Enhanced and Embedded FPGA GNU Radio Flow

Presented by Ryan Marlow

Agenda

- Introduction of GReasy
- GReasy developments discussed in this paper
- Progress since then

The Team

Advisor: Dr. Peter Athanas

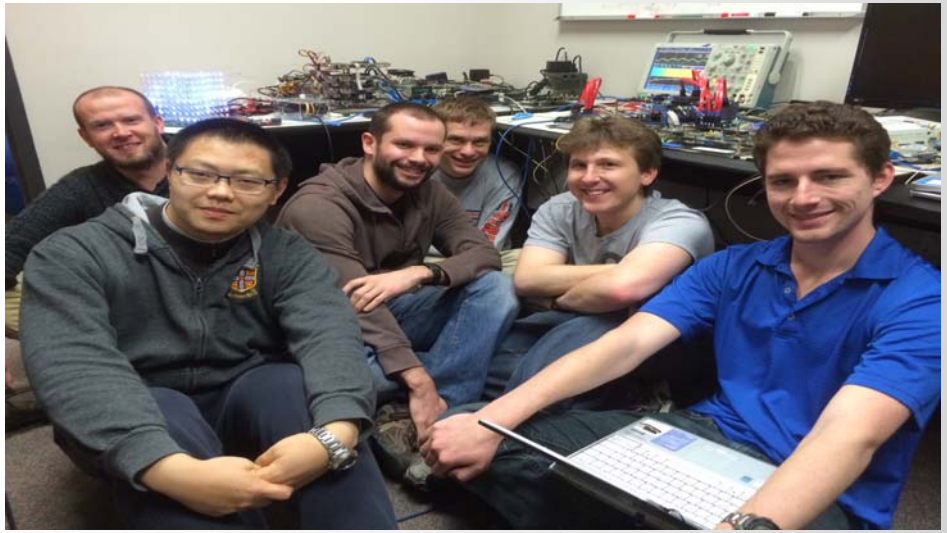
Post-Doc: Krzysztof Kepa

PhD students:

- Andrew Love

Master's Students:

- Ryan Marlow
- Chris Dobson
- Kurt Rooks
- Kevin Lee
- Shenghou Ma



Motivation

- GNU Radio is limited by GPP.
  - Only a certain class of radio designs can be prototyped
- We want to expand GNU Radio processing capability by adding additional hardware in the loop.
  - Expand the design space with FPGA
- Preserve Instant Gratification!

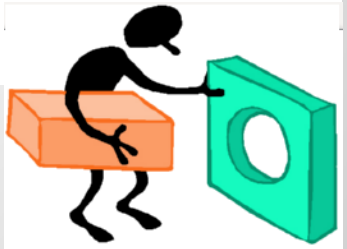
FPGA Accelerated GNU Radio?

GNU Radio
THE FREE & OPEN SOFTWARE RADIO ECOSYSTEM

- Instant Gratification
- Intuitive to Radio Designers
- Allows Design-Space Exploration w/ Real Data

XILINX®

- Long Compile Times
- Comm People Hate HDLs
- Lengthy Simulations with Limited Data

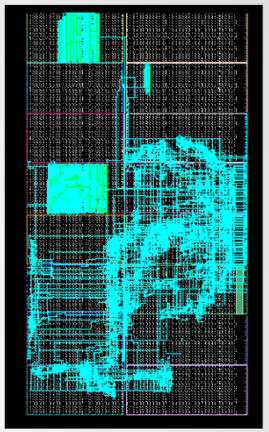


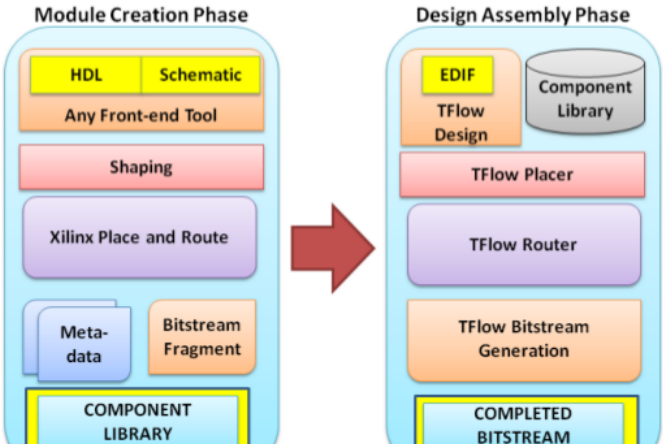
General Overview of GReasy

- Making Radios the (GR)Easy way
- Pre-Compiled FPGA Block Library (tFlow)
- Drag and Drop Hardware Blocks in GRC
- Alternative Bitstream Assembly (tFlow)
- Speed, Agility, Complexity => Productivity (GReasy)

What is TFlow?

