Iris Software Radio Architecture

Paul Sutton

2nd February 2014

FOSDEM

SRS Software Radio Systems

CTVR the telecommunications research centre

Outline

- Iris Overview
- Iris Architecture
- Controllers

Iris Overview

What is Iris?

Iris Overview

What is Iris?

A Software Radio Architecture

Iris Overview

What is Iris?

Reconfigurable

A Software Radio Architecture

Iris Overview

What is Iris?

Reconfigurable

A Software Radio Architecture

GPP – Based (primarily)

Iris Overview

What is Iris?

Reconfigurable

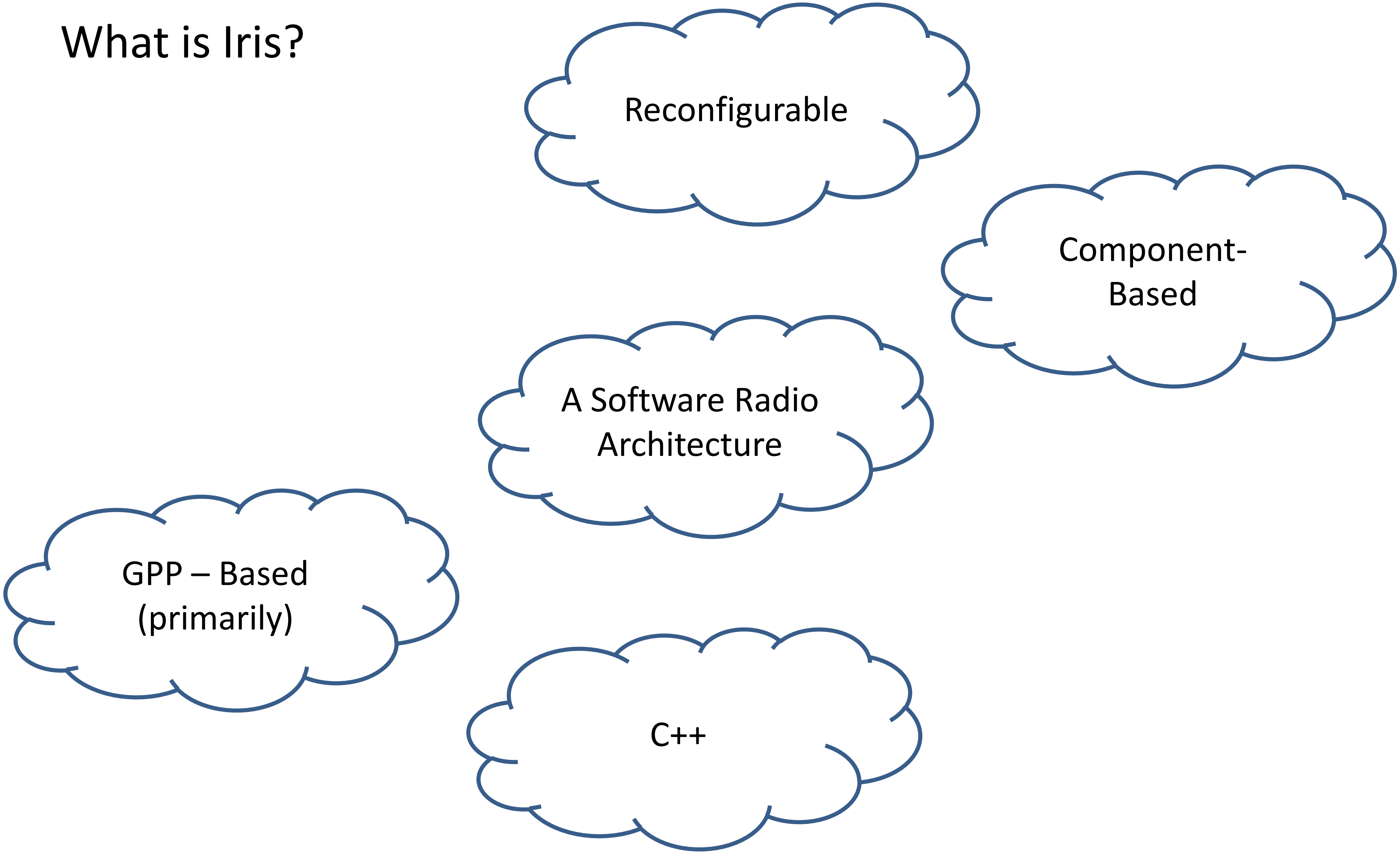
Component-Based

A Software Radio Architecture

GPP – Based (primarily)

Iris Overview

What is Iris?



Reconfigurable

Component-Based

A Software Radio Architecture

GPP – Based (primarily)

C++

Iris Overview

What is Iris?

Reconfigurable

Component-Based

A Software Radio Architecture

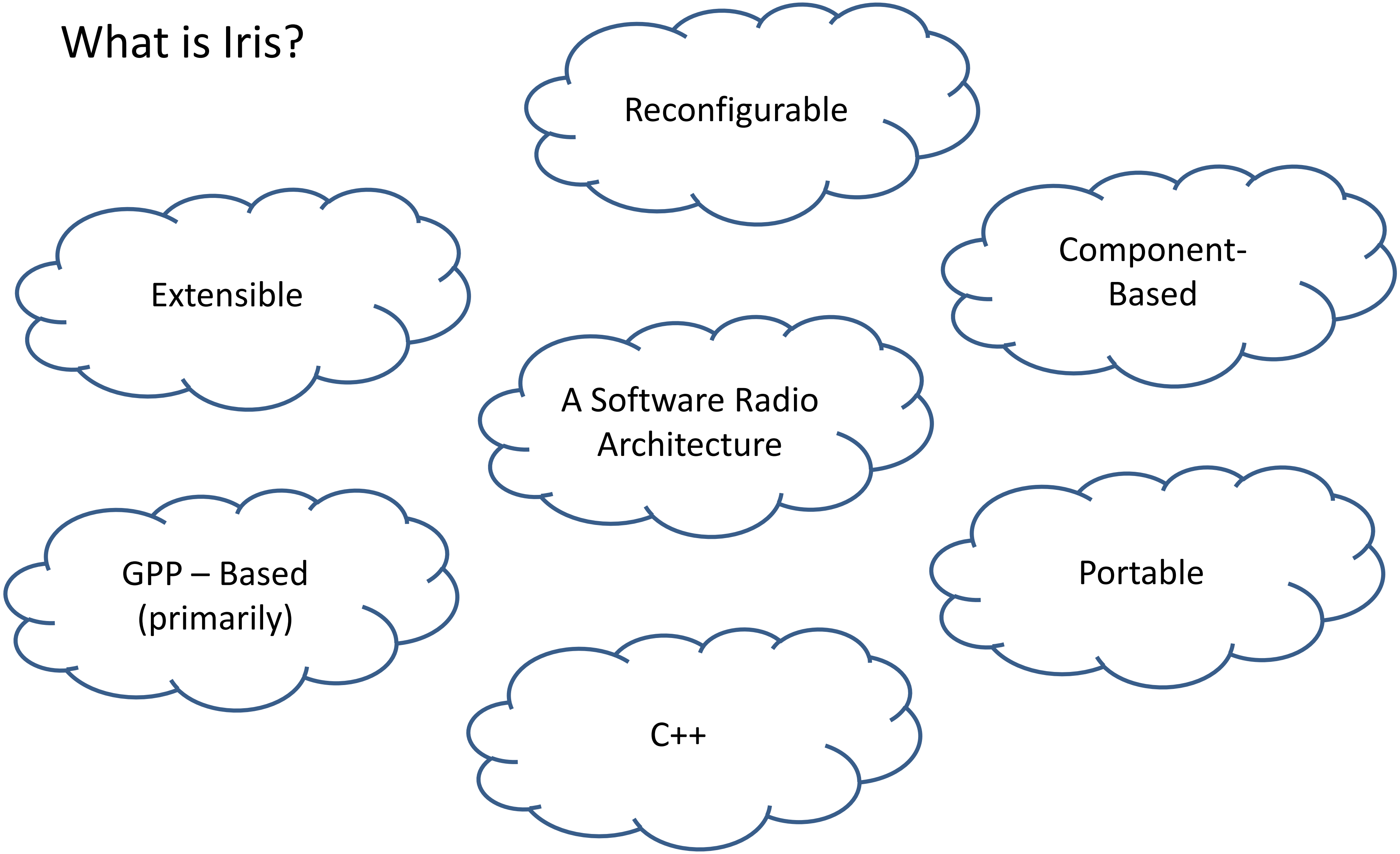
GPP – Based (primarily)

Portable

C++

Iris Overview

What is Iris?



Reconfigurable

Extensible

Component-Based

A Software Radio Architecture

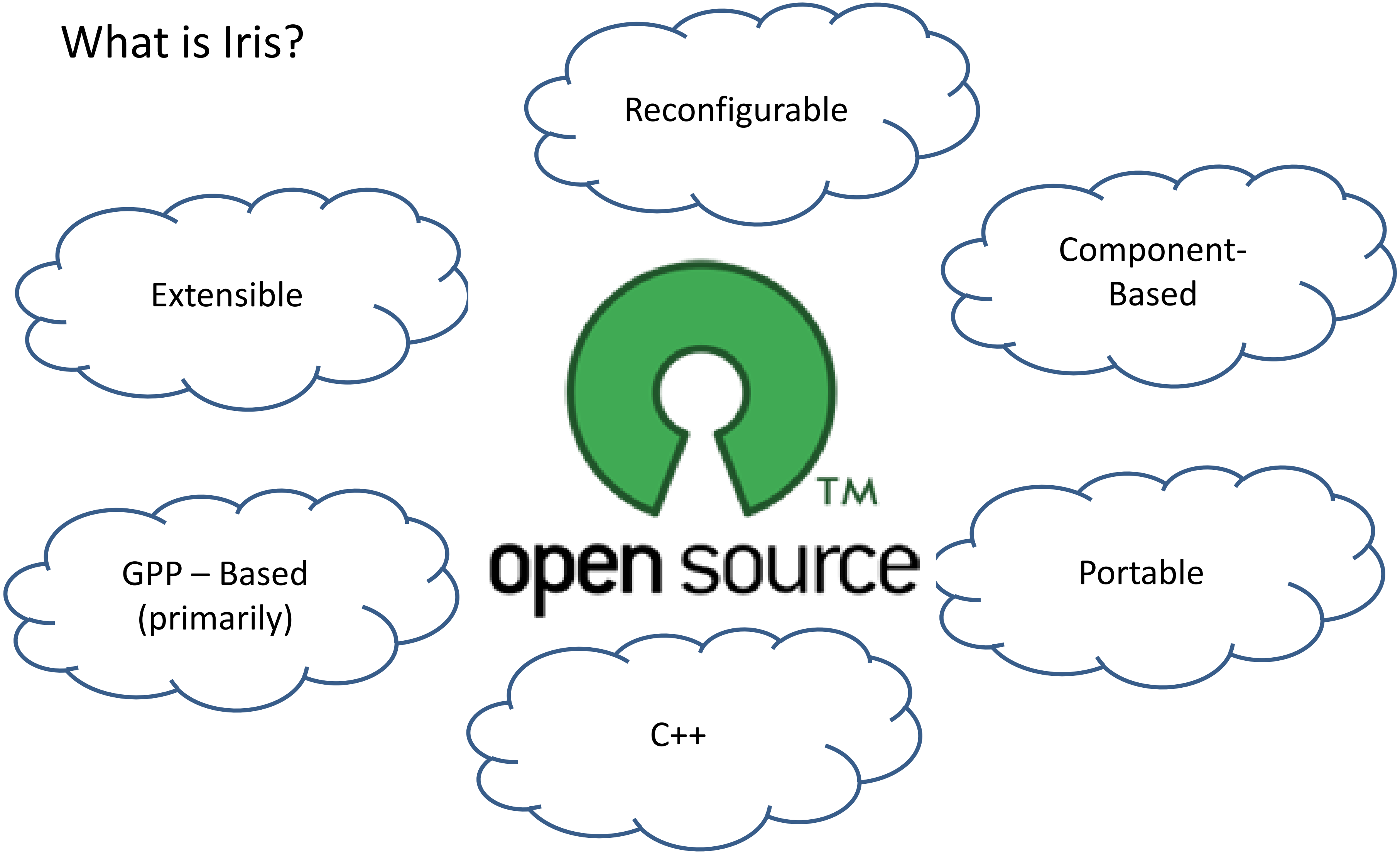
GPP – Based (primarily)

Portable

C++

Iris Overview

What is Iris?



Reconfigurable

Extensible

Component-Based

GPP – Based (primarily)

Portable

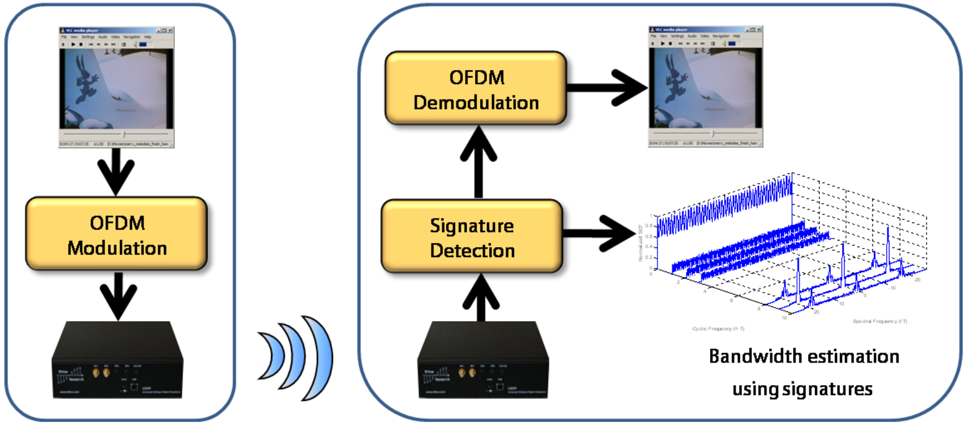
C++

Iris Overview

What can I do with Iris?

