2014

HUMIDITY SENSOR CIRCUIT USING REAL TIME OPERATING SYSTEM (FREERTOS) KERNEL

Bojie Chen
University of Kentucky, cbj19870526@gmail.com

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Bojie Chen, Student

Dr. Zhi D. Chen, Major Professor

Dr. Caicheng Lu, Director of Graduate Studies

HUMIDITY SENSOR CIRCUIT USING REAL TIME OPERATING SYSTEM (FREERTOS) KERNEL

THESIS

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science
in Electrical Engineering in the College of Engineering
at the University of Kentucky

By

Bojie Chen

Lexington, Kentucky

Technical Director: Dr. Zhi D. Chen, Advanced Semiconductor Processing Technology, LLC, Lexington, Kentucky

Administrative Director: Dr. YuMing Zhang, Professors of Electrical Engineering, University of Kentucky

Lexington, Kentucky

2014

ABSTRACT OF THESIS

HUMIDITY SENSOR CIRCUIT USING REAL TIME OPERATING SYSTEM (FREERTOS) KERNEL

A humidity sensor can be used to measure the moisture content of the environment. The physical change of the sensor expresses as the change of electrical property like capacitance, resistance, voltage, current, frequency, etc. In order to process these analog signals digitally, microprocessor is involved in the measurement. This thesis presents design of a circuit to measure low moisture levels. The 16-bit RISC mixed signal microcontroller MSP430F249 from Texas Instruments will be used. The circuit has good performance at extremely low humidity levels.
Meanwhile, a small real time operating system kernel FreeRTOS, a market leading RTOS from Real Time Engineer Ltd is ported to the microcontroller. The basic concept about FreeRTOS and how to port this RTOS to MSP430F249 microcontrollers will be the topics of this thesis as well.

KEYWORDS: Sensor, Circuit, Microprocessor, Embedded System, FreeRTOS

Bojie Chen

December 7, 2014

HUMIDITY SENSOR CIRCUIT USING REAL TIME OPERATING SYSTEM (FREERTOS) KERNEL

By

Bojie Chen

Dr. Zhi D. Chen
Technical Director of Thesis

Dr. YuMing Zhang
Administrative Director of Thesis

Dr. Caicheng Lu
Director of Graduate Studies

December 7, 2014

ACKNOWLEDGEMENTS

I would like to thank my academic advisor, Dr. Zhi David Chen, for the opportunity he gave me to pursue my Master Degree in the area of embedded system, and all the guidance and help I've received from him all through these years. This thesis would be impossible without his extensive knowledge and innovative ideas in this field.

I would also like to thank Dr. YuMing Zhang for serving as administrative advisor, and Dr. James Lumpp for serving as committee member, and for the insightful guidance I've received from them.

Special thanks to Jason Rexroat and Timothy Lim for providing technical assistance.

I would also like to thank all my group members.

Last but not least, I would like to express my deepest gratitude to my parents, for the endless love and support I have always been with since I was born.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1 Introduction

1.1 Background

A humidity sensor measures the humidity level by measuring the change in the resistance of an element or the change in the electrostatic capacity of that element as it absorbs or releases moisture. Humidity sensors can be used not only to measure the humidity in an atmosphere, but also to automatically control humidifiers, dehumidifiers, and air conditioners for humidity adjustment (1).They are important for extremely environment where water or vapor has vital influence. These sensors are widely used in industry, agriculture, medicine, and many other areas today. People are trying to use different materials to develop humidity sensor with good performance. At the present time, much has been reported about humidity sensors making use of materials such as electrolytes, organic polymers, alumina thin films, and other metal oxides (2). However, many humidity sensors suffered from serious drawbacks, e.g. long term instability, large humidity hysteresis, and slow response, etc (3).

According to the measurement range and the responded physical variable, humidity sensors are divided into two types: relative humidity (RH) sensors and absolute humidity (moisture) sensors. Relative humidity sensors are used to measure the ratio of water vapor pressure to the saturated water vapor pressure at a fixed temperature, in the range from 10% to 100% RH, referred to as high humidity. Absolute humidity sensors are used to detect the water vapor concentration in a flow of gas or in a sealed package, ranging from -80°C to +20°C dew/ frost point. Absolute humidity covers a

wide range of humidity compared with relative humidity, i.e. -80°C to -10°C frost point corresponding to 0. 5 to 2300 parts per million (ppm_v), at 1 atmosphere, is referred to as trace moisture, and -10°C to +20°C dew/frost point corresponding to 10% to 100% RH at 20°C , overlaps the relative humidity at 20°C (3).

The research of this thesis is based on an alpha alumina-based humidity sensor, which is reliable and drift-free humidity sensor. It is essential to have a robust sensor to design a competitive product. Long time repeatability, long term stability, slow response time, wide measurement range, high resolution, good sensitivity, calibration, and temperature independence are the main factors decide the performance of the dew point transmitter.

1.2 Motivation

A Sensor can be used to measure physical quantities of environment. Usually, these physical quantities are in electrical property like capacitance, resistance, voltage, current, frequency, magnetic field, etc. Sensor is a bridge between physical world and engineering world. Through sensor, engineers can measure lots of physical parameters of the world. People develop temperature sensor, pressure sensor, flow sensor, biosensor, image sensor, acceleration sensor, and speed sensor, just name a few. The physical quantities need to be converted to analog signal or digital data which can be read by an experiment observer, instrument, digital equipment, or microcontroller. General speaking, microcontroller is the first choice for developing of the transmitter, especially for the purpose of intelligent operation. The transmitter should have

abilities to measure capacitance automatically, finish calculation quickly, show humidity or dew point to users, and output an industry standard 4-20mA analog current signal.

Sensor acts as the vital interface between environment and electrical system. Microcontroller is a digital chip. It is capable of dealing with “1” and “0”.Nevertheless, it cannot process the analog signal from outside directly. For the analog signal to be processed by digital microcontroller, it needs A/D analog to digital convertor. How to convert the capacitance to digital data is the key part of the research. Due to the SoC (system on a chip) technology, it is quite feasible to implement this conversion on a single chip. SoC is a concept that semiconductor manufacturer integrates analog unit, digital unit and mixed-signal all on one chip.

In order to develop a functional humidity analyzer (Dew Point meter), the research has to solve the following main issues:

1). Analog sensor conditioning circuit design

2). Embedded hardware design (circuit schematic design, PCB design, manufacture, PCB assembly, and test)

3). Embedded firmware design

4). Sensor calibration

5). 4-20mA current loop design (output analog signal to user)

6). Final test

1.3 Problem Statement

As I mentioned earlier, the dew point meter has an alpha-alumina based humidity sensor inside, which is highly sensitive for detection of a wide range of moisture levels (-80°C - +20°C dew point). It is a capacitive sensor. In the lab, I used Agilent 4284A precision LCR meter to measure the sensor's capacitance and resistance, which showed excellent performance for moisture measurement. This instrument is designed specifically for the LCR measurement. So I need to design an electronic circuit to measure the capacitance of this sensor, or to capture data which is in proportion to the capacitance. The data can be converted to dew point or ppm_v moisture levels after calibration. The circuit converts the physical change into microprocessor accessible data.

The capacitance of the sensor is about 1nf to 10nf. The sensor has voltage limitation, lower than 5V. Experiment suggests it is better to measure the sensor under low frequency condition, lower than 2KHz. On the market, there are some chips designed particularly for small capacitance measurement. However, they cannot meet the requirements of this sensor. Some chips provide low resolution. Some supply high voltage on their pins, which may break the sensor. The solution to these problems is to build a circuit, which is dedicated to this sensor to achieve very high accuracy for detection of extremely low moisture levels (<1 ppm_v or -80°C dew point).

1.4 Thesis Overview

This thesis research, as illustrated throughout the Chapter 1, leads to the development

of a microcontroller based embedded system. In Chapter 2, the definition of Dew Point is presented. Chapter 3 describes some analog sensor conditioning circuits. Chapter 4 concentrates on an economic measurement technology, slope A/D conversion, which is the research topic of this thesis. Chapter 5 introduces a microcontroller, TI MSP430F249, which is a mixed signal processor. Chapter 6 explains two lines 4-20mA current loop design. Chapter 7 described IN4148 diode temperature sensor. Chapter 8 introduces the FreeRTOS real time operation system, and how to port FreeRTOS to this processor. Chapter 9 is about the test and calibration. Chapter 10 is the conclusion and future work.

Chapter 2 Dew Point Measurement

2.1 Concepts of Humidity and Dew Point

The dew point is the temperature below which the water vapor in air at constant barometric pressure condenses into liquid water at the same rate at which it evaporates. The condensed water is called dew when it forms on a solid surface. The dew point is a water-to-air saturation temperature (4). Humidity is the amount of water vapor in the air. Water vapor is the gaseous state of water and is invisible (5). Absolute, relative, and specific humidity are the three main different measurements of humidity. This research focuses on absolute humidity measurement. Absolute humidity is the mass of water vapor per unit volume of total air and water vapor mixture.

The Dew Point is a temperature at which water vapors condense on a mirror surface, which uses the same unit as the temperature, Celsius or Fahrenheit. The Dew Point is always lower than (or equal to) the air temperature. When the water vapor which is mixed with the air, can no long stay in the air as the gas state, some of the water vapor needs to condense into liquid water. At this point, the temperature equals to the Dew Point temperature, and the relative humidity is 100%, whereas the Dew Point is lower than temperature if the air can mix with more water vapor. During daytime, the temperature is higher than the Dew Point temperature. The air can hold more water vapor. So we cannot see dew (liquid water) on leaves, grass, or ground. When the temperature falls below the Dew Point temperature at night or on early morning, the

water vapor cannot stay in the air any more. The water must condense into liquid water, which makes leaves, grass, ground, and car windows wet. This is a natural phenomenon. Dew Point also has influence on human body. Below is a table from Wikipedia, it shows the dew point influence on human body.

Table 2.1: Dew Point has influence on human comfort

Dew Point (°C/°F)		**Human perception**	**Relative humidity at at 32 °C (90 °F)**
Over 26	Over 80	Severely high. Even deadly for asthma related illnesses	65% and higher
24-26	75-80	Extremely uncomfortable, fairly oppressive	62%
21-24	70-74	Very humid, quite uncomfortable	52-60%
18-21	65-69	Somewhat uncomfortable for most people at upper edge	44-52%
16-18	60-64	OK for most, but all perceive the humidity at upper edge	37-46%
13-16	55-59	Comfortable	38-41%
10-12	50-54	Very comfortable	31-37%
Under 10	Under 50	A bit dry for some	30%

2.2 Relation between Absolute Humidity and Dew Point

In order to measure Dew Point temperature, people develop many different devices.

One of the devices uses polished metal mirror which is cooled as air is passed over it. The Dew Point is the temperature at which dew forms on the surface of mirror. This type of device is very expensive and usually is used as lab reference instrument to calibrate other humidity sensors.

For this research, I used alpha-alumina based humidity sensor, which can measure absolute humidity. Dew Point is the temperature below which the water vapor in air must condense into liquid water, which means the water vapor is saturated in air. The moisture content stays the same in air when the temperature falls below the Dew Point. As long as the Dew Point temperature is measured, the absolute humidity is known. The Dew Point and absolute humidity have one to one relation. The same reason if I measure the absolute humidity of air, I can find out the Dew Point temperature.

2.3 Dew Point Absolute Humidity Conversion Table

A Dew Point vs. PPM vs. Absolute humidity conversion table is presented below.

Formula for Celsius vs. Fahrenheit conversion:

$$(℃)\ 9/5 + 32 = (℉)$$

The formula above calculates the Dew Point in Celsius to Fahrenheit. The conversion table provides data in Celsius.

Table 2.2: Dew Point vs. PPM vs. Absolute Humidity table(6)

DP(°C)	PPM	Absolute Humidity(g/m*3)	DP(°C)	PP_{mv}	Absolute Humidity(g/m*3)
0	6033	4.517	-50	38.89	0.02912
-2	5111	3.827	-52	30.32	0.02270
-4	4318	3.233	-54	23.51	0.01761
-6	3640	2.725	-56	18.16	0.01360
-8	3060	2.292	-58	13.96	0.01045
-10	2566	1.921	-60	10.68	0.007998
-12	2145	1606	-62	8.128	0.006087
-14	1789	1.339	-64	6.154	0.004608
-16	1487	1.113	-66	4.635	0.003471
-18	1233	0.9233	-68	3.471	0.002599
-20	1019	0.7629	-70	2.584	0.001935
-22	840	0.6291	-72	1.914	0.001433
-24	690.2	0.5169	-74	1.409	0.001055
-26	565.3	0.4233	-76	1.031	0.0007717
-28	461.3	0.3454	-78	0.7492	0.0005610
-30	375.3	0.2810	-80	0.5410	0.0004051

Table 2.2: Dew Point vs. PPM vs. Absolute Humidity table (Continued…)

-32	304.1	0.2278	-82	0.3881	0.0002906
-34	245.8	0.1841	-84	0.2764	0.0002070
-36	197.8	0.1481	-86	0.1955	0.0001464
-38	158.7	0.1189	-88	0.1372	0.0001028
-40	126.8	0.09491	-90	0.09564	0.00007161
-42	100.9	0.07555	-92	0.06611	0.00004950
-44	80.03	0.05993	-94	0.0452	0.00003394
-46	63.19	0.04732	-96	0.03087	0.00002308
-48	49.67	0.03720	-98	0.02077	0.00001555
			-100	0.01387	0.00001039

2.4 Alpha-Alumina Based Dew Point (DP) Transmitter

The Dew Point vs. Absolute humidity conversion table is used for better understanding of the relation between Dew Point and absolutely humidity. In order to design a good transmitter, for this research, I need a good humidity measurement device to calibrate the transmitter. When using this Alpha Alumina DP sensor, the microprocessor actually is measuring the change of capacitance, and relating the capacitance to the humidity. It is the relation between humidity and capacitance. If the microprocessor can avoid errors in capacitance measurements, and process the relations between capacitance and humidity more accurately. The transmitter may

have better performance. It is quite hard to measure accurate capacitance of the humidity sensor, even with the precision LCR meter. The microcontroller needs toapply different techniques to minimize the measurement errors. Based on the good design of hardware and software, in the end, calibration is critical for the DP transmitter. In this research, I used another DP transmitter, FA410 (CS INSTRUMENTS), to calibrate the Alpha-Alumina based DP transmitter in the lab temporarily. The calibration condition in the lab needs to be improved in the future for the sake of better device performance.

Chapter 3 Analog Sensor Conditioning Circuit

Analog sensors produce a change in an electrical property, which needs to be conditioned by analog circuit before conversion to digital data. This chapter explains some analog sensor conditioning circuits that used to convert capacitance into property like voltage, frequency, time, impedance. The preparation helps me gain new insight into the world of circuit design.

3.1 Capacitance to Frequency Convertor (555 Timer Chip)

The 555 timer integrated circuit is a very common chip used in a variety of applications such as timer, pulse generation, oscillator circuit, and pulse width modulation. The 555 timer has three operation modes: monostable, astable, and bistable. For the purpose of capacitance measurement, astable mode is used. Figure 3.1 is the circuit diagram of the capacitance to frequency convertor.

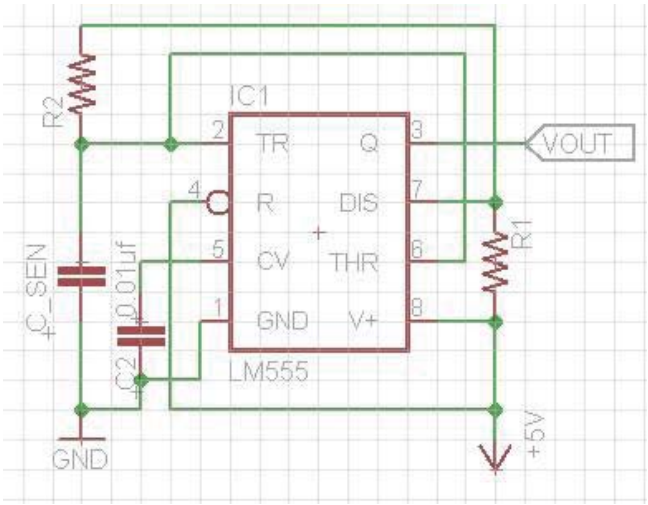

Figure 3.1 Capacitance to frequency convertor

In astable mode, pin 3 of 555 timer outputs a rectangular pulse at a specific frequency. Resistor R1 is connected between +5V power supply and the discharge pin (pin7),

another resistor R2 is connected to R1 and pin7, the trigger (pin 2), and threshold (pin 6), pin 2 and pin 6 share a common node. The capacitance C_SEN which represents the sensor is charged through R1 and R2, and discharged through R2 only. R1, R2, and C_SEN together decide the frequency. Figure 3.2 is the inside schematic of 555 timer chip.

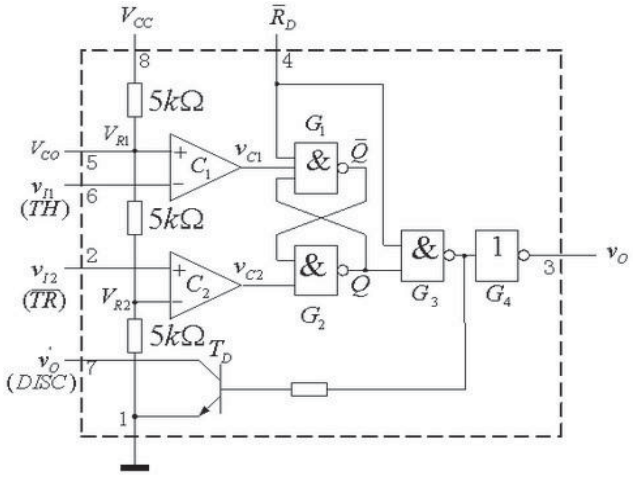

Figure 3.2 Inside schematic of 555

Table 3.1: Input and output relation of 555

Input			Output		
Rd	Vt1	Vt2	Q	Td	Vo
0	--	--	0	on	0
1	>2/3Vcc	>1/3Vcc	0	on	0
1	<2/3Vcc	>1/3Vcc	No change	No change	No change
1	<2/3Vcc	<1/3Vcc	1	off	1
1	>2/3Vcc	<1/3Vcc	1	off	1

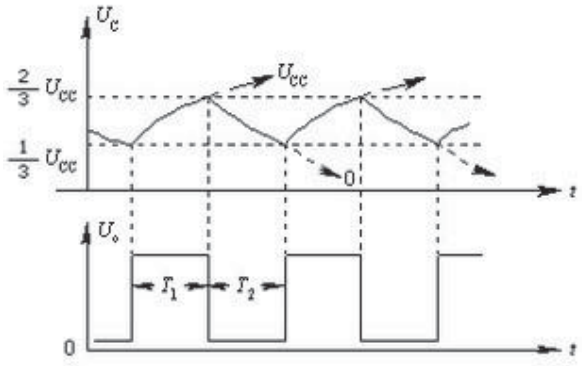

Figure 3.3 Astable waveforms

Figure 3.3 shows the output of the 555 timer at different input voltage. Rd is the reset signal, it has to be high voltage. In actual use, Rd is connected to power supply Vcc. The 555 chip is in an astable mode to form an astable multi-vibrator, it produces a square wave. The frequency is independent of supply voltage. Usually, the frequency of this wave can be captured by microprocessor. The frequency includes two parts, T1 and T2.

