2011

Implementation of a wireless sensor network with eZ430-RF2500 development tools and MSP430FG4618/F2013 experimenter boards from Texas Instruments

Lu De Yang
Louisiana State University and Agricultural and Mechanical College, lyang8@tigers.lsu.edu

IMPLEMENTATION OF A WIRELESS SENSOR NETWORK WITH EZ430-RF2500 DEVELOPMENT TOOLS AND MSP430FG4618/F2013 EXPERIMENTER BOARDS FROM TEXAS INSTRUMENTS

A Thesis
Submitted to Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

in

The Department of Electrical & Computer Engineering

by
Lu De Yang
Bachelor of Science in Electrical Engineering, Jilin University, China, 2009
August 2011

Dedicated to my parents

ACKNOWLEDGEMENTS

I would like to appreciate, profoundly, my major advisor Dr. Morteza Naraghi-Pour who supports me consistently and has contributed his time, talent, and friendship so much. His knowledge of research is invaluable. He has provided excellent suggestion and feedback on my thesis with constructively pointing out my mistakes. Without his patient guidance, I cannot grow so fast in my research.

I would also like to thank Gustavo Chacon Rojas who has provided technique supports and information which helps me a lot.

Finally I would give profound thank to my beloved parents who stand behind me and encourage me all the time. And my uncle and auntie here also support me. Without them, I could not be able to finish my thesis.

TABLE OF CONTENTS

ACKNOWLEDGMENTS……………………………………………………………………….....iii

LIST OF TABLES……………………………………………………………………..…vi

LIST OF FIGURES……………………………………………………………………...vii

ABSTRACT…………………………………………………………………..……… ix

CHAPTER 1. OVERVIEW OF WIRELESS SENSOR NETWORK…………………………….1
1.1 Introduction……………………………………………………………………1
1.2 Sensor Nodes…………………………………………………………………….1
1.3 Wireless Sensor Networks………………… ……………………………………2
1.4 Issues in Wireless Sensor Networks……………………………………………..3
1.4.1 Energy Efficiency…………………………………………………...…4
1.4.2 Localization………………………………………………………………….5
1.4.3 Routing………………………………………………………………………5
1.5 Thesis Contribution and Organization……………………………………………...6

CHAPTER 2. TI SENSOR NODE AND BOARD……………………………………………7
2.1 eZ430RF2500 Development Tools………………………………………………7
2.1.1 Introduction…………………………………………………………………7
2.1.2 SimpliciTI Network Protocol……………………………………………..9
2.1.3 eZ430-RF2500 Wireless Sensor Monitor………………………………...9
2.2 MSP430FG4618/F2013 Experimenter Board……………………………………..12
2.3 CC2500 Wireless RF Transceiver………………………………………………...16
2.3.1 Introduction……………………………………………………………...16
2.3.2 Configuration Registers…………………………………………………19
2.3.3 Transaction through SPI………………………………………………...21
2.3.4 Packet Format…………………………………………………………...22
2.3.5 Radio Control……………………………………………………………23
2.3.6 Frequency Programming……………………………………………….....25
2.3.7 Modulation Format and Output Power Programming…………………………..26
2.4 Microcontrollers……………………………………………………………………...28
2.5 IAR Embedded Workbench Kickstart……………………………………………31
2.5.1 Creating a New Project………………………………………………….31
2.5.2 Setting Project Option……………………………………………………...33
2.5.3 Downloading and Debugging……………………………………………...35

CHAPTER 3. IMPLEMENTATION OF A WIRELESS SENSOR NETWORK..……………...37
3.1 Sensor Network Structure…………………………………………………………37
3.2 Basic Communication……………………………………………………………..39
3.3 Multiple Access and Channelization……………………………………….........39
3.3.1 Time Division Multiple Access (TDMA)……………………………………..40

3.3.2 Scheduling……………………………………………………………………40
3.3.3 Frequency Division Multiple Access (FDMA)…………………………………40
3.3.4 Polling Scheme………………………………………………………………42
3.4 Transmission of the Temperature……………………………………………………43
3.5 Operation of the Whole Network……………………………………………………43
3.5.1 Design for Power Saving……….…………………………………………………44
3.5.2 Program Flow…………………………………………………………………46
3.6 Real World Application………………………………………………...………49

CHAPTER 4. CONCLUSION……………………………………………………………..51
4.1 Summary…………………………………………………………………………..51
4.2 Future Work………………………………………………………………………….52

REFERENCES…………………………………………………………………………..53

APPENDIX A. CODE OF EZ430RF2500 FOR CHANNELIZATION………………………...54

APPENDIX B. CODE OF BOARD1 FOR CHANNELIZATION……………………………...56

APPENDIX C. CODE OF BOARD2 FOR CHANNELIZATION……………………………...59

VITA……………………………………………………………………………………..61

LIST OF TABLES

Table 1. External Components…………………………………………………………………...18

Table 2. Configuration Registers……………………………………………………………...19

Table 3. Status Byte…………………………………………………………………………...22

Table 4. TXOFF_MODE and RXOFF_MODE states………………………………………...25

Table 5. Output Power Levels………………………………………………………………...27

Table 6. Power Mode and Status of Clocks and CPU………………………………………...30

LIST OF FIGURES

Figure 1. eZ430-RF2500T Board and USB Debugging Interface……………………………...8

Figure 2. eZ430-RF2500 Battery Board……………………………………………………..8

Figure 3. eZ430-RF2500 Wireless Sensor Monitor…………………………………………...11

Figure 4. MSP430FG4618/F2013 Experimenter Board………………………………………12

Figure 5. CC2500EMK……………………………………………………………………..13

Figure 6. Experimenter Board Block Diagram……………………………………...…...14

Figure 7. MSP-FET430UIF…………………………………………………………………...15

Figure 8. Jumper Settings for Power Selection………………………………………………..15

Figure 9. CC2500 Simplified Block Diagram………………………………………………...17

Figure 10. Pin Configuration of CC2500(QLP 4×4 mm Package, 20 pins)……………………..17

Figure 11. Typical Application of CC2500 and Evaluation Circuit…………………………..18

Figure 12. Packet Format……………………………………………………………………...23

Figure 13. Radio Mode Control State Diagram……………………………………………….24

Figure 14. Patable……………………………………………………………………………..27

Figure 15. MSP430F2274……………………………………………………………………..28

Figure 16. Current Consumption in Different Modes…………………………………………29

Figure 17. Temperature Sensor Transfer Function……………………………………………30

Figure 18. Create New Project………………………………………………………………...32

Figure 19. Workspace…………………………………………………………………………32

Figure 20. Add Files…………………………………………………………………………..33

Figure 21. Choose Device……………………………………………………………………..33

Figure 22. Choose Driver……………………………………………………………...…34

Figure 23. Choose Connection……………………………………………………………...35

Figure 24. Message Window……………………………………………………………….35

Figure 25. General Wireless Sensor Network………………………………………………38

Figure 26. Channelization…………………………………………………………………..41

Figure 27. Simplified Polling Scheme……………………………………………………...43

Figure 28. Wireless Sensor Network with MSP430FG4618/F2013 Experimenter Boards and eZ430RF2500 Wireless Development Tools………………………………………………….44

Figure 29. Polling Scheme with Channelization……………………………………………46

Figure 30. Program Flowchart for Master Board…………………………………………...47

Figure 31. Real World Application of Wireless Sensor Network………………………………49

ABSTRACT

Wireless sensor networks have found a great deal of applications in diverse areas. Recent interest has been focused on low-power feature of the sensor nodes because the power consumption is always an issue for wireless sensor nodes which are supplied from the batteries. The eZ430RF2500 Development Tool and MSP430FG4618/F2013 Experimenter Board from Texas Instruments have integrated MSP430 family of ultralow-power microcontrollers and CC2500 low-power wireless RF transceivers which are suitable for low-power, low-cost wireless applications. In this thesis, the features of these TI devices are explored and a wireless sensor network is implemented with these devices. To implement the routing algorithms we have assumed a hierarchical architecture, where one (slave) experimenter board serves as the access point for a number of sensor nodes. A master board controls the slave boards. Multiple access control protocols are developed using the features of these devices, using channelization and polling. Energy efficiency of the wireless sensor network is also addressed by using the wake on radio feature of the devices. An application of this wireless sensor network is described in this thesis which is to measure temperature of several rooms in a building and display all the temperature data on a PC.

CHAPTER 1 OVERVIEW OF WIRELESS SENSOR NETWORKS

1.1 Introduction

Wireless sensor networks (WSNs) have been utilized all over the world. Significant development of smart sensors has been made in recent years. The responsibilities of those sensors include sensing, measuring, gathering information from the environment, and transmitting the sensed data to each other or to the user by some specific topology. With limited processing and computing resources, the sensors are built smaller and they are cheaper than the traditional sensors [1]. But there still is a project that wireless sensor network technology now focuses on -- low-power and low-cost.

Wireless sensor networks have been used for many applications. The military applications, such as military target tracking and battlefield surveillance, improved the technology of WSNs. Besides that, wireless sensor networks can be used in many consumer and industrial applications, such as machine health monitoring and control, environment and habitat monitoring, biomedical health monitoring, home automation, traffic control, natural disaster relief, and seismic sensing.

1.2 Sensor Nodes

The WSN is built of sensor nodes, from a few to several thousands, where each node is connected to one or several sensors. Sensor nodes have variable sizes and, with the difference in size and cost, they can have some different constraints.

There are four sub-systems in a sensor node: a computing subsystem, a communication subsystem, a sensing subsystem, and a power supply subsystem. Computing subsystem consists of a microprocessor (MCU) which is used to control the sensors by some specific communication protocols. Communication subsystem has a short range radio which is used to communicate with other nodes in the sensor networks. Radios can operate under the Transmit (TX), Receive (RX), IDLE and SLEEP modes. A radio has to be in the transmit mode for the sensor to send data, while it has to be in receive mode if the sensor wants to receive data. With no packets to transmit or receive, the radio stays in IDLE mode. The radio goes to SLEEP mode after a long time that it stays in IDLE mode, which can save the power of the sensor. Sensing subsystem may consist of a number of sensors such as thermal, chemical, optical, biological, and magnetic sensors to measure properties of the environment. Depending on the application and the type of sensors used, actuators, which are applied to actuate the sensing devices and adjust the parameters of the sensor nodes, may be incorporated [1]. Battery is the primary component in the power supply subsystem in most sensor nodes. Depending on the environment where the sensor is deployed, secondary power supply may be added. Solar panel is a common choice which harvests the energy from the sun and works for most of the sensor nodes except the ones working underground.

1.3 Wireless Sensor Networks

A wireless sensor network usually contains tens to thousands of sensor nodes working together to collect data about the environment and share the data they have. The WSNs can be divided into structured WSNs and unstructured WSNs. In an unstructured WSN, sensor nodes may be deployed in an ad-hoc manner. Ad-hoc deployment, which can have a large number of sensor nodes, is for the region which has no infrastructure and usually implemented by tossing the sensor nodes from an airplane or a missile. After deployed, the network is going to perform sensing and reporting functions unattended. However， due to the huge amount of sensor nodes, network maintenance and management is difficult and may cost a lot. Different with ad-hoc manner, sensor nodes are deployed in a pre-planned manner in a structured WSN. With pre-planned manner, sensor nodes are placed at specific locations to provide communication, resulting in lower network maintenance and management cost.

An unstructured WSN with ad-hoc deployment can be usually called wireless ad-hoc network. The wireless ad-hoc network does not have a certain routing scheme because of no infrastructure as mentioned before and the locations of the sensor nodes cannot be determined until the ad-hoc deployment. So every sensor node participates in the routing in wireless ad-hoc network and normally flooding routing technique is applied. Emergent situations such as natural disaster and military conflict are examples suitable for wireless ad-hoc network.

Current wireless sensor networks are deployed basically on land, underground, or underwater. There are different challenges and constraints depending on the environment [1]. According to variable functions and structures, wireless sensor networks can have five types: terrestrial WSN, underground WSN, underwater WSN, multi-media WSN, and mobile WSN.

Terrestrial WSNs are typically composed by hundreds to thousands of sensor nodes deployed either in an ad-hoc or in a pre-planned manner. The sensor node in terrestrial WSN is relatively inexpensive, for it does not need to fulfill the resistance to stress as in underground WSNs or the waterproofness as in underwater WSNs. Only terrestrial WSN with pre-planned deployment is concerned in this thesis.

1.4 Issues in Wireless Sensor Networks

A WSN has its own design and resource constraints due to the limited size and cost of sensor node such as limited energy and storage, short communication range, low bandwidth, and limited processing, which is different with traditional sensor networks. Recent technology in wireless communications and electronics have overcome some of the constraints and enabled the development of sensor nodes which are low-power, low-cost, multifunctional, and small in size. However, three primary problems such as energy efficiency, localization, and routing still hold.

1.4.1 Energy Efficiency

The most important factor to determine the life of a sensor network is the energy consumption [2]. Driven by battery which is limited in power and may not be rechargeable, sensor node is facing the big challenge of conserving energy.

This problem could be found in almost any part of a sensor node and the sensor network. In computing subsystem, MCU's have various operating modes. The power of these modes should be carefully considered to prolong the lifetime of the network. In addition, another kind of consumption of power which cannot be ignored is the changing between these modes. Communication is a major consumer of energy, especially in RX and TX mode. However, radio standing in IDLE mode still consumes as high power as in RX mode. And again, startup in some cases and changing in the radio's operation mode can consume a large amount of power. So it could be a better way to turn the radio into SLEEP mode to minimize the energy consumption than turn it to IDLE mode when it is not transmitting or receiving. And a fast startup transmitter architecture may be applied. A sensor has three sources of power consumption including signal sampling and conversion of physical signals to electrical ones, signal conditioning, and analog-to-digital conversion [3]. Energy can be reduced by using low power sensor in the sensing subsystem. As a vital role in determining sensor node lifetime, battery supplies the power of sensor node. By reducing the current or even turn it off often, the lifetime of the sensor node can be increased effectively.

If energy awareness can be applied in every stage of wireless sensor, the lifetime of the wireless sensor network can be maximized. Moreover, it will be more powerful for wireless sensor networks to have the ability to make tradeoffs between energy consumption and system performance. [3] gives a detailed design of energy-aware sensor network.

1.4.2 Localization

In some sensor networks, particularly in wireless ad-hoc network, sensor nodes are deployed unplanned and the location of each sensor node cannot be predicted. So a problem about determining the location of the nodes needs to be solved and this problem is called localization. Since Global Positioning System (GPS), which comes to mind first, has some strong factors that are not suitable for sensor networks such as the high cost and the limitation of application in the presence of any obstruction like dense foliage, sensor nodes would need to have other means of localization [2].

Various localization techniques can be classified as fine-grained and coarse-grained. Fine-grained includes Timing, Signal strength, Signal pattern matching, and Directionality while Proximity based localization, basing on the technique of recursive trilateration/multilateration, is an example of coarse-grained localization.

1.4.3 Routing

Routing scheme is another factor that can influence the lifetime of wireless sensor networks. Flooding routing techniques, as mentioned above, is an old technique that has some deficiencies and is not suitable for the requirement of the wireless sensor networks. Considering prolonging the lifetime of the wireless sensor network, routing scheme should be designed with the requirement of constraints such as energy, memory, communication bandwidth, and computation capabilities.

There are two types of routing protocols, proactive protocols and reactive protocols. Proactive protocols try to maintain consistent routing between all the nodes while reactive protocols create the routes only when they are needed. SPIN (Sensor Protocols for Information via Negotiation) and Directed Diffusion are two primary routing protocols in wireless sensor networks. SPIN protocols use information description for negotiation among all the sensor nodes before transmission of the data. Directed diffusion is a reactive routing technique in which a node sends out a sensing task for the data it needs throughout the network and then the node which has the data sends it back to the former node [4]. Recently various energy-aware routing protocols, one of which for ad-hoc sensor network is discussed in [5], are proposed in different researches.

1.5 Thesis Contribution and Organization

In this thesis, a wireless sensor network with protocol for power saving is implemented. The eZ430RF2500 development tools and MSP430FG4618/F2013 Experimenter Boards from Texas Instruments are applied in this wireless sensor network.

Here is the organization of the thesis. Chapter 1 mainly reviews the general concepts and some issues of wireless sensor network. Chapter 2 is the introduction of eZ430RF2500 development tools and MSP430FG4618/F2013 Experimenter Boards. Chapter 3 provides the construction and the protocols of the wireless sensor network implemented. Chapter 4 is the summary of the thesis.

CHAPTER 2 TI SENSOR NODE AND BOARD

Since wireless sensor networks can be invaluable in many applications for collecting, processing, and transmitting environmental data, they have gained attention of research in the last few years. Some researches attempt to integrate multifunctional capabilities into a small form factor to enable low-cost sensor nodes in large numbers. Several other projects aim to discuss about efficient hardware/software system architectures, network protocols, and signal processing algorithms for wireless sensor networks. Due to the wide range and a large number of potential applications of wireless sensor networks, many companies engage in developing efficient sensor nodes which have low power, low cost, and long lifetime, including Texas Instruments.

2.1 eZ430-RF2500 Wireless Development Tool

2.1.1 Introduction

The eZ430-RF2500, one of the products from Texas Instruments, is a complete wireless development tool which can be used to develop an entire wireless sensor project. The eZ430-RF2500 consists of MSP430F2274 microcontroller and CC2500 2.4-GHz wireless transceiver, which are the two core components, and all the hardware and software required for them. The eZ430-RF2500T target board is an out-of-the-box wireless system that may be used with the USB debugging interface or with a battery expansion board with the AAA batteries to develop the projects, as shown in Figure 1&2, respectively. The MSP430F2274 microcontroller combines 16-MIPS performance with a 200-ksps 10-bit ADC and 2 op-amps, while the CC2500 multi-channel RF transceiver is designed for low-power wireless applications. The USB debugging interface enables eZ430-RF2500 to connect with a PC to send and receive data using the MSP430 application UART, which is recognized as the application backchannel.

