Implementing SCADA System for Industrial Environment Using 'IEEE C37.1' Standards

Student: Mr. Maldar Aman Malikamber
Mentor: Mr. Tamhankar S.G.
Walchand College of Engineering, Sangli
Maharashtra, India
E-Mail: amanmaldar@gmail.com

***Abstract*— In the current era of industrialization, the field of automation has great opportunities for R&D. When we talk about automation of any industry, an engineer has to develop a system such that a person can have a view of various processes going on in the industry, status of various machines in the industry, status of certain parameters (e.g. temperature, humidity, light, etc) in the industry remotely. Along with this, a system should be capable of providing certain control facilities of the industrial processes, parameters or instruments from distant place. The stated objectives can be accomplished by developing SCADA system (IEEE C37.1 Standard). The project involves the development of concerned standards.**

The project paper focuses on developing IEEE C37.1 Standard comprising of a combination of hardware and software to monitor and control various parameters in industry. The scope of the project extends from providing a system status on a single machine (usually PC) to a distant place using IEEE 802.3 LAN protocol and web based access.

The system is also capable of controlling devices in the industry from the remote place using internet with authentication and security.

***Keywords*— IEEE C37.1, SCADA, IEEE 802.3, Automatic Control, Data Acquisition, Intelligent Electronic Device, Substation Integration, Substation Automation, Supervisory Control, PHP, Data Logging**

I. INTRODUCTION

The project best applies to the industrial environment, to monitor & control certain industrial parameters with/without human interference and to have graphical view of the current status of industry. This project is helpful in terms of learning SCADA system as a whole, role of embedded system in automation and server side programming concepts. Thus, project forms the prerequisites to understand industrial automation concepts and work on the same.

II. PROJECT DESIGN IMPLIMENTATION

The design implementation comprises of following designing stages to establish a complete system. The stages consists of designing,

1) Intelligent Electronic Device (IED)
2) Remote Terminal Unit (RTU)
3) "Control Center" (CC) aka. "Master Station"
4) Graphical User Interface (GUI) at the CC
5) Dedicated server for networking and remote access

III. SYSTEM OVERVIEW AND DESCRIPTION

The schematic diagram for the implemented system is given in Image_1: System Overview at the end of report in ANNEXURE section. System Overview gives schematic of all system components along with interconnectivity among them. Section (III) gives details of various components involved in the system.

A. INTELLIGENT ELECTRONIC DEVICES (IED)

In SCADA system the processing power is distributed among various Control Centers, Remote Terminal Units (RTUs) and Intelligent Electronic Devices (IEDs). IEDs have limited processing power associated with it. IED consists of sensors connected to it. Here, one of the IDE is Reservoir Level Sensor.

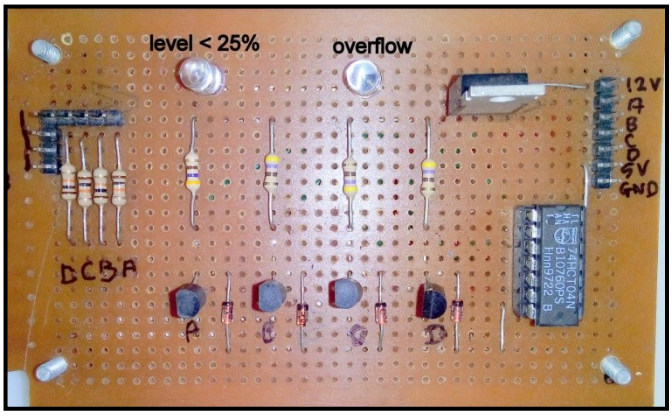


Figure_1: Reservoir Level Sensor

In any industry we need to monitor reservoir level of few quantities like oil, water or simply any coolant. The above circuit is Reservoir Level Sensor. It is used to sense level of water in the tank. It is capable of sensing 5 water levels i.e. EMPTY, 25%, 50%, 75% and FULL.

The heart of the circuit is a sensor strip which is actually inserted in the water tank to detect the water level.

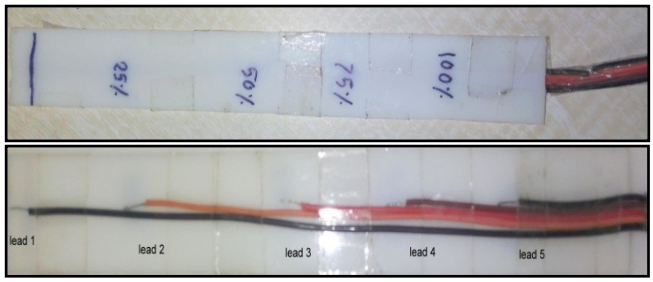


Figure_2: Water Level Sensing Strip (Front & Back view)

This strip comprises of 5 connections (leads) on it. The lowest lead is a power lead i.e. supply is placed at the bottom in the water tank. The remaining 4 leads sense 4 discrete water levels. When user turns on motor manually, water level will increase and touch to lead 2, which in turn will cause a completion of circuit from lead 1 to lead 2. This drives base of one of the transistors (BC547). This causes concerned transistor to turn on and its collector voltage is reduced to logically zero volt, while collector voltages of remaining 3 transistors are logically high as they are still in off condition. The voltages are inverted by an inverter IC to get digitized output. In the same way when water level continues to rise further, it will reach to lead 3, turning on transistor 2; and again reducing collector voltage to 0 volts. Finally the digitized outputs are given to MTU using simple wired connections. The sensor not only senses the water levels, but it also gives an indication (glows LED) when water level reaches 100%. So a user will be intimated and he can turn off the motor. In the same way, when water level is less than 25%, LED is turned on so user can turn on motor. *This demonstrates processing power associated with IED.*

B. SENSORS IN THE DESIGN

1) Reservoir Level Sensor

The description for the sensor is given in previous section. The rest of the sensors are discussed below.

2) Ambient Light Intensity Sensor

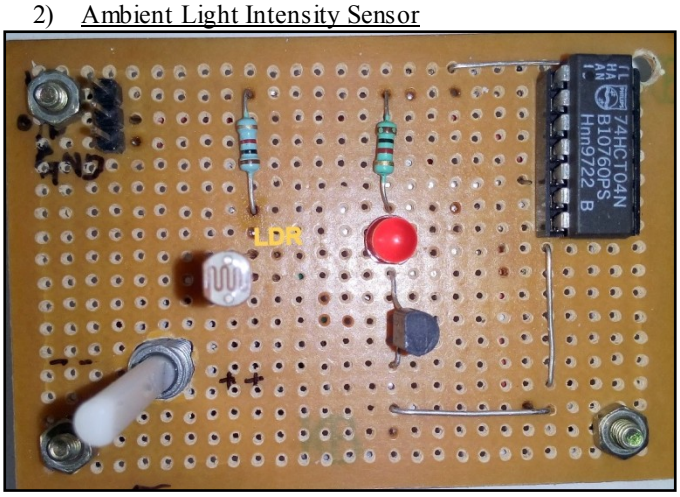


Figure_3: Ambient Light Intensity Sensor

Above circuit comprises of a Light Dependent Resistor (LDR). The resistance of the LDR is inversely proportional to intensity of the ambient light. This property is used to detect if ambient light intensity is lower than the predefined threshold or not. When ambient light intensity goes below predefined threshold, corrective measures are taken RTU to turn on flood lights (here, Orange LED mounted on RTU PCB). *Thus RTU has processing powers associated with it to control flood lights.* Ambient Light Intensity is one of the parameters that we are concerned about. The output of the designed circuit is digitized into two level i.e. 'logic 0' and 'logic 1' using an inverter IC (74LS04). Finally output of this circuit is given to MTU using simple wired connections. The potentiometer is provided for adjusting the threshold. So it is very much possible to adjust intensity threshold as per the application demands.

