

# Development and Design of Firmware Programming Tools for the openHPSDR Hardware

David R. Larsen<sup>1</sup>

<sup>1</sup>KV0S  
Columbia, Missouri, USA  
OpenHPSDR Development Group  
[openhpsdr.org](http://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.

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

The screenshot shows a web browser window with the address bar at `openhpsdr.org`. The page title is "High Performance Software Defined Radio - Chromium". The main heading is "High Performance Software Defined Radio" with the subtitle "An Open Source Design". A navigation menu includes: HOME, DOWNLOADS, DOCUMENTS, SUPPORT, WIKI, DISCUSSION LIST, TEAMSPEAK, RESOURCES, PROJECT OUTLINE, PUBLICATIONS, VIDEOS, MANUFACTURER LINKS, and DERIVATIVE PROJECTS. Below the menu is an "ARCHIVES" section.

**Project Description**

**Introduction – What's It All About?**

The HPSDR is an open source (GNU type) hardware and software project intended as a "next generation" Software Defined Radio (SDR) for use by Radio Amateurs ("hams") and Short Wave Listeners (SWLs). It is being designed and developed by a group of SDR enthusiasts with representation from interested experimenters worldwide.

The rationale behind the project is to break the overall design up into a number of modules. Each module is designed by an individual or group and connects to other modules using a pre-defined and common bus -- rather like plugging boards into a PC motherboard.

This modular approach enables prospective users to incorporate just the modules that interest them as well as designing their own variants if desired. The approach also enables new ideas and circuits to be tested by replacing an existing module. Since the majority of modules will be retained, such experimentation can be done with minimum disruption to an existing working system.

The modules vary in complexity from simple bandpass filters and input/output interfaces, to full blown DSP functions. Such variety enables experimenters with varying degrees of experience to contribute.

Thus far, the modules have each been named for easier identification when talking or writing about them. On this website, each module has its own web page, as noted by the tab selectors near the top of the page. Some of the modules are being designed so that they can be either used in conjunction with others or stand-alone. Each module board size (except the backplane) will be 100 mm. by 120 to 220 mm. and use either a 96 pin or 64 pin DIN41612 type connector.

**Components**

- HARDWARE
  - ATLAS - BACKPLANE
  - PRINOCCHIO - EXTENDER
  - JANUS - IQ SOUND
  - QZY - USB INTERFACE
  - MAGISTER - USB INTERFACE
  - MERCURY - RECEIVER

**Status**

Note **black titles** above are completed,  
**navy titles** are near completion,  
**DarkCyan titles** are proposed

**Updates**

**July 13, 2012** Hermes order page is open. See the TAPR page for details. Must order before July 25, 2012.

**March 10, 2012** Hermes interest list is open. See the TAPR page for details. To show your interest, you can sign up here: [www.tapr.com](http://www.tapr.com) (Note: Look under the tab after you log in)

**January 28, 2012** SVN

- 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.