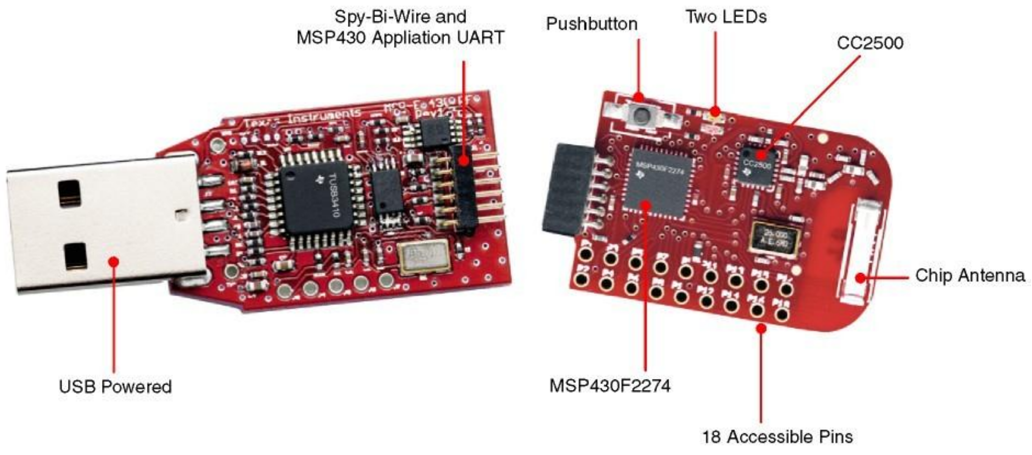


Figure 1. eZ430-RF2500T Board and USB Debugging Interface

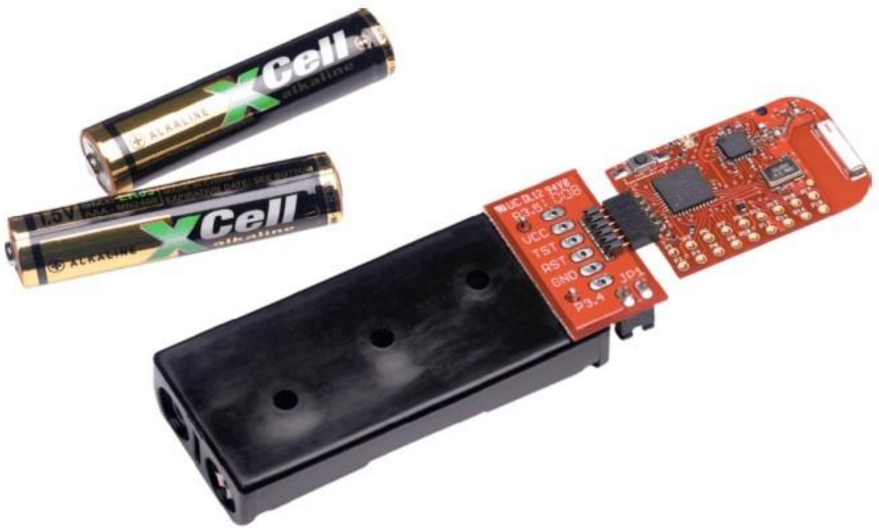


Figure 2. eZ430-RF2500 Battery Board

Besides the USB debugging and programming interface and the highly integrated ultra-low-power MSP430 MCU, the eZ430-RF2500 has some other features such as 21 available development pins, two general-purpose input/output (GPIO) digital I/O pins connected to green and red LEDs for visual feedback, and an interruptible push button for user feedback.

2.1.2 SimpliciTI Network Protocol

The eZ430-RF2500 comes with the SimpliciTI Network Protocol which is a radio-frequency protocol for simple and small radio-frequency (RF) networks [6]. The low-power RF networks usually have some battery-based devices. So the long battery life, low date rate, low duty cycle, and limited number of nodes are the resource requirements. With SimpliciTI Network Protocol, the network can be implemented with minimal microcontroller resources requirement, which results in lower system cost for low-power RF networks. Based on this advantage, a wide range of low-power applications can utilize the SimpliciTI Network Protocol. Alarm and security detectors, gas and water meter reading, and home automation are some example of wireless sensor networks with SimpliciTI Network Protocol. Texas Instruments has included a simple temperature sensor network application, which is the eZ430-RF2500 wireless sensor monitor, with SimpliciTI Network Protocol to provide a starting introduction to develop a wireless application.

2.1.3 eZ430-RF2500 Wireless Sensor Monitor

The wireless temperature sensor networks firmware is preloaded in the eZ430-RF2500 tool. However, two eZ430-RF2500T target boards, which are identical in hardware, have got distinct firmwares to play different rolls in the Wireless Sensor Monitor network. The one used with the USB debugging interface acts as a SimpliciTI Access Point (AP) while the other one with a battery expansion board acts as a SimpliciTI End Device (ED). They work together to make a star topology to measure temperature, which can be shown on PC when the AP is connected to the PC.

The AP measures and transmits the temperature data to the PC once a second and spends the left time of the period in collecting the packets from the EDs in the network and the query messages for any new EDs attempting to join the network. UART backchannel is the way APs transmit any data to the PC, including the measurements of the EDs. The two LEDs on the target board of AP work as indicators of network. The red one flashes when there is a transmission of AP's measurement and the green one tells a receipt of a measurement packet from an ED in the network [7].

The ED begins to search for an AP around it by sending a query message repeatedly after startup and joins in the network after an AP receives the query and responses to the ED to build a link between them. Then the ED spends most of the time in low power mode 3 (LPM3), which is one of the power modes of MSP430F2274 microcontroller and will be discussed later in this thesis, and wake up once a second to measure the temperature and the voltage of the battery and send the result within a packet to the AP. The AP sends it to the PC through the UART once it gets the packets. The two LEDs work in a different way as the ones in the APs. Both LEDs are flashing when ED is searching for an AP. Once AP is discovered, the ED attempts to establish a network link. The red LED of ED does not stop flashing until ED links up with AP and joins in the network. After that, both LEDs turn off with the ED entering into LPM3. Only green LED will be toggled once a second indicating that the ED has woken up to measure the temperature and voltage.

All the date coming from AP can be shown on PC with PC Sensor Monitor Visualizer. In the PC Sensor Monitor Visualizer, the center node indicates Access Point and the nodes around it are the End Devices, as shown in Figure 3. The PC application displays the temperature of both the End Devices and Access Point and changes the display immediately after receiving any date from the

AP. In addition, the PC application is capable of simulating distance between AP and EDs. When moving the EDs, PC will show a similar move between the node indicating the moving ED and the central AP. Up to eight End Devices can be connected to an AP and shown on PC.

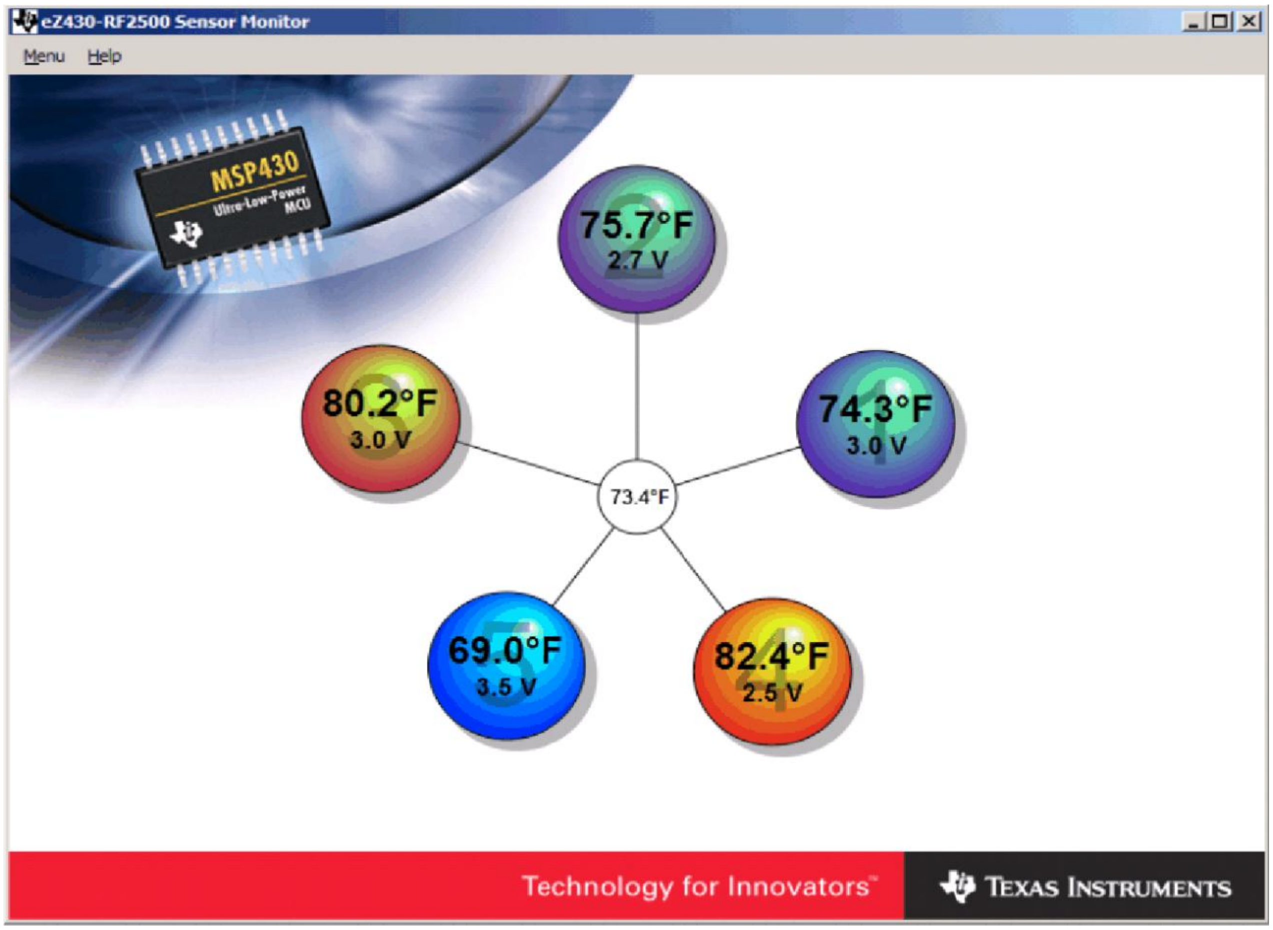


Figure 3. eZ430-RF2500 Wireless Sensor Monitor

A restore tool of the temperature sensor networks firmware can be used to program the eZ430-RF2500 with the original code of the firmware and can be found at the following website:

http://processors.wiki.ti.com/index.php?title=EZ430-RF2500

Also, the sensor monitor software can be found at http://www.ti.com/lit/zip/slac139

2.2 MSP430FG4618/F2013 Experimenter Board

The MSP430FG4618/F2013 Experimenter Board, another product of Texas Instruments, is a versatile development board that can be used in many applications. Two microcontrollers, MSP430FG4618 and MSP430F2013, reside on the board and can be used to manage the whole board. A figure of MSP430FG4618/F2013 Experimenter Board is shown below.

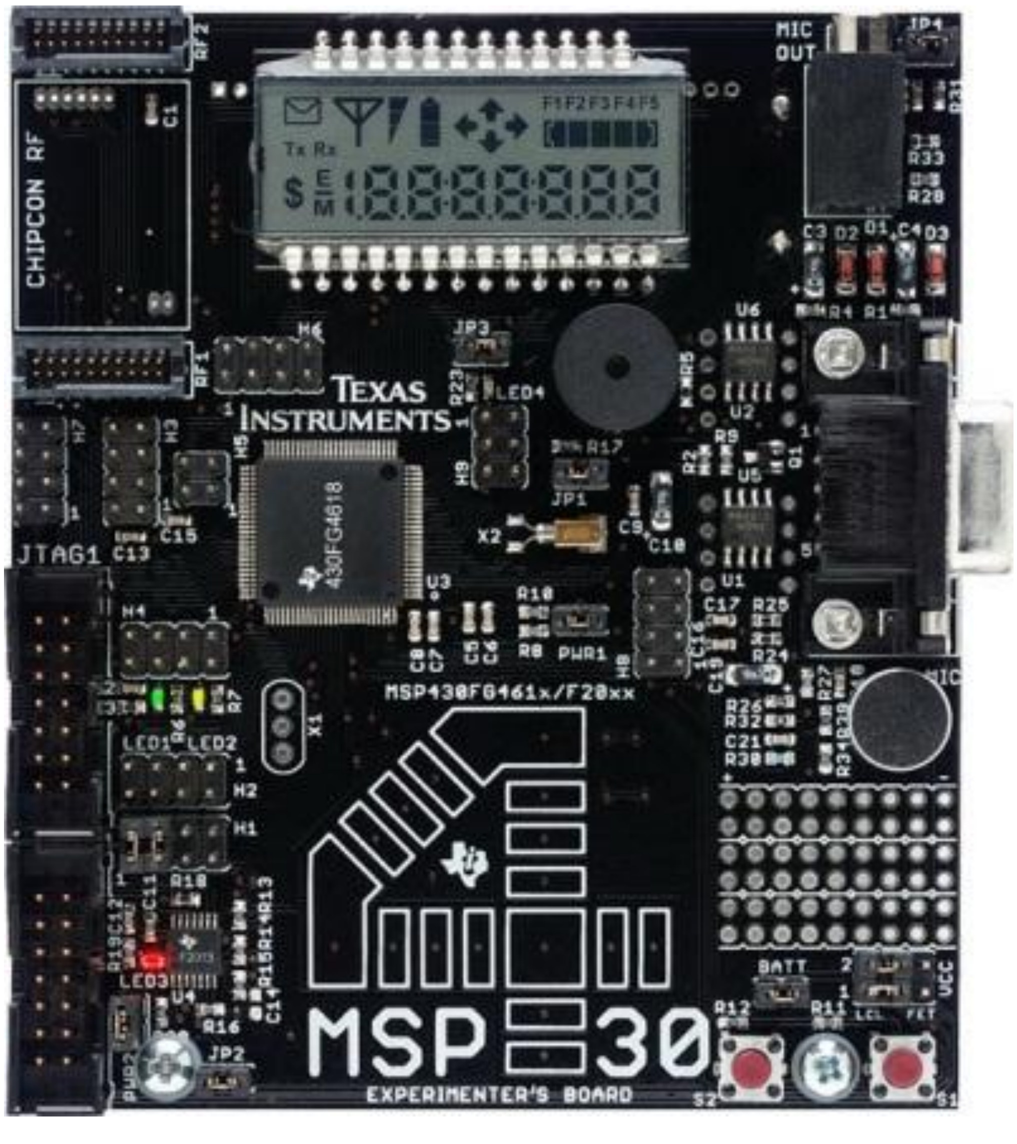


Figure 4. MSP430FG4618/F2013 Experimenter Board

The versatility of the board, besides the two MCU's working together, comes from various interfaces to a number of on- and off- board components. The one that is important to this thesis is Chipcon Wireless Evaluation Module which makes the wireless communication possible. The Chipcon Wireless Evaluation Module supports any CCxxxxEMK board from Texas Instruments including CC2500EMK. CC2500EMK is a small board integrated with CC2500 multi-channel wireless RF transceiver to achieve the wireless application along with the antenna. Figure of CC2500EMK is shown below.

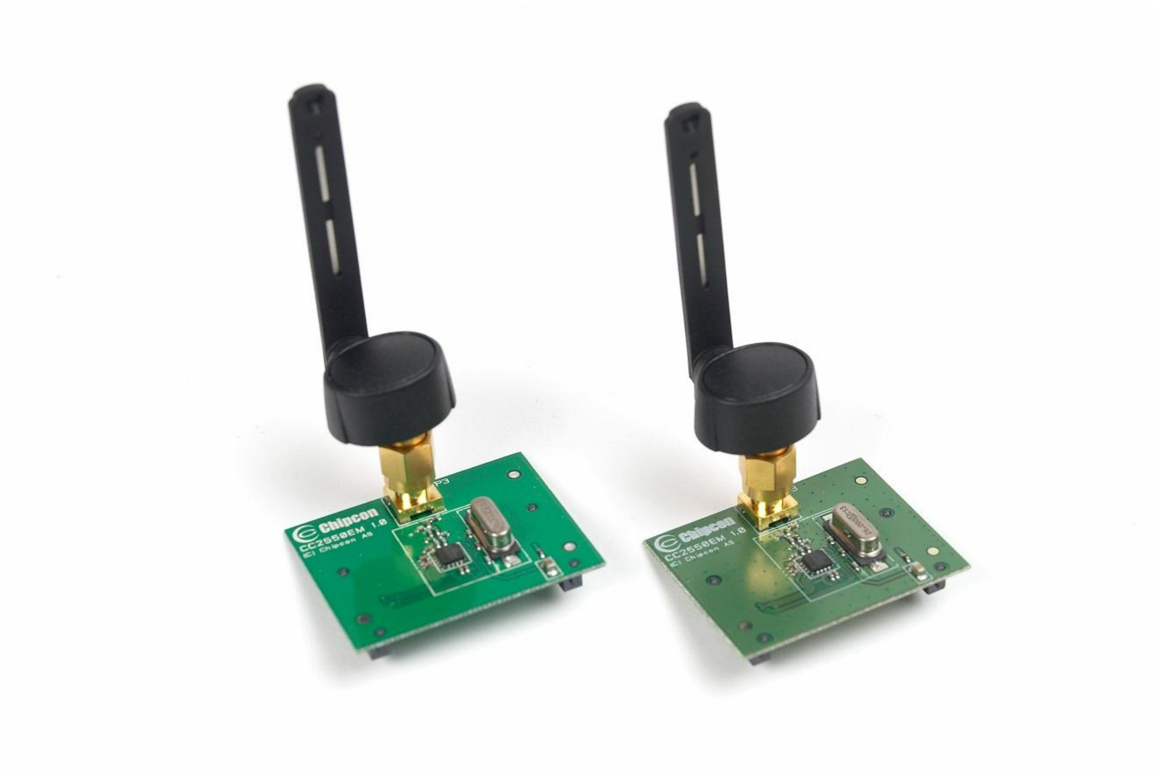


Figure 5. CC2500EMK

Other peripherals and interfaces include 4-MUX LCD display, two momentary-on push buttons, four light emitting diodes (LEDs), buzzer, microphone, audio output jack, UART connection, and capacitive touch pad [8]. The block diagram of MSP430FG4618/F2013 Experimenter Board is shown in Figure 6.