Iris Overview

What can I do with Iris?



OFDM Modulation

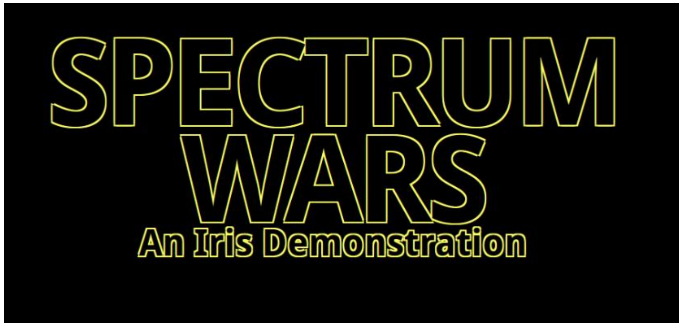
OFDM Demodulation

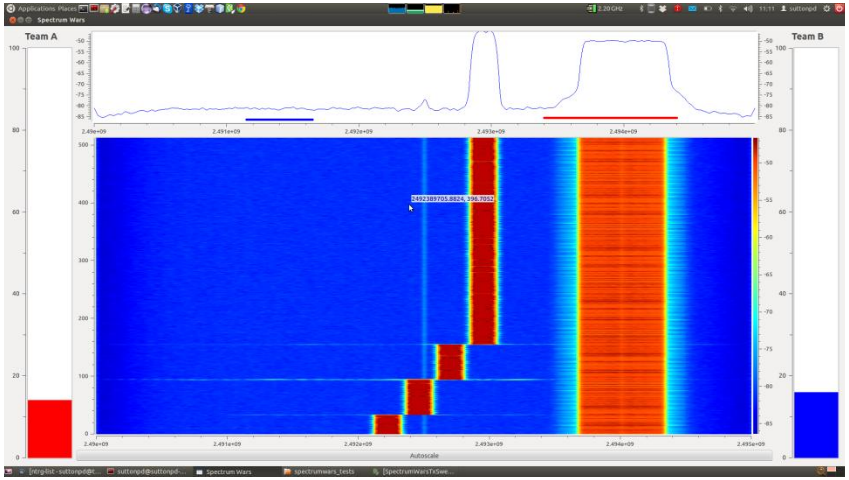
Signature Detection

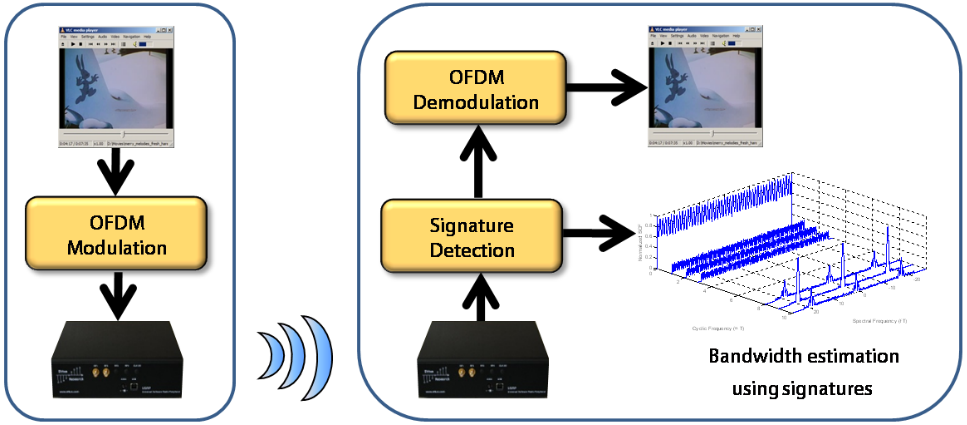
Bandwidth estimation using signatures

Iris Overview

What can I do with Iris?







OFDM Modulation

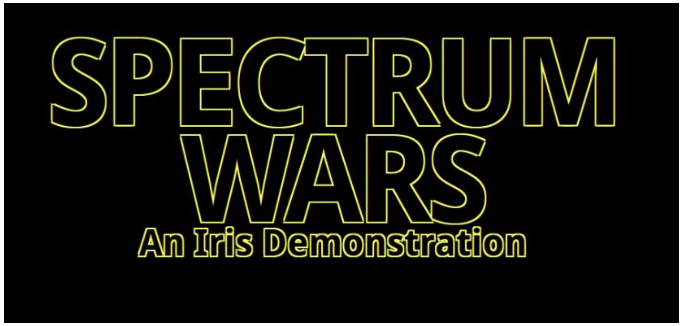
OFDM Demodulation

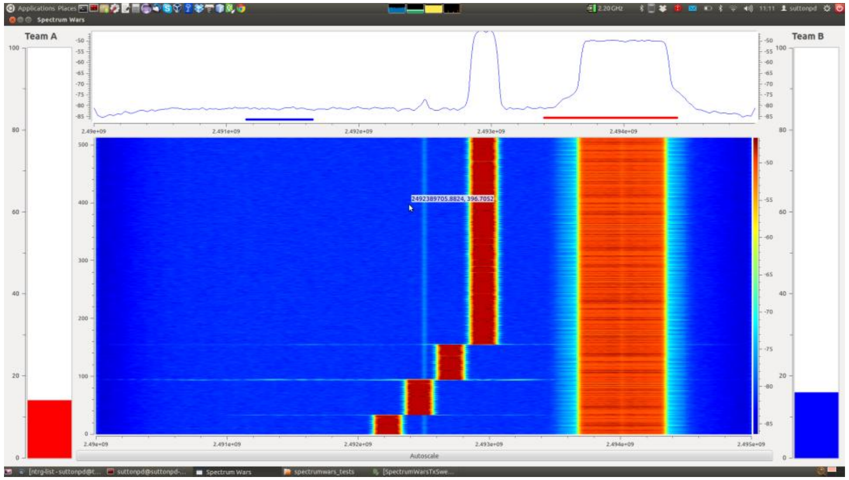
Signature Detection

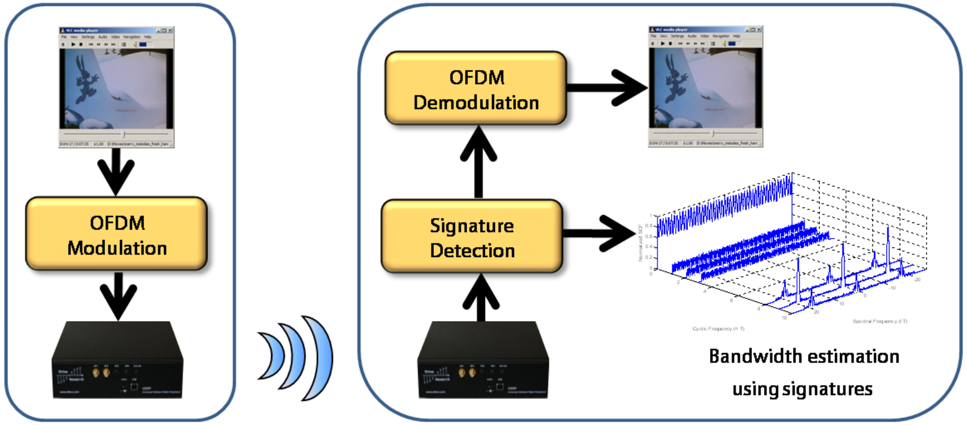
Bandwidth estimation using signatures

Iris Overview

What can I do with Iris?



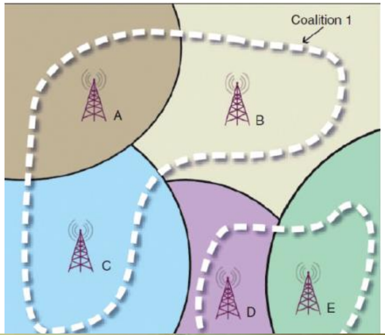



OFDM Modulation

OFDM Demodulation

Signature Detection

Bandwidth estimation using signatures

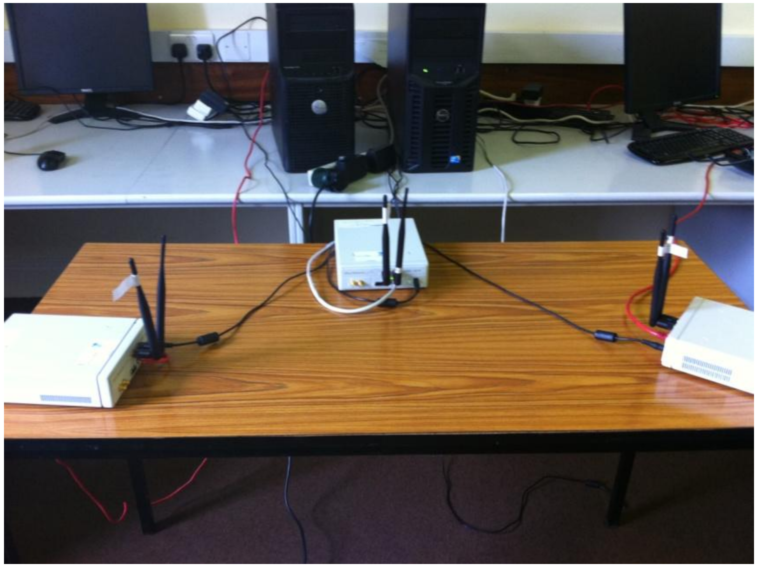


SRS Software Radio Systems

CTVR the telecommunications research centre

Iris Overview

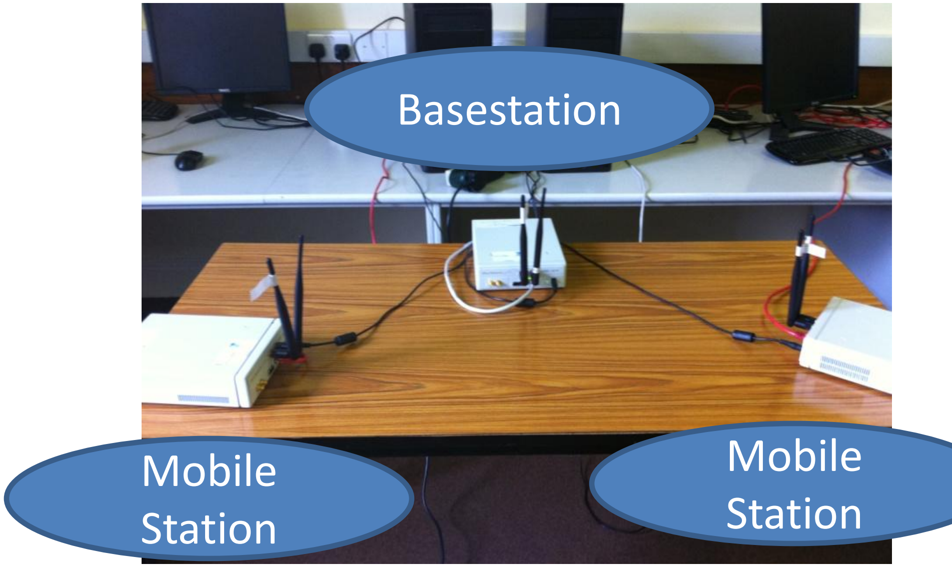
What can I do with Iris?



- Jacek Kibilda
- COST Short-Term Scientific Mission
- 2 weeks (no prior knowledge of Iris)
- DSA demo (primary user avoidance)

Iris Overview

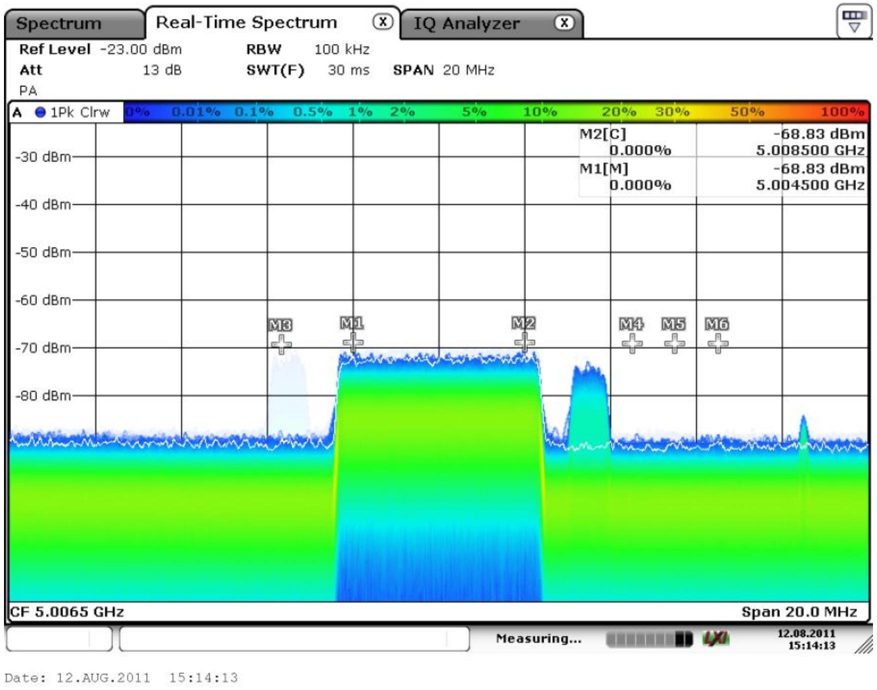
What can I do with Iris?



