

CHAPTER 1
INTRODUCTION TO SCIENTIFIC CALCULATOR FOR
BLIND

INTRODUCTION:

1.1 Introduction

Talking calculator is a device that can built-in speech synthesizer that reads a loud each number, symbol, or operation key a user presses. It also vocalizes the answer to the problem. This talking calculator is very easy to use. If the users press the keypad or button, it can make sound and display what the user press. It can make disabled feel interesting and happy to learn mathematics.

The talking calculator can help disabled to solve a simple mathematic question. Nowadays, the device or equipment for disabled to learn mathematics is limited and hard to find the suitable device for disabled. The existing talking calculator for disabled in market is very expensive. The disabled need more money to buy the suitable equipment just for to learn mathematics. In their daily use, the disabled need money for treatment their disease.

With the increasing number of disabled people, the Indian government is not keeping numb to provide the relevant and education quality for disabled. It is the government's policy that education for persons with disabilities should form an integral part of national educational planning, curriculum development and school organization. Thus, new design of talking calculator with a low cost can lower the burden for disabled. It can help and make disabled interested in learning. The disabled can follow the curriculum development and not left behind the technology advances.

1.2 Problem Statement

Currently, the existing talking calculator for disabled is very expensive and they cannot focus to buy the talking calculator. Besides that, the talking calculator in market is not a good design. That mean the talking calculator is not portable. Some

disabled have a difficulty to use the talking calculator. Thus, the lower cost of talking calculator and a simple design will help the disabled in learning the mathematic and uses in daily live.

1.3 Objectives:

There are four objectives in this project. The main objective is to develop a talking calculator for disabled in a lower cost. It was including the cost for the software and hardware material to design the talking calculator. The second objective is to design a simple and user friendly device. The next objective is to facilitate blind people especially children to solve the mathematical equation. Finally is to evaluate performance one device that can perform a good efficiency for calculate mathematical equation. The aims for design this talking calculator is low cost material and simple design.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The talking calculator is developing for blind people for their use to learn and solve mathematics equation. New and improved calculators have been introduced almost daily by various manufactures. These calculators are indicated to working adults and school children. Thus, for the disabled especially blind people cannot take advantages to explore the technology because pocket calculator is visual.

The disabled cannot use the calculator and need a suitable calculator. The motivation for these efforts stems largely from the realization that an effective calculator for the blind would not only provide a quick and reliable means for doing calculation, but also would offer the possibility of increasing the mathematical skills of the blind. Mathematics has traditionally been a difficult subject for blind students, partly because of the complications of displaying mathematics in Braille.

2.2 Basic Mathematical operation

The basic calculator operations at least need four functions. It need add, subtract, multiply, and divide. Basically, the disabled just need talking calculator, they do not deserve the scientific calculator. The disabled feel happy if get the talking calculator because it can be use in daily life and get advantage to learn mathematics.

Nearly half of the survey respondents indicated that they would use a calculator primarily for personal use, and only 5 percent needed full scientific capability [1]. Having a calculator is fine and dandy, but how will know what operations to use calculator. A good analogy is why should learn directions if you have a GPS. The GPS will get you around if you know the exact address, but if you don't then it's useless. A calculator only works if you know which buttons to press. Studying math is not just about learning how the functions work, but also learning how to think about a problem that you don't understand and come up with a way to solve it. The calculator can only do your work if you tell it what to do. So, the disabled need a simple directions or instructions to use talking calculator.

Also, the talking calculator must be portable for disabled use it and battery operation consumption must be well for daily use. This is because disabled need a portable unit as close as possible to pocket size. So, it will be easy to carry the calculator. The utility of a

calculator would also be increased considerably if those disabled who cannot read ordinary calculators would be able to use it.

2.3 Braille numbers- International Standard

These familiarize with Braille its the first thing need to do because this talking calculator will have special button will make the blind people more easy to use. The push button will have Braille numbers. The Braille cell is the basic unit of the Braille system. A Braille cell is a group of six dots.

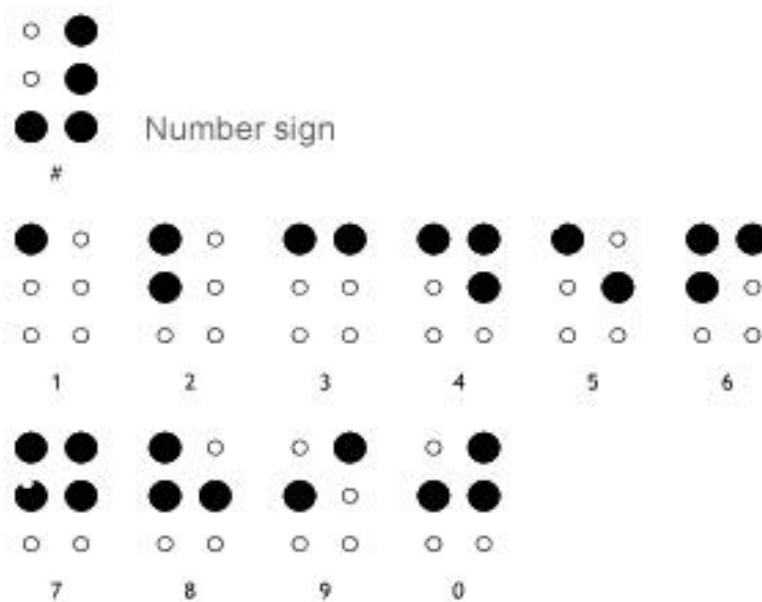


Figure 2.1: Number Braille

The basic grid of a Braille alphabet character consists of six dots, positioned like the figure six on a die, in two parallel vertical lines of three dots each. From the six dots that make up the basic grid, 64 different signs can be created. Reading direction of Braille is the same as for regular type and the rules for hyphenation that apply for regular fonts also apply in Braille. The guidelines on Braille requirements for pharmaceutical label and packaging recommend that an un-contracted Braille alphabet system, conforming to the Marburg Medium format, should be used. Braille character sets consist of letters, numbers, punctuation, symbols and special characters. Some parts of character sets are common between countries whereas other parts differ.

The technologies nowadays move like speed of light. Now, have are software that can be use for a Braille number. Math2braille is the opening access to mathematics for a blind people. It is clear that Braille has many problems when representing the complex information associated with mathematics. The reliance on linear representation removes much of the structure which aids the sighted user to quickly navigate an equation the concept and process in converting MathML to Braille take a similar route as the conversion of music to accessible standards. The work is based on recent experience of multimedia modeling techniques and the development of software for producing Braille Music.

2.4 Audio output :

The sound system is major input for disabled to use the calculator. They cannot see the display or visual, with sound system they will make less error for use the talking calculator. Thus, to make that special to other disabled, choosing the right output for calculator can make their easy to use the calculator. The concept of sound system in calculator is low user frustration for disabled. This subjective parameter relates to the others, and means basically that the display is easy to use. The operator should be able to concentrate on the mathematical problem at hand, not on just reading the display.

The packaging problem involved innumerable details, but was centrally concerned with satisfying the following requirements:

1. Small and lightweight enough to be hand-held
2. Keyboard designed expressly for the blind.

The Audio-Keyboard Display was eliminated early in the evaluation because we determined that it required too much effort on the part of the user. From seven evaluators who say the remaining three displays, six preferred speech output, and speech proved to be the fastest and most accurate.

2.5 Hardware Study

The major hardware component need to control the program of this project is Arduino Due. Arduino Due is microcontroller. All the coding of C language will be compiling into Arduino. The push button was the input of this project. Thus, for the complete this project the mp3 player shield was integrate with Arduino as to store the memory of sound for each push button. This is because the Arduino cannot make sound. Hence, that why this project need mp3 player shield is needed for this project to produce the sound.

For the display Lcd modules was used to display what user press. That was confirming that the user press push button. So, the output for this project is the display and the sound for what the user press.



Figure 2.2: Arduino Due

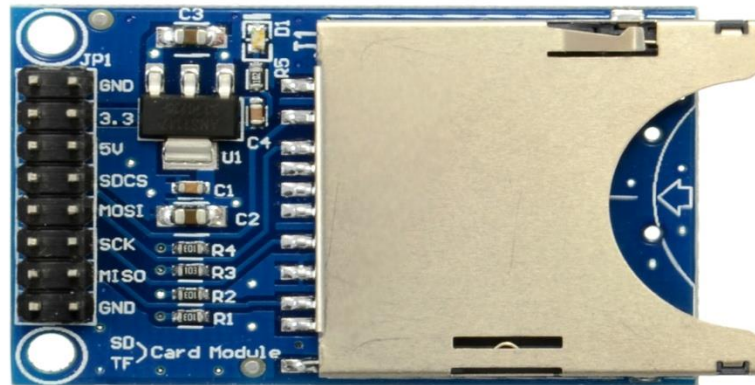


Figure 2.3: SD card module

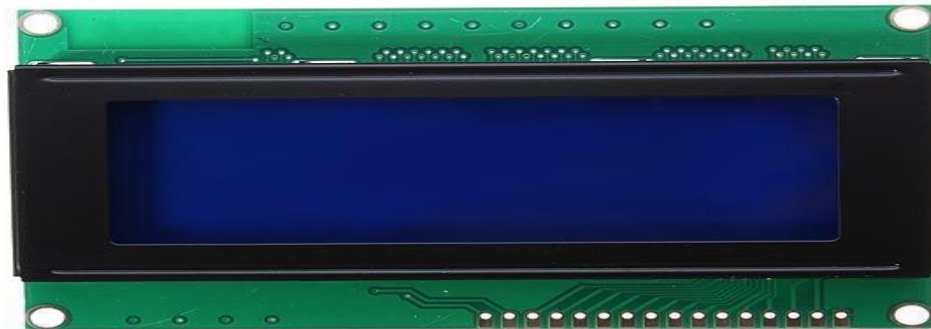


Figure 2.4: LCD Display

CHAPTER 3

METHODOLOGY

METHODOLOGY:

4.1 Introduction

This chapter will discuss the methodology that was used in this project. This project involves programming software and hardware development.

