UNIVERSITY OF VAASA

FACULTY OF TECHNOLOGY

TELECOMMUNICATION ENGINEERING

Gülcan Çuhac

DESIGN AND IMPLEMENTATION OF A WIRELESS HOME AUTOMATION CONTROL SYSTEM WITH SPEECH RECOGNITION

Master´s thesis for the degree of Master of Science in Technology submitted for inspection, Vaasa, 06 May, 2013.

Supervisor Professor Mohammed Salem Elmusrati

Instructor M. Sc. (Tech.) Tobias Glocker

TABLE OF CONTENTS

ABBREVIATIONS

ACLK	Auxiliary Clock
ADC	Analog-to-Digital Converter
AGC	Automatic Gain Control
ASK	Amplitude Shift Keying
ASR	Automatic Speech Recognition
BFSK	Binary Frequency Shift Keying
BPSK	Binary Phase Shift Keying
CAN	Controller Area Network
CPFSK	Continuous-Phase Frequency Shift Keying
CPHA	Clock Phase
CPOL	Clock Polarity
CPU	Central Processing Unit
DAC	Digital-to-Analog Converter
DCO	Digitally Controlled Oscillator
EEPROM	Electrically Erasable Programmable Read-Only Memory
EMF	Electromotive Force
FIFO	First In First Out
FSK	Frequency Shift Keying
GND	Ground
GPIO	General Purpose Input/Output
HMM	Hidden Markov Model
IC	Integrated Circuit
I²C	Inter-Integrated Circuit
IF	Intermediate Frequency
ISR	Interrupt Service Routine
I/O	Input/Output
LCD	Liquid Crystal Display
LED	Light-Emitting Diode
LNA	Low- Noise Amplifier
LPM	Low Power Mode

MCLK	Master Clock
MFSK	Multilevel Frequency Shift Keying
MSK	Minimum Shift Keying
PCB	Printed Circuit Board
PSK	Phase Shift Keying
PWM	Pulse Width Modulation
QPSK	Quadrature Phase Shift Keying
RF	Radio Frequency
RSSI	Received Signal Strength Indicator
Rx	Reception
SDI	Serial Data Input
SDO	Serial Data Output
SMCLK	Sub-System Master Clock
SNR	Signal to Noise Ratio
SPI	Serial Peripheral Interface
TI	Texas Instruments
Tx	Transmission
UART	Universal Asynchronous Receiver/Transmitter
WSN	Wireless Sensor Network
XML	Extensible Markup Language

UNIVERSITY OF VAASA
Faculty of technology
Author: Gülcan Çuhac
Topic of the Thesis: Design and Implementation of a Wireless Home Automation Control System with Speech Recognition
Supervisor: Mohammed S. Elmusrati
Instructor: Tobias Glocker
Degree: Master of Science in Technology
Department: Department of Computer Science
Degree Programme: Degree Programme in Information Technology
Major of Subject: Telecommunications Engineering
Year of Entering the University: 2012
Year of Completing the Thesis: 2013 Pages: 81

ABSTRACT

Home automation systems became one of the most interesting areas for both the construction and the electronics sector. Changing the state of the home appliances easily, scheduling events, and remote control capabilities using high technologies attract modern home residents everyday. This thesis researches the possibilities of applying speech recognition solutions to automated homes. Speech based solutions would provide great benefits especially to disabled or elder people. In addition, wireless devices prevent cabling complications through the walls.

An open source software based on the Hidden Markov Models called Sphinx 4 has been used to realize the speech recognition in this thesis. The speech recognition system has been developed in a Linux PC and a wireless node was attached to it, so that it became a small command center. Another wireless node was connected to a lighting control system and a servo motor so that it became an actuator, wirelessly controlled from the command center. This way the skeleton of a speech based home automation system has been built and verified to work. In the results section, the recognition accuracy analysis, power consumption tests, and range tests were performed to verify the robustness of the system.

KEYWORDS: Home Automation, Speech Recognition, Wireless Node

1. INTRODUCTION

Home automation systems have become a quite remarkable area of interest for many construction companies, and thus it has become popular in the recent years due to its attractive offerings to customers (Massé 2012). These systems have been improving day after day and the variety of home automated applications is still on rise (Yuksekkaya, Kayalar, Ozcan & Alkar 2006). In an automated home, heating, cooling, lighting, various appliances, entertainment and security systems can be controlled automatically. In addition to that, these systems are able to reduce the energy consumption. In short, home automation systems offer safety, convenience, comfort and more efficient life to their users.

Wireless sensor and actuator nodes are used in many areas and one of them is home automation systems. If everything in an automated home would be controlled by cabled connections, the cabling complexity inside the house and the costs would be high. Moreover, the maintenance and repairing would be extremely difficult. At this point, using wireless sensor nodes would solve these issues. Nodes constitute a wireless network between themselves for communication. Sensors on the nodes collect information according to their functionality by listening to their environment. Using the sensor data and the control information from the network, the node performs its role in the system.

The home automation systems alone is still not convenient for people who has some special needs, such as, elderly and disabled. Besides, usage and understanding of the system's default control panel can generally be complicated for some users. Integrating speech recognition to home automation systems provides extra easiness and comfort by the means of usage and it is a convenient way to control the home appliances for all kind of users (AlShu’eili, Gupta & Mukhopadhyay 2011).

As the name implies, the speech recognition technology allows computers to translate speech in pure audio form to a computer understandable form or even to text. In today's technology, not all the spoken words are understood 100% correctly. This makes speech

recognition improper for safety critical applications. However, in a home environment, the commands used to control home devices and appliances usually are short sentences and only contain a certain amount of words, making speech recognition a feasible solution (Zeng, Fapojuwo & Davies 2006).

The aim of this thesis is to design and implement a basic wireless home automation system, controlled by human speech. The practical demonstration of this work contains two wireless sensor nodes. One of them is connected to the computer, and the other node controls a motor and light by using the information coming from the first node. For its easiness and simplicity, TI's eZ430-RF2500 model wireless sensor nodes are chosen to demonstrate the control mechanism applicable to real home environment. A servo motor and a relay circuit to control the light are represented in a real home environment. When a user speaks a previously defined command word, the speech recognition system converts the speech to a computer understandable form. Regarding to the extracted result from the speech, the computer sends a serial command to the node attached to it. Then this node forwards the command to the other node using radio frequency (RF). Since the successful communication is provided, the servo motor can be implemented to control the heating or irrigation systems, and the home lighting can be switched on or off remotely.

The remaining chapters of the thesis is structured as follows; in chapter 2 and 3, some basic information is provided about home automation, speech recognition and wireless sensor nodes. After the theory and background information, chapter 4 and 5 describe the hardware and software parts of the control system. Then, the experiments and results are presented in chapter 6. Finally, chapter 7 concludes the thesis.

2. THEORY AND BACKGROUND INFORMATION

2.1. Serial Peripheral Interface

The serial peripheral interface (SPI) is a synchronous serial data link protocol developed by Motorola for hardware or firmware communications. Later on, this protocol has been adopted by other companies in the industry. SPI allows two devices to communicate with each other simultaneously in a full duplex operation. A device operating with SPI can have master/slave mode for data communication. Mostly, the SPI is used between a central processing unit (CPU) and peripheral devices. However, it can connect two microcontrollers to each other or a microcontroller to an external device. SPI is also known as four wire serial bus.

Full duplex mode means that the signals carrying data go in both directions. While the master is sending data to the slave device, the slave device also outputs its data simultaneously to the master side. SPI can support data rates up to 10Mbps. In an SPI bus usually there is only one master device and multiple slave devices but in some rarely seen configurations there can be multiple master devices. In those situations a multi-master protocol is used.

Due to the high speed communication of the SPI bus, the bus lines can not be too long since the reactance of the lines increase too much. For that reason the SPI bus is usually used only on the printed circuit board (PCB) and not for external connections going outside of the PCB. However, the speed of the SPI bus is configurable and some designers see no harm in operating at a lower speed for connections which extends out of the PCB.

The peripherals operating with SPI can be memory modules like EEPROM and flash memory, sensors like light sensors and pressure sensors, real time clocks, analog to digital converter (ADC) or digital to analog converter (DAC) devices, signal mixers, controller area network (CAN) controllers, potentiometers, liquid-crystal display (LCD)

displays or touchscreens, universal asynchronous receiver/transmitter (UART) or universal serial bus (USB) interfaces, or even amplifiers (EE Herald, 2013).

2.1.1. SPI signal lines for data and control

SPI uses 4 signal lines:

1. Serial Clock (SCLK or SCK) – SCLK is the clock signal generated by the master to synchronize data transfers.
2. Master Out Slave In (MOSI) – This is the data output line from master to slave.
3. Master In Slave Out (MISO) – This is the data output line from slave to master.
4. Slave Select (SS or SSEL) – SS is used to select individual slave/peripheral devices. Master device drives this line. The signal is connected to CS (Chip Select) input of the slave device. The SS/CS is usually an active low polarity.

Different manufacturers tend to use different notations for those signals. Some may call MOSI as Serial Data Output (SDO) and MISO as Serial Data Input (SDI) . MOSI and MISO can be grouped as data lines and the other two (SS and SCLK) as control lines.

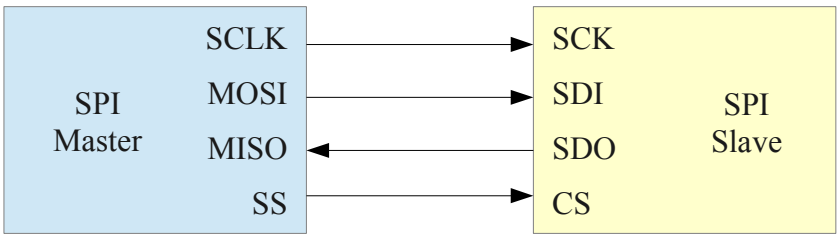


Figure 1. Single master and single slave implementation.

In an ordinary SPI bus there is only one master, and one or more slaves (see **Figure 1**). The master device controls the data flow in the following way. First of all, the master should configure itself and determine a clock rate that is lower than the slave device supports. To start the communication, the master device selects the slave device to that

wants to communicate by driving the corresponding SSEL line to low. At this point, the slave device becomes aware that the master device wants to talk to it. Then the master device starts to generate the clock signal. The data bits are exchanged in each clock pulse. The slave device is responsible of reading one bit of information in every clock cycle from the MOSI line, and it simultaneously needs to output a data bit to the master device on the MISO line. During that information exchange process, the master is also doing the similar thing. It is reading and writing data bits to its corresponding lines.

Every clock pulse that the master generates is received by all the slave devices on the bus whether they are selected or not. A non selected device will ignore all of the clock signals it receives. The data transfer may be in 8 bit bursts or continuous transfer. Some slave devices require that after receiving 8 bits of information, its chip select pin should be driven high for storing the data inside its shift register. On the other hand, some devices have internal data handling capabilities that longer transmissions will still be received and understood correctly although the SS pin stays low during the entire transfer.

All of those 4 pins may not be present in every SPI compatible device. For example an ADC may not require a MOSI line so it may feature only 3 of those SPI lines. In case the microcontroller needs to talk with only one device, that slave device's CS pin may be grounded since the communication line is already dedicated to that specific device. When communicating with multiple slave devices an independent SSEL signal is needed for each of the slaves.

2.1.2. Importance of the SPI frame format

The clock polarity (CPOL) and the clock phase (CPHA) of the SPI defines when the data is sampled and the polarity of the clock signal (see **Figure 2**). In addition to clock frequency, the master device needs to know the SPI protocol of the slave device and configure itself accordingly to be able to establish a reliable communication. In case the master device is not properly configured, it will send and receive wrong (shifted, inverted or totally corrupt) data.

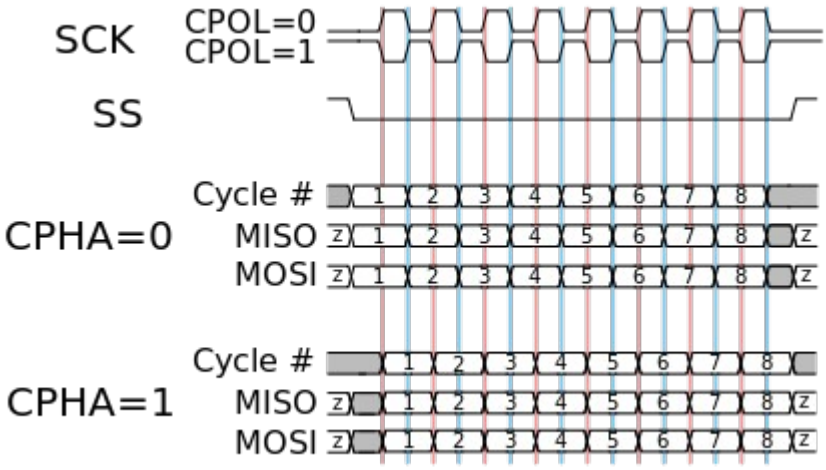


Figure 2. A timing diagram showing clock polarity and phase (Wikipedia).

Table 1 below lists the possible configurations.

Table 1. Modes of polarity and phase.

SPI Mode	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1

Advantages of SPI:

- Full duplex communication.
- High throughput.
- Not limited to 8-bit words.
- Simple hardware interfacing.
- Slaves use the master's clock, and don't need precision oscillators.
- Transceivers are not needed.

- At most one "unique" bus signal per device (CS); all others are shared.

Disadvantages of SPI:

- Requires more pins on IC packages than I²C.
- Separate chip select signals are required on shared busses.
- No hardware flow control.
- No slave acknowledgment (except the slave features additional answering method).
- Multiple frame formats.
- Shorter bus distances compared to RS-232, RS-485, or CAN. (EE Herald, 2013.)

2.2. Hidden Markov Model

It is important to grasp the idea behind the Hidden Markov Model (HMM) in order to understand the configuration options of the speech recognition software. The success ratio of the recognition process depends on how well the configuration fits the application. For that reason, this section provides information about the Hidden Markov Model.

The purpose of the Automatic Speech Recognition (ASR) is to find out the sequence of words which represents a linguistic message in a given acoustic data. ASR uses some special techniques to construct the statistical model of the speech. One of those techniques is the HMM (Rajman 2007: 288–297).

The term 'hidden' is used because in HMM, the underlying stochastic process is not observable, but it effects an observable sequence of events.

Sequential signals like speech are not stationary signals but HMM pretends that they are stationary by dividing the signal into smaller pieces. These pieces can be considered as piecewise stationary.

In other words, the sequence $X = x_1^N$ is modeled as a sequence of discrete stationary states $Q = \{q_1, \ldots, q_k, \ldots, q_K\}$ where there is an instantaneous transition between these states. In this case, a HMM is defined as a stochastic finite state automata which has a certain topology. This topology is strictly from left to right for speech data. A simple HMM example is represented in **Figure 3**. In speech recognition HMM is assumed to be a model of a word of phoneme that is made up of three stationary parts.

