altera™
An Intel Company

November 13–14, 2024 | Online

Simulink Models for Algorithms to Silicon

Dr. Deepak Dasalukunte, Technology Pathfinding, Altera

Collaborators: Technology Pathfinding team, Altera
MathWorks.

MATLAB EXPO

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA).
The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.
Distribution Statement "A" (Approved for Public Release, Distribution Unlimited

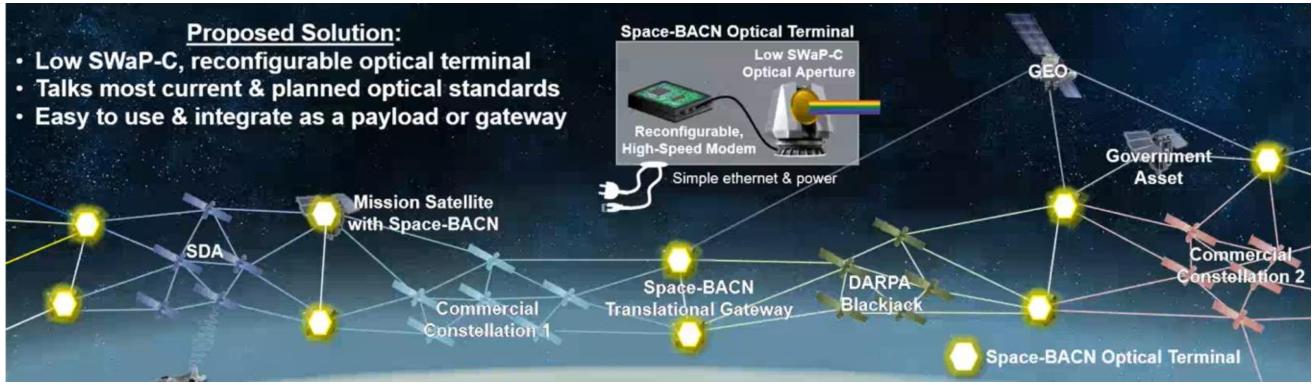
Outline

- Project overview
- Simulink based modeling for hardware – Problem statement
- Proposed methodology & advantages
- Receiver block case-study

Project overview

intel INTELS-BACN
Intel Network Terminal Enabling Low-Cost Low-Power SPACE Based Adaptive Communications Node

- Space BACN: Space-Based Adaptive Communications Node
- Swiss army knife optical terminal interconnecting commercial and Government constellations.
- 100^3 goal: 100Gbps – 100W – $100k for the terminal
- Reconfigurable modem under Technical Area 2 (TA2) of DARPA's Space-BACN project.
  - Power budget of 40W



Proposed Solution:
- Low SWaP-C, reconfigurable optical terminal
- Talks most current & planned optical standards
- Easy to use & integrate as a payload or gateway

Space-BACN Optical Terminal
Low SWaP-C Optical Aperture
Reconfigurable, High-Speed Modem
Simple ethernet & power
GEO
Government Asset
Mission Satellite with Space-BACN
SDA
Commercial Constellation 1
Space-BACN Translational Gateway
DARPA Blackjack
Commercial Constellation 2
Space-BACN Optical Terminal

Source: https://www.darpa.mil/work-with-us/space-based-adaptive-communications-node

Key challenges

- Reconfigurable modem with multi-standard support
  - OpenZR (optical transport networks): high data rates (>100Gbps).
  - LCRD, SDA: long range, low data rates (≤10 Gbps).

- Wide range of data rates (1 – 100Gbps) over long distances (65,000km – 2,500km)

- Modulation waveforms: OOK/PPM + QAM + DPSK
  - Appropriate receiver processing algorithms to be developed.

