FluidNC DIY CNC

Mitch Bradley

What Does This Have to Do with Forth ???

- In a word, Nothing (FluidNC is written in C++). Sorry
- Blame Kevin - he asked me to present, despite my insistence that I haven't been working on Forth lately
- Oh, yeah, the base class for the FluidNC settings engine is named "Word"
- But …

Laurels, Resting Upon

There is a checkin of armv64 support in the

Openfirmware Forth repo, from an engineer at Apple

Mike Perry works for the Apple CPU silicon verification team.

You might be using a processor that was tested with Forth …

Computer Numerical Control Machines

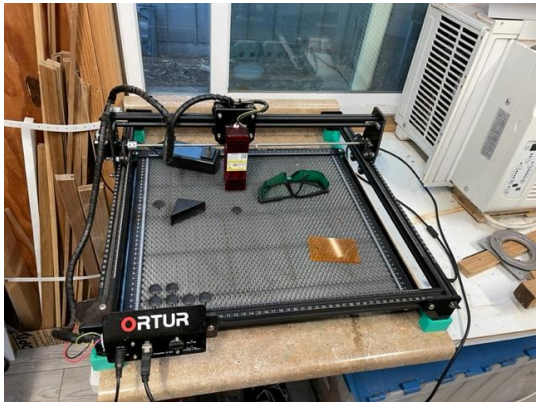
- Milling Machines
- Routers
- Lathes
- Laser engravers
- Cutters - laser, plasma, water jet, knife, glass, hot wire foam, …
- Art bots - sand tables, string art, embroidery, …
- 3D printers
- Bending brakes
- Industrial robots of all sorts

The cost of CNC machines has dropped precipitously. You can buy a laser engraver for < $200.

How I Use CNC

- My business HonuGolf makes golf putters from bamboo laminate

Many CNC Machines Involved



Personal Project

Penrose Tiling Table

Laser-cut - maple and cherry veneer

GCode Language

- Most computer controlled machines use the GCode language
- GCode is an old, dumb language that originally ran from paper tape
- Commands like
  - G0 X10 Y200 - move to that XY coordinate as fast as possible
  - G1 X100 Y50 Z10 F50 - move to that XYZ coordinate at a given “feedrate”
  - S5000 - set spindle speed
  - M7 - turn on coolant
- The language is goofy by modern standards
- There are many dialects with incompatible extensions
- There is sort of a standard - NIST RS-274

Motion Control is Hard

- GCode programs consist of a series of motions (vectors) that must occur at specific speeds
- Machines have friction, inertia, power, acceleration and precision limits that differ across axes
- Low-end machines often use stepper motors with their fixed step sizes
- The motion control software must consider all those factors and compute a multi-dimensional step timing pattern to approximate the requested motion
- Spindle speed / cutter power must be coordinated with the motion
- Cornering adds additional difficulties
- This is a “hard realtime” problem
- Safety …..

Evolution of CNC Software/Firmware/Hardware

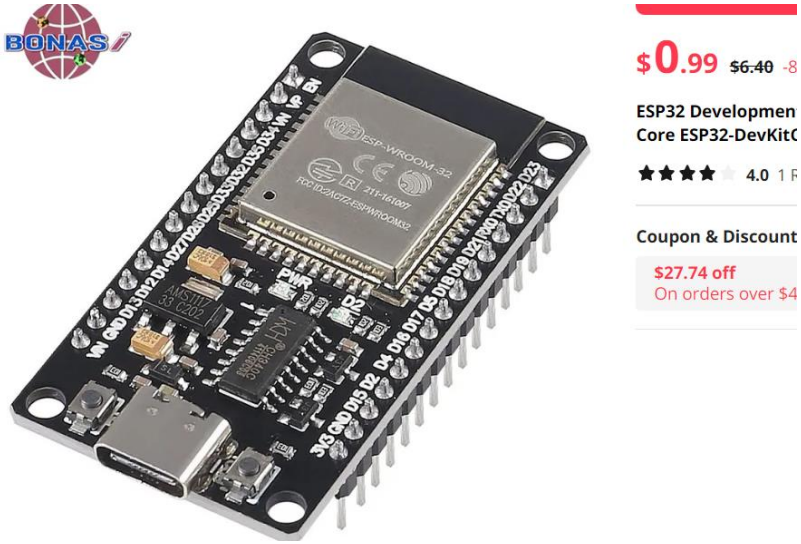
- Used to be expensive/specialized/bundled with turnkey industrial machines
- 1190s - NIST released EMC (now LinuxCNC) software
- 2001 - Art Fenerty created Mach as an offshoot of EMC. Mach ran on Windows, using a parallel port to send step and direction signals to external stepper drivers. It brought CNC to the masses
- 2009 - Simen Svale Skogsrud created GRBL - GCode parser and motion control firmware that ran on 8-bit AVR processors (Arduinos).
- Unlike previous CNC software where the UI and the motion control were on the same processor, the GRBL UI (aka the “sender”) runs elsewhere, like on a PC, talking over a serial line. There are many different UIs

GRBL Offshoots

- Classic 8-bit AVR GRBL is still in use, but there have been many offshoots -
- Various specialized forks for more powerful AVRs, ARM, etc (e.g. g2core, Smoothieware)
- Derivatives for 3D printing - e.g. Marlin
- GrblHAL, for many different MCUs
- Grbl_Esp32 (Bart Dring), for ESP32 with browser-based UI over WiFi

Why ESP32?

