e-ISSN 2409-0360
Zimbabwej.sci.technol

Centralised Incubator Control System

[a] Busiso Mtunzi, [b] Malvern Chamisa, [c] Zedekia Madumbu, [d] Nyathi Reginald Gonye, [e] Fidelis Nhenga-Mugarisanwa

[a,b,c,d,e] Department of Electronic Engineering, National University of Science and Technology
P O Box AC939 Ascot Bulawayo,
Zimbabwe

[a] Email: busiso.mtunzi@nust.ac.zw

ABSTRACT

This paper presents an infant incubator system that protects and helps incubate a premature infant while in hospital. The device is economical, robust, and uses easily replaceable parts. It consists of a control mechanism that makes use of fans, a humidifier and a home heater rod to provide heat. The temperature and humidity were automatically controlled and maintained by a microcontroller. The incubator had a base designed to automatically weigh and log the infant's weight. A Visual Studio interface was used to provide incubator chamber data and to evaluate the performance of the design. The system could weigh the weight of the infant as well as control the chamber temperature at 36.5°C and humidity at 40%. The system was able to maintain these conditions within a period of 5 seconds.

Keywords: Premature; Child mortality; Incubator; Microcontroller; Humidity control

Received: 27.01.16 Accepted: 15.07.16

1. INTRODUCTION

Every year, over 4 million infants die within a month of birth. Of this number, 3.9 million belong to the developing world. Twenty five percent of the deaths were found to be due to complications of prematurity, most often heat and water loss (UN Inter-agency Group for Child Mortality Estimation, 2013). Premature infants lack muscle mass, which allows adults to shiver and produce heat when necessary. Furthermore, their immature skin allows excessive water loss from the body causing a considerable evaporative heat loss and a potentially fatal imbalance of salts and acids in the infant's system (WHO, 1998). Incubators help in keeping high, the air humidity and this helps in preventing too great a loss of heat from skin and through respiration (WHO, 1998).

In Zimbabwe, general hospitals use imported infant incubators which are difficult to maintain and the replacement parts are not readily available, hence a solution to this life saving equipment is needed. There is a need for an effective, low cost incubator made using locally available parts which can be easily accessed by the nurses and doctors.

2. DESIGN METHODOLOGY

The Centralised incubator control system consisting of nine functional blocks was designed. The block diagram illustrating the system was as shown in Figure 1.

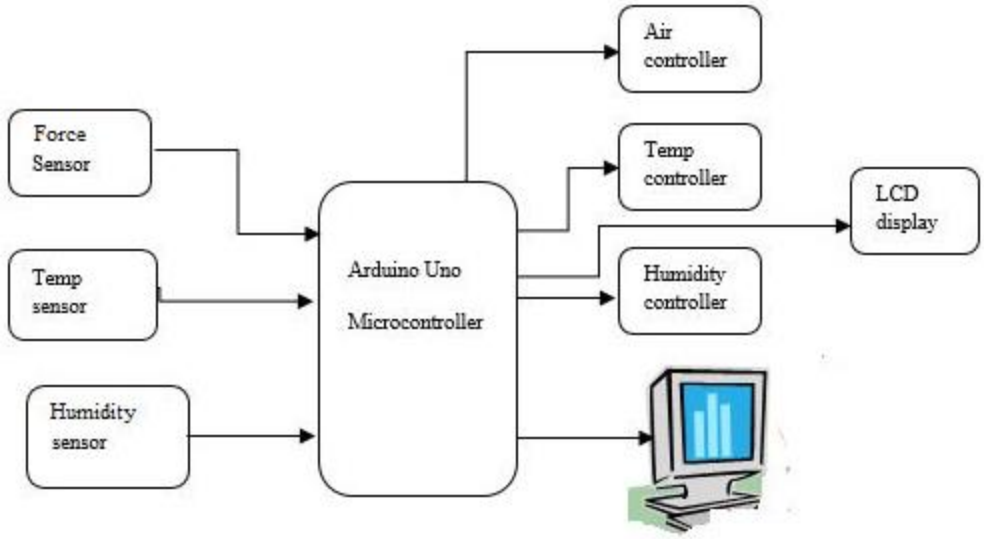

Figure 1. Block diagram of the Centralised incubator control system

As shown in Figure 1, the embedded system basically consisted of humidity sensor, force sensor and temperature sensor and the respective parameter controllers, microcontroller, LCD and a computer. The microcontroller's function was to read signals from the sensors, digitise as necessary and compare them to set points. Depending on whether a respective signal was above or below the set point, the microcontroller would turn on or off the respective actuator for control of humidity and temperature. The microcontroller also enabled display of temperature, humidity and weight on a LCD as well as sending of these values to a computer. The computer was used for display of these values on graphic user interface (GUI) and storage.

2.1 Centralised Incubator Control System

The detailed design of the Centralised Incubator Control system was as shown in Figure 2. The nine elements mentioned above which make up the system were as shown in Figure 2.

