

Reduction of Post-Harvest Crop Loss in Developing Countries using Sensor Network

Luis Ramirez

Haowen Zhang

August 10, 2016

Abstract

This project is focused on creating a hub system that allows data collection from multiple sensors which can detect temperature and humidity. The sensor hub system will include a scheduling system that will allow the hub to log the data from the sensors at a set interval. The hub system will also allow for text notifications to be sent out to farmers when the sensors detect unstable conditions for storage.

Contents

1. Introduction.....	3
1.1 Purpose.....	3
1.2 Objectives.....	3
2. Hardware Design.....	3
2.1 Design overview.....	3
2.1.1 Block Diagram.....	4
2.1.2 Circuit Schematic.....	5
2.1.3 PCB Layout.....	6
2.2 Microcontrollers (Atmega328P).....	7
2.2.1 MCU-A.....	7
2.2.2 MCU-B.....	7
2.2.3 Inter-MCU Communication.....	8
2.3 Real-Time Clock (DS3231).....	8
2.3.1 Time and Date Format.....	8
2.3.2 Alarms.....	8
2.4 GSM Module (SIM800L).....	8
2.4.1 Voltage and current requirement.....	9
2.4.2 AT commands.....	9
2.5 SD Card Module.....	10
2.5.1 Files.....	10
2.5.2 Data Storage.....	10
2.6 Bluetooth Module (HM-10)	10
2.6.1 Voltage and Current Requirement.....	10
2.6.2 Bluetooth Low Energy.....	10
3. Software Design.....	11
3.1 Software Flowchart.....	11
3.2 Sensor Data Structure.....	12
4. Costs.....	13
5. Troubleshooting.....	14
6. Conclusion.....	16
6.1 Future Developments.....	16
6.2 Future Improvements.....	16
7. References.....	17
Appendix A: Mobile Application Commands.....	18
Appendix B: Inter-MCU Commands.....	19

1. Introduction

1.1 Purpose

Every year there is a large post-harvest crop loss that are caused by various reasons such as bad storage conditions which enable mold growth. To curb this issue, a network of sensors would allow the monitoring of post-harvest crops and allow farmers to be alerted when their crops are in ideal conditions for mold to grow, effectively ruining the crop. By collecting the sensor data, researchers can also monitor the crops and help farmers reduce their losses.

1.2 Objectives

The object of this project is to

- Provide a sensor hub systems that can record data from multiple sensors.
- Alert farmers of potential crop spoilage.
- Send data back to researchers to further study how to prevent post-harvest crop loss.

2. Hardware Design

2.1 Design Overview

The sensor hub can be divided into two parts functionally: the communication part and management part. Each part is controlled by a Atmega328P microcontroller individually and able to communicate with each other via softserial communication.

The communication part consists of a microcontroller(MCU A), a Bluetooth Low Energy(BLE) module and a GSM module, providing the communication between:

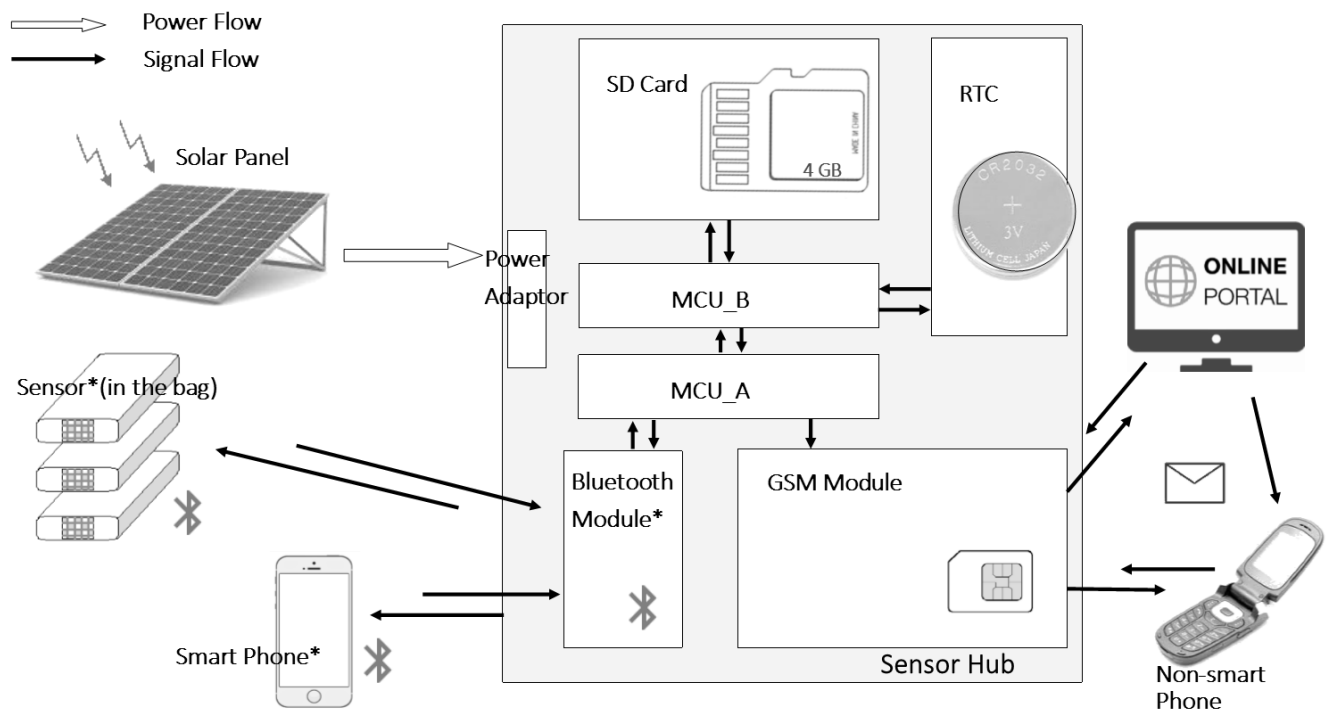
- Hub and sensors via BLE(data collection)
- Hub and smartphones via BLE(configuration setup)
- Hub and farmer's cell phone via GSM(alert messaging)
- Hub and online portal via GSM(data reporting)

