AT89S52 BASED EXHAUST FAN CONTROLLER BASED ON SENSOR READING USING PHASE FIRING ANGLE METHOD

A final project report
presented to
the Faculty of Engineering

By

Bagoes Dwijadmika
00220090002

in partial fulfillment
of the requirements of the degree
Bachelor of Science in Electrical Engineering

President University
March 2013

DECLARATION OF ORIGINALITY

I declare that this final project report, entitled "AT89S52 Based Exhaust Fan Controller Based on Sensor Reading Using Phase Firing Angle Method" is my own original piece of work and, to the best of my knowledge and belief, has not been submitted, either in whole or in part, to another university to obtain a degree. All sources that are quoted or referred to are truly declared.

Cikarang, Indonesia, March 2013

Bagoes Dwijadmika

AT89S52 BASED EXHAUST FAN CONTROLLER BASED ON SENSOR READING USING PHASE FIRING ANGLE METHOD

By

Bagoes Dwijadmika

00220090002

Approved by

Antonius Suhartomo, Ph.D
Final Project Supervisor

Dr.-Ing Erwin Sitompul, M.Sc
Head of Study Program
Electrical Engineering

Dr.-Ing Erwin Sitompul, M.Sc
Dean of Faculty of Engineering

ACKNOWLEDGEMENT

First of all, I would like to thank both of my parent for their support and love. Their motivation inspires me to do this final project as good as posible, and gives me spirit to study in electrical engineering department. I also thank President University for giving me a chance to enroll and study in this great university.

I also would like to thank mister Antonius Suhartomo, Ph.D for being my final project adviser. He guides me and helps me solve problems that I had during the making of the device for this final project.

Besides them, I would like to thank DiverVenturE as my small family; I love you all, guys. Also electrical engineering colleagues for exchanging knowledge, experience, and support that also emotionally inspires and motivates me to make this final project paper happen. I also would like to thank my D3 dorm mates for their motivation and support.

Last but not the least, I personally want to thank pak Totok Budioko, ST that guides me to solve problems regarding to microcontroller hardware and software. It is nice to know you.

Cikarang, March 2013,

Bagoes Dwijadmika

ABSTRACT

Exhaust fan as one of the common electrical appliance that widely used for home and industry, is mostly still operated manually. However, this condition can be modified into automatic operated by adding processor unit and also several supporting electrical components. Automatic in this project means exhaust fan can operate in different levels of power, based on sensor TGS 2600 reading. However, the range of sensor reading is decided manualy. In order to control and to process, microcontroller AT89S52 is utilized both in hardware and software to be able to control the fan, as the load. To limit amount of electric source supplied to the load, triac is used as high speed switching device. Zero crossing detector will be required in AC volage controller to obtain ideal range of triac firing, by pulsing each time AC wave form hit zero point. Since the load is AC sourced, the suitable theory that used is phase control using triac, or also called phas firing angle. The result of this project is obtained by comparing collected data with calculation; however, there is deviation between calculated result practical result. On the other hand, the load is responding very well toward sensor reading. By controlling its voltage, it means power consumed also reduces which directly contributes to energy saving and efficiency.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATION

CHAPTER 1
INTRODUCTION

1.1 Final Project Background

Exhaust fan is an electrical device which has ability to transfer smoke, odor, steam, or unwanted air from inside of a room to outside, or vice versa [1]. It is commonly used whether in home usage or industry, such as in the bathroom, kitchen, garage, warehouse, a room with plants inside, or any room that require air quality control. The one that extract air from inside often called outtake fan, while the intake one is the opposite. The ability to extract and/or pull air to inside makes exhaust fan is the appliance that used for circulating air between inside and outside of the room to maintain certain condition of air in a room.

Figure 1.1 Conventional exhaust fan

Until these days, conventional exhaust fan still operated manually by switch. Based on its function and ability, author eager to modify current exhaust fan to be controllable automaticaly by implementing the principal of phase firing angle.

Phased firing angle itself is often used to control voltage that sourced from AC main. It works like PWM (Pulse Width Mudulation); which in PWM, the pulse is fired in certain time to get desired avearage voltage. The different between PWM and phase firing angle is PWM is not required to syncronize between time and giving pulse, to turn on load. It happens because in AC main, the waveform is sinusoidal and always cross zero at certain point, while in DC it is not, because the wave form is flat line and will not cross zero. Phase firing angle works by firing triac in ideal range, whereas the range is between wave form's zero (from 0 to 180 degree). Hence, phase firing angle is suitable to control AC sourced load.

1.2 Problem Statement

Exhaust fan still become most household appliance that used whether in house, office, or factory. Until these day, exhaust fan still operated manualy. Sometimes it is not necessary to extract whole air in the room to just get rid of small amount of contaminant that pollutes a room, except if the room is full of contaminant. To be able to work automaticaly, addition of electrical components such as microcontroller, sensor, and switching device are required. Reliable principal to limit voltage supply also need to be applied; since the load is AC sourced, then phase firing angle is suitable. Based on condition stated, the following problem statement has been used for guiding investigation:

How to modify exhaust fan to works conditionally based on sensor reading by applying phase firing angle.

1.3 Objective of the Project

The purpose of this project is to modify current manual device to operate automatically in different levels of power; as the load is operated based on AC source, author finds it is suitable to use principal of phase firing angle. This project is an innovation for current device, to turn manual operated device into automatic operated device.

Exhaust fan is chosen because until these day, it still operates manualy. For this project, the objective is to modify current exhaust fan to work automaticaly and conditionaly by implementing phase firing angle.

1.4 Scope and Limitation

In the making of this project, there are several limitations and scope that have to be applied.

Here are the scope and limitations for this project:

Scope:

The project is built in scale of laboratorium experiment.

Limitation:

- The contaminant that used is gas from lighter.
- The crystal oscillator used for microcontroller is 11.0592 MHz
- Source code for this project is compiled using Keil C51.
- The output displayed on LCD is ADC value.
- Delays used before firing triac are 3ms, 4ms, and 5ms.[1]

1.5 Final Project Outline

The final project report consists of five chapters and is outlined as follows:

Chapter 1: Introduction. This chapter consists of problem background, Final Project statement, Final Project objective, Final Project scope and limitation, and Final Project outline.

Chapter 2: Literature Study. This chapter describes about the component that will be used in this final project. The description includes characteristic, work mechanism, etc. The components consist of microcontroller AT89S52, smoke sensor, opto electronics, triac, and adc. Theory that will be used in this final project also will be explained in this chapter.

Chapter 3: Design and Implementation. This chapter delivers conceptual design and real implementation for software and hardware. This chapter also will explain for circuitry design and programming.

[1] Those delays were chosen to give distinct difference of supplied voltage.

Chapter 4: Project Result and Analysis. This chapter consists of the analysis of the hardware and software. Simulation results are examined to finally conclude the strengths and weakness of the proposed system according to objectives in previous chapter.

Chapter 5: Conclusions and Recommendations. This chapter consists of conclusions obtained throughout this project and recommendations for future projects.

CHAPTER 2
LITERATURE STUDY

2.1 Preliminary Remarks

This chapter elaborates main idea and knowledge that required in the research and writing of this final year project. All of the knowledge regarding to this project will be explain in this chapter with objective to help reader understand the concept of this project and its supporting elements. The objective of this project is to control the speed of an exhaust fan based on smoke sensor reading, using simple implementation of phase firing angle theory and all of the information related to it.

The main hardware that required for this project are microcontroller Atmel AT89S52, LCD 16X2 LMB162ABC, analog to digital converter ic 0809,and Figaro smoke sensor TGS 2600 . The reason behind components choosing is based on availability, ease of interfacing, and low cost price.

Section 2.2 will explains about triac, section 2.3 will mainly explains about microcontroller and its features, section 2.4 is about adc 0809, section 2.5 explains about LCD, while section 2.6 explains briefly about Figaro smoke sensor, and final section; 2.7 will explains zero crossing and its relation to phase firing angle in brief.

2.2 Triac Component

Triac is a bidirectional device consists of 2 scr (silicone controlled rectifier) that connects in parallel way with gate terminal; Triac pins consist of A1, A2, and G. A1 and A2 will conducted if it given a triggering signal such as pulse or AC current.

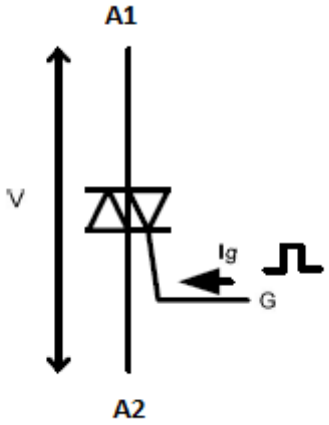

Figure 2.2 Triac layout in symbol

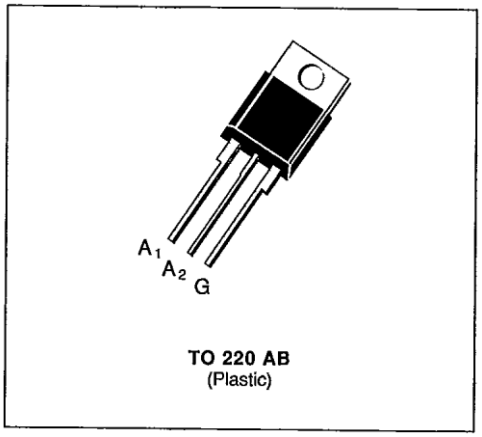

Figure 2.1 Triac layout in hardware

Bidirectional means that it terminals/pins (A1 and A2) cannot determine neither as cathode nor anode. If A2 is positive towards A1, triac can be turned off by giving positive signal to G between A1. Once gate is triggered, current will flow through A1 and A2. Triac will automatically switched off when ac sine wave crossing at zero point, hence there is no current flows through A1 and A2.

There many types, models, and brands for triac those make us wonder which types of triac suitable for a project. Thus there are several key conditions that should be considered to choose a triac such as [9]:

1. V_{drm} is the maximum voltage that allowed to flow through triac. To be safe, the V_{drm} should above the source's voltage, in this case main's maximum voltage is around 311 Volt.
2. To make G (gate) pin conducts, gate should be given current.
3. $I_{T(RMS)}$ should be higher than current that given from load.

Based on its datasheet, here is maximum value required to run triac:

Table 2.1 Triac main specification

Symbol	**Parameter**	**Typical**	**Maximum**
V_{DRM}	Repetitive peak off-state voltage	-	600 V
$I_{T(RMS)}$	RMS on-state current	-	16A
I_{GT}	Gate trigger current	2.5 mA	10 mA

2.3 Microcontroller Atmel AT89S52

AT89S52 is a microcontroller from 8051 family. 8051 itself is a family of Intel's micro controller MCS-051 which is very old and popular on its time, and also becoming a standard for today's microcontroller [4]. 8051 series is very universal and until now there is so much micro controller that originally based on 8051 and manufactured under different name.

AT89S52 is a microcontroller manufactured by Atmel using high non-volatile memory technology and compatible with the industry standard of 8051C instructions and pin out [5]. The 8K bytes programmable flash memory makes the program memory able to be reprogrammed.

The packaging of AT89S52 that used in this project is 40-lead PDIP as shown in figure below.

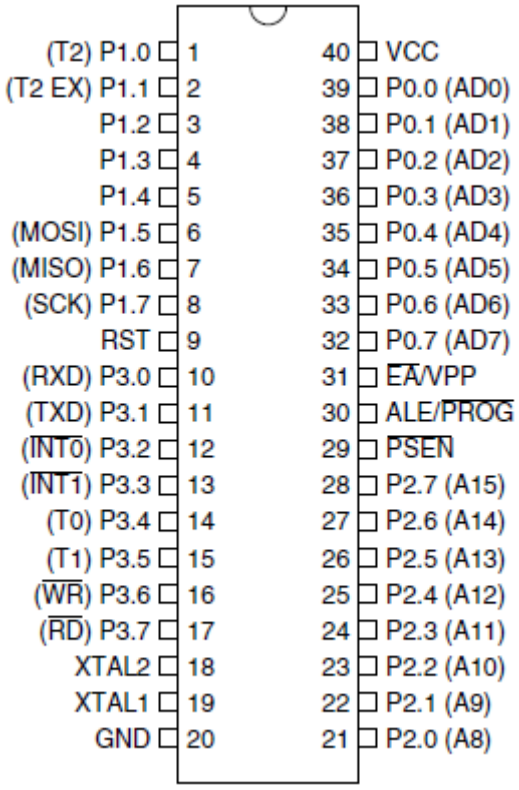

Figure 2.3 AT89S52 pin layout

Port 0.0-0.7, 1.0-1.7, 2.0-2.7, and 3.0-3.7 can be used for input and input purpose. The initialization of the port itself can be done using program (software.).

2.3.1 Atmel AT89S52 features

The different between microcontrollers of 8051 are not significant. Most of them have same pins layout and functions. Thus AT89S52 have different features from its predecessor such as [5]:

• Compatible with MCS®-51 Products

• 8K Bytes of In-System Programmable (ISP) Flash Memory

– Endurance: 10,000 Write/Erase Cycles

• 4.0V to 5.5V Operating Range

• Fully Static Operation: 0 Hz to 33 MHz

• 256 x 8-bit Internal RAM

• 32 Programmable I/O Lines

• Three 16-bit Timer/Counters

• Eight Interrupt Sources

• Low-power Idle and Power-down Modes

• Interrupt Recovery from Power-down Mode

• Fast Programming Time

• Flexible ISP Programming (Byte and Page Mode)

• Green (Pb/Halide-free) Packaging Option

With plenty of input/output pins, competitive price, and bunch of features, AT89S52 is more than enough for entry level project. Programming for 8051 can be compiled through wide variety of language compilers, such as C, assembly, and basic.

2.3.2 Port Configuration

- Port 0 : Port 0 is an input/output bidirectional port. When port 0 given '1' (high), it can be used for high impedance input. It has internal pull-ups but weaker than other ports, and it can be overcome by adding external pull-ups [7].
- Port 1 : Port 1 is a bidirectional input/output that has internal pull-up. When logic '1' given to port 1, each pin on port 1 will be pulled up by internal pull-up, hence it can be used as input. If the pins connected to ground (example: led ground) it will gives current (output) [7].
- Port 2 : Port 2 is another bidirectional output/input that will sink current around 1.6mA if it been given logic 1 and will source current if given logic 0; the same as other port [7].
- Port 3 : Port 3 is just the same as the other ports. It is a bidirectional output/input port that have internal pull-up and also have ability to sink or source current.

All of the port seems very similar each other yet several of the pins held special function that will be showed in Table 2.2.

