

UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS♦

JUDUL: **AUTOMATIC DETECTION TEMPERATURE
TRANSMITTER FOR CALIBRATION PROCESS USING
THERMOCOUPLE**

SESI PENGAJIAN: 2010/2010

Saya MUHAMAD FARID BIN A.WAHAB (880927-23-5453)
(HURUF BESAR)

mengaku membenarkan tesis (Sarjana Muda/~~Sarjana~~ /~~Doktor Falsafah~~)* ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Tesis adalah hakmilik Universiti Malaysia Pahang (UMP).
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (✓)

SULIT

(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

TERHAD

(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)

TIDAK TERHAD

Disahkan oleh:

(TANDATANGAN PENULIS)

(TANDATANGAN PENYELIA)

Alamat Tetap:

**NO.2 JALAN HANG JEBAT 49
TAMAN SKUDAI BARU 81300
JOHOR BAHRU JOHOR**

NAJIDAH BINTI HAMBALI
(Nama Penyelia)

Tarikh: **29 NOVEMBER 2010**

Tarikh: : **29 NOVEMBER 2010**

- CATATAN:
- * Potong yang tidak berkenaan.
 - ** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh tesis ini perlu dikelaskan sebagai atau TERHAD.
 - ♦ Tesis dimaksudkan sebagai tesis bagi Ijazah doktor Falsafah dan Sarjana secara Penyelidikan, atau disertasi bagi pengajian secara kerja kursus dan penyelidikan, atau Laporan Projek Sarjana Muda (PSM).

AUTOMATIC DETECTION TEMPERATURE TRANSMITTER FOR CALIBRATION
PROCESS USING THERMOCOUPLE

MUHAMAD FARID BIN A.WAHAB

This thesis is submitted as partial fulfillment of the requirements for the award of the Bachelor of
Electrical Engineering (Hons.) (Electronics)

Faculty of Electrical & Electronics Engineering
Universiti Malaysia Pahang

NOVEMBER, 2010

“All the trademark and copyrights use herein are property of their respective owner. References of information from other sources are quoted accordingly; otherwise the information presented in this report is solely work of the author.”

Signature : _____

Author : MUHAMAD FARID BIN A. WAHAB

Date : 29 NOVEMBER 2010

“I hereby acknowledge that the scope and quality of this thesis is qualified for the award of the
Bachelor Degree of Electrical Engineering (Electronics)”

Signature : _____

Name : NAJIDAH BINTI HAMBALI

Date : 29 NOVEMBER 2010

DEDICATION

Specially dedicated to

My beloved family, and those who have guided and inspired me

Throughout my journey of learning

ACKNOWLEDGEMENT

Throughout the development of this project I have gained chances to learn new skills and knowledge. I wish to express my sincere appreciation and gratitude to my supervisor, Puan Najidah binti Hambali for his continuous guidance, concern, encouragement and advices which gave inspiration in accomplishing my final year project.

Special thanks to Universiti Malaysia Pahang for supporting and providing equipment and information sources that assisted my studies and projects.

Thanks also to the Engineer Instructor, Mr. Shahrizal bin Saat and laboratory assistants, Mr. Muhammad Hamka bin Embong for the support and guidance not know the meaning of tireless in assisting me in completing this project.

My sincere appreciation to the lecturers of Faculty of Electrical and Electronics Engineering who have put in effort to the lectures and always nurture and guide us with precious advices. Thank you for sharing those experiences.

To all my lovely current and ex roommates and friends who always willingly assist and support me throughout my journey of education, you all deserve my wholehearted appreciation. Many thanks.

Last but not least, my beloved family members who always stand by my side concerning the ups and downs of my life. Home is where I find comfort. Endless love.

ABSTRACT

The purpose of this project is to detect temperature for calibration process using type K thermocouple automatically. This project will be implementing using Visual Basic 2008 software to develop Graphic User Interface (GUI). The Type K thermocouple will be use as temperature sensor in calibration process. The implementation of proportional–integral–derivative controller (PID controller) will be monitor the automatic temperature set point and detection system. The Data Acquisition (DAQ) card will be use for interfacing process is to implementation of an automatic detection system for temperature measuring during calibration process. The accuracy of the measurement will be monitor besides the analysis of uncertainty and confidence limit.

ABSTRAK

Projek ini bertujuan untuk mengesan suhu untuk proses kalibrasi dengan menggunakan *thermocouple* jenis K secara automatik. Projek ini akan menggunakan Visual Basic 2008 iaitu perisian untuk membangunkan Grafik Pengguna Antaramuka (APK). *thermocouple* jenis K akan digunakan sebagai pengesan suhu dalam proses kalibrasi. Penerapan kawalan PID (*proportional–integral–derivative controller*) akan memantau titik permulaan suhu automatik dan sistem pengesanan. Kad Pengambilalihan Data (DAQ) akan digunakan untuk proses antaramuka bagi pelaksanaan sistem pengesanan automatik untuk mengukur suhu semasa proses kalibrasi. Ketepatan pengukuran akan dipantau selain analisis ketidakpastian dan had keyakinan.

TABLE OF CONTENTS

	TITLE PAGE	PAGE
	TITLE PAGE	i
	DECLARATION	ii
	DEDICATION	iv
	ACKNOWLEDGEMENT	v
	ABSTACT	vi
	ABSTRAK	vii
	TABLE OF CONTENTS	viii
	LIST OF TABLES	xi
	LIST OF FIGURES	xii
	LIST OF ABBREVIATIONS	xiv
	LIST OF APPENDICES	xv
CHAPTER 1	INTRODUCTION	
	1.1 Background	1
	1.2 Objective	3
	1.3 Scope	4
	1.4 Research Methodology	5
CHAPTER 2	LITERATURE REVIEW	
	2.1 Literature Review	7
	2.2 type K Thermocouple sensor	7

	2.3 Data Acquisition	10
	2.4 Visual Basic	11
	2.5 Graphical User Interface (GUI)	13
	2.6 Calibration	15
	2.7 Standard Deviation	16
CHAPTER 3	HARDWARE	
	3.1 Overall system connection	18
	3.1.1 Basic System connection	19
	3.2 Instrument	
	3.2.1 Thermocouple	20
	3.2.2 HART 375 Field Communicator	21
	3.2.3 Yokogawa Temperature Transmitter YT110	22
	3.2.4 Endress+Hauser Ecograph T RSG30	23
	3.2.5 Isotech Jupiter 650B Temperature Bath	24
	3.2.6 Advantech USB-4716 DAQ Card	25
	3.2.7 Decade Box	30
CHAPTER 4	SOFTWARE	
	4.1 Software Development	31
	4.1.1 Driver Installation	31
	4.1.2 General Procedure on Installing Driver	32
	4.2 Creating Graphical User Interface (GUI)	36
	4.2.1 Uncertainty Calculation	38
	4.3 Connecting USB-4716 to Computer	41
	4.4 General Procedure Using the Software	46

CHAPTER 5	RESULT AND ANALYSIS	
	5.1 Introduction	47
	5.2 Result of the Experiment	47
	5.3 Mean and Standard Deviation	51
	5.4 Percentage Error	53
	5.5 Uncertainty	
	5.5.1 Calculation of Uncertainty Using Software	54
	5.5.2 Calculation of Uncertainty Using Formula	67
	5.6 Result Analysis	61
CHAPTER 6	CONCLUSION & RECOMMENDATIONS	
	6.1 Conclusion	63
	6.2 Obstacles	64
	6.3 Recommendation	65
REFERENCES		66
APPENDIX A		68
APPENDIX B		79
APPENDIX C		80
APPENDIX D		82

LIST OF TABLES

TABLE NO.	TITLE	PAGE
Table 2.1	Thermocouple types and normal range	9
Table 3.1	Pin assignment references	27
Table 5.1	Result from software after export to Microsoft Excel	49
Table 5.2	Mean and standard deviation result	57
Table 5.3	Percentage of error result	53
Table 5.4	Comparison between two method results	61

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
Figure 1.1	Research flowcharts	6
Figure 2.1	Thermocouple sensor constructions	8
Figure 2.2	Data Acquisition connections	10
Figure 2.3	Heterogeneous computing environments	12
Figure 2.4	Sources of uncertainties grouped by information/energy forms	14
Figure 3.1	Overall system connections	19
Figure 3.2	Basic instrument connections	20
Figure 3.3	HART Communicator Field	21
Figure 3.4	Temperature transmitter	22
Figure 3.5	Ecograph T RSG30	23
Figure 3.6	Temperature Bath	24
Figure 3.7	Data Acquisition	25
Figure 3.8	I/O connector pin assignment	26
Figure 3.9	Single-ended input channel connection	28
Figure 3.10	Differential input channel connection	29
Figure 3.11	Decade box	30
Figure 4.1	Software installation flow chart	33
Figure 4.2	Advantech Device Manager	34
Figure 4.3	Advantech Device Test	35
Figure 4.4	Set the maximum and minimum of measurement	36
Figure 4.5	Data Logging	37
Figure 4.6	Device Control & Preview after complete 1 st reading	37
Figure 4.7	Uncertainty calculator button	38
Figure 4.8	Uncertainty due to repeatability of the experiment calculator	39
Figure 4.9	Combined uncertainty calculator	39

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
Figure 4.10	Calculate the effective degree of freedom	40
Figure 4.11	Find the confidence interval	41
Figure 4.12	Setting Tab	42
Figure 4.13	Selecting Device	43
Figure 4.14	Running software	44
Figure 4.15	1 st reading vs. MSU applied	45
Figure 4.16	Percentage of Error vs. MSU applied	45
Figure 4.17	General Procedure of using software	46
Figure 5.1	Result from software	48
Figure 5.2	1 st Reading Temperature vs. MSU applied (°C)	49
Figure 5.3	2 nd Reading Temperature vs. MSU applied (°C)	50
Figure 5.4	3 rd Reading Temperature vs. MSU applied (°C)	50
Figure 5.5	Percentage of Error vs. MSU applied	53
Figure 5.6	Calculation Uncertainty due to repeatability of the experiment, U_1	55
Figure 5.7	Uncertainty contributions due to MSU error, U_2	55
Figure 5.8	Uncertainty due to UUT resolution/MSU resolution, U_3	56
Figure 5.9	Combined uncertainty, U_c and calculate the confidence limits	56

LIST OF ABBREVIATIONS

DAQ	Data Acquisition
VB	Microsoft visual Basic software
USB	Universal Serial Bus
GUI	Graphical User Interface
HART	Highway Addressable Remote Transducer
PID	Proportional-Integral-Derivative
Ni	Nickel
Pt	Platinum
Ge	Germanium
PV	Process Variable
LRV	Lower Range Value
URV	Upper Range Value
AGND	Analog Ground
AI0	Analog Input
LED	Light Emitting Diode
V	Volts
°C	Celsius
Ma	mili-ampere
IDE	Integrated Development Environment

LIST OF APPENDICES

APPENDIX NO.	TITLE	PAGE
Appendix A	Software coding	68
Appendix B	Student's t-distribution Table	79
Appendix C	Temperature Transmitter	80
Appendix D	Advantech USB-4716 DAQ card specification	82

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Development of computer-based instrumentation system is an important as so far there are no mechanisms incorporated in software offered that allows instrument to be tailed to individual (researchers, industrial engineers) needs. No measurement is ever guaranteed to be perfect. In many cases results of temperature measurements have to be presented together with the uncertainty of these measurements. It concerns mainly the measurements performed by companies or organisations which have introduced quality management systems consistent with requirements of the ISO 9000 and EN 45000 series of standards and the higher numbers of these standards [3]. Uncertainty of measurement is the doubt that exists about the result of any measurement. By quantifying the possible spread of measurements, the confident of the result can be determined. The uncertainty derives from the measuring device and from the skill of the person doing the measuring.

Currently, the standard limit of error for most thermocouples in industrial measurements is 0.75% with special limits of error at 0.4% across the range. Since most competent labs can calibrate with an uncertainty of better than 1°C, the typical temperature measurement can be vastly improved by utilizing information from a custom calibration. This custom characterization is stored in the memory module and utilized by the data conversion system.

Nowadays, the major change occurring at the present is the increasing number of user friendly software that make it possible for user to experience new and fast ways of learning. In minutes, simulation, controller and real world interfacing can be created instantly. In this project, the software is developed to help user to learn and explore the calibration and uncertainty process with an interesting and interactive way in order to reduce the human error. Besides that, the temperature measurement calibration will consume a long process compared to pressure measurement calibration due to measurement repeatability and therefore, it needs the monitoring of the operator until the process is finish. Therefore, the automatic detection temperature measurement from the temperature source using the Type K thermocouple is proposed in this project. The computer software is menu-based to give the user flexibility and ease of use. The user needs no programming experience to operate the systems.

1.2 OBJECTIVE

The objectives of this project are to

- I. To design an automatic detection of temperature measurement in the software from the temperature source port within the range of the setting temperature.
- II. To develop software application to help in student learning process. Visual Basic 2008 Express Edition will be use as main programming language. The software is developed to be interactive and friendly user.
- III. To interface the temperature transmitter output using thermocouple to Visual Basic application. This interfacing between instrument and computer will be done by using data acquisition (DAQ) card.

1.3 SCOPE

This project is to develop software application to help in student learning process. Visual Basic (VB) 2008 Express Edition will be used as a main programming language. The software is developed to be interactive and user friendly for student. The software that will be includes the calculation for uncertainty and confidence limits for temperature measurement.

Temperature transmitter output will be interface to visual basic software by using Data Acquisition process (DAQ). Advantech USB-4716 DAQ card will be used to interface between instrument and computer.

The automatic detection of temperature measurement will be design in the software from the temperature source port within the range of the setting temperature. All the complete set of the temperature readings will be used directly for the calibration and uncertainty calculation process.

1.4 RESEARCH METHODOLOGY

There are several research methodologies that need to follow:

- i. Understand the concept of temperature measurement and indentify the correct method to do the measurement.
- ii. Set up the instrument with several reference instruments such as HART, recorder and transmitter.
- iii. Understand the method on how to communicate between computer and instrument.
- iv. Design and writing the program based on supervisor's opinion.
- v. Interface software to thermocouple via DAQ card.
- vi. Test the software and compare the result with measurement recorder and temperature transmitter

Design step of work methodology can be simplified as shown in figure 1.1.

Flow chart:

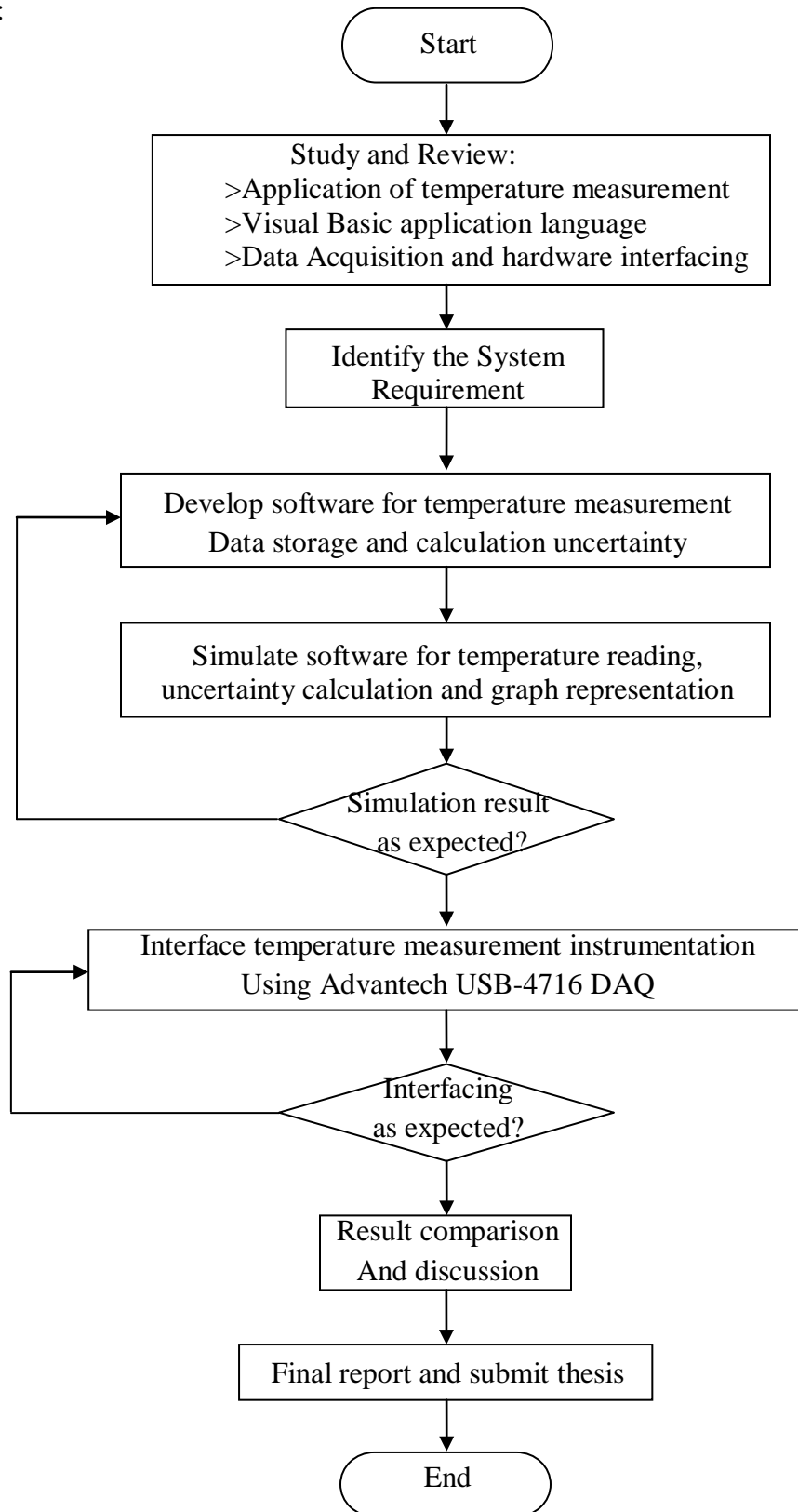


Figure 1.1: Research flowcharts

CHAPTER 2

LITERATURE REVIEW

2.1 LITERATURE REVIEW

For this project, there are some previous researches that are used to be referred to develop an automatic temperature measurement. These are researches and journals are related to this project either directly or indirectly.

2.2 TYPE K THERMOCOUPLE SENSOR

Thermocouples, as in figure 2.1, contain two electrical conductors made of different materials which are connected at one end. The end of the conductors which will be exposed to the process temperature is called the measurement junction. The point at which the thermocouple conductors end which is usually where the conductors connect to the measurement device is called the reference junction.

When the measurement and reference junctions of a thermocouple are at different temperatures, a millivolt potential is formed within the conductors. Knowing the type of thermocouple used, the magnitude of the millivolt potential within the thermocouple, and the temperature of the reference junction allows the user to determine the temperature at the measurement junction.

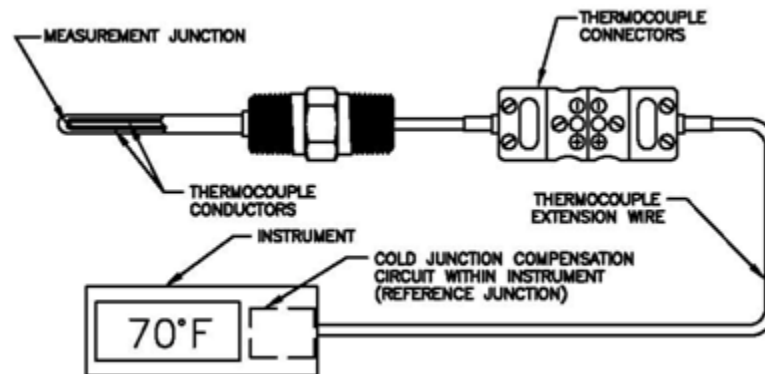


Figure 2.1: Thermocouple sensor constructions

The millivolt potential that is created in the thermocouple conductors differs depending on the materials used. Some materials make better thermocouples than others because the millivolt potentials created by these materials are more repeatable and well established. These thermocouples have been given specific type designations such as Type E, J, K, N, T, B, R and S. The differences between these thermocouple types will be explained in table 2.1.

Table 2.1: Thermocouple types and normal range

Type	Material	Range
E	Chromel/Constantan	0 To 340C
J	Iron/Constantan	-40 to +750 °C
K	Chromel/Alumel	-270 °C to +1372 °C
N	Nicrosil/Nisil	(293 To 1260C
T	Copper/Constantan	-200 to 350 °C
B	Platinum/Platinum-30% Rhodium	800 To 1700C
R	Platinum/Platinum-13% Rhodium	0 To 600C 600 To 1450C
S	Platinum/Platinum-10% Rhodium	0 To 600C 600 To 1450C

The paperwork titled Smart Thermocouple system for Industrial Temperature Measurement by Bill Schuh and Watlow create thermocouple with integral memory, complementary instrumentation and software algorithms. The information data stored in the memory of the sensor allow for enhanced measurements by improving the traceability, calibration uncertainty and robustness. While various features of this smart system have been utilized in other temperature measurement systems these have not taken full advantage of sensor knowledge in conjunction with application to provide an industrial temperature measurement [4].

The other paperwork is Temperature Measurement System Based on Thermocouple with Controlled Temperature Field. This sensor uses known method of rejection of systematic error or stabilization on influence factor. In this case it is proposed to make stabilization of temperature field along electrodes of thermocouple. Including several additional temperature control subsystems provides this stabilization during exploitation. Each such subsystem includes additional thermocouple and heater. These additional thermocouple and heater are shifted along the main axis of main thermocouple.

It provides stabilization of this form during testing and during exploitation independently of changing of temperature field of measurement object [5].

2.3 DATA ACQUISITION

The purpose of data acquisition is to measure an electrical or physical phenomenon such as voltage, current, temperature, pressure, or sound. Figure 2.2 shows data acquisition system typically involves the conversion of analog waveforms into digital values for processing. The components of data acquisition systems include:

- Sensors that convert physical parameters to electrical signals.
- Signal conditioning circuitry to convert sensor signals into a form that can be converted to digital values.
- Analog-to-digital converters, which convert conditioned sensor signals to digital values.
- Data acquisition applications are controlled by software programs developed using various general purpose programming languages such as BASIC, C, FORTRAN, Java, Lisp, and Pascal. COMEDI is an open source API (application program Interface) used by applications to access and controls the data acquisition hardware.

