International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

A Microcontroller Based Embedded System Design for Device Automation and Control in Intelligent Buildings

[1]Oluwole O. Oyetoke & [2]Adedayo A. Adedapo
[1]Research Scholar, Covenant University, Ota, Nigeria
[2]Research Scholar, Ladoke Akintola University of Technology, Oyo State, Nigeria
EMAIL: [1]oluwoleoyetoke@gmail.com ; [2]dayofemi97@yahoo.com

Abstract -

Over the past decades, the world of technology has transitioned from the use of gigantic electronic centers to micro programmable systems and chips which can be embedded within other systems to make them more effective and efficient. These systems have now found applications in modern intelligent building designs which are established on the use of technology in the creation of buildings that are safer, more productive and operationally efficient for their occupants. This paper explores the mechanism involved when utilization an embedded system (PIC microcontroller and other electronic peripherals) in the development of a system for automated Fan and Air Conditioner switching operation in an intelligent building. This is especially giving consideration to user's preset Switch ON and Switch OFF temperatures. The practical representation of this system is shown using the ISIS Proteus 8.0 simulation package.

Key Words: Embedded Systems; Microcontroller; Intelligent Building; Temperature; Automation; PIC; programming; LCD; LM35

I. INTRODUCTION

The natural consciousness of switching off air-cooling and conditioning devices at moments when there is a considerably high level of diminishing return is notably absent in many buildings today. Home owners at times leave their cooling devices switched on for stretch hours within which some break-time could have been well afforded, without adverse effect(s) on the condition of the environment within which they are serving. This explains why it is expedient to consider this method of energy management in the holistic design of an intelligent building.

As buildings become increasingly automated, it will be an advancement to have hundreds and thousands of these buildings regulate their cooling devices' switch-ON and OFF periods to only when they are effectively needed. This will go a long way in reducing energy wastages. As they say, little drops of water make the ocean.

In this design, the regulation is done in the building by simply interfacing a Microcontroller Unit (MCU) with an analogue thermometer and switch circuit. The MCU carries out the periodical switch ON/OFF conditional checks and switch trigger action, based on the temperature it receives from the thermometer. Also, combinations of transistors and relays do the actual switching.

The circuit is designed and tested using the Proteus 8.0 simulation package. Proteus 8.0 is a Virtual System Modeling (VSM) that combines circuit simulation, animated components and microprocessor models to co-simulate the complete microcontroller based designs. This is the perfect tool for engineers to test their microcontroller designs before constructing a physical prototype in real time. It allows users to interact with the design using on-screen indicators and/or LED and LCD displays and, if attached to the PC, switches and buttons. Proteus is generally referred to be the best simulation software for various designs and microcontroller involved projects One of the main components of Proteus 8.0 is the Circuit Simulation -- a product that uses a SPICE3f5 analogue simulator kernel combined with an event-driven digital simulator that allow users to utilize any SPICE model by any manufacturer.

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

II. RELATED WORK

1. Adewale A. A., Isaac Samuel, Awelewa A. A. and Dike U. Ike proposed the "Design and Development of a Microcontroller Based Automatic Switch for Home Appliances". In this paper, the design and implementation of 2 devices where explored. One was a device which helps detect the presence or movement of persons in a room, so as to switch ON the light (240V) if this condition is true and OFF if false. The other device was used to turn ON or OFF the fan in the room based on a certain preset temperature of the room and also *iff* a presence or movement is detected [1].

2. Poonam Lakra and Dr. R. P. Gupta proposed the design of a "Microcontroller Based Automatic Control Home Appliances". The paper presented the Automatic control of home appliances including Room Light and Fan Controller Using Microcontroller AT89C51, a reliable circuit that takes over the task of controlling the room fan and room lights as well as counting number of persons / visitors in the room very accurately .When somebody enters into the room then the counter is incremented by one and the light in the room will be switched ON and when any one leaves the room then the counter is decremented by one. The speed control of fan will depend on PWM signal. It contain temperature sensor that can sense the temperature & gives the control commands for microcontroller .Then microcontroller increases as well as decreases the speed of the fan.. The total number of persons inside the room is also displayed on the liquid crystal display. The microcontroller does the above job. The main objective of control is to get the desired output and in energy conservation [2].

III. INTELLIGENT BUILDING DESIGN THEORY

Over the last 20 years, there has been a lot of discussion and debate about the concept of an "intelligent building." Work has gone on in many forums to define and quantify what the term really means. The end result of all of these efforts is that an intelligent building is not just one thing.