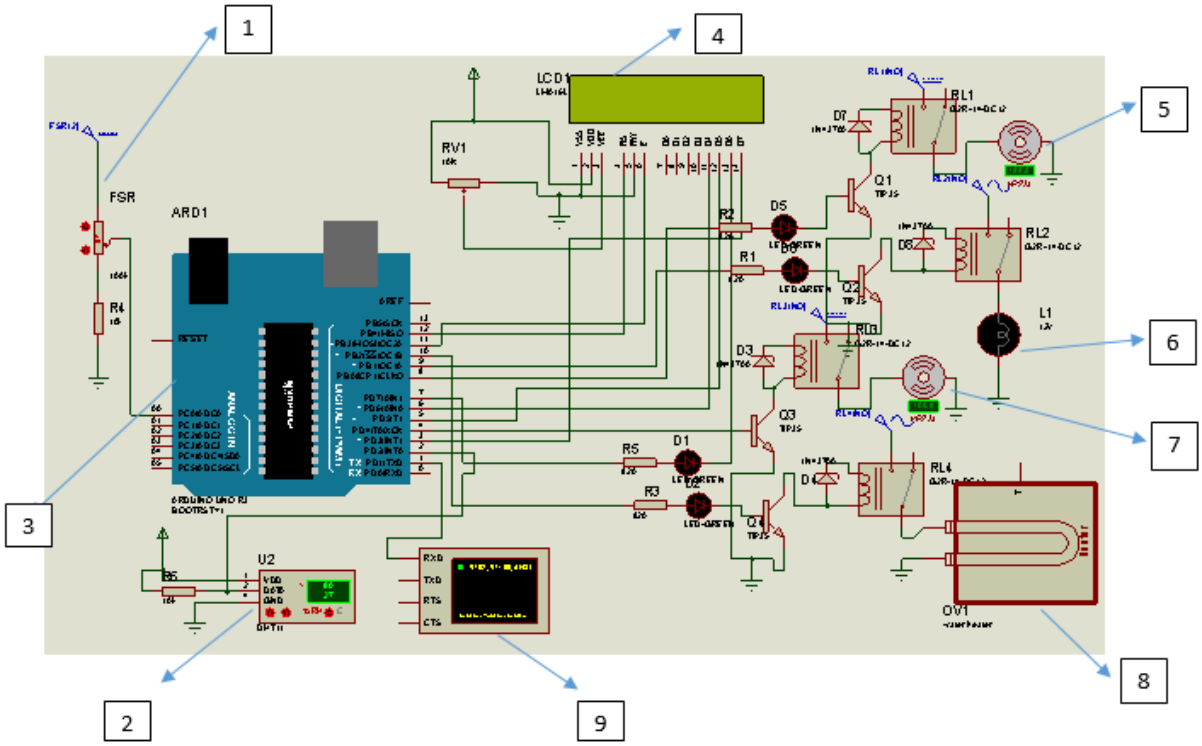

Figure 2. System diagram of the Centralised Incubator Control System

1-is the force sensitive resistor (FSR). It is basically a resistor that changes its resistive value (in ohms Ω) depending on how much it is pressed (Interlink Electronics - Sensor Technologies, 2010).

2- DAT11 humidity and temperature sensor. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high-performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness (D- Robotics UK, 2010).

3-Aurdino Uno kit. It is an open source electronics prototyping platform based on flexible, easy to use hardware and software. The microcontroller on the board is programmed using the Arduino programming language (based on wiring) and the Arduino development environment (based on Processing) (Durfee, 2011).

4- LMB162A a Liquid crystal display (LCD) (Datasheet.Hk. (2015).The LCD was used to display the temperature and humidity to enable the medical staff to monitor the baby's environment.

5-Control Actuators

6- Heater for temperature control

7- Control Actuators

8-Water heater for humidity control

9- Phototherapy control for adjusting ultraviolet light intensity.

2.2 Implementation of Force Sensitive Resistor (FSR)

The FSR was connected in a voltage divider connection with one pin of the FSR connected to +5V as shown in Figure 3. The 10K resistor was connected to the other FSR pin and to the Arduino as shown in Figure 3.

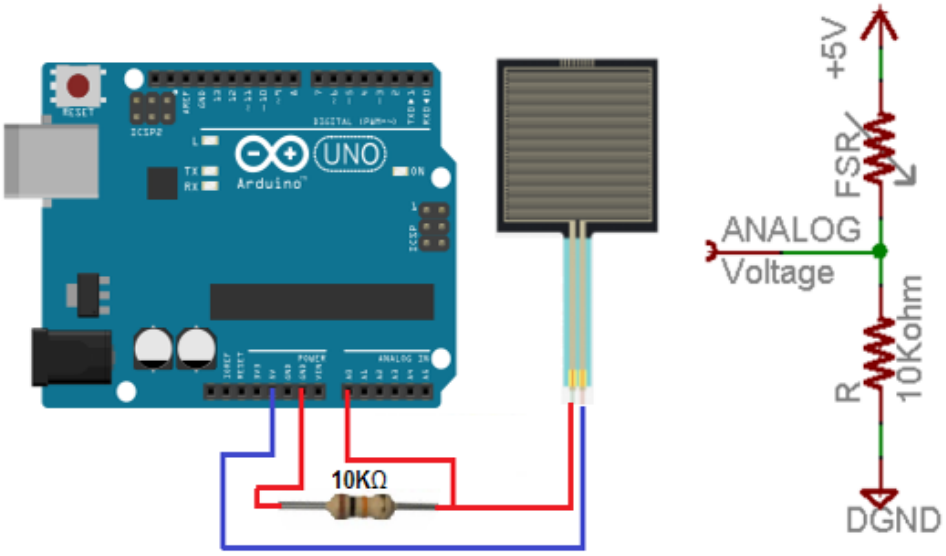

Figure 3. Circuit representation of Force Sensitive Resistor connection to Arduino

The voltage across the 10Kohm at different FSR resistances is given by equation 1.

$$V_R = V_{CC}\left(\frac{R}{R+FSR}\right) \qquad [1]$$

Where:-
V_R is voltage across the pull-up resistor,
R is the pull-up resistance
FSR is the force sensitive resistance and
V_{CC} is the supply voltage.

Some voltages across the 10K resistor were measured against variable resistances of the FSR and the respective results were as shown in Table 1.

Table 1. Results of FSR resistances versus voltage

FSR resistance	Voltage(R)
Infinite	0V
30kΩ	1.3V
6 kΩ	3.1V
1 kΩ	4.5V
250 Ω	4.9V

The FSR resistance varied with varying force exerted on the sensor as noted in Table 1. The FSR was calibrated and five objects of different weights in the range 1 to 5kg were measured and used. A graph of FSR resistance against weight was created and the results were as shown in Figure 4.

