GNU Radio: Lessons from the past, recommendations for taking it to the future

Eric Blossom
eb@comsec.com

European GNU Radio Days ‘24, 27th to 30th August 2024
FAIR - Facility for Antiproton and Ion Research in Europe GmbH

Introduction

Why this talk?

A bit of context

1993 NSA Introduces the “Clipper Chip” (MYK-78T)

Promoted to secure “voice and data communications”

80-bit symmetric encryption (“Skipjack”)

Diffie-Hellman Key Exchange

Had a built in “backdoor” (Key Escrow)

https://en.wikipedia.org/wiki/Clipper_chip

AT&T Introduces TSD-3600E Telephone Encryptor

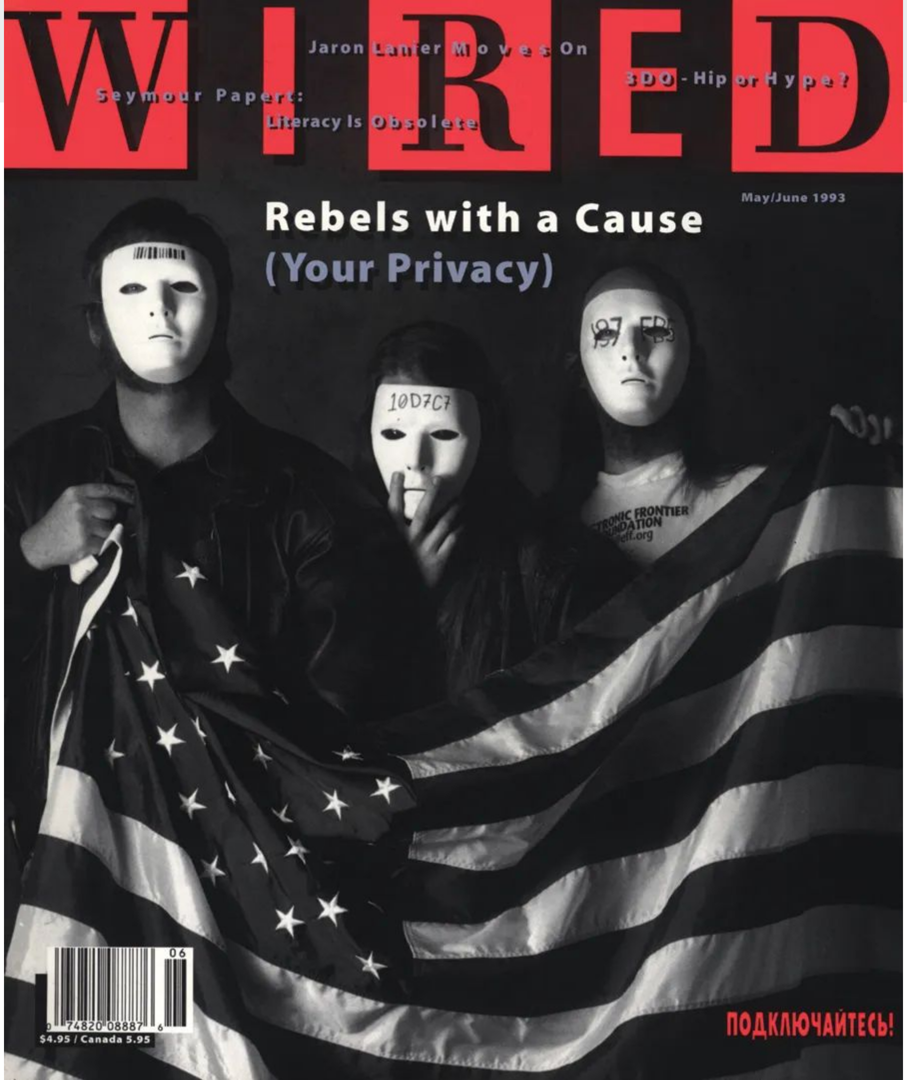
How hard could it be?

- Modem
- Crypto
- Vocoder

Cypherpunks!

A great place to meet interesting people!

https://en.wikipedia.org/wiki/Cypherpunk



John Gilmore

- Employee #5 at Sun Microsystems
- Co-Founder of Cygnus Solutions
- Co-Founder of the Electronic Frontier Foundation (EFF)

By Kushal Das - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=73312575

https://en.wikipedia.org/wiki/John_Gilmore_(activist)

Building (wired) telephone encryptors

Eric + team(s) built two generations of them (in two different companies)

1. GSP 8191 aka “Bat Phone” / POTS / 2048-bit Diffie-Hellman key exchange / 3DES
2. Starium 100 / in the handset cord / 2048-bit DH / 3DES / Optional battery operation

Both used 13,000 bit/s GSM full-rate vocoder and 4800 bit/s CELP vocoder.