The charge time is:

$$T1 = 0.693\ R1 + R2\ C$$

The discharge time is:

$$T2 = 0.693R2C$$

Thus the total time is:

$$T = T1 + T2 = 0.693\ R1 + 2R2\ C$$

The frequency is:

$$f = \frac{1.44}{R1 + 2R2\ C}$$

Given the R1, R2, and frequency, the capacitance is:

$$C = \frac{1.44}{R1 + 2R2\ f}$$

Thus the sensor capacitance is:

$$C_{sen} = \frac{1.44}{R1 + 2R2\ f}$$

For this design, the microprocessor should have a timer module to capture the frequency of the square wave. The frequency is represented as the count number of timer. The key advantage of this circuit is that an analog to digital convertor ADC is not needed.

3.2 Indirect Capacitance to Voltage Convertor

An indirect capacitance to voltage convertor is described (7). From 3.2, the output frequency of 555 is described as:

$$f = \frac{1.44}{R1 + 2R2\ C}$$

For this equation, if R1, R2, and C are fixed, the frequency f is fixed as well. LM2917 is a linear frequency to voltage chip, so LM2917 can be used as a frequency to voltage convertor (FVC), the equation is given:

$$V_{out} = f_{in} C_x R_3\ Vcc$$

Where:

f_{in} - the frequency of output square wave of 555 timer

R_3 - the resistance that is connected to pin 3 of LM2917.

V_{cc} - supply voltage that is connected to LM2917

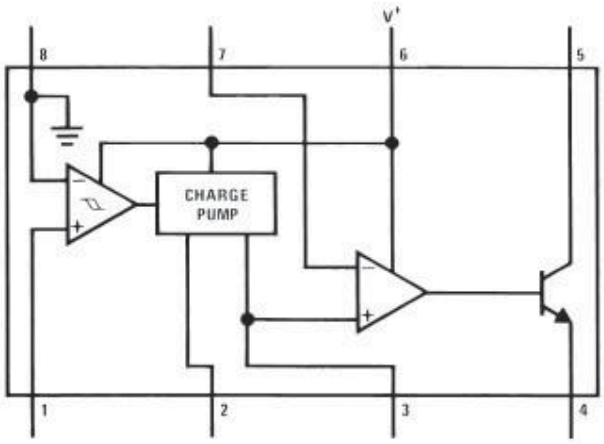

Figure 3.4 TI LM2917 inside diagram (8)

Figure 3.4 is the inside view of LM2917 chip. Pin 3 of 555 is connected to Pin 1 of LM2917. Pin 2 is connected to a capacitance. There are some typical application circuits in the Texas Instruments LM2917 datasheet (8). The indirect capacitance to voltage converter circuit diagram is given:

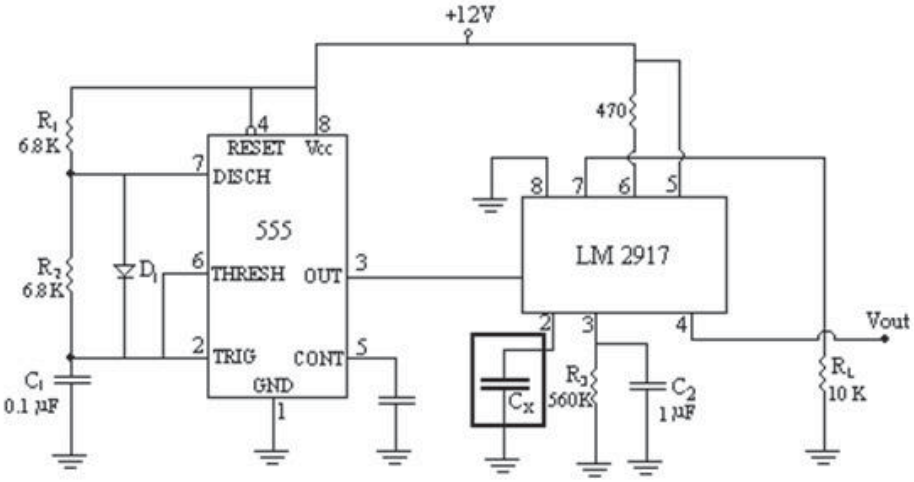

Figure 3.5 Capacitance to voltage convertor (7)

This circuit provides linear relation between capacitance and voltage. This is the advantage of this indirect capacitance to voltage C/V convertor. It outputs a voltage which is directly proportional to the change of measured capacitance. An analog to digital convertor ADC is needed to measure the output voltage.

3.3 Direct Capacitance to Voltage Convertor

Voltage divider is a circuit where two resistors are connected in series. It is a linear circuit has an output voltage that is a fraction of its input signal. It also applies to inductive and capacitive. For the measurement of capacitance, a capacitive divider is introduced to build a direct capacitance to voltage convertor. The capacitance voltage divider schematic diagram is shown below:

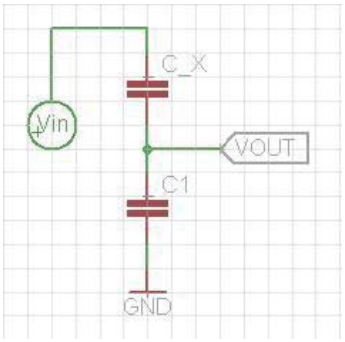

Figure 3.6 Direct capacitance to voltage convertor diagram

Vin should be a sinusoidal signal or AC signal which is applied to the capacitive voltage divider as shown in Figure 3.6, then the output voltage V_{out} can be described by:

$$Vout = \frac{Vin\, C\_x}{(C1 + C\_x)}$$

Where: C_x is the measured capacitance while C1 is a fixed capacitance.

To achieve good linearity, the paper (7) suggests the fixed capacitance C1 must be at least ten times bigger than the C_x. Under this condition:

$$Vout = \frac{VinC\_x}{C1}$$

Thus the C_x equals to:

$$C\_x = \frac{C1}{Vin} Vout$$

Where: V_{out}, V_{in}, and C_1 are known values.

This circuit implies good linear relation between C_x and input signal. A new direct capacitance to voltage convertor is described based on the capacitance voltage divider (7). The schematic diagram is given:

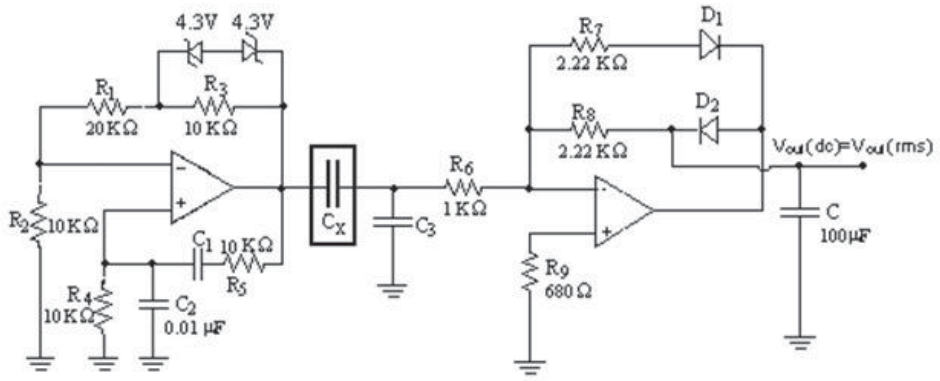

Figure 3.7 Direct capacitance to voltage convertor circuit

The input stage of this circuit is a sine wave generator. It provides the input signal to the capacitance voltage divider. The right part is an AC/DC convertor. The big capacitance C of the AC/DC is a capacitance filter. The rest of the right circuit is a precision half wave rectifier. For precision rectifier, it is useful for precision signal processing as well as very small signal processing. C_x in the range of 1 μF to 1 mF is tested use this circuit. Experiment data shows this direct capacitance to voltage convertor has very good linear characteristic. This method requires an analog to digital convertor to interactive with microprocessor.

3.4 Relaxation Oscillator

Another good circuit for conditioning a capacitive sensor is to change capacitance into oscillation frequency. The similar method is introduced in Microchip's Application

note AN866 and AN895 (10) (11). The oscillator generates a square wave. The change of the wave frequency is a function of the capacitance. Operational amplifier(op amp)is used to build a relaxation oscillator. An ideal op amp is a three terminals device consists of inverting and non-inverting terminal and the output. Transfer characteristic of ideal op amp is given below:

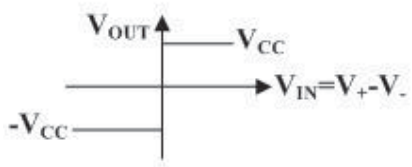

Figure 3.8 Transfer characteristic of an ideal op amp

If the voltage on non inverting terminal is higher than the voltage on inverting terminal, the output of the amplifier goes to high voltage V_{cc}. On the contrary, the voltage on inverting terminal is higher than the voltage on non inverting terminal, the output goes to low voltage $-V_{cc}$. Operational amplifier is widely used for analog signal processing.

A relaxation oscillator is a circuit that combines a comparator, some resistors and one capacitor. Output voltage oscillates from $+V_{cc}$ to $-V_{cc}$ back and forth, as a result, the oscillator generates a square wave. Change of the output voltage automatically charge the capacitor C or discharge it. The relaxation oscillator schematic diagram is given:

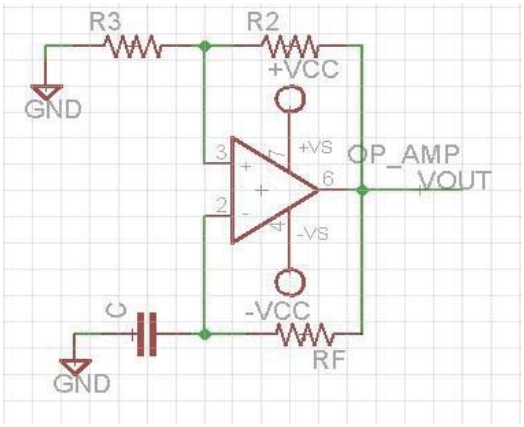

Figure 3.9 Relaxation oscillator circuit

The analysis of this circuit begins by assuming that the output voltage is high ($+V_{cc}$), so the voltage on the non inverting terminal (pin 3) is:

$$Vin+ = Vcc\frac{R2}{R2 + R3}$$

Inverting terminal is on potential:

$$Vin- = -Vcc\frac{R2}{R2 + R3}$$

The output voltage starts to charge the capacitor through the feedback resistor R_f with time constant $\tau = CR_f$. When the voltage across the capacitor becomes slightly higher than the Vin+, the output voltage oscillates from $+V_{cc}$ to $-V_{cc}$, and the voltage on non inverting terminal is:

$$Vin+ = -Vcc\frac{R2}{R2 + R3}$$

This process takes time:

$$T1 = R_fC\ln\left(1 + \frac{2R3}{R2}\right)$$

After this, the voltage across the capacitor discharges through R_f with time constant $\tau = CR_f$.When the voltage across the capacitor becomes slightly lower than the Vin+, the output voltage swings from $-V_{cc}$ to $+V_{cc}$, and the voltage on non inverting terminal

is:

$$Vin+ = Vcc\frac{R2}{R2 + R3}$$

The discharging process takes the same time:

$$T2 = R_f C\ln(1 + \frac{2R3}{R2})$$

Thus the total time:

$$T = T1 + T2 = 2R_f C\ln(1 + \frac{2R3}{R2})$$

The frequency of the output wave:

$$f = \frac{1}{T} = \frac{1}{2R_f C\ln(1 + \frac{2R3}{R2})}$$

If R3 equals to R2 in the circuit, f equals to $1 / (2.2\, R_f C)$. Capacitor value can be calculated based on the frequency.

Online circuit simulator is available(12). For instance, when C equals to 5.501nf, V_{cc} to 12V, R_f to 5K, R3 to 10K, and R2 to 10K, the square wave is shown:

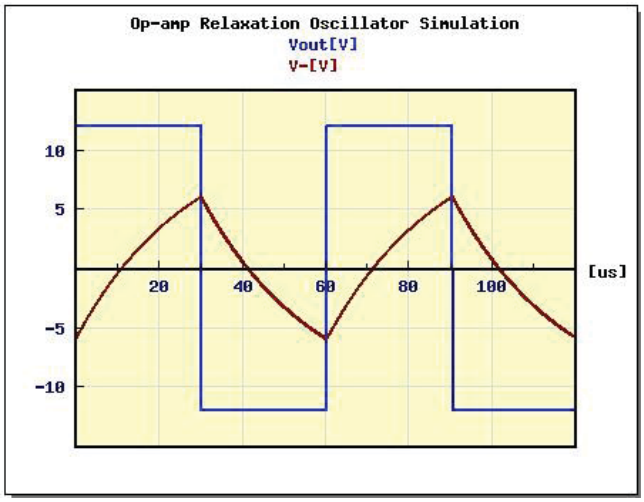

Figure 3.10 Relaxation oscillator waveform

Relaxation oscillator offers some advantages over other circuits. The main advantage

of the oscillator is that an ADC is not required. Another key attribute of oscillators is that these circuits can produce accuracy and resolution that is much better than an analog output voltage circuit (11). The accuracy of the frequency to capacitance conversion is limited only by the accuracy of microprocessor's timer unit. Furthermore, relaxation oscillator only needs a comparator, a few resistors, and one capacitance. It is pretty easy to implement this circuit into actual application. The relaxation oscillator will be a good alternative method for the humidity capacitive sensor measurement in the future.

3.5 Switched Capacitor Circuit

There is another strategy which is illustrated in an application note (AN1014) for the measurement of small change of capacitance. Although this circuit is not appropriate for this project, it is very helpful for my research. It helps me to broaden my knowledge in the area of circuit. This application note describes a switched capacitor circuit, which only needs a microprocessor, minimal external passive components, to measure small changes in capacitance (e.g., 0.001pf)(13).

Analog circuit design most often requires the use of resistors, capacitors, and integrated active devices. It is the nature of integrated circuitry that small, accurate resistors are harder to build and more expensive than capacitors. Given that making capacitors is easier and cheaper, it follows that techniques would be developed to use capacitors to build accurate analog circuitry. These techniques lead to switched capacitor (SC) architectures that control the movement of charge between capacitors

with the precise timing of switches, instead of relying on resistors to move current from one node to another (14). Figure 3.11 shows charge movement for a resistor and for a switched capacitor.

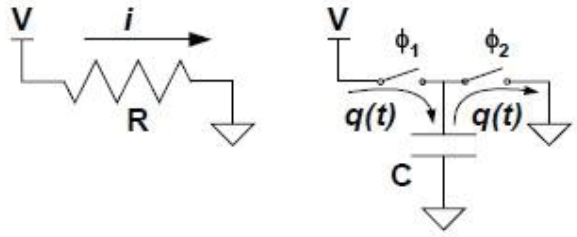

Figure 3.11 Charge movement for R and C (14)

The current flows from a power supply source to ground through a resistor can be expressed as:

$$i = \frac{V}{R} (1)$$

When the switch φ1 is closed, and the switch φ2 is open, the capacitor C is charged to full voltage, the charging process is:

$$q = CV (2)$$

When the switchφ1 is open, and the switch φ2 is closed, all the charge which stores in the capacitor C moves to ground. If these switches are controlled at the rate of f_s, the charge quanta also move at this rate (13). Equation describes the current as:

$$i = \frac{q}{t} = f_s q = f_s CV (3)$$

Based on these three equations, we have:

$$R = \frac{V}{I} = \frac{1}{f_s C} (4)$$

The two switches and the capacitance behave like a resistor whose value is inversely proportional to the capacitance and the switching frequency. The switched capacitor

has a benefit that the resistor's value can be altered by merely changing the switching frequency.

The switched capacitor C_{sen} transfers charge from a voltage source Vs to an integrating capacitor C_{int}. Figure 3.12 shows the schematic diagram for this design. Each time S1 closes and S2 opens, charge is transferred from Vs to C_{sen}, then

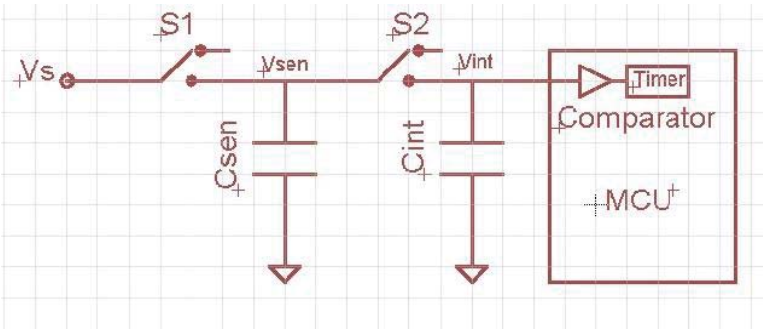

Figure 3.12 Switched capacitor schematic diagram

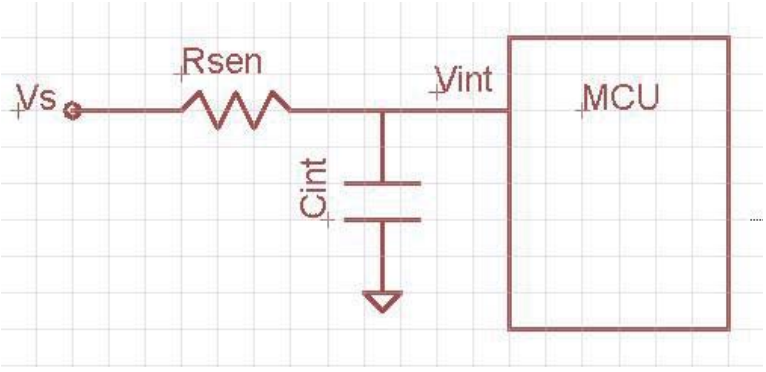

Figure 3.13 Equivalent model for switched capacitor

S1 opens and S2 closes, charge is transferred from C_{sen} to C_{int} to equalize the voltage, which causes the voltage exponentially approaches V_{dd} on both capacitors C_{sen} and C_{int}. The comparator of the MCU monitors the voltage across the C_{int}. Once the comparator captures an event, which means the voltage on C_{int} is higher than the threshold voltage value programmed in the comparator, the timer records the time for the charging process. Based on the capacitor charging equation, R_{sen} can be calculated,

which equals to Tsw / C (Tsw is the switching cycle for the two switches). As a consequence, the capacitor C_{sen} can be measured successful.