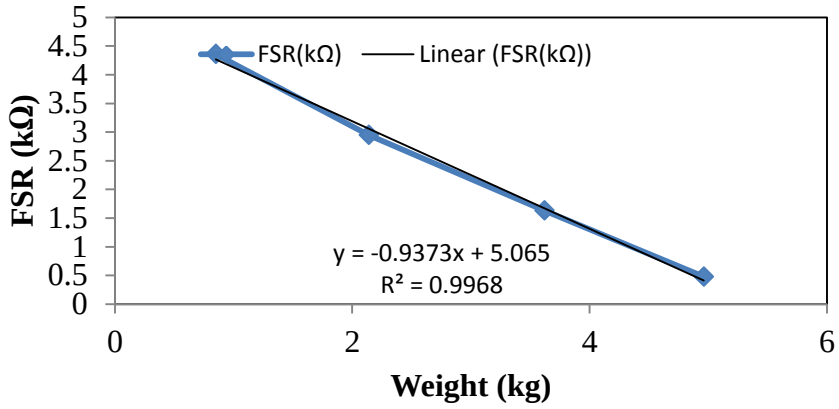

Figure 4. Graphical relationship between FSR resistance and weight.

The relationship between the FSR analog values and the weight was found to be linear for weights of 1kg to 5kg and gave an R^2 value approximately =1, implying a linear relationship. From this relationship the corresponding weight on the FSR could be obtained.

2.3 Temperature Control Implementation

Two circuits were used for temperature control; one circuit was used for heating and the other for cooling. To achieve this, actuation circuits were used and these made use of 12V DC relays which operated in the normally open mode (NO). Relay coils were energized through use of TIP35C, NPN transistors. These served as switches triggered by a high input on their base from the Arduino after respective temperature comparisons. Figure 5 illustrates the circuit diagram for the temperature control.

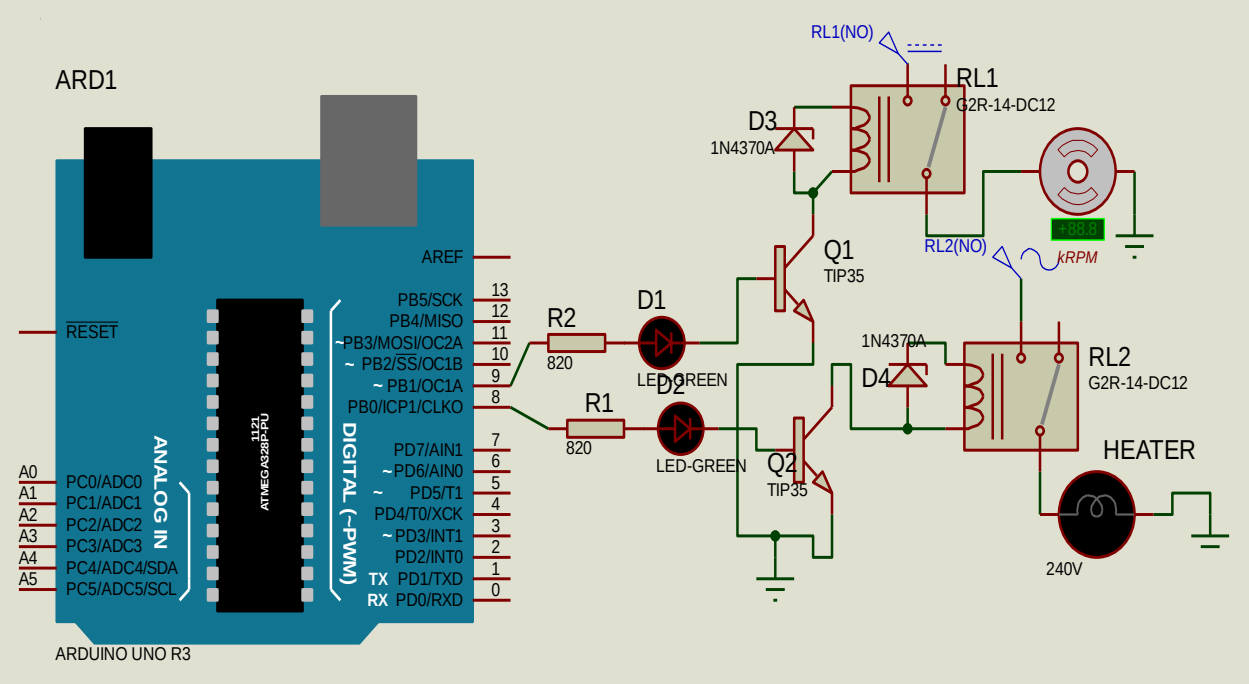

Figure 5. Circuit diagram for temperature control system.

Cooling of the system was done using 12V DC fan circuit through RL1 and the heating was performed by a 240V AC home heater rod through RL2. Once the respective control conditions were attained, the correct signal was then sent by the Arduino to the respective TIP transistor to either switch ON or OFF the heater or fan. However, a slight overshoot of temperature up to 38**°C** was noted after the heater had switched off and this increase was attributed to the DHT11 sensor being a relatively slow sensor. The overshoot was quickly noted and the fans were quickly set ON to offset this temperature increase.

2.4 Humidity Control Implementation

The humidity control had two circuits, one for driving humid air into the chamber and the other for reducing humidity. Humidifying of the system was done using an external humidifier circuit. The external humidifier used a 240AC water heating element to boil water the steam was driven into the chamber using a fan. Reducing humidity was initiated by driving in dry air using 12V DC fans. Actuation of these circuits was performed by 12V DC relays which operated in the normally open mode (NO). Energising of the relay coils was performed by TIP35c NPN transistors which served as switches and was triggered by a high input on the base (connected to the Arduino digital output pins through an LED). LEDs operated as indicators to show when the Arduino output was high or low. Circuit connections for humidity control were as illustrated in Figure 6.

