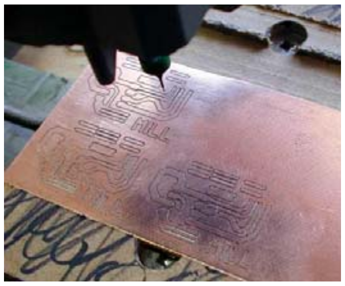
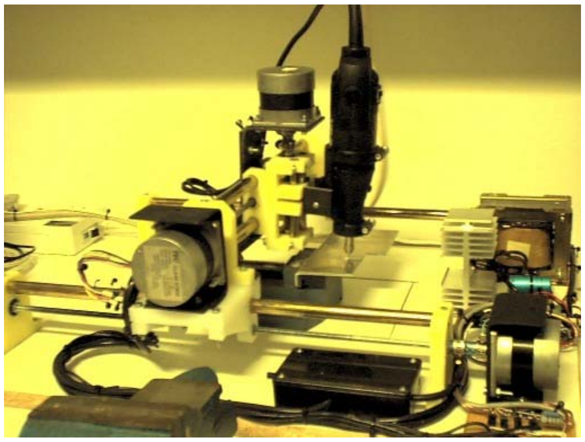
Building your own PCB Router

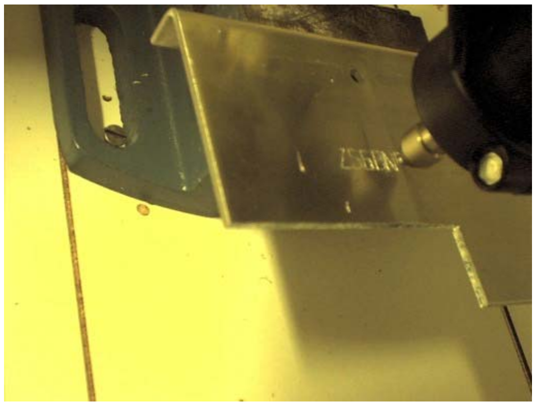
Part 2 (Stepper motor control)
Eddie Leighton ZS6BNE
www.qsl.net/zs6bne
zs6bne@webmail.co.za



In the previous issue I made mention of an Internet address where you could view pictures of my first attempts at building plotter like devices. The link was displayed as an Internet address but you would not be aware of the fact that an “Underscore” character formed part of the address in “/edleighton_za” but here is a picture of the “AGEA4” 3 axis CNC drill prototype.



Although this prototype was not as succesful as I would liked it to have been , I did learn a lot from it! The slides were made from curtain rails and the plastic bits were made from plastic chopping board! The stepper motors came from scrapped CITOH printers. This design had inherent problems making it too unstable to perform any real work but it worked! See the machine in action to the right.



This machine was run with a program I wrote in Visual basic 6 which read HPGL , the Hewlett Packard Graphics Language which standard plotters understand.

Another great problem was the Y – axis being driven from one side and not in the middle. This created an enormous amount of backlash and instablity on the carriage carrying the X – Axis motor and Z – Axis assembly together with the Dremel drill.

In the last issue of Radio ZS I mentioned we would discuss more on stepper motor electronics and control in Part 2. I also mentioned the fact that there were two types of stepper motor being Bi-Polar and Uni-Polar. The easiest to understand and most popular is the Uni-Polar type although some motors are universal in other words they can be driven in both configurations!

Although the zip files for EasyTrax and TurboCNC can be found on the Internet , they are also available on my web site at ……..

http://www.qsl.net/zs6bne/files/hobbycnc/

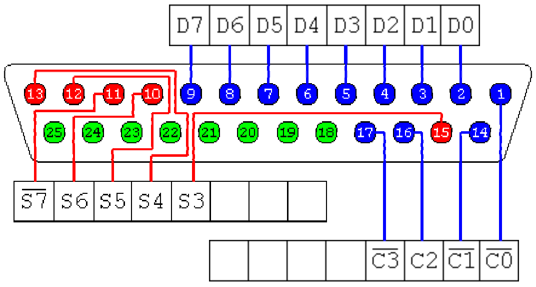
We will use TurboCNC's setup files as a guideline for the control of our stepper motors. The Internet has numerous links for stepper driver circuits but we will discuss the most basic operation here. As I mentioned previously , controllers can be very complex or simple and here we will use the simplest of motor control.

