Development and Design of Firmware Programming Tools for the openHPSDR Hardware

David R. Larsen[1]

[1]KV0S
Columbia, Missouri, USA
OpenHPSDR Development Group
openhpsdr.org

Digital Communication Conference, 2017
St Louis, Missouri

Profession

- Professor of Forestry.
- Work on Forest Dynamics.
- Growth Modeling.
- Statistical Analysis.
- Code development for Forestry applications.
- Unix, Linux programmer.

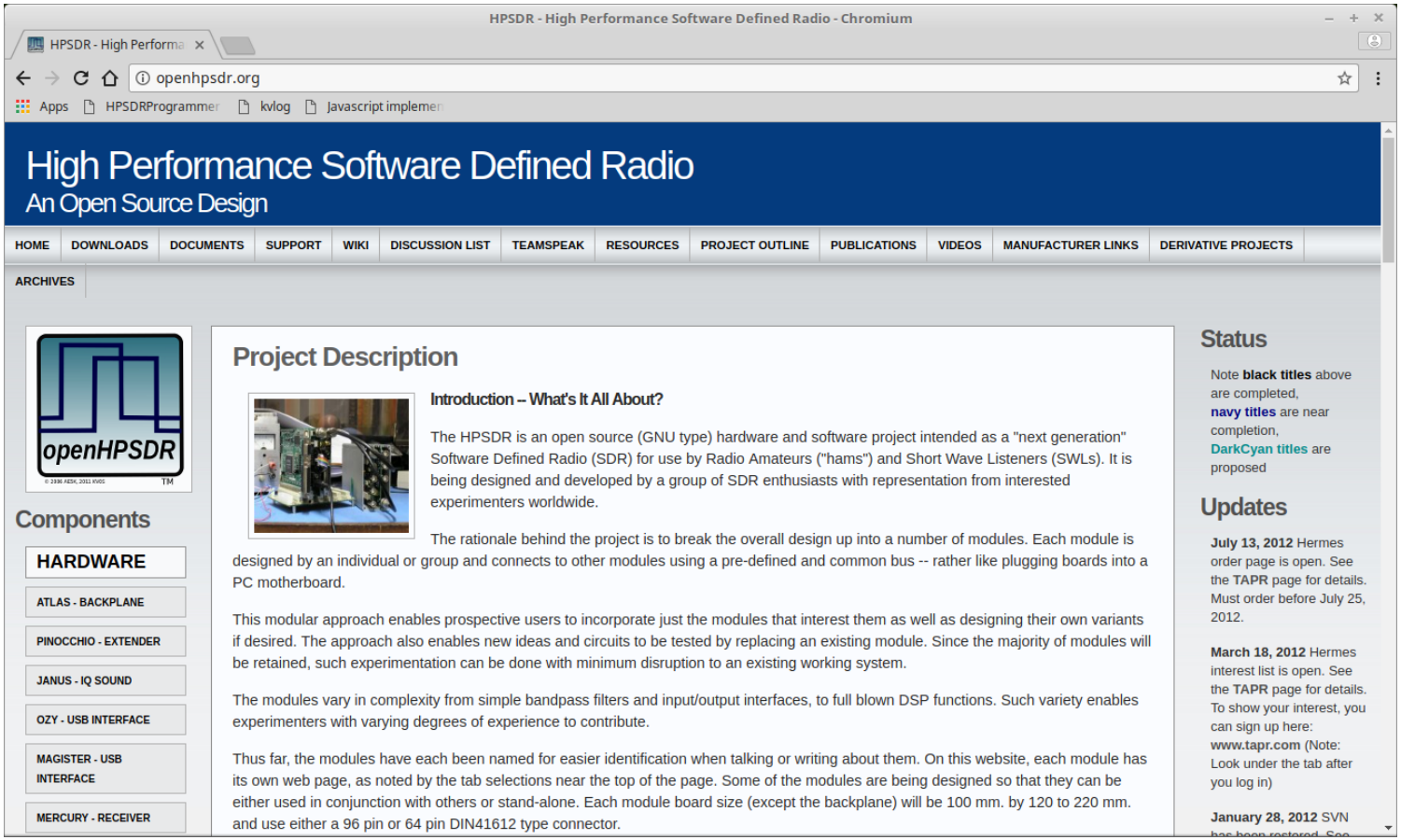
Amateur Radio

- First licensed in 1975.
- Live happened!
- Relicensed in 2004.
- First Dayton Hamvention, 2007.
- First DCC, Chicago, 2008.

openHPSDR.org

- Scotty asked me to take it on in 2008.
- Also started and attend the Friday night openHPSDR teamspeak sessions.
- Role in organizing the repositories.

openHPSDR.org



openHPSDR.org

- Maintain the legacy of old work completed and not.
- Help people to find what they need to get started.
- Archive teamspeak, listserver, code, papers and videos.
- The Wiki.

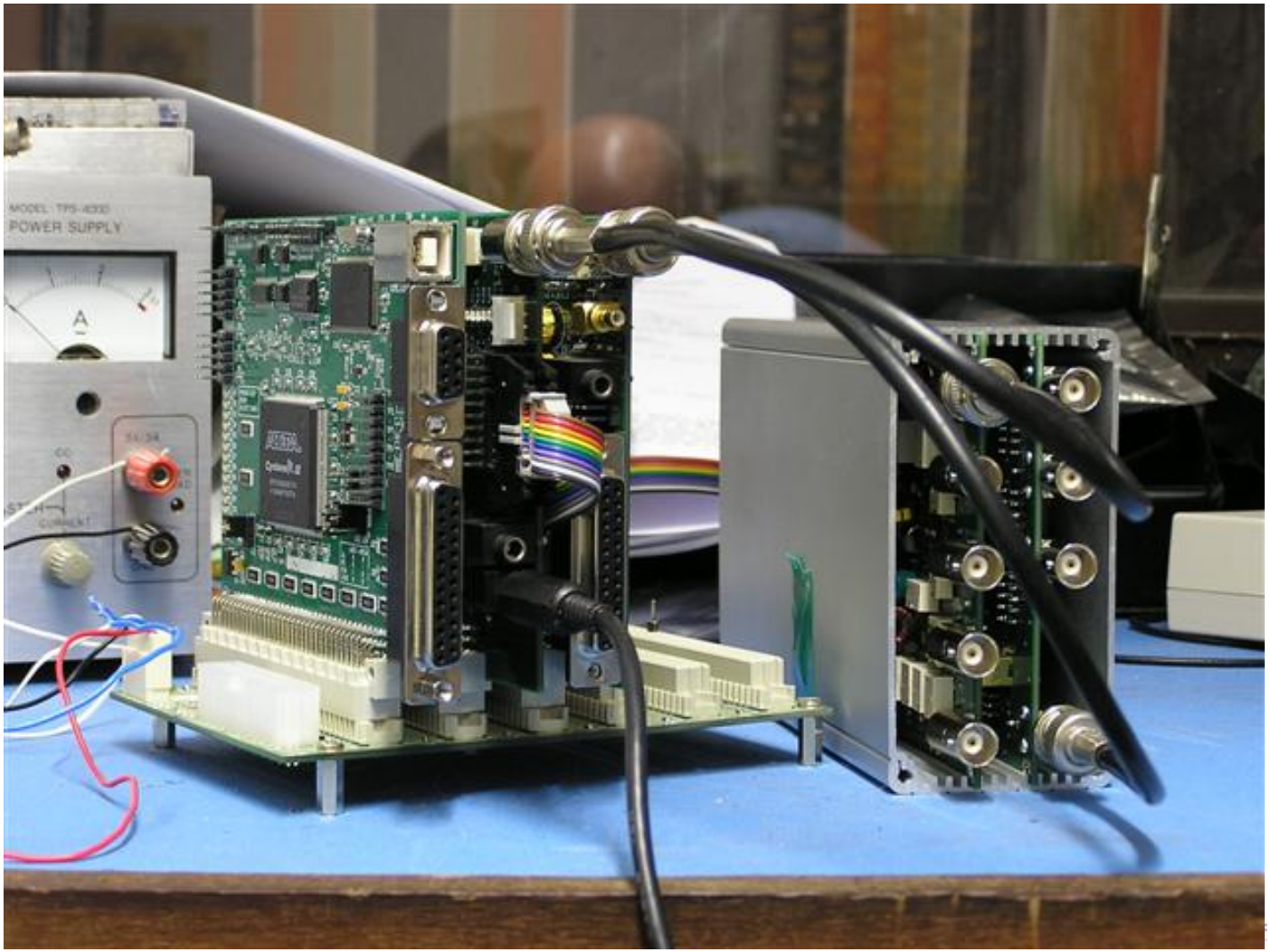
Early days

- First Boards.
  - Atlas back backplane
  - Janus sound interface
  - Ozymandias first board with
    - Cyclone II FPGA,
    - USB interface
    - Loaded firmware from PC through the USB line

First radio

- Penelope
  - Digital up converter(DUC) a 1/2-watt transmitter/exciter board,
  - Cyclone II FPGA
  - Initially programmed with a USB Blaster, Quartus programmer, using JTAG or Flash memory.
- Mercury
  - A 0-65MHz Direct Sampling Receiver
  - Cyclone III FPGA
  - Initially programmed with a USB Blaster, Quartus programmer, JTAG or Flash memory.

