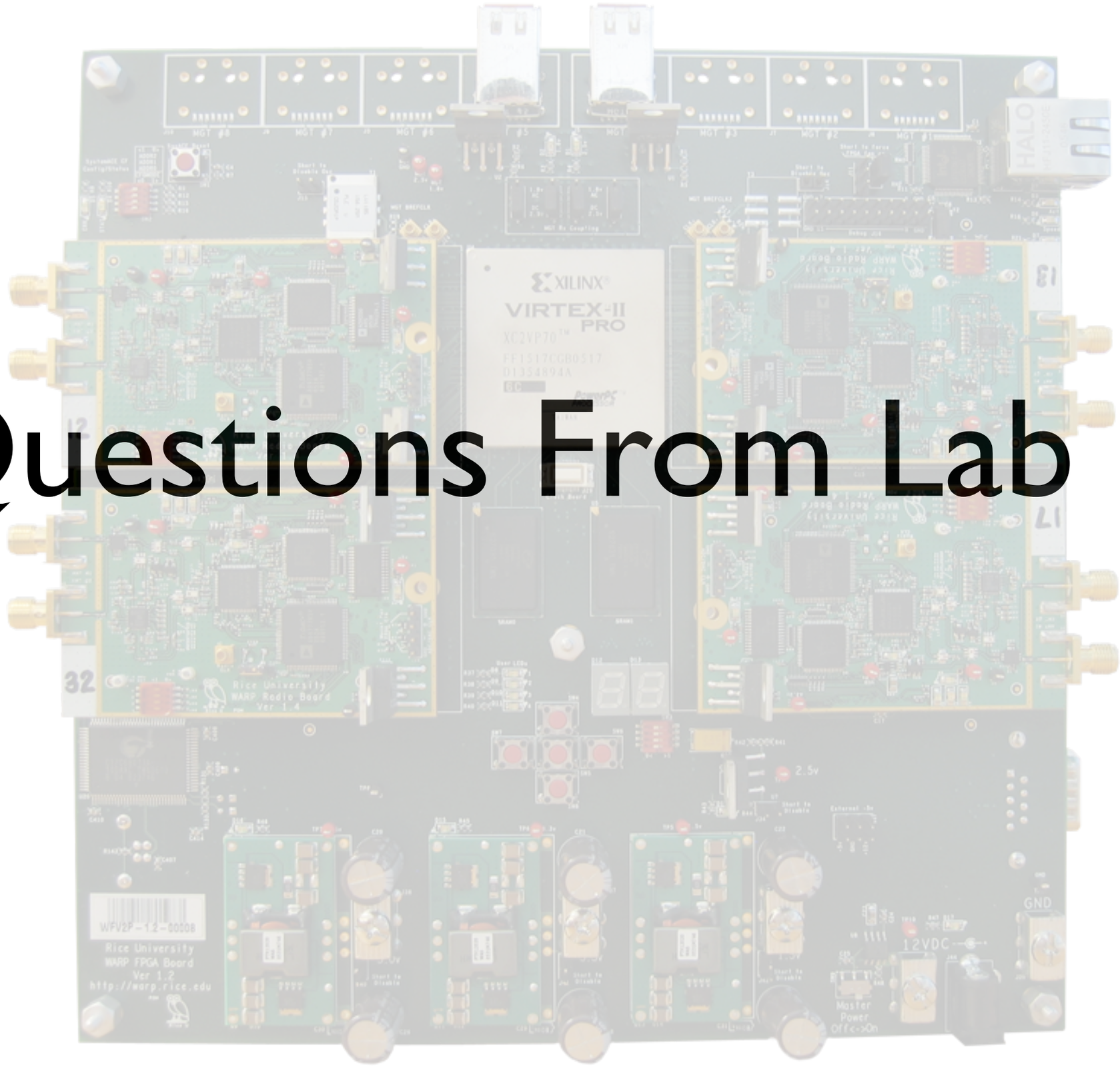
WARP: Physical Layer Design

Patrick Murphy
Rice University

WARP Workshop
November 1, 2007

warp.rice.edu

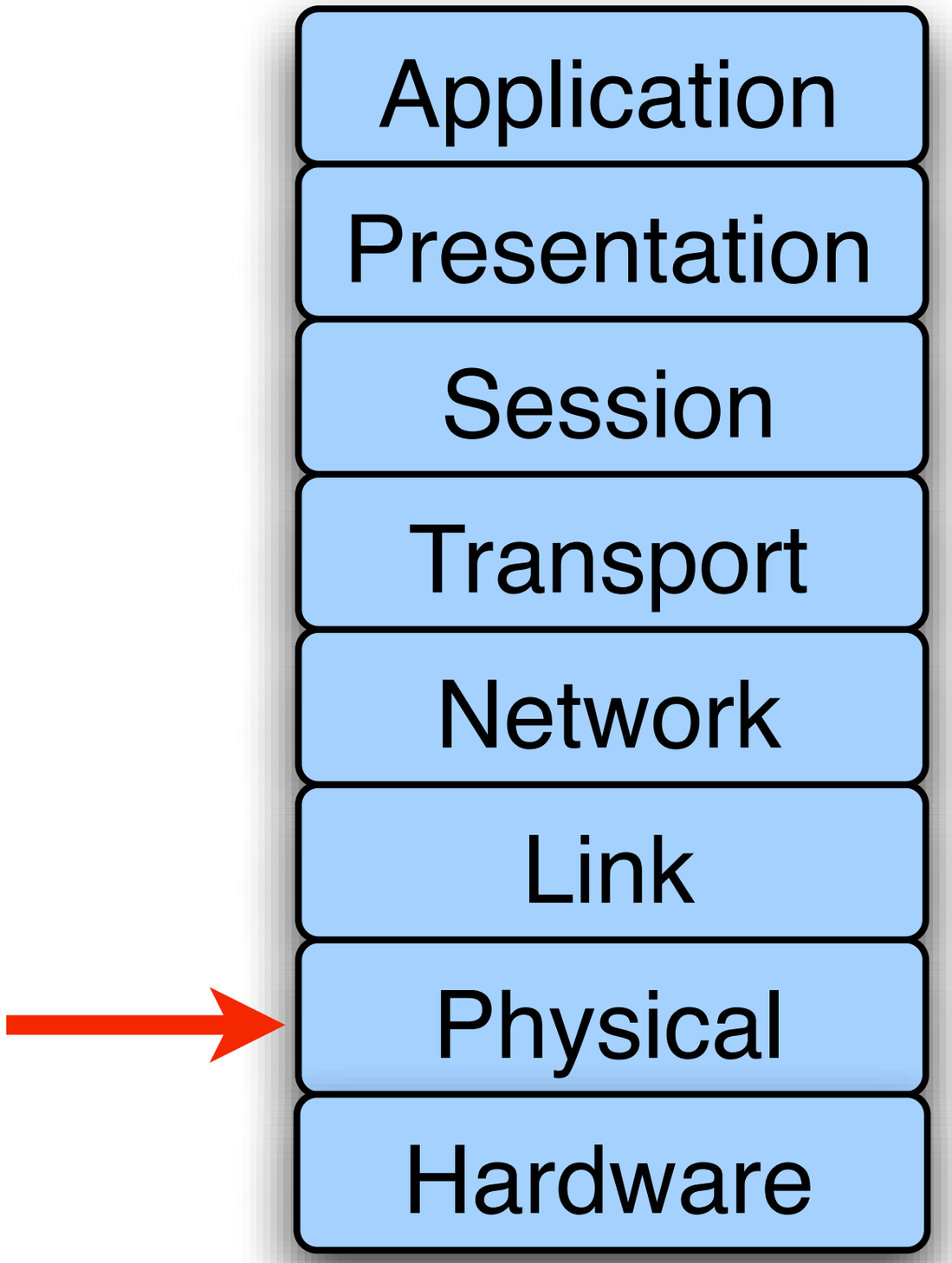
Questions From Lab 1?



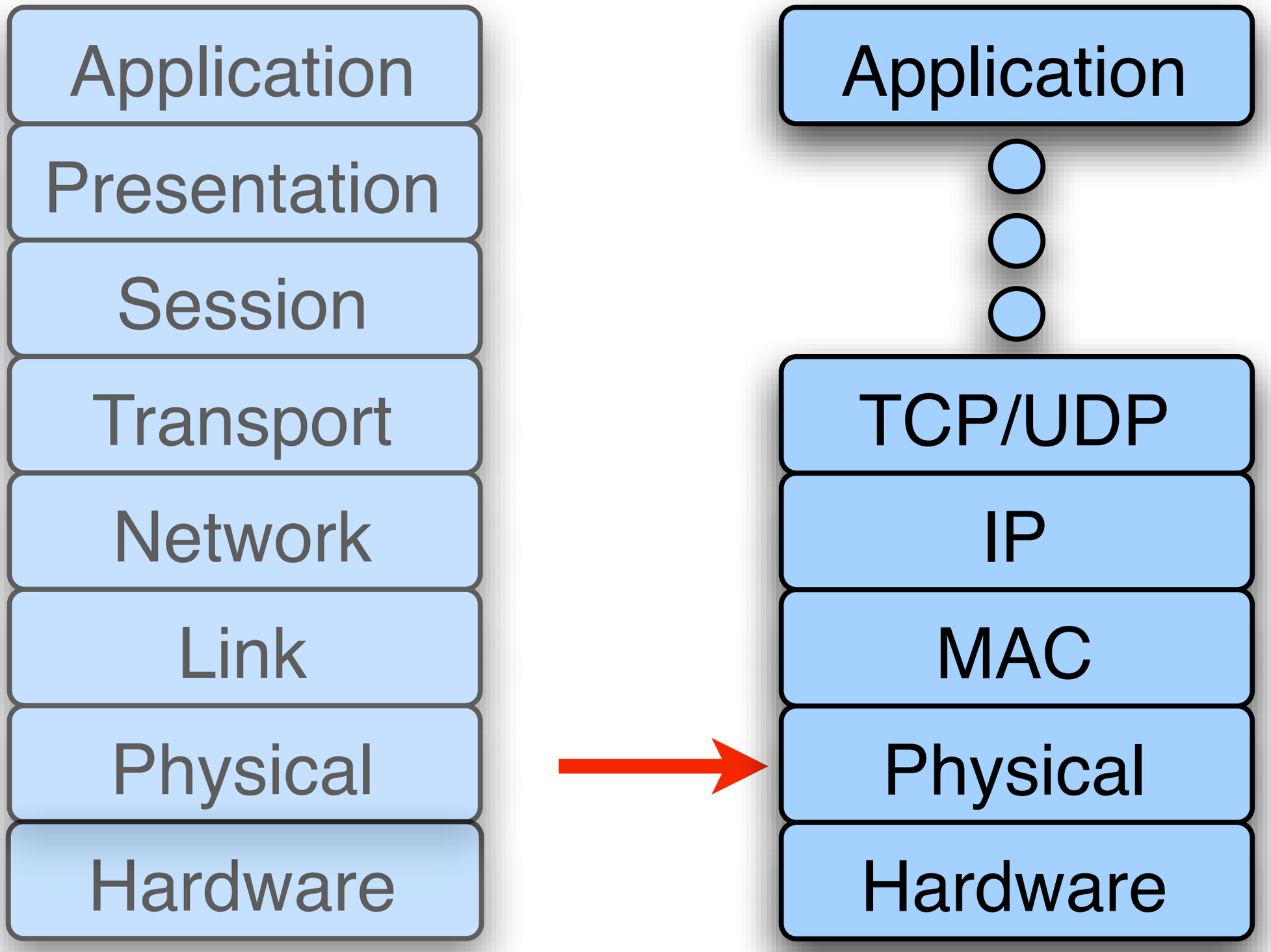
PHY Design - Outline

- Physical layer basics
- Intro to WARPLab PHY design flow
- Lab 2: Using WARPLab
- Real-time PHY design flow
- Lab 3: Building a simple transmitter

