PCB CNC Machine

Prepared by:

Hassam Salamah && Ja'far Yasin

Submitted to:

Dr. Sufyan Samara

CNC machine that designed for milling and drilling PCB boards

Department of Computer Engineering

College of Engineering

An-Najah national University

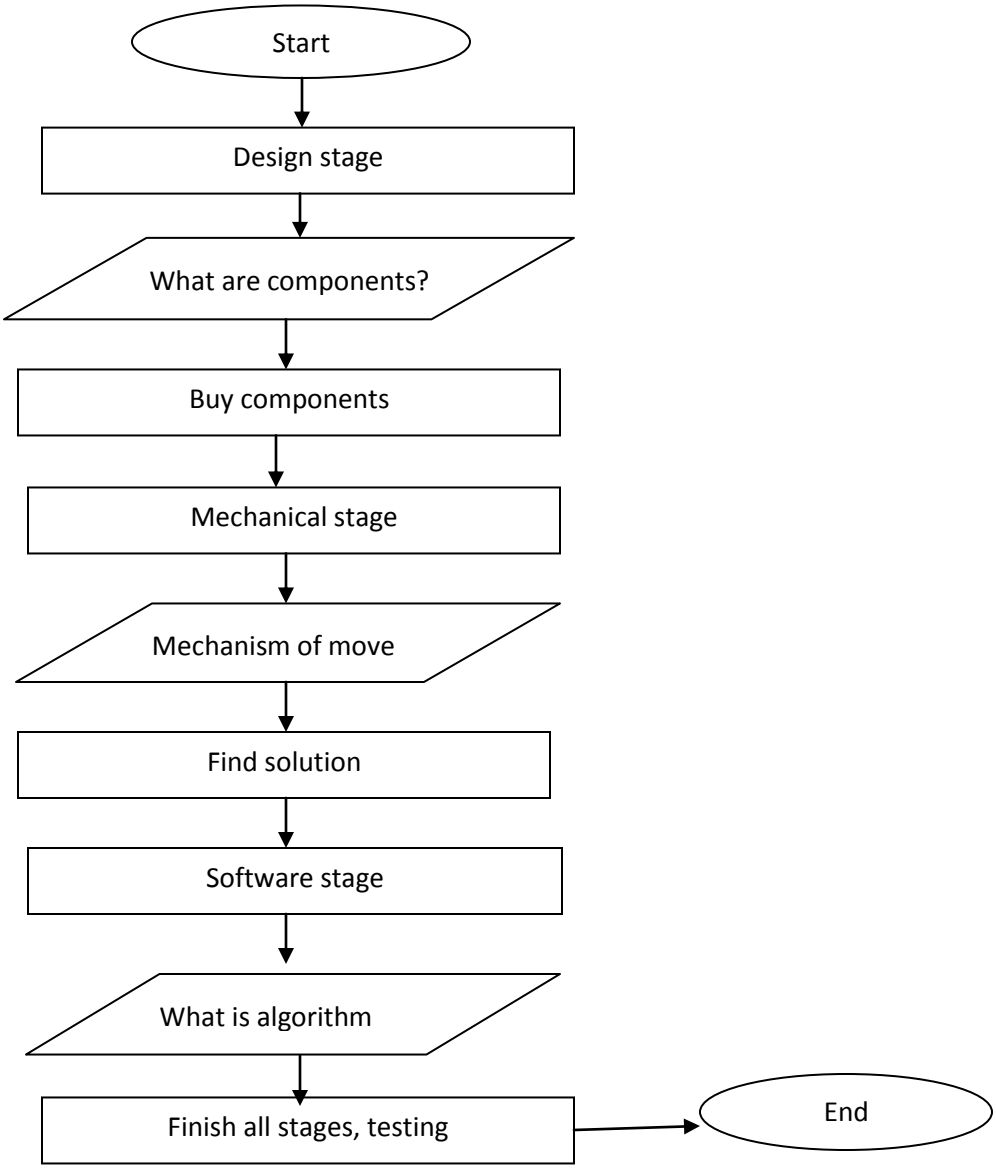
Table of Contents

Abstract

This project will include the design of and implementation of a CNC machine to make PCB (Printed Circuit Board). The first portion of the project will include a history of CNC in general case. The second part of the project will include the analysis about design our machine. Next, we will explain mechanical stage in detail with photos. Next, Will show the movement mechanism of our machine and what type of motor we use with driven circuit. At last, software stage to explain how we program our micro controller by an appropriate algorithm.

As in blow flow chart explain the progress of our project:

Figure 1 project progress



Introduction:

CNC stands for Computer Numerical Control and has been around since the early 1970's, before computer it was called NC, for Numerical Control.

CNC has touched almost every form of manufacturing process in one way or another. If you'll be working in manufacturing, it's likely that you'll be dealing with CNC on a regular basis.

Objectives:

In our project CNC it's an instrument to make PCB (Printed Circuit Board).

PCB is used to mechanically support and electrically connect electronic components using conductive pathways, tracks or signal traces etched from copper sheets laminated onto a non-conductive substrate.

History:

1955 - John Parsons and US Air Force define a need to develop a machine tool capable of machining complex and close tolerance aircraft parts with the same quality time after time (repeatability). MIT is the subcontractor and builds the machine for the project.

1959 - MIT announces Automatic Programmed Tools (APT) programming language.

1960 - Direct Numerical Control (DNC). This eliminates paper tape punch programs and allows programmers to send files directly to machine tools.

1968 - Kearney & Trecker machine tool builders market first machining center.

1970's - CNC machine tools & Distributed Numerical Control.

1980's - Graphics based CAM systems introduced. UNIX and PC based systems available.

1990's - Price drop in CNC technology.

1997 - PC- Windows/NT based "Open Modular Architecture Control (OMAC)" systems introduced to replace "firmware" controllers.

Parts of project:

1- Design part
2- Mechanical part
3- Electrical part
4- Software part

Design Part:

All projects in there design stage must be interface many things to be standard design, our machine design considered as standard; why?

- Machine uses commonly available tooling and accessories.
- Machine is easy to maintain, no specially trained service personal needed.
- Machine manufacturer cannot “hold you hostage” on replacement parts.
- Obsolescence risk is eliminated.
- Flexible use; machine can be easily modified or incorporated into a larger manufacturing system.

First sketch:

So, what about directions and axis machine will be move?
We need three axis (Y, X, Z), Y-X for milling, Z for drilling, as we see in figure (1)

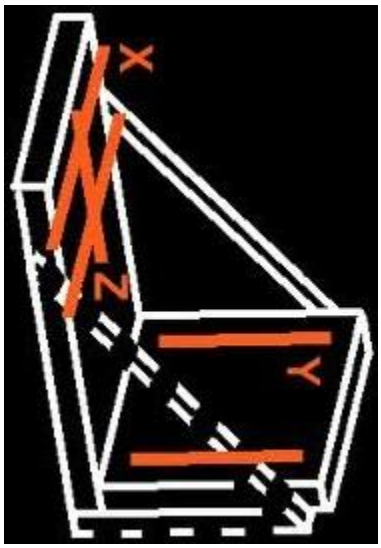


Figure 2 first sketch

There is six direction (Y, -Y, X, -X, Z, -Z).