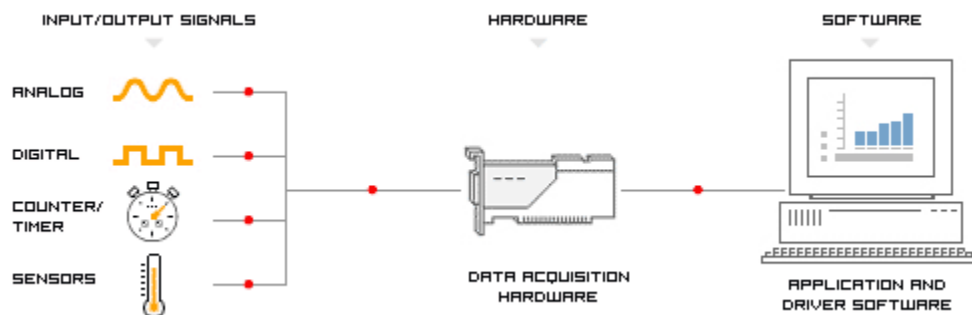


Figure 2.2: Data Acquisition connections

2.4 VISUAL BASIC

Visual basic is a programming language and environment developed by Microsoft. Based on the BASIC language, Visual Basic was one of the first products to provide a graphical programming environment and a paint metaphor for developing user interfaces. Instead of worrying about syntax details, the Visual Basic programmer can add a substantial amount of code simply by dragging and dropping controls, such as buttons and dialog boxes, and then defining their appearance and behavior.

Although not a true object-oriented programming language in the strictest sense, Visual Basic nevertheless has an object-oriented philosophy. It is sometimes called an event-driven language because each object can react to different events such as a mouse click.

Since its launch in 1990, the Visual Basic approach has become the norm for programming languages. Now there are visual environments for many programming languages, including C, C++, Pascal, and Java. Visual Basic is sometimes called a Rapid Application Development (RAD) system because it enables programmers to quickly build prototype applications.

In this journal can describes how supervisory data at PLC level can be collected and stored at Access database, and how these data can be retrieved later for graphic user interface purpose or raw data processing. Basically, the hardware setup is as Figure 2.3, where PC and PLC are connected through RS-232[9, 10, 11]. Visual Basic is used to develop the communication program, where then take advantage of the available mscomm.vbx object provided by Visual Basic to access PC's RS-232 port. With

mscomm.vbx object, program developers can save the otherwise trouble of coding Windows API's to control RS-232 port [12]. The portion of the program is listed in the following to show how the mscomm.vbx control tool is used for PC to request state data from PLC [16]. The overall system was summarized into flow chart in figure 2.3 below.

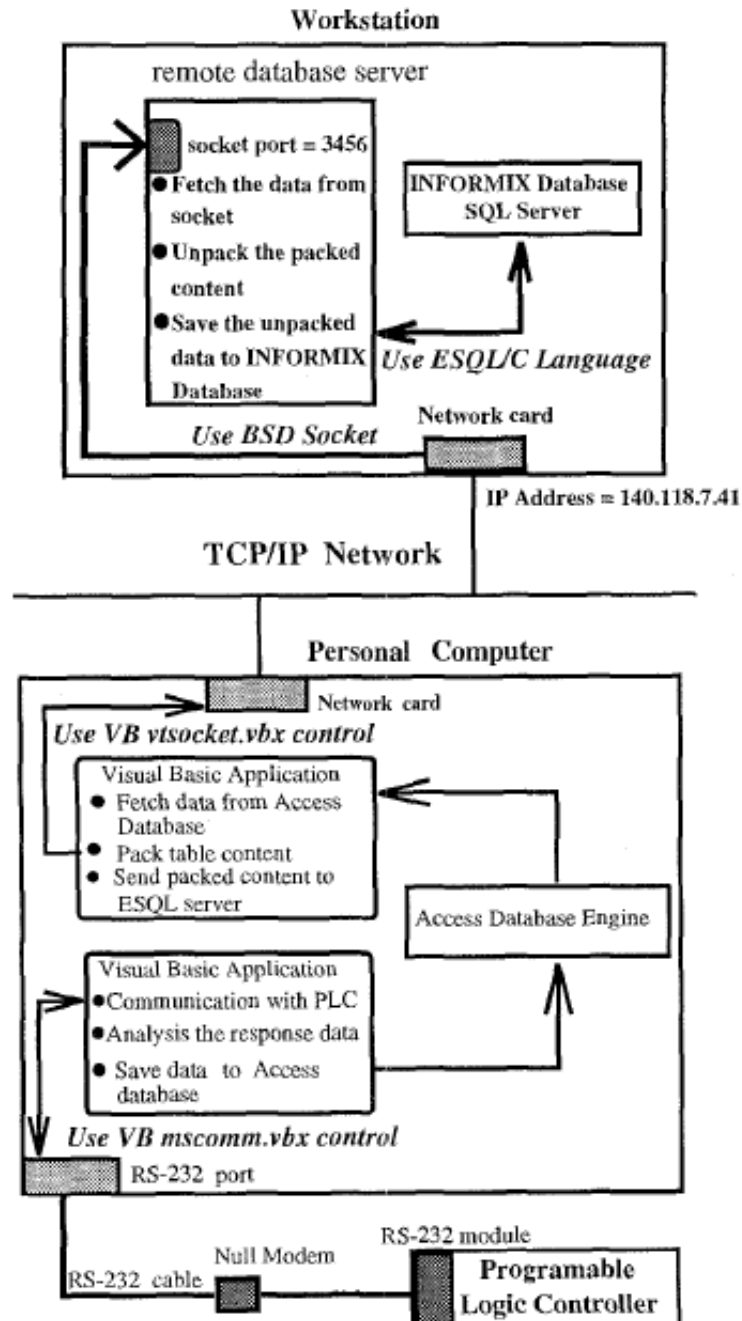


Figure 2.3: Heterogeneous computing environments.

2.5 GRAPHICAL USER INTERFACE (GUI)

A graphical user interface (GUI) is a human-computer interface that uses windows, icons and menus and which can be manipulated by a mouse and limited extent by a keyboard. A major advantage of GUIs is that they make computer operation more intuitive, and thus easier to learn and use. For example, it is much easier for a new user to move a file from one directory to another by dragging its icon with the mouse than by having to remember and type seemingly arcane commands to accomplish the same task.

Adding to this intuitiveness of operation is the fact that GUIs generally provide users with immediate, visual feedback about the effect of each action. For example, when a user deletes an icon representing a file, the icon immediately disappears, confirming that the file has been deleted. This is contrast with the situation for a CLI, in which the user types a delete command but receives no automatic feedback indicating that the file has actually been removed.

In addition, GUIs allow users to take full advantage of the powerful multitasking capabilities of modern operating systems by allowing such multiple programs and/or instances to be displayed simultaneously. The result is a large increase in the flexibility of computer use and a consequent rise in user productivity.

The paperwork of Marian Jerzy titled A Calculation of Uncertainties in Virtual Instrument said that the one of the way to distinguish sources of uncertainties in an application of deliberation developed in literature [7, 8, 9], which concern approaching from energy or information propagation through any system. At a virtual instrument, the system looks upon two boundaries which are intrinsic and extrinsic. A boundary may be

visualized between the Virtual Instrument and the human machine super-system by observer. Impacts of sources of uncertainties from outside the instrument penetrate the extrinsic boundary and cause effects within the system from those outside system. Thus, modeling of virtual instruments using extrinsic and intrinsic boundaries illustrated in figure 2.4 is essentially the same as modeling, for example, and operational amplifier or analogue sensor, which are parts of the very front end elements of virtual instruments [6].

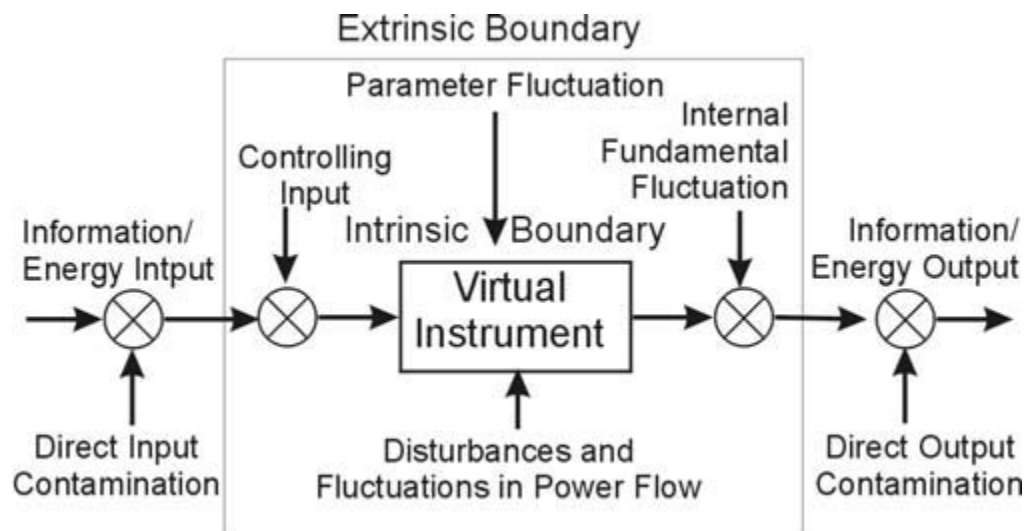


Figure 2.4: Sources of uncertainties grouped by information/energy forms

2.6 CALIBRATION

To calibrate an instrument means to check and adjust (if necessary) its response so the output accurately corresponds to its input throughout a specified range. The purpose of calibration is to ensure the input and output of an instrument corresponds to one another predictably throughout the entire range of operation [1]. For the five-point calibration of the instrument, the span of the UUT is divided into five equal parts with the first point at the low range and the top point at the high range. For example the temperature transmitter has the range between 0 to 20 mA. Therefore, span is $20-0=20$ mA. Dividing the span by four we get 5. Hence, the five equal points are 0, 5, 10, 15, 20 mA.

To measure a quantity without needing calibration, the used basic model must satisfy some conditions with respect to its structure and to the nature of its parameters. A model contains in general nature constants, directly measurable parameters, and unknown parameters. For a calibration free measurement the quantity being measured must be calculable without needing to predetermine any of the other unknown parameters in the model [2].

The journal titled Measuring Temperature Calibration Free with Bipolar Transistor by O. Kanoun done the temperature measurement based on transistor which is considered under the aspect of calibration. The necessity of calibration increases the production and maintenance costs. Therefore, both manufacturers and appliers prefer to use sensors, which dispose of calibration. The method offers the possibility of calibration free temperature measurement. However, this fact could not be used in practice because of the low accuracy reached. Calibration free temperature measurement must guarantee a certain accuracy class to be profitable for some applications with corresponding requirements [1].

The journal from Matthew Harker and Paul O’Leary titled Calibration, Measurement and Error analysis of Optical Temperature Measurement via Laser Induced Fluorescence said that the experiment at hand is cooling of water as cell walls are cooled from 42°C to 5°C, which mimics a die cast model. The physical portion of the device is based on the principle of Laser Induced Fluorescence (LIF); however, this technique must be calibrated, hence a portion of the measurement device is a mathematical model. The interested not only temperature measurement but also its accuracy. The accuracy of the physical measurement also must take account and propagate the uncertainty in estimated parameters throughout the theoretical portion of the measurement device [2].

2.7 STANDARD DEVIATION

The standard deviation is a measure of the dispersion of a set of values. It can apply to a provability distribution, a random variable, a population or a multiset. The standard deviation is usually denoted with the letter σ (lowercase sigma). It is defined as the root-mean-square (RMS) deviation of the values from their mean, or as the square root of the variance. $x_i - \bar{x}$.

The calculation is described by the following formula:

$$\sigma = \sqrt{\sum_{k=0}^{\infty} \frac{1}{N} (x_i - \bar{x})^2} \text{-----} (2.1)$$

Where the mean of x' is defined as:

$$\bar{x} = \sum_{i=0}^n \frac{1}{n} (x_i) \text{-----} (2.2)$$

From the journal Standard Deviation Method for Determining the Weights of Group Multiple Attribute Decision Making under Uncertain Linguistic Environment by Yejun Xu and Zhijian Cai, The standard deviation method is proposed by Wang [17] to deal with MADM problems with numerical information. Xu and Da [18] also use this method to deal with the uncertain multiple attribute decision making problems, in which the information about the attribute weights is unknown completely and the attributes values are in the forms of interval numbers. Its main ideal is as follows. For the MADM problems, the collective preference values is compare to rank the alternatives, the larger the ranking value $z_i(w)$, the better the corresponding alternative x_i is. If the performance values of each alternative have little differences under an attribute, it shows that such an attribute plays a small important role in the priority procedure.

Contrariwise, if some attribute makes the performance values among all the alternatives have obvious differences, such an attribute plays an important role in choosing the best alternative. So, to the view of sorting the alternatives, if one attribute has similar attribute values across alternatives, it should be assigned a small weight; otherwise, the attribute which makes larger deviations should be evaluated a bigger weight, in spite of the degree of its own importance. Especially, if all available alternatives score about equally with respect to a given attribute, then such an attribute will be judged unimportant by most experts. In other word, such an attribute should be assigned a very small weight. Wang [17] suggests that zero should be assigned to the attribute of this kind. The difference of attribute values can be measured using standard deviation [11].

CHAPTER 3

INSTRUMENT AND HARDWARE

3.1 OVERALL SYSTEM CONNECTION

From figures 3.1, there are four major parts for this project which is thermocouple sensor, temperature transmitter and data acquisition. Type k thermocouple is a sensor to detect the temperature change and transmit signals in a voltage value. Temperature transmitter which is the instrument that needs to calibrate automatically by software. Temperature transmits milivolt signal from thermocouple and then convert to miliamp signal after through the temperature transmitter. Data Acquisition is use to convert analog signal to digital signal so that can be read by computer software. All the temperature measurement system is control by computer software. Visual basic 2008 is use to captured, stored and analysis data from sensor.

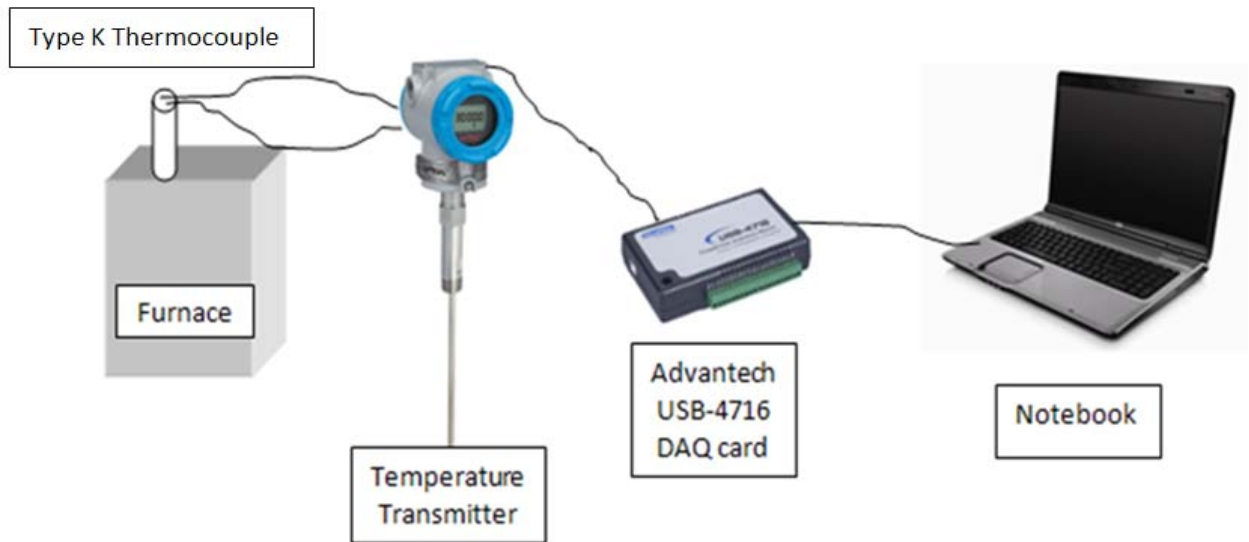


Figure 3.1: Overall system connections

3.1.1 BASIC INSTRUMENT CONNECTIONS

From the Figure 3.2 and 3.3, the instrument connection includes Emerson 375 HART Field Communicator and Endress+Hauser Ecograph T RSG30 as reference temperature for this project. Type k thermocouple measured temperature heat from Isotech Jupiter 650B temperature bath. Yokogawa temperature transmitter YT110 functioning to convert signal from thermocouple to current value so that DAQ can read the data and convert it into digital signal. Since the Advantech USB-4716 DAQ card only receives data in voltage, 250 Ω resistances has been added to convert current signal to voltage signal. 250 Ω resistances can be taken either from decade box or 250 Ω resistors.

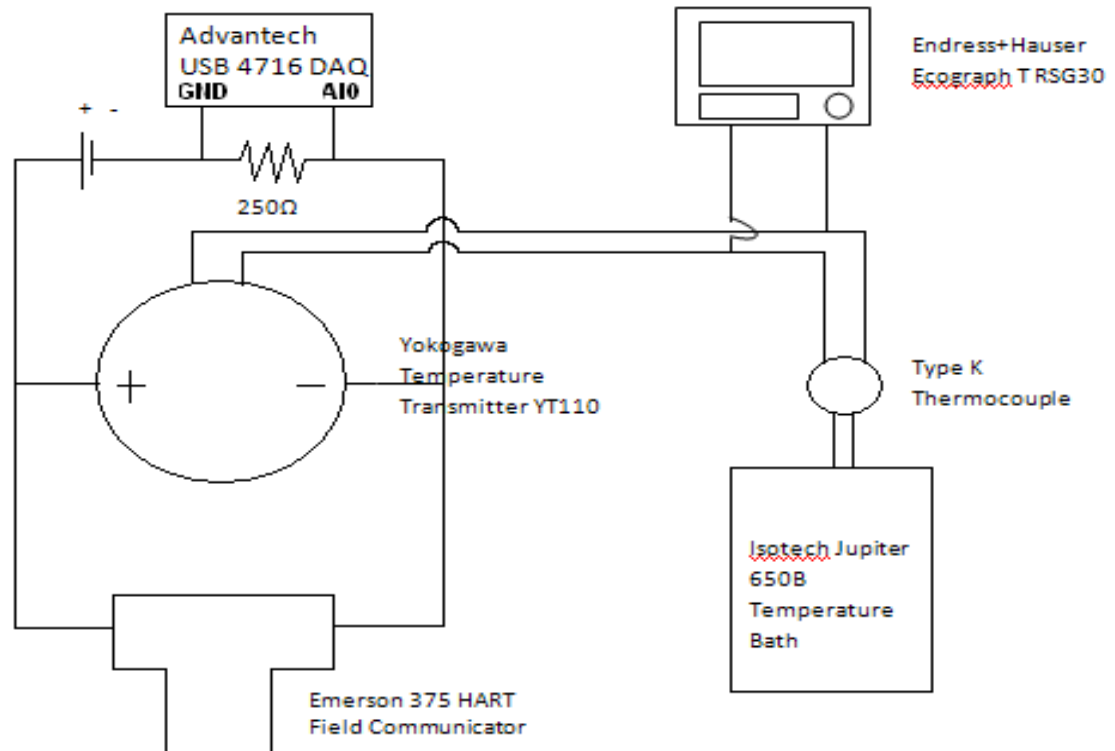


Figure 3.2: Basic instrument connections

3.2 INSTRUMENT

3.2.1 Thermocouple

Thermocouple is a transducer use to convert temperature value to voltage signal. In this project, type K thermocouple as figure 3.4 has been use to detect temperature from the temperature bath. This thermocouple connected to the both temperature transmitter and recorder in parallel.

Sensor type : K (Chromel-Alumel)

Lower Range Value : -270 °C (-454°F)

Upper Range Value : 1372 °C (2501.6°F)

Minimum range : 50 °C

Accuracy : ± 0.45 °C

3.2.2 HART 375 field Communicator

HART 375 field Communicator as figure 3.3 is functioning to calibrate the temperature transmitter. By set up the type of sensor and range of measurement, it will automatically convert signal from sensor to current value. HART also can be reference point for this project since it display very accurate measurement compared to temperature transmitter measurement.



Figure 3.3: HART Communicator Field

3.2.3 Yokogawa Temperature Transmitter YT110

Temperature transmitter model YT110 as figure 3.4 converts thermocouple and RTD's signal into current 4 to 20 mA signal. Signal from thermocouple is too low for DAQ to receive. By convert thermocouple signal to current signal, its boost milivolt signal so then DAQ can read the signal before its convert it to digital signal.



Figure 3.4: Temperature transmitter

3.2.4 Endress+Hauser Ecograph T RSG30

Ecograph has been using widely in processes and industries such as a quality and quantity monitoring in the water and wastewater industry, monitoring of processes power station, food and dairy industry processes, displaying and recording critical parameters in production cycles, tank and level monitoring, temperature monitoring and metal working, and cold storage and transportation monitoring. In this project, ecograph model T RSG30 as figure 3.5, is used for reference point to the software measurement.

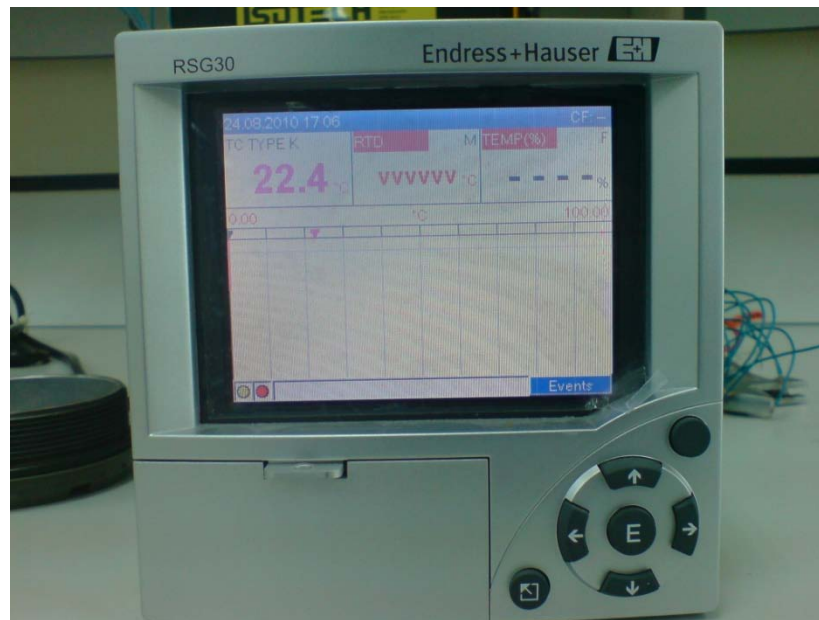


Figure 3.5: Ecograph T RSG30

3.2.5 Isotech Jupiter 650B Temperature Bath

Isotech Jupiter model 650B as figure 3.6 is a temperature bath provides an isothermal enclosure (metal block) in which allows thermometers and thermostats be checked against the temperature indicated on the temperature controller. In this project, Isotech Jupiter is used as temperature source to the sensor. By setting the temperature on the display panel, the temperature will increase until to a temperature required.