Joint Test Action Group (JTAG) is used to debug the software and test the function of the board. There are two JTAG interfaces, each one supporting one of the microcontrollers. With an

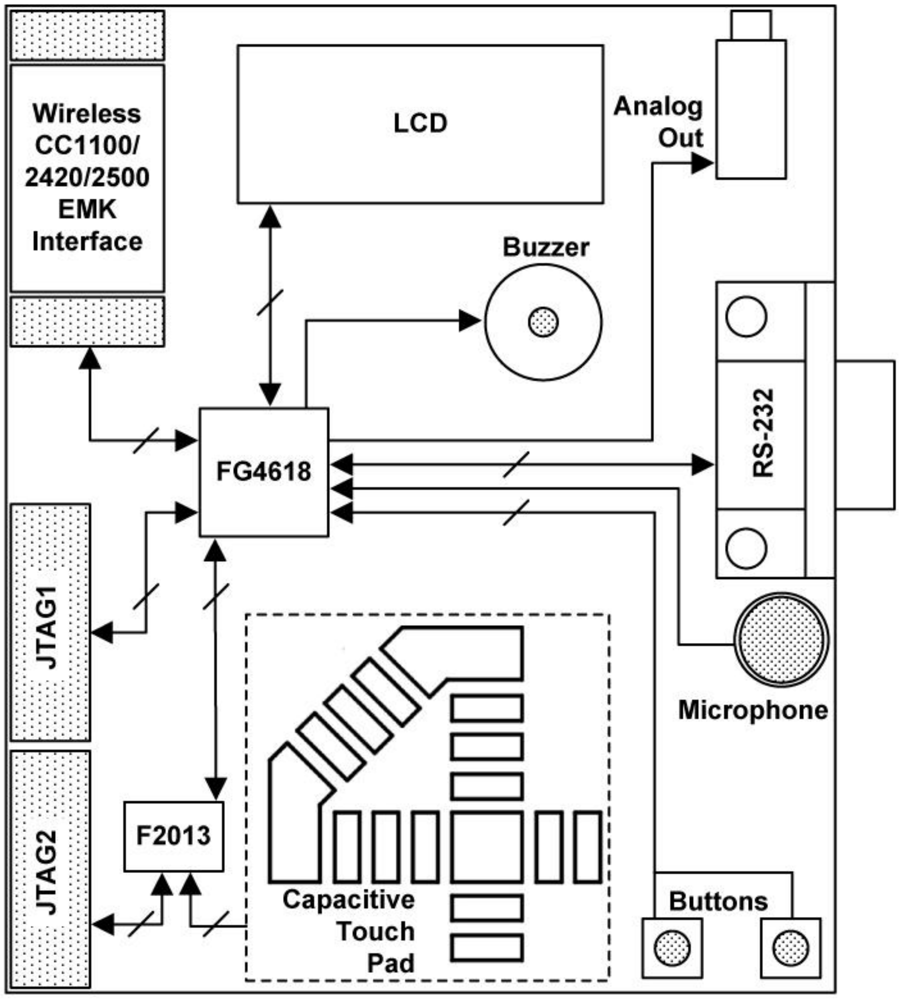


Figure 6. Experimenter Board Block Diagram

MSP430 Flash Emulation Tool (MSP-FET430UIF), as shown in Figure 7, the microcontrollers can download the code from PC.

Besides the AAA batteries, MSP-FET430UIF can also supply the power to the board. Which source should be used is what some jumpers of the board are responsible to configure, as shown in Figure 8. VCC_1, VCC_2, BATT, PWR1, and PWR2 are the five jumpers to determine the power supplier.

Figure 7 MSP-FET430UIF

PWR1 and PWR2 supply MSP430FG4618 and MSP320F2013, respectively. On-board battery is chosen when BATT jumper is connected. The two rows of 3-pin header jumper, VCC_1 and

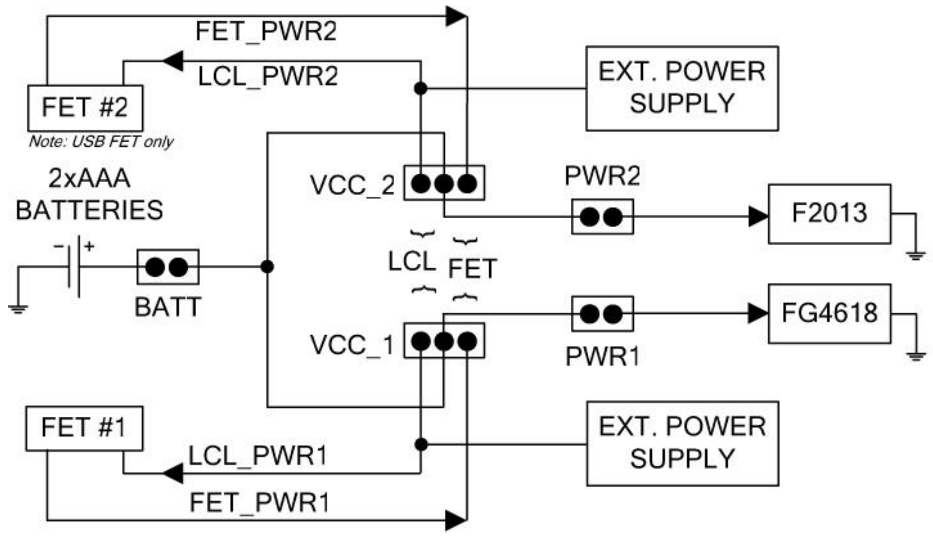


Figure 8. Jumper Settings for Power Selection

VCC_2, choose the connection type between MCUs and the MSP-FET430UIF interfaces. VCC_1 is for MSP430FG4618 while VCC_2 is for MSP430F2013. To select JTAG FET to be the power supplier, the rightmost two pins should be connected by the jumper, whichever row it is. Some other jumpers can be used to mute the buzzer, disconnect the LEDs, or enable some interfaces when those peripherals are not in use, which saves power of the system.

Since the Chipcon Wireless Evaluation Module is connected with MSP430FG4618, which is shown in the diagram, only function of this microcontroller will be discussed in this thesis.

2.3 CC2500 Wireless RF Transceiver

2.3.1 Introduction

The CC2500 RF transceiver, used both in MSP430FG4618/F2013 experimenter board and eZ430-RF2500 wireless development tool, plays a great roll in the wireless application. It works in 2400-2483.5 MHz frequency range and has a data rate which is typically 250 kBaud and configurable to be from 1.2 to 500 kBaud. With the typical current consumption of 13.3mA in receiving mode, 11.1mA in transmitting mode, and 400nA in sleep mode, the low-power feature of CC2500 is obviously shown. In addition, Wake-On-Radio (WOR) and fast startup time which is generally 240 µs from sleep mode to receiving or transmitting mode contribute to the low-power feature too.

Besides the Wake-On-Radio mentioned above, CC2500 also supports packet handling, data buffering, a few modulation schemes, link quality indication, and some other functions. Two 64-byte FIFOs, for transmitting and receiving, respectively, are included in CC2500 and can be controlled via an SPI interface. Usually, CC2500 works with a microprocessor, which is MSP430FG4618 or MSP430F2274 in this thesis, and some other components.

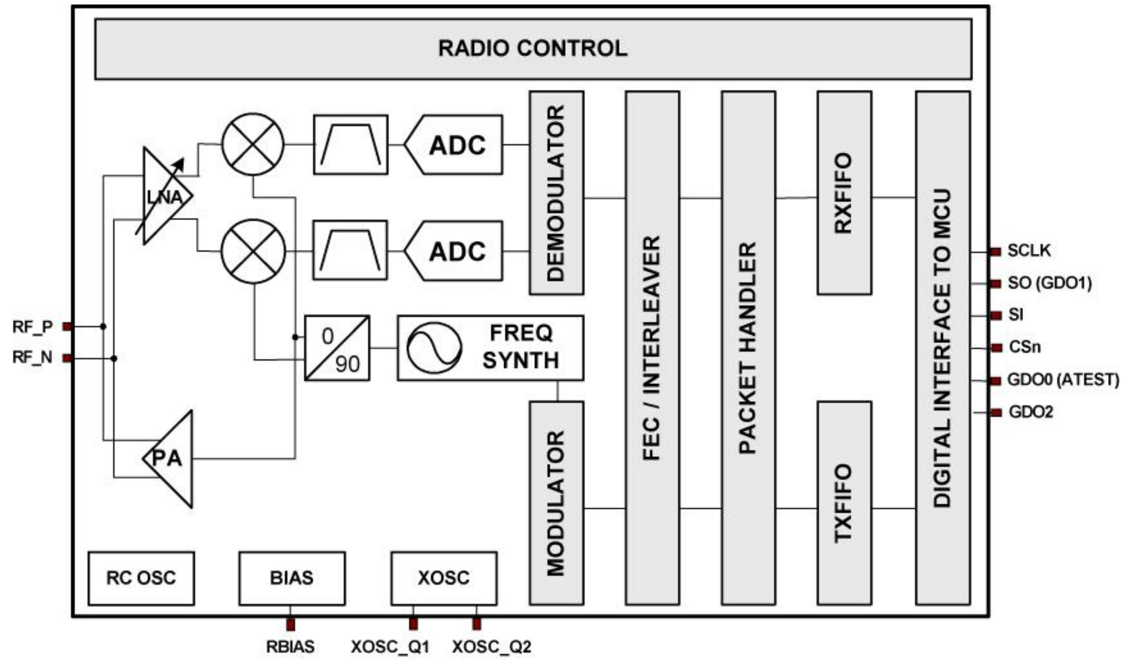


Figure 9. CC2500 Simplified Block Diagram

Above is a figure of simplified block diagram of CC2500. The low-IF receiver part leads the received RF signal through a low-noise amplifier (LNA), the process of down-converting to the intermediate frequency (IF), channel filters, and ADCs to be digitized [9]. The transmitter part relies on the frequency synthesizer with the frequency generated by the crystal oscillator. SO, SI, CSn, and SCLK compose the 4-wire SPI serial interface to configure the CC2500.

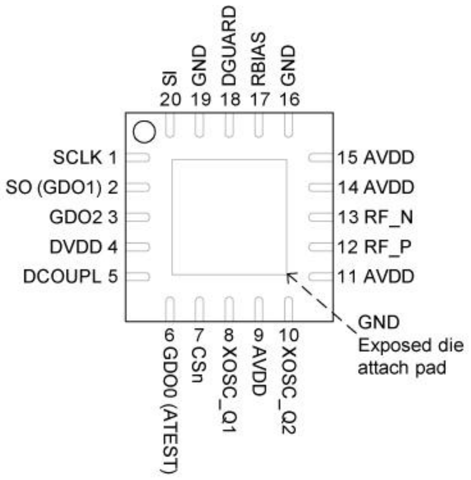


Figure 10. Pin Configuration of CC2500(QLP 4×4 mm Package, 20 pins)

The pin configuration of CC2500 is shown in Figure 10, and a typical application [9] can be achieved by adding several external components, shown in Figure 11. The functions of the components are listed in Table 1. The CC2500EMK evaluation module, discussed in former section, is based on this circuit with a set of certain value for the external components.

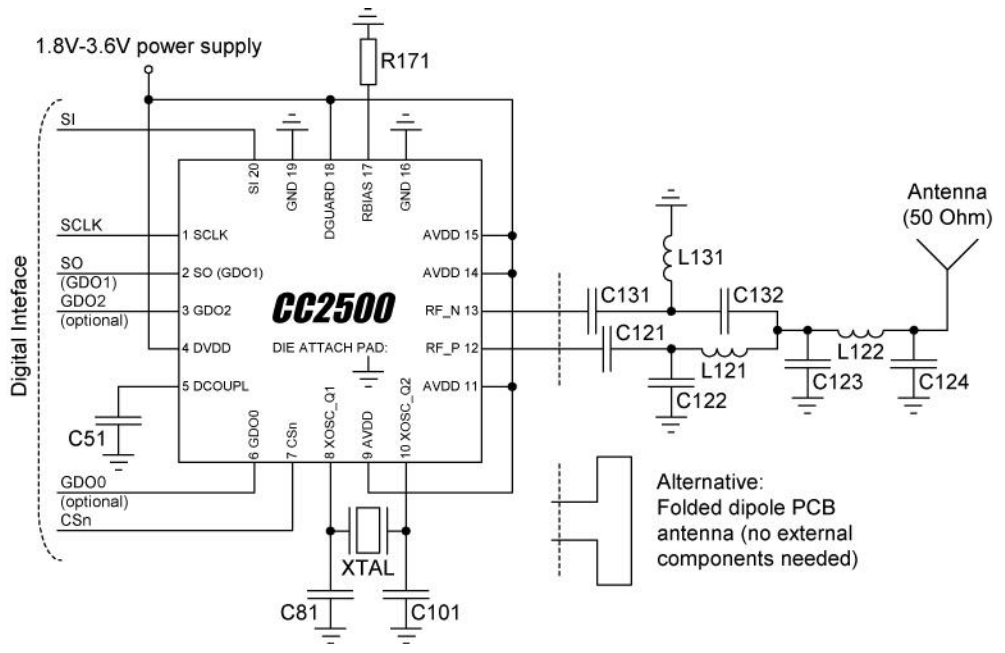


Figure 11. Typical Application of CC2500 and Evaluation Circuit

Table 1. External Components

Component	Description
C51	Decoupling capacitor for on-chip voltage regulator to digital part
C81/C101	Crystal loading capacitors, see Section 26 on page 50 for details
C121/C131	RF balun DC blocking capacitors
C122/C132	RF balun/matching capacitors
C123/C124	RF LC filter/matching capacitors
L121/L131	RF balun/matching inductors (inexpensive multi-layer type)
L122	RF LC filter inductor (inexpensive multi-layer type)
R171	Resistor for internal bias current reference
XTAL	26-27 MHz crystal, see Section 26 on page 50 for details

2.3.2 Configuration Registers

Configuration of CC2500 is achieved by programming the configuration registers including 47 normal 8-bit registers, 12 status registers, and 13 command strobes, shown in Table 2.

Table 2. Configuration Registers

	Write		Read	
	Single	Burst	Single	Burst
0x00	IOCFG2			
0x01	IOCFG1			
0x02	IOCFG0			
0x03	FIFOTHR			
0x04	SYNC1			
0x05	SYNC0			
0x06	PKTLEN			
0x07	PKTCTRL1			
0x08	PKTCTRL0			
0x09	ADDR			
0x0A	CHANNR			
0x0B	FSCTRL1			
0x0C	FSCTRL0			
0x0D	FREQ2			
0x0E	FREQ1			
0x0F	FREQ0			
0x10	MDMCFG4			
0x11	MDMCFG3			
0x12	MDMCFG2			
0x13	MDMCFG1			
0x14	MDMCFG0			
0x15	DEVIATN			
0x16	MCSM2			
0x17	MCSM1			
0x18	MCSM0			
0x19	FOCCFG			
0x1A	BSCFG			
0x1B	AGCCTRL2			
0x1C	AGCCTRL1			
0x1D	AGCCTRL0			
0x1E	WOREVT1			
0x1F	WOREVT0			
0x20	WORCTRL			

(TABLE 2 Continued)

0x21	FREND1			
0x22	FREND0			
0x23	FSCAL3			
0x24	FSCAL2			
0x25	FSCAL1			
0x26	FSCAL0			
0x27	RCCTRL1			
0x28	RCCTRL0			
0x29	FSTEST			
0x2A	PTEST			
0x2B	AGCTEST			
0x2C	TEST2			
0x2D	TEST1			
0x2E	TEST0			
0x2F				
0x30	SRES		SRES	PARTNUM
0x31	SFSTXON		SFSTXON	VERSION
0x32	SXOFF		SXOFF	FREQEST
0x33	SCAL		SCAL	LQI
0x34	SRX		SRX	RSSI
0x35	STX		STX	MARCSTATE
0x36	SIDLE		SIDLE	WORTIME1
0x37				WORTIME0
0x38	SWOR		SWOR	PKTSTATUS
0x39	SPWD		SPWD	VCO_VC_DAC
0x3A	SFRX		SFRX	TXBYTES
0x3B	SFTX		SFTX	RXBYTES
0x3C	SWORRST		SWORRST	RCCTRL1_STATUS
0x3D	SNOP		SNOP	RCCTRL0_STATUS
0x3E	PATABLE	PATABLE	PATABLE	PATABLE
0x3F	TX FIFO	TX FIFO	RX FIFO	RX FIFO

The normal configuration registers are mostly used for test purposes. The command strobes will start some internal sequences to change the mode or status of CC2500. And the status registers, which are read only, show the status of CC2500. The two FIFOs can be accessed by 0x3F register with writing to TX FIFO and reading from RX FIFO. The 0x3E PATABLE register is used to configure the output power of CC2500.

