Development of Low Power Wireless Sensor Network using Zigbee Protocol for Terrace Farm

EE 692 R&D Project

Submitted by
Saurav Shandilya (153076004)

Under the guidance of
Prof. Kavi Arya

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY BOMBAY

May 4, 2017

Abstract

The objective of this project is to design a Wireless Sensor Network (WSN) to facilitate automatic irrigation and monitoring of a Terrace Farm setup in KReSIT building. The WSN comprises of multiple sensor nodes, collecting sensors data at different points. These data are transmitted periodically to a central server for logging, analysis and further processing.

In this project we have designed a low power sensor nodes using Zigbee Protocol (IEEE 802.15.4). This project focuses on optimizing the power consumption on sensor node by using various software and hardware techniques. Sensor node is designed using ultra-low power MSP430 series microcontrollers, further optimized to work in low power configuration. Xbee modules (working on Zigbee protocol) are used for communication. Each sensor node consist of temperature and humidity sensors for validating the low power operation and proper working of communication. These sensors nodes are battery operated using a 4.2V LiPo battery.

In this report we present and discuss our design methodology, system architecture, low power optimization techniques and results obtained by testing the system.

Contents

1 Introduction

A wireless sensor network (WSN) consist of spatially distributed autonomous tiny computing devices, equipped with sensors, wireless radio, microcontroller, and a power source. These sensor networks are deployed in the different physical environment to monitor a wide range of environmental phenomena. Application of WSN can be found in very wide application area such as healthcare, defence, environment monitoring, home and office automation, wildlife etc.

Implementation of WSN has various challenges. One major major challenges in sensor networks is Energy efficiency. Energy conservation schemes are used to decrease the energy consumption of device and increase the device and network life. However, nowadays, the focus is more on wireless, distributed, sensing nodes. In most cases, the environment to be monitored does not have an existing infrastructure for either energy or communication. It becomes imperative for sensor nodes to survive on small, finite sources of energy and communicate through a wireless communication channel.

The backbone of a sensor node is microcontroller and communication module. The general architecture of wireless sensor network we have developed is shown in Figure 1

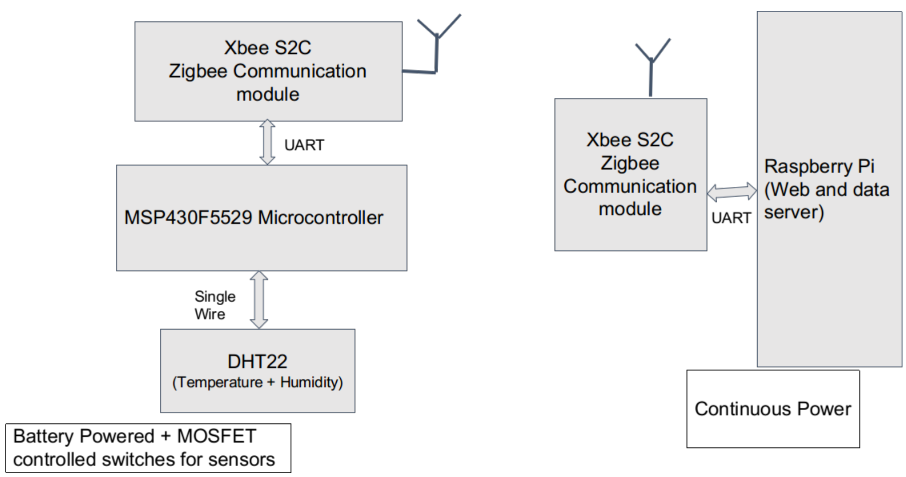

Figure 1: Architecture of Wireless sensor network

The system has two components, first is sensor node designed using MSP430F5529 microcontroller, DHT22 sensor and Xbee communication module. The second part of sensor network is Raspberry Pi based central server.

Xbee connected to Raspberry Pi (know as coordinator) sends signals to sensor node to read the data and send it back. Data received by central server is then plotted on thinkspeak cloud service for real-time monitoring.

Details about each component is discussed in section 3.

