T.C.
İSTANBUL UNIVERSITY
INSTITUTE OF GRADUATE STUDIES IN
SCIENCE AND ENGINEERING

M.Sc. THESIS

DESIGN OF INFANT INCUBATOR ANALYZER

Ahmed Talal SALLAM

Department of Biomedical Engineering

Biomedical Engineering Programme

SUPERVISOR
Assoc. Prof. Dr. Mana SEZDI

June, 2018

İSTANBUL

This study was accepted on 27/6/2018 as a M. Sc. thesis in Department of Biomedical Engineering, Biomedical Engineering Programme by the following Committee.

Examining Committee Members

Assoc. Prof. Dr. **Mana SEZDI**(Supervisor)
İstanbul University
Faculty of Engineering

Prof. Dr. Mahmut ÜN
Yeni Yüzyıl University
Faculty of Engineering

Assoc. Prof. Dr. Yasin OZÇELEP
İstanbul University
Faculty of Engineering

Assist. Prof. Dr. Abdurrahim AKGÜNDOĞDU
İstanbul University
Faculty of Engineering

Assist. Prof. Dr. Rana Ortaç KABAOĞLU
İstanbul University
Faculty of Engineering

As required by the 9/2 and 22/2 articles of the Graduate Education Regulation which was published in the Official Gazette on 20.04.2016, this graduate thesis is reported as in accordance with criteria determined by the Institute of Graduate Sttudies in Science and Engineering by using the plagiarism software to which İstanbul University is a subscriber.

FOREWORD

I express my sincerest gratitude to my father Prof. Talal A. SALLAM for supporting me and guiding me through my masters journey. I will always and forever be thankful for his sacrifices to finance my studies and making me be who I am. I also thank my lovely mother Basma who has always supported me with her prayers. I cannot forget my beloved wife Lamis, she was nothing but supportive and encouraging all the way.

I am also thankful to my supervisor Assoc. Prof. Dr. Mana SEZDI, for her guidance and advice through my studies.

June 2018 Ahmed Talal SALLAM

TABLE OF CONTENTS

Page

FOREWORD .... **iv**
TABLE OF CONTENTS .... **v**
LIST OF FIGURES .... **vii**
LIST OF TABLES .... **ix**
LIST OF SYMBOLS AND ABBREVIATIONS .... **x**
ÖZET .... **xii**
SUMMARY .... **xiv**
1. INTRODUCTION **1**
1.1 Neonatal health care 1
1.2 History of neonatal health care 1
1.3 Neonatal healthcare in Turkey 2
1.4 Temperature effect on neonatal babies 4
1.5 Humidity effect on neonatal babies 5
1.6 Sound noise effect on neonatal babies 7
1.7 Preventive maintenance 7
1.8 Infant incubator analyzer 8
1.9 Aim of the study 8
2. MATERIALS AND METHODS **10**
2.1 AVR microcontroller 11
2.2 Display 12
2.3 LM35 temperature sensor 15
2.4 DHT11 humidity / temperature sensor 17
2.5 Sound noise sensor 21
3. RESULTS **26**
3.1 LM35 temperature sensor 26
3.2 DHT11 humidity / temperature sensor 27
3.3 Sound noise sensor 28
3.4 GLCD display 29
3.5 Final implementation 30

3.6 Testing the unit...32
4. DISCUSSION...35
5. CONCLUSION AND RECOMMENDATIONS...37
REFERENCES...38
APPENDICES...42
1. GLCD code...42
2. Sensors code...49
CURRICULUM VITAE...58

LIST OF FIGURES

Page

Figure 1.1: Trainer's incubator……………………………………………………...…....2
Figure 1.2: Role of neonatal health care intervention decreasing mortality rate in Turkey…..4
Figure 1.3: Experimental arrangement for active humidification………………………..…..6
Figure 1.4: Passive humidity control system………………………….………...…....6
Figure 1.5: IncuTest Infant Incubator Analyzer..8
Figure 1.6: INCU Infant Incubator Analyzer...8
Figure 2.1: Infant incubator analyzer design block diagram……………….……………..…...9
Figure 2.2: 8 channel 10 bit ADC…………………………………….………………..…10
Figure 2.3: Internal block diagram of 128x64 GLCD along with its pin out……….………12
Figure 2.4: LM35 Circuit diagram…………………………………...……….……......14
Figure 2.5: Centigrade temperature sensor…………..……………………….…………..14
Figure 2.6: Resistive-type relative humidity sensor………………………....………...16
Figure 2.7: Typical application of DHT11………………………………..…………….…..17
Figure 2.8: DHT11 overall communication process……………………………..………18
Figure 2.9: Condenser microphone diagram…………………………………….……….20
Figure 2.10: LM386 Pin diagram……………………………………………………..…20
Figure 2.11: Schematic for the amplification circuit..………………………………………21
Figure 3.1: LM35 sensor simulation circuitry on Proteus……………….………………….24
Figure 3.2: Measuring body temperature through LM35………………………..………..24
Figure 3.3: DHT11 sensor simulation circuitry on Proteus……………………………..….25
Figure 3.4: Measuring room humidity and temperature with DHT11……………..……..25
Figure 3.5: Sound noise sensor circuitry…………………………………………………….26
Figure 3.6: Sound sensor obtained result in a quiet room………………………..…….26
Figure 3.7: Sound sensor obtained result with music turned on…………………..……...27
Figure 3.8: Graphical display code test on Proteus…………………………………..…...27
Figure 3.9: Graphical display code results on testing board………………………………...28
Figure 3.10: All components circuitry simulation on Proteus ……………………….………..29
Figure 3.11: Circuit implemented on a test board ……………………………………….30

Figure 3.12: Enclosed device..31

Figure 3.13: Testing our device in a Drager Isolette® Infant Incubator with a Datrend IncuTest incubator Tester …………………………………………………………32

Figure 3.14: Testing our device in Girffe OmniBed Infant Incubator with a INCU Fluke incubator analyzer……………………………………………………………….…33

LIST OF TABLES

Page

Table 1.1: The percentage of the new-born that were treated with hospitalization in 2014 and 2015 in İstanbul……………………………….…………………...……………..….3

Table 1.2: Effect of relative humidity on normal new-born babies under controlled environmental conditions……………………….………………………...…..…..5

Table 2.1: Design Parameters ....………………………......……………..…........…..10

Table 2.2: All GLCD detaild pins info……………………......…………….........……..11

Table 2.3: 10 ADC values and its equivalent in dB……………………….........…....….23

Table 3.1: Results obtained from testing our device in a Drager Isolette® Infant Incubator with a Datrend IncuTest incubator Tester …………………………………………….33

Table 3.2: Results obtained from testing our device in Girffe OmniBed Infant Incubator with a INCU Fluke incubator analyzer……………………………………...…………34

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol	Explanation
dB	: Decibel
^{o}C	: Degree Celsius
DC	: Direct Current
GND	: Ground
I/O	: Input/Output
µF	: Microfarad
MIC	: Microphone
µs	: Microsecound
mV	: Millivolts
ms	: Millisecond
Vss	: Most negaive supply terminal
Ω	: Ohm
ω	: Omega
%	: Percentage
±	: Plus or minus
Vdd	: Positive voltage supply (drain)
POT	: Potentiometer
RST	: Reset
ρ	: Rho
Avcc	: The supply voltage for Port A and A/D converter
V	: Volt
Vcc	: Voltage at the common collector
Vref	: Voltage reference

Abbreviation	Explanation
ADMUX	: ADC Multiplexer Selection Register
AAP	: American Academy of Pediatrics
AIC	: American Interdisciplinary Committee
ADC	: Analog to Digital Converter
EEPROM	: Electrically Erasable Programmable Read-Only Memory

GLCD	: Graphical Liquid Crystal Display
IC	: Integrated Circuit
KB	: Kilo Bite
LCD	: Liquid Crystal Display
MCU	: Microcontroller Unit
NTC	: Negative Temperature Coefficient
NICU	: Neonatal Intensive Care Unit
PM	: Preventive Maintenance
PWM	: Pulse Width Modulation
RAM	: Random-Access Memory
RH	: Relative Humidity
RISC	: Reduced Instruction Set Computing
ROM	: Read-Only Memory
SPI	: Serial Peripheral Interface
SPL	: Sound Pressure Level
SRAM	: Static Random Access Memory
USART	: Universal Asynchronous Receiver-transmitter
USB	: Universal Serial Bus
WHO	: World Health Organization

ÖZET

YÜKSEK LİSANS TEZİ

KUVÖZ ANALİZ CİHAZI TASARIMI

Ahmed Talal SALLAM

İstanbul Üniversitesi

Fen Bilimleri Enstitüsü

Biyomedikal Mühendisliği Anabilim Dalı

Danışman : Doç. Dr. Mana SEZDI

Türkiye'de yeni doğan bebeklerin yaklaşık % 80'i kuvöze yerleştirilmektedir. Kuvözde anne karnı simule edilerek yeni doğanların yaşamsal parametrelerinin sürdürülmesi sağlanmaktadır. Kuvöz, bebeklere uygun sıcaklık ve nemde gürültüsüz bir ortam sunmaktadır. Kuvözlerin bu fonksiyonunu eksiksiz yerine getirmesi, kuvözlerin koruyucu bakımlar ile kontrol altında tutulması ile mümkündür. Bu çalışmanın amacı, kuvözlerin sürdürülebilir çalışmasını sağlayan güvenilir bir kuvöz analizörü tasarlamak ve geliştirmektir. Bu çalışma, bir AVR Atmega32 mikrodenetleyici birimi tarafından kontrol edilen ve bir LCD ekran ile arayüzlenen dört farklı sensör vasıtasıyla kuvözün fonksiyonel parametrelerini izlemek için kullanılabilecek bir analizörün ayrıntılı tasarımını sunmaktadır. Analizörde, sıcaklık, nem ve gürültü ölçümü hedeflenmiştir. Önerilen tasarım, daha fazla optimizasyon ve test gerektiren bir ön prototiptir. Bu çalışmadan elde edilen tasarım önerilerinin, gelecekteki benzer tasarımların temelini oluşturacağı düşünülmektedir.

Haziran 2018, 73 sayfa.

Anahtar kelimeler: Kuvöz, Yenidoğan İnkubatörü, İnkubatör Analizörü, Koruyucu Bakım, Yenidoğan Yoğun Bakım Ünitesi

SUMMARY

M.Sc. THESIS

DESIGN OF INFANT INCUBATOR ANALYZER

Ahmed Talal SALLAM

İstanbul University

Institute of Graduate Studies in Science and Engineering

Department of Biomedical Engineering

Supervisor : Assoc. Prof. Dr. Mana SEZDI

Approximately 80% of newborn babies are placed in incubators in Turkey. In the incubator, the mother's belly is simulated and the vital parameters of the newborns are maintained. The incubator provides a noiseless environment for babies with suitable temperature and humidity. The complete functioning of the incubator is possible only if the incubators are kept under control by preventive maintenance. The purpose of this work is to design and develop a reliable incubator analyzer that ensures the sustainability of incubators. This work presents a detailed design of an analyzer that can be used to monitor the functional parameters of the incubator by means of four different sensors controlled by an AVR Atmega32 microcontroller unit and interfaced with an LCD display. In the analyzer, temperature, humidity and noise measurements have been targeted. The proposed design is a preliminary prototype that needs further optimization and testing. It is believed that design recommendations generated from this study will form the basis of future similar designs.

June 2018, 73 pages.

Keywords: Infant Incubator, Infant Incubator Analyzer, Preventive Maintenance, Infant Incubator Maintenance, Neonatal Intensive Care Unit

1. INTRODUCTION

1.1. NEONATAL HEALTH CARE

Each year, about one million infants die due to problems that can be prevented by an intensive care unit [1]. Premature and neonatal babies sometimes needs extra attention and health care because of their problematic conditions [2]. Congenital anomaly and new-born baby's inability to regulate their body temperature is a leading cause of premature infant death as well as permanent disability [3][4][5]. Other factors such as the quality of care during pregnancy and delivery and the immediate care of the new-born play a major role in infant mortality [6]. An important approach of neonatal care is the use of incubators [7] which are a baby crib equipped with a thermal controlled environment. In addition to these incubators protects babies from infections, allergies, and harmful noise or light [8]. Therefore, vulnerable new-borns especially premature ones are placed in incubators providing them with the closest environment to that of the mother's uterus. In these incubators infants are kept for observation and intensive care by controlling vital parameters such as desirable temperature, humidity and airflow. Infants are kept in neonatal intensive care unit NICU until a full development of their organs. Infant incubators protect the babies from NICU disturbance and infections. Basically infant incubator is a chamber with mattress covered with a plastic cover. This chamber provides the required environment for infants to be hospitalized. The incubator heater adjusts to maintain the babies' temperature, and a temperature sensor is tapped to the babies' skin to keep temperature controlled. Generally, incubators are used for maintaining a stabilized thermoneutral ambient, provision of desired humidity and oxygenation, surveillance for highly sick infants, isolation newborn babies from infections and unfavorable external environment.

1.2. HISTORY OF NEONATAL CARE

As the high mortality within premature babies in the 17th and 18th centuries due to congenital diseases, scholarly papers were published to highlight this dilemma. However, it was until the early 19th century neonatal babies began to receive intensive care [9]. In the 18^{th} century doctors had an increasing role in childbirth. However neonatal care was left in the hand of mothers. Pierre-Constant Budin, was a pioneer in infants intensive care in the early 19^{th} century by devoting his career to reducing infant mortality [9][10]. He encouraged new mothers education about hygiene and nutrition [11]. He also discovered the process of feeding directly to the stomach via a tube to those infants who were unable to feed normally. His work increased the survival rate among infants by 28% over three years [11]. At this time, pediatricians started to realize the seriousness of this issue and hospitals started treating neonatal babies in groups called neonatal intensive care unit NICU [12]. By bringing light to neonates and premature babies health care in the 18th century, prototypes of infant incubators started to develop in Russia and France between 1830 to 1890 [11]. They were essentially developed to keep babies warm. The first invention was associated with the French obstetrician Stephane Tarnier. This incubator was warmed using hot water reservoirs. Tarnier's incubator (Figure 1.1) made a quantum leap in warming premature babies [11]. However this device was large and expensive. After that, Budin continued developing upon Tarnier's incubator improving upon the original incubator model [12]. Over the years, several improvements have been made to the basic design which contributed dramatically in the decrement of neonates mortality [13]. Over the time, the infant incubator technology advancement has slowed and the basic design has remained basically the same in the last 30 years [14]. This has necessitates continuous improvement in the design of infant incubators [15].

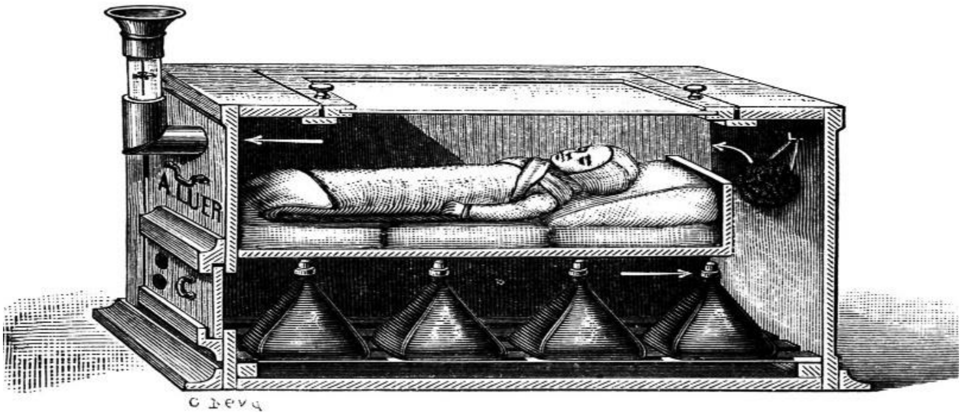

Figure 1.1: Trainer's incubator.