There are a multitude of definitions with different levels of detail and varying degrees of emphasis on various aspects of building intelligence. The first definition, coined by the Intelligent Buildings Institute, defines an intelligent building as 'one which provides a productive and cost-effective environment through optimization of four basic elements: structure, systems, services and management, and the interrelationship between them'. According to this initial definition, an intelligent building is one that optimally matches these four elements to the users' needs with an emphasis on the technology that makes the interrelationship between the elements possible. [3]

Other sources term building automation to be the automatic centralized control of a building's heating, ventilation and air conditioning, lighting and other systems through a Building Management System or Building Automation System (BAS). The objectives of building automation are improved occupant comfort, efficient operation of building systems, and reduction in energy consumption and operating costs [4]. Although there are multiple and evolving perspectives on all these subjects, it is becoming increasingly clear that an intelligent building is a connected and efficient building which proffers easy management to the house/building owner

The origins of Intelligent Buildings and Building Management Systems have roots in the industrial sector in the 1970's, from the systems and controls used to automate production processes and to optimize plant performances.

Programmable Logic Controllers (PLC's) formed the original basis of the control technologies and then later, developments in commercial and residential applications, were based on 'distributed-intelligence microprocessors' [4].

Automated Control of services within building is broadly categorized into four method types:

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

1. ***Time-based Control:*** This method provides heating, lighting, etc services, given preferences to the *time* due
2. ***Temperature & Humidity - based Control:*** This method provides heating or cooling services given preferences to the *temperature* or *humidity* level
3. ***Motion Based Control:*** This method provides services given preferences to the *motion actions*
4. ***Optimizer Parameter based:*** This is more of a general category representing services offered based on key parameters set. E.g Security key codes for safe door openings or sophisticated gadget switching ON and OFF

IV. METHODOLOGY

A major phase of interactivity in this module is between the thermometer (LM35) and the ADC terminal of the MCU. The data derived from this interaction is the adjudging parameter used by the MCU for the switch on/off decision.

