ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Software-Defined Radio with RF Systems on Module Workshop

TRAVIS COLLINS, PHD

ROBIN GETZ

ANALOG DEVICES

NOAM LEVINE

MATHWORKS

Agenda

	Day 1 – Tools and Physical Layer Algorithms	Day 2 – Hardware Implementation and MAC Layer
AM	• Introduction and demo • Workflow overview • **LAB**: IIO-Scope • Simulation, hardware evaluation with algorithms • **LAB:** MATLAB Streaming • Modem overview • **LAB**: Modem test harness • Discussion	• Day 1 review • Preparing for HDL and codegen • Deployment limitations and debugging • **LAB: Observable deployed designs** • Traditional HDL debugging • Discussion

PM	• Introduction to modeling with Simulink
	  • Simulation and testing
	  • Fixed-Point conversion and HDL capability
	• **LAB:** Fixed-Point conversion example
	• Discussion
		• MODEM demo on RF SOM Box
		• HSP vs Custom Reference designs
		  • Linux/HDL/BSP
		• **LAB:** TUN/TAP with OSC and debug registers
		• Summary and discussion
		  • Next steps and advanced topics

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

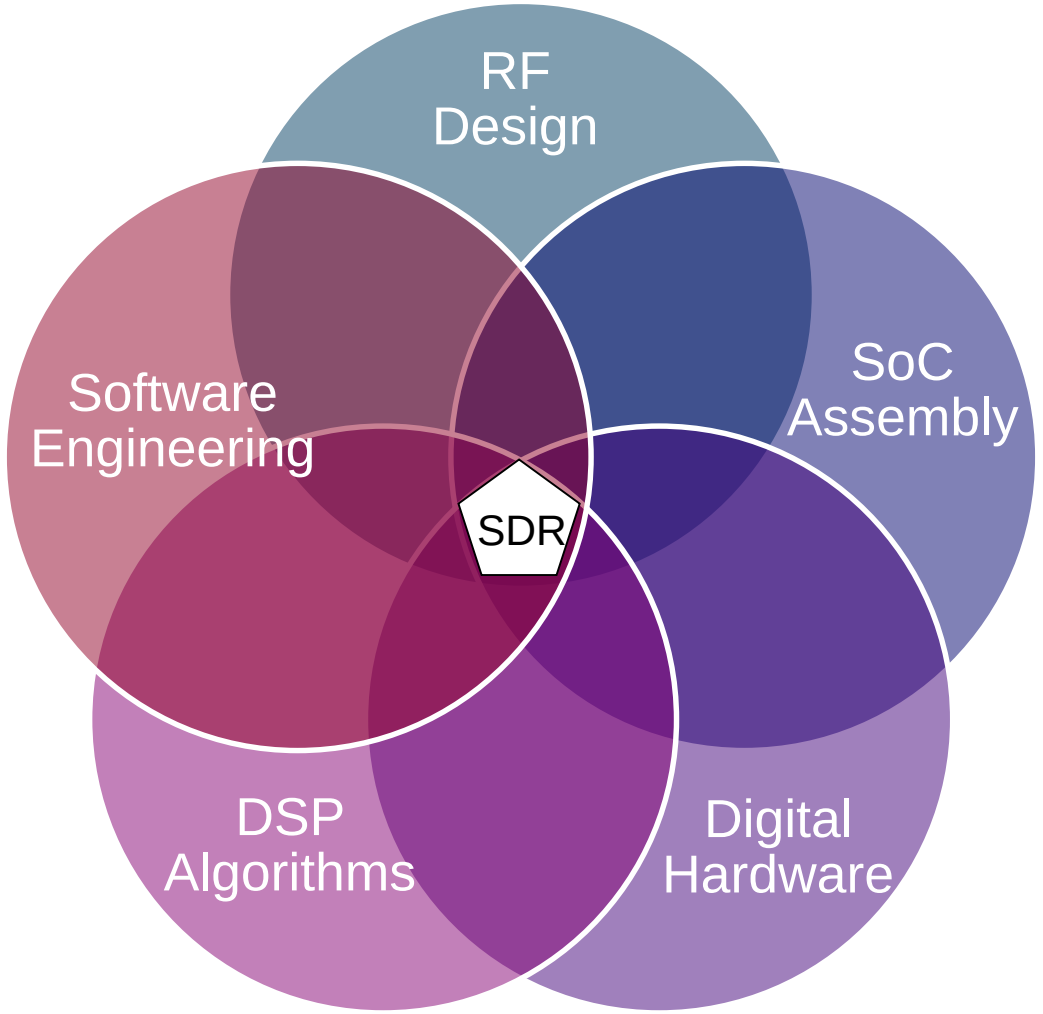
MathWorks®

Overview

ROBIN GETZ

What are the SDR problems ADI and MathWorks address?

- Software Defined Radio (SDR) is the unique combination
  - RF Design
  - SoC Assembly
  - Digital Hardware
  - DSP Algorithms
  - Software Engineering
- Few people are the experts on all aspects
- Academically – there is little overlap



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

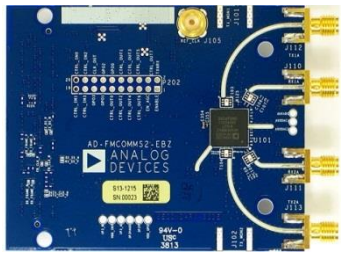
MathWorks®

ADI (and other) General Purpose SDR Boards which use the AD9361

Designed, manufactured, tested and sold by ADI

AD-FMCOMMS2

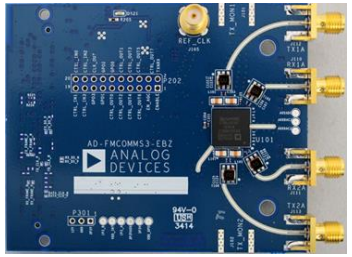
- AD9361 Integrated
- 2 x Rx, 2 x Tx
- *2.2 GHz – 2.6GHz tuning range*
- 200kHz - 56 MHz channel bandwidth
- Shipping Now!



Power, Transceiver

AD-FMCOMMS3

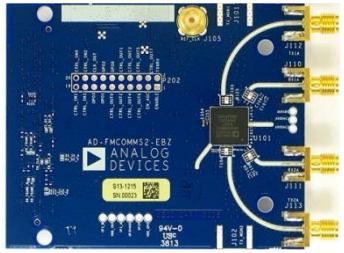
- AD9361 Integrated
- 2 x Rx, 2 x Tx
- *70 MHz – 6GHz tuning range*
- 200kHz - 56 MHz channel bandwidth
- Shipping Now!



Power, Transceiver

AD-FMCOMMS4

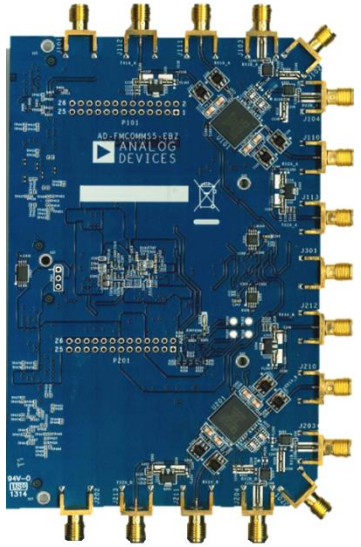
- *AD9364 Integrated*
- *1 x Rx, 1 x Tx*
- 70 MHz – 6GHz tuning range
- 200kHz - 56 MHz channel bandwidth
- Shipping Now



Power, Transceiver

AD-FMCOMMS5

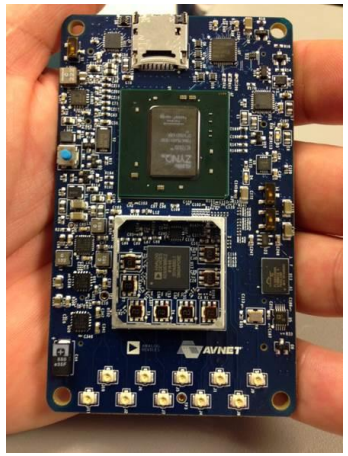
- *2 x AD9361 Integrated*
- *4 x Rx, 4 x Tx*
- *Synchronized RF*
- 70 MHz – 6GHz tuning range
- 200kHz - 56 MHz channel bandwidth
- Shipping Now!



Power, Transceiver, PLL, LNA

RF SOM

- *AD9361 Integrated 2 x Rx, 2 x Tx*
- 70 MHz – 6GHz tuning range
- 200kHz - 56 MHz channel bandwidth
- Shipping Now!
- Zynq 7035 + 1GB DDR + 32MB FLASH



ARRADIO

- AD9361 Integrated
- 2 x Rx, 2 x Tx
- *2.2 GHz – 2.6GHz tuning range*
- 200kHz - 56 MHz channel bandwidth
- Shipping Now!

Power, Transceiver Projects for : Altera SOCKIT

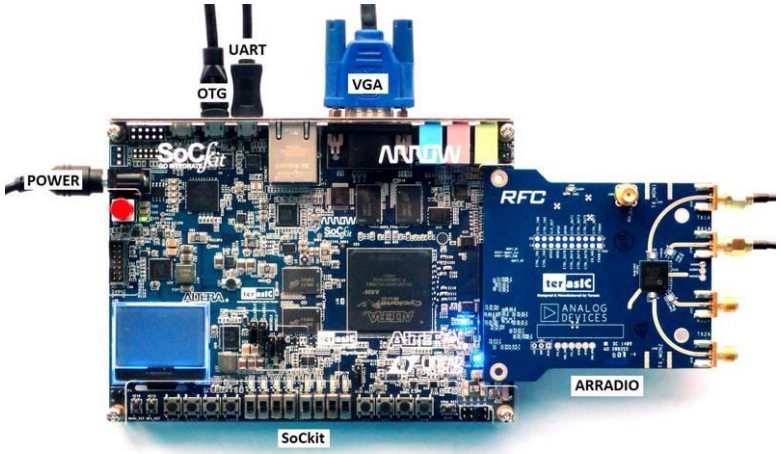
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Evaluating the AD9361

- Hardware
  - AD-FMCOMMS2-EBZ (AD9361)
    - Narrow RF Tuning Range
  - AD-FMCOMMS3-EBZ (AD9361)
    - Wide RF Tuning Range
  - AD-FMCOMMS4-EBZ (AD9364)
    - Narrow and Wide tuning range
  - AARADIO (AD9361)
    - Narrow RF Tuning Range
  - RF SOM (AD9361)
    - Wide RF Tuning Range
- Software
  - Device drivers
    - Linux and/or No-OS
  - FPGA HDL
  - IIO scope
    - Data visualization application
    - Graphical configuration application
- Not enough to make a data link

ZC706 + FMCOMMS2



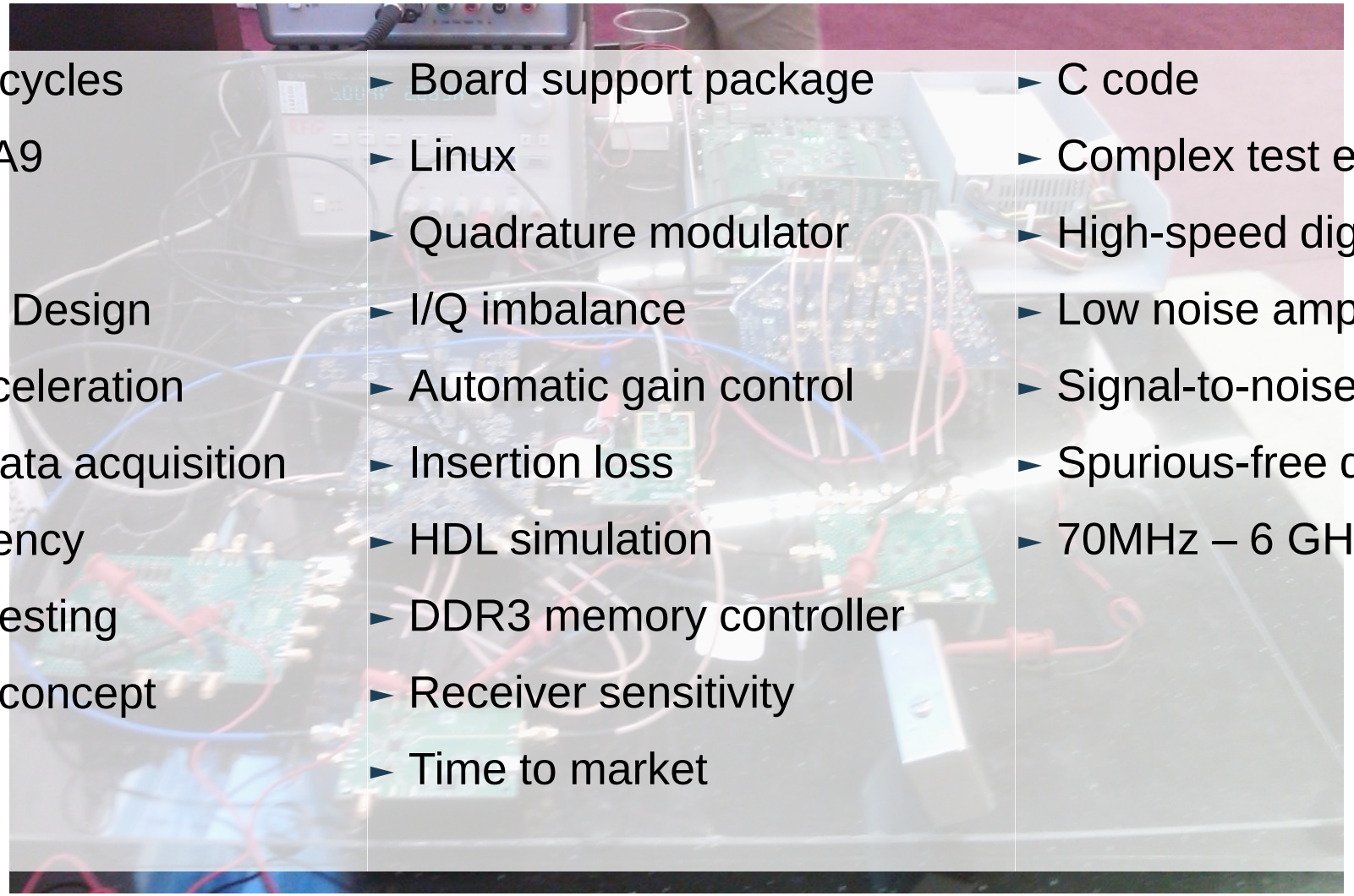
Arrow SoCKit + ARRADIO

RF SOM + Breakout Board

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Challenges of Software-Defined Radio System Design



- Short design cycles
- ARM Cortex A9
- RF expertise
- Model-Based Design
- Hardware acceleration
- High speed data acquisition
- Cache coherency
- Over-the-air testing
- Fast proof of concept
- FPGA design
- Board support package
- Linux
- Quadrature modulator
- I/Q imbalance
- Automatic gain control
- Insertion loss
- HDL simulation
- DDR3 memory controller
- Receiver sensitivity
- Time to market
- C code
- Complex test equipment
- High-speed digital signal processing
- Low noise amplifier
- Signal-to-noise ratio
- Spurious-free dynamic range
- 70MHz – 6 GHz

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Historical Modem Design Flows

Modem Model

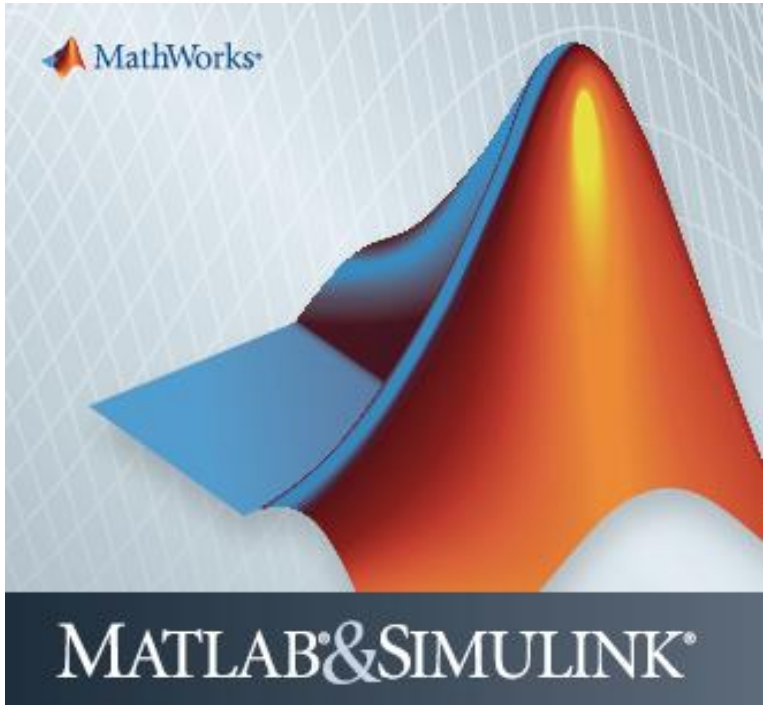
Create MATLAB Floating Point Reference Design → Create Python Implementation For Data Streaming → Create C/C++ Fixed-Point Version → Create Verilog/VHDL Version

Hardware

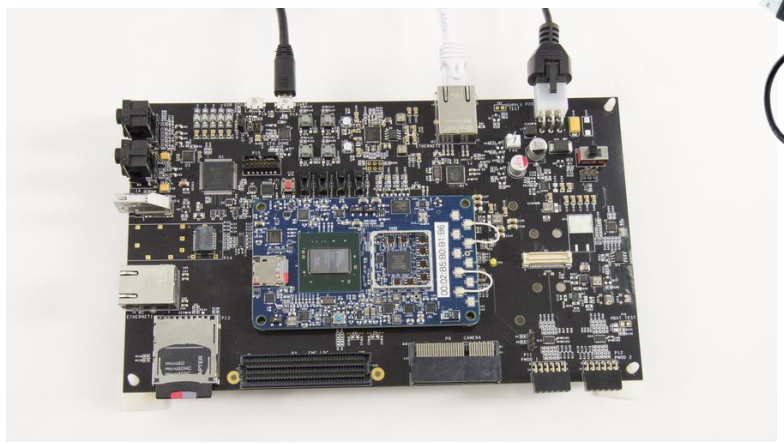
Start with Evaluation Board/Hardware → Build Custom Demo/Eval Board To Test Functionality → Build Final Production Board For Deployment

MathWorks/ADI Workflow Advantage

- Single software environment



- Appropriate hardware for each design phase

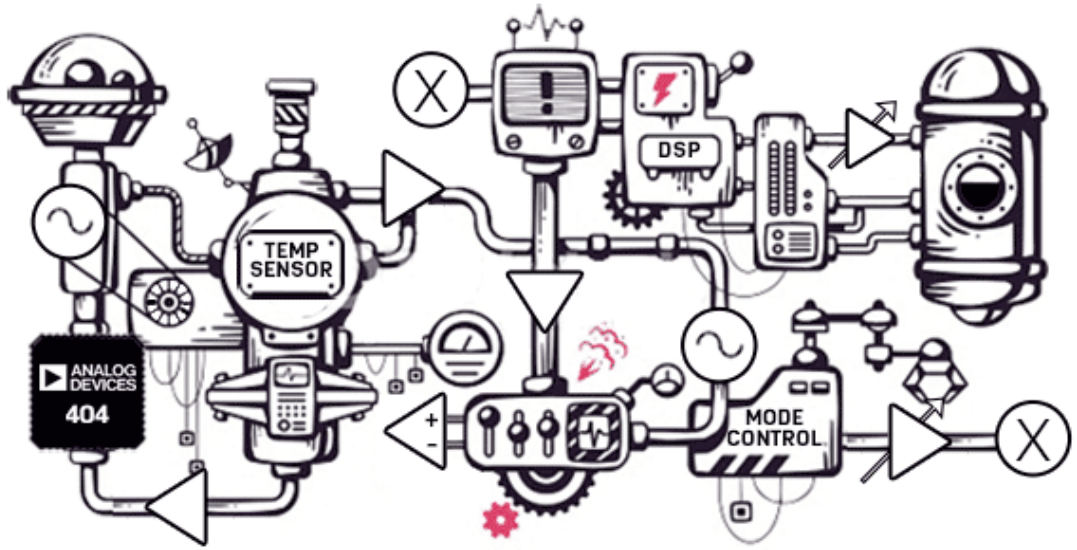


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Things we will discuss

- Workflow Introduction
- Example Design Introduction
  - MATLAB reference design
- Model-Based Design
- Fixed-Point Conversion
- HDL Deployment and Debugging
- Production Deployment Strategies

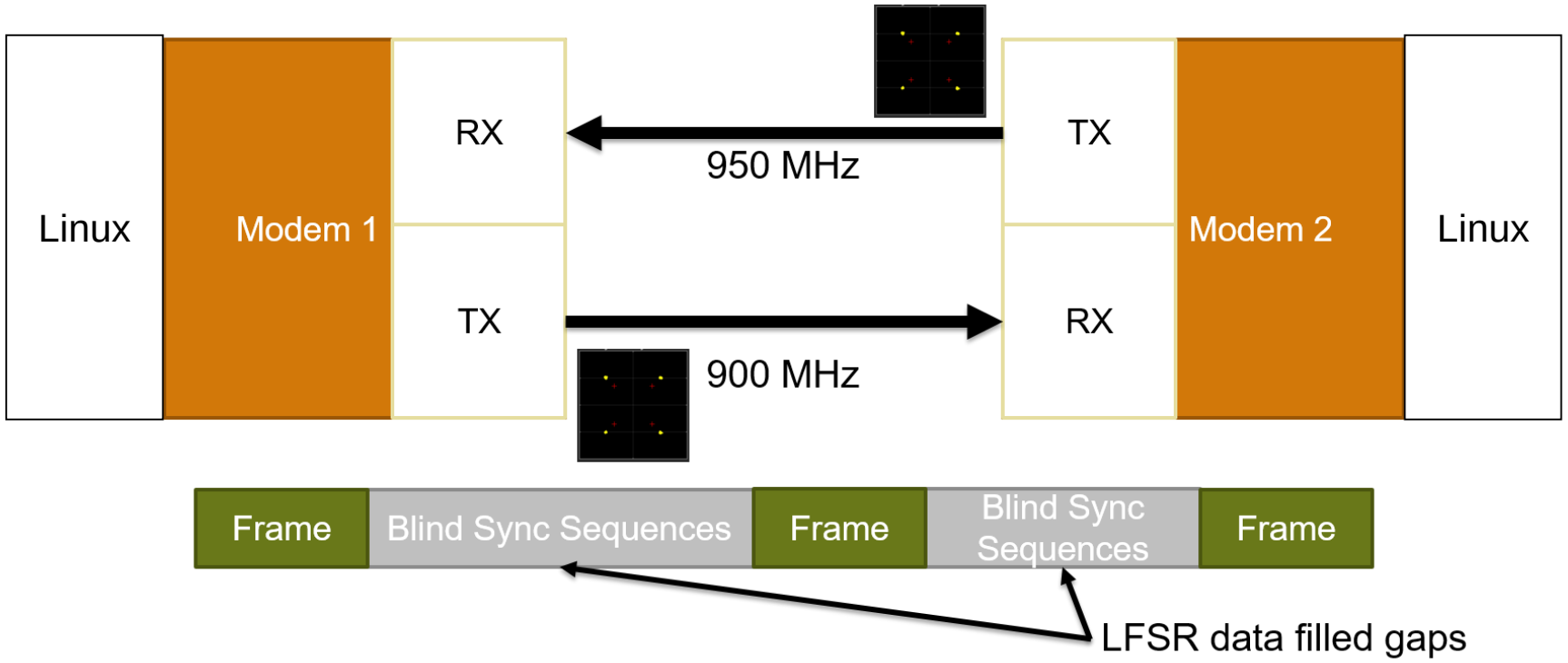


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Teaching the Workflow by Example: Modem Demo

- Modem functional details
  - QPSK single carrier system
  - FDD MAC layer
  - Point-To-Point link
- Software pieces
  - MATLAB reference design
  - Simulink float and fixed models
  - Testing harness integration with hardware support



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Demo

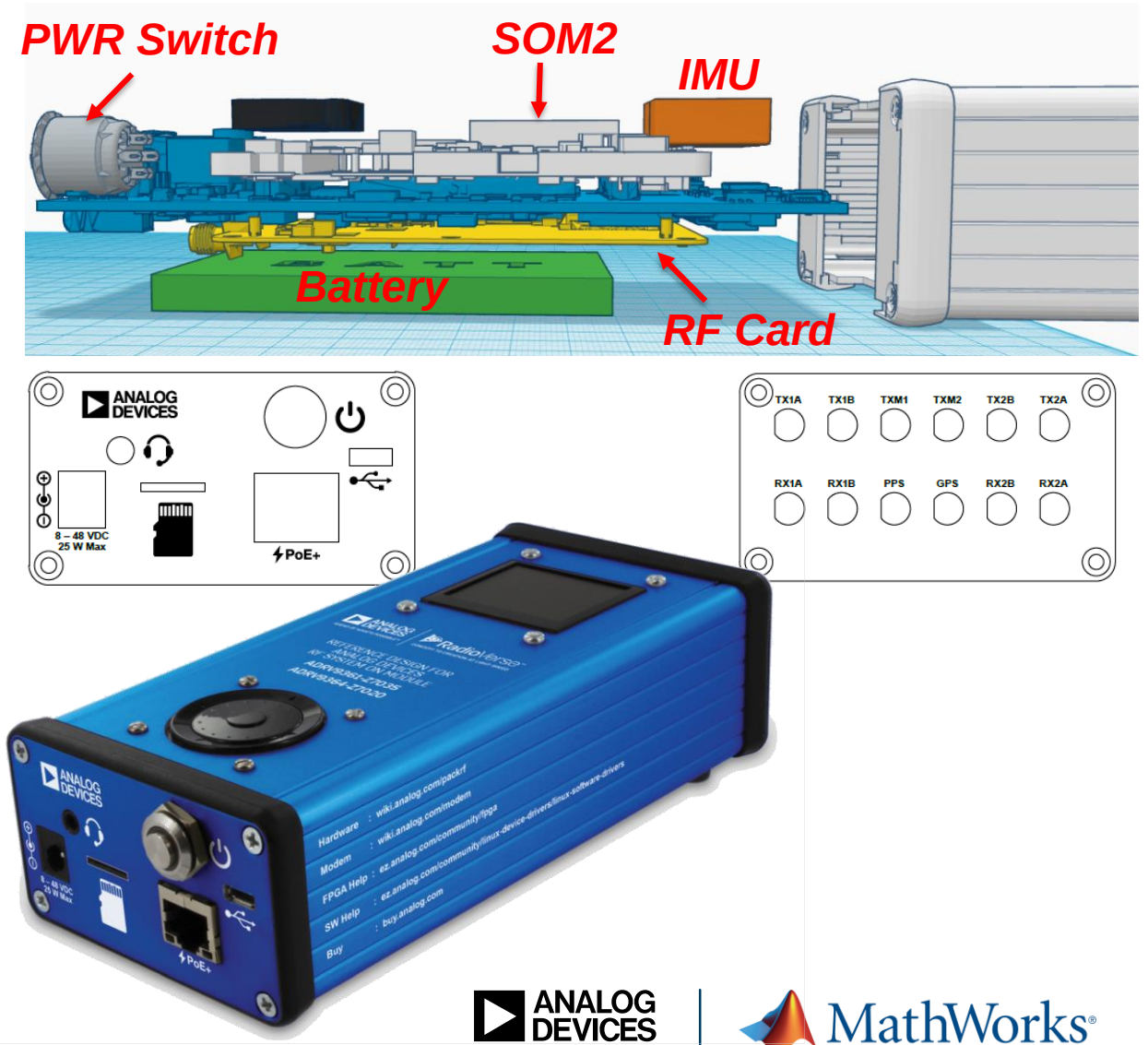
VIDEO LINK USING QPSK MODEM BUILT IN MATLAB AND SIMULINK

Demo you saw

- Data link between 2 radios, sending video
  - FDD
  - 5MHz (20 MSPS)
  - H.264 Video
    - ffmpeg in software on ARM
- Uses ADI RF SOM
  - Based on AD9361