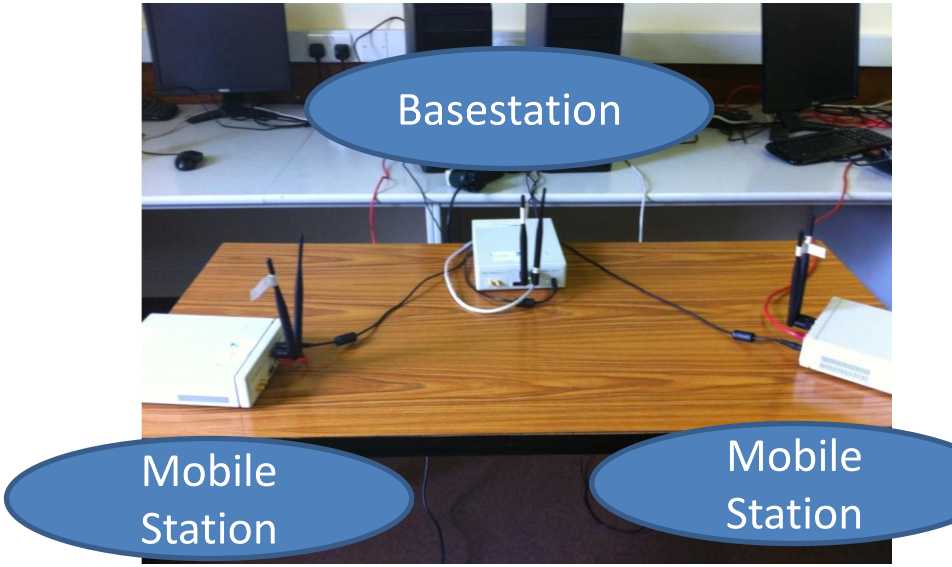
- Jacek Kibilda
- COST Short-Term Scientific Mission
- 2 weeks (no prior knowledge of Iris)
- DSA demo (primary user avoidance)

Iris Overview

What can I do with Iris?



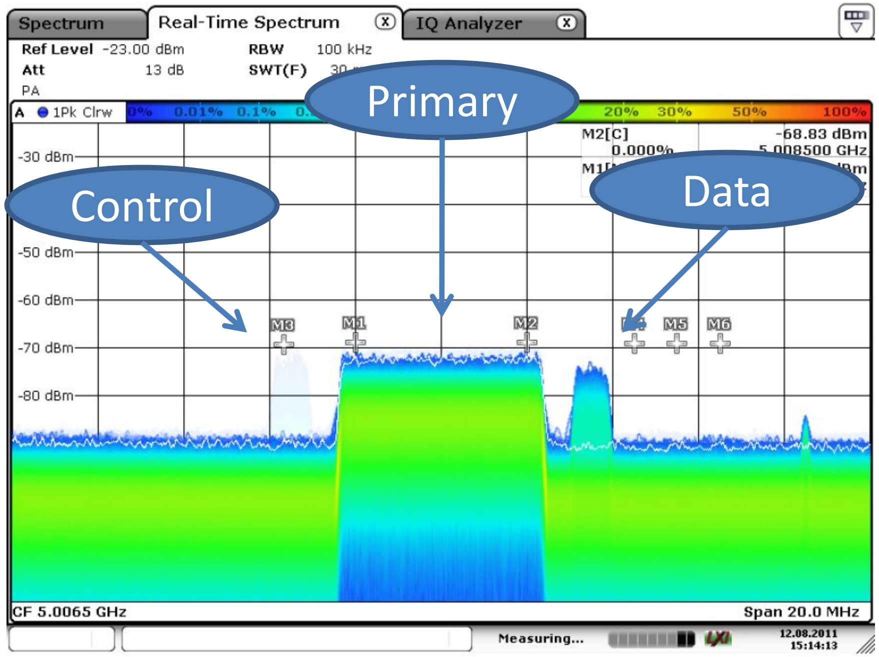
```
Ref Level  -23.00 dBm     RBW        100 kHz
Att            13 dB      SWT(F)      30 ms     SPAN  20 MHz
PA

M2[C]                     -68.83 dBm
      0.000%            5.008500 GHz
M1[M]                     -68.83 dBm
      0.000%            5.004500 GHz

CF 5.0065 GHz                                     Span 20.0 MHz

12.08.2011
15:14:13

Date: 12.AUG.2011  15:14:13
```



- Jacek Kibilda
- COST Short-Term Scientific Mission
- 2 weeks (no prior knowledge of Iris)
- DSA demo (primary user avoidance)

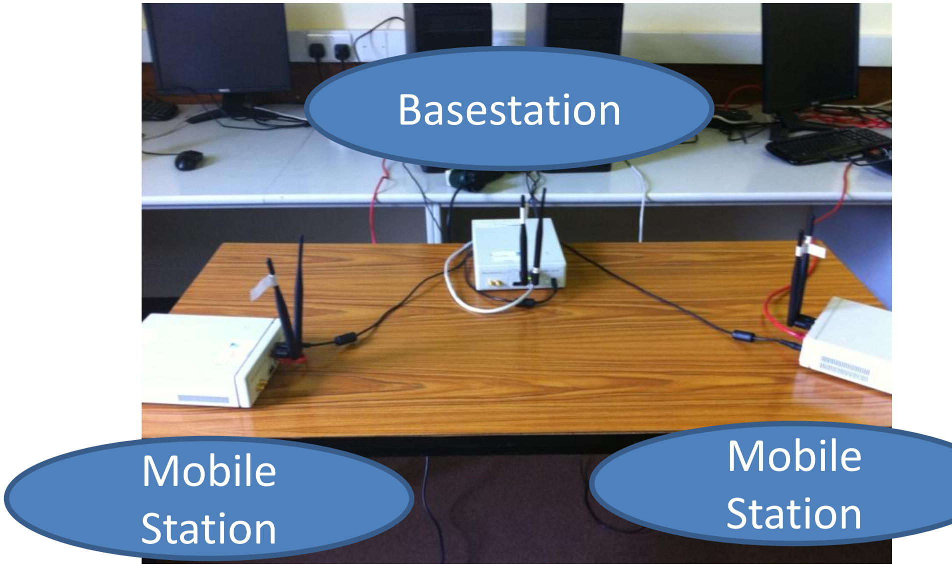
Iris Overview

What can I do with Iris?



Ref Level -23.00 dBm RBW 100 kHz
Att 13 dB SWT(F) 30
PA
A 1Pk Clrw
M2[C] -68.83 dBm
0.000% 5.008500 GHz
CF 5.0065 GHz Span 20.0 MHz
12.08.2011
15:14:13

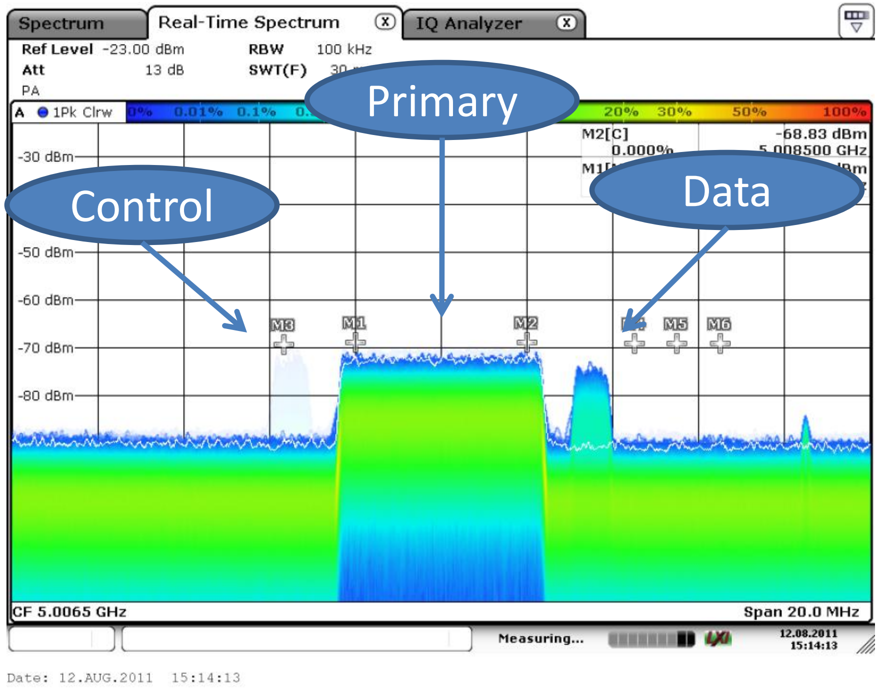
Date: 12.AUG.2011 15:14:13



- Jacek Kibilda
- COST Short-Term Scientific Mission
- 2 weeks (no prior knowledge of Iris)
- DSA demo (primary user avoidance)

Iris Overview

What can I do with Iris?



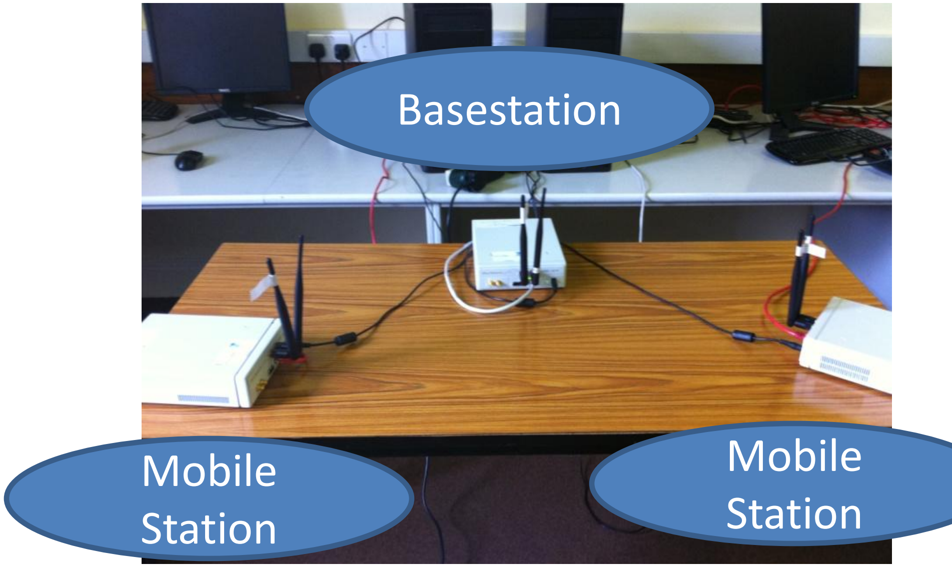
Ref Level -23.00 dBm   RBW 100 kHz
Att 13 dB   SWT(F) 30
PA

M2[C] 0.000%   -68.83 dBm   5.008500 GHz

CF 5.0065 GHz   Span 20.0 MHz

Measuring...   12.08.2011 15:14:13

Date: 12.AUG.2011 15:14:13



- Jacek Kibilda
- COST Short-Term Scientific Mission
- 2 weeks (no prior knowledge of Iris)
- DSA demo (primary user avoidance)

http://ledoyle.wordpress.com/2011/08/14/speedy-creation-of-a-cognitive-radio-demo/

Iris Architecture - The Basics

The Basics...

- A GPP-based software radio architecture
  - Fundamental block is the component

Iris Architecture - The Basics

The Basics...

- A GPP-based software radio architecture
  - Fundamental block is the component

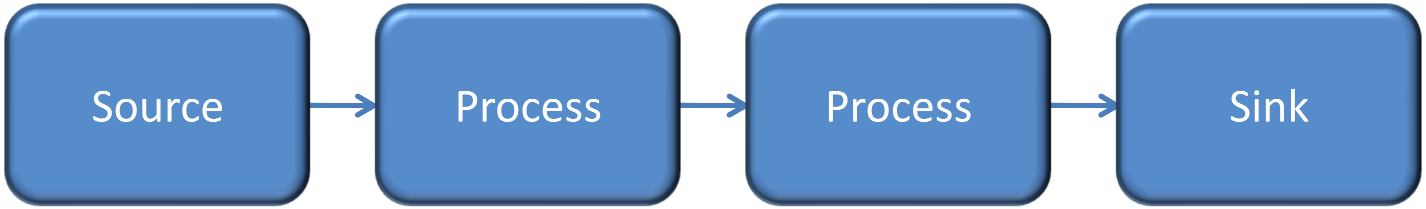
- Most basic configuration :
  - A source component
  - A sink component
  - Some processing components

Iris Architecture - The Basics

The Basics...

- A GPP-based software radio architecture
  - Fundamental block is the component

- Most basic configuration :
  - A source component
  - A sink component
  - Some processing components

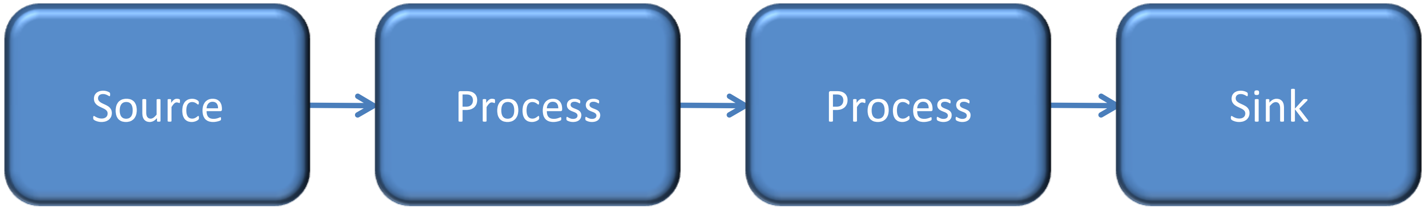


Iris Architecture - The Basics

The Basics...