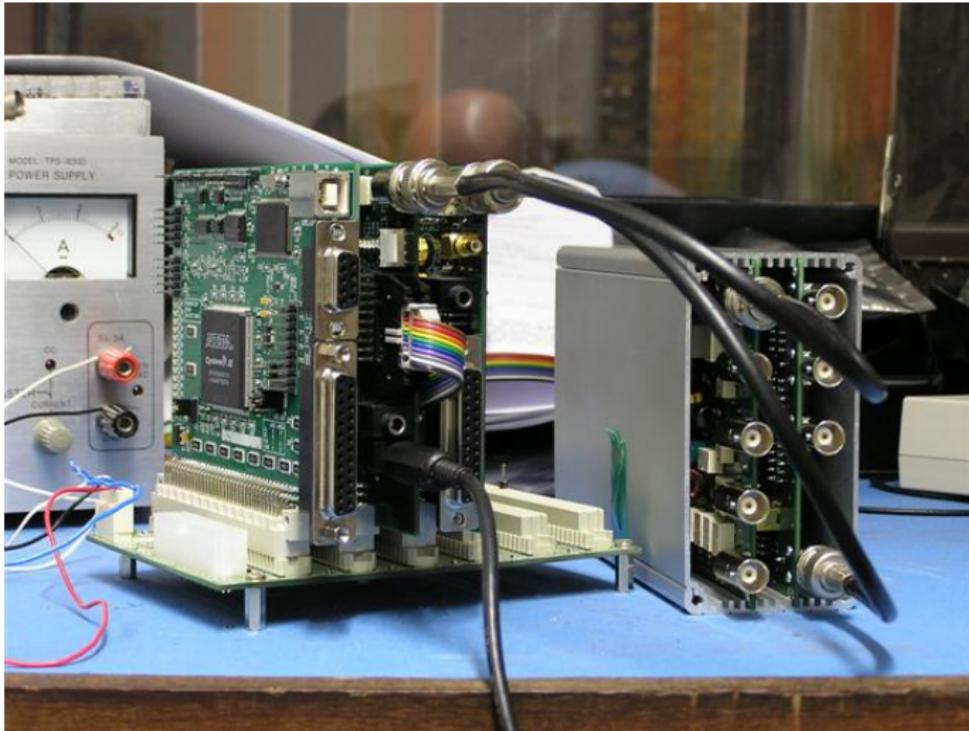
# Early days

- 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 Bandwidth

- 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 HPSPDRBootloader by John, G0ORX
  - Used PCAP
  - Written in C++ with Qt GUI
    - Required Administrator login.
    - Required setting jumpers on board.
    - Work flow was confusing to occasional 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

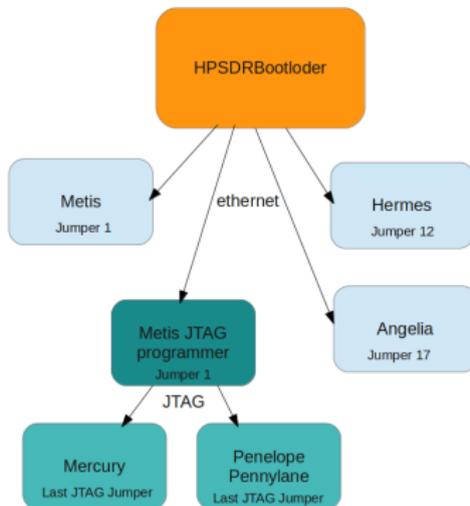
- HPDRBootloader is a program for recovery from programming failure.
- HPDRProgrammer is a program for normal updating.

# HPSDRBootloader

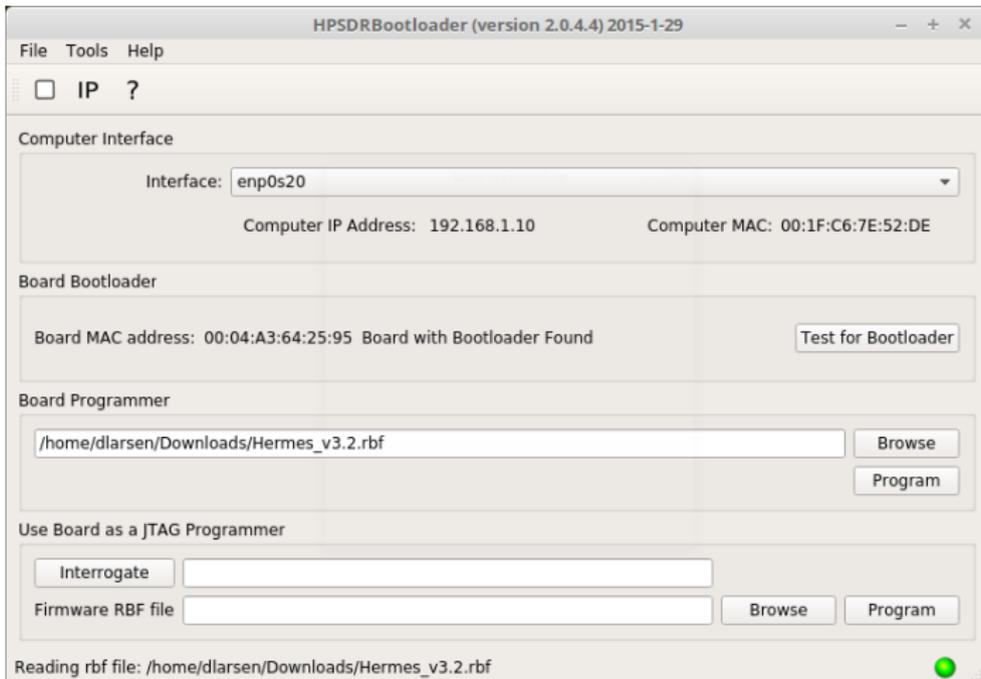
## Design Criteria

- Use PCAP protocol. (MAC Addressing)
- Require Administrator 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