2.3.3 Transaction through SPI

The CSn pin from the 4-wire SPI interfaces must be in low status to start a transaction. A header byte is generated through SI pin at the beginning of any transactions. The header byte consists of an R/W bit indicating whether it is Read or Write, a burst access bit telling whether it is single access or burst access, and a 6-bit address(A_5-A_0) showing the address of the register it attempts to work on. In burst access, the address bit should be the first byte of a number of continuous bytes. Data bytes, following header byte, are sent on SI pin if it is in writing mode with R/W bit set to 1 and on SO pin if in reading mode with R/W set to 0. Command strobes are accessed without any data bytes except header byte.

When the address bit of the header byte is 0x3F, CC2500 aims to read from the RX FIFO or write to the TX FIFO, determined by the R/W bit of the header byte. So the CC2500 is half-duplex which means a transceiver supports both transmission and reception but only in one direction any given time.

A chip status byte is created by CC2500 during the transaction. In the writing mode, a status byte is sent on SO pin each time a header byte or a data byte is sent on SI pin. In the reading mode, there is a status byte on SO pin during the transmission of header byte on SI pin. Then the data bytes will be transmitted on SI pin. So there are no status bytes during the transmission of data bytes because the status byte can be only sent on SI pin.

A summary of status byte is shown in Table 3. Low value of CHIP_RDYn indicates that the crystal oscillator is running. 3-bit STATE shows the state of the chip. The 4-bit FIFO_BYTES_AVAILABLE reflects the number of bytes to be read in RX FIFO in reading mode or the number of free bytes in TX FIFO in writing mode.

Table 3: Status Byte

Bits	Name	Description	
7	CHIP_RDYn	Stays high until power and crystal have stabilized. Should always be low when using the SPI interface.	
6:4	STATE[2:0]	Indicates the current main state machine mode	
		Val ue	State
		000	IDLE
		001	RX
		010	TX
		011	FSTXON
		100	CALIBRATE
		101	SETTLING
		110	RXFIFO_OVERFLOW
		111	TXFIFO_UNDERFLOW
3:0	FIFO_BYTES_AVAILABLE[3:0]	The number of bytes available in the RX FIFO or free bytes in the TX FIFO	

2.3.4 Packet Format

The data format can have several parts such as preamble bytes, synchronization word, packet length byte, address byte, data (payload), and 2-byte CRC, as shown in Figure 12. The preamble bytes, synchronization word, and the CRC are inserted to the packet in transmit mode by the packet handler [9]. A transmission starts by sending the preamble bytes, the minimum number of which is programmable and is typically 4 bytes for 250 kBaud data rate. More bytes may be needed for higher data rate. After the minimum preamble bytes, the synchronization word and the data in TX FIFO will be sent, except when there are no bytes in TX FIFO, in which case the modulator will keep sending preamble bytes until the first byte enters into TX FIFO. The synchronization word is responsible for the byte synchronization of the packet with typically 4 bytes. The CRC (Cyclic Redundancy Check) is used to check the possible change about the payload in receiving mode. For this kind of format, several steps of de-construct by packet handling in receive mode may include preamble detection, synchronization word detection, CRC computing and CRC check, one byte address check, packet length check, and decoding.

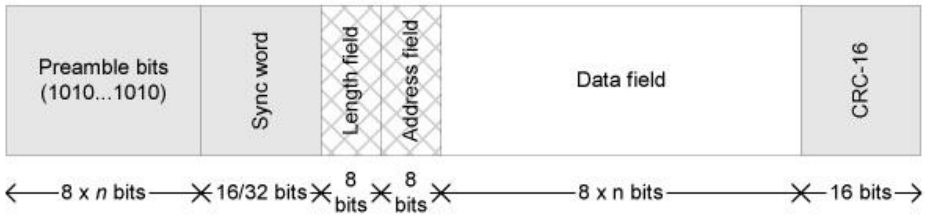


Figure 12. Packet Format

CC2500 support 3 modes of packet transmission and reception which can be chosen by setting the 2-bit LENGTH_CONFIG in MDMCFG register. Fixed packet length mode is issued when MDMCFG .LENGTH_CONFIG is set to 0, with packet length set by the PKTLEN register. The PKTLEN register stores the maximum packet length in variable packet length mode with LENGTH_CONFIG set to 1 and the extra bytes beyond the packet length will be discarded. Note that these two modes only satisfy the packet with length up to 256 bytes. Infinite packet length mode will be used for longer packets. In this mode, LENGTH_CONFIG is set as 2 and the process never stops until it is turned off manually.

2.3.5 Radio Control

Figure 13 shows the complete radio mode control where TX, RX, IDLE, and SLEEP are the four most significant modes. It can be configured to change the radio mode by issuing some command strobes. Several internal events may also result in the mode change. For example, radio will go into mode TXFIFO_UNDERFLOW when TX FIFO underflows.

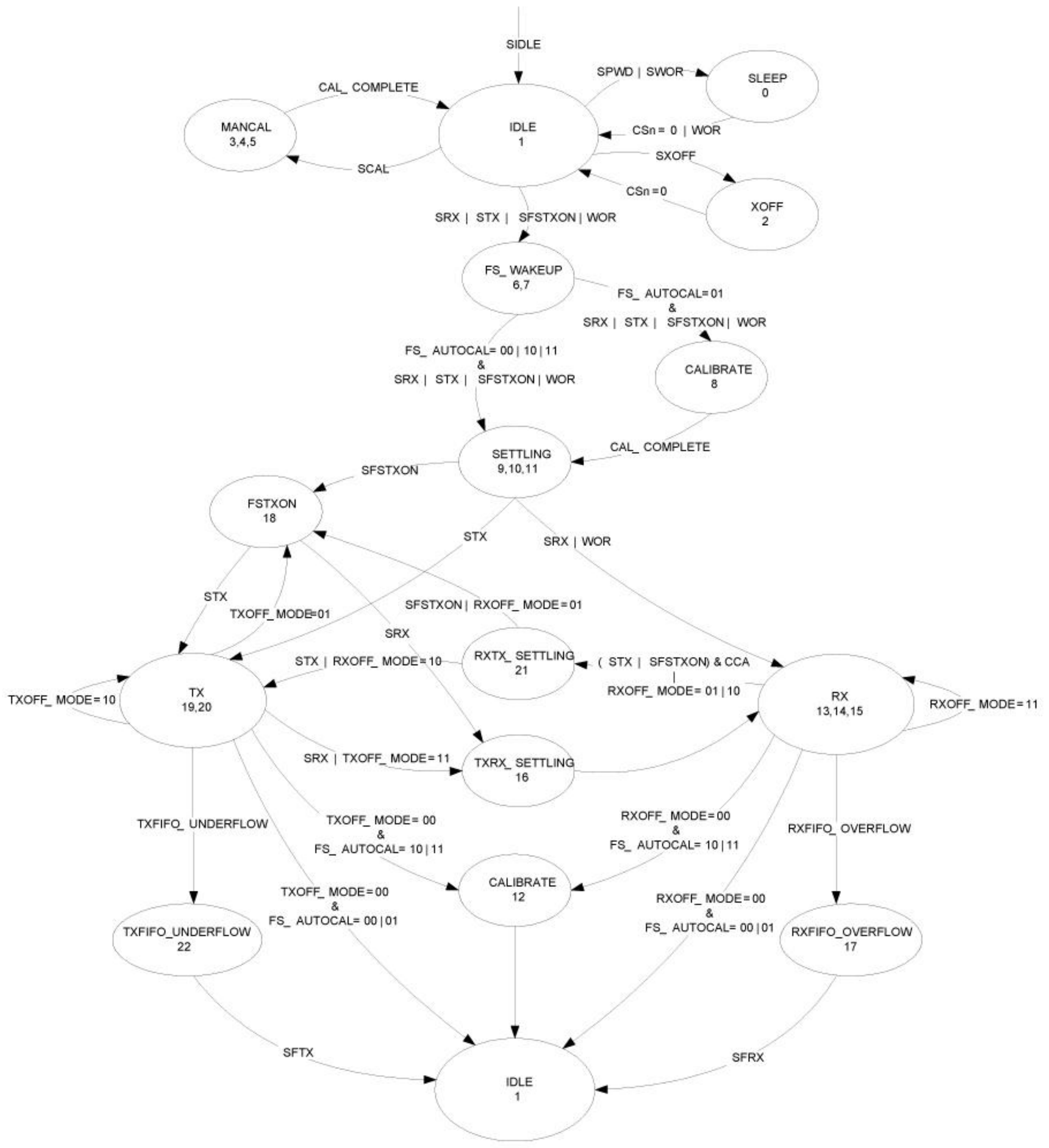


Figure 13. Radio Mode Control State Diagram

One of the two possible types of reset, automatic power-on reset (POR) and manual reset, will be operated after the power is turned on. The registers go to default value and the radio goes to IDLE mode after whichever reset is completed. Receive(RX) and transmit(TX) mode are the two active modes of the radio. After the chip enters TX mode by issuing the STX strobe, it stays in TX state until the transmission of the packet has been done. Then the radio goes to check the 2-

bit TXOFF_MODE in MCSM1 register to determine which state it should enter. Table 4 lists the four states of the MCSM1. TXOFF_MODE register. RX mode can be activated by SRX strobe. Similarly, the chip stays in RX until the whole packet has been received and then check the MCSM1.RXOFF_MODE register which has the same states as Table 4 shows. TX and RX can be manually changed by SRX and STX strob. In this case, the radio changes to the other mode by terminating the current job.

Table 4. TXOFF_MODE and RXOFF_MODE states

Bit Value	State	Decription
00	IDLE	Enter IDLE state
01	FSTXON	Frequency synthesizer on and ready at the TX frequency
10	TX	Start sending preamble
11	RX	Start search for a new packet

RX mode can be also activated by Wake On Radio(WOR). Wake On Radio allows for the CC2500 to stay in SLEEP mode for most of the time and wake up periodically into RX mode to search for packets [10]. This saves the system cost because the chip consumes relatively lower power in SLEEP mode. The period and the time the transceiver stays in RX mode each period can be programmed. Wake On Radio is issued by SWOR strobe and can be ended by setting strobes to enter IDLE mode.

2.3.6 Frequency Programming

The carrier frequency can be configured by programming the related register. In CC2500, the start carrier frequency is calculated by

$$f_{carrier} = \frac{f_{xosc}}{2^{16}} \cdot FREQ$$

where FREQ is a 24-bit register and f_{xosc} is the crystal frequency. The channel spacing is given by

$$\Delta f_{channel} = \frac{f_{xosc}}{2^{18}} \cdot \left(256 + CHANSPC\_M\right) \cdot 2^{CHANSPC\_E}$$

with 2-bit MDMCFG0.CHANSPC_E and 8-bit MDMCFG0.CHANSPC_M register to be programmed. The 8-bit register CHANNR.CHAN dertermine which carrier frequency will be used. This indicates that CC2500 can have up to 256 carrier frnquencies based on the start carrier frequency and the channel spacing. The final formula of the carrier frequencies is as follows:

$$f_{carrier} = \frac{f_{xosc}}{2^{16}} \cdot \left(FREQ + CHAN \cdot \left(\left(256 + CHANSPC\_M\right) \cdot 2^{CHANSPC\_E}\right)\right)$$

The receiver channel filter bandwidth and the IF frequency can also be programmed by the next formula.

$$BW_{CHANNEL} = \frac{f_{XOSC}}{8 \cdot \left(4 + CHANBW\_M\right) \cdot 2^{CHANBW\_E}}\text{, where}$$

CHANBW_M and CHANBW_E are 2-bit registers in MDMCFG4 to configure the receiver channel filter bandwidth, and

$$f_{IF} = \frac{f_{XOSC}}{2^{10}} \cdot FREQ\_IF\text{, where}$$

FREQ_IF is the 5-bit register in FSCTRL1 to configure the IF frequency.

2.3.7 Modulation Format and Output Power Programming

CC2500 supports 4 types of modulation formats: 2-FSK, GFSK, MSK, and OOK. The MDMCFG2.MOD_FORMAT register determines the specific format.

PATABLE register is used to program the output power of the device. In 2-FSK, GFSK, and MSK format, only index 0 of PATABLE is occupied. Comparing to that, index 0 and 1 are needed for OOK format because OOK has two kinds of logic probability (on and off). Figure 14 and Table 5 show the PATABLE register and the different power levels.

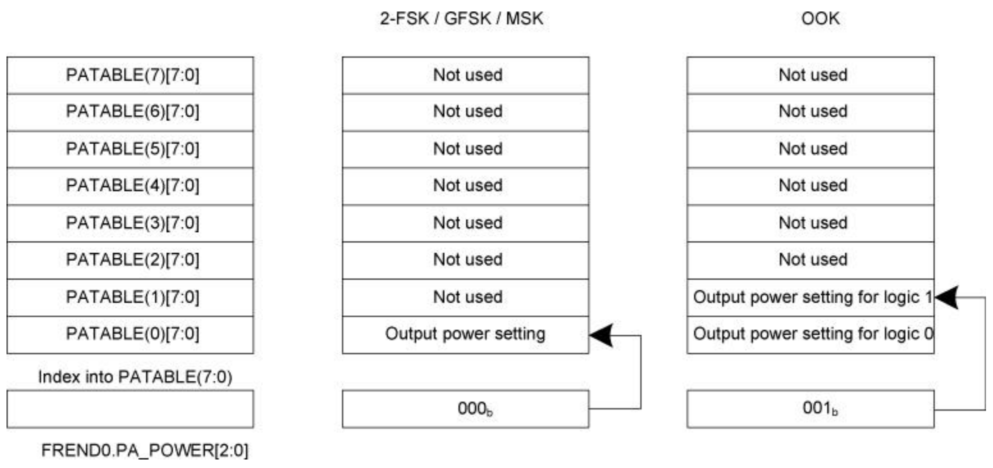


Figure 14. Patable

Table 5 Output Power Levels

Output Power, Typical, +25 ℃, 3.0 V [dBm]	PATABLE Value	Current Consumption, Typical [mA]
(–55 or less)	0x00	8.4
-30	0x50	9.9
-28	0x44	9.7
-26	0xC0	10.2
-24	0x84	10.1
-22	0x81	10.0
-20	0x46	10.1
-18	0x93	11.7
-16	0x55	10.8
-14	0x8D	12.2
-12	0xC6	11.1
-10	0x97	12.2
-8	0x6E	14.1
-6	0x7F	15.0
-4	0xA9	16.2
-2	0xBB	17.7

(TABLE 5 Continued)

0	0xFE	21.2
+1	0xFF	21.5

In addition, 3-bit FREND0.PA_POWER register should be set to 1 when in OOK format and 0 for other three formats.

2.4 Microcontrollers

MSP430F2274 and MSP430FG4618 are two of the MSP430 series microcontroller from Texas Instruments. MSP430 is a series of ultralow-power microcontroller with variable peripherals. With 16-bit RISC CPU, 16-bit registers, and other variable peripherals, MSP430 could be suitable for many different applications. Even though MSP430FG4618 consists of more devices and is more powerful than MSP430F2274, both of them have the feature of low-power mode and temperature sensor, which contributes most to the wireless sensor network built in this thesis. The figure of MSP430F2274 is shown below.

Figure 15. MSP430F2274

There are three clock signals at the basic clock module of MSP430 family. Master clock (MCLK) is used by CPU and system. Auxiliary clock (ACLK) and Sub-main clock (SMCLK) are used by some individual peripheral modules. Master clock is sourced by a high-speed digitally controlled oscillator (DCO). Normally all the clocks are active, which is called the Active Mode (AM). The ultralow-power feature of the MSP430 family comes from the five low-power modes (LPM0-LPM4) achieved by deactivating selectively some of the clocks. Figure 16 shows the current consumption of MSP430F2274 in different power modes.

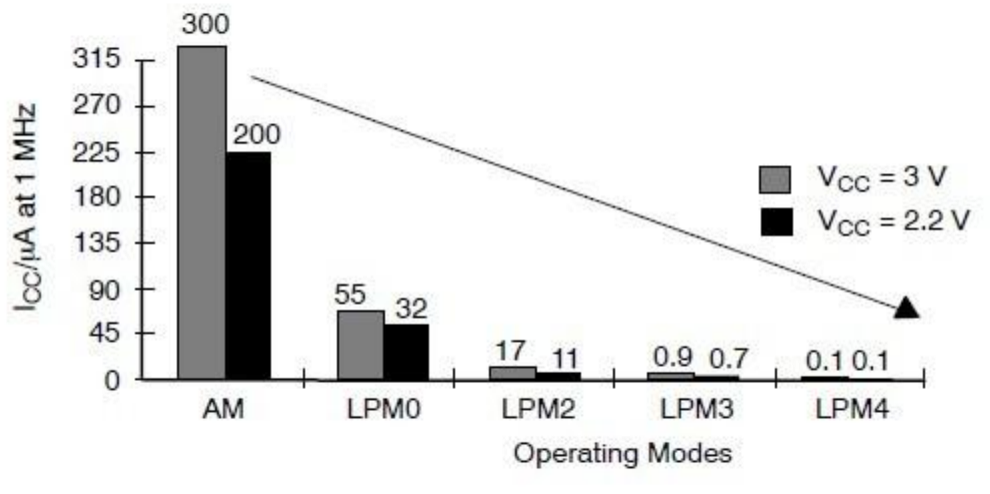


Figure 16. Current Consumption in Different Modes

The modes can be determined by CPUOFF, OSCOFF, SCG0, and SCG1, four bits in the status register of the MCU. Table 6 shows the value of the bits and the status of CPU and clocks in each mode. Changing between the modes is done by setting the status register during the interrupt service routine. The MCU will go to the mode indicated by the value of the four bits after the interrupt service routine. If the status register has no change during the interrupt service routine, the MCU will return to the previous mode before.

