

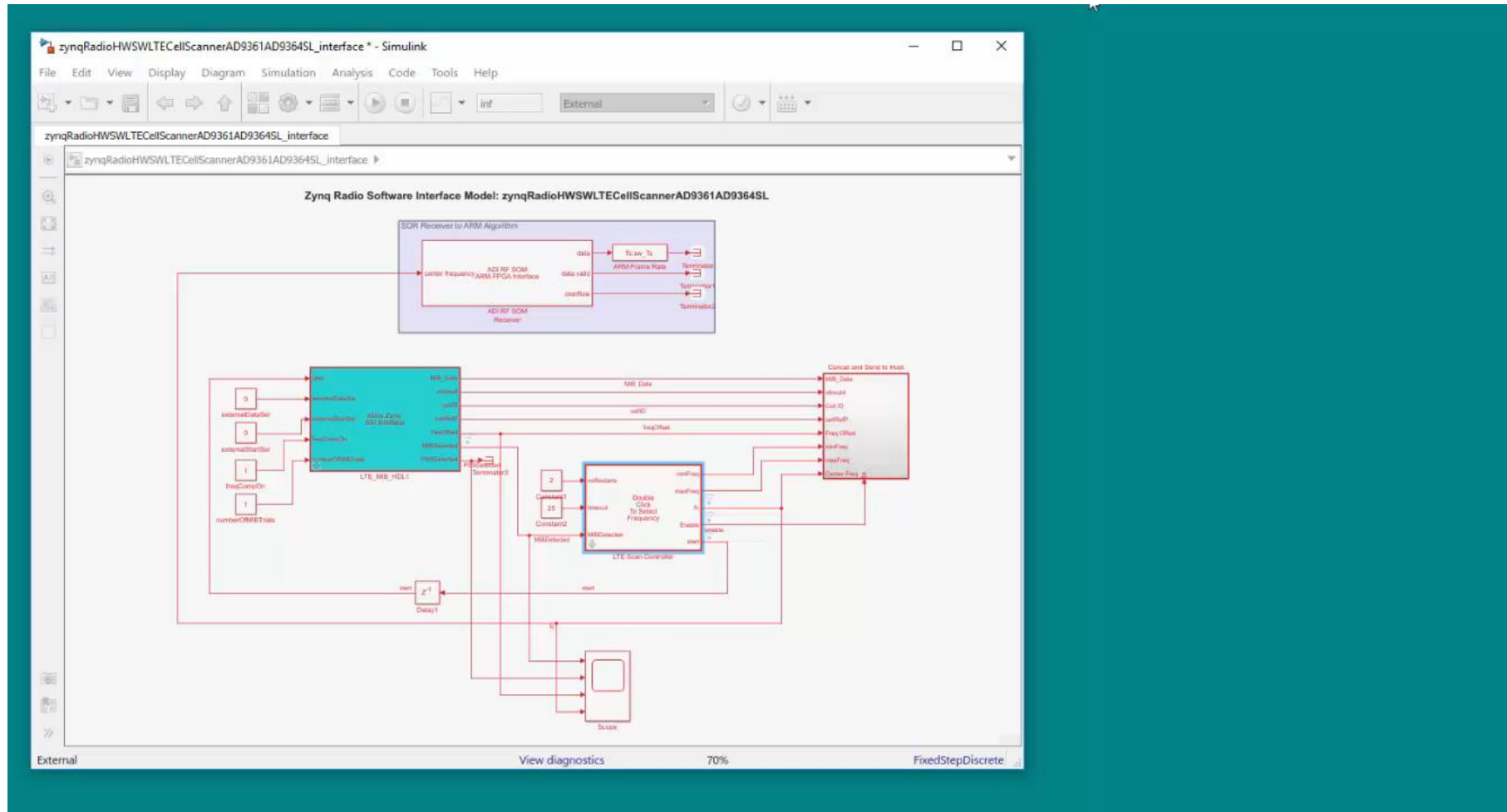
The background features a dark blue field on the left with white text. On the right, there are geometric shapes: a grey trapezoid at the top with white wave-like lines, a blue trapezoid below it, and a 3D wireframe plot with a color gradient from yellow to blue. Faint circuit board patterns are visible in the bottom right.

# MATLAB EXPO 2017

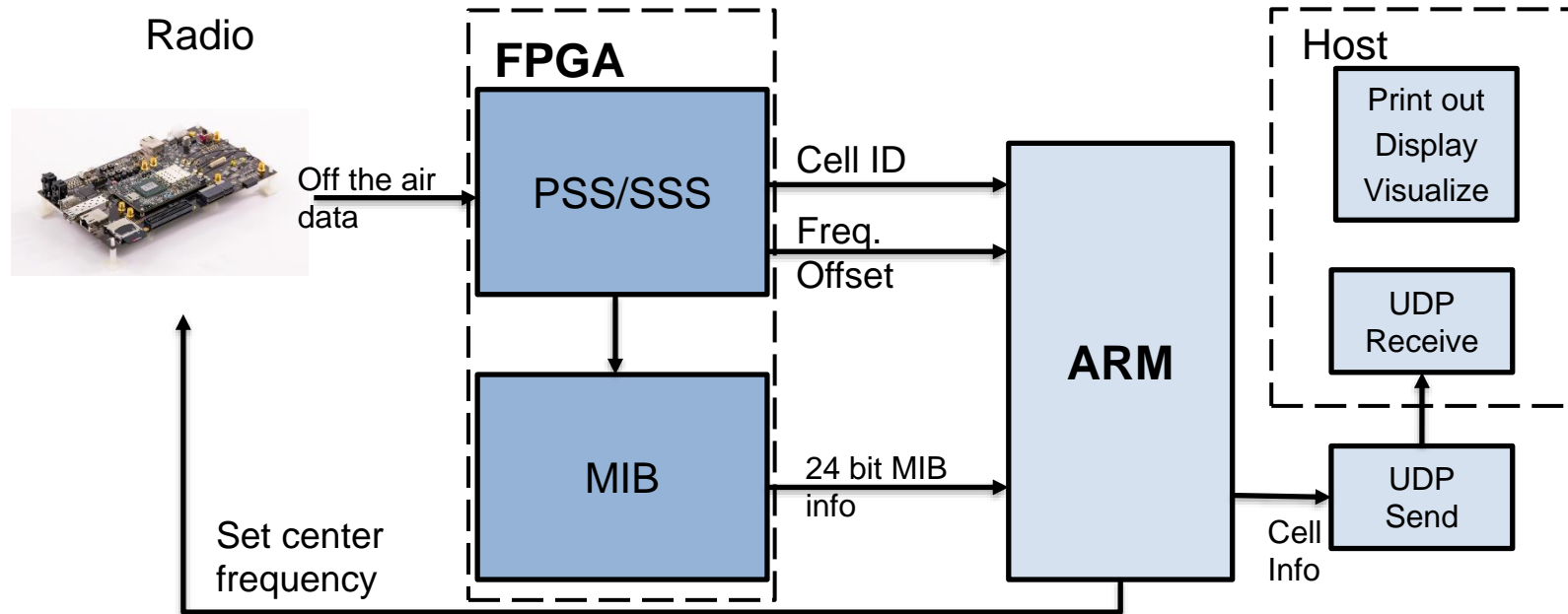
Simulation, prototyping and  
verification of standards-based  
wireless communications

Colin McGuire, Neil MacEwen

# Real Time LTE Cell Scanner with MATLAB and Simulink

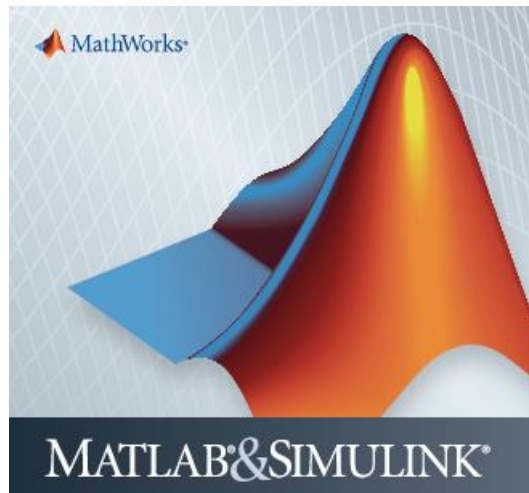


# Real time LTE Frequency Scanner

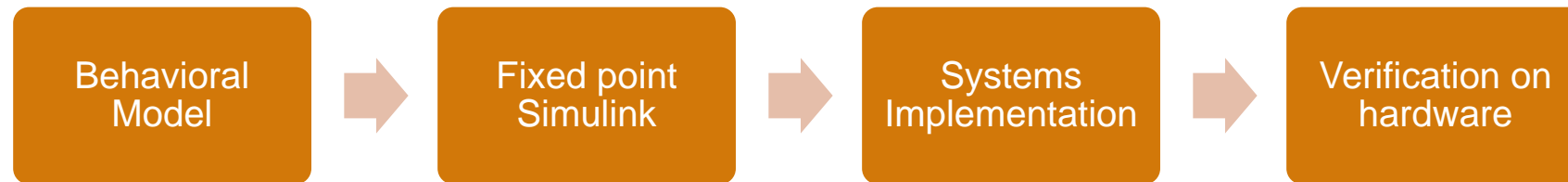


# From Design to Prototype

- A common design environment across multiple teams
  - Systems Engineers, RF Engineers, Algorithm Developers, HDL Engineers
- Target off-the-shelf hardware for prototype development



# From Design to Prototype



# Modeling Wireless Standards with MATLAB & SIMULINK

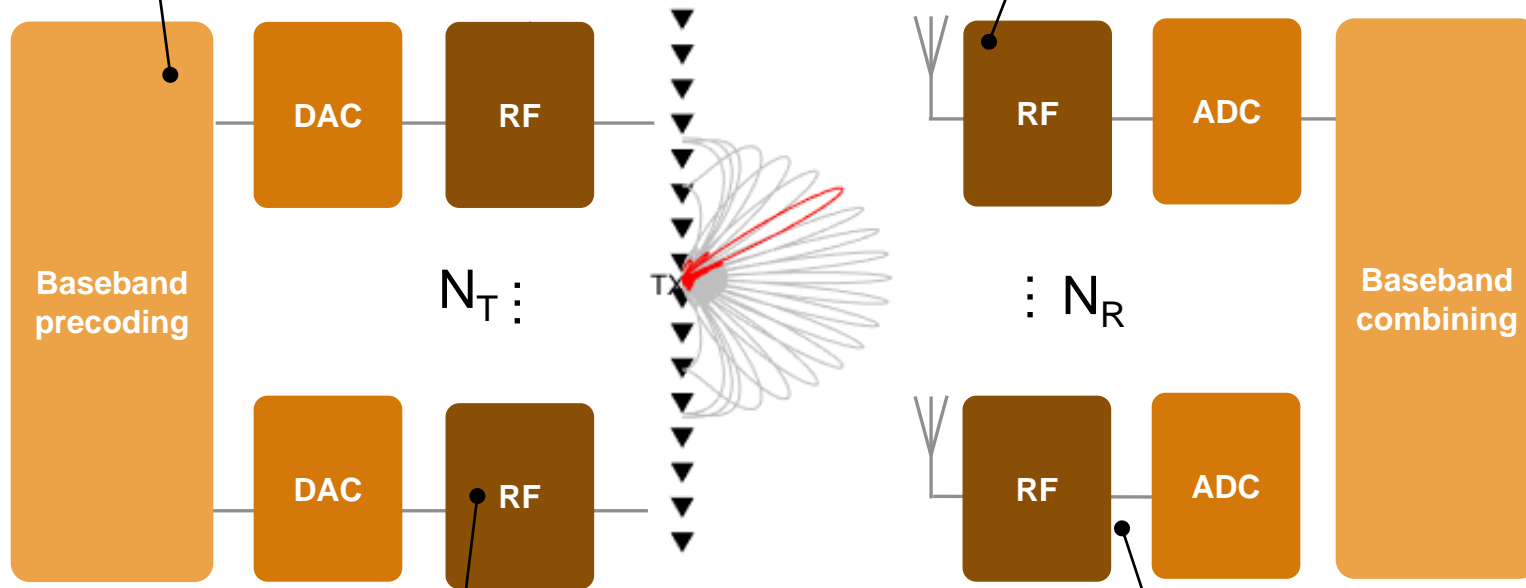
# Wireless Modeling Challenges

## Baseband DSP development

- Is my implementation correct?
- How can I evaluate link performance with my algorithm?
- 5G challenges, e.g. out of band emission..

## Antenna array design and evaluation

- Element coupling
- Edge effects
- Imperfections



## Explore beamforming trade-offs

- Baseband, analogue or hybrid beamforming?
- Simulate capabilities and limitations
- Trade-off ADCs vs RF components

## Investigate the impact of RF impairments

- Frequency dependency
- Non-linearities
- Mismatches and coupling

**Rx Constellation**

Quadrature Amplitude


In-phase Amplitude

**Wireless Digital Video Broadcasting with RF Beamforming**

Model a digital video broadcasting system which includes phased array antennas. The baseband transmitter, receiver and channel are realized

[Open Script](#)

# Modeling Wireless Standards with MATLAB & SIMULINK



LTE



5G




WLAN



UMTS/cdma2000



ZigBee



NFC



# From Design to Prototype

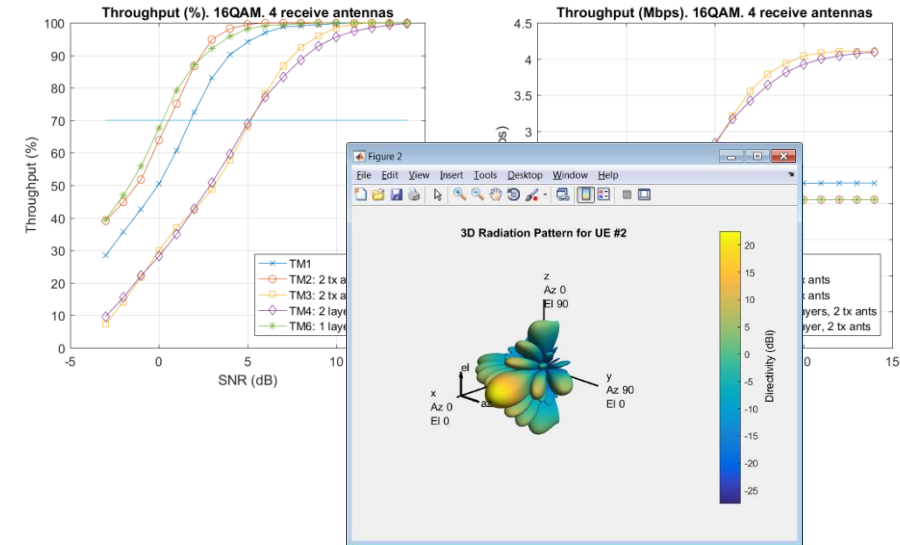
How can I evaluate the performance of my algorithm?

Behavioral Model

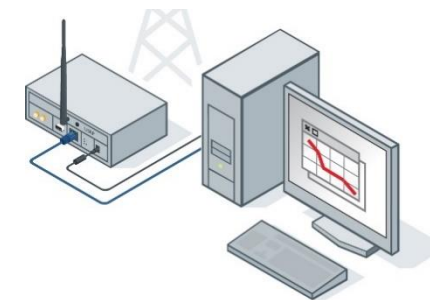
Fixed point Simulink

Is my implementation correct?

