

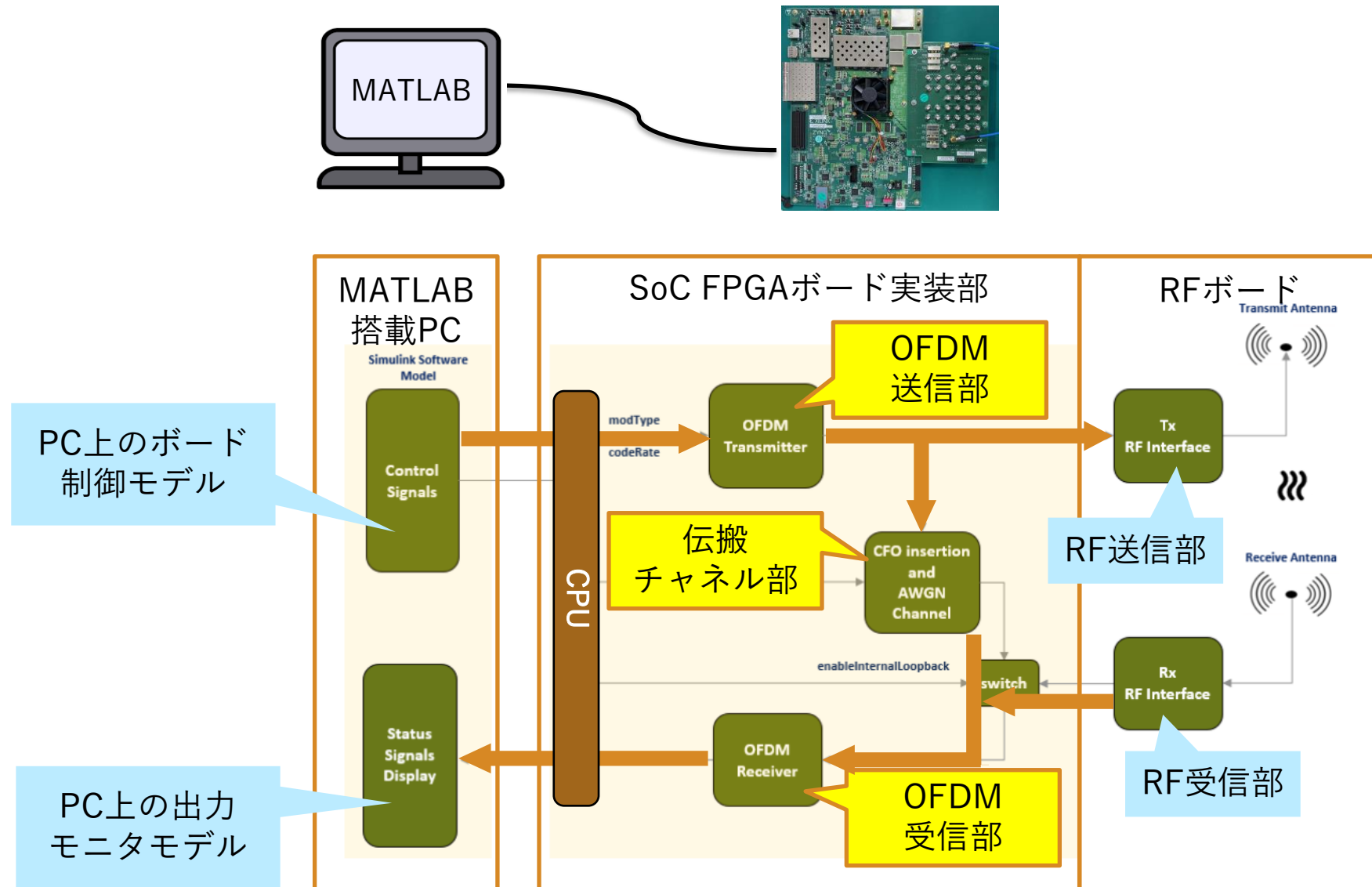
# モデルベースデザインによる無線通信システムの SoC FPGA実装ワークフロー

MathWorks  
アプリケーションエンジニアリング部  
中村 勝

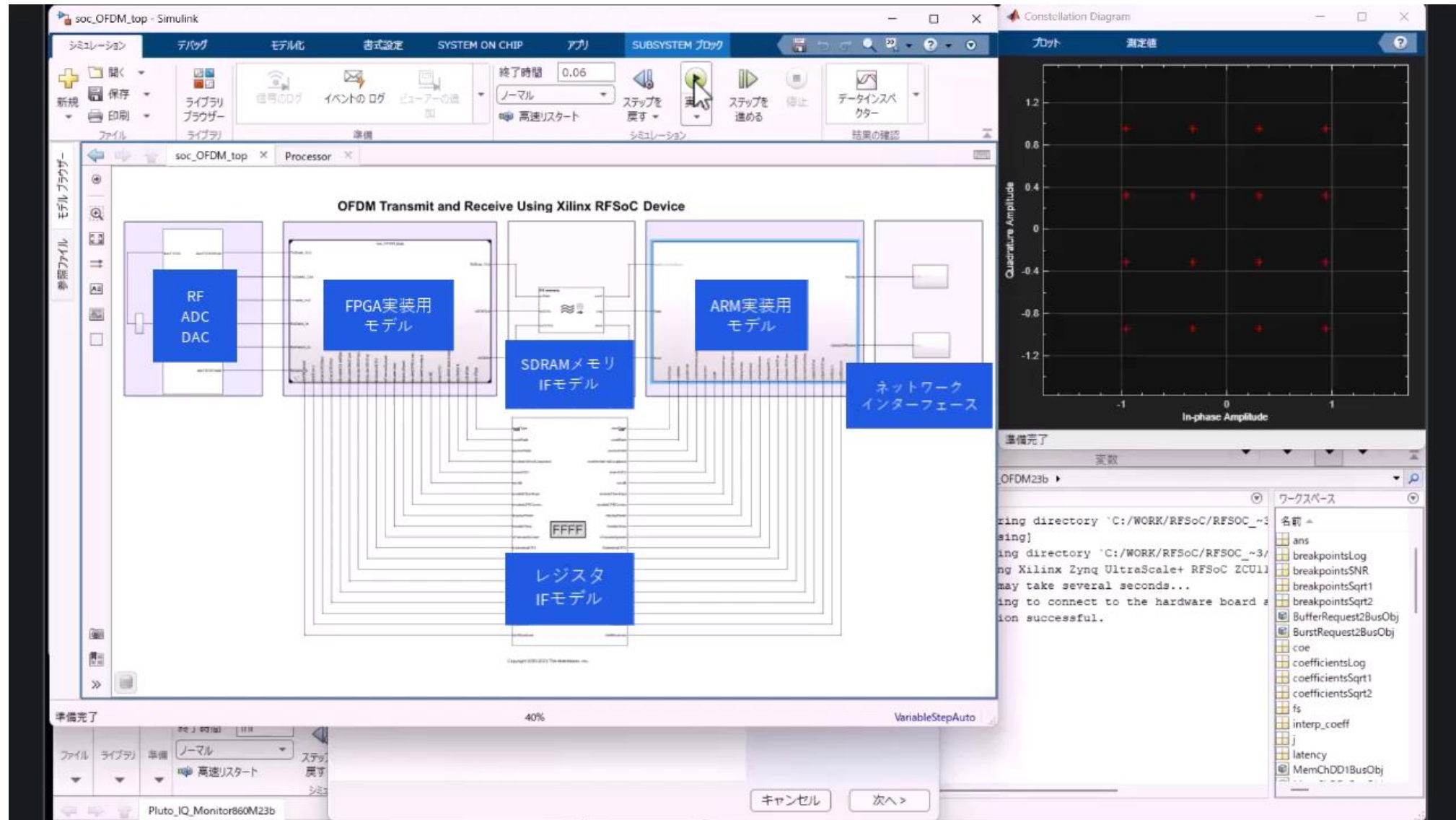
## はじめに

- 無線通信システム開発では、仕様検討から機能設計、コーディング、FPGAボード等による試作、及び各工程での評価検証と様々な工程を多様なツールを用いて進める必要があります
- モデルベースデザインの適用により仕様検討から試作評価までを同一ツール環境で実施可能
- 本手法ではモデルから実装コードやテストベンチを自動生成でき、これらを検証に再利用可能
- 工程間の齟齬が減り効率的な検証が進むため、開発期間短縮や品質向上が期待できます
- 本セッションでは、モデルベースデザインによるSoC FPGA実装例としてOFDM送受信機を評価ボードへ実装する際の一連のワークフローについてご紹介します

