Creating Open Source Electronic Hardware with Open Source Software

Tom Anderson
Overview

• Open Source hardware
  – Business model
  – Example circuit, concept -> production

• Make circuit boards with Open Source tools
  – Design and prototype the hardware
  – Using Open Source design software
  – Enter the schematic
  – Draw the layout

• Simulate circuits
  – Analog simulation
  – Digital simulation
Creating Open Source Hardware

- Start with an idea
- In this example the idea is:
  - “High-speed photography pictures would look good in Make magazine”
- Build crude prototype
- Get approval for Make Magazine Article
- Build a better prototype that can be made into a kit
- Write the article
- Sell kits through www.makershed.com
Design Constraints

Customer Usability, 80% of users

- Don't want to solder
- Don't know anything about electronics
- Don't know how to debug a circuit
- Want a project that lasts about an hour or two
- Want it to work perfectly the first time
Working within the limits

- We build and package the board
- Customer solders a few wires (or pays more to get an already-built solution)
Contract Manufacturers

- Very expensive, or
- Want huge volumes
- Want you to redesign the product
- Won't do a much hand labor – they want to use only the machines that they have
Commercial Open Source Hardware

- Idea -> Find sales channel -> Sell idea to sales channel
- Schematic -> Firmware -> Prototype -> PCB
- Have bare boards built -> Buy parts
- Lead-form parts -> Load -> Solder
- Assemble product -> Test
- Package -> Ship product -> Invoice
- Promote -> Sell -> Support
Does it scale to full-time self-employment?
Don't quit your day job
The kit at the Make Magazine store
What's In the Kit
The Printed Circuit Board

In-Circuit Programming Connector
(Compatible with Microchip ICD 2)

“Spare” Pin 2 PIC16F616
Jumpers in the kit

Flash Controller Version 4 -- Jumper Location and Function

Gain Jumper IN: Normal Gain
Gain Jumper OUT: High Gain

Logic Lvl Jumper IN: 5V Output
Logic Lvl Jumper OUT: Snap Shot Output

Blank Jumper IN: 3 sec Blanking
Blank Jumper OUT: 0 sec Blanking

Delay Jumper IN: Delay Range 0 - 11 msec
Delay Jumper OUT: Delay Range 0 - 0.5 sec
Creating Circuit Boards with Open Source tools

- Design
- Prototype
- Enter Schematic
- Draw Layout
- Build
2. This means that (a) the voltage across $R_2$ is $V_{out}$ and (b) the voltage across $V_{in}$.

Figure 4.4. Inverting amplifier.

3. So, using rule II, we have

$$V_{out}/R_2 = -V_{in}/R_1$$

In other words,

voltage gain = $V_{out}/V_{in} = -R_2/R_1$

A bit old but still an excellent way to learn design
Good book in case you:
• Have seen it before but have forgotten
• Need more details about some aspect of op amps
Electrical Engineering and Computer Science

Electrical engineering, originally taught at MIT in the Physics Department, became an independent degree program in 1882.

The Department of Electrical Engineering was formed in 1902, and occupied its new home, the Lowell Building, when MIT was still located near Copley Square in Boston. The Department dedicated its present facilities in the Sherman Fairchild Electrical Engineering and Electronics complex in fall 1973, and a year later, it recognized its growing activity in computer science by changing its name to Electrical Engineering and Computer Science. The Department’s activities in computer science, communications, and control moved into the architecturally unique and exciting Ray and Maria Stata Center for Computer, Information, and Intelligence Sciences in Spring 2004.

The primary mission of the Department is the education of its students. Its three undergraduate programs attract more than 30 percent of all MIT undergraduates, and its doctoral programs are highly ranked and selective. A leader in cooperative education, the Department has operated the highly successful VI-A Internship Program since 1917. It has recently established a five-year Master of Engineering program, under which students stay for a fifth year and receive simultaneously a Bachelor’s degree and a Master’s of Engineering degree.
Manhattan Breadboarding

See http://www.k7qo.net
Disks provide insulation

- Made from copper-clad PC board material
- Cut into a circle with a sheet metal punch
The Prototype
Why Open Source CAD Software?

- Plenty of closed-source CAD software to choose from at the day job
- Personal preference for Open Source tools
- Wouldn't want to spend my own money on any of the closed-source tools anyway
- Avoid vendor lock-in
gEDA/gaf + PCB

• gEDA: GPL'd suite of Electronic Design Automation tools
• gaf: gschem and friends
• Schematic capture, netlister, symbols, symbol checker, and utilities
• PCB: printed circuit board layout program
• Relatively easy to build from source code
The Environment

- Everything I will talk about today I built from source on top of a clean install of Fedora 8
- There are some dependency issues: in some cases I had to figure out which package was missing and install it
- I chose the gEDA tools, others are available
File Formats

- All file formats are human-readable ASCII
- All files can be created and edited with the GUI
- Some configuration files hand-edited
- Avoid vendor lock-in: “Own your data”
Entering a Schematic