4.2 Software Design

The function of software design is to build a code use C language to operate same as usual calculator. After research on several journals, a system is developed that user just enter the 1st number, select operation, and enter the 2nd number. So, the user can get the answer for the mathematical equation. It is easier for the people do not have visual impairments get the answer. Before start write the code, the flow chart is needed to the program code run properly and follow the sequences.

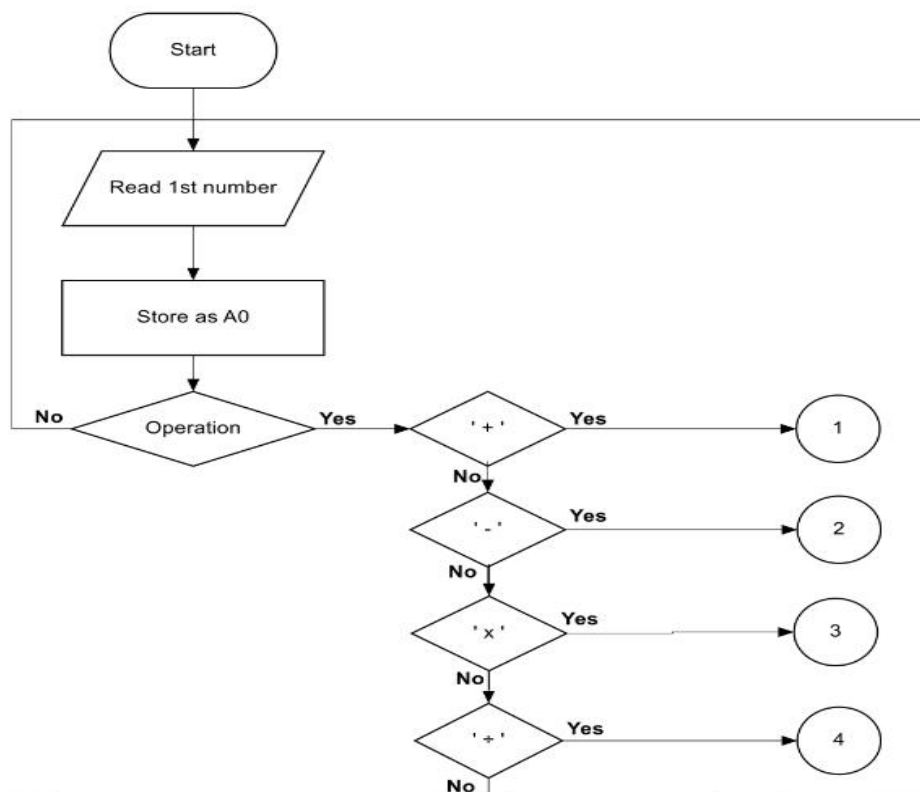


Figure 3.1: The main flowchart of the talking calculator

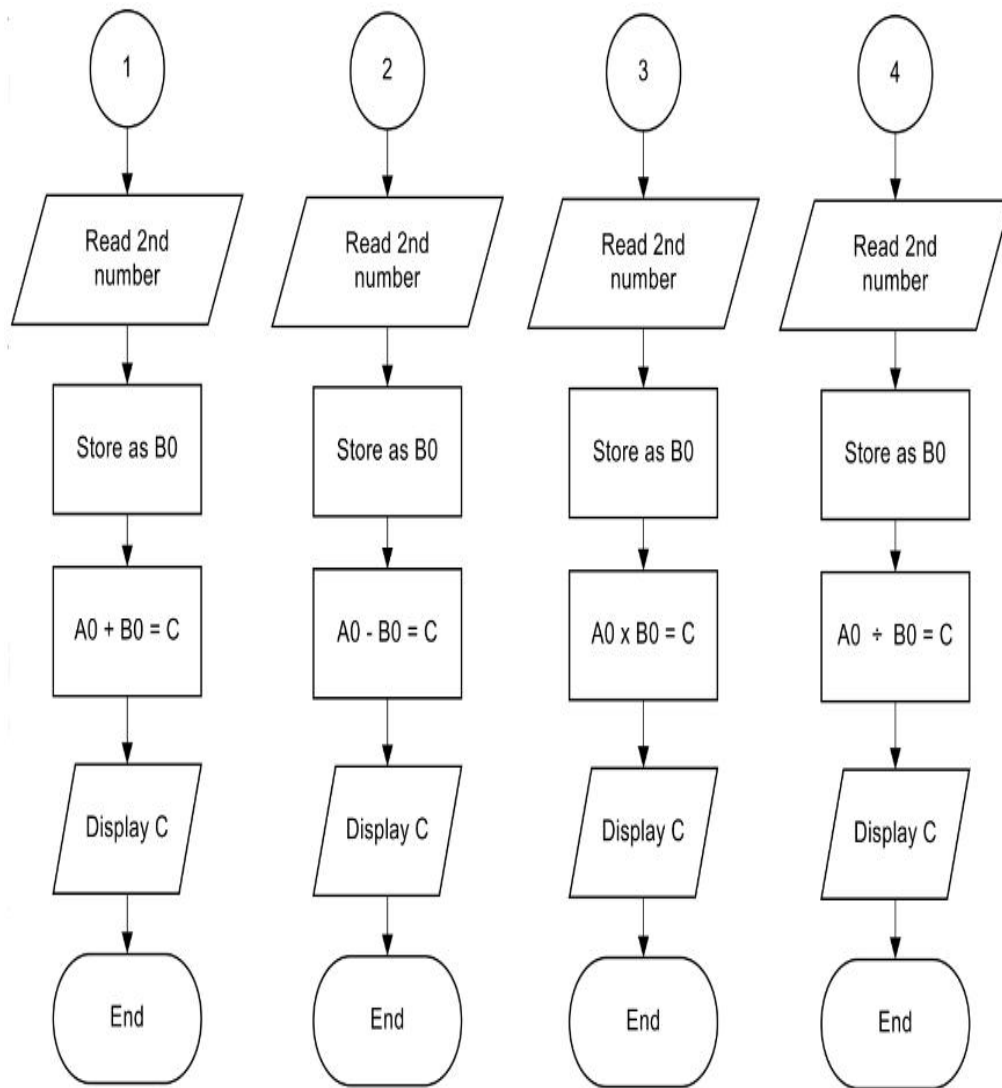


Figure 3.2: The flowchart for each operation

Figure 3.1 shows flowchart of main part of this project. Referring to the figure3.1, the user cannot proceed used the calculator if they do not enter the operation. It will loop to first step that user needed to insert the first number. So, the user do not worry if enter the wrong number. Just do not enter the operation. The system will loop and ask back user to enter the first number. So, it can make easier for blind people to use it. For this system, after user chooses the operation it will go to function that user chose.

Therefore, Figure3.2 shows the extended flowchart from main chart. It also shows the operation and the sequences of this talking calculator. Basically, to build a coding for this system needs a main part to loop the system. So the system never ends. It can make user use this talking calculator until it turns off by using a switch button. For the looping process, basically it just the same steps from the main chart but the different in operation. After operation was chosen, user needs to enter the second number and the system will calculate the answer automatically.

3.2.1 Arduino IDE

A minimal Arduino C/C++ sketch, as seen by the Arduino IDE programmer, consist of only two functions:

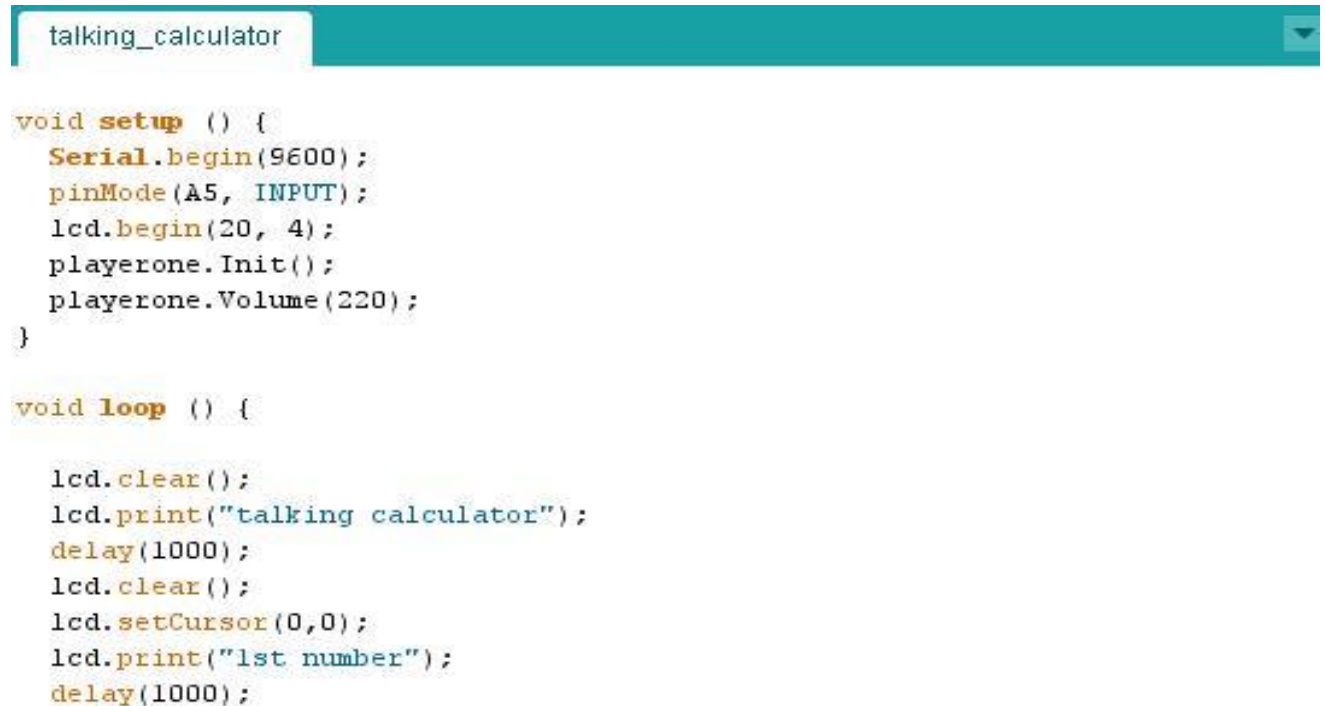
- *setup*: This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- *loop*: After *setup* has been called, function *loop* is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

Most Arduino boards contain a [light-emitting diode](#) (LED) and a load resistor connected between pin 13 and ground, which is a convenient feature for many tests and program functions. A typical program for a beginning Arduino programmer blinks an LED repeatedly.