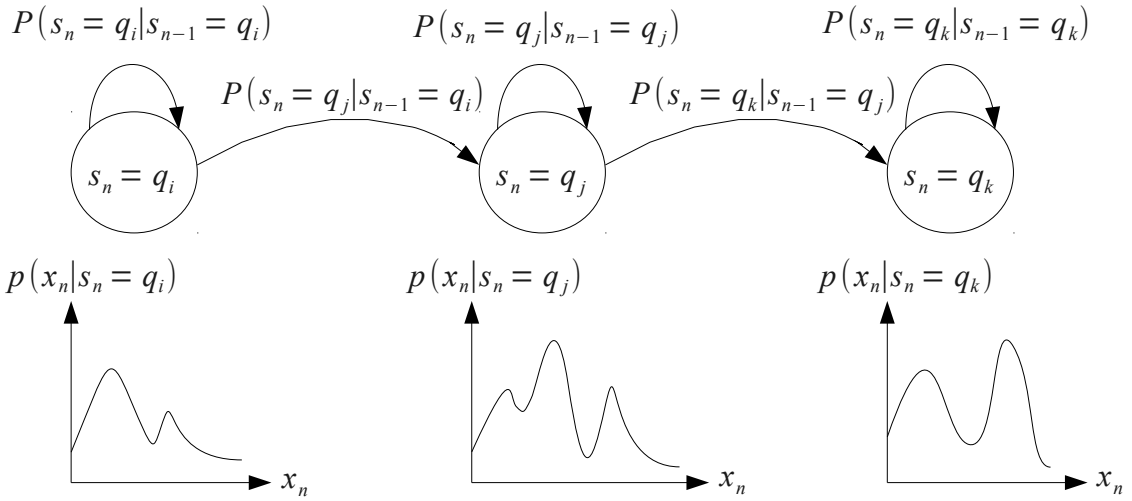


Figure 3. A schematic of a three state, left-to-right HMM.

Let us denote the probability that the observed vector sequence X is produced from the Markov model M as $P(X|M,\Theta)$. After the HMM topology is defined, the main criteria for training and decoding is based on this probability.

For a more convenient notation, q_k^n is used in this thesis to mention the state q_k is visited at the time n $\{s_n = q_k\}$ and in **Table 2**, some explanation about parameters and sets which are used in this section are given.

Table 2. HMM automata that can be used to process sequences.

	HMM
States	$q_k \in Q$
Input symbols	--
Output symbols	$x_n = z_n \in Z$
Transition law	trans. probabs. $P(s_n \mid s_{n-1})$
Emission law	
Mealy	emission on transition $x_n = g(s_n,\ s_{n-1})$, $P(x_n \mid s_{n-1},\ s_n)$
Moore	emission on state $x_n = g(s_n)$, $P(x_n \mid s_n)$
Training Methodology	EM Viterbi
Training Criterion	Maximum Likelihood

2.2.1. Full likelihood

For an observation sequence X, and the model M, the full likelihood $P(X|M)$ can be calculated as the sum of all possible paths Q having the length of N in M.

$$P(X|M) = \sum_{l=1}^{L} P(q_l^n, X|M), \forall n \in [1, N] \quad (1)$$

Here each term of the sum represents the probability that X is generated by M while visiting the state q_l at time n by assuming the observations are conditionally independent. This equation can be decomposed as:

$$P(q_l^n, X|M) = P(q_l^n, X_1^n|M)\, P(M_{n+1}^N | q_l^n, X_1^n, M) \quad (2)$$

Here the X_m^n represents the observed sequence $\{x_m, \ldots, x_n\}$.

Full likelihood $P(X|M)$ is the product of two probabilities. Forward probabilities and backward probabilities.

2.2.2. Forward probabilities

The equation:

$$\alpha_n(l|M) = P\left(q_l^n, X_1^n \middle| M\right) \tag{3}$$

represents the probability that the subsequence X_1^n has already been generated by the model M while being in the state q_l at time n. This probability can directly be rewritten as:

$$\begin{aligned} P\left(q_l^n, X_1^n \middle| M\right) &= \sum_{k=1}^{L} P\left(q_k^{n-1}, q_l^n, X_1^{n-1}, x_n \middle| M\right) \\ &= \sum_{k=1}^{L} P\left(q_k^{n-1}, X_1^{n-1} \middle| M\right) P\left(q_l^n, x_n \middle| q_k^{n-1}, X_1^{n-1}, M\right) \end{aligned} \tag{4}$$

and can be estimated through the forward recurrence:

$$\alpha_n(l|M) = \sum_k \alpha_{n-1}(k|M) P\left(q_l^n, x_n \middle| q_k^{n-1}, X_1^{n-1}, M\right) \tag{5}$$

Here the sum over k covers all possible predecessor states q_k of q_l. Initialization can be given as:

$$\alpha_1(l|M) = P_{Il}(M) \tag{6}$$

Here the right part means the initial state distribution for model M, and it can be

represented by:

$$\Pi = \left\{ P_{\Pi}(M), \forall\, l = 1, \ldots, L \right\} \tag{7}$$

Now it is assumed that the model becomes a Moore automaton, and a first order Markov model, the forward recurrences can be expressed as:

$$\alpha_n(l|M) = P\left(x_n \middle| q_l^n\right) \sum_k \alpha_{n-1}(k)\, p\left(q_l^n \middle| q_k^n\right) \tag{8}$$

This is applied over all possible values of n and l until the final state q_F is reached and n becomes $N+1$, resulting in:

$$P(X|M, \Theta) = \alpha_{N+1}(F|M) \tag{9}$$

2.2.3. Backward probabilities

The probability that the model M will generate the remaining part of the sequence X_{n+1}^N by starting from the current state q_l at the time n can be written as:

$$\beta_n(l|M) = P\left(X_{n+1}^N \middle| q_l^n, X_1^n, M\right) \tag{10}$$

Using the same logic, this probability can also be estimated by using the backward recurrence:

$$\begin{aligned} \beta_n(l|M) &= P\left(X_{n+1}^N \middle| q_l^n, X_1^n, M\right) \\ &= \sum_k \beta_{n+1}(k|M)\, P\left(q_k^{n+1} \middle| q_l^n, M\right) P\left(x_{n+1} \middle| q_k\right) \end{aligned} \tag{11}$$

Here the sum over k covers all possible successor states of q_l. This equation has the same form as equation (5). The only difference is that this one proceeds backwards in

time.

The initialization for this recurrence is:

$$\beta_N(l|M) = P_{lF}(M) \tag{12}$$

Here the right side of the equation is the probability to jump to the final state q_F from q_l. By referring to equation (1), and having the definition of the α:

$$P(X|M) = \sum_{l=1}^{L} P\left(q_l^N, X_1^N \middle| M\right) = \sum_{l=1}^{L} \alpha_N(l|M) \tag{13}$$

This sum is generally applied to the states defined as the possible final states of model M. Full likelihood can be calculated as follows:

where I and F represents the set of possible initial and final states of M respectively.

$$\begin{aligned} P(X|M) &= \sum_{l=1}^{L} \sum_{n=1}^{N} \alpha_n(l|M)\beta_n(l|M) \\ &= \sum_{[F]} \alpha_N(F|M) \\ &= \sum_{[I]} \beta_0(I|M) \end{aligned} \tag{14}$$

2.2.4. Viterbi approximation

The 'sum' operator in equation (8) can be replaced by 'max' operator in order to find the most possible path with a length of N, which generates the sequence X. The equation is now turned into Viterbi recursion and it is represented as:

$$\bar{P}\left(X_1^n, q_l^n\right) = P\left(x_n \middle| q_l^n\right) \max_{[q_k]} \left\{ \bar{P}\left(X_1^{n-1}, q_k^{n-1}\right) P\left(q_l^n \middle| q_k^{n-1}\right) \right\} \tag{15}$$

where $\bar{P}\left(X_1^n, q_l^n\right)$ represents the probability that the partial observation sequence $X_1^n = x_1, x_2, \ldots, x_n$ is generated by having followed the most probable path while being in state q_l at the time n. The summation over q_k is applied to the set of possible predecessor states of q_l. The likelihood of the most probable path in M $\bar{P}(X|M,\Theta)$ is obtained at the end of the sequence and it is equal to $\bar{P}(x_1^N, F)$.

In Viterbi approximation, there are also some performance enhancing techniques for digital signal processing.

2.3. Available Modulation Options for eZ430-RF2500

In this section the modulation techniques available in eZ430-RF2500 are presented since they play an important role in the trade-off between the data rate and the Signal to Noise Ratio (SNR). In case the channel conditions are bad, the modulation technique can be switched to binary phase shift keying (BPSK) instead of quadrature phase shift keying (QPSK).

There are three basic modulation techniques for transforming digital data into analog form; amplitude shift keying (ASK), frequency shift keying (FSK), and phase shift keying (PSK) (Stallings 2007: 151–161). The resulting signal is centered on the carrier frequency in all these three cases. In addition to those basic types, eZ430-RF2500 also supports minimum shift keying (MSK).

Amplitude Shift Keying

ASK represents two binary values by using two different amplitudes in a signal. One of those values is zero and the other is one. In zero case, the signal has no amplitude and one is represented by the signals presence at a constant amplitude. The binary ASK can be represented as:

$$s(t) = \begin{cases} A\cos(2\pi f_c t) & binary\ \ 1 \\ 0 & binary\ \ 0 \end{cases} \tag{16}$$

ASK is quite vulnerable to sudden changes in gain and it is considered to be an inefficient modulation technique for wireless medium. It is typically used in optical fiber communication lines. For light emitting diode (LED) transmitters the equation (16) is valid so that one signal state can be represented by a light pulse and the other one with the absence of the light.

Frequency Shift Keying

FSK is mainly used in binary form (BFSK) where the two binary values are represented by using two closely spaced frequencies near the carrier frequency. For the binary case, the transmitted signal can be expressed as:

$$s(t) = \begin{cases} A\cos(2\pi f_1 t) & binary\ \ 1 \\ A\cos(2\pi f_2 t) & binary\ \ 0 \end{cases} \tag{17}$$

where f_1 and f_2 are typically offset from the carrier frequency f_c by equal but opposite amounts.

BFSK is more error prone than ASK. It is commonly used for high-frequency (3 to 30 MHz) radio transmissions. It can also be used at even higher frequencies on local area networks that use coaxial cable.

Phase Shift Keying

In PSK, the phase of the carrier signal is shifted regarding to the transmitted bit. The simplest PSK uses two phases corresponding to two binary values. This binary PSK is known as BPSK. Finally the transmitted signal can be represented as:

$$s(t) = \begin{Bmatrix} A\cos(2\pi f_c t) \\ A\cos(2\pi f_c t + \pi) \end{Bmatrix} = \begin{Bmatrix} A\cos(2\pi f_c t) & binary\ \ 1 \\ -A\cos(2\pi f_c t) & binary\ \ 0 \end{Bmatrix} \quad (18)$$

A phase shift of 180° (π) is equivalent to multiplying the sine wave by -1. For that reason the expressions in equation (18) can be used.

If a bit stream is to be transmitted and $d(t)$ is considered as a discrete function taking the values +1 and -1 for the bit values 1 and 0 respectively, then the transmitted signal can be defined as:

$$s_d(t) = Ad(t)\cos(2\pi f_c t) \quad (19)$$

Minimum Shift Keying

Minimum shift keying (MSK) is a form of modulation which provides superior bandwidth efficiency to BFSK with only a small decrease in error performance (Stallings 2005: 140–141). Multilevel frequency shift keying (MFSK) can be considered as a form of BFSK. For MFSK, the transmitted signal for one bit time is:

$$s(t) = \begin{Bmatrix} \sqrt{\frac{2E_b}{T_b}}\cos(2\pi f_1 t + \theta(0)) & binary\ \ 1 \\ \sqrt{\frac{2E_b}{T_b}}\cos(2\pi f_2 t + \theta(0)) & binary\ \ 0 \end{Bmatrix} \quad (20)$$

Here E_b is the transmitted signal energy per bit and T_b is the bit duration. The phase $\theta(0)$ denotes the value of the phase at time t = 0. It would be appropriate to mention that MSK is a form of FSK known as continuous-phase FSK (CPFSK), in which the phase is continuous during the transition from one bit time to another.

For MSK, the two frequencies satisfy the following equations:

$$f_1 = f_c + \frac{1}{4T_b} \tag{21}$$

$$f_2 = f_c - \frac{1}{4T_b} \tag{22}$$

The word 'minimum' implies that the spacing between the two frequencies is the minimum that can be used, permitting successful detection of the signal at the receiver. This is the reason for the term minimum in MSK.

For MSK, the carrier is multiplied by a sinusoidal function, as follows:

$$s(t) = I(t)\cos(\frac{\pi t}{2T_b})cos2\pi f_c t + Q(t-T_b)\sin(\frac{\pi t}{2T_b})\sin 2\pi f_c t \tag{23}$$

3. WIRELESS SENSOR NODES AND HOME AUTOMATION

3.1. Wireless Sensor Node

A wireless sensor node is an electronic device that consists of one or more sensors, a transceiver, and a battery. The sensors are used for collecting data from the surrounding environment depending on the sensor type. Some examples would be temperature, humidity, acceleration, orientation, light and proximity sensors. The battery powered transceiver then passes the measured information to another wireless sensor node that is connected to a central computer. Wireless sensor nodes are usually referred as 'nodes'.

A wireless sensor network (WSN) is a group of nodes organized as a cooperative network where nodes can talk and listen to each other. This communication organization can be used to circulate commands inside the network. For instance, a node which is located far away can receive a command from another node to measure the temperature, and send that measurement to a central computer as shown in **Figure 4.**

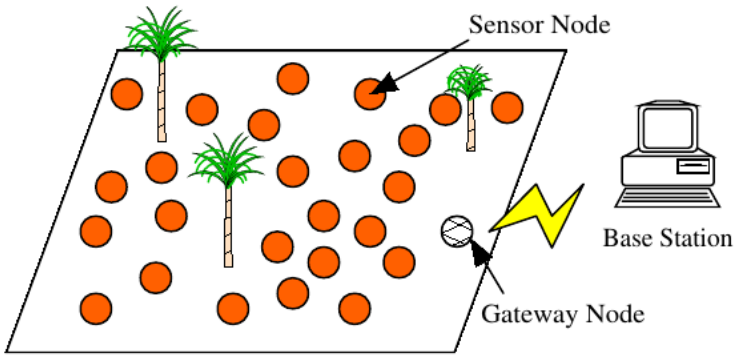


Figure 4. Wireless sensor network architecture example (Vieira, Cunha & Silva 2006).

These networks were firstly developed for military applications but today the applications are extended to industry as well as for environment monitoring (Dargie & Poellabauer 2010: 8). One important usage of wireless sensor networks is the natural

disaster monitoring. In case of a forest fire, the wireless sensor node that detects a fire and smoke, can send an alarm to the fire departments in advance.

3.2. Home Automation

Home automation is one of the fastest growing industries capable of altering the way that the human beings live. Some of those home automation systems target the needs for seeking more comfortable, sophisticated and luxury products. In addition it provides advantages for disabled or old people who may have special needs.

The category of the home automation applications is still evolving and thus there are multiple standards incompatible with each other. In order to get these systems work together may require deeper knowledge today, but once everything is standardized the setup times, costs and maintenance efforts are expected to reduce to much smaller amounts compared to today's systems. (Derene 2009.)