The management part consists of a microcontroller(MCU B), a SD module and a Real Time Clock(RTC) module, providing the function below:

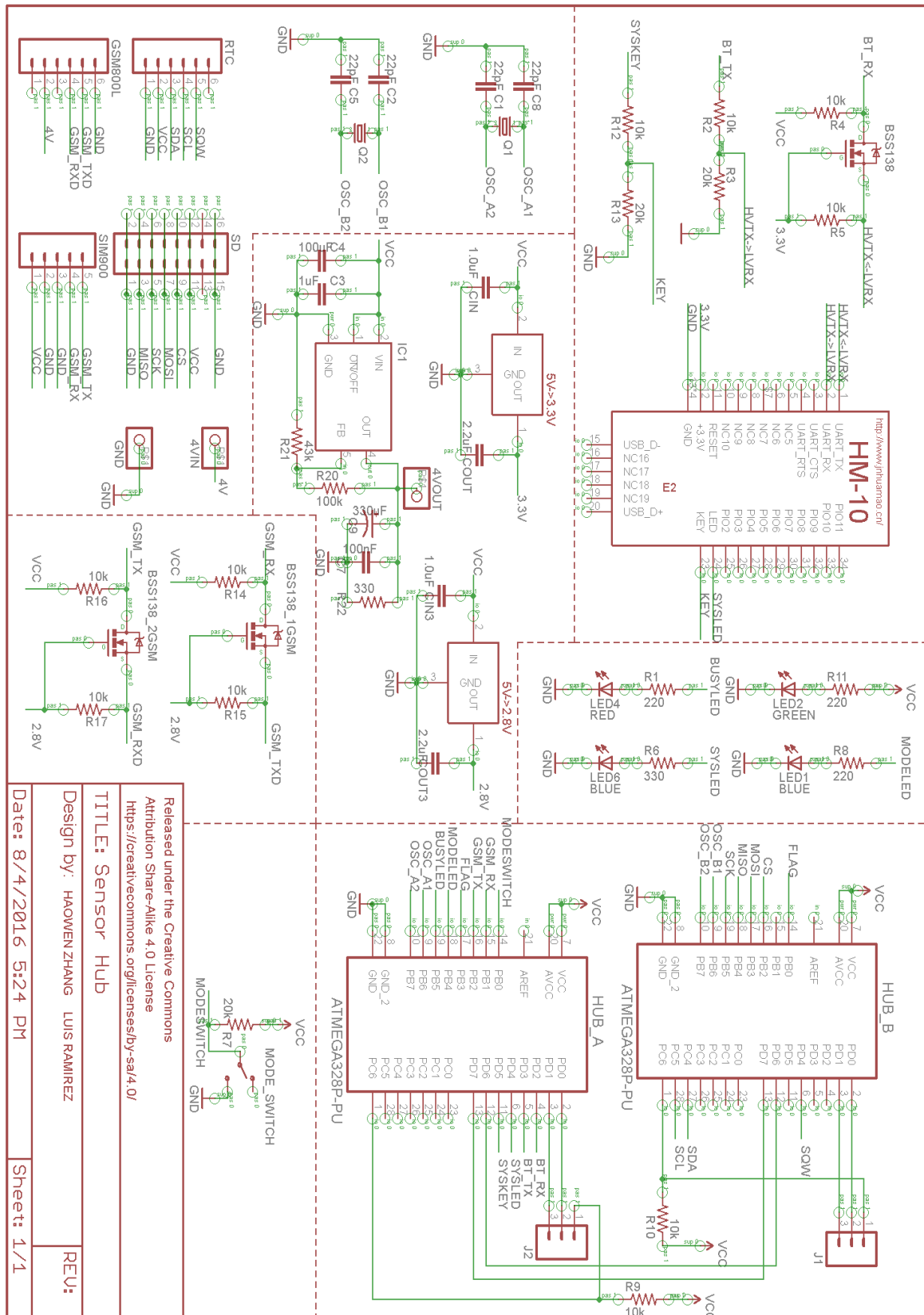
- Store configuration into local SD card
- Store and organize sensor data in comma-separated values(.CSV) files
- Alarm for sensor checking and data reporting
- Read and search data in SD card

The selection of hardware components is based on two principles: cheap and reliable.

