AIP LAB

A MIMO Modeling Framework Using a Software Defined Radio Paradigm

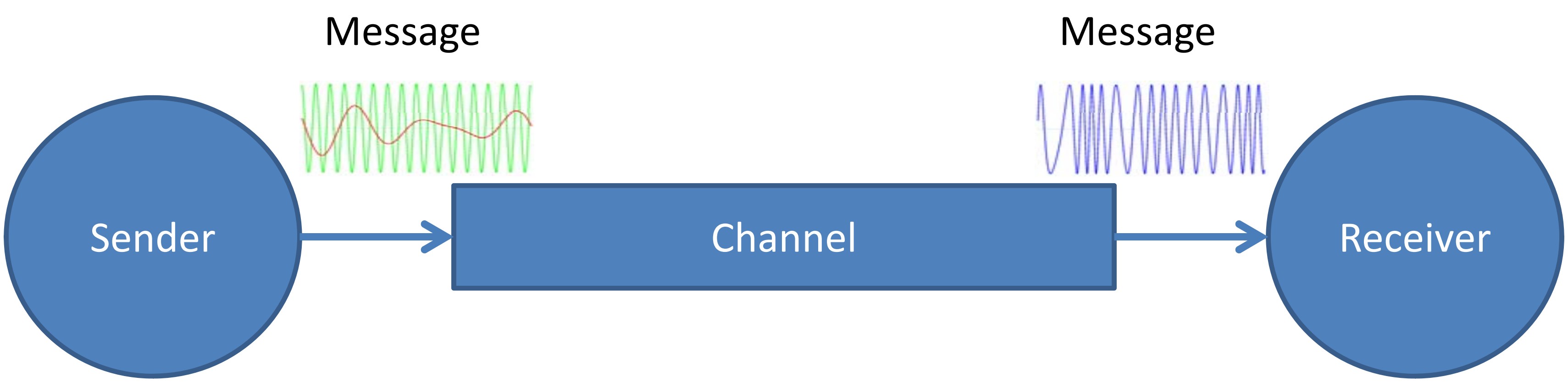
Angel Camelo Vasquez
Advisor: Dr. Domingo Rodriguez
University of Puerto Rico
July 2012

Edited by D. Rodriguez – Sept. 2012

Outline

- Motivation
- Objectives
- Background
- Methodology
- Design
- Implementation
- Testing
- Conclusions

Motivation



Motivation

- Multiple Input Multiple Output (MIMO): use of multiple antennas, both, at transmitter and receiver.
  - MIMO is a trending technology into the signal communications realm.
  - Actual implementations with 3GPP, WiMAX, WiFi.
- Software Defined Radio
  - Radio implemented by means of software.
- Time-frequency Representations

Objectives

- Design, implement, and test a framework for MIMO channel monitoring and simulation
  - Redesign the existent **SI**gnal **R**epresentation **LAB**oratory application software named **SIRLAB.**
  - Implement time-frequency Representations.
  - Provide a MIMO simulation.

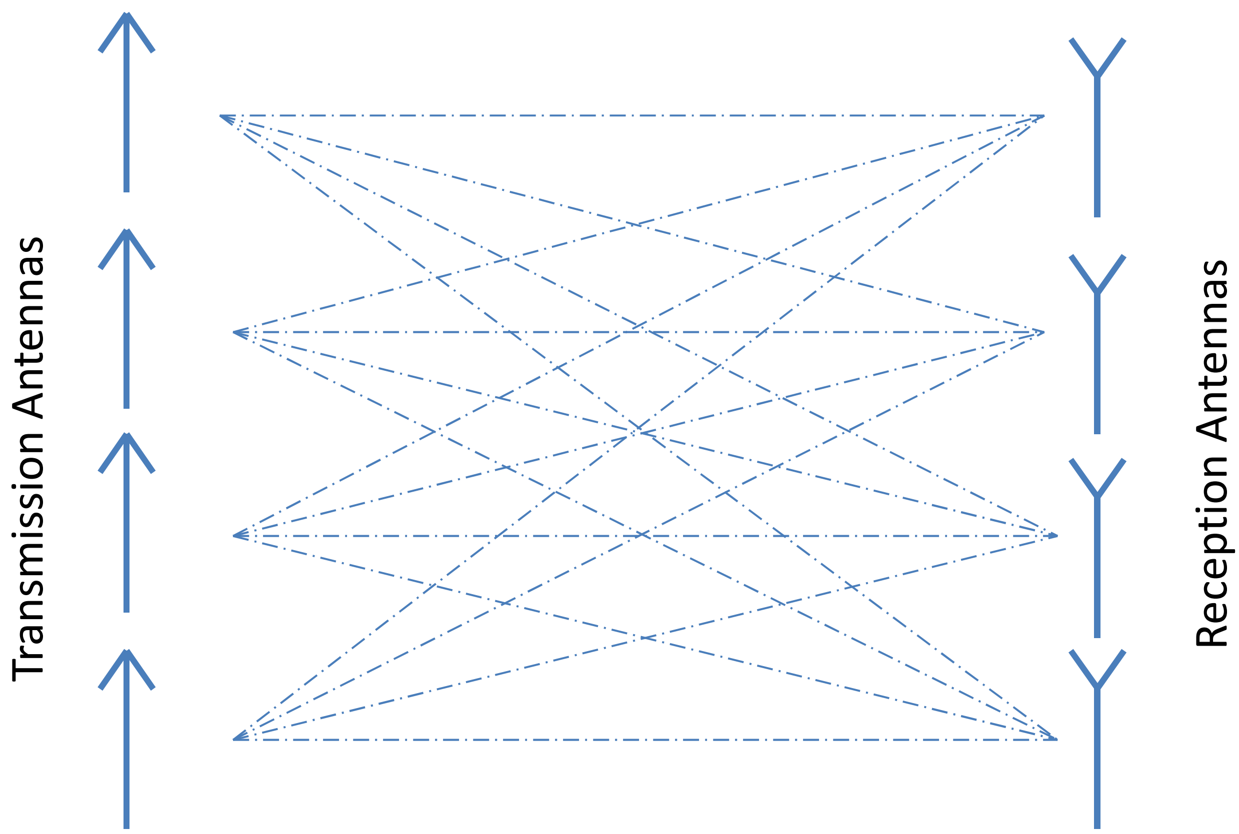
Background

- Communications
- Software Defined Radio
- Time-Frequency Representations

Communications

- Communications
  - Send multiple symbols with different antennas.
  - The signal is modified by the channel and possibly reflected by several scatterers.
  - Multiple copies of the signal may be received and processed.

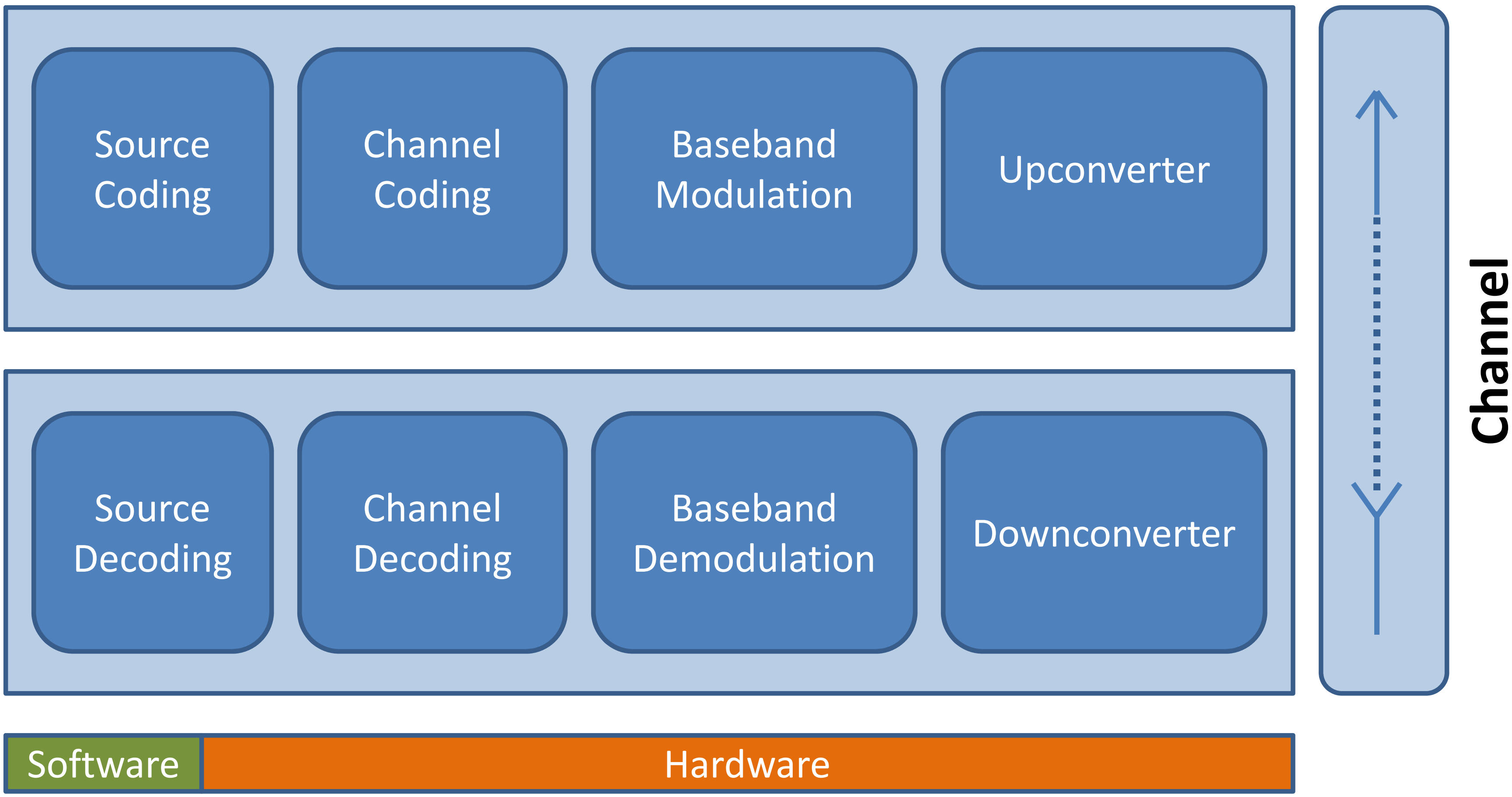
Communications Channel



Software Defined Radio

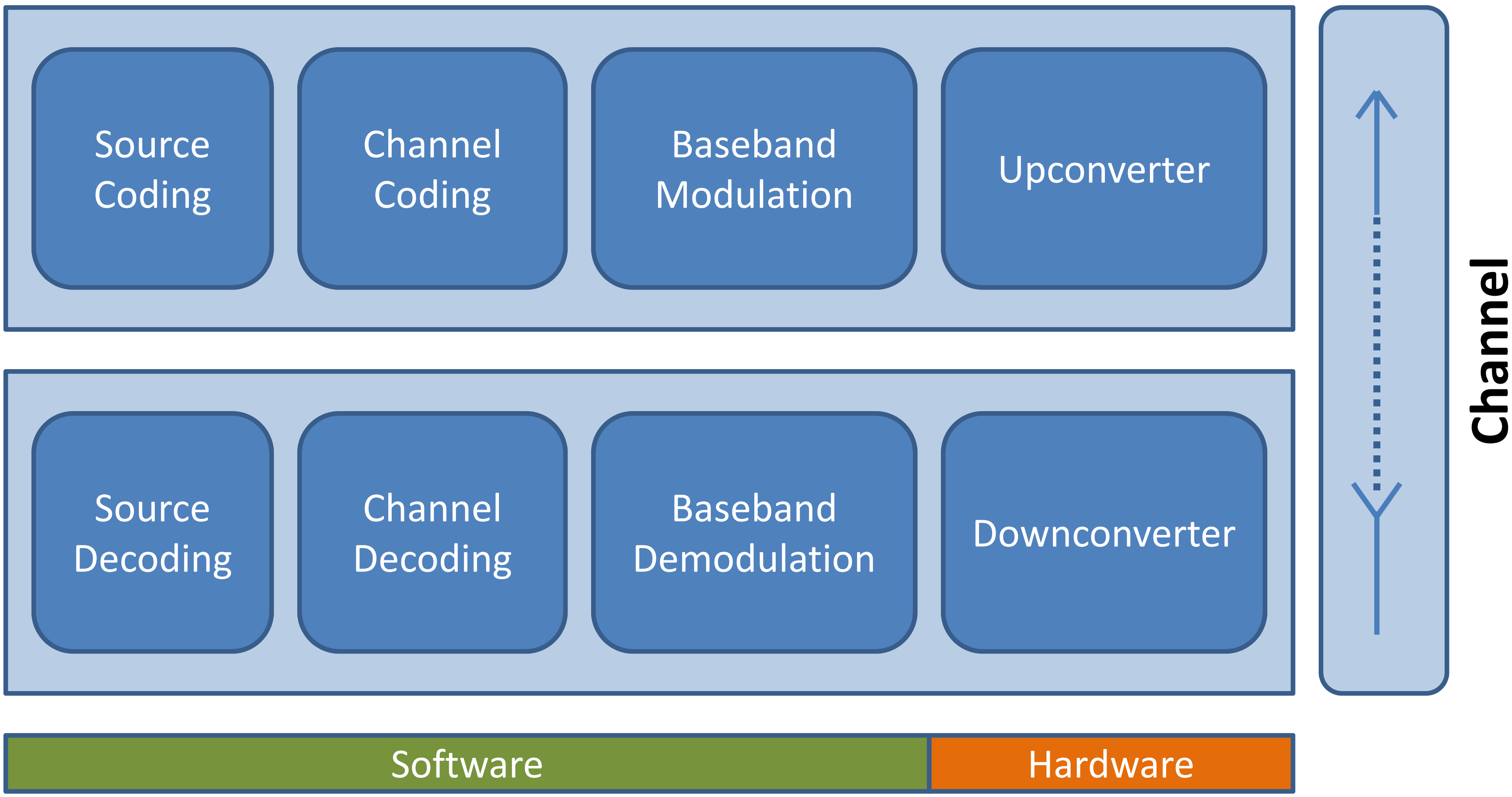
- Like traditional radio
  - It does what a typical radio does.
- Unlike traditional radio
  - Several components have been implemented computationally.
- Software can perform jobs that before were done by hardware.
- Extending the software towards the antenna