2 Literature Survey

2.1 Wireless Protocols

Bluetooth (over IEEE 802.15.1),ZigBee (over IEEE 802.15.4), and Wi-Fi (over IEEE 802.11) are some of the protocol standards for short-range wireless communications with low power consumption. From an application point of view, Bluetooth is intended for a cordless mouse, keyboard, and hands-free headset, ZigBee is designed for reliable wireless monitoring and control, while Wi-Fi is directed at computer-to-computer connections as an extension or substitution of cabled networks.

Table 1: Comparision of Bluetooth, Zigbee and Wifi [5])

	Bluetooth	Zigbee	Wi-Fi
IEEE Specs	802.15.1	802.15.4	802.11
Frequency Band	2.4GHz	868/915 MHz or 2.4 GHz	2.4 GHz; 5GHz
Signal Rate	1Mb/s	250Kb/s	54Mb/s
Range	10m	10-100m	100m
T_x Current (mA)	57	24.7	219
R_x Current (mA)	47	27	215

[5] presented a summary stating, Bluetooth and Zigbee are suitable for low data rate applications running on battery power. Protocol like Wi-Fi are better solution for high data rate application such as audio/video streaming, surveillance etc.

3 Sensor Node Design

3.1 Hardware Design

3.1.1 Microcontroller

Microcontroller used for our development is MSP430F5529. It is a 16-Bit RISC Architecture based MSP430 series microcontroller. It has 63 I/O lines, 12bit ADC, UART, SPI, I2C. However the main reason for choosing this microcontroller are the following features which are very important from low power design perspective

- 5 different clock sources and 3 clock signals offering wide operating frequency range
- Supply voltage ranging from 3.6V to 1.8V
- 7 different Low power modes

3.1.2 Communication Module

Communication module used for implementation is Xbee which uses Zigbee protocol.

ZigBee is a communication protocols developed by ZigBee Alliance. The ZigBee protocol is adopted from IEEE 802.15.4 as its Physical Layer (PHY) and Medium Access Control (MAC) protocols [5]. Thus, a ZigBee-compliant device is compliant with the IEEE 802.15.4 standard as well. ZigBee protocol layers are based on the Open System Interconnect (OSI) basic reference model. Protocol is divided into number of layers to facilitate easy modification and abstraction in development. Figure 2 shows the Zigbee protocol layers. The bottom two networking layers are defined by the IEEE 802.15.4 standard[1]. This standard is developed by the IEEE 802 standards committee in 2003 IEEE 802.15.4 defines the specifications for Physical (PHY) and Medium Access Control (MAC) layers of wireless networking. IEEE 802.15.4

does not specify any requirements for higher networking layers. ZigBee standard defines the Network (NWK), Application (APL), and security layers of the protocol and adopts IEEE 802.15.4's Physical and Medium Access Control layers as part of the ZigBee networking protocol.

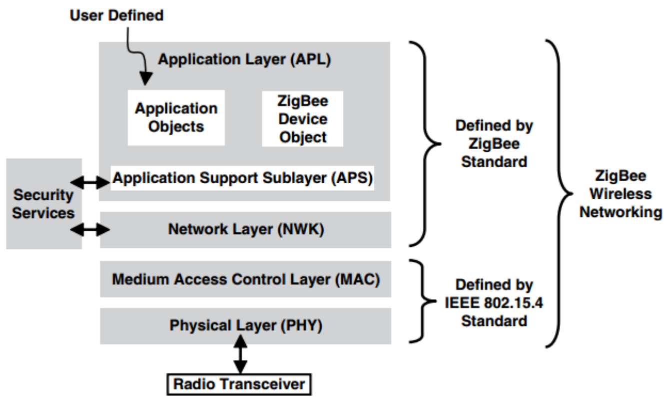

Figure 2: Zigbee Networking Protocol Layers
image courtesy: [1]

The main feature of Zigbee is

- low data rate - Offer a maximum data rate of 250Kbs
- short range
- Frequency band of 2.4GHz. Some countries support 868MHz and 915MHz. However 2.4GHz is most widely used because it is accepted world-wide and they offer maximum data rate and channels.

2.4GHz is widely accepted frequency band and is being used in communication module we are using. However it has a serious limitation of coexistence of IEEE 802.11b (Wi-Fi) in the same band. Also it is difficult for signals at higher frequency to pass through walls as compared to lower frequencies.