Table 2.2 AT89S52 Pin Alternate Function

Port Pin	**Alternate Function**
P1.0	T2 (external count input to timer/counter 2), clock out.
P1.1	T2EX (timer/counter 2 capture/reload trigger and direction control)
P1.5	MOSI (used for in-system programming)
P1.6	MISO (used for in-system Programming)
P1.7	SCK (used for in-system programming)
P3.0	RXD (serial input port)
P3.1	TXD (serial input port)
P3.2	INT0 (external interrupt 0)
P3.3	INT1 (external interrupt 1)

P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	WR (external data memory write strobe)
P3.7	RD (external data memory read strobe)

2.3.3 Memory Organization

There are 2 types of memory in 8051 family. They are program memory and data memory. The program memory is read only while data memory is read and write-able. In case the memory (RAM or ROM) inside the 8051 is not sufficient, there is possibility it can be added with 2 external memory chips with capacity 64Kb each. For the addressing it can use port 2 and 3 [4].

Program memory is a space memory that used to store code program and constants. This memory is read only; means when the program in this memory is executed, the program cannot be changed, While data memory is an internal ram (on chip) [8].

2.3.4 Timer

Counter in 8051 can acts as a counter or a timer. When in counter mode, the frequency of the crystal oscillator is varied, thus every pulse signal is counted until it reach desired amount then overflow. When in timer mode, the frequency of crystal oscillator is fixed [7].

AT89S52 have 3 timers that can be used as timer or counter. These timers are timer 0, timer 1, and timer 2. Each of these timers work independently, means we can use timer 0 as a counter, timer 1 and 2 as a timer at the same time, on the same process.

Timer has 2 register type, timer mode register (TMOD) and timer control register (TCON). TMOD is an operational mode of T0 and T1. TMOD consist of 8 bits with configuration the low 4 bits is configuration for T0, while the rest is for T1 [4].

Table 2.3 TMOD Bit Mapping

	0	0	0	0	0	0	0	0	Value after reset
TMOD	GATE1	C/T1	T1M1	T1M0	GATE0	C/T0	T0M1	T0M0	Bit name
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

Not like TMOD, TCON configuration is only using 4 bits while the rest used for interrupt [4].

Table 2.4 TCON Bit Mapping

	0	0	0	0	0	0	0	0	Value after Reset
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	Bit name
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

Timer/counter have 4 modes which are [9]:

1. Mode 0; the mode when timer/counter is 13 bit (counting ability up to 8192).
2. Mode 1; the mode when timer/counter is 16 bit (counting ability up to 65536).
3. Mode 2; the mode when timer/counter is 8 bit, also called auto reload.
4. Mode 3; the split counter mode.

Referring to figure of TMOD diagram above:

1. To activate mode 0 in timer/counter 0 (works also for timer/counter 1) we should assign both M0 and M1 to logic 0.
2. To activate mode 1 in timer/counter 0 (works also for timer/counter 1) we should assign M0 to logic 1 while M1 to logic 0.
3. To activate mode 2 in timer/counter 0 (also works for timer/counter 1) we should assign both M0 and M1 to logic 1.

In this project, the timer register that used is TMOD register, to generates interrupt timer.

2.3.5 Interrupt

Interrupt is an input/output handling technique that works by input/output trigger a signal that can recognized by microcontroller (this signal often called interrupt signal). Later on, every time the microcontroller reads the interrupt signal when microcontroller is executing an instruction, the microcontroller will stop current process and alter to interrupt's routine. Interrupt routine is a process that will be executed every time interrupt signal occurs. After interrupt's routine is finish, it continuous the main process.

Registers of interrupt enable (IE) is shown in this diagram below [8]:

Table 2.5 Interrupt Bit Mapping

BIT	7	6	5	4	3	2	1	0
REGISTER	EA	N/A	ET2	ES	ET1	EX1	ET0	EX0

The explanation of this registers are elaborated in this table [8]:

Table 2.6 Interrupt's Bit Function

BIT #	**NAME**	**FUNCTION**
7	EA	Enable all. If EA=1, it enable all interrupts; EA=0 disabled all interrupts
6	N/A	Not used
5	ET2	Enable interrupt Timer 2. ET2=1 interrupt timer 2 enables; if ET2=0 it is disabled
4	ES	Enable interrupt serial. If ES=1 it enables interrupt serial; if ES=0 it is disabled.
3	ET1	Enable interrupt Timer 1. ET1=1 interrupt timer 1 enables; if ET1=0 it is disabled.
2	EX1	Enable interrupt external 1. If EX1=1 it enables external interrupt 1; if it is EX1=0 then it is disabled.
1	ET0	Enable interrupt Timer 0. ET0=1 interrupt timer 1 enables; if ET0=0 it is disabled.
0	EX0	Enable interrupt external 0. If EX0=1 it enables external interrupt 0; if it is EX0=0 then it is disabled.

2.3.6 Crystal Oscillation

AT89S52 needs external oscillator (named crystal quartz) to execute every instruction given. This microcontroller Consist of 12 oscillations period. In this project, the external crystal oscillator used is 11.0592 MHz. The frequency can be acquired by this calculation:

$$\frac{Crystal\ value\ (Mhz)}{12} = Frequency(Mhz) \quad \textbf{(2.1)}$$

$$\frac{1}{Frequency(Mhz)} = Time\ needed\ per\ machine\ cycle(uS) \quad \textbf{(2.2)}$$

Referring to the formula, then with 11.0592 MHz crystal oscillator we will have frequency of 0.9216 MHz, and time required per machine cycle is 1.085 uS. This calculation is very useful when timer function is used.

2.3.7 TMOD timer value

AT89S52 timer will act as timer when pin XTAL1 and XTAL2 is connected to quartz crystal. This quartz crystal will determine any value needed for delay and time process, since the time required to complete a machine cycle is varying depends on microcontroller's crystal value.

For the lower bit and higher bit needed to be loaded in TL and TH register, this calculation is needed when using timer mode 1(16bit) [16]:

$$\mathrm{YYXX}_{\mathrm{hexadecimal}} = 65536_{\mathrm{decimal}} - \left(\mathrm{delay} \div \frac{12_{\mathrm{decimal}}}{\mathrm{XTAL\ value\ in\ Hz}}\right) \quad \textbf{(2.3)}$$

Note that the result is in decimal; hence it should change to hexadecimal before loaded to TH (YY) and TL (XX) registers.

2.4 ADC 0809

ADC 0809 is a data acquisition component with 8 bit analog to digital converter, 8 channels multiplexer and microprocessor compatibility control logic [6]. Below is the block diagram of ADC 0809.

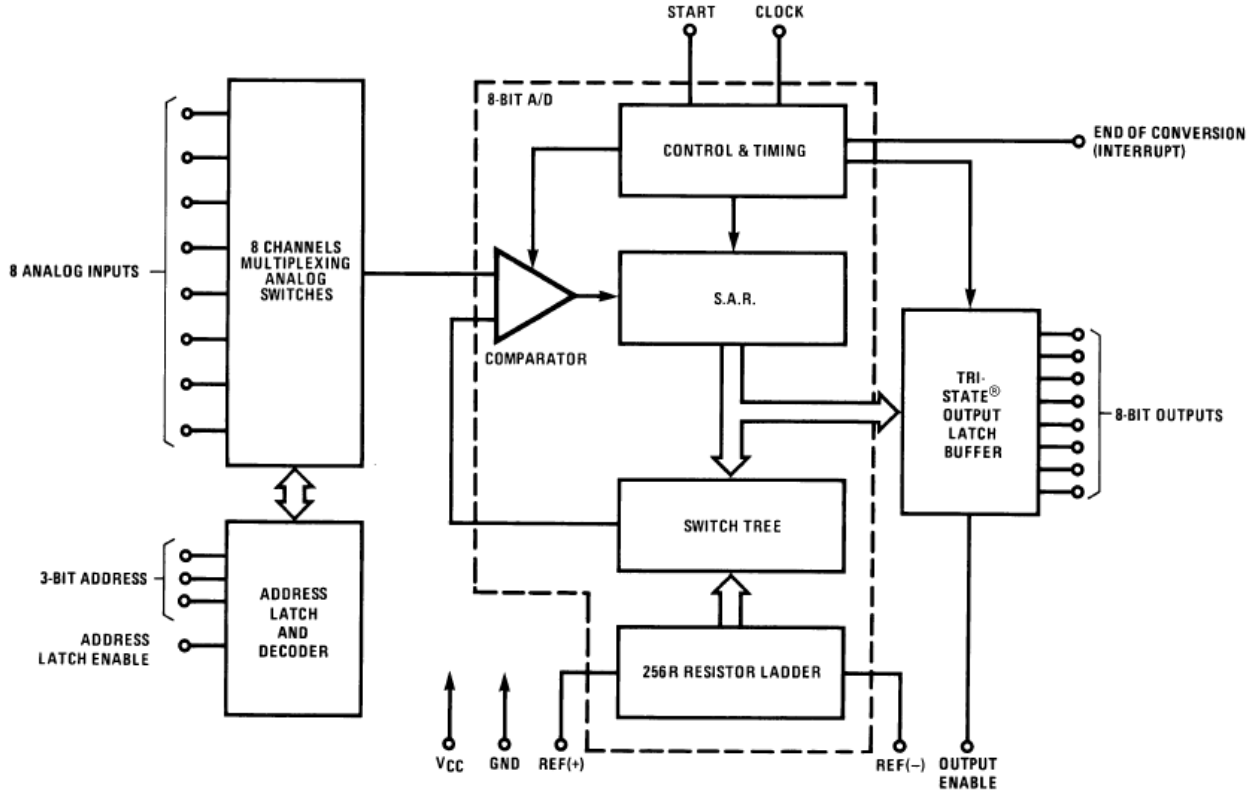

Figure 2.4 ADC 0809 block diagram

ADC stands for analog to digital conversion. In this project, ADC is used for converting analog signal given by analog sensor into digital sensor; hence the signal can be processed by microcontroller and displays it to LCD in unit of bit. Since there is 8 bit resolution, the range of the number that displayed will be between 0-255 and keep updating continuously as the characteristic of ADC 0809.

The digital data converted is proportional to analog input. Conversion cannot be done perfectly because analog voltage changes continuously while digital is step. Each binary from 0-255 represent certain value of analog voltage. The higher the bit, the longer the range; which means digital value is close to true analog voltage value.

2.4.1 ADC pin configuration

ADC 0809 has total 28 pins as pictured below:

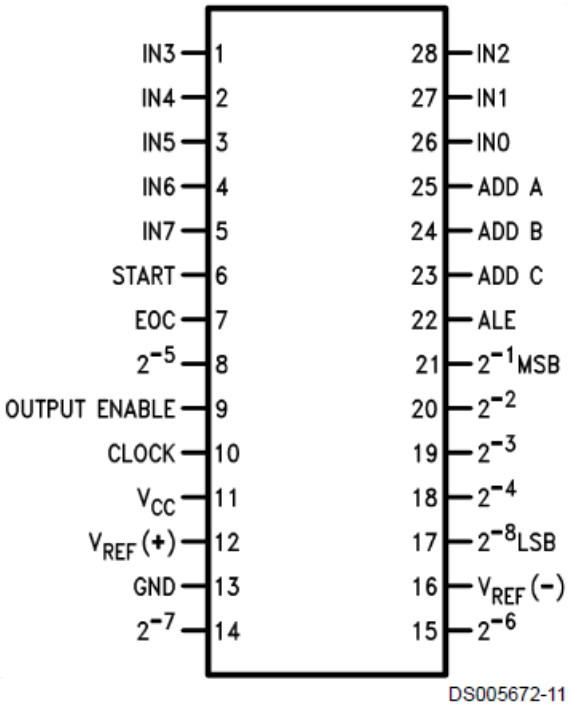

Figure 2.5 ADC 0809 physical pin layout

The list of pin function will be explained in table below [18]:

Table 2.7 ADC 0809 Pin Description

Pin #	Label	Description
1 : : : 5	IN3 : : : IN5	Analog data input.
6	Start	Instruction to start converting.
7	EOC	Stands for end of conversion. This signal given from ADC
8	2^{-5}	Bit of digital converted input.

9	OE	Output enables.
10	Clock	Clock signal given by mcu or other external source. i.e 555 timer.
11	Vcc	Power for IC, ranged from 4.5V-6V.
12	Vref+	Voltage reference.
13	GND	Ground pin.
14 : 15	2^{-7} : 2^{-6}	Bit of digital converted input.
16	Vref-	Voltage reference.
17	2^{-8}	Bit of digital converted input.
18 : : 21	2^{-4} : : 2^{-1}	Bit of digital converted input.
22	ALE	Pulse that sent by mcu when address ready to be loaded to ADC.
23	ADDC	Address for multiplexer.
24	ADDB	
25	ADDA	
26 : 28	IN0 : IN2	Analog data input.

Algorithm of the ADC is:

- Select the channel.
- Enable ALE(address latch enable).
- Give logic high to start.
- Set ALE logic to low.
- Set start conversion to low.
- Waiting until conversion is done.
- Set output enable pin to high, to read data of ADC output.
- Stop retrieving data from ADC.

All of the steps above are done using program/software.

2.5 LCD 16x2

LCD 16x2 means it has 16 characters length and 2 rows. The LCD used in this project is LMB 162ABC manufactured by TOPWAY, and this LCD supports HD44780 controllers. HD44780 standard requires 3 control lines, which are RS, RW, and E. RS is high when it requires to send text data that should be displayed on LCD, e.g printing a word; it is low for commanding special function, such as clear screen, move cursor, and etc. E line is high, which tells it to receive data; first it sets to low, then decide the value of RW and RS, after that bring E high logic to tells it that data is sent, then give low logic again to E.