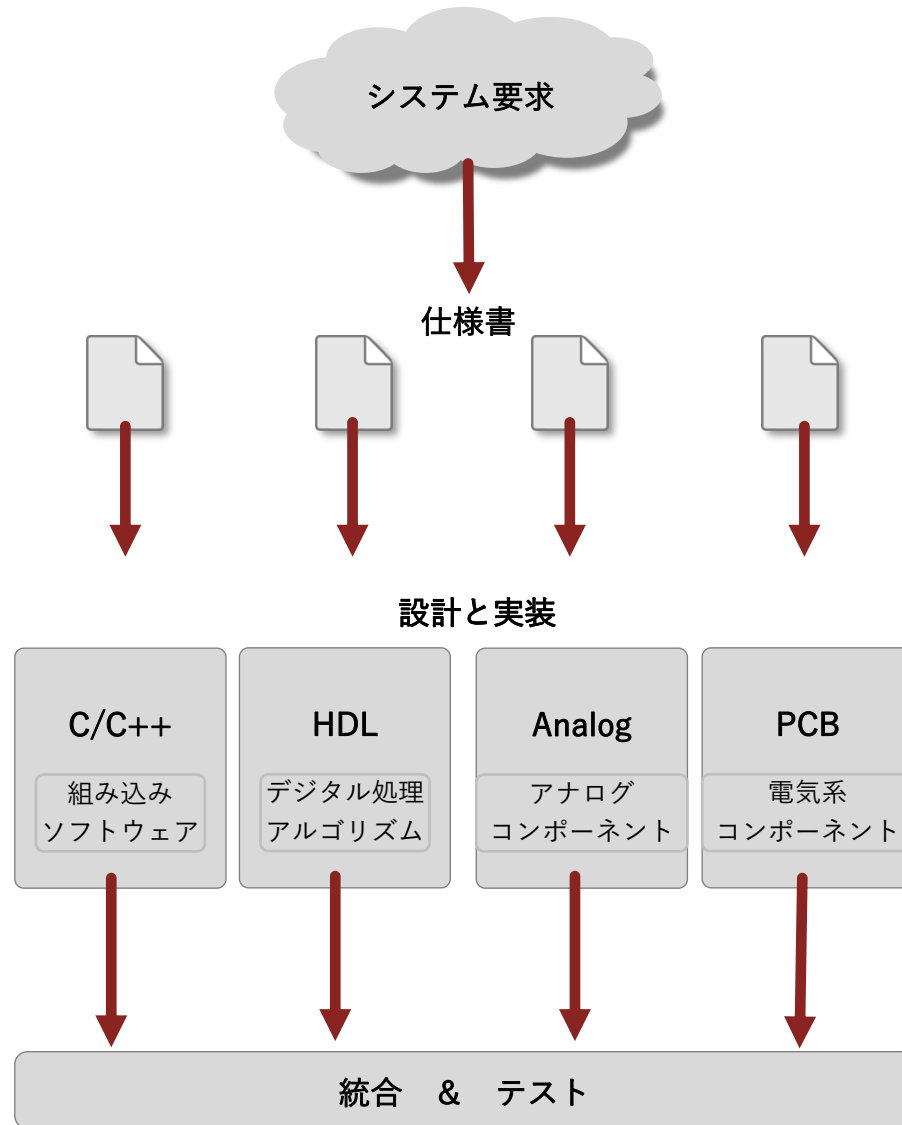
# SoC FPGA実装用OFDM送受信モデルの概要



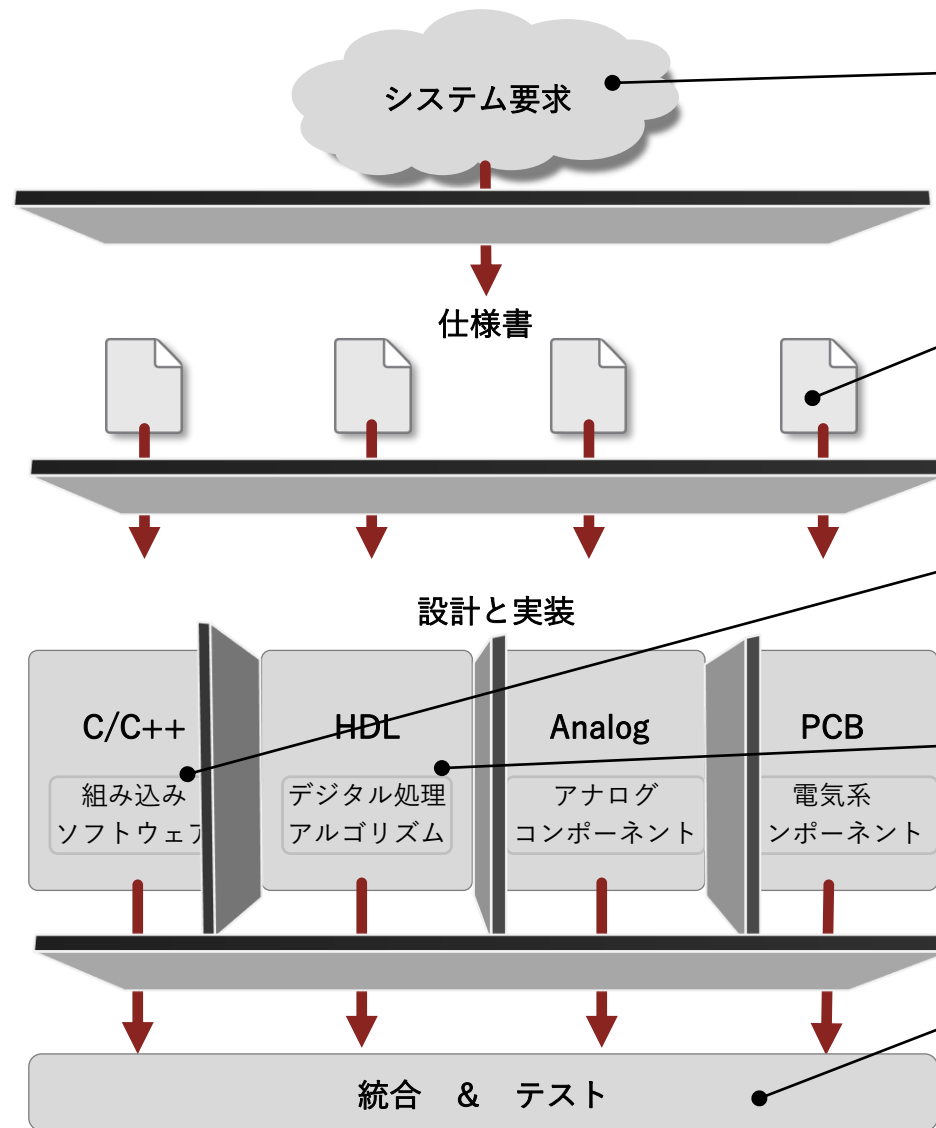
# OFDM送受信モデルのシミュレーションとSoC FPGAボードへの実装デモ



# 現在主流の開発フロー



# 現在主流の開発フローにおける課題



## 要求仕様書

妥当性の確認、内容の解析が困難

## 紙ベースの仕様書

エンジニアによって理解が異なり  
実際に実装されたものと差異が生じ易い

## 手書きのC/HDLコード

長いコーディング期間、エンジニアスキル  
に依存するコード、バグ混入可能性大

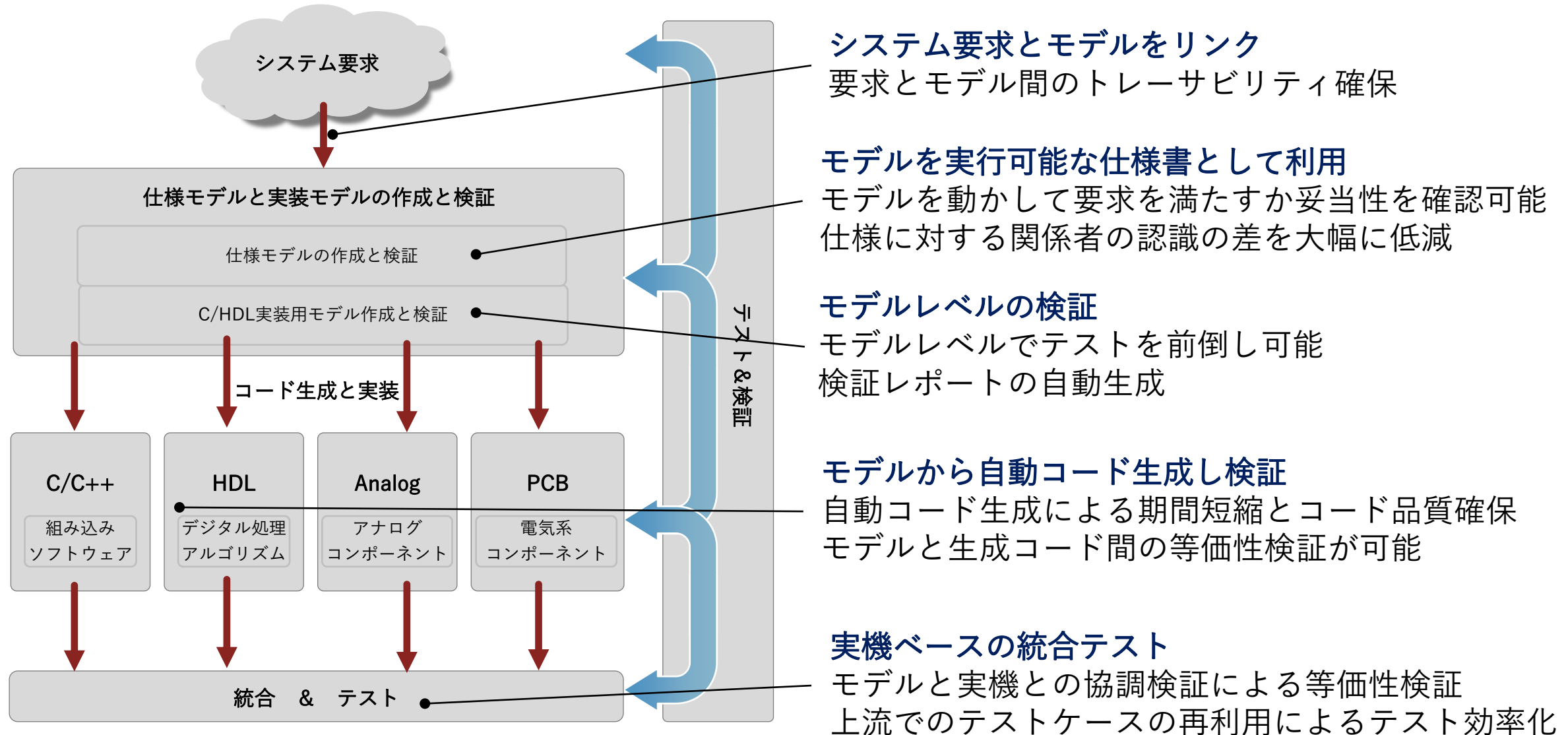
## 試作機による検証

非常に高価、作り直しが容易でない

## 実機ベースの統合テスト

仕様書や設計段階で混入した問題が、  
開発後期で顕在化

# モデルベースデザインによる開発フローとメリット

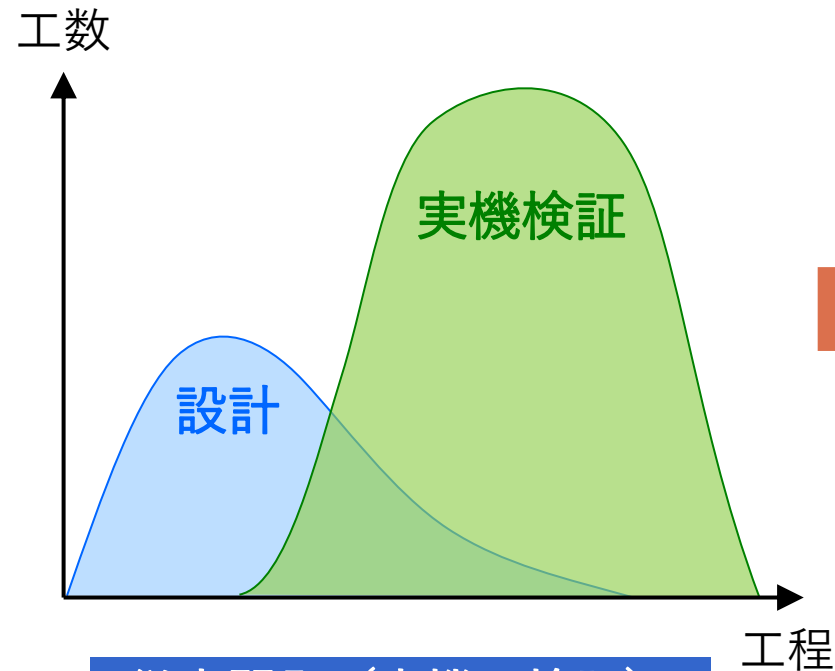




# モデルベースデザイン(MBD)ワークフロー導入で期待される効果

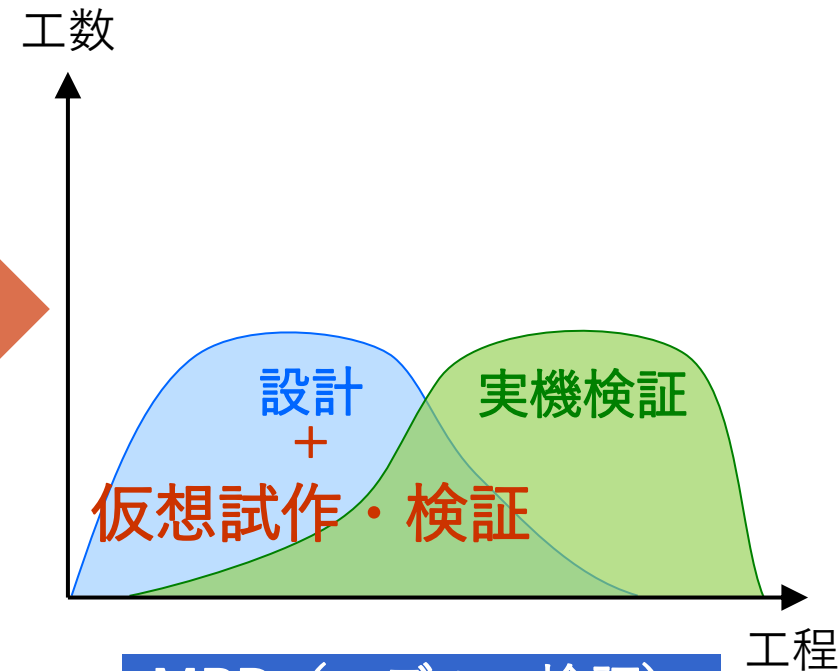
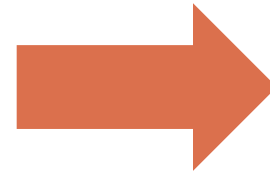
モデルを使った早期検証と自動コード生成によりQCDを向上する