ZigBee is targeted mainly for battery-powered applications where low data rate, low cost, and long battery life are main requirements. In many ZigBee applications, the total time the wireless device is engaged in any type of activity is very limited; the device spends most of its time in a power-saving mode, also known as sleep mode. As a result, ZigBee enabled devices are capable of being operational for several years before their batteries need to be replaced.

The Zigbee standard defines three types of device:

- Coordinators -
- Routers
- End Devices

3.1.3 Sensors

We have used a DHT22[2] - Temperature and Humidity sensor to log the data on different locations on terrace farm. Uses capacitive humidity sensing and thermistor for temperature sensing [2]. The sensor comes precalibrated and outputs digital temperature and relative humidity percentage on a single wire.
Operating range of Temperature sensor is -40°C to 80°C
Operating range of Humidity sensor is 0-100%RH
Resolution of Temperature sensor is $\pm$ 0.1°C
Resolution of Humidity sensor is $\pm$ 0.1%RH

As our major objective is to setup a Zigbee based sensor network, we used only one sensor on every sensor node to get the data. We plan to use moisture sensors, soil temperature sensors and leaf-wetness sensors after setting up the basic network infrastructure.

3.1.4 Power Supply

Figure 3 is a rechargeable single cell 4.2V Li-Ion battery for powering our sensor node. Capacity of cell is 2200mAh.

Battery is monitored using the ADC present on the microcontroller. Microcontroller is operating at 3.3V reference voltage. Therefore voltage divider circuit as shown in Figure 4 is used to bring down the voltage to measurable range of 3.3V. Values of resistors are choosen to keep the current across resistors to as low as possible. This ensure low power consumption.

3.2 Low Power Optimization

It is very important for sensor nodes to operate at low power to ensure longevity It is desirable to run the sensor nodes on a compact power source like AA/AAA or coin size battery. In order to ensure long battery life, we need to decrease the current consumption of whole system. There are

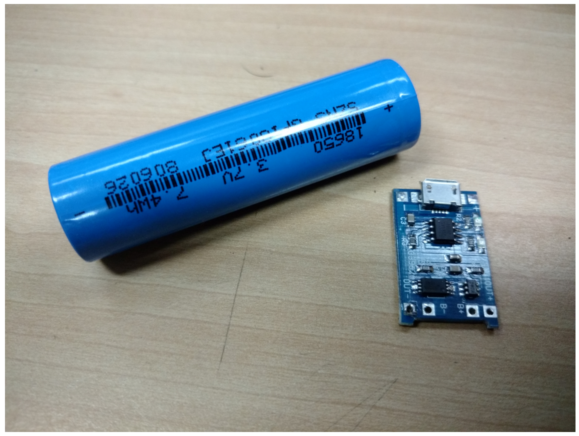

Figure 3: 4.2V Li-Ion battery + charge-discharge protection circuit

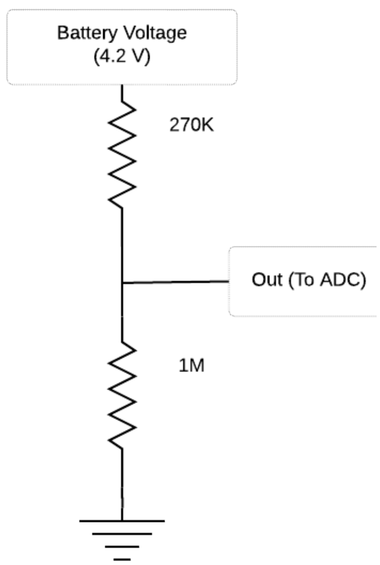

Figure 4: Voltage divider circuit for voltage measurement

various techniques and design consideration that need to get the low power consumption.