3) Motor RPM measurement

Many times in industry we need to monitor the rotational speed of certain machines/motors. The circuit designed here is intended to measure the rotational speed of the motor. This is one of the demo parameters which are to be measured and displayed on the GUI.

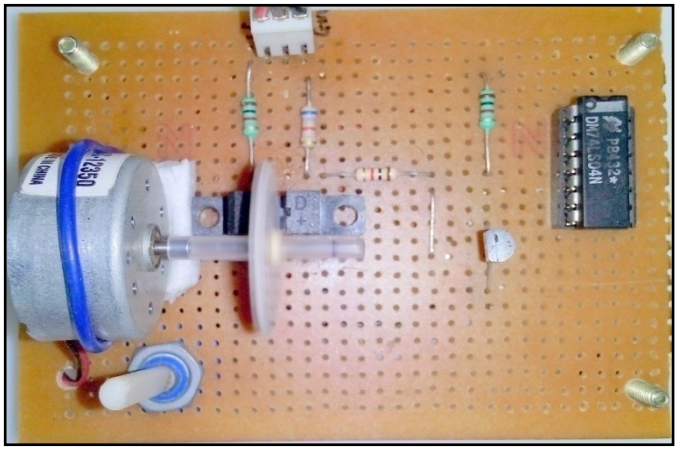


Figure_4: RPM Measurement Module

The circuit uses IC 7811 optocoupler at its core. A motor is fitted on the PCB as shown. A disc is connected to the shaft of the motor. The arrangements are made in such a way that the disc is rotated through the slot present on the optocoupler IC. The disc has a small hole on it.

IC7811: It's an optocoupler IC having a slot with an infrared LED on one side and photo-transistor on the other side. When light from IR-LED falls on the base of photo-transistor, it is turned on. If there are some obstacle in-between them, photo-transistor is turned off. This principle is used to generate the pulses.

Principle: When an IR-LED, photo-transistor and a hole on the disk comes in a single line, photo-transistor is turned on; which in turn drives base of another transistor (Q2); which reduces collector voltage of Q2 to logically zero volt. When disc produces obstacle between photo-transistor and IR-LED, photo-transistor is turned off; which in turn turns off transistor Q2, and hence collector voltage of Q2 becomes

logically high. The disc continues to rotate and pulses are obtained at collector of Q2. These pulses are now applied to MTU for measurement of RPM.

4) Temperature Measurement

Temperature measurement is carried out with the help of IC LM35. It is a digital IC with resolution of 10mV/°C. The IC here operates on the voltage range of 0 Volt to 5 Volt. It produces the analog voltage corresponding to the existing atmospheric temperature.

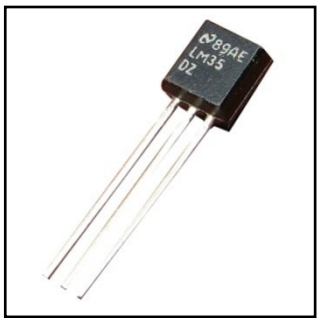


Figure_5: Temperature Sensor

5) Humidity Measurement

Figure_6: Humidity Sensor Module

The humidity sensors module used for humidity measurement is SHR01. The module is capable of indicating the relative humidity in the range of 20% to 90%. The operating voltage is 0 volt to 5 volt. The sensor has negative coefficient. i.e. as humidity increases, voltage decreases. It produces the analog voltage corresponding to the existing atmospheric humidity.

Analog to Digital Converter: IC 0808

ADC-0808 data acquisition component is a monolithic CMOS device with 8-bit analog-to-digital converter, 8-channel multiplexer and microprocessor compatible control logic. The 8-bit A/D converter uses successive approximation as conversion technique. Two analog input channels of the ADC-0808 are used in project. One of them is used for temperature sensor and other one is used for humidity sensor. ADC-0808 produces 8 bit digital output corresponding to the analog input voltage. The resolution of ADC-0808 is (5V/2^8) = 19.87 mV. The ADC-0808 has reference voltages of +5 Volts and 0 Volt.

Procedure:

1. ADD_A, ADD_B, ADD_C are input lines for ADC-0808; used to select one of the 8 available analog inputs. These signals are provided by microcontroller i.e. MTU.
2. Once channel is selected ALE (Address Latch Enable) pin is made high.

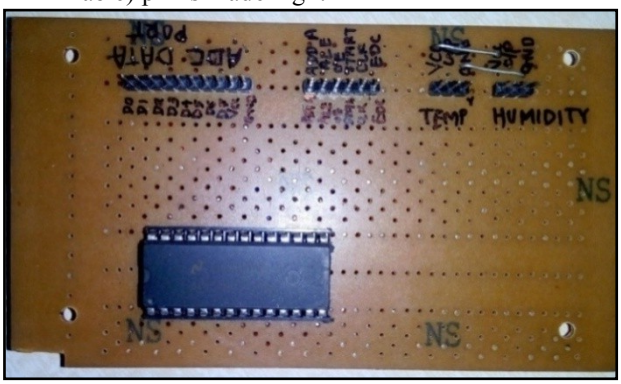


Figure_7: Analog to Digital Converter

3. After this, to command the IC ADC-0808 to start the analog to digital conversion SOC (Start of Conversion) pin is made high.
4. The ADC-0808 requires external crystal of 250 KHz to perform the conversion. In order to reduce the hardware, the external clock is generated and provided by microcontroller itself on CLK pin of the ADC-0808 for duration of 10mSec.
5. After 10mSec EOC (End of Conversion) pin of ADC-0808 is checked to see whether conversion is complete. If EOC pin is still low, CLK is continued further.
6. If conversion is complete, OE (Output Enable) pin of the ADC-0808 is made high by microcontroller to command the ADC-0808 to send the 8 bit converted digital data onto its data port.
7. Finally 8 bit digital data is read by the microcontroller and stored for reference.
8. To select the next analog input for the conversion, addresses ADD_A, ADD_B, ADD_C are changed and same steps are carried out.

In this way analog voltages obtained from temperature sensor and humidity sensor are converted into their digital equivalent.

6) Fire Indication

In industry it is necessary to have protection against the fire. Hence fire detectors are installed at various locations in the industry to form a fire alarm system.

In the project a push button is provided to generate instance of fire mishap in the industry. As soon as this signal is detected by RTU, a loud buzzer is given to warn all the people. The fire alarm push button gives an interrupt to RTU as fire is the most critical event that a RTU is supposed to look after. As soon as push button is pressed, Microcontroller executes its ISR (Interrupt Service Routine) and required actions can be performed in the ISR like buzzing an emergency alarm.

C. REMOTE TERMINAL UNIT (RTU)

Remote terminal unit is basically a substation for gathering a data from various intelligent electronic devices (IEDs) or from various sensors. The RTU may have wired connection with the peripherals, sensors, IEDs or it may use some wireless means of communication. In this project simply a wired connection is established between sensors/IEDs and MTUs.

The substation RTU may still be a traditional RTU with hard-wired inputs and outputs and communicating with one or more control centers. With the proliferation and increasing capabilities of IEDs, however, the RTU may actually combine several different elements into one device. For example, the RTU may provide the substation HMI, the data concentrator, and remote access controller.