Home automation allows remote and automatic control of wide range of devices. These systems can also send alert messages to the mobile devices whenever there is something that needs attention at home such as water leaks of thievery. It is also capable of providing a control screen to the user that allows taking actions over the secure communication protocols. The application level can also control how a device should react and when. The users may be able to set schedules to specific events like watering the plants. The main advantages of home automation can be listed as convenience, safety and security, energy savings and entertainment.

Providing a remote control system and automating the home appliances have advantages in terms of time and convenience. The user can dim the lights inside the house while still sitting on the couch (Rand 2013), adjust the temperature from the bed, control the sound level of the audio devices and can set schedules for bathroom heating.

The safety advantages of these systems are also great. For example a water sensor can detect a possible water leak as soon as something goes wrong and can prevent costly damage repairs. A motion sensor can be connected to a lighting system so that the lights may go on immediately in case there is a movement detected within the house. In this situation it is also possible to alert the police, if desired.

The ability to set the personal preferences, using voice recognition to control the entertainment systems at home enhances the entertainment experience of the residents.

From the previous examples it can be seen that home automation technologies offer many solutions for controlled systems, such as; lighting, cameras, security systems and access control, home theater and entertainment, phone systems, thermostats, irrigation and cable and structured wiring (Harper 2003). This work provides a way to control the lighting, heating, and irrigation systems.

Lighting control:

This application allows the user to control the home lighting over the network independently from wherever he or she is located inside the house. The control is done with speech recognition by using external microphones located within the house. In this work, the lighting is represented by a light bulb, and the microphone is represented by an external microphone connected to a PC.

Heating:

A heating system inside the house can be controlled by a motor which is capable of receiving commands over the wireless sensor network. The wireless sensor device eZ430-RF2500 of the Texas Instruments already has an in-built temperature sensor on its microcontroller MSP430F2274. This sensor can be used to apply a closed loop heating control algorithm. Such an application has also a potential to be used for greenhouse and animal shelter heating systems in addition to automated homes.

Irrigation:

Home automation is not constrained to indoors. It can also be used in gardens for a more efficient plant life. The user may program a schedule and the rest can be handled by the irrigation system. Adding a rain sensor would provide a way to halt the system in case the weather is rainy. This leads to the advantage that tap water can be saved. In addition to those, irrigation systems can be combined with motion sensors to protect the garden from wild animals.

4. HARDWARE

4.1. System Overview

The home automation system provided in this thesis can be divided into two main parts. One of those parts is PC controlled and acts as a *command center*. This part receives the input speech. After the speech command is recognized it forwards a one byte command via USB to the wireless node which is connected the PC. From these, the single byte command will be transmitted to the wireless node belonging to the second part. The other part consists of a wireless sensor device with a battery box that controls various devices and acts as an *actuator part*.

Figure 5. Command center of the home automation system.

The command center part can be seen in **Figure 5**. Here there is a PC, an external microphone, and a wireless node. The microphone and the wireless sensor node are both

connected to the PC via USB interfaces. All the home automation systems are controlled over this PC by the software applications installed on it. The devices and the software applications used in this system are shortly explained further in this section regarding to their functionalities.

Logitech USB Desktop Microphone is used for passing the voice commands to the PC in digital format. The pure audio signal that is captured with the microphone is converted to digital format inside this microphone and the data is sent to the PC over the USB interface. On the PC, there is a speech recognition application which directly receives the data. The speech recognition application used in this work is 'Sphinx4'. This software is entirely written in Java and applies 'Hidden Markov Model' concept for converting the speech to a computer understandable form (Derbali, Jarrah & Wahid 2012).

Sphinx4 project is an open source software project developed by Carnegie Mellon University. This software and the speech recognition process is explained in detail in section 5.1 of this thesis. In the system shown in **Figure 5**, the voice command that arrives to the PC are best matched regarding to the words contained inside the word pool of the dictionary defined in the Java application. After the matching process, the estimated result is converted to a text. The steps after this point become easier since the command in a text format will only be processed. In this thesis it was decided to just send one character that represents the command ID to the wireless sensor node over the USB. The UART receiver of the MSP430F2274 immediately stores this value inside the memory.

In this thesis, the eZ430-RF2500T target board of the Texas Instruments is used as a wireless sensor node. One of those nodes is connected to PC via USB debugging interface and it is called the *access point* node. The function of this access point node is to forward the incoming PC commands to the other nodes over the RF connection. The node on the receiving side is called *end device*. Texas Instrument's Code Composer Studio is used as a development environment for embedded application development. At the same time, it is used for downloading and debugging the applications for both

nodes.

The information coming to the access point is immediately forwarded to the RF chip from the SPI interface. Detailed information about the SPI is provided in section 2.1 of the thesis. According to the settings in the software, the bytes received on the UART are sent via the radio to the end device. This explains how the command center part of the home automation system in this thesis works.

The actuator part of the system takes physical action based on the transmitted information over the wireless network. As illustrated in **Figure 6**, this part is generally a wireless sensor node connected to a servo motor, a lighting control board with a light bulb, and a battery for power requirements.

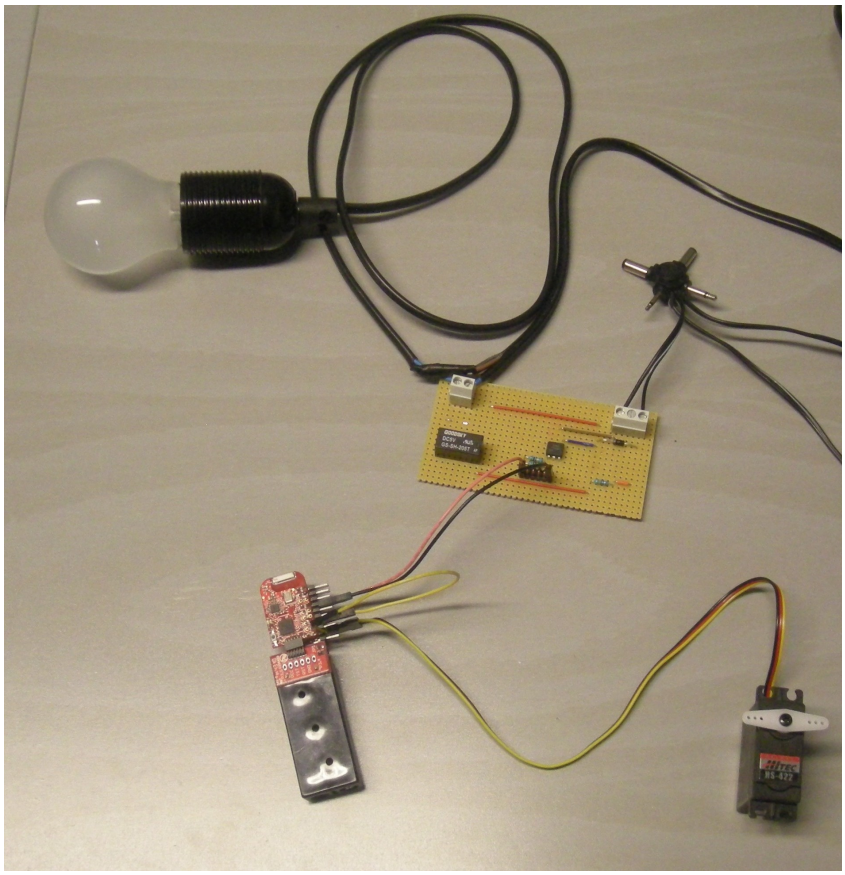


Figure 6. Actuator part of the home automation system.

As soon as the wireless data is received on the radio transceiver of the end device, it is directed to the MSP430F2274 microcontroller over the SPI interface. At this point the microcontroller checks if the received information is a valid command and if it is correct, it will take an action defined in the software.

The servo motor used in this work is Hitec HS-422 - Standard Deluxe Servo Motor. Combining this motor with wireless devices, a heating system or an irrigation system of a house can be controlled in a wireless manner. When the correct byte arrives to the microcontroller, the timer counter value is updated and the duty cycle of the servo motor is changed and the motor turns to the desired direction.

The lighting control board consists of a LTV4N35 optocoupler, a GS-SH-205T relay and a LED. The end device is attached to this board it has the capability to turn the light on or off with the wireless commands coming from the access point. In fact, the relay can switch much higher voltages and current as required from a LED. This means that a LED can easily be replaced by a home light. This summarizes the second part of the automation system.

It is now possible to describe how an example application implemented in this thesis works in reality. In order to test the functionality of the whole system several tests have been done. One of the tests has been done so that the user speaks to an external microphone saying "Motor Left". Then the speech recognition program processes the word as described further in section 5.1. The PC then sends the command byte '0x47' to the wireless node connected with the PC. From these the command will be sent over radio to the wireless node that controls several devices such as motor and light. After the reception the command is evaluated with a switch case statement which changes the duty cycle of the pulse width modulation (PWM) signal. Finally the motor turns to left independently from its previous position. In case this motor would be connected to a valve, it would switch the irrigation system on and off.

4.2. eZ430-RF2500 Wireless Development Tool

The eZ430-RF2500 development tool that is produced by Texas Instruments was used for the wireless communication between the *command center* part and the *actuator part.* The development tool includes two eZ430-RF2500T target boards, one eZ430-RF USB debugging interface and one AAA battery pack with expansion board, as shown in the **Figure 7**. The eZ430-RF2500 also features 21 available development pins, two general purpose digital input/output (GPIO) pins connected to green and red LEDs for visual feedback and an interruptible push button for user feedback.

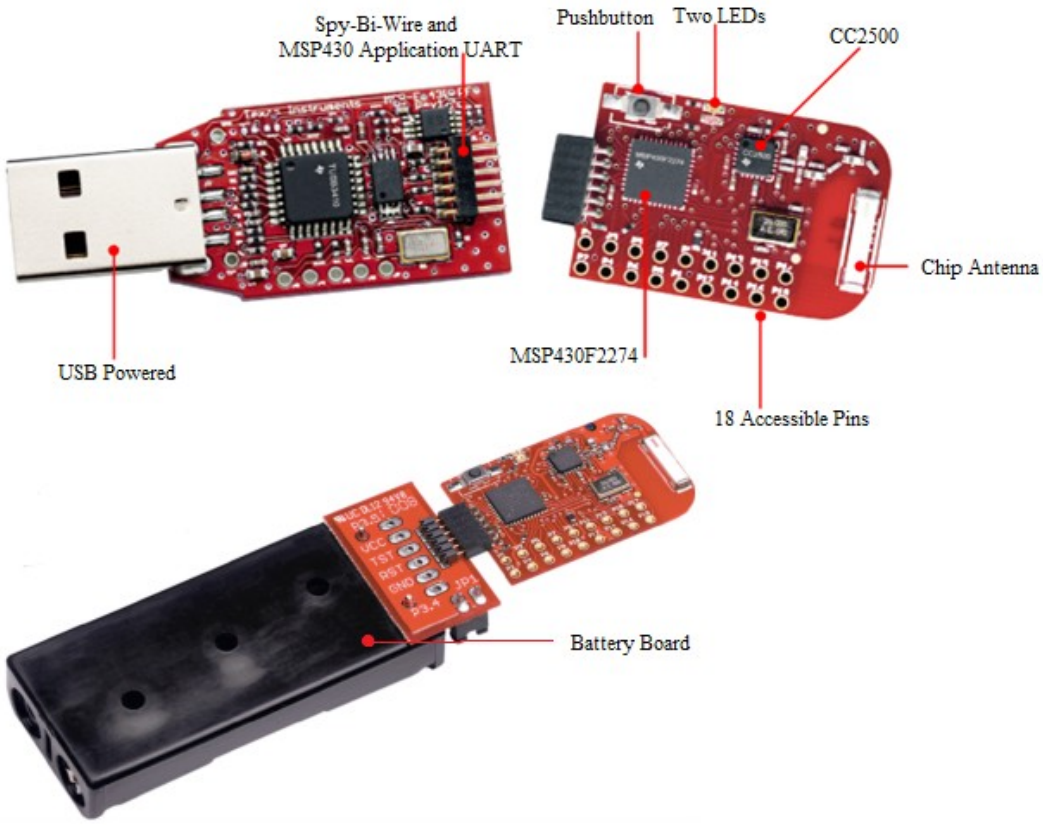


Figure 7. eZ430-RF2500 Wireless Development Tool (Texas Instruments Incorporated 2009).

The target board which was connected to the PC with the USB debugging interface is named as *access point*. It sends and receives data from PC using MSP430 application

UART as an out-of-the box wireless system. The other target board is named *end device* and it is connected with the battery board. End device is also connected with the motor and light control board.

Each target board has its own MSP430F2274 microcontroller and CC2500 2.4-GHz wireless transceiver. Also, most of MSP430F2274's pins can be accessible with pinouts on the boards. The functionalities of those pins are given in appendix 1.

MSP430F2274 microcontroller has been developed for ultra-low power applications and supports many low power operating modes. These modes allow the microcontroller to activate only the necessary hardware blocks inside it so that it can save power. Various clock sources can be used for many clocking hardware blocks so that user application can select which one suits his/her requirements the best. A good example applied in this thesis is the PWM usage and the RF communication. Here the clocking system is used so efficiently that the minimum power is used to generate the PWM using the timer. The experimental results about the current consumption are represented further in section 6.1. In the application, the microcontoller is always in sleep mode until an interrupt occurs in the SPI module. When a byte is received on the RF chip, the microcontroller wakes up, processes the command, updates the timer counter and goes back to sleep mode. All these things happen in a very short duration to maximize the battery life. While the microcontroller is in sleep mode, PWM is continuously generated to keep the motor position. The possible low power modes and the corresponding clocking information of the microcontroller are represented in the **Table 3**.

Table 3. Low power modes for MSP430F2274.

Mode	**CPU and Clocks Status**
Active	CPU is active, all enabled clocks are active.
LPM0	CPU, MCLK are disabled, SMCLK, ACLK are active.
LPM1	CPU, MCLK are disabled. DCO and DC generator are disabled if the DCO is not used for SMCLK. ACLK is active.
LPM2	CPU, MCLK, SMCLK, DCO are disabled. DC generator remains enabled. ACLK is active.
LPM3	CPU, MCLK, SMCLK, DCO are disabled. DC generator disabled. ACLK is active.
LPM4	CPU and all clocks disabled.

4.2.1. CC2500

CC2500 is a low power transceiver chip. It means that, the chip acts like both transmitter and receiver with low current consumption. It is a part of the eZ430-RF2500 wireless development kit and communicates with MSP430F2274 chip via SPI interface. The frequency range is from 2400MHz to 2483.5MHz.

Furthermore, the data rate is configurable between 1.2 and 500kBaud. Thus, the current consumption can be reduced for applications which do not require a high speed transmission by reducing the data rate. In addition, the packet error rate is maximum 1% when using a baudrate of 2.4 kBaud.

CC2500 supports easy packet handling, data buffering, burst transmission, clear channel assessment, link quality indication, and wake-on-radio. It has 64 byte transmission (Tx) and reception (Rx) first in first out (FIFO). Additional information can be found in

appendix 2.