Here is the block diagram [11]:

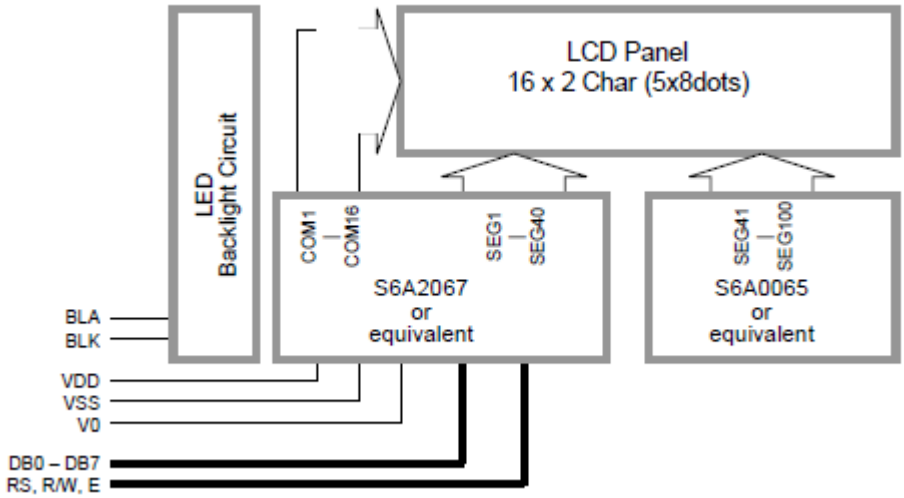

Figure 2.5 LCD block diagram

This LCD consist of 16 pins which are [11]:

Table 2.8 LCD Pin Description

Pin #	Pin name	I/O	Description
1	VSS	Power	Ground
2	VDD	Power	Positive power supply
3	V0	Power	LCD contrast reference supply
4	RS	Input	Register select
5	R/W	Input	Read/write control bus
6	E	Input	Data enable
7	DB0	I/O	Bi-directional tri-state data bus
:	:		
14	DB7		
15	BLA	Power	Backlight positive supply
16	BLK	Power	Backlight negative supply

There are also commonly used commands and instructions for LCD, such as [12]:

Table 2.9 LCD Instructions

Number	Instruction	Hex
1	Function Set: 8-bit, 1 Line, 5x7 Dots	0x30
2	Function Set: 8-bit, 2 Line, 5x7 Dots	0x38
3	Function Set: 4-bit, 1 Line, 5x7 Dots	0x20
4	Function Set: 4-bit, 2 Line, 5x7 Dots	0x28
5	Entry mode	0x06

6	Display off cursor off	0x08
7	Display on cursor on	0x0E
8	Display on cursor off	0x0C
9	Display on cursor blinking	0x0F
10	Shift entire display left	0x18
11	Shift entire display right	0x1C
12	Move cursor left by 1 character	0x10
13	Move cursor right by 1 character	0x14
14	Clear display, also DDRAM content	0x01
15	Set DDRAM address or cursor position on display	0x80+add
16	Set CGRAM address or set pointer to CGRAM location	0x40+add

2.6 Smoke Sensor Figaro TGS 2600

TGS 2600 is a smoke sensor manufactured by Figaro that can detect various kinds of gases such as carbon monoxide, methane, iso-buthane, and hydrogen.the sensor works by comparing the sensor's resistance ratio.

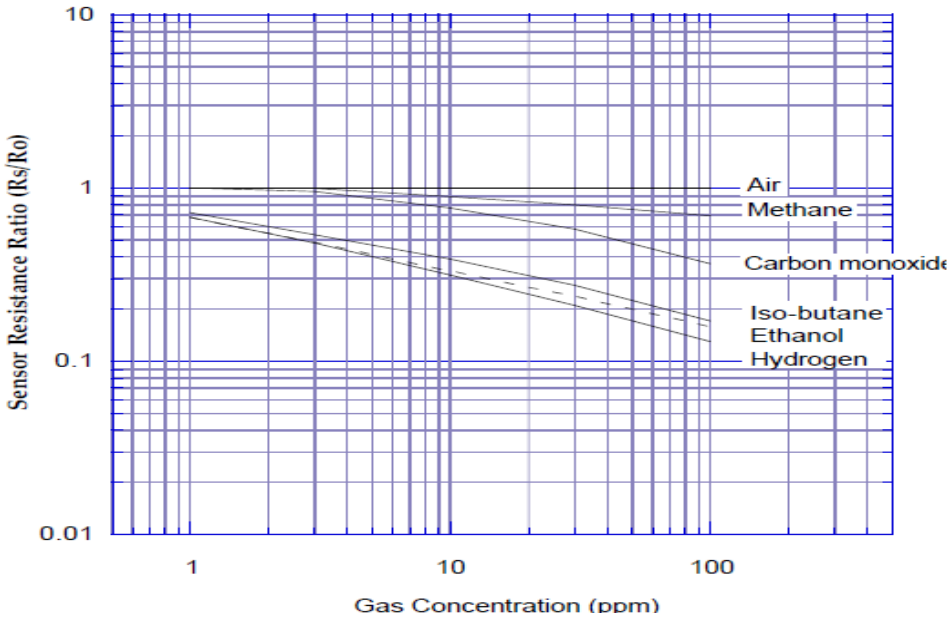

Figure 2.6 Detection characteristic

The sensor has internal heater that works with 5Vdc to clean inner part of sensor from gas contaminant. The sensor itself is installed as figure below shows:

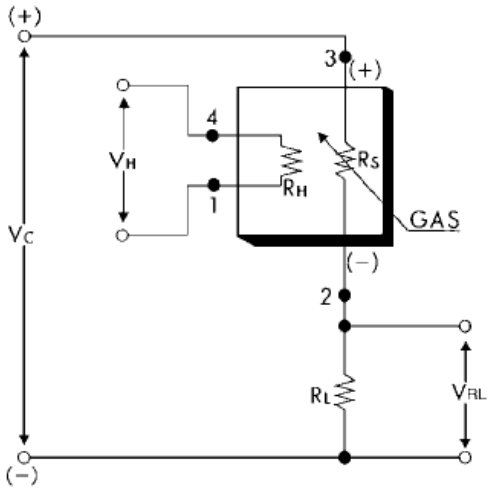

Figure 2.7 Sensor circuitry

V_{RL} is where the measurement given to ADC to convert it into digital signal. This sensor works maximum 15mW, and has wide variation of internal resistance (Rs). R_L is chosen based on sensor's datasheet; R_L is free to choose as long as it is >= 450Ω.

2.7 Phase Firing Angle

Phase firing angle is a method of voltage controller; it is an electronic module based on certain switching devices, such as triac (triode for Alternating Current), scr (silicon controlled rectifier), or igbt (insulated gate bipolar)[2]. The idea of this method is to obtain a variable voltage in output that will be delivered to a load, while the input is a fixed voltage. The example of phase firing angle is showed in Figure 2.1 below, suppose the load is resistive.

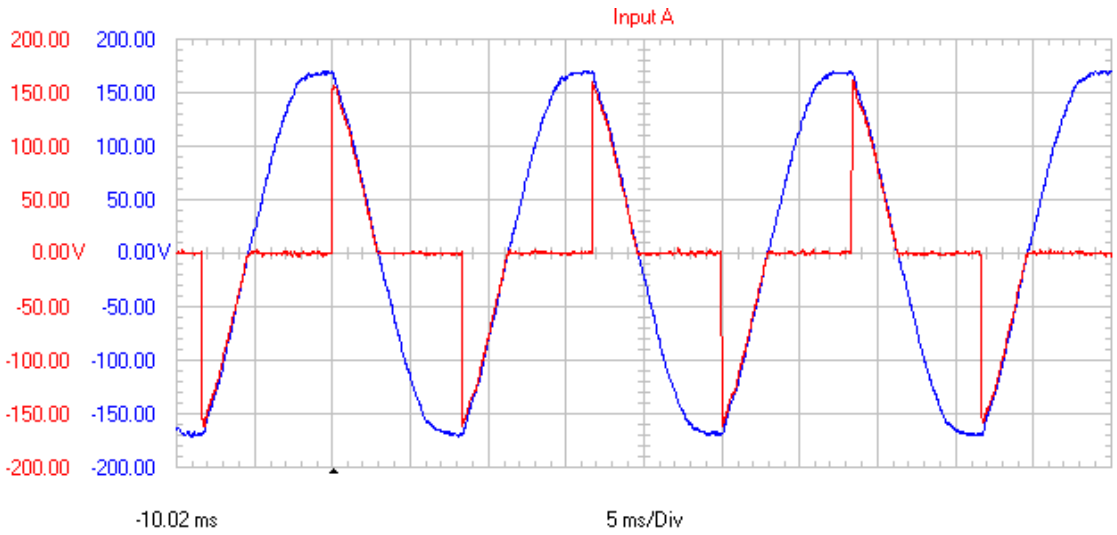

Figure 2.8 Phase firing angle for 40% of main voltage

By its usage, phase firing control often used to control voltage, current, or even power that a power supply supplies it to a load. It seems similar with pulse width modulation, where PWM (pulse width modulation) would pulsing on and off for certain amount of time to get the desired voltage flowing through load [3]. In DC powered output, PWM can be used instead phase firing angle because the time base is not important.

Phase firing angle circuit is requiring zero cross detector and triac circuit. Zero cross is point where line voltage is 0. In AC 50 Hz main, the voltage will be zero each 10 ms. Zero cross detector required to makes triac firing in correct area. The application is triac will be fired after certain delay (predetermined) after zero cross point.

2.7.1 Relation between angle and voltage

What really makes the voltage decreasing is triac firing at certain angle ranged from 0 degree-180 degree. The higher the angle means less voltage delivered to load. However, triac firing is not effective when firing with less than 30 degrees delay angle or more than 150 degrees delay angle, since there will be no significant change of output voltage[17]. This graph is used for knowing maximum voltage delivered at certain millisecond(s) of delay:

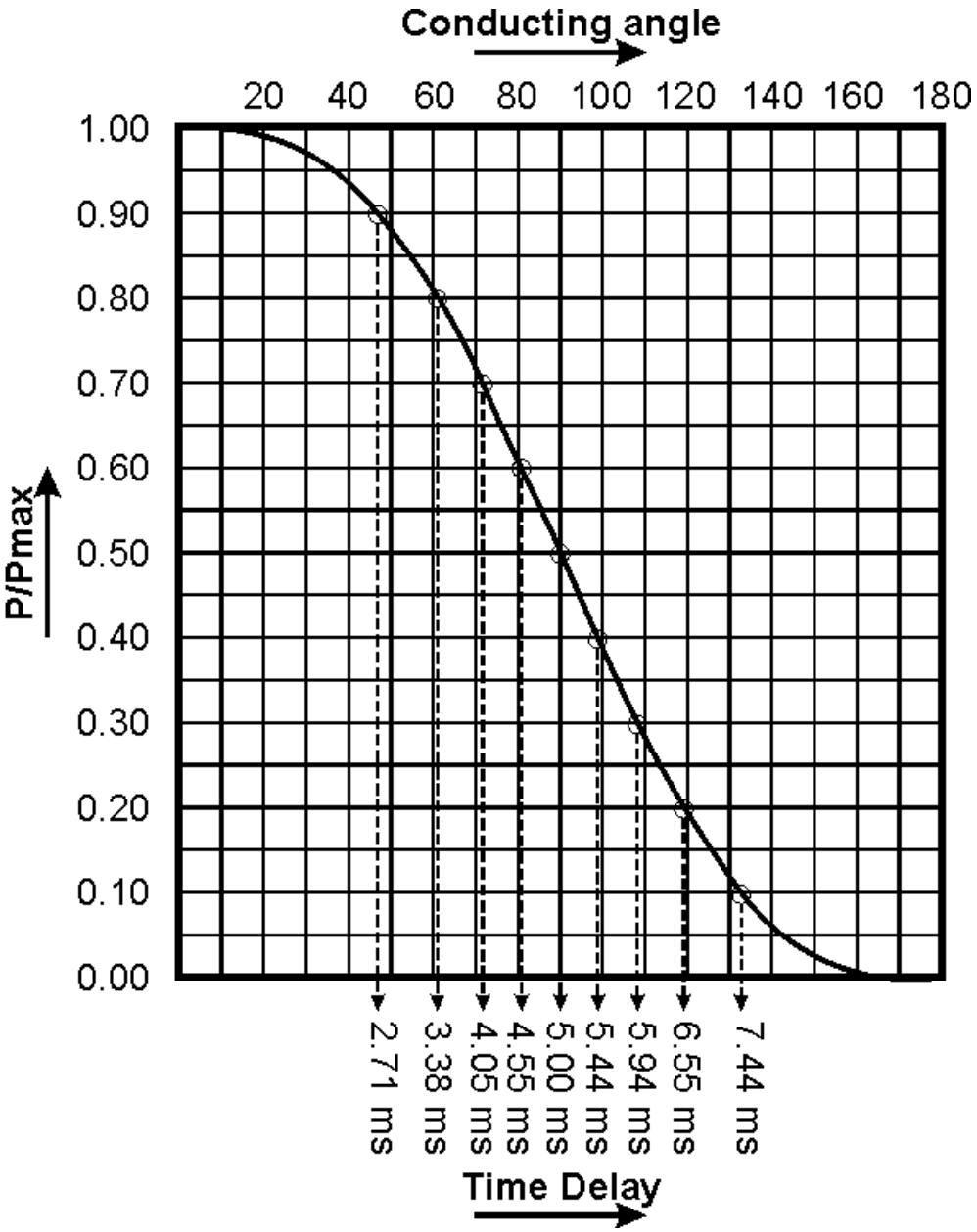

Figure 2.9 Graph of voltage vs delay

To get phase angle (α), this formula can be used [13]:

$\alpha = \Delta t\ X\ f\ X\ 360$ **(2.4)**

Assuming the output wave is pure sinusoidal, voltage rms at certain delay angle can be found by integrating sine wave equation to certain amount of angle:

$$Vout = \sqrt{\frac{1}{\pi}\int_{\alpha}^{\pi} Vpeak^2 \sin^2 \omega t d\omega t}$$

$$Vout = Vpeak\sqrt{\frac{1}{\pi}\int_{\alpha}^{\pi} \frac{1-\cos 2\alpha}{2} d\alpha}$$

$$\therefore Vout = Vpeak\sqrt{\frac{1}{2\pi}\times(\pi - \alpha + \frac{\sin 2\alpha}{2})} \quad \textbf{(2.5)}$$

CHAPTER 3
DESIGN AND IMPLEMENTATION

3.1 Preliminary Remarks

These chapters will mainly talks about implementation and design of the desired project. All of the designs written in this chapter are based on theories and knowledge that been gathered by the author. The designs that will be elaborated in this chapter are circuitries (microcontroller, LCD, power supply, phase firing angle, sensor, and ADC), while the other material required for this chapter is formula of phase firing angle itself.

3.2 Hardware Design

The main hardware consisting of microcontroller unit, sensor, triac,and zero crossing. The overall of the project is pictured in figure below.

