT-to-MongoDB Service:** Python program running as a Linux service to subscribe to MQTT stream and save Temperature, Pressure, Humidity and Battery Voltage data to a MongoDB database
  - **Flask Server:** Flask server for providing overview of current sensor readings as well as for plotting data over a specific time period by location
- **Gigabyte Mini PC GB-BXi5-4570R**: Hosting MongoDB in Ubuntu 16.04 LTS Linux
- NOTE: In the future, the MQTT Gateway and MQTT-to-MongoDB Service may be combined into one service and the MQTT portion eliminated.

Moteino with RFM69HCW Transceiver & BME280 Shield

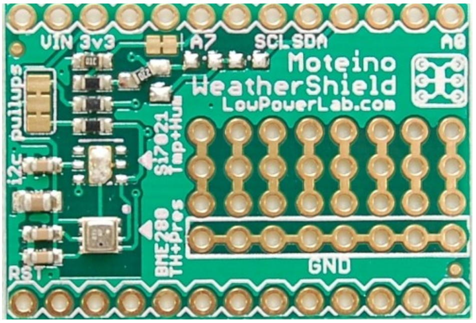
Moteino Front



Moteino Back w/RFM69HCW


868-915 Mhz PCB Antenna
A0 A1 A2 A3 A4 A5 A6 A7 0 1 3V3 VIN GND
DTR
VIN
GND
13 12 11 10 9 8 7 6 5 4 3 2 RST
RF69 1410 W1T938

BME280 Shield

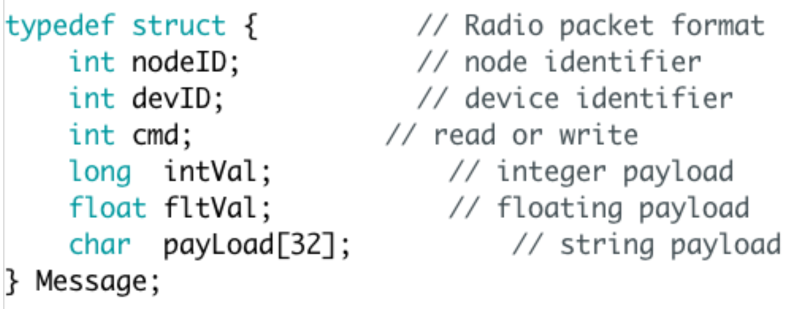


Onboard battery monitor from VIN, reading on A7, through a 2/3 (1MEG+2MEG resistor divider) and p-channel MOSFET to control powering on & off.

Supplier: https://lowpowerlab.com

Moteino Sensor Message Protocol

Structure for Data Transmission over RFM69

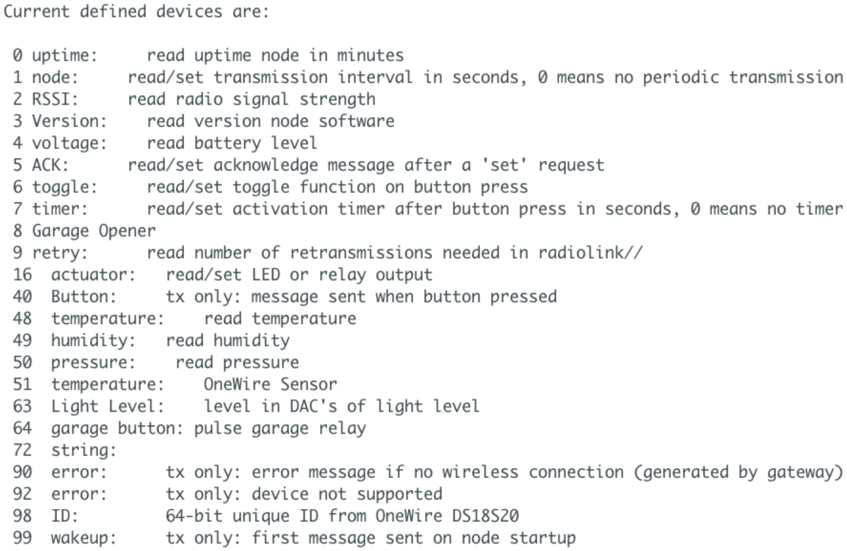


```
typedef struct {          // Radio packet format
    int nodeID;           // node identifier
    int devID;            // device identifier
    int cmd;            // read or write
    long  intVal;           // integer payload
    float fltVal;           // floating payload
    char  payLoad[32];          // string payload
} Message;
```

