

CHAPTER 1

INTRODUCTION

1.1 EMBEDDED SYSTEM

An embedded system is a computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, is designed to be flexible and to meet a wide range of an end-user's needs. Embedded systems control many of the common devices in use today. Embedded systems are controlled by a main processing core that is typically either a microcontroller or a digital signal processor (DSP). Since the embedded system is dedicated to specific tasks, design engineers can optimize it reducing the size and cost of the product and increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically, embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure. In general, "embedded system" is not an exactly defined term, as many systems have some element of programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them, but are not truly embedded systems, because they allow different applications to be loaded and

An embedded system is one kind of a computer system mainly designed to perform several tasks like to access, process and store and also control the

data in various electronics-based systems. Embedded systems are a combination of hardware and software where software is usually known as firmware that is embedded into the hardware. One of its most important characteristics of these systems is, it gives the o/p within the time limits. Embedded systems support to make the work more perfect and convenient. So, we frequently use embedded systems in simple and complex devices too. The applications of embedded systems mainly involve in our real life for several devices like microwave, calculators, TV remote control, home security and neighbourhood traffic control systems, etc.

1.1.1 RTOS (Real Time Operating System)

A system which is essential to finish its task and send its service on time, then only it said to be a real time operating system. RTOS controls the application software and affords a device to allow the processor run. It is responsible for managing the different hardware resources of a personal computer (PC) and also host applications which run on the PC.

This operating system is specially designed to run various applications with an exact timing and a huge amount of consistency. Particularly, this can be significant in measurement & industrial automation systems where a delay of a program could cause a safety hazard.

1.1.2 Embedded system characteristics

- Generally, an embedded system executes a particular operation and does the similar continually. For instance: A pager is constantly functioning as a pager.
- All the computing systems have limitations on design metrics, but those can be especially tight. Design metric is a measure of an execution features like size, power, cost and also performance.

- It must perform fast enough and consume less power to increase battery life.
- Several embedded systems should constantly react to changes in the system and also calculate particular results in real time without any delay. For instance, a car cruise controller; it continuously displays and responds to speed & brake sensors.
- It must be based on a microcontroller or microprocessor based.
- It must require a memory, as its software generally inserts in ROM. It does not require any secondary memories in the PC.
- It must need connected peripherals to attach input & output devices.
- An Embedded system is inbuilt with hardware and software where the hardware is used for security and performance and Software is used for more flexibility and features.

1.2 CNC MACHINE TOOLS

The Computer Numerical Control (CNC) machining is a process used in the manufacturing sector that involves the use of computers to control machine tools. Under CNC machining, machine tools function through numerical control. A computer program is customized for an object and the machines are programmed with language called G-code that essentially controls all features like feed rate, coordination, axis motions, location and speed.

1.3 ADVANTAGES AND DISADVANTAGES OF CNC MACHINING

1.3.1 Advantages

- CNC machines can be used continuously 24 hours a day, 365 days a year and only need to be switched off for occasional maintenance.

- Less skilled/trained people can operate CNCs unlike manual lathes / milling machines which need skilled engineers.
- CNC machines can be updated by improving the software used to drive the machines.
- Modern design software allows the designer to simulate the manufacture of his/her idea. There is no need to make a prototype or a model. This saves time and money.
- One person can supervise many CNC machines as once they are programmed they can usually be left to work by themselves.

1.3.2 Disadvantages

- CNC machines are more expensive than manually operated machines, although costs are slowly coming down.

1.3.3 Applications of CNC Machines

- The machines remove excess metal from raw materials to create complex parts. A good example of this would be the automotive industries where gears, shafts and other complex parts are carved from the raw material.
- CNC machines are also used in the manufacturing industries for producing rectangular, square, rounded and even threaded jobs.
- These industries use CNC machines for a number of machining operations such as plasma or flame cutting, laser cutting, shearing, forming and welding to create these plates.

CHAPTER 2

LITERATURE SURVEY

2.1 SUEVEY PAPERS

a) TITLE : Design and Development of ARDUINO controlled writing robot

JOURNAL : International Research Journal of Engineering and Technology
(IRJET)

YEAR : Apr -2017

AUTHOR : R.Balathangam, P.Mathipriya, R.Pavithra, G.Prithiviraj,
U.Poornima

CONTENT:

The main aim of this project is to develop a writing robot in order to help the education system to make more interesting by speech recognition technique. Speech recognition has been implemented using ARDUINO microcontroller (ATMEGA328). So in this paper a new idea has been proposed to implement robots in education. The automation is playing important role to save human efforts in most of the regular and frequently carried works e.g. most of the industrial jobs like welding, painting, assembly, container filling etc.

ADVANTAGES

- (i) This project made innovation and interest in education field.
- (ii) Also used to develop the curiosity of learning.

DISADVANTAGES

- (i) This neuro fuzzy technique entirely based on trial and error method.

b) TITLE : G-Code Controlled 2D Robotic Plotter

JOURNAL : International Journal on Recent and Innovation Trends in
Computing and Communication (IJRITCC)

YEAR : Dec-2017

AUTHOR : T.Jobby Titus,P.Vinitha, M.Nivetha, M.Vasanthaalakshmi

CONTENT:

2D Robotic Plotter is implemented based on the principle of Computer Numeric Control (CNC). Normally, Computer Aided Design software (CAD) can be used as the replacement for manual drafting with an automated process. A 2D Robotic Plotter basically works with two stepped motors and a Servo motor with the control of ATMEGA168p Microcontroller. In our proposed methodology an ARDUINO IDE platform controls the Motor Driver Shield (L293D) for the movement of the pen in the x and y direction to the rotation in stepper motors. In Ink-scape (0.48.5) is open source software that binds the program along with the diagram for the efficient plotting. Also G-code is a numeric control programming language which is used mainly in computer aided manufacturing to control automated machine tools. The 2D robotic plotter is a low cost embedded equivalent CNC controller and can be widely used in areas where the accuracy is considered

ADVANTAGES

- (i) INKSCAPE tool is effectively utilized to generate this drawing control code.

DISADVANTAGES

- (i) Two stepper motor and servo motor control is utilized to obtain this movement. So system bulky

c) TITLE : Arduino based cost effective CNC plottermachine

JOURNAL: International Journal of Emerging Technologies in Engineering
Research (IJETER)

YEAR : Feb - 2018

AUTHOR : Arpita Chirde, Puja Girhe, Shubham Yenkar

CONTENT:

Due to the rapid growth of technology the usage & utilization of CNC machine in industries are increased. The fabrication of low cost CNC machine is used to reduce cost and complexity of machine. This paper deals with the design of automatic mini CNC machine for PCB drawing and drilling. The Idea behind our project is to design and drill PCB based on low cost CNC system the lower cost is achieved by incorporating features of PC with ATMEGA 328 controller in an arduino. We have use an G code for whole system operation G code is nothing but a language in which people tell computerized machine tools 'How to make something'.

ADVANTAGES

- (i) It consumes low power and works with high accuracy due to precise controlling of stepper motors. This is a low cost project as compared to other CNC product.
- (ii) It is designed for private manufacturing and small scale applications in educational institutes.

DISADVANTAGES

- (i) Stepper motor used here. So speed of operation is slow.

d) TITLE : Microcontroller – Based Plotter Machine

JOURNAL : Al-Nahrain Journal for Engineering Sciences (NJES)

YEAR : Apr -2018

AUTHOR : Aman Ismail Nsayef, AnasLateefMahmood

CONTENT:

The main idea behind this paper is to design and implement a cheap, smaller size, easily operable, easy interface and flexible 3-axis Computer Numerical Control (CNC) plotter machine. The lower cost is achieved by using 2CD drives from old PC's with their stepper motors as the main structure for the hardware. The two stepper motors already found in the CD drives used to control the pen movements onto X and Y axis and one servo motor on the Z axis. An Arduino Uno microcontroller is used to controls the proper synchronization of these three motors during printing/drawing process. The plotter machine is implemented and tested by printed different images and texts on papers (8cm × 8cm) using a pen, the small size of the papers because of the small plotter size. The motors winding voltages were displayed on the oscilloscope during the printing process to investigate the synchronization between the three motors. The design of the circuit is simple, inexpensive and can be accomplished using commercially available components.