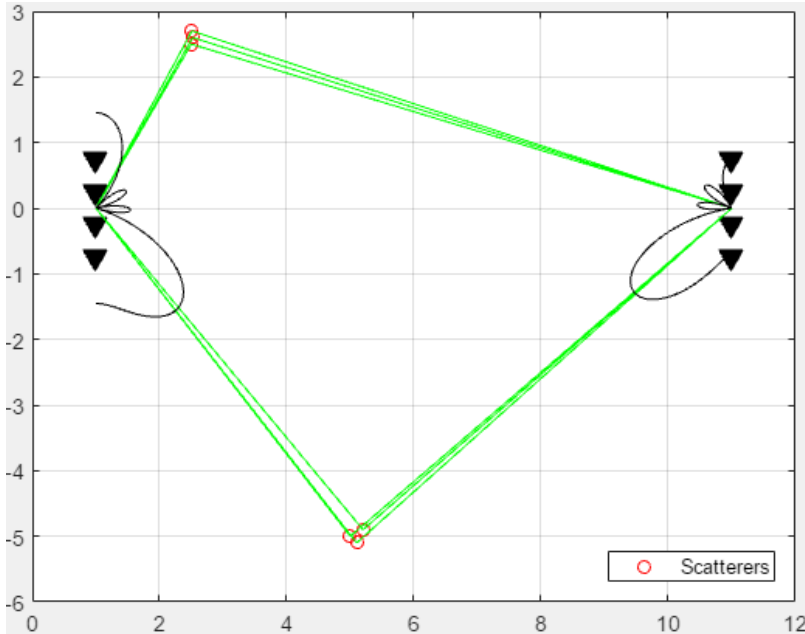
## System simulation



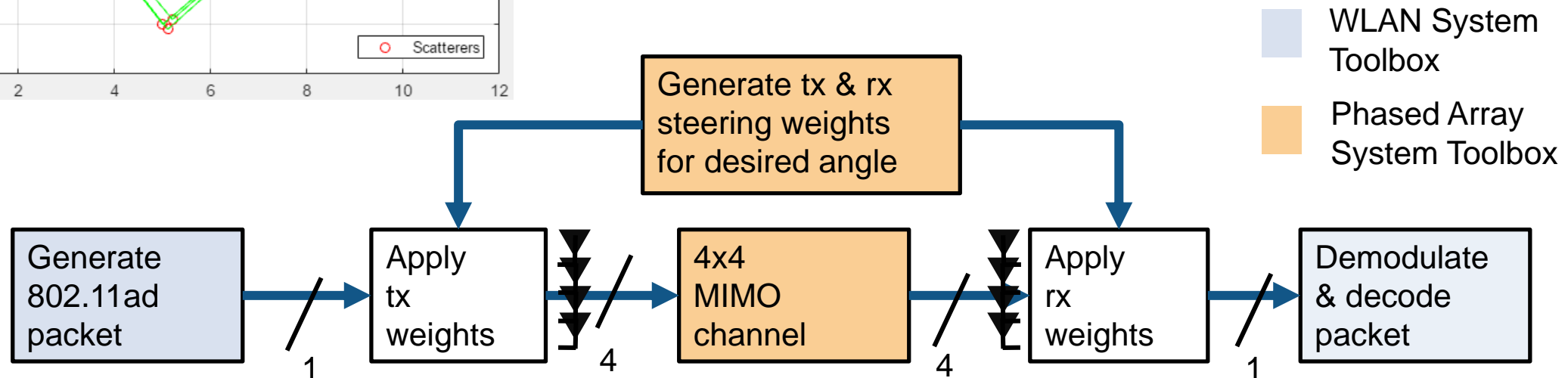
## Working with real signals



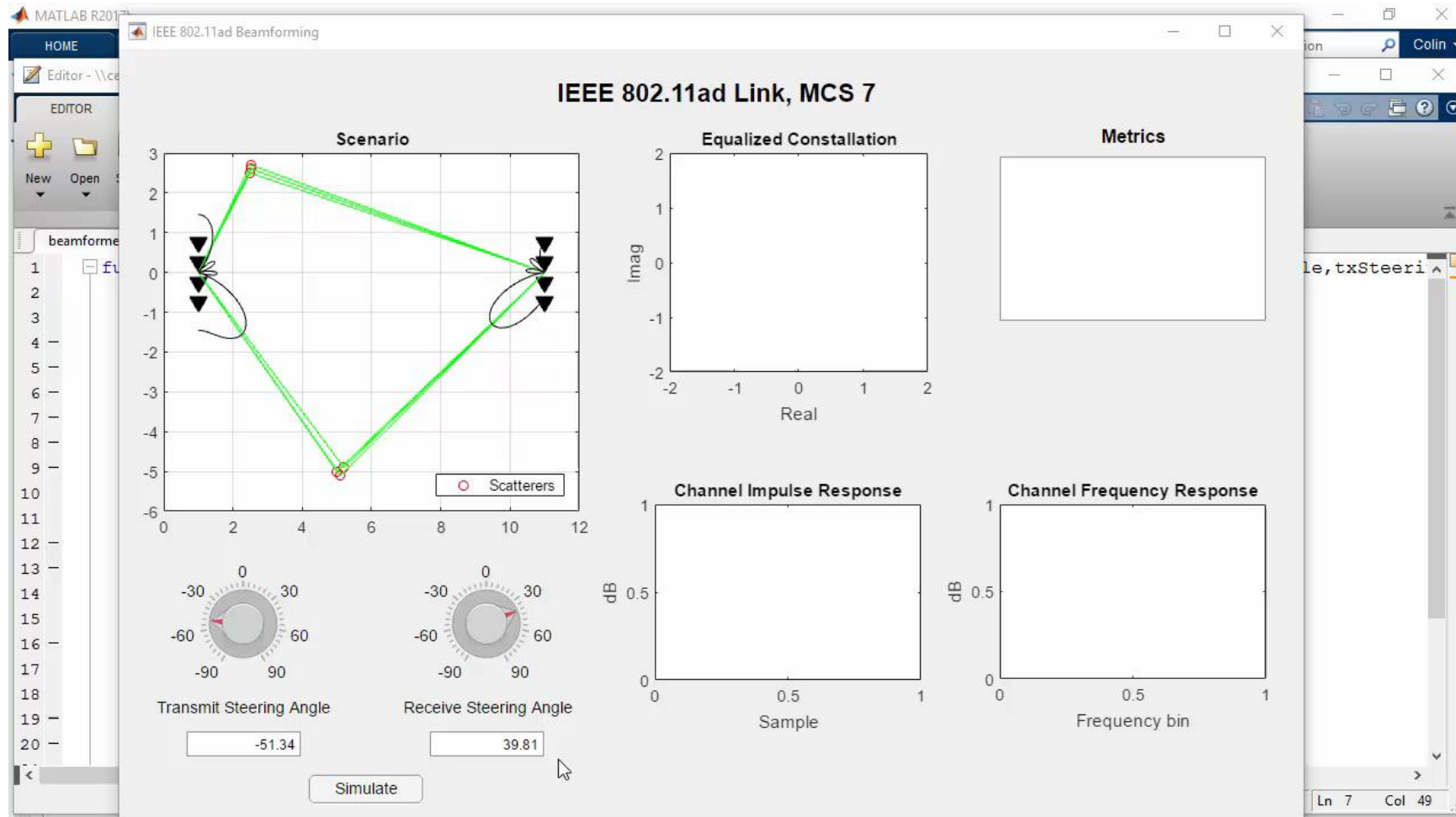
# Demo: Modeling 802.11ad Beamforming



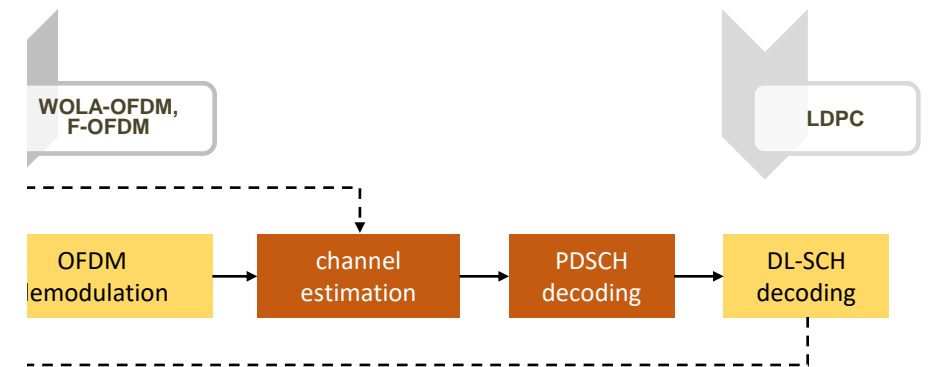
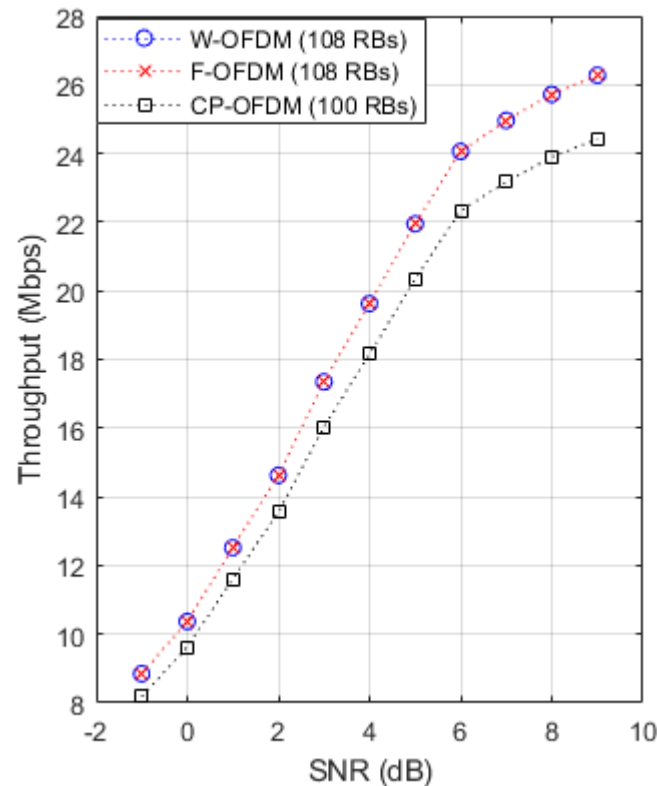
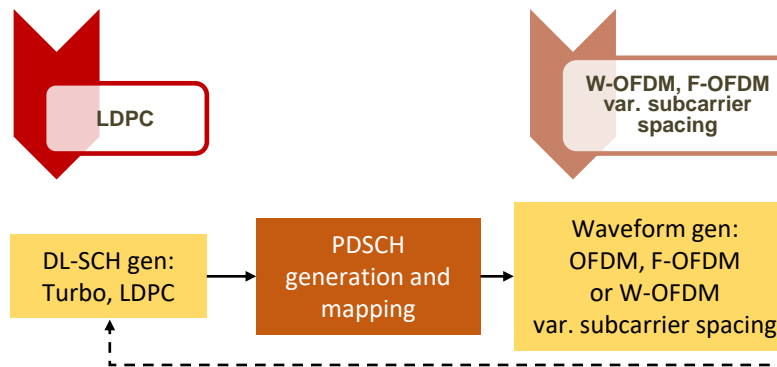
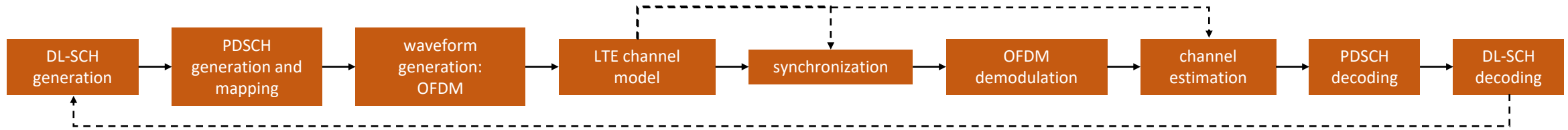
- Uniform linear array of 4 elements at transmitter and receiver
- MIMO channel with 6 scatterers
- PER and EVM for 802.11ad link



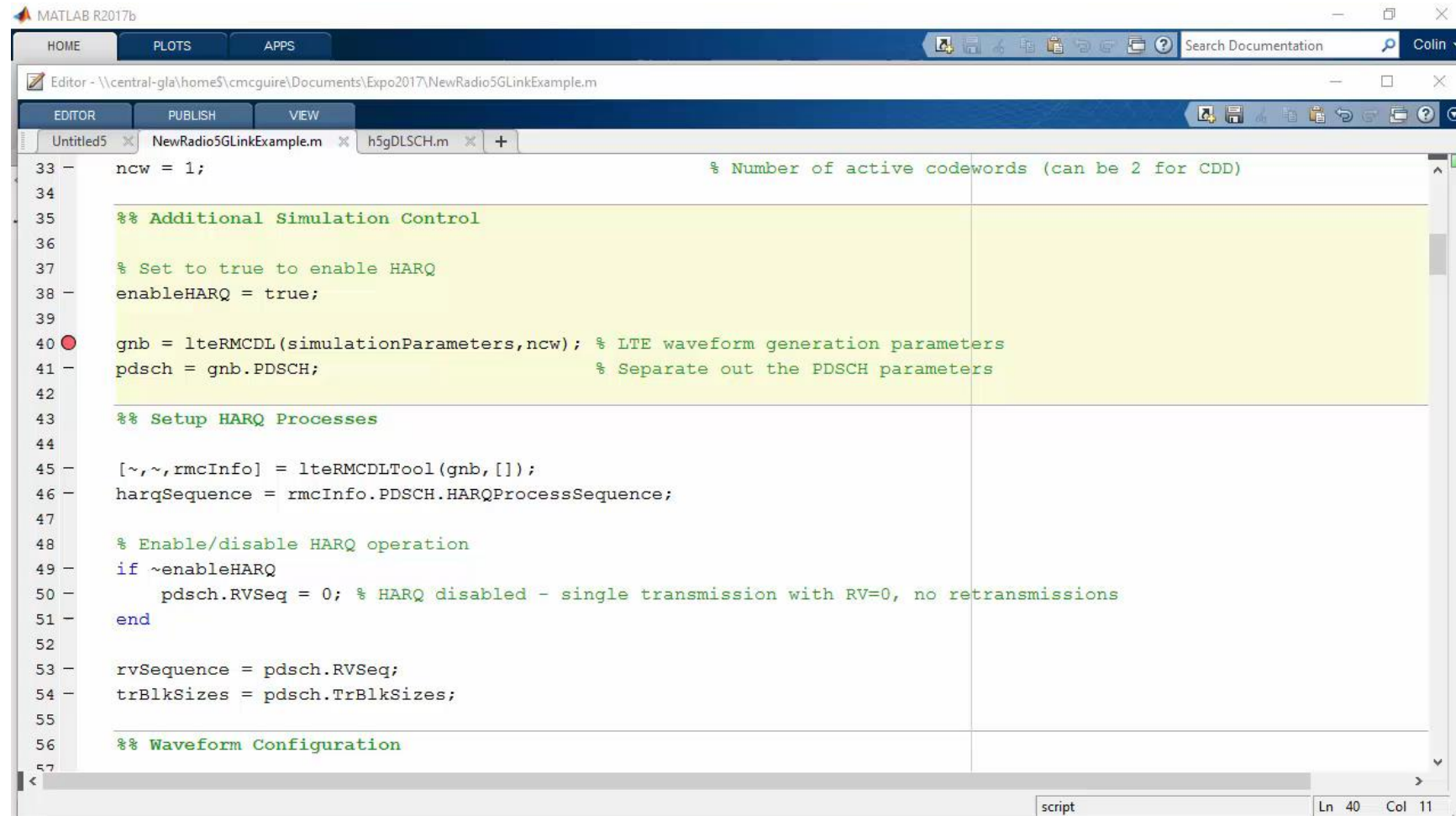
# Demo: Modeling 802.11ad Beamforming



# Extending standards... LTE to 5G



# Demo – Extending LTE for 5G link level simulation



The image shows a MATLAB R2017b editor window with the following code:

```
33 - ncw = 1; % Number of active codewords (can be 2 for CDD)
34
35 %% Additional Simulation Control
36
37 % Set to true to enable HARQ
38 - enableHARQ = true;
39
40 ● gnb = lteRMCDL(simulationParameters,ncw); % LTE waveform generation parameters
41 - pdsch = gnb.PDSCH; % Separate out the PDSCH parameters
42
43 %% Setup HARQ Processes
44
45 - [~,~,rmcInfo] = lteRMCDLTool(gnb, []);
46 - harqSequence = rmcInfo.PDSCH.HARQProcessSequence;
47
48 % Enable/disable HARQ operation
49 - if ~enableHARQ
50 -     pdsch.RVSeq = 0; % HARQ disabled - single transmission with RV=0, no retransmissions
51 - end
52
53 - rvSequence = pdsch.RVSeq;
54 - trBlkSizes = pdsch.TrBlkSizes;
55
56 %% Waveform Configuration
57
```

# From Design to Prototype

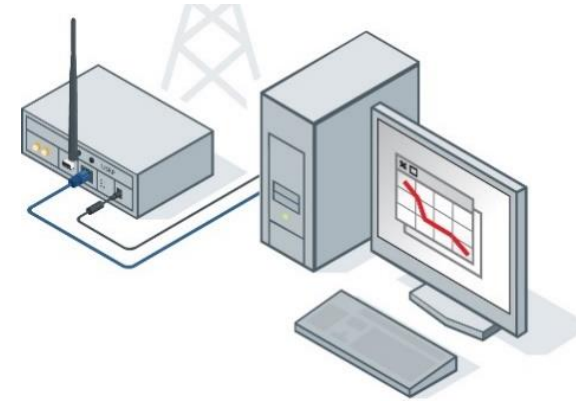
How can I evaluate the performance of my algorithm?

Behavioral Model

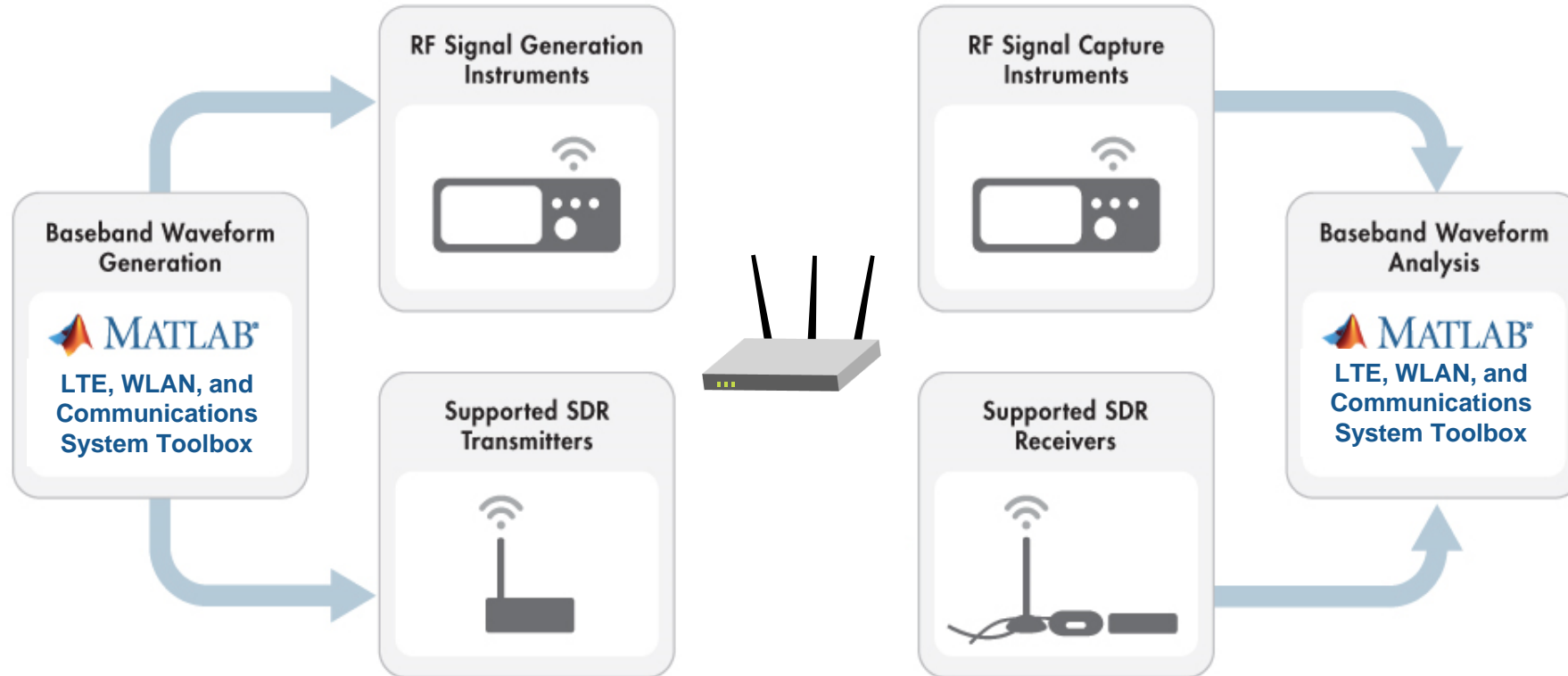
Fixed point Simulink

Is my implementation correct?