Chapter 4 Single Slope A/D Conversion

4.1 Theory of Operation

As the thesis mentioned above, analog to digital conversion is critical to microprocessor based applications. Most real-world signals are analog signals. To implement this conversion, an analog to digital convertor ADC is necessary for this process. However, if an ADC module is not available on a specific chip, it is possible to develop an ADC with the resources inside the chip. Single slop A/D conversion is a technique using the integrated comparator and timer.

Slop A/D conversion actually is an analog to digital convertor, which can be implemented with a comparator, a voltage source, a resistance, a capacitor and a timer. The technique is based on the charge or discharge time of a capacitor with a known value. The number of clock cycles necessary to charge or discharge the capacitor is then counted. Longer charge or discharge times indicate larger voltages. The voltage is derived from the charge or discharge time using the standard equation for capacitor charge or discharge process (15). Some applications use a fixed value for the capacitor for the sake of measuring resistance(16) (17). This technique is valuable in measuring any component that has a characteristic of varying resistance. For measurement of voltage, the relationship is between voltage and time while the resistance is unchanging. For measurement of resistance, the relationship is between resistance and time while the capacitor is constant.

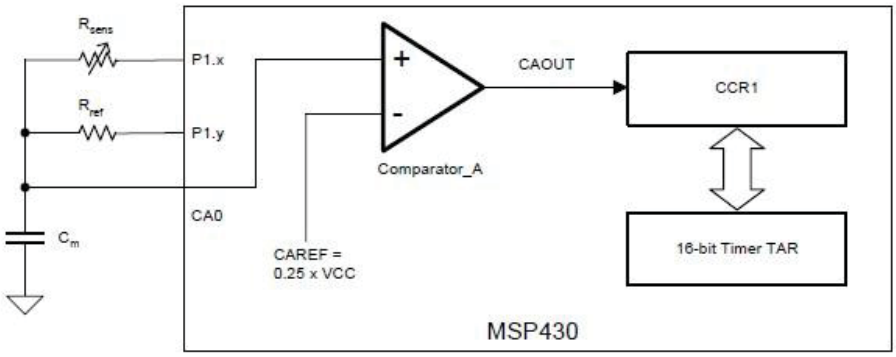

Figure 4.1 Measurement of resistors (15)

Figure 4.1 shows the hardware diagram of a slope A/D conversion for measurement of resistance using MSP430. The circuit measures the resistance of R_{sen} by charging or discharging capacitor C_m. The resistance of R_{ref} is also measured using the same method, and then used as a reference value to calculate R_{sen}.

To measure the resistor value R_{sen}, capacitor C_m is first discharged fully to zero voltage by outputting a low level voltage on either P1.x or P1.y. After configuring the timer, the capacitor is charging through R_{sen} by outputting a high level voltage. At the beginning of charging, register TAR of timer is cleared, and the timer is started to run. When the voltage across capacitor Cm reaches a value of $0.5V_{cc}$ or $0.25V_{cc}$, which equals the comparator reference voltage on the inverting terminal. Once the voltage on non inverting terminal is slightly higher than the voltage on inverting terminal, the positive edge of the comparator output CAOUT causes the TAR to be captured in register CCR1. This value is the charging time for the RC circuit. The reference resistor R_{ref} value is measured in the same process to capture another time. This time will be used to translate T_{sen} into the resistor value R_{sen}. The most important step for

this strategy is that the capacitor should be discharged fully before each measurement.

4.2 Comparator_A+

Comparator is an essential part of single slope A/D conversion, which supports precision slope analog to digital conversion. It cannot perform a conversion by itself, it needs to be used together with timer module to measure time-constant of an external RC circuit. An analog comparator compares the voltages on its two input terminals, *V*+ and *V*-. Its output CAOUT is high if *V*+ is more positive than *V*- and low if *V*+ is lower than *V*-. The comparator can be switch on or off using CAON bit. The output CAOUT only has two states, so the comparator behaves like a 1 bit ADC. The comparator should be turn off when not in use to reduce current consumption.

The comparator module is controlled by two registers, CACTL1 and CACTL2. Figure 4.2 is a block diagram of the Comparator_A+:

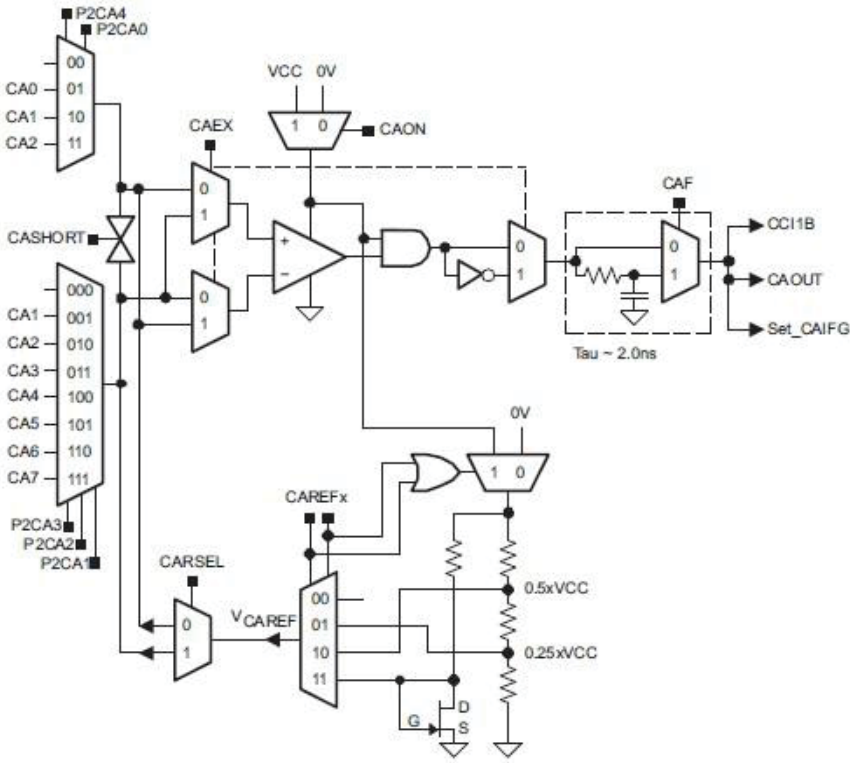

Figure 4.2 Comparator_A+block diagram,

Source: MSP430x2xx datasheet

The non inverting input V+ can be connected to external signals CA0-CA2 or without an external connection when CAEX equals to 0. P2CA4 and P2CA0 control the CA0-CA2.

P2CA3, P2CA2, and P2CA select the inverting input from CA1-CA7 when CAEX equals to 0. CAEX exchanges the comparator inputs and inverts the comparator output. The bit CAREFx selects the reference voltage V_{caref}, which can be chosen from 1/4Vcc, 1/2Vcc or diode reference voltage (about 0.6V). The bit CARSEL decides which terminal the V_{caref} is applied to.

The output is brought to an external pin CAOUT. This voltage level can be measured from outside of chip. The output is also connected to input CCI1B of Timer_A

internally. CCI1B can trigger Timer_A to capture a switch from comparator. This allows precise timing without delays if communication is needed between different modules, for example when Timer_A is used to record the capacitor charging/discharging time. The rising or falling edge of the comparator output can be used to set flag CAIFG.

4.3 Single Slope A/D Conversion Applications

4.3.1 Resistance Measurement

Single slope A/D conversion provides an economic measurement technology to perform precise analog measurement. The slope ADC consists of an on chip comparator, timer, and an external RC circuit. This technique is usually optimized to precisely measure resistive element. The MSP430x2xx datasheet has a temperature measurement system example. Figure 4.3 shows the system diagram:

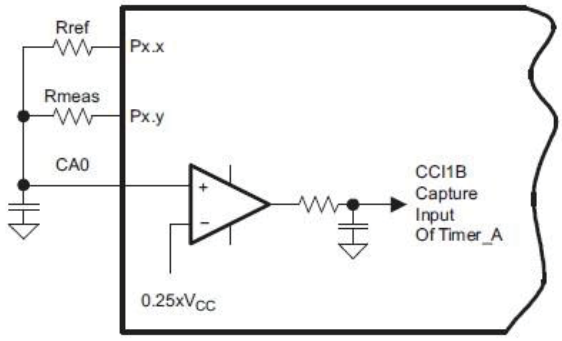

Figure 4.3 Temperature measurement system

source: MSP430x2xx datasheet

The example implements discharge strategy. Figure 4.4 is the timing for the temperature measurement system. Equations show below:

$$\frac{N_{means}}{N_{ref}} = \frac{-R_{means}C\ln\frac{Vref}{Vcc}}{-R_{ref}C\ln\frac{Vref}{Vcc}}$$

$$\frac{N_{means}}{N_{ref}} = \frac{R_{means}}{R_{ref}}$$

$$R_{means} = R_{ref}\frac{N_{means}}{N_{ref}}$$

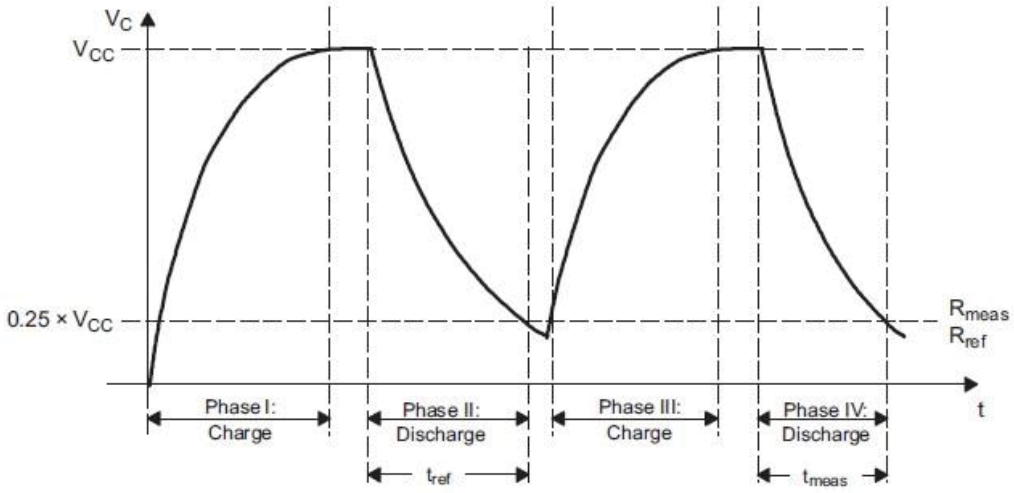

Figure 4.4 Timing for temperature measurement system

source: MSP430x2xx datasheet

The timing diagram shows the charging phase and discharging phase. The T_{ref} represents the discharging time through R_{ref}, and the T_{means} is the discharging time through R_{means}. For each cycle, the capacitors are charged to V_{cc}, and then the capacitors are discharged by setting Px.x or Px.y to low level voltage. This example proves the availability of the slop A/D conversion for resistance measurement.

4.3.2 Capacitance Measurement

The same single slope A/D conversion circuit applies to capacitance measurement. The only difference for capacitance measurement is that the circuit uses a resistor with a fixed value. The resistor and the capacitor form a RC circuit. Here the

capacitor value is changeable. This method is optimized to precisely measure capacitive element, such as capacitive sensor. A proposed slope A/D conversion schematic diagram is shown:

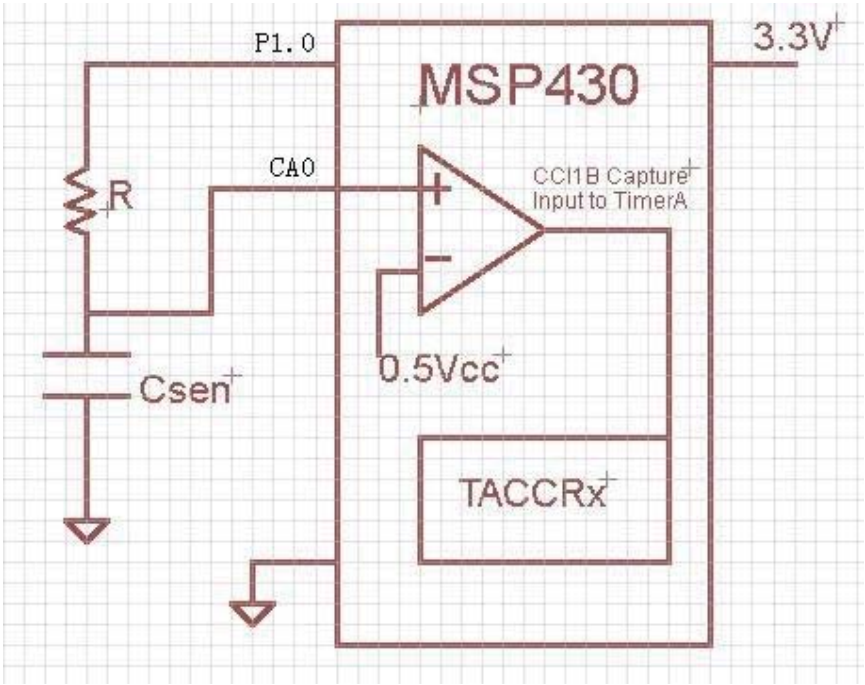

Figure 4.5 Capacitance measurement schematic

The inverting terminal is configured to internal reference voltage $0.5V_{cc}$, the non inverting terminal is configured to CA0, which is the pin for analog input. The internal reference voltage has to be proportional to the V_{cc}. P1.0 charges the RC circuit here by outputting a high level voltage. P1.0 discharges the RC circuit by outputting a low level voltage. There are some different ways to operate this circuit. The C_{sen} represents a humidity sensor, and the R has to be a low temperature coefficient resistor.

4.4 Slope Measurement of Capacitance

The exact same circuit and equations as the resistance measurement can be used when

the single slope A/D conversion technique applies to capacitance measurement. In addition, several different methods can be used to obtain capacitance value: 1) The Timer_A captures the time to charge the capacitance to 0.5Vcc or 0.25Vcc from zero voltage. 2) The Timer_A captures the time to discharge the capacitance to 0.25Vccc from V_{cc}. 3) The Timer_A captures the time to charge and discharge the capacitance between 0.25Vcc and 0.5Vcc.

4.4.1 Charge the Capacitance to 0.5Vcc

For this method, the same circuit schematic as the Figure 4.5 is used. P1.0 of the microprocessor is used to charge the capacitance to 0.5Vcc. CA0 is used to monitor the voltage across the capacitance, which is also configured to connect to the non inverting terminal of the comparator. The inverting terminal of the comparator is programmed to 0.5Vcc. This is the sequences of operations to take measurement by charging the capacitance through the resistor:

1. Configure the comparator and timer correctly.
2. Drive the output pin P1.0 low for a long time to make sure the capacitor is discharged fully. This is a critical step for this method. The voltage across the capacitance has to be zero.
3. Drive the output pin P1.0 high to start to charge the capacitor.
4. Wait until the voltage across the capacitance rises above the reference voltage on the inverting terminal. The output of the comparator switches and generates a capture in TACCR1.

Equation for the capacitor to be charged from zero voltage to V (t):

$$V\ t\ =\ Vcc(1-e^{\frac{-t}{RC}})$$

Where $V\ (t)\ =\ 0.5Vcc$,

$$t = RC\ln\ 2$$

$$C = \frac{t}{Rln2}$$

Where $t\ =\ N\ /\ f$, N is the value captured by timer, f is the frequency of the microcontroller.

$$C = \frac{N}{Rfln2}$$

Partial code for the operations described below:

//

P1DIR |=0x01; //configure P1.0 to output

P1OUT |=0x00;

Delay (0xffff); //discharge C for a long time

/////configure the comparator and timer/////////

CACTL1 = CAON+CAREF_2+CARSE; // Enable comp, V_{ref} = 0.5*V_{cc}

CACTL2 = P2CA4+CAF // Pin 2.4 non inverting terminal to CA0

//CACTL1 |= CAIE; // Setup interrupt for Comparator

TACCTL1=CCIS_1+SCS+CAP+CCIE+CM_1; //set the timer to capture

TACTL|=TACLR; //Clear timer

TACTL = TASSEL_2+ID_0+MC_2;

```
P1OUT |=0x01; //set P1.0 to charge the C
_NOP();
_NOP();
_NOP();
__low_power_mode_0();    //wait for timer interrupt
////////////////////////////////////////////////////////////////
```

4.4.2 Discharge the Capacitance to 0.25Vcc

Another different method can be applied to this circuit. This method measures the capacitance discharging time. First, pin P1.0 charges the capacitance fully to the supply voltage. CA0 monitors the voltage across the capacitance. The sequences of operations to take measurement by discharging the capacitance through the resistor:

1. Configure the comparator and timer correctly.
2. Drive the output pin P1.0 to high level voltage to charge the capacitance fully for a long time. It is better to configure timer in compare mode (reset mode) for the most accurate timing. If the compare mode is selected, the output pin has to be a pin with alternative function of compare output. Otherwise, TAR has to be recorded.
3. Drive the output pin P1.0 to low level voltage to start to discharge the capacitance.
4. Wait until the voltage across the capacitance falls below the reference voltage $0.25*V_{cc}$ on the inverting terminal. The output of the comparator switches and generates a capture in TACCR1.