QCD = Quality, Cost, Delivery



従来開発（実機で検証）

- 後工程にテストが集中
- 設計抜け漏れが起こり易い
- ハンドコードの不具合修正コストが高い



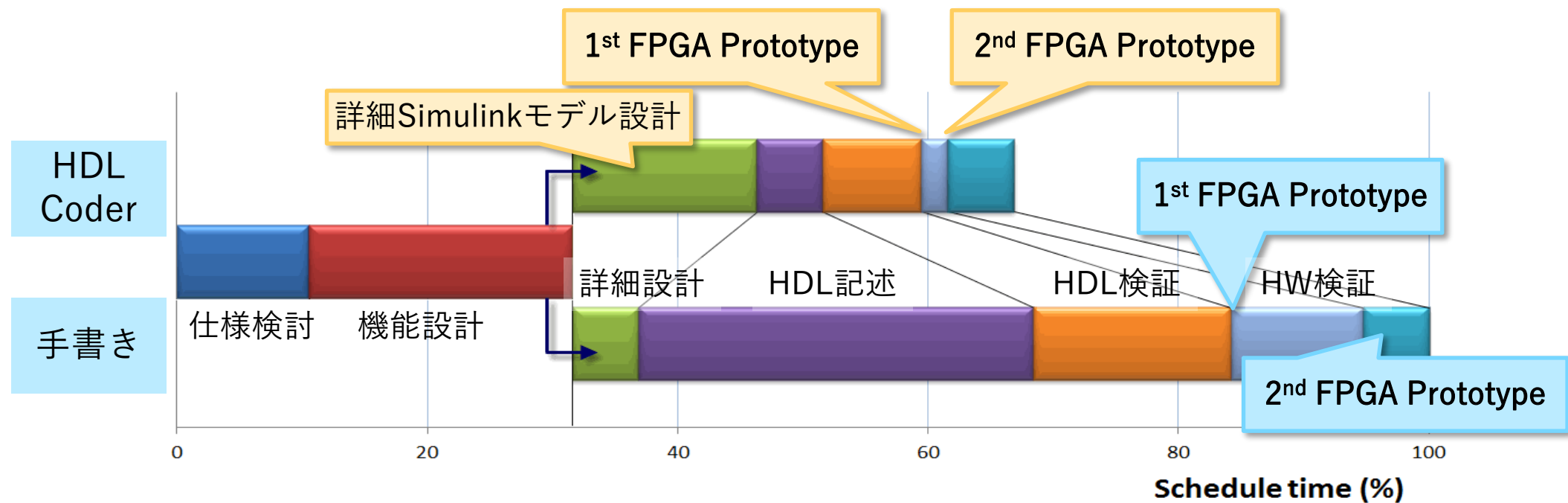
MBD（モデルで検証）

- 開発上流にテストを前倒し
- 設計抜け漏れを早期に発見・修正
- 自動コード生成により不具合修正コストが低い



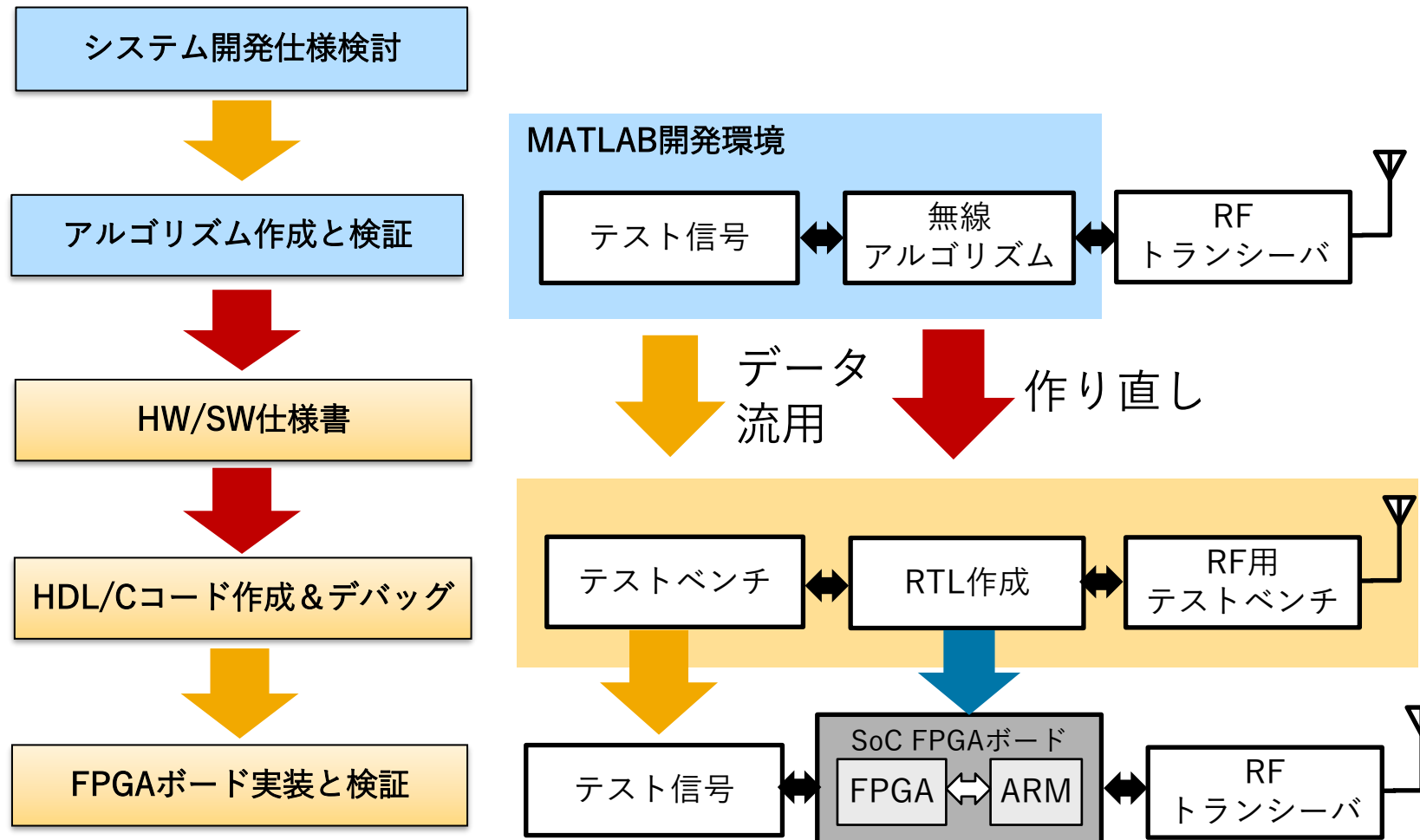
# ASIC開発工数のベンチマーク（ユーザ事例）

- 48% 実装の工数削減（プロジェクト全体では33%）
- 47% FPGAプロトタイプ開発期間の削減
- 80% HW（実機）検証削減



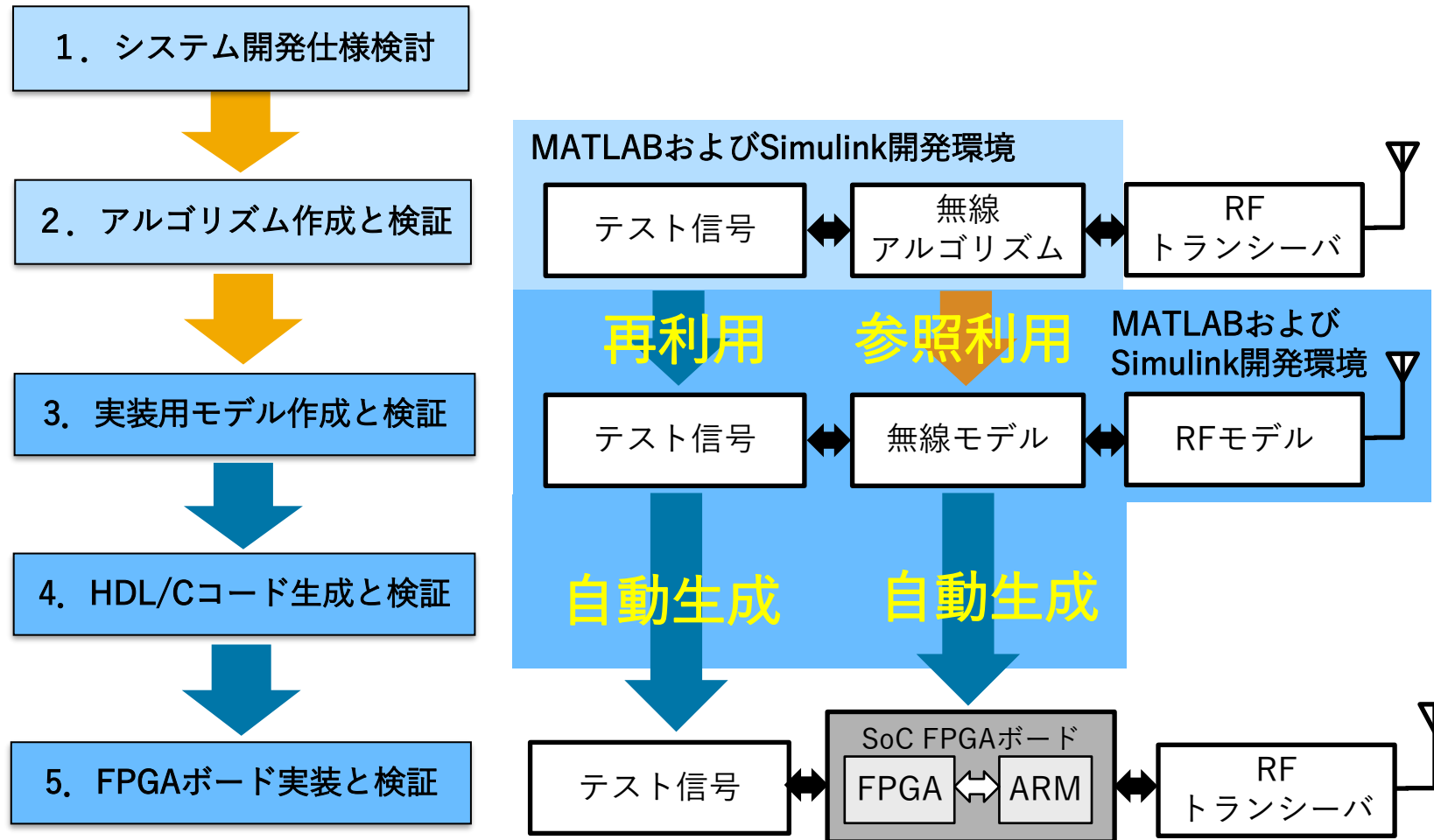
# 従来の無線通信システムのSoC FPGA実装フロー

仕様書に沿った実装コード作成からボード実装検証に至るまでほぼ手作業



# モデルベースデザインによる無線通信システムのSoC FPGA実装フロー

実装用モデルからボード実装検証まで自動コード生成により半自動化



# Step1 無線システムの開発仕様の検討

無線系の豊富な関数を提供 → シミュレーションにより仕様検討を効率化

- LTE、5G、WLAN等の規格に沿った各種シミュレーション用関数群
- 例) OFDMTx()関数を用いて各種OFDM変調波形を簡単に生成し特性解析可能

1. システム開発  
仕様検討

2. アルゴリズム  
作成と検証

3. 実装用モデル  
作成と検証

4. HDL/Cコード  
生成と検証

5. FPGAボード  
実装と検証

ダウンリンク チャンネル

ダウンリンク物理量信号

同期信号

nrPSS	PSS シンボルの生成
nrPSSIndices	Generate PSS resource element indices
nrSSS	SSS シンボルの生成
nrSSSIndices	Generate SSS resource element indices

PDSCH 復調基準信号

nrPDSCHDMRS	PDSCH DM-RS シンボルの生成 (R2020a 以降)
nrPDSCHDMRSIndices	Generate PDSCH DM-RS indices (R2020a 以降)
nrPDSCHDMRSConfig	PDSCH DM-RS configuration parameters (R2020a 以降)

PBCH 復調基準信号

nrPBCHDMRS	PBCH DM-RS シンボルの生成
nrPBCHDMRSIndices	Generate PBCH DM-RS resource element indices

チャンネル状態情報基準信号

nrCSI-RS	CSI-RS シンボルの生成 (R2019b 以降)
nrCSI-RSIndices	Generate CSI-RS resource element indices (R2019b 以降)
