DEVELOPMENT OF A DIGITAL BASED TEMPERATURE CONTROL SYSTEM WITH DATA LOGGING FACILITY

***E.Obi, I. Yau, I.K Musa and A. Mohammed**

Department of Communications Engineering, Ahmadu Bello University, Zaria.

elvisobi@gmail.com, ishaaqyau@gmail.com, ismkmusa1@gmail.com

ABSTRACT

This paper presents the development of an improved Arduino based microcontroller temperature control system. The developed system has a recording mechanism, this provides information about the temperature behavior of the device to be cooled. This information can be used to determine whether the device is beginning to deteriorate or develop fault, as such prevention of imminent damage or catastrophic failure can be avoided. The control and monitoring of the cooling fan is designed such that the speed of the cooling fan varies with the temperature of the device to be cooled. The results showed that the fan is switched on at a simulated temperature value of 98 ^oC using the potentiometer and runs at 1373 rpm and the device being cooled is also switched off at a simulated maximum temperature of 392 ^oC or above.

Keywords *- Arduino Development Board, Analog to Digital Conversion, microcontroller, SD – Card, Pulse Width Modulation.*

1.0 INTRODUCTION

The fundamental parameter often used to measure environmental hazard of industrial systems like furnace engines and ovens is the temperature. This might be expected since most physical, electronic, chemical, mechanical and biological systems are affected by temperature and perform optimally within designed specified temperature ranges (Hsiao, 2011). The cooling fan is normally used as a means of cooling industrial and electronic devices. Fans can be controlled manually, but any slight change in the temperature will not give a change in the fan speed (Ghana & Ram, 2013). Also, running the fan at a fixed temperature is not cost effective and inefficient in energy management. Therefore, an automatic temperature control system is needed for the monitoring and controlling purpose that works according to the changes in device temperatures (Jagale, 2015). The automatic control of the fan speed with respect to the working temperature of a device reduces its power consumption, increases efficiency, reliability and effectiveness of the device being cooled (Veram et al., 2014). Various works have been done on automatic temperature cooling fan, some of which are: Pravesh (2011) developed a temperature controlled fan using an LM35 temperature sensor and analog digtal converter which converted the output of the sensor to digital values that are feed into a microcontroller 8051. A program was integrated into the microcontroller to enable the selection of unique values of temperature from a look up table based on different ranges of sensed

temperature values. A seven segment display was use to show the simulated temperature sensed and speed of the cooling fan. Jagdale (2015) developed an automatic direct current fan controller using thermistor as the temperature sensor. An operational amplifier IC741 was used as a voltage comparator which compared the voltage between its two inputs, as the temperature of the surrounding increased, the output voltage of the comparator increases causing increase in fan speed. Soren and Gupta (2016) developed a program for an Arduino microcontroller to control the speed of a direct current (DC) fan while using an LM35 as the temperature sensor. When the temperature is greater than the threshold, the microcontroller gives output to the motor driver for starting the DC fan. Now, the previous works reviewed lack data recording mechanism during the operation of the cooling system, this result to non-availability of information that can be used to determine if the device to be cooled is beginning to deteriorate or develop fault, as such prevention of imminent damage or catastrophic failure is difficult. This work is set out to address the aforementioned limitation. The regulation, control and monitoring of the cooling fan is done with the aid of an Arduino development board using the Atmeg2560 microcontroller in which the speed of the fan is controlled at various temperatures detected by the LM35 sensor. The temperature sensed is due to the heat produced by the device which is being cool. At a temperature that goes beyond the normal working temperature of the device being cooled, the microcontroller switches on the cooling fan to regulate the temperature of the device and also raise an alerts via a buzzer. The fan also is turned on to reduce the heat produced before going off. The microcontroller also sends at various time intervals the temperature readings to an SD card module which serves as important data in analyzing the reliability of the device being cooled as well for corrective or preventive maintenance.

2.0 DESIGN METHODOLOGY

2.1 Introduction

This section gives the detailed procedure followed for the development of the system which entails hardware design and software development/programming. The block diagram of the proposed system with various input and output devices to the Arduino board is shown in Figure 1.