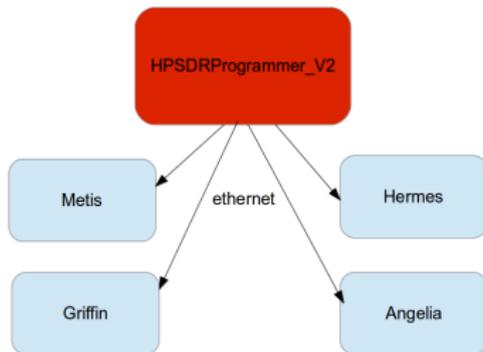


# HPSDRProgrammer

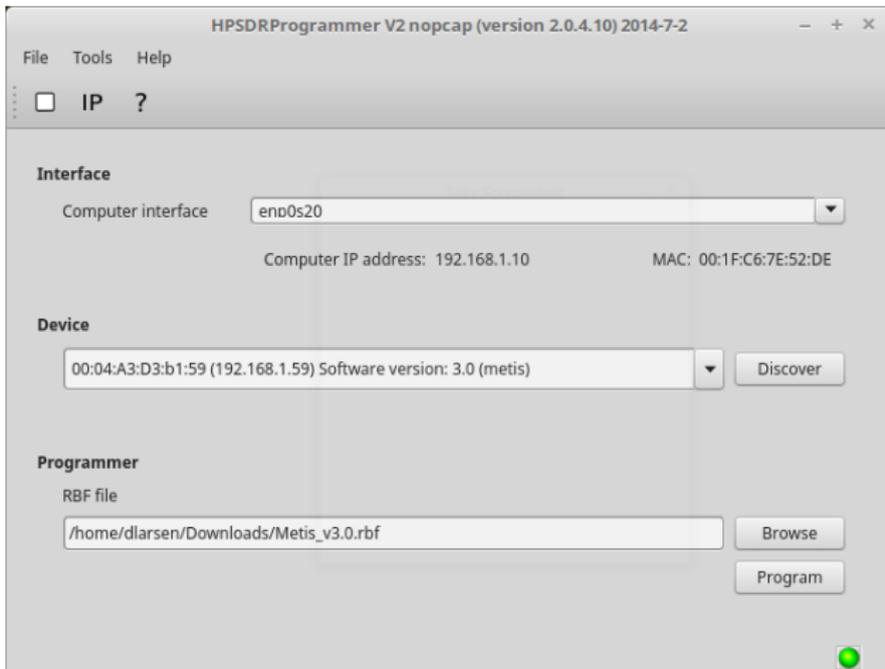
## Design Criteria

- 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 screenshot shows the Go Programming Language website in a Chromium browser. The browser's address bar displays "https://golang.org". The website's navigation bar includes links for "Documents", "Packages", "The Project", "Help", "Blog", and a search box. The main content area is divided into several sections:

- Try Go:** A code editor with a yellow background containing the following Go code:

```
// You can edit this code!  
// Click here and start typing.  
package main  
  
import "fmt"  
  
func main() {  
    fmt.Println("Hello, 世界")  
}
```

Below the code editor is a dropdown menu showing "Hello, World!" and three buttons: "Run", "Share", and "Tour".
- Go is an open source programming language that makes it easy to build simple, reliable, and efficient software.** This text is accompanied by the Go mascot, a cartoon bear.
- Download Go:** A button with the text "Binary distributions available for Linux, Mac OS X, Windows, and more."
- Featured video:** A video player showing a video titled "Andrew Gerrand - Go: a simple programming environment - Railsbery 2013" by Andrew Gerrand.
- Featured articles:** A section titled "Go 1.9 is released" with a sub-header "Go 1.9 is released". The text below reads: "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 Greisner

# 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. Communicating 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 monoproccessor.

