製造までの4つのステップ：モデル・ベース設計で実現するソフトウェア無線

Part 1：ADI/Xilinx社のSDR向けラピッド・プロトタイピング用プラットフォーム――その機能、メリット、開発ツールについて学ぶ

著者：Di Pu/Andrei Cozma/Tom Hill

概要

ワイヤレス・システムの開発では、概念設計から実際に動作する回路の設計までの間に大きなギャップがあります。このギャップを埋めるには、RF、ソフトウェア、D-SP、HDL（ハードウェア記述言語）、組み込みLinux®など、異なる領域を対象としたスキル・セットを有する技術者のチームが必要になります。しかし、それぞれに異なる視点で進められる個々の設計作業の調整を行うのは容易なことではありません。多くの場合、開発初期の段階でプロジェクトの計画から逸脱していくことになるでしょう。

このシリーズでは、4部構成でワイヤレス・システムの開発について説明していきます。その中で、ワイヤレス・システムの設計から製造までの間で実践可能な道筋を確立することを試みます。特に、開発者が容易にワイヤレス・システムのシミュレーションやプロトタイピングを行えるよう支援する最先端のプラットフォームやツールについて詳しく説明します。また、開発プロセスについて解説するうえでの実例として、ソフトウェア無線向けプラットフォームのプロトタイプを作成することにします。そのプラットフォームでは、ADS-B（Automatic Dependent Surveillance-Broadcast：放送型自動従属監視）信号を受信してデコード（復号化）することで、近くを飛ぶ民間航空機の位置、高度、速度の検知と報告を行えるようにします。そのために必要なリソースは、「MATLAB®」と「Simulink®」（いずれも、MathWorks社製）に加え、ハードウェア/ソフトウェアを統合して組み込むためのスキルです。ハードウェア・プラットフォームとしては、アナログ・デバイセズ（ADI）とXilinx社が提供するソフトウェア無線向けのプロトタイピング・システムを使用します。また、MATLABとSimulinkでは、以下のような作業を行います。

- ADS-Bのメッセージをデコードするための信号処理のアルゴリズムを設計する
- ADS-Bの信号を受信するRFトランシーバのシミュレーションを行う
- C言語とHDLのコードを生成する
- ターゲットとなるトランシーバとFPGA上で、記録済みのデータとライブ・データを使ってHDLのコードを検証する

最終的な成果物は、製品に相当するハードウェア上で実際に動作するRF対応のソフトウェア無線回路です。この無線回路を近くの空港に持ち込んで、性能と機能の検証も実施します。

本シリーズのPart 1では、ADIとXilinx社が提供するソフトウェア無線向けのプロトタイピング・システムを取り上げます。その機能と得られるメリットについて説明したうえで、ツールを使った開発フローを簡単に紹介します。Part 2では、MATLAB/Simulinkを用いたシミュレーションにより、ADS-Bの信号をデコードして情報を得る方法を説明します。Part 3では、HIL（Hardware in the Loop）を使用し、ターゲットとなるトランシーバによって信号を取得する方法を説明します。ただし、この時点では、信号処理は検証用のSimulink内のホスト上で行います。Part 4では、Part 2で開発し、Part 3で検証したアルゴリズムを基に、MathWorks社の「HDL Coder」と「Embedded Coder」を使用してコードを生成します。そのうえで、それらのコードをハードウェアに配備（デプロイ）する方法を示します。さらに、空港において実際のADS-B信号に対してプラットフォームを動作させます。

はじめに

通信の実現手段は継続的に進化し、現在では非常に多くの方式が使われています。今後もその数は増える一方だと考えられるため、ビジネスの観点からも、無線機器をコスト効率良く容易に変更できるようにすることが不可欠となっています。この要件に対応するために、最近ではソフトウェア無線（SDR：Software-defined Radio）を採用するケースが増えています。この技術は柔軟性とコスト効率が高く、通信をさらに進化させられるだけの能力を備えているからです[1]。SDRシステムでは、できるだけ多くの変調/復調処理とデータ処理のアリゴリズムを、ソフトウェアと再プログラムが可能なロジック回路で実行するようにします。それによって、ハードウェア・プラットフォームには一切変更を加えることなく、ソフトウェアと再プログラムが可能なロジック回路を更新するだけで、通信システムを容易に再構成することができます。

Xilinx社の「Zynq®-7000 All Programmable SoC（以下、Zynq SoC）」は、CPUの汎用性とFPGAの処理能力を組み合わせたSoC（System on Chip）です。このような製品が登場したことで、SDRシステムのデータ処理機能をその他の処理タスクと共に単一のデバイスに統合することが可能になりました。その場合、データの変調/復調アルゴリズムのような集中的な処理を要するタスクはFPGAで実行します。一方、データのデコードやレンダリング、システムの監視/診断、ユーザー・インターフェースなどのタスクはCPUで実行します。

ワイヤレス・システムのプロトタイピングについては、数十年も前から議論されてきました。そして、FPGAの利用を前提とすることで、そうしたプロトタイピング手法の1つが完全な設計フローとしてここ数年の内に構築されました。その設計フローでは、モデルの作成から最終的な実装までを網羅します。このようなことが可能になったのは、MATLABやSimulinkといったモデリング・ツールやシミュレーション・ツールが進化したためです。そうしたプロトタイピング手法によって、技術者や研究者の作業

