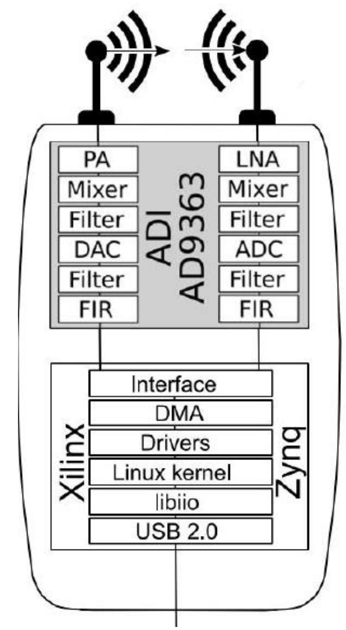
Digital Systems for Telecommunications

Using the Software Defined Radio
Adalm PLUTO

The Adalm-PLUTO device

- The Pluto SDR includes:
  - An analog RF section (atennna, RF filters, input mux, LNA, gain, attenuation, mixer)
  - Analog baseband part (analog filters, ADC or DAC) is implmented in the AD9363, Integrated RF Agile Transceiver
  - Digital signal processing unit for dedicated RF processing
  - Xilinx Zynq' FPGA for further signal processing
  - ARM cortex 9 processor for embedded LINUX OS
- Antenna and RF filters are expected to be done outside the Pluto SDR and are the responsibility of the end user.

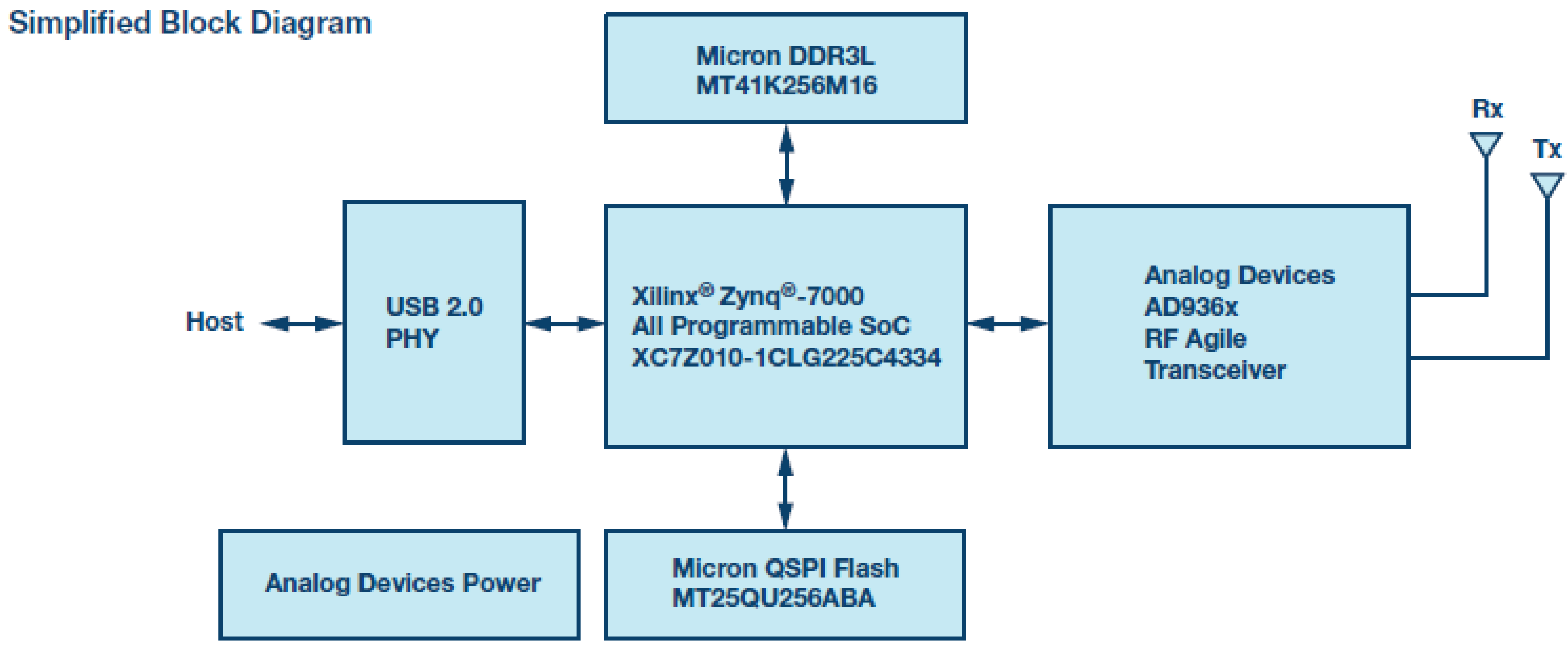


The Adalm-PLUTO specs

- TX/RX side:
  - Center frequency: 300 MHz-3,8 GHz, 70 MHz – 6 GHz via SW modification and reduced specifications (2.4 Hz LO step size).
  - Channel bandwidth 200 kHz – 20 MHz
  - Sample rate: 65 kSPS–61 MSPS (5 Hz sample rate step)
  - 12-bit ADCs and DACs
- Input/Output
  - USB 2 OTG (480 Mbits/second), device mode; behave as a network device as well
  - libiio library, for transfering IQ data from/to the RF device to the host

Specifications	Typical
Power	
DC Input (USB)	4.5 V to 5.5 V
Conversion Performance and Clocks	
ADC and DAC Sample Rate	65.2 kSPS to 61.44 MSPS
ADC and DAC Resolution	12 bits
Frequency Accuracy	±25 ppm
RF Performance	
Tuning Range	325 MHz to 3800 MHz
Tx Power Output	7 dBm
Rx Noise Figure	<3.5 dB
Rx and Tx Modulation Accuracy (EVM)	–34 dB (2%)
RF Shielding	None
Digital	
USB	2.0 On-the-Go
Core	Single ARM Cortex®-A9 @ 667 MHz
FPGA Logic Cells	28k
DSP Slices	80
DDR3L	4 Gb (512 MB)
QSPI Flash	256 Mb (32 MB)
Physical	
Dimensions	117 mm × 79 mm × 24 mm 4.62" × 3.11" × 0.95"
Weight	114 g
Temperature	10°C to 40°C