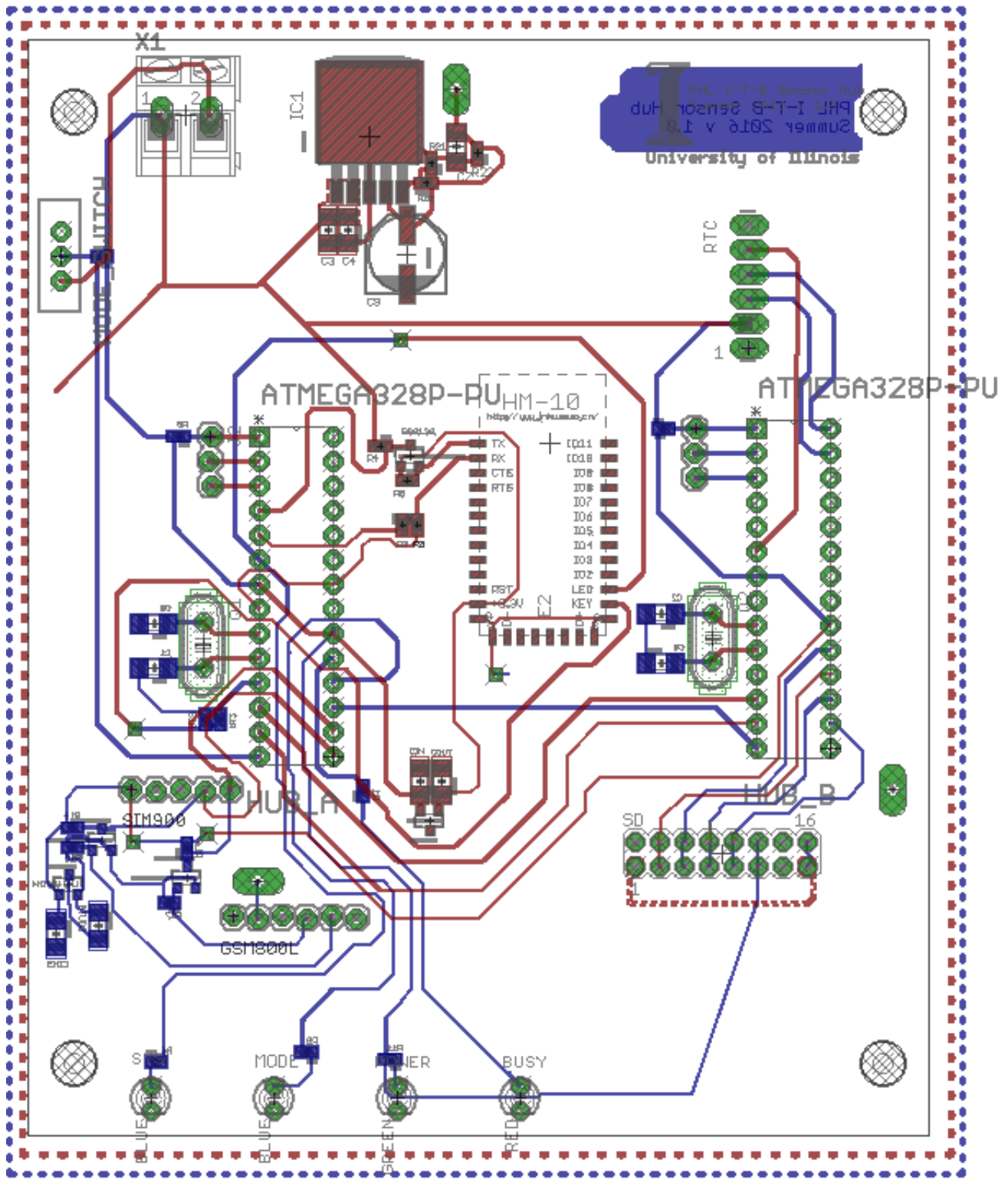
2.1.1 Block Diagram



2.1.2 Circuit Schematic



2.1.3 PCB Layout



2.2 Microcontrollers (Atmega328P)

The microcontrollers of the hub system are two Atmega328P chips. Two chips are used due to memory restrictions of the Atmega328P when using all modules. Each microcontroller handles a separate set of modules to offload the memory usage for each module. MCU-A handles the Bluetooth communication and the GSM Text Notifications while MCU-B handles reading and writing to the SD card along with keeping track of time using the real-time clock. Both microcontrollers communicate using serial communication to send commands and data to each other where MCU-A sends a command to MCU-B when necessary.