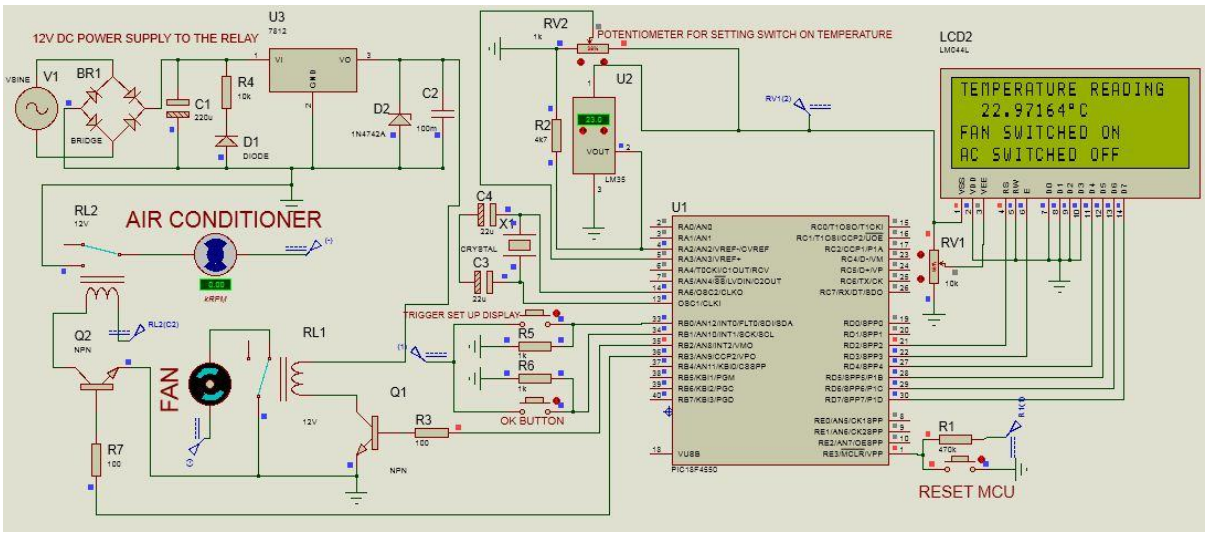

Fig. 1. Schematic of the Module

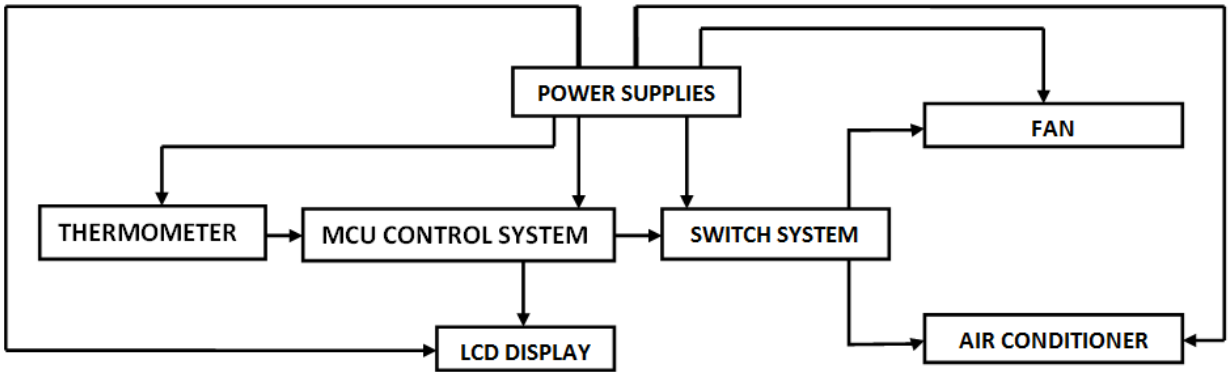

Fig. 2. Block Diagram of the Embedded System's Design

1. Power Supply

The module requires two 12V DC Source (for the 12-volt relays) and a number of other 5V DC supplies to power components such as the MCU, LCD, thermometer and the MCU's set control buttons. In the case of the 12V DC supply, the 240V, 50Hz AC signal is stepped down through a transformer and rectified using a bridge rectifier. The 220uF electrolytic

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

capacitor stabilizes the output of the rectifier circuit as the7812 regulator regulates the rectified voltage to 12V DC (max). The combination of the Zener and regulator makes sure the output does not exceed 12V. Capacitor C2 helps to protect the regulator from excessive voltage demand by the load.

2. Thermometer Reading

The thermometer, an LM35 module, passes an analogue signal proportional to its temperature reading into the MCU. The MCU converts this analogue value into digital values using its ADC channel, sends it to the LCD for display and also uses it for the switching trigger decision making.

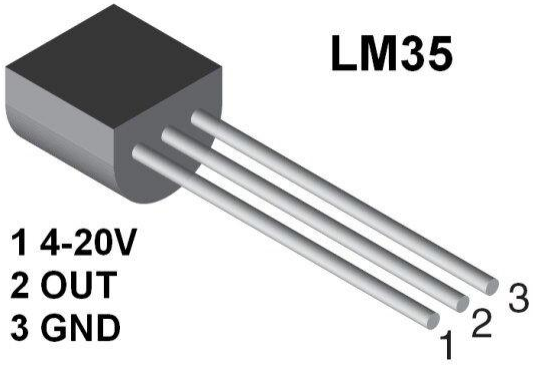

Fig 3. Image of the LM35 Module [6]

3. MCU Control System

The Microcontroller (PIC18F4550) which is a single chip computer, serves technically as the brain of the system. Synonymous to its name is its function. Microcontrollers are used to execute simple and not so complex task within a system. A close relative of these are the microprocessors which are far more sophisticated and can execute numerous operations, simultaneously (such as you have in our modern day desktop computers). A microcontroller is also called an embedded controller because the microcontroller and its support circuits are often built into or embedded in the devices they control. In this system, it is programmed to:

i. Read the instantaneous temperature value from the thermometer
ii. Request, read and store the Fan switch on and switch off temperature
iii. Request, read and store the AC switch on and switch off temperatures
iv. Decide the switching on and off of the air cooling and conditioning systems by comparing the instantaneous temperature with the switch on and off temperature parameters provided by the user
v. Electrically triggers the switching on and off of the air cooling and conditioning stems in the home after making the decision in 'iv' above
vi. Send display data to the interfacing LCD

The MCU uses an interrupt system to ensure that the user can at any instance of time change the switch on and off temperature parameters. This is enabled by the connection of a 5V source through a press button to the interrupt pin (INT0) of the MCU. Once the button is pressed, the interrupt routine is triggered, thereby calling the setup display screen.

Interrupts form the basis for separating the time-critical events from the others and execute them in a prioritized manner. An interrupt tells the microcontroller to drop whatever it is doing and execute another program stored at a predefined place in the program memory. Interrupt requests are asynchronous, which means that an interrupt request can occur at any time during the execution of a program. Note that a microcontroller has several sources of interrupts which can be external or internal.

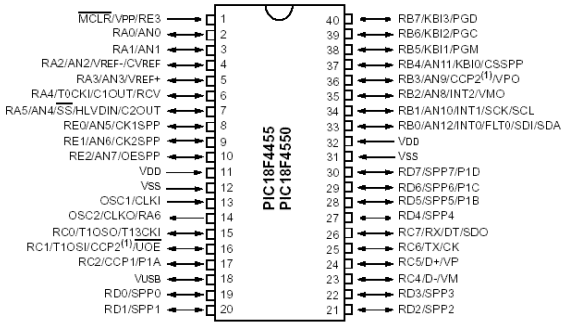

Fig. 4. Schematic of the PIC18F4550 MCU [7]

The analogue temperature reading is converted and interpreted by the MCU using the Analogue to Digital Conversion Channels. Details of this is explained in the section below

3.1. Digital Nature of the MCU and Resolution

MCUs are digital in nature. They can only differentiate between HIGH or LOW level on input pins. For example if input is more than 2.5V it will be read as 1

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

and if it is below 2.5 then it will be read as 0 (in case of 5v systems). So we cannot measure voltage directly from MCUs. To solve this problem most modern MCUs have an Analogue to Digital Converter (ADC) unit. This helps in converting a voltage value to a number so that it can be processed by a digital system like MCU.