5. The difference of the two times is the capacitance discharging time.

Equation to describe the discharging process:

$$V\ t\ = Vcce^{-t/RC}$$

Where $V\ (t)\ =\ 0.25Vcc$

$$t = RCln4$$

$$C = \frac{t}{Rln4}$$

Where $t = N / f$, N is the discharging time, f is the frequency of the microcontroller.

$$C = \frac{N}{Rfln4}$$

Partial code for the operations described below:

```
/////configure the comparator and timer/////////

CACTL1 = CAREF_1+CARSE; // Enable comp, Vref = 0.5*Vcc

CACTL2 = P2CA4+CAF // Pin 2.4 non inverting terminal to CA0

//CACTL1 |= CAIE;        // Setup interrupt for Comparator

TACTL|=TACLR;      //Clear timer

TACTL = TASSEL_2+ID_0+MC_2;

For ( ; ; )

{

CACTL1 |=CAON; // turn on comparator

//compare mode, set output high first, charge C fully, reset output, wait for capture

TACCR0 = Charging_Time; //charge fully
```

```
TACCTL0 = OUTMODE_5; //reset output to start discharge
TACCTL1=CCIS_1+SCS+CAP+CCIE+CM_1; //set the timer to capture
///////enter low power mode, wait for timer capture////////
__low_power_mode_0();
Discharging_Time = TACCR1 – TACCR0;//capacitance charging time
}
```

4.4.3 Charge the Capacitance from 0.25Vcc to 0.5Vcc

For this method, the operation is almost the same as the 4.4.1. Both of the $0.25V_{cc}$ and the $0.5V_{cc}$ should be detected for this method. The timer needs to capture both of the output of Comparator_A, and save the charging times. The difference of the two times is the capacitance charging time. The trick for this method is the reference voltage on the inverting terminal has to be changed between the two events. This method offers some advantages. The voltage at which charging starts is not important, as long as the voltage is below $0.25V_{cc}$. This method depends entirely on capture, so only one channel of timer is needed. However, this method has a disadvantage the voltage range from $0.25V_{cc}$ to $0.5V_{cc}$ is too small, which results in a small count number.

Example code is given:

```
//////////////////////////////////////////////////////////////////////////
   P1DIR |=0x01;
   P1OUT &=~BIT0;
   Delay (0xffff); //do nothing, just delay
```

```
  CACTL1 = CAON+CAREF_1+CARSEL; // Enable comp, Vref = 0.25*Vcc
  CACTL2 = P2CA4+CAF;                    // Pin 2.4 to CA0
  CACTL1 |= CAIE;                        // Setup interrupt for Comparator
  TACTL|=TACLR;                          //Clear timer
  P1OUT |=0x01;                          //set P1.0 to charge the C
  _BIS_SR (LPM0_bits + GIE);             // Enter LPM0, interrupts enabled
////Changingre ference voltage in the ISR of Compartor_A+////
/////Comp_A interrupt service routine//////////////////
#pragma vector=COMPARATORA_VECTOR
__interrupt void Comp_A_ISR (void)
{
  CACTL1 = CAON+CAREF_2+CARSEL;          //Vref=0.5Vcc
  TACTL = TASSEL_2+ID_0+MC_2;
  TACCTL1=CCIS_1+SCS+CAP+CCIE+CM_1;      //set timer to capture
  CACTL1 &= ~CAIFG;                      // Clear Interrupt flag
}
```

Chapter 5 Introduction to MSP430F249

In this Chapter, the mixed signal processing chip MSP430F249 is introduced. This microprocessor incorporates a 16-bit RISC CPU, some basic digital and analog peripheral modules, and a flexible clock system. Furthermore, MSP430F249 provides some low power operation modes, which is good to save power when the chip is not in use. The chapter concentrates on the modules which are used in this research.

5.1 MSP430F249 Architecture

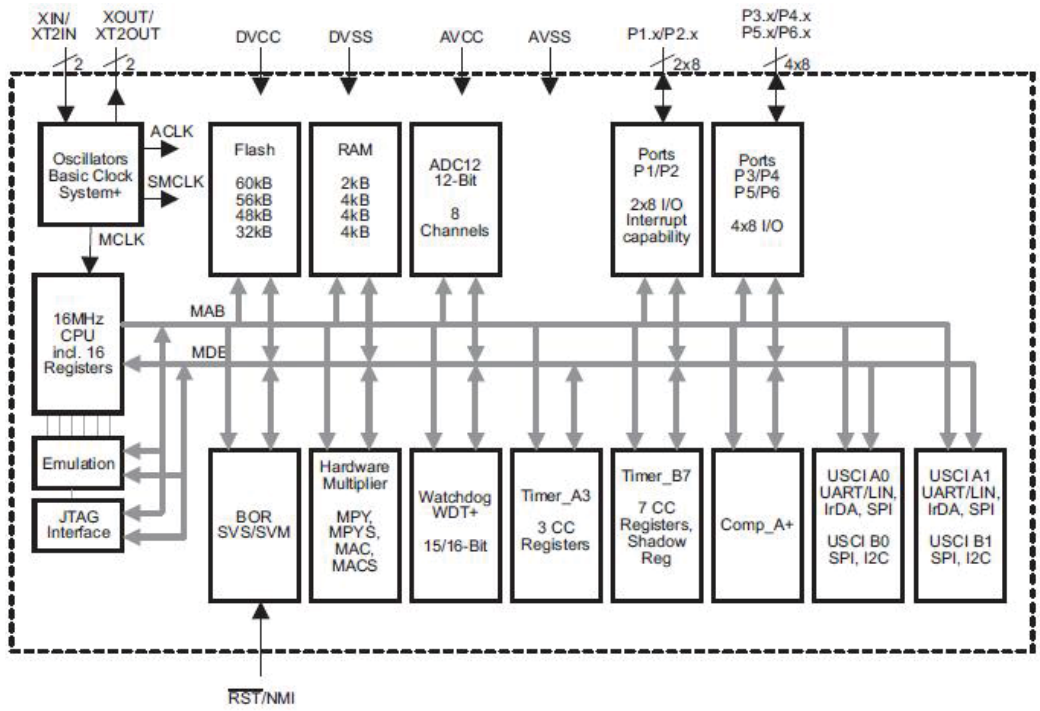

Figure 5.1 MSP430 architecture

source: TI MSP430x2xx datasheet

Figure 5.1 shows the internal architecture of the MSP430F249. MSP430F249 can run at different clock. MCLK determines the system clock. The chip has ADC12 module, Timer_A3 module, and Comparator_A+ module. Those modules have abilities to deal

with analog signal from outside. The single slope A/D conversion is the example which is implemented by the Timer_A3 and the Comparator_A+. The ADC12 can convert analog voltage directly into digital data. The chip has port P1 to port P6, totally 48 pins. JTAG is used to download and debug the chip. Other peripheral modules like watchdog, hardware multiplier, SPI, I2C, etc.

5.2 Basic Clock Module

The clock module is the most important part of a microcontroller. The clock module decides how fast the chip can run. For some 16-bit, 32-bit microcontrollers, they have very complicated clock system. They have both internal and external clock source, high speed and low speed clock source. Software needs to configure the clock system. High speed clock enables microcontroller to have the best performance, while low speed clock usually put chip into low power mode in order to save power.

The basic clock module of MSP430F249 supports low system cost and low power consumption. The basic clock module can be configured to operate without any external components, with only one low speed crystal or with two crystals. The basic clock module has three clock sources, which is shown in the Figure 5.2.

1. LFXT1CLK: Low-frequency/high-frequency oscillator that can be used with 32768 Hz watch crystal or with standard crystal source in 400-kHz to 16-MHz rang.
2. XT2CLK: External high frequency source in 400-kHz to 16-MHz can be selected
3. DCOCLK: Internal digital controlled oscillator (DCO)

Three clock signals are available from the basic clock module:

ACLK: Auxiliary clock. ACLK is selected as LFXT1CLK in MSP430F249. ACLK is divided by 1, 2, 4 or 8.

SMCLK: Sub-main clock. SMCLK is selected as one of LFXT1CLK, XT2CLK, or DCOCLK. It is also divided by 1, 2, 4, or 8.

MCLK: Master clock. MCLK is selected as one of LFXT1CLK, XT2CLK, or DCOCLK. It is also divided by 1, 2, 4, or 8. MCLK is used to clock the CPU and system.

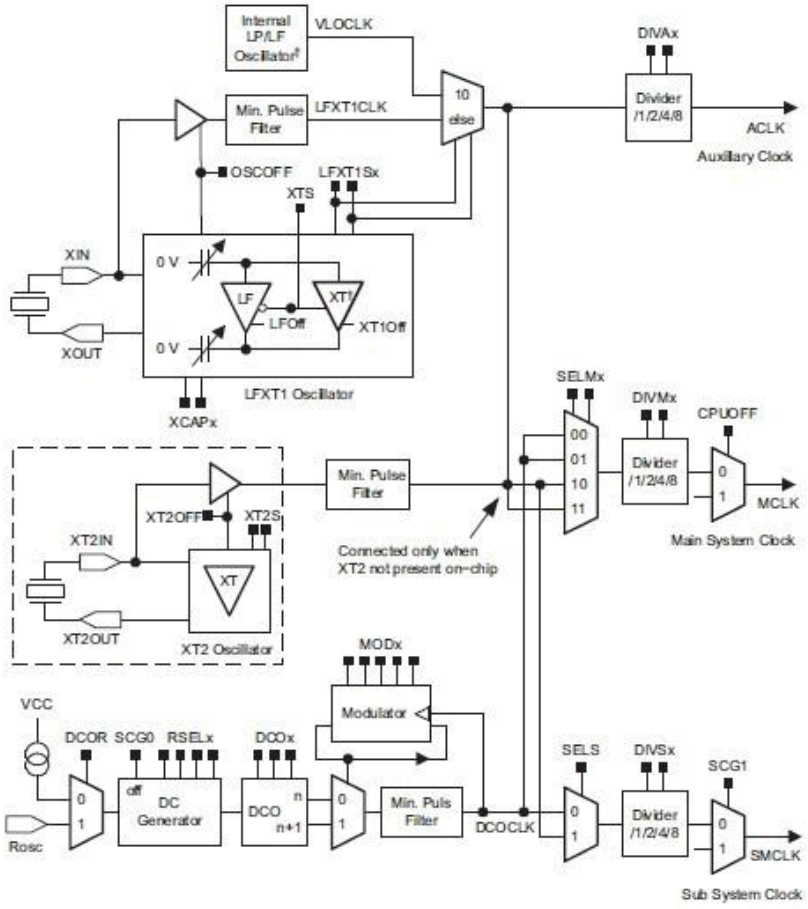

Figure 5.2 MSP430 basic clock module block diagram

source: TI MSP430x2xx datasheet

Register DCOCTL, BCSCTL1, BCSCTL2, BCSCTL3, IE1, and IFG1 are used to configure the basic clock module. Clock is the heart beat for a CPU, the clock system has to be configured right to enable the chip function correctly.

5.3 Comparator_A+

In Chapter 4, the Comparator_A+ has been introduced to form a single slope A/D conversion. The internal architecture and the detailed operation are explained in this chapter. Comparator_A+ control register 1 CACTL1, Comparator_A+ control register 2 CACTL2, and Comparator_A+ port disable CAPD are used to configure the Comparator_A+.

5.4 Timer_A3

Timer_A is a 16-bit timer/counter with multiple capture/compare registers in the MSP430F249 and is included in all devices. Timer_A plays an important role in the slope A/D conversion. This part describes Timer_A and its operation. Timer_A+ consists of two main parts: Timer block and Capture/Compare channels. Figure 5.3 shows the timer block:

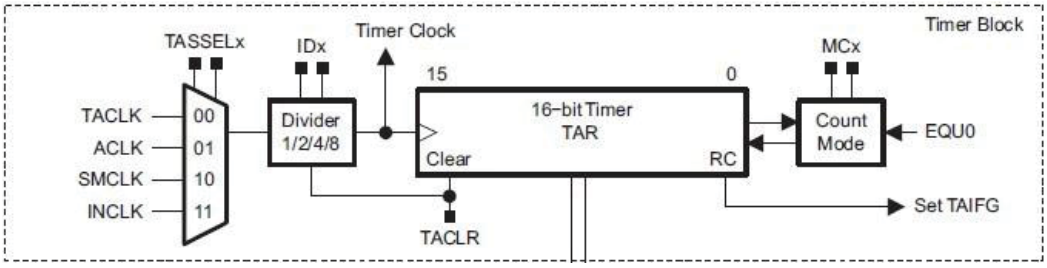

Figure 5.3 Basic timer block

source: MSP430x2xx datasheet

The timer block has different clock sources: TACLK, ACLK, SMCLK, and INCLK. The frequency can be divided down by 2, 4, or 8. The 16-bit TAR register is shared by all the channels in Timer_A. There is no output for timer block, just a flag TAIFG is used to get into an interrupt. MCx configures the timer to four different operation modes: stop mode, continues mode, up mode, or up/down mode. For the stop mode, the timer is halted. The counter runs from 0x0000 to 0xFFFF at continues mode. The up mode is that the counter counts from 0 to a value in TACCR0. For the up/down mode, the counter counts from 0 to TACCR0, and back to 0 again.

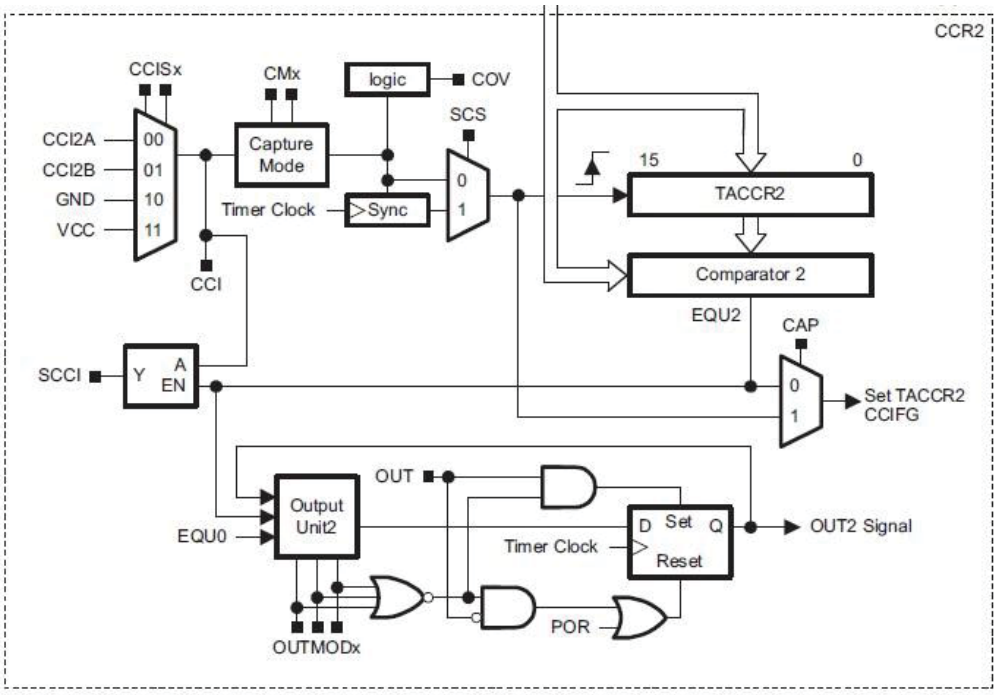

Figure 5.4 Timer_A channel 2 internal diagram

source: MSP430x2xx datasheet

Timer_A has three Capture / Compare channels in most MSP430 chips. Capture mode is used to record the TAR register at which the input changes. The input signal to the TACCR0 can be external or internal. For this research, the output of Comparator_A+

triggers the capture. Compare mode is used to compare the value of TAR register with the value of TACCR0, as long as the two numbers match, an update an output. The figure x above shows the architecture of channel 2. The register TACCTLn controls each channel.

Table 5.1: The TACCTLn register

CMx_2	CMx_1	CCISx_2	CCISx_1	SCS	SCCI	----	CAP
OUTMODx_3	OUTMODx_2	OUTMODx_1	CCIE	CCI	OUT	COV	CCIFG

Table 5.2: Timer_A signal connection for CCI1B

source: MSP430x2xx datasheet

TA0	CCI0A	TACCR0	TA0
TA0	CCI0B		
DVss	GND		
DVcc	Vcc		
TA1	CCI1A	TACCR1	TA1
CAOUT(internal)	CCI1B		
DVss	GND		
DVcc	Vcc		
TA2	CCI2A	TACCR2	TA2
ACLK(internal)	CCI2B		
DVss	GND		
DVcc	Vcc		

In this thesis, the capture mode is used in the slope A/D conversion. For capture mode, there should be an input to the TACCR0. Bits CMx_2 and CMx_1 select the rising edge, falling edge, or either edge. Bits CCISx_2 and CCISx_1 select the inputs CCInA or CCInB to be captured. The internal connection of CCInA and CCInB for MSP430F249 is shown in Table 5.2.

Usually, CCInA is connected to external resource. CCInB is connected to an internal module. Different chips have different connections. This table is the signal connection for MSP430F249. For the slope A/D conversion technique, the output of comparator needs to be captured by timer. Here CCI1B of channel 1 TACCR1 is used, because the CAOUT of comparator is connected to CCI1B. The internal connection enables peripherals to work together more effectively. The biggest advantage of this connection is to avoid delays between different modules. It is important to respond rapidly to capture an accurate time for the slope A/D conversion.

Interrupt is another key part of the timer, which can be generated by the timer block and each capture/compare channel. TACCR0 has its own interrupt vector, TIMERA0_VECTOR. It is the highest vector among the other timer vectors.TIMERA0_VECTOR interrupt service routine code:

```
#pragma vector=TIMERA0_VECTOR
__interrupt void Timer_A (void)
{
//code to execute in the ISR
```

}

TIMERA1_VECTOR is shared by the remaining channels and the time block. For those interrupts, the interrupt service routine (ISR) needs to find out the source of the interrupt. The MSP430 provides an interrupt vector register (TAIV) to identify the source of the interrupt. So when one or more of those shared interrupts is set, TAIV is loaded with the value for the source with highest priority. Table 5.3 listed below is the TAIV values for Timer_A:

Table 5.3: Interrupt vector register TAIV for Timer_A

TAIV values	Source	Flag	Priority
0x0002	Capture/compare channel 1	CCIFG1	Highest
0x0004	Capture/compare channel 2	CCIFG2	Middle
0x000A	Timer overflow	TAIFG	Lowest

Timer_A3 Interrupt Vector (TAIV) handler:

```
#pragma vector=TIMERA1_VECTOR
__interrupt void Timer_A(void)
{
 switch( TAIV )
 {
    case 2:    // code to execute in the ISR
                break;                              // CCR1 not used
    case 4:// code to execute in the ISR
                break;                              // CCR2 not used
    case 10:    // code to execute in the ISR
                 break;// overflow
 }
}
```

The channel 1 TACCR1 is used to capture the output CAOUT of comparator. Once the interrupt flag CCIFG1 is set, which means the output CAOUT switches. Then ISR is serviced, and code in case 2 will be executed.

5.5 Low Power Operation Modes

The MSP430 has one active mode and five low-power modes. The six difference modes are: active mode, low-power mode 0 (LPM0), low-power mode1 (LPM1), low-power mode2 (LPM2), low-power mode3 (LPM3), and low-power mode 4 (LPM4). Bits CPUOFF, OSCOFF, SCG0, and SCG1 in the status register are used to configured the low-power modes 0 to 4.