A screenshot of the Arduino IDE interface. The top toolbar shows icons for checking, running, uploading, and downloading. Below the toolbar, the sketch name 'sketch_jun07b' is displayed. The main code area contains the following C++ code:

```
void setup() {  
  // put your setup code here, to run once:  
  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
  
}
```

Figure 3.3: Basic Arduino software



```
void setup () {  
  Serial.begin(9600);  
  pinMode(A5, INPUT);  
  lcd.begin(20, 4);  
  playerone.Init();  
  playerone.Volume(220);  
}  
  
void loop () {  
  lcd.clear();  
  lcd.print("talking calculator");  
  delay(1000);  
  lcd.clear();  
  lcd.setCursor(0,0);  
  lcd.print("1st number");  
  delay(1000);  
}
```

Figure 3.4: The main coding

Figure 3.4 show the main coding after follow the flowchart. It involved all functions to make a talking calculator. It can show, make a sound, and calculate simple mathematical question. After write the code based on C language, it need to compile using arduino software. It show if the coding have an error or not. For early in compile this coding, it show have many errors but all error can solve properly. Figure 3.5 show the coding is done in compile process and ready to upload at microcontroller, arduino due.

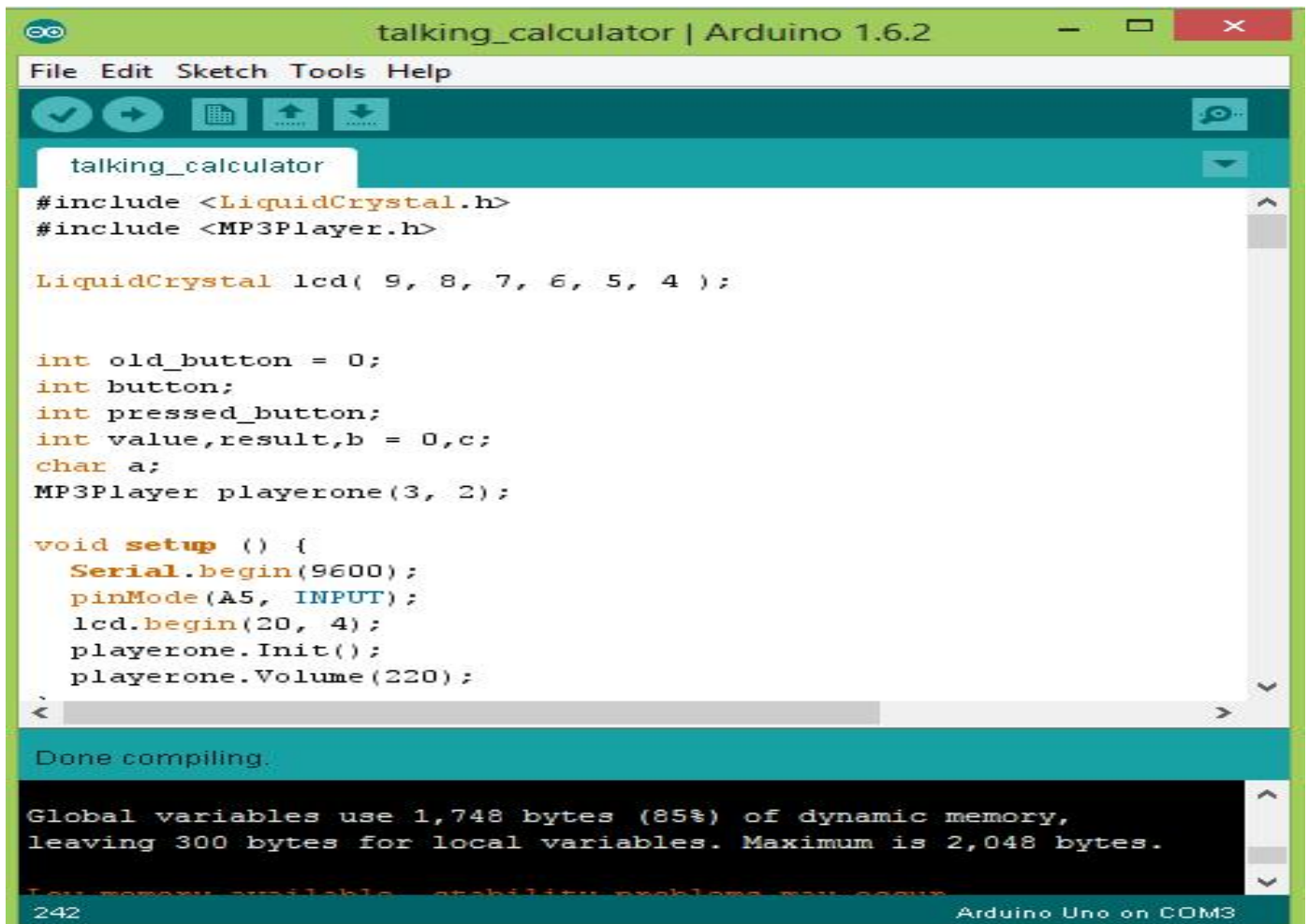


Figure 3.5: The code is done compiling

4.3 Hardware Design

The step after done in software is to list the part by part to complete the hardware design. This part is really important for this project. The result of this project is depending on hardware design. The faster complete in hardware design, it easier to get the result for effectiveness this talking calculator. The design of this talking calculator must be easy to use and portable for people do not have visual

impairment. A several part is needed to complete for hardware design. It involved push button, lcd display, mp3 shield configuration and box of talking calculator.

3.3.1 5x5 Keypad



Figure 3.6: 5x5 Keypad

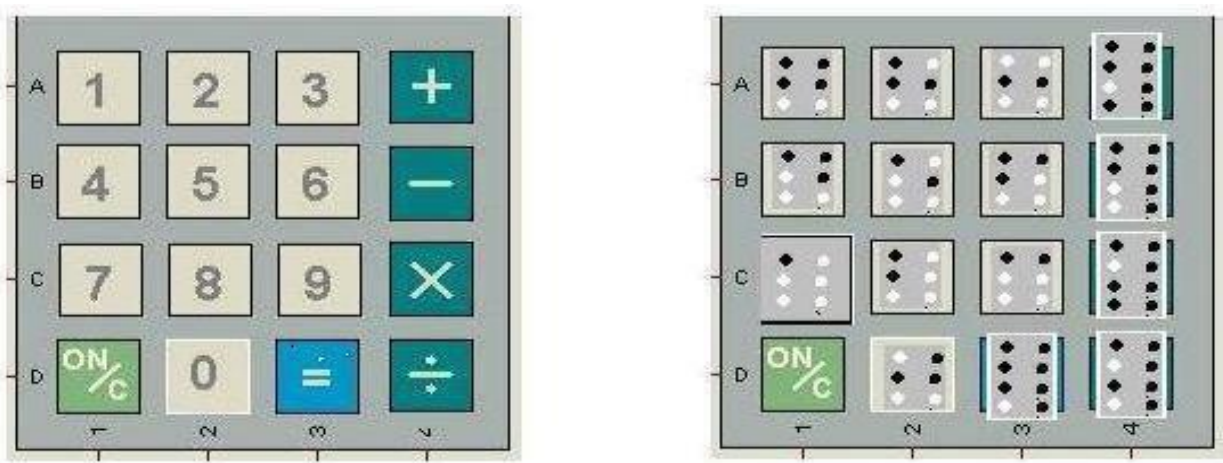


Figure no. 3.7: Braille embossed keypad


```
final_test | Arduino 1.8.2
File Edit Sketch Tools Help

final_test

//int arcsin=28;
//int arccos=30;
//int arctan=32;
//int root=36;
int resetpin=38;

int i=0;

char keys[ROWS][COLS] = {
  {'s','o','t','x','y'},
  {'1','2','3','+','z'},
  {'4','5','6','-','e'},
  {'7','8','9','*','1'},
  {'C','0','=','/','r'}
};

byte rowPins[ROWS] = {8,9,10,11,12}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {3,4,5,6,7}; //connect to the column pinouts of the keypad
void setup() {
  // put your setup code here, to run once:
  Keypad customKeypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup()
{
  digitalWrite(resetpin, HIGH);
  //pinMode(sine, INPUT);
  //pinMode(cosine, INPUT);
  //pinMode(tangent, INPUT);
  //pinMode(logue, INPUT);
  //pinMode(sine, INPUT);
```

