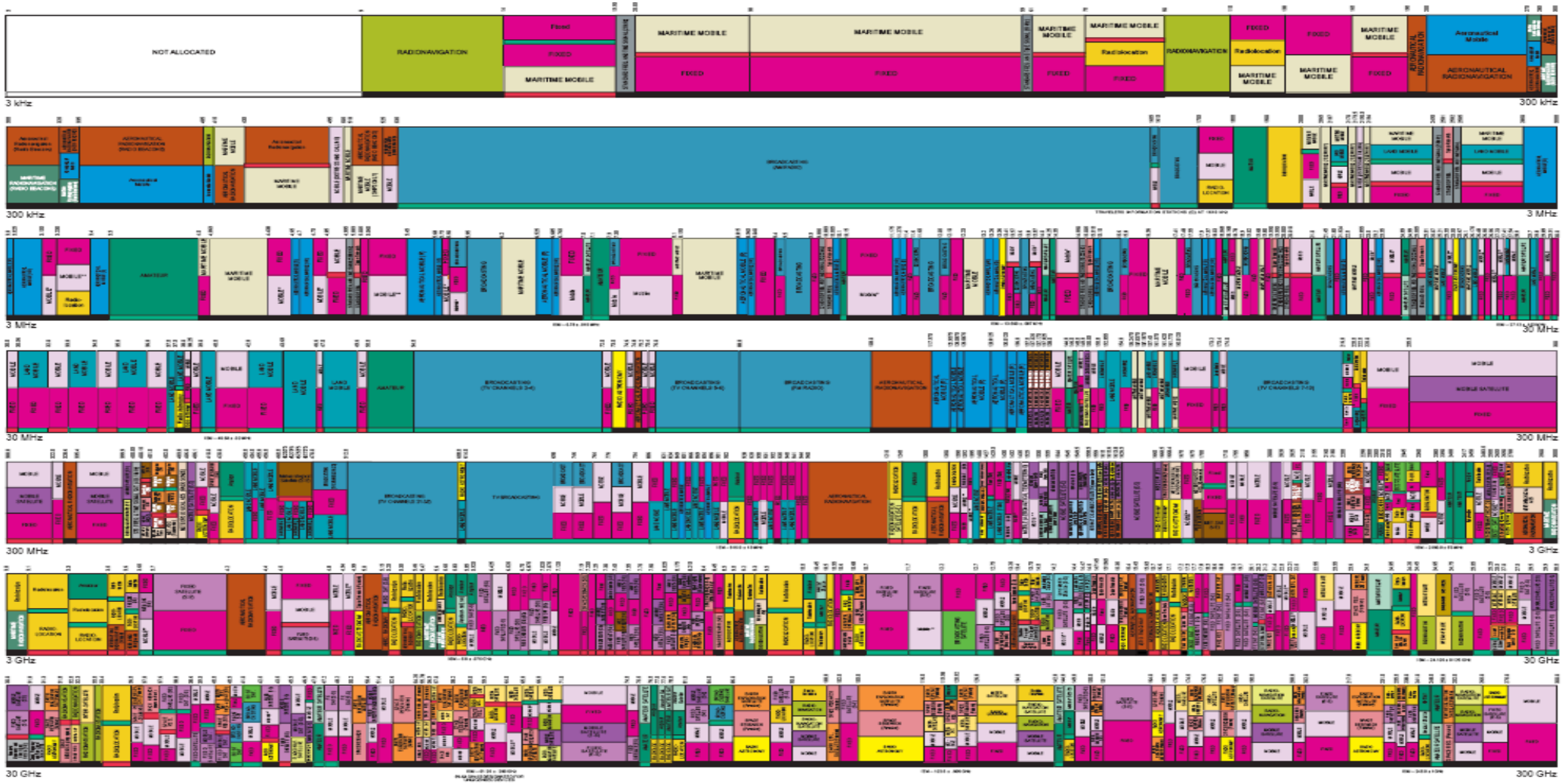
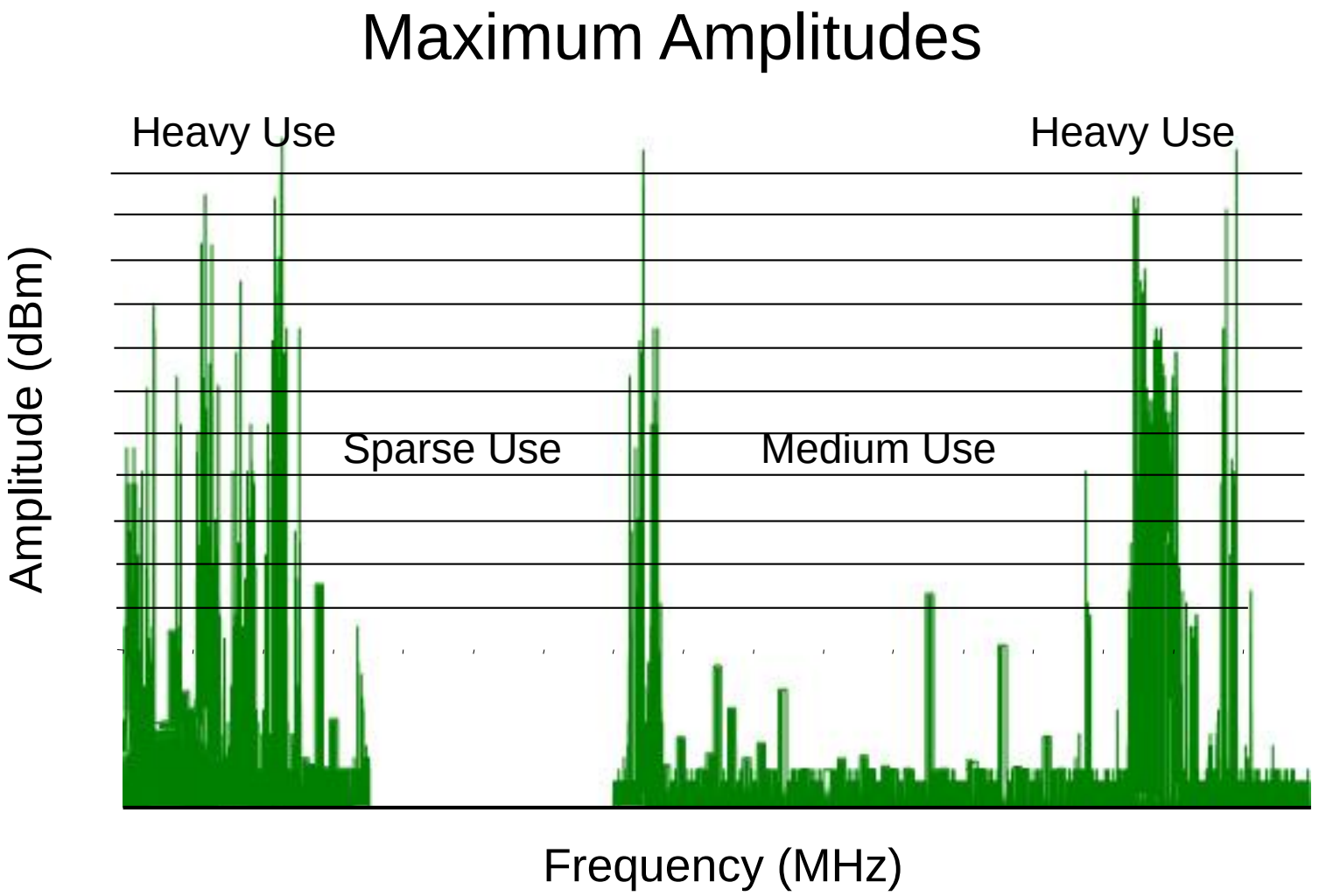
CHAPTER 1. INTRODUCTION