Figure 3.6: Temperature Bath

3.2.6 Advantech USB-4716 DAQ Card

Advantech USB-4716 offers a rich set of DLL drivers, third-party driver support and application software on the companion CD-ROM to help fully exploit the functions of your device. Advantech's Device Drivers feature a complete I/O function library to help boost your application performance and work seamlessly with development tools such as Visual Basic. USB-4716 is equipped with plug-in screw-terminal connectors that facilitate connection to the module without terminal boards or cables. Figure 3.7 on next page, in figure 3.8 and table 3.1 shows the pin assignments for the five 10-pin I/O connectors on USB-4716.

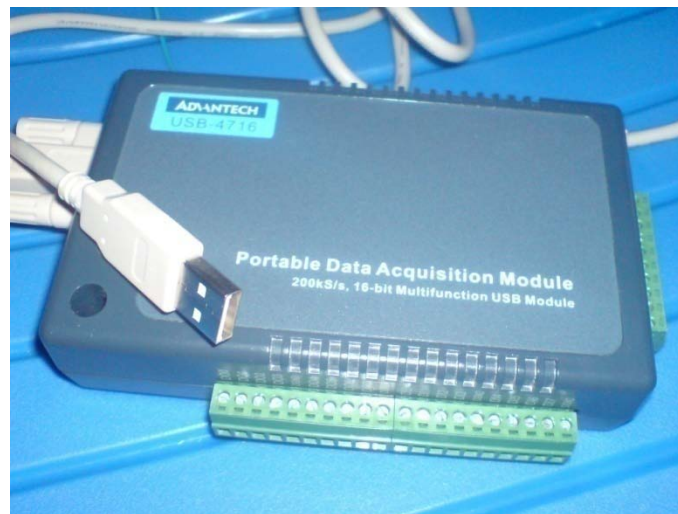


Figure 3.7: Data Acquisition

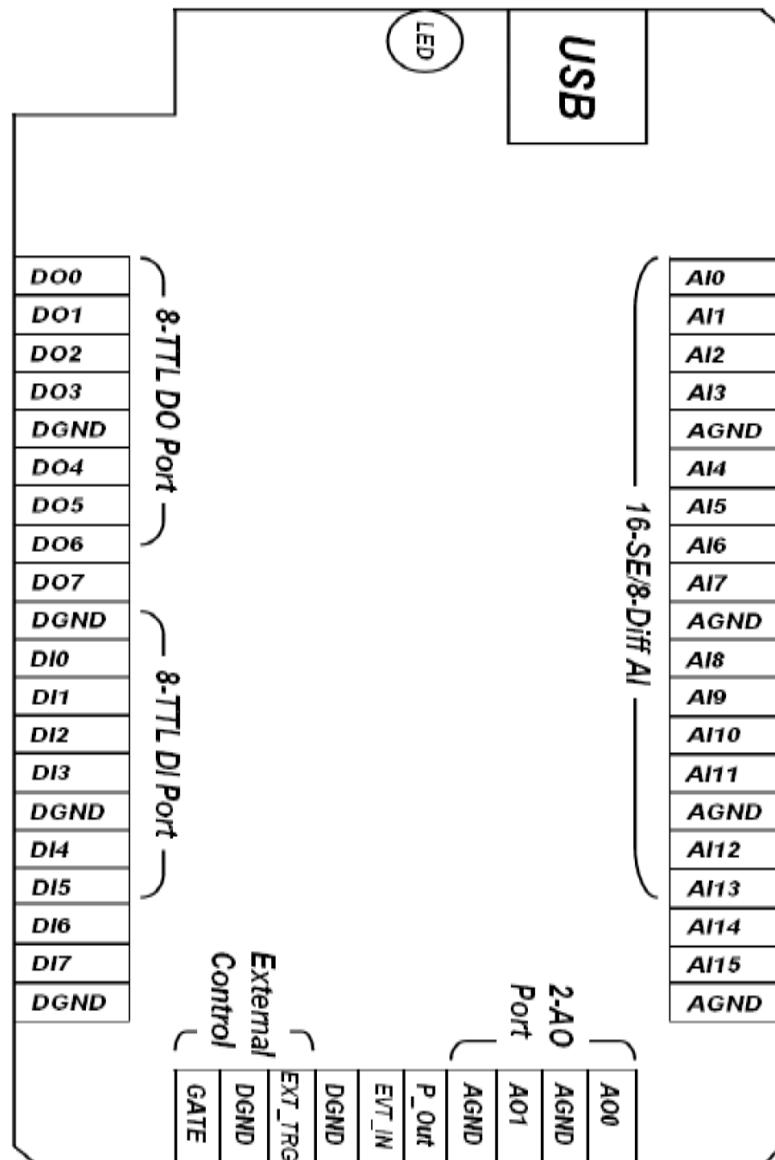


Figure 3.8: I/O connector pin assignment

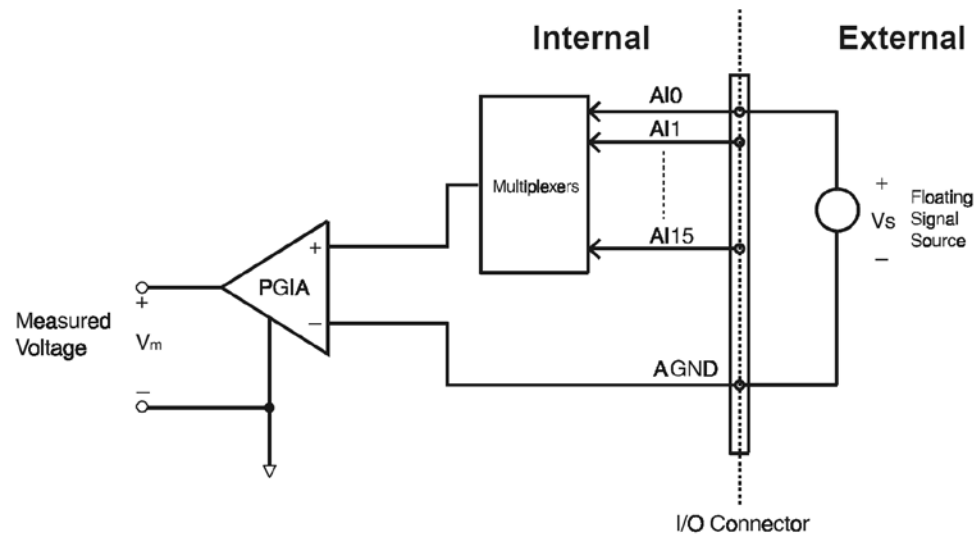
Table 3.1: Pin assignment references

Signal Name	Reference	Direction	Description
AI <0...15>	AGND	Input	Analog Input Channels 0 through 15
AIGND	-	-	Analog Input Ground
AO0	AGND	Output	Analog Output Channels 0/1
AO1			
AOGND	-	-	Analog Output Ground. The analog output voltages are referenced to these nodes
DI<0...7>	DGND	Input	Digital Input Channels
DO<0...7>	DGND	Output	Digital Output Channels
DGND	-	-	Digital Ground. This pin supplies reference for the digital channels at the I/O connector
GATE	DGND	Input	A/D External Trigger Gate. When GATE is connected to +5V, it will disable the external trigger signal to input.
EXT_TRG	DGND	Input	A/D External Trigger. This pin is external trigger signal input for the A/D conversion. A low to high edge triggers A/D conversion to start
EVT_IN	DGND	Input	External events input channel.
P_OUT	DGND	Output	Pulse output Channel

Analog Input Connections:

Single-ended Channel Connections

The single-ended input configuration has only one signal wire for each channel, and the measured voltage (V_m) is the voltage of the wire as referenced against the common ground. A signal source without a local ground is also called a “floating source”. It is fairly simple to connect a single-ended channel to a floating signal source. In this mode, USB-4716 provides a reference ground for external floating signal sources. Figure 3.9 shows a single-ended channel connection between a floating signal source and an input channel on USB-4716.



. Figure 3.9: Single-ended input channel connection

Differential Input Connections

The differential input channels operate with two signal wires for each channel, and the voltage difference between both signal wires is measured. On USB-4716, when all channels are configured to differential input, up to 8 analog channels are available. If one side of the signal source is connected to a local ground, the signal source is ground-referenced. Therefore, the ground of the signal source and the ground of the card will not be exactly of the same voltage. The difference between the ground voltages forms a common mode voltage (V_{cm}). To avoid the ground loop noise effect caused by common-mode voltages, users can connect the signal ground to the *Low* input. Figure 3.12 shows a differential channel connection between a grounded-reference signal source and an input channel on USB-4716. With this connection, the PGIA rejects a common-mode voltage V_{cm} between the signal source and USB-4716 ground, shown as V_{cm} in Figure 3.10

Note: In differential input mode, the input channel n should be used with channel $n+1$. ($n=0, 2, 4\dots14$)

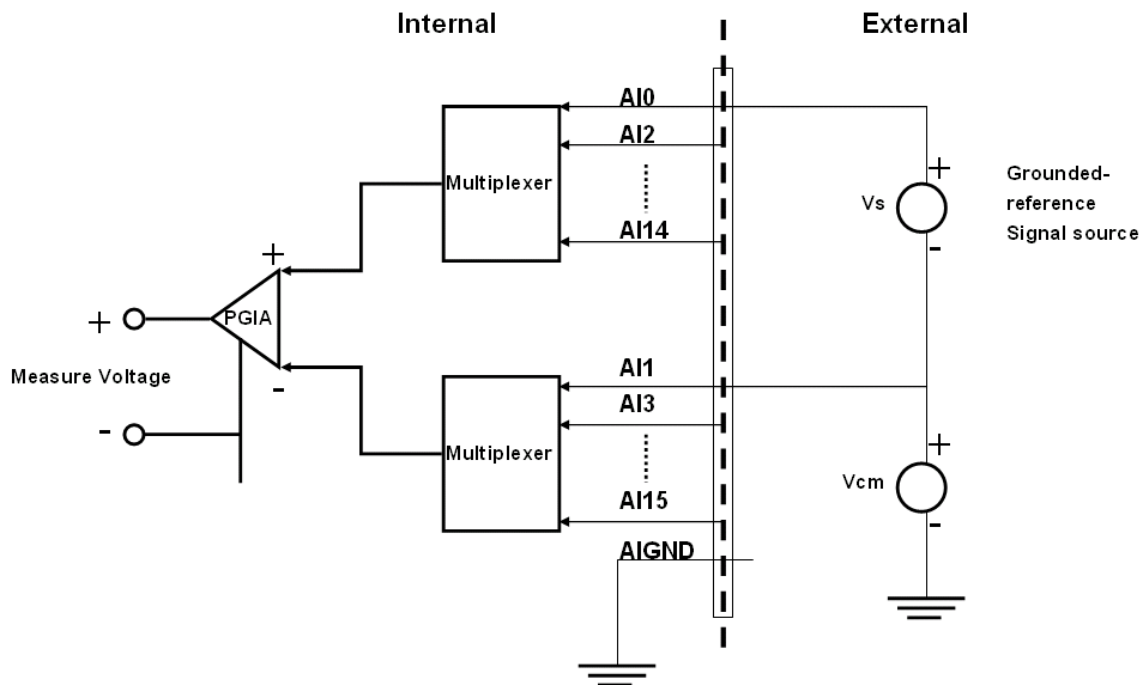


Figure 3.10: Differential input channel connection

3.6.7 Decade Box

In figure 3.11 is Decade box which is used to convert 4 to 20mA signal from temperature transmitter to 1 to 5V signal because Advantech USB-4716 DAQ only receive signal in voltage. Decade Box also use in HART protocol to ensuring that a field device has sufficient voltage to operate. Besides using decade box, user also may use 250 Ω resistance which is in compact version and space saving.



Figure 3.11: Decade box

CHAPTER 4

SOFTWARE

4.1 SOFTWARE DEVELOPMENT

In this project, the software is developed using Microsoft Visual Basic 2008 Express Edition. The software is connected to hardware using Advantech USB-4716 Data Acquisition which is can receive analog input from temperature transmitter. In this software also contain Data Record, Uncertainty application, Graph result and Device configuration application.

4.1.1 Driver Installation

There are several steps while installing the driver for this project. The main driver that used in this project is driver from Advantech USB-4716 DAQ driver supplied by manufacturer. Here are the steps:

1. Insert Advantech Disc Driver.
2. Select “Installation button and then install “Advantech Device Manager”.
3. After finish the installation, click “Individual Driver, select “USB” and then install “USB-4716”.
4. After that, back to the main menu and then click “Advance Option”. Select “Active DAQ Pro to install the interfacing file.

4.1.2 General Procedure on Installing Driver

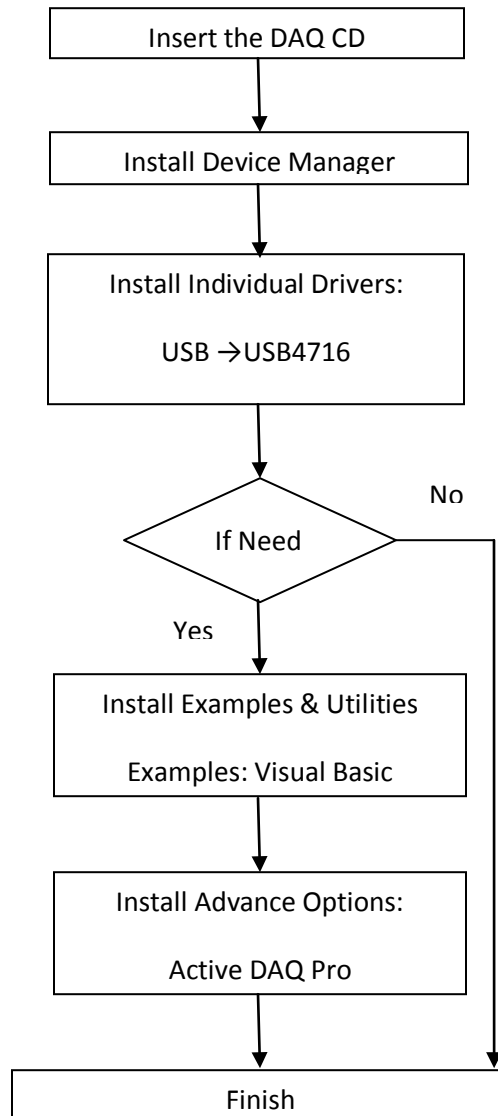


Figure 4.1: Software installation flow chart

After finish all installation, test the DAQ hardware to make sure the hardware are ready to use by open the Advantech Device Manager in the program menu, as figure 4.2 below:

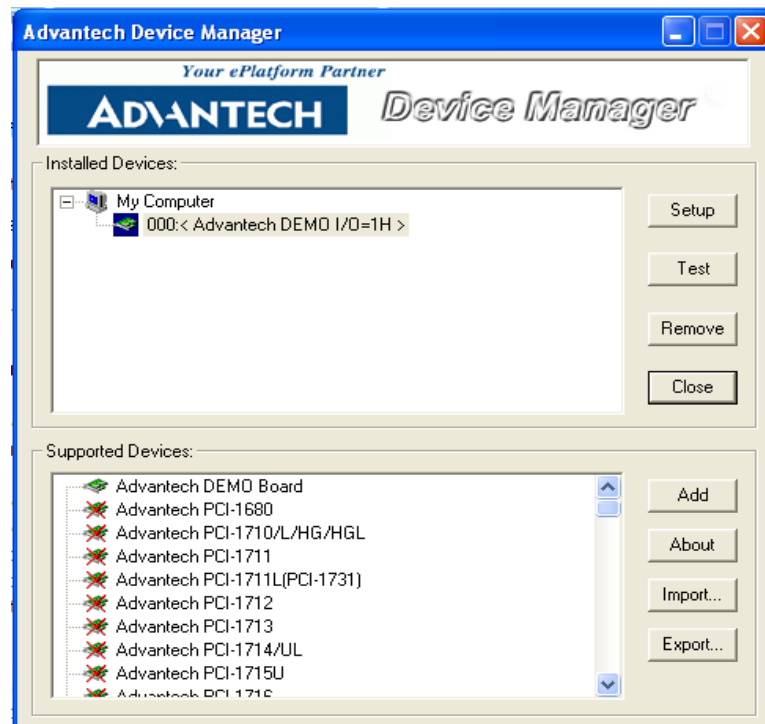


Figure 4.2: Advantech Device Manager

For the first using it, select device that connect to the PC. As example, in this project using USB-7416. So, select Advantech USB-7416 from Supported Devices combo box and then click button "Add". The device will be seen in the Installed Device box. Then, click button "Test" and display below will appear as figure 4.3:

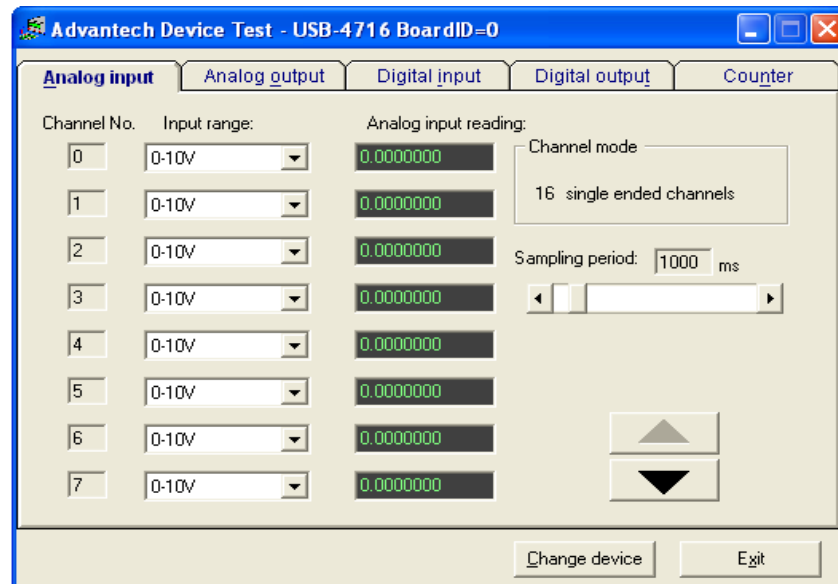


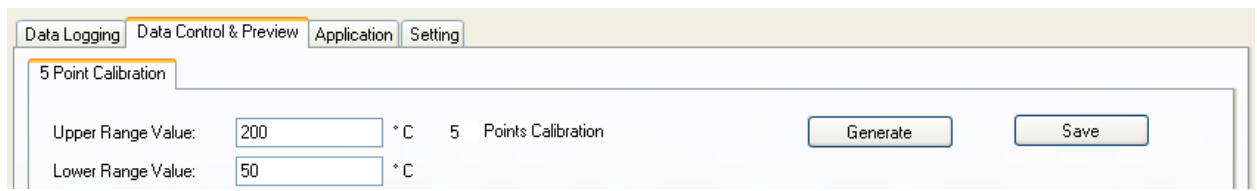
Figure 4.3: Advantech Device Test

By supplying certain voltage to the channel AI0 to AI7, the analog input reading will display the voltage in the input range. If the voltage out of range, it will display the maximum value of voltage only. As an instrument safety procedure, do not supply voltage more than the maximum range to avoid damage to DAQ.

4.2 CREATING GRAPHICAL USER INTERFAC (GUI)

The main GUI in this project is the recording of the data. Data from temperature transmitter will transmit to the DAQ and then DAQ read that data before it can be read by computer.

Based on the figure 4.4 below, firstly, user need to make sure that the maximum and minimum temperature those need to calibrate by this software is correct in the 'Device Control & Preview' tab. Then, select the reading and then click the Start button on the 'Data Logging' tab as figure 4.5 and the software will automatically capture data to the table on 'Device Control & Preview' tab. After first reading complete, the measurement will automatically stop the data recording as figure 4.6. Then, select the other reading in the 'Data logging and click Start button again. The step is also same for the third reading. After third reading, the software will automatically calculate the mean value of the measurement. Then click 'STD Dev' and 'Error' button to calculate standard deviation and Error of the Output.



The screenshot shows a software window with four tabs: 'Data Logging', 'Data Control & Preview', 'Application', and 'Setting'. The 'Data Control & Preview' tab is active, and within it, the '5 Point Calibration' sub-tab is selected. The interface contains two input fields: 'Upper Range Value:' with the value '200' and a unit of '° C', and 'Lower Range Value:' with the value '50' and a unit of '° C'. To the right of these fields, the text '5 Points Calibration' is displayed. Further right, there are two buttons: 'Generate' and 'Save'.