1.3. NEONATAL HEALTH CARE IN TURKEY

In 1957, the first neonatal intensive care unit NICU was built in Hacettepe University Children's Hospital. Nowadays, Turkish health infrastructure has about 6728 infant beds (Infant Incubators) in a total of 194 health centers and hospitals [6]. In Turkey 1,279,864 births occurred in 2012 with more than 80% of neonatal babies were hospitalized in infant incubators

[2]. Even with the presence of infant incubators, 14,845 infants died. Sixty five present of these died during the neonatal period [16]. A major problem affecting neonates health in Turkey is the quality of health care given to the baby [17]. A study was conducted by S. Uslu et al., 2016 showed that the percentage of the new-born that were treated with hospitalization in 2014 and 2015 in Istanbul has been shown to be between 24.8% to 48.0% (Table 1.1) [18].

Table 1.1: The percentage of the new-born that were treated with hospitalization in 2014 and 2015 in İstanbul [18].

Level of Beds	State Hospitals		Private Health Institutions		Total	
	Number of Hospitalized Patients (%)		Number of Hospitalized Patients (%)		Number of Hospitalized Patients (%)	
	2014	2015	2014	2015	2014	2015
Level I	4,844 (33.7)	4,996 (34)	7,132 (24.3)	7,196 (23.7)	11,976 (27,4)	12,165 (26,9)
Level II	5,521 (38.4)	5,669 (38.6)	5,330 (18.2)	5,689 (18.6)	10,851 (24,8)	11,358 (25,1)
Level III	3,997 (27.9)	4,033 (27.4)	16,907 (57.6)	17,651 (57.7)	20,904 (47,8)	21,684 (48,0)
Total	14,362 (100)	14,698 (100)	29,369 (100)	30,536 (100)	43,731 (100)	45,234 (100)

State Hospitals*: Turkey State Hospitals Association and State University Hospitals.*
Private Health Institutions: *Private Hospitals and Private University Hospitals.*

Since 1998, Turkey has accomplished more advancements to their health policy [19]. Data from a study made by Demirel & Dilmen., 2011 showed that neonatal health care interventions played a major role in decreasing neonatal mortality rate in Turkey (Figure 1.2).

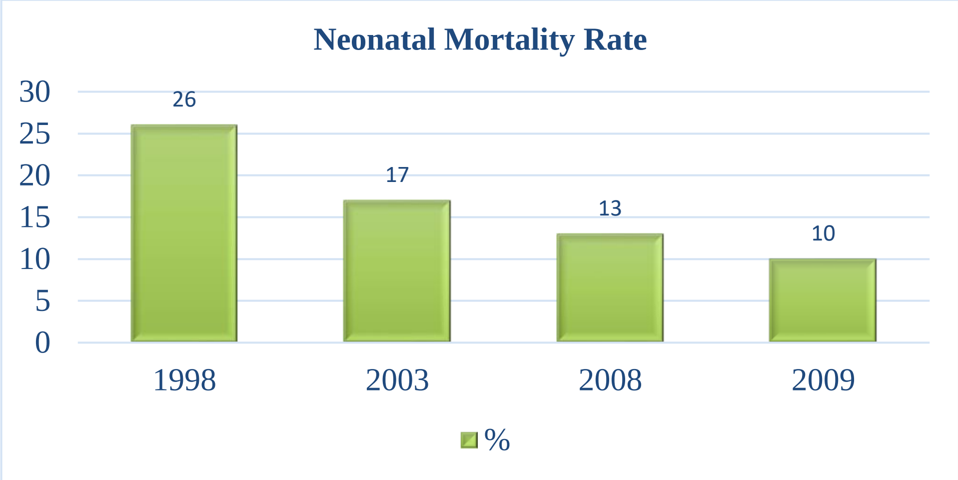

Figure 1.2: Role of neonatal health care intervention decreasing mortality rate in Turkey.

However studies revealed that a lot of neonatal babies deaths were preventable [6][19]. Therefore prevention strategies and solutions should be applied. Each major cause of neonatal mortality can be prevented or treated. Stepping up the quality of incubators preventive maintenance and inspection could be a major strategy to prevent deaths.

1.4. TEMPERATURE EFFECT ON NEONATAL BABIES

After delivery, neonates must control their body temperature. Normal body temperature is between 37 °C and 38 °C [20]. In uterus babies depend on mother's core temperature. After birth they rely on external heat to maintain their core temperature. Therefore their thermal instability expose them to hypothermia and hyperthermia [21]. Hypothermia is a main reason for infants mortality around the world [22]. Babies are not adaptable to temperature as adults. A baby can lose heat very fast due to their low body fat, especially premature babies [23]. Healthy babies as well may not be capable of maintaining their body temperature. When infants are put to bed, their temperature drops by about 1 °C [24]. With neonatal babies this small drop my increase until below 36.5 °C [24]. Therefore an infant should be placed in an ideal place post-delivery which is the infant incubator. Careful and continuous surveillance of neonatal babies' temperature have increased the survival rate of infants [25]. Neonatal textbooks give general temperature ranges for infants by 36.5 °C to 37.5 °C, skin as 36.2 °C–37.2 °C, and 36.5 °C to 37.3 °C for axillary sites [26]. A study carried out reported that infant mortality have decreased by 22% when neonates were nursed in incubators where air temperature is controlled [27]. The safe temperature for a standard incubator is around 32 °C [28].

1.5. HUMIDITY EFFECT ON NEONATAL BABIES

Controlling and monitoring neonates' temperature is not enough to protect them from heat loss. In compression to adults, infants are very sensitive to climate changes because physiologically they are less effectively adapting to humidity and other weather changes [29]. Neonates, especially premature babies, have an underdeveloped skin. Therefore they are at an increased risk of water evaporation through skin, leading to temperature instability, dehydration, electrolyte imbalance and calorie loss [19]. This could be protected by increasing the environment humidity. A study conducted by Hey & Maurice., 1968 have estimated the effect of relative humidity on normal new-born babies under controlled environmental conditions [30]. Results in Table 1.2 have been observed.

Table 1.2: Effect of relative humidity on normal new-born babies under controlled environmental conditions [30].

No. of experiments	Environmental Temp (ºC.) Mean ± SD	Rectal Temp. (ºC.) Mean ± SD	Relative Humidity (%) Mean ± SD	Heat Produced Mean ± SD	Heat Loss Mean ± SD	Mean Difference in Heat Production Mean ± SD	Mean Difference in Heat Loss Mean ± SD	Predicted deference in Respiratory Heat Loss
11	31.1 ± 0.30	36.8 ± 0.32	19.2 ± 1.8	42.7 ± 5.04	48.8 ± 5.73	-1.66 ± 1.51	-2.00 ± 0.96 $p < 0.05$	-1.1
			44.5 ± 2.4	41.0 ± 4.15	46.8 ± 9.92			
13	31.2 ± 0.26	36.7 ± 0.40	22.5 ± 4.1	45.0 ± 6.24	47.5 ± 4.84	-2.09 ± 1.34	-5.10 ± 0.94 $p < 0.001$	-2.7
			85.0 ± 2.8	42.9 ± 3.48	42.4 ± 3.50			
13	35.1 ± 0.20	36.8 ± 0.36	18.6 ± 5.3	34.6 ± 2.90	28.6 ± 5.72	+0.14 ± 0.75	-1.49 ± 1.36 $p < 0.3$	-1.4
			47.8 ± 1.5	34.7 ± 4.92	27.8 ± 6.76			

Note: Mean heat production and heat loss in calories/kg. min. at the levels of humidity investigated together with mean paired differences and tests of significance. The effect of the change in humidity on predicted evaporative heat loss from the respiratory tract is also given in the same units, based on an assumption that expired air leaves the nose 90% saturated ºC. below deep body temperature, and that pulmonary ventilation is 160 ml. BTPS/kg. min. at 35ºC. and 200 ml./kg. min. at 31ºC.

Usually there are two ways for incubators to control humidity. These are active and passive humidity control. In active humidity control an ultrasonic vaporizer is used to control the vapor (Figure 1.3). The system is supplied with a humidity sensor, therefor the relative humidity of air is measured and controlled.

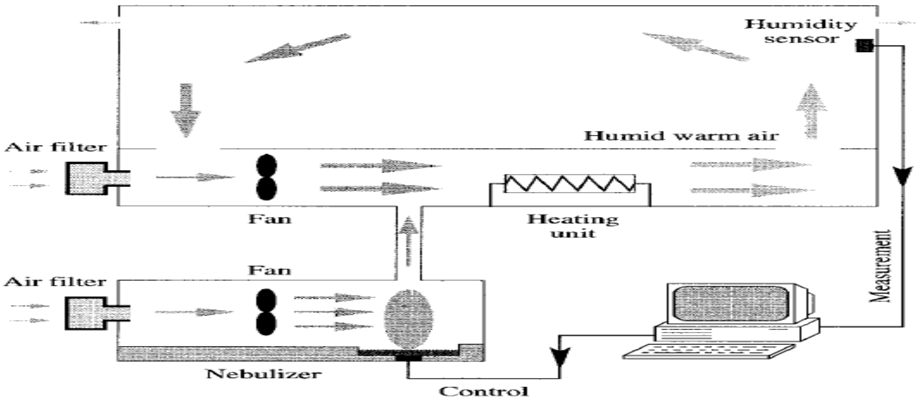

Figure 1.3: Experimental arrangement for active humidification [31].

The passive humidity control (Figure 1.4) is composed of a reservoir and a water whose crossed on the airflow generator to moisturize the air. Meaning that the humidity of air occurs when the air passes in the reservoir. There is no control mechanism because the relative humidity is not controlled [32].

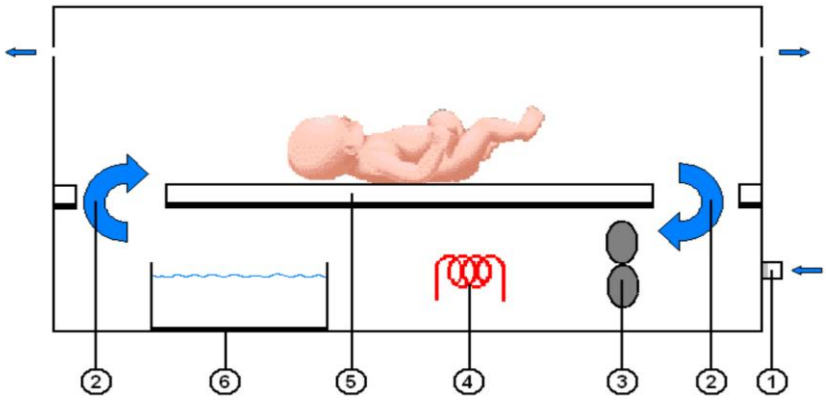

Figure 1.4: Passive humidity control system. 1: Air filter, 2: Air flow, 3: Weathercock, 4: Heater element, 5: Mattress, 6: Water reservoir [32].

Most incubators use passive systems to supply humidification however this system have a bad quality of humidity control. It also cannot achieve a constant level of humidity [32]. In order to keep track of humidity levels within the incubator, a humidity control or monitor system should be applied [33]. This encourages more to use an Infant Incubator analyzer which will provide the accurate value of humidity within the incubator**.**

1.6. SOUND NOISE EFFECT ON NEONATAL BABIES

Infants generally are fragile human beings and noise is an important source of stress for them [34]. One of their basic needs is a peaceful environment. In NICU neonatal babies are exposed to high sound pressure levels (SPL) which disrupts normal growth by hindering the ability to stay in deep sleep [35]. Exposure to excessive noise has been associated with infants' heart rate acceleration, bradycardia, decline in oxygenation, increase in muscle tension, blood and intracranial pressure, sleep disturbance, agitation, and auditory disorders [36]. Recommendations for sound levels within NICU have stated a range from 50 dB to 90 dB. According to the World health organization (WHO) recommendations the noise in hospitals and medical facilities should be maximum of 30 dB [37]. The American interdisciplinary committee (AIC) recommends Lmax 65 dB [38]. However these values are acceptable for nursery but doesn't specify NICU limits. The American academy of pediatrics (AAP) recommended to lower the NICU monitoring devices and alarm volumes [39]. These recommendations also not tapping on the incubators, and urged for wearing soft shoes. The study made by (Y. Chang et al., 2006) monitored the environment sound level continuously before and after using a light alarm in the NICU. The study showed that sound level in NICU has great variations [36]. Reducing sound levels in the NICU has a big and important role to support neonatal development [34].

1.7. PREVENTIVE MAINTENANCE

Preventive maintenance (PM) plays a crucial role in insuring neonatal baby's safety and survival. Incubators should be inspected and tested, specifically their indicators and sensors, to ensure efficacy and safety of this device. PM mainly consists of a qualitative test and quantitative tests. The qualitative test covers the inspection of the equipment, including line cord, circuit breaker, heater, alarm, and power cord [40]. Considering the vital importance of incubators, they should be kept under continuous tracing, maintenance and calibration. Although preventive maintenance is a fundamental safety measure, flaws could occur due to various reasons specially in qualitative tests which is a critical issue effecting infants directly [41]. Thus, incubators PM should be supplemented with analyzers insuring these incubators are operating conveniently and measuring all parameters accurately. These analyzers test the incubator values and insure they are corresponding to and meet the regulations and standards that prevent hidden failures. Standing out the importance of infant incubators in treatment of neonates, it is necessary for them to keep functioning properly. It should be satisfactory to the IEC 601-2-19, IEC 601-2-20, and IEC601-2-21 standards [42]. Incubators analyzers can revel defects that would risk neonates.

1.8. INFANT INCUBATOR ANALYZER

Although measuring new-born babies' vital parameters are efficient, precise measurements are needed inside the incubator. If the measurements fail, the infant could be exposed to serios health complications or death. There are two types of infant incubator analyzers currently in used for PM in Turkey [2]. The first is INCU Incubator analyzer (Figure 1.5) manufactured by Fluke Corporation in the U.S. The second one is IncuTest manufactured by Datrend Systems Inc. (Figure 1.6) in Canada [2]. These two incubators work in accordance with IEC 601-2-19, IEC 601-2-20, and IEC601-2-21 standards which specifies safety and performance requirements for a baby incubators by the International Electrotechnical Commission. Basically, these analyzers measure temperature, humidity, sound noise, and air flow.

Figure 1.5: IncuTest Infant Incubator Analyzer.

Figure 1.6: INCU Infant Incubator Analyzer.

1.9. AIM OF THE STUDY

Although the Turkish medical devices industry has developed rapidly within the past decade in most fields [43], no analyzers of infant incubators have been manufactured in Turkey. In this study, we designed an infant incubator analyzer by using unsophisticated electronic components. This prototype incubator analyzer is a preliminary designed to be optimized with an ultimate aim to be manufactured by Turkish medical manufacturers in order to supply local medical facilities with an essential maintenance and calibration tool.

The design approach will be tested and comparable to other infant incubator analysts in order to obtain results for further optimization and development.

2. MATERIALS AND METHODS

As the infant incubator provides neonates with the appropriate parameters corresponding to their health needs we needed multiple types of sensors to measure these parameters. The block diagram in Figure 2.1 shows the principal parts of the device. All these sensors were interfaced with the AVR microcontroller where analog signals are converted to digital via the Atmega32 built-in ADC unit. The measured parameters were designed to be displayed on a graphical display interfaced with the AVR microcontroller.