PackRF Details

- Example design which shows how to design RF SOM into a custom carrier
- Custom Carrier includes:
  - OLED
  - Nav Switch
  - Power Button
    - Wake on RTC
  - Power over Ethernet (PoE+)
  - Automotive DC-DC converter
    - 8 – 48V DC input
  - Battery Management
  - Hot Power swap
  - Inertial Measurement Unit
    - Six Degrees of Freedom
  - GPS Chipset
    - 1 PPS in and out
  - Audio headset (stereo headphones, mic and button control)



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

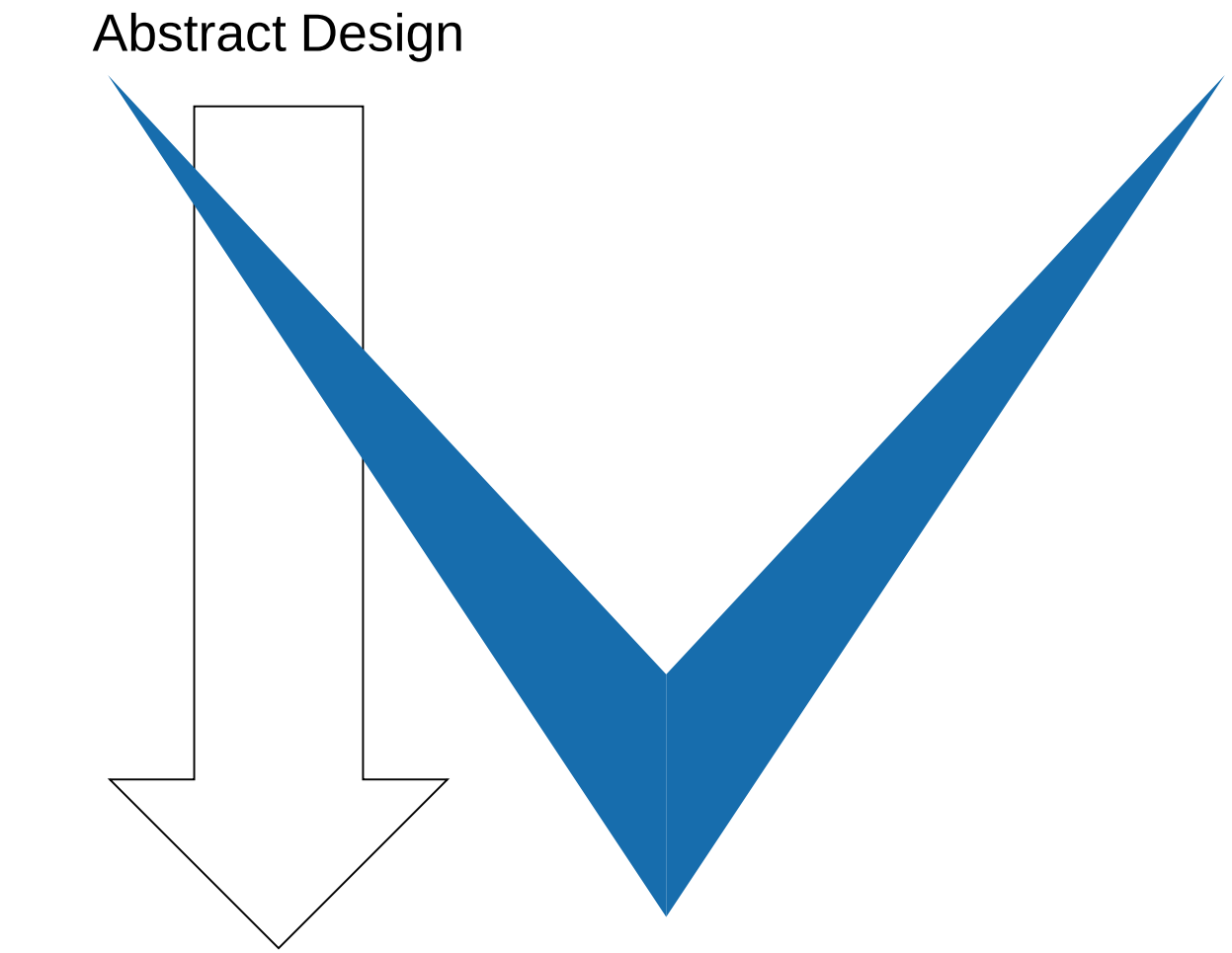
Tools, Workflow, and System Overview

NOAM LEVINE

How did we create modem?
Model-Based Design

- Modeling and simulation of the RF signal chain
- Development, modeling, and simulation of communications algorithms
  - MATLAB and Simulink
- Testing and verification of algorithms with real-world data
  - Streaming from RF hardware
- Deployment of communications system to hardware for prototyping and production
  - Streaming vs. frame-based implementations
  - Fixed-point implementation
  - Code generation and targeting

Workflow description

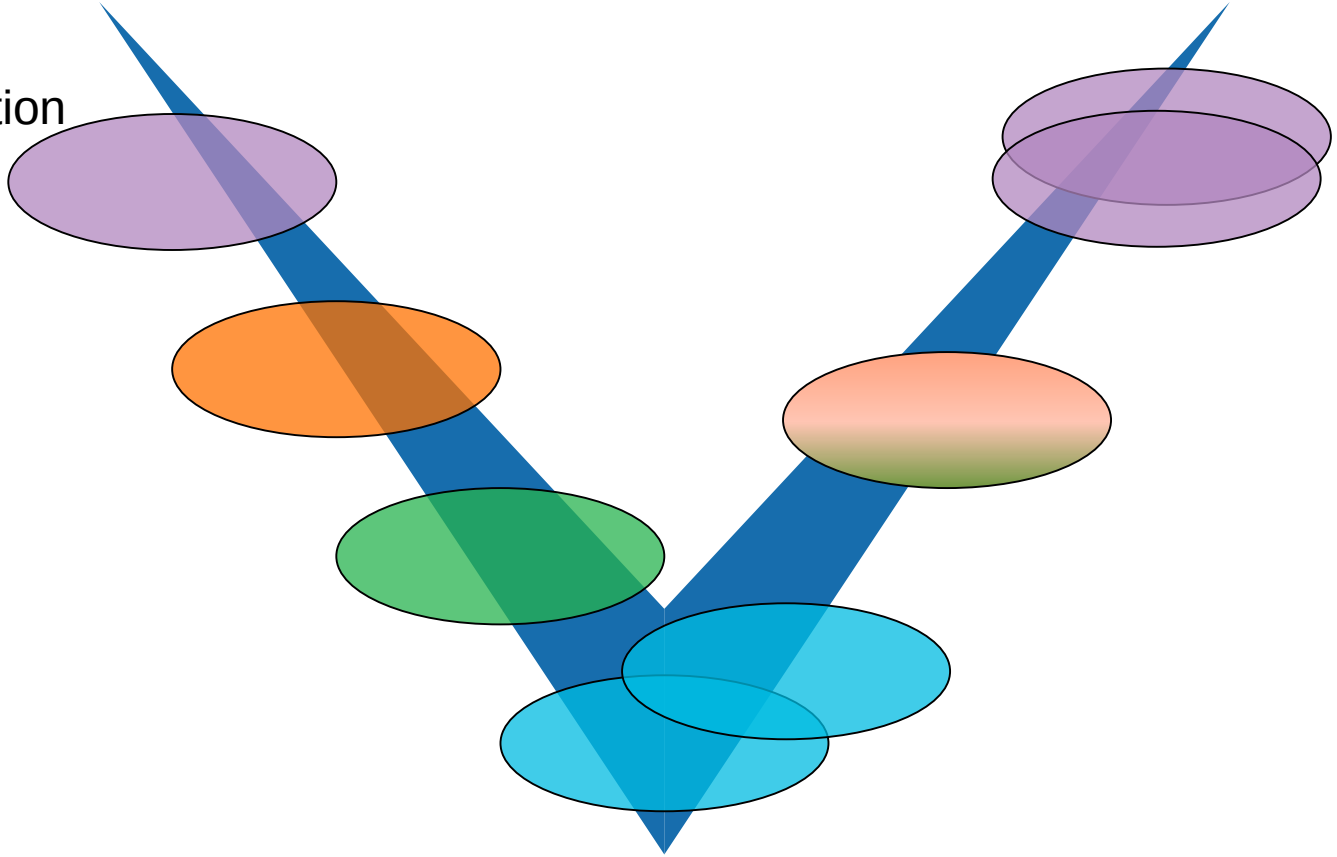


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Workflow description

- Four phases of design
  - System Architecture / Requirements Specification
  - Requirements Verification
  - Detailed Design
  - Deployment
- Five hardware verification methods
  - Component Evaluation
  - System-level Simulation
  - Radio I/O (streaming data)
  - Prototype Deployment
  - Production Deployment

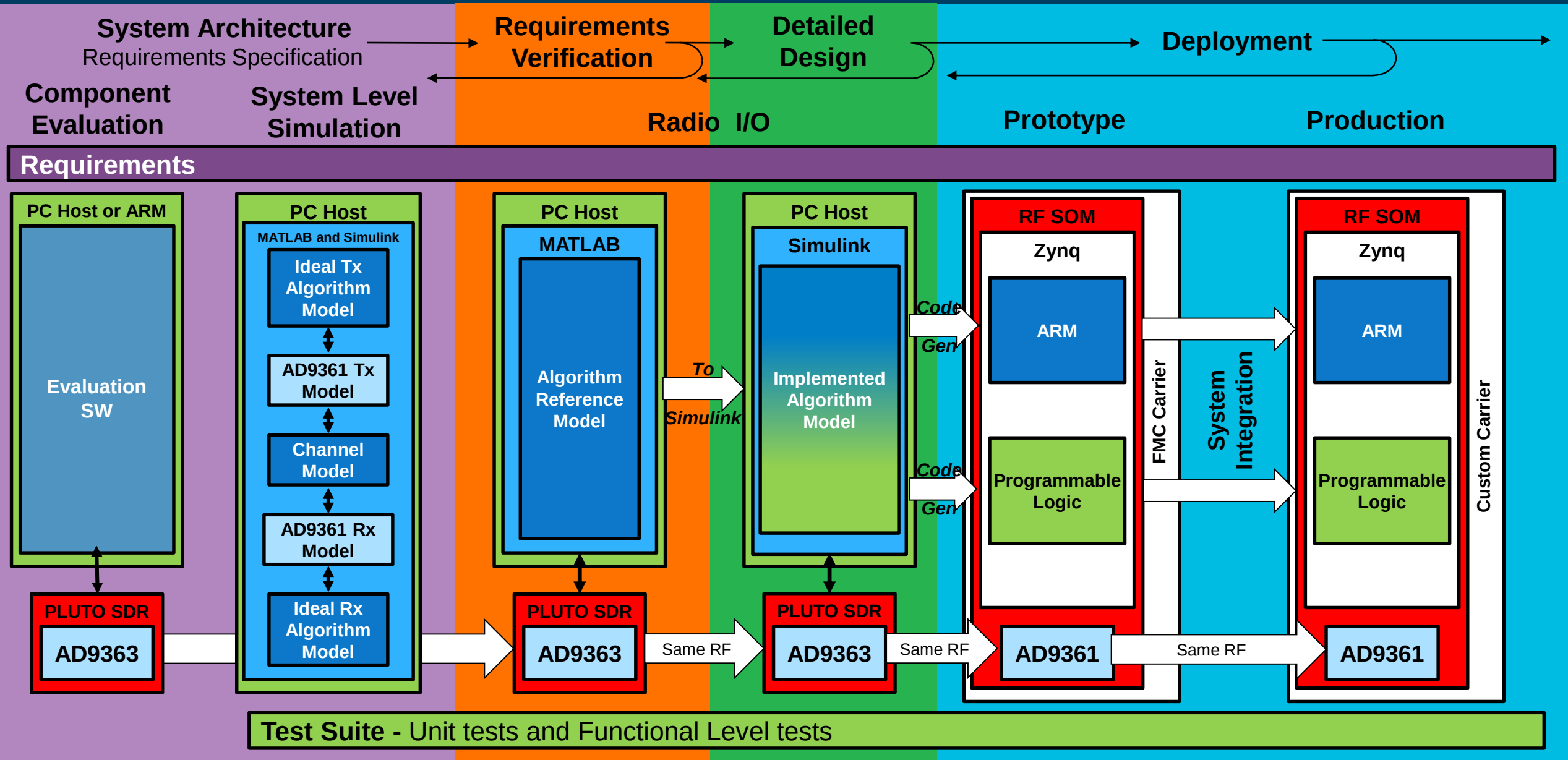


- **One Golden Reference, One Set of Requirements, One Test Infrastructure**

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Model-Based Design for SDR



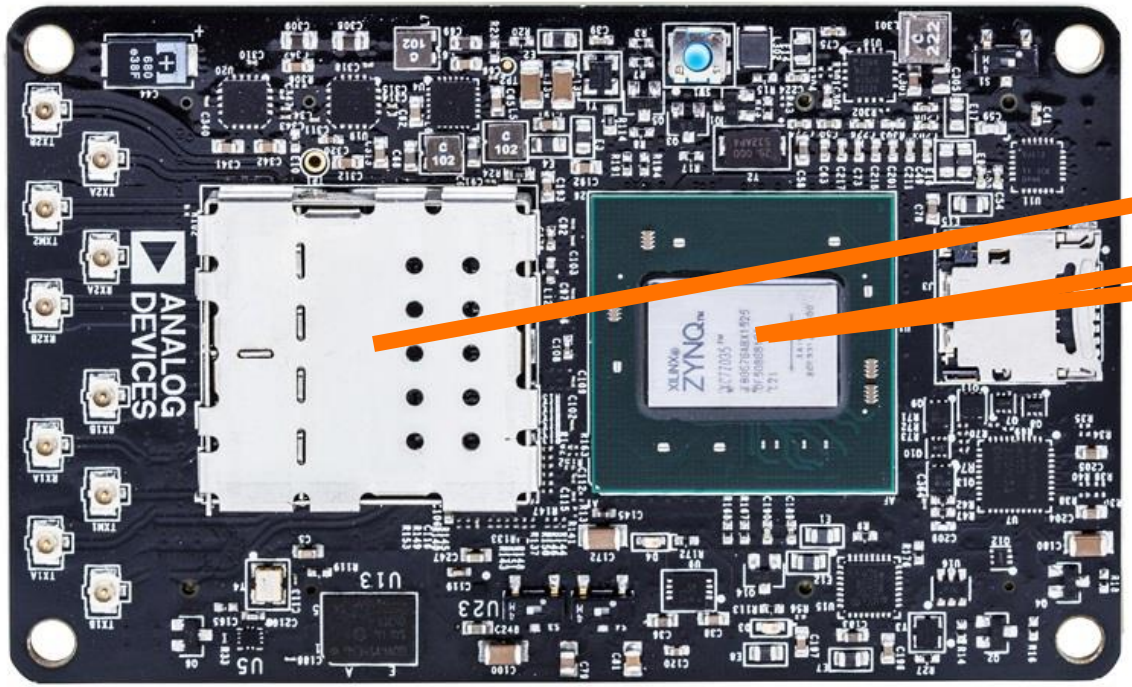
AD936X Transceiver Family

- **AD936X** is the SDR standard for high performance agile transceivers
- **RF-SOM** helps streamline system integration and development
- **PackRF** is a complete deployable system example

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

System-On-Module (SOM)



Hardware:

Transceiver

AD9361

HDL Reference Designs:

Xilinx
Intel

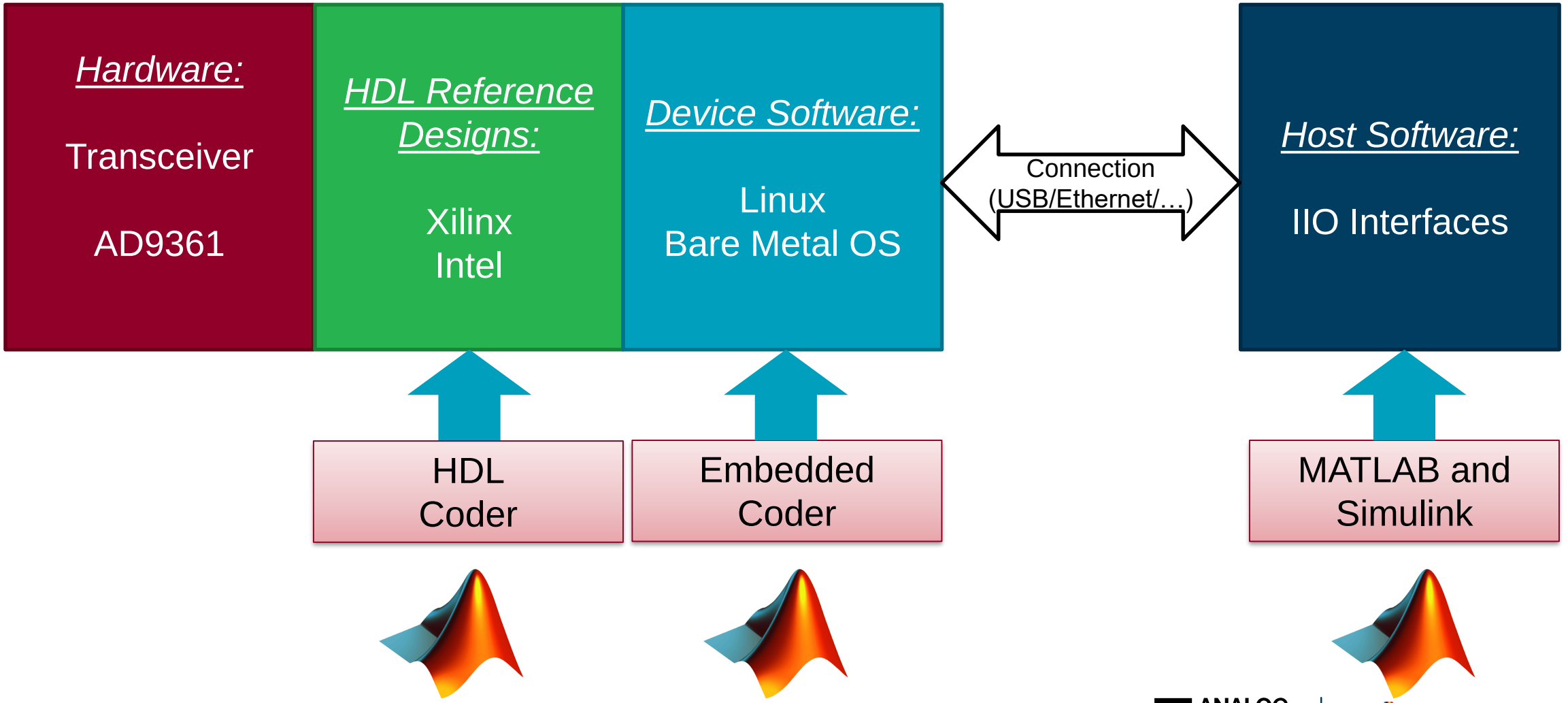
Device Software:

Linux
Bare Metal OS

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

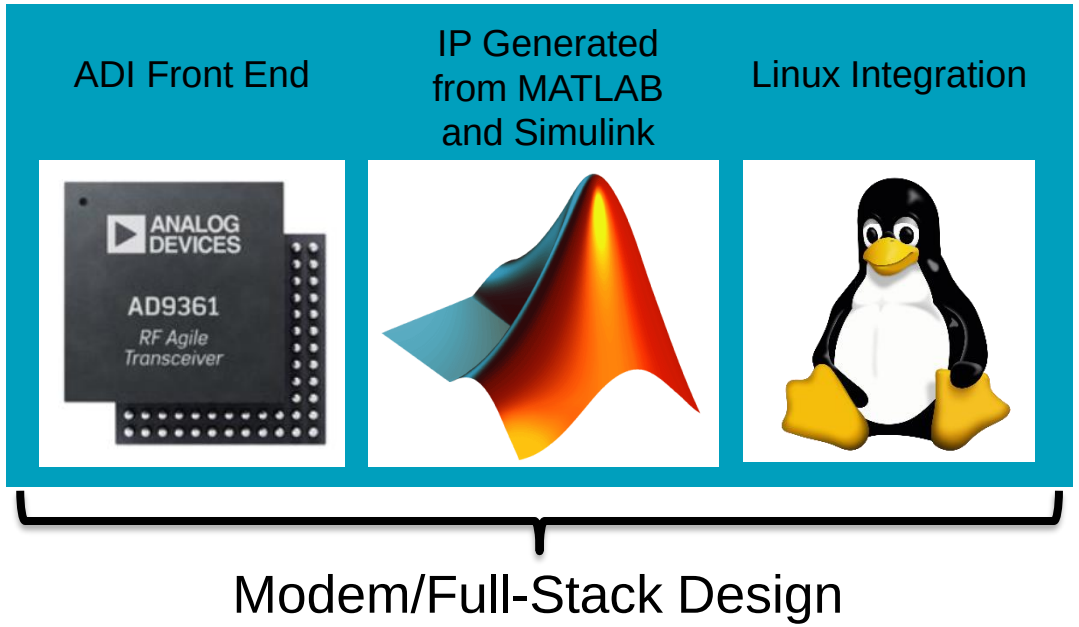
Hardware and Software Together: The Big Picture