2.2.1 MCU-A

MCU-A is dedicated to handling the Bluetooth communication and the GSM module. MCU-A also acts as a master to MCU-B by requesting data from the SD card for the initial setup phase as well as when the hub is connected to the phone app. When the hub is set into configuration mode, it will wait for the phone app to connect to it and send commands and send any additional commands to MCU-B to store the updated configurations in the SD card. While in configuration mode, the hub will also ignore the alarm from the RTC for the most part, but it will still be handled by MCU-B which will set the Flag pin HIGH so that MCU-A can handle checking the sensors once it finishes with its configurations from the phone app.

2.2.2 MCU-B

MCU-B is dedicated to handling the RTC and the SD card functionality and acts as a slave to MCU-A. When the Atmega328P pins are initialized to HIGH through the bootloader, the setup of MCU-B waits until the end to set the Flag pin LOW so that MCU-A knows that MCU-B is done with its setup. This is to ensure that MCU-B is ready to start processing the initial setup commands of MCU-A , which requires multiple accesses to the SD card in order to initialize the sensor list structure and other configuration variables. Once MCU-B is done with its setup and has set the Flag pin LOW, it will use the Flag pin to notify MCU-A of when the alarm has been set off in the RTC by setting the Flag pin HIGH.

2.2.3 Inter-MCU Communication

Both MCUs use a software serial interface to communicate asynchronously. MCU-A will receive a command from the Bluetooth module and process it, where then it can send the appropriate command to MCU-B to either store the updated configuration or to get stored data.

2.3 Real-Time Clock (DS3231)

The real-time clock (RTC) allows the hub system to keep track of time when logging the data from the sensors. The RTC also has alarm functionality which enables an alarm to be set for the next sensor check. Using the RTC allows for consistent logging intervals and allows a timestamp to be added to the data logs of when each data point was recorded. Even while the hub is powered off, the RTC will still keep track of time using a 3v Lithium coin battery which will allow the hub to continually keep track of time.

2.3.1 Time and Date Format

The RTC will output the format of “Hr:Min:Sec” for time and “Year/Month/Day.” The time will be also be in a 24-hour format to simplify the time schedule when setting the alarms for data logging and sending data to the portal. This date format is used due to its international understandability.

2.3.2 Alarms

The DS3231 is capable of using two alarms and setting a pin once an alarm goes off. Both alarms are used to keep track of when to perform certain actions. One alarm is for the checking the sensors while the other alarm is used for sending data back to the portal using the GSM module. These separate alarms are needed as sending data back to the portal may not be needed as frequently or may not be viable due to text messaging rates that may apply. This also allows the sensors’ data to be logged at a higher frequency and to be stored in the SD card for the entire scope of the data.

2.4 GSM Module (SIM800L)

The GSM module is used to send text notifications to the farmer when a sensor detects a temperature or humidity that is past the critical point and in ideal conditions for mold growth. Data from the

sensors can also be sent to the University Portal through the usage of text messages with the data from the sensors.

2.4.1 Voltage and Current Requirement

The SIM800L module has a relatively strict requirement on voltage and current. The power supply range is from 3.4v to 4.4v. Due to the possible massive data transmission, a 2A current should be provided by power supply.^[1] In the circuit design we are using the same power supply system as recommended by SIMCom company in their hardware design manual.

As for Serial Communication:

Table 1: Logic levels of the serial port

Parameter	Min	Max	Unit
V _{IL}	0	0.4	V
V _{IH}	2.4	3.0	V
V _{OL}	0	0.1	V
V _{OH}	2.7	3.0	V

If user's MCU or PC's voltage is out of the range, level shifter needs to be used.

Two voltage level shifter is used between MCU serial port and SIM800L serial port. The reference voltage for SIM800L communication is 2.8v, generated by a 5v to 2.8 voltage regulator.

2.4.2 AT Commands

As we only use GSM module to send SMS, the AT command is really simple. First we make sure GSM works in the full function module, then we select SMS format. After setting the phone number, we should be able to send SMS to the number. A softwareserial instance is used in the data transmission between GSM module and MCU. Always make sure using "listen" to switch to GSM softwareserial communication, because only one softwareserial line is available at one time.

We might want to implement sleep function later for future development^[7]. All the AT Commands can be found in the official AT command manual^[6].

2.5 SD Card Module

The SD card module is necessary for storing that data from the sensors. The SD card contains multiple files used for storing configuration data, sensor MAC addresses and sensor data. On boot up, the SD card will provide the initial configuration setup and will update the configuration file as modifications are made through the mobile application. There will also be a separate file that contains the sensor MAC addresses that are used to load up in a list structure. Lastly, there are also data files which contain the sensor data gathered from the sensors during a sensor check^[2].