RTU architecture:

Figure_8: Remote Terminal Unit shows hardware architecture of RTU. Features of the designed remote terminal unit are,

1. Uses Microcontroller P89V51RD2 (8051 family)
2. LCD (Liquid Crystal Display) to show status of industry. Its features are,
    a. 16x2 character display
    b. 4 pin interface with microcontroller
    c. Green backlight
3. Serial communication with the central server at the baud rate of 9600.
4. Processing power is distributed/given to RTU
5. RTU Rings a buzzer when water level reaches 100%
6. RTU Rings a buzzer when fire is detected

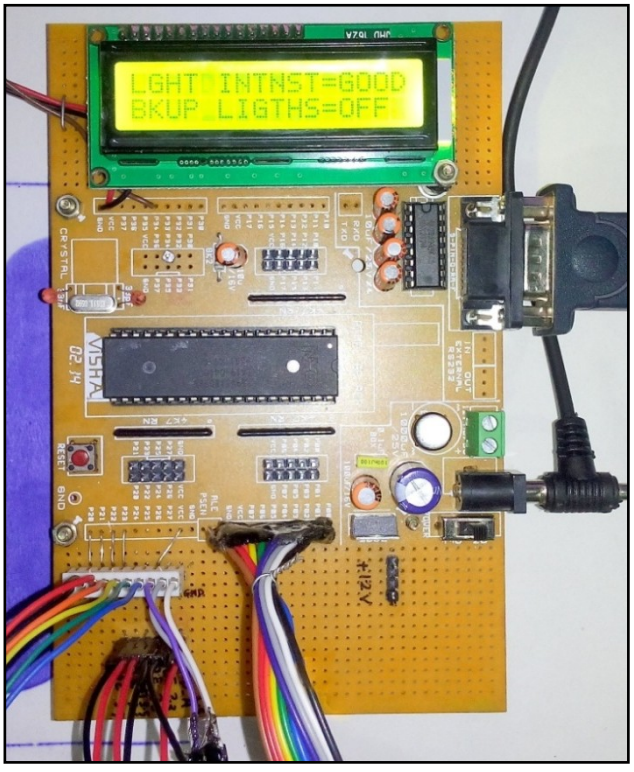


Figure_8: Remote Terminal Unit

Features of Microcontroller P89V51RD2:

- 80C51 Central Processing Unit
- 5 V Operating voltage from 0 MHz to 40 MHz
- 64 Kbytes of on-chip Flash user code memory with ISP (In-System Programming) and IAP (In-Application Programming)
- Supports 12-clock (default) or 6-clock mode selection via software or ISP
- SPI (Serial Peripheral Interface) and enhanced UART
- PCA (Programmable Counter Array) with PWM and Capture/Compare functions
- Four 8-bit I/O ports with three high-current Port 1 pins (16 mA each)
- Three 16-bit timers/counters
- Programmable watchdog timer
- Eight interrupt sources with four priority levels
- Second DPTR register
- Low EMI mode (ALE inhibit)
- TTL- and CMOS-compatible logic levels
- Brown-out detection
- Low power modes
- Power-down mode with external interrupt wake-up Idle mode
- DIP40, PLCC44 and TQFP44 packages

Interfacing of RTU with IEDs and sensors:

LCD: connected to P1.4 to P1.7

LCD control pins:

a. RS : P1.3
b. EN : P1.2

ADC data pins: connected to Port 0 (P0)

ADC control pins:

a. ADD_A : P2.5
b. ALE : P1.1
c. OE : P3.3
d. START : P3.5
e. CLK : P3.6
f. EOC : P3.7

Water level sensor output: P2.0 to P2.3

Ambient light intensity sensor output: P2.4

RPM measurement input: P3.4

Flood Light: P1.0 Fire Alarm: P3.2

Device_1 (LED):P2.6 Device_2 (Buzzer):P2.7

Data sent by Remote Terminal Unit

RTU reads the data coming from the sensor and makes a data packet from it. In this project Microcontroller P89V51RD2 reads the status of various parameters form the firm and makes an array of it which is to be sent to Control Center later. The data collected for each parameter is stored into one of the fixed location of array named as Status_Array. This array is updated at the regular interval of 5 seconds by RTU by reading the status of firm. Then this

Status_Array is sent to Control center. The structure of the array is given in Table_1:Status_Array.

BYTE 0	PULSES
BYTE 1	LIGHT
BYTE 2	WATER
BYTE 3	HUMIDITY
BYTE 4	TEMPERATURE

Table_1: Status_Array

D. MASTER STATION (CONTROL CENTER)

Modern Supervisory Control and Data Acquisition (SCADA) master stations have both software and hardware in a distributed architecture. The processing power is distributed among various computers and servers that communicate with each other through a real-time dedicated LAN in the control center. Further, to access a particular Master station through any place, internet access is provided with password security. Control Center consists of following programs running on it.

1) Serial Data Logging Software
2) Server Side Programming & Web Hosting

Serial data logging software

The name for the software is given as "Data logging and Device Control". The serial data logging software is developed by using VISUAL BASIC 6 with GUI. The software performs following tasks.

- Reads the data coming on the serial port (i.e. the data sent from Remote Terminal Unit)
- Saves the data into "log.txt" file on the server PC.
- Checks if any command character is saved into "control.txt" file on server PC. ("control.txt" is explained later)
- If any control character is present in "control.txt" file, sends that character to RTU using serial communication

The layout of the software window is given into Figure_9: Data Logging and Device Control.

Figure_9: Data Logging and Device Control

The following section gives a description of the software.

```
/*Variable Declaration*/
Dim gateway As String, encoded As String,
controlbit As String, readdone As Long
```

```
/*Code for form Load Event*/

Private Sub Form_Load()
readdone = 0
Text3.Visible = False
End Sub
```

```
/*Code for START button*/

Private Sub Start_Click()
   gateway = Text1.Text
   Text1.Visible = False
   Label1.Visible = False
   Start.Visible = False
 /*Communication settings*/
   MSComm1.Settings = "9600,n,8,1"
   MSComm1.RThreshold = 5
   MSComm1.DTREnable = False
   MSComm1.CommPort = gateway
   MSComm1.PortOpen = True
End Sub
```

```
/*Code for Serial Communication Event*/
Private Sub MSComm1_OnComm()
        If MSComm1.CommEvent = comEvReceive
Then
        encoded = MSComm1.Input

/*Save status of firm (encoded) into file -- to be read by
website*/
Open "C:\xampp\htdocs\explore\log.txt" For Append
As #1
Print #1, encoded Close #1

/*Sending command characters to serial port*/
Open "C:\xampp\htdocs\explore\control.txt" For Input
As #2

        While EOF(2) = False
        Line Input #2, controlbit
        Text3.Text = controlbit
        MSComm1.Output = controlbit
        Wend

        readdone = 1
        Close #2

        If readdone = 1 Then
/*new blank file is created for control characters*/
Open "C:\xampp\htdocs\explore\control.txt" For Output
As #2
```

```
/*Code for Close (X) button*/

Private Sub Close_Click()
Unload Me
End Sub
```

Sequence of Execution

1) The software reads the data coming from RTU on COM port of computer.
2) Saves that data in the text file “log.txt” at location "C:\xampp\htdocs\explore\log.txt" (Default path).
3) Software now checks if any command character is present into text file “control.txt” at location "C:\xampp\htdocs\explore\control.txt" (Default path).
4) If any command character is present, software sends that character to RTU via COM port.
5) The MTU sends the data to Control Center at the regular interval of 5 seconds. So above sequence is repeated on every 5 seconds.