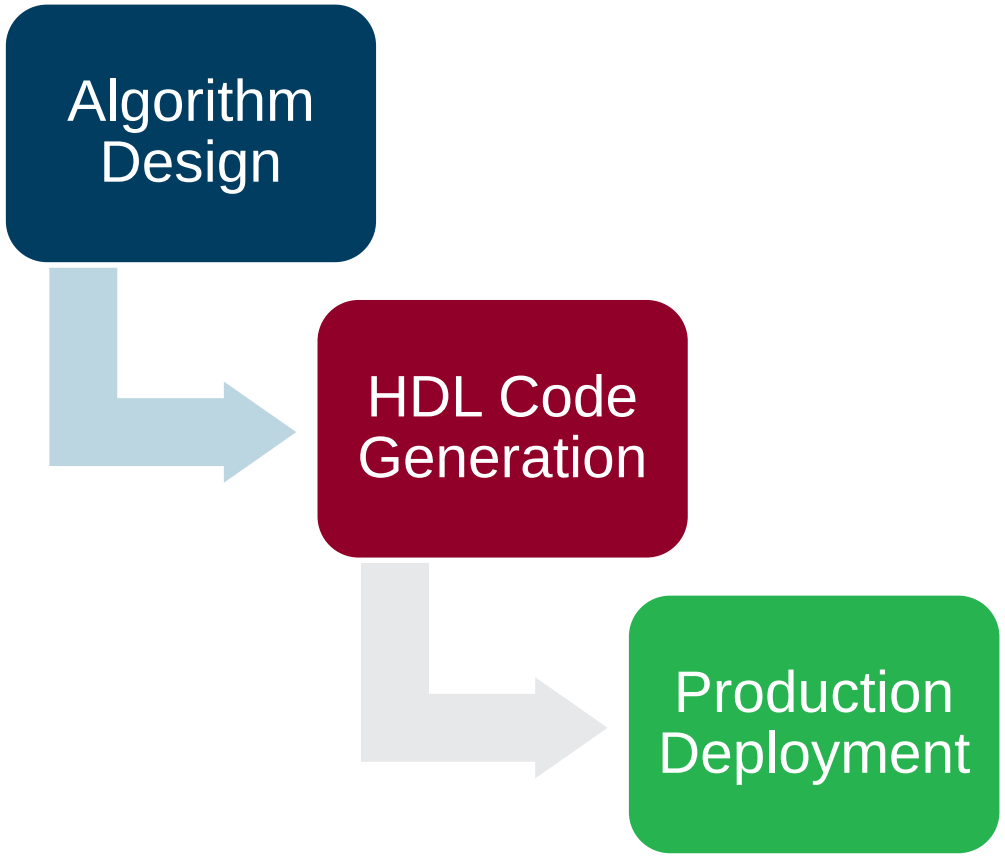


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

QPSK Modem Demonstration



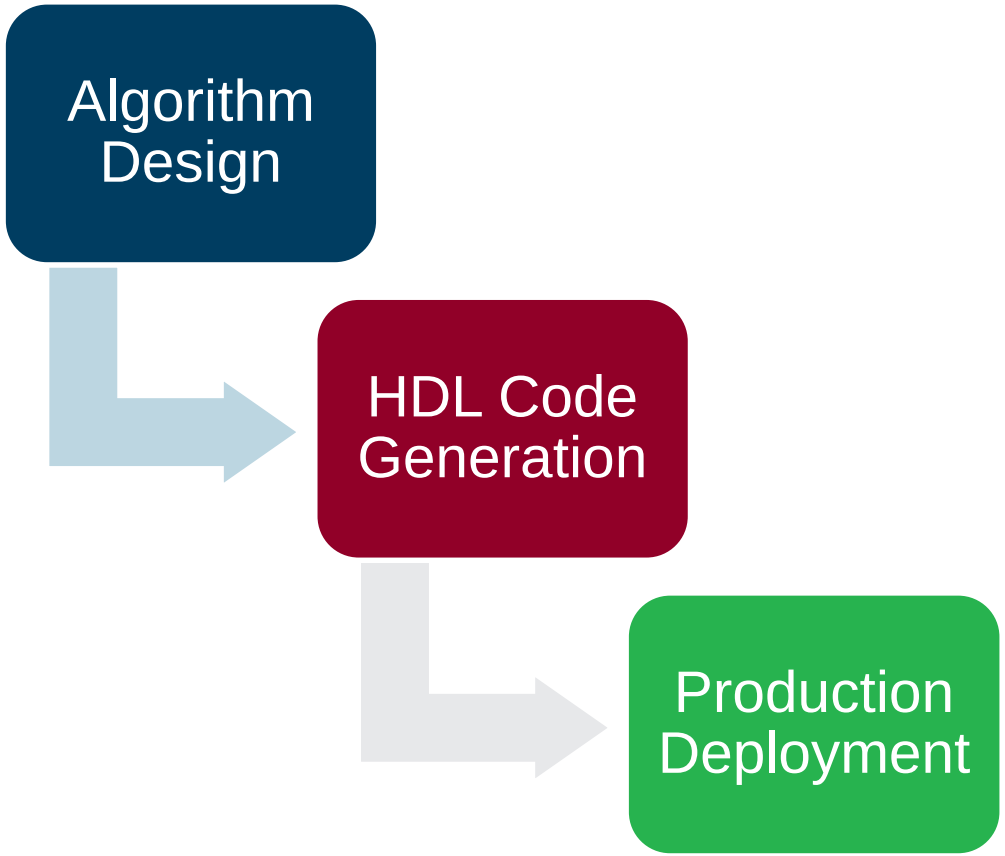


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

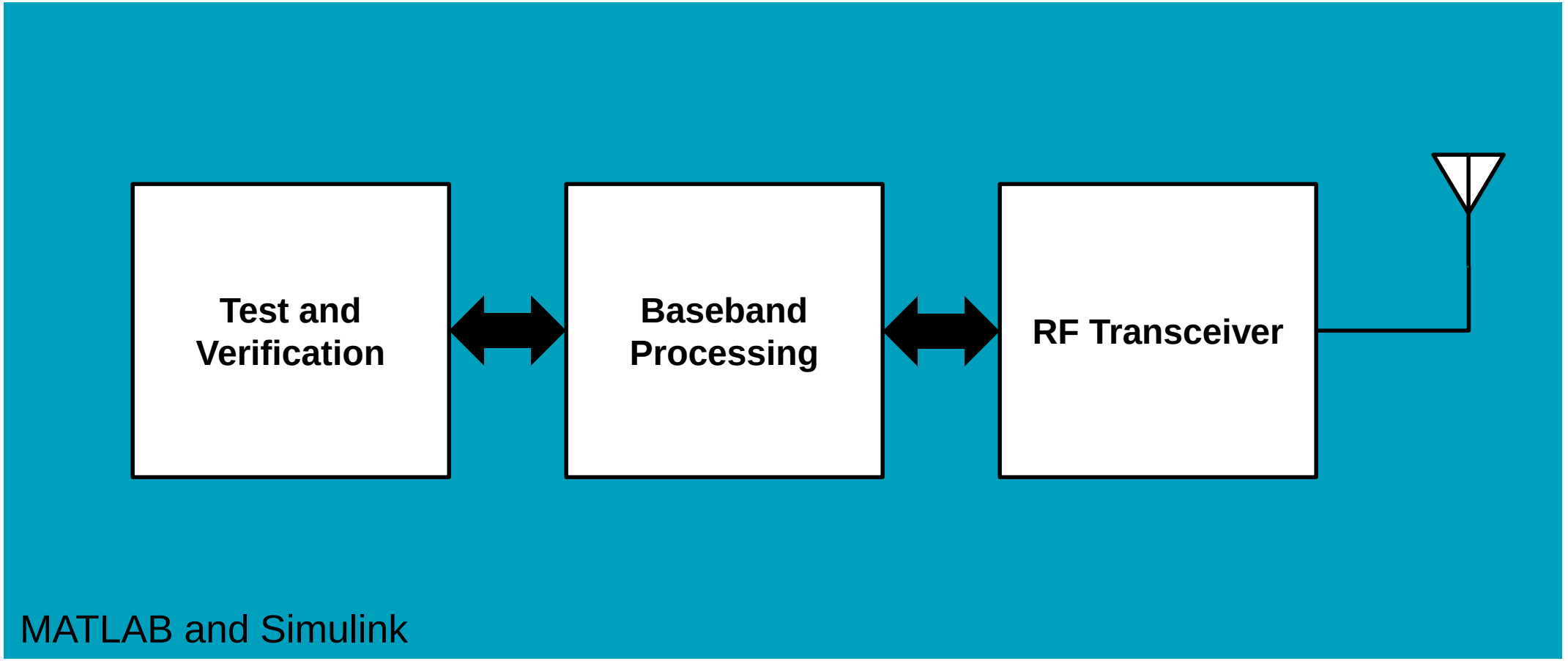
MathWorks®

What the workflow consists of

- Modeling and simulation of the RF signal chain
- Development, modeling and simulation of communications algorithms
  - MATLAB and Simulink
- Testing and verification of algorithms with real-world data
  - Streaming from RF hardware
- Deployment of communications system to hardware for prototyping and production
  - Streaming vs. frame-based
  - Fixed-point implementation
  - Code generation and Targeting



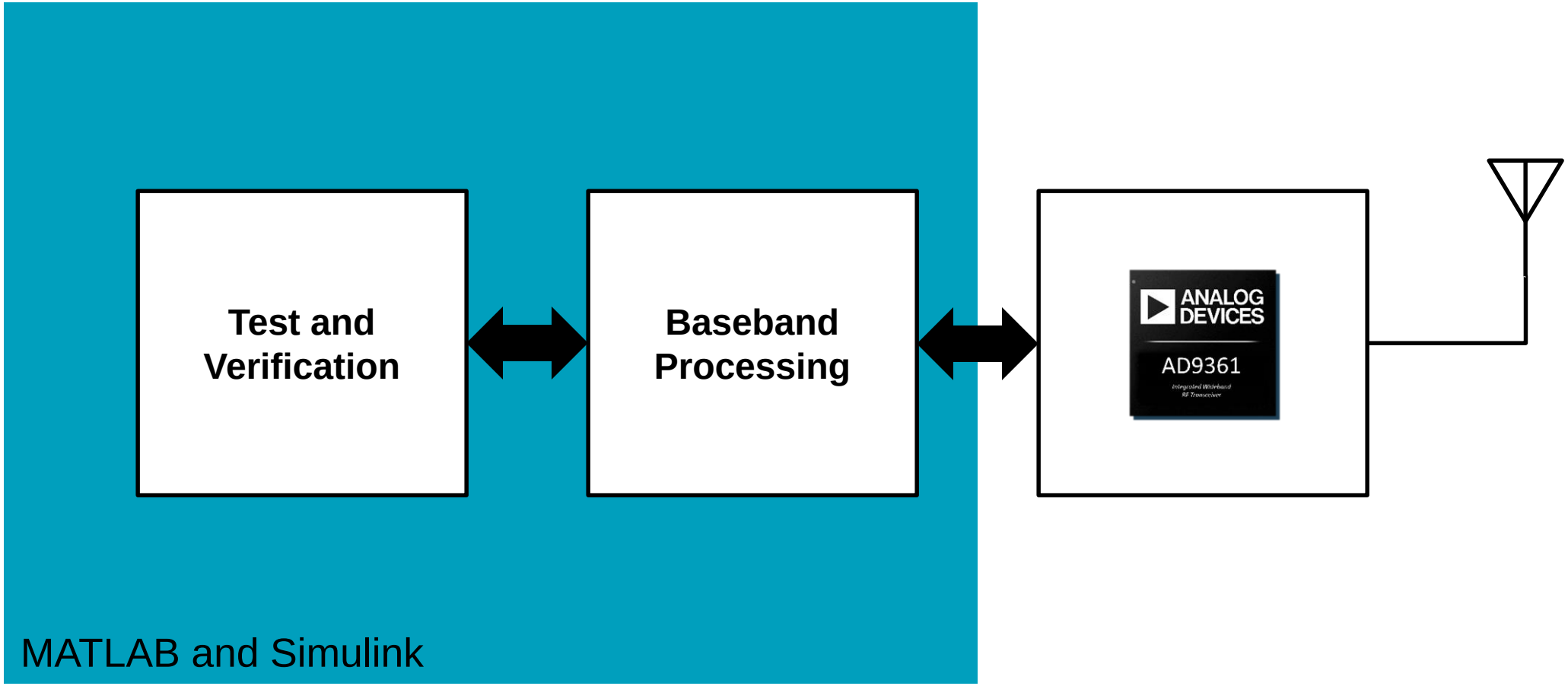
Elements of a Software-Defined Radio System
Modeling and simulation of the entire signal chain



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

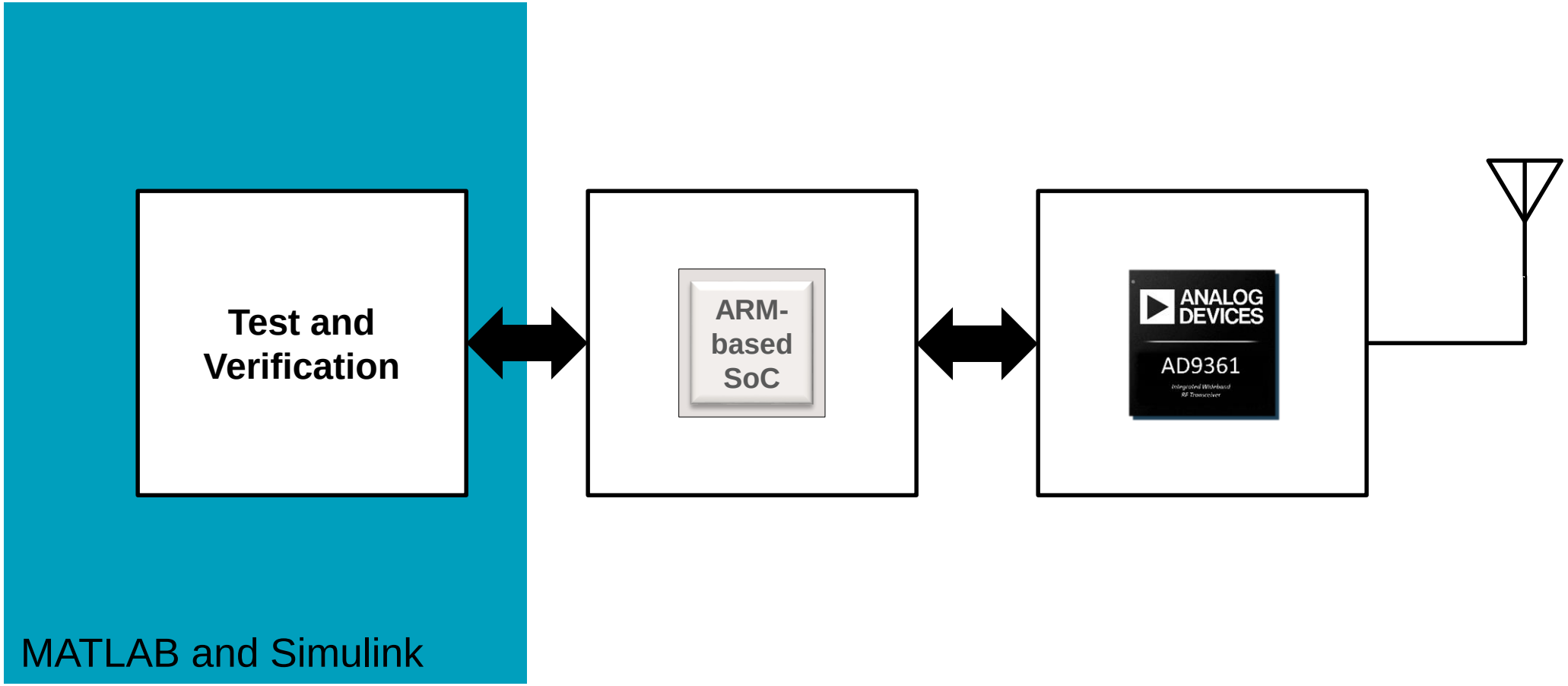
Elements of a Software-Defined Radio System
Algorithm simulation with streaming RF data



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

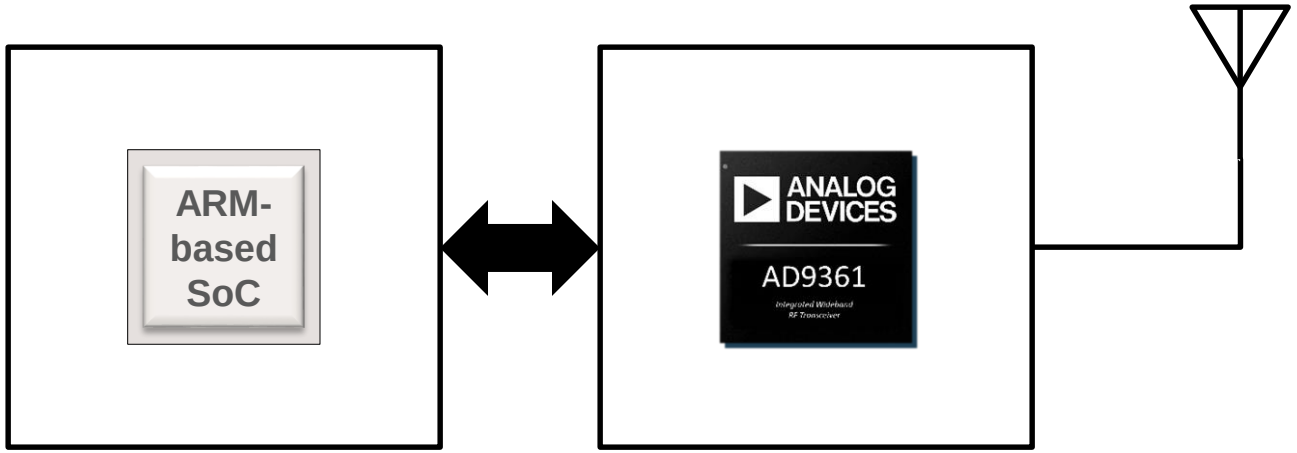
Elements of a Software-Defined Radio System
Prototype deployment with real-time data logging and parameter tuning



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

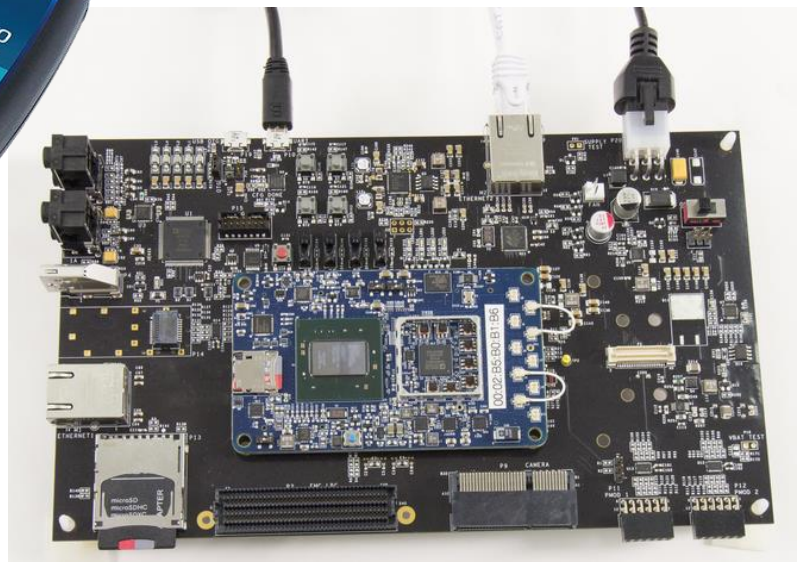
MathWorks®

Elements of a Software-Defined Radio System
Stand-alone operation



Hardware Platforms to be used in this Workshop

- Streaming Platform: ADALM-PLUTO
  - Based on AD9363 RF Transceiver
  - RF coverage from 325MHz to 3.8GHz
  - Single-channel receive and transmit
- Prototyping Platform: ADRV9361 RF SOM
  - Based on AD9361 RF Transceiver
  - RF coverage from 70MHz to 6GHz
  - Dual-channel receive and transmit
  - Fully-verified, production-ready SOM
- Custom Hardware: “PackRF”
  - Based on ADRV9361 RF SOM with custom carrier board
  - Portable radio with video I/O capabilities
  - Custom reference design example

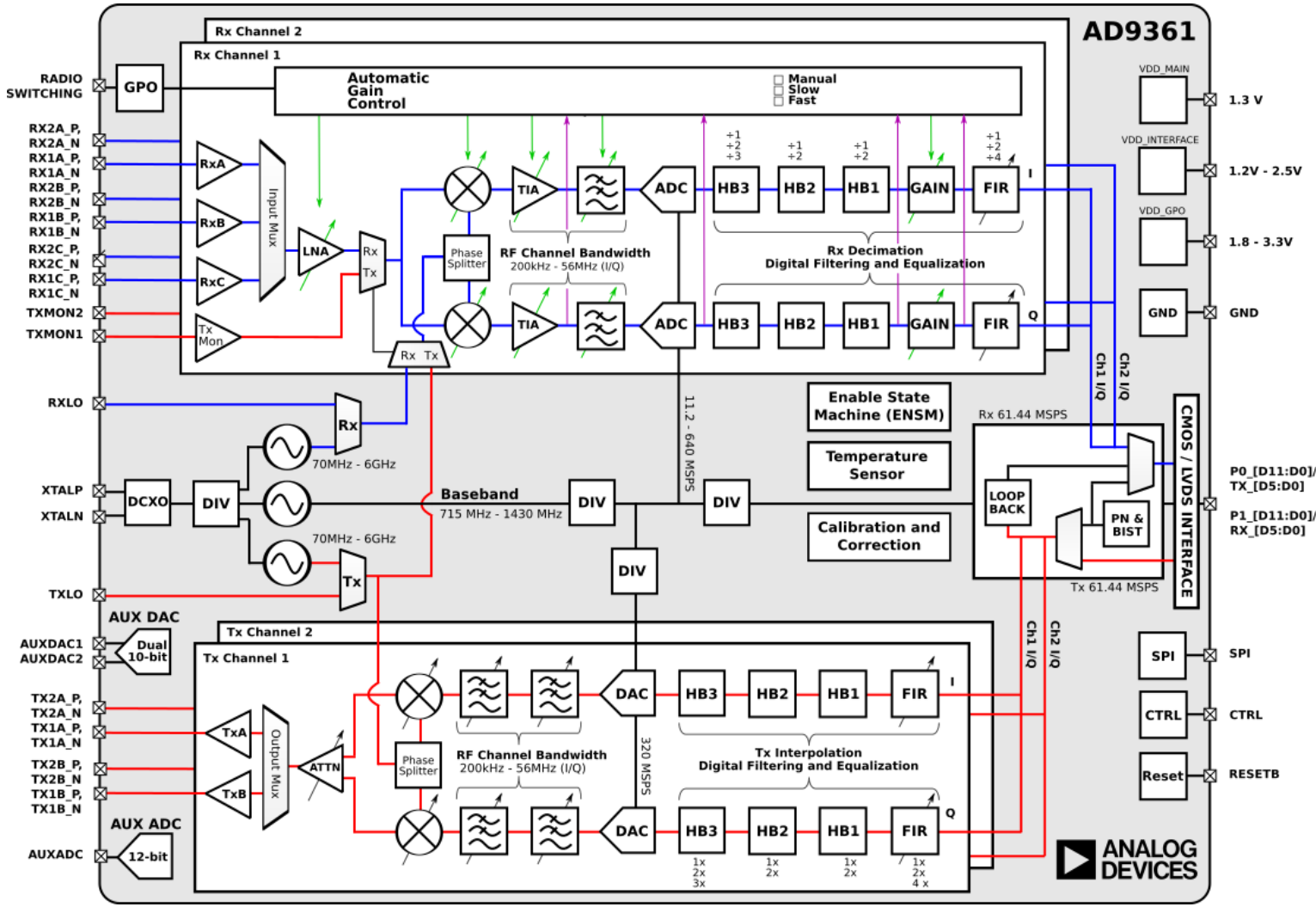


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

AD9361 / AD9364 Under the Hood

- AD9361: 2 Rx + 2 Tx
- AD9363: 2 Rx + 2 Tx
- AD9364: 1 Rx + 1 Tx
- Major sections:
  - RF input/output paths
  - RF PLL/LO
  - Clock generation
  - ADC/DAC
  - Digital filters
  - Digital interface
  - Enable state machine
  - RX Gain (AGC)
  - TX Attenuation
  - Aux DAC/ADC and GPOs
  - Analog and Digital Correction/Calibration



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Complexity of Transceiver and Support Provided

- AD9361 control
  - 1000+ registers
  - Internal automatic gain control
  - Internal TDD state machine control
  - Multiple data rate conversion states
  - Built-in calibrations and corrections
    - VCO
    - IQ
    - DC
  - Temperature sensors
  - XO control
  - …
- Infrastructure design to help development with hardware
- Software provided by ADI
  - Device drivers
    - Linux and/or No-OS
  - FPGA HDL
  - IIO scope
    - Data visualization application
    - Graphical configuration application
  - AD9361 Filter Wizard

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

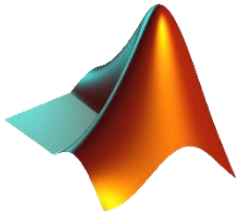
MathWorks®

Demo: IIO-Scope and Filter Wizard

ROBIN GETZ

Lab 1 Part 1: Streaming with IIO-Scope

MATLAB and Simulink



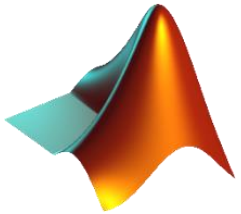
MATLAB

- Language of Technical Computing
- Large data sets
- Explore mathematics
- Data visualization

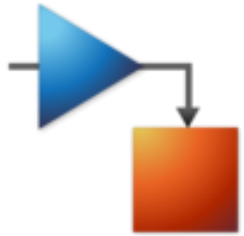
Simulink

- Model-Based Design
- Parallel architectures
- Timing
- Data type propagation
- Code generation

MATLAB and Simulink



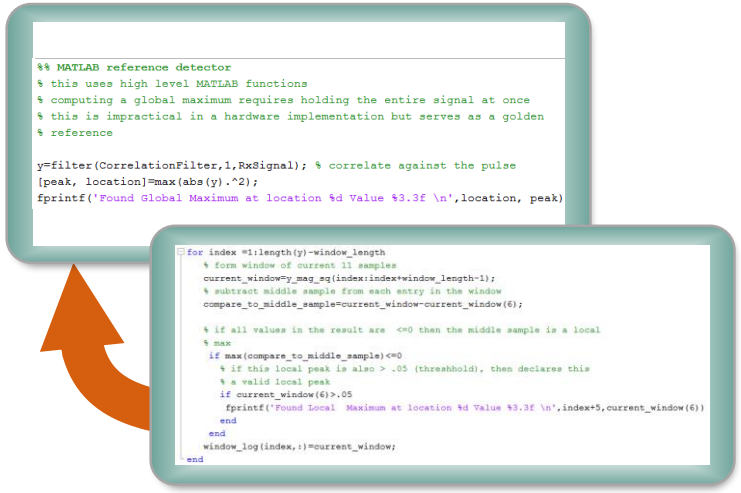
MATLAB



Simulink

Golden Reference

Hardware-connected algorithm



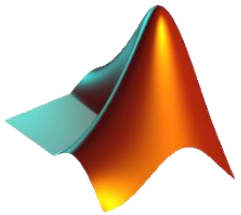
```
%% MATLAB reference detector
% this uses high level MATLAB functions
% computing a global maximum requires holding the entire signal at once
% this is impractical in a hardware implementation but serves as a golden
% reference

y=filter(CorrelationFilter,1,RxSignal); % correlate against the pulse
[peak, location]=max(abs(y).^2);
fprintf('Found Global Maximum at location %d Value %3.3f \n',location, peak)
```

```
for index =1:length(y)-window_length
    % form window of current 11 samples
    current_window=y_mag_sq(index:index+window_length-1);
    % subtract middle sample from each entry in the window
    compare_to_middle_sample=current_window-current_window(6);

    % if all values in the result are  <=0 then the middle sample is a local
    % max
     if max(compare_to_middle_sample)<=0
       % if this local peak is also > .05 (threshhold), then declares this
       % a valid local peak
       if current_window(6)>.05
        fprintf('Found Local  Maximum at location %d Value %3.3f \n',index+5,current_window(6))
       end
     end
    window_log(index,:)=current_window;
end
```