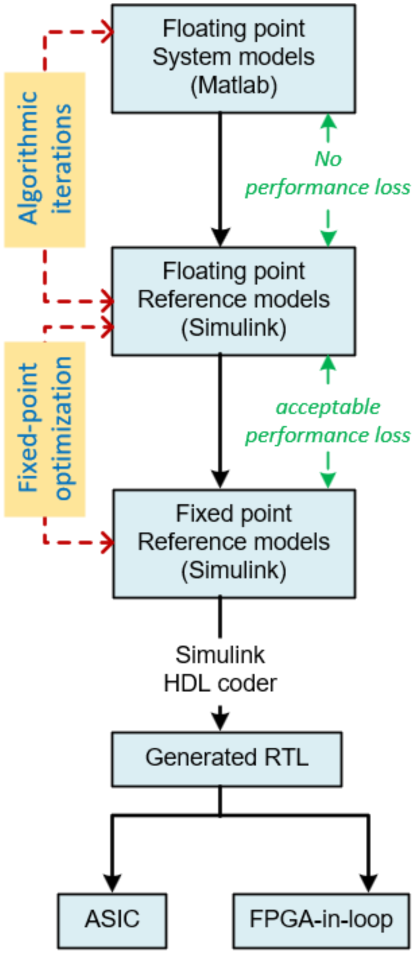
https://openzrplus.org/
https://www.nasa.gov/mission_pages/tdm/lcrd/index.html
https://www.sda.mil/wp-content/uploads/2022/04/SDA-OCT-Standard-v3.0.pdf

Algorithms to Silicon – Problem statement

- System model (MATLAB) is developed to validated the algorithms.
  - No consideration on computational complexity or implementation aspects

- A reference model (Simulink) is developed considering implementation aspects.
  - Floating-point to match with System model
  - Fixed-point optimized model for RTL generation

- Algorithm development cycle can eat into timelines of reference model development and fixed-point conversion.
  - can add to delays to hardware development cycle (RTL).



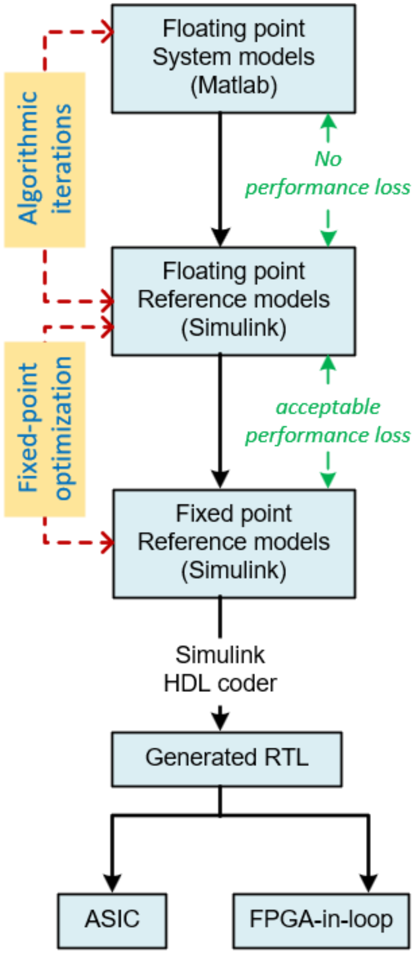
Floating point System models (Matlab)
Algorithmic iterations
No performance loss
Floating point Reference models (Simulink)
Fixed-point optimization
acceptable performance loss
Fixed point Reference models (Simulink)
Simulink HDL coder
Generated RTL
ASIC
FPGA-in-loop

Algorithms to Silicon – Problem statement

- The Space-BACN modem hardware is an entirely new communication system.

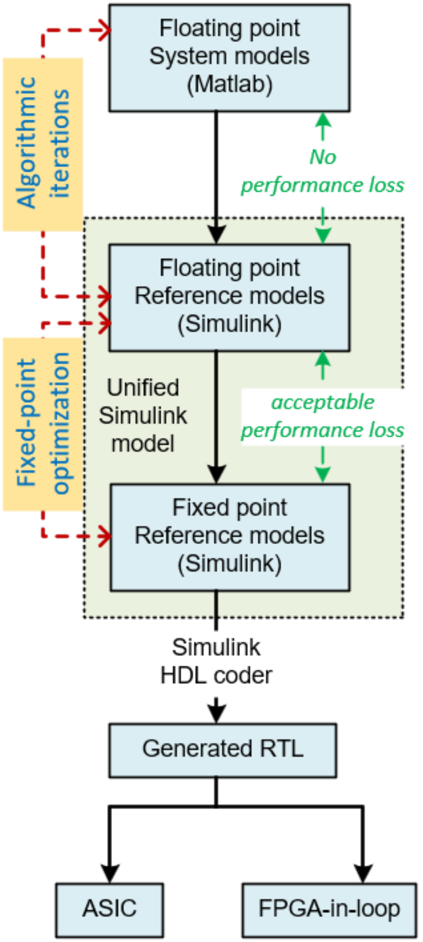
- Tx and Rx blocks as well as the entire DSP line-up underwent manifold changes
  - to meet the system performance for various protocol standards.