Size:

Because our CNC machine used for making PCB (20cm *20cm) so our machine size not huge and not small, it fits with project target.

axis	X-axis	Y-axis	Z-axis
Length(cm)	55	50	20

Table1- dimensions

Material:

We need wooden model of light to achieve standard model, we choose wood material due to lightness and durability.

Figure 3 wood material

Thickness of the wood is 35 mm.

Flexibility:

CNC model can be easy to modify, if there is a necessary to add new component or remove, its easy to do that, every component fixed by bolts.

Figure 4 bolts

Components:

As standard form , every component can be found easy in our environment, can bought easy without waiting many days.

Virtual CNC:

There are many software programs that let us to build our CNC machine with our Properties.

We choose CAD soft ware, its easy to deal and much known in mechanical field.

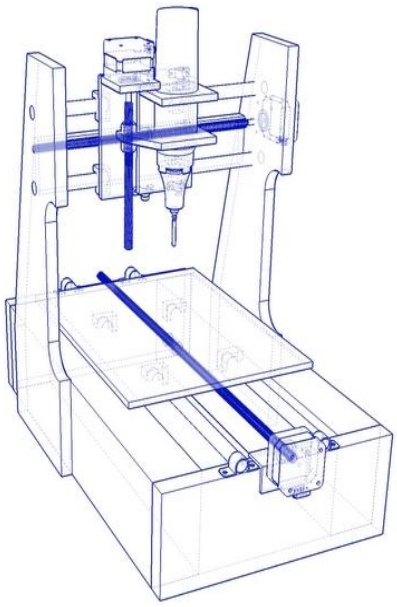


Figure 5 perspective view

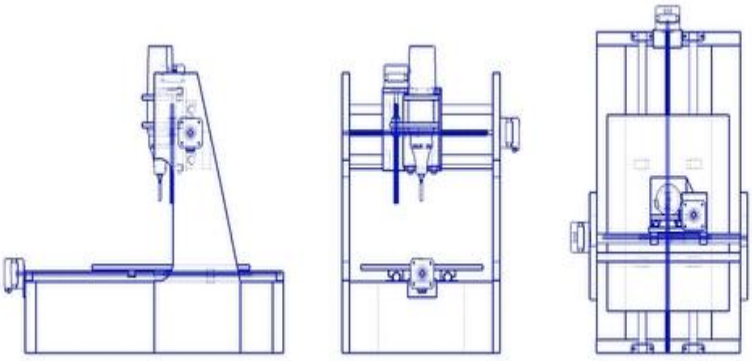


Figure 6 side, front, top views

Mechanical stage:

In this stage we worked in three parts, first one how to build each axes (wooden model), second one what about mechanism of movement for each one, last part to configure drilling part. As we imagine in design stage we have three axes integrate together as in first sketch and the wood is the best material for structure.

Y-axis: basic axis carries X-axis move from front to back.

X-axis: carries Z-axis move from left to right.

Z-axis: carries drill part move from top to bottom.

Building model:

From the beginning we should think about components and arrange table of contents.
As we say our model is wooden, and the flowing table shows every wooden Piece in model with its size.

Name of Part	Number of Pieces	Length cm	Width cm
Y Axis Back Support	2	10 1/16	2
Z Axis Rail Support	1	10 1/16	4
X Table End Feet	2	16	6 25/32
Gantry Sides	2	17 3/4	7
Gantry Bottom Suppont	1	2	4
Y Axis Rail Support Front Reinforcement	1	23 11/16	6
Gantry Bottom	1	26 11/16	7
Y Axis Rail Support	1	26 11/16	8
Motor Mounts	6	3 7/16	2 1/2
Table	2	49	24
Z Axis Back Supports	2	6 1/16	1 1/2
Router Base	1	6 1/16	8
Y Axis Linear Bearing Supports	2	6 7/8	4
Z-Axis Bearing Supports	2	8 13/32	7

Table 1 components list

Mechanism of movement:

There are some proprieties that must be found for each axis movement:

1- Smooth.
2- Easy.
3- On the same pattern.

How we will get these proprieties in a simple way?

As most mechanical system that required smooth movement they use **bearing.**

A **bearing** is any of various machine elements that constrain the relative motion between two or more parts to only the desired type of motion. This is typically to allow and promote free rotation around a fixed axis or free linear movement; it may also be to *prevent* any motion, such as by controlling the vectors of normal forces.

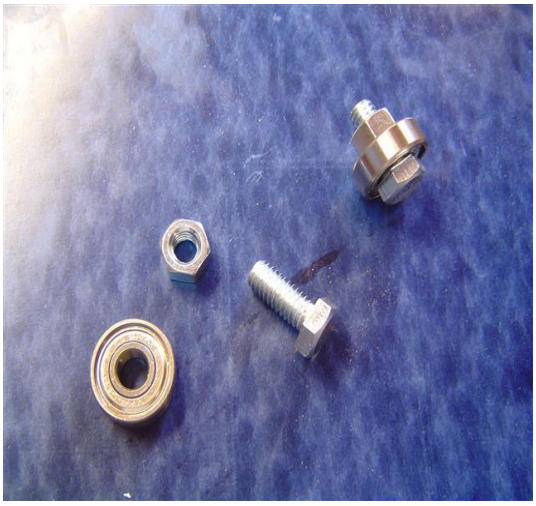


Figure 7 to fix bearing

Figure 8 Bearing

So, how we use it to achieve mechanism of movement?