## Working with real signals



# Working with Real Signals... Beyond Simulation



# Supported Hardware for Radio Connectivity



	<p><b>Signal Generator and Analyser</b></p> <p>Keysight, R&amp;S, NI, Tektronix, ... High quality RF front end Wide frequency range, high bandwidth</p>
	<p><b>SDR</b></p> <p>USRP, PLUTO, Zynq, ... Customizable RF front end Sizable FPGA for targeting designs</p>
	<p><b>Ultra low-cost SDR</b></p> <p>RTL-SDR, ... Low bandwidth Receive only</p>

**Instrument Control Toolbox**

**SDR Hardware Support Package**

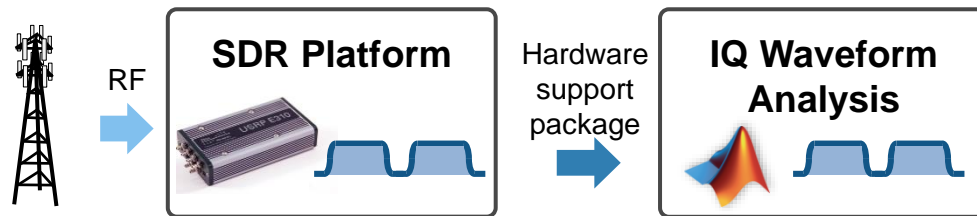


# Prototype with Real Signals

- SDR platforms can be used as low-cost RF interface
- Transmit repeat capability allows USRP E310 or Zynq to be used as an RF signal generator



- Capture a burst of IQ and process in MATLAB



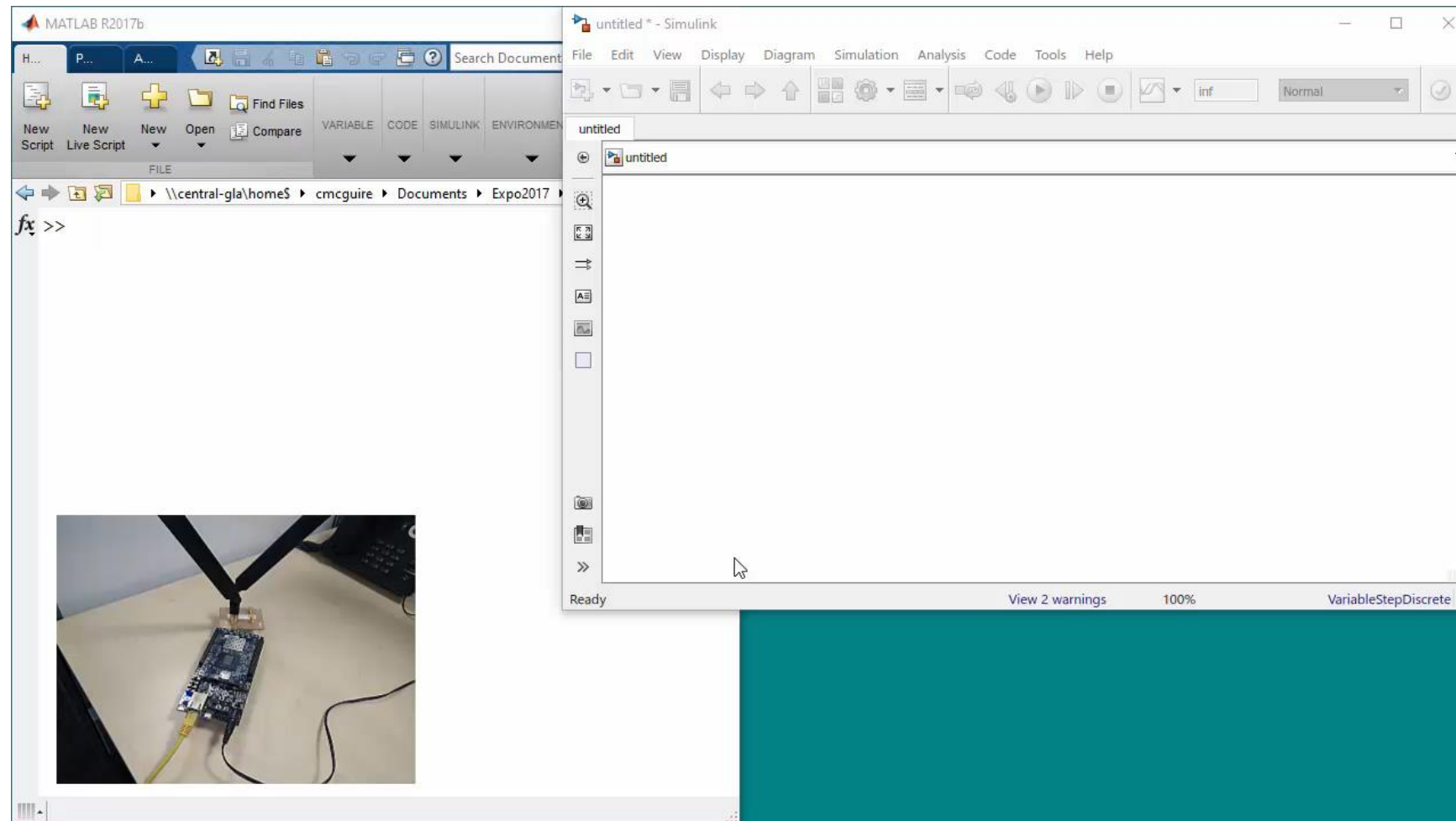
**LTE Cell Search, MIB and SIB1 Recovery with Two Antennas**  
 Uses both channels of USRP® B210, X300 or X310 to receive an LTE downlink signal. The LTE System Toolbox™ is used to

[Open Script](#)

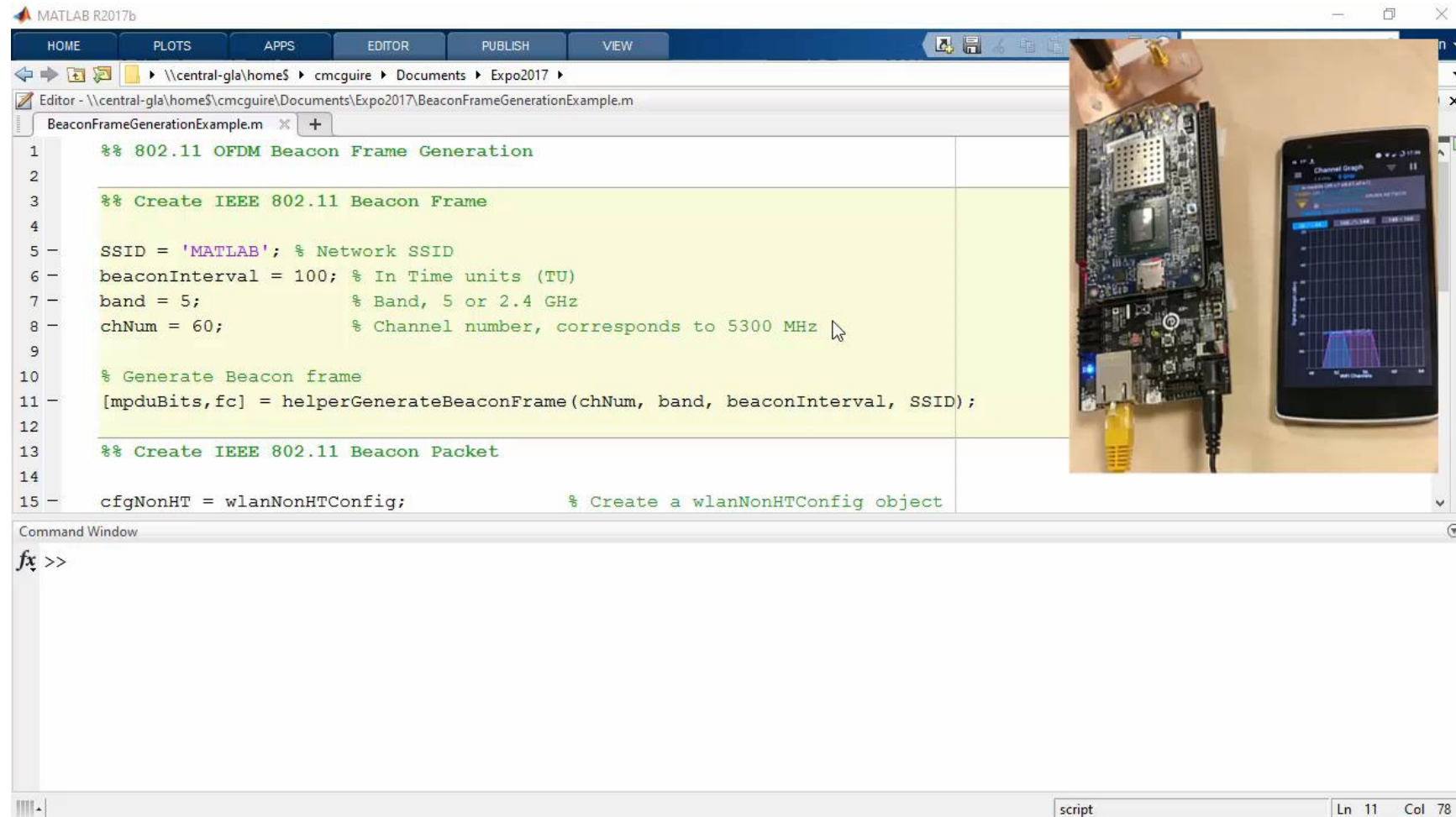
**802.11a Transmission and Reception Using Analog Devices AD9361/AD9364**  
 Transmit and receive WLAN packets on a single SDR platform, using Xilinx® Zynq-Based Radio Support Package with MATLAB® and WLAN

[Open Script](#)

# Demo: SDR as a low-cost RF interface



# Demo: Generating WLAN Beacons



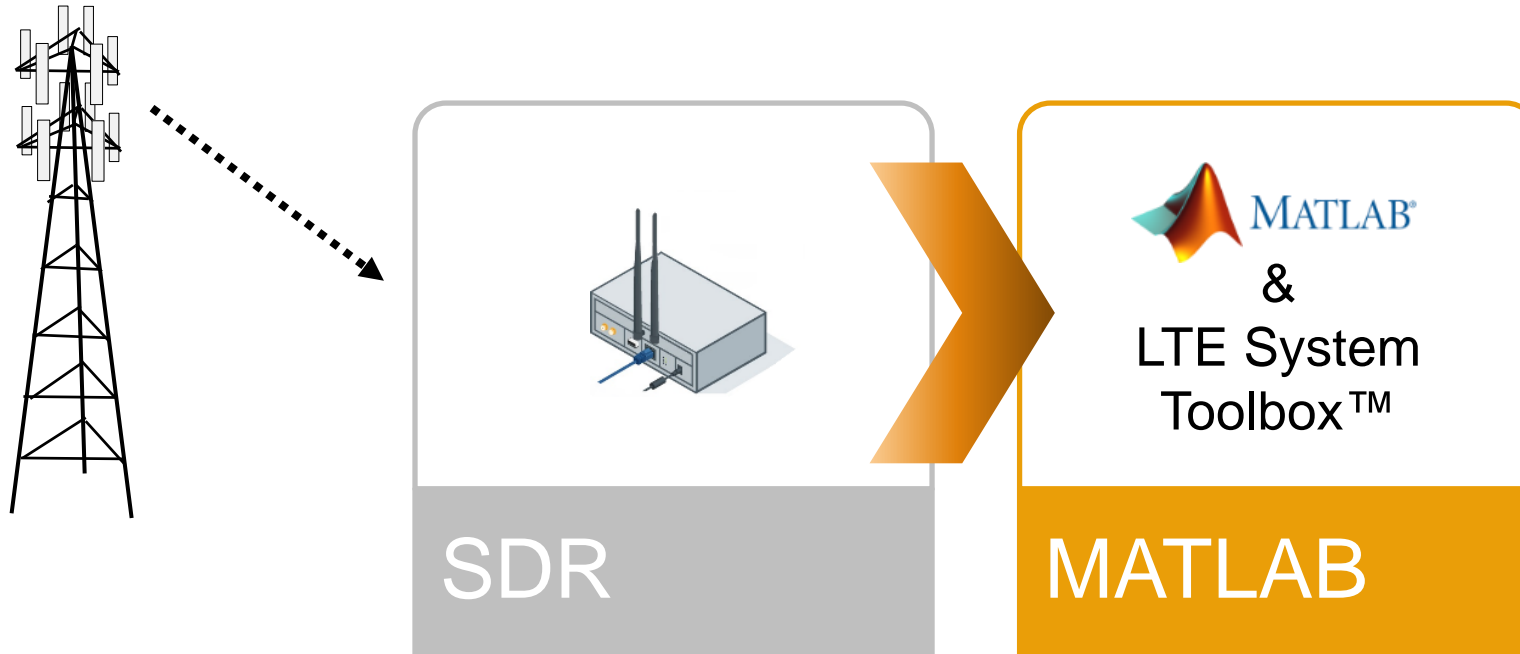
The image displays the MATLAB R2017b software interface. The main window shows a script titled "BeaconFrameGenerationExample.m" with the following code:

```
1 %% 802.11 OFDM Beacon Frame Generation
2
3 %% Create IEEE 802.11 Beacon Frame
4
5 SSID = 'MATLAB'; % Network SSID
6 beaconInterval = 100; % In Time units (TU)
7 band = 5; % Band, 5 or 2.4 GHz
8 chNum = 60; % Channel number, corresponds to 5300 MHz
9
10 % Generate Beacon frame
11 [mpduBits,fc] = helperGenerateBeaconFrame(chNum, band, beaconInterval, SSID);
12
13 %% Create IEEE 802.11 Beacon Packet
14
15 cfgNonHT = wlanNonHTConfig; % Create a wlanNonHTConfig object
```

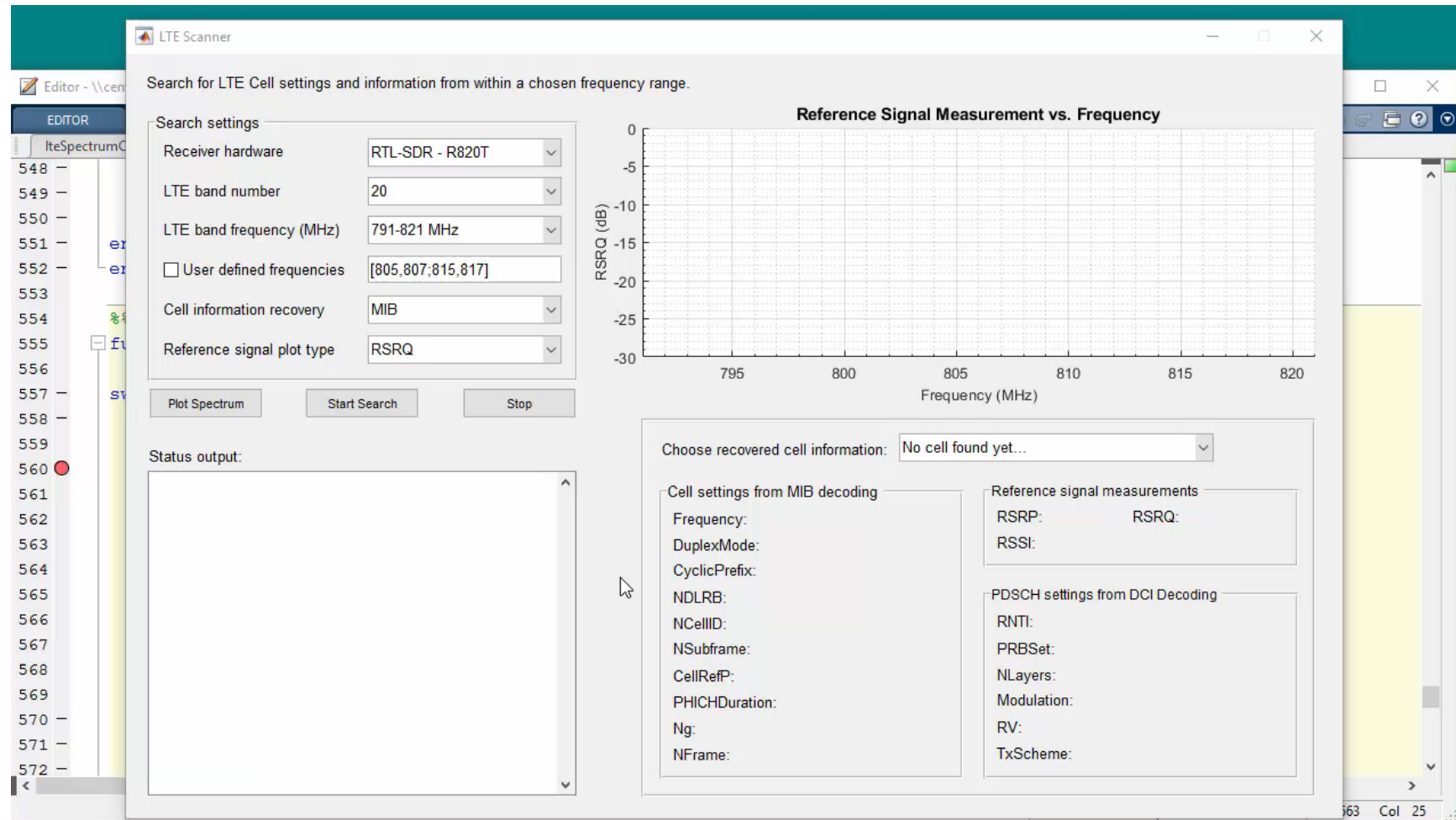
The Command Window shows the prompt `fx >>`. The status bar at the bottom indicates "script" and "Ln 11 Col 78".

Overlaid on the right side of the MATLAB window is a photograph of a hardware setup. It features a circuit board with a WLAN module, connected to a smartphone. The smartphone screen displays a "Channel Graph" application, showing a spectral plot with a prominent peak in the 5.3 GHz range, indicating the beacon signal being generated.