Table 5.4: MSP430 Low Power Operating Modes

SCG1	SCG0	OSCOFF	CPUOFF	Mode
0	0	0	0	ACTIVE
0	0	0	1	LPM0
0	1	0	1	LMP1
1	0	0	1	LMP2
1	1	0	1	LMP3
1	1	1	1	LMP4

Chapter 6 4-20mA Current Loops

This chapter is dedicated to 4-20mA current loops. 4-20mA is a very common and robust sensor signaling standard. Analog 4–20mA and 10–50mA current loops are commonly used signals for industrial control instruments. 4mA represents the lowest range and 20mA is the highest. The key advantages of the current loop are that the accuracy of the signal is not affected by voltage drop in the interconnecting wiring, and that the loop can supply operating power to the device (18). The Dew Point transmitter has a 1602LCD to display Dew Point and temperature. The Dew Point transmitter also needs to generate a 4-20mA analog current signal for the purpose of control. 4mA represents the lowest Dew Point for this transmitter, and 20mA is the highest Dew Point.

6.1 4-20mA Current Loop Basics

In a 4-20mA current loop, the same current signal flows through the whole system. All the components in the loop have voltage drops due to the current flowing through them. However, the current signal is not affected by the voltage drops as long as the power supply is strong enough than all the voltage drops happen along the loop. This is the main reason why 4-20mA current is chosen as a standard industry signal. The current signal is also insensitivity to electrical noise. At the receive terminal, the analog current signal can be converted into voltage using a precision resistor. Usually, there are four components in a current loop:

1. A DC power supply

2. A 2-wire transmitter

3. A receiver resistor that converts the current signal into a voltage

4. The wire that interconnect it all (19)

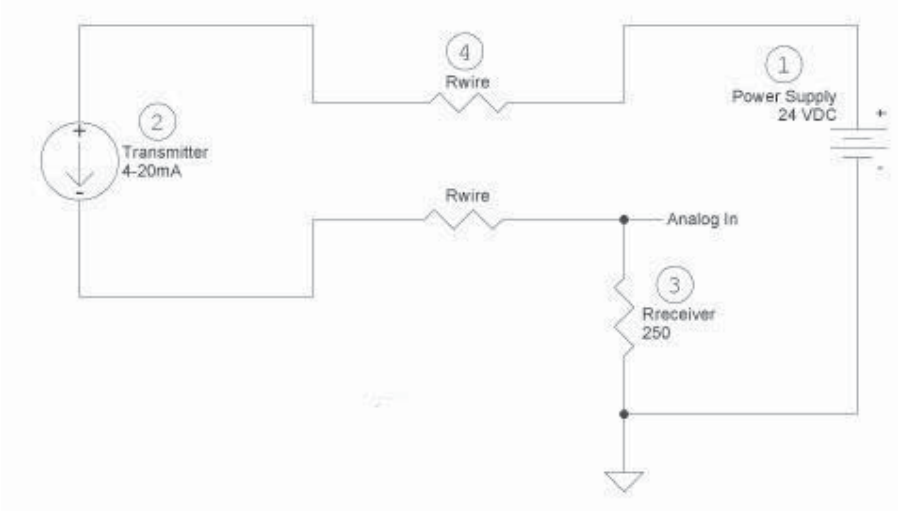

Figure 6.1 Basic current loop (15)

The power supply should be a DC source because the change in current represents the measured parameter. If AC were used, the current in the loop would be changing all the time. This is difficult to measure the exact change of the parameter. For 4-20mA current loop, 36VDC, 24VDC, 15VDC and 12VDC are commonly used.

In Figure 6.1, a precision receiver resistor 250 ohm is used to convert the current signal into a voltage. Voltage is easier to be measured than current signal. For the 250 ohm resistor, the voltage will be 1V at 4mAand 5V at 20mA of loop current. 250 ohm is the most common receiver resistor in a 4-20mA current loop.

The greatest advantage of 4-20ma current loop is the loop's inherent insensitivity to electrical noise. The current transmitter usually has a very large output resistance. This huge resistance can be represented as a series resistance in the circuit.

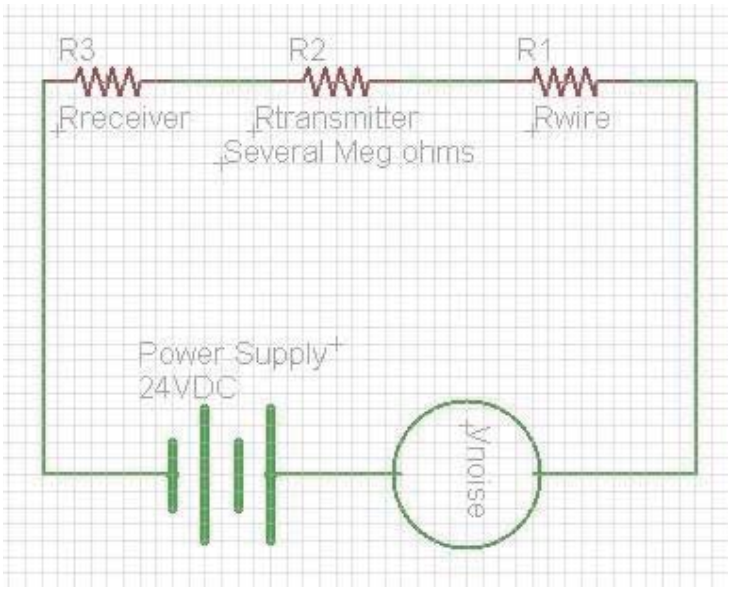

Figure 6.2 Current loop noise model

Figure 6.2 shows the circuit schematic for 4-20mA current loop with a noise source. Due to the high output resistance of the transmitter (in several Meg ohms), the most of the noise voltage is dropped across the transmitter, only a very small fraction is added to the receiver terminal. The noise has almost no influence on the current loop. A noise reduction example is given (19):

Suppose $R_{receiver}$ equals to 250 ohm, R_{wire} equals to 10 ohm, $R_{transmitter}$ equals to 3.64Meg ohm. The noise source has amplitude of 20 Volts, so the voltage across the $R_{receiver}$ is only 0.0014 Volts, which is a very small voltage to the receiver.

$$V_{noise} = 20 * 250 / (3{,}640{,}260) = 0.0014\,V$$

Five of the most basic 4-20mA current loop configurations are described in the 4-20mA current loop configuration application note (20):

1. A two-wire transmitter with external power and an external resistor
2. A two-wire transmitter with controller power and a 4-20mA input

3. A two-wire transmitter with external power and a 4-20mA input

4. A three-wire transmitter with one 4-20mA signal

5. A three-wire transmitter with two independent 4-20mA signals

6.2 Develop 4-20mA Current Loop for Dew Point Transmitter

In thesis, a two-wire transmitter is developed for the Dew Point Transmitter. Texas Instruments has some 4-20mA current loop transmitters which are dedicated to produce analog 4-20mA signals, such as XTR115, XTR116, and XTR110. Another digital to analog convertor DAC7611 is essential to provide an input current to the XTR115/6.

6.2.1 XTR115/6

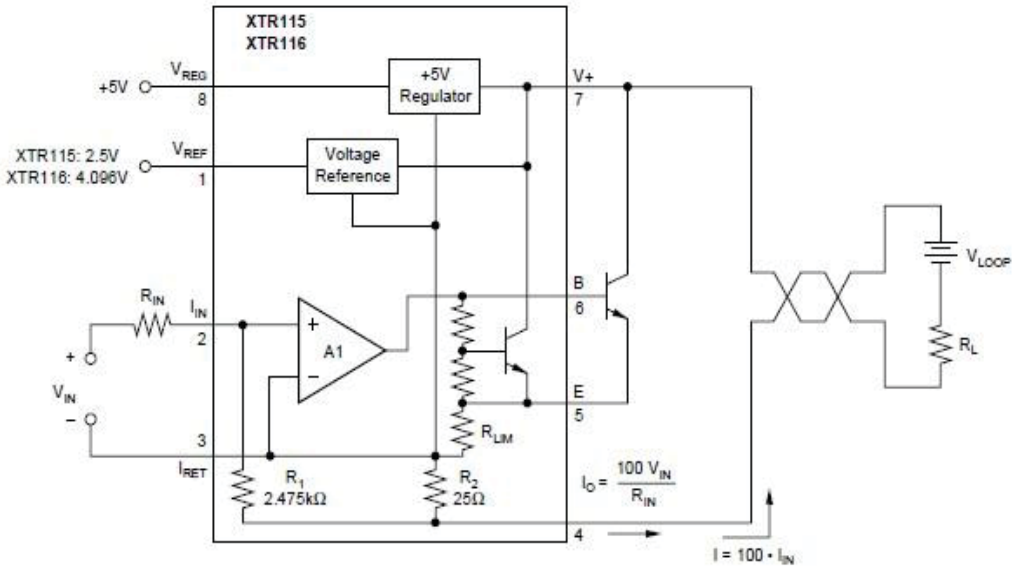

Figure 6.3 XTR115 internal diagram

source: Texas Instruments XTR115 (21)

The +5V on-chip voltage regulator can be used to provide power to external module. An on-chip precision reference voltage can be used as a reference voltage to ADC or DAC. For XTR115, the reference voltage is 2.5V, and for XTR116, the reference voltage is 4.096V. Vin from DAC7611 is applied on pin 2, and generates an input current to the chip. This input current I_{in} controls the output current Io. XTR115/6 is a current input device with a gain of 100:

$$To = 100 * Iin$$

In the Figure 6.3, resistor R1 is 2.475kohm which is connected to the input non inverting terminal. Resistor R2 is 25ohm which is connected to the inverting terminal. The voltage on the non inverting terminal equals to the voltage on the inverting terminal, which leads to

$$I+ * 2.475k = I- * 25$$

Where I+ is I_{in}, I- is the current flows through R2.

The output current Io:

$$Io = Iin + I-$$

Where I- is:

$$I- = \frac{2.475k}{25} * Iin$$

So the output current Io:

$$Io = 2.475k/(25 + 1) * Iin$$

$$Io = 100Iin$$

The XTR115/6 is designed to use virtually any NPN transistor with sufficient voltage,

current and power rating. Case style and thermal mounting considerations often influence the choice for any given application (21). The purpose of the external transistor is to avoid on-chip thermal-induced errors. It is important to locate this NPN transistor away from sensitive analog circuit and the XTR115/6 in order to minimize the heat effects.

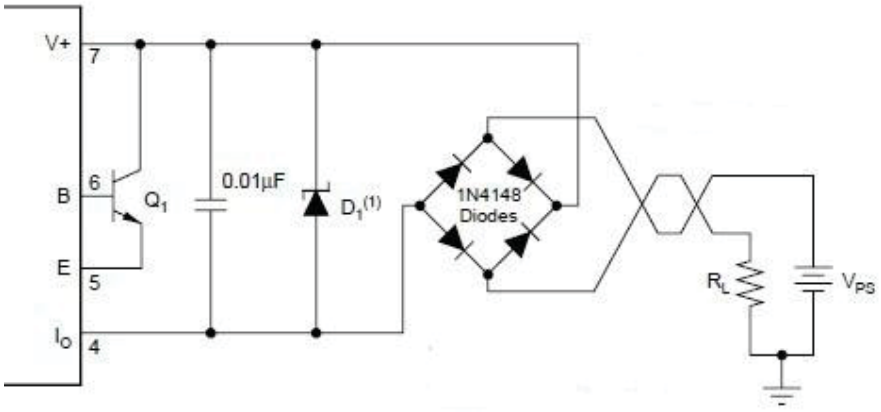

Figure 6.4 Reverse voltage protection and over-voltage surge protection

source: Texas Instruments XTR115 (21)

Figure 6.4 shows reverse voltage protection and over voltage surge protection circuit schematic. The diode bridge circuit acts as a rectifier which guarantees normal operation even when the power is supplied in reverse. The diode bridge causes an approximately 1.4V voltage drop. The zener diode is used to limit the maximum surge voltage applied to the XTR115/6, keep the XTR115/6 operates safely.

6.2.2 DAC7611

DAC7611 is a 12-bit serial input digital to analog convertor chip. It requires a single +5V supply and contains an input shift register, latch, 2.435V reference, DAC, and

high speed rail-to-rail output amplifier (22). The synchronous serial interface is compatible with many DSPs and microprocessors. The serial interface consists of a serial clock (CLK), a serial data (SDI), and a load strobe $\overline{LD}$. $\overline{CS}$ is the chip select input, and $\overline{CLR}$ is the asynchronous clear input.

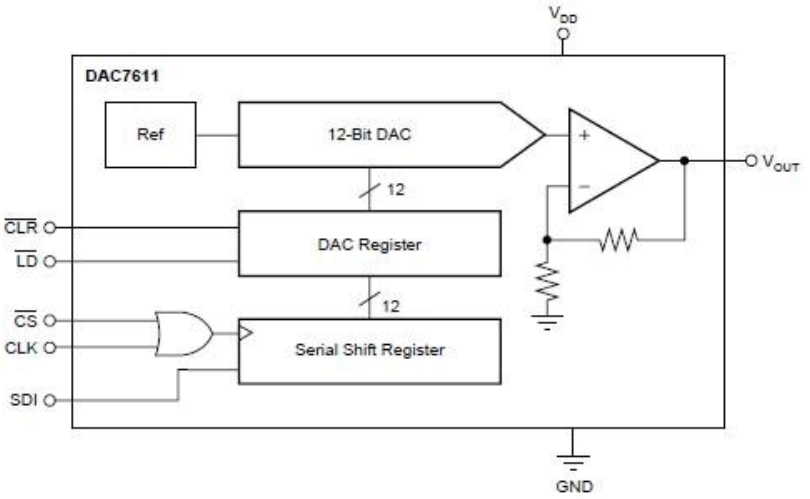

Figure 6.4 DAC7611 diagram block (22)

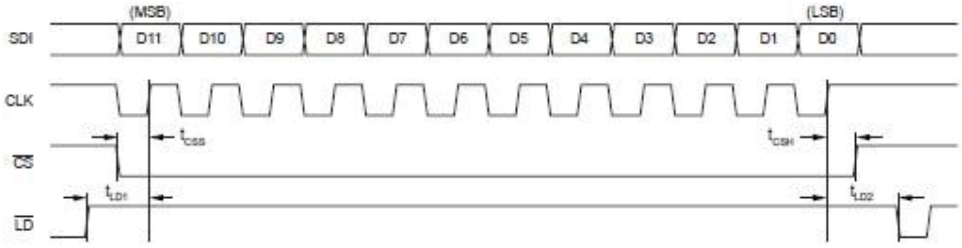

Figure 6.5 DAC7611 timing diagram (22)

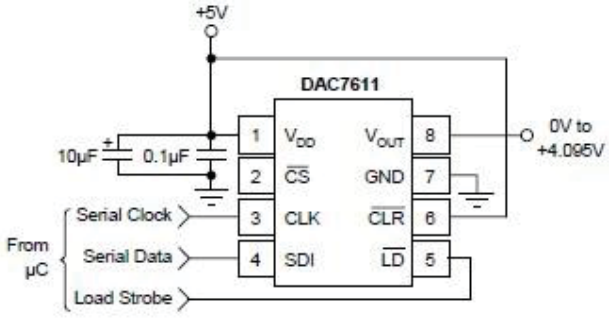

Figure 6.6 DAC7611 basic connection (22)

$\overline{CS}$ has to be low to enable the chip to work. The analog output voltage is 0V if $\overline{CLR}$ is set to low. In practical use, $\overline{CLR}$ is connected to power to keep it high. The 12-bit serial data can be transferred into the DAC register with a high to low transition of the $\overline{LD}$ pin. The timing diagram shows the whole process.

Chapter 7 Temperature Calibration

The humidity sensor is not a temperature independent device. In order to measure Dew Point more accurately, the Dew Point transmitter needs to do temperature calibration. Otherwise the measured value will introduce error to the final result. Another purpose of the temperature calibration is to provide a temperature module to the transmitter, which acts as a temperature indicator for the system. Temperature sensors are widely used to collect temperature data. Different temperature sensors are developed such as thermocouples, resistive temperature sensor (negative temperature coefficient and positive temperature coefficient), infrared radiator, bimetallic devices, liquid expansion devices, molecular change of state and silicon diode (23). The first task of the temperature sensor in this research is to do temperature calibration, which measures the surface temperature of the humidity sensor. A small temperature sensor has to be located close enough to the humidity sensor in the air filter. Based on the previous long term experiment results, the humidity sensor does not have huge temperature drift effect. 1°C is a good resolution to do temperature calibration for the humidity sensor.

7.1 Temperature Sensor Selection

Temperature sensors such as DS18B20, precision thermistor (both NTC and PTC) are discussed. These sensors have good performance and are widely used in real product development. The precision thermistor is suitable for this transmitter development. However, it is not simpler to use the NTC or PTC precision thermistor. Thermistor has

non linear resistance temperature relation. The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor (24).

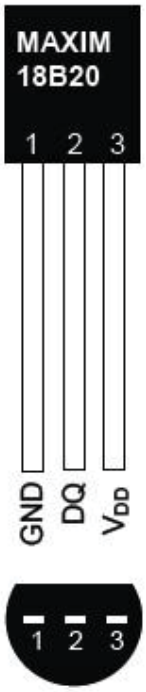

Figure 7.1 DS18B20 temperature sensor (24)

DS18B20 only has three lines to communicate with a central microprocessor. One is GND, one is V_{dd}, and the third one is a data line. The operation temperature range is from -55°C to +125 °C and is accurate to ±0.5°C over the range of -10°C to 85°C. So the DS18B20 is simple to use. It can be used in applications like thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system. However, it cannot be fit into the air filter with the humidity sensor.

DS18B20 is not the ideal temperature sensor for the project even through it offers good advantages.