Traditional Radio



Traditional Radio Paradigm

Software Defined Radio

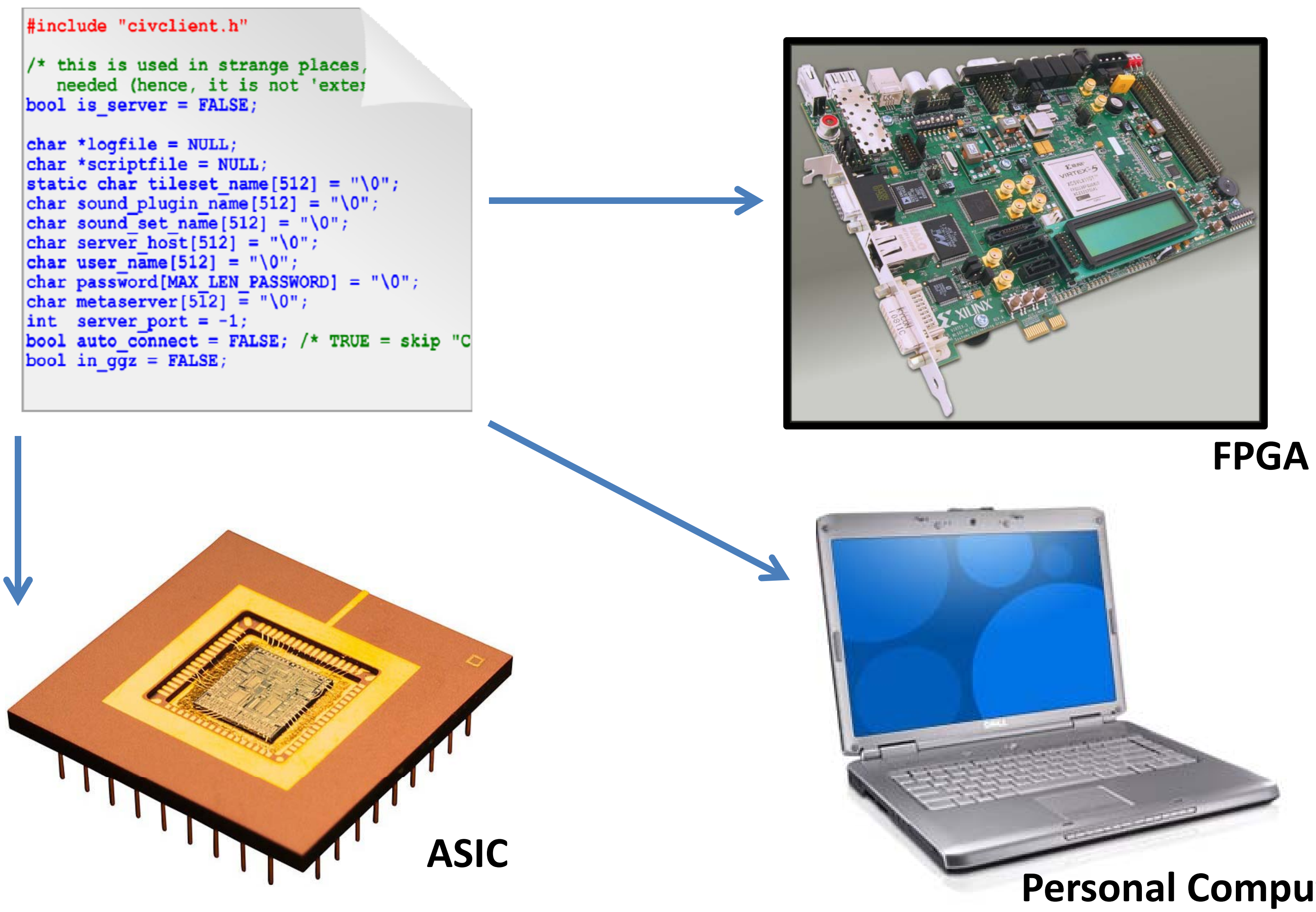


SDR Paradigm

Software Defined Radio Tools

- SoRa, by Microsoft
- Simulink, by MATLAB
- GNUradio, as an Open Source

Software Defined Radio Implementation



Software Defined Radio Implementation

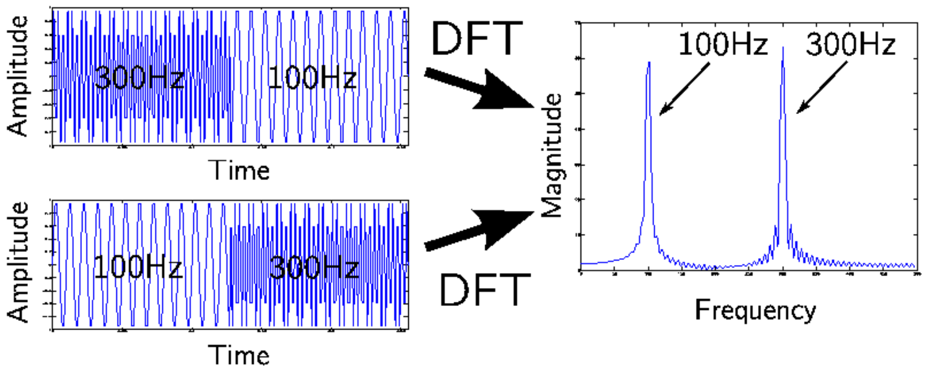
Personal Computer

Time-Frequency Representations

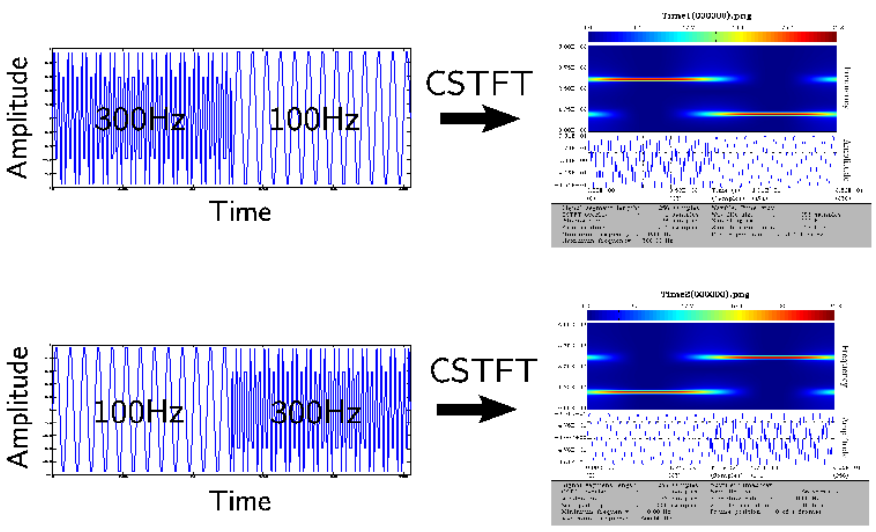
- Techniques and methods
- Signals whose characteristics are changing with time
- Allow to extract additional information
- High computational requirements

Time-Frequency Representations

NO DISCRIMINATION



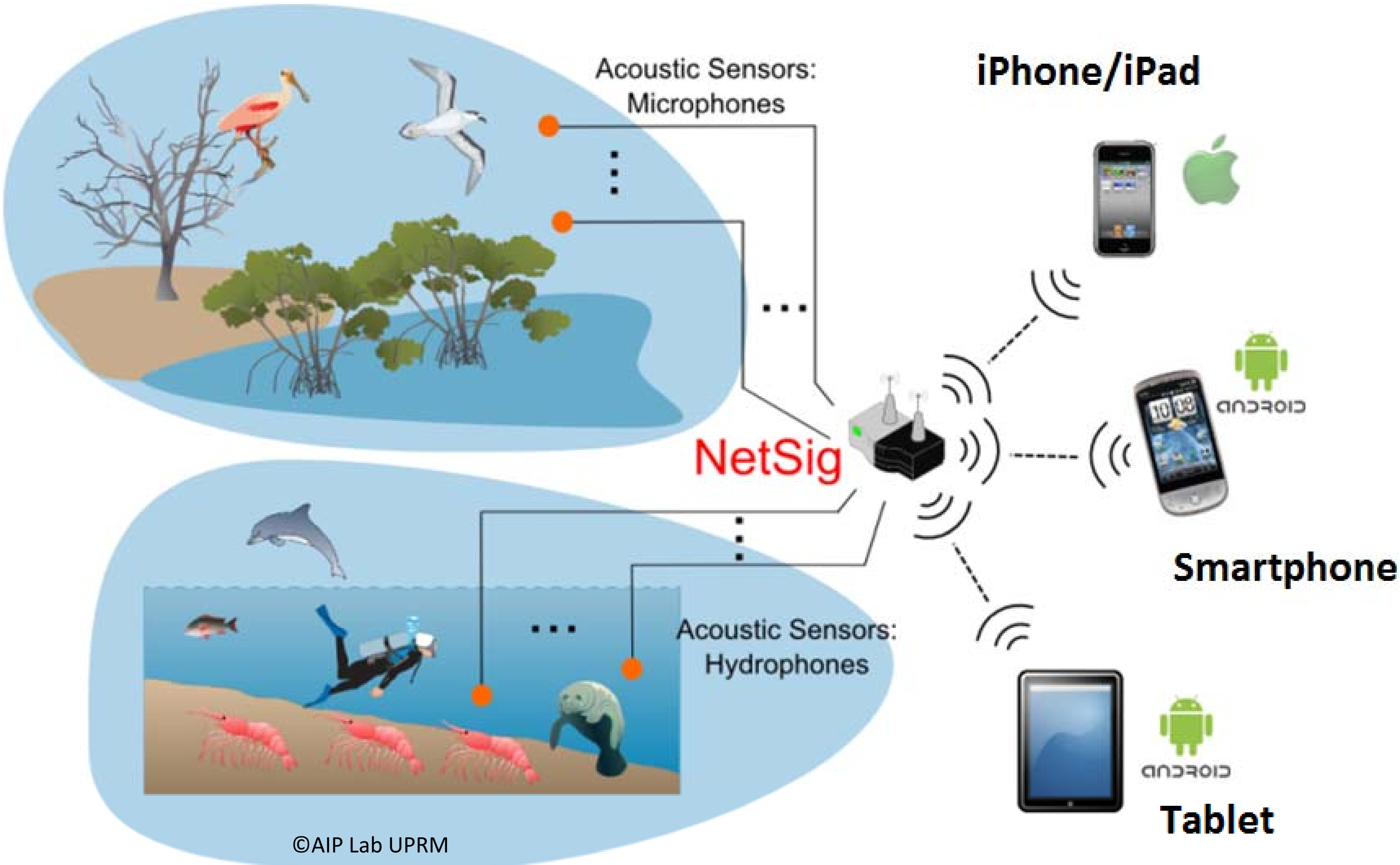
DISCRIMINATION



Methodology

- Available tools
  - SIRLAB, by the AIP Lab
  - GNUradio, as an Open Source
- Preliminary Results
  - SIRLAB Parallelization
  - User Integration: webSIRLAB, SIRDroid
- Framework for MIMO channel monitoring and simulation
  - Design
  - Implementation
  - Testing

SIRLAB on a NetSig Node Processor



SIRLAB

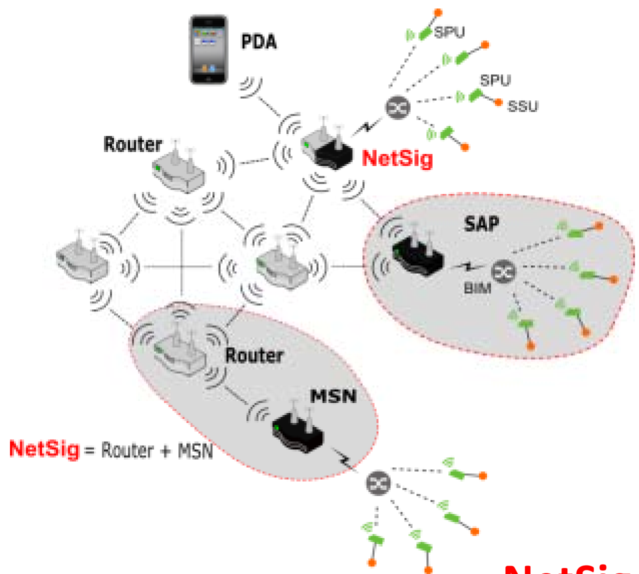
SIRLAB: A Tool for Research and Education

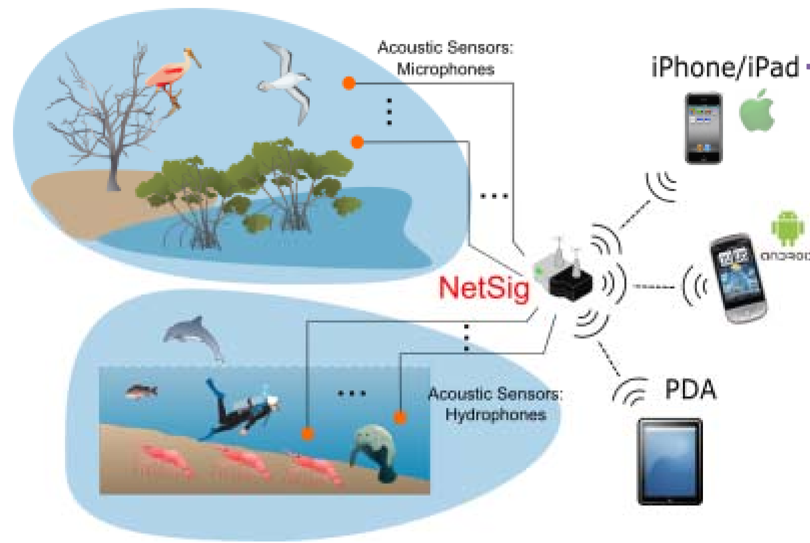
- **SIRLAB**: **SI**gnal **R**epresentation **LAB**oratory
- Open-Source Framework Developed at the AIP Lab
- Research Tool for Time-Frequency Analysis
- Educational Tool for Signal Representation
- Based on OpenCV and FFTW Software Libraries
- Uses R. Tolimieri's Theory on Time-Freq. Repres.
- **Input Format:** WAV File
- **Language Written:** C/C++
- **Speed Up Time:** 20+ Times Faster than Most Known MATLAB-based Time-Frequency Tools

- Known Time-Frequency (T-F) Toolboxes
  - http://tftb.nongnu.org (TFTB)
    - Developed by ***Rice Univ.:*** http://dsp.rice.edu/ & ***CN Recherche Scientifique***: http://www.cnrs.fr/
  - http://espace.library.uq.edu.au/view/UQ:211321
    - **T-F Signal Analysis (TFSA) Matlab Toolbox v.5.5**
    - Developed by ***Univ. of Queensland: uq.edu.au/***
  - http://www.mathworks.com/matlabcentral
    - **DiscreteTFDs – T-F Signal Analysis Software**
    - Developed by ***Jeff O'Neill: jco8@cornell.edu***