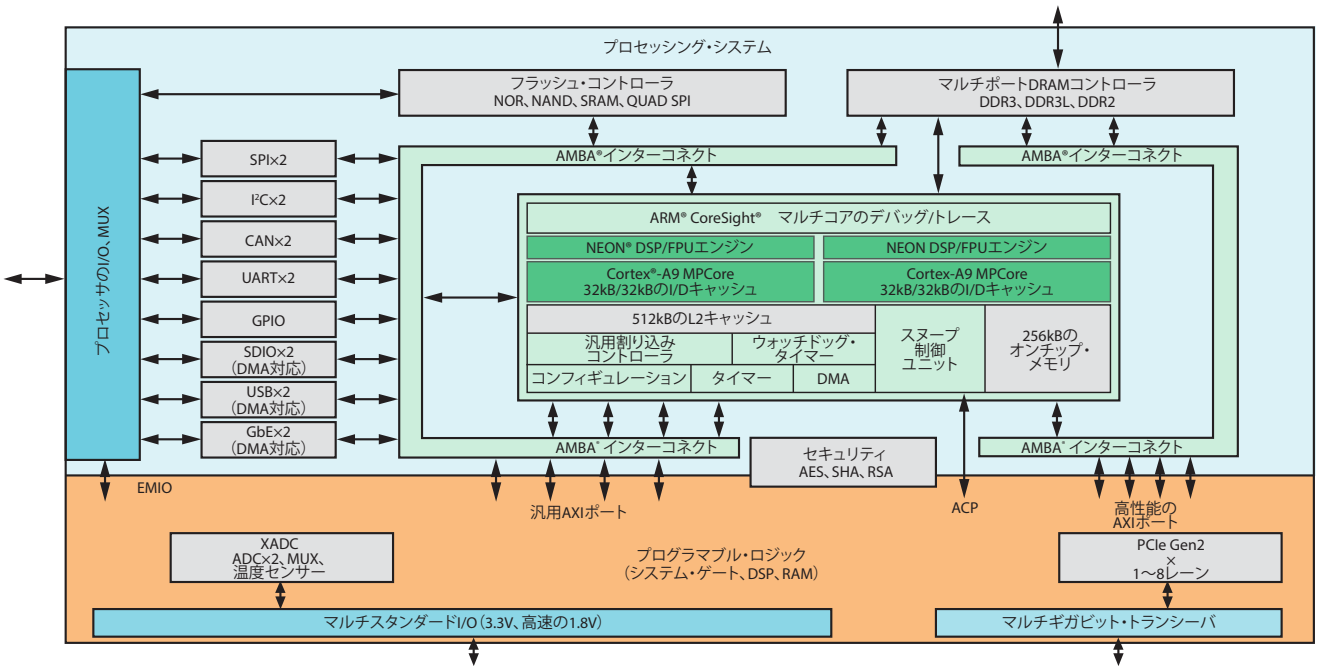

図1. Zynq SoCのブロック図

方法にも変化がもたらされました[2]。設計作業を行う場が、研究施設や現場からデスクトップへと移行したのです。現在ではSDRシステムのようなワイヤレス・システムの全体をモデル化することが可能となっています。技術者はシステムを実際に現場に導入する前に、その動作を確認してチューニングを行うことができます。これによりいくつかのメリットが得られます。例えば、システムの統合を迅速に行えるようになり、装置の可用性に対する依存度が低下します。また、SDRシステムのSimulinkモデルを一度完成させれば、Zynq SoCに実装するC言語やHDLのコードを自動的に生成することができます。このことは、開発期間の短縮と、手作業でコーディングを行うことで生じる初期エラーの回避につながります。システムのモデルを、現実的な条件下でSDRシステムを動作させられるラピッド・プロトタイピング環境にリンクさせることで、さらにリスクが軽減されます。

本シリーズのPart 1では、ADIとXilinx社が提供するソフトウェア無線向けのプロトタイピング・システムを取り上げます。その機能と得られるメリットについて説明するほか、ツールを使った開発フローを簡単に紹介します。そのうえで、ADIのRF IC技術と、リファレンス設計のハードウェア/ソフトウェアにより、広範なスキルを有していなくても設計を行うことが可能になり、リスクの軽減と市場投入までの時間の短縮を実現できることを示します。

Zynq SoCでSDRを実現

先進的なSDRシステムでは、データ処理、通信、ユーザー・インターフェースのタスクを組み合わせて実行しなければなりません。各タスクは、処理帯域幅に対する要件やリアルタイム性に関する制約の面でもそれぞれに異なります。このようなシステムの実装に使われるハードウェア・プラットフォームは堅牢であるだけでなく、将来発生するであろう改変や規模の拡大に対応できるように拡張性を備えている必要があります。図1に示すように、Xilinx社のZynq SoCは高性能のプロセッシング・システムとプログラマブル・ロジックを組み合わせたものであるため、そうした要件を満たすことができます[3]。プロセッシング・システムとプログラマブル・ロジックの組み合わせにより、優れた並列処理能力、リアルタイム性能、高速演算性能、多くの用途に対応可能な接続性が実現されます。

Zynq SoCのプロセッシング・システムは、「ARM® Cortex®-A9」のデュアルコア版、コプロセッサの「NEON」、ソフトウェアの実行を高速化する浮動小数点拡張機能で構成されています。デュアルコアのCortex-A9は組み込みLinuxまたはリアルタイムOSと組み合わせて使用可能であり、システムの能力を最大限に活用することができます。また、プロセッサはプログラマブル・ロジックを構成していなくても独立して使用することが可能です。この点は、ハードウェア技術者によるFPGAファブリックの設計が完了するのを待つことなく、コードの開発を行いたいと考えるソフトウェア開発者にとって不可欠な要素です。

プログラマブル・ロジック部には、最大44万4000個のロジック・セルと2200個のDSPスライスを集積しています。これらにより、莫大な処理能力が得られるため、Zynq SoCは、要件の厳しいさまざまな信号処理アプリケーションに対応することができます。プログラマブル・ロジックは、AMBA（Advanced Microcontroller Bus Architecture）-4 AXI（Advanced Extensible Interface）に対応する5つの高速インターコネクトによって、プロセッシング・システムに密に結合されています。このことから、3000本以上のピンを使用することに相当する有効帯域幅が実現されています[4]。

SDR向けのRFアジャイル・トランシーバIC

ここ数年間で、SDRならびにそのシステム・アーキテクチャに求められる要件は非常に厳しくなりました。それに対応すべく、ADIは革新的なSDR対応製品を市場に投入してきました。この分野における最も重要な製品としては、高い集積度と性能を誇るRFアジャイル・トランシーバ（RF Agile Transceiver™）IC「AD9361」、「AD9364」が挙げられます。AD9361[5]は2×2のトランシーバ、AD9364[6]は

