PERFORMANCE MONITORING AND TRACKING PROTOTYPE FOR SOLAR PHOTOVOLTAIC (PV) INTEGRATED RENEWABLE ENERGY SYSTEM (PEMANTAU)

MOHD NAIM 'AFIFI BIN ISHAK

ELECTRICAL AND ELECTRONIC ENGINEERING
UNIVERSITI TEKNOLOGI PETRONAS

SEPTEMBER 2012

PERFORMANCE MONITORING AND TRACKING PROTOTYPE FOR SOLAR PHOTOVOLTAIC (PV) INTEGRATED RENEWABLE ENERGY SYSTEM (PEMANTAU)

by

Mohd Naim 'Afifi Bin Ishak

Dissertation submitted in partial fulfillment of the requirements for the

BACHELOR OF ENGINEERING (Hons) in

ELECTRICAL AND ELECTRONIC ENGINEERING

Dr. Mohd Zuki Bin Yusoff (Supervisor)

Assoc. Prof. Dr. Balbir Singh Mahinder Singh (Co-Supervisor)

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

September 2012

CERTIFICATION OF APPROVAL

PERFORMANCE MONITORING AND TRACKING PROTOTYPE FOR SOLAR PHOTOVOLTAIC (PV) INTEGRATED RENEWABLE ENERGY SYSTEM (PEMANTAU)

by

Mohd Naim 'Afifi Bin Ishak

A project dissertation submitted to the

Electrical and Electronic Programme

Universiti Teknologi PETRONAS

in partial fulfillment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(ELECTRICAL AND ELECTRONIC ENGINEERING)

Approved by,

____
(Dr. Mohd Zuki B. Yusoff)

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

September 2012

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

September 2012

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

____

(MOHD NAIM 'AFIFI BIN ISHAK)

ACKNOWLEDGMENT

Firstly, I would like to thank the Almighty God for giving me the strength and time in lesson of completion of my Final Year Project (FYP) in Universiti Teknologi PETRONAS. I would like to express my most gratitude to my supervisor, Dr. Mohd Zuki B. Yusoff for keep believing in me, for always giving me supports, with first class guidance, for encouraging deeper perseverance and spending precious time throughout various stage of the project completion. I would like also to express my gratitude to my co-supervisor, Assoc. Prof. Dr. Balbir Singh for his supervision, and supportive advices throughout the project period.

In addition, my thanks also goes to my fellow friends and individuals who gave lots of encouragement, and this will always be a pleasant memory throughout my life. Last but not least, I would like to thank my family for their love and supports while I was facing hardship completing the project. With full cooperation and encouragement from all above, I have successfully completed the project. Thank you and may our relationship bonds forever.

Thank you in advance,

....

MOHD NAIM 'AFIFI BIN ISHAK

Electrical and Electronic Engineering

Universiti Teknologi PETRONAS

ABSTRACT

The purpose of this project is to develop a monitoring and tracking system for renewable energy generation system. The system is called as PEMANTAU, which stands for Performance Monitoring and Tracking Prototype for Integrated Renewable Energy System, consisting of tools for measurements, analysis, and controls of the renewable electricity generating system. PEMANTAU is an important tool for monitoring the performance of solar electricity generating system, for optimum operation. The idea of PEMANTAU is to provide capability to capture important data for analysis which can be used for optimizing the renewable electricity generating system. It is important now to realize that optimization and improvements to the renewable energy system efficiency will bring the change to the nation where, energy consumption today is too dependent on the fossil type of energy. This project is also to support the strategic plan framework of the 10th Malaysia Plan that emphasizes on the importance of using renewable energy to meet Malaysia’s growing energy demand and to reduce the nation's reliance and utilization of fossil fuel for power generation. The earlier phase of PEMANTAU will support a solar-based electricity generating system. This will serve as an early prototype to build the platform for the renewable electricity system. Solar electricity is proved to be an ideal source of potential future electricity. Thus a quick method using simulation is needed to model the impact of solar energy where it can be used to help distribution planners to perform the necessary research and improvements. This report will demonstrate and document all the functionalities and explain methods of the real time monitoring system that models resulting from output to the end user. The GUI-based is designed to make the representation of data more user-friendly. It acquires the measured data which have been transmitted wirelessly. Overall, the objectives of this project have been fully achieved, whereby the PEMANTAU system has been successfully designed and tested. PEMANTAU can be further improved by extending it for other renewable energy based systems.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ASS	Analog Sensor Source
ADC	Analog Digital Conversion
FF	Fill factor
GUI	Graphical User Interface
IEEE	Institute of Electrical and Electronics Engineers
I-V	Current and Voltage Curve
HM	Host Monitoring
MPP	Maximum Power Point
PC	Personal computer
PV	Photovoltaic
PS	Processing System
PEMANTAU	Performance Monitoring And Tracking Prototype For Solar Photovoltaic (PV)Renewable Energy System
ReTOUCH	The Touch Screen with Sensor Module System
RTOS	Real Time Operating System
SDK	Software Development Kit
UART	Universal Asynchronous Receiver/Transmitter

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND STUDIES

Data monitoring and tracking is vital for renewable energy power generation system especially for the solar power generation. The electricity is generated from photovoltaic panel or PV where it captures energy from radiance of the sun. Thus it is important to measure every bits of energy, electricity and the power generation surrounding condition. The information must be monitored and recorded on every second. It will be as guidelines to review PV solar power generation such as to oversaw the performance before it fall below expectation. Tools provided for the user to have access on accurate measurement like the power flow testing to evaluates the power quality and produce kinds of useful data.

A normal solar power generation consist of PV module, inverters, batteries and charge controller, can be applied to yield energy from the sun and produce amounts of electricity. But it happens to be ineffective when the fluctuation of energy due to the inconsistency of the light received to the PV panel. For example even with the hazy cloud on the sky and tiny spot of dust could covered on the PV panel, it will result lower electricity to the output. Even the wind, humidity surrounding and temperatures are affect the consistency of the PV panel. The charge controller today can achieve good control on the electricity charge, but only manage the electricity not the others efficiency parameters. To make it solar power generation and other renewable energy system to be reliable supply, it need to integrated with the intelligent monitoring system whereas consist of measurement and control.

The project title is called as Performance Monitoring And Tracking Prototype For Solar Photovoltaic (PV) Integrated Renewable Energy System or PEMANTAU which are basically a built-in system or device that can adopted with any renewable energy source to provide real time data monitoring. For example, a single PV solar power generation system or to have a hybrid source of renewable energy system, such like wind generator and together with the PV solar power generation system. The PEMANTAU will be as a system that have flexibility to work together with other the renewable energy system. The project in a relatively large in term of scope development, thus due to time limitation and constraints, the project will only focus on the PV solar power generation system.

PV solar power generation system can be separated into a grid-connected systems or a stand-alone systems. Designed to captures solar energy which is ecological friendly, with no carbon emissions (CO^2) and in returns helps the nation energy demand. Today the PV solar power generation system is beginning on stage of the consumer to implemented as a standard system.

To monitor the PV solar power generation system and developed a data acquisition system that retrieve, record, store and display information over instance, by means in real time; a list of parameters is included and measured accurately from the analog sources. For examples DC voltage, DC current, AC voltage, AC currents, DC battery, loads, light, solar radiation, temperatures and more. This to proved a system can do lots of measurement and to make the user to understand the characteristic of the conversion and verifying the data.

In additions, the monitoring module can also control the system operation remotely. The system supposed to record operational events and display data on the internet; this is called as cloud computing where the data can be access anywhere. From here, the control system can be designed to located remotely far from the PV solar power generation system.

This report will describes more in-depth details about the data monitoring system for the PV solar power generation system. It documented methods of development for a new design, a new topology of circuit and the development of software whereas features lots on techniques of measuring, monitoring, analyzing and simulating. When the project is completed, the project can be extend to improved on any matter. The name of PEMANTAU will became as a standard for the monitoring tools. It is also targeted not to be specific to any kind of PV solar power generation system and also can be adopted to other renewable power generation. The system is suitable for end user, researcher, education practices and production industry.

Although there are already existing systems or devices of this kind monitoring and tracking system commercially available. But, it is utmost cases do not provide in-depth on the visualization of the data, which is one of important key to provide more details on the renewable energy system. The aims for PEMANTAU is to improved the traditional renewable energy system data monitoring and especially for the PV solar power generation system.

1.2 PROBLEM STATEMENT

There is no better method in predicting PV panel performance than exposing it to real condition on the field. Environmental conditions, such as changes in irradiance, temperature, and other external factors where it can affect a lot on PV cells.

Although laboratory test can determine PV cell characteristics. The monitoring system can boost confidence to the end user for product reliability and overall performance. A fast, accurate and carefully synchronized measurement approach is required to obtain meaningful data in the ever-changing conditions like as described. By obtaining the analog data for examples voltage, current, temperature and quality of light. We can summarize and simulate operation on many different with optimum variations which is useful for been used in research and development. It also can helped during planning or to maintaining the PV solar generation system.

Others issue could be is how to measure on large scale of PV panels. It will be a huge disaster if the electricity is not in the best condition on the production time. This will be a total lost for the cost. A perfect to handle this is to developed excellent monitoring and analysis that can help reveal data, brief issues and also expected to bring cost savings opportunities. There are numbers of parameters taken into considerations such like inverter, the point range on DC side (input and output) and also for batteries and the charge controller.

1.3 OBJECTIVES AND SCOPE OF STUDY

The project is known as PEMANTAU - an abbreviation for Performance Monitoring And Tracking Prototype For Solar Photovoltaic (PV) Integrated Renewable Energy System. The main idea of this project is to developed a system which will operate in real-time and to represents the entire PV solar generation system on single devices and communicate to data center. By default the system will communicate with the integrated controller that supports together with the PV panel.

To overcome the challenges, a list of objectives are identified. The project will aim on building a prototype circuit, then to the develop a software module for simulation and analysis program. This project effort encompasses the following activities;

- To verify the operating value of the output circuit of the PV panel.
- To measure and verify the overall efficiency and conversion of the PV panel.
- To simulate the I-V curve under the actual environment with real time simulation.
- To provide performance analysis in GUI for performance that yield in real time.
- To develop cloud computing network, real-time data will connect to the Internet by integrating with a dedicated database, that can be access anywhere.

In summary the PEMANTAU project effort is to focus on enhancing the reliability of the measurement equipment in PV solar generation systems. The PEMANTAU is monitoring system that have lots of features, flexibility and feedback, afterwards the system can be enhanced not just provide data but it can also evaluate new design and improved PV panels, materials, and other processes.

CHAPTER 2

LITERATURE REVIEWS

2.1 INTRODUCTION

With growing concerns for the upcoming future and security of the world's energy supply; renewable resources such as solar power are becoming increasingly importance. Various solar technologies have through millennia of human history. However, practically, the photovoltaic technology happen not having so much change since the history of the beginning of photovoltaic technology.

Soon, there will be crisis, facing of dilemma of cost producing energy and the depletion of the fossil resource is still uncertain. Thus this is a good time to expand the research and come out the new plan to improve the solar renewable energy system. And, one day renewable energy such like solar power will be a better platform for power generation system, and it will be widely available for everyone to savor the benefit.

The energy comes from the Sun; it is renewable, infinite and has zero emissions. It has a huge potential on bringing the good amount of electricity, thus it is important to study on the parameters. This literature review will cover the theories of energy of the sun, solar cells technology and the theoretical studies of energy drained from the solar cells.

2.2 ENERGY FROM THE SUN

The sun is a gaseous body and is composed mostly of hydrogen, with some helium and traces of heavier elements. The gases swirl and flow under the pressure of magnetic fields, gravity and heat energy[1]. Gravity also causes intense pressure and heat at the core, which initiates nuclear fusion reactions. The sun fuses hydrogen into helium at its core and pushes the resulting energy outward. The energy then travels to the earth's photosphere, where it escapes into space in the form source of light and heat radiation. Radiation is energy that expands outward from the source in the form of waves or particles[2].

Figure 1: Sun during cloudy day

The energy that come from the sun, is varies in terms of the solar radiation is scattering the atmosphere. The total solar energy obtained from the sun is approximately about 3,850,000 exajoules (EJ) per year[3]. This solar energy is regarded as of solar radiation that is scattered around or reflected on the earth's atmospheric surface. And this is called the "Direct Insolation", where it comes from the sun to the higher layer of the earth's atmosphere[4]. Indeed the solar are varied due to the fixed distance; but still the concentration of light depends on the earth's daylight hours and it is depending on solar elevation angle[5].

About 30% of the light is reflected back to space and the others are engaged by clouds, and other surfaces such as oceans and land [6]. The solar lights that arrive onto the earth's surface are considered as visible spectrum, it also released small portion of temperature depending on the concentration of the light[7]. For example, an oceans is containing the evaporated water which causes movements of the atmosphere. The heat the water from the oceans is becoming one of the cycle of condensation. That means, the temperature is low when the air is reaching a high altitude; as a result this water are condensed into clouds.

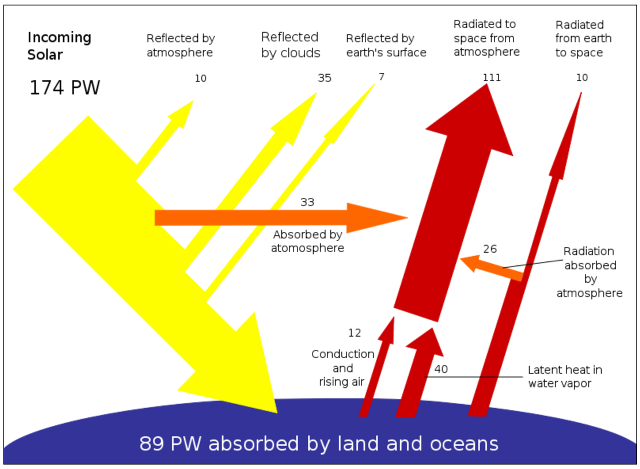


Figure 2: Half of incoming solar energy reaches the Earth's surface.

Atmospheric phenomena such as wind, storms and even the cyclone are also amplify the air to be vaporized from the condensed water. On average the temperature of the surface is kept to 14 ^{o}C, while the rest of the temperature is absorbed by the ocean and land masses [8].

2.3 SOLAR IRRADIANCE

Solar irradiance is expressed in units of watts per square meter (W/m^2) or kilowatts per square meter (kW/m^2). The irradiance is measured with respect to the area due to solar radiation reflection on the unit surface[9].

Solar irradiance is used to estimate the performance of solar energy system output at a specified point in time, or the peak output for solar energy equipment.

The inverse square law is physical law that indicate that the amount of radiation is proportional to the inverse of the square of the distance from the source[10].

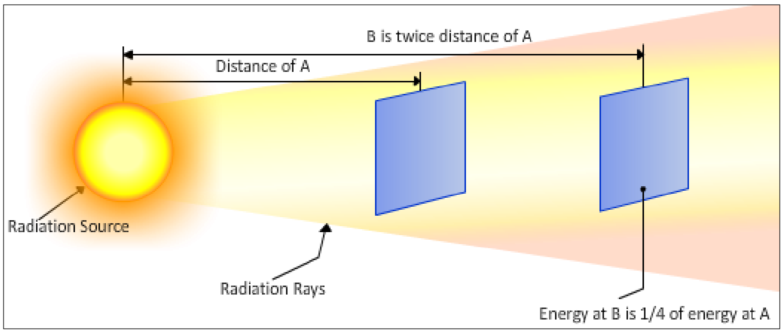


Figure 3: Radiation energy is reduced in proportion to inverse square of the distance from the source

The amount of solar energy is accumulate on area over time. A period that represents an a hour, a day, a month or a year. The higher irradiance will result in greater energy[11]. Solar irradiance begin from zero at night hour. It then increases during the sun rise, reaching at noon and it decreases during the sun fall[12]. Show in Figure 4, is a plot of solar irradiance versus time; the solar irradiation is equivalent to the area under the irradiance curve.

Solar irradiance can be calculated by applying the formula:

$$H = E \times t \qquad (1)$$

where, H is the solar irradiation (Wh/m^2); E, is the solar irradiance (W/m^2); t is time (hr).

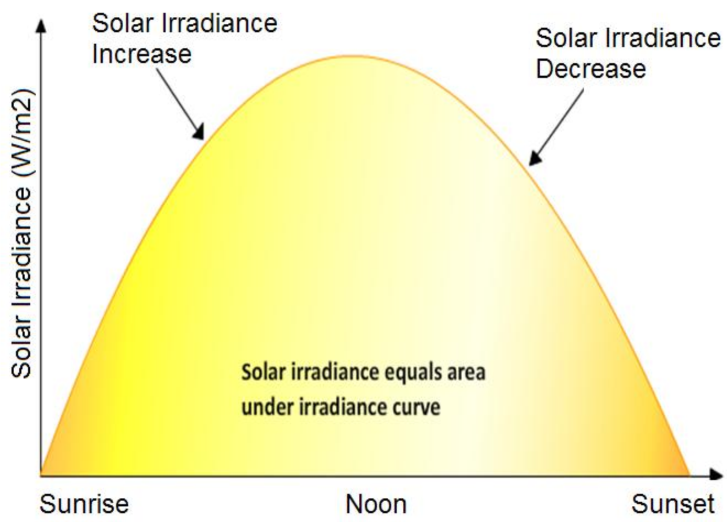


Figure 4: Solar irradiance equals the total solar irradiance over time

2.4 PHOTOVOLTAIC CELLS AND EFFECT

The component that produce electricity on the solar panel is known as a photovoltaic cells or PV. It is a technology that uses silicon properties, composed into a semiconductor material which is very unique due to effect producing electricity from the light radiance[13]. The PV panel comprises layers of wafer which consists of unique crystalline that is sensitive to the light. When the light is exposed to the PV panel, it will produce a small quantity of direct current (DC). The silicon layer of the cells consists of interconnection of element of Si in atom scales that provide activities for electrons to move. When photons of lights strike to the PV panel surface, it will giving up the free electrons, this is result the charged flow on the P-N connection and hence electricity is generated.

The process of producing electricity on a PV cells is known as the photovoltaic effect. The effect is due to the movements of the electrons that absorb light energy which is the photons on certain range of spectrum. Electromagnetic radiation is a photon that contains energy which is dependent on a specific wavelength.

The photoelectric effect was first introduced by French physicist, Edmund Bequerel, on 1839. He discovered that certain materials, turn out to have an effect of producing a small amounts of current when exposed it into the light[14]. PV cells come from particular materials that called as semiconductors or silicon, normally materialize from a raw material, that used mostly for manufacturing the computer chips. The Figure 5 illustrates the operation of a basic photovoltaic effect on PV cell.

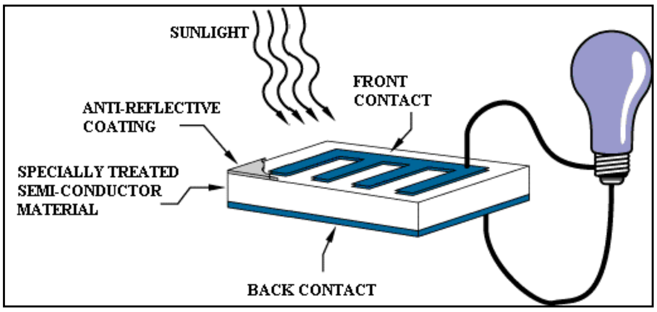


Figure 5: Operation of a basic photovoltaic cell, also called a solar cell

The cells are connected together in a sustainable formation or framework that is called a PV module. These modules are considered to provide electricity at a fixed voltage value, for example the peak of certain small module can be up to 12 volts.

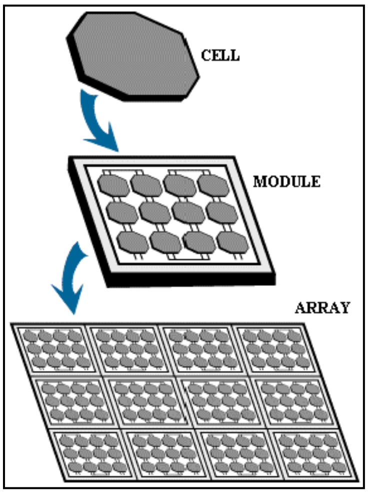


Figure 6: Multiple modules can be wired together to form an array

A PV cell consists of a layer of thin wafer consisting a P-N junction. The material of the of P-N junction semiconductor is the periphery of the bordering layer of P-type and N-type. When photons are absorbed, due to the photoelectric effect on the surface, it will produce an effect of the conductivity, whereas the electrons can move to each other's and the flow of the electron will provide potential energy of voltage.

A PV module can be wired to the multiple modules; which is called a PV array. More electricity can be gather when there is a more arrays in a single panel. The array can configured to as series or by parallel, resulting different total output, but it is also depend on the material of PV cells. [15]

2.5 PV CELLS MATERIALS

Materials for PV cells came from mixtures of semiconductor materials, the crystalline silicon is a type of PV cells commonly manufacture today. At least 99.99% pure silicon material built on for a single crystalline silicon cell [16].

Commercial PV panels that is manufactured today involves silicon wafers that fabricated into cells and then assembled into modules. Different types of silicon material have their own advantages and disadvantages in terms of cost and efficiency. There are 3 basic type of crystalline materials: polycrystalline, monocrystalline and ribbon silicon[17].

Other types of cell material like armosphous, polymer, copper and graetzel are cheaper than crystalline but the effectiveness of the cells is lower.

Table 1: Comparison of various silicon PV material

PV Cell Material	PV Module Efficiency %	Energy Density kWp/m3	Cost
Monocrystalline silicon PV	13-17		
Polycrystalline silicon PV	11-15		
Amorphous silicon PV	6-8		

2.6 CURRENT-VOLTAGE (I-V) CURVE

The performance manufacture's of PV panel can be determined by the output and the operating values. Based on characteristic point given on the manufacture specification and laboratory test. The performance of PV panel can be obtain by measuring the voltage (V) and current (I) coming from the output cells. This performance indicator is called maximum power output (MPP). The performance of the PV panel can be determined by I-V characteristic. It can shows operating point and power output.

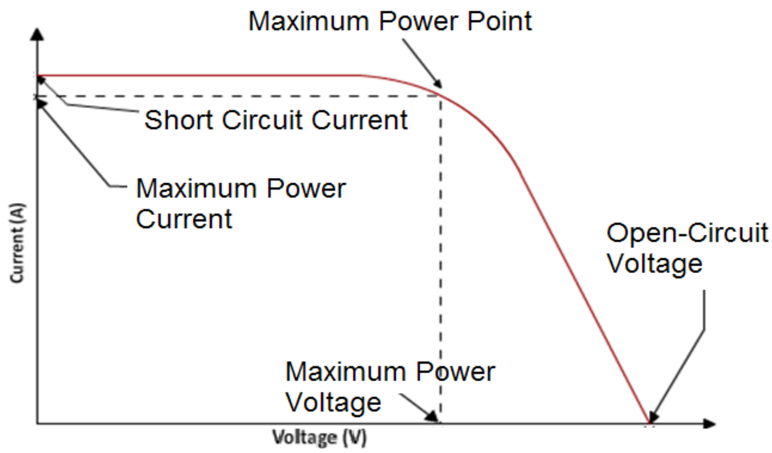


Figure 7: The current-voltage (I-V) characteristic curve

By understanding the points of an I-V curve of the PV panel, we are should be able to measure the performance. The points of parameters is from open-circuit voltage, short-circuit current, maximum power voltage, maximum power current and maximum power. Other factors for measuring performance using the I-V curve such as temperature and irradiance are meant to be together on measurement to PV panel condition.

2.6.1 Open-Circuit Voltage

The operating point for a PV panel during no current output can be measured during load connect to the output of PV panel. During the open-circuit voltage, the output power will show a zero value. To measure the open-circuit voltage, of a PV panel, firstly the PV panel must be exposed into the sunlight, then measured back across the DC voltage across the output using a voltmeter or a multi-meter.

The surrounding temperature also affects the effectiveness of PV panel. During high temperature, it will reduce the open-circuit voltage for the PV panel[18]. The open-circuit voltage is typically 0.5V to 0.6V at 25C (77F) for the crystalline silicon cells.

2.6.2 Short-Circuit Current

The short-circuit or no-load condition, happen during output power where shows result as zero value, this is because the voltage is zero during short-circuit current. Significantly, the short-circuit current can be determine by measuring the maximum current of PV panel that exposing to solar irradiance. A short-circuit current can be obtained by exposing the PV panel to the sunlight and measuring back the current with a ammeter or a multi-meter.

2.6.3 Maximum Power Point

Maximum power output can be determine on the I-V characteristic which in between the open-circuit and short-circuit. This happen during connected with the loaded or somewhat finite space of resistance. The operating point of current and voltage during maximum is called as the Maximum Power Point or MPP. MPP is used to detect the peak power. The maximum power point parameter is nominated with Wp of peak watts.

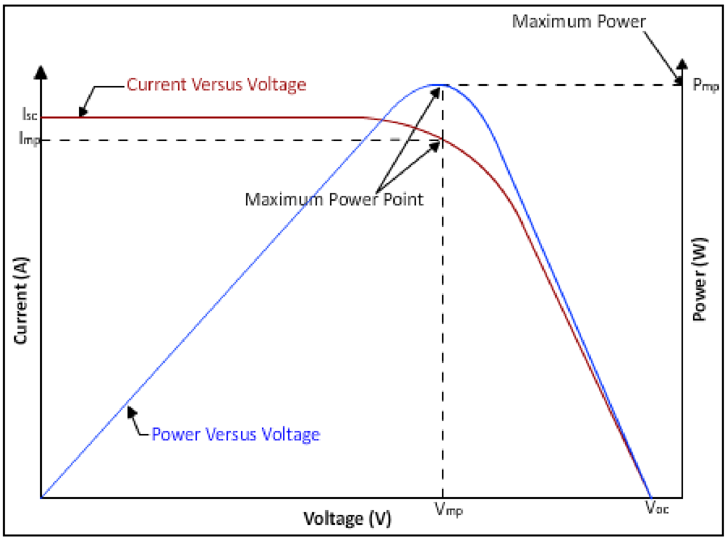


Figure 8: Power against voltage curve shows the maximum power point

MPP is consist of maximum power voltage and maximum power current presented from the I-V graph. The operating voltage where the power is maximum is the maximum power voltage and also for the maximum power current can be retrieved from the I-V graph. Maximum power point can be calculated by defined formula:

$$P_{MP} = V_{MP} \times I_{MP} \qquad (2)$$

where

$$P_{MP} = maximum\ power\ (in\ W)$$
$$V_{MP} = maximum\ power\ voltage\ (in\ V)$$
$$I_{MP} = maximum\ power\ current\ (in\ A)$$

Fill factor is the ratio of open-circuit voltage and short-circuit voltage to perform the performance quality for a PV panel. Maximum power point that are closer to the open-circuit voltage and short-circuit voltage indicating it is now the highest fill factor that are showing rectangular area inside the I-V curve.