Note:

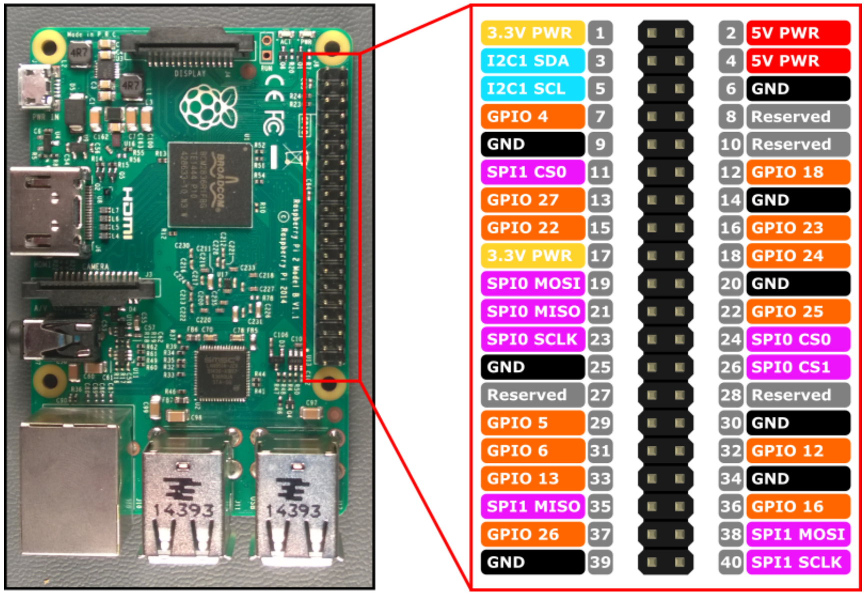
- Each Moteino Sensor has a unique nodeID (Each sensor is initialized using RFM_Initializer code)
- devID indicates type of data being transmitted
- RFM_Node is normal operational code for running sensor
- Sensors transmits data ~once per minute and then goes to deep sleep

Device Definitions



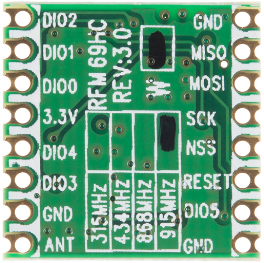
```
Current defined devices are:

 0 uptime:     read uptime node in minutes
 1 node:     read/set transmission interval in seconds, 0 means no periodic transmission
 2 RSSI:     read radio signal strength
 3 Version:    read version node software
 4 voltage:    read battery level
 5 ACK:      read/set acknowledge message after a 'set' request
 6 toggle:     read/set toggle function on button press
 7 timer:      read/set activation timer after button press in seconds, 0 means no timer
 8 Garage Opener
 9 retry:      read number of retransmissions needed in radiolink//
 16  actuator:   read/set LED or relay output
 40  Button:     tx only: message sent when button pressed
 48  temperature:    read temperature
 49  humidity:   read humidity
 50  pressure:    read pressure
 51  temperature:    OneWire Sensor
 63  Light Level:    level in DAC's of light level
 64  garage button: pulse garage relay
 72  string:
 90  error:      tx only: error message if no wireless connection (generated by gateway)
 92  error:      tx only: device not supported
 98  ID:         64-bit unique ID from OneWire DS18S20
 99  wakeup:     tx only: first message sent on node startup
```

Raspberry Pi 3 with RFM69HCW Transceiver

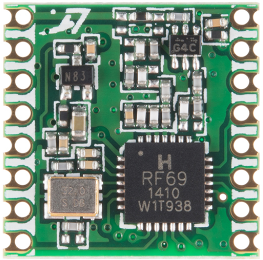


3.3V PWR	1	2	5V PWR
I2C1 SDA	3	4	5V PWR
I2C1 SCL	5	6	GND
GPIO 4	7	8	Reserved
GND	9	10	Reserved
SPI1 CS0	11	12	GPIO 18
GPIO 27	13	14	GND
GPIO 22	15	16	GPIO 23
3.3V PWR	17	18	GPIO 24
SPI0 MOSI	19	20	GND
SPI0 MISO	21	22	GPIO 25
SPI0 SCLK	23	24	SPI0 CS0
GND	25	26	SPI0 CS1
Reserved	27	28	Reserved
GPIO 5	29	30	GND
GPIO 6	31	32	GPIO 12
GPIO 13	33	34	GND
SPI1 MISO	35	36	GPIO 16
GPIO 26	37	38	SPI1 MOSI
GND	39	40	SPI1 SCLK