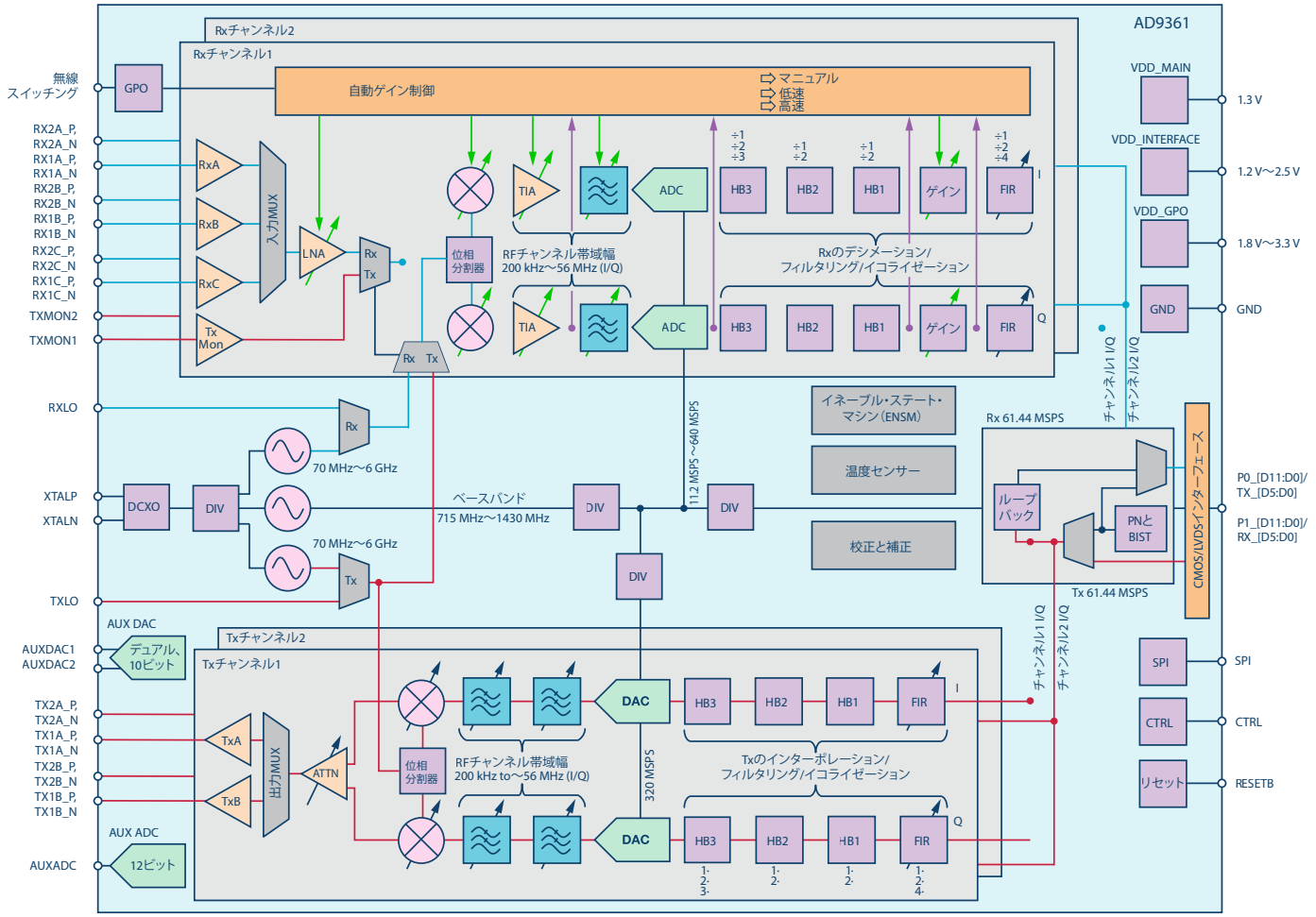

図2. AD9361のブロック図

1×1のトランシーバです。いずれも、ワイヤレス通信のインフラ、防衛用エレクトロニクス・システム、RFテスト装置/計装機器、一般的なSDRプラットフォームなどの分野におけるSDRアーキテクチャの用途をターゲットとしています。両ICは、RFフロント・エンド、柔軟性の高いミックスド・シグナルのベースバンド部、周波数シンセサイザを集積しています。また、プロセッサやFPGAに対する構成可能なデジタル・インターフェースを提供することによって、デザイン・インを簡素化します。対応周波数範囲は70MHz～6GHzで、免許を必要とする帯域と必要としない帯域の大部分をカバーします。さらに、サンプル・レート、デジタル・フィルタ、デシメーション処理もすべてAD9361/AD9364の内部でプログラムすることが可能です。それらを変更することにより、200kHz～56MHzのチャンネル帯域幅に対応できます[7]。図2に、AD9361のブロック図を示しました。

SDRに取り組んでいる企業は、製品の市場投入までにかかる時間と全体的な開発時間を短縮したいと考えています。ADIは、そうした顧客を支援するための一歩進んだ手段として、FPGAとのシームレスな接続性を備える完全なエコシステムの中でSDR向けのソリューションを提供しています。そのソリューションでは、無線システムの設計全体を網羅するラピッド・プロトタイピング環境と開発環境を実現しています。ラピッド開発/プロトタイピング・ボード「AD-FMCOMMSx-EBZ（以下、FMCOMMS）」は、高速アナログFMC（FPGA Mezzanine Card）モジュールのファミリー製品です。RFアジャイル・トランシーバICであるAD9361/AD9364、またはディスクリート構成のシグナル・チェーンを搭載する同ボードは、Xilinx社のFPGA開発プラットフォームのエコシステムにシームレスに接続できます。また、各ボードはハードウェアを一切変更することなく、ソフトウェアによって完全にカスタマイズすることが可能です。付属のLinuxドライバとベア・メタルのソフトウェア・ドライバ、回路図、ボードのレイアウト・データ、設計支援用のドキュメントは、すべてAnalog Devices Wikiの各サイトからダウンロードできます。表1に、各FMCOMMSプラットフォームの特徴をまとめました。