- A GPP-based software radio architecture
  - Fundamental block is the component

- Most basic configuration :
  - A source component
  - A sink component
  - Some processing components



Source

Process

Process

Sink

- XML document describes radio structure

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

        <engine name="phyengine1" class="phyengine">

                <component name="filerawreader1" class="filerawreader">
                        <parameter name="filename" value="testdata.txt"/>
                        <port name="output1" class="output"/>
                </component>

                <component name="ofdmmod1" class="ofdmmodulator">
                        <port name="input1" class="input"/>
                        <port name="output1" class="output"/>
                </component>

                <component name="signalscaler1" class="signalscaler">
                        <port name="input1" class="input"/>
                        <port name="output1" class="output"/>
                </component>

                <component name="usrptx1" class="usrptx">
                        <parameter name="frequency" value="5010000000"/>
                        <parameter name="rate" value="1000000"/>
                        <port name="input1" class="input"/>
                </component>

        </engine>

        <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
        <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
        <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

        <engine name="phyengine1" class="phyengine">

                <component name="filerawreader1" class="filerawreader">
                        <parameter name="filename" value="testdata.txt"/>
                        <port name="output1" class="output"/>
                </component>

                <component name="ofdmmod1" class="ofdmmodulator">
                        <port name="input1" class="input"/>
                        <port name="output1" class="output"/>
                </component>

                <component name="signalscaler1" class="signalscaler">
                        <port name="input1" class="input"/>
                        <port name="output1" class="output"/>
                </component>

                <component name="usrptx1" class="usrptx">
                        <parameter name="frequency" value="5010000000"/>
                        <parameter name="rate" value="1000000"/>
                        <port name="input1" class="input"/>
                </component>

        </engine>

        <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
        <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
        <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

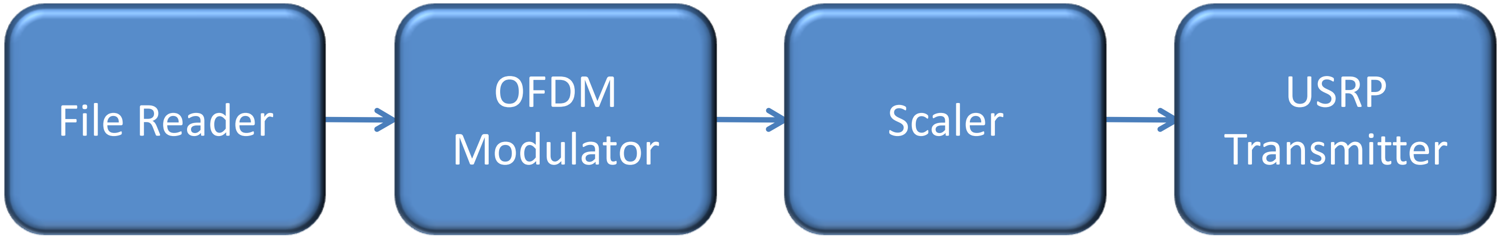
Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

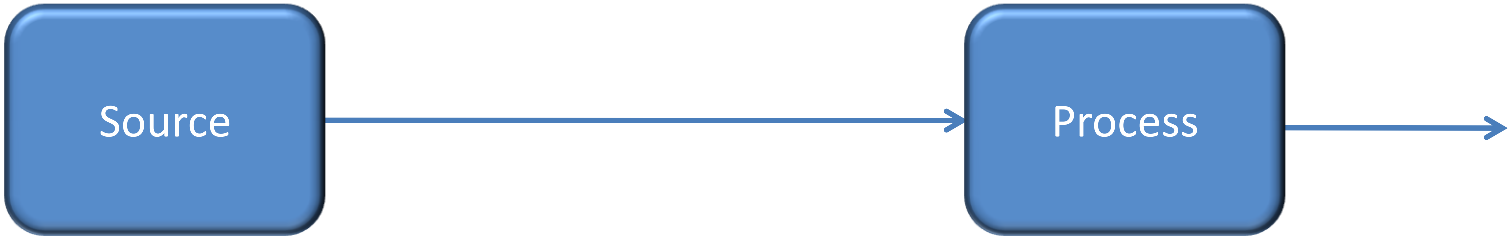
Iris Architecture - The Basics

```
<softwareradio name="Radio1">

    <engine name="phyengine1" class="phyengine">

        <component name="filerawreader1" class="filerawreader">
            <parameter name="filename" value="testdata.txt"/>
            <port name="output1" class="output"/>
        </component>

        <component name="ofdmmod1" class="ofdmmodulator">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="signalscaler1" class="signalscaler">
            <port name="input1" class="input"/>
            <port name="output1" class="output"/>
        </component>

        <component name="usrptx1" class="usrptx">
            <parameter name="frequency" value="5010000000"/>
            <parameter name="rate" value="1000000"/>
            <port name="input1" class="input"/>
        </component>

    </engine>

    <link source="filerawreader1.output1" sink="ofdmmod1.input1" />
    <link source="ofdmmod1.output1" sink="signalscaler1.input1" />
    <link source="signalscaler1.output1" sink="usrptx1.input1" />

</softwareradio>
```

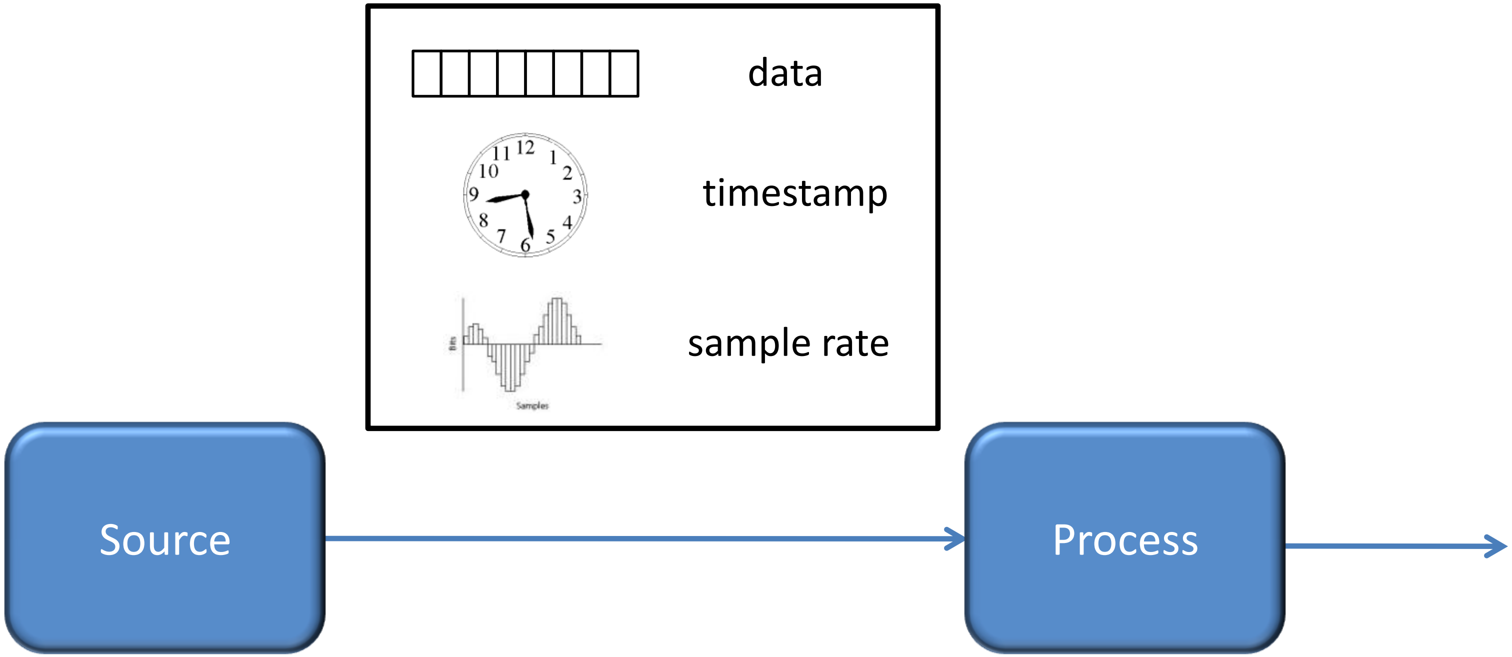
Iris Architecture - The Basics



Iris Architecture - The Basics

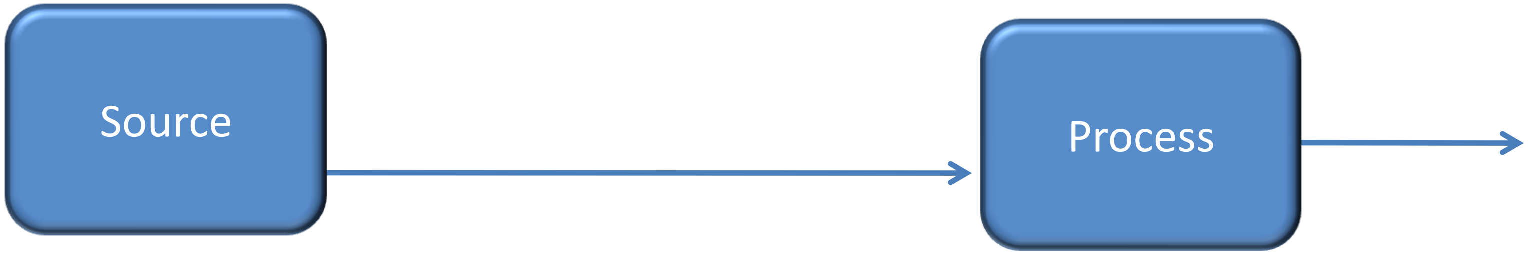


Iris Architecture - The Basics

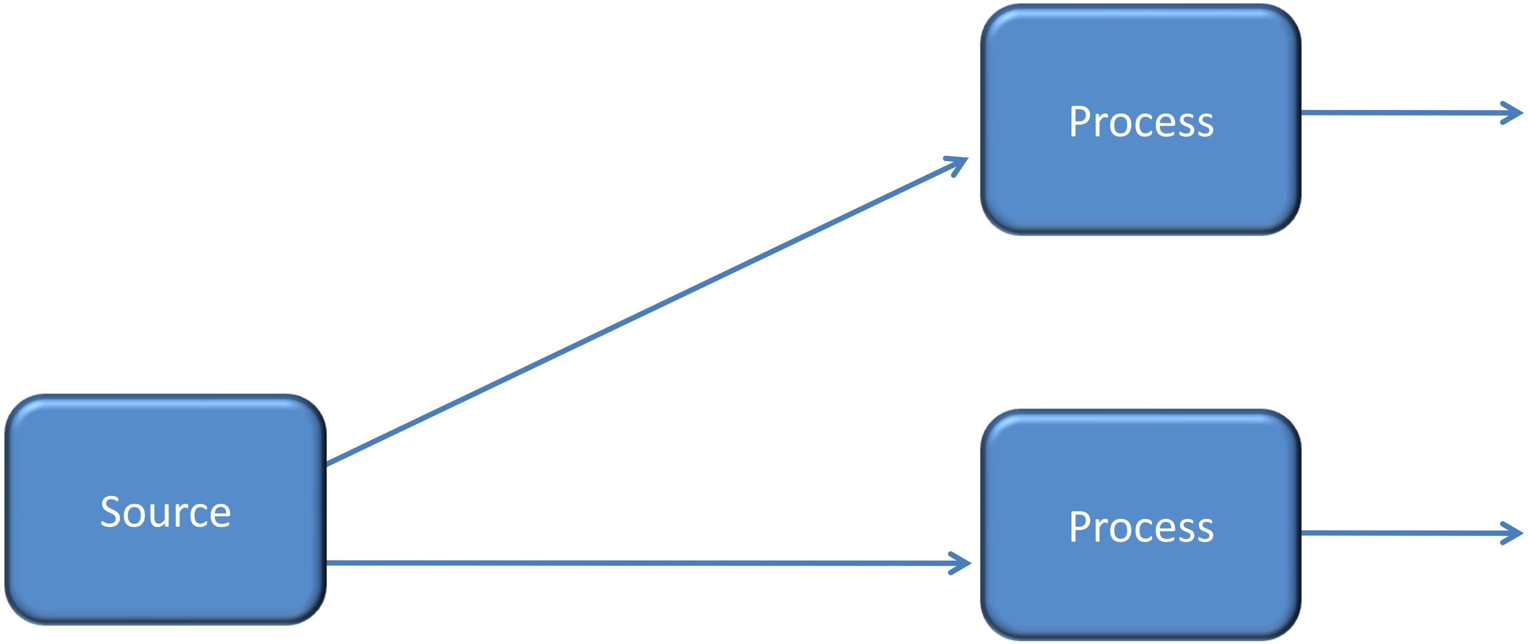


- Data is passed between components in blocks – a DataSet
- Vector of data samples
- Metadata – e.g. timestamp, sample rate

