ホワイトペーパー

5G NR 無線通信の FPGA への展開: MATLAB と Simulink の 完全なワークフロー

MathWorks®

はじめに

アルゴリズムの技術革新は、無線通信の進歩を促進し、個人間の接続、宇宙通信および衛星通信、高信頼性自動運転システム、IoT (Internet of Things) の発展を可能にします。これらのシステムを設計、実装、テストするには、複数の専門分野にわたる密接な共同作業が必要です。

アルゴリズムモデルを FPGA ハードウェアに展開することで、無線テストおよび検証が可能になります。システムレベルのアルゴリズムやモデルから直接 HDL コードを自動生成することで、エンジニアは仕様書に頼ったり、コードを手書きで設計したり記述する必要がなくなります。

このホワイトペーパーでは、MATLAB® アルゴリズムや Simulink® モデルを FPGA 用の HDL に直接変換するプロセスについて説明します。トピックには以下の内容が含まれます。

- MATLAB および 5G Toolbox を使用した 5G New Radio (NR) 規格に準拠したアルゴリズムモデリング
- フレームベースの MATLAB アルゴリズムから Simulink のストリーミング実装への移行
- Fixed-Point Designer™ を使用した固定小数点の実装とターゲットハードウェアの知識
- 実証済みの IP ブロックを使用した設計時間の短縮
- ターゲットハードウェア (今回は Xilinx® Zynq® デバイス) 向け HDL の生成および展開

プロセスの解説には、5G NR セルサーチ設計を使用します。MathWorks は当初、この設計をお客様の重要な要件を満たすために作成しました。その後、この設計は Wireless HDL Toolbox™ で利用可能なリファレンス アプリケーションに進化しました。このワークフローでは、5G Toolbox および HDL Coder™ も使用します。

無線通信アルゴリズムのハードウェアへの実装

通信エンジニアは、MATLAB および Simulink をアドオンのツールボックスとともに使用して、アルゴリズムレベルでのアプリケーションの開発、シミュレーション、および調整を行うことができます。ただし、これらのアルゴリズムをプロトタイピングや量産展開のために実装するには、幅広い役割間での調整が必要となり、プロジェクトの複雑度が大幅に増加します。

異なるスキルと作業環境

図 1 は、無線通信アプリケーションをハードウェアで構築するために必要なスキルを示したものです。エンジニアは通常、これらのうち 1 つまたは 2 つの分野に関する専門知識しか持たないため、プロジェクトでは多くの場合、役割や部門を超えた調整が必要になります。

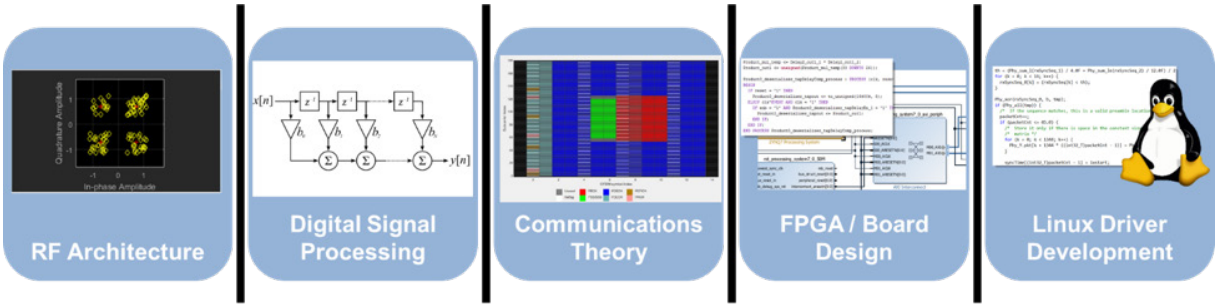

図 1. 無線通信アプリケーションをハードウェア上でプロトタイプ化または展開するために必要なスキルと作業環境。

ハードウェア設計の相違点

FPGA または ASIC デバイス上でのハードウェアのプログラミングは、汎用プロセッサ上で実行する命令セットにコンパイルされるソフトウェアの記述とは大きく異なります。デジタルハードウェアは、データの流れに基づいて動作する、あらかじめ用意された固定リソースの回路です。ハードウェアは高速化をもたらしますが、その構成には専門知識が必要です。大まかな相違点として、以下のようなものがあります。

- ストリーミングデータ: ソフトウェアには、任意のサイズや任意の次元数のデータを読み込むことができます。ただし、ハードウェアは、1 と 0 をつなぐ回路です。つまり、アルゴリズムは経時的にデータストリームに適応し、データの流れを管理するためのロジックを含んでいる必要があります。
- 並列処理: ハードウェア回路では、データストリームの処理を並列実行して高速化できますが、想定どおりにデータが到着するように、並列パスのタイミングを下流での処理向けに調整する必要があります。
- 固定小数点データタイプ: 浮動小数点演算は、一般的に固定小数点よりも多くの演算と長いデータ語長を必要とします。ハードウェアはリソースセットが固定されており、ほとんどのチームができるだけ小さいチップを使用したいと考えているため、ハードウェア設計では固定小数点が幅広く使用されています。しかし、浮動小数点アルゴリズムを固定小数点ハードウェアに対応する実装に変換するプロセスは、経験豊富なハードウェア設計者にとっても難しいものです。
- リソース使用量とクロックサイクルの管理: 最初に FPGA をターゲットにしようとすると、デバイスのリソースに適合しなかったり、想定より実行速度が遅かったりすることがよくあります。これらの相反する 2 つのパラメーターを最適化することは、ハードウェア設計において習得に時間がかかることの 1 つです。
- ソフトウェアや他のハードウェアデバイスとの相互作用: システムオンチップ (SoC) デバイスは、少なくとも 1 つのプロセッサと FPGA ハードウェアファブリックをデバイス上に組み合わせたものです。ハードウェアとソフトウェアは、あらかじめ指定されたアドレスのメモリの場所を経由して、相互にデータ通信を行います。これらのレジスタやその他の外部メモリ、およびチップ I/O にアクセスするには、デバイスのアーキテクチャを理解している必要があります。

FPGA ハードウェアのターゲット化を成功させるためには、上記のパラメーターを考慮して、アルゴリズムを効率的に動作させることが必要です。通信/信号処理エンジニアとハードウェア設計担当者の密接な共同作業により、最良の結果が得られます。仕様書や合成結果レポートを使用してコミュニケーションしているエンジニアは、革新的なアルゴリズムを FPGA ハードウェアに効率的に実装することができません。これには、密接に共同作業を行い、設計を模索する必要があります。

ワークフロー: アルゴリズム設計とハードウェア設計の分断を橋渡しする

ジョブに適したツール

MATLAB は、数学的アルゴリズムの開発、大規模なデータセットの操作、数学の探求、データの可視化に適しています。ただし、ハードウェアの実装には、タイミング、並列処理、およびカスタムの固定小数点の語長が必要です。これらの構造は、VHDL® や Verilog® などの専用のハードウェア記述言語 (HDL) では普通ですが、これらの HDL は MATLAB との接点がありません。このため、アルゴリズム設計者がその意図を仕様書に記述し、詳細な HDL や検証用テストベンチを記述する設計者がこれを解釈する必要があります。

Simulink は、MATLAB と緊密に連携し、アルゴリズム設計とハードウェア実装の橋渡しをする、視覚的な設計およびシミュレーション環境です。Simulink に備わっているタイミングを使用して、並列アーキテクチャの設計とシミュレーションを行い、設計全体にわたるデータ型を可視化することができます。また、Simulink では組み込みソフトウェアやアナログ回路をモデル化することにより、ハードウェアで実装する前に、システム全体のハイレベルなシミュレーションを行うことができます。実装の準備が整うと、HDL Coder や Embedded Coder® を使用して論理合成可能な HDL や組み込み C コードを生成し、ターゲットデバイスに展開することができます。