The VP1 Protocol for Voice Privacy Devices Version 1.2

Ultimately discovered that “just because I want them, it doesn’t mean it’s a good business idea.”

I learned many useful and interesting things along the way.

Early North American Digital Cellular Standards

- IS-136 Digital AMPS (TDMA)
  - Had forward error correction on the wrong side of the crypto
- IS-95 CDMA
  - Had linear “crypto”
  - 42 equations in 42 unknowns (in Z_2)
  - Variable rate vocoder
  - Discontinuous transmission
  - Thus the end of a talk spurt was effectively known plaintext
  - There was a paper published on the attack (sorry, can’t find it now)

After crypto, the New Idea…

Let’s teach myself **Digital Comms!**

My Dinner with Gilmore

Where to start?

Step 1: Assemble an AM/FM radio kit that is built out of discrete components (14 transistors)

Step 2: Look for existing free SDRs

MIT SpectrumWare Project (1)

Per 2017 archive.org capture:

SpectrumWare Team: Matt Welborn, Sunil Rao, Ripal Nathuji, Rattapoom Tuchinda, John Ankcorn, Steve Garland, John Guttag

Past members: Vanu Bose, Matt Brown, Andrew Chiu, Mike Ismert, Mike Saginaw, Dave Tennenhouse, Brett Vasconcellos

http://www.sds.lcs.mit.edu/SpectrumWare/home.html
https://web.archive.org/web/20171218132859/http://nms.csail.mit.edu/projects/spectrumware/

MIT SpectrumWare Project (2)

Wrote 14 papers. There are some great ideas in some of them.

Generated something like 5-ish PhDs

pspectra code base (dead link) ftp://salsa.lcs.mit.edu/pub/jca/pspectra/pspectra2000_05_09.tgz

Guppi: Custom PCI board with A/Ds and D/As
Guppi Overview: http://www.sds.lcs.mit.edu/SpectrumWare/guppi.html

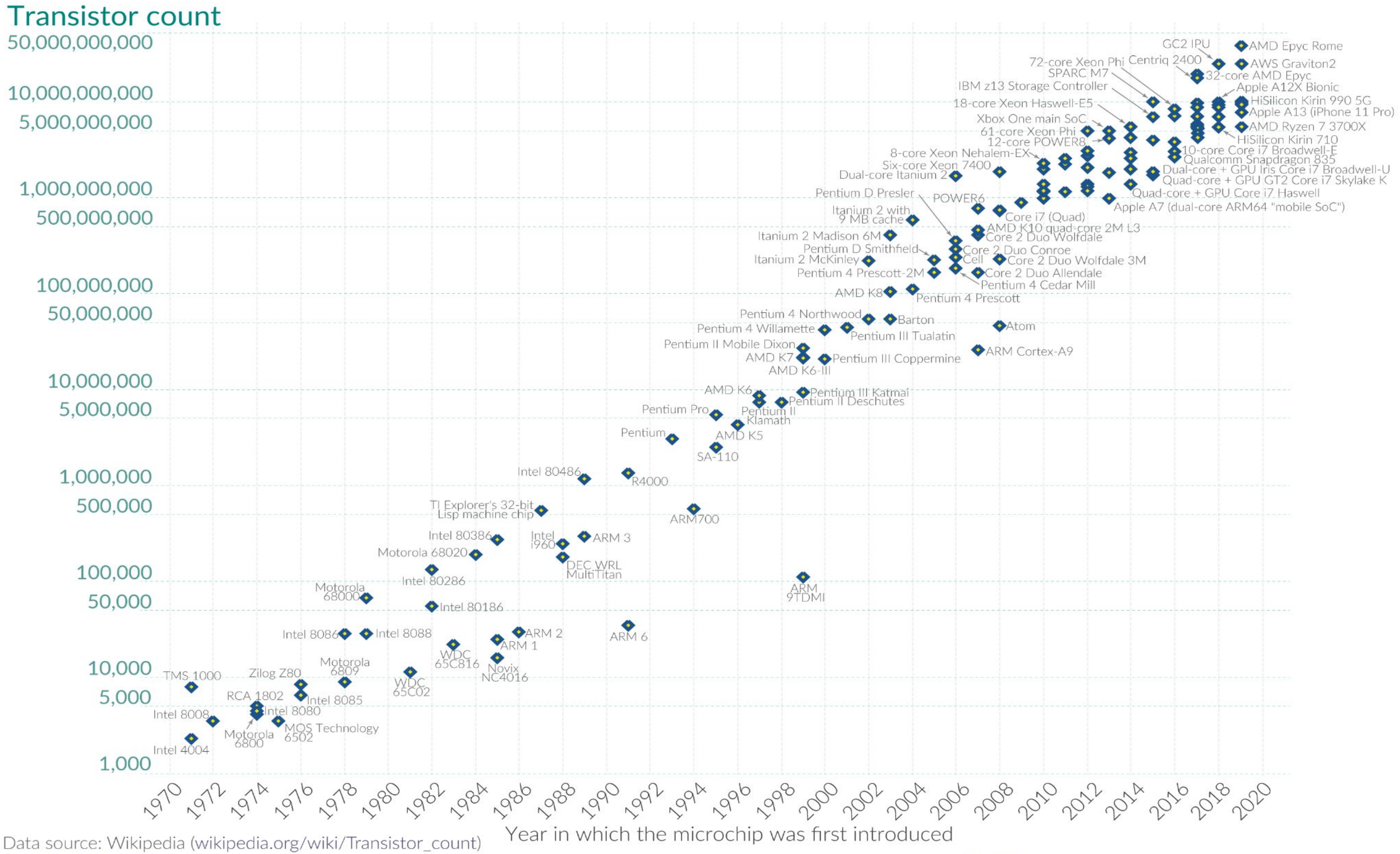
First Commit! Start of pspectra clean up

```
2001-09-18  Eric Blossom  <eb@comsec.com>

     * src/pspectra/lib/vrio/VrAudioSink.h (work): Return value was off
       by a factor of sizeof(iType).

     * src/pspectra/lib/vrio/VrFileSink.h (VrFileSink, setFilename, getFilename):
       fixed strlen related buffer overrun.
```