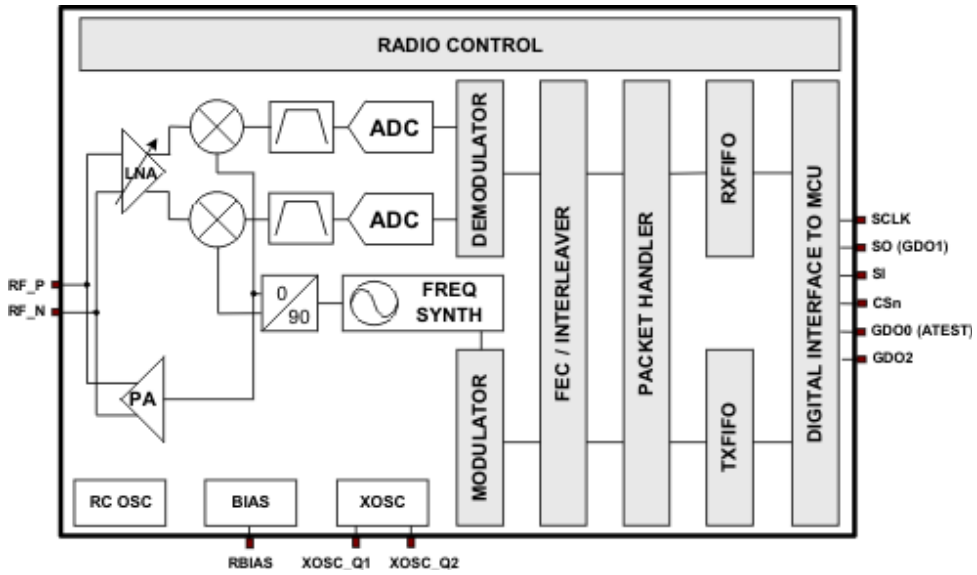


Figure 8. Block Diagram of CC2500 (Texas Instruments Incorporated 2011).

Figure 8 shows the block diagram for the CC2500. The CC2500 features a low-intermediate frequency (IF) receiver. This received RF signal is then amplified by the low- noise amplifier (LNA). After that, it is down-converted to I and Q components to the intermediate frequency. At IF the I/Q signals are digitized by the ADCs. The automatic gain control (AGC), fine channel filtering, demodulation, bit/packet synchronization are performed on digital form of the signal.

The transmitter of the CC2500 is based on direct synthesis of the RF frequency. A crystal connected to XOSC_Q1 and XOSC_Q2 provides reference frequency for the synthesizer as well as the clocks of the receiver ADCs and the digital parts.

The SPI interface is used for chip's configuration and data buffer access. The CC2500 also includes support for channel configuration, packet handling, and data buffering configurations.

Modulation Formats

CC2500 supports amplitude, frequency and phase shift modulation formats as described in section 2.3. The desired modulation format is set in the MDMCFG2.MOD_FORMAT register.

4.2.2. SPI communication with CC2500

The CC2500 chip is the slave device of the SPI link. The background information about the SPI is given previously in section 2.1. The configurations for the wireless communication are loaded to the chip from the MSP430F2274 so that only one application software is required. For easy usage, Texas Instruments has provided a code library for the developers. This work takes advantage of this library and ready functions inside it. Each of those functions are explained briefly in the **Table 4**.

Table 4. SPI register functions provided by the CC2500 library.

Functions	**Descriptions**
`void TI_CC_SPISetup(void)`	Configures the assigned interface to function as a SPI port and initializes it.
`void TI_CC_SPIWriteReg(char addr, char value)`	Writes "value" to a single configuration register at address "addr".
`void TI_CC_SPIWriteBurstReg(char addr, char *buffer, char count)`	Writes values to multiple configuration registers, the first register being at address "addr". First data byte is at "buffer", and both addr and buffer are incremented sequentially (within the CC2500 and MSP430F2274, respectively) until "count" writes have been performed.
`char TI_CC_SPIReadReg(char addr)`	Reads a single configuration register at address "addr" and returns the value read.
`void TI_CC_SPIReadBurstReg(char addr, char *buffer, char count)`	Reads multiple configuration registers, the first register being at address "addr". Values read are deposited sequentially starting at address "buffer", until "count" registers have been read.
`char TI_CC_SPIReadStatus(char addr)`	Special read function for reading status registers. Reads status register at register "addr" and returns the value read.
`void TI_CC_SPIStrobe(char strobe)`	Special write function for writing to command strobe registers. Writes to the strobe at address "addr".

For SPI connection, the RF chip has a clock input (SCLK), data output (SO), chip select (CSn) and data input (SI) pin. The pin connection between the CC2500 and MSP430F2274 is represented in the **Figure 9**. Also, in appendix 3 there is a schematic design of SPI link between MSP430F2274 and CC2500.

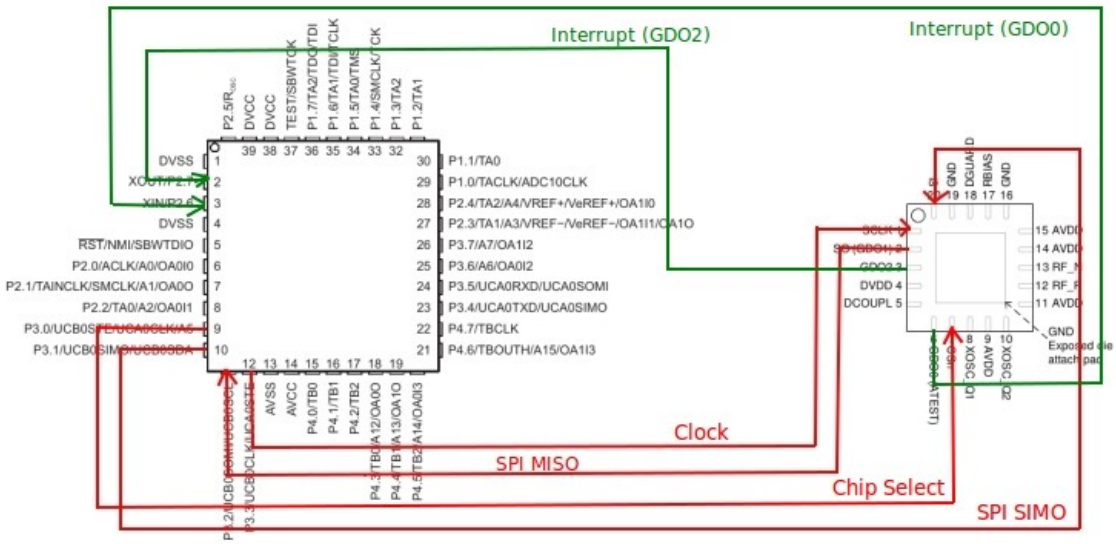


Figure 9. SPI and interrupt pin connections between the CC2500 and the MSP430F2274 (Texas Instruments Incorporated 2011).

In addition to the SPI connection, there are also interrupt pins connected between these two chips. These interrupt pins allow the ultra low power modes to be utilized. As soon as the CC2500 has some data to pass to the microcontroller, it sets an interrupt pin to logic high, so that the MSP430F2274 can enter directly to the interrupt service routine (ISR) while it was asleep.

4.3. Servo Motor Control

Hitec HS-422 - Standard Deluxe Servo Motor has been chosen as a servo motor that represent the heating and irrigation systems' motor. The motor operates with a voltage between 4.8V–6V. In **Figure 10** the used servo motor is illustrated. The motor is controlled with a 3.3V peak square wave pulse. Different duty cycles in the range of 0.9ms to 2.1ms are used to control the servo motor. The time period for each pulse is 20ms. More information about the motor can be found in appendix 4.

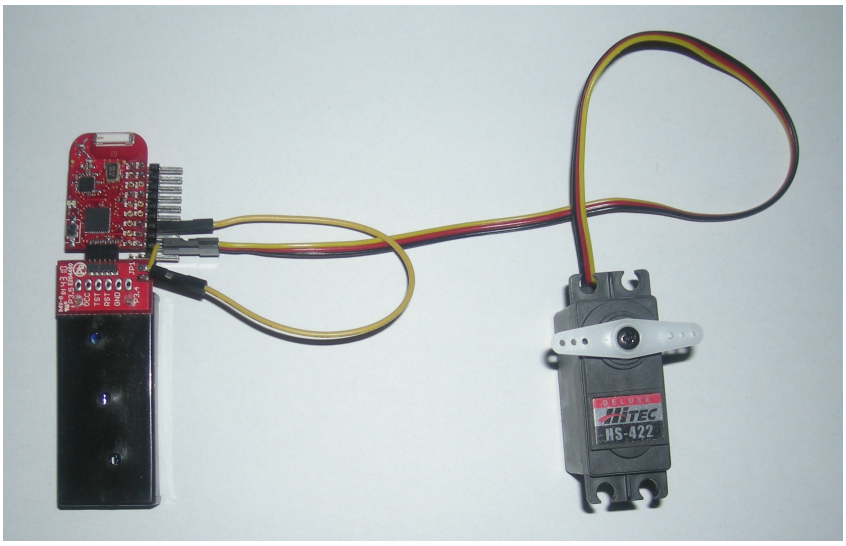


Figure 10. Hitec HS-422 servo motor connected with the end device.

Timer_A of MSP430F2274 was used as a PWM generator for the motor control. The timer has a 16 bit timer register and it can be increased or decreased to achieve the desired duty cycle. The clock source was selected as 1.2MHz SMCLK without any prescaler. The timer was configured to count from zero to the value of the compare register repeatedly. The value of the compare register was calculated to attain a 20ms timer period as follows:

Firstly, the duration of one clock cycle (T) was found as:

$$T = \frac{1}{f} = \frac{1}{1.2 \cdot 10^{6}} \Rightarrow T = 0.833\,\mu s \tag{24}$$

Then, the register value was calculated as:

$$\frac{2 \cdot 10^{-2}}{8.33 \cdot 10^{-7}} \approx 24009 \tag{25}$$

Pulse duration in a timer period is directly related with the angle of the motor position. The pulse durations of Hitec motor should be 0.9ms for 0 degree, 1.5ms for 90 degrees and 2.1ms for 180 degrees. The timer compare values can be calculated by using the equations (24) and (25). The correct values for 0.9ms, 1.6ms and 2.1ms have been calculated as 1080, 1800 and 2520 respectively. The values for any angle can be calculated by using the same equations as well.

Three pins on the end device were used for motor connection. As it can be seen in **Figure 11,** these pins are: P1 for ground, P2 for power and P6 for the PWM signal connection.

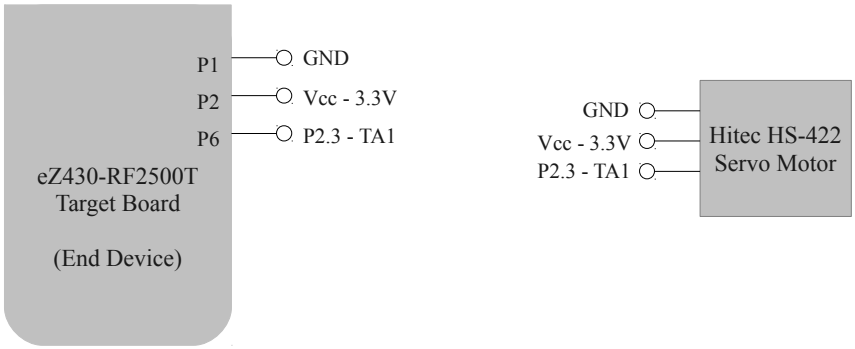


Figure 11. Pin connections between the end device and the motor.

4.4. Lighting Control Board

In the context of this thesis, lighting control for home automation has also been implemented in addition to the servo motor control (**Figure 12**). In order to realize this, a circuit has been designed with an optocoupler and a relay. The relay on the circuit is capable of switching up to 2A current. The general functionality of the light control system and the circuit details are explained briefly in this section.

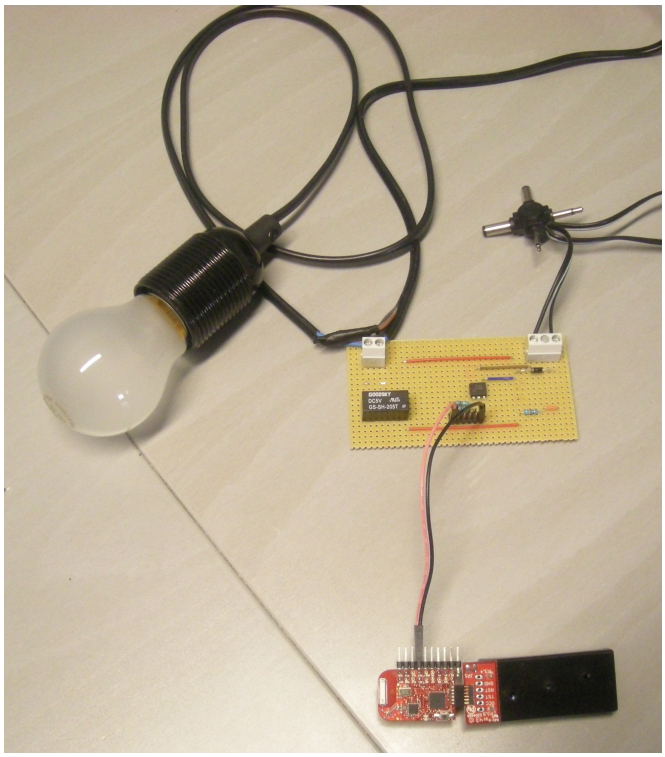


Figure 12. Lighting control board connected with the end device and the light bulb.

The purpose of this circuit is to amplify the 3.3V output signal of the eZ430-RF2500 wireless node up to 220V for switching. Since this is a great amount of amplification, a direct connection between the light relay and the microcontroller is not possible. Even if it would be possible, the microcontroller needs to be isolated from high voltages and Electromotive Force (EMF) of the relay. For those two reasons, an optocoupler circuit was placed between the microcontoller and the relay part.

The LTV4N35 general purpose type optocoupler has been used in the optocoupler circuitry. Inside an optocoupler there are two different parts. On the input side, there is a LED that acts as an optical transmitter and on the other side there is a phototransistor or a light-triggered triac which acts as a optical receiver. Between those two, there is a transparent barrier which prevents the electrical current but allows the light propagation. When there is a voltage on the input side, the internal LED becomes active and triggers the phototransistor receiver instantly. This allows the current to flow along the output

side.

The current flowing through the optocoupler output goes through the relay circuit to activate it. The manufacturer code of the relay used in this work is GS-SH-205T. Relays work as electrically controlled switches. They allow controlling the devices that work with higher power by using a lower power switching. A relay consists of two independent electrical circuits (Sullivan 2013). One of them contains just an electromagnet and the other side contains a switch, that can be activated by this electromagnet. When the current flows throughout the input side it moves the electromagnet. Then the electromagnet pulls down the armature in the relay and the second circuit is closed. Thus, the second circuit is used for opening the light in this work. When the electromagnet is deenergized, the armature is pulled up and the second circuit is opened.