このため、図 2 に示すように、アルゴリズムおよびハードウェア設計の共同作業環境として、多くのチームが Simulink を使用しています。

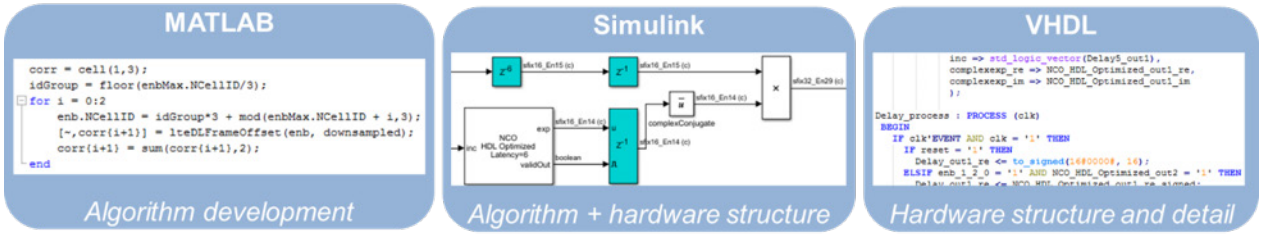

図 2. アルゴリズムおよびハードウェアの設計環境。アルゴリズム開発者は MATLAB を使用し、ハードウェア設計者は VHDL や Verilog を使用します。Simulink はこのギャップを埋め、この 2 つの業務を連携させてさらに高品質な実装に向けた共同作業を行うことを可能にします。

設計の概要

このワークフローを説明するために、5G NR 無線設計を使用しますが、説明する原則はどのような無線設計にも当てはまります。

NR HDL セルサーチ設計は、5G NR 同期信号ブロック (SSB) を検出して復調します。プライマリ同期信号とセカンダリ同期信号は、ユーザー機器 (UE) が gNodeB (gNB) 基地局からセル ID とフレームタイミングを取得するために使用されます。

図 3 は、この設計アーキテクチャの概要をブロック線図で示したものです。PSS ブロックが最も強いプライマリ同期信号 (PSS) を返す前に、デジタル ダウンコンバーター (DDC) ブロックが周波数オフセットを補正します。最も強い PSS を使用して、直交周波数分割多重 (OFDM) の復調を実行します。この最終段階では、適切なリソース要素内でセカンダリ同期信号 (SSS) を検索します。

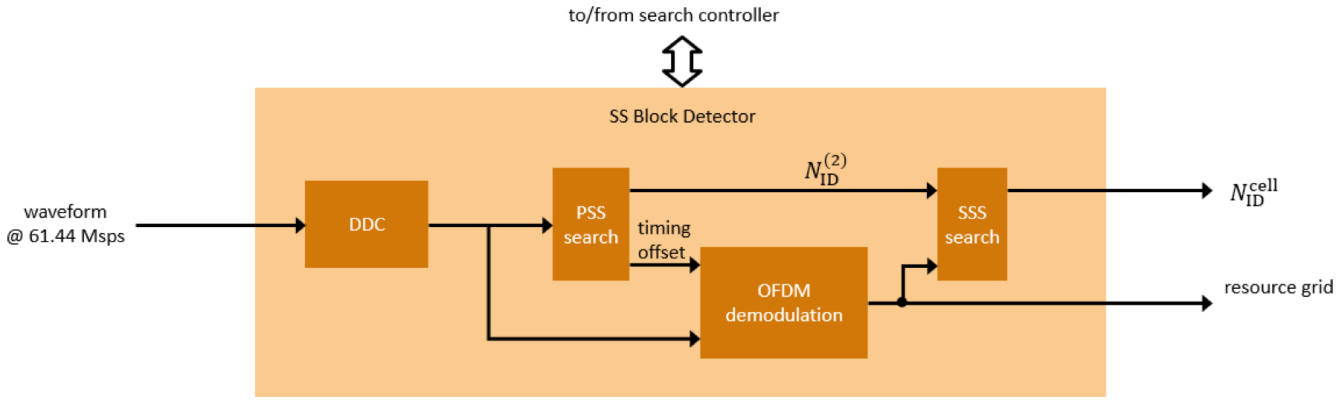

図 3. NR HDL セルサーチ ハードウェア アーキテクチャの概要を示すブロック線図。

これらのアルゴリズムをストリーミング処理を行うハードウェアに適用する際には、いくつかの検討事項やトレードオフがあります。そのうちの一部を、以下のワークフローの説明で取り上げます。

ワークフローの概要

MATLAB アルゴリズムから FPGA への展開に至るまでには、多くのパスが存在します。図 4 に示すトップダウンで調整するアプローチにより、チームは無線設計を確定的かつアジャイルな方法で FPGA に展開することができます。

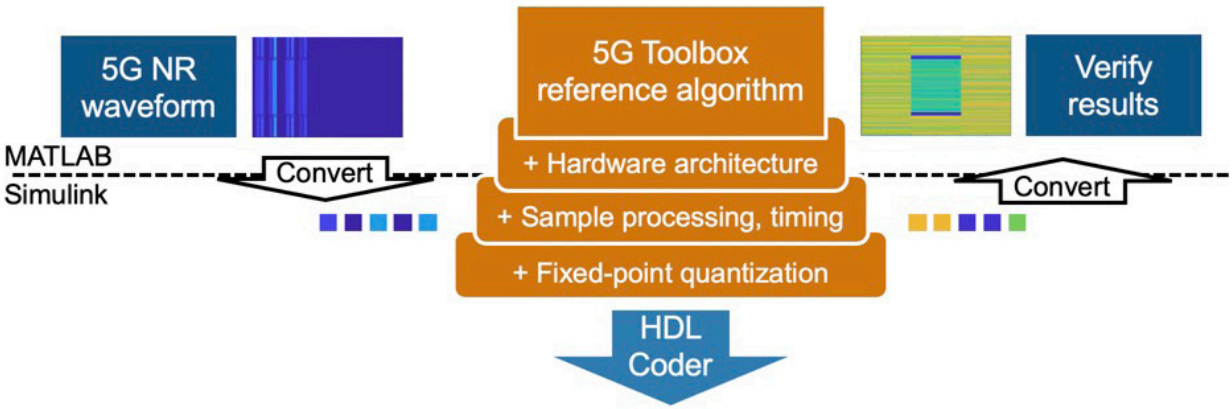

図 4. 無線アルゴリズムから FPGA への展開に至る、トップダウンの調整ワークフロー。

これらの調整手順は、ブロック線図に示された順序に従う必要はありません。調整の順序は、用途やユーザーの好みによって異なります。この手法では、MATLAB との接続を維持しながら、Simulink に徐々に移行していくことが可能です。Simulink には時間の概念が備わっているため、サンプルベースのストリーム処理を追加することは、Simulink で調整を行うのに適しています。これらの各調整手順は、前の段階または元のアルゴリズムの結果と照合して検証する必要があります。

図 4 にある青色のボックスは、テストベンチの要素である入力波形の生成と結果の検証を表しています。これには、ワークフローの段階に応じて複数の層があります。また、設計に調整を加えるにつれて、テストベンチは前の段階を比較の基準として使用するようになるため、前の段階がテストベンチの一部になります。下流工程で検証するための確認済みバージョンを得るためには、バグが発生した場合にそれらを特定して修正することができるように、各段階で十分な検証を行うことが重要です。