RFM69HCW Back



RFM69HCW Front



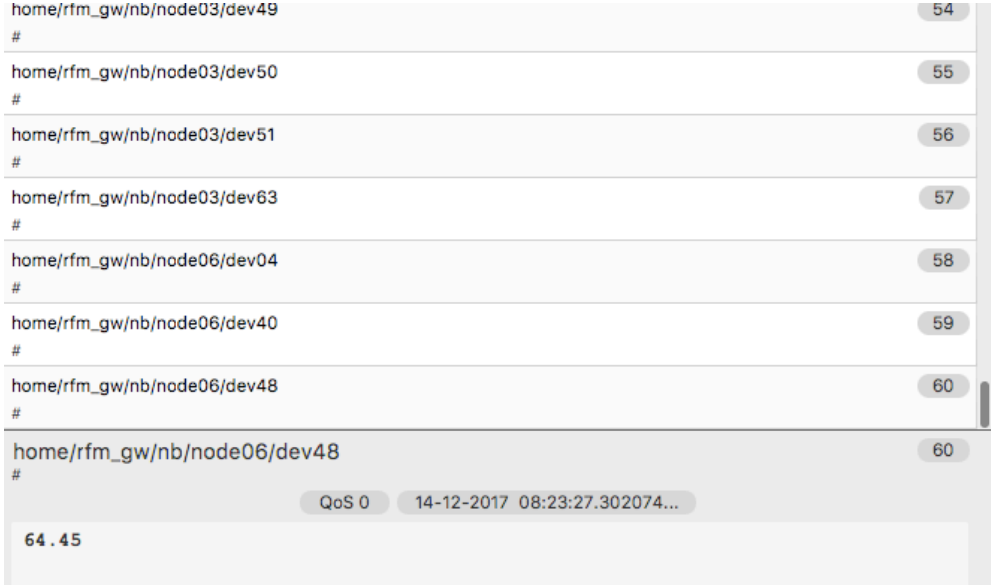
- RFM69HCW transmits at 915 MHz
- 300 kb/sec data rate
- 128-bit AES hardware encryption
- Max. 255 nodes per network
- Range: >500 meters line-of-site

Wiring Connections (RFM69 to Pi):

- 3.3V to pin 17
- GND to pin 20
- SLCK to pin 23
- MISO to pin 21
- MOSI to pin 19
- NSS to pin 24
- DID0 to pin 22

RFM-to-MQTT Gateway on Raspberry Pi 3

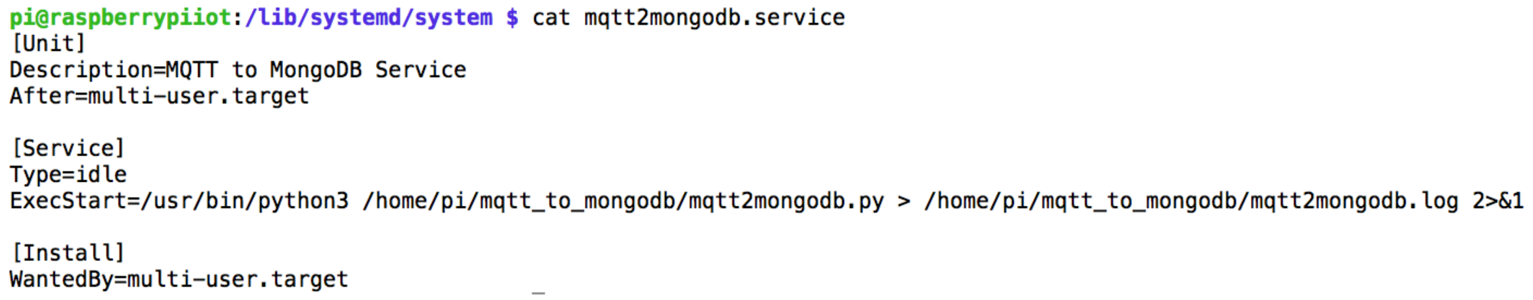
- A C program runs as a daemon on the Raspberry Pi 3 with the RFM69 transceiver connected to it through SPI and using interrupt lines
- Program interprets data structure protocol from Moteino sensor transmissions and resends data as a MQTT stream that can be subscribed to by devices in the network



```
home/rfm_gw/nb/node03/dev49                                   54
#
home/rfm_gw/nb/node03/dev50                                   55
#
home/rfm_gw/nb/node03/dev51                                   56
#
home/rfm_gw/nb/node03/dev63                                   57
#
home/rfm_gw/nb/node06/dev04                                   58
#
home/rfm_gw/nb/node06/dev40                                   59
#
home/rfm_gw/nb/node06/dev48                                   60
#
home/rfm_gw/nb/node06/dev48                                   60
#
                 QoS 0   14-12-2017 08:23:27.302074...
64.45
```