This can expressed as percentage, below showed the formula:

$$FF = \frac{P_{MP}}{(V_{OC} \times I_{SC})} \qquad (3)$$

where

$$FF = is\ fill\ factor$$
$$P_{MP} = maximum\ power\ (in\ W)$$
$$V_{OC} = open - circuit\ voltage\ (in\ V)$$
$$I_{SC} = short - circuit\ current\ (in\ A)$$

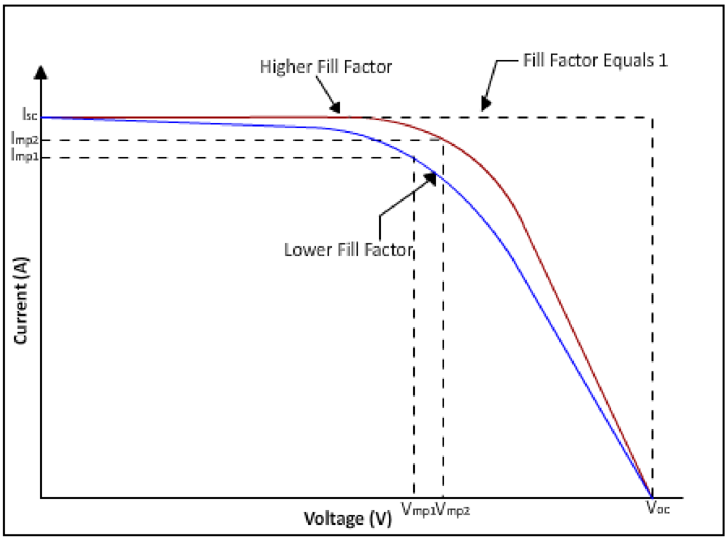


Figure 9: Fill factor represent shape for an I-V curve

PV panel can be compared by measuring the efficiency of the ratio of power output to power input and the solar irradiance is versus to the area of the PV which can be compared directly. Different PV cells technologies will shows efficiencies due for different material that been used.

$$n = \frac{P_{MP}}{(E \times A)} \qquad (4)$$

where

$$n = efficiency$$
$$P_{MP} = maximum\ power\ (in\ W)$$
$$E = solar\ irradiance\ \left(in \frac{W}{m^2}\right)$$
$$A = area\ (in\ m^2)$$

Normally PV cells will operate effectively to secure to highest maximum power points. Nevertheless, the maximum power point is continuously shifting due to changes in the of solar irradiance and cell temperature[19]. As a result, some system vigorously equal to PV panel output and to the loads. This to make sure the system can optimize the performance of PV panel. This also helped by conversion module system whereas the charge controllers.

The load to operate a PV panel can be determine by ohm's law, the measurement value can retrieve during on the device at maximum power:

$$R_{MP} = \frac{V_{MP}}{I_{MP}} \qquad (5)$$

where

$$R_{MP} = resistance\ at\ maximum\ power\ point\ (in\ \Omega)$$
$$V_{MP} = maximum\ power\ voltage\ (in\ V)$$
$$I_{MP} = maximum\ current\ (in\ A)$$

2.7 ENVIRONMENTAL CONDITIONS

The most important aspect of environment condition is the amount of sunlight absorb by the PV panels. Environmental condition such like air temperature, humidity, wind, rainfall patterns influence quantity electricity.

On the statically data, during the cloudy daylight, the PV panels stays at what is supposed to be, as the high peak solar light ray occurs at 12PM to 2PM. So at 4PM, by means evening sun, it is more less to produce the electricity. The effectiveness can be is around 8% or a 10%.

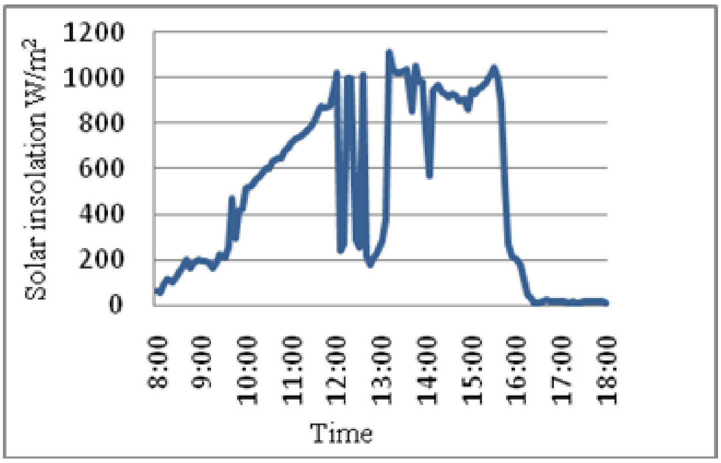


Figure 10: Solar insolation changes during day time

The more light receives by the PV panel, means, the more power will produced. On the bright sunny days, ideally PV panel can supply up to the operational of peak power. But during a day with amounts of clouds, the electricity production will be less than the average. There is cloudiness phenomenon that exists when there are group of cumulus clouds itinerant through the sunlight, and this was called as the rim of cloud effect, as the sun rays throughout holes in between the clouds, with combining the reflective light will indicate to boost electricity productivity. This can be good fluctuation but it is unpredictable. However this be noted as cautions because there is a danger during unstable source, during high peak energy received, it may result maximum voltage capacity to the battery where it may damaged the whole system. Thus to fixed this, the inverters will be allow to bring surge to the power, this will help to protect the solar PV panel and also the battery.

Figure 11: Solar panel exposed to dusty condition

Others constraint for positioning PV, like place and location for example like desert happen to have more problem on environment condition. Dust problem effect the efficiency of PV panels, because is drop due to less solar energy absorption. It's almost the same to the cloudiness effect which it blocking the light absorption to the PV panels. One example is on the first solar power plant in Abu Dhabi, it happen that the reduction in electricity production is going to 40% during season of dust storm.

2.8 TILT ANGLE

PV panels depends on the amount of light from the sun that not in the same angle, thus the PV panel must point to the directly where it get most light to the most sun. To tackle this situation, optimization to PV panel must be master where it can trust itself on tracking part to find the best angle to get the most sunlight. The following table shows the adjustment angle that can help PV solar power generation optimization. Using the 40 latitude as example, each is been compared from data that have produced by the sensor:

- latitude is below 25°, use the latitude times 0.87.
- latitude is between 25° and 50°, use the latitude, times 0.76, plus 3.1 degrees.
- latitude is above 50°

Table 2 below shows the latitudes examples. The table also explains the average insolation on PV panels over year in kWh/m^2 per day, shows the optimum value for the panel and comparisons among selected places/cities.

Table 2: Average Insolation over a year by places

Latitude	Full year angle	Average insolation on PV panel	% of optimum
0° (Quito)	0.0	6.5	72%
5° (Bogotá)	4.4	6.5	72%
10° (Caracas)	8.7	6.5	72%
15° (Dakar)	13.1	6.4	72%
20° (Mérida)	17.4	6.3	72%
25° (Key West, Taipei)	22.1	6.2	72%
30° (Houston, Cairo)	25.9	6.1	71%
35° (Albuquerque, Tokyo)	29.7	6.0	71%
40° (Denver, Madrid)	33.5	5.7	71%
45° (Minneapolis, Milano)	37.3	5.4	71%
50° (Winnipeg, Prague)	41.1	5.1	70%

2.9 SOFTWARE MODEL

Monitoring software for the PV solar generation system must be comprehensive and precision. It is must be reliable enough to keep measuring the of PV solar power generation system and determined the daily energy yields for the system.

These monitor software must allow values to be accessed and analyzed at any time. It is need to be suitable for large system, and can be configure on PV panel escalation. Thus a perfect software models based on the circuit design, networks, data measurement, analyzing and circuit are need to outline before developing it.

The microcontroller will be used for sensors unit, for the displaying, controlling and transmit the data to the control center. In one level, the microcontroller can handle all the processing source. In later the circuit will be more complex, more sensors units and control input. Thus a proper circuit and design required to satisfied monitoring system and to accomplished the cost effective of PV solar power generation.

The circuit will work into a module for example the sensors circuit, it will be as a single module before connecting to the microcontroller. Here the data will be process to become calibrated value and display to the user. The system will have wireless interconnectivity support, transmit the data wirelessly to the control system.

The technology here is called as Zigbee wireless technology. It can support to multipoint of data transmission from point A to point B. The PC will receives real-time data from the microcontroller and downloads all the collected data to the database. The software will not just stores data but also records in a local database and generates daily reports that can be into numerous document compatible format.

By addition of cloud computing technology, the data is available on the internet. It allows any devices that connect via the internet to view real-time data and it is happen on user demand. Handheld devices for example like the Smartphone, tablets can accesses this features, to view the data where ever there is connecting to the internet. This hoping to improve the way of retrieving and analyze data.

2.10 PRODUCT COMPARISONS

Below is a table for comparison product studies to commercial product on market. From the comparison, there is disadvantages, for one important criteria whereas no fully cover on the environment tracking system and simulation. And the system is consider huge and very expensive.

Table 9: Product Comparison for PV Monitor System

PV Monitor System	Advantages	Disadvantages
SCADA Tools + PV	Suitable for controlling the power plant. Consist of standard power measurement tools	Is complicated to configure Not dedicated for PV monitoring Not user friendly PLC Module system is huge
TNB AC Power and Energy Meters for PV	Suitable for effective PV output power generation. Include energy management, equipment performance monitoring, and diagnostics.	Not focus on environment monitoring. Need to purchase multiple product, inverter, converter and software differently
Other PV System with software: Example: Firstsolar PV system, SMA analysis, PVSpot, Sunways Monitor, TRNSYS, Insel, Homer Energy, SAM PV system	The monitoring PV system consist with wide details analysis for performance and simulation	Not consist on controlling the PV angle Expensive, less detection, less environment parameters, less analysis, less control function

CHAPTER 3

METHODOLOGY

3.1 RESEARCH METHODOLOGY

The objective of the PEMANTAU is to developed a real time monitoring system that includes the analysis software where it functioning as a guidelines that demonstrate the electric energy production and simulate in technical view and also provide the analysis tools for the maintenance and enhancement to the system.

The PEMANTAU also allows the user to see, track and analyze the solar output production in real time on the internet via a graphics-rich public online dashboard, to monitor the energy generation, load demand, irradiance, and performance data down to the user in real time.

The PEMANTAU will provide a remote monitoring solution that allows the user to manage and view the PV solar power generation system. The data will be store, can be retrieve, viewed anytime and anywhere using a web browser or any internet-connected device.

The concept of the PEMANTAU is to provide the renewable energy system support on monitor the condition of renewable energy with including features like visualization tools for full capability to understand the operating characteristic. This solution allows to improve metering results output. With the PEMANTAU, the user can have an accurate and real-time view of the PV panels output and production. It allows users to view whole PV panel in graphical view and can monitored more than a PV panel on systems with just one single view.

3.2 PROJECT ACTIVITIES

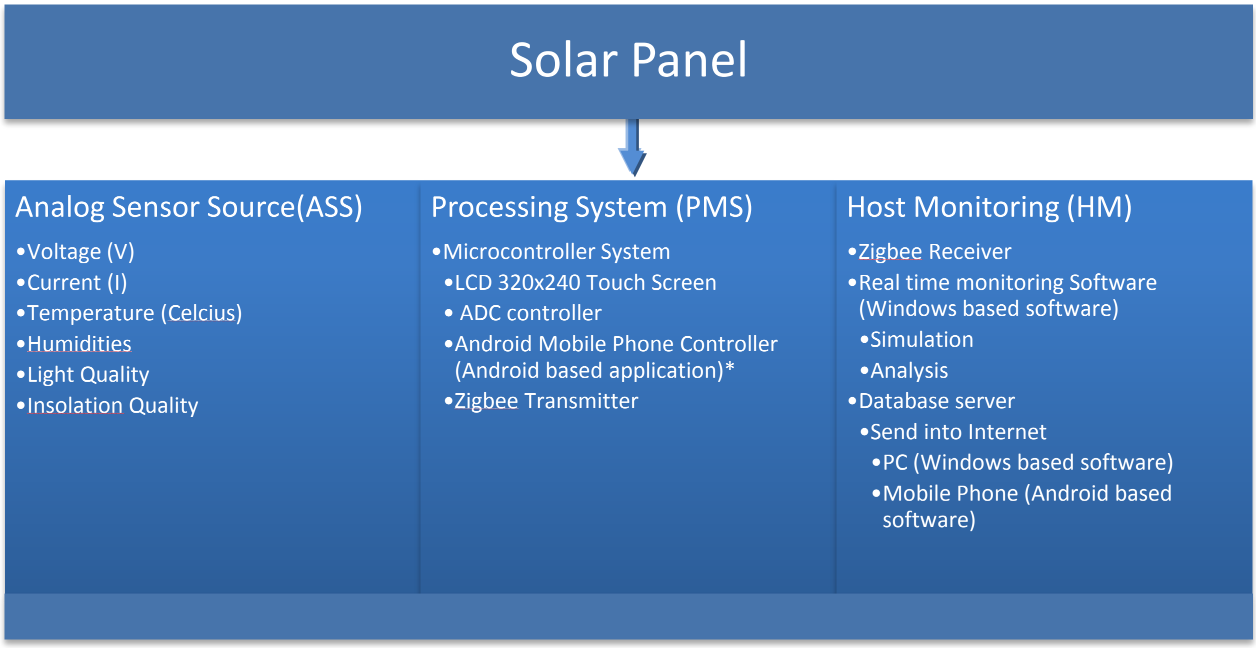


Figure 12: Task module block diagram

The project is divided into 3 states, namely Analog Sensor Source (ASS), Processing System (PS) and Host Monitoring (HM). The project activity is separated into two, the hardware and the software. On the hardware part, it will cover on the sensory units comprising by multiple sensors acting as analog inputs for the microcontroller to read them and use them for processing the information.

The microcontroller will interact with the LCD, used for display purposes; the LCD acts also as interactive device utilizing Touch screen features for user input, enabling user to control the microcontroller and perform manual configurations. Another feature for this system is the capability to interact between Android phone; the interface module will be connected to the microcontroller digital input/output.

Key features of the system:

- Wide range of parameters, can sense by multiple sensors (Voltage, Current, Temperature, Light, Humidity, Insolation, and Angle of panel tilt).
- Remote integration, the capture system that is processing module (ASS and PS) can be far away from data center HM, the target will be around 1.6 km.
- Multifunction graphical LCD display on the processing module (PS) providing details about the current output, energy yields, operating parameters, and date/time.
- Touch Screen features for navigating the configuration of the sensor parameters.
- Data logging in GUI, in order to provide easy overview of the system, all the parameter will be measured and rapidly changed in a real-time.
- Simulation of I-V content, acquisition graph that changes in the real-time.
- Remote monitoring anywhere, data is accessible via Internet connection anywhere, either using a computer or mobile phone.
- Providing analysis tools as solution, verifying the MPP tracking efficiency, performance, and conversion for the PV panel.
- Performing snapshot analysis of identified critical times, history logs.

3.3 MILESTONES

FINAL YEAR PROJECT 1 - DURATION: 28 May - 20 August 2012

Table 10: FYP1 project gantt chart

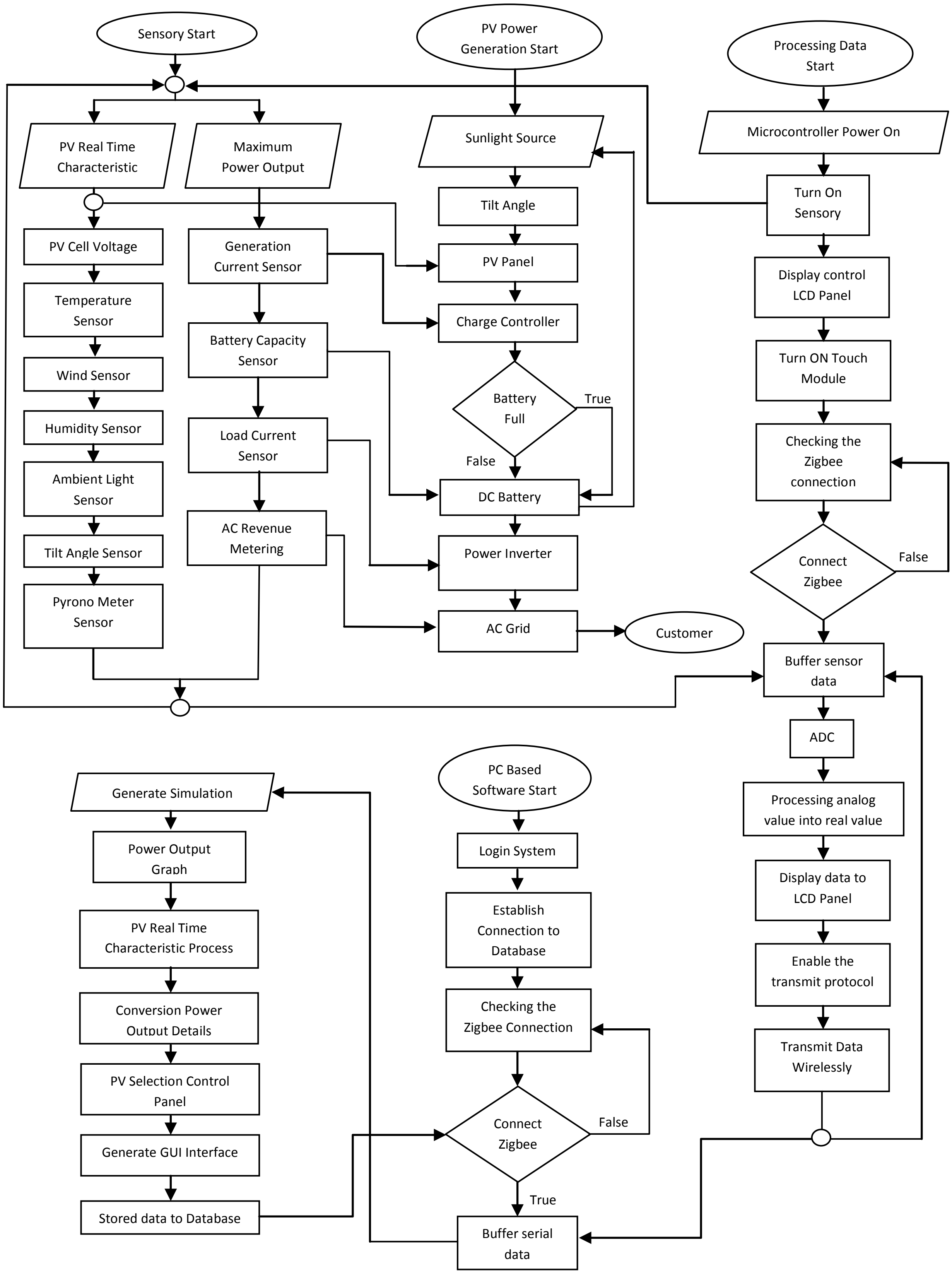
	Phase 1					Phase 2					Phase 3				
Task	**W1**	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**	**W9**	**W10**	**W11**	**W12**	**W13**	**W14**	**W15**
Project Confirmation and Clarification from SV															
Finalize Concept Design and Literature Review															
Circuit Design, Item Received and Circuit Installation															
Input System and Output System from Touch Screen LCD microcontroller script															
Extended Proposal Writing															
Real Time Analog Data (voltage, current and etc) retrieve microcontroller script															
Transmitting and Receiving using Zigbee module to target PC microcontroller script and VB script															
Developing clean interface for PC based using VB2010															
Viva: Proposal defense and Progress Evaluation															
Upgraded Analysis module for PC based															
Draft Report															
Final Report															

FINAL YEAR PROJECT II - DURATION: 18 September - 27 December 2012

Table 11: FYP2 project gantt chart

	Phase 1					Phase 2					Phase 3				
	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	W15
New GUI Integration and Touch Interaction - Touch Screen Module															
Analog Circuit work, Transmit Data, and Processing Programming - Touch Screen Module															
Zigbee Network Multiple Data and Asynchronous Way Development - Touch Screen Module															
Retrieve Data, Application Structure Programming and GUI - VB Real Time Module															
Solid Interface for Interactive Visualization - Android Module															
Human Interaction Programming - Android Module															
Progress Report															
Server, Internet and Database Programming - VB Real Time Module															
Analysis Application for Insolation, Tilt Angle and Sizing Programming - VB Real Time Module															
Historical System - VB Real Time Module															
Data Retrieve from Internet Programming - Android Module															
Setup Processing Module into Prototype Box															
Data Embedded Storage Programming - Touch Screen Module															
Validate the Real Analog Data - Touch Module Module															
Validation Development - Touch Module Module															
Validation Development - VB Real Time Module															
Validation Development - Android Module															
Final Product Install to Prototype Box															
Presentation Slide Preparation															
VIVA															
Draft Report															
Final Report															

3.4 FLOW CHART



3.5 CIRCUIT DIAGRAM

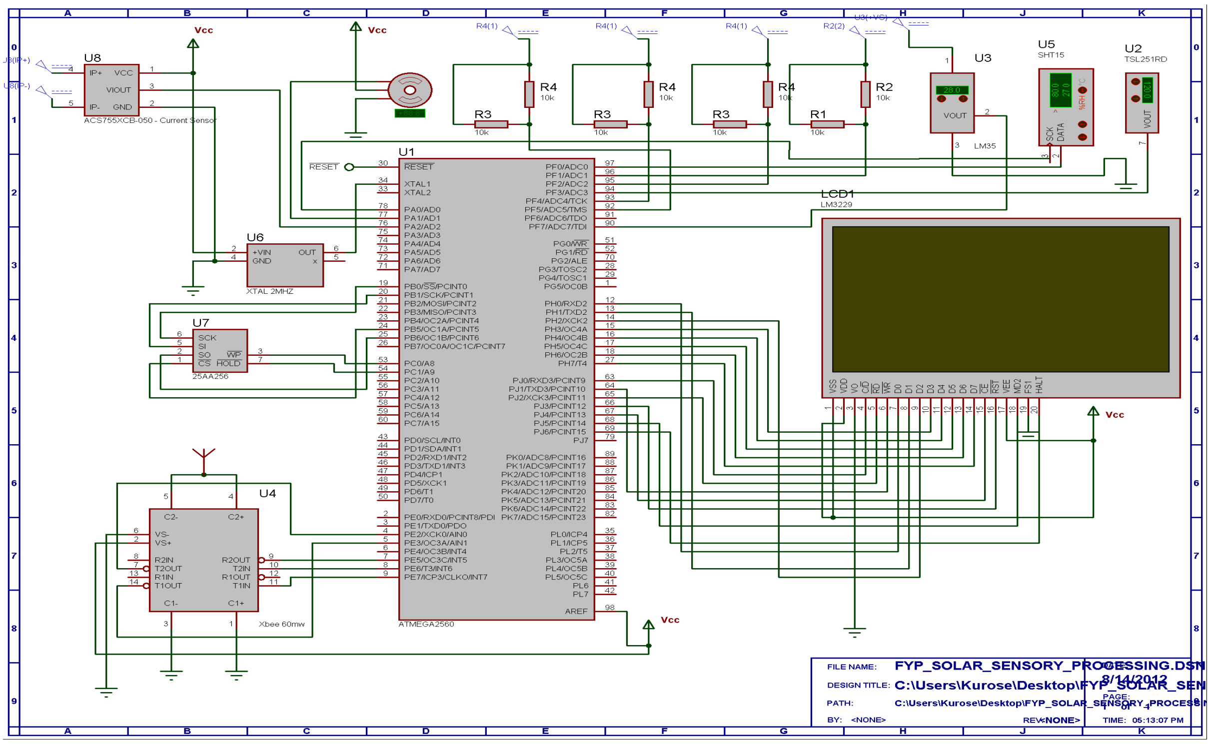


Figure 13: Processing Module Hardware Circuit Schematic Diagram

3.6 PROJECT BLOCK DIAGRAM

There are 5 modules on the hardware topology. The Power Module, it is consist of the standard power generation for solar electricity system that is the battery, the charge controller and the inverter.

The Sensor Module is divided into 2 categories, the PV characteristic and the power output. The sensors unit of PV characteristic will cover on the environment tracking that range from weather, temperature and quality of light. Together all will be in single analog sensor module, the environment sensor and the electricity sensor. The electricity sensor unit consist of power output will have couple of measurement the conversion voltage output, the usage of current, the battery capacity and revenue metering.

The Processing Module will continuously capture all the data and process it became recognized value, from here the data will be use to display on the LCD part and transmit wirelessly by using Zigbee protocol. The insertion of the touch screen module is for enabling user to have control access to the sensory parameter and some adjustment to the system, this reduces switching and wire issues.

The real time software is running on the Host Monitoring Module, here another Zigbee protocol is connected, the successful connection will bring the synchronization to the Processing Module. The data is fetch serially and this will be used by different method from Simulation, Analysis, Control and Data Logger. The end of the module is the Cloud Module. This is enhanced system that made for wide data availability that can accessed anywhere by internet connectivity. The Host Monitoring Module will basically upload the data into the internet database, the specialize application for browser or smart phone application is made to fetch the data from the database and then provide end user real time monitoring system.

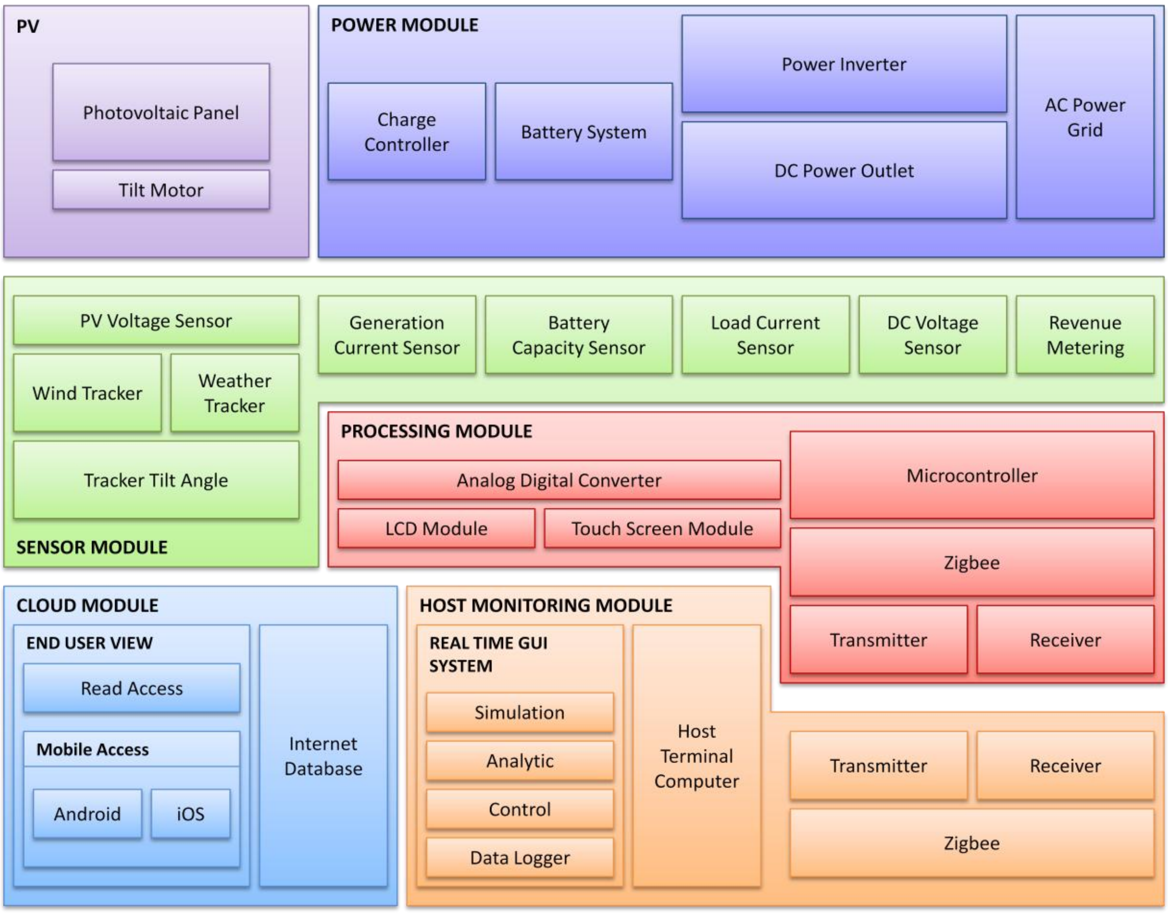


Figure 14: PEMANTAU Module Block Diagram

3.7 HARDWARE

3.7.1 List Tools

Table 12: Tools for PEMANTAU

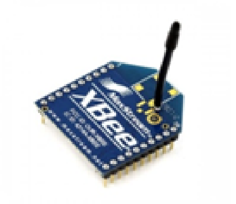
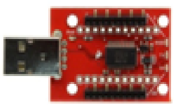
TOOLS	DESCRIPTION
 **Figure 15: Solar Photovoltaic (PV) Panel**	**Solar Photovoltaic (PV) Panels** - designed to operate independently of the electric utility grid, and are generally designed and sized to supply certain DC.
 **Figure 16: Atmega 2560 Microcontroller + Arduino Board**	**Atmega2560 Microcontroller + Arduino Board** - used for the main unit of the processing power, controlling the input and output. It is powerful enough to bring all these things together in a single chip.
 **Figure 17: 2.4 GHz Zigbee**	**2.4 GHz Zigbee** - allows a very dependable and simple communication between microcontrollers, computers, systems.
 **Figure18: Zigbee Explorer Dongle**	**Zigbee Explorer Dongle** - to connect the Zigbee module to USB.