Figure 3.8: Code for Keypad

3.3.2 LCD

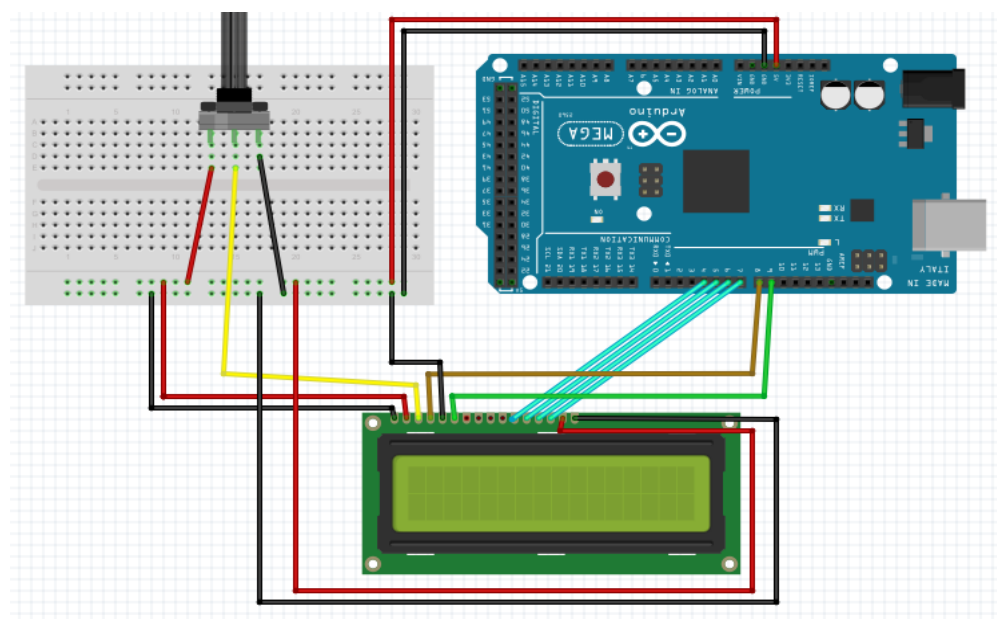


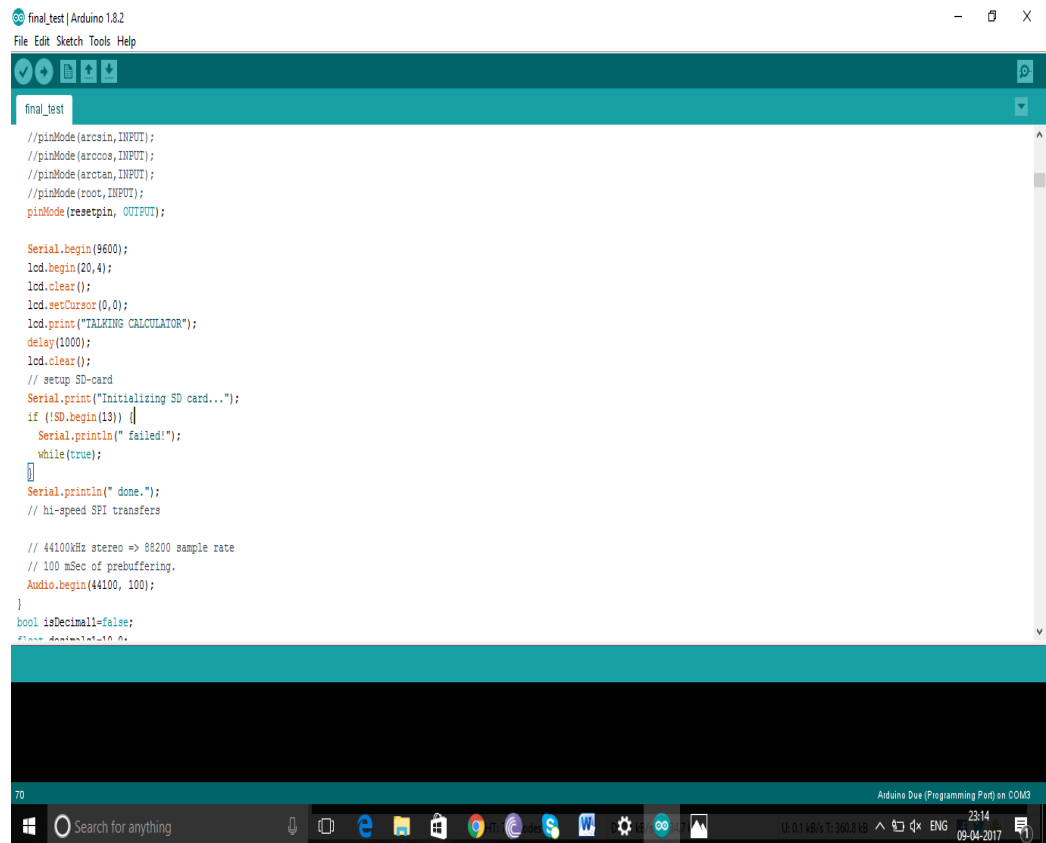
Figure 3.9: Basic interfacing of LCD

```
void loop () {  
  
  lcd.clear();  
  lcd.print("talking calculator");  
  delay(1000);  
  lcd.clear();  
  lcd.setCursor(0,0);  
  lcd.print("1st number");  
  delay(1000);  
}
```



Figure 3.10: Code for LCD

3.3.3 SD Card Module



The screenshot shows the Arduino IDE interface with a sketch named 'final_test'. The code is as follows:

```
//pinMode(arcsin, INPUT);
//pinMode(arccos, INPUT);
//pinMode(arctan, INPUT);
//pinMode(root, INPUT);
pinMode(resetpin, OUTPUT);

Serial.begin(9600);
lcd.begin(20, 4);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("TALKING CALCULATOR");
delay(1000);
lcd.clear();
// setup SD-card
Serial.print("Initializing SD card...");
if (!SD.begin(13)) {
  Serial.println(" failed!");
  while(true);
}
Serial.println(" done.");
// hi-speed SPI transfers

// 44100kHz stereo => 88200 sample rate
// 100 mSec of prebuffering.
Audio.begin(44100, 100);
}
bool isDecimal=false;
float decimal=10.0;
```

The IDE status bar at the bottom indicates 'Arduino Due (Programming Port) on COM3' and shows system information: 'U: 0.1 KB; T: 350.8 KB', '23:14', and '09-04-2017'.

Figure 3.11: Code for interfacing SD card

3.3.4 Arduino DUE

AVR Arduino microcontroller

Microcontroller	AT91SAM3X8E
Operating Voltage	3.3V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-16V
Digital I/O Pins	54 (of which 12 provide PWM output)
Analog Input Pins	12
Analog Output Pins	2 (DAC)
Total DC Output Current on all I/O lines	130 mA
DC Current for 3.3V Pin	800 mA
DC Current for 5V Pin	800 mA
Flash Memory	512 KB all available for the user applications
SRAM	96 KB (two banks: 64KB and 32KB)
Clock Speed	84 MHz
Length	101.52 mm
Width	53.3 mm
Weight	36 g

Figure 3.12: Technical specifications

3.3.4.1 Benefits of ARM-CORTEX

A 32-bit core, that allows operations on 4 bytes wide data within a single CPU clock.
(for more information go to int type page).

CPU Clock at 84Mhz.

96 KBytes of SRAM.

512 KBytes of Flash memory for code.

A DMA controller, that can relieve the CPU from doing memory intensive tasks.

3.3.4.2 Power Supply

The Arduino Due can be powered via the USB connector or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart)

or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

Vin. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or if supplying voltage via the power jack, access it through this pin.

5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 800 mA. This regulator also provides the power supply to the SAM3X microcontroller.

GND. Ground pins.

IOREF. This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V.

3.3.4.3 Programming

The Due can be programmed with the Arduino Arduino Software (IDE). For details, see the reference and tutorials.

Uploading sketches to the SAM3X is different than the AVR microcontrollers found in other Arduino boards because the flash memory needs to be erased before being re-programmed. Upload to the chip is managed by ROM on the SAM3X, which is

run only when the chip's flash memory is empty.

Either of the USB ports can be used for programming the board, though it is recommended to use the Programming port due to the way the erasing of the chip is handled :

Programming port: To use this port, select "Arduino Due (Programming Port)" as your board in the Arduino IDE. Connect the Due's programming port (the one closest to the DC power jack) to your computer. The programming port uses the 16U2 as a USB-to-serial chip connected to the first UART of the SAM3X (RX0 and TX0). The 16U2 has two pins connected to the Reset and Erase pins of the SAM3X. Opening and closing the Programming port connected at 1200bps triggers a "hard erase" procedure of the SAM3X chip, activating the Erase and Reset pins on the SAM3X before communicating with the UART. This is the recommended port for programming the Due. It is more reliable than the "soft erase" that occurs on the Native port, and it should work even if the main MCU has crashed.

Native port: To use this port, select "Arduino Due (Native USB Port)" as your board in the Arduino IDE. The Native USB port is connected directly to the SAM3X. Connect the Due's Native USB port (the one closest to the reset button) to your computer. Opening and closing the Native port at 1200bps triggers a 'soft erase' procedure: the flash memory is erased and the board is restarted with the bootloader. If the MCU crashed for some reason it is likely that the soft erase procedure won't work as this procedure happens entirely in software on the SAM3X. Opening and closing the native port at a different baudrate will not reset the SAM3X.

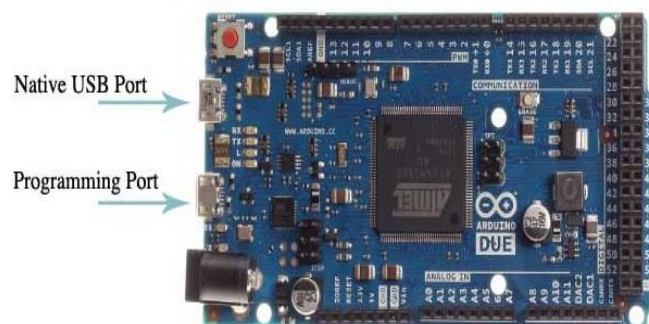


Figure 3.13: Programming Port of Arduino DUE

CHAPTER 4

DESIGN ASPECTS

DESIGN ASPECTS:

5.1 Block Diagram

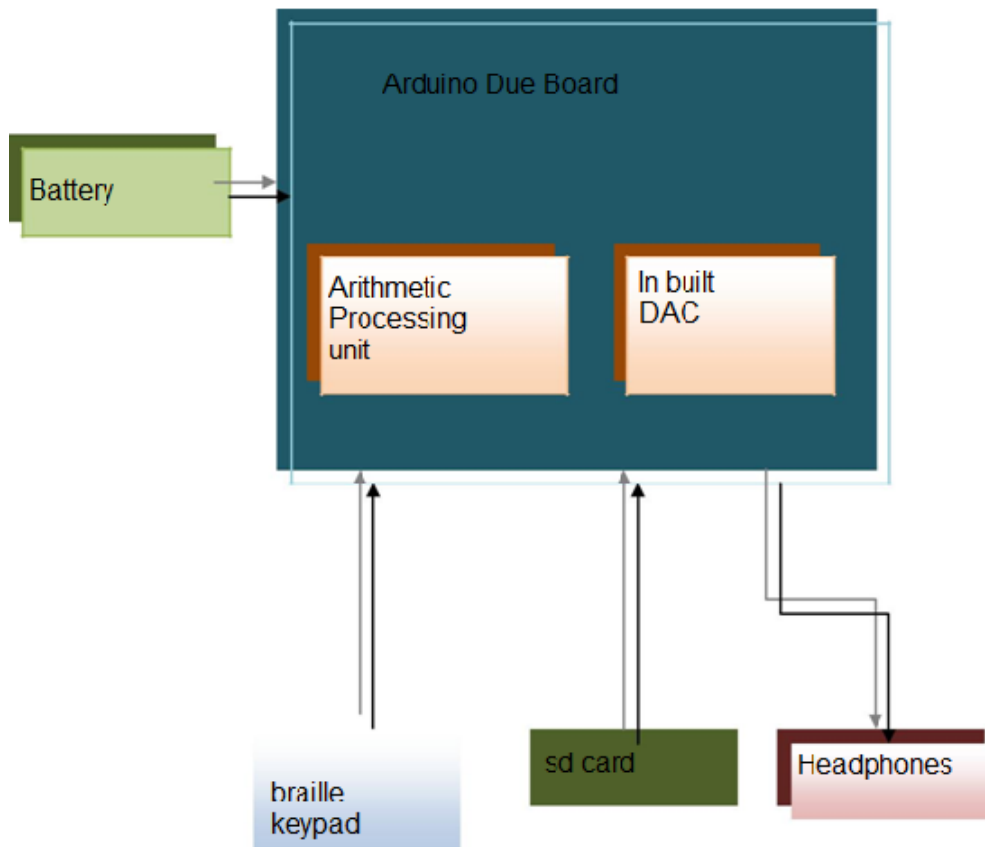


Figure 4.1: Block Diagram

This block diagram shows that microcontroller used is arduino due. The main purpose of using Arduino Due is that, it has inbuilt DAC which is used for conversion of digital audio signal into analog audio signals.

A 9 volt battery source is used as a power supply to the Arduino Due.

Input to microcontroller in the form of basic calculator keypad which is converted into Braille keypad . The SD card module contains .wav file corresponding to each key (numbers and operators) on the keypad along with some extra values like ‘point’ (to denote correct representation following a division operation). Once the equal to button is pressed, Arduino breaks down the result into single digits and outputs the

corresponding audio file for each number of the result, starting from the left-most digit.

The output of the calculator is given out through audio. The model has provision for both headphone and speaker. The headphone is for personal use, and the speaker is used for demonstrative purpose after passing through a low voltage audio amplifier.

5.2 Schematics

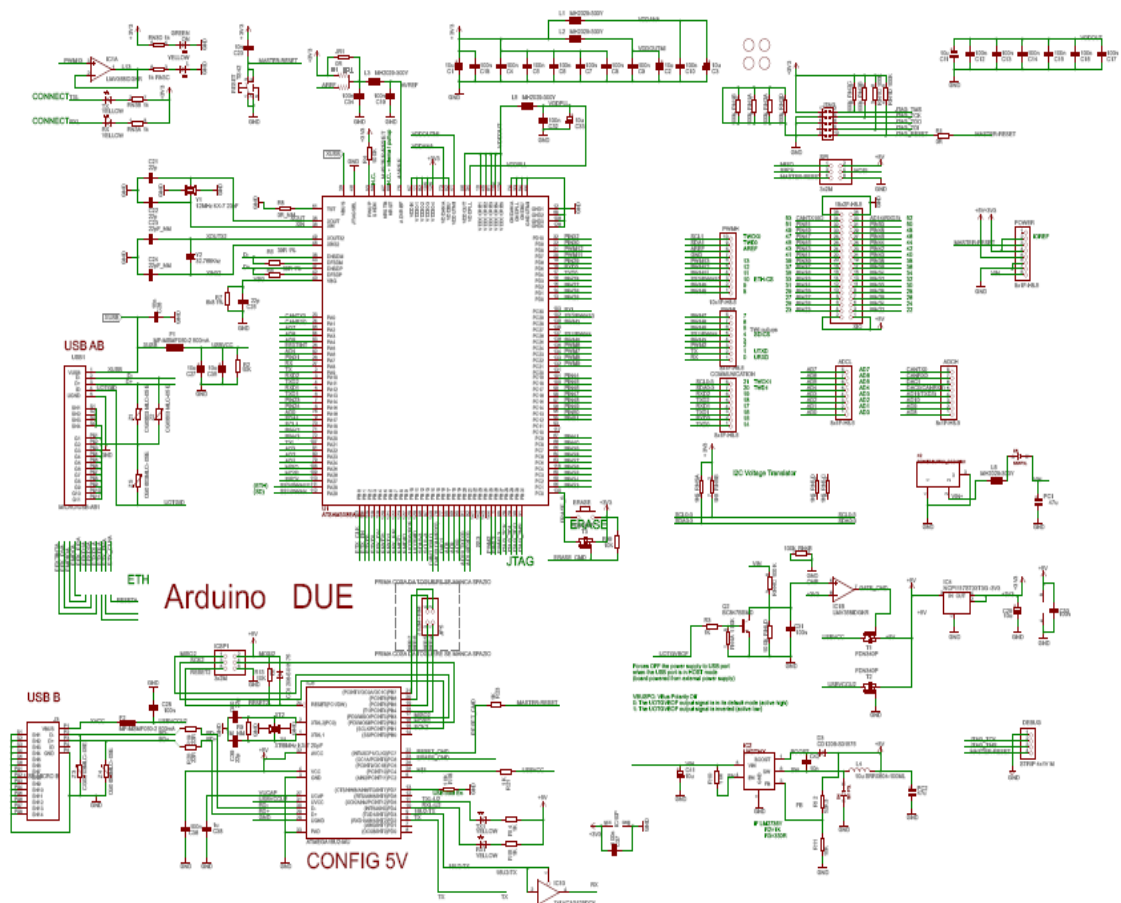


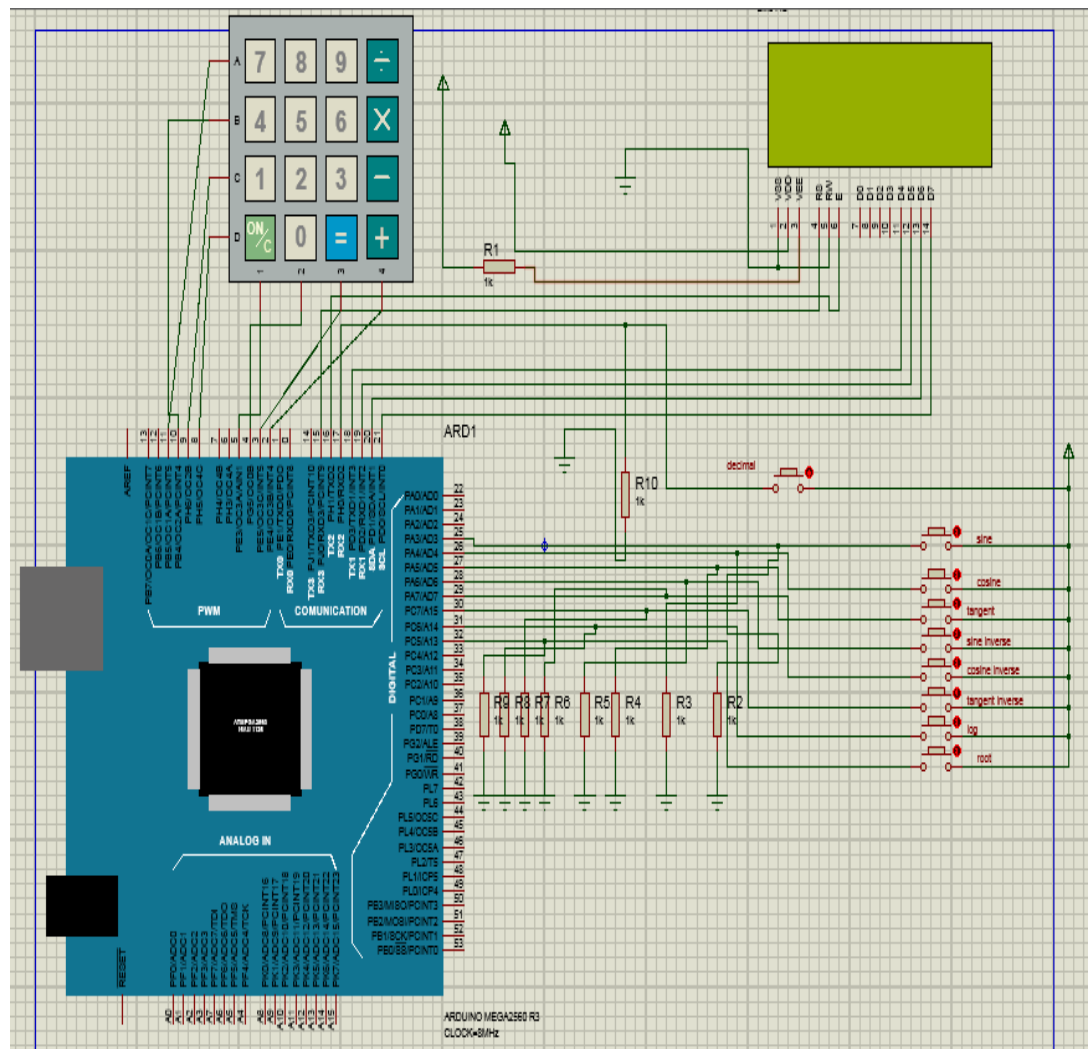
Figure 4.2: Arduino DUE schematics

The Arduino Due is a microcontroller board based on the [Atmel SAM3X8E ARM Cortex-M3 CPU](#). It is the first Arduino board based on a 32-bit ARM core microcontroller. It has 54 digital input/output pins (of which 12 can be used as

PWM outputs), 12 analog inputs, 4 UARTs (hardware serial ports), a 84 MHz clock, an USB OTG capable connection, 2 DAC (digital to analog), 2 TWI, a power jack, an SPI header, a JTAG header, a reset button and an erase button.

The board contains everything needed to support the microcontroller; simply connect it to a computer with a micro-USB cable or power it with a AC-to-DC adapter or battery to get started. The Due is compatible with all Arduino shields that work at 3.3V and are compliant with the 1.0 Arduino pinout.