- The iterative nature quickly became challenging to perform fixed point optimizations as well as keep the reference models up to date with the system models.


Floating point System models (Matlab)
Algorithmic iterations
No performance loss
Floating point Reference models (Simulink)
Fixed-point optimization
acceptable performance loss
Fixed point Reference models (Simulink)
Simulink HDL coder
Generated RTL
ASIC
FPGA-in-loop

Algorithms to Silicon – Proposed solution & advantages

- Unified Simulink model approach
  - Methodology applied to all blocks in the receiver line-up.
- Seamless switching between fixed- and floating-point formats.
  - Algorithmic changes can be absorbed by updating the Simulink model in floating point mode.
  - Fine tune the updated Simulink model for fixed point performance
- Selectively specify floating- or fixed-point modes to Simulink models when evaluating entire line-up.
- Better visibility and debug through probe points within the design.
- De-modulation block as a case-study to demonstrate the proposed methodology.


Floating point System models (Matlab)
Algorithmic iterations
No performance loss
Floating point Reference models (Simulink)
Unified Simulink model
acceptable performance loss
Fixed-point optimization
Fixed point Reference models (Simulink)
Simulink HDL coder
Generated RTL
ASIC
FPGA-in-loop

Demodulation – a case study

- Demodulation comprises
  - Error computation
  - Noise averaging
  - Reciprocal and
  - Scaling.
- Define quantization points at desired points.
  - Data Type Conversion block from Simulink library
- Define fixed- and floating-point datatypes.
- One time build out of some test infrastructure.
  - Compare signals at Q between MATLAB model and Simulink models