ADVANTAGES

- (i) Investigation static rigidity, positioning accuracy and repeatability.
- (ii)The device needs to be very stable during printing process because any simple vibration can cause errors in printing

DISADVANTAGES

- (i) Difficult to maintain and requires highly skilled operators.

2.2 OVERVIEW OF LITERATURE SURVEY

YEAR	TITLE	JOURNAL	TECHNIQUES USED	ADVANTAGES	DISADVANTAGES
Apr - 2017	Design and Development of ARDUINO controlled writing robot	International Research Journal of Engineering and Technology (IRJET)	Using ARDUINO microcontroller (ATMEGA328)	Used to develop the curiosity of learning	This Neuro fuzzy technique entirely based on trial and error method
Dec- 2017	G-Code Controlled 2D Robotic Plotter	International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC)	CAD can be used as the replacement for manual drafting with an automated process	INKSCAPE tool is effectively utilized	System bulky
Feb - 2018	Arduino based cost effective CNC plotter machine	International Journal of Emerging Technologies in Engineering Research (IJETER)	Use an code for whole system operation	Low power with high accuracy Low cost project	Stepper motor used here. So speed of operation is slow
Apr - 2018	Microcontroller – Based Plotter Machine	Al-Nahrain Journal for Engineering Sciences (NJES)	The lower cost is achieved by using 2 CD drives from old PC's with their stepper motors	Positioning accuracy and repeatability	Difficult to maintain and requires highly skilled operators

Table No. 2.1 Summary of Literature Survey

CHAPTER 3

EXISTING SYSTEM

3.1 EXISTING SYSTEM TECHNIQUE

The technological advancements in robotics field, efforts are being taken in researching, designing and development of robots for different practical purposes. Robots designed to assist human in their work and reduced human efforts. Nowadays, robots are designed to mimic human behaviour and perform tasks similar to human. Many research companies are developing robotic arm for performing basic functions like human arm. Among different functions, writing skills is one of function. The proposed robotic arm can be used by physically challenged person for writing operation. The main aim of developing the proposed system is to facilitate the physically challenged persons to write what they speak. Presently, the physically challenged persons need a scribe/paper writer during exams to write their examinations. It is very hectic work to find out the writer.

3.2 BLOCK DIAGRAM

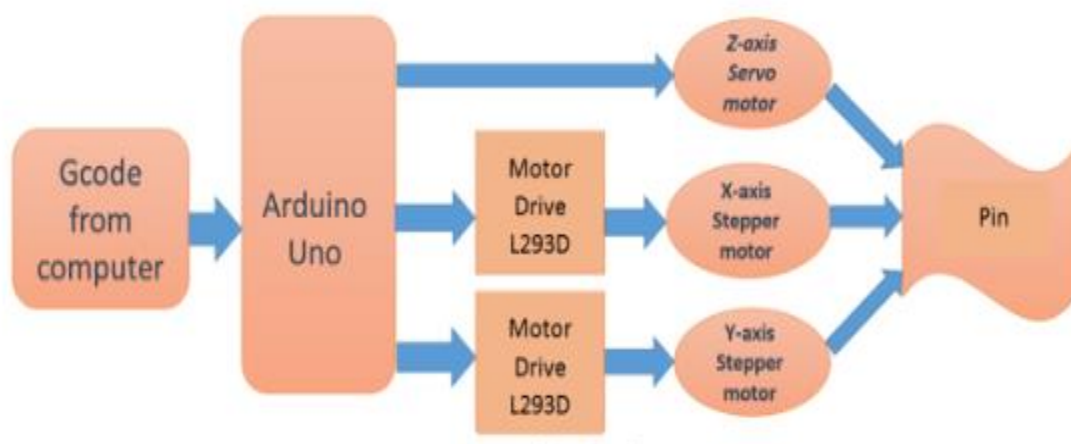


Fig. No. 3.1 Block Diagram of Existing sytem

3.3 CNC PLOTTER DESIGN

The above figure shows the circuit block diagram of the designed plotter machine. There are three axes namely X, Y and Z which we have to control. X and Y axes controls the motion of worktable while Z axis controls the motion of pen upward or downward. The PC program converts the text into G-code and sending it sequentially to the microcontroller. The Arduino start to controls the three motors according to the received G-code locations with a proper synchronization of these three axes motors during printing.

The two stepper motors taken from an old two DVD/CD drive represent X and Y axes controls the motion of work table and one servo motor represent Z axis for controls the motion of pen up and down. The movement control for these three motors is done through microcontroller, the Arduino microcontroller is used in this work to control motors rotation. The Arduino microcontroller is an open source microcontroller, easy to use, has a good number of input/output ports with suitable memory size and can easily interact with computer physical computing platform for creating interactive objects that standalone or collaborate with software on the computer. It has gained considerable attraction in the professional market.

The microcontroller used in the hardware implementation is Arduino Uno microcontroller. The microcontroller connected with a PC through USB serial port in order to receive the G-code from the PC. The Arduino here acts as the brain that controls the speed and directions of the stepper and servo motors. In the PC a C language program is programmed to generate the control signals in the G-code and transmit these codes to the Arduino microcontroller through G-code interpreter via USB port. These command signals directly controls the motion and speed of the three motors in order to controls the drawing tool.

The stepper motors can rotate clockwise or counter clockwise direction with discrete steps. The appropriate voltages sequence that energizing the motor coils is responsible about motor direction of rotation. The Arduino microcontroller generates these voltages sequence and drives the stepper motors with it through drive circuit. The drive circuits receive voltage signals from Arduino ports and these signals controls the speed and direction of motors rotations. The stepper motors consider the heart of CNC plotter because the size and type of motor speed and accuracy depend on it. Servo motor tiny and light weight DC motor, it can rotate 90 degrees in each direction, in order to rotate the servo motor 90 degree a 1.5 ms (about 5V) pulse is applied to its control pin from a total period of 20ms. The servo motor is used to left the pen up or down on the paper, because that the current from the servomotor is too small and the Arduino port can hold this small current there is no need for a drive circuit. This motor has three legs (VCC, GND, and Control) the microcontroller sends an ON/OFF switch signal to the control pin of the servo motor in order to start or stop motor rotation. The GND (-) pin has brown colour and the VCC (+) has red colour.

The computer (PC) is connected to the Arduino-based circuit via the USB serial port. Two L293 motor drive ICs connect the two stepper motors to Arduino at pins 2,3,4,5,8,9,10,11. because the current limitation of the Arduino output port pin to about 40 mA while the stepper motor draw more than 150 mA. The drawing current of the stepper motor varied around this typical value when the motor rotate at low or high step speed. The control pin (Orange color) of the servo motor received the pulse signal from pin 6 (PWM) of the Arduino Uno microcontroller.

3.4 STEPPER MOTOR OPERATION

When the stepper motors switch ON it draws high starting currents because of the motor coils which cause a drawn reactive power and this significant sudden current cause a voltage drop in the supply voltage. While when the motors switch OFF the reactive power drops and the supply voltage rises. The rises and drops in the supply voltage cause the voltage fluctuations that is noticed in the figures and these fluctuations are repetitive or random variations due to sudden changes in the real and reactive power drawn by a load.

3.5 DISADVANTAGES

- This movement of human arm is achieved by using three stepper motors moving in three different directions.
- The working of our robotic arm basically involves two parts. The first part consist reception of speech signal and converting it into text and other part involves mechanical action of motor to obtain written text.

CHAPTER 4

PROPOSED SYSTEM

4.1 INTRODUCTION

The paper proposes the design of writing robotic arm by speech recognition. The objective of the system to build a robotic arm that would showcase the writing skills based on speech recognition is successfully implemented. The developed system can proved helpful for different categories of people, specifically for physically challenged persons to write and express their thoughts in written manner. With the advent of new technology, additional facilities like pick and place can also be embed into it. Voice Interfaced Arduino Robotic arm for detecting the objects and also for classifying the objects is successfully implemented. In this paper we mainly focus on the shapes of the objects for detecting and classifying. In future, it can be further continued for detecting the objects based on the colour, shape deformation, etc.