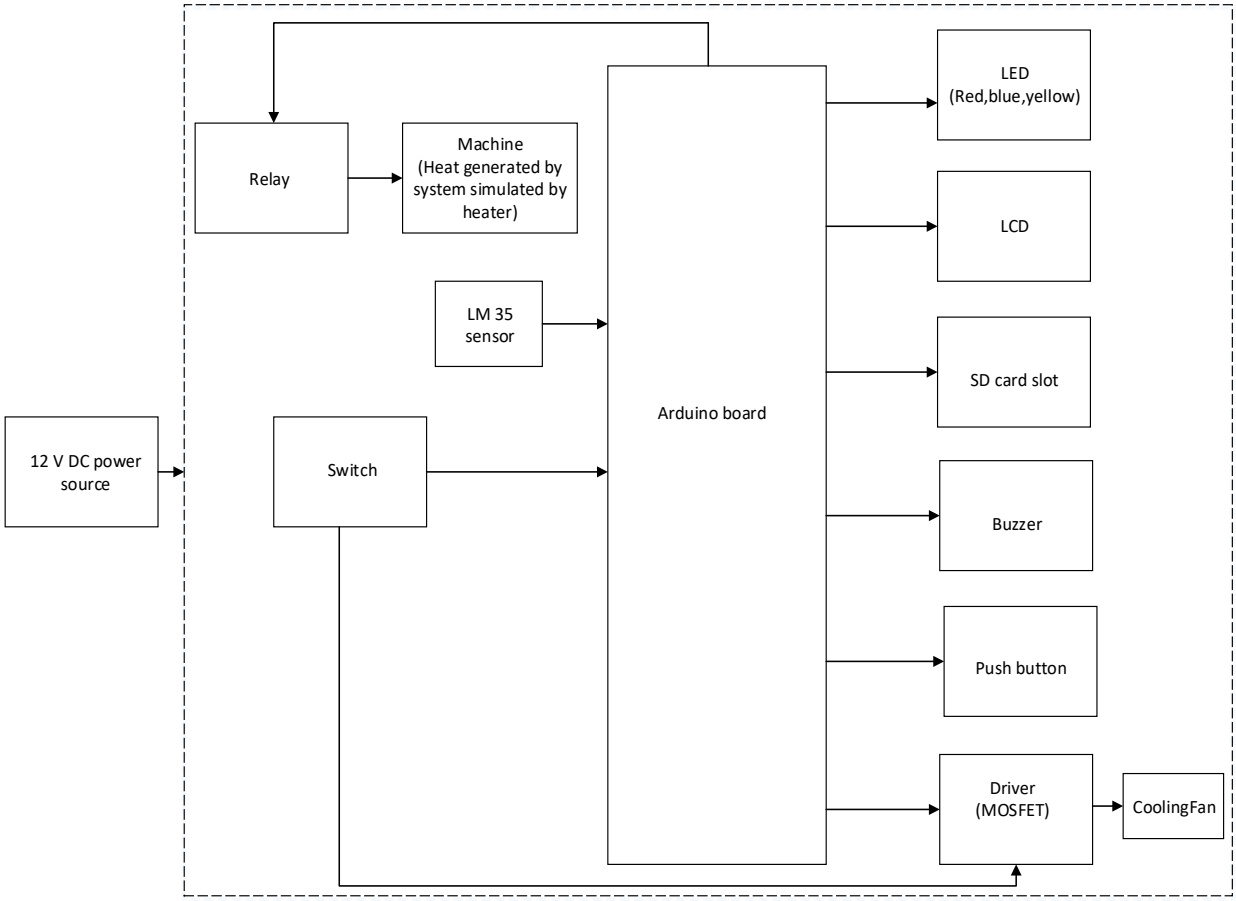

Figure 1: Block Diagram of the proposed system

2.2 Interfacing the LM35 Temperature Sensor with the Arduino Board

Figure 2 shows Connections between the Arduino Board and the LM35 sensor.