2.5.1 Files

The currently used files in the SD card include:

- **“CONFIG.TXT”** - Stores hub configuration data that is used for initialization during the boot up process and is updated when the mobile application sends a command updating
- **“SENSORS.TXT”** - Contains all the data for the sensors that the hub gathers data from. This data is used to create and initialize the sensor list that is used so the hub can connect to each sensor. This file is updated each time a sensor is added or removed.

2.5.2 Data Storage

In addition to the configuration files detailed above, there will be a set of data files which contain the data gathered from the sensors. As for the file names, they will be generated using the date obtained from the RTC. This ensures that there is a unique filename for each day that has been logged. It also allows for faster searching for the needed file when it is requested by the phone app.

2.6 Bluetooth Module (HM-10)

2.6.1 Voltage and Current Requirement

The HM-10 requires a voltage of 3.3v for its Vcc and also requires a level conversion from the MCU of 5v->3.3v, as voltages any higher can cause damage to the module^[3].

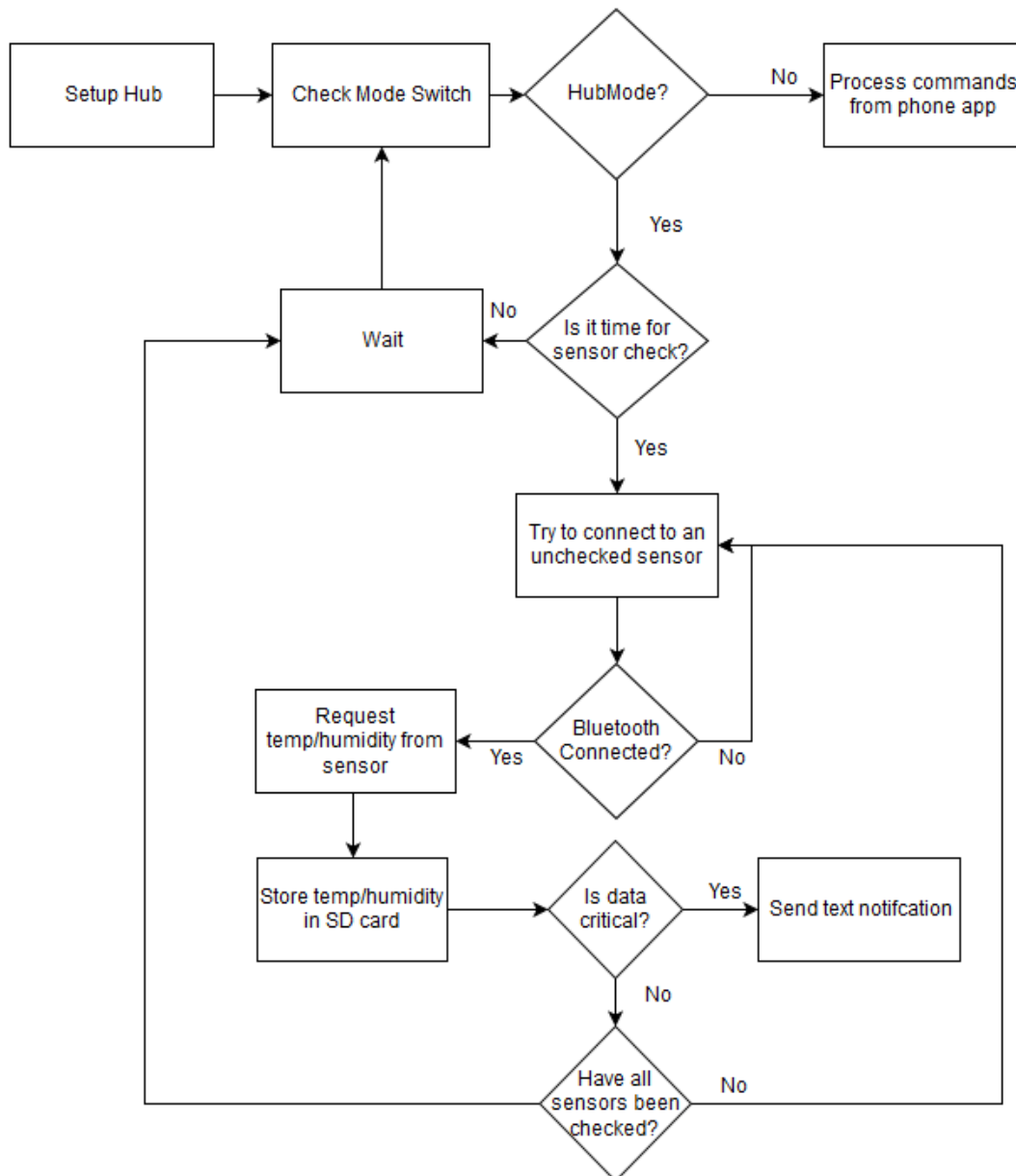
2.6.2 Bluetooth Low Energy

The purpose of using Bluetooth Low Energy (BLE) is to lower the power consumption of the hub and the connected sensors. By using BLE instead of regular Bluetooth, the sensors are able to also use BLE

and be compatible with the hub. This ensures that the sensors are also able to use less power and have longer battery lives.