4.2 BLOCK DIAGRAM

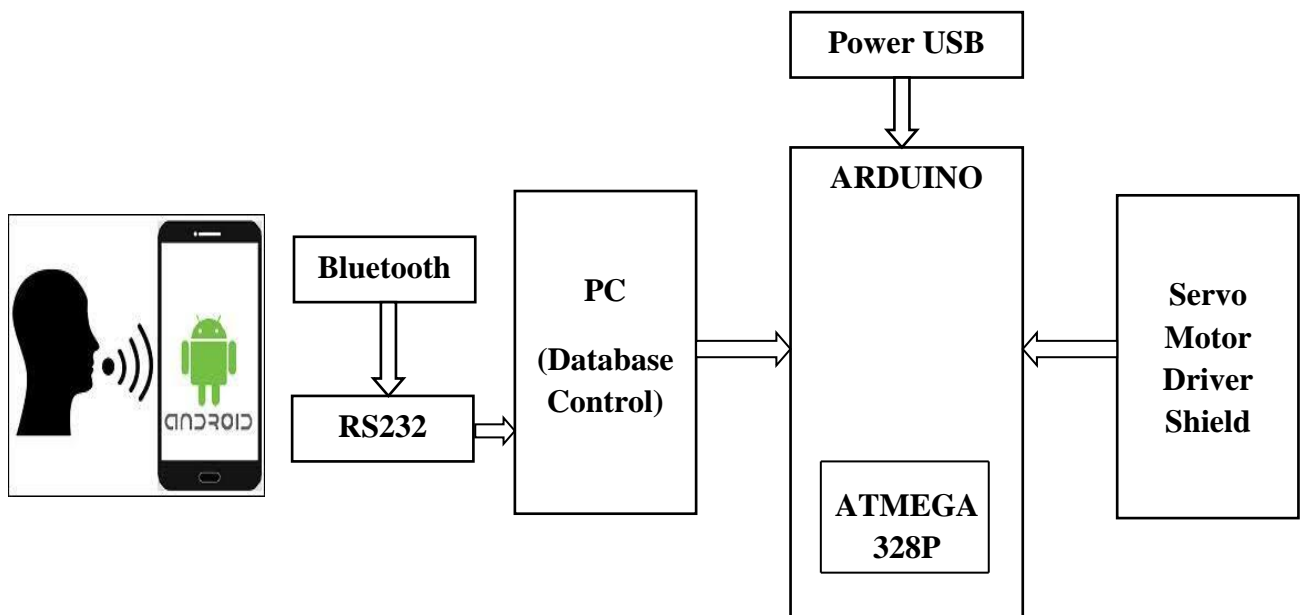


Fig. No.: 4.1 Block diagram of Proposed System

4.3 MODULE SEPARATION

- **HARDWARE REQUIRED**
- **SOFTWARE REQUIRED**
- **LANGUAGE USED**

4.3.1 HARDWARE REQUIRED

- Arduino UNO
- Servo motor driver shield
- Bluetooth
- Servo motor
- Serial to USB cable

4.3.1.1 Arduino UNO

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

The key features are

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).

- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

Arduino - Board Description

In this chapter, we will learn about the different components on the Arduino board. We will study the Arduino UNO board because it is the most popular board in the Arduino board family. In addition, it is the best board to get started with electronics and coding. Some boards look a bit different from the one given below, but most Arduino have majority of these components in common.

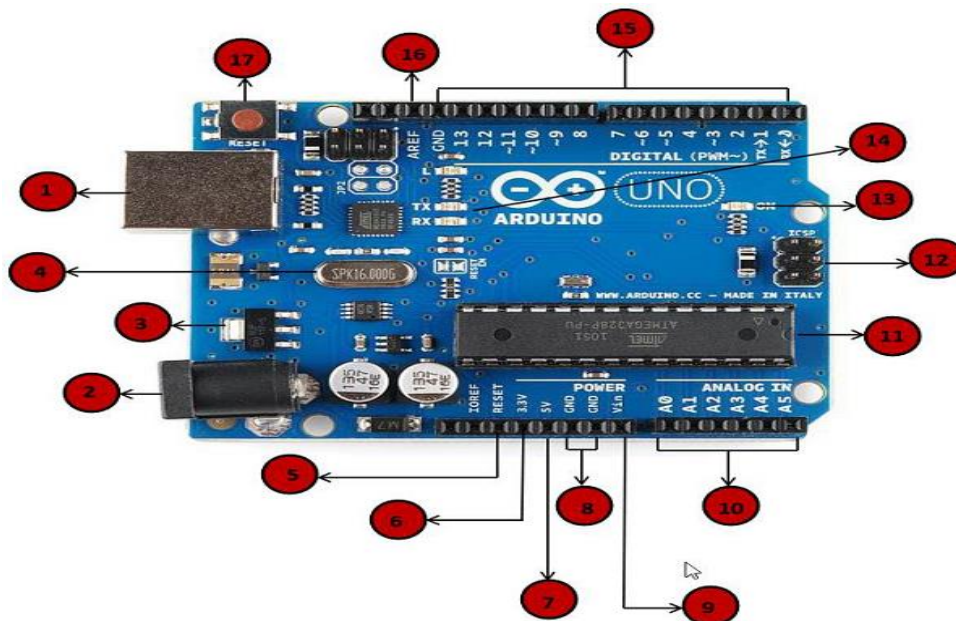














Fig. No.: 4.2 ARDUINO Board Model

Pin No.	Pin Description
	Power USB Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1).
	Power (Barrel Jack) Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).
	Voltage Regulator The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.
	Crystal Oscillator The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.
	Arduino Reset You can reset your Arduino board, i.e., starts your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5).
	Pins (3.3, 5, GND, Vin) 3.3V (6) – Supply 3.3 output volt 5V (7) – Supply 5 output volt Most of the components used with Arduino board works fine with 3.3 volt and 5 volt. GND (8)(Ground) – There are several GND pins on the Arduino, any of which can be used to ground your circuit. Vin (9) – This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

	<p>Analog pins</p> <p>The Arduino UNO board has five analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.</p>
	<p>Main microcontroller</p> <p>Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (Integrated Circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.</p>
	<p>ICSP pin</p> <p>Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.</p>
	<p>Power LED indicator</p> <p>This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.</p>
	<p>TX and RX LEDs</p> <p>On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.</p>
	<p>Digital I/O</p> <p>The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output</p>


	pins to drive different modules like LEDs, relays, etc. The pins labelled “~” can be used to generate PWM.
	AREF AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Table No. 4.1 ARDUINO Board pin description

4.3.1.2 Servo Motor Driver Shield

A **servomotor** is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration.^[1] It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors. Servomotors are not a specific class of motor although the term servomotor is often used to refer to a motor suitable for use in a control system. Servomotors are used in applications such as robotics, CNC machinery or automated manufacturing.

A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft.

The motor is paired with some type of encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero and the motor stops.

The very simplest servomotors use position-only sensing via a potentiometer and bang-bang control of their motor; the motor always rotates at full speed (or is stopped). This type of servomotor is not widely used in industrial motion control, but it forms the basis of the simple and cheap servos used for radio-controlled models.