SIRLAB: Running on a NetSig Node

Monitoring Network

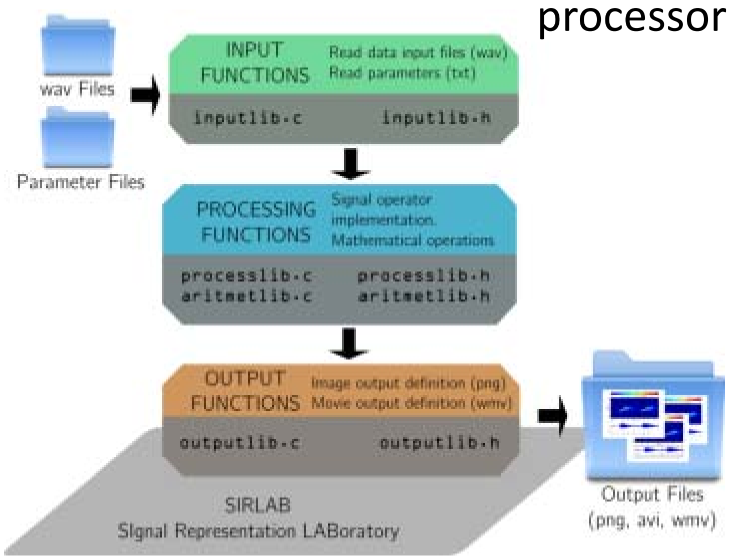


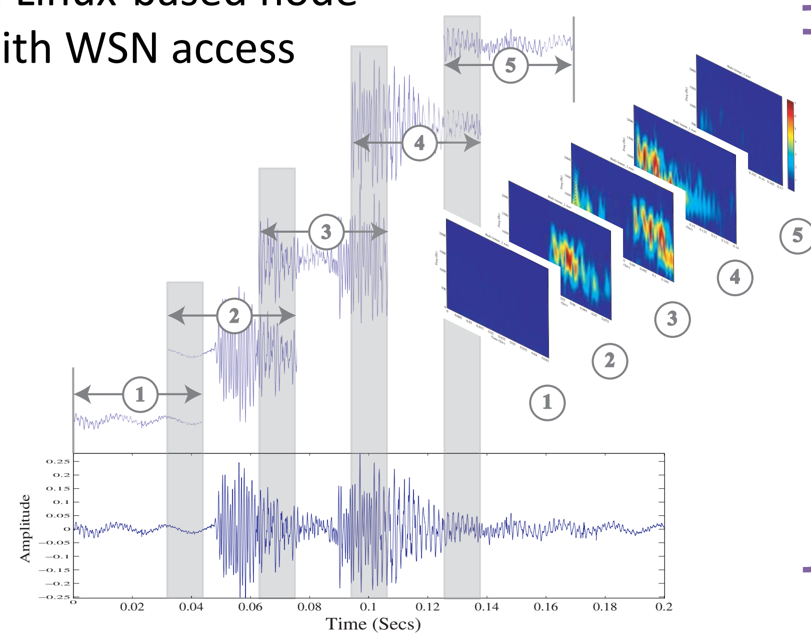


IWITDroid

NetSig: on a Linux-based node processor with WSN access

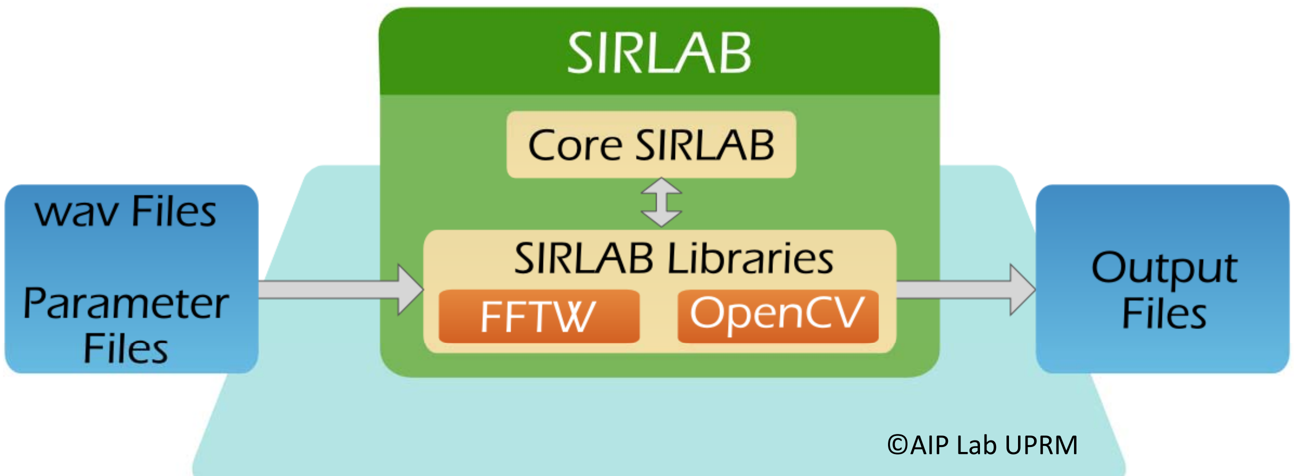
SIRLAB Framework



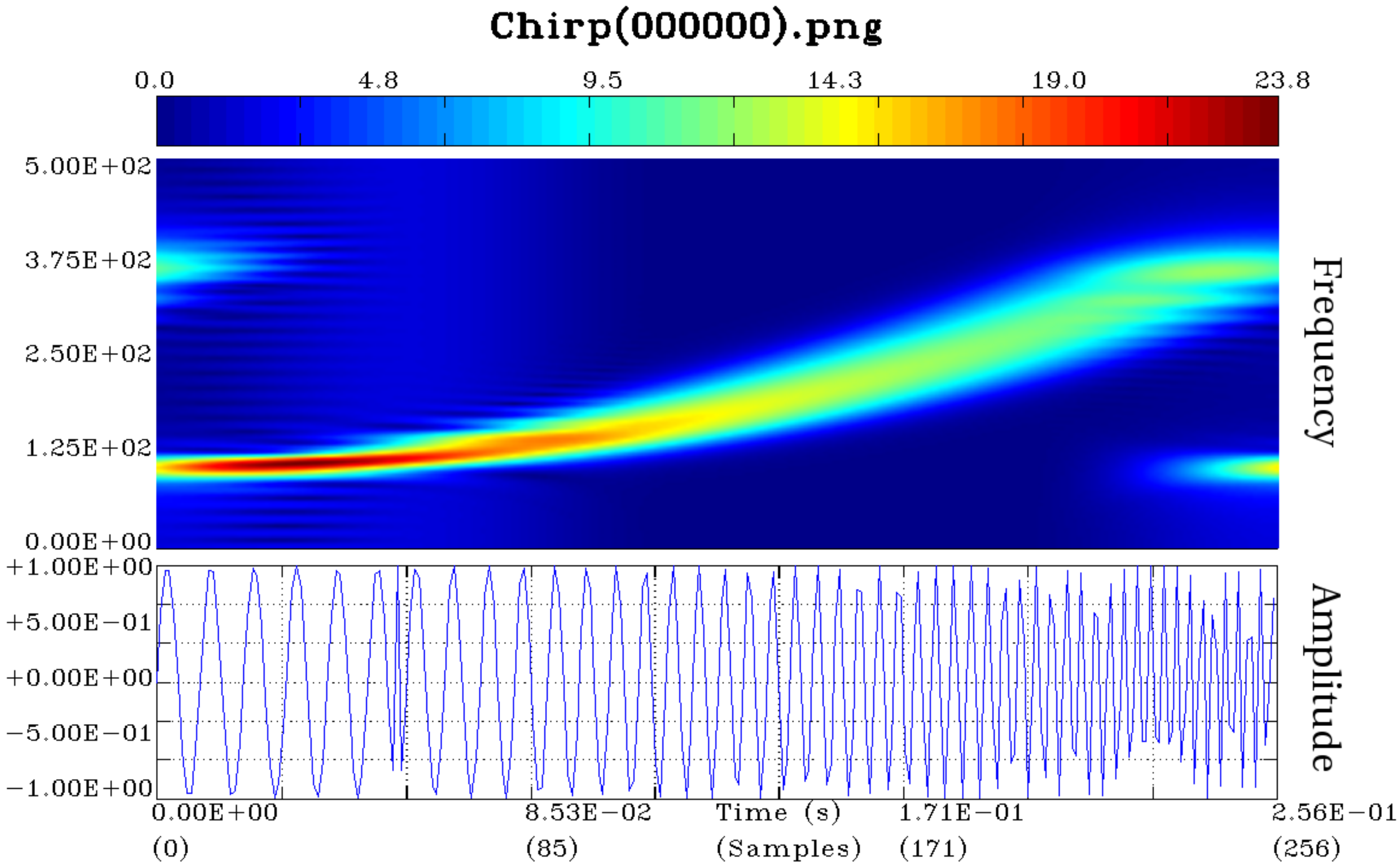


Spectrogram Displays

SIRLAB Architecture

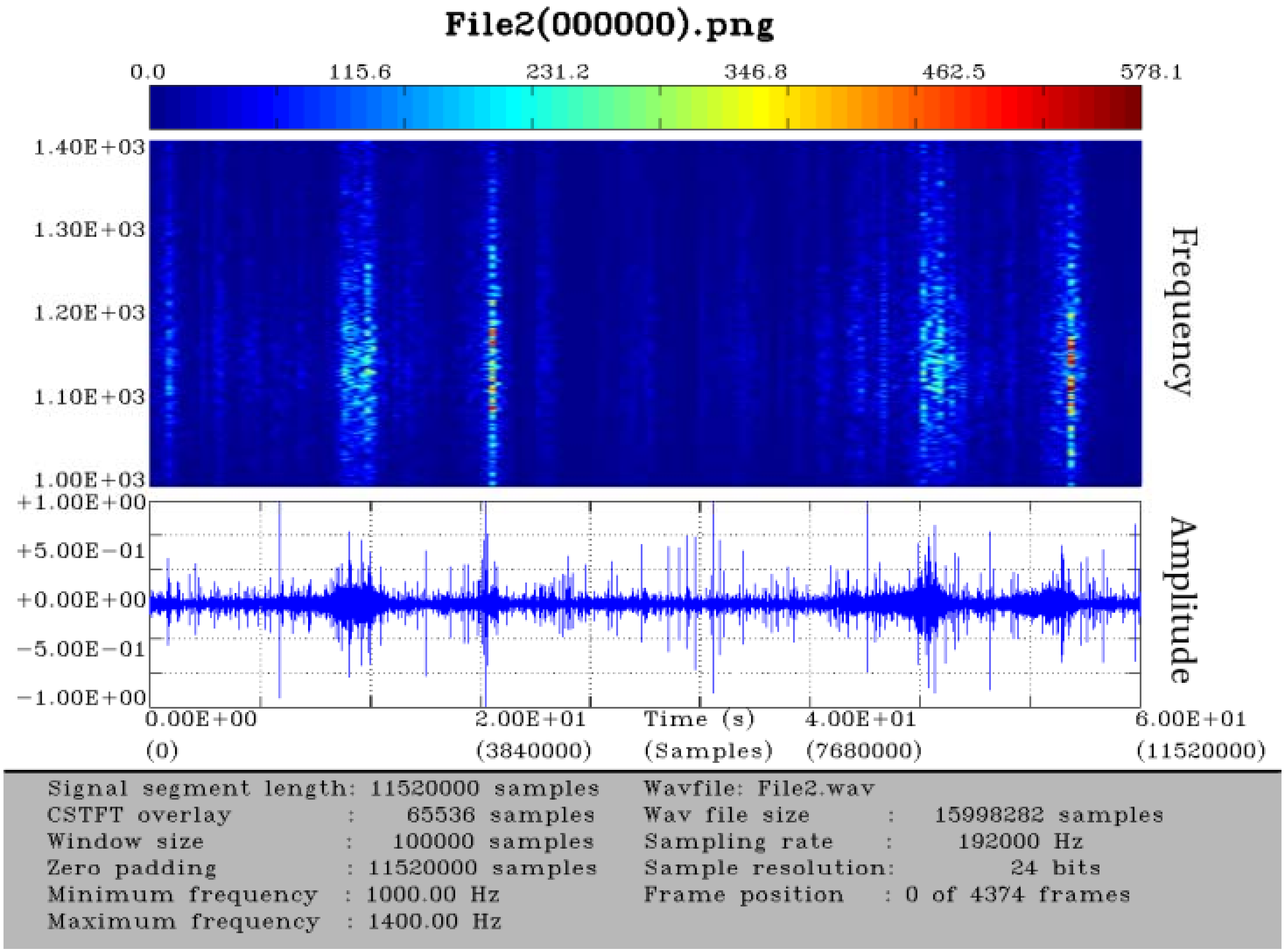


SIRLAB Frame Example



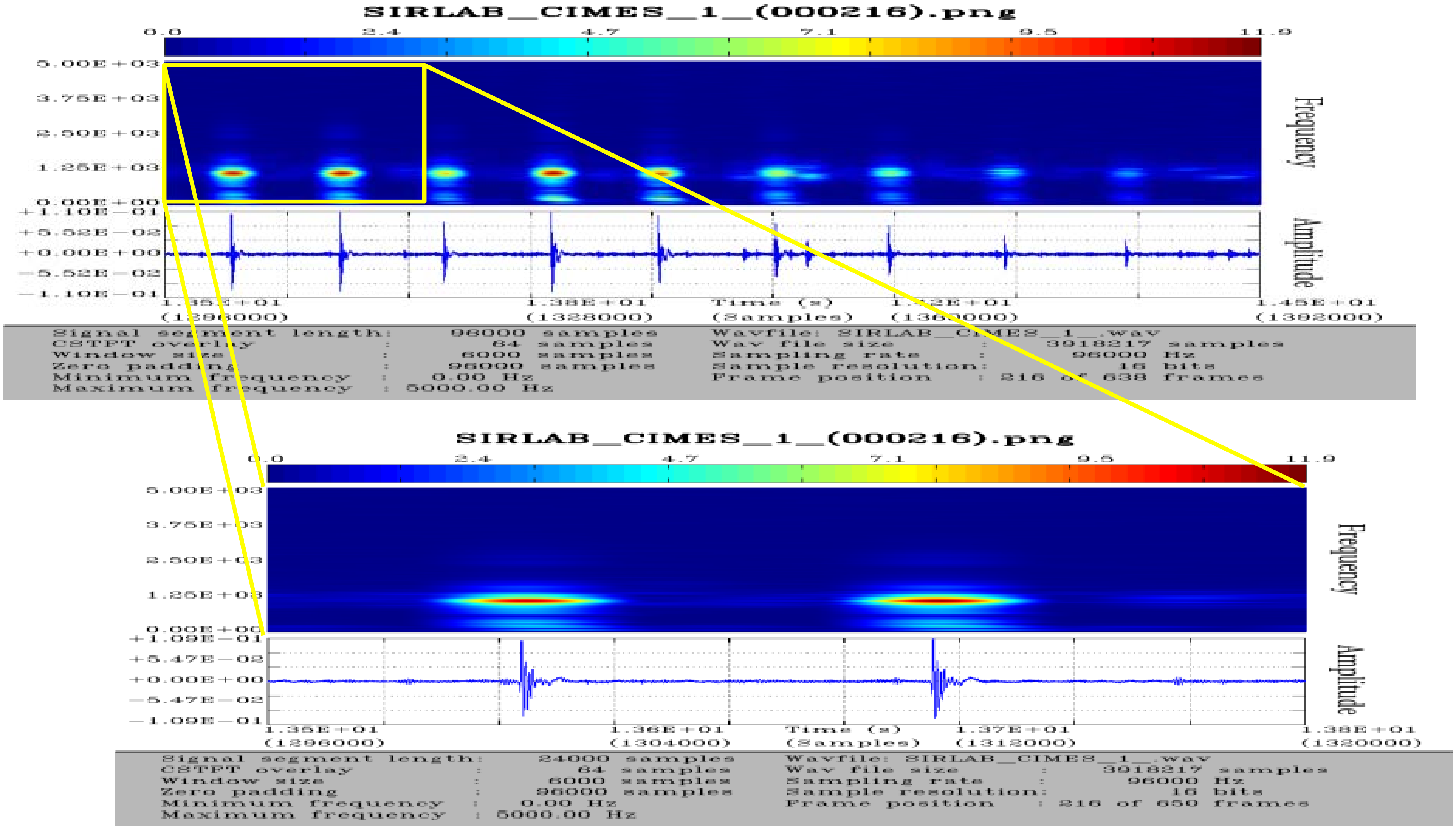
Signal segment length:	256 samples	Wavfile: Chirp.wav	
CSTFT overlay :	1 samples	Wav file size :	256 samples
Window size :	128 samples	Sampling rate :	1000 Hz
Zero padding :	256 samples	Sample resolution:	16 bits
Minimum frequency :	0.00 Hz	Frame position :	0 of 1 frames
Maximum frequency :	500.00 Hz		

SIRLAB's Standard Output

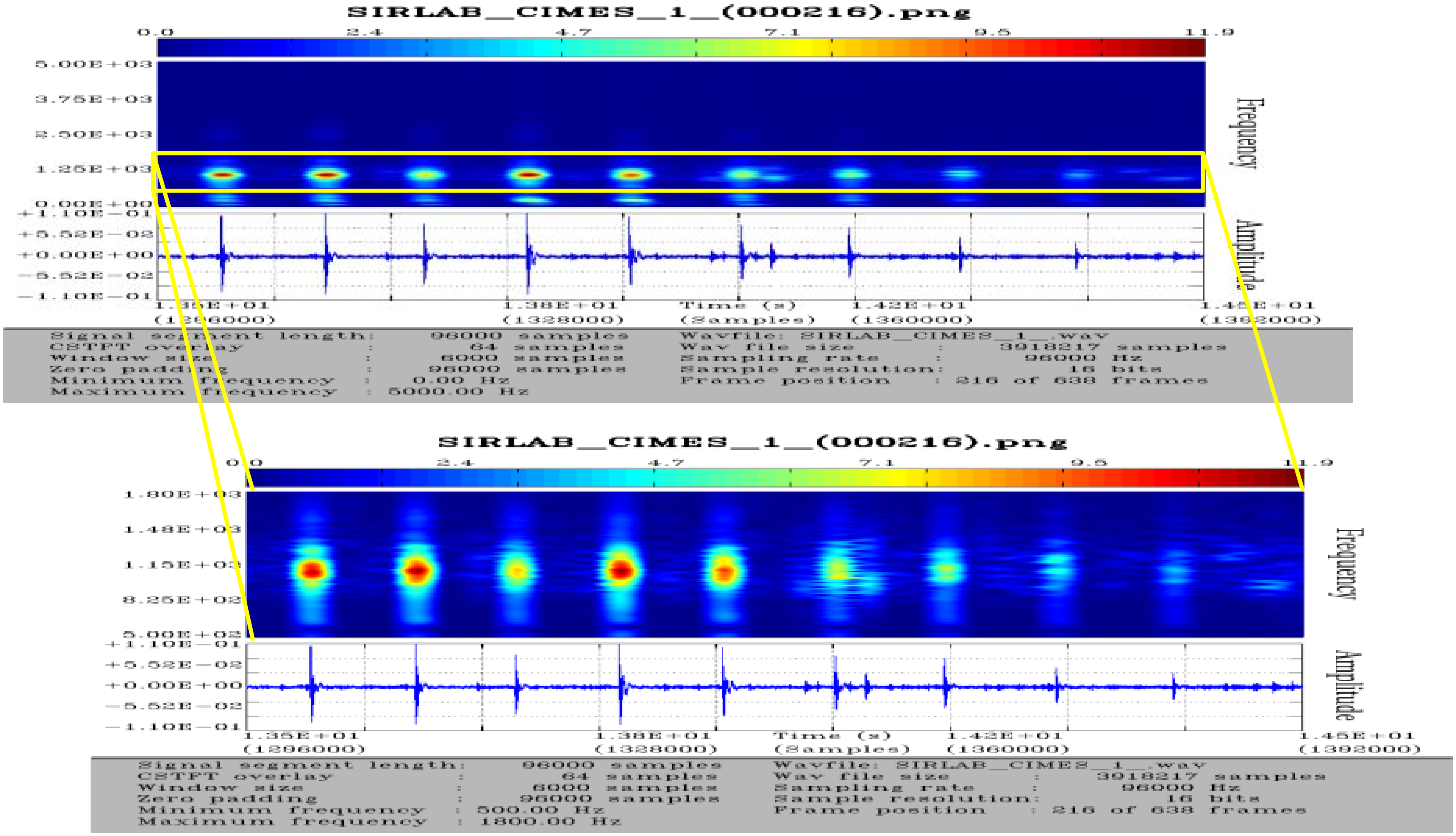


Sea waves…

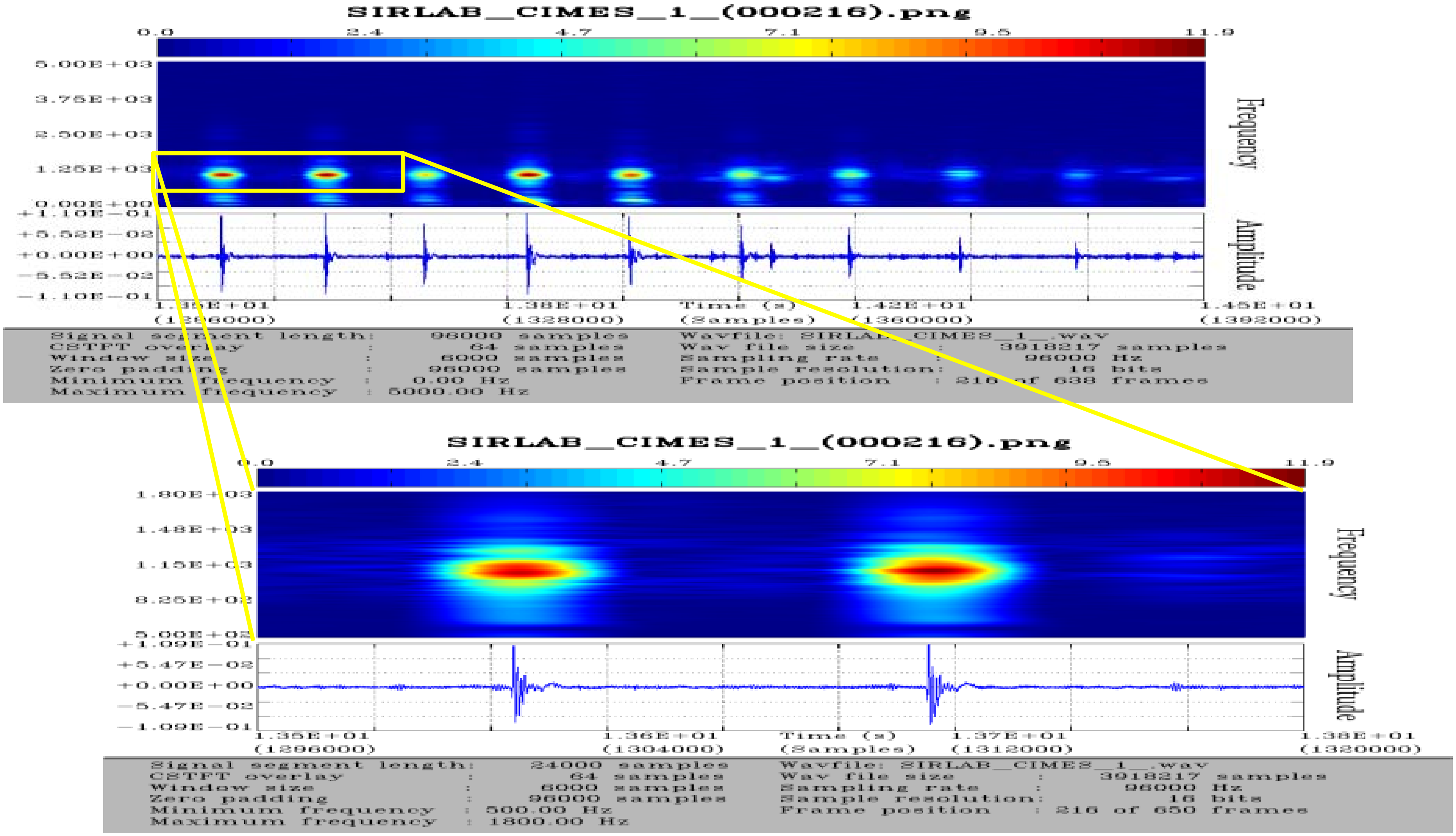
Frame Magnification: Time



Frame Magnification: Frequency



Frame Magnification: Time & Frequency



SIRLAB Challenges

- Parallelization Into Multiple Cores
- Improve the User Experience
  - Minimum Installation Effort
  - User Friendly Interaction
- Client – Server Architecture
- Integration with Mobile Devices