MCU's ADCs are in different specifications, mostly differentiated by their resolutions (8bit, 10bit, 12bit etc). Putting into considerations the fact that the MCU analogue pins can only measure a range of 0 to 5V input voltage. The level of precision to which this voltage can be measured is highly dependent on the resolution of the ADC. For example an 8 bit ADC will successfully break the 0 to 5V range to 256 different levels, meaning, it can measure accurately, a 19mV (5/256) change in voltage level conveniently. Likewise, a 10 bit ADC will divide the measurement range to 1024 various levels, meaning a 5/1024 = 4.8mV change can be successfully detected.

Table 1: ADC Resolution Parameters

Resol-ution	Bits	Resol-ution	Maximum	
			ADC Value	Voltage Value
8 bit	11111111	19mV	256	5V
10 bit	1111111111	4.887 mV	1024	5V
12 bit	111111111111	1.22m V	4096	5V

$$RESOLUTION = \frac{(v_{ref+} - v_{ref-})}{(Maximum\ ADC\ Value - 1)} \ldots\ldots\ldots (Eq\ 1)$$

$$(v_{ref+} - v_{ref-}) = 5\ by\ default\ for\ MCU, except\ changed$$

$$RESOLUTION = \frac{5}{Maximum\ ADC\ Value - 1} \ldots\ldots\ldots\ldots (Eq\ 2)$$

The key point to note here is that, using a 10 bit ADC, for every 4.887mV change in input voltage level, the ADC value changes by 1 e.g. a change in input by 39.04mV will give a change in ADC value of 8

3.2. ADC Reference Voltage

The reference voltage specifies the minimum and maximum voltage range of analogue input. For example if the input signal Vref- is applied to an analogue input channel then the result of conversion will be 0 and if voltage equal to Vref+ is applied to the input channel the result will be 1023 (max value for a 10 bit ADC).

More than one Analogue signal can be connected to the MCU, this is because the ADC module of the MCU is connected to several channels via a multiplexer. The multiplexer can connect the input of the ADC to any of the available channels.

3.3. ADC Acquisition Time

When a specific channel is selected the voltage from that input channel is stored in an internal holding capacitor. It takes some time for the capacitor to get fully charged and become equal to the applied voltage.

3.4. ADC Clock

ADC requires a clock source to do its conversion, this is called ADC Clock. The time period of the ADC Clock is called T_{AD}. It is also the time required to generate 1 bit of conversion. The ADC requires 11 T_{AD} to do a 10 bit conversion.

3.5. Digital Conversion and Interpolation

The ADC used for this device is a 10bit ADC. The read ADC value is converted to its 5V equivalent by the equation below

$$0\ to\ 5\ volts\ equivalent = \frac{Read\ ADC\ Value}{204.6} \ldots\ldots\ldots (Eq\ 3)$$

This is because the maximum possible ADC reading for a 10 bit ADC is 1023, and 1023/204.6 = 5, therefore, 204.6 is the divider value necessary to get the 0 to 5 volt equivalent of the ADC reading (for a 10 bit ADC).

4. Switch Circuit

The switching circuit is achieved by the interaction between relays and an NPN transistors used to make an emitter follower switch. The relay is an electromagnetic switch with an electromagnetic centre operated by a relatively small electric current. The coil of wire becomes a temporary magnet when electricity flows through it.

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

When the MCU tests the instantaneous temperature and decides that the Air conditioner or Fan needs to be switched on, its PIN B2 or B3 connected to the Fan and AC respectively are made high. This high is equivalent to about a 2.5 to 5 volt output signal depending on its design. The output signal is passed through a base resistor into the base of the NPN transistor which has the ability to exert control over a much larger flow of electrons through the collector due to a relatively small flow of electrons sent through the base of the transistor. To trigger this large flow, a signal of not less than 0.7 volt has to be passed through the base of the transistor with a small resistor attached to it (the base) to make sure that the transistor is driven to saturation.
Considering the fact that the emitter terminal has been grounded, as soon as the required base signal is passed, the collector grounds the terminal of the relay connected to it , and then, the 12 volt relay triggers on the Fan or AC by completing its own circuit connection.

5. LCD – 'el-cee-dee'

The Liquid Crystal Display (LCD) LM044L is used to display the metre reading. This LCD has 4 visible display rows.