MQTT-to-mongDB Service on Raspberry Pi 3

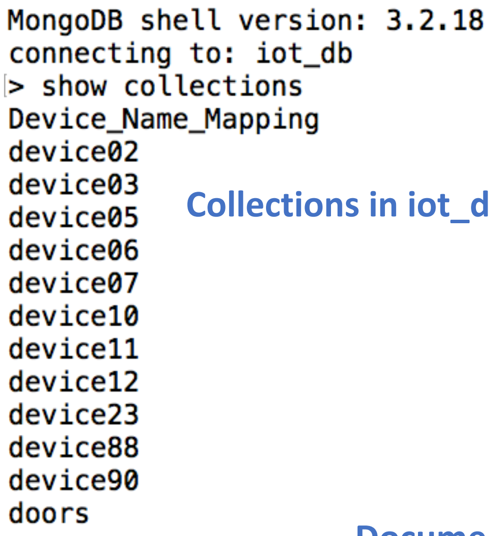
Configuration file on Raspberry Pi to run mqtt2mongodb.py as a service



```
pi@raspberrypiiot:/lib/systemd/system $ cat mqtt2mongodb.service
[Unit]
Description=MQTT to MongoDB Service
After=multi-user.target

[Service]
Type=idle
ExecStart=/usr/bin/python3 /home/pi/mqtt_to_mongodb/mqtt2mongodb.py > /home/pi/mqtt_to_mongodb/mqtt2mongodb.log 2>&1

[Install]
WantedBy=multi-user.target
```

- To enable and start service:
  - `sudo chmod 644 /lib/systemd/system/mqtt2mongodb.service`
  - `sudo systemctl daemon-reload`
  - `sudo systemctl enable mqtt2mongodb.service`
  - `sudo systemctl start mqtt2mongodb.service`
- Service will re-start at boot and run continously
- Program subscribes to MQTT stream, grabs temperature, humidity, pressure and battery voltage data and packages it together to save as a document in a mongoDB collection

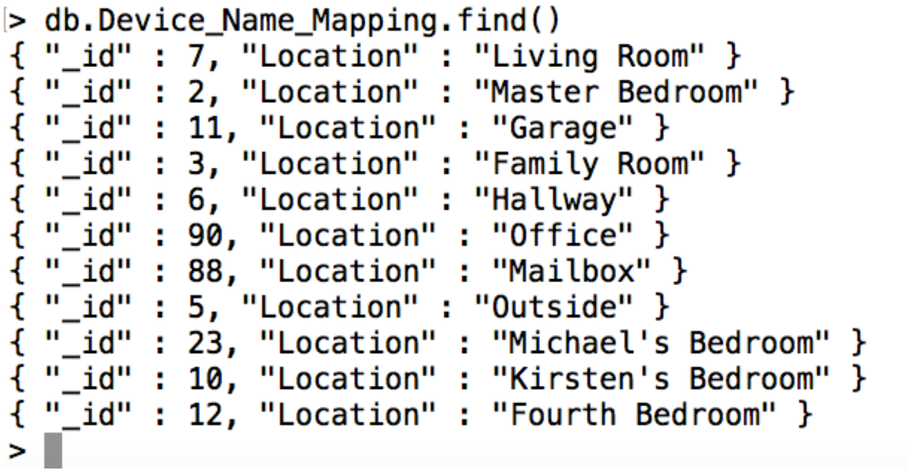
mongoDB iot_db database



```
MongoDB shell version: 3.2.18
connecting to: iot_db
> show collections
Device_Name_Mapping
device02
device03
device05
device06
device07
device10
device11
device12
device23
device88
device90
doors
```