De-modulation algorithm

$$x_d[n] = \text{Decision}(x[n])$$

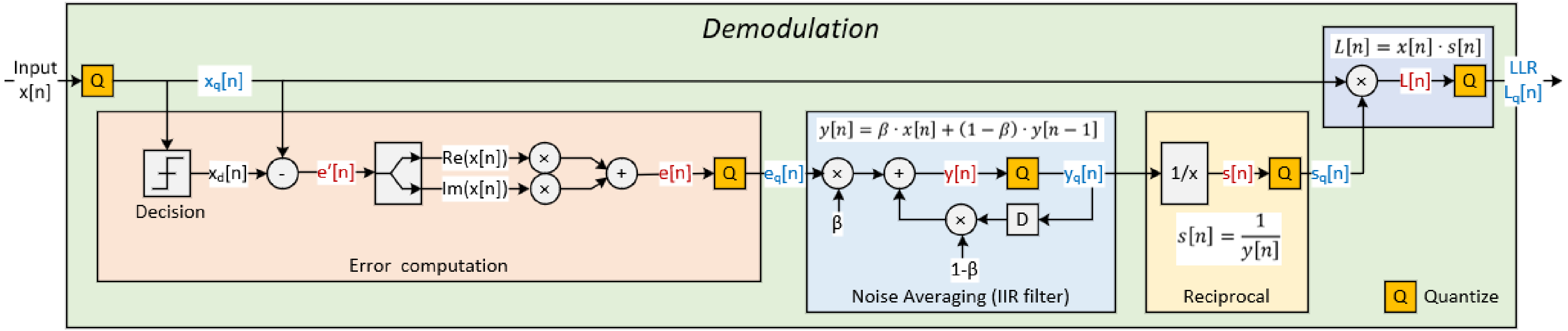
$$e'[n] = x[n] - x_d[n]$$

$$e[n] = \sum \left(Re(e'[n])\right)^2 + \left(Im(e'[n])\right)^2$$

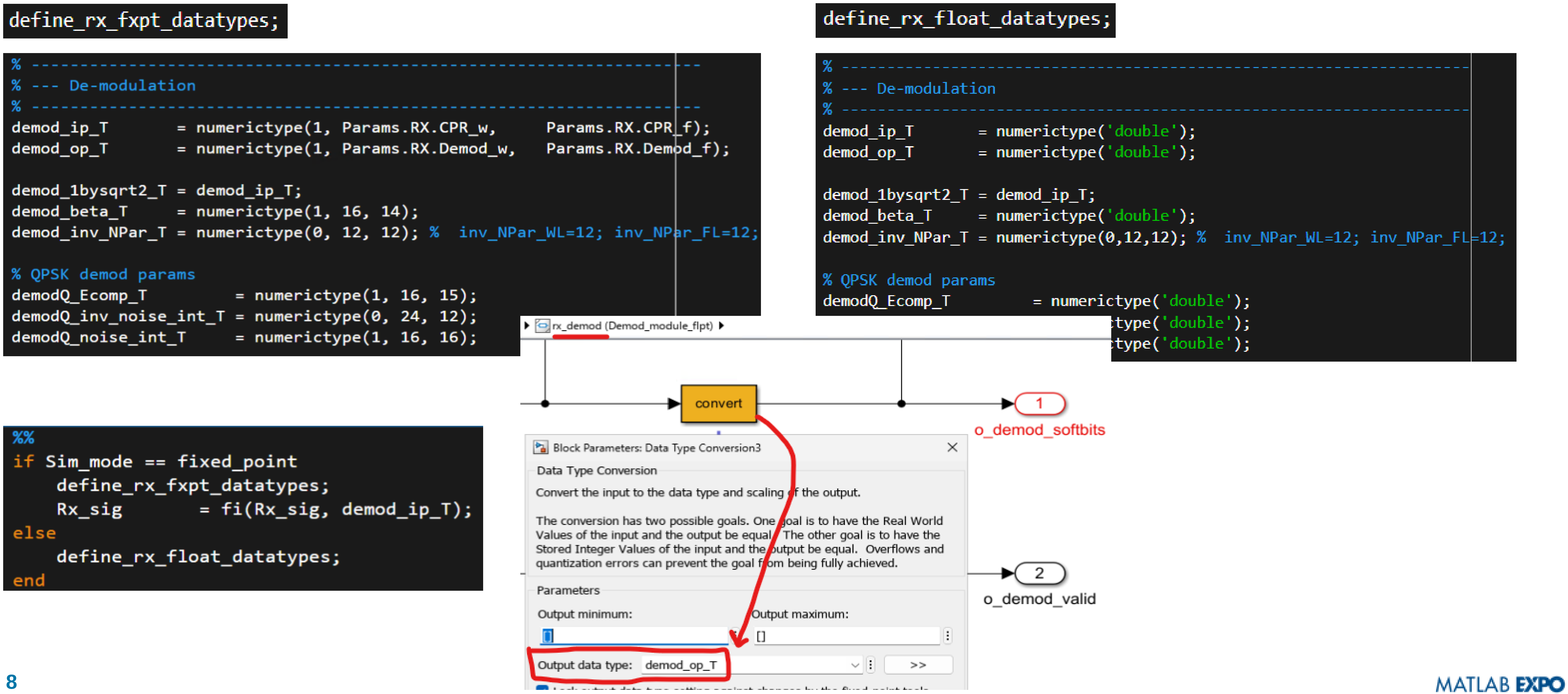
$$y[n] = \beta \cdot e[n] + (1 - \beta) \cdot e[n-1]$$

$$s[n] = \frac{1}{y[n]}$$

$$L[n] = x[n] \cdot s[n]$$



Proposed methodology – specifics



```
define_rx_fxpt_datatypes;
```

```
% ----------------------------------------------------------------------
% --- De-modulation
% ----------------------------------------------------------------------
demod_ip_T       = numerictype(1, Params.RX.CPR_w,     Params.RX.CPR_f);
demod_op_T       = numerictype(1, Params.RX.Demod_w,   Params.RX.Demod_f);

demod_1bysqrt2_T = demod_ip_T;
demod_beta_T     = numerictype(1, 16, 14);
demod_inv_NPar_T = numerictype(0, 12, 12); %  inv_NPar_WL=12; inv_NPar_FL=12;

% QPSK demod params
demodQ_Ecomp_T          = numerictype(1, 16, 15);
demodQ_inv_noise_int_T = numerictype(0, 24, 12);
demodQ_noise_int_T      = numerictype(1, 16, 16);
```

```
define_rx_float_datatypes;
```

```
% ----------------------------------------------------------------------
% --- De-modulation
% ----------------------------------------------------------------------
demod_ip_T       = numerictype('double');
demod_op_T       = numerictype('double');

demod_1bysqrt2_T = demod_ip_T;
demod_beta_T     = numerictype('double');
demod_inv_NPar_T = numerictype(0,12,12); %  inv_NPar_WL=12; inv_NPar_FL=12;

% QPSK demod params
demodQ_Ecomp_T          = numerictype('double');
                          type('double');
                          type('double');
```

```
%%
if Sim_mode == fixed_point
    define_rx_fxpt_datatypes;
    Rx_sig        = fi(Rx_sig, demod_ip_T);
else
    define_rx_float_datatypes;
end
```