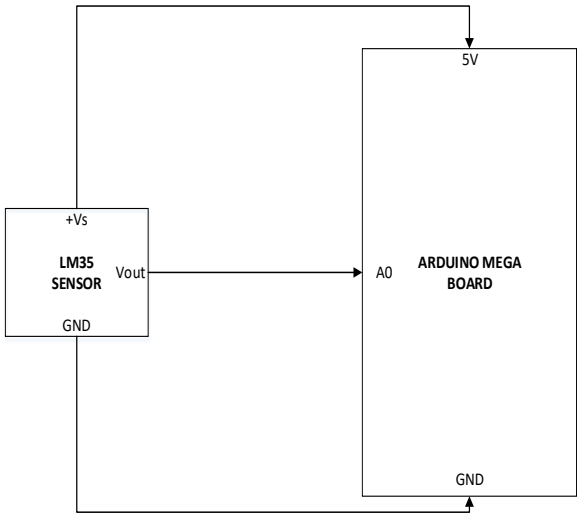

Figure 2: Connections between the Arduino Board and the LM35 sensor

The LM35 sensor has three pins, GND (-), +Vs (+), Vout. The GND pin is connected with GND on Arduino. The +Vs pin is connected to +5v on the Arduino board and the Vout pin is connected to Analog 0 /A0 (or any analogue pin) on Arduino board.

The LM35 sensor being a transducer converts temperature (physical signal) to analogue voltage (electrical signal). The output voltage of an LM35 sensor (Vout) per degree rise is 10mV. The analogue signal from the sensor is converted to digital by the Analogue to Digital converter (ADC) in the microcontroller. This is done through the following steps (Nate, 2012).

i. The analogue output signal from the LM35 ranges from 0 to 5V, since the supply is 5V.

ii. The ADC of the controller samples the signal into a 10bit digital signal (with a 2^{10} =1024 distinct values ranging from 0-1023).

iii. From here the output voltage can be converted to centigrade by programming the controller with equation 1

$$\frac{\text{Re}\,solution\ of\ ADC}{System\ Voltage} = \frac{ADC\ \ \text{Re}\,ading}{Ana\log\ Voltage} \qquad (1)$$

Where: Resolution of the ADC is 1023

System voltage is 5000mV

ADC reading ranges from 0-1023

Analogue voltage measured ranges from 0-5000mV

Since each 10mV increase in temperature represents 1°C raise in temperature, therefore:

$$Temperature\ (^{o}C) = \frac{Ana\log\ Voltage}{10} \qquad (2)$$

2.3 Interfacing the LCD with the Arduino Board

Before connecting the LCD screen to the Arduino board, pin header strips are soldered to the 14 (or 16) pin connectors of the LCD screen. In this work interfacing of the LCD screen to the board is done using the following configuration.

The Register select (RS) pin is connected to digital pin 26 of the Arduino board. The Enable pin is connected to digital pin 32. The LCD data pins D4, D5, D6 are connected to pin 33, 34, 35 and 36 of the board. The Read Write (RW) LCD pin is grounded together with the V_{SS} and K pin while the V_{DD} and A pin are connected to +5V from the Arduino Board. A 10k potentiometer is connected to both the +5V and GND of the Arduino, with its output joined to the LCD screen's voltage output pin.

2.4 Interfacing the SD – Card Module with the Arduino Board

The micro- SD Card Module is a simple solution for transferring data to and from a standard SD card. The pin out is directly compatible with Arduino, but can also be used with other microcontrollers. It allows for mass storage and data logging. The connections are done in the following manner. Pin 50 of the board is interconnected with the Master in Slave Out (MISO) pin on the module while Pin 51 to Master Out Slave In (MOSI) Pin 52 to SCK (System Clock) and Pin 53 to SS Save Select. Supply is taking from the Arduino board and the module is grounded. The configuration is shown in Figure 3.