3. Software Design

3.1 Software Flowchart



3.2 Sensor Data Structure

The current data structure that is used to store the sensors consists with the a 12 byte array that contains the MAC address of the sensors. This sensor structure can later be adapted to also store the sensor ID along with any other relevant data about the sensor. These sensor structures are then inserted into a linked list structure which allows for easy adding/removing of a sensor and allows the hub to loop through each sensor and try to connect to it using the stored MAC address ^[4]. The list structure adds a new sensor structure every time there is a new sensor added to the hub through the phone app. When there is a change to the list, the hub also updates the sensors' data in the SD card in order to be able to retrieve the list of sensors the next time the hub boots up.

4. Costs

Part Number	Part	Price	Qty	Part Total	Package
Atmega328P-PU	Microcontroller	\$3.70	2	\$7.40	
HM-10	Bluetooth Module	\$6.50	2	\$6.50	
DS3231(With Alarm)	RTC Module	\$6.00	1	\$6.00	
SIM800L	GSM Module	\$4.00	1	\$4.00	
SD Module	SD Module	\$2.00	1	\$2.00	
Crystal Oscillator	Oscillator	\$0.33	2	\$0.66	
AP2210N-3.3TRG1DICT-ND	5v->3.3v Regulator	\$0.42	1	\$0.42	SOT-23-3
MIC29302AWU-TR	Adj (5v->4v) Regulator	\$1.95	1	\$1.95	TO-263-5
AP2120N-3.0TRG1DICT-ND	5v->3v Regulator	\$0.37	1	\$0.37	SOT-23-3
Resistor 100K	Resistor	\$0.10	1	\$0.10	(Series:RC) 0402
Resistor 43K	Resistor	\$0.10	1	\$0.10	(Series:RC) 0402
Resistor 10K	Resistor	\$0.10	10	\$1.00	(Series:RC) 0402
Resistor 20K	Resistor	\$0.10	3	\$0.30	(Series:RC) 0402
Resistor 220	Resistor	\$0.10	3	\$0.30	(Series:RC) 0402
Resistor 330	Resistor	\$0.10	2	\$0.20	(Series:RC) 0402
Capacitor 330uF	Capacitor	\$0.51	1	\$0.51	PANASONIC_E(8mm)
Capacitor 100uF	Capacitor	\$0.62	1	\$0.62	(Series:CC) 1206
Capacitor 2.2uF	Capacitor	\$0.16	2	\$0.32	(Series:CC) 1206
Capacitor 1.0uF	Capacitor	\$0.17	3	\$0.51	(Series:CC) 1206
Capacitor 100nF	Capacitor	\$0.11	1	\$0.11	(Series:CC) 1206
Capacitor 22pF	Capacitor	\$0.17	4	\$0.68	(Series:CC) 1206
BSS138	Transistor	\$0.22	3	\$0.66	SOT23-3
EG1903-ND	Slide Switch	\$0.58	2	\$1.16	SWITCH-SPDTPTH
LED	Red LED	\$0.28	1	\$0.28	3mmLED
LED	Green LED	\$0.25	1	\$0.25	3mmLED
LED	Blue LED	\$0.40	2	\$0.80	3mmLED
Total				\$37.20	

5. Troubleshooting

Issue: The serial monitor is showing “e 1” for the configurations from the SD card.

Solution: Make sure the SD card is properly inserted into the SD slot and restart the Hub. If issue persists then look at the SD card files to make sure they are formatted correctly.

Issue: The sensor is returning a zero value for temperature/humidity to the Hub.

Solution: This is most likely an issue with the sensor itself as there may be an issue communicating with it or an issue with the Bluetooth on the sensor board. The fix would be to make sure the sensor is able to send correct values by using the phone app.

Issue: MCU-B is not returning “d” after a command has been sent.

Solution: This is usually caused if MCU-B has not finished setting up and MCU-A has started sending commands. The fix would be to restart MCU-B first and wait until it is done and then restart MCU-A so that it can properly send its commands to get the configuration data.

Issue: One of the microcontrollers keeps resetting during an operation.

Solution: This is usually caused by a memory issue, either by accessing random memory or not having enough memory that the stack and heap collide. The solution to this issue is to make sure that memory is being handled properly and that not too many variables are stored in dynamic memory.

Issue: The Bluetooth isn’t connecting or responding properly.

Solution: Make sure that the module is getting its proper voltage of 3.3v and that the Rx is receiving 3.3v signal, as higher voltages could damage the module. Also there could be an issue with the Bluetooth settings not being properly set to allow proper communication.

Issue: The GSM module is not sending messages.