The schematic of the designed circuit is represented in the **Figure 13**. A GPIO pin of the MSP430F22274 is connected to the optocoupler's input over a 150Ω resistor. This GPIO pin was configured to be initially in low state and it is represented by the switch J1 in the schematic design**.** After the speech recognition process, the 'Light Open' command arrives to the MSP430F2274, and the GPIO pin is set to a logic high level. Thus the LED inside the optocoupler is switched on by the 3.3V signal. The optical receiver senses the light emission and activates the circuit on its own side. At this moment the relay becomes energized and the internal electromagnet is activated. The second circuit inside the relay is triggered by this magnetic field and the light is turned on. The similar process happens when the 'Light Close' command is received by the microcontroller. Finally the light is turned off when the relay is deenergized.

The functionality of D1 in this circuit is only for protection purposes. As soon as the relay is deenergized, the electromagnet tends to produce a high voltage spike. In case the diode would be missing, the optocoupler's output might be in danger. Since the diode will pass all the voltages over 0.7 volts over itself, the optocoupler will be protected.

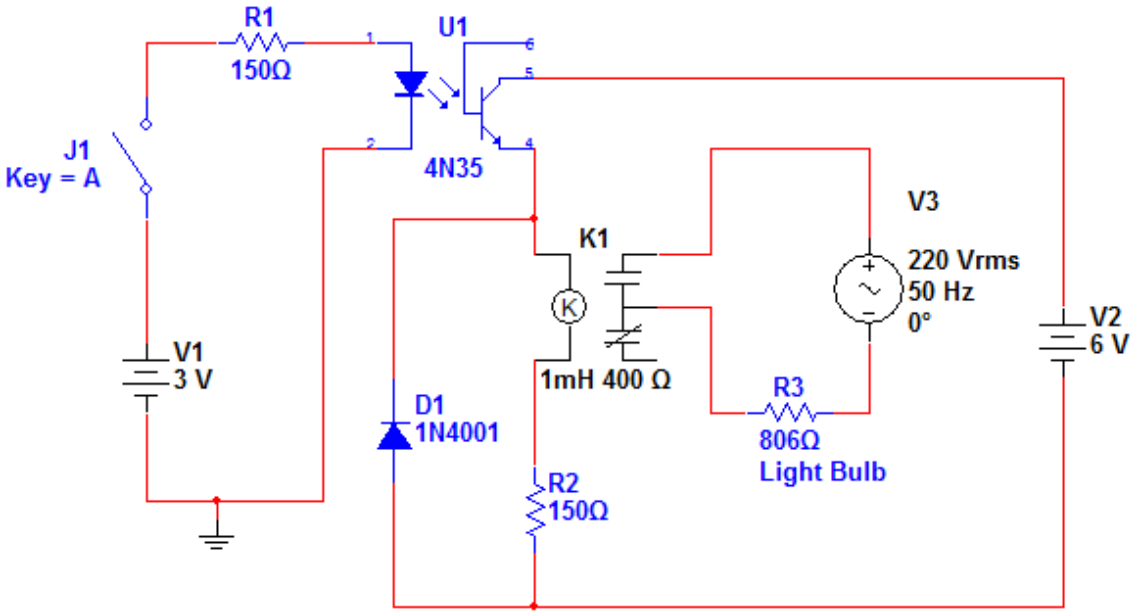


Figure 13. Schematic design of the lighting control board.

5. SOFTWARE

As described previously in section 4.1, the entire system can be divided into two main parts. The command center and the actuator part. Both of these run their own software. On the command center side there is a Java software for speech recognition and the access point software for wireless communication. The actuator wireless device is called end point and has a different functionality than the access point. Its functionality is explained in section. This chapter describes all those software with the explanations of the algorithms and presents flowcharts.

5.1. Recognizing the Speech

The Java software on the PC continuously gets the audio data from an external microphone and performs the recognition. Based on these results it sends a message to the access point device for relaying the message over RF to the end device. In this section the Java software and the access point software are explained.

The Java application depends on a speech recognition software tool called CMU Sphinx (CMU Sphinx 2013). CMU Sphinx offers two alternative solutions for speech recognition. The first one is Sphinx which is targeted for devices with high computation capabilities. The second one is called PocketSphinx that is more suitable for embedded computers or hand-held devices like mobile phones or tablets.

In this thesis Sphinx 4 is used to perform the speech recognition since the computation power in the PC is sufficient. This open source project allows the users to download the code into their computers and setup their own projects based on that. Although Sphinx offers a very convenient way to recognize speech, it must be configured properly since the Hidden Markov Model based algorithms rely on different parameters and configuration options.

5.1.1. Sphinx 4

The sphinx software uses 'phones', 'diphones' and 'senones' to understand the speech structure. Speech is a continuous stream which involves both dynamic and rather stable states. Phones are the classes of sounds defined in the sequence of states. Understanding the words is performed based on phones but that is not the only criteria in this decision.

The acoustic waveform properties of a phone can greatly vary depending on its context, speaker, speech style and so on. These transition regions are called diphones which are parts between two consecutive phones. The transitions between words are more informative than stable regions.

Senones are multiple phones considered in context. Senones dependence on context is more complicated than just preceding and following the phone relationship. Often the senones are made of three or four phones. Apart from that, a senone contains HMM stream emission probabilities.

There are also subphonetic units representing sub-states of a phone. The first part depends on its preceding phone, the middle one is the stable one and the latter part depends on the subsequent phone.

As stated before, Sphinx uses phones, diphones and subphones to recognize a word. If the considered language has 40 phones and each word in average is made of 7 phones, there should be 40^7 words to constructed. But a person who speaks a language uses a maximum of 20.000 words, so there is a certain word pool making the recognition more feasible for a certain language. In this work the word pool is limited in order to minimize the wrong decision probabilities.

Sphinx 4 configuration:

Sphinx 4 uses a configuration file (Lamere Kwok, Gouvea, Raj, Singh, Walker & Wolf 2003). This file contains the following configurations:

- Word recognizer configuration
- Decoder configuration
- Linguist configuration
- Grammar configuration (Here the expected words/sentences are specified)
- Dictionary configuration (This is the pool of words and their spellings)
- Acoustic model configuration (Here the sample rate is specified)
- Unit manager configuration
- Frontend configuration
- Monitors

The configuration file is explained in detail on CMU Sphinx's web page. The recognition process is given in **Figure 14**.

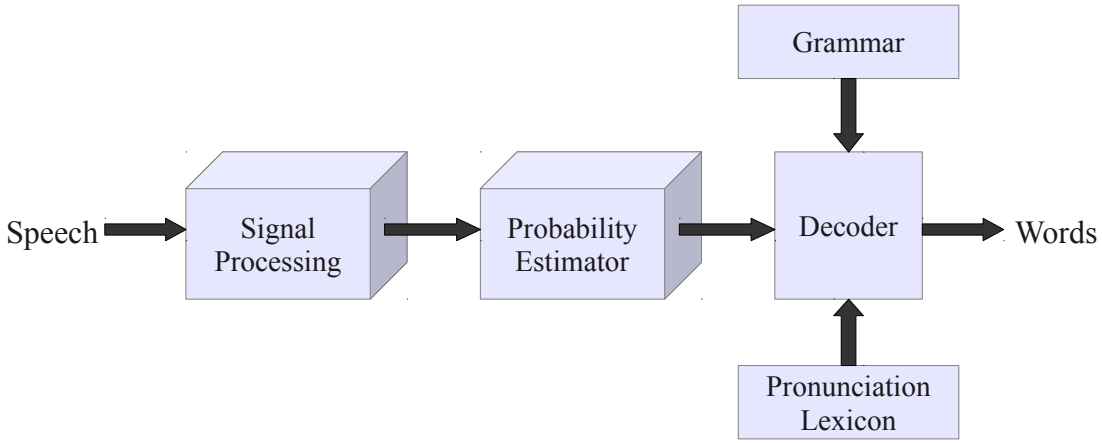


Figure 14. Block diagram of the speech recognition system.

The software basically takes a waveform, and then splits it into pieces by silences, then it tries to recognize what is said in each of the non-silent parts. This process takes the combination of all possible words and it tries to match them with the audio stream. The algorithm tries to choose the best possible algorithm.

One of the concepts that lies behind the Sphinx recognition process is 'features'. These features are simply numbers extracted from a speech frame. But these numbers have a specific meaning and each number is used somewhere in the algorithm. Each speech frame is 10ms long and 39 features are extracted from each frame. Extracted features are called 'feature vector'. A quite heavy computation process needs to be done to extract the feature vector. The extraction methods from an audio stream are still subject to research.

Another concept of the audio streams is the 'model'. It is basically a mathematical object, consisting of common attributes of the spoken word. In practice the model is a most probable feature vector that represents a word. There are many practical issues related to models. The model should fit to its corresponding word in practice but this is never completely true. Currently, the ways to improve the models to fit into the practice, and to eliminate the effects of the changing conditions are under research.

Using the features and the models, Sphinx enters to a matching process. Comparing all the feature vectors with all models would certainly take a huge amount of time, so that the recognition process would be totally unusable. But at this point the search is optimized by many enhancements. The use of HMM allows the software to determine the next possible frames and reduces the search time.

Models:

In speech recognition, three speech structure models are used. Acoustic model, phonetic dictionary, and language model. The speech recognition engine is composed of these three entities.

An acoustic model contains the acoustic properties of a senone. Some acoustic models contain context-independent models, like feature vectors for each phone. On the other hand some models depend on the context. Those models are built from senones with context.

Phonetic dictionary provides mapping from words to phones. Mapping is not always accurate because sometimes there are multiple pronunciations for some words and sometimes there are such pronunciations corresponding to different word mappings. The dictionary is not the only mapper from words to phones. This process is usually done with the aid of a machine learning algorithm that has a complex functionality.

A language model restricts the word pool to certain possibilities. Here the model takes the advantage of previously recognized words to determine what may be the next possible word in sequence. This method significantly decreases the matching process by eliminating words which are not in the dictionary.

The most common language models used in practice is n-gram language models. These models contain a statistic of word sequences. The second model type of the models is the finite state language models that defines the speech sequences by finite state automation or sometimes with weights. Achieving a good accuracy depends on the success of the search space restriction mechanism. Search space restriction is very important both in terms of search speed and accuracy of the results.

5.1.2. Java application

On the application level, the software firstly reads the defined configuration parameters from an XML file. This file contains all the necessary predefined parameters and where to search for the dictionary information to create a word pool. The following line is used to create a configuration manager and to read the content of the XML file:

```
cm = new
ConfigurationManager(MotorControl.class.getResource("motorcontrol.conf
ig.xml"));
```

After that, the microphone must be accessed. Sphinx 4 already defines a microphone class so the user can easily create a microphone object as:

```
Microphone microphone = (Microphone) cm.lookup("microphone");
```

When the initialization is over, the software will print the following line to indicate the possible meaningful words to the user:

```
System.out.println("Say: ( Motor | Light ) ( Left | Right | Open |
Close )");
```

Here the user can choose one word from each group. For example 'Motor Left', 'Light Open' or 'Light Close' etc. In case the user would say 'Close Left', the software will ignore this and continue to listen.

Whenever a meaningful combination is detected the software opens a file descriptor on the USB device and sends a message. The code section which does that is represented below:

```
try
{
      FileOutputStream out = new FileOutputStream("/dev/ttyACM0");
      switch (resultText) {
      case "motor left":
                  out.write('C');
                  break;
      case "motor right":
                  out.write('G');
                  break;
      case "light open":
                  out.write('L');
                  break;
      case "light close":
                  out.write('O');
                  break;
      default:
                  break;
      }
      out.close();
}
```

Here it can be noticed that the messages sent to the access point device are only single byte messages. With a single byte command up to 255 different devices can be controlled.

The flowchart of the speech recognition process is illustrated in **Figure 15**.

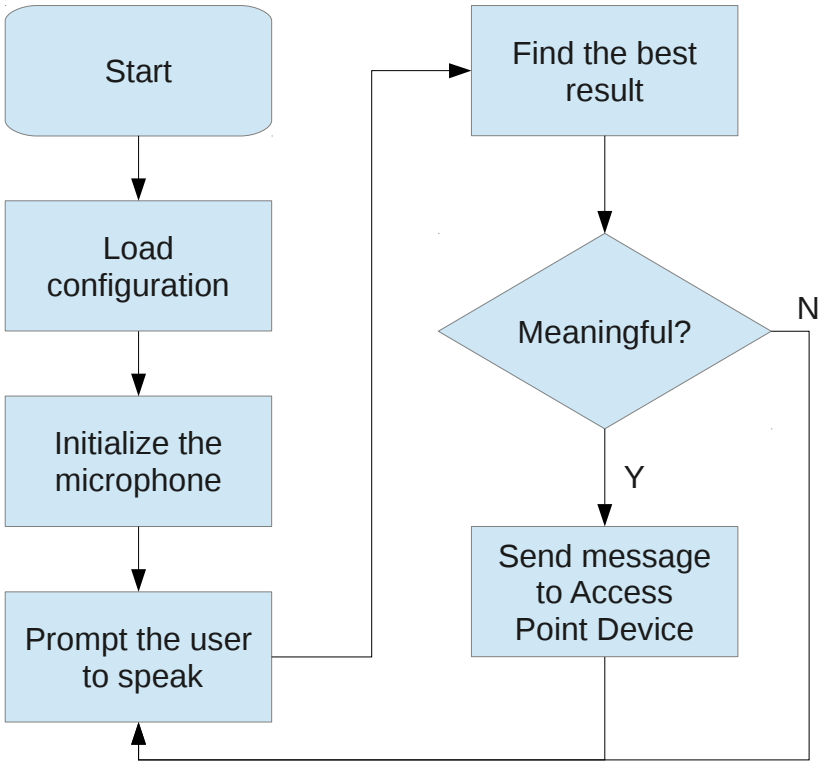


Figure 15. Speech recognition application flowchart.

5.2. Access Point Relaying

The software written for this part is simple. The access point goes to sleep mode immediately after it enables the UART interrupt mechanism. As soon as the buffer receives, the microcontroller wakes up and performs a one byte transmission. Then it goes back to sleep for the next round.

The code inside the interrupt routine is only few lines:

```
__interrupt void USCI0RX_ISR(void)
{
      unsigned char uart_command;
      uart_command=UCA0RXBUF;
      // Build packet
      txBuffer[0] = 2;            // Packet length
      txBuffer[1] = 0x01;         // Packet address
      txBuffer[2] = uart_command; // Load the command
```

```
        RFSendPacket(txBuffer, 3);  // Send value over RF
        while (!(IFG2&UCA0TXIFG));  // USCI_A0 TX buffer ready?
        UCA0TXBUF = uart_command;   // TX -> RXed character
}
```

The flowchart for the access point software is represented in the following **Figure 16**.