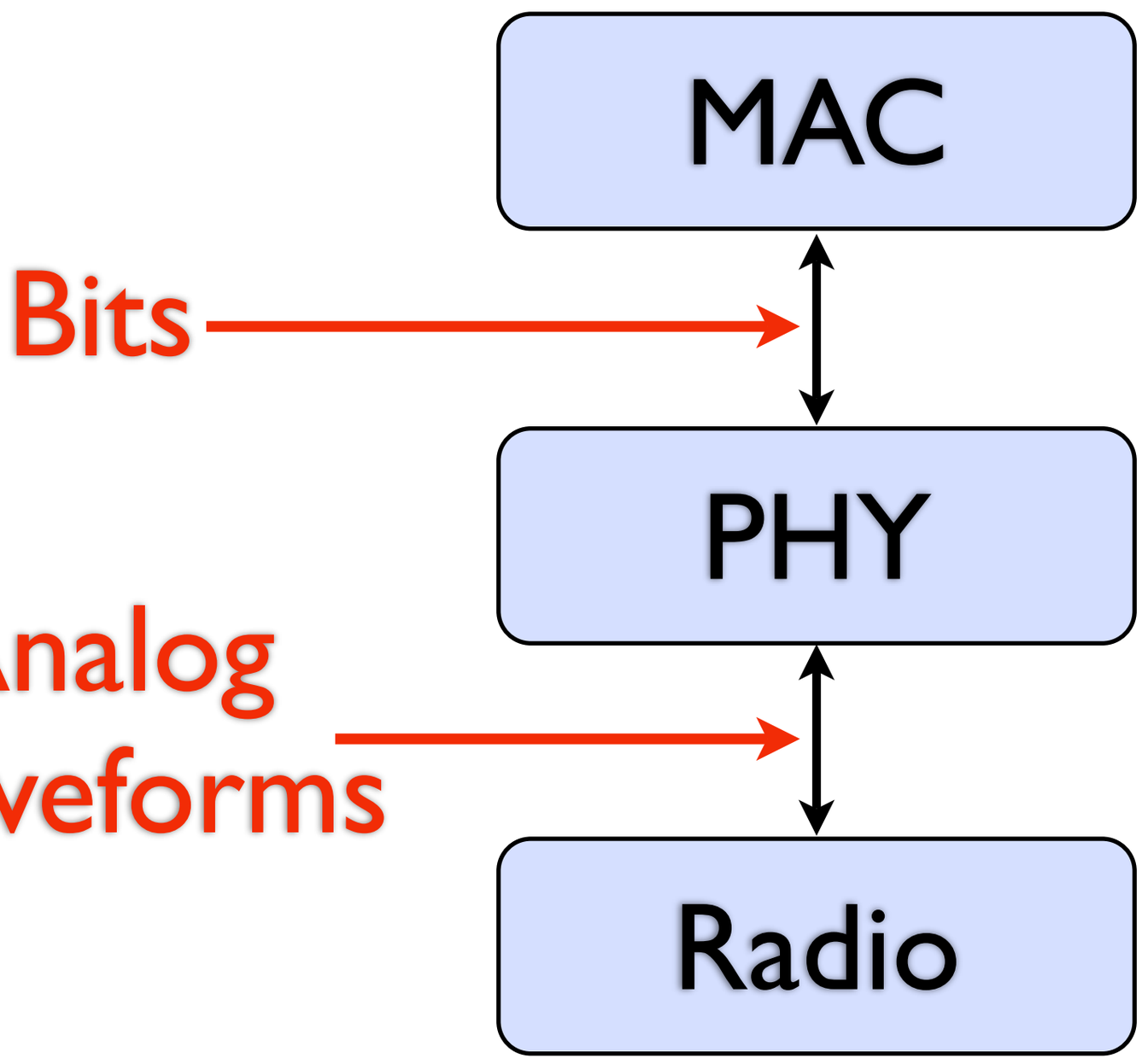
Physical Layer Basics



Physical Layer Basics

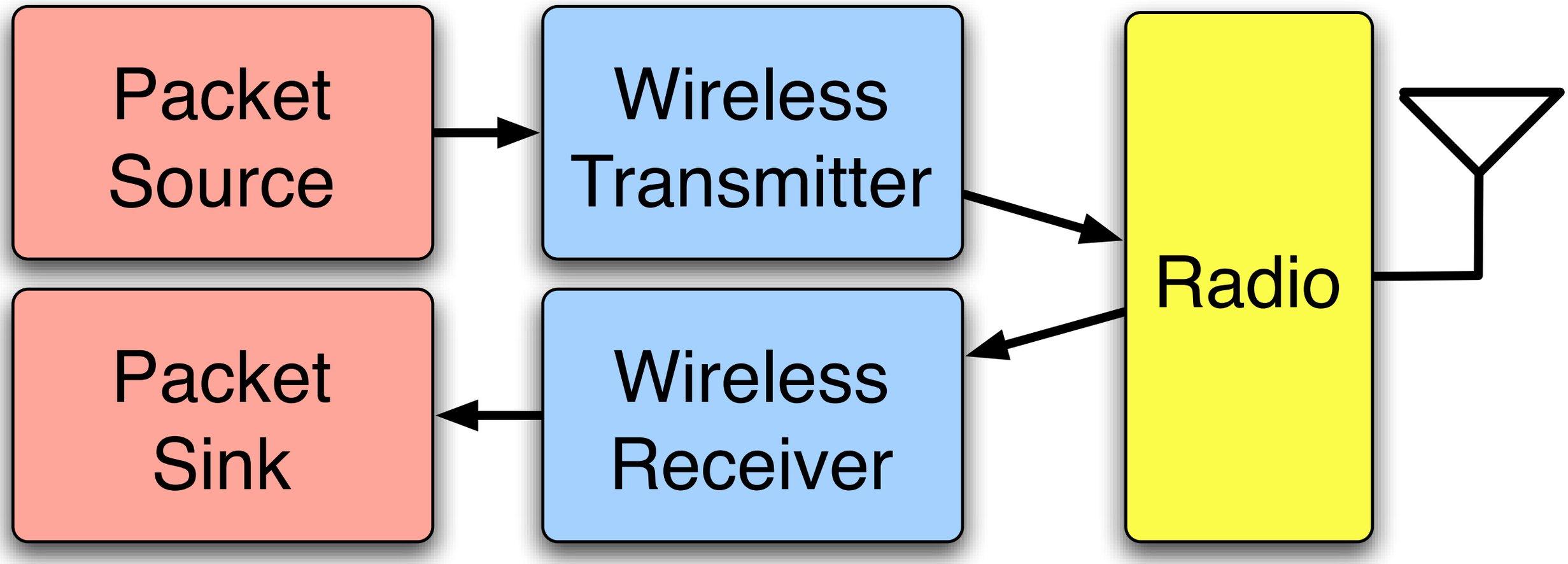


Physical Layer Basics



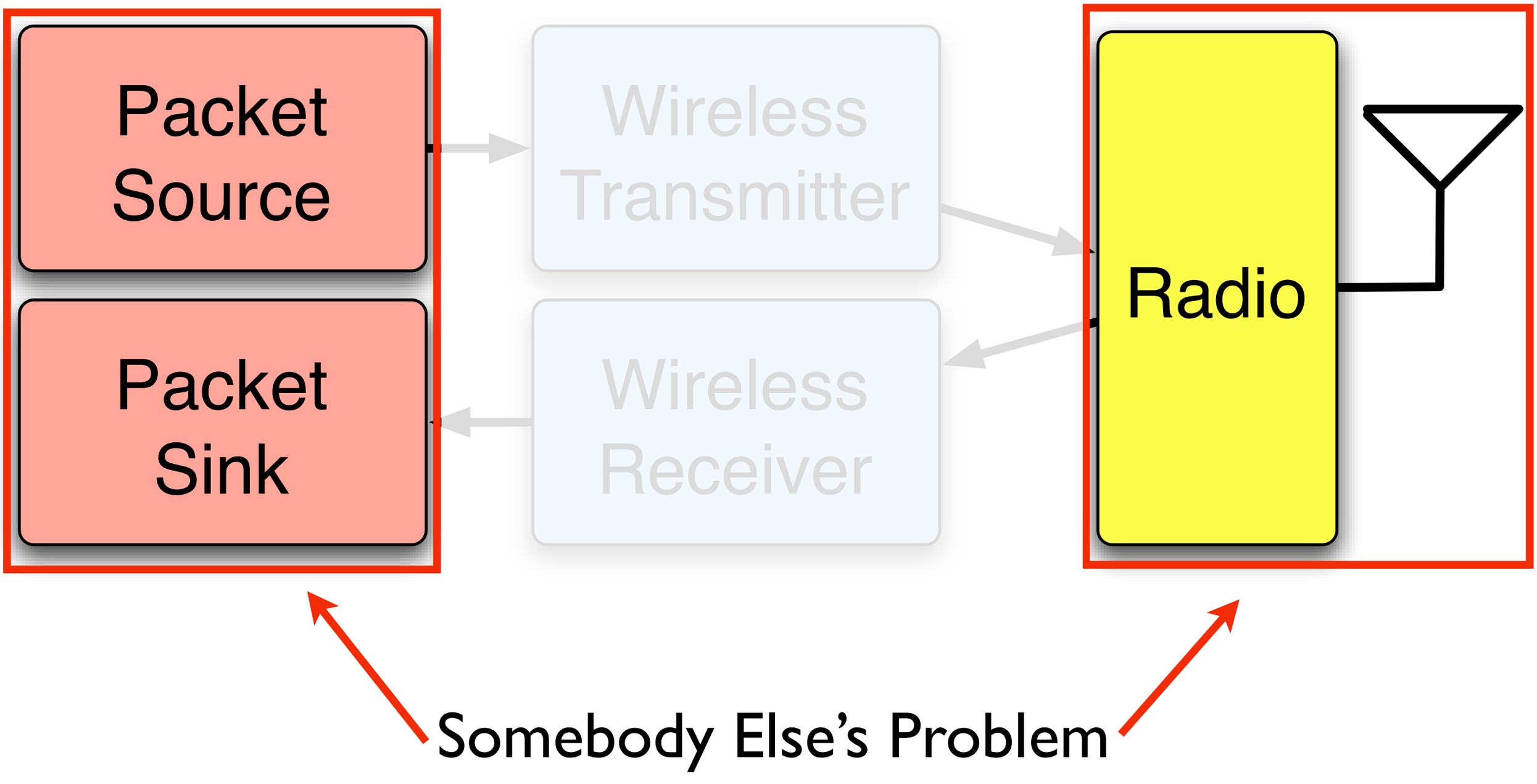
Physical Layer Basics

Simple Wireless Node



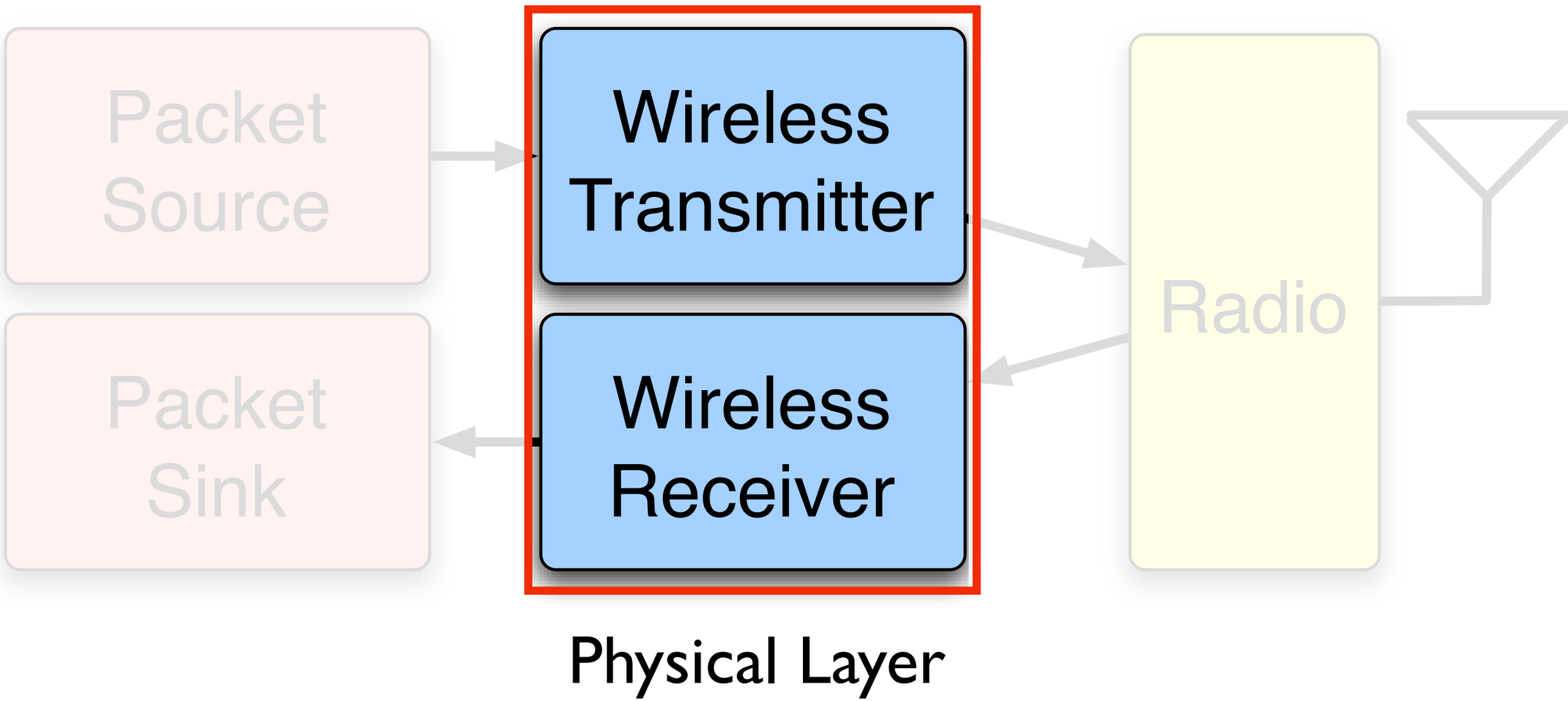
Physical Layer Basics

Simple Wireless Node



Physical Layer Basics

Simple Wireless Node



Physical Layer

PHY Design Flows

- WARPLab
  - MATLAB↔WARP Link
    - Very rapid prototyping of PHY algorithms
- Real-time PHY design
  - Low-level FPGA design
  - Putting it all together

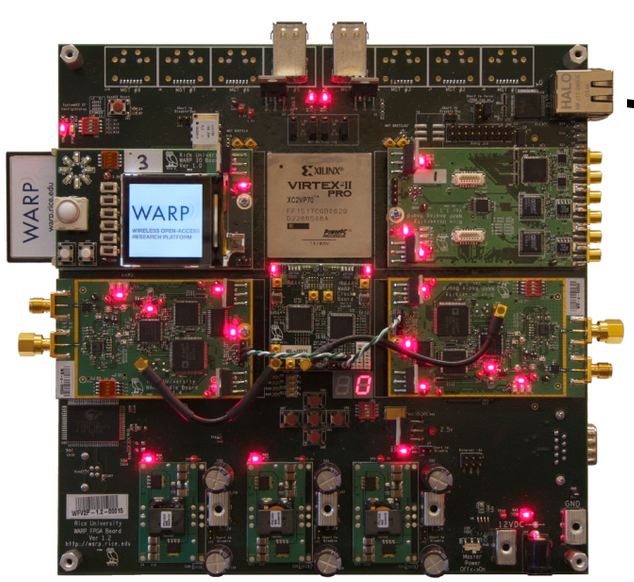
PHY Design Flows

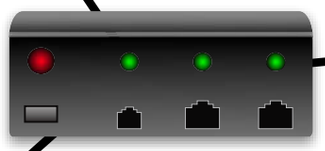
- WARPLab
  - MATLAB↔WARP Link
  - Very rapid prototyping of PHY algorithms
- Real-time PHY design
  - Low-level FPGA design
  - Putting it all together

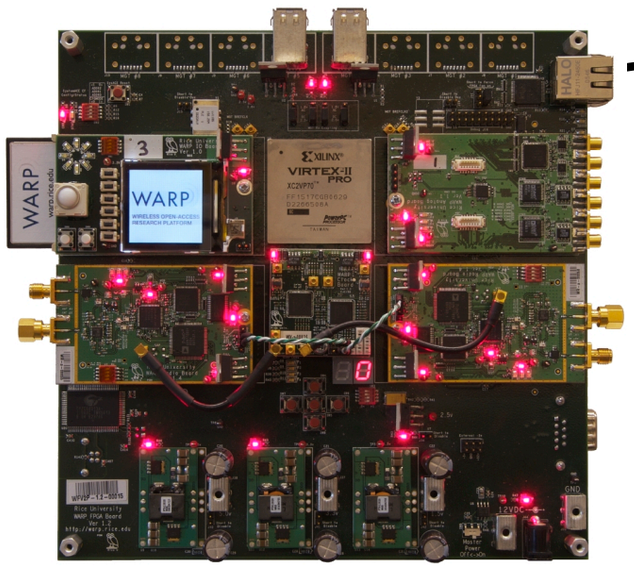
WARPLab Overview

- MATLAB for signal processing
- WARP for wireless interfaces
- Real-time channel use
- Non-real-time processing
- One PC controls many WARP nodes

WARPLab Overview

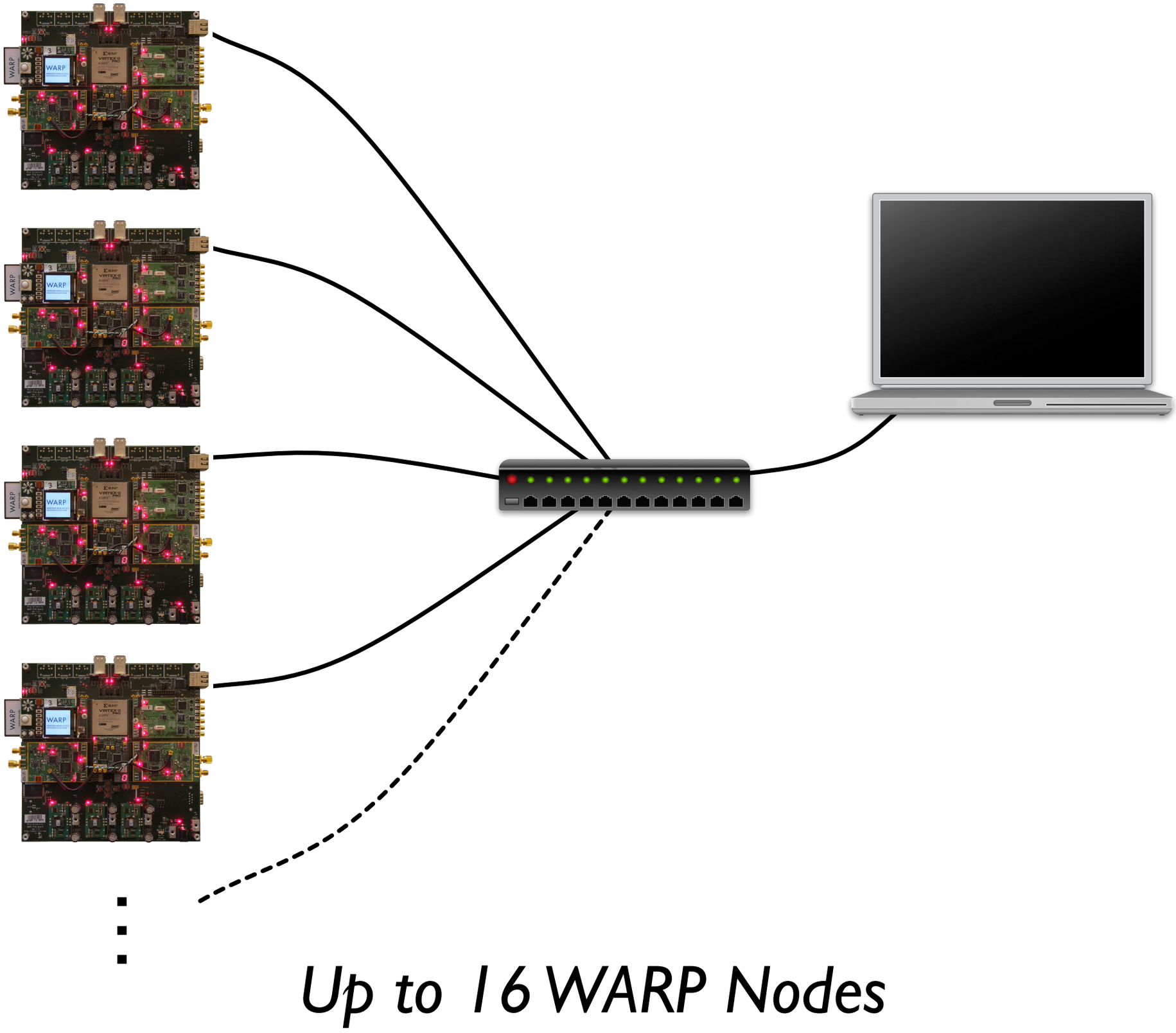






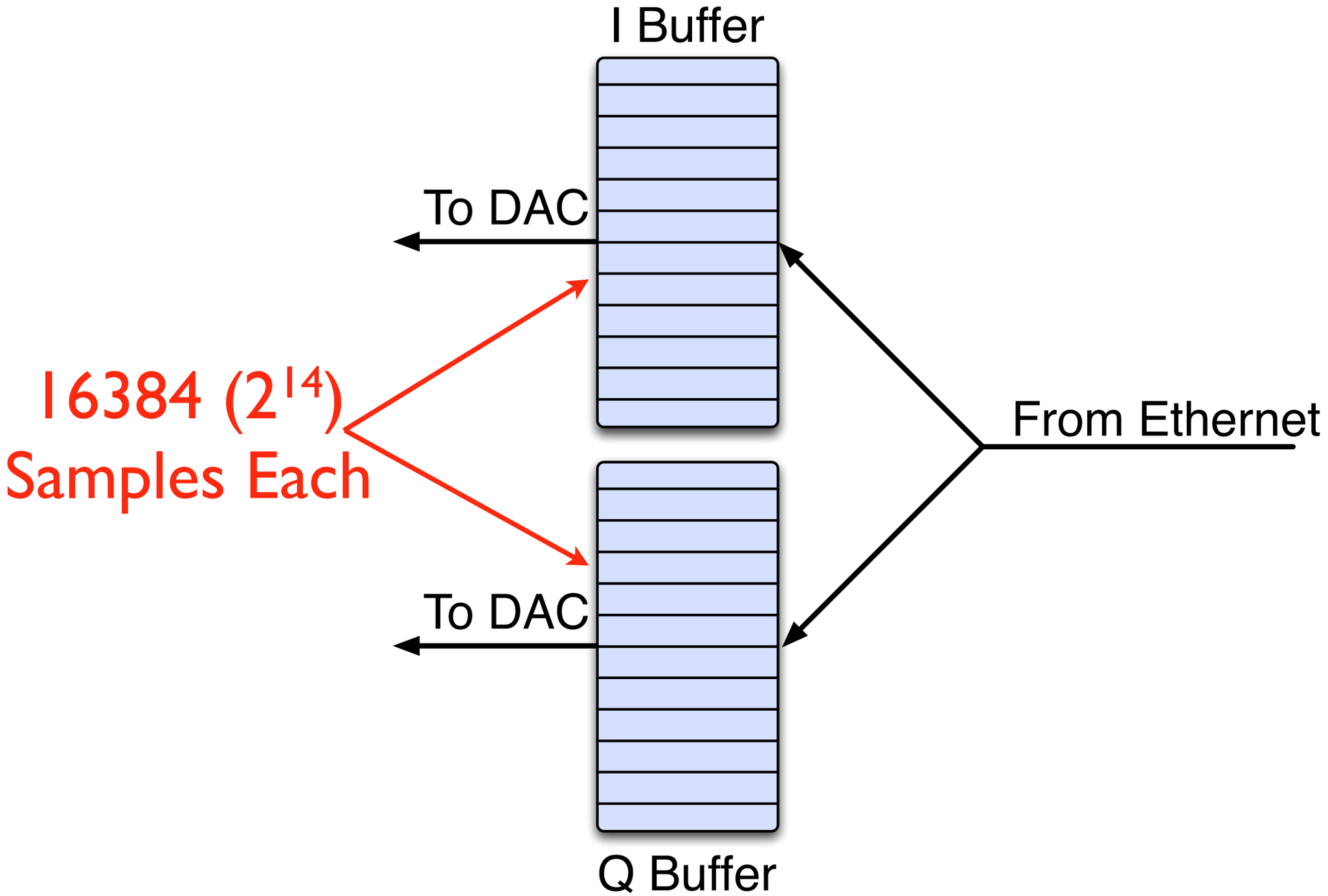
Ethernet links

WARPLab Overview



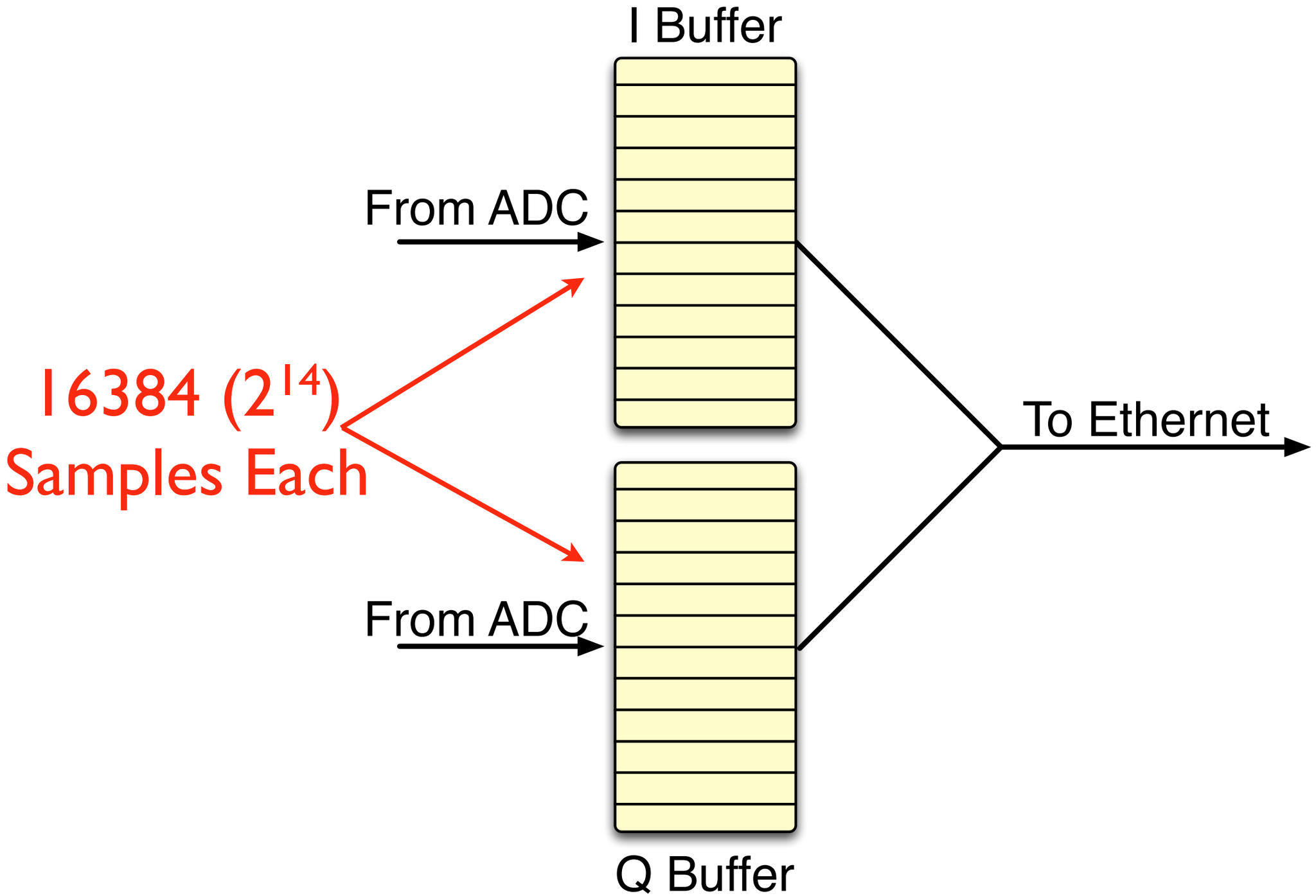
Up to 16 WARP Nodes

WARPLab Architecture



Transmit Path

WARPLab Architecture



Receive Path

WARPLab Architecture

- Independent Tx/Rx buffers
  - Buffers persist between triggers
  - Rx path captures I/Q and RSSI
- Low-level radio control
  - Tx/Rx gains
  - Center frequency
- Control and synchronization by Ethernet