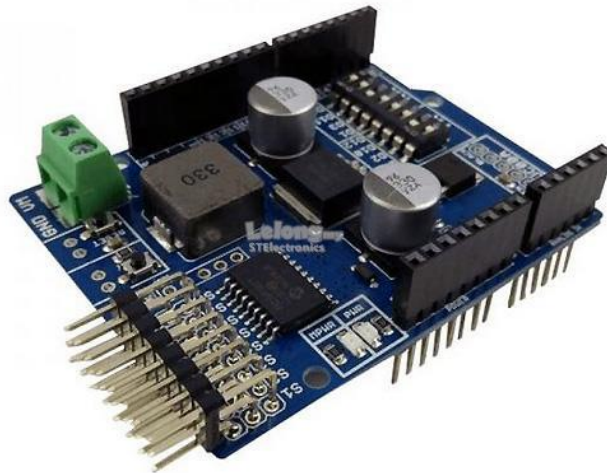


Fig. No.: 4.3 Servo motor driver shield model

More sophisticated servomotors use optical rotary encoders to measure the speed of the output shaft and a variable-speed drive to control the motor speed. Both of these enhancements, usually in combination with a PID control algorithm, allow the servomotor to be brought to its commanded position more quickly and more precisely, with less overshooting. Servo motor driver may be used to control a single servo or even a group of servo motors. In many projects where servo motor controlling is the mainstay of the task to be accomplished, the controller must drive more than one servo. An example of this is an RC airplane, which uses many servos.

Essential Components

- A micro-controller
- A power supply unit

Miscellaneous Components

- A potentiometer
- Connectors, wires etc.

4.3.1.3 Bluetooth

Bluetooth is a wireless technology standard for exchanging data over short distances using short-wavelength UHF radio waves in the ISM band from 2.400 to 2.485 GHz from fixed and mobile devices, and building Personal Area Networks (PANs). It was originally conceived as a wireless alternative to RS-232 data cables.

Bluetooth is managed by the Bluetooth Special Interest Group (SIG), which has more than 30,000 member companies in the areas of telecommunication, computing, networking, and consumer electronics. The IEEE standardized Bluetooth as IEEE 802.15.1, but no longer maintains the standard. The Bluetooth SIG oversees development of the specification, manages the qualification program, and protects the trademarks. A manufacturer must meet Bluetooth SIG standards to market it as a Bluetooth device. A network of patents apply to the technology, which are licensed to individual qualifying devices

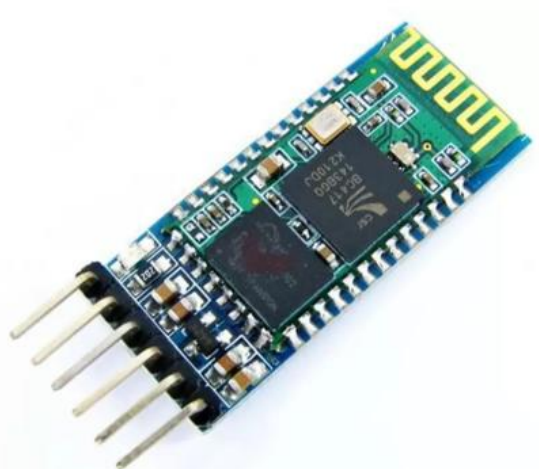


Fig. No.: 4.4 Arduino Bluetooth device

4.3.1.4 Servo Motor

A **servomotor** is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.

Servo motors are DC motors that allows for precise control of angular position. They are actually DC motors whose speed is slowly lowered by the gears. The servo motors usually have a revolution cutoff from 90° to 180° . A few servo motors also have revolution cutoff of 360° or more. But servo motors do not rotate constantly. Their rotation is limited in between the fixed angles.

Servomotors are not a specific class of motor although the term servomotor is often used to refer to a motor suitable for use in a closed-loop controlsystem. Servomotors are used in applications such as robotics, CNC machinery or automated manufacturing.



Fig. No.: 4.5 Image of Industrial servomotors

The servo motor is actually an assembly of four things: a normal DC motor, a gear reduction unit, a position-sensing device and a control circuit. The DC motor is connected with a gear mechanism which provides feedback to a position sensor which is mostly a potentiometer. From the gear box, the output of the motor is delivered via servo spine to the servo arm. For standard servo

motors, the gear is normally made up of plastic whereas for high power servos, the gear is made up of metal.

A servo motor consists of three wires- a black wire connected to ground, a white/yellow wire connected to control unit and a red wire connected to power supply. The function of the servo motor is to receive a control signal that represents a desired output position of the servo shaft and apply power to its DC motor until its shaft turns to that position.

It uses the position sensing device to figure out the rotational position of the shaft, so it knows which way the motor must turn to move the shaft to the instructed position. The shaft commonly does not rotate freely around similar to a DC motor, however rather can just turn 200 degrees.

From the position of the rotor, a rotating magnetic field is created to efficiently generate torque. Current flows in the winding to create a rotating magnetic field. The shaft transmits the motor output power. The load is driven through the transfer mechanism. A high-function rare earth or other permanent magnet is positioned externally to the shaft. The optical encoder always watches the number of rotations and the position of the shaft.

Working of a Servo Motor

The Servo Motor basically consists of a DC Motor, a Gear system, a position sensor and a control circuit. The DC motors get powered from a battery and run at high speed and low torque. The Gear and shaft assembly connected to the DC motors lower this speed into sufficient speed and higher torque. The position sensor senses the position of the shaft from its definite position and feeds the information to the control circuit. The control circuit accordingly decodes the signals from the position sensor and compares the actual position of the motors with the desired position and accordingly controls the direction of

rotation of the DC motor to get the required position. The Servo Motor generally requires DC supply of 4.8V to 6 V.



Fig. No.: 4.6 Servo Motor model

4.3.1.5 USB to serial RS232 adapters

DB9-USB-RS232 modules are designed to directly replace an existing DB9 (the industry accepted name for a DE9 connector) RS232 connection with a drop-in USB replacement connection. Pin for pin, these modules will replace a male or female DB9 RS232 connector with a USB mini-B connector. The application PCB containing the existing DB9 RS232 connector does not require any modification, only the replacement of the D-type connector with the appropriate DB9-USB-RS232 module. A male DB9 should be replaced by a male DB9-USB-RS232-M and a female DB9 should be replaced by a female DB9-USB-RS232-F. The modules contain all necessary electronics to convert between USB and RS232.

The purpose of the modules is to provide a simple method of adapting legacy serial devices with RS232 interfaces to modern USB ports by replacing the DB9 connector with this miniaturized module that closely resembles a DB9 connector. This is accomplished by incorporating the industry standard FTDI FT232R USB-Serial Bridge IC plus the required level shifters inside the module.

The module uses a standard USB-MINI-B connector for connection to an upstream host or hub port. RS232-level signals, including modem handshake signals, can transmit data at rates up to 1MBd.

Features

- Adds one USB serial port by connecting to the RS232 DB9 footprint of a device
- Easy placement for an standard male and female RS232 DB9 footprint of a device
- Works with USB 1.1 & 2.0 host and hub ports
- Industry standard FTDI chip set and device drivers for maximum compatibility
- Microsoft Windows® WHQL-certified, Mac OS X, Linux and Windows CE device drivers
- Installs as a standard Windows COM port
- Supports Windows servers 2008, 2003, Vista, XP 2000, Linux, MacOSX
- 128 byte transmit buffer, 256 byte receive buffer
- RS-232 data signals: TxD, RxD, RTS, CTS, DSR, DTR, DCD, RI, GND
- Powered by USB port. No external power adapter required.
- Serial port speed up to 1 Mbps
- Serial communication parameters
 - Parity: none, even, odd
 - Data bits: 7, 8
 - Flow control: RTS/CTS , DSR/DTR, X-ON/X-OFF, none
- Operating temperature of -40°C to +85°C



Fig. No.: 4.7 DB9-USB-RS232

The app Voice Access allows users to navigate through apps, write and edit text, and talk to the Google Assistant. Users can get more specific with controls, like being able to use their voice to tap buttons or adjust controls within apps. They can also scroll or navigate app screens.

If a user wants to write or edit text, for example, they can start by saying “OK Google,” and then open their preferred app with an “open” command. The interface mostly relies on numbers assigned to areas they can interact with, so users will then select where they want to type by speaking the number Voice Access has assigned. Users can then say their message aloud and edit it as they go. They can say “delete the line” or “undo,” or say “stop listening” once they finish writing. Google lists a bunch of commands on its Support Page, including how to adjust specific phone settings and general commands.