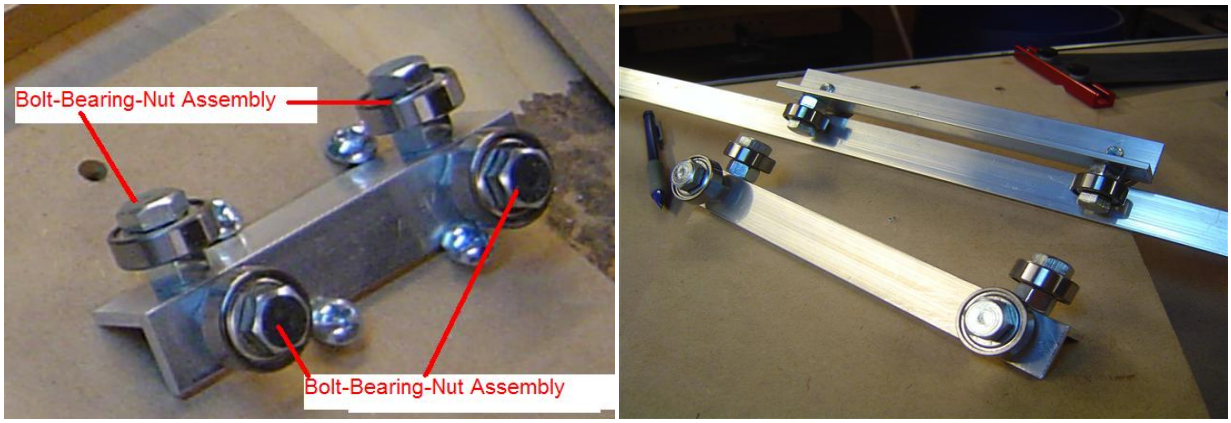


Figure 9 to make moving angle

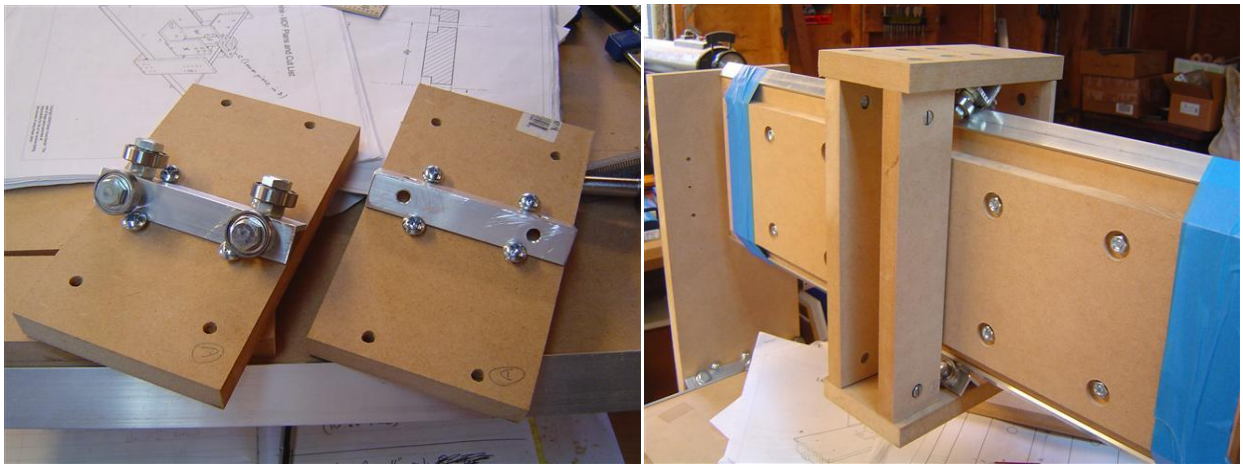


Figure 10 fix moving angle on model

Axis Lead Screw:

The key is to drill the proper size holes in the sides of the gantry. The skate bearing that I use to secure the lateral position of the screw (I mean to say, to keep the screw from moving in and out), I first drill a hole that has the same outside diameter as the bearing only half way through the wood. I use a 1.5 cm hole drilling bit. This makes a nice seat for the bearing. A nut is secured on each side of the bearing, so I drill a 1 cm hold the rest of the way through the gantry side. The same is done on the other gantry side.

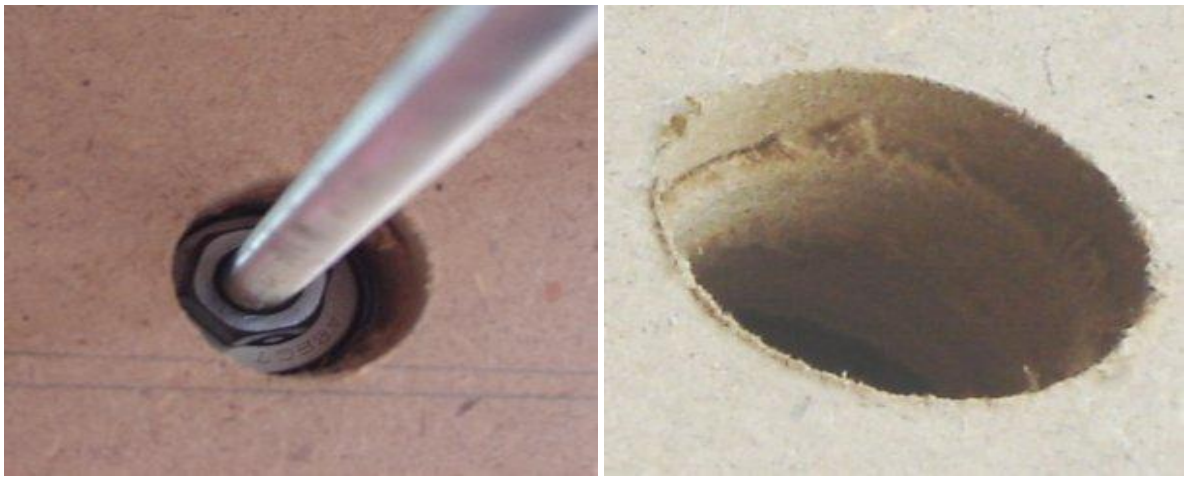


Figure 11 lead screw

This will be for each axis, and each screw links with its base.

Axis Motor Mount:

We connect the motor with the screw for each axis, as the following figure

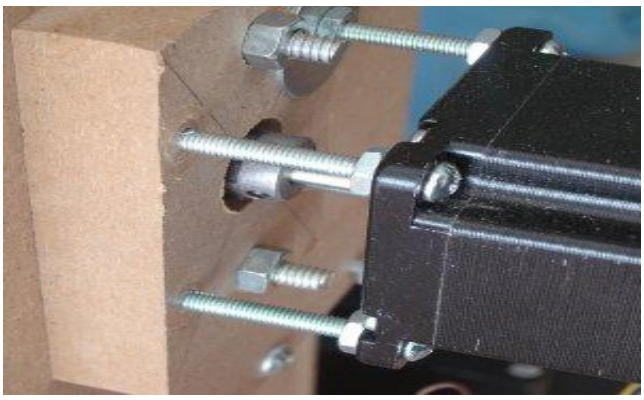


Figure 12 motor mount

Motors:

Because we have 3 axes, we need 3 motors.

We chose **stepper** motor. Because of the following reasons:

1. The rotation angle of the motor is proportional to the input pulse.
2. The motor has full torque at standstill (if the windings are energized).
3. Precise positioning and repeatability of movement since good stepper motors have an accuracy of 3 to 5% of a step and this error is non-cumulative from one step to the next.
4. Excellent response to starting/stopping/reversing.
5. Very reliable since there are no contact brushes in the motor. Therefore the life of the step motor is simply dependant on the life of the bearing.
6. The stepper motors response to digital input pulses provides open-loop control, making the motor simpler and less costly to control.
7. It is possible to achieve very low speed synchronous rotation with a load that is directly coupled to the shaft.
8. A wide range of rotational speeds can be realized as the speed is proportional to the frequency of the input pulses.

Characteristics of our stepper motor:

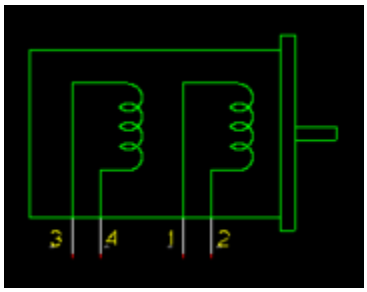
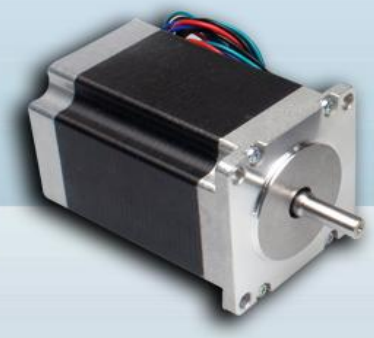


Figure 13 bipolar motor

These motors are the strongest type of stepper motor. You identify them by counting the leads - there should be four or eight. They have two coils inside, and stepping the motor round is achieved by energizing the coils and changing the direction of the current within those coils. This requires more complex electronics than a unipolar motor.