5.3 Circuit Diagram



When input is given through the braille keypad, the values get stored in the flash memory of processor and then the processor performs various arithmetic operations based on the key pressed such as addition, subtraction, multiplication, division, etc.

The digits of the result are separated, and based on each digit respective .wav files are played and audio output is generated.

DAC converts digital signals into analog voice signals which are audible through headphones. The audio files are stored in the sd card.. Everytime the audio files need to be played, SD card is accessed.

CHAPTER 5
IMPORTANT TERMINOLOGIES

IMPORTANT TERMINOLOGIES:

5.1 Arduino

Arduino is an open source, computer hardware and software company, project, and user community that designs and manufactures microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as do-it yourself kits.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.

The Arduino project started in 2003 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats, and motion detectors.

The name Arduino comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after Arduin of Ivrea, who was the margrave of the March of Ivrea and King of Italy from 1002 to 1014.

5.2 DAC(Digital to Analog Converter)

In electronics, a digital-to-analog converter (DAC, D/A, D–A, D2A, or D-to-A) is a device that converts a digital signal into an analog signal. An analog-to-digital converter (ADC) performs the reverse function.

There are several DAC architectures; the suitability of a DAC for a particular application is determined by three main parameters: resolution, maximum sampling frequency and accuracy. Due to the complexity and the need for precisely matched components, all but the most specialized DACs are implemented as integrated circuits (ICs). Digital-to-analog conversion can degrade a signal, so a DAC should be specified that has insignificant errors in terms of the application.

DACs are commonly used in music players to convert digital data streams into analog audio signals. They are also used in televisions and mobile phones to convert digital video data into analog video signals which connect to the screen drivers to display monochrome or color images. These two applications use DACs at opposite ends of the speed/resolution trade-off. The audio DAC is a low speed high resolution type while the video DAC is a high speed low to medium resolution type. Discrete DACs would typically be extremely high speed low resolution power hungry types, as used in military radar systems. Very high speed test equipment, especially sampling oscilloscopes, may also use discrete DACs.

5.3 SPI Protocol

This library allows you to communicate with SPI devices, with the Arduino as the master device. A Brief Introduction to the Serial Peripheral Interface (SPI)

Serial Peripheral Interface (SPI) is a synchronous serial data protocol used by microcontrollers for communicating with one or more peripheral devices quickly over short distances. It can also be used for communication between two microcontrollers.

With an SPI connection there is always one master device (usually a microcontroller) which controls the peripheral devices. Typically there are three lines common to all the devices:

MISO (Master In Slave Out) - The Slave line for sending data to the master,

MOSI (Master Out Slave In) - The Master line for sending data to the peripherals,

SCK (Serial Clock) - The clock pulses which synchronize data transmission generated by the master and one line specific for every device:

SS (Slave Select) - the pin on each device that the master can use to enable and disable specific devices.

When a device's Slave Select pin is low, it communicates with the master. When it's high, it ignores the master. This allows you to have multiple SPI devices sharing the same MISO, MOSI, and CLK lines.

To write code for a new SPI device you need to note a few things:

The SPI standard is loose and each device implements it a little differently.

Generally speaking, there are four modes of transmission. These modes control whether data is shifted in and out on the rising or falling edge of the data clock signal (called the clock phase), and whether the clock is idle when high or low (called the clock polarity). The four modes combine polarity and phase according to this table:

Mode	Clock Polarity (CPOL)	Clock Phase (CPHA)	Output Edge	Data Capture
SPI_MODE0	0	0	Falling	Rising
SPI_MODE1	0	1	Rising	Falling
SPI_MODE2	1	0	Rising	Falling
SPI_MODE3	1	1	Falling	Rising

Once you have your SPI parameters, use `SPI.beginTransaction()` to begin using the SPI port. The SPI port will be configured with your all of your settings. The simplest and most efficient way to use `SPISettings` is directly inside `SPI.beginTransaction()`. For example:

```
SPI.beginTransaction(SPISettings(14000000, MSBFIRST, SPI_MODE0));
```

If other libraries use SPI from interrupts, they will be prevented from accessing SPI until you call `SPI.endTransaction()`. The SPI settings are applied at the begin of the transaction and `SPI.endTransaction()` doesn't change SPI settings. Unless you, or some library, calls `beginTransaction` a second time, the setting are maintained. You

should attempt to minimize the time between before you call `SPI.endTransaction()`, for best compatibility if your program is used together with other libraries which use SPI.

With most SPI devices, after `SPI.beginTransaction()`, you will write the slave select pin LOW, call `SPI.transfer()` any number of times to transfer data, then write the SS pin HIGH, and finally call `SPI.endTransaction()`.

Note that MISO, MOSI, and SCK are available in a consistent physical location on the ICSP header; this is useful, for example, in designing a shield that works on every board.

All AVR based boards have an SS pin that is useful when they act as a slave controlled by an external master. Since this library supports only master mode, this pin should be set always as OUTPUT otherwise the SPI interface could be put automatically into slave mode by hardware, rendering the library inoperative.

It is, however, possible to use any pin as the Slave Select (SS) for the devices. For example, the Arduino Ethernet shield uses pin 4 to control the SPI connection to the on-board SD card, and pin 10 to control the connection to the Ethernet controller.

5.4 Micro SD card

microSD is a type of removable flash memory card used for storing information. SD is an abbreviation of Secure Digital, and microSD cards are sometimes referred to as μ SD or uSD. The cards are used in mobile phones. They are also used in newer types of handheld GPS devices, portable media players, digital audio players, expandable USB flash drives, Nintendo DS flashcards, and digital cameras.

It is the smallest memory card that can be bought; at 15 mm \times 11 mm \times 1 mm (about the size of a fingernail), it is about a quarter of the size of a normal-sized SD card. There are adapters that make the small microSD able to fit in devices that have slots for standard SD, miniSD, Memory Stick Duo card, and even USB. But, not all of the different cards can work together. Many microSD cards are sold with a standard SD adapter, so that people can use them in devices that take standard SD

but not microSD cards.

TransFlash and microSD cards are the same (they can be used in place of each other), but microSD has support for SDIO mode. This lets microSD cards do non-memory jobs like Bluetooth, GPS, and Near Field Communication.

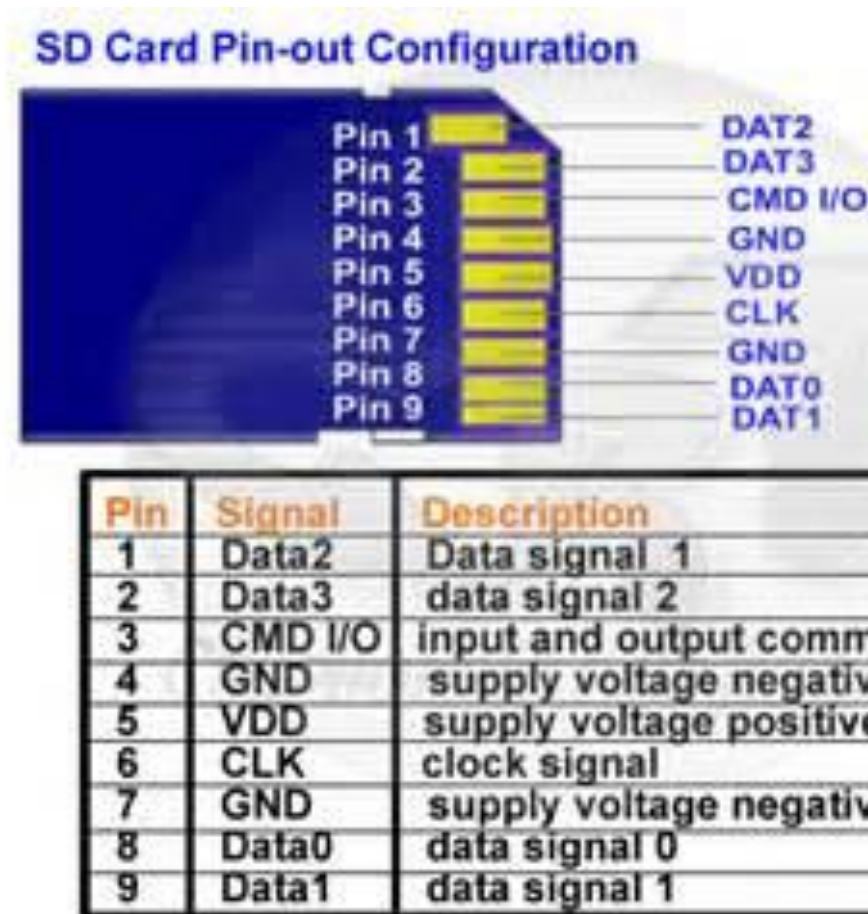


Figure 5.1: SD card

5.5 .WAV Files

Waveform Audio File Format (WAVE, or more commonly known as WAV due to its filename extension) (rarely, Audio for Windows) is a Microsoft and IBM audio file format standard for storing an audio bitstream on PCs. It is an application of the Resource Interchange File Format (RIFF) bitstream format method for storing data in "chunks", and thus is also close to the 8SVX and the AIFF format used on Amiga

and Macintosh computers, respectively. It is the main format used on Windows systems for raw and typically uncompressed audio. The usual bitstream encoding is the linear pulse-code modulation (LPCM) format.

CHAPTER 6
PROJECT MANAGEMENT

PROJECT MANAGEMENT:

6.1 Work flow

For this project, there are five stages of development; project planning, design and development, combining, test the effectiveness and lastly troubleshooting. This talking calculator complete when all the five stage is done. Figure 6.1 shows the flowchart on work progress. Figure shows the details the flowchart for overall part to complete this talking calculator. Figure shows the Gantt chart for progress in according to the task and week during these two semesters.