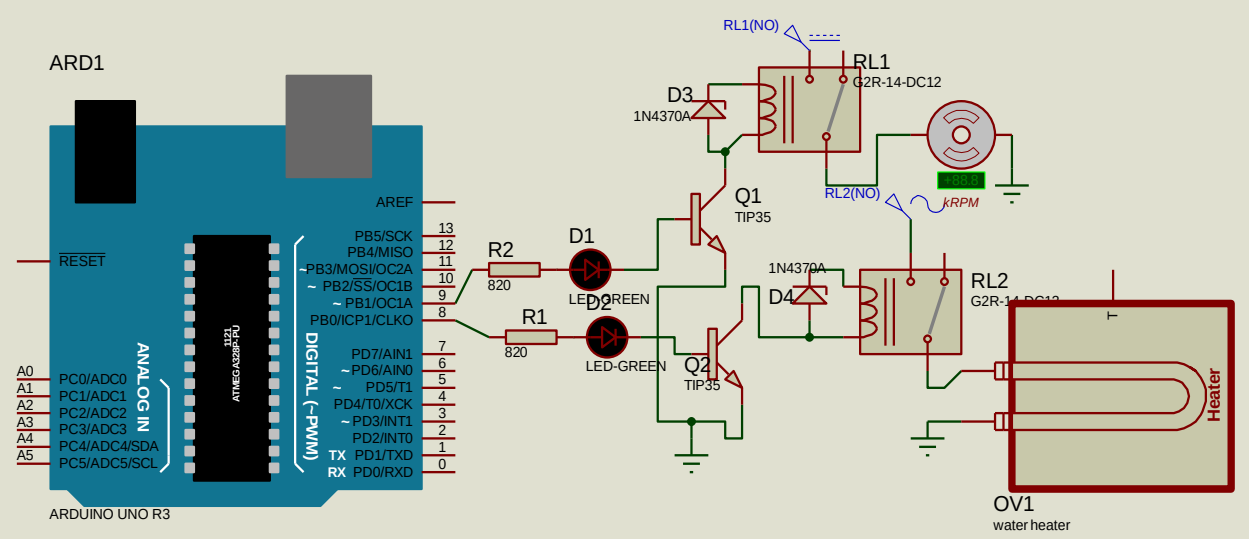

Figure 6. Circuit diagram for humidity control system.

The results of the set up were as shown in Figure 7;

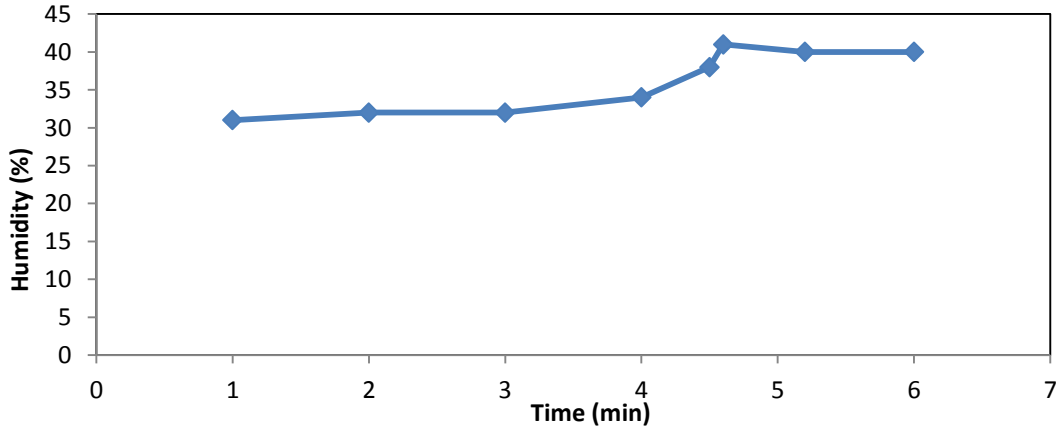

Figure 7. Humidity responds overtime

As shown in Figure 7, it took about 4.5 minutes for the humidity to attain the expected humidity of 42% in the chamber and also there was no overshot of humidity as was noted in the other trials due to control circuits

2.5 Phototherapy Hardware Design

The phototherapy design was meant to provide heating in the chamber. The circuit design shown in Figure 8, had a 10KΩ variable resistor RV1 used to adjust Ultra Violet light intensity.

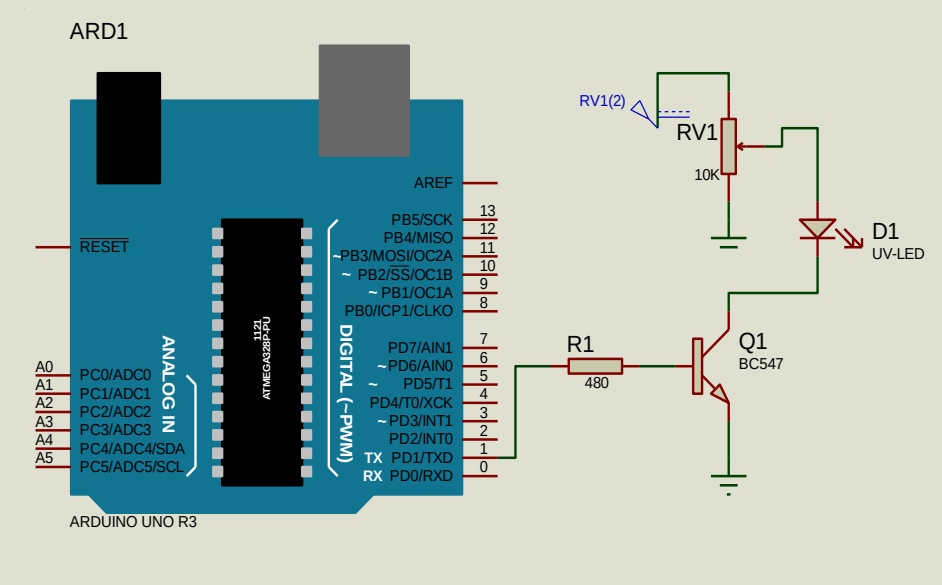

Figure 8. Phototherapy circuit diagram.