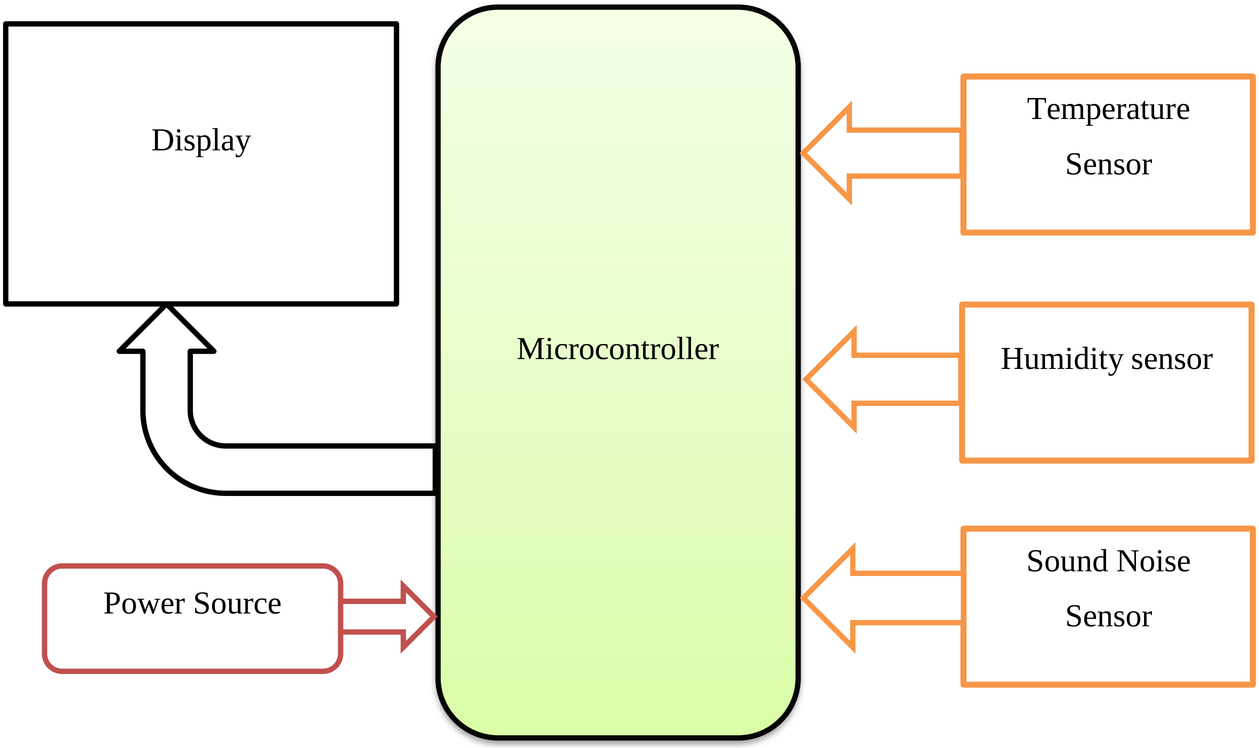

Figure 2.1: Infant incubator analyzer design block diagram.

Design Parameters

The parameters of this design are shown in Table 2.1. The design parameters where established to insure portability, efficiency and easy usage within all kind of infant incubators.

Table 2.1: Design parameters.

Design Specification

Hight	6 cm
Length	13 cm
Width	18 cm
Air temperature accuracy	± 1 ºC
Mattress temperature accuracy	± 1 ºC
Relative humidity accuracy	± 1 %
Sound noise accuracy	± 10 dB
Battery life	> 10 Hours
Display	128 × 64 GLCD

2.1. AVR MICROCONTROLLER

By delivering a low power consumption and high level of integration, we used AVR MCUs as they provide a unique combination of performance, power efficiency and design flexibility. They are also based on the industry most code-efficient architecture for C programming. The AVR is an 8-bit RISC single chip microcontroller which comes with some standard features such as on-chip code ROM, data RAM, data EEPROM, timers and I/O ports. In addition some AVR includes important features like internal ADC, PWM, USART, SPI, USB [44]. Due to the importance of the peripherals, we decided to use an AVR MCU, specifically the Amega32 Microcontroller.

Atmega32 is an 8-bit AVR RISC based on high performance and low power. It combines 32KB ISP flash memory with read-while-write capabilities including 1KB EEPROM, 2KB SRAM, 32 I/O lines, three flexible timers, internal and external interrupts, serial programmable USART, in addition to an 8-channel 10 bit ADC which is the most important feature for our study [45]. This allowed us to interface multiple sensors to the MCU. Currently considered about the 8-channel 10 bit resolution feature (Figure 2.2). The 8-channel implies that there are 8 ADC pins located across PORT A (from PA0 to PA7). The 10 bit resolution implies that there are 2^10 = 1024.

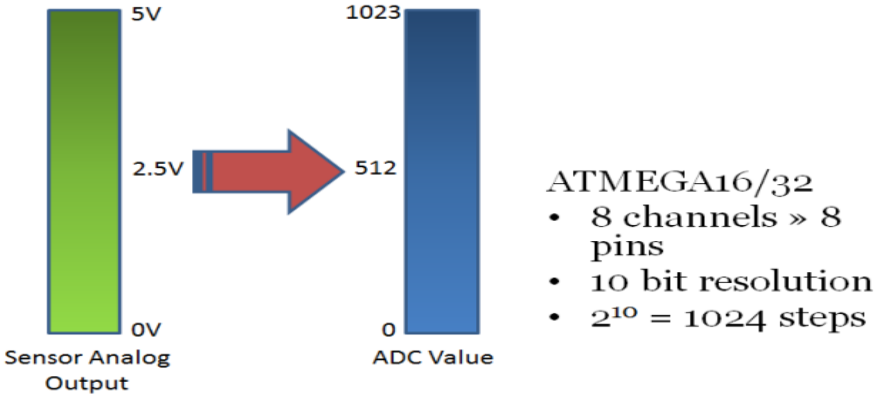

Figure 2.2: 8 channel 10 bit ADC [45].

In this study, the microcontroller receives input signals through ADC pins PA0 to PA3. In order to process these signals we had to set the ADC prescaler which is determined by the microcontroller clock frequency. Normally the operation scale of the ADC is somewhere between 50KHz to 200KHz. In this study the ADC frequency was set at 128KHz. Setting the ADC prescaler was implemented through the ADC multiplexer selection register (ADMUX). These bits were used to choose the reference voltage. We also set our reference voltage at Avcc with an external capacitor. A clear implementation in C as shown:

```
/*ADC Function Definitions*/
void adc_init(void)
{
    DDRA = 0x00;                                   /* Make ADC port as input */
    ADCSRA = 0x87;                                 /* Enable ADC, with freq/128  */
    ADMUX = 0x40;                                  /* Vref: Avcc */
}
```

2.2. DISPLAY

There are a lot of graphical LCDs that could be used. Generally they all function the same way with minor variations. Usually an alphanumeric display is used in prototypes. However in this study we needed more space to display the graphics and all the sensors readings simultaneously. We chose to use a 128×64 graphical LCD (Figure 2.3). GLCDs are different from the ordinary alphanumeric LCDs such as 16×2, 16×4, 20×4 etc. These ordinary LCDs only print numbers and letters in a single size while in the GLCD there are 128×64 = 8192 dots, and we can selectively lit any dot as desired. Therefore we can design the desired characters and pictures. Table 2.1 provides the detailed information of all the GLCD pins [46]

Table 2.2: All GLCD detailed pins info [46].

Pin Number	Symbol	Pin Function
1	VSS	Ground
2	VCC	+5v
3	VO	Contrast adjustment (VO)
4	RS/DI	Register Select/Data Instruction. 0:Instruction, 1: Data
5	R/W	Read/Write, R/W=0: Write & R/W=1: Read
6	EN	Enable. Falling edge triggered
7	D0	Data Bit 0
8	D1	Data Bit 1
9	D2	Data Bit 2
10	D3	Data Bit 3
11	D4	Data Bit 4
12	D5	Data Bit 5
13	D6	Data Bit 6
14	D7	Data Bit 7/Busy Flag
15	CS1	Chip Select for IC1/PAGE0
16	CS2	Chip Select for IC2/PAGE1
17	RST	Reset the LCD module
18	VEE	Negative voltage used along with Vcc for brightness control
15	A/LED+	Back-light Anode(+)
16	K/LED-	Back-Light Cathode(-)

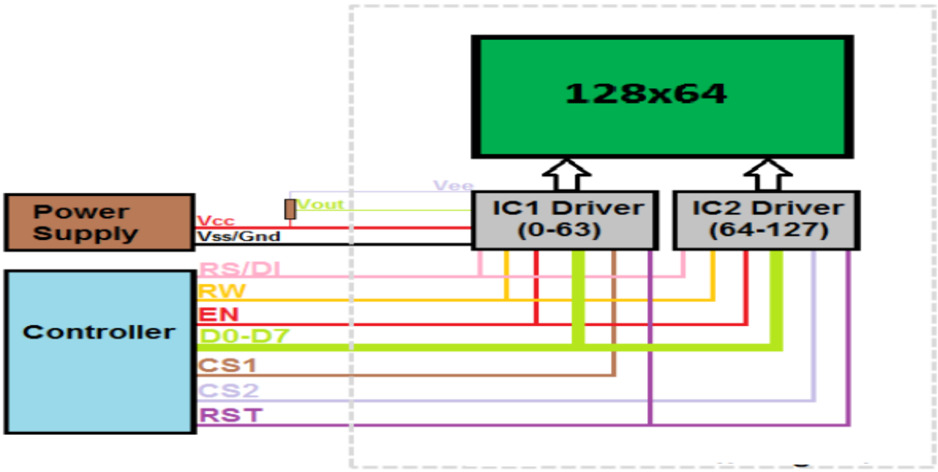

Figure 2.3: Internal block diagram of 128x64 GLCD along with its pin out [46].

To initialize the display we need to send display off command i.e. 0x3E, send Y address e.g. here 0x40 (Start address), send X address (Page) e.g. here 0xB8 (Page0), send Z address (Start line) e.g. here 0xC0 (from 0th line), and finally send display ON command i.e. 0x3F. Code implementation is as follows:

```
void GLCD_Init()                    /* GLCD initialize function */
{
    Data_Port_Dir = 0xFF;
    Command_Port_Dir = 0xFF;
    /* Select both left & right half of display & Keep reset pin high */
    Command_Port |= (1 << CS1) | (1 << CS2) | (1 << RST);
    _delay_ms(20);
    GLCD_Command(0x3E);             /* Display OFF */
    GLCD_Command(0x40);             /* Set Y address (column=0) */
    GLCD_Command(0xB8);             /* Set x address (page=0) */
    GLCD_Command(0xC0);             /* Set z address (start line=0) */
    GLCD_Command(0x3F);             /* Display ON */
}
```

To write commands we had to send commands on data pins. Then select the control registry by making the RS low (RS=0). This was followed by selecting the writing operation making RW low (RW = 0). Finally we send a high to low transitions through the E pin by enabling the E pin. Code implementation is as follows:

```c
void GLCD_Command(char Command)      /* GLCD command function */
{
    Data_Port = Command;             /* Copy command on data pin */
    Command_Port &= ~(1 << RS);      /* Make RS LOW to select command register */
    Command_Port &= ~(1 << RW);      /* Make RW LOW to select write operation */
    Command_Port |=  (1 << EN);      /* Make HIGH to LOW transition on Enable pin */
    _delay_us(5);
    Command_Port &= ~(1 << EN);
    _delay_us(5);
}
```

To write data we sent the character to GLCD. Then we selected the data registry and writing operation by making the RS and RW low (RS & RW=0). Then we sent a high to low transitions through the E pin by enabling the E pin. Code implementation in as follows:

```c
void GLCD_Data(char Data)            /* GLCD data function */
{
    Data_Port = Data;                /* Copy data on data pin */
    Command_Port |=  (1 << RS);      /* Make RS HIGH to select data register */
    Command_Port &= ~(1 << RW);      /* Make RW LOW to select write operation */
    Command_Port |=  (1 << EN);      /* Make HIGH to LOW transition on Enable pin */
    _delay_us(5);
    Command_Port &= ~(1 << EN);
    _delay_us(5);
}
```

Appendix shows the full code for display on GLCD.

2.3. LM35 TEMPERATURE SENSOR

Measurement of temperature is fundamental to insure infants safety. There are many sensors that can be employed as medical grade temperature sensors to help fit particular application. Typically, IC sensors are quit ideal for monitoring temperature at 37 °C (normal human body temperature). Therefore in this study we decided to use LM35 temperature sensor. The main reason of adopting an IC temperature sensor is its advantage over other sensors calibrated in Kelvin. The LM35 has a linearity proportion to the celsius (centigrade) temperature. Moreover, measurements will be more accurate then with a thermistor. The work range of LM35 is from -55 °C to 150 °C. The LM35 is easy to be included in a measurement application without any elaborate scaling schemes nor offset voltage subtraction [47]. Also it has a low cost which makes it a good choice for a prototype. LM35 circuit diagram is in Figure 2.4 [1].

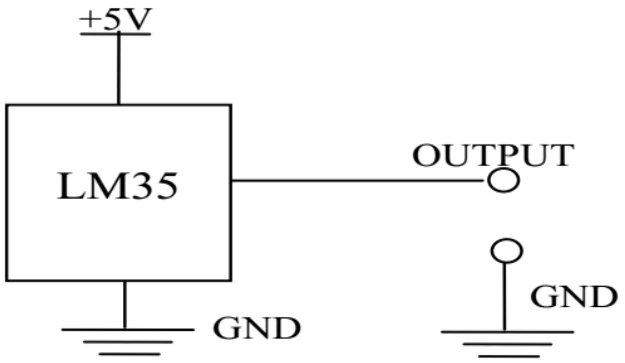

Figure 2.4: LM35 Circuit diagram [1].

LM35 provides a voltage output biased on the variation of temperature. As shown in Figure2.5 Each +1 °C is represented with a +10mV [48]. Since the output signal from LM35 is analog it was connected to one of the ADC pins of the microcontroller (PA0). The signal is converted to digital, then the digital value is converted to a centigrade value.

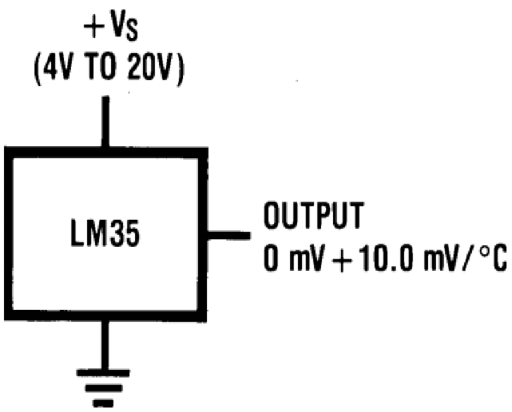

Figure 2.5: Centigrade temperature sensor [48].

Subsequently we initialized the ADC port by enabling the ADC feature then started the ADC conversion. Code implementation is as follows:

```
int read_adc(unsigned char channel)
{
    ADMUX = 0x40 | (channel & 0x07);        /* set input channel to read */
    ADCSRA |= (1<<ADSC);                    /* Start ADC conversion */
    while (!(ADCSRA & (1<<ADIF)));          /* Wait until end of conversion by polling ADC interrupt flag */
    ADCSRA |= (1<<ADIF);                    /* Clear interrupt flag */
    _delay_ms(1);                           /* Wait a little bit */
    return ADCW;                            /* Return ADC word */
}
```

The converted value is displayed on screen. Code implementation is as follows:

```
while(1)
{
    Lcd8_Cmd(0x80);
    Lcd8_Write_String("Temperature");
    celsius = (read_adc(0)*4.88);
    celsius = (celsius/10.00);
    sprintf(Temperature,"%d%cC  ", (int)celsius, degree_sysmbol);        /* convert integer value to ASCII string */
    Lcd8_Write_String(Temperature);                                       /* send string data for Display */
    _delay_ms(1000);
    memset(Temperature,0,10);
}
return 0;
```

2.4. DHT11 HUMIDITY / TEMPERATURE SENSOR

Humidity sensors are one of the most important devices used in biomedical applications for measuring and monitoring. Humidity is a measure of the moisture content of the air and can be measured as absolute humidity or relative humidity. Relative humidity is highly dependent upon air temperature as the higher the temperature the greater the capacity of the air to hold water vapor. Infants are in increased risk of high water loss through the skin leading to temperature instability, dehydration and electrolyte imbalance, as well as heat and calorie loss. Infants nursed without humidity frequently became hypothermic in spite of incubator air temperature. Increasing humidity to the surrounding environment can significantly reduce these from occurring.