- Rapid Bitstream Generation Tool
- Splits FPGA into two regions:
  - Static Design and Sandbox Design
- Works by splitting bitgen into two phases
  1. module creation
  2. design assembly

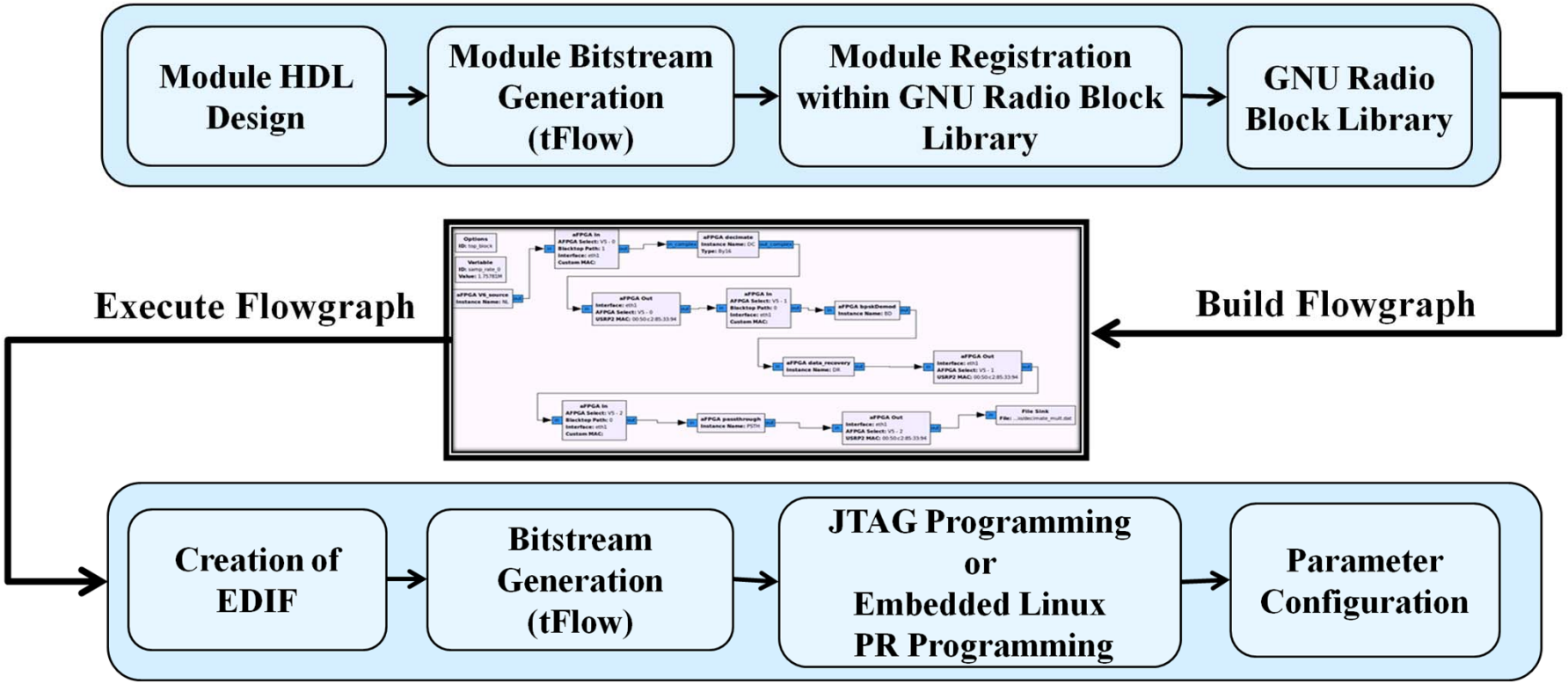