Solution: There might be multiple reasons to this issue. A low voltage supply, a weak signal environment, a loosen SIM card contact could all lead to this problem. Always check the status indication LED on the module first. See the status LED behavior table below:

Table 33: Status of the NETLIGHT pin

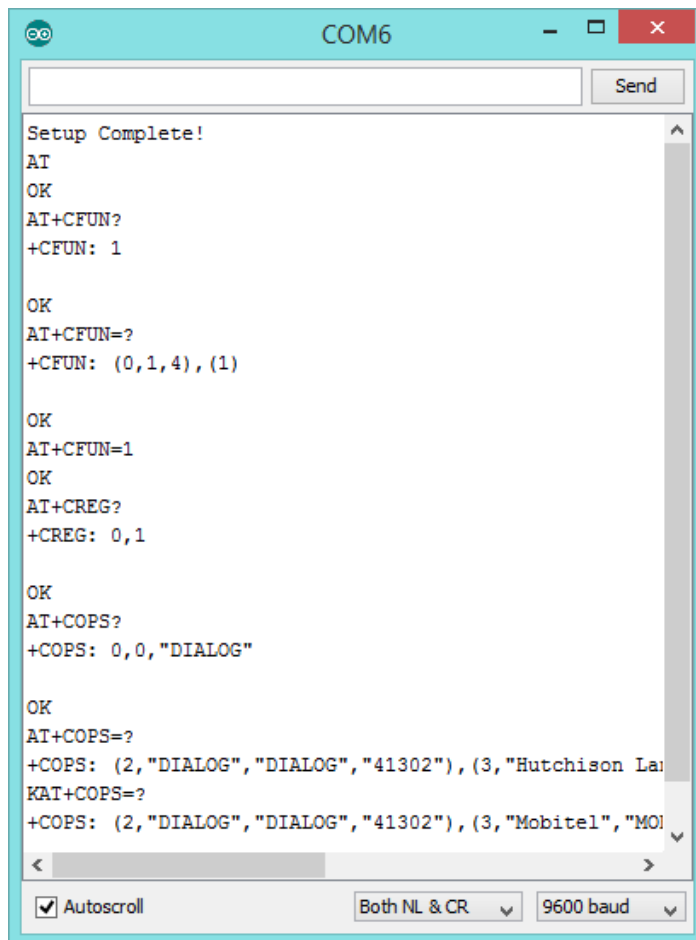
Status	SIM800 behavior
Off	SIM800 is not running
64ms On/ 800ms Off	SIM800 not registered the network
64ms On/ 3000ms Off	SIM800 registered to the network
64ms On/ 300ms Off	GPRS communication is established

If SIM800L is not running, then check the input voltage level. A 3.4v-4.4v voltage input is preferred. A lower voltage or higher voltage will both make SIM800L doesn't work.

If SIM800L is not registered to the network, then there might be a weak signal issue. Solution:

a. Always use an antenna.

b. Use AT commands to check if the initialization of GSM module is stuck on some certain step



```
Setup Complete!
AT
OK
AT+CFUN?
+CFUN: 1

OK
AT+CFUN=?
+CFUN: (0,1,4), (1)

OK
AT+CFUN=1
OK
AT+CREG?
+CREG: 0,1

OK
AT+COPS?
+COPS: 0,0,"DIALOG"

OK
AT+COPS=?
+COPS: (2,"DIALOG","DIALOG","41302"), (3,"Hutchison La
KAT+COPS=?
+COPS: (2,"DIALOG","DIALOG","41302"), (3,"Mobitel","MOI
```

AT+CSQ//Check the signal strength

AT+CREG? //Check the network registration status (if searching, if registered, if roaming, if denied)

AT+CFUN? //Check if some functionality is disabled

AT+COPS? //Check which network operator is currently working on.

A good response of the AT command is shown in this picture on the left^[5].

All the AT commands can be found on the AT command manual^[6].

6. Conclusion

6.1 Future Developments

For the hub, there can be multiple additions which give it a better scope to operate. One such development would be to use the GSM to send data directly to the University Portal using its 3G capabilities to send data directly to a server which might help alleviate the issue of sending a certain amount of text data through SMS to the University Portal. Another possible development would be to allow the University Portal to be able to request data from the SD card similar to how the phone app is able to. This allows the University portal to further dictate when they want the full data from the hub instead of getting it in pieces through text messages. Lastly, one larger development would be to change out the Atmega328P microcontroller for an ARM based microcontroller, such as the LPC1114, which contains more dynamic memory to run all the modules, while keeping the cost relatively low with one chip instead of two.

6.2 Future Improvements

As for future improvements on the hub there are a few things that could be done to allow the hub to run more smoothly. One possible improvement would be to better detect when there is an SD card present and handle the condition of if it is not present. This is helpful, as the SD card can encounter issues if it is removed while the Hub is running or if it is writing data into a file. There is also an improvement of allowing the Hub to be able to sleep while it is waiting for the alarm from the RTC. This would allow the Hub to greatly use less power from the solar charging system. Another improvement would be the usage of error codes sent to the phone app and to the University Portal, in order to help know of any potential issues, such as not connecting to a sensor or an issue with reading/writing data in a file from the SD card.