FIXED SPECTRUM ASSIGNMENT

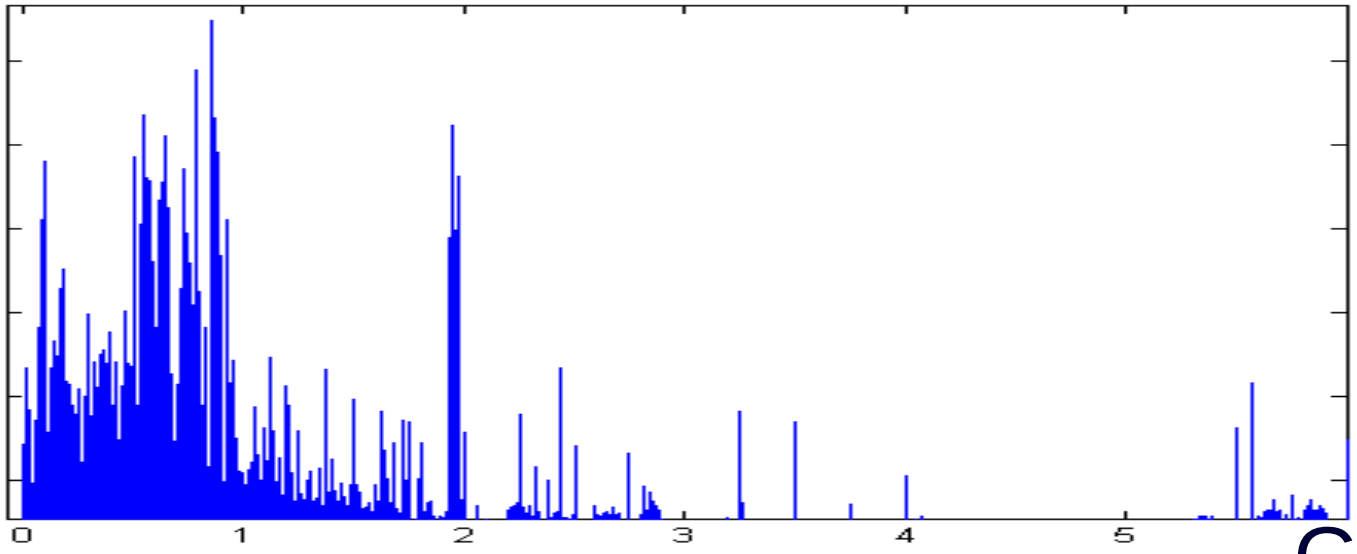


Fixed Spectrum Utilization



Fixed Spectrum Utilization

PSD (Power Spectrum Density)