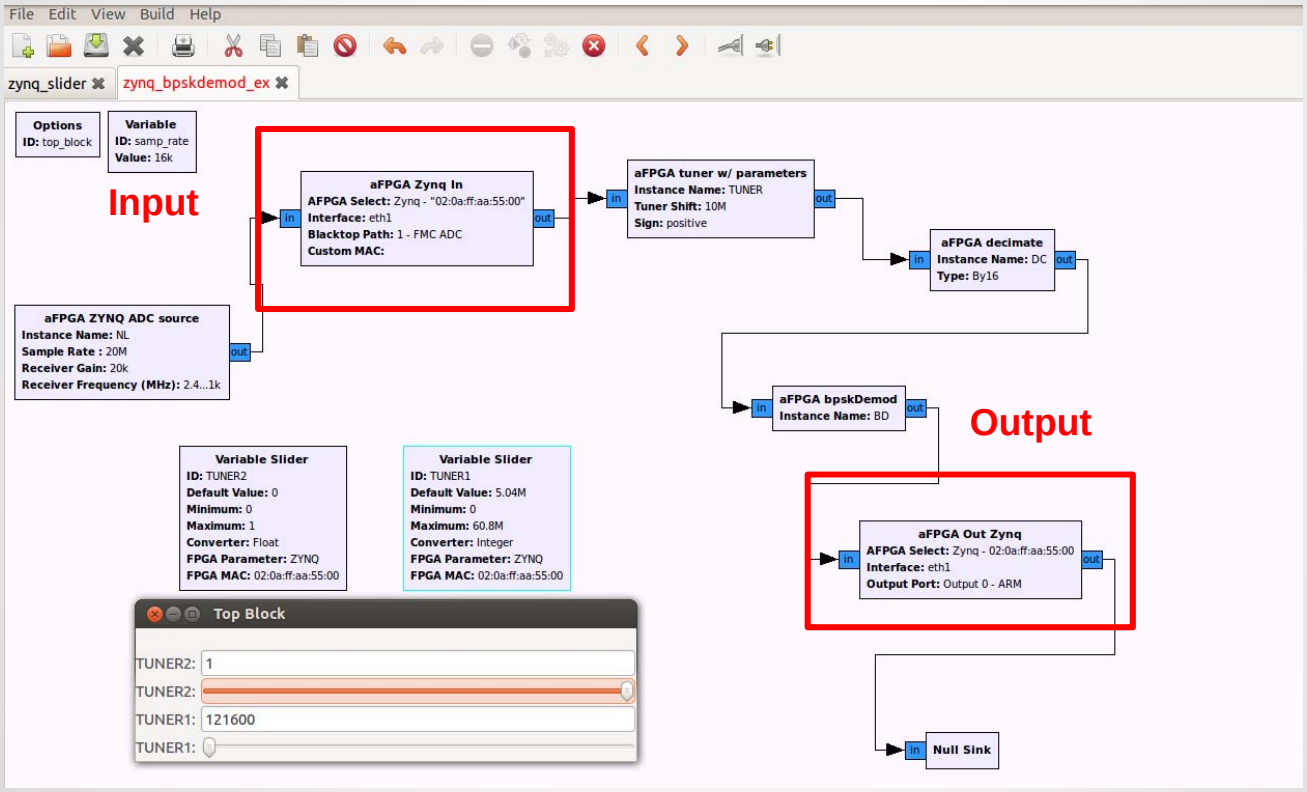
Sandbox Model

- Designs built inside the TFlow Sandbox
- Modules connect to each other and sandbox ports:
  - Input / Output
  - Parameterization
  - Multiple Clocks
  - Reset
- Automation in generation