The Adalm-PLUTO – block diagram



The Adalm-PLUTO – PCB

TX/RX SMA connectors

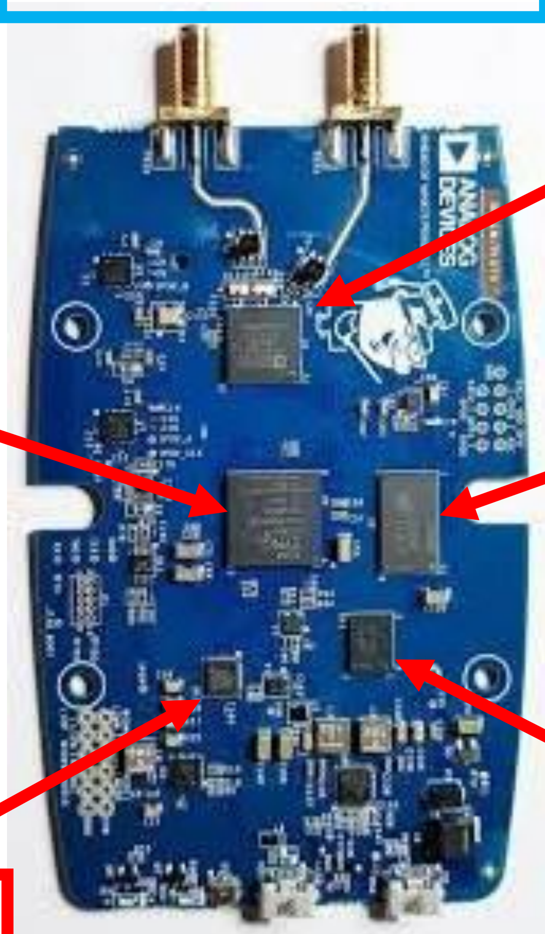
AD9363 RF Frontend

Xilinx ZYNQ 7000

DDR3L VM

SPI Flash NVM

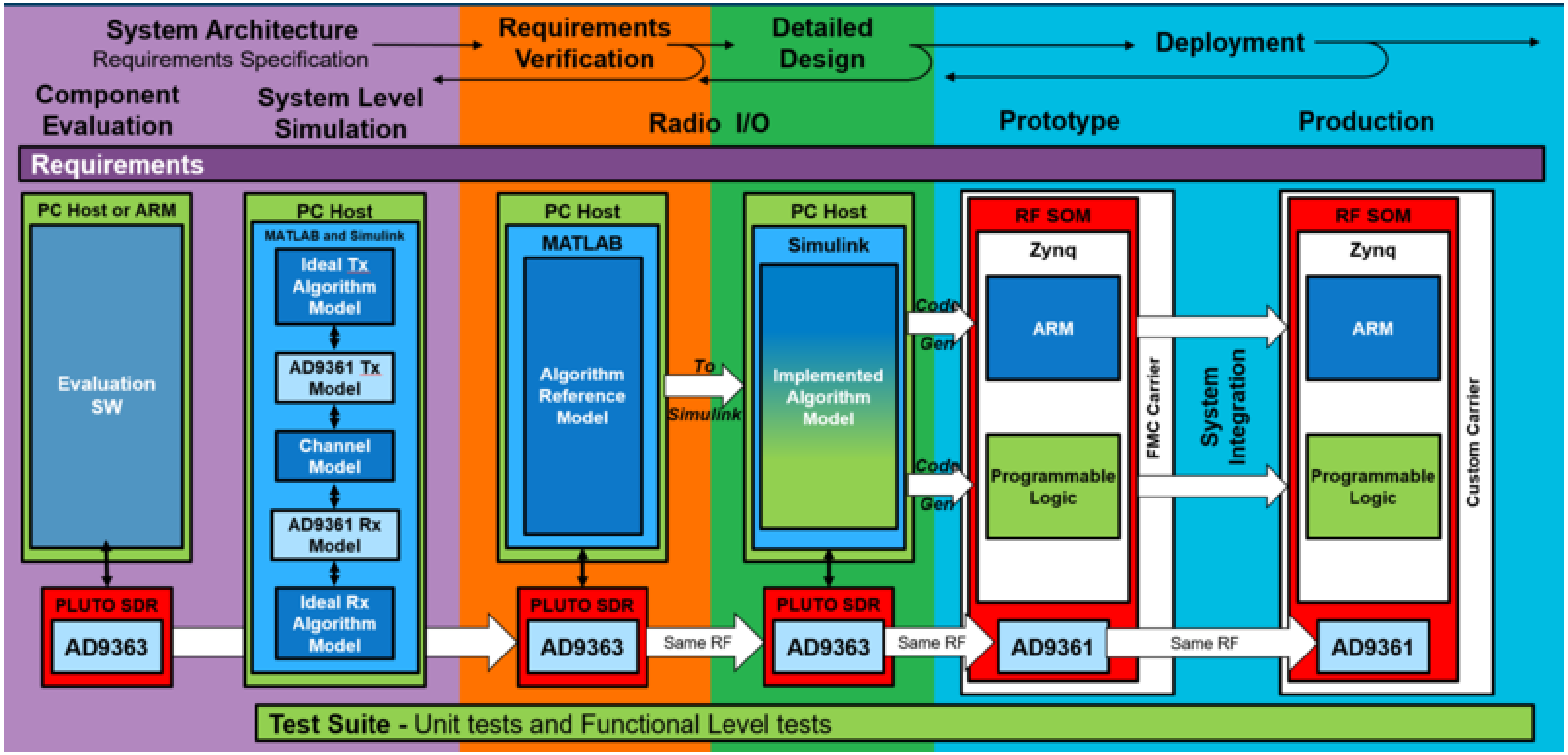
USB 2.0 Phy



USB connectors (OTG & Supply)

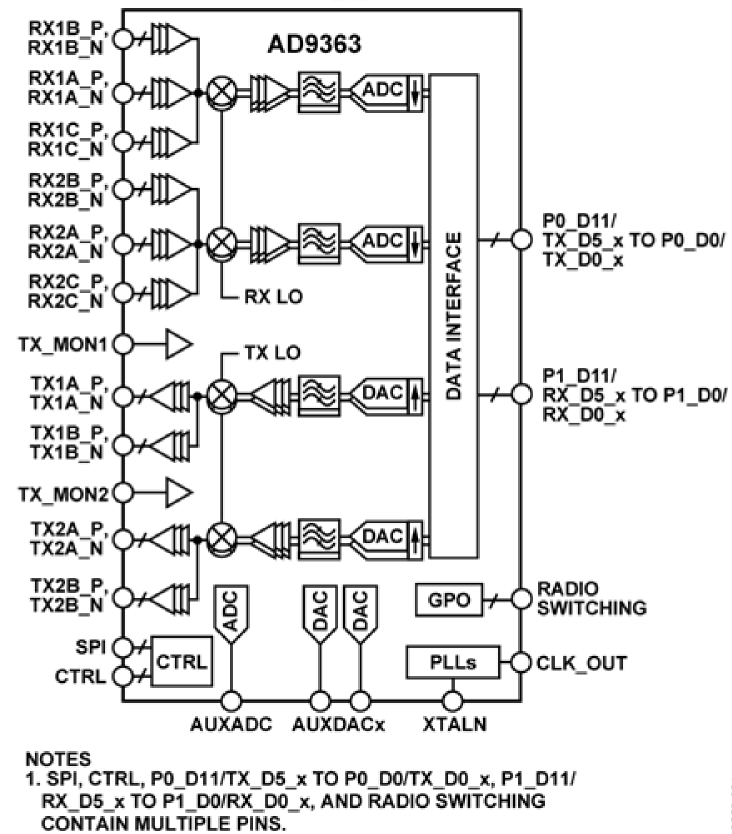
Design flow

- The journey from simulation to complete standalone deployment



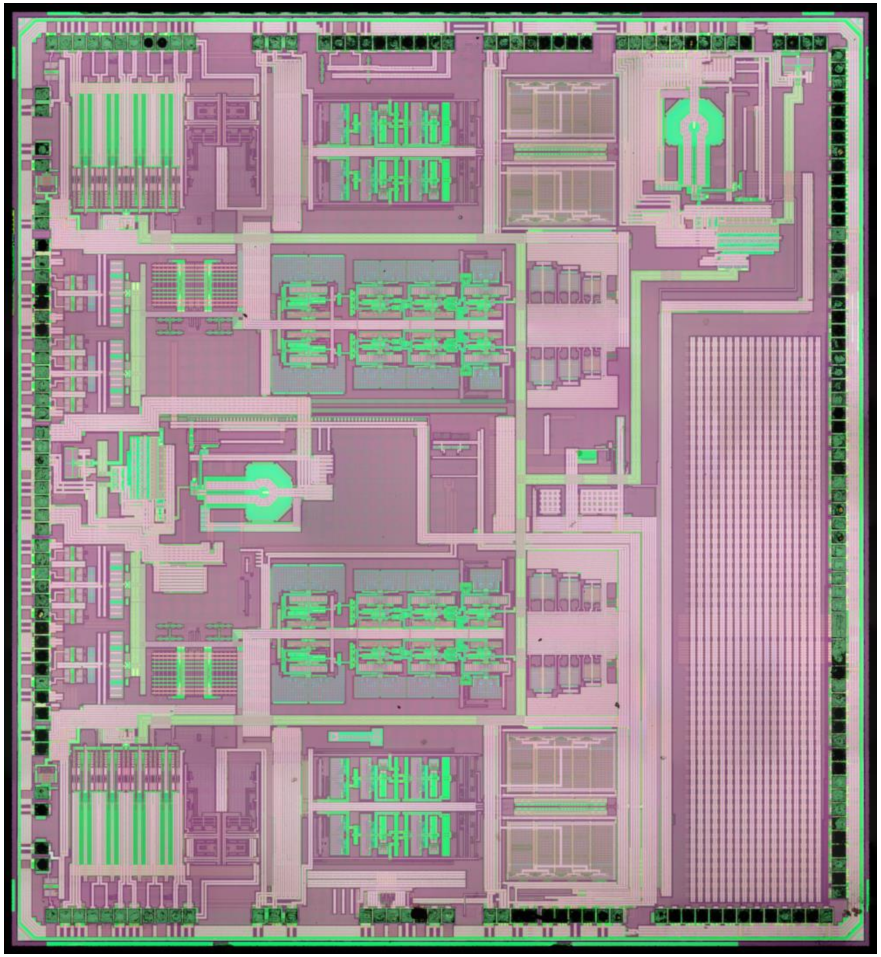
RF front end

- The Analog Devices AD9363/AD9364 single chips combine an RF front end with a flexible mixed-signal baseband section and integrated frequency synthesizers
  - The receiver section contains all blocks necessary to receive RF signals and convert them to digital data that is usable by a BBP.
  - The transmitter section implement a zero-IF (ZIF) system while sharing a common frequency synthesizer.
  - Digital interpolation/decimation filters to up/down convert from the digital baseband rate (64.11MSPS max) to the actual ADC (640MSPS) or DAC (320MSPS) rates.



NOTES
1. SPI, CTRL, P0_D11/TX_D5_x TO P0_D0/TX_D0_x, P1_D11/ RX_D5_x TO P1_D0/RX_D0_x, AND RADIO SWITCHING CONTAIN MULTIPLE PINS.