Freq (GHz)	0~1	1~2	2~3	3~4	4~5	5~6
Utilization(%)	54.4	35.1	7.6	0.25	0.128	4.6

Measurements show that there is wide range of spectrum utilizations across 6 GHz of spectrum

Problems of Fixed Spectrum Utilization

Spectrum usage is concentrated on certain portions of the spectrum

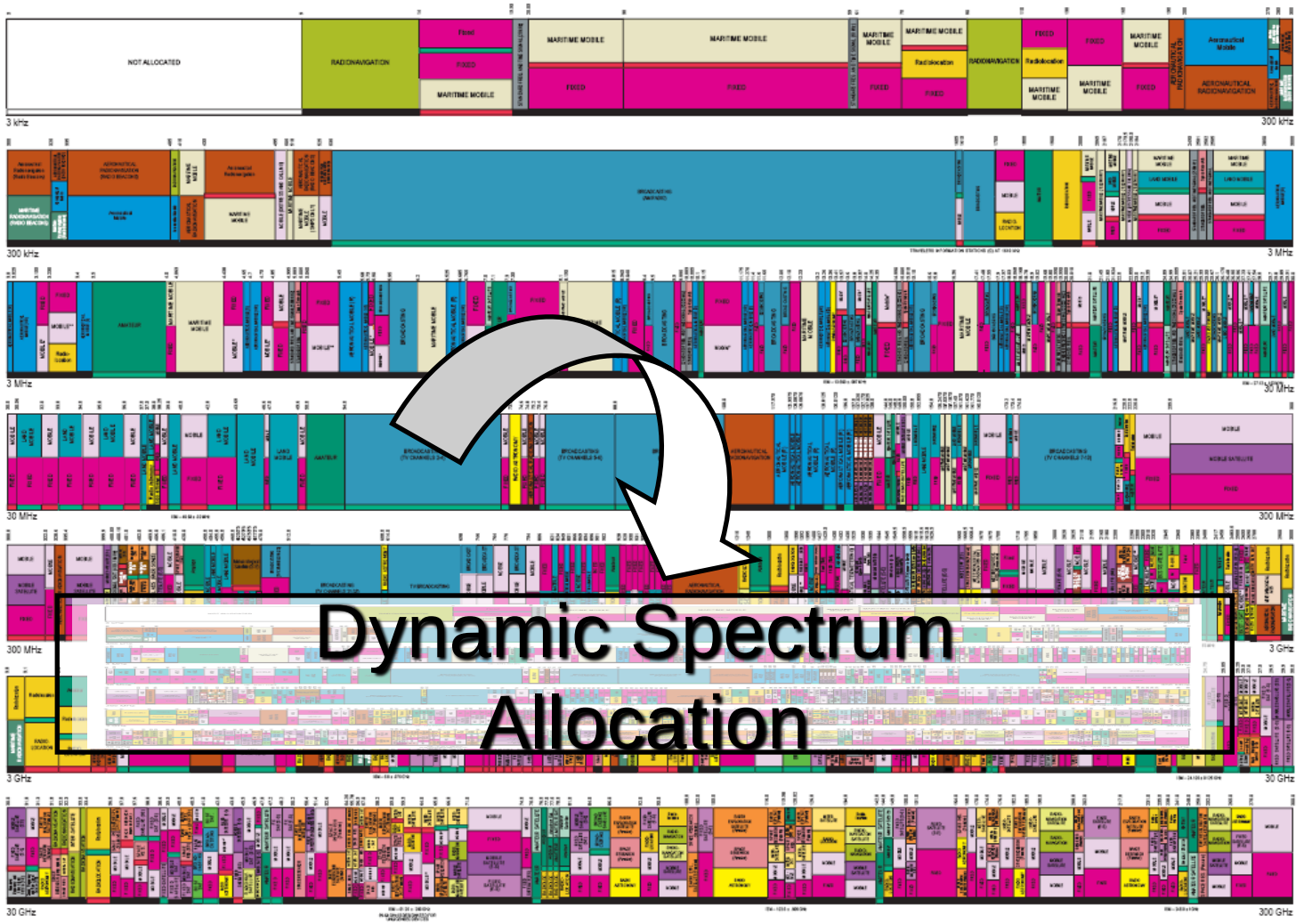
A significant amount of the spectrum remains unutilized.

According to FCC (Federal Communication Commission):