Moore’s Law: The number of transistors on microchips doubles every two years

Our World in Data

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.



Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)

OurWorldinData.org – Research and data to make progress against the world’s largest problems.

Licensed under CC-BY by the authors Hannah Ritchie and Max Roser.

https://ourworldindata.org/uploads/2020/11/Transistor-Count-over-time.png

Matt Ettus

Matt was involved from very near the beginning.

IEEE Fellow in August 2024

Initial hardware

Measurement Computing **MC4020** 4-channel PCI A/D board
I wrote a streaming driver for it, similar to the interface that the GuPPI provided.
We ran it in **single channel** mode, **20M Samples/s** (limited by PCI bus bandwidth).
The MC4020 cost more than the PC.

RF front end was a cable tuner eval board.
It would tune from about 57MHz to 800-something MHz.
5.75 MHz IF output (6MHz wide US TV channel fit)

Initial software target

Receive broadcast FM.

This exercised quite a few things:

Samples in, software downconversion and channel selection

FM demodulation

Audio output

“The two clock problem” (not solved in general, perhaps even today)

Next Target: ATSC

- ATSC is the US terrestrial digital TV waveform
- It is a reasonably complex waveform, and requires a large number of signal processing blocks.
- We built an ATSC modulator and a demodulator based on a suggestion from Gilmore.
- EFF was interested in the proposed “Broadcast Flag” and how we could kill it using a “software is First Amendment protected speech” argument. It was eventually killed using a different legal argument. See ALA-v-FCC.

gnuradio-0.9

Was the first public release of GNU Radio. It released 2003-12-31, and was about 30% pspectra code and 70% new code. It was also the end of that development effort. From beginning to first release was about 2 years of effort.

The USRP beginnings

The work on what would become the USRP most likely started in sometime in 2002.

[See if Matt has any better idea about this]

The first software commit on the USRP 0 firmware was 2003-07-30.

The USRP 0 was very similar to what ended up being the USRP 1, but without a built in RF front end. We used the micotune cable modem tuner as the RF front end.

GNU Radio (2.2)

This was a restart using everything we learned building gnuradio-0.9.

It contains the initial implementations of the single writer/multi reader buffers, the flowgraph, streaming blocks, the “single threaded scheduler”, and support for the USRP.

2.2 had python bindings and almost all of the unit test code was in python.

This work started in January 2004 and GR 2.2 was released on 2004-10-11.

Early funding

Community building

Acknowledgements

Looking to the future

Now what?

With 4.0 we are at a transitional stage.

What communities are using GR?

A few ideas and questions

- How many people are currently contributing to the core of GNU Radio (3.10 and/or 4.0)?
- How many of them are being paid/incentivized somehow?
  - Part of their job
  - (Grad) Students
  - Consultant
  - etc.

Uses of GR in teaching / learning / research?

- Where? Who? What level?
- Number of educational grants that are somehow using GR (e.g., NSF in US)?
- PI or Co-PIs that we know are using GR
- …

Questions / Discussion / Inquiry