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

MATLAB and Simulink



MATLAB

Simulink

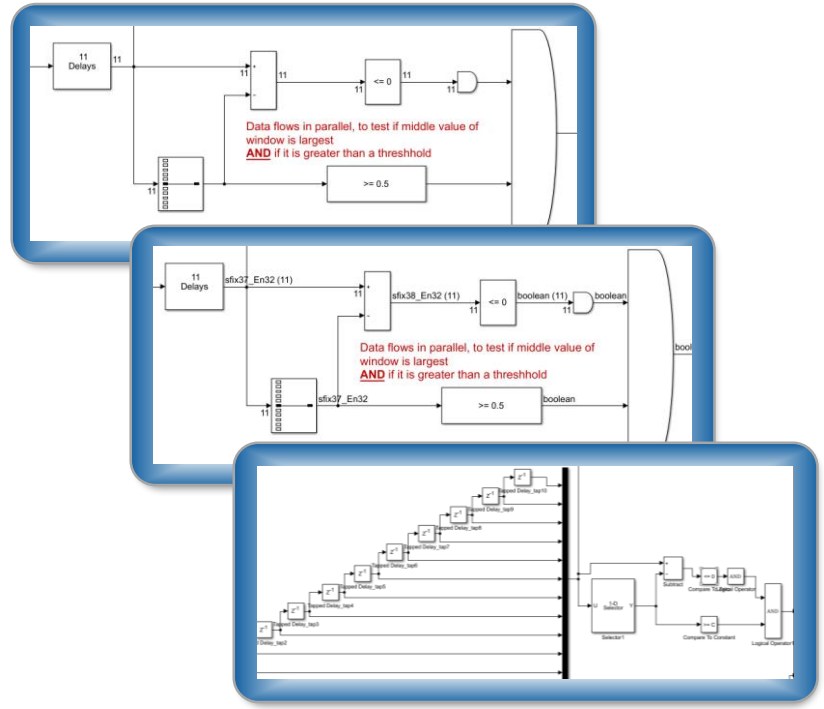
Golden Reference

Hardware-connected algorithm

Data-flow architecture

Fixed-point implementation

Optimization and code generation



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

MATLAB and Simulink

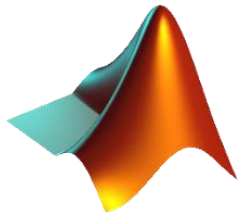
Golden Reference

Hardware-connected algorithm

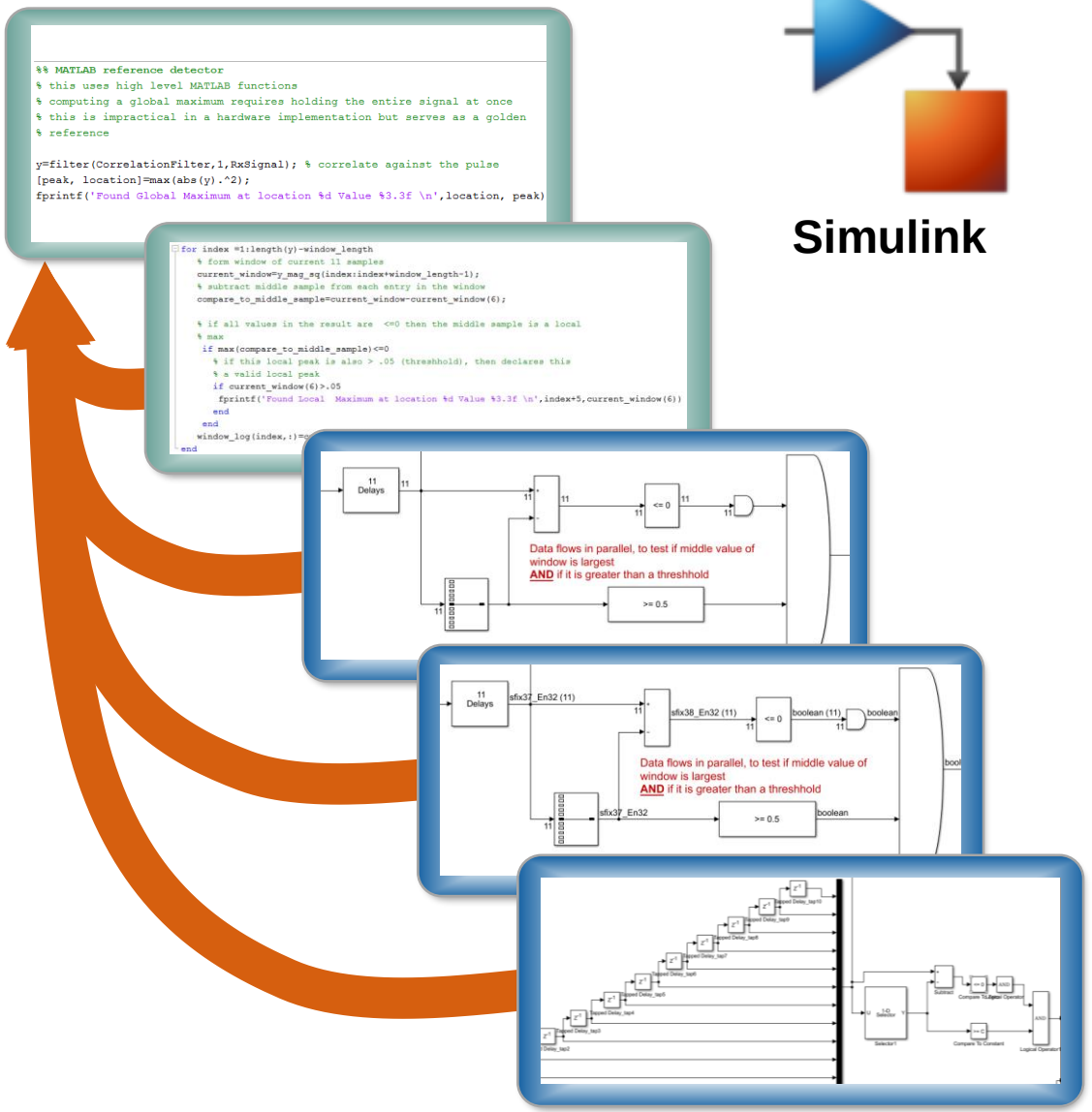
Data-flow architecture

Fixed-point implementation

Optimization and code generation



MATLAB



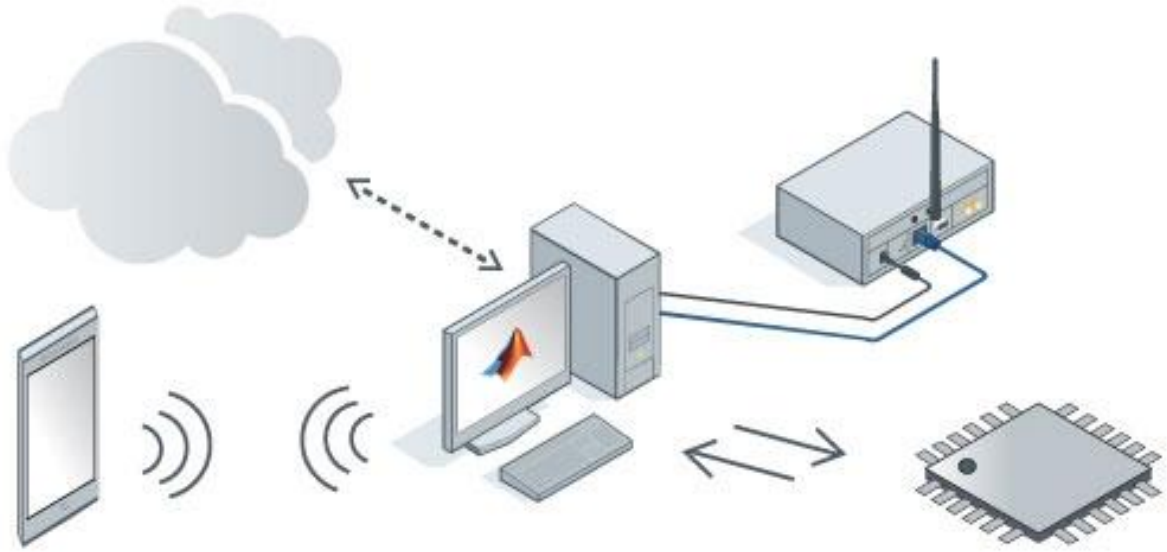
Simulink

MathWorks®

Connecting MATLAB and Simulink to Hardware

- MATLAB and Simulink are used for algorithm development and code generation
  - Not tied to any specific hardware platform

- Need a bridge to hardware system deployment
  - Hardware Support Package (HSP)
  - Reference Designs for custom hardware

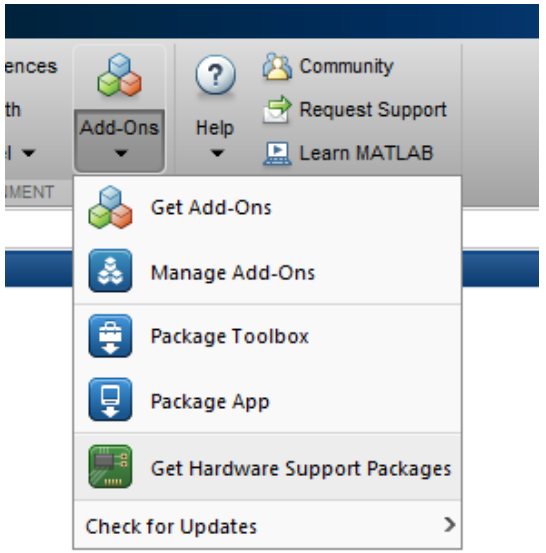


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Support Packages

- The bridge between MATLAB and Simulink and Hardware
- Enable radio I/O, prototyping, and production deployment
- **Hardware Support Packages** are available via the add-on explorer in MATLAB
  - Provide board-specific **reference designs** for C and HDL code generation
  - Provide portable Linux drivers for data I/O
- Third-party-authored **reference designs** enable custom hardware targeting
  - Leverages published APIs



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

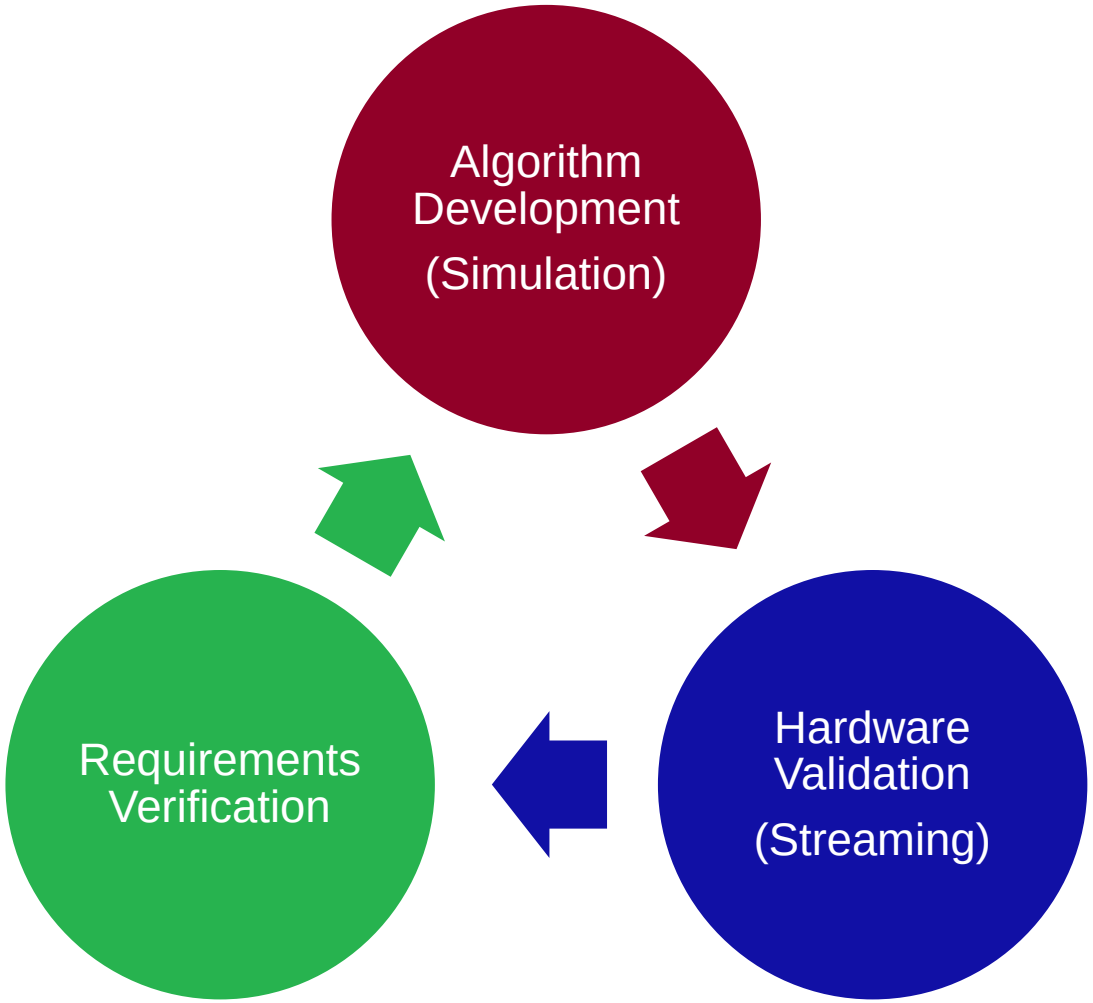
MathWorks®

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

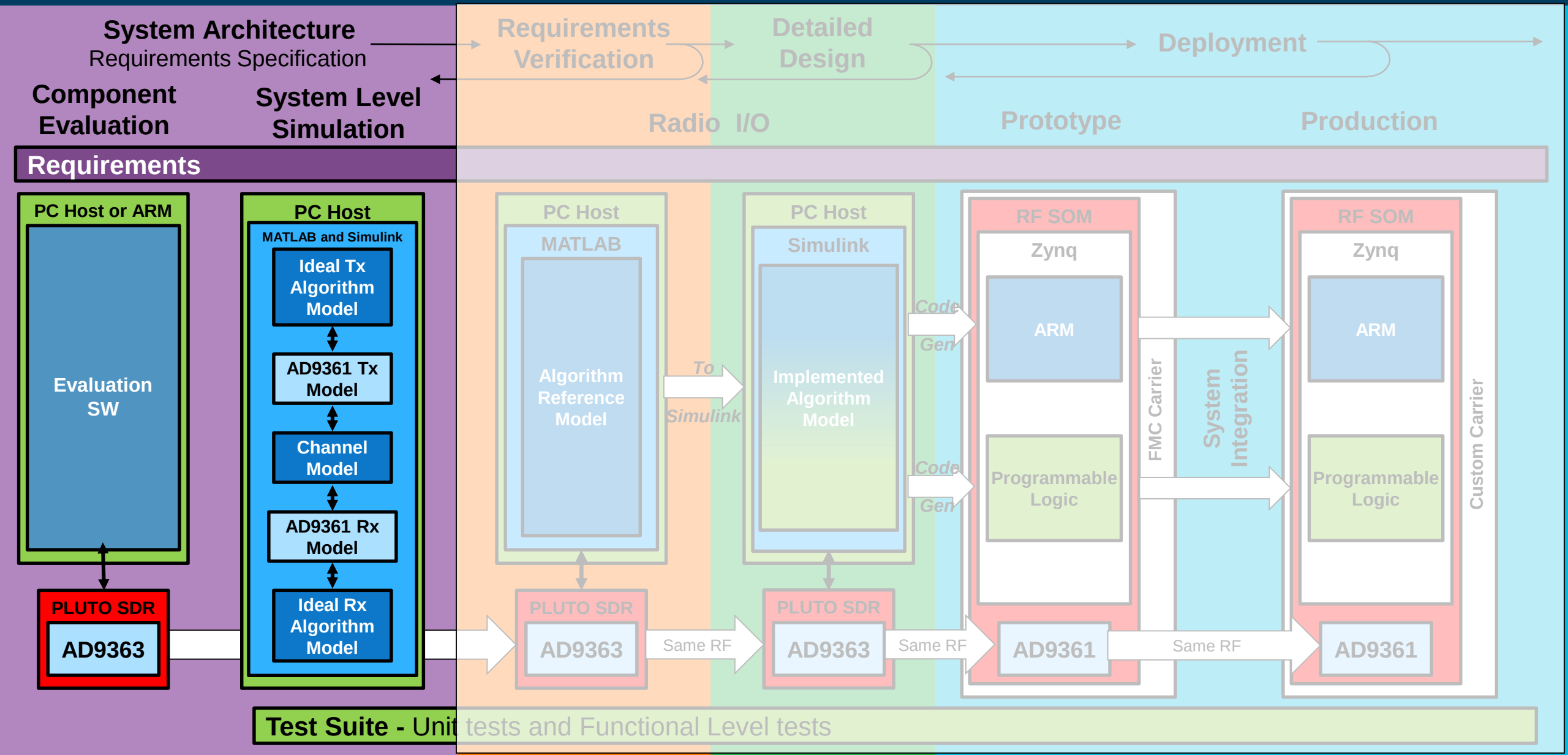
Morning Coffee Break

Algorithm Development and Hardware Evaluation

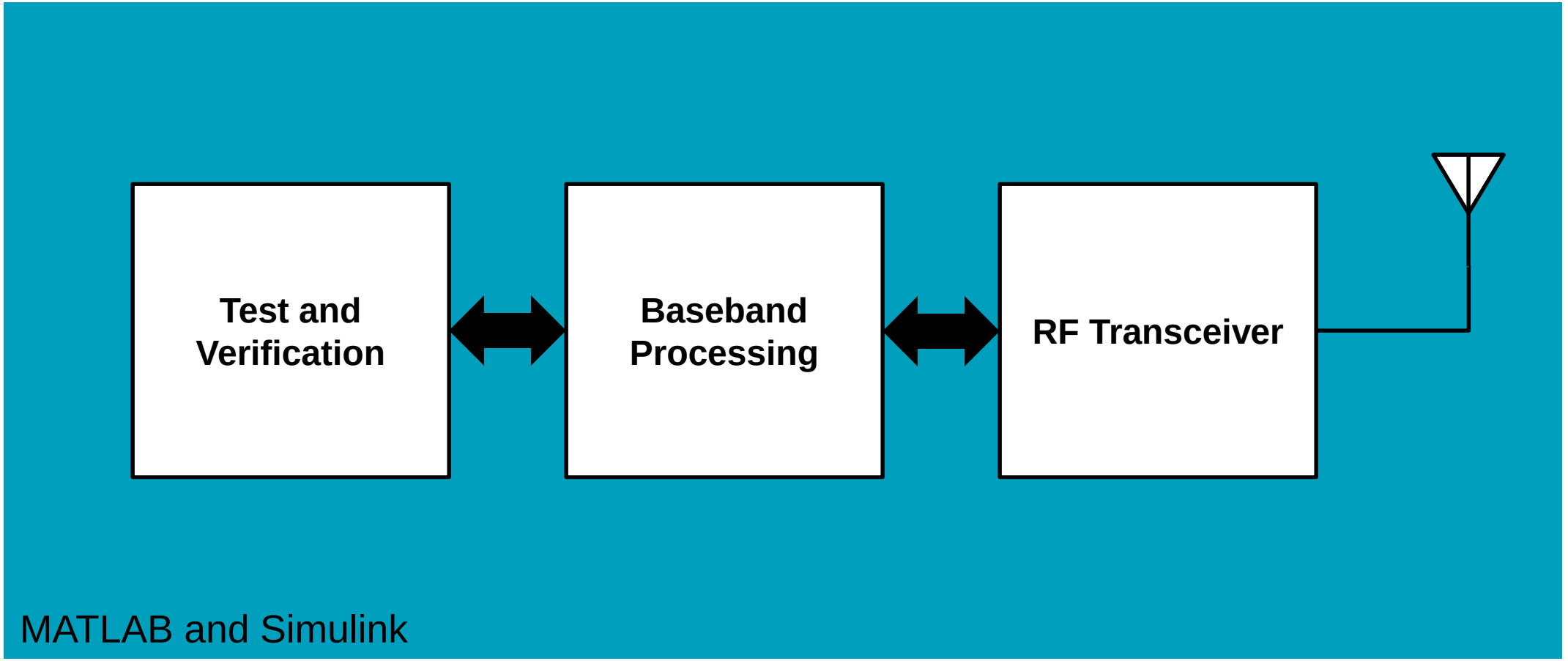


- Simulation of transmit and receiver path
  - RF Blockset™ model
  - Built-In and custom channel models
- Stream read data from identical deployment transceiver
  - PLUTOSDR
  - FMComms 2/3/4
  - RF-SOM
  - PackRF
  - Custom hardware

Model-Based Design for SDR



Elements of a Software-Defined Radio System
Modeling and simulation of the entire signal chain



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

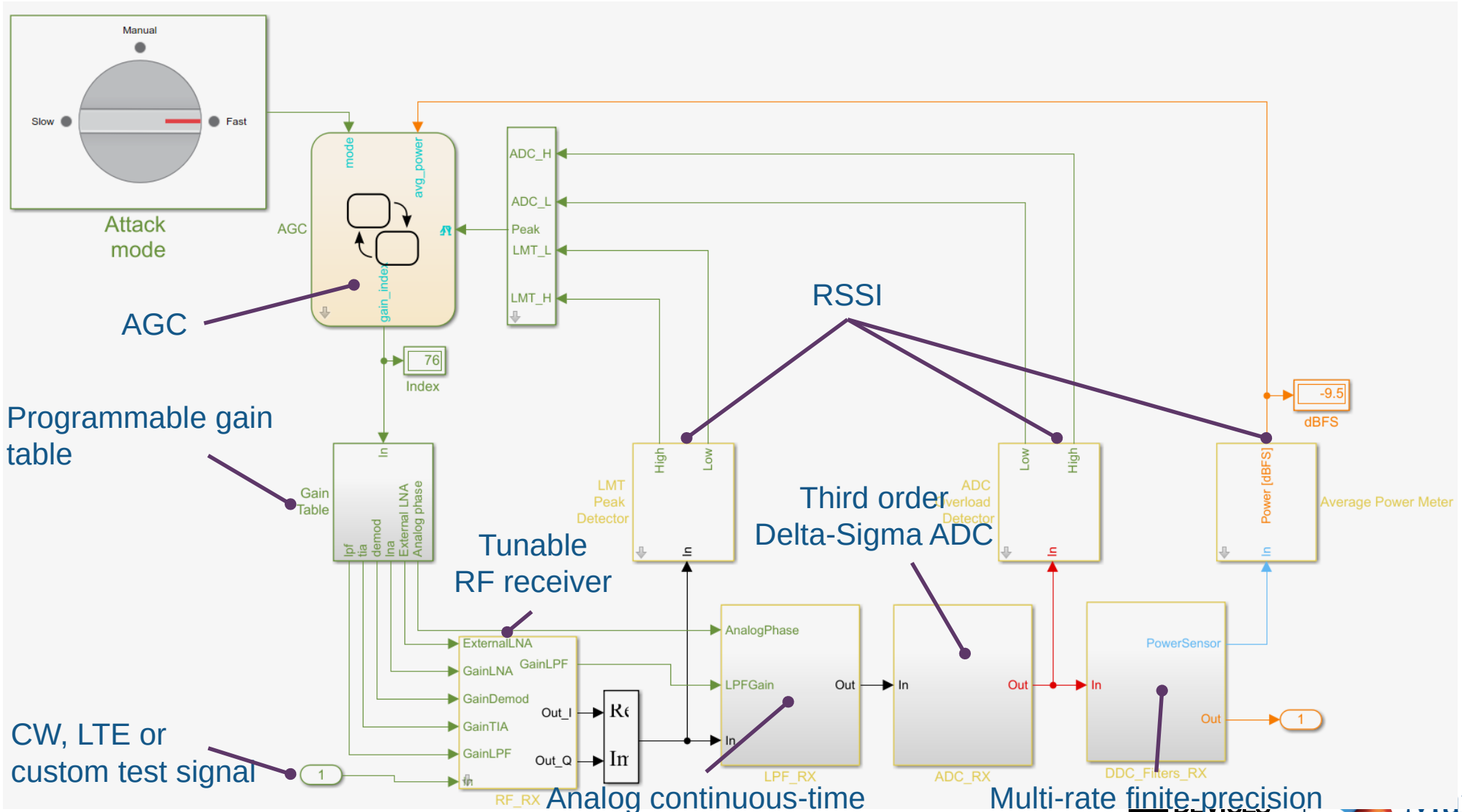
MathWorks®

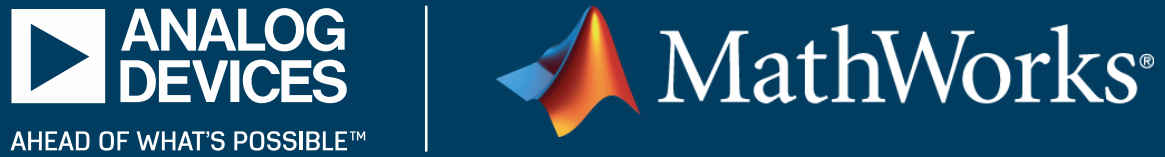
A True Multi-Domain System-Level Model

- Standard and custom test signals
- Tunable RF receiver
  - Gain dependent IP2, IP3, LO leakage, I/Q imbalance
- Third order delta-sigma ADC
- Programmable analog and digital filters
- AGC described with a time-triggered state machine
- Simulates 1 LTE frame (10ms) in minutes