webSIRLAB

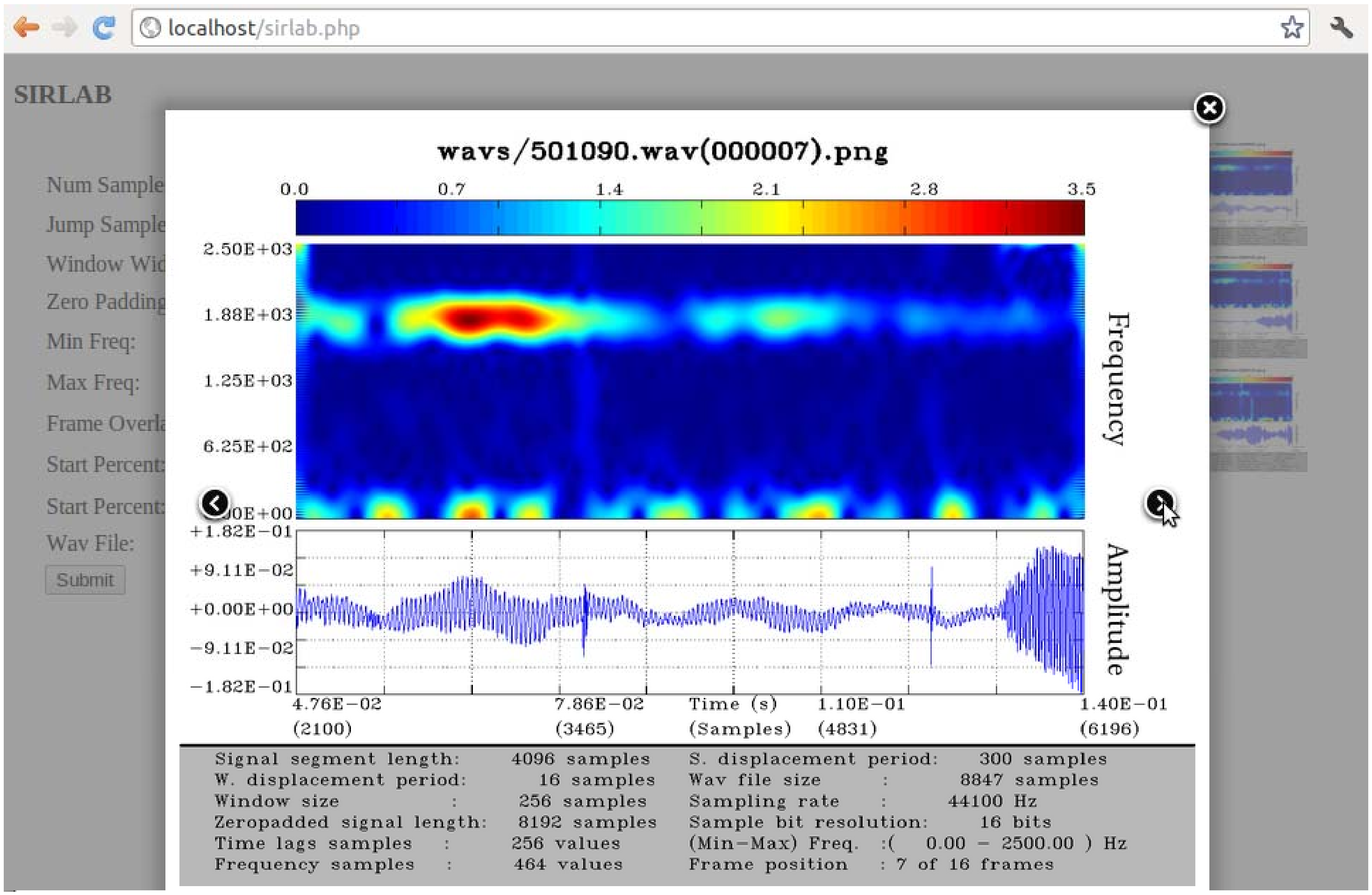
- Web Interface for SIRLAB
- Client Server Architecture
- Flexible Parameter Selection
- Easy Signal Submission for Processing at Server
- Friendly Display for Results
- Minimum Installation Effort
  - Modern **html5** web browser is required.

Research contribution

webSIRLAB's User Interface

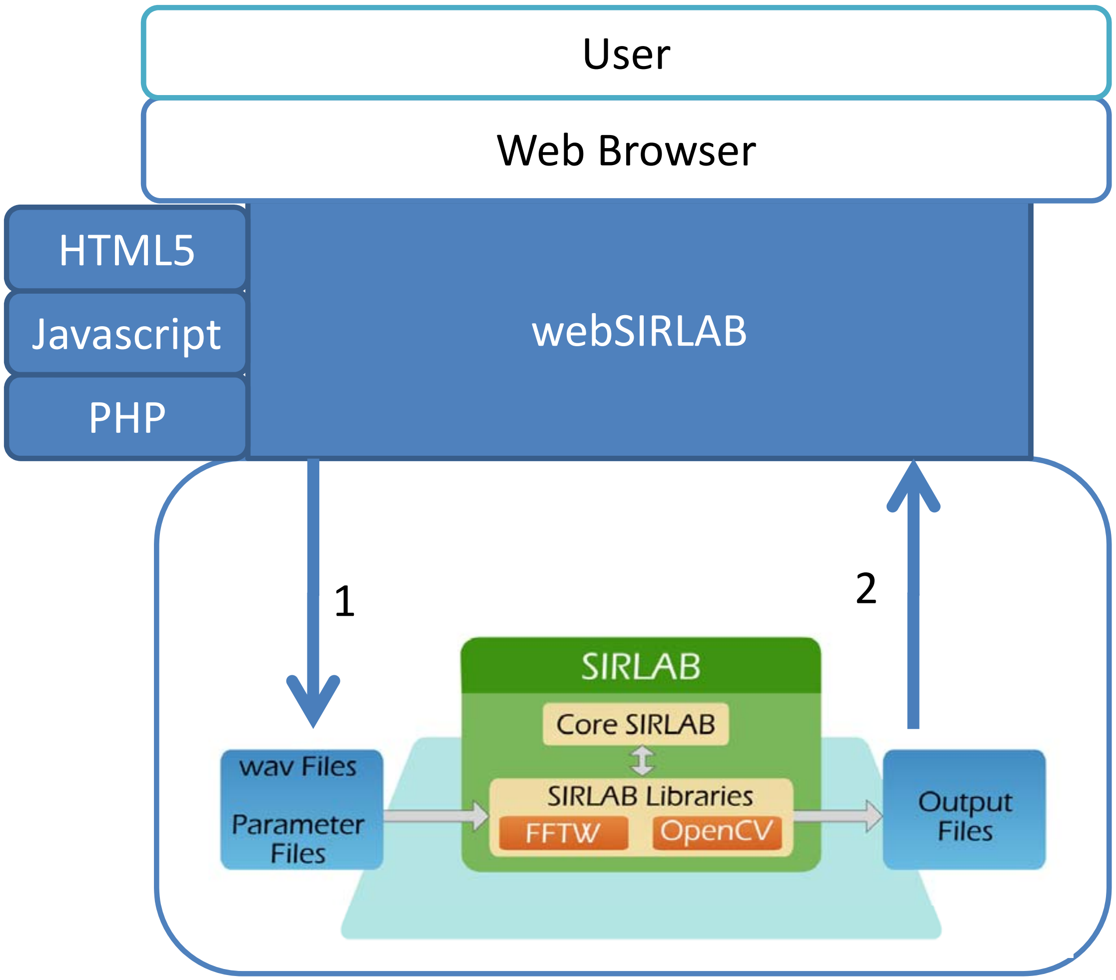
Research contribution

webSIRLAB



Research contribution

webSIRLAB Architecture



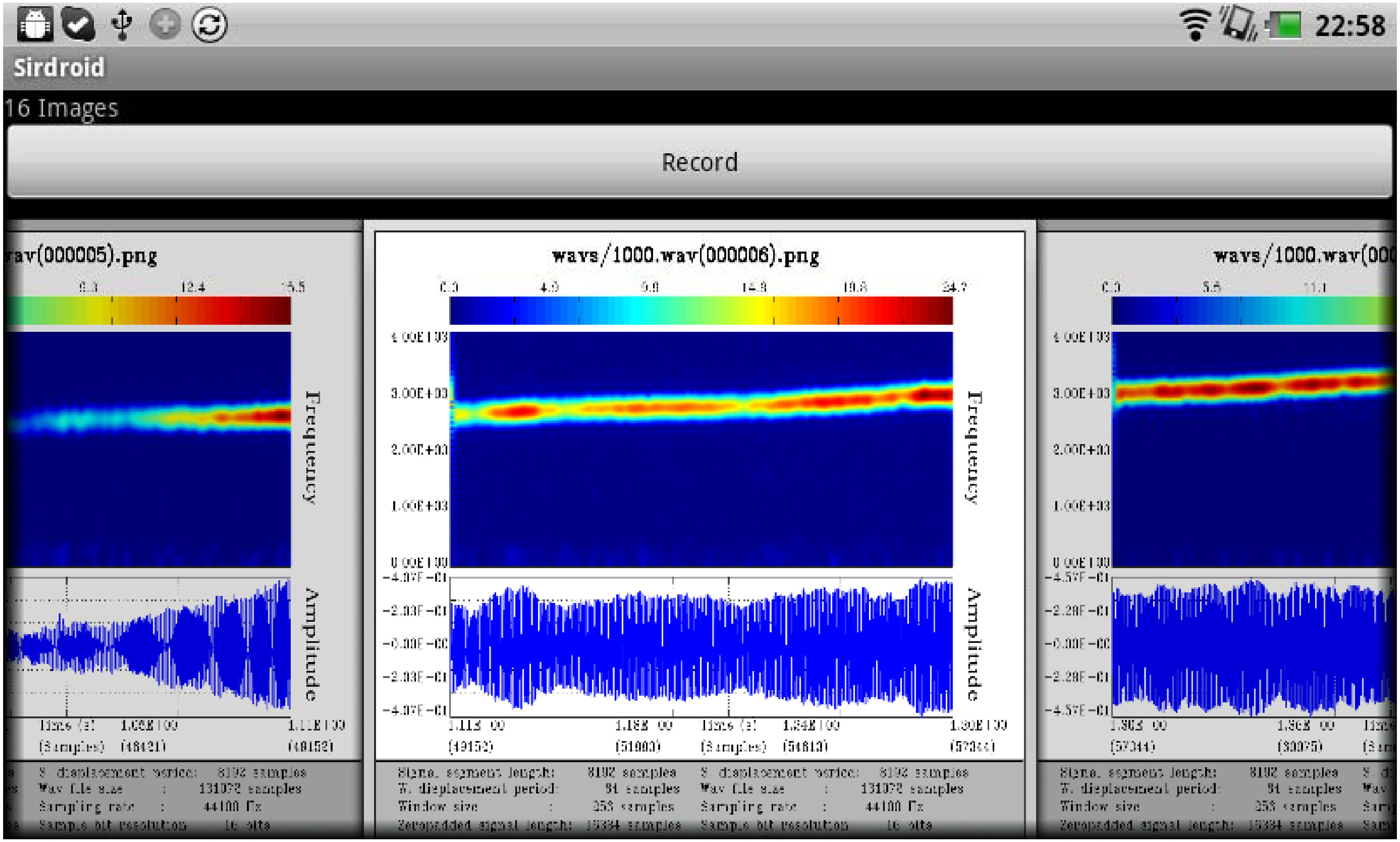
Research contribution

SIRDroid

- Android interface for SIRLAB
  - Android is the leading mobile operating system
- Client Server Architecture
- Can record data on real environment and get results in a few seconds

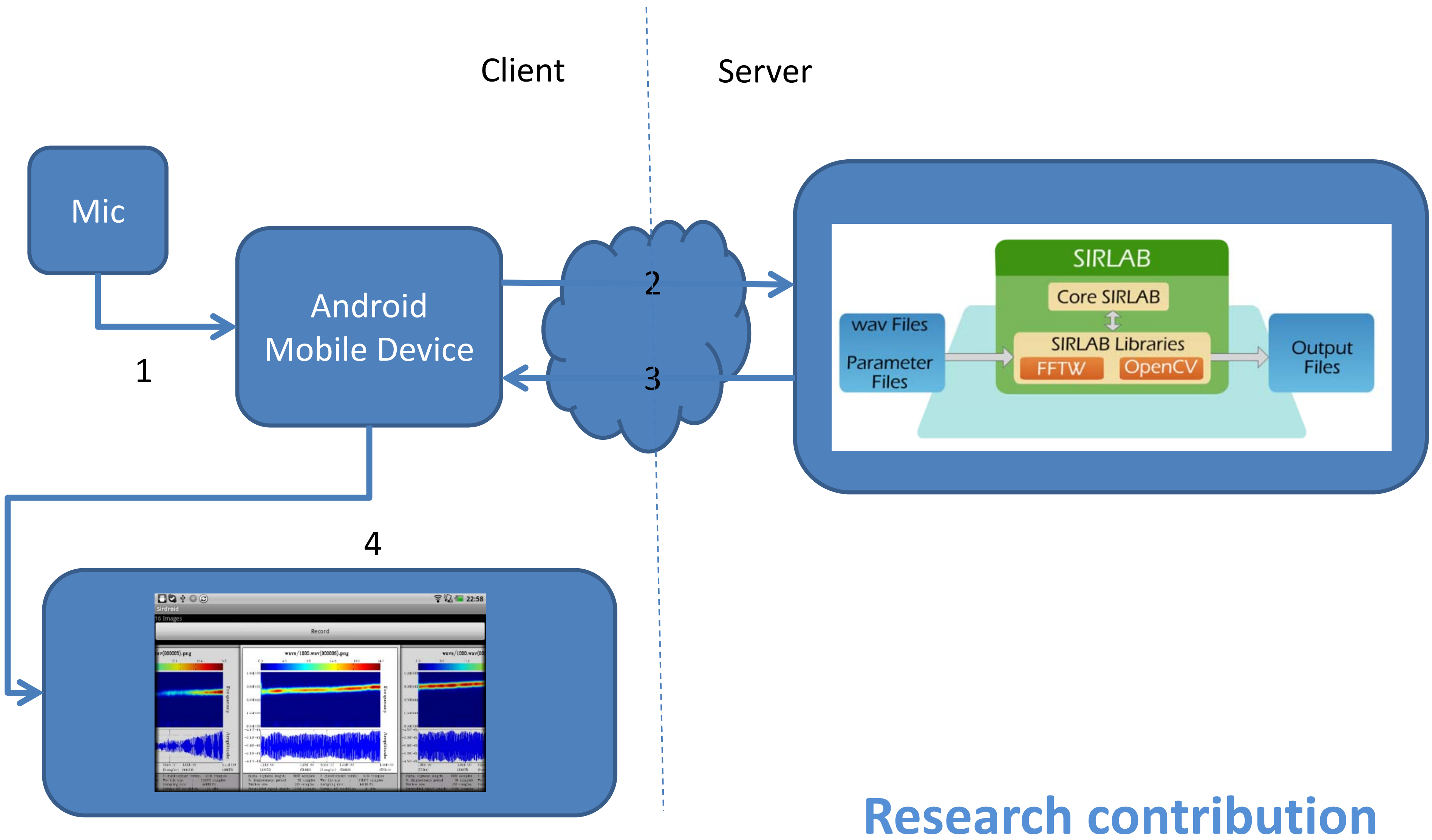
Research contribution

SIRDroid



Research contribution

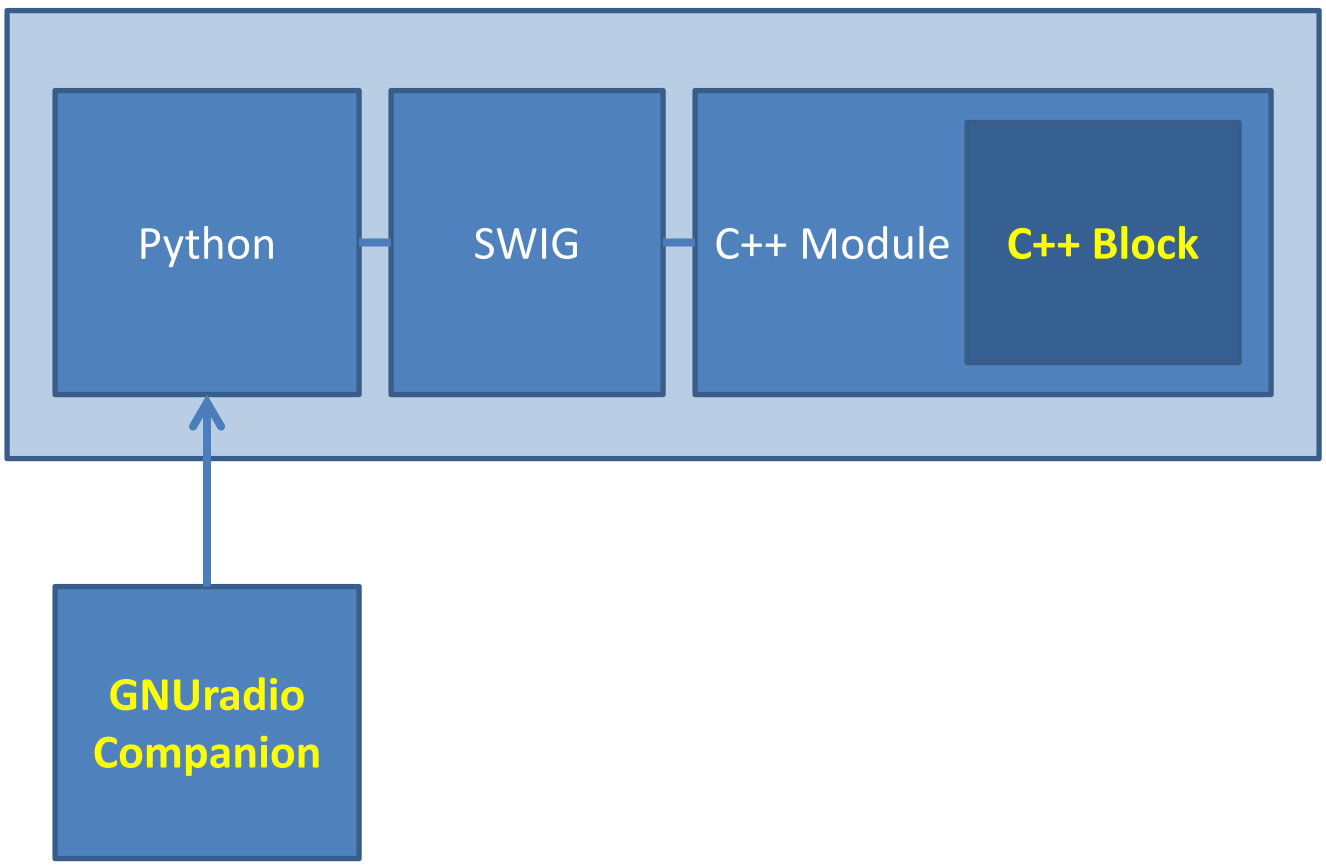
SIRDroid Architecture



GNUradio

- Open Source Software
- Provides a library of signal processing blocks
- Sends and receives signals
- Audio, USRP, Network (TCP, UDP), File, Wav File
- USRP makes GNUradio wireless capable
- Software Defined Radio