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

```
dLarsen@dave-Radio ~/drl/src/gocode/src/oak.snr.missouri.edu/daveradio/radio
File Edit View Search Terminal Help
dLarsen@dave-Radio ~/drl/src/gocode/src/oak.snr.missouri.edu/daveradio/radio $ ./CndHPSDRProgrammer -interface-emp0s20
Computer: (00:1f:c6:7e:52:de)
OS: linux (amd64) 4 CPU(s)
IPv4: 192.168.1.10
Mask: [255 255 255 0]
Network: 192.168.1.0
IPv6:

HPSDR Board: (0:4:a3:d3:b1:59)
IPv4: 192.168.1.59
Port: 1024
Type: Metis
Firmware: 3.0
Status: not running
PC : 192.168.1.10:1024
dLarsen@dave-Radio ~/drl/src/gocode/src/oak.snr.missouri.edu/daveradio/radio $
```

## 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

```
darlarsen@dave-Radio ~/drl/src/gocode/src/oak.sn.missouri.edu/daveradio/newradio
File Edit View Search Terminal Help
dlarsen@dave-Radio ~/drl/src/gocode/src/oak.sn.missouri.edu/daveradio/newradio $ ./HPSDRProgrammer_cmd -index-3
2017/08/05 18:13:42 Computer: (00:1f:c6:7e:52:de)
2017/08/05 18:13:42 OS: linux (amd64) 4 CPU(s)
2017/08/05 18:13:42 Username: Dave Larsen (dlarsen) /home/dlarsen
2017/08/05 18:13:42 IPV4: 192.168.1.10
2017/08/05 18:13:42 IPV6: fe80::6448:da02:1cb:973d
2017/08/05 18:13:42 Discover: 192.168.1.18:0 -> 255.255.255.255:1024
2017/08/05 18:13:42 Received data: 60 bytes from 192.168.1.25:1024
2017/08/05 18:13:42 0
2017/08/05 18:13:42 Board Type: HERMES
2017/08/05 18:13:42 HPSDR Board: (8:4:a3:64:25:95)
2017/08/05 18:13:42 Board Address: 192.168.1.25:1024
2017/08/05 18:13:42 Protocol: 2.9
2017/08/05 18:13:42 Firmware: 16.0
2017/08/05 18:13:42 Receivers: 2
2017/08/05 18:13:42 Freq. Input: Phase_word
2017/08/05 18:13:42 IQ data format: Big-Endian IQ in 3 byte format
2017/08/05 18:13:42 Status: not running
dlarsen@dave-Radio ~/drl/src/gocode/src/oak.sn.missouri.edu/daveradio/newradio $
```

# Programmers in Go

- At this point I have two fast and clean packages to program HPSSDR boards.
- Shared these with others for testing.
- Most users we uncomfortable with the 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.