 **Figure 19: TFT LCD Screen Module**	**TFT LCD Screen Module -** LCD interface with the Touch, SD card and Flash design.
 **Figure 20: Temperature and Humidity Sensor**	**Temperature and Humidity Sensor -** features a calibrated digital signal output with the temperature and humidity
 **Figure 21: Hall Effect Based Current Sensor**	**Hall Effect Based Current Sensor -** The sensor gives accurate current measurement for both AC and DC signals.
 **Figure 22: Ambient Light Sensor**	**Ambient Light Sensor -** sensor that changes the voltage value from the incoming light.
 **Figure 23: Android Smart Phone**	**Android Smart Phone** - for software control and monitoring purposes on the next development of the project.

 **Figure 24: Dedicated Database Internet Server**	**Dedicated Database Internet Server**- Database is accessible via Internet connection and available to the end user.
 **Figure 25: Host computer PC/Laptop**	**Host computer PC/Laptop -** As platform for real-time simulation software.

3.7.2 Hardware Interconnection

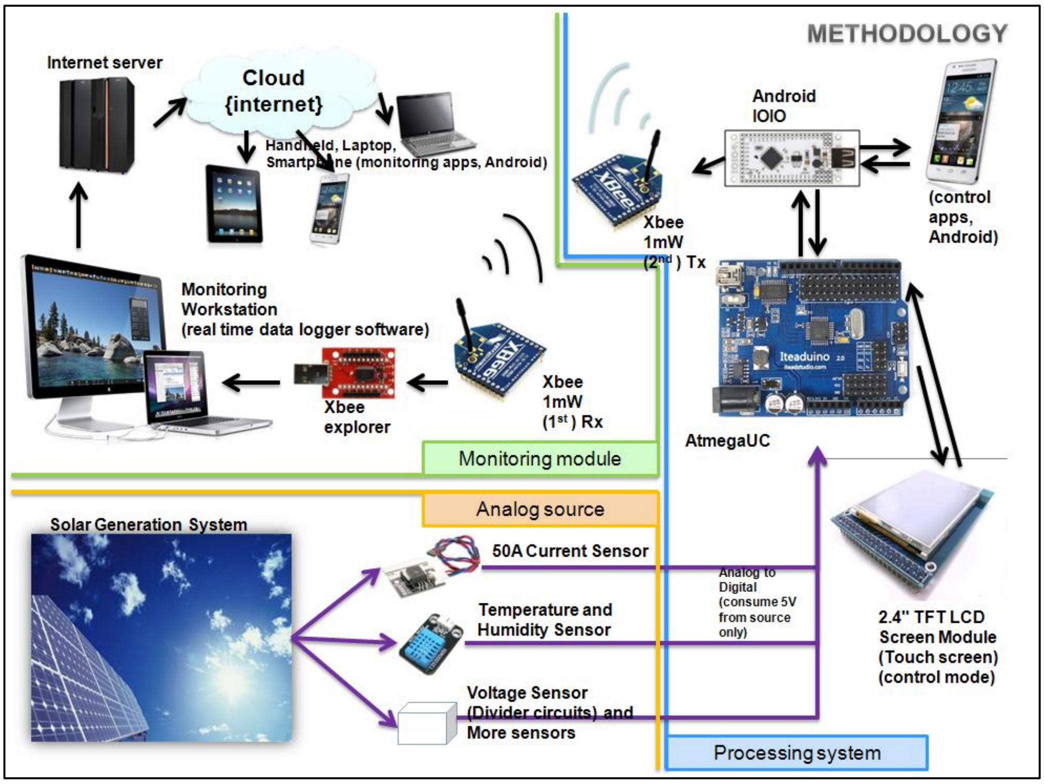


Figure 26: Basic connectivity for the hardware based by module

3.7.3 Project Concept

Figure 27: Software Interface for Monitoring System.

3.8 PROJECT DEVELOPMENT

3.8.2 ReTOUCH - Touch Screen with Sensor Module System (Processing Module)

ReTOUCH - A Touch Screen with Sensor Module System is a Processing Module whereby consists of microcontroller that connect all the parameters sensors, a touch screen alongside with LCD and the Zigbee network circuit. It is programmed to processed the data and then shows the outputs by displaying the the graphical user interface (GUI). There are 4 types of features for the users to select and view the contents, PV Output, Graph, Ports and Setup.

On the PV Output panel it will display the real time voltage (V) and current (I) that get readings from the PV panel. Data will continuous changes as the microcontroller will kept capturing and converting the values into the V and I value. The Graph panel is to presents I-V curve that characterizes the PV outputs.

User can also updates the system on Setup. Setup option here is to allow users to customize parameters such as adding or editing the parameters. It enable the user to have ehance or improve the system. The Port option here is for the user to set the Zigbee network, for an example the user can select whether to is connect or not for transmitting the data, this features will useful during maintenances whereas the system need to be stop monitoring for configuration purpose.

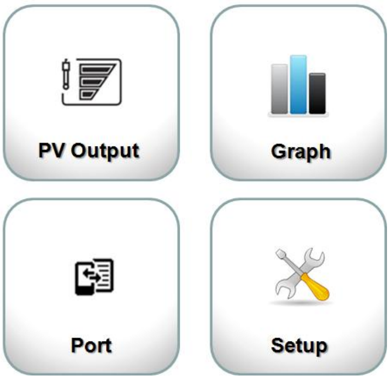


Figure 28: Processing Module System - Touch Screen with Sensor

3.8.3 Embedded Real Time Operating System

There is lots of function on the ReTOUCH System and it which requires more execution activity at the same time, this from fetching the data from analog input, goes to processing the raw data to become calibrated data, displaying the data to the LCD and at same time it is required to listen to the touch input from user. And at the same time, ReTOUCH System is transmitting the data wirelessly by Zigbee network circuit. Due to lots of functions running and try to be in sequence, there will delays on every task on the execution. This seems not efficient way to make it as real time application for the project.

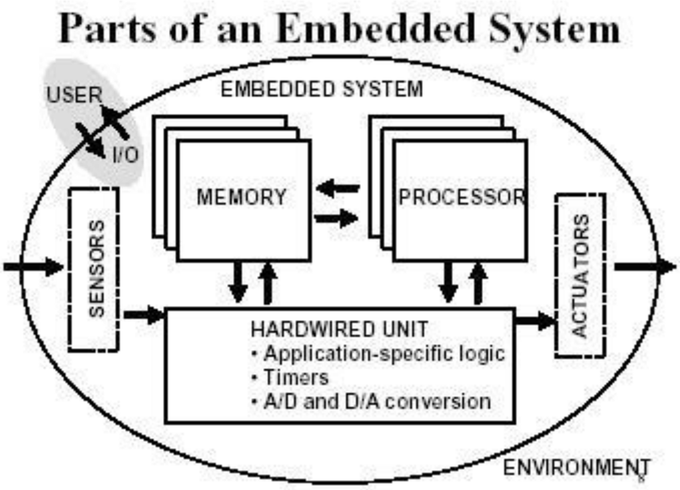


Figure 29: Parts of an Embedded System

The best implementation for the PEMANTAU is by implementing a embedded Real Time Operating System(RTOS). RTOS is a program that manages the memory, speed and timing that count on a specific "lag time"; the time between the request for action and the noticeable execution of the user request. It also to achieve time reliability, real-time programs and prioritize deadline actualization before anything else. The implementation of RTOS can make the system to run faster and seems to look like a real time device.

3.8.4 PEMANTAU Admin System - PC Based Software (Host Monitoring Module)

Figure 30: Host Monitoring Module - PC Software

The PEMANTAU Admin System is a desktop software installed in a host computer. The software is wrote in Visual Basic.Net; a framework for windows based application. The software is run together with the ReTOUCH System; where it need to connects together with the single Zigbee module to retrieve the from the ReTOUCH System. The Admin System consist of high end tools for brief more data and present in more details with more tools that can be fully utilized for renewable energy system. There is 4 type of tools, Simulation, Analysis, Data Logger and Database.

Simulation - This is to track the energy production in real time via a simulation graph. Here it will bring detailed explanations about the output characteristic. It also present the environmental conditions, energy generations, load demands, sun irradiance.

Analysis - It is to promote user to find out the efficiency for the PV panels, generation conversion system, and tilt angle configuration. It will provide tools for trouble shooting and suggest the best setting to generate the best outcome for electricity productivity.

Data Logger - Is a reposition list of historical data about energy generation, energy usage, and environmental information show daily, weekly, monthly and yearly comparisons. The logger will store the data into another list that is the Database, stored in server on internet.

3.9 SOFTWARE DEVELOPMENT

3.9.1 Arduino Mega 2560 Microcontroller

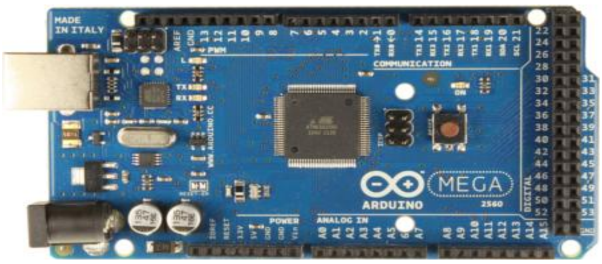


Figure 31: Arduino Mega 2560 board

The platform for the microcontroller for the ReTOUCH System, can be identified as the Arduino Mega 2560 that is based on the ATmega2560 microcontroller chip by Atmel[22].The Arduino Mega 2560 board can support up to 54 digital input/output pins (of which 14 pins can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

The programming languages used in this microcontroller is the modified C languages for the Arduino compiler. The editor for writing programming code is using the open-source interface provided by Arduino which is free to download. Shown here in the Figure 32 below is a sample of the sensor module source code to display the PV voltage and current output.

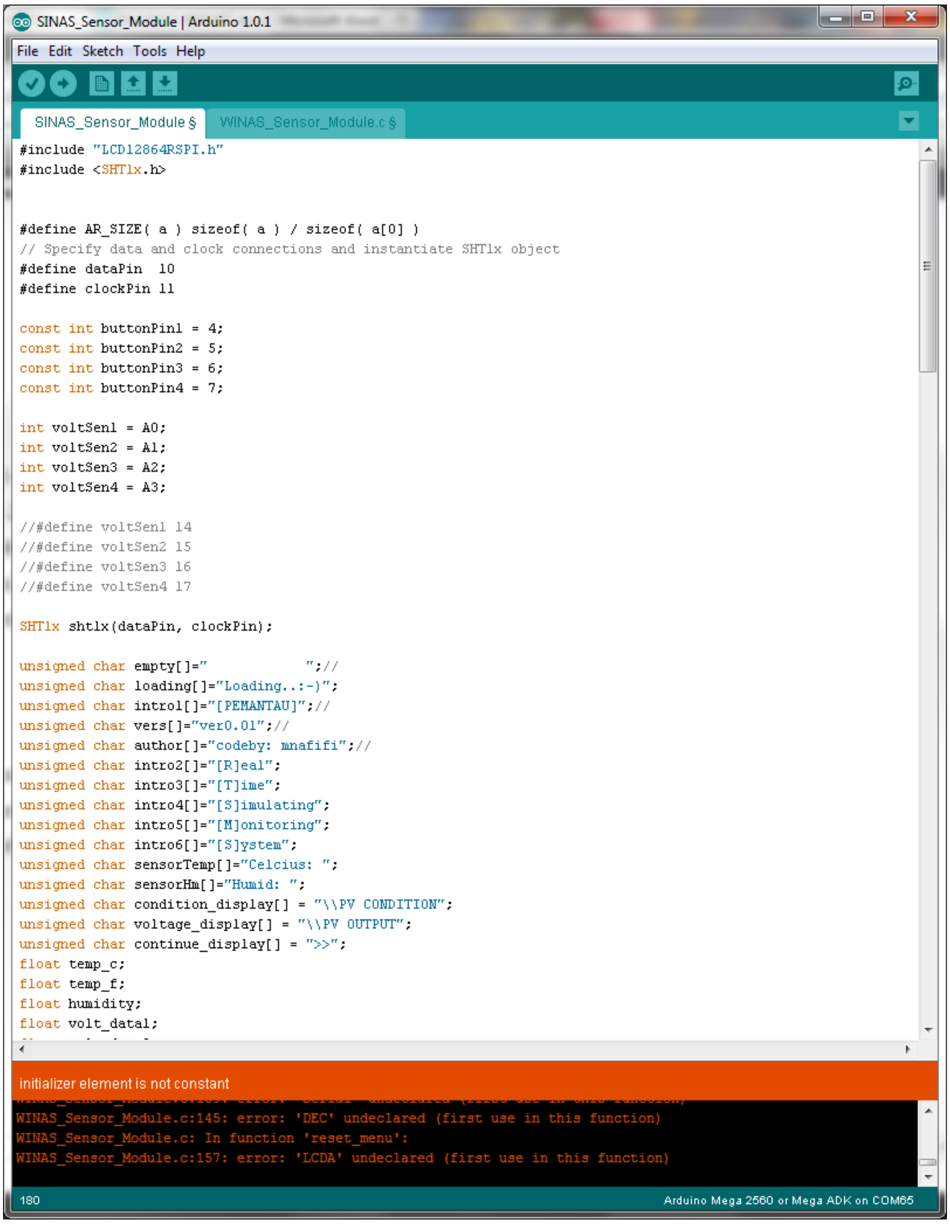


Figure 32: The Arduino IDE with sensor module source code

3.9.2 ReTouch System

The output for displaying processed data is the Touch LCD unit (3.2" TFT LCD Screen Module: ITDB02-3.2) that connected together with the Processing Module. The basic functionality is to display graphical output that is designed to use 16-bit mode. It is designed with a touch controller in it, Figure 33 shown the pin-out for ITDB02. The touch IC is ADS7843 and the touch interface is included in the 40 pins breakout.

The IDTB02 pin output of 40 pins interface that require to connect with the microcontroller, the interface include LCD bus, SD card bus, Touch screen bus and the Flash bus.

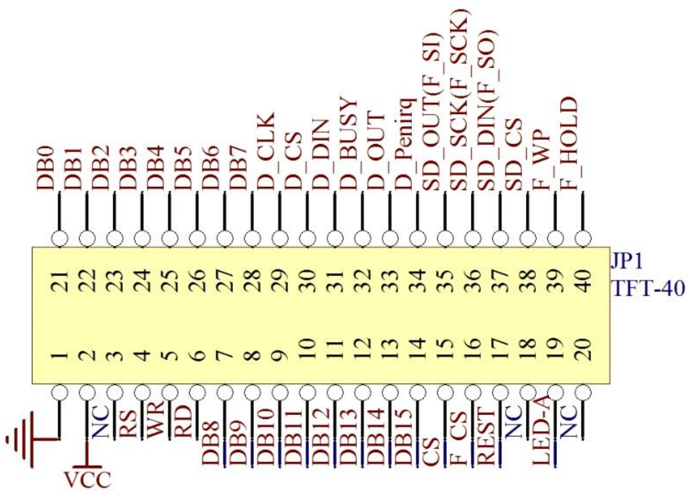
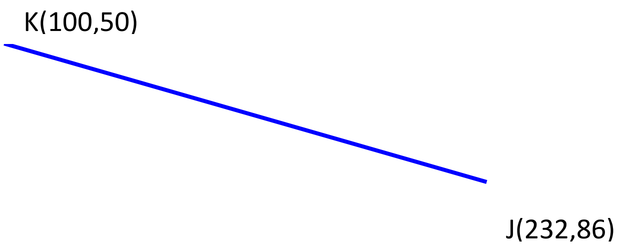


Figure 33: ITDB02-3.2S pin-out

To display data to the 16-bit LCD, it requires a specific step of function(source code) that have provided by the manufacturer datasheet. The flexibility of displaying the data depends on sequences of codes. Thus it is important to manage the code like time scheduling, variable recycling and memory clearing. This is because it can consume to much on the microcontroller memory whereas LCD graphic require lines of function and more processing power.

The LCD consist of pixels of colors that support up to 16-bit. It use RGB as the colors library. To draw a line for one need program it from one location of X position and Y position to another location of X position and Y position.

For example to draw one line from one point 'K' location,(X=100 and Y=50) to next point 'J' location, (X=232 and Y=86). We color the line into blue color.



the RGB Hex value for blue color is 0000FFh or (Red = 0, Green = 0, Blue = 255), so the program code will look like:

```
myGLCD.setColor(0,0,255);
myGLCD.drawLine(100,50,232,86);
```

Basically all the graphical interface that is displayed on graphical LCD will drawn in layers. Below shows the code fragment that used to draw the fonts, line, rectangular, positioning and colors. Others critical codes that are important for providing the user input is the sequences for touch event. The codes user touch input, which is the same case as to get the location (X and Y) - of the touch gesture is being pointed to.

```
x=myTouch.getX();
y=myTouch.getY();
myGLCD.printNumF(x,3, 190, 210);
myGLCD.printNumF(y,3, 190, 220);
if ((y>=13) && (y<=30)){
  if((x>=148) && (x<=200)){
    display_workspace();
    display_computation();
  }
  if((x>=201) && (x<=259)){
    myOS.pauseTask(data_captures);
    myOS.pauseTask(data_processing);
  }

  if((x>=260) && (x<=319)){
    myOS.pauseTask(data_captures);
    myOS.pauseTask(data_processing);
  }
}
```

3.9.3 Sensor Conversion

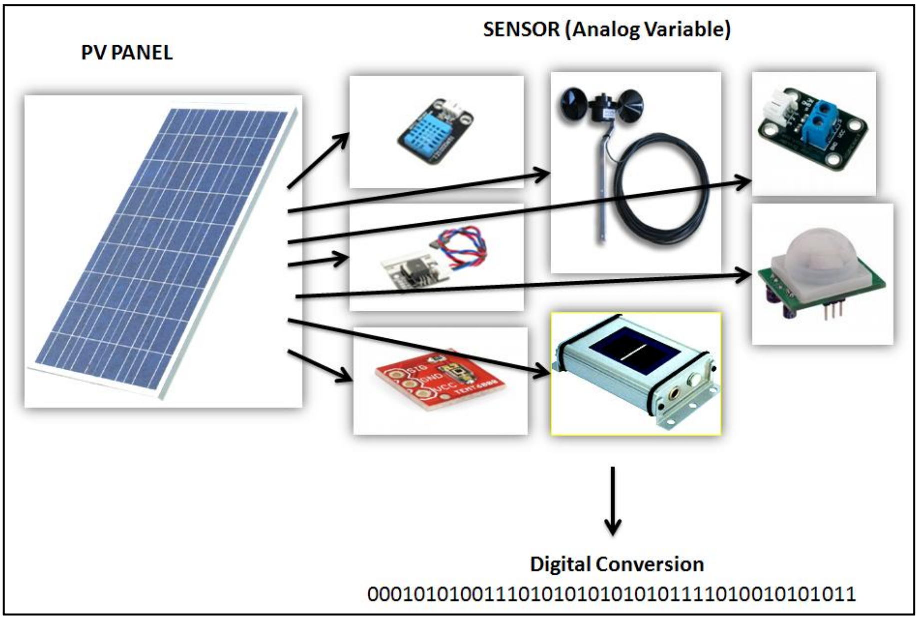


Figure 34: Conversion from analog to digital concept

To fetch the data from the various types of sensor, we can use the analog-to-digital (ADC) conversion circuit to convert the analog voltage values into the digital data. These digital data consist of range of voltage values that determined by the voltage reference. For an example, when the voltage reference is 5V, the range for analog input must be equal or lower than 5V. The 5V is will be converted into a digital value where the bit numbers of the conversion goes from the max voltage reference to bringing the step size of the voltage; so in this case it is 1023. Then this range of data needs to be convert again to make it accurately calibrated data.

For the ReTOUCH system, it has been tested by the ideal multi-meter. By default the user can simply do the adjustment, by selecting the Setup on the menu. This features enable the user to change the calibration parameters. Show below is the code fragment on the conversion analog sensor data to digital data.

```
float sensor_current_PV = 0;
float sensor_temperature_PV = 0;
float sensor_humidities_PV = 0;
float sensor_wind_PV = 0;
float sensor_light_PV = 0;
float sensor_voltage_INV = 0;
float sensor_voltage_BAT = 0;
float sensor_voltage1_PV = 40;
float sensor_voltage2_PV = 0;
float sensor_voltage3_PV = 0;
float sensor_voltage4_PV = 0;
float sensor_voltage5_PV = 0;
float sensor_voltage6_PV = 0;
float sensor_voltage7_PV = 0;
float sensor_voltage8_PV = 0;
float temp_c;
float temp_f;
float humidity;

data_light_PV = sensor_light_PV;
data_temperature_PV = temp_c;
data_humidities_PV = humidity;
data_wind_PV = random(0,100);

//PV voltage module processing data
data_voltage1_PV = ((sensor_voltage1_PV)*5)/1023;
data_voltage2_PV = ((sensor_voltage2_PV)*5)/1023;
data_voltage3_PV = ((sensor_voltage3_PV)*5)/1023;
data_voltage4_PV = ((sensor_voltage4_PV)*5)/1023;
data_voltage5_PV = ((sensor_voltage5_PV)*5)/1023;
data_voltage6_PV = ((sensor_voltage6_PV)*5)/1023;
data_voltage7_PV = ((sensor_voltage7_PV)*5)/1023;
data_voltage8_PV = ((sensor_voltage8_PV)*5)/1023;

//Conversion module processing data
data_voltage_INV = ((sensor_voltage_INV)*5)/1023;
data_voltage_BAT = ((sensor_voltage_BAT)*5)/1023;
data_percent_BAT = ((data_voltage_BAT)*100)/5;


data_max_voltage = (data_voltage1_PV + data_voltage2_PV +
data_voltage3_PV + data_voltage4_PV + data_voltage5_PV +
data_voltage6_PV + data_voltage7_PV + data_voltage8_PV);
data_max_power = data_max_voltage * data_max_current;
```

3.9.4 Zigbee Setup and Programming

To transfer data from one location to another location, say from a distance maybe around 500 meters, it may require long wiring for just to sending the data. This is not economical and the cost for wiring is expensive. In today technology, the wireless transmission are made easier. There is quite a number of wireless technology such as Bluetooth, Wifi, HIPERLAN, DASH7. And one of them is the Zigbee that is basely on an IEEE 802 standard for personal area networks protocol.

The Zigbee networks can be formed in an ad-hoc fashion, with no centralized control or high-power transmitter/receiver is able to reach all of the devices. It supports typical both star and tree networks, and generic mesh networks. Every network must have one coordinator device, tasked with creation, control of its parameters and basic maintenance. Different from other wireless technology, the Zigbee is better in term of low power module, mesh extension and long distance data transmission, and it is also far cheaper.

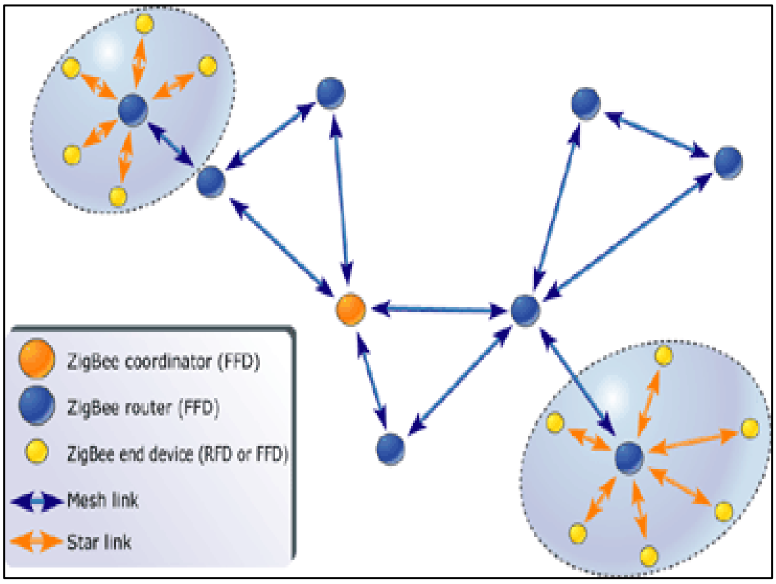


Figure 35: Zigbee network device discovery

The component here to integrate with the standard Zigbee technology is renowned in the communication industry, and is called the Xbee Series Family produced by Digi International.

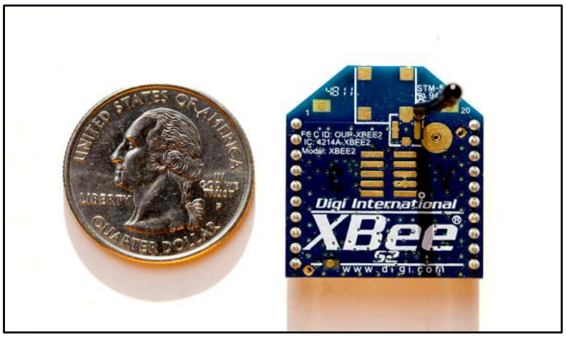


Figure 36: The Xbee module

The PEMANTAU project used Xbee Series 1 60mW which is a point-to-point connection. It can support a range up to 1 mile (1500 meter range). It use same concept of serial data transfer or UART. Thus it can be integrated with any devices that are programmed for serial transmission cycle; this means it can support any microcontroller, and Arduino is no exception.