GNUradio Architecture

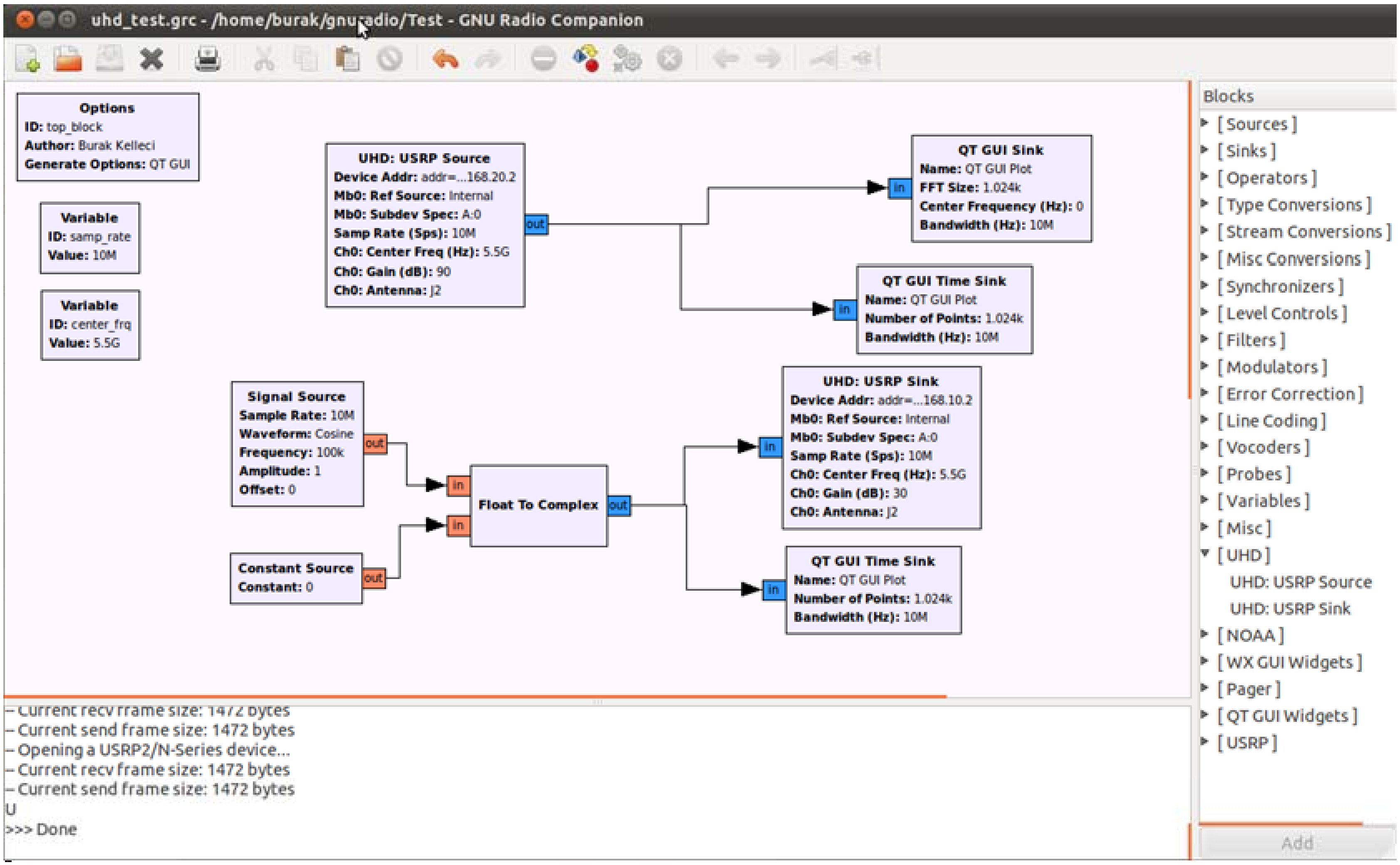


*SWIG: Simplified Wrapper Interface Generator

GNUradio Companion

- Graphical Interface to GNUradio
- Iconic Programming Language
- Powerful Editor
  - Place the Required Blocks
  - Configure Each Block
  - Connect them Together
  - Build and Run

GNUradio Companion



GNUradio Challenges

- Build Time-frequency Representations
  - Short Time Fourier Transform
  - Ambiguity Function
  - Wigner Distribution
  - Choi – Williams Distribution

The theory behind the solution

DESIGN

Design Outline

- Time-Frequency Representations
  - Algorithms, complexity and parallelization
- MIMO Channel
  - MIMO System Modeling
  - Proposed Channel Modeling
    - Modulation-Convolution-Delay
    - Delay-Convolution-Modulation

Time-Frequency Representations

Short Time Fourier Transform

$$S_{x,w}[k,m] = \sum_{n\in\mathbb{Z}_{\mathbb{N}}} x[n]w[n-m]e^{-j2\pi\frac{kn}{N}}$$

Ambiguity Function

$$A_{f,g}[m,k] = \sum_{n\in Z_N} f[n]\, g^*\left[\langle n+m\rangle_N\right] e^{-j2\pi\frac{kn}{N}}$$

Wigner Distribution

$$W_x[n,k] = \frac{1}{N}\sum_{\tau=0}^{N-1}\sum_{v=0}^{N-1}\sum_{l=0}^{N-1} \rho_N e^{j2\pi vl} x[\langle l+\tau\rangle_N]x^*[l]e^{-j\frac{2\pi}{N}(nv+k\tau)}$$

Choi-Williams Distribution

$$C_x[t,f] = \sum_{m=0}^{N-1}\sum_{k=0}^{N-1} A_x[m,k]\Phi[m,k]e^{j\frac{2\pi}{N}(kt-mf)},$$

Short Time Fourier Transform

Algorithm 3 Short Time Fourier Transform

Input: Signal $x[n] \in l^2(\mathbb{Z}_N)$, Window width $w > 0 \in \mathbb{Z}$, Zero-padding $P > 0 \in \mathbb{Z}$.

Output: Matrix $s[M][P]$

$M \leftarrow \lfloor \frac{N}{w} \rfloor$
for $i = 0 \rightarrow M - 1$ **do**
    $wn \in \mathbb{Z}^N \leftarrow 0$
    $y \in \mathbb{Z}^P \leftarrow 0$
    for $j = 0 \rightarrow w - 1$ **do**
        $wn[j + w \cdot i] \leftarrow 1$
    end for
    for $j = 0 \rightarrow N - 1$ **do**
        $y[j] \leftarrow x[j] \cdot wn[j]$
    end for
    $s[i] \leftarrow \mathrm{dft}(y)$
end for

Computational Time Complexity: $O(n^2 \log(n))$

Research contribution

Ambiguity Function

Algorithm 4 Ambiguity Function

Input: Signal $x[n] \in l^2(\mathbb{Z}_N)$, $y[n] \in l^2(\mathbb{Z}_N)$

Output: Matrix $A[N][N]$

for $i = 0 \to N - 1$ **do**
    for $j = 0 \to N - 1$ **do**
        $y1[j] \leftarrow \mathrm{conj}(y[\mathrm{mod}(j + i, N)])$
    end for
    for $j = 0 \to N - 1$ **do**
        $z[j] \leftarrow x[j] \cdot y1[j]$
    end for
    $A[i] \leftarrow \mathrm{dft}(z)$
end for

Computational Time Complexity: $O(n^2 \log(n))$

Research contribution

Wigner Distribution

Algorithm 5 Wigner

Input: Signal $x[n] \in l^2(\mathbb{Z}_N)$, $y[n] \in l^2(\mathbb{Z}_N)$
Output: Matrix $W[N][N]$

for $i = 0 \rightarrow N - 1$ **do**
    for $j = 0 \rightarrow N - 1$ **do**
        $y1[j] \leftarrow \text{conj}(y[\text{mod}(j + i, N)])$
    end for
    for $j = 0 \rightarrow N - 1$ **do**
        $z[j] \leftarrow x[j] \cdot y1[j]$
    end for
    $A[i] \leftarrow \text{dft}(z)$
end for
$W \leftarrow \text{dft2}(A)$

Computational Time Complexity: $O(n^2 \log(n))$

Research contribution

Choi-Williams Distribution

Algorithm 6 Choi-Williams

Input: Signal $x[n] \in l^2(\mathbb{Z}_N)$, $y[n] \in l^2(\mathbb{Z}_N)$, α
Output: Matrix $W[N][N]$

for $i = 0 \to N-1$ **do**
    for $j = 0 \to N-1$ **do**
        $y1[j] \leftarrow y[\mathrm{mod}(j+i, N)]$
    end for
    for $j = 0 \to N-1$ **do**
        $z[j] \leftarrow x[j] \cdot y1[j]$
    end for
    $A[i] \leftarrow \mathrm{dft}(z)$
    for $j = 0 \to N-1$ **do**
        $A[i][j] \leftarrow A[i][j] \cdot e^{-\alpha(i \cdot j)^2}$
    end for
end for
$W1 \leftarrow \mathrm{dft2}(A)$
$W \leftarrow \mathrm{conj}(W1)$

Computational Time Complexity: $O(n^2 \log(n))$ **Research contribution**

MIMO System

The channel is seen as a linear system

$$y = Hx + N.$$

H then is decomposed

$$H = U\Sigma V^*$$

Finally

$$U^{-1}y = \Sigma(V^*x) + N$$

The symbols are preprocessed at the transmitter and post-processed at the receiver

MIMO System

- The channel must be estimated
  - Several mechanisms have been proposed and implemented
  - Subject to errors
- The Shannon capacity increases linearly with every pair of antennas at the transmitter and receiver.

Formulated MIMO Channel

- A “MxN” MIMO channel can be seen as a collection of C=MxN Single Input Single Output (SISO) Channels.
- Every SISO Channel is then modeled like a channel with 3 operators:
  - **Delay:** The signal has a latency while is propagating
  - **Convolution:** The medium acts like a FIR Filter
  - **Modulation:** The medium inserts a Doppler shift
- The order is important, this yields two approximations:
  - Delay-Convolution-Modulation (DCM)
  - Modulation-Convolution-Delay (MCD)

Research contribution

DCM

Operators

$$y_{a,b} = g_{a,b} \odot_D X_{k_{a,b}},$$

where $g_{a,b} \in l^2(\mathbb{Z}_D)$

$$g_{a,b} = T_{h_{a,b}}\{f_{a,b}\},$$

where $f_{a,b} \in l^2(\mathbb{Z}_D)$

$$f_{a,b} = x_b \circledast_D \delta_{m_{a,b}}.$$

Composition

$$y_{a,b} = (T_{h_{a,b}}\{x_b \circledast_D \delta_{m_{a,b}}\}) \odot_D X_{k_{a,b}}.$$

MIMO

$$y_a = \sum_{b=0}^{M-1} (T_{h_{a,b}}\{x_b \circledast_D \delta_{m_{a,b}}\}) \odot_D X_{k_{a,b}}.$$

Research contribution

MCD

Operators

$$y_{a,b} = g_{a,b} \circledast_D \delta_{m_{a,b}},$$

where $g_{a,b} \in l^2(\mathbb{Z}_D)$

$$g_{a,b} = T_{h_{a,b}}\{f_{a,b}\},$$

where $f_{a,b} \in l^2(\mathbb{Z}_D)$

$$f_{a,b} = x_b \odot_D X_{k_{a,b}}.$$

Composition

$$y_{a,b} = (T_{h_{a,b}}\{x_b \odot_D X_{k_{a,b}}\}) \circledast_D \delta_{m_{a,b}}.$$

MIMO

$$y_a = \sum_{b=0}^{M-1} (T_{h_{a,b}}\{x_b \odot_D X_{k_{a,b}}\}) \circledast_D \delta_{m_{a,b}},$$

Research contribution

The code behind the solution

IMPLEMENTATION

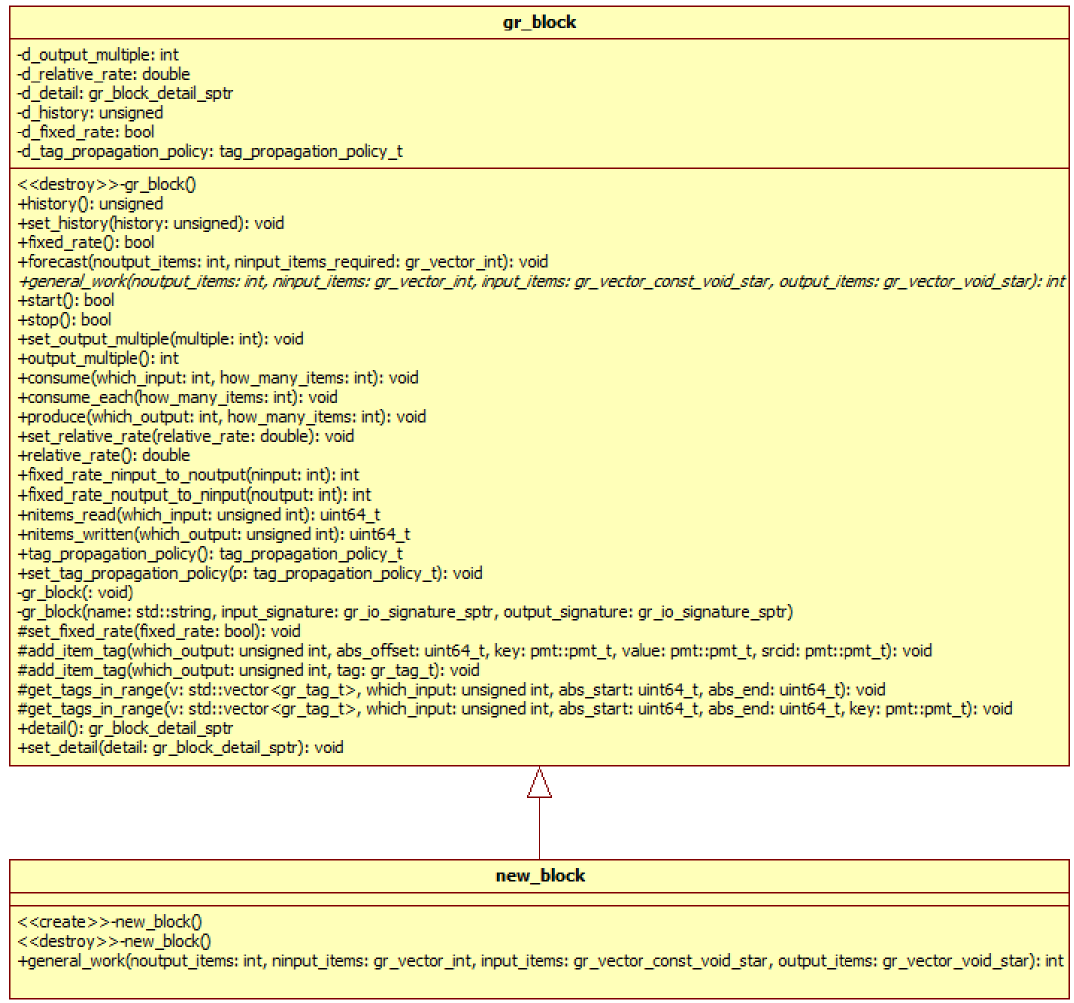
Time Frequency Representations

- The main implementation is done in C++
- Object oriented programming
- Each time-frequency representation is a class
  - Implements from *gr_block* interface
  - The method *general_work* is implemented
  - A constructor sets the initial properties of each time frequency representation

Time-Frequency Representations

Block Interface

Constructor, destructor and general_work are implemented



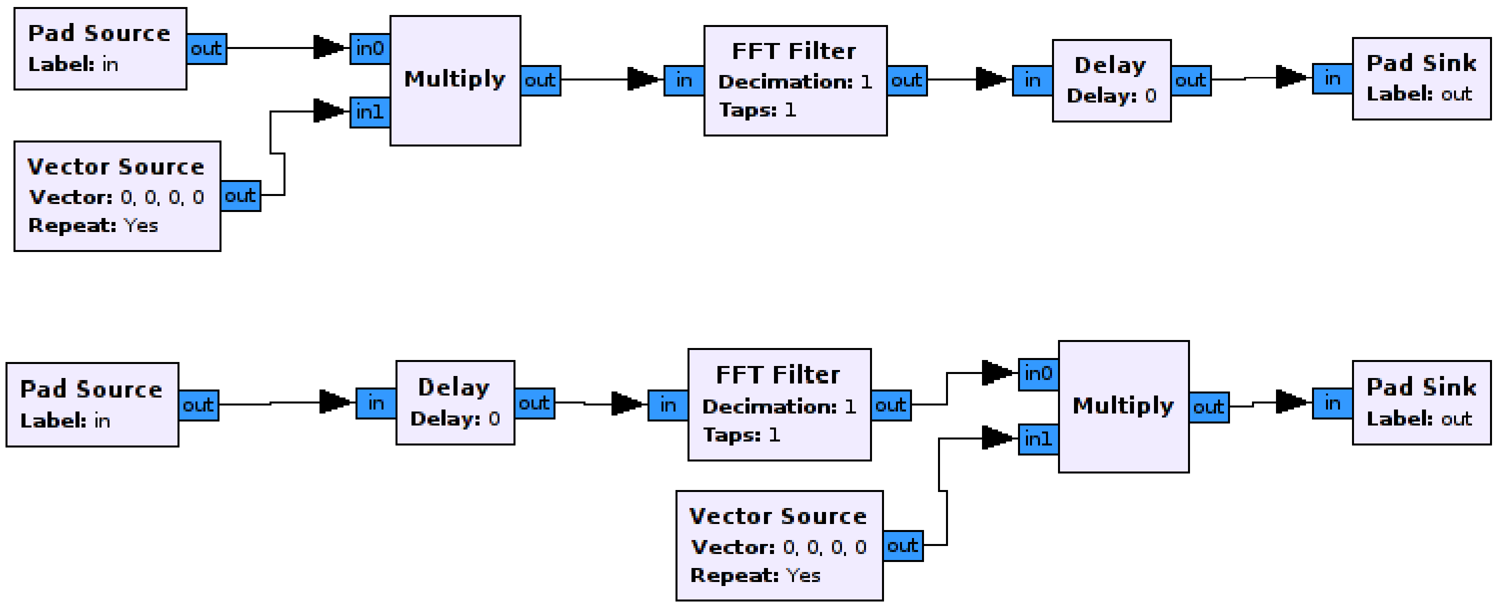
Parallelization

- Each running block is a thread into the whole system
- Our time-frequency representation blocks are executed in parallel as an option
  - Data parallelism
  - OpenMP was selected for parallelization
    - API for automatic parallelization, include compiler directives, libraries and functions
    - Cross compatibility, great support and meets the project requirement

