A Traceable Workflow for Software Defined Radio Development

Travis Collins, PhD
Andrei Cozma, PhD
Analog Devices, Inc.
<travis.collins@analog.com>
Problems with SDR

• 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
Typical Toolsets and Workflow For Algorithms

Create MATLAB Floating Point Reference Design

Create Python Implementation For Data Streaming

Create C/C++ Fixed-Point Version

Create Verilog/VHDL Version

Stream Data into MATLAB design

Convert to Simulink and use Fixed-Point Tools

Generate HDL From Simulink Model

Create MATLAB Floating Point Reference Design

Create SysGen/DSPBuilder Models

Generate HDL From Simulink Model

Create MATLAB Floating Point Reference Design

Create Python Implementation For Data Streaming

Create C/C++ HLx Version

Generate HDL From HLx Blocks

Create MATLAB Floating Point Reference Design

Create Python Implementation For Data Streaming

Create C/C++ HLx Version

Generate HDL From HLx Blocks
Hardware

1. Start with Evaluation Board/Hardware
2. Build Custom Demo/Eval Board To Test Functionality
3. Build Final Production Board For Deployment
• 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 + AARRADIO

RF SOM + Breakout Board
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
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)
Model Based Design Workflow With Hardware
PlutoSDR
Streams over USB
Includes: Host Libraries (libiio, libad9361-iio), GUI Software, GNU Radio and MATLAB application interfaces

RFSoM+FMC Carrier or Eval FMC + FPGA Carrier
Streams over USB/Ethernet, allows access to FPGA and local CPU (standalone operation), blue wire to HW
Includes above plus: Device Drivers, HDL interfaces, HDL libraries, Schematics, Gerber

PackRF or RFSoM + Custom Carrier
Prototype field testing, trials or bake off
Includes above plus standard peripheral access (screen, battery, GPS, PoE, Audio, etc)

Custom
Does whatever you want
Could include one or more or none of ADI: Host Libraries, GUI Software, Device Libraries, Device Drivers, HDL, Schematics, Gerber
Example Reference Design Demonstrating Workflow

- Example design works through QPSK modem development
- Example details:
  - QPSK PHY with continuous link
  - Simple FDD system (MAC)
  - Built with common algorithms
Where can I get the code?

- Available in Add-On Explorer today
- 3 main reference designs
  - MATLAB Floating-Point
  - Simulink Floating-Point
  - Simulink Fixed-Point
- 5 deployable examples that show debugging techniques
  - Standard IQ
  - External Mode
  - AXI-MM
  - FPGA Capture
  - PackRF Custom BSP

- Testing harness
- Utility scripts
Workflow

Research → Algorithm Development → Design Elaboration → Prototype → Production
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
```
>> rx = sdrxx('ZC706 and FMCOMMS2/3/4');
>> rx = sdrxx('ZedBoard and FMCOMMS2/3/4');
>> rx = sdrxx('ADI RF SOM');
>> rx = sdrxx('Pluto')
```
Integration with custom software – IIO, A Kernel Subsystem for Converters

- The Linux industrial I/O subsystem is intended to provide support for devices that, in some sense, are analog-to-digital or digital-to-analog converters
  - Devices that fall into this category are:
    - Precision ADCs, high-speed ADCs
    - Precision DACs, high-speed DACs
    - Accelerometers, gyroscopes, IMUs
    - Capacitance-to-Digital converters (CDCs)
    - Pressure, proximity, temperature and light sensors
    - Health, chemical, magnetometer, amplifiers, etc.
  - Can be used on ADCs ranging from a SoC ADC to >1000 MSPS
  - Mostly focused on user-space abstraction, but also in-kernel API for other drivers exists
    - IIO to Linux input or HWMON subsystem bridges
Workflow

- Research
- Algorithm Development
- Design Elaboration
- Prototype
- Production
<table>
<thead>
<tr>
<th>MATLAB: Floating-Point Reference Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Vector based</td>
</tr>
<tr>
<td>- Fast running</td>
</tr>
<tr>
<td>- Simple to debug</td>
</tr>
<tr>
<td>- Best possible algorithmic performance</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simulink: Implemented Algorithm Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Scalar design</td>
</tr>
<tr>
<td>- Control signal propagation model</td>
</tr>
<tr>
<td>- Algorithmic performance identical to MATLAB only</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simulink Fixed-Point HDL Capable Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Fixed-Point HDL Capable</td>
</tr>
<tr>
<td>- Meets test points for performance validation</td>
</tr>
</tbody>
</table>