Table 7: Pin assignment for Xbee

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / **CONFIG**	Input	UART Data In
4	DO8*	Output	Digital Output 8
5	**RESET**	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI	Output	PWM Output 0 / RX Signal Strength Indicator
7	PWM1	Output	PWM Output 1
8	[reserved]	-	Do not connect
9	DTR / SLEEP_RQ / DI8	Input	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	**CTS** / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	ON / **SLEEP**	Output	Module Status Indicator
14	VREF	Input	Voltage Reference for A/D Inputs
15	Associate / AD5 / DIO5	Either	Associated Indicator, Analog Input 5 or Digital I/O 5
16	**RTS** / AD6 / DIO6	Either	Request-to-Send Flow Control, Analog Input 6 or Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

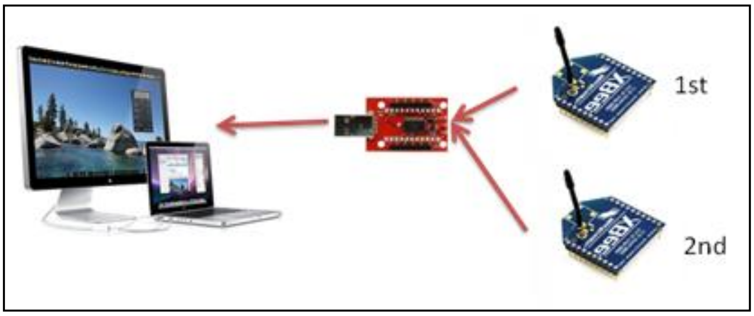


Figure 37: Both must be configure using Xbee Explorer

To program the Xbee module for point-to-point connection, the module parameters must first be setup for transmitter and receiver configuration. This is to make sure that each Xbee module know each other, to avoid interference with other devices, to set the speed of data transfer and also the size of data transfer. To do this, both must be connected with the Xbee explorer then connected using the USB cable to the Host PC. The software for configuration the module is called X-CTU which is provided by the manufacturer (Digi International) itself - this is freely available from the website. The new configuration is normally performed for one time only, and it will retain its configuration until new the parameters are rewritten later, should there is a need for reconfiguration.

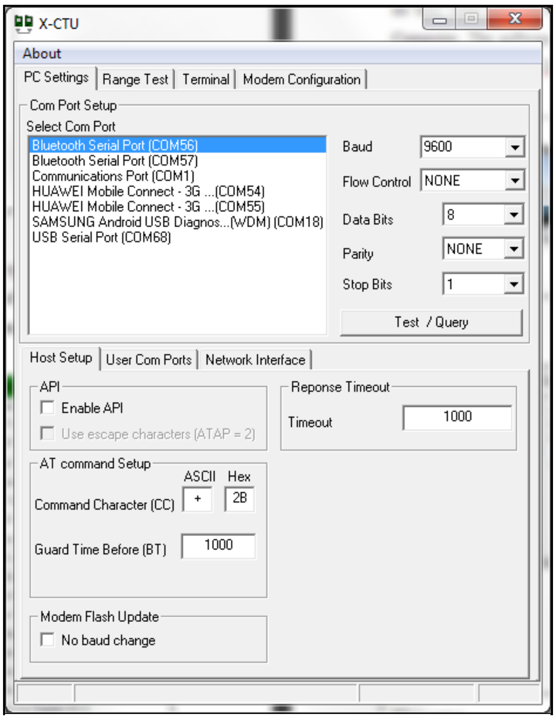


Figure 39: X-CTU PC Settings

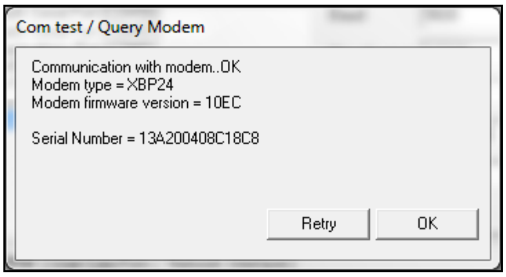


Figure 38: Checking the Xbee is connected to the PC communication port

Both Xbee modules must use the same ID. Only modules with matching IDs can communicate with each other. Unique PAN IDs enable the control of which RF packets are received by a module. Setting the ID parameter to 0xFFFF indicates a global transmission for all PANs. The same goes to ATMY, ATDL and ATDH. Below is the configuration parameter that we use to make both Xbee modules communicate with each other.

Table 8: Xbee parameter configuration

Xbee setup		
	XBEE1	XBEE 2
ATMY (Xbee Name)	0125	0493
ATDL (Who Talks to Who)	0493	0125
ATDH	0	0
ATID (Network)	0987	0987

ATID = Network
ATMY = Xbee Name
ATDH = Destination Address High
ATDL = Who Talks to Who
ATBD = Speed to Talk

The communication between the microcontroller requires same parameters on both side.
The data that have been transmitted by the Touch Module consist of a long list of value ranging from raw sensor data, the processed sensors data and the parameter values. Therefore, the data that have been sent must be separated; otherwise the Host PC will get ab unknown value which is not useful.

Thus, here we designed a new type of data link that consists list of data. The design considers the flexibility for fetching the data at the Host PC side, to let the Host PC know that the new line of data is here and to make it easier for new sensors to be added in the future implementation. The new line of data will start as "NW", the "|" serves to separate a value from other parameters values. "E" is declared as environment data, where "[n]" is going to be an array of variable sensors when it reaches up to the Host PC. The same goes to others; "P" for power, "V" for voltage, "C" for current, "M" for conversion sensor, "XB" is the Xbee setting and "PT" is the useful information provided by microcontroller to Host PC for other setting purposes.

.|NW|E0:|23.34|E1:|434|E2:|44.55|E3:|54.66|E4:|65.33|E5:|23.34|E6:|434|E7:|44.55|E8:|54.66|E9:|99|P0:|434.4|C0:|4.11|C1:|2|C2:|5.2|V0:|10.1|V1:|13.1|V2:|4|V3:|10.2|M0:|20.21|M1:|133.1|M2:|432|XB:|CN|PT:|156636344|

NW: New data link
E[n]: Environment Sensor Data
P[n]: Power Output Data
V[n]: Voltage Output Data
C[n]: Current Output Data
M[n]: Conversion Sensor Data
XB: Xbee Setting
PT: Touch Module Setting

Figure 40: The data link layer for PEMANTAU

Below is the example of code of Xbee function on the PEMANTAU source code:

```
char_array_enviroment[0] = "||E0:" + String(data_light_PV) + "E";
char_array_enviroment[1] = "||E1:" + String(data_temperature_PV) + "E";
char_array_enviroment[2] = "||E2:" + String(data_humidities_PV) + "E";
char_array_enviroment[3] = "||E3:" +
String(floatToString(test,data_wind_PV,3,3,true)) + "E";

  char_array_voltage_PV[0] = "||V0:" +
String(floatToString(test,data_voltage1_PV,3,2,true)) + "V";
  char_array_voltage_PV[1] = "||V1:" +
String(floatToString(test,data_voltage2_PV,3,2,true)) + "V";
  char_array_voltage_PV[2] = "||V2:" +
String(floatToString(test,data_voltage3_PV,3,2,true)) + "V";
  char_array_voltage_PV[3] = "||V3:" +
String(floatToString(test,data_voltage4_PV,3,2,true)) + "V";
  char_array_voltage_PV[4] = "||V4:" +
String(floatToString(test,data_voltage5_PV,3,2,true)) + "V";
  char_array_voltage_PV[5] = "||V5:" +
String(floatToString(test,data_voltage6_PV,3,2,true)) + "V";
  char_array_voltage_PV[6] = "||V6:" +
String(floatToString(test,data_voltage7_PV,3,2,true)) + "V";
  char_array_voltage_PV[7] = "||V7:" +
String(floatToString(test,data_voltage8_PV,3,2,true)) + "V";

  char_array_conversion[0] = "||C0:" +
String(floatToString(test,data_voltage_INV,3,2,true)) + "C";
  char_array_conversion[1] = "||C1:" +
String(floatToString(test,data_voltage_BAT,3,2,true)) + "C";
  char_array_conversion[2] = "||C2:" +
String(floatToString(test,data_percent_BAT,3,2,true)) + "C";

  //char_array_max[0] = "|M0|" +
String(floatToString(test1,data_max_voltage,3,2,true));
  //char_array_max[1] = "|M1|" +
String(floatToString(test1,data_max_power,3,2,true));
```

```
  string_array_enviroment = char_array_enviroment[0] +
char_array_enviroment[1] + char_array_enviroment[2] +
char_array_enviroment[3];
  string_array_PV_voltage = char_array_voltage_PV[0] +
char_array_voltage_PV[1] + char_array_voltage_PV[2];// +
char_array_voltage_PV[3];
  string_array_conversion = char_array_conversion[0] +
char_array_conversion[1] + char_array_conversion[2];

xbee.begin(56700);

xbee.println("|NW" + string_array_enviroment + string_array_conversion +

string_array_PV_voltage);
```

3.9.5 Admin System - PC Based Software

Admin System - PC Based Software is developed using Visual Basic.Net framework. Visual Basic.NET (VB.NET), is a is an object-oriented languages which are from the legacy languages called Visual Basic (VB), but now it the languages emulate the application into a .Net framework.

The VB.Net is one of the best tools to create the graphical user interface (GUI). There are hundreds more languages out there like C++, C#, Java, Python, etc that function like Visual Basic. This is an advantages for the PEMANTAU system, since the GUI can be improved easily without tirelessly working on by each class of object.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 FINAL YEAR PROJECT 1

4.1.1 Prototype Sensor Circuit

Figure 40 below shows the picture taken during the experiment on the prototype sensory unit to the normal LCD screen. The outcome of the prototype is to obtaining on the best calibration that is required for developing the next program.

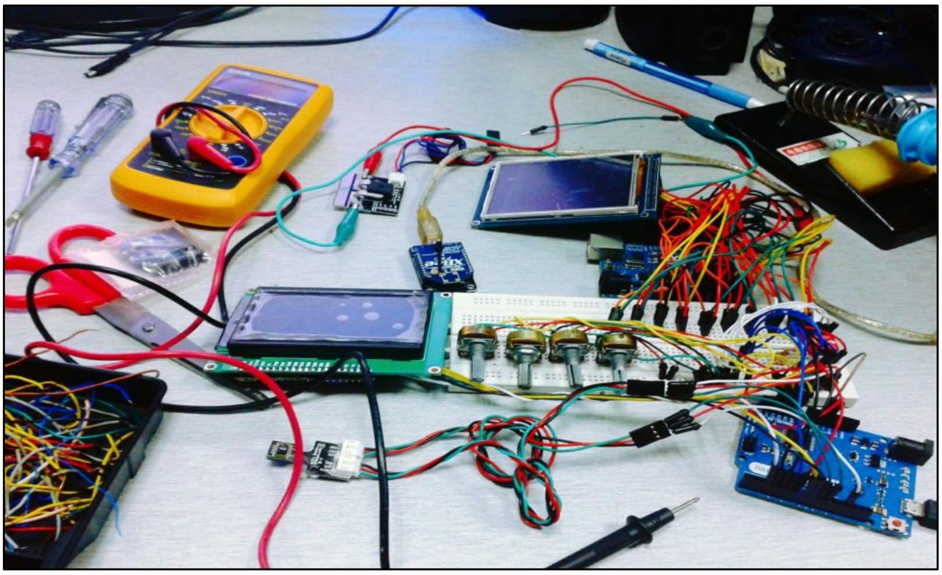


Figure 41: Prototype Sensory Unit

Figure 42: Sensory Unit Calibration Testing on LCD

4.1.2 Touch Screen Module For In-System Control Panel

Figure 42 shows the picture taken during the Processing Module system that has been programmed in real time.

Figure 43: Touch Screen Module for In-System Control Panel (PMS)

4.1.3 Captured Power Output Data Captured

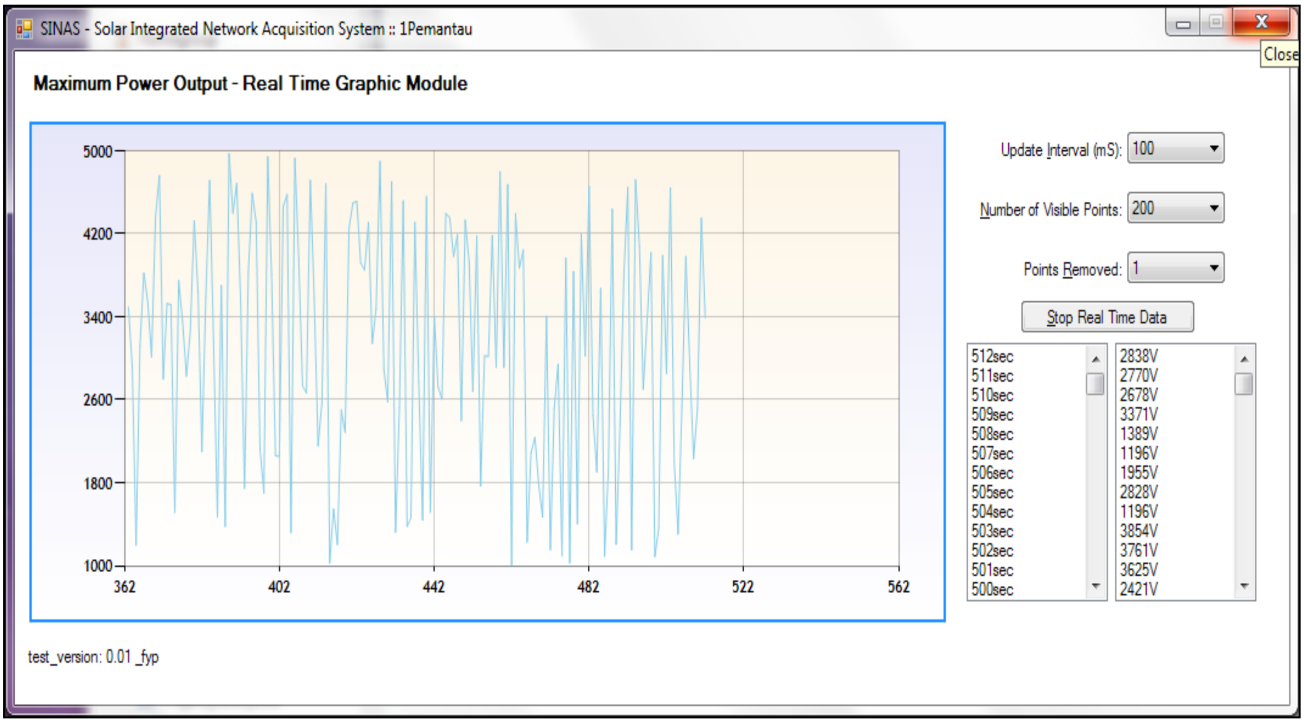


Figure 44: Prototype Power Output - Real Time Graphic Module

Analysis of Data

- The basic idea of this design is to enable the user to understand the power output in real time manner. The user can follow the output to make sure that the output is regulatory compliant.
- A record of every minute is then saved and this can be easily accessed from the system.

4.2 FINAL YEAR PROJECT 2

4.2.1 Final Prototype Sensor Circuit

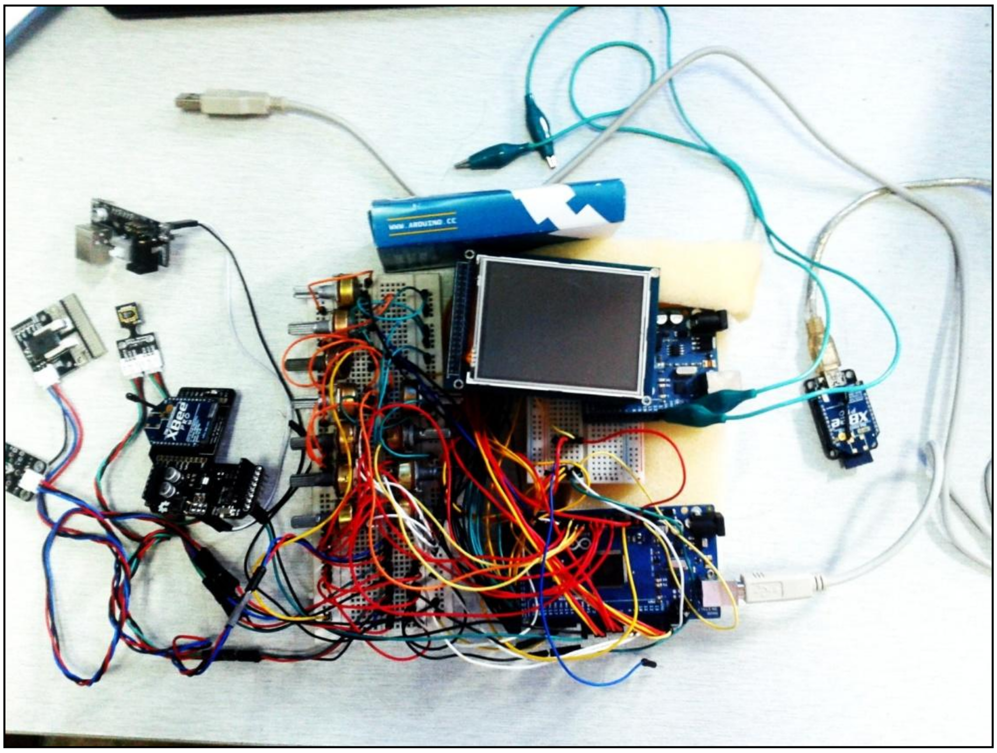


Figure 45: The Touch Module Circuit with Sensor circuit and Zigbee circuit

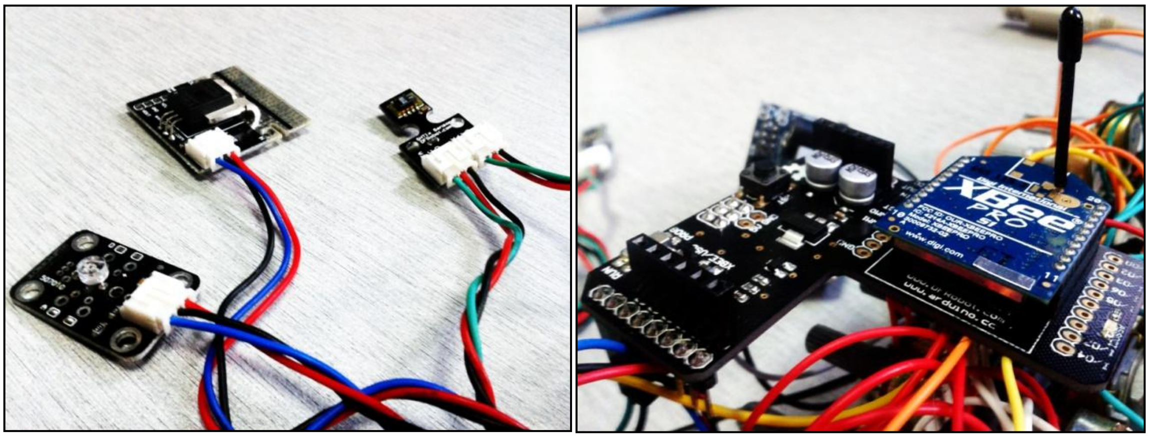


Figure 46: (Left) Light for radiance sensor, 50A current sensor, temperature and humidity sensor. (Right) the Zigbee module connection to the microcontroller (Transmitter)

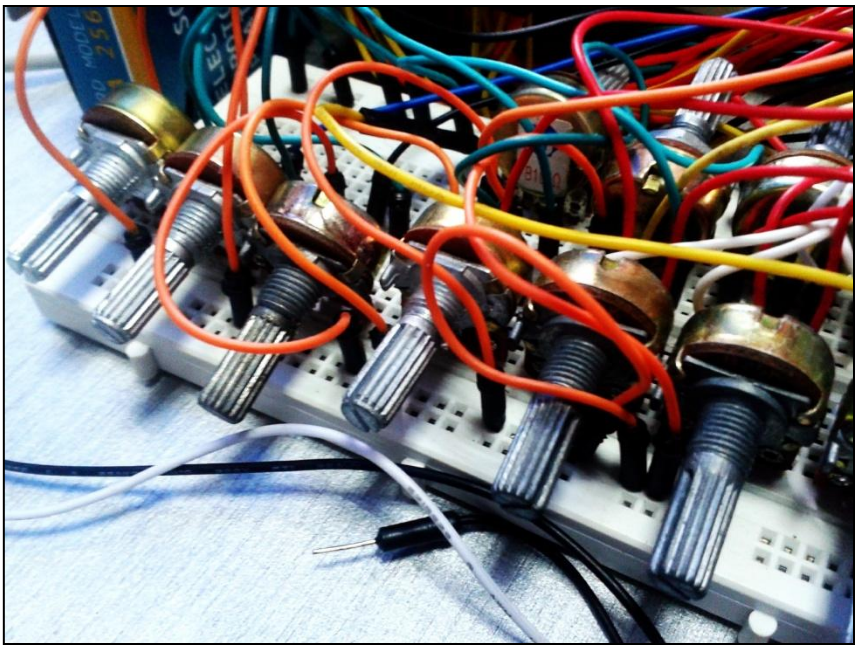


Figure 47: Arrays of potentiometers for PV voltage parameters

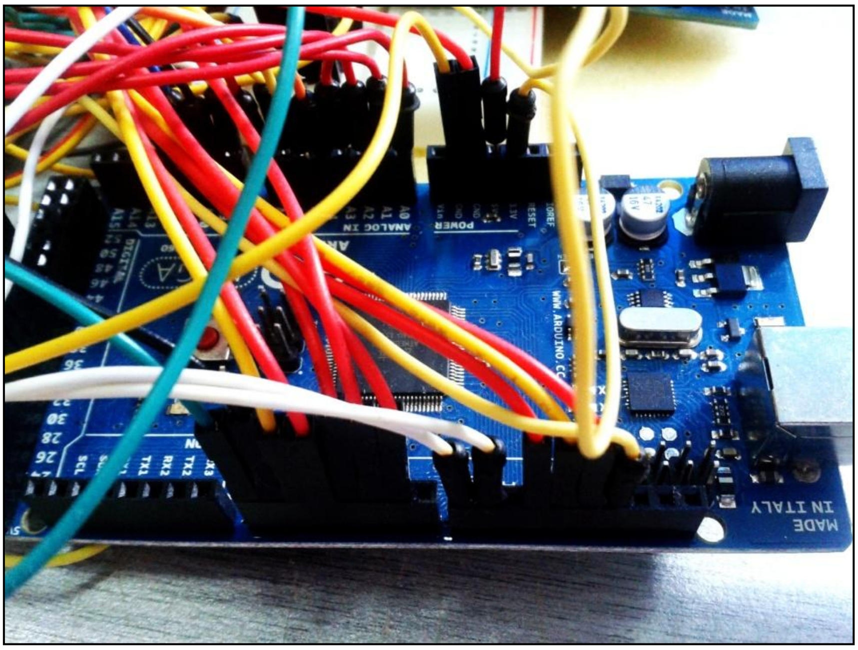


Figure 48: Arduino Mega 2560 Microcontroller

4.2.2 ReTOUCH - Touch Screen Module Output Control Panel

Figure 49: ReTOUCH System shows output for PV characteristic operating values, the battery voltage capacity, inverters AC voltage and the environment sensors comprising irradiance, temperature, humidity and wind.

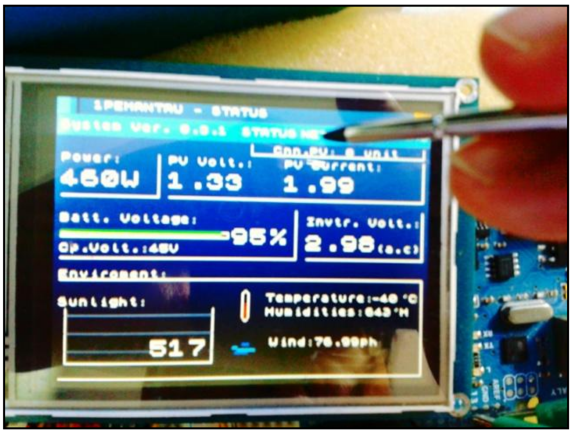


Figure 50: Touch effect on the resistive module

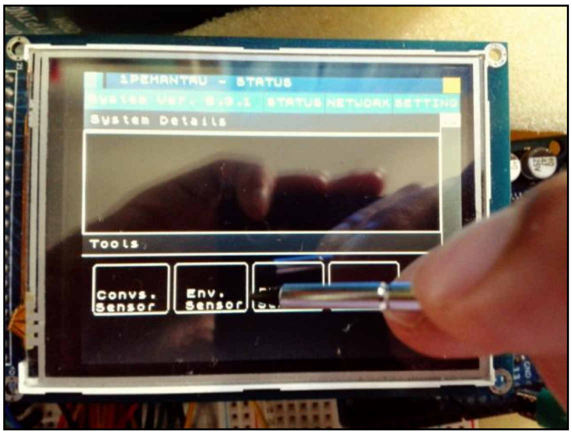


Figure 51: The setting features for user to adjust or adding new parameters that is sensors and system preferences

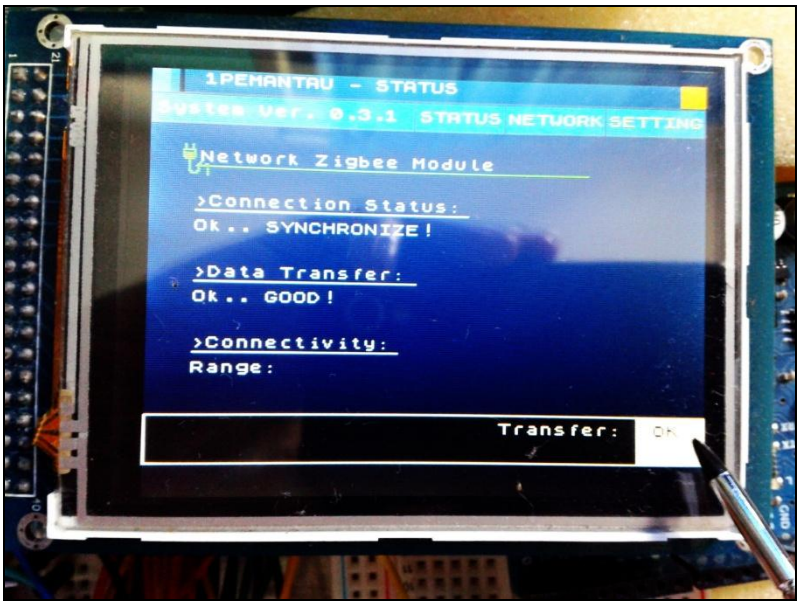


Figure 52: The network features shows here the connectivity for Zigbee module. It enables user to have control on the data transfer to the Host PC

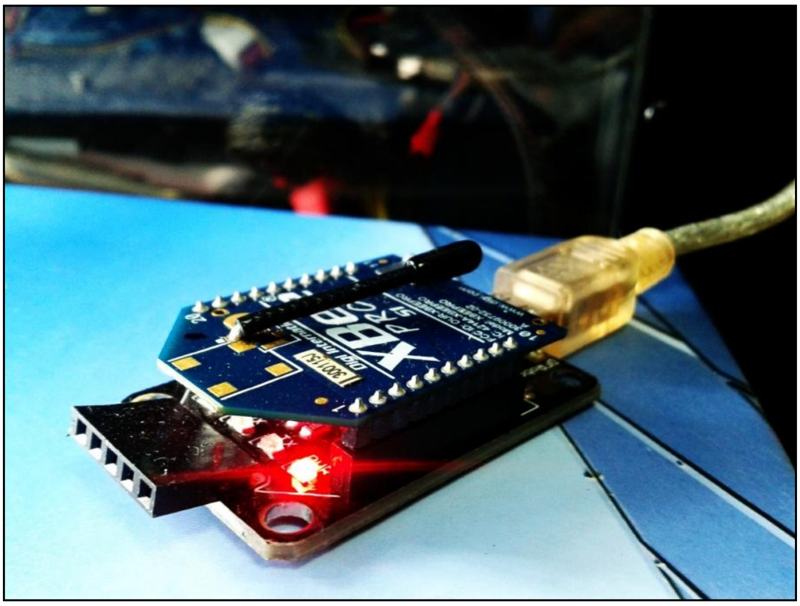


Figure 53: The Zigbee module connected to Host PC using Xbee Explorer and wired by USB cable (Receiver). The Led blinking in Red shows that it is now receiving the data from another Zigbee module (Transmitter)

4.2.3 Admin System - PC Based Software

Figure 54 below show the GUI for the main component of the Admin System PC Based Software. The system can be use as control unit that can monitors the PV operating values, battery consumption, loads and environment condition in real-time.