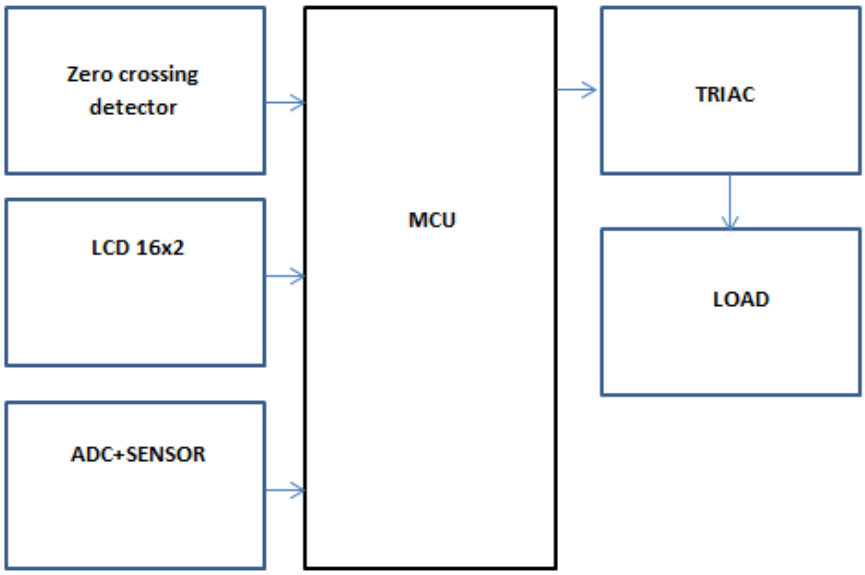

Figure 3.1 Final project block diagram

Microcontroller acts as processor that will compile all the inputs and give output regarding to several perimeters set for this project.

Firstly 5Vdc power supply powers the entire component except in load circuitry, which uses 220Vac. The 5Vdc should have sourced by single reference point because the time between zero crossing and triac firing must be in same range of time line. If different sources are used, the triac firing will not be precise.

ADC is used to convert the analog voltage into digital signal, hence it can be read by microcontroller. Since AT89S52 does not have internal ADC, then external ADC is used to fulfill the purpose of the project. By converting the analog signal becomes digital, LCD can display the sensor measurement through ADC. The value of this ADC later on will be divided into few ranges to determine different fan's speed with respect to sensor reading.

The condition desired for triac firing is written in interrupt's routine. Interrupt's routine is an instruction that will be executed when interrupt occurs. For this project, the interrupt source is opto transistor's output. Every zero crossing output pulses (interrupt occurs), the routine is executed, which is the routine is checking ADC value regarding to sensor reading; in certain value range, the triac will responses whether to makes load working on full power, half or less power, or nothing at all. The triac firing is done by software/program that will be explained in this chapter.

3.2.1 Zero crossing

Zero crossing is a circuitry that will give pulse to the output every 10ms. 10ms obtained because in Indonesia the frequency of main is 220Vac 50Hz. The zero cross works to pulse when main voltage is 0. In 50Hz, voltage will be 0 every half cycle or 10ms.

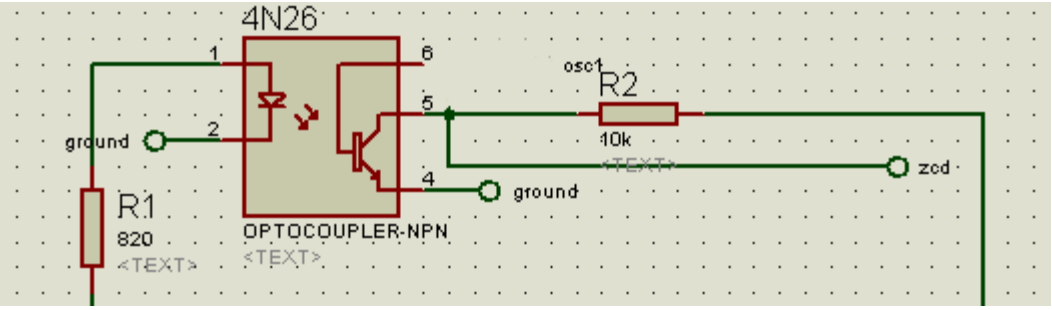

Figure 3.2 Opto transistor npn circuitry

4N26 is opto electronic specified in opto transistor NPN. Pin 1 is the input given by rectifier that not been smoothed yet, so author can acquire sine half cycle form that going through the led of 4N26.

Later on, every zero, led will conduct photo transistor that will make pin 5 of 4N26N pulsing. The pulse of the zero crossing detector will be used as external interrupt 0 (EX0/pin 3.2) interrupt source. Interrupt is a function that will alter current main process to interrupt's routine. The interrupt routine itself contains program to check sensor reading and then firing triac as desired.

The circuitry of zero crossing is assembled in this way

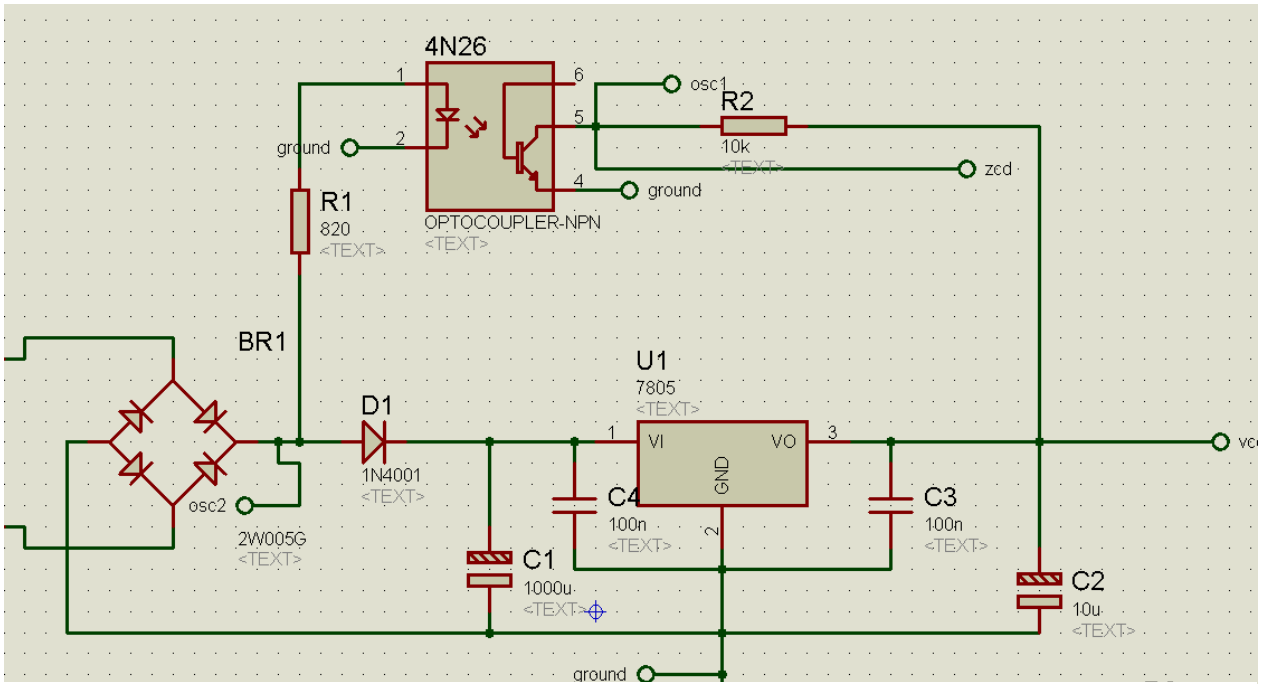

Figure 3.3 Power supply circuitry

The source is feed to cathode (pin 1) to turn on led inside opto transistor. Every led is on, transistor will be conducted and will pulse in pin 5. This pin 5 will be connected to microcontroller interrupt pin, as an external interrupt source.

3.2.2 Load circuit

Phase firing angle itself is about firing triac after certain milseconds of delay to get desired voltage. Example the application of firing triac after 5ms/ half power.

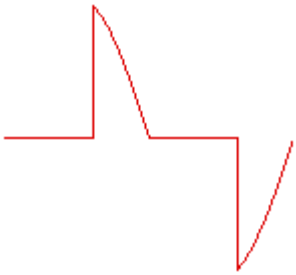

Figure 3.4 5ms delay

By figure it can be explained roughly that triac is fired after 5mS from zero crossing. Once fired, triac always goes logic high until it meet the next zero crossing.

For the triac, BT139-600E is chosen because it characteristics are:

- Has $I_{T(RMS)}$= 16A
- V_{RMS}=600V

By the specification, the triac safe for ac circuit of 220V . Triac itself need driver IC. MOC 3021 is chosen because it can conduct triac's gate which require 10mA max. MOC 3021 is an isolation device to separate MCU circuit and load's.

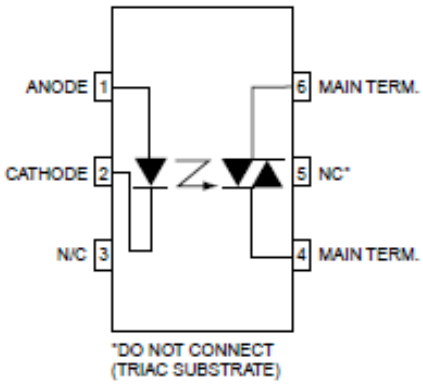

Figure 3.5 MOC3021 layout

[14]Opto diac is required because as explained before, triac has to be fired right away after certain delay, which ranges from 1-9mS. Opto electronic can reacts better that mechaic switch, means it needs device that can integrate with time precisely. The circuit of opto diac and triac is showed in figure below.

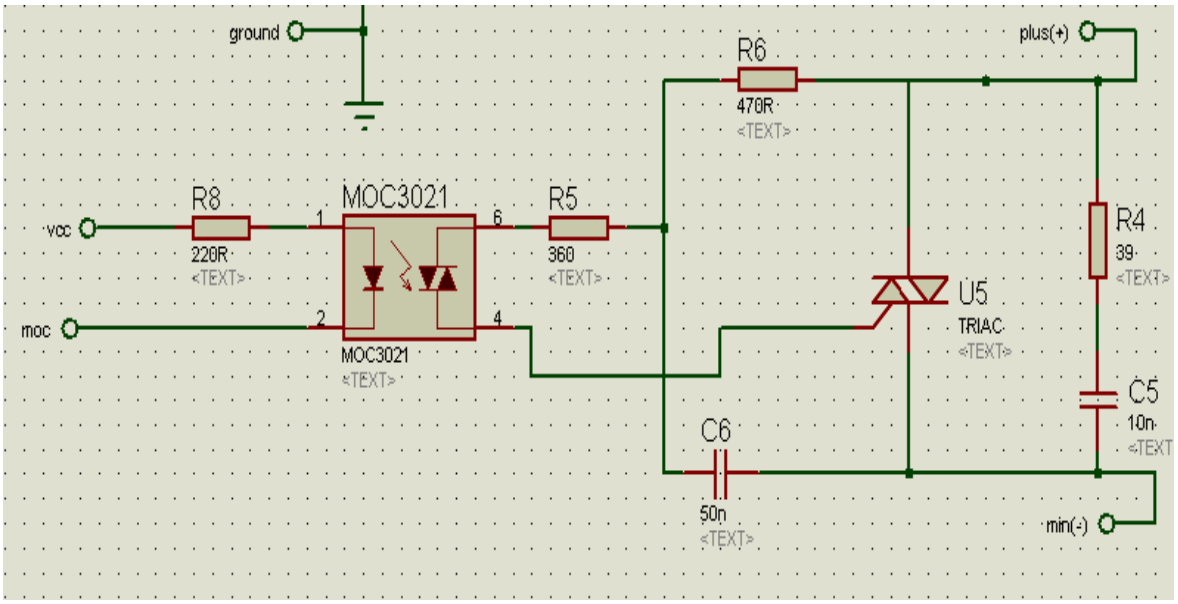

Figure 3.6 MOC3021 circuitry

The ground of led side is connected to microcontroller (active low), because by characteristic microcontroller is not stable enough when sourcing. When led is on, the triac inside MOC 3021 will connects load's circuitry.

For inductive load, snubber is often required to avoid "false turn on" .[15]. By the figure above, 470 Ohm resistor and 50nF capacitor is snubber for opto triac, while 39 Ohm resistor and 10nF capacitor is snubber for BT139-600E.

3.2.3 Microcontroller circuit

To be operated, microcontroller needs basic configuration called as minimum system. Minimum system required circuit for microcontroller to operates successfully. It contains quartz crystal for on chip oscillator, and reset. Quartz crystal assembled by parallel it with two capacitors. Reset is assembled with a resistor and a switch. Reset button is useful to reset ongoing process, and start it over again from the begining of the instruction/program.

Here is the figure of minimum system for AT89S52:

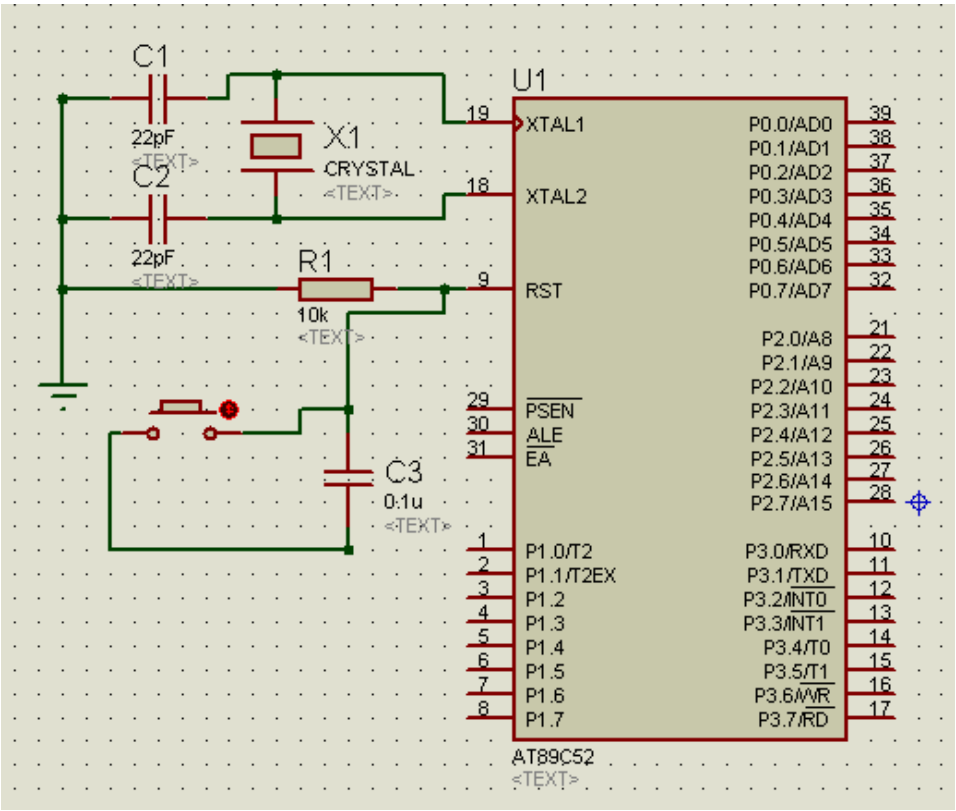

Figure 3.7 AT89S52 minimum system

Author uses quartz crystal valued 11.0592 MHz No specific reason behind the choosing of quartz crystal, since author uses pre-made AT89S52 module sold on internet. The reason of this decision is to acquire a module of microcontroller that ease to interface and programmed, since it has integrated downloader pins.