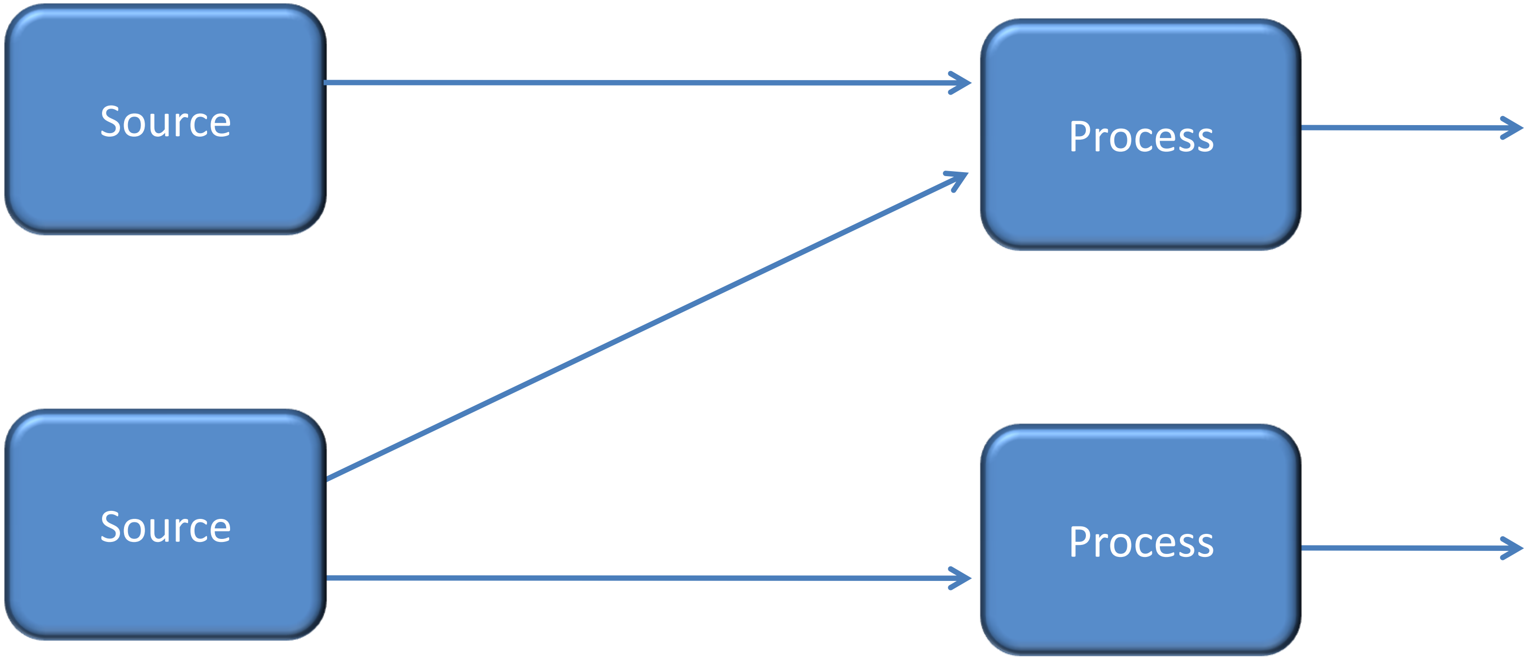
Iris Architecture - The Basics



Iris Architecture - The Basics

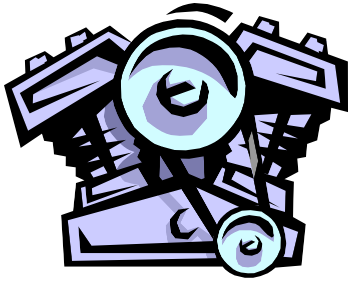


Iris Architecture - The Basics



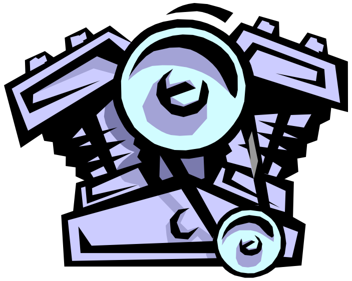
Iris Architecture - Engines

Engines



Iris Architecture - Engines

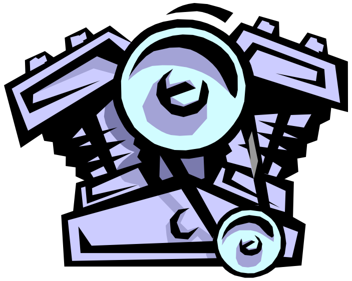
Engines



- An *engine*
  - The environment within which one more components operates
  - Defines its own data-flow mechanism
  - Defines its own reconfiguration mechanisms
  - Runs one or more of its own threads
  - Provides a clean interface for the Iris system

Iris Architecture - Engines

Engines

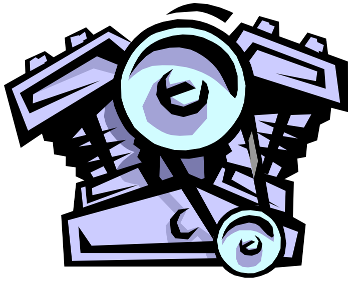


- An *engine*
  - The environment within which one more components operates
  - Defines its own data-flow mechanism
  - Defines its own reconfiguration mechanisms
  - Runs one or more of its own threads
  - Provides a clean interface for the Iris system

Executes a section of the flow graph

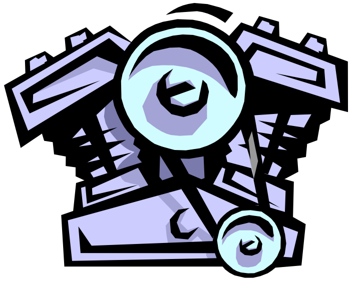
Completely up to the engine how that's done

Iris Architecture - Engines



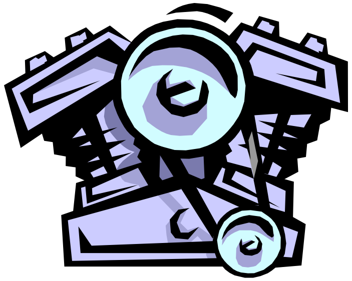
- Two engine types:
  - PHY Engine
  - Stack Engine

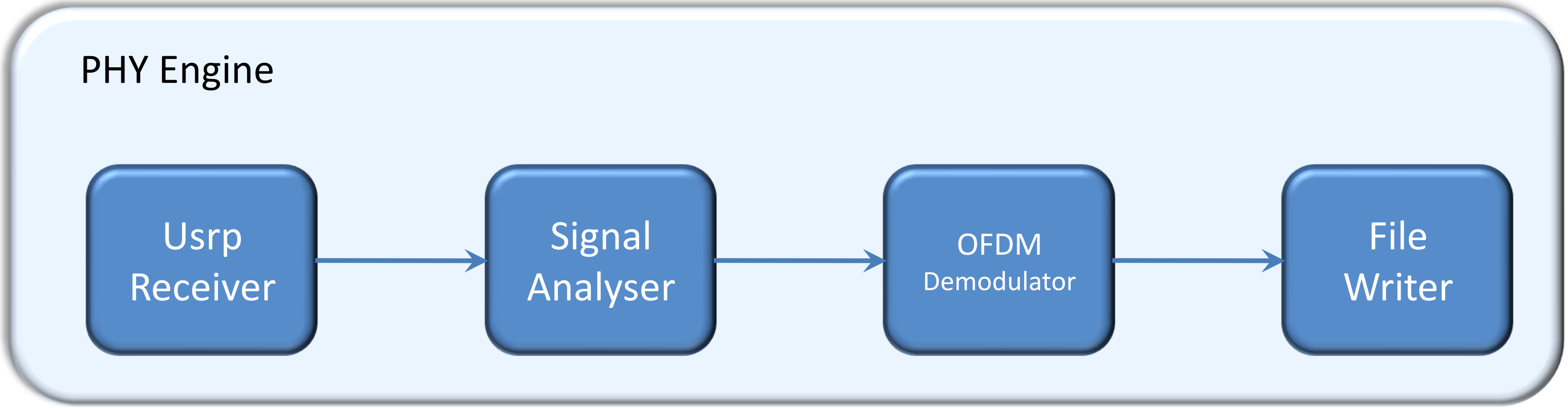
Iris Architecture - Engines



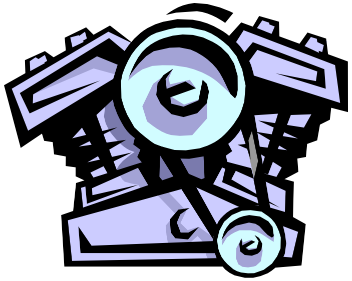
- PHY Engine
  - Maximum flexibility
  - One thread per engine
  - Data-driven execution
  - One or more components per engine
  - Multiple component inputs / outputs
  - Unidirectional data flow
  - No fixed relationship between the inputs and outputs of a component
  - Flexible blocksizes

Iris Architecture - Engines



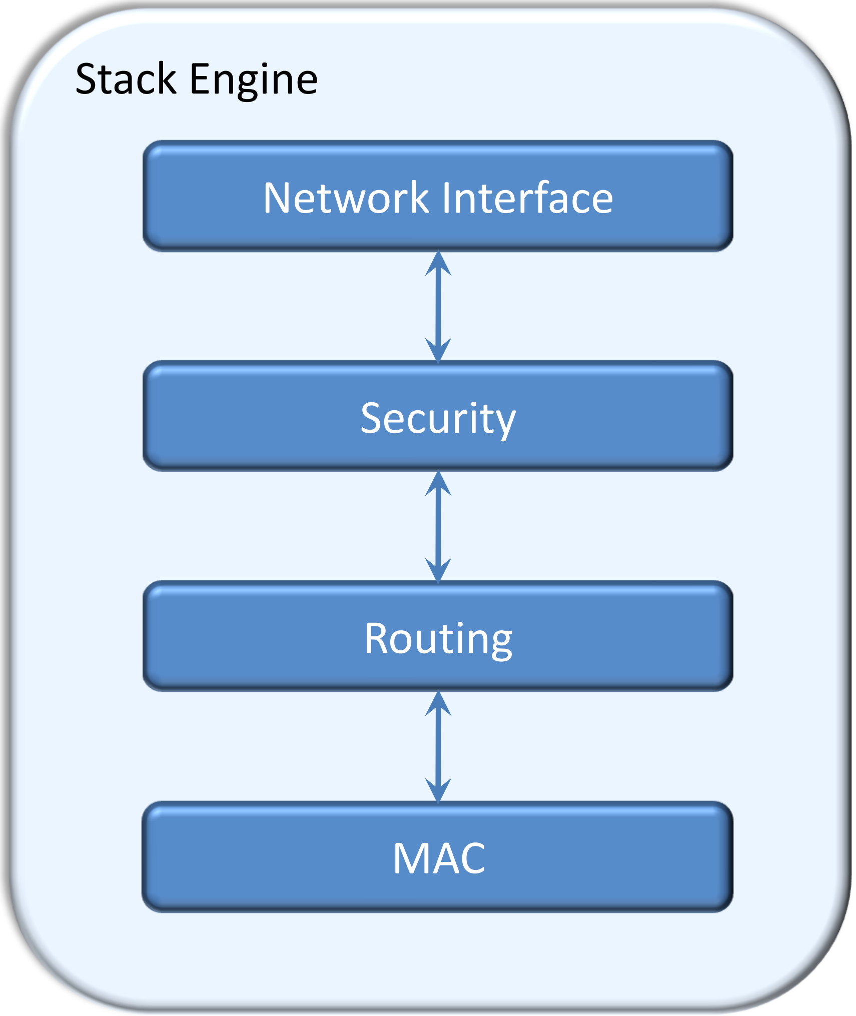


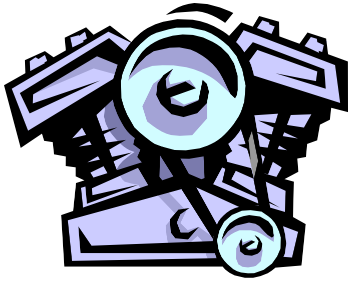
Iris Architecture - Engines



- Stack Engine
  - Network stack architecture
  - Components are layers within the stack
  - Each component runs its own thread
  - Bidirectional data flow
  - Supports e.g. MAC layer implementations

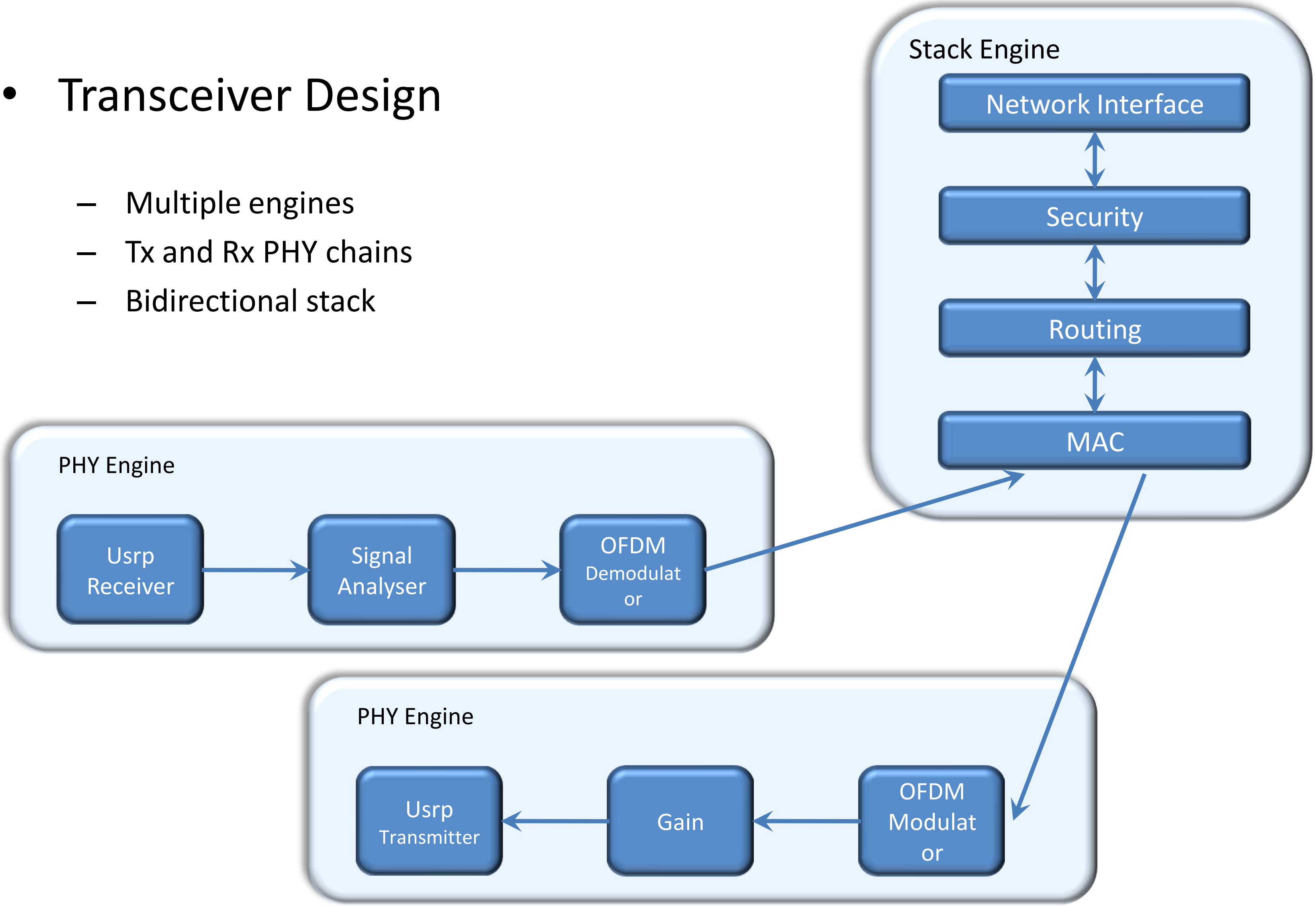
Iris Architecture - Engines



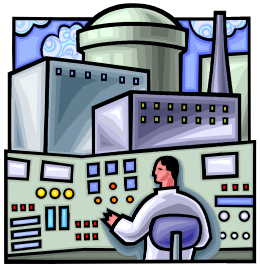


Iris Architecture - Engines

- Transceiver Design
  - Multiple engines
  - Tx and Rx PHY chains
  - Bidirectional stack