First radio



Search for Bandwith

- Problems with USB2
  - Limited bandwidth.
  - USB Driver issues.
    - Windows Centric
    - General not open source
    - No Volunteers to write drivers
    - USB3 a possibility but not widely in 2010
  - Ethernet as a alternative
    - Very stable drivers in all OS platforms.
    - Sufficient bandwidth.

A New Interface Board

- Metis
  - Ethernet Interface module.
  - This required writing Basic TCPIP interface code in FPGA firmware.
  - Introduction of the Bootloader Mode.
  - Introduction of in Metis JTAG programmer.
  - Introduction of simplified Programmer.
  - Communication by raw PCAP or UDP protocols

Tools for programming firmware.

- First Tool HPSDRBootloader by John, G0ORX
  - Used PCAP
  - Written in C++ with Qt GUI
    - Required Administrator login.
    - Required setting jumpers on board.
    - Work flow was confusing to ocaisional users.

Structuring Programs

- In 2012, I started maintain and writing programmer code.
- Seperate functionality.
- Use PCAP for recovery.
- Use UDP for normal updates.

Structuring Programs

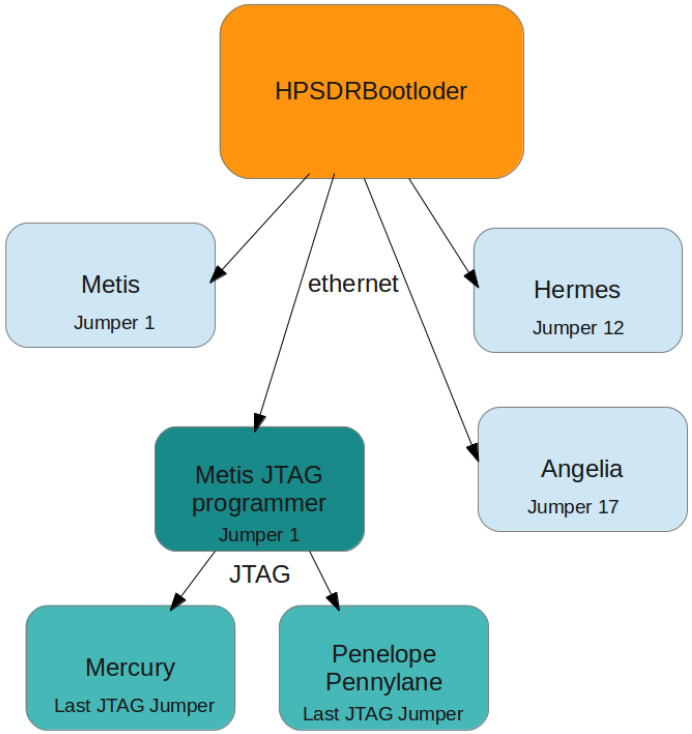
- HPSDRBootloader is a program for recovery from programming failure.
- HPSDRProgrammer is a program for normal updating.

HPSDRBootloader

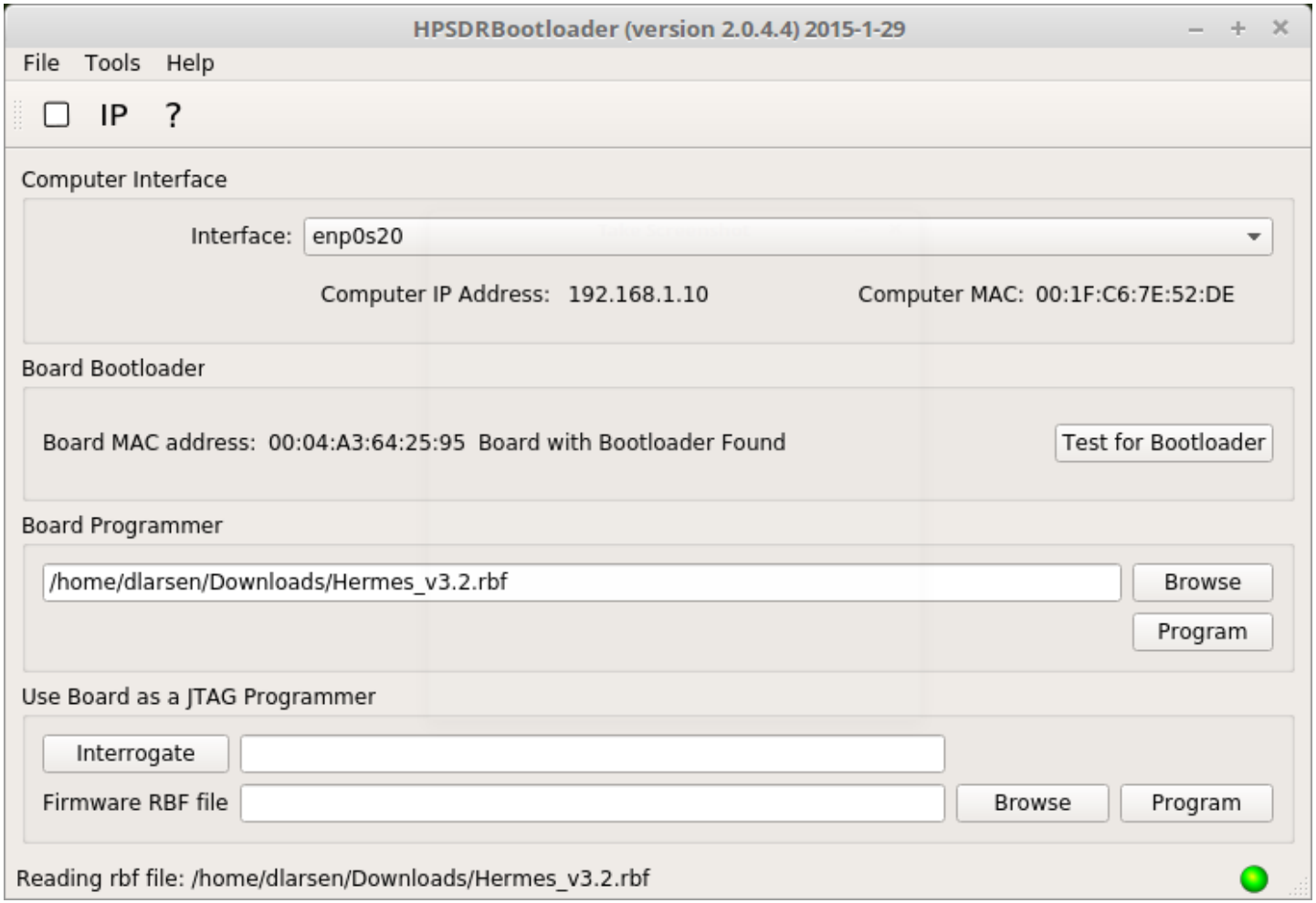
Design Criteria

- Use PCAP protocol. (MAC Addressing)
- Require Adminstrator login on computer.
- Require jumpers being set on PCB board.
- Board Discovery uses bootloader firmware
- Allow Metis to function as a JTAG programmer for Penelope and Mercury.