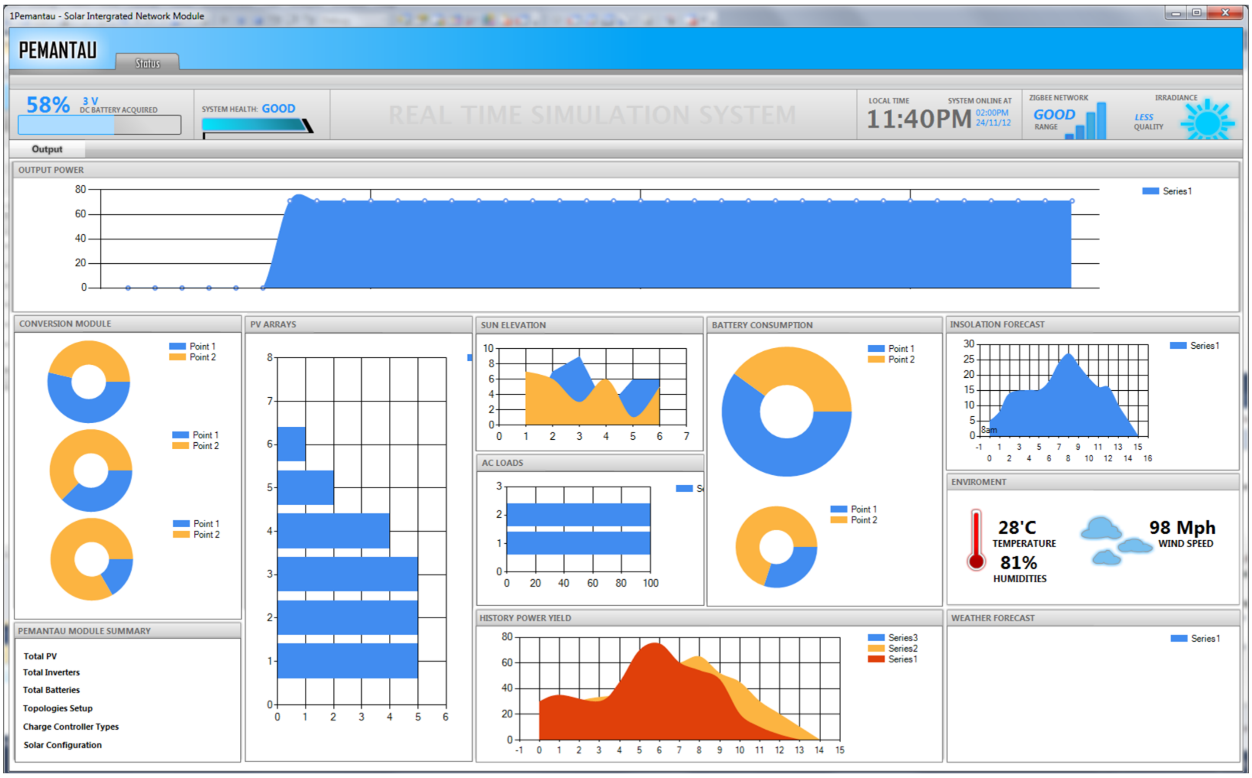


Figure 54: The PC Based system called as PEMANTAU

The tools included with Analysis that provide access on the performance analysis, the PV sizing, array conversion configuration and more, the History tab will function as real-time data logger where the user can view the previous data and then export them to document formats such as Excel or PDF. The Network tab serves as configuration tool for the all the connections that go into the ReTOUCH - Touch Screen Module; the user can adjust the various connections to fit the output requirements. Preference tab allows the administrator to set and modify options like Host PC ports, database connection, user accounts and more.

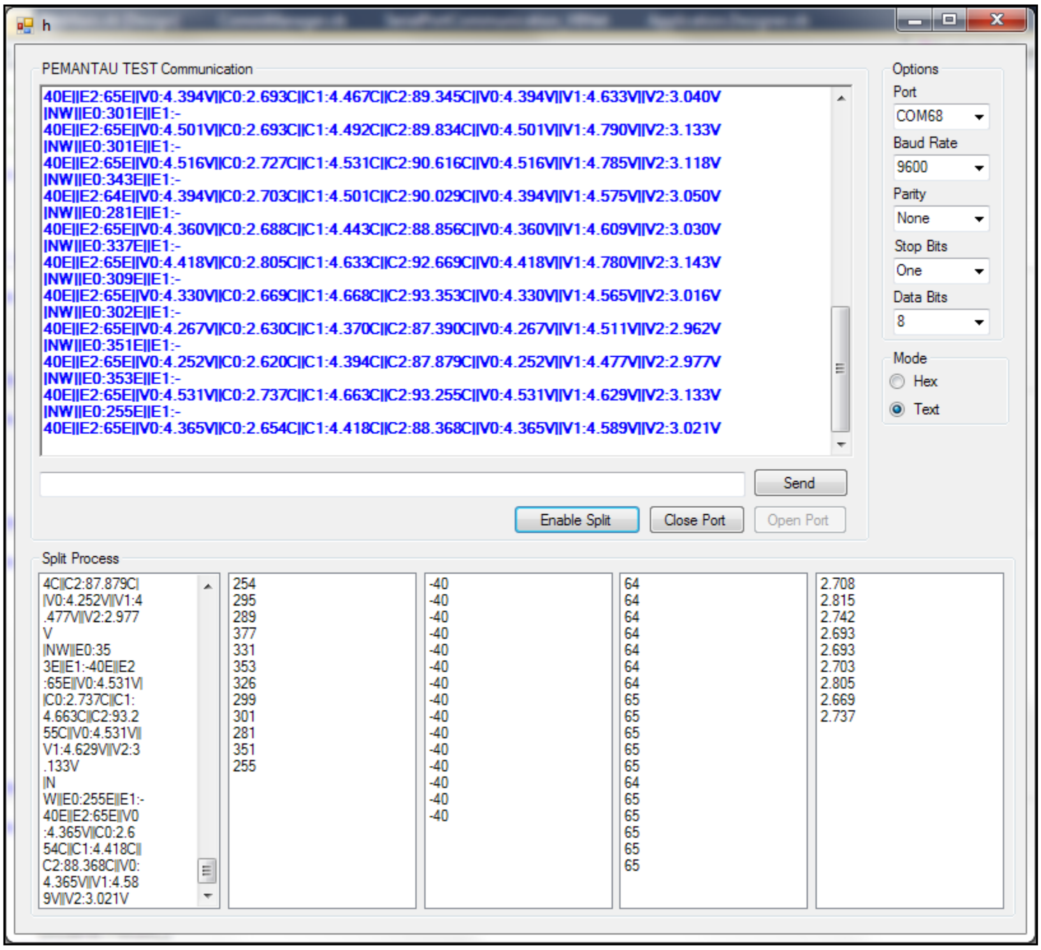


Figure 55: Testing application for fetching the data from the microcontroller transmitting wirelessly using Zigbee connection

4.2.4 PEMANTAU Android Application

Figure 56: The test program for Android application (Testing on Samsung Galaxy Ace emulator using Eclipse Java Android SDK)

4.3 OVERALL ACHIEVEMENTS

The end result of the Final Year Project 1 (FYP1) contributed to the successful completion of development for first prototype of ReTOUCH - Touch Screen Module and now we are able to focus on the improvement of the GUI side. During the project milestones, new features and ideas been added, thanks to the support of project grant giving by the Fundamental Applied sciences. Now we were able to optimize the system to become more solid which is more too configurable and flexibility to the system. On the ReTOUCH, we have also added RTOS features combining the speed execution times and more function for flexibility adding the new sensor without going into programming side. To be note that this was a discovery to wrote a program run that so smoothly. It was a challenge to wrote RTOS for Arduino because there is no information provided showed on the internet, but yet we am able to come up with necessary coding and where no one seems to accomplish, and we am able to shows the result.

During the development of the Final Year Project 2 (FYP2) for the last 14 weeks, we are able to show a new design of the ReTOUCH, to completing the RTOS tweaking, to make it more flexible in terms memory consumption and to the implementation of Zigbee transmission, and to completed necessary codes that can to transmit huge some of data from different of sensor values and parameters to the Host PC.

CHAPTER 5

RECOMMENDATION AND CONCLUSION

5.1 RECOMENDATION

From the prototype development, I have learned and discovered various ways of measurement, tracking the data, programming, and make a own algorithm. The system now are more intelligent whereas can do a request from the user requirement together the accuracy or measurement, and with the full flexibility. The PEMANTAU will be as important instrument to perform data analysis, to generate the simulation data. The PEMANTAU can help the users, to understand the conditions of renewable energy system like the PV panels in real time. The result not just based on values, but it comes together with analysis and network configuration system. The PV solar generation system can used the data for optimization can vary individual inputs parameters and provide a detailed curve showing characteristic of the current operating data period. It is targeted to be a very comprehensive system, yet it is meant to be affordable but better in term functionality, it is better than the commercial product. The PEMANTAU is a step further from a simple idea monitoring data to the huge possibility on the analysis and measurement design.

5.2 CONCLUSION AND FUTURE WORK

The PEMANTAU - Performance Monitoring And Tracking Prototype For Solar Photovoltaic (PV) Panel Integrated Renewable Energy System is expected to become a very important tool to assist solar researchers in the design and analysis of PV solar power generation system. The objectives of this project have been achieved, whereby the PEMANTAU system has been designed and tested. The PEMANTAU is an important tool for monitoring the performance of solar electricity generating system, for optimum operation. The PEMANTAU can be further improved by extending it for other renewable energy based systems.

REFERENCES

[1] Jonatjam F. Gosse, Todd W. Stafford. *Photovoltaic Systems*. Vol. 1. 1 ed., Edited by Inc. Editorial Staff American Technical Publishers: NJATC Journal, 2007.

[2] James, James &. *Planning and Installing Photovoltaic Systems: A Guide for Installler, Architects and Engineers*
the German Solar Energy Society. Vol. 1 Earthscan Series. Camden High Street: Earthscan, 2005.

[3] Glaser, D. P. (2005). *The Sun's Energy Distribution. Solar Center In-Depth, Ohio University.* http://solarcellcentral.com/solar_page.html

[4] Briney, A. (2008). *Solar Radiation and the Earth's Albedo,* About.com - Education Geography. http://geography.about.com/od/physicalgeography/a/solarradiation.htm

[5] USGCRP (2009). *Global Climate Change Impacts in the United States* . Thomas R. Karl, Jerry M. Melillo, and Thomas C. Peterson (eds.). United States Global Change Research Program. Cambridge University Press, New York, NY, USA.

[6] Dagmar Budikova; C Michael Hogan; Mryka Hall-Beyer, Galal Hassan Galal Hussein, Michael Pidwirny (2012) *"Albedo". In: Encyclopedia of Earth*. Environmental Information Coalition, National Council for Science and the Environment). http://www.eoearth.org/article/Albedo

[7] Shaw, John H. *Solar Radiation*. Department of Physics and Astronomy, 1, (1953): 258. *The Ohio State University, Columbus 10*

[8] Trenberth, K. a. (1997). Natural *Climate Variations 1.2.1 Natural Forcing of the Climate System*, Annual Global Mean Energy Budget. http://www.ipcc.ch/ipccreports/tar/wg1/041.htm

[9] Keogh, W. M. (2001). *Accurate performance measurement of silicon solar cells* (Doctoral dissertation, Australian National University).

[10] Munowitz, M. *Knowing:The Nature of Physical Law: The Nature of Physical Law*: Oxford University Press,
USA, 2005.

[11] Ottinger, R.L., N. Robinson, and V. Tafur. *Compendium of Sustainable Energy Laws*: Cambridge University
Press, 2005.

[12] Edward J. Denecke, J. *Let's Review: Earth Science*: Barron's Educational Series, 2006.

[13] Council, European Renewable Energy. *Renewable Energy in Europe: Markets, Trends and Technologies*: Taylor & Francis, 2012.

[14] Burnett (2002) *The Basic Physics and Design of III-V Multijunction Solar Cell*, Journal of US Department of Energy National Center for Photovoltaics

[15] Antonio Luque and Steven Hegedus (2003). *Handbook of Photovoltaic Science and Engineer*ing. John Wiley and Sons. ISBN 0-471-49196-9.

[16] Electric, Bosch. "Photovoltaics How to Turn Sunlight into Electricity." *Solar Energy Factsheet*, (2011).

[17] Patrina Eiffert, G.J.K. *Building-Integrated Photovoltaic Designs for Commercial and Institutional Structures:*

A Sourcebook for Architects: DIANE Publishing.

[18] Tiwari, G.N., S. Dubey, and J.C.R. Hunt. *Fundamentals of Photovoltaic Modules and Their Applications*: Royal Society of Chemistry, 2009.

[19] *Measurement of Pv Maximum Power Point Tracking Performance*. Netherlands Energy Research Foundation ECN, 1997.

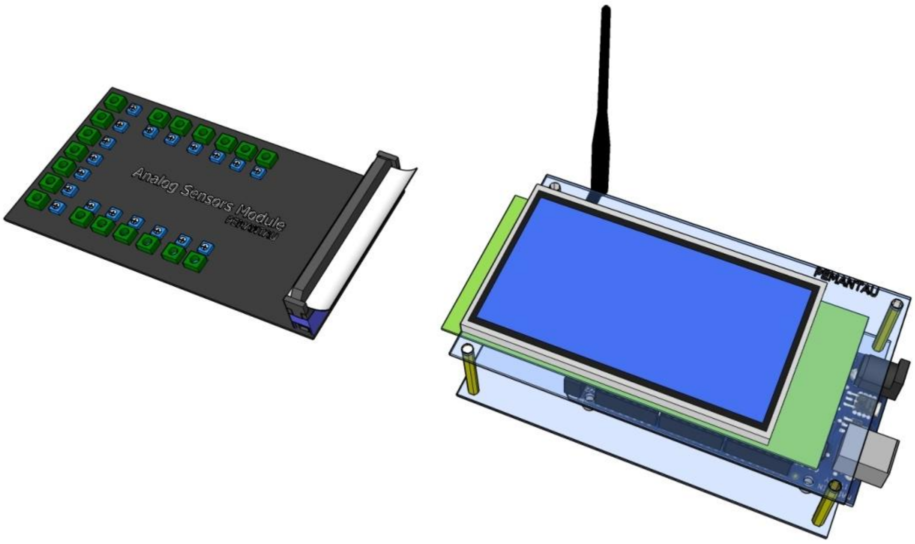
[20] Gordon, J., and International Solar Energy Society. *Solar Energy: The State of the Art : Ises Position Papers*: James & James, 2001.

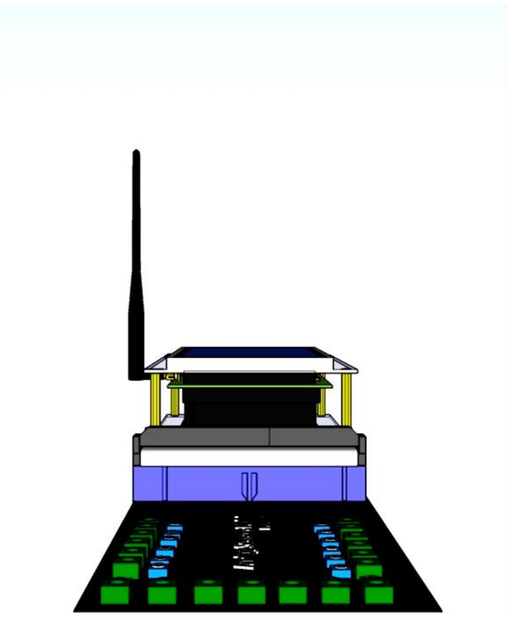
[21] Zade, A. *Z-Theory and Its Applications*: AuthorHouse, 2011.

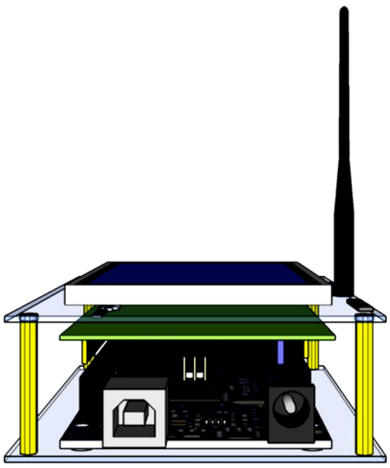
[22] Banzi, M. *Getting Started with Arduino*: O'Reilly Media, 2011.