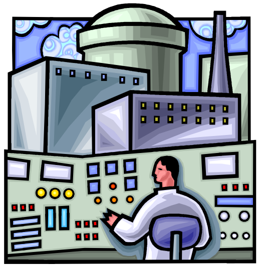


Controllers



- So far...
  - We can create a radio
  - and reconfigure it manually

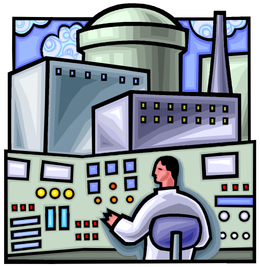
Controllers

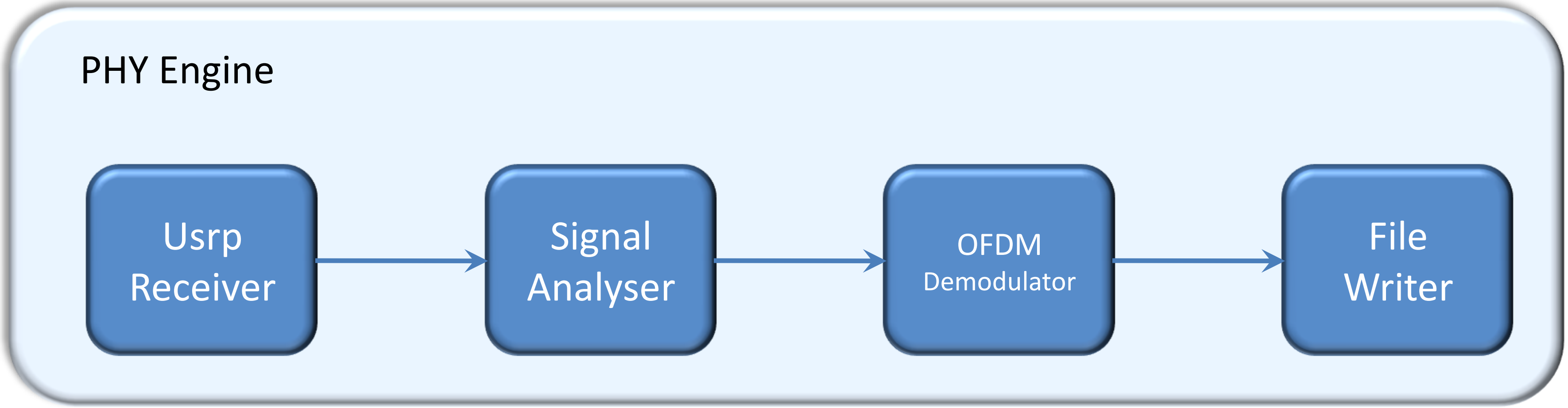


- So far...
  - We can create a radio
  - and reconfigure it manually

- How to reconfigure dynamically?