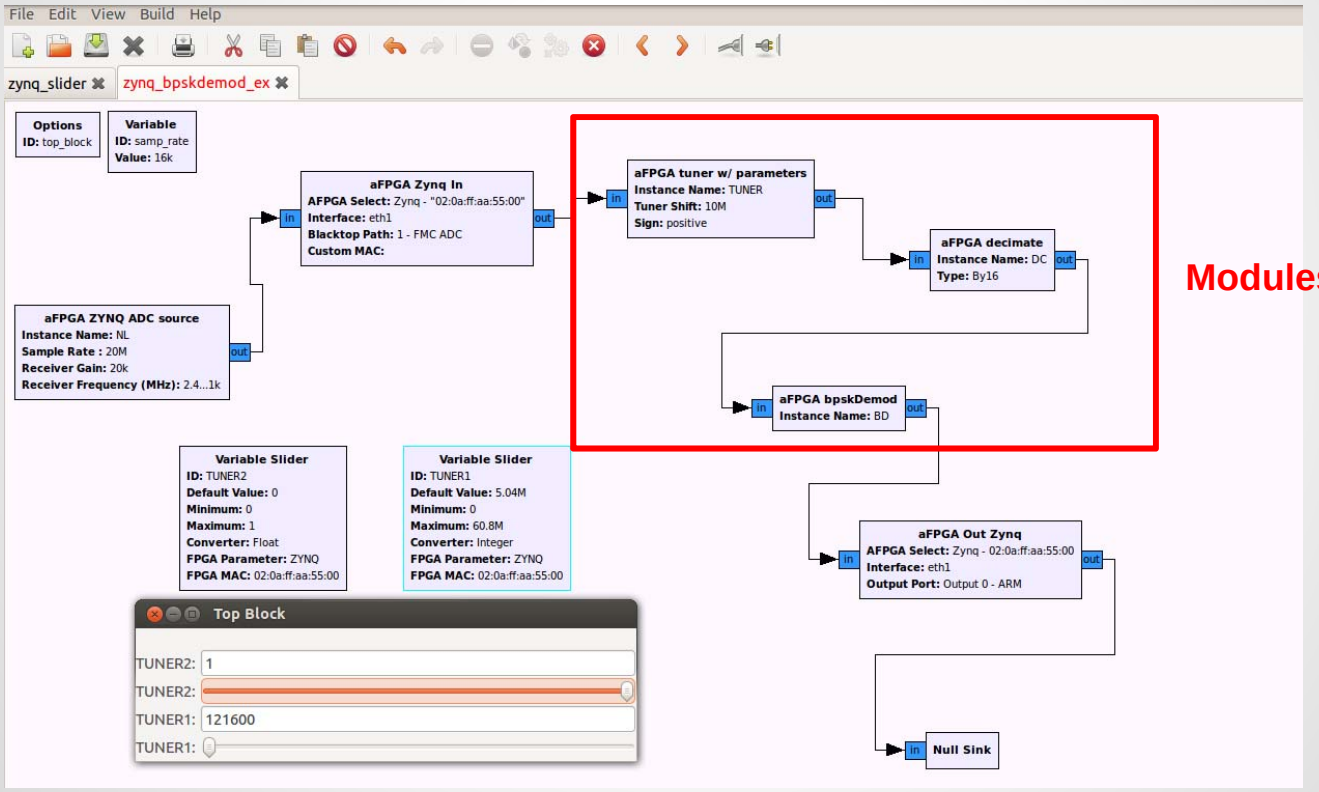
Design Flow with GReasy



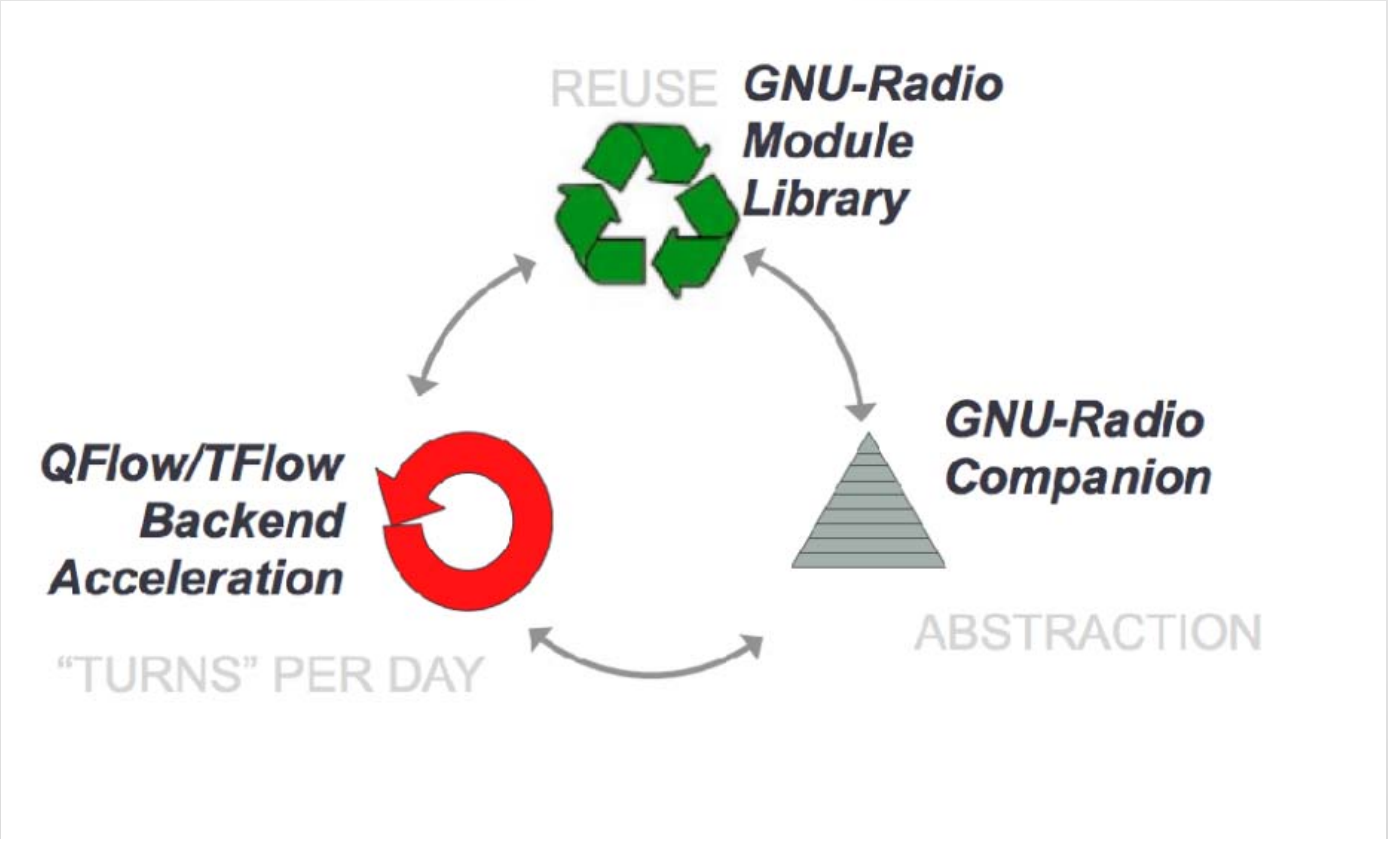
Important GNU Radio Components



Important GNU Radio Components



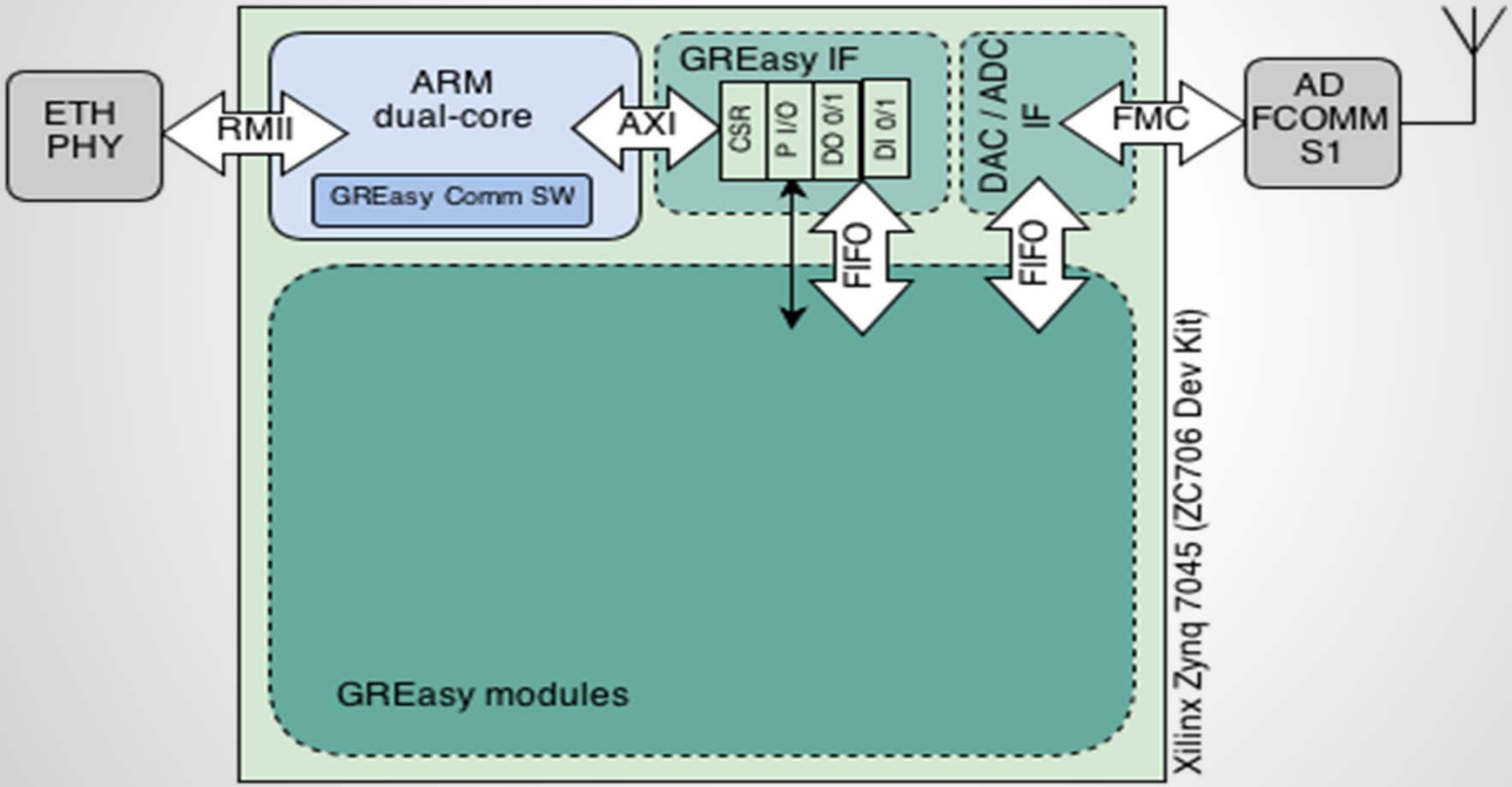