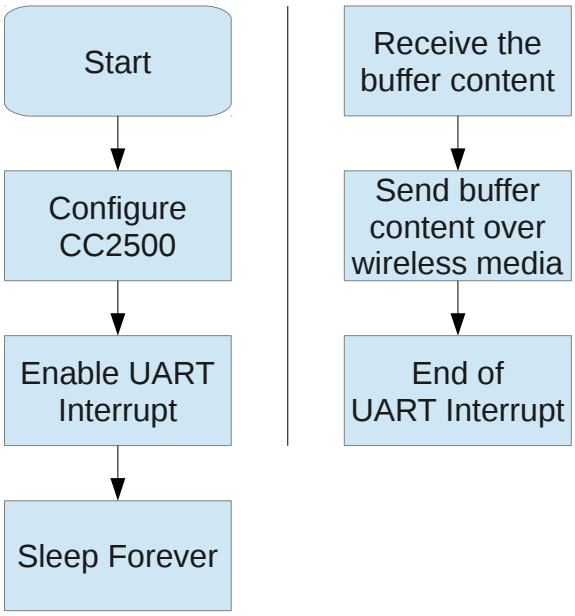


Figure 16. Access point flowchart. The UART interrupt is on the right side.

Before enabling the UART interrupt the radio chip must be configured. The parameters are adjusted in a separate header file and the used settings in this system are represented in **Table 5**. These parameters are loaded to the chip by using a single line of code.

```
writeRFSettings();
```

Table 5. Selected configuration for CC2500.

Radio frequency band	2.4 GHz
Data rate	250 kbps
Modulation	MSK
RF output power	0 dBm
Channel spacing	199.95 kHz

5.3. Actuator Device

The actuator device also keeps itself in the sleep mode as much as possible and only wakes up when there is an interrupt request from the radio transceiver CC2500 device. On the actuator device the PWM output is initialized before going into sleep mode. As explained before the MSP430F2274 supports many power options and the sleep mode used in here (LPM1) keeps the PWM clock source running.

As soon as a command is received over the wireless radio, the CC2500 raises a pin to logic high and triggers an interrupt. Then the MSP430F2274 wakes up and requests the received data from the transceiver device via SPI connection.

After the data is received, the message is extracted and evaluated by a switch case statement. If the message tells that the light should be switched on or off, just a pin is toggled and the relay circuit handles the rest. If the message tells the microcontroller to turn the motor left or right, then the PWM overflow values are adjusted accordingly. The servo motor hardware is already made in such a way that it responds to the rotation information from the PWM duty cycle. The duty cycles can be found in section 4.3. The flowchart for the actuator device is shown in **Figure 17**.

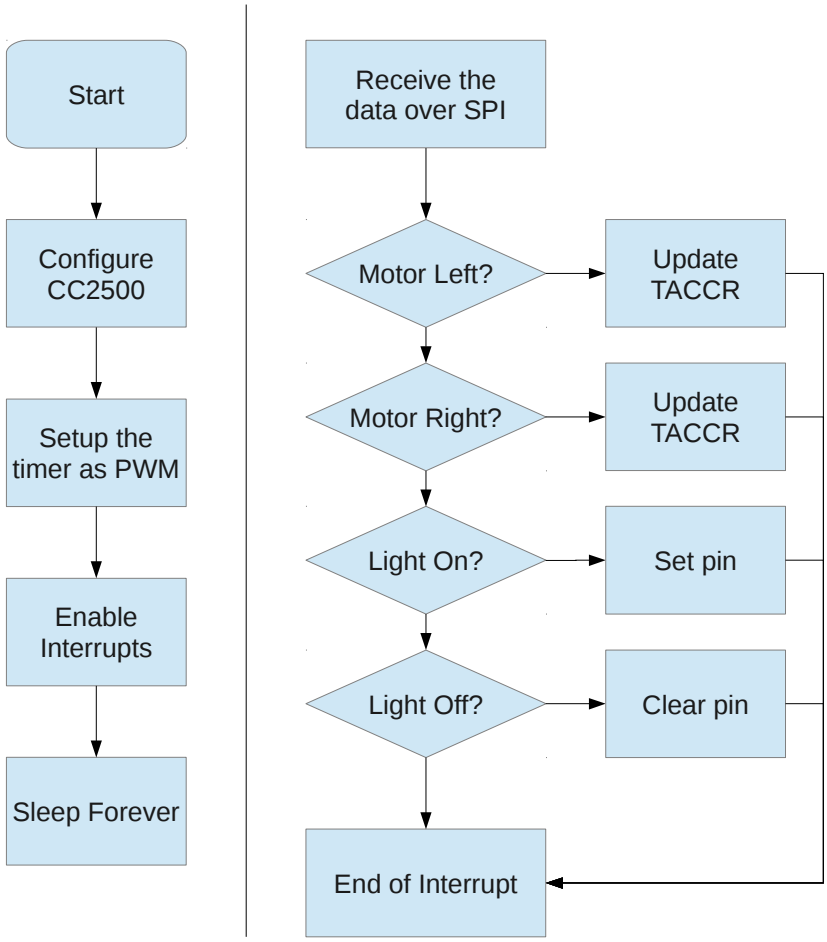


Figure 17. Actuator flowchart.

5.4. Measuring the RSSI

In order to verify that the wireless nodes are able to cover a sufficient communication range for home automation, some experiments have been done and presented further in section 6.3 of this document. A simple test software was written to log the RSSI values.

It is not necessary to present a flowchart for the measurements since it only contains one line of code. After a radio packet is received, the RSSI value can easily be obtained by its return value.

```
rssi_val = RFReceivePacket(rxBuffer,&len);
```

During the tests, this byte was sent to the PC over serial connection and recorded.

6. EXPERIMENTS AND RESULTS

In this section, the measurements and the results extracted from those measurements are represented regarding to the system described in chapter 4 and 5. First, the current measurements and discussions are given and compared to the ideal values provided by the manufacturer. Next, the error rates regarding to the speech recognition are provided and finally the RSSI measurements for both indoors and outdoors are plotted with graphs.

6.1. Current Consumption

The current consumption of the wireless sensor node has been analyzed regarding to the states of the components on the circuit. In wireless sensor nodes, it is important to keep the processor and the radio in sleep state as much as possible since the battery life is limited.

The MSP430F2274 offers different power configuration options to fulfill various needs. It supports active mode, sleep mode, and low power modes (LPM). In LPM modes, only necessary clock modules are running but not the CPU. In LPM the interrupts can be processed as they occur. In addition to that, the active mode CPU frequency can be scaled so that optimum computation power can be chosen.

The radio chip CC2500 also offers active mode and different sleep modes. Measurements represented in this section are done in both of these modes, in combination with different microcontroller power modes. The **Table 6** summarizes the measurement values for a operating voltage of 3.3V.

Table 6. Current consumption measurements.

MSP430F2274 mode	CC2500 mode	Current
– (~1 MHz)	Active	19.1 mA
LPM1	Active	18 mA
LPM1	Disabled	1.6 mA
LPM4	Disabled	1.56 mA

The power consumption values for LPM0, LPM2 and LPM3 are not shown in the table since the measurement values do not differ so much. This is due to the power consumed internally by the rest of the passive components on the PCB. The difference in terms of μA for the microcontroller modes are shadowed by the entire power consumption of the PCB. In fact, the current consumption for the MSP430F2274 can be as low as 300nA. In this thesis, the microcontroller starts in active mode, but then after the initialization stage it enters the LPM1 mode.

As it can be seen from the **Table 6**, the measurement values while the CC2500 is active are significantly higher. These results give a better understanding why the radio should be used only when it is needed. On the other hand, even though both the radio and the microcontroller would be active all the time, the maximum power consumption is 19.1mA which is still lower compared to other existing communication devices. These nodes produced by the TI have been developed for low power consumption.

In this work the system developed for home automation uses the LPM1 sleep mode and the CC2550 chip waits for reception of a character in active mode. The current consumption that has been measured for this case is 18mA.

6.2. Speech Recognition Accuracy

In this section the error rates of the speech recognition process have been discussed based on the experimental results. The test cases include three different scenarios. In the first scenario, the error rate only for the 'motor right' command has been tested. The second scenario is similarly, but it includes only the 'motor left' command. Finally the third test includes a random mixture of both commands.

The first test case was repeated 8 times to improve the consistency of the outcome. In each test the 'motor right' command was repeated. The details for each trial are given in **Table 7**.

Table 7. Results of the 'motor right' command tests.

Test Number	Correct Answer	Error	No Decision	Total Number of Tries
1	28	2	0	30
2	30	0	0	30
3	32	3	0	35
4	32	3	0	35
5	29	4	1	33
6	32	3	0	35
7	29	1	0	30
8	31	0	0	31

There is also the possibility for the program to completely ignore the command, and output that no decision has been made.

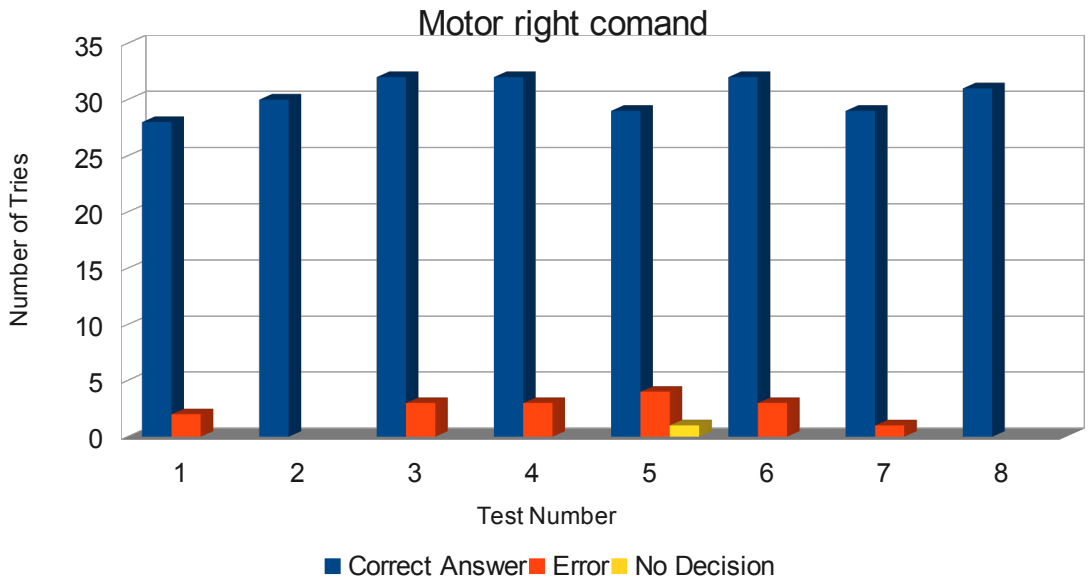


Figure 18. Estimation accuracy for 'motor right' command.

Figure 18 shows different results for each independent test. Regarding to these results, the total number of given commands is 259 and the total number of errors is 16. Consequently, the error rate for the 'motor right' command was calculated as:

$$\frac{16}{259} = 0.061776 \tag{26}$$

The command which was not evaluated (no decision) is not included in this result.

After this analysis, in the second test case the success of the 'motor left' command was examined. Nine different tests were carried out, and in each test the command was repeated several times. The results are summarized in **Table 8**.

Table 8. Results of the 'motor left' command tests.

Test Number	Correct Answer	Error	Total Number of Tries
1	27	3	30
2	29	1	30
3	34	1	35
4	30	0	30
5	35	0	35
6	30	0	30
7	35	0	35
8	30	0	30
9	30	0	30

During the tests for this command there has never been a no decision output. **Figure 19** visualizes the results based on **Table 8**.

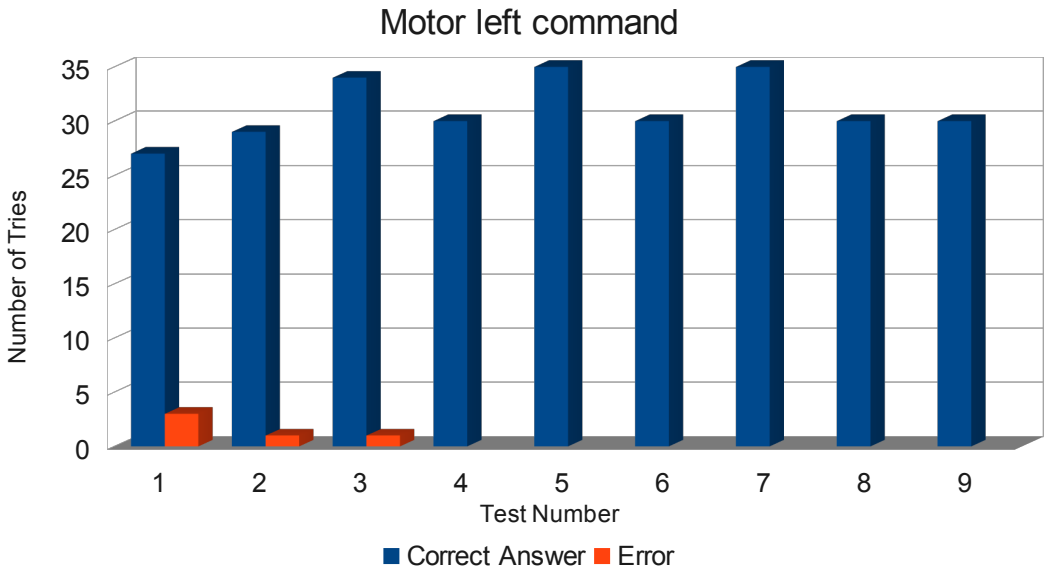


Figure 19. Estimation accuracy for 'motor left' command.

The different tests for the 'motor left' command shows that the error rates are independent from each other. Regarding to the collected data, the total number of commands is 285 and the total number of errors is 5. This means that the error rate for the second test case can be calculated as:

$$\frac{5}{285} = 0.01754 \tag{27}$$

Surprisingly the 'motor left' command was understood more accurately than the 'motor right' command.

The third test case involves the combination of both previous cases. In this case 'motor left' and 'motor right' words are spoken in a random way. In total six tests were carried out. Experimental results are represented in **Table 9**.

Table 9. Results of mixed commands tests.

Test Number	Correct Answer	Error	No Decision	Total Number of Tries
1	37	2	0	39
2	42	2	0	44
3	44	6	0	50
4	18	0	0	18
5	28	5	1	33
6	39	3	1	42

Here it can be seen that two commands were not evaluated and there was no output. The graph below visualizes the findings in this test case.

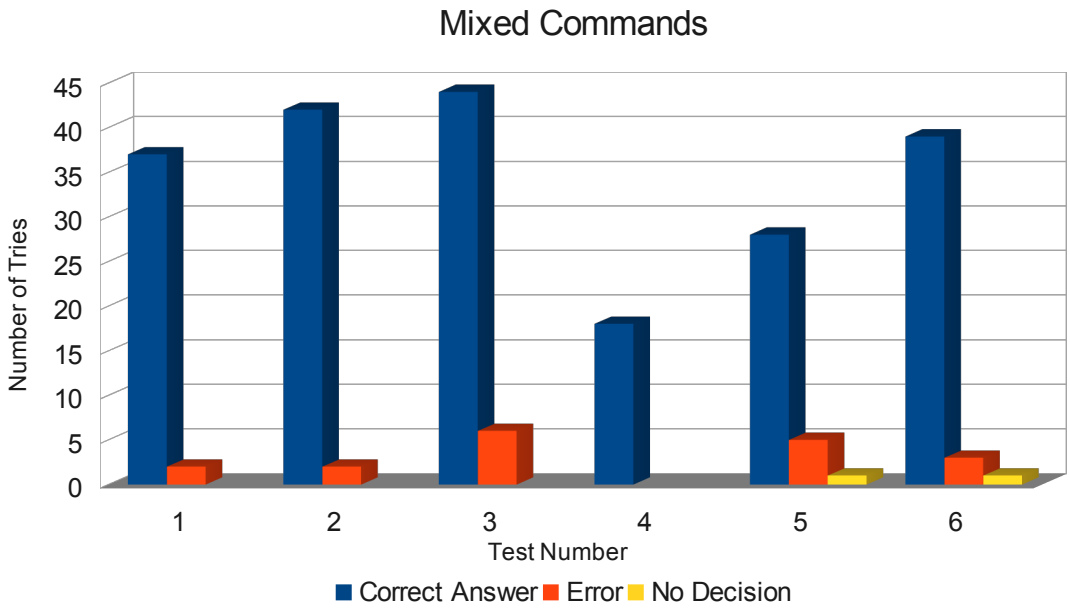


Figure 20. Estimation accuracy for both commands ('motor right' or 'motor left').