nrCSI-RSConfig	CSI-RS 構成パラメーター (R2019b 以降)

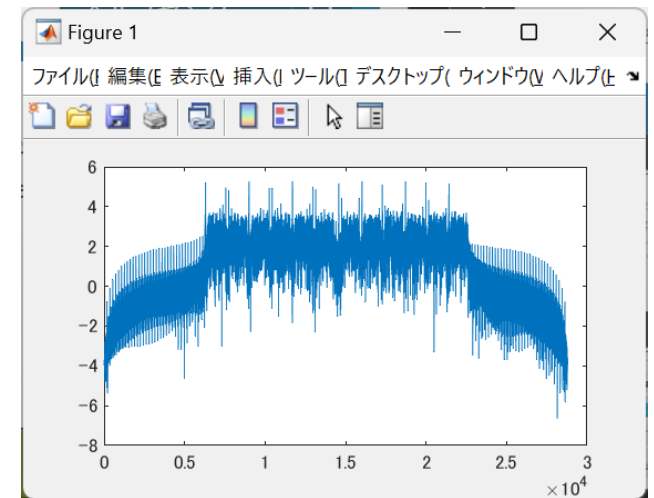
PDSCH 位相トラッキング基準信号

nrPDSCHPTRS	PDSCH PT-RS シンボルの生成 (R2020a 以降)
nrPDSCHPTRSIndices	Generate PDSCH PT-RS Indices (R2020a 以降)
nrPDSCHPTRSConfig	PDSCH PT-RS configuration parameters (R2020a 以降)

```
txParam.modOrder = 16; % Modulation order corresponding to 16-QAM
txParam.codeRateIndex = 0; % Code rate index corresponding to 1/2
txParam.numFrames = 5; % Number of frames to be generated

% Calculate transport block size (trBlkSize) using parameters
numSubCar = 72; % Number of subcarriers per symbol
pilotsPerSym = 12; % Number of pilots per symbol
numDataOFDMSymbols = 32; % Number of data OFDM symbols
bitsPerModSym = log2(txParam.modOrder); % Bits per modulated symbol
codeRate = 1/2; % Punctured code rate
dataConvK = 7; % Constraint length of convolutional code polynomial
dataCRCLen = 32; % Data CRC length
trBlkSize = ((numSubCar-pilotsPerSym)*numDataOFDMSymbols* ...
    bitsPerModSym*codeRate) - (dataConvK-1) - dataCRCLen;
txParam.txDataBits = randi([0 1],txParam.numFrames*trBlkSize,1);

% Generate complex baseband transmitter waveform
fprintf('\n-----\n');
[txWaveform,txGrid,txDiagnostics] = whdlexamples.OFDMTx(txParam);
fprintf('\n-----\n');
```



5 G用関数(一部)

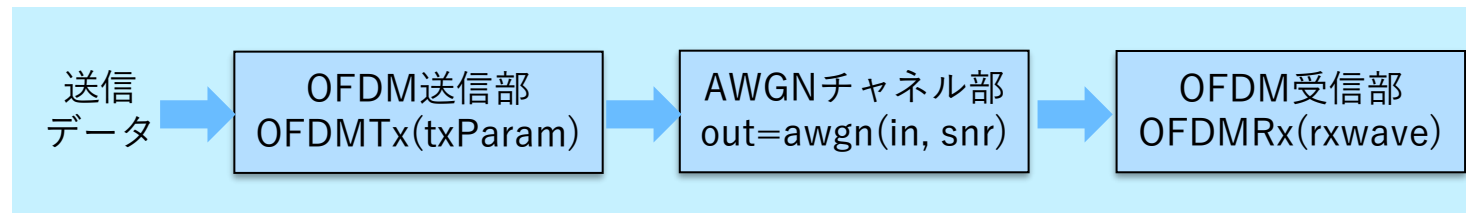
OFDM変調のコード例(18行)