GReasy Productivity Solution



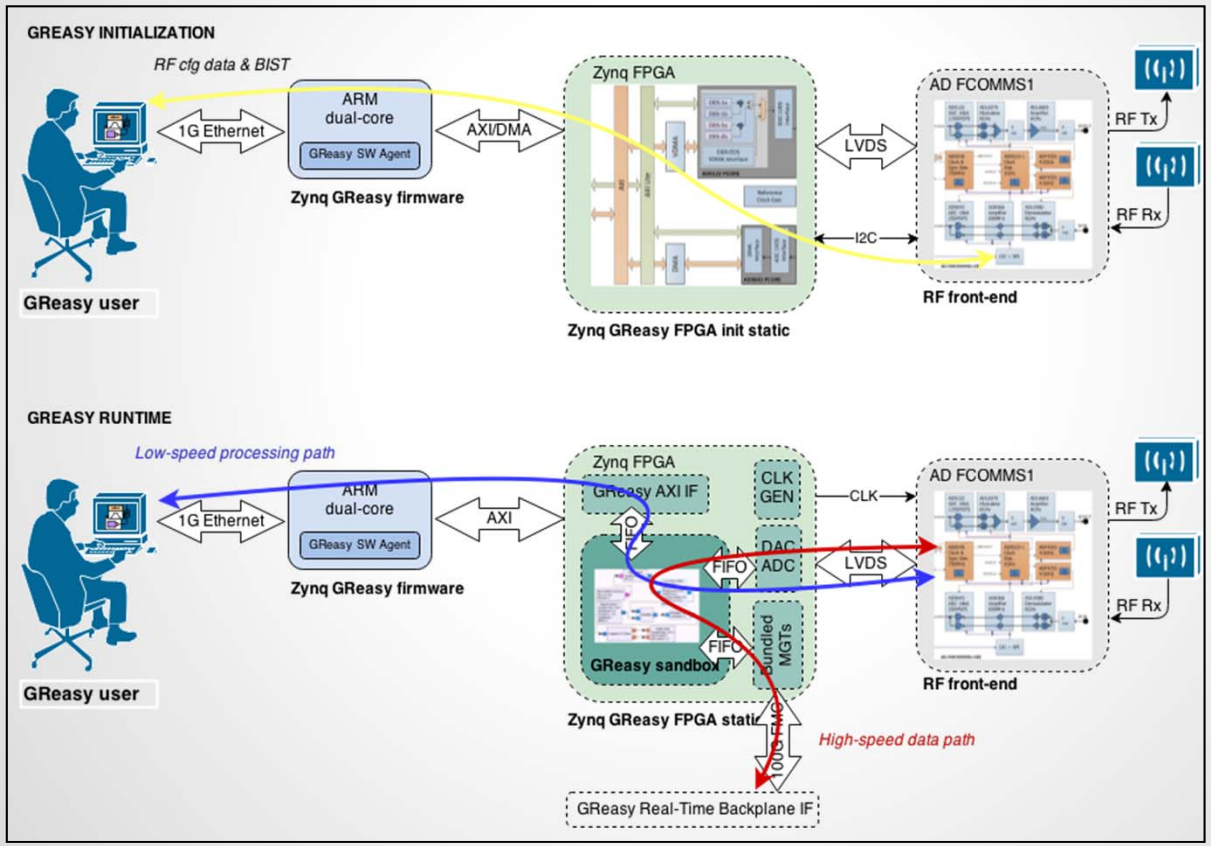
Significant Updates in this Paper

- Upgrade to Zynq Architecture
  - ARM and FPGA on same fabric
  - New ADC/DAC Frontend
- Multi-family/FPGA, heterogenous designs
  - GNU Radio FPGA module library is more generic and easily expandable
- Modules are parameter configurable
  - expanded communication scheme