- Should be easy
- Should have good libraries
- Should link seamlessly with downstream tools
Entering the Schematic
; Start of color section

; Load up a color scheme has a light (almost white) background
; Comment out the first line and comment in the second line for a dark (black) background. The dark background is the original look.
;(load (build-path geda-rc-path "gschem-darkbg")) ; dark background
;(load (build-path geda-rc-path "gschem-lightbg")) ; light background
Changing the Guile

; Start of color section
;
; Load up a color scheme has a light (almost white) background
; Comment out the first line and comment in the second line for a
; dark (black) background. The dark background is the original look.
;
;(load (build-path geda-rc-path "gschem-darkbg")) ; dark background
(load (build-path geda-rc-path "gschem-lightbg")) ; light background
Light Background
Not Anti-aliased
Drawing Schematics for Looks

- Documentation
- Manuals
- Books
- Magazines

Requirements:
- Scalable fonts
- Vectors
- Work with Adobe Illustrator
SVG Symbols from Wikipedia
gEDA Symbol Library

• Many basic components are available
• They typically need customization
Symbol File Format

v 20050313 1
L 200 900 200 100 3 0 0 0 -1 -1
{
  T 250 450 3 8 1 1 0 0 1
device=MCP601
  T 200 2150 5 8 0 0 0 0 1
documentation=http://wb1.microchip.com/21314f.pdf
  T 200 1950 5 8 0 0 0 0 1
description=Single Supply Rail-to-Rail Opamp
  T 200 1750 5 8 0 0 0 0 1
  net=GND:4
  T 200 1550 5 8 0 0 0 0 1
  net=Vcc:7
  T 200 1350 5 8 0 0 0 0 1
  pins=8
  T 200 1150 5 8 0 0 0 0 1
class=IC
}
L 300 850 300 750 3 0 0 0 -1 -1
L 250 800 350 800 3 0 0 0 -1 -1
L 250 200 350 200 3 0 0 0 -1 -1
P 0 800 200 800 1 0 0

{  
  T 50 850 5 10 1 1 0 0 1
  pinnumber=3
  T 100 600 5 10 0 1 0 0 1
  pinseq=1
}
P 0 200 200 200 1 0 0
{
  T 50 250 5 10 1 1 0 0 1
  pinnumber=2
  T 50 50 5 10 0 1 0 0 1
  pinseq=2
}
P 900 500 1100 500 1 0 1
{
  T 950 550 5 10 1 1 0 0 1
  pinnumber=6
  T 950 300 5 10 0 1 0 0 1
  pinseq=3
}
L 200 900 200 1000 3 0 0 0 -1 -1
L 200 100 200 0 3 0 0 0 -1 -1
L 200 0 900 500 3 0 0 0 -1 -1
L 900 500 200 1000 3 0 0 0 -1 -1
T 800 800 8 10 1 1 0 0 1
refdes=U?
Why not XML?

- Installed base of code and data
- Development time
- Don't want two formats
- Formats were designed by developers who admire lisp/scheme/guile
- More compact than XML and somewhat similar in functionality
PCB Workflow

schematic

layout
PCB Workflow

gschem → gsch2pcb → pcb

schematic → layout
PCB Workflow

Symbol library

gschem

.sch

gsch2pcb

.pcb

Footprint library

.pcb
Footprint Library

Element(0x00 "Dual in-line package, narrow (300 mil)" "" "DIP8" 220 100 3 100 0x00)
(
    Pin(50 50 60 28 "1" 0x101)
    Pin(50 150 60 28 "2" 0x01)
    Pin(50 250 60 28 "3" 0x01)
    Pin(50 350 60 28 "4" 0x01)
    Pin(350 350 60 28 "5" 0x01)
    Pin(350 250 60 28 "6" 0x01)
    Pin(350 150 60 28 "7" 0x01)
    Pin(350 50 60 28 "8" 0x01)

    ElementLine(0 0 0 400 10)
    ElementLine(0 400 400 400 10)
    ElementLine(400 400 400 0 10)
    ElementLine(0 0 150 0 10)
    ElementLine(250 0 400 0 10)
    ElementArc(200 0 50 50 0 180 10)
    Mark(50 50)
)

www.gedasymbols.org
Footprint library

- Many footprints are available
- They need checking
- Hole sizes are often wrong
Library Management

• Keep a private library of symbols and footprints that are known to work
• Do not use library parts of unknown quality!
• Reset the library path to keep bad stuff out
• I use a library local to the design:
  
  (reset-component-library)
  (component-library "./symbols")
Searching the gEDA mailing list

- Search the gEDA mailing list
- Copies of mailing list archives are now often link-spam sites, such as osdir.com
- Want to get to: http://archives.seul.org/geda/user
- Gap: Need a custom Google search for all things geda!
Fabrication History
Photoplotter