OFDM送信信号スペクトラム

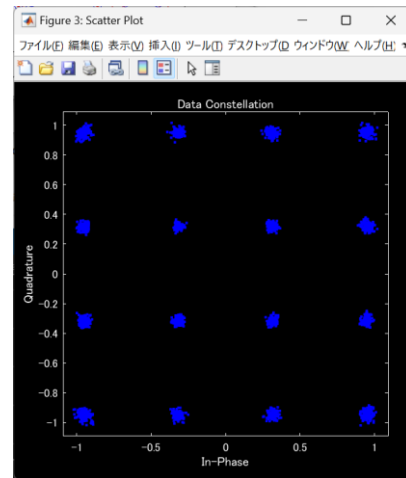
# Step2 アルゴリズム作成とシミュレーションによる検証

無線用MATLAB関数の利用でシステム全体のアルゴリズム作成と特性評価が容易

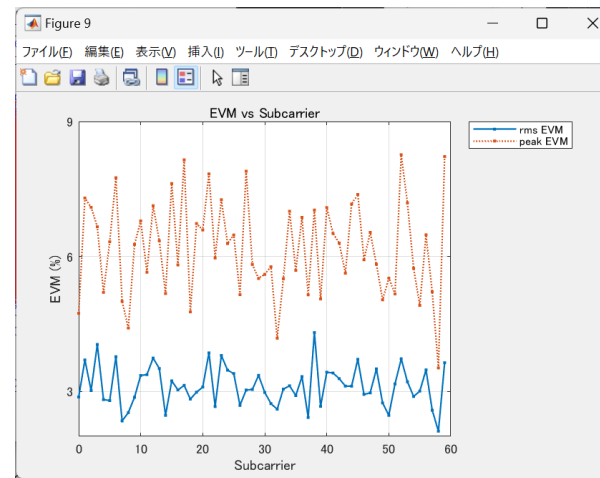
- MATLAB関数を用いたOFDM送受信システムの構成例



- 各種測定・表示関数により容易に特性を確認可能



コンスタレーション



EVM特性

```

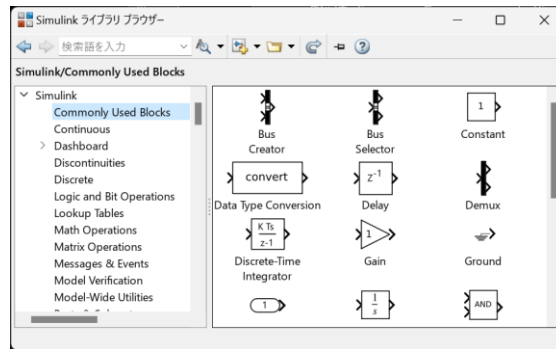
1 % HDL OFDM Transmitter MATLAB Reference
2
3 % Parameters
4 txParam.modOrder = 16; % Modulation order corresponding to 16-QAM
5 txParam.codeRateIndex = 0; % Code rate index corresponding to 1/2
6 txParam.numFrames = 5; % Number of frames to be generated
7
8 % Calculate transport block size (trbSize) using parameters
9 numSubCar = 72; % Number of subcarriers per symbol
10 pilotsPerSym = 12; % Number of pilots per symbol
11 numDataOFDMSymbols = 32; % Number of data OFDM symbols
12 bitsPerModSym = log2(txParam.modOrder); % Bits per modulated symbol
13 codeRate = 1/2; % Punctured code rate
14 dataConvK = 7; % Constraint length of convolutional code polynomial
15 dataCRCLen = 32; % Data CRC length
16 trbSize = ((numSubCar - pilotsPerSym) * numDataOFDMSymbols * ...
17     bitsPerModSym * codeRate) - (dataConvK - 1) - dataCRCLen;
18 txParam.txDataBits = randi([0 1], txParam.numFrames * trbSize, 1);
19
20 % Generate complex baseband transmitter waveform
21 fprintf('n\n');
22
23 % Transmitting N frames ...
24 [txWaveform, txDiag, txDiagnostics] = whdlexamples.OPDMTx(txParam);
25 fprintf('n Transmission successful.\n');
26
27 % HDL OFDM Receiver MATLAB Reference
28
29 % Parameters
30 FFTLen = 128;
31 CPLen = FFTLen/4;
32 usedSubCar = 72; % Out of 128 subcarriers, 72 subcarriers are loaded with data
33
34 SNRdB = 30;
35 SNRdBInInput = SNRdB * ones(length(txWaveform)/633, 1);
36 seedsURNG1 = [121 719 511]; % Seeds for TausURNG1
37 seedsURNG2 = [2343 323 833]; % Seeds for TausURNG2
38 txScaleFactor = FFTLen / sqrt(usedSubCar);
39
40 awgnNoise = whdlexamples.hdlawgn(SNRdB * SNRdBInInput, seedsURNG1, seedsURNG2);
41
42 % 633 initial samples of the noise are discarded to match the MATLAB data
43 % with that of the simulink model
44 rxWaveform = txWaveform + (1/txScaleFactor) * awgnNoise(634:end);
45 fprintf('n Applying the AWGN channel at N dB ...\n', SNRdB);
46
47 % HDL OFDM Receiver MATLAB Reference
48
49 % Receiving process started ...
50 [rxDataBits, rxDiag] = whdlexamples.OPDMRx(double(fi(rxWaveform, 1, 16, 14)));
51 fprintf('n Reception completed.\n');
52
53 % Plot constellation of header and data
54 scatterplot(rxMatDiag.matHeaderConstellation(:, 1, 0, 'b'));
55 title('Header Constellation');
56 axisObj = gca;
57 axisObj.XColor = 'w';
58 axisObj.YColor = 'w';
59
60 scatterplot(rxMatDiag.matDataConstellation(:, 1, 0, 'b'));
61 title('Data Constellation');
62 axisObj = gca;
63 axisObj.XColor = 'w';
64 axisObj.YColor = 'w';
  
```

# Step3-1 実装用OFDM送受信モデルの作成

ブロック線図形式で実装モデルを作成でき、モデル全体を視覚的に設計可能

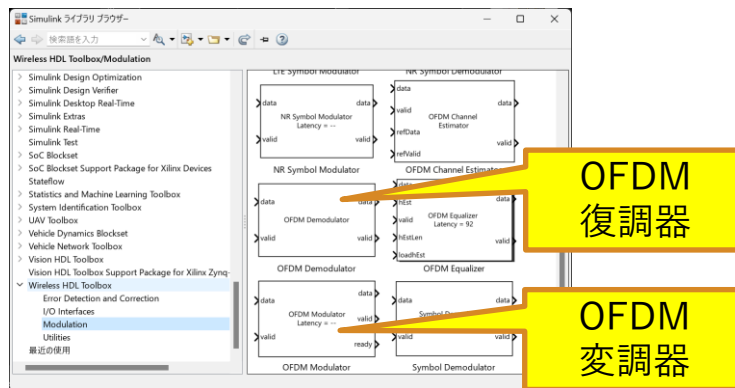
豊富なブロックライブラリ

・ 基本ブロック

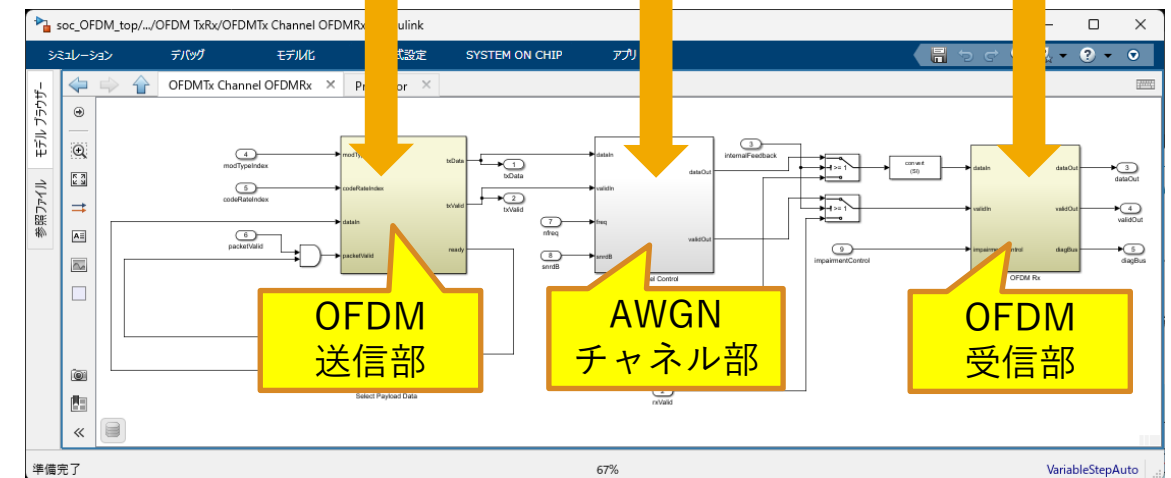
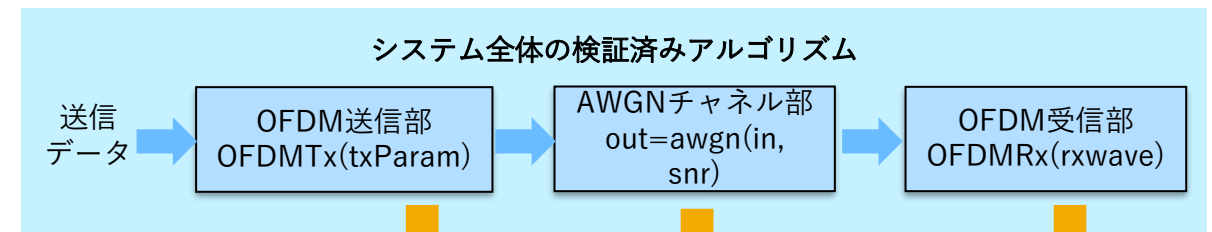


・ 用途別ブロック(無線、画像、DSP)