Figure 4.4: Set the maximum and minimum of measurement

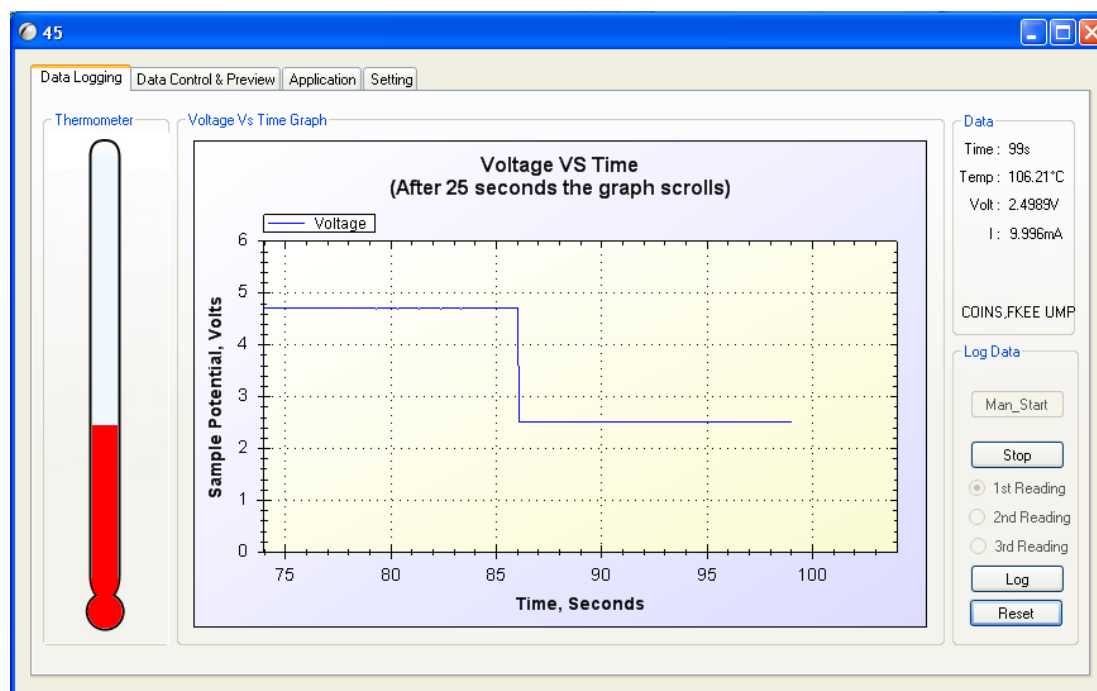
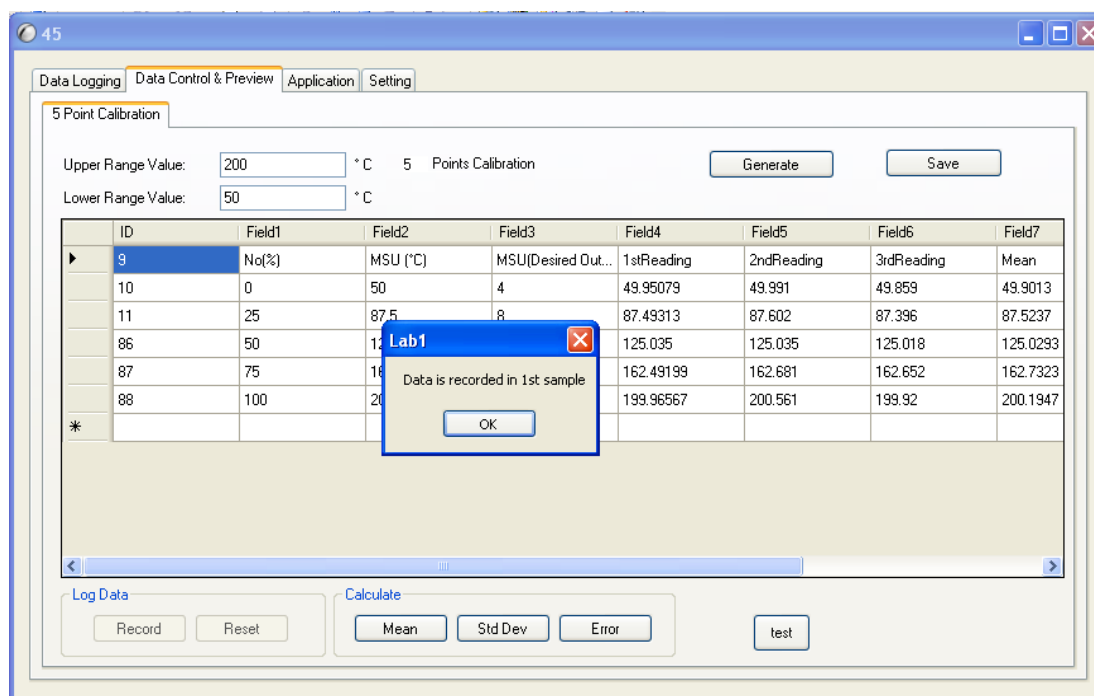


Figure 4.5: Data Logging

Figure 4.6: Device Control & Preview after complete 1st reading

4.2.1 Uncertainty Calculation

To calculate the uncertainty of measurement, click the 'Uncertainty' button on the 'Application' tab as figure 4.7. Then, uncertainty calculator appears in different form as figure 4.8. The first calculator, which is U1 is use to calculate the uncertainty due to repeatability of measurement. Click the calculate button and the value of uncertainty will appear.

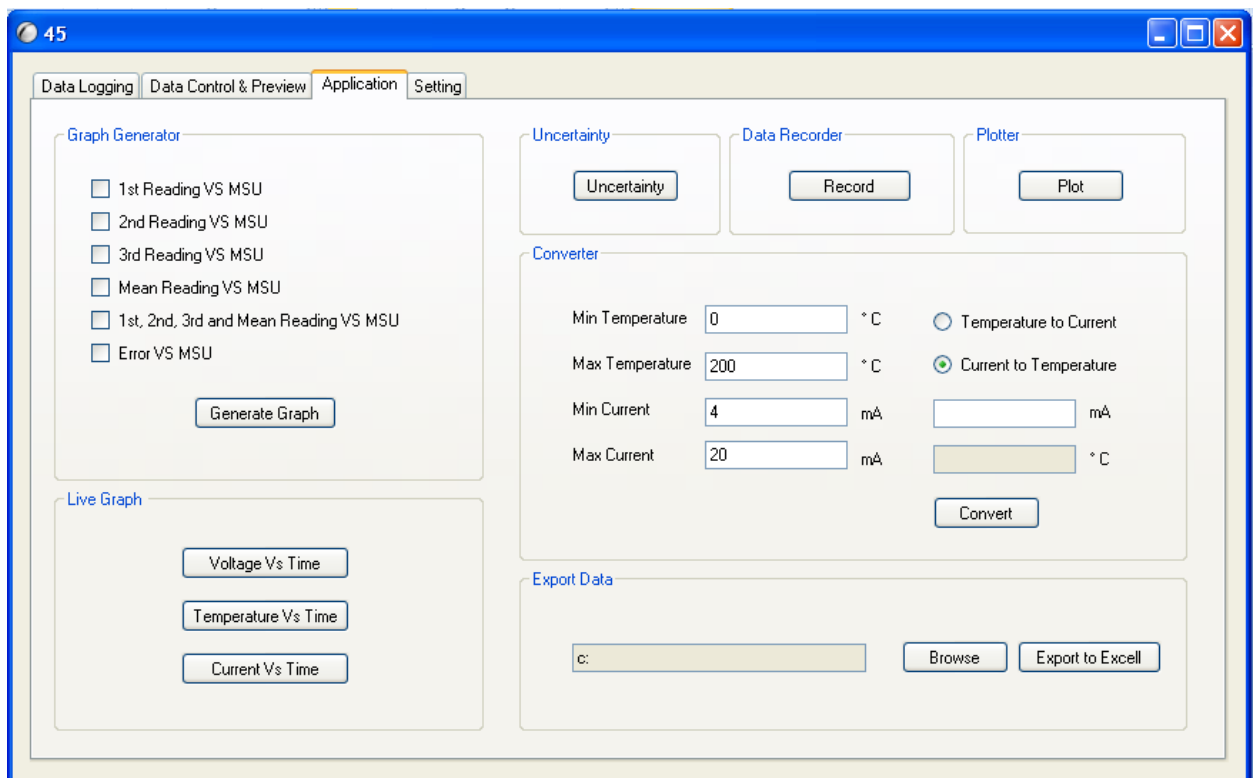


Figure 4.7: Uncertainty calculator button

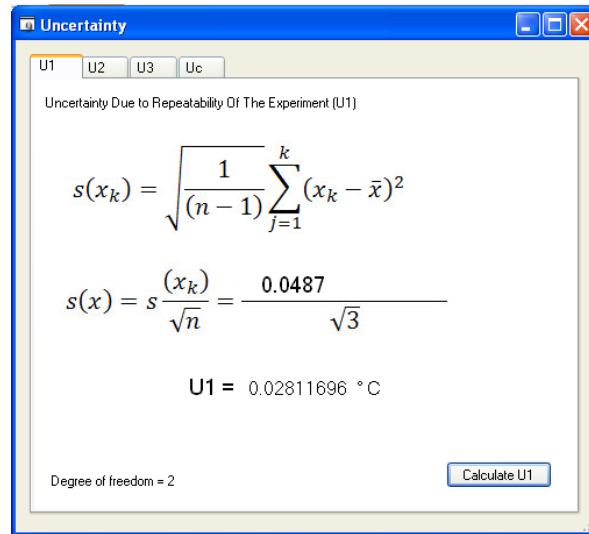


Figure 4.8: Uncertainty due to repeatability of the experiment calculator

Proceed to the next tab which is U2 and U3. The last tab includes the calculation the combination of uncertainty, effective degree of freedom and the confidence limit. From figure 4.9, when user click button Calculate Uc, message box will appear and give the value of Uc.

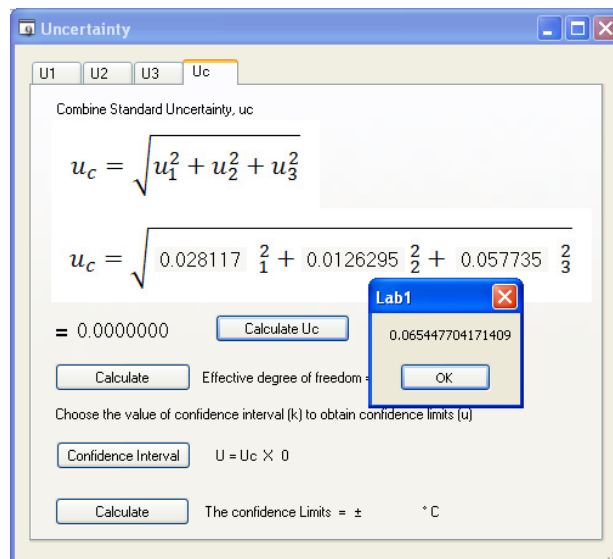


Figure 4.9: Combined uncertainty calculator

After that, click Calculate to find the value of effective degree of freedom as figure 4.10. The value will use to find the confidence interval in the figure 4.11 below after clicking Confidence Interval button. From figure 4.10, the value of effective degree of freedom is 58.7199. So, the best confidence interval user can select is 2.68 which is nearest to the value degree of freedom.

The screenshot shows a software window titled "Uncertainty" with four tabs: U1, U2, U3, and Uc. The Uc tab is active. The window displays the formula for combining standard uncertainty:

$$u_c = \sqrt{u_1^2 + u_2^2 + u_3^2}$$

Below the formula, the values are entered into a calculator-like interface:

$$u_c = \sqrt{0.028117 \frac{2}{1} + 0.0126295 \frac{2}{2} + 0.057735 \frac{2}{3}}$$

The result is shown as:

$$= 0.0654477$$

A "Calculate Uc" button is present. Below this, a "Calculate" button is shown next to the text "Effective degree of freedom = 58.71299".

The next step is to "Choose the value of confidence interval (k) to obtain confidence limits (u)". A "Confidence Interval" button is shown next to the text "U = Uc × 2.68".

Finally, a "Calculate" button is shown next to the text "The confidence Limits = ± 0.1754 °C".

Figure 4.10: Calculate the effective degree of freedom

Degree of Freedom	68.27*	90.00	95.00	95.45	99.00	99.73*
16	1.03	1.75	2.12	2.17	2.92	3.54
17	1.03	1.74	2.11	2.16	2.90	3.51
18	1.03	1.73	2.10	2.15	2.88	3.48
19	1.03	1.73	2.09	2.14	2.86	3.46
20	1.03	1.72	2.09	2.13	2.85	3.42
25	1.02	1.71	2.06	2.11	2.79	3.33
30	1.02	1.70	2.04	2.09	2.75	3.27
35	1.01	1.70	2.03	2.07	2.72	3.23
40	1.01	1.68	2.02	2.06	2.70	3.20
45	1.01	1.68	2.01	2.06	2.69	3.18
50	1.01	1.68	2.01	2.05	2.68	3.16
100	1.005	1.660	1.984	2.025	2.626	3.077
	1.000	1.645	1.960	2.000	2.576	3.000

Figure 4.11: Find the confidence interval

4.3 CONNECTING USB-4716 TO COMPUTER

In this project, computer will connect to hardware via USB-4716 DAQ card. After connect the analog signal to AI0, and then connect DAQ to the USB port via cable provided by manufacturer. Then go to the software and select 'Setting' tab to connect the software to hardware as figure 4.12:

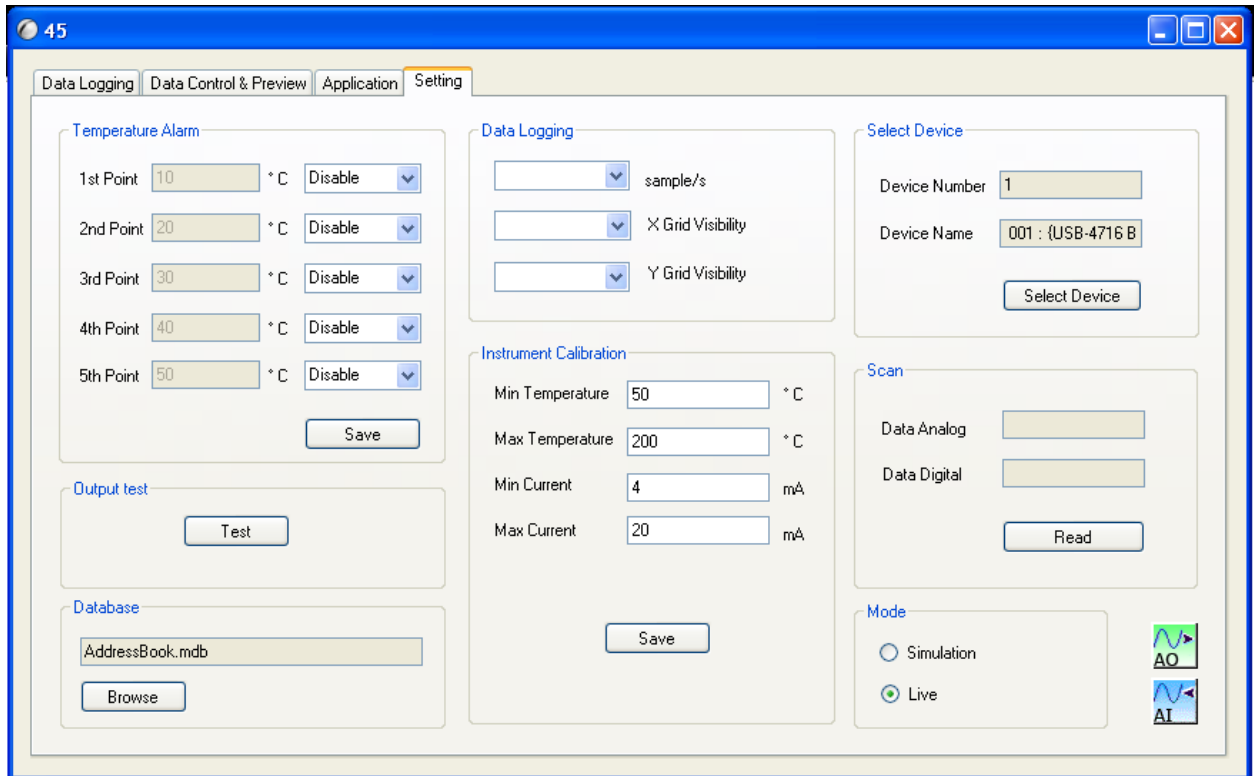


Figure 4.12: Setting tab

The click the 'Select Device' button as figure 4.13 is to select which device need to used. In this project, 001 [USB-4716 BoardID=0] is used to running the software. Then, make sure mode for this measurement is Live before start running this software.

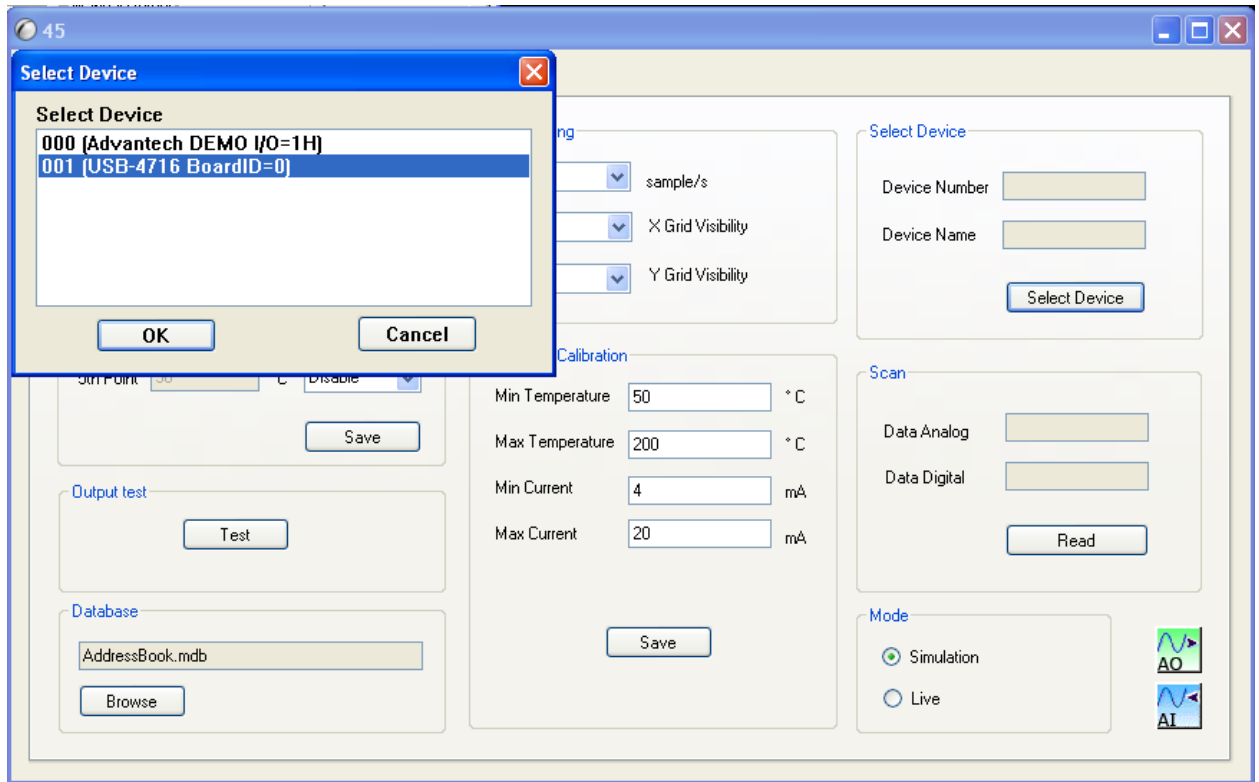


Figure 4.13: Selecting Device

To run this software, click the button 'Start' as figure 4.14 below. User can see the data view as temperature, voltage and current in the Data group box. The live graph show in figure below will increase and decrease according to the voltage that supply to the DAQ card.

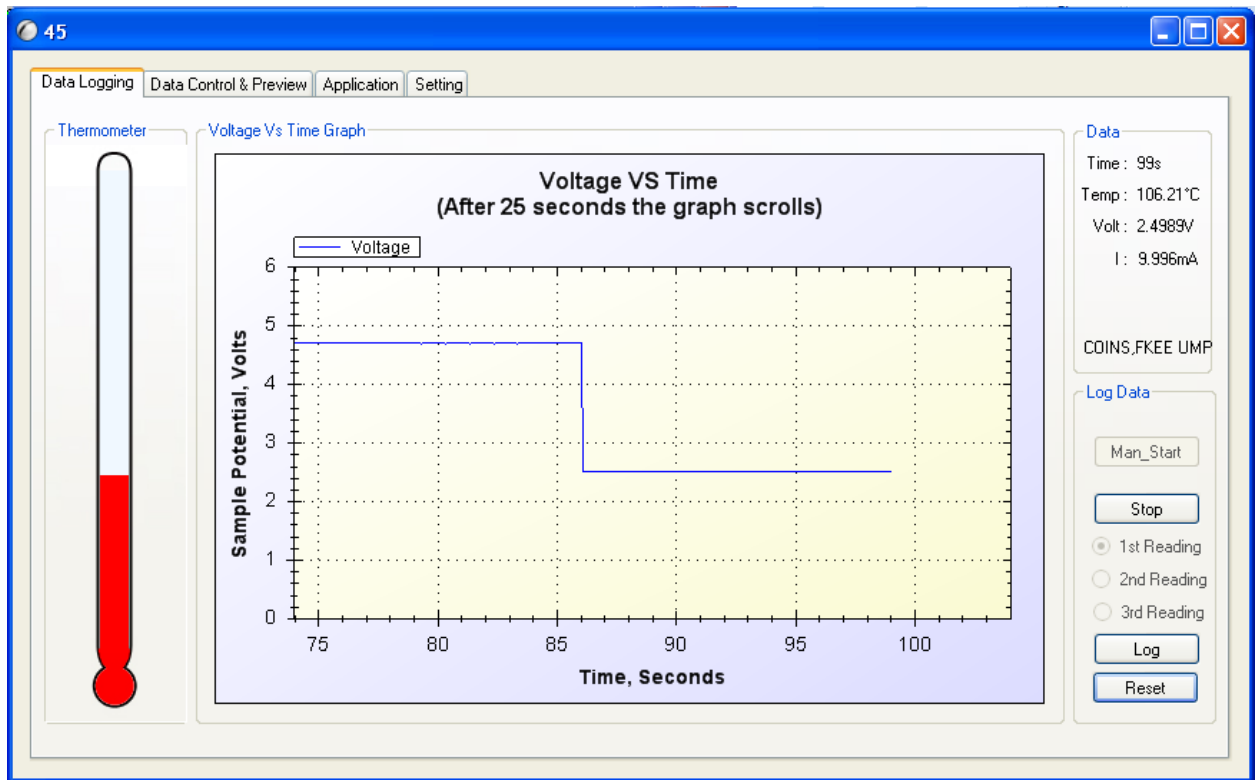


Figure 4.14: Running software

From the result of data record in figure 4.6, results of first reading and percentage of error graph are as shown as figure 4.15 and 4.16:

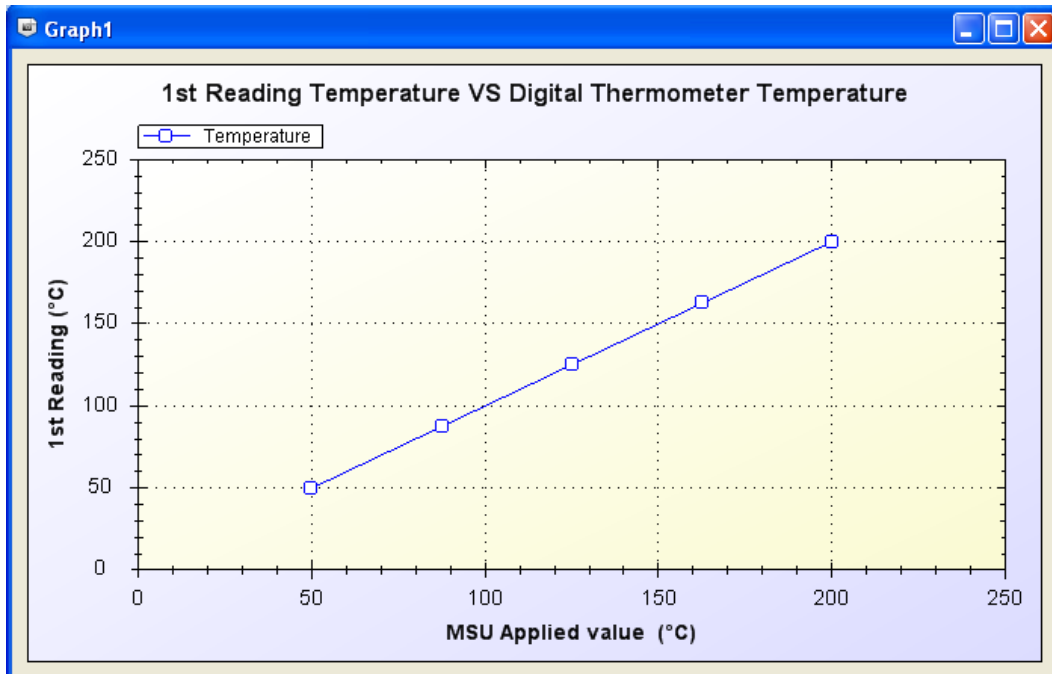
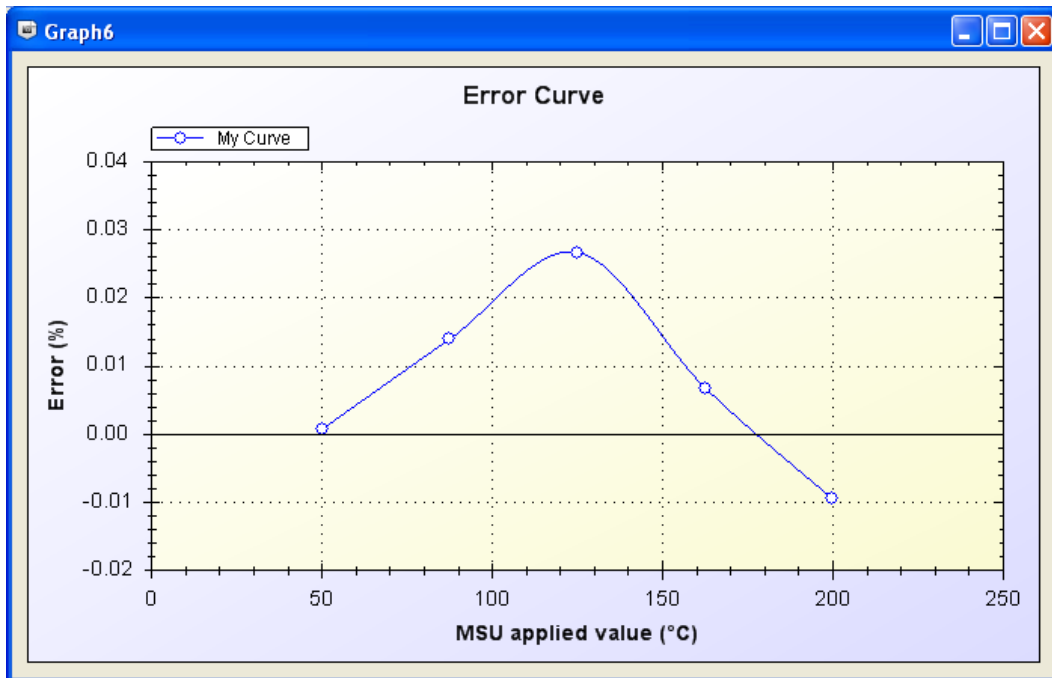
Figure 4.15: 1st reading vs. MSU applied

Figure 4.16: Percentage of Error vs. MSU applied

4.4 GENERAL PROCEDURE ON USING THE SOFTWARE

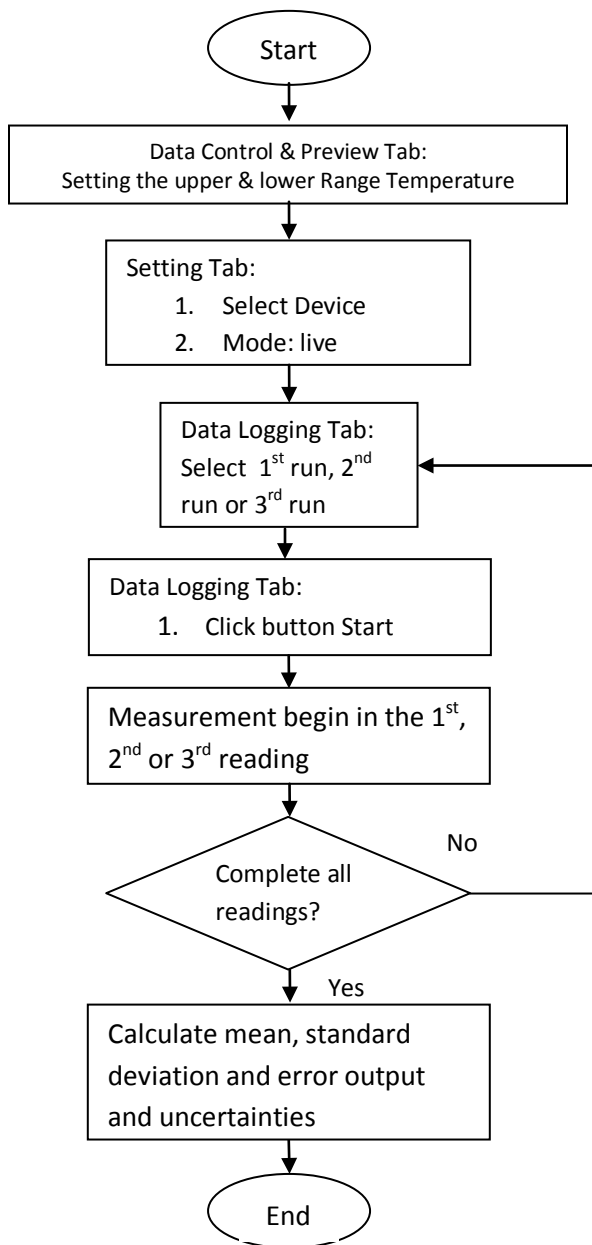


Figure 4.17: General Procedure of using software

CHAPTER 5

RESULT AND ANALYSIS

5.1 INTRODUCTION

The main objective of this project is to develop an automatic temperature measurement application for industry since temperature measurement consumed long process compared to pressure measurement. The software has successfully developed using Microsoft Visual Basic 2008 Express Edition with friendly-user GUI which is simple to operate.

5.2 RESULT OF THE EXPERIMENT

For the five-point calibration of the instrument, the span of the UUT is divided into five equal parts with the first point at the low range and the top point at the high range. For example the temperature transmitter has the range between 50 to 200°C. Therefore, span is $200-50=150^{\circ}\text{C}$. Dividing the span by four we get 37.5°C . Hence, the

five equal points are 50, 87.5, 125, 162.5 and 200 °C. The desired output for 4 to 20mA range is based on temperature range; 50 to 200°C, calculated using equation below:

$$\text{Desired Output} = \frac{x}{100} (\text{URV} \times \text{LRV}) + \text{LRV} \text{-----} (5.1)$$

Where;

- x = measurement point
 URV = Upper Range Value temperature
 LRV = Lower Range Value for temperature

Result:

From the software data and calculation, the result as figure 5.1 was export to Microsoft Excel as in Table 5.1:

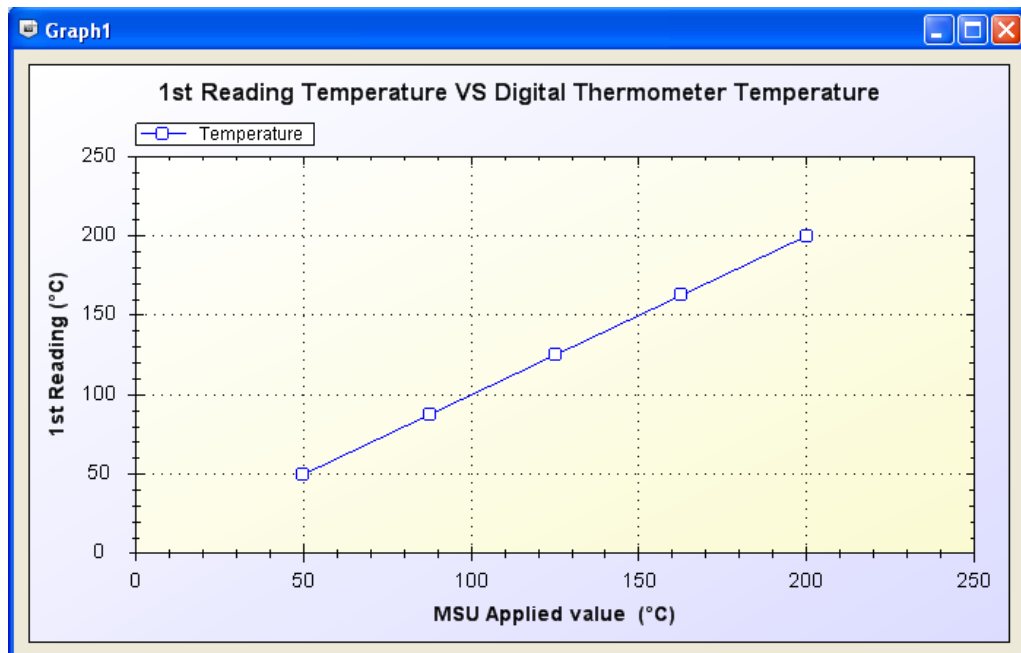
ID	Field1	Field2	Field3	Field4	Field5	Field6	Field7	Field8	Field9
9	No(%)	MSU (°C)	MSU(Desired Output mA)	1stReading	2ndReading	3rdReading	Mean	Std	Error
10	0	50	4	49.95079	50.00229	50.04807	50.0004	0.0487	0.0008
11	25	87.5	8	87.49313	87.51602	87.52747	87.5122	0.0175	0.014
86	50	125	12	125.035	125.02975	125.03548	125.0334	0.0032	0.0267
87	75	162.5	16	162.49199	162.52632	162.51488	162.5111	0.0175	0.0068
88	100	200	20	199.96567	199.97711	200	199.9809	0.0175	-0.0095
*									

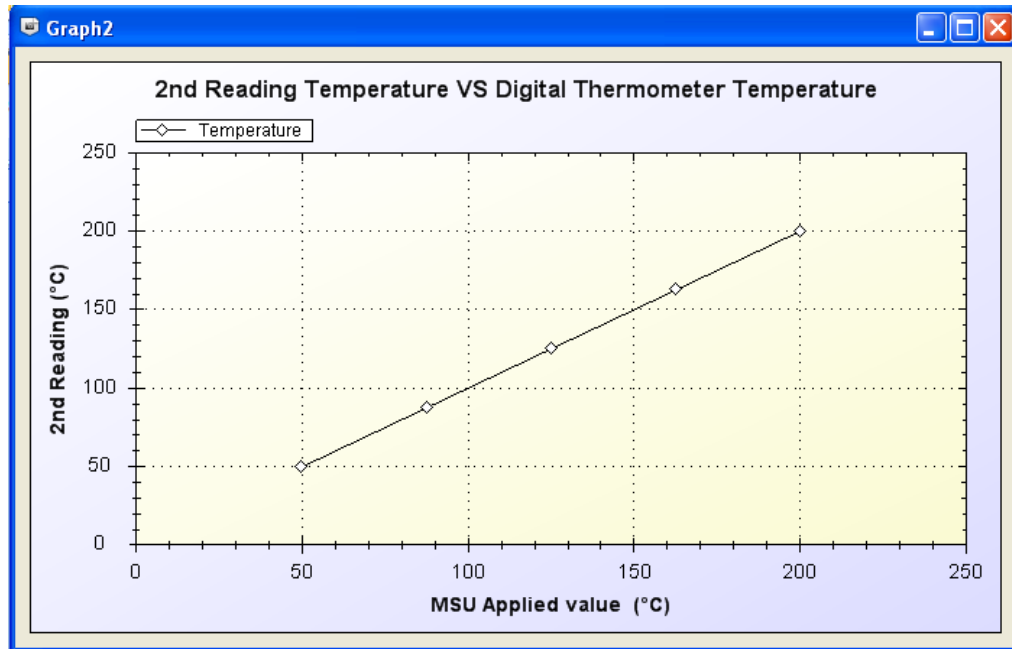
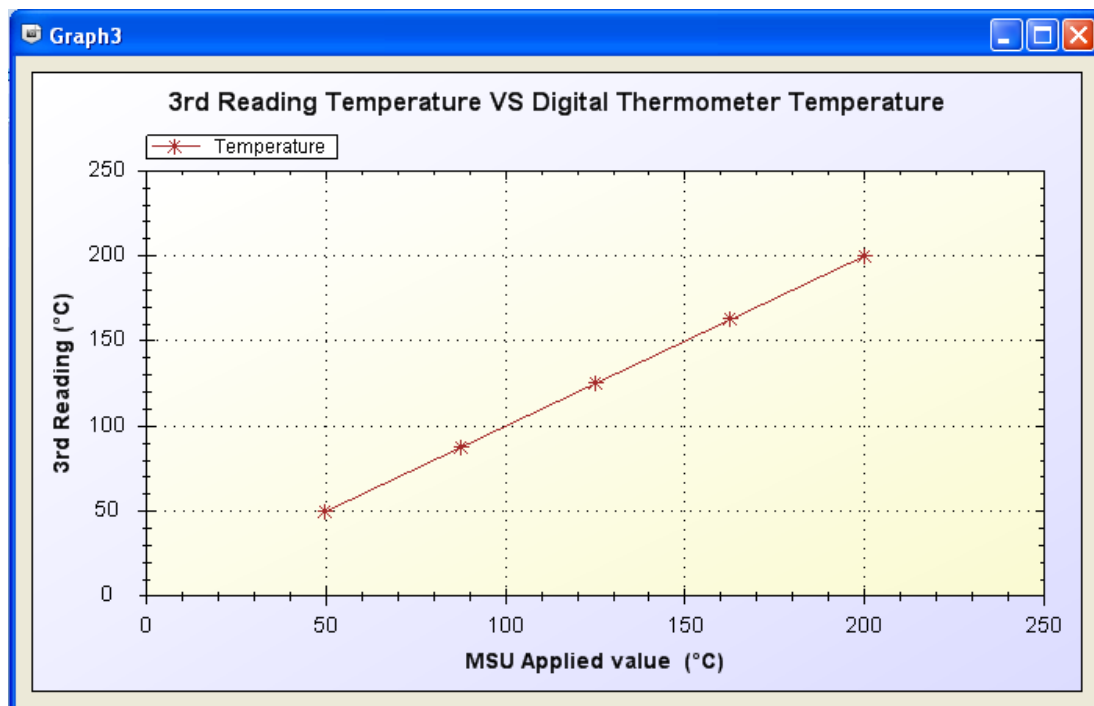
Figure 5.1: Result from software

Table 5.1: Result from software after export to Microsoft Excel

No (%)	MSU (°C)	MSU(Desired Output mA)	1stReading (°C)	2ndReading (°C)	3rdReading (°C)	Mean (°C)	Standard Deviation	Error (%)
0	50	4	49.95079	50.00229	50.04807	50.0004	0.0487	0.0008
25	87.5	8	87.49313	87.51602	87.52747	87.5122	0.0175	0.014
50	125	12	125.035	125.0298	125.0355	125.0334	0.0032	0.0267
75	162.5	16	162.492	162.5263	162.5149	162.5111	0.0175	0.0068
100	200	20	199.9657	199.9771	200	199.9809	0.0175	-0.0095

From the result above, the relationship between first reading, second reading and third reading with MSU in Celsius are shown in figure 5.21, 5.3 and 5.4 below:

Figure 5.2: 1st Reading Temperature vs. MSU applied (°C)

Figure 5.3: 2nd Reading Temperature vs. MSU applied (°C)Figure 5.4: 3rd Reading Temperature vs. MSU applied (°C)

5.3 MEAN AND STANDARD DEVIATION

The mean of the any distribution is a measure of centrality, but in case of the normal distribution, it is equal to the mode and median of the distribution. The standard deviation is a measure of data dispersion or variability. In the case of the normal distribution, the mean and the standard deviation are the two parameters of the distribution; therefore they completely define the distribution. In this case, the calculation of mean and standard deviation is using in this software to find the centrality and data dispersion measurement. The result from the experiment is shown in table 5.2:

Table 5.2: Mean and standard deviation result

MSU (°C)	1stReading (°C)	2ndReading (°C)	3rdReading (°C)	Mean (°C)	Standard Deviation
50	49.95079	50.00229	50.04807	50.0004	0.0487
87.5	87.49313	87.51602	87.52747	87.5122	0.0175
125	125.035	125.0298	125.0355	125.0334	0.0032
162.5	162.492	162.5263	162.5149	162.5111	0.0175
200	199.9657	199.9771	200	199.9809	0.0175

The calculation of mean and standard deviation based on the following formula:

Mean:

$$\bar{x} = \sum_{i=0}^n \frac{1}{n} (x_i) \text{-----} (5.2)$$

Standard deviation:

$$\sigma = \sqrt{\sum_{k=0}^{\infty} \frac{1}{N} (x_i - \bar{x})^2} \text{----- (5.3)}$$

Where;

\bar{x} is mean

σ is standard deviation

x_i is readings value

N is number of readings

As example, in this experiment, the MSU value is 50 °C and the first, second and third reading is 49.95079°C, 50.00229°C and 50.04807°C. The result of mean value and standard deviation value as below:

$$\bar{x} = \frac{49.95079^\circ\text{C} + 50.00229^\circ\text{C} + 50.04807^\circ\text{C}}{3}$$

$$\bar{x} = 50.00038^\circ\text{C}$$

$$\sigma = \sqrt{\frac{(49.95079 - 50.00038)^2 + (50.00229 - 50.00038)^2 + (50.04807 - 50.00038)^2}{3}}$$

$$\sigma = 0.039737$$

5.4 PERCENTAGE ERROR

Different computers may not have the same capability to perform complex mathematical operations and may produce significantly different results for the same problem. Because computers must manipulate data in a digital format, numerical errors in processing can lead to inaccurate results. In this case, calculation of error due to MSU is implementing in this software. The result from the experiment is shown in table 5.3:

Table 5.3: Percentage of error result

MSU (°C)	Mean (°C)	Error (%)
50	50.0004	0.0008
87.5	87.5122	0.014
125	125.0334	0.0267
162.5	162.5111	0.0068
200	199.9809	-0.0095

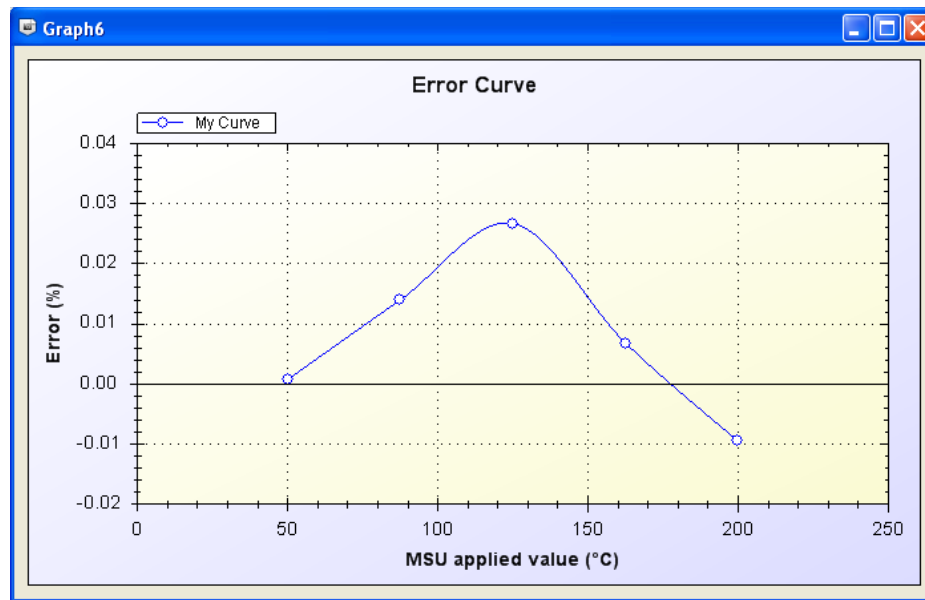


Figure 5.5: Percentage of Error vs. MSU applied

The calculation of percentage error based on the following formula:

$$E = \left| \frac{\text{Actual} - \text{Measured}}{\text{Actual}} \right| \times 100 \text{ ----- (5.4)}$$

As example, in this experiment, the MSU value which is actual value is 50 °C and the mean value which is measured value is 50.0004°C. The percentage of error calculated as below:

$$E = \left| \frac{50 - 50.0004}{50} \right| \times 100$$

$$E = 0.0008 \%$$

5.5 UNCERTAINTY

5.5.1 Calculate the uncertainty using software

This software also includes the calculator for uncertainty. Figure 5.6 to figure 5.9 show the result of calculation uncertainty using this software:

Uncertainty

U1 U2 U3 Uc

Uncertainty Due to Repeatability Of The Experiment (U1)

$$s(x_k) = \sqrt{\frac{1}{(n-1)} \sum_{j=1}^k (x_k - \bar{x})^2}$$

$$s(x) = \frac{s(x_k)}{\sqrt{n}} = \frac{0.0487}{\sqrt{3}}$$

U1 = 0.02811696 °C

Degree of freedom = 2

Calculate U1

Figure 5.6: Calculation Uncertainty due to repeatability of the experiment, U_1

Uncertainty

U1 U2 U3 Uc

Uncertainty Contribution Due To MSU Error (U2)

Accuracy Specification for Instrument
 ± (0.01 % of reading + 0.005 % range)

Maximum reading Range
 200 °C 37.5 °C

U2 = 0.01262953713852 °C

Calculate U2

Degree of freedom = infinity

Figure 5.7: Uncertainty contribution due to MSU error, U_2

Uncertainty

U1 U2 U3 U_c

Uncertainty Due To UUT Resolution / MSU Resolution (U3)

Please insert the Maximum Resolution

0.1 °C

U3 = 0.05773502691896 °C

Calculate U3

Degree of freedom = infinity

Figure 5.8: Uncertainty due to UUT resolution/MSU resolution, U_3

Uncertainty

U1 U2 U3 U_c

Combine Standard Uncertainty, u_c

$$u_c = \sqrt{u_1^2 + u_2^2 + u_3^2}$$

$$u_c = \sqrt{0.028117 \frac{2}{1} + 0.0126295 \frac{2}{2} + 0.057735 \frac{2}{3}}$$

= 0.0654477 Calculate Uc

Calculate Effective degree of freedom = 58.71299

Choose the value of confidence interval (k) to obtain confidence limits (u)

Confidence Interval U = $U_c \times 2.68$

Calculate The confidence Limits = $\pm 0.1754 \text{ °C}$

Figure 5.9: Combined uncertainty, U_c and calculate the confidence limits

5.5.2 Calculation of uncertainty using formula

Uncertainty due to repeatability of the experiment

From the data record tab, we select the worst case of standard deviation. The Uncertainty we are looking for is the experimental standard deviation of mean $s(x')$. this $s(x')$ is the estimation of the spread of the distribution of the means. For a sample size $n=3$ the formula for standard deviation of the mean is given by:

$$s(x) = \frac{s(x_k)}{\sqrt{n}} \text{-----} (5.5)$$

With degree of freedom $\gamma_1=3-1=2$

The worst case of standard deviation, $s(x_k)$ is = 0.039737

$$\begin{aligned} s(x) &= \frac{s(x_k)}{\sqrt{n}} = \frac{0.039737}{\sqrt{3}} \\ &= 0.022942 \end{aligned}$$

Uncertainty contribution due to MSU error

The MSU for this calibration is the model YT110, Temperature Transmitter. For the 100°C range accuracy specification for this instrument provided by the manufacturer is the following:

$$\pm(0.01\% \text{ of reading} + 0.005\% \text{ range}) \text{-----} (5.6)$$

So, a maximum reading of measurement is 200°C and range of instrument is 100°C. Hence the error in MSU is,

$$\begin{aligned} a &= \pm [(0.01\% \times 200^\circ\text{C}) + (0.005\% \times 37.5^\circ\text{C})] \\ &= \pm 0.021875^\circ\text{C} \end{aligned}$$

The uncertainty contribution due to MSU error is defined as u_2 and is given by

$$\begin{aligned} u_2 &= \frac{a}{\sqrt{n}} \text{-----} (5.7) \\ &= \frac{0.021875}{\sqrt{3}} \\ &= 0.01263 \end{aligned}$$

The degree of freedom γ_2 for this uncertainty is assumed to be ∞ since the manufacturer is expected to provide the error data after a large number of tests.