The special registers that critical for this final project are timer interrupt and external interrupt.

3.2.3.1 Timer interrupt

Interrupt timer in this chapter is required to generates clock for ADC 0809, since that IC is not equiped by internal clock. The requirement of the clock is 500 KHz, and it can be supplied by microcontroller using timer function. Interrupt timer routine will be executed once the timer overflow. To enable interrupt timer, this bit register of interrupt enable should be assign to high logic:

Table 3.1 Enable Interrupt Register Bit

1	0	0	0	0	0	1	0
EA	N/A	ET2	ES	ET1	EX1	ET0	EX0

Enabling interrupt timer can be done using software. Explanation about code and software will be elaborated in section 3.2.

3.2.3.2 External interrupt

External interrupt is an interrupt that sourced outside of microcontroller. In this project, the source of the external interrupt is zero crossing pulse. As explained in chapter 2, the zero cross circuit will pulse every sine half cycle is zero. If the zero cross is high, the routine will check ADC value; the action for each ADC value condition is executed by program.

The ilustration of the external interrupt in this project is ilustrated below:

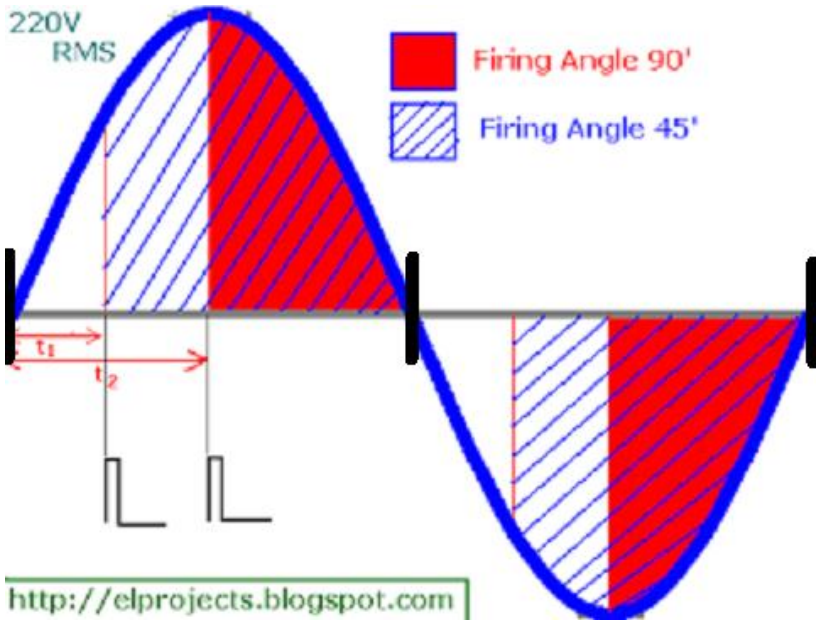

Figure 3.8 Firing angle of 90 degree

Zero cross is pulsing in the black line,where the wave hits zero. The picture shows that the triac is fired 2.5 mS after zero cross (t1), and fired 5 mS after zero cross (t2). The relation of this ilustration with author's project is external interrupt routine will executed every zero crossing pulses, then the program inside routine will check the value of ADC, whether

it is necessary or not to delay triac firing. Let the condition is triac should be fired after 5 mS, then every zero crossing pulses the routine will use delay function to generates delay of 5ms, then fires triac right away which give author the value of 90 degree or half voltage applied to load. Triac will stays high in the remaining time until next zero. External interrupt used in this project is external interrupt0 (EX0).

3.2.4 Analog to digital IC 0809

Circutry layout for ADC 0809, microcontroller, and smoke sensor are assembled in this way:

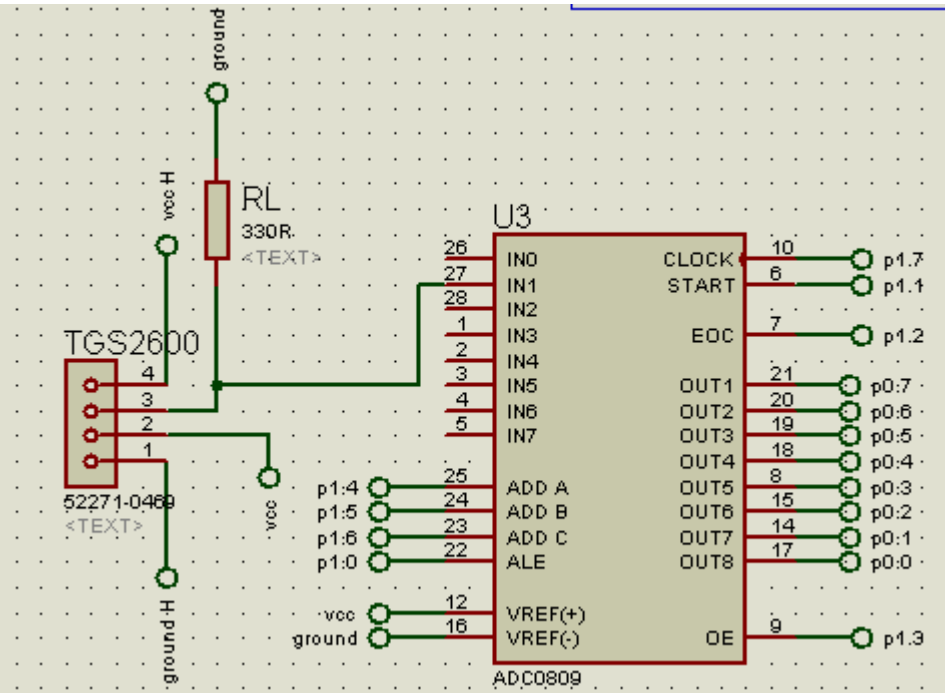

Figure 3.9 ADC 0809 circuitry

Author uses input channel 2 (IN1) for receiving data from sensor, and port 0 in microcontroller as input data from ADC. Since it is 8 channels adc, the ranges of ADC is 0-255. By using 5Vdc for Vref+ and ground for Vref-, the voltage required to step up or down ADC value can be calculated this way:

$$x = \frac{(Vref+)-(Vref-)}{2^8-1} \quad \textbf{(3.1)}$$

In this project, Vref+ is vcc (5Vdc) and Vref- is ground (0). According to calculation above, the voltage required to increase or decrease value is 19.6mV. To aqcuire more

accurate result, ADC with higher channel can be used; such as 10 channels. More channels mean longer range and more accurate result.

ADC functions such as start conversion,end of conversion, and clock have to be done using software through microcontroller.

3.2.5 LCD 16X2

Author use LCD 16X2 to displays value of ADC relating to sensor reading. It is useful to provide that information, since the controller works at certain ranges of ADC value. By displaying it, it will make author observates easier.

The connection to microcontroller is assembled this way:

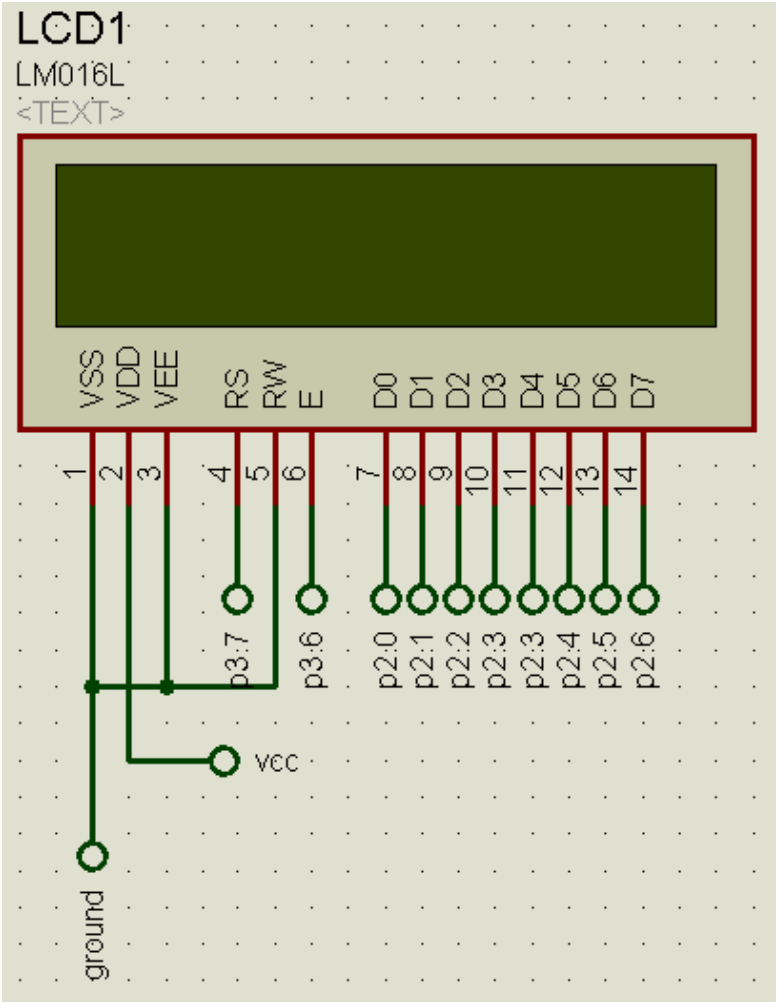

Figure 3.10 LCD 16x2 circuitry

For this project, author do not use pin 15 and 16 which are pins for backlight. This decision taken because by experiment, the controller do not works properly which caused by these pins consume high current.

3.3 Software Design

For programming purpose, author use C language instead of assembly language because author finds C language is more familiar. The C language compiler that author use is Keil uVision 3. This sub-chapter will explains the usage of the software to utilize components, such as LCD 16X2, MOC 3021, ADC 0809, and also the functions that required to make this project works as desired.

3.3.1 Delay function

Delay function is used quite common when creating microcontroller based device. It is usually used to separate instructions in several applications; such as to separate instructions between LCD data and command, separate keypad instructions, separates switch/push button instructions, and many more.

Delay that used in this project is polling mode. For crystal oscillation of 11.0592 MHz, the delay function is written this way:

```
void delay(unsigned int count) //delay function in mS
{
unsigned int i;
   while(count) {
      i = 115;
            while(i>0) i--;
      count--;
  }}
```

This delay generates in unit of mS. If the value of i is 0, then it is 1 mS. To generates x mS, later on in the program it can be done by writing:

```
delay(x);
```

Where x is desired delay in millisecond(s).

3.3.2 Timer interrupt 0

Timer interrupt is interrupt that will be interrupted once the timer overflow. This interrupt is used to generates pulse for ADC 0809 clock requirement, which is 500 KHz. To enables this function, there should be this command in the program; whether being called as a function, or written in body of program (main):

```
TMOD=0x02;  //timer0 setting for generating clock of 500KHz using interrupt enable mode
TH0=254;  //value of timer
IE=0x82;      //interrupt enable register
TR0=1;        //timer run; 1 means start, 0 means zero
```

Every the timer is overflow, the interrupt timer 0's routine is executed. The routine of this interrupt is:

```
void timer0() interrupt 1  // Function to generate clock of frequency 500KHZ using Timer 0 interrupt.
{
clk=~clk;
}
```

Where clk is pin 1.7 of microcontroller that connected to clock pin of ADC 0809. If value of clk/pin 1.7 is not predefined, then the default logic of pin 1.7 is high/1. ~clk means it negates the value of previous condition.

3.3.3 External interrupt 0

External interrupt is interrupt that its routine will be executed if there is external pulse that received by external interrupt 0 pin or pin 3.2 of microcontroller. To enables this interrupt, there must be this function whether called as a function or just written in the body of program (main):

```
void initINT0(void)      //activate int external0
 {
 IT0=1;           //interupted when transition from logic 1 to 0 (zero cross detector)
 EX0=1;         //activating external interrupt 0
 EA=1;
}
```

IT0=1 used because most of the time, zero cross consists 0 logic rather than one. If this function is used, microcontroller pin 3.2 will be interrupted only when zero cross pulse on edge falling.

3.3.4 LCD 16X2 programming

LCD needs initialization before it can be functioning. In this project, the initialization is done this way:

```
void lcd_ini()  //Function to inisialize the LCD
{
lcd_command(0x38); //set lcd to 16x2 mode
delay(5);
lcd_command(0x0E); //display on, cursor on
delay(5);
lcd_command(0x80);  //Force cursor to blink at line 1 positon 0
delay(5);
}
```

The function above will be called in the main program. Before command and data function can be use, the following initialization for command and data should be done like this:

```
void lcd_command(unsigned char comm)  //Function to send command to LCD.

{
    lcd_data_pin=comm;
    en=1;
    rs=0;
    rw=0;
    delay(10);
    en=0;
}
```

The algorithm of sending command to LCD is [19]:

- Move data to LCD port
- select command register
- select write operation
- send enable signal
- wait for LCD to process the command

For data, initialization is done this way:

```
void lcd_data(unsigned char disp)  //Function to send data to LCD.
    {
      lcd_data_pin=disp;
      en=1;
      rs=1;
      rw=0;
     delay(10);
     en=0;
     }
```

The algorithm of sending data to LCD is:

- Move data to LCD port
- select data register
- select write operation
- send enable signal
- wait for LCD to process the data

3.3.5 ADC 0809 programming

ADC needs driver function before it can operate as converter. Author use IN1 as input from sensor reading. According to its datasheet, for IN1 the address line is [6]:

Table 3.2 Address Line For IN1

Analog channel	Address line		
	C	B	A
IN1	0	0	1

Using software, the function that required to drives ADC is like this:

```
    void adc()  //Function to drive ADC
    {
```

```
        while(1)
        {
          ADD_C=0;  // Selecting input channel 2 using address lines
          ADD_B=0;
          ADD_A=1;
          delay(2);
          ale=1;
          delay(2);
          sc=1;
          delay(1);
          ale=0;
          delay(1);
          sc=0;
          while(eoc==1);
          while(eoc==0);
          oe=1;
          BCD();
          lcd_command(0x88);
         delay(2);
         oe=0;
        }
}
```

The algorithm of ADC in this project is :

- Select the channel, in this case is channel 2/ IN1.
- Enable ALE(address latch enable). Address is latched on rising edge of clock.
- Give high logic to start conversion (reset conversion). On the rising edge, the conversion is reseted; while falling edge begins the conversion.
- Set ALE logic to low.
- Set start conversion to low (begin the conversion).
- Waiting until conversion is done.
- Set output enable pin to high, to read data of ADC output.
- Send the value to LCD through BCD function.