There are multiple types of humidity sensors (Hygrometers) such as capacitive, resistive, and thermal conductivity humidity sensor. For the sake of our study we chose the resistive type, specifically DHT11. It contains a resistive humidity sensor in addition to an NTC temperature sensor. DHT11 measures relative humidity. The formula to calculate relative humidity is:

$$RH = \left(\frac{\rho\omega}{\rho s}\right) x\ 100\% \tag{2.1}$$

$RH: Relaitive\ Humidity$

$\rho\omega: Density\ of\ water\ vapor$

$\rho s: Density\ of\ water\ vapor\ at\ saturation$

This type of humidity sensors is made up with low resistive materials that changes significantly with the change of humidity (Figure 2.6). This assisted us to obtain our desired accuracy. Basically, this sensor consists of two conductive plates within a non-conductive base. The moisture from the air changes the voltage between the plates.

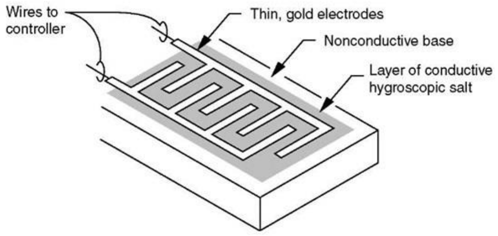

Figure 2.6: Resistive-type relative humidity sensor.

The built-in NTC negative temperature coefficient sensor is a variable resistor that changes according to the change in temperature. It is made of semi-conductive materials compressed to form a temperature sensitive conducting material. The electrical current flows through the charge carriers within the conductive material. High temperatures cause the semiconducting material to release more charge carriers and that is how temperature is measured.

The values from the sensor are read by the microcontroller and converted into a digital values considering the air temperature. DHT11 is a perfect and popular model. It is a cheap, sufficient, and reliable sensor [1]. It also has the ability to be connected to a 8-bit microcontroller. DHT11 digital temperature and humidity sensor is a composite sensor contains a calibrated digital signal output of the temperature and humidity [49].

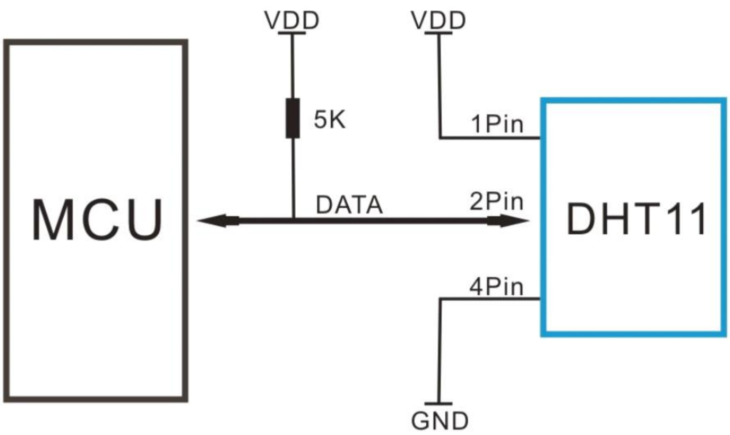

Figure 2.7: Typical application of DHT11 [49].

DHT11 uses one line for communication with microcontroller. Therefor to start the communication we needed to send a start pulse from the microcontroller to DHT11. That is implemented by pulling down the data pin for about 20ms then pulling up. Start pulse process code implementation is as follows:

```
void Request()                      /* Microcontroller send start pulse/request */
{
    DDRA |= (1<<DHT11_PIN);
    PORTA &= ~(1<<DHT11_PIN);       /* set to low pin */
    _delay_ms(20);                  /* wait for 20ms */
    PORTA |= (1<<DHT11_PIN);        /* set to high pin */
}
```

To indicate that DHT11 has received the start pulse from the microcontroller, DHT11 send a response pulse back to the microcontroller. This pulse should be low for 54µs then it goes high for 80µs. Response pulse code implementation is as follows:

```
void Response()                     /* receive response from DHT11 */
{
    DDRA &= ~(1<<DHT11_PIN);
    while(PINA & (1<<DHT11_PIN));
    while((PINA & (1<<DHT11_PIN))==0);
    while(PINA & (1<<DHT11_PIN));
}
```

After implementing the response pulse, DHT11 send the humidity and temperature values with checksum. The data frame is 5 byte, 8-bit each. Receiving data code implementation is as follows:

```
uint8_t Receive_data()                          /* receive data */
{
    for (int q=0; q<8; q++)
    {
        while((PINA & (1<<DHT11_PIN)) == 0);    /* check received bit 0 or 1 */
        _delay_us(30);
        if(PINA & (1<<DHT11_PIN))               /* if high pulse is greater than 30ms */
        c = (c<<1)|(0x01);                      /* then its logic HIGH */
        else                                    /* otherwise its logic LOW */
        c = (c<<1);
        while(PINA & (1<<DHT11_PIN));
    }
    return c;
}
```

The first two bytes (I_RH and A_RH) contains the humidity value in decimal which give the relative humidity value. The next two bytes (I_Temp and A_Temp) contains the temperature value also in decimal which give the temperature in celsius. The last byte contains the checksum which is basically a calculation using the algorithms provided in the programming specifications and communication protocol to determine the data integrity. Storing data code implementation is as follows:

```
while(1)
{
    Request();                          /* send start pulse */
    Response();                         /* receive response */
    I_RH=Receive_data();                /* store first eight bit in I_RH */
    A_RH=Receive_data();                /* store next eight bit in D_RH */
    I_Temp=Receive_data();              /* store next eight bit in I_Temp */
    A_Temp=Receive_data();              /* store next eight bit in D_Temp */
    CheckSum=Receive_data();            /* store next eight bit in CheckSum */

    if ((I_RH + A_RH + I_Temp + A_Temp) != CheckSum)
    {
        Lcd8_Cmd(0x80);
        Lcd8_Write_String("Error");
    }
```

Once the microcontroller completes receiving the data, DHT11 pin goes low until another start pulse is sent by the microcontroller. Overall communication process illustrated in Figure 2.8.

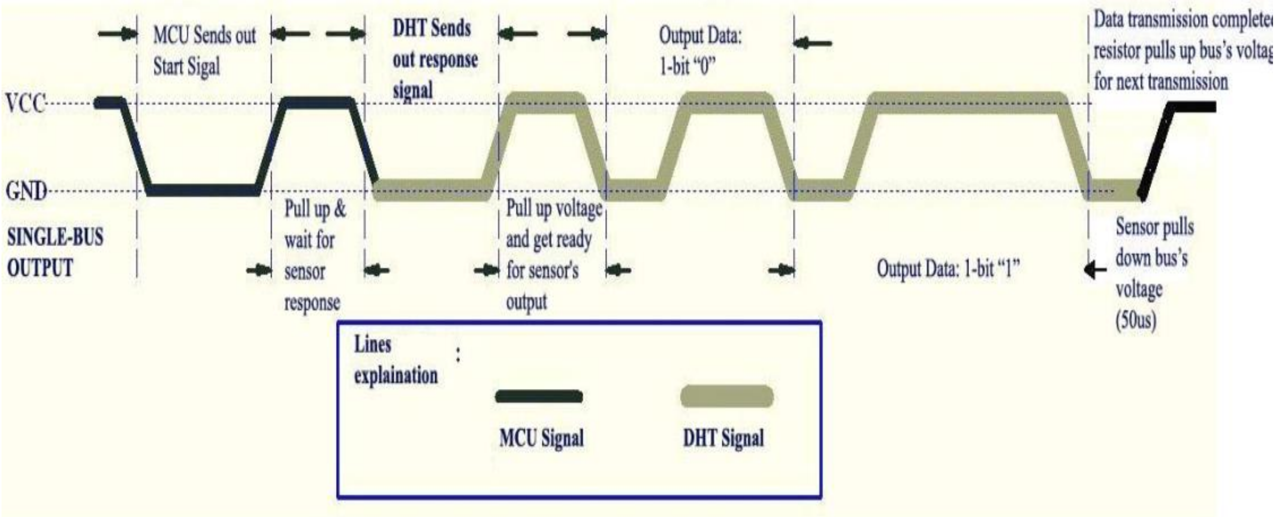

Figure 2.8: DHT11 overall communication process [49].

The stored value is displayed on screen. Code implementation is as follows:

```
{
    itoa(I_RH,data,10);
    Lcd8_Cmd(0x8b);
    Lcd8_Write_String(data);
    Lcd8_Write_String(".");

    itoa(A_RH,data,10);
    Lcd8_Write_String(data);
    Lcd8_Write_String("%");

    itoa(I_Temp,data,10);
    Lcd8_Cmd(0xC6);
    Lcd8_Write_String(data);
    Lcd8_Write_String(".");

    itoa(A_Temp,data,10);
    Lcd8_Write_String(data);
    Lcd8_Cmd(0xc9);
    Lcd8_Write_String("C ");

    itoa(CheckSum,data,10);
    Lcd8_Write_String(data);
    Lcd8_Write_String(" ");
}
```

2.5. SOUND NOISE SENSOR

Sound noise sensor is used to measure sound levels by measuring sound pressure. It is often called sound pressure level (SPL) meter. This sensor is a different because they are expensive and rare. Furthermore we can’t have an out of box sensor that could measure in decibels. In this study, we designed our own sound noise sensor using a regular DC based condenser microphone. Condenser microphones are best known for their sensitivity and wide range frequency. A condenser microphone is constructed of two parallel copastor plates (Figure 2.9). As the sound produce vibrations, the frontal plate (diaphragm) vibrates resulting a change in the distance between the copastor plates. This makes a change in the capacitance creating a change in the discharge current.

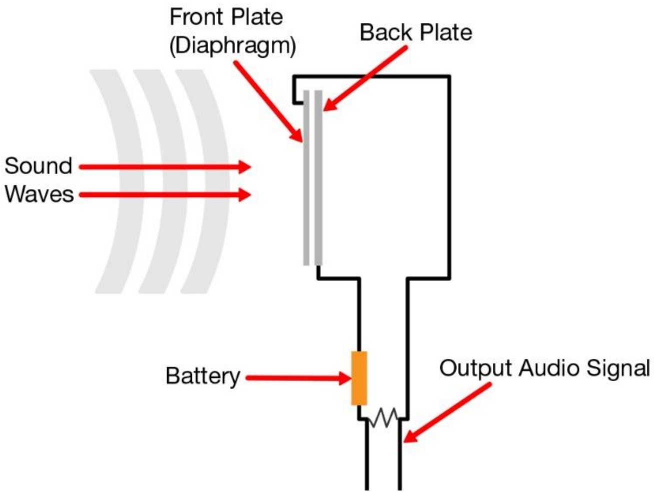

Figure 2.9: Condenser microphone diagram [50].

Amplifying the signal

Unlike temperature and humidity, measuring the sound noise is not a straightforward function. It is necessary to amplify the microphone output signal first to more easily detect it. Usually the microphone output electrical signals are faint or low, and they would not be detected by the ADC. Therefore we chose the LM386 low voltage audio power amplifier. This amplifier give us a gain from 20 to 200 [51]. Figure 2.10 shows the pin diagram of LM386 in order to explain the connections.

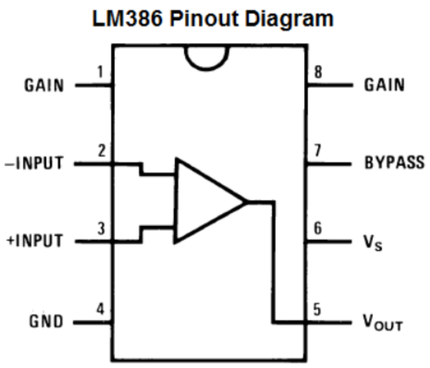

Figure 2.10: LM386 pin diagram [51].

Terminals 1 and 8 represent the gain control of the amplifier. These are the terminals where the gain can be adjusted by placing a resistor and capacitor or just capacitor between these terminals. In this circuit, we placed a 10µF capacitor between these terminals to obtain the highest voltage gain. Terminals 2 and 3 are the sound input signal terminals. In our case for this circuit, the condenser microphone was connected to these terminals. Terminal 2 is the -input and terminal 3 is the +input. In our circuit, the positive microphone terminal was placed on terminal 3 and terminal 2 was connected to the negative microphone terminal, tied to ground. Terminal 4 is GND (ground). This is where the negative voltage of the power source is connected to. Terminal 5 is the output of the amplifier. This is the terminal in which the amplified sound signal comes out. Terminal 6 is the terminal which receives the positive DC voltage so that the op amp can receive the power it needs to amplify signals. Terminal 7 is the bypass terminal. This pin is usually left open or is wired to ground. However, for better stability, a capacitor is added in our circuit because this can prevent oscillations in the amp chip. The schematic for the amplification circuit is shown in Figure 2.11.

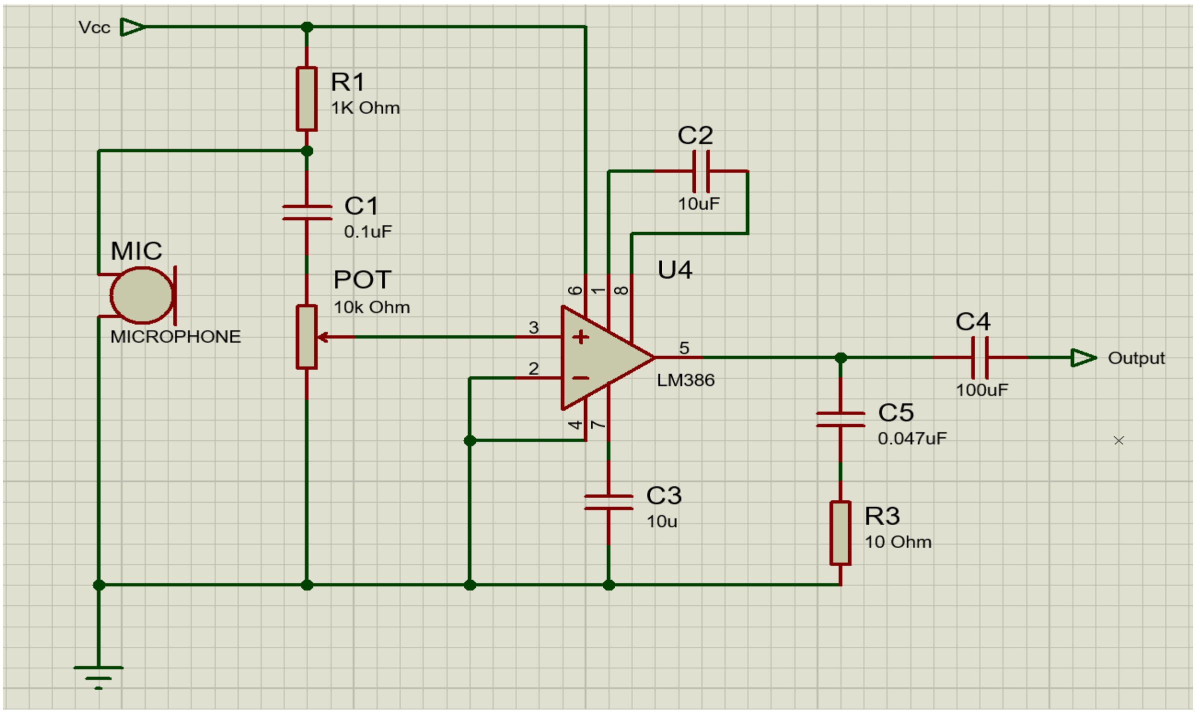

Figure 2.11: Schematic for the amplification circuit.