Therefore $u_2 = 0.01263^\circ\text{C}$ and $\gamma_2 = \infty$.

Uncertainty due to UUT Resolution/MSU resolution

For type B uncertainty, we can decide on resolution of MSU or resolution of UUT. Generally, if the UUT is analog, we will use the resolution of MSU. If the UUT is digital, we will use the resolution of digital UUT. The resolution of the UUT model YT110 by using METHOD 1, which is taken from instrument datasheet.

Considering the worst case scenario, the maximum resolution of YT110 is 0.1 °C. the uncertainty u_3 is calculated as:

$$u_3 = \frac{\text{Resolution}}{\sqrt{3}} \text{-----} (5.8)$$

$$\begin{aligned} u_3 &= \frac{0.1}{\sqrt{3}} \\ &= 0.057735 \end{aligned}$$

It can consider the degree of freedom as ∞ .

$$u_3 = 0.057735 \text{ and } \gamma_3 = \infty$$

Combined standard uncertainty, u_c

The combined standard uncertainty u_c is determined from the individual uncertainties u_1, u_2 and u_3 by the following formula:

$$u_c = \sqrt{u_1^2 + u_2^2 + u_3^2} \text{-----} (5.9)$$

$$\begin{aligned} u_c &= \sqrt{0.022942^2 + 0.01263^2 + 0.057735^2} \\ &= \sqrt{4.01918 \times 10^{-3}} \\ &= 0.063397 \end{aligned}$$

The effective degree of freedom γ_e is given by

$$\gamma_e = \frac{u_c^4}{\frac{u_1^4}{\gamma_1} + \frac{u_2^4}{\gamma_2} + \frac{u_3^4}{\gamma_3}} \text{-----} (5.10)$$

$$\gamma_e = \frac{0.063397^4}{\frac{0.022942^4}{2} + \frac{0.01263^4}{\infty} + \frac{0.057735^4}{\infty}}$$

$$= 116.6218$$

Substituting the values we get, $\gamma_e = 116.218 = 100$

The total uncertainty at any confidence level is determined using the user's distribution. The coverage factor k is determined from user table. Referring to the degree of freedom table in **Appendix B**, the value of γ is 100 and 99% confidence interval $k = 2.576$.

The confidence limit are obtained by the formula,

$$u = u_c k \text{ ----- (5.11)}$$

Therefore,

$$u = 0.063397 \times 2.576$$

$$= \pm 0.1633^\circ\text{C}$$

The confidence limits in a measurement are determined by the use of calibration techniques together with statistical principle.

5.6 RESULT ANALYSIS

From the software and calculation method that has been discuss before, the difference between this two methods are conclude in the Table 5.4:

Table 5.4: Comparison between two method results

	Software Calculation	Manual Calculation
Mean, \bar{x}	50.0004°C	50.00038°C
Standard Deviation Value, σ	0.0487	0.039737
Percentage of Error, E	0.0008%	0.0008%
U_1	0.028117	0.022942
U_2	0.0126295	0.01263
U_3	0.057735	0.057735
U_c	0.0654477	0.063397
Effective Degree of Freedom, γ_e	58.71299	116.6218
Confidence Interval	± 0.1754	± 0.1633

From the result of software and calculation method, it can be conclude that even the smallest different at the standard deviation value may cause largest different at the effective degree of freedom. However, the value of standard deviation did not affect much to the confidence interval since the difference only 0.01. Which mean, the software calculation method is have the same result to the manual calculation method. This software works well as a temperature transmitter calibrator according to the objective in this project.

From this calculation example also we could see that the worst case standard deviation is 0.0487 and generate 0.0008 % of error. Standard deviation statement makes

us know how much the data diverge from UUT. The smallest standard deviation, smallest error will occur and its may increase the accuracy of this software.

The graph shows in figure 5.2 until 5.4 proved that this software accuracy is high because of the data measured directly proportional to the unit under test (UUT) and produce linear graph. The error graph as shown in figure 5.5 explained that the range of error for this experiment is between -0.0095 % to 0.0267 %. The gap between maximum and minimum error is 0.1217 % which means almost no error in this experiment.

CHAPTER 6

CONCLUSION RECOMMENDATIONS

6.1 CONCLUSION

Designing software application for temperature transmitter calibration has been presented in this project. The development of automatic calibration transmitter using type K thermocouple has been done. Through these development it has conclude that Visual Basic software can be a good method for learn and explore the calibration and uncertainty process with and interactive way in order to reduce the human error.

The main objective for this project is to develop an automatic calibration transmitter using type k thermocouple via Microsoft Visual Basic application. The automatic calibration was successfully completed in accordance with the required specifications. This software comes with several basic applications such as uncertainty calculation, graph generator, live graph, temperature-to-current converter and etc. data that has been captured in this software also can save to the Microsoft Excel for references.

6.3 OBSTACLES FACES

While this project in progress, there are several problems that has been encountered such as limited time, instrument selection, and limitation of freeware.

i. Limited time

The time that provides to student is so limited since time for buying component and instrument from outside country is taking long time. The main idea in this project is to control the temperature bath automatically using computer. Since the limited time of researching and for buying component, the idea is not used.

ii. Instrument selection

When talk about automatic calibration, the system must all about measurement and calculation by itself. So the problem is which controller can be use to calibrate the temperature transmitter automatically. There are several suggestions for controller which is using PID controller and built PIC circuit. The problem if control by using PID controller is, it is difficult to interfacing the PID controller to the computer. Control the system using PIC circuit is a good idea, but the risk is PIC circuit is not sure can work properly since the main objective is to calibrate temperature transmitter and its cause wasting time.

iii. Limitation of freeware

Microsoft Visual Basic 2008 Express Edition used in this project is a freeware version. This free of charge version do not support GUI for mobile devices because of no templates and emulator provided. GUI in mobile devices is more advance and easy to carry to the factory site.

6.4 RECOMMENDATION

There are several recommendations that can be suggests for the improvement of this project which is:

- i. Use microcontroller in this project to get the best measurement and improve it with auto calibrator to the device that has been monitored their measurement. Generally, this project only doing monitoring and check the uncertainty of measurement but cannot repair the precision of instrument because of lack of time. The instrument precision is important in industry because the larger error on the instrument may cause lost thousand of dollar to the company.
- ii. Control the temperature bath automatically because it can improve this system. The main problem while done this experiment is its difficult to capture data of temperature since it increase the temperature very fast and sometimes it skip the MSU value. It cause data cannot been captured.
- iii. Make a portable instrument calibrator device. The easy-carry device can help engineer's job easy in the plant which is high and dangerous places. This portable device also need to robust.

REFERENCE

- [1] O.Kanoun, “*Measuring Temperature Calibration Free With Bipolar Transistor*”, Journal, Institut Fur Me& und automatisierungstechnik, Universitat der Bundeswehr Munchen, Neubiberg, Germany, 1998, pp. 619
- [2] M. Harker, P. O’Leary, “*Calibration, Measurement and Error Analysis of Optical Temperature Measurement via Laser Induced Fluorescence*”, Journal, Institute for Automation, University of Leoben, Leoben, Austria, May 1-3, 2007, PP. 1
- [3] T.Walach, “*Uncertainty in Temperature Infrared Measurement of Electronic Microcircuits*”, 15th International Conference on Mixed Design of Integrated Circuits and Systems, MIXDES 2008, Poznan, Poland, 19-21 June 2008, pp.359
- [4] B.Schuh, Watlow “*Smart Thermocouple System for Industrial Temperature Measurement*”, Sicon ’01 Sensor for Industry Conference Rosemounth, Illionis, USA, 5-7 November 2001, pp.9
- [5] O. Kochan, R. Kochan, O. Bojko, M. Chyrka, “*Temperature Measurement system Based on Thermocouple with Controlled Temperature Field*”, IEEE International Workshop on Intelligent data Acquisition and Advanced Computing systems: Technology and Applications, Dortmund, Germany, 6-8 September 2007, pp.48
- [6] M. Jerzy, K.A. Hetman, “*A calculation of Uncertainties in Virtual Instrument*”, Instrumentation and Measurement Technology Conference Ottawa, Canada, 17-19 May 2005, pp.1698
- [7] McGhee J., Kulesza W., Korczynski M.J., Henderson I., “*Scientific Metrology*”, ACGM LODART, Lodz, Poland, First Edition, September, 1996, reprint, July, 1998, ISBN83 90429993
- [8] McGhee J., Kulesza W., Korczynski M.J., Henderson I., “*Measurement Data Handling*”, ”, ACGM LODART, Lodz, Poland, First Edition, 2001, ISBN83-7283-007-x

- [9] McGhee J., Kulesza W., Korczynski M.J., Henderson I., “*The Sensor effect tetrahedron: an extended transducer space*”, Measurement Journal of the International Measurement Confederation ISSN 0263-2241, pp. 217-236
- [10] Y.M.Wang, “*A method based on standard and mean deviations for determining the weight coefficients of multiple attributes and its applications*,” Mathematical Statistics and Management, vol.22 pp.22- 26, 2003.
- [11] Y.Xu, Z.Cai, “*Standard Deviation Method for Determining the Weights of Group Multiple Attribute Decision Making under Uncertain Linguistic Environment*,” the 7th World Congress on Intelligent Control and Automation June 25 - 27, 2008, Chongqing, China, pp 8313.
- [12] B. Vilters, C. Jakus and J. Peperstraete. “*An integrated process controller, programmable logic controller and personal computer setup for didactical purposes*,” *Trends in Control and Measurement Education. Selected Papers from the IFAC Symposium*, pages 33 - 6, July 1988.
- [13] R. C. Ciammichella and D. W Appleby. “*Automating the small batch operation*,” *InTech*, Vol: 41, Iss: 3, pp. 39 - 42, Mar. 1994.
- [14] J. F. Manji., “*IC Integrating PCs and PLCs yields a formula for success*,” *Controls i3 Systems*, Vol: 39, Iss: 11, pp. 46 - 48, Nov. 1992.
- [15] A. Daniel. *Visual Basic Programmer’s Guide to the Windows API*. Ziff-Davis Press, 1.993. pp. 621 - 659.
- [16] S.L Chung, W.F Yang, “*Data Acquisition and Integration in Heterogeneous Computing Environment*”, Industrial Automation and control: Emerging Technologies, 1995., IEEE/IAS Conference on May 2005. Pp 598-599.

APPENDIX A

SOFTWARE CODING

```

Imports System.Data
Imports ZedGraph
Imports System.Drawing.Drawing2D
Imports System.Data.OleDb

Public Class Form1
    Dim inc As Integer
    Dim MaxRows As Integer
    Dim con As New OleDb.OleDbConnection
    Dim sql As String
    Dim tickStart As Integer = 0
    Dim x As Integer
    Dim paint1 As Integer
    Dim tempIns As Double
    Dim volt As Double
    Dim incr As Integer
    Dim alm1 As Integer
    Dim alm2 As Integer
    Dim alm3 As Integer
    Dim alm4 As Integer
    Dim alm5 As Integer
    Const DATA_FILE_EXTENSION As String = ".mdb"

    Private da As OleDbDataAdapter
    Private ds As New DataSet()

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        timer()

        btnRecord.Enabled = False
        btnReset.Enabled = False
        btnStop.Enabled = False
        btnRecord2.Enabled = False
        btnReset2.Enabled = False

        database()

    End Sub
    Private Sub Load_Excel_Details()
        'Extracting from database
        Dim filename As String

    Try
        da.Fill(ds, "proto")
        If ds.Tables.Count < 0 Or ds.Tables(0).Rows.Count <= 0 Then
            Exit Sub
        End If
    Catch ex As Exception
        MsgBox(ex.Message)
    End Try
    Dim Excel As Object = CreateObject("Excel.Application")
    If Excel Is Nothing Then
        MsgBox("It appears that Excel is not installed on this machine. This operation requires MS Excel to be installed on this machine.", MsgBoxStyle.Critical)
        Return
    End If

    'Export to Excel process
    Try
        With Excel
            .SheetsInNewWorkbook = 1
            .Workbooks.Add()
            .Worksheets(1).Select()

            Dim i As Integer = 1

                For col = 0 To ds.Tables(0).Columns.Count - 1
                    .cells(1, i).value = ds.Tables(0).Columns(col).ColumnName
                    .cells(1, i).EntireRow.Font.Bold = True
                    i += 1
                Next
                i = 2
                Dim k As Integer = 1
                For col = 0 To ds.Tables(0).Columns.Count - 1
                    i = 2
                    For row = 0 To ds.Tables(0).Rows.Count - 1
                        .Cells(i, k).Value = ds.Tables(0).Rows(row).ItemArray(col)
                        i += 1
                    Next
                    k = 2
                Next
                filename = txtPath.Text & "\" & Format(Now(), "dd-MM-yyyy_hh-mm-ss") & ".xls"
                .ActiveCell.Worksheet.SaveAs(filename)
            End With

            System.Runtime.InteropServices.Marshal.ReleaseComObject(Excel)
            Excel = Nothing
            MsgBox("Data's are exported to Excel Successfully in '" & filename & "'", MsgBoxStyle.Information)

        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

        ' The excel is created and opened for insert value. We most close this excel using this system
        Dim pro() As Process = System.Diagnostics.Process.GetProcessesByName("EXCEL")
        For Each i As Process In pro
            i.Kill()
        Next

    End Sub

    Private Sub timer()
        Dim myPane As GraphPane = ZedGraphControl1.GraphPane
        myPane.Title.Text = "Voltage VS Time" & Chr(10) & _
            "(After 25 seconds the graph scrolls)"
        myPane.XAxis.Title.Text = "Time, Seconds"
        myPane.YAxis.Title.Text = "Sample Potential, Volts"

        ' Save 1200 points. At 50 ms sample rate, this is one minute
        ' The RollingPointPairList is an efficient storage class that always
        ' keeps a rolling set of point data without needing to shift any data values
        Dim list As New RollingPointPairList(1200)

        ' Initially, a curve is added with no data points (list is empty)
        ' Color is blue, and there will be no symbols
    End Sub

```

```

    Dim curve As LineItem =
myPane.AddCurve("Voltage", list, Color.Blue,
SymbolType.None)

    ' Sample at 50ms intervals
Timer1.Interval = 50

    ' Just manually control the X axis range so
it scrolls continuously
    ' instead of discrete step-sized jumps
myPane.XAxis.Scale.Min = 0
myPane.XAxis.Scale.Max = 30
myPane.YAxis.Scale.Min = 0
myPane.YAxis.Scale.Max = 6
myPane.XAxis.Scale.MinorStep = 1
myPane.XAxis.Scale.MajorStep = 5

    'curve.Line.StepType = StepType.ForwardStep

myPane.Chart.Fill = New Fill(Color.White,
Color.LightGoldenrodYellow, 45.0F)

    ' Fill the pane background with a color
gradient
myPane.Fill = New Fill(Color.White,
Color.FromArgb(220, 220, 255), 45.0F)

myPane.XAxis.MajorGrid.IsVisible = True
myPane.YAxis.MajorGrid.IsVisible = True

    ' Scale the axes
ZedGraphControl1.AxisChange()

    ' Save the beginning time for reference
tickStart = Environment.TickCount
End Sub

Private Sub Button5_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs)
Dim y As Double

inc = 0
y = 0
For i = 0 To MaxRows - 1
    y = y +
ds.Tables("proto").Rows(inc).Item(2)
    inc = inc + 1
Next i
MsgBox(y)

End Sub

Private Sub Button9_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button9.Click

If CheckBox1.Checked Then
    Graph1.Visible = True
End If
If CheckBox2.Checked Then
    Graph2.Visible = True
End If
If CheckBox3.Checked Then
    Graph3.Visible = True
End If
If CheckBox4.Checked Then
    Graph4.Visible = True
End If
If CheckBox5.Checked Then
    Graph5.Visible = True
End If
If CheckBox6.Checked Then
    Graph6.Visible = True
End If

End Sub
Private _drawInsidePanel As Boolean

Private Sub Timer1_Tick(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Timer1.Tick
Dim textA1 As Integer
Dim textA2 As Integer
Dim textA3 As Integer
Dim textA4 As Integer
Dim textA5 As Integer

AutoKeyIn()
MeanAuto()

    ' Make sure that the curvelist has at least
one curve
If
ZedGraphControl1.GraphPane.CurveList.Count <= 0 Then
Return

    ' Get the first CurveItem in the graph
Dim curve As LineItem =
ZedGraphControl1.GraphPane.CurveList(0)
If curve Is Nothing Then Return

    ' Get the PointPairList
Dim list As IPointListEdit = curve.Points
    ' If this is null, it means the reference at
curve.Points does not
    ' support IPointListEdit, so we won't be
able to modify it
If list Is Nothing Then Return

    ' Time is measured in seconds
Dim time As Double = (Environment.TickCount
- tickStart) / 1000.0

If rbtnSimulation.Checked Then
    ' 3 seconds per cycle
    ' Produce dummy data range 1-5V
    volt = (Math.Sin(2 * Math.PI * time /
3.0)) * 2 + 3

Else
    volt = AxAdvA11.DataAnalog
End If

list.Add(time, volt)
    ' Keep the X scale at a rolling 30 second
interval, with one
    ' major step between the max X value and the
end of the axis
Dim xScale As Scale =
ZedGraphControl1.GraphPane.XAxis.Scale
If time > xScale.Max - xScale.MajorStep Then
    xScale.Max = time + xScale.MajorStep
    xScale.Min = xScale.Max - 30.0
End If

    ' Make sure the Y axis is rescaled to
accommodate actual data
ZedGraphControl1.AxisChange()
    ' Force a redraw
ZedGraphControl1.Invalidate()

    ' Data
InstrumentCalibration()

txtDataTime.Text = Math.Round(time, 1) & "s"
txtDataTemp.Text = Math.Round(tempIns, 3) &
"°C"
txtDataVolt.Text = Math.Round(volt, 4) & "V"
txtDataCurrent.Text = Math.Round(((volt /
250) * 10 ^ (3)), 3) & "mA"

textA1 = txtA1.Text
textA2 = txtA2.Text
textA3 = txtA3.Text

```

```

textA4 = txtA4.Text
textA5 = txtA5.Text

    If Math.Round(textA1, 0) >
Math.Round(tempIns, 0) And cmbA1.SelectedItem =
"Enable" And alm1 = 1 Then
    alm1 = 0
    MsgBox("Temperature is recorded in
database.")
    ElseIf Math.Round(textA2, 0) >
Math.Round(tempIns, 0) And cmbA2.SelectedItem =
"Enable" And alm2 = 1 Then
    alm2 = 0
    MsgBox("Temperature is recorded in
database.")
    ElseIf Math.Round(textA3, 0) >
Math.Round(tempIns, 0) And cmbA3.SelectedItem =
"Enable" And alm3 = 1 Then
    alm3 = 0
    MsgBox("Temperature is recorded in
database.")
    ElseIf Math.Round(textA4, 0) >
Math.Round(tempIns, 0) And cmbA4.SelectedItem =
"Enable" And alm4 = 1 Then
    alm4 = 0
    MsgBox("Temperature is recorded in
database.")
    ElseIf Math.Round(textA5, 0) >
Math.Round(tempIns, 0) And cmbA5.SelectedItem =
"Enable" And alm5 = 1 Then
    alm5 = 0
    MsgBox("Temperature is recorded in
database.")
    End If

    paint1 = Math.Round(time, 0)
    _drawInsidePanel = True
    Panell.Invalidate()
' force to redraw the Panell

    End Sub

    Private Sub Panell_Paint(ByVal sender As
System.Object, ByVal e As
System.Windows.Forms.PaintEventArgs) Handles
Panell.Paint
    Dim g As Graphics = e.Graphics
    Dim rect As New Rectangle
    Dim y As Integer

    y = 85 * volt - 85

    g.FillRectangle(Brushes.Red, 0, 0, 20, 340)
    If _drawInsidePanel Then
        ' Draw inside the panel
        rect = New Rectangle(0, 0, 80, 340 - y)
        g.FillRectangle(Brushes.AliceBlue, 0, 0,
80, 340 - y)
    End If
    End Sub

    Private Sub Button3_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnStart.Click
    Timer1.Enabled = True
    Timer1.Start()
    RadioButton1.Enabled = False
    RadioButton2.Enabled = False
    RadioButton3.Enabled = False
    btnRecord.Enabled = True
    btnReset.Enabled = True
    btnStart.Enabled = False
    btnStop.Enabled = True
    btnRecord2.Enabled = True
    btnReset2.Enabled = True
    My.Forms.Voltage.Timer1.Enabled = True
    My.Forms.Voltage.Timer1.Start()

    incr = 1
    End Sub

    Private Sub Button4_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnStop.Click
    Timer1.Enabled = False
    Timer1.Stop()
    RadioButton1.Enabled = True
    RadioButton2.Enabled = True
    RadioButton3.Enabled = True
    btnRecord.Enabled = False
    btnReset.Enabled = False
    btnStart.Enabled = True
    btnStop.Enabled = False
    btnRecord2.Enabled = False
    btnReset2.Enabled = False
    My.Forms.Voltage.Timer1.Enabled = False
    My.Forms.Voltage.Timer1.Stop()

    incr = 1
    End Sub

    Private Sub Button10_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button10.Click
    Form2.ShowDialog()
    End Sub

    Private Sub Button1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnRecord.Click
    MaxRows = ds.Tables("proto").Rows.Count

    InstrumentCalibration()
    If RadioButton1.Checked And MaxRows <> incr
Then
        ds.Tables("proto").Rows(incr).Item(4) =
Math.Round(tempIns, 3)
        MsgBox("Data " & incr & " is recorded in
1st sample")
        incr = incr + 1
        ElseIf RadioButton2.Checked And MaxRows <>
incr Then
            ds.Tables("proto").Rows(incr).Item(5) =
Math.Round(tempIns, 3)
            MsgBox("Data " & incr & " is recorded in
2nd sample")
            incr = incr + 1
            ElseIf RadioButton3.Checked And MaxRows <>
incr Then
                ds.Tables("proto").Rows(incr).Item(6) =
Math.Round(tempIns, 3)
                MsgBox("Data " & incr & " is recorded in
3rd sample")
                incr = incr + 1
            End If
        End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
ComboBox1.SelectedIndexChanged
    Dim myPane As GraphPane =
ZedGraphControl1.GraphPane
    myPane.XAxis.MajorGrid.IsVisible =
ComboBox1.Text
    End Sub

    Private Sub ComboBox2_SelectedIndexChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
ComboBox2.SelectedIndexChanged
    Dim myPane As GraphPane =
ZedGraphControl1.GraphPane
    myPane.YAxis.MajorGrid.IsVisible =
ComboBox2.Text
    End Sub