- Send the cursor on LCD to 9th character from upper left handside.
- Stop retrieving data from ADC.
- While(1) will ensure the process inside its bracket is repeated infinitely.

The value range of this 8 channels ADC is from 0-255. Since the author use 5Vdc as Vref+, the step size is roughly 19.53 mV; this means the value of ADC will increment or decrement every 19.53 mV.

3.3.6 Sending data string to LCD

LCD needs function to be able to display certain character on its screen. The function used to send data string to LCD. The function is utilize using this code:

```
        lcd_dataa(unsigned char *disp)  //Function to send string data to LCD.
{
        int x;
        for(x=0;disp[x]!=0;x++)
        {
          lcd_data(disp[x]);
        }
}
```

Then, LCD can displays character(s) just by calling this function, such as lcd_dataa("value :") to displays "value :".

3.3.7 External interrupt 0 routine

Once external interrupt 0 has been initialized, the pin 3.2 of microcontroller will reacts to pulse given to that pin, in this case zero crossing pulse. Once the interrupt 0 trigerred, following routine is executed:

```
void external0_isr(void) interrupt 0 //interrupt routine
```

```
        {
        triac=1;

        if((input_port>=10) && (input_port<=20))   //1st condition
        {
        delay(5);
        triac=0;
        }
           else if((input_port>20) && (input_port<=30))      //2nd condition
            {
                  delay(4);
                  triac=0;
                 }
                 else if((input_port>30) && (input_port<=40))        //3rd condition
               {
                     delay(3);
                           triac=0;
                  }

               else if(input_port>40) //4th condition; full power
               {
                           triac=0;
                          }
                           else
                            {}
}
```

Initialy, the logic of triac pin (p3.0) is high. Triac will be turned on if its logic is low because the configuration used is active low. When interrupt detects zero crossing pulse, its routine firstly will set triac to off position, then regarding to ADC's value one of the if function will be conducted. After done executing a condition, the microcontroller process will return to ADC reading. When interrupt occurs again, its routine will set triac to off then executing one of the if function again regarding to ADC's value. The process is repeated that way.

3.3.8 Main program

Main program is the body of program. In this function, several/all of the functions will be called and synergized one another. Main program also define the initial value of microcontroller pin(s), if needed.

For this project, the main program is written this way:

```
void main()
{
intr0=1;
triac=1;
eoc=1;
ale=0;
oe=0;
sc=0;
key1=0;
TMOD=0x02;  //timer0 setting for generating clock of 500KHz using interrupt enable mode.
TH0=234;
IE=0x82;
initINT0();
TR0=1;
lcd_ini();
lcd_dataa("Value: ");
lcd_command(0x88);
adc();
}
```

From program above, it tells that it define initial logic of severeal pins of microcontroller, followed by instruction of timer interrupt, and enabling external interrupt 0 again for the second time because IE=0x82 disables it for interrupt timer purpose. The end of the program is calling ADC function, where inside ADC function there is while(1) function that will repeat the process all the the time. When interrupt external 0 occur, it will intercept process that occurs inside adc(); and back again to adc(); once interrupt routine is finished.

3.3.9 Project algorithm

Here is the flow chart of project algorithm:

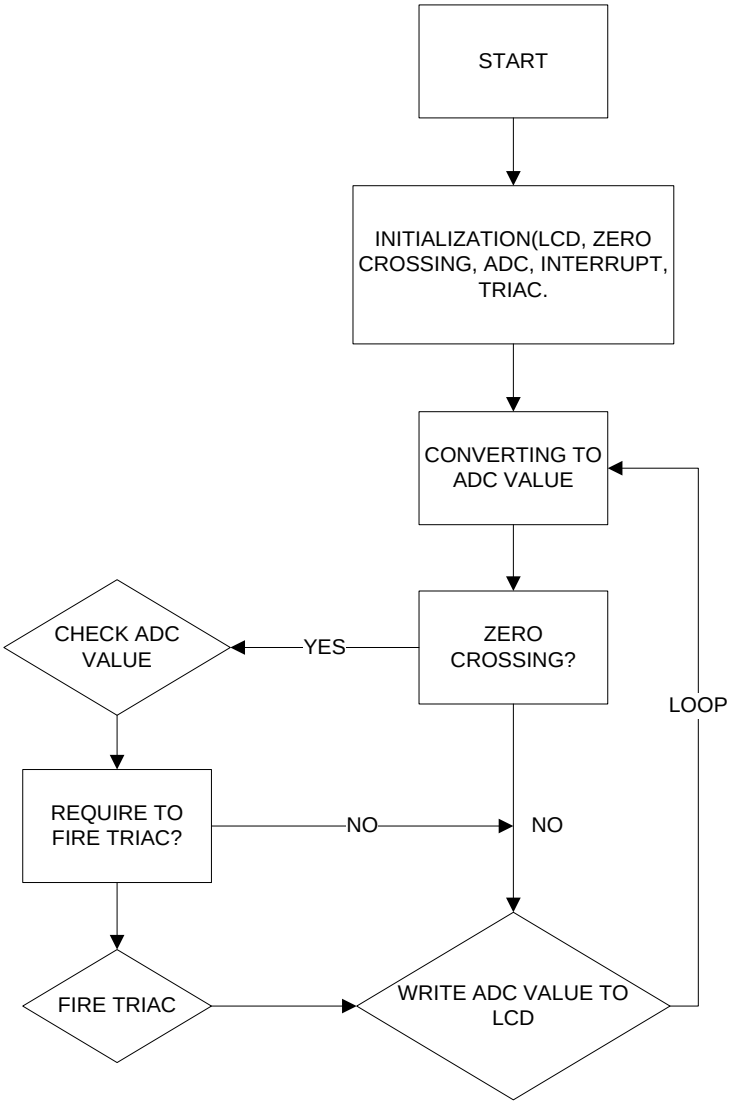

Figure 3.11 Flow chart of the project

Firstly, initialization is done by software as explained above. Then ADC 0809 is start converting based on sensor resistance ratio. Function adc(), which is funtion to drives ADC , always looping its process; it keeps converting. After that, every zero cross pulse occurs the external interrupt 0 is executing its routine, which is check whether it needed to fire triac or not. The end of the process is LCD displaying current ADC value, then the process is returned to converting ADC value.

CHAPTER 4
PROJECT RESULT AND ANALYSIS

4.1 Preliminary Remarks

This chapter contains final result of the project, and analysis from author's point of view during working on this project.

4.2 Project Result

Configurations of delay before firing triac in this project are limited to 3mS, 4mS, and 5mS. By testing each of delay for 10 times, here are the average of result:

Table 4.1 Collected data

Delay (milli second)	Voltage (Volt)
0	215.9
3	188.84
4	154.08
5	108.66

For ADC value >=10 and <=20, delay is 5mS; ADC value >20 and <=30, delay is 4mS; ADC value >30 and <=40, delay is 3ms; and ADC>40, delay is 0 or no delay.

This final project's zero crossing is shown in figure 4.1 below:

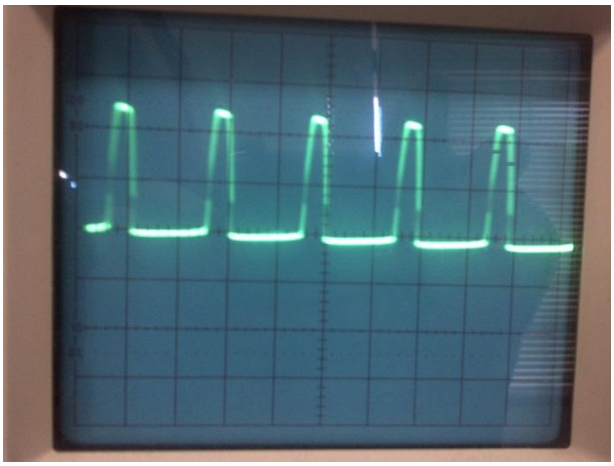

Figure 4.1 Zero crossing pulses

Combining with main's signal that enters opto transistor ic, the signal will be formed like figure 4.2 below:

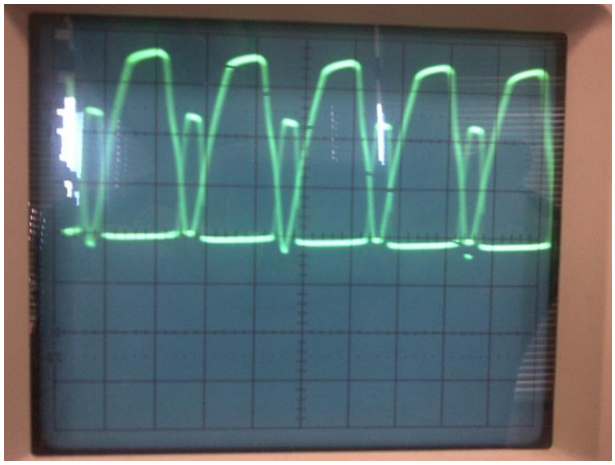

Figure 4.2 Zero crossing pulsing each 0

It is seen that every zero, there is pulse which become interrupt external source. Both figures above taken with scale 5ms, and 1 Volt per division. Compared to figure 2.10, the data collection is quite close[2]. Vout graph is refering to figure 2.10.

Table 4.2 Data comparison

Delay	Vout measured (Volt)	Vout graph (Volt)	Difference (%)
0	215,9	220	1,86

[2] Data obtained by assuming 3ms is 85% of max voltage, 4ms is 70%, and 5ms is 50%.

3	188,84	187	0,98
4	154,08	154	0,05
5	108,66	110	1,2

From table above, the following graph can be obtained:

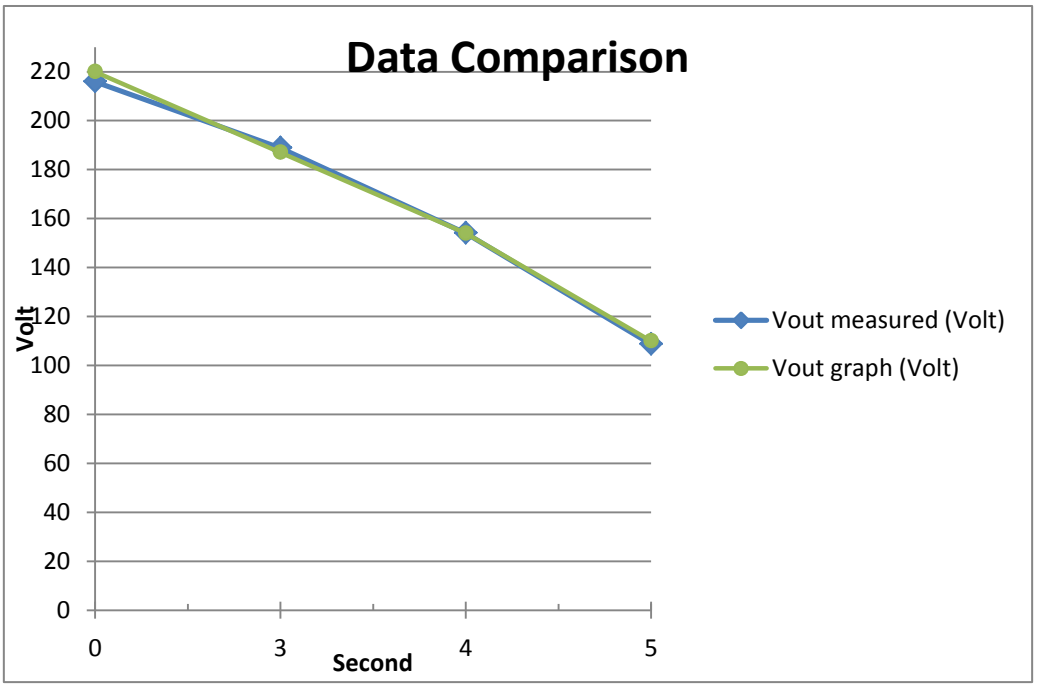

Figure 4.3 Comparison chart

However, the voltage measured by voltmeter due to triac firing is neither root mean square value, nor absolute average value. It happen because voltmeter commonly designed to read in term of root mean square that can be done either by 'true RMS' voltmeter or by altering correction factor (form factor) for absolute average-responding voltmeter. Signal chopped by triac firing is not in form of sinusoidal, that is why voltmeter will display faulty value [21].

Because the wave of chopped signal as phase firing angle product is not pure sine, voltmeter will not give correct reading in term of root mean square. Absolute average value itself is the absolute value of sine wave average value [21].

Pure sine wave will have zero average because positive cycle will have same area as negative cycle, hence it will cancel each other. To get the absolute average value or average rectified value, sine wave cycle should considered to be positive [20].

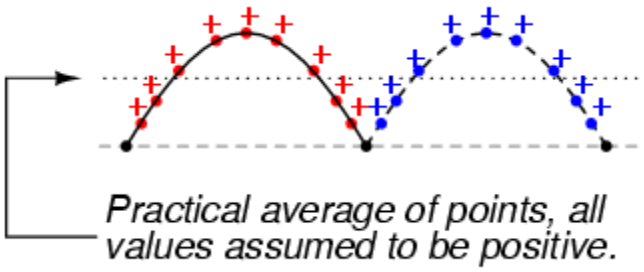

Figure 4.4 Average rectified value

There is a factor that will convert absolute average value to root mean square. The value can be obtained by calculating ratio of peak value to root mean square, and peak value to absolute average value [21].

Since root mean square is 0.707 times peak value, and absolute average value is 0.637 times peak value, the factor is ratio between root mean square and absolute rectified value is 1.1099 [21]. However, it only applies for pure sine wave. Table 4.3 will shows the relation and comparison for each term of voltage.

Table 4.3 Comparison between voltage

Voltage abs average (Volt)	Voltage RMS (Volt)	Conducting angle (degree)	Voltage measured (Volt)
197,9179286	220	180	215,9
193,0671677	219,289	162	
178,9904334	214,583	144	
157,0677496	202,992	126	188,84
129,4483243	183,215	108	154,08
98,83985006	155,563	90	108,66

68,24305326	121,787	72	
40,65751551	84,816	54	
18,78760721	48,517	36	
4,777362415	17,67	18	
0,000286749	0	0	

From data of table 4.3, chart in figure 4.5 Can be obtained.