MIMO Channel

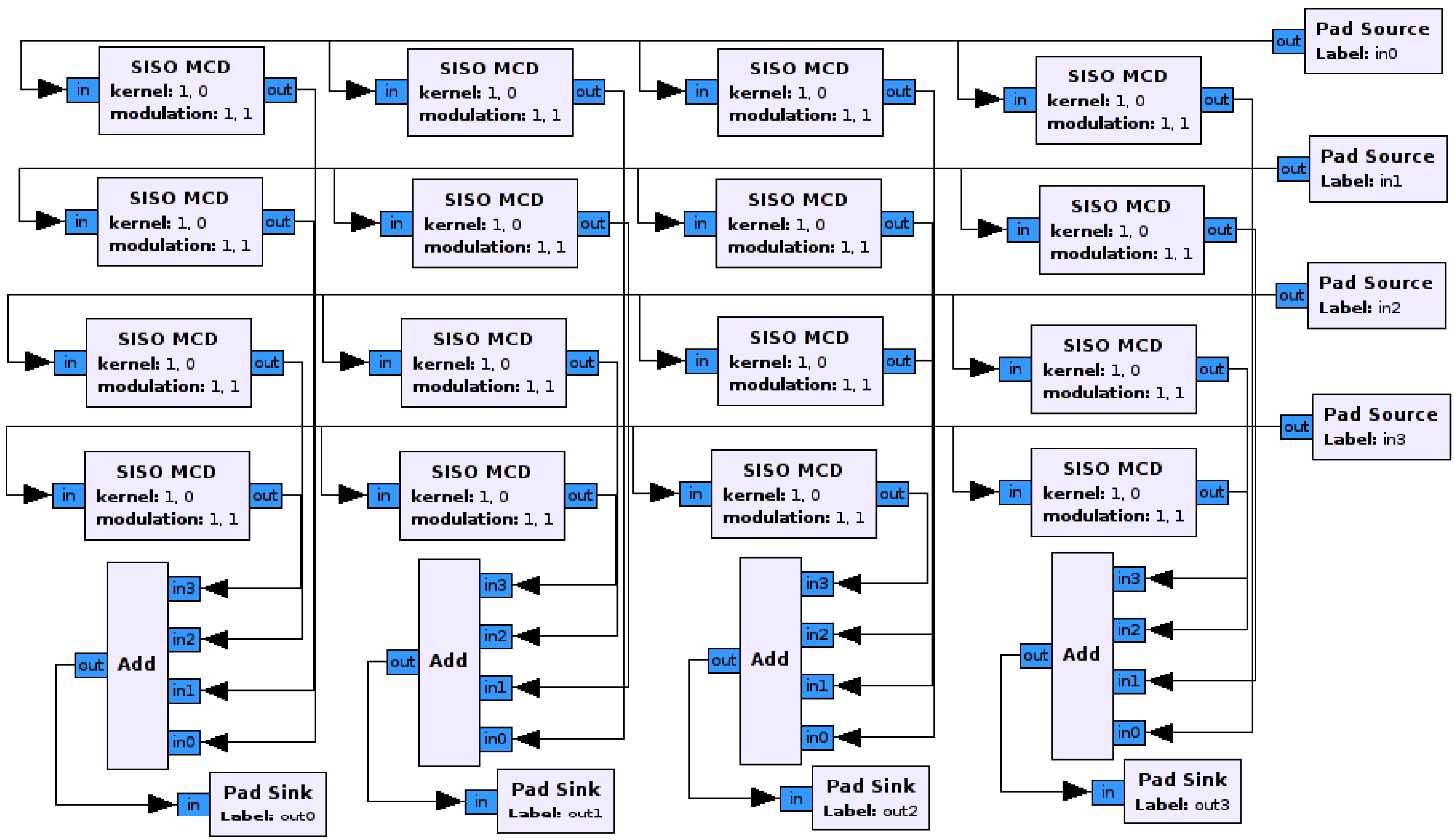
- Development of each block, MCD and DCM internally with GNUradio Companion
- Iconic Programming

MCD and DCM Blocks



Research contribution

4x4 MIMO



Research contribution

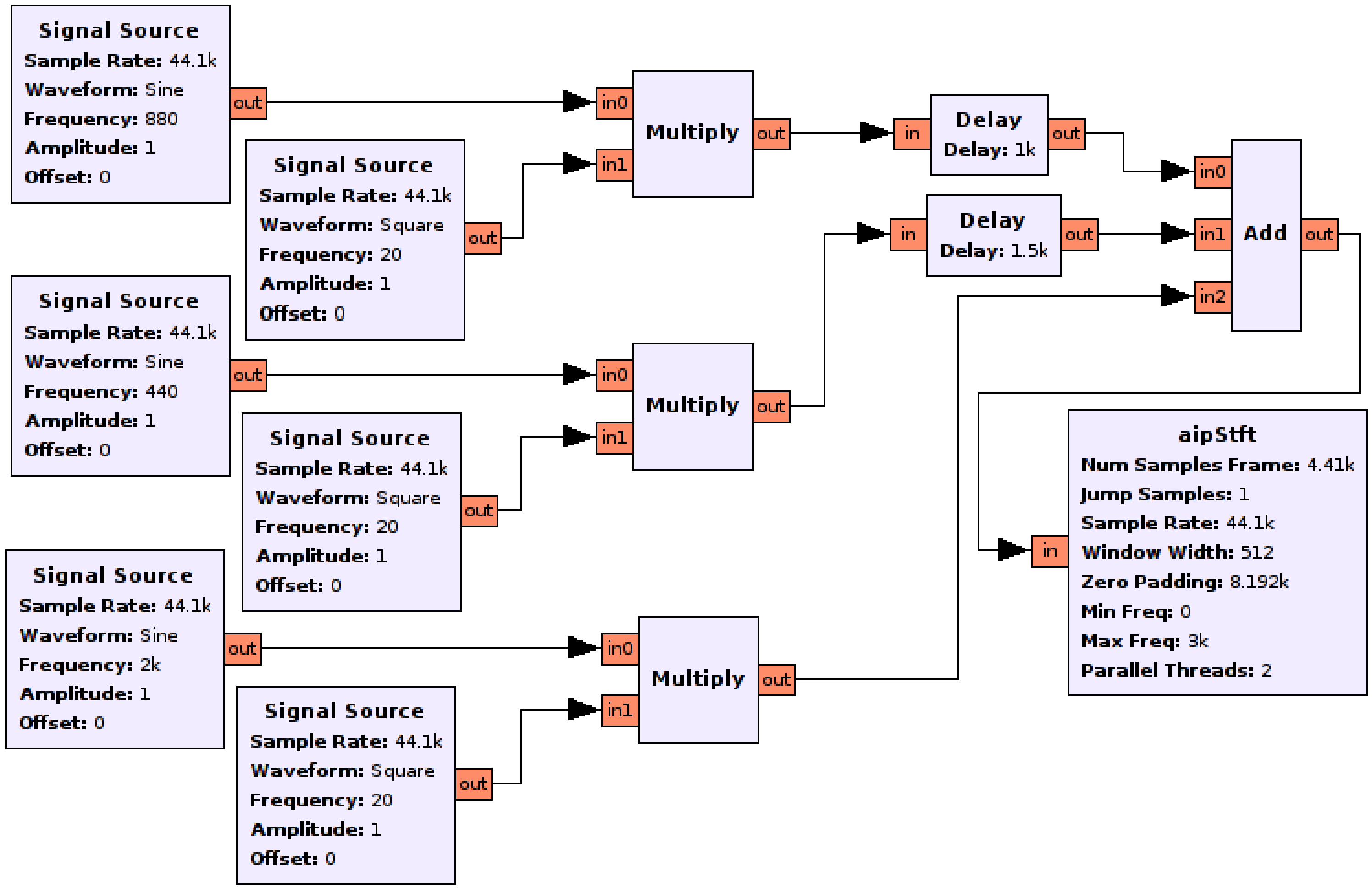
The results behind the solution

TESTING

Acoustic Channel Surveillance

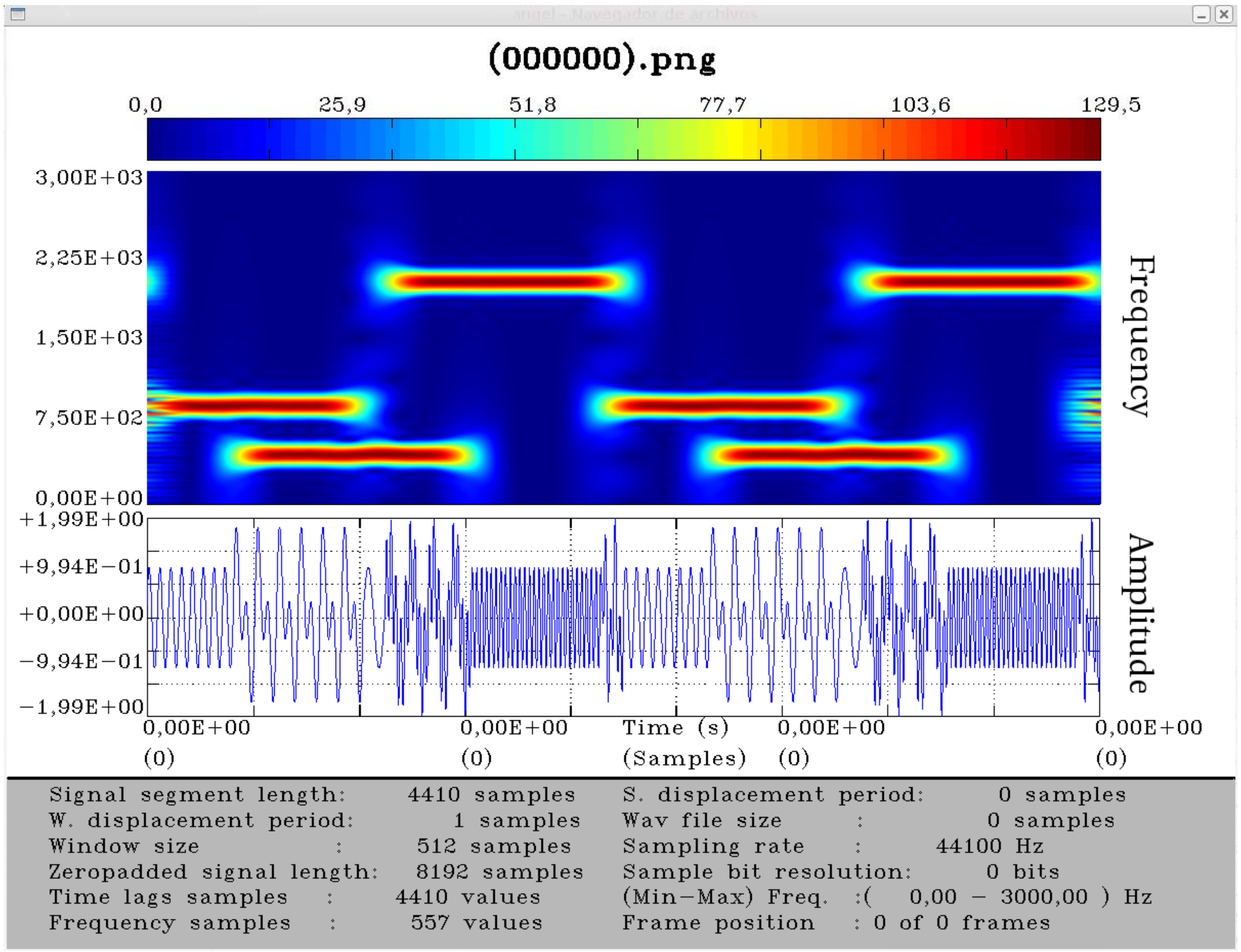
- The STFT can be used to monitor the environment
  - Species characterization in bioacoustics
  - Sonar

Acoustic Channel Surveillance



Research contribution

Channel Surveillance

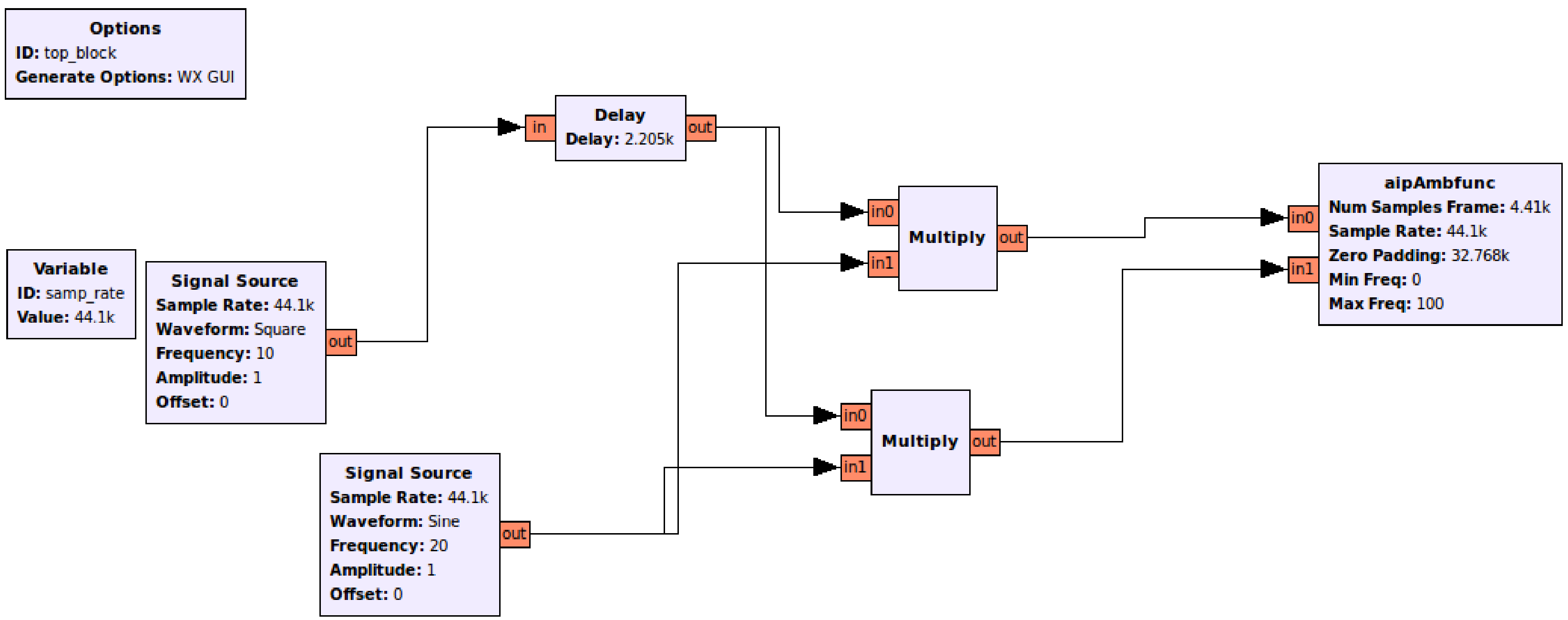


Research contribution

Ambiguity Function

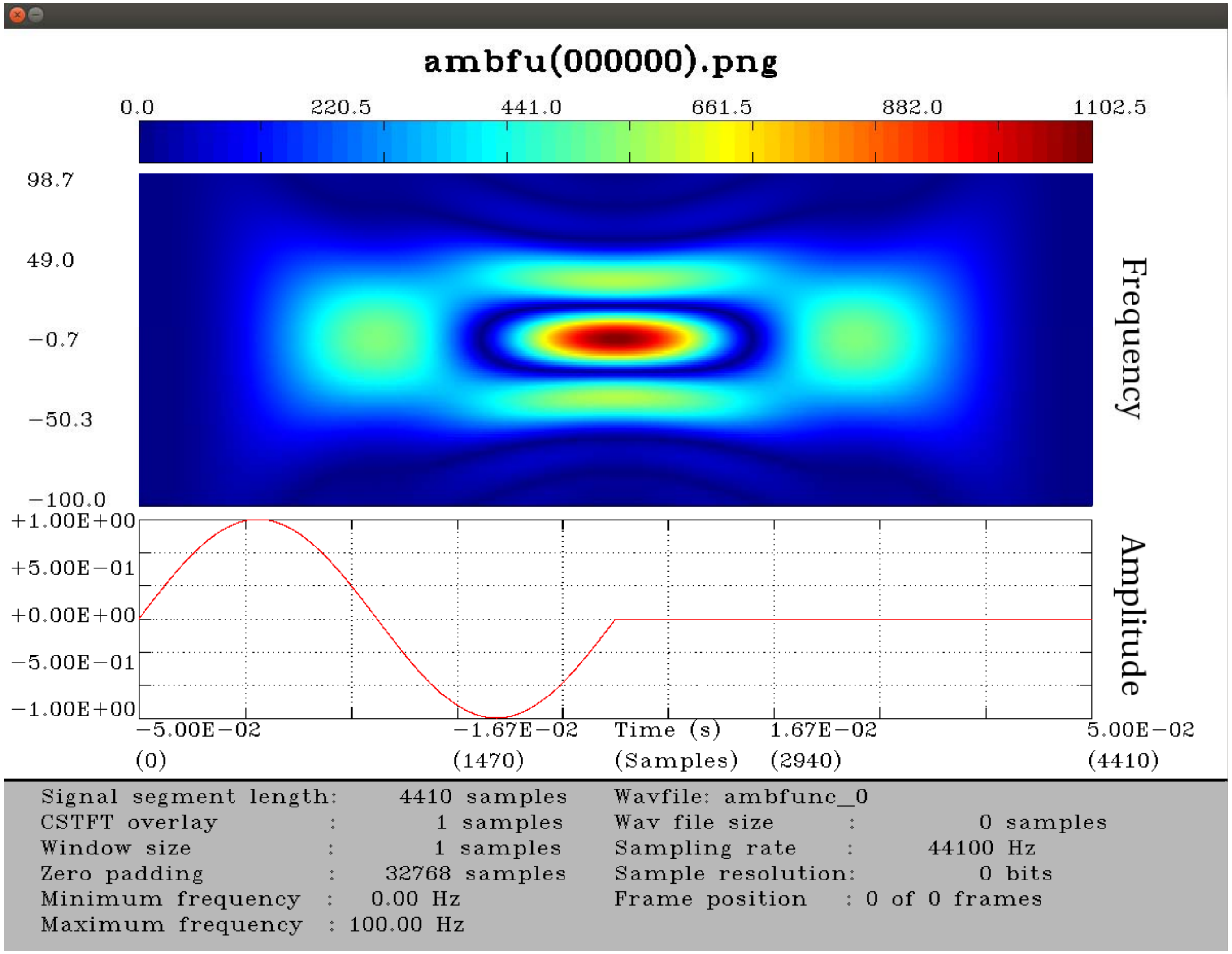
- Used for Sonars
- Design of Signals
- Detection of Delay and Doppler Shifts
- Testing With Well Known Signals
  - Sine
  - Square