Although Google says it focused on developing the app for people with Parkinson’s disease, multiple sclerosis, arthritis, and spinal cord injuries, it also notes that people who have their hands full could use it, too. For now, the app only supports English, but additional languages are coming in the future.

4.3.2 SOFTWARE REQUIRED

- Voice control android APP
- Visual studio software
- Arduino ide
- G code software

4.3.2.1 Voice control android APP

Control your Arduino with voice commands using an Android smartphone. Before we make a voice activated home automation system, we must first learn the basic principles of the experiment. This guide will let you command the Arduino using your Android smartphone and a HC-05 Bluetooth module.

Android has one and you can use it to control your Arduino, via Bluetooth. The App works by pressing the MIC button, then it will wait for you to say a command. The app will then display the word's that you've stated and will send data strings for the Arduino to process.

Before you program the arduino, you must first learn how the app works. The app work by recognizing your voice command, it will then display the words that you've spoken then sending data/ strings to the Arduino via Bluetooth. A string is like a word, you can make conditional statements out of it [ex: `if (voice == "*computer on") { // turn Pin #2 on }`]. The "voice" is your string, "==" is your condition (means equal to), "*computer on" is your command and the code inside the curly-braces "{ }" are the codes to be executed once your string matches the command condition.

The app sends strings in this format ***command#**, the asterisk (*) indicates the start of a new command and the hash-tag (#) indicates the end of a command. I was able to remove the hash-tag (#) after each word in the

conditional statement was not able to remove the asterisk (*). You'll need to start your command condition with an asterisk otherwise the sketch will not work.

Android Software

Android is an open source and Linux-based operating system for mobile devices such as smart phones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies. This tutorial will teach you basic Android programming and will also take you through some advance concepts related to Android application development.

This tutorial has been prepared for the beginners to help them understand basic Android programming. After completing this tutorial you will find yourself at a moderate level of expertise in Android programming from where you can take yourself to next levels.

Android tutorial or android development tutorial covers basic and advanced concepts of android technology. Our android tutorial is developed for beginners and professionals.

Android is a complete set of software for mobile devices such as tablet computers, notebooks, smart phones, electronic book readers, set-top boxes etc.

It contains a **linux-based Operating System, middleware and key mobile applications**. It can be thought of as a mobile operating system. But it is not limited to mobile only. It is currently used in various devices such as mobiles, tablets, televisions etc.

Android is an open source and Linux-based **Operating System** for mobile devices such as smart phones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies.

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.

The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008.

On June 27, 2012, at the Google I/O conference, Google announced the next Android version, 4.1 **Jelly Bean**. Jelly Bean is an incremental update, with the primary aim of improving the user interface, both in terms of functionality and performance.

The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version.

Features of Android

Android is a powerful operating system competing with Apple 4GS and supports great features.

Few of them are listed below:

S. No.	Feature & Description
1	Beautiful UI: Android OS basic screen provides a beautiful and intuitive user interface.
2	Connectivity: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.
3	Storage: SQLite, a lightweight relational database, is used for data storage purposes.
4	Media support: H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF and BMP

5	Messaging: SMS and MMS
6	Web browser: Based on the open-source WebKit layout engine, coupled with Chrome's V8 JavaScript engine supporting HTML5 and CSS3.
7	Multi-touch: Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero.
8	Multi-tasking: User can jump from one task to another and same time various applications can run simultaneously.
9	Resizable widgets: Widgets are resizable, so users can expand them to show more content or shrink them to save space.
10	Multi-Language: Supports single direction and bi-directional text.
11	GCM: Google Cloud Messaging (GCM) is a service that lets developers send short message data to their users on Android devices, without needing a proprietary sync solution.
12	Wi-Fi Direct: A technology that lets apps discover and pair directly, over a high-bandwidth peer-to-peer connection.
13	Android Beam: A popular NFC-based technology that lets users instantly share, just by touching two NFC-enabled phones together.

Table No.: 4.2 Feature & Description of Android



Fig. No.: 4.8 Android developments

4.3.2.2 Visual studio software

Microsoft Visual Studio is an Integrated Development Environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger.

Other built-in tools include a code profiler, forms designer for building GUI applications, web designer, class designer and database schema designer. It accepts plug-ins that enhance the functionality at almost every level including adding support for source control systems (like Subversion and Git) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists.

Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) was supported in the past.

The most basic edition of Visual Studio, the Community edition, is available free of charge. The currently supported Visual Studio version is 2017.

Microsoft announced 2019 on June 6, 2018, with its release timing to be shared "in the coming months," promising "to deliver ... quickly and iteratively."

4.3.2.3 G code software

G-code (also RS-274), which has many variants, is the common name for the most widely used numerical control (NC) programming language. It is used mainly in computer-aided manufacturing to control automated machine tools.

G-code is a language in which people tell computerized machine tools how to make something. The "how" is defined by g-code instructions provided to a machine controller (industrial computer) that tells the motors where to move, how fast to move, and what path to follow.

The two most common situations are that, within a machine tool such as a lathe or mill, a cutting tool is moved according to these instructions through a toolpath cutting away material to leave only the finished workpiece and/or, an unfinished workpiece is precisely positioned in any of up to 9 axis^[1] around the 3 dimensions relative to a toolpath and, either or both can move relative to each other.

The same concept also extends to noncutting tools such as forming or burnishing tools, photoplotting, additive methods such as 3D printing, and measuring instruments.

Specific codes

G-codes, also called preparatory codes, are any word in a CNC program that begins with the letter G. Generally it is a code telling the machine tool what type of action to perform, such as:

- Rapid movement (transport the tool as quickly as possible in between cuts)

- Controlled feed in a straight line or arc
- Series of controlled feed movements that would result in a hole being bored, a workpiece cut (routed) to a specific dimension, or a profile (contour) shape added to the edge of a workpiece
- Set tool information such as offset
- Switch coordinate systems

There are other codes; the type codes can be thought of like registers in a computer. People¹ have pointed out over the years that the term "G-code" is imprecise. It comes from the literal sense of the term, referring to one letter address and to the specific codes that can be formed with it (for example, G00, G01, G28). But every letter of the English alphabet is used somewhere in the language. Nevertheless, "G-code" is established as the common name of the language.

G-CODE SOFTWARE

CNC Programming (Computer Numerical Control Programming) is the art of programming CNC machines to make parts. A CNC Program is a text file that contains g-code.

What is G-Code

G-Code is the language used to control CNC machines. It's one type of CNC programming that CNC programmer's use, the other type being CAM programming. CAM programs will generate g-code from a CAD drawing, but the end result is still g-code. Your machine's CNC controller probably executes g-code, although there are other possibilities—Heidenhain, Mazak, Shopbot, and others have proprietary formats. Some machines with proprietary formats can also run g-code. It is the Lingua Franca (working language) of CNC.

In order to make a part on a CNC machine, you tell it how to make the part using a G-Code Program.

G-Code or Geometric Code is the generic name for a control language for CNC machines. It is a way to tell the machine to move to various points at a desired speed, control the spindle speed, turn on and off various coolants, and all sorts of other things. It is fairly standard, and is a useful tool. The standard version of G-code is known as RS-274D. Since G-codes are preparatory codes, in a CNC program they begin with the letter G and direct the machine.

Typical actions G-code directs include.

- Changing a pallet
- Rapid movement
- A series of controlled feed moves, resulting in a workpiece cut, a bored hole, or a decorative profile shape
- Controlling feed movement, in an arc or a straight line
- Setting tool information

How G-Code Works

In order to achieve these particular kinds of movement, Numerical Control uses a block as its basic unit—when printed, it resembles a line of text. Each block carries one or more words (of sorts) each consisting of a letter—detailing the function to be performed— followed by a number that assigns value to the function. Currently, a block of input is limited to a maximum of 256 characters. Below are some common individual codes, that when combined, guide a machine's movement.