Average power consumption of a system can be expressed by equation 1:

$$P_{avg} = (P_{active} + P_{sleep} + P_{transition})/T_{total} \tag{1}$$

where:

P_{active} = Power being consumed in active mode * Time for which system is active
P_{sleep} = Power being consumed in sleep mode * Time for which system is sleep
$P_{transition}$ = Power consumed while making transition from sleep mode to active mode

We discuss various techniques used to reduce the power consumption

3.2.1 Active mode power consumption

Microcontrollers is major component controlling sensors, radio modules and running software algorithm. Therefore It is important to consider power consumption of microcontroller. Microcontrollers are based upon CMOS logic, which consumes power during switching of transistors. In active mode, main power is consumed by the CMOS circuits and called as Active power. These CMOS circuits are digital devices which consume power during transition or switching action. The power consumption is given by the following equation [7]:

$$P = f * C_L * V_{dd}^2 \tag{2}$$

Where:
P: Power Consumed
f: Operating Frequency of controller
C_L: Load capacitance
V_{dd}: Supply Voltage used to run the controller

Load capacitance cannot be changed by application designers because it is determined by the various parameters used during IC fabrication. We are hence left with operating frequency and supply voltage at which the controller can run. From Equation 2 Power consumption of controller is directly proportional to Operating frequency and supply voltage. Lowering these will decrease the power consumption.

Our Design Considerations:

- **Voltage consideration**

  According to datasheet [3] of MSP430F5529, permissible operating voltage is 3.6V to 1.8V. However the launchpad used in project provides input for 5V or 3.3V. We choose the lower one which is 3.3V for our application. Another consideration to be taken care off while choosing supply voltage is kind of load driven by IO pins and sensors output voltage range. If output voltage of sensor is 5V, we can not interface the sensor directly to IO pins of microcontroller operating at 3.3V.

- **Frequency consideration**
  MSP430F5529 controller has a Unified Clock System (UCS) which provides option to run the controller from five different clock sources, each provides frequency range from 10KHz to 25MHz. UCS module of MSP430 architecture provides 3 different clock signals (as shown in Figure 6:

  1. Auxiliary clock (ACLK): Low frequency clock source used by peripherals. Provides software selectable frequency for individual peripherals
  2. Master clock (MCLK): High frequency clock. It is used by the CPU and system.
  3. Subsystem master clock (SMCLK): High frequency clock source used by peripherals. It supplements ACLK which runs peripherals at low frequencies.

     We are using ACLK running at 32768 Hz as our clocking signal. This provides us options to run peripherals like UART and ADC at required rate.

- **Other Considerations**

  1. According to datasheet of MSP430F5529 for preventing leakage all unused GPIO ports they must not be in floating state. All unused GPIO ports and pins must be kept at low level output mode. This is very important for preventing leakage current during Standby mode of the system.
  2. MSP430F5529 launchpad has a USB port for powering of system. It incorporates a dual LDO regulators (3.3 V and 1.8 V) that allow the entire MSP430 device to be powered from 5-V VBUS when it is made available from the USB host. As our application is battery powered, we do not need powering from USB, therefore current consumption through LDO can be decreased by disabling the LDOs in USBPWRCTL register.