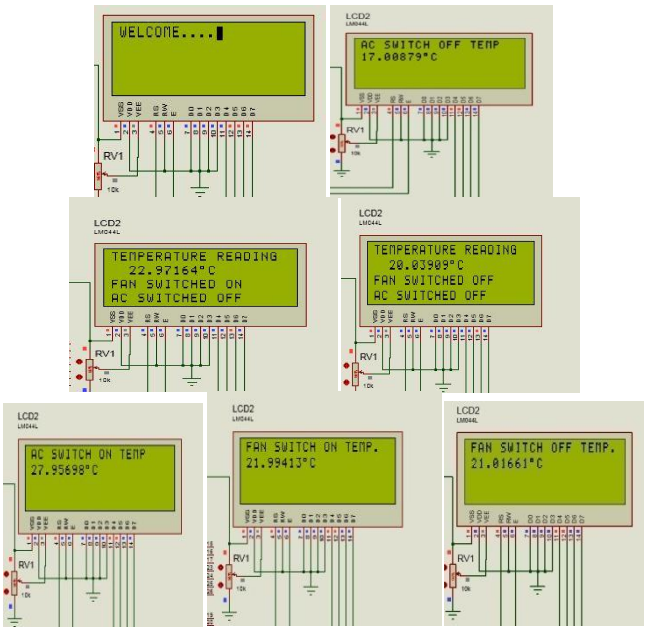

Fig. 5. Figure showing the various LCD Displays

V. DEVICE OPERTATIONS AND DIAGRAMATIC VIEW

1. *How to start Up*

i. Switch on Controller Device
ii. Set fan switch on temperature by turning the potentiometer RV2 (Maximum of 100^{o}C)
iii. Press the 'OK button'
iv. Set fan switch off temperature by turning the potentiometer RV2 (Minimum of 0^{o}C)
v. Press the 'OK button'
vi. Set AC switch on temperature by turning the potentiometer RV2 (Maximum of 100^{o}C)
vii. Press the 'OK button'
viii. Set AC switch off temperature by turning the potentiometer RV2 (Minimum of 0^{o}C)
ix. Press the 'OK button'

2. How to Change Temperature Settings

i. Press the Trigger Set-up display button
ii. Repeat the How to stat up procedure above

3. How to Reset System

i. Press the MCU Reset button

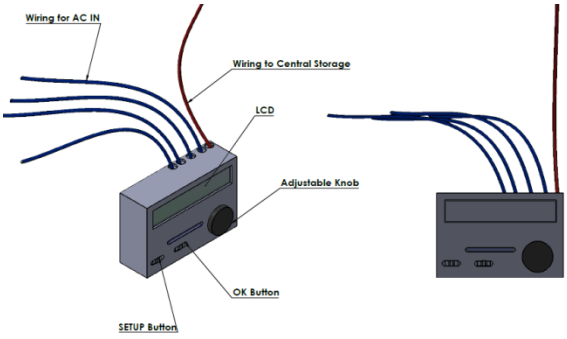

Fig. 6. Figure showing the AUTOCAD design of the cased device

For the temperature regulated appliance control in the intelligent building, this device can be placed at a strategic location in the building and thereafter, wired directly to the Fans and Air conditioning unit supply switches in the building. At operation, this system will automatically trigger ON or OFF the

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

cooling systems based on the inputted/set parameters.

VI. FIRMWARE CODE

Below is the full firmware code for the PIC18F4550 Microcontroller. The compiler used is the MikroC Pro Compiler, engaging the C programming language.

Code 1: Firmware Code for the MCUOperation

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <float.h>
#define INT_RANGE 1000
#define DEC_RANGE 10

// LCD module connections
sbit LCD_RS at RD2_bit;
sbit LCD_EN at RD3_bit;
sbit LCD_D4 at RD4_bit;
sbit LCD_D5 at RD5_bit;
sbit LCD_D6 at RD6_bit;
sbit LCD_D7 at RD7_bit;
sbit LCD_RS_Direction at TRISD2_bit;
sbit LCD_EN_Direction at TRISD3_bit;
sbit LCD_D4_Direction at TRISD4_bit;
sbit LCD_D5_Direction at TRISD5_bit;
sbit LCD_D6_Direction at TRISD6_bit;
sbit LCD_D7_Direction at TRISD7_bit;
// End LCD module connections

//Variable Declaration
unsigned int temperature, fanDecision, ACDecision; //On Temp. ADC Value read
unsigned int fanoffDecision, ACoffDecision; //Off Temp. ADC Value read
unsigned int tempset=0, k=0, setupFlag=1; //For flags and for loops
float floatTemperature, floatfanDecision, floatACDecision;/* On Temp. ADC
converted to actual values */
float floatfanoffDecision, floatACoffDecision;/*off Temp. ADC converted
to actual values */
float checkValue=0, oldcheckValue=0; //check values for fan switch on
float accheckValue=0, acoldcheckValue=0;//check values for Air Conditioner on
float checkoffValue=0, oldcheckoffValue=0; //check values for fan switch off
float accheckoffValue=0, acoldcheckoffValue=0;//check off values for AC
float fanswitchTemp=0, acswitchTemp=0;//Selected AC and FAN switch on temp.
float fanswitchoffTemp=0, acswitchoffTemp=0;//Selected AC & FAN switch off temp.
char *temperatureTXT[15], *ACsettemperatureTXT[15], *FansettemperatureTXT[15];