7. References

- [1] *SIM800_Hardware_Design_V1.05*. Tech. Shanghai Simcom Wireless Solutions Ltd., 25 Mar. 2014. Web. <https://www.itead.cc/wiki/images/c/c7/Sim800_hardware_design_v1.05.pdf>.
- [2] "SD Library." Arduino - SD. Arduino LLC, n.d. Web. 28 Aug. 2016. Web. <<https://www.arduino.cc/en/Reference/SD>>.
- [3] Bluetooth 4.0 BLE Module Datasheet. Tech. JNHuaMao Technology Company, 8 Mar. 2014. Web. <http://fab.cba.mit.edu/classes/863.15/doc/tutorials/programming/bluetooth/bluetooth40_en.pdf>.
- [4] "Ivanseidel/LinkedList." *GitHub*. N.p., n.d. Web. 28 Aug. 2016.
- [5] Wijethunga, Ayoma Gayan. "Quickstart SIM800 (SIM800L) with Arduino." Website and Blog of Ayoma Wijethunga. N.p., 25 Nov. 2015. Web. <<http://www.ayomaonline.com/programming/quickstart-sim800-sim800l-with-arduino/>>.
- [6] *SIM800 Series_AT Command Manual_V1.09*. Tech. Shanghai Simcom Wireless Solutions Ltd., 3 Aug. 2015. Web. <https://www.itead.cc/wiki/images/c/c7/Sim800_hardware_design_v1.05.pdf>.
- [7] *SIM800 Series_Embedded AT_Sleep_Application_Note_V1.01.pdf*. Rep. Shanghai Simcom Wireless Solutions Ltd., 10 Feb. 2015. Web. <https://cdn-shop.adafruit.com/product-files/1946/SIM800+Series+Embedded+AT+Sleep+Application+Note_V1.01.pdf>.

Appendix A: Mobile Application Commands

CMD	Function	Parameter	Return
1	AT Test (or Reset to Factory)	None	
2	Get Hub ID	None	ID # (4 bytes)
3	Set Hub ID	ID # (4 bytes)	
4	Get Number of Sensors	None	# of Sensors
5	Get List of Sensors	None	ID Address
6	Get a Sensor ID/Address	Sensor #	ID Address
7	Set a Sensor ID	Sensor# ID	
8	Add a Sensor	MAC Address (12 bytes)	
9	Remove a Sensor	Sensor#	
10	Remove all Sensors	None	
11	Get Alert Phone #	None	
12	Set Alert Phone #	Phone #	
13	Get Portal Phone #	None	
14	Set Portal Phone #	Phone #	
15	Get Portal Notification Freq.	None	
16	Set Portal Notification Freq.	# of hours between notifications	
17	Get Logging Frequency	None	
18	Set Logging Frequency	# of hours between sensor checks	
19	Get Hub Time	None	hour min sec
20	Set Hub Time	Time (8 bytes)	
21	Get Hub Date	None	year month day
22	Set Hub Date	Date (10 bytes)	
23	Get Critical Temperature	None	
24	Set Critical Temperature	Temperature	
25	Get Critical Humidity	None	
26	Set Critical Humidity	Humidity	20
27	Get Data	year month day hour duration	date, time, sensor#, T, H \n

Appendix B: Intra-MCU Commands

CMD	Function	Parameters	Return
0	Reset	/	sensor# MAC phone# status(d/e)
1	SetTime	hour min sec	d
2	GetTime	/	hour min sec
3	SetDate	year month day	d
4	GetDate	/	year month day
5	CheckErrorHistory	/	date time errorcode
6	SetData	sensor# T H	d
7	GetData	year month day hour duration	date, time, sensor#, T, H \n
8	SetMacAddress	MAC	d
9	GetMacAddress	sensor#	MAC
10	SetPortalPhone#	Phone# (12 bytes)	d
11	GetPortalPhone#	/	Phone# (12 bytes)
12	SetNotificationPhone#	Phone# (12 bytes)	d
13	GetNotificationPhone#	/	Phone# (12 bytes)
14	SetPortalNotificationFreq	Frequency	d
15	GetPortalNotificationFreq	/	Frequency
16	SetLoggingFreq	Frequency	d
17	GetLoggingFreq	/	Frequency
18	SetCritTemp	Critical Temperature	d
19	GetCritTemp	/	Critical Temperature
20	SetCritHumidity	Critical Humidity	d
21	GetCritHumidity	/	Critical Temperature
22	SetHubID	HubID (4 bytes)	d
23	GetHubID	/	HubID (4 bytes)
24	TurnOffFlag	/	d
25	RemoveAllSensors	/	d