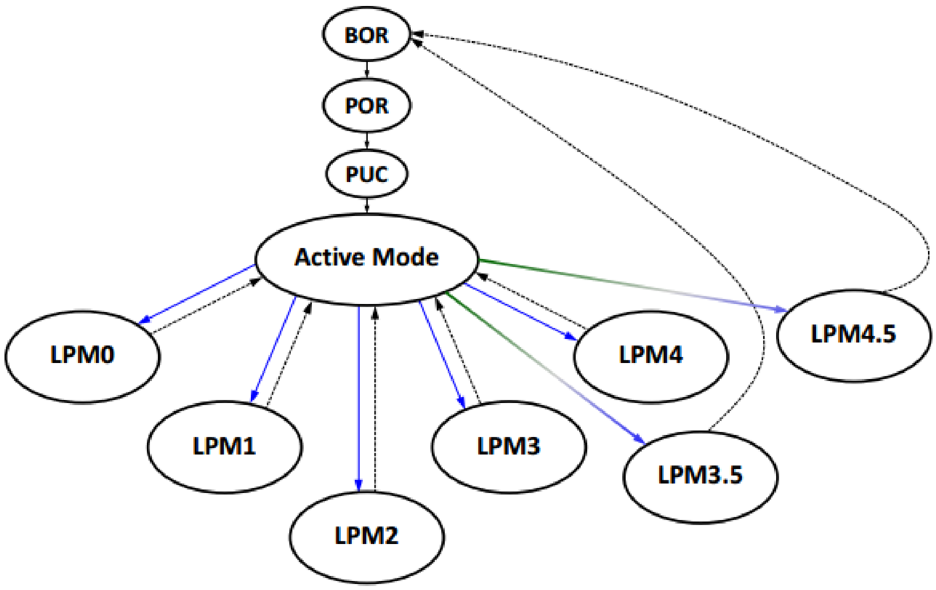

Figure 5: MSP430 Low Power Modes(LPM)
image courtesy:[4]

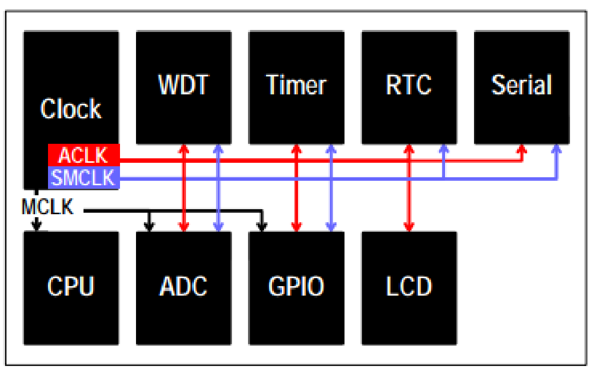

Figure 6: MSP430 Clock Source
image courtesy:[4]

3.2.2 Sleep mode Power Consumption

Another major factor which determines battery life is the power consumption of system in sleep mode. In sleep mode microcontroller is doing nothing and in such case it should consume the least power. We need to make proper configuration to ensure microcontroller is in sleep mode. Beside microcontroller, there are other peripheral devices and circuit components which has their drive current requirements and/or associated leakage current. Therefore in order to reduce power consumption in standby mode, peripheral with low standby current must be used while designing system.

Finally, the power consumed while transitioning from low power standby mode to active mode should not be overlooked. Devices may end up wasting a significant amount of power while transitioning between these two modes.

Our Design Considerations:
MSP430F5529 controller provides 7 different Low Power Modes(LPM) as shown in 5. Each LPM modes deactivates different part of MCU by disabling different types of clocks, thus sending microcontroller into sleep (low power) mode.

There are different ways in which system can switch from sleep mode to active mode. Figure 7 summarizes different effect of different LPM modes, clock sources disabled in each mode and Interrupt source which can make a transition from sleep mode to active mode.

Operating Mode	CPU (MCLK)	SMCLK	ACLK	RAM Retention	BOR	Self Wakeup	Interrupt Sources
Active	☒	☒	☒	☒	☒		Timers, ADC, DMA, WDT, I/0, External Interrupt, COMP, Serial, RTC, other...
LPM0		☒	☒	☒	☒	☒	
LPM1		☒	☒	☒	☒	☒	
LPM2			☒	☒	☒	☒	
LPM3			☒	☒	☒	☒	
LPM3.5					☒	☒	External Interrupt, RTC
LPM4				☒	☒		External Interrupt
LPM4.5					☒		External Interrupt

Figure 7: MSP430 LPM effects on clock source
image courtesy:[4]

- Active Mode: All clocks are active
- LPM0: MCLK is disabled and hence CPU is disabled
- LPM3: MCLK and SMCLK are disabled, hence peripherals running on SMCLK and CPU is diabled. However peripherals running on ACLK is active.

We have chosen LPM3 mode for our application. The reason for choosing LPM3, is that modes below it needs an external interrupt source for transition to active mode. Our application do not have any such provision.