WARPLab Flow

1. Initialize nodes & radio settings
2. Download Tx vectors
3. Enable Tx/Rx radio paths
4. Prime Tx/Rx state machines
5. Trigger the transmission and capture
6. Retrieve Rx vectors

WARPLab Examples

- Hardware characterization
- Channel measurement
- Beamforming
- Cooperative communications

Lab 1: WARPLab

- WARPLab graphical interface
- Measuring the wireless channel
- Building a real bits-to-RF transmitter

PHY Design Flows

- WARPLab
  - MATLAB↔WARP Link
  - Very rapid prototyping of PHY algorithms
- Real-time PHY design
  - Low-level FPGA design
  - Putting it all together

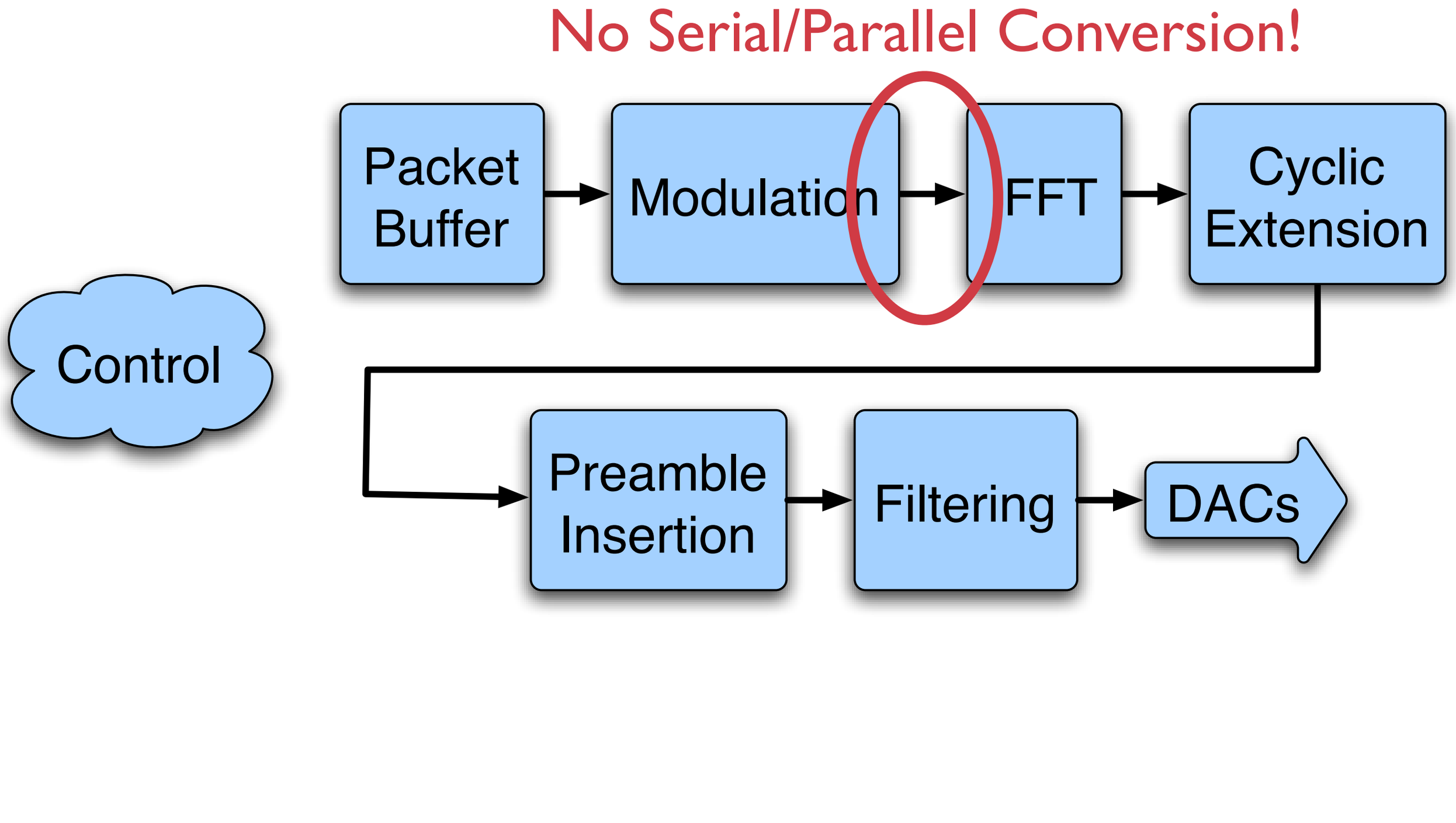
PHY Design Decisions

- Packet-based or streaming?
- Wideband or narrowband?
  - Pipelined vs. iterative
  - Think clock cycles per channel sample
- Complete synchronization or "cheating"?
  - Independent vs. common reference clocks
  - Packet detection vs. shared timing signals
  - Real channel vs. coax cable
- Runtime vs. hard-coded parameters

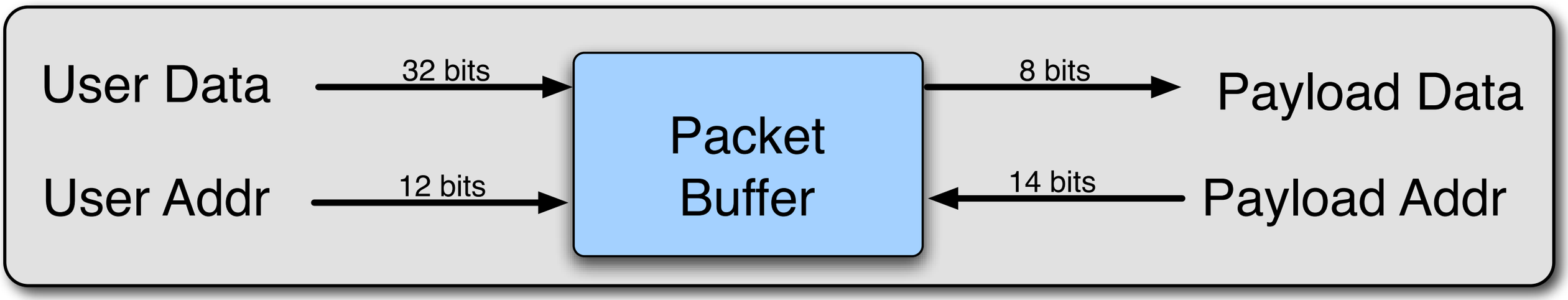
PHY Example: OFDM

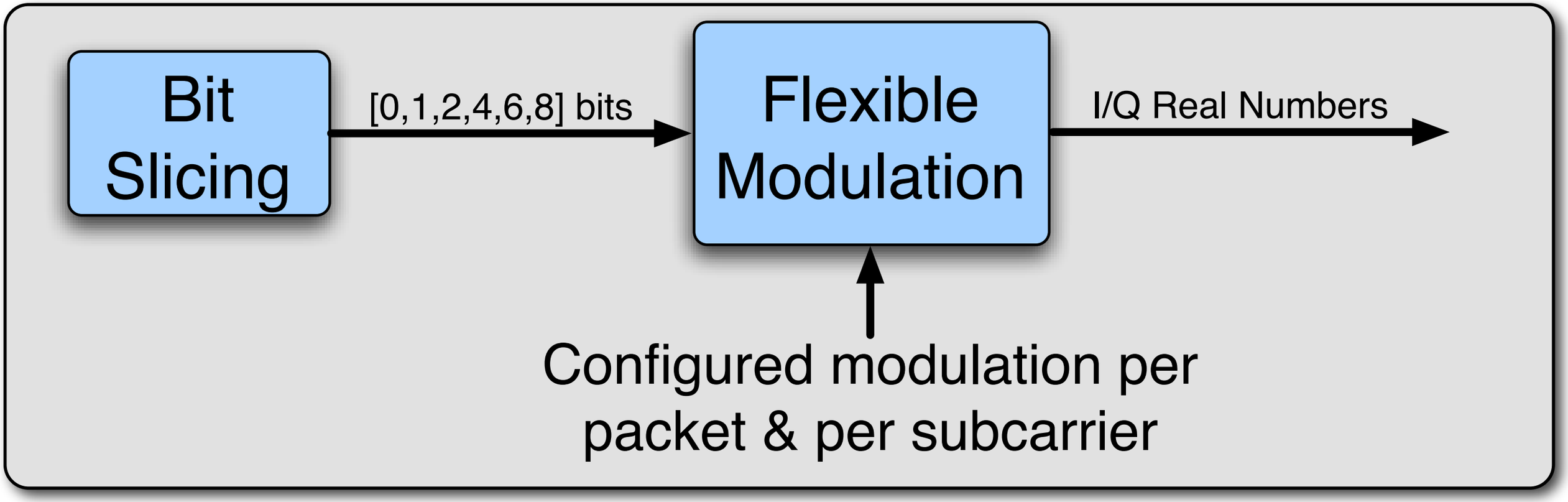
- Designed for wireless networking
  - Modeled on 802.11a (but not compliant)
- Packet-based OFDM transceiver
  - Packets source/sink in PowerPC code
- Wideband, real-time design
  - 5 cycles per sample
  - 10 MHz bandwidth at 50 MHz clock
- Full synchronization for standalone operation
- Implemented entirely in System Generator