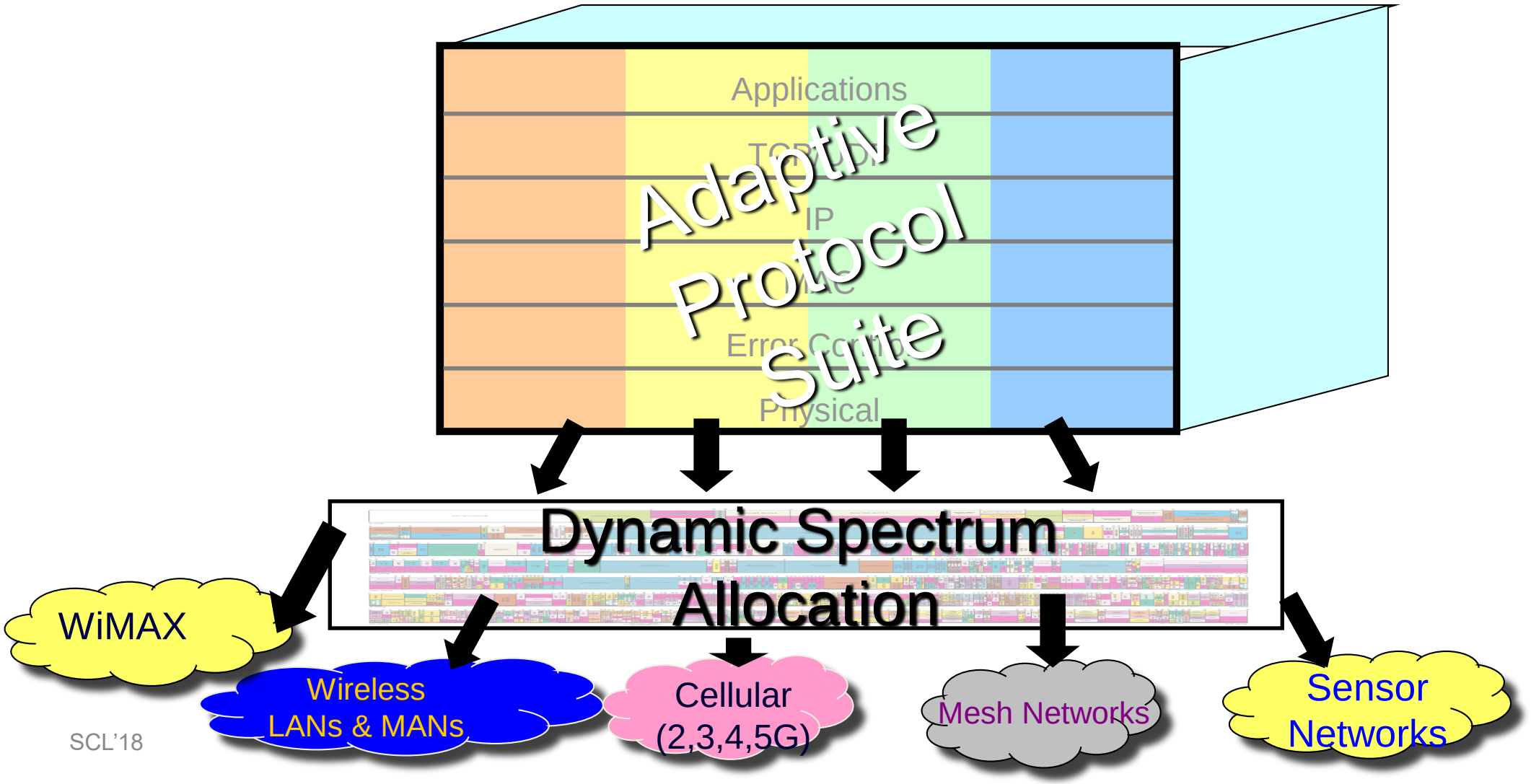
Utilization of the fixed spectrum assignment is approx.
15-85% based on temporal and geographical variations

→ Limited Available Spectrum and Inefficient Spectrum Usage!

COGNITIVE RADIO NETWORKS; DYNAMIC SPECTRUM ALLOCATION NETWORKS (DSANs); xG INITIATIVE



OVERALL VIEW



COGNITIVE RADIO

A "*Cognitive Radio*" is the key enabling technology for Dynamic Spectrum Access!!

Capability to use or share the spectrum in an opportunistic manner.

"BANDWIDTH HARVESTING"

Dynamic spectrum access techniques allow the CR to operate in the best available channel.