リファレンス アルゴリズム

無線通信の設計は、多くの場合、MATLAB でのアルゴリズム開発とテストから始まります。この MATLAB アルゴリズムは、残りのプロセスを進める上での "ゴールデン リファレンス" になります。従来のワークフローでは、アルゴリズムは機能仕様書のソースとして使用されます。このワークフローでは、アルゴリズムはさらなる調整に向けた開始点であると同時に、下流の調整を検証するための実行可能なモデルでもあります。

5G NR 無線規格は奥が深く複雑ですが、5G Toolbox にあらかじめ用意された、規格に準拠した関数を使用することで、アルゴリズム開発プロセスを高速化することができます。これらのアルゴリズムと Communications Toolbox™ の無線関数を組み合わせることで、確認済みの 5G 通信モジュールを素早く構築することができ、独自のアルゴリズムの開発に専念することができます。

NR HDL セルサーチ設計では、5G Toolbox に同梱されている *NR 同期手順*の例の PSS 検索、OFDM 復調、SSS 検索の手順を使用します。

ハードウェア アーキテクチャ

実装に向けたアルゴリズムの構築は、分割から始まります。5G Toolbox の *NR 同期手順*の例では、波形生成、チャネル伝播、受信機、および結果の検証向けの MATLAB コードが 1 つのスクリプトに組み合わされています。図 4 で示すように受信機を実装するには、受信機の機能を分割し、入力と出力を定義する必要があります。

Wireless HDL Toolbox リファレンス アプリケーションには、MATLAB バージョンの分割された受信機である、*NR HDL セルサーチ MATLAB リファレンス*が含まれます。このバージョンでは、高速信号処理タスクを実行する受信機機能が独自の MATLAB 関数に分割され、図 5 のオレンジ色のボックスで示すように、入力と出力を備えたハードウェアに展開されます。図 5 の青色のボックスが、ハードウェアのテストベンチになります。セルサーチの制御機能は、システムオンチップ (SoC) デバイスのプロセッサをターゲットに、ソフトウェアとして実装することができます。このリファレンス アプリケーションでは、この機能がテストベンチの一部として MATLAB に実装されています。このような種類のハードウェアとソフトウェアの分割は、ソフトウェアがハードウェアにおける高速処理を制御するもので、無線アプリケーションでは一般的なアプローチです。

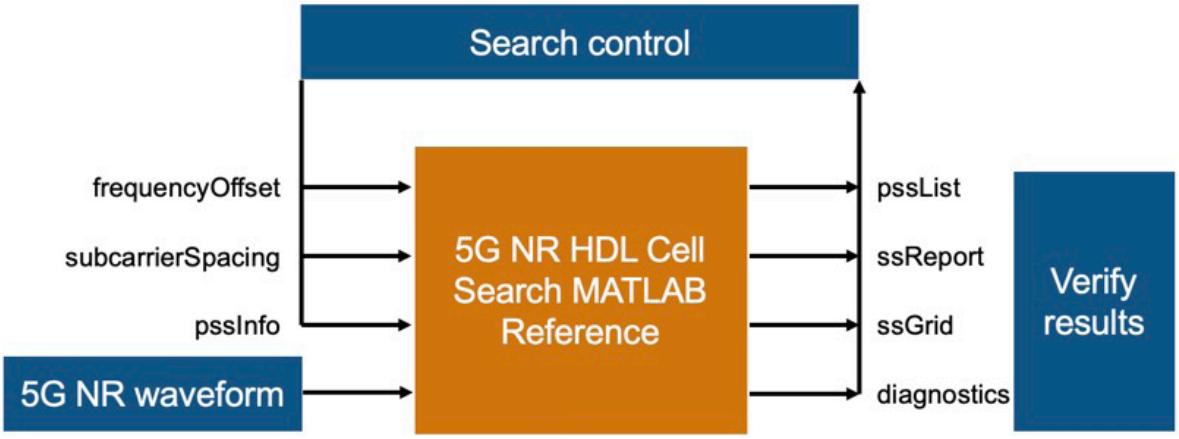

図 5. ソフトウェアおよびテストベンチからのハードウェア機能の分割。

ハードウェアは、まずセルサーチのコントローラから渡される周波数オフセットを修正し、サンプリングレートを下げるように構築されています。これにより、ソフトウェアは粗周波数オフセットの範囲全体でセルサーチすることができます。ハードウェアは PSS 相関を実行し、残留する細かい周波数オフセットを測定し、その結果を最も強いピークを決定するセルサーチの制御に戻します。この情報は、OFDM 復調と SSS 検出が実行される細かい周波数オフセット値とともにハードウェアに戻されます。

これまでのすべての処理は、波形全体または "フレームベース" で処理を行ってきました。ハードウェアは、信号が経時的に連続して流れる実機同様に実装されることを想定して構築されています。次の手順では、サンプルのストリームで動作するようにアルゴリズムを調整します。

サンプルベースの処理とタイミング

アルゴリズムを調整してストリーミング サンプルデータを処理するには、いくつか考え方を大きく変える必要があります。アルゴリズムが機能する方法を変えるだけでなく、データストリームを管理することも必要です。大幅な変更が必要ですが、ハードウェアのターゲット化に不可欠なものです。また、この処理形態は、Simulink にサンプル時間の概念が備わっているため、一般的に Simulink を使用して簡単に設計することができます。

NR HDL セルサーチ設計では、PSS の検出に既知のシーケンスと入力信号との相関が必要です。純粋な数学的アルゴリズムでは、相関は単純です。一方の信号をもう一方の信号にスライドして、そのサンプルの積を合計し、その合計の中で十分なピークを特定します。

PSS 検出の場合、3 つのセル ID PSS のそれぞれが、入力信号と互いに相関している必要があります。PSS はチップ内の RAM に格納されている固定値であり、入力信号はストリーミングで入力されるため、片方の信号をもう片方の信号にスライドするのが自然です。ただし、相互相関の計算を実行するためには、入力ストリームのウィンドウをバッファリングする必要があります。これはすべて、図 6 に示す PSS 相関器の Discrete FIR Filter HDL Optimized ブロックで処理されます。これらのブロックには、シリアルおよびパラレルアーキテクチャの選択肢があり、パラメーター設定に基づいてシミュレーション用のレイテンシが自動的に決まります。

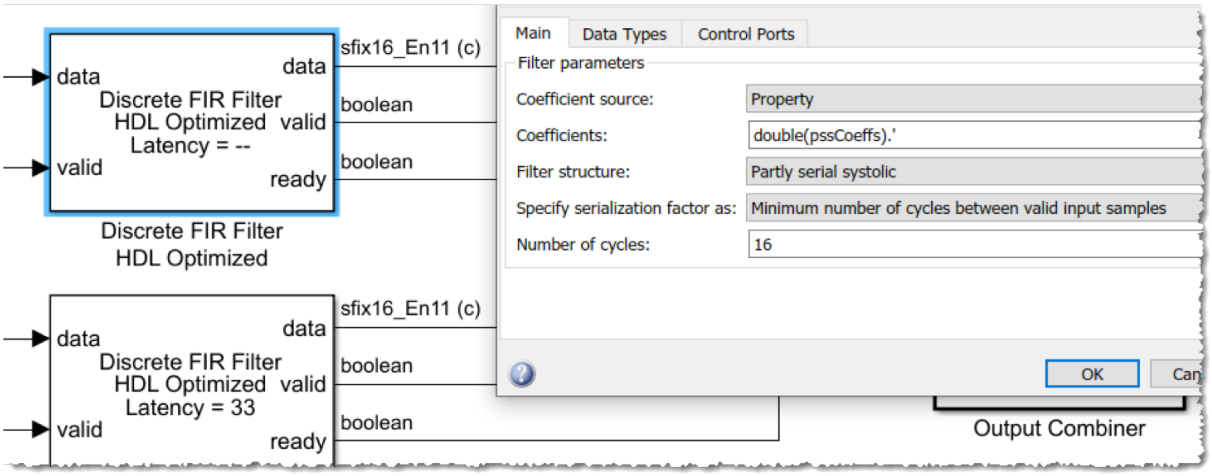

図 6. Discrete FIR Filter HDL Optimized ブロックの設定。

ハードウェアでは、3 つの PSS 相関を並列で実行し、ピークの検出後、その結果を制御ソフトウェアに渡します。ピークを特定するために、各相関結果が平均信号強度に比例したしきい値よりも高いかどうかをチェックします。しきい値を超えると、このローカルの時間枠で真のピーク値を検索することができます。図 7 は、このアーキテクチャを示したもので、PSS 相関結果および平均信号強度がプロットされています。