# Demo: LTE Scanner

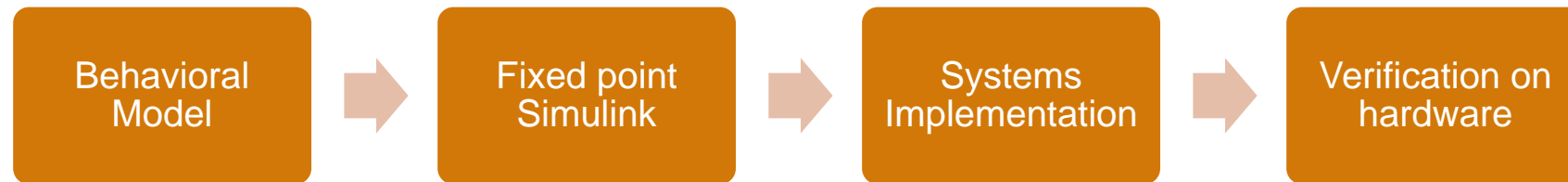


# Demo: LTE Cell Scanner



# Simulink for Wireless System Design

# From Design to Prototype



# From Design To Hardware

## MATLAB

- ✓ Large data sets
- ✓ Explore mathematics
- ✓ Data visualization

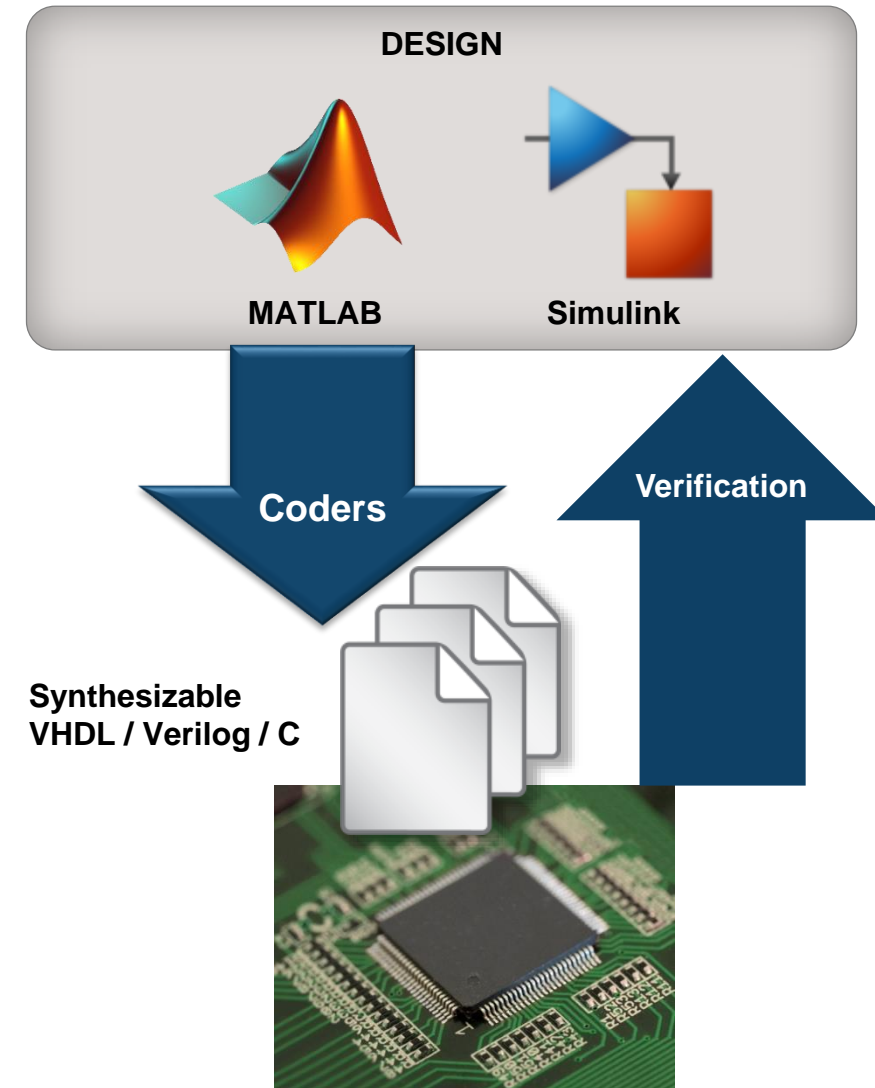
## Simulink

- ✓ Parallel architectures
- ✓ Timing
- ✓ Data propagation



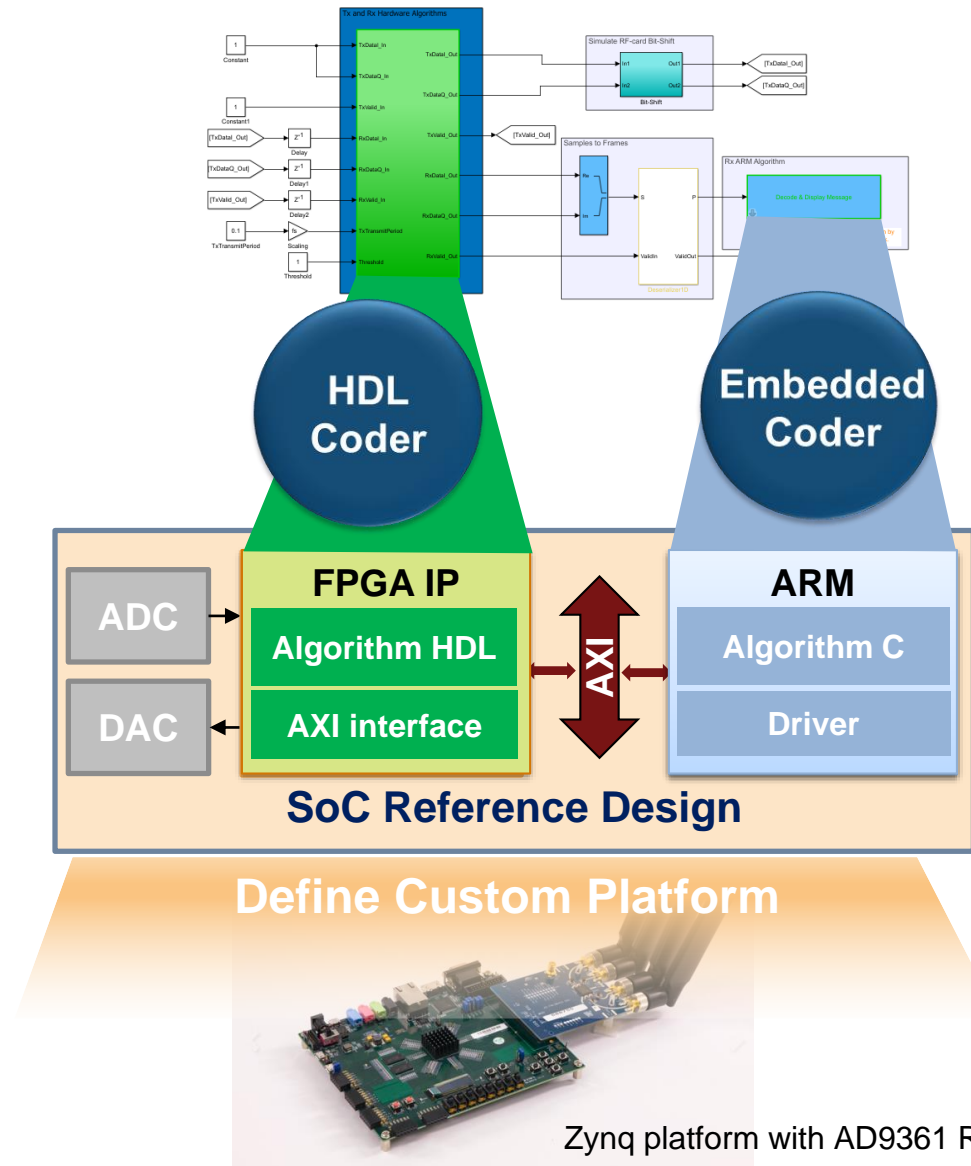
## Targeting FPGA and ASIC

- Streaming design
- Implementation detail
- Architectural specification
- Verification





# Generate C and HDL Code and Implement On Hardware



# Verification: MATLAB reference model to FPGA implementation

```
% generate bits for transmission
txBits = randi([0 1],6144,1);
% encode and modulate bits
codedData = lteTurboEncode(txBits);
txSymbols = lteSymbolModulate(codedData,'QPSK');

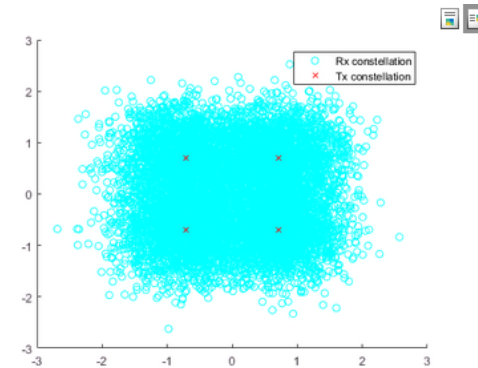
% add noise
noise = 0.5*complex(randn(size(txSymbols)),randn(size(txSymbols)));
rxSymbols = txSymbols + noise;
```

```
% plot transmitted and received symbols
scatter(real(rxSymbols),imag(rxSymbols),'co'); hold on;
scatter(real(txSymbols),imag(txSymbols),'rx')
legend('Rx constellation','Tx constellation')
```

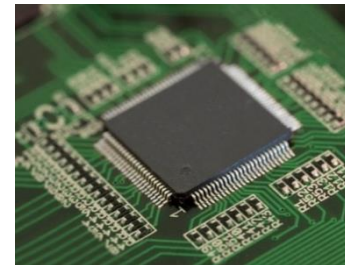
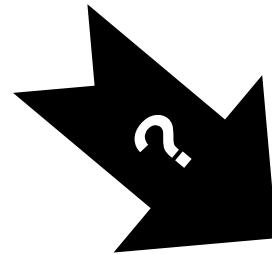
```
% demodulate data
softBits = lteSymbolDemodulate(rxSymbols,'QPSK','Soft');

% decode
rxBits = lteTurboDecode(softBits);

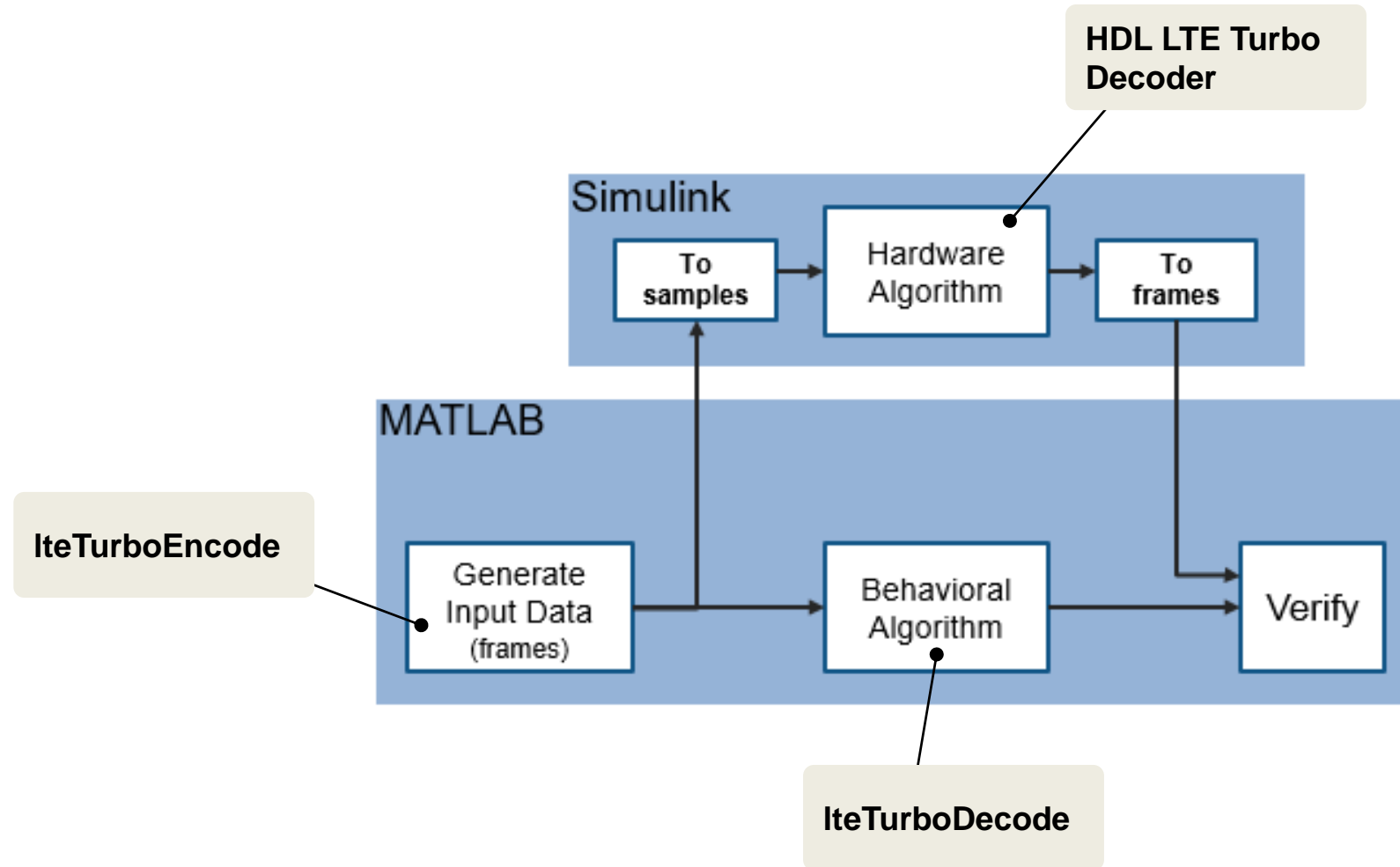
% check number of errors
numberErrors = sum(rxBits ~= int8(txBits))
```