Sine



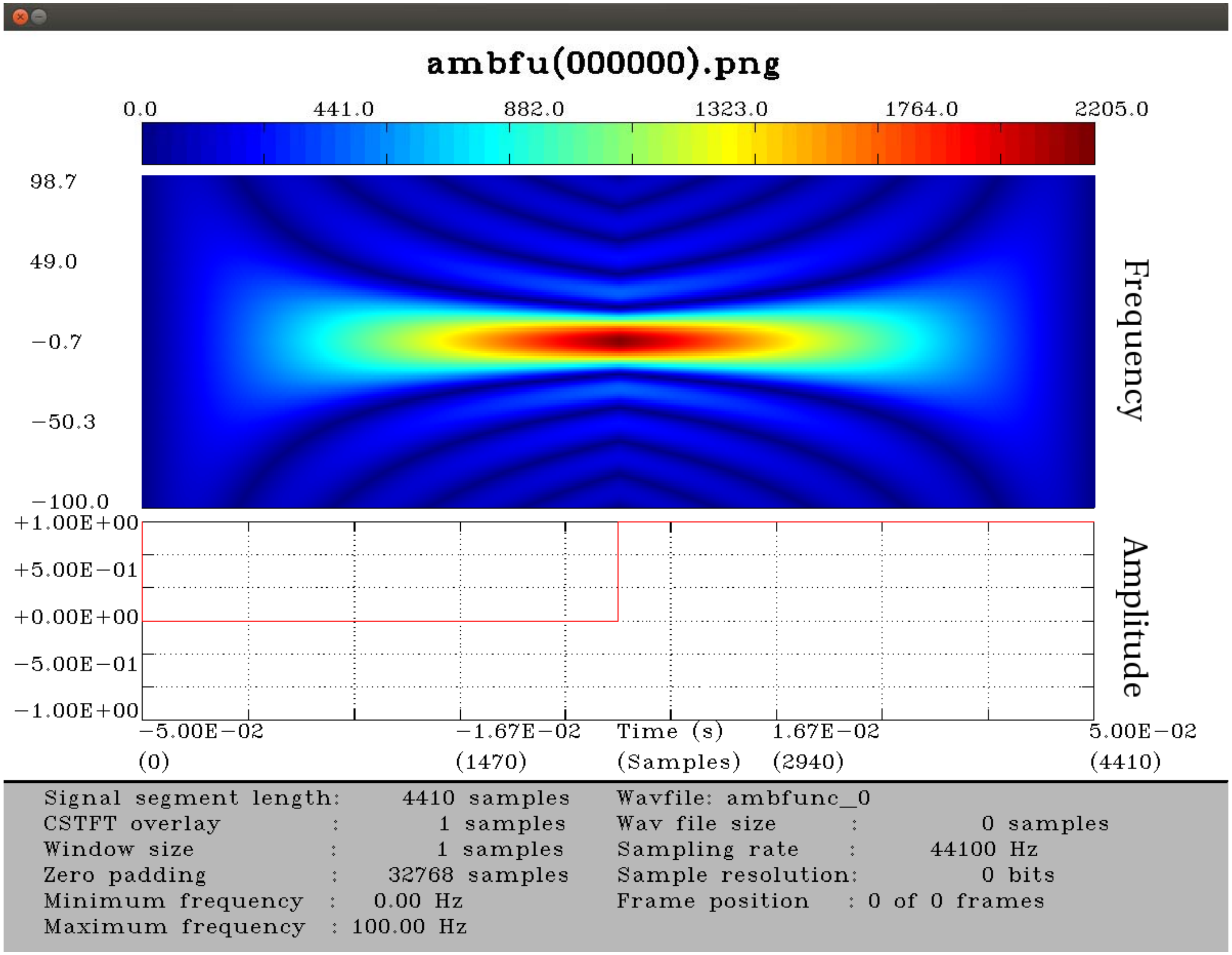
Research contribution

Sine



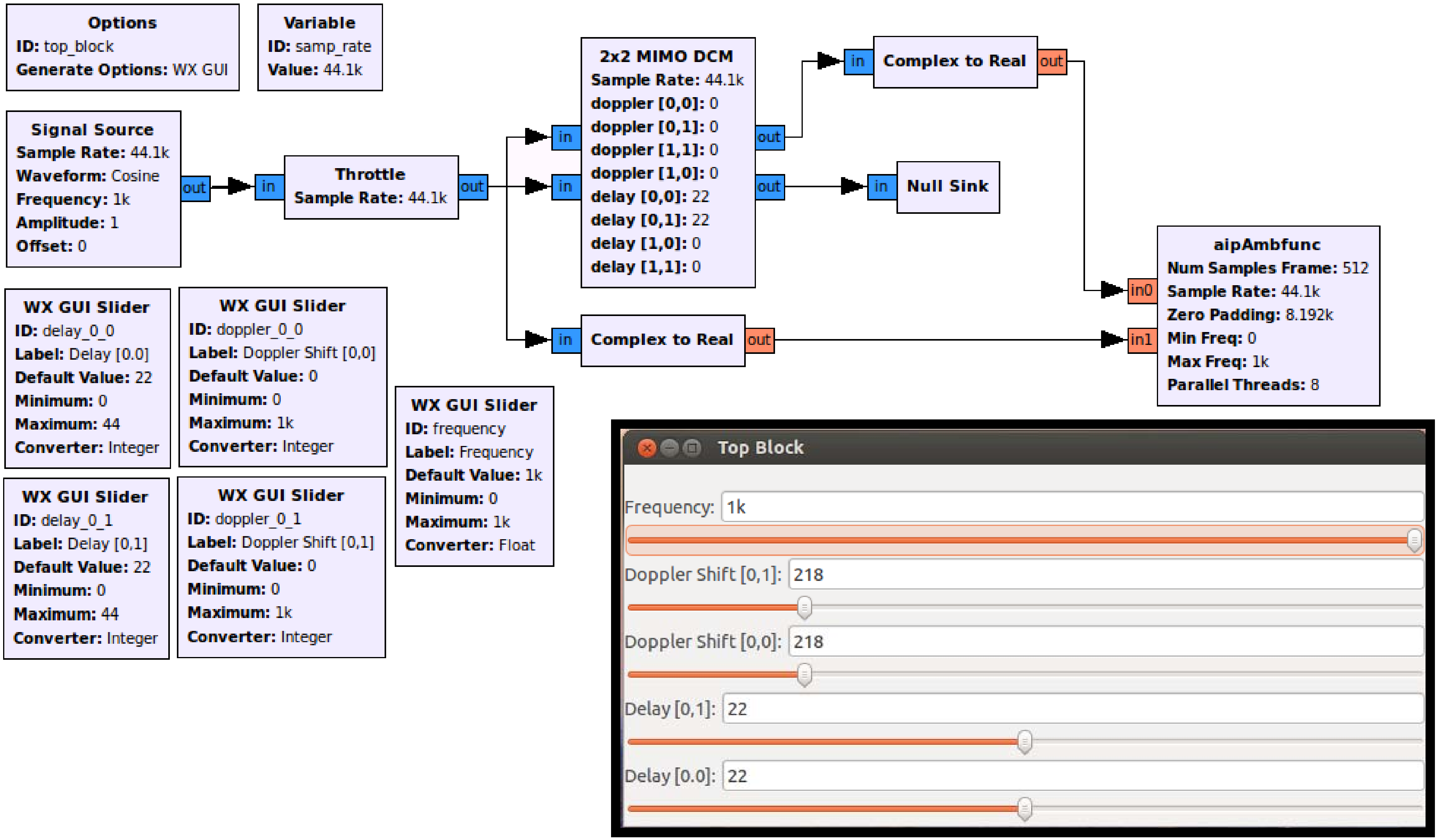
Research contribution

Square



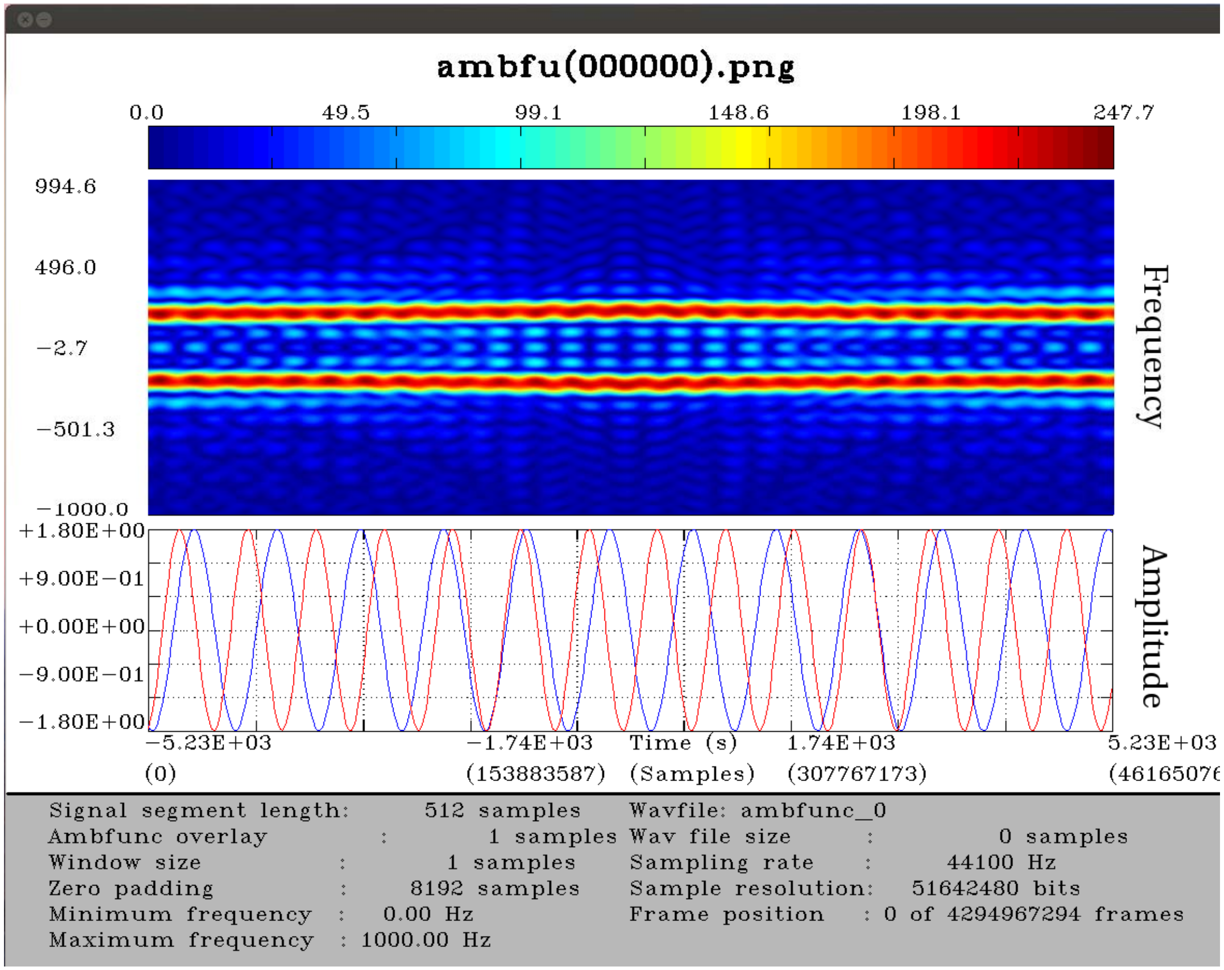
Research contribution

2x2 MIMO



Research contribution

2x2 MIMO



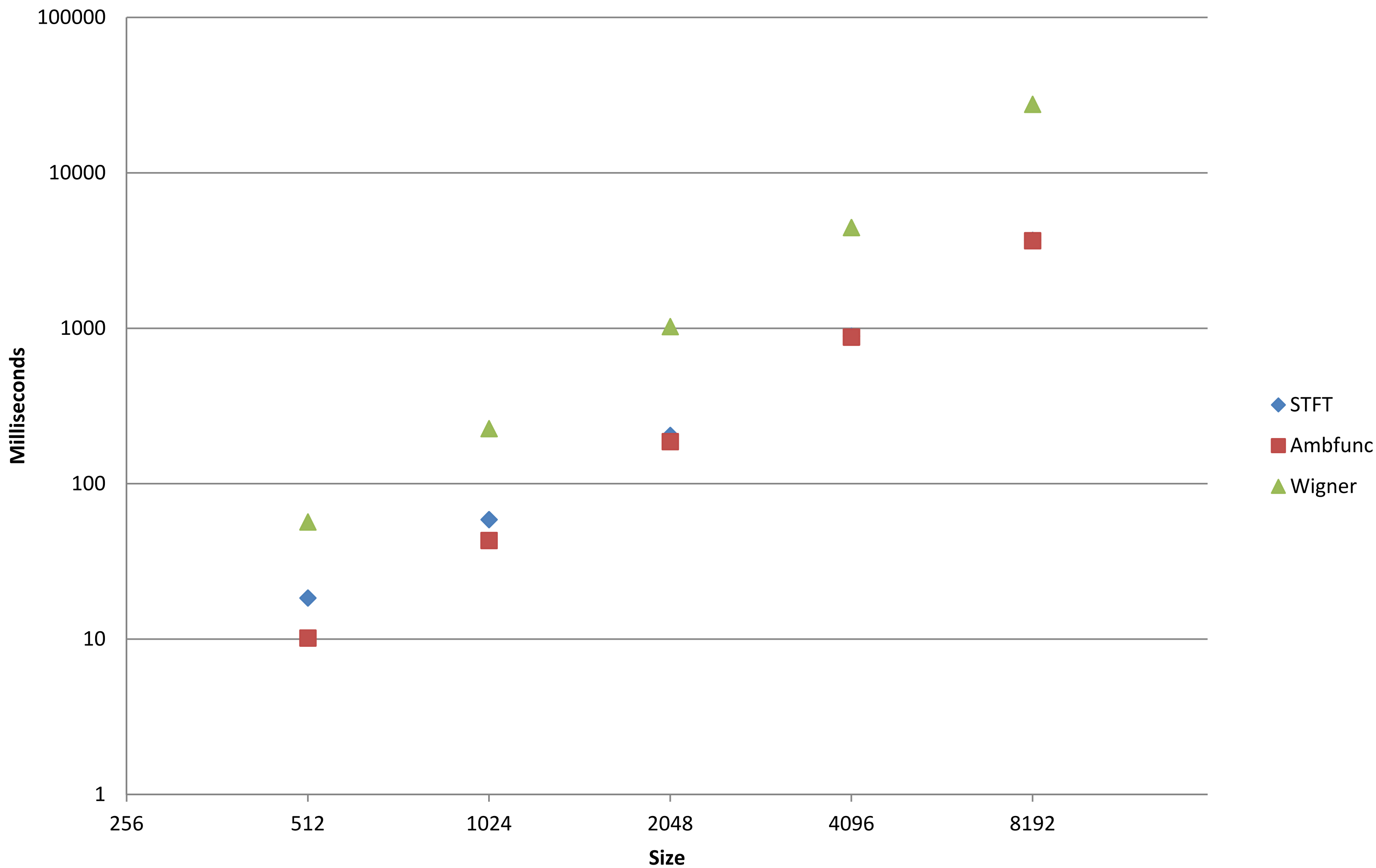
Research contribution

Parallelization Metrics

- Dell 1520
- Intel Core2 T5850
  - 2.16 GHz
  - 4MB Cache
  - 2 Cores, 2 Threads
- 4GB@667MHz
- Linux 2.32

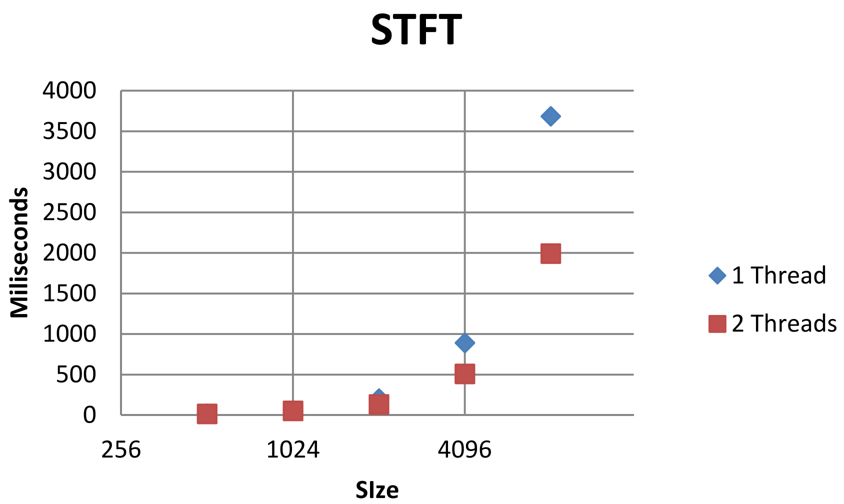
- Samsung RC512
- Intel Core i7 2630QM
  - 2.0 -> 2.9 GHz
  - 6MB Cache
  - 4Cores, 8 Threads
- 6GB@1333MHz
- Linux 3.2