HPSDRBootloader



HPSDRBootloader

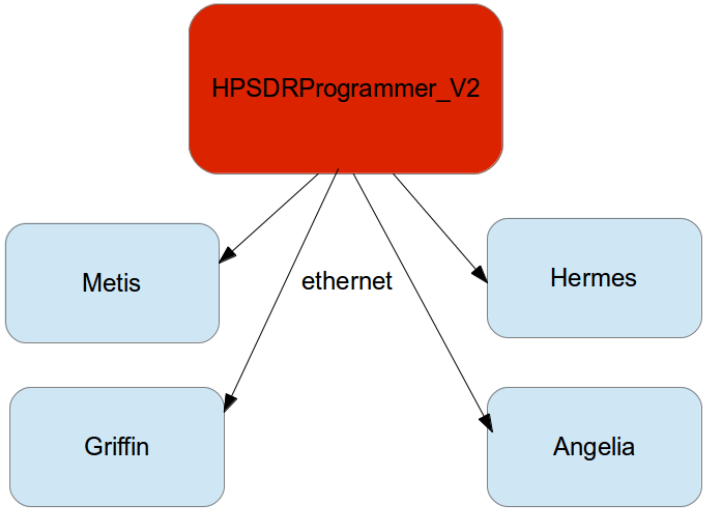


HPSDRProgrammer

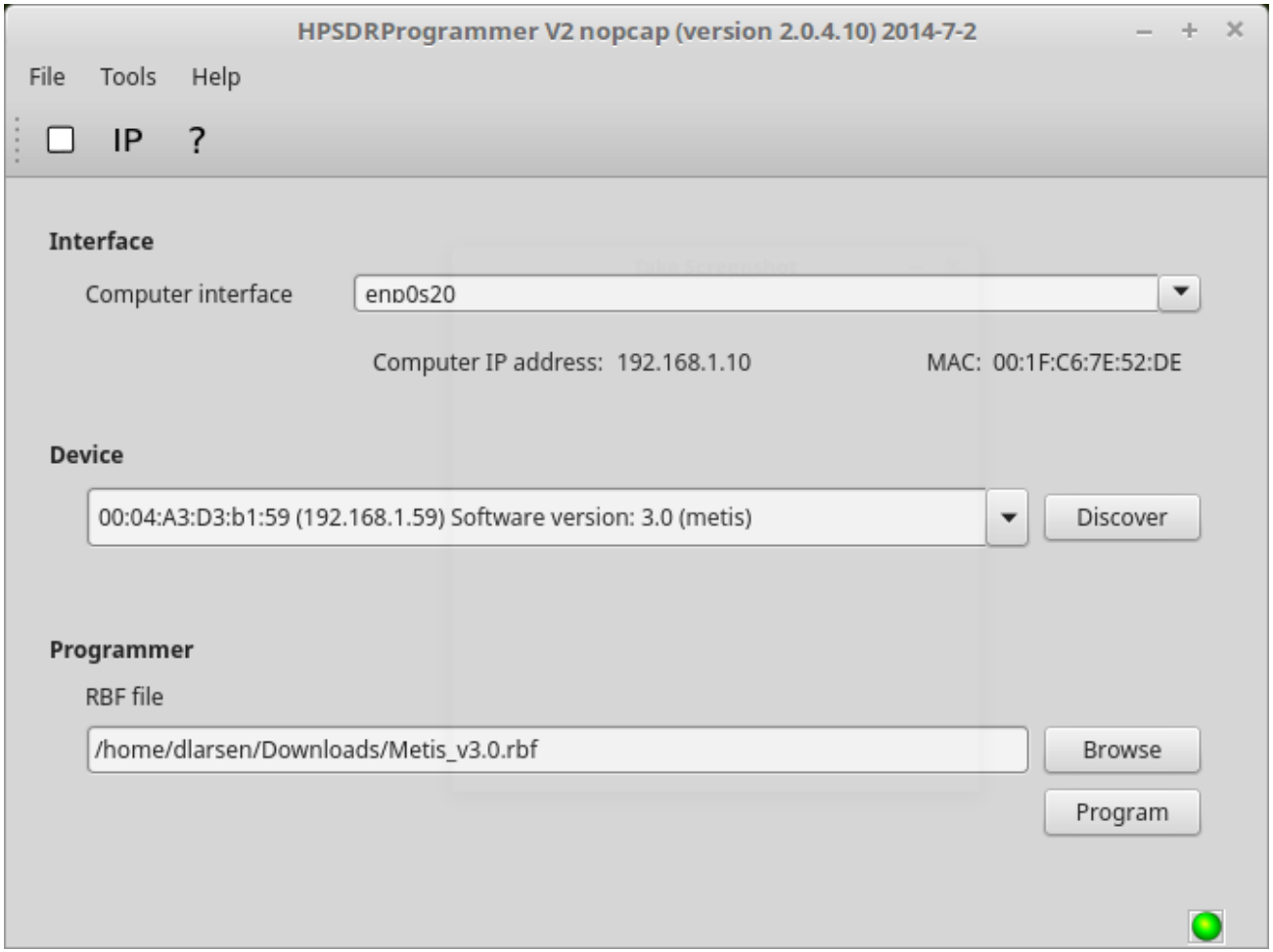
Design Criteria

- Use UDP protocol. (TCPIP Addressing)
- No jumpers being set on PCB board.
- Board Discovery uses past Radio firmware
- Can only program boards with an ethernet connector

HPSDRProgrammer



HPSDRProgrammer



All is well but time Marches On

Maintainence issues

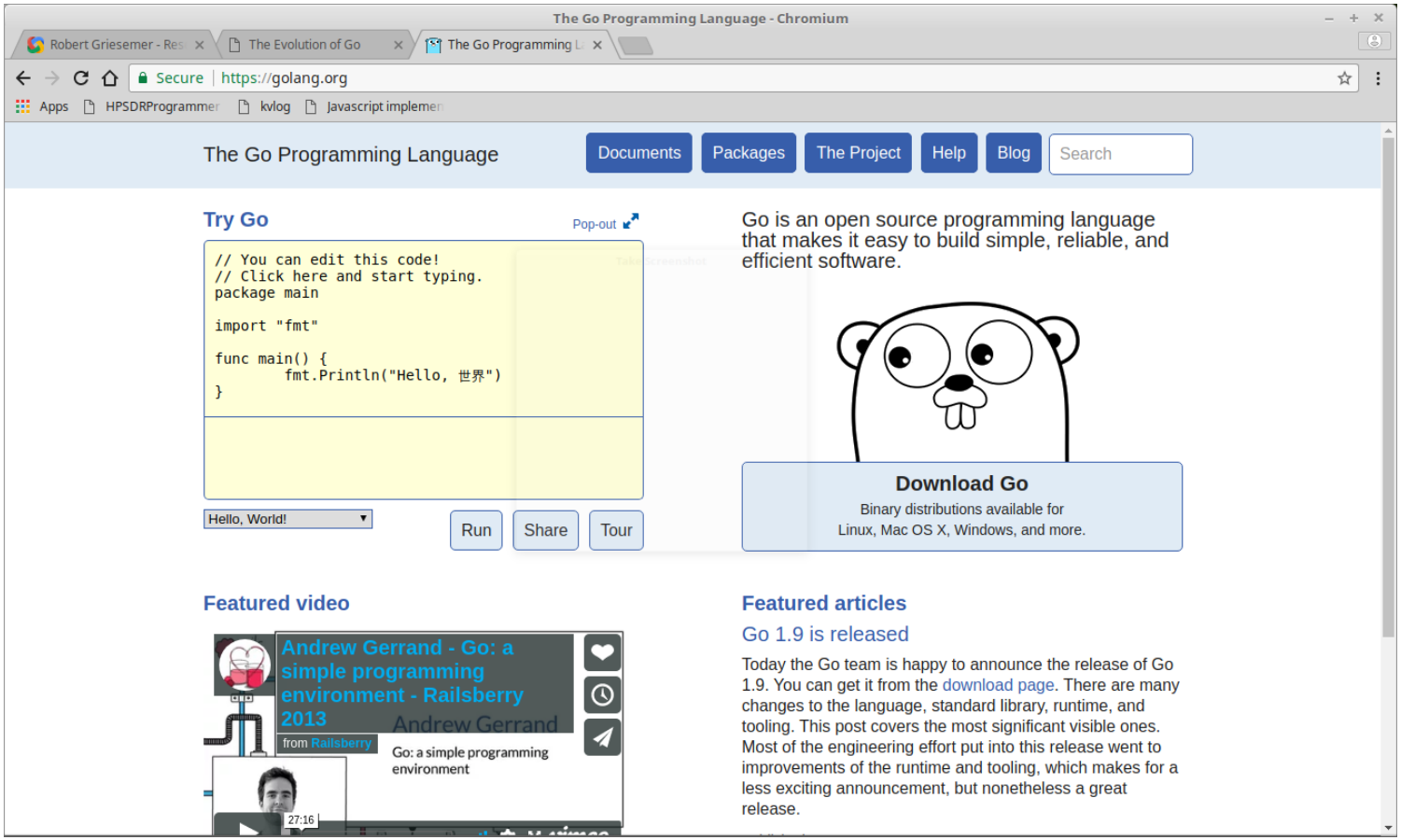
- Stable and working.
- Changes in the Code (Easy to fix)
- Changes in installers (Easy if you use all the time, Hard if you use once every 3 years).
- Code detritus (Unused left over bits and pieces).

This is my Hobby!

Why start over?

- This is my hobby! I want to learn something new.
- Learning a new computer language.
- Want to clean up the structure and process.

Go Language


The Go Programming Language - Chromium
Robert Griesemer - Res
The Evolution of Go
The Go Programming L
Secure | https://golang.org
Apps
HPSDRProgrammer
kvlog
Javascript implemen
The Go Programming Language
Documents
Packages
The Project
Help
Blog
Search
Try Go
Pop-out
// You can edit this code!
// Click here and start typing.
package main
import "fmt"
func main() {
fmt.Println("Hello, 世界")
}
Hello, World!
Run
Share
Tour
Go is an open source programming language that makes it easy to build simple, reliable, and efficient software.
Download Go
Binary distributions available for Linux, Mac OS X, Windows, and more.
Featured video
Andrew Gerrand - Go: a simple programming environment - Railsberry 2013
Andrew Gerrand
from Railsberry
Go: a simple programming environment
27:16
Featured articles
Go 1.9 is released
Today the Go team is happy to announce the release of Go 1.9. You can get it from the download page. There are many changes to the language, standard library, runtime, and tooling. This post covers the most significant visible ones. Most of the engineering effort put into this release went to improvements of the runtime and tooling, which makes for a less exciting announcement, but nonetheless a great release.