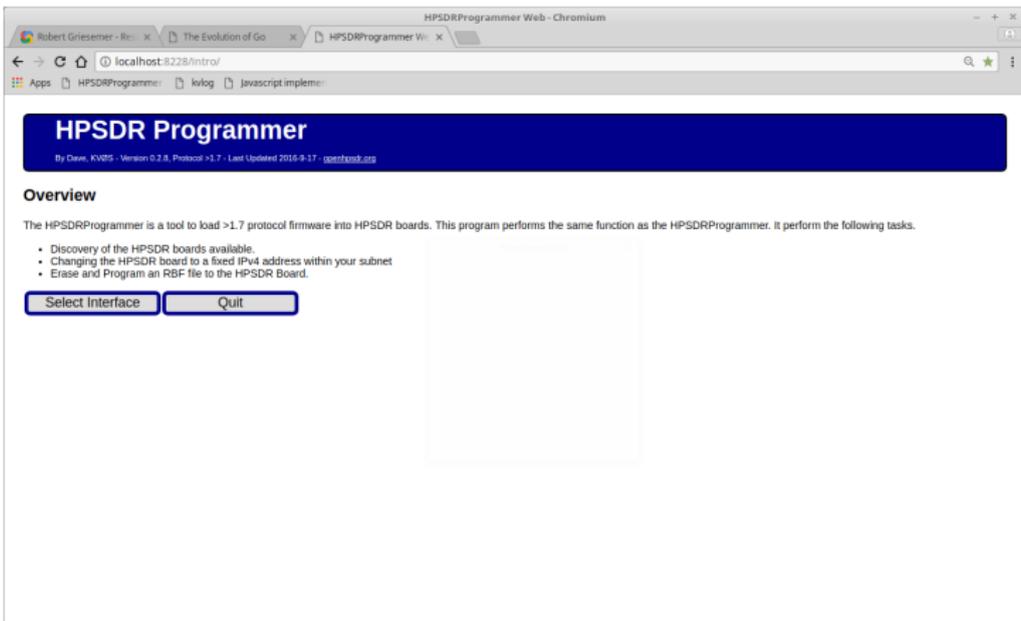
# Local Web Programmer

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

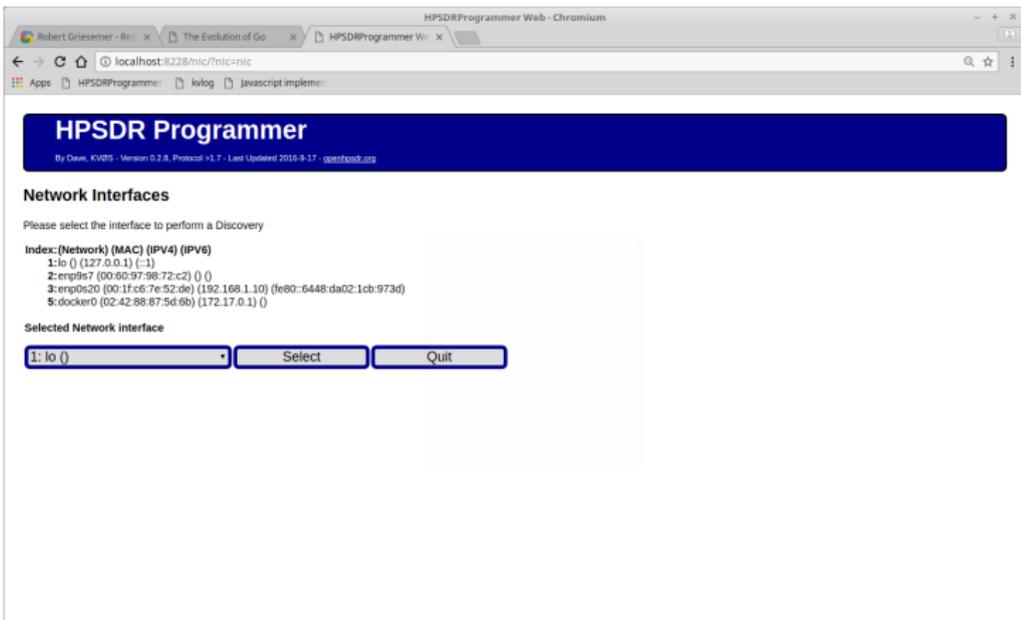
# Local Web Programmer

```
dlarsen@dave-Radio ~/drl/src/gocode/src/oak.snr.missouri.edu/daveradio/newradio
File Edit View Search Terminal Help
dlarsen@dave-Radio ~/drl/src/gocode/src/oak.snr.missouri.edu/daveradio/newradio $ ./HPSDRProgramer_web
2017/09/04 11:17:40 For a list of commands use --help
2017/09/04 11:17:40 RBF directory /home/dlarsen/Downloads/HPSDRfiles/
2017/09/04 11:17:40 Listening ...
2017/09/04 11:17:40 Point your web browser to: http://localhost:8228/intro/
```