```

```

Private Sub ComboBox3_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbA1.SelectedIndexChanged
    If cmbA1.SelectedIndex = 1 Then
        txtA1.Enabled = False
        alrm1 = 1
    Else
        txtA1.Enabled = True
    End If
End Sub

Private Sub cmbA2_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbA2.SelectedIndexChanged
    If cmbA2.SelectedIndex = 1 Then
        txtA2.Enabled = False
        alrm2 = 1
    Else
        txtA2.Enabled = True
    End If
End Sub

Private Sub cmbA3_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbA3.SelectedIndexChanged
    If cmbA3.SelectedIndex = 1 Then
        txtA3.Enabled = False
        alrm3 = 1
    Else
        txtA3.Enabled = True
    End If
End Sub

Private Sub cmbA4_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbA4.SelectedIndexChanged
    If cmbA4.SelectedIndex = 1 Then
        txtA4.Enabled = False
        alrm4 = 1
    Else
        txtA4.Enabled = True
    End If
End Sub

Private Sub cmbA5_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbA5.SelectedIndexChanged
    If cmbA5.SelectedIndex = 1 Then
        txtA5.Enabled = False
        alrm5 = 1
    Else
        txtA5.Enabled = True
    End If
End Sub

Private Sub cmbSample_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmbSample.SelectedIndexChanged
    If cmbSample.SelectedItem = "1" Then
        Timer1.Interval = 1000
    ElseIf cmbSample.SelectedItem = "2" Then
        Timer1.Interval = 500
    ElseIf cmbSample.SelectedItem = "4" Then
        Timer1.Interval = 250
    ElseIf cmbSample.SelectedItem = "8" Then
        Timer1.Interval = 125
    ElseIf cmbSample.SelectedItem = "16" Then
        Timer1.Interval = 62.5
    ElseIf cmbSample.SelectedItem = "32" Then
        Timer1.Interval = 31.25
    End If
End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnExport.Click
    Load_Excel_Details()
End Sub

Private Sub cmdSelectDevice_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmdSelectDevice.Click
    AxAdvA11.SelectDevice()
    txtDeviceNumber.Text = AxAdvA11.DeviceNumber
    txtDeviceName.Text = AxAdvA11.DeviceName
End Sub

Private Sub cmdRead_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmdRead.Click
    txtDataDigital.Text = Hex(AxAdvA11.DataDigital)
    txtDataAnalog.Text = Format(AxAdvA11.DataAnalog, "0.#####0")
End Sub

Private Sub InstrumentCalibration()
    Dim y1 As Double
    Dim y2 As Double
    Dim x1 As Double
    Dim x2 As Double
    Dim m As Double
    Dim c As Double
    Dim yIns As Double

    y1 = txtTempLR.Text
    y2 = txtTempUR.Text
    x1 = (txtTcLR.Text) * (1 * 10 ^ (-3))
    x2 = (txtTcUR.Text) * (1 * 10 ^ (-3))

    m = (y2 - y1) / (x2 - x1)
    c = y2 - (x2 * m)
    yIns = m * (volt / 250) + c
    tempIns = yIns
End Sub

Private Sub btnMean_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnMean.Click
    Dim y As Double
    Dim a As Double
    Dim b As Double
    Dim c As Double
    Dim obj As Object
    Dim obj2 As Object
    Dim obj3 As Object
    Dim g As Integer

    For i = 1 To MaxRows - 1

        obj = ds.Tables("proto").Rows(i).Item(4)
        obj2 = ds.Tables("proto").Rows(i).Item(5)
        obj3 = ds.Tables("proto").Rows(i).Item(6)

        If IsDBNull(obj) Or IsDBNull(obj2) Or IsDBNull(obj3) Then
            g = 1
        Else
            a = ds.Tables("proto").Rows(i).Item(4)
            b = ds.Tables("proto").Rows(i).Item(5)
            c = ds.Tables("proto").Rows(i).Item(6)
            y = (a + b + c) / 3
            ds.Tables("proto").Rows(i).Item(7) = Math.Round(y, 4)
        End If

        If g = 1 And i = MaxRows - 1 Then
            MsgBox("Not Enough Data")
        End If

    Next i
End Sub

```

```

Private Sub btnStd_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnStd.Click
    Dim avg As Double
    Dim a As Double
    Dim b As Double
    Dim c As Double
    Dim sum As Double
    Dim obj As Object
    Dim obj2 As Object
    Dim obj3 As Object
    Dim g As Integer

    For i = 1 To MaxRows - 1

        obj = ds.Tables("proto").Rows(i).Item(4)
        obj2 =
ds.Tables("proto").Rows(i).Item(5)
        obj3 =
ds.Tables("proto").Rows(i).Item(6)

        If IsDBNull(obj) Or IsDBNull(obj2) Or
IsDBNull(obj3) Then
            g = 1
        Else
            a =
ds.Tables("proto").Rows(i).Item(4)
            b =
ds.Tables("proto").Rows(i).Item(5)
            c =
ds.Tables("proto").Rows(i).Item(6)
            avg = (a + b + c) / 3
            sum = Math.Sqrt((0.5) * ((a - avg) ^
2 + (b - avg) ^ 2 + (c - avg) ^ 2))

            ds.Tables("proto").Rows(i).Item(8) =
Math.Round(sum, 4)
        End If

        If g = 1 And i = MaxRows - 1 Then
            MsgBox("Not Enough Data")
        End If
    Next i
End Sub

Private Sub btnError_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnError.Click
    Dim avg As Double
    Dim a As Double
    Dim b As Double
    Dim c As Double
    Dim x As Double
    Dim sum As Double
    Dim obj As Object
    Dim obj2 As Object
    Dim obj3 As Object
    Dim obj4 As Object
    Dim g As Integer

    For i = 1 To MaxRows - 1

        obj = ds.Tables("proto").Rows(i).Item(4)
        obj2 =
ds.Tables("proto").Rows(i).Item(5)
        obj3 =
ds.Tables("proto").Rows(i).Item(6)
        obj4 =
ds.Tables("proto").Rows(i).Item(2)

        If IsDBNull(obj) Or IsDBNull(obj2) Or
IsDBNull(obj3) Or IsDBNull(obj4) Then
            g = 1
        Else
            a =
ds.Tables("proto").Rows(i).Item(4)

```

```

        b =
ds.Tables("proto").Rows(i).Item(5)
        c =
ds.Tables("proto").Rows(i).Item(6)
        x =
ds.Tables("proto").Rows(i).Item(2)
        avg = (a + b + c) / 3

        If x > avg Then
            sum = ((avg - x) / x) * 100
        End If

        ds.Tables("proto").Rows(i).Item(9) = Math.Round(sum,
4)

        ElseIf avg > x Then
            sum = ((avg - x) / x) * 100
        End If

        ds.Tables("proto").Rows(i).Item(9) = Math.Round(sum,
4)

        ElseIf x = avg Then
            sum = 0
        End If

        ds.Tables("proto").Rows(i).Item(9) = Math.Round(sum,
4)

        End If
    End If

    If g = 1 And i = MaxRows - 1 Then
        MsgBox("Not Enough Data")
    End If

Next i
End Sub

Private Sub btnReset_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnReset.Click
    If RadioButton1.Checked And incr <> 1 Then
        incr = incr - 1
        ds.Tables("proto").Rows(incr).Item(4) =
0
        MsgBox("Data " & incr & " is deleted
from 1st sample")
    ElseIf RadioButton2.Checked And incr <> 1
Then
        incr = incr - 1
        ds.Tables("proto").Rows(incr).Item(5) =
0
        MsgBox("Data " & incr & " is deleted
from 1st sample")
    ElseIf RadioButton3.Checked And incr <> 1
Then
        incr = incr - 1
        ds.Tables("proto").Rows(incr).Item(6) =
0
        MsgBox("Data " & incr & " is deleted
from 1st sample")
    End If
End Sub

Private Sub btnAlarmSave_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnAlarmSave.Click
    Dim lowRange As Double
    Dim highRange As Double

    lowRange = txtTempLR.Text
    highRange = txtTempUR.Text

    If cmbA1.SelectedItem = "Enable" Then
        alrm1 = 1
    ElseIf cmbA1.SelectedItem = "Disable" Then
        alrm1 = 0
    End If

    If cmbA2.SelectedItem = "Enable" Then
        alrm2 = 1
    ElseIf cmbA2.SelectedItem = "Disable" Then
        alrm2 = 0
    End If

```



```

If cmbA3.SelectedItem = "Enable" Then
    alm3 = 1
ElseIf cmbA3.SelectedItem = "Disable" Then
    alm3 = 0
End If

If cmbA4.SelectedItem = "Enable" Then
    alm4 = 1
ElseIf cmbA4.SelectedItem = "Disable" Then
    alm4 = 0
End If

If cmbA5.SelectedItem = "Enable" Then
    alm5 = 1
ElseIf cmbA5.SelectedItem = "Disable" Then
    alm5 = 0
End If

If txtA1.Text < lowRange Or txtA1.Text >
highRange Then
    MsgBox("Out of range.The range is
between " & lowRange & "°C" & " to " & highRange &
"°C")
    txtA1.Text = lowRange
    alm1 = 0
End If

If txtA2.Text < lowRange Or txtA2.Text >
highRange Then
    MsgBox("Out of range.The range is
between " & lowRange & "°C" & " to " & highRange &
"°C")
    txtA2.Text = lowRange
    alm2 = 0
End If

If txtA3.Text < lowRange Or txtA3.Text >
highRange Then
    MsgBox("Out of range.The range is
between " & lowRange & "°C" & " to " & highRange &
"°C")
    txtA3.Text = lowRange
    alm3 = 0
End If

If txtA4.Text < lowRange Or txtA4.Text >
highRange Then
    MsgBox("Out of range.The range is
between " & lowRange & "°C" & " to " & highRange &
"°C")
    txtA4.Text = lowRange
    alm4 = 0
End If

If txtA5.Text < lowRange Or txtA5.Text >
highRange Then
    MsgBox("Out of range.The range is
between " & lowRange & "°C" & " to " & highRange &
"°C")
    txtA5.Text = lowRange
    alm5 = 0
End If

End Sub

Private Sub Button4_Click_2(ByVal sender As
System.Object, ByVal e As System.EventArgs)
'Sets the device number of AdvA01 to 0
AxAdvA01.DeviceNumber = 0

'Set the ChannelNow of AdvA01 to 0
AxAdvA01.ChannelNow = 0

'Set the ChannelNow as a voltage output
'AxAdvA01.DataPhysics = 0

'Set the value range
AxAdvA01.SetValueRange(AxAdvA01.ChannelNow,
0, 5)
AxAdvA01.DataAnalog = 4.5

End Sub

Private Sub Button5_Click_1(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button5.Click
Voltage.Visible = True
End Sub

Private Sub btnBrowse_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnBrowse.Click
Dim objFolderDialog As New
FolderBrowserDialog()
'==== Pass object as Parameter and get
Selected network folder
txtPath.Text =
GetNetworkFolders(objFolderDialog)
End Sub
Public Shared Function GetNetworkFolders(ByVal
oFolderBrowserDialog _
As FolderBrowserDialog) As String

If oFolderBrowserDialog.ShowDialog() =
DialogResult.OK Then
Return oFolderBrowserDialog.SelectedPath
Else
Return ""
End If
End Function

Private Sub txtTempLR_TextChanged(ByVal sender
As System.Object, ByVal e As System.EventArgs)
Handles txtTempLR.TextChanged

End Sub

Private Sub btnSave_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnSave.Click
' Make a command builder to generate INSERT,
UPDATE, and DELETE commands as necessary.
Dim command_builder As New
OleDbCommandBuilder(da)

' Save any changes.
da.Update(ds, "proto")
MsgBox("Data has been saved")
End Sub

Private Sub Button7_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button7.Click
vCurrent.Visible = True
End Sub

Private Sub Button6_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button6.Click
vTemperature.Visible = True
End Sub

Private Sub btnRecord2_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnRecord2.Click
MaxRows = ds.Tables("proto").Rows.Count

If RadioButton1.Checked And MaxRows <> incr
Then
ds.Tables("proto").Rows(incr).Item(4) =
Math.Round(volt / 0.25, 5)
MsgBox("Data " & incr & " is recorded in
1st sample")
incr = incr + 1

```

```

ElseIf RadioButton2.Checked And MaxRows <>
incr Then
    ds.Tables("proto").Rows(incr).Item(5) =
Math.Round(volt / 0.25, 5)
    MsgBox("Data " & incr & " is recorded in
2nd sample")
    incr = incr + 1
ElseIf RadioButton3.Checked And MaxRows <>
incr Then
    ds.Tables("proto").Rows(incr).Item(6) =
Math.Round(volt / 0.25, 5)
    MsgBox("Data " & incr & " is recorded in
3rd sample")
    incr = incr + 1
End If
End Sub

Private Sub btnReset2_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnReset2.Click
    If RadioButton1.Checked And incr <> 1 Then
        incr = incr - 1
        ds.Tables("proto").Rows(incr).Item(4) =
0
        MsgBox("Data " & incr & " is deleted
from 1st sample")
    ElseIf RadioButton2.Checked And incr <> 1
Then
        incr = incr - 1
        ds.Tables("proto").Rows(incr).Item(5) =
0
        MsgBox("Data " & incr & " is deleted
from 1st sample")
    ElseIf RadioButton3.Checked And incr <> 1
Then
        incr = incr - 1
        ds.Tables("proto").Rows(incr).Item(6) =
0
        MsgBox("Data " & incr & " is deleted
from 1st sample")
    End If
End Sub

Private Sub btnConvert_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnConvert.Click
    Dim y1 As Double
    Dim y2 As Double
    Dim x1 As Double
    Dim x2 As Double
    Dim m As Double
    Dim c As Double
    Dim x As Double
    Dim y As Double

    If txtMinTemp.Text = "" Or txtMaxTemp.Text =
"" Or txtMaxCur.Text = "" Or txtMinCur.Text = ""
Then
        MsgBox("Please fill in the required
value")
    ElseIf rbCurtoTemp.Checked And
txtValCur.Text = "" Or rbTempToCur.Checked And
txtValTemp.Text = "" Then
        MsgBox("Please fill in the required
value")
    Else
        y1 = txtMinTemp.Text
        y2 = txtMaxTemp.Text
        x1 = (txtMinCur.Text)
        x2 = (txtMaxCur.Text)

        m = (y2 - y1) / (x2 - x1)
        c = y2 - (x2 * m)

        If rbCurtoTemp.Checked Then
            x = txtValCur.Text
            y = m * x + c
            txtValTemp.Text = Math.Round(y, 5)
        ElseIf rbTempToCur.Checked Then
            y = txtValTemp.Text
            x = (y - c) / m

            txtValCur.Text = Math.Round(x, 5)
        End If
    End Sub

Private Sub rbCurtoTemp_CheckedChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles rbCurtoTemp.CheckedChanged
    If rbCurtoTemp.Checked Then
        txtValTemp.ReadOnly = True
        txtValCur.ReadOnly = False
    ElseIf rbTempToCur.Checked Then
        txtValTemp.ReadOnly = False
        txtValCur.ReadOnly = True
    End If
End Sub

Private Sub btnGenerate_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnGenerate.Click
    Dim command_builder As New
OleDbCommandBuilder(da)
    Dim x As Integer
    Dim percent As Single
    Dim a As Single
    Dim incrx As Integer
    Dim temp As Single
    Dim upper As Single
    Dim lower As Single
    Dim uppCur As Single
    Dim lowCur As Single
    Dim current As Single
    Dim more As Integer
    Dim less As Integer

    If txtUpperVal.Text = "" Or txtLowerVal.Text
= "" Then
        MsgBox("Please enter a correct value")
    Else
        upper = txtUpperVal.Text
        lower = txtLowerVal.Text
        uppCur = txtTcUR.Text
        lowCur = txtTcLR.Text
        x = txtPoint.Text - 1
        percent = 100 / x

        MaxRows = ds.Tables("proto").Rows.Count

        If txtPoint.Text > (MaxRows - 1) Then
            more = txtPoint.Text - (MaxRows - 1)
            For i = 1 To more
                ds.Tables("proto").Rows.Add()
                da.Update(ds, "proto")
            Next i
        ElseIf txtPoint.Text < (MaxRows - 1)
Then
            less = (MaxRows - 1) - txtPoint.Text
            For i = 1 To less
                MaxRows =
ds.Tables("proto").Rows.Count
                ds.Tables("proto").Rows(MaxRows
- 1).Delete()
                da.Update(ds, "proto")
            Next i
        End If

        MaxRows = ds.Tables("proto").Rows.Count
        a = 0
        For i = 1 To MaxRows - 1
            incrx = incrx + 1

            ds.Tables("proto").Rows(incrx).Item(1) = a
            temp = (a / 100) * (upper - lower) +
lower
        Next i
    End Sub

```

```

ds.Tables("proto").Rows(incr).Item(2) = temp
    current = (a / 100) * (uppCur -
lowCur) + lowCur

ds.Tables("proto").Rows(incr).Item(3) = current
    a = a + percent
    Next i
End If
End Sub

Private Sub btnDel_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs)
    Dim command_builder As New
OleDbCommandBuilder(da)
    MaxRows = ds.Tables("proto").Rows.Count
    ds.Tables("proto").Rows(MaxRows -
1).Delete()
End Sub
Private Sub btnSaveData_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnSaveData.Click
    Data_Recorder.Visible = True
End Sub

Private Sub btnPlotter_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnPlotter.Click
    OpenFileDialog1.Filter = DATA_FILE_EXTENSION
& _
    " files (*.*) & DATA_FILE_EXTENSION & "|*"
& DATA_FILE_EXTENSION
    OpenFileDialog1.FilterIndex = 1
    OpenFileDialog1.RestoreDirectory = True
    OpenFileDialog1.ShowDialog()
End Sub

Private Sub OpenFileDialog1_FileOk(ByVal sender
As System.Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles
OpenFileDialog1.FileOk
    Plotter.Visible = True
End Sub

Private Sub btnBrowseDatabase_Click(ByVal sender
As System.Object, ByVal e As System.EventArgs)
Handles btnBrowseDatabase.Click
    OpenFileDialog2.Filter = DATA_FILE_EXTENSION
& _
    " files (*.*) & DATA_FILE_EXTENSION & "|*"
& DATA_FILE_EXTENSION
    OpenFileDialog2.FilterIndex = 1
    OpenFileDialog2.RestoreDirectory = True
    OpenFileDialog2.ShowDialog()
End Sub

Private Sub OpenFileDialog2_FileOk(ByVal sender
As System.Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles
OpenFileDialog2.FileOk
    txtDatabase.Text = OpenFileDialog2.FileName
database()
End Sub
Private Sub database()

    con.ConnectionString =
"PROVIDER=Microsoft.Jet.OLEDB.4.0;Data Source =" &
txtDatabase.Text

    con.Open()

    sql = " select*from tblContacts"
    da = New OleDb.OleDbDataAdapter(sql, con)
    da.Fill(ds, "proto")

    con.Close()

    MaxRows = ds.Tables("proto").Rows.Count
    inc = -1

```

```

x = 0

'datagrid view
Try
    ds.Reset()
    da.Fill(ds, "proto")
    DataGridView1.DataSource = ds.Tables(0)
Catch ex As Exception
    MsgBox(ex.Message)
End Try
End Sub

Private Sub RadioButton5_CheckedChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs)
    Dim y As Double
    Dim a As Double
    Dim b As Double
    Dim c As Double
    Dim obj As Object
    Dim obj2 As Object
    Dim obj3 As Object
    Dim g As Integer
    Dim y1 As Double
    Dim y2 As Double
    Dim x1 As Double
    Dim x2 As Double
    Dim m As Double
    Dim co As Double
    Dim yo As Double

    For i = 1 To MaxRows - 1

        obj = ds.Tables("proto").Rows(i).Item(4)
        obj2 =
ds.Tables("proto").Rows(i).Item(5)
        obj3 =
ds.Tables("proto").Rows(i).Item(6)

        If IsDBNull(obj) Or IsDBNull(obj2) Or
IsDBNull(obj3) Then
            g = 1
        Else
            a =
ds.Tables("proto").Rows(i).Item(4)
            b =
ds.Tables("proto").Rows(i).Item(5)
            c =
ds.Tables("proto").Rows(i).Item(6)
            y = (a + b + c) / 3

            y1 = txtTempLR.Text
            y2 = txtTempUR.Text
            x1 = (txtTcLR.Text)
            x2 = (txtTcUR.Text)

            m = (y2 - y1) / (x2 - x1)
            co = y2 - (x2 * m)

            yo = m * a + co
            ds.Tables("proto").Rows(i).Item(4) =
yo

        End If

        If g = 1 And i = MaxRows - 1 Then
            MsgBox("Not Enough Data")
        End If

    Next i
End Sub

Private Sub RadioButton4_CheckedChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs)
    Dim y As Double
    Dim a As Double
    Dim b As Double
    Dim c As Double
    Dim obj As Object
    Dim obj2 As Object