RF front end



AD9361

Die size:

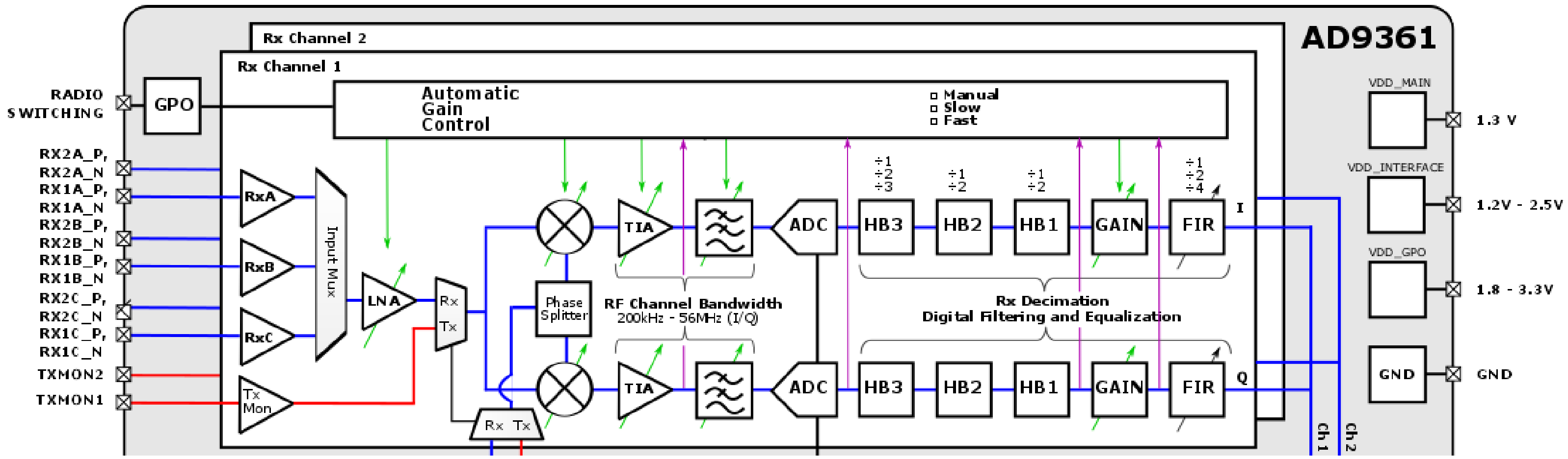
- 4336x4730 µm

Technology node:

- 65nm

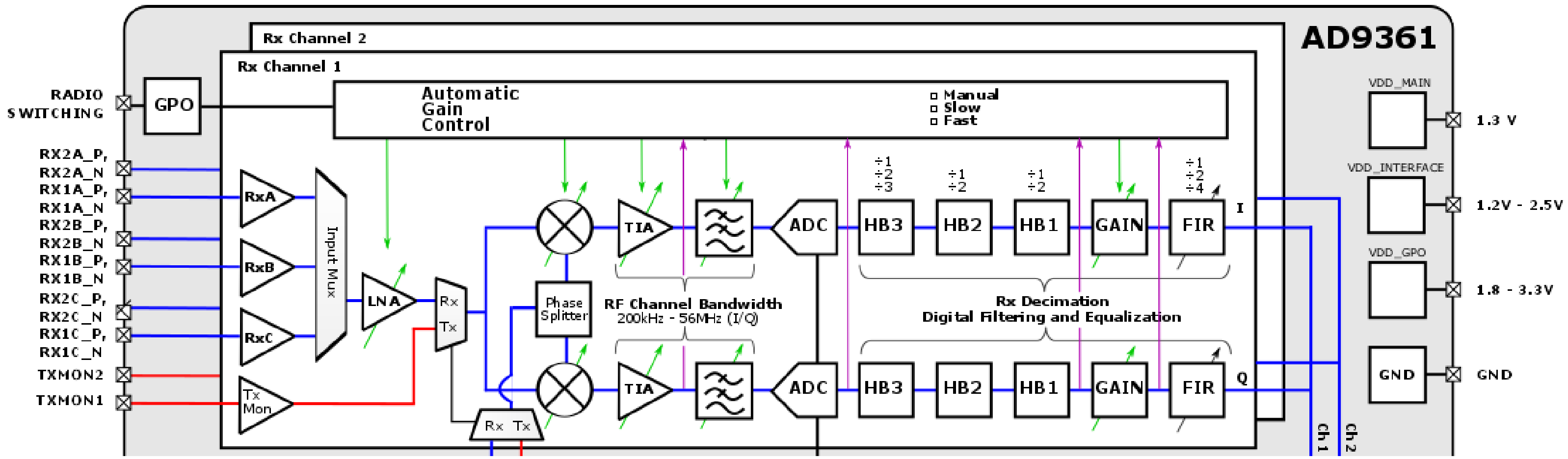
AD936x – Receiver path

- The baseband RX signal path is composed of two programmable analog low-pass filters, a 12-bit ADC, and four stages of digital decimating filters; I and Q paths are identical to each other.
  - Transimpedance amplifier (TIA) behaves as a I-order LPF; it is followed by a III-orde Butterworth LPF
  - The 12-bit, sigma-delta (Σ-Δ) ADCs offer adjustable sample rates that produce data streams from the received signals.
  - Each of the four decimating filters can be bypassed and the corner frequency is programmable.



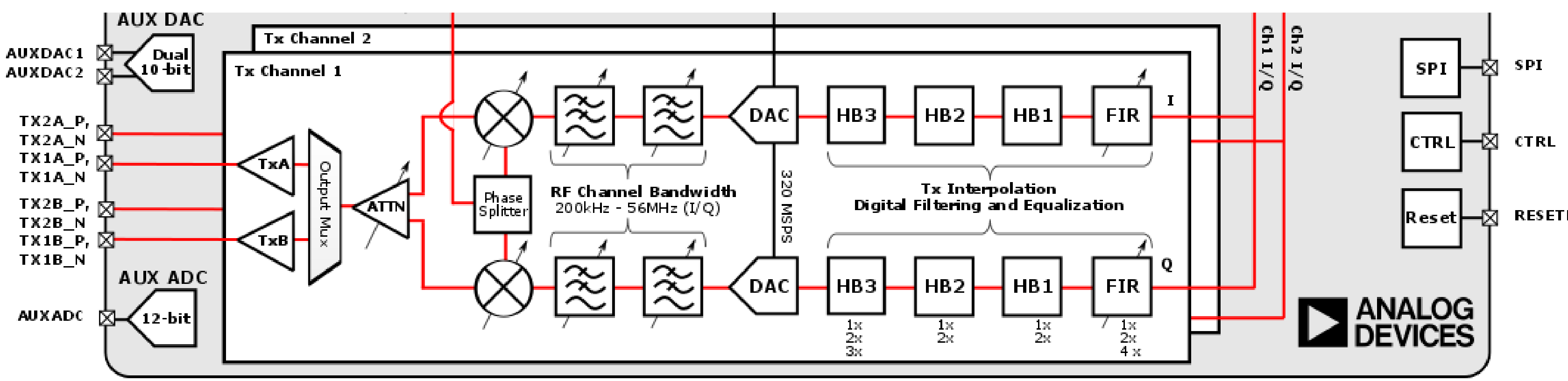
AD936x – Receiver path

- The digitized signals can be conditioned further by a series of decimation filters and a fully programmable 128-tap FIR filter with additional decimation settings.
  - The sample rate of each digital filter block is adjustable by changing decimation factors to produce the desired output data rate.
- Gain control is achieved by a preprogrammed gain index map that distributes gain among the blocks for optimal performance at each level; RSSI measurement and dc offset tracking is carried out.
  - An internal AGC (fast or slow mode) or manual control can be used.



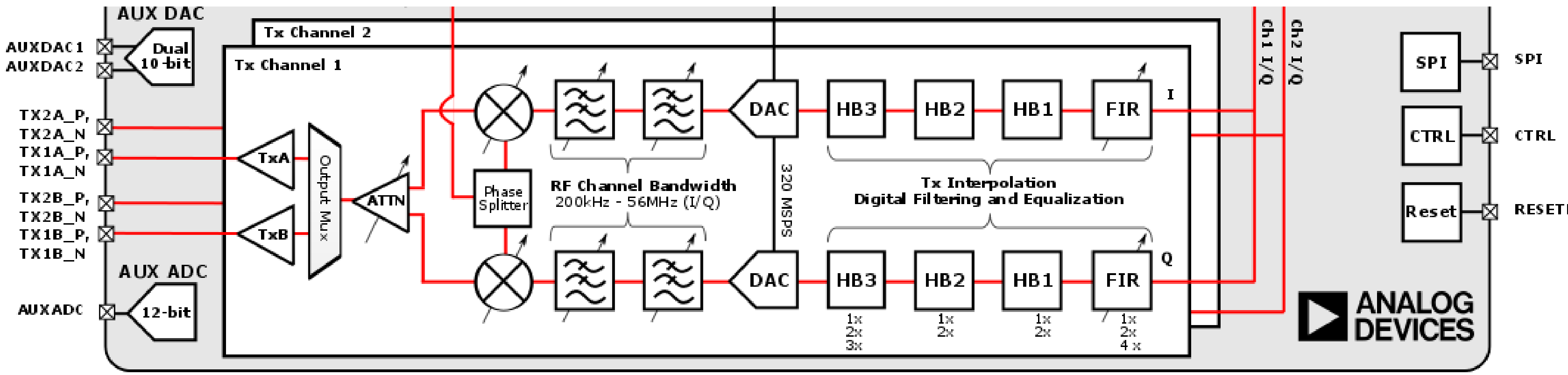
AD936x – Transmitter path

- The TX signal path receives 12-bit 2s complement data in I-Q format from the digital interface, and each channel (I and Q) passes this data through a fully programmable 128-tap FIR filter with interpolation options.
- The FIR output is sent to a series of additional interpolation filters that provide additional filtering and data rate interpolation prior to reaching the 12-bit DAC.
- The FIR filter, and of the three interpolating filters can individually be controlled and bypassed if desired.