Wireless HDL Toolbox他

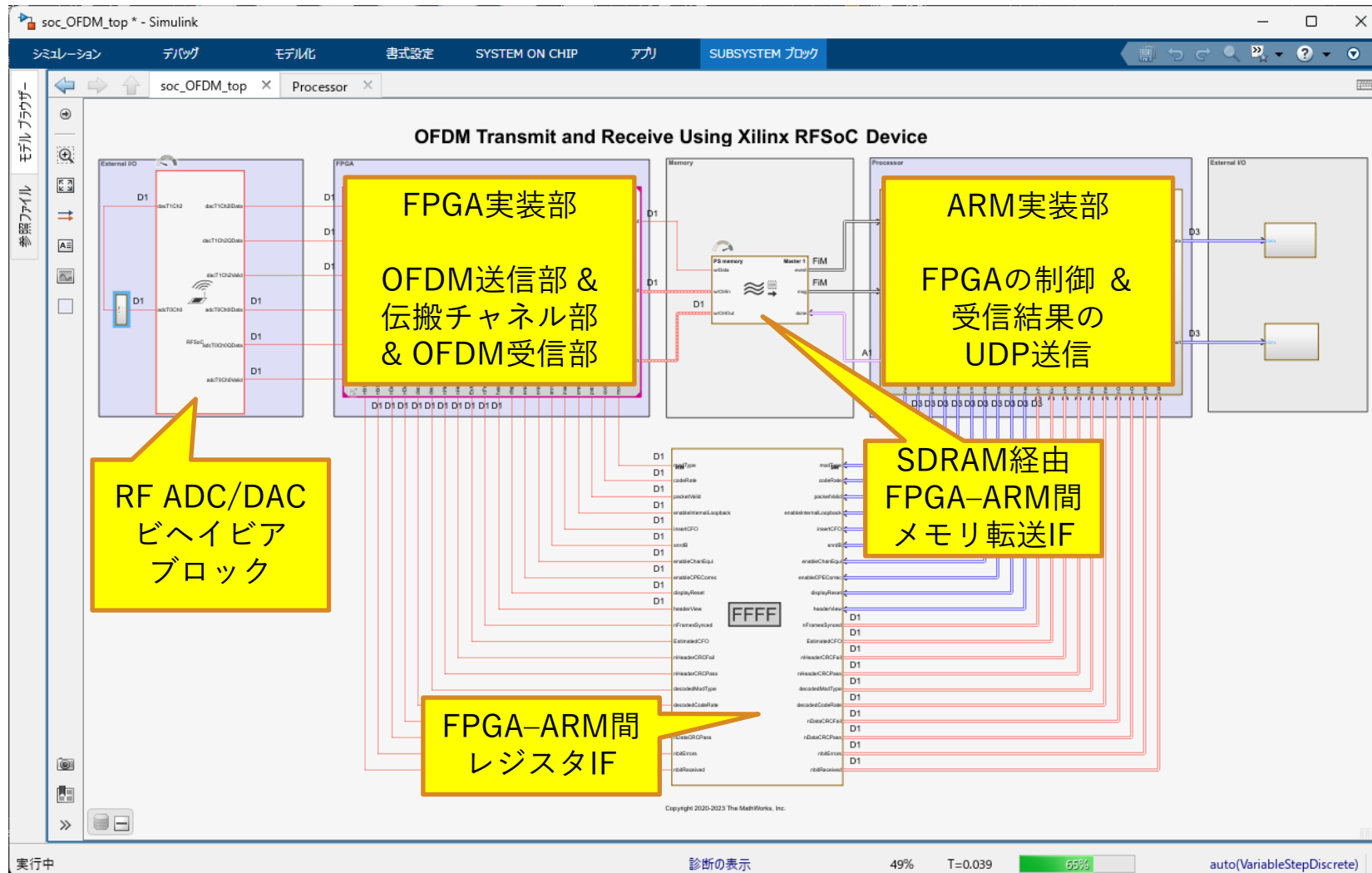


検証済みアルゴリズムを基準に実装用モデルを構成

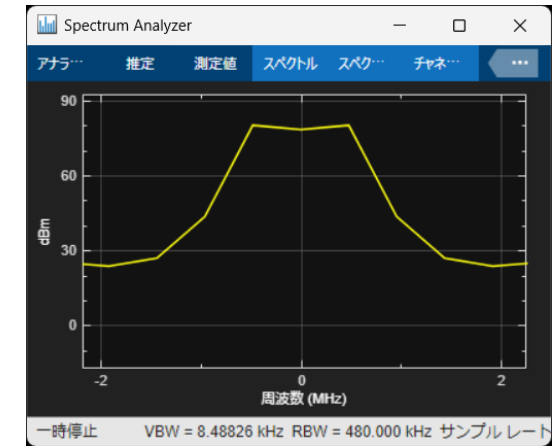


実装用SimulinkモデルのOFDM送受信部

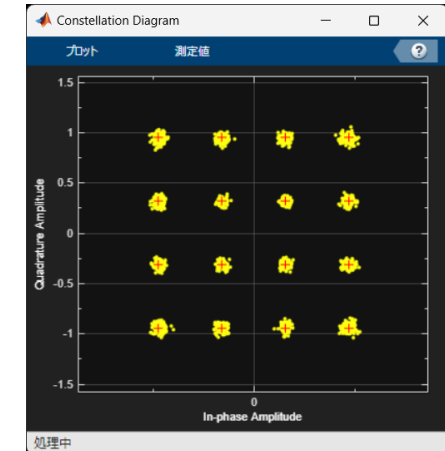
# 作成後のSoC FPGA実装用OFDM送受信モデルとシミュレーション結果



## 送信スペクトラム



## 受信側コンスタレーション





## Step3-2 実装用モデルと基準アルゴリズムとの等価性検証

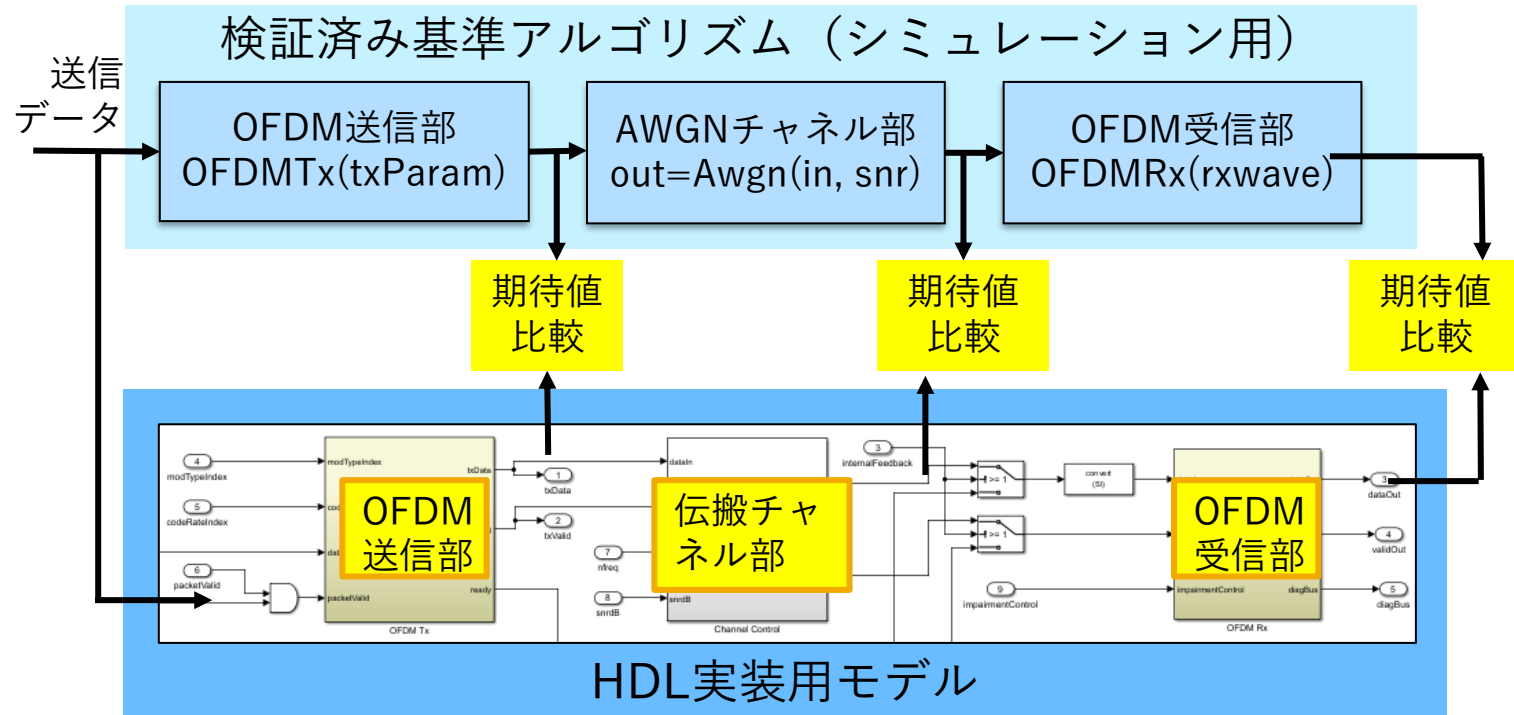
基準アルゴリズムと実装用モデルでは構造が大きく異なり等価性検証が不可欠

→ MATLABでは両者を同一環境で比較可能なため等価性検証が容易

基準アルゴリズムと実装用モデルの差

	基準アルゴリズム	実装用モデル
形式	MATLAB コード	Simulink モデル
処理単位	フレーム	サンプル
データ型	浮動小数点	固定小数点

アルゴリズムと実装モデル間の等価性検証手法



# Step4-1 実装モデルからのHDL/Cコードの自動生成

実装モデルから自動でHDL/Cコードを生成可能  
→ハンドコードによるバグ混入を回避

1. システム開発仕様検討

2. アルゴリズム作成と検証

3. 実装用モデル作成と検証

4. HDL/Cコード生成と検証

5. FPGAボード実装と検証

HDLコード生成 (HDL Coder)

対応言語: VHDL、Verilog、SystemVerilog

Cコード生成 (Embedded Coder)

対応言語: C、C++

2つのコード生成用アプリを用意

- HDLワークフローアドバイザ
- SoC Builder (SoC Blockset)