SPECTRUM MANAGEMENT FRAMEWORK

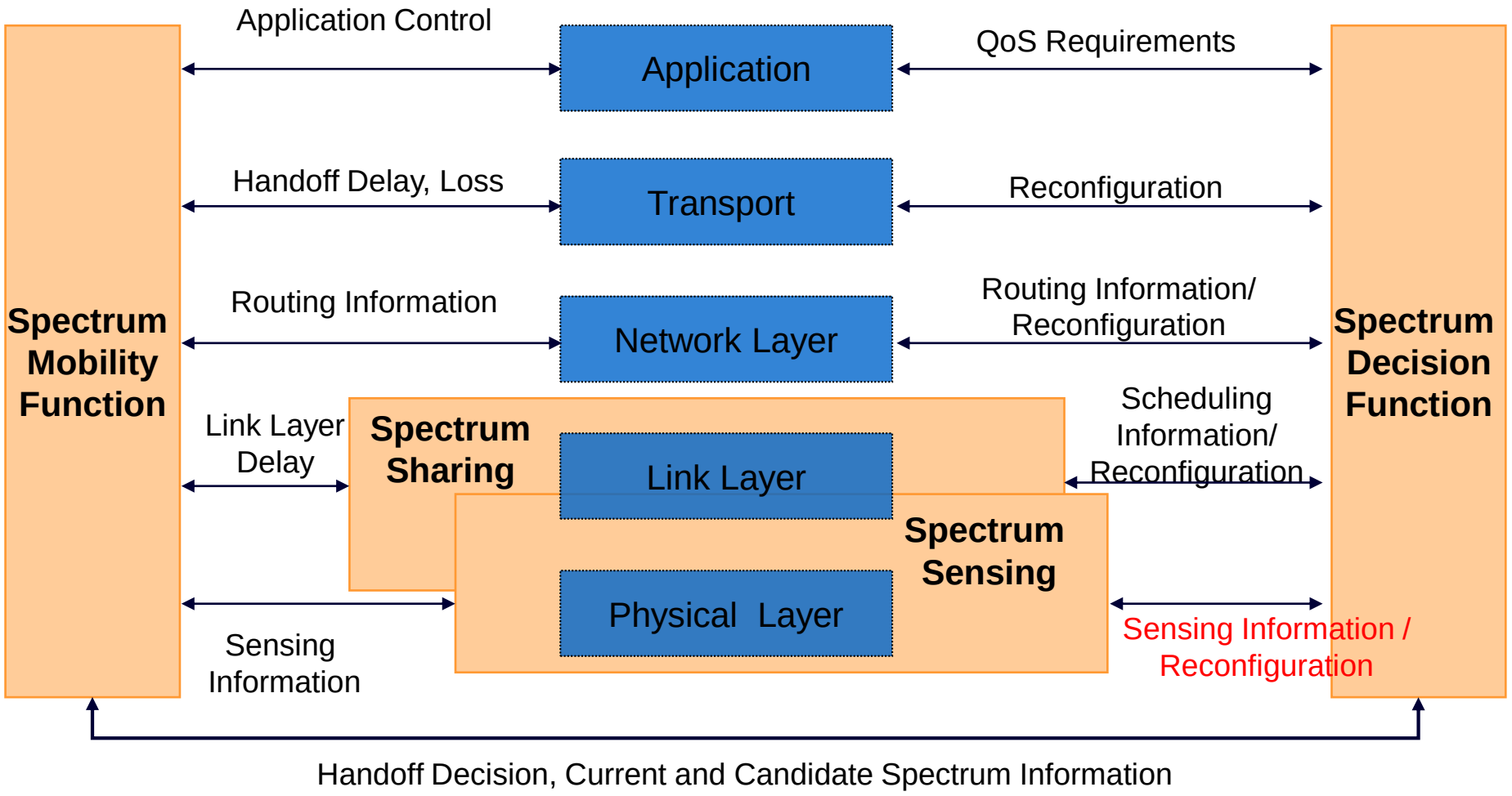
1) Determine which portions of the spectrum is available and detect the presence of licensed users when a user operates in a licensed band *(Spectrum Sensing)*

2) Select the best available channel *(Spectrum Decision)*

3) Coordinate access to this channel with other users *(Spectrum Sharing)*

4) Vacate the channel when a licensed user is detected

CRN COMMUNICATION FUNCTIONALITIES



BUILDING BLOCK: SOFTWATE DEFINED RADIO

SDRs: perform the majority of signal processing in the **digital domain** using programmable DSPs and hardware support

- Some signal processing is still done in the **analog domain**, such as in the RF and IF circuits

- Get the software **close to the antenna**
- Software **defines** the waveforms
- **Replace analog** signal processing **with digital** signal processing

Some of the slides are from presentations by Eric Bloom and John Ackermann. Thank you.

WHY SDR?

- Flexible
- Reduced Obsolescence
- Enhances Experimentation
- Brings Analog and Digital World Together

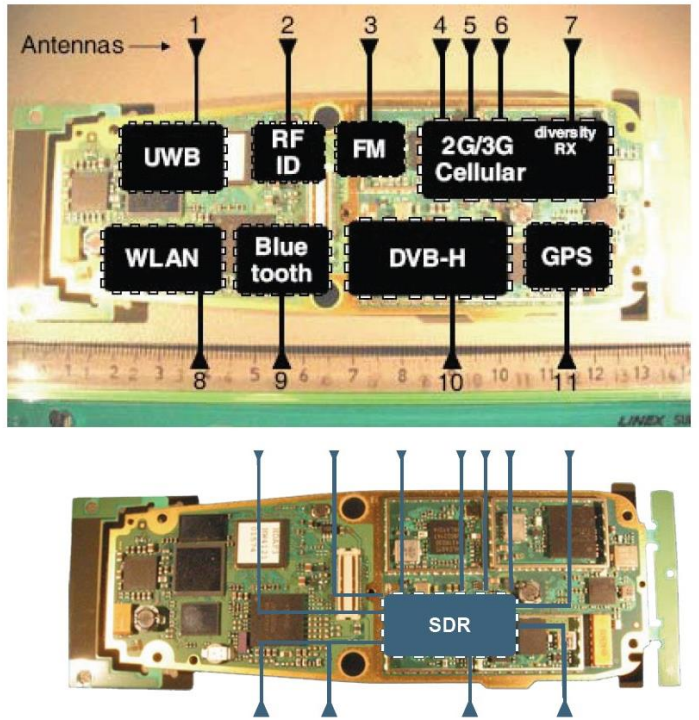
- Quicker time to market
- Multiple personalities (chameleon)
- New things are possible:
  - Multiple channels at the same time
  - Better spectrum utilization
  - “Cognitive Radios”