The most useful application to save the system cost is to extend the time in LPM3 because power consumption is less than 2 μA and all of the interrupts are active with only the auxiliary clock working. The wireless temperature sensor networks firmware, introduced in section 2.1, has applied this feature.

Table 6 Power Mode and Status of Clocks and CPU

SCG1	SCG0	OSCOFF	CPUOFF	MODE	CPU and Clocks Status
0	0	0	0	Active	CPU is active. All enabled clocks are active
0	0	0	1	LPM0	CPU, MCLK are disabled. SMCLK, ACLK are active.
0	1	0	1	LPM1	CPU, MCLK are disabled. DCO and DC generator are disabled if the DCO is not used for SMCLK. ACLK is active
1	0	0	1	LPM2	CPU, MCLK, SMCLK, DCO are disabled. DC generator remains enabled. ACLK is active.
1	1	0	1	LPM3	CPU, MCLK, SMCLK, DCO are disabled. DC generator disabled. ACLK is active.
1	1	1	1	LPM4	CPU and all clocks are disabled.

As mentioned previously the MSP430 family of microcontrollers have integrated temperature sensors. The sensed temperature is converted to reference voltage using the formula below:

$$V_{TEMP} = 0.00355(TEMP_C) + 0.986$$

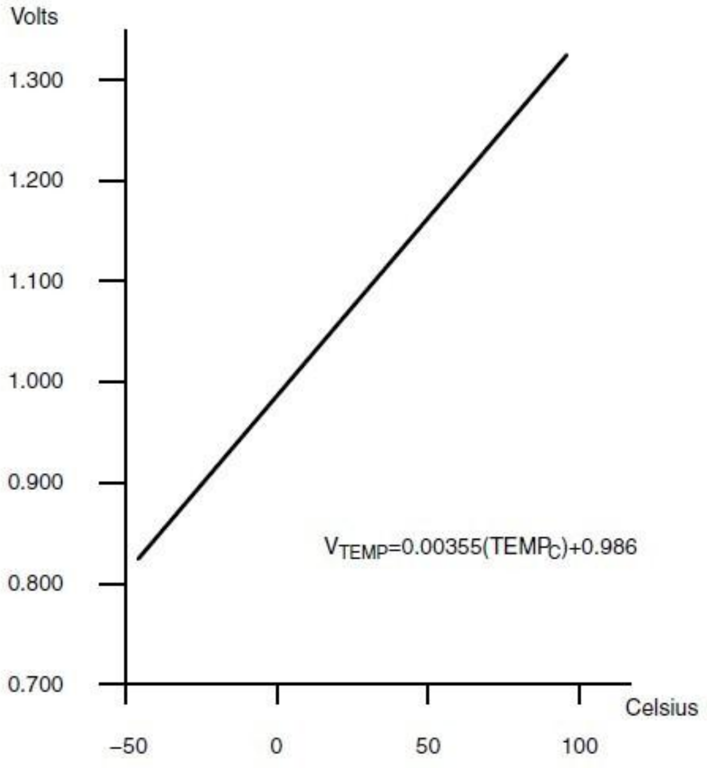


Figure 17. Temperature Sensor Transfer Function

The relation between the voltage and the temperature can be easily seen in the Figure 17. Then the voltage is digitized after going through the analog-to-digital-convertor (ADC) and stored in a register of the MCU. By reading the register, the temperature can be shown or be transmitted to other devices by a connected CC2500.

2.5 IAR Embedded Workbench Kickstart

IAR Embedded Workbench is a very powerful Integrated Development Environment (IDE) used to build and debug the embedded applications with assembler, C, and C++. The IAR Embedded Workbench Kickstart, which is a code-size limited version of IAR Embedded Workbench and consists of a code-size limited C-Compiler, an unlimited assembler, a simulator, and an FET debugger, is used for MSP430 family microcontroller [13].

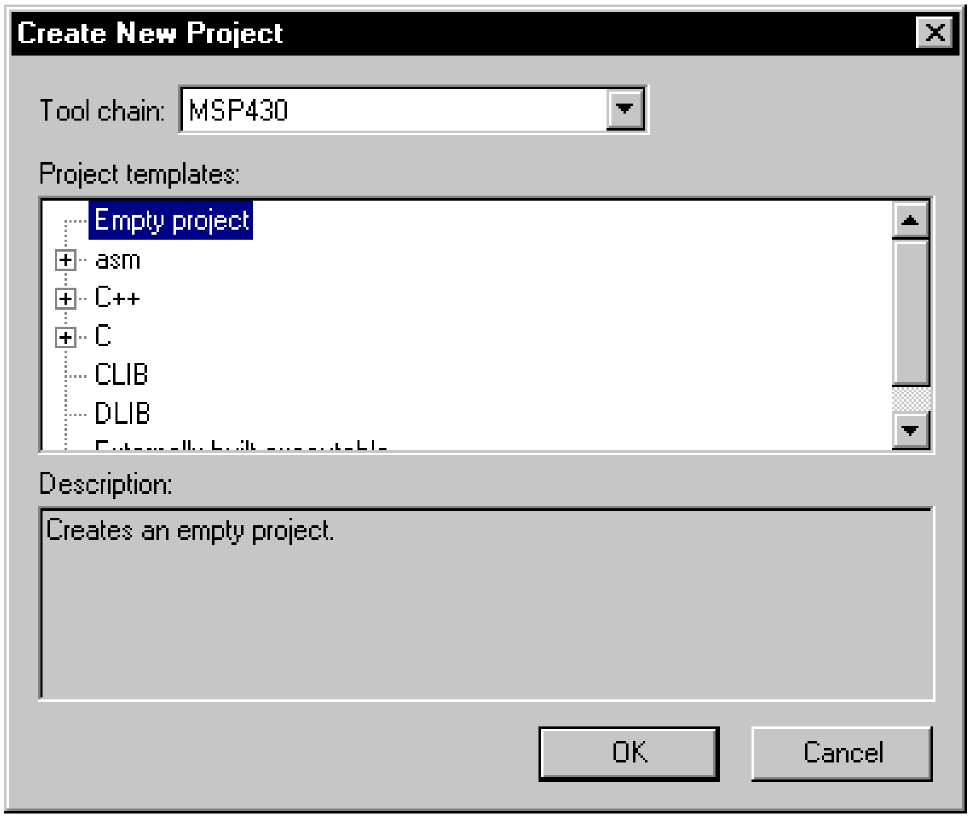
A whole program, which may contain a number of Assemble language files and C/C++ files, is called a project in IAR Embedded Workbench Kickstart and the project creating and debugging can be achieved with the IAR Embedded Workbench Kickstart. The IAR Embedded Workbench Kickstart can be downloaded from the following TI website and installed for free. http://focus.ti.com/docs/toolsw/folders/print/iar-kickstart.html

2.5.1 Creating a New Project

To create a new project, select **Project->Create New Project**. In the appearing dialog box, choose the **MSP430** in the Tool chain and **Empty project** in the Project templates, as shown in Figure 18. Click OK to create a new project with default setting.

Then the empty project appears in the Workspace window at left. Before adding any files to the project, the workspace should be saved by **File->Save Workspace** with a file name, as shown in Figure 19.

Choose **File->Add Files** to open a dialog box in which the files can be selected and click **open** to add into the project, as shown in Figure 20. Up to now, a new project has been created.



Tool chain: MSP430

Creates an empty project.

Figure 18. Create New Project

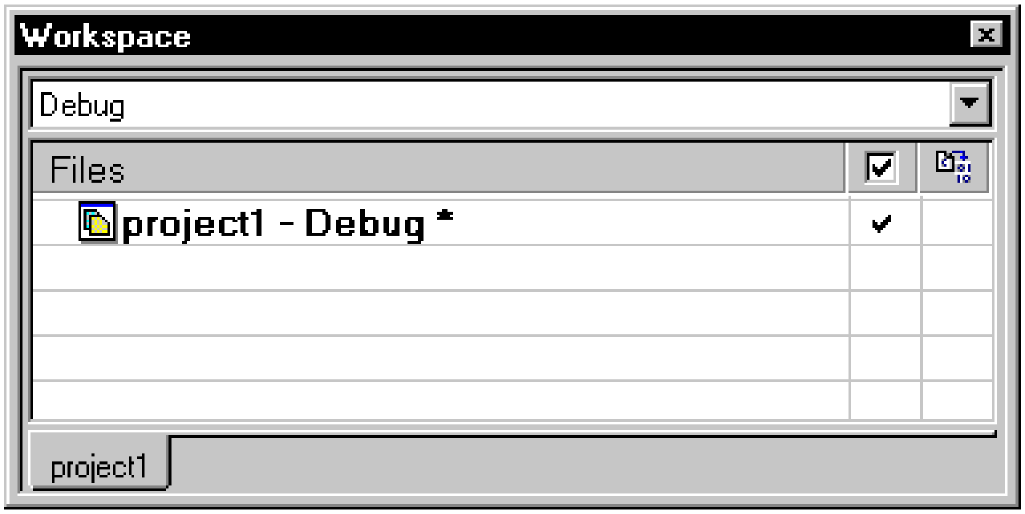
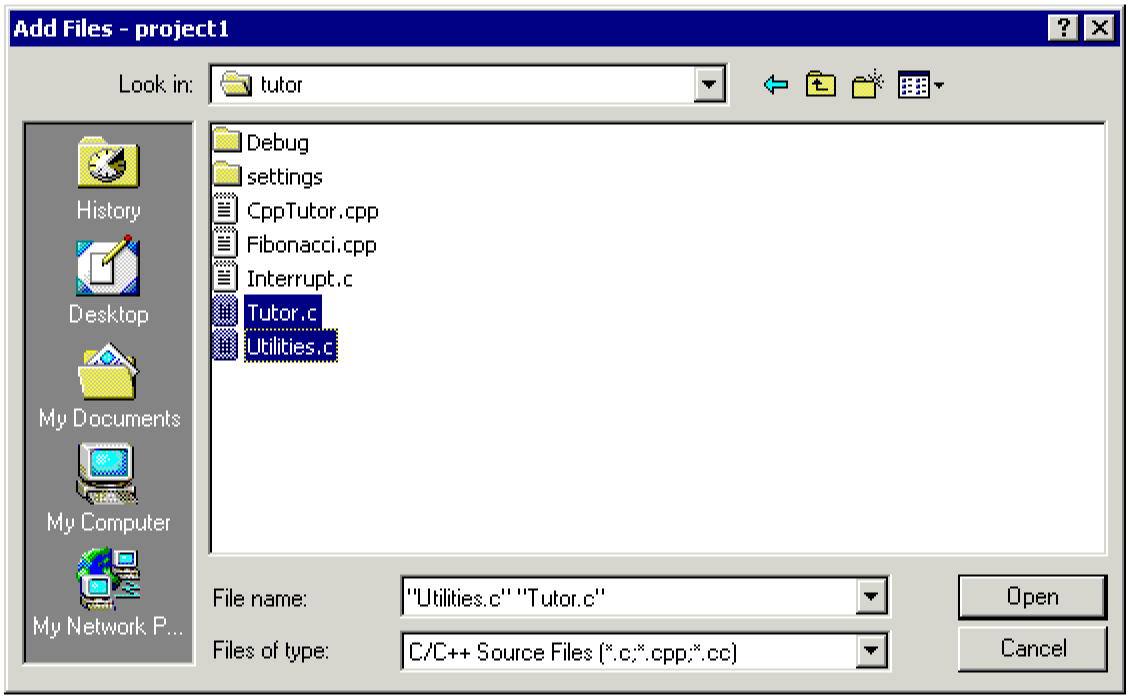


Figure 19. Workspace

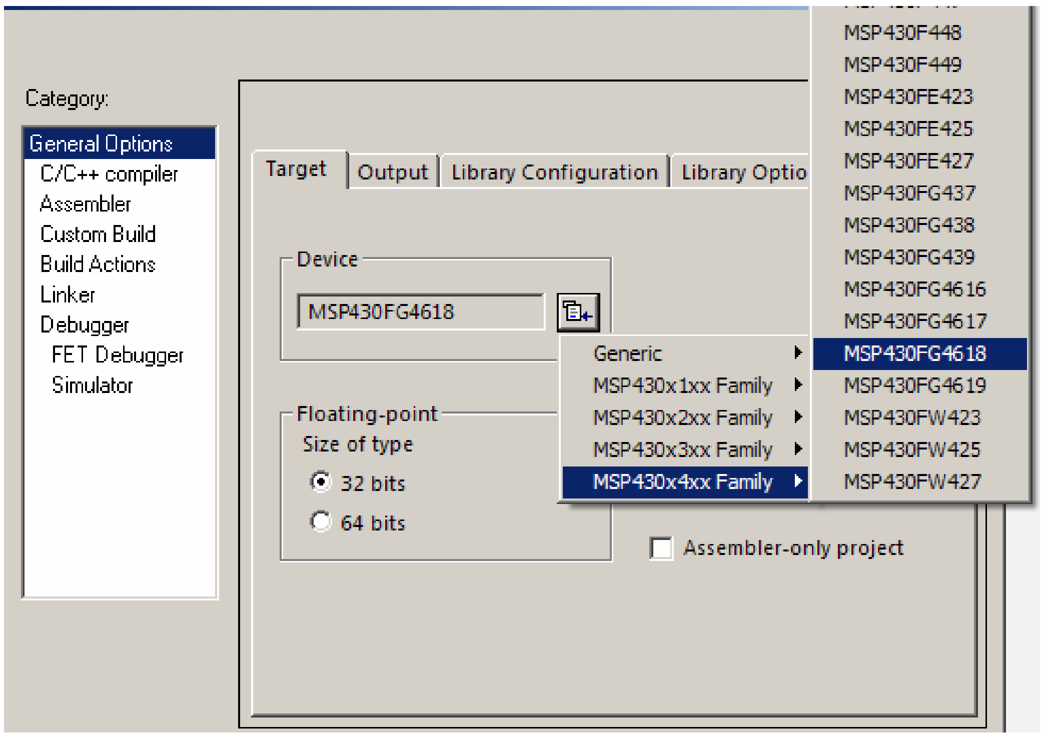


File name: "Utilities.c" "Tutor.c"

Files of type: C/C++ Source Files (*.c;*.cpp;*.cc)

Figure 20. Add Files

2.5.2 Setting Project Option



Device: MSP430FG4618

Floating-point

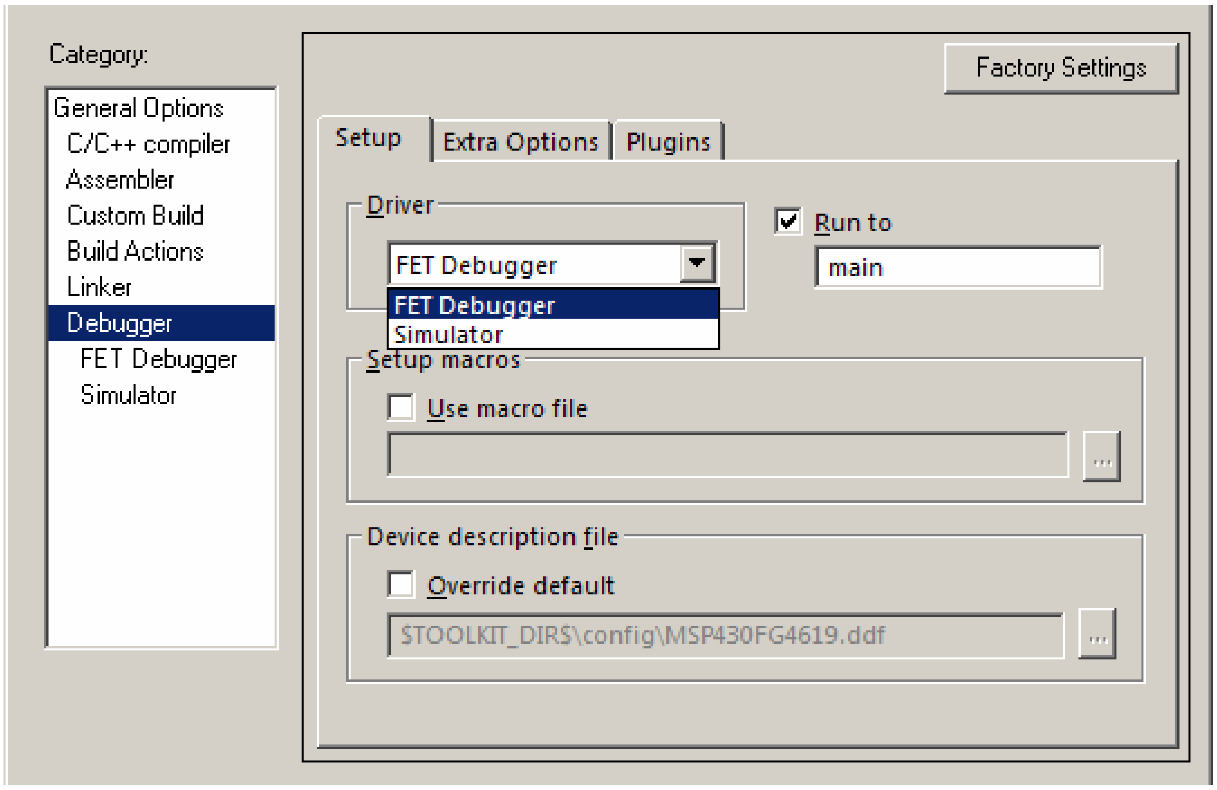
Size of type: [✓ 32 bits] 64 bits

☐ Assembler-only project

Figure 21. Choose Device

The project option should be changed because the default setting may not be suitable for all MSP430 microcontrollers. Select the project in the workspace window and choose **Project->Options** to open the dialog and the **General Options** category is shown. Under the Target option, choose the Device the project is working for. **MSP430FG4618** should be selected for MSP430FG4618/F2013 Experimenter Board, as shown in Figure 21. Similarly, choose **MSP430F2274** for eZ430RF2500 wireless development tool.

