# Table of Contents

- **Overview** 3
- **Git + MinGW Command Line** 4
- **Cygwin** 10
- **ARM Toolchain** 17
- **AVR Toolchain** 20
- **Python** 23
  - Install Python 24
  - Download and run the Installer 24
  - Python Package Management 26
- **Serial Terminal** 27
  - Putty 28
  - X-CTU 30
- **Gerber Viewers** 33
  - gerbv 33
  - Zofz 3D Gerber Viewer 33
- **MIDI** 36
  - Find your MIDI Device 36
  - Software Options 37
  - MIDI-OX 38
  - Basic MIDI message send/receive 39
  - MIDITester 41
  - Haskell OX 44
- **VirtualBox** 45
  - Check for Updates 45
  - Windows on Windows 45
- **Binary/HEX Editor** 47
- **Other Tutorials & Tips** 48
Overview

You're an electrical engineer. And you have to run Windows - because a vast number of utilities and eval tools are Windows only! Or perhaps you're running windows out of choice. No biggie, either way you can make your environment pretty flexible with some tools.

This tutorial, not-ironically, does not cover EDA tools themselves. That's because chances are you have little choice as to what chipset or layout or modelling software you are using and also because those are much more specific than the general purpose tools covered here.

Is this The Official Definitive List Of Windows Tools for EEs?

No!

It's just what I use.

So chillax and if you have something you like more, use that instead!
Git + MinGW Command Line

You'll end up using the command line a bunch for stuff, mostly because there's a lot of tools that are Mac/Linux ports.

You can launch the basic cmd command prompt from Windows either by selecting the Command Prompt program

or just typing in cmd into the Start Box
The problem is that the default cmd utility totally sucks and is not Linux/BSD compliant.

For example, you can’t even use ls

We can fix that a few ways, such as installing MinGW or MSYS manually - but the fastest and easiest is to install Git (which is really handy in and of itself) and also the MinGW32 command line tools that come with it. That will give you some basics!
Visit [https://git-scm.com/download/win](https://adafru.it/nkD) to download the latest. It's all open source, yay. In the off chance that the site is not up, here's a link to the 64 bit Windows version. You should not download this unless the git website is down!

**Git-2.8.2-64-bit.exe**

[https://adafru.it/nkE](https://adafru.it/nkE)

And run the installer to install.

The default install directory is good to keep the same
I like using a TrueType Font in all console windows so I check it (normally it is not)

I want to use the unix tools from the command prompt, that's pretty much the point of this tutorial, so be sure to check that box!

The default CRLF handling is OK, this sometimes will bite you but just be aware of it!
MinTTY is more powerful than the command prompt and while you can always use cmd its nice to have an option (we'll see that later) so select Use MinTTY

Finish the installation. Then close any open command line windows and re-run cmd

You now have proper cd and ls!
You can run the fancy MinTTY when you want from the Git installed tools folder:
Note that this does not give you build tools like make!

For that, and a compiler, you'll need to install either cygwin (comes with everything) or the AVR toolchain (has just basic utils) (https://adafruit.it/nle)

---

**Cygwin**

Git + MinGW will get you some basic command line tools but they're just the most basic tools. If you want to do some serious command line compiling and hacking, cygwin will give you as-close-as-possible-to-Unix like command capabilities

If you need things like yacc, gperf, curses and who knows what else - cygwin is for you!

A kind reader wrote in and suggested [https://babun.github.io/] but I haven't had a chance to check it out!
Visit https://cygwin.com/install.html (https://adafruit.it/nmc) to download the installer, which is also the package manager.

**Installing and Updating Cygwin Packages**

**Installing and Updating Cygwin for 32-bit versions of Windows**

Run `setup.exe` any time you want to update or install a Cygwin package for 32-bit Windows. The signature for `setup.exe` can be used to verify the validity of this binary using this public key.

**Installing and Updating Cygwin for 64-bit versions of Windows**

Run `setup.exe` any time you want to update or install a Cygwin package for 64-bit Windows. The signature for `setup.exe` can be used to verify the validity of this binary using this public key.

Unlike the other software, cygwin is basically installed from online package sources that constantly update.
The default install into C:\cygwin is suggested

Select whichever mirror you want, some might not be up but one will work!
There's hundreds of packages you can install into cygwin, chances are you need the basic defaults and maybe a few special ones.
Say you need the compiler tool bison - type bison into the search box and you'll get two possible 'categories' - debug and devel

I just need the developer tool, not the debug info (since im not debugging bison itself) So where it says skip - click that to install it. I rarely need the source code (src) so i just install the binaries (bin)

Cygwin will try to install as many sub-dependencies you need as possible
It will then install all the packages you selected.