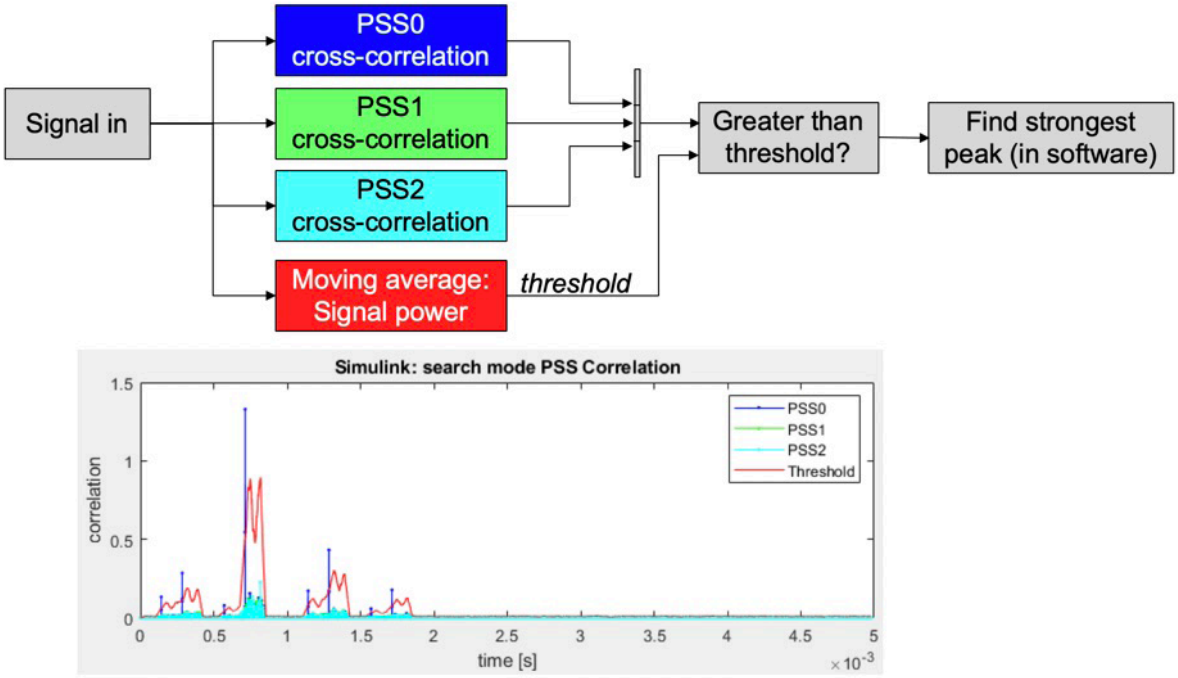

図 7. PSS 検出のフローチャート (上)、PSS の相互相関と移動平均しきい値の結果のプロット (下)。

連続したデータストリームの平均信号強度を計算するには、MATLAB でのグローバル平均の計算とは異なるアプローチが必要です。これにはスライディング ウィンドウ手法も必要であり、図 7 のしきい値プロットが時間の経過とともに変化するのはそのためです。平均信号強度は、相関器と同じ時間枠で測定されます。この手法では、小さいメモリで必要な検出性能を実現できます。この設計上のトレードオフには、アルゴリズム開発者とハードウェアエンジニアの密接な共同作業が必要です。

図 8 に移動平均アルゴリズムを示します。"Z^{-1}" などの Delay ブロックは、ハードウェアでレジスタにマッピングされて値を格納します。"Z^{-128}" ブロックは、受信するサンプルと並列で FIFO バッファーを挿入し、前の 129 クロックサイクルのサンプルを差し引くことで、常に移動平均向けの最新の 128 サンプルを継続的に格納します。図 8 の青色のレジスタは、パイプライン ステージとして用意されています。演算はハードウェアで時間を消費するため、分割してステージ間の演算量を減らすことでクロック周期を縮小し、最大クロック周波数を上げることができます。HDL Coder を使用してパイプライン ステージを自動的に挿入することもできますが、適切な数の遅延を設計でモデル化し、HDL を生成する前に、シミュレーションで並列パスが正しく同期していることを確認する手順をお勧めします。図 8 では、並列の validIn 信号の遅延が一致していることに注目してください。イネーブル信号を遅延させると、適切なクロックサイクルで到達します。HDL Coder の適応パイプラインなどの最適化機能を利用してパイプラインの遅延を挿入する場合は、[最適化] 設定で [遅延の均衡化] がオン (既定の設定) になっていることを確認してください。

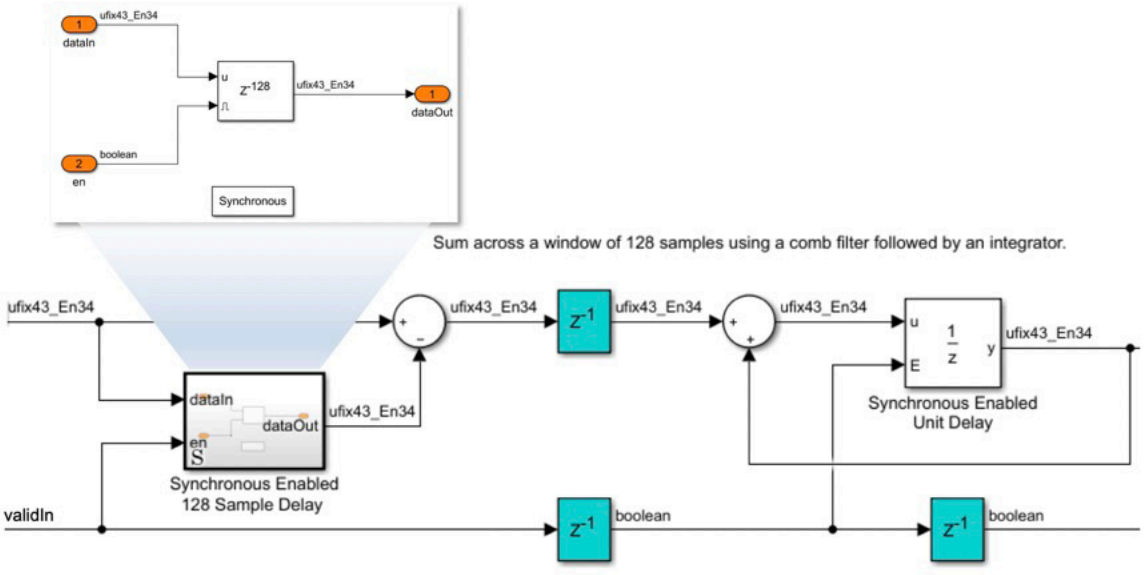

図 8. Simulink における移動平均アルゴリズムとハードウェア実装の詳細。

連続的なデータストリーミングでは、パケットなどのシーケンスの開始と終了、または中断されるタイミングをアルゴリズムで把握する必要があります。これには、ハードウェアでの制御信号が必要です。制御信号の最小単位は “valid” 信号で構成されますが、多くの場合、“enable”、“start”、“end” などが含まれます。NR HDL セルサーチ設計では、図 9 に示すように、OFDM 復調器は MATLAB Function ブロックを使用し、startIn 信号および validIn 信号に基づく有効な信号を生成します。