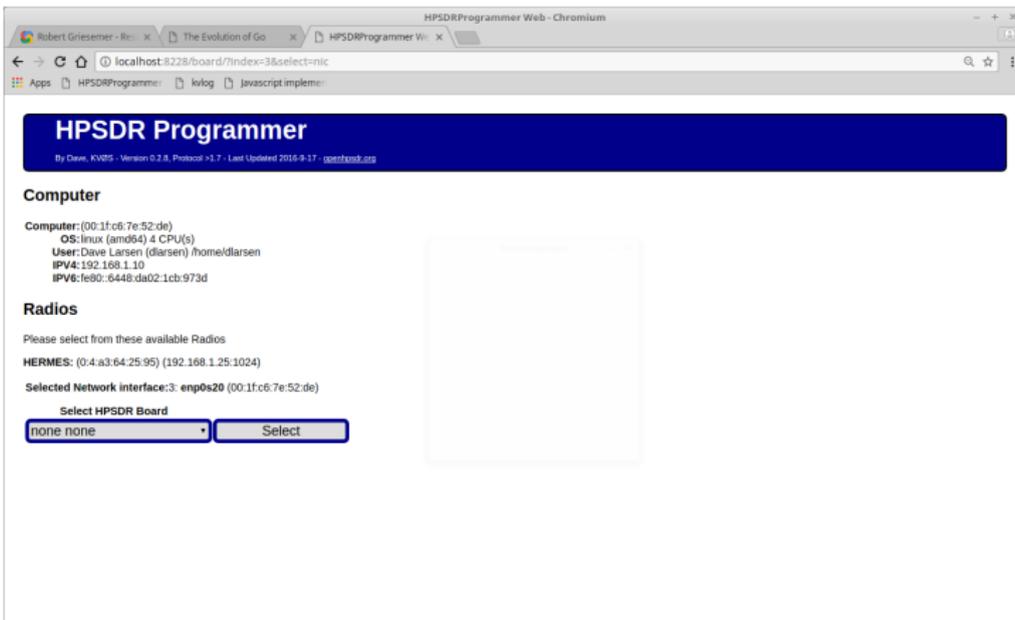
# Local Web Programmer



# Local Web Programmer



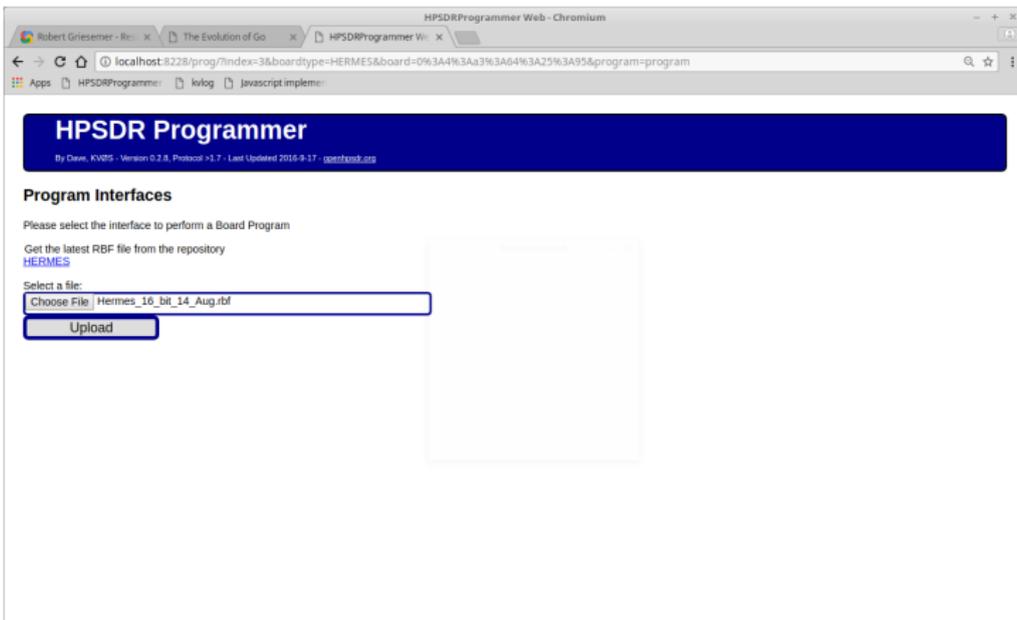
# Local Web Programmer



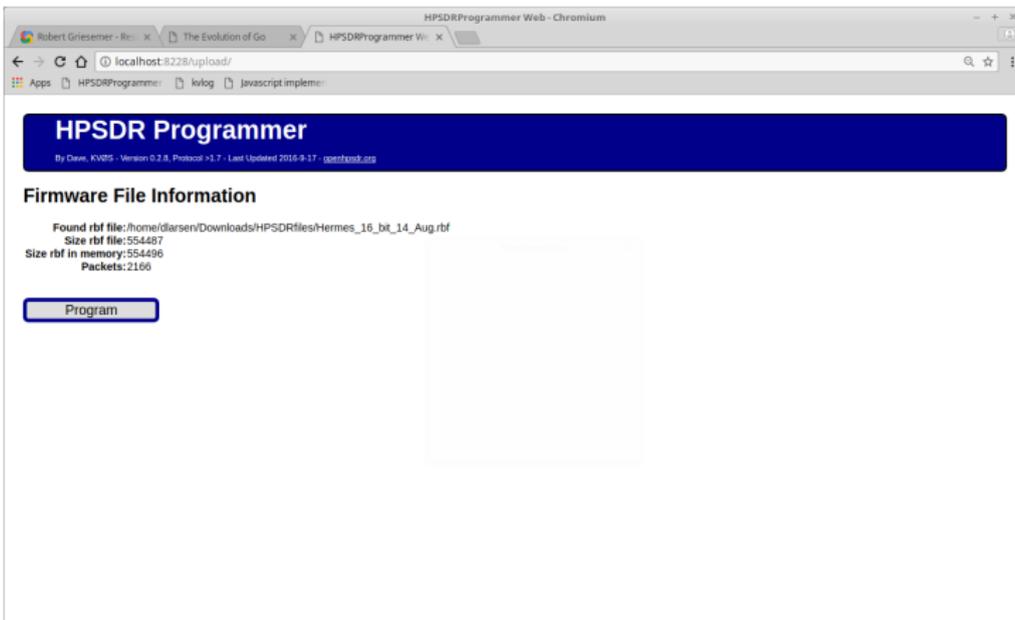
# Local Web Programmer

The screenshot shows a web browser window titled "HPSDRProgrammer Web - Chromium". The address bar shows the URL "localhost:8228/board/?board=0%3A4%3Aa3%3A04%3A25%3A95&index=3&boardtype=none". The page content includes a blue header with the text "HPSDR Programmer" and "By Dave, KV2S - Version 0.2.8, Protocol >1.7 - Last Updated 2016-9-17 - [gsatz@psk.org](mailto:gsatz@psk.org)". Below the header, there is a "Computer" section with details: "Computer: (00:1f:c6:7e:52:de)", "OS: linux (amd64) 4 CPU(s)", "User: Dave Larsen (dlarsen) /home/dlarsen", "IPv4: 192.168.1.10", and "IPv6: fe80::6448:da02:1cb:973d". A "Radios" section follows, with the instruction "Please select from these available Radios". It lists "HERMES: (0:4:a3:64:25:95) (192.168.1.25:1024)" and "Selected Network interface: 3: enp0s20 (00:1f:c6:7e:52:de)". A dropdown menu is set to "HERMES (0:4:a3:64:25:95)" with a "Select" button next to it. Below this, it says "Select HPSDR Board" and lists "Board: HERMES", "Board Mac: 0:4:a3:64:25:95", "Board Address: 192.168.1.25:1024", "Board Status: not running", "Protocol: 2.9", "Firmware: 10.0", and "Receivers: 2". At the bottom, there are three buttons: "Change IP", "Program", and "Quit".