Validation

Model Representation + Scalar Conversion

Fixed-Point Conversion + Model Pipelining
Fixed-Point Conversion

• Fixed-Point conversion tools
  – Fixed-Point Designer
  – Fixed-Point Tool
  – Toolbox mirrored functionality between MATLAB and Simulink
  – MATLAB Function blocks
  – Iterative optimization from high level constraints
Similar features apply to Simulink models
Workflow

Research -> Algorithm Development -> Design Elaboration -> Prototype -> Production
Simulink
Fixed-Point HDL Capable Model

- Fixed-Point HDL Capable
- Meets test points for performance validation

Code Generation
Deployed Model
Optimize
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
Hardware: Transceiver AD9361

HDL Reference Designs: Xilinx Intel

Device Software: Linux Bare Metal OS
**Hardware:**
- Transceiver
- AD9361

**HDL Reference Designs:**
- Xilinx
- Intel

**Device Software:**
- Linux
- Bare Metal OS

**Host Software:**
- IIO Interfaces

**Connection (USB/Ethernet/...):**

**Software Tools:**
- HDL Coder
- Embedded Coder
- MATLAB and Simulink
libIIO

```c
struct iio_context *ctx;
struct iio_device *dev;
struct iio_channel *ch;

/* Error handling is missing */
ctx = iio_create_default_context();
dev = iio_context_get_device(ctx, 0);
ch = iio_device_get_channel(dev, 0);

iio_device_attr_write_longlong(dev, "sample_rate", 1000);
iio_channel_attr_write_double(ch, "scale", 0.525);
```
Deployed Debugging Options

• Standard streaming from base reference design:
  – These will always exists and always must be mapped
  – Can handle high speed data
• External Mode:
  – Utilize for low-speed analysis and tuning
• IIO AXI-MM:
  – Built on top of IIO infrastructure
  – Works at high speed without Embedded Coder
• FPGA Capture:
  – Timing diagram debugging
  – Very useful for debugging IP integrations
% Writers

% Frequency Recovery Loop Bandwidth
w1 = matlabshared.libio.aximm.write('uri',radioIP);
w1.AddressOffset = hex2dec('100');
w1.HardwareDataType='int16';

% Equalizer Step Size
w2 = matlabshared.libio.aximm.write('uri',radioIP);
w2.AddressOffset = hex2dec('104');
w2.HardwareDataType='int16';

% IQ Scope Selection
w3 = matlabshared.libio.aximm.write('uri',radioIP);
w3.AddressOffset = hex2dec('108');
w3.HardwareDataType='int8';

% Debug Status Signal Selection
w4 = matlabshared.libio.aximm.write('uri',radioIP);
w4.AddressOffset = hex2dec('10c');
w4.HardwareDataType='int8';
Targeting Custom Hardware
Workflow
Moving to the Production Design

- Path from supported development hardware to production hardware
- Documented process from MathWorks called **Board Support Packages**
  - ADI maintains examples for different board variants
- Provides same connectivity in final production hardware back to MATLAB
• Hardware Support Package (HSP)
  – Standard development kits
  – Fixed reference designs
  – Fixed board registration
  – End targets:
    • MATLAB/Simulink
    • ARM codegen application

• Board Support Package (BSP)
  – Custom boards (ex: PackRF board)
  – Custom reference designs
  – Custom registration API
  – End target:
    • Up to user
    • TUN/TAP in our design
• **Without custom BSP**
  – Back & forth between teams
  – Interface definition mismatch probability
  – Prone to errors between steps

• **With custom BSP**
  – Same person does all the steps
  – Stay in the same flow as for the previous designs
  – Reduced integration time

**Generate IP from Simulink model**

**Integrate IP in HDL design**

**Generate Linux image**

**Test new design on hardware**

**Generate IP from Simulink model**

**Integrate IP in HDL design using WA flow**

**Generate Linux image using WA flow**

**Test new design on hardware**
Process is well documented with MATLAB

ADI BSP is an example of using this workflow

HSPs from MathWorks are also examples of this workflow
Where can I get the code?
• Thank You!

• Questions?