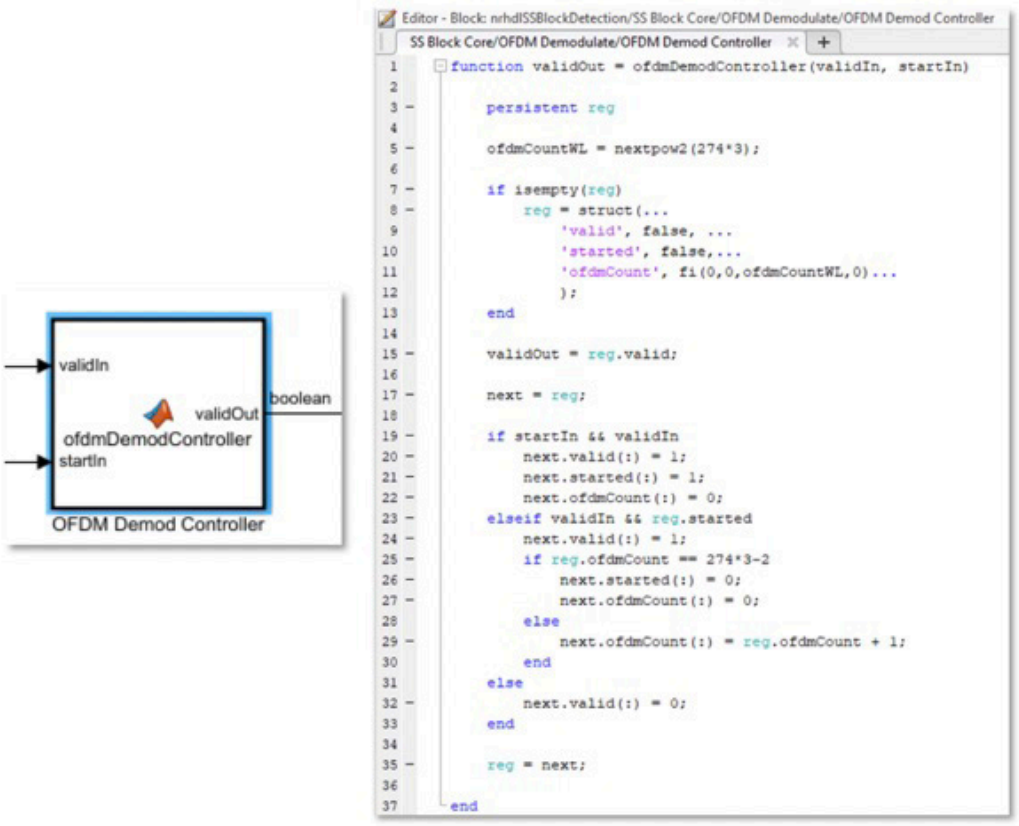

図 9. OFDM Demodulation Controller MATLAB Function ブロックと、対応する MATLAB コード。

前述のストリーミング アルゴリズムでは、メモリのシンプルな形式である FIFO バッファーを使用し、データストリームからウィンドウを一時的に保存します。非連続的に保存または使用されるデータには、アドレス指定可能な RAM または読み取り専用の LUT リソースを使用する必要があります。メモリ アーキテクチャは、ハードウェア実装の性能やリソースの使用状況に大きく影響する可能性があります。図 10 は、SSS 検出サブシステムにおいてメモリがどのように使われているかを示しています。MATLAB Function ブロックは状態を制御します。状態は、"アイドル"、OFDM 復調されたシンボルの RAM への書き込み、SSS シーケンスとの相関のための読み取りのいずれかになります。SSS 配列は固定値であるため、読み取り専用のメモリリソースである LUT に格納されます。

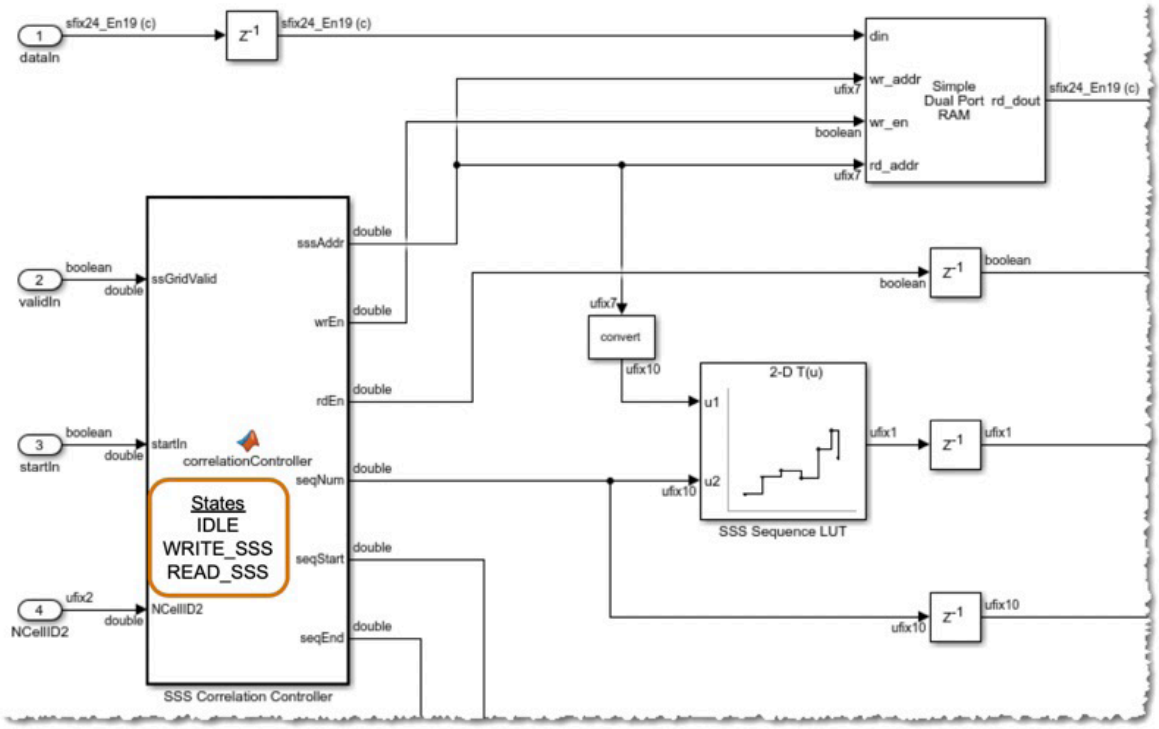

図 10. SSS 検出アルゴリズムへの入力用に RAM および LUT ハードウェアリソースを使用。

性能とリソース使用量の目標を達成するためには、メモリサイズとハードウェアデバイス機能を理解することが重要です。この場合、RAM はデュアルポートであるため、読み取り中に書き込みができます。アドレスラインは 7 ビットであるため、128 箇所にデータを格納でき、データは 48 ビット幅であるため 6144 ビットの RAM が必要になりますが、ほとんどの場合、デバイス上のブロック RAM にマッピングされます。