And run the Cygwin Terminal to use cygwin's tools. Note you cannot use the plain cmd shell to run cygwin programs.
If you need to find your files or go to the Desktop, you can go to your 'windows home directory' with /cygdrive/c/Users/username

If you ever need another package, just re-run the installer. For example, halfway thru this build I realized that gperf was required
Just relaunch the cygwin terminal when done updating

ARM Toolchain

If you want to develop for ARM chipsets, first step is to install the ARM compiler into a set location. This toolchain is used by dozens of different chip companies so its always handy to have

Go to https://launchpad.net/gcc-arm-embedded/ (https://adafru.it/lKc) to download the toolchain for your operating system

You may need a particular version, for example https://launchpad.net/gcc-arm-embedded/4.8/4.8-2014-q1-update (https://adafru.it/lKc) for what you're doing, check your chipset or Makefiles for any hints
Download & run the installer

However, when it comes time to install into a folder...

I like to install into C:\ARM\4.8_2014q1 (or whatever the version you're using is) which doesn't have odd characters or spaces - sometimes makefiles choke on spaces or the (x86) in the default path.
And add the path to the environment variable

Once installed, make sure you've got the files installed in the right place
From the command line you will now have the arm-none-eabi tools available.

Note that you do not get other tools like make! You can get those by installing the AVR toolchain, though (https://adafru.it/nle)

### AVR Toolchain

Even if you aren't using AVR processors, the MHV AVR toolchain comes with a bunch of command line tools that are handy (like, say, make)

We pretty much just use this toolchain rather than any other at this point

You can download it from https://infernembedded.com/products/avr-tools/release (https://adafru.it/nlf)
On the off chance the site is down, here is a mirror (Don't use this download unless the main site is down since there may be a newer version!)

MHV_AVR_Tools_2013101.zip
https://adafru.it/nIA

Download, unzip and run the installer

The default settings are best, including the already-clicked Add MHV Tools to Path
You can see during installation that you get a mess load of command line tools installed for you. They aren't the latest versions but they work pretty well!

For example, you now have make (finally!)
and, of course, avr-gcc

Python

![Python logo]

Python is the most popular scripting language being used at the time of this writing, having surpassed perl and being more cross-platform-friendly than bash and other shell scripting languages.

Python is similar to C and perl languages and isn't too hard to pick up if you are familiar with programming. It's also often used for teaching programming. Really, it's worth getting familiar with.

I strongly recommend installing python after the Git+MinGW tutorial
Python 2 or 3?

Haha, welcome to the biggest debate in the Universe. My suggestion is, you should stick with Python 2.7 - that's what everything is compatible with. Many programs/scripts and libraries are not Py3 compatible and the changes, while often back-compatible, just aren’t ported to older things.

So, if you write new code, it’s a good idea to make em work with Python 3 but I still use Py 2.7

Install Python

Visit [https://www.python.org/downloads/](https://www.python.org/downloads/) (https://adafru.it/fa7) to download the latest version. As of this writing it’s 2.7.11

Download and run the Installer

The default location C:\Python27\ is a fine place
I do suggest Add python.exe to Path so you can just run it from the command line.
Once done, you can start up a new command line and run python

Python Package Management

You will almost certainly want to install a python package manager

Which python pkg manager should I use?

Welcome to the second-biggest debate in the Universe! My suggestion is you stick to pip as it seems to be the most commonly used

Lucky for you, it comes with the Python 2.7 installer above. You can run pip from the command line to check
So say you want to install the python packages click and gitpython - just run pip install click gitpython

and it'll be done for you!

Serial Terminal

You'll probably need to connect to a serial port now and then. I have two programs I use for connecting to Serial devices, putty and X-CTU
Putty

While not the most powerful serial port software, putty does a good job. It also does telnet and ssh so that’s handy as well

You can download putty from http://www.putty.org/ (https://adafruit.it/aUb) or http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

On the off chance the site is down, here is a mirror

putty-0.67-installer.msi
https://adafruit.it/nma

The installation defaults are pretty good
Putty is pretty simple to run, just run the Putty tool

For serial ports first click Serial in the radio buttons. Then type in the Serial name in Serial line (e.g. COM5) and finally, set the Speed to be whatever speed you like

Then click Open to open the port
I also sort of like X-CTU which has more low-level tools like the ability to toggle and monitor the flow control lines, view hex codes, see both incoming and outgoing bytes, generate and send packets, etc.

X-CTU is available from Digi here (https://adafru.it/nmb)

Once installed, you can run just the Serial Console from the Tools menu

Start off by Configuring the serial connection

Unlike Putty, you'll get a GUI interface to select the serial port, byte encoding, baud rate, and whether you want flow control
Once configured click the Open Port button
If you type into the console the sent data is shown in blue. Incoming data is shown in red.

On the left you see the ASCII values, on the right, HEX bytes. In this case I sent the text abcde to a USB serial cable with nothing connected to the end. I then connect RX and TX pins together so that sent data would come back as received data and typed in ECHO

What is really easy is setting and unsetting the flow control lines, handy if you, for example, want to test that DTR is resetting an Arduino compatible chip, or that the CTS line goes high/low based on the pin settings of a breakout board
Gerber Viewers

You're making hardware! You want to send out for PCBs, but you also want to make sure the files are right before sending - thus you need a Gerber Viewer. I use two viewers for my Gerber'in.