PHY Example: OFDM Tx

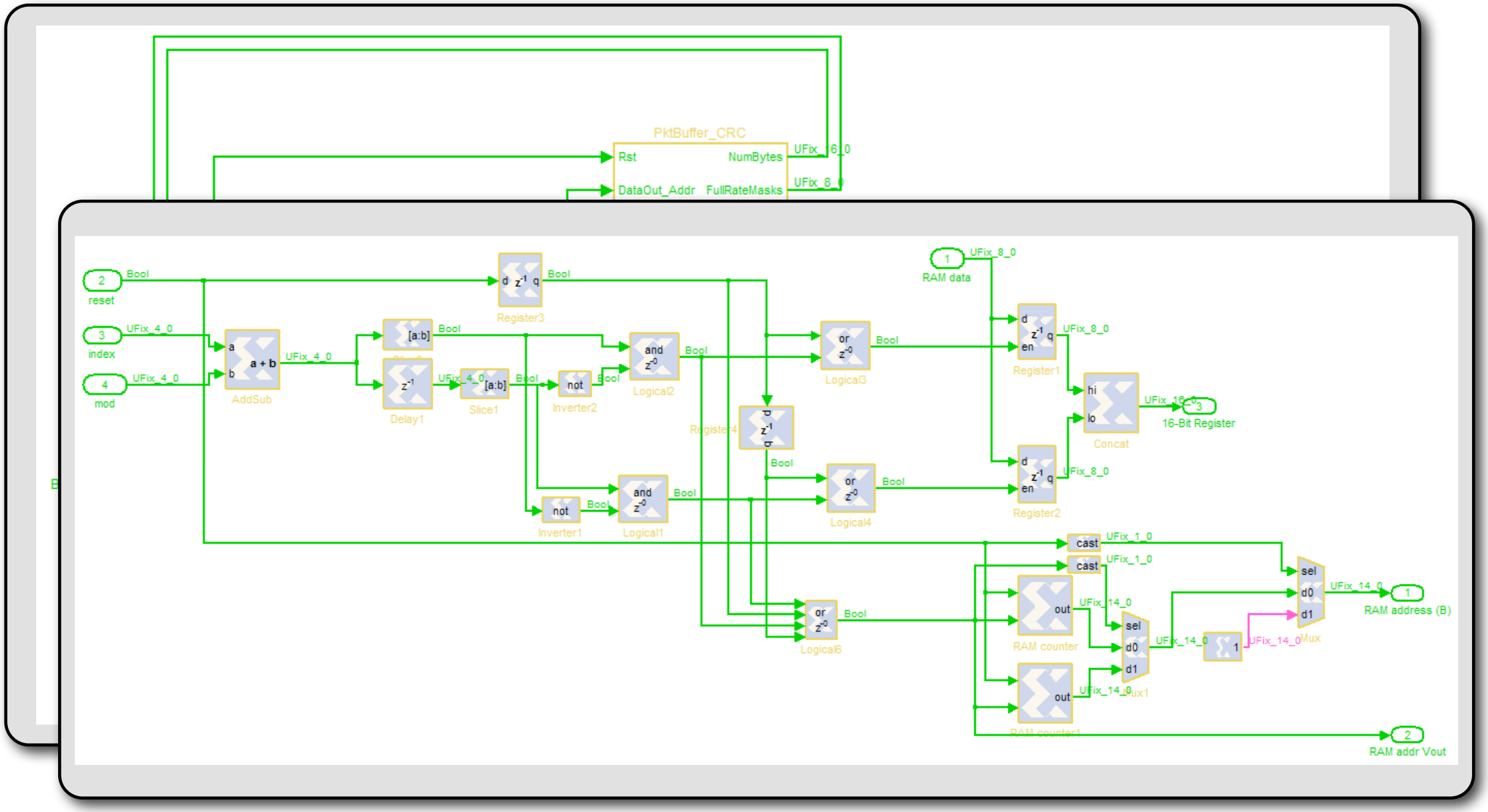


PHY Example: OFDM Tx



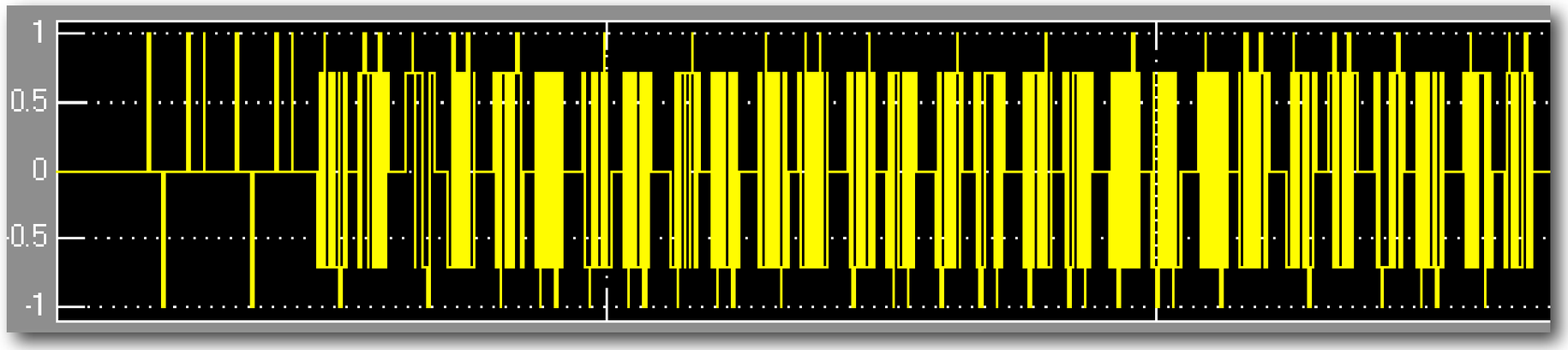


PHY Example: OFDM Tx



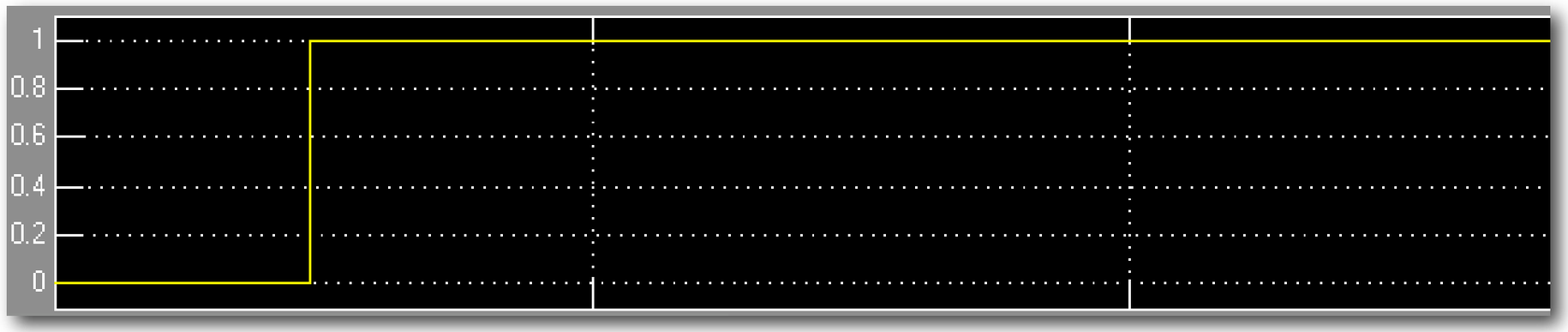
PHY Example: OFDM Tx

Modulator Output

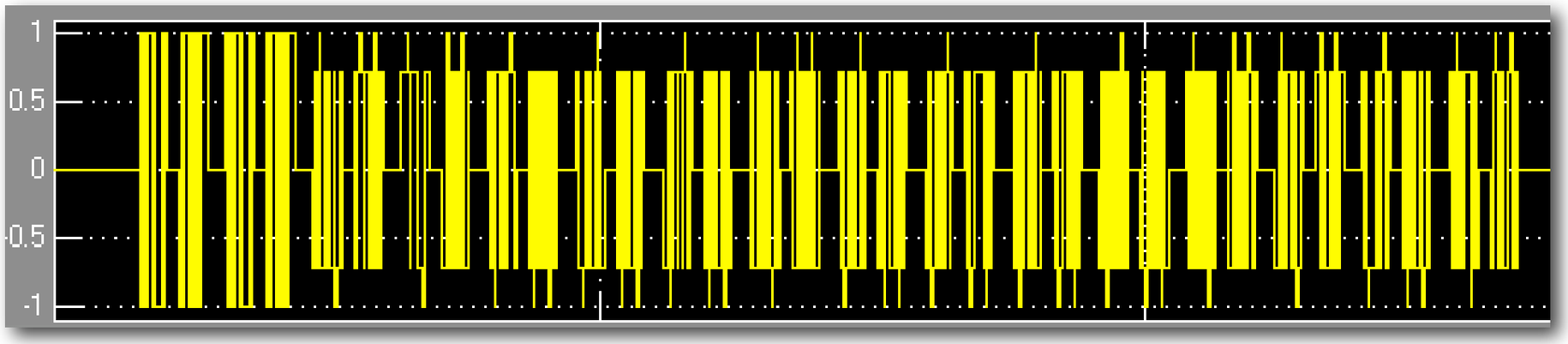


Stored Training Sequence

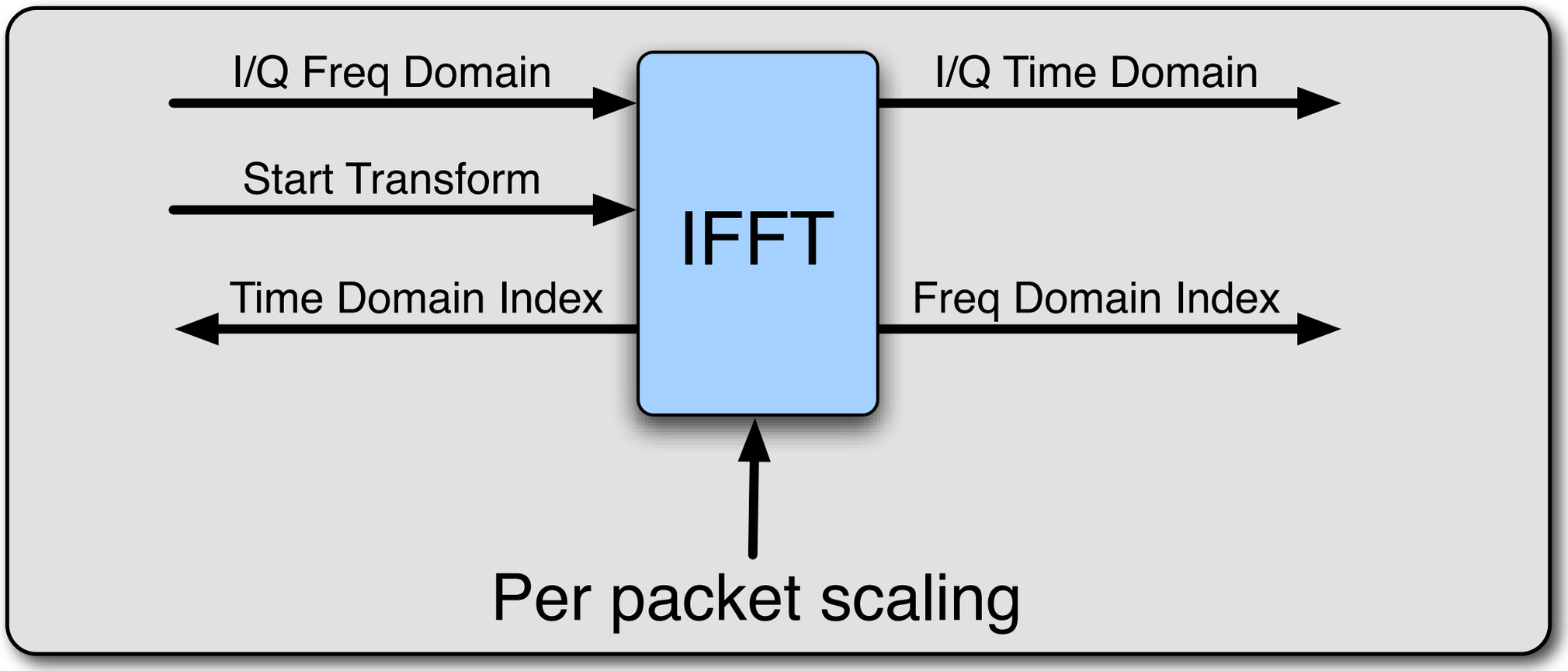
Source Mux Select



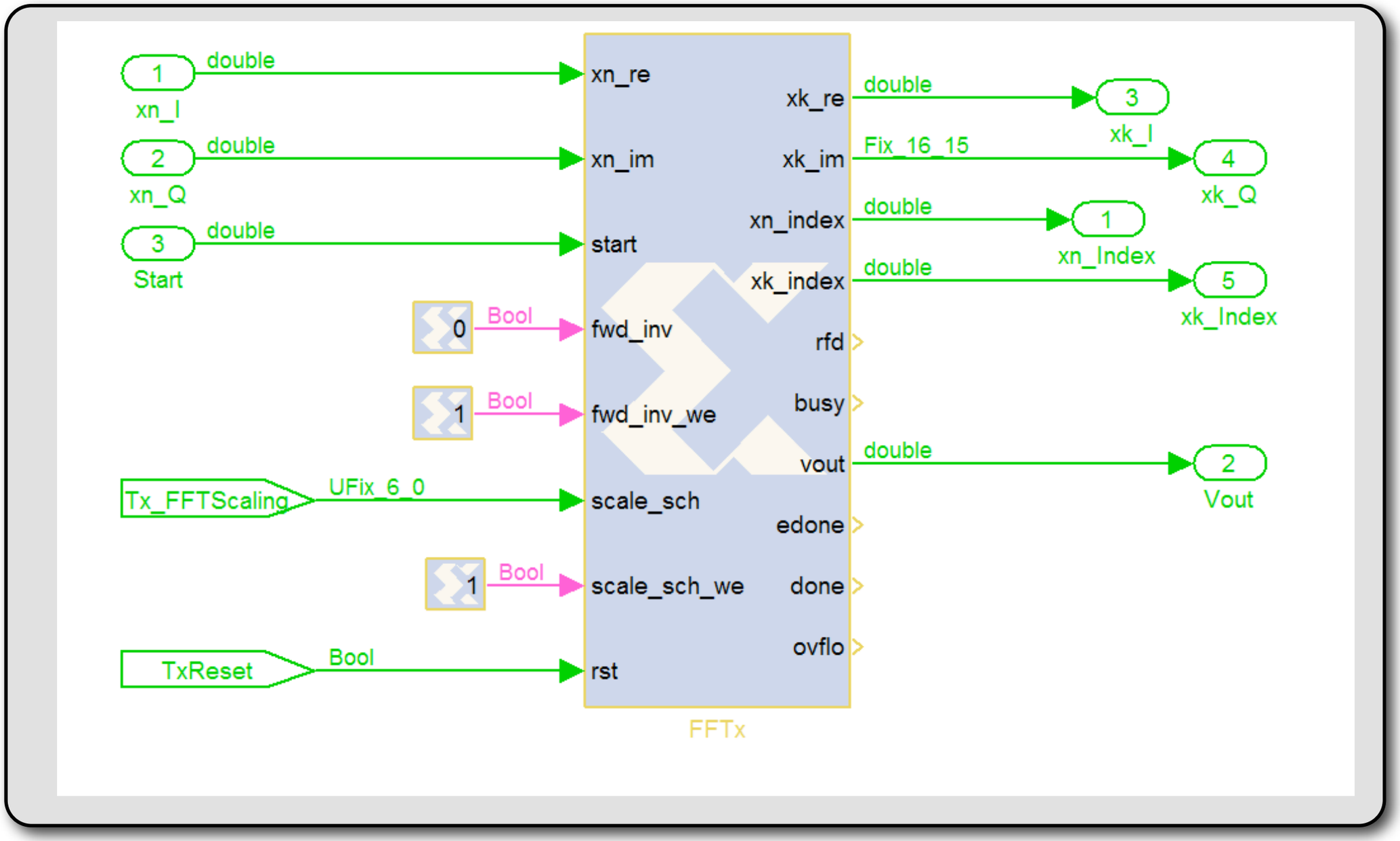
Input IFFT



PHY Example: OFDM Tx



PHY Example: OFDM Tx



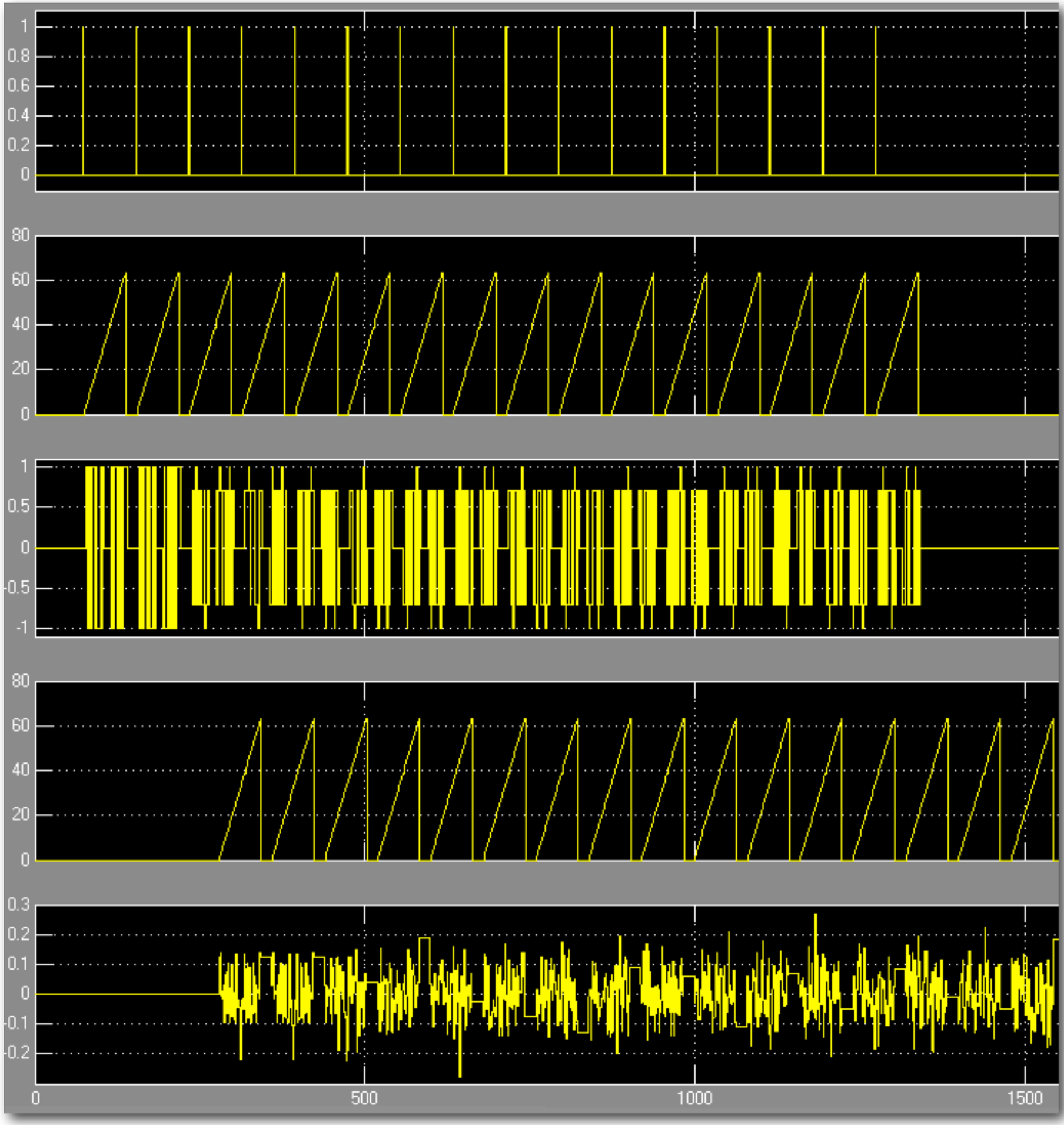
IFFT Start

Input Index

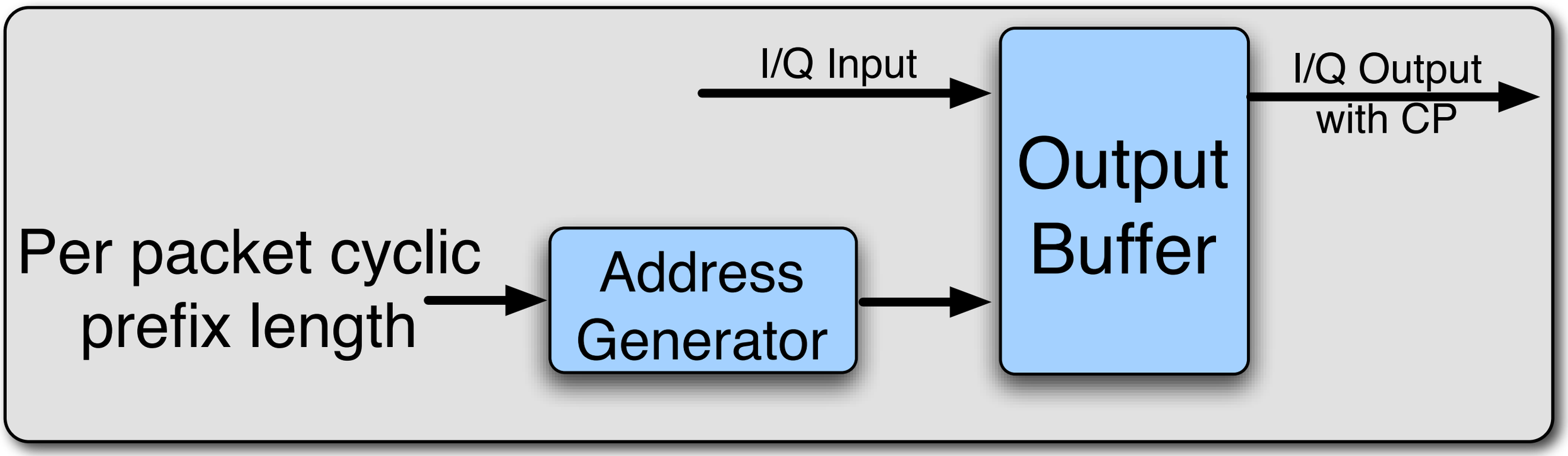
Input Data

Output Index

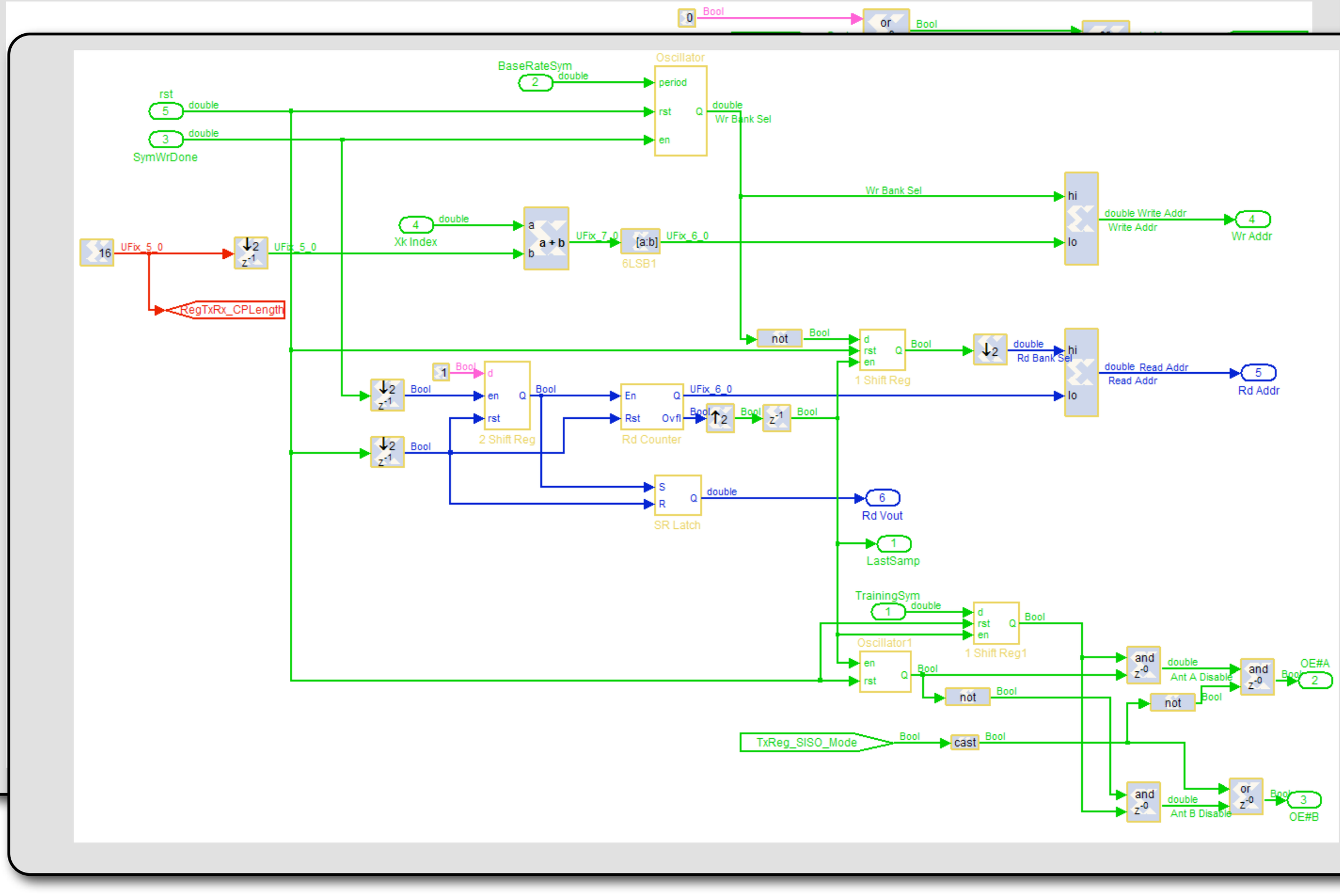
Output Data



PHY Example: OFDM Tx

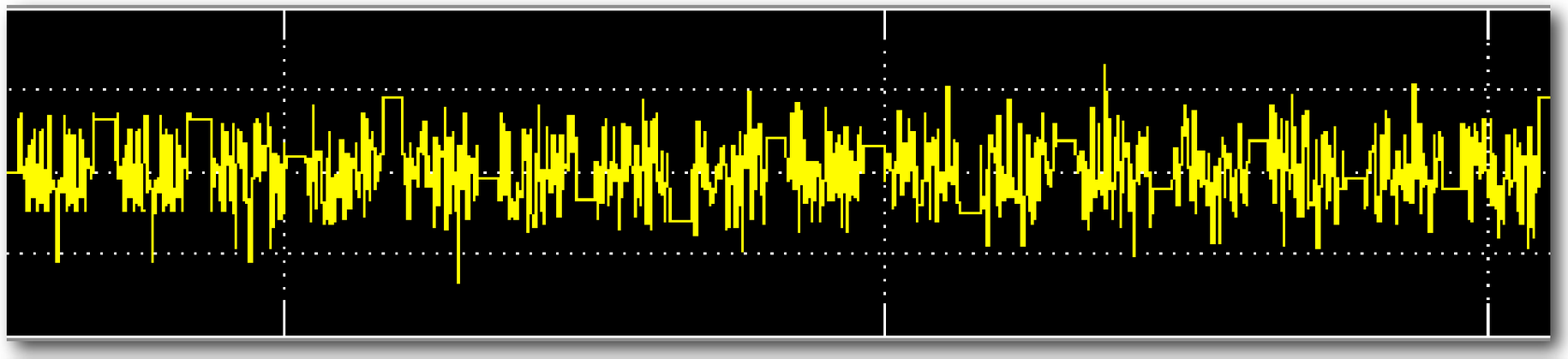


PHY Example: OFDM Tx

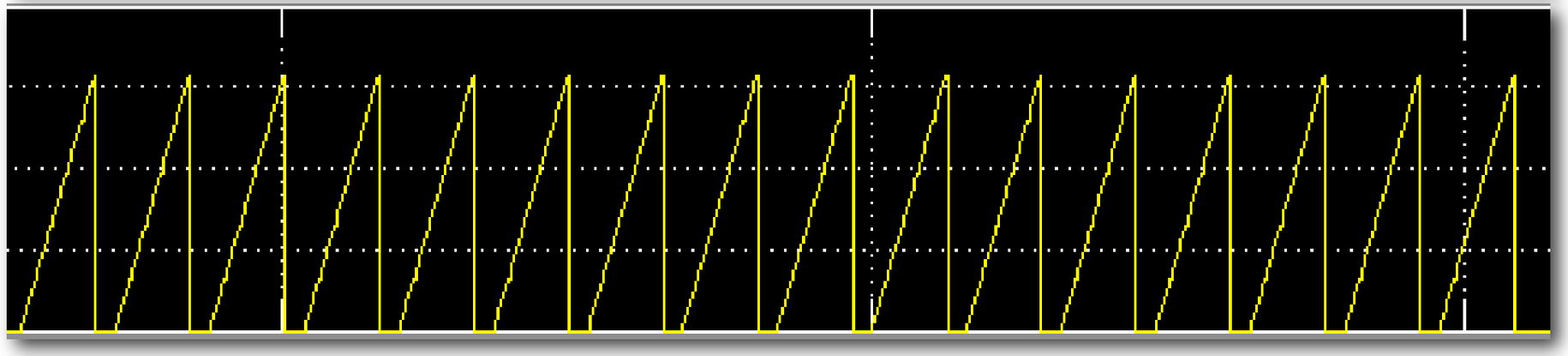