Block Parameters: Data Type Conversion3

Data Type Conversion

Convert the input to the data type and scaling of the output.

The conversion has two possible goals. One goal is to have the Real World Values of the input and the output be equal. The other goal is to have the Stored Integer Values of the input and the output be equal. Overflows and quantization errors can prevent the goal from being fully achieved.

Parameters

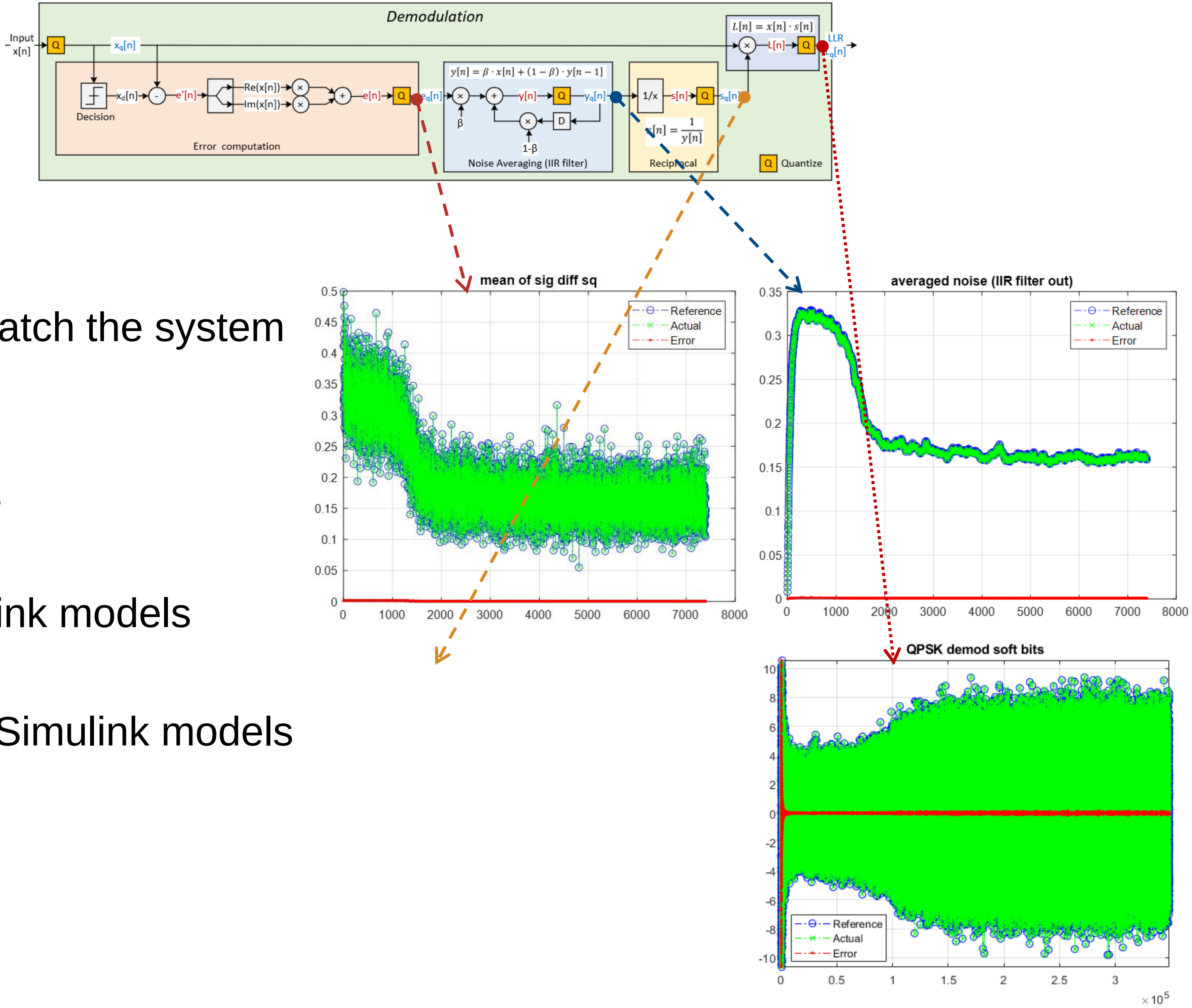
Output minimum: []