LPM3 consumes $1.9\mu A$ and Active mode consumes 180 $\mu A/MHz$ [4]

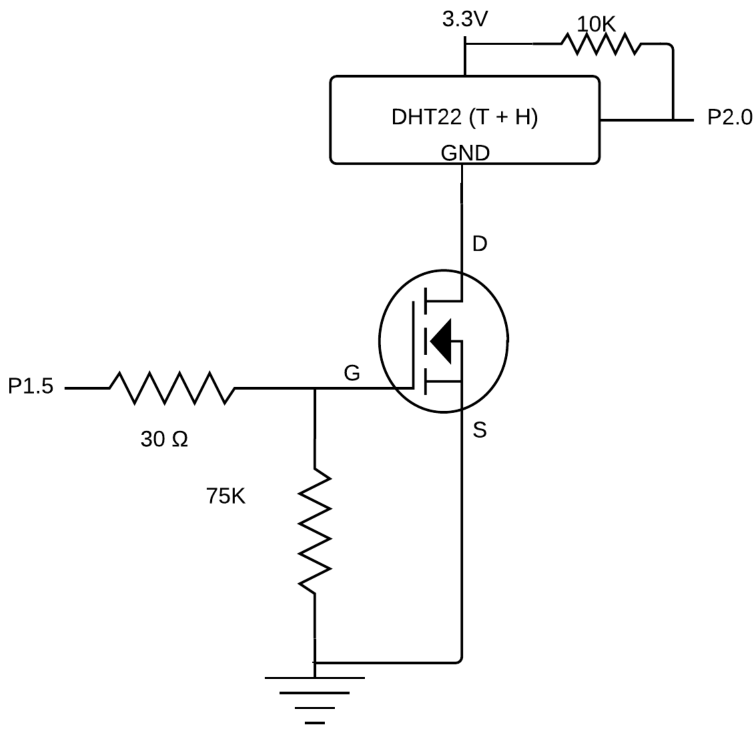

Figure 8: DHT22 Interfacing

Power management of sensors:
In low power battery operated system, controlling the peripheral on and off period is very important. We just keep the device in on state continuously when the system is in sleep mode. During sleep mode all peripherals must be switched off. So best option is to power these devices from the microcontroller GPIO ports only. But power supplied by GPIO may not be sufficient for running the peripherals. So external switch like MOSFET is required to source or sink current of the peripheral. This MOSFET switch can be turned on or off by the GPIO and completely cut off the power of the peripherals. We have used MSOFET to turn on/off the DHT22 sensors.

3.3 Circuit Design

As discussed in previous section, our objective is to desing a low power sensor node. We designed a stackable PCB for our sensor node. This give us provision to plug and play any extra sensors as per need of application.

Circuit for DHT22 interfacing with microcontroller is shown in Figure 8.

Figure 9 and Figure 10 shows pin connections made on microcontroller and Raspberry Pi.

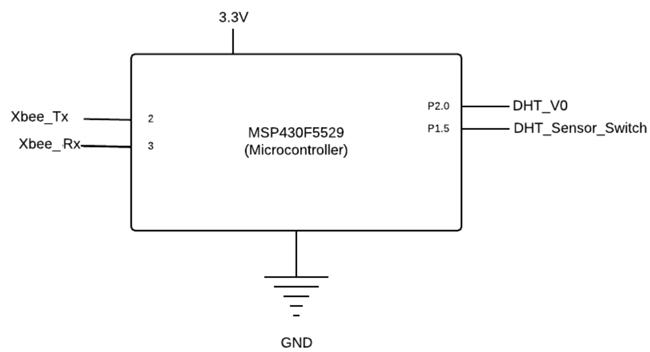

Figure 9: MSP430 Pin Connections

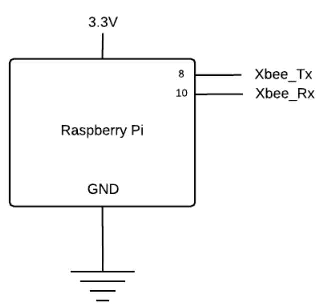

Figure 10: Raspberry Pi Pin Connections

We designed a stackable PCB for the sensor node. Stack has connections for DHT22 sensor and Xbee module. This stack can be fixed on top of 40 pin booster pack header of MSP430F5529 launchpad. Schematic and PCB board is shown in Figure 11 and Figure 12 respectively.