- G00: Rapid positioning
- G01: Linear interpolation
The machine will move in a straight line, performing the appropriate machining (milling, cutting, etc).
- G02: Circular/Helical Interpolation

The machine will move clockwise in a circular or helical pattern, performing the appropriate machining process

- G03: Circular/Helical Interpolation

This code is the same as G02, but enables counter clockwise movement.

- G17: X-Y plane selection
- G18: X-Z plane selection
- G19: Y-Z plane selection

These codes maneuver the machine onto different planes for coordinated motion.

- G20: Programming in inches
- G21: Programming in mm

Changes in programming units occur short-term with these particular codes.

The above codes are the same for both milling and turning, but other units may vary. In terms of software specifications, most g-code files can be created using CAM, but certain CNC machines rely on “conversational” programming, which either hides or bypasses the use of g-code completely.

What are the different methods of CNC Programming

CNC Machines are programmed using one of three methods:

- CAM Software
- Conversational Programming
- G-Code Programming

Often, it’s advantageous to use multiple methods together. For example, you might create an initial CNC program using CAM Software and then edit the g-code from the CAM Software using G-Code Programming to make the program manufacture a part faster.

Every CNC machinist should know g-code. If you're interested in CNC and machining, you should too. We recently did a survey to assess the g-code skills of our readership.

We were impressed at how many readers can write g-code programs from scratch. In fact the overwhelming majority read, write, or tweak programs on a regular basis. If you're not yet able to do that, you need to learn if you want your skills to be on par with others.

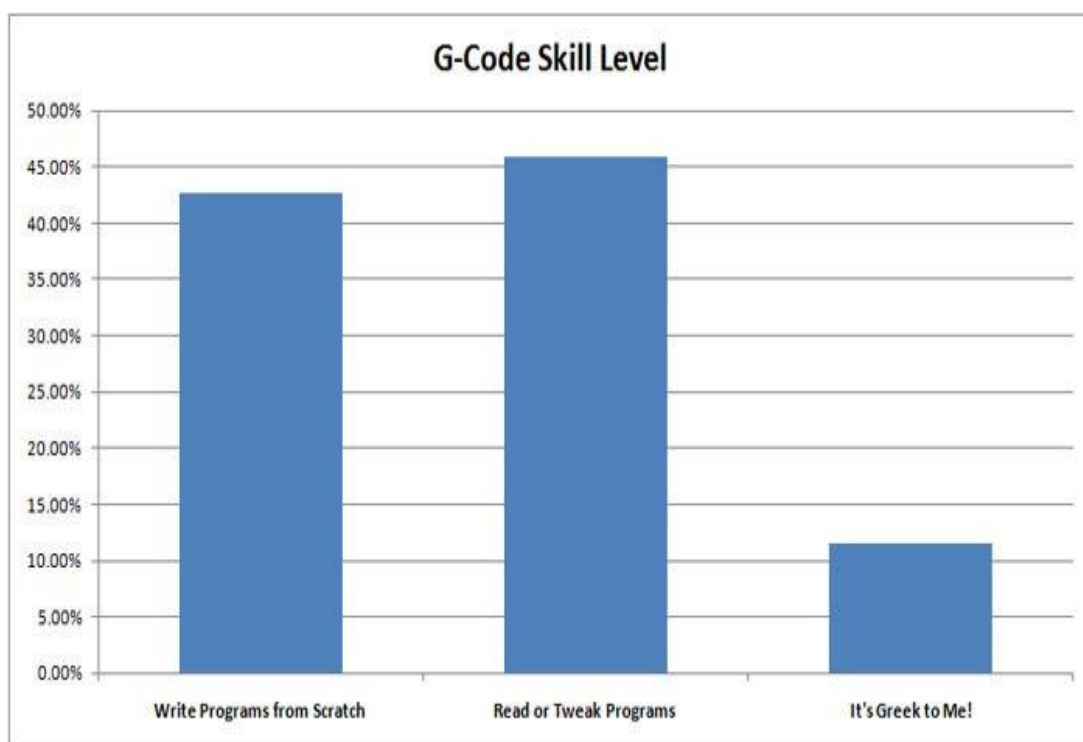


Fig. No. : 4.9 G-Code skill level

Programming is a fundamental skill for all **types of CNC machining**, even as automation and new technology seem to be replacing programming tasks. Every machinist still needs to understand how their programs and tools work. Whether you're new to CNC programming and its most common language, g-code, or you've been writing code by scratch for years, CNC codes can still feel like a foreign language. And to make things worse, every machine speaks a different dialect you have to understand. Do you understand what

they're saying? Here are the **g-code basics** you need to know to efficiently understand and write programs that produce high quality products.

G-code stands for "geometric code," and follows some variation of the alpha numeric pattern:

N## G## X## Y## Z## F## S## T## M##

N: Line number

S: Spindle speed

G: Motion

T: Tool selection

X: Horizontal position

M: Miscellaneous functions

Y: Vertical position

I and J: Incremental center of an arc

Z: Depth

R: Radius of an arc

F: Feed rate

Alpha numeric codes are used for programming as they are a simple way to:

1. Define motion and function (G##)
2. Declare a position (X## Y## Z##)
3. Set a value (F## and/or S##)
4. Select an item (T##)
5. Switch something on and off (M##), such as coolant, spindles, indexing motion, axes locks, etc.

For example, **G01 X1 Y1 F20 T01 M03 S500** would generally indicate a linear feed move (G01) to the given XY position at feed rate of 20. It is using Tool 1, and the spindle speed is 500. Miscellaneous functions will vary from machine to machine, so in order to know what the m-code means, the machine's instruction manual will need to be referenced.

Machine Motion

Everything a machine can do is based on three basic types of motion:

1. Rapid move: a linear move to an XYZ position as fast as possible
2. Feed move: a linear move to an XYZ position at a defined feed rate

3. Circular move: a circular move at a defined feed rate

Every G-code tells the machine which variation of these basic motions to perform, and how to perform it.

X and Y are Cartesian coordinates for horizontal and vertical position, and Z represents the depth of the machine. These alpha numerals will follow the motion/function command (G) to declare the position of the machine.

Next, F determines the feed rate (for feed moves or circular moves), while S determines the spindle speed. T is used to select a tool. Other alpha numerals used in programming might include I, J, and R, which have to do with arc centers and radii.

Miscellaneous Codes

The line of a program might also include m-codes, which are generally codes that tell a machine how to perform an action. While not guaranteed to be the same across machines, some common, standard m-codes are:

- M00: Program stop
- M01: Optional program stop
- M02: End of program
- M03: Spindle on clockwise
- M04: Spindle on counter clockwise
- M05: Spindle stop
- M06: Tool change
- M08: Flood coolant on
- M09: Flood coolant off
- M30: End of program/return to start
- M41: Spindle low gear range
- M42: Spindle high gear range

Modality

Just like a light will stay on until it's turned off, g-code functions (on controllers that support modality) will remain active until they are deactivated by another code. In other words, only one function can be active at any given time. To deactivate a function, just select a new function.

```
G00 X1 Y1
      X2 Y2
G01 X3
      Y3
      X2
      Y2
```

For example, say a code begins with a linear rapid move at X1 Y1 (G00 X1 Y1). If the next function is another linear rapid move, it is not necessary to write G00 again. All that is needed on the next line of code is the new position (say, X2 Y2) because the **modal condition** is the same. Then, to change the function to a linear feed (G01), programming G01 on the following line would deactivate the linear rapid move and activate the linear feed.

Once a condition is set, it stays active until it is turned off or another condition overrides it.

Canned Cycles

Canned cycles are a kind of modal condition that incorporate all the motions to complete a common task into one code.

For example, oftentimes G81 is code for a basic drilling function. In the case of basic drilling, the tool would have to be 1) moved to the starting point of the hole's location, 2) rapid to the clearance plane, 3) fed to the depth, and 4) rapid out. That would be four lines of code in the program that would have to be repeated for every new drill position! With the canned cycle G81, only the hole locations need to be specified after activation. **Canned cycles** like G81 significantly reduce the amount of code by incorporating multiple motions into one code.