Output maximum: []

Output data type: demod_op_T

☑ Lock output data type setting against changes by the fixed-point tools

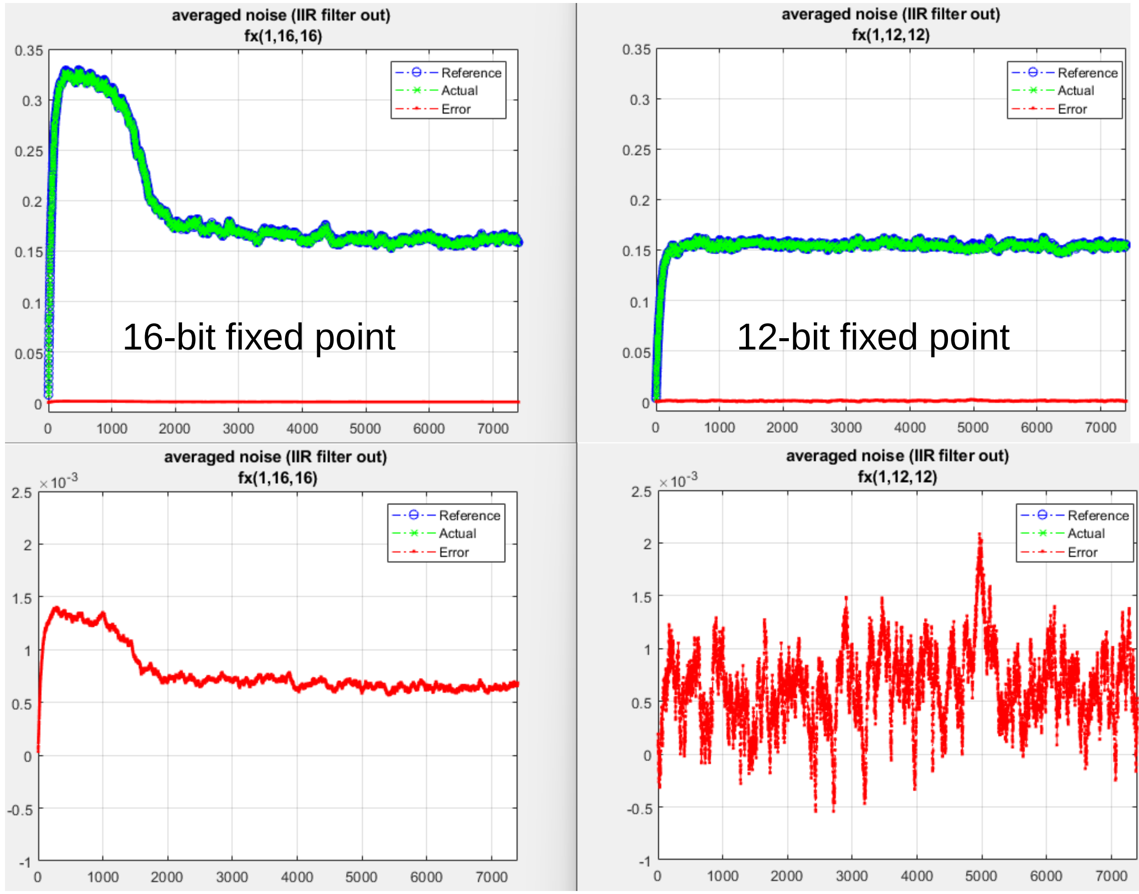
MATLAB vs. Simulink



- Simulink model development to match the system model (reference).
- Signal comparison at probe points
- Floating point MATLAB and Simulink models
- Validate functional correctness of Simulink models