NTC or PTC precision thermistor is small enough to be fit into the air filter together with the humidity sensor. It offers features like low price, easy to use, high accuracy, high stability, fast thermal response, and long life. A thermistor actually is a type of resistor whose resistance value is temperature dependent, the value varies significantly with temperature. As long as the resistance is measured, the temperature is known based on the RT curve of the thermistor. The schematic diagram to measure the resistance of thermistor is given below:

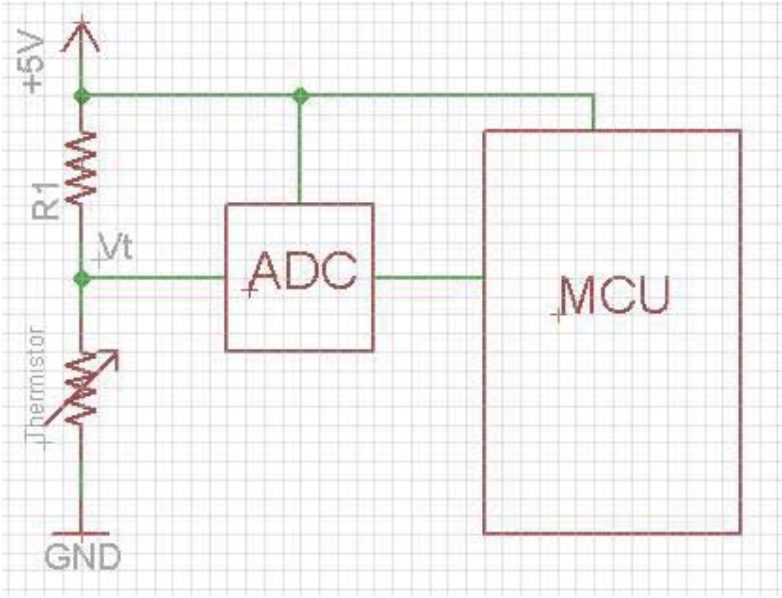

Figure 7.2 Thermistor schematic diagram

R1 and thermistor form a voltage divider. V_t is the voltage across the thermistor, and the R1 is a fixed value low temperature coefficient resistor. So V_t is:

$$Vt = 5V \frac{Rt}{R1 + Rt}$$

The resistance is:

$$R_{thermistor} = Vt\frac{R1}{5V - Vt}$$

Where V_t is measured by microprocessor.

It is easy to measure the resistance of the thermistor with a microprocessor, an analog to digital convertor ADC, and a voltage divider. Each thermistor has a resistance vs. temperature curve (RT curve) shown as below:

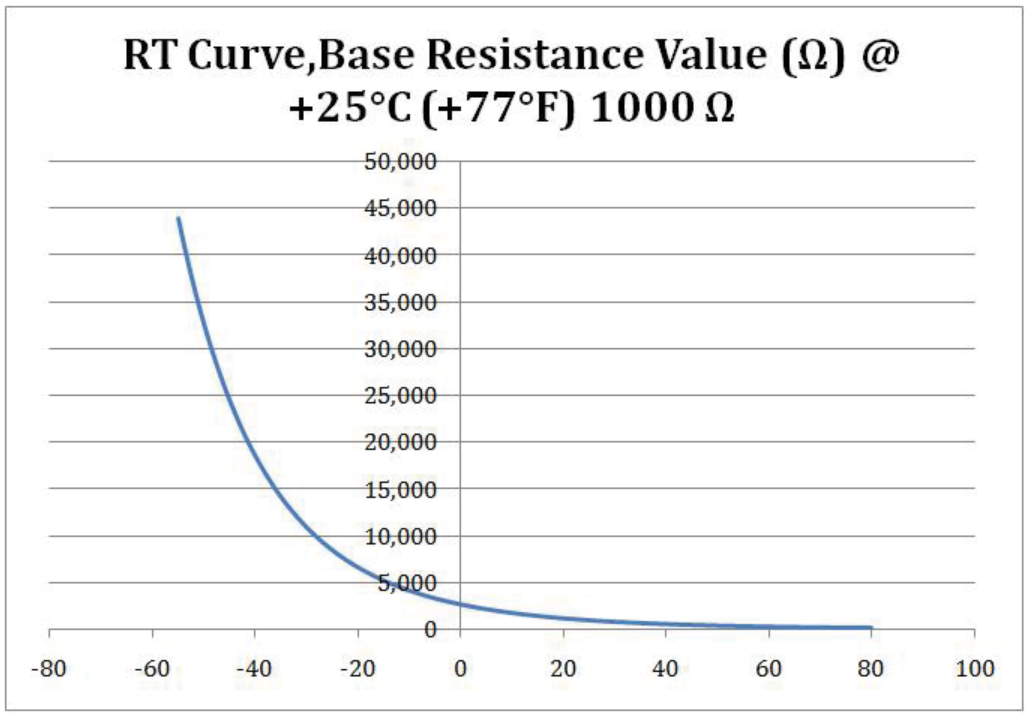

Figure 7.3 RT curve date from U.S. SENSOR

This RT curve is for a precision thermistors from U.S. SENSOR, part number PX102E3. This curve shows the temperature range from -50°C to 80°C. Thermistors have non linear R/T relationship for the whole temperature range. Piecewise linear is one of the common techniques which can be applied to build "linear" relationship between resistance and temperature. Piecewise linear divides the whole temperature range into small temperature ranges. During each small range, the RT relationship can

be treated as linear relationship, and the R/T linear equations can be built. Microprocessor calculates temperature value based on these equations. For the same precision thermistor from U.S. SENSOR, part number PX102E3, the piecewise linear applies to the 0°C to 5°C:

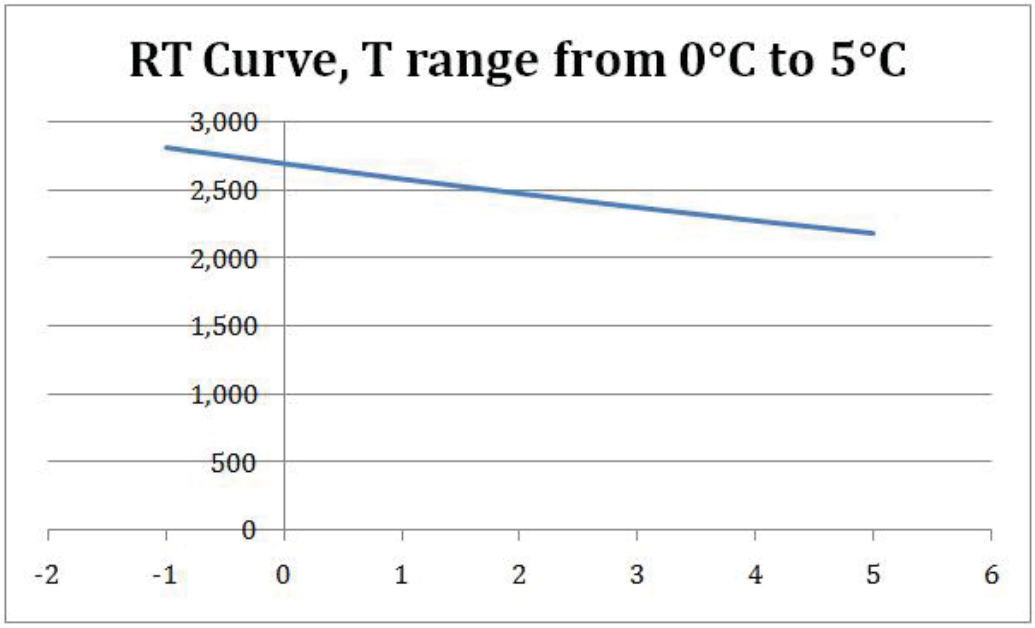

Figure 7.4 Piecewise linear for thermistor during 0°C to 5°C

Figure 7.4 shows a small temperature range from 0°C to 5°C, during the small range, the RT is in linear relationship. The temperature calibration range for the humidity transmitter is from -10°C to 60°C, which means there are 14 small ranges and results in 14 R/T linear equations.

Theoretically, thermistor meets all design requirements. It is a perfect temperature measurement device. However, this strategy cannot be implemented easily. A large amount of time is needed to acquire the equations or R/T table. An alternate way uses just one common diode as the temperature sensor will be described later. The diode

shows very good linear characteristic.

7.2 Diode based Temperature Sensor

A different but simple temperature measurement technique is presented here. This application offers a huge advantage that is the linear relationship between the voltage drop across the diode and the temperature. The temperature coefficient is approximately -2mV/°C for a normal diode. The temperature coefficient is negative, an increase in the temperature reduces the voltage across the diode needed to produce the same current (25). The diode characteristics are calculated by the SPICE model for temperature T=100°C, 27°C, and -50°C to illustrate the temperature effects in (25).

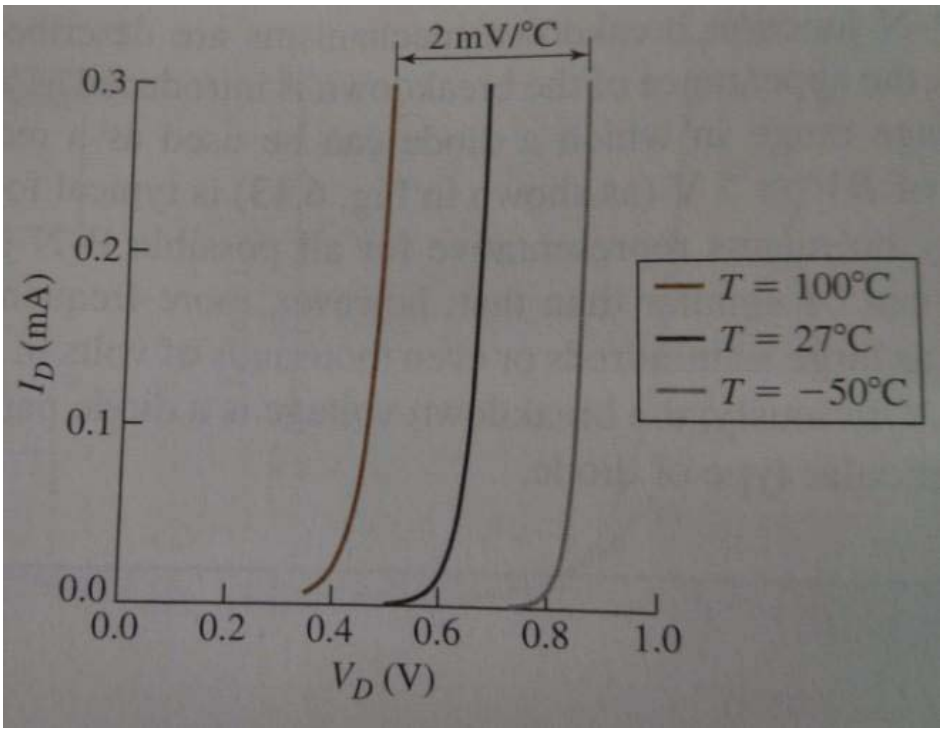

Figure 7.5 Temperature dependence of the diode characteristic (25)

The temperature coefficient -2mV/°C means when the temperature increases by 1°C,

the voltage across the PN junction drops about 2mV. So when the temperature decreases by 1°C, the voltage across the PN junction increases by 2mV. The forward voltage of diode is temperature dependent, and shows good linear relationship between the temperature and the voltage. This characteristic can be used to detect the temperature as long as the voltage across the PN junction is measured. The only problem goes to how to measure the small change of the PN junction voltage accurately. Some good analog to digital convertors ADC can capture the small change of 2mV per degree. However, the whole range of the voltage change is not obviously even the temperature changes dramatically. A big conversion range is desired for software process. The simplest way is to use an op-amp to amplify the voltage across the diode, which leads to a much bigger voltage change. The schematic diagram shows below:

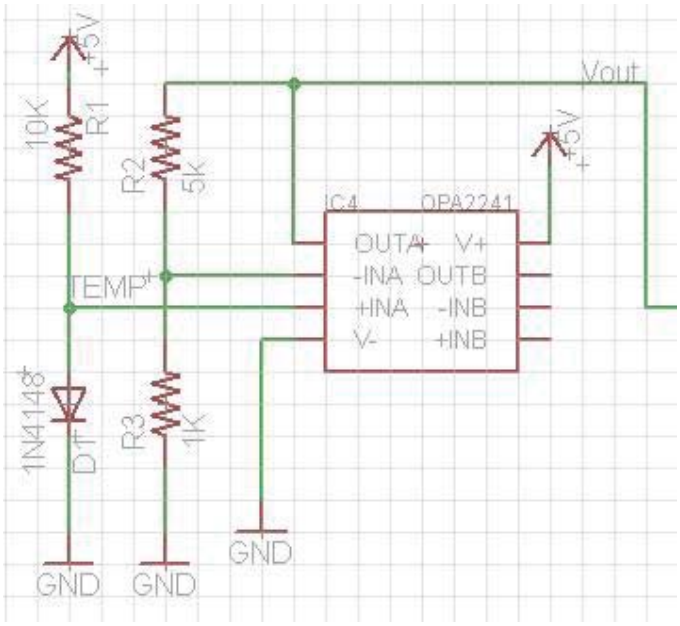

Figure 7.6 Diode based temperature measurement circuit

In Figure 7.6, the fast speed diode is IN4148, which features small shape, fast switching speed and fast reverse recovery. The voltage across the diode is connected to the non-inverting of the op-amp. The voltage +5V supplies power to the R1 and the diode. The close loop negative feedback (R3 and R5) determines the voltage gain (1+R3 / R2) of the entire circuit. In this circuit, the gain is 6. So the small voltage 2mV/°C change across the PN junction produces 12mV/°C on the output terminal of op-amp chip, which is easy to be detected.

The operational amplifier I am using is the OPA2241 from Texas Instruments. It features single supply (+5V), wide supply range, high open loop gain, low offset voltage, and low power consumption. The OPA241 series is designed for portable application, low power supply application. The chip can be operated ordinary under voltage +2.7V to +36V single power supply. For some operational amplifiers, they need to work under dual power supply, positive and negative voltage. The negative voltage usually brings extra components and effect to the design.

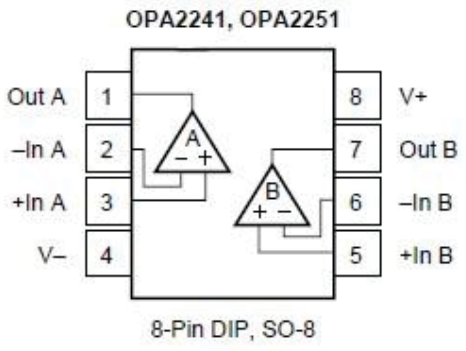

Figure 7.7 OPA224, OPA2251

source: Burr-Brown datasheet

The output of the OPA2411 is still an analog signal. In order to interact with

microcontroller, the analog signal has to be converted into digital value. An analog to digital converter ADS7818 is used together with OPA2241 to convert temperature into digital value.

ADS7818 is a 12-bit high speed successive approximation register (SAR) low power sampling analog to digital converter. It features 2.5 internal reference, low power consumption, single power supply (+5V), serial interface, and mini DIP-8/MSOP-8 package. Low power, small size, and high-speed make the ADS7818 ideal for battery operated systems such as wireless communication devices, portable multi-channel data loggers, and spectrum analyzers (26).

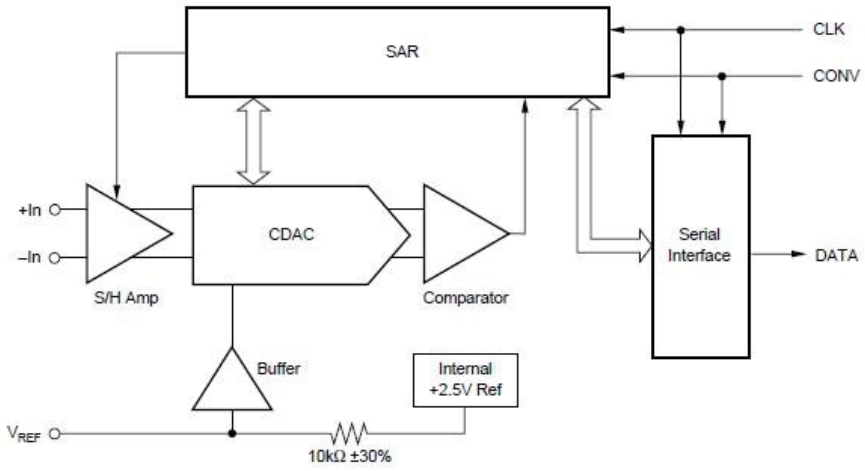

Figure 7.8 ADS7818 diagram block(26)

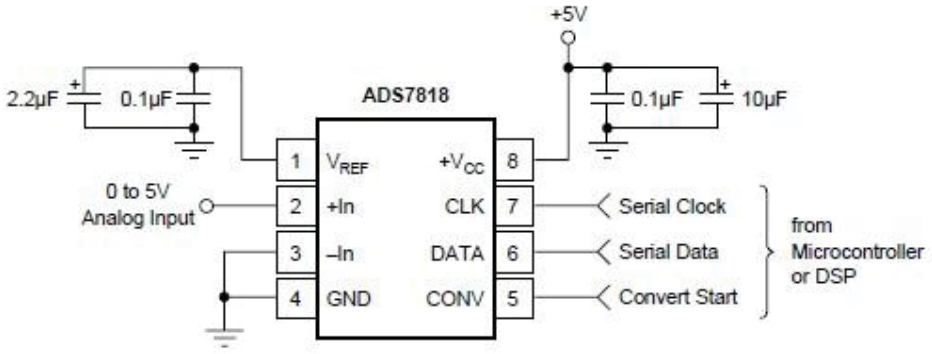

Figure 7.9 ADS7818 basic connection (26)

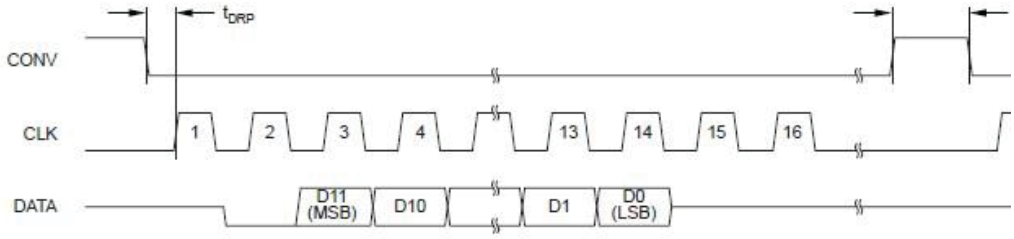

Figure 7.10 ADS7818 SPI interface timing (26)

Pin1 is the 2.5V reference output, which can be used as the input reference voltage resource to DAC. Pin2 is non-inverting input. Pin3 is inverting input, which is connected to ground. CONV is the convert input, DATA is the serial data output, and CLK is the clock input. The ADS7818 requires an external clock source to run the conversion. The voltage on Pin V_{ref} determines the range of the analog input. With the internal 2.5V voltage only, the input analog range is 0-5V. An external voltage 2.0V-2.55V can be placed on Pin V_{ref}, which leads to an input voltage range of 4.0V to 5.1V. The communication between the chip and the microcontroller is in serial manner. CONV and CLK control the conversion process, and the result is provided most significant bit first.

Figure 7.10 shows the typical timing diagram for a serial peripheral interface (SPI).