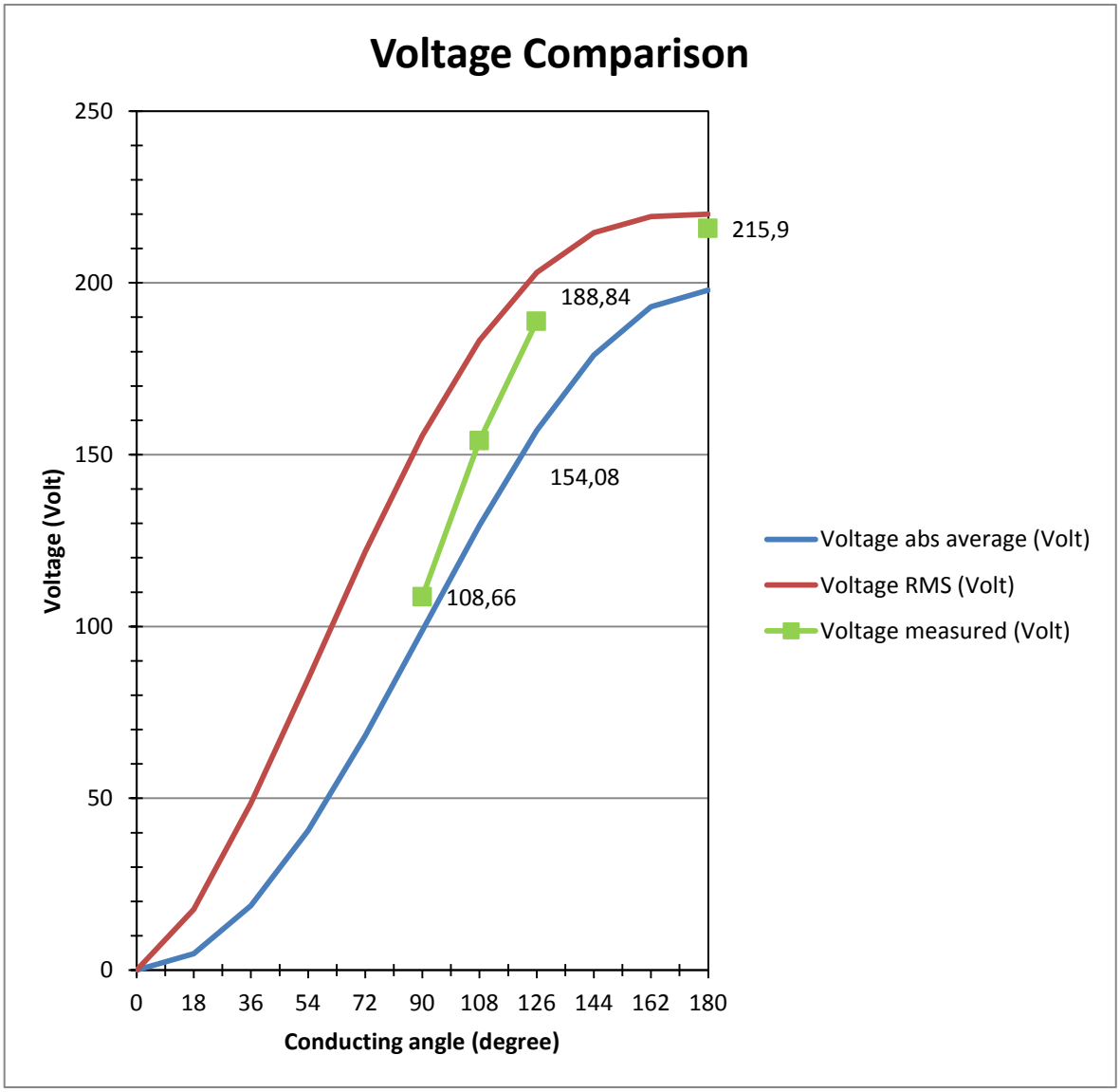

Figure 4.5 Voltage comparison

To get what the voltmeter shows, absolute average value have to be multiplied by 1.11. Table 4.4 will show the difference between practical and calculation.

Table 4.4 Absolute average volttage comparison

Conducting angle (degree)	Delay time (millisecond)	Practical (Volt)	Calculation (Volt)	Difference (%)
180	0	215.9	219.68	1.72
126	3	188,84	174.34	8.3
108	4	154,08	143.68	7.2
90	5	108,66	109.71	0.95

The absolute average voltage is obtained from this calculation

$$Vavg = \int_{\alpha}^{\pi} \tfrac{1}{\pi} \times Vpeak \times \sin(\omega t) d\omega t$$

$$Vavg = \frac{Vpeak}{\pi} \times \int_{\alpha}^{\pi} \sin(\omega t) d\omega t$$

$$Vavg = \frac{Vpeak}{\pi} \times (\cos\alpha + 1)$$

$$\therefore Vavg = \frac{Vpeak}{\pi} \times (\cos\alpha + 1) \qquad \textbf{(4.1)}$$

Inductive load waveform is not in phase between voltage and current. It also not forming pure sinusoidal wave form. Hence it contributes to slight result difference. The example of inductive load waveform is shown in figure 4.6 below [22].

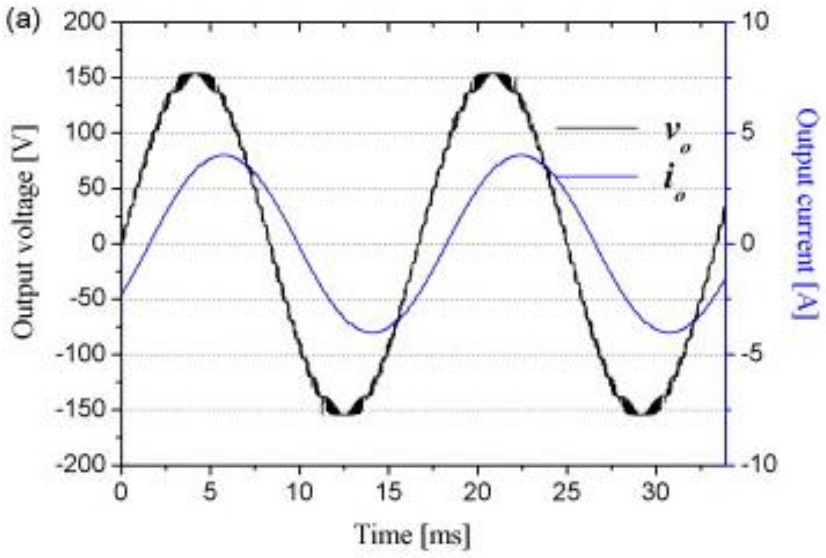

Figure 4.6 Inductive load waveform

4.3 Analysis

As can be seen in data above, slight differences can occur and possible because of sevral aspects:

- Inductive load output voltage is distorted-sine wave.
- Misreading the measurement devices (digital volt meter, and osilloscope).
- Error tollerance from measurement devices, and electrical components itself.
- Fluctuative main voltage.
- Several approaches are done during the making of this project.
- Number correction.

4.4 Discussion

When working on this project, author find several problems. To overcome the problems, author seek aid from final project supervisor, and also other reliable references. The major problems that faced by author will be elaborated below.

Circuit will not work if pin 15 and 16 of LCD is connected. Pin 15 and 16 are Vcc and ground pin for LCD back light. Posible cause is LCD backlight consumes voltage and require current. Since author's power supply is only 1 ampere maximum, then author decides to not use pin 15 and 16 of LCD. LCD display still visible, but not having backlight.

Circuit will not work if heater for sensor is supplied by the same power supply that powers the main microcontroller circuitry. To overcome this, author makes a simple power supply just to provide power to sensor's heater pins.

Multimeter is commonly designed to measured in term of root mean square. Since the chopped signal due to triac firing is not forming pure sinusoidal wave, the multimeter shows 'wrong' value. Approximation can be taken by calculating ratio between root mean square value and absolute average value.

Crystal oscillator in amount of 11.0592 MHz will not gives propper 500 KHz clock required by adc. It is because 11.0592 MHz will spent 1.085 to finish 1 machine cycle. 500 KHz clock means microcontroller should pulse every 2 microseconds, which equals to 2 machine cycles for 12 MHz crystal oscillator; while 11.0592 MHz will pulse every 2.17 microseconds. Hence using timer interrupt function, 500 KHz clock cannot be achieved with 11.0592 MHz crystal oscillator.

CHAPTER 5
CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This project provides a model of controlable load based on certain input, in this case ADC value of a sensor. Exhaust fan these days usually works manualy; even when there are several modes in an exhaust fan, they still accessed manualy. The result of this project is showing that exhaust fan can be modified to be operated automatically. By decreasing the voltage flows through load, means the power that consumed by the load is reduced as well.

5.2 Recommendations

This project is met its objective, which is controlling the voltge that received by load based on sensor reading, by utilizing phase control using triac. However, this project can be improved and expanded further by:

- Adding several sensors to get better sensitivity.
- Implement it in kitchen or smoking room to get better power efficiency.
- Replace microcontroller with other microcontroller that have more features than 8051 series (such as internal ADC, easier program compiler, easier interfacing).
- Replace all of the wiring into full PCB or at least always minimize wire usage, to avoid project malfunctioning caused by defected cable/wire.

That are some of the possible suggestions to make this project better.

REFERENCES

[1] Wisegeek.com. “What Is an Exhaust Fan?” Available : http://www.wisegeek.com/what-is-an-exhaust-fan.htm, 2003 [December 2013].

[2] En.wikipedia.org. “Voltage controller.” Available : http://en.wikipedia.org/wiki/Voltage_controller, 20 December 2012 [December 2012]

[3] En.wikipedia.org. “Phase fired controllers.” Available : http://en.wikipedia.org/wiki/Phase_fired_controllers, 10 February 2013 [February 2013]

[4] Mikroe.com “Architecture and programming of 8051 MCU's.” Available : http://www.mikroe.com/chapters/view/65/chapter-2-8051-microcontroller-architecture/, [February 2013]

[5] Atmel.com “AT89S52 data sheet.” Availabe : http://www.atmel.com/images/doc1919.pdf, [Octoberr 2012]

[6] National.com “ADC0808/ADC0809 data sheet.” Available : http://www.national.com/ds/AD/ADC0808.pdf, [December 2013]

[7] Agfianto Eko Putra. *Belajar Mikrokontroler AT 89C51/52/55 (Teori dan Aplikasi) Edisi 2*. Jakarta, Gavamedia, 2005, ISBN: 979-3469-16-1.

[8] Totok Budioko. *Belajar Dengan Mudah dan Cepat Pemrograman Bahasa C Dengan SDCC (Smole Device C Compiler) Pada Mikrokontroler AT89X051/AT89C51/52 Teori, Simulasi dan Aplikasi*. Jakarta, Gavamedia, 2005, ISBN: 979-3469-59-5.

[9] M. Arif Imron. ‘PENGATURAN pH AIR PADA KOLAM LOBSTER MENGGUNAKAN POMPA MOTOR INDUKSI SATU FASA RUN KAPASITOR DENGAN TRIAC BERBASISKAN MIKROKONTROLER AT89S52.’ B. Eng thesis, Dept. Electrical Engineering, Universitas Katolik Indonesia Atma Jaya, Jakarta, Indonesia, 2009.

[10] Nxp.com. “Triac BT139-600E data sheet.” Available : http://www.nxp.com/documents/data_sheet/BT139-600E.pdf, [January 2013]

[11] Digiware.com. “TOPWAY 16x2 LCD data sheet.” Available : https://docs.google.com/file/d/0BzkNNhuEnaF-

MjYxMTIxNzYtNmQ2Yi00MzM4LWJhNGUtNzNjOWRlY2FiZGI5/edit?pli=1&hl=en#, [January 2013]

[12] 8051projects.net. “commands and instuction set.” Available : http://www.8051projects.net/lcd-interfacing/commands.php, [January 2013]

[13] Sengpielaudio.com. “The Phase in the Acoustics.” Available : http://www.sengpielaudio.com/calculator-timedelayphase.htm, [February 2013]

[14] Datasheetcatalog.org. “MOC3021 data sheet.” Available : http://www.datasheetcatalog.org/datasheet/motorola/MOC3021.pdf, [December 2012]

[15] Muhammad Ahmed. “MOC3021 Circuitry for Inductive Load with Snubber circuit.” Available : http://elprojects.blogspot.com/2011/07/moc3021-circuitry-for-inductive-load.html, [January 2013]

[16] Voltsandbytes.com. “8051 Tutorial 4: 8051 Timer/Counter Programming in C.” Available : http://voltsandbytes.com/8051-tutorial-4-8051-timercounter-programming-in-c/3/, [December 2012]

[17] Littlefuse.com. “Phase Control Using Thyristors” Available : http://www.littelfuse.com/products/~/media/Files/Littelfuse/Technical%20Resources/Documents/Application%20Notes/an1003.pdf, [February 2013]

[18] Ece.ualberta.ca. “Getting an ADC0809 Analog to Digital Converter to Work for You.” Available : http://www.ece.ualberta.ca/~elliott/ee552/studentAppNotes/1999f/ad_converter/, [February 2013]

[19] 8051projects.net. “Sending commands to LCD.” Available : http://www.8051projects.net/lcd-interfacing/lcd-programming.php, [February 2013]

[20] allaboutcircuits.com.”Measurements of AC magnitudes.”Available : http://www.allaboutcircuits.com/vol_2/chpt_1/3.html#02412.png, [March 2013]

[21] allaboutcircuits.com.”AC Voltmeters and Ammeters.”Available : http://www.allaboutcircuits.com/vol_2/chpt_12/1.html, [March 2013]

[22] sciencedirect.com.”Modified multilevel inverter employing half- and full-bridge cells with cascade transformer and its extension to photovoltaic power generation”Available : http://www.sciencedirect.com/science/article/pii/S0378779610001446, [March 2013]