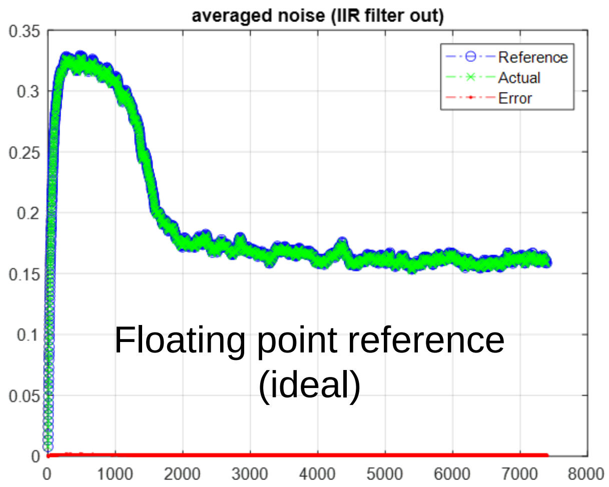
Exploring fixed point bit width optimization

- Optimization for hardware area, speed and power.
- Final bit width choice to meet system performance requirements.
- Example demonstrating 16-bit vs. 12-bit word length tradeoff at IIR filter output.



16-bit fixed point

12-bit fixed point



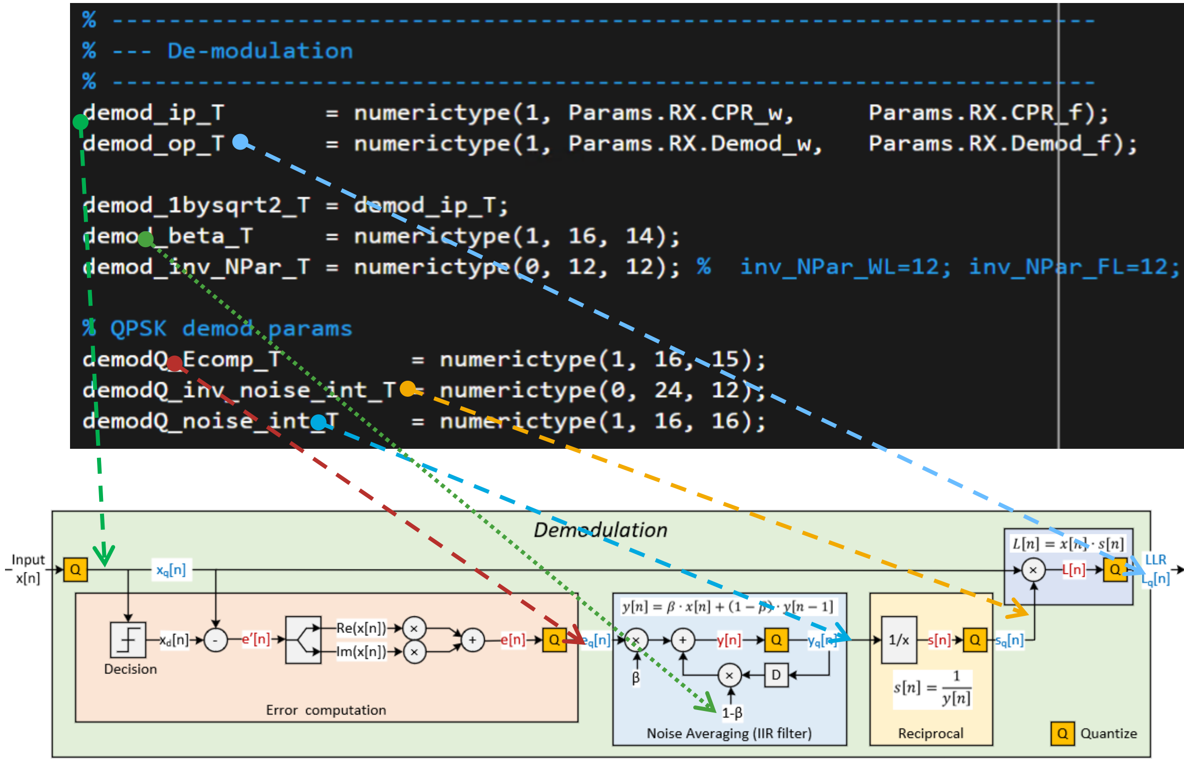
Floating point reference (ideal)

Beyond the basics

- The methodology can used to apply advanced settings such as
  - Rounding modes – floor, ceil, round…
  - Overflow actions – saturation, wrap-around…
  - Defining fixed- and floating-point Look-up Tables.