G81 X1.5 Y1.5 R.1 Z.25 F12
X3.5 Y2.5
X2.5

The PIC microcontroller family is manufactured by Microchip Technology Inc. Currently they are one of the most popular microcontrollers, used in many commercial and industrial applications. Over 120 million devices are sold each year. The programmability and the versatile applications for which a PIC microcontroller can be used have made it a first choice in small scale applications and project work.

The PIC microcontroller architecture is based on a modified Harvard RISC (Reduced Instruction Set Computer) instruction set with dual-bus architecture, providing fast and flexible design with an easy migration path from only 6 pins to 80 pins, and from 384 bytes to 128 kilobytes of program memory. PIC microcontrollers are available with different specifications depending on:

- Memory Type
 - Flash
 - OTP (One-time-programmable)
 - ROM (Read-only-memory)
 - ROM less
- Input–Output (I/O) Pin Count
 - 4–18 pins
 - 20–28 pins
 - 32–44 pins
 - 45 and above pins
- Memory Size
 - 2–4 K
 - 8–16 K
 - 24–32 K
 - 48–64 K

All PIC microcontrollers offer the following features

- RISC instruction set with only handful of instructions to learn
- Digital I/O
- On chip timer
- Power-on-reset
- Power-saving sleep mode
- High source and sink current
- Direct indirect and relative addressing modes
- External clock interface
- RAM data memory
- EPROM or flash program memory

Some devices offer the following additional features

- Analog input channel
- Analog comparators
- Additional timer circuit
- EEPROM data memory
- External and internal interrupts
- Internal oscillator
- PWM output
- USART serial interface.

4.3.3 LANGUAGE USED

- Embedded C
- Android
- Visual studio

4.3.3.1 Embedded C

Embedded C is a set of language extensions for the C programming language by the C Standards Committee to address commonality issues that exist between C extensions for different embedded systems.

Historically, embedded C programming requires nonstandard extensions to the C language in order to support exotic features such as fixed-point arithmetic, multiple distinct memory banks, and basic I/O operations. In 2008, the C Standards Committee extended the C language to address these issues by providing a common standard for all implementations to adhere to. It includes a number of features not available in normal C, such as fixed-point arithmetic, named address spaces and basic I/O hardware addressing. Embedded C uses most of the syntax and semantics of standard C, e.g., `main()` function, variable definition, data type declaration, conditional statements (if, switch case), loops (while, for), functions, arrays and strings, structures and union, bit operations, macros, etc.

We can broadly define an embedded system as a microcontroller-based, software-driven, reliable, real-time control system, designed to perform a specific task. It can be thought of as a computer hardware system having software embedded in it. An embedded system can be either an independent system or a part of a large system.

4.3.3.2 Android

Android is a mobile operating system developed by Google. It is based on a modified version of the Linux kernel and other open source software and is designed primarily for touch screen mobile devices such as smart phones and tablets. In addition, Google has further developed Android TV for television, Android Auto for cars and Wear OS for wrist watches, each with a specialized

user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronics.

Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The operating system has since gone through multiple major releases, with the current version being 9 "Pie", released in August 2018. The core Android source code is known as Android Open Source Project (AOSP), and is primarily licensed under the Apache License.

Android is also associated with a suite of proprietary software developed by Google, called Google Mobile Services (GMS) that very frequently comes pre-installed in devices, which usually includes the Google chrome web browser and Google search and always includes core apps for services such as Gmail, as well as the application store and digital distribution platform Google play, and associated development platform. These apps are licensed by manufacturers of Android devices certified under standards imposed by Google, but AOSP has been used as the basis of competing Android ecosystems, such as Amazon.com's Fire OS, which use their own equivalents to GMS.

Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013. As of May 2017, it has over two billion monthly active users, the largest installed base of any operating system, and as of December 2018, the Google Play store features over 2.6 million apps.

Android - Environment Setup

You will be glad to know that you can start your Android application development on either of the following operating systems –

- Microsoft Windows XP or later version.
- Mac OS X 10.5.8 or later version with Intel chip.
- Linux including GNU C Library 2.7 or later.

Second point is that all the required tools to develop Android applications are freely available and can be downloaded from the Web. Following is the list of software's you will need before you start your Android application programming.

- Java JDK5 or later version
- Android Studio

Here last two components are optional and if you are working on Windows machine then these components make your life easy while doing Java based application development. So let us have a look how to proceed to set required environment.

Set-up Java Development Kit (JDK)

You can download the latest version of Java JDK from Oracle's Java site – Java SE Downloads. You will find instructions for installing JDK in downloaded files, follow the given instructions to install and configure the setup. Finally set PATH and JAVA_HOME environment variables to refer to the directory that contains **java** and **javac**, typically java_install_dir/bin and java_install_dir respectively.

If you are running Windows and installed the JDK in C:\jdk1.8.0_102, you would have to put the following line in your C:\autoexec.bat file.

```
set PATH=C:\jdk1.8.0_102\bin;%PATH%  
set JAVA_HOME=C:\jdk1.8.0_102
```

Alternatively, you could also right-click on My Computer, select Properties, then Advanced, then Environment Variables. Then, you would update the PATH value and press the OK button.

On Linux, if the SDK is installed in /usr/local/jdk1.8.0_102 and you use the C shell, you would put the following code into your **.cshrc** file.

```
setenv PATH /usr/local/jdk1.8.0_102/bin:$PATH
setenv JAVA_HOME /usr/local/jdk1.8.0_102
```

Alternatively, if you use Android studio, then it will know automatically where you have installed your Java.

Android IDEs

There are so many sophisticated Technologies are available to develop android applications, the familiar technologies, which are predominantly using tools as follows

Android Libraries

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access.

A summary of some key core Android libraries available to the Android developer is as follows –

- **android.app** – Provides access to the application model and is the cornerstone of all Android applications.
- **android.content** – Facilitates content access, publishing and messaging between applications and application components.
- **android.database** – Used to access data published by content providers and includes SQLite database management classes.
- **android.opengl** – A Java interface to the OpenGL ES 3D graphics rendering API.
- **android.os** – Provides applications with access to standard operating system services including messages, system services and inter-process communication.

- **android.text** – Used to render and manipulate text on a device display.
- **android.view** – The fundamental building blocks of application user interfaces.
- **android.widget** – A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
- **android.webkit** – A set of classes intended to allow web-browsing capabilities to be built into applications.

Having covered the Java-based core libraries in the Android runtime, it is now time to turn our attention to the C/C++ based libraries contained in this layer of the Android software stack.

Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

The Android framework includes the following key services –

- **Activity Manager** – Controls all aspects of the application lifecycle and activity stack.
- **Content Providers** – Allows applications to publish and share data with other applications.
- **Resource Manager** – Provides access to non-code embedded resources such as strings, color settings and user interface layouts.
- **Notifications Manager** – Allows applications to display alerts and notifications to the user.
- **View System** – An extensible set of views used to create application user interfaces.

Android - Application Components

Application components are the essential building blocks of an Android application. There are following four main components that can be used within an Android application –

S. No.	Components & Description
1	Activities: They dictate the UI and handle the user interaction to the smart phone screen.
2	Services: They handle background processing associated with an application.
3	Broadcast Receivers: They handle communication between Android OS and applications.
4	Content Providers: They handle data and database management issues.

Table No.: 4.3 Components & Description of Application Components

An activity represents a single screen with a user interface, in-short Activity performs actions on the screen. For example, an email application

might have one activity that shows a list of new emails, another activity to compose an email, and another activity for reading emails. If an application has more than one activity, then one of them should be marked as the activity that is presented when the application is launched.

An activity is implemented as a subclass of **Activity** class as follows –

```
public class MainActivity extends Activity {  
  
}
```

Services

A service is a component that runs in the background to perform long-running operations. For example, a service might play music in the background while the user is in a different application, or it might fetch data over the network without blocking user interaction with an activity.

A service is implemented as a subclass of **Service** class as follows –

```
public class MyService extends Service {  
  
}
```

Broadcast Receivers

Broadcast Receivers simply respond to broadcast messages from other applications or from the system. For example, applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is broadcast receiver who will intercept this communication and will initiate appropriate action.