Server Side programming & GUI

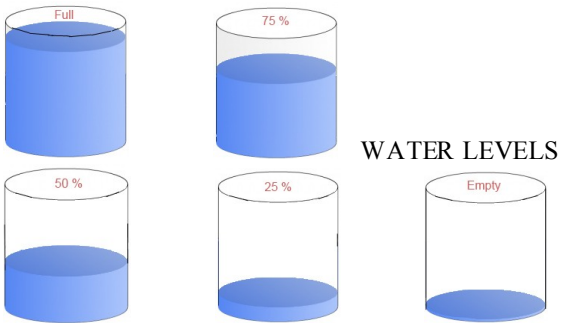
The programming language used for the website designing is HTML (Hyper Text Markup Language) and PHP (HyperText Processor). PHP code is executed on the server itself. HTML and PHP both are open source programming languages which reduces system cost. Tasks performed by PHP code are;

1) Reads the “log.txt” file from the location "C:\xampp\htdocs\explore\log.txt" (Default path).
2) Extracts the last 5 characters from the file, which is nothing but the current status of the firm.
3) The code analyses string & selects corresponding image to be displayed for various parameters.

The webpage showing the status of the firm is given in Image_2: Webpage_Status in the ANNEXURE at the end of the report. The abstract code for the PHP programming is given below.

```
/*code for reading “log.txt” file
$myFile = "C:/xampp/htdocs/explore/log.txt";
$fh = fopen($myFile, 'r');

fseek($fh,-7,2);
$pulses=fread($fh, 1);
$light=fread($fh, 1);
$water=fread($fh, 1);
$humid=fread($fh, 1);
$temp=fread($fh, 1);
fclose($fh);
```

Thus status of the firm is read from “log.txt” file. Various elements of array are extracted and stored into 5 different variables. Later on, switch case statement is used to select and update images. The database for water levels comprises of 5 different images stored on the server.




```
/* code for updating water status */

switch($water)
{
case 'A':
{
echo '<table><td><img border="2"
src="images/banner/00.jpg"
width="240" height="365" alt == "image 0 of 5">
</td></table>';
break;
}

case 'B':
{
echo '<table><td><img border="2"
src="images/banner/01.jpg"
width="240" height="365" alt = "image 1 of 5">
</td></table>';
break;
}

case 'C':
{
echo '<table><td><img border="2"
src="images/banner/02.jpg"
width="240" height="365" alt = "image 2 of 5">
</td></table>';
break;
}

case 'D':
{
echo '<table><td><img border="2"
src="images/banner/03.jpg"
width="240" height="365" alt = "image 3 of 5">
</td></table>';
break;
}

case 'E':
{
echo '<table><td><img border="2"
src="images/banner/04.jpg"
width="240" height="365" alt = "image 4 of 5">
</td></table>';
break;
}

default:
{
echo '<table><td><img border="2"
src="images/banner/00.jpg"
width="240" height="365" alt = "image 0 of 5">
</td></table>';
break;
}
}
```

The same procedure is used to indicate the ambient light intensity. The database comprises of 2 different images to

indicate the status of flood light, which in turn depends on status of ambient light intensity. If ambient light intensity is less than threshold, flood light are turned on. If ambient light intensity is greater than threshold, flood light are turned off. Images in database are given below,


Flood Lights ON


Flood Lights OFF

```
/*code to indicate ambient light intensity & flood light
status*/

switch($light)
{

case 'A':
{
echo '<table><td><img border="2"
src="images/bulb/on.jpg"
width="240" height="365" alt = "image 0 of 2">
</td></table>';
break;
}

case 'B':
{
echo '<table><td><img border="2"
src="images/bulb/off.jpg"
width="240" height="365" alt = "image 1 of 2">
</td></table>';
break;
}

default:
{
echo '<table><td><img border="2"
src="images/bulb/on.jpg"
width="240" height="365" alt = "image 1 of 2">
</td></table>';
break;
}
}
```

The remaining parameters i.e. humidity, temperature, RPM and fire alarm status are indicated using following code.

```
/*calculating status of parameters*/
$rpm= ord($pulses)*60;
//ascii to integer conversion
$temperature=ord($temp)*2;
$humidd=ord($humid)* 0.2745;
$humidity=abs(90-$humidd);
```

```
/* Displaying Status of parameters*/
<?php
echo "Relative Humidity:  $humidity %";
?>
<?php
echo "Spindle Speed:  $rpm RPM";
?>
<?php
echo "Temperature:  $temperature °C";
?>
<?php
echo "Fire Alarm:  $alarm ";
?>
```

Command buttons for controlling the devices

In the hardware, 2 devices are used for demonstration purpose that can be controlled from internet. The devices used in the hardware are buzzer and LED. In PHP programming, 4 command buttons are used to control the devices. The buttons are Device_1 ON, Device_1 OFF, Device_2 ON and Device_2 OFF which are used to Turn On LED, Turn Off LED, Turn On Buzzer and Turn Off Buzzer respectively. On the click of command buttons, various actions can be performed in PHP programming. The task performed here is, saving a control character in “control.txt” file at the path given by C:/xampp/htdocs/explore/control.txt. The code for saving control character is given below,

```
/* code for saving the control character*/
<?php
        $myFile = "control.txt";
        $fh = fopen($myFile, 'a+');

        if($_GET["formVar"] == "formA")
                $stringData = 'A';
        if($_GET["formVar"] == "formB")
                $stringData = 'B';
        if($_GET["formVar"] == "formC")
                $stringData = 'C';
        if($_GET["formVar"] == "formD")
                $stringData = 'D';
        fwrite($fh, $stringData);
        fclose($fh);
?>
```

On click of Device_1 ON button, character ‘A’ is store in “control.txt” file. In the same way character ‘B’,’C’,’D’ are stored in control.txt file on the click of Device_1 OFF, Device_2 ON, Device_2 OFF buttons respectively. The same “control.txt” file is used by Data logging and Device control software at regular intervals. The file is read and control characters are sent serially to MTU as stated earlier.

Authentication for webpage access

Figure_10: Password Authentication

On typing the URL of the Control Center, first webpage appears as a password authentication webpage. A status webpage will be displayed only after entering valid password. Server has IPSec security running on it in order to ensure security. This can be extended to VPN (Virtual Private Network) also.

GUI for local machine

In order to give the status of the firm only on local machine, Visual BASIC 6 software is used for developing GUI. The implemented GUI is shown in the Image_3: GUI Local Machine in ANNEXURE at the end of report.

GUI window is divided into two sections.

-System Status

-System Control

System Status

Tasks performed by program are,

1) Reads the data coming from RTU i.e. Microcontroller P89V51RD2. i.e. it receives a Status_Array.
2) The received array is decoded i.e. string parsing.
3) Depending upon the characters received particular image is selected from database and the values for different parameters are updated.

Abstract code and description is given in Code_block_1.

The program works similar to PHP code. Various switch case statements are given below. The Status_Array is decoded and stored into individual variables as shown Table_2: Array_Extraction

Status_Array[0]	pulses
Status_Array[1]	light
Status_Array[2]	water
Status_Array[3]	humidity
Status_Array[4]	temperature