In a situation where two commands are combined, the system outputs behaves worse than the two previous tests (see **Figure 20**). The total number of the trials in this test case is 226 and the total number of errors is 18. As a result, the total error rate can be calculated as:

$$\frac{18}{226} = 0.079646 \tag{28}$$

When all these three test cases are considered, the 'motor left' command was understood better with respect to the other commands. The highest number of errors was found when two commands were used in a random way. This results shows that the speech recognition systems in today's technology are not safe enough to use it in safety critical applications. Since the home automation systems do not always require such a sensitivity, these results can be considered as acceptable for many applications. For example, in case the recognition process fails for opening the curtains, the user may easily repeat the command.

6.3. RSSI Measurements

The Received Signal Strength Indicator (RSSI) is a measure of the RF power at the transceiver input. The RSSI value is based on the measured signal level in the channel, and the gain of the Rx chain. In Rx mode, as soon as a packet sync word is detected, the RSSI status register of the CC2500 is stored until it is updated with the next sync word's reading. This mechanism allows every packet to be individually RSSI stamped.

During the RSSI measurement tests, two wireless nodes were used. One of them was connected to the PC and the other one was used as a transmitter. The transmitter node sends a packet everytime a pushbutton is pressed. When a packet arrives at the receiver node, it gets the RSSI value of the received packet and sends this value to the PC. On the PC side there was a program which monitors and logs the measured values.

The measurements were done on two different locations. The RSSI values in the RSSI status register were read purely in hexadecimal format. In the datasheet of the CC2500 chip, the formula is given to calculate the read values in terms of the absolute power level (RSSI_dBm). The values have been converted in the following way:

- Convert the hexadecimal RSSI value into decimal.
- If the reading is 128 or greater than that, the following formula is applied.

$$RSSI\_dBm = \frac{(RSSI\_dec - 256)}{2} - RSSI\_offset \tag{29}$$

- If the reading is less than 128, the following formula is applied.

$$RSSI\_dBm = \frac{RSSI\_dec}{2} - RSSI\_offset \tag{30}$$

In both of these formulas the RSSI_offset refers to a constant value defined for each data rate that CC2500 supports. The **Table 10** lists all the possible offset values. During these measurements the data rate was 250 kBaud/s, so that the reading from the table gives the offset of 72 dBm.

Table 10. Typical RSSI_offset values.

Data Rate [kBaud]	RSSI_offset [dBm]
2.4	71
10	69
250	72
500	72

6.3.1. Indoors

Indoor measurements were done on the 4th floor of the Fabriikki building in the University of Vaasa. The indoor measurements were done for two cases. The first case was when both nodes are located on the ground, and the second case was when both of the nodes were about 1m above the ground.

The results obtained from the first test case are represented in **Table 11**. The measured RSSI value for each distance given in the table, is the average RSSI value of eight transmitted packets from the same distance.

Table 11. Measured RSSI values while both nodes are on the ground.

Distance (m)	0	1.5	3	6	9	12	15	18	21	24	27	30
Average RSSI Value in dBm	-17	-77	-76	-78	-73	-90	-78	-86	-88	-93	-87	-93

Figure 21 shows the plotted measurements.

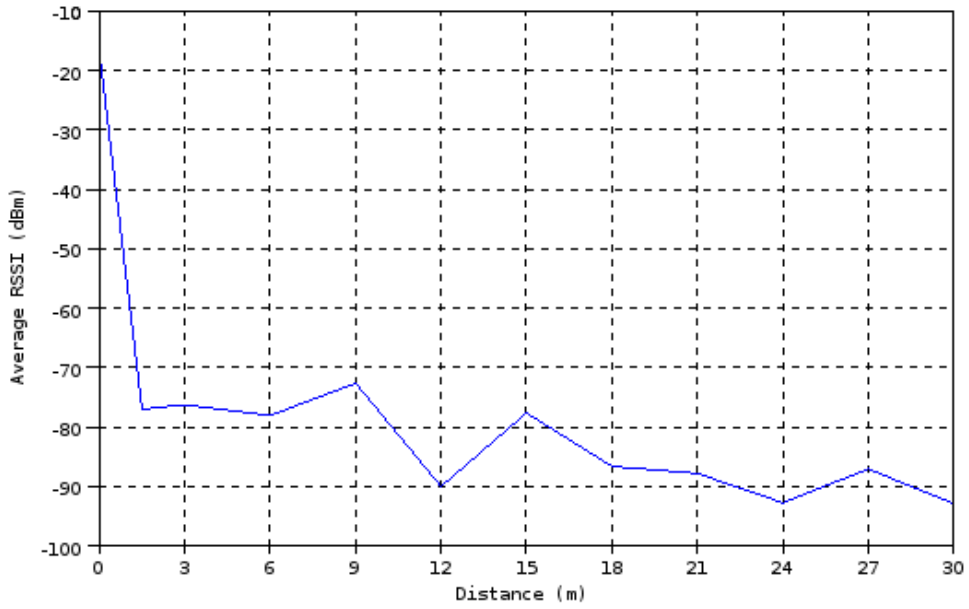


Figure 21. Indoor RSSI measurements for ground level communication.

Figure 21 shows that the RSSI measurements are generally decreasing with the distance, which is an expected situation. Apart from that, between 1.5m and 6m the RSSI value does not change so much. Going further, at 15m there is an increase in the measurements. This unexpected situation may be caused by the multipath effect. After 30m no signal was received.

In the second test case both of the nodes were 1m above the ground. The data collection method is identical to the method applied in the first case. The collected data is represented in **Table 12**.

Table 12. Measured RSSI values while both nodes are 1m above the ground.

Distance (m)	0	3	6	9	12	15	18
Average RSSI Value in dBm	-17	-68	-70	-75	-79	-80	-79
Distance (m)	21	24	27	30	33	36	39
Average RSSI Value in dBm	-87	-92	-80	-87	-84	-86	-84

As expected the RSSI values again decrease as the distance increases (see **Figure 22)**. In this graph, the measured values do not change significantly between 3m to 6m and between 12m to 18m. Additionally, at 27m there is an increase and up to 39m the value is stable and almost at constant level. The reason for that is probably again the multipath effect because at 39m there was a metal elevator door behind the transmitter which is able to reflect the signal.

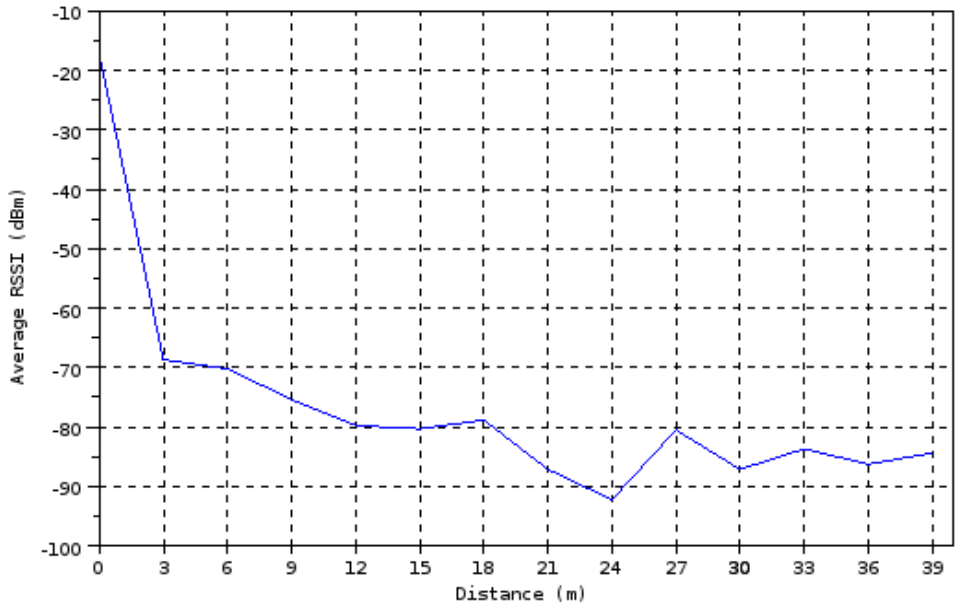


Figure 22. Indoor RSSI measurements for 1m above the ground level communication.

When **Figure 21** is compared with **Figure 22**, it obvious that the measurements could go up to 39m but the communication was lost at 30m. This probably because of the signal absorption caused by the ground.

6.3.2. Outdoors

The outdoor measurements were done near the shore, far away from any obstacles. The measurements were collected while the nodes are approximately 1m above the ground. Similar methods were applied to capture and evaluate the data as in the previous measurements. Measurement results are given in **Table 13**.

Table 13. RSSI measurements for outdoors.

Distance (m)	0	3	6	9	12	15	18	21	24	27
Average RSSI Value in dBm	-19	-67	-70	-83	-84	-85	-87	-87	-82	-85

In **Figure 23**, the RSSI values given in the table are plotted. The decrease in the measurements are somehow smoother compared to the indoor measurements. From 9m to 21m the decreasing seems to be linear in dBm scale. The results obtained here are somehow closer to the ideal situation. After 27m the communication was lost.

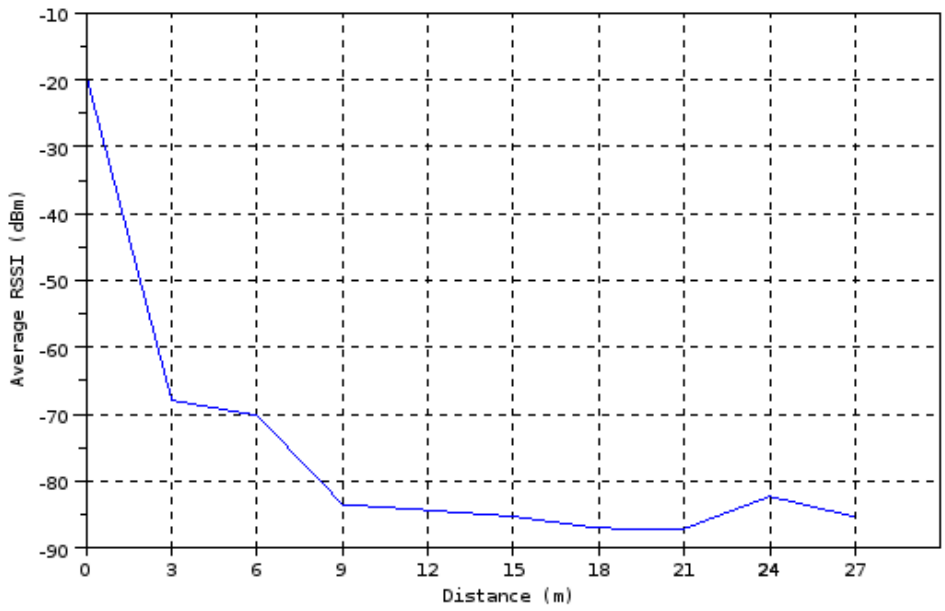


Figure 23. Outdoor RSSI measurements plot.

7. CONCLUSIONS

The target of this thesis was to research the possibilities on how to apply speech recognition methods in wireless home automation. Sphinx 4 software which utilizes Hidden Markov Models was used to recognize the spoken words and to convert them into text form. The results were passed to a wireless node via USB connection. Then this wireless node transmitted the message to another node to control home appliances. Thus, the speech recognition methods were combined with wireless sensor nodes to suit the needs of a wireless home automation control system.

The achieved results from the speech recognition are acceptable for many applications which do not require real time. In contrast, it would not be suitable for some safety critical home appliances like a cooking system, since this method is prone to errors. On the other hand, this system would be very helpful for old or disabled people. The ability to open the curtains by using speech would be a great help for elder people or for those who need a wheelchair.

In order to verify the capabilities of the system, two control mechanisms were implemented. The first mechanism is a speech based wireless motor control and the second mechanism is a speech based wireless light control. The servo motor did not require any additional circuitry but for the light control a switching circuit was designed using an optocoupler and a relay.

One constraint of the wireless systems is the transmission range. During the experimental tests, it was verified that the communication range is around 30m which is suitable for home applications. For applications which require longer transmission range, it is possible to add a relaying node easily.

In the future, this work may be improved with more accurate speech recognition techniques, more advanced software, and improved speech models. Sphinx 4 supports voice training so that it can achieve better results. Additionally, language models can be improved or new language models may be ported so that it might be used in different

countries. Since the software is able to hear, it is also possible to react for hand claps or finger snaps, too.

Furthermore, there are light speech recognition algorithms for platforms having low computational power. These speech recognition algorithms can be implemented inside a wireless sensor node for in-node computation independently from a PC.

REFERENCES

AlShu'eili, Humaid, Gourab Sen Gupta & Subhas Mukhopadhyay (2011). Voice Recognition Based Wireless Home Automation System. In: *International Conference on Mechatronics*. New Zealand: IEEE [2013].

CMU Sphinx. Open Source Toolkit For Speech Recognition [online] [cited 20 Feb. 2013]. Available from Internet: <URL: http://cmusphinx.sourceforge.net/>.

Dargie, Waltenegus & Christian Poellabauer (2010). *Fundamentals of Wireless Sensor Networks: Theory and Practice*. UK: John Wiley and Sons.

Derbali, Morched, Mu'tasem Jarrah & Mohd Taib Wahid (2012). A Review of Speech Recognition with Sphinx Engine in Language Detection. *Journal of Theoretical and Applied Information Technology* 40:2, 147–155.

Derene, Glenn (2009). Home Control. *Popular Mechanics* 186:2, 96-97. ISSN 00324558.

EE Herald. SPI Bus Interface [online] [cited 27 Feb. 2013]. Available from Internet: <URL: http://www.eeherald.com/section/design-guide/esmod12.html>.

Harper, Richard (2003). *Inside the Smart Home*. Landon: Springer. ISBN 1–85233–688–9.

Lamere, Paul, Philip Kwok, Evandro B. Gouvea, Bhiksha Raj, Rita Singh, William Walker & Peter Wolf (2003). The CMU Sphinx-4 Speech Recognition System. In: *Conference on Acoustics, Speech and Signal Processing* [online]. Hong Kong: IEEE [cited 05 Feb. 2013]. Available from Internet: <URL: www.cs.cmu.edu/~rsingh/homepage/papers/icassp03-sphinx4_2.pdf>.