PHY Example: OFDM Tx

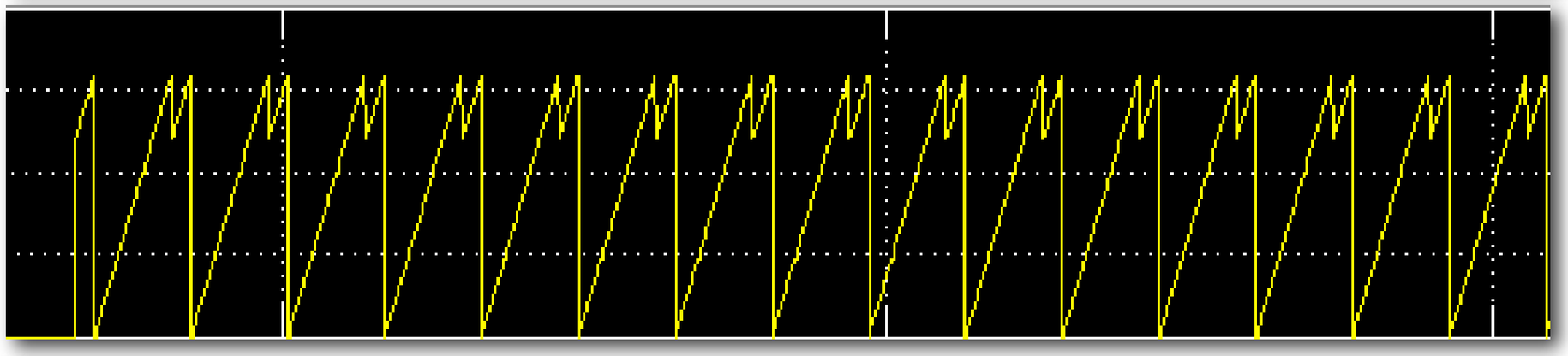
IFFT Output



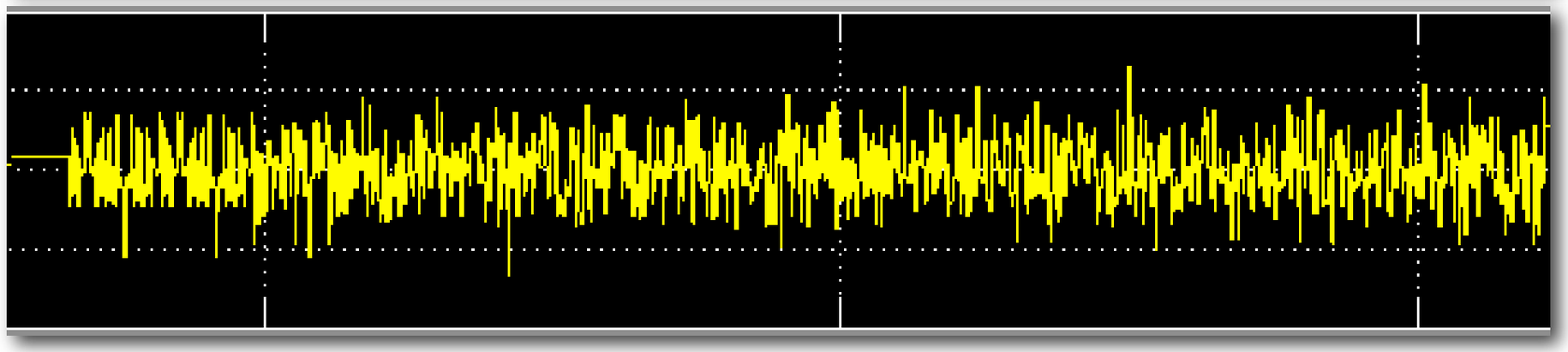
RAM Write Address



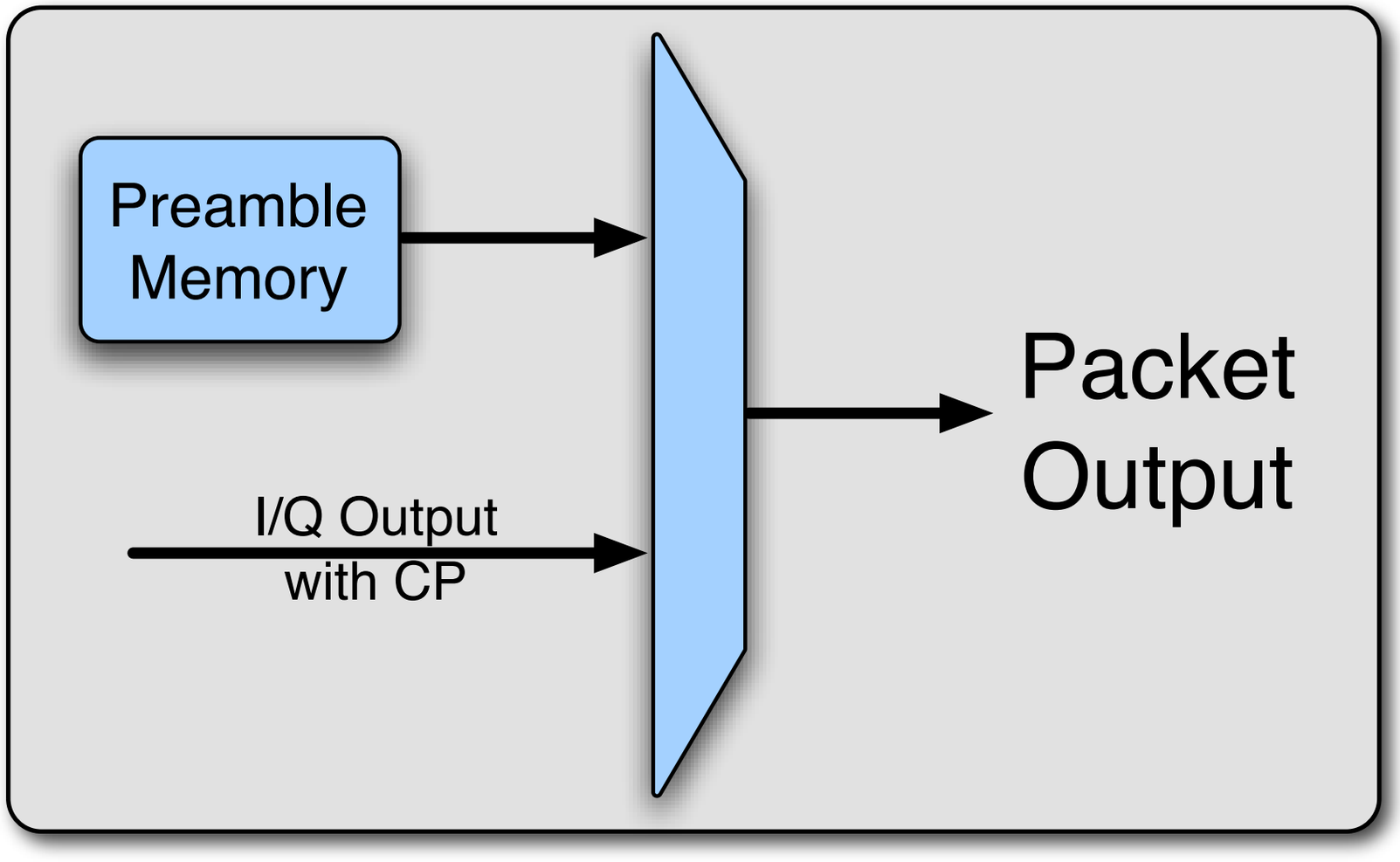
RAM Read Address



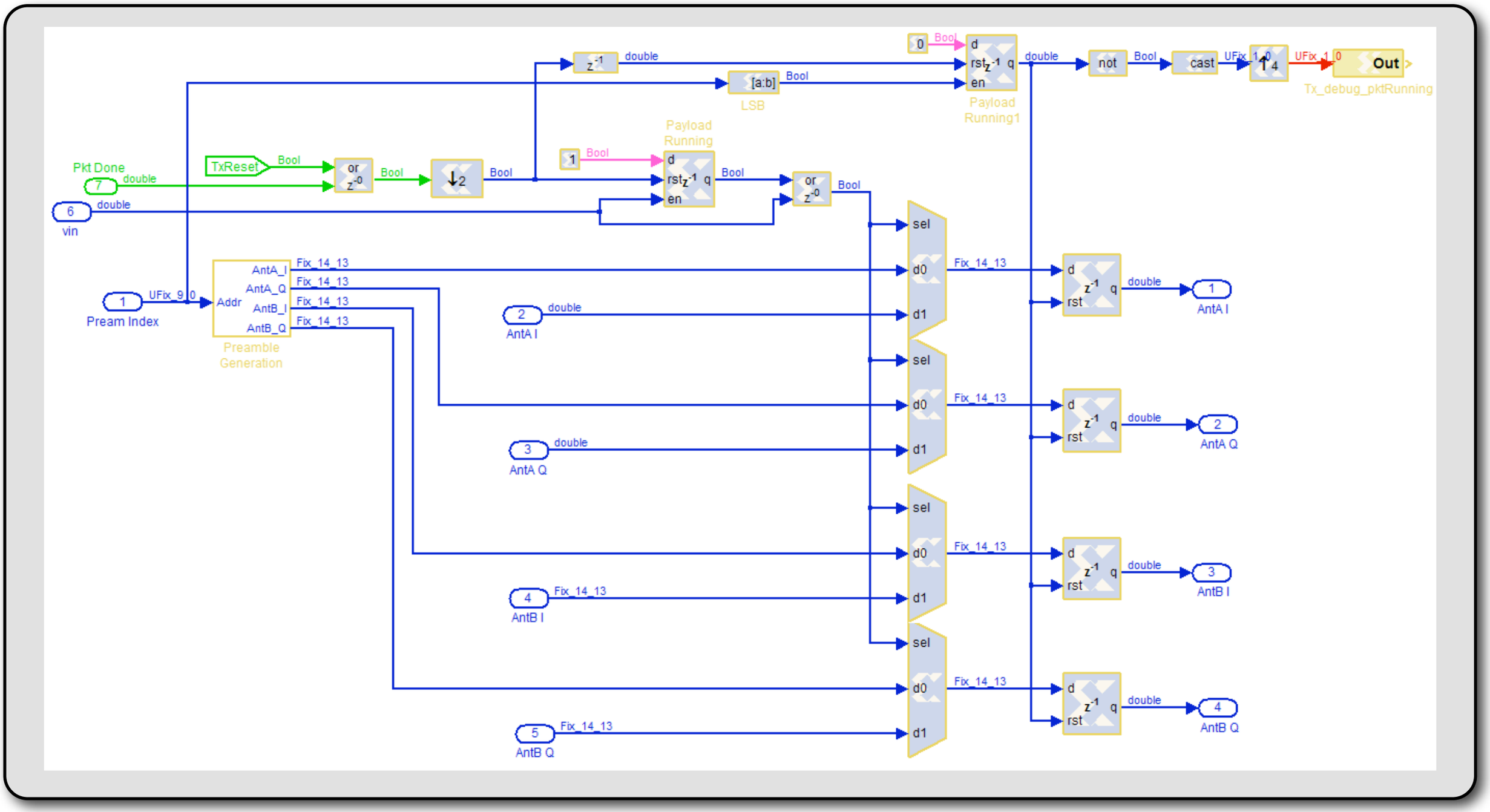
Cyclically Extended



PHY Example: OFDM Tx

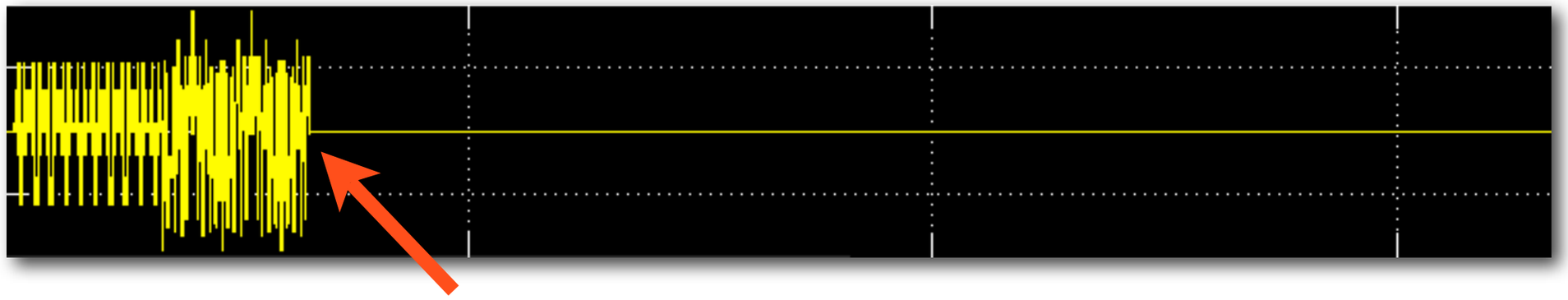


PHY Example: OFDM Tx

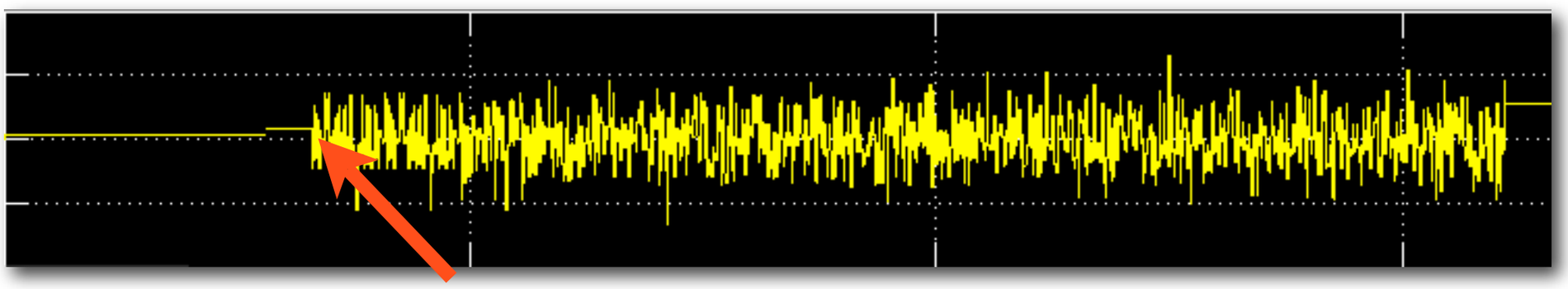


PHY Example: OFDM Tx

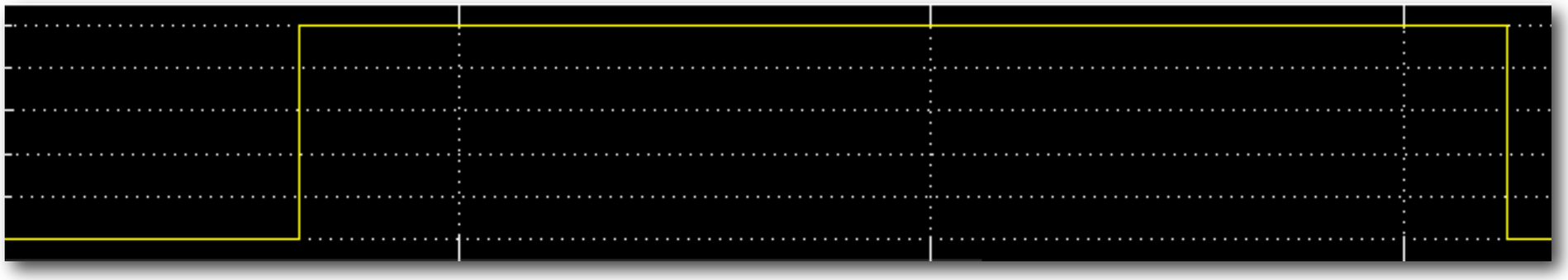
Stored Preamble



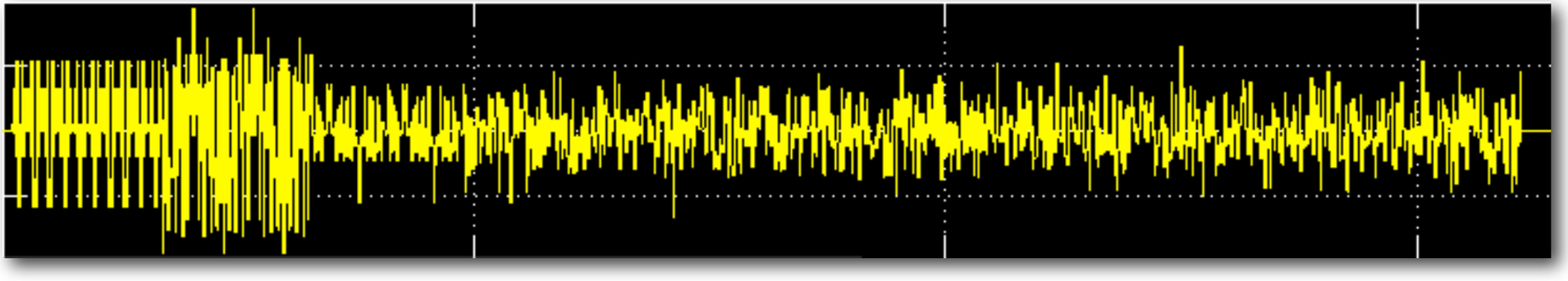
OFDM Output



Output Mux Selection



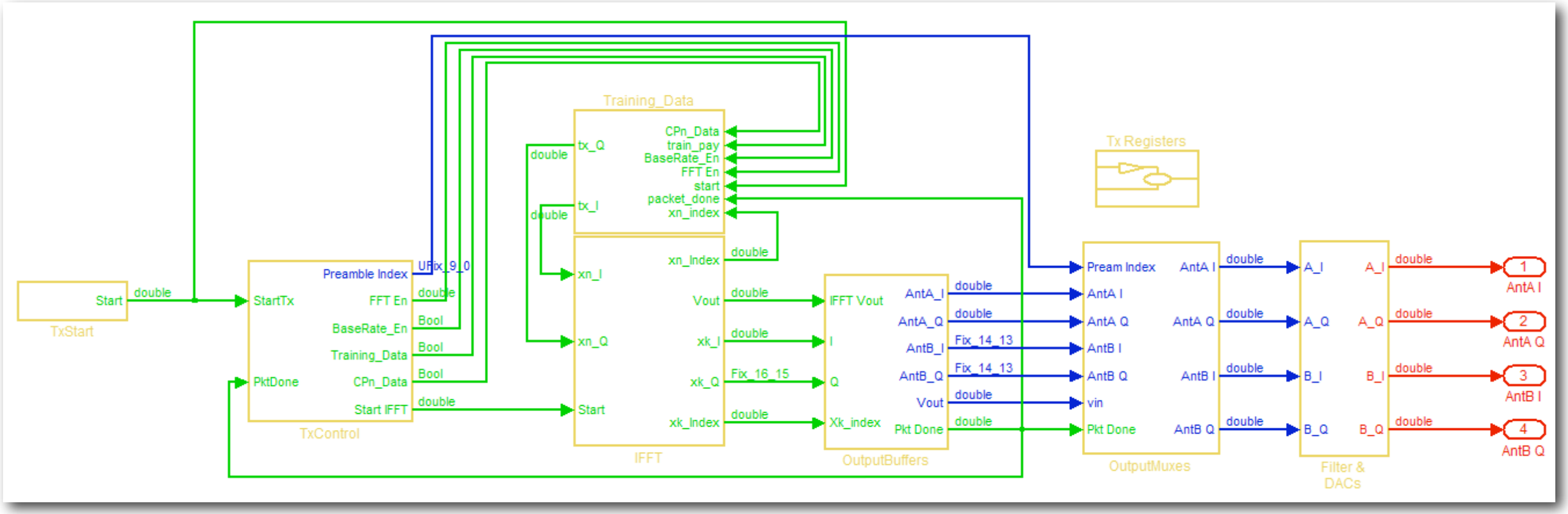
Final Output to DACs



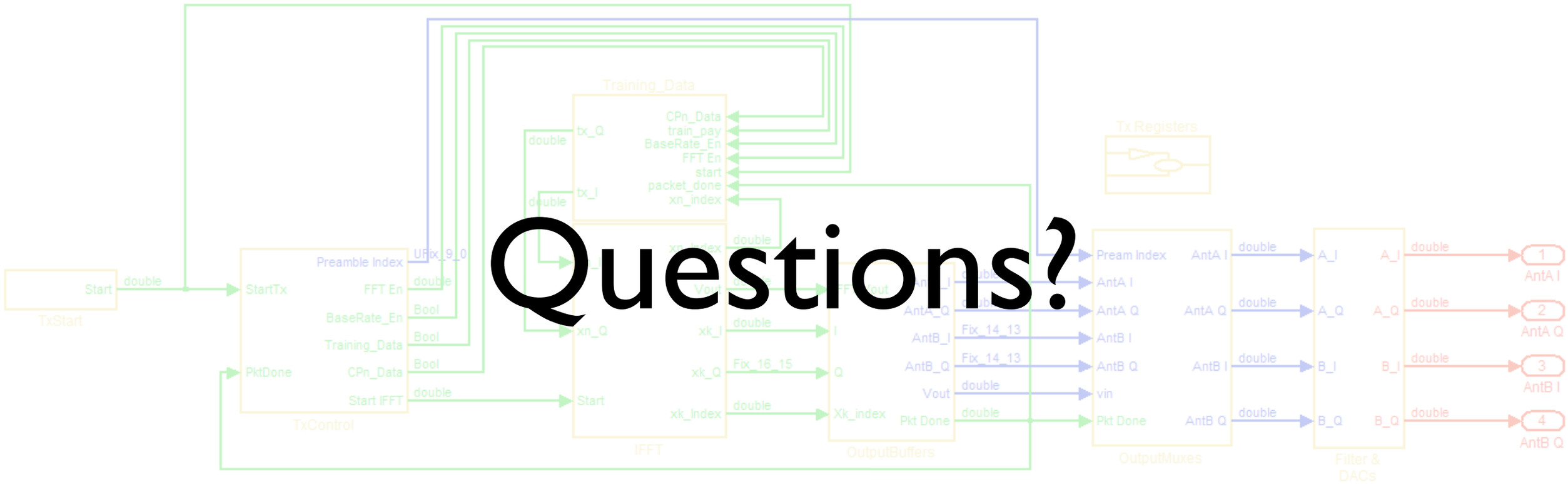
PHY Example: OFDM Tx

- Per packet configuration drives control system
  - Number of training symbols