Table_2: Array_Extraction

System control

In order to control the output devices connected, four control buttons are provided. The buttons are Device_1: ON & OFF, Device_2: ON & OFF. On click of the command buttons, characters are sent serially to the microcontroller. The characters sent are 'A','B','C','D' for click on Device_1: ON, OFF and Device_2: ON,OFF respectively. The characters are sent serially to RTU. Abstract code for system control section is given in Code_block_2.

```
/*TEXT BOX 1 GIVES PULSE COUNT== RPM*/
a = Asc(Mid(encoded, 1, 1))
pulses = a * 60
 /* number of pulses in 1 second *60 is RPM*/
Label13.Caption = pulses

/*TEXT BOX 2 AMBIENT LIGHT INTENSITY
STATUS*/
  light = Asc(Mid(encoded, 2, 1))

Select Case light
  Case 65       /*A means good intensity*/
  Label9.Caption = "Ambient Light GOOD"
  Label10.Caption = "Flood Lights OFF"
  Image2.Picture = LoadPicture(App.Path &
"\bulb_off.jpg")

  Case 66       /*B means bad intensity*/
  Label9.Caption = "Ambient Lights BAD"
  Label10.Caption = "Flood Light ON"
  Image2.Picture = LoadPicture(App.Path &
"\bulb_on.jpg")

   /*TEXT BOX 3 water level status*/
  water = Asc(Mid(encoded, 3, 1))

  Select Case water
  Case 65          'check for character A
  Image1.Picture = LoadPicture(App.Path & "\0
thumb.jpg")
  Label8.Caption = "EMPTY"

  Case 66          'check for character B
  Image1.Picture = LoadPicture(App.Path & "\1
thumb.jpg")
  Label8.Caption = "25%"

  Case 67          'check for character C
  Image1.Picture = LoadPicture(App.Path & "\2
thumb.jpg")
 Label8.Caption = "50%"

  Case 68
  Image1.Picture = LoadPicture(App.Path & "\3
thumb.jpg")
 Label8.Caption = "75%"

  Case 69
  Image1.Picture = LoadPicture(App.Path & "\4
thumb.jpg")
  Label8.Caption = "FULL"
  End Select

  /*TEXT BOX 4 gives humidity percentage*/
    h = Asc(Mid(encoded, 4, 1))
    humid = h * 0.2745
    humidity = 90 - humid
  Label11.Caption = humidity

  /*'TEXT BOX 5 gives temperature*/
  t = Asc(Mid(encoded, 5, 1))
  temperature = t * 2
  Label12.Caption = temperature
                  //****Code_block_1****//
```

```
Private Sub Command1_Click()
 MSComm1.Output = "A"
Command1.BackColor = vbGreen
Command1.Enabled = False
Command2.Enabled = True
Label22.Caption = "Device 1 is ON"
 End Sub

Private Sub Command2_Click()
 MSComm1.Output = "B"
 Command1.Enabled = True
 Command2.BackColor = vbRed
 Command2.Enabled = False
 Command1.BackColor = vbButtonFace
 Label22.Caption = "Device 1 is OFF"
 End Sub

Private Sub Command3_Click()
MSComm1.Output = "C"
Command3.BackColor = vbGreen
Command3.Enabled = False
Command4.Enabled = True
Label24.Caption = "Device 2 is ON"
End Sub

Private Sub Command4_Click()
MSComm1.Output = "D"
 Command3.Enabled = True
 Command4.BackColor = vbRed
 Command4.Enabled = False
 Command3.BackColor = vbButtonFace
  Label24.Caption = "Device 2 is OFF"
End Sub
```

//**Code_block_2****//**

Server configuration for control center

The software, XAMPP is used to configure personal computer as a server. XAMPP is open source software which eliminates software licensing cost.

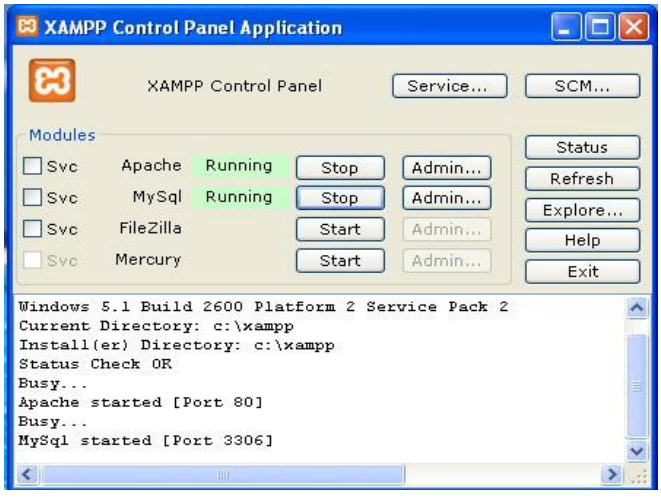


Figure 11: XAMPP Control Panel

The website for the project is saved into the directory "C:\xampp\htdocs\explore" (default path). So, to access a "home.php" page from browser we need to specify IP address or domain name of Control Center, i.e. http://<ip>or<domain_name>/explore. User is automatically redirected to password authentication window (Figure_10).

IV. USE OF IEEE STANDARDS

1) IEEE C37.1 SCADA and automation standards are implemented.
2) Two control centers can communicate with each other by using IEEE 802.3 LAN standard. (Project currently does not involve multiple Control Centers)
3) Internet is used to access particular Control Center with static IP address/ Domain name from distant place.

V. CURRENT STATE OF THE PROJECT (WORKING PROTOTYPE)

The following section will give the exact details of the project functionality in sequential manner with few snapshots. Abstract code for microcontroller program is given in Code_block_3 on next page.

Program Execution flow:

- ***a)*** All the port directions are initialized as per the hardware demands. The pins are configured as input if we want to read the pins. The pins are configured as output if we want to use that pins to signal the other devices/pins.
- ***b)*** LCD which is present on RTU is initialized for displaying the status of the firm locally.
- ***c)*** ADC-0808 is initialized for reading the analog voltages coming from temperature sensor and humidity sensor.
- ***d)*** Serial communication of the microcontroller is configured for communicating serially with Control Center.
- ***e)*** Welcome message is displayed on the screen after all the initializations are made.

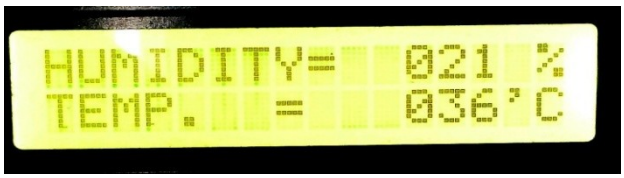
Now the following sequence of code instructions is executed repeatedly.

- ***f)*** Analog Channel zero (AN0) of ADC-0808 is read for reading analog voltage coming from humidity sensor.
- ***g)*** The humidity is displayed on the LCD screen as well as Status_Array[3] is also updated.
- ***h)*** Analog Channel one (AN1) of ADC-0808 is read for reading analog voltage coming from temperature sensor.

```
/*HEADER FILES FOR PROGRAM*/
#include <reg51.h>
#include <math.h>
#include <serial_functions.h>
#include <lcd_4bit_functions.h>
#include <adc0808.h>
#include <misc_functions.h>
#include <LCD_and_TRANSMISION.h>

void main (void)
{
Initialize_port_directions();                          (a)
InitLCD();          //initialize LCD                   (b)
InitADC();          //initialize ADC                   (c)
configure_serial();  //initialize serial communication (d)
ClearLCDScreen();
welcome_message();                                     (e)

while (1)
{
/*HUMIDITY MEASUREMNT*/
read_adc_channel_zero();                               (f)
display_lcd_channel_zero();                            (g)

/*TEMPERATURE MEASUREMENT*/
read_adc_channel_one();                                (h)

display_lcd_channel_one();                             (i)
delay_1s();                                            (j)
delay_1s();

/*SENSING WATER LEVEL*/
detect_water_level();                                  (k)
display_lcd_water_level();                             (l)

/*RPM MEASUREMENT*/
WriteCommandToLCD(0xc0);
WriteStringToLCD("MEASURING RPM...");                  (m)

measure_pulses();
display_lcd_rpm();                                     (n)
delay_1s();                                            (o)
delay_1s();

/*AMBIENT LIGHT INTENSITY MEASUSREMNT*/
detect_ambient_light_intensity();                      (p)
display_lcd_ambient_light_intensity();                 (q)
delay_1s();                                            (r)
delay_1s();

/*SEND STATUS_ARRAY TO CONTROL CENTER*/
send_status_to_serial_port();                          (s)

 }/*while(1) ends here*/
 }/*end of main*/
```