両者ともHDL/C生成とビルド後FPGA側とARM側双方に書き込み実行可能

**Cコード生成レポート**

**SoCビルドツール**

**HDLコード生成レポート**

```

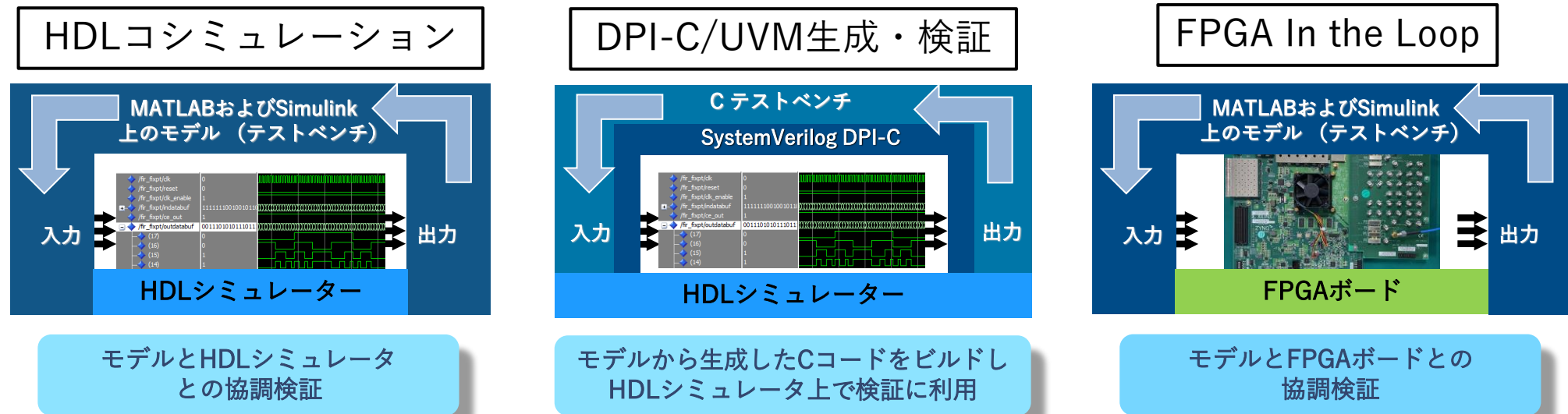
54
55 ofdm_ip_src_who10FDMTx_who10FDMTx u_OFDMTx (.clk(clk),
56 .reset(reset),
57 .enb(enb),
58 .modTypeIndex(modTypeIndex), //
59 .codeRateIndex(codeRateIndex),
60 .data(dataIn),
61 .valid(packetValid),
62 .txData_re(OFDMTx_out1_re), //
63 .txData_im(OFDMTx_out1_im), //
64 .txValid(OFDMTx_out2), //
65 .ready(OFDMTx_out3)
66 );
67
68 assign txData_re = OFDMTx_out1_re;
69
70 assign txData_im = OFDMTx_out1_im;
71
72 assign txValid = OFDMTx_out2;
73
74 assign ready = OFDMTx_out3;
75
76 endmodule // ofdm_ip_src_OFDMTx
77
  
```

生成Verilogコード例

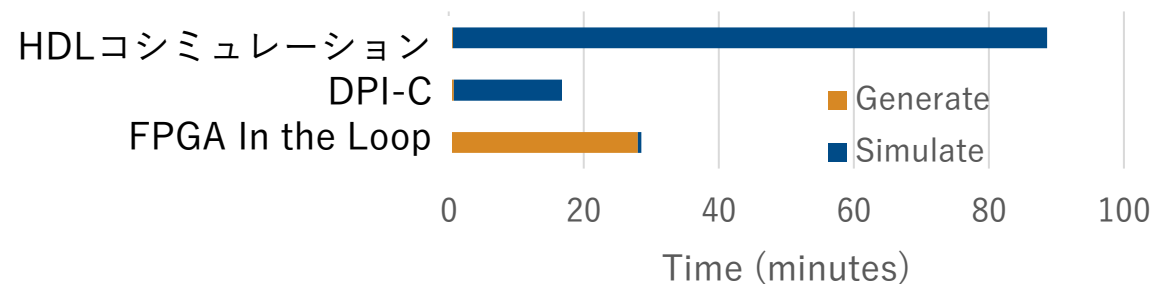
## Step4-2 実装用モデルからのテストベンチ生成とHDL検証

モデルをテストベンチとして再利用 → 検証作業を自動化し時間短縮

- モデルと生成HDLコード間の等価性検証手法として3つの検証手法を用意



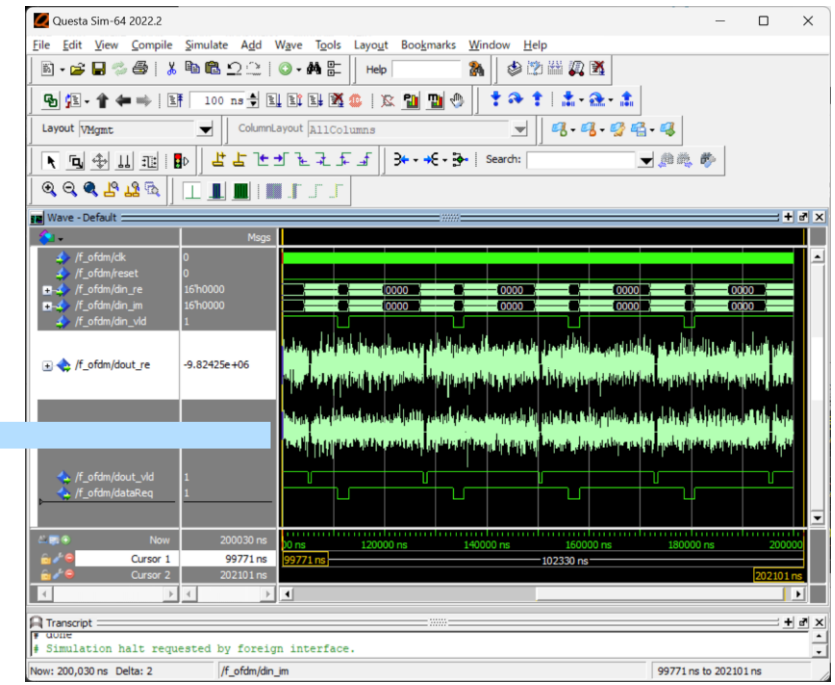
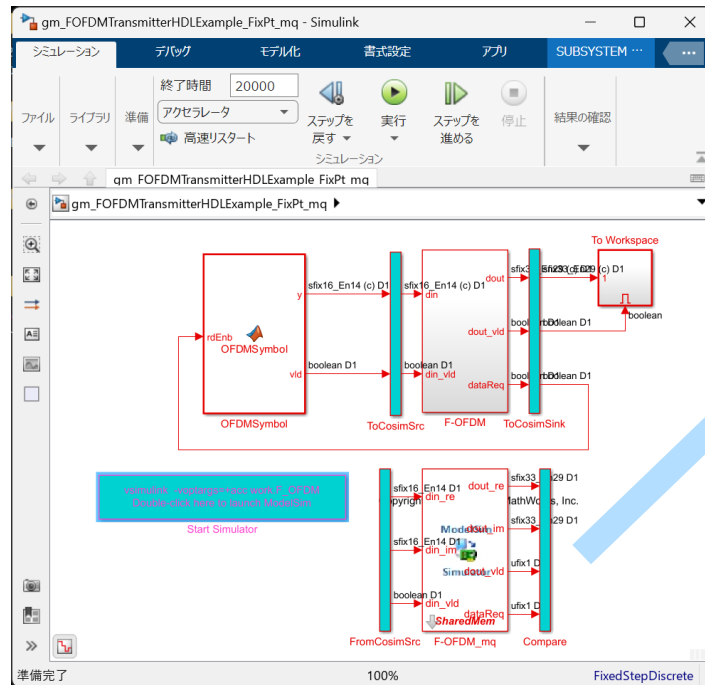
生成時間と検証実行時間を比較