  - Number of bytes
  - Modulation choices
- Block specific control blocks
  - IFFT start signal
  - Memory address generation
- Triggers & status between core and PowerPC

PHY Example: OFDM Tx



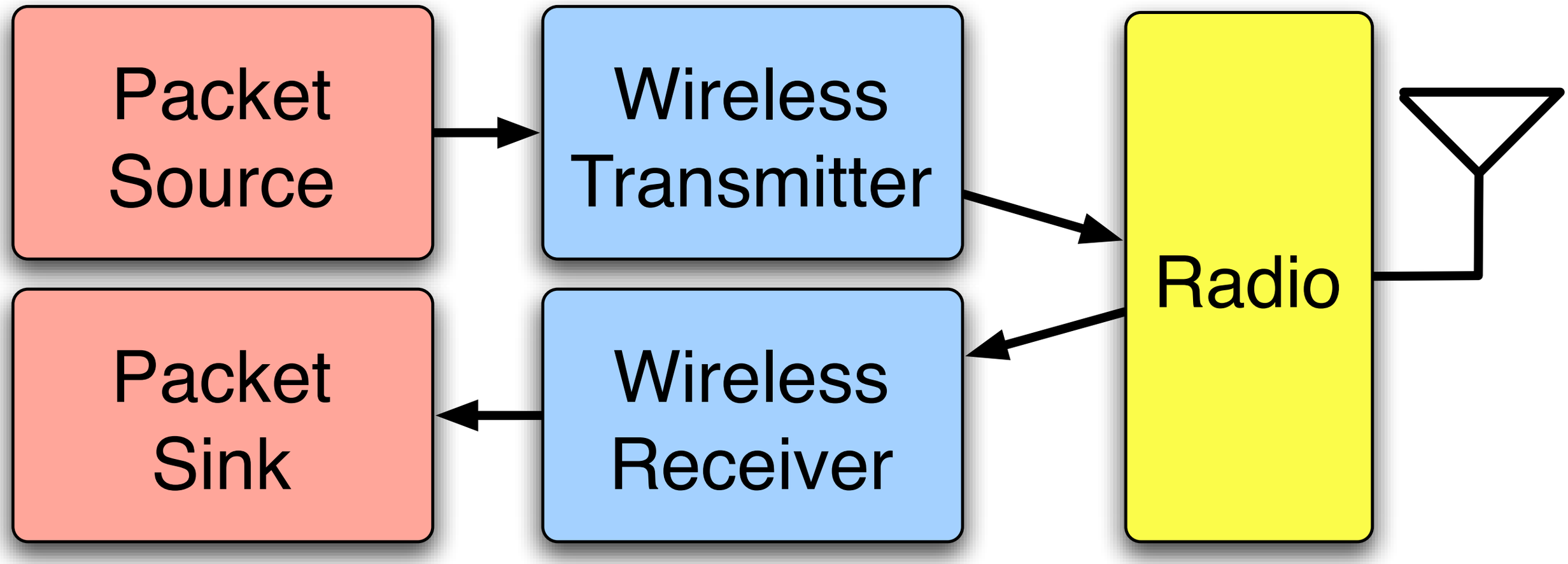
Complete model is available in the WARP repository


Training_Data
CPn_Data
train_pay
BaseRate_En
FFT En
start
packet_done
xn_index
tx_Q
tx_I
double
Tx Registers
Start
TxStart
StartTx
Preamble Index
FFT En
BaseRate_En
Training_Data
CPn_Data
Start IFFT
PktDone
TxControl
Ufix_9_0
Bool
Bool
xn_Q
xn_I
Start
IFFT
xk_Q
xk_I
xk_Index
Fix_16_15
Q
I
Xk_index
OutputBuffers
AntB_I
AntB_Q
Vout
Pkt Done
Fix_14_13
Pream Index
AntA I
AntA Q
AntB I
AntB Q
vin
Pkt Done
OutputMuxes
A_I
A_Q
B_I
B_Q
Filter &
DACs
1
AntA I
2
AntA Q
3
AntB I
4
AntB Q

Questions?