APPENDICES

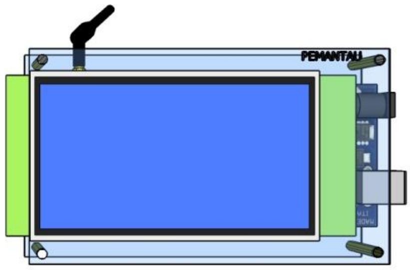
7.1 APPENDIX 1 - PEMANTAU END PRODUCT

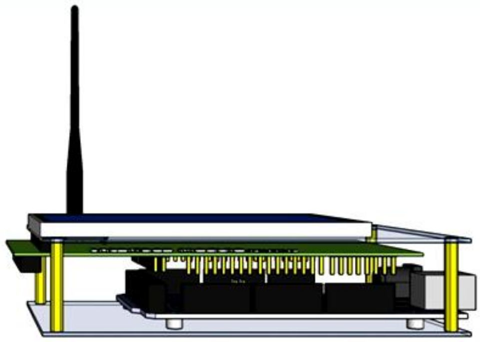


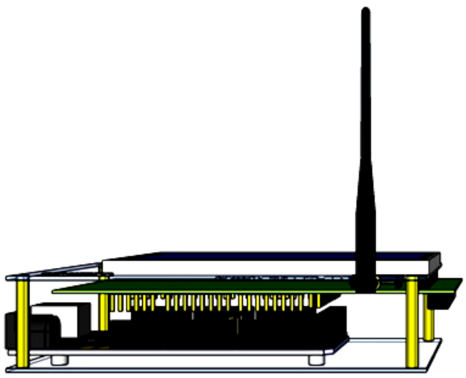




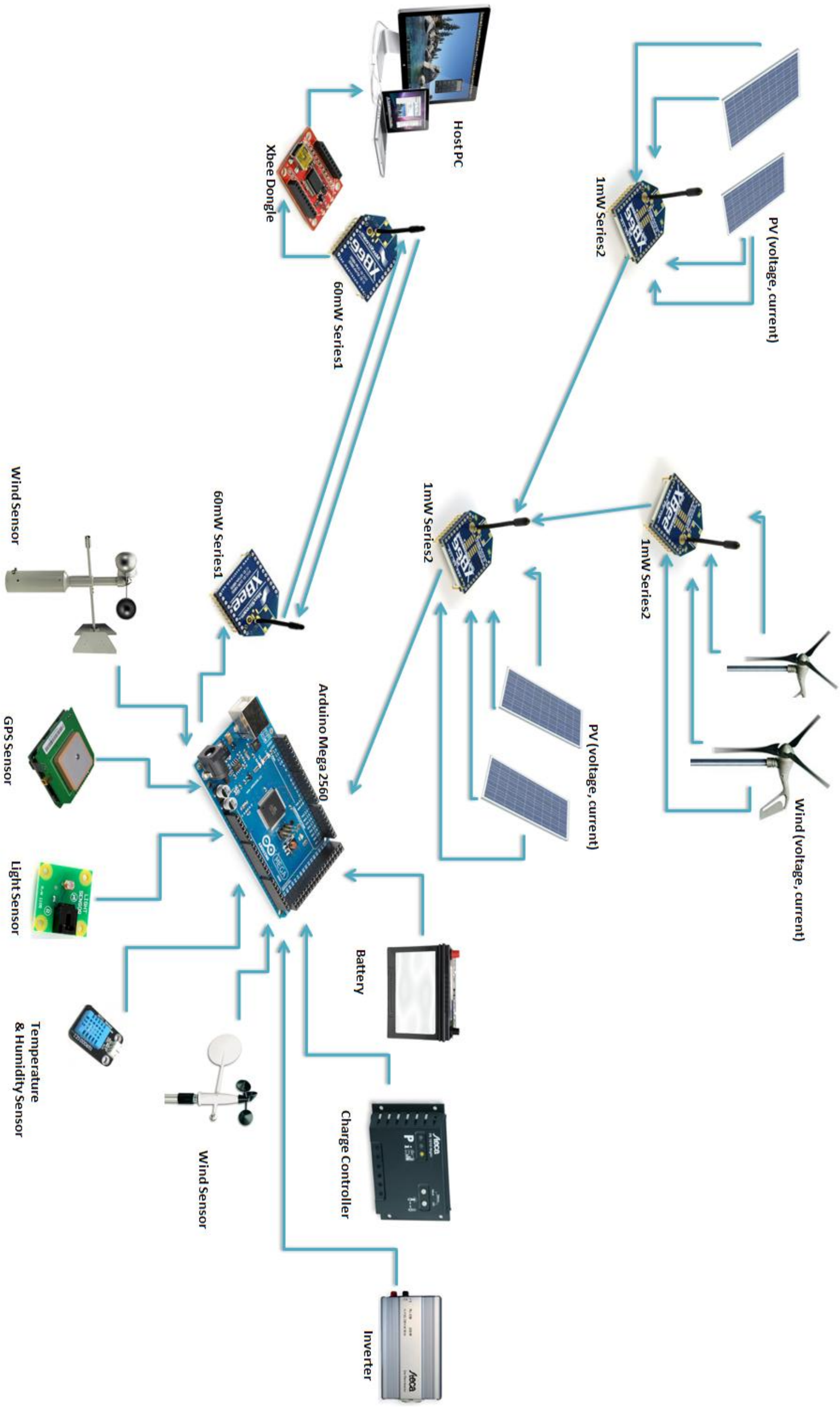

Analog Sensors Module
PEMANTAU


PEMANTAU





7.2 APPENDIX 2 - PEMANTAU HYBRID NETWORK TYPE TOPOLOGY (FUTURE DEVELOPMENT)



7.3 APPENDIX 3 - RETOUCH MODULE SOURCE CODE

```
// Library used for the scripting=============================
#include <stdlib.h>
#include <ITDB02_Touch.h>
#include <ITDB02_Graph16.h>
#include <leOS.h>
#include <SHT1x.h>
#include <SoftwareSerial.h>
#include <floatToString.h>

#define dataPin 8
#define clockPin 9

SoftwareSerial xbee(0, 1); // RX, TX
SHT1x sht1x(dataPin, clockPin);

// Create OS instance=============================================
leOS myOS;

// Library font being used=========================================
extern uint8_t SmallFont[];
extern uint8_t BigFont[];
extern uint8_t DotMatrix_M[];
extern uint8_t Sinclair_M[];
extern uint8_t Sinclair_S[];
extern uint8_t OCR_A_Extended_M[];
extern uint8_t Dingbats1_XL[];

// Initialize touch module pins
ITDB02        myGLCD(38,39,40,41);
ITDB02_Touch  myTouch(6,5,4,3,2);

// Global variable and constant
int x, y;
char stCurrent[20]="";
int stCurrentLen=0;
char stLast[20]="";
char direction = -1;

// Sensors variable [ Total ADC = 16 ]
float sensor_current_PV = 0;
float sensor_temperature_PV = 0;
float sensor_humidities_PV = 0;
float sensor_wind_PV = 0;
float sensor_light_PV = 0;
float sensor_voltage_INV = 0;
float sensor_voltage_BAT = 0;
float sensor_voltage1_PV = 40;
float sensor_voltage2_PV = 0;
float sensor_voltage3_PV = 0;
float sensor_voltage4_PV = 0;
float sensor_voltage5_PV = 0;
float sensor_voltage6_PV = 0;
float sensor_voltage7_PV = 0;
float sensor_voltage8_PV = 0;
float temp_c;
float temp_f;
float humidity;

// Process Sensor variable
float data_current_PV = 0;
int data_temperature_PV = 0;
int data_humidities_PV = 0;
int data_wind_PV = 0;
int data_light_PV = 0;
float data_voltage_INV = 0;
float data_voltage_BAT = 0;
```

```
float data_voltage_PV_total = 0;
float data_voltage1_PV = 0;
float data_voltage2_PV = 0;
float data_voltage3_PV = 0;
float data_voltage4_PV = 0;
float data_voltage5_PV = 0;
float data_voltage6_PV = 0;
float data_voltage7_PV = 0;
float data_voltage8_PV = 0;

// Conversion data variable
float data_percent_BAT = 0;
float data_max_power = 0;
float data_max_voltage = 0;
float data_max_current = 0;
String char_transmit;

// float buffer
char test[25];
char test1[25];

int sunlight_count = 0;
int real_time_condition = 1;

String char_array_enviroment[4];
String char_array_voltage_PV[8];
String char_array_conversion[3];
String char_array_max[1];
String char_array_power_max[1];

String string_array_enviroment;
String string_array_PV_voltage;
String string_array_conversion;
String string_array_max;
String string_module_setting;


const int numReadings = 30;
float readings[numReadings];      // the readings from the analog input
int index = 0;                   // the index of the current reading
float total = 0;                  // the running total
float average = 0;                // the average

float currentValue = 0;

void setup()
{
  myGLCD.InitLCD(LANDSCAPE);
  myGLCD.clrScr();
  myTouch.InitTouch(LANDSCAPE);
  myTouch.setPrecision(PREC_MEDIUM);

  myOS.begin();

  myOS.addTask(touch_workspace,50);
  myOS.addTask(data_captures,10);
  myOS.addTask(data_processing,20);
  myOS.addTask(transmit_data_xbee,30);
  //myOS.addTask(data_process_pv,20);
  //myOS.addTask(testrt,10);

  //myOS.pauseTask(testrt);
   myOS.pauseTask(touch_workspace);
  myOS.pauseTask(data_captures);
  myOS.pauseTask(data_processing);
  myOS.pauseTask(transmit_data_xbee);
  //myOS.pauseTask(data_process_pv);
```

```
  system_boot();
  display_workspace();
  display_computation();
  xbee.begin(9600);

  for (int thisReading = 0; thisReading < numReadings; thisReading++)
    readings[thisReading] = 0;

  //Serial.begin(9600);
myOS.restartTask(touch_workspace);
  myOS.restartTask(data_captures);
  myOS.restartTask(data_processing);
  myOS.restartTask(transmit_data_xbee);


  //myOS.restartTask(testrt);
}

void loop()
{
     if(real_time_condition==1){
       temperature_sensor_manual();
     }

     if((y>=213)&&(y<=234)){
        if((x>=280)&&(x<=314)){
          if(real_time_condition == 1){
            real_time_condition = 0;
            //myOS.pauseTask(touch_workspace);
            myOS.pauseTask(data_captures);
            myOS.pauseTask(data_processing);
            myOS.pauseTask(data_process_pv);
          }
          else{
            real_time_condition = 1;
           // myOS.restartTask(touch_workspace);
            myOS.restartTask(data_captures);
            myOS.restartTask(data_processing);

          }
          }

      }

      if ((y>=13) && (y<=30)){
        if((x>=148) && (x<=200)){
          display_workspace();
          display_computation();
          real_time_condition = 1;
          x=0;
          y=0;
          myOS.restartTask(data_captures);
          myOS.restartTask(data_processing);
        }
         if((x>=201) && (x<=259)){
          real_time_condition = 0;
          x=0;
          y=0;
          myOS.pauseTask(data_captures);
          myOS.pauseTask(data_processing);
          myOS.pauseTask(data_process_pv);
          display_workspace();
          display_network();
        }

         if((x>=260) && (x<=319)){
          real_time_condition = 0;
          x=0;
```

```
            y=0;
            myOS.pauseTask(data_captures);
            myOS.pauseTask(data_processing);
            myOS.pauseTask(data_process_pv);
            display_workspace();
            display_setting();
          }
           if((x>=110) && (x<=150)){
            real_time_condition = 0;
            x=0;
            y=0;
            myOS.pauseTask(data_captures);
            myOS.pauseTask(data_processing);
            display_workspace();
            display_PV_list();
          }
        }
  //touch_workspace();
  //delay(100000);
}

void system_boot(){
  int i = 0;
  int j = 0;
  // Start boot the loading system
  myGLCD.clrScr();
  myGLCD.fillScr(0,0,0);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(DotMatrix_M);
  myGLCD.setBackColor(0,0,0);
  myGLCD.print("1",  12, 103);
  myGLCD.setFont(Sinclair_M);
  myGLCD.print("PEMANTAU",30,102);

  myGLCD.setFont(Sinclair_S);
  delay(1000);
  myGLCD.setBackColor(239,0,255);
  myGLCD.setColor(255,255,255);
  myGLCD.print("Solar Intergrated Simulation Module",30,120);

  myGLCD.setBackColor(0,0,0);
  myGLCD.setColor(255,255,255);
  myGLCD.print("Loading",135 + j, 164);
  myGLCD.setColor(0,0,0);
  myGLCD.fillRoundRect (110 + j, 180, 220 + j, 186);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (110 + j, 180, 220 + j, 186);

  for(i=110;i<220;i++){
    delay(10);
    myGLCD.setColor(0,255,239);
    myGLCD.fillRoundRect (110 + j, 180, i, 186);
  }
}

void temperature_sensor_manual(){
     temp_c = sht1x.readTemperatureC();
    humidity = sht1x.readHumidity();

    data_temperature_PV = temp_c;
    data_humidities_PV = humidity;
}

void touch_workspace(){

//    temp_c = sht1x.readTemperatureC();
//    humidity = sht1x.readHumidity();
//
```

```
//    data_temperature_PV = temp_c;
//    data_humidities_PV = humidity;

    if (myTouch.dataAvailable()){
      myTouch.read();
      x=myTouch.getX();
      y=myTouch.getY();
      //myGLCD.setBackColor(0,0,0);
      //myGLCD.setColor(255,255,255);
      //myGLCD.printNumF(x,3, 190, 220);
      //myGLCD.printNumF(y,3, 190, 230);


    }
        //x=0;
        //y=0;

}

void transmit_data_xbee(){

  string_array_enviroment = char_array_enviroment[1] + char_array_enviroment[2] +
char_array_enviroment[3] + char_array_enviroment[4] + char_array_power_max[1] ;
  string_array_PV_voltage = char_array_voltage_PV[1] + char_array_voltage_PV[2] +
char_array_voltage_PV[3] + char_array_voltage_PV[4] + char_array_voltage_PV[5] +
char_array_voltage_PV[6] +
    char_array_voltage_PV[7] + char_array_voltage_PV[8];
  string_array_conversion = char_array_conversion[1] + char_array_conversion[2] +
char_array_conversion[3];
  string_array_max = char_array_max[1];
  string_module_setting = "||PV:8PV||Mod:2Mod";

  xbee.println("|NW" + string_array_enviroment + string_array_conversion +
string_array_PV_voltage + string_array_max + string_module_setting);

}

void data_captures(){

  //Sensor analog to digital retrieving=======================
  sensor_voltage1_PV = analogRead(0);
  sensor_voltage2_PV = analogRead(1);
  sensor_voltage3_PV = analogRead(2);
  sensor_voltage4_PV = analogRead(3);
  sensor_voltage5_PV = analogRead(4);
  sensor_voltage6_PV = analogRead(5);
  sensor_voltage7_PV = analogRead(6);
  sensor_voltage8_PV = analogRead(7);
  sensor_light_PV = analogRead(8);
  sensor_voltage_INV = analogRead(9);
  sensor_voltage_BAT = analogRead(10);
  sensor_current_PV = analogRead(11);
  sensor_wind_PV = analogRead(12);

  //temp_c = sht1x.readTemperatureC();
  //temp_f = sht1x.readTemperatureF();
  //humidity = sht1x.readHumidity();

  //Enviroment processing data
  data_light_PV = sensor_light_PV;
  data_temperature_PV = temp_c;
  data_humidities_PV = humidity;
  data_wind_PV = random(0,100);

  //PV voltage module processing data
  data_voltage1_PV = ((sensor_voltage1_PV)*5)/1023;
  data_voltage2_PV = ((sensor_voltage2_PV)*5)/1023;
  data_voltage3_PV = ((sensor_voltage3_PV)*5)/1023;
```

```
  data_voltage4_PV = ((sensor_voltage4_PV)*5)/1023;
  data_voltage5_PV = ((sensor_voltage5_PV)*5)/1023;
  data_voltage6_PV = ((sensor_voltage6_PV)*5)/1023;
  data_voltage7_PV = ((sensor_voltage7_PV)*5)/1023;
  data_voltage8_PV = ((sensor_voltage8_PV)*5)/1023;

  //Conversion module processing data
  data_voltage_INV = ((sensor_voltage_INV)*5)/1023;
  data_voltage_BAT = ((sensor_voltage_BAT)*5)/1023;
  data_percent_BAT = ((data_voltage_BAT)*100)/5;

  total= total - readings[index];
  readings[index] = analogRead(15); //Raw data reading
  readings[index] = (readings[index]-510)*5/1024/0.04-0.04;//Data processing:510-
raw data from analogRead when the input is 0; 5-5v; the first 0.04-
0.04V/A(sensitivity); the second 0.04-offset val;
  total= total + readings[index];
  index = index + 1;
  if (index >= numReadings){
      index = 0;
  }
  average = total/numReadings;   //Smoothing algorithm
(http://www.arduino.cc/en/Tutorial/Smoothing)
  currentValue= average * (-1);
  data_max_current = currentValue;//random(0,1.2);
  data_max_voltage = (data_voltage1_PV + data_voltage2_PV + data_voltage3_PV +
data_voltage4_PV + data_voltage5_PV + data_voltage6_PV + data_voltage7_PV +
data_voltage8_PV);
  data_max_power = data_max_voltage * data_max_current;
  //============================================================
}

void data_processing(){

  //
  char_array_enviroment[1] = "||E1:" + String(data_light_PV) + "E";
  char_array_enviroment[2] = "||E2:" + String(data_temperature_PV) + "E";
  char_array_enviroment[3] = "||E3:" + String(data_humidities_PV) + "E";
  char_array_enviroment[4] = "||E4:" +
String(floatToString(test1,data_wind_PV,3,2,true)) + "E";


//=================================================================================
===========================

  char_array_voltage_PV[1] = "||V1:" +
String(floatToString(test,data_voltage1_PV,3,2,true)) + "V";
  char_array_voltage_PV[2] = "||V2:" +
String(floatToString(test,data_voltage2_PV,3,2,true)) + "V";
  char_array_voltage_PV[3] = "||V3:" +
String(floatToString(test,data_voltage3_PV,3,2,true)) + "V";
  char_array_voltage_PV[4] = "||V4:" +
String(floatToString(test,data_voltage4_PV,3,2,true)) + "V";
  char_array_voltage_PV[5] = "||V5:" +
String(floatToString(test,data_voltage5_PV,3,2,true)) + "V";
  char_array_voltage_PV[6] = "||V6:" +
String(floatToString(test,data_voltage6_PV,3,2,true)) + "V";
  char_array_voltage_PV[7] = "||V7:" +
String(floatToString(test,data_voltage7_PV,3,2,true)) + "V";
  char_array_voltage_PV[8] = "||V8:" +
String(floatToString(test,data_voltage8_PV,3,2,true)) + "V";


//=================================================================================
===========================

  char_array_conversion[1] = "||C1:" +
String(floatToString(test,data_voltage_INV,3,2,true)) + "C";
```

```
  char_array_conversion[2] = "||C2:" +
String(floatToString(test,data_voltage_BAT,3,2,true)) + "C";
  char_array_conversion[3] = "||C3:" +
String(floatToString(test,data_percent_BAT,3,2,true)) + "C";


//==================================================================================
============================

  char_array_max[1] = "||M1:" +
String(floatToString(test,data_max_voltage,3,2,true)) + "M";
  char_array_power_max[1] = "||P1:"+
String(floatToString(test,data_max_power,3,2,true)) + "P";



  //string_array_max = char_array_max[0] + char_array_max[1];

  //Display the continuous data to Status dashboard================
  myGLCD.setBackColor(14,36,107);
  myGLCD.setColor(255,255,255);
  //Power continuous data
  myGLCD.setFont(Sinclair_M);
  myGLCD.print("   ", 5,58);
  myGLCD.printNumI(data_max_power, 5,58);
  //Voltage continuous data
  myGLCD.setFont(Sinclair_M);
  myGLCD.print("    ",91,58);
  myGLCD.printNumI(data_max_voltage,91,58);
  myGLCD.print("V",147,58);
  //myGLCD.printNumF(data_voltage7_PV,2,91,58);
  //Current continuous data
  myGLCD.setFont(Sinclair_M);
  myGLCD.print("       ",230,58);
  myGLCD.printNumF(data_max_current,2, 190,58);
  myGLCD.print("A",270,58);
  //Battery continuous data
  myGLCD.setFont(Sinclair_M);
  myGLCD.printNumI(data_percent_BAT, 145,105);
  //Battery loading bar interface
  int bar_change = 0;
  bar_change = (data_percent_BAT * 141)/100;
  myGLCD.setColor(0,0,0);
  myGLCD.fillRect (5, 110, 141, 114);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect (5, 110, 141, 114);
  myGLCD.setColor(0,255,100);
  myGLCD.fillRect (5, 110, bar_change, 114);
  //Inverter voltage AC continuous data
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_M);
  myGLCD.printNumF(data_voltage_INV,2, 210,110);
  //Sunlight continuous data

  //Temperature continuous data

  myGLCD.setFont(Sinclair_S);
  myGLCD.printNumI(data_temperature_PV,270,162);
  myGLCD.printNumI(data_humidities_PV,265,174);
  //Wind continuous data
  myGLCD.printNumF(data_wind_PV,2,220,200);


//=====================================================
  int cnv_light =0;
  int cnv_check =0;
  //int sunlight_data = getMag(cnv_light);
```

```
  cnv_light = ((1000-data_light_PV)*100)/1000;
  cnv_check = ((cnv_light)*45)/100;
  //cnv_light = data_light_PV*45;
  myGLCD.setColor(14,36,107);
  myGLCD.fillRect(11,173,90,220-cnv_check);

  myGLCD.setColor(255,255,255);
  myGLCD.drawLine(10,175,10,220);
  myGLCD.drawLine(10,220,130,220);
  myGLCD.drawLine(130,175,130,220);

  if(cnv_light>36){
    myGLCD.setColor(255,255,0);
  }
  else if((cnv_light<36)||(cnv_light>27)){
    myGLCD.setColor(213,213,0);
  }
  else if((cnv_light<27)||(cnv_light>18)){
    myGLCD.setColor(185,185,0);
  }
  else if((cnv_light<18)||(cnv_light>9)){
    myGLCD.setColor(147,147,0);
  }
  else if((cnv_light<9)||(cnv_light>0)){
  myGLCD.setColor(117,117,0);
  }

  myGLCD.fillRect(10,220-cnv_check,90,220);
  //myGLCD.fillRect(10,220-cnv_light,90,220);
  cnv_light = ((data_light_PV)*100)/1000;
  myGLCD.setColor(255,255,255);
 myGLCD.setFont(Sinclair_M);
  myGLCD.printNumI(cnv_light,80,200);
}


void display_workspace(){
  //myGLCD.clrScr();
  myGLCD.fillScr(14,36,107);
  myGLCD.setFont(Sinclair_S);
  myGLCD.setColor(65,136,211);
  myGLCD.fillRect(0,0,10,13);

  myGLCD.setColor(13,88,166);
  myGLCD.fillRect(10,0,15,13);

  myGLCD.setColor(0,0,0);
  myGLCD.fillRect(15,0,18,13);

  myGLCD.setColor(4,55,108);
  myGLCD.fillRect(18,0,305,13);

  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(4,55,108);
  myGLCD.print("1PEMANTAU - STATUS",30,1);

  myGLCD.setColor(255,197,1);
  myGLCD.fillRect(305,0,319,13);

  myGLCD.setColor(0,145,255);
  myGLCD.fillRect(260,13,319,30);
  myGLCD.setColor(75,75,75);
  myGLCD.drawRect (260, 13, 319, 30);

  myGLCD.setColor(0,145,255);
  myGLCD.fillRect(201,13,260,30);
  myGLCD.setColor(75,75,75);
  myGLCD.drawRect (201, 13, 260, 30);
```

```
  myGLCD.setColor(0,145,255);
  myGLCD.fillRect(148,13,201,30);
  myGLCD.setColor(75,75,75);
  myGLCD.drawRect (148, 13, 201, 30);

  myGLCD.setColor(0,145,255);
  myGLCD.fillRect(110,13,150,30);
  myGLCD.setColor(75,75,75);
  myGLCD.drawRect (110, 13, 150, 30);

  myGLCD.setBackColor(0,145,255);
  myGLCD.setColor(255,255,55);
  myGLCD.print("SETTING",263,18);
  myGLCD.print("NETWORK",204,18);
  myGLCD.print("STATUS",152,18);
  myGLCD.print("PVS",120,18);

  myGLCD.setColor(255,60,60);
  myGLCD.fillRect(0,13,110,30);
  myGLCD.setColor(75,75,75);
  myGLCD.drawRect (0, 13, 110, 30);

  myGLCD.setBackColor(255,60,60);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Ver.0.4a",4,18);

}



void display_computation(){
  myGLCD.setBackColor(14,36,107);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Power:",5,45);
  myGLCD.setFont(Sinclair_M);
  myGLCD.print("200W", 5,58);

  myGLCD.drawLine(5, 80, 82, 80);
  myGLCD.drawLine(82, 45, 82, 80);

  myGLCD.setFont(Sinclair_S);
  myGLCD.print("PV Volt.:",91,45);

  myGLCD.setFont(Sinclair_S);
  myGLCD.print("PV Current:",190,45);
  myGLCD.setFont(Sinclair_M);
  myGLCD.print("", 190,58);

  myGLCD.drawLine(91, 80, 314, 80);
  myGLCD.drawLine(314, 45, 314, 80);

  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Cnn.PV: 8 unit",180,33);
  myGLCD.drawLine(160, 41, 314, 41);
  myGLCD.drawLine(314, 33, 314, 41);
  myGLCD.drawLine(160, 33, 160, 41);

  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Batt. Voltage:",5,95);
  myGLCD.setFont(Sinclair_M);
  myGLCD.print("30%", 145,105);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Cp.Volt.:12V", 5,120);

  myGLCD.setFont(Sinclair_S);
```

```
myGLCD.print("Invtr. Volt.:",210,95);
myGLCD.setFont(Sinclair_M);
myGLCD.print("100V", 210,110);
myGLCD.setFont(Sinclair_S);
myGLCD.print("(a.c)", 273,118);

myGLCD.setColor(0,0,0);
myGLCD.fillRect (5, 110, 141, 114);
myGLCD.setColor(255, 255, 255);
myGLCD.drawRect (5, 110, 141, 114);
myGLCD.setColor(0,255,100);
myGLCD.fillRect (5, 110, 80, 114);

myGLCD.setColor(255,255,255);
myGLCD.drawLine(5, 132, 200, 132);
myGLCD.drawLine(200, 90, 200, 132);

myGLCD.setColor(255,255,255);
myGLCD.drawLine(210, 132, 314, 132);
myGLCD.drawLine(314, 90, 314, 132);

myGLCD.setFont(Sinclair_S);
myGLCD.print("Enviroment:", 5,142);
myGLCD.drawLine(5, 150, 314, 150);

myGLCD.print("Sunlight:", 5,165);
myGLCD.setColor(0,255,100);

myGLCD.setColor(255,255,255);
myGLCD.drawLine(10,175,10,220);
myGLCD.drawLine(10,220,130,220);
myGLCD.drawLine(130,175,130,220);

myGLCD.setColor(172,172,172);
myGLCD.drawLine(10,180,130,180);
myGLCD.drawLine(10,195,130,195);
myGLCD.drawLine(10,210,130,210);

myGLCD.setColor(217,240,64);
//  myGLCD.setFont(Sinclair_M);
//  myGLCD.printNumI(data_light_PV,80,200);
//myGLCD.print("00%",80,200);

myGLCD.setFont(Sinclair_S);
myGLCD.setColor(255,255,255);
myGLCD.print("Temperature:",176,162);
myGLCD.setColor(255,255,255);
myGLCD.print("Humidities: ",176,174);
myGLCD.setColor(0,0,0);
myGLCD.fillRoundRect(155,160,160,185);
myGLCD.setColor(239, 87, 64);
myGLCD.fillRoundRect(155,172,160,185);
myGLCD.setColor(255,255,255);
myGLCD.drawRoundRect(155,160,160,185);

myGLCD.setColor(255,255,255);
myGLCD.print("Wind:",180,200);

myGLCD.setColor(64,129,240);
myGLCD.drawLine(156,205,161,205);
myGLCD.setColor(64, 217, 240);
myGLCD.drawLine(148,208,151,208);
myGLCD.setColor(64,129,240);
myGLCD.drawLine(150,212,160,212);
myGLCD.setColor(64, 217, 240);
myGLCD.drawLine(152,210,166,210);

myGLCD.setColor(255, 255, 255);
```

```
  myGLCD.drawLine(5, 233, 314, 233);
  myGLCD.drawLine(314, 150, 314, 233);

  myGLCD.setColor(54,54,54);
  myGLCD.fillRect(280,213,314,233);
  myGLCD.setColor(255,255,255);
  myGLCD.drawRect (280, 213, 314, 233);
  myGLCD.setFont(Dingbats1_XL);
  myGLCD.print("r",283,215);

}

void display_setting(){
  myGLCD.setBackColor(0,0,0);
  myGLCD.setColor(0,0,0);
  //myGLCD.setColor(181,230,29);
  myGLCD.fillRect(0,31,319,239);

  myGLCD.setColor(93,93,93);
  myGLCD.fillRect(302,31,320,240);

  myGLCD.setColor(60,60,60);
  myGLCD.fillRect(0,31,303,48);


  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(60,60,60);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("System Details",6,35);
  myGLCD.drawLine(1,48,301,48);

  myGLCD.setColor(255,255,255);
  myGLCD.drawLine(302,31,302,240);


  myGLCD.setColor(255,255,255);
  myGLCD.fillRect(303,31,320,42);

  myGLCD.setColor(0,0,0);
  myGLCD.drawLine(310,34,305,39);
  myGLCD.drawLine(310,34,315,39);
  myGLCD.drawLine(305,39,315,39);

  myGLCD.setColor(0,0,0);
  myGLCD.fillRect(0,49,301,133);
  myGLCD.setColor(255,255,255);
  myGLCD.drawRect(0,49,301,133);

  myGLCD.setColor(60,60,60);
  myGLCD.fillRect(0,134,301,151);
  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(60,60,60);

  myGLCD.setColor(255,255,255);
  myGLCD.print("Tools",6,138);
  myGLCD.drawLine(0,151,301,151);

  myGLCD.setColor(0,0,0);
  myGLCD.fillRoundRect (9, 159, 69, 199);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (9, 159, 69, 199);
  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(0,0,0);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Convs.", 13,180);
  myGLCD.print("Sensor", 12,190);

  myGLCD.setColor(0,0,0);
```

```
  myGLCD.fillRoundRect (74, 159, 134, 199);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (74, 159, 134, 199);
  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(0,0,0);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Env.", 85,180);
  myGLCD.print("Sensor", 84,190);

  myGLCD.setColor(0,0,0);
  myGLCD.fillRoundRect (139, 159, 199, 199);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (139, 159, 199, 199);
  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(0,0,0);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("PV.", 145,180);
  myGLCD.print("Sensor", 144,190);

  myGLCD.setColor(0,0,0);
  myGLCD.fillRoundRect (204, 159, 264, 199);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (204, 159, 264, 199);
  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(0,0,0);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Style", 209,190);


  myGLCD.setColor(255,255,255);
  myGLCD.print("Total Connected PV: 8  Zigbee: ON", 5,60);
  myGLCD.print("Battery Max: 12        Inverter: Regulated", 5,75);
  myGLCD.print("Charge Controler: Mod  Light Max: 100", 5,90);
  myGLCD.print("Env.Sensors: 4", 5,105);


}

void display_network(){
  myGLCD.setBackColor(14,36,107);
  myGLCD.setColor(43,214,43);
  myGLCD.setFont(Dingbats1_XL);
  myGLCD.print("4", 10,40);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Network Zigbee Module", 28,45);
  myGLCD.setColor(43,214,43);
  myGLCD.drawLine(25,54,250,54);

  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print(">Connection Status:", 25,70);
  myGLCD.drawLine(25,79,180,79);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Ok.. SYNCHRONIZE!", 25,84);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print(">Data Transfer:", 25,110);
  myGLCD.drawLine(25,119,150,119);
  myGLCD.setColor(255,255,255);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print("Ok.. GOOD!", 25,124);
  myGLCD.setColor(255,255,255);
  myGLCD.setFont(Sinclair_S);
  myGLCD.print(">Connectivity:", 25,150);
  myGLCD.drawLine(25,159,140,159);
```

```
  myGLCD.print("Range:", 25,164);

  myGLCD.setColor(60,60,60);
  myGLCD.fillRect(0,194,319,221);
  myGLCD.setColor(255,255,255);
  myGLCD.drawRect(0,194,319,221);
  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(60,60,60);
  myGLCD.setColor(255,255,255);
  myGLCD.fillRect(280,194,319,221);
  myGLCD.setColor(255,255,255);
  myGLCD.drawRect(280,194,319,221);
  myGLCD.setFont(Sinclair_S);
  myGLCD.setBackColor(60,60,60);
  myGLCD.print("Transfer:",200,198);
  myGLCD.setColor(0,0,0);
  myGLCD.setBackColor(255,255,255);
  myGLCD.print("OK",290,198);
}

void display_PV_list(){

  myGLCD.setColor(0,0,0);
  myGLCD.fillRect(0,33,320,240);
  myGLCD.setColor(255,255,255);
  myGLCD.drawRect(0,33,320,240);

  myGLCD.setFont(Sinclair_S);

  myGLCD.setColor(255,255,255);
  myGLCD.fillRoundRect (20, 40, 30, 70);
  myGLCD.setColor(15,255,15);
  myGLCD.fillRoundRect (20, 50, 30, 70);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (20, 40, 30, 70);

  myGLCD.setFont(SmallFont);
  myGLCD.setColor(255,255,255);
  myGLCD.setBackColor(0,0,0);
  myGLCD.print("Sensor V1:",35, 40);
  myGLCD.print("Voltage:",35, 50);
  myGLCD.print("Percent: 50%",35, 60);


  myGLCD.setColor(255,255,255);
  myGLCD.fillRoundRect (20, 90, 30, 120);
  myGLCD.setColor(15,255,15);
  myGLCD.fillRoundRect (20, 100, 30, 120);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (20, 90, 30, 120);
  myGLCD.print("Sensor V2:",35, 90);
  myGLCD.print("Voltage: 2.5 V",35, 100);
  myGLCD.print("Percent: 50%",35, 110);


  myGLCD.setColor(255,255,255);
  myGLCD.fillRoundRect (20, 140, 30, 170);
  myGLCD.setColor(15,255,15);
  myGLCD.fillRoundRect (20, 150, 30, 170);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (20, 140, 30, 170);
  myGLCD.print("Sensor V3:",35, 140);
  myGLCD.print("Voltage: 2.5 V",35, 150);
  myGLCD.print("Percent: 50%",35, 160);


  myGLCD.setColor(255,255,255);
  myGLCD.fillRoundRect (20, 190, 30, 220);
```

```
  myGLCD.setColor(15,255,15);
  myGLCD.fillRoundRect (20, 200, 30, 220);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (20, 190, 30, 220);
  myGLCD.print("Sensor V4:",35, 190);
  myGLCD.print("Voltage: 2.5 V",35, 200);
  myGLCD.print("Percent: 50%",35, 210);

  myGLCD.print("Output Current:",170, 40);
  myGLCD.setColor(255,255,255);
  myGLCD.fillRoundRect (170, 60, 280, 80);
  myGLCD.setColor(231,58,150);
  myGLCD.fillRoundRect (170, 60, 200, 80);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (170, 60, 280, 80);
  myGLCD.print("Value: 1.0A",170, 90);
  myGLCD.print("Status: Normal",170, 100);

  myGLCD.print("Output Power:",170, 120);
  myGLCD.setColor(255,255,255);
  myGLCD.fillRoundRect (170, 140, 280, 160);
  myGLCD.setColor(255,248,64);
  myGLCD.fillRoundRect (170, 140, 200, 160);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (170, 140, 280, 160);
  myGLCD.print("Value: 10W",170, 170);
  myGLCD.print("Status: Normal",170, 180);

}

void data_process_pv(){

}
```