AD936x – Transmitter path

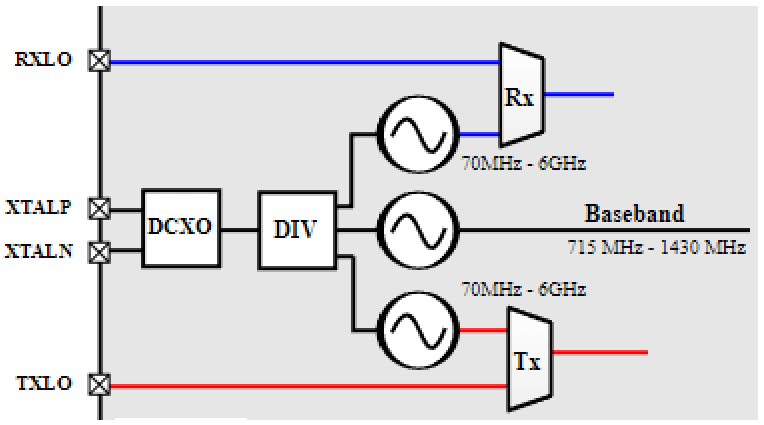
- Each 12-bit DAC has an adjustable sampling rate.
- The DAC's analog output is passed through two low pass filters (to remove sampling artifacts) prior to the RF mixer. The corner frequency for each low-pass filter is programmable.
- At this point, the I and Q signals are recombined and modulated on the carrier frequency for transmission to the output stage. The combined signal also passes through analog filters that provide additional band shaping, and then the signal is transmitted to the output amplifier.



AD936x – Transmitter path

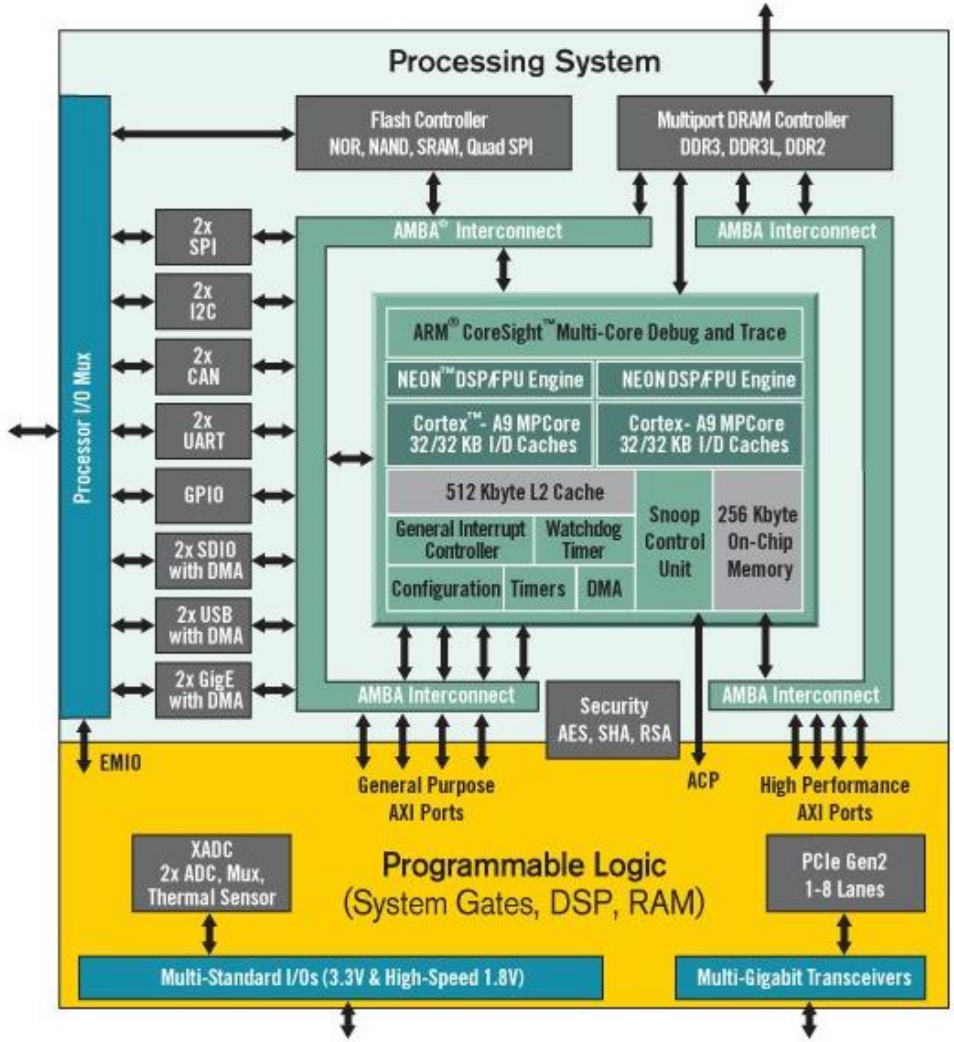
- Each transmit channel provides a wide attenuation adjustment range with fine granularity to help designers optimize signal-to-noise ratio (SNR).
- Note that both the I and the Q paths are schematically identical to each other.
- Self-calibration circuitry is built into each transmit channel to provide automatic real-time adjustment.

- RF Tx and Rx carrier signals and baseband ADC and DAC clock signals are derived from an internal Digitally-Compensated Crystal Oscillator (DCXO) or from external sources



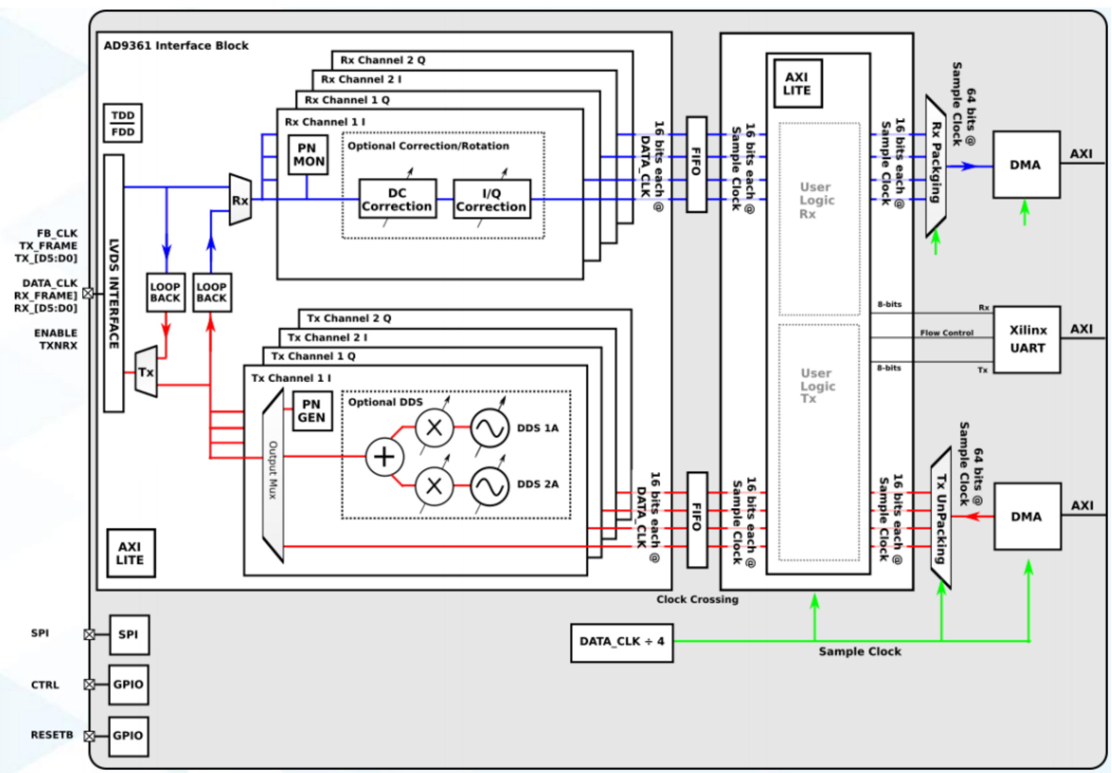
The Zynq-7000 SoC

- Once the data is digitized it is passed to the Xilinx Zynq System on Chip
- The Zynq-7000 family includes an FPGA for flexibility and scalability (eFPGA approach)
  - Integrated ARM Cortex-A9 based processing system (PS) and programmable logic (PL).
- The Zynq is used in the Pluto SDR as the main controller