- The methodology has been extended for
  - Programmatic generation of RTL
  - Programmatic generation of DPIC models

- The fixed-point Simulink model is
  - FPGA-in-the-loop ready for validation on hardware
  - Synthesizable RTL for ASIC

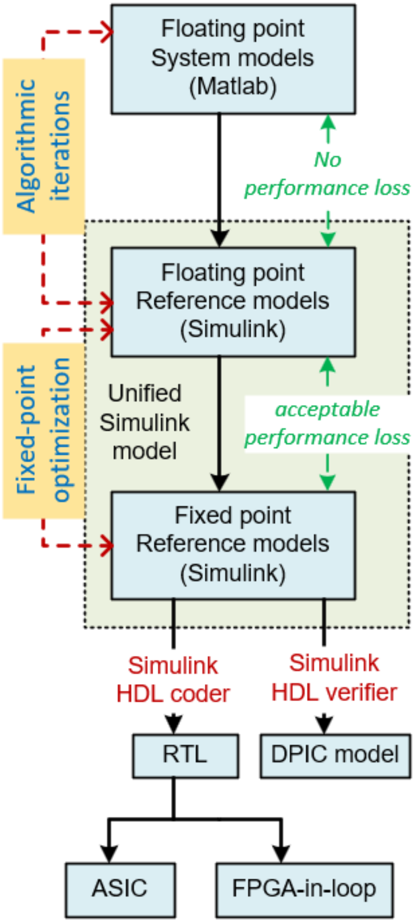


```
% ---------------------------------------------------------------------------
% --- De-modulation
% ---------------------------------------------------------------------------
demod_ip_T        = numerictype(1, Params.RX.CPR_w,     Params.RX.CPR_f);
demod_op_T        = numerictype(1, Params.RX.Demod_w,   Params.RX.Demod_f);

demod_1bysqrt2_T = demod_ip_T;
demod_beta_T      = numerictype(1, 16, 14);
demod_inv_NPar_T = numerictype(0, 12, 12); %  inv_NPar_WL=12; inv_NPar_FL=12;

% QPSK demod params
demodQ_Ecomp_T            = numerictype(1, 16, 15);
demodQ_inv_noise_int_T = numerictype(0, 24, 12);
demodQ_noise_int_T        = numerictype(1, 16, 16);
```

https://www.mathworks.com/help/hdlverifier/fpga-in-the-loop.html

Conclusion

- Versatile mode switching *(between floating- and fixed-point)*.
- Swift turnaround on algorithmic updates *(verify in floating-point mode)*.
- Better debugging capability *(1:1 comparison at probe points)*.
- Faster convergence on fixed-point optimization *(when using type defines)*.
- Easily scalable - across hierarchies and multiple blocks
- Support for integrating advanced features from MathWorks *(programmatic flows)*.
- Leveraging functional verification in MATLAB/Simulink world *(with both floating- and fixed-point models)*.
- Quick turn-around of synthesizable RTL *(HDL coder).*
- Functionally correct DPIC models for verification *(HDL verifier).*



MATLAB EXPO

Thank you

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA).

The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

Distribution Statement “A” (Approved for Public Release, Distribution Unlimited

MathWorks®

Algorithms to Silicon – Problem statement (Abstract)

Algorithm development cycle can eat into timelines of reference model development and fixed-point conversion.

This can further push out the hardware development cycle (RTL). We have developed a methodology that can leapfrog incorporating the algorithm changes all the way to RTL. This methodology demonstrates its versatility of using a single Simulink model to incorporate and validate algorithmic changes (block/system performance), fixed point conversion effects and finally generate synthesizable RTL (leveraging Simulink HDL coder) from the same model. In addition, the fixed point Simulink model can further extended to generate a bit accurate DPIC model for RTL verification.

- A real-life wireless receiver algorithm (that is a part of an entire wireless modem being developed) will be demonstrated.
- Running the Simulink reference model in floating point mode and compare against MATLAB algorithm reference.
- Seamless switching to use the same Simulink reference model in fixed point mode, showing fixed point effects.
- Attendees will learn a design methodology of using a single Simulink reference model that can be used for performance evaluation of algorithms, evaluate the model's fixed point performance and finally generate synthesizable RTL.