Controllers





PHY Engine

Usrp Receiver

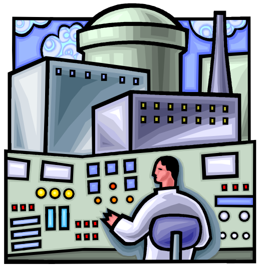
Signal Analyser

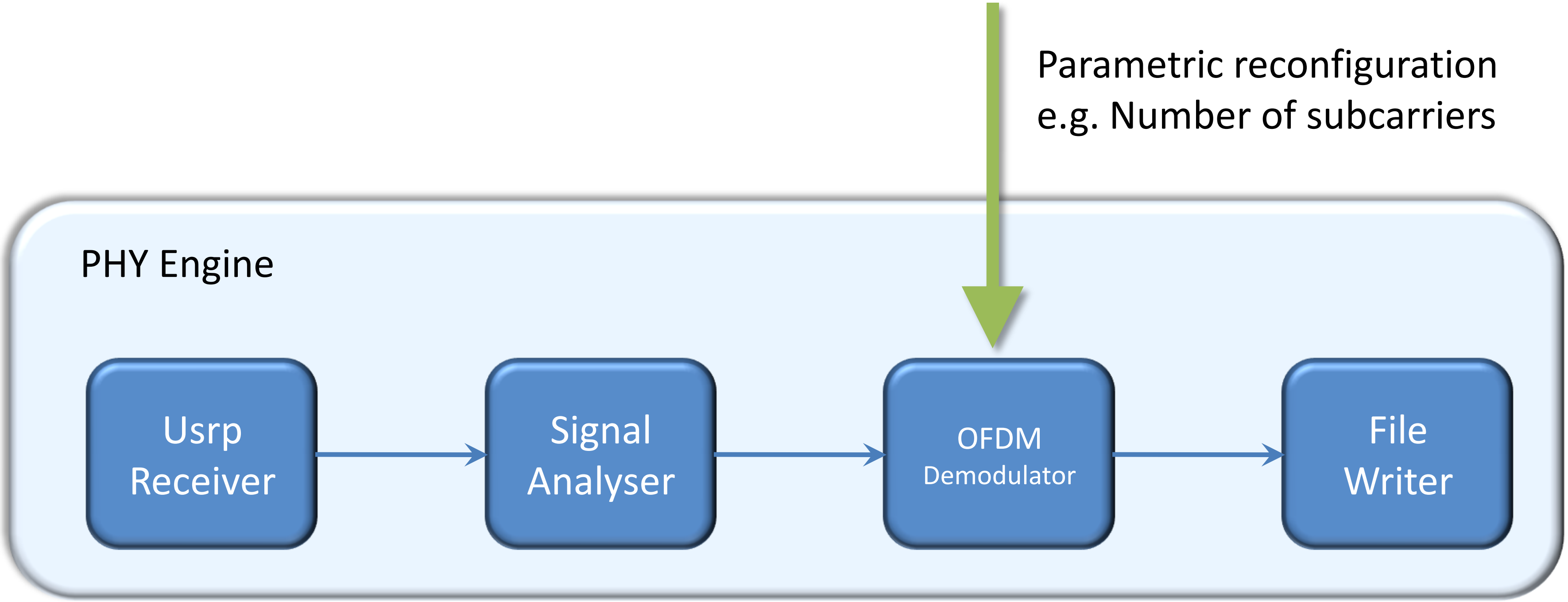
OFDM Demodulator

File Writer

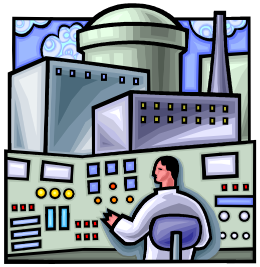
Controllers

- Parameters

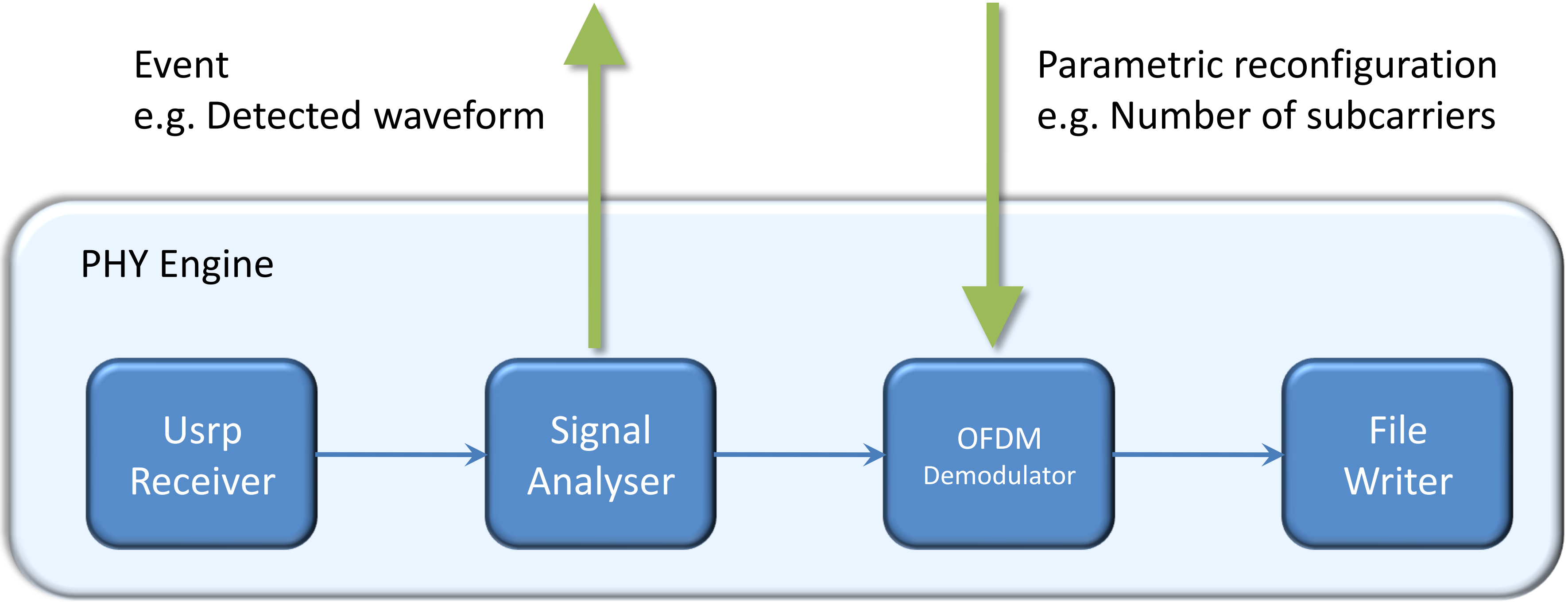




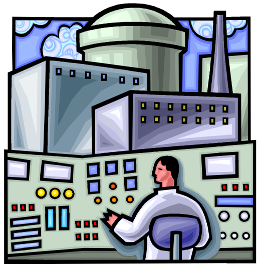
Controllers

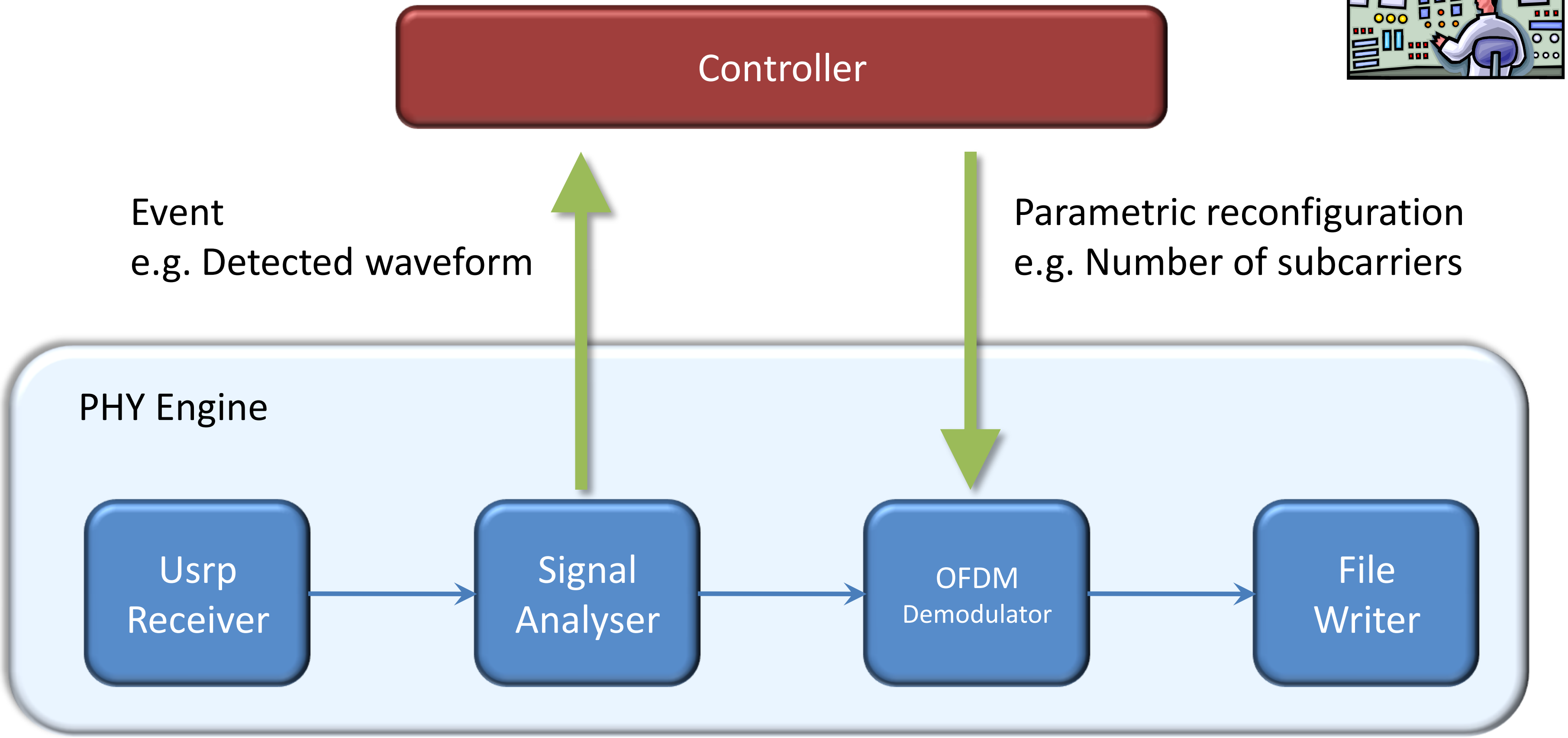


- Events



Controllers





Controller

Event
e.g. Detected waveform

Parametric reconfiguration
e.g. Number of subcarriers

PHY Engine

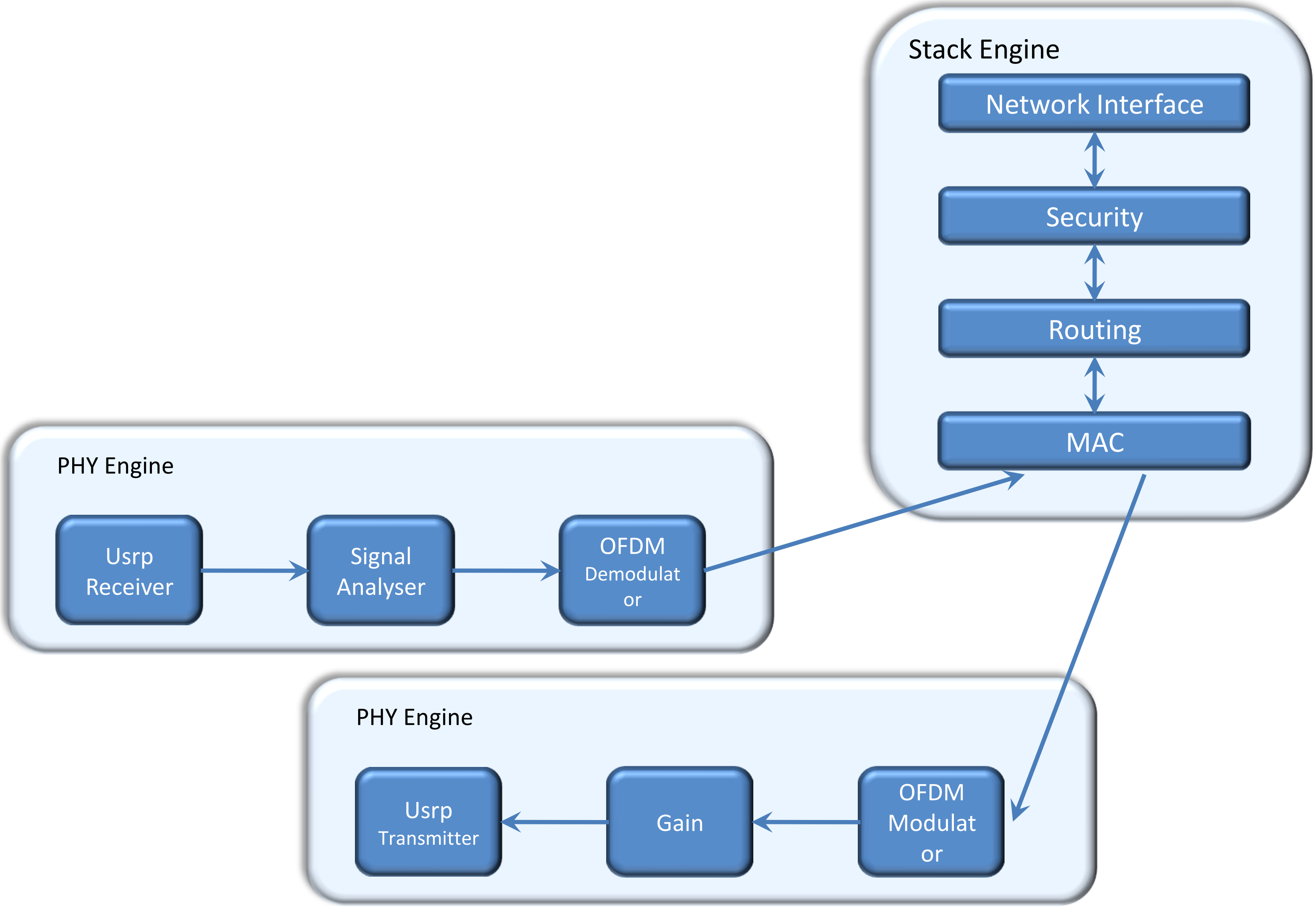
Usrp Receiver

Signal Analyser

OFDM Demodulator

File Writer

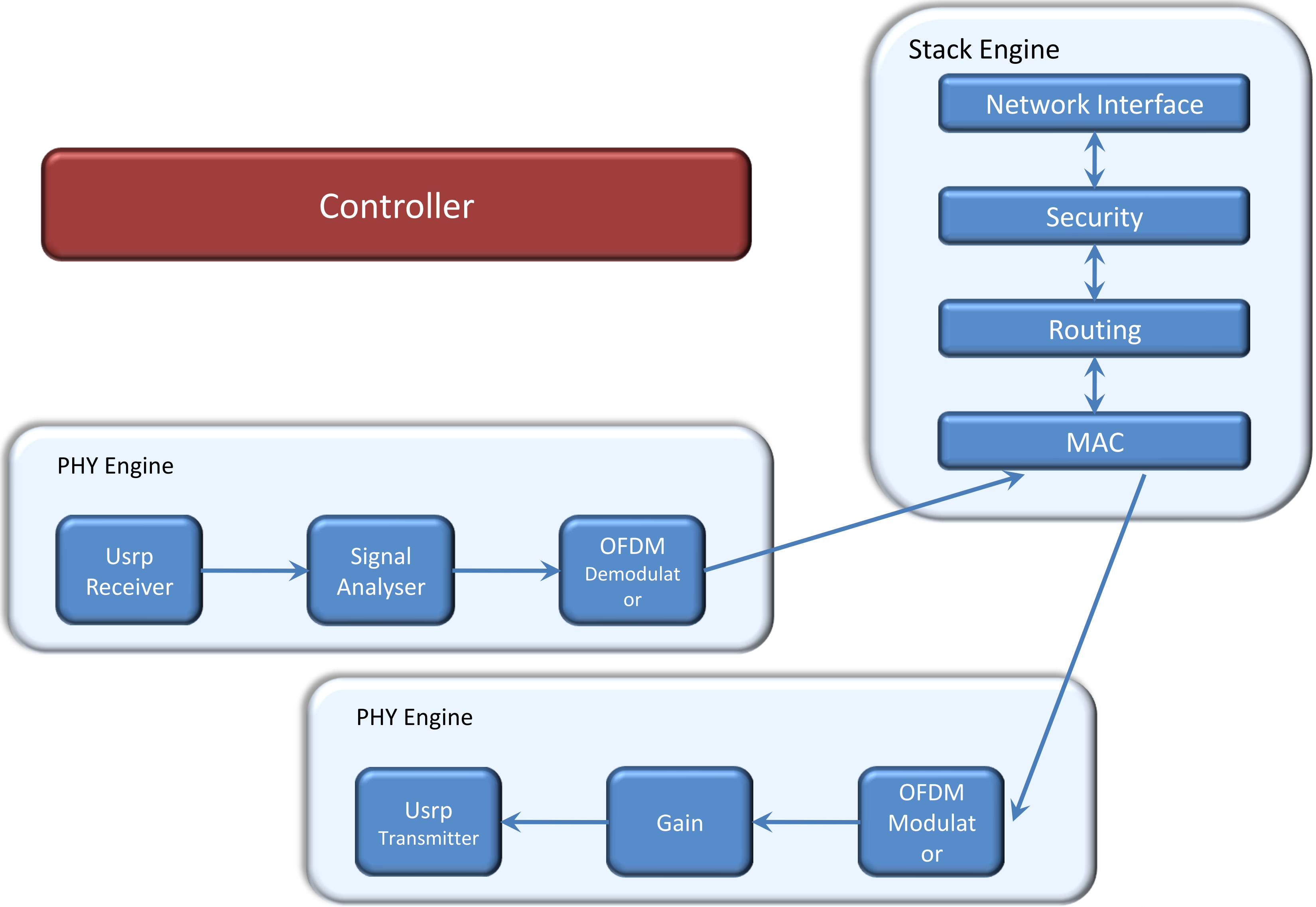
Controllers



Software Radio Systems

CTVR the telecommunications research centre

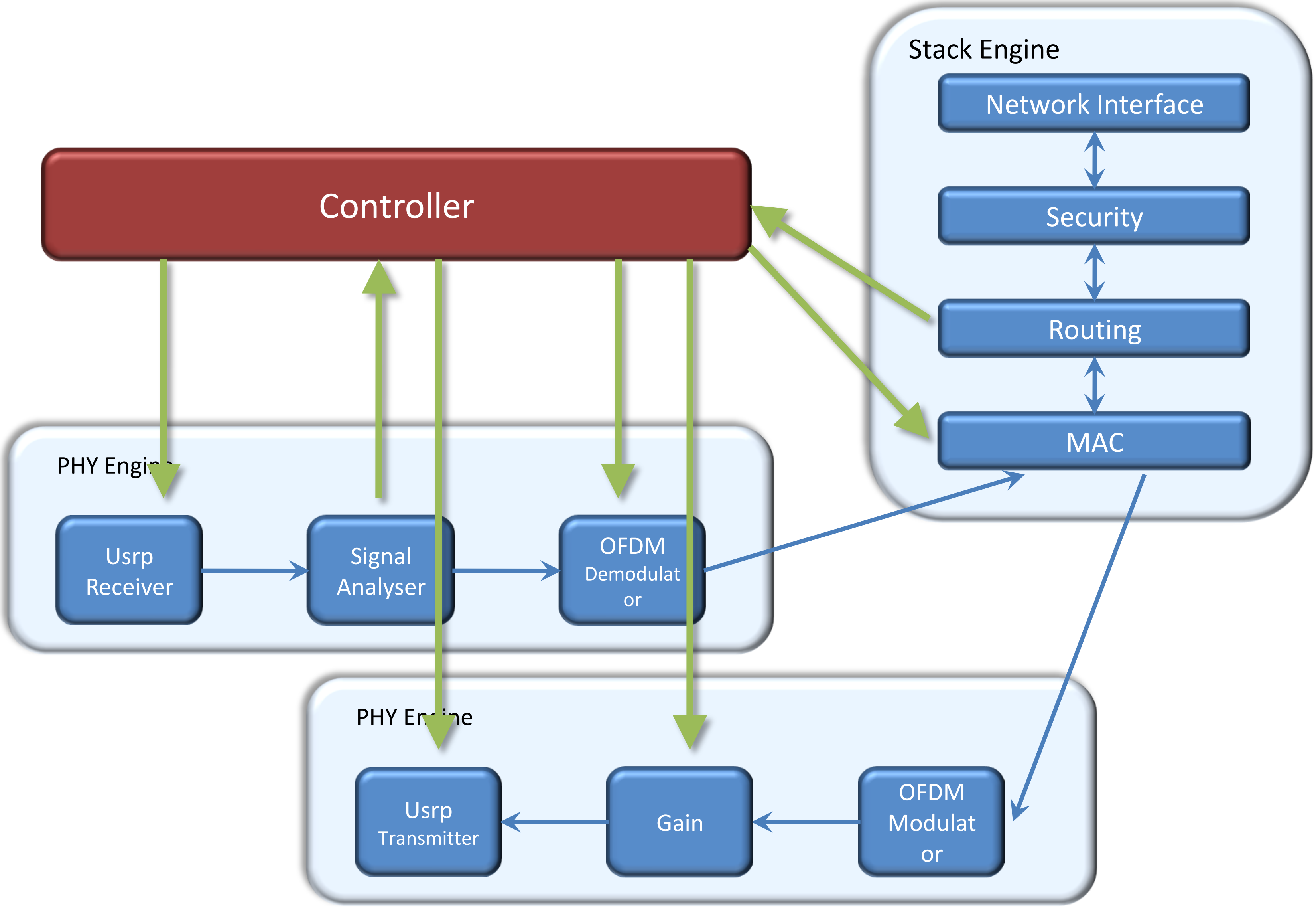
Controllers



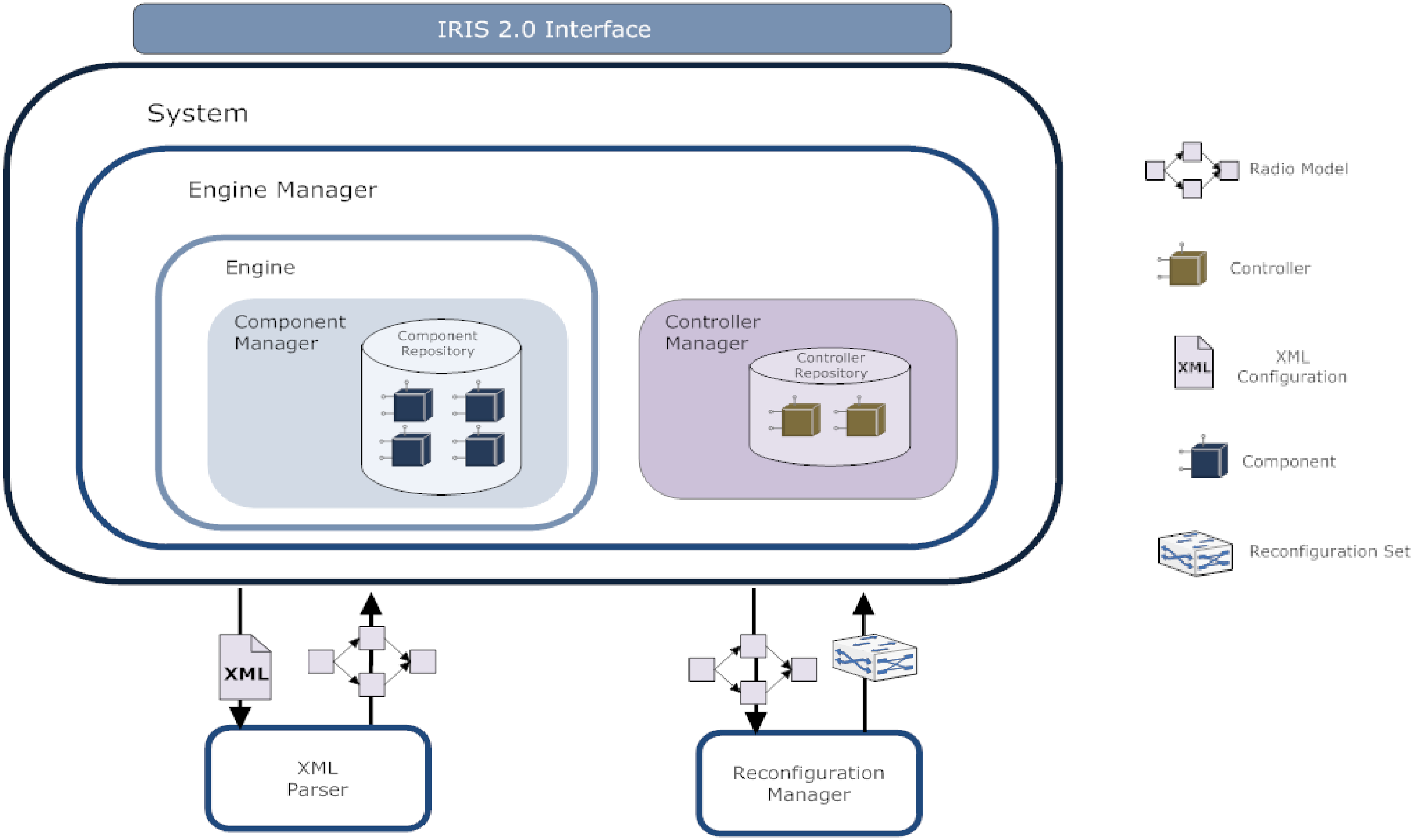
Software Radio Systems

CTVR the telecommunications research centre

Controllers



Iris Architecture - Core



SRS Software Radio Systems

CTVR the telecommunications research centre

Why use Iris?

Why use Iris?

Quick Learning Curve

Why use Iris?

Open Source

Quick Learning Curve

Why use Iris?

Open Source

Quick Learning Curve

Easy to Contribute

Why use Iris?

Open Source

Quick Learning Curve

Easy to Contribute

Small Project

Why use Iris?

Quickly Implement Complex Systems

Open Source