Zynq Signal Flow



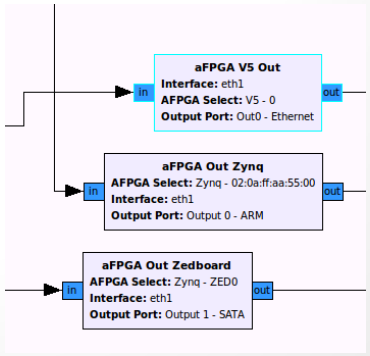
Radio Frontend User Interfacing

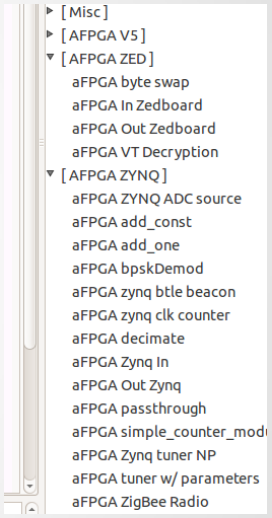


Heterogeneous Library

Multiple FPGA Families/Devices

- Virtex 5 - V5LX110T
- Zynq - ZC7045
- Zynq - ZC7020 / ZED
- Easily Expandable



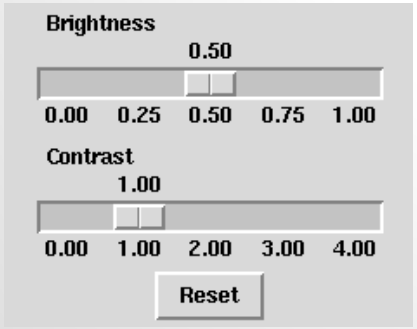


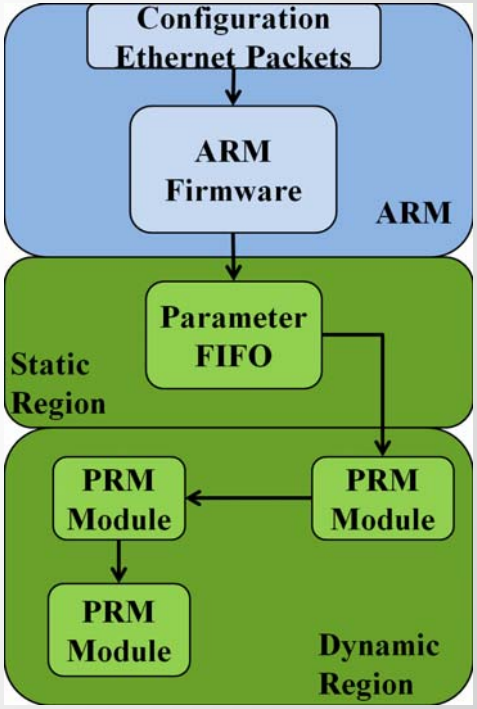
Parameterization

Alter FPGA design after programming!