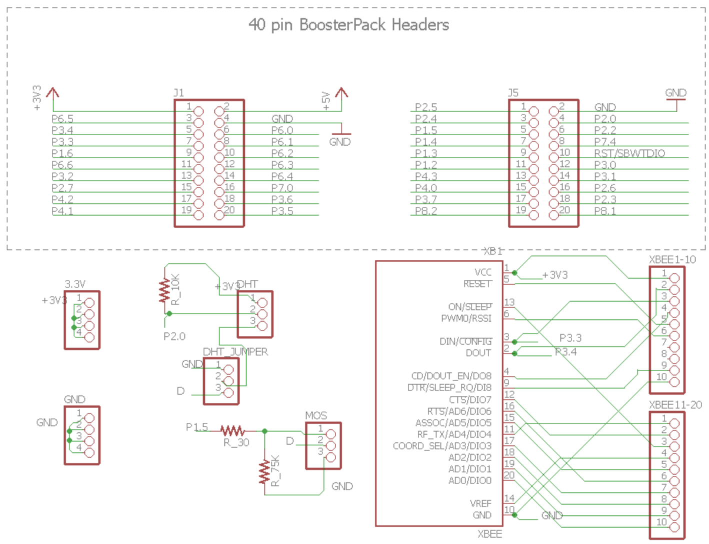

Figure 11: Schematic for Stack

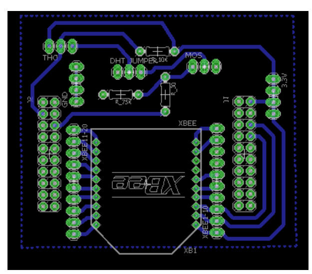

Figure 12: PCB Board for Stack

3.4 Software

The whole system consist of two parts,

- Sensor node - Microcontroller with sensor,Xbee and battery.
- Central Server - Raspberry Pi with Xbee.

For the sensor node, program is written on microcontroller in C language.

On Raspberry Pi, Python language is used to read the serial data, do necessary processing and pushing it on cloud for real-time plotting.

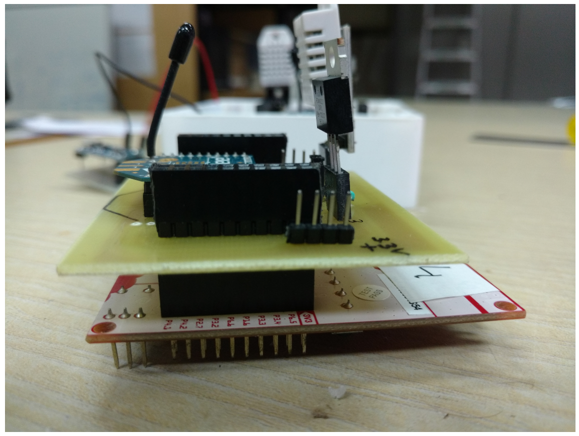

Figure 13: Sensor Node in form of a Stack

4 Experiment and Observation

We designed and deployed 3 sensor nodes on the terrace farm. The readings for temperature, humidity and battery voltage are logged and displayed on thinkspeak cloud service. Figure 14 shows the reading

In order to measure the power consumption of microcontroller we performed set of experiments and recorded the current following through the system. We used Digital multimeter to record the readings.

Figure 15 shows that microcontroller in LPM3 mode. The ideal LPM3 mode current consumption as mentioned in datasheet should be around 2μA. We got around 6μA.

5 Challenges/Issues:

- Low power configuration and current measurement in existing system is a cumbersome process.
- Lack of good datasheet and tutorials for microcontroller.
- Creating a algorithm to ensure proper data transmission between a coordinator and multiple node.
- We have implemented Star topology considering it to be more power efficient and given the perimeter of terrace farm, extender is not re-