//** Code_block_3*****//**

i) The temperature is displayed on the LCD screen as



well as Status_Array [4] is also updated.

j) The status of humidity and temperature is kept on LCD for 2 seconds.

k) The status of the water level in the reservoir tank is read by reading status of port pins. The Status_Array[2] is also updated.

l) Water level is displayed on the LCD in terms of percentage in the steps of 0%, 25%, 50%, 75% and 100%. 0% and 100% corresponds to EMPTY and

FULL level.

m) The measurement of the RPM is carried out by the referred function. Pulses coming from the sensor are measured for the duration of 2 seconds. The count of pulses is updated in TL0 register of microcontroller. The Status_Array[0] is also updated. RPM can be calculated as,
RPM= number of Pulses*30;

n) The water level and RPM is displayed simultaneously on the screen.

o) The status of water level and RPM is kept on the screen for 2 seconds.

p) The status of pin to which output of ambient light intensity sensor is connected is now read. If ambient light intensity is less than particular threshold level, microcontroller itself turns on flood light connected to one of its pins. *This is how; processing power is given to RTU.* At the same time Status_Array[1] is also updated.

q) Status of ambient light intensity along with status of

flood light is displayed on LCD screen.

r) The status of ambient light intensity and status of flood lights is kept on LCD screen for 2 seconds

s) Once all the measurement and display function are over, Status_Array contains 5 latest values of system status. This Status_Array is sent serially to microcontroller.

Now, programs flow goes back to stage (f). This is how a RTU works.

VI. CONCLUSION

This project describes the implementation of the IEEE C37.1 SCADA and automation standard terminologies, having a combination of hardware and software. The project involves 6 parameters to be monitored, one RTU and one Control Center to form a small firm to demonstrate SCADA system.

Implementation of the terminologies mentioned in the documents is the heart of the project. The expected implementation of IEDs, RTUs, Control Center and sensors is done to fulfill their features. The project involves a challenging part of developing software applications on the server side to emulate the functions of the Control Center.

The Visual Basic 6 and PHP programming can be used at the Control Center with limited scope of automation i.e. system can be developed for a small firm with few parameters to be monitored and displayed. The project uses website designing languages like HTML, PHP which are open source which eliminates the investment cost so as to reduce the development cost of project. The XAMPP software which is again open source provides an efficient tool for server configuration at no cost.

VII. REFERENCES

[1]http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?reload=true&punumber=4518928
[2]http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=04518930
[3]http://msdn.microsoft.com/en-us/library/kehz1dz1%28v=vs. 90%29.aspx
[4] http://en.wikipedia.org/wiki/SCADA
[5] http://www.w3schools.com/PHP/
[6] http://www.w3schools.com/html/default.asp

About the author:

Aman M. Maldar is a Final Year Electronics student (B.Tech. Electronics) in Walchand College of Engineering, Sangli (an Autonomous Institute) from Maharashtra, India. He has received a grant from "IEEE Standards Education" for the project "Implementing SCADA System for Industrial Environment Using 'IEEE C37.1' Standards". He has presented his project "Smart Public Transport System with Bus Arrival Timing and Vacancy Indication" in "Texas Instrument Analog Design Contest 2014". He has been also shortlisted in "EnginX project competition 2013" organized by Tata Consultancy Services, India.

About the mentor:

Sunil G. Tamhankar (M. E. Electronics) is Assistant Professor Electronics Engineering, in Walchand College of Engineering, Sangli from Maharashtra, India. He has 2 years of industrial and 15 years of academic Experience, currently working in the area of **C**yber **P**hysical **S**ystem (CPS) communication protocols and security.

ANNEXURE: Images

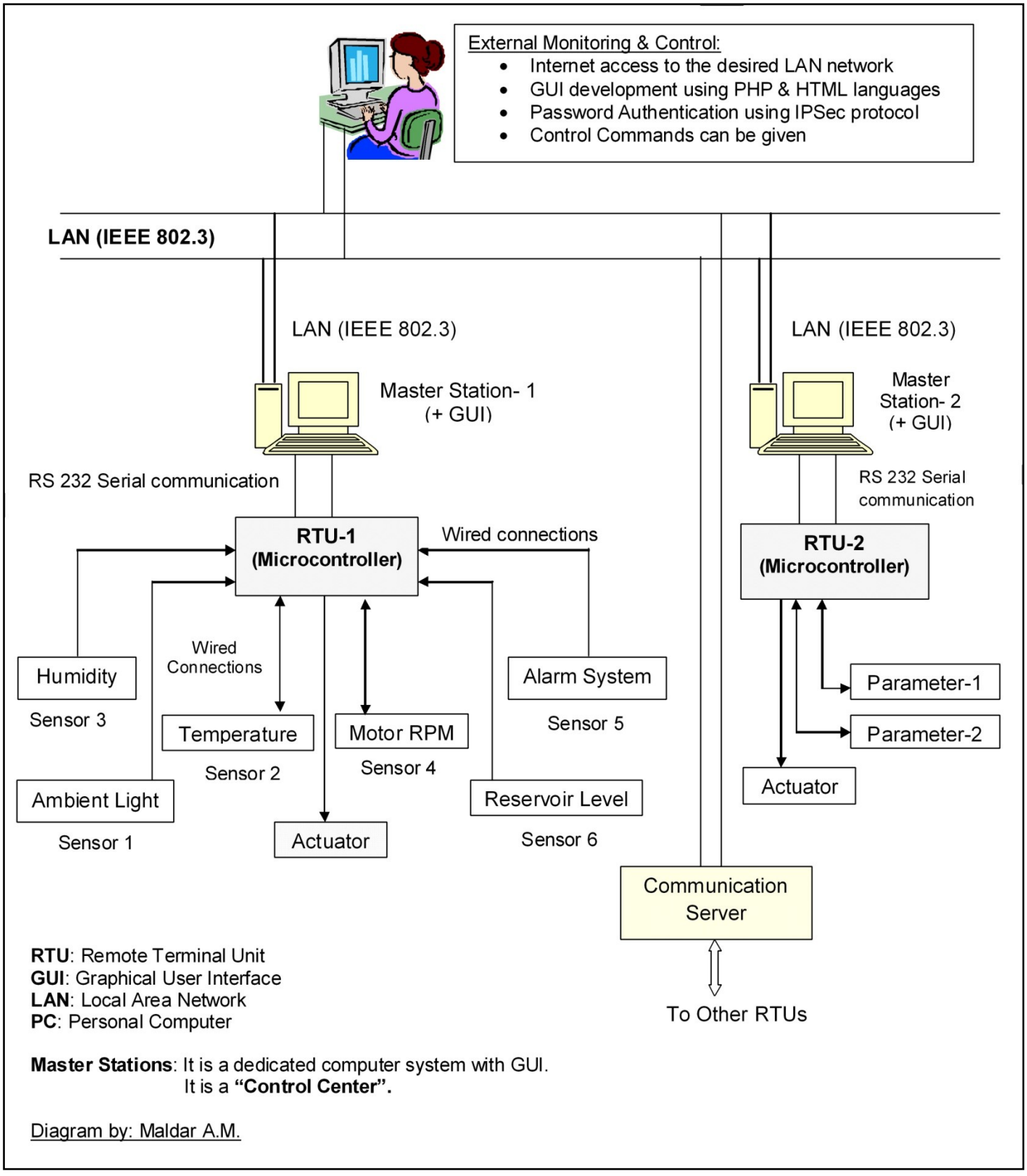


Image_1: System Overview

Image_2 (next image) show snapshot of actual webpage designed for the showing system status.