7.4 APPENDIX 4 - PEMANTAU - ADMIN SYSTEM SOURCE CODE

> frm_main.vb

```
Imports System
Imports System.ComponentModel
Imports System.Threading
Imports System.IO.Ports
Imports System.Windows.Forms.DataVisualization.Charting

Public Class frm_main
    Inherits System.Windows.Forms.Form
    Dim states As Integer
    Dim myPort As Array  'COM Ports detected on the system will be stored here
    Delegate Sub SetTextCallback(ByVal [text] As String) 'Added to prevent threading
errors during receiveing of data
    Dim dataReceived As RichTextBox
    Private random As New Random()
    Private pointIndex As Integer = 0
    Dim numberOfPointsInChart As Integer = 200 '200
    Dim numberOfPointsAfterRemoval As Integer = 75 '150

    Private comm As New CommManager()
    Private Sub frm_main_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim myControl As New frm_output
        'Me.pnPemantau.Controls.Add(myControl)
        dataReceived = New RichTextBox
        power(0) = New DataTable
        power(0).Columns.Add("Time")
        power(0).Columns.Add("Value")

        For i As Integer = 0 To 1
            battery(i) = New DataTable
            battery(i).Columns.Add("Time")
            battery(i).Columns.Add("Value")
        Next
        For i As Integer = 0 To 8
            enviroment(i) = New DataTable
            enviroment(i).Columns.Add("Time")
            enviroment(i).Columns.Add("Value")
        Next
        For j As Integer = 0 To 8
            voltage(j) = New DataTable
            voltage(j).Columns.Add("Time")
            voltage(j).Columns.Add("Value")
        Next
        For j As Integer = 0 To 3
            chrgecn(j) = New DataTable
            chrgecn(j).Columns.Add("Time")
            chrgecn(j).Columns.Add("Value")
        Next
        For j As Integer = 0 To 3
            inv(j) = New DataTable
            inv(j).Columns.Add("Time")
            inv(j).Columns.Add("Value")
        Next

        serial_connect()
        sun_elevation()
```

```vb
        cn_pv_array.Series(0).ChartType = SeriesChartType.StackedBar
        ' Show point labels
        'cn_pv_array.Series(0).IsValueShownAsLabel = True
        'cn_pv_array.ChartAreas(0).Area3DStyle.Enable3D = True

        'conversion_module()
    End Sub

    Private Sub ToggelPanel(ByVal whichPanel As Panel)
        Dim startPoint As New Point(10, 10)
        whichPanel.Visible = Not whichPanel.Visible

        For Each pnl As Panel In Me.Controls.OfType(Of Panel)(). _
                                 Where(Function(x) x.Visible). _
                                 OrderBy(Function(x) x.TabIndex)
            pnl.Location = startPoint
            startPoint.Y += pnl.Height + 4
        Next
    End Sub

    Private Sub btn_status_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btn_status.Click
        If states <> 1 Then
            states = 1
            Me.btn_status.Image = My.Resources.btn_status_d
            Me.btn_status.Location = New Point(138, 33)
            Me.btn_analysis.Image = My.Resources.btn_analysis_n
            Me.btn_analysis.Location = New Point(234, 31)
            Me.btn_history.Image = My.Resources.btn_history_n
            Me.btn_history.Location = New Point(330, 31)
            Me.btn_network.Image = My.Resources.btn_network_n
            Me.btn_network.Location = New Point(415, 31)
            Me.btn_preference.Image = My.Resources.btn_preference_n
            Me.btn_preference.Location = New Point(505, 31)
            'Me.pnPemantau.Controls.Clear()
            'Me.pnPemantau.Controls.Add(New frm_output)
        End If
    End Sub

    Private Sub btn_analysis_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btn_analysis.Click
        If states <> 2 Then
            states = 2
            Me.btn_status.Image = My.Resources.btn_status_n
            Me.btn_status.Location = New Point(138, 31)
            Me.btn_analysis.Image = My.Resources.btn_analysis_d
            Me.btn_analysis.Location = New Point(234, 33)
            Me.btn_history.Image = My.Resources.btn_history_n
            Me.btn_history.Location = New Point(330, 31)
            Me.btn_network.Image = My.Resources.btn_network_n
            Me.btn_network.Location = New Point(415, 31)
            Me.btn_preference.Image = My.Resources.btn_preference_n
            Me.btn_preference.Location = New Point(505, 31)

        End If
    End Sub

    Private Sub btn_history_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btn_history.Click
        If states <> 3 Then
            states = 3
            Me.btn_status.Image = My.Resources.btn_status_n
```

```
            Me.btn_status.Location = New Point(138, 31)
            Me.btn_analysis.Image = My.Resources.btn_analysis_n
            Me.btn_analysis.Location = New Point(234, 31)
            Me.btn_history.Image = My.Resources.btn_history_d
            Me.btn_history.Location = New Point(330, 33)
            Me.btn_network.Image = My.Resources.btn_network_n
            Me.btn_network.Location = New Point(415, 31)
            Me.btn_preference.Image = My.Resources.btn_preference_n
            Me.btn_preference.Location = New Point(505, 31)
        End If
    End Sub

    Private Sub btn_network_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btn_network.Click
        If states <> 4 Then
            states = 4
            Me.btn_status.Image = My.Resources.btn_status_n
            Me.btn_status.Location = New Point(138, 31)
            Me.btn_analysis.Image = My.Resources.btn_analysis_n
            Me.btn_analysis.Location = New Point(234, 31)
            Me.btn_history.Image = My.Resources.btn_history_n
            Me.btn_history.Location = New Point(330, 31)
            Me.btn_network.Image = My.Resources.btn_network_d
            Me.btn_network.Location = New Point(415, 33)
            Me.btn_preference.Image = My.Resources.btn_preference_n
            Me.btn_preference.Location = New Point(505, 31)

            'Me.pnPemantau.Controls.Clear()
            'Me.pnPemantau.Controls.Add(New frm_network)
        End If
    End Sub

    Private Sub btn_preference_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btn_preference.Click
        frm_preferences.Show()
        'If states <> 5 Then
        '    states = 5
        '    Me.btn_status.Image = My.Resources.btn_status_n
        '    Me.btn_status.Location = New Point(138, 31)
        '    Me.btn_analysis.Image = My.Resources.btn_analysis_n
        '    Me.btn_analysis.Location = New Point(234, 31)
        '    Me.btn_history.Image = My.Resources.btn_history_n
        '    Me.btn_history.Location = New Point(330, 31)
        '    Me.btn_network.Image = My.Resources.btn_network_n
        '    Me.btn_network.Location = New Point(415, 31)
        '    Me.btn_preference.Image = My.Resources.btn_preference_d
        '    Me.btn_preference.Location = New Point(505, 33)

        'End If
    End Sub

    Private Sub frmMain_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        'When our form loads, auto detect all serial ports in the system and populate
the cmbPort Combo box.
        myPort = IO.Ports.SerialPort.GetPortNames() 'Get all com ports available
    End Sub

    Private Sub serial_connect()
        SerialPort1.PortName = "COM68"        'Set SerialPort1 to the selected COM
port at startup
        SerialPort1.BaudRate = 9600         'Set Baud rate to the selected value on
```

```
        SerialPort1.Parity = IO.Ports.Parity.None
        SerialPort1.StopBits = IO.Ports.StopBits.One
        SerialPort1.DataBits = 8            'Open our serial port
        SerialPort1.Open()

    End Sub

    Private Sub serial_disconnect()
        SerialPort1.Close()             'Close our Serial Port
    End Sub

    Private Sub serial_send()
        'SerialPort1.Write(txtTransmit.Text & vbCr)
    End Sub

    Private Sub SerialPort1_DataReceived(ByVal sender As Object, ByVal e As
System.IO.Ports.SerialDataReceivedEventArgs) Handles SerialPort1.DataReceived
        ReceivedText(SerialPort1.ReadExisting())    'Automatically called every time a
data is received at the serialPort
    End Sub
    Private Sub ReceivedText(ByVal [text] As String)

        Dim txTest As String
        txTest = [text]

        Dim brokenLayer() As String
        Dim vals As String

        brokenLayer = Split(txTest, "||")

        For Each vals In brokenLayer
            If vals.StartsWith("P1:") And vals.EndsWith("P") Then
                power_real = vals.Replace("P1:", "").Replace("P", "").Replace("|", "")
                power(0).Rows.Add(Date.Now, power_real)
            End If
            If vals.StartsWith("E1:") And vals.EndsWith("E") Then
                env_light = vals.Replace("E1:", "").Replace("E", "").Replace("|", "")
                enviroment(0).Rows.Add(Date.Now, env_light)
            End If
            If vals.StartsWith("E2:") And vals.EndsWith("E") Then
                env_temp = vals.Replace("E2:", "").Replace("E", "").Replace("|", "")
                enviroment(1).Rows.Add(Date.Now, env_temp)
            End If
            If vals.StartsWith("E3:") And vals.EndsWith("E") Then
                env_humid = vals.Replace("E3:", "").Replace("E", "").Replace("|", "")
                enviroment(2).Rows.Add(Date.Now, env_humid)
            End If
            If vals.StartsWith("E4:") And vals.EndsWith("E") Then
                env_wind = vals.Replace("E4:", "").Replace("E", "").Replace("|", "")
                enviroment(3).Rows.Add(Date.Now, env_wind)
            End If
            If vals.StartsWith("V1:") And vals.EndsWith("V") Then
                voltage_PV(0) = vals.Replace("V1:", "").Replace("V", "").Replace("|",
"")

                voltage(0).Rows.Add(Date.Now, voltage_PV(0))
            End If
            If vals.StartsWith("V2:") And vals.EndsWith("V") Then
                voltage_PV(1) = vals.Replace("V2:", "").Replace("V", "").Replace("|",
"")

                voltage(1).Rows.Add(Date.Now, voltage_PV(1))
            End If
```

```
            If vals.StartsWith("V3:") And vals.EndsWith("V") Then
                voltage_PV(2) = vals.Replace("V3:", "").Replace("V", "").Replace("|",
"")
                voltage(2).Rows.Add(Date.Now, voltage_PV(2))
            End If
            If vals.StartsWith("V4:") And vals.EndsWith("V") Then
                voltage_PV(3) = vals.Replace("V4:", "").Replace("V", "").Replace("|",
"")
                voltage(3).Rows.Add(Date.Now, voltage_PV(3))
            End If
            If vals.StartsWith("V5:") And vals.EndsWith("V") Then
                voltage_PV(4) = vals.Replace("V5:", "").Replace("V", "").Replace("|",
"")
                voltage(4).Rows.Add(Date.Now, voltage_PV(4))
            End If
            If vals.StartsWith("V6:") And vals.EndsWith("V") Then
                voltage_PV(5) = vals.Replace("V6:", "").Replace("V", "").Replace("|",
"")
                voltage(5).Rows.Add(Date.Now, voltage_PV(5))
            End If
            If vals.StartsWith("V7:") And vals.EndsWith("V") Then
                voltage_PV(6) = vals.Replace("V7:", "").Replace("V", "").Replace("|",
"")
                voltage(6).Rows.Add(Date.Now, voltage_PV(6))
            End If
            If vals.StartsWith("C1:") And vals.EndsWith("C") Then
                inverter(0) = vals.Replace("C1:", "").Replace("C", "").Replace("|",
"")
                inv(0).Rows.Add(Date.Now, inverter(0))
            End If
            If vals.StartsWith("C2:") And vals.EndsWith("C") Then
                max_battery(0) = vals.Replace("C2:", "").Replace("C", "").Replace("|",
"")
                battery(0).Rows.Add(Date.Now, max_battery(0))
            End If
            If vals.StartsWith("C3:") And vals.EndsWith("C") Then
                max_battery(1) = vals.Replace("C3:", "").Replace("C", "").Replace("|",
"")
                battery(1).Rows.Add(Date.Now, max_battery(1))
            End If
            If vals.StartsWith("V8:") And vals.EndsWith("V") Then
                charge_controller(0) = vals.Replace("V8:", "").Replace("V",
"").Replace("|", "")
                chrgecn(0).Rows.Add(Date.Now, charge_controller(0))
            End If
        Next
        'End If

    End Sub

    Private Sub timerRealTimeData_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles timerRealTimeData.Tick
        chart_real_power()
        enviroment_condition()
        conversion_module()
        pv_arrays()
        sun_new_data()
        'sun_elevation()
        ' max_battery(0)
        'Dim yValues() As Double = New Double() {battery(0).Columns(1).Container}
        'cn_battery.Series(0).Points.AddY(max_battery(0))
        'cn_battery.Series(0).YValuesPerPoint(0) = max_battery(0)
```

```
        'cn_battery.DataSource = battery(0)
        'cn_battery.Series(0).XValueMember = "Time"
        'cn_battery.Series(0).YValueMembers = "Value"
        'conversion_module()
    End Sub

    Private Sub chart_real_power()
        Dim numberOfPointsAddedMin As Integer = 5
        Dim numberOfPointsAddedMax As Integer = 10
        Dim pointNumber As Integer

        numberOfPointsInChart = 60 '23
        numberOfPointsAfterRemoval = numberOfPointsInChart - 1

        For pointNumber = 0 To (random.Next(numberOfPointsAddedMin,
numberOfPointsAddedMax)) - 1
            pointIndex = pointIndex + 1
            chrt_power_kw.Series(0).Points.AddXY(pointIndex, power_real)
'random.Next(2000, 4000))
        Next pointNumber

        chrt_power_kw.ResetAutoValues()

        While chrt_power_kw.Series(0).Points.Count > numberOfPointsInChart
            While chrt_power_kw.Series(0).Points.Count > numberOfPointsAfterRemoval
                chrt_power_kw.Series(0).Points.RemoveAt(0)
            End While

            chrt_power_kw.ChartAreas(0).AxisX.Minimum = pointIndex -
numberOfPointsAfterRemoval
            chrt_power_kw.ChartAreas(0).AxisX.Maximum =
chrt_power_kw.ChartAreas(0).AxisX.Minimum + numberOfPointsInChart
        End While

        chrt_power_kw.ChartAreas(0).AxisX.LabelStyle.ForeColor = Color.White

        chrt_power_kw.Series(0).BorderWidth = 3

        chrt_power_kw.Series(0).MarkerStyle = MarkerStyle.Circle
        chrt_power_kw.Series(0).MarkerSize = 5
        chrt_power_kw.Series(0).MarkerColor = Color.White
        chrt_power_kw.Series(0).MarkerBorderColor = Color.CornflowerBlue
        chrt_power_kw.Series(0).MarkerBorderWidth = 2
        chrt_power_kw.Series(0).ChartType = SeriesChartType.SplineArea
        chrt_power_kw.Invalidate()
    End Sub

    Private Sub enviroment_condition()
        lbl_temp_cn.Text = Str(env_temp) + "'C"
        lbl_humid_cn.Text = Str(env_humid) + "%"
        lbl_wind_cn.Text = Str(env_wind) + " Mph"
    End Sub

    Private Sub conversion_module()
        cn_battery.Series(0).Points(0).YValues = {max_battery(1) * 2} ' max_battery(0)
        cn_battery.Series(0).Points(1).YValues = {100}

        cn_charge_controller.Series(0).Points(0).YValues = {charge_controller(0)}
        cn_charge_controller.Series(0).Points(1).YValues = {5}

        cn_inverter.Series(0).Points(0).YValues = {inverter(0)}
        cn_inverter.Series(0).Points(1).YValues = {5}
```

```vb
    End Sub

    Private Sub pv_arrays()
        cn_pv_array.Series(0).Points(0).YValues = {voltage_PV(0)}
        cn_pv_array.Series(0).Points(0).YValues = {voltage_PV(1)}
        cn_pv_array.Series(0).Points(2).YValues = {voltage_PV(2)}
        cn_pv_array.Series(0).Points(3).YValues = {voltage_PV(3)}
        cn_pv_array.Series(0).Points(4).YValues = {voltage_PV(4)}
        cn_pv_array.Series(0).Points(5).YValues = {voltage_PV(5)}
        cn_pv_array.Series(0).Points(6).YValues = {voltage_PV(6)}
    End Sub

    Private Sub sun_elevation()
        cn_sun_elevation.Series(0).Points.AddY(8.1)
        cn_sun_elevation.Series(0).Points.AddY(7.6)
        cn_sun_elevation.Series(0).Points.AddY(9.5)
        cn_sun_elevation.Series(0).Points.AddY(8.5)
        cn_sun_elevation.Series(0).Points.AddY(9.0)
        cn_sun_elevation.Series(0).Points.AddY(8.0)

        cn_sun_elevation.Series(1).Points.AddY(2.3)
        cn_sun_elevation.Series(1).Points.AddY(4.2)
        cn_sun_elevation.Series(1).Points.AddY(3.6)
        cn_sun_elevation.Series(1).Points.AddY(2.3)
        cn_sun_elevation.Series(1).Points.AddY(1.6)
        cn_sun_elevation.Series(1).Points.AddY(2.9)
    End Sub

    Private Sub sun_new_data()
        cn_sun_elevation.Series(0).Points(0).YValues = {random.Next(0, 8.1)}
        cn_sun_elevation.Series(0).Points(1).YValues = {random.Next(4, 7.7)}
        cn_sun_elevation.Series(0).Points(2).YValues = {random.Next(7, 9.5)}
        cn_sun_elevation.Series(0).Points(3).YValues = {random.Next(2, 5)}
        cn_sun_elevation.Series(0).Points(4).YValues = {random.Next(5, 9)}
        cn_sun_elevation.Series(0).Points(5).YValues = {random.Next(6, 8.1)}

        cn_sun_elevation.Series(1).Points(0).YValues = {random.Next(0, 8.1)}
        cn_sun_elevation.Series(1).Points(1).YValues = {random.Next(0, 8.1)}
        cn_sun_elevation.Series(1).Points(2).YValues = {random.Next(0, 8.1)}
        cn_sun_elevation.Series(1).Points(3).YValues = {random.Next(0, 8.1)}
        cn_sun_elevation.Series(1).Points(4).YValues = {random.Next(0, 8.1)}
        cn_sun_elevation.Series(1).Points(5).YValues = {random.Next(0, 8.1)}
    End Sub


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        lst_system_log.DataSource = power(0).DataSet
        DataGridView1.DataSource = chrgecn(0)
        conversion_module()
    End Sub


    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        'lst_system_log.DataSource = power(0).DataSet
        DataGridView1.DataSource = power(0)
        'conversion_module()
    End Sub
End Class
```

>frm_output.vb

```
Imports System.Windows.Forms.DataVisualization.Charting

Public Class frm_output
    Private random As New Random()
    Private pointIndex As Integer = 0
    Dim numberOfPointsInChart As Integer = 200 '200
    Dim numberOfPointsAfterRemoval As Integer = 75 '150

    Private Sub timerRealTimeData_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles timerRealTimeData.Tick
        chart_real_power()
        enviroment_condition()
    End Sub

    Private Sub chart_real_power()
        Dim numberOfPointsAddedMin As Integer = 5
        Dim numberOfPointsAddedMax As Integer = 10
        Dim pointNumber As Integer

        numberOfPointsInChart = 60 '23
        numberOfPointsAfterRemoval = numberOfPointsInChart - 1

        For pointNumber = 0 To (random.Next(numberOfPointsAddedMin,
numberOfPointsAddedMax)) - 1
            pointIndex = pointIndex + 1
            Chart1.Series(0).Points.AddXY(pointIndex, power_real) 'random.Next(2000,
4000))
        Next pointNumber

        Chart1.ResetAutoValues()

        While Chart1.Series(0).Points.Count > numberOfPointsInChart
            While Chart1.Series(0).Points.Count > numberOfPointsAfterRemoval
                Chart1.Series(0).Points.RemoveAt(0)
            End While

            Chart1.ChartAreas(0).AxisX.Minimum = pointIndex -
numberOfPointsAfterRemoval
            Chart1.ChartAreas(0).AxisX.Maximum = Chart1.ChartAreas(0).AxisX.Minimum +
numberOfPointsInChart
        End While

        Chart1.ChartAreas(0).AxisX.LabelStyle.ForeColor = Color.White

        Chart1.Series(0).BorderWidth = 3

        Chart1.Series(0).MarkerStyle = MarkerStyle.Circle
        Chart1.Series(0).MarkerSize = 5
        Chart1.Series(0).MarkerColor = Color.White
        Chart1.Series(0).MarkerBorderColor = Color.CornflowerBlue
        Chart1.Series(0).MarkerBorderWidth = 2
        Chart1.Series(0).ChartType = SeriesChartType.SplineArea
        Chart1.Invalidate()
    End Sub

    Private Sub enviroment_condition()
        lbl_temp_cn.Text = Str(env_temp) + "'C"
        lbl_humid_cn.Text = Str(env_humid) + "%"
        lbl_wind_cn.Text = Str(env_wind) + " Mph"
    End Sub
```

```
    Private Sub frm_output_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        pointIndex = 0
    End Sub
End Class
```

>frm_preferences.vb

```
Imports _1PemantauSys
Public Class frm_preferences
    Private comm As New CommManager()
    Private transType As String = String.Empty

    Private Sub cboPort_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles cbPorts.SelectedIndexChanged
        comm.PortName = cbPorts.Text()
    End Sub

    Private Sub SetDefaults()
        cbPorts.SelectedIndex = 0
        cbBaud.SelectedText = "9600"
        cbParity.SelectedIndex = 0
        cbStop.SelectedIndex = 1
        cbData.SelectedIndex = 1
    End Sub

    Private Sub LoadValues()
        comm.SetPortNameValues(cbPorts)
        comm.SetParityValues(cbParity)
        comm.SetStopBitValues(cbStop)
    End Sub

    Private Sub SetControlState()
        rdText.Checked = True
        btn_close_ports.Enabled = False
    End Sub

    Private Sub frmMain_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        LoadValues()
        SetDefaults()
        SetControlState()
    End Sub

    Private Sub cmdClose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btn_close_ports.Click
        comm.ClosePort()
        btn_open_ports.Enabled = True
        SetControlState()
        SetDefaults()
    End Sub

    Private Sub cmdOpen_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btn_open_ports.Click
        comm.Parity = cbParity.Text
        comm.StopBits = cbStop.Text
        comm.DataBits = cbData.Text
        comm.BaudRate = cbBaud.Text
        comm.DisplayWindow = RichTextBox1
```

```
        comm.OpenPort()

        btn_open_ports.Enabled = False
        btn_close_ports.Enabled = True
    End Sub


    Private Sub rdoHex_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles rdHex.CheckedChanged
        If rdHex.Checked() Then
            comm.CurrentTransmissionType =
_1PemantauSys.CommManager.TransmissionType.Hex
        Else
            comm.CurrentTransmissionType =
_1PemantauSys.CommManager.TransmissionType.Text
        End If
    End Sub

    Private Sub cboBaud_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles cbBaud.SelectedIndexChanged
        comm.BaudRate = cbBaud.Text()
    End Sub

    Private Sub cboParity_SelectedIndexChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles cbParity.SelectedIndexChanged
        comm.Parity = cbParity.Text()
    End Sub

    Private Sub cboStop_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles cbStop.SelectedIndexChanged
        comm.StopBits = cbStop.Text()
    End Sub

    Private Sub cboData_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles cbData.SelectedIndexChanged
        comm.StopBits = cbStop.Text()
    End Sub

End Class
```

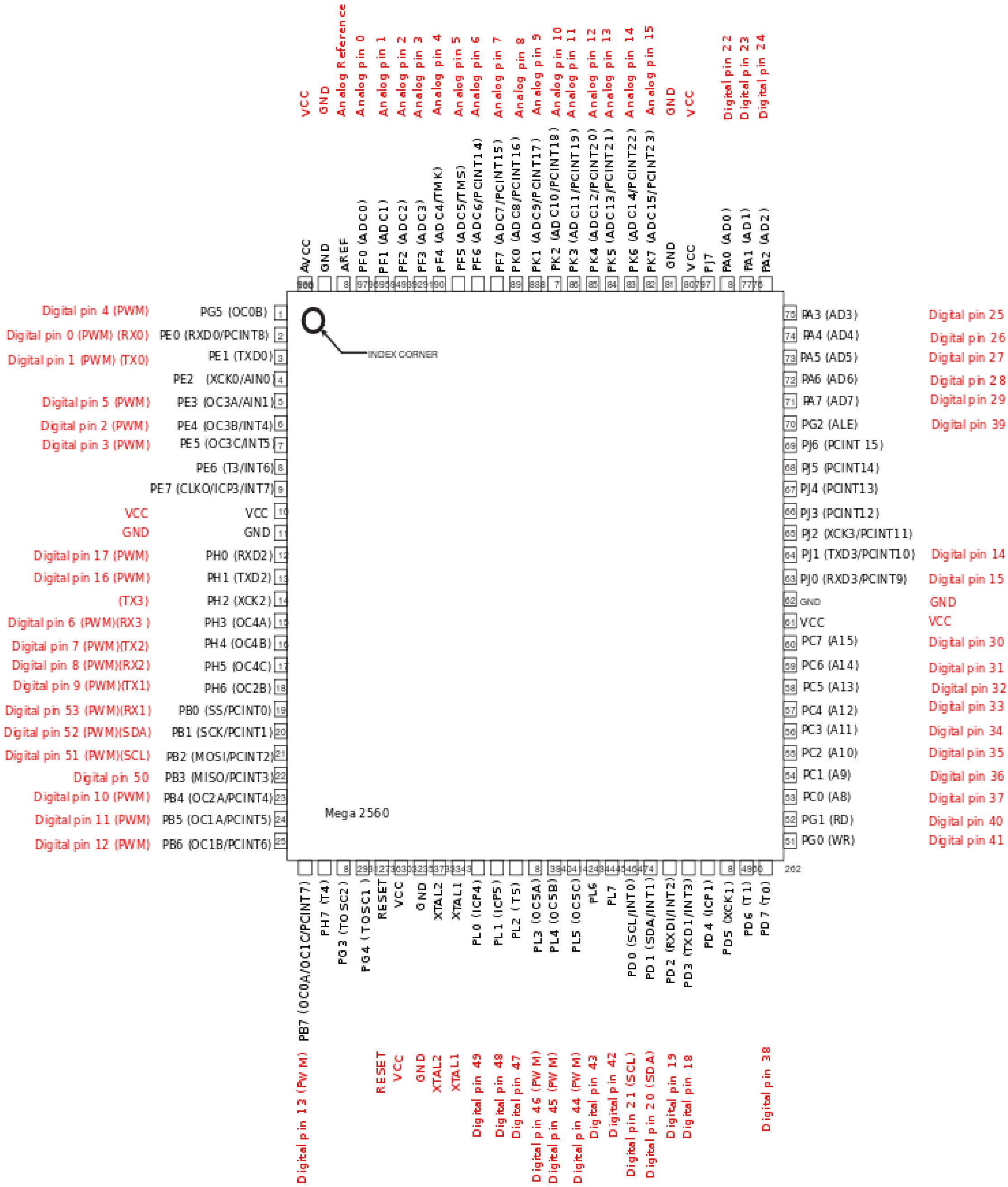
>Sensors.vb

```
Module Sensors
    Public enviroment(10) As DataTable
    Public voltage(8) As DataTable
    Public current(5) As DataTable
    Public power(3) As DataTable
    Public battery(1) As DataTable
    Public inv(3) As DataTable
    Public chrgecn(3) As DataTable
    Public power_real As Integer
    Public env_light As Integer
    Public env_temp As Integer
    Public env_humid As Integer
    Public env_wind As Integer
    Public voltage_PV(8) As Integer
    Public max_battery(1) As Integer
    Public inverter(3) As Integer
    Public charge_controller(3) As Integer
End Module
```

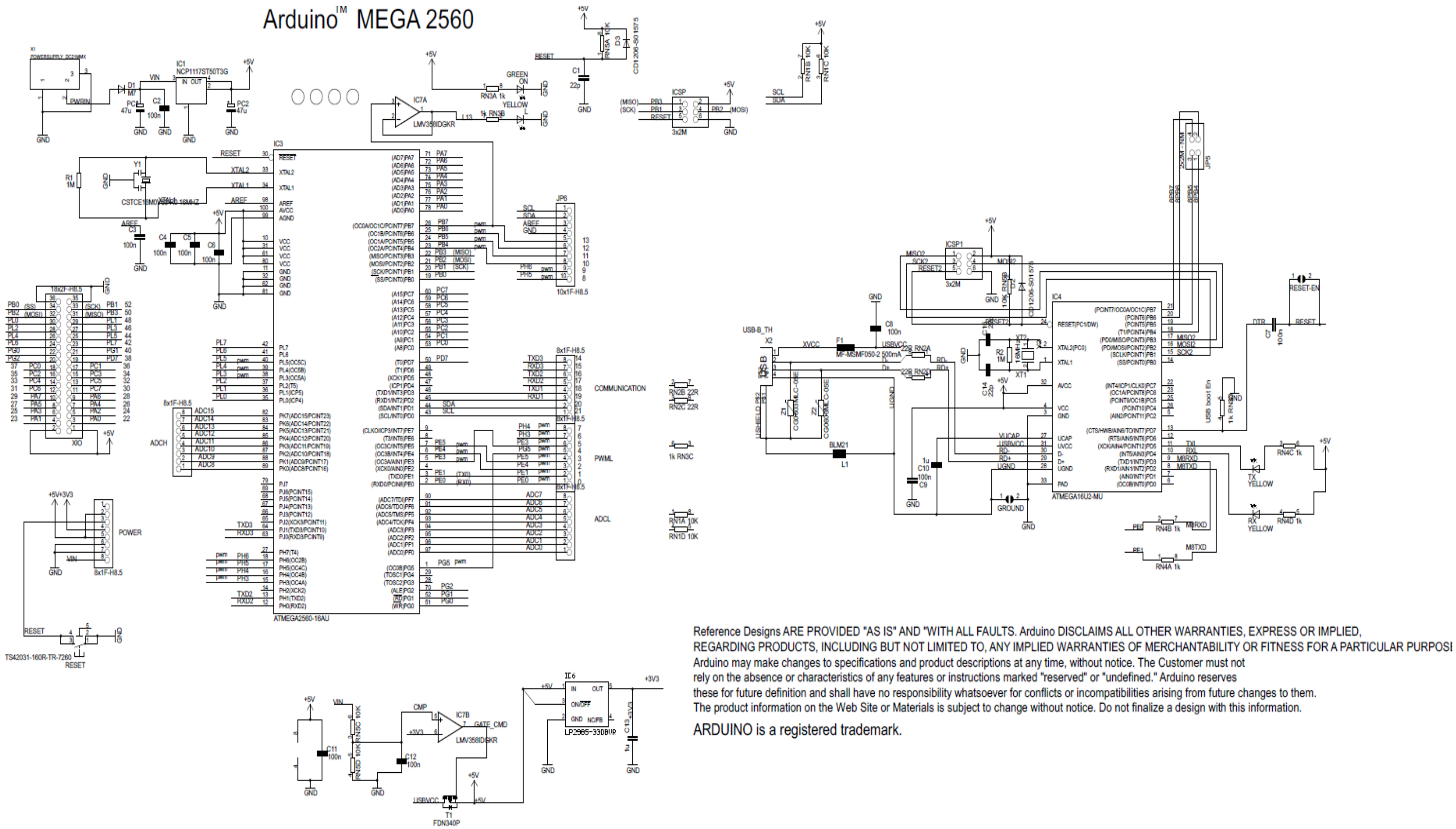
>Constant.vb

```vb
Module Constants
    Public portsCn As String
    Public baudCn As Integer
    Public parityCn As String
    Public stopCn As Integer
    Public dataCn As Integer
End Module
```

7.5 APPENDIX 5 - ATMEGA 2560 PIN MAPPING



7.6 APPENDIX 6 - ATMEGA 2560



Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE
Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Arduino reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.
The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

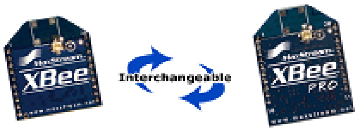
ARDUINO is a registered trademark.