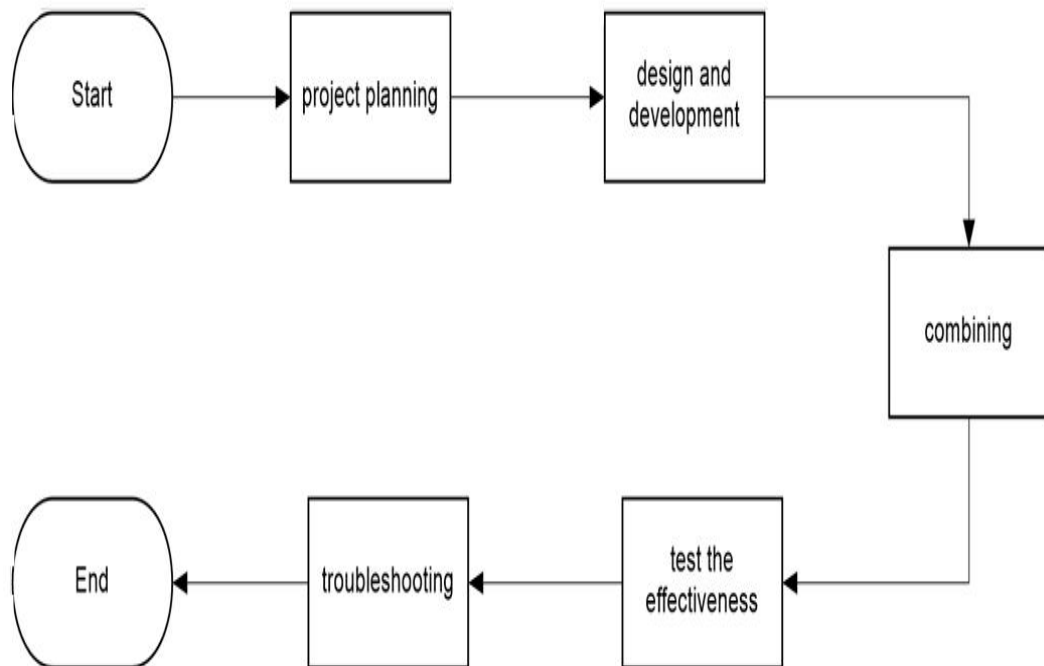


Figure 6.1: Flowchart of work process

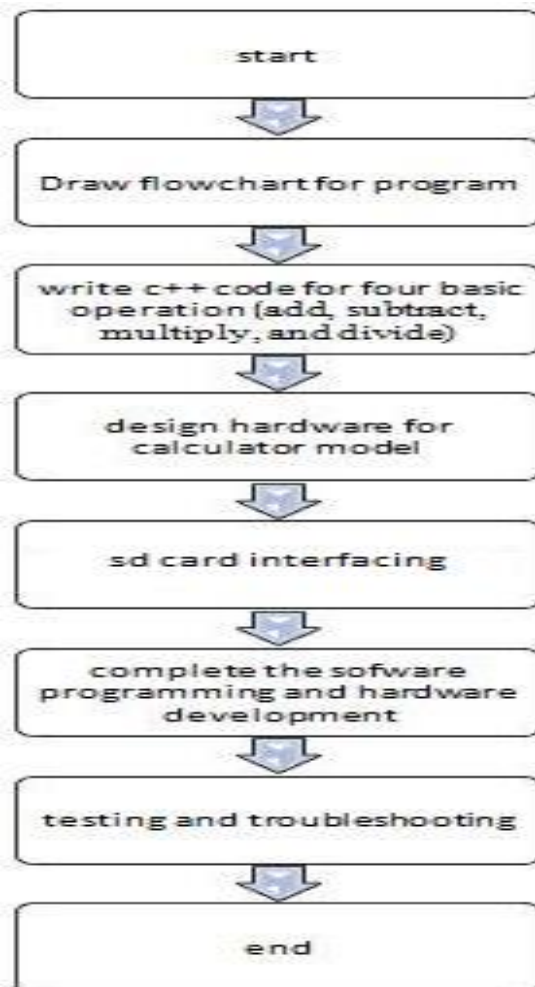


Figure 6.2: Detail work flow

The details project flow is shown in Figure 6.2. The project started by deciding what operation are needed for this talking calculator and get the algorithm for this talking calculator. Then, the step is build the code for four basic operation for talking calculator and this is the most important part of this project. This is because at this part need more time and testing and more friendly system. It will make the low user frustration for use this calculator. The system for the integrated the sound and display also do at this part.

After complete built the code for talking calculator, the designing model of talking calculator taking a part. Also, the design hardware model is important because the pocket size of calculator easier for the disabled to use it. Then, the sd card interfacing is followed. After completed the hardware and software part, combine all the part is the last part before testing and troubleshooting for this project.

CHAPTER 7

RESULTS AND DISCUSSION

RESULTS:

Addition: The result given by audio output for the operation $3 + 2$ is given by the audio output.

“Three plus two equals to five point zero zero.”

Subtraction: The result given by audio output for the operation $3 - 2$ is given by “Three minus two equals to one point zero zero.”

Multiplication: The result given by audio output for the operation $3 * 2$ is given by “Three multiplied by two equals to six point zero zero.”

Division: The result given by audio output for the operation $3 / 2$ is given by “Three divided by two equals to one point five zero.”

It also performs various scientific calculations such as sin, cosine, tan, sine inverse, cosine inverse, tangent, inverse, factorial, logarithmic, antilog, inverse, etc

There is also provision for a clear button, denoted by the only smooth key of the keypad (denoted as ‘C’). It clears the last digit entered, which allows for rectification of any errors during input.

Since it is the only smooth key, it can be identified easily, even by the visually impaired. The figure below shows one of our fellas checking the unit using headphone.

7.1 Portability

Based on Figure 4.5, it shows that 50% of the respondents disagree that this talking calculator is portable. Another 30% of respondents say average for portability this talking calculator. Next 10% respondents agree and strongly agree that this talking calculator portable for blind people. After take results for the interview, respondents disagree about the portability of this device. The portability of this talking calculator is still to be adjusting to make it really easy to carry for everyday uses. To overcome this problem, the size of this device should be adjust

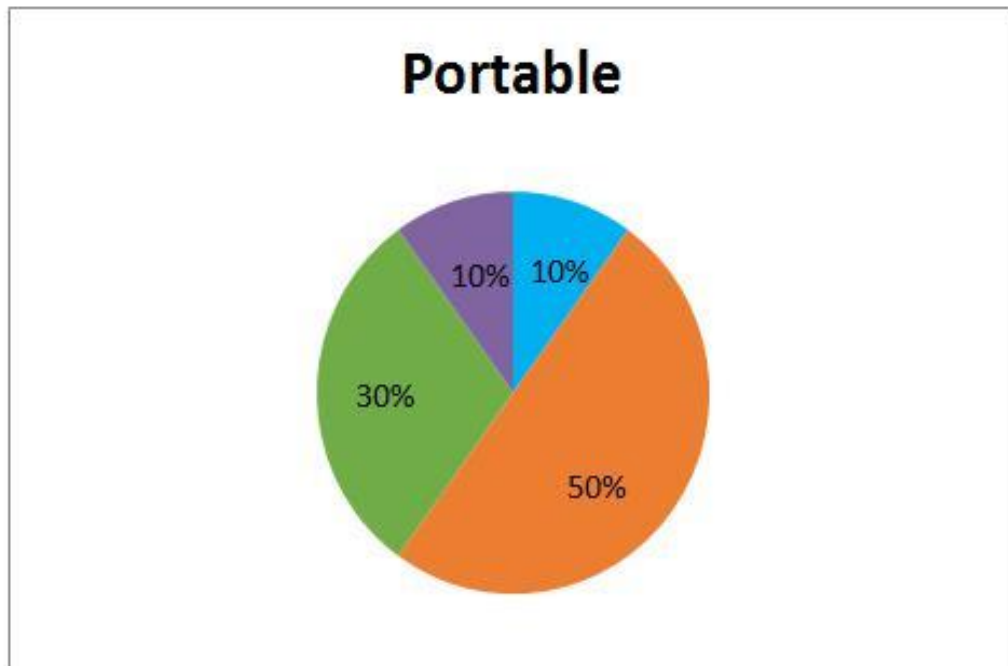


Figure 7.1: Pie chart for portable results on survey fo

7.2 User-friendly

As can be seen at figure 4.6, 50% of the respondents can be categorized agree that this device is user-friendly, while about 10% disagree with the fact. After getting the result from interview, the respondents agree with the device is user friendly because they say it can be used for blind people for children and adult. Next, it really easy to use, the user just have to switch on the device and have an instruction for using this device. The sound of instruction will guide to use this device. The instruction is provided at the talking calculator. It is literally for blind people. The instruction will guide the user to use this talking calculator.