The switching circuit was designed using a BC547 transistor and this was to ensure that the lights were powered by the main 5V supply. A small current from the Arduino output into the base of the NPN transistor forms a short circuit between the emitter and the collector and thus completes the UV light circuit.

2.5 Incubator Alarm circuit

Incubator alarm circuit was design to provide a means to notify the user of any malfunction of the chamber. The alarm design was such that whenever the temperature dropped below 25 °C or exceeded 39 °C and the humidity dropped below 30% or exceeded 55% the buzzer would ring and a warning message would be displayed on the LCD showing the parameter that has gone into unsafe range.

The alarm circuit was design by connecting the buzzer's positive to the dedicated microcontroller output pin and the negative to ground. This meant that whenever the output pin status was set to 'HIGH' the buzzer would be powered and ring.

There was software design used to log the following; weight, temperature and humidity.

For temperature control, the microcontroller was programmed to check if the temperature reading from the DHT11 sensor was less than 36°C. If this condition was found to be true, then the controller would initiate actuation of the heater circuit. When the temperature became greater than 37°C the heater was turned off. At temperatures greater than 37°C the controller would initiate the actuation of the fan circuit. The fan would run until the new temperature was less than or equal to 37°C and the fan would be turned off and the loop restarts.

For Humidity control program, the sequence of flow was such that the microcontroller would check if the humidity value from the DHT11 sensor was less than 40%. Each time the condition is not satisfied then the controller initiates actuation of the humidifying circuit. When humidity becomes greater than or equal to 40% then the humidifier was turned off. In a different loop the controller would check if the humidity was greater than 40 percent, if this condition was true the controller would initiates the actuation of the fan circuit to drive dry air into the system. When humidity reached a value less than or equal to 40%, the fan system was turned off and the loop restarts.

Temperature and humidity were read by the DHT11 sensor and the Arduino microcontroller was programmed as shown in Figure 6 to read the temperature and humidity values

2.6 Microcontroller Program Design

The microcontroller program was designed to be made up of a main routine and a set of subroutines. The main loop algorithm was:

- Read sensors, compare read values to set points and set user defined flags as according to the relationship between the read values and respective set points.
- Run the temperature control actuators based on flag states.
- Run the humidity control actuators based on flag states.
- Send read values to the LCD for display.
- Send read values to the computer for display.

Each of the above steps was written as a subroutine. In addition a program section to configure the microcontroller was written at the start of the main loop. The configuration code was only run once when the microcontroller is first switched. The rest of the subroutines were called in turns and in a continuous loop.

Aurduino C programming language was used to write the microcontroller program. The program was written in such a way that sampling of sensors whose values was carried out at regular intervals by ensuring the same number of instructions was executed between samples.

2.6.1 Reading Sensors

All the sensors were read and user defined flags were set to respectively indicate the following:

- Temperature above 37°C
- Humidity below 40%
- Humidity above 50%

The flags were then used by other subroutines within the program.

2.6.2 Temperature Control

Figure 9 shows the subroutine that was used to control the temperature. The sequence of flow was such that the microcontroller checked if the temperature reading from the DHT11 sensor was less than 37°C. If this condition was true the controller initiated actuation of the heater circuit. When the temperature became greater than 37°C, the heater was turned off. At this instance the controller checked if the temperature was greater than 37°C.