Go Language

Go Language (golang.org)

- Has the flavor of C.
- First developed at Google but has been open sourced
- Original Three designers.
  - Ken Thompson
  - Rob Pike
  - Robert Greismer

Go Language

Ken Thompson

- Formerly Bell Labs.
- Wrote the B programming language.
- Wrote Plan 9.

Go Language

Rob Pike

- Formerly Bell Labs.
- Author on The Practice of Programming and The Unix Programming Environment, and UNIX
- Author on Plan 9.
- Author of UTF-8

Go Language

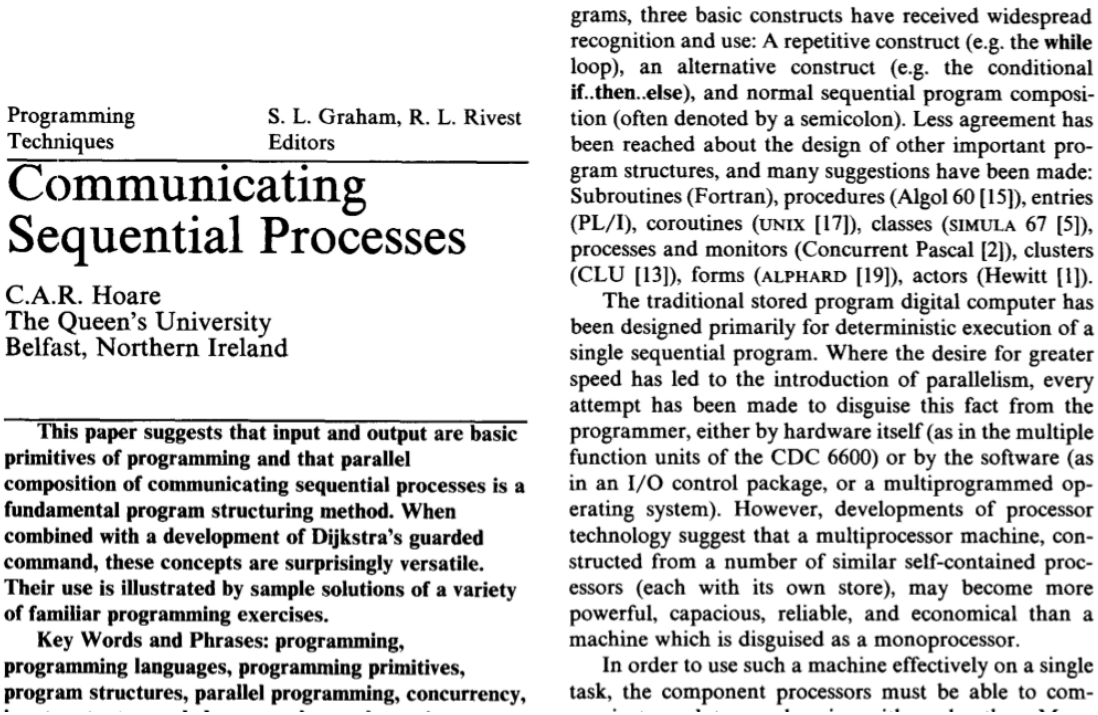
Robert Greismer

- Formerly Bell Labs.
- Wrote Sawsall
- Little C background

Go Language

C. A. R. Hoare. 1978. Communcating Sequential Processes. Communications of the ACM. 21(8):666-677.

- The group was influenced by Hoare



Programming Techniques | S. L. Graham, R. L. Rivest Editors

Communicating Sequential Processes

C.A.R. Hoare
The Queen's University
Belfast, Northern Ireland

This paper suggests that input and output are basic primitives of programming and that parallel composition of communicating sequential processes is a fundamental program structuring method. When combined with a development of Dijkstra's guarded command, these concepts are surprisingly versatile. Their use is illustrated by sample solutions of a variety of familiar programming exercises.

Key Words and Phrases: programming, programming languages, programming primitives, program structures, parallel programming, concurrency,

grams, three basic constructs have received widespread recognition and use: A repetitive construct (e.g. the **while** loop), an alternative construct (e.g. the conditional **if..then..else**), and normal sequential program composition (often denoted by a semicolon). Less agreement has been reached about the design of other important program structures, and many suggestions have been made: Subroutines (Fortran), procedures (Algol 60 [15]), entries (PL/I), coroutines (UNIX [17]), classes (SIMULA 67 [5]), processes and monitors (Concurrent Pascal [2]), clusters (CLU [13]), forms (ALPHARD [19]), actors (Hewitt [1]).

The traditional stored program digital computer has been designed primarily for deterministic execution of a single sequential program. Where the desire for greater speed has led to the introduction of parallelism, every attempt has been made to disguise this fact from the programmer, either by hardware itself (as in the multiple function units of the CDC 6600) or by the software (as in an I/O control package, or a multiprogrammed operating system). However, developments of processor technology suggest that a multiprocessor machine, constructed from a number of similar self-contained processors (each with its own store), may become more powerful, capacious, reliable, and economical than a machine which is disguised as a monoprocessor.

In order to use such a machine effectively on a single task, the component processors must be able to com-

Go Language

Feature of Go that I found helpful in this project.

- Function testing.
- Cross platform compiling. Linux, Windows, MacOS, Arm, BSD.
- Good packages for networking including PCAP, UDP, HTTP.
- Static binaries.

Go Language

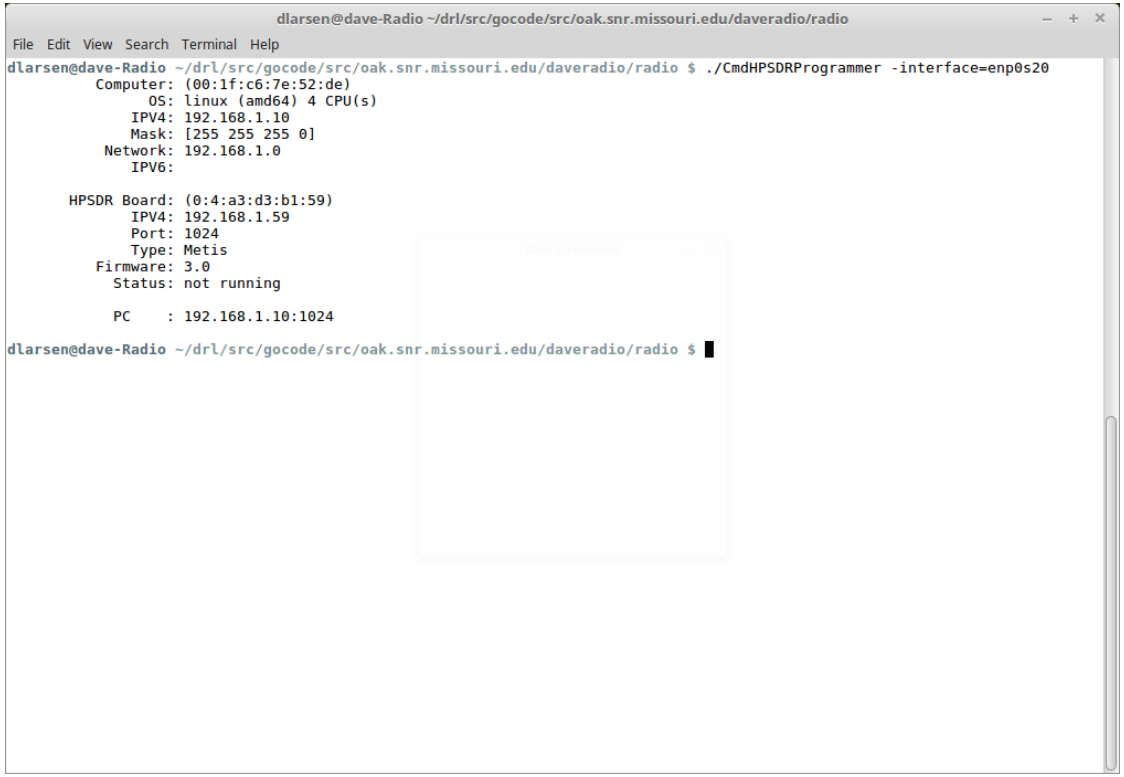
The disadvantages of Go

- No GUI package, either Command line or HTTP

First Programmer in Go

- Started the process before Protocol 2 was available for testing.
- So Started with Protocol 1
- Built and test each component functions.
- Use the command line interface.