The FPGA role in the ADALM-Pluto

- Provides two-tones DDS generators
- Provides DC and I/Q unbalance corrections
- Implements TX and RX FIFO buffers and additional loopback paths
- Manages buses towards the CPU

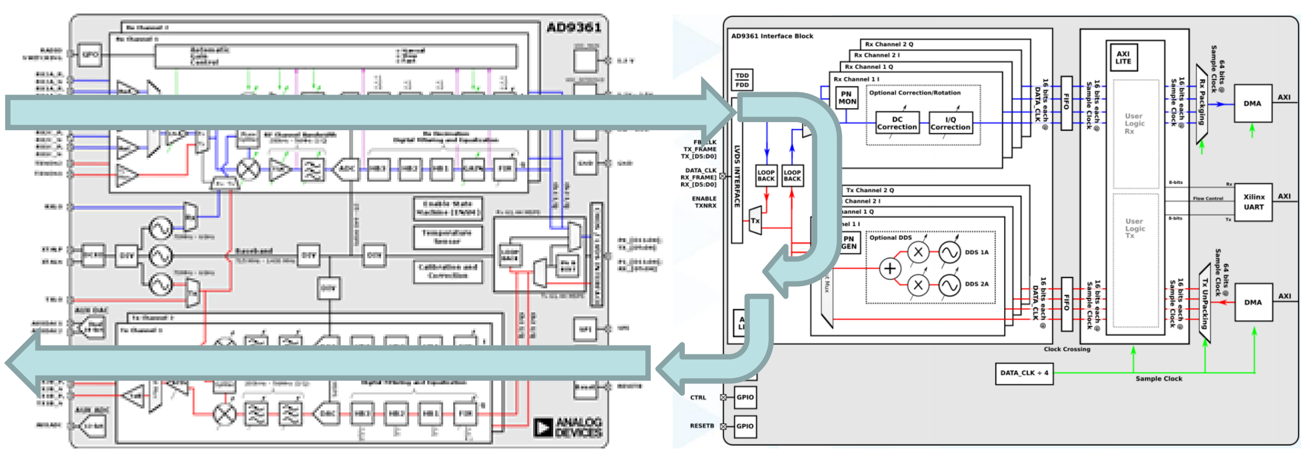


AXI is part of ARM AMBA, a family of master-slave micro controller buses allowing for memory mapped peripherals

- *AXI-Lite is for simple, low-throughput memory-mapped communication (e.g., to and from control and status registers)*

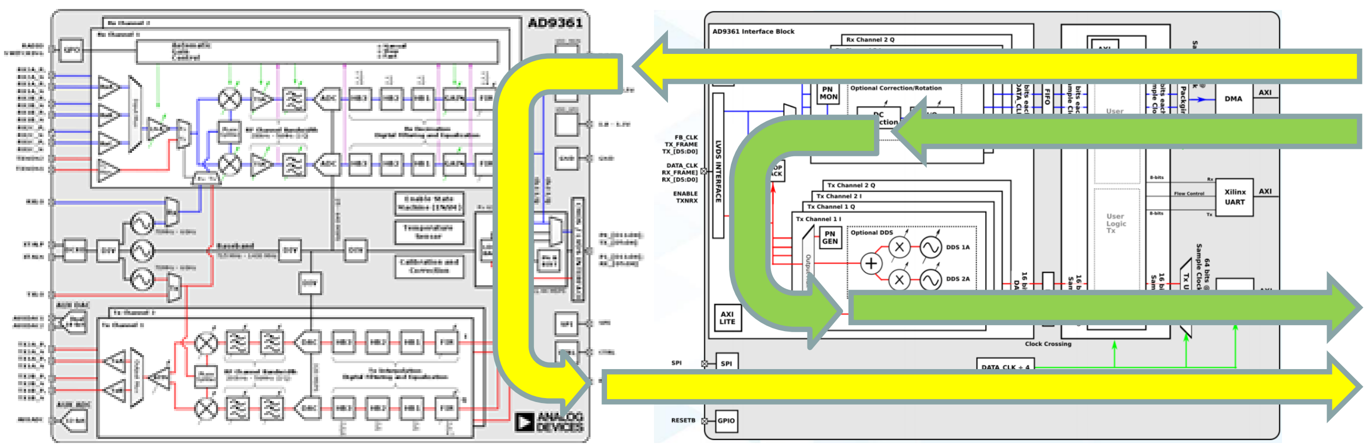
Loopbacks

- Loopback paths are for testing and development
- Investigate hardware RF issues without software or FPGA interaction
  - Signal still digitalized, still goes through filters; will see RF impairments due to AD936X setup and output is baseband copy of input
  - LO (Rx/Tx) interaction will occur if the same frequency – use different frequencies



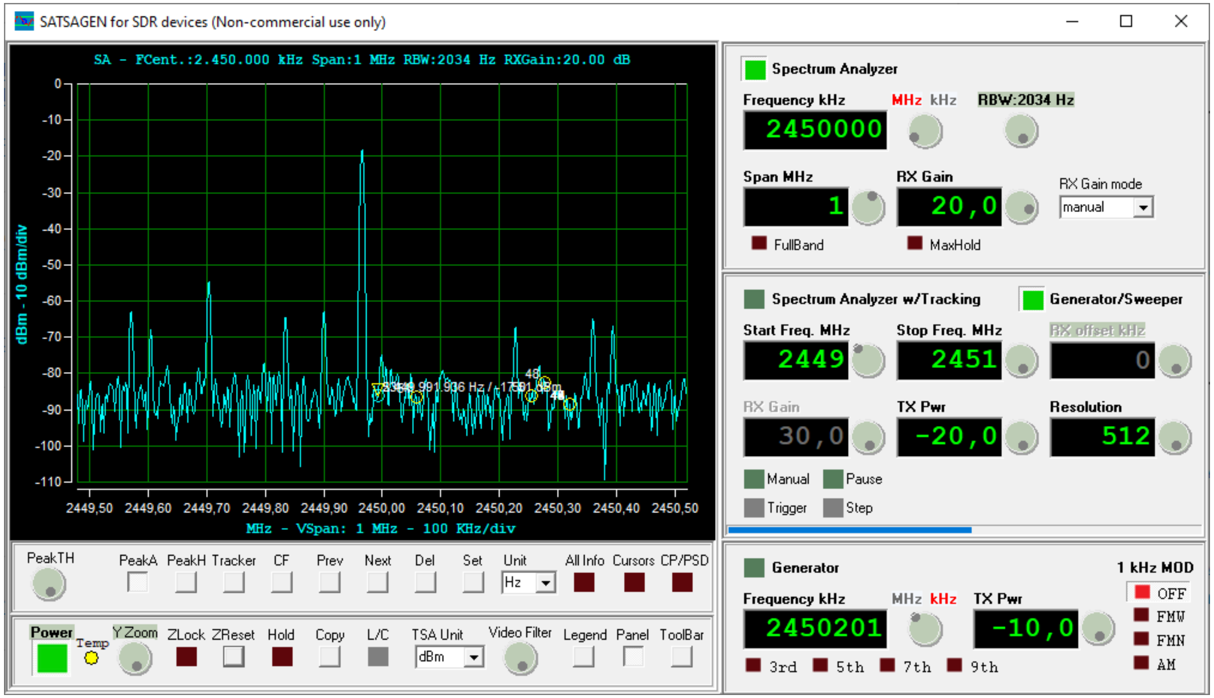
Loopbacks

- Loopback paths are for testing and development
- Investigate hardware RF issues without software or FPGA interaction
  - Signal still digitalized, still goes through filters; will see RF impairments due to AD936X setup and output is baseband copy of input
  - LO (Rx/Tx) interaction will occur if the same frequency – use different frequencies
- Investigate digital FPGA or software algorithm issues without RF impairments