Technique parameter

Item	Specifications
Step Angle Accuracy	±5%(full step, no load)
Resistance Accuracy	±10%
Inductance Accuracy	±20%
Temperature Rise	80℃Max.(ratod current, 2 phase on)
Ambient Temperature	-10℃-+50℃
Insulation Class	B
Dielectric Strength	500VAC for one minute
Shaft Radial Play	0.06Max.(450 g-load)
Shaft Axial Play	0.08Max.(450 g-load)

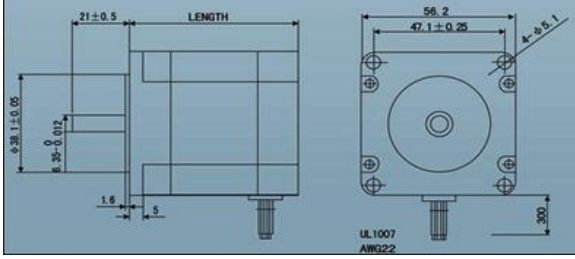


Technique Specification

Model		Step angle	Motor length	Rated current	Phase resistance	Phase inductance	Holding torque		lead wire	Rotor inertia	Weight
单出轴 Single Shaft	双出轴 Double Shaft	°	L(mm)	A	Ω	mH	Oz-in	kgf.cm	No.	g.cm²	kg
57BYGH41-601A	57BYGH41-601B	1.8	41	1.0	5.7	5.4	55	3.9	6	120	0.47
57BYGH41-602A	57BYGH41-602B	1.8	41	2.0	1.4	1.4	55	3.9	6	120	0.47
57BYGH41-401A	57BYGH41-401B	1.8	41	2.8	0.7	1.4	76	5.5	4	120	0.47
57BYGH51-601A	57BYGH51-601B	1.8	51	1.0	6.6	8.2	100	7.2	6	275	0.65
57BYGH51-602A	57BYGH51-602B	1.8	51	2.0	1.65	2.2	100	7.2	6	275	0.65
57BYGH51-401A	57BYGH51-401B	1.8	51	2.8	0.83	2.2	140	10.1	4	275	0.65
57BYGH56-601A	57BYGH56-601B	1.8	56	1.0	7.4	10	125	9.0	6	300	0.7
57BYGH56-602A	57BYGH56-602B	1.8	56	2.0	1.8	2.5	125	9.0	6	300	0.7
57BYGH56-401A	57BYGH56-401B	1.8	56	2.8	0.9	2.5	175	12.6	4	300	0.7
57BYGH76-601A	57BYGH76-601B	1.8	76	1.0	8.6	14	187	13.6	6	480	1.0
57BYGH76-602A	57BYGH76-602B	1.8	76	2.0	2.25	3.6	187	13.5	6	480	1.0
57BYGH76-401A	57BYGH76-401B	1.8	76	2.8	1.13	3.6	263	18.9	4	480	1.0

◆ We also manufacture products according to customer's requirements.

Dimensions



Wiring Diagram

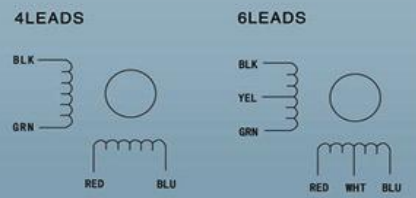


Figure 14 characteristics of motor

Drill tools:

1- Electrical drill with DC voltage input 12V, and links it on Z-axis.
This for milling cupper in board as circuit input that we want to print it, and drilling the board to make holes in board as want.

Figure 15 drilling motor

2- Carbide-milling-cutter:
We need two cutters one for milling, other for drilling the main deference is the shape of the head.

Figure 16 drilling tools

Electrical part:

Introduction

In this part we will discuss the electrical characteristics of steppers motor, the driver circuits for the stepper motor using L297, L298 H-Bridge, building the basic circuit for PIC18f4620 microcontroller, how to connect limit switches to the microcontroller, and how to connect drill motor to the microcontroller, finally we will discuss how to put everything together .

Stepper Motors:

In our project we use three bipolar stepper motors to make movement in X, Y, Z axis, the bipolar stepper motor has four, six, or eight wires come from its coils. The bipolar stepper motor has two coils, there are not electrically connected, you can identify separate coils by touching the terminal wires together, if terminal of a coil are connected the shaft becomes harder to turn

The controller of bipolar stepper motor must be able to reverse the polarity of voltage across either coil,

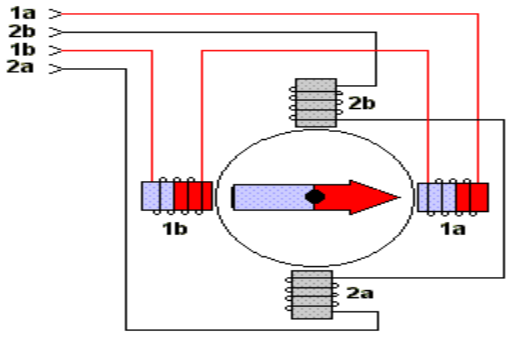


Figure 17 stepper coils

so current can flow in both directions, and it must be able to energize these coils in sequence as shown in the figure:

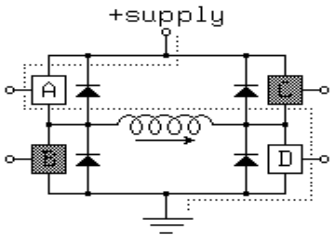


Figure 18 H-Bridge Circuit

This circuit is called H-bridge that can be able to reverse current by closing and opening the appropriate switches, in our project we use L298 chip as an H-Bridge stepper controller, we will discuss it in the following section.

L298 H-Bridge:

L298 is a Dual Full-Bridge Driver which has the following characteristics:

- Operating supply voltage up to 64V
- Total DC current up to 4 A
- Low saturation voltage
- Over temperature protection
- High noise immunity

The L298 has four output pins that connected to the stepper motor, and we put on the output Fast 2 Ampere diodes to protect chip from reverse current of the coils as shown in the figure:

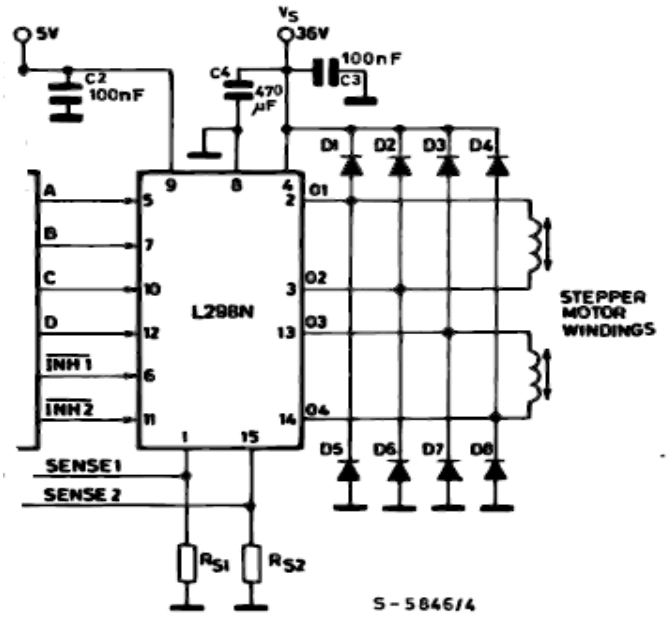


Figure 19 L298 Circuit

As shown in the figure we connect the stepper motors coils to the output of the L298, each stepper needs one L298 chip. The L298 chip also has four input pins that connected to the microcontroller, which define the pattern of the stepper motor according to the following table:

A	B	C	D
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Table 2 L298 Full step sequence

This sequence for full step movement, we can also make a half step movement by assigning a sequence to the input pins according to the following table:

A	B	C	D
1	1	0	0
0	1	1	0
0	0	1	1
1	0	0	1

Table 3 L297 half step sequence

L297 Controller:

L297 is stepper motor controller IC generates four phase drive signals for two phase bipolar and four phase unipolar stepper motors.