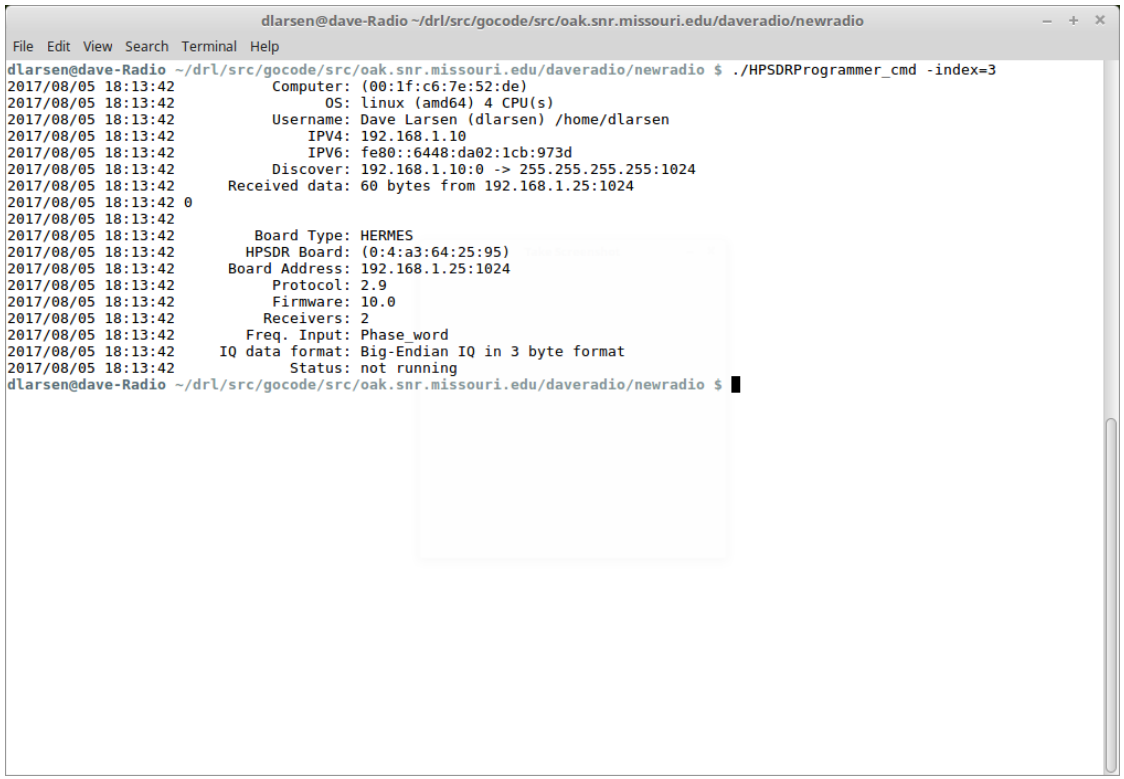
First Programmer in Go



Second Programmer in Go

- Started the process Protocol 2 building to the Protocol.
- Built and test each component functions.
- Use the command line interface.
- Very few corrections to make the functions work once Protocol 2 was available.

Second Programmer in Go



Programmers in Go

- At this point I have two fast and clean packages to program HPSDR boards.
- Shared these with others for testing.
- Most users we uncomfortable withteh command line interface.

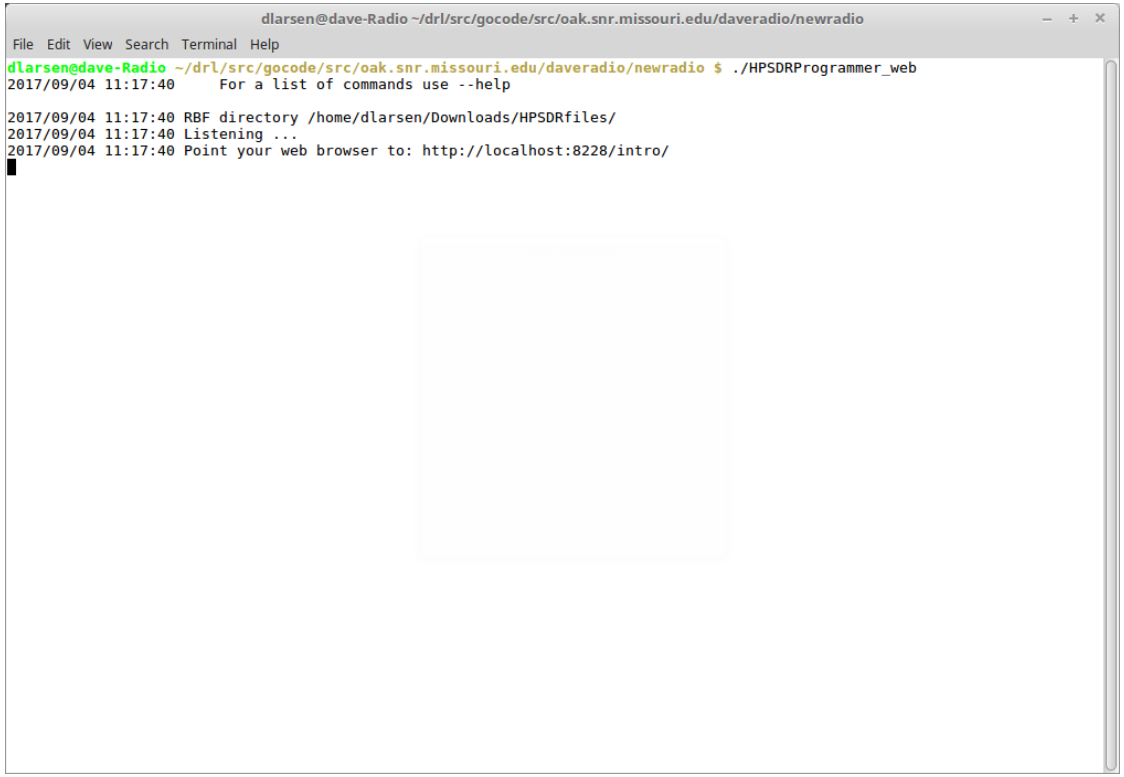
The next step

- Go have the package to allow the creation of small local httpservers
- The next step is to make a HTTP interface for the packages.
- Program can be contained in a single executable file.
- The server cannot be access outside of the excuting computer, unless you configure that access.

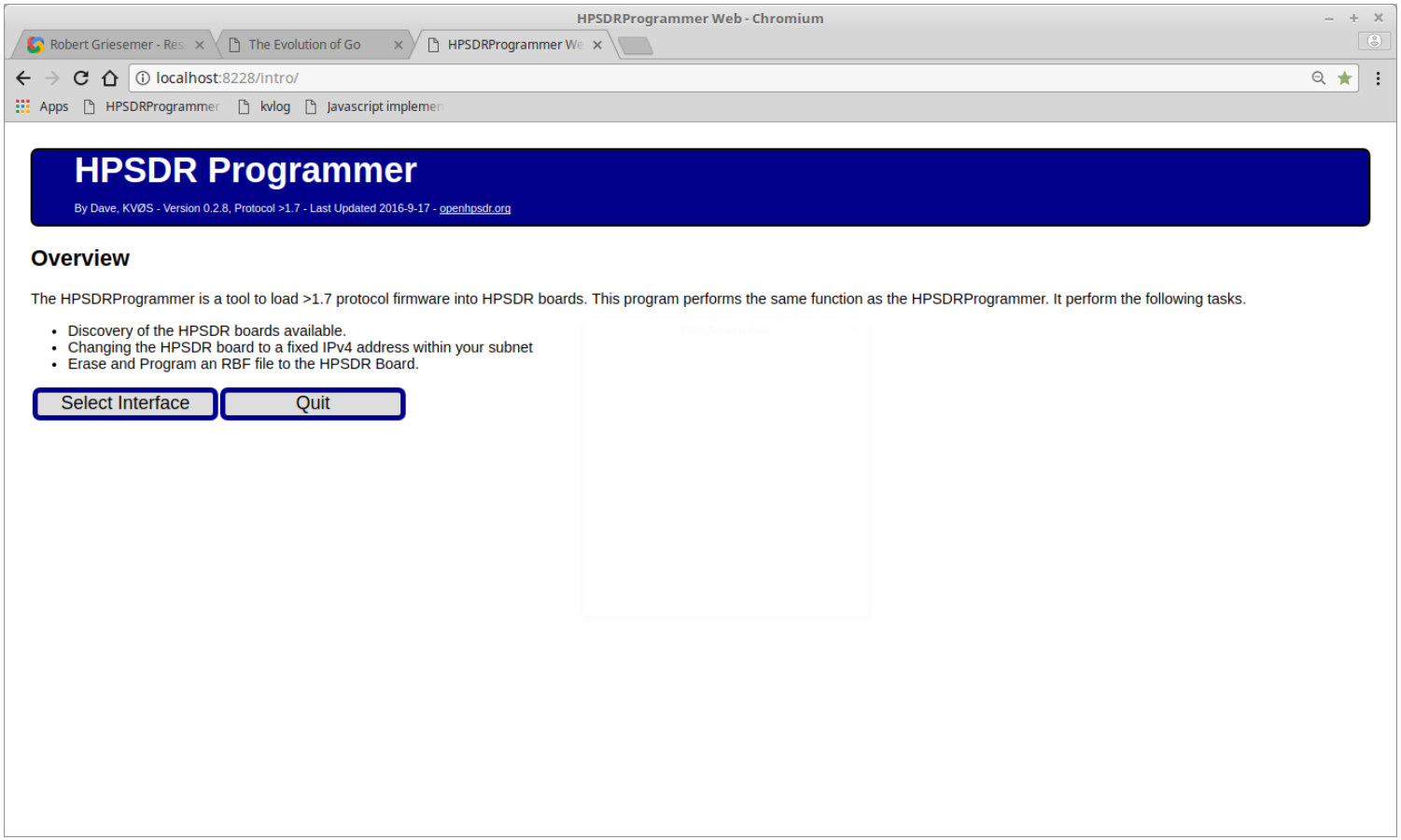
Local Web Programmer

- In this process, I just had to work on the user interface.
- The packages from could be resued without modification.
- In general the extra code is mostly html.
- I need a small it of javascript to improve user feedback.