APPENDIX A

SOURCE CODE

```
// Program to control triac firing based on sensor reading (ADC 0809 value)
#include<reg52.h>
#include<stdio.h>
sbit ale=P1^0;  //address latch enable
sbit oe=P1^3;  //output enable
sbit sc=P1^1;  //start conversion
sbit eoc=P1^2;  //end of conversion
sbit clk=P1^7;  // clock
sbit ADD_A=P1^4;  // Address pins for selecting input channels.
sbit ADD_B=P1^5;
sbit ADD_C=P1^6;
sfr lcd_data_pin=0xA0;  //P2 port
sbit rs=P3^7;
sbit rw=P3^1;
sbit en=P3^6;
sbit triac=P3^0;  //triac controller pin
sbit intr0=P3^2;
sfr input_port=0x80;  //P0 port
unsigned                                                                    int
bitvalue,decimal_value,key,left_value,value,number,ascii1,ascii2,ascii3,flag,key1;
void initINT0(void);


void timer0() interrupt 1  // Function to generate clock of frequency 500KHZ using Timer
0 interrupt.
{
clk=~clk;
}

void delay(unsigned int count) //delay function in mS
{
unsigned int i;
   while(count) {
      i = 115;
            while(i>0) i--;
      count--;
}}



void initINT0(void)      //activate int external0
 {
 IT0=1;           //interupted when transition from logic 1 to 0 (zero cross detector)
```

```
 EX0=1;
 EA=1;
 }


void lcd_command(unsigned char comm)  //Function to send command to LCD.
{
lcd_data_pin=comm;
en=1;
rs=0;
rw=0;
delay(10);
en=0;
}

void lcd_data(unsigned char disp)  //Function to send data to LCD.
{
lcd_data_pin=disp;
en=1;
rs=1;
rw=0;
delay(10);
en=0;
}

lcd_dataa(unsigned char *disp)  //Function to send string data to LCD.
{
int x;
for(x=0;disp[x]!=0;x++)
{
  lcd_data(disp[x]);
}
}

void lcd_ini()  //Function to inisialize the LCD
{
lcd_command(0x38);   //Set LCD mode to 2 line (for 8 bit)
delay(5);
lcd_command(0x0E);  //Display on, cursor on
delay(5);
lcd_command(0x80);  //Force cursor to blink at line 1 positon 0
delay(5);
}

void BCD()  // Binary to decimal conversion to send the data to LCD
{
  key1++;
  key=0;
  flag=0;
    number=input_port;
```

```
    value=number%10;
number=number/10;
ascii1=value+48;
if(number!=0)
{
  value=number%10;
  number=number/10;
  ascii2=value+48;
    flag=1;
}
else
{
   ascii2=48;
   flag=1;
}
if(number!=0)
{
  value=number%10;
    number=number/10;
    ascii3=value+48;
    key=2;
}
else
{
  ascii3=48;
  key=2;
}
if(key==2)
lcd_data(ascii3);
if(flag==1)
lcd_data(ascii2);
lcd_data(ascii1);
if(key1==3)
{
key1=0;
ascii3=0;
ascii2=0;
ascii1=0;
delay(10);
}
}

void adc()  //Function to drive ADC
{
  while(1)
  {
  ADD_C=0;  // Selecting input channel 2 using address lines
  ADD_B=0;
  ADD_A=1;
  delay(2);
```

```
 ale=1;
 delay(2);
 sc=1;
 delay(1);
 ale=0;
 delay(1);
 sc=0;
 while(eoc==1);
 while(eoc==0);
 oe=1;
 BCD();
 lcd_command(0x88);
 delay(2);
 oe=0;
 }
 }

void main()
{

intr0=1;
triac=1;
eoc=1;
ale=0;
oe=0;
sc=0;
key1=0;
TMOD=0x02;  //timer0 setting for generating clock of 500KHz using interrupt enable mode, and timer 1 mode 1.
TH0=234;
IE=0x82;
initINT0();
TR0=1;
lcd_ini();
lcd_dataa("Value: ");
lcd_command(0x88);
adc();

}

void external0_isr(void) interrupt 0 //interrupt routine
{EX0=0;

triac=1;

if((input_port>=10) && (input_port<=20))   //1st condition 5ms delay
{
delay(5);
triac=0;
```

```
}
   else if((input_port>20) && (input_port<=30))      //2nd condition 4ms delay
     {
            delay(4);
            triac=0;
           }
           else if((input_port>30) && (input_port<=40))        //3rd condition 3ms delay
        {
              delay(3);
                    triac=0;
            }

        else if(input_port>40) //4th condition; full power no delay
        {
                    triac=0;
                   }
                    else
                     {}
         EX0=1;
}
```

APPENDIX B

DATA COLLECTION

Table B.1 Data collection after 10 times observation

Delay (ms)	**Voltage output (volt)**										**Average (volt)**
0	215	216	216	216	216	216	216	216	216	216	215.9
3	188.5	187.8	188.7	188.8	189.4	188.2	189.9	189.3	189.1	188.7	188.84
4	154	155.1	153.6	153.3	153.7	154.1	153.7	155	154.6	154.7	154.08
5	104.6	108	107.8	108.1	109.7	109.5	109.2	109.8	110.1	109.8	108.66

APPENDIX C

HARDWARE AND FULL CIRCUIT OF THE PROJECT

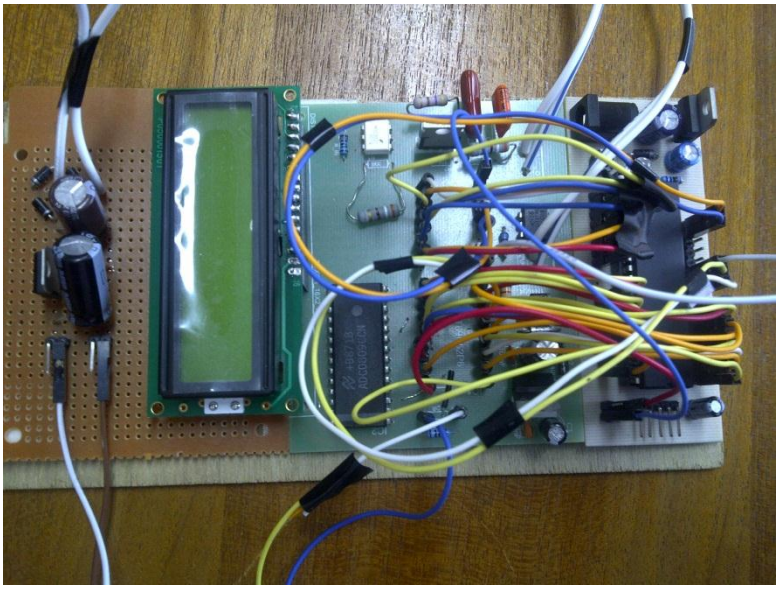

Figure C.1 Controller

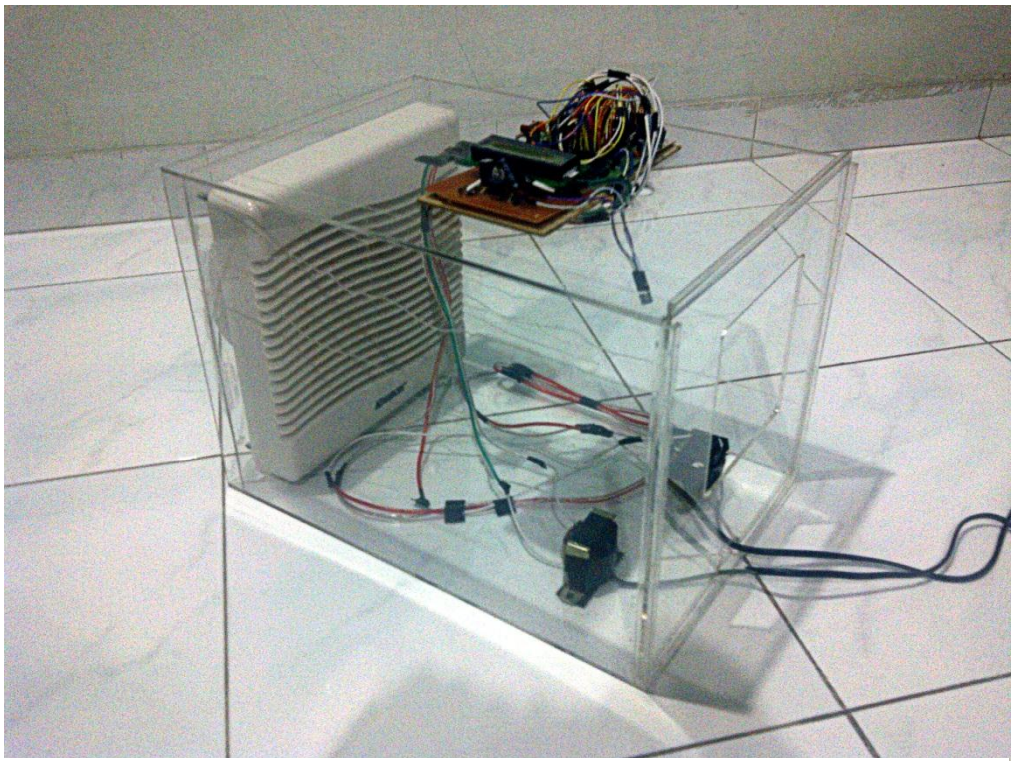

Figure C.2 Final appearance of the model

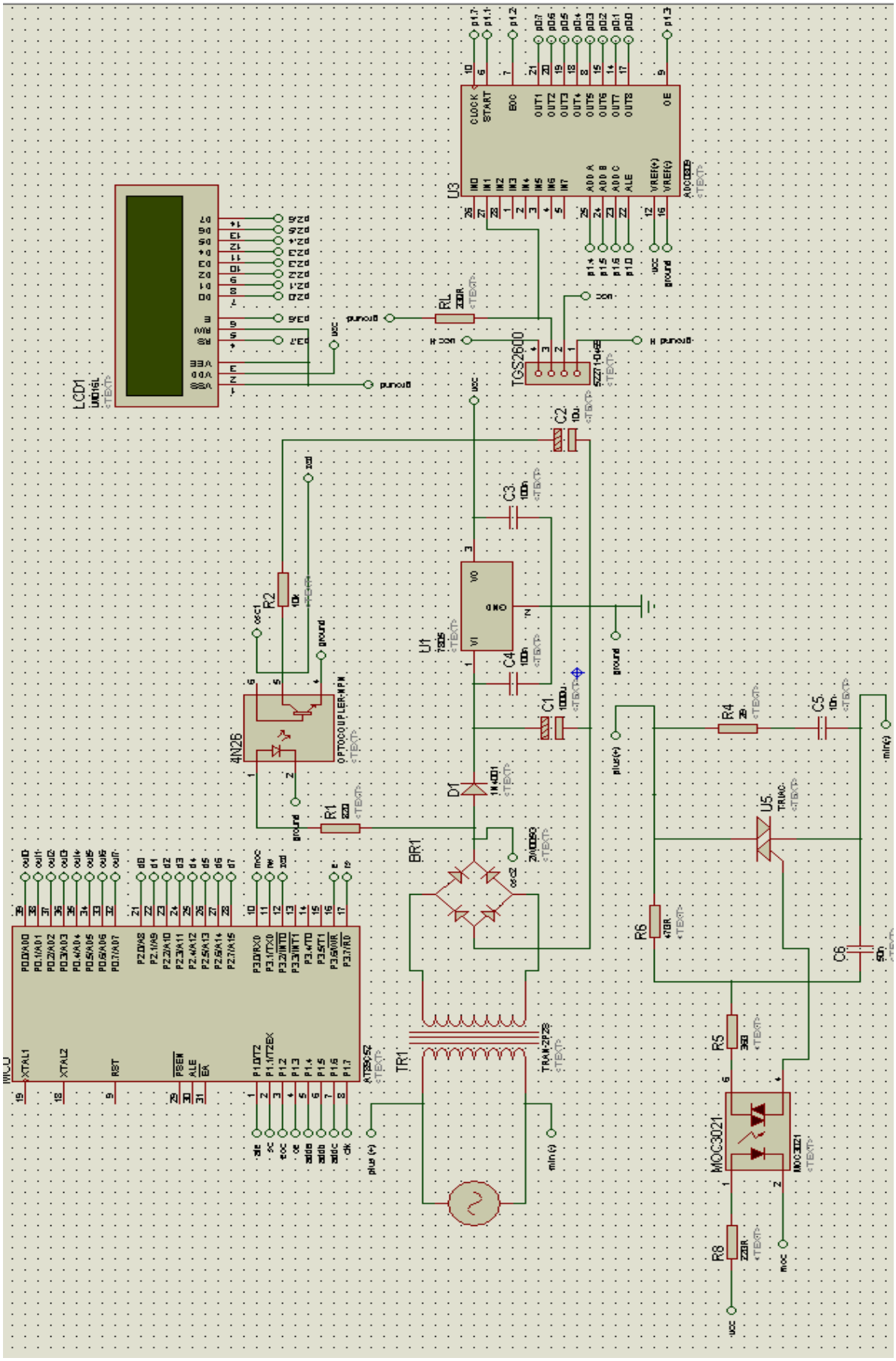

Figure C.3 Full circuit

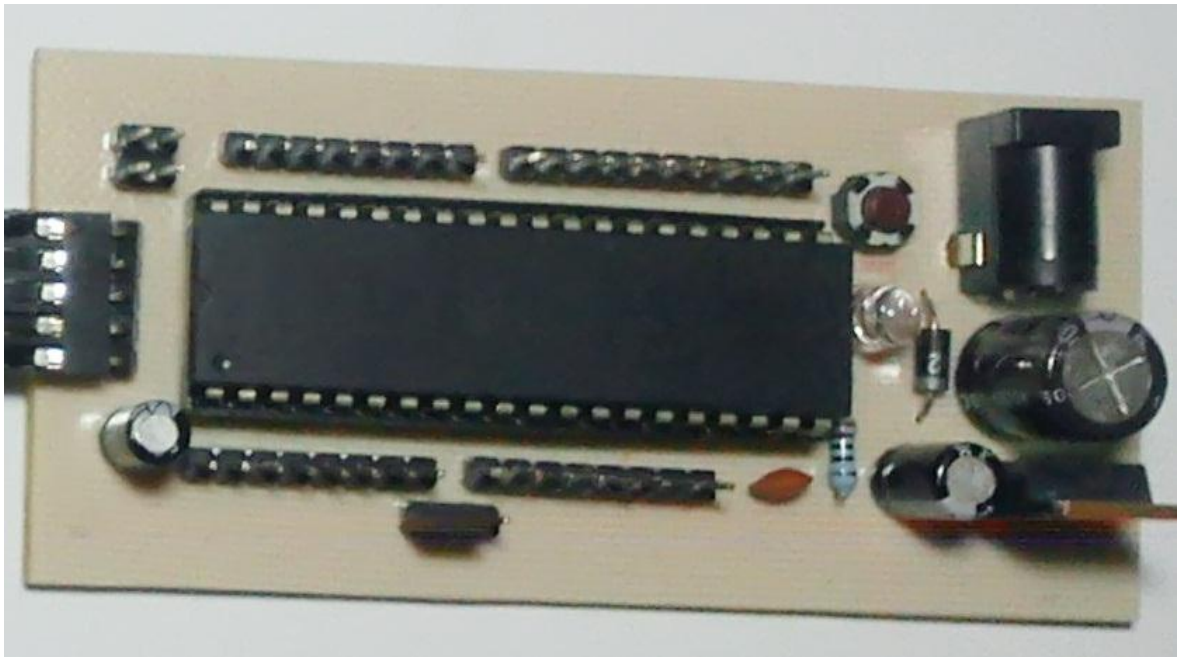

Figure C.4 Minimum system of AT89S52