numberErrors = 0

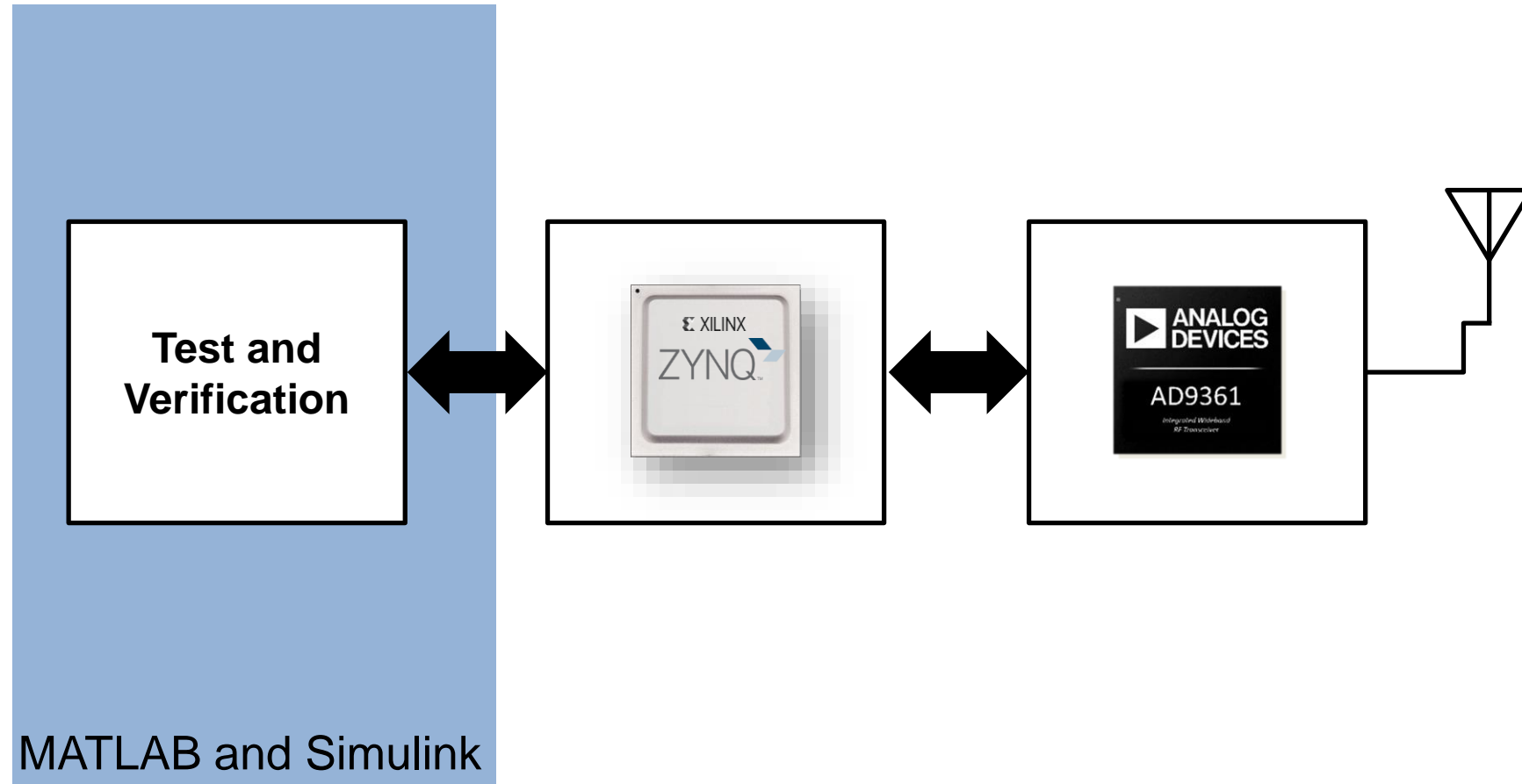


# Demo - Verify Turbo Decoder with LTE System Toolbox

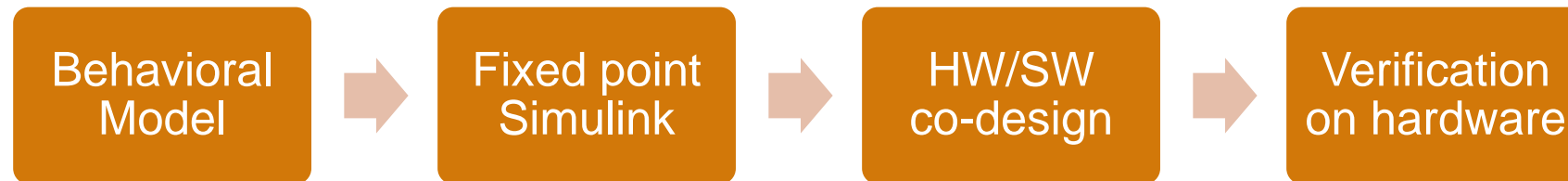


# Demo: LTE Turbo Decoder

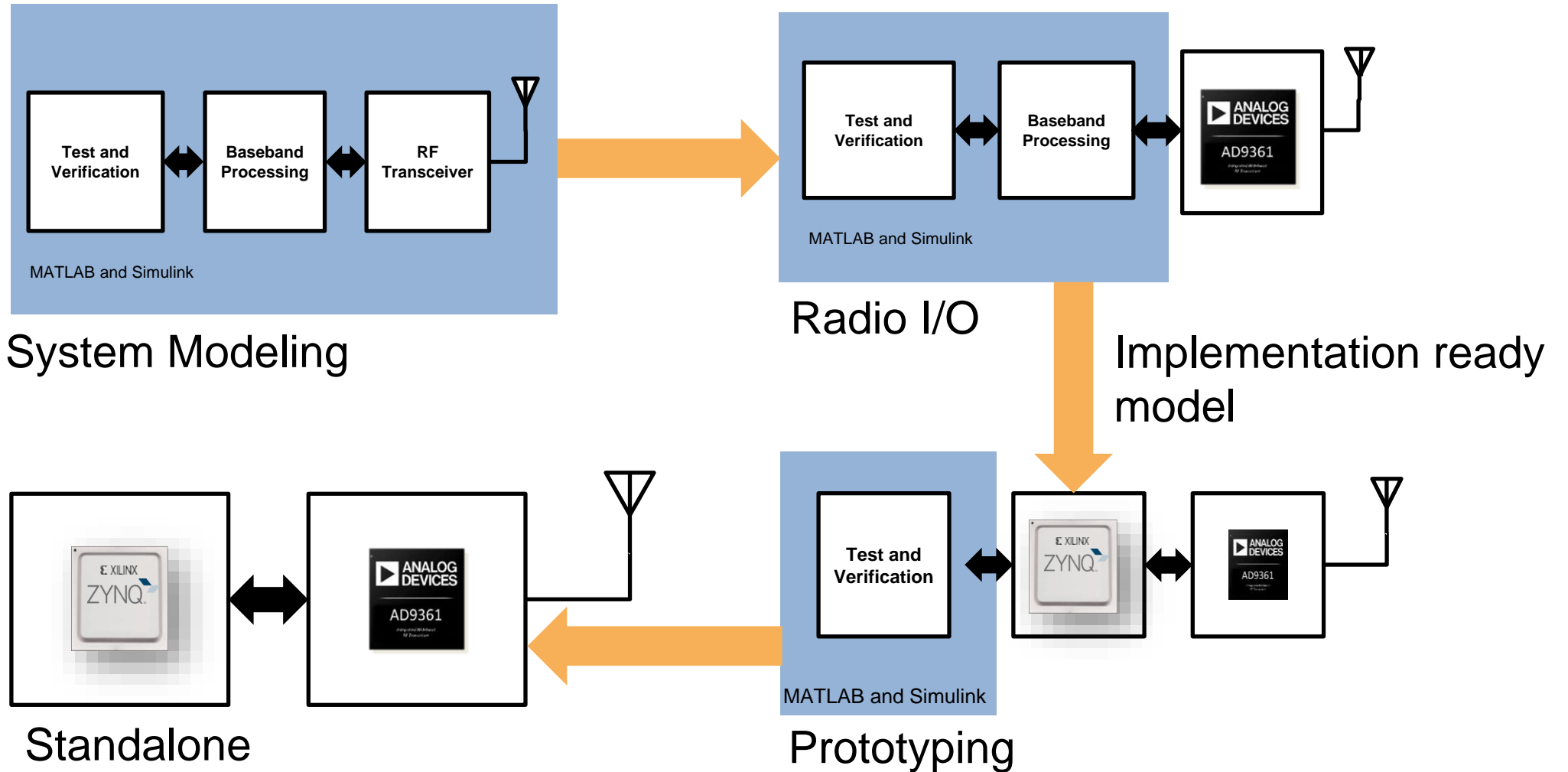
# Verification via SDR Prototyping



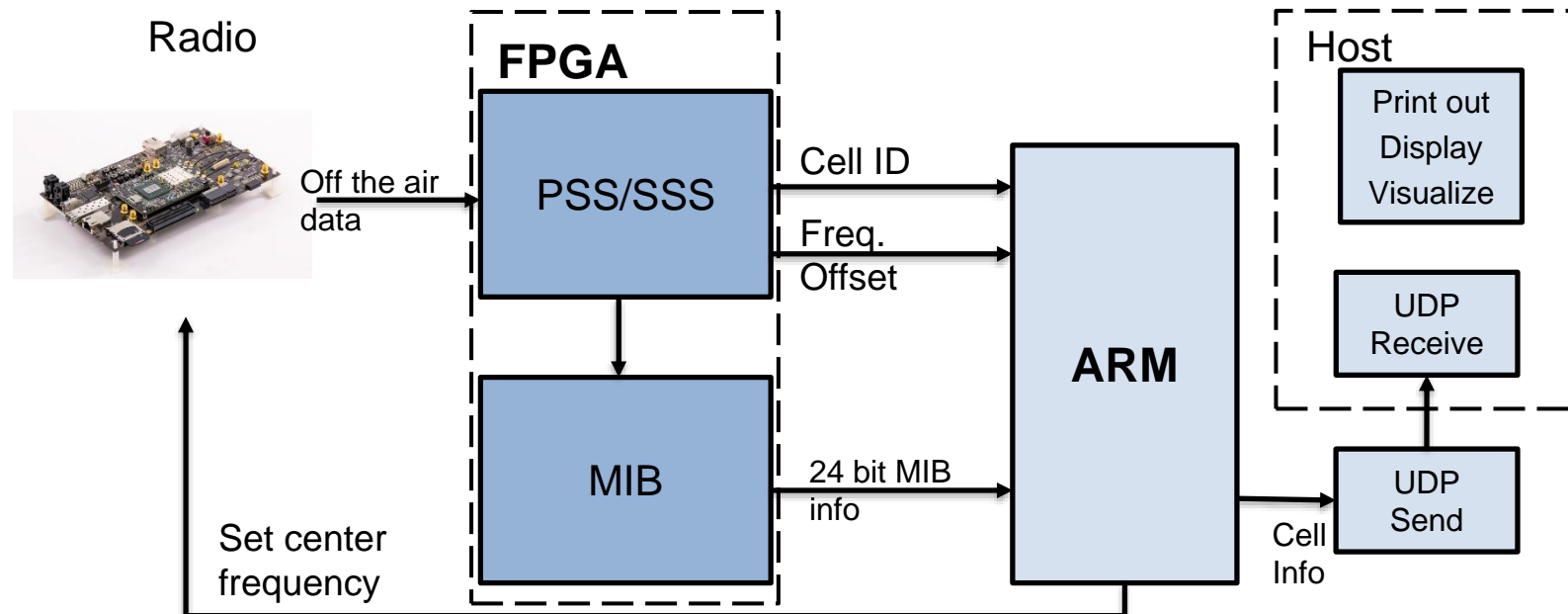
# From Design to Prototype



# Verification via SDR Prototyping

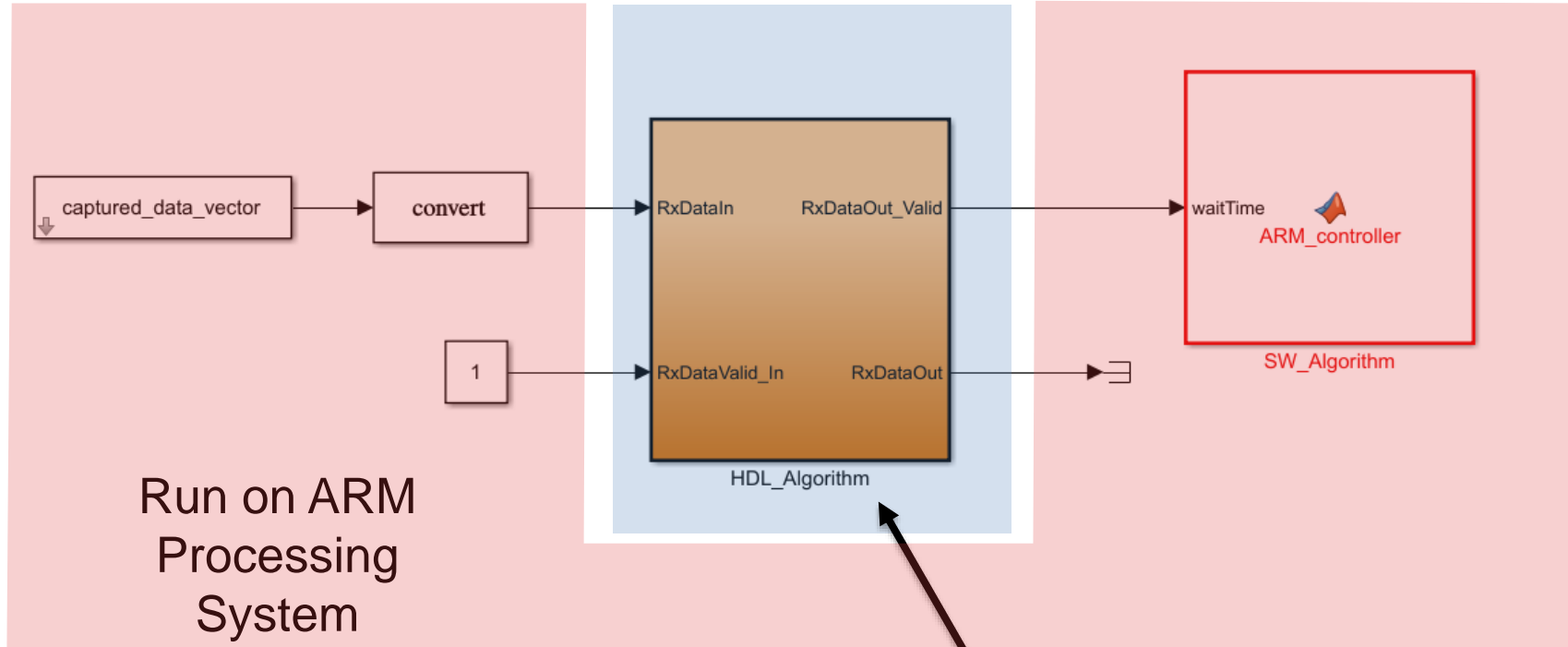


# Real time LTE Frequency Scanner



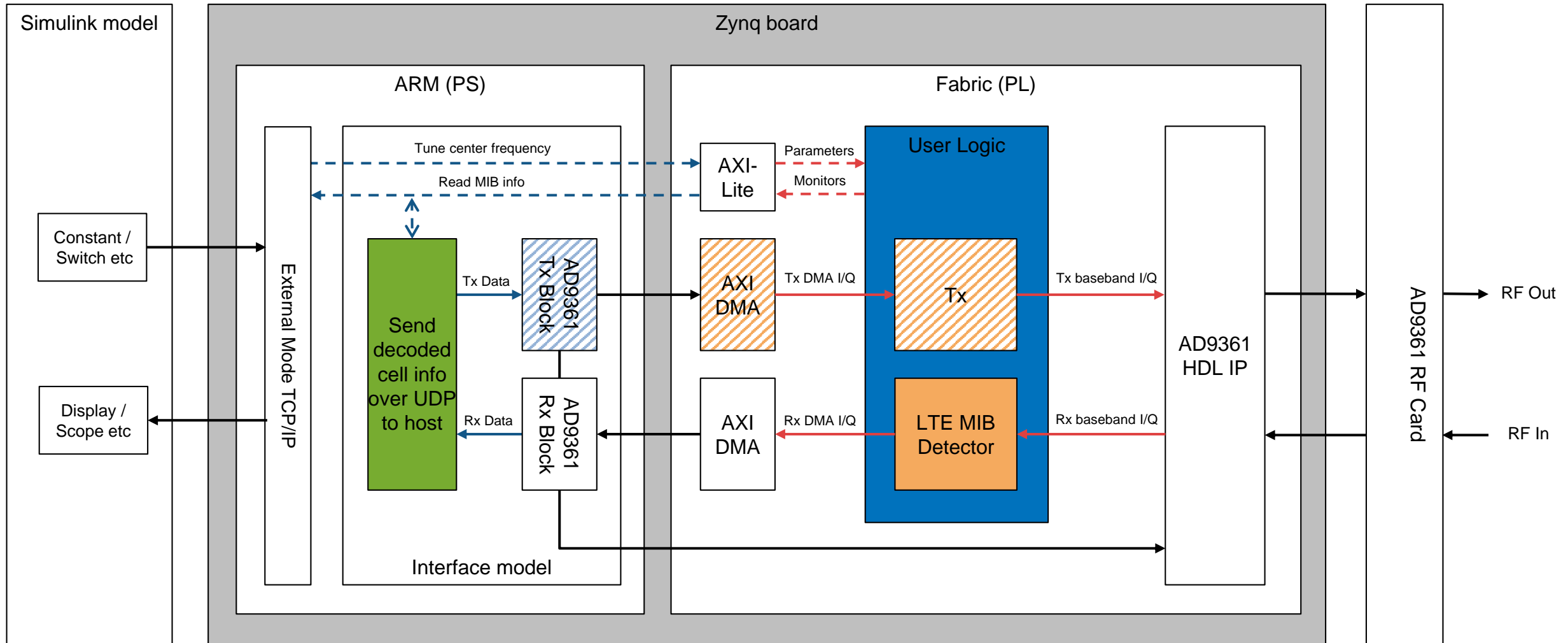


# Targeting an algorithm to the FPGA and ARM



Run on Programmable Logic

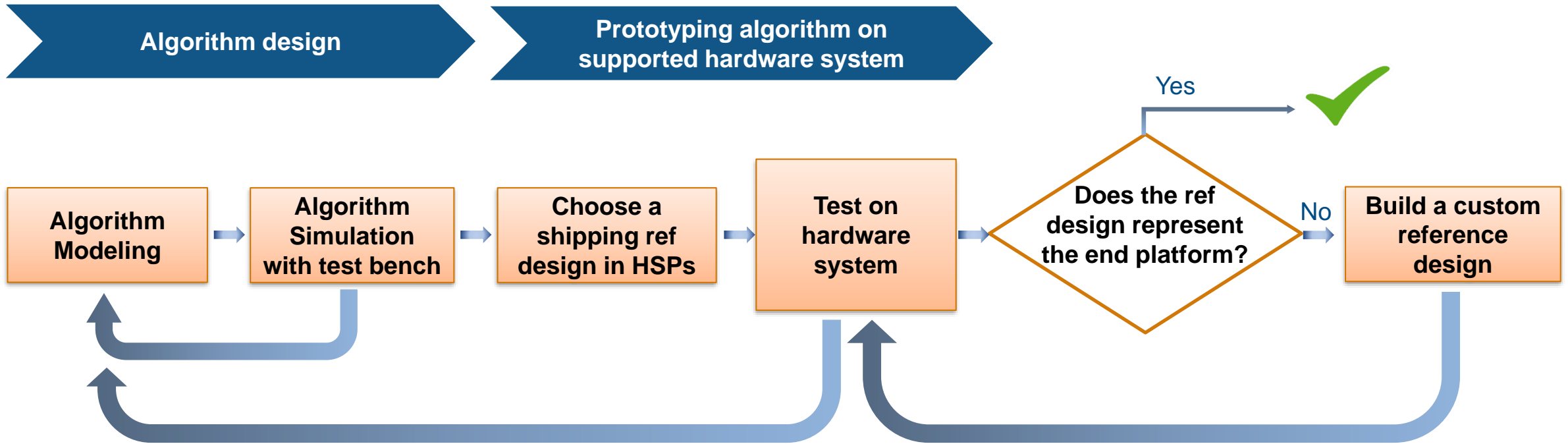
# HW/SW Co-Design Implementation of MIB Detector