- The simulation behavior validated against actual silicon

Executable Specification of AD9361 receive path

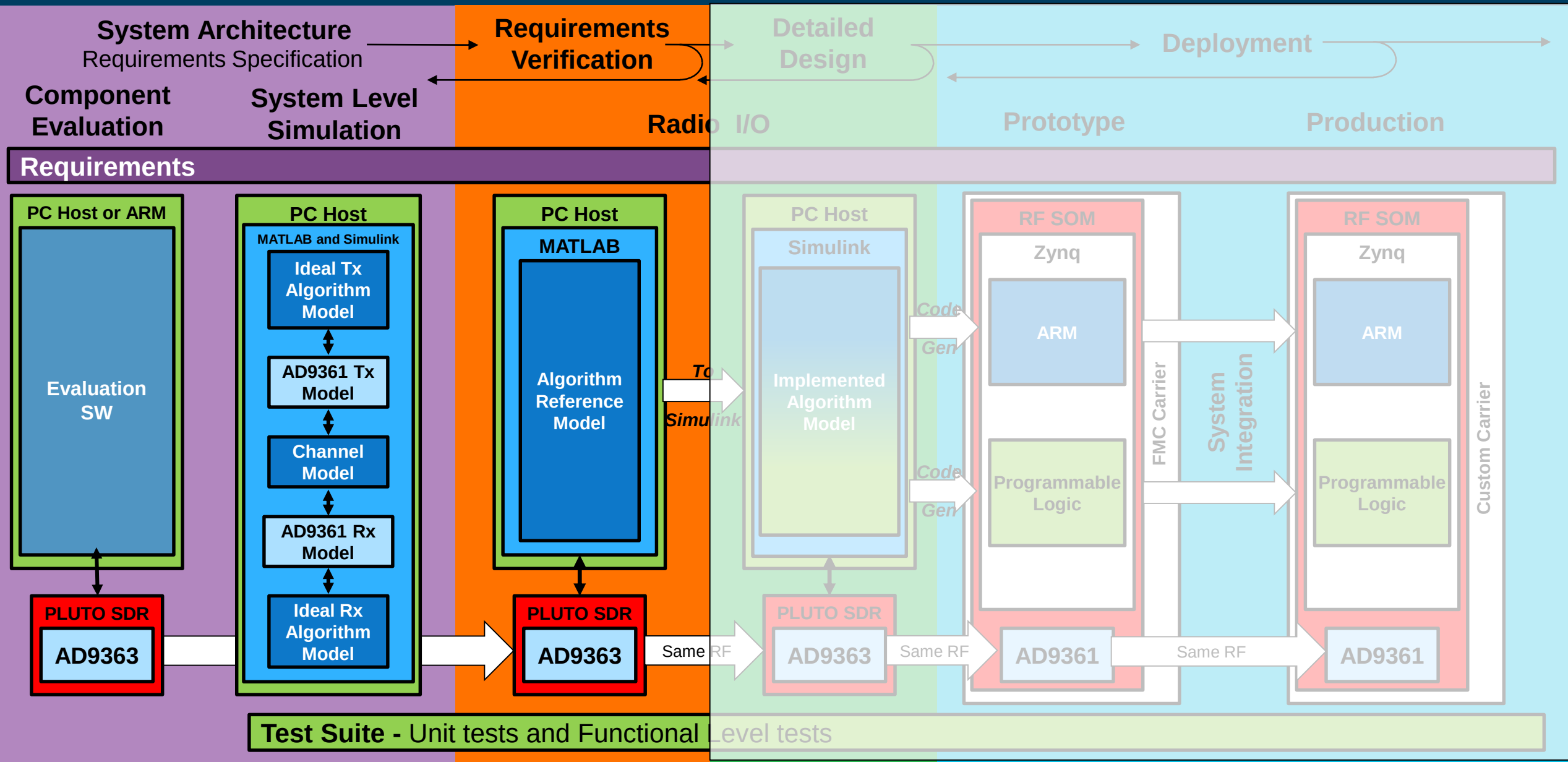




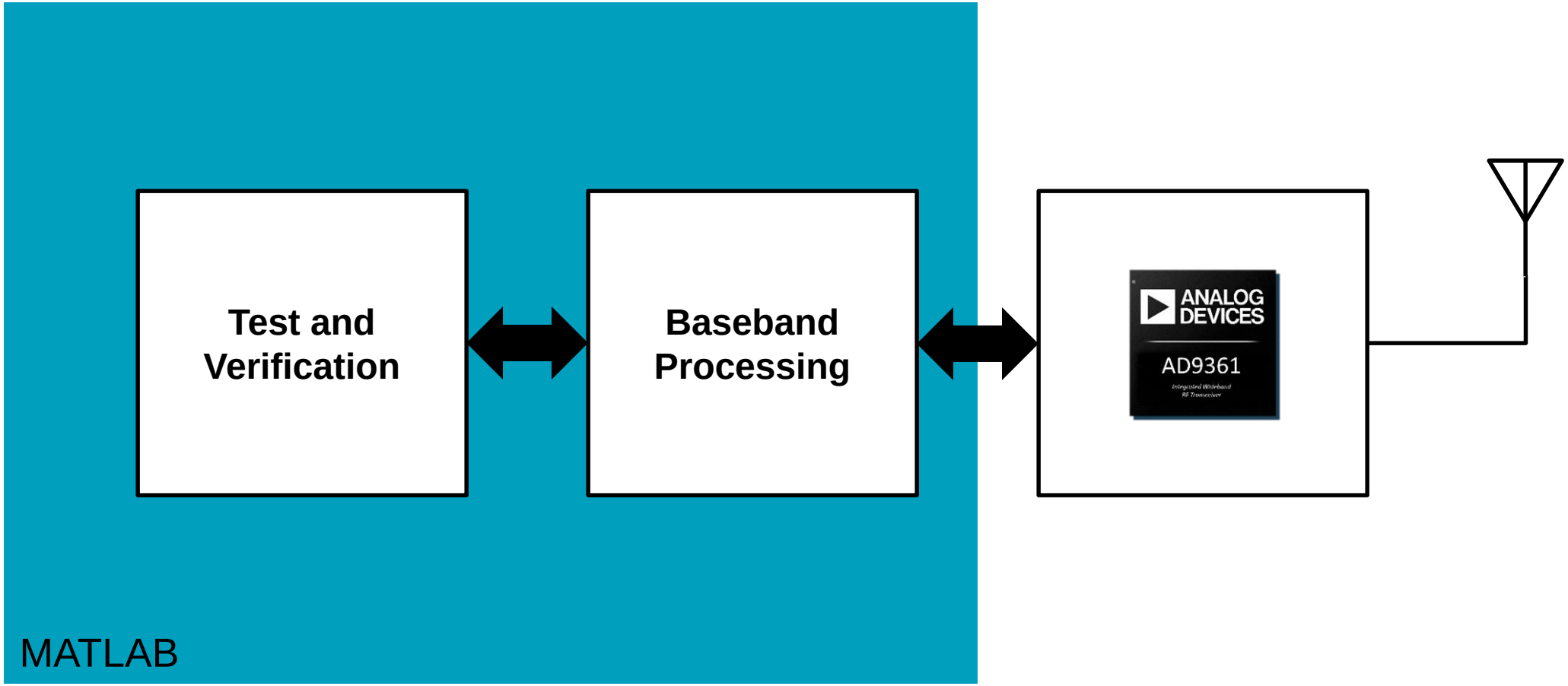
Streaming Data

DR. TRAVIS COLLINS, PHD

Model-Based Design for SDR



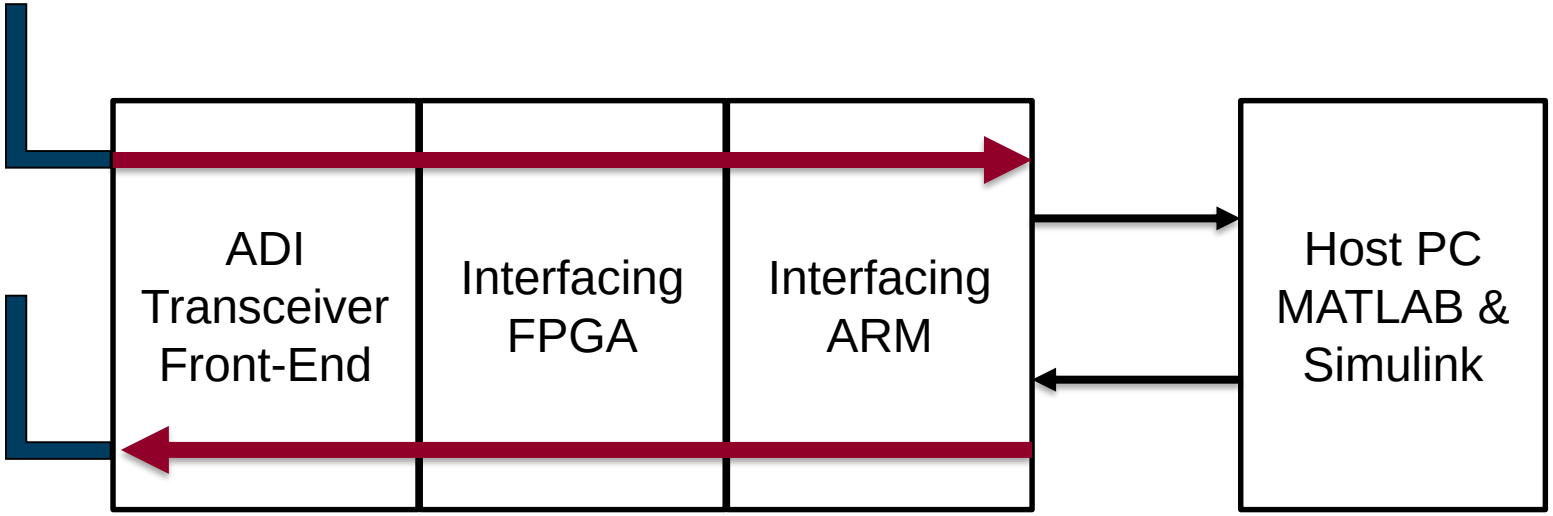
Data streaming with MATLAB



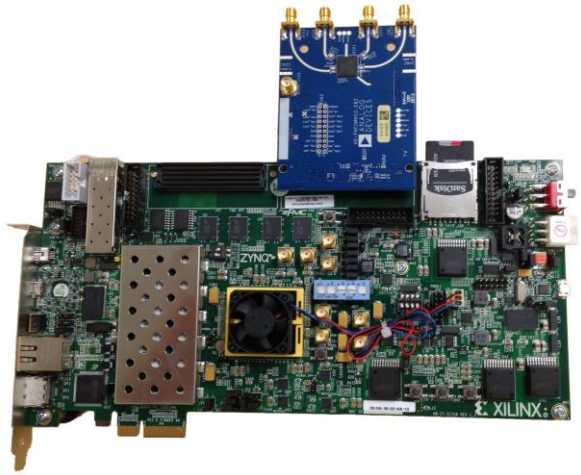
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

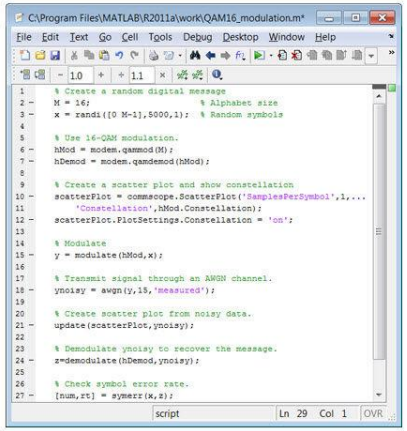
MathWorks®

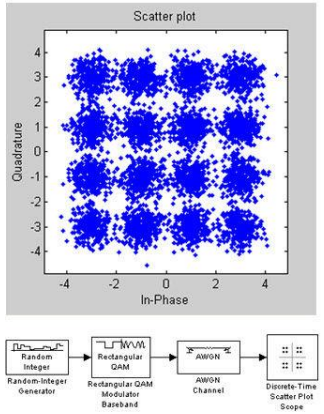
Transceiver Focused: Basic Idea



- Controlling transceiver
- Getting data to and from transceiver



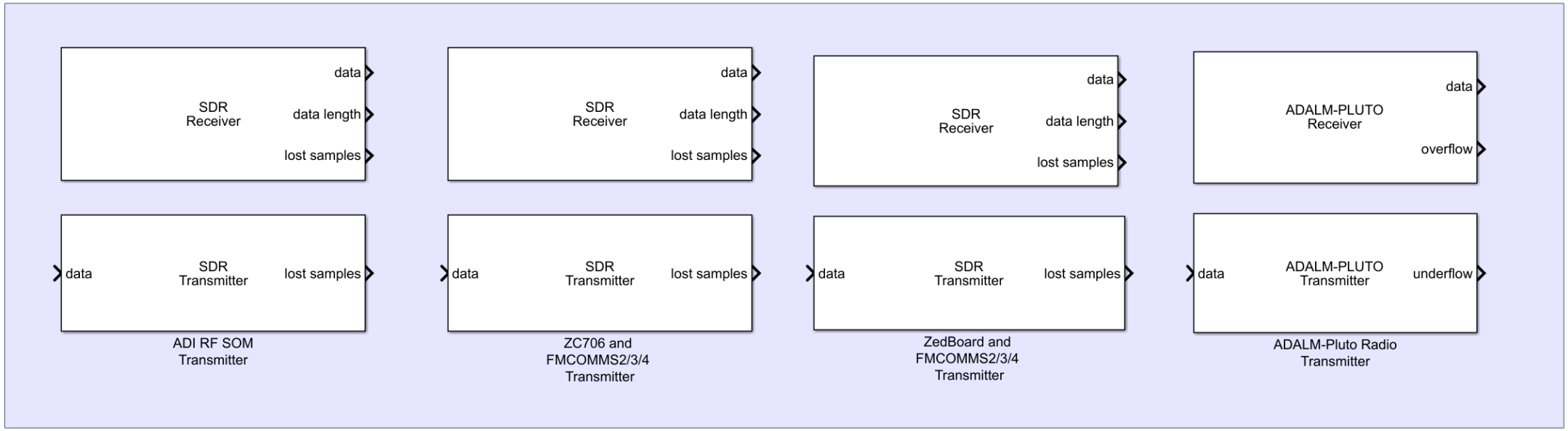




ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Blocks and System Objects




```
>> rx = sdrrx('ZC706 and FMCOMMS2/3/4');
>> rx = sdrrx('ZedBoard and FMCOMMS2/3/4');
>> rx = sdrrx('ADI RF SOM');
>> rx = sdrrx('Pluto')
```

```
rx =

  comm.SDRRxPluto with properties:

                DeviceName: 'Pluto'
                   RadioID: 'usb:0'
           CenterFrequency: 2.4000e+09
                GainSource: 'AGC Slow Attack'
            ChannelMapping: 1
        BasebandSampleRate: 1000000
            OutputDataType: 'int16'
           SamplesPerFrame: 3660
     ShowAdvancedProperties: false
```

Data Streaming with MATLAB

- This example shows data streaming from hardware into MATLAB
- Uses PLUTOSDR module to stream data into MATLAB

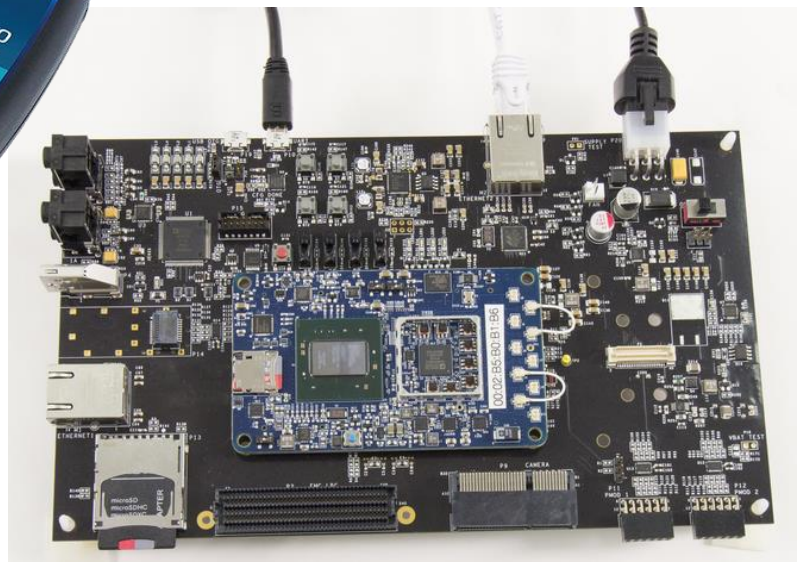
- Tools used:
  - Software
    - MATLAB, DSP System Toolbox, Signal Processing Toolbox, Communications System Toolbox
  - Support Package
    - Communications System Toolbox Support Package for Analog Devices ADALM-PLUTO Radio

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Hardware Platforms to be used in this Workshop

- Streaming Platform: ADALM-PLUTO
  - Based on AD9363 RF Transceiver
  - RF coverage from 325MHz to 3.8GHz
  - Single-channel receive and transmit
- Prototyping Platform: ADRV9361 RF SOM
  - Based on AD9361 RF Transceiver
  - RF coverage from 70MHz to 6GHz
  - Dual-channel receive and transmit
  - Fully-verified, production-ready SOM
- Custom Hardware: “PackRF”
  - Based on ADRV9361 RF SOM with custom carrier board
  - Portable radio with video I/O capabilities
  - Custom reference design example



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Demo

MATLAB INSTRUCTOR LEAD DEMO

Real time streaming

DR. TRAVIS COLLINS, PHD

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

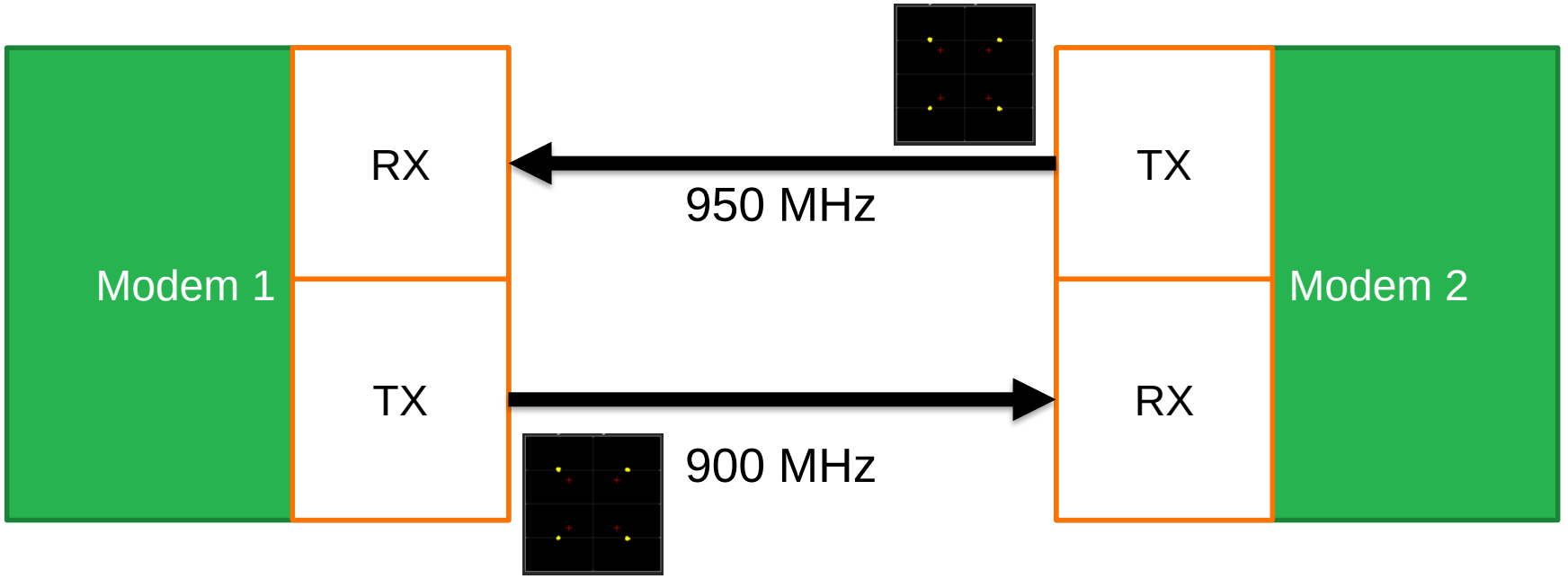
Lab 1 Part B: Streaming in MATLAB

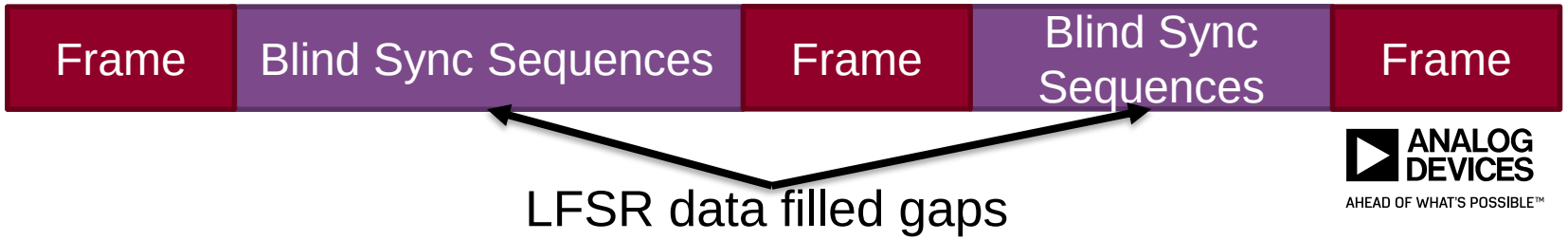
Modem Design Intro

DR. TRAVIS COLLINS, PHD

Physical layer Operational Design

- QPSK physical layer with continuous transmission/reception based link
- Simple FDD system (MAC)
- Built with common algorithms



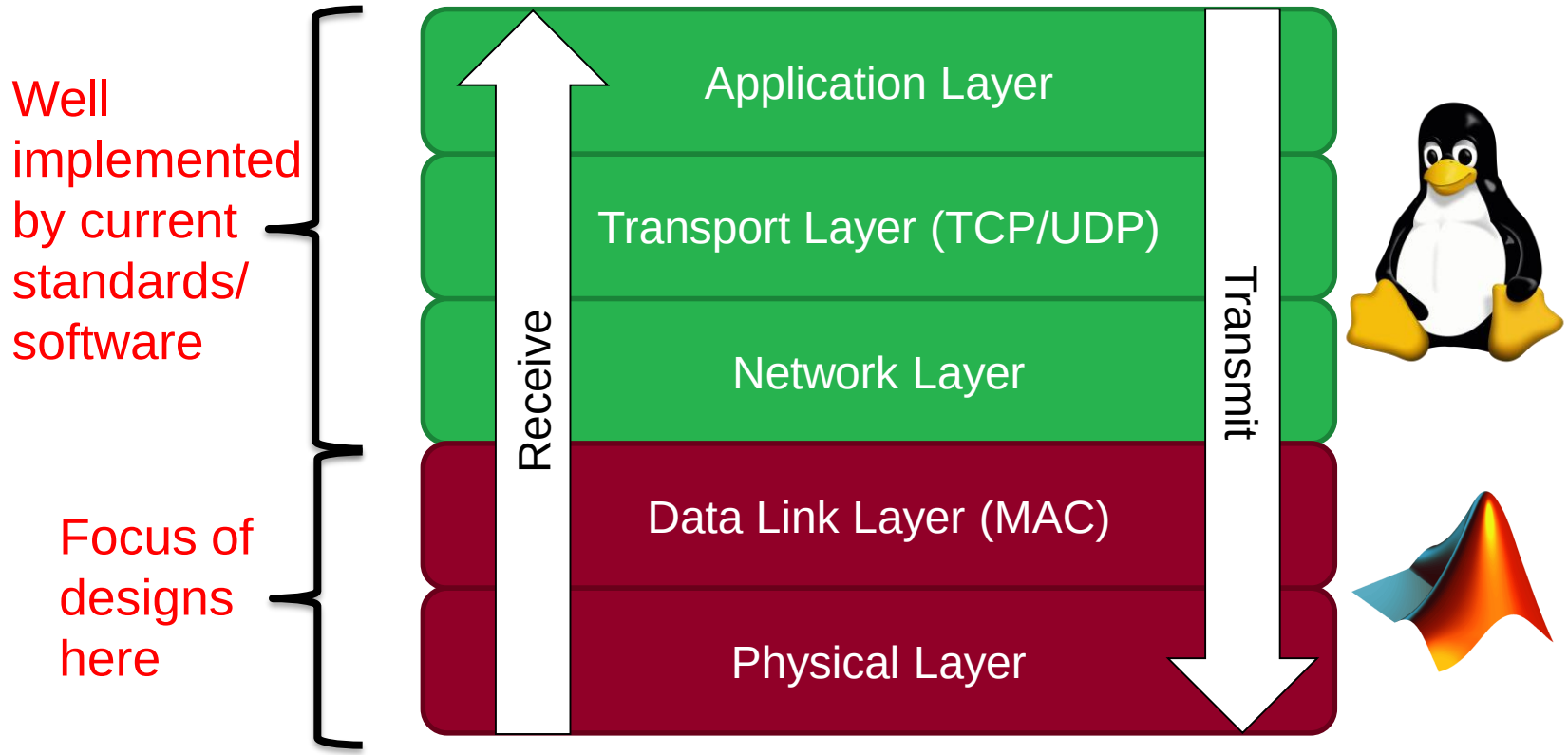


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Modem Pieces and the OSI/TCP-IP Model

- SDR Focused
  - Data Link Layer (MAC)
    - Channel access
    - Frame validation and error checking
  - Physical Layer
    - Transmit bits
    - Recover bits
- Operating System and User Space Focused
  - Application/Transport/Network

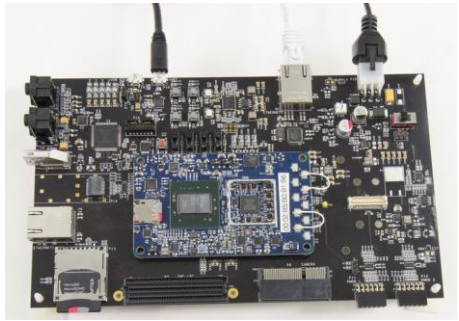


Useful to understand stack interfaces since layers will influence one another

Modem Build Process

- Modem built through workflow previously outlined
- All algorithm design was done initially in MATLAB
  - Testing harness was started when MATLAB reference design was built
  - Streaming was heavily used for verification
  - Utilization of many off the shelf algorithms (CST)
- Reference was converted into Simulink in stages

MATLAB Reference Design → Simulink Floating-Point Model → Simulink HDL Capable Model

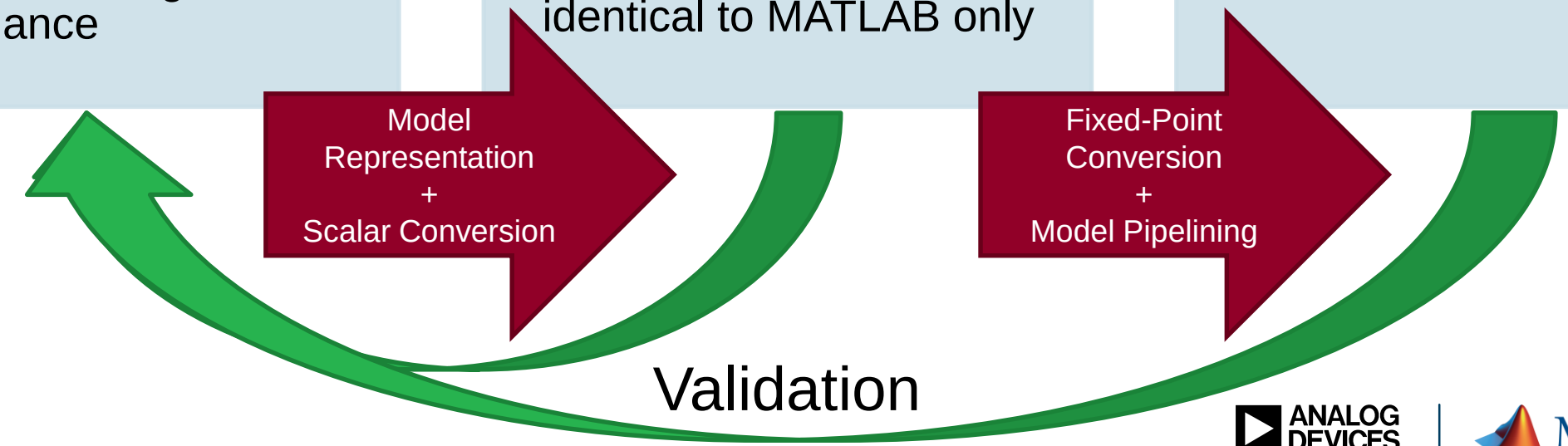


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Algorithm Design Process: Code (MATLAB) and Models (Simulink)

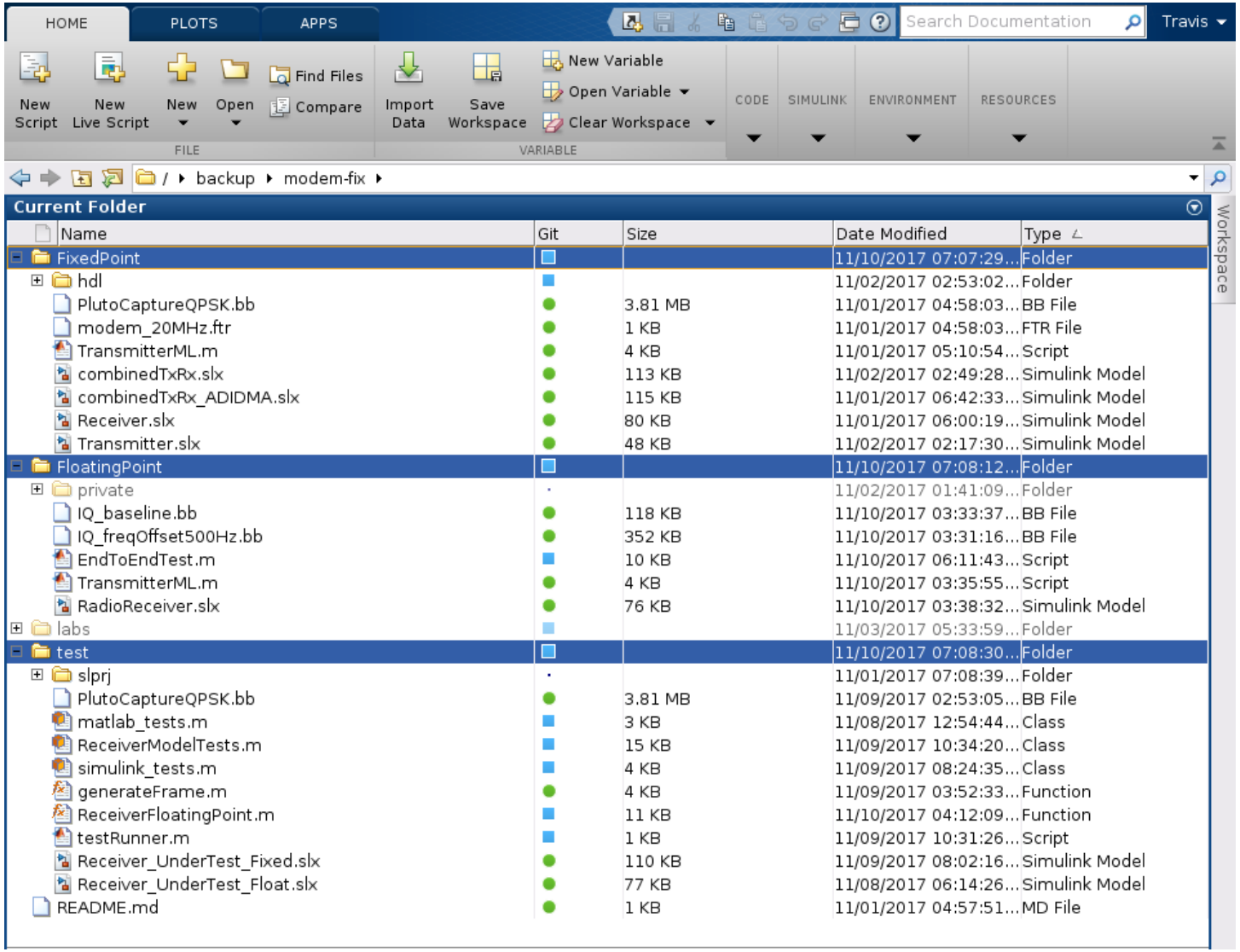
MATLAB: Floating-Point Reference Model	*Simulink:* Implemented Algorithm Model	*Simulink* Fixed-Point HDL Capable Model
• Vector based • Fast running • Simple to debug • Best possible algorithmic performance	• *Scalar design* • Control signal propagation model • Algorithmic performance identical to MATLAB only	• *Fixed-Point HDL Capable* • Meets test points for performance validation



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

What's Included?