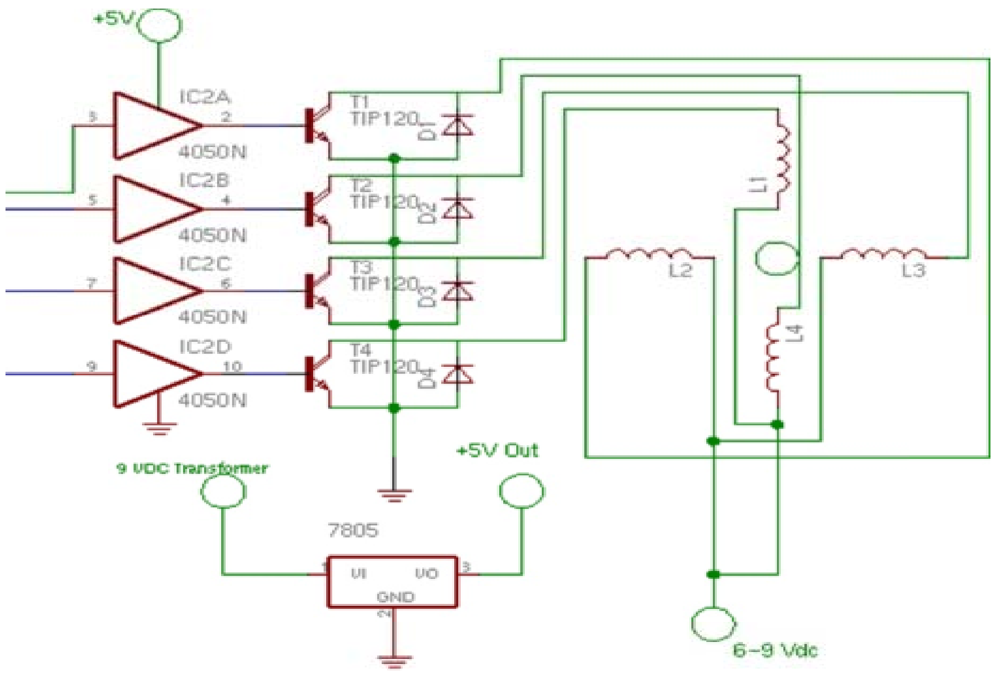
Note the "centre tapped" coils made common and taken to the positive motor power supply. A current limiting high wattage resistor of +- 5 ohms can be put in series with the positive supply to prevent the transistors from drawing too much current and being blown. Alternatively MOSFETS can be used in place of the darlington transistors.

The circuit below is the most basic configuration for driving Uni–polar stepper motors. The 4050 buffers are probably a good idea and can limit the current drawn from the printer port of the PC and also giving maximum drive to the transistors to switch them on fully providing maximum current flowing through the windings of the stepper motor coils.