Simulink ハードウェア実装モデルをシミュレーションする準備ができると、Simulink は MATLAB ワークスペースと連携します。NR HDL セルサーチ設計の最上位では、From Workspace ブロックを使用して、フレームベースの信号データを入力用の MATLAB ワークスペース構造から取り込み、サンプル単位でハードウェア実装にストリーミング入力します。出力は To Workspace ブロックに収集され、MATLAB で使用するために出力用の MATLAB ワークスペース構造に書き込まれます。図 11 は、Simulink ブロック線図とともに、パラメーターを設定し、セルサーチの Simulink モデルを実行し、結果を読み取り、復調用の Simulink モデルを実行する MATLAB コードを示したものです。図 11 には示されていませんが、MATLAB リファレンス アルゴリズムも呼び出して Simulink ストリーミング処理の結果を検証しています。

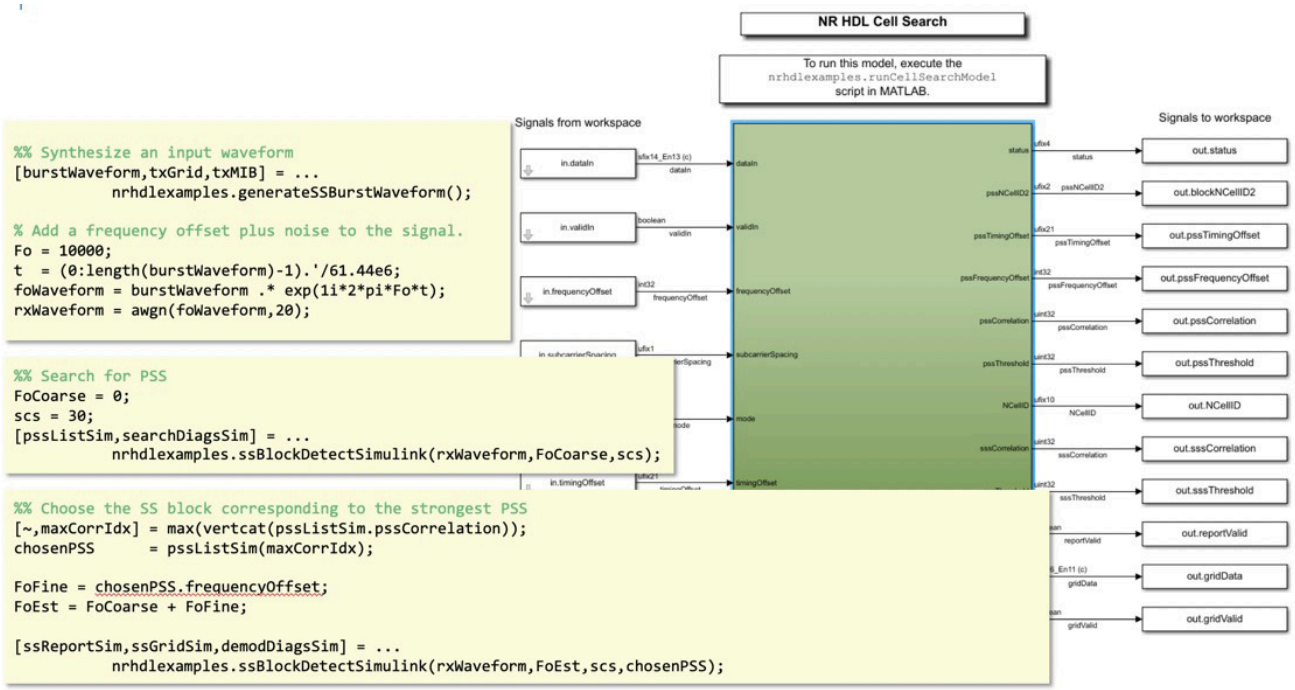

図 11. NR HDL セルサーチ リファレンス アプリケーション最上位の Simulink ブロック線図と、5G NR 準拠の入力用の波形を生成し、PSS 検索モードで MATLAB から Simulink を呼び出し、SSS 検出のために最も強い PSS を Simulink モデルに戻すための MATLAB コード。

実証済み IP ブロックの活用

ストリーミング アルゴリズムのハードウェア アーキテクチャを実装する場合、それらをゼロから設計するコストとリスクを負うよりも、あらかじめ用意されたブロックを再利用する方が実践的です。これらの IP ブロックは、外部ソースや社内の他のプロジェクトから得られる可能性があります。どのようなソースにも、使用するための 3 つの重要な原則があります。

1. **設計そのものを変更するのではなく、機能として備わったカスタマイズ性を利用する。**このカスタマイズ性は、一般的にパラメーターの形をとります。たとえば、図 6 に示す PSS 相関器で使用される Discrete FIR Filter HDL Optimized ブロックでは、MATLAB ワークスペースからの係数の再利用、HDL フィルターアーキテクチャのカスタマイズ、演算のシリアル化、その他のパラメーターの設定を行うことができます。パラメーターが設定され、設計が構築されると、ブロックは設定を反映したレイテンシを考慮して、このハードウェア実装用モデルをシミュレーションしますこのモデルは最適化され、下流向けの HDL として生成されます。
2. **設計への組み込みが簡単な IP ブロックを使用する。**図 12 に示す NR HDL セルサーチ設計における SS Block Core サブシステムの OFDM Demodulator ブロックは、この原則を説明するものです。このブロックは、さまざまな OFDM ベースの規格の復調をサポートしています。5G、LTE、ワイヤレス ローカル エリア ネットワーク (WLAN 802.11a/b/g/n/ac)、WiMAX (802.16m および 802.16e)、デジタルビデオ放送 (DVB)、デジタルオーディオ放送 (DAB) に対応しています。設計要件に応じて、巡回プレフィックス、OFDM シンボルガードバンド、FFT の実装に必要なパラメーターを素早く設定することができます。なお、入力における制御機能では、startIn 信号を使用して復調器をリセットするタイミングを特定し、valid 信号を駆動して OFDM 復調器にデータを渡します。

3. **必要な機能や性能をもたらすことが実証されている IP ブロックを選択する。**
これは当たり前のことのように感じられるかもしれませんが、この原則を守らないと、設計に精通していないため問題が発生した場合に対処するのが難しくなります。このため、*NR HDL セルサーチ設計*では、Xilinx Zynq ハードウェアをターゲットとして、無線信号を使用して現場でテストが行われています。