Characteristics of L297 IC:

- Normal/Wave drive
- Half/Full step modes
- Clockwise/ anticlockwise direction
- Programmable load current
- Reset input
- Enable input

We connect the l297 controller to the l298 H-Bridge as the following figure:

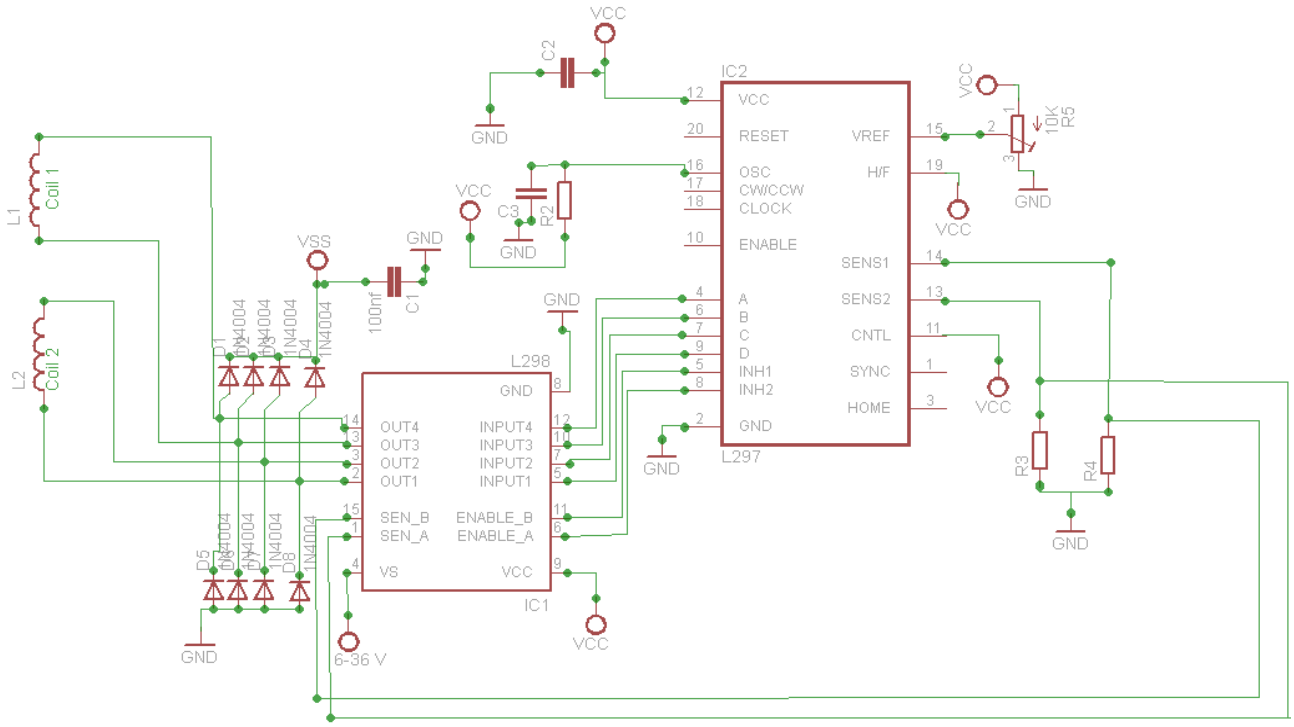


Figure 20 Stepper driver circuit

As shown in the figure the L297 provides sequence pattern to the L298 H-Bridge , we connect the sensing pins of l298 and l297 with 1 ohm resistor, this resistor works for limiting current of the motor according to “Vref” pin in l297, for example if we want to limit current of the motor to 1.5 Ampere and we have a sense resistor of 1 ohm, the we assigns 1.5 volt to the “Vref” pin

Now we can control the stepper using four pins as the following table:

Pin #	Description
18	External input clock to operate controller
10	Enable Stepper motor bridge and controller
17	Direction of stepper, 0 :CW, 1: CCW
19	Half or full step, 0: Half, 1: Full

Table 4 L297 control pins

PIC18f4620 microcontroller:

In our project we use PIC18f4620 microcontroller to control the whole tasks in the project, the first to deal with the PIC microcontroller is to build its basic circuit, after that you can write any software using PICC or other compilers and send it serially to the PIC microcontroller, the basic circuit for our controller is shown in the figure below:

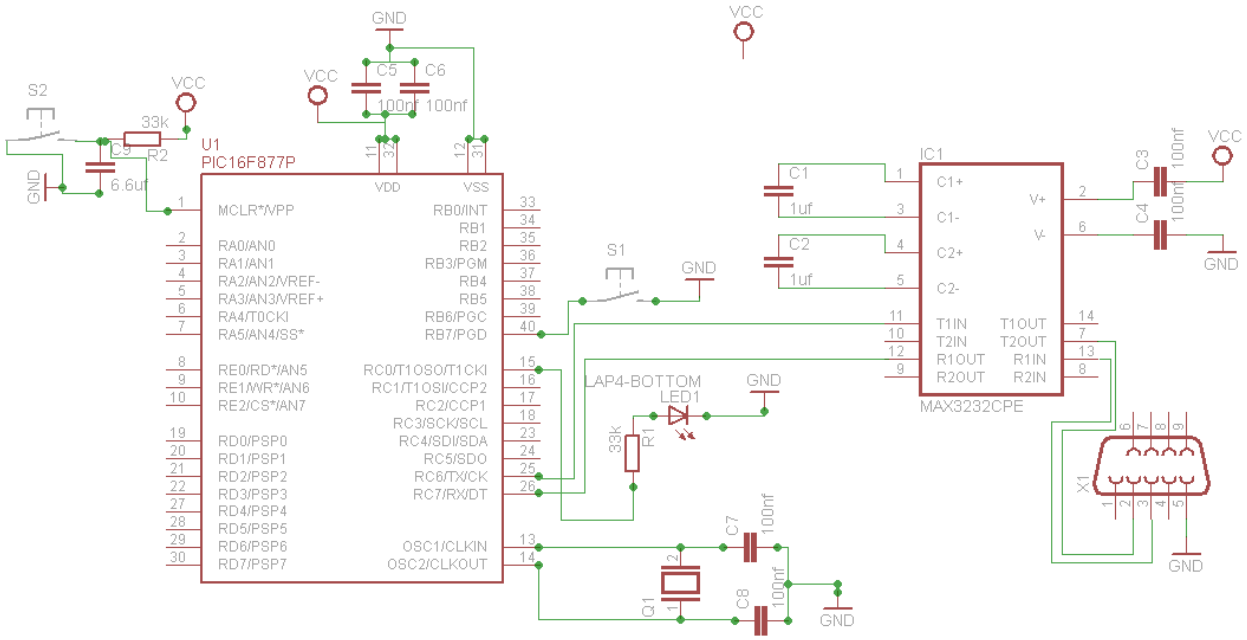


Figure 21 PIC18 basic circuit

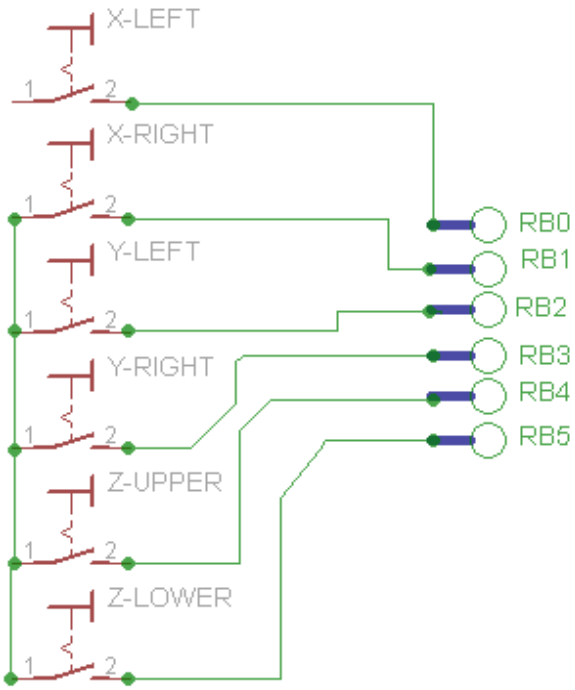
As shown in the figure, we use MAX232 IC to transfer date from/to PIC serially, and we use 4MHz oscillator

Limit Switches:

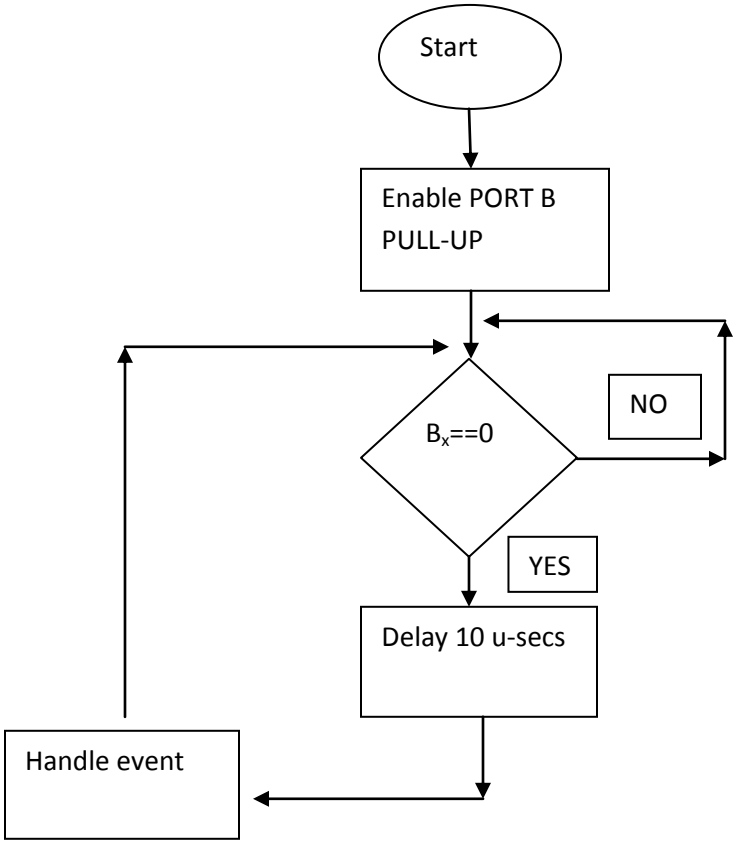
We use six limit switches in our CNC project, two for X-axis, two for Y-axis, and two for Z-axis, these switches guarantee that the machine doesn't exceed path during its movement, and to define original point for the machine during its first run.

We connect limit switches to port B in the PIC microcontroller because port B has internal pull-ups resistors, so we don't need to add any external pull-up resistor.

Figure 22 Limit Switches connection



We detect the switch pressed by writing a program using PICC compiler and handle *debounce* using software according to the following chart:

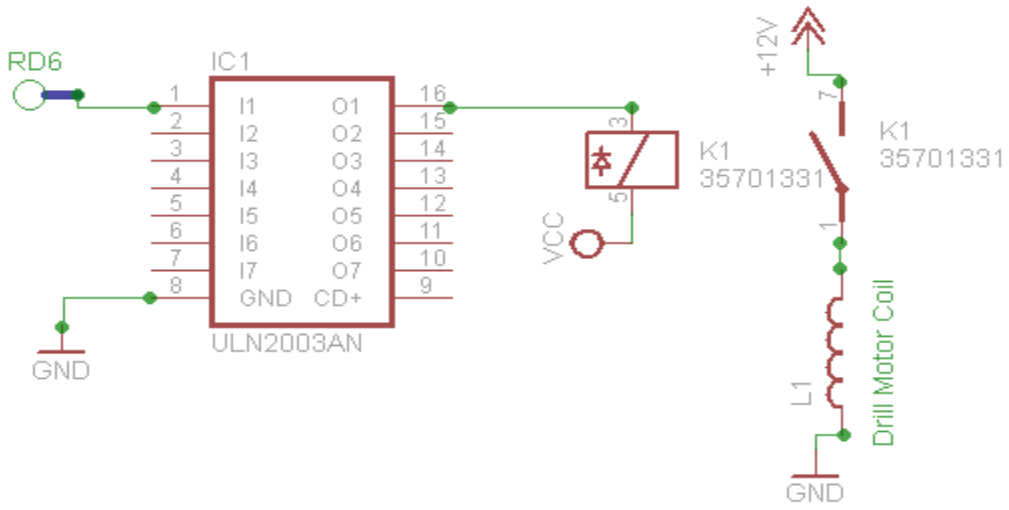


Drill Motor:

We use a 12V Dc drill motor in our CNC project to perform milling and drilling tasks, Drill must be controlled using PIC microcontroller to turn it on and off .

This figure shows how we connect drill motor in our design:

Figure 23 Drill Motor Connection



We use ULN2003 which is a Darlington array to drive relay, and protect PIC microcontroller from reverse current from the coil of the relay, and we use pin #6 in port D to control the relay.

On the other side of relay we connect 12V DC which is the voltage of the drill motor as shown in the figure above.

Now you can control the drill motor by writing a software using PICC or other compilers and send it serially to the PIC.

```
Void motorOn(){
      Output_high(PIN-D6);
}
Void motorOFF(){
      Output_low(PIN_D6)
}
```

Putting all together:

In this part we connect everything in our model to the PIC18f4620 microcontroller, so in this part we can complete the electrical design, and can move to the next part 'Software'.