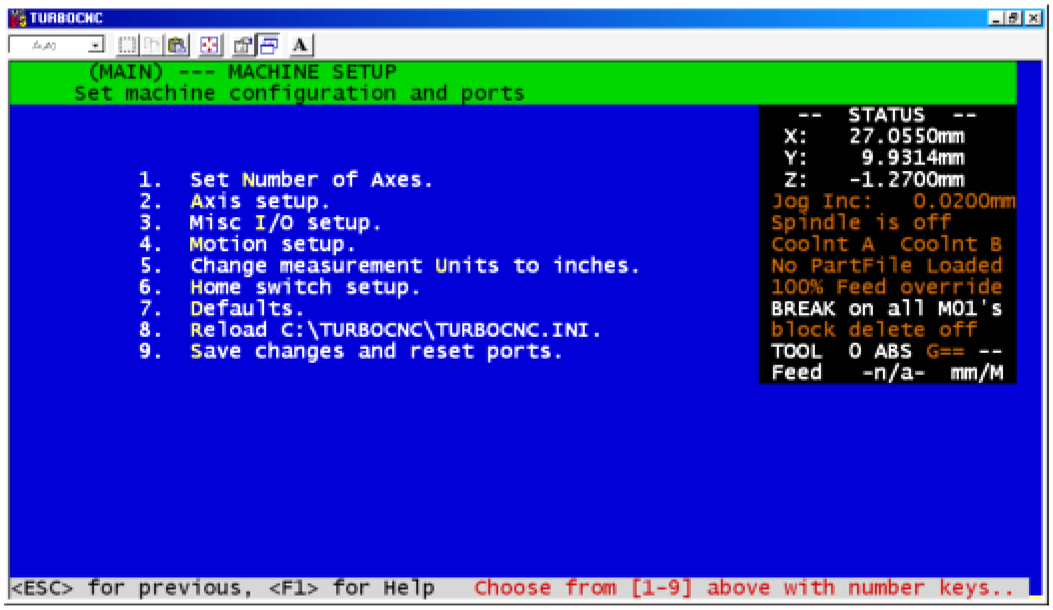
Here are the typical input / output pins of a PC's parallel printer port.

- 8 output pins accessed via the **DATA Port**
- 5 input pins (one inverted) accessed via the **STATUS Port**
- 4 output pins (three inverted) accessed via the **CONTROL Port**
- The remaining 8 pins are grounded





Setting up TurboCNC with the printer port Interface



```
(MAIN) --- MACHINE SETUP
Set machine configuration and ports

1.  Set Number of Axes.
2.  Axis setup.
3.  Misc I/O setup.
4.  Motion setup.
5.  Change measurement Units to inches.
6.  Home switch setup.
7.  Defaults.
8.  Reload C:\TURBOCNC\TURBOCNC.INI.
9.  Save changes and reset ports.

 --   STATUS   --
  X:    27.0550mm
  Y:     9.9314mm
  Z:    -1.2700mm
Jog Inc:    0.0200mm
Spindle is off
Coolnt A  Coolnt B
No PartFile Loaded
100% Feed override
BREAK on all M01's
block delete off
TOOL   0 ABS G== --
Feed    -n/a-   mm/M

<ESC> for previous, <F1> for Help   Choose from [1-9] above with number keys..
```

The screen capture above shows TurboCNC's main menu for machine setup. TurboCNC is very versatile supporting a number of controller configurations. The settings are stored in a text file by the name of "TurboCNC.INI". There are so many settings that a document like this is too short to describe everything , so I will show you what my turbocnc.ini file looks like. Every machine could be different but I will use mine as a standard.

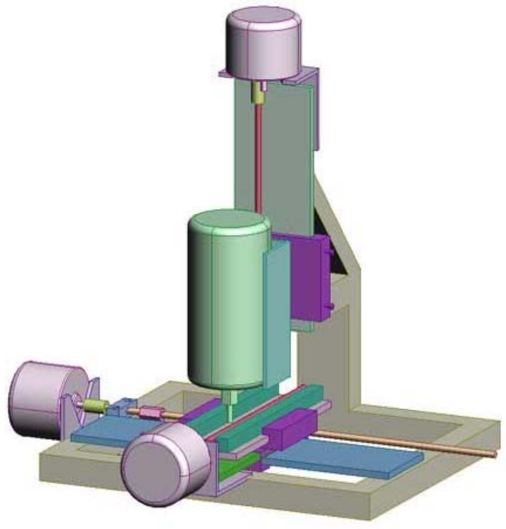
The most important option here is option 2 which sets up the way TurboCNC uses the printer port to send signals to the stepper motor driver. We will use the "Phase control" method where all the coil firing intelligence will be controlled by TurboCNC.