Select the **Debugger** category. Choose the **FET Debugger** under the Driver option, as shown in Figure 22.



Driver: FET Debugger

[✓] Run to: main

[ ] Use macro file

[ ] Override default

$TOOLKIT_DIR$\config\MSP430FG4619.ddf

Figure 22. Choose Driver

In the **FET Debugger** category, choose **Texas Instrument USB-IF** under connection option, as shown in Figure 23. Then close the option window to finish.

2.5.3 Downloading and Debugging

After programming, the application needs to be downloaded into the hardware. Before this step, choose **Project->Rebuild All** to finish the compiling and linking. There should be no errors or warnings in the message window, as shown in Figure 24. If not, some code should be revised.

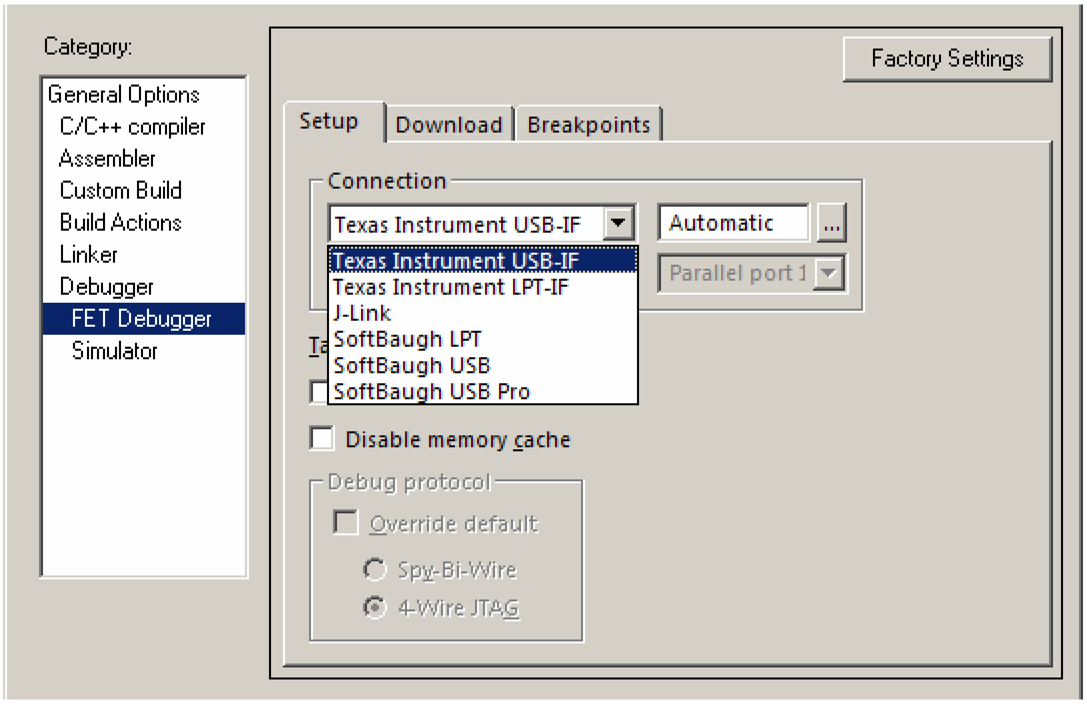
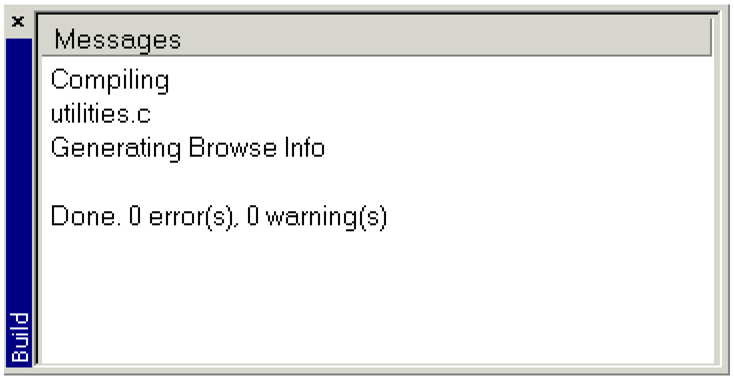


Figure 23.Choose Connection



```
Compiling
utilities.c
Generating Browse Info

Done. 0 error(s), 0 warning(s)
```

Figure 24. Message Window

Then connect the MSP430FG4618/F2013 Experimenter Board or eZ430RF2500 wireless development tool with the computer and choose **Project->Debug**. The program will be downloaded into the MSP430 and the debugger will be ready to start. Click **Debug->Go** to start debugging and **Debug->Stop Debugging** to stop. The hardware cannot be disconnected until the debugging is stopped.

When debugging in PC, the power supply comes from the FET tool. After disconnecting, the hardware can run the application with the power from AAA batteries. Every time the battery jumper is re-attached, the program will start over.

CHAPTER 3 IMPLEMENTION OF A WIRELESS SENSOR NETWORK

This project aims to implement a wireless sensor network to provide the temperature information of a given field by the MSP430FG4618/F2013 Experimenter Boards and the eZ430RF2500 wireless development tools. This sensor network can be applied in a building, for example. By placing sensor nodes everywhere in the building, the temperature information can be obtained from a PC collecting the information from all the sensor nodes using some network structure. There may be similar projects before. However, with the ultralow-power MSP430 family microcontrollers and the CC2500 RF transceivers, the system cost could be reduced, which prolongs the lifetime of the network without the need for changing the batteries, and the performance could be better. Furthermore, the nodes can be easily equipped with other types of sensors (e.g., chemical, biological, etc.) to allow the network to collect other information from the environment.

3.1 Sensor Network Structure

Either an MSP430FG4618/F2013 Experimenter Board or an eZ430RF2500 wireless development tool can be regarded as a sensor node with the temperature sensor integrated in the microcontroller. Considering the different sizes of these two systems, the bigger MSP430FG4618/F2013 experimenter board should be used as Base Station (BS) collecting information from several eZ430RF2500 nodes. There may have a number of base stations in the network with one board, designated as Master Board, used to collect the information from these BS’s which we refer to as Slave Board. The slave boards send the collected data from eZ430RF2500 nodes and data of the temperature they sense to the master board together. In this way, the master board that connects to a PC with a MSP-FET430UIF tool can deliver all the temperature data to the PC. Figure 25 shows the hierarchical structure of the network.

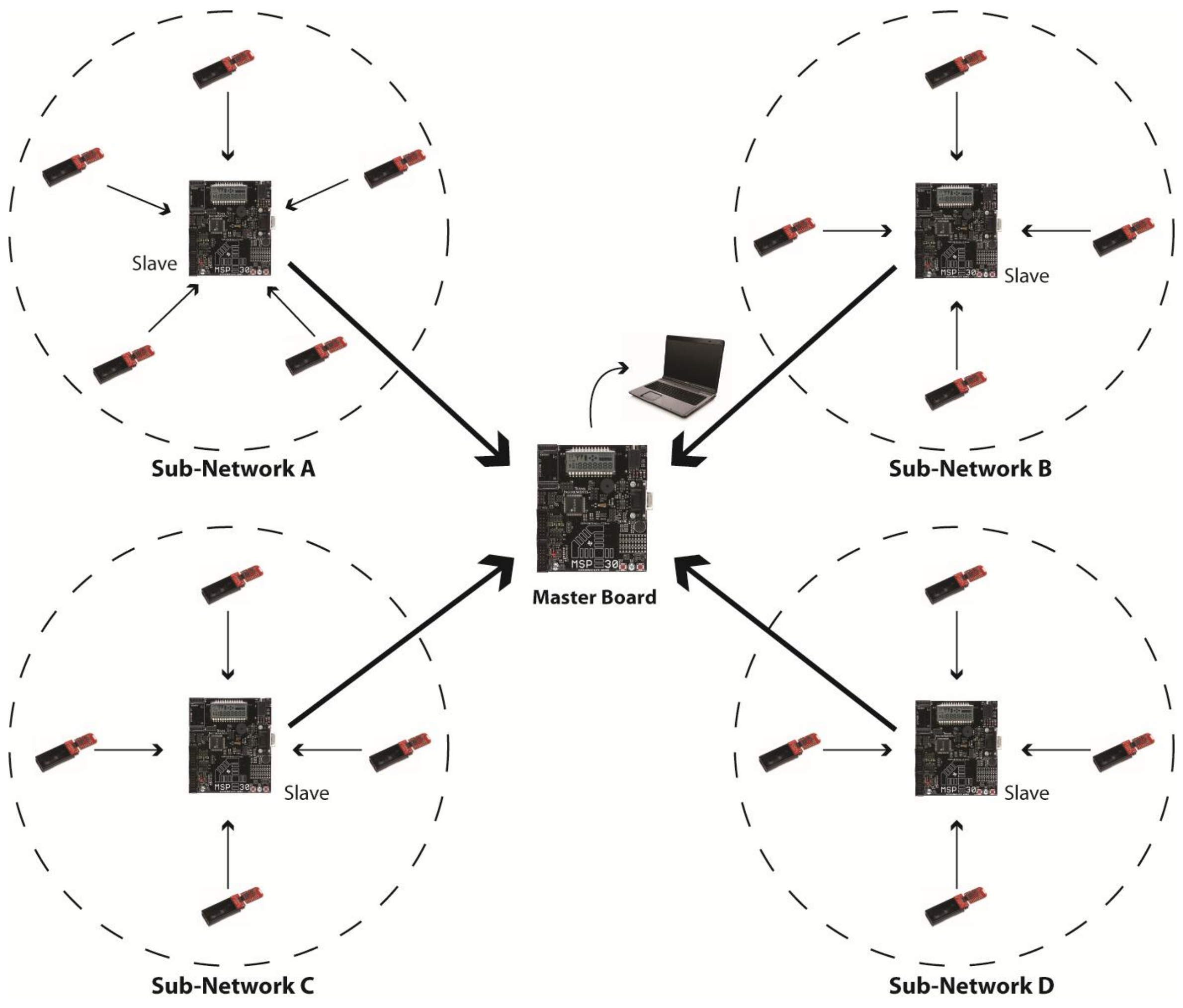


Figure 25. General Wireless Sensor Network

A slave board and the sensor nodes communicating to it can be regarded as a sub-network, as shown in Figure 25. Note that the master board may receive not only from the slave board but also from some eZ430RF2500 nodes, so the master board will be responsible for three kinds of sources: the temperature it senses, all the temperature data it receives from eZ430RF2500 nodes, and all the temperature data it receives from slave boards.

3.2 Basic Communication

The first job is to make the sensor nodes communicate, between two boards, or between a board and an eZ430RF2500 node, considering the communication between two eZ430RF2500 nodes is not required. Texas Instruments has some sample code for the boards which can be downloaded from the TI websites:

http://focus.ti.com/general/docs/litabsmultiplefilelist.tsp?literatureNumber=swra141

In those programs, some packets with dummy data are created and can be transmitted continuously or one at a time by pushing a button. Similarly, a program for eZ430RF2500 can be found at

http://focus.ti.com/general/docs/litabsmultiplefilelist.tsp?literatureNumber=slaa325a

In the programs above, the configurations of CC2500's are the same as the default setting. So it is not a problem to make the board and eZ430RF2500 nodes communicate with each other.

Then there comes some issues which will be discussed in the next section.

3.3 Multiple Access and Channelization

In a sensor network, it is possible that some nodes are in RX mode at the same time. This could cause confusion since the data transmitted to one node will be also received by other nodes in RX mode. This confusion arises if the receivers are set to the same channel. Another collision occurs when the BS is receiving signals from multiple sensor nodes, which results in the confusion for BS about where the signal comes from. Multiple access and channelization can be used to avoid these kinds of collisions.

Multiple access schemes are for the sensor nodes to transmit at a same time period while the issue of setting different channels for the sensor nodes is called channelization. Some choices of multiple access are discussed in the following sections.

3.3.1 Time Division Multiple Access (TDMA)

If a bandwidth is used, TDMA access scheme can be applied. TDMA uses time division multiplexing (TDM) for the BS to communicate with the sensor nodes in different time-slots one by one, which means each sensor node will be asked to transmit packets periodically. TDMA can be used in this wireless sensor network with the synchronization of clocks for all the sensor nodes, which is very difficult to achieve. So TDMA will not be considered in this thesis.

3.3.2 Scheduling

Scheduling is another access scheme that uses TDM besides TDMA. A sequence of time-slots is set for the network in the scheduling scheme, which is same as TDMA. However, transmissions that cannot collide with each other will be assigned in the same time-slot. So there could be more than one channel used in the whole system. Considering it also requires clock synchronization as TDMA does, scheduling will not be used here.

3.3.3 Frequency Division Multiple Access (FDMA)

FDMA access scheme provides different frequency bands for different sensor nodes to communicate. With FDMA, a BS can communicate with each sensor node in a single channel and the communication for all the sub-networks at the same time can be achieved. FDMA scheme can be absolutely applied in this thesis.

Recall the formula about carrier frequency in section 2.3.6:

$$f_{carrier} = \frac{f_{xosc}}{2^{16}} \cdot \left(FREQ + CHAN \cdot \left(\left(256 + CHANSPC\_M \right) \cdot 2^{CHANSPC\_E} \right) \right)$$

The default setting of the registers (FREQ, CHAN, CHANSPC_M, and CHANSPC_E), with f_{XOSC} of 26 MHz, gives a series of continuous channels with the channel spacing of 199.951 KHz. Channel can be changed by adding several channel spacings to the starting carrier frequency and the number of channel spacing added is decided by the value of CHAN register. So the only work is to change the value of the 8-bit CHAN register using the code.

Normally the configuration of a transceiver is unique, which means the transceiver in RX mode and TX mode will apply the same channel. However, it is possible for a transceiver to have two different configurations for RX mode and TX mode, respectively. If a sensor node is programmed to work in one channel for RX mode and another one for TX mode, even in several channels in TX or RX mode to communicate with different receivers or thansmitters, the channelization can be completed. Then all the sub-networks can perform transmission in different channels at the same time, which forms FDMA.

A brief test for channelization can be performed with one eZ430RF2500 sensor node and two boards as shown in Figure 26.

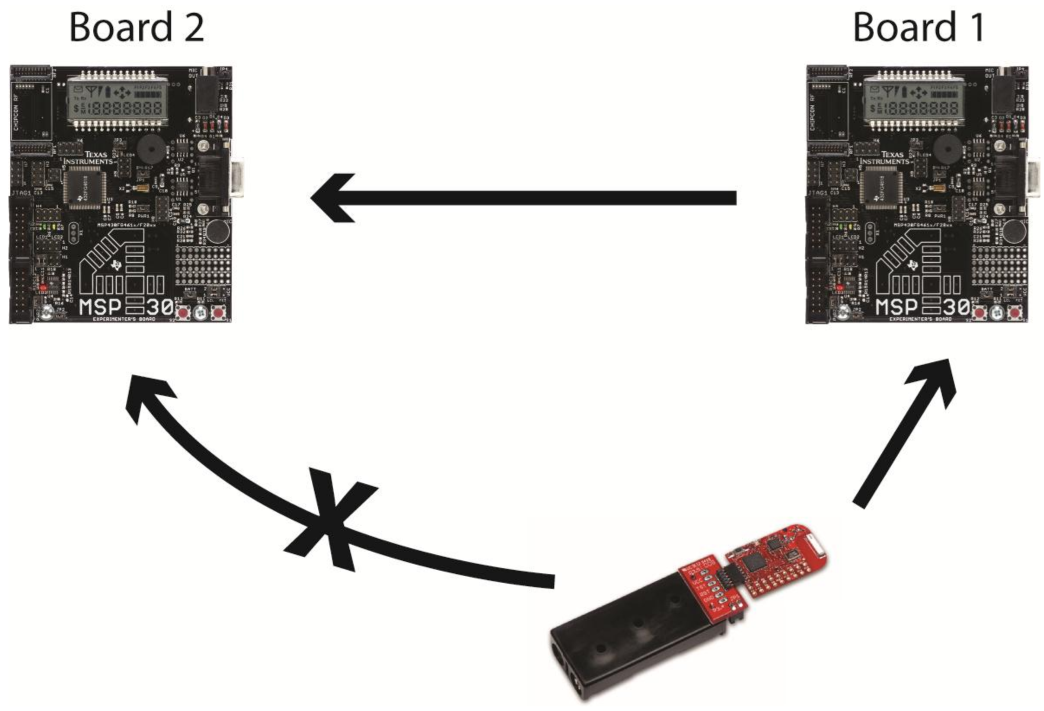


Figure 26. Channelization

The eZ430RF2500 node in TX mode is set to have the same channel with Board1 in RX channel while Board1 in TX mode is set to have the same channel, which is different with the former one, with Board2 in RX mode. In this case, only Board1 will receive the packet from the transmitting eZ430RF2500 node when both boards are set in RX mode. Then Board1 is programmed to deliver the received packet to Borad2 in the other channel. Code for the eZ430RF2500 and the two boards is shown in Appendix A, B, C, respectively.