表1. FMCOMMSプラットフォーム

プラットフォーム	特徴
AD-FMCOMMS5-EBZ	2×2のRFアジャイル・トランシーバICであるAD9361を2個搭載するSDRラピッド・プロトタイピング・ボード。4つのレシーバ（Rx）チャンネルと4つのトランスミッタ（Tx）チャンネルの完全な同期をとり、4×4 MIMOシステムのあらゆるサブセットを構築できる。70MHz～6GHzという広い帯域に対応するポートと、2.4GHzにチューニングされたポートを備えている AD-FMCOMMS5-EBZのリソース（Wikiページ）：http://wiki.analog.com/resources/eval/user-guides/ad-fmcomms5-ebz

AD-FMCOMMS4-EBZ	RFアジャイル・トランシーバICであるAD9364を搭載する1×1のSDRラピッド・プロトタイピング・ボード。2400～2500MHzの範囲で最高のRF性能が得られるようにするか、あるいはAD9364の全RFチューニング範囲である70MHz～6GHzで動作させるのかをソフトウェアで設定し、システムのプロトタイピングや開発を行うことができる AD-FMCOMMS4-EBZのリソース（Wikiページ）：http://wiki.analog.com/resources/eval/user-guides/ad-fmcomms4-ebz
AD-FMCOMMS3-EBZ	RFアジャイル・トランシーバICであるAD9361を搭載する2×2のSDRラピッド・プロトタイピング・ボード。AD9361の全チューニング範囲である70MHz～6GHzに対応する。広範囲でチューニングが可能な統合開発プラットフォームを必要とするワイヤレス通信SDRシステムの設計者に最適なキットである AD-FMCOMMS3-EBZのリソース（Wikiページ）：http://wiki.analog.com/resources/eval/user-guides/ad-fmcomms3-ebz
AD-FMCOMMS2-EBZ	RFアジャイル・トランシーバICであるAD9361を搭載する2×2のSDRラピッド・プロトタイピング・ボード。2400～2500MHzの範囲で最高のRF性能が得られるようにチューニングされている。この周波数範囲内で、AD9361のデータシートに記載されたとおりの最高のシステム性能を必要とするRF技術者に最適なキットである AD-FMCOMMS2-EBZのリソース（Wikiページ）：http://wiki.analog.com/resources/eval/user-guides/ad-fmcomms2-ebz