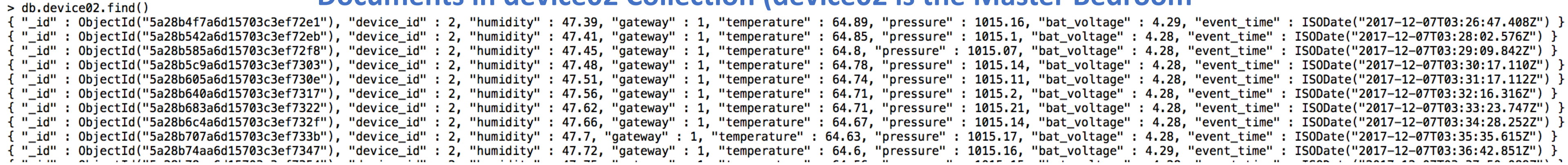
Collections in iot_db

Documents in Device_Name_Mapping Collection



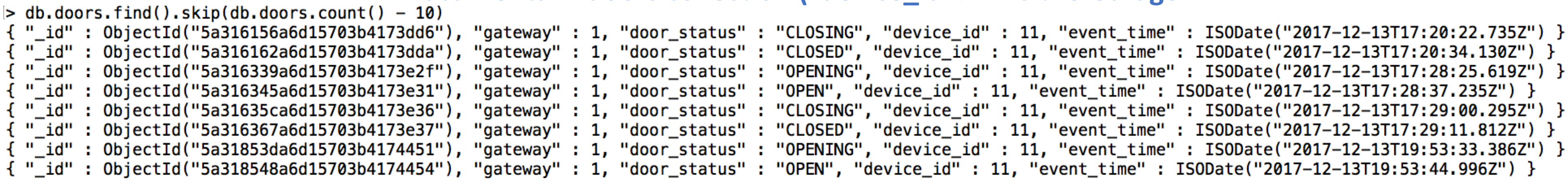
```
> db.Device_Name_Mapping.find()
{ "_id" : 7, "Location" : "Living Room" }
{ "_id" : 2, "Location" : "Master Bedroom" }
{ "_id" : 11, "Location" : "Garage" }
{ "_id" : 3, "Location" : "Family Room" }
{ "_id" : 6, "Location" : "Hallway" }
{ "_id" : 90, "Location" : "Office" }
{ "_id" : 88, "Location" : "Mailbox" }
{ "_id" : 5, "Location" : "Outside" }
{ "_id" : 23, "Location" : "Michael's Bedroom" }
{ "_id" : 10, "Location" : "Kirsten's Bedroom" }
{ "_id" : 12, "Location" : "Fourth Bedroom" }
>
```