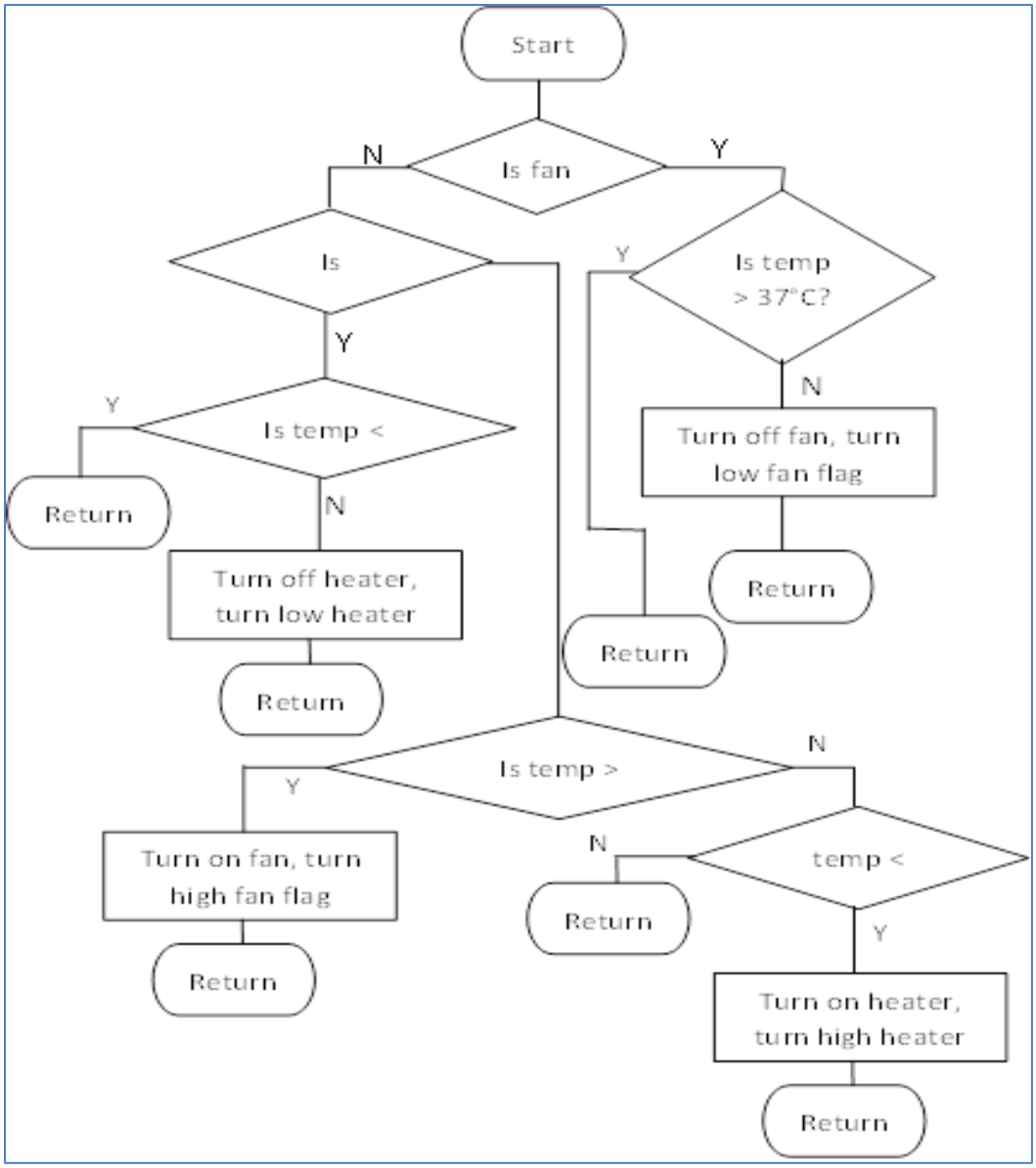

Figure 9. Temperature control flow diagram

If this condition was true the controller initiated the actuation of the fan circuit. When the new temperature value was now less than or equal to 37°C, the fan was turned off.

The design of the algorithm made this to be possible using subroutines. The use of user defined flags enabled the software to tell where to start checks and whether to actuate the heater or the cooling fan in each run of the loop.

2.6.3 Humidity Control

Figure 10 shows the subroutine for control of humidity. It follows the same structure as that used for temperature control to enable use of subroutines to do more than one task in the same loop. Again user defined flags were used to enable the software to tell whether to actuate the humidifier or the evacuation fan in each run of the loop.

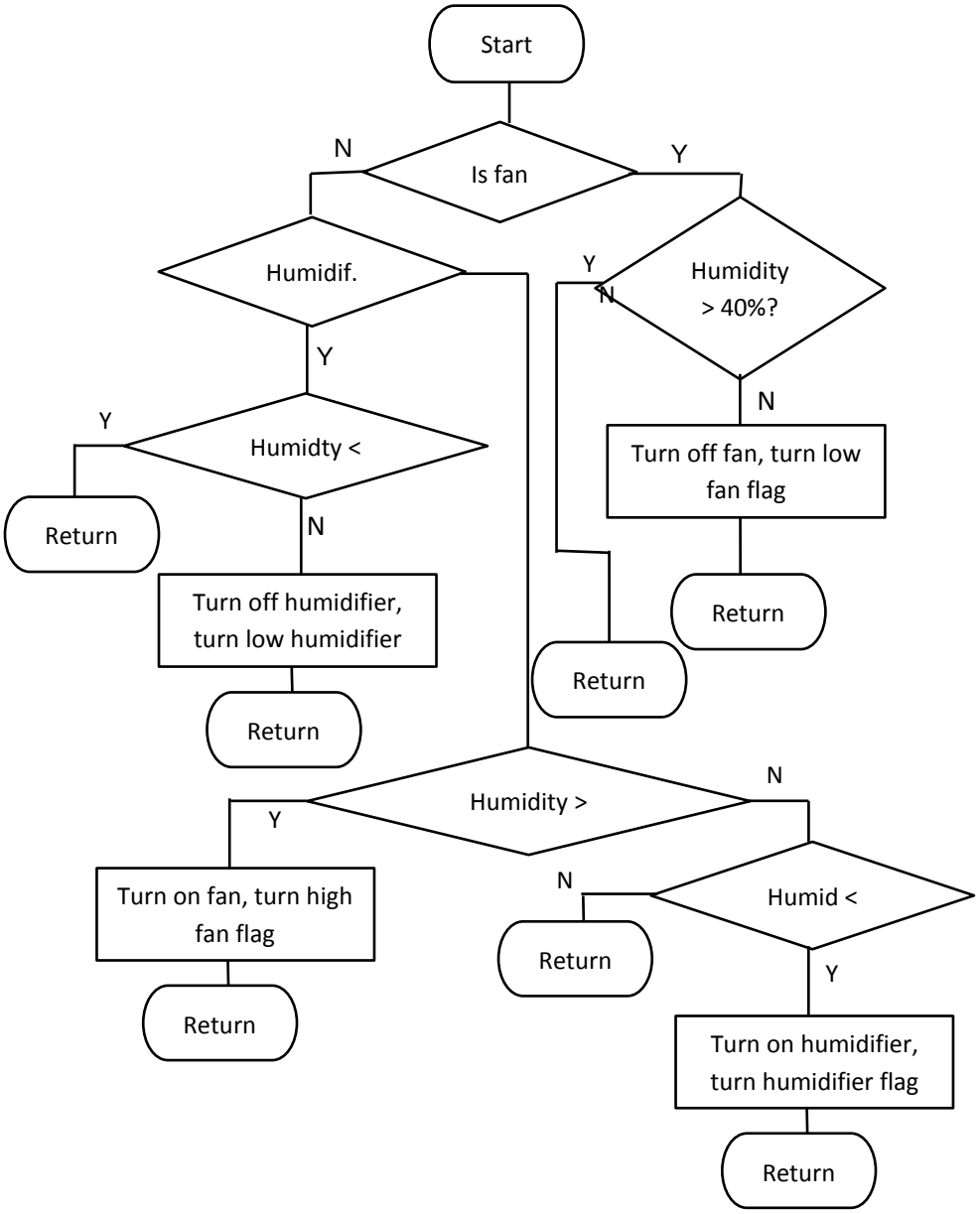

Figure 10. Humidity control flow diagram

The sequence of flow was such that the microcontroller checked if the humidity value from the DHT11 sensor was less than 40%. Each time the condition was satisfied, the microcontroller initiated actuation of the humidifying circuit. When humidity became greater than or equal to 40%, the humidifier was turned off. The microcontroller also checked if the humidity was greater than 50%. If this condition was true the microcontroller initiated the actuation of the fan circuit to drive dry air into the system. When humidity was now less than or equal to 50, the fan system was turned off.