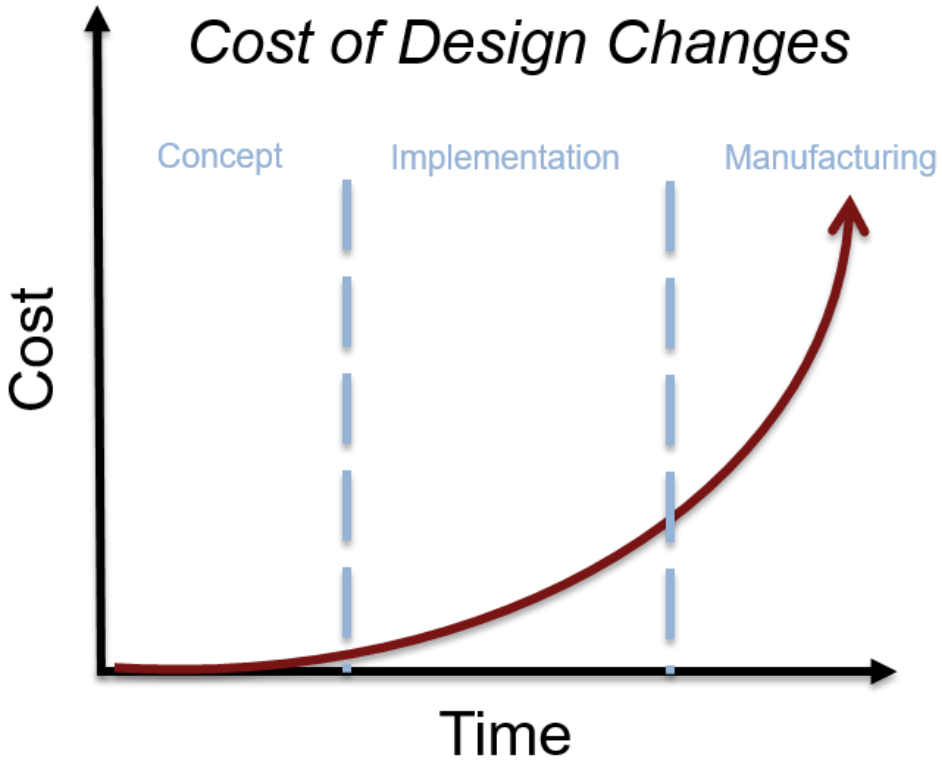
Name	Git	Size	Date Modified	Type
FixedPoint			11/10/2017 07:07:29...	Folder
hdl			11/02/2017 02:53:02...	Folder
PlutoCaptureQPSK.bb		3.81 MB	11/01/2017 04:58:03...	BB File
modem_20MHz.ftr		1 KB	11/01/2017 04:58:03...	FTR File
TransmitterML.m		4 KB	11/01/2017 05:10:54...	Script
combinedTxRx.slx		113 KB	11/02/2017 02:49:28...	Simulink Model
combinedTxRx_ADIDMA.slx		115 KB	11/01/2017 06:42:33...	Simulink Model
Receiver.slx		80 KB	11/01/2017 06:00:19...	Simulink Model
Transmitter.slx		48 KB	11/02/2017 02:17:30...	Simulink Model
FloatingPoint			11/10/2017 07:08:12...	Folder
private			11/02/2017 01:41:09...	Folder
IQ_baseline.bb		118 KB	11/10/2017 03:33:37...	BB File
IQ_freqOffset500Hz.bb		352 KB	11/10/2017 03:31:16...	BB File
EndToEndTest.m		10 KB	11/10/2017 06:11:43...	Script
TransmitterML.m		4 KB	11/10/2017 03:35:55...	Script
RadioReceiver.slx		76 KB	11/10/2017 03:38:32...	Simulink Model
labs			11/03/2017 05:33:59...	Folder
test			11/10/2017 07:08:30...	Folder
slprj			11/01/2017 07:08:39...	Folder
PlutoCaptureQPSK.bb		3.81 MB	11/09/2017 02:53:05...	BB File
matlab_tests.m		3 KB	11/08/2017 12:54:44...	Class
ReceiverModelTests.m		15 KB	11/09/2017 10:34:20...	Class
simulink_tests.m		4 KB	11/09/2017 08:24:35...	Class
generateFrame.m		4 KB	11/09/2017 03:52:33...	Function
ReceiverFloatingPoint.m		11 KB	11/10/2017 04:12:09...	Function
testRunner.m		1 KB	11/09/2017 10:31:26...	Script
Receiver_UnderTest_Fixed.slx		110 KB	11/09/2017 08:02:16...	Simulink Model
Receiver_UnderTest_Float.slx		77 KB	11/08/2017 06:14:26...	Simulink Model
README.md		1 KB	11/01/2017 04:57:51...	MD File

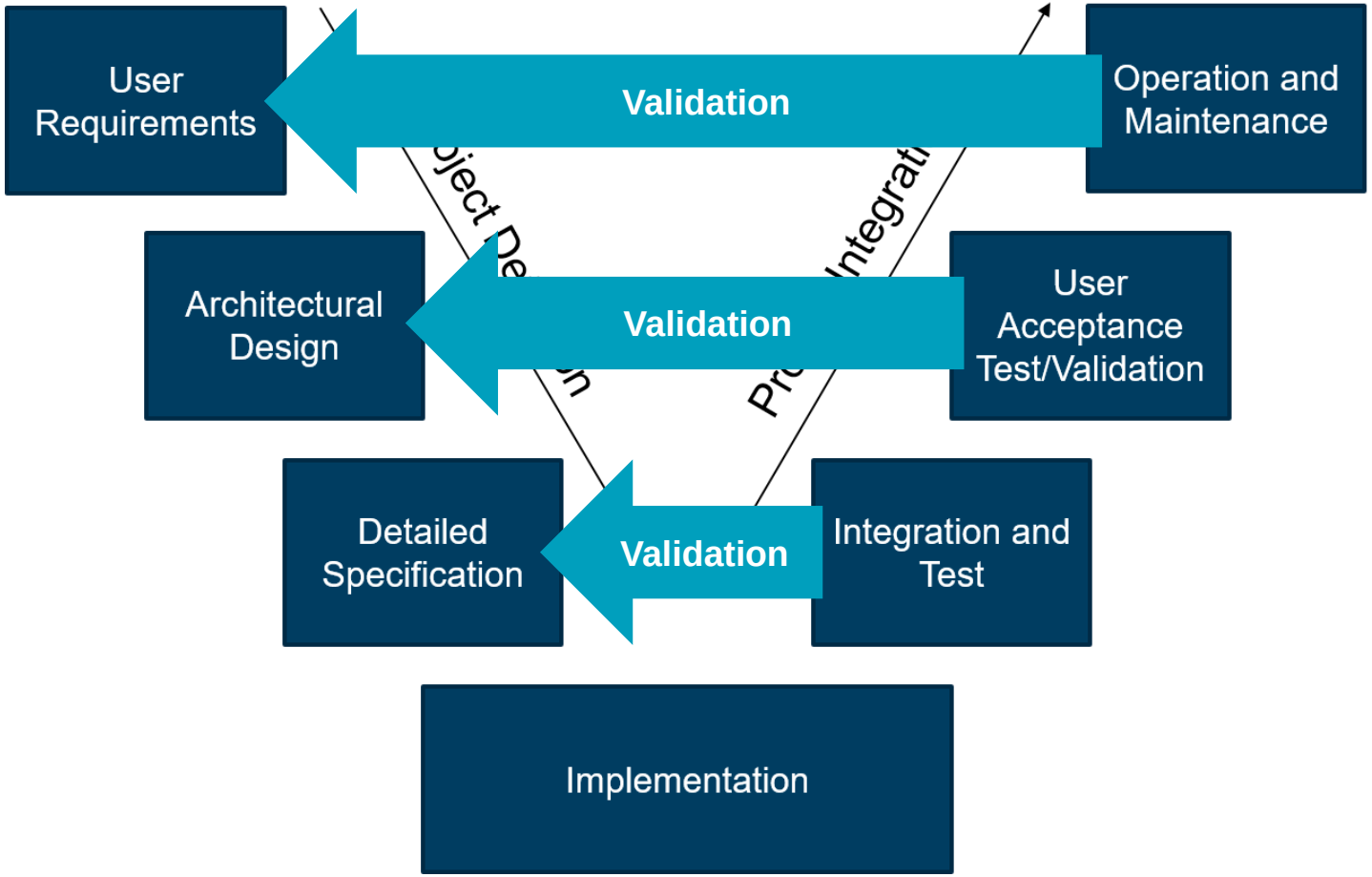
- Three main folders
  - FixedPoint
  - FloatingPoint
  - test
- Included
  - Data captures
  - Simulink Libraries
  - Models
  - Scripts
  - Test harness

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Using Best Design and Implementation Practices



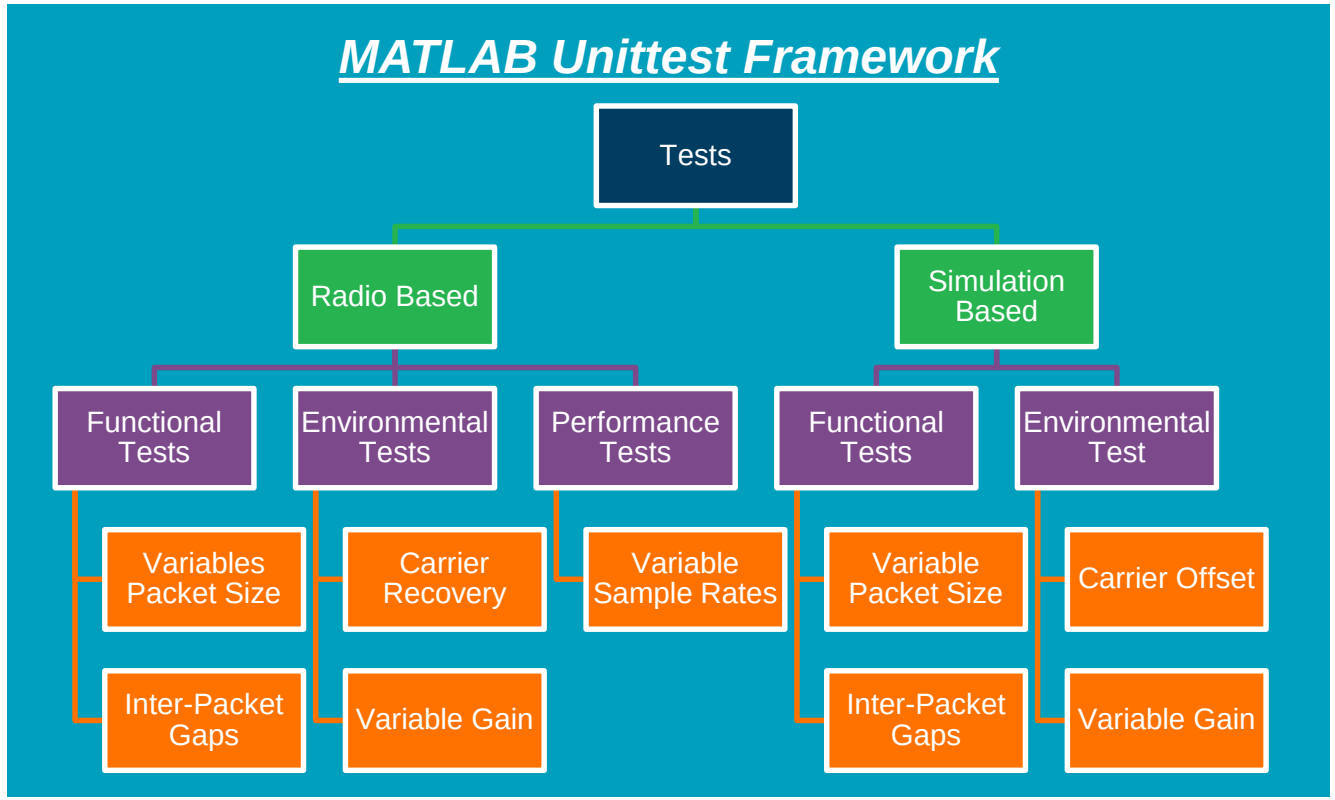


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

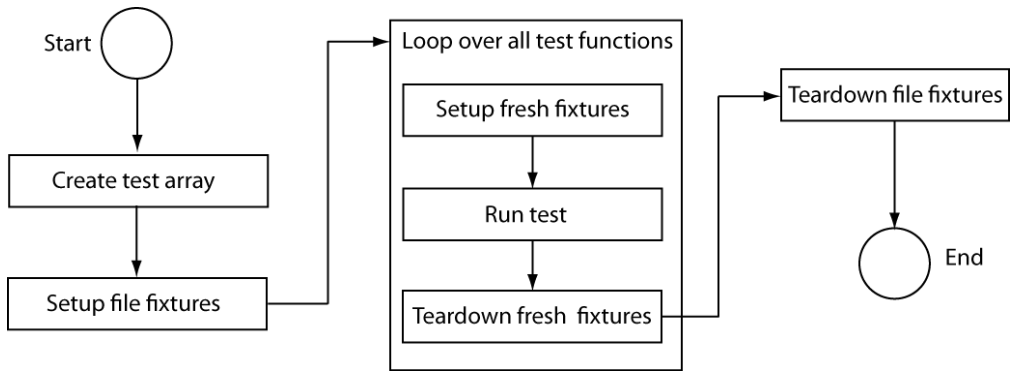
MathWorks®

Testing Harness

Requirements → *Test Cases*



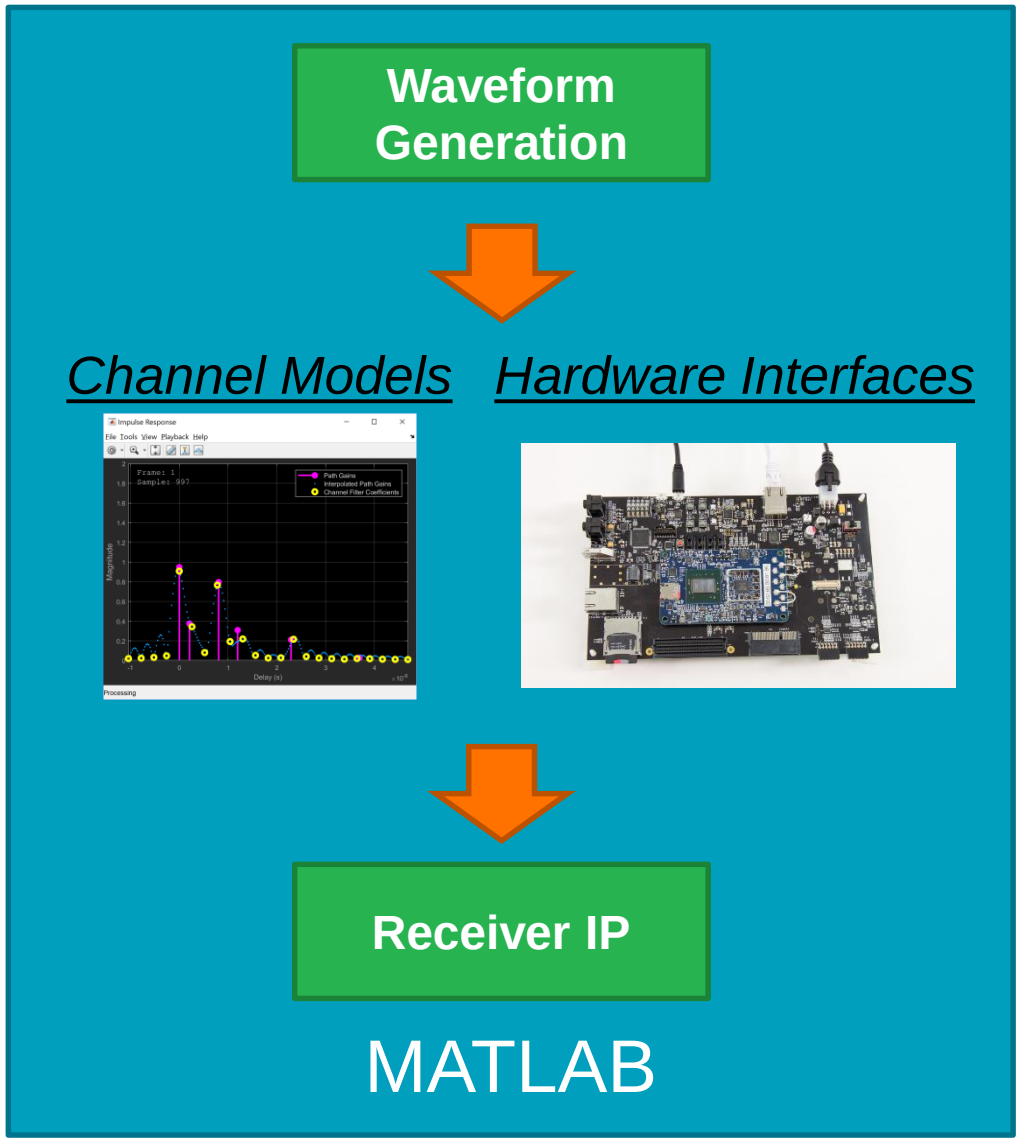
- Testing is driven by requirements
- Built on top of MATLAB Unittest
  - xUnit style
- Harness Features
  - All three models are tested
  - Leverages both channel models and hardware connectivity



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

MATLAB/Simulink Test Integration




```
    % Descramble
    rxDescram = descr(rxData);
    % Viterbi decode the demodulated data
    dataHard = vitdec(rxDescram,trellis,tbl,'cont','hard');
    % Removing coding delay
    rxDataWithTail = dataHard(tbl+1:end);
    % Remove tail bits
    rxDataWithCRC = rxDataWithTail(1:end-nTail);
    % Check CRC
    [rxData,e] = crcDec(rxDataWithCRC);
    if e
        log(testCase,2,'CRC Failed.');
        crcChecks = [crcChecks;1];
        failures = [failures;4];
    else
        log(testCase,2,'CRC Passed.');
        crcChecks = [crcChecks;0];
        failures = [failures;0]; %#ok<*AGROW>
        packetBits = {packetBits;rxData};
    end

end

log(testCase,4,'Receiver done processing data.');
% Pack results
results = struct('packetsFound',packetsFound,...
    'crcChecks',crcChecks,'failures',failures);
```

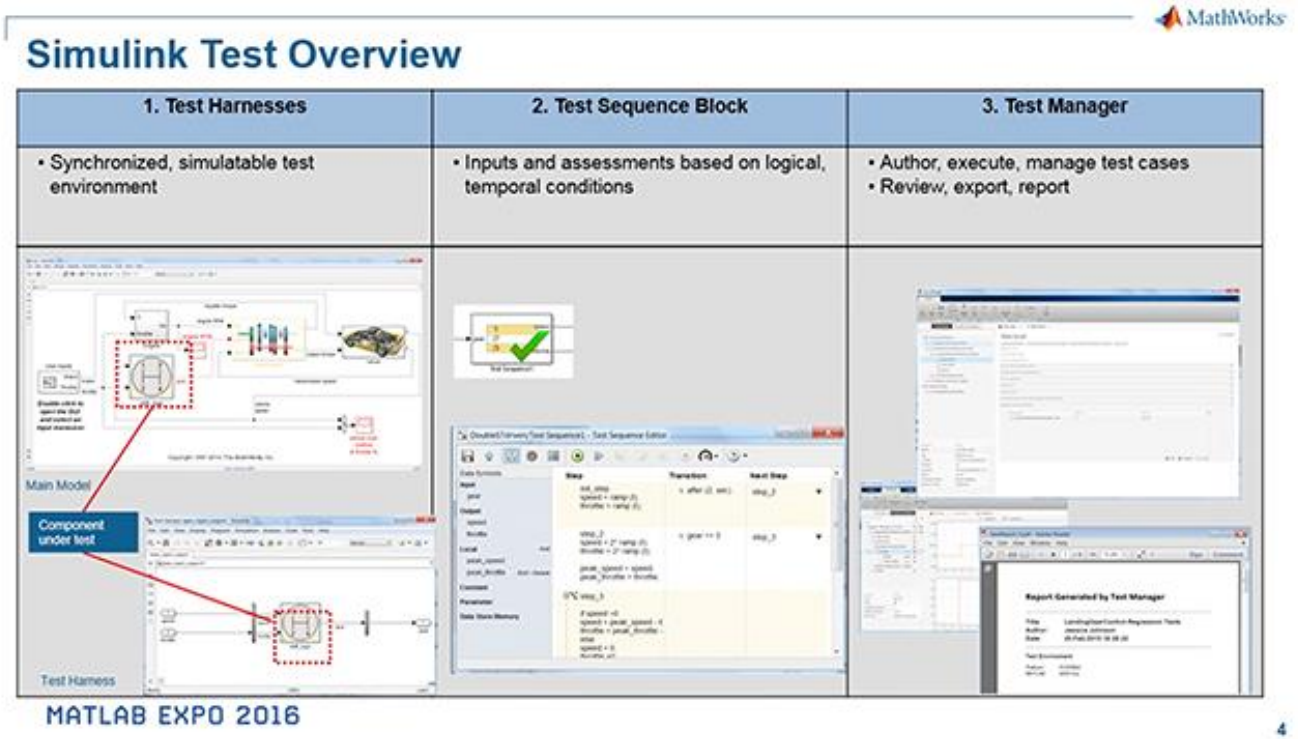
Similar features apply to Simulink models

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

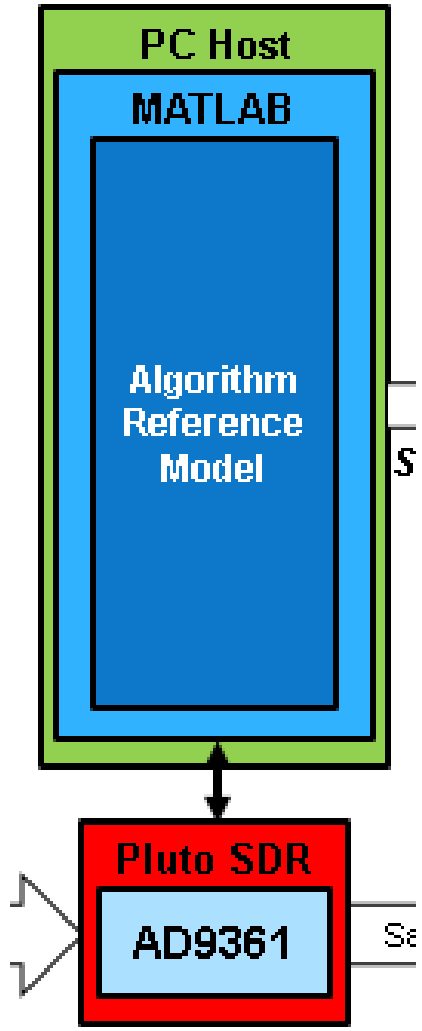
Deep Test Infrastructure In Simulink

- With Simulink Test and Simulink Requirements™, you can link test cases to requirements
  - Microsoft® Word
  - IBM® Rational® DOORS®
  - Other documents types
- Support for industry standards is available through:
  - IEC Certification Kit (for IEC 61508 and ISO 26262)
  - DO Qualification Kit (for DO-178).