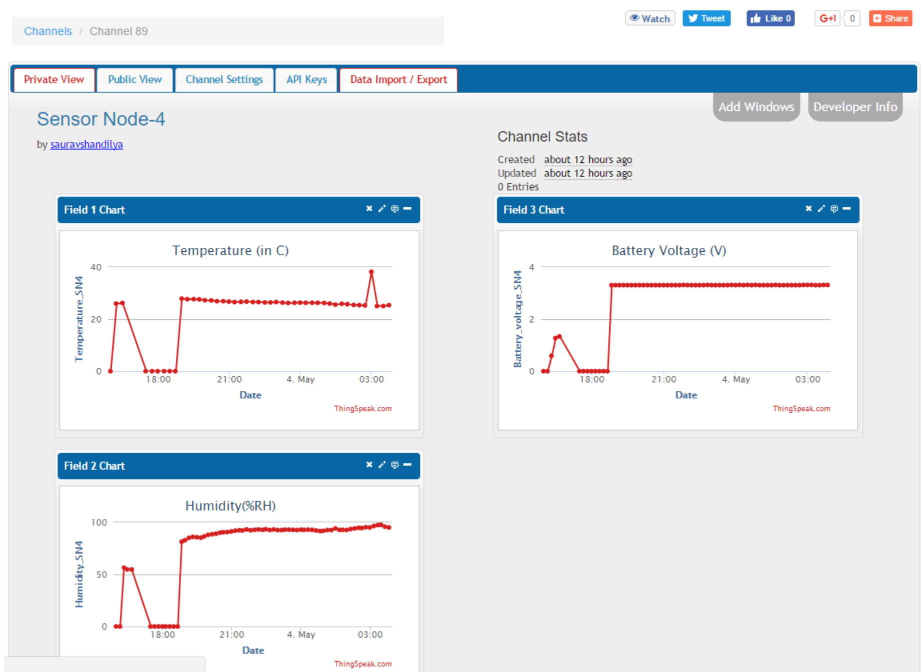

Figure 14: Sensor Node reading

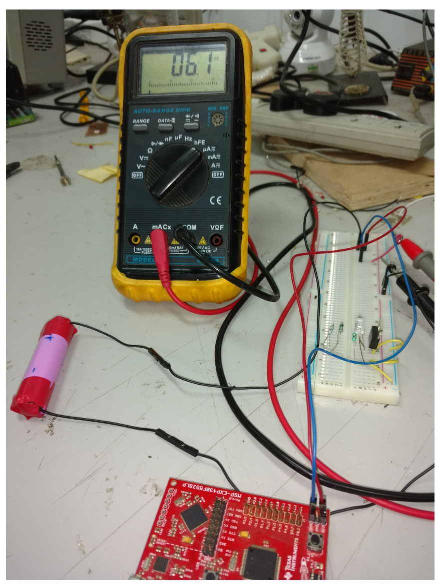

Figure 15: LPM3 Sleep

quired. However a more robust solution would be to use mesh network which gives feature of auto-healing and network discovery, at cost of more power consumption. A through testing needs to be done to check the range of existing system.

6 Future work

- Doing a range test on terrace farm and adding more number of sensor nodes
- Explore option of creating mesh network with xbee
- Develop sensor node with bluetooth and compare and contrast performance of both the wireless protocol.
- Add other required sensors like moisture, leaf wetness, soil temperature in the node
- Solar power options to run the devices
- Using Network simulator

References

[1] Shahin Farahani, "ZigBee Wireless Networks and Transceivers", Book chapters.

[2] Aosong Electronics Co.Ltd, DHT22 Datasheet, `https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf`

[3] Texas Instruments, "Datasheet for MSP430F552x, MSP430F551x Mixed-Signal Microcontroller" `http://www.ti.com/lit/ds/symlink/msp430f5529.pdf`

[4] Texas Instruments, "MSP430 Design Workshop, Student Guide", Rev.4.01, Feburary,2015

[5] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen, "A Comparative Study of Wireless Protocols: Bluetooth, ZigBee, and Wi-Fi", The 33rd

Annual Conference of the IEEE Industrial Electronics Society (IECON), Nov. 5-8, 2007, Taiwan

[6] Robert Faludi, "Building Wireless Sensor Networks"

[7] Sanam Shakya, "Sensor Development and Energy Analysis of Low Power Nodes for Wireless Sensor Network", Master of Technology, Dissertation Report, IIT Bombay

[8] Tamoghna Ojha, Sudip Misra, Narendra Singh Raghuwanshi, "Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges", Computers and Electronics in Agriculture, September,2015

[9] A. Cebrian, J. Rey, A. Tomos, J. Millet, "Adapting power consumption to performance requirements in a MSP430 microcontroller", IEEE, 2005