**gerbv**

Part of the geda project, it's both free and open-source!

[http://gerbv.geda-project.org/](https://adafruit.it/LBC)

**Zofz 3D Gerber Viewer**

This tool is not as precise for viewing Gerbers, but it is great for proofing your PCB look, especially silkscreen, mask, and coloring. You can see the PCB in 3D which is handy for visualizing the design.

It's available from [https://www.zofzpcb.com/](https://adafruit.it/nme)

Once you've downloaded and installed it, you can run the Auto Import tool.
And navigate to a folder with gerber files in it. If you use the default exporter, then Zofz will do its best to match the file types (last three letters) to determine what layer it is 

And sort them into the proper layers and 'types'
it will then render it. You can click and drag to rotate the board around.
There are various color-styles that are pre-programmed in, so you know what mask and silk will look like. Or you can make your own (like my 'adafruit blue with ENIG' look above!)

**MIDI**

If you're developing anything with MIDI you may want to send/receive MIDI messages. While MIDI is technically a Serial protocol, often times you'll be working with a 'native MIDI HID USB' device, so it doesn't show up as a COM port.

**Find your MIDI Device**

Native MIDI USB is a class device, you do not need a driver, and it should appear under your Sound Controllers devices in the dev manager.
Software Options

There's a few different ways to debug MIDI so I'll give you some options

- **MIDI-OX** - the most popular and full featured MIDI software, but is not open source.
- **MIDI Tester** - not good for testing received messages, but good for sending data to a MIDI device. Open source.
- **Haskell-OX** - doesn't display messages but is very fast and easy to tie, say, incoming MIDI messages to another MIDI device or to the Windows Synth. Free.
A handy, if slightly older, tool that I like is MIDI-OX. Note it isn't open source, and is not free for commercial use, but it does have a trial period and its pretty nice!

Download it from http://www.midiox.com/ (https://adafru.it/nkF)

Install it as normal:
Basic MIDI message send/receive

You'll need to tell MIDI OX which device to listen to for MIDI messages. Click the blue MIDI Icon in the menu bar or select the Options->MIDI Devices... menu

You'll get this window
Deselect "Automatically attach Inputs to Outputs during selection"

then Click on the MIDI Input ("Circuit Play MIDI") in the top left box, and it will automatically get added to the devices we listen to.

Now in the main windows, it will start printing out incoming messages
Another option is the Japanese-written MIDITester which is available from http://openmidiproject.osdn.jp/MIDITester_en.html

On the off chance that site is down, here's a mirror of version 4.1 Don't download this unless the main site is down!

This program doesn't have an installer, just uncompress it to run
You'll probably want to start by changing the language to English.
After restarting, you can select the MIDI device you're testing, for my test I am not linking the input to output

OK so it's a little different than MIDI-OX, you can see at the very bottom incoming messages appear at the bottom status bar. You can send outgoing messages by clicking on the piano keys.
Haskell OX

This is the simplest tool but it works fine for just connecting two MIDI devices together, or connecting MIDI In messages to the Windows built in MIDI Synth. It's written more as a demo of Haskell + MIDI and seems to be free (but not technically open source)


On the off chance the site is down, here's a mirror (Don't use this link if the site is up!)

![HaskellOx](https://adafru.it/nld)

I just tied the Circuit Playground to the wavetable synth so it would play notes on the computer when the MIDI notes were sent
OK you love Windows but sometimes you really have to use Linux or some other version of Windows (like, I have one dev kit that only works on windows XP!) That's where VirtualBox is really great.

You can download VirtualBox from https://www.virtualbox.org/wiki/Downloads (https://adafru.it/cBK)

Check for Updates

Always manually check for updates, make sure you're running the latest!

Windows on Windows

You can grab an image of Windows 7 for testing - its designed for testing IE but really you can use it for anything. It's also great for testing installation procedures on clean OS's to make sure you didn't miss any dependencies.
Check out all the images from Microsoft here [https://adafru.it/nkC]

It 'expires' after 90 days but for most people that's perfectly fine for testing!

In particular we used it to test some of the tool installs in this very tutorial! How meta...

Uncompress and double click the IE11 - Win7.ova file

After 5-10 minutes you'll have a new entry in VirtualBox
Which you can then click to run!

Binary/HEX Editor

If you're editing files like, say EDID binaries, you may want a handy hex/binary editor that can handle very large files. I like HXD - its very fast and light, and doesn't have any issue editing big files or zipping around to different offsets. It's really a hacker dream!
You can grab HXD from https://mh-nexus.de/en/hxd/ (https://adafru.it/nmf)

Its pretty straight-forward, open a file and edit it. You can edit files that are open or will be opened which is great for live-hex-editing

Other Tutorials & Tips

We also have some older tutorials you may find handy in the Adafruit Learning System

- Removing old and unused COM ports (https://adafru.it/nlC)
- Signing drivers, apps and creating installers (https://adafru.it/nlD)
- Setting a caps-sensitive filesystem (https://adafru.it/nlE)