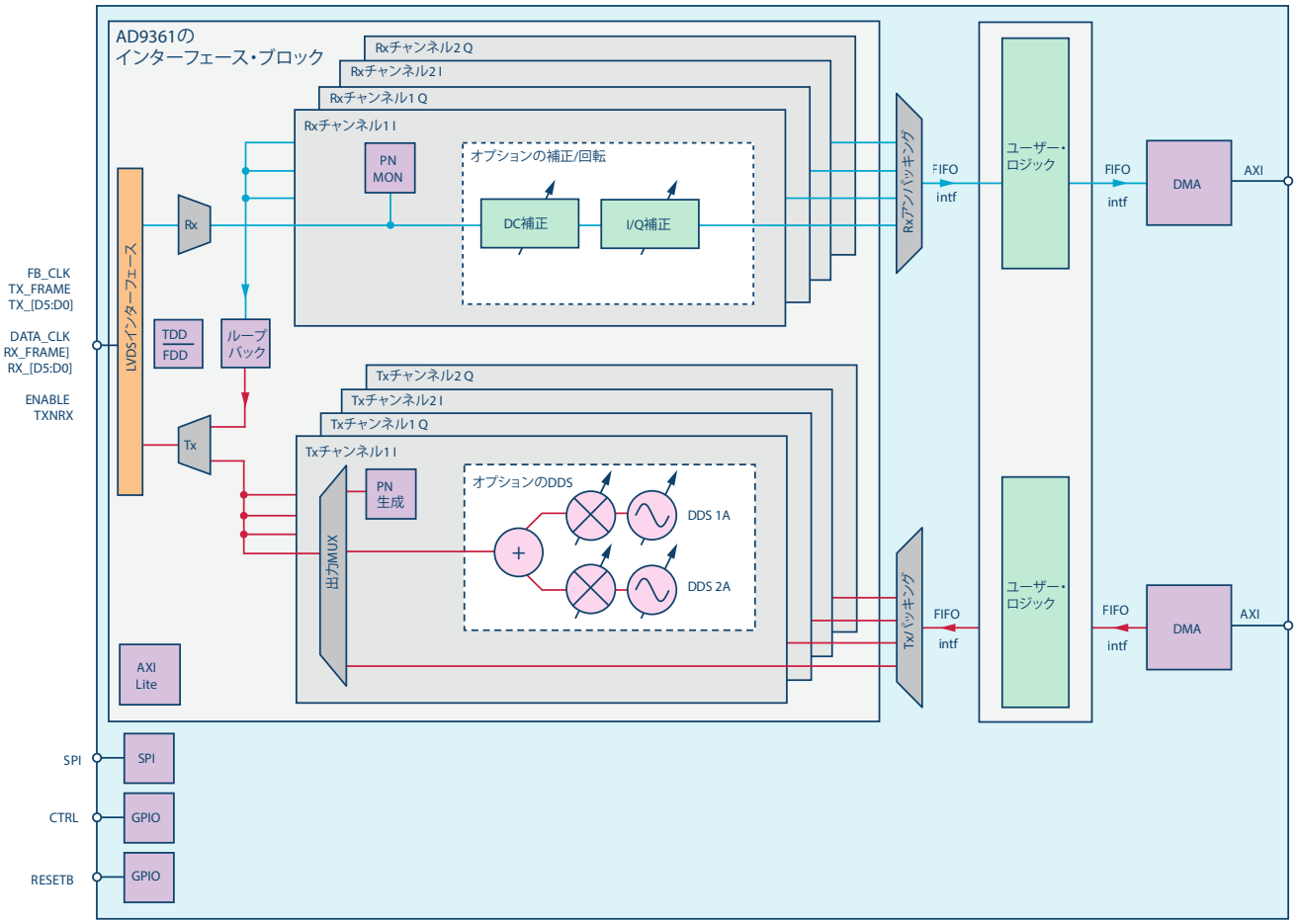

図3. HDL/ソフトウェア向けのインフラ

Zynq SoCを利用したSDR向けプラットフォームのリファレンス設計

ADIはFMCOMMSプラットフォームと共に、完全な「Vivadoフレームワーク」を提供しています。このフレームワークには、プロトタイピングのためだけでなく、最終的に製造するシステムの一部としても使用可能なLinuxとベア・メタルのソフトウェア・インフラが含まれています。図3に、FMCOMMSにおけるZynq SoCに対応するためのインフラを示します。

このハイ・レベルの構成図には、ADIのリファレンス設計がZynq上で分割される様子が示されています。Linuxのインターフェースは、HDMI出力を利用してモニターに表示されます。また、キーボードとマウスはUSB 2.0ポートによってシステムに接続できます。ARM Cortex-A9をベースとしたプロセッシング・システムは、ADIが提供する「Ubuntu（Linux）」を搭載しています。これには、FMCOMMSのハードウェアとの通信に必要なLinux IIOドライバ、監視/制御用のユーザー空間アプリケーション「IIO Oscilloscope（Scope）」[8]、リモートのコンピュータ上で動作するクライアント・ソフトと連携してTCP（Transmission Control Protocol）

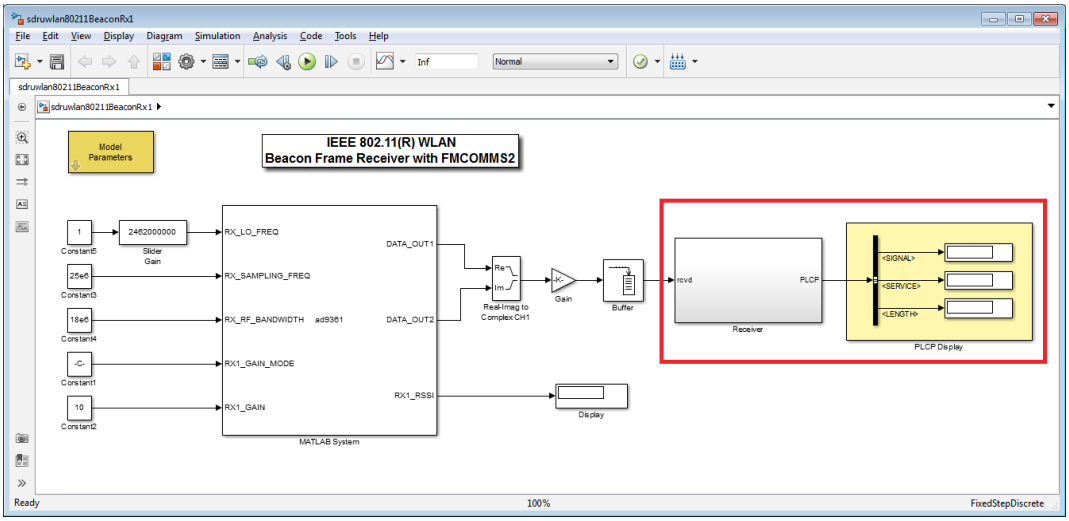

図4. ビーコン・フレーム・レシーバのサンプル（スクリーン・ショットの抜粋）

を介したリアルタイムのデータ取得とシステム制御を可能にするサーバー・ソフト「libiio」[9]、Simulinkモデルから Embedded Coderによって生成されたC言語のコードが組み込まれるオプションのユーザー・アプリケーションが含まれています。

ソフトウェア・インフラ

ADIのLinuxドライバは、いずれも「Linux Industrial I/O（IIO） Subsystem」をベースにしています。現在、このサブシステムはすべての主要なLinuxカーネルに含まれています。IIO ScopeはADIが開発したオープンソースのLinuxアプリケーションです。Zynq SoCが搭載するデュアルコアのCortex-A9上で動作し、Zynq SoCのプラットフォームに接続された任意のADI製FMCカードから取得したデータをリアルタイムに表示することができます。表示形式としては、時間領域、周波数領域、コンスタレーションに対応します。また、取得したデータを保存してさらに詳しく解析できるように、CSVファイルやMATLAB形式ファイル（拡張子はmat）といったファイル形式をサポートしています。IIO Scopeは、ADIのFMCカードの構成を変更したり、再度読み込んだりするためのGUIを備えています。リモートのコンピュータ上ではクライアント・ソフトが実行され、それと連携して、TCPを介したリアルタイムでのデータの取得とシステムの制御が行われます。これを可能にするのがlibiioサーバーです[10]。このサーバーは、Linuxを搭載する組み込みターゲット上で動作するほか、ターゲットとリモート・クライアントの間で行われるTCPを介したリアルタイムのデータ交換を管理します。このライブラリは、ハードウェアの低レベルの詳細を抽象化する役割を果たします。そして、シンプルでありながら、高度なプロジェクトにも利用できる完全なプログラミング・インターフェースを提供します。モジュール式のアーキテクチャ、優れた設計のAPI（Application Programming Interface）、組み込みネットワーク機能によって、ユーザーは、IIOに対応するデバイスが接続されているシステム上で実行されるアプリケーションだけでなく、ネットワークを介してリモートで実行されるアプリケーションも開発することが可能です。当初はLinuxをターゲットにしていましたが、ライブラリのリモート・バックエンドを使用することにより、現在ではWindows上でも同様に使用できるようになっています。このライブラリはC言語で記述され、LGPL（GNU Lesser General Public License）の下でライセンス提供されています。C#、Python、MATLABとのバインドが可能です。MathWorks社のIIOクライアントが、MATLABとSimulinkのネイティブ・アプリケーションに統合されるシステム・オブジェクトとして提供されます。ADIのLinuxディストリビューションはFPGA/SoCプラットフォームが実行します。IIOクライアントは、同プラットフォームに接続されたADIのハードウェア・システムとEthernetを介したデータ交換を行うために設計されています。これにより、MATLABまたはSimulinkのモデルは、以下のような機能を実行できます。