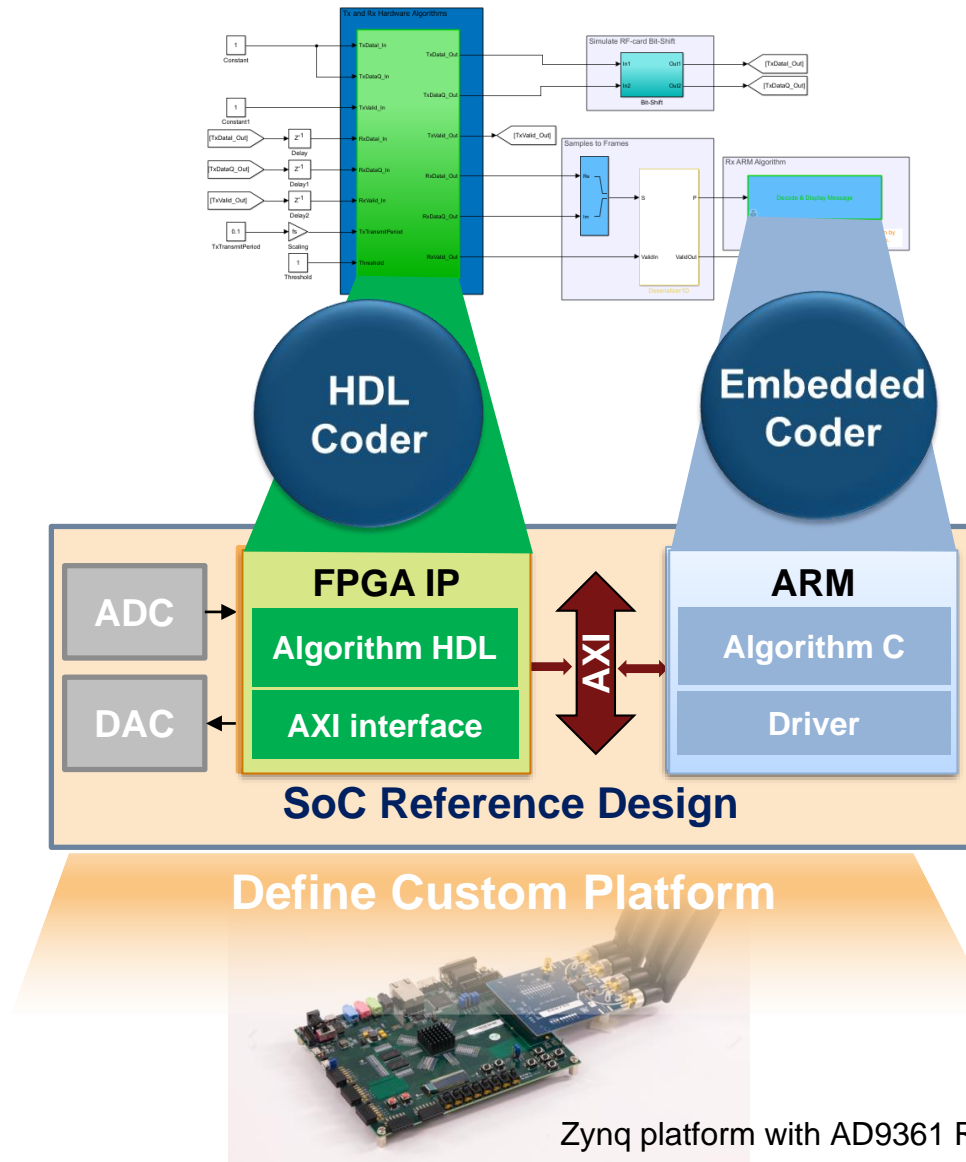
# SoC Workflow: HW/SW Co-design



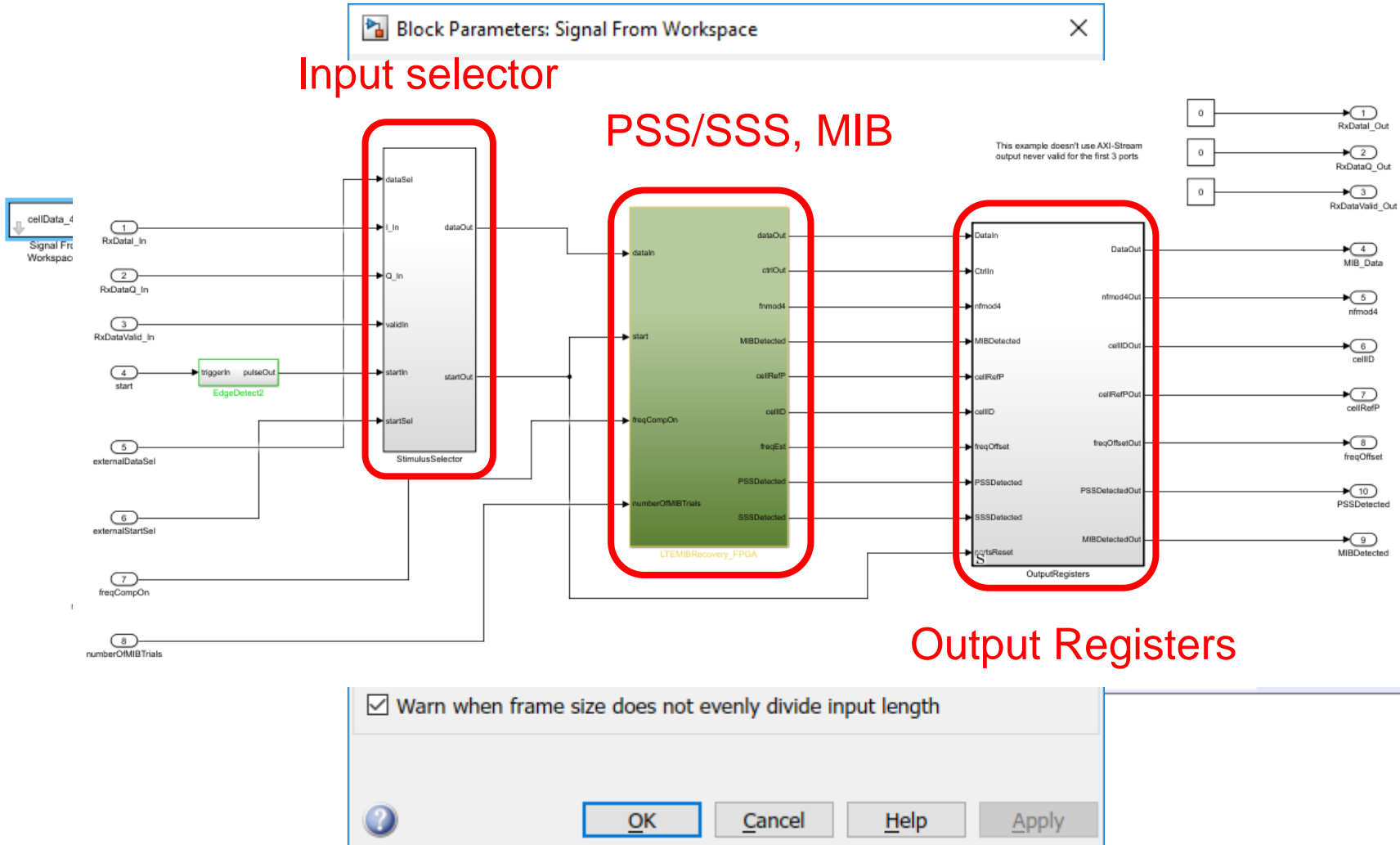
**SDR HSP**  
Zynq HSPs  
Vision HSP



# Generate C and HDL code and implement on hardware

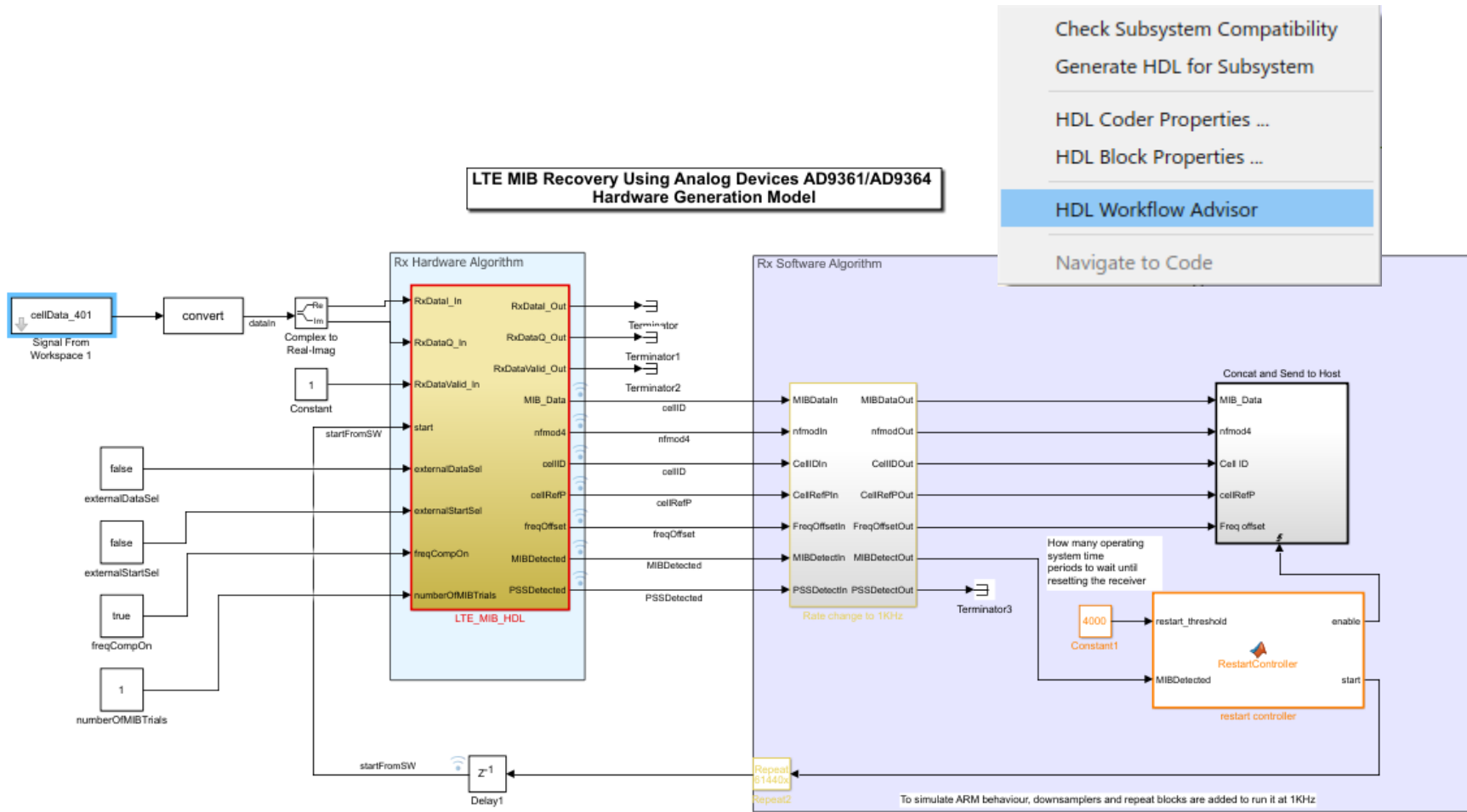


# LTE Cell Scanner Example: Algorithm



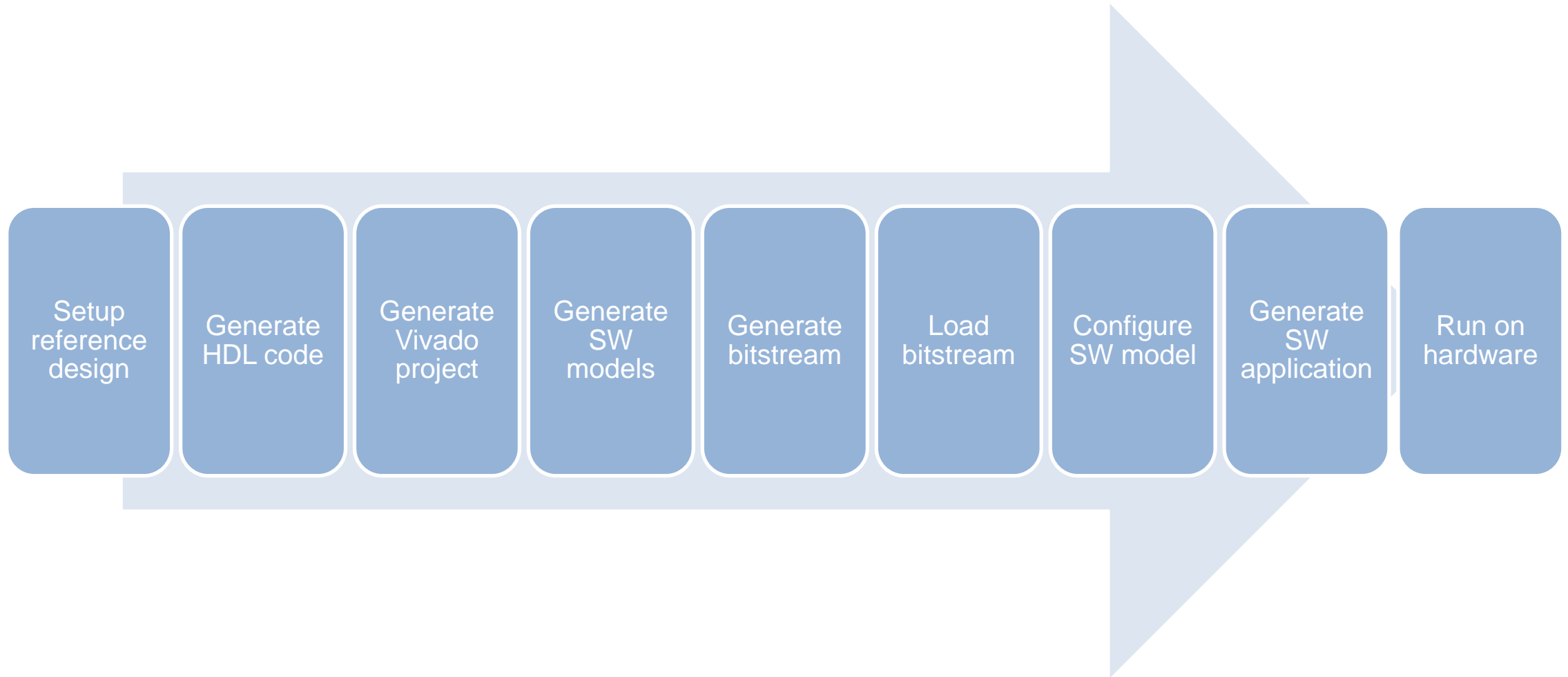
- Model algorithm
- Generate FPGA bitstream
- SW interface model
- Run on hardware

# LTE Cell Scanner Example: Generation

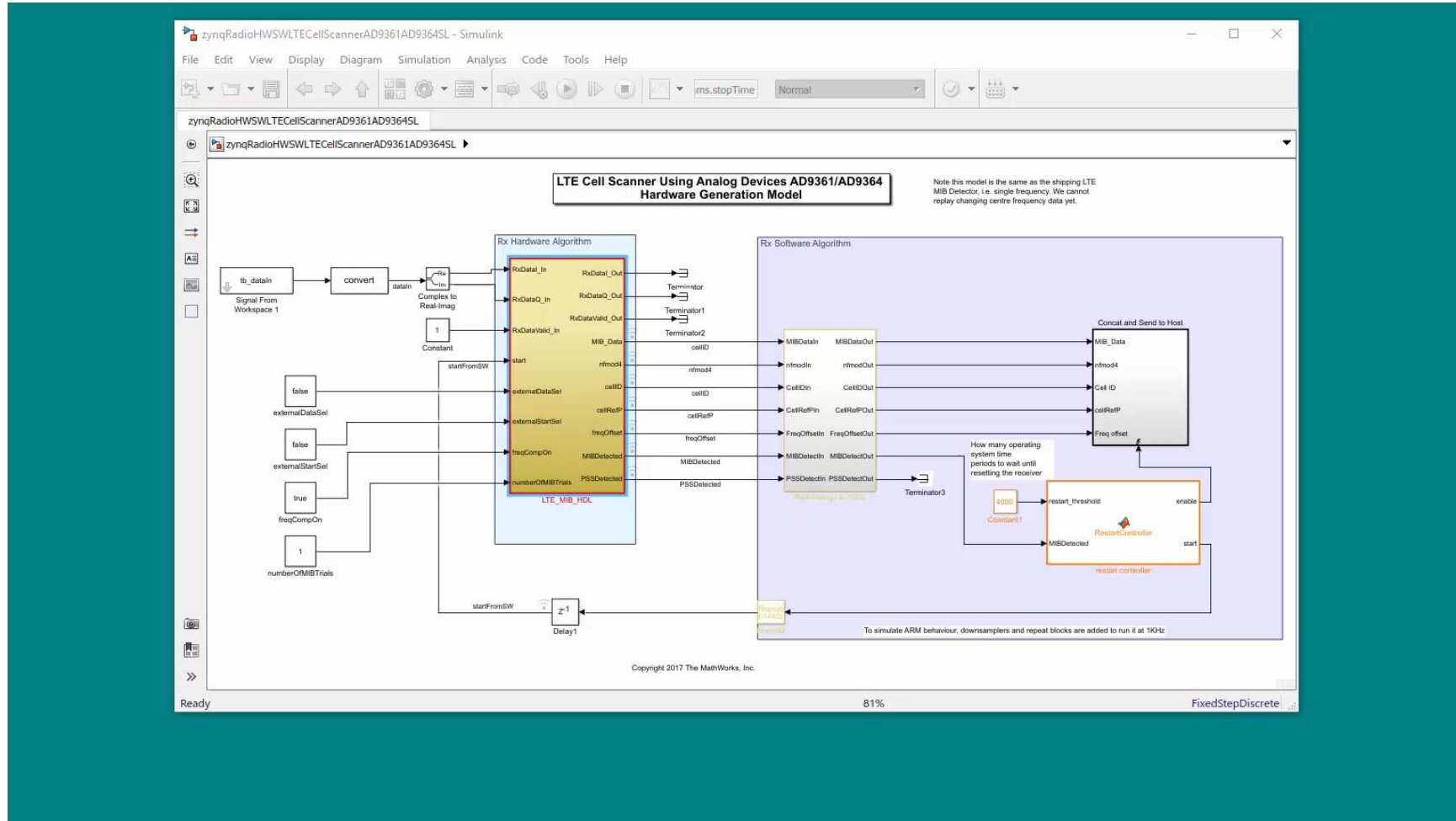


- Model algorithm
- Generate FPGA bitstream
- SW interface model
- Run on hardware