//Interrupt to activate Set_Temperature Set-up mode
void interrupt(){
if(INT0IF_bit) { //If INTO interrupt has happened
INT0IF_bit = 0;
setupFlag=1;
}
}

//Functions for different display states on the LCD
void Basic_Display(int type){
if(type==0){
LCD_Cmd(_LCD_CLEAR);
LCD_Cmd(_LCD_CURSOR_OFF);
}
else if(type==1){
LCD_Cmd(_LCD_CLEAR);
LCD_Cmd(_LCD_CURSOR_OFF);
LCD_Out(1,1,"FAN SWITCH ON TEMP.");
}
else if(type==2){
LCD_Cmd(_LCD_CLEAR);
LCD_Cmd(_LCD_CURSOR_OFF);
LCD_Out(1,1,"AC SWITCH ON TEMP");
}
```



```
else if(type==3){
LCD_Cmd(_LCD_CLEAR);
LCD_Cmd(_LCD_CURSOR_OFF);
LCD_Out(1,1,"TEMPERATURE READING");
}
else if(type==4){
LCD_Cmd(_LCD_CLEAR);
LCD_Cmd(_LCD_CURSOR_OFF);
LCD_Out(1,1,"FAN SWITCH OFF TEMP.");
}
else if(type==5){
LCD_Cmd(_LCD_CLEAR);
LCD_Cmd(_LCD_CURSOR_OFF);
LCD_Out(1,1,"AC SWITCH OFF TEMP");
}
}

//Get Current Temperature Reading
void Get_Temperature(){
temperature = ADC_Read(2); //Read Voltage output from thermometer
floatTemperature = (float) temperature/204.6; //Convert to a scale of 0 to 5v
```

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

```
84: floatTemperature = floatTemperature*100; //To get the actual temperature
85: }
86:
87: //Display Current Theremometer Temperature Reading
88: void Display_Temperature(){
89: FloatToStr(floatTemperature, temperatureTXT); //Convert temperature to String
90: LCD_Out(2,3,temperatureTXT); //Display Temperature
91: Lcd_Chr_Cp(223); // different LCD displays have different char code for degree
92: Lcd_Chr_Cp('C');
93: }
94:
95: //Used while trying to set fan switch on temperature
96: void Fan_SetSwitchOnTemperature_Checker(){
97: fanDecision = ADC_Read(3); //Read Input from potentiometer
98: floatfanDecision = fanDecision/10.23; //Convert to 0 - 100 Volts range
99: checkValue=floatfanDecision;
100: }
101: //Used while trying to set fan switch off temperature
102: void Fan_SetSwitchOffTemperature_Checker(){
103: fanoffDecision = ADC_Read(3); //Read Input from potentiometer
104: floatfanoffDecision = fanoffDecision/10.23; //Convert to 0 - 100 Volts range
105: checkoffValue=floatfanoffDecision;
106: }
107: //Used while trying to set AC switch on temperature
108: void AC_SetSwitchOnTemperature_Checker() {
109: ACDecision = ADC_Read(3); //Read Input from potentiometer
110: floatACDecision = ACDecision/10.23; //Convert to 0 - 100 Volts range
111: accheckValue=floatACDecision;
112: }
113:
114: //Used while trying to set AC switch off temperature
115: void AC_SetSwitchOffTemperature_Checker() {
116: ACoffDecision = ADC_Read(3); //Read Input from potentiometer
117: floatACoffDecision = ACoffDecision/10.23; //Convert to 0 - 100 Volts range
e
118: accheckoffValue=floatACoffDecision;
119: }
120: //Set Desired FAN Switch On Temperature
121: void Set_Fan_SwitchON_Temperature(){
122: while(tempset==0){
123: Fan_SetSwitchOnTemperature_Checker();
```

2/6 mikroC PRO for PIC by mikroElektronika
Temperature Controlled Appliances.c HEMS TEMPERATURE CONTROLLED AC & FAN FIRMWARE CODE