New Breed of Radio

- Reprogrammable
- Multiband/Multimode
- Networkable
- Simultaneous voice, data, video
- Full convergence of digital networks and radio science

Yrjö Neuvo, CTO, Nokia at ISSCC2004

Large number of parallel radio is a blocking factor for size and cost



Software-Defined Radio Approach:

→ Cover several standards with one flexible/programmable chip

imec *Dirk Hamelinck ; IMEC 2007*

Disadvantages

- Higher power consumption than dedicated ASIC approach
- More MIPS required
- Higher cost (today), but potential for lower cost in the future
- Multi-band operation limited by antennas

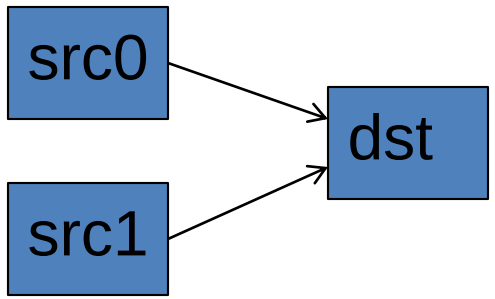
SOFTWARE ARCHITECTURE: GNU RADIO

- GNU Radio: a library of signal processing blocks & the glue to tie it all together
- The programmer builds a radio by creating a graph (as in graph theory)
  - Vertices: signal processing blocks
  - Edges represent the data flow between them
- The signal processing blocks are implemented in C++
  - Conceptually, blocks process infinite streams of data flowing (input ports to output ports)
  - Blocks' attributes: the number of input and output ports as well as the type of data
  - The most frequently used types are short, float and complex
- Some blocks have only output ports or input ports (as data sources and sinks)
  - There are sources that read from a file or ADC, and sinks that write to a file, DAC or graphical display
  - About 100 blocks come with GNU Radio
  - Writing new blocks is not very difficult