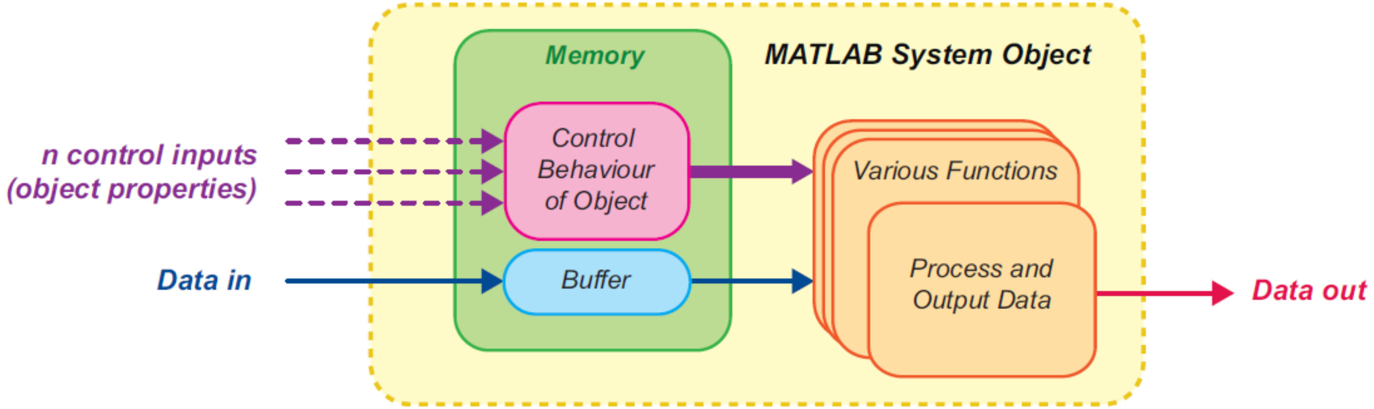
The SATSAGEN for SDR

- Windows APP implementing an IIO client
- Transform the Pluto into:
  - Sweep SA and RT SA
  - RF Generator
  - Impedance analyzer and VNA



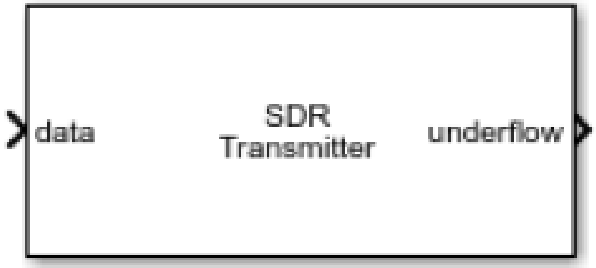
Adalm-PLUTO in Matlab

- The SDR is abstracted into a «System Objects»:
  - sdrtx('Pluto’) and sdrrx('Pluto');
- MATLAB (which is based on Java) is an OO language, using a specific type of object called a System Object that allows dynamic systems to be initialised both in MATLAB and Simulink.
  - These dynamic systems output different signals depending on the values of their inputs, which can change over time.
  - They contain memory which holds their parameters, current state and past behaviour, used during the next computational step.



Adalm-PLUTO in Simulink

- The ADALM-PLUTO radio transmitter block prepares input data for transmission
  - To detect underflow during the transmission of radio signals, check the underflow output port on the SDR transmitter block.
- The ADALM-PLUTO radio receiver block furnishes output data to process data received over the air
  - To detect overflows during the reception of radio signals, check the overflow output port on the SDR receiver block.

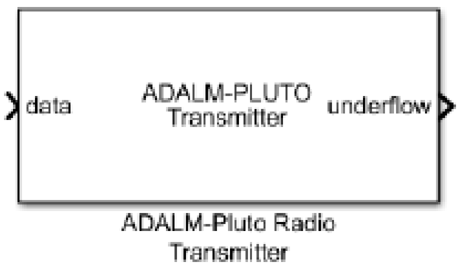


ADALM-Pluto Radio
Transmitter

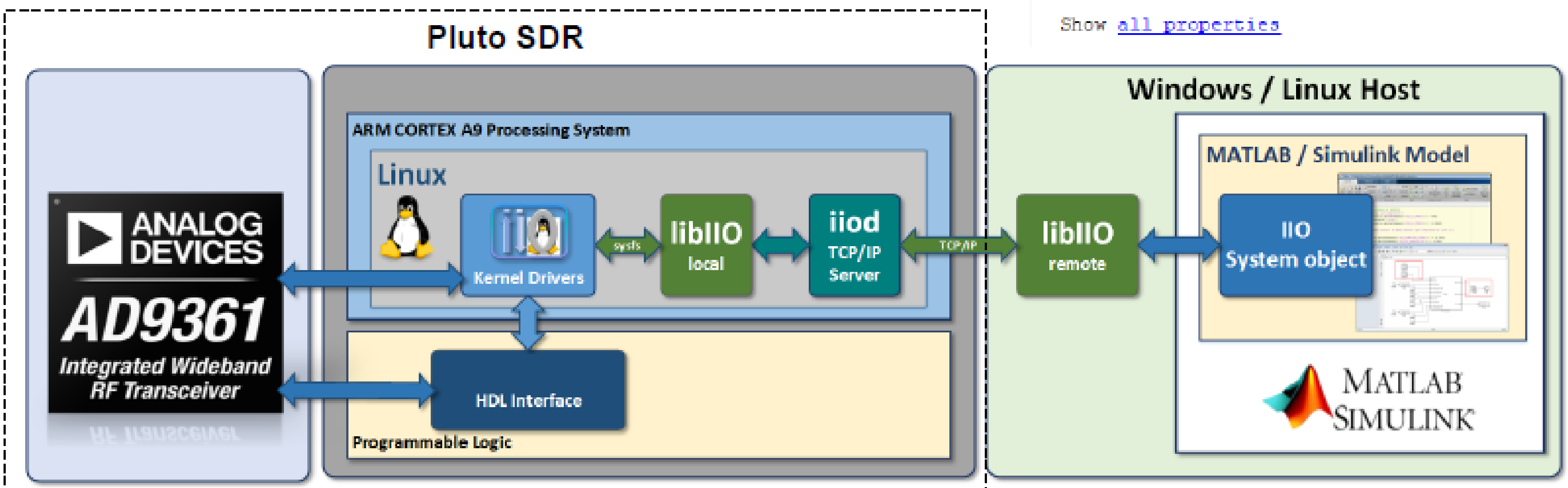
ADALM-Pluto Radio
Receiver

What happens under the hood

- Try typing: sdrrx('Pluto') and sdrtx('Pluto')

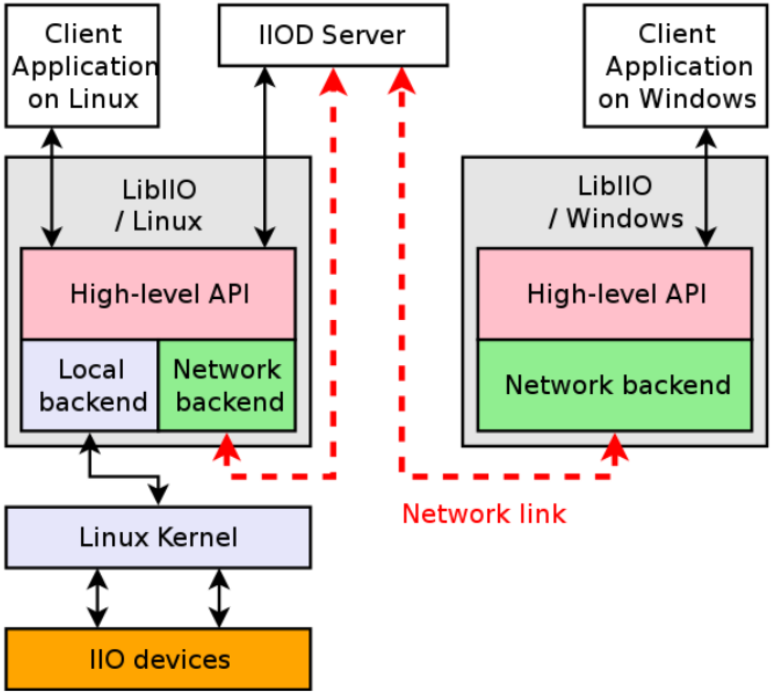



```
rx =

  comm.SDRRxPluto with properties:

  Main
              DeviceName: 'Pluto'
                 RadioID: 'usb:0'
         CenterFrequency: 2.4000e+09
              GainSource: 'AGC Slow Attack'
          ChannelMapping: 1
     BasebandSampleRate: 1000000
          OutputDataType: 'int16'
         SamplesPerFrame: 20000
         EnableBurstMode: false
  ShowAdvancedProperties: false

  Show all properties
```



The libiio

- The Pluto is accessed via the libiio framework
- Provides hardware abstraction layer
  - Above Linux Kernel drivers (where low-level & low-latency access to hardware occur)
- Support for network backend (remote clients connected via TCP/IP; USB; Serial)
  - Backend takes care of low-level communication details
  - Provide the same API for applications
  - Transparent from the applications point of view

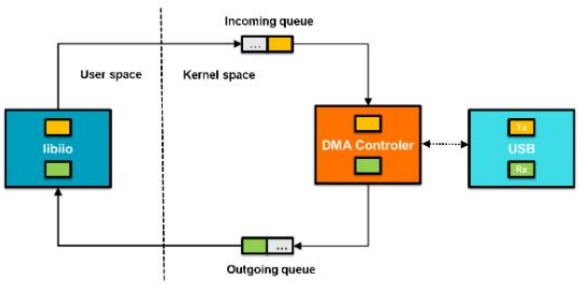


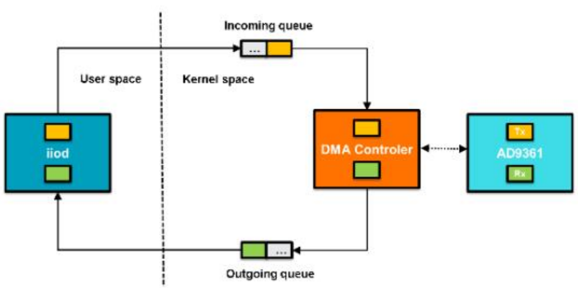
SDRToolboxes that are required to use MATLAB with the PlutoSDR:

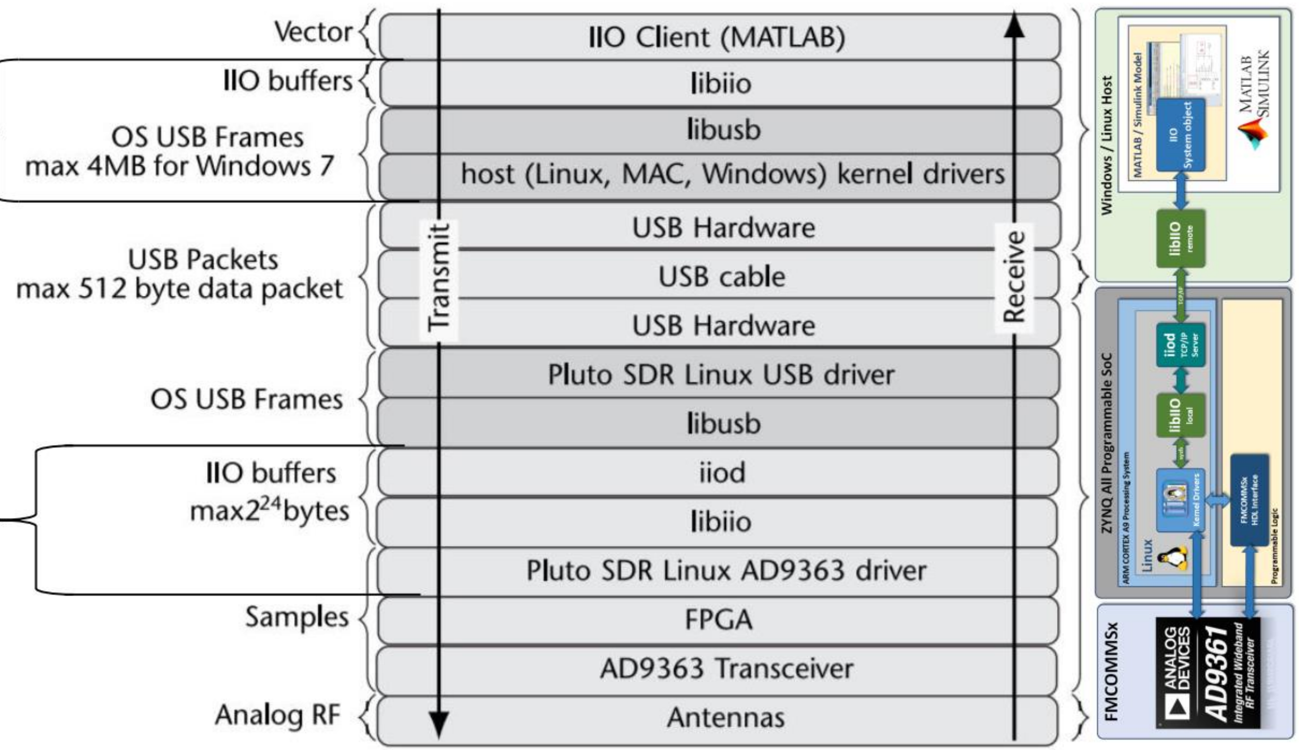
- DSP System Toolbox
- Signal Processing Toolbox
- Communications System Toolbox
- Communications Toolbox Support Package for ADALM-PLUTO Radio

The stack

- Abstraction does not come for free
  - Many levels
  - Delays?