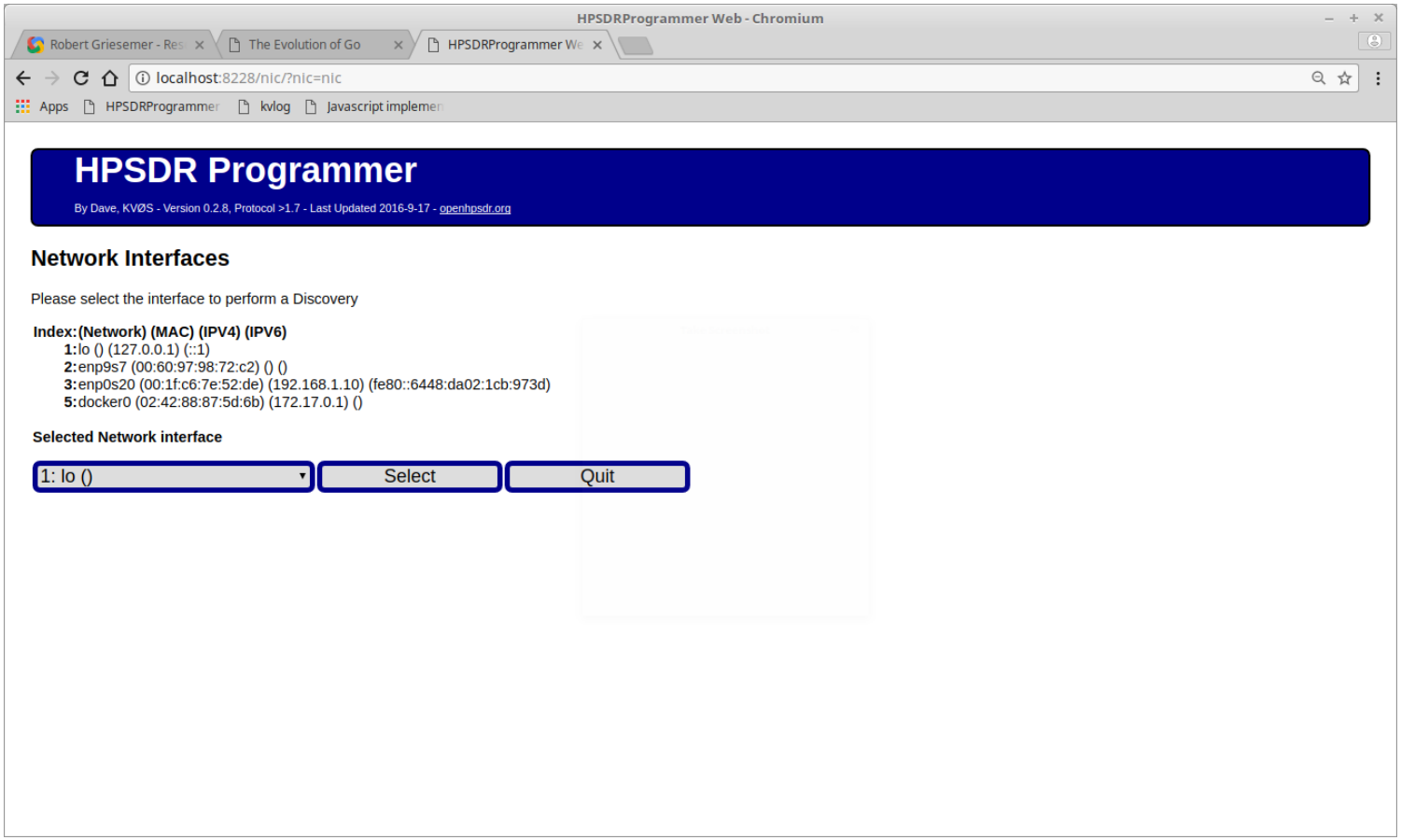
Local Web Programmer



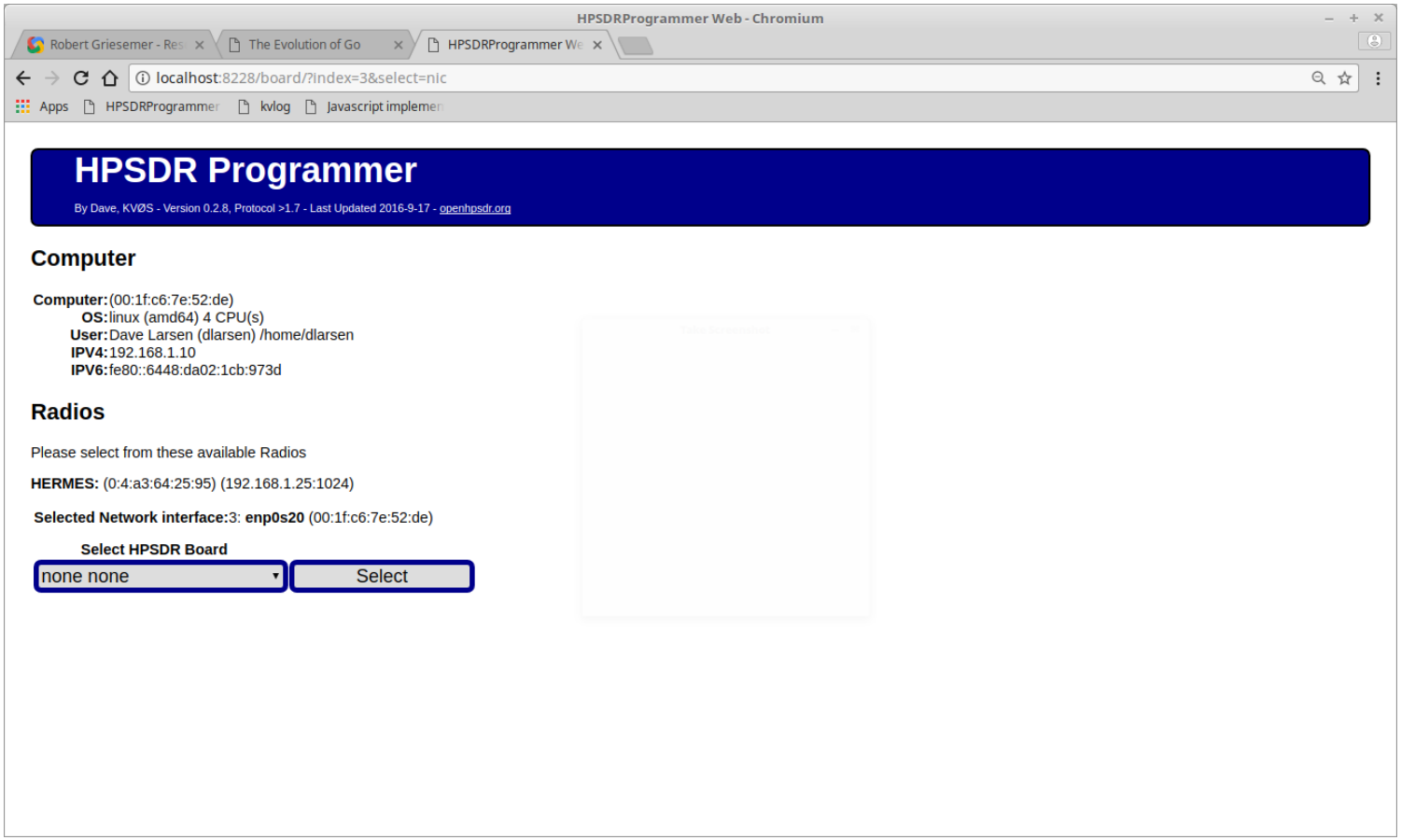
Local Web Programmer



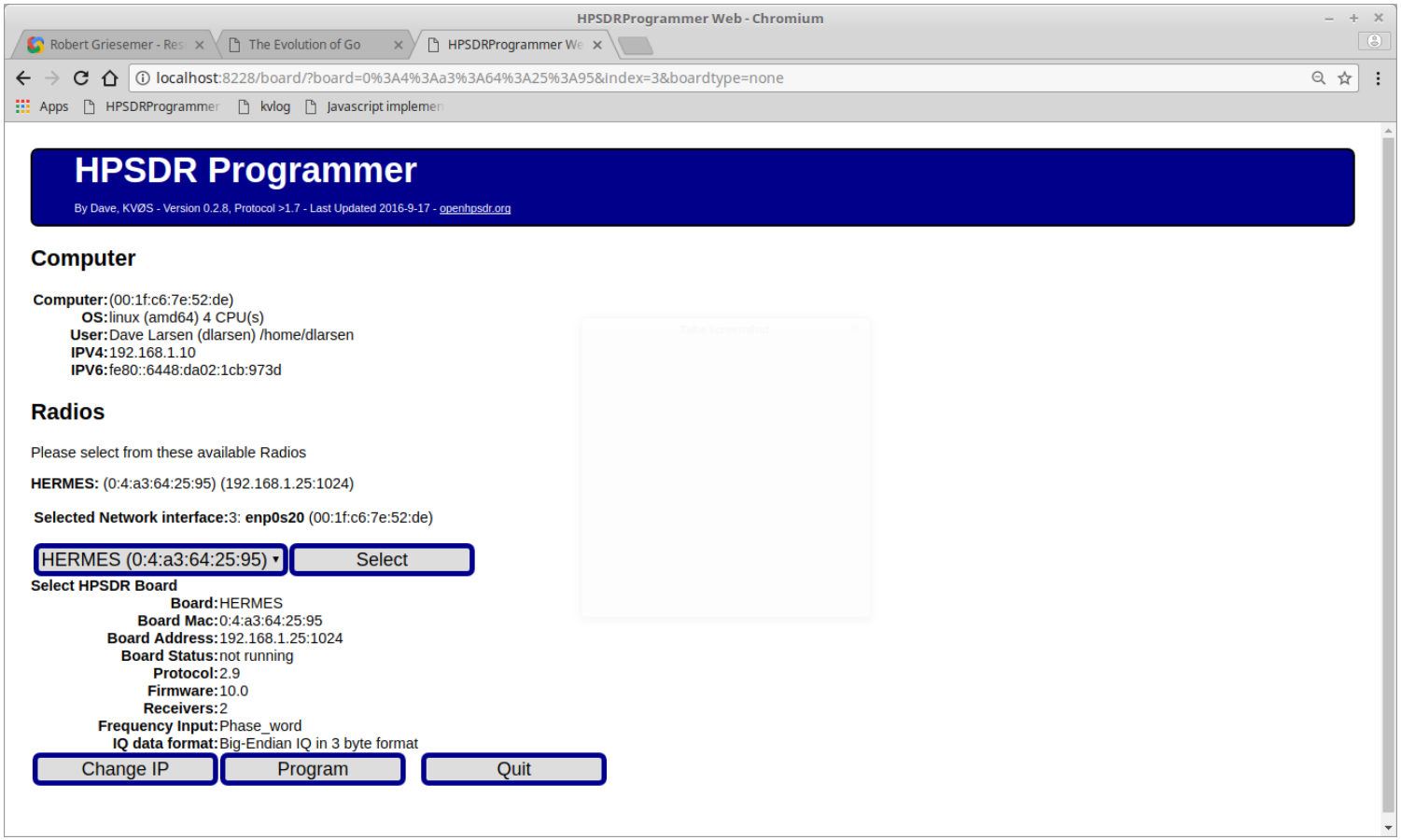
Local Web Programmer



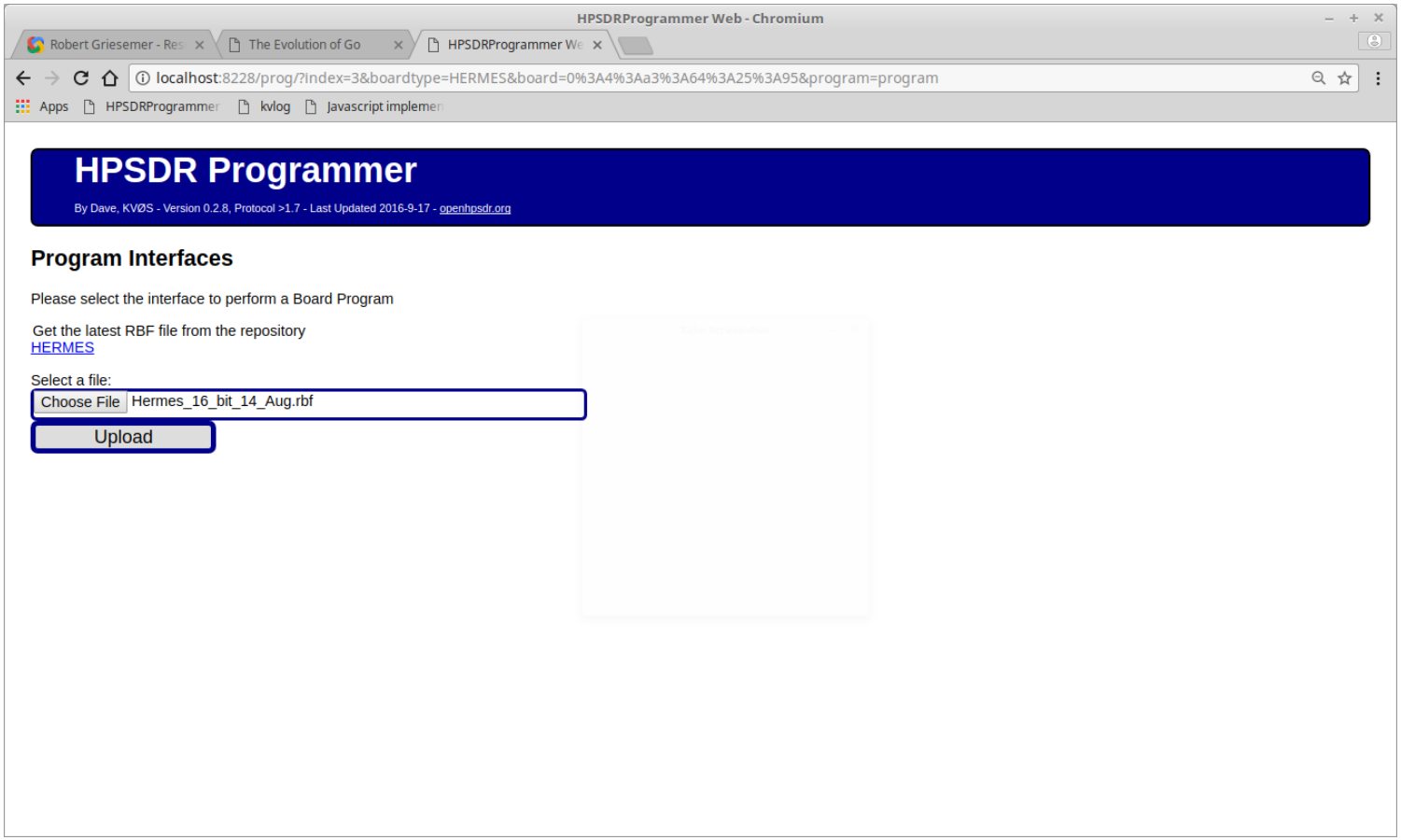
Local Web Programmer



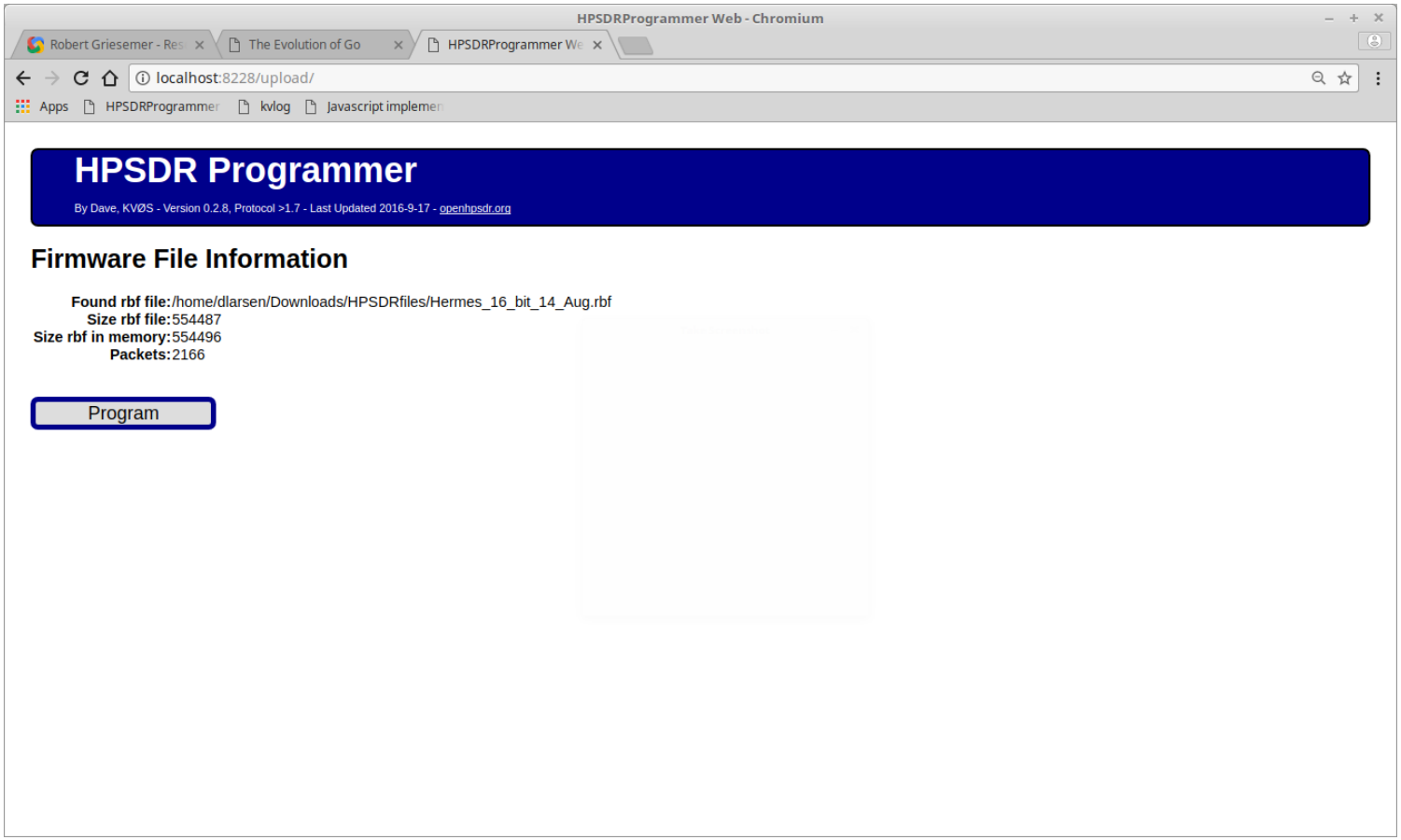
Local Web Programmer



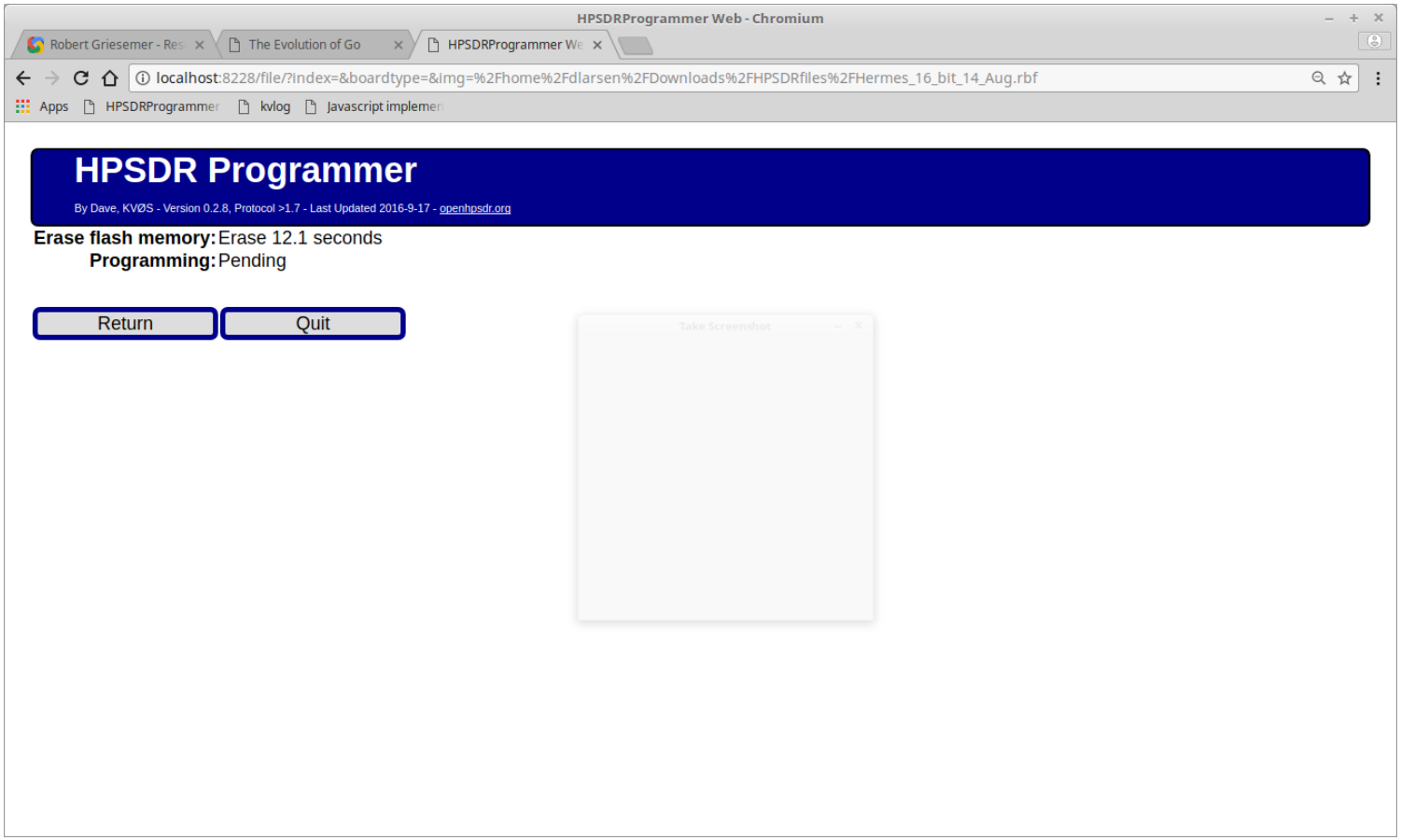