C1 (0.1µF): To block DC voltage on the input and allow AC signal which is the recived voice signal. **R1 (1KΩ):** Pull up resistor determined according to the L-KLS3-MH-HM9767P datasheet. **R2 (10KΩ POT):** Controls the input volume. **C2 (10µF):** Sets the voltage gain of the amplifier LM386. It provides us with a maximum gain of 200. **C3 (10µF):** Improves the amplifier LM386. **C4 (100µF):** Removes any DC offset from the amplifier LM386 output. **C5 (0.047µF):** Current bank for the output, it discharges whenever there is a current need. The condenser microphone receives the sound signal and convert it into an electrical signal.

This signal is inputted via the pin 5 of the LM385. This signal is amplified by the by the LM386. It is input to the Microcontroller ADC pin to be converted into a digital value.

Reading the signal through the microcontroller

Once the signal is ready to be read by the microcontroller, it is input into the ADC port in the Atmega32 to be converted to a digital value then to a dB value. As we mentioned before we cannot get a common multiplier or the ADC values and dB. ADC code implementation is as follows:

```
int read_adc(unsigned char channel)
{
    ADMUX = 0x40 | (channel & 0x07);           /* set input channel to read */
    ADCSRA |= (1<<ADSC);                        /* Start ADC conversion */
    while (!(ADCSRA & (1<<ADIF)));              /* Wait until end of conversion by polling ADC interrupt flag */
    ADCSRA |= (1<<ADIF);                        /* Clear interrupt flag */
    _delay_ms(1);                               /* Wait a little bit */
    return ADCW;                                /* Return ADC word */
}
```

In this study we have designed our own sound noise sensor, so we are satisfied with an aproximate accuracy of dB values as long as we are designing a preliminary prototype. We used an android application called sound meter as a reference to obtain an equivalent dB value to the ADC values. Table 2.2 shows the equivalent values.

Table 2.3: 10 ADC values and its equivalent in dB.

dB Value	ADC Value
30	360
35	485
40	410
45	470
50	490
55	500
60	507
62	540
65	600
70	615

After getting these values we were able to form the following equation:

$$ADC = (11.005 \times dB) - 83.2075 \qquad (2.2)$$

Which can be implanted in our programming code in the following equation:

$$dB = (ADC + 83.2075) \div 11.005 \qquad (2.3)$$

Code implementation is as follows:

```
    int dB, Sound;

/*Start of infinite loop*/
while(1)
{
    Lcd8_Cmd(0x80);
    Lcd8_Write_String("Sound");
    dB = (read_adc(0));                          /* Reading the ADC value*/
    Sound=(dB+83.2075)/11.005                    /* Converting the ADC value in dB*/
    Lcd8_Cmd(0x8d);
    Lcd8_number_write(Sound,10);                  /* send string data for printing */
    Lcd8_Write_String("dB   ");
    _delay_ms(200);

}
return ;
}
/*End of Program*/
```

3. RESULTS

3.1. LM35 TEMPERATURE SENSOR

We have implemented a simulation for temperature measurement and its value display on proteus as shown in Figure 3.1. We ran our source code through the simulator and the reading was observed through the LCD. The readings was accurate.

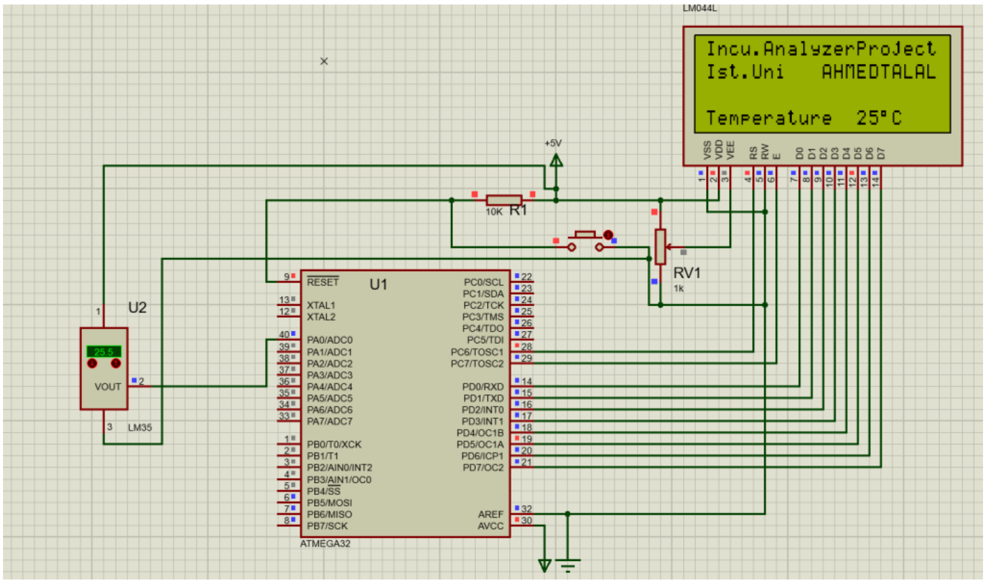

Figure 3.1:LM35 sensor simulation circuitry on Proteus.

We implemented the circuit on a testing board using a 20×4 LCD. Then we burned our source code into the Atmega32 using USBasp. We measured human body temperature by holding the LM35 sensor (Figure 3.2). A measurement result of 37 ^{o}C was observed through the LCD which is the normal temperature for a human being.

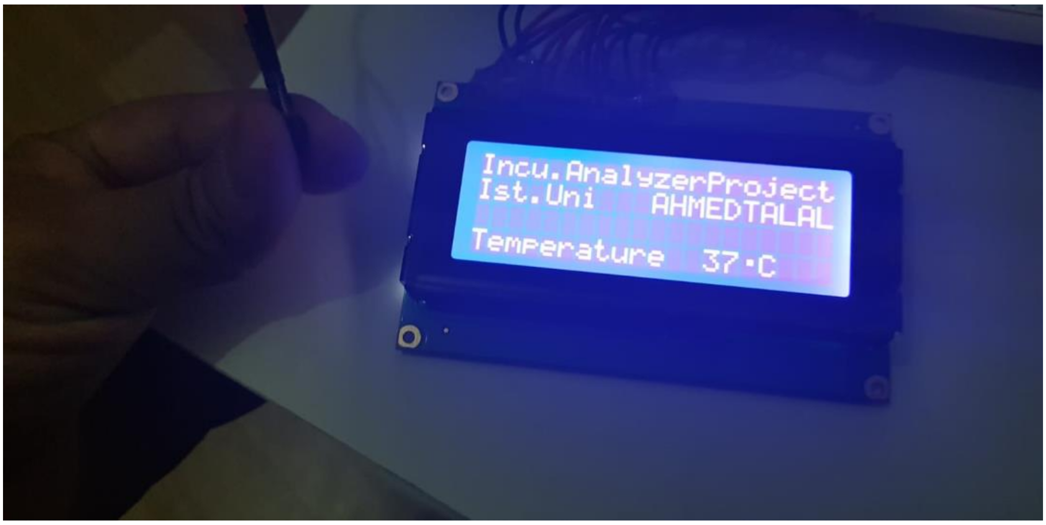

Figure 3.2: Measuring body temperature through LM35.

3.2. DHT11 TEMPERATURE/HUMIDITY SENSOR

DHT11 sensor circuitry has been implemented on Proteus. We ran our source code on the simulator and an accurate reading was observed trough the LCD (Figure 3.3).

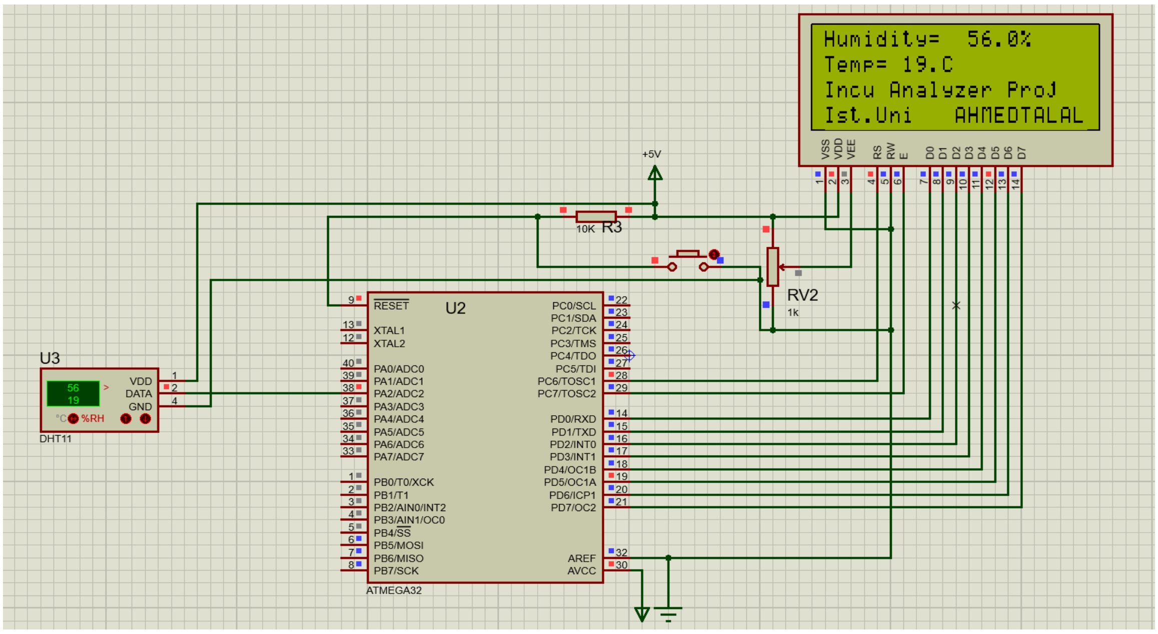

Figure 3.3: DHT11 sensor simulation circuitry on Proteus.

We implemented the circuit on a testing board using a 20×4 LCD. Then we burned our source code into the Atmega32. A measurement reading was obtained of 56.0% for humidity and 27 ºC Temperature (Figure 3.4) knowing that these measurements were taken in a warm room.

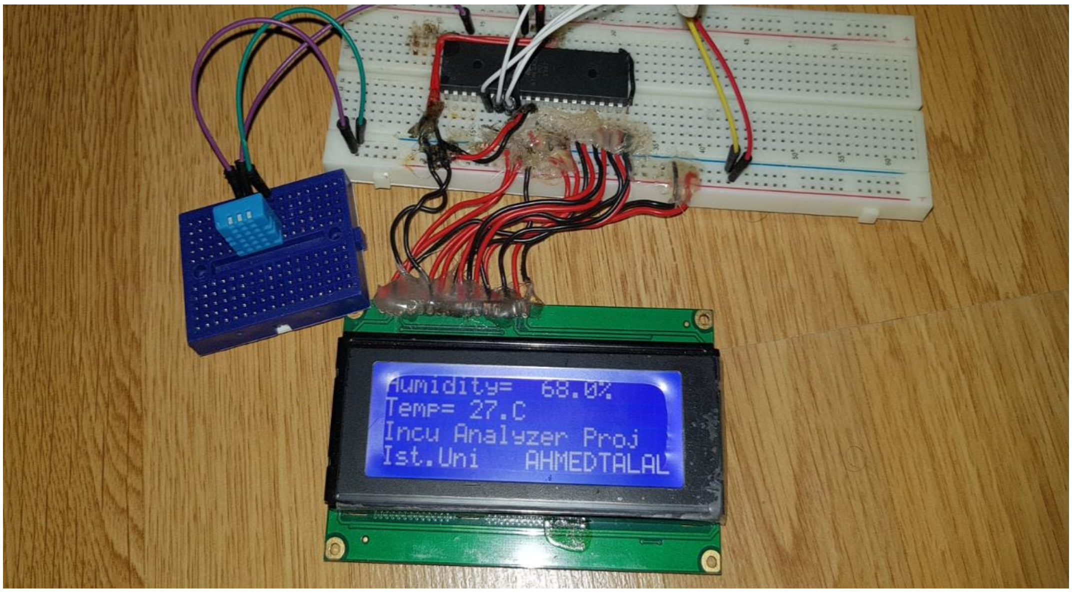

Figure 3.4: Measuring room humidity and temperature with DHT11.

3.3. SOUND NOISE SENSOR

The designed sensor was implemented on Proteus. We were unable to get an actual simulation because of the simulator`s inability to provide any audio input. The designed circuitry is shown in Figure 3.5.

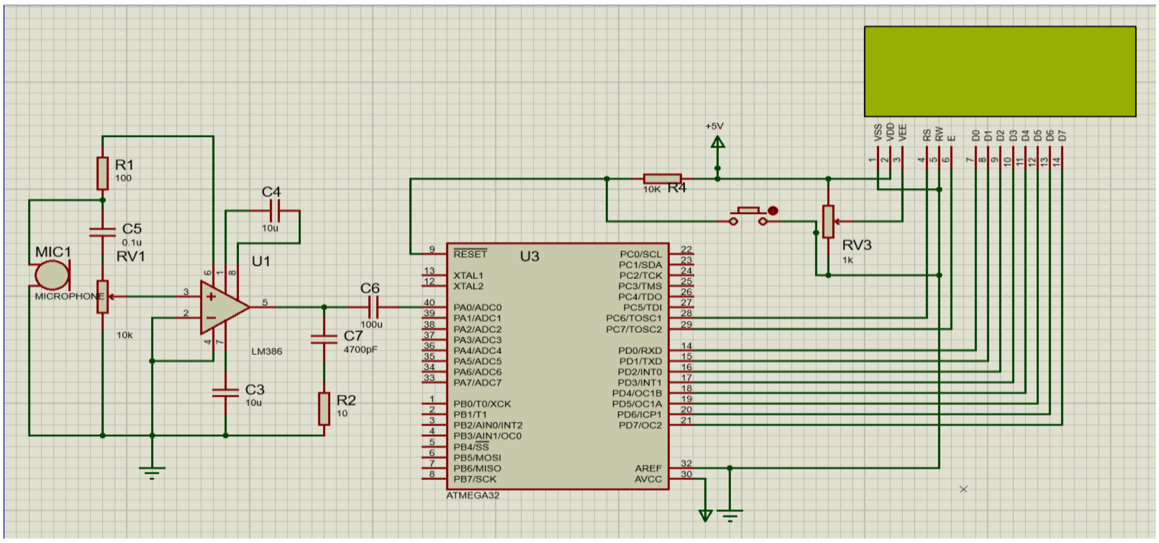

Figure 3.5: Sound noise sensor circuitry.

The designed circuit was tested and calibrated using an android application and the obtained readings were. We first tested the sensor in a quiet room and the obtained reading was 37 dB as compared to 39 dB on the android application (Figure 3.6).