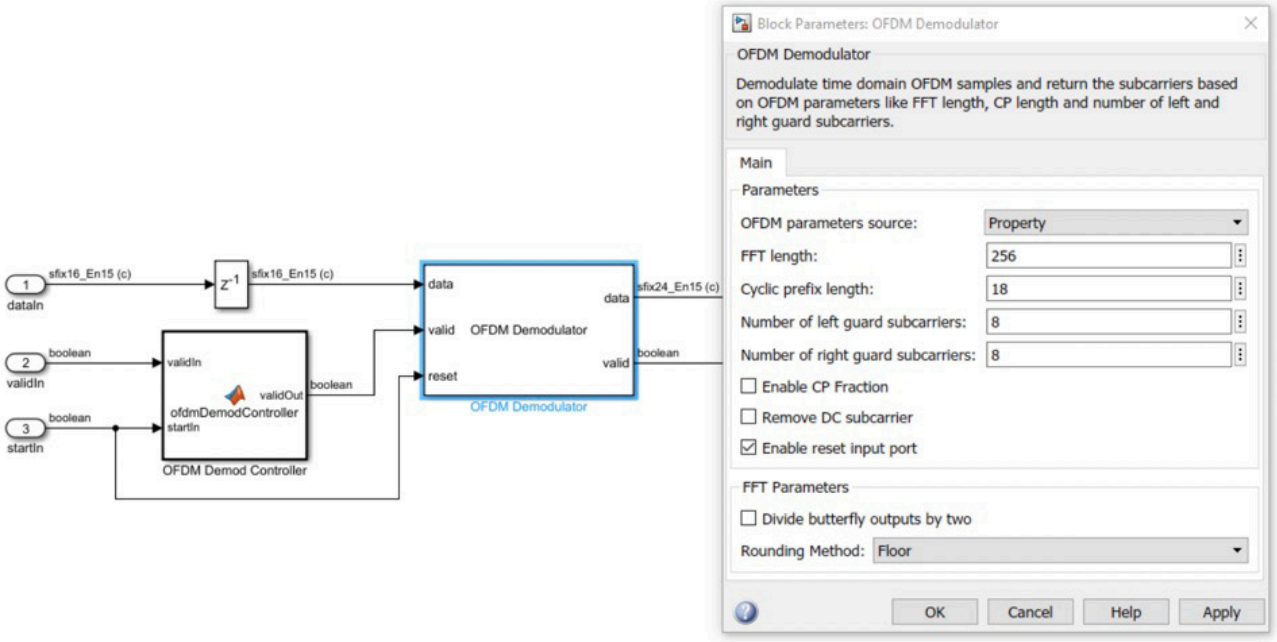

図 12. 設計のコンテキストに組み込む IP ブロック。

固定小数点の実装

ほとんどのデジタルハードウェア設計では、より効率的な実装のために固定小数点演算を使用します。精度と効率のトレードオフは困難なことがあり、フィードバックループや高ダイナミックレンジの処理がある場合は特に難しくなります。HDL Coder には、*ネイティブ浮動小数点*の実装があり、LTE の OFDM イコライゼーションに必要な除算演算に役立つ場合があります。これらの演算をデータ型変換ブロックで分離し、スケーラブルな精度を得るために単精度の浮動小数点演算を生成します。

5G NR セルサーチ設計をなどのほとんどの設計で、十分な精度の固定小数点を実装することが可能であり、Fixed-Point Designer を使用してプロセスを管理することができます。ただし、これは*固定小数点理論*の基礎、特に、異なる算術演算でどのように語長が伸びるかを理解するのに役立ちます。たとえば、乗算結果の語長は、入力した語長の合計になります。これを Simulink で観察した場合、図 13 で示すようになります。図の表記では、sfix は符号付き固定小数点を意味し、最初の数字は語長、2 番目の数字は小数部のビット数です。つまり、sfix16_En11 は、語長が 16 の符号付き固定小数点の数で、そのうち 11 ビットは小数部のビット数です。これら 2 つの 16 ビットの数値を掛け合わせると、32 ビットの結果が得られます。

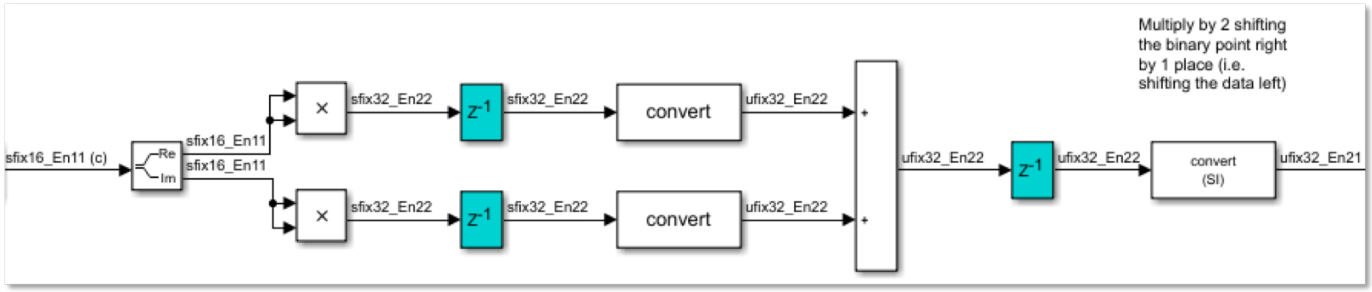

図 13. 算術演算を通した固定小数点語長の伝播。

乗算演算はハードウェアリソースの使用量が多いため、乗算器が固定小数点変換を行う際にフォーカスする主要な領域になりますほとんどの FPGA は、DSP スライスと呼ばれる効率的な乗算リソースを備えており、そのサイズは通常 18x18 または 25x18 です。乗算器の入力ビット幅が大きい場合、乗算演算が複数の DSP スライスに分割され、リソースをすぐに使い切ってしまう可能性があります。そこで、ここでも効率と精度のバランスを取ります。

FPGA ハードウェアのターゲット化

固定小数点設定が満足のいく精度になったら、HDL Coder でターゲット化を行うプロセスを開始することができます。まず、SS Block Detection サブシステムを右クリックし、[**HDL コード**]、[**HDL ワークフロー アドバイザー**]の順に選択します。このウィザードのようなインターフェイスによって、目的のデバイスをターゲットとする VHDL または Verilog HDL を手順に従って生成することができます。HDL Coder は、図 14 の HDL コード例に示すように、Simulink 設計に対応した階層名および信号名、すべてのビットレベルのポートマッピングを備えた、デバイスに依存しないコードを生成します。

```
-- -------------------------------------------------------------
--
-- Module: nrhdlSSBlockDetection_Magnitude_Squared_2
-- Source Path: nrhdlSSBlockDetection/SS Block Core/PSS Detection/Correlators/Magnitude Squared 2
-- Hierarchy Level: 5
--
-- -------------------------------------------------------------
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE IEEE.numeric_std.ALL;

ENTITY nrhdlSSBlockDetection_Magnitude_Squared_2 IS
  PORT( clk                               :   IN    std_logic;
        reset                             :   IN    std_logic;
        enb                               :   IN    std_logic;
        dataIn_re                         :   IN    std_logic_vector(15 DOWNTO 0);  -- sfix16_En11
        dataIn_im                         :   IN    std_logic_vector(15 DOWNTO 0);  -- sfix16_En11
        validIn                           :   IN    std_logic;
        dataOut                           :   OUT   std_logic_vector(31 DOWNTO 0);  -- ufix32_En24
        validOut                          :   OUT   std_logic
        );
END nrhdlSSBlockDetection_Magnitude_Squared_2;


ARCHITECTURE rtl OF nrhdlSSBlockDetection_Magnitude_Squared_2 IS

  -- Signals
  SIGNAL dataIn_re_signed                 : signed(15 DOWNTO 0);  -- sfix16_En11
  SIGNAL dataIn_im_signed                 : signed(15 DOWNTO 0);  -- sfix16_En11
  SIGNAL Delay3_out1_re                   : signed(15 DOWNTO 0);  -- sfix16_En11
  SIGNAL Delay3_out1_im                   : signed(15 DOWNTO 0);  -- sfix16_En11
  SIGNAL Product4_out1                    : signed(31 DOWNTO 0);  -- sfix32_En22
  SIGNAL Product5_out1                    : signed(31 DOWNTO 0);  -- sfix32_En22
```

図 14. 振幅二乗サブシステム用に生成された VHDL の抜粋。

HDL Coder は、FPGA 論理合成やターゲット化プロセスを推進することもできますが、これはもちろん、ベンダーやデバイスによって異なります。ターゲットにできるデバイスやボードは非常に多いため、このホワイトペーパーでは一般的なターゲット化のカテゴリを超えた詳細を網羅することはできません。

プロトタイプボードをターゲット化する。 Xilinx Zynq ベースのソフトウェア無線 (SDR) キットなどの既製のボードを使用する場合は、*ハードウェアサポートパッケージ*をダウンロードして、ハードウェアおよび ARM® プロセッサで実行するソフトウェアの両方のセットアップとターゲット化を簡略化することができます。*この例*は、この NR HDL セルサーチ設計を使用して、Xilinx Zynq ベースのボードをターゲット化するための手順を説明しています。

量産用のデバイスをターゲット化する。 一般的には、ハードウェアチームとの共同作業によってカスタムボードの定義を行います。この定義では、ボード上のデバイス、ボード上のペリフェラル、デバイスのピンをボードレベル I/O にマッピングする方法を指定します。HDL ワークフロー アドバイザーを実行する場合、手順 1.1 でこのボードをターゲットとして選択することができます。次に、設計の I/O のマッピングを、ソフトウェアと通信を行うためのレジスタまたはデバイスのピンのいずれかに指定します。

検証およびデバッグ

ワークフローの概要で説明したように、テストベンチは設計に合わせて発展していきます。ここでは、テストベンチの主な要素と、それらがどのように発展していくのかについて簡単に説明します。

スティミュラス

アルゴリズム開発の初期段階では、MATLAB ベースの機能を使用して波形を生成することができます。NR HDL セルサーチ設計では、テストベンチが 5G Toolbox の機能を使用して規格に準拠した波形を合成し、ノイズを含むチャネルモデルを通して入力します。スティミュラス波形をどれだけモデル化しても、現場のテストではいつでも信号に予期せぬ事態が発生します。そのため、*この LTE 信号の例*に示すように、可能な限りライブの無線波形をキャプチャして、デバッグが困難なハードウェアでの作業に取り掛かる前に MATLAB および Simulink 内でテストするのが良い方法です。

参照モデル

ハードウェアの詳細によって設計を調整し、サンプルベースのストリーム処理を実装し、固定小数点への量子化を行った際に、その結果を以前に検証されたアルゴリズムのバージョンと比較することができます。オリジナルのリファレンス アルゴリズムの結果と比較することもできますが、通常は、より類似したバージョンの設計と結果を比較する方がより簡単です。分かりやすい例として、一度ストリーム処理に変換すると、もう一度変換してフレームに戻すよりも、ストリーミング出力同士を比較する方が簡単になります。NR HDL セルサーチの例では、Simulink 実装モデルと前回調整した MATLAB ハードウェア アーキテクチャ モデルを比較しています。

評価とデバッグ

初期アルゴリズムの開発時には、波形、スコープ、スペクトラムプロットなどを調査して正確性を評価するのが一般的です。ただし、アルゴリズムに調整を加えた場合、検証では、追加された実装の調整がオリジナルのアルゴリズムと機能的に一致しているかどうかを測定します。異なる調整による結果は、一般的には正確に一致することはありません。浮動小数点バージョンと固定小数点バージョンの比較がその良い例です。そのため、あるアルゴリズムが各テストに合格とみなされるか、または不合格とみなされるか評価する方法を決定する必要があります。NR HDL セルサーチの例では、MATLAB モデルと Simulink モデルの結果を、相対的な平均二乗誤差を測定して比較しています。無線通信向けの他の一般的な手法には、ビットエラーレートがあります。合否の測定に関する合意は、プロジェクトレベルの意思決定です。

テストが失敗した際のデバッグは、常に、バグが発生した段階で行う方がより簡単です。後でバグを見つけようとすると、より詳細な調査が必要になり、バグの発生につながったと考えられる設計上の決定からも離れてしまうため、課題が大幅に増えてしまいます。設計が FPGA で実行されると、内部信号の可視性が制限されるため、複雑度の方程式に別の要因が加わります。

*カバレッジ*の測定は、HDL 生成や FPGA のプログラミングに移行する前に、設計の機能がどれだけテストされているかを評価する方法です。また、テストで同じ機能を何度も実行していないか、それによってシミュレーション時間を無駄にしていないかを評価するのにも役立ちます。カバレッジ解析からのフィードバックに基づいて、どのテストを実行するかを調整することができます。

下流工程における検証の負担を軽減

検証およびデバッグは、詳細度が高いほど時間や手間がかかる可能性があります。そのため、バグをできるだけ早く特定し、修正することが極めて重要です。レジスタ転送レベル (RTL) からデバイスに実装されるまでの間に、詳細度と機能が追加されるため、バグが発生する可能性が高まります。MATLAB および Simulink で実行してきた作業を活用して、これらの問題をより効率的に解決できるようになります。*HDL コシミュレーション*や *FPGA データキャプチャ*などの手法により、MATLAB および Simulink 内でデバッグを行うことができ、テストベンチモデルから*検証コンポーネント*を直接*生成*して、量産検証チームによるテストベンチ構築をサポートすることができます。

結果

NR HDL セルサーチ設計を、 Analog Devices® AD9361 RF カードが接続された Xilinx Zynq-7000 All Programmable SoC ZC706 Evaluation Kit に展開しました。周波数範囲をスキャンし、ライブ無線信号から、最も強いセル信号向けの PSS と SSS を検出しました。OFDM 復調された同期ブロック リソース グリッドを SoC から MATLAB に渡し、5G Toolbox を使用して復号化しました。

さらに重要なこととして、MATLAB アルゴリズムから開始し、ハードウェア実装の詳細に沿って調整し、同じ環境から自動的に生成して、このデバイスをターゲット化しました。この手法では、調整を重ねるごとに、MATLAB アルゴリズムと同じ結果が得られるかどうかについてハードウェアで簡単に検証することができました。シミュレーションには 5G NR 波形を使用し、ライブ無線信号を使用して PSS と SSS も検出しました。

1 人のエンジニアがこれらすべてを実現するスキルを有することはあまりありません。ここで示した共同作業環境は、これらのスキルを集約し、役割や部門を超えたエンジニア達が設計、シミュレーションをより迅速に反復できるようにします。システムレベルのコンテキストで、実際のスティミュラスを使用して早期にシミュレーションを行うことで、エンジニアは詳細なハードウェア設計を始める前に、機能や性能に関する問題を発見し、修正することができます。変更を素早く反映してシミュレーションを行い、更新されたコードをターゲットハードウェアに対して自動的に生成することができます。この機敏性により、エンジニアは、何千行もの VHDL を記述している場合よりも、製品の結果により大きな影響を与える高位の問題解決に注力することができます。

最終的な目標がプロトタイプであるか量産展開であるかにかかわらず、このワークフローにより、高品質な無線通信アルゴリズムを FPGA ハードウェアにターゲット化する時間を短縮することができます。

関連情報

NR HDL セルサーチ設計に関するビデオを視聴する (リファレンス アプリケーションは Wireless HDL Toolbox で利用可能): *5G NR HDL セルサーチ リファレンス アプリケーション*

Zynq SDR ハードウェア サポートパッケージをダウンロードする:
Communications Toolbox での Zynq SDR のサポート

リソースセンターを参照する: *MATLAB による無線通信システムの設計*

導入のためのサービスを探す

MathWorks では、アプリケーション エンジニアによる導入サポートから、詳細なライブ形式のトレーニングコース、カスタマイズしたコンサルティングサービスまで、このワークフローの効果的な活用を支援するさまざまなサービスを提供しています。

- *MATLAB と Simulink のトレーニング*
  - *MATLAB を使用した LTE および LTE Advanced 物理層システムの設計*
  - *FPGA 向け DSP*
  - *Simulink と Zynq によるソフトウェア無線*
- *MathWorks Consulting Services*