3.3.4 Polling Scheme

Recall the application of eZ430RF2500 mentioned in section 2.1. When an End Device is powered on, it starts to search the Access Point by sending a query which the Access Point can recognize. Here is a similar way that is called rolling scheme. In the rolling scheme, it is the base station that sends the query and the node which can identify the query will response to the base station in this scheme. The query can be a short packet including payload of only one byte that indicates the ID of the node. So the base station sends the queries containing the IDs one by one in a certain order and waits for response after each query.

Rolling scheme is another access scheme that can be applied in this network and is easy to build. A test for polling scheme is can be completed with three boards, as shown in Figure 27. Two slave boards are in RX mode. The master board sends a packet including the ID of one slave board. The two slave boards receive the packet and check whether the ID in it is same with the ID of its own. The one that has the same ID will send the date back to the master board and then turns back to RX mode while the other one will just keep staying in RX mode to wait for next query from the master board. Then the master board sends the ID of the second board and the two slave boards will do the opposite after receiving.

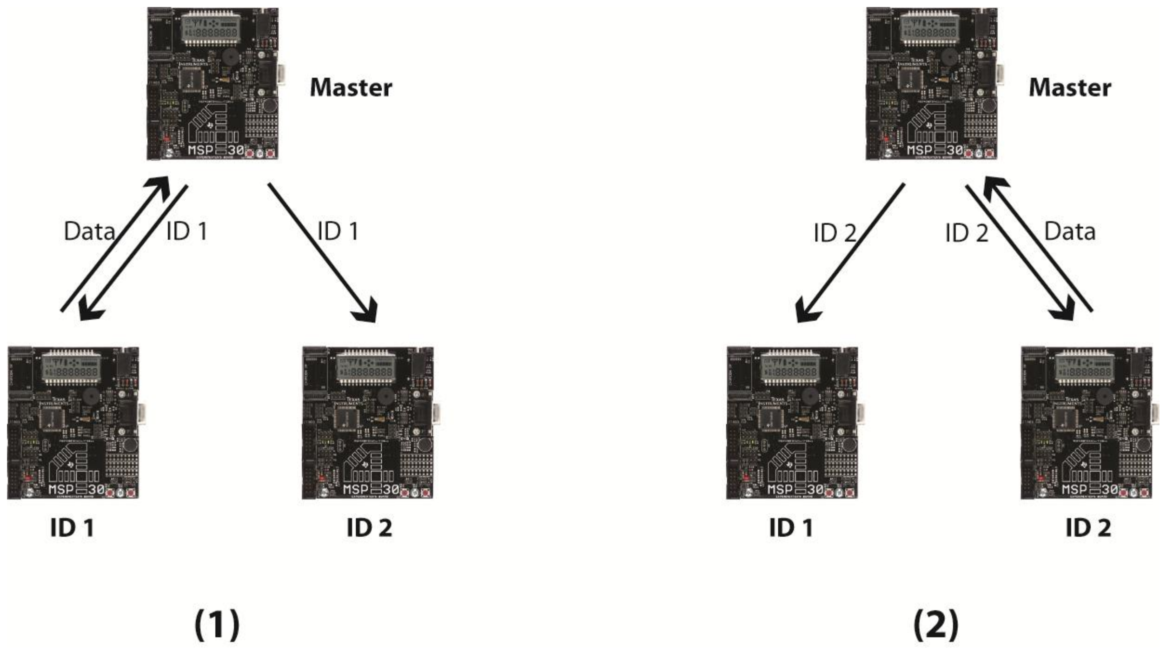


Figure 27. Simplified Polling Scheme

3.4 Transmission of the Temperature

As mentioned previously, the temperature sensed by the integrated temperature sensor is stored in a register of the microcontroller. However, the value of the register is the digitized reference voltage but not the temperature itself. Programming is needed to read the value of the register and transfer it into the real temperature value. By replacing the data of the packet used in the test of section 3.3.3 with the temperature value, the temperature can be transmitted.

3.5 Operation of the Whole Network

By applying the sensor network structure in Figure 25, the wireless sensor network in this thesis can be built, as shown in Figure 28.

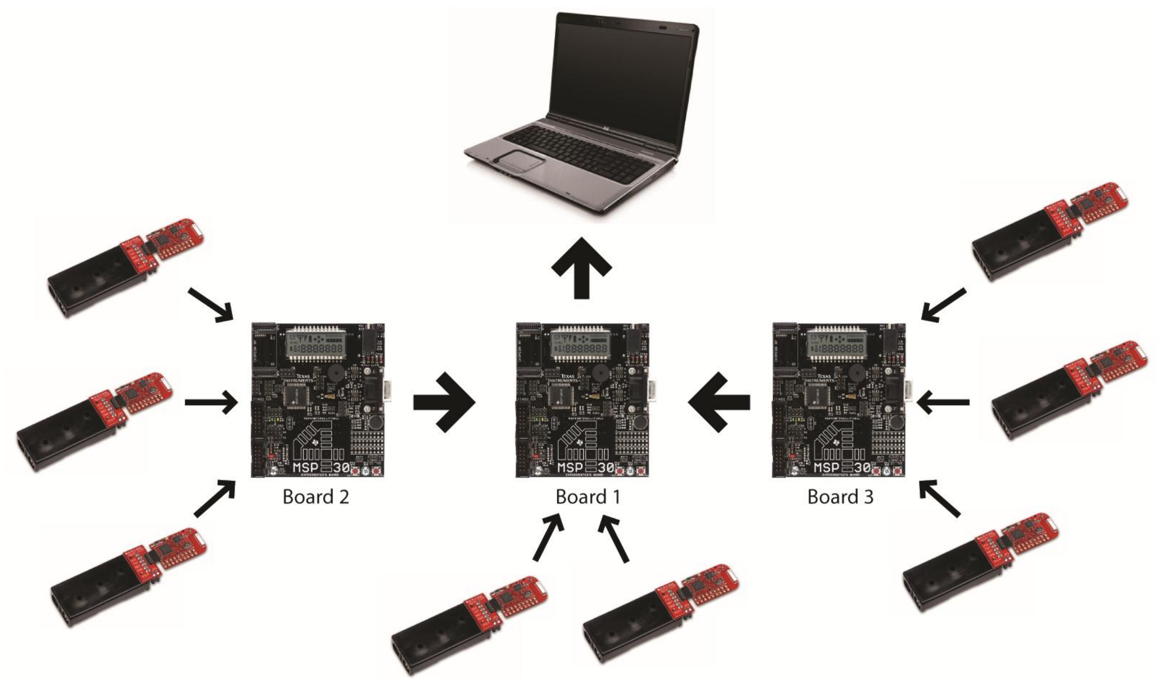


Figure 28. Wireless Sensor Network with MSP430FG4618/F2013 Experimenter Boards and eZ430RF2500 Wireless Development Tools

In this network, Board1 is the master board connecting to a PC. Board2 and Board3 are the two slave boards each of which communicates with three eZ430RF2500 nodes. Besides from the slave boards, the master board also receives data from two eZ430RF2500 nodes.

It surely works with the polling scheme. However, in the real-world application, power consumption should be considered to extend the lifetime as much as possible. A lot of energy, for example, will be consumed for nothing if making the slave boards stay in RX mode when waiting for the queries for them. So more details for power saving are needed to design the protocols of the wireless sensor network.

3.5.1 Design for Power Saving

In the polling scheme designed in section 3.3.4, the nodes waiting for queries stay in RX mode all the time, which consumes a lot of power. To reduce the power, the microcontrollers can enter into any low-power mode (usually LPM3 mode) and the transceiver can enter into SLEEP mode. However, the transceiver will be unable to receive anything under the SLEEP mode. So the Wake-ON-Radio can be applied to solve this problem.

The Wake-ON-Radio, described in section 2.3, keeps the CC2500 transceiver to wake up periodically from SLEEP mode and stay in RX mode for a short time. The queries will be received when CC2500 is in RX mode. The incoming signal can generate an interrupt in which the mode of the microcontroller will be changed to exit LPM3 mode and then the microcontroller can process the data and send it back to the board that transmits the query. After the transmission, the microcontroller goes back to LPM3 mode while the CC2500 transceiver enters into SLEEP and wakes up again after a period.

Care must be taken that the query for one node should be sent repeatedly and the interval should be smaller than the time that the receiving node stays in RX mode when operating Wake-ON-Radio. Otherwise, there is probability that the receiving node will miss the query.

Another issue is that all the eZ430RF2500 nodes in the sub-network can receive every query from the base station and wake up the microcontroller. After checking the ID in the received query, only one node will continue working and the other nodes will be turned back to SLEEP mode, which results in unnecessary power consumption from the mode changing and computing of the microcontroller.

A way to solve this problem is to combine the polling scheme with FDMA scheme by setting a unique channel for each node, as shown in Figure 29.

In this case, it is even not required to check ID for polling scheme because only the node in the specific channel can receive the query. Then the node wakes up its microcontroller and sends the data back to the base station. After receiving the data, the base station sends the second query in another channel to the second node in the sub-network.

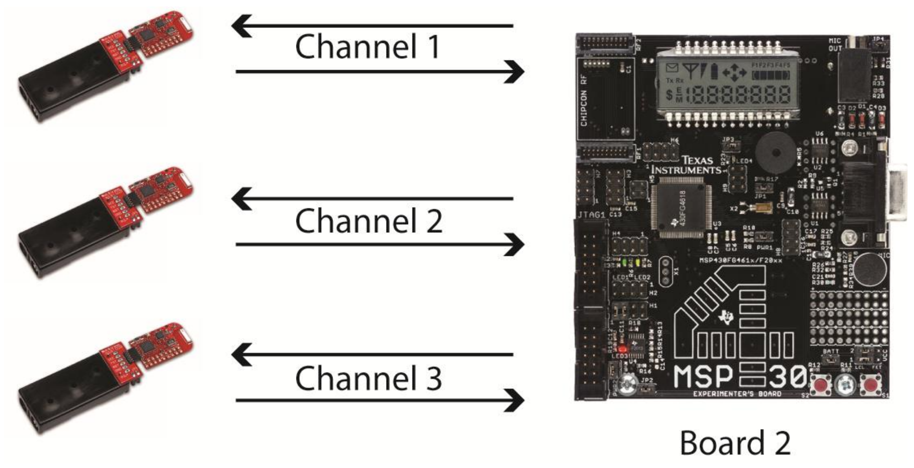


Figure 29. Polling Scheme with FDMA

This scheme can be also applied by the master board. The two slave boards and the two eZ430RF2500 nodes connecting to the master board will stand in RX mode in different channels to wait for the master board sending queries one by one.

3.5.2 Program Flow

First of all, all sensor nodes, including the two slave boards, are programmed to stay in low-power mode (LPM3 for microcontrollers and Wake-On-Radio for transceivers) after they collect the temperature data and the packets to be sent are ready. Then the master board sends query to start the process. The program flow for the master board is shown in Figure 30.

The master board sets channel and sends query to the first slave board. Considering that the CC2500 of the slave board could be in SLEEP mode when the master board is querying, the master board should send the query repeatedly. After each time of querying, the master board

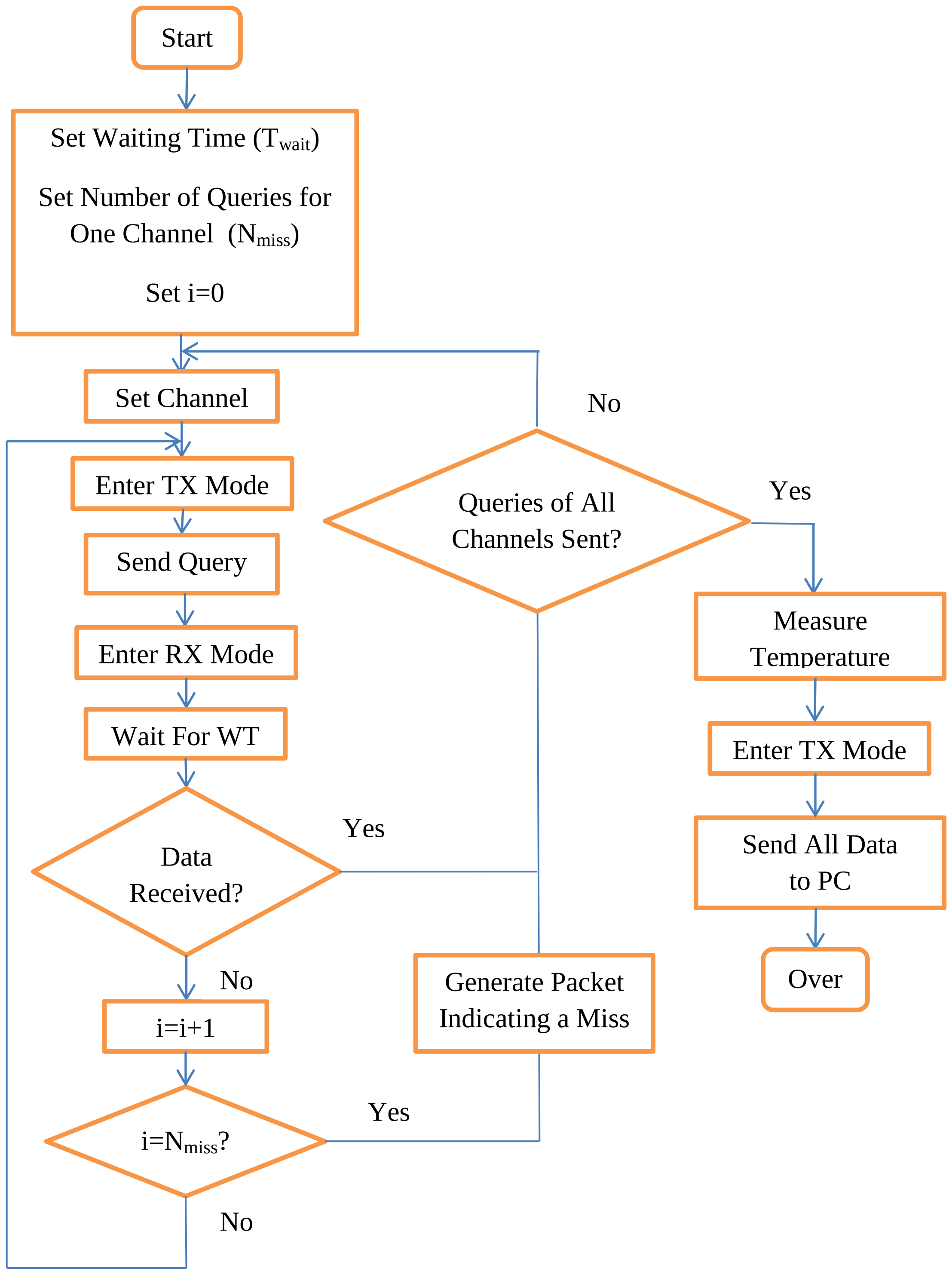


Figure 30. Program Flowchart for Master Board

should be set into RX mode to wait for the response of the slave board. The Waiting Time (T_{wait}) should be long enough for the slave board to receive the query and send data back but shorter than the time the slave board stays in RX mode during a period of WOR to make sure the slave board will not miss the query, which is discussed in section 3.5.1. Once the CC2500 of the slave board wakes up and enters into RX mode, it will receive the query and immediately send the data back to the master board.

Since the master board queries the slave board repeatedly, it cannot stop querying if there is something wrong with the slave board that cannot response, which results in a stagnation of the whole process. So a Missing Time is needed to make the master board quit the querying and generate a packet indicating that the slave board it has been querying is missing. The missing time is set to be a number (N_{miss}) of several query processes which makes the total time be equal to or a little longer than the period of the WOR of the slave board to avoid a wrong decision.

After either receiving data or generating a packet for missing, the master board will be set to change channel and query next slave board or node. Once all the querying work is done, the master board will measure temperature itself and send all the data to the PC. Then the master board can stay in IDLE mode to wait for next process. Since the power supply of master board is from PC, it is not required to design power saving for the master board.

Slave boards wake up by queried and send the data back to the master board. Then they start to collect data from the nodes in their sub-networks by querying, which is independent with the master board querying the left nodes and saves time. The querying protocol is same as that of the master board except the channels it uses. After querying and measuring themselves, the slave boards arrange the packets to be ready to transmit and enter into SLEEP/LPM3 mode.

Each eZ430RF2500 node wakes up by queried and sends the data to the slave board in its sub-network. Then they measure the temperature, prepare the packets, and enter into SLEEP/LPM3 mode, which is also independent with other devices.

3.6 Real World Application

This wireless sensor network can be applied in a building to measure temperature of several rooms, as shown in Figure 31.