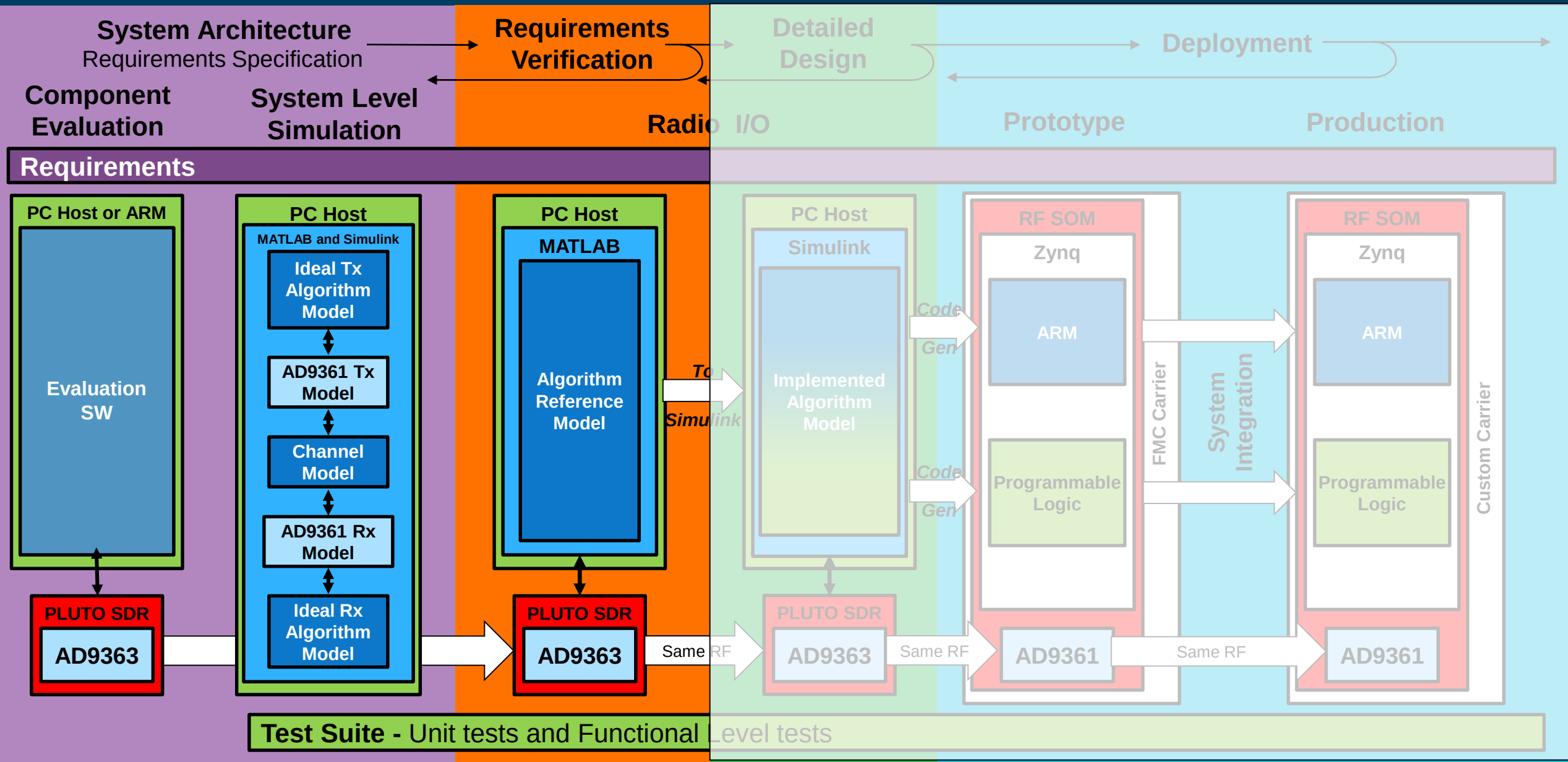


Review

- MATLAB is a powerful tool to develop algorithms
- MATLAB can stream data directly to and from ADI hardware
- Detailed RF models of AD9361 transceiver available for complete simulations
- The modem example so far:
  - PHY and MAC layer simple
  - Uses a test harness to automate simulations and radio-in-the-loop testing



Model-Based Design for SDR



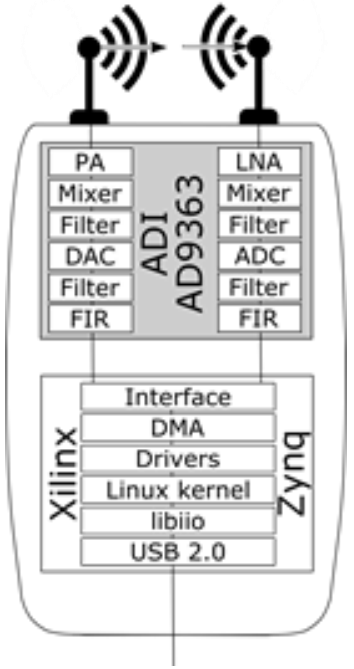
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

Lab 1 Part C: Test Harness Intro

TRAVIS COLLINS

Data Streaming with MATLAB

- This lab will explore data streaming from hardware into MATLAB
- Tools used:
  - Software
    - MATLAB, DSP System Toolbox, Signal Processing Toolbox, Communications System Toolbox
  - Support Package
    - Communications System Toolbox Support Package for Analog Devices ADALM-PLUTO Radio
  - Hardware
    - ADALM-PLUTO from Analog Devices
      - Based on AD9363 RF Transceiver



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Lunch

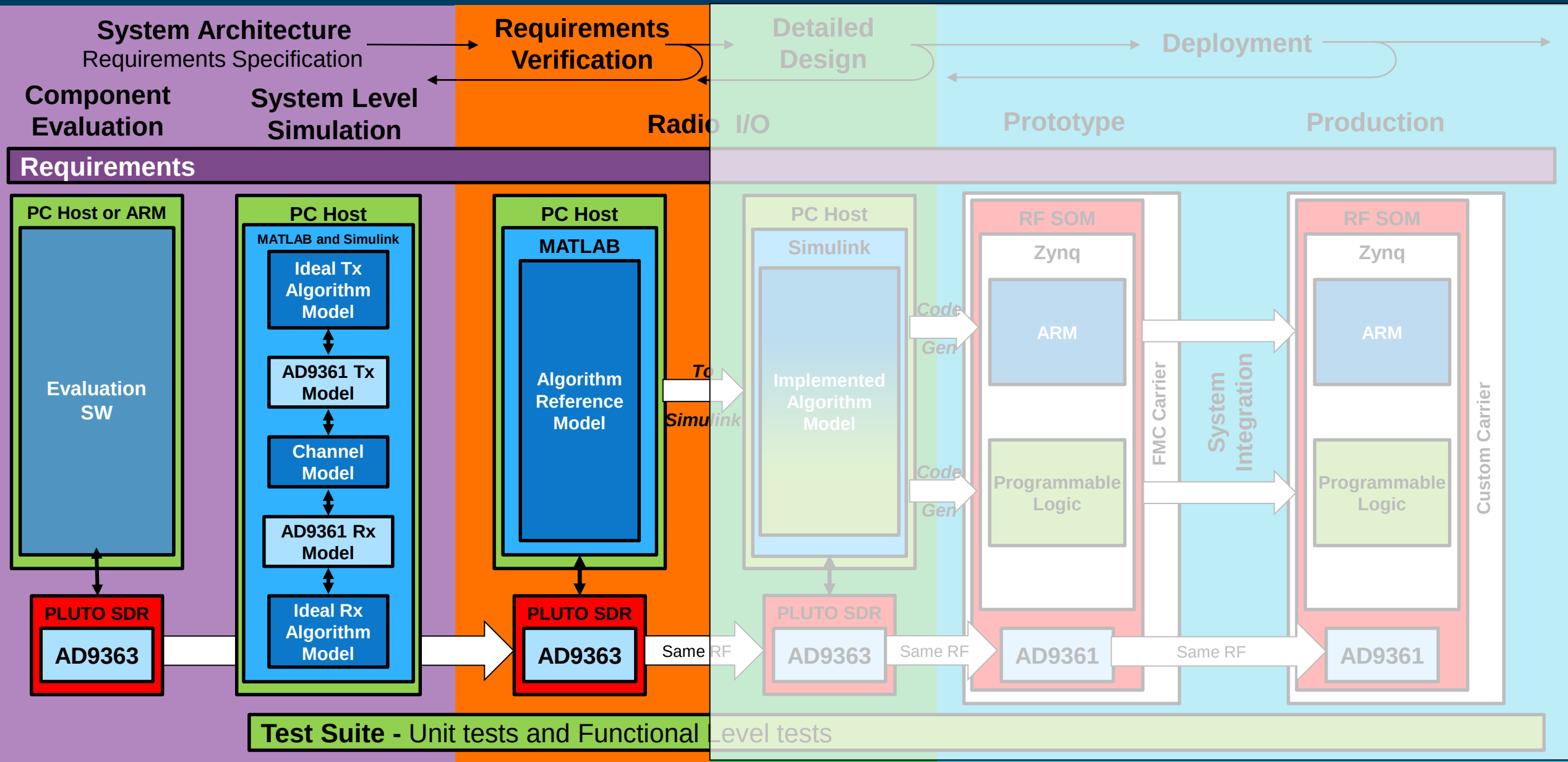
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Software-Defined Radio with RF Systems on Module Workshop

DAY 1 AFTERNOON

Model-Based Design for SDR



Review Morning

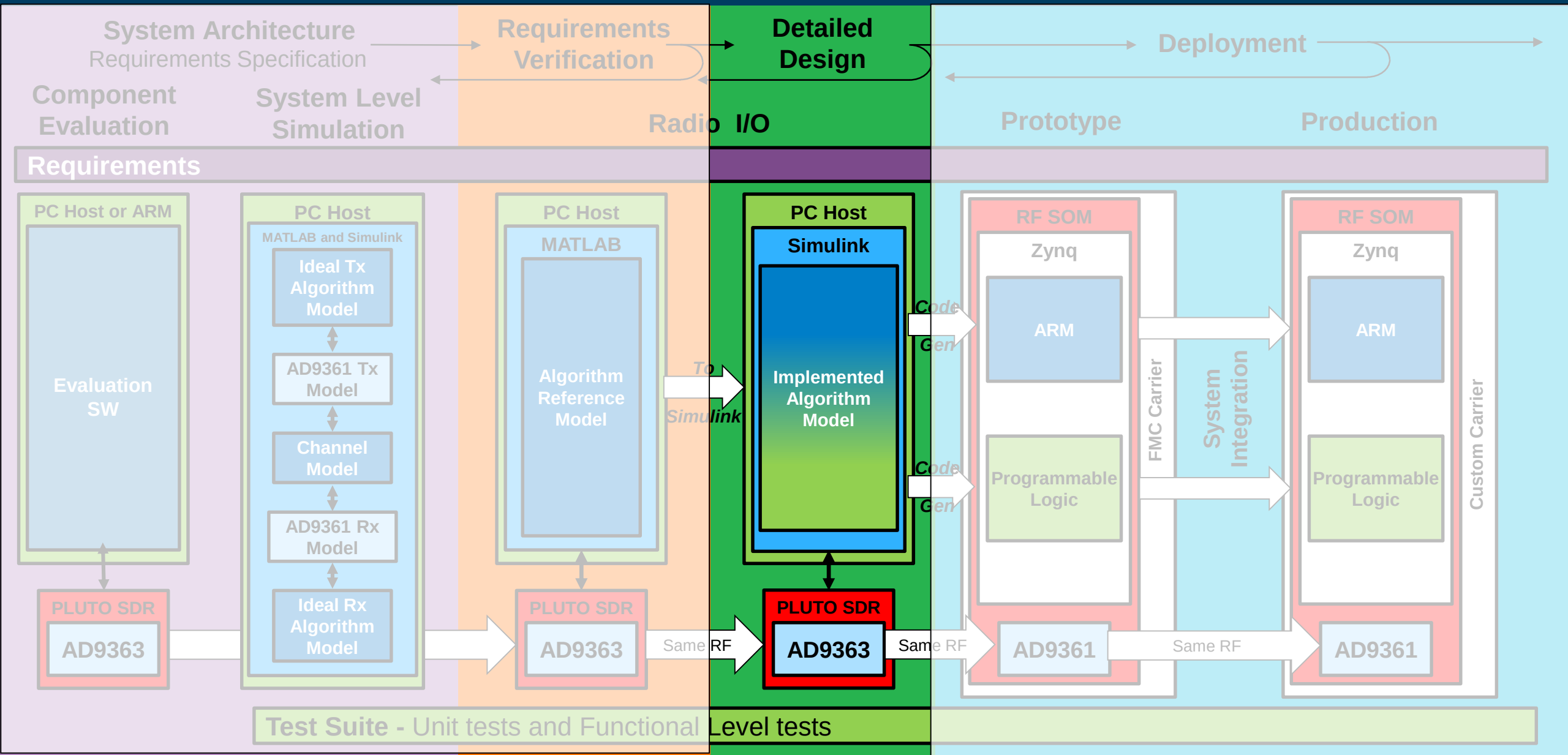
- Design process – high-level abstraction to production deployment
- One Golden Reference, One Set of Requirements, One Test Infrastructure
- Algorithm development and verification can be enhanced through:
  - Utilizing true end-to-end simulation
  - Streaming real data
- Modem overview – P2P link whose PHY will be designed in MATLAB and Simulink

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

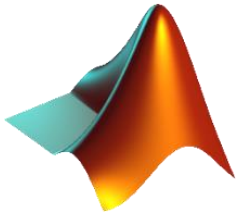
Model-Based Design with Simulink

NOAM LEVINE

Model-Based Design for SDR

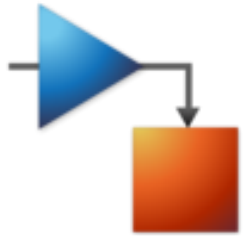


MATLAB and Simulink



MATLAB

- Language of Technical Computing
- Large data sets
- Explore mathematics
- Data visualization



Simulink

- Model-Based Design
- Parallel architectures
- Timing
- Data type propagation
- Code generation

Simulink

- Model-Based Design
- Parallel architectures
- Timing
- Data type propagation
- Code generation

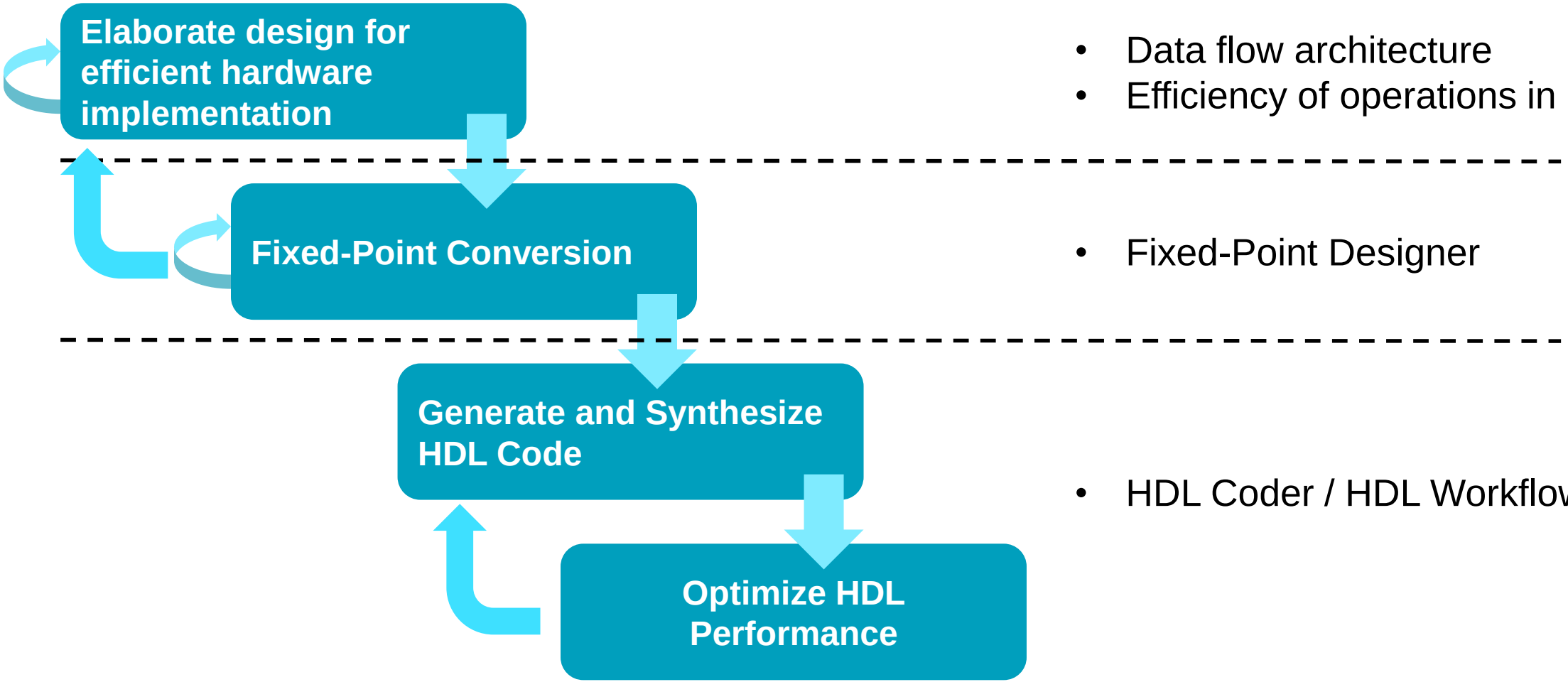
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Why Simulink?

- Concept of time
- Explore algorithm architectures
- Path to code generation and deployment
- Common language across departments / disciplines
  - Model algorithms, components, RF, transmission channels, etc.

Getting the Algorithm to Hardware



- Data flow architecture
- Efficiency of operations in hardware

- Fixed-Point Designer

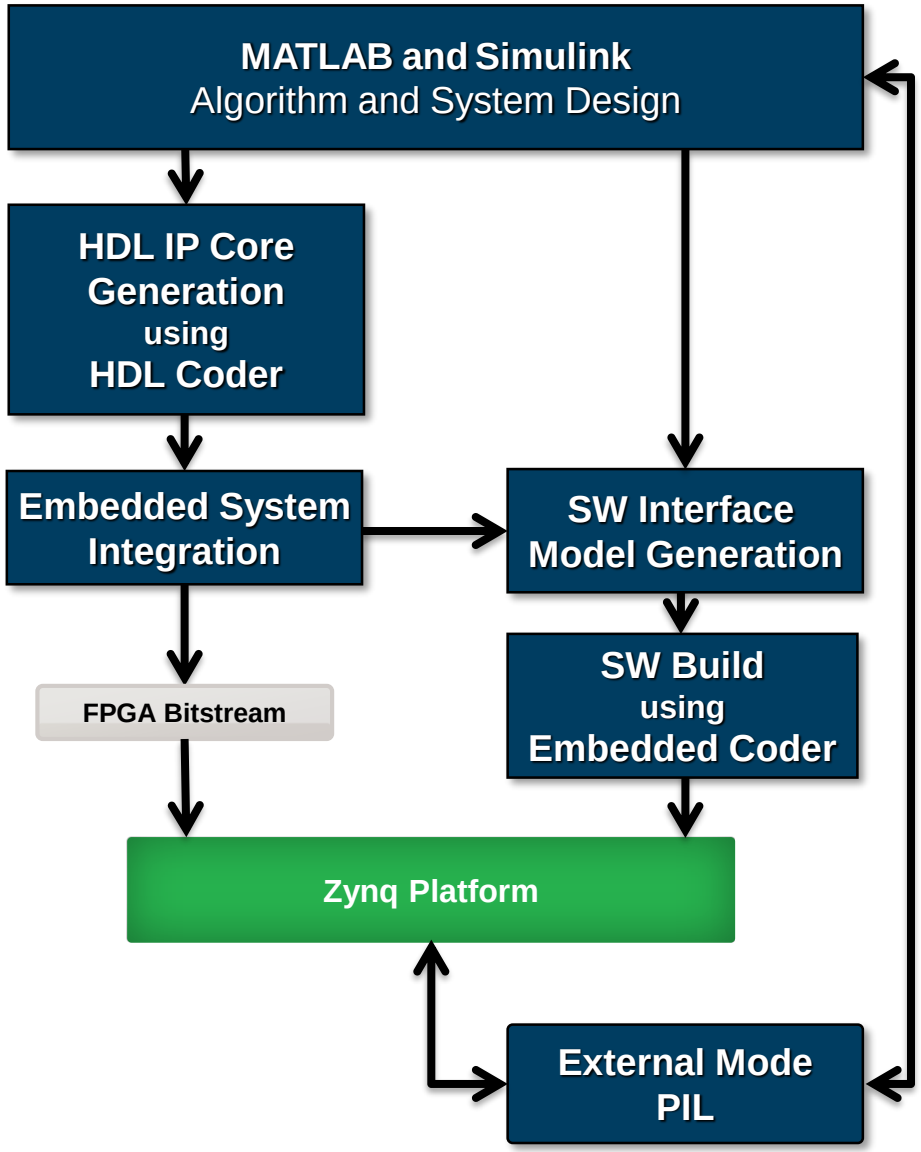
- HDL Coder / HDL Workflow Advisor

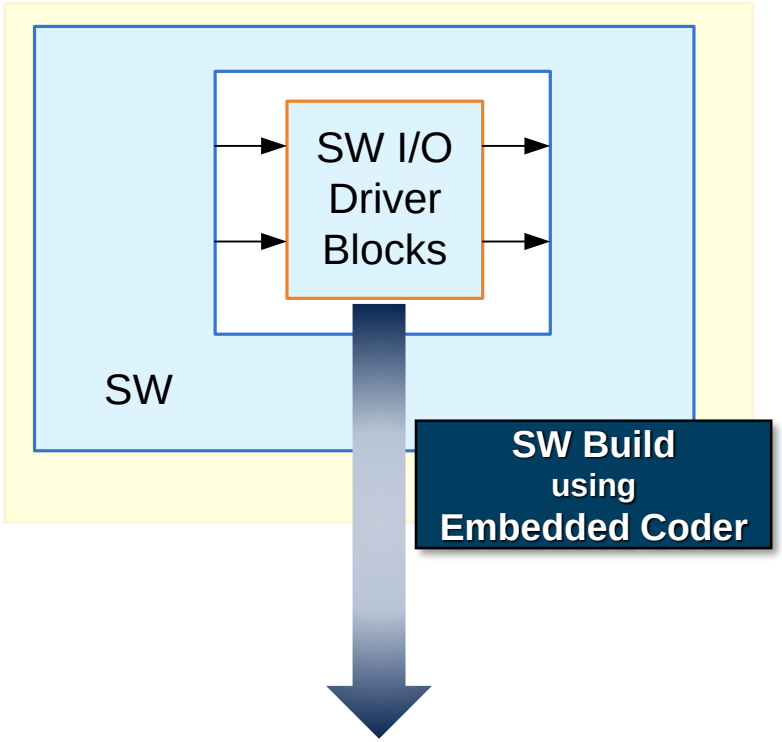
Debugging tools exist at the simulation level and for the deployed code

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Real-Time Execution on Zynq Platform





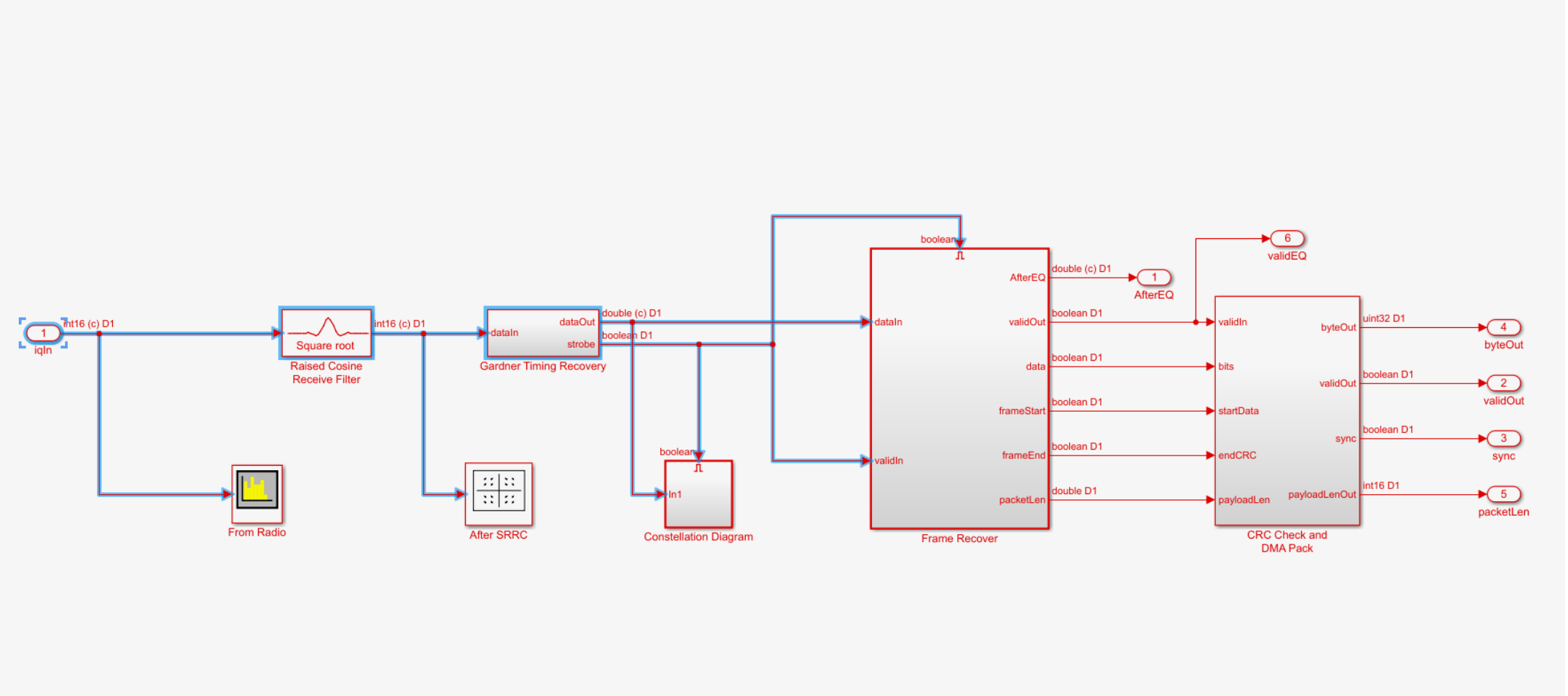
- Real-time Parameter Tuning and Verification
  - External Mode
  - Processor-in-the-loop