# Targeting workflow

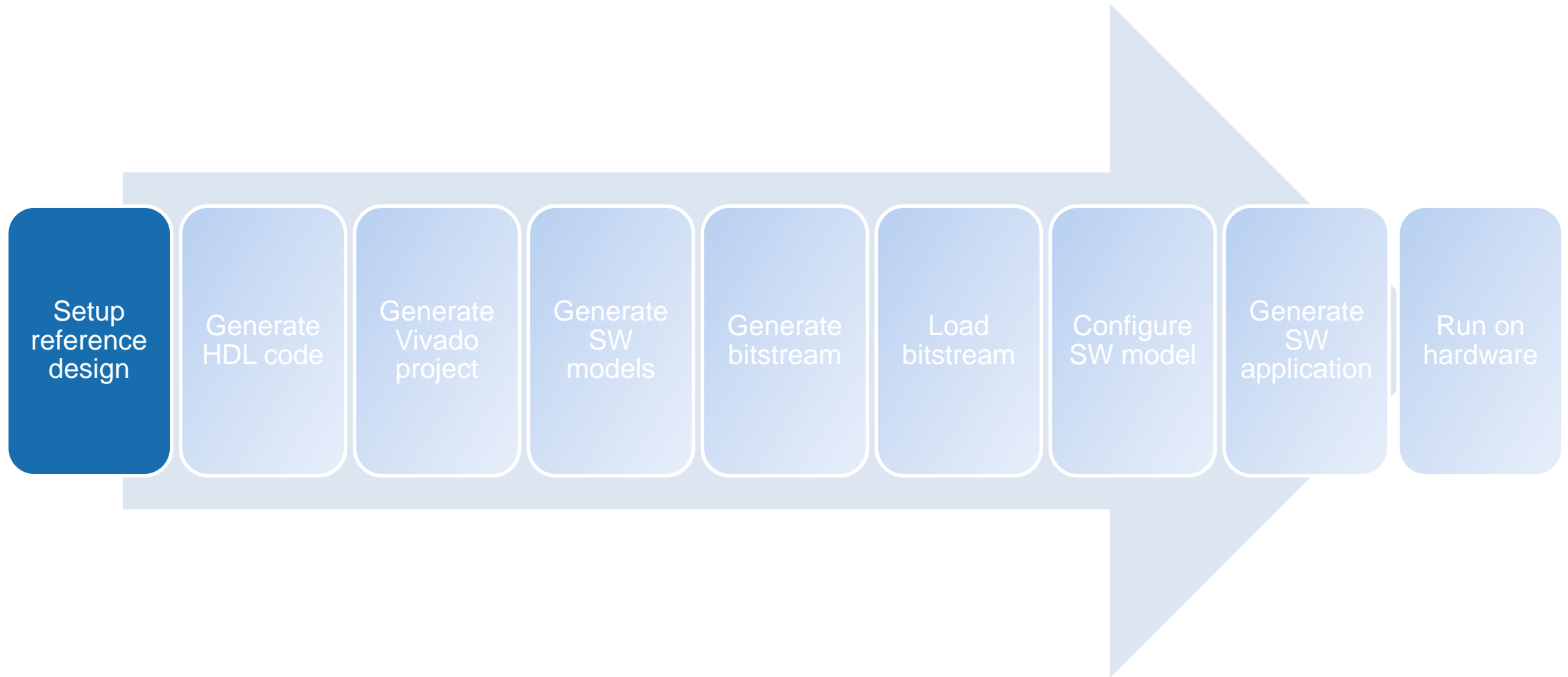


# Video: Compile Model

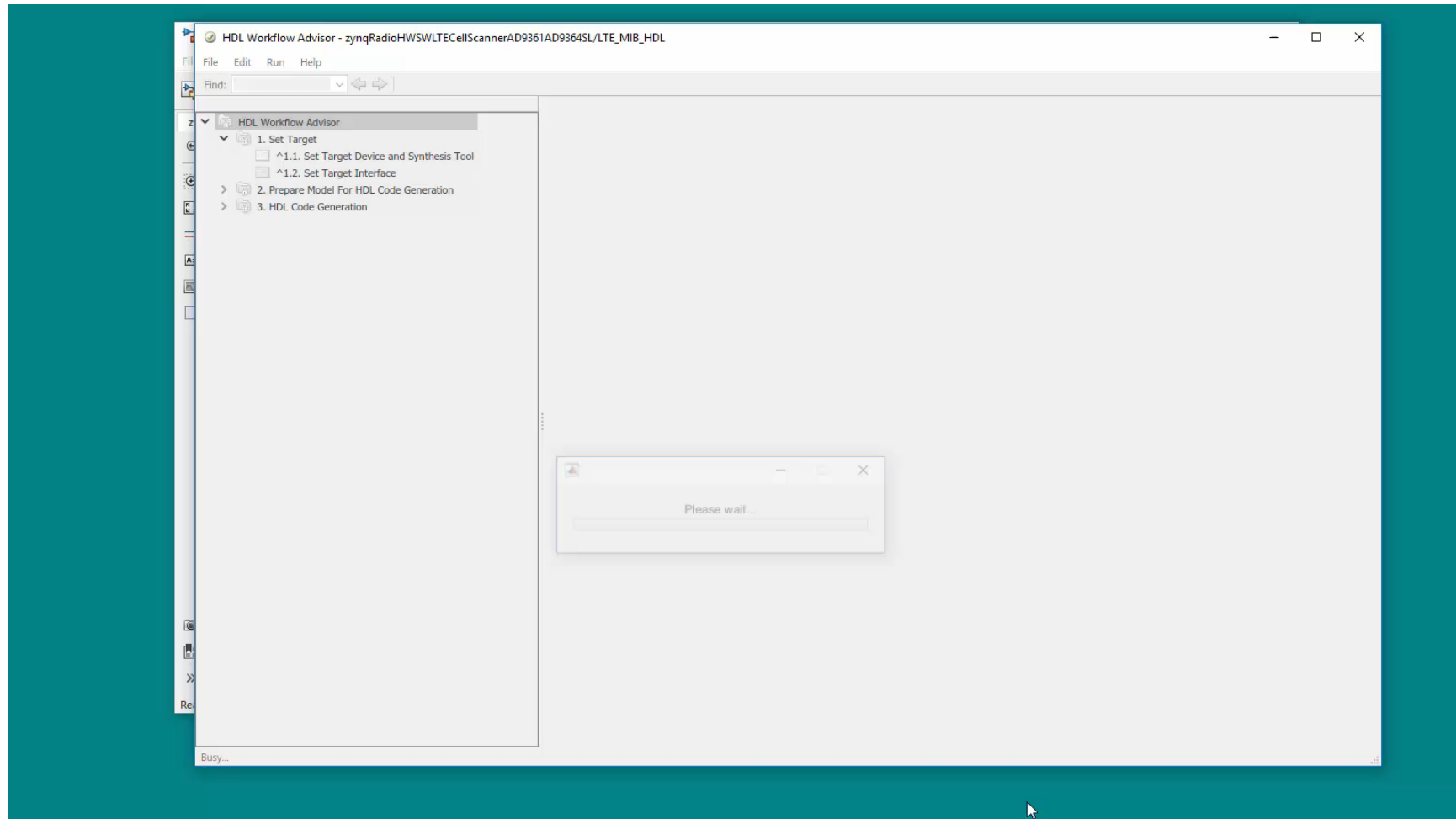




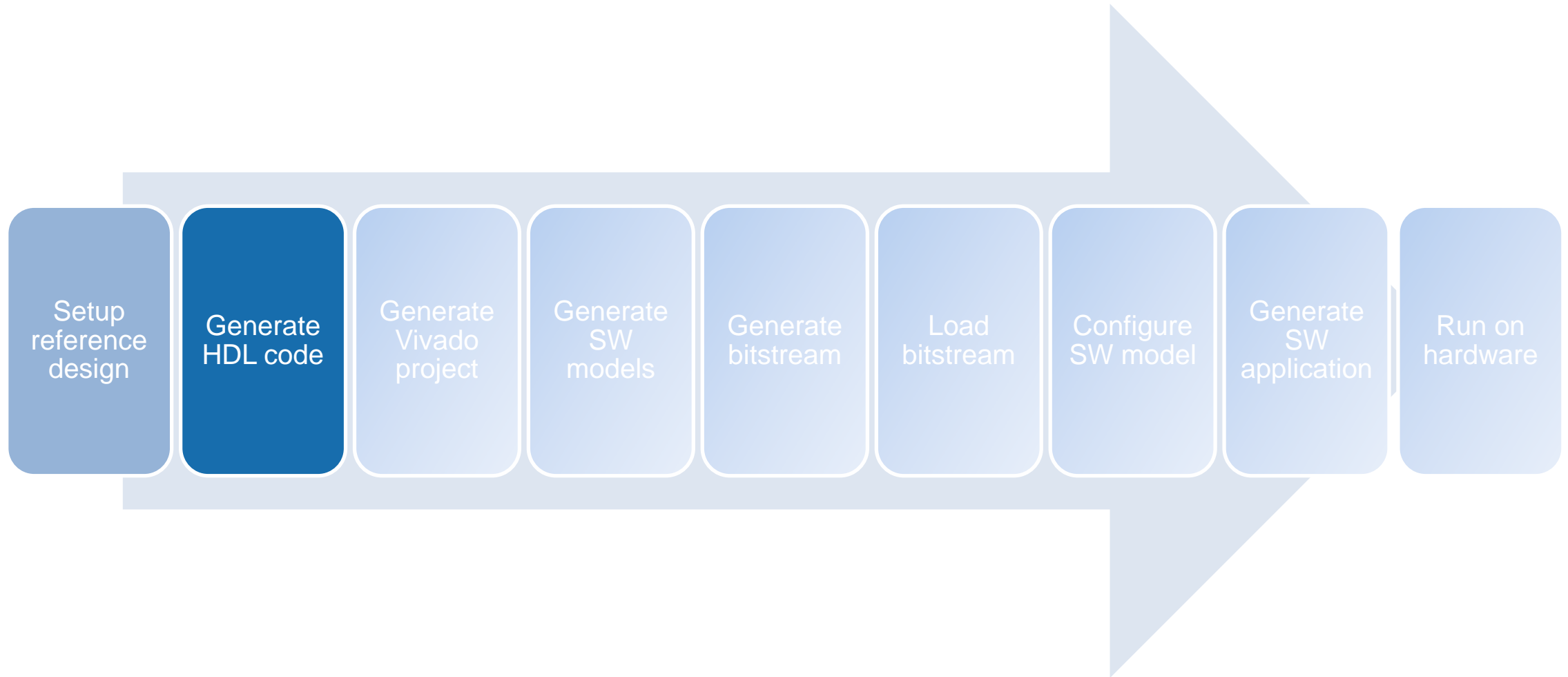
# Targeting workflow



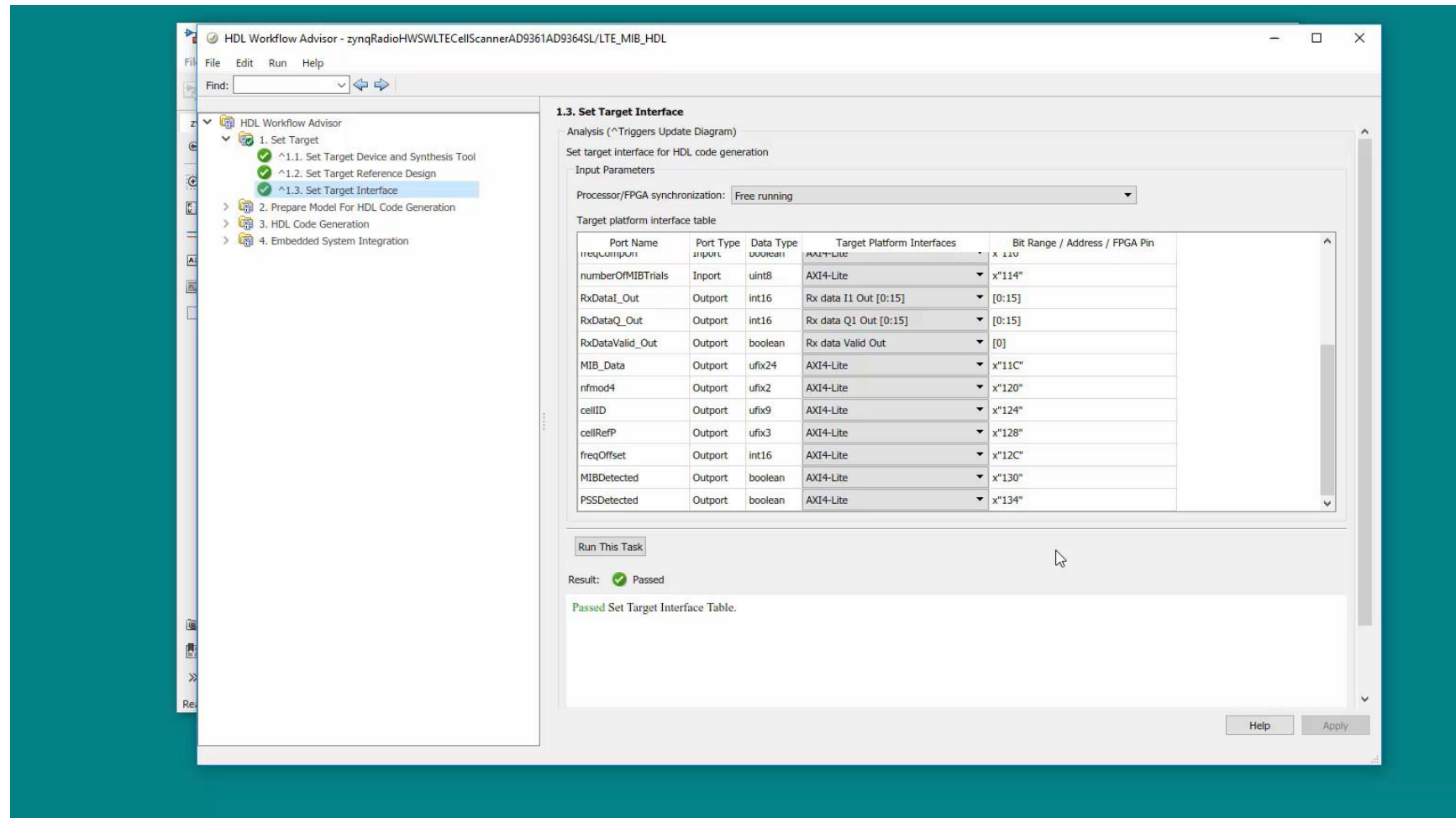
# Video: Setup reference design



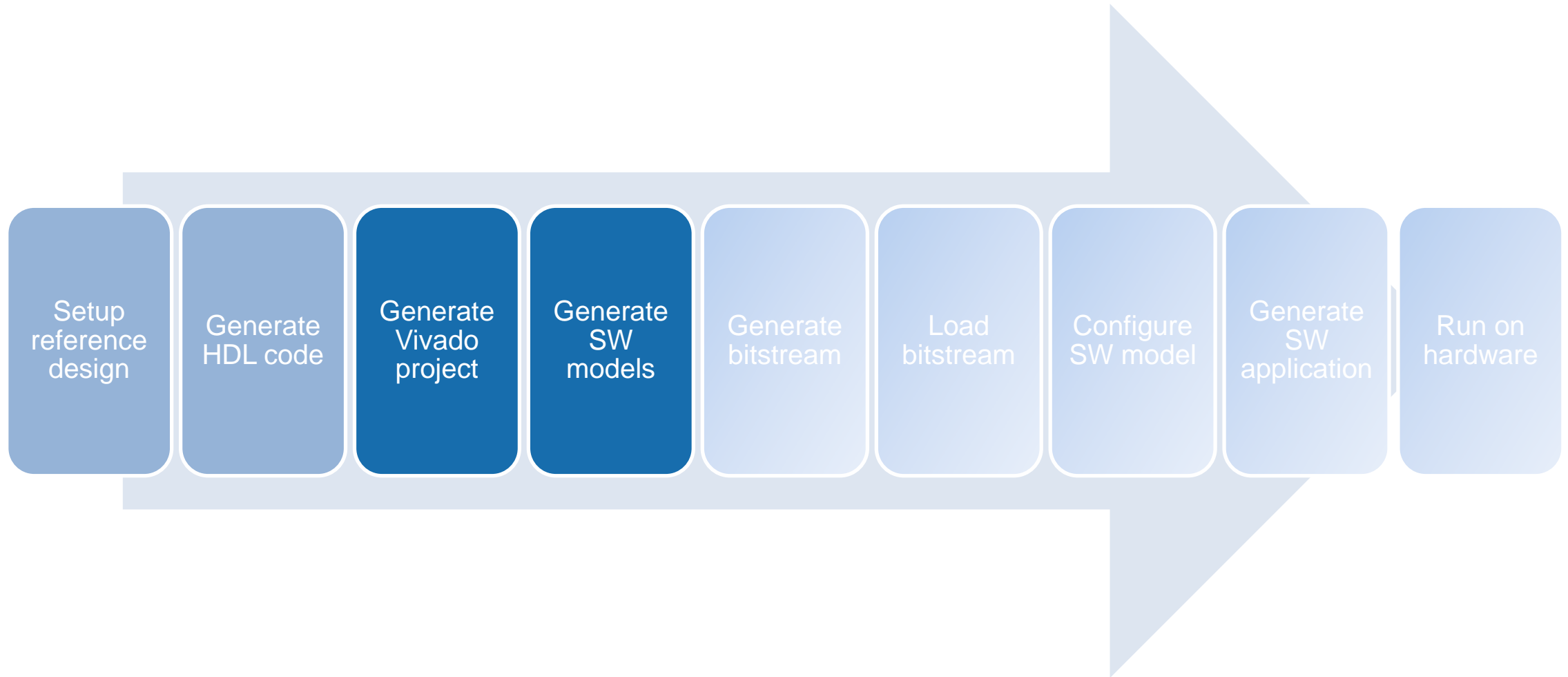
# Targeting workflow



# Video: Generate HDL code



# Targeting workflow



# Video: Generate Vivado project and software models

The screenshot displays the HDL Workflow Advisor interface for a project named 'zynqRadioHWSWLTECellScannerAD9361AD9364SL/LTE\_MIB\_HDL'. The left pane shows a tree view of the workflow tasks, with '3.2. Generate RTL Code and IP Core' selected. The right pane shows the configuration for this task, including input parameters and a 'Run This Task' button. Below the configuration, the execution result is shown as 'Passed', followed by a log of messages detailing the HDL generation process and pipeline delays.

**3.2. Generate RTL Code and IP Core**  
Analysis (^Triggers Update Diagram)  
Generate RTL code and IP core for embedded system

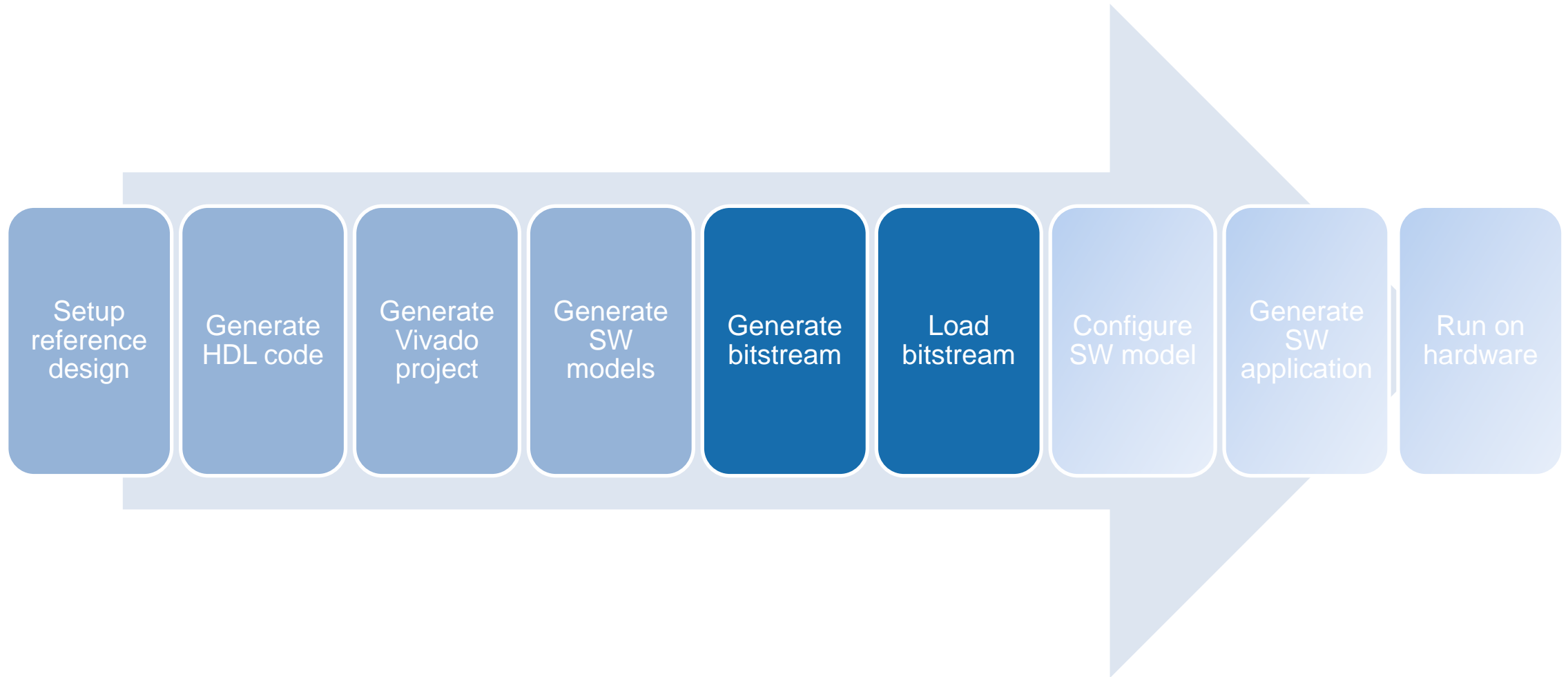
Input Parameters

IP core name: LTE\_MIB\_H\_ip  
IP core version: 1.0  
IP core folder: ADI\_RF\_SOM\_CellScanner\ipcore\LTE\_MIB\_H\_ip\_v1\_0  
IP repository:  Browse...  
Additional source files:  Add Source...  
 Generate IP core report