2.6.4 Display of values on the LCD

A subroutine that sent values of weight, humidity and temperature was developed to enable display of these values on the LCD. It ran once in each run of the loop.

2.6.5 Sending of Values to the Computer

A subroutine was also developed to send values of weight, humidity and temperature to a computer via the serial port for display on the computer. It also ran once in each run of the loop.

2.7 Computer Software

The computer provided the user interface in the form of a GUI. Design of this interface was done using C# programming in Visual Studio. The interface allowed user to start logging real time data from the incubator with date and time stamp. The user would stop the log and save it in a text format for future use. Logging was done with a spacing interval of 5 minutes to allow user to note any changes in system behaviour.

2.8 The Complete System

The complete centralised incubator system was as shown in Figure 11.

Figure 11. Complete Centralised Incubator system.

3. RESULTS AND DISCUSSION

Several tests were carried out to check on the performance of the incubator system. At first, the temperature and humidity performance test was carried out.

3.1 Temperature and Humidity Control

Temperature and humidity were both lower than the required range when the performance test was carried out. This activated the heater and humidifier at the same time. As the heater increased the chamber temperature, it also dried the incubator air thus reducing humidity. This triggered the humidifier ON and as the

humidifier switched ON, it introduced humid air which in turn increased the chamber temperature. The results were such that the chamber quickly attained optimum temperature level and took much longer to achieve the required humidity. The temperature overshoots caused the fans to turn ON, catalysing the rate at which the chamber was humidified. This brought about the warm air being evenly distributed in the chamber at a faster rate. Results from the performance test were as shown in Figure 12.

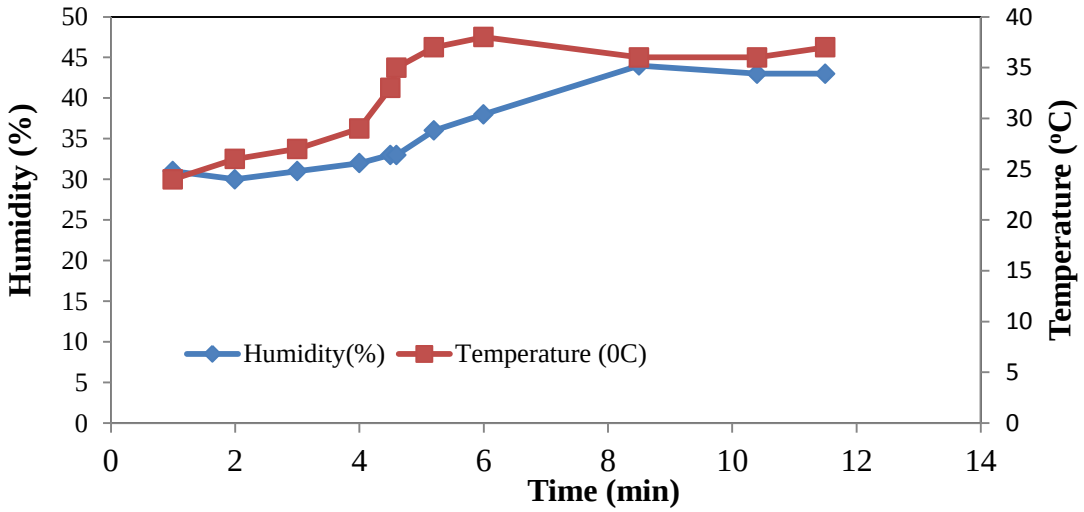

Figure 12. Humidity/Temperature responds overtime

From Figure 12, it can be noted that the temperature of the chamber gradually rose to 37^{o}C in approximately 5 minutes. The rise in temperature was considered to be safe for the infant since it was gradual and as such it would not bring about the shock effect on the infant. The humidity rose to 40% in 7 minutes and the rise was also gradual, meaning no shock effect to the baby. A sudden rise of these parameters is not safe for the infant.

The system was powered and left for a time frame of about 5hrs in three different occasions to test its performance accuracy as an integrated system and the system performance was as shown in Figure13:

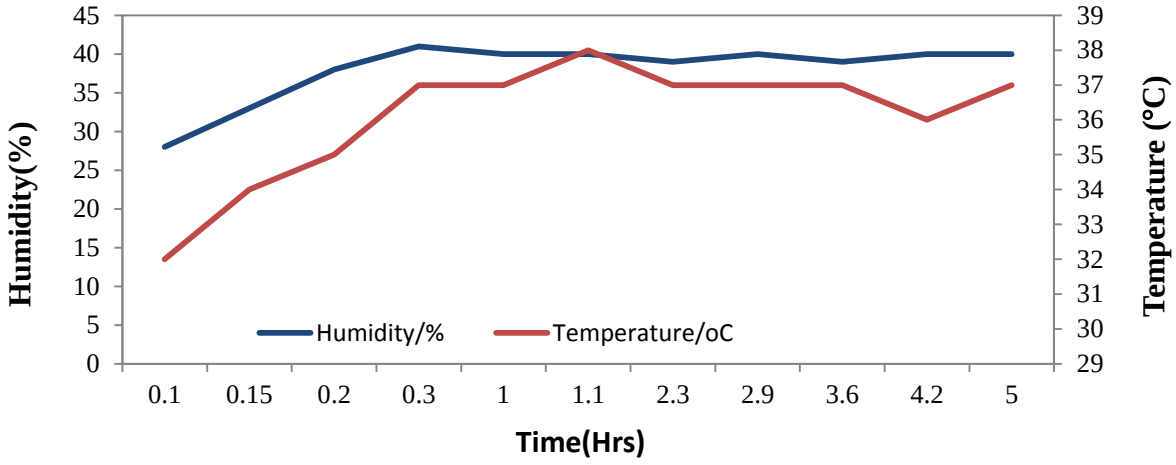

Figure 13. Humidity/Temperature response in 5 hours

It took an average of 15 minutes for the system to attain both the required temperature and humidity levels. Slight deviations were quickly normalized by respective control circuits. Mainly the control loops that were activated in these tests were low humidity and low temperature loops.