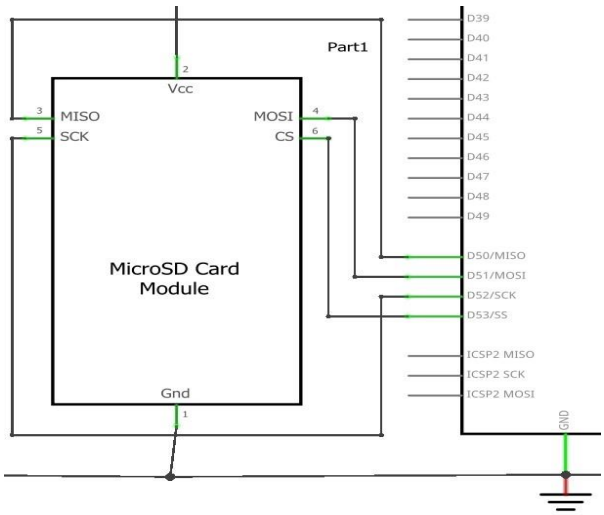

Figure 3: Connections of the SD – Card to the Arduino Board.

The SD card stores temperature values in this paper.

2.5 Interfacing the Buzzer and LEDs

The LEDs are combined in different configurations to display the various

temperature levels and working modes of the fan. An LED can only take 5-20mA current and around 1.7- 3.8V depending on it brightness. The supplies 5V and 20mA to it I/O pin is such as the LED require a current-limiting resistor whose value is calculated as (Boxall, 2013):

$$R = \frac{V_s - V_f}{I} \quad (3)$$

Where R is the current limiting resistor, V_s is the supply voltage, V_f is LED forward voltage drop and I is the current required by for the LED. Therefore the value of the various LED indicators are calculated as follows:

$$R_{RED} = \frac{(5-2)V}{15mA} = 200\Omega$$

$$R_{RED} = 220\Omega$$

$$R_{YELLOW} = \frac{(5-2.4)}{15mA} = 173.33\Omega$$

The resistant value is increased to one of the standard resistor value and while keeping the current within the stipulated range

$$R_{YELLOW} = 200\Omega$$

Similarly,

$$R_{BLUE} = \frac{(5-3.2)V}{15mA} = 120\Omega$$

$$R_{BLUE} = 150\Omega$$

When connecting the LED, the cathode is connected to ground and the anode is combined in series with a resistor and connected to the I/O pin of the Arduino.

The buzzer (Alerting circuit) is connected directly to Arduino with the positive pin to 5V and the negative pin to ground.

2.6 Design of the Switching Circuit

In order to switch the cooling DC fan on and off, a 12V relay is employed. This is used because the whole system is powered by a 12V battery. The connections are explained as seen in Figure 4.

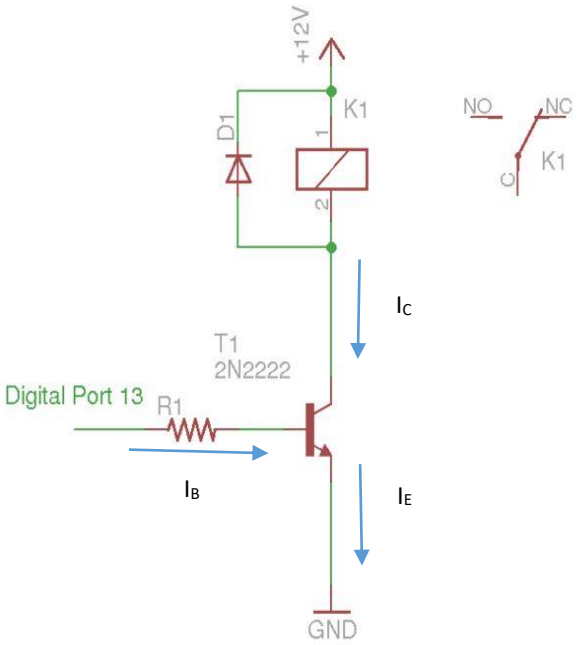

Figure 4: Transistor connected with a 12V relay (Hamid, 1999)

In other to successfully switch the relay on and off a BJT (2NN222) is used to amplify the current from the board as it is not sufficient on its own.