- ターゲットとの間で行うデータのストリーミング
- ターゲットの設定の制御
- ターゲットのさまざまなパラメータの監視

「IIO System Object™」は、ユーザーがそれをMATLABスクリプトから呼び出すか、MATLAB Systemブロックに組み込むことによって、MATLABとSimulinkの両方で使用できます。ADIがFMCOMMSプラットフォーム用に提供するLinuxソフトウェアとHDLのインフラを、MathWorks社とXilinx社が提供するツールと共に使用すれば、SDRアプリケーションのプロトタイピングに最適な環境が得られます。SDRシステムに組み込むことのできる製品レベルのコンポーネントも含まれているため、概念設計の段階から製造までにかかる時間を短縮し、コストを削減することが可能になります。

ユーザーがIIO System Objectを即座に容易に使い始められるように、ADIはそのインターフェースに基づくMATLAB/Simulinkのサンプルを複数提供しています。例を挙げると、ビーコン・フレーム・レシーバ[12]、QPSKトランスミッタ/レシーバ[13]、LTEトランスミッタ/レシーバ[14]などがあります。これらのサンプルにおいて、FMCOMMSプラットフォームは、IIO System Objectによって設定され、アナログ信号を無線で送受信するRFフロント・エンドとして使われています。送受信される信号は、IIO System Objectを介してターゲットとの間でストリーミングされます。その他の信号処理はすべて、MATLABまたはSimulinkで行われます。図4に、ビーコン・フレーム・レシーバのサンプルを示しました。IIO System Objectと他のSimulinkブロックの間で標準的な接続が行われています。

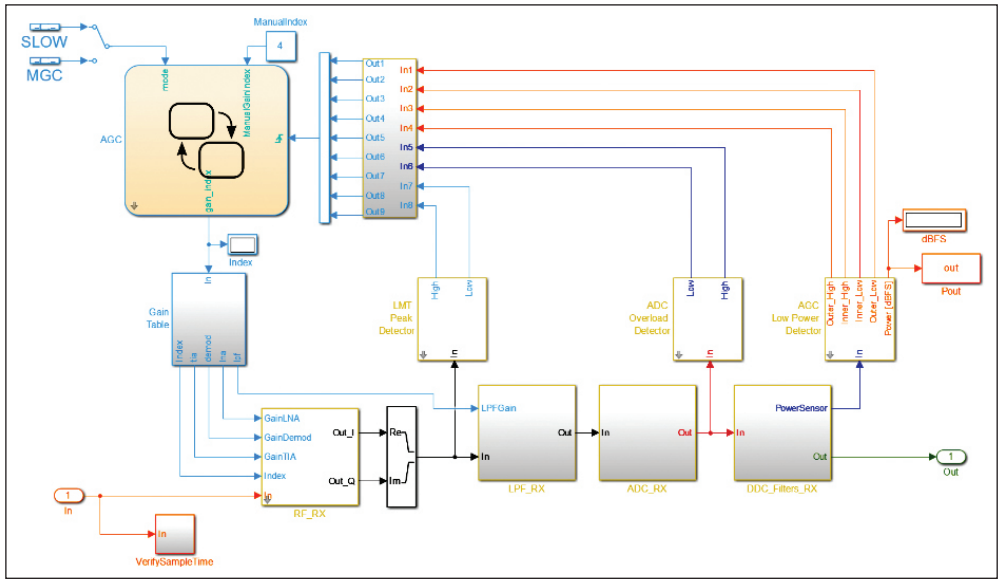

図5. AD9361のSimRFモデル

Zynqに対するMathWorks社のサポート

MathWorks社は、Zynq SoCをベースとしたSDRの実現を4つの側面からサポートしています。以下、それぞれについて説明します。

1. AD9361のSimulinkモデル

AD9361は集積度の高いRFトランシーバICであり、内部信号のプローブや内部動作の監視を行うことはできません。そこで、MathWorks社とADIはAD9361のSimRF™モデルを共同で開発しました。現実の環境では再現しにくいものも含めて、さまざまな条件下における同ICの内部動作をシミュレーションによって詳しく観測することができます。SimRFでは、ベースバンド・ブロックや、アンプ、ミキサー、Sパラメータ・ブロックなどの回路エンベロープ・ブロックを使ってRFシステムを設計するためのコンポーネント・ライブラリとシミュレーション・エンジンが提供されています。SimRFは、RFアジャイル・トランシーバであるAD9361をモデル化するための適切かつ便利なツールです。図5にAD9361のシステム・レベルのモデルを示しました。これはAD9361の機能をそのまま再現しており、MathWorks社のハードウェア・サポート・パッケージによってユーザーに提供されています[15]。

SimRFモデルについては、研究施設におけるパワー・スペクトル測定によって検証済みです。トランシーバのノイズと非直線性は、異なる周波数とパワー・レベルにおける特性評価によって確認しています。それと同じ特性を示すようにモデルを設計し、同等の値が得られることを確認しています。

AD9361のSimRFモデルを使用することにより、以下のようなことが行えます。

- RFの不具合がテストの際に信号に与える影響を予測する
- リファレンスとなるトーン信号やLTEの信号を使用できる
- テスト用のベクトルを生成またはインポートし、RFトランスミッタ/レシーバに起因する非直線性、ノイズ、ゲイン、位相の不均衡、スペクトル・リークといった不具合の影響を評価する
- 干渉信号を加えて、その結果を時間領域/周波数領域で評価する