```
124: if(checkValue!=oldCheckValue){
125: FloatToStr(floatfanDecision,FansettemperatureTXT);
126: LCD_Out(2,1, FansettemperatureTXT);
127: Lcd_Chr_Cp(223); // different LCD displays have different char code for degree
128: Lcd_Chr_Cp('C');
129: oldCheckValue = checkValue;
130: }
131:
132: if(PORTB.F1==1){
133: tempset=1; //To Conclude AC & FAN Switch on Temperature settings (incase)
134: Basic_Display(0);
135: fanswitchTemp = checkValue;
136: LCD_Out_Cp("SWITCH ON TEMPERATURE FOR FAN SUCCESSFULLY SET");
137: Delay_ms(500);
138: Basic_Display(0);
139: PORTB.F1=0;
140: Delay_ms(500);
141: }
142: }
143: }
144: //Set Desired FAN Switch Off Temperature
145: void Set_Fan_SwitchOFF_Temperature(){
146: while(tempset==0){
147: Fan_SetSwitchOffTemperature_Checker();
148: if(checkoffValue!=oldCheckoffValue){
149: FloatToStr(floatfanoffDecision,FansettemperatureTXT);
150: LCD_Out(2,1, FansettemperatureTXT);
151: Lcd_Chr_Cp(223); // different LCD displays have different char code for degree
152: Lcd_Chr_Cp('C');
153: oldCheckoffValue = checkoffValue;
154: }
155:
156: if(PORTB.F1==1){
157: tempset=1; //To Conclude AC & FAN Switch on Temperature settings (incase)
158: Basic_Display(0);
159: fanswitchoffTemp = checkoffValue;
160: LCD_Out_Cp("SWITCH OFF TEMPERATURE FOR FAN SUCCESSFULLY SET");
161: Delay_ms(500);
162: Basic_Display(0);
163: PORTB.F1=0;
164: Delay_ms(500);
165: }
166: }
167: }
168:
169:
170: //Set Desired AC Switch On Temperature
171: void Set_AC_SwitchON_Temperature(){
172: while(tempset==0){
```

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

```
173: AC_SetSwitchOnTemperature_Checker();
174: if(accheckValue!=acoldCheckValue){
175: FloatToStr(floatACDecision,ACsettemperatureTXT);
176: LCD_Out(2,1, ACsettemperatureTXT);
177: Lcd_Chr_Cp(223); // different LCD displays have
different char code for degree
178: Lcd_Chr_Cp('C');
179: acoldCheckValue = accheckValue;
180: }
181:
182: if(PORTB.F1==1){
183: tempset=1;
184: Basic_Display(0);
185: acswitchTemp = accheckValue;
```

3/6 mikroC PRO for PIC by mikroElektronika
Temperature Controlled Appliances.c HEMS TEMPERATURE CONTROLLED AC & FAN FIRMWARE CODE

```
186: LCD_Out_Cp("SWITCH ON TEMPERATURE FOR
AC SUCCESSFULLY SET");
187: Delay_ms(500);
188: Basic_Display(0);
189: PORTB.F1=0;
190: Delay_ms(500);
191: }
192: }
193:
194:
195: }
196: //Set Desired AC Switch Off Temperature
197: void Set_AC_SwitchOFF_Temperature(){
198: while(tempset==0){
199: AC_SetSwitchOffTemperature_Checker();
200: if(accheckoffValue!=acoldCheckoffValue){
201: FloatToStr(floatACoffDecision,ACsettemperatureTXT);
202: LCD_Out(2,1, ACsettemperatureTXT);
203: Lcd_Chr_Cp(223); // different LCD displays have
different char code for degree
204: Lcd_Chr_Cp('C');
205: acoldCheckoffValue = accheckoffValue;
206: }
207:
208: if(PORTB.F1==1){
209: tempset=1;
210: Basic_Display(0);
211: acswitchoffTemp = accheckoffValue;
212: LCD_Out_Cp("SWITCH OFF TEMPERATURE FOR
AC SUCCESSFULLY SET");
213: Delay_ms(500);
214: Basic_Display(0);
215: PORTB.F1=0;
216: Delay_ms(500);
217: }
218: }
219: }
220:
221: //Check to know if current temperature requires Fan to
be switched on or off
222: void Fan_SwitchONOFF_Decider(){
223: if(floatTemperature>=fanswitchTemp){
224: PORTB.F2=1;
225: LCD_Out(3,1,"FAN SWITCHED ON ");
226: }
227:
228: else if(floatTemperature<=fanswitchoffTemp){
229: PORTB.F2=0;
230: LCD_Out(3,1,"FAN SWITCHED OFF ");
231: }
232: }
233:
234:
235: //Check to know if current temperature requires Fan to
be switched on or off
236: void AC_SwitchONOFF_Decider(){
237: if(floatTemperature>=acswitchTemp){
238: PORTB.F3=1;
239: LCD_Out(4,1,"AC SWITCHED ON ");
240: }
241: else if (floatTemperature<=acswitchoffTemp){
242: PORTB.F3=0;
243: LCD_Out(4,1,"AC SWITCHED OFF");
244: }
245: }
246:
247: //Bring up the fan/ac switch on temperature set up
screen
```

4/6 mikroC PRO for PIC by mikroElektronika
Temperature Controlled Appliances.c HEMS TEMPERATURE CONTROLLED AC & FAN FIRMWARE CODE