7.7 APPENDIX 7 - XBEE DATA SHEET

1. XBee®/XBee-PRO® RF Modules

The XBee and XBee-PRO RF Modules were engineered to meet IEEE 802.15.4 standards and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of data between devices.

The modules operate within the ISM 2.4 GHz frequency band and are pin-for-pin compatible with each other.



Key Features

Long Range Data Integrity

XBee

- Indoor/Urban: up to 100' (30 m)
- Outdoor line-of-sight: up to 300' (90 m)
- Transmit Power: 1 mW (0 dBm)
- Receiver Sensitivity: -92 dBm

XBee-PRO

- Indoor/Urban: up to 300' (90 m), 200' (60 m) for International variant
- Outdoor line-of-sight: up to 1 mile (1600 m), 2500' (750 m) for International variant
- Transmit Power: 63mW (18dBm), 10mW (10dBm) for International variant
- Receiver Sensitivity: -100 dBm

RF Data Rate: 250,000 bps

Advanced Networking & Security

Retries and Acknowledgements

DSSS (Direct Sequence Spread Spectrum)

Each direct sequence channels has over 65,000 unique network addresses available

Source/Destination Addressing

Unicast & Broadcast Communications

Point-to-point, point-to-multipoint and peer-to-peer topologies supported

Low Power

XBee

- TX Peak Current: 45 mA (@3.3 V)
- RX Current: 50 mA (@3.3 V)
- Power-down Current: < 10 µA

XBee-PRO

- TX Peak Current: 250mA (150mA for international variant)
- TX Peak Current (RPSMA module only): 340mA (180mA for international variant
- RX Current: 55 mA (@3.3 V)
- Power-down Current: < 10 µA

ADC and I/O line support

Analog-to-digital conversion, Digital I/O

I/O Line Passing

Easy-to-Use

No configuration necessary for out-of box RF communications

Free X-CTU Software
(Testing and configuration software)

AT and API Command Modes for configuring module parameters

Extensive command set

Small form factor

Worldwide Acceptance

FCC Approval (USA) Refer to Appendix A [p64] for FCC Requirements.
Systems that contain XBee®/XBee-PRO® RF Modules inherit Digi Certifications.

ISM (Industrial, Scientific & Medical) **2.4 GHz frequency band**

Manufactured under **ISO 9001:2000** registered standards

XBee®/XBee-PRO® RF Modules are optimized for use in the United States, Canada, Australia, Japan, and Europe. Contact Digi for complete list of government agency approvals.

Specifications

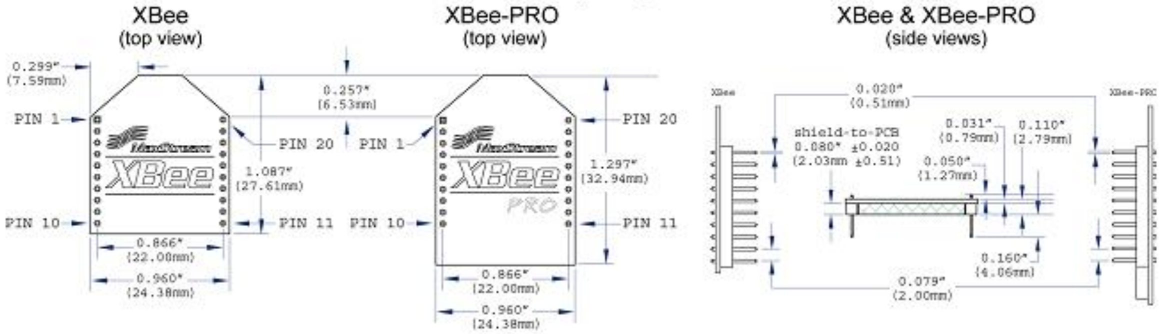
Table 1-01. Specifications of the XBee®/XBee-PRO® RF Modules

Specification	XBee	XBee-PRO
Performance		
Indoor/Urban Range	Up to 100 ft (30 m)	Up to 300 ft. (90 m), up to 200 ft (60 m) International variant
Outdoor RF line-of-sight Range	Up to 300 ft (90 m)	Up to 1 mile (1600 m), up to 2500 ft (750 m) international variant
Transmit Power Output (software selectable)	1mW (0 dBm)	63mW (18dBm)* 10mW (10 dBm) for International variant
RF Data Rate	250,000 bps	250,000 bps
Serial Interface Data Rate (software selectable)	1200 bps - 250 kbps (non-standard baud rates also supported)	1200 bps - 250 kbps (non-standard baud rates also supported)
Receiver Sensitivity	-92 dBm (1% packet error rate)	-100 dBm (1% packet error rate)
Power Requirements		
Supply Voltage	2.8 – 3.4 V	2.8 – 3.4 V
Transmit Current (typical)	45mA (@ 3.3 V)	250mA (@3.3 V) (150mA for international variant) RPSMA module only: 340mA (@3.3 V) (180mA for international variant)
Idle / Receive Current (typical)	50mA (@ 3.3 V)	55mA (@ 3.3 V)
Power-down Current	< 10 µA	< 10 µA
General		
Operating Frequency	ISM 2.4 GHz	ISM 2.4 GHz
Dimensions	0.960" x 1.087" (2.438cm x 2.761cm)	0.960" x 1.297" (2.438cm x 3.294cm)
Operating Temperature	-40 to 85° C (industrial)	-40 to 85° C (industrial)
Antenna Options	Integrated Whip, Chip or U.FL Connector, RPSMA Connector	Integrated Whip, Chip or U.FL Connector, RPSMA Connector
Networking & Security		
Supported Network Topologies	Point-to-point, Point-to-multipoint & Peer-to-peer	
Number of Channels (software selectable)	16 Direct Sequence Channels	12 Direct Sequence Channels
Addressing Options	PAN ID, Channel and Addresses	PAN ID, Channel and Addresses
Agency Approvals		
United States (FCC Part 15.247)	OUR-XBEE	OUR-XBEEPRO
Industry Canada (IC)	4214A XBEE	4214A XBEEPRO
Europe (CE)	ETSI	ETSI (Max. 10 dBm transmit power output)*
Japan	R201WW07215214	R201WW08215111 (Max. 10 dBm transmit power output)*
Austraila	C-Tick	C-Tick

* See Appendix A for region-specific certification requirements.

Antenna Options: The ranges specified are typical when using the integrated Whip (1.5 dBi) and Dipole (2.1 dBi) antennas. The Chip antenna option provides advantages in its form factor; however, it typically yields shorter range than the Whip and Dipole antenna options when transmitting outdoors.For more information, refer to the "XBee Antennas" Knowledgebase Article located on Digi's Support Web site

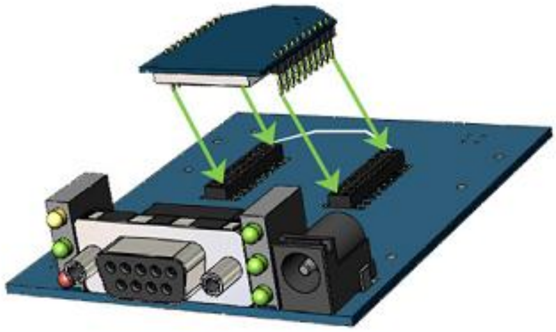
The XBee and XBee-PRO RF Modules are pin-for-pin compatible.



Mounting Considerations

The XBee®/XBee-PRO® RF Module was designed to mount into a receptacle (socket) and therefore does not require any soldering when mounting it to a board. The XBee Development Kits contain RS-232 and USB interface boards which use two 20-pin receptacles to receive modules.

Figure 1-02. XBee Module Mounting to an RS-232 Interface Board.



The receptacles used on Digi development boards are manufactured by Century Interconnect. Several other manufacturers provide comparable mounting solutions; however, Digi currently uses the following receptacles:

- Through-hole single-row receptacles -
  Samtec P/N: MMS-110-01-L-SV (or equivalent)
- Surface-mount double-row receptacles -
  Century Interconnect P/N: CPRMSL20-D-0-1 (or equivalent)
- Surface-mount single-row receptacles -
  Samtec P/N: SMM-110-02-SM-S

Digi also recommends printing an outline of the module on the board to indicate the orientation the module should be mounted.

Pin Signals

Figure 1-03. XBee®/XBee-PRO® RF Module Pin Numbers

(top sides shown - shields on bottom)

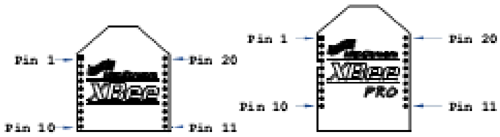


Table 1-02. Pin Assignments for the XBee and XBee-PRO Modules
(Low-asserted signals are distinguished with a horizontal line above signal name.)

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / $\overline{\text{CONFIG}}$	Input	UART Data In
4	DO8*	Output	Digital Output 8
5	$\overline{\text{RESET}}$	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI	Output	PWM Output 0 / RX Signal Strength Indicator
7	PWM1	Output	PWM Output 1
8	[reserved]	-	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ / DI8	Input	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	$\overline{\text{CTS}}$ / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	ON / $\overline{\text{SLEEP}}$	Output	Module Status Indicator
14	VREF	Input	Voltage Reference for A/D Inputs
15	Associate / AD5 / DIO5	Either	Associated Indicator, Analog Input 5 or Digital I/O 5
16	$\overline{\text{RTS}}$ / AD6 / DIO6	Either	Request-to-Send Flow Control, Analog Input 6 or Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

* Function is not supported at the time of this release

Design Notes:

- Minimum connections: VCC, GND, DOUT & DIN
- Minimum connections for updating firmware: VCC, GND, DIN, DOUT, RTS & DTR
- Signal Direction is specified with respect to the module
- Module includes a 50k Ω pull-up resistor attached to $\overline{\text{RESET}}$
- Several of the input pull-ups can be configured using the PR command
- Unused pins should be left disconnected

7.8 APPENDIX 8 - ITDB02 LCD WITH TOUCH MODULE

ITDB02_Graph16 - Arduino library support for ITDB02 LCD Board

Basic functionality of this library are based on the demo-code provided by ITead studio. You can find the latest version of the library at http://www.henningkarlsen.com/electronics

This library has been made especially for the 3.2" TFT LCD Screen Module: ITDB02-3.2 by ITead studio. This library has been designed to use 16bit mode, and it should work with the 2.4" Module in 16bit mode as well, although I do not have one, so this is untested.

If you make any modifications or improvements to the code, I would appreciate that you share the code with me so that I might include it in the next release. I can be contacted through http://www.henningkarlsen.com/electronics/contact.php

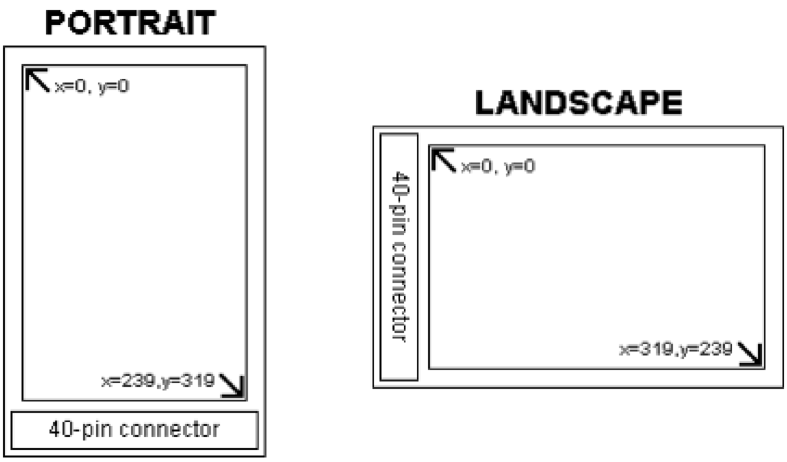
Version:	2.0	Aug 15 2010	• initial release
	2.1	Sep 30 2010	• Added Arduino Mega compatibility • Fixed a bug with CENTER and RIGHT in LANDSCAPE mode • Fixed a bug in printNumI and printNumF when the number to be printed was 0
	2.2	Oct 14 2010	• Added support for ITDB02-3.2WC • Added drawBitmap() with its associated tool
	2.3	Nov 24 2010	• Added Arduino Mega2560 compatibility • Added support for rotating text and bitmaps.
	2.4	Jan 18 2011	• Fixed an error in the requirements
	2.5	Jan 30 2011	• Added loadBitmap() • Optimized drawBitmap() when not using rotation
	2.6	Mar 4 2011	• Fixed a bug in printNumF when the number to be printed was (-)0.something
	3.0	Mar 19 2011	• General optimization
	3.01	Mar 20 2011	• Reduced memory footprint slightly
	4.0	Mar 27 2011	• Remade the font-system to make it more flexible
	4.1	Apr 19 2011	• Remade the tinyFAT integration. Moved loadBitmap() to the ITDB02_tinyFAT library

(*) Initial release is v2.0 to keep it in sync with the 8bit library.

IMPORTANT:
If you are upgrading from a version below v4.0 you have to delete the old library before unpacking v4.0+

INTEGRATION WITH tinyFAT:
tinyFAT integration has been moved to a separate library. Please use the ITDB02_tinyFAT16 library to enable integration.

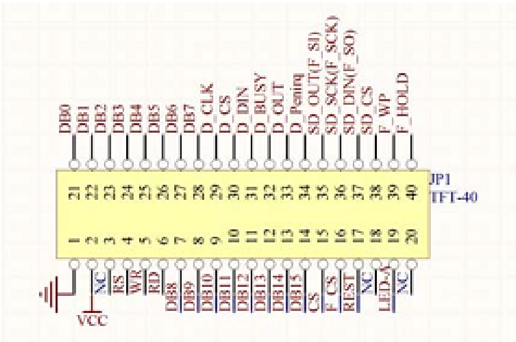
DISPLAY ORIENTATION:



Requirements:

The library require the following connections:

Signal	ITDB02 pin	Arduino pin*	Arduino Mega pin
DB0	21	D8	D37
DB1	22	D9	D36
DB2	23	D10	D35
DB3	24	D11	D34
DB4	25	D12	D33
DB5	26	D13	D32
DB6	27	A0 (D14)	D31
DB7	28	A1 (D15)	D30
DB8	7	D0	D22
DB9	8	D1	D23
DB10	9	D2	D24
DB11	10	D3	D25
DB12	11	D4	D26
DB13	12	D5	D27
DB14	13	D6	D28
DB15	14	D7	D29



ITDB02 pinout

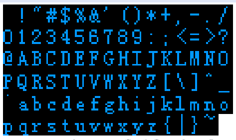
Defined Literals:

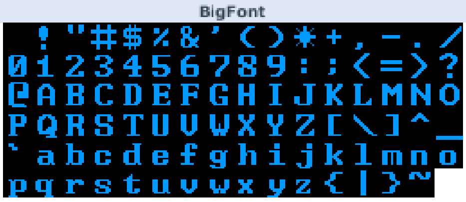
Alignment
For use with print(), printNumI() and printNumF()
LEFT: 0 RIGHT: 9999 CENTER: 9998

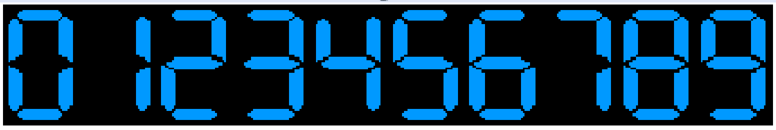
Orientation
For use with InitLCD()
PORTRAIT: 0 LANDSCAPE: 1

Aspect Ratio
For use with ITDB02()
ASPECT_4x3: 0 ASPECT_16x9: 1

Included Fonts:

SmallFont

Character size: 8x12 pixels Number of characters: 95

BigFont

Character size: 16x16 pixels Number of characters: 95

SevenSegNumFont

Character size: 32x50 pixels Number of characters: 10

Functions:

ITDB02(RS, WR, CS, RST[, Aspect]);	
The main class of the interface.	
Parameters:	RS: Arduino pin for Register Select WR: Arduino pin for Write CS: Arduino pin for Chip Select RST: Arduino pin for Reset Aspect: **<optional>** ASPECT_4x3 for ITDB02-2.4 and ITDB02-3.2 (both 240x320 pixels) **(default)** ASPECT_16x9 for ITDB02-3.2WC (240x400 pixels)
Usage:	`ITDB02 myGLCD(19,18,17,16); // Start an instance of the ITDB02 class`

InitLCD([orientation]);	
Initialize the LCD and set display orientation.	
Parameters:	Orientation: **<optional>** PORTRAIT **(default)** LANDSCAPE
Usage:	`myGLCD.initLCD(); // Initialize the display`
Notes:	This will reset color to white with black background. Font size will be reset to FONT_SMALL.

clrScr();	
Clear the screen. The background-color will be set to black.	
Parameters:	None
Usage:	`myGLCD.clrScr(); // Clear the screen`

fillScr(r, g, b);	
Fill the screen with a specified color.	
Parameters:	r: Red component of an RGB value (0-255) g: Green component of an RGB value (0-255) b: Blue component of an RGB value (0-255)
Usage:	`myGLCD.fillScr(255,127,0); // Fill the screen with orange`

setColor(r, g, b);	
Set the color to use for all draw*, fill* and print commands.	
Parameters:	r: Red component of an RGB value (0-255) g: Green component of an RGB value (0-255) b: Blue component of an RGB value (0-255)
Usage:	`myGLCD.setColor(0,255,255); // Set the color to cyan`

setBackColor(r, g, b);	
Set the background color to use for all print commands.	
Parameters:	r: Red component of an RGB value (0-255) g: Green component of an RGB value (0-255) b: Blue component of an RGB value (0-255)
Usage:	`myGLCD.setBackColor(255,255,255); // Set the background color to white`

drawPixel(x, y);	
Draw a single pixel.	
Parameters:	x: x-coordinate of the pixel (0-239) y: y-coordinate of the pixel (0-319)
Usage:	`myGLCD.drawPixel(119,159); // Draw a single pixel at the center of the screen`

drawLine(x1, y1, x2, y2);	
Draw a line between two points.	
Parameters:	x1: x-coordinate of the start-point (0-239) y1: y-coordinate of the start-point (0-319) x2: x-coordinate of the end-point (0-239) y2: y-coordinate of the end-point (0-319)
Usage:	`myGLCD.drawLine(0,0,239,319); // Draw a line from the upper left to the lower right corner`

drawRect(x1, y1, x2, y2);	
Draw a rectangle between two points.	

Parameters:	`x1: x-coordinate of the start-corner (0-239)`
	`y1: y-coordinate of the start-corner (0-319)`
	`x2: x-coordinate of the end-corner (0-239)`
	`y2: y-coordinate of the end-corner (0-319)`

Usage:	`myGLCD.drawRect(119,159,239,319); // Draw a rectangle in the lower right corner of the screen`

drawRoundRect(x1, y1, x2, y2);	
Draw a rectangle with slightly rounded corners between two points. The minimum size is 5 pixels in both directions. If a smaller size is requested the rectangle will not be drawn.	

Parameters:	`x1: x-coordinate of the start-corner (0-239)`
	`y1: y-coordinate of the start-corner (0-319)`
	`x2: x-coordinate of the end-corner (0-239)`
	`y2: y-coordinate of the end-corner (0-319)`

Usage:	`myGLCD.drawRoundRect(0,0,119,159); // Draw a rounded rectangle in the upper left corner of the screen`

fillRect(x1, y1, x2, y2);	
Draw a filled rectangle between two points.	

Parameters:	`x1: x-coordinate of the start-corner (0-239)`
	`y1: y-coordinate of the start-corner (0-319)`
	`x2: x-coordinate of the end-corner (0-239)`
	`y2: y-coordinate of the end-corner (0-319)`

Usage:	`myGLCD.fillRect(119,0,239,159); // Draw a filled rectangle in the upper right corner of the screen`

fillRoundRect(x1, y1, x2, y2);	
Draw a filled rectangle with slightly rounded corners between two points. The minimum size is 5 pixels in both directions. If a smaller size is requested the rectangle will not be drawn.	

Parameters:	`x1: x-coordinate of the start-corner (0-239)`
	`y1: y-coordinate of the start-corner (0-319)`
	`x2: x-coordinate of the end-corner (0-239)`
	`y2: y-coordinate of the end-corner (0-319)`

Usage:	`myGLCD.fillRoundRect(0,159,119,319); // Draw a filled, rounded rectangle in the lower left corner of the screen`

drawCircle(x, y, radius);	
Draw a circle with a specified radius.	

Parameters:	`x: x-coordinate of the center of the circle (0-239)`
	`y: y-coordinate of the center of the circle (0-319)`
	`radius: radius of the circle in pixels`

Usage:	`myGLCD.drawCircle(119,159,20); // Draw a circle in the middle of the screen with a radius of 20 pixels`

fillCircle(x, y, radius);	
Draw a filled circle with a specified radius.	

Parameters:	`x: x-coordinate of the center of the circle (0-239)`
	`y: y-coordinate of the center of the circle (0-319)`
	`radius: radius of the circle in pixels`

Usage:	`myGLCD.fillCircle(119,159,10); // Draw a filled circle in the middle of the screen with a radius of 10 pixels`

print(st, x, y[, deg]);	
Print a string at the specified coordinates. An optional background color can be specified. Default background is black. You can use the literals LEFT, CENTER and RIGHT as the x-coordinate to align the string on the screen. *Changed in v2.3*	

Parameters:	`st: the string to print`
	`x: x-coordinate of the upper, left corner of the first character (0-239)`
	`y: y-coordinate of the upper, left corner of the first character (0-319)`
	`deg: <optional>`
	`Degrees to rotate text (0-359). Text will be rotated around the upper left corner.`

Usage:	`myGLCD.print("Hello, World!",CENTER,0); // Print "Hello, World!" centered at the top of the screen`
Notes:	`CENTER and RIGHT will not calculate the coordinates correctly when rotating text.`

printNumI(num, x, y);

Print an integer number at the specified coordinates. An optional background color can be specified. Default background is black.
You can use the literals LEFT, CENTER and RIGHT as the x-coordinate to align the string on the screen.

Parameters:
```
num: the value to print (-2,147,483,648 to 2,147,483,647) INTEGERS ONLY
x:   x-coordinate of the upper, left corner of the first digit/sign (0-239)
y:   y-coordinate of the upper, left corner of the first digit/sign (0-319)
```
Usage:
```
myGLCD.print(num,CENTER,0); // Print the value of "num" centered at the top of the screen
```

printNumF(num, dec, x, y);

Print a floating-point number at the specified coordinates. An optional background color can be specified. Default background is black.
You can use the literals LEFT, CENTER and RIGHT as the x-coordinate to align the string on the screen.
WARNING: Floating point numbers are not exact, and may yield strange results when compared. Use at your own discretion.

Parameters:
```
num: the value to print (See note)
dec: digits in the fractional part (1-5) 0 is not supported. Use printNumI() instead.
x:   x-coordinate of the upper, left corner of the first digit/sign (0-239)
y:   y-coordinate of the upper, left corner of the first digit/sign (0-319)
```
Usage:
```
myGLCD.print(num, 3, CENTER,0); // Print the value of "num" with 3 fractional digits top centered
```
Notes: Supported range depends on the number of fractional digits used.

Fractional digits	Approx range
1	+/- 200000000
2	+/- 20000000
3	+/- 2000000
4	+/- 200000
5	+/- 20000

setFont(fontname);

Select font to use with print(), printNumI() and printNumF().

Added in v4.0

Parameters:
```
fontname: Name of the array containing the font you wish to use
```
Usage:
```
myGLCD.setFont(BigFont); // Select the font called BigFont
```
Notes: You must declare the font-array as an external or include it in your sketch.

drawBitmap (x, y, sx, sy, data[, scale]);

Draw a bitmap on the screen.

Added in v2.2

Parameters:
```
x:     x-coordinate of the upper, left corner of the bitmap
y:     y-coordinate of the upper, left corner of the bitmap
sx:    width of the bitmap in pixels
sy:    height of the bitmap in pixels
data:  array containing the bitmap-data
scale: <optional>
       Scaling factor. Each pixel in the bitmap will be drawn as <scale>x<scale> pixels on screen.
```
Usage:
```
myGLCD.drawBitmap(0, 0, 32, 32, bitmap); // Draw a 32x32 pixel bitmap in the upper left corner
```
Notes: You can use the online-tool "*ImageConverter 565*" or "*ImageConverter565.exe*" in the Tools-folder to convert pictures into compatible arrays. The online-tool can be found on my website.
Requires that you #include <avr/pgmspace.h>

drawBitmap (x, y, sx, sy, data, deg, rox, roy);

Draw a bitmap on the screen with rotation.

Added in v2.3

Parameters:
```
x:    x-coordinate of the upper, left corner of the bitmap
y:    y-coordinate of the upper, left corner of the bitmap
sx:   width of the bitmap in pixels
sy:   height of the bitmap in pixels
data: array containing the bitmap-data
deg:  Degrees to rotate bitmap (0-359)
rox:  x-coordinate of the pixel to use as rotational center relative to bitmaps upper left corner
roy:  y-coordinate of the pixel to use as rotational center relative to bitmaps upper left corner
```
Usage:
```
myGLCD.drawBitmap(50, 50, 32, 32, bitmap, 45, 16, 16); // Draw a bitmap rotated 45 degrees around its center
```
Notes: You can use the online-tool "*ImageConverter 565*" or "*ImageConverter565.exe*" in the Tools-folder to convert pictures into compatible arrays. The online-tool can be found on my website.
Requires that you #include <avr/pgmspace.h>