The base of the transistor is connected to the Arduino board and the collector is connected to the 2 pin of the relay. The first pin is connected to a 12V supply. A freewheeling diode is connected in parallel with the relay to prevent the flow of current in the reverse direction. A resistor (R1) is connected to the base is calculated by the following process (Hamid, 1999).

i. Firstly, the resistance of the coil R_c is measured. The value gotten was 0.421k Ω

ii. To get the collector current we use the equation

$$I = \frac{V}{R} \quad (4)$$

$$I_c = \frac{12}{4210} = 2.85\,mA$$

Therefore, the collector current must be equal to or greater than 2.85 *mA* to switch the relay. The transistor has a gain of 75 (2N2222 DATA SHEET, 1997). But the gain of a transistor is given as:

$$h_{fe} = \frac{I_c}{I_b} \quad (5)$$

Therefore,

$$I_b = \frac{2.85}{75} = 0.38\,mA$$

The current that is drawn from the Arduino should be as little as possible.

To get the resistor at the base we use equation (4)

$$R_b = \frac{5}{0.00038} = 1315\,\Omega$$

2.7 Interfacing the Fan Motor with the Arduino Board

The fan is connected to the pulse width modulation (PWM) pins of Arduino. The PWM pins help vary the voltage supplied to the fan and hence the speed. The I/O pins of Arduino supply either a HIGH (5V) or LOW (0V). A diagram of the PWM wave is shown in Figure 5

Figure 5: A pulse width modulated wave (Recktenwald, 2011)

A PWM train operates at 500 Hz. The duty cycle (width) of the PWM starts from 0 to 255 (Recktenwald, 2011). The microcontroller converts a certain voltage to the pulse train using the flowing equation

$$V_{eff} = V_S \frac{\tau_e}{\tau_o} \quad (6)$$

Where V_{eff} is the voltage level required, τ_e is the pulse width required to produce the voltage level τ_o is the total pulse width which is 255 and V_s is the supply voltage from the I/O pin which is 5V

The fan speed N_S in rpm is calculated using

$$\frac{N_{s2}}{N_{S1}} = \frac{PWM_2}{PWM_1} \quad (7)$$

Where N_{S2} is the maximum speed of the DC fan with value of 3500 rpm corresponding to the maximum PWM width of 255, N_{S1} is the fan speed corresponding to PWM_2. The fan is operated below the maximum speed to prevent damage and this done by choosing arbitrary values of PWM_2 as 0, 100, 150, 200 and 250 to represent very low, low, medium, high and very high fan speed respectively.

The DC fan being 12V cannot be effectively switched by the Arduino as such a depletion MOSFET (IRFZ44N) is employed. The MOSFET is an N channel with a gate to source voltage (V_{GS}) of 4V (IRFZ44N DATA SHEET, 2001).

The drain of the MOSFET is connected to the motor of the fan. The source is grounded and the gate is connected to the board at pin 50. A pull down resistor is attached to differentiate a high from low signal. A freewheeling diode is

also attached in parallel to the motor to pass excess current when the motor changes speed or is turned off.

2.8 General Circuit Diagram and Operation Principle

The complete circuit arrangement of the proposed automatic temperature controlled system is as shown in Figure 6.