```

```

Dim obj3 As Object
Dim g As Integer
Dim y1 As Double
Dim y2 As Double
Dim x1 As Double
Dim x2 As Double
Dim m As Double
Dim co As Double
Dim x As Double
Dim yo As Double

For i = 1 To MaxRows - 1

    obj = ds.Tables("proto").Rows(i).Item(4)
    obj2 =
ds.Tables("proto").Rows(i).Item(4)
    obj3 =
ds.Tables("proto").Rows(i).Item(6)

    If IsDBNull(obj) Or IsDBNull(obj2) Or
IsDBNull(obj3) Then
        g = 1
    Else
        a =
ds.Tables("proto").Rows(i).Item(4)
        b =
ds.Tables("proto").Rows(i).Item(5)
        c =
ds.Tables("proto").Rows(i).Item(6)
        y = (a + b + c) / 3

        y1 = txtTempLR.Text
        y2 = txtTempUR.Text
        x1 = (txtTcLR.Text)
        x2 = (txtTcUR.Text)

        m = (y2 - y1) / (x2 - x1)
        co = y2 - (x2 * m)

        yo = a
        x = (yo - co) / m
        ds.Tables("proto").Rows(i).Item(4) =
x

        yo = b
        x = (yo - co) / m
        ds.Tables("proto").Rows(i).Item(5) =
x

        yo = c
        x = (yo - co) / m
        ds.Tables("proto").Rows(i).Item(6) =
x

    End If

    If g = 1 And i = MaxRows - 1 Then
        MsgBox("Not Enough Data")
    End If

    Next i
End Sub

Private Sub btnSaveINst_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnSaveINst.Click
    If txtTempLR.Text = "" Or txtTempUR.Text =
"" Or txtTcLR.Text = "" Or txtTcUR.Text = "" Then
        MsgBox("Please insert the value")
        txtTempLR.Text = 0
        txtTempUR.Text = 200
        txtTcLR.Text = 4
        txtTcUR.Text = 20
    End If
End Sub

Private Sub AxAdvA01_OnTimeout(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
AxAdvA01.OnTimeout

End Sub

Private Sub Button1_Click_1(ByVal sender As
System.Object, ByVal e As System.EventArgs)
    Timer1.Enabled = True
    Timer1.Start()
    RadioButton1.Enabled = False
    RadioButton2.Enabled = False
    RadioButton3.Enabled = False
    btnRecord.Enabled = True
    btnReset.Enabled = True
    btnStart.Enabled = False
    btnStop.Enabled = True
    btnRecord2.Enabled = True
    btnReset2.Enabled = True
    My.Forms.Voltage.Timer1.Enabled = True
    My.Forms.Voltage.Timer1.Start()

    incr = 1

    MaxRows = ds.Tables("proto").Rows.Count
    InstrumentCalibration()

End Sub

Private Sub DataGridView1_CellContentClick(ByVal
sender As System.Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs)
Handles DataGridView1.CellContentClick

End Sub

Private Sub AutoKeyIn()
    MaxRows = ds.Tables("proto").Rows.Count

    '1st run
    If RadioButton1.Checked And
ds.Tables("proto").Rows(1).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(1).Item(4) =
Math.Round(tempIns, 5)
    ElseIf RadioButton1.Checked And
ds.Tables("proto").Rows(2).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(2).Item(4) =
Math.Round(tempIns, 5)
    ElseIf RadioButton1.Checked And
ds.Tables("proto").Rows(3).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(3).Item(4) =
Math.Round(tempIns, 5)
    ElseIf RadioButton1.Checked And
ds.Tables("proto").Rows(4).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(4).Item(4) =
Math.Round(tempIns, 5)
    ElseIf RadioButton1.Checked And
ds.Tables("proto").Rows(5).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(5).Item(4) =
Math.Round(tempIns, 5)
    ElseIf RadioButton1.Checked And
ds.Tables("proto").Rows(1).Item(2) =
Math.Round(tempIns, 5)
        Timer1.Stop()
        Timer1.Enabled = False
        RadioButton1.Enabled = True
        RadioButton2.Enabled = True
        RadioButton3.Enabled = True
        btnRecord.Enabled = False
        btnReset.Enabled = False
        btnStart.Enabled = True
        btnStop.Enabled = False
        btnRecord2.Enabled = False
        btnReset2.Enabled = False
        My.Forms.Voltage.Timer1.Enabled = False
        My.Forms.Voltage.Timer1.Stop()
        MsgBox("Data is recorded in 1st sample")
    End If

    '2nd run
    If RadioButton2.Checked And
ds.Tables("proto").Rows(1).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(1).Item(5) =
Math.Round(tempIns, 5)

```

```

        ElseIf RadioButton2.Checked And
ds.Tables("proto").Rows(2).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(2).Item(5) =
Math.Round(tempIns, 5)
        ElseIf RadioButton2.Checked And
ds.Tables("proto").Rows(3).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(3).Item(5) =
Math.Round(tempIns, 5)
        ElseIf RadioButton2.Checked And
ds.Tables("proto").Rows(4).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(4).Item(5) =
Math.Round(tempIns, 5)
        ElseIf RadioButton2.Checked And
ds.Tables("proto").Rows(5).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(5).Item(5) =
Math.Round(tempIns, 5)
        Timer1.Stop()
        Timer1.Enabled = False
        RadioButton1.Enabled = True
        RadioButton2.Enabled = True
        RadioButton3.Enabled = True
        btnRecord.Enabled = False
        btnReset.Enabled = False
        btnStart.Enabled = True
        btnStop.Enabled = False
        btnRecord2.Enabled = False
        btnReset2.Enabled = False
        My.Forms.Voltage.Timer1.Enabled = False
        My.Forms.Voltage.Timer1.Stop()
        MsgBox("Data is recorded in 2nd sample")

    End If
    '3rd run
    If RadioButton3.Checked And
ds.Tables("proto").Rows(1).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(1).Item(6) =
Math.Round(tempIns, 5)
        ElseIf RadioButton3.Checked And
ds.Tables("proto").Rows(2).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(2).Item(6) =
Math.Round(tempIns, 5)
        ElseIf RadioButton3.Checked And
ds.Tables("proto").Rows(3).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(3).Item(6) =
Math.Round(tempIns, 5)
        ElseIf RadioButton3.Checked And
ds.Tables("proto").Rows(4).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(4).Item(6) =
Math.Round(tempIns, 5)
        ElseIf RadioButton3.Checked And
ds.Tables("proto").Rows(5).Item(2) =
Math.Round(tempIns, 1) Then
        ds.Tables("proto").Rows(5).Item(6) =
Math.Round(tempIns, 5)
        Timer1.Stop()
        Timer1.Enabled = False
        RadioButton1.Enabled = True
        RadioButton2.Enabled = True
        RadioButton3.Enabled = True
        btnRecord.Enabled = False
        btnReset.Enabled = False
        btnStart.Enabled = True
        btnStop.Enabled = False
        btnRecord2.Enabled = False
        btnReset2.Enabled = False
        My.Forms.Voltage.Timer1.Enabled = False
        My.Forms.Voltage.Timer1.Stop()
        MsgBox("Data is recorded in 3rd sample")
    End If
End Sub
Private Sub MeanAuto()
    Dim a, b, c, d, e, f, g, h, i, j, k, l, m,
n, o As Double
    Dim mean1, mean2, mean3, mean4, mean5 As
Double
    a = ds.Tables("proto").Rows(1).Item(4)
    b = ds.Tables("proto").Rows(1).Item(5)
    c = ds.Tables("proto").Rows(1).Item(6)
    d = ds.Tables("proto").Rows(2).Item(4)
    e = ds.Tables("proto").Rows(2).Item(5)
    f = ds.Tables("proto").Rows(2).Item(6)
    g = ds.Tables("proto").Rows(3).Item(4)
    h = ds.Tables("proto").Rows(3).Item(5)
    i = ds.Tables("proto").Rows(3).Item(6)
    j = ds.Tables("proto").Rows(4).Item(4)
    k = ds.Tables("proto").Rows(4).Item(5)
    l = ds.Tables("proto").Rows(4).Item(6)
    m = ds.Tables("proto").Rows(5).Item(4)
    n = ds.Tables("proto").Rows(5).Item(5)
    o = ds.Tables("proto").Rows(5).Item(6)
    If RadioButton3.Checked And Timer1.Enabled =
False Then
        mean1 = (a + b + c) / 3
        ds.Tables("proto").Rows(1).Item(7) =
Math.Round(mean1, 4)
        mean2 = (d + e + f) / 3
        ds.Tables("proto").Rows(2).Item(7) =
Math.Round(mean2, 4)
        mean3 = (g + h + i) / 3
        ds.Tables("proto").Rows(3).Item(7) =
Math.Round(mean3, 4)
        mean4 = (j + k + l) / 3
        ds.Tables("proto").Rows(4).Item(7) =
Math.Round(mean4, 4)
        mean5 = (m + n + o) / 3
        ds.Tables("proto").Rows(5).Item(7) =
Math.Round(mean5, 4)
    End If
End Sub
Private Sub StdAuto()
    Dim a, b, c, d, e, f, g, h, i, j, k, l, m,
n, o As Double
    Dim mean1, mean2, mean3, mean4, mean5 As
Double
    Dim std1, std2, std3, std4, std5 As Double
    a = ds.Tables("proto").Rows(1).Item(4)
    b = ds.Tables("proto").Rows(1).Item(5)
    c = ds.Tables("proto").Rows(1).Item(6)
    d = ds.Tables("proto").Rows(2).Item(4)
    e = ds.Tables("proto").Rows(2).Item(5)
    f = ds.Tables("proto").Rows(2).Item(6)
    g = ds.Tables("proto").Rows(3).Item(4)
    h = ds.Tables("proto").Rows(3).Item(5)
    i = ds.Tables("proto").Rows(3).Item(6)
    j = ds.Tables("proto").Rows(4).Item(4)
    k = ds.Tables("proto").Rows(4).Item(5)
    l = ds.Tables("proto").Rows(4).Item(6)
    m = ds.Tables("proto").Rows(5).Item(4)
    n = ds.Tables("proto").Rows(5).Item(5)
    o = ds.Tables("proto").Rows(5).Item(6)
    mean1 = ds.Tables("proto").Rows(1).Item(7)
    mean2 = ds.Tables("proto").Rows(2).Item(7)
    mean3 = ds.Tables("proto").Rows(3).Item(7)
    mean4 = ds.Tables("proto").Rows(4).Item(7)
    mean5 = ds.Tables("proto").Rows(5).Item(7)
    If RadioButton3.Checked And Timer1.Enabled =
False Then
        std1 = Math.Sqrt(((a - mean1) ^ 2 + (b -
mean1) ^ 2 + (c - mean1) ^ 2) / 2)
        ds.Tables("proto").Rows(1).Item(8) =
Math.Round(std1, 4)
        std2 = Math.Sqrt(((d - mean2) ^ 2 + (e -
mean2) ^ 2 + (f - mean2) ^ 2) / 2)
        ds.Tables("proto").Rows(2).Item(8) =
Math.Round(std2, 4)
        std3 = Math.Sqrt(((g - mean3) ^ 2 + (h -
mean3) ^ 2 + (i - mean3) ^ 2) / 2)
        ds.Tables("proto").Rows(3).Item(8) =
Math.Round(std3, 4)
        std4 = Math.Sqrt(((j - mean4) ^ 2 + (k -
mean4) ^ 2 + (l - mean4) ^ 2) / 2)

```

```

        ds.Tables("proto").Rows(4).Item(8) =
Math.Round(std4, 4)
        std5 = Math.Sqrt(((m - mean5) ^ 2 + (n -
mean5) ^ 2 + (o - mean5) ^ 2) / 2)
        ds.Tables("proto").Rows(5).Item(8) =
Math.Round(std5, 4)
        End If

    End Sub
    Private Sub ErrorAuto()
        Dim mean1, mean2, mean3, mean4, mean5 As
Double
        Dim error1, error2, error3, error4, error5
As Double
        Dim a, b, c, d, e As Double
        mean1 = ds.Tables("proto").Rows(1).Item(7)
        mean2 = ds.Tables("proto").Rows(2).Item(7)
        mean3 = ds.Tables("proto").Rows(3).Item(7)
        mean4 = ds.Tables("proto").Rows(4).Item(7)
        mean5 = ds.Tables("proto").Rows(5).Item(7)
        a = ds.Tables("proto").Rows(1).Item(2)
        b = ds.Tables("proto").Rows(2).Item(2)
        c = ds.Tables("proto").Rows(3).Item(2)
        d = ds.Tables("proto").Rows(4).Item(2)
        e = ds.Tables("proto").Rows(5).Item(2)

        If RadioButton3.Checked And Timer1.Enabled =
False Then
            error1 = ((mean1 - a) / a) * 100

            ds.Tables("proto").Rows(1).Item(9) =
Math.Round(error1, 3)
            error2 = ((mean2 - b) / b) * 100
            ds.Tables("proto").Rows(2).Item(9) =
Math.Round(error2, 3)
            error3 = ((mean3 - c) / c) * 100
            ds.Tables("proto").Rows(3).Item(9) =
Math.Round(error3, 3)
            error4 = ((mean4 - d) / d) * 100
            ds.Tables("proto").Rows(4).Item(9) =
Math.Round(error4, 3)
            error5 = ((mean5 - e) / e) * 100
            ds.Tables("proto").Rows(5).Item(9) =
Math.Round(error5, 3)
            End If
        End Sub

        Private Sub Button2_Click_1(ByVal sender As
System.Object, ByVal e As System.EventArgs)

            End Sub

        Private Sub Button1_Click_2(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
            ds.Tables("proto").Rows(6).Item(9) = ""
        End Sub
    End Class

```

APPENDIX B
STUDENT'S T-DISTRIBUTION TABLE

Value of $t_p(v)$ from the t-distribution for degree of freedom γ that defines an Interval - $t_p(v)$ to $+t_p(v)$ that encompasses the fraction p of the distribution.

Degree of Freedom γ	Fraction p in percent					
	68.27*	90.00	95.00	95.45	99.00	99.73*
1	1.84	6.31	12.71	13.97	63.66	235.8
2	1.32	2.92	4.30	4.53	9.92	19.21
3	1.20	2.35	3.18	3.31	5.84	9.22
4	1.14	2.13	2.78	2.87	4.606	6.62
5	1.11	2.02	2.57	2.65	4.03	5.51
6	1.09	1.94	2.45	2.52	3.71	4.90
7	1.08	1.89	2.36	2.43	3.50	4.53
8	1.07	1.86	2.31	2.37	3.36	4.28
9	1.06	1.83	2.26	2.32	3.25	4.09
10	1.05	1.81	2.23	2.28	3.17	3.96
11	1.05	1.80	2.20	2.25	3.11	3.85
12	1.04	1.78	2.18	2.23	3.05	3.76
13	1.04	1.77	2.17	2.21	3.01	3.69
14	1.04	1.76	2.14	2.20	2.98	3.64
15	1.03	1.75	2.13	2.18	2.95	3.59
16	1.03	1.75	2.12	2.17	2.92	3.54
17	1.03	1.74	2.11	2.16	2.90	3.51
18	1.03	1.73	2.10	2.15	2.88	3.48
19	1.03	1.73	2.09	2.14	2.86	3.45
20	1.03	1.72	2.09	2.13	2.85	3.42
25	1.02	1.71	2.06	2.11	2.79	3.33
30	1.02	1.70	2.04	2.09	2.75	3.27
35	1.01	1.70	2.03	2.07	2.72	3.23
40	1.01	1.68	2.02	2.06	2.70	3.20
45	1.01	1.68	2.01	2.06	2.69	3.18
50	1.01	1.68	2.01	2.05	2.68	3.16
100	1.005	1.660	1.984	2.025	2.626	3.077
	1.000	1.645	1.96	2.000	2.576	3.000

APPENDIX C TEMPERATURE TRANSMITTER

YTA Input/Output Selection Guide

There's a Yokogawa solution for virtually every process application.

■ currently available ■ coming soon

Application	Standard	Input Range	YTA50	YTA70	YTA110	YTA310	YTA320
RTD	Pt100	IEC751	-200 to 850°C -328 to 1562°F				
	Pt200	IEC751	-200 to 850°C -328 to 1562°F				
	Pt500	IEC751	-200 to 850°C -328 to 1562°F				
	JPt100	JIS C1604	-200 to 850°C -328 to 932°F				
	Ni120	120 ohm nickel 0.00672 coefficient	-70 to 320°C -94 to 608°F				
	Cu	9.042 ohm copper 0.0042 coefficient	-50 to 250°C -58 to 482°F				
	Sensor Matching						
Thermocouple	B	IEC584	100 to 1820°C 212 to 3308°F				
	E	IEC584	-200 to 1000°C -328 to 1832°F				
	J	IEC584	-200 to 1200°C -328 to 2192°F				
	K	IEC584	-200 to 1372°C -328 to 2500°F				
	L	DIN43710	-200 to 900°C -328 to 1652°F				
	N	IEC584	-200 to 1300°C -328 to 2372°F				
	R	IEC584	-50 to 1768°C -58 to 3214°F				
	S	IEC584	-50 to 1768°C -58 to 3214°F				
	T	IEC584	-200 to 400°C -328 to 752°F				
	U	DIN43710	0 to 2300°C 32 to 4172°F				
	W3	ASTM E988	0 to 2300°C 32 to 4172°F				
	W5	ASTM E988	0 to 2300°C 32 to 4172°F				
	mV		-10 to 800mV				
	Ohm		0 to 2000 ohms				
Single Sensor							
Dual Sensor							
Automatic Sensor Backup							
Head Mounted							
Digital Indicator							
BRAIN							
HART							
FIELDBUS							

SPECIFICATIONS

Temperature Transmitter Selection

Model	Ambient Temperature Limits	A/D Accuracy (100 ohm RTD)	D/A Accuracy (100 ohm RTD)	Ambient Temperature Effect	Input Types
Low Cost					
YTA70 YTA50	-40 to 185° F (-40 to 85° C)	+/- 0.2% of calibrated Span		+/- 0.2% of calibrated Span per 10° C change	2 RTD, 12 T/C
Mid Range					
YTA110	-40 to 185° F (-40 to 85° C)	+/- 0.1% of calibrated Span		+/- 0.2% of calibrated Span per 10° C change	6 RTD, 12 T/C, Millivolt, Ohm
High Performance					
YTA310	-40 to 185° F (-40 to 85° C)	+/- 0.14° C	+/- 0.02% Span	+/- 0.00726° C per 10° C change (Pt100, 200° C range)	6 RTD, 12 T/C Millivolt, Ohm,
MultiVariate					
YTA320	-40 to 185° F (-40 to 85° C)	+/- 0.14° C	+/- 0.02% span	+/- 0.0726° C per 10° C change (Pt100, 200° C range)	6 RTD, 12 T/C, Millivolt, Ohm, Differential



YOKOGAWA

YOKOGAWA ELECTRIC CORPORATION

World Headquarters

9-32, Nakacho 2-chome, Musashino-shi, Tokyo 180-8750, JAPAN
Tel: 81-422-52-5690 Fax: 81-422-52-2018
www.yokogawa.co.jp

YOKOGAWA CORPORATION OF AMERICA

2 Dart Road, Newnan, GA 30265, U.S.A.
Tel: 770-254-0400 Fax: 770-254-0928
www.yca.com

YOKOGAWA AMERICA DO SUL

Avenida Jurua, 149-AlphaVil, 06455-010, Barueri
Sao Paulo, Brazil
Tel: 55-11-7295-1433 Fax: 55-11-7295-1329
www.yca.yokogawa.com.br

YOKOGAWA EUROPE B.V.

P.O. Box 163, 3800 AD Amersfoort, Vanadiumweg 11, 3812 PX Amersfoort
THE NETHERLANDS
Tel: 31-33-4641611 Fax: 31-33-4631202
www.yokogawa-europe.com

YOKOGAWA ENGINEERING ASIA PTE. LTD.

5 Bedok South Road, Singapore 469270, SINGAPORE
Tel: 65-241-9933 Fax: 65-241-2606
www.yokogawa.com.sg

Represented by:

APPENDIX D

ADVANTECH USB-4716 DAQ CARD SPECIFICATION

A.1 Analog Input

Channels	16-ch single-ended/ 8-ch differential						
Resolution	16 bits	FIFO Size	1024 Samples				
Sampling Rate	200 kS/s						
Input Range and Gain List	Gain	0.5	1	2	4	8	
	Gain Code	4	0	1	2	3	
	Bipolar (V)	±10	±5	±2.5	±1.25	±0.625	
	Unipolar (V)	N/A	0~10	0~5	0~2.5	0~1.25	
Drift	Gain	0.5	1	2	4	8	
	Zero ($\mu\text{V}/^\circ\text{C}$)	±30					
	Gain ($\text{ppm}/^\circ\text{C}$)	30	30	30	30	30	
Small Signal Bandwidth for PGA	Gain	0.5	1	2	4	8	
	Bandwidth (MHz)	1.1	1.1	1.1	1.1	1.1	
Input Protection	30 V max.						
Input Impedance	1GW						
Input Comm. Mode Voltage	11V						
Accuracy	DC	INLE	1LSB				
		DNLE	3LSB				
		Gain	0.5	1	2	4	8
		Gain Error (% of FSR)	0.015	0.03	0.03	0.05	0.1
	AC	SINAD	83 dB				
		THD	-88 dB				
		ENOB	13.5 bit				

A.2 Analog Output

Channels	2		
Resolution	16 bits	FIFO Size	N/A
Throughput	2 kHz		
Operating Mode	Single output		
Output Range	0~5, 0~10, ± 5 , ± 10 V		
Accuracy	DC	INLE	± 2 LSB
		DNLE	± 1 LSB
Dynamic Performance	Slew Rate	0.125 V/ μ s	
	Settling Time	150 μ s (to $\pm 1/2$ LSB of FSB)	
Driving Capability	5 mA		
Output Impedance	0.1W max.		

A.3 Non-Isolated Digital Input/Output

Input Channels	8 Non-Isolation TTL	
Input Voltage	Low	0.0 Vdc (Min) / 1.0Vdc (Max)
	High	2.0 Vdc (Min) / 5.0Vdc (Max)
Output Channels	8 Non-Isolation TTL	
Output Voltage	Low	0.4 Vdc / -6mA (Sink)
	High	2.4 Vdc / 6mA (Source)

A.4 Counter

Channels	1		
Resolution	32-bit software base	Capability	TTL level
Input Frequency	1 kHz max.		
Clock Input	Low	0.0 Vdc (Min) / 1.0 Vdc (Max)	
	High	2.0 Vdc (Min) / 5.0 Vdc (Max)	
Gate Input	Low	0.0 Vdc (Min) / 1.0 Vdc (Max)	
	High	2.0 Vdc (Min) / 5.0 Vdc (Max)	

A.5 General

I/O Connector Type	Removable 10-pin screw terminal x 5	
Dimensions	132 X 80 X 32 mm (5.2" X 3.2" X 1.3")	
Power Consumption	360 mA @ +5.0V Typical 450 mA @ +5.0 V max.	
Temperature	Operation	0~60° C (32~140° F) (refer to IEC 68-2-1, 2)
	Storage	-20~70° C (-4~158° F)
Relative Humidity	5~ 95 % RH non-condensing (refer to IEC 68-2-1, 2)	