3.2 Weight measurement

The force sensor pad used was relatively accurate for weights greater than 0.4kgs. The main challenge was for low weight values, the FSR's resistance increased towards infinity. However premature babies' weights have been found to be always greater than 1kg. Linear properties were attained for weights greater than 0.4kgs (see Figure 4).

3.3 Buzzer alarming circuit

The buzzer circuit was designed to give an alarm whenever the chamber reached critical levels. Figure 14 shows some of the alarms tested on.

Figure 14. Incubator warning messages on LCD

The conditions were tested by inducing the temperature and humidity conditions and the results were as shown in Figure 14. Each alarm was supported by a message displayed on the LCD.

3.4 Ultra Violet light circuit

The circuit for UV light control was operating with a provision for adjusting the light intensity as shown in Figure 15.

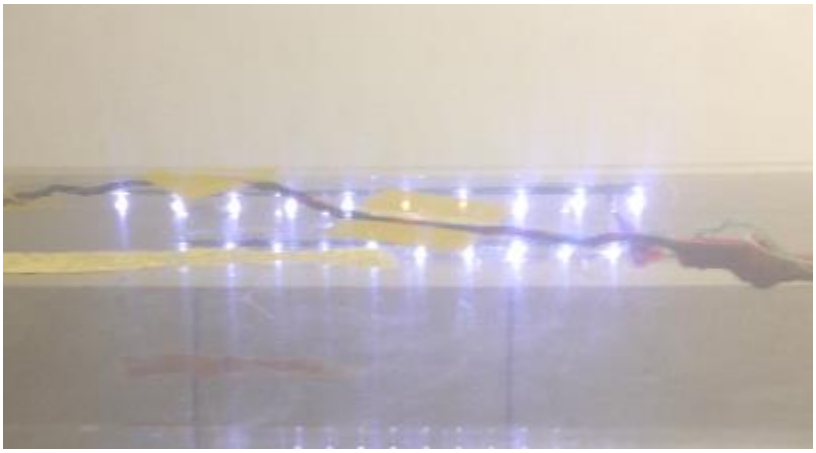

Figure 15. Ultra-violet lights for phototherapy

The circuit was however not tested to determine the range of intensity that could be suitable for different jaundice levels. The intensity could be varied. The respective levels of the intensity would require expensive equipment and expertise.

3.5 User Interface

The user interface designed could enable real time events from the incubator to be displayed after every minute as shown in Figure 16. It allowed the user to start and stop logging the data. Data was logged with a time stamp; this was to allow referencing of events in the future.

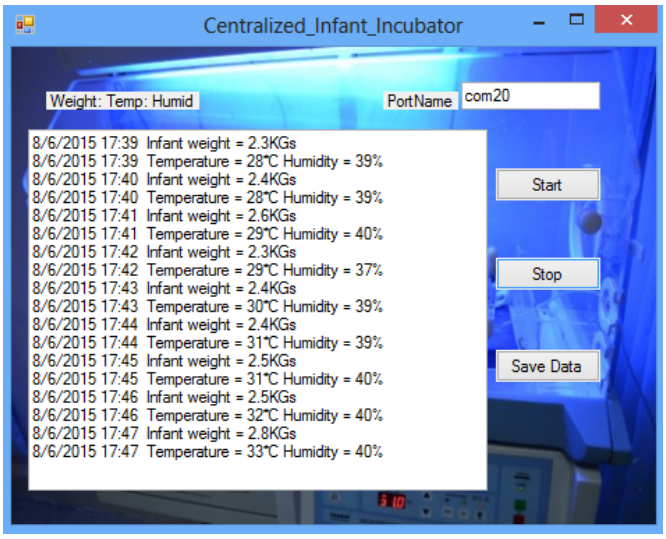

Figure 16. User interface for Centralised Incubator

The data saved could be used during investigations in the event that the incubator operation was compromised. The date/time stamp allows the user to know the exact time certain events occurred.

4. CONCLUSION

The design and implementation of the centralised incubator control system was achieved. The temperature and humidity were measured and could be controlled continuously. Any slight changes in the chamber temperatures, set at 36.5°C and humidity, set at 40% could be maintained within a period of 5 minutes. The system could also monitor the weights from 1kg to 5kg. A serial computer user interface was established and this could allow an efficient system monitoring, saving and storage of the incubator data. Challenges encountered included slow humidity increase and weight sensor accuracy for weights less than 0.4Kg.

REFERENCES

UN Inter-agency Group for Child Mortality Estimation. (2013). Levels& Trends in Child Mortality [online]. Available from: http://www.childinfo.org/files/Child_Mortality_Report_2013.pdf[Accessed 16 July 2014]

World health organization Geneva. (1998). The World Health Report 1998; *Life in the 21st century: Chapter 5 (Achieving health for all)* [online]. Available from http://www.who.int/whr/1998/en/whr98_en.pdf. [Accessed 10th September 2014]

Interlink Electronics - Sensor Technologies, (2010). FSR 400 Data Sheet [online]. Available from: http://www.trossenrobotics.com/productdocs/2010-10-26-DataSheet-FSR400-Layout2.pdf [Accessed 24th January 2015].

D- Robotics UK, (2010). DHT11 Humidity & Temperature Sensor [online]. Available from: http://www.micropik.com/PDF/dht11.pdf[Accessed 20th February 2015]

W. Durfee. (2011). Arduino Microcontroller Guide [online]. Available from: http://www.me.umn.edu/courses/me2011/arduino/arduinoGuide.pdf. [Accessed 16thOctober 2014]

Datasheet.Hk. (2015). Specification Model LMB162A [online]. Available from: http://www.datasheet.hk/view_online.php?id=1143259&file=0069%5clmb162a_570598.pdf
[Accessed 13th October 2014]