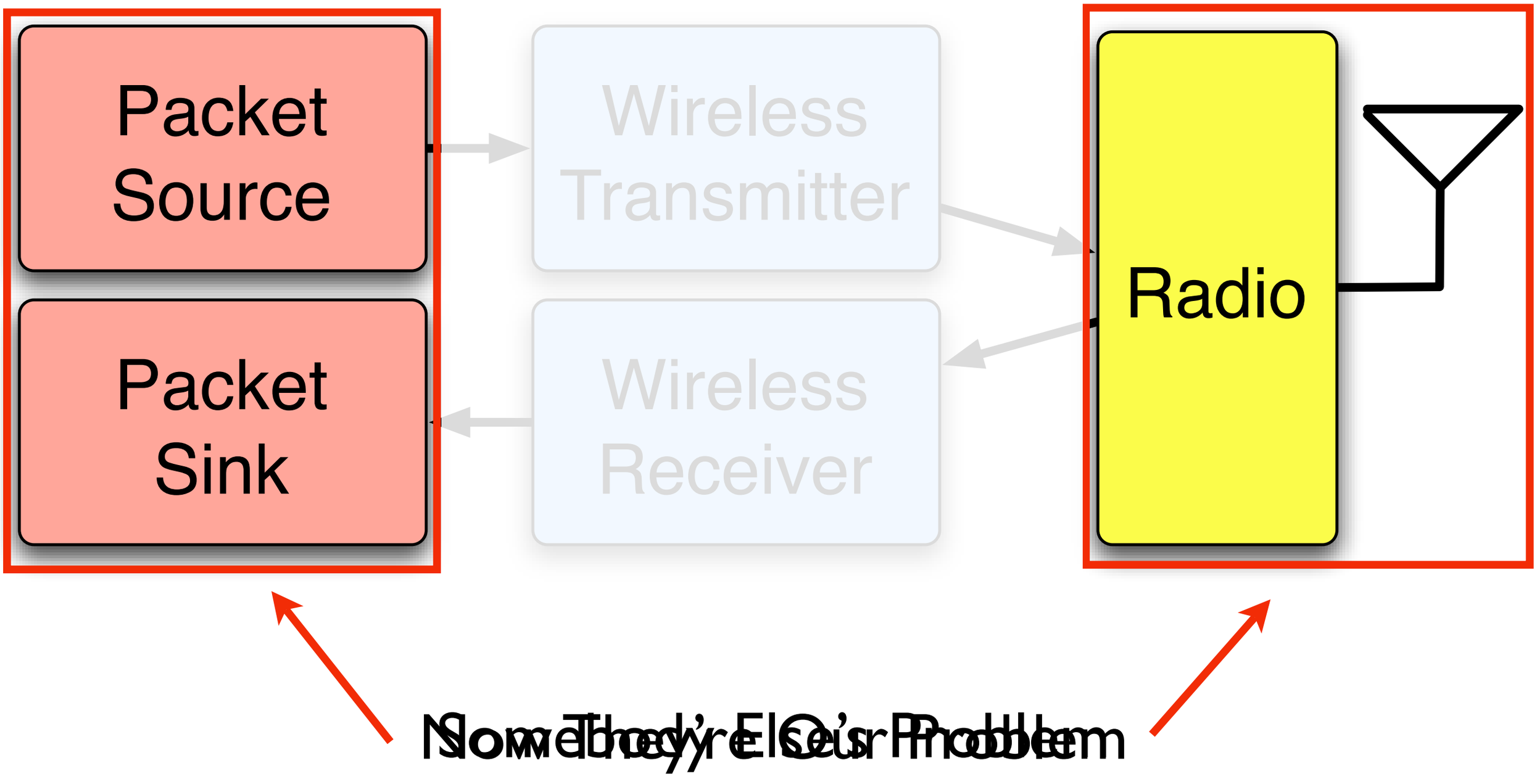
Physical Layer Basics

Simple Wireless Node

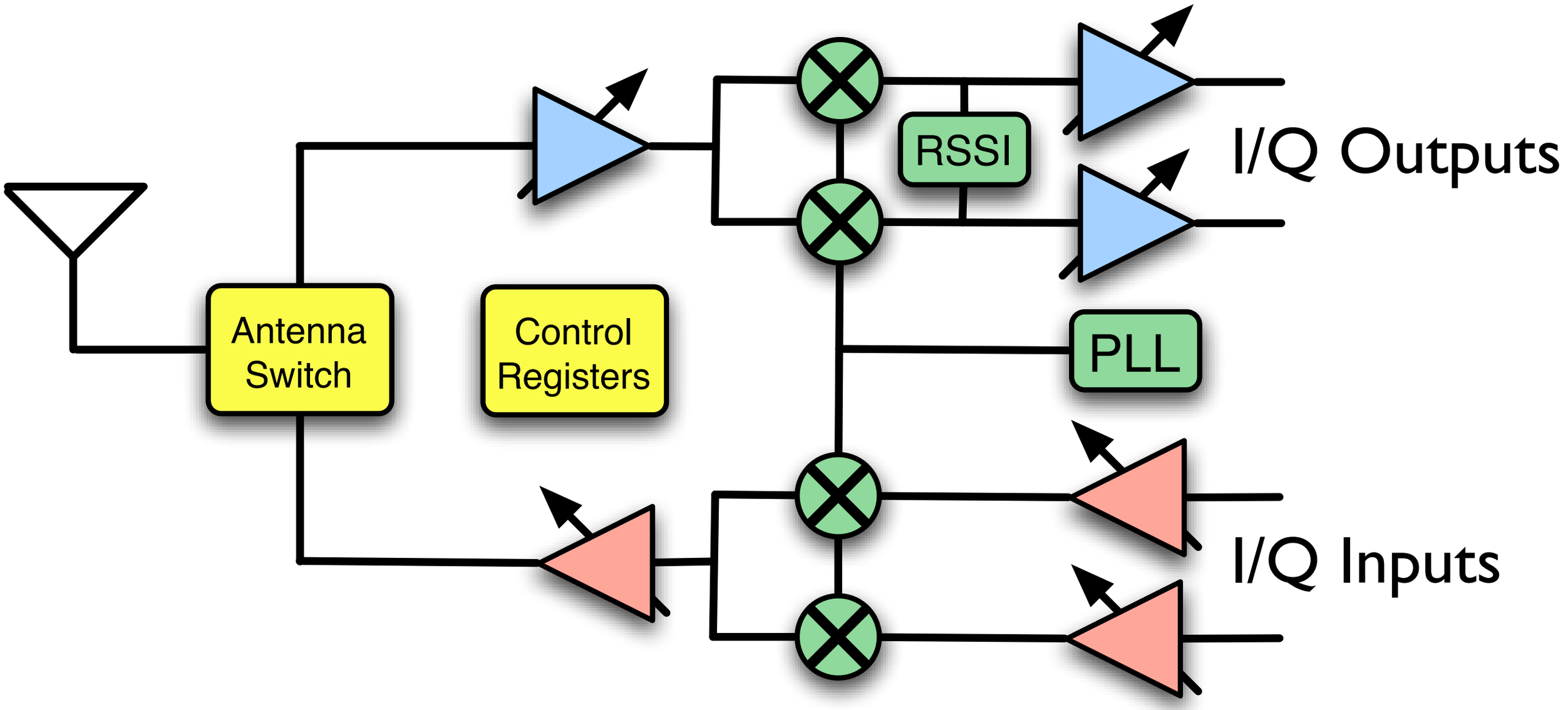


Physical Layer Basics

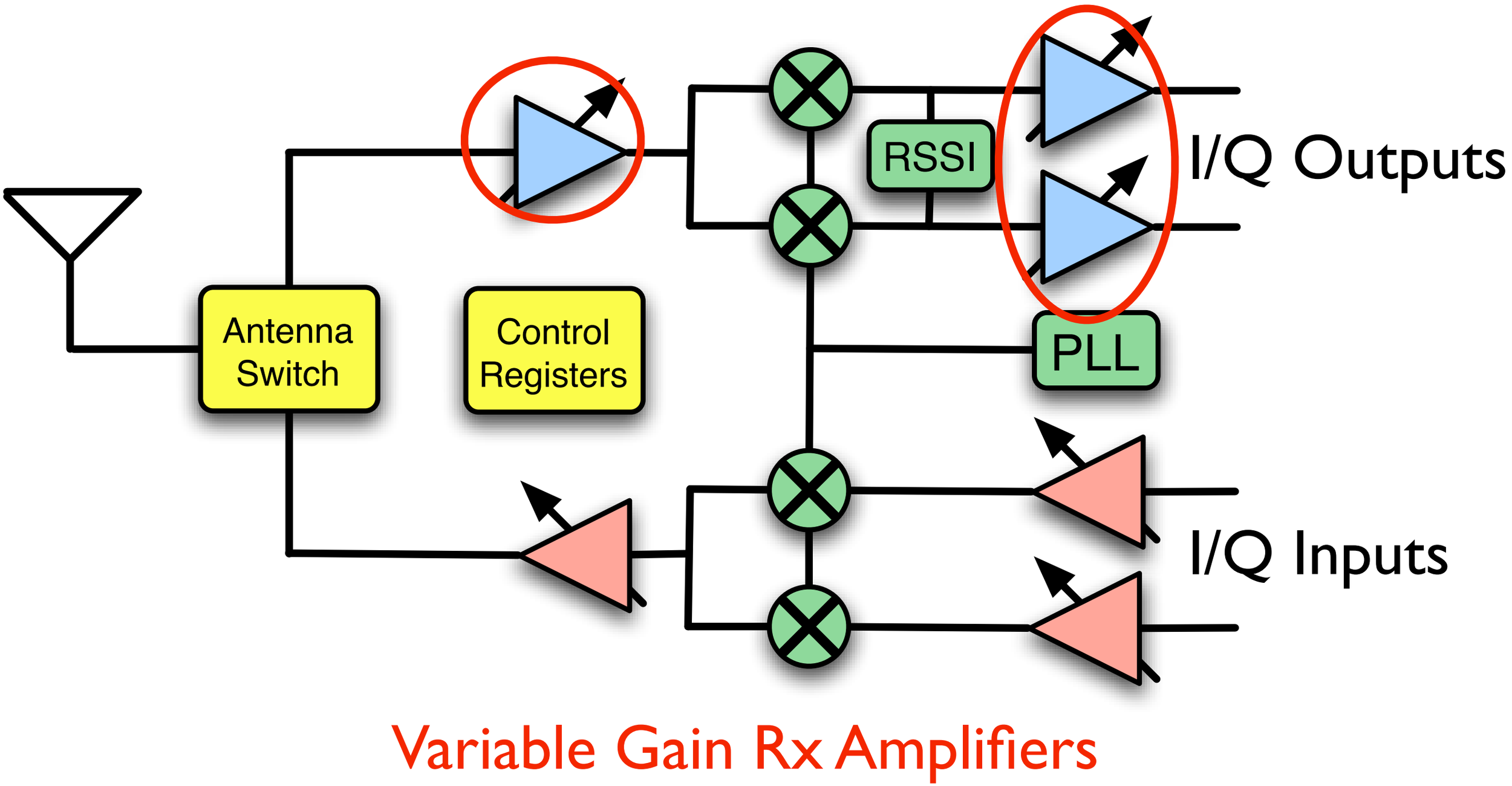
Simple Wireless Node



Radio Transceiver

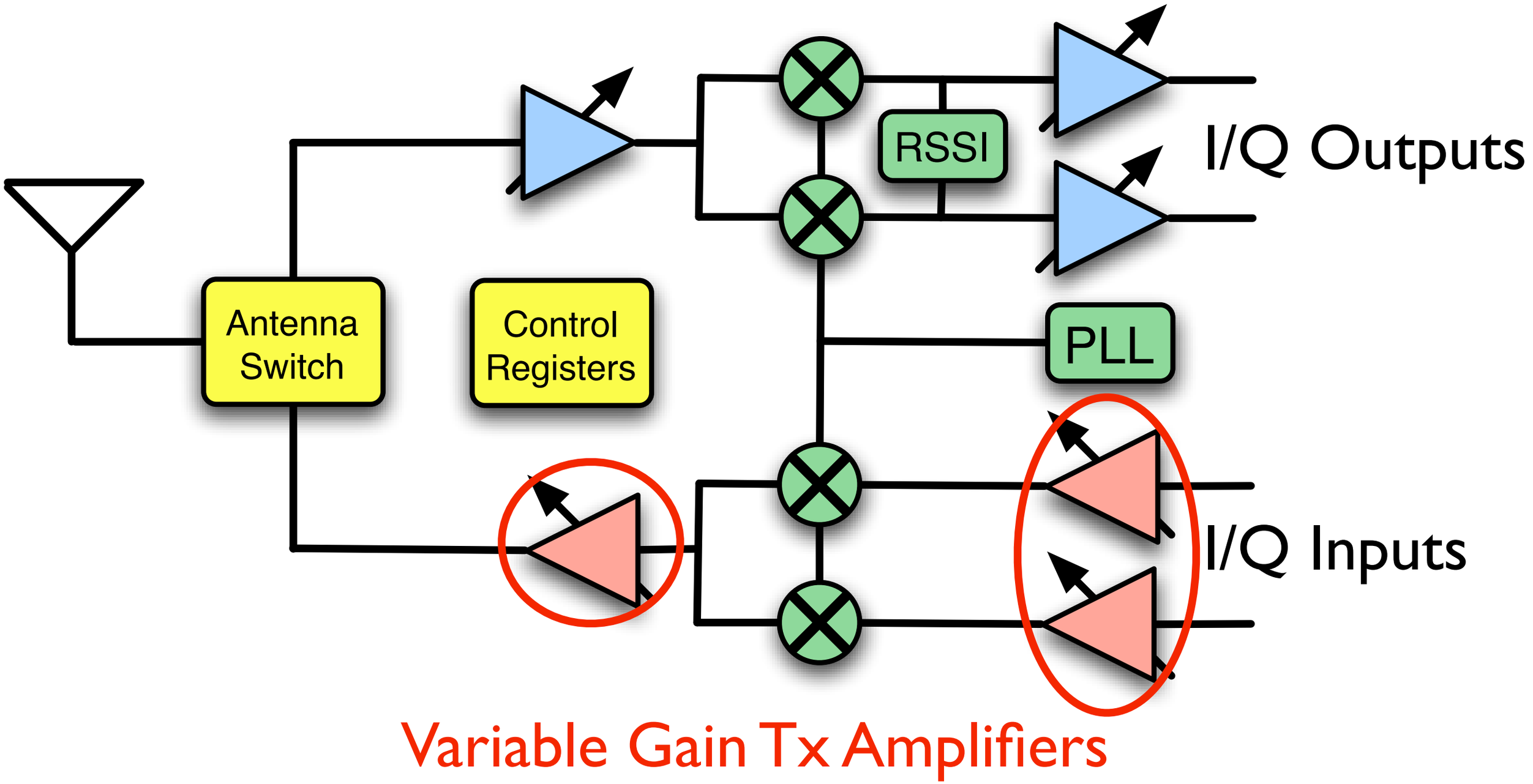


Radio Transceiver

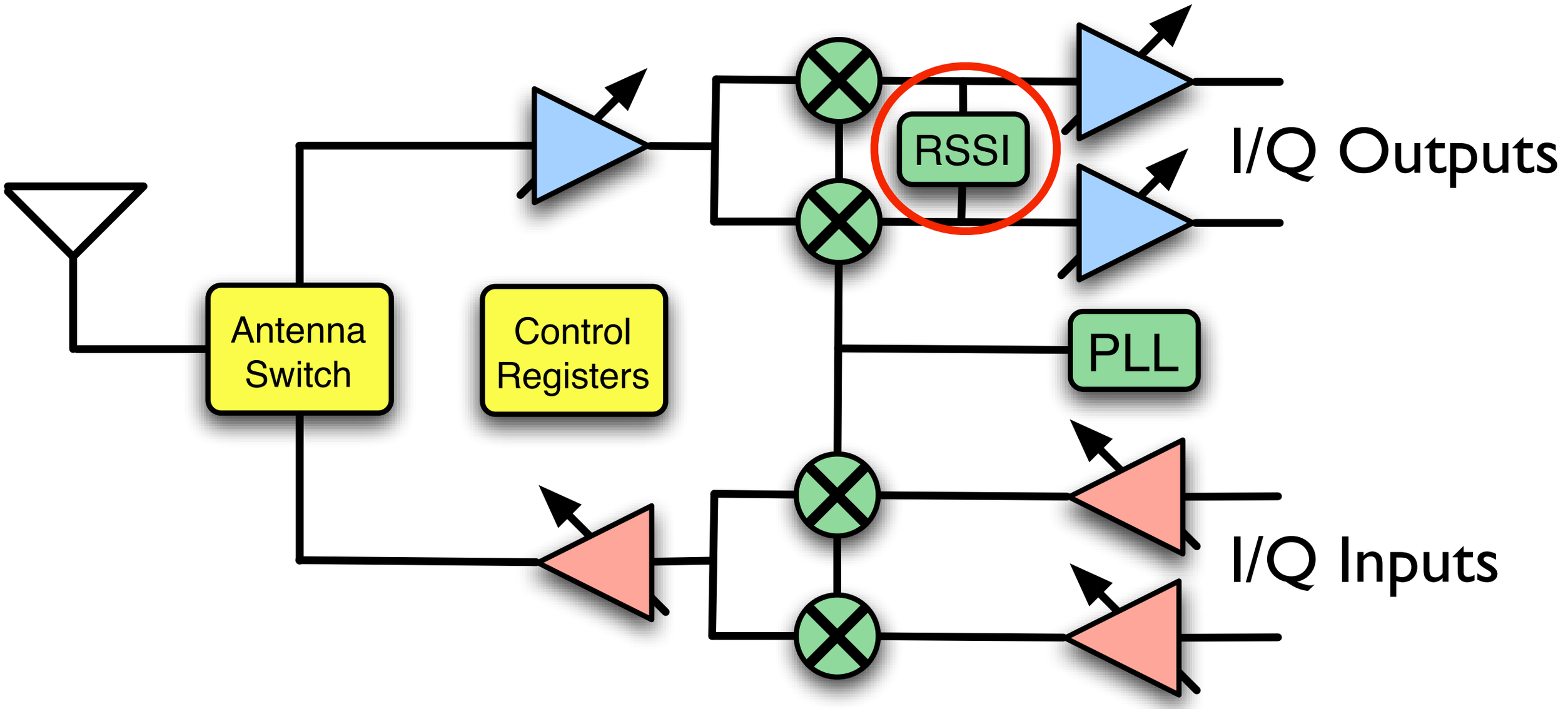


Variable Gain Rx Amplifiers

Radio Transceiver

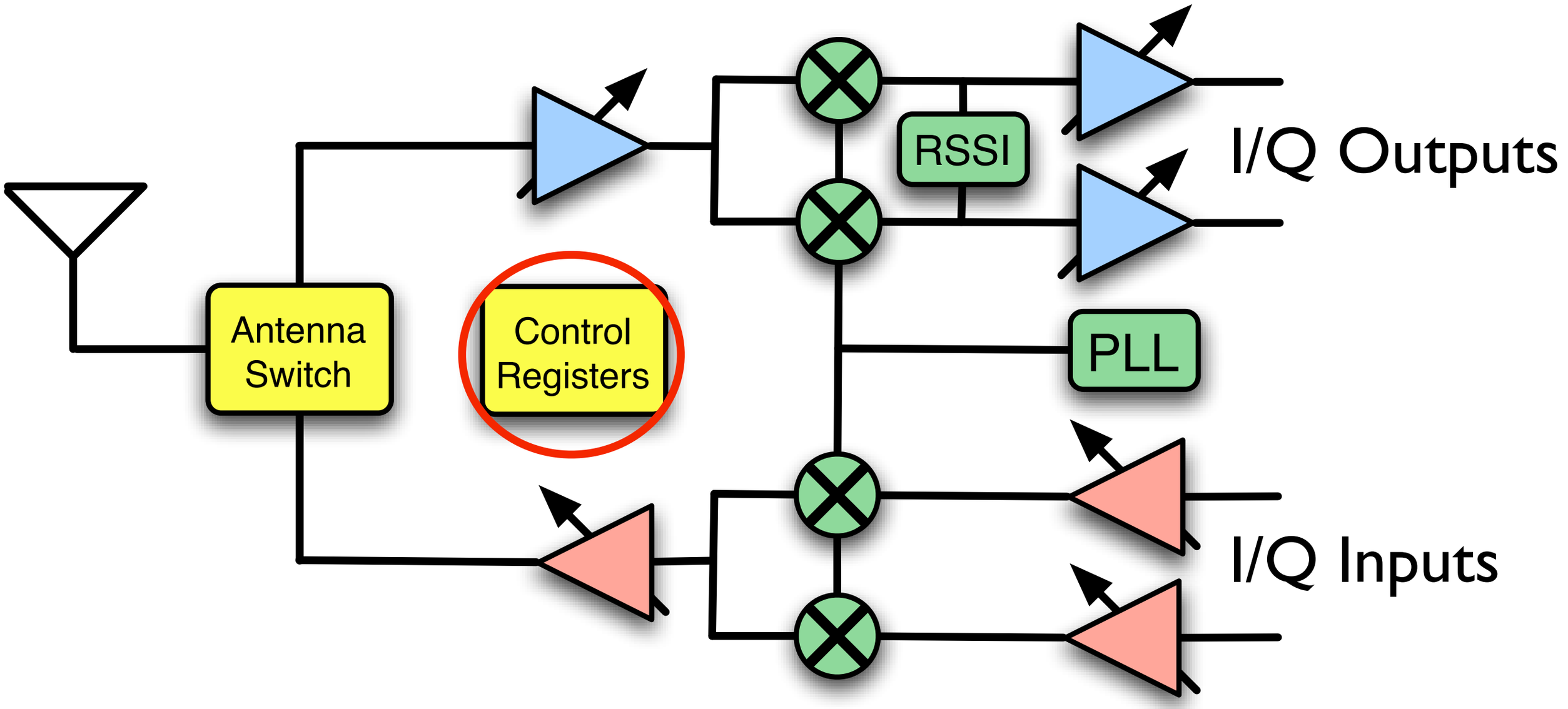


Radio Transceiver



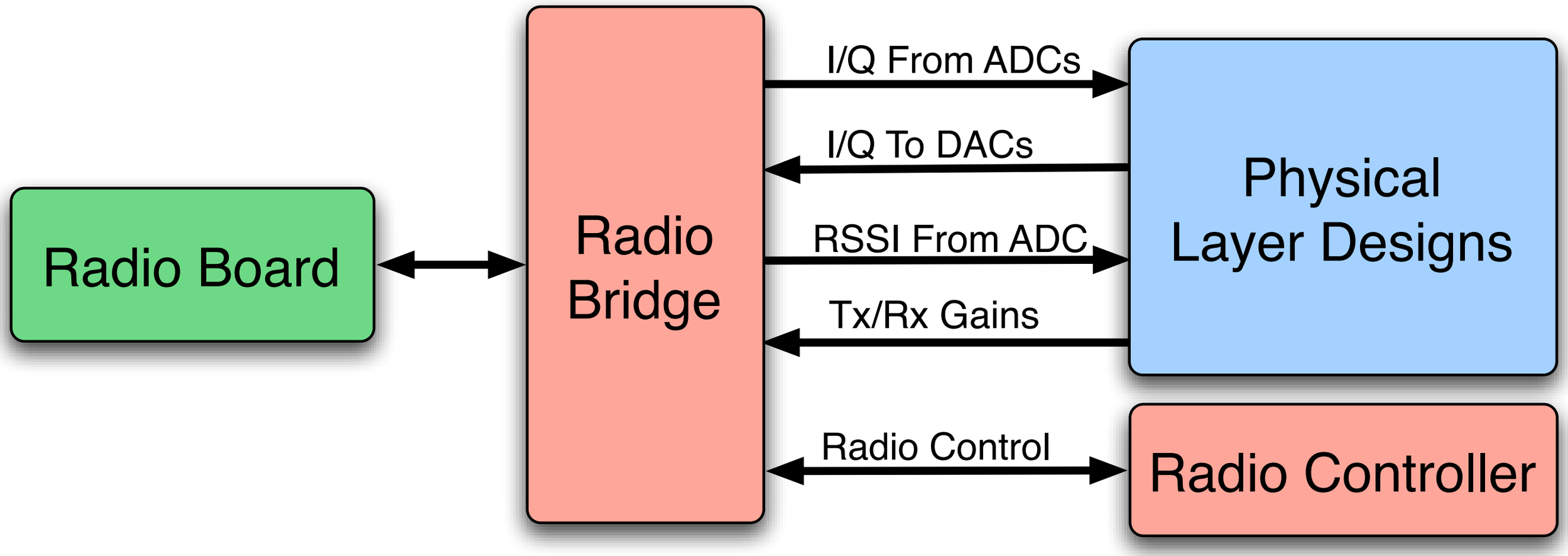
Received Signal Strength Indicator
After RF Gain

Radio Transceiver

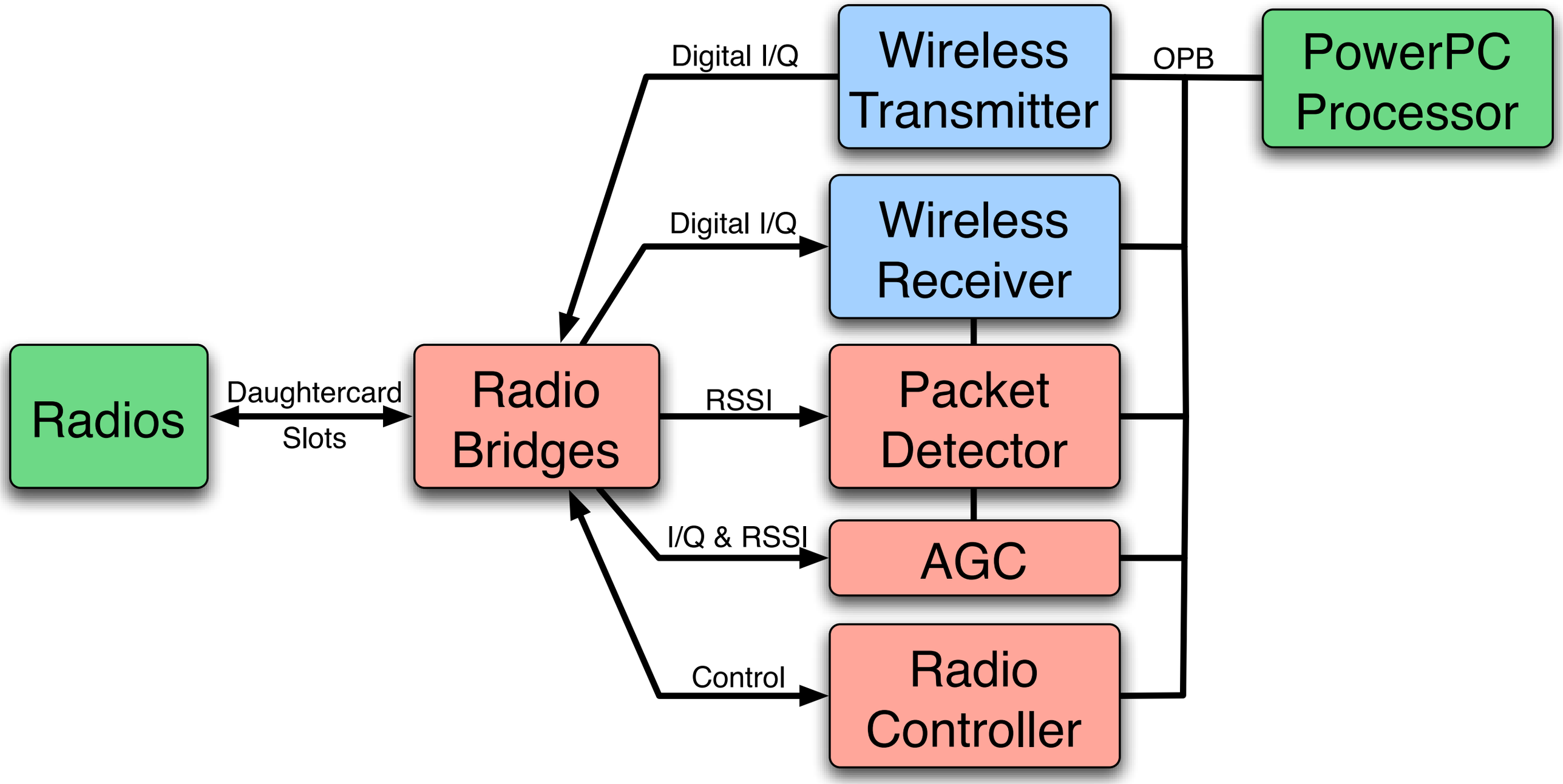


Register Bank
(controlled by SPI interface)