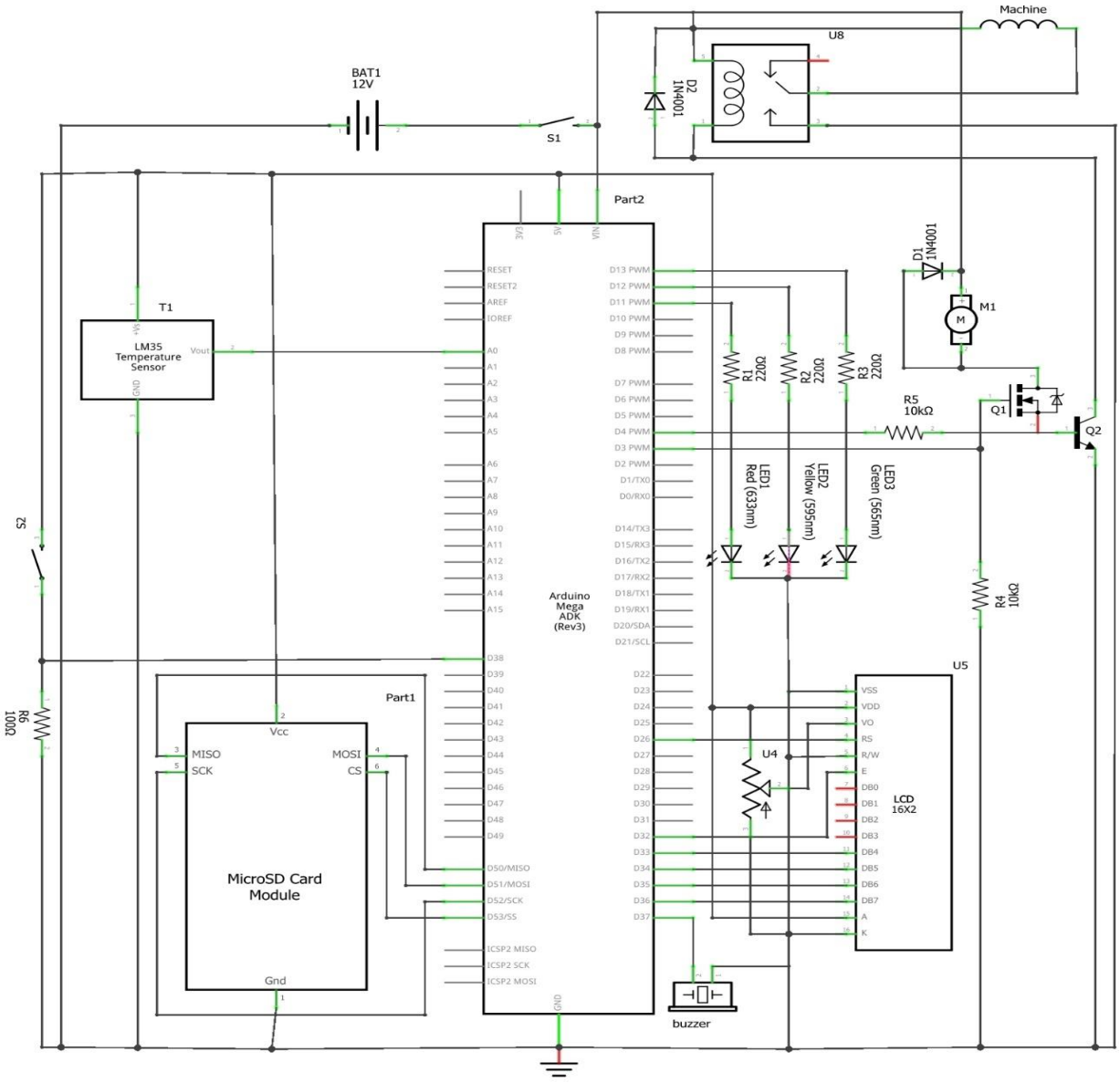

Figure 6: Ciruit Diagram of Proposed System

Referring to Figure 6 which shows the complete circuit arrangement of an automatic temperature control system, the system is powered by a 12V battery. Both the fan motor and the Arduino board draw their power from this source. The push button (S2), Micro SD card module and LCD are all powered by 5V gotten from the Arduino board.

In this system temperature sensor LM35 (T1) is used to detect temperature due to the heat produced by the device to be being cool, and calibrate it to voltage values. This voltage being in analogue form is fed to the Arduino board. An analogue to digital converter converts the signal to digital values. According to the

developed program, it processes the digital signal and forms a particular temperature level for a particular voltage level. The temperature level is sent to the SD card module to store on the MicroSD card every.

The LEDs (LED1, LED2 and LED3) light up in different configurations as explained in the process diagram from Green (LED3) when the temperature is below the threshold to Red (LED1) when the temperature goes above the maximum.