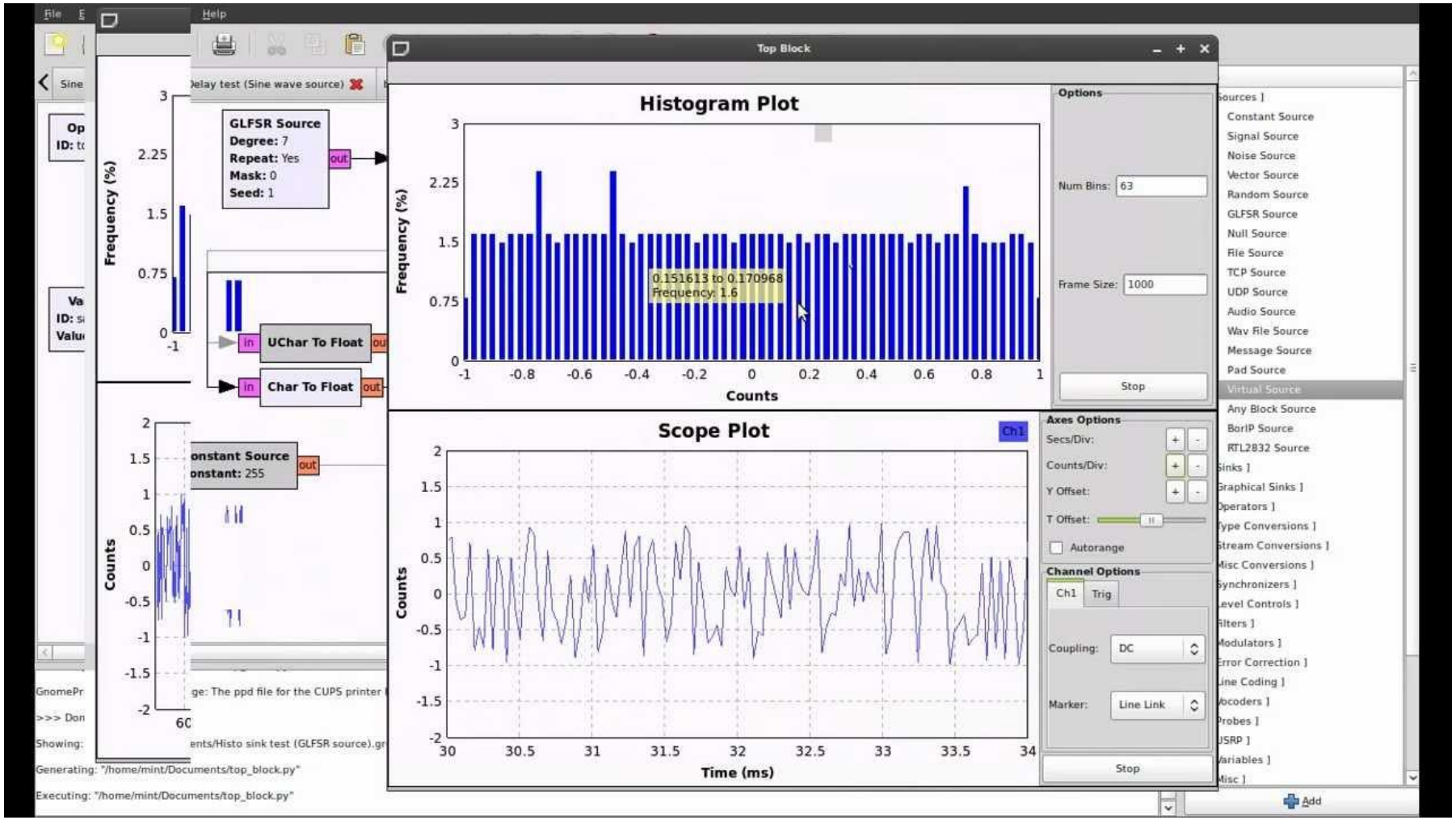
Dial tone generator

Dial Tone Output

```
#!/usr/bin/env python
from gnuradio import gr
from gnuradio import audio
def build_graph ():
  sampling_freq = 48000
  ampl = 0.1
  fg = gr.flow_graph ()
  src0 = gr.sig_source_f (sampling_freq, gr.GR_SIN_WAVE, 350, ampl)
  src1 = gr.sig_source_f (sampling_freq, gr.GR_SIN_WAVE, 440, ampl)

  dst = audio.sink (sampling_freq)

  fg.connect ((src0, 0), (dst, 0))
  fg.connect ((src1, 0), (dst, 1))
return fg

if __name__ == '__main__':
  fg = build_graph ()
  fg.start ()
  raw_input ('Press Enter to quit: ')

fg.stop ()
```