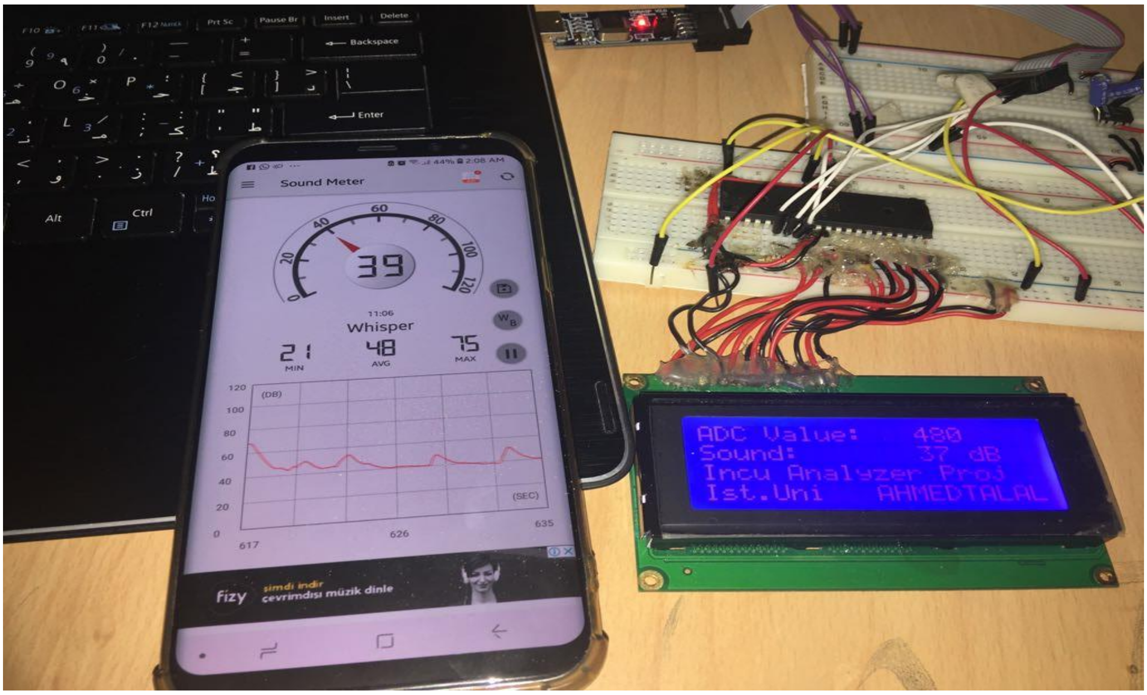

Figure 3.6: Sound sensor obtained result in a quiet room.

Then we tested the sensor with some music turned on to check the sensor accuracy and the reading obtained was 69 dB as compared to 73 dB on the android application (Figure 3.7).

Figure 3.7: Sound sensor obtained result with music turned on.

3.4. GLCD DISPLAY

We tested our graphical display code on Proteus (Figure 3.8) and we obtained identical results after implementing the circuit on a test board (Figure 3.9). Thus desired results were achieved.

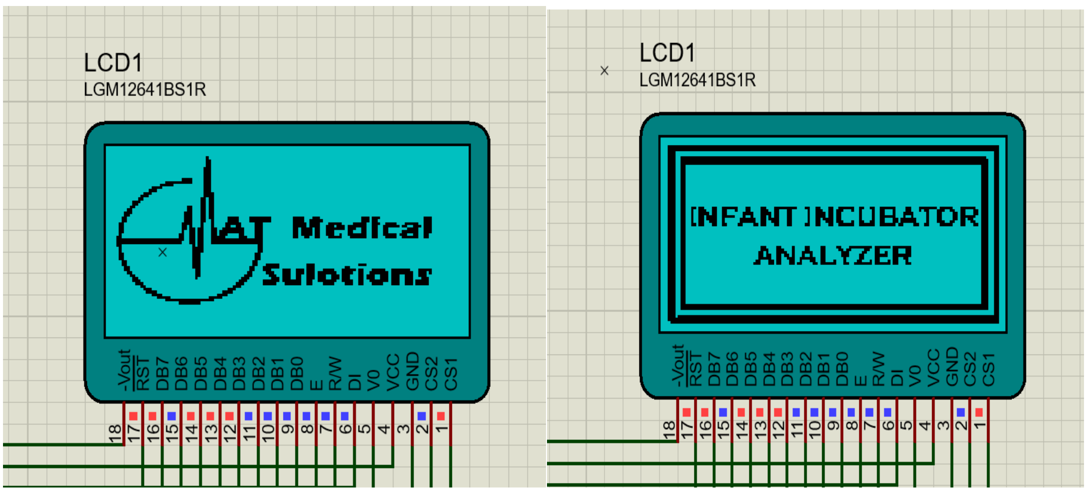

Figure 3.8: Graphical display code test on Proteus.

Figure 3.9: Graphical display code results on testing board.

3.5. FINAL IMPLEMENTATION

We installed all sensors together in one circuit and preformed the appropriate adjustments to the source code in order for all these components to function simultaneously in harmony. We implemented a simulation on Proteus to test the circuitry and the code (Figure 3.10). The code and circuit were found to be working properly.

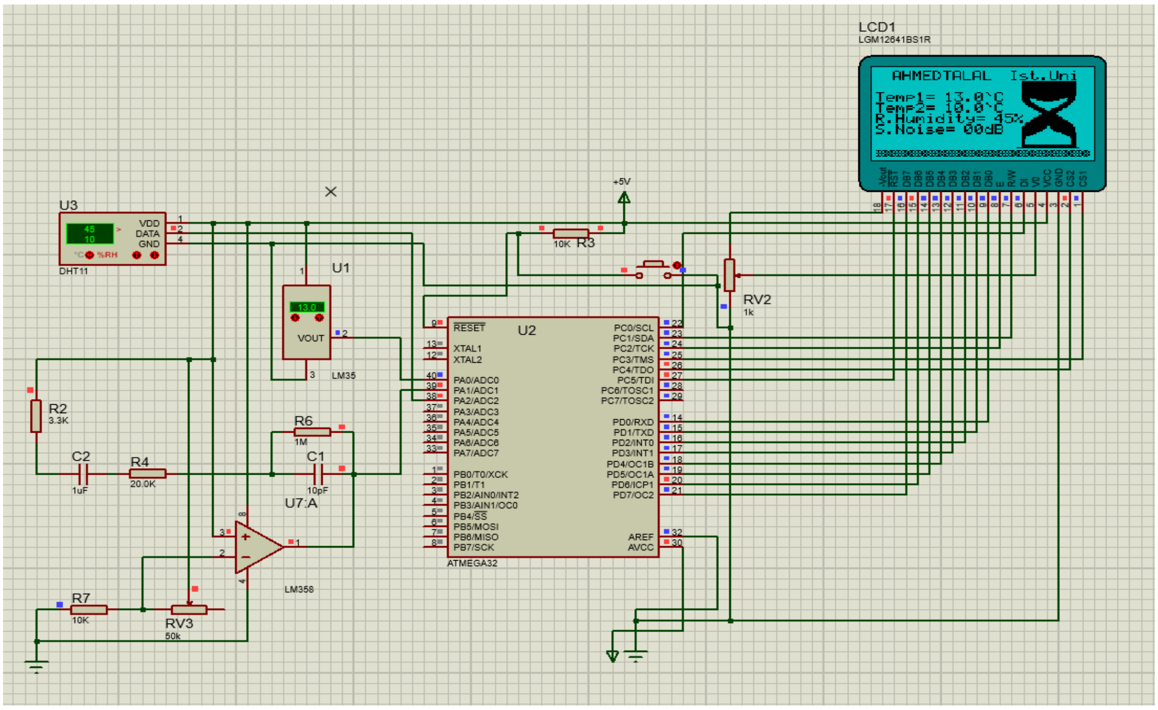

Figure 3.10: All components circuitry simulation on Proteus.

We implemented the circuit on a testing board and burned our source code into the Atmega32 (Figure 3.11). The circuit and code were found to be working properly.

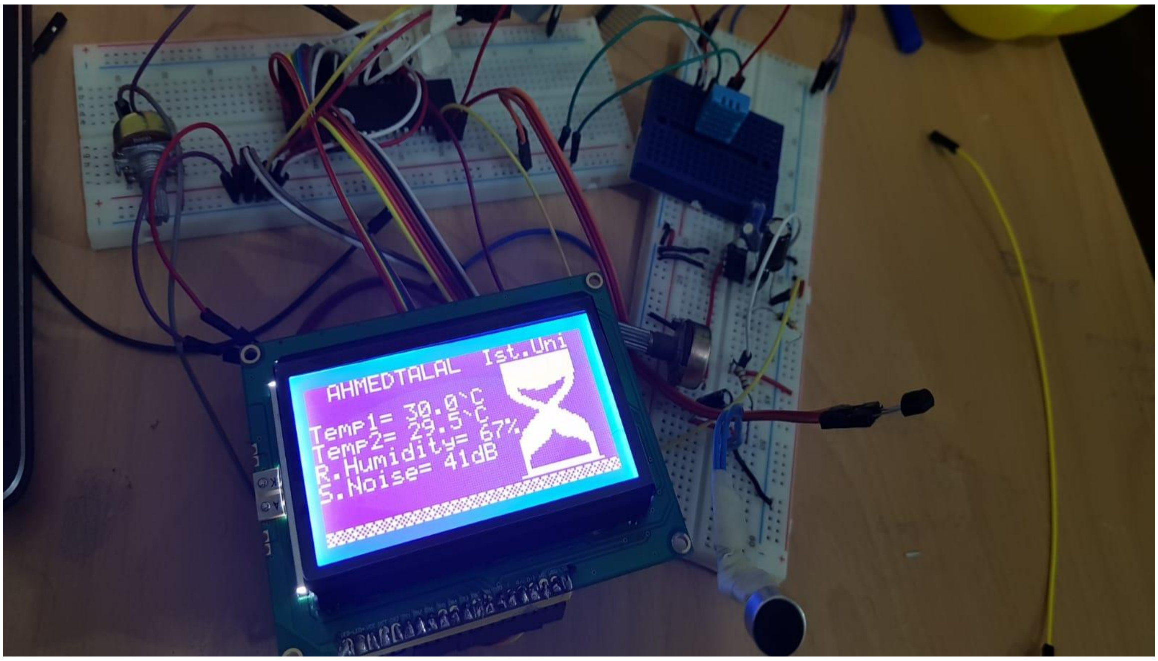

Figure 3.11: Circuit implemented on a test board.

After final emplimentation and testing the device was inclosed and put together in a boxing unit shown in Figure 3.12.

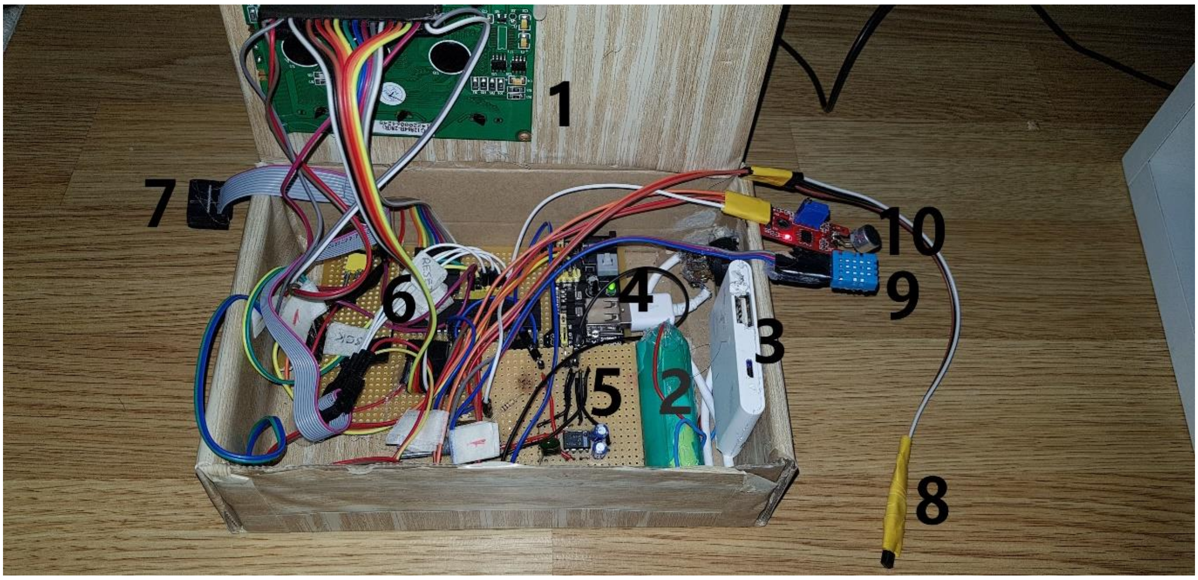

Figure 3.12: Enclosed device 1: Display, 2: Battery, 3: Charging unit, 4: Power suply unit, 5: Sound noise sensor circuit, 6: Microcontroller unit circuit, 7: Software progrming port, 8: Mattress tempreature probe, 9: Humidity/Air temperature sensor, 10: Sound noise sensor.

3.6. TESTING THE UNIT

The designed approach was tested and compered to other infant incubator analyzers. We tested our device in a Drager Isolette® Infant Incubator (Model C2000) at the same time with a Datrend IncuTest incubator Tester (Figure 3.13). Data in Table 3.1 shows the obtained results . The Drager Isolette® does not provide a mattress heating mechanism.

Figure 3.13: Testing our device in a Drager Isolette® Infant Incubator with a Datrend IncuTest incubator Tester.

Table 3.1: Results obtained from testing our device in a Drager Isolette® Infant Incubator with a Datrend IncuTest incubator Tester.

	Drager Isolette® Infant Incubator	Datrend InuTest	Our Incubator Analyzer		
			Air Temp	Mattress Temp	Relative Humidity
Temperature	29.0 ºC	28.73 ºC	28 ºC	26 ºC	64.0%
	30.5 ºC	29.81 ºC	29 ºC	27 ºC	58.0%
	31.5 ºC	31.29 ºC	30 ºC	27 ºC	54.0%
	33.0 ºC	32.32 ºC	31 ºC	28 ºC	50.0%
	34.4 ºC	33.29 ºC	34 ºC	30 ºC	44.0%
	35.2 ºC	34.65 ºC	35 ºC	31 ºC	39.0%
Sound Noise		41.2 dB	39.0 dB		

In the second test we tested our device in a Girffe OmniBed Incubator with a INCU Fluke infant incubator analyzer (Figure 3.14). Data in Table 3.2 shows the obtained results.

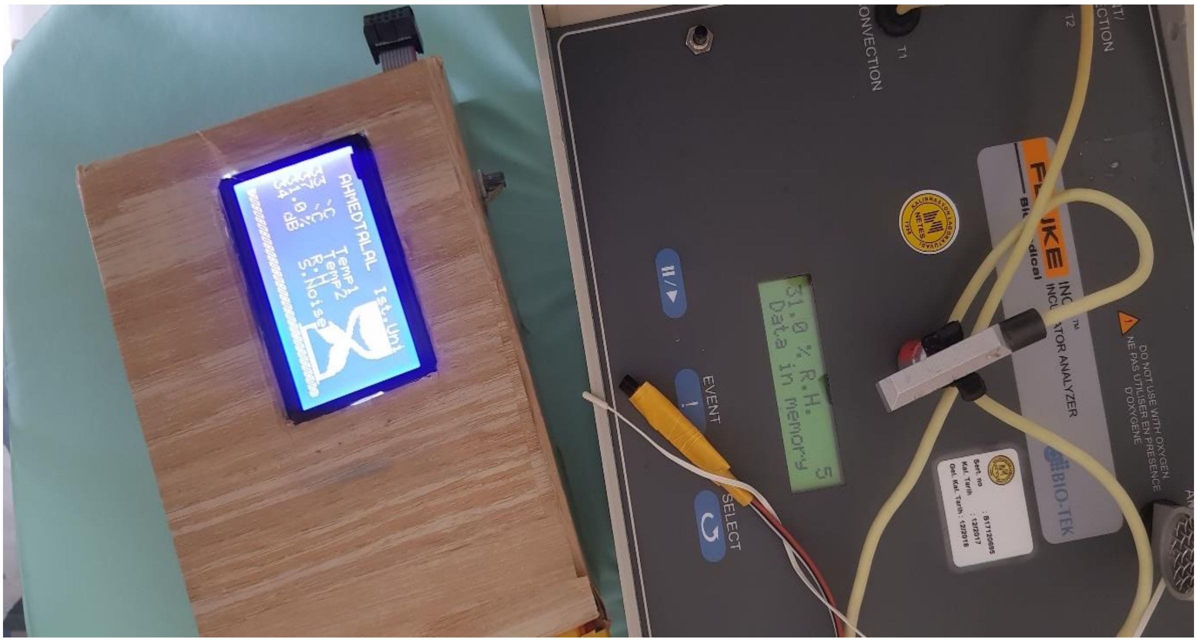

Figure 3.14: Testing our device in Girffe OmniBed Infant Incubator with a INCU Fluke incubator analyzer.