Massé, Dan (2012). 90 Million Homes Worldwide Will Employ Home Automation Systems by 2017. *Microwave Journal* 55:7, 49.

Rabiner, Lawrence & Biing-Hwang Juang (1993). *Fundamentals of Speech Recognition*. New Jersey: PTR Prentice Hall. ISBN 0–13–015157–2.

Rajman, Martin (2007). *Speech and Language Engineering*. Florida: EPFL Press. ISBN 978–2–940222–04–9.

Rand, Peder (2013). *Wireless Lighting Control: The Bright Road Ahead* [online] [cited 15 March 2013]. Available from the Internet: <URL: www.eetimes.com/ContentEETimes/Documents/TI%20paper.pdf>.

Stallings, William (2005). *Wireless Communications & Networks*. 2nd Ed. New Jersey: Pearson Prentice Hall.

Stallings, William (2007). *Data and Computer Communications*. 8th Ed. New Jersey: Pearson Prentice Hall. ISBN: 0–13–243310–9.

Sullivan, Kevin R. (2013). *Understanding Relays* [online] [cited 4 Feb. 2013]. Available from the Internet: <URL: www.autoshop101.com/forms/hweb2.pdf>.

Texas Instruments Incorporated (2009). *eZ430-RF2500 Development Tool User's Guide* [online] [cited 25 Jan. 2013]. Available from the Internet: <URL: http://www.ti.com/general/docs/lit/getliterature.tsp?baseLiteratureNumber=slau227&fileType=pdf>.

Texas Instruments Incorporated (2011). *CC2500 Low-Cost Low-Power 2.4 GHz RF Transceiver* [online] [cited 28 March 2013]. Available from the Internet: <URL: http://www.ti.com/general/docs/lit/getliterature.tsp?genericPartNumber=cc2500&fileType=pdf>.

Vieira, Marcos A.M., Adriano B. da Cunha & Diogenes C. da Silva Jr. (2006). *Designing Wireless Sensor Nodes*. Embedded Computer Systems: Architectures, Modeling and Simulation 4017, 99–108.

Wikipedia. Serial Peripheral Interface Bus [online] [cited 27 Feb. 2013]. Available from Internet: <URL: http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus>.

Yuksekkaya, Baris, A. Alper Kayalar, M. Bilgehan Tosun, M. Kaan Ozcan & Ali Ziya Alkar (2006). A GSM, Internet and Speech Controlled Wireless Interactive Home Automation System. *Consumer Electronics, IEEE Transactions on* 52:3, 837–843. ISSN 0098-3063.

Zeng, Xiaohua, Abraham O. Fapojuwo & Robert J. Davies (2006). Design and Performance Evaluation of Voice Activated Wireless Home Devices. *Consumer Electronics, IEEE Transactions on* 52:3, 983–989.

APPENDICES

APPENDIX 1. eZ430-RF2500T Target Board Pinout.

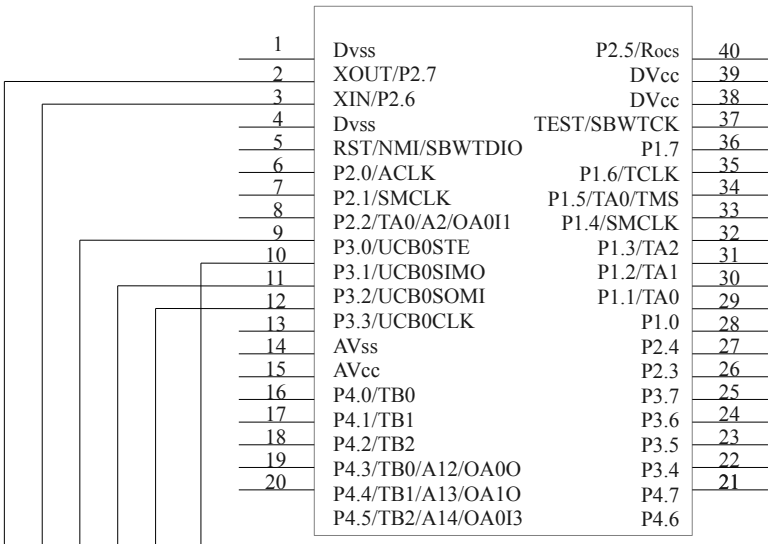
Pin	Functions	Description
1	GND	Ground reference
2	VCC	Supply voltage
3	P2.0 / ACLK / A0 / OA0I0	General-purpose digital I/O pin / ACLK output / ADC10, analog input A0
4	P2.1 / TAINCLK / SMCLK / A1 / A0O	General-purpose digital I/O pin / ADC10, analog input A1 Timer_A, clock signal at INCLK, SMCLK signal output
5	P2.2 / TA0 / A2 / OA0I1	General-purpose digital I/O pin / ADC10, analog input A2 Timer_A, capture: CCI0B input/BSL receive, compare: OUT0 output
6	P2.3 / TA1 / A3 / VREF– / VeREF– / OA1I1 / OA1O	General-purpose digital I/O pin / Timer_A, capture: CCI1B input, compare: OUT1 output / ADC10, analog input A3 / negative reference voltage output/input
7	P2.4 / TA2 / A4 / VREF+ / VeREF+ / OA1I0	General-purpose digital I/O pin / Timer_A, compare: OUT2 output / ADC10, analog input A4 / positive reference voltage output/input
8	P4.3 / TB0 / A12 / OA0O	General-purpose digital I/O pin / ADC10 analog input A12 / Timer_B, capture: CCI0B input, compare: OUT0 output
9	P4.4 / TB1 / A13 / OA1O	General-purpose digital I/O pin / ADC10 analog input A13 / Timer_B, capture: CCI1B input, compare: OUT1 output
10	P4.5 / TB2 / A14 / OA0I3	General-purpose digital I/O pin / ADC10 analog input A14 / Timer_B, compare: OUT2 output
11	P4.6 / TBOUTH / A15 / OA1I3	General-purpose digital I/O pin / ADC10 analog input A15 / Timer_B, switch all TB0 to TB3 outputs to high impedance

12	GND	Ground reference
13	P2.6 / XIN (GDO0)	General-purpose digital I/O pin / Input terminal of crystal oscillator
14	P2.7 / XOUT (GDO2)	General-purpose digital I/O pin / Output terminal of crystal oscillator
15	P3.2 / UCB0SOMI / UCB0SCL	General-purpose digital I/O pin USCI_B0 slave out/master in when in SPI mode, SCL I2C clock in I2C mode
16	P3.3 / UCB0CLK / UCA0STE	General-purpose digital I/O pin USCI_B0 clock input/output / USCI_A0 slave transmit enable
17	P3.0 / UCB0STE / UCA0CLK / A5	General-purpose digital I/O pin / USCI_B0 slave transmit enable / USCI_A0 clock input/output / ADC10, analog input A5
18	P3.1 / UCB0SIMO / UCB0SDA	General-purpose digital I/O pin / USCI_B0 slave in/master out in SPI mode, SDA I2C data in I2C mode

APPENDIX 2. Key Features of the CC2500.

Parametrics	CC2500
Frequency (Min)	2400MHz
Frequency (Max)	2483.5MHz
Device Type	Transceiver
Data Rate (Max) (kbps)	500
Operating Voltage (Min) (V)	1.8
Operating Voltage (Max) (V)	3.6
Rx Current (Lowest) (mA)	13.3
Wakeup Time (PD-->RX/TX) (uS)	240
Modulation Techniques	OOK, 2-FSK, GFSK
Sensitivity (Best) (dBm)	-104
Tx Power (Max) (dBm)	1
Programmable Output Power Ranging From (dBm)	-30 to 1
Antenna Connection	Differential

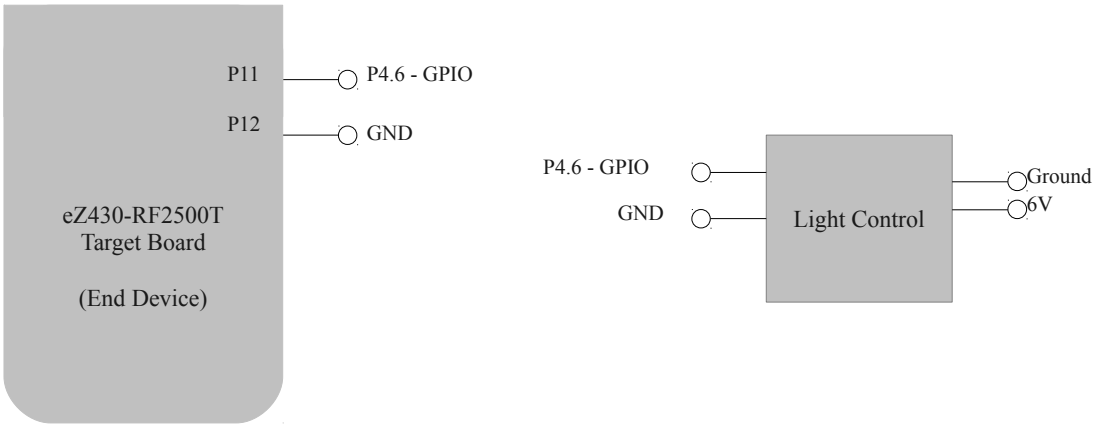
APPENDIX 3. Schematic design of the SPI link between MSP430F2274 and CC2500.



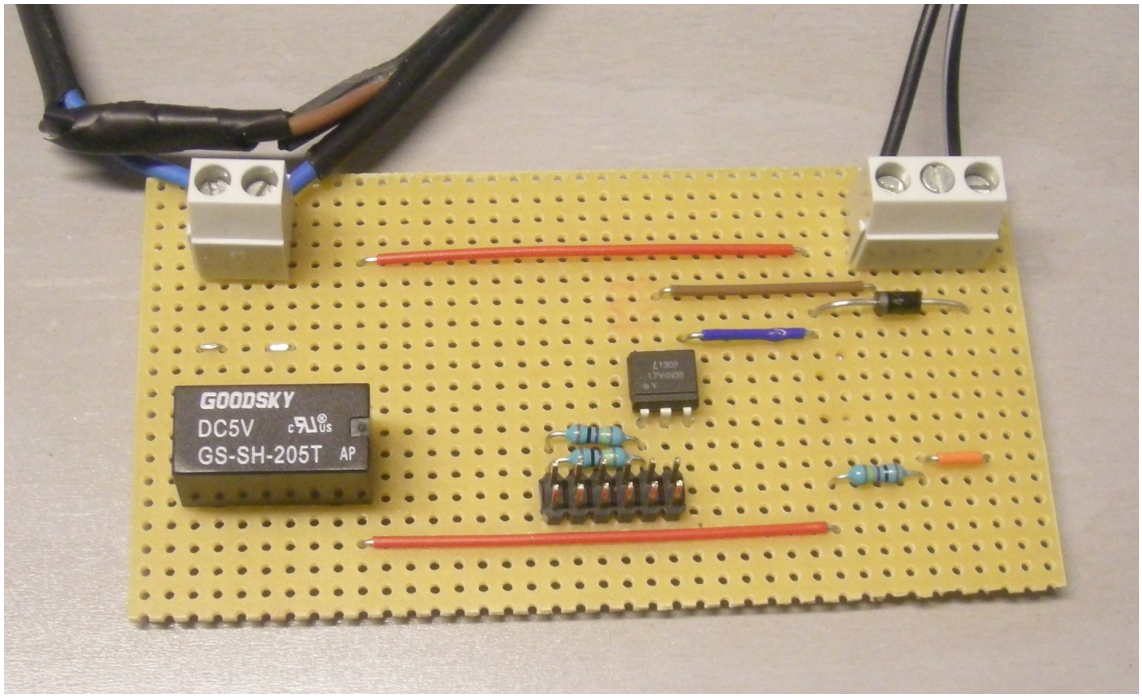
APPENDIX 4. Basic information about Hitec HS-422 servo motor.

Modulation:	Analog
Torque:	4.8V: 45.8 oz-in (3.30 kg-cm) 6.0V: 56.9 oz-in (4.10 kg-cm)
Speed:	4.8V: 0.21 sec/60° 6.0V: 0.16 sec/60°
Weight:	1.60 oz (45.5 g)
Dimensions:	Length: 1.59 in (40.4 mm) Width: 0.77 in (19.6 mm) Height: 1.44 in (36.6 mm)
Motor Type:	3-pole
Gear Type:	Plastic
Rotation/Support:	Bushing
Pulse Cycle:	20 ms
Pulse Width:	900-2100 µs

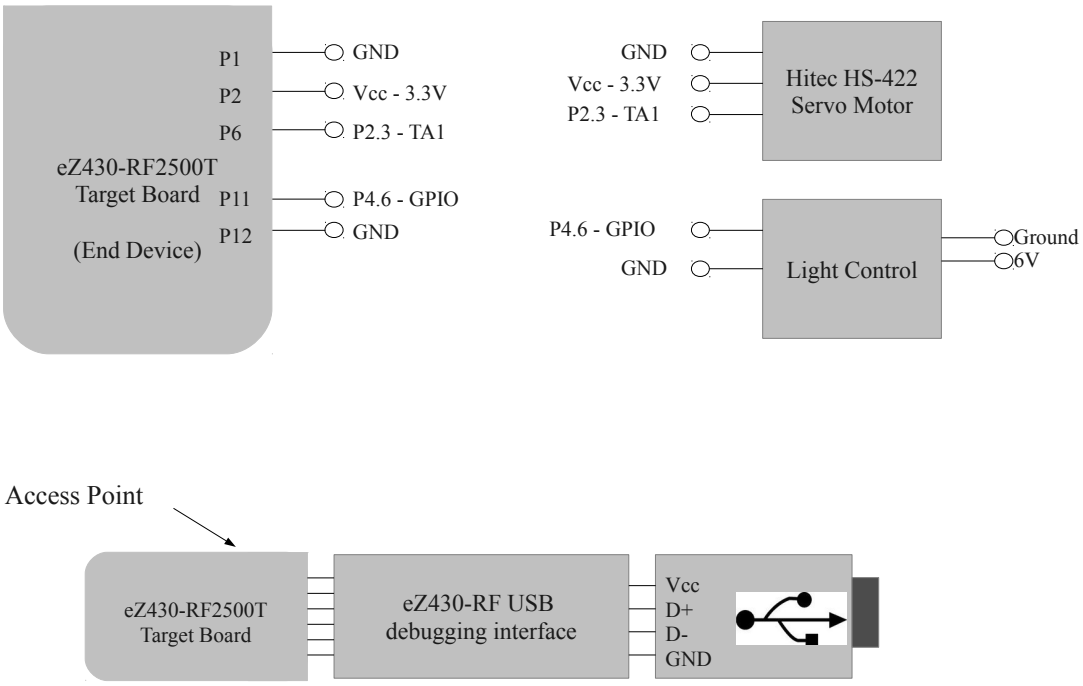
APPENDIX 5. Pin connection between the end device and the light control board.



APPENDIX 6. Picture of the lighting control board.



APPENDIX 7. Pin connection for the home automation system.



APPENDIX 8. Picture of the home automation system.