A broadcast receiver is implemented as a subclass of **BroadcastReceiver** class and each message is broadcaster as an **Intent** object.


```
public class MyReceiver extends BroadcastReceiver {  
  
    public void onReceive(context,intent){}  
  
}
```

Content Providers

A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the ContentResolver class. The data may be stored in the file system, the database or somewhere else entirely.

A content provider is implemented as a subclass of **ContentProvider** class and must implement a standard set of APIs that enable other applications to perform transactions.

```
public class MyContentProvider extends ContentProvider {  
  
    public void onCreate(){}  
  
}
```

We will go through these tags in detail while covering application components in individual chapters.

Additional Components

There are additional components which will be used in the construction of above-mentioned entities, their logic, and wiring between them.

These components are –

S. No.	Components & Description
1	Fragments: Represents a portion of user interface in an Activity.
2	Views: UI elements that are drawn on-screen including buttons, lists forms etc.
3	Layouts: View hierarchies that control screen format and appearance of the views.
4	Intents: Messages wiring components together.
5	Resources: External elements, such as strings, constants and drawable pictures.
6	Manifest: Configuration file for the application.

Table No.: 4.4 Components & Description of Additional Components

4.3.3.3 Visual Studio

Visual Studio Code is a source code editor developed by Microsoft for windows, Linux and MacOS. It includes support for debugging, Embedded Git control, Syntax Highlighting, intelligent code completion, snippets, and code refactoring.

It is also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences. The source code is free and open source and released under the permissive MIT License. The compiled binaries are freeware and free for private or commercial use.

Visual Studio Code is based on Electron, a framework which is used to deploy Node.js applications for the desktop running on the Blink layout engine. Although it uses the Electron framework, the software does not use Atom and instead employs the same editor component (codenamed "Monaco") used

in Azure Dev Ops (formerly called Visual Studio Online and Visual Studio Team Services).

In the Stack Overflow 2018 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 34.9% of 75,398 respondents claiming to use it.

4.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

4.4.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

4.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

4.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it.

4.5 SYSTEM ANALYSIS AND DESIGNING

4.5.1 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy.

Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

4.5.2 OBJECTIVES

- Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system
- It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities
- When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

4.5.3 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output

design it is determined how the information is to be displayed for immediate need and also the hard copy output. It is the most important and direct source information to the user.

Efficient and intelligent output design improves the system's relationship to help user decision-making.

- Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily
- Select methods for presenting information
- Create document, report, or other formats that contain information produced by the system
- The output form of an information system should accomplish one or more of the following objectives.
- Convey information about past activities, current status or projections of the Future
- Signal important events, opportunities, problems, or warnings
- Trigger an action
- Confirm an action

4.6 SPEECH RECOGNITION

Speech recognition (also known as voice recognition) is the process of converting spoken words into computer text. Speech recognition is the process of converting an acoustic signal, captured by microphone or a telephone, to a set of words.

There two important part of in Speech Recognition:

- Recognize the series of sound and

- Identify the word from the sound. This recognition technique depends also on many parameters - Speaking Mode, Speaking Style, Speaker Enrolment, Size of the Vocabulary, Language Model, Perplexity, Transducer etc. The user speaks into a microphone and the computer creates a text file of the words they have spoken. Speech recognition (SR) is the inter-disciplinary sub-field of computational linguistics that develops methodologies and technologies that enables the recognition and translation of spoken language into text by computers. It is also known as "automatic speech recognition" (ASR), "computer speech recognition", or just "speech to text" (STT). It incorporates knowledge and research in the linguistics, computer science, and electrical engineering fields. The process of a machine's listening to speech and identifying the words is called Speech Recognition System.
- In our system, microphone acts as a input source for speech signal. Microphone performs the function of converting obtained analog signal into suitable digital form for further signal processing. The obtained is signal is compared with the already stored signal or data.
For speech recognition, following techniques are used:
 - Mel's Frequency Cepstral Coefficient
 - Dynamic Time Wrapping

Mel's Frequency Cepstral Coefficient

Mel-Frequency Cepstral Coefficients (MFCC) is the most commonly used feature extraction method in automatic speech recognition. To extract a feature vector containing all information about the linguistic message, MFCC mimics some parts of the human speech production and speech perception. MFCC mimics the logarithmic perception of loudness and pitch of human auditory system and tries to eliminate speaker dependent characteristics by

excluding the fundamental frequency and their harmonics. Most speech signal recognition systems use the so-called Mel's frequency cepstral coefficients (MFCC) and its first (and sometimes second) derivative in time to better reflect dynamic changes.

Dynamic Time Wrapping

Dynamic time wrapping is an algorithm for measuring similarity between two sequences which may vary in time or speed. DTW is used in automatic speech recognition to cope up with different speaking speeds. DTW is one of the algorithms for measuring similarity between two temporal sequences, which may vary in speed. DTW algorithm compares the parameters of an unknown word with the parameters of one reference template. This algorithm is used to differentiate the speech signals of different users depending upon the tone, speed, frequency, etc. of the speaker.

The user pronounces the character what he wants to write. The speech signal received through the microphone is given to the computer where the data is recorded to create the database of the speech signal. The recorded signal is cropped to eliminate the noise signal present. Artificial neural network algorithm is used to provide the training to system for sound classification purpose. The ANN compares the received speech and database and provides the respective signal. Mel's frequency cepstral coefficient i.e. MFCC algorithm is used for feature extraction of signal.

After receiving speech signal, it is check whether the received is valid or not and if found valid then it is compared with the already stored speech database. If the signal matches with the database then the respective signal is converted to text and applied to the Arduino board. Based on the signal received, Arduino activates the arm and triggers the servo motor for drawing the character. Servo motors are used drawing character.

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENT

5.1 CONCLUSION

The proposed work is designed using Arduino microcontroller to draw the characters such as Numerical values and Alphabets. For better results and accuracy, larger database is used to store maximum number of voice samples. The system also designed with writing robotic arm by speech recognition. In this writing mode only one font style and font size are used. The developed system can be helpful for different categories of people, specifically for physically challenged persons to express their thoughts in written manner. Based on the results, the robotic arm is capable to write what user pronounces. The mechanical movements are used to draw the desired character. The main aim of the effort is very useful for the physically challenged persons while appearing for the exams.

5.2 FUTURE ENHANCEMENT

In future more font style and font size are included for the benefits of several organizations to avoid the written effort.

REFERENCES

- [1] C.B.Kare1 , Mrs.V.S.Navale2. (2015) "Speech recognition by Dynamic Time Warping", IOSR Journal of Electronics and Communication Engineering (IOSR-JECE).
- [2] Haibin Yin, Shansheng Huang, Mingchange He and Junfeng Li, (2016) "An overall structure optimization for a light weight robotic arm", 978-1-4673-8644-9/16.
- [3] M.Balaganesh, E.Logashanmugam, C.S.Aadhitya and R.Manikandan, (2010) "Robotic Arm Showing Writing Skills by Speech Recognition", 978-1-4244-9005-9/10
- [4] Ramish, Syed Baqar Hussain and Farah Kanwal, (2016) "Design of a 3 DoF Robotic Arm", The Sixth Conference on Innovative Computing Technology (INTECH 2016), 978-1-5090-2000-3/16.
- [5] Salman Yussof, Adzly Anuar and Karina Fernandez, (2005) "Algorithm for robot writing using character segmentation", proceedings of the third international conference on information technology and applications (ICITA'05) 0-7695-2316-1/05.
- [6] Seima Saki, Sabita Devi and Manoj S, (2010) "Design Of Intelligent Robotic Arm For Visually Challenged", Proceedings of the 2010 IEEE Students' Technology Symposium, 978-1-4244-5974-2/10.
- [7] Sun Xihao and Yoshikazi Miyanaga, (2013) "Dynamic time warping for speech recognition with training part to reduce the computation", 978-1-4673-6143-9/13.