Table 3.2: Results obtained from testing our device in Girffe OmniBed Infant Incubator with a INCU Fluke incubator analyzer.

	Girffe OmniBed Incubator		Fluke INCU Incubator Analyzer		Our Incubator Analyzer	
	Mattress	Air	Mattress	Air	Mattress	Air
Temperature	36.5 ºC	36.6 ºC	35.3 ºC	37.2 ºC	35 ºC	36 ºC
	36.9 ºC	36.8 ºC	36.4 ºC	37.8 ºC	36 ºC	36 ºC
	37.4 ºC	37.7 ºC	36.8 ºC	38.2 ºC	37 ºC	37 ºC
Relative Humidity			31.0%		31%	
			31.2%		31%	
			31.3%		32%	
Sound Noise			61.6 dB		40 Db	
			67.2 dB		41 dB	

4. DISCUSSION

These analyzers are supposed to check the current parameters value which corresponds to safety standards. In this study, we investigated the major parameters that effect neonates health directly and their safety standards. We also studied the possible ways to monitor and measure these parameters which are produced by the infant incubator. This study have focused on using unsophisticated electronic sensors and components for the design.

We used LM35 and DHT11 sensor to obtain temperature values. Both sensors provided accurate measurements. LM35 shown to have a greater response time and accuracy in conduction measurements. DHT11 temperature readings had a fast response time as well as better sensing response to convection and radiant temperature. These particularities get along with our aspirations to use LM35 to obtain temperature from the incubator mattress, and to use DHT11 as a measuring tool for the air temperature inside the infant incubator chamber.

Humidity measurement was also made by the DHT11 sensor. Despite that this sensor is no use below 20% or above 90%, however recommendations say that the perfect humidity for neonates is a level between 30% and 50%. This makes the DHT11 appropriate for our use. Humidity readings we obtained from DHT11 were accurate as desired.

With no specific sound noise sensors in the market are available for such uses, we designed our own sensor using a condenser microphone and an LM386 amplifier. Due to the nonlinearity between the ADC value and the dB, we were unable to obtain a common constant multiplier for all ADC values. The obtained readings from our designed sensor were compared to an android SPL application readings. We have obtained a nearly accurate measurements with an error rate up to ± 10 dB. The readings obtained from our sensor were convergent to those readings from the android application. Further advanced optimization to this sensor is recommended.

The result test, presented in Table 2.4 and Table 2.5 has evidenced a minor lack of accuracy and response regarding temperature measurments compeared to the other analyzers speacially in high temperatures. Initially we attribute that to the amount of sensors in other analyzers where multiple (around four) temperatures sensors measure the temperature simutanusly. Also the design of these sensors in the other analyzers are designed to operate for such specific application. However this deficiency could be improved by some software optimazation. Relative humidity mesurments were acurate compeared to the Fluke INCU analyzer. The Datrend Incutest analyzer had a problem with it’s humidity sensor, therefore we were not able to compere the relative humidity results. Sound noise measurments had a problem and a major lack of accuracy specially with high measurments. During the test, our sensor demonsrated some impracticality which could be resulted from the hardware design. The error ratio exceeded ± 20 dB. However this also could be improved by further software optimization.

5. CONCLUSION AND RECOMMENDATIONS

In conclusion, this study details a fundamental design approach for an infant incubator analyzer that can be manufactured with low cost in Turkish local manufactures. The proposed design is a preliminary prototype that needs further optimization and testing. A major limitation in this study should be recognized. We were unable to design a highly advanced sound noise sensor due to limitation of time and resources. More efforts needed to develop highly advanced and accurate sound sensor suitable for this particular application which involves neonate's safety. Design recommendations generated from this study are intended to influence future designs. Major aspects to be considered encompass airflow as well as other measurements that are not available in existing infant incubator analyzers which are very important for the neonate's safety. These are oxygen saturation and light intensity measurements. Moreover, future designs should consider accordance to the IEC 601-2-19, IEC 601-2-20, and IEC601-2-21 standards.

REFERENCES

[1]. Bansal, H,. Mathew, L., Gupta, A., June 2015, *Controlling of Temperature and Humidity for an infant incubator using Microcontroller*, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 4.

[2]. Özdemđrcđ, E., Yatak, M., Duran, F., Canal, M., 2014, *Reliability assessments of infant incubator and the analyzer*, Gazi University Journal of Science, 27(4):1169-1175, 2014.

[3]. Daniel, R., Vecchione, D., Foley, R., Krishnan, S., Zenouzi, M., *Design a low cost neonatal incubator*, Electronics and Mechanical Department, Wentworth Institute of Technology, Boston

[4]. Fransson, A., Karlsson, H., Nilsson, K., May 2018, *Temperature variation in new-born babies: importance of physical contact with the mother*, group.bmj.com.

[5]. Dhatrak, V., Gholap, R., Patil, S., Bhaldar, N., Mhetre, M., Apr 2014, *Intelligent baby incubator using labview*, 2nd International Conference On Emerging Trends in Engineering & Techno-Sciences (ETETS), ISBN:978-93-81693-687.

[6]. Erdem, Y., Topuz, S., Güneş, N., 2014, *Progress related to maternal and neonatal health in turkey*, Erdem Y, J Neonatal Biol, 3:2.

[7]. Perez, JMR., Perez FR., Golombek, A., November 2017, *Comparative trial between neonatal intensive Care incubator, neonatal laminar flow unit and radiant warmer*, International Neonatal Neurodevelopment Center, Brazil, ISSN: 2576-9200.

[8]. İpek, M., Özbek, E., 2016, *Bloodstream infections in a neonatal intensive care unit*, J Clin Anal Med;7(5): 625-9.

[9]. Wafaa, A., Supervisor Karrar, A., *Development of baby incubator*, Thesis (Phd), University of Khartoum.

[10]. Soler, C., Supervisor: Abbas, A., November 2009, *Prototyping a closed loop control system for a neonatal incubator*, Helmohlotz - Institute for Biomedical Engineering , RWTH Aachen University.

[11]. Baker, J., 2000, *The incubator and the medical discovery of the premature infant*, Journal of Perinatology, 5:321–328.

[12]. Gartner, L., Gartner, C., Gluck, L., Butterfield, J., Stahlman, M., Duxbury, M., Adams, L., Little, G., Sunshine, P., Alexander, D., October 1992, *Neonatal intensive care a history of excellence*, U.S. Department Of Health And Human Services, NIH Publication No. 92-2786.

[13]. Baker, JP., *The incubator and the medical discovery of the premature infant*, J Perinatol; 5: 321–328.

[14]. Ferris, TK., and Shepley, M., 2013, *The design of neonatal incubators: a systems-oriented, human-centered approach*, Journal of Perinatology, 33, S24–S31.

[15]. Antonucci, R., Porcella, A., Fanos ,V., 2009, *The infant incubator in the neonatal intensive care unit: unresolved issues and future developments*, J Perinat Med, 37: 587–598.

[16]. Health Statistics Yearbook, 2012, *Birth/Mortality rate of Neonatal in 2011-2012*, Turkey Statistical Institute.

[17]. Alparslan, Ö., and Demırel, Y., 2013, *Traditional neonatal care practices in Turkey*, Japan Journal of Nursing Science, 10, 47–54.

[18]. Uslu, S., Yuksel, A., Uslu, A., Turan, B., Bulbul, A., Egici, M.T., Albayrak, S., Bektemur, G., November 2016, *Neonatal Intensive Care Units in Istanbul (2014-2015)*, The Medical Bulletin of Şişli Etfal Hospital, Volume: 50.

[19]. Demirel, G., and Dilmen, U., 2011, *Success of decreasing neonatal mortality in Turkey*, Medical Journal of Islamic World Academy of Sciences 19:4, 161-164.

[20]. Frischer, R., Penhaker, M., Krejcar, O., Kacerovsky, M., Selamat, A., 2014, *Precise temperature measurement for increasing the survival of new-born babies in ıncubator environments*, Sensors, 14, 23563-23580; doi:10.3390/s141223563.

[21]. Ashford and St. peter`s Hospital, *Incubator Humidity for Premature Neonates*, Neonatal Intensive Care Unit Clinical Guideline.

[22]. Fransson, A.L., Karlsson, H., Nilsson, K., May 2018, *Temperature variation in new-born babies: importance of physical contact with the mother*, group.bmj.com.

[23]. Dhatrak, V., Holap, R., Patil, S., Haldar, N., Mhetre, M., April 2014, *Intelligent baby incubator using labview*, 2nd International Conference On Emerging Trends in Engineering & Techno-Sciences (ETETS), ISBN: 978-93-81693-68-7.

[24]. Tuffnell, C. S., Petersen, S.A., Wailoo, M.P, 1995, *Factors affecting rectal temperature in infancy*, Archives of Disease in Childhood, 73: 443-446.

[25]. Agate, F. J., and Silverman, W. A., 1963, *The control of body temperature in the small new-born infant by low-energy infra-red radiation*, Pediatrics, 31, 725.

[26]. Buetow, K., and Klein, S.W., August 1964, *Effect of maintenance of "normal" skin temperature on survival of infants of low birth weight*, Paediatrics.

[27]. Perlsten, P. H., Edwards, N. K., Atherton, H. D., Sutherland, J. M., 1976, *Computer-assisted new-born intensive care*, ped. 57, p. 494.

[28]. Ferris, T.K., and Shepley, M.M., 2013, *The design of neonatal incubators: A systems-oriented, human-centered approach*, Journal of Perinatology (2013) 33, S24–S31; doi:10.1038/jp.2013.11.

[29]. Betts, K., March 2011, *The role of diesel particulate matter in lung ınflammation*, Environmental Health Perspectives, volume 119, number 3.

[30]. Hey, E., and Maurice, N., 1968, E*ffect of humidity on production and loss of heat in the newborn baby,* Medical Research Council Research Group on Respiration and energy Metabolism in the Newborn, The London Hospital Medical College, London E.1 Arch. Dis. Childh., 1968, 43, 166.

[31]. Bouattoura, D., Villon, P., Farges, G., January 1998, *Dynamic programming approach for new-born's incubator humidity control*, IEEE Transactions on Biomedical Engineering, Vol. 45, No. 1.

[32]. Costa, E., Freire, R., Silva, J., Cursino, C., Oliveira, C., Pereira, B., Silva, R., 2009, H*umidity control system in new-born incubator,* ISBN 978-963-88410-0-1, IMEKO.

[33]. Abdiche, M., Farges, G., Delanaud, S., Bach, V., Villon, P., Libert, J., 1998, *Humidity control tool for neonatal incubator*,' Med. Biol. Eng. Comput., 36, 241-245.

[34]. Knutson, A., May 2013, A*cceptable noise levels for neonates in the neonatal intensive care unit*,' Doctor of Audiology Report Submitted to Washington University School of Medicine.

[35]. Nogueira, M., Di Piero, K., Ramos, E., De Souza, M., Dutra, M., February 2011, *Noise measurement in NICUs and incubators with new-borns a systematic literature review*, Rev. Latino-Am. Enfermagem, 19(1):212-21.

[36]. Chang, Y., Pan, Y., Lin, YJ., Chang, YZ., Lin, CH., 2006, *A noise-sensor light alarm reduces noise in the new-born ıntensive care unit*, DOI 10.1055/s-2006-941455. ISSN 0735-1631.

[37]. World Health Organization (WHO), 1999, G*uidelines for community noise*. Noise sources and their measurement. [cited].

[38]. Committee to Establish Recommended Standards for New-born ICU Design, 2007, *Report of seventh census conference on new-born ICU design*. [cited]

[39]. American Academy of Paediatrics, Joint Committee on Infant Hearing, 2007, *Position statement: Principles and guidelines for early hearing detection and intervention programs*, Paediatrics. 2007;120(4):898-921.

[40]. Ibrahim, F., Ding, J., Taib, M., Babu, P., 2002, S*afty and performance compliance test of an infant incubator*, University of Malaya, Malaysia, 2002 Student Conference on Research and Development Proceedings, IEEE, 2002.

[41]. ECRI, 1995, *Inspection and preventive maintenance system*. A Nonprofit Agency, USA.

[42]. Ministry of Economy, *Medical devices and supply industry*, Republic of Turkey.

[43]. Barrett, S., and Pack, D., 2008, *Atmel AVR microcontroller primer: Programming and interfacing*, Morgan & Claypool Publishers series.

[44]. Mayanak (MAX), June 2011, *The ADC of AVR*, Retrieved from: https://maxembedded.wordpress.com/2011/06/20/the-adc-of-the-avr/comment-page-4/

[45]. Explore Embedded, Retrived from: https://exploreembedded.com/wiki/Interfacing_glcd_with_8051

[46]. Suruthi, M., Suma, S., *Microcontroller based baby incubator using sensors*, International Journal of Innovative Research in Science, Engineering and Technology. ISSN(Online): 2319-8753.

[47]. Texas Instruments, August 2016, *LM35 precision centigrade temperature sensors*, Dallas, Texas 75265 Copyright © 2017, Texas Instruments Incorporated.

[48]. D-Robotics UK, July 2010, *DHT11 humidity & temperature sensor*, D-Robotics UK (www.droboticsonline.com).