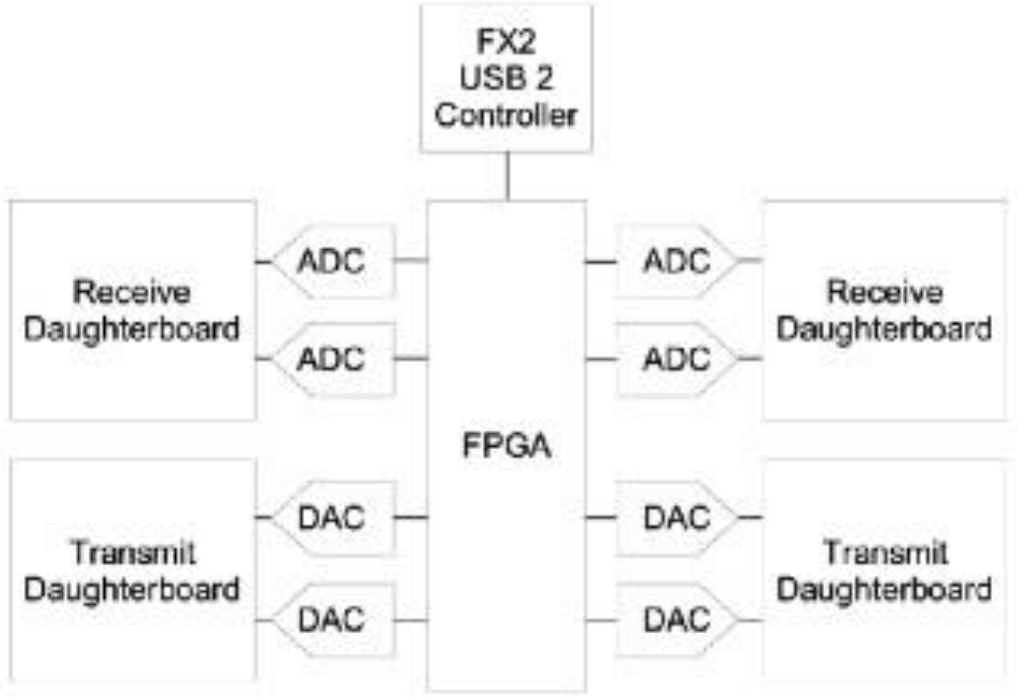


GNU Radio now has a nice GUI

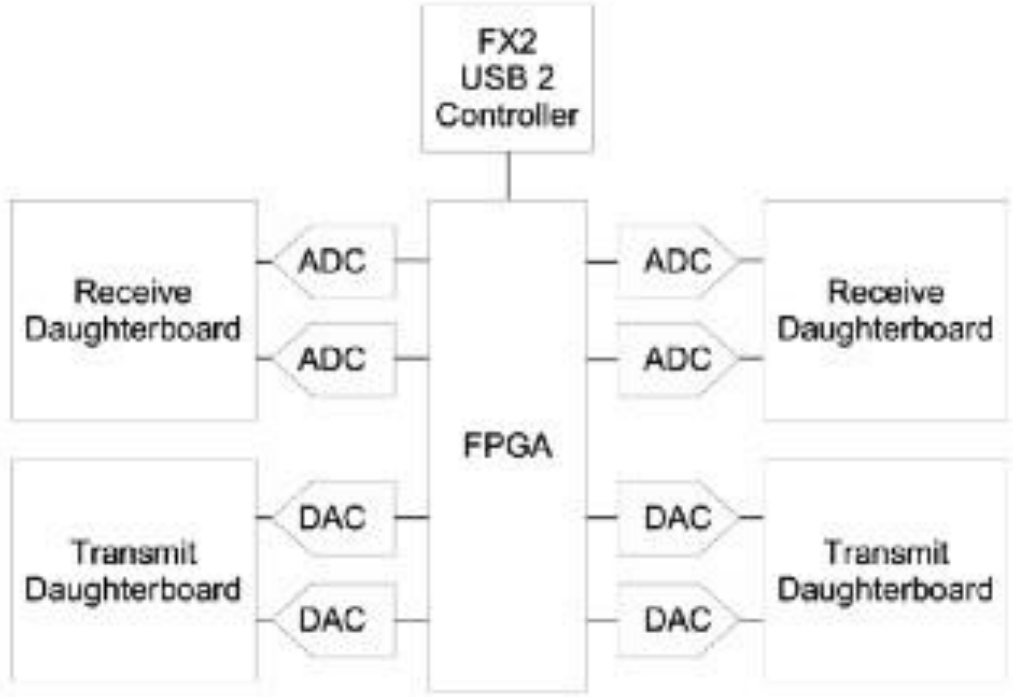


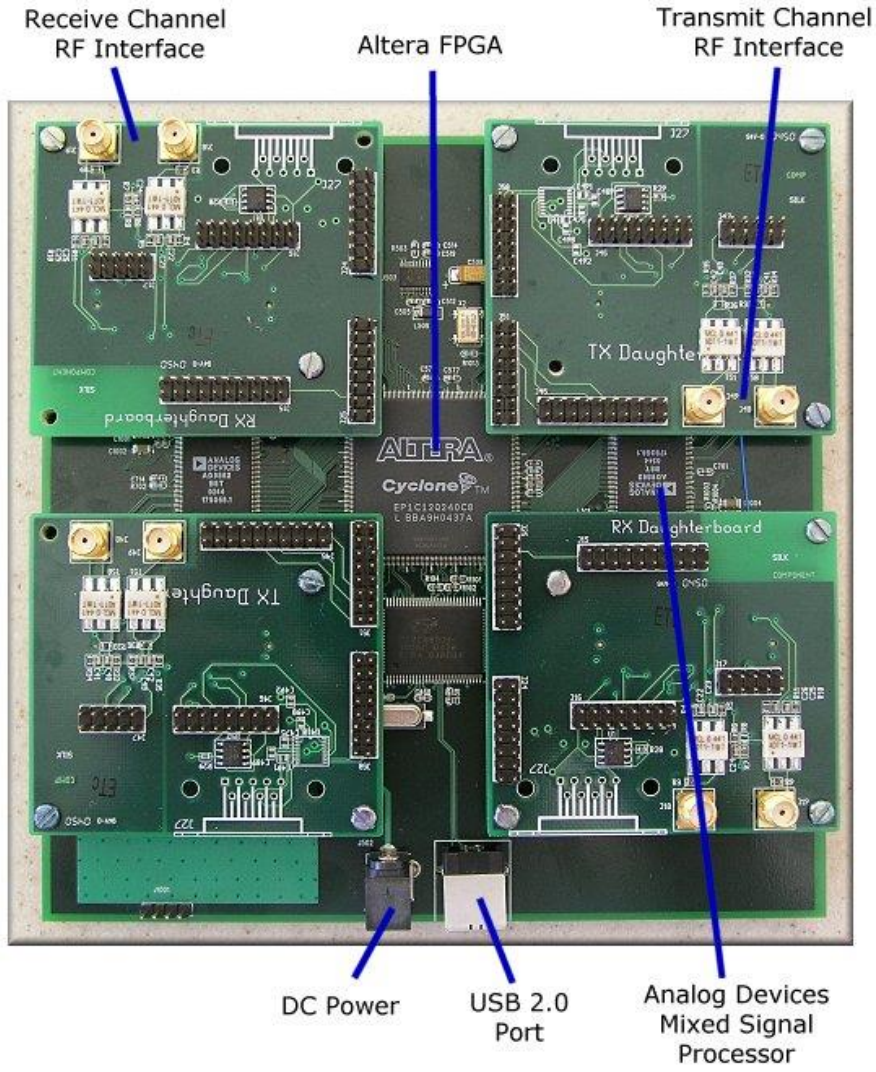
UNIVERSAL SOFTWARE RADIO PERIPHERAL

- USB 2.0 interface
- 4 High-Speed AD Converters (64 MS/s, 12-bit Analog Devices AD9862)
- 4 High-Speed DA Converters (128 MS/s, 14-bit) to generate signals up to about 50 MHz (same chip as above)
- Altera EP1C12 Q240C8 "Cyclone" FPGA for high bandwidth math, and to reduce the data rates to something you can squirt over USB2
- Lots of digital, serial, and low speed analog IO for controlling daughterboards
- 2 Daughterboard slots to hold any RF receiver interface or tuner
- 2 Daughterboard slots to hold any RF transmitter



Universal Software Radio Peripheral (USRP)





GNU Radio resources

- Home page (links to source code)
  - http://www.gnu.org/software/gnuradio
- Mailing list
  - discuss-gnuradio-request@gnu.org
- Archive
  - http://mail.gnu.org/mailman/listinfo/discuss-gnuradio

- Open source hardware
  - http://www.opencores.org/projects/pci
  - PCI bridges, ethernet, memory controllers, etc.