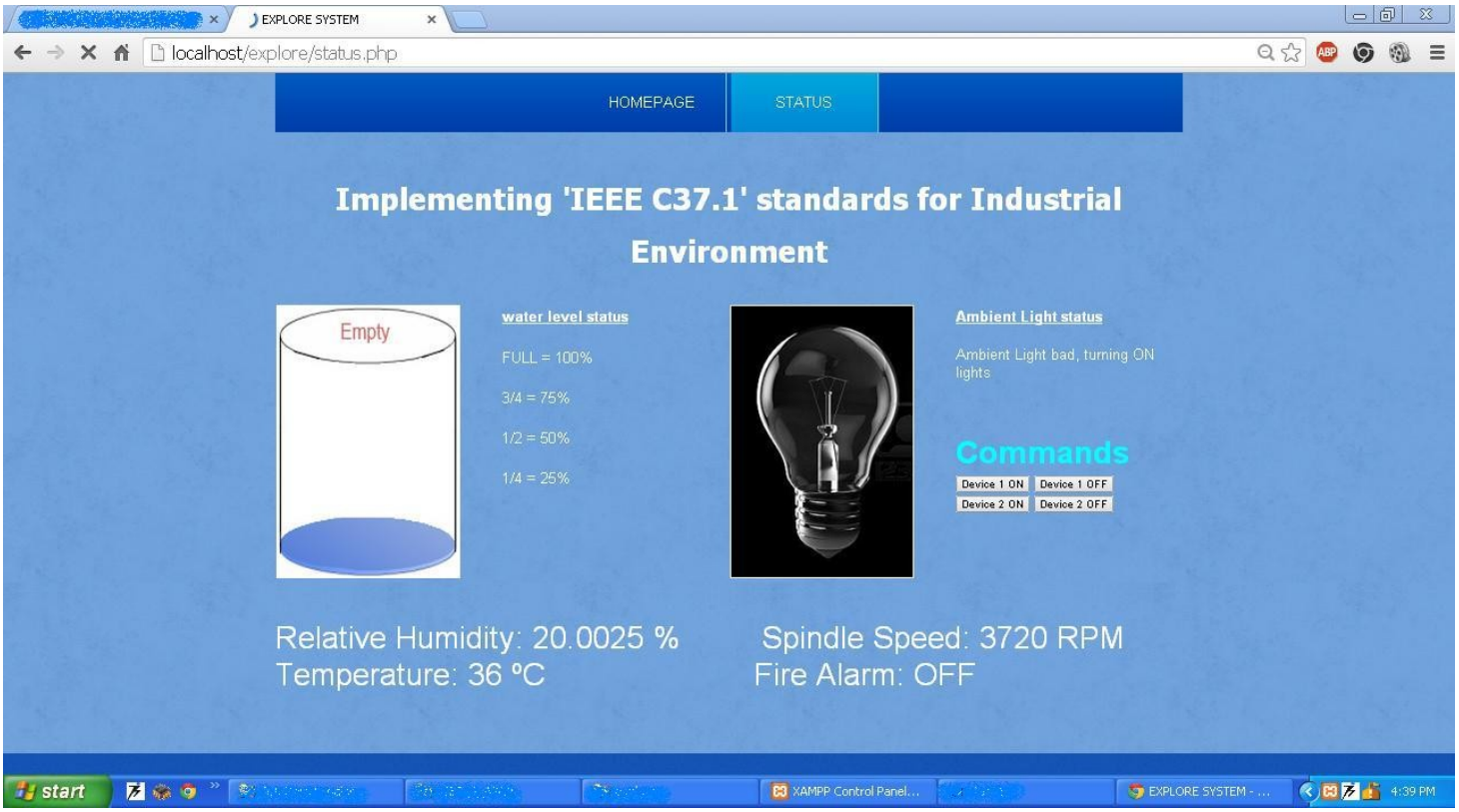


Image_2: Webpage Status

Image_3 (next image) shows snapshot of the actual GUI at the Control Center designed using VISUAL BASIC 6.