Host sends parameter values at start up and concurrent to data streaming

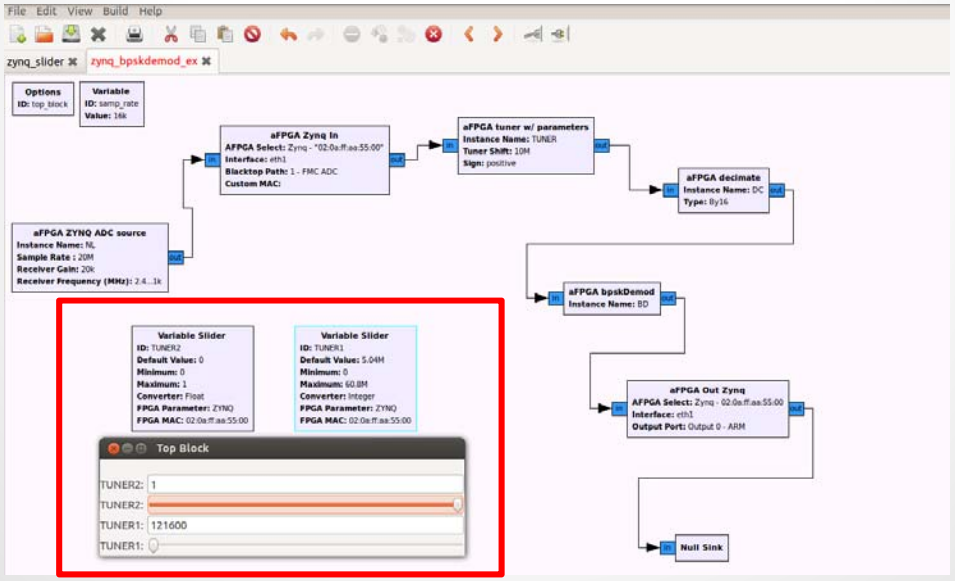
- updating parameters is instantaneous





Parameter Slider

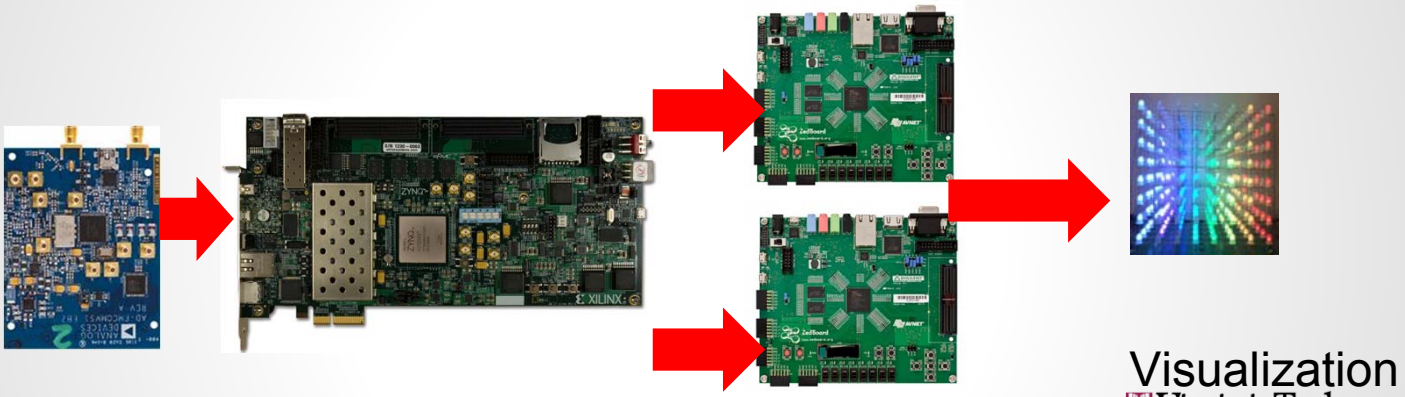
- adjust sliders while hardware is running
- see changes in real time



Newest Additions

- Constructed a New Demo Platform
- New flexibilities to make GReasy easier
  - multiple clock domains
  - multiple “shapes” for tighter placement of modules
- tFlow based Partial Reconfiguration

Demo Component Flow



RADIO
CARD RX/TX
ANALOG DEVICES

Zynq ZC706
FRONT END /
PROCESSING
XILINX

Zynq ZED
PROCESSING
XILINX

Visualization
VirginiaTech
Invent the Future

Multiple Clocks

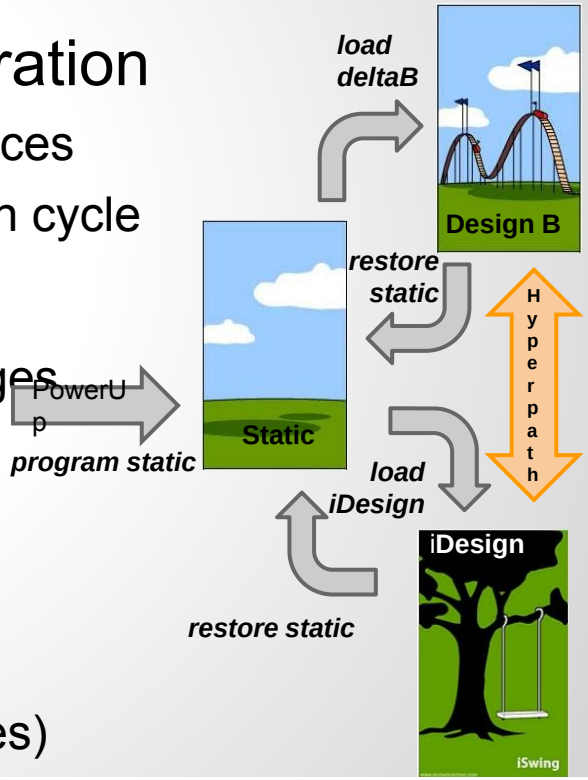
Support for multiple clocks driving modules.

Example:

- DAC clock
- ADC clock
- ARM clock

Partial Bitstreams

- Until now TFlow used full reconfiguration
  - Prohibitive configuration time for large devices
  - device is out-of-service during configuration cycle
- TFlow support for PR
  - productivity boost due to small ***delta*** changes
    - reduced system transition time
  - increases embedded autonomy
    - self-adaptation
    - **no** vendor tools and constraints
  - eliminates system down-time (local changes)



Wrap Up

- Paper presents significant developments and capabilities to GReasy
- On-going project: still developing new capabilities
- Questions?