CLK, DATA, and CONV are connected to the general I/O pins of microcontroller, respectively. The transition from high to low of CONV means the beginning of the conversion. After second clock cycle, the 12-bit are read into microcontroller. The CLK may be kept low or high after the conversion.

Analog input voltage is calculated as:

$$V_{analog} = ADS7818_{output} \frac{FS}{4096}$$

Where FS means full scale $voltage = 2 * V_{ref} = 5.0V$

The temperature is:

$$T = \frac{|(V/n) - V°C|}{2mV}$$

Where n is the gain of OPA2241, which equals to 6 in this design. V°Cis the voltage across the diode at a known temperature and used as the calibration value. 2mV is the temperature coefficient of the diode. V_{analog} is the analog voltage value calculated by ADS7818.

7.3 Self-Heating Effect

The diode based temperature sensor offers several advantages, such as reliable, low cost, linear relationship between temperature and voltage, easy to operate, and decent resolution. When a large current flows through a diode, it will generate heat that will raise the diode temperature. This phenomenon introduces temperature measurement error to the system. The easiest way to solve this problem is to minimize the current flows through the diode. A small current around 1mA is good enough to bias the diode. In this design, a 10k ohm resistor is connected to the diode in series, which

results in a very small current. The self-heating effect is negligible under this condition.

Chapter 8 Study of FreeRTOS Kernel

FreeRTOS is a mini, free, and open source real time operating system from Real Time Engineers Ltd. It is designed to be concise, portable, and simple, specifically for embedded system. Currently, the FreeRTOS supports 34 architectures from 8-bit small MCUs to 32-bit powerful ARM chips. The main part of kernel consists of only three C files, which are list.c, queue.c, and task.c. Those files are all written in C language that makes the FreeRTOS easy to read, to port, and to maintain. In addition, there are a few assembly files for the purpose of adapting to different architectures. FreeRTOS provides basic services such as multi-tasks, binary semaphore, counting semaphore, queue, and memory allocation methods. There are no more advanced features like device driver, advanced memory management, network communication, and file system. FreeRTOS provides pre-emptive and cooperative scheduling policies.

8.1 Tasks in FreeRTOS

FreeRTOS allows an unlimited number of tasks to be run as long as hardware and memory can handle it. As a real time operating system, FreeRTOS is able to handle both cyclic and acyclic tasks. In RTOS, a task is defined by a simple C function, taking a void* parameter and returning nothing (void) (27).

Tasks have priority in FreeRTOS. Tasks with high priority can preempt tasks with low priority if the scheduler is configured in pre-emptive. Tasks have several states in FreeRTOS: running state, blocked state, suspended state, and the ready state. The FreeRTOS scheduler is the only decision maker who puts tasks in the running state, or

not running states. The not running states include all the task states except the running state.

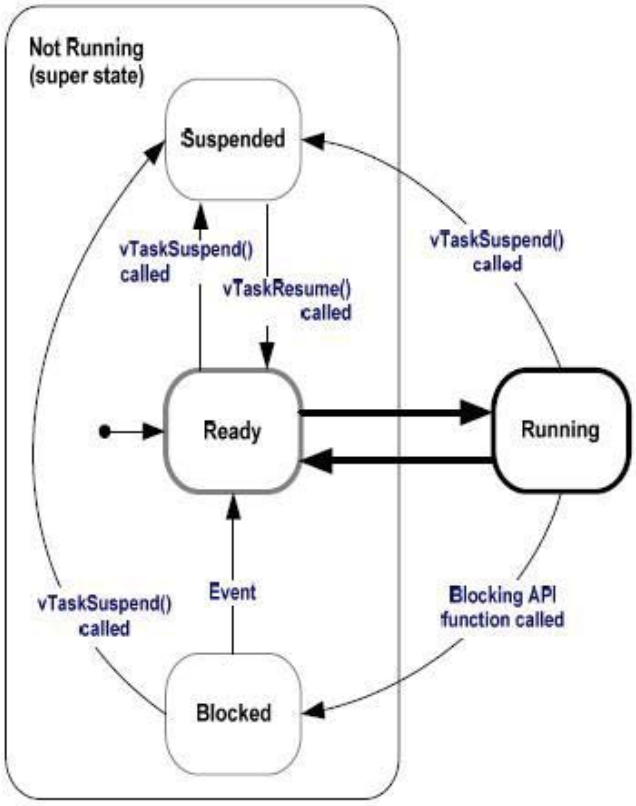

Figure 8.1 Full task state machine(28)

There are several reasons for a task not in the running state. A high priority task preempts low priority task, which makes the low priority task goes into the not running state. When a task is waiting for the CPU, its state is in the ready state. This can happen when a higher priority is using the CPU now. When a task is delayed or in sleep mode, the task is in the blocked state. A task can also be suspended by scheduler. But a call to vTaskResume() can switch the task back to the ready state. Figure 8.1 describes all the transitions between the different task states. FreeRTOS provides all

the necessary API for Task_Create(), Task_Delete(), Task_Suspend(), Task_Delay(), TaskPriority_Set(), etc.

8.2 Scheduling

The scheduler is the most important part of the kernel responsible for deciding which task should be executing at any particular time. The kernel can suspend and later resume a task many times during the task lifetime. The scheduling policy is the algorithm used by the scheduler to decide which task to execute at any point in time (29). FreeRTOS achieves this purpose with priorities given to tasks while they are created. Priority of a task is the only element the scheduler takes into account to decide which task has to be switched in (27).

The maximum number of priorities can be configured in the system configuration head file FreeRTOSConfig.h. FreeRTOS doesn't limit the maximum value of priorities. However, the more priorities are assigned, the more RAM the kernel is consumed. FreeRTOS allows any number of tasks share the same priority. Each task also can be configured to have its unique priority. For the FreeRTOS, low numeric priority means low priority tasks. Priority zero is the lowest priority task. High numeric priority denotes high priority tasks. The scheduler always chooses the highest priority task and puts it in the running state. When more than one task of the same priority is ready to run, the scheduler switches each of them into and out of the running state in turn. For those tasks have the same priority, each task executes for a time slice. Task enters the running state at the beginning of the time slice and exits the

running state at the end of the time slice.

In order to switch between the different priorities tasks, a timer interrupt (the RTOS tick interrupt) enables the scheduler to choose the next highest priority task to run in the tick ISR. Real time kernel uses a variable called tick count to measure the time. Whenever the tick interrupt happens, the tick count increases by one. RTOS tick is the heart beat of the OS. Usually, it sources from external oscillator. So the tick provides accurate timing to the RTOS tick interrupt. The tick rate is configured by configTICK_RATE_HZ in the FreeRTOSConfig.h file. For example, if the tick rate is set to 1000 (Hz), then the tick interrupt is generated each 1ms.

Each time the tick count is incremented the real time kernel must check to see if it is now time to unblock or wake a task. It is possible that a task woken or unblocked during the tick ISR will have a priority higher than that of the interrupted task. If this is the case the tick ISR should return to the newly woken/unblocked task - effectively interrupting one task but returning to another (29).

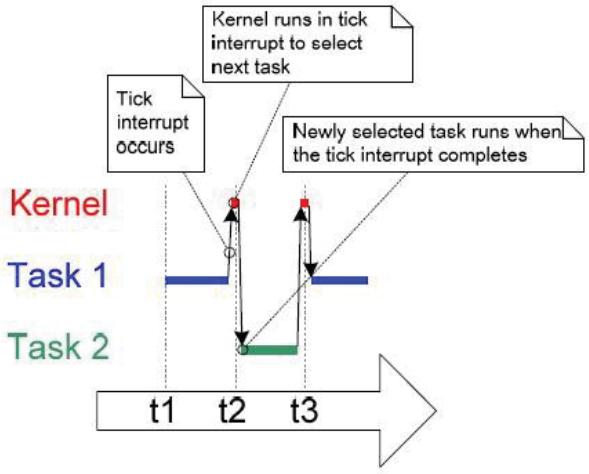

Figure 8.2 The tick interrupt makes the scheduler to run a higher priority task(28)

In the tick ISR, the scheduler puts the higher priority task2 in the running state. After the completion of the task2, task2 enters into not running state. The tick ISR returns task1 to the running state.

8.3 Inter-task communication

FreeRTOS provides several ways to enable tasks communicate with each other, such as queue, binary semaphore, counting semaphore, etc. A queue can store a finite number of fixed size data. They have a length which is the maximum number of items that the queue can hold. Both the data size and the queue length are given when the queue is created. Queue can be accessed by multiple tasks. Some tasks can write (copy) data to the queue and some tasks can read (copy) data from the queue. This is a very important mean for multiple tasks to talk to each other.

Binary semaphore is the easiest way to synchronize tasks. A binary semaphore uses the existing queue mechanism. The binary semaphore actually is a special queue, the length is 1 and the data size is 0. It can be used to unblock a task each time a particular interrupt occurs, effectively synchronizing the task with the interrupt. This allows the majority of the interrupt event processing to be implemented within the synchronized task, with only a very fast and short portion remaining directly in the ISR (28).

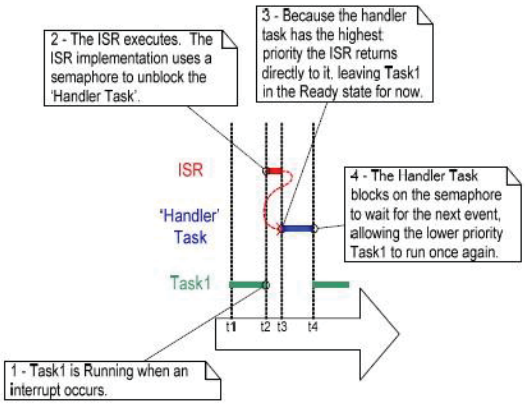

Figure 8.3 The implementation of binary semaphore to synchronize the task with the interrupt(28)

In Figure 8.3, the handler task has higher priority than task1. It processes data from the ISR. The scheduler needs to synchronize the ISR handler task with the ISR. An interrupt happens to interrupt the running task1. In the ISR, the ISR implementation uses a binary semaphore to unblock the handler task. The ISR returns directly to the handler task if the handler task has the highest priority.

Counting semaphore has the same mechanism as the binary semaphore. It can be thought as a queue that has a length more than one. The counting semaphore can be taken several times before it becomes unavailable. It has a count value which is increased when the semaphore is given, and decreased when the semaphore is taken. As long as the semaphore is unavailable, tasks can't take semaphores any more, they have to be blocked and wait for the available semaphore. In the figure x, a counting semaphore is created without any available semaphores. An interrupt gives one semaphore to the count, which adds value one to the count. The task takes the semaphore leads to the unavailability of the semaphore. Another two interrupts

happen while the task is processing the last event. The ISR gives two semaphores to the semaphore count. After the completion of the last event, the task takes another semaphore without entering the blocked state. With the counting semaphore, tasks would not miss any interrupt events.

8.4 Introduction to some Basic FreeRTOS API

Create a task:

```
portBASE_TYPE xTaskCreate(
                                    pdTASK_CODE pvTaskCode,
                                    const portCHAR * const pcName,
                                    unsigned portSHORT usStackDepth,
                                    void *pvParameters,
                                    unsigned portBASE_TYPE uxPriority,
                                    xTaskHandle *pvCreatedTask
                                    );
```

This API is used to create a new task and add it to the ready to run task list. PvTASK_Code is the pointer to the task entry function. pcNAME is the name for the task. usStackDepth is the size of the task stack. pvParameters is the pointer that will be used as the parameter for the task being created. uxPriority is the priority for the task. pvCreatedTask is used to pass back a handle by which the created task can be referenced.

Delete a task:

Void vTaskDelete (xTaskHandle pxTask);

This API is used to remove a task from RTOS. The task will be removed from all ready, blocked, suspended and even lists.

Delay a task:

Void vTaskDelay (portTickType xTicksToDelay);

Void vTaskDelayUntil (portTickType *pxPreviousWakeTime,

portTickType xTimeIncrement);

Those API are used to delay a task. The first API delays a task for a given number of ticks. The actual time delay time depends on the tick rate. The second API can delay a task for a specific time.

Start scheduler:

Void vTaskStartScheduler (void)

This API is used to start the real time kernel tick processing. The idle task is created automatically when vTaskStartScheduler() is called.

Stop scheduler:

Void vTaskEndScheduler (void)

This API is used to stop the real time kernel tick. All tasks will be automatically deleted and the scheduler will stop.

Create a Queue:

xQueueHandle xQueueCreate(

unsigned portBASE_TYPE uxQueueLength,

```
                                    unsigned portBASE_TYPE uxItemSize
                                    );
```

This API is used to create a new queue. uxQueueLength is the maximum number of items that the queue can hold. uxItemSize if the number of bytes each item in the queue needs.

Write to a Queue:

```
portBASE_TYPE xQueueSend(
                                    xQueueHandle xQueue,
                                    const void * pvItemToQueue,
                                    portTickType xTicksToWait
                                    );
```

This API writes an item in a queue. xQueue is the handle to the queue on which the item is to be posted. pvItemToQueue is a pointer to the item that is to be placed on the queue. xTickToWait is the maximum time to wait before the queue becomes available.

Read from a Queue:

```
portBASE_TYPE xQueueReceive(
                                    xQueueHandle xQueue,
                                    void *pcBuffer,
                                    portTickType xTicksToWait
                                    );
```

This API reads an item from a queue. xQueue is the handle to the queue on which the item is to be posted. pcBuffer is a pointer to the buffer where the read item will be placed to. xTicksToWait defines the maximum time to wait.

Create a semaphore:

vSemaphoreCreateBinary(xSemaphoreHandle xSemaphore);

xSemaphore to be created.

Take a semaphore:

```
xSemaphoreTake(
                    xSemaphoreHandle xSemaphore,
                    portTickType xBlockTime
                    );
```

xSemaphore is the handle to the semaphore. This is returned by vSemaphoreCreateBinary(). xBlockTime is the time, in clock ticks, to wait for the semaphore to be available.

Give a semaphore:

xSemaphoreGive(xSemaphoreHandle xSemaphore);

xSemaphore is the handle to the semaphore being released. This is returned by vSemaphoreCreateBinary().

Create a counting semaphore:

```
xSemaphoreHandle xSemaphoreCreateCounting(
                                    unsigned portBASE_TYPE uxMaxCount,
```

unsigned portBASE_TYPEuxInitialCount

);

uxMaxCount is the maximum number of semaphores can be taken. uxInitialCount is the initial value after the counting semaphore is created.

8.5 Port FreeRTOS to MSP430F249

FreeRTOS is a concise, portable, and simple real time operating system. It is designed specifically for small embedded system. It can be ported to more than 30 architectures. FreeRTOS has demo projects for different architectures. Westmoreland Engineering also provides a demo project to port FreeRTOS to MSP430F169 (30).

```
FreeRTOS
    |
    +-Demo          Contains the demo application source and projects.
    |
    +-Source        Contains the implementation of the real time kernel.
        |
        +-tasks.c   One of the three core kernel files.
        +-queue.c   One of the three core kernel files.
        +-list.c    One of the three core kernel files.
        +-portable  The sub-directory that contains all the port specific files.
```

Figure 8.4 The top level directories (28)

The core FreeRTOS source code can be downloaded from the FreeRTOS.org website. The three core kernel C files are located under the source directory. They are common for all ports. Nothing needs to be modified to them. The port specific files are located under the portable directory too. These files are complier and architecture related. Those files are located within the FreeRTOS\Source\Portable\ [complier]\ [architecture] directory, where [complier] is the tool chain being used and [architecture] is the microcontroller being used. For this thesis, MSP430 and IAR are

used. So the port specific files can be found under FreeRTOS\Source\Portable\ IAR\ MSP430. They are port.c, portasm.h, portext.s43, and portmacro.h.

In order to port FreeRTOS to MSP430F249, the three core C files and the three port specific files should be included to the project. In addition, there are some places need to be modified. The head file FreeRTOSConfig.h should include msp430x24x.h instead of msp430x4x.h.

```
#ifndef FREERTOS_CONFIG_H
#define FREERTOS_CONFIG_H
#include <msp430x24x.h>//////////use msp430x24x.h here//////////////////
#define configINTERRUPT_EXAMPLE_METHOD 1
#define configUSE_PREEMPTION     1
#define configUSE_IDLE_HOOK      1
#define configUSE_TICK_HOOK      0
#define configCPU_CLOCK_HZ       ( ( unsigned long ) 7995392 ) /* Clock setup from main.c in the demo application. */
#define configTICK_RATE_HZ       ( ( portTickType ) 1000 )
#define configMAX_PRIORITIES           ( ( unsigned portBASE_TYPE ) 5 )
#define configMINIMAL_STACK_SIZE   ( ( unsigned short ) 100 )
#define configTOTAL_HEAP_SIZE        ( ( size_t ) ( 1800 ) )
#define configMAX_TASK_NAME_LEN      ( 8 )
#define configUSE_TRACE_FACILITY          0
#define configUSE_16_BIT_TICKS            1
#define configIDLE_SHOULD_YIELD           1

/* Co-routine definitions. */
#define configUSE_CO_ROUTINES        0
#define configMAX_CO_ROUTINE_PRIORITIES ( 2 )
/* Set the following definitions to 1 to include the API function, or zero
to exclude the API function. */
#define INCLUDE_vTaskPrioritySet                 0
#define INCLUDE_uxTaskPriorityGet                0
#define INCLUDE_vTaskDelete                      0
#define INCLUDE_vTaskCleanUpResources            0
#define INCLUDE_vTaskSuspend                          0
#define INCLUDE_vTaskDelayUntil                  1
#define INCLUDE_vTaskDelay                       1
```

```
#define INCLUDE_uxTaskGetStackHighWaterMark                0
#endif /* FREERTOS_CONFIG_H */
```

The MSP430 chip needs to provide a RTOS tick rate to the kernel. In the demo project, they use Timer_A0 to generate the tick interrupt. However, in my project, the Timer_A0 is used for other purpose. So Timer_B0 is used instead of Timer_A0. First, the Timer_B0 has to be initialized in the port C file.

```
/*
 * Hardware initialisation to generate the RTOS tick.    This uses Timer_B0
 * but could alternatively use the watchdog timer or timer 1.
 */
void vPortSetupTimerInterrupt( void )
{
    /* Ensure the timer is stopped. */
    TBCTL = 0;

    /* Run the timer of the ACLK. */
    TBCTL = TASSEL_1;

    /* Clear everything to start with. */
    TBCTL |= TACLR;

    /* Set the compare match value according to the tick rate we want. */
    TBCCR0 = portACLK_FREQUENCY_HZ / configTICK_RATE_HZ;

    /* Enable the interrupts. */
    TBCCTL0 = CCIE;

    /* Start up clean. */
    TBCTL |= TACLR;

    /* Up mode. */
    TBCTL |= MC_1;
}
/*-----------------------------------------------------------*/
```

After initialization of Timer_B0, Timer_B0 has to be enabled to generate interrupt. This is modified in the porttext.s43 file.

```
/*-----------------------------------------------------------*/
/* Install vTickISR as the timer B0 interrupt. */
    //ASEG
COMMON INTVEC:CODE:ROOT(1)
ORG TIMERB0_VECTOR
DW vTickISR


/*
 * The RTOS tick ISR.
 *
 * If the cooperative scheduler is in use this simply increments the tick
 * count.
 *
 * If the preemptive scheduler is in use a context switch can also occur.
 */
vTickISR:
    portSAVE_CONTEXT

    call #vTaskIncrementTick

    #if configUSE_PREEMPTION == 1
        call #vTaskSwitchContext
    #endif

    portRESTORE_CONTEXT
/*-----------------------------------------------------------*/
```

The vTickISR is used to switch context of tasks. This is the key part of scheduler when a high priority task needs to preempt a low priority task. The whole process is accomplished by hardware.