Documents in device02 Collection (device02 is the Master Bedroom



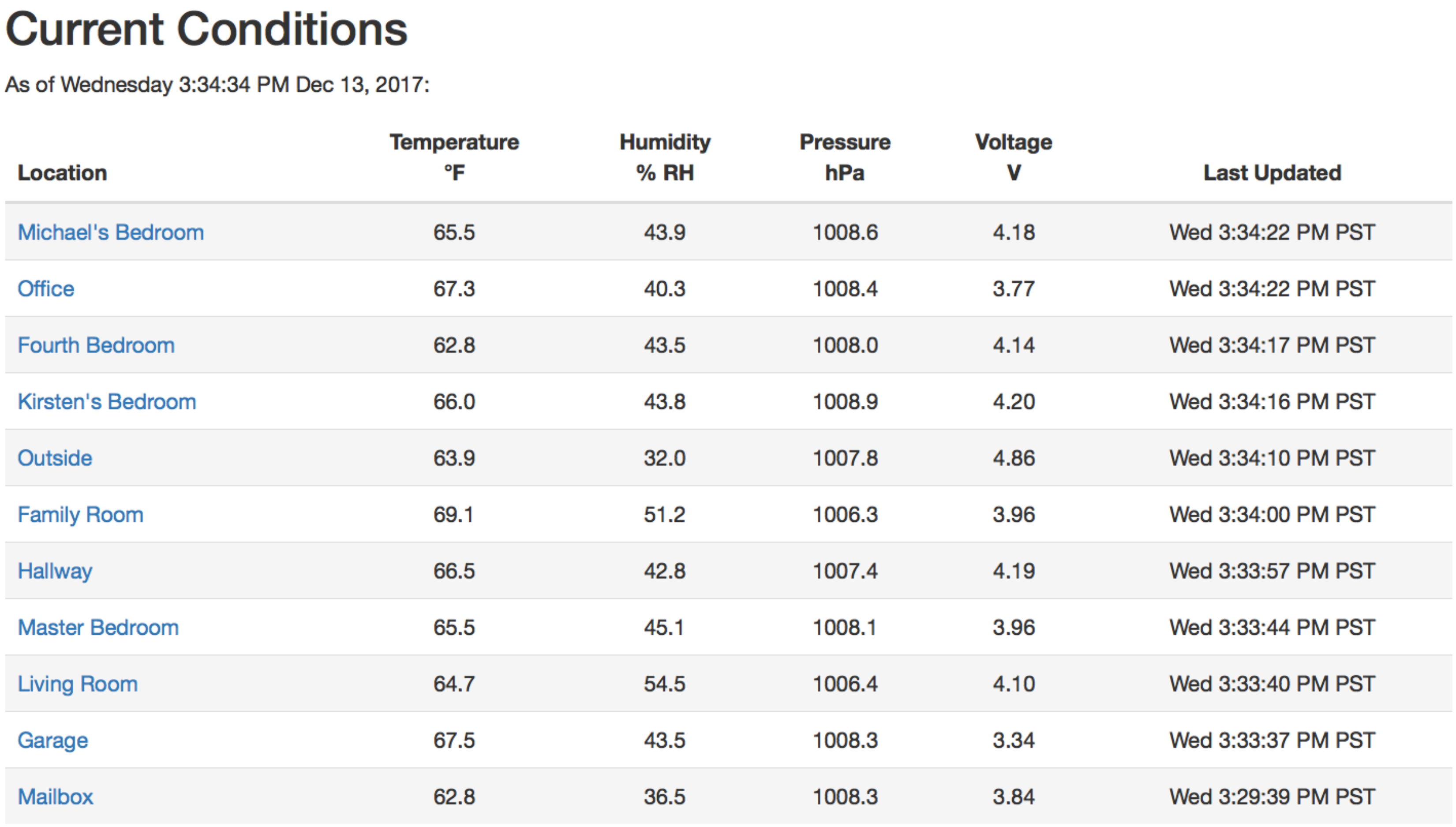
```
> db.device02.find()
{ "_id" : ObjectId("5a28b4f7a6d15703c3ef72e1"), "device_id" : 2, "humidity" : 47.39, "gateway" : 1, "temperature" : 64.89, "pressure" : 1015.16, "bat_voltage" : 4.29, "event_time" : ISODate("2017-12-07T03:26:47.408Z") }
{ "_id" : ObjectId("5a28b542a6d15703c3ef72eb"), "device_id" : 2, "humidity" : 47.41, "gateway" : 1, "temperature" : 64.85, "pressure" : 1015.1, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:28:02.576Z") }
{ "_id" : ObjectId("5a28b585a6d15703c3ef72f8"), "device_id" : 2, "humidity" : 47.45, "gateway" : 1, "temperature" : 64.8, "pressure" : 1015.07, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:29:09.842Z") }
{ "_id" : ObjectId("5a28b5c9a6d15703c3ef7303"), "device_id" : 2, "humidity" : 47.48, "gateway" : 1, "temperature" : 64.78, "pressure" : 1015.14, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:30:17.110Z") }
{ "_id" : ObjectId("5a28b605a6d15703c3ef730e"), "device_id" : 2, "humidity" : 47.51, "gateway" : 1, "temperature" : 64.74, "pressure" : 1015.11, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:31:17.112Z") }
{ "_id" : ObjectId("5a28b640a6d15703c3ef7317"), "device_id" : 2, "humidity" : 47.56, "gateway" : 1, "temperature" : 64.71, "pressure" : 1015.2, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:32:16.316Z") }
{ "_id" : ObjectId("5a28b683a6d15703c3ef7322"), "device_id" : 2, "humidity" : 47.62, "gateway" : 1, "temperature" : 64.71, "pressure" : 1015.21, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:33:23.747Z") }
{ "_id" : ObjectId("5a28b6c4a6d15703c3ef732f"), "device_id" : 2, "humidity" : 47.66, "gateway" : 1, "temperature" : 64.67, "pressure" : 1015.14, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:34:28.252Z") }
{ "_id" : ObjectId("5a28b707a6d15703c3ef733b"), "device_id" : 2, "humidity" : 47.7, "gateway" : 1, "temperature" : 64.63, "pressure" : 1015.17, "bat_voltage" : 4.28, "event_time" : ISODate("2017-12-07T03:35:35.615Z") }
{ "_id" : ObjectId("5a28b74aa6d15703c3ef7347"), "device_id" : 2, "humidity" : 47.72, "gateway" : 1, "temperature" : 64.6, "pressure" : 1015.16, "bat_voltage" : 4.29, "event_time" : ISODate("2017-12-07T03:36:42.851Z") }
```