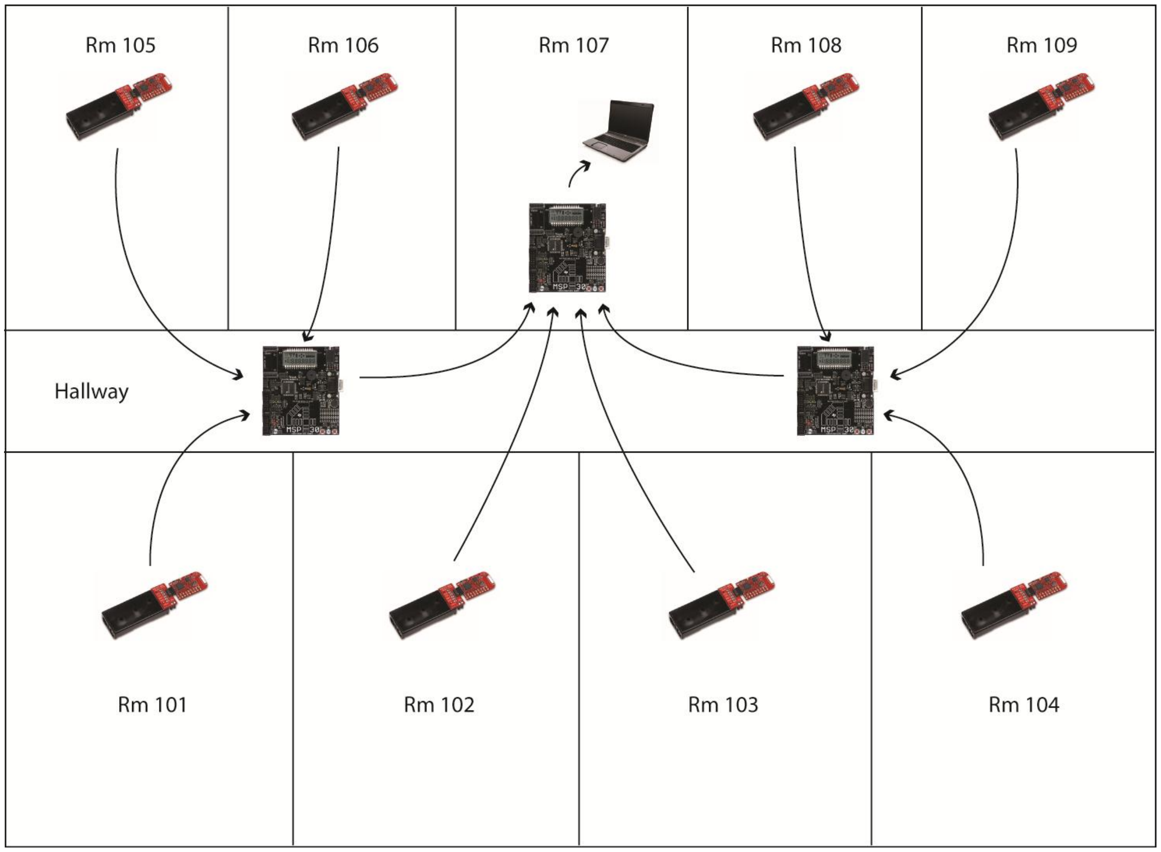


Figure 31. Real World Application of Wireless Sensor Network

In this application, the eight eZ430RF2500 nodes reside in eight rooms, respectively. The two slave boards are set in the hall way, collecting temperature data of six rooms. And the master board is connected with the PC in the ninth room, measuring temperature of that room and receiving data from the two slave boards and the left two eZ430RF2500 nodes. All the eZ430RF2500 nodes are programmed to send packets that include the temperature data and their own IDs which can be the number of rooms they reside in. Then temperature of all rooms and the hall way can be displayed on the PC. User can check the current temperature by pushing a REFRESH button on PC, which start the process beginning with the master board, or let the system refresh itself in a period (five or ten seconds).

CHAPTER 4 CONCLUSION

4.1 Summary

A wireless sensor network is implemented with eZ430RF2500 wireless development tools and MSP430FG4618/F2013 experimenter boards, the ultralow-power and low-cost devices from Texas Instruments. By applying polling scheme with channelization and Wake-On-Radio, the reduction of power consumption is achieved. The total system power can be obtained by measuring every single device.

In this system, it can be proved that the temperature displayed on PC is one period ago for the slave boards and two periods ago for the eZ430RF2500 nodes each time when the display is refreshed. Because the slave boards store the sensed data in memory and transmit it next time when queried and eZ430RF2500 nodes have to wait two periods to have their data transmit to PC. It is still fine for general application like measuring the temperature in a building as shown in section 3.6. However, this protocol has to be revised if the temperature of the current moment is required.

In that case, every device has to be programmed to sense the temperature immediately after queried. Based on this protocol, the slave boards start querying the eZ430RF2500 nodes in their own sub-network after queried by the master board and the eZ430-RF2500 nodes measure the temperature after queried by the slave boards and send the data back. Then the slave boards sense their temperature and transmit all the data to the master board. So the Waiting Time's for the boards need to be set longer because the devices they query cannot response immediately. And it will take long time totally for a single process because the sensing work of the nodes is set in an orderly sequence but never be independent of each other any more.

4.2 Future Work

Recall that the sensed temperature needs to be computed to transfer from an 8-bit binary value, which is done by the microcontrollers of the sensor nodes. Future work can be programming to directly transfer the 8-bit binary values and finish all the computation in PC, which can save time of computing for the devices with batteries and then save power.

Since the MSP430FG4618/F2013 Experimenter Board has integrated a number of on-board components, perhaps some of these can be used in the future. For example, the buzzer can be utilized to call for help if there is something wrong with the board. The LEDs of eZ430RF2500 can be set to flash to indicate a problem of the device. In addition, the ultralow-power feature of this network can be shown by comparing with another sensor network of similar size.

REFERENCES

[1]Jennifer Yick, Biswanath Mukherjee, Dipak Ghosal, Wireless Sensor Network Survey [J]. Computer Networks, v 52, n 12, p 2292-2330, August 22, 2008

[2]M. Tubaishat, S. Madria, Sensor Network: An Overview [J]. IEEE Potentials, v 22, n 2, p 20-23, May 07, 2003

[3]V. Raghunathan, C. Schurgers, Sung Park, M.B. Srivastava, Energy-aware Wireless Microsensor Networks [J], IEEE Signal Processing Magazine, v 19, n 2, p 40-50, August 07, 2002

[4]I.F. Akyildiz, Weilian Su, Y. Sankarasubramaniam, E. Cayirci, A Survey on Sensor Networks [J], IEEE Communication Magazine, v 40, n 8, p 102-114, November 07, 2002

[5]R.C. Shah, J.M. Rabaey, Energy-aware Routing for Low Energy Ad Hoc Sensor Networks [J], IEEE Wireless Communications and Networking Conference, v 1, p 350-355, August 07, 2002

[6] eZ430-RF2500 Development Tool User's Guide
http://focus.ti.com/lit/ug/slau227e/slau227e.pdf

[7] Wireless Sensor Monitor Using the eZ430-RF2500
http://focus.ti.com/lit/an/slaa378d/slaa378d.pdf

[8]MSP430FG4618/F2013 Experimenter's Board User Guide
http://focus.ti.com/lit/ug/slau213a/slau213a.pdf

[9] CC2500 low-cost low-power 2.4 GHz RF Transceiver
http://focus.ti.com/lit/ds/swrs040c/swrs040c.pdf

[10] CC1100/CC2500 - Wake-On-Radio
http://focus.ti.com/lit/an/swra126b/swra126b.pdf

[11] MSP430x4xx Family User's Guide
http://focus.ti.com/lit/ug/slau056j/slau056j.pdf

[12] MSP430x2xx Family User's Guide
http://focus.ti.com/lit/ug/slau144h/slau144h.pdf

[13]MSP430 IAR Embedded Workbench IDE User Guide for Texas Instruments' MSP430 Microcontroller Family
http://www.hep.princeton.edu/~marlow/rrs/Guides/Workbench.pdf

[14] Software for CC1100CC2500 and MSP430
http://focus.ti.com/lit/an/swra141/swra141.pdf

APPENDIX A. CODE OF EZ430RF2500 FOR CHANNELIZATION

```
#include "include.h"

extern char paTable[];

extern char paTableLen;

char txBuffer[4];

char rxBuffer[4];

unsigned int i = 0;

char channel = 0x00;

void main (void)

{WDTCTL = WDTPW + WDTHOLD;                  // Stop WDT

  TI_CC_SPISetup();                         // Initialize SPI port

  P2SEL = 0;                                // Sets P2.6 & P2.7 as GPIO

  TI_CC_PowerupResetCCxxxx();               // Reset CC2500

  writeRFSettings(channel);                 // Write RF settings to configure register

  TI_CC_SPIWriteBurstReg(TI_CCxxx0_PATABLE, paTable, paTableLen);//Write PATABLE

  // Configure ports -- switch inputs, LEDs, GDO0 to RX packet info from CC2500

  TI_CC_SW_PxREN = TI_CC_SW1;               // Enable Pull up resistor

  TI_CC_SW_PxOUT = TI_CC_SW1;               // Enable pull up resistor

  TI_CC_SW_PxIES = TI_CC_SW1;               // Interrupt on falling edge

  TI_CC_SW_PxIFG &= ~(TI_CC_SW1);           // Clear flags

  TI_CC_SW_PxIE = TI_CC_SW1;                // Activate interrupt enables

  TI_CC_LED_PxOUT &= ~(TI_CC_LED1 + TI_CC_LED2); // Outputs = 0
  TI_CC_LED_PxDIR |= TI_CC_LED1 + TI_CC_LED2; // LED Direction to Outputs
  TI_CC_GDO0_PxIES |= TI_CC_GDO0_PIN;       // Interrupt on falling edge (end of packet)
  TI_CC_GDO0_PxIFG &= ~TI_CC_GDO0_PIN;      // Clear flag
  TI_CC_GDO0_PxIE |= TI_CC_GDO0_PIN;        // Enable interrupt on end of packet
  TI_CC_SPIStrobe(TI_CCxxx0_STX);           // Initialize CC2500 in RX mode.
                            //When a packet is received, it will signal on GDO0 and wake CPU
 __bis_SR_register(LPM3_bits + GIE);       // Enter LPM3, enable interrupts
}
// The ISR assumes the interrupt came from a pressed button
#pragma vector=PORT1_VECTOR
__interrupt void Port1_ISR (void)
{
 // If Switch was pressed
  if(TI_CC_SW_PxIFG & TI_CC_SW1)
  {// Build packet
    txBuffer[0] = 2;                        // Packet length
    txBuffer[1] = 0x01;                     // Packet address
    txBuffer[2] = 40;                       // Data
    RFSendPacket(txBuffer, 3);              // Send value over RF
    __delay_cycles(5000);                   // Switch debounce
  }
  TI_CC_SW_PxIFG &= ~(TI_CC_SW1);           // Clear flag that caused interrupt
}
```

APPENDIX B. CODE OF BOARD1 FOR CHANNELIZATION

```
#include "include.h"

extern char paTable[];

extern char paTableLen;

char txBuffer[4];

char rxBuffer[4];

unsigned int i = 0;

char channel[] = {0x00 , 0x01};

void main (void)

{

  WDTCTL = WDTPW + WDTHOLD;                // Stop WDT

  TI_CC_SPISetup();                        // Initialize SPI port

  TI_CC_PowerupResetCCxxxx();              // Reset CC2500

  writeRFSettings(channel[0]);                       // Write RF settings to configure register

  TI_CC_SPIWriteBurstReg(TI_CCxxx0_PATABLE, paTable, paTableLen);//Write PATABLE

                // Configure ports -- switch inputs, LEDs, GDO0 to RX packet info from CC2500

  TI_CC_SW_PxIES = TI_CC_SW1+TI_CC_SW2;     // Interrupt on falling edge

  TI_CC_SW_PxIFG &= ~(TI_CC_SW1+TI_CC_SW2); // Clear flags

  TI_CC_SW_PxIE = TI_CC_SW1+TI_CC_SW2;      // Activate interrupt enables

  TI_CC_LED_PxOUT &= ~(TI_CC_LED1 + TI_CC_LED2); // Outputs = 0

  TI_CC_LED_PxDIR |= TI_CC_LED1 + TI_CC_LED2; // LED Direction to Outputs
  TI_CC_GDO0_PxIES |= TI_CC_GDO0_PIN;       // Interrupt on falling edge (end of packet)
  TI_CC_GDO0_PxIFG &= ~TI_CC_GDO0_PIN;      // Clear flag
  TI_CC_GDO0_PxIE |= TI_CC_GDO0_PIN;        // Enable interrupt on end of packet
  TI_CC_SPIStrobe(TI_CCxxx0_SRX);           // Initialize CC2500 in RX mode.
                                    // When a packet is received, it will signal on GDO0 and wake CPU
  InitTimerB(60000);
  __bis_SR_register(LPM3_bits + GIE);       // Enter LPM3, enable interrupts
}
void InitTimerB(unsigned long rate)
{
   TBCCR0 = rate;// max: TBR=0xFFFF
   TBCTL = 0;
   TBCTL = CNTL_0 + TBSSEL_2 + ID_0 + MC_3;
}
// The ISR assumes the interrupt came from a press of one of the two buttons
// and pin attached to GDO0. GDO0 fires indicating that CC2500 received a packet
#pragma vector=PORT1_VECTOR
__interrupt void port1_ISR (void)
{
  if(P1IFG & TI_CC_GDO0_PIN)
  {
   char len=2;                        // Length of packet to be received (only address
                          // plus data; size byte not including because stripped away within RX function)
   if (RFReceivePacket(rxBuffer,&len))
   {
         LcdWriteValue(rxBuffer[1]);
         McuWaitUs(65000, 5);
         TI_CC_SPIWriteReg(TI_CCxxx0_CHANNR,   channel[1]);
         txBuffer[0] = 2;                        // Packet length
         txBuffer[1] = 0x01;                     // Packet address
         txBuffer[2] = 50;                       // Data
         RFSendPacket(txBuffer, 3);              // Send value over RF
         LcdWriteValue(txBuffer[2]);
         TBCCTL0 |= CCIE;
   }
  }
  TI_CC_GDO0_PxIFG &= ~TI_CC_GDO0_PIN;    // After packet is transmitted, this flag is set.
}                                         // Clear it.
#pragma vector=TIMERB0_VECTOR
__interrupt void TB0_ISR(void)
{
  RFSendPacket(txBuffer, 3);
}
```

APPENDIX C. CODE OF BOARD2 FOR CHANNELIZATION

```
#include "include.h"

extern char paTable[];

extern char paTableLen;

char txBuffer[4];

char rxBuffer[4];

unsigned int i = 0;

char channel = 0x01;

void main (void)

{

  WDTCTL = WDTPW + WDTHOLD;                // Stop WDT

  TI_CC_SPISetup();                        // Initialize SPI port

  TI_CC_PowerupResetCCxxxx();              // Reset CC2500

  writeRFSettings(channel);                // Write RF settings to configure register

  TI_CC_SPIWriteBurstReg(TI_CCxxx0_PATABLE, paTable, paTableLen);//Write PATABLE

                 // Configure ports -- switch inputs, LEDs, GDO0 to RX packet info from CCxxxx

  TI_CC_SW_PxIES = TI_CC_SW1+TI_CC_SW2;    // Interrupt on falling edge

  TI_CC_SW_PxIFG &= ~(TI_CC_SW1+TI_CC_SW2); // Clear flags

  TI_CC_SW_PxIE = TI_CC_SW1+TI_CC_SW2;     // Activate interrupt enables

  TI_CC_LED_PxOUT &= ~(TI_CC_LED1 + TI_CC_LED2); // Outputs = 0

  TI_CC_LED_PxDIR |= TI_CC_LED1 + TI_CC_LED2; // LED Direction to Outputs

  TI_CC_GDO0_PxIES |= TI_CC_GDO0_PIN;      // Interrupt on falling edge (end of packet)
  TI_CC_GDO0_PxIFG &= ~TI_CC_GDO0_PIN;      // Clear flag
  TI_CC_GDO0_PxIE |= TI_CC_GDO0_PIN;        // Enable interrupt on end of packet
  TI_CC_SPIStrobe(TI_CCxxx0_SRX);           // Initialize CC2500 in RX mode.
                                    // When a packet is received, it will signal on GDO0 and wake CPU
  __bis_SR_register(LPM3_bits + GIE);       // Enter LPM3, enable interrupts
}
// The ISR assumes the interrupt came from a press of one of the two buttons
// and pin attached to GDO0. GDO0 fires indicating that CC2500 received a packet
#pragma vector=PORT1_VECTOR
__interrupt void port1_ISR (void)
{
  if(P1IFG & TI_CC_GDO0_PIN)
  {
    char len=2;                             // Length of packet to be received (only address
                        // plus data; size byte not including because stripped away within RX function)
    if (RFReceivePacket(rxBuffer,&len))
    {
        LcdWriteValue(rxBuffer[1]);
    }
  }
  TI_CC_GDO0_PxIFG &= ~TI_CC_GDO0_PIN;    // After packet is transmitted, this flag is set.
}                                          // Clear it.
```

VITA

Lu De Yang was born in Tianjin, China. His interest in electrical engineering comes from the hearing aid he has been wearing from his childhood. Lu went to Jilin University to study and got the bachelor degree of electrical engineering in July, 2009. Then he came to Louisiana State University in United State of America. pursuing the master degree of electrical engineering. After the first semester of study, he met his future major advisor Dr. Morteza Naraghi-Pour and began his research for thesis in Fall 2010. Lu finished his thesis in Summer 2011 and will be awarded the degree of Master of Science in Electrical Engineering in August 2011.