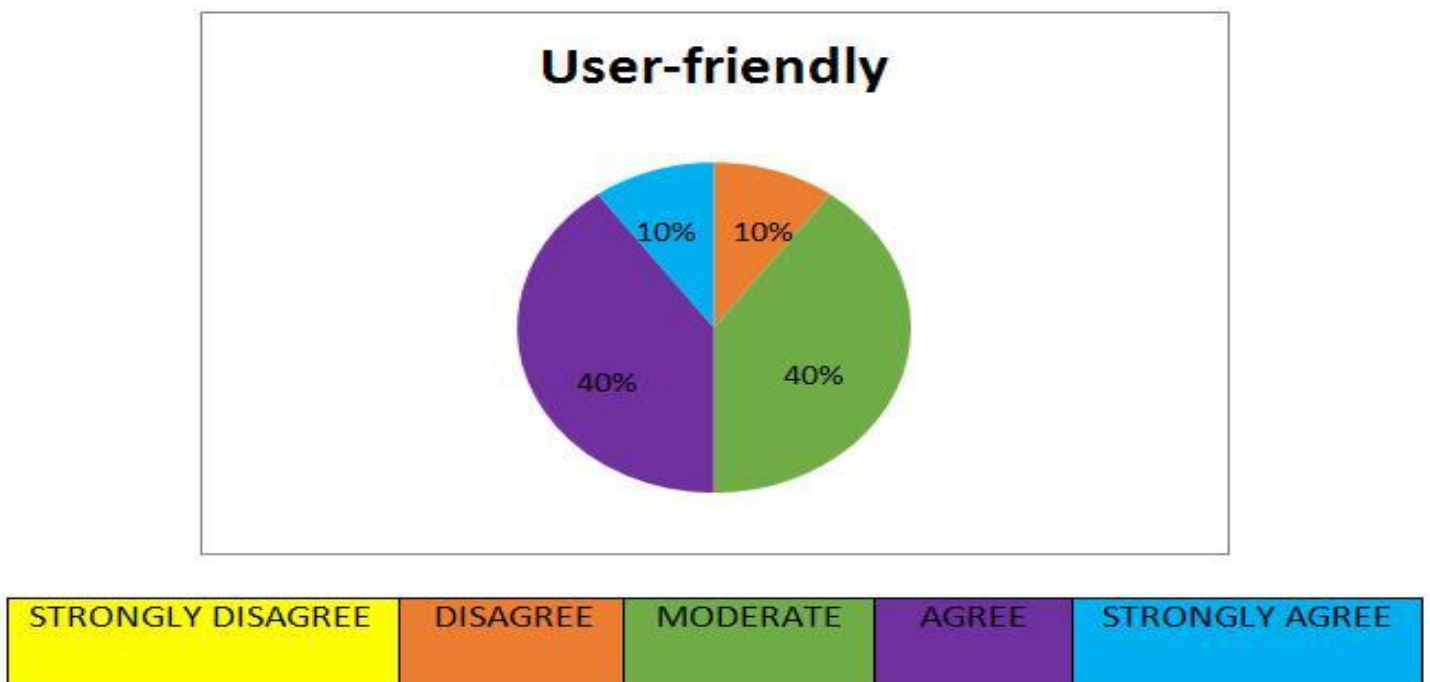


Figure7.2: Pie chart for user-friendly results on survey form

7.3 Cost

From the figure , it can be explained that 50% of the respondents agree this device is cheaper than the one currently sold at market. Another 10% of the respondents disagree this device is still expensive for blind people to buy it. The details cost for this project will be explained at project management.

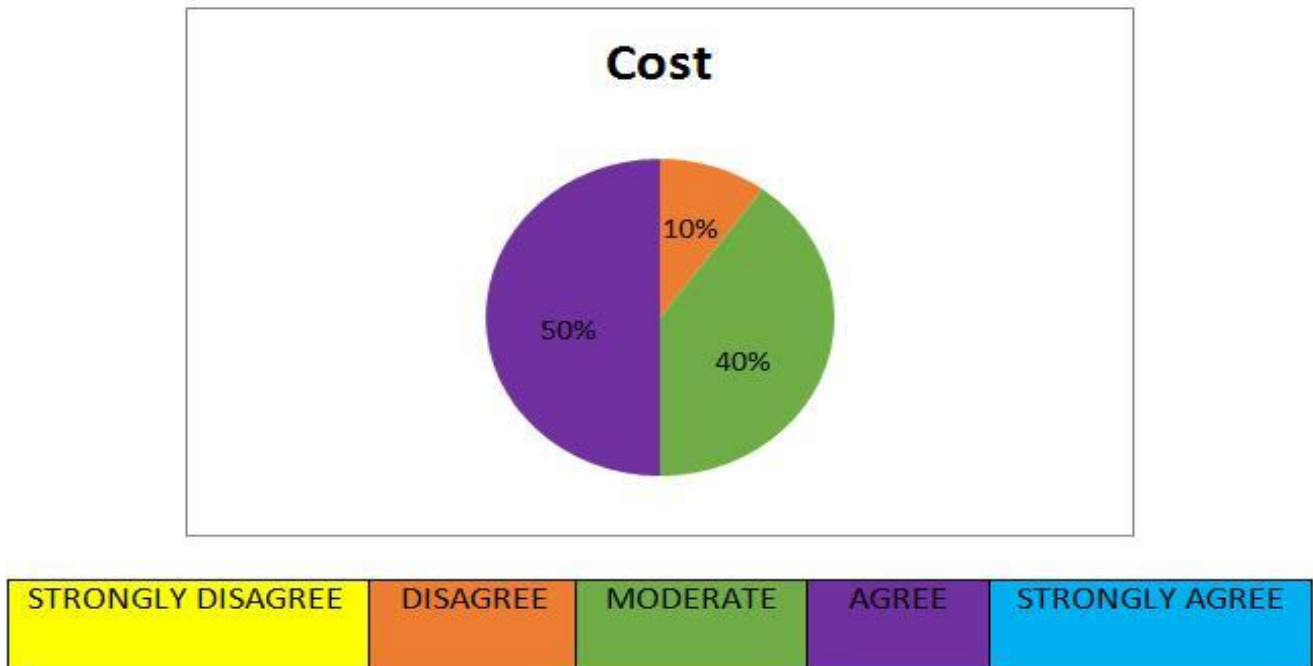


Figure 7.3: Pie chart for cost results on survey form

7.4 Effectiveness

Figure indicates how people response to the effectiveness of this device to blind people. As a result, 70% agreed that this device is very useful for blind people. No one of the respondents say this device is not effective. The effectiveness of this device was measured on how the calculator can solve a simple operation and the sound at the headphone for the answer of the question. The sound instruction is the major factor why respondents agreed that this device is effective.

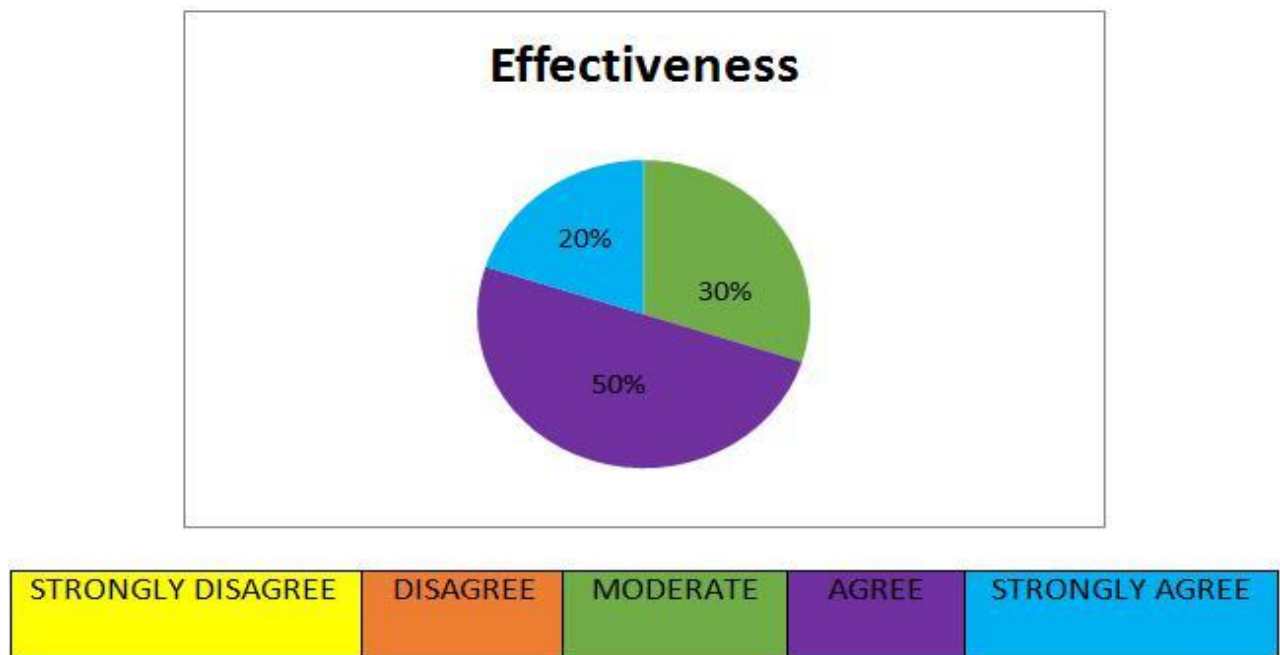


Figure 7.3: Pie chart for effectiveness results on survey form

CHAPTER 8
LIST OF COMPONENTS

LIST OF COMPONENTS:

SR. NO.	COMPONENTS	PRICE(in Rs.)
1	Arduino Due	1500
2	LCD 20x4	350
3	SD card module	200
4	Micro SD card	75
5	Keypad	200
6	10k potentiometer	10
7	Resistors	10
8	Push buttons	50
9	Jumper wires	275
10	Headphones	40
11	Battery	20
12	Head phone socket	10
13	Zero PCB	50
14	Switch	10
	TOTAL	2800

Table 8.1: List of components

CHAPTER 9

CONCLUSION

CONCLUSION:

As a conclusion, this device and program was designed to lightweight and low cost to support blind people for having this talking calculator. This talking calculator can be affordable for blind people to have at their bag for everyday used. This device also gives an advantage for blind people to improve their learning in mathematics and improve their skill to learning soft skill to use the new technology.

This method actually has been doing but for this design it more suitable and easy for blind people especially children, they really like a new technology. The important thing, the blind people can buy this with a low price and can lower their burden.

This project is really good to do for further work because many functions using arduino software. Arduino is a new microcontroller that can make any programs.

CHAPTER 10

FUTURE SCOPE

FUTURE SCOPE:

Hence we implemented scientific calculator for blind people it is easy to handle so that any type mathematical problem can be solved instantly.

In future we are planning to give input in the form of audio instead of traditional keypad input using algorithms of human interfacing i.e. by using IC such as HM2007, which is used to identify commands through audio/voice input.

CHAPTER 11

REFERENCES

REFERENCES:

- Saviour, R., E., Brugler, J., S., and Bliss, J. C., "Development of a Hand-held Talking Calculator for the Blind". *Proceeding from AFIPS conference on American Federation of Information Processing Societies*, California. 2010. vol. 45: pp. 221.
- Yoshit V. Gidh, Mahesh S. Latey, Arpita Roy, Kunal Shah, Savita , "Braille Calculator", *International Journal Of Engineering And Computer Science* ISSN:2319-7242 Volume 2 Issue 2 Feb 2013 Page No. 481-382
- Sunil Kumar M.E –Karnataka University Braille language learner for blinds *International Journal Of infinite Innovations in technology* 2012-2013Reg. No.:20120905|DOI:V1I2P05
- <http://arduino.cc/en/Main/arduinoBoardDue>
- <http://en.wikipedia.org/wiki/Braille>