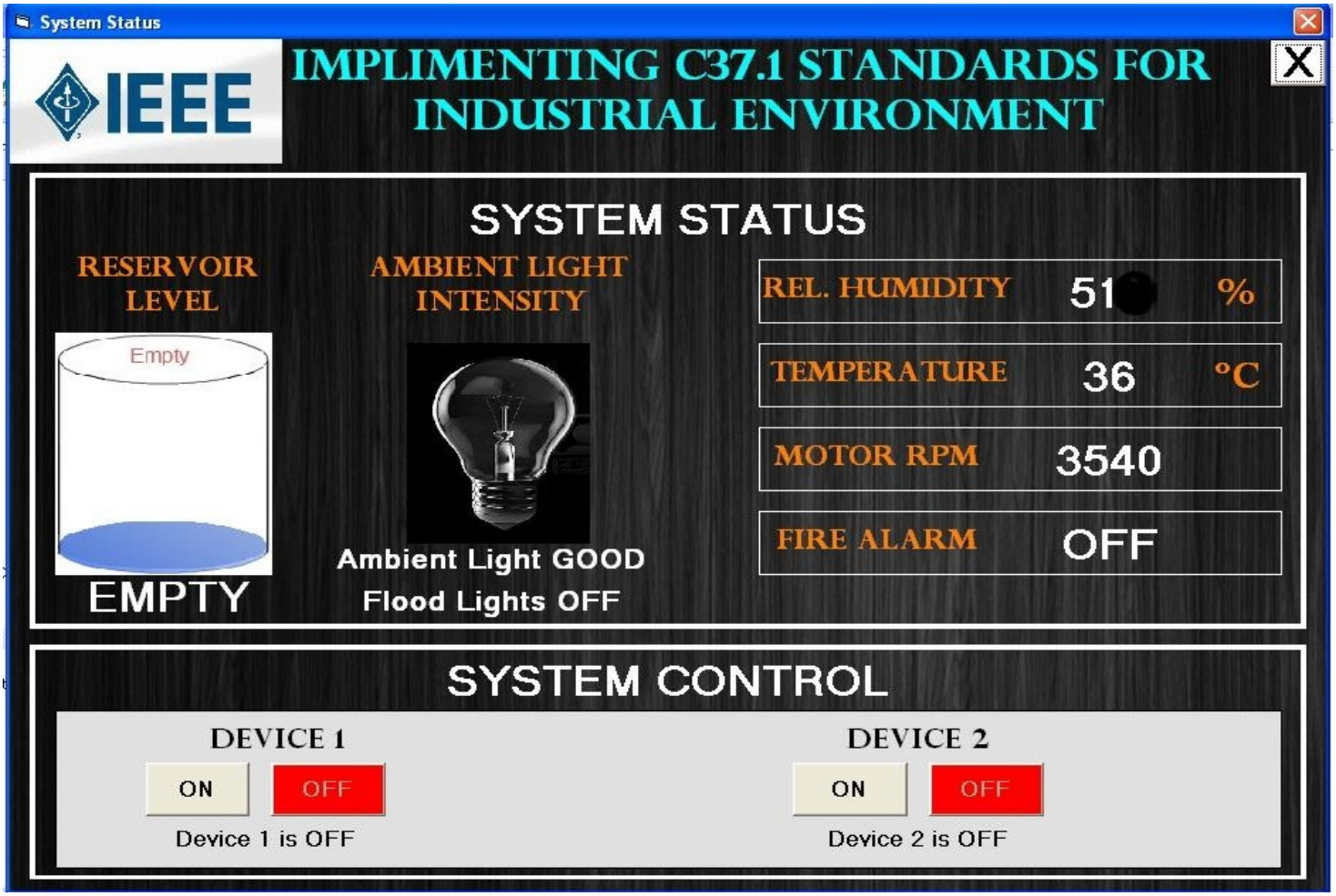


Image_3: GUI Local Machine

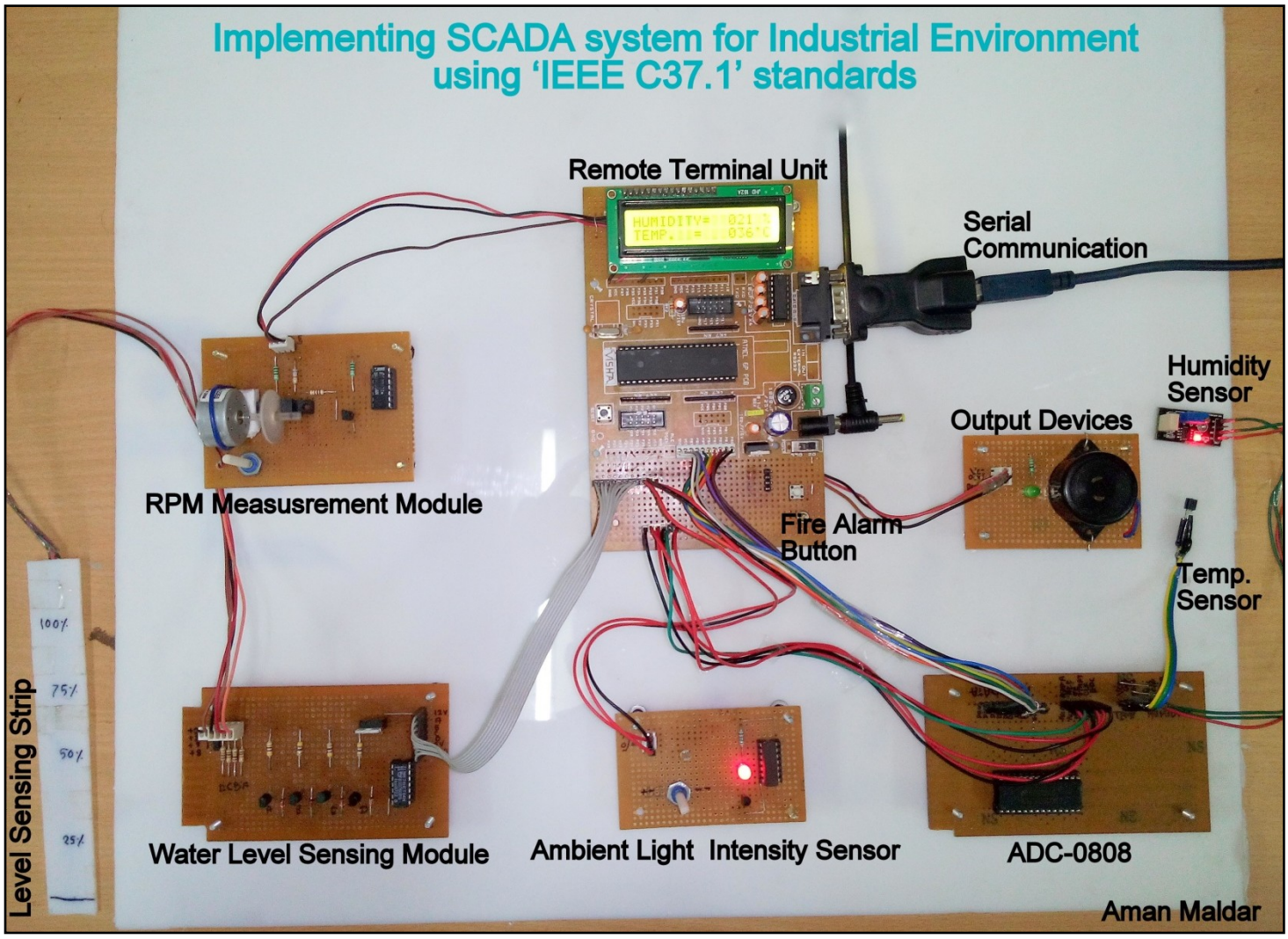


Image_4: Project Hardware (Actual project hardware designed for the system)

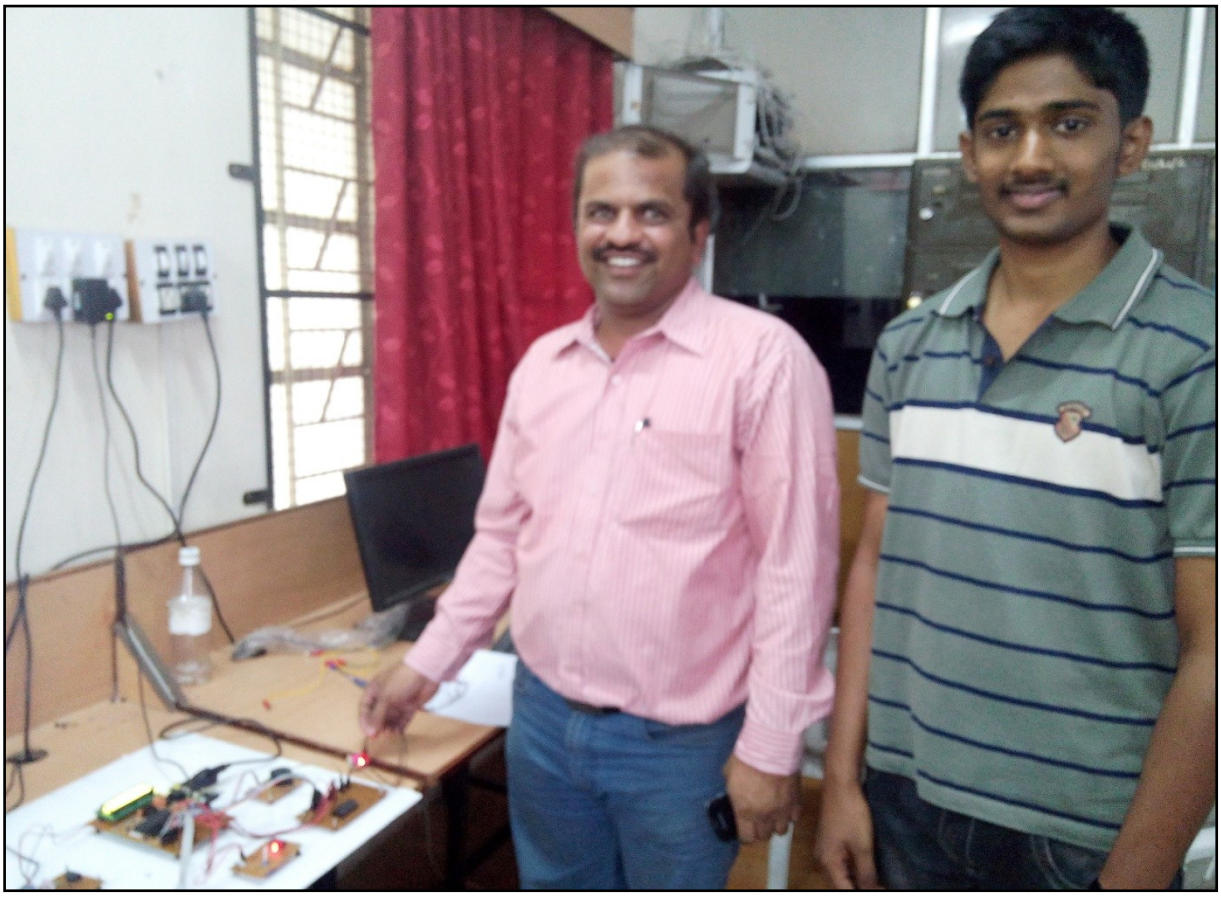


Image_5: Author (Right) & Mentor (Left) with project