[49]. MXL Microphones, *How condenser microphones work*, Retrieved from: http://www.mxlmics.com/blog/?p=334

[50]. Texas Instruments, May 2017, *LM386 low voltage audio power amplifier*, Dallas, Texas 75265 Copyright © 2017, Texas Instruments Incorporated

APPENDICES

Appendix 1: GLCD code.

```
/*Infaant Incubator Analyzer.c
*
* ENG. AHMED TALAL
* MASTER DEGREE THESIS
* BIOMEDICAL ENGINEERING
* ISTANBUL UNIVERSITY, FACULTY OF ENGINNEERING
*
*   GLCD CODE
*/

#define F_CPU 8000000UL     /* Define CPU clock Frequency 8MHz */
#include <avr/io.h>         /* Include AVR std. library file */
#include <util/delay.h>     /* Include defined delay header file
*/#include <stdio.h>        /* Include standard i/o library file */
#include <stdlib.h>
#include "Image.h"
#include "Font_Header.h"

#define Data_Port     PORTD      /* Define data port for GLCD */
#define Command_Port    PORTC      /* Define command port for GLCD
*/
#define Data_Port_Dir   DDRD     /* Define data port for GLCD */
#define Command_Port_Dir  DDRC     /* Define command port for GLCD
*/
#define RS          PC0        /* Define GLCD control pins */
#define RW          PC1
#define EN          PC2
#define CS1           PC3
#define CS2           PC4
```

```
#define RST           PC5

#define TotalPage     8

void GLCD_Command(char Command)    /* GLCD command function */
{
  Data_Port = Command;           /* Copy command on data pin */
  Command_Port &= ~(1 << RS);/* Make RS LOW to select command
register */
  Command_Port &= ~(1 << RW);      /* Make RW LOW to select write
operation */
  Command_Port |=  (1 << EN);      /* Make HIGH to LOW transition on
Enable pin */
  _delay_us(5);
  Command_Port &= ~(1 << EN);
  _delay_us(5);
}

void GLCD_Data(char Data)        /* GLCD data function */
{
  Data_Port = Data;            /* Copy data on data pin */
  Command_Port |=  (1 << RS);      /* Make RS HIGH to select data
register */
  Command_Port &= ~(1 << RW);      /* Make RW LOW to select write
operation */
  Command_Port |=  (1 << EN);      /* Make HIGH to LOW transition on
Enable pin */
  _delay_us(5);
  Command_Port &= ~(1 << EN);
  _delay_us(5);
}
```

```
void GLCD_Init()                /* GLCD initialize function */
{
  Data_Port_Dir = 0xFF;
  Command_Port_Dir = 0xFF;
/* Select both left & right half of display & Keep reset pin high
*/
  Command_Port |= (1 << CS1) | (1 << CS2) | (1 << RST);
  _delay_ms(20);
  GLCD_Command(0x3E);           /* Display OFF */
  GLCD_Command(0x40);           /* Set Y address (column=0) */
  GLCD_Command(0xB8);           /* Set x address (page=0) */
  GLCD_Command(0xC0);           /* Set z address (start line=0) */
  GLCD_Command(0x3F);           /* Display ON */
}

void GLCD_ClearAll()            /* GLCD all display clear function */
{
  int i,j;
  /* Select both left & right half of display */
  Command_Port |= (1 << CS1) | (1 << CS2);
  for(i = 0; i < TotalPage; i++)
  {
    GLCD_Command((0xB8) + i);   /* Increment page each time after
64 column */
    for(j = 0; j < 64; j++)
    {
      GLCD_Data(0);             /* Write zeros to all 64 column */
    }
  }
  GLCD_Command(0x40);           /* Set Y address (column=0) */
  GLCD_Command(0xB8);           /* Set x address (page=0) */
}
```

```
void GLCD_Imig(const char* image)  /* GLCD string write function */
{
  int column,page,page_add=0xB8,k=0;
  float page_inc=0.5;
  char byte;

  Command_Port |= (1 << CS1);/* Select first Left half of display
*/
  Command_Port &= ~(1 << CS2);

  for(page=0;page<16;page++)    /* Print 16 pages i.e. 8 page of
each half of display */
  {
    for(column=0;column<64;column++)
    {
      byte = pgm_read_byte(&image[k+column]);
      GLCD_Data(byte);    /* Print 64 column of each page */
    }
    Command_Port ^= (1 << CS1);     /* If yes then change segment
controller */
    Command_Port ^= (1 << CS2);
    GLCD_Command((page_add+page_inc));/* Increment page address */
    page_inc=page_inc+0.5;
    k=k+64;                       /* Increment pointer */
  }
  GLCD_Command(0x40);             /* Set Y address (column=0) */
  GLCD_Command(0xB8);             /* Set x address (page=0) */
}

void GLCD_String(char page_no, char *str)      /* GLCD string write
function */
```

```
{
  unsigned int i, column;
  unsigned int Page = ((0xB8) + page_no);
  unsigned int Y_address = 0;
  float Page_inc = 0.5;

  Command_Port |= (1 << CS1);            /* Select first Left half
of display */
  Command_Port &= ~(1 << CS2);

  GLCD_Command(Page);
  for(i = 0; str[i] != 0; i++)           /* Print each char in
string till null */
  {
    if (Y_address > (1024-(((page_no)*128)+FontWidth))) /* Check
Whether Total Display get overflowed */
    break;                          /* If yes then break writing */
    if (str[i]!=32)                     /* Check whether character is not
a SPACE */
    {
      for (column=1; column<=FontWidth; column++)
      {
        if ((Y_address+column)==(128*((int)(Page_inc+0.5))))  /* If
yes then check whether it overflow from right side of display */
        {
          if (column == FontWidth)    /* Also check and break if
it overflow after 5th column */
          break;
          GLCD_Command(0x40);       /* If not 5th and get
overflowed then change Y address to START column */
          Y_address = Y_address + column;  /* Increment Y address
count by column no. */
```

```
            Command_Port ^= (1 << CS1);    /* If yes then change
segment controller to display on other half of display */
            Command_Port ^= (1 << CS2);
            GLCD_Command(Page + Page_inc);/* Execute command for
page change */
            Page_inc = Page_inc + 0.5;     /* Increment Page No. by
half */
          }
        }
      }
      if (Y_address>(1024-(((page_no)*128)+FontWidth)))    /* Check
Whether Total Display get overflowed */
      break;                        /* If yes then break writing */
      if((font[((str[i]-32)*FontWidth)+4])==0 || str[i]==32)/* Check
whether character is SPACE or character last column is zero */
      {
        for(column=0; column<FontWidth; column++)
        {
          GLCD_Data(font[str[i]-32][column]);  /* If yes then then
print character */
          if((Y_address+1)%64==0)        /* check whether it gets
overflowed  from either half of side */
          {
            Command_Port ^= (1 << CS1);    /* If yes then change
segment controller to display on other half of display */
            Command_Port ^= (1 << CS2);
            GLCD_Command((Page+Page_inc));/* Execute command for
page change */
            Page_inc = Page_inc + 0.5;     /* Increment Page No. by
half */
          }
```

```
        Y_address++;           /* Increment Y_address count per
column */
      }
    }
    else                       /* If character is not SPACE or
character last column is not zero */
    {
      for(column=0; column<FontWidth; column++)
      {
        GLCD_Data(font[str[i]-32][column]); /* Then continue to
print hat char */
        if((Y_address+1)%64==0)       /* check whether it gets
overflowed  from either half of side */
        {
          Command_Port ^= (1 << CS1);   /* If yes then change
segment controller to display on other half of display */
          Command_Port ^= (1 << CS2);
          GLCD_Command((Page+Page_inc));/* Execute command for
page change */
          Page_inc = Page_inc + 0.5;    /* Increment Page No. by
half */
        }
        Y_address++;           /* Increment Y_address count per
column */
      }
      GLCD_Data(0);                /* Add one column of zero to
print next character next of zero */
      Y_address++;               /* Increment Y_address count for
last added zero */
      if((Y_address)%64 == 0)         /* check whether it gets
overflowed  from either half of side */
      {
```

```
        Command_Port ^= (1 << CS1);     /* If yes then change
segment controller to display on other half of display */
        Command_Port ^= (1 << CS2);
        GLCD_Command((Page+Page_inc));  /* Execute command for
page change */
        Page_inc = Page_inc + 0.5;      /* Increment Page No. by
half */
      }
    }
  }
  GLCD_Command(0x40);               /* Set Y address (column=0) */
}
```

Appendix 2: Sensors code.

```
/*Infaant Incubator Analyzer.c
*
* ENG. AHMED TALAL
* MASTER DEGREE THESIS
* BIOMEDICAL ENGINEERING
* ISTANBUL UNIVERSITY, FACULTY OF ENGINNEERING
*
*      FINAL IMPLEMENTATION CODE
*/

#include <avr/io.h>     /*Includes io.h header file where all the
Input/Output Registers and its Bits are defined for all AVR
microcontrollers*/
#define F_CPU 8000000   /*Defines a macro for the delay.h header
file. F_CPU is the microcontroller frequency value for the delay.h
header file. Default value of F_CPU in delay.h header file is
1000000(1MHz)*/
```

```
#include <util/delay.h>   /*Includes delay.h header file which
defines two functions, _delay_ms (millisecond delay) and _delay_us
(microsecond delay)*/
#include <stdio.h>
#include <stdlib.h>     /*Includes stdlib.h header file which
defines different standard library functions.*/
#include <avr/interrupt.h>

#define degree_sysmbol 0x00, 0x05, 0x03, 0x00, 0x00
#define DHT11_PIN 2

extern const unsigned char Logo[];
extern const unsigned char IIA[];
extern const unsigned char face[];

void adc_init(void);                  /*ADC Function Declarations*/
int read_adc(unsigned char channel);
uint8_t Receive_data();
uint8_t c=0,I_RH,A_RH,I_Temp,A_Temp,CheckSum,celsius,dB;

  char adcResult[4];

int main(void)
{
//adc_init();                         /*ADC initialization*/
  GLCD_Init();                       /*LCD Initialization*/



  GLCD_Imig(Logo);
  _delay_ms(2000);
```

```
GLCD_ClearAll();
GLCD_Imig(IIA);
_delay_ms(3000);

GLCD_ClearAll();
GLCD_String(3," READING PARAMETERS!");
GLCD_String(4," .");
_delay_ms(500);
GLCD_String(4," ..");
_delay_ms(500);
GLCD_String(4," ...");
_delay_ms(500);
GLCD_String(4," ....");
_delay_ms(500);
GLCD_String(4," .....");
_delay_ms(500);
GLCD_String(4," ......");
_delay_ms(500);
GLCD_String(4," .......");
_delay_ms(500);
GLCD_String(4," ........");
_delay_ms(500);
GLCD_String(4," .........");
_delay_ms(500);
GLCD_String(4," ..........");
_delay_ms(500);
GLCD_String(4," ............");
_delay_ms(500);
GLCD_String(4," .............");
_delay_ms(500);
GLCD_String(4," ..............");
_delay_ms(500);
```

```
GLCD_String(4," ...............");
_delay_ms(500);
GLCD_String(4," ................");
_delay_ms(500);
GLCD_String(4," .................");
_delay_ms(500);
GLCD_String(4," ..................");
_delay_ms(500);
GLCD_String(4," ...................");
_delay_ms(500);
GLCD_String(4," ....................");
_delay_ms(500);

GLCD_ClearAll();

GLCD_String(3," READING PARAMETERS!");
GLCD_String(4," .");
_delay_ms(500);
GLCD_String(4," ..");
_delay_ms(500);
GLCD_String(4," ...");
_delay_ms(500);
GLCD_String(4," ....");
_delay_ms(500);
GLCD_String(4," .....");
_delay_ms(500);
GLCD_String(4," ......");
_delay_ms(500);
GLCD_String(4," .......");
_delay_ms(500);
GLCD_String(4," ........");
_delay_ms(500);
```

```
GLCD_String(4," .........");
_delay_ms(500);
GLCD_String(4," ..........");
_delay_ms(500);
GLCD_String(4," ...........");
_delay_ms(500);
GLCD_String(4," ............");
_delay_ms(500);
GLCD_String(4," .............");
_delay_ms(500);
GLCD_String(4," ..............");
_delay_ms(500);
GLCD_String(4," ...............");
_delay_ms(500);
GLCD_String(4," ................");
_delay_ms(500);
GLCD_String(4," .................");
_delay_ms(500);
GLCD_String(4," ..................");
_delay_ms(500);
GLCD_String(4," ...................");
_delay_ms(500);

GLCD_ClearAll();
GLCD_Imig(face);

GLCD_String(0,"  AHMEDTALAL  Ist.Uni");
GLCD_String(2,"          Temp1 ");
GLCD_String(3,"          Temp2 ");
GLCD_String(4,"          R.H ");
GLCD_String(5,"          S.Noise");
_delay_ms(500);
```

```
  DDRA = 0x00;                          /* Make ADC port as input */
  ADCSRA = 0x87;                       /* Enable ADC, with freq/128  */
  ADMUX = 0x40;                        /* Vref: Avcc, ADC channel: 0 */
  ADCSRA |= 1<<ADPS2;
  //ADMUX |= 1<<ADLAR;
  ADMUX |= 1<<REFS0;
  ADCSRA |= 1<<ADIE;
  ADCSRA |= 1<<ADEN;
  sei();
  ADCSRA |= 1<<ADSC;

/*Start of infinite loop*/
while(1)
{

}
return 0;
}
/*End of Program*/

void Request()                  /* Microcontroller send start
pulse/request */
{
  DDRA |= (1<<DHT11_PIN);
  PORTA &= ~(1<<DHT11_PIN);      /* set to low pin */
  _delay_ms(20);                /* wait for 20ms */
  PORTA |= (1<<DHT11_PIN);       /* set to high pin */
}

void Response()                 /* receive response from DHT11 */
{
  DDRA &= ~(1<<DHT11_PIN);
```

```
  while(PINA & (1<<DHT11_PIN));
  while((PINA & (1<<DHT11_PIN))==0);
  while(PINA & (1<<DHT11_PIN));
}

uint8_t Receive_data()      /* receive data */
{
  for (int q=0; q<8; q++)
  {
    while((PINA & (1<<DHT11_PIN)) == 0);  /* check received bit 0
or 1 */
    _delay_us(30);
    if(PINA & (1<<DHT11_PIN))/* if high pulse is greater than 30ms
*/
    c = (c<<1)|(0x01);  /* then its logic HIGH */
    else     /* otherwise its logic LOW */
    c = (c<<1);
    while(PINA & (1<<DHT11_PIN));
  }
  return c;
}

int read_adc(unsigned char channel)
{
  ADMUX = 0x40 | (channel & 0x07);       /* set input channel to
read */
  ADCSRA |= (1<<ADSC);                /* Start ADC conversion */
  while (!(ADCSRA & (1<<ADIF)));         /* Wait until end of
conversion by polling ADC interrupt flag */
  ADCSRA |= (1<<ADIF);                /* Clear interrupt flag */
  _delay_ms(1);                     /* Wait a little bit */
  return ADCW;                     /* Return ADC word */
```

```
}
ISR(ADC_vect)
{
  uint8_t theLOW = ADCL;
  uint16_t tenBitValue = ADCH << 8 | theLOW;

  char data[5];
  char Temperature[10];
  float celsius;

  Request();   /* send start pulse */
  Response();    /* receive response */
  I_RH=Receive_data(); /* store first eight bit in I_RH */
  A_RH=Receive_data(); /* store next eight bit in D_RH */
  I_Temp=Receive_data(); /* store next eight bit in I_Temp */
  A_Temp=Receive_data(); /* store next eight bit in D_Temp */
  CheckSum=Receive_data(); /* store next eight bit in CheckSum */
  celsius = (read_adc(1)*4.88);
  celsius = (celsius/10.00);
  sprintf(Temperature,"%d%c`C  ", (int)celsius,0);               /*
convert integer value to ASCII string */
  GLCD_String(2,"    `C");
  GLCD_Command(0x40);
  GLCD_String(2,Temperature);                /* send string data
for printing */
  memset(Temperature,0,10);
  _delay_ms(1000);

  itoa(I_RH,data,10);
  GLCD_Command(0x40);
  GLCD_String(4,data);
  GLCD_Command(0x4C);
```

```
    GLCD_String(4,".");

    itoa(A_RH,data,10);
    GLCD_Command(0x51);
    GLCD_String(4,data);
    GLCD_Command(0x5A);
    GLCD_String(4,"%");

    itoa(I_Temp,data,10);
    GLCD_Command(0x40);
    GLCD_String(3,data);

    GLCD_String(3,data);
    GLCD_Command(0x4F);
    GLCD_String(3," `C");

    GLCD_String(5,"    dB");
    itoa(tenBitValue/2.744681,adcResult, 10);
    GLCD_Command(0x40);
    GLCD_String(5,adcResult);
    ADCSRA |= 1<<ADSC;
}
```

CURRICULUM VITAE

Personal Information	
Name Surname	Ahmed Talal SALLAM
Place of Birth	Sana’a, Yemen
Date of Birth	14.02 1991
Nationality	❒ T.C. ❒ Other: Yemeni
Phone Number	+90 541 493 41 84
Email	ahmedtalal_22@yahoo.com ahmedtalal1991@gmail.com
Web Page	

Educational Information	
B. Sc.	
University	University of Science and Technology
Faculty	Faculty of Engineeering
Department	Biomedical Engineering
Graduation Year	01.01.2013

M. Sc.	
University	Istanbul University
Institute	Institute of Graduate Studies in Science and Engineering
Department	Biomedical Engineering
Programme	Biomedical Engineering