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

System Simulation

- Floating-Point Simulink Model

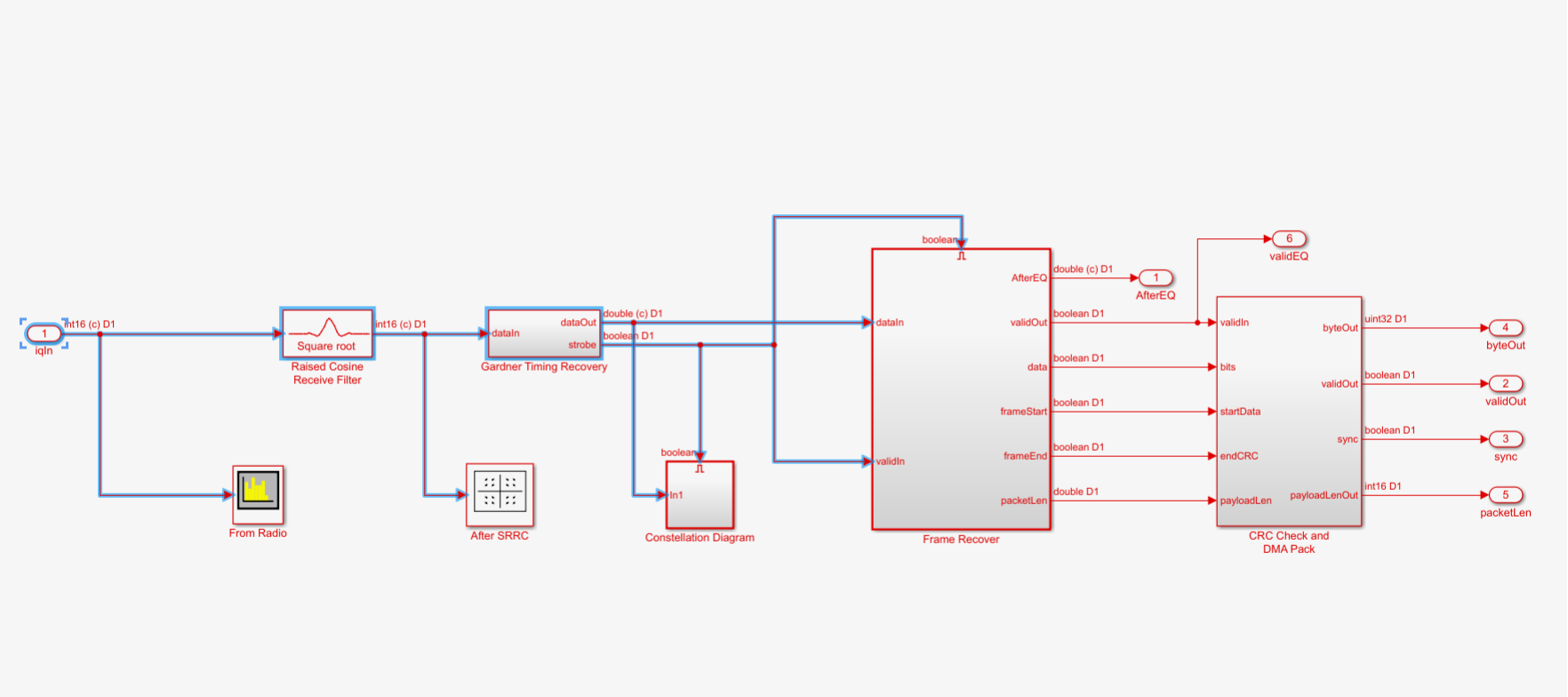


ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Create Floating-Point Reference Model

- Simulink provides a natural environment to model timing
- Reference for subsequent stages
- Compare simulation against MATLAB golden reference



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

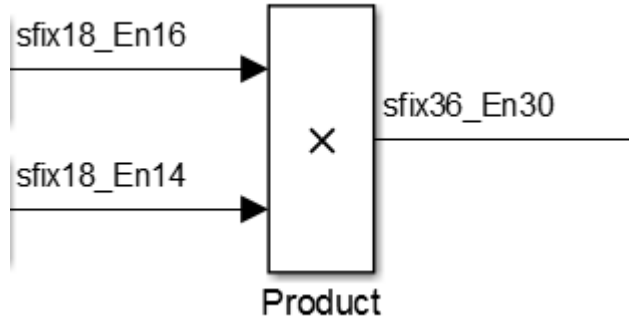
Elaborate Design for Efficient HW Implementation

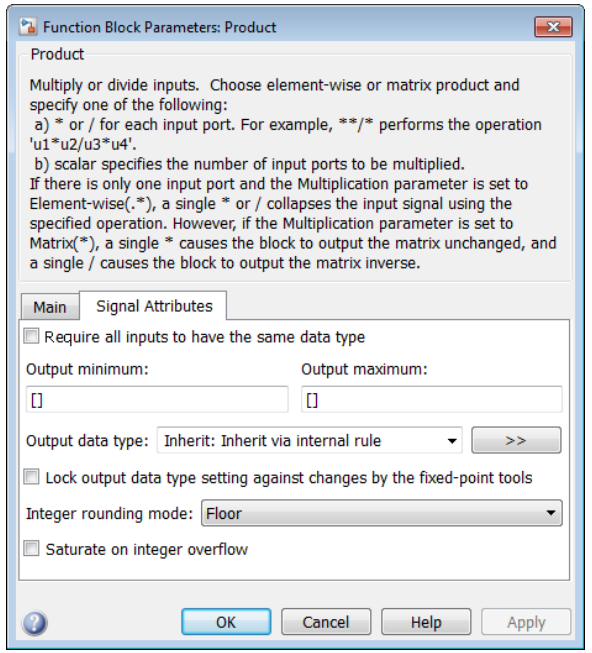
- Consider HDL-friendly architectures
  - Expensive operations, “slow” operations, etc
- Use HDL-compatible blocks
- Pipelining

- Compare against floating / fixed-point reference – **continuous verification**

Convert to Fixed-Point Data Types

- Leverage Simulink fixed-point modeling capabilities
  - Visualize fixed-point data type and overflow warnings
  - “Inherit” full-precision fixed-point automatically
  - Choose rounding & saturation settings
- Determine data dynamic range and propose fixed-point data type using *Fixed-Point Tool*
- Compare fixed-point model against floating-point reference
  - Analyze quantization effects of different word lengths
  - Perform precision & resource trade-offs





ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

Detailed Design Examples

TRAVIS COLLINS, PHD

Model Progression – Frequency Recovery Loop

- MATLAB System object – comm.CarrierSynchronizer

Construction

`S = comm.CarrierSynchronizer` creates a compensator System object, `S`, that compensates for the carrier frequency and phase offsets.

`S = comm.CarrierSynchronizer(Name,Value)` creates a compensator object with each specified property `Name` set to the specified `Value`. You can specify additional name-value pair arguments in any order as (`Name1`,`Value1`,...,`NameN`,`ValueN`).

Properties

`Modulation`	Modulation type Specify the modulation type as `BPSK`, `QPSK`, `OQPSK`, `8PSK`, `QAM`, or `PAM`. The default value is `QAM`. This property is nontunable. This object supports CPM. It has been tested for a CPM signal having 1 sample per symbol and a modulation index of 0.5.
`ModulationPhaseOffset`	Modulation phase offset method Specify the method used to calculate the modulation phase offset as either `Auto` or `Custom`. • `Auto` applies the traditional offset for the specified modulation type. • `Custom` enables the `CustomPhaseOffset` property, which you can use to specify your own phase offset. The default value is `Auto`. This property is tunable.
`CustomPhaseOffset`	Phase offset Specify the phase offset in radians as a real scalar. This property is available only when the `ModulationPhaseOffset` property is set to `Custom`. The default value is `0`. This property is tunable.
`SamplesPerSymbol`	Samples per symbol Specify the number of samples per symbol as a positive integer scalar. The default value is `2`. This property is tunable.
`DampingFactor`	Damping factor of the loop Specify the damping factor of the loop as a positive real finite scalar. The default value is `0.707`. This property is tunable.
`NormalizedLoopBandwidth`	Normalized bandwidth of the loop Specify the normalized loop bandwidth as a real scalar between 0 and 1. The loop bandwidth is normalized by the sample rate of the synchronizer. The default value is `0.01`. This property is tunable.

Modulation	Phase Offset
`BPSK`	0
`QPSK` or `OQPSK`	$\pi/4$
`8PSK`	$\pi/8$
`QAM` or `PAM`	0

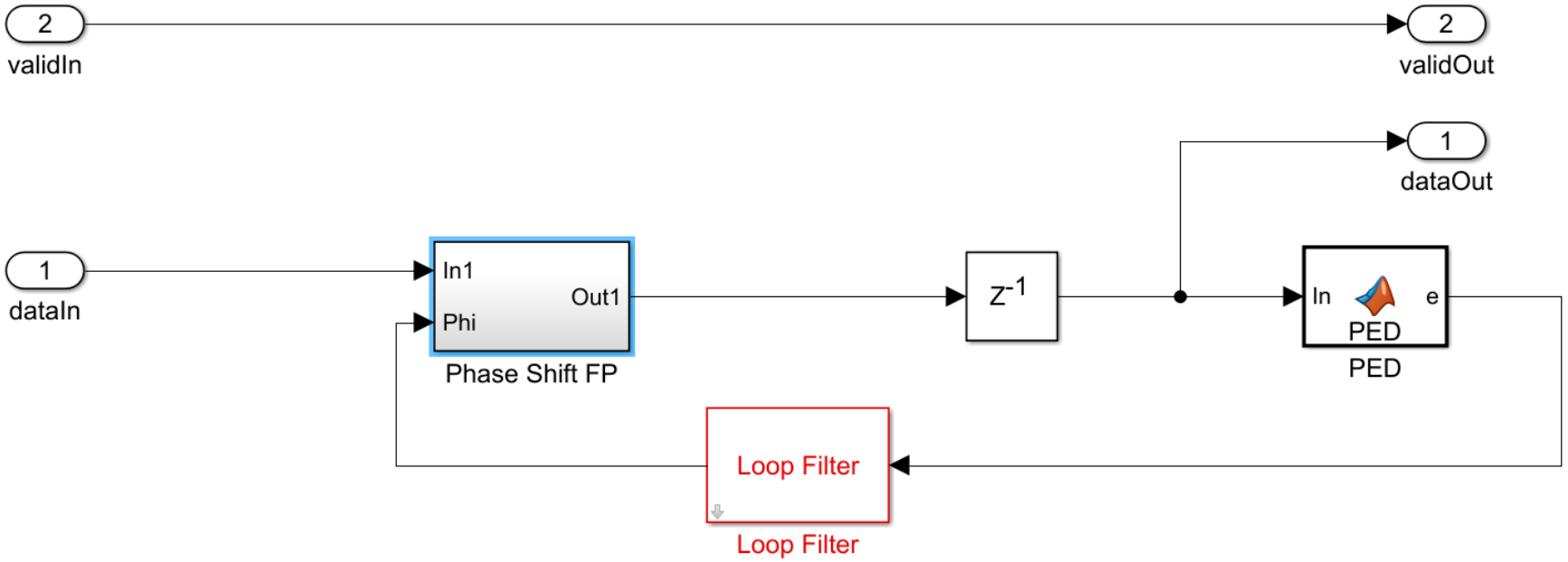
Methods

info	Characteristic information about carrier synchronizer
reset	Reset states of the carrier synchronizer object
step	Compensate for carrier frequency and phase offset

Common to All System Objects	
`clone`	Create System object with same property values
`getNumInputs`	Expected number of inputs to a System object
`getNumOutputs`	Expected number of outputs of a System object
`isLocked`	Check locked states of a System object (logical)
`release`	Allow System object property value changes

Model Progression – Frequency Recovery Loop

- Floating-Point Model



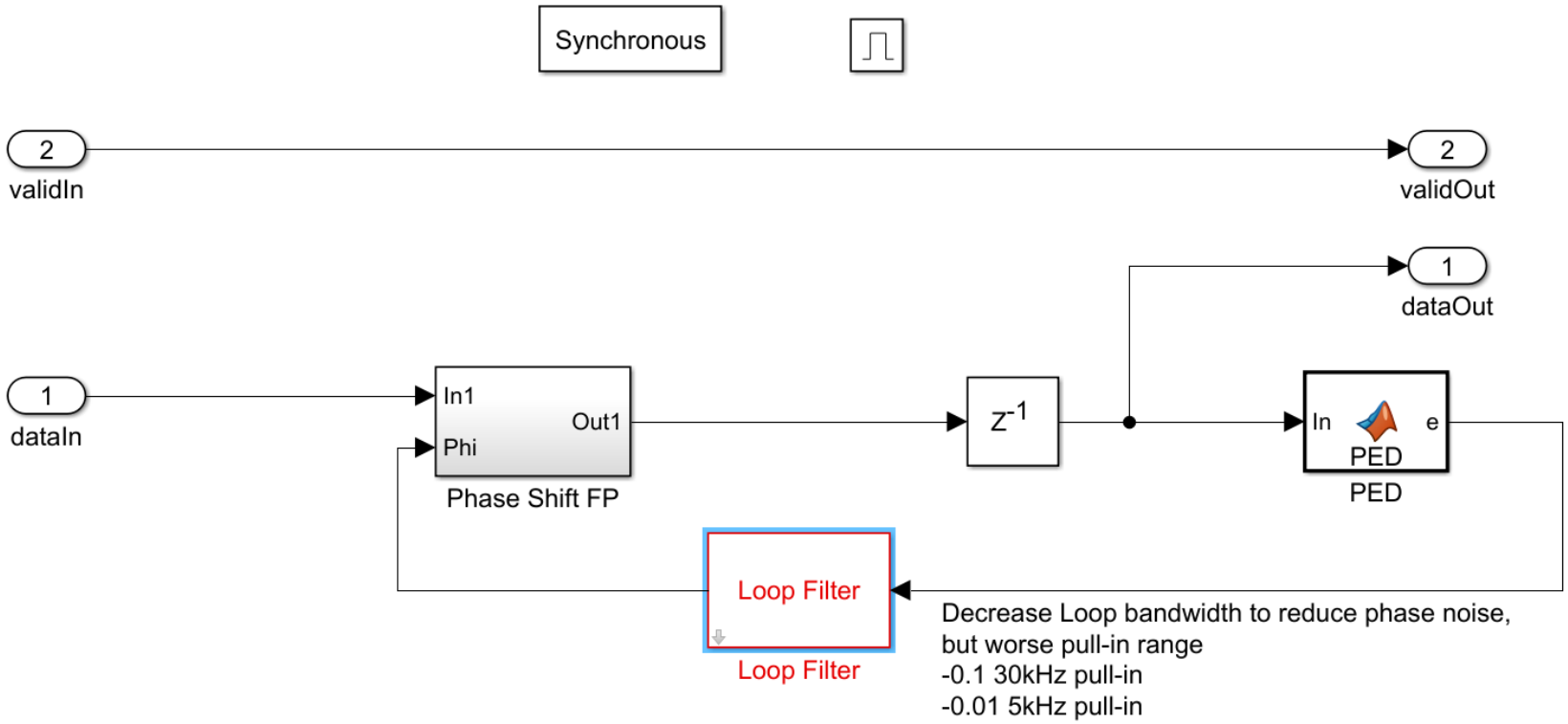
Pull in range (Hz) = 2*pi*sqrt(2)*DampingFactor*NormLoopBandwidth*SampleRate

Max Phase Lock Delay= 1.3/(NormLoopBandwidth*SampleRate)

Frequency Lock Delay = SampleRate*4*NormalizedPullInRange^2)/ (SampleRate*NormLoopBandwidth)^3;

Model Progression – Frequency Recovery Loop

- Fixed-Point Model



Pull in range (Hz) = 2*pi*sqrt(2)*DampingFactor*NormLoopBandwidth*SampleRate

Max Phase Lock Delay= 1.3/(NormLoopBandwidth*SampleRate)

Frequency Lock Delay = SampleRate*4*NormalizedPullInRange^2)/ (SampleRate*NormLoopBandwidth)^3;

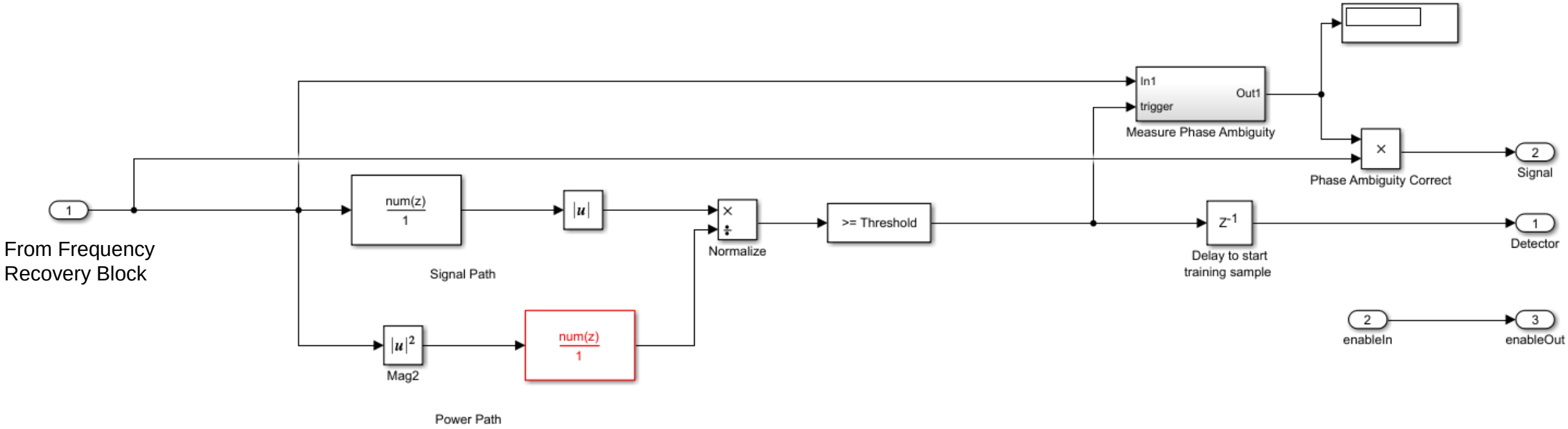
Model Progression – Packet Detector

- Floating-point MATLAB
  - FindFrameStart.m

```
function [frame,ind] = FindFrameStart(signal, xPreamble)

preambleLength = length(xPreamble);

% Estimate start of frame
eng = mean(abs(signal).^2); % Mean power
cor = abs(filter(xPreamble(end:-1:1).',1,signal));
% look in first half only
cor = cor(1:floor(length(cor)/2));
[val,ind] = max(cor);

% The max should be at least X times the mean
if (val/eng) > 4 %(Larger makes more selective)
    % Correct to edge of preamble
    ind = ind - preambleLength;
    frame = signal(ind+1:end); % Includes preamble
    % Get orientation
    phaseEst = round(angle(mean(conj(xPreamble) .*
frame(1:preambleLength)))*2/pi)/2*pi;
    % Compensating for the phase offset
    frame = frame .* exp(-1i*phaseEst);
    %[frame(1:10),xPreamble(1:10)]
else
    frame = [];
end

end
```

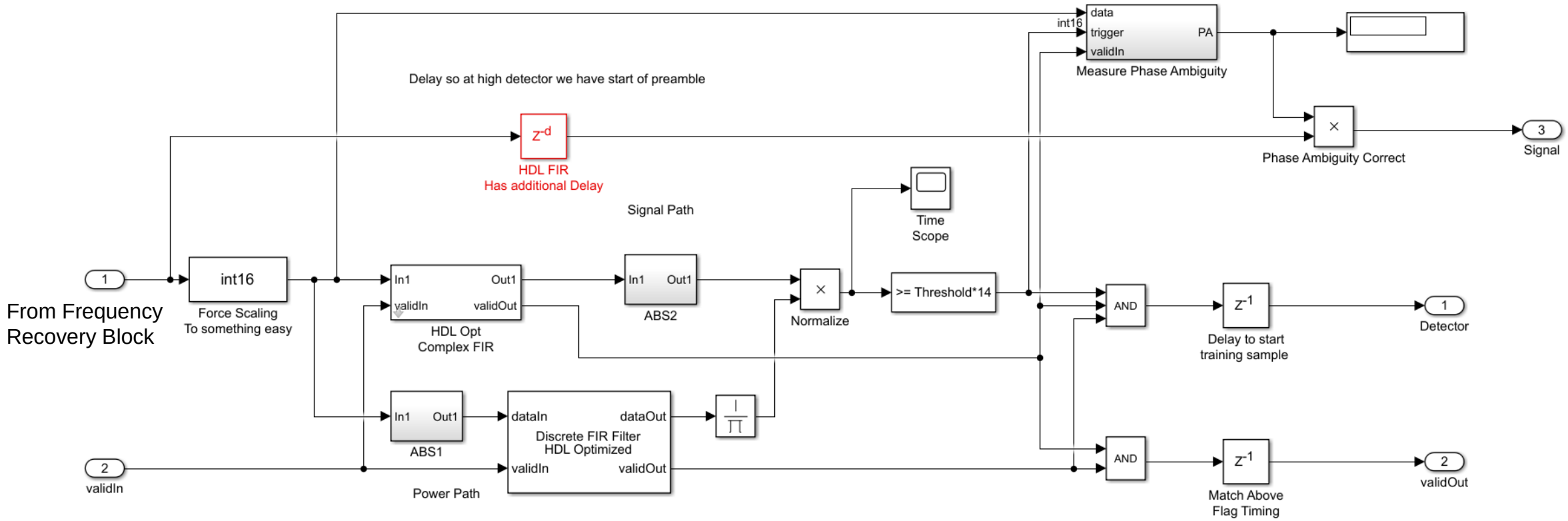
Model Progression – Packet Detector

- Floating Point Simulink Model
  - Barker Locate

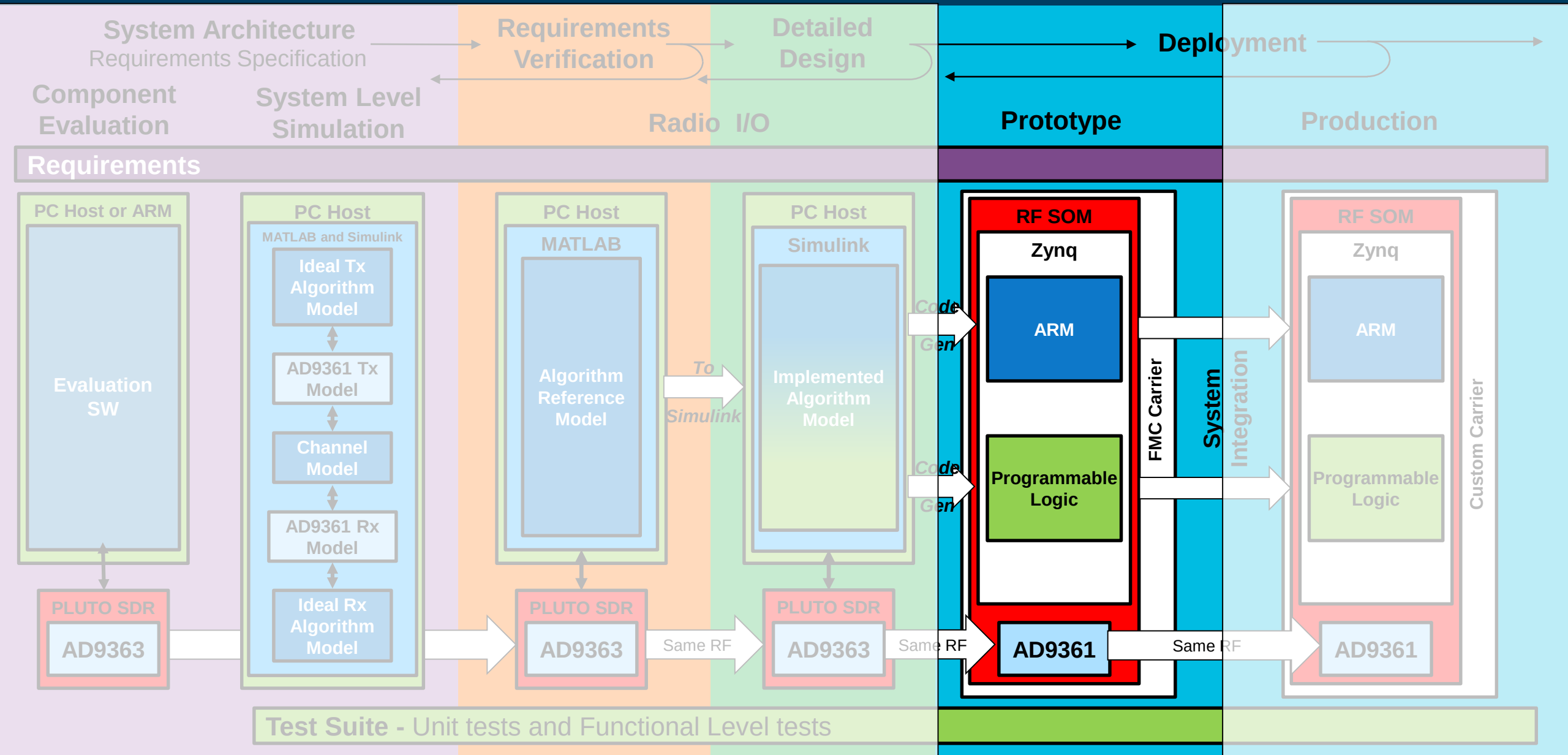


Model Progression – Packet Detector

- Fixed-Point Simulink Model (HDL-ready)
  - Barker Locate



Model-Based Design for SDR



Lab 2

FIXED POINT CONVERSION

TRAVIS COLLINS, PHD

Lab 2: Fixed-Point Conversion

- Task:
  - Example walkthrough of manual Fixed-Point conversion of frequency recovery

- Topics of note:
  - Fixed-Point data types
  - Simulink data flow and block manipulation
  - Debugging in Simulink

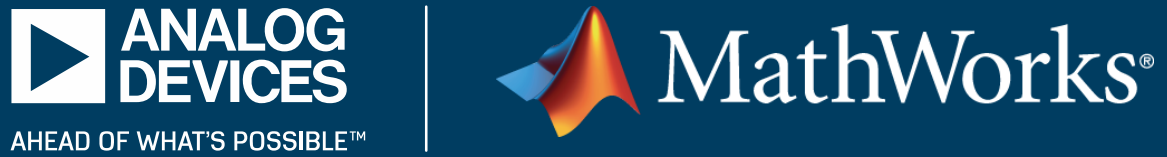
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

MathWorks®

Day 1 PM Review

Key points

- MATLAB
  - Algorithm development
  - Explore large data sets
  - Visualizations
- Simulink
  - Time-based simulation
  - Gateway to higher-level capabilities
    - Fixed-Point conversion
    - Code generation
- Example Modem Design:
  - MATLAB and Simulink focus on PHY+MAC layer development
  - Workflow process includes three designs:
    - MATLAB Floating-Point Reference Design
    - Simulink Floating-Point Scalar Model
    - Simulink Fixed-Point Scalar Model
  - Testing harness maintains requirement consistency across designs
- MATLAB and Simulink have many tools and features to help with fixed-point conversion process
- Testing early with real data saves time and money



Dinner Break