The other option is "Step/Direction" which requires a different stepper motor driver and the intelligence is shifted slightly to the controller itself. In other words the controller will determine which coils will be activated and only requires a step pulse and direction signal from TurboCNC in order to step the stepper motor in the required direction. This method does save on the available I/O pins on the printer port but the controller having a little more intelligence is then also a little more complicated!

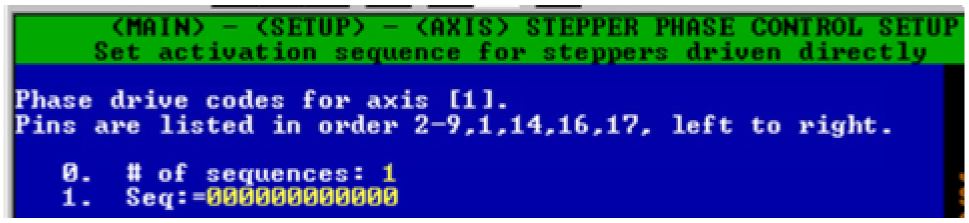
Keeping the control withinTurboCNC keeps us in control too and instead of physically building a lot of logic in hardware we do it in software! A lot less expensive too.

Another important area in the setup is option 4 , the motion setup. This determines the maximum step rate for your steppers before they start to miss steps so keep these values optimised so as not to overstep your machine. The program gradually increases the speed of the motors and it sounds impressive when the axis actually move – it sounds just like a lathe! There is a pull away step rate setting too and this must be set high enough that the machine is not too slow.

It is truly amazing to see all three axis being operated in unison all for a common purpose and under complete computer control! All we are discussing here can be applied to robotics too , but that is another subject altogether!



Phase control setup



```
     <MAIN> - <SETUP> - <AXIS> STEPPER PHASE CONTROL SETUP
      Set activation sequence for steppers driven directly

Phase drive codes for axis [1].
Pins are listed in order 2-9,1,14,16,17, left to right.

   0.   # of sequences: 1
   1.   Seq:=000000000000
```

The extract from the phase control setup screen above shows a few important points. Note the pins 2 to 17 above corrospond to the bits specified in the sequence. There will be a minimum of 4 sequences to control one motor. The four motor lines will be connected to , for example , pins 2,3,4 and 5 on LPT1 and the second motor on 6,7,8 and 9. The third motor will use pins 1,14,16 and 17 and they can be named as X,Y and Z respectively.

A 1 specifies that the line will be active , a 0 , inactive and an X means “Don’t care”. (Remember logics?) Each axis gets set up individually with it’s bit sequence in the correct position and X’s marked in the positions used by the other axis. My turbocnc.ini should explain it all and can be used on your setup too. I will make the file available under the web address mentioned at the beginning of this document so all the main editing will already be done for you.

Please let me know how your machine is going. A good test before we do any PC Board routing , is to do a little engraving. An ideal freeware program to generate G-Code for engraving goes by the name of “DeskEngrave”. This program takes any Windows type font and will convert a line of text to G-Code which can be imported into TurboCNC that will control your machine to do a little engraving!

The turbocnc.ini file

Here are a few important extracts from the file. Note that two coils are being energised by the software for every step on all three motors X , Y and Z. This gives added torque to the motors.

```
[General]
NumberOfAxes=3

[AXIS1]
Designator=X
IsStep/Dir=False
Phases=4
Phase1=0011XXXXXXXX
Phase2=0110XXXXXXXX
Phase3=1100XXXXXXXX
Phase4=1001XXXXXXXX

[AXIS2]
Designator=Y
IsStep/Dir=False
Phases=4
Phase1=XXXX0011XXXX
Phase2=XXXX0110XXXX
Phase3=XXXX1100XXXX
Phase4=XXXX1001XXXX

[AXIS3]
Designator=Z
IsStep/Dir=False
Phases=4
Phase1=XXXXXXXX1100
Phase2=XXXXXXXX0110
Phase3=XXXXXXXX0011
Phase4=XXXXXXXX1001
```

In the next issue we will discuss the **implementation** of our machines typically doing engraving at first. We will discuss how to homebrew your own cutters in the cheapest possible way!