2. 各種ツールの提供

MathWorks社は「Communications System Toolbox™」[16]、「Signal Processing Toolbox™」[17]、「DSP System Toolbox™」[18]、SimRF[19]といったツールを提供しています。これらを使えば、SDRシステムを体系的に解析/設計/チューニングするための業界標準のアルゴリズムとアプリケーションを活用できます。また、これらのツールを使用することで、忠実度の高いSDRのモデルを作成することも可能です。そのようなモデルを使用することで、実際に実装に移る前にシステムの動作と性能の検証を行うことが可能になります。

3. Zynq SoC向けのSimulinkのワークフロー

MATLABとSimulinkは、マルチドメイン・シミュレーションとモデル・ベース設計のための環境です。これらは、通信アルゴリズムを使用するSDRシステムのシミュレーションに非常に適しています。通信アルゴリズムは、トランスミッタ・システムとレシーバ・システムの間でよりよい同期を実現することを主目的とし、ゲイン、周波数オフセット、タイミング・オフセットといった性能にかかわる変数の値を調整するために使用します。

通信アルゴリズムの評価は、実機でも行えますが、シミュレーションによって行うことも可能です。ハードウェアを使用するテストには相応のコストがかかります。そのようなテストを行う前にシミュレーションを実施するのは、SDRシステムの設計の妥当性を判断し、アルゴリズム開発の時間とコストを削減するための有効な手段になります。図6は、通信アルゴリズムを設計するための効率的なワークフローを示したものです。このワークフローの図において、青色の網掛け部分では、以下のような手順で作業を行います。

- モデル・ベース設計用の環境で提供されているライブラリを使用して、SDRのモデルを正確に作成する
- システムのシミュレーションを行い、期待通りに動作することを確認する
- リアルタイム・テストと実装用にC言語とHDLのコードを生成する

• プロトタイプのハードウェアを使用して通信アルゴリズムのテストを行う

シミュレーションを実施し、プロトタイプのハードウェアを使ったテストを行うことにより、SDRシステムの性能が要件を満たすことが実証されれば、安心してその設計を基に最終的なシステムの実装を行うことができます。

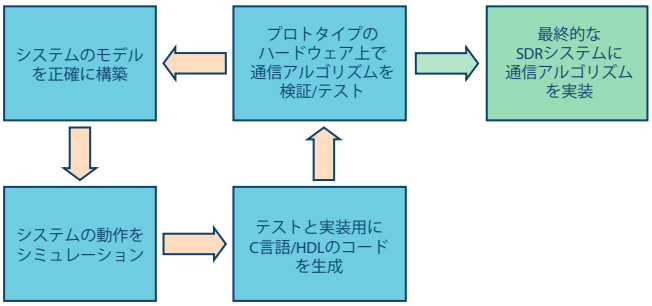

図6. 通信アルゴリズムを設計するためのワークフロー

4. Zynq SoCのSDR用キットにSimulinkプラットフォームを統合

シミュレーション環境では、Embedded Coder®[20]やHDL Coder™[21]などのツールを使用してSDRシステムの完全な検証を行います。それが完了したら、Embedded CoderでC言語のコードを生成し、HDL CoderでVHDL/Verilogのコードを生成します。次に、それらのコードをプロトタイプのハードウェアに配備してテストを行います。テストの結果に問題がなければ、それらのコードを最終的なシステムに配備することもできます。これらの作業を行っている時点で、小数点の扱いや動作タイミングといったソフトウェア/ハードウェアの実装における要件が決定されます。コードの自動生成は、概念設計からシステムの実装までに要する時間の短縮と、手作業でコーディングを行うことで生じる初期エラーの回避につながります。さらに、コードの自動生成を利用することで、SDRのモデルと実装が一致することが保証されます。

図7は、SimulinkにおいてSDRシステムをモデル化し、そこからZynq SoCをベースとする最終的なシステムに移行するまでに必要な手順を表しています。最初の作業は、SimulinkでSDRシステムをモデル化してシミュレーションを実施することです。この段階で、ソフトウェアで実装するブロックとプログラマブル・ロジックで実装するブロックに通信アルゴリズムを分割します。シミュレーションと分割が完了したら、Embedded CoderとHDL Coderを使用してSDRのモデルからC言語のコードとHDLのコードを生成します。そして、Zynq SoCをベースとするプロトタイピング・システムによって通信アルゴリズムの性能を検証し、製造段階に移行する前にSDRのモデルをさらにチューニングします。製造段階では、自動生成されたC言語/HDLのコードを複雑な製品システムのフレームワークに実装します。このワークフローによって、製造段階に達した時点で通信アルゴリズムの検証/テストが完全に行われたことが保証されます。その結果、システムの堅牢性の面で高い信頼性を確保することが可能になります。Embedded CoderとHDL Coder向けのZynq Hardware Support Packagesには、Zynq SoCプラットフォームのプログラミングが容易に行えるように、集積度の高いハードウェアとソフトウェアを設計/シミュレーション/検証するためのフレームワークが用意されています。モデル・ベース設計がワークフローに組み込まれたこのフレームワークでは、設計サイクルを迅速に繰り返すことが可能です。それにより、仕様/設計上の誤りを早い段階で検出して修正することができます[22]。

まとめ

本稿では、SDRに対応する最新システムの要件と動向について説明しました。MathWorks社、Xilinx社、ADIは、そのような要件を満たし、より性能の高いSDRソリューションの開発を支援するために、ツール/システムを市場に提供しています。そうしたツール/システムとしては、MathWorks社が提供するモデル・ベース設計ツールとコードの自動生成ツール、Xilinx社が提供する高性能のZynq SoC、ADIが提供する集積度の高いRFトランシーバが挙げられます。これらを組み合わせて利用することにより、SDRシステムの設計/検証/テスト/実装をかつてないほど効率的に実施できるようになります。結果として、無線システムの性能が向上するだけでなく、製品を市場に投入するまでにかかる期間の短縮が可能になります。ADIのFMCOMMSプラットフォームとXilinx社のZynq SoCを併用すれば、MathWorks社のMATLAB/Simulinkを使用して設計したSDR用の通信アルゴリズムに対する優れたプロトタイプ環境が得られます。FMCOMMSプラットフォームには、システムを評価する際の出発点として使用でき、任意のSDRプロジェクトに着手できるよう支援することを目的としたオープンソースのリファレンス設計が用意されています。