# HDLコシミュレーションによる検証例

実装用モデルからHDLコシミュレーションモデルを自動生成可能

HDLコシミュレーションモデルを実行するだけでモデルと生成HDLコードの等価性を検証可能



※コシミュレーションモデル生成はHDL Coderの機能  
コシミュレーション実行はHDL Verifierの機能

# Step5 生成HDL/CコードのSoC FPGAボードへの実装と実機検証

エクスターナルモード機能によりSoC FPGAボードとPCをEthernetで接続し  
PC上のMATLABからボードの制御や動作状況のモニターが可能

1. システム開発  
仕様検討

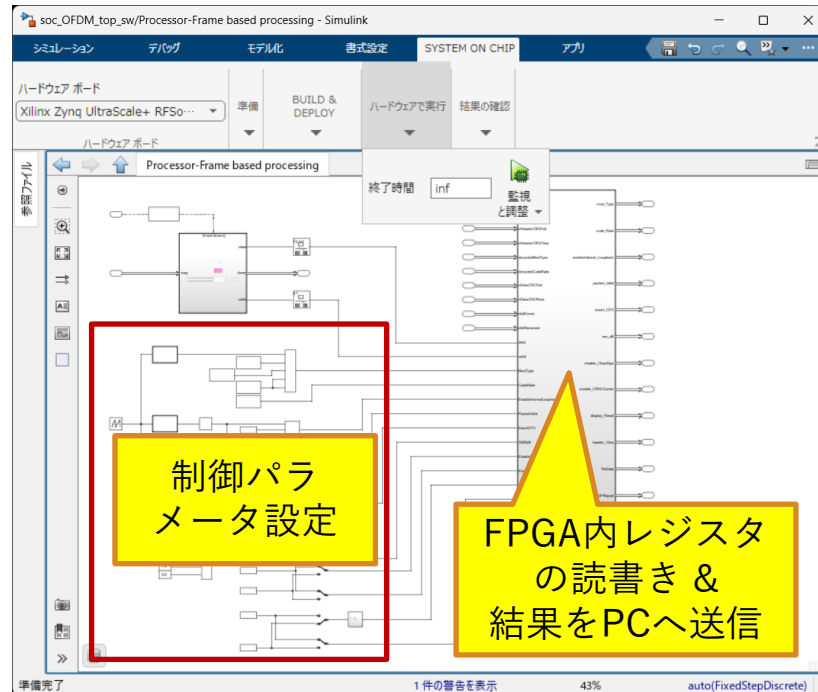
2. アルゴリズム  
作成と検証

3. 実装用モデル  
作成と検証

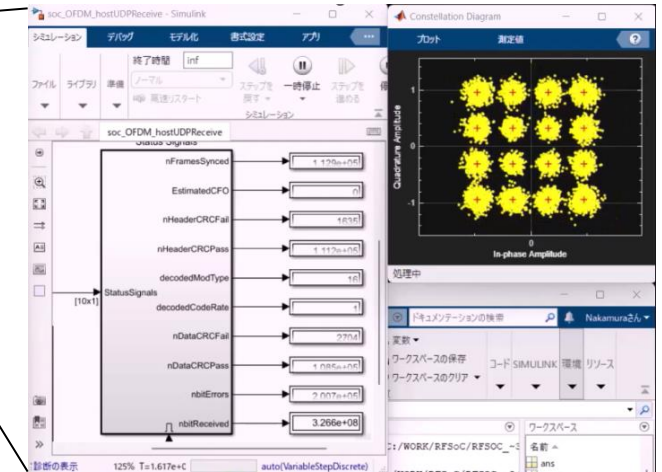
4. HDL/Cコード  
生成と検証

5. FPGAボード  
実装と検証

ARM用Cコード実装モデル



動作状況モニターモデル



MATLAB

Ethernet



ARM側：モデルから  
Cコード生成&ビルド  
ボードに転送し実行

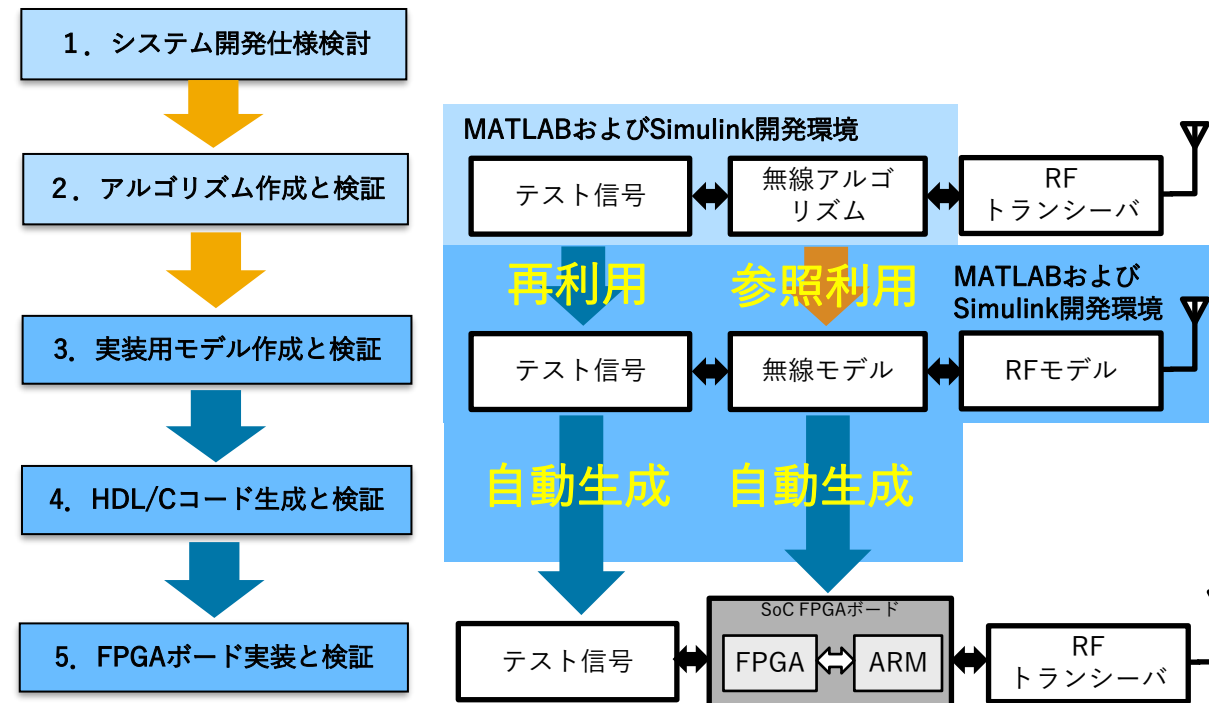
FPGA側：別途  
HDL生成&合成  
FPGA書込み

# モデルベースデザインによるSoC FPGA実装ワークフローのまとめ

**MBDはモデルを使った早期検証と自動コード生成によりQCDを向上**

作成したモデルを実装工程にまでコードの生成と検証に再利用します

- 工程間の齟齬が減り効率的な検証が可能
- 開発期間短縮や品質向上が期待できます





# 本日のセッションで利用したOFDM送受信モデル実装例の詳細

- 本実装例は、無償提供のSoC Blockset Support Package for Xilinx Devicesに付属

<https://jp.mathworks.com/matlabcentral/fileexchange/70616-soc-blockset-support-package-for-xilinx-devices>

- 本実装例のドキュメントページリンク

<https://jp.mathworks.com/help/soc/ug/OFDMTransmitandReceiveUsingXilinxRFSocDevice.html>

- 利用にあたって必要となるツール

- MATLAB, Simulink
- HDL Coder, MATLAB Coder, SoC Blockset
- Communication System Toolbox
- Wireless HDL Toolbox, DSP HDL Toolbox
- Embedded Coder, Simulink Coder
- AMD社ツール：Vivado 2022.1、ボード：ZCU111

The screenshot shows the MathWorks Help Center page for the document "OFDM Transmit and Receive Using Xilinx RFSoc Device". The page is titled "OFDM Transmit and Receive Using Xilinx RFSoc Device" and is dated R2023b. It includes a table of contents on the left with sections like "Documentation Home", "FPGA, ASIC, and SoC Development", "SoC Blockset", "Applications", "Wireless Communication", "SoC Blockset", "SoC Blockset Supported Hardware", "Xilinx Devices", and "Hardware I/O Devices". The main content area includes a "Supported Hardware Platforms" section listing "Xilinx Zynq UltraScale+ RFSoc ZCU111 evaluation kit + XM500 Balun card" and a "Design Task" section describing the design of a wireless communication system. A block diagram at the bottom illustrates the system architecture, showing a Simulink Software Model with Control Signals, Status Signals Display, and OFDM Transmitter/Receiver blocks, connected to an OFDM HDL block, which is then interfaced with Tx and Rx RF Interfacing blocks and antennas.



# 参考情報リンク

## FPGA/ASIC/SoC実装系資料まとめページ

MathWorks

共有

### FPGA/ASIC実装ツール

FPGA/ASIC実装ツールに関する技術コンテンツや役立つリンクをご紹介します。

#### 最新情報

##### Live Events

###### Implementing Motor and Power Electronics Control on an FPGA-Based SoC

Live webinar "Implementing Motor and Power Electronics Control on an FPGA-Based SoC"

###### Designing AI Engines of Xilinx Versal ACAP Using Simulink and Vitis Model Composer

On demand webinar "Designing AI Engines of Xilinx Versal ACAP Using Simulink and Vitis Model Composer"

###### FPGA および ASIC でのアルゴリズムの検証

eBook

新しいFPGAおよびASIC設計の経験には、新しいアルゴリズムの実装が含まれており、アルゴリズム開発者、ハードウェア設計者、検証エンジニアに検証の経験が示されます。

このeBookでは、MATLAB®とSimulink®を使用して次を行うことにより、検証の課題に対処し、開発作業を簡素化する方法について説明しています。

eBook: 「FPGAおよびASICでのアルゴリズムの検証」

#### お勧め技術資料

##### モデルベースデザインによるSoC FPGA/ASIC実装実装関連製品紹介

MathWorks Japan

モデルベースデザインによるSoC FPGA/ASIC実装・実...

PDF

##### モータ制御のFPGA実装向けHDLモデリングガイドライン

MathWorks Japan

モータ制御のFPGA実装向けHDLモデリングガイドラ...

PDF

##### HDLコード/FPGAデザインの検証効率化

MathWorks Japan

HDL検証セミナー

PDF



<https://mathworks.highspot.com/viewer/61c3fce10d1505ac35cc749b>

# NEC、宇宙開発・宇宙利用に向けた探査機と搭載機器の開発をモデルベース技術を使って迅速化

月スイングバイ航行技術や高速フライバイ探査技術の実証、さらにはデジタルとアナログが協調動作する搭載機器の機能設計のためにモデルベース技術を活用

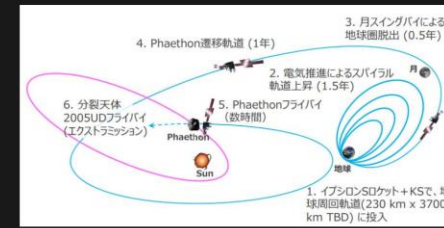
## 主な成果/利点:

- オートコーディングを推進し、計算負荷の試算の迅速化、コミュニケーション齟齬による後戻り防止、モデルの高い可読性/流用性を実現
- 宇宙空間での実機を用いた事前テストの代わりにシミュレーションを活用、プログラムの網羅的な検証やモデリング標準準拠を確認
- 搭載機器開発においても、自動コード生成機能、デジタルとアナログが混在する信号処理部や電源系のシステムのモデル化、仕様検討・機能確認などを活用

### DESTINY+の技術的挑戦

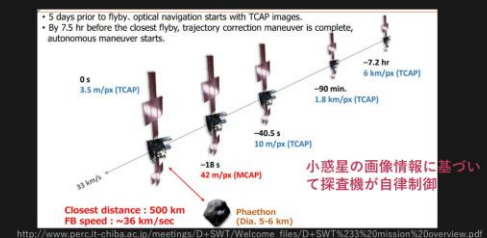
#### 電気推進によるスパイラル軌道上昇と月スイングバイ航行技術

電気推進を用いて、地球周回低軌道から深宇宙へ挑む世界初の技術



#### 高速フライバイ探査技術

Phaethonから約500km離れた地点を相対速度30km/s以上でフライバイし、フライバイしている数分間で、カメラによる表面撮像とダストアナライザによるダスト分析を実施



これらの技術は、探査機のオンボード処理による自律制御が必要

ミッションクリティカルで高度な技術である一方で、計算機リソースの制約も考慮する必要がある

Embedded Coder、Simulink Coverage、Simulink Checkなどを活用し、開発の迅速化と信頼性確保を実現

民生用として多くのユーザーによって培われてきたモデルベース技術と、宇宙で長年培われてきた品質基準・信頼性を融合することを目指しています。2024年の打ち上げと、その3年後のフライバイに向けて、モデルベース技術で開発の迅速化と高い信頼性を実現し、ミッションを達成します。



© 2023 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [www.mathworks.com/trademarks](http://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.