```
248: void setup(){
249: tempset=0;
250:
251: /*Disperse the check and old check values to make sure
they aren't == to each
252: other @ the Set_Fan/AC_SwitchON_Temperature()
reading stage */
253: checkValue=oldcheckValue-1,
oldcheckValue=checkValue+4;
254: accheckValue=acoldcheckValue-1,
acoldcheckValue=accheckValue+4;
255: fanswitchTemp=25, acswitchTemp=28;
256:
257: checkoffValue=oldcheckoffValue-1,
oldcheckoffValue=checkoffValue+4;
258: accheckoffValue=acoldcheckoffValue-1,
acoldcheckoffValue=accheckoffValue+4;
259: fanswitchoffTemp=20, acswitchoffTemp=18;
260:
261: Basic_Display(0);
262: Basic_Display(1);
263: Set_Fan_SwitchON_Temperature();
```

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 12, December 2015
Available at http://internationaljournalofresearch.org

```
264: tempset=0;
265: Basic_Display(4);
266: Set_Fan_SwitchOFF_Temperature();
267: tempset=0;
268: Basic_Display(2);
269: Set_AC_SwitchON_Temperature();
270: tempset=0;
271: Basic_Display(5);
272: Set_AC_SwitchOFF_Temperature();
273: Basic_Display(3);
274: setUpFlag=0;
275: }
276:
277: void main() {
278: TRISB = 0b00000011;//set OIN 0 and 1 of PortB as Input while the rest as output
279: PORTB = 0x00;
280: LATB = 0x00;
281:
282: CMCON = 0x07; // Turn off comparators
283: CVRCON = 0x00;
284: //Configure AN0:4 Pins as Analogue, but AN12,AN11,AN10,AN9,AN8 as digital
285: ADCON1 = 0b00000111;
286: ADCON2 = 0x0F;
287:
288: INTEDG0_bit = 1;//Interrupt happen on the rising edge of the signal into INT0
289: INT0IE_bit = 1; //Enable interrupt which depends on the stae of the INT pin
290: INT0IF_bit = 0; //Set INT0 interrupt flag initially to 0
291: PEIE_bit = 1; //Enable pheripherial Interrupt
292: GIE_bit = 1; //Enable lobal Interrupt
293:
294: Lcd_Init(); // Initialize LCD
295:
296: //Animate System Loading Pattern
297: LCD_Out(1,1,"WELCOME");
298: for(k=0; k<5; k++){
299: LCD_Out_Cp(".");
300: delay_ms(100);
301: }
302: do{
303:
304: if(setupFlag==1){ //Check to know if the set screen has been triggered
305: setup(); //Call Set up screen function
306: }
307: get_Temperature();
308: Display_Temperature(); //Call display temperature function
309: Fan_SwitchONOFF_Decider(); //Check to know if to switch on Fan
```

5/6 mikroC PRO for PIC by mikroElektronika

Temperature Controlled Appliances.c HEMS TEMPERATURE CONTROLLED AC & FAN FIRMWARE CODE

```
310: AC_SwitchONOFF_Decider(); //Check to know if to switch on AC
311: }
312:
313: while(1);
314: }
```

End of Code

VII. CONCLUSION

Embedded systems is indeed the future of our technology oriented world. Details from this paper elucidates a cost effective design of an automated temperature controller module geared toward offering effective, efficient, convenient and most of all, an energy conservation mechanism to building owners who wish to manage the switching of their temperature dependent devices in the home. This thereby draws the building another step closer towards being fully automated and intelligently controlled.

REFERENCES

[1] Adewale A. A. et al, "Design and Development of a Microcontroller Based Automatic Switch for Home Appliances",International Journal of Engineering Science Invention, Volume 2 Issue 10 ǁ October. 2013 ǁ PP.24-31

[2] Poonam Lakra1, Dr. R. P. Gupta2, "Microcontroller Based Automatic Control Home Appliances", International Journal of Innovative Research in Science,

[3] Defining Todays Inteligent Buldings, http://www.commscope.com/Blog/Defining-Todays-Intelligent-Building/, Retreived 2015

[4] Building Automation, https://en.wikipedia.org/wiki/Building_automation, retreived 2015

[5] Intelligent buildings design and building management systems, http://www.businessballs.com/intelligentbuildingsdesign.htm, Retreived, 2015

[6] LM35 Temperature Sensor, http://www.instructables.com/id/LM35-Temperature-Sensor/, Retreived 2015

[7] PIC18F4550-I/P - PIC18F4550 Flash 40-pin 32kB Microcontroller with USB - Buy PIC18F4550-I/P, http://www.futurlec.com/Microchip/PIC18F4550.shtml, Retreived 2015