Documents in doors collection (“device_id”: 11 is the Garage



```
> db.doors.find().skip(db.doors.count() - 10)
{ "_id" : ObjectId("5a316156a6d15703b4173dd6"), "gateway" : 1, "door_status" : "CLOSING", "device_id" : 11, "event_time" : ISODate("2017-12-13T17:20:22.735Z") }
{ "_id" : ObjectId("5a316162a6d15703b4173dda"), "gateway" : 1, "door_status" : "CLOSED", "device_id" : 11, "event_time" : ISODate("2017-12-13T17:20:34.130Z") }
{ "_id" : ObjectId("5a316339a6d15703b4173e2f"), "gateway" : 1, "door_status" : "OPENING", "device_id" : 11, "event_time" : ISODate("2017-12-13T17:28:25.619Z") }
{ "_id" : ObjectId("5a316345a6d15703b4173e31"), "gateway" : 1, "door_status" : "OPEN", "device_id" : 11, "event_time" : ISODate("2017-12-13T17:28:37.235Z") }
{ "_id" : ObjectId("5a31635ca6d15703b4173e36"), "gateway" : 1, "door_status" : "CLOSING", "device_id" : 11, "event_time" : ISODate("2017-12-13T17:29:00.295Z") }
{ "_id" : ObjectId("5a316367a6d15703b4173e37"), "gateway" : 1, "door_status" : "CLOSED", "device_id" : 11, "event_time" : ISODate("2017-12-13T17:29:11.812Z") }
{ "_id" : ObjectId("5a31853da6d15703b4174451"), "gateway" : 1, "door_status" : "OPENING", "device_id" : 11, "event_time" : ISODate("2017-12-13T19:53:33.386Z") }
{ "_id" : ObjectId("5a318548a6d15703b4174454"), "gateway" : 1, "door_status" : "OPEN", "device_id" : 11, "event_time" : ISODate("2017-12-13T19:53:44.996Z") }
```

Index Page for Home Monitoring Website



Current Conditions

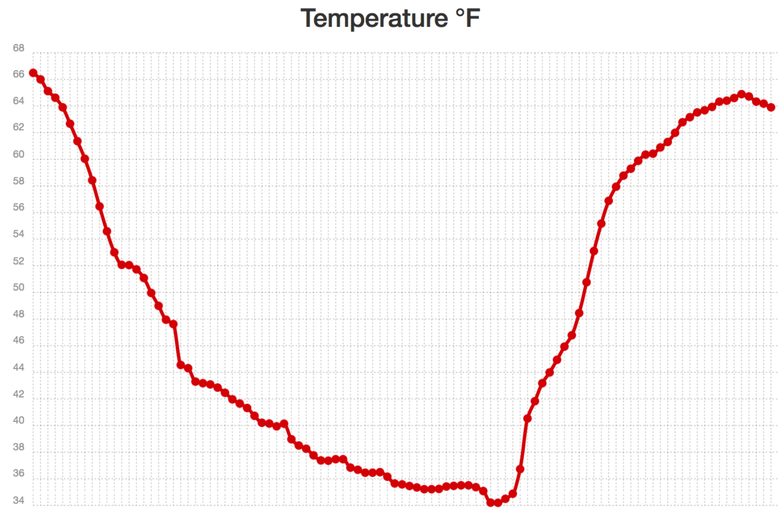
As of Wednesday 3:34:34 PM Dec 13, 2017:

Location	Temperature °F	Humidity % RH	Pressure hPa	Voltage V	Last Updated
Michael's Bedroom	65.5	43.9	1008.6	4.18	Wed 3:34:22 PM PST
Office	67.3	40.3	1008.4	3.77	Wed 3:34:22 PM PST
Fourth Bedroom	62.8	43.5	1008.0	4.14	Wed 3:34:17 PM PST
Kirsten's Bedroom	66.0	43.8	1008.9	4.20	Wed 3:34:16 PM PST
Outside	63.9	32.0	1007.8	4.86	Wed 3:34:10 PM PST
Family Room	69.1	51.2	1006.3	3.96	Wed 3:34:00 PM PST
Hallway	66.5	42.8	1007.4	4.19	Wed 3:33:57 PM PST
Master Bedroom	65.5	45.1	1008.1	3.96	Wed 3:33:44 PM PST
Living Room	64.7	54.5	1006.4	4.10	Wed 3:33:40 PM PST
Garage	67.5	43.5	1008.3	3.34	Wed 3:33:37 PM PST
Mailbox	62.8	36.5	1008.3	3.84	Wed 3:29:39 PM PST