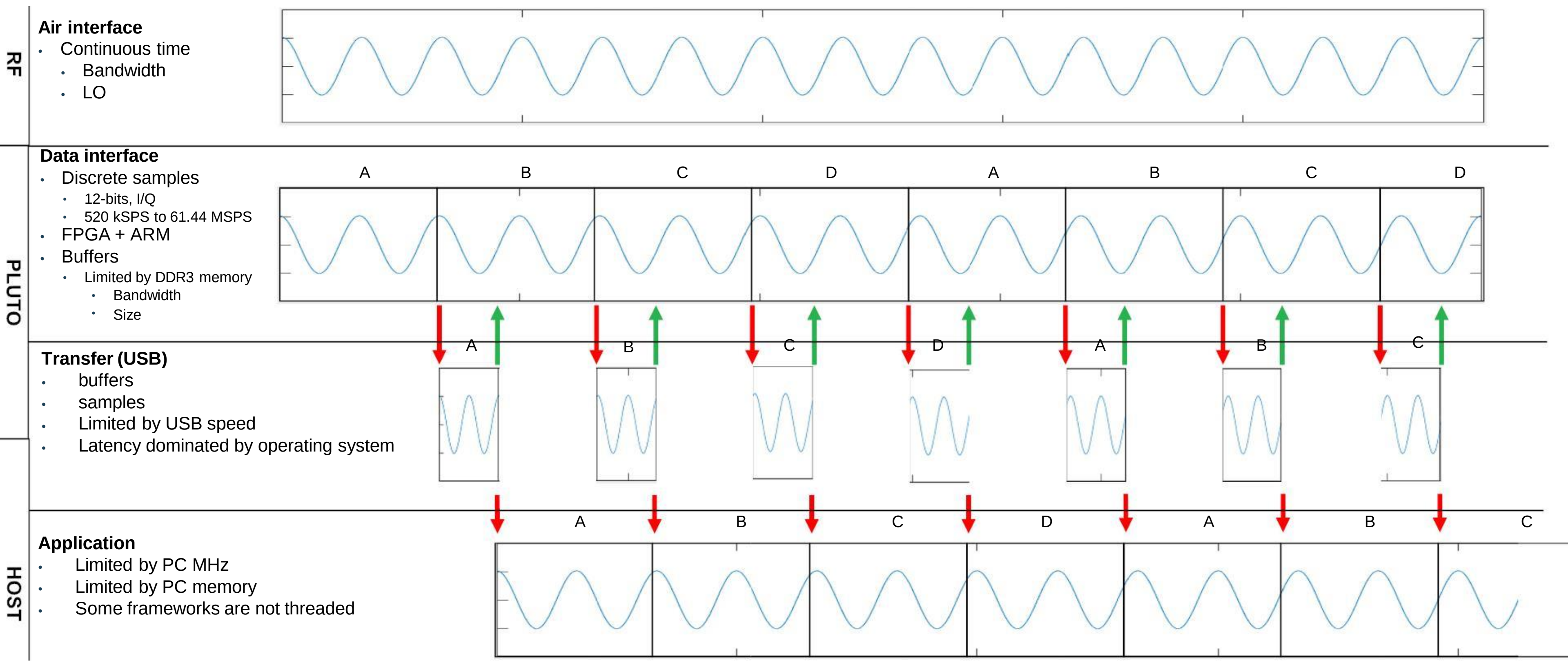






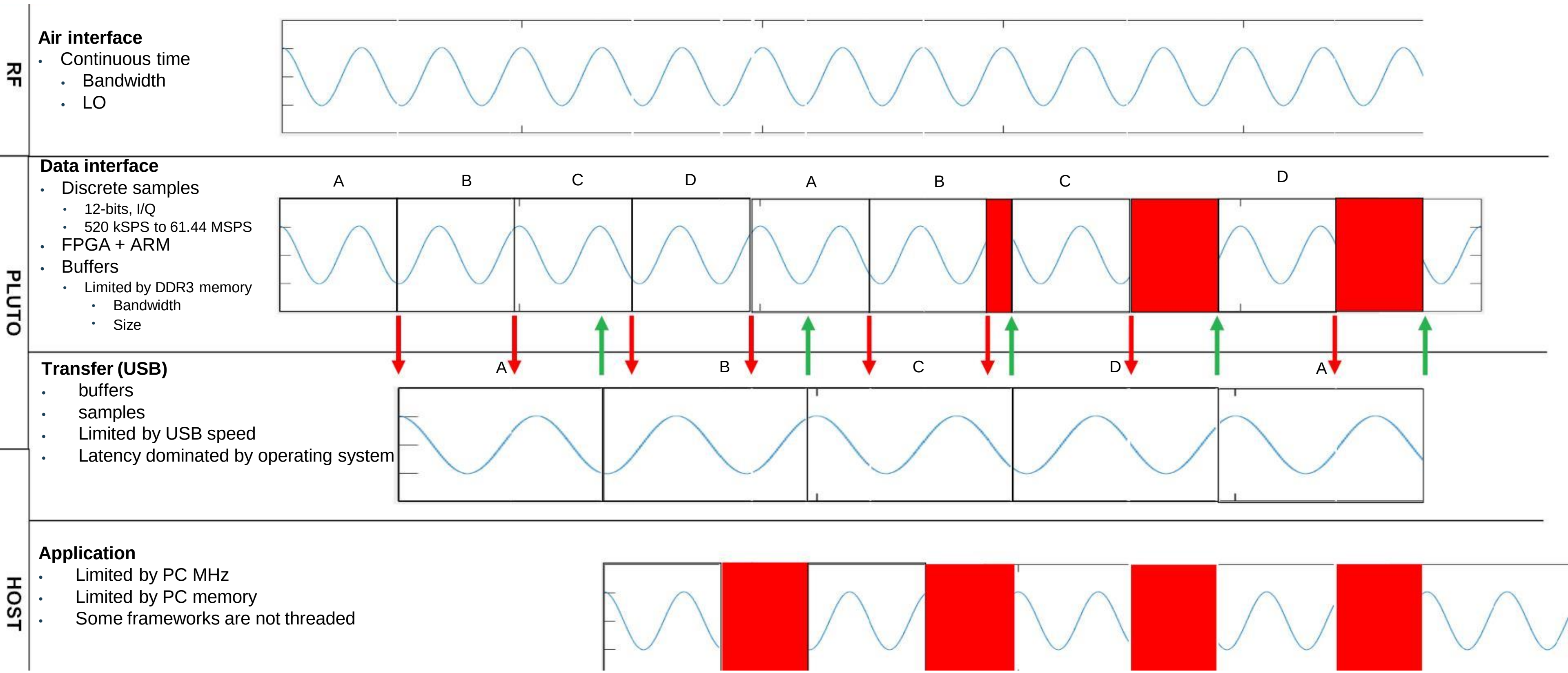
Buffer management

- Low data rates, low host DSP – real time



Buffer management

- High data rates, high DSP – lost samples, first n (=4) buffers are continuous

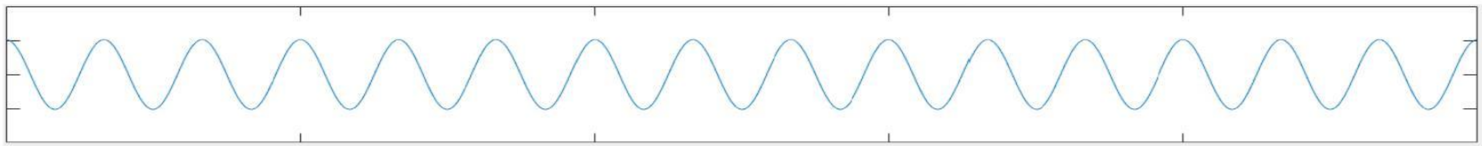


Buffer management

- Pause = stale data (backpressure management)

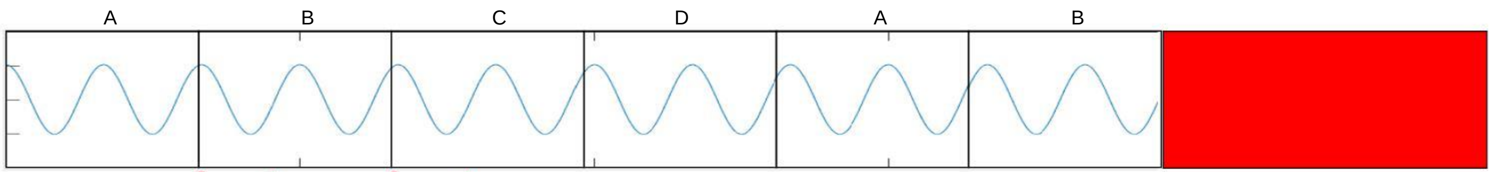
RF

Air interface
- Continuous time
  - Bandwidth
  - LO

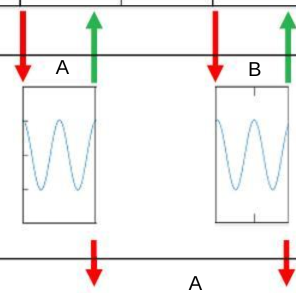


PLUTO

Data interface
- Discrete samples
  - 12-bits, I/Q
  - 520 kSPS to 61.44 MSPS
- FPGA + ARM
- Buffers
  - Limited by DDR3 memory
    - Bandwidth
    - Size

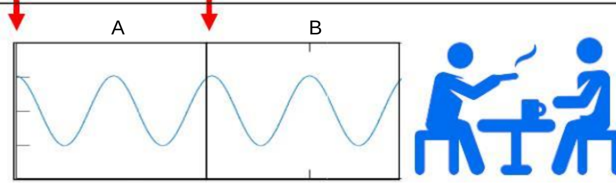


Transfer (USB)
- buffers
- samples
- Limited by USB speed
- Latency dominated by operating system



HOST

Application
- Limited by PC MHz
- Limited by PC memory
- Some frameworks are not threaded



The ISM band

- The industrial, scientific and medical (ISM) radio bands are radio bands (portions of the radio spectrum) reserved internationally for the use of radio frequency (RF) energy for industrial, scientific and medical purposes other than telecommunications.
  - Services operating within these bands must accept harmful interference which may be caused by these applications and they are subject to specific rules (max power and/or Duty-Cycle)

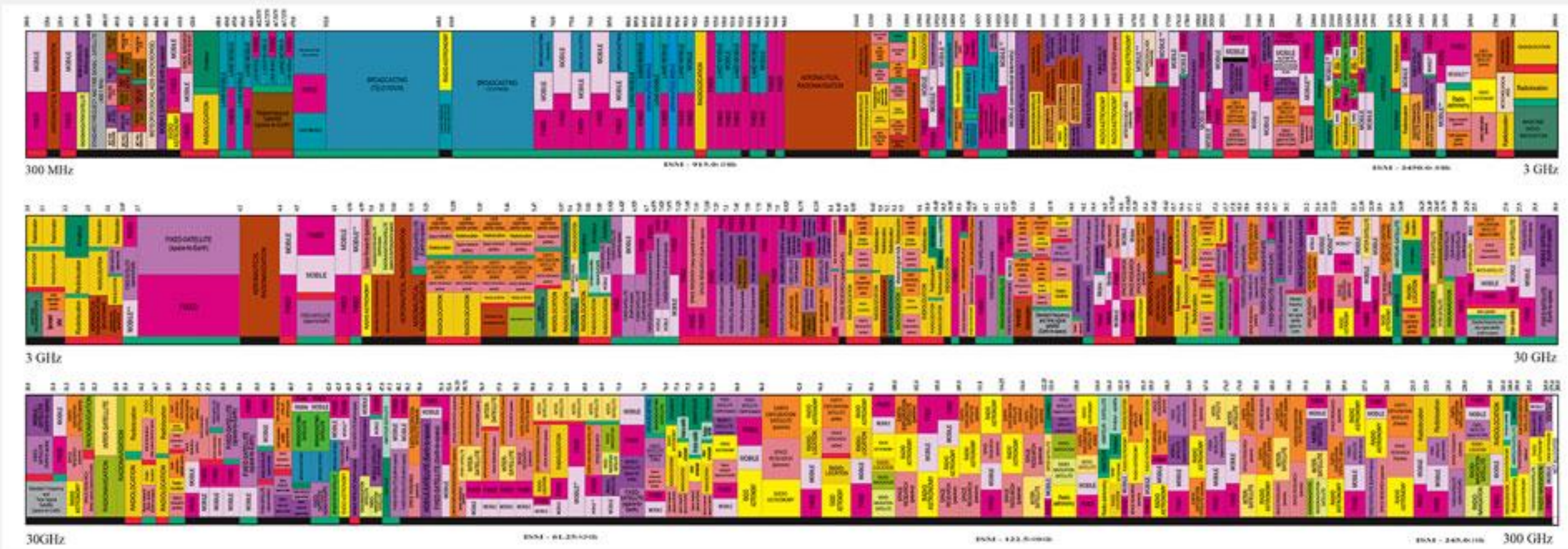
863-870 MHz	**SRD**
2.4 – 2.5 GHz	**ISM**
5.725-5.875 GHz	**ISM**

Use the ISM bands fort TX! Nothing but those.

- Anytime the Pluto SDR is powered on, it begins to operate even if the user did not intend to. When powered on Pluto SDR will transmit!
- There are two ways to reduce the self interference when receiving:
  1. instantiate a TX System object and write a vector of zeros
  2. shift the LO of the TX to a frequency beyond the RX bandwith.

Spectrum allocation

- Frequency allocation (or spectrum allocation or spectrum management) is the allocation and regulation of the electromagnetic spectrum into radio frequency bands, which is normally done by governments in most countries



United States Department of Commerce / Public Domain

This image depicts a part of the current spectrum allocation in the United States. The complexity indicates the challenge that international harmonization poses.