Local Web Programmer



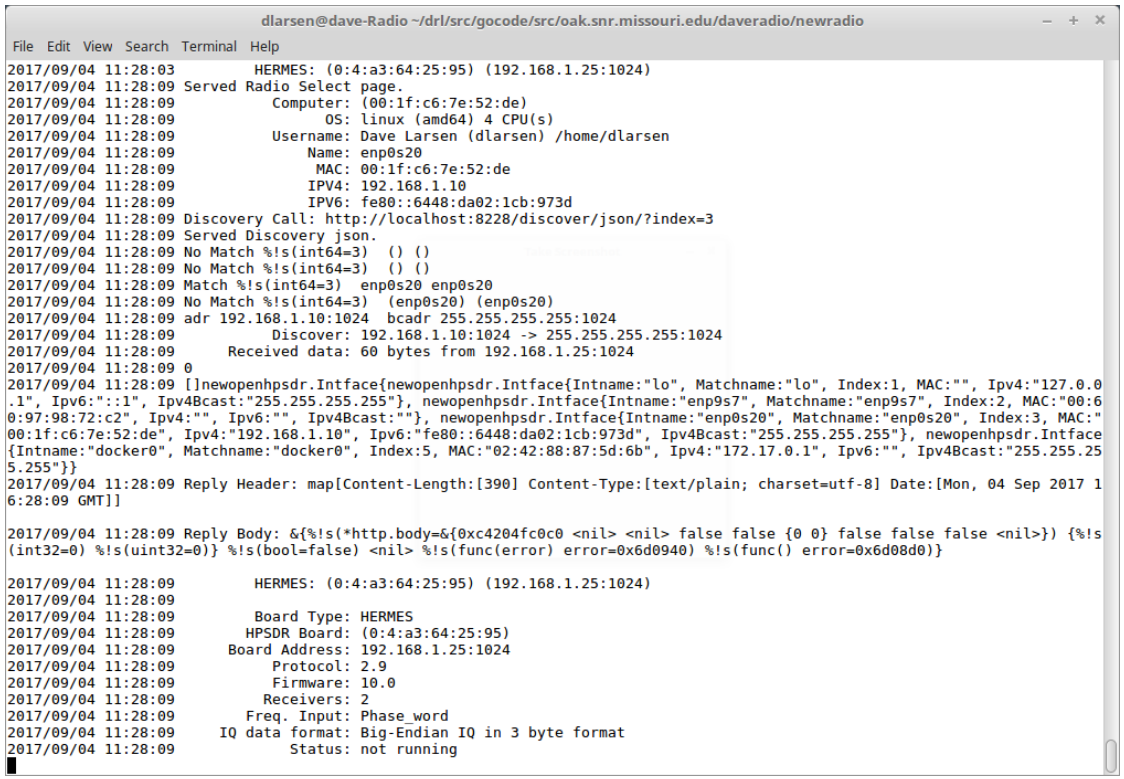
Local Web Programmer



Local Web Programmer



Local Web Programmer



Summary

- openhpsdr.org has a large number of new and old projects.
- Projects are both harware and software.
- Many people ask for changes. The bottle neck is the number of people to work on each item.
- Join us, Have fun making your favorite feature or part.

- Outlook
  - The HPSDR project is over 12 years old at this point.
  - I beleive that together we have made an impact on Amateur Radio.

For Further Reading I

- *Open High Performance Software Defined Radio*.
  http://openHPSDR.org
- *Open High Performance Software Defined Radio*.
  http://openHPSDR.org/beta/