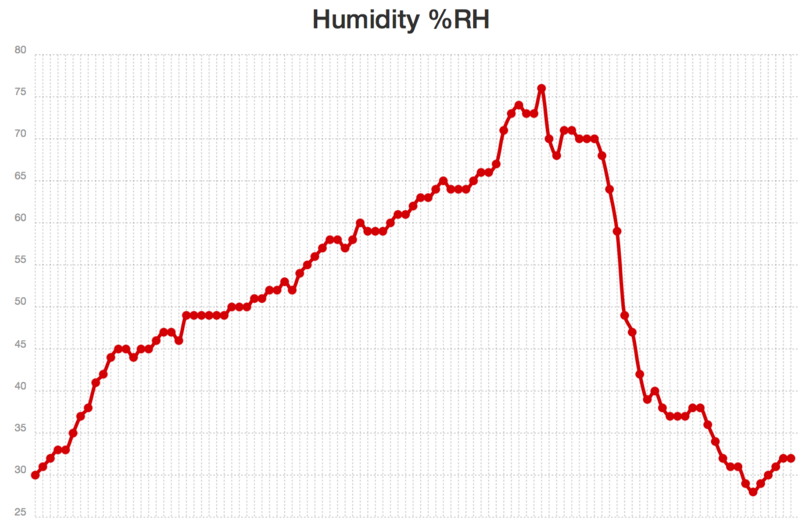
Specific Location Conditions
(Reached by clicking on Location)

Outside Conditions

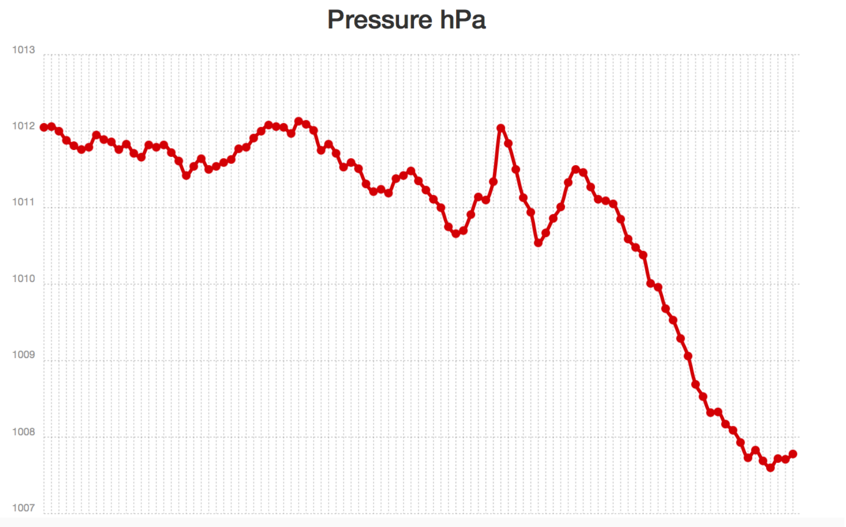
over the past 24 hours as of Wednesday 3:38:05 PM Dec 13, 2017:



Temperature °F



Humidity %RH



Pressure hPa

Security Enhancements

- Need to add login system onto flask website server
  - Username and hashed password will be stored on mongoDB database
- Change website from http: to https:
- Add encryption to MQTT (currently is password protected but sent in clear text)
  - Other option is to eliminate MQTT and just have a RFM-to-mongoDB service running, skipping the MQTT all together

Updates

- Updates to Moteino Sensors can be done over-the-air (OTA) although not currently implemented. Also there is an issue trying to wake them up from deep sleep to do the OTA update that has to be addressed
- Firmware running on Raspberry Pi gateway could occasionally call to a centralized server to check for any software updates for either the gateway or the Moteino Sensors and then push the updates accordingly
- Lots of work still needed to address details

Final Comments

- Enhancements needed to the Flask web server:
  - Provide login security
  - Add ability to get automatic update of garage door opening and closing while website is being viewed,
  - Add ability to get summary statistics of conditions in each room
  - Add electronics to control HVAC system (furnace) instead of a normal thermostat

- Code for project is available on my github:

  - https://github.com/wachenda/IOT_Sensors_Platforms_Communications-Class.git