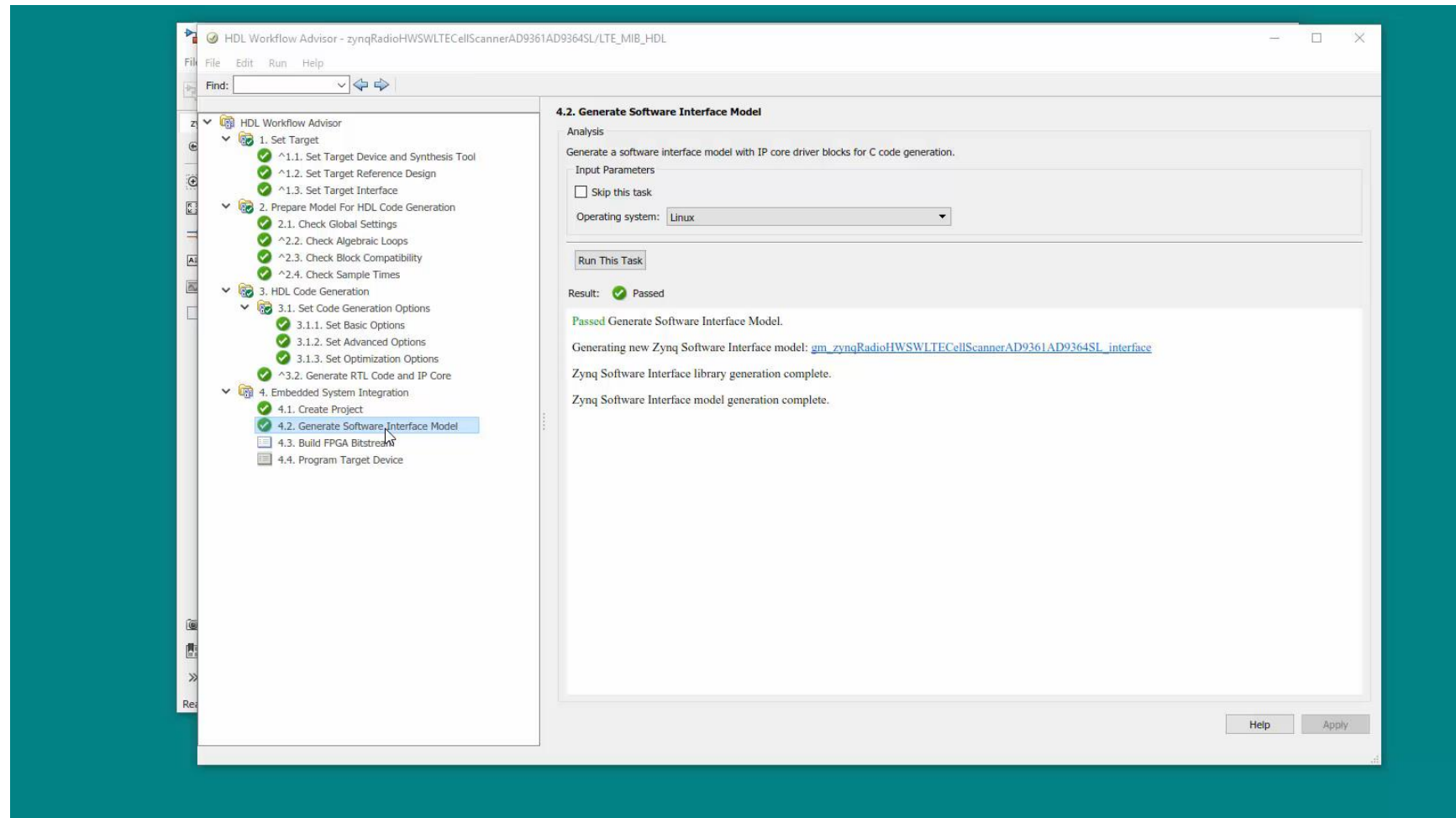
Result:  Passed

```
### Generating HDL for 'zynqRadioHWSWLTECellScannerAD9361AD9364SL/LTE_MIB_HDL'.  
### Using the config set for model zynqRadioHWSWLTECellScannerAD9361AD9364SL for HDL code generation parameters.  
### Starting HDL check.  
### The code generation and optimization options you have chosen have introduced additional pipeline delays.  
### The delay balancing feature has automatically inserted matching delays for compensation.  
### The DUT requires an initial pipeline setup latency. Each output port experiences these additional delays.  
### Output port 0: 32 cycles.  
### Output port 1: 32 cycles.  
### Output port 2: 32 cycles.  
### Output port 3: 32 cycles.
```

# Targeting workflow

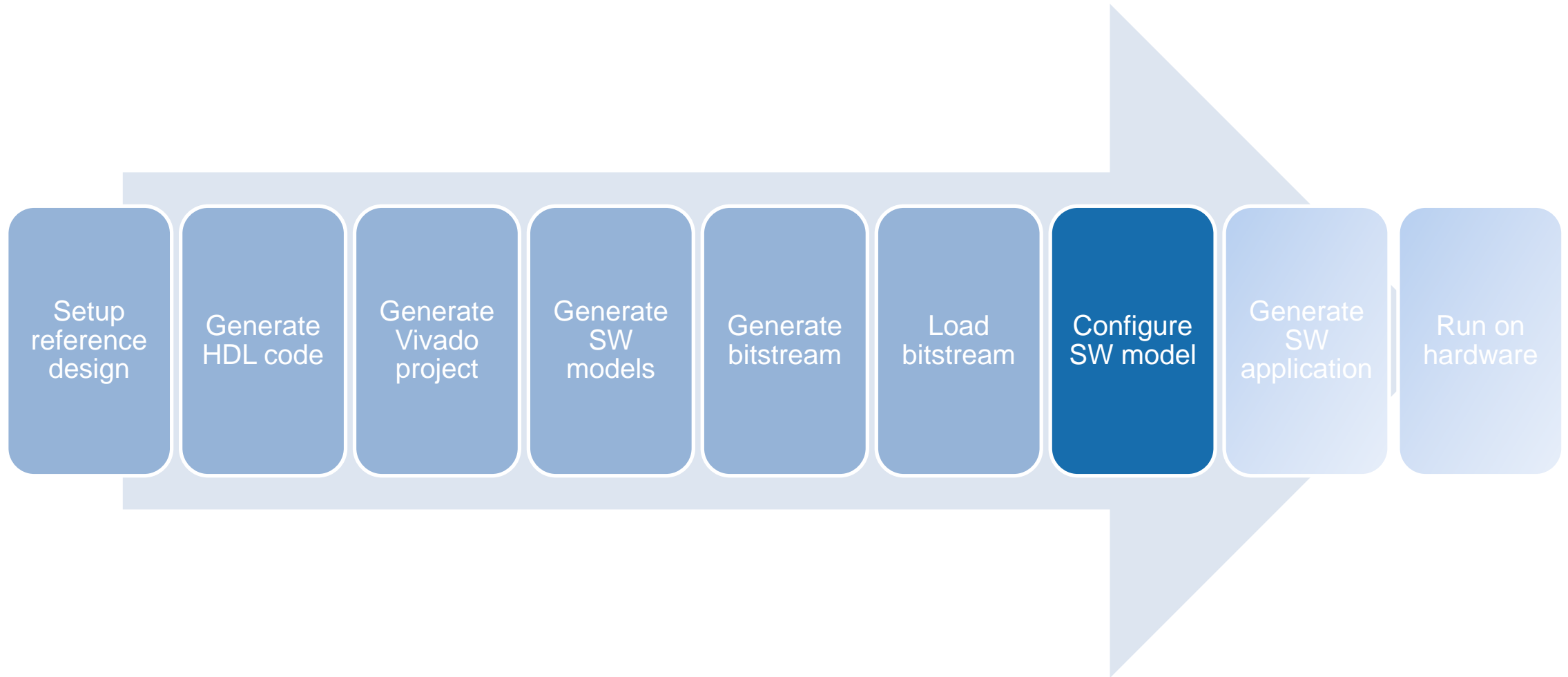


# Video: Generate bitstream



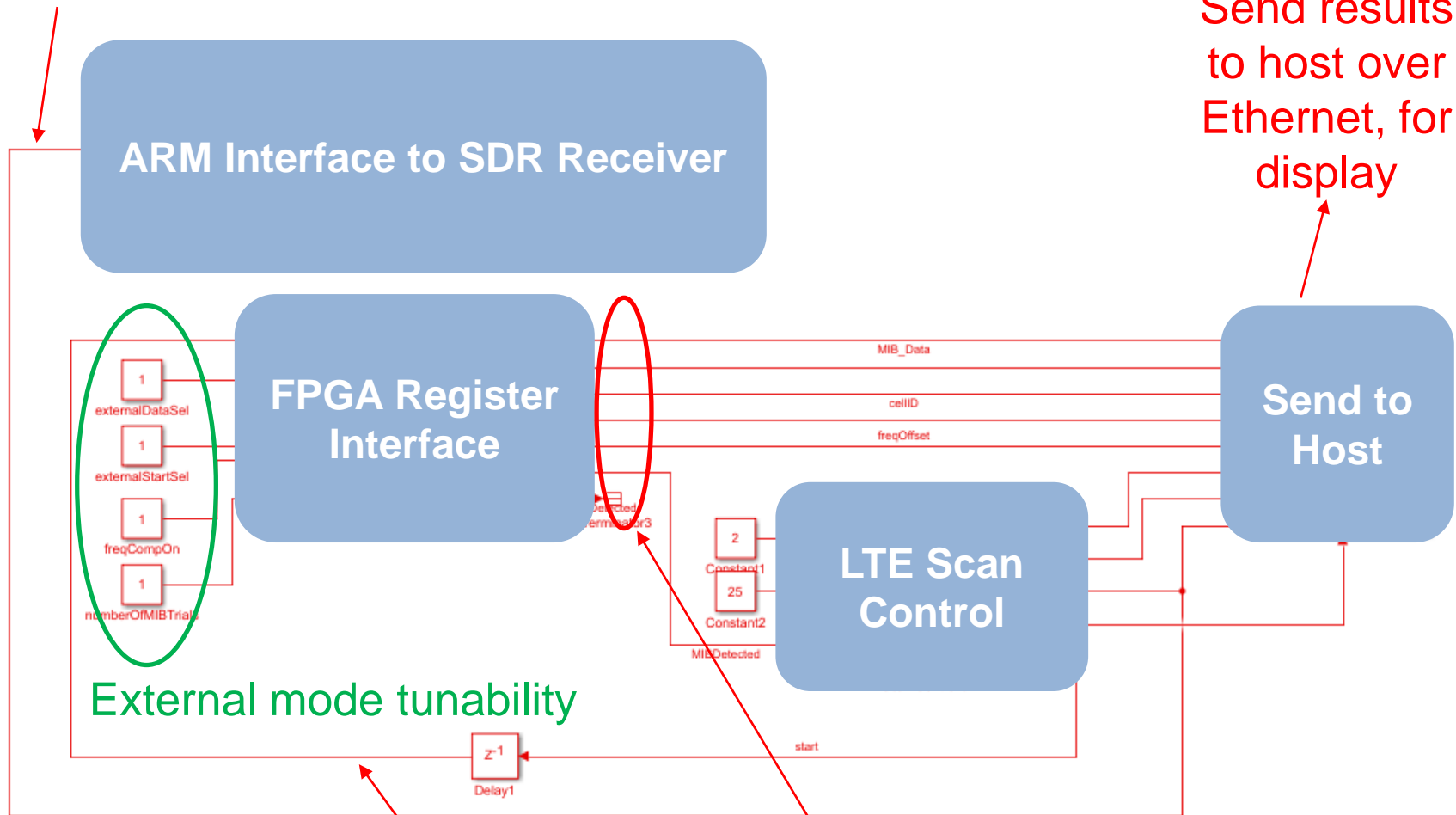


# Targeting workflow



# LTE Cell Scanner Example: Software

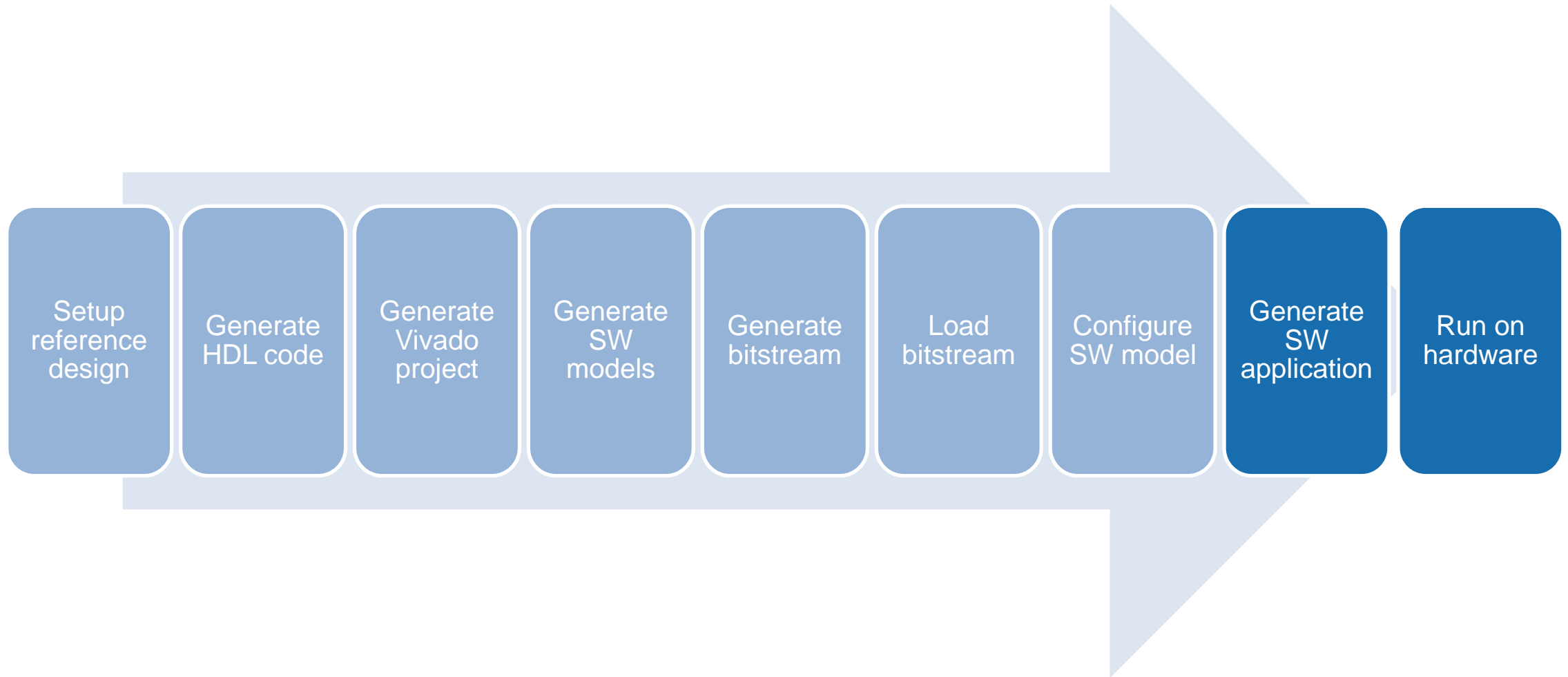
Center Frequency



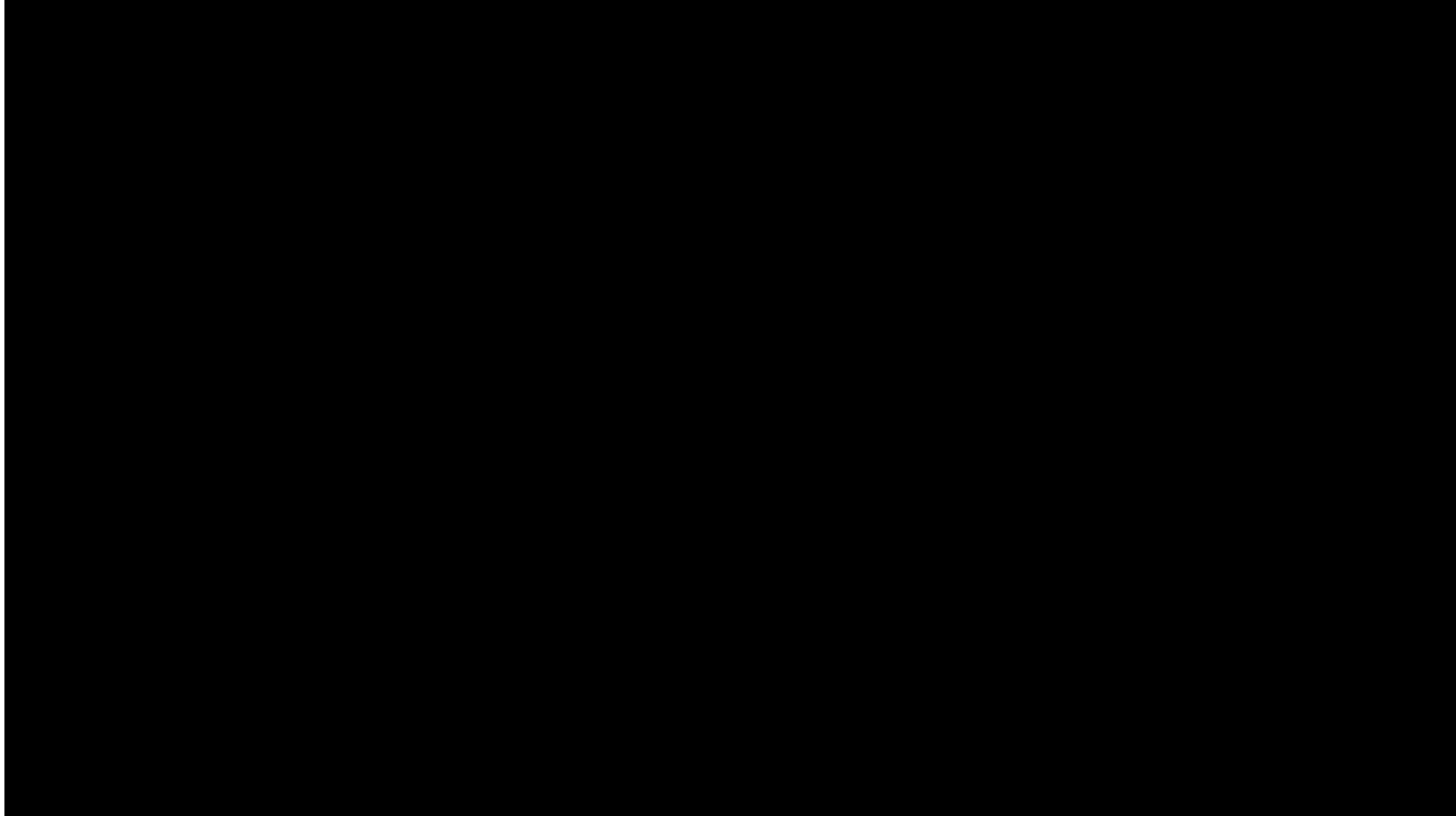
Send results to host over Ethernet, for display

- Model algorithm
- Generate FPGA bitstream
- SW interface model
- Run on hardware

# Targeting workflow



## Video: Run on hardware



**Thank you!**