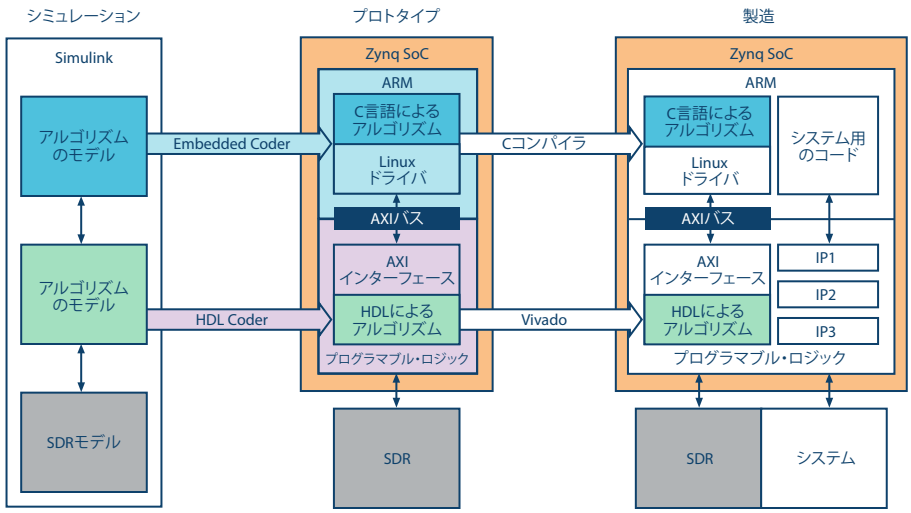

図7. シミュレーションから製造までの流れ

本稿の続編となるPart 2では、SDRの設計プロセスへと話を進めます。ADS-B信号の特性を示し、その信号をMATLAB/Simulinkを用いたシミュレーションにおいてデコードする方法を紹介します。

本稿で取り上げた内容の詳細、ドキュメント、ビデオ、リファレンス設計などについては、各参考文献をご覧ください。

参考文献

1 「What is Software-Defined Radio?」Wireless Innovation Forum

2 「Model-Based Design」MathWorks社

3 「Zynq-7000 All Programmable SoC」Xilinx社

4 Tom Hill「Motor Drives Migrate to Zynq SoC with Help from MATLAB」Xcell Journal, Issue 87, Second Quarter, 2014年

5 AD9361

6 AD9364

7 「Software-Defined Radio Solutions from Analog Devices」Analog Devices

8 「IIO Oscilloscope」Analog Devices Wiki

9 「Simulink Libiio」Analog Devices Wiki

10 「What Is Libiio?」Analog Devices Wiki

11 「IIO System Object」Analog Devices Wiki

12 「Beacon Frame Receiver Example」Analog Devices Wiki

13 「QPSK Transmitter and Receiver Example」Analog Devices Wiki

14 「LTE Transmitter and Receiver Example」Analog Devices

15 AD9361

16 「Communications System Toolbox」MathWorks社

17 「Signal Processing Toolbox」MathWorks社

18 「DSP System Toolbox」MathWorks社

19 「SimRF」MathWorks社

20 「HDL Coder」MathWorks社

21 「Embedded Coder」MathWorks社

22 「Xilinx Zynq Support from Simulink」MathWorks社

著者：

Di Pu（di.pu@analog.com）は、ADIのシステム・モデリング・アプリケーション・エンジニアです。ソフトウェア無線（SDR）に対応するプラットフォームやシステムの設計/開発をサポートしています。特にMathWorks社と密に連携することで、両社の顧客が抱える課題の解決に取り組んでいます。2007年に中国南京にある南京理工大学（NJUST）で理学士の学位を取得しています。また、マサチューセッツ州ウースターにあるウースター工科大学（WPI）で、2009年に電気工学の修士号、2013年に博士号を取得しました。WPIでは、2013年の「Sigma Xi Research Award for Doctoral Dissertation」を受賞しています。

Di Pu

Andrei Cozma（andrei.cozma@analog.com）は、ADIのエンジニアリング・マネージャーとしてシステム・レベルのリファレンス設計の開発を支援しています。産業用オートメーションと情報科学に関する学士号に加えて、電子工学と電気通信工学の博士号を取得しています。モーター制御、産業用オートメーション、ソフトウェア無線（SDR）、電気通信など、さまざまな分野にわたって設計/開発プロジェクトに従事した経験を持ちます。

Andrei Cozma

この著者が執筆した他の技術文書

FPGAベースのシステムで、モーター制御の性能を向上

Analog Dialogue 49-03

Tom Hill氏（tom.hill@xilinx.com）は、Xilinx社のシステム・ジェネレータ製品マネージャーです。18年以上にわたって、EDA分野の業務に従事してきた経験を持ちます。現在は、Xilinx社のDSP用ターゲット デザイン プラットフォームに関連するすべての製品、戦略、コーポレート・マーケティング活動を統括しています。それ以前はXilinx社が買収したAccelChip社の技術マーケティング・マネージャーとして、製品のディレクションや、DSP向けの高度な設計手法/ツールを担当していました。AccelChip社の前は、製品マネージャー、技術マーケティング・マネージャー、技術マーケティング・エンジニア、フィールド・アプリケーション・エンジニアとして、FPGA/ASIC向けのさまざまな合成ツールに取り組んでいました。さらに以前は、Allen-Bradley社やLockheed社でハードウェア/ASICの設計技術者としてキャリアを積んでいました。クリーブランド州立大学で電気工学の学士号を取得しています。

Tom Hill