The microcontroller switches on the DC fan via the PWM pin of the board. The PWM pins have the ability to produce varied output voltages. An N-channel MOSFET (Q1) is used to drive the fan as its capacity is 12V. The speed at which the fan runs is determined by the voltage supplied to the gate of the MOSFET. If the temperature goes beyond the maximum value a signal is sent to the buzzer which produces and alarm, the device being cooled is switched off via the RELAY (U8) using the BJT Transistor (Q2) to amplify the current sent by the I/O pin of the board, thereby acting as an amplifier. A freewheeling diode (D2) is also connected across the relay to pass excess current from the relay coil when the device being cooled is suddenly switched off. The fan runs for a few minutes before going off, then a Red LED on the system turns on signifying the fan is powered down. A push button (S2) is used as a denouncer to reset the microcontroller so as to start the cooling process instead of shutting down the whole system as is in the case of switch (S1). Resistors R1, R2 and R3 are used to limit the current from the Arduino board to the LED to a normal functional value. R4 and R6 are used as pull down resistors to differentiate a high signal from a low one and finally R5 is used at the base of the BJT transistor to reduce the current at the base of the transistor to get an almost equal collector current which will reduce the heating effect in the relay coils. The potentiometer (U4) is used as contrast adjustment for the LCD. Data is saved in the SD card in a text file format (.txt). This text file is exported to excel for more analysis.

3. RESULTS AND ANALYSIS

In the temperature test, various graduations of temperature were found to represent each range at which the LM35 should operate putting into cognizance the temperature at which polystyrene melts. In the fan speed test the fan was powered at different PWM duty cycles to determine how fast it ran. This test involved heating of the LM35 at various tested temperature ranges and determining the speed at which it could be cooled by the fan. A potentiometer was used to simulate the heat being produced. From the program written there are five graduations. Very low, low, medium, high and very high. Each graduation level of the fan is at a certain speed when the temperature of the system is at a given level as shown in Table 1. The analogue values for the temperature and fan speed gotten from the test conducted are mapped to each level. The analogue voltage value is converted to temperature in degrees using equations (2) and the fan speed corresponding to the PWM width calculated using equation (7) as shown in Table 1

Table 1 Temperature and Speed Values of the Cooling System

Speed	Level	Analog Voltage (mV)	T (^{O}C)	Fan Speed (rpm)
Very Low	0	0-200	0-98	0
Low	1	201-400	98-196	1373
Medium	2	401-600	196-294	2059
High	3	601-800	294-392	2745
Very High	4	801-above	392 - above	3431

Fan speed and temperature test were carried out to validate the efficiency of the prototype under different simulated working temperature values. The temperature at different speed saves in the SD card at every 1 second for 1 minute. An extract of the saved data at the interval of 5 seconds is given in Appendix 1. The graph showing the variation of the speed level with time and the voltage/temperature against time is shown in Figure 7 and 8 respectively.