Quick Learning Curve

Easy to Contribute

Small Project

Getting Started

- Code: https://github.com/softwareradiosystems
- Redmine: http://www.softwareradiosystems.com/redmine/projects/iris
- Mailing Lists: http://www.softwareradiosystems.com/mailman/listinfo
- Blog: http://irissoftwareradio.wordpress.com/

Try it out

https://github.com/softwareradiosystems

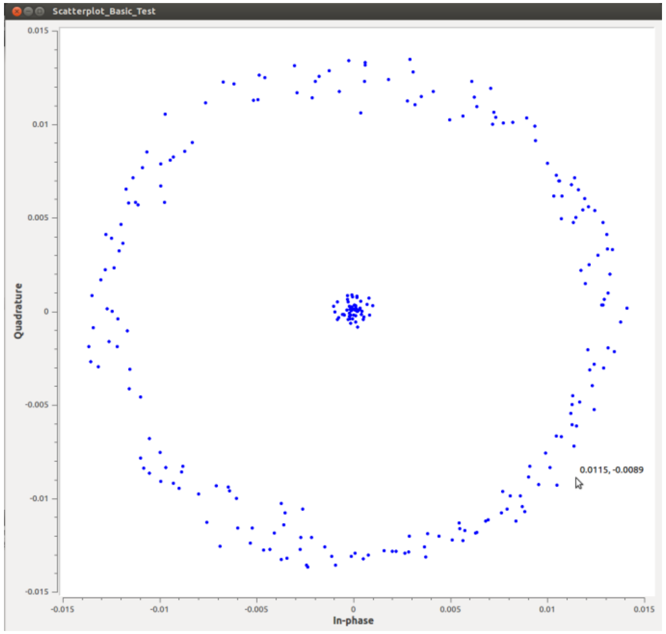
Thank you

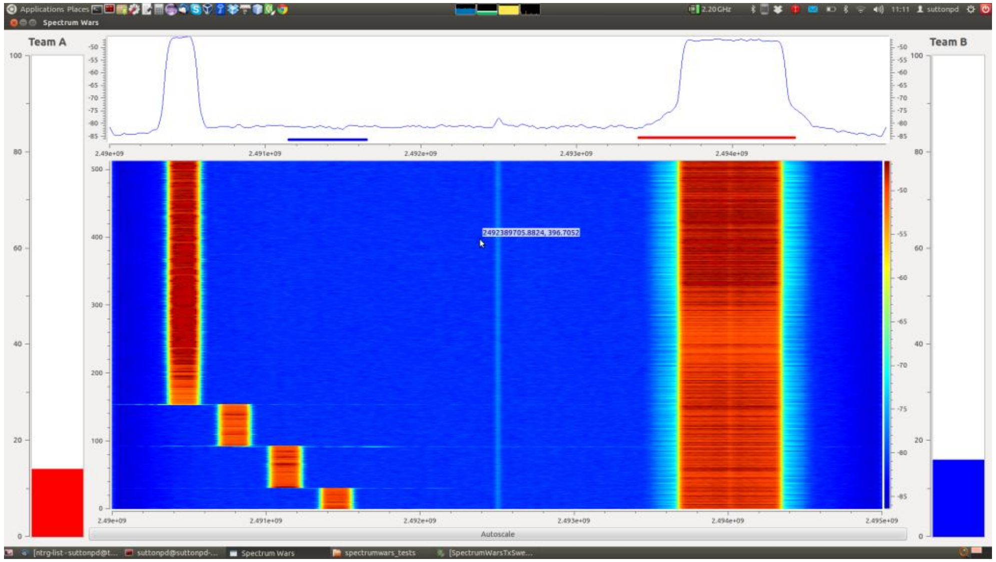
suttonpd@tcd.ie

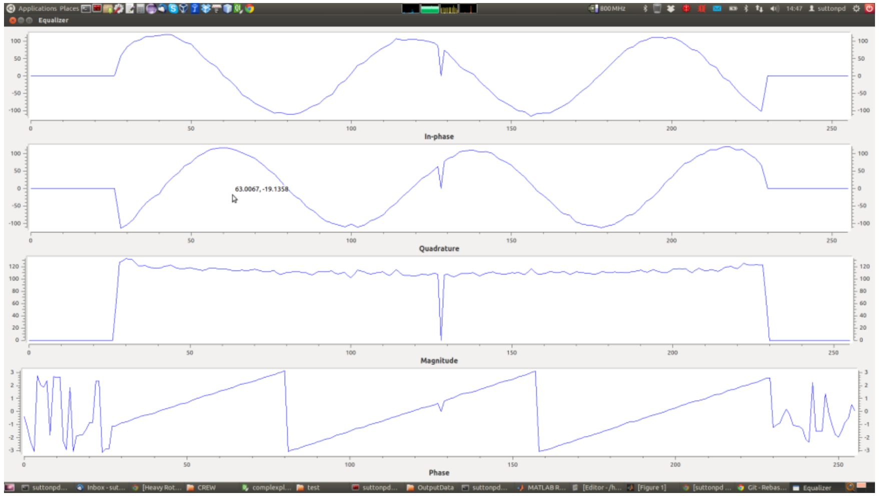
paul@softwareradiosystems.com

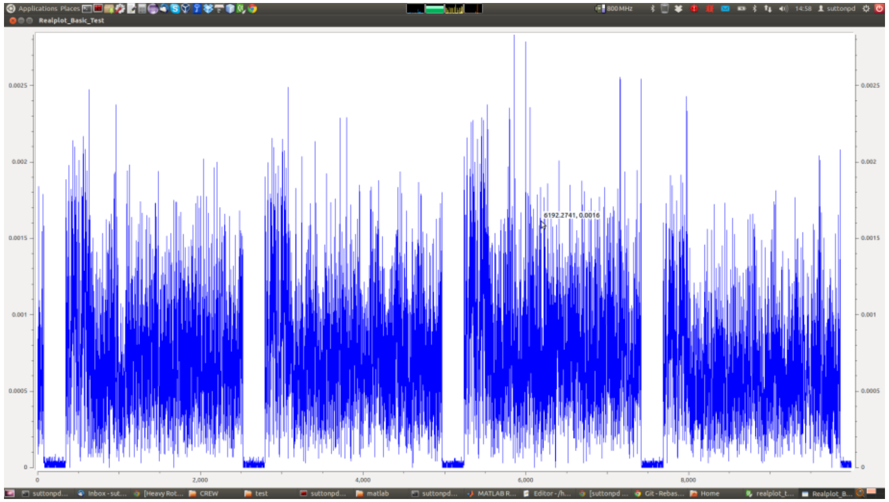
Additional Material

Release 1.1.0







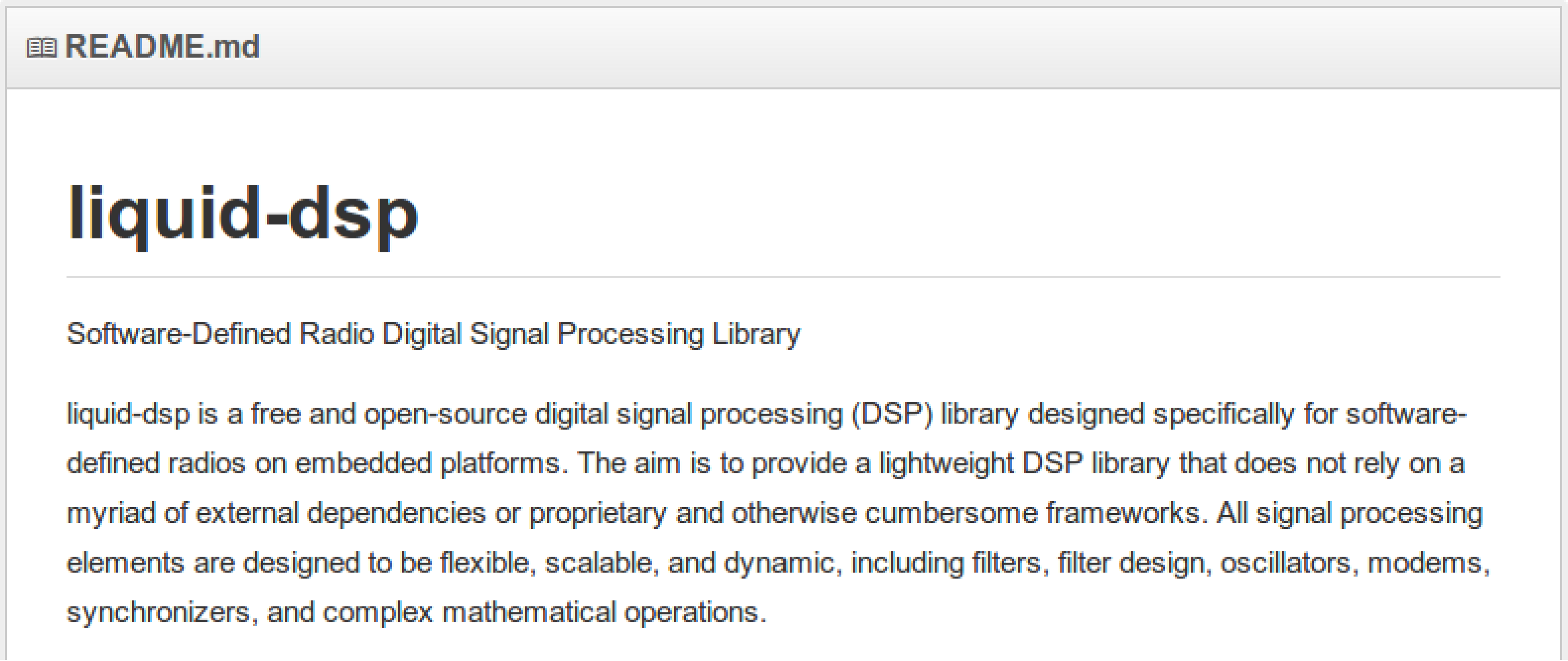


Software Radio Systems

CTVR the telecommunications research centre

Release 1.1.0

- Liquid-DSP Components



README.md

liquid-dsp

Software-Defined Radio Digital Signal Processing Library

liquid-dsp is a free and open-source digital signal processing (DSP) library designed specifically for software-defined radios on embedded platforms. The aim is to provide a lightweight DSP library that does not rely on a myriad of external dependencies or proprietary and otherwise cumbersome frameworks. All signal processing elements are designed to be flexible, scalable, and dynamic, including filters, filter design, oscillators, modems, synchronizers, and complex mathematical operations.