- 2 core, 240 MHz, 300K RAM, built-in WiFi and Bluetooth for a few bucks
- Very low system cost for a powerful, fast-booting CNC controller
- (Yes, my cforth implementation runs on ESP32 …)



My Involvement with CNC Controller Development

- Mach3 became hard to use with newer PCs that lack parallel ports
- I switched to g2core on an ARM, with a browser-based UI (CNCjs) running on a Raspberry Pi, and did some g2core and CNCjs development
- G2core wasn’t going in the direction I was interested in, and CNCjs development was stalled, so I became interested in Grbl_ESP32
- Grbl_ESP32 let me make a super-low-cost controller that I could control from an Android tablet; the whole setup including the tablet cost about $100.
- After many improvements, I became one of two lead developers.
- Eventually, after deep architectural changes, we rebranded it as FluidNC

FluidNC

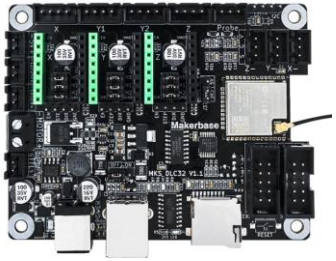
- Grbl_ESP32 had to be recompiled to configure for different boards and machines - so users have to wrangle compilers, and support is hard
- FluidNC is “one binary fits all” - configuration via a text file loaded at startup
- FluidNC supports many different situations
  - Spindles controlled by relays, PWM, VFDs (modbus), lasers, BESC
  - Multi-motor per axis for gantries, with auto-squaring
  - Different stepper drivers including “smart” Trinamic drivers, RCServos, Dynamixels
  - Different kinematics - traditional Cartesian, CoreXY, polar, wall plotter, Delta
  - IO Expanders and Display Pendants
- Web-based UI and installation tools
- Many different controller boards, from us and third parties ($15 from China), many Open Source hardware
- Support via GitHub, Wiki, and Discord

Ecosystem

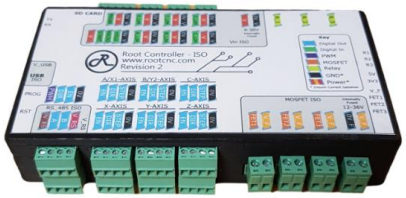

FluidNC


E4 V1.0

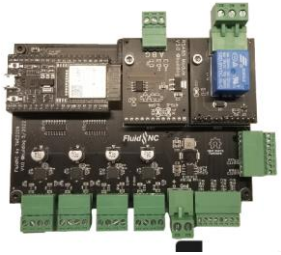

$4 off every
$19.38
Choice
FYSETC E4 Boa
3D Printer Con
Color: Built-in


Makerbase

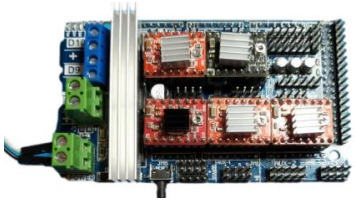

CNC xPRO V5
OSHUCNC

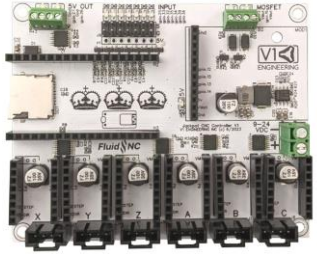

Root Controller - ISO
www.rootcnc.com
Revision 2


NIGHTHAWK
CNC

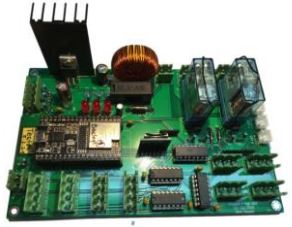

FluidNC


ESP32 Pen/Laser TMC2130 V2




FluidNC


OLED SHIELD




SourceRabbit 4-Axis CNC Motherboard

Challenges

- The software is very complex and flexible
- Fitting everything into the ESP32 RAM is getting harder. The WiFi stack and other system libraries use a lot of it
- Pin limitations
- Users are wildly different - controller boards, machines, usage, knowledge, language skills
- People can’t or won’t read the docs
- Continual requests for one-off new features
- $3 donations
- People buy 3rd party hardware and expect us to support it for free