We three stepper motors in our CNC machine with three drivers, we connect them to the microcontroller according to the following tables:

X-axis Stepper motor

Driver Pin #	Driver Pin name	PIC Pin #	PIC pin name
10	Enable	21	RD2
17	CW/CCW	20	RD1
18	CLK	19	RD0

Table 5 X-axis stepper motor connections

Y-axis Stepper motor

Driver Pin #	Driver Pin name	PIC Pin #	PIC pin name
10	Enable	28	RD5
17	CW/CCW	27	RD4
18	CLK	12	RD3

Table 6 Y-axis stepper motor connections

Z-axis Stepper motor

Driver Pin #	Driver Pin name	PIC Pin #	PIC pin name
10	Enable	10	RE2
17	CW/CCW	9	RE1
18	CLK	8	RE0

Table 7 Z-axis stepper motor connections

We connect limit switches to the Port B of microcontroller according to the following table:

Switch name	PIC pin #	PIC pin name
X-left	33	RB0
X-right	34	RB1
Y-left	35	RB2
Y-right	36	RB3
Z-left	37	RB4
Z-right	38	RB5

Table 8 Limit switches connections

Finally we connect input for ULN2003 that drives relay to RD6 in port D.

Software Part:

Introduction:

Our CNC model moves according to G-Code that comes from the computer over serial port. G-Code is the CNC programming language that describes the direction and the amount of movement axis must move. In this part we will discuss the G-Code commands that our machine supports and the meaning of each command, how to write program which can talk with the PIC serially, how to write program using PICC which can talk with the computer software serially, and how we parse G-Code in the microcontroller.

G-Code:

G-code is the programming language for CNC machine, it's a very simple language understood by the CNC controller and tell the machine what to do.

The table below shows the G-Code commands that our machine supports, and the meaning of each command:

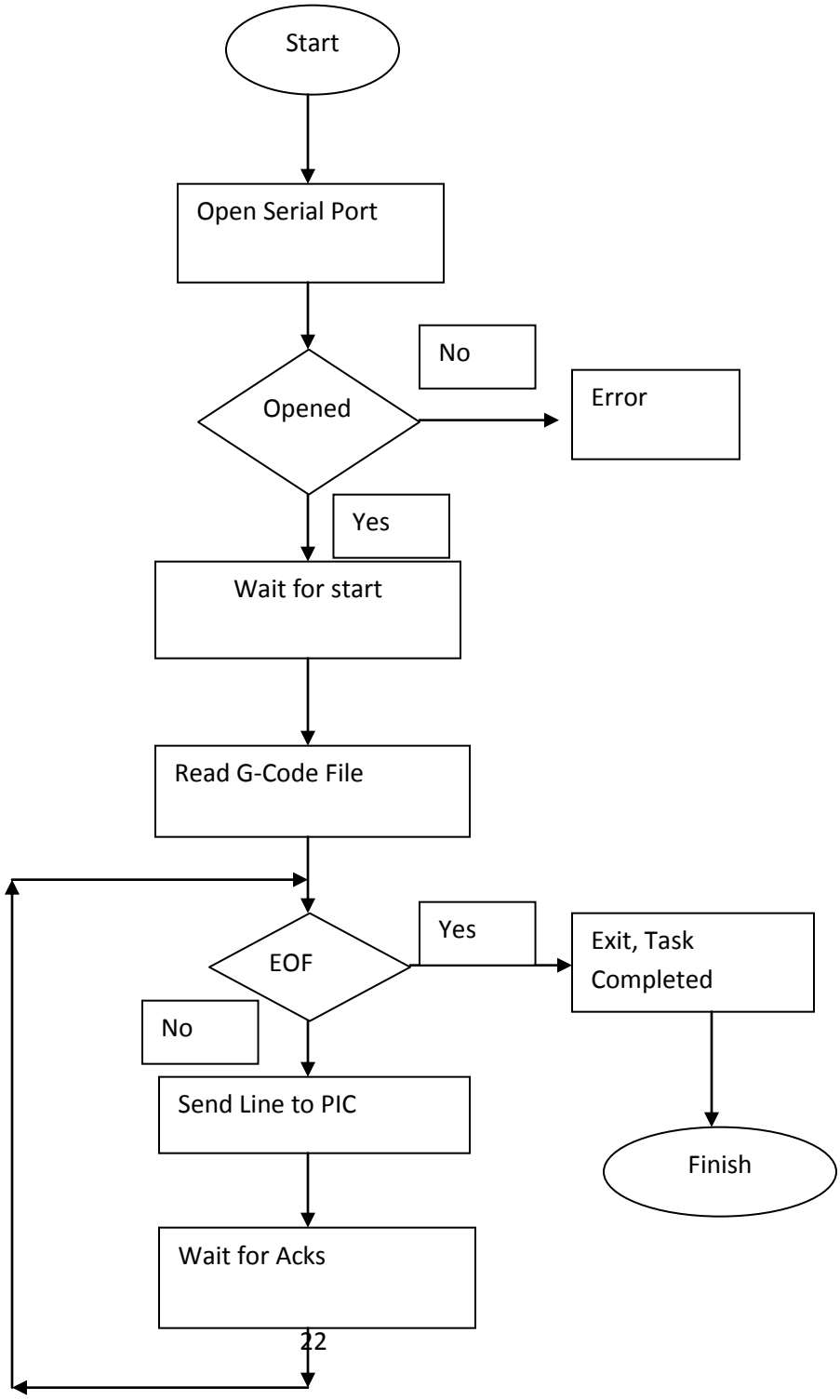
G-Code Command	Description
G00	Rapid motion
G01	Coordinate motion
G02, G03	Coordinate helical motion
G04	Dwell (delay)
G81, G82	Drilling Cycles
G90	Absolute mode
G91	Incremental mode
M3,M4	Turn on the Spindle
M5	Stop the Spindle
G21	Millimeters

Table 9 G-code table

Computer Software:

Since our project depends on the G-Code commands that come from computer serially, we wrote a java application that opens serial port, read G-Code file and sends it serially to the PIC microcontroller.

The figure below describes our software algorithm:



The first step of the program is to open serial port and check if this port is opened by another program or not, we use “RXTXCOMM” java API to use serial communication, the code below shows us how to open serial port using this API:

```
void connect(String portname) throws Exception{

//Get Serial Port by name. COM1,COM2,COM3,…

CommPortIdentifier port_id = CommPortIdentifier.getPortIdentifier(portname);

            if(port_id.isCurrentlyOwned()){
// check if serial port is used by another program

                  System.out.println("Error : Port is currently in use \n");
            }

            else{
                  int timeout = 5000;
            //open serial port

      CommPort commport = port_id.open(this.getClass().getName(), timeout);
                  if(commport instanceof SerialPort){
                        SerialPort serial_port = (SerialPort)commport;
//set parameter of opened serial port

serial_port.setSerialPortParams(9600, SerialPort.DATABITS_8,
SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);

                        System.out.println("port is opened");



            InputStream in = serial_port.getInputStream();
            this.out= serial_port.getOutputStream();

//add an event listener to the port to notify us if a data received

            serial_port.addEventListener(new SerialReader(in,out,fr));
            serial_port.notifyOnDataAvailable(true);

                  }

                  else{
System.out.println("error : only serial porty is handled by this class");

                  }
            }

}
```