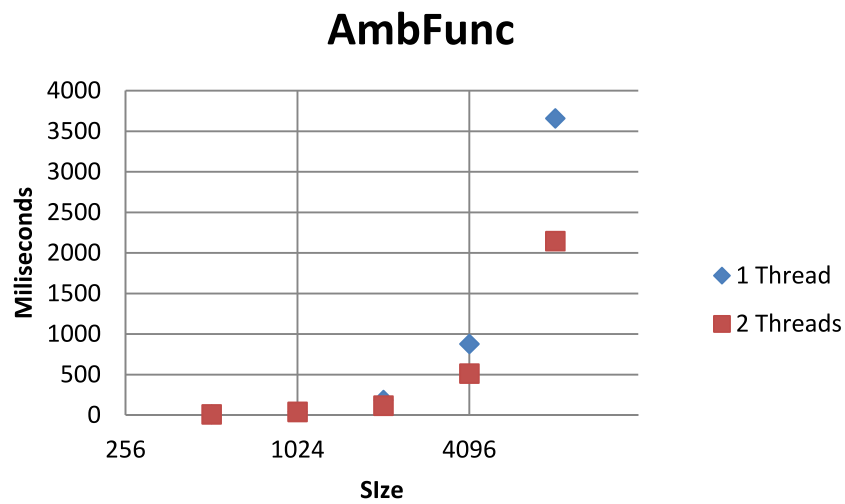
Dell 1520

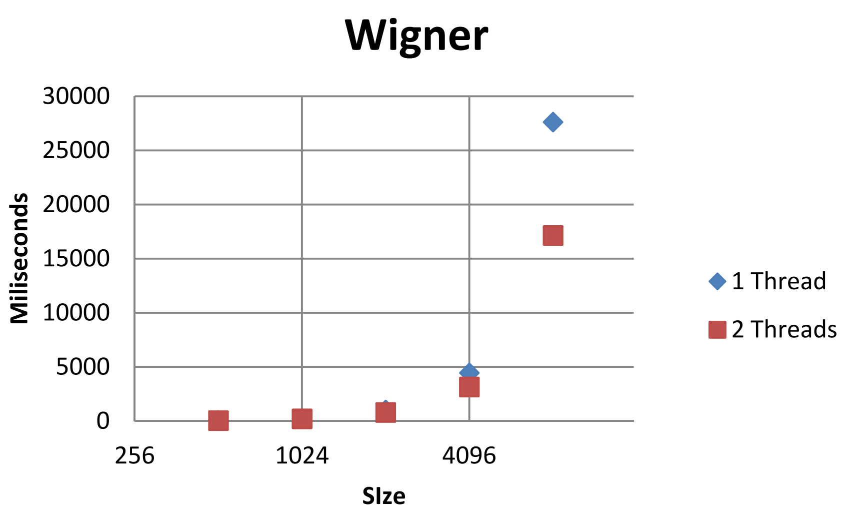


Research contribution

Dell 1520

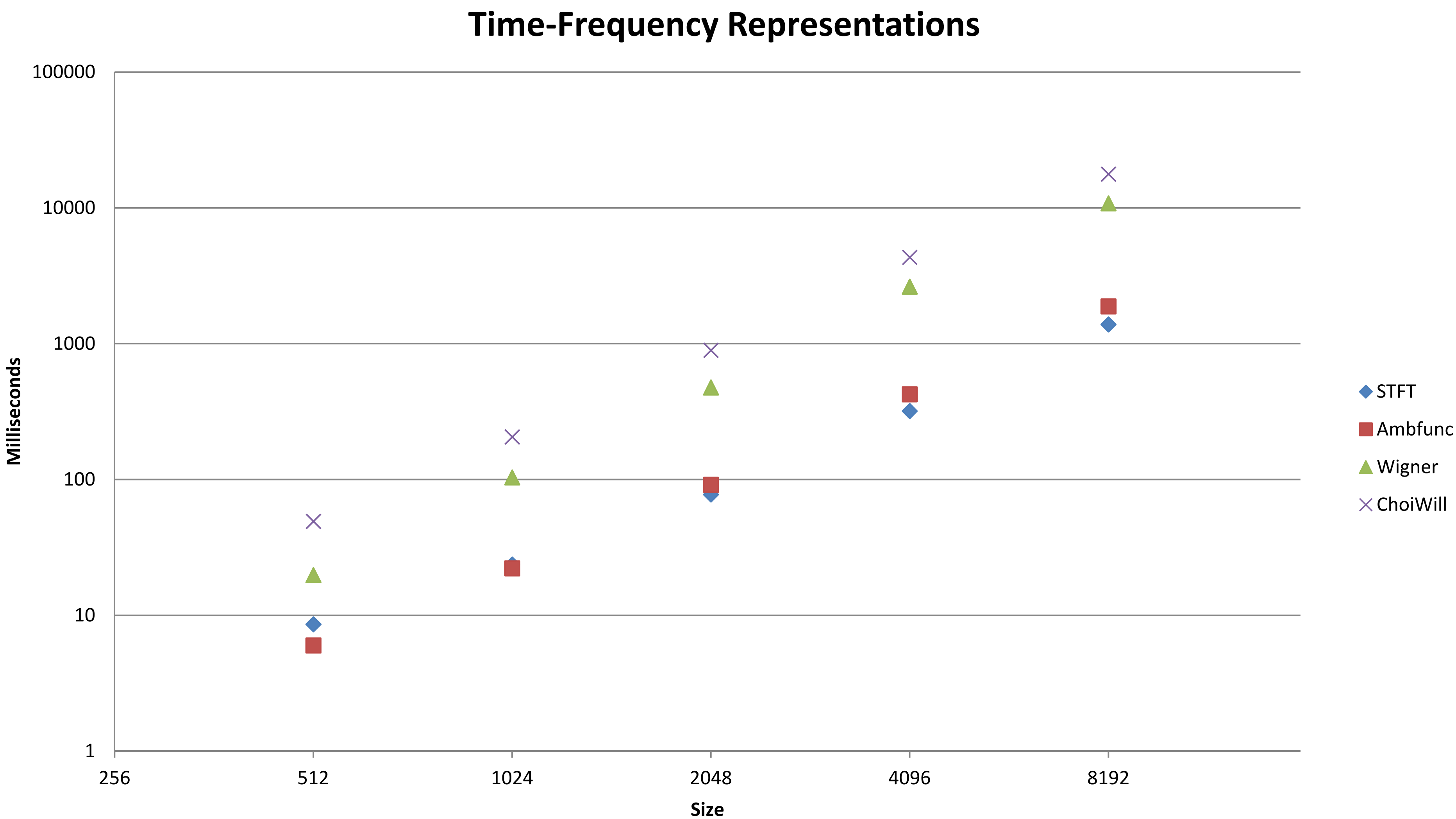






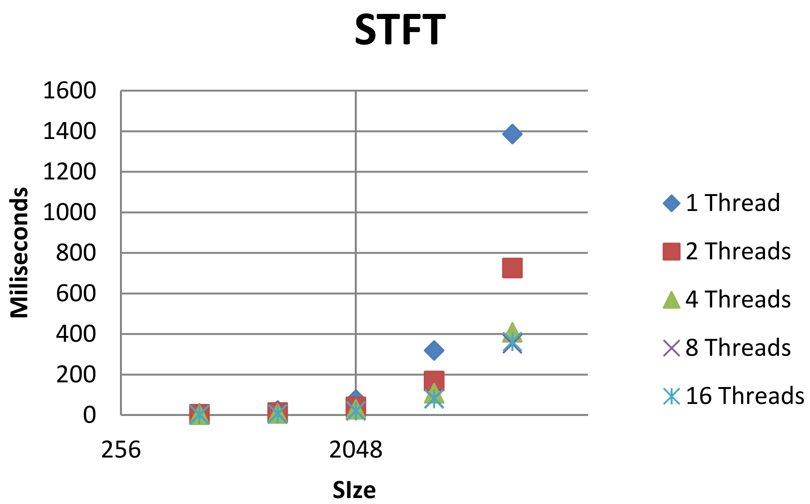
Research contribution

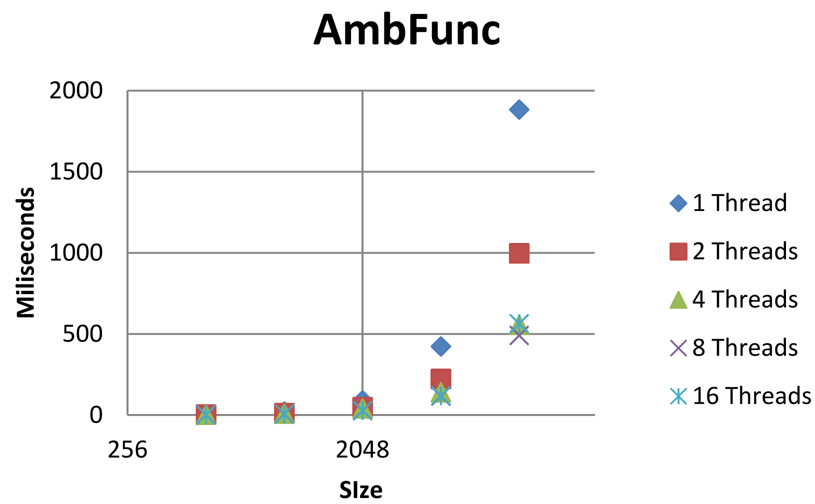
Samsung RC512

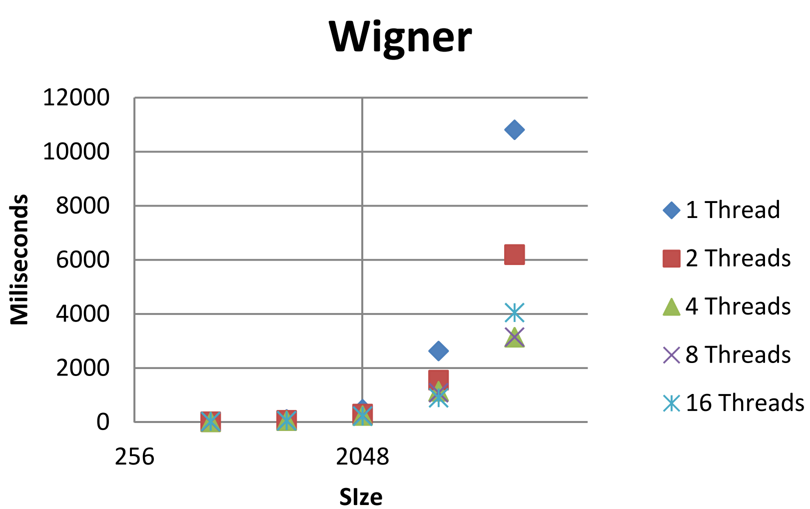


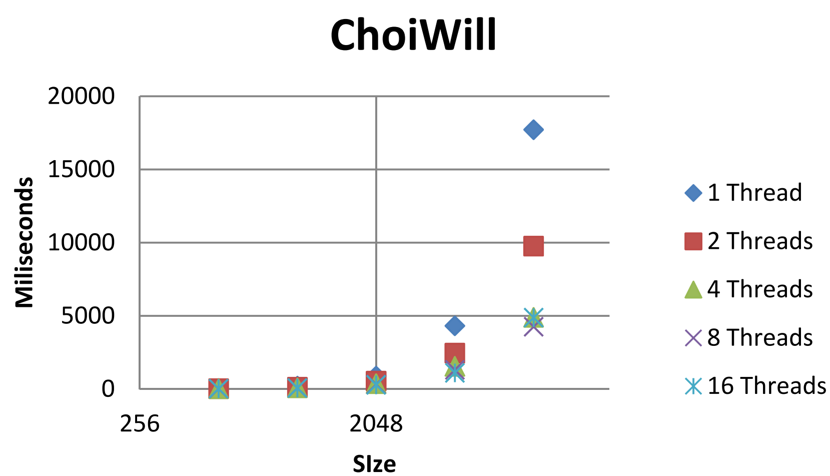
Research contribution

Samsung RC512





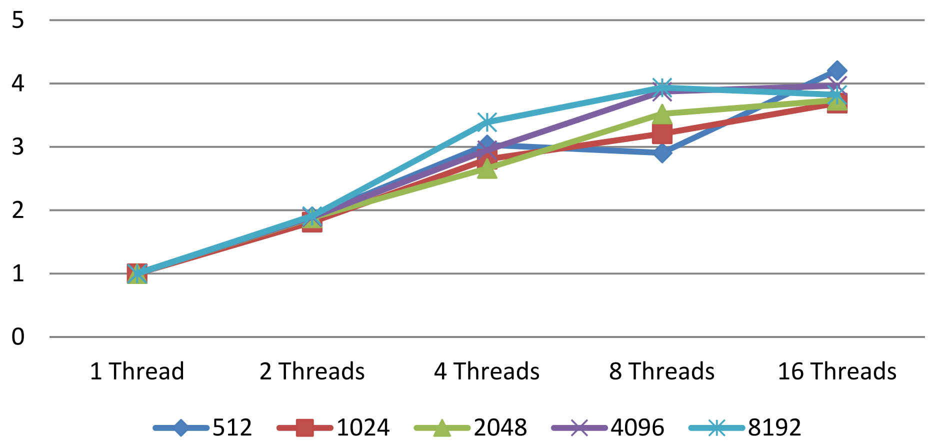




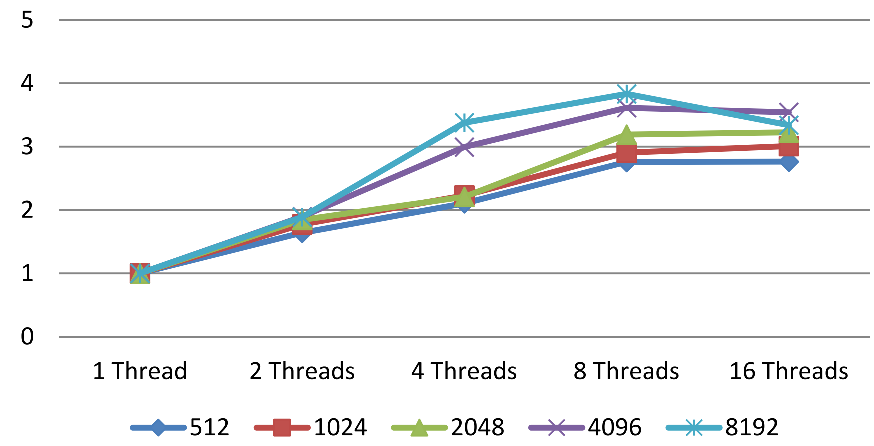
Research contribution

Speed Ups

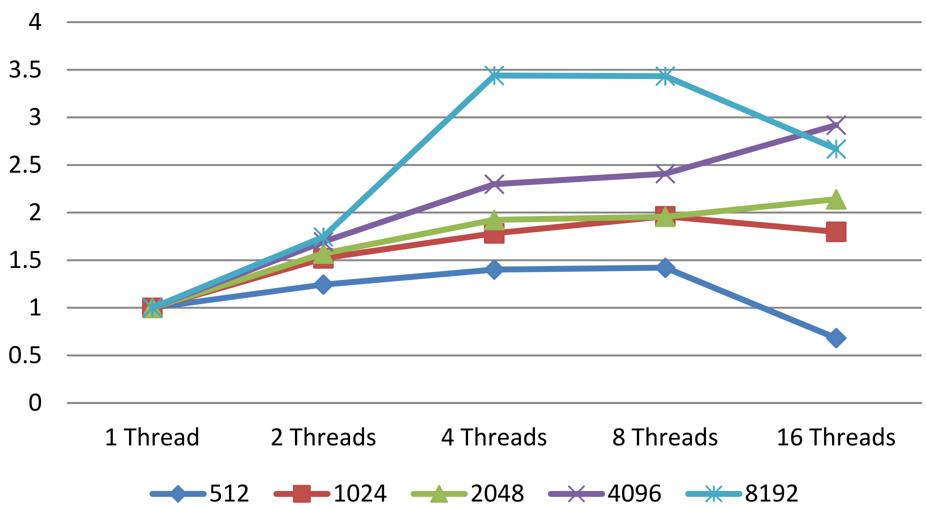
Speed Up STFT



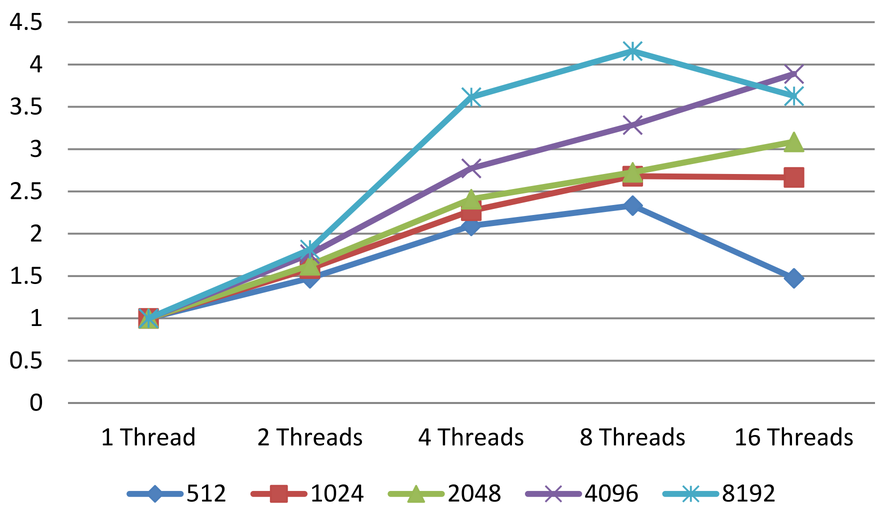
Speed Up AmbFunc



Speed Up Wigner



Speed Up ChoiWill



Conclusions

- A MIMO Modeling framework was presented using a software defined radio paradigm into GNUradio.
- Several time-frequency blocks were designed into GNUradio using the proposed framework.
- A testing workflow was defined using the models and algorithms developed.
- Several works requiring a time-frequency representation can take advantage of the developed framework.

Contributions

- MIMO Channel Modeling
- SIRLAB Web Integration
- SIRLAB Android Integration
- STFT, AmbFunc, Wigner, Choi-Williams SDR Implementation
- MIMO Channel SDR Implementation
- Applications: Channel Surveillance
- Publications:
  - IEEE LASCAS
  - IBERSENSOR

Future work

- Channel Estimation
- Develop Input Signals With the Framework
- DSP, FPGA Integration
- ***NetSig Integration***