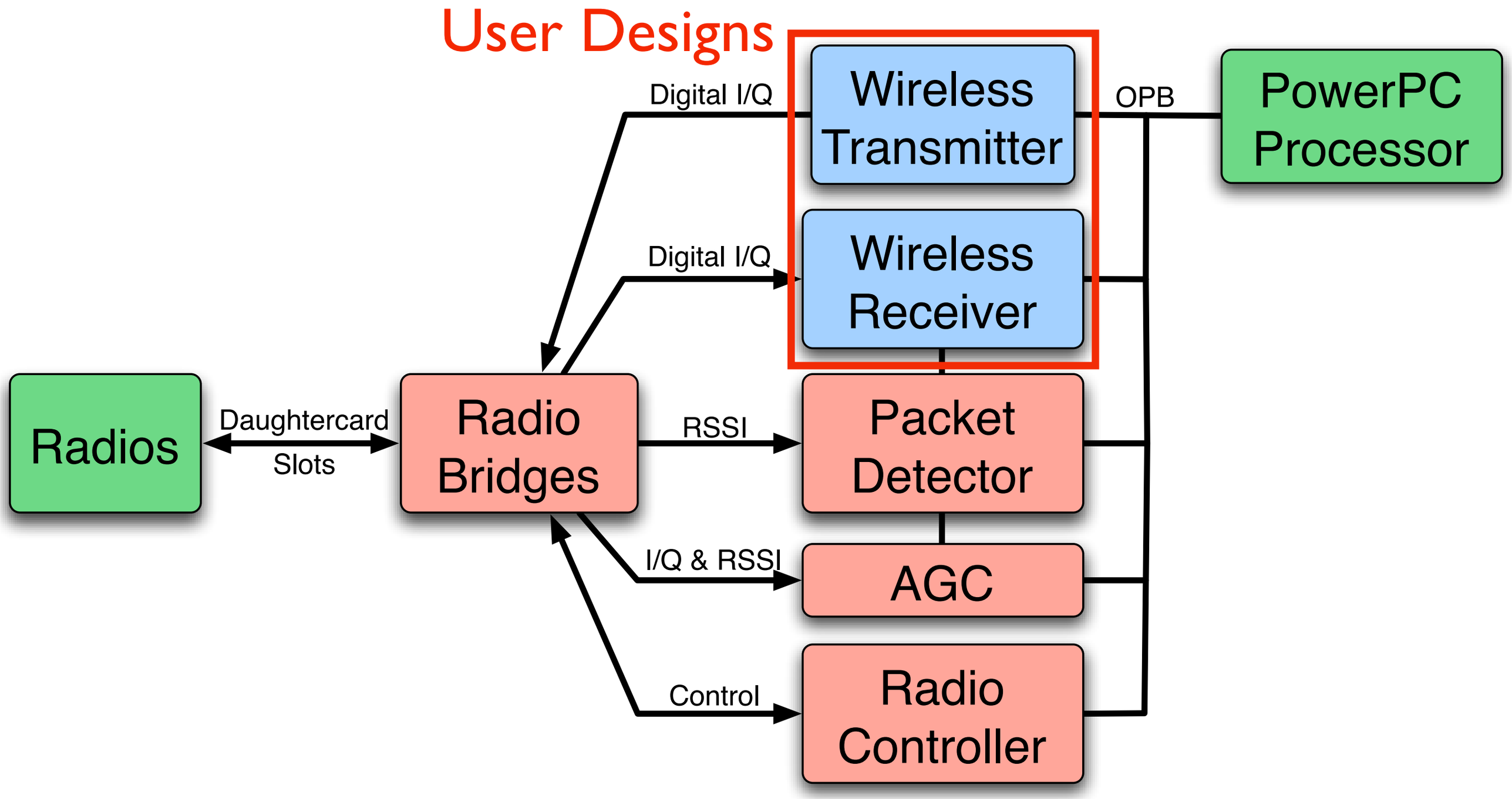
Physical Layer in Hardware



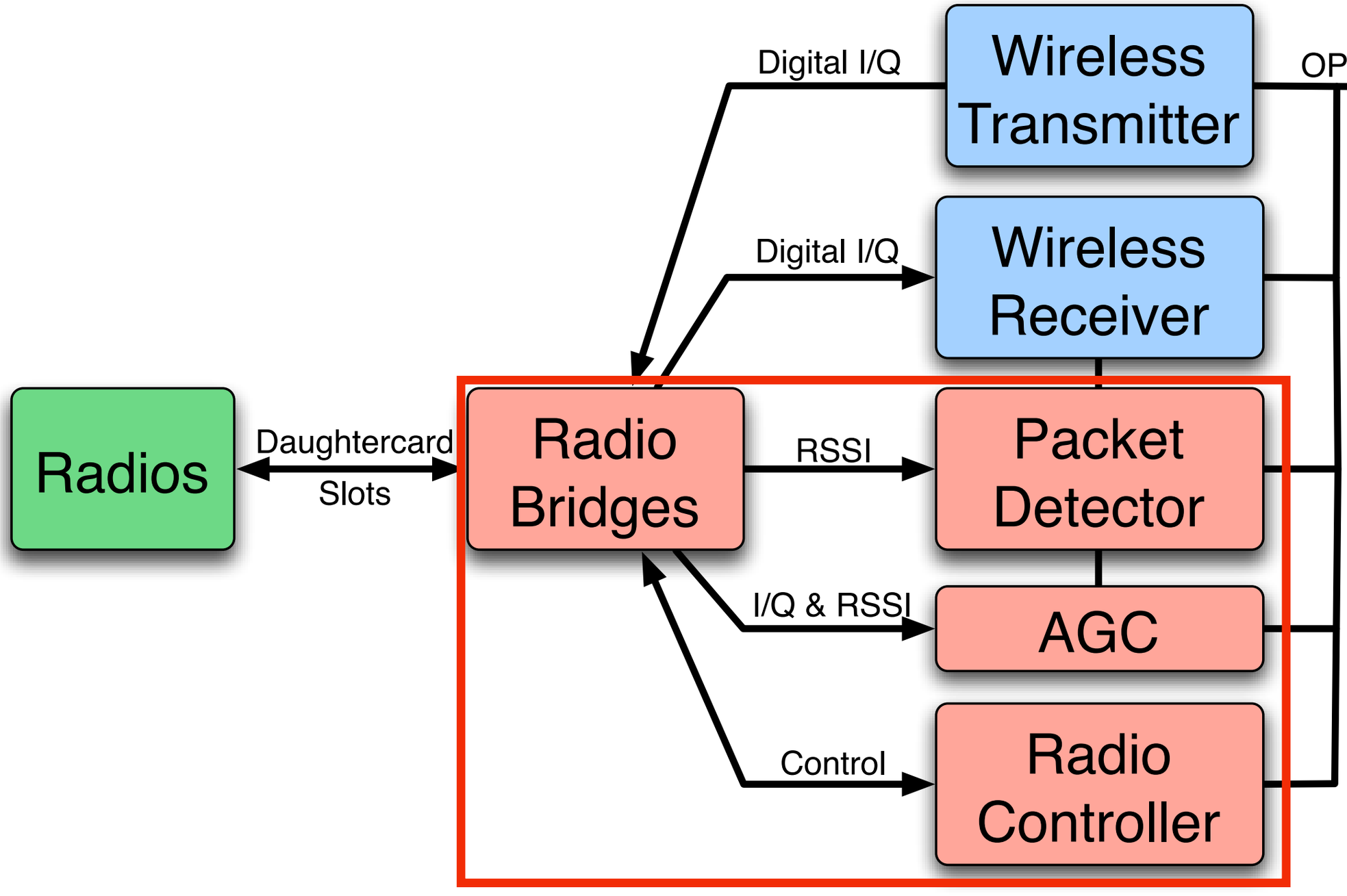
Physical Layer in Hardware



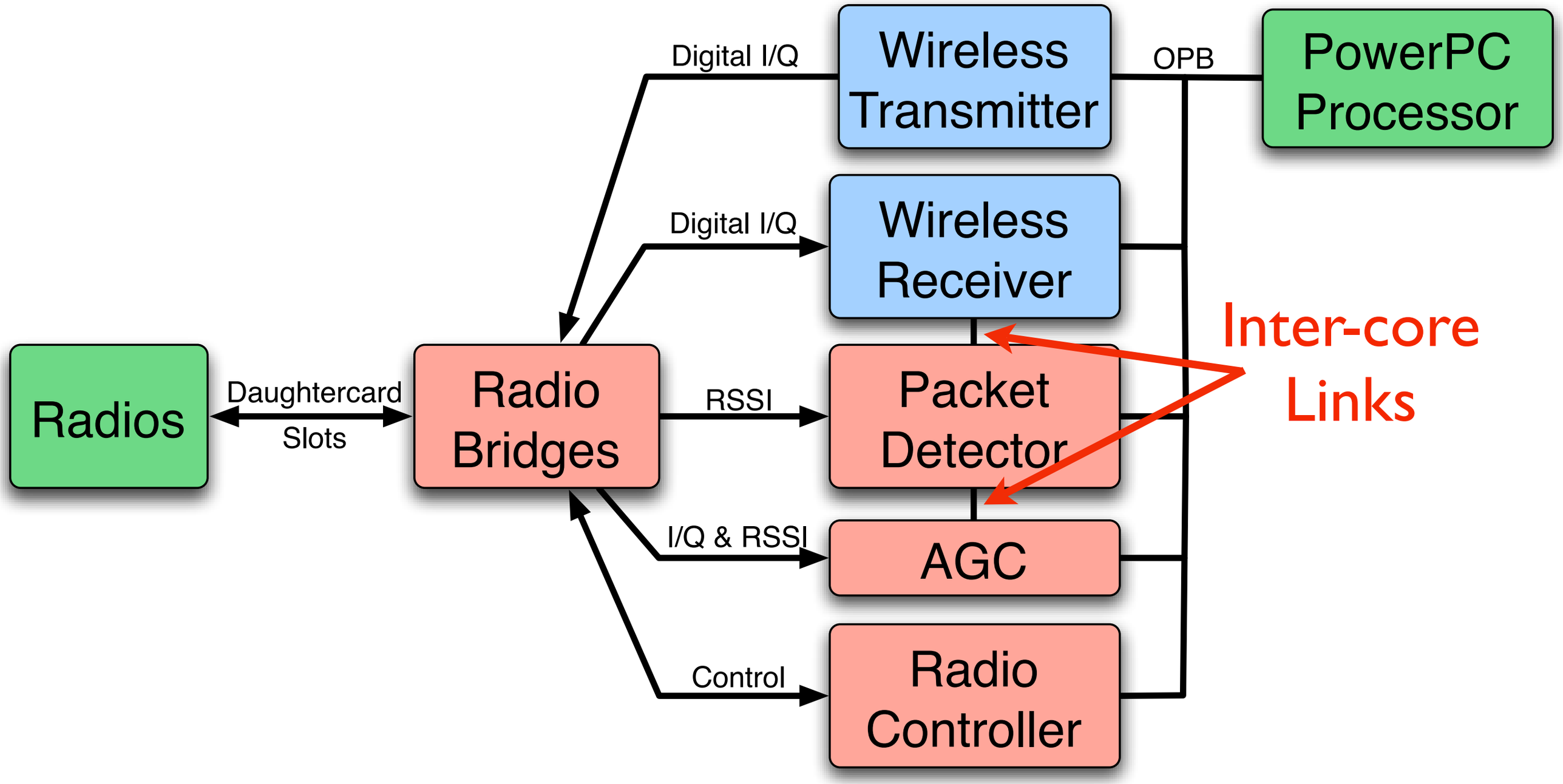
Physical Layer in Hardware



Physical Layer in Hardware



Physical Layer in Hardware



Packet Detection

- Triggers AGC & receiver models
- Detection based only on received energy
  - I/Q saturated and too corrupted
  - Gain adjusted *after* detection
- Detection confirmed/rejected by Rx PHY
  - Requires some data-aided detection
  - Correlates against every packet's preamble

RSSI

Incoming Packet

700
600
500
400
300
200
100
0

0
500
1000
1500
2000
2500
3000
3500

Automatic Gain Control

- Receiver has 90 dB gain range
  - RF gain of 0, 15 or 30 dB
  - Baseband gain of 0...60 dB
- Amplifiers start max gain with each packet
- AGC reduces gain in first 5 µs
  - RF gain set by RSSI
  - Baseband gain set by I/Q averages

Radio Controller

- Controller hardware
  - I/O registers & SPI controller
  - One core controls all 4 radios & DACs
- Controller software
  - Full C API for radio board control
  - All radio features controlled by C functions
  - Simple functions required
  - Advanced functions optional

Radio Controller API

```
WarpRadio_v1_Reset()

WarpRadio_v1_ TxEnable()
WarpRadio_v1_ SetCenterFreq2GHz()
WarpRadio_v1_ BaseBandTxGain()
WarpRadio_v1_ TxVGAGainControl()
WarpRadio_v1_ 24AmpEnable()

WarpRadio_RxEnable()
WarpRadio_RxLNAGainControl()
WarpRadio_RxVGAGainControl()
WarpRadio_RxLpfCornFreqCoarseAdj()
```

Radio Controller API

WarpRadio_v1_Reset()
WarpRadio_v1_TxEnable()
WarpRadio_v1_SetCenterFreq2GHz()
WarpRadio_v1_BaseBandTxGain()
WarpRadio_v1_TxVGAGainControl()
WarpRadio_v1_24AmpEnable()

WarpRadio_RxEnable()
WarpRadio_RxLNAGainControl()
WarpRadio_RxVGAGainControl()
WarpRadio_RxLpfCornFreqCoarseAdj()

Full API online:

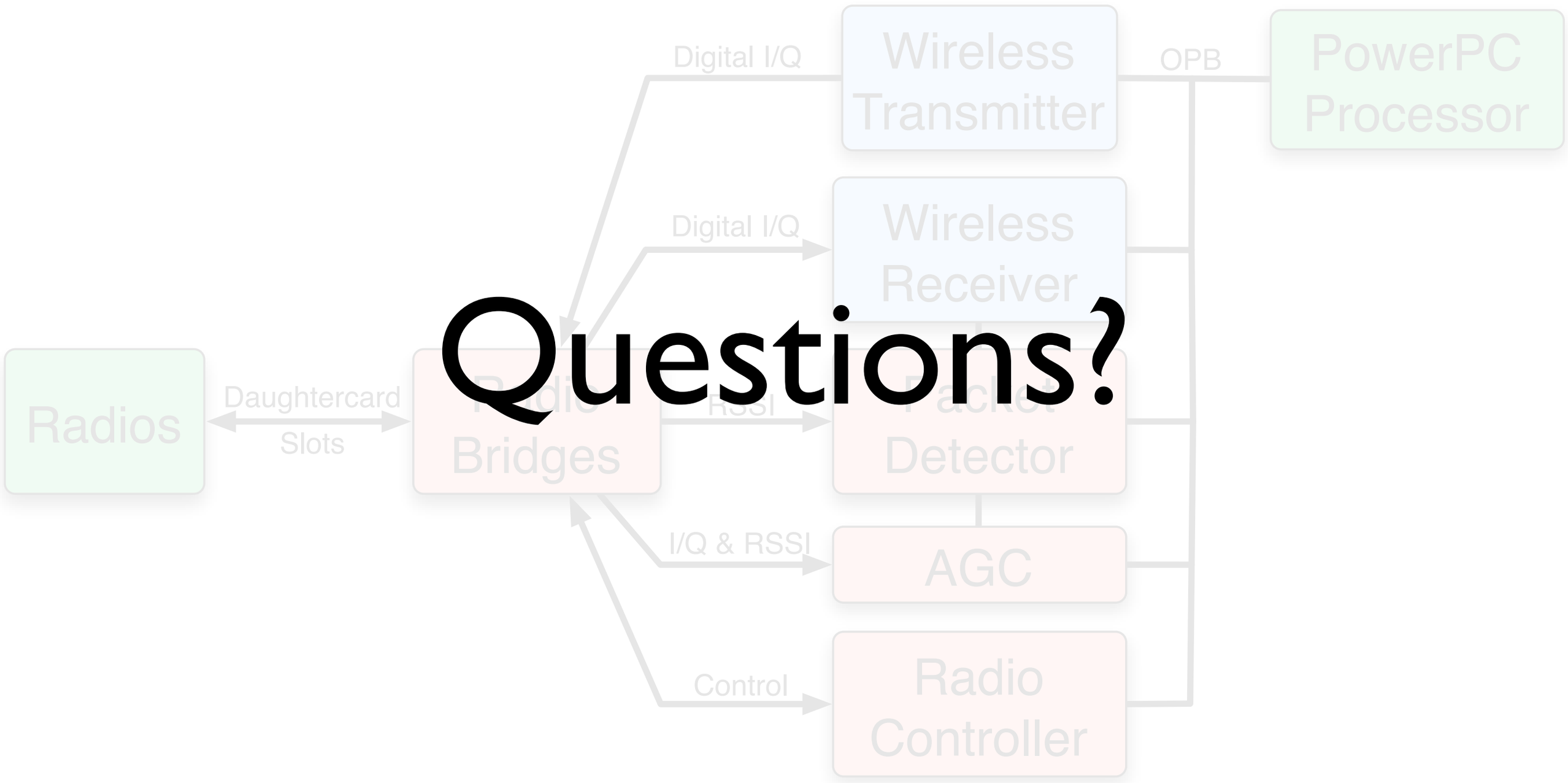
http://warp.rice.edu/WARP_API

Radio Bridge

- Ties user designs to radio hardware
  - Ports for user signals (ADC, DAC, gains)
  - Ports for radio controller I/O
- Users instantiate one bridge per radio board
- All constraints & most links are automatic
- Custom Verilog peripheral

PHY Design Review

- Build & verify PHY in FPGA design tool
  - System Generator is a good choice
  - Make sure everything works in simulation
- Generate simple Tx/Rx peripherals
  - “Cheating” is good at first
- Hook up your core in the EDK
  - Use our radio bridges & controller
- Generate the platform & test it in hardware


Digital I/Q
Wireless Transmitter
OPB
PowerPC Processor
Digital I/Q
Wireless Receiver
Radios
Daughtercard Slots
Radio Bridges
RSSI
Packet Detector
I/Q & RSSI
AGC
Control
Radio Controller

Questions?

Lab 2: Simple Transmitter

- Build a sinusoid generator in Sysgen
- Convert the model to an OPB peripheral
- Connect the Tx core to the radio bridge
- Test the model at RF