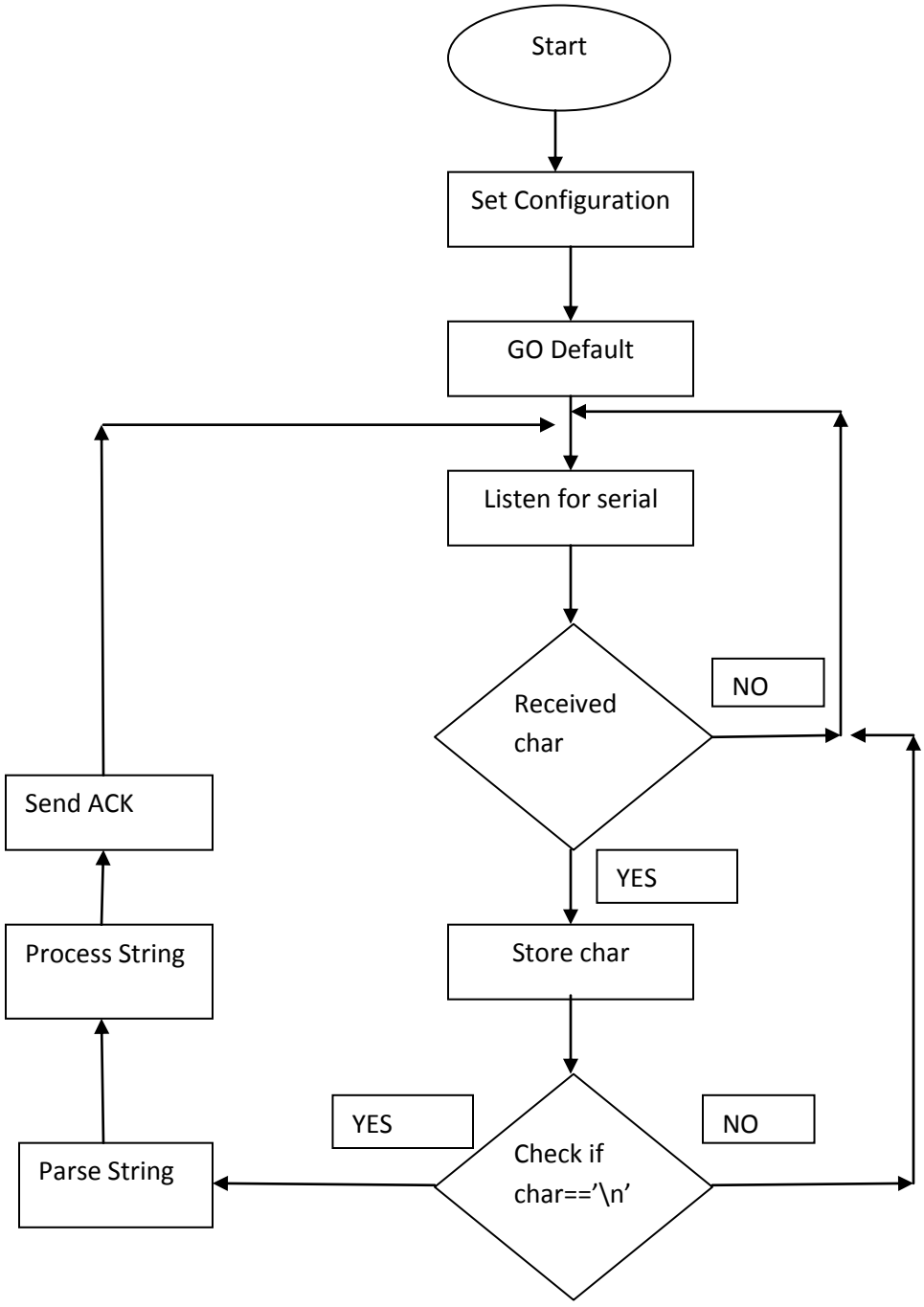
After we open serial port and adjust its parameter, we add an event listener to the port to notify us if data is received

After we open a port and add an event listener to it, we need another function executed by separate thread to write data on serial.

```
private synchronized static void sendData(OutputStream out_stream,String
out_data){

        try{


                    char []arr=out_data.toCharArray();
                    for(int i=0;i<arr.length;i++){
                          //send char over serial port
                          out_stream.write((int)arr[i]);
                          Thread.sleep(1);

                    }


                    System.out.println("data sent over serial port\n");


              }
              catch (Exception e) {
                    e.printStackTrace();
                    System.exit(-1);
              }

      }
}
```

Microcontroller software:

In our CNC project, PIC18f4620 represents the brain of the project. It listens to the serial port, receive G-Code commands, and parse them, after parsing them, it executes each command and tell the machine to perform task, the figure shown the algorithm of PIC software:



The code below shows how to listen to the serial port and waits for receiving character, it store character in buffer until the end of line ('\n') character detected.

```
void GetString(){

            i=0;
//received char detected
            if(kbhit()){
                  c=getc();
                  delay_ms(1);
//read chars until end line char detected

                  while(c!='\n'){
//store received char in a buffer
                        recv_chars[i++]=c;
                        c=getc();
                        delay_ms(1);
                  }
                  recv_chars[i]='\n';
                  recv_chars[i+1]='\0';

      }
```

We need another function in PICC to send String of characters through serial port

```
void SendString(char string[]){
            i=0;
            while(string[i]!='\n'){
//send string through serial port and wait for 1ms
                  putc(string[i]);
                  i++;
                  delay_ms(1);
            }
            putc('\n');
            delay_ms(1);
      }
```

After the whole string received we parse it to determine the type of the C-code command (G or M) and its parameter for example if received command is G00 X100 Y100, the type of the command is G and its Type number is 0, it means to move X-axis 100mm and the Y-axis 100mm. So we define a structure to represent the command in the PICC code as shown in the code below:

```
            struct Command{
                  char type;
                  int type_number;
                  struct Point point;
                  int delay;
            };
```

the last step after the command parsed, we processed the command to tell the machine what to do. The following pseudo code shows how command is processed in PICC:

```
if(command_type=='G'){
            switch(command_type_number){

            case 0:
                  move_x(#of steps);
                  move_y(#of steps);
                  move_z(#of steps);
                  break;

            case 1:
                  move_x(#of steps);
                  move_y(#of steps);
                  move_z(#of steps);
                  break;

            case 4:
                  delay_ms(command_delay);
                  break;
                  .
                  .
                  .
            }
      }

      if(command_type=='M'){
            switch(command_type_number){

            case 3:
                  turn_on_the_drill_motor();
                  break;

            case 5:
                  turn_off_the_drill_motor();
                  break;
            }
      }
```

Conclusion:

Since our project is made by hand from the scratch, we have some source of errors like accuracy of drill motor and accuracy of model measurements, and float numbers rounding, we tested our project carefully and we make several PCB samples. We can say that the accuracy of the cutting is 80% and the accuracy of milling is 85%.

References:

- www.linuxcnc.org
- www.wikipedia.org/wiki/**G**-**code**
- www.**cnc**zone.com
- http://reprap.org/wiki/Main_Page
- *CNC Machining handbook, building programming and implementation Alan Overby*
- *Build your own CNC Machine, Patrlck Hood, James Floyed*