Chapter 9 Implementation and Testing

A prototype dew point meter can be built based on the previous 8 chapters. The meter has one 1602LCD module, one capacitance measurement module, one 4-20mA current loop module, one temperature measurement module and one humidity calibration module. Each module is deeply elaborated above except the humidity calibration module. The calibration part is not covered in this thesis.

9.1 Firmware Architecture and Flowchart

Totally, the firmware of this system can be separated into 6 parts: one main file, 1602LCD driver, sensor capacitance measurement, ADS7818 driver, DAC7611 driver and dew point calibration section.

The main file is used to initialize hardware, create tasks, and start task scheduler. For the rest modules, each of them has a header file and an executable body. The header provides an interface between different modules.

main.c

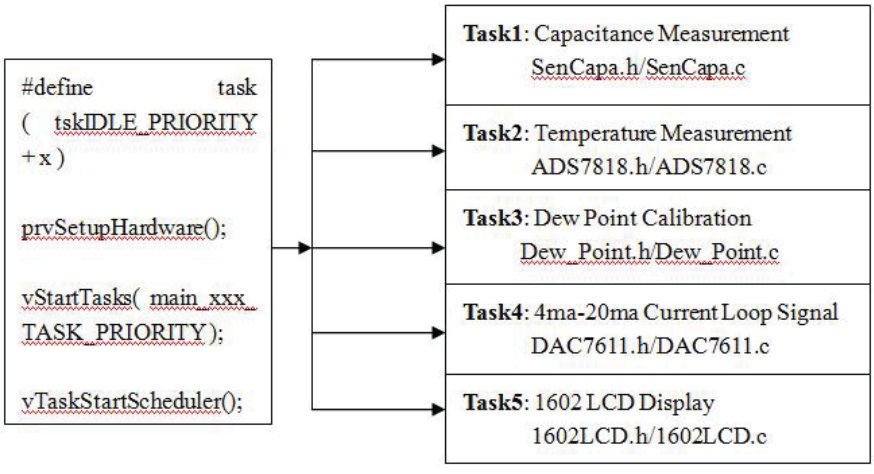

Figure 9.1 Firmware architecture

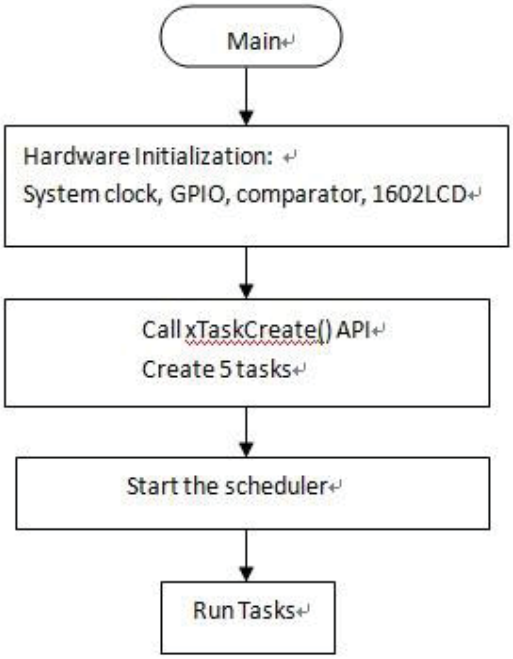

Figure 9.2 Main flowchart

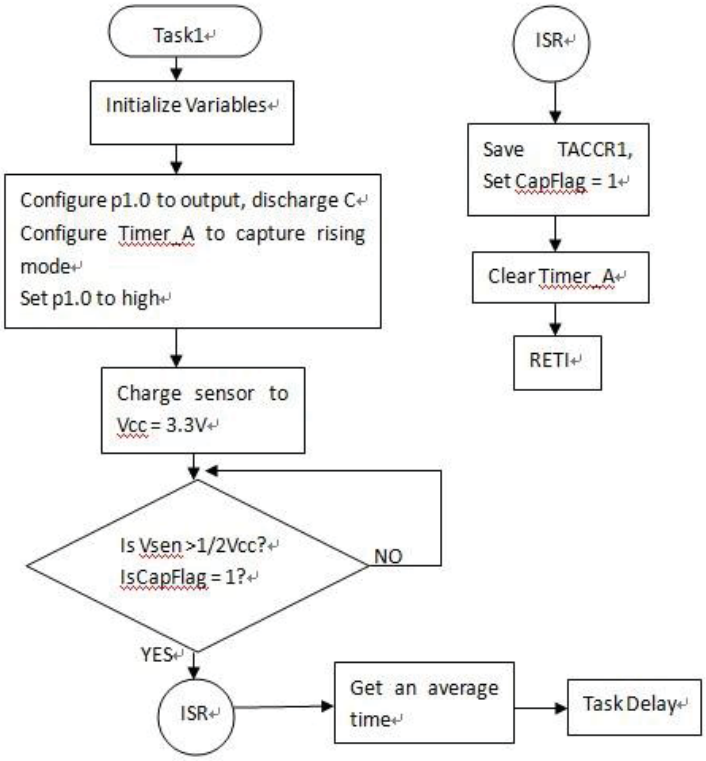

Figure 9.3 Task1 flowchart

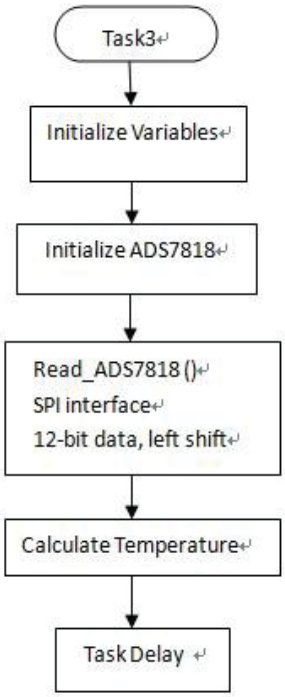

Figure 9.4 Task3 flowchart

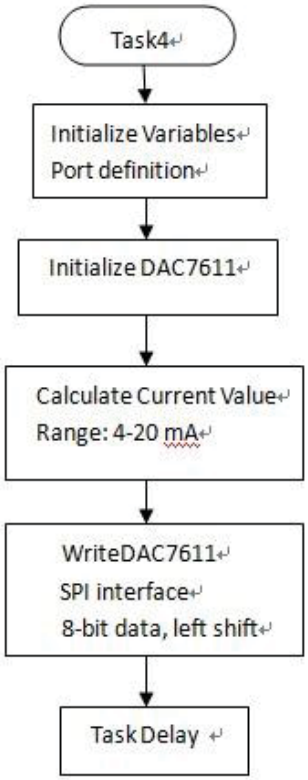

Figure 9.5 Task4 flowchart

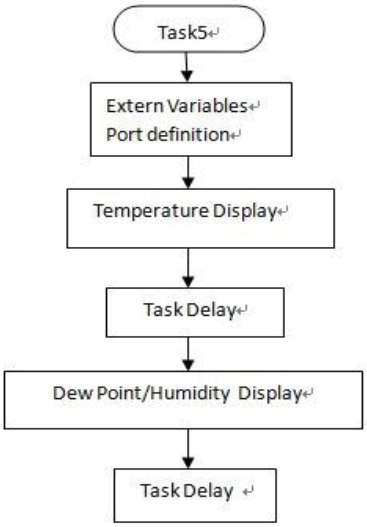

Figure 9.6 Task 5 flowchart

Each task is an infinite loop. At the end of each task, task needs to call a delay API, which puts the calling task into the blocked state for a fixed period. That's why all tasks with different priorities have opportunity to run.

9.2 Hardware Prototype

The CadSoft EAGLE PCB design software is used for schematic and PCB design. EAGLE represents for easily applicable graphical layout editor. A freeware edition of the software is used in this project. This version has some limitation: the useable board area is 100 * 80 mm (4 *3.2 inches), only two signal layers can be used (top and bottom), the schematic can only be one sheet, only for non-profit applications or evaluation purposes. The software can be updated to standard or professional version if the application exceeds the non-profit domain.

9.3 Experimental Results

Currently, there are two humidity sensors under test in the lab. Those two samples are

made at the same time using same technology. They have been tested under Agilent 4284A precision LCR meter for more than two years. Sample IV is a reference sensor, and the sample V is used in the transmitter prototype. Another dew point sensor FA410-H2 from CS Instruments is used as the calibration sensor for the prototype. The output of FA410-H2 is 4-20mA current. A better calibration sensor is highly needed in the future, especially for the low dew point calibration.

Ultra high purity grade nitrogen compressed gas and water are controlled by flow meter to create an adjustable environment for transmitter test. Below is a table for 8 different points which covers from low dew point to high dew point test.

Table 9.1: Different Dew Point Test Points

N_2(lit/min)	H_2O(lit/min)	FA410(mA)	DP(°C)	Time
0	~800	19.7	18.2	30(min)
~750	~600	18.23	8.9	30(min)
~1100	~300	16.80	0	30(min)
~1100	~200	15.68	-7	1(hr)
~1100	~40	13.28	-22	1(hr)
~1100	~10	11.07	-34	1(hr)
~1100	~5	9.0	-50	1(hr)
~1100	~0	Dry(exceed range)	Dry(exceed range)	1(hr)

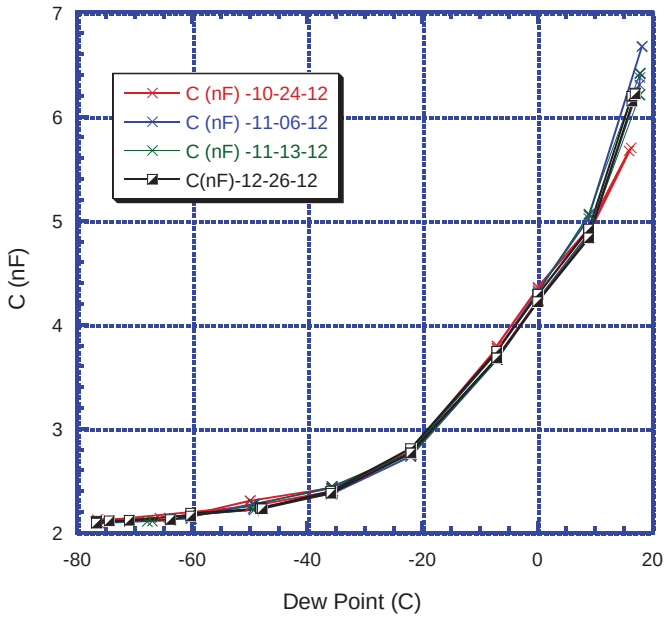

Figure 9.7 Humidity sensor IV (from -80°C to 20°C) by Agilent LCR Meter

Figure 9.7 shows the capacitance-dew point characteristic of humidity sensor IV. This plot has 4 different groups of test data. Each group starts at high dew point to low dew point, and stays overnight at low dew point to reach the lowest dew point. For the next day, each sensor starts at lowest dew point to highest dew point. This is an entire cycle for one measurement. The DP value is calibrated by the FA410.

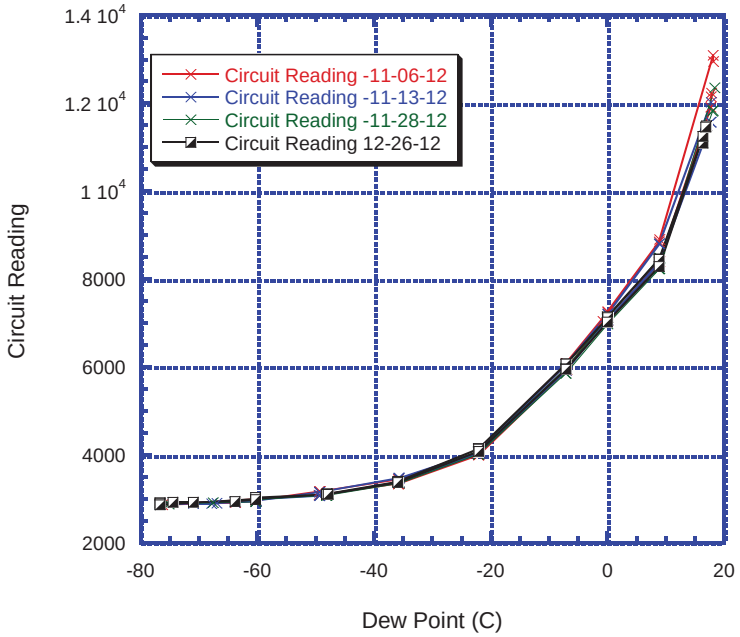

Figure 9.8 Humidity sensor V (from -80°C to 20°C) by Circuit Board

Figure 9.8 shows the capacitance vs. dew point characteristic of humidity sensor V. This plot also has 4 different groups of test data. It has the same test procedure as the sensor IV. The Y axis represents the time captured by the Timer_A inside the MSP430. The circuit is tested under room temperature 23°C, and the resistor in series with the sensor is 125KΩ. A different resistor will have a different capacitance vs. dew point plot, because of the time constant RC. The sensor has temperature coefficient, which is not considered here now. Further research concerns temperature coefficient will be performed in the future. From Figure 9.8, it can be found that stable, drift free, highly sensitive, and low dew point measurement are the main characteristics of this sensor.

Chapter 10 Conclusion and Future Work

10.1 Conclusion

The dew point meter developed in this thesis is based on an alpha-alumina-based humidity sensor, which is a reliable and drift-free humidity sensor. The sensor has shown good performance for long term measurement. This thesis develops a microprocessor based circuit to interface with the sensor. The analog sensor conditioning circuit successfully converts the sensor capacitance into digital data. An economic measurement technology slope A/D conversion is implemented in this conditioning circuit. MSP430F249 plays an important role in the whole system, which is the key part of the slope A/D convertor and the entire circuit. MSP430 can add more new features to the system in the future.

Another two modules of the sensor meter are 4-20mA current loop and temperature sensor. The current loop enables the meter to have a standard output current signal, which represents the dew point. It provides current input for other system. The temperature sensor actually is a common diode. Due to the reliable and linear temperature coefficient characteristic of the diode, it can be easily used as a temperature detector. The humidity sensor needs the detector to calibrate temperature influence. The diode also provides temperature measurement for the meter.

10.2 Future Work

The prototype dew point meter has been built in the lab. It is running quite stable for

more than one year. Some new features need to be added to the system. Meanwhile, more temperature controlled measurement should be done to further improve the performance of the sensor meter.

In future, a few buttons are necessary for the transmitter, which act as the human-machine interface. A custom design LCD display is also important for the sensor meter instead of the common 1602LCD.

Reference

[1]*http://en.wikipedia.org/wiki/Humistor*

[2]T. Nitta and S. Hayakawa, "Ceramic humidity sensors", *IEEE Trans, Comp., Hybrids, Manuf. Technod.,* Vol. CHMT-3, no.2, pp237-243, June 1980

[3]Z. Chen and Mao Chang Jin, An Alpha-Alumina Moisture Sensor for Relative and Absolute Humidity Measurement

[4]Definition of dew point*http://en.wikipedia.org/wiki/Dew_point*

[5]Definition of humidity *http://en.wikipedia.org/wiki/Humidity*

[6]Conversion Table for DP vs. PPm vs. Absolute, *http://www.bchp.com.cn/uploadfile/2011/0610/20110610110411816.pdf*

[7]Mashhour M. Bani Amer, *Assistant Professor,*Department of Biomedical Engineering, Faculty of Engineering, Jordan University of Science and Technology: Design of Reliable and Low-cost Capacitance to Voltage Converters

[8]LM2917 datasheet Texas Instruments

[9]*http://en.wikipedia.org/wiki/Voltage_divider* Capacitance voltage divider

[10]Microchip Application Note AN866

[11]Microchip Application Note AN895

[12]*www.cirvirlab.com/simulation/op_amp_relaxation%20oscillator_online.php*

[13]Microchip Application Note AN1014

[14]AN2041, Understanding of Switched Capacitor Analog Blocks

[15]Application Report, SLAA129B – JANUARY 2006 Implementing An Ultralow-Power Thermostat With Slope A/D Conversion

[16]Application Report SLAA104 – November 2000, An MSP430F11x1 Sigma-Delta Type Millivolt Meter

[17]Application Report, SLAA071 - October 1999 Economic Measurement Techniques With the Comparator_A Module

[18]*http://en.wikipedia.org/wiki/Current_loop* , 4ma-20ma current loop

[19]BAPI Understanding 4-20ma current loops, Application Note rev.10/05/06

[20]BAPI 4-20ma current loop configurations, Application Note REV.10/05/06

[21]Texas Instruments BURR-BROWN XTR115 datasheet

[22]Texas Instruments BURR-BROWN DAC7611 datasheet

[23]*www.wwdmag.com/water/seven-basic-types-temperature-sensors*
The seven basic Types of Temperature Sensors

[24]DS18B20, Maxim Integrated PDF datasheet

[25]Principles of SEMICONDUCTOR DEVICE by SIMA DIMITRIJEV

[26]Texas Instruments ADS7818 datasheet

[27]Study of an operating system: FreeRTOS, Nicolas Melot, Operating system for embedded devices

[28]USING THE FREERTOS REAL TIME KERNEL

[29]FreeRTOS Manual

[30]http://www.westmorelandengineering.com/, Westmoreland Engineering

Vita

Bojie Chen was born in Hangzhou, Zhejiang Province, P. R. China.

Education

August, 2011 --- Present

M.S. Student

Department of Electrical and Computer Engineering, University of Kentucky,

September, 2005 --- July, 2009

Bachelor of Science

Department of Electrical and Information Engineering

Zhejiang University of Technology, P. R. China