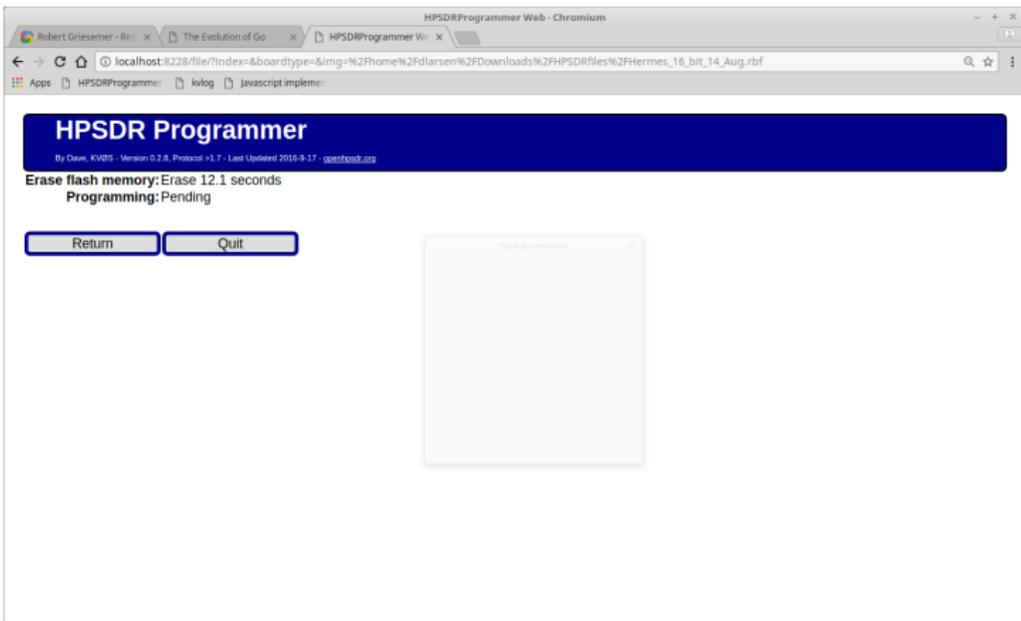
# Local Web Programmer



# Local Web Programmer



# Local Web Programmer



# Local Web Programmer

```

darsen@dave-Radio ~/dir/src/gocode/src/noak.snr.missouri.edu/daveradio/newradio
File Edit View Search Terminal Help
2017/09/04 11:28:03 HERMES: (0:4:a3:64:25:95) (192.168.1.25:1024)
2017/09/04 11:28:09 Served Radio Select page.
2017/09/04 11:28:09 Computer: (00:1f:c6:7e:52:de)
2017/09/04 11:28:09 OS: linux (amd64) 4 CPU(s)
2017/09/04 11:28:09 Username: Dave Larsen (darsen) /home/darsen
2017/09/04 11:28:09 Name: emp0s20
2017/09/04 11:28:09 MAC: 00:1f:c6:7e:52:de
2017/09/04 11:28:09 IPv4: 192.168.1.10
2017/09/04 11:28:09 IPv6: fe80::6448:da02:1cb:973d
2017/09/04 11:28:09 Discovery Call: http://localhost:8228/discover/json?index=3
2017/09/04 11:28:09 Served Discovery json.
2017/09/04 11:28:09 No Match %s(int64-3) {} {}
2017/09/04 11:28:09 No Match %s(int64-3) {} {}
2017/09/04 11:28:09 Match %s(int64-3) emp0s20 emp0s20
2017/09/04 11:28:09 No Match %s(int64-3) (emp0s20) (emp0s20)
2017/09/04 11:28:09 adr 192.168.1.10:1024 bcadr 255.255.255.255:1024
2017/09/04 11:28:09 Discover: 192.168.1.10:1024 -> 255.255.255.255:1024
2017/09/04 11:28:09 Received data: 60 bytes from 192.168.1.25:1024
2017/09/04 11:28:09 0
2017/09/04 11:28:09 [newopenhpsdr.Interface{newopenhpsdr.Interface{Intname:"lo", Matchname:"lo", Index:1, MAC:"", IpV4:"172.0.0.1", IpV6:"", IpV4BCast:"255.255.255.255"}, newopenhpsdr.Interface{Intname:"emp0s7", Matchname:"emp0s7", Index:2, MAC:"00:68:07:08:72:c2", IpV4:"", IpV6:"", IpV4BCast:""}, newopenhpsdr.Interface{Intname:"emp0s20", Matchname:"emp0s20", Index:3, MAC:"00:1f:c6:7e:52:de", IpV4:"192.168.1.10", IpV6:"fe80::6448:da02:1cb:973d", IpV4BCast:"255.255.255.255"}, newopenhpsdr.Interface{Intname:"docker0", Matchname:"docker0", Index:5, MAC:"02:42:88:87:5d:6b", IpV4:"172.17.0.1", IpV6:"", IpV4BCast:"255.255.255.255"}]}
2017/09/04 11:28:09 Reply Header: map[Content-Length:[390] Content-Type:[text/plain; charset=utf-8] Date:[Mon, 04 Sep 2017 16:28:09 GMT]]
2017/09/04 11:28:09 Reply Body: &{%s(*http.body={@xc4204fc0c @nil @nil false false {0 0} false false @nil}) %s(int32=0) %s(uint32=0)} %s(bool=false) <nil> %s(func(error) error=@x6d9940) %s(func() error=@x6d9940)}
2017/09/04 11:28:09 HERMES: (0:4:a3:64:25:95) (192.168.1.25:1024)
2017/09/04 11:28:09 Board Type: HERMES
2017/09/04 11:28:09 HPSDR Board: (0:4:a3:64:25:95)
2017/09/04 11:28:09 Board Address: 192.168.1.25:1024
2017/09/04 11:28:09 Protocol: 2.0
2017/09/04 11:28:09 Firmware: 10.0
2017/09/04 11:28:09 Receivers: 2
2017/09/04 11:28:09 Freq. Input: Phase_word
2017/09/04 11:28:09 IQ data format: Big-Endian IQ in 3 byte format
2017/09/04 11:28:09 Status: not running
  
```

# Summary

- [openhpsdr.org](http://openhpsdr.org) has a large number of new and old projects.
- Projects are both hardware 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 believe 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/>