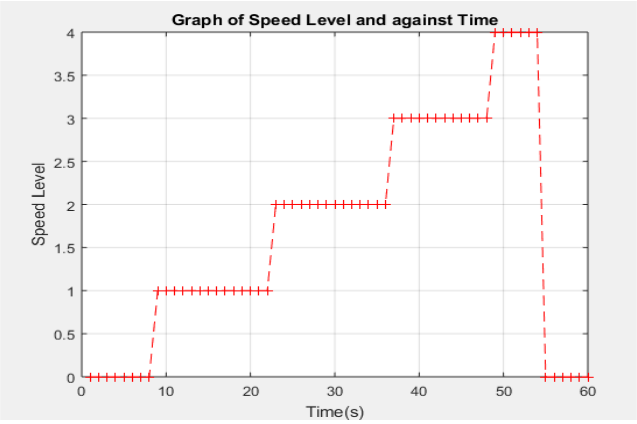

Figure 7: A graph of Speed Level with Time

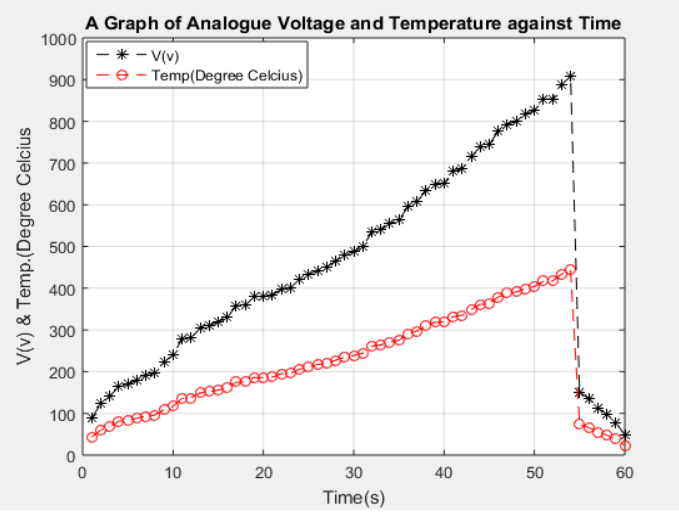

Figure 8: A graph of Voltage and Temperature with Time

The results showed that the fan is switched on at a simulated temperature value of 98^{o}C using the potentiometer and runs at 1373 rpm and the device being cooled is also switched off at a simulated maximum temperature of 392^{o}C or above. The maximum speed of the fan was 3431rpm before going off. The graph of the temperature and Fan speed against time is used to determine how effective the cooling system is and also the cause of a fault during corrective maintenance.

4. CONCLUSION

In this work, automatic 'Arduino based temperature controlled system' using LM35 temperature sensor was implemented. From the results obtained it can be seen that the Fan can easily cool and protect any industrial system (with each being given a pre-set temperature range value) from heat damage. It can also effectively ease maintenance processes with the data that is saved.

REFERENCES

22N2222 DATA SHEET. (1997). *Discrete Semiconductors*. Netherlands: Philips.

Boxall, J. (2013). *Adruino Workshop.* San Francisco: William Pollock.

Ghana, S. S., & Ram, A. G. (2013). *Temperature Controlled DC fan using microcontroller*.National Institute of Technology, Rourkela.

Hamid, B. (1999). *Connecting a 12V relay to Arduino*. Retrieved from Instructables : www.instructables.com, August 2017.

Hsiao, H. H. (2011, May 28). *Temperature sensor*. Retrieved from Sensor Workshop at ITP.

IRFZ44N DATA SHEET. (2001, March 1). *International Rectifier*, 8. Kansas, California, USA. Retrieved from www.irf.com

Jagale, P. M. (2015). *Automatic DC fan controller using thermistor*. Mumbai: Rajiv Gandhi Institute of Technology, Mumbai.

Kushagra. (2012). *LCD*. Retrieved from EngineersGarage: www.engineersgarge.com, June 2017.

Nate. (2012). *Analog to Digatal Conversion.* Retrieved from Sparkfun: www.learn.sparkfun.com: May 2017.

N-Channel MOSFET 60V 30A. (2014). Retrieved from Spark fun:

Pravesh, T. (2011). *Temperature Controlled Fan.* Birla Institute of Technilogy and Science, Pilanai-Hyderabad.(Unpublished)

Recktenwald, G. (2011). *Basic Pulse Width Modulation.* Retrieved from CECS: www.cecs.pdx.edu: May 2017.

Soren, S., & Gupta, R. (2016). *Temperature Controlled DC Fan using Microcontroller*. Department of Electrical Enginnering National Institute of Teechnology, Rourleka.

Veram, S., Dey, A., Subham, D., & Parijat, C. (2014). Automatic Temperature Controlled Fan Using Thermistor. *International Journal of Innovative Research in Technology & Science(IJIRTS)*, 3-5.

APPENDIX 1

Speed Level	**t(s)**	**Analogue V (v).**	**Celcius Temp.**
0	1	88	43
0	5	170	83
1	10	240	117
1	15	320	156
1	20	380	186
2	25	432	211
2	30	488	239
2	35	563	275
3	40	652	319
3	45	745	364
4	50	826	404
0	55	150	73
0	60	48	23