Figures from www.artwork.com
Gerber (RS-274X) format

G04 Title: 750-0002, topsilk *
G04 Format: Gerber/RS-274X *
G04 PCB-Dimensions: 310000 235000 *
G04 PCB-Coordinate-Origin: lower left *
%MOIN*%
%FSLAX24Y24*%
%LNFRONTSILK*%
%ADD11C,0.0000*%
%ADD12C,0.0080*%
G54D11*X31000Y23500D02*Y0D01*
X3500D01*
Y23500D02*X31000D01*
X3500Y0D02*Y3000D01*
Y20500D02*Y23500D01*
X0Y3000D02*X50Y20500D01*
X3500Y3000D02*X0D01*
Y20500D02*X3500D01*
G54D12*X26230Y14060D02*Y13660D01*
gerbv
gerbv
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>D11</td>
<td>CIRCLE</td>
<td>0.020</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D12</td>
<td>CIRCLE</td>
<td>0.085</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D13</td>
<td>RECTANGLE</td>
<td>0.085</td>
<td>0.085</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D14</td>
<td>CIRCLE</td>
<td>0.091</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D15</td>
<td>RECTANGLE</td>
<td>0.091</td>
<td>0.091</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D16</td>
<td>CIRCLE</td>
<td>0.080</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D17</td>
<td>CIRCLE</td>
<td>0.112</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D18</td>
<td>CIRCLE</td>
<td>0.009</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D19</td>
<td>CIRCLE</td>
<td>0.012</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D20</td>
<td>CIRCLE</td>
<td>0.025</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D21</td>
<td>CIRCLE</td>
<td>0.010</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D22</td>
<td>CIRCLE</td>
<td>0.008</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D23</td>
<td>CIRCLE</td>
<td>0.040</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D24</td>
<td>CIRCLE</td>
<td>0.055</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D25</td>
<td>CIRCLE</td>
<td>0.061</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D26</td>
<td>RECTANGLE</td>
<td>0.055</td>
<td>0.055</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>D27</td>
<td>CIRCLE</td>
<td>0.152</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>
Getting PC Boards Built

- Clean/Fast/Cheap
- Good support
- Can use a credit card
- Electronic data transfer (RS274-X)
- www.4pcb.com
- www.sunstone.com
- www.apcircuits.com
Circuit Simulation

- Analog
- Digital
- Mixed Analog and Digital
Many versions of SPICE

- Most based on Berkeley SPICE3f5, SPICE2G6
- Rewrites/repackages (ngspice, TclSpice)
- Extensions (Cider, Xspice, Spectre)
- Commercializations, eval copies, educational versions (HSPICE, PSPICE)
- Corporate versions (TISPIECE, HPSPICE, Agilent Spice, Avago Spice, etc)
- Closed-source freeware (LTSpice)
ngspice

- Open Source
- Part of gEDA project
- Sourceforge
  - 12 Developers
  - 8 project administrators
  - 99.71 Activity Percentile
ngspice

• Text-based language
• Can be schematic-driven, but there is a text description underneath the GUI
Example

Title
Test circuit
VIN 1 0 AC 1
R1 1 2 1k
C1 2 0 .1u
.ac dec 40 10 10meg
.end

Circuit elements

Control statements
Graphics Output

vdb(2)

frequency
Analog Circuit Simulation Flow

gschem → .sch → gnetlist → .cki → ngspice → gattrib

Diagram with flowchart showing the steps involved in an analog circuit simulation, starting with gschem, followed by .sch, then gnetlist, .cki, ngspice, and finally gattrib.
FPGA Design Flow

Simulation

- verilog
  .v

- Icarus

Hardware

- .vcd
- .xnf

- Gtkwave
- Xilinx

Requires Xilinx Software
The Verilog Language

• Structural
  – About circuits, gates, latches, wires, buses

• Behavioral
  – About state machines, test structures, higher level things that would be otherwise tedious to build
module mylogic(a,b,c,d,e);
  input a;
  input b;
  output c;
  output d;
  output e;

  wire c,d,e;
  assign c = a|b;
  assign d = b|c;
  assign e = ! (c & d);
endmodule
Verilog Example

`include "mylogic.v"
module main();
reg j,k;
wire x,y,z;

initial begin
  j = 0;
  k = 1;
  $display("j\t\tk\tx\ty\tz\t");
  $monitor("%d\t%d\t%d\t%d\t%d\t%d", j,k,x,y,z);
end

mylogic my1 (j,k,x,y,z);
endmodule
Running from the command line

compile:  iverilog -o main main.v
run:  vvp main
output:  j k x y z
         0 1 1 1 0
IVI – Eclipse Plugin for Verilog

- Code editor
- Waveform view
- Data monitor view
Waveform and Data Monitor View

---

Minisrc_configuration (IVI Cover Verilog).cver
Test Sanity 2 PASSED !!!

----------------------------------------

-------- Starting Register File 1 Test --------

stop requested...

Sim Time: 2709000000ps

Launching Minisrc_c: (70%)
Recommendations from the BOF

● Open digital designs:
